Commando's en functies

VERSIE 5.0

dBASE® voor Windows

Borland International, Inc., 100 Borland Way
P.O. Box 660001, Scotts Valley, CA 95067-0001

2E0R694

9495969798-98765432

W1

Inhoud

Inleiding

In *Commando's en functies* wordt de dBASE-taal beschreven zoals deze is geïmplementeerd in dBASE voor Windows. Dit handboek bevat gedetailleerde informatie over de commando's, functies, systeemgeheugenvariabelen, preprocessor-instructies en klassen.

Commando's en functies en *Programmeren* vormen samen het naslagwerk over de dBASE-taal en over het schrijven van dBASE-applicaties.

Raadpleeg dit handboek voor informatie over de werking van een commando of een functie, over de syntaxis die daarvoor is vereist en voor voorbeelden van het gebruik. Dezelfde informatie is ook beschikbaar in het helpsysteem.

Structuur van dit handboek

- Hoofdstuk 1, "Definitie van de taal": hierin worden de basisbegrippen beschreven, namelijk componenten van de dBASE-taal, elementen van opdrachten, gegevenstypen, uitdrukkingen en operatoren.
- Hoofdstuk 2, "Syntaxis": hierin wordt beschreven welke symbolen en conventies gelden voor de syntaxis, alsmede een aantal richtlijnen voor het interpreteren van de syntactische notatie.
- Hoofdstuk 3, "Commando's per categorie": hierin worden alle commando's, functies en andere taalelementen opgesomd, ingedeeld naar het type bewerking.
- Hoofdstuk 4, "Commando's en functies": hierin worden in alfabetische volgorde alle commando's en functies beschreven.
- Hoofdstuk 5, "Systeemgeheugenvariabelen": hierin worden in alfabetische volgorde de systeemgeheugenvariabelen beschreven voor het maken van printer- en omgevingsinstellingen.
- Hoofdstuk 6, "Preprocessor-instructies": hierin worden in alfabetische volgorde de preprocessor-instructies opgesomd voor het compileren van programma's.
- Hoofdstuk 7, "Klassen": hierin worden in alfabetische volgorde de standaard klassen van dBASE voor Windows opgesomd.

- Hoofdstuk 8, "Kenmerken": hierin worden in alfabetische volgorde de standaard klassekenmerken opgesomd.
- Appendix A, "Verschillen met dBASE IV 2.0": hierin wordt een overzicht gegeven van de verschillen met dBASE IV in de vorm van een opsomming van de nieuwe, gewijzigde en verdwenen commando's en functies.
- Appendix B, "Technische specificaties van dBASE voor Windows": hierin worden de technische specificaties van dBASE voor Windows beschreven.
- Appendix C, "Bestandsstructuren": hierin wordt de structuur van de tabel- en memobestanden beschreven van dBASE voor Windows.
- Appendix D, "Waarden voor INKEY() en READKEY()": hierin worden de waarden opgesomd die worden teruggegeven door de functies INKEY( ) en READKEY( ).
- Appendix E, "ASCII-tekentabel (codepagina 437)": hierin wordt een overzicht gegeven van de tekenset op codepagina 437.

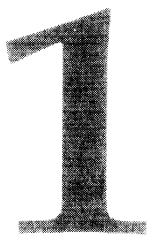
Typografische conventies

In *Commando's en functies* worden de verschillende elementen van de taal en de syntaxis elk met een eigen typografie aangegeven. Deze typografie wordt gebruikt om de elementen beter van elkaar te kunnen onderscheiden en het handboek beter leesbaar te maken.

Typografie	Gebruik	Voorbeelden
Cursief	Namen van variabelen en arrays, argumenten	Variabele *cNamen*, array *aKosten*, argument *<bestandsnaam>*
Kapitalen	Namen van commando's en functies, sleutelwoorden, DOS-bestanden en directory's	BROWSE, EOF(), INDEX ...TAG CUSTOMER.DBF
Beginkapitaal	Procedures, applicaties, tabelnamen, veldnamen, menu-opdrachten	Applicatie Rekeningen, veld Prijs
Kameelkapitalen	Kenmerken	OnLeftDblClk
Helvetica	Commandosyntaxis	FOUND([*alias*])
Futura smal	Namen van interface-onderdelen	Dialoogvenster **Bestand openen**
Helvetica cursief	Toetsen op het toetsenbord	Druk op *Ctrl-F10*
Typemachineletter	Codevoorbeelden	`USE Namen`

Zie Hoofdstuk 2 voor nadere informatie over de syntactische symbolen in commando's, functies en andere taalelementen.

Hoofdstuk



Definitie van de taal

De dBASE-taal is een gestructureerde programmeertaal die in de eerste plaats is bedoeld voor de ontwikkeling van database-applicaties. De taal bestaat uit standaardcommando's, -functies en -klassen waarmee u applicaties maakt om gegevens op te slaan, te beheren en te bewerken.

In dit hoofdstuk worden de standaardtaalelementen in dBASE beschreven, evenals de basiselementen die nodig zijn om *opdrachten* te maken die u in het commandovenster of in een programma kunt invoeren. Een opdracht is een complete instructie waarmee u dBASE een bepaalde taak kunt laten uitvoeren.

Taalelementen

De standaardtaalelementen in dBASE voor Windows zijn:

- Commando's
- Functies
- Systeemgeheugenvariabelen
- Preprocessor-instructies
- Klassen

Commando's

U gebruikt commando's om dBASE bepaalde taken te laten uitvoeren. Een opdracht begint vaak met een commando (opdrachten worden ook wel *commando-opdrachten* genoemd). Een opdracht kan andere taalelementen zoals functies bevatten, maar slechts één commando.

In de volgende opdracht worden bijvoorbeeld het commando ? en de functie DATE() gebruikt om de huidige datum op te vragen:

```
? DATE( )
```

De meeste opdrachten kennen opties (ook wel *schakeloptie* of *clausule* genoemd) die u gebruikt om de werking van een commando aan te passen aan een specifieke taak. Met het commando USE kunt u gewoon een tabel openen. U kunt dit commando echter ook in combinatie met de optie EXCLUSIVE gebruiken voor exclusief gebruik van die tabel.

```
USE Tabel              && Tabel openen met naam tabel.dbf
USE Tabel EXCLUSIVE    && Tabel openen met naam tabel.dbf voor exclusief gebruik
```

Een opdracht begint met een sleutelwoord of met een *impliciete* commando. Impliciete commando's zijn onder meer:

- Een toewijzing van een geheugenvariabele. Dit is impliciet het commando STORE:

```
x=45   && is hetzelfde als STORE 45 TO x
```

- Een recordnummer. Dit is impliciet het commando GOTO :

```
238     && is hetzelfde als GOTO 238
```

Functies

In dBASE worden twee soorten functies gebruikt: ingebouwde functies en door de gebruiker gedefinieerde functies. Het resultaat van een functie is een waarde. Een functie bestaat uit een sleutelwoord gevolgd door set haakjes (). Tussen de haakjes kunnen argumenten worden geplaatst die aan de functie worden doorgegeven om te worden verwerkt. In de onderstaande voorbeeldopdracht is het resultaat van de functie SQRT( ) de vierkantswortel van 36:

```
? SQRT(36)
```

Ingebouwde functies

De ingebouwde functies van dBASE worden meestal gebruikt voor reken-, tekst- of datumbewerkingen en voor logische bewerkingen en conversies. Het resultaat van sommige functies is een waarde die u vervolgens in een query kunt gebruiken. De functie RECNO( ) resulteert bijvoorbeeld in het nummer van het huidige record. Sommige functies, zoals RLOCK( ) en TRIM( ), voeren een bewerking uit en resulteren in een waarde.

Door de gebruiker gedefinieerde functies

Door de gebruiker gedefinieerde functies zijn functies die u maakt en vervolgens in een programma op dezelfde manier aanroept als een ingebouwde functie. U kunt zelf een functie definiëren met het commando FUNCTION, gevolgd door de naam van de functie, een optionele lijst met parameters en de reeks commando's die moeten worden uitgevoerd wanneer deze zelf gedefinieerde functie wordt aangeroepen. Zie het commando FUNCTION in Hoofdstuk 4 van deze handleiding en in Hoofdstuk 4 van *Programmeren* voor meer informatie.

Systeemgeheugenvariabelen

Systeemgeheugenvariabelen zijn geheugenvariabelen die in dBASE automatisch worden onderhouden. Deze variabelen worden meestal gebruikt om de omgevings- en printerinstellingen te definiëren en om de schermopmaak en de opmaak van de afgedrukte gegevens te bepalen.

Bij het starten van dBASE worden de systeemgeheugenvariabelen op de standaardinstellingen ingesteld. De standaardinstellingen van enkele systeemgeheugenvariabelen voor de printerinstellingen, zoals _pdriver en _porientation, zijn afhankelijk van de standaardinstellingen die in Windows voor de printer zijn gedefinieerd.

Om systeemgeheugenvariabelen te onderscheiden van andere geheugenvariabelen, beginnen systeemgeheugenvariabelen met een onderstrepingsteken (_). U kunt niet zelf geheugenvariabelen definiëren waarvan de naam met een onderstrepingsteken begint.

Voor het bereik van systeemgeheugenvariabelen gelden dezelfde regels als voor het bereik van geheugenvariabelen die u kunt definiëren. Zodra een programma met anders gedefinieerde systeemgeheugenvariabelen is beëindigd, worden de variabelen weer op de oorspronkelijke waarde ingesteld.

In tegenstelling tot door de gebruiker gedefinieerde geheugenvariabelen, kunnen systeemgeheugenvariabelen niet uit het geheugen worden verwijderd. U kunt alleen de waarde van deze variabelen wijzigen en opvragen.

Preprocessor-instructies

Preprocessor-instructies zijn opdrachten die u in dBASE-programma's kunt opnemen en waarmee u kunt aangeven hoe het programma moet worden gecompileerd. U kunt deze instructies gebruiken om tekst in een programma te vervangen, een voorwaardelijke compilatie te definiëren of om compileeropties op te geven.

Op deze manier kunt u bijvoorbeeld één basisprogrammacode onderhouden en verschillende delen van die code compileren, afhankelijk van het platform waarop het programma moet draaien.

Preprocessor-instructies beginnen met een hekje (#). Zie Hoofdstuk 7 in *Programmeren* voor meer informatie over het gebruik van preprocessor-instructies.

Klassen

Klassen zijn specificaties, of sjablonen, waarmee u objecten kunt maken. dBASE voor Windows kent vele standaardklassen waarmee u veel gebruikte Windows-objecten kunt maken zoals keuzerondjes, knoppen en invoervakken.

Als u een object maakt, kunt u een *kopie* maken van de klasse van het object. Als u dat doet, krijgt het object de vooraf voor die klasse gedefinieerde attributen. Deze attributen worden *kenmerken* genoemd. Wanneer u een object hebt gemaakt, kunt u de kenmerken aanpassen door aan elk kenmerk een waarde toe te kennen. Enkele objectkenmerken

zijn aan programmacode gekoppeld en worden gebruikt om objectkenmerken te wijzigen. Dergelijke kenmerken worden *methoden* genoemd.

Voor u als programmeur zijn objecten hetzelfde als een verzameling geheugenvariabelen, elk kenmerk en elke methode is een geheugenvariabele. Hierdoor is het voor u eenvoudig om dynamisch kenmerken aan objecten toe te voegen of objectkenmerken te wijzigen.

In het volgende voorbeeld ziet u hoe u een invoervak maakt uit de klasse Entryfield. Ook ziet u enkele kenmerken die u kunt definiëren:

```
DEFINE ENTRYFIELD KlantNaam OF KlantForm;
   PROPERTY;
      Width 20.00,;
      Height 2.00,;
      Top 7,;
      Left 18,;
      FontSize 10.00,;
      DataLink "Klant->KlantNaam"
      ...
```

Zoals het voorbeeld laat zien, kan de grootte van het invoervak eenvoudig worden gewijzigd door andere waarden toe te kennen aan de kenmerken Width en Height.

Standaardklassen

De meeste standaardklassen in dBASE voor Windows bestaan uit specificaties voor standaardobjecten die in Windows worden gebruikt (die ook wel *stuurelementen* worden genoemd) en die u maakt en in door de gebruiker gedefinieerde vensters plaatst die formulieren worden genoemd. Als u uw eigen objecten wilt maken, moet u eerst de bijbehorende klassen maken (zie de volgende paragraaf).

Eigen klassen

U kunt geheel nieuwe klassen maken of een standaardklasse wijzigen. Daartoe gebruikt u het commando CLASS...ENDCLASS. Een klasse die van een andere klasse is afgeleid, wordt een *subklasse* genoemd. De subklasse heeft kenmerken en methoden gemeen met de klasse waarvan deze is afgeleid. Vervolgens kunt u de attributen van de subklasse aanpassen door kenmerken toe te voegen, te verwijderen of te wijzigen.

In het volgende voorbeeld ziet u hoe u een nieuwe klasse maakt met de naam Afsluitknop die is afgeleid van de klasse Pushbutton:

```
CLASS Afsluitknop(f) OF PUSHBUTTON(f)
   this.text = "Afsluiten"
   this.FontName = "Arial"
   this.FontBold = .T.
   this.FontSize = 12
   this.OnClick = {;QUIT}
   ...
ENDCLASS
```

Als u een nieuwe klasse hebt gedefinieerd, kunt u op basis van die klasse objecten maken. In het volgende voorbeeld wordt een object gemaakt dat tot de klasse Afsluitknop behoort. Vervolgens wordt het object op het formulier KlantForm geplaatst.

```
DEFINE AFSLUITKNOP Afsluiten OF KlantForm;
   PROPERTY;
      Width 15.00,;
      Top 18.00,;
      Height 2.00,;
      ...
```

Zie Hoofdstuk 9 tot en met 12 in *Programmeren* voor meer informatie over klassen en objecten.

dBASE-opdrachten

U schrijft opdrachten om dBASE de gewenste bewerkingen uit te laten voeren. De meeste opdrachten beginnen met een commandosleutelwoord en bevatten opties en argumenten.

Syntaxis

De opbouw van een opdracht wordt de *syntaxis* genoemd. In het volgende voorbeeld ziet u de syntaxis van het commando REPLACE.

```
REPLACE <veld 1> WITH<uitdr 1> [ADDITIVE]
      [, <veld 2> with <uitdr 2> [ADDITIVE]...]
      [ALL | REST | NEXT <Nuitdr> | RECORD <Nuitdr>]
      [FOR <voorwaarde 1>]
      [WHILE <voorwaarde 2>]
      [REINDEX]
```

In de volgende tabel vindt u de elementen uit de syntaxis van het commando REPLACE.

Element in syntaxis	**Onderdeel van het commando REPLACE**
Commandosleutelwoord	REPLACE
Opties	ADDITIVE, ALL, REST, NEXT, RECORD, FOR, WHILE, REINDEX
Argumenten	*<veld 1>, <veld 2>, <uitdr 1>, <uitdr 2>, <Nuitdr>, <voorwaarde 1>, <voorwaarde 2>*

In deze handleiding worden speciale opmaakconventies gebruikt voor de syntaxis van een commando om te laten zien welke regels gelden voor de structuur van een opdracht. Zie Hoofdstuk 2 voor meer informatie over syntactische conventies. In de

volgende tabel vindt u enkele van de belangrijkste symbolen die in de syntaxis worden gebruikt.

Symbool	Omschrijving
[ ]	Optioneel onderdeel van de syntaxis.
< >	Vereist argument: *<Nuitdr>* geeft bijvoorbeeld aan dat u een numerieke uitdrukking moet opgeven.
|	Twee of meer opties die elkaar uitsluiten: ON | OFF.
...	Onderdeel dat een willekeurig aantal keren kan worden herhaald: [, *<veld 2> WITH <uitdr 2> [ADDITIVE]*...] geeft bijvoorbeeld aan dat u meer dan een veld kunt opgeven om te vervangen.

Opmerking In dBASE kunt u bij opdrachten de namen van commando's, functies en opties afkorten tot de eerste vier tekens. U kunt MODIFY COMMAND bijvoorbeeld afkorten tot MODI COMM. Dit is echter niet aan te bevelen omdat hierdoor de leesbaarheid afneemt en de opdracht niet sneller wordt uitgevoerd (opdrachten worden gecompileerd voordat ze worden uitgevoerd).

Aangezien u in dBASE klassen en kenmerken zelf kunt definiëren, kunt u de namen daarvan niet afkorten. De naam van de klasse Pushbutton kan bijvoorbeeld niet worden afgekort tot Push, omdat dBASE deze als een andere klasse beschouwt.

Argumenten

Een argument is een woord of een reeks van woorden tussen spitse haakjes die staan voor een deel van de syntaxis dat u moet opgeven, bijvoorbeeld *<bereik>*, *<gehvar>*, *<veld>*, *<bestandsnaam>* of *<uitdr>*. Als u een argument bij een opdracht opgeeft, moet u de spitse haakjes weglaten.

U kunt een specifieke waarde, maar ook een uitdrukking opgeven als argument. In het volgende voorbeeld ziet u een deel van de syntaxis van het commando REPLACE en vervolgens een voorbeeld van de waarden of uitdrukkingen (vetgedrukt) die u bij het commando kunt opgeven.

Syntaxis:

```
REPLACE <veld 1> WITH<uitdr 1> [ADDITIVE]
      [, <veld 2> WITH <uitdr 2> [ADDITIVE]...]
      [ALL | REST | NEXT <Nuitdr> | RECORD <Nuitdr>]
      [FOR <voorwaarden 1>]
      ...
```

Voorbeeld van opdracht:

```
REPLACE Postcode WITH "1101AA" FOR Postcode = "1191AA"
```

In de volgende paragrafen worden enkele van de meest gebruikte argumenten beschreven die u bij opdrachten kunt gebruiken.

Namen

Bij sommige commando's en functies moet u een naam opgeven, bijvoorbeeld een bestandsnaam, tabelnaam, veldnaam, naam van een geheugenvariabele enzovoorts.

Namen mogen maximaal 32 tekens lang zijn en kunnen bestaan uit elke willekeurige combinatie van letters, cijfers en het onderstrepingsteken. Hierop gelden de volgende uitzonderingen:

- Bestandsnamen voor DOS mogen niet langer dan 8 tekens zijn (plus een extensie van 3 tekens) en moeten voldoen aan de conventies voor bestandsnamen die in DOS van toepassing zijn.
- U kunt ook een naam van een tabel, veld of geheugenvariabele opgeven die een spatie bevat door de naam tussen dubbele punten te zetten, bijvoorbeeld *:nieuwe naam:*
- Namen van variabelen en veldnamen moeten met een letter beginnen.

In de paragrafen "Tabel- en bestandsnamen", "Tabel- en bestandsnamen", "Tabel- en bestandsnamen" en "Tabel- en bestandsnamen" worden de regels nader beschreven die gelden voor het opgeven van verschillende soorten namen.

Tabel- en bestandsnamen

Opmerking In eerdere versies van dBASE werden tabellen of .DBF-bestanden databasebestanden genoemd. In dBASE voor Windows wordt met een database een verzameling tabellen bedoeld. De term database verwijst ook naar SQL-databases waartoe u toegang verkrijgt via Borland SQL Link.

U kunt ook een bestandsnaam opgeven in de vorm van een vaste tekenreeks of in de vorm van een tekenuitdrukking die resulteert in een bestandsnaam. Ook kunt u de macro-operator & gebruiken om de naam van een bestand tijdens de uitvoering op te geven.

In dBASE voor Windows (evenals in dBASE IV) is het een goede gewoonte om bestandsnamen op te geven met behulp van indirecte verwijzingen. Bijvoorbeeld:

```
Mbestand = "Klanten"   && Naam bestaande tabel
USE (Mbestand)
```

Een volledige bestandsnaam bestaat uit het station en het pad van het bestand op de schijf, de eigenlijke naam van het bestand en de extensie:

```
[<station>] [<pad>] <naam> [ <extensie> ]
```

U kunt bijvoorbeeld de volgende naam opgeven:

```
C:\DBASEWIN\VOORBD\DIEREN.DBF
```

De volledige bestandsnaam, inclusief directory's, mag maximaal 79 tekens lang zijn. Het station, het pad en de extensie zijn optioneel. Alle beperkingen die gelden voor bestandsnamen komen overeen met de conventies onder DOS.

Als u de plaats van een bestand ten opzichte van de huidige directory weet, kunt u de plaats van dat bestand aangeven met de aanduidingen onder DOS voor de huidige (.) en *hoofd*directory *(..)*. Bijvoorbeeld:

```
USE ..\klanten
```

U kunt de jokertekens ? en * gebruiken in bestandsnamen om te verwijzen naar alle bestandsnamen die overeenkomen met een bepaald patroon. In de meeste gevallen

wordt in dBASE het dialoogvenster **Bestand openen** weergegeven waarin u de gewenste tabel kunt kiezen.

Aliassen

Een alias is een alternatieve naam voor een geopende tabel of het werkgebied van een geopende tabel. Als u in dBASE voor Windows een tabel opent, wordt een werkgebied gebruikt (in het geheugen) om de tabel en eventuele bijbehorende bestanden zoals een indexbestand te openen. Er kan steeds maar één werkgebied actief zijn.

U gebruikt een alias in een opdracht om een werkgebied aan te duiden en om toegang te krijgen tot een tabel die in dat werkgebied is geopend. U kunt het werkgebied aanduiden met een nummer (1 tot en met 255), met een letter (A tot en met J) of met een alias. In dBASE voor Windows kunnen maximaal 255 tabellen tegelijkertijd geopend zijn door aan elke geopende tabel een eigen, uniek werkgebied toe te kennen.

Als u een tabel opent met het commando USE, wordt in dBASE voor Windows de naam van de tabel automatisch beschouwd als de alias van die tabel (met de functie ALIAS( ) kunt u de alias van elke geopende tabel opvragen). Desgewenst kunt u ook uw eigen aliassen opgeven door een naam van maximaal 32 tekens in te voeren. Bijvoorbeeld:

```
USE Klanten ALIAS AlleKlant
```

Net zoals een bestandsnaam kunt u een alias opgeven als een vaste tekenreeks of in de vorm van een uitdrukking. Als een vaste tekenreeks kan worden verward met een andere naam of een commando-optie, moet u een werkgebieduitdrukking gebruiken. Het resultaat van een werkgebieduitdrukking is de aanduiding van een werkgebied met de tabel waartoe u toegang wilt krijgen. Om te voorkomen dat de naam van een geheugenvariabele met een werkgebied wordt verward met een vaste tekenreeks, plaatst u de naam van de variabele tussen aanhalingstekens zoals u in het volgende voorbeeld kunt zien.

```
cAliasNaam = "AlleKlant"
SELECT(cAliasNaam)
```

Databasenamen

In dBASE voor Windows wordt gebruik gemaakt van databasenamen of aliassen (toegekend met het IDAPI-configuratieprogramma IDAPICFG.EXE) om een station en directory op te geven van waaruit toegang kan worden verkregen tot tabellen en bestanden die bij dezelfde applicatie horen (zie Appendix B in *Aan de slag* voor meer informatie over het gebruik van het hulpprogramma ODAPICFG.EXE).

U kunt ook databasenamen definiëren om de locatie op te geven van waaruit toegang moet worden verkregen tot tabellen op een database-server. Om toegang te kunnen krijgen tot database-servers moet u Borland SQL Link installeren.

In dBASE voor Windows kan steeds maar één database tegelijkertijd actief zijn (hoewel meerdere databases tegelijkertijd geopend kunnen zijn). Net zoals tabellen en werkgebieden, kunt u een databasenaam zowel in de vorm van een vaste tekenreeks als in de vorm van een uitdrukking opgeven. Om bijvoorbeeld de naam van een database die u wilt openen, op te geven via een geheugenvariabele, gaat u als volgt te werk:

```
cDatabaseNaam = "MktgSqlServer"
OPEN DATABASE (cDatabaseNaam)
```

Nadat u databases hebt geopend, kunt u één database activeren met het commando SET DATABASE.

Veldnamen

Een *veldnaam* is de naam van een veld, of informatie, in een record van een tabel. Het woord ANAAM is bijvoorbeeld de naam van een veld waarin de achternamen van klanten staan vermeld. In elk record van een tabel wordt in dat geval de achternaam van een klant in het veld ANAAM ingevuld.

Een volledige veldnaam bestaat uit een verwijzing naar een alias (de aanduiding van het werkgebied) en de veldnaam in de tabel met die alias. Als de veldnaam niet in de tabel in het actieve werkgebied voorkomt, moet u de veldnaam nader specificeren met behulp van een alias. Als u een veldnaam nader wilt specificeren, gebruikt u het aliassymbool (->) tussen de naam van de alias en de veldnaam, zoals u in het volgende voorbeeld kunt zien.

```
Klanten->Anaam
```

In het bovenstaande voorbeeld wordt met de uitdrukking het veld ANAAM in de tabel Klanten aangeduid.

Opmerking Als een veld dezelfde naam heeft als een geheugenvariabele, heeft de veldnaam prioriteit boven de geheugenvariabele. Om de geheugenvariabele te onderscheiden van een veldnaam, laat u de aanduiding van een geheugenvariabele met m-> beginnen, bijvoorbeeld m->Anaam.

Geheugenvariabelen

Geheugenvariabelen zijn benoemde locaties in het geheugen waar u gegevens opslaat: tekenreeksen, getallen, datums, logische waarden, codeblokken, objectverwijzingen of functie-aanwijzers. Aan elk van deze waarden kent u een naam toe zodat u deze later kunt opvragen.

U kunt deze waarden gebruiken om ingevoerde gegevens van een gebruiker op te slaan, berekeningen uit te voeren, vergelijkingen te maken of waarden te definiëren die worden gebruikt als argument bij andere commando's en functies in dBASE. U maakt een geheugenvariabele door eenvoudigweg een waarde toe te kennen of door het commando STORE te gebruiken. De opdrachten in het volgende voorbeeld hebben hetzelfde effect.

```
cProdukt = "dBASE voor Windows"
STORE "dBASE voor Windows" TO cProdukt
```

Het bereik van geheugenvariabelen is gewoonlijk beperkt tot het dBASE-programma of de dBASE-procedure waarin deze zijn gedefinieerd. Als u een bereik expliciet wilt definiëren, gebruikt u de commando's PUBLIC, PRIVATE, LOCAL of STATIC. Zie de beschrijving in Hoofdstuk 4 van deze handleiding en in Hoofdstuk 5 van *Programmeren* voor meer informatie.

Array's

Met dBASE voor Windows kunt u ook een speciale set geheugenvariabelen maken, een zogenaamde array. Een array is een *n-dimensionale tabel* met waarden die in het geheugen zijn opgeslagen. Elk onderdeel van de array wordt een element genoemd. Elk element in een array kan worden gebruikt als een geheugenvariabele en kan ook in een uitdrukking worden opgenomen.

U maakt een arrray met het commando DECLARE. U kunt ook een array in de vorm van een object maken. Als u een array in een uitdrukking gebruikt, gelden dezelfde regels als voor geheugenvariabelen. U kunt een element van een array in elke willekeurige uitdrukking gebruiken waarin ook een geheugenvariabele mogelijk is. Bovendien kunt u in een element van een array en een geheugenvariabele hetzelfde soort gegevens opslaan.

Jokertekens

Als u in een argument een veld, bestand, index of een zoekreeks opgeeft, kunt u jokertekens (? en *) gebruiken. U gebruikt bijvoorbeeld het volgende commando om alle velden te selecteren waarvan de naam begint met de letter *c:*

```
SET FIELDS TO ALL LIKE c*
```

Met het jokerteken * geeft u aan dat op de positie van het jokerteken nul of meer tekens mogelijk zijn, met het jokerteken ? geeft u aan dat in de tekenreeks op de positie van het jokerteken elk teken mogelijk is. Het gebruik van jokertekens in dBASE verschilt in zoverre van DOS, dat in dBASE het jokerteken overal en in elke volgorde in de tekenreeks kan worden gebruikt:

```
SET FIELDS TO ALL LIKE *naam?
```

Opties voor het argument bereik

In commando's waarmee records in een tabel worden verwerkt, kunt u een bereiksleutelwoord gebruiken om het aantal records op te geven dat moet worden verwerkt. Alle commando's met de optie <*bereik*> hebben een standaardbereik, meestal ALL of NEXT 1. In de volgende tabel vindt u de sleutelwoorden die u voor het argument bereik kunt gebruiken.

Sleutelwoord	**Bereik**
ALL	Alle records in de tabel , te beginnen met het eerste
REST	Alle records vanaf het huidige record tot en met het laatste record in de tabel
NEXT <*Nuitdr*>	<*Nuitdr*> records vanaf het huidige record
RECORD <*Nuitdr*>	Het record aangeduid met <*Nuitdr*>

Bij veel commando's waarmee u records verwerkt, kunnen bovendien met FOR en WHILE voorwaarden worden opgegeven waarmee de te verwerken records nader worden gespecificeerd. Records worden in een logische volgorde verwerkt of, als een hoofdindex wordt gebruikt, in de volgorde die is aangegeven in die hoofdindex. Verder kunnen records nader worden gespecificeerd met een filtervoorwaarde of een query-bestand.

Uitdrukkingen

U gebruikt uitdrukkingen om waarden voor argumenten op te geven in een commando-opdracht. Een eenvoudige uitdrukking kan bestaan uit één enkel gegeven zoals een veldnaam, een constante of de naam van een geheugenvariabele. Bij het impliciete commando STORE is bijvoorbeeld een uitdrukking als argument vereist:

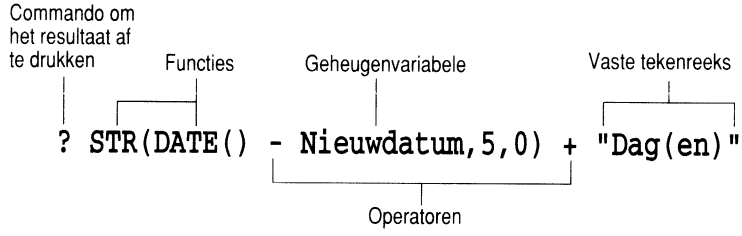
```
Naam = "Peter"
```

In dit voorbeeld is "Peter" een tekenuitdrukking.

Complexere uitdrukkkingen worden gevormd door een combinatie van verschillende gegevens en operatoren. U kunt in uitdrukkingen een of meer van de volgende elementen gebruiken:

- Veldnamen
- Geheugenvariabelen, inclusief systeemgeheugenvariabelen en array-elementen
- Constanten
- Functies
- Operatoren

In de volgende afbeelding ziet u een voorbeeld van een complexe uitdrukking, STR(DATE( ) - Nieuwdatum, 5, 0) + " Dag(en)", in een opdracht.



Operatoren (die verderop worden beschreven) verbinden elk van deze elementen zodat dBASE de uitdrukking als één geheel kan evalueren.

Als u velden, geheugenvariabelen, constanten en het resultaat van een functie in een uitdrukking combineert, moeten deze hetzelfde gegevenstype hebben (met uitzondering van datums, deze kunnen bij een getal worden opgeteld). U kunt zo nodig functies gebruiken om alle elementen tot hetzelfde gegevenstype te converteren, meestal zal dat een teken zijn. In het voorbeeld in de bovenstaande afbeelding wordt de functie STR( ) gebruikt om een getal in een teken om te zetten voordat dit en een ander teken worden aaneengeschakeld.

In de volgende paragrafen worden alle gegevenstypen en operatoren beschreven die u in dBASE voor Windows kunt gebruiken om uitdrukkingen te maken.

Gegevenstypen

In dBASE worden gegevens onderscheiden op basis van het type. Aan de basisgegevens (velden, geheugenvariabelen, constanten en functies) worden, afhankelijk van de wijze waarop de gegevens zijn gemaakt, gegevenstypen toegewezen. Het type uitdrukking dat u bijvoorbeeld toewijst aan een geheugenvariabele, bepaalt het gegevenstype van die variabele net zoals het resultaat van een functie het gegevenstype van die functie bepaalt.

Opmerking U bepaalt het gegevenstype met de functie TYPE( ).

In de volgende tabel vindt u de gegevenstypen die worden ondersteund door dBASE voor Windows, evenals het resultaat van de functie TYPE( ) bij elk gegevenstype.

Gegevenstype	Resultaat TYPE( )
Teken	C
Datum	D
Numeriek	N
Zwevend	F
Logisch	L
Memo	M
Binair	B
OLE	G
Bladwijzer	BM
Functieaanwijzer	FP
Codeblok	CB
Objectverwijzing	O

In de volgende paragrafen worden de ondersteunde gegevenstypen beschreven.

Zie Hoofdstuk 6 in *Programmeren* voor meer informatie over het werken met de verschillende gegevenstypen.

Teken

Het gegevenstype teken omvat elke combinatie van letters, cijfers, spaties en speciale tekens. In .DBF-tabellen kan een tekenveld maximaal 254 tekens lang zijn, een geheugenvariabele of uitdrukking van het gegevenstype teken kan uit maximaal 32.766 tekens bestaan.

Tekenconstanten of vaste tekenreeksen moeten tussen enkele of dubbele aanhalingstekens staan of tussen vierkante haakjes zoals u in de volgende voorbeelden kunt zien:

```
a = 'tekst'
b = "tekst"
c = [tekst]
```

Met de functie CHR( ) kunt u ook een decimale ASCII-waarde opslaan als een tekenreeks. Met de volgende opdrachten wordt de letter A in een geheugenvariabele opgeslagen. De geheugenvariabele is van het type teken.

```
STORE "A" TO cLetter
STORE CHR(65) TO cLetter
```

U kunt bovendien tekenreeksen in een vaste tekenreeks nesten door de geneste reeks tussen andere scheidingstekens te plaatsen. Met de volgende opdracht wordt bijvoorbeeld de daaropvolgende tekenreeks weergegeven:

```
? "Dit is een 'teken'reeks"

Dit is een 'teken'reeks
```

Opmerking In een vaste tekenreeks mag de null-waarde (ASCII 0), het EOF-teken (ASCII 26) of het LF-teken (ASCII 10) niet voorkomen.

Datum

Datumgegevens vormen een manier om een datum op te slaan in een notatie als *dd-mm-jj*. Intern wordt een datum echter altijd opgeslagen in de notatie *jjjjmmdd*.

De grootte van een datumvariabele of -veld is altijd 8 bytes. In dBASE voor Windows wordt een datum steeds gecontroleerd wanneer deze wordt ingevoerd of gewijzigd. Standaard wordt de datumnotatie ingesteld bij de optie **Internationaal** in het Configuratiescherm van Windows. U kunt deze notatie negeren met het commando SET DATE of met de instelling DATE in het bestand DBASEWIN.INI.

U kunt een vaste datumwaarde opgeven door de datum tussen accolades ({}) te plaatsen. U kunt ook een datum als een tekenreeks opgeven en deze met de functie CTOD( ) omzetten in het gegevenstype datum. Bijvoorbeeld {20-12-59} is hetzelfde als CTOD("20-12-59").

Als u een blanco datum wilt invoeren, typt u { }, { - - } of CTOD(" - - ").

U kunt een datum van een andere datum aftrekken. Het resultaat is het aantal dagen tussen de twee datums. U kunt ook een getal (een aantal dagen) bij een datum optellen of van een datum aftrekken. Het resultaat is weer een datum.

Numerieke gegevens

Met dit gegevenstype kunt u rekenkundige bewerkingen uitvoeren op gegevens (optellen, vermenigvuldigen en andere rekenkundige functies).

Wanneer u de structuur van een tabel opzet, kunt u numerieke velden definiëren als numeriek of zwevend. Verder kunt u bij zowel het numerieke als het zwevende type maximaal 20 cijfers met maximaal 18 decimale plaatsen opgeven.

Anders dan bij dBASE IV is er in dBASE voor Windows geen verschil in de manier waarop numerieke en zwevende gegevens worden verwerkt. Bij wiskundige bewerkingen, biedt dBASE voor Windows dezelfde nauwkeurigheid (19 cijfers) voor beide typen. In dBASE IV worden numerieke en zwevende gegevens verschillend verwerkt. Getallen van het numerieke type hebben een nauwkeurigheid van maximaal

20 cijfers en getallen van het zwevende type van 15 cijfers. In dBASE voor Windows wordt het zwevende type onderhouden om reden van compatibiliteit met dBASE IV.

Als u een getal toekent aan een geheugenvariabele, is het resultaat van de functie TYPE( ) altijd N. Alleen in geval van zwevende velden is het resultaat van de functie TYPE( ) F.

Als u met hele grote of hele kleine getallen werkt, worden deze in dBASE voor Windows in de exponentiële wetenschappelijke notatie getoond, bijvoorbeeld 6E + 23.

Logische gegevens

Logische velden en geheugenvariabelen worden opgeslagen als true (.T.) of false (.F.). Logische velden of variabelen beschouwen T, t, J of j als true en F, f, N of n als false. In opdrachten of uitdrukking gebruikt u bij logische waarden de punt als scheidingsteken (bijvoorbeeld STORE .T. to Mlogic).

Memogegevens

Memogegevens zijn alle gegevens die zijn opgeslagen in memovelden. Een memoveld is een veld met een variabele lengte dat wordt gebruikt voor grote hoeveelheden tekens (de grootte wordt alleen beperkt door de hoeveelheid geheugen). Memovelden worden meestal gebruikt voor teksten hoewel u deze velden ook kunt gebruiken voor de inhoud van OLE-velden, bitmaps, afbeeldingen, geluid en alle andere binaire gegevens (in dBASE voor Windows worden twee nieuwe gegevenstypen gebruikt voor respectievelijk OLE-gegevens en binaire gegevens).

dBASE slaat de eigenlijke memogegevens op in een memobestand met de extensie .DBT.

Binaire - en OLE-gegevens

dBASE voor Windows kent twee nieuwe gegevenstypen: binair en OLE. Deze gegevens worden opgeslagen in tabelvelden. De standaard binaire bestandstypen waaronder .BMP, .PCX en .WAV worden ondersteund.

Bladwijzer

Een bladwijzer is een verwijzing naar een bepaald record in een tabel. Deze lijkt op de recordaanwijzer in dBASE.

dBASE ondersteunt bladwijzers voornamelijk ten behoeve van Paradox- en SQL-tabellen waarin niet met recordnummers wordt gewerkt. Omdat in Paradox- en SQL-tabellen de records geen nummer hebben, kunt u bijvoorbeeld niet een commando als GOTO 23 gebruiken om naar record 23 te gaan of een commando als RLOCK("1, 2, 3") om records 1, 2 en 3 te vergrendelen.

Met bladwijzers kunt u records markeren zodat u deze kunt opvragen en bewerken. Markeer het huidige record door met de functie BOOKMARK( ) de bladwijzerwaarde in een geheugenvariabele op te slaan.

In het volgende voorbeeld ziet u hoe u bladwijzers gebruikt om records in een Paradox-tabel op te vragen:

```
USE Klanten.db                          && Paradox-tabel openen
GO TOP                                  && Naar eerste record
   bMark1= BOOKMARK()                   && Record markeren
   SKIP 2                               && 2 records verder gaan
   bMark2 = BOOKMARK()                  && Record markeren
   ? RLOCK("bMark1, bMark2", 1)         && Eerste en derde record vergrendelen in werkgebied 1
   GOTO bMark1                          && Terug naar eerste record
```

De bladwijzerwaarde is een speciaal gegevenstype dat niet kan worden afgedrukt. U kunt een bladwijzer niet in een query gebruiken. Als u een bladwijzerwaarde hebt toegewezen aan een geheugenvariabele en vervolgens de waarde van de variabele opvraagt, is het resultaat "BookMark". Dit ziet u in het volgende voorbeeld.

```
Record_a = BOOKMARK()
? Record_a                              && Het resultaat is "BookMark"
```

U kunt echter bladwijzerwaarden in een tabel vergelijken. Met operatoren als >, = en <, kunt u de waarden van de functie BOOKMARK( ) vergelijken om de relatieve positie van twee records in een tabel te bepalen. Dit ziet u in het volgende voorbeeld.

```
USE Klanten.db
GO TOP                              && Naar eerste record
   Record_a = BOOKMARK()
   GO BOTTOM                        && Naar laatste record
   Record_b = BOOKMARK()
   ? Record_a < Record_b            && Resultaat is .T. omdat Record_a voorafgaat aan Record_b
```

Bij commando's en functies als RLOCK( ) en EDIT, waarbij een recordaanwijzer als argument kan worden gebruikt, kan ook een bladwijzer als argument worden gebruikt.

Functie-aanwijzer

Zoals de naam al aangeeft is een functie-aanwijzer een verwijzing naar een functie of procedure. Met behulp van functie-aanwijzers kunt u eenvoudig functies en procedures definiëren en vervolgens, wanneer nodig, aanroepen.

Elk actiekenmerk van standaardobjecten, zoals bijvoorbeeld OnClick en OnOpen, is een variabele van het type functie-aanwijzer of codeblok (codeblokken worden in de volgende paragraaf beschreven). In het geval van een functie-aanwijzer kunt u een verwijzing toekennen aan een constante of variabele die verwijst naar een functie of procedure.

In het volgende voorbeeld ziet u hoe u bij het actiekenmerk OnClick een verwijzing kunt opgeven naar de procedure Ga_Volgend.

```
DEFINE PUSHBUTTON Volgendknop OF KlantForm
   Property;
      OnClick Ga_Volgend
...

PROCEDURE Ga_Volgend
```

```
   IF .NOT. EOF()
      SKIP
ELSE
   GO TOP
ENDIF
```

Codeblok

Een codeblok is een logische groepering van een of meer opdrachten of uitdrukkingen in dBASE. Net zoals de functie-aanwijzer vormt een codeblok een manier om opdrachten in dBASE uit te voeren. Een codeblok is vooral handig in combinatie met het actiekenmerk van een object.

In tegenstelling tot functie-aanwijzers zijn codeblokken echter vaste opdrachten die u toekent aan een actiekenmerk. Een codeblok is een gegevenstype dat programmacode van dBASE *bevat*. Net zoals andere gegevenstypen kunt u een codeblok aan een geheugenvariabele toewijzen of een codeblok als een parameter doorgeven.

dBASE voor Windows kent twee typen codeblokken: *opdrachten* en *uitdrukkingen*. Een codeblok met opdrachten kan een of meer dBASE-opdrachten bevatten terwijl een codeblok met uitdrukkingen alleen dBASE-uitdrukkingen kan bevatten.

Codeblok met opdrachten

Hieronder ziet u de syntaxis en regels die gelden voor het schrijven van een codeblok met opdrachten:

```
{[|<parameters>|]; <opdracht> [; <opdracht> ...]}
```

- De accolades, { }, zijn verplicht.
- Als u parameters doorgeeft, moet u deze onderling scheiden met | |.
- Elke opdracht moet worden voorafgegaan door een puntkomma (;).

In het volgende voorbeeld ziet u hoe u een codeblok met opdrachten toewijst aan het actiekenmerk OnClick:

```
DEFINE PUSHBUTTON Sluitknop OF KlantForm;
   Property;
         OnClick {;QUIT}
         ...
```

In het volgende voorbeeld ziet u hoe u een codeblok met opdrachten en parameters toewijst aan een geheugenvariabele:

```
TelOp = {|a,b|; ? a+b}
```

Ga als volgt te werk om het codeblok uit te voeren.

```
TelOp(10,10)   && Resultaat is 20
```

Codeblok met uitdrukkingen

Hieronder vindt u de syntaxis en regels die gelden voor het schrijven van een codeblok met uitdrukkingen:

```
{[|<parameters>|] <uitdrukking>}
```

- De accolades { } zijn verplicht.
- Als u parameters doorgeeft, moet u deze onderling scheiden met | |.

In het volgende voorbeeld ziet u hoe u een codeblok met uitdrukkingen toewijst aan het actiekenmerk Valid:

```
DEFINE ENTRYFIELD Salaris OF WerknemerForm;
   PROPERTY;
      Valid {Salaris >= 10000}
      ...
```

In het volgende voorbeeld ziet u hoe u een codeblok met uitdrukkingen en parameters toewijst aan een geheugenvariabele. Anders dan bij een codeblok met opdrachten, hoeft u nu geen puntkomma te gebruiken na de parameter.

```
TelOp = {|a,b| a+b}
```

Ga als volgt te werk om het codeblok uit te voeren:

```
TelOp(10,10)   && Resultaat is 20
```

Zie Hoofdstuk 4 in *Programmeren* voor meer informatie over de technieken voor het werken met codeblokken.

Objectverwijzing

Het gegevenstype objectverwijzing biedt de mogelijkheid verwijzingen te maken naar objecten in dBASE voor Windows en naar de variabelen (kenmerken en methoden) die deze objecten bevatten.

Als u een object maakt, wordt in dBASE voor Windows een *objectverwijzingsvariabele* gemaakt die naar dat object verwijst. In tegenstelling tot andere geheugenvariabelen die daadwerkelijk een waarde bevatten zoals een tekenreeks, bevatten objectverwijzingvariabelen een verwijzing naar het object en niet het object zelf.

Als u de ene objectverwijzingvariabele aan de andere toewijst, maakt u geen duplicaat van het object, u krijgt gewoon een andere verwijzing naar hetzelfde object.

In het volgende voorbeeld ziet u hoe u de operator NEW gebruikt om een formulierobject uit de klasse Form te maken (u kunt op twee manieren objecten maken: met de operator NEW of het commando DEFINE).

```
KlantForm = NEW Form( ) && KlantForm is de objectverwijzingvariabele
```

Zodra u het formulier hebt gemaakt, kunt u de kenmerken definiëren door te verwijzen naar de kenmerkvariabelen. Zoals al eerder is aangegeven, bestaat een object uit een verzameling variabelen, en elk kenmerk is een variabele.

In het volgende voorbeeld ziet u hoe u met de punt (.) als operator verwijst naar het kenmerk van een object. Het gebruik van de punt als operator wordt in de volgende paragraaf over operatoren beschreven.

```
KlantForm.Text = "Klantgegevens"
KlantForm.Width = 80.00
```

```
KlantForm.Top = 2.00
KlantForm.Left = 2.00
KlantForm.Height = 24.75
```

Zoals u in het voorbeeld ziet, verwijst u naar de kenmerken van een object door de objectverwijzingvariabele en de kenmerkvariabele op te geven.

Zie Hoofdstuk 9 tot en met 12 in *Programmeren* voor meer informatie over objecten en object-georiënteerd programmeren.

Operatoren

Een operator is een symbool of een sleutelwoord waarmee een bewerking wordt uitgevoerd op gegevens of op uitdrukkingen. In dBASE voor Windows worden de volgende operatoren gebruikt:

- Toewijzingsoperatoren
- Numerieke operatoren
- Relationele operatoren
- Logische operatoren
- Tekenreeksoperatoren
- Objectoperatoren
- Functie-operatoren
- Bereikoperatoren

Opmerking dBASE kent nog een andere operator, & (de macro-operator), die niet in deze paragraaf wordt beschreven. Zie Hoofdstuk 5 in *Programmeren* voor meer informatie over het gebruik van deze operator.

Bij alle operatoren zijn een of twee argumenten vereist die *operanden* worden genoemd. Operatoren waarbij één operand moet worden opgegeven, worden monadische operatoren genoemd. Operatoren waarbij twee operanden moeten worden opgegeven, zijn binaire operatoren.

Hieronder vindt u een voorbeeld van de monadische operator .NOT.:

```
lTrue = .T.
? .NOT. lTrue && Resultaat is .F.
```

Hieronder vindt u een voorbeeld van de binaire operator * :

```
? 7 * 7
```

Toewijzingsoperatoren

In dBASE wordt het is-gelijk-teken (=) gebruikt als operator om waarden van uitdrukkingen toe te wijzen aan geheugenvariabelen. U kunt deze operator bijvoorbeeld gebruiken om een tekenreeks op te slaan.

```
x = "Sla deze reeks op in x"
```

De toewijzingsoperator kan in uitdrukkingen van elk gegevenstype worden gebruikt.

Numerieke operatoren

Met numerieke operatoren worden berekeningen uitgevoerd en numerieke resultaten gegenereerd. Hieronder vindt u de numerieke operatoren die u in dBASE voor Windows kunt gebruiken.

Operator	Omschrijving
+	Optellen/monadisch positief
-	Aftrekken/monadisch negatief
*	Vermenigvuldingen
/	Delen
** of ^	Machtsverheffen
( )	Haakjes voor groeperen (om de volgorde waarin uitdrukkingen worden uitgerekend expliciet aan te geven)

Numerieke operatoren worden in numerieke uitdrukkingen gebruikt. U kunt numerieke operatoren ook gebruiken in uitdrukkingen van het type datum om twee datums van elkaar af te trekken, bij elkaar op te tellen en om een getal bij een datum op te tellen of van een datum af te trekken (waarbij het getal staat voor een bepaald aantal dagen).

Relationele operatoren

Relationele operatoren worden gebruikt om twee uitdrukkingen van hetzelfde gegevenstype te vergelijken. Het resultaat van een dergelijke vergelijking is de logische waarde true (.T.) of false (.F.). De operatoren kunnen worden gebruikt in numerieke, teken- en datumuitdrukkingen. Hieronder vindt u de relationele operatoren die in dBASE voor Windows worden ondersteund:

Operator	Omschrijving
<	Kleiner dan
>	Groter dan
=	Gelijk aan
==	Exact gelijk aan
<> of #	Niet gelijk aan
<= of =<	Kleiner dan of gelijk aan
>= of =>	Groter dan of gelijk aan
$	Vergelijking subreeks

Bij het vergelijken van tekenreeksen wordt onderscheid gemaakt tussen kleine en hoofdletters. Verder zijn de taalaansturing en de instelling van het commando SET EXACT belangrijk. Als het commando SET EXACT is ingesteld op OFF, worden twee

tekenreeksen als identiek beschouwd, indien alle tekens in de reeks links van het is-gelijk-teken (=) overeenkomen met de eerste tekens in de reeks rechts van het is-gelijk-teken. De rechterreeks mag korter maar niet langer zijn dan de linkerreeks.

Als het commando SET EXACT is ingesteld op ON, moeten de tekens in beide reeksen identiek zijn. Spaties aan het einde van een reeks worden echter genegeerd. In het volgende voorbeeld ziet u hoe het commando SET EXACT werkt:

```
SET EXACT OFF
? "abcde" = "abcd"       && Resultaat is .T.
   ? "abc" = ""          && Resultaat is .T.
   ? "abc" = "abcd"      && Resultaat is .F.
   SET EXACT ON
   ? "abcde" = "abcd"    && Resultaat is .F.
   ? "abc" = "abc "      && Resultaat is .T. (spaties aan het einde worden genegeerd)
```

Als de overeenkomst tussen twee reeksen exact moet zijn, ongeacht of het commando SET EXACT is ingesteld op ON of OFF, gebruikt u == in plaats van =.

Met de vergelijkingsoperator $ worden twee tekenreeksen vergeleken om te controleren of de linkerreeks voorkomt in de tekenreeks rechts van het is-gelijk-teken. Het resultaat van A$B is bijvoorbeeld true als tekenreeks A identiek is aan tekenreeks B of in tekenreeks B voorkomt. U kunt een subreeks opgeven in de vorm van een vaste tekenreeks, een tekengeheugenvariabele, een tekenveld of een memoveld.

Logische operatoren

Logische operatoren worden gebruikt om twee logische uitdrukkingen te vergelijken en als resultaat een logische true (.T.) of false (.F.) te verkrijgen. In dBASE voor Windows worden de volgende operatoren ondersteund:

Operator	Omschrijving
.AND.	Logisch AND; resultaat is true als beide uitdrukkingen waar zijn.
.OR.	Logisch OR; resultaat is true als een van beide uitdrukkingen waar is.
.NOT.	Logisch NOT; resultaat is false als een uitdrukking waar is en true als een uitdrukking onwaar is.
( )	Haakjes voor groeperen (om de volgorde waarin uitdrukkingen worden geëvalueerd expliciet aan te geven).

Tekenreeksoperatoren

Met tekenreeksoperatoren worden twee of meer tekenreeksen aaneengeschakeld tot één tekenreeks. In dBASE voor Windows worden de volgende tekenreeksoperatoren ondersteund:

Operator	Omschrijving
+	Combineert twee tekenreeksen tot één tekenreeks, waarbij de spaties tussen de reeksen blijven gehandhaafd.
-	Combineert twee tekenreeksen, waarbij de spaties aan het einde van de eerste tekenreeks naar het einde van de tweede worden verplaatst.
( )	Haakjes voor groeperen (om de volgorde waarin uitdrukkingen worden geëvalueerd expliciet aan te geven).

Objectoperatoren

Objectoperatoren worden gebruikt om objecten en verwijzingen naar objecten, kenmerken en methoden te maken. dBASE voor Windows ondersteunt de volgende objectoperatoren.

Operator	Omschrijving
NEW	Hiermee maakt u een nieuwe kopie van een object.
[ ]	Indexoperator. Hiermee vraagt u de inhoud van een object op via een numerieke waarde.
. (punt)	Stip-operator. Hiermee vraagt u de inhoud van een object op via een naam.

De operator NEW

Met de operator NEW maakt u een object van een specifieke klasse.

Hieronder ziet u de syntaxis voor de operator NEW:

```
<naam object> = NEW <naam klasse>([<parameters>])
```

In het volgende voorbeeld ziet u hoe u de operator NEW gebruikt om een formulier te maken uit de klasse Form. Standaard wordt in dBASE een objectverwijzingsvariabele gemaakt met dezelfde naam als die u voor het formulier gebruikt (KlantForm in dit voorbeeld).

```
KlantForm = NEW Form()
```

Het object blijft aanwezig zolang er verwijzingen naar bestaan.

Opmerking Het commando DEFINE vormt een andere manier om een object te maken.

Indexoperator

Met de indexoperator, [ ], verkrijgt u toegang tot de kenmerken of methoden van een object via een waarde, meestal een getal. Hieronder ziet u de syntaxis voor het gebruik van de indexoperator (die vaak de array-indexoperator wordt genoemd):

```
<objectverwijzing>[<Nuitdr>]
```

U gebruikt de indexoperator meestal om naar elementen van array-objecten te verwijzen zoals in het volgende voorbeeld:

```
TArray = NEW Array(20)    && Nieuwe array maken met 20 elementen
TArray[1] = 10            && Waarde van het eerste element wijzigen in 10
? TArray[1]               && Resultaat is 10
```

Stip-operator

Met de stip-operator, (“.”), verkrijgt u toegang tot de kenmerken of methoden van een object via een naam. Hieronder ziet u de syntaxis voor het gebruik van de stip-operator:

```
<objectverwijzing>.[<objectverwijzing>.]<naam kenmerk>
```

De stip-operator is de meest gebruikte manier om naar de kenmerken of methoden van een object te verwijzen. De opdracht in het volgende voorbeeld laat zien hoe u de stip-operator gebruikt om waarden toe te wijzen:

```
DEFINE FORM KlantFrm              && Nieuw formulierobject maken
KlantFrm.Text = "Klanten"         && Kenmerk Text definiëren van formulierobject KlantFrm
KlantFrm.OnOpen = OpenProc        && Actiekenmerk OnOpen definiëren van formulierobject
                                  && KlantFrm
```

Als een object een ander object bevat, kunt u de kenmerken van het sub-object opvragen door een pad van objectverwijzingen op te geven dat leidt naar het gewenste kenmerk zoals u in de volgende opdracht kunt zien.

```
DEFINE ENTRYFIELD NaamVeld OF KlantFrm   && Invoervak maken in formulierobject KlantFrm
KlantFrm.NaamVeld.Width = 20             && Kenmerk Width instellen van invoervak
```

Functie-operatoren

Functie-operatoren worden gebruikt om functies en procedures aan te roepen. In dBASE voor Windows worden de volgende functie-operatoren ondersteund.

Operator	Omschrijving
`<fp>([params])`	Aanroepoperator. Hiermee roept u een functie of procedure aan die bij de functie-aanwijzer *<fp>* hoort.
`<obj-ver>.<naam variabele>([params])`	Aanroepoperator voor element. Hiermee roept u een functie, procedure of codeblok aan dat hoort bij de opgegeven objectverwijzing.
`<obj-ver>[<uitdr>](<params>)`	Aanroepoperator voor index. Deze lijkt op de aanroepoperator voor een element maar hiermee roept u de functie of het codeblok met een index aan.

Aanroepoperator

Met de aanroepoperator, ( ), roept u een procedure of codeblok aan bij een bepaalde functie-aanwijzer of codeblokvariabele. Ook wordt de opgegeven parameter doorgegeven aan de procedure of het codeblok. Hieronder ziet u de syntaxis voor de aanroepoperator:

```
<functie-aanwijzer|codeblokvariabele>([parameters])
```

De volgende opdrachten zijn voorbeelden van het gebruik van de aanroepoperator om een procedure aan te roepen die bij een functie-aanwijzer hoort:

```
? Kwadraat(10)              && Procedure Kwadraat aanroepen. Resultaat is 100
? TYPE("Kwadraat")          && Resultaat is FP; Kwadraat is een constante van het type
                            && functie-aanwijzer
Bereken = Kwadraat          && Variabele maken van het type functie-aanwijzer
? Bereken(7)                && Procedure Kwadraat aanroepen. Resultaat is 49

PROCEDURE Kwadraat
PARAMETERS n
RETURN n*n
```

Bij de volgende opdracht ziet u hoe u met de aanroepoperator een codeblok aanroept:

```
x = {|z| z + 1}         && Variabele maken van het type codeblok
? x(6)                  && Codeblok aanroepen dat in de variabele is opgeslagen
```

Aanroepoperator voor elementen

Net zoals met de gewone aanroepoperator worden met de aanroepoperator voor elementen procedures of codeblokken aangeroepen die bij een bepaalde functie-aanwijzer of codeblokvariabele horen. Het verschil is dat de functie-aanwijzer of de codeblokvariabele zich nu in een object bevindt.

Als u de procedure of het codeblok aanroept, maakt dBASE bovendien een speciale objectverwijzingvariabele met de naam *this*. De variabele *this* verwijst naar het object met de functie-aanwijzer of codeblokvariabele. Hieronder ziet u de syntaxis voor de aanroepoperator voor elementen:

```
<objectverwijzing>.<functie-aanwijzer|codeblokvariabele>([parameters])
```

In het volgende voorbeeld ziet u hoe u de aanroepoperator voor elementen gebruikt.

```
x = NEW OBJECT()            && Nieuw object maken
x.Proc = ToonWaarde         && Procedure toewijzen aan variabele van het type
                            && functie-aanwijzer
x.Waarde = 7                && Getal toewijzen aan de variabele Waarde
? x.Proc()                  && Procedure aanroepen in de variabele fp in x. Resultaat is 7

PROCEDURE ToonWaarde
RETURN this.waarde
```

Zoals het voorbeeld laat zien, verwijst de variabele *this* (net zoals de objectverwijzingvariabele x) naar het object. In feite zou het resultaat van het commando `RETURN x.waarde` identiek zijn aan het resultaat van het commando `RETURN this.waarde`. Het verschil is dat u met x een vaste verwijzing naar het object gebruikt. U krijgt alleen het juiste resultaat bij object x. Op de getoonde manier kunt u de procedure bij elk ander object ongewijzigd gebruiken.

Aanroepoperator voor indexen

De aanroepoperator voor indexen werkt op dezelfde manier als de aanroepprocedure voor elementen. Echter, ditmaal worden procedures of codeblokken aangeroepen via een indexwaarde. Net zoals bij de aanroepoperator voor elementen maakt ook hier dBASE een speciale objectverwijzingvariabele met de naam *this* wanneer de procedure

of het codeblok wordt aangeroepen. Hieronder ziet u de syntaxis voor de aanroepoperator voor indexen.

```
<functie-aanwijzer|codeblokvariabele>[Nuitdr]([parameters])
```

U kunt via de index toegang krijgen tot elementen zonder naam uit een bepaalde klasse, net zoals u kunt verwijzen naar array-elementen.

```
x = NEW ARRAY()     && Nieuw array-object maken
x[1] = MijnProc     && Procedure toewijzen aan array-element 1
x.waarde = 10       && Getal toewijzen aan de variabele Waarde
? x[1]()            && Procedure aanroepen die is opgeslagen in element 1. Resultaat is 10

PROCEDURE MijnProc
RETURN this.waarde
```

Bereikoperator

Met de bereikoperator, (::), kunt u naar methoden in een klasse verwijzen. Dit is handig als u een hiërarchische structuur van klassen hebt gemaakt en u een methode uit een hogere klasse wilt aanroepen. U kunt bijvoorbeeld een uit een hogere klasse gekopieerde methode in een subklasse wijzigen door de methode direct in de hogere klasse aan te roepen en deze vervolgens in de subklasse aan te passen. Zie Hoofdstuk 11 in *Programmeren* voor meer informatie over dit onderwerp.

Hieronder ziet u de syntaxis voor het gebruik van de bereikoperator:

```
<naam klasse> | class | super :: <naam methode>
```

Zoals de syntaxis al aangeeft, kunt u naar de hogere klasse verwijzen door middel van de naam van die klasse of met een van de sleutelwoorden *class* of *super*. Het sleutelwoord *class* verwijst naar de huidige klasse en het sleutelwoord *super* naar de klasse één niveau hoger.

In het volgende voorbeeld ziet u hoe u met de knop PiepVerplaats toegang krijgt tot de procedure PiepProc uit de hogere klasse PiepKnop. Vervolgens wordt een nieuwe functie aan de knop toegevoegd.

```
CLASS PiepKnop(form) OF MijnKnop(form)
   this.text = "Piep"
   this.Onclick = PiepProc

   PROCEDURE PiepProc
      ? CHR(7)
   RETURN
ENDCLASS


CLASS PiepVerplaats(form) OF PiepKnop(form)
   this.text = "Piep en verplaats"   && Kenmerk Text van de hogere klasse vervangen

      PROCEDURE OnClick
         super::PiepProc()            && Verwijzen naar methode Piepproc uit hogere klasse
         this.Left = this.Left+1      && Nieuwe functie toevoegen aan methode OnClick
```

```
      RETURN
   ENDCLASS
```

Opmerking De opdracht `super::PiepProc()` is identiek aan `PiepKnop::PiepProc()`.

Met de bereikoperator kunt u een methode uit een klasse aanroepen. Net zoals bij elke andere procedure moet de specificatie van de klasse in hetzelfde programmabestand staan als de aanroep of als procedurebestand worden geladen met het commando SET PROCEDURE. In het volgende voorbeeld ziet u hoe u bereikoperatoren kunt gebruiken om een methode in een klasse aan te roepen.

```
? MijnKlasse::ZegHallo()   && Methode aanroepen in MijnKlasse. Resultaat is "Hallo" op
                              het scherm

CLASS MijnKlasse
      PROCEDURE ZegHallo
         ? "Hallo"
      RETURN .T.
   ENDCLASS
```

Volgorde van operatoren

U kunt verschillende operatoren in één uitdrukking gebruiken. De wijze waarop in dBASE voor Windows een uitdrukking wordt geëvalueerd, wordt bepaald door de prioriteit van de verschillende operatoren. In de volgende tabel vindt u de volgorde waarin in dBASE voor Windows de operatoren worden geëvalueerd.

Operator	Omschrijving
()	Haakjes voor groeperen in alle uitdrukkingen
[ ]	Veldnaam, tekengeheugenvariabele en array-indexen
->	Alias-symbool
. (punt)	Stip-operator
NEW	Operator voor nieuw object
+, -	Monadisch plus- (+) en minteken(-)
**, ^	Machtsverheffen
*, /	Vermenigvuldigen en delen
+, -	Optellen en aftrekken
=, <>, <, <=, >, >=, $	Relationele operatoren
.NOT., .AND, .OR.	Logische operatoren

U kunt haakjes gebruiken om uitdrukkingen te groeperen en expliciet de gewenste volgorde waarin uitdrukkingen worden geëvalueerd aan te geven. Als u geneste haakjes gebruikt, worden de uitdrukkingen in de binnenste haakjes als eerste geëvalueerd.

Hoofdstuk

2

Syntaxis

In *Commando's en functies* wordt een aantal symbolen en conventies gebruikt voor de syntactische notatie van de dBASE-taal. In dit hoofdstuk worden de syntactische symbolen beschreven. Tevens wordt beschreven hoe u de syntactische conventies dient te interpreteren.

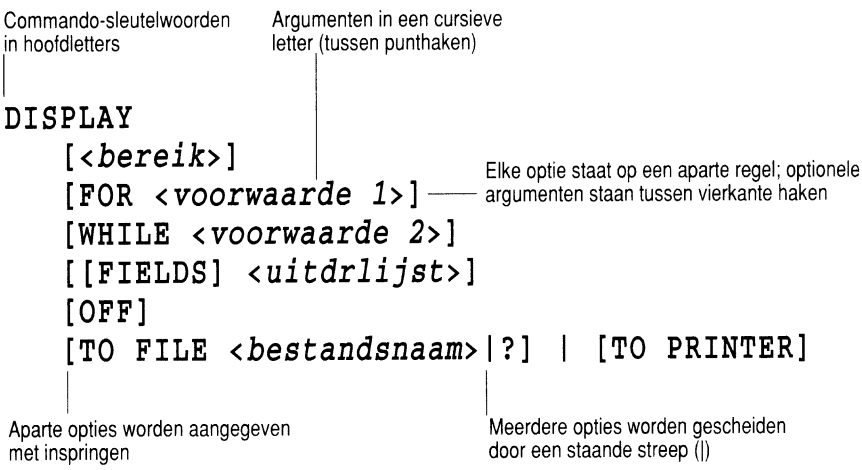
Syntactische notatie

In de volgende tabel worden de syntactische symbolen beschreven.

Tabel 2.1

Symbool	Beschrijving
[ ]	Geeft een optioneel element aan.
< >	Geeft een verplicht argument aan. Bijvoorbeeld, <*Nuitdr*> geeft aan dat u een numerieke uitdrukking moet opgeven.
{ }	Geeft een codeblok aan.
| |	Geeft een parameterlijst in een codeblok aan.
|	Geeft twee of meer elkaar uitsluitende opties aan, bijvoorbeeld ON | OFF.
...	Geeft een element aan dat een willekeurig aantal malen kan worden herhaald. bijvoorbeeld [, <*veld2*> WITH <*exp2*> [ADDITIVE]...] geeft een dat u meer dan één veld kunt opgeven om te vervangen.
,	Lijstscheidingsteken
;	Scheidingstekens tussen commando's in een codeblok. Geeft in een commandoregel aan dat het commando wordt voortgezet op de volgende regel.

In de volgende afbeelding ziet u een voorbeeld van commando-syntaxis.



In tegenstelling tot de sleutelwoorden van commando's en functies in dBASE, die worden aangegeven in hoofdletters, hebben de namen van kenmerken een andere notatie. Voor de namen van kenmerken worden kameelkapitalen gebruikt, dat wil zeggen afwisselend hoofd- en kleine letters, als de naam meer dan één woord bevat. Is het kenmerk een methode, dan wordt de naam gevolgd door haakjes. Voorbeelden van kenmerken zijn onder meer OnAppend, OnRightMouseDown, Checked en Close( ).

Deze conventies helpen u een duidelijk onderscheid te maken tussen de verschillende elementen van de taal. Bijvoorbeeld:

- DELETE is een opdracht
- Delete is een kenmerk
- DELETED( ) is een functie
- Delete( ) is een methode

Deze typografische conventies worden alleen gebruikt voor de leesbaarheid. Bij het schrijven van code kunt u willekeurig gebruik maken van hoofd- en kleine letters.

Opmerking De meeste sleutelwoorden in de dBASE-taal kunnen worden afgekort tot de eerste vier letters. Dit is echter niet raadzaam omwille van de leesbaarheid. Klassen en kenmerken moet u echter opgeven met de volledige naam. U kunt immers eigen klassen en kenmerken definiëren, zodat alleen met behulp van de volledige naam de gewenste klasse of het gewenste kenmerk kan worden geïdentificeerd.

Richtlijnen voor het interpreteren van de syntaxis

In deze sectie wordt aangegeven hoe u de meer ingewikkelde syntactische conventies interpreteert.

- Opties staan in alfabetische volgorde, behalve als een andere volgorde vereist of zinvoller is. Bijvoorbeeld:

```
DEFINE <klassenaam> <objectnaam>
  [OF <container-object>]
  [FROM <rij, kol> ...]
```

Bij het schrijven van een dBASE-opdracht kunt u de opties meestal in een willekeurige volgorde opgeven.

- De staande streep (|) wordt gebruikt voor opties die elkaar uitsluiten. Als de opties niet op één regel passen, springen de tweede en eventuele volgende regels in, waarbij de staande streep telkens aan het einde van de voorgaande regel staat. Bijvoorbeeld:

```
MONO | COLOR | EGA25 | EGA43 | MONO43 |
   VGA25 | VGA43 | VGA50
```

- Als een sleutelwoord voorafgaat aan een reeks optie-sleutelwoorden, staat elke optie op een aparte regel die inspringt ten opzichte van het sleutelwoord waarbij de optie wordt gebruikt:

```
PROMPT
  ARRAY <array> |
  FIELD <veld> |
  FILES [LIKE <bestandsnaamfilter>] |
  STRUCTURE [IN <alias Tuitdr>]
```

In dit voorbeeld kunt u een van de vier opties kiezen: PROMPT ARRAY..., PROMPT FIELD..., PROMPT FILES... of PROMPT STRUCTURE...

- Vierkante haken [ ] geven een optioneel element aan. In sommige gevallen zijn optionele elementen genest in andere optionele elementen. Bijvoorbeeld:

```
ACOPY(<naam bron-array>, <naam doel-array>
   [, <beginelement Nuitdr> [, <elementen Nuitdr> [, <doelelement Nuitdr>]]])
```

Bij het commando ACOPY( ) geeft u een lijst van argumenten op die onderling worden gescheiden door een komma. Het derde, vierde en vijfde argument zijn optioneel. In het voorbeeld volgen na het derde en na het vierde argument geen sluithaken. Dit houdt in, dat voor het derde argument een waarde is vereist als u voor het vierde argument een waarde wilt opgeven. Op dezelfde wijze zijn waarden vereist voor het derde en het vierde argument als u voor het vijfde argument een waarde wilt opgeven. Vergelijk dit voorbeeld met het volgende.

- Soms zijn optionele elementen genest in andere optionele elementen, maar onderling onafhankelijk. Bijvoorbeeld:

```
SET COLOR TO
  [<standaard tekst>][,[<benadrukte tekst>][,[<buitenrand>][,[<achtergrond>]]]]
```

Zoals in het vorige voorbeeld met ACOPY( ) heeft ook SET COLOR TO een lijst van optionele argumenten die onderling worden gescheiden door een komma. In dit voorbeeld echter staat elk argument tussen vierkante haken. Dit houdt in, dat u elk argument kunt opgeven zonder eerst een ander argument op te geven. Laat u een argument weg, dan geeft u wel de positie van dit argument aan met een komma. Bijvoorbeeld, `SET COLOR TO , , B` houdt in, dat de buitenrand is ingesteld op blauw.

- Argumenten waarvoor u een waarde of een uitdrukking opgeeft staan in cursieve letters tussen punthaken. In plaats van alleen het vereiste gegevenstype te vermelden

(bijvoorbeeld <*Tuitdr*> or <*Nuitdr*>) geeft de syntaxis een meer beschrijvende argumentnaam. Bijvoorbeeld:

```
<doelelement Nuitdr>
```

Hoofdstuk

3

Taalelementen per categorie

dBASE bevat meer dan 600 commando's, functies, standaard klassen, systeemgeheugenvariabelen en preprocessor-instructies. In dit hoofdstuk wordt aangegeven welke taalelementen beschikbaar zijn voor het programmeren van specifieke taken. Dit hoofdstuk bevat een opsomming van taalelementen in taakgerichte categorieën.

Opmerking Naast de namen van een taalelementen zoals die verderop in dit boek worden besproken, treft u ook de naam van de bijbehorende categorieën aan.

Programma's

&&	DO	IIF()	RETURN
*	DO CASE	MODIFY COMMAND	SCAN
CANCEL	DO WHILE	NOTE	SET DEVELOPMENT
CLEAR PROGRAM	DO...UNTIL	PARAMETERS	SET LIBRARY
CLOSE PROCEDURE	FOR...NEXT	PCOUNT()	SET PROCEDURE
CREATE COMMAND	FUNCTION	PROCEDURE	SLEEP
COMPILE	IF	QUIT	

Geheugenvariabelen

ACOPY()	AGROW()	ASUBSCRIPT()	RELEASE
ADEL()	AINS()	CLEAR MEMORY	RESTORE
ADIR()	ALEN()	DECLARE	SAVE
AELEMENT()	ARESIZE()	LOCAL	STATIC
AFIELDS()	ASCAN()	PRIVATE	STORE
AFILL()	ASORT()	PUBLIC	

Foutafhandeling en testen op fouten

CERROR()	GENERATE	PROGRAM()	SET STEP
DBERROR()	LINENO()	RESUME	SQLERROR()
DBMESSAGE()	LIST COVERAGE	RETRY	SQLMESSAGE()
DEBUG	MESSAGE()	SET COVERAGE	SUSPEND
DISPLAY COVERAGE	ON ERROR	SET ECHO	
ERROR()	ON READERROR	SET ERROR	

Tekenreeksgegevens

ANSI()	LEFT()	RAT()	STUFF()
AT()	LEN()	REPLICATE()	SUBSTR()
CENTER()	LIKE()	RIGHT()	TRANSFORM()
DIFFERENCE()	LOWER()	RTRIM()	TRIM()
ISALPHA()	LTRIM()	SET EXACT	UPPER()
ISLOWER()	OEM()	SOUNDEX()	
ISUPPER()	PROPER()	SPACE()	

Numerieke gegevens

ABS()	EXP()	PAYMENT()	SET POINT
ACOS()	FLOOR()	PI()	SET PRECISION
ASIN()	FV()	PV()	SET SEPARATOR
ATAN()	INT()	RANDOM()	SIGN()
ATN2()	LENNUM()	ROUND()	SIN()
CEILING()	LOG()	RTOD()	SQRT()
COS()	LOG10()	SET CURRENCY	TAN()
DTOR()	MOD()	SET DECIMALS	

Datum- en tijdgegevens

CDOW()	DOW()	SECONDS()	SET MARK
CMONTH()	ELAPSED()	SET CENTURY	SET TIME
DATE()	MDY()	SET DATE	TIME()
DAY()	MONTH()	SET DATE TO	YEAR()
DMY()			

Uitdrukkingen en gegevenstypeconversie

ASC()	DTOS()	HTOI()	STR()
CHR()	EMPTY()	ITOH()	TYPE()
CTOD()	GETEXPR()	MAX()	VAL()
DTOC()	FLOAT()	MIN()	

Tabellen

ALIAS()	COPY TO...STRUCTURE EXTENDED	IMPORT	SET CATALOG
APPEND FROM	CREATE	ISTABLE()	SET DATABASE
CATALOG()	CREATE CATALOG	LIST STRUCTURE	SET DBTYPE
CLOSE ALL	CREATE...FROM	MODIFY STRUCTURE	SET TITLE
CLOSE DATABASES	CREATE...STRUCTURE EXTENDED	OPEN DATABASE	SQLEXEC()
CLOSE TABLES	DATABASE()	REFRESH	USE
COPY	DBF()	RENAME TABLE	WORKAREA()
COPY STRUCTURE	DELETE TABLE	SELECT	
COPY TABLE	DISPLAY STRUCTURE	SELECT()	

Velden en records

APPEND	COUNT	LUPDATE()	REPLACE OLE
APPEND AUTOMEM	DELETE	MEMLINES()	SET AUTOSAVE
APPEND FROM ARRAY	DELETED()	MLINE()	SET BLOCKSIZE
APPEND MEMO	EDIT	PACK	SET CARRY
BINTYPE()	EOF()	RECALL	SET DELETED
BLANK	FDECIMAL()	RECCOUNT()	SET FIELDS
BOF()	FIELD()	RECNO()	SET MBLOCK
BOOKMARK()	FLDCOUNT()	RECSIZE()	SET MEMOWIDTH
BROWSE	FLDLIST()	RELEASE AUTOMEM	SET WINDOW OF MEMO
CHANGE	FLENGTH()	REPLACE	SKIP
CLEAR AUTOMEM	FLUSH	REPLACE AUTOMEM	STORE AUTOMEM
CLEAR FIELDS	GO	REPLACE BINARY	STORE MEMO
COPY BINARY	INSERT	REPLACE FROM ARRAY	ZAP
COPY MEMO	INSERT AUTOMEM	REPLACE MEMO	
COPY TO ARRAY	ISBLANK()	REPLACE MEMO... FROM	

Tabelindeling

AVERAGE	FOR()	ORDER()	SET UNIQUE
CALCULATE	FOUND()	REINDEX	SET VIEW
CLOSE INDEXES	INDEX	RELATION()	SORT
CONTINUE	JOIN	SEEK	SUM
COPY INDEXES	KEY()	SEEK()	TAG()
COPY TAG	KEYMATCH()	SET FILTER	TAGCOUNT()
CREATE QUERY	LIST	SET IBLOCK	TAGNO()
CREATE VIEW	LOCATE	SET INDEX	TARGET()
CREATE VIEW... FROM ENVIRONMENT	LOOKUP()	SET KEY TO	TOTAL
DELETE TAG	MDX()	SET NEAR	UNIQUE()
DESCENDING()	MODIFY QUERY	SET ORDER	UPDATE
DISPLAY	MODIFY VIEW	SET RELATION	
FIND	NDX()	SET SKIP	

Afdrukken

???	_pdriver	_pspacing	PCOL()
_alignment	_peject	_rmargin	PRINTJOB
_box	_pepage	_tabs	PRINTSTATUS()
_indent	_pform	_wrap	PROW()
_lmargin	_plength	CHOOSEPRINTER()	SET MARGIN
_padvance	_plineno	CLOSE PRINTER	SET PCOL
_pageno	_ploffset	DEFINE BOX	SET PRINTER
_pbpage	_porientation	EJECT	SET PROW
_pcolno	_ppitch	EJECT PAGE	
_pcopies	_pquality	ON PAGE	

Invoer/Uitvoer

?	CLEAR GETS	MODIFY LABEL	SET DELIMITERS
??	CLEAR SCREENS	MODIFY REPORT	SET DEVICE
@...CLEAR	CLOSE ALTERNATE	READ	SET FORMAT
@...FILL	CLOSE FORMAT	RELEASE SCREENS	SET HEADINGS
@...SAY...GET	COL()	REPORT FORM	SET SPACE
@...SCROLL	CREATE LABEL	RESTORE SCREENS	TEXT
@...TO	CREATE REPORT	ROW()	UPDATED()
ACCEPT	INPUT	SAVE SCREEN	VARREAD()
ACTIVATE SCREEN	LABEL FORM	SET ALTERNATE	WAIT

dBASE IV-vensters

ACTIVATE WINDOW	DEFINE WINDOW	RELEASE WINDOWS	SAVE WINDOW
CLEAR WINDOWS	MOVE WINDOW	RESTORE WINDOW	WINDOW()
DEACTIVATE WINDOW			

dBASE IV-menu's

ACTIVATE MENU	DEFINE BAR	ON EXIT POPUP	PADPROMPT()
ACTIVATE POPUP	DEFINE MENU	ON MENU	POPUP()
BAR()	DEFINE PAD	ON PAD	PROMPT()
BARCOUNT()	DEFINE POPUP	ON POPUP	RELEASE MENUS
BARPROMPT()	MENU()	ON SELECTION BAR	RELEASE POPUPS
CLEAR MENUS	ON BAR	ON SELECTION MENU	SHOW MENU
CLEAR POPUPS	ON EXIT BAR	ON SELECTION PAD	SHOW POPUP
DEACTIVATE MENU	ON EXIT MENU	ON SELECTION POPUP	
DEACTIVATE POPUP	ON EXIT PAD	PAD()	

Formulieren

_curobj	CREATE MENU	MODIFY MENU	OPEN FORM
CLOSE FORMS	CREATE SCREEN	MODIFY SCREEN	READMODAL()
CREATE APPLICATION	MODIFY APPLICATION	MSGBOX()	SET CUAENTER
CREATE FORM	MODIFY FORM	ON SELECTION FORM	

Objecten

_app	INSPECT()	PLAY SOUND	RESTORE IMAGE
CLASS...ENDCLASS	LISTCOUNT()	REDEFINE	SHOW OBJECT
DEFINE	LISTSELECTED()	RELEASE OBJECT	

Toetsenbord- en muisacties

CLEAR TYPEAHEAD	KEYBOARD	ON ESCAPE	SET ESCAPE
FKLABEL()	LASTKEY()	ON KEY	SET FUNCTION
FKMAX()	MCOL()	ON MOUSE	SET KEY
INKEY()	MROW()	READKEY()	SET MOUSE
ISMOUSE()	NEXTKEY()	SET CURSOR	SET TYPEAHEAD

Kleuren en fonts

DEFINE COLOR	GETFONT()	SET COLOR OF	SET COLOR TO
GETCOLOR()	ISCOLOR()		

Omgeving

CHARSET()	LIST STATUS	SET EDITOR	SET TALK
CLEAR	MEMORY()	SET FULLPATH	SET WP
CLEAR ALL	SET BELL	SET INTENSITY	SET
CREATE SESSION	SET BORDER	SET LDCHECK	SET()
DISPLAY MEMORY	SET CONFIRM	SET MESSAGE	SETTO()
DISPLAY STATUS	SET CONSOLE	SET ODOMETER	SHELL()
LDRIVER()	SET DESIGN	SET SAFETY	VERSION()
LIST MEMORY	SET DISPLAY		

Stations- en bestandsfuncties

!	DOS	GETFILE()	RUN
_dbwinhome	ERASE	HOME()	RUN()
CD	FDATE()	LIST FILES	SET DEFAULT
COPY FILE	FILE()	MD	SET DIRECTORY
CREATE FILE	FSIZE()	MKDIR	SET PATH
DELETE FILE	FTIME()	MODIFY FILE	TYPE
DIR	FUNIQUE()	OS()	VALIDDRIVE()
DISKSPACE()	GETDIRECTORY()	PUTFILE()	
DISPLAY FILES	GETENV()	RENAME	

Toegang op laag niveau

FCLOSE()	FERROR()	FOPEN()	FSEEK()
FCREATE()	FFLUSH()	FPUTS()	FWRITE()
FEOF()	FGETS()	FREAD()	

Preprocessor-instructies

#define	#ifdef	#include	#undef
#if	#ifndef	#pragma	

Gedeelde gegevens

BEGINTRANS()	ID()	ON NETERROR	SET LOCK
CHANGE()	LKSYS()	RLOCK()	SET REFRESH
COMMIT()	LOCK()	ROLLBACK()	SET REPROCESS
CONVERT	NETWORK()	SET EXCLUSIVE	UNLOCK
FLOCK()			

Programmeren in Windows

BITAND()	BITSET()	HELP	RESOURCE()
BITLSHIFT()	BITXOR()	LOAD DLL	SET HELP
BITOR()	EXTERN	RELEASE DLL	SET TOPIC
BITRSHIFT()			

Klassen

ARRAY	EDITOR	LISTBOX	RADIOBUTTON
BROWSE	ENTRYFIELD	MENU	RECTANGLE
CHECKBOX	FORM	OBJECT	SCROLLBAR
COMBOBOX	IMAGE	OLE	SPINBOX
DDELINK	LINE	PUSHBUTTON	TEXT
DDETOPIC			

Commando's en functies

Hoofdstuk

4

Commando's en functies

! Stations- en bestandsfuncties

Voert een enkele DOS-opdracht uit of start een programma vanuit dBASE.

Syntaxis

! *<DOS-opdracht>*

<DOS-opdracht>
Een opdracht die door het besturingssysteem DOS wordt herkend.

Beschrijving

! en RUN zijn onderling uitwisselbaar. Zie de beschrijving van RUN.

Voorbeeld

In de volgende voorbeelden worden ! en RUN gebruikt om met de DOS-opdracht COPY de bestanden KLANTEN.DBF en KLANTEN.MDX naar een diskette te kopiëren:

```
! COPY C:\DBASEWIN\VOORBD\KLANTEN.DBF B:
RUN COPY C:\DBASEWIN\VOORBD\KLANTEN.MDX B:
```

Zie ook

DOS, RUN

Markeert de tekens rechts van het commando als een commentaar in plaats van uitvoerbare code.

Syntaxis

&& *<commentaar>*

<commentaar>
Een reeks tekens op dezelfde regel achter &&. Als het laatste teken een puntkomma is, wordt ook de volgende regel als een commentaar beschouwd.

Beschrijving

Gebruik het dubbele en-teken (&&) om rechts op een programmaregel een kort commentaar toe te voegen. U kunt && gebruiken voor een commentaar van een hele regel, maar in dat geval is een asterisk (*) of het commando NOTE gebruikelijker.

Met commentaarregels kunt u de functie of logica van een reeks commando's in een programma documenteren. Als u het programma op een later tijdstip wilt veranderen, zijn commentaarregels nuttig om te zien wat de bedoeling van de verschillende onderdelen is.

U mag && gebruiken na de puntkomma die aangeeft dat een instructie op de volgende regel verder gaat. Tevens mag u een puntkomma aan het eind van een commentaar plaatsen om aan te geven dat het commentaar op de volgende regel wordt voortgezet. Als op een regel voor && een puntkomma wordt aangetroffen, wordt aangenomen dat de volgende regel deel uitmaakt van het commando. Als de puntkomma na && staat, wordt de volgende regel beschouwd als een voortzetting van het commentaar.

Voorbeeld

In het volgende voorbeeld wordt && gebruikt om aantekeningen toe te voegen aan programmaregels:

```
SET TALK OFF                && Weergave geheugenvariabelen;
                               onderdrukken
CLEAR                       && Resultatenpaneel wissen
SELECT 1                    && Werkgebied 1 selecteren
USE Contact ORDER CompCode  && .MDX-labelvolgorde aanroepen
SELECT 2                    && Werkgebied 2 activeren
USE Bedrijf ORDER CompCode
SET RELATION TO CompCode INTO A;
                            && Werkgebieden A en B (1 en 2);
                               koppelen;
                               Zie Alias in Grondbeginselen
SET FIELDS TO Bedrijf, Regio, A->Contact;
                            && Voor velden in huidige;
                               werkgebied zijn geen;
                               aliasnamen vereist.
LIST TO PRINT OFF           && OFF onderdrukt;
```

```
                                    recordnummers.
RETURN
```

Zie ook

*

*

Programma's

Markeert een volledige programmaregel als een commentaar in plaats van een regel met uitvoerbare code.

Syntaxis

* *<commentaar>*

<commentaar>

Een tekst van maximaal 1023 tekens op dezelfde regel. Als het laatste teken op de regel een puntkomma is, wordt ook de volgende regel als een commentaar beschouwd.

Beschrijving

Gebruik een asterisk (*) om aan te geven dat een volledige programmaregel een commentaar bevat en geen uitvoerbare code. De asterisk moet het eerste teken op de regel zijn (eventueel voorafgegaan door spaties of tabs). Als u het programma uitvoert, wordt de regel door dBASE genegeerd.

Met commentaarregels kunt u de functie of logica van een reeks commando's in een programma documenteren. Als u het programma op een later tijdstip wilt veranderen, kunt u aan de commentaarregels zien wat de bedoeling van de verschillende onderdelen is. U kunt de asterisk ook gebruiken om tijdens het testen een commandoregel tijdelijk te veranderen in een commentaar.

Gebruik && om een commentaar op te nemen op dezelfde regel als een programma-instructie.

De commando's * en NOTE zijn onderling uitwisselbaar.

Voorbeeld

In het volgende voorbeeld worden asterisken gebruikt om commentaren op te nemen tussen programmacode:

```
** AFNMRRPT.PRG **
USE Afnemers ORDER AfnemerNr
* Tabel gerangschikt door AfnemerNr .MDX indexlabel
SET FIELDS TO Bedrijf, Contact
* Weergave maken die uit slechts twee velden bestaat
LIST OFF TO PRINT
* Uitvoer naar printer sturen van geselecteerde velden;
  voor volledige tabel;
```

```
  OFF onderdrukt afdrukken van recordnummers
CLOSE DATABASES
```

Zie ook

&&

? Invoer/uitvoer

Evalueert nul of meer uitdrukkingen. Het resultaat wordt weergegeven of afgedrukt op een nieuwe regel.

Syntaxis

?
[<*uitdr1*>
[PICTURE <*opmaak Tuitdr*>]
[FUNCTION <*functie Tuitdr*>]
[AT <*kolom Nuitdr*>]
[STYLE [<*fontstijl Nuitdr*>] [<*fontstijl Tuitdr*>]]
[,<*uitdr2*>...]
[,]

<*uitdr1*>[,<*uitdr2*> ...]

Een of meer uitdrukkingen die u wilt evalueren. De uitdrukkingen kunnen van elk willekeurig gegevenstype zijn.

PICTURE <*opmaak Tuitdr*>

Maakt <*uitdr1*> of een opgegeven gedeelte daarvan op met het veldopmaaksjabloon <*opmaak Tuitdr*>. Deze tekenuitdrukking bestaat uit een van de volgende onderdelen:

- Sjabloontekens.
- Functiesymbolen voorafgegaan door @. (U kunt ook de optie FUNCTION gebruiken; zie verderop.)
- Vaste tekens.
- Een combinatie van sjabloontekens, functiesymbolen en vaste tekens.
- Een variabele die de tekenuitdrukkking bevat.

U mag alle sjabloontekens en functiesymbolen gebruiken met uitzondering van A, M, R en S. Zie Picture in Hoofdstuk 8 voor meer informatie over sjabloontekens.

FUNCTION <*functie Tuitdr*>

Maakt alle tekens in <*uitdr1*> op met het functiesjabloon <*functie Tuitdr*>, dat één of meer functiesymbolen moet bevatten. Als u functiesymbolen opgeeft met de optie FUNCTION, hoeft u die niet vooraf te laten gaan door @. Zie Function in Hoofdstuk 8 voor meer informatie over functiesymbolen.

AT <*kolom Nuitdr*>

Bepaalt de tekenkolom, <*kolom Nuitdr*>, waarin het commando ? begint met het weergeven of afdrukken van <*uitdr1*>. Het argument <*kolom Nuitdr*> moet tussen 0 en 255 liggen.

STYLE [<*fontstijl Nuitdr*>] [<*fontstijl Tuitdr*>]

Bepaalt een optioneel font of een optionele stijloptie. De fonts (<*fontstijl Nuitdr*>) worden opgegeven in het gedeelte [Fonts] van DBASEWIN.INI. De syntaxis daarvoor is als volgt:

```
[Fonts]
1=Times New Roman,12,ROMAN
2=Courier New,10,MODERN
```

Het eerste argument is de lettertypenaam van het font, het tweede argument de puntgrootte en het derde argument de font-familie. U kunt maximaal 32.766 fontstijlen definiëren. (Gebruik geen scheidingstekens voor duizendtallen als u getallen groter dan 999 opgeeft.)

Als u een font wilt toevoegen aan DBASEWIN.INI maar niet de exacte naam of familie kent, gebruikt u GETFONT( ). De informatie die GETFONT( ) levert, kunt u toevoegen aan DBASEWIN.INI.

De volgende tabel bevat de codes die u kunt gebruiken voor <*fontstijl Tuitdr*>:

<fontstijl Tuitdr>	Resultaat
"B"	Vet
"I"	Cursief
"U"	Onderstrepen
"R"	Superscript
"L"	Subscript

,

De volgkomma is optioneel en heeft geen functie; deze is uitsluitend aanwezig vanwege de compatibiliteit met dBASE IV.

Beschrijving

Gebruik ? om de waarde van geldige uitdrukkingen van elk willekeurig type weer te geven of af te drukken (naar een bestand of op de printer). U kunt ? op dezelfde manier gebruiken als DISPLAY (om bijvoorbeeld de inhoud van het huidige record weer te geven). In programma's kunt u ? zonder uitdrukking gebruiken als u in de uitvoer een regel wilt overslaan.

Het enige verschil tussen de commando's ? en ?? is dat ? naar een nieuwe regel gaat voordat de waarde wordt weergegeven, terwijl ?? dat niet doet.

Bij SET PRINTER ON wordt de uitvoer van het commando ? naar de standaardprinter gestuurd of naar de printer of het bestand dat u hebt opgegeven met SET PRINTER TO. Bij SET ALTERNATE ON wordt de uitvoer van het commando ? opgeslagen in het bestand dat u hebt opgegeven met SET ALTERNATE TO.

Met de optie STYLE kunt u bij ? en ?? de fontstijlen van afzonderlijke onderdelen wijzigen.

Als u wilt dat een regel tekst uit een programmabestand in afgedrukte uitvoer wordt doorgehaald (eerst wordt afgedrukt en vervolgens overschreven door een volgende regel), gebruikt u de optie AT met false (.F.) als instelling voor _wrap. Als u wilt dat tekst in afgedrukte uitvoer niet wordt doorgehaald maar overschreven, gebruikt u de optie met true (.T.) als instelling voor _wrap. In het laatste geval wordt alleen de tweede regel afgedrukt.

Met de functies B, I en J kunt u zowel een algemene instelling voor _alignment als afzonderlijke alinea-uitlijningen in een memoveld wijzigen.

Geef het commando SET SPACE ON (de standaardinstelling) als u wilt dat spaties worden ingevoegd tussen uitdrukkingen in de lijst als die wordt weergegeven of afgedrukt.

Voorbeeld

In het volgende voorbeeld worden ? en ?? gebruikt om een voornaam en een achternaam op verschillende wijzen opgemaakt weer te geven:

```
Voornaam   ="Sofie    "
Achternaam ="Stevens  "
? Voornaam,Achternaam
* eenvoudige weergave, geen opmaak of plaatsing
* Sofie     Stevens

? Voornaam PICTURE "@T"
?? " "
?? Achternaam PICTURE "@!"
* ingekorte voornaam, achternaam in hoofdletters
* Sofie STEVENS

? Achternaam STYLE "B"
?? Voornaam AT 20
* weergave in vaste kolommen.
* Achternaam wordt vet afgedrukt
* Stevens            Sofie
```

Dit voorbeeld maakt -3273,68 op vier verschillende manieren op:

```
N=-3273.68
? N PICTURE "9,999,999.99" && punten invoegen
*    -3.273,68
? N PICTURE "9,999,999"    && geen decimalen
*    -3.274
? N PICTURE "@L 9,999,999" && opvullen met voorloopnullen
* -0003.274
? N PICTURE "@(BT"         && () voor negatieve getallen;
                              links uitlijnen en inkorten
* (3273,68)
```

Overdraagbaarheid

De opties AT, PICTURE, FUNCTION, STYLE en volgkomma's worden niet ondersteund in dBASE III PLUS.

Zie ook

??, _alignment, _wrap, DISPLAY, GETFONT( ), LIST, PRINTJOB, SET MEMOWIDTH, SET ALTERNATE, SET PRINTER, SET SPACE, TEXT

??

Invoer/uitvoer

Evalueert nul of meer uitdrukkingen. Het resultaat wordt afgedrukt of weergegeven op de huidige regel.

Syntaxis

```
??
   [<uitdr1>
      [PICTURE <opmaak Tuitdr>]
      [FUNCTION <functie Tuitdr>]
      [AT <kolom Nuitdr>]
      [STYLE [<fontstijl Nuitdr>] [<fontstijl Tuitdr>] ]
   [,<uitdr2>[,...]]
   [;]
```

Beschrijving

Het commando ?? komt overeen met het commando ?, maar geeft weer of drukt af op de huidige regel in plaats van op een nieuwe regel. Zie het commando ? voor een beschrijving van de opties voor beide commando's.

Voorbeeld

Zie het voorbeeld bij het commando ?, waar ook een voorbeeld met het commando ?? staat.

Zie ook

?

???

Afdrukken

Stuurt uitvoer rechtstreeks naar de printer en gaat daarbij voorbij aan het geïnstalleerde printerstuurprogramma. Het commando is beschikbaar vanwege de compatibiliteit met dBASE IV. Het gebruik van dit commando wordt in dBASE voor Windows echter afgeraden.

Syntaxis

??? <*Tuitdr*>

<Tuitdr>
Een tekenreeks die naar de printer moet worden gestuurd.

Beschrijving

??? wordt in de DOS-omgeving gebruikt om printerbesturingscodes door te geven als de huidige printeraansturing een bepaalde afdrukfaciliteit niet ondersteunt. Deze codes geven de printer opdracht een bepaalde afdrukstijl (cursief, vet of onderstreept) of afdrukstand (liggend of staand) te gebruiken.

Als u in dBASE voor Windows ??? wilt gebruiken om codes naar de printer te sturen, doet u er goed aan het commando eerst te testen in combinatie met het printerstuurprogramma dat u wilt gebruiken. ??? wordt alleen ondersteund voor een aantal veel gebruikte printers, zoals HP Laserjet-printers, en wordt in uw configuratie misschien niet ondersteund.

In dBASE voor Windows kunt u ? toepassen met de optie STYLE voor een fontstijl en de systeemgeheugenvariabele _porientation voor de afdrukstand.

Voorbeeld

In de volgende voorbeelden worden vier verschillende manieren getoond om de code Esc-E naar een HP Laserjet te sturen:

```
??? CHR(27)+"E" && ASCII-code + letter
??? "{ESC}E"    && besturingstekens
??? "{27}{69}"  && alleen ASCII
??? "{27}E"     && ASCII en letters
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

?, _porientation, CLOSE..., SET DEVICE, SET PRINTER

@...CLEAR

Invoer/uitvoer

Wist een deel van het resultatenpaneel in het commandovenster of het huidige dBASE IV-venster. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE om *formulieren* te maken in plaats van dBASE IV-vensters.

Zie Help voor meer informatie over de syntaxis van @...CLEAR. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

@...FILL

Invoer/uitvoer

Bepaalt de kleuren van een rechthoekig deel van het resultatenpaneel in het commandovenster of het huidige dBASE IV-venster. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE om *formulieren* te maken in plaats van dBASE IV-vensters.

Zie Help voor meer informatie over de syntaxis van @...FILL. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

@...SAY...GET

Invoer/uitvoer

Geeft informatie met een opgegeven opmaak weer of accepteert deze op een opgegeven plaats in het resultatenpaneel van het commandovenster of het huidige dBASE IV-venster. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE met de klassen Text en EntryField voor het weergeven en accepteren van informatie op een formulier.

Zie Help voor meer informatie over de syntaxis van @...SAY...GET. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het werken met formulieren in dBASE voor Windows.

@...SCROLL

Invoer/uitvoer

Verschuift de inhoud van een opgegeven deel van het resultatenpaneel in het commandovenster of het huidige dBASE IV-venster naar links, naar rechts, omhoog of omlaag. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows beschikken formulieren over schuifbalken waarmee de gebruiker door een formulier kan bladeren.

Zie Help voor meer informatie over de syntaxis van @...SCROLL. Zie DEFINE en Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het werken met formulieren in dBASE voor Windows.

@...TO

Invoer/uitvoer

Tekent een kader in het resultatenpaneel van het commandovenster of het huidige dBASE IV-venster. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE om *formulieren* te maken in plaats van dBASE IV-vensters.

Zie Help voor meer informatie over de syntaxis van @...TO. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ABS( ) — Numerieke gegevens

Geeft de absolute waarde van het opgegeven getal als resultaat.

Syntaxis

ABS(<*Nuitdr*>)

<Nuitdr>

Het numerieke of zwevende gegevenstype waarvan de absolute waarde als resultaat moet worden gegeven.

Beschrijving

ABS( ) geeft de absolute waarde van een getal (numeriek of zwevend gegevenstype). Het resultaat van ABS( ) is van hetzelfde type als het opgegeven getal.

Voorbeeld

In het volgende voorbeeld wordt ABS( ) gebruikt om de positieve waarde van een getal te berekenen ongeacht het teken:

```
? abs(0.2)      && 0,2
? abs( -5)      && 5
? abs(0.2)<1    && waar, want 0,2 is kleiner dan 1
? abs( -5)<1    && onwaar, want 5 is groter dan 1
```

Zie ook

CEILING( ), FLOOR( ), INT( ), ROUND( )

ACCEPT — Invoer/uitvoer

Accepteert een door de gebruiker ingevoerde tekenreeks en slaat die op in een geheugenvariabele. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE met de klassen Text en EntryField voor het weergeven en accepteren van informatie op een formulier.

Zie Help voor meer informatie over de syntaxis van ACCEPT. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het werken met formulieren in dBASE voor Windows.

ACOPY( ) — Geheugenvariabelen

Kopieert elementen uit een array naar een andere array en geeft het aantal gekopieerde elementen als resultaat.

Syntaxis

ACOPY(<*bronarray-naam*>, <*doelarray-naam*>
[, <*eerste element Nuitdr*> [, <*elementen Nuitdr*> [, <*doelelement Nuitdr*>]]])

<*bronarray-naam*>

De naam van de gedefinieerde array van waaruit elementen worden gekopieerd.

<*doelarray-naam*>

De naam van de gedefinieerde array waar de elementen uit <*bronarray-naam*> naar toe worden gekopieerd.

<*eerste element Nuitdr*>

De positie van het eerste element in <*bronarray-naam*> dat moet worden gekopieerd. Zonder <*eerste element Nuitdr*> kopieert ACOPY( ) alle elementen uit <*bronarray-naam*> naar <*doelarray-naam*>.

<*elementen Nuitdr*>

Het aantal elementen in <*bronarray-naam*> dat moet worden gekopieerd. Zonder <*elementen Nuitdr*> kopieert ACOPY( ) alle elementen uit <*bronarray-naam*> vanaf <*eerste element Nuitdr*> tot het eind van de array. Als u een waarde opgeeft voor <*elementen Nuitdr*>, moet u ook een waarde opgeven voor <*eerste element Nuitdr*>.

<*doelelement Nuitdr*>

De positie in <*doelarray-naam*> waar de elementen naar toe moeten worden gekopieerd. Zonder <*doelelement Nuitdr*> begint ACOPY( ) met kopiëren bij de eerste positie in <*doelarray-naam*>. Als u een waarde opgeeft voor <*doelelement Nuitdr*>, moet u ook waarden opgeven voor <*eerste element Nuitdr*> en <*elementen Nuitdr*>.

Beschrijving

ACOPY( ) kopieert elementen vanuit een bronarray naar een doelarray zonder acht te slaan op de gegevenstypen van de elementen op de opgegeven posities in de doelarray. Als het gegevenstype van een element in de bronarray anders is dan dat van het element op de doelpositie, overschrijft ACOPY( ) de waarde en het gegevenstype op die positie.

Als de doelarray niet groot genoeg is voor alle elementen die uit de bronarray worden gekopieerd, wordt een foutmelding getoond en worden er geen elementen gekopieerd.

Voorbeeld

In het volgende voorbeeld wordt ACOPY( ) gebruikt om een tweede array te maken. het tweede deel van het voorbeeld, dat bestaat uit een tellus en het commando DISPLAY (?), stuurt het resultaat naar het commandovenster:

```
PUBLIC Avan, Anaar, Gekopieerd
SET TALK OFF
DECLARE Avan[4],Anaar[4]
Avan[1] = 1
```

```
Avan[2] = "Twee"
Avan[3] = .t.
Avan[4] = 4
Aant=1
Gekopieerd=ACOPY(Avan,Anaar)
DO WHILE Aant<=4
  ? Anaar[Aant]
Aant=Aant+1
ENDDO
? "Gekopieerde elementen ",Gekopieerd
SET TALK ON
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

AFIELDS( ), ALEN( ), APPEND FROM ARRAY, COPY TO ARRAY, DECLARE, REPLACE FROM ARRAY

ACOS( )

Numerieke gegevens

Geeft de inverse cosinus (arccosinus) van een getal als resultaat.

Syntaxis

ACOS(<*Nuitdr*>)

<Nuitdr>

De cosinus van een hoek, een waarde tussen -1 en +1.

Beschrijving

ACOS( ) geeft als resultaat de grootte in radialen van de hoek waarvan de cosinus gelijk is aan <*Nuitdr*>. Het resultaat is een zwevende waarde van 0 tot pi radialen. Als <*Nuitdr*> gelijk is aan 1 geeft ACOS( ) nul als resultaat. Als *x* een waarde is van 0 tot pi, geeft ACOS(Y) *x* als resultaat als COS(X) gelijk is aan *y*.

Met RTOD( ) kunt u de waarde in radialen converteren naar graden. Als het aantal standaard decimalen bijvoorbeeld 2 is, geeft ACOS(.5) 1,05 radialen als resultaat, terwijl RTOD(ACOS(.5)) 60,00 graden oplevert.

Het aantal decimalen in het resultaat stelt u in met SET DECIMALS.

De arcsecans van een waarde bepaalt u door 1 te delen door de arccosinus van de waarde. De arcsecans van pi is bijvoorbeeld ACOS(1/PI( )) ofwel 1,25 radialen.

Voorbeeld

In het volgende voorbeeld wordt ACOS( ) gebruikt om arccosinus te bepalen van een reeks cosinussen:

```
? ACOS(-1)        &&        3,14
? ACOS( 0)        &&        1,57
? ACOS( 1)        &&        0,00
? RTOD(ACOS(-1)) &&      180,00
? RTOD(ACOS( 0)) &&       90,00
? RTOD(ACOS( 1)) &&        0,00
```

RTOD( ) zet radialen om in graden.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ASIN( ), ATAN( ), ATN2( ), COS( ), DTOR( ), RTOD( ), SET DECIMALS

ACTIVATE MENU

dBASE IV-menu's

Toont een gedefinieerde dBASE IV-menubalk en schakelt die menubalk in. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) om menu's te maken en te activeren bij formulieren.

Zie Help voor meer informatie over de syntaxis van ACTIVATE MENU. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ACTIVATE POPUP

dBASE IV-menu's

Toont een dBASE IV popup-menu en schakelt dat in. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) om menu's bij formulieren te maken en te activeren.

Zie Help voor meer informatie over de syntaxis van ACTIVATE POPUP. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ACTIVATE SCREEN

Invoer/uitvoer

Stuurt uitvoer naar het resultatenpaneel van het commandovenster in plaats van naar het huidige dBASE IV-venster. Dit commando wordt hoofdzakelijk ondersteund

vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) om formulieren te maken en te activeren.

Zie Help voor meer informatie over de syntaxis van ACTIVATE WINDOW. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ACTIVATE WINDOW

dBASE IV-vensters

Opent dBASE IV-vensters en stuurt alle volgende scherminvoer en -uitvoer naar het laatst geopende venster. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) om formulieren te maken en te activeren.

Zie Help voor meer informatie over de syntaxis van ACTIVATE WINDOW. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ADEL( )

Geheugenvariabelen

Verwijdert een element uit een eendimensionale array of verwijdert een rij of een kolom elementen uit een tweedimensionale array. Geeft 1 als resultaat als de bewerking met succes is voltooid en een fout als dat niet zo is.

Syntaxis

ADEL(<*array-naam*>, <*positie Nuitdr*> [, <*rij/kolom Nuitdr*>])

<array-naam>

De naam van een gedefinieerde een- of tweedimensionale array waar gegevens uit moeten worden verwijderd.

<positie Nuitdr>

Als <*array-naam*> een eendimensionale array is, bepaalt <*positie Nuitdr*> het aantal te verwijderen elementen.

Als <*array-naam*> een tweedimensionale array is, bepaalt <*positie Nuitdr*> het nummer van de rij of kolom waarvan u de elementen wilt verwijderen. Het derde argument (dat in de volgende alinea wordt besproken) bepaalt of <*positie Nuitdr*> een rij of een kolom is.

<rij/kolom Nuitdr>

Of 1 of 2. Als u dit argument achterwege laat of 1 opgeeft, wordt uit een tweedimensionale array een rij verwijderd. Als u een 2 opgeeft, wordt een kolom verwijderd. Er wordt een fout als resultaat gegeven als <*rij/kolom Nuitdr*> wordt gebruikt voor een eendimensionale array.

Beschrijving

Gebruik ADEL( ) om bepaalde elementen uit een array te verwijderen zonder de grootte van de array te wijzigen. ADEL( ) heeft de volgende gevolgen:

- Verwijdert een element uit een eendimensionale array of verwijdert een kolom of een rij uit een tweedimensionale array
- Verplaatst de overgebleven elementen in de richting van het begin van de array (omhoog als er een rij is verwijderd of naar links als een element of een kolom is verwijderd)
- Voegt .F. waarden in op de laatste positie(s)

Zie AINS( ) voor meer informatie over het verwijderen van elementen door .F. waarden in te voegen en de overige elementen op te schuiven in de richting van het *eind* van de array. Zie AFILL( ) voor meer informatie over het vervangen van elementen zonder de overige elementen te verplaatsen. Gebruik AGROW( ) of ARESIZE( ) om de grootte van een array te wijzigen.

Eendimensionale array's

Als u ADEL( ) toepast op een eendimensionale array, wordt het element op de opgegeven positie verwijderd en schuiven de overige elementen één positie op in de richting van het begin van de array. De logische waarde .F. wordt opgeslagen in het element op de laatste positie.

Als u bijvoorbeeld een eendimensionale array definieert met DECLARE eenArray[3] en met STORE de waarden "A", "B" en "C" opslaat in de array, heeft de array één rij en ziet die er zo uit:

A B C

Het commando ADEL(eenArray, 2) verwijdert uit de array element nummer 2 (waarde "B"), verplaatst de waarde uit eenArray[3] naar eenArray[2] en slaat .F. op in eenArray[3] zodat de array daarna de volgende waarden bevat:

A C .F.

Tweedimensionale array's

Als u ADEL( ) toepast op een tweedimensionale array, worden de elementen in de opgegeven rij of kolom verwijderd en schuiven de elementen in de overige rijen of kolommen één positie op in de richting van het begin van de array. De logische waarde .F. wordt opgeslagen in de elementen in de laatste rij of kolom.

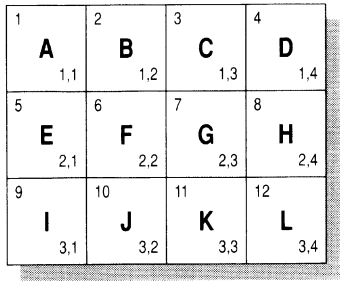
Stel dat u bijvoorbeeld een tweedimensionale array definieert met DECLARE eenArray[3,4] en letters opslaat in de array. De volgende afbeelding laat zien hoe de array wordt gewijzigd door het commando ADEL(eenArray, 2, 2).

Afbeelding 4.1 Gebruik van ADEL() met een twee-dimensionale array

ADEL (eenARRAY, 2, 2)

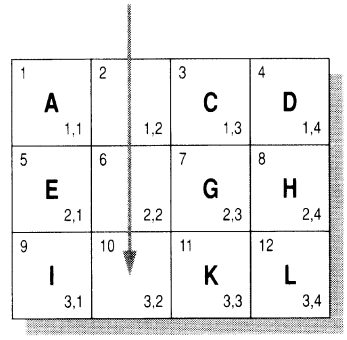
1. Oorspronkelijke array gedeclareerd als:

```
DECLARE eenArray[3,4]
STORE "A" TO eenArray[1,1]
STORE "B" TO eenArray[1,2]
:
STORE "L" TO aArray[3,4]
```

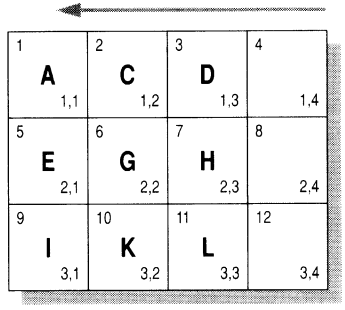


Inhoud van eenArray bij aanvang

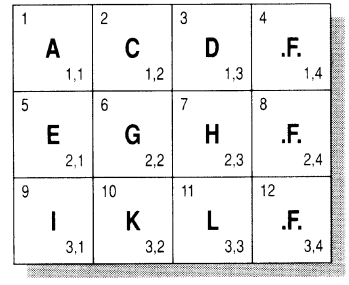
2. ADEL(eenArray,2,2) verwijderd de elementen in de tweede kolom...



3. Verschuift de elementen in de overige kolommen naar het begin van de array...



4. en in de elementen in de laatste kolom worden logische .F.-waarden geschoven met als resultaat:



Inhoud van het array na het comando ADEL(eenArray,2,2)

Voorbeeld

In het volgende voorbeeld worden ADEL( ) en AGROW( ) gebruikt om dynamische elementen toe te voegen aan en te verwijderen uit een array die wordt bewerkt met @ GET-commando's:

```
DECLARE eenTest[3]
AFILL(eenTest, space(10))
@0,10 SAY "  ALT-A = Element toevoegen ;
  ALT-D = Element verwijderen"
ON KEY LABEL ALT-A ArrayGroter()
ON KEY LABEL ALT-D ArrayKleiner()
DO WHILE READKEY() <> 12 .and. eenTest.size > 0
   @1,1 CLEAR TO eenTest.size, 30
   FOR i = 1 to eenTest.size
```

```
      @i, 1 SAY i GET eenTest[i]
   NEXT
   READ
ENDDO
ON KEY LABEL ALT-A
ON KEY LABEL ALT-D
RETURN

FUNCTION ArrayKleiner
   ADEL(eenTest, eenTest.size)
   KEYBOARD CHR(3)
   RETURN .T.

FUNCTION ArrayGroter
   AGROW(eenTest, 1)
   eenTest[eenTest.size] = SPACE(10)
   KEYBOARD CHR(3)
   RETURN .T.
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

AFILL( ), AGROW( ), AINS( ), ARESIZE( ), DECLARE

ADIR( )

Geheugenvariabelen

Slaat in een gedefinieerde array vijf kenmerken op van de opgegeven bestanden: naam, grootte, datumstempel, tijdstempel en DOS-attributen. Als resultaat wordt het aantal bestanden waarvan de kenmerken zijn opgeslagen, gegeven.

Syntaxis

ADIR(*<array-naam>*
   [, *<bestandsnaamfilter Tuitdr>* [, *<DOS-bestandskenmerkenlijst Tuitdr>*]])

<array-naam>

De naam van de gedefinieerde array van één of meer dimensies waarin de informatie over de bestanden moet worden opgeslagen. De grootte van *<array-naam>* wordt dynamisch bepaald, zodat het aantal rijen in de array gelijk is aan het aantal bestanden die overeenkomen met *<DOS-bestandskenmerk Tuitdr>*. Het aantal kolommen bedraagt vijf.

<bestandsnaamfilter Tuitdr>

Het bestandsnaampatroon (met jokers) dat de bestanden beschrijft waarover u informatie wilt opslaan in *<array-naam>*.

<DOS-bestandskenmerkenlijst Tuitdr>

Eén of meer van de D, H, S en/of V voor de gewenste bestandskenmerken.

Als u een waarde wilt opgeven voor *<DOS-bestandskenmerk Tuitdr>*, moet u ook een waarde of "*.*" opgeven voor *<bestandsnaamfilter Tuitdr>*.

De betekenis van de verschillende kenmerkletters is als volgt:

Teken	Betekenis
D	Directory's
H	Verborgen bestanden
S	Systeembestanden
V	Volumelabel (schijfnaam)

Als u meer dan één letter opgeeft voor *<DOS-bestandskenmerk Tuitdr>*, moet u de groep tussen aanhalingstekens plaatsen, zoals in ADIR(eenArray, "*.PRG", "HS").

Beschrijving

Gebruik ADIR( ) om informatie over bestanden op te slaan in een gedefinieerde array, waarvan de grootte dynamisch wordt aangepast zodat alle informatie die als resultaat wordt gegeven, in de array past.

Zonder *<bestandsnaamfilter Tuitdr>* slaat ADIR( ) informatie over alle bestanden in de huidige directory op, met uitzondering van verborgen bestanden en systeembestanden. Wilt u bijvoorbeeld alleen informatie over tabellen, dan geeft u "*.DBF" op voor *<bestandsnaamfilter Tuitdr>*.

Als u ook informatie wilt over directory's, verborgen bestanden en/of systeembestanden, gebruikt u *<DOS-bestandskenmerk Tuitdr>*. Als *<DOS-bestandskenmerk Tuitdr>* de letters D, H en/of S bevat worden ook alle directory's, verborgen bestanden en/of systeembestanden die overeenkomen met *<bestandsnaamfilter Tuitdr>* aan de array toegevoegd.

Als *<DOS-bestandskenmerk Tuitdr>* de letter V bevat, negeert ADIR( ) *<bestandsnaamfilter Tuitdr>* en de overige letters in de attributenlijst en wordt alleen het volumelabel (de schijfnaam) opgeslagen in een array met één element.

ADIR( ) slaat in de array voor elk bestand een rij op met de volgende informatie (het gegevenstype voor elk onderdeel staat tussen haakjes):

Kolom 1	Kolom 2	Kolom 3	Kolom 4	Kolom 5
Bestandsnaam (teken)	Grootte (numeriek)	Datum (datum)	Tijd (teken)	DOS-attributen (teken)

De laatste kolom (DOS-attributen) kan een of meer van de volgende DOS-attributen bevatten:

Attribuut	Betekenis
R	Alleen te lezen bestanden
A	Bestanden waarvan nog geen reservekopie is gemaakt (archiefbit)
S	Systeembestand
H	Verborgen bestand
D	Directory

Een bestand zonder een van de bovengenoemde attributen heeft in kolom 5 bijvoorbeeld de volgende tekenreeks:

```
.....
```

Een uitsluitend leesbaar, verborgen bestand heeft in kolom 5 deze tekenreeks:

```
R..H.
```

Voorbeeld

In het volgende voorbeeld wordt ADIR( ) gebruikt om de bestandsnaam, bestandsgrootte en de datum en tijd van de laatste wijziging van alle .DBF-bestanden in de huidige directory op te slaan in de array Dir_Arr. De tellende DO WHILE-lus geeft de resultaten weer in het resultatenpaneel van het commandovenster:

```
DECLARE Dir_Arr[1]
Aant_Bests=ADIR(Dir_Arr,"*.DBF")
Tel=1
DO WHILE Tel<=Aant_Bests
  ? Dir_Arr[Tel,1], Dir_Arr[Tel,2] AT 20,;
    Dir_Arr[Tel,3] AT 35, Dir_Arr[Tel,4] AT 45,;
    Dir_Arr[Tel,5] AT 55
    Tel=Tel+1
ENDDO
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ACOPY( ), AFIELDS( ), ASORT( ), CD, DIR, DECLARE, FDATE( ), FSIZE( ), FTIME( )

AELEMENT( )

Geheugenvariabelen

Geeft als resultaat het nummer van een opgegeven element in een een- of tweedimensionale array.

Syntaxis

AELEMENT(<*array-naam*>, <*indexteken1 Nuitdr*>
[, <*indexteken2 Nuitdr*>])

<*array-naam*>

Een gedefinieerde een- of tweedimensionale array.

<*indexteken1 Nuitdr*>

Het eerste indexteken van het element. In een eendimensionale array is dit gelijk aan het elementnummer. In een tweedimensionale array is dit de rij.

<*indexteken2 Nuitdr*>

Als <*array-naam*> een tweedimensionale array is, geeft <*indexteken2 Nuitdr*> het tweede indexteken, of de kolom, van het element aan.

Als <*array-naam*> een tweedimensionale array is en u geeft geen waarde op voor <*indexteken2 Nuitdr*>, wordt de waarde 1 gebruikt. Er wordt een fout als resultaat gegeven als u <*indexteken2 Nuitdr*> gebruikt met een eendimensionale array.

Beschrijving

Gebruik AELEMENT( ) als u de indextekens van een element in een tweedimensionale array weet en het elementnummer nodig hebt voor een andere functie, zoals ACOPY( ) of ASCAN( ).

In eendimensionale array's is het nummer van een element gelijk aan het indexteken, dus is het niet nodig om AELEMENT( ) te gebruiken. AELEMENT(eenEenArray,3) geeft 3 als resultaat, AELEMENT(eenEenArray,5) geeft 5 als resultaat, enzovoort.

AELEMENT( ) is het omgekeerde van ASUBSCRIPT( ), dat als resultaat de indextekens (rij en kolom) van een element geeft als u het elementnummer opgeeft.

Voorbeeld

In het eerste deel van dit voorbeeld worden een eendimensionale array en een tweedimensionale array geïnitialiseerd:

```
DECLARE aLeraar[4]
DECLARE aStudent[3,4]
DISPLAY MEMORY
```

De waarden in het geheugen worden geïnitialiseerd als logische waarden en bevatten de waarde .F. let op de rangschikking van de indextekens voor de tweedimensionale array ASTUDENT:

```
*ALERAAR
*           [   1]    L .F.
*           [   2]    L .F.
*           [   3]    L .F.
*           [   4]    L .F.
*ASTUDENT
*       [   1,   1]    L .F.
*       [   1,   2]    L .F.
```

```
*       [  1,  3]    L .F.
*       [  1,  4]    L .F.
*       [  2,  1]    L .F.
*       [  2,  2]    L .F.
*       [  2,  3]    L .F.
*       [  2,  4]    L .F.
*       [  3,  1]    L .F.
*       [  3,  2]    L .F.
*       [  3,  3]    L .F.
*       [  3,  4]    L .F.
```

De volgende instructies gebruiken AELEMENT( ) om het nummer te bepalen van het element met de opgegeven indextekens:

```
? AELEMENT(aLeraar, 3)              && Levert  3
? AELEMENT(aStudent, 1,  2)         && Levert  2
? AELEMENT(aStudent, 2,  2)         && Levert  6
? AELEMENT(aStudent, 3,  4)         && Levert 12
? AELEMENT(aStudent, 3)             && Levert  9
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ACOPY( ), ADEL( ), AFIELDS( ), AINS( ), ALEN( ), ASCAN( ), ASORT( ), ASUBSCRIPT( ), DECLARE

AFIELDS( )

Geheugenvariabelen

Slaat structuurinformatie over de huidige tabel op in een gedefinieerde array en geeft het aantal velden in de array als resultaat.

Syntaxis

AFIELDS(<*array-naam*>)

<array-naam>

De naam van een gedefinieerde array van een of meer dimensies.

Beschrijving

Gebruik AFIELDS( ) om informatie over de structuur van de huidige tabel op te slaan in een gedefinieerde array. U kunt dan verwijzen naar elementen in de array om informatie te krijgen over zaken als een veldnaam en -type. Die informatie kunt u dan gebruiken met andere functies of voor het maken van rapporten. Elke rij in de array bevat informatie over een bepaald veld in de huidige tabel.

AFIELDS( ) bepaalt dynamisch de grootte van <*array-naam*> zodat het aantal rijen in de array minimaal gelijk is aan het aantal velden in de huidige tabel en het aantal

kolommen aan vier. Als u een array hebt gedefinieerd die groter is dan nodig, kan het zijn dat het aantal rijen niet gelijk is aan het aantal velden en dat het aantal kolommen groter is dan vier.

In de volgende tabel ziet u welke veldkenmerken AFIELDS( ) opslaat en in welke kolom van de array de informatie wordt geplaatst:

Kolom 1	Kolom 2	Kolom 3	Kolom 4
Veldnaam (teken)	Veldtype (teken)	Veldlengte (numeriek)	Decimalen (numeriek)

Voor de veldtypen worden de volgende codes gebruikt: B – dBASE of binair Paradox-veld (BLOB), C – teken, D – datum, G – OLE (algemeen), L – logisch, M – memo, N – numeriek, F – zwevend.

AFIELDS( ) slaat dezelfde informatie op in een array die COPY TO...STRUCTURE EXTENDED opslaat in een tabel, behalve dat AFIELDS( ) geen rij maakt met FIELD_IDX-informatie.

Voorbeeld

In het volgende voorbeeld wordt AFIELDS( ) gebruikt om de array Stru_Arr te initialiseren met de structuur van de tabel Bedrijf. De tweedimensionale array heeft vier kolommen met respectievelijk de veldnaam, het veldtype, de veldlengte en het aantal decimale posities, en zoveel rijen als de tabel velden heeft. De volgende DO WHILE-lus toont alleen de eerste kolom, dat wil zeggen de veldnamen in de huidige tabel:

```
USE BEDRIJF
DECLARE Stru_Arr[1]
Aant_Velden=AFIELDS(Stru_Arr)
Tel1=1
DO WHILE Tel1<=Aant_Velden
    ? Stru_Arr[Tel1,1]
    Tel1=Tel1+1
ENDDO
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

COPY TO ARRAY, COPY TO...STRUCTURE EXTENDED, DECLARE, FDATE( ), FSIZE( ), FTIME( )

AFILL( ) Geheugenvariabelen

Voegt een opgegeven waarde in op een of meer posities in een gedefinieerde array en geeft het aantal ingevoegde elementen als resultaat.

Syntaxis

AFILL(<*array-naam*>, <*uitdr*>
[, <*begin Nuitdr*> [, <*aantal Nuitdr*>]])

<*array-naam*>

De naam van de gedefinieerde een- of tweedimensionale array waarin de opgegeven waarde <*uitdr*> moet worden ingevoegd.

<*uitdr*>

Een uitdrukking van het gegevenstype teken, datum, logisch, numeriek of zwevend dat in de opgegeven array moet worden ingevoegd.

<*begin Nuitdr*>

Het nummer van het element waar met invoegen moet worden begonnen. Als u geen <*begin Nuitdr*> opgeeft, wordt bij het eerste element in de array begonnen.

<*aantal Nuitdr*>

Het aantal elementen waarin <*uitdr*> moet worden ingevoegd, te beginnen bij het element <*begin Nuitdr*>. Als u geen <*aantal Nuitdr*> opgeeft, wordt <*uitdr*> ingevoegd vanaf <*begin Nuitdr*> tot het laatste element in de array. Als u een waarde wilt opgeven voor <*aantal Nuitdr*>, moet u ook een waarde opgeven voor <*begin Nuitdr*>.

Als u noch <*begin Nuitdr*> noch <*aantal Nuitdr*> opgeeft, worden alle elementen in de array gevuld met <*uitdr*>.

Beschrijving

Met AFILL( ) voegt u een waarde in in alle of een aantal elementen van een gedefinieerde array. Als u bijvoorbeeld elementen van een array wilt gebruiken om totalen te berekenen, kunt u AFILL( ) gebruiken om alle waarden in de array te initialiseren met de waarde 0.

AFILL( ) voegt sequentieel waarden in in een array. Vanaf het begin van de array of vanaf <*begin Nuitdr*> voegt AFILL( ) waarden in in elk element in een rij en gaat vervolgens verder met het eerste element in de volgende rij tot de gehele array is gevuld of <*aantal Nuitdr*> elementen zijn ingevoegd. Bestaande gegevens in de array worden overschreven.

Als u de indextekens van een element weet, kunt u AELEMENT( ) gebruiken om het nummer van dat element te bepalen zodat u dat voor <*begin Nuitdr*> kunt gebruiken.

Voorbeeld

In het volgende voorbeeld wordt AFILL( ) gebruikt om de huidige waarde voor VerkTotNu in de tiende kolom van de array Bed_Arr te vervangen. ASCAN( ) geeft als resultaat het elementnummer voor de gewenste bedrijfsnaam die als referentiepunt wordt gebruikt door AFILL( ):

```
SET TALK OFF
CLEAR
```

```
USE Bedrijf
Opzoeken="InterSafe"
Verkoop=143325552.20
Tel=RECCOUNT()
Vldn=FLDCOUNT()
DECLARE Bed_Arr[Tel,Vldn]
COPY TO ARRAY Bed_Arr
Element=ASCAN(Bed_Arr,Opzoeken)
IF Element>0
  Vervang=AFILL(Bed_Arr,Verkoop,Element+9,1)
ENDIF
Telling=1
DO WHILE Telling<=Tel
  ? Bed_Arr[Telling,1], Bed_Arr[Telling,10]
  Telling=Telling+1
ENDDO
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ADEL( ), AELEMENT( ), AINS( ), DECLARE

AGROW( )

Geheugenvariabelen

Voegt een element, rij of kolom toe aan een array en geeft het aantal toegevoegde elementen als resultaat.

Syntaxis

AGROW (*<array-naam>*, *<Nuitdr>*)

<array-naam>

De naam van de gedefinieerde een- of tweedimensionale array waaraan u elementen wilt toevoegen.

<Nuitdr>

Of 1 of 2. Als u 1 opgeeft, voegt AGROW( ) een enkel element toe aan een eendimensionale array of een rij aan een tweedimensionale array. Als u 2 opgeeft, voegt AGROW( ) een kolom toe aan de array.

Beschrijving

Met AGROW( ) voegt u een element, rij of kolom toe aan een array. De grootte van de array wordt aangepast op basis van de toegevoegde elementen. AGROW( ) kan een tweedimensionale array maken van een eendimensionale. Alle toegevoegde elementen worden geïnitialiseerd met .F. waarden.

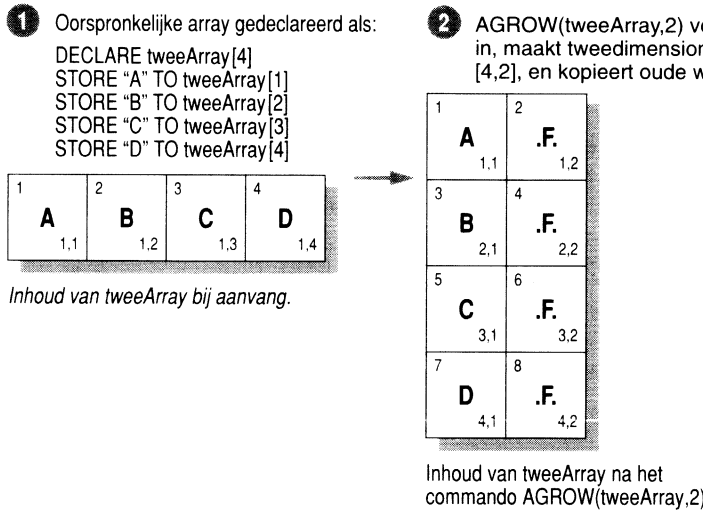
Gebruik AINS( ) om .F. waarden in te voegen zonder de grootte van de array te wijzigen.

Eendimensionale array's

Als u 1 opgeeft voor *<Nuitdr>*, voegt AGROW( ) een enkel element toe aan de array. Als u 2 opgeeft, maakt AGROW( ) de array tweedimensionaal en worden bestaande elementen in de eerste kolom geplaatst. In de volgende afbeelding ziet u hoe dat in zijn werk gaat:

Afbeelding 4.2 Een eendimensionale array tweedimensionaal maken met AGROW

AGROW(tweeARRAY,2)



Tweedimensionale array's

Als u 1 opgeeft voor *<Nuitdr>*, voegt AGROW( ) een rij toe aan het eind van de array. In de volgende afbeelding ziet u hoe dat in zijn werk gaat:

Afbeelding 4.3 Met AGROW een regel toevoegen aan een tweedimensionale array met gebruikmaking van AGROW)

AGROW (eenARRAY,1)

1. Oorspronkelijke array gedeclareerd als:

```
DECLARE eenArray[3,4]
STORE "A" TO eenArray[1,1]
STORE "B" TO eenArray[1,2]
:
STORE "L" TO eenArray[3,4]
```

1 A 1,1	2 B 1,2	3 C 1,3	4 D 1,4
5 E 2,1	6 F 2,2	7 G 2,3	8 H 2,4
9 I 3,1	10 J 3,2	11 K 3,3	12 L 3,4

Inhoud van eenArray bij aanvang.

2. AGROW(eenARRAY,1) voegt nieuwe rij toe aan array.

1 A 1,1	2 B 1,2	3 C 1,3	4 D 1,4
5 E 2,1	6 F 2,2	7 G 2,3	8 H 2,4
9 I 3,1	10 J 3,2	11 K 3,3	12 L 3,4
13 .F. 4,1	14 .F. 4,2	15 .F. 4,3	16 .F. 4,4

Inhoud van de array na commando AGROW(eenArray,1)

Als u 2 opgeeft voor <*Nuitdr*>, voegt AGROW( ) een kolom toe aan de array en wordt in elk element in de kolom de waarde .F. geplaatst.

Voorbeeld

Het volgende voorbeeld begint met het definiëren van een array met drie elementen en gebruikt vervolgens AGROW( ) om een vierde element toe te voegen. Daarna wordt een kolom toegevoegd en tenslotte wordt een rij toegevoegd aan de tweedimensionale array. Met DISPLAY MEMORY worden na elke bewerking met AGROW( ) de waarden in de array weergegeven:

```
RELEASE ALL
DECLARE A[3]
A[1]="x"
A[2]="y"
A[3]="z"
DISPLAY MEMORY
N=AGROW(A,1)      && Element toevoegen aan A
DISPLAY MEMORY
N=AGROW(A,2)      && Kolom toevoegen aan A
DISPLAY MEMORY
N=AGROW(A,1)      && Rij toevoegen aan A
DISPLAY MEMORY
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

AINS( ), ALEN( ), ARESIZE( ), DECLARE

AINS( )

Geheugenvariabelen

Voegt in een eendimensionale array een element in met de waarde .F. of voegt in een tweedimensionale array een rij of kolom elementen in met de waarde .F.. Geeft 1 als resultaat als de bewerking met succes is uitgevoerd, anders een fout.

Syntaxis

AINS(<*array-naam*>, <*positie Nuitdr*> [, <*rij/kolom Nuitdr*>])

<array-naam>

De naam van de gedefinieerde een- of tweedimensionale array waarin gegevens moeten worden ingevoegd.

<positie Nuitdr>

Als <*array-naam*> een eendimensionale array is, bepaalt <*positie Nuitdr*> het nummer van het element waar de waarde .F. moet worden ingevoegd.

Als <*array-naam*> een tweedimensionale array is, bepaalt <*positie Nuitdr*> het nummer van de rij of kolom waarin .F. waarden moeten worden ingevoegd. Het derde argument (dat in de volgende alinea wordt besproken) bepaalt of <*positie Nuitdr*> een rij of een kolom is.

<rij/kolom Nuitdr>

Of 1 of 2. Als u dit argument achterwege laat of 1 opgeeft, wordt in een tweedimensionale array een rij ingevoegd. Geeft u 2 op, dan wordt een kolom ingevoegd. Er wordt een fout als resultaat gegeven als u <*rij/kolom Nuitdr*> gebruikt met een eendimensionale array.

Beschrijving

Met AINS( ) kunt u .F. waarden invoegen in bepaalde elementen in een array zonder de grootte van de array te wijzigen. AINS( ) doet het volgende:

- Voegt een element in in een eendimensionale array of een rij of kolom in een tweedimensionale array
- De overige elementen worden in de richting van het eind van de array opgeschoven (in het geval van een rij omlaag en in het geval van een element of kolom naar rechts)
- In de nieuwe posities worden .F. waarden geplaatst

Zie ADEL( ) voor meer informatie over het invoegen van elementen door de overige elementen te verschuiven in de richting van het *begin* van de array en het invoegen van .F. waarden aan het eind van de array. Zie AFILL( ) voor meer informatie over het

vervangen van elementen zonder de overige elementen te verschuiven. Met AGROW( ) of ARESIZE( ) kunt u een eendimensionale array tweedimensionaal maken.

Eendimensionale array's

Als u AINS( ) toepast op een eendimensionale array, wordt op de plaats van het opgegeven element de logische waarde .F. ingevoegd. De overige elementen worden één plaats in de richting van het eind van de array opgeschoven. Het element op de laatste positie wordt verwijderd.

Als u bijvoorbeeld met DECLARE de eendimensionale array eenArray[3] definieert en in de array de waarden "A", "B" en "C" opslaat, bestaat de array uit één rij die als volgt kan worden weergegeven:

A B C

AINS(eenArray, 2) plaatst de waarde .F. in eenArray[2], verplaatst de oorspronkelijke waarde van eenArray[2] (de waarde "B") naar eenArray[3]. De oorspronkelijke waarde "C" in eenArray[3] verdwijnt. De array bevat nu deze waarden:

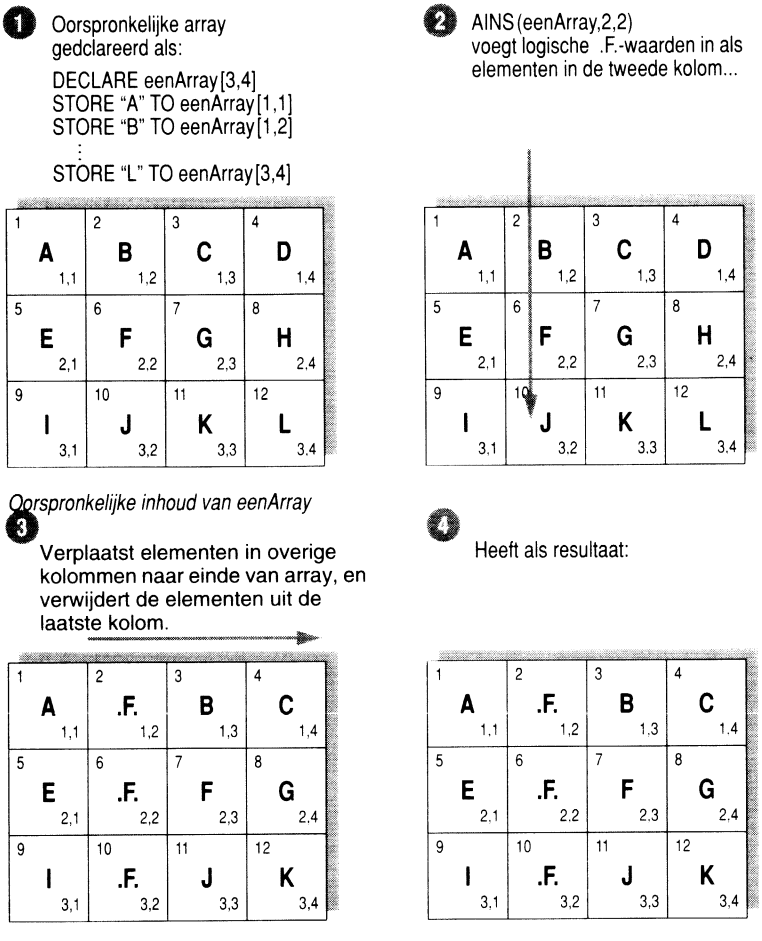
A .F. B

Tweedimensionale array's

Als u AINS( ) toepast op een tweedimensionale array, wordt de logische waarde .F. ingevoegd in elke positie in de opgegeven rij of kolom. De elementen in de overige kolommen worden één positie in de richting van het eind van de array opgeschoven. De elementen in de oorspronkelijk laatste rij of kolom gaan verloren.

Stel dat u bijvoorbeeld een tweedimensionale array definieert met DECLARE aArr[3,4] en letters opslaat in de array. In de volgende afbeelding ziet u hoe de array wordt veranderd door het commando AINS(eenArray, 2,2):

Afbeelding 4.4 AINS() toepassen op een tweedimensionale array



Voorbeeld

In het volgende voorbeeld wordt een tweedimensionale array gebruikt die als volgt is gedefinieerd:

```
PUBLIC aAlpha
DECLARE aAlpha[2,3]
STORE "een" TO aAlpha[1,1]
STORE "twee" TO aAlpha[1,2]
STORE "drie" TO aAlpha[1,3]
STORE "vier" TO aAlpha[2,1]
STORE "vijf" TO aAlpha[2,2]
STORE "zes" TO aAlpha[2,3]
```

De array aAlpha heeft nu de volgende inhoud:

```
* aAlpha
*  [1,1]    C    "een"
*  [1,2]    C    "twee"
*  [1,3]    C    "drie"
*  [2,1]    C    "vier"
*  [2,2]    C    "vijf"
*  [2,3]    C    "zes"
```

Nu wordt AINS( ) gebruikt om de eerste kolom te vullen met .F. en de overige elementen in op te schuiven in de richting van het eind van de array:

```
? AINS(aAlpha,1,2)  && Geeft bij succes 1 als resultaat
```

aAlpha heeft nu de volgende inhoud:

```
* aAlpha
*  [1,1]    C    .F.
*  [1,2]    C    "een"
*  [1,3]    C    "twee"
*  [2,1]    C    .F.
*  [2,2]    C    "vier"
*  [2,3]    C    "vijf"
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ADEL( ), AFILL( ), AGROW( ), ARESIZE( ), DECLARE

ALEN( )

Geheugenvariabelen

Geeft het aantal elementen, rijen of kolommen in een array als resultaat.

Syntaxis

ALEN(<*array-naam*> [, <*Nuitdr*>])

<array-naam>

De naam van de gedefinieerde een- of tweedimensionale array.

<Nuitdr>

Het getal 0, 1 of 2. Hiermee geeft u aan welk aantal u als resultaat wilt hebben: elementen, rijen of kolommen. Er wordt een fout als resultaat gegeven als u <*Nuitdr*> gebruikt voor een array met meer dan twee dimensies.

In de volgende tabel ziet u wat ALEN( ) als resultaat geeft voor de verschillende waarden voor <*Nuitdr*>:

<Nuitdr>...	**Resultaat ALEN( )**
niet opgegeven	Het aantal elementen in de array
0	Het aantal elementen in de array
1	Bij een eendimensionale array: het aantal elementen in de array
	Bij een tweedimensionale array: het aantal rijen (het eerste indexteken in de definitie van de array)
2	Bij een eendimensionale array: 0 (nul)
	Bij een tweedimensionale array: het aantal kolommen (het tweede indexteken in de definitie van de array)
een andere waarde	0 (nul)

Beschrijving

Met ALEN( ) kunt u de afmetingen van een gedefinieerde array bepalen: het aantal elementen of het aantal rijen of kolommen.

Als u zowel het aantal rijen als het aantal kolommen van een tweedimensionale array wilt weten, geeft u het commando ALEN( ) twee keer, een keer met 1 voor <*Nuitdr*> en een keer met 2 voor <*Nuitdr*>. Met de volgende commando's bepaalt u bijvoorbeeld het aantal rijen en kolommen in eenArray:

```
? ALEN(eenArray,1) && aantal rijen als resultaat
? ALEN(eenArray,2) && aantal kolommen als resultaat
```

Voorbeeld

In het volgende voorbeeld wordt ALEN( ) gebruikt in een FOR...NEXT-lus om de inhoud van een array af te drukken:

```
USE Klanten
DECLARE acContact[RECCOUNT(),1]
COPY TO ARRAY acContact
USE
ToonArray(acContact)
RETURN

FUNCTION ToonArray
PARAMETER avArray
FOR i = 1 to ALEN(avArray)
  ? avArray[i]
NEXT
RETURN .T.
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ADEL( ), AELEMENT( ), AFIELDS( ), AGROW( ), AINS( ), ARESIZE( ), ASCAN( ), ASUBSCRIPT( ), DECLARE

ALIAS( ) Tabellen

Geeft als resultaat de aliasnaam van het huidige of een opgegeven werkgebied. Als een werkgebied geen geopende tabel bevat, geeft ALIAS( ) een lege tekenreeks ("") als resultaat.

Syntaxis

ALIAS([<*alias*>])

<alias>

Het werkgebied waarvan u de aliasnaam wilt weten. U kunt een werkgebiednummer (1 tot en met 255), -letter (A tot en met J) of een werkgebiednaam opgeven. De werkgebiednaam kan een tabelnaam zijn of een aliasnaam die is opgegeven met het commando USE. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

ALIAS( ) geeft als resultaat de aliasnaam van een werkgebied in de huidige sessie (als **Sessies** is ingeschakeld in het dialoogvenster **Kenmerken bureaublad**). Als u geen werkgebied opgeeft, wordt aangenomen dat u het huidige werkgebied bedoeld. Als in het opgegeven werkgebied geen tabel is geopend, geeft ALIAS( ) een lege tekenreeks ("") als resultaat.

Voorbeeld

In het volgende voorbeeld wordt ALIAS( ) om de naam (of het alias) als resultaat te geven van de tabel die wordt gebruikt in werkgebied 20:

```
USE Klanten ALIAS Klanten2 IN 20
? ALIAS(20)            && Geeft Klanten2
? ALIAS("Klanten2")    && Geeft Klanten2
SELECT 20
? ALIAS()              && Geeft Klanten2
CLOSE DATABASES
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

DBF( ), SELECT( ), USE, WORKAREA( )

ANSI( )

Tekenreeksgegevens

Geeft als resultaat een tekenreeks die de ANSI-waarde is van een OEM-tekenuitdrukking. ANSI staat voor American National Standards Institute en OEM is een afkorting van Original Equipment Manufacturer.

Syntaxis

ANSI(<*Tuitdr*>)

<*Tuitdr*>

De OEM-tekenuitdrukking die moet worden geconverteerd naar ANSI-tekens.

Beschrijving

Met ANSI( ) en OEM( ) kunt u tekens converteren van en naar de ANSI- en de OEM-tekenset. ANSI( ) krijgt een OEM-tekenuitdrukking als argument en zet de tekens in die uitdrukking om in tekens uit de huidige ANSI-tekenset, terwijl OEM( ) een ANSI-tekenuitdrukking als argument krijgt en de tekens in die uitdrukking omzet in tekens uit de huidige OEM-tekenset. Zie Appendix C in *Programmeren* voor meer informatie over tekensets.

In dBASE wordt een OEM-tekenset gebruikt, terwijl andere Windows-toepassingen een ANSI-tekenset gebruiken. De meeste alfabetische en numerieke tekens in OEM-tekensets komen overeen met die in ANSI-sets, maar tussen de tekens met hoge ASCII-waarden (van 128 tot en met 255) bestaan verschillen. Het kan gebeuren dat u ANSI( ) moet gebruiken om een tekenreeks vanuit dBASE naar een andere Windows-toepassing te sturen, bijvoorbeeld als u het commando EXTERN toepast.

Als u een tekenreeks wilt versturen met verschillende ANSI- en OEM-waarden, gebruikt u ANSI( ). Als u een tekenreeks ontvangt van een Windows-toepassing, gebruikt u OEM( ) om de reeks te converteren naar het type dat door dBASE wordt gebruikt.

Het volgende voorbeeld laat zien hoe u kunt bepalen of twee tekens dezelfde of verschillende ANSI- en OEM-waarden hebben.

Voorbeeld

Het volgende voorbeeld geeft de 255 tekens weer die mogelijk zijn met de ASCII-, ANSI- en OEM-tekensets:

```
FOR i=1 to 255
  ASCII=CHR(i)
  ?? i,ASCII,ANSI(ASCII),OEM(ASCII)
  * De volgende 3 commando's zorgen voor een pauze
  * na elke 10 regels
  IF MOD(i,10)=0
    WAIT
  ENDIF
NEXT i
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CHR( ), EXTERN, OEM( )

APPEND

Velden en records

Voegt nieuwe records toe aan een tabel.

Syntaxis

APPEND
[BLANK]
[NOWAIT]

BLANK

Voegt een leeg record toe aan het end van de tabel en maakt dat lege record het huidige record.

NOWAIT

Roept het formulier aan om een nieuw record te bewerken, maar verplaatst de focus daar niet naar toe. Als dit commando in een programma wordt gebruikt, gaat de uitvoering verder met de instructie na het commando APPEND NOWAIT.

Beschrijving

APPEND geeft het venster **Tabelrecords** weer in formulierweergave. In dit venster kunt u naar bestaande records gaan, bestaande records bekijken en bewerken en tevens nieuwe records toevoegen. Zie het *Handboek* voor meer informatie over het bewerken van gegevens en verplaatsing in formulieren.

Als u een indelingsbestand kiest met SET FORMAT, geeft APPEND recordgegevens weer aan de hand van het indelingsbestand. Als er geen indeling is opgegeven, wordt een standaardformulier weergegeven. Het standaardformulier toont één record per keer, met elk veld van links naar rechts.

Het commando APPEND BLANK voegt een leeg record toe aan de huidige tabel en plaatst de recordaanwijzer naar het nieuwe record, maar toont geen venster voor het bewerken van de gegevens. In het geval van SQL-tabellen staan sommige database-servers u niet toe lege records in te voegen. Verder kunnen beperkingen voor tabellen die zijn gemaakt met niet-null velden voorkomen dat records met lege velden worden ingevoegd. Zie de documentatie bij Borland SQL Link voor uw specifieke database-server voor meer informatie.

Het invoeren van gegevens in een toegevoegd record en vervolgens de recordaanwijzer voorbij het eind van het record verplaatsen, heeft tot gevolg dat APPEND een volgend

record toevoegt (dat geldt niet voor APPEND BLANK). Als SET CARRY is uitgeschakeld (OFF) wordt elk nieuw toegevoegde record weergegeven met lege velden. Als SET CARRY is ingeschakeld (ON), bevat elk nieuw toegevoegde record een kopie van de gegevens uit het voorgaande record.

Het commando APPEND gebruikt alleen de velden die zijn opgegeven met SET FIELDS TO, als dat commando is gebruikt. Als dBASE-tabellen zijn gekoppeld met het commando SET RELATION, besturen de opties CONSTRAIN en INTEGRITY bewerkingen die nieuwe records toevoegen aan hoofd- en subtabellen. Zie het commando SET RELATION voor meer informatie.

APPEND werkt automatisch geopende indexbestanden bij als u records toevoegt. APPEND geeft records in een tabel met een hoofdindex geïndexeerd weer en voegt nieuwe records toe aan het eind van de tabel.

Voorbeeld

In het volgende voorbeeld wordt APPEND BLANK gebruikt om een leeg record toe te voegen aan het eind van de natuurlijke-volgordetabel:

```
USE Klanten
? RECCOUNT()                && Geeft aantal records in;
                               tabel als resultaat
APPEND BLANK
? RECCOUNT()                && Geeft vorige nummer;
                               plus 1 als resultaat
EDIT                        && Opent nieuw record;
                               voor gegevensinvoer
```

In het volgende voorbeeld wordt APPEND gebruikt om een leeg record toe te voegen aan het eind van de huidige tabel en een bewerkingsvenster te openen zonder het commando EDIT te gebruiken zoals in het vorige voorbeeld:

```
USE Klanten
APPEND
```

Zie ook

APPEND AUTOMEM, APPEND FROM, BROWSE, CLASS BROWSE, EDIT, SET CARRY, SET FORMAT, SET RELATION, SET WINDOW OF MEMO

APPEND AUTOMEM

Velden en records

Voegt een nieuw record toe aan een tabel met gebruik van de waarden die zijn opgeslagen in automatische geheugenvariabelen.

Syntaxis

APPEND AUTOMEM

Beschrijving

APPEND AUTOMEM voegt een nieuw record toe aan een tabel en vervangt vervolgens de waarde in de velden in de tabel door de inhoud van de bijbehorende automatische geheugenvariabelen. Automatische geheugenvariabelen zijn variabelen die dezelfde naam en hetzelfde gegevenstype hebben als de velden in de huidige tabel.

Het commando APPEND toont een weergave waarin het mogelijk is interactief records toe te voegen aan een tabel, maar APPEND AUTOMEM levert in een programma mogelijkheden voor de besturing van gegevensinvoer. U kunt het bewerkingsvenster aanpassen en de geldigheid van gegevens controleren voordat ze aan de tabel worden toegevoegd. APPEND AUTOMEM is ook een doelmatiger manier om records toe te voegen aan een tabel dan APPEND BLANK en REPLACE.

Om APPEND AUTOMEM te kunnen gebruiken voor het toevoegen van records aan een tabel moet eerst een verzameling automatische geheugenvariabelen worden gemaakt. Het commando USE...AUTOMEM opent een tabel en maakt de bijbehorende lege automatische geheugenvariabelen voor die tabel. CLEAR AUTOMEM maakt een verzameling lege automatische geheugenvariabelen voor de huidige tabel of initialiseert bestaande automatische geheugenvariabelen met lege waarden.

Als u in commando's die dat mogelijk maken, automatische geheugenvariabelen opgeeft (zoals in het commando @...SAY...GET), moet u de naam van een automatische geheugenvariabele vooraf laten gaan door m-> om die variabele te onderscheiden van velden met dezelfde naam. @ 0,0 GET Prijs geeft bijvoorbeeld de waarde in het veld Prijs van het huidige record weer, terwijl @ 0,0 GET m->Prijs de waarde van de automatische geheugenvariabele met de naam Prijs toont.

Voorbeeld

In het volgende voorbeeld wordt APPEND AUTOMEM gebruikt om een nieuw record toe te voegen aan de tabel Klanten, waarbij de waarden in de velden afkomstig zijn uit automatische geheugenvariabelen. In dit geval zal het nieuwe record leeg zijn omdat vlak voor APPEND AUTOMEM het commando CLEAR AUTOMEM is gegeven:

```
SET TALK OFF
STORE " " TO Antwoord
USE Klanten
? RECCOUNT()
DO WHILE UPPER(Antwoord) <> "S"
   CLEAR AUTOMEM
   APPEND AUTOMEM
   ACCEPT "Druk op S om te stoppen of Enter om verder te gaan ";
   to Antwoord
   ? RECCOUNT()
ENDDO
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

APPEND, CLEAR AUTOMEM, INSERT AUTOMEM, REPLACE AUTOMEM, STORE AUTOMEM, USE

APPEND FROM

Tabellen

Kopieert records uit een bestaande tabel naar het eind van de huidige tabel.

Syntaxis

APPEND FROM *<bestandsnaam>* | ? | *<bestandsnaamfilter>*
[FOR *<voorwaarde>*]
[[TYPE] SDF | PARADOX | DBASE |
DELIMITED [WITH
<teken> | BLANK]]
[NOVERIFY]
[POSITION]
[REINDEX]

<bestandsnaam> | ? | *<bestandsnaamfilter>*

De naam van de tabel waaruit u records wilt toevoegen aan de huidige tabel. APPEND FROM ? en APPEND FROM *<bestandsnaamfilter>* tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand eerst gezocht in de huidige directory en vervolgens in het pad dat u opgeeft met SET PATH. Als u een bestand zonder extensie opgeeft, wordt het standaard tabeltype gebruikt dat is opgegeven met het commando SET DBTYPE.

U kunt ook records toevoegen uit een tabel in een database (gedefinieerd met de IDAPI Configuration Utility) door voor de naam van de tabel de naam van de database op te geven (tussen dubbele punten): *:database-naam:tabelnaam*. Als de database nog niet open is, wordt een dialoogvenster getoond waarin u de parameters opgeeft die nodig zijn om een verbinding met de database tot stand te brengen, zoals een gebruikersnaam en wachtwoord.

FOR *<voorwaarde>*

Beperkt APPEND FROM tot records in *<bestandsnaam>* die voldoen aan *<voorwaarde>*. U mag FOR *<voorwaarde>* alleen gebruiken voor velden die bestaan in de huidige tabel.

[TYPE] SDF | PARADOX | DBASE | DELIMITED [WITH *<teken>* | BLANK]

Bepaalt de indeling van de gegevens die u toevoegt. Het sleutelwoord TYPE is er alleen voor de leesbaarheid; het heeft geen gevolgen voor de werking van het commando. De volgende tabel geeft een overzicht van de ondersteunde bestandsindelingen:

Type	Omschrijving
SDF	System Data Format. Records in een SDF-bestand hebben een vaste lengte en het eind van een record wordt aangegeven door een regelterugloop en een regeldoorvoer. In de doeltabel worden ongebruikte delen van velden opgevuld met spaties. Als u geen extensie opgeeft, wordt .TXT gebruikt.
PARADOX	Een Paradox-tabel (met de extensie .DB). Paradox-rijen worden gekopieerd naar dBASE-records en elke Paradox-kolom wordt naar een dBASE-veld gekopieerd.
DBASE	Een dBASE-tabel. Als u geen extensie opgeeft voor *<bestandsnaam>*, wordt de extensie .DBF gebruikt.
DELIMITED	Een bestand met tekstindeling. Gegevens in de volgende indelingen worden door dBASE voor Windows als volgt vertaald naar de huidige tabel:
	Gegevens die worden gescheiden door aanhalingstekens of door het teken dat u opgeeft met WITH *<teken>* wordt toegevoegd aan tekenvelden.
	Gegevens met de opmaak JJJJMMDD worden toegevoegd aan datumvelden.
	Gegevens die uitsluitend bestaan uit het teken T of F wordt toegevoegd aan logische velden.
	Getallen worden toegevoegd aan numerieke velden.
	Elke regelterugloop en regeldoorvoer geven een nieuw record aan.
	WITH *<teken>*
	Geeft aan dat tekengegevens worden gescheiden door het teken *<teken>* in plaats van door aanhalingstekens.
	WITH BLANK
	Geeft aan dat gegevens in *<bestandsnaam>* worden gescheiden doorspaties in plaats van komma's of andere scheidingstekens. Als u DELIMITED opgeeft zonder WITH BLANK, neemt dBASE voor Windows aan dat de gegevens in *<bestandsnaam>* worden gescheiden door komma's.

NOVERIFY

Schakelt de controle op de geldigheid van gegevens uit tijdens APPEND FROM van SDF-bestanden om de verwerking sneller te maken. Als u NOVERIFY gebruikt, worden de brongegevens rechtstreeks gekopieerd naar de doelvelden. Zonder controle op de geldigheid van gegevens worden door dBASE voor Windows geen brongegevens gecontroleerd en geconverteerd om ze aan te passen aan het gegevenstype van de doelvelden.

POSITION

Voegt velden toe op basis van hun relatieve positie in de tabel in plaats van op basis van overeenkomstige veldnamen.

REINDEX

Stelt alle open indexbestanden opnieuw samen nadat APPEND FROM is voltooid. Zonder REINDEX werkt dBASE alle open indexen telkens bij nadat een record uit *<bestandsnaam>* is toegevoegd. Als de huidige tabel over meerdere open

indexbestanden beschikt of veel records bevat, wordt APPEND FROM sneller uitgevoerd als de optie REINDEX is gebruikt.

Beschrijving

Met het commando APPEND FROM kunt u gegevens uit een ander bestand of een andere tabel toevoegen aan het eind van de huidige tabel. U kunt gegevens toevoegen uit dBASE-tabellen en uit bestanden met een andere indeling. De gegevens worden aan de huidige tabel toegevoegd in de volgorde waarin ze in het opgegeven bestand zijn opgeslagen.

Als u een dBASE-tabel opgeeft als bron van de gegevens, wordt alleen de inhoud van de velden waarvan de namen en typen overeenkomen met de huidige tabel toegevoegd. In dBASE voor Windows worden datumgegevens echter toegevoegd aan een tekenveld met dezelfde naam en tekengegevens met een datumopmaak aan een datumveld met dezelfde naam.

Als de velden in de huidige tabel en de brontabel niet dezelfde breedte hebben, wordt een van de volgende oplossingen toegepast:

- Als het veld in de huidige tabel langer is dan de bijbehorende brongegevens, worden de gegevens aangevuld met spaties.
- Als het veld in de huidige tabel korter is dan is de bijbehorende brongegevens, worden de gegevens afgekapt.

Als SET DELETED is uitgeschakeld (OFF), voegt dBASE records uit een dBASE-brontabel ook toe als die zijn gemarkeerd voor verwijdering. Deze records worden in de huidige tabel *niet* gemarkeerd voor verwijdering. Als SET DELETED is ingeschakeld (ON), worden geen records uit de brontabel toegevoegd die zijn gemarkeerd voor verwijdering.

Als u gegevens importeert uit bestanden met een andere indeling, moet u er op letten kolomkoppen en lege rijen en kolommen aan het begin te verwijderen, want anders worden die ook toegevoegd.

Voorbeeld

In het volgende voorbeeld wordt APPEND FROM gebruikt om records toe te voegen aan een lege tabel vanuit een bestand met een SDF-indeling (een opgemaakt tekstbestand, .TXT-bestand):

```
USE Contact ORDER CompCode IN SELECT()
USE Bedrijf IN SELECT()
SELECT Bedrijf
SET RELATION TO CompCode INTO Contact
COPY STRUCTURE TO CntctLst;
  FIELDS Bedrijf->Bedrijf, ;
  Contact->CompCode, Contact->Contact, ;
  Bedrijf->Straat1, Bedrijf->Straat2, ;
  Bedrijf->Plaats, Bedrijf->Regio,;
  Bedrijf->Postcode
COPY TO CntctLst.TXT TYPE SDF;
  FIELDS Bedrijf->Bedrijf, ;
```

```
    Contact->CompCode, Contact->Contact, ;
    Bedrijf->Straat1, Bedrijf->Straat2, ;
    Bedrijf->Plaats, Bedrijf->Regio,;
    Bedrijf->Postcode
CLOSE ALL
USE CntctLst
APPEND FROM CntctLst.TXT TYPE SDF
CLOSE ALL
```

In het volgende voorbeeld wordt APPEND FROM gebruikt om records toe te voegen aan de tabel klanten vanuit een tekstbestand met aanhalingstekens (") als scheidingstekens (de velden in POSTLST.TXT hebben dezelfde volgorde als die in KLANTEN):

```
USE Klanten
COPY TO PostLst.TXT TYPE DELIMITED
APPEND FROM PostLst.TXT TYPE DELIMITED
```

Zie ook

APPEND, APPEND AUTOMEM, COPY, IMPORT, REINDEX, SET DELETED

APPEND FROM ARRAY

Velden en records

Voegt aan de huidige tabel een of meer records toe met gegevens die zijn opgeslagen in de opgegeven array.

Syntaxis

APPEND FROM ARRAY *<array-naam>*
[FIELDS *<veldenlijst>*]
[FOR *<voorwaarde>*]
[REINDEX]

<array-naam>

De array met de gegevens die als records in de huidige tabel moeten worden opgeslagen.

FIELDS *<veldenlijst>*

Voegt gegevens uit *<array-naam>* alleen toe aan de velden in *<veldenlijst>*. Zonder FIELDS *<veldenlijst>*, voegt APPEND FROM ARRAY aan elk record een element toe uit de opgegeven rij van *<array-naam>*, en wel in dezelfde volgorde als waarin de gegevens in de array zijn opgeslagen.

FOR *<voorwaarde>*

Beperkt APPEND FROM ARRAY tot array-rijen in *<array-naam>* die voldoen aan *<voorwaarde>*. In de *<voorwaarde>* kan naar een veld worden verwezen; dBASE voor Windows herkent welk array-element bij welk veld hoort en evalueert of een rij in de array moet worden toegevoegd alsof het array-element het benoemde veld was.

REINDEX
Stelt geopende indexen opnieuw samen nadat alle records zijn toegevoegd.

Beschrijving
APPEND FROM ARRAY behandelt een- en tweedimensionale array's als tabellen, waarbij de kolommen overeenkomen met velden en de rijen met records. Een eendimensionale array werkt als een tabel met slechts één rij; vandaar dat u slechts één record kunt toevoegen met een eendimensionale array. Een tweedimensionale array werkt als een tabel met meerdere rijen; vandaar dat u uit een tweedimensionale zoveel records kunt toevoegen als de array rijen heeft. Bij array's met meer dan twee dimensies bepaalt het laatste indexteken het aantal velden waaraan een array-rij wordt toegevoegd en het een-na-laatste veld geeft het aantal records aan dat moet worden toegevoegd. Array-indextekens voor de laatste twee worden genegeerd.

Als u gegevens uit een array toevoegt aan de huidige tabel, wordt elke rij uit de array toegevoegd als een enkel record. Als de tabel meer rijen heeft dat de array kolommen bevat, worden de overige velden leeg gelaten. Als de array meer kolommen heeft dan de tabel velden bevat, worden de overige kolommen genegeerd. De gegevens in de eerste kolom worden toegevoegd aan de inhoud van het eerste veld, de gegevens in de tweede kolom worden toegevoegd aan de inhoud van het tweede veld, enzovoort.

De gegevenstypen van de array moeten overeenkomen met de bijbehorende velden in de tabel waaraan u de gegevens toevoegt. Als het gegevenstype van een array-element en een bijbehorend veld niet overeenkomen, wordt een fout als resultaat gegeven.

Als de huidige tabel een memoveld bevat, wordt dat veld genegeerd. Als het tweede veld bijvoorbeeld een memoveld is, worden de gegevens uit de eerste kolom in de array toegevoegd aan de inhoud van het eerste veld en de gegevens uit de tweede kolom aan de inhoud van het derde veld, enzovoort.

Gebruik APPEND FROM ARRAY als alternatief voor automatische geheugenvariabelen in het geval u gegevens moet overbrengen tussen tabellen waarvan de structuren overeenkomen, maar de veldnamen verschillen.

Voorbeeld
In het volgende voorbeeld worden bepaalde velden uit de tabel Bedrijf naar een array gekopieerd en wordt APPEND FROM ARRAY gebruikt om een subverzameling records met Regio = NW naar Tijd.DBF te kopiëren:

```
CLOSE ALL
SET SAFETY OFF
USE Bedrijf EXCLUSIVE
INDEX ON CompCode TAG CompCode
DECLARE Kopie[RECCOUNT(),3]
COPY FIELDS Bedrijf, CompCode,Regio TO ARRAY Kopie
COPY STRUCTURE TO TEMP.DBF WITH PRODUCTION
USE TEMP EXCLUSIVE IN SELECT()
SELECT TEMP
APPEND FROM ARRAY Kopie ;
  FIELDS Bedrijf, CompCode, Regio ;
  FOR Regio = "NW" REINDEX
```

```
SET FIELDS TO Bedrijf, CompCode, Regio
GO TOP
BROWSE
RETURN
```

Zie ook

APPEND AUTOMEM, COPY TO ARRAY, DECLARE, REPLACE FROM ARRAY, STORE AUTOMEM

APPEND MEMO

Velden en records

Kopieert een tekstbestand naar een memoveld.

Syntaxis

APPEND MEMO *<memoveld>*
FROM *<bestandsnaam>* | ? | *<bestandsnaamfilter>*
[OVERWRITE]

<memoveld>

Het memoveld in de huidige tabel waar de inhoud van *<bestandsnaam>* aan moet worden toegevoegd.

FROM *<bestandsnaam>* | ? | *<bestandsnaamfilter>*

Geeft het bestand aan dat aan het memoveld in het huidige record moet worden toegevoegd. APPEND MEMO ? en APPEND MEMO *<bestandsnaamfilter>* toont een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand gezocht in de huidige directory en vervolgens in het pad dat is ingesteld met SET PATH. Als u een bestand zonder extensie opgeeft, wordt de extensie .TXT gebruikt.

OVERWRITE

Wist de inhoud van het memoveld in het huidige record alvorens de inhoud van *<bestandsnaam>* te kopiëren.

Beschrijving

Met APPEND MEMO kunt u een tekstbestand naar een memoveld in het huidige record kopiëren. U kunt één tekstbestand kopiëren naar elk memoveld van elk record in een tabel. Gebruik OVERWRITE om de inhoud van het memoveld in het huidige record te overschrijven.

Hoewel dBASE-memovelden andere soorten informatie dan tekst mogen bevatten, wordt voor afbeeldingen, geluid en andere door de gebruiker gedefinieerde binaire informatie het gebruik van binaire velden aangeraden. Zie het commando REPLACE BINARY voor meer informatie.

Voorbeeld

In het volgende voorbeeld wordt een nieuw, leeg record toegevoegd aan de tabel Klanten, wordt APPEND MEMO gebruikt om de inhoud van een tekstbestand in het memoveld Aantek te plaatsen en wordt het record geopend in een Tabel-editor om de gebruiker in staat te stellen andere velden te wijzigen of de inhoud van het memoveld aan te passen:

```
USE Klanten
APPEND BLANK
APPEND MEMO Aantek FROM C:\DBASEWIN\Memo1.TXT
BROWSE
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

COPY MEMO, REPLACE BINARY, REPLACE MEMO, REPLACE MEMO...FROM, REPLACE OLE

ARESIZE( ) — Geheugenvariabelen

Vergroot of verkleint een array op basis van de opgegeven afmetingen en geeft als resultaat een numerieke waarde die gelijk is het aantal elementen in de aangepaste array.

Syntaxis

ARESIZE(*<array-naam>*, *<nieuwe rijen Nuitdr>*
[,*<nieuwe kolommen Nuitdr>* [, *<waarden behouden Nuitdr>*]])

<array-naam>

De naam van de gedefinieerde een- of tweedimensionale array die u wilt vergroten of verkleinen.

<nieuwe rijen Nuitdr>

Het aantal rijen in de aangepaste array. *<nieuwe rijen Nuitdr>* moet altijd een positieve waarde (ongelijk aan nul) zijn.

<nieuwe kolommen Nuitdr>

Het aantal kolommen in de aangepaste array. *<nieuwe kolommen Nuitdr>* moet altijd een positieve waarde of nul zijn. Als u deze optie achterwege laat, verandert ARESIZE( ) het aantal rijen in de array, maar blijft het aantal kolommen ongewijzigd.

<waarden behouden Nuitdr>

Bepaalt wat er moet gebeuren met de waarden in de array als er rijen worden toegevoegd of verwijderd. Als u een waarde wilt opgeven voor *<waarden behouden Nuitdr>*, moet u ook een waarde opgeven voor *<nieuwe kolommen Nuitdr>*.

Beschrijving

Met ARESIZE( ) kunt u de afmetingen van een gedefinieerde array wijzigen, dat wil zeggen, u kunt de array groter of kleiner maken. Met ALEN( ) kunt u het aantal rijen en kolommen in een bestaande array bepalen.

Als u kolommen toevoegt aan of verwijdert uit een array, kunt u *<waarden behouden Nuitdr>* gebruiken om aan te geven hoe bestaande elementen in de nieuwe array moeten worden geplaatst. Als *<waarden behouden Nuitdr>* gelijk is aan nul of achterwege is gelaten, worden de elementen opnieuw gerangschikt: de nieuwe rijen of kolommen worden gevuld of aangepast op basis van verwijderde elementen en aan het eind van de array worden indien nodig elementen toegevoegd of verwijderd. In de volgende twee afbeeldingen ziet u wat er precies gebeurt. Dit commando zult u het meest gebruiken als u niet naar bestaande elementen in de array hoeft te verwijzen; dat wil zeggen, als u de array wilt bijwerken met nieuwe waarden.

Afbeelding 4.5 Regel en kolom toevoegen aan een 3x4-array, elementen herschikken

ARESIZE (eenARRAY,4,5)

1. Oorspronkelijke array gedeclareerd als:

```
DECLARE eenArray[3,4]
STORE "A" TO eenArray[1,1]
STORE "B" TO eenArray[1,2]
    :
STORE "L" TO eenArray[3,4]
```

1 A 1,1	2 B 1,2	3 C 1,3	4 D 1,4
5 E 2,1	6 F 2,2	7 G 2,3	8 H 2,4
9 I 3,1	10 J 3,2	11 K 3,3	12 L 3,4

Oorspronkelijke inhoud van eenArray.

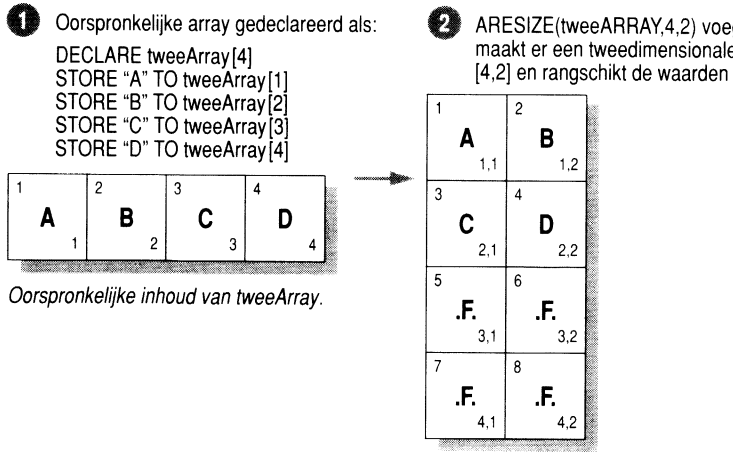
2. ARESIZE(eenArray,4,5) voegt nieuwe regel en kolom toe aan array en herschikt waarden van elementen.

1 A 1,1	2 B 1,2	3 C 1,3	4 D 1,4	5 E 1,5
6 F 2,1	7 G 2,2	8 H 2,3	9 I 2,4	10 J 2,5
11 K 3,1	12 L 3,2	13 .F. 3,3	14 .F. 3,4	15 .F. 3,5
16 .F. 4,1	17 .F. 4,2	18 .F. 4,3	19 .F. 4,4	20 .F. 4,5

Inhoud van de array na het commando ARESIZE(eenArray,4,5)

Afbeelding 4.6 Een kolom toevoegen aan een eendimensionale array, elementen herschikken

ARESIZE (tweeARRAY,4,2)



Oorspronkelijke inhoud van tweeArray.

Inhoud van de array na het commando ARESIZE(tweeArray,4,2)

Als u ARESIZE( ) gebruikt voor een eendimensionale array, kunt u de oorspronkelijke rij de eerste kolom in de nieuwe array maken. Gebruikt u ARESIZE( ) voor een tweedimensionale array, dan kunnen de bestaande array-elementen op hun oorspronkelijke posities blijven. dat is handig als u naar bestaande elementen in de array moet kunnen verwijzen; dat wil zeggen, als van plan bent nieuwe waarden aan de array toe te voegen zonder de bestaande waarden de verliezen.

Als *<waarden behouden Nuitdr>* een waarde ongelijk aan nul is, zorgt ARESIZE( ) dat bestaande elementen hun oorspronkelijke waarden behouden. In de volgende twee afbeeldingen worden de instructies uit de vorige twee afbeeldingen herhaald, maar nu met de waarde 1 voor *<waarden behouden Nuitdr>*.

Afbeelding 4.7 Regel en kolom toevoegen aan een 3x4-array, elementen behouden

ARESIZE (eenARRAY,4,5,1)

1. Oorspronkelijke array gedeclareerd als:

```
DECLARE eenArray[3,4]
STORE "A" TO eenArray[1,1]
STORE "B" TO eenArray[1,2]
   :
STORE "L" TO eenArray[3,4]
```

1	2	3	4
A 1,1	B 1,2	C 1,3	D 1,4
5	6	7	8
E 2,1	F 2,2	G 2,3	H 2,4
9	10	11	12
I 3,1	J 3,2	K 3,3	L 3,4

Oorspronkelijke inhoud van eenArray.

2. ARESIZE(eenARRAY,4,5,1) voegt een nieuwe rij en kolom toe en behoudt de waarden van de elementen.

1	2	3	4	5
A 1,1	B 1,2	C 1,3	D 1,4	.F. 1,5
6	7	8	9	10
E 2,1	F 2,2	G 2,3	H 2,4	.F. 2,5
11	12	13	14	15
I 3,1	J 3,2	K 3,3	L 3,4	.F. 3,5
16	17	18	19	20
.F. 4,1	.F. 4,2	.F. 4,3	.F. 4,4	.F. 4,5

Inhoud van de array na het commando ARESIZE(eenArray,4,5,1)

Afbeelding 4.8 Een kolom toevoegen aan een eendimensionale array, elementen behouden

ARESIZE (tweeARRAY,4,2,1

1. Oorspronkelijke array gedeclareerd als:

```
DECLARE tweeArray[4]
STORE "A" TO tweeArray[1]
STORE "B" TO tweeArray[2]
STORE "C" TO tweeArray[3]
STORE "D" TO tweeArray[4]
```

1	2	3	4
A 1	B 2	C 3	D 4

Oorspronkelijke inhoud van tweeArray.

2. ARESIZE(tweeARRAY,4,2,1) voegt een nieuwe kolom toe, maakt er een tweedimensionale array van met dimensie [4,2]. Elk bestaand element is nu het eerste element in een rij.

1	2
A 1,1	B 1,2
3	4
C 2,1	D 2,2
5	6
.F. 3,1	.F. 3,2
7	8
.F. 4,1	.F. 4,2

Inhoud van de array na het commando ARESIZE(tweeArray,4,2,1)

Voorbeeld

In het volgende voorbeeld wordt in eerste instantie een array gedefinieerd met drie elementen. Vervolgens wordt ARESIZE( ) gebruikt om de grootte achtereenvolgens te wijzigen in A[5] en A[5,2] en vervolgens weer terug naar A[3]. DISPLAY MEMORY wordt gebruikt om de waarden in de array te tonen na elke instructie met ARESIZE( ):

```
RELEASE ALL
DECLARE A[3]
A[1]="x"
A[2]="y"
A[3]="z"
DISPLAY MEMORY
N=ARESIZE(A,5)          && A heeft nu 5 elementen
A[4]="nieuw1"
A[5]="nieuw2"
DISPLAY MEMORY
N=ARESIZE(A,5,2,1)
* A heeft nu 5 rijen en 2 kolommen.
* Nieuwe kolommen hebben allemaal de waarde .t.
* Oude waarden blijven behouden.
DISPLAY MEMORY
N=ARESIZE(A,3,1,1)
* A bestaat nu weer uit de oorspronkelijke 3 elementen
* Gebruik:
DISPLAY MEMORY
N=ARESIZE(A,3,1,0)
* als u de oorspronkelijke waarden niet meer nodig hebt
DISPLAY MEMORY
WAIT
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ADEL( ), AGROW( ), AINS( ), ALEN( ), DECLARE

ASC( )

Uitdrukkingen en gegevenstypeconversie

Geeft de numerieke ASCII-waarde van het opgegeven teken als resultaat.

Syntaxis

ASC(<*Tuitdr*>)

<*Tuitdr*>

Het teken waarvan u de ASCII-waarde wilt weten. U kunt meer dan één teken opgeven, maar alleen het eerste teken wordt gebruikt.

Beschrijving

ASC( ) is de tegenovergestelde functie van CHR( ). ASC( ) krijgt een teken als argument en geeft de ASCII-waarde als resultaat, een getal van 0 tot en met 255. CHR( ) krijgt een ASCII-waarde als argument en geeft een teken als resultaat.

U kunt ASC( ) gebruiken om ASCII-waarden van tekens te manipuleren met rekenkundige operatoren. Een lijst van ASCII-waarden en de bijbehorende tekens vindt u in de ASCII-tabel in Appendix E.

Voorbeeld

In het volgende voorbeeld wordt ASC( ) gebruikt om de ASCII-waarde te bepalen van een opgegeven teken:

```
SET TALK OFF
plusminus = CHR(241)
? "De ASCII-waarde van  " + plusminus + " is "
?? + LTRIM(STR(ASC(plusminus),3,0))
```

Zie ook

ANSI( ), CHR( ), OEM( )

ASCAN( )

Geheugenvariabelen

Zoekt een uitdrukking in een array. Geeft als resultaat het eerste teken dat overeenkomt met de uitdrukking als de uitdrukking is gevonden of 0 als die niet is gevonden.

Syntaxis

ASCAN(*<array-naam>*, *<uitdr>*
[, *<eerste element Nuitdr>* [, *<elementen Nuitdr>*]])

<array-naam>

Een gedefinieerde een- of tweedimensionale array.

<uitdr>

De uitdrukking die moet worden gezocht in *<array-naam>*.

<eerste element Nuitdr>

Het elementnummer in *<array-naam>* waar met zoeken moet worden begonnen. Zonder *<eerste element Nuitdr>* begint ASCAN( ) met zoeken bij het eerste element.

<elementen Nuitdr>

Het aantal elementen in *<array-naam>* dat ASCAN( ) moet doorzoeken. Zonder *<elementen Nuitdr>*, zoekt ASCAN( ) in *<array-naam>* vanaf *<eerste element Nuitdr>* tot het eind van de array. Als u een waarde wilt opgeven voor *<elementen Nuitdr>*, moet u ook een waarde opgeven voor *<eerste element Nuitdr>*.

Beschrijving

Met ASCAN( ) kunt u in een array zoeken naar een uitdrukking die u opgeeft in <*uitdr*>. Als een array bijvoorbeeld de namen van klanten bevat, kunt u ASCAN( ) gebruiken om de positie van een bepaalde naam te bepalen.

ASCAN( ) geeft als resultaat het elementnummer van het eerste element dat overeenkomt met <*uitdr*>. Met ASUBSCRIPT( ) kunt u vervolgens de indextekens van het element bepalen.

Als <*uitdr*> tekenreeksgegevens bevat, maakt ASCAN( ) onderscheid tussen hoofdletters en kleine letters; U kunt UPPER( ), LOWER( ) of PROPER( ) gebruiken om de hoofdletters of kleine letters van <*uitdr*> in overeenstemming te brengen met die van de gegevens in de array.

Als <*uitdr*> tekenreeksgegevens bevat, zoekt ASCAN( ) naar een uitdrukking op basis van de regels die zijn ingesteld met SET EXACT. Als SET EXACT is ingeschakeld (ON), wordt 0 als resultaat gegeven als de waarde in <*uitdr*> niet gelijk is aan de gegevens in een element in de array. Als SET EXACT is uitgeschakeld (OFF), wordt 0 als resultaat gegeven als de tekens in <*Nuitdr*> niet overeenkomen met de eerste tekens van gegevens in een element in de array. Het volgende voorbeeld maakt dit nog wat duidelijker. Zie EXACT voor meer informatie.

```
DECLARE eenArray[3,4]          && 3 rijen, 4 kolommen
? AFILL(eenArray,"abcd",6,1)   && "abcd" plaatsen in;
                                  het zesde element
SET EXACT OFF
? ASCAN(eenArray,"abcd")       && Heeft 6 als resultaat
? ASCAN(eenArray,"abc")        && Heeft 6 als resultaat
? ASCAN(eenArray,"bcd")        && Heeft 0 als resultaat
SET EXACT ON
? ASCAN(eenArray,"abcd")       && Heeft 6 als resultaat
? ASCAN(eenArray,"abc")        && Heeft 0 als resultaat
? ASCAN(eenArray,"abc")        && Heeft 0 als resultaat
```

Voorbeeld

In het volgende voorbeeld wordt ASCAN( ) gebruikt om het elementnummer voor de gewenste tekenreeks in de array te bepalen. Met ASUBSCRIPT( ) worden vervolgens de rij- en kolomcoördinaten binnen de array bepaald:

```
CLEAR
DECLARE A_Dir[1]
BestNaam="KLANTEN.DBF"
Bestanden=ADIR(A_Dir,"*.*")  && Array initialiseren met;
                                inhoud van directory
Asort=ASORT(A_Dir)           && Array rangschikken
Element=ASCAN(A_Dir,BestNaam)&& Heeft de plaats van;
                                bestandsnaam als resultaat
IF Element > 0               && ASCAN() heeft bij succes;
                                1 als resultaat
  Rij=ASUBSCRIPT(A_Dir,Element,1)
  Kol=ASUBSCRIPT(A_Dir,Element,2)
  ? "Bestandsnaam   : ", A_Dir[Rij,Kol]
  ? "Bestandsgrootte: ", A_Dir[Rij,Kol+1]
  ? "Bestandsdatum  : ", A_Dir[Rij,Kol+2]
```

```
    ? "Bestandstijd   : ", A_Dir[Rij,Kol+3]
  ENDIF
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ACOPY( ), AFIELDS( ), AFILL( ), ASORT( ), ASUBSCRIPT( ), DECLARE, LOWER( ), PROPER( ), SET EXACT, UPPER( )

ASIN( )

Numerieke gegevens

Geeft de inverse sinus (arcsinus) van een getal als resultaat.

Syntaxis

ASIN(<*Nuitdr*>)

<Nuitdr>

De sinus van een hoek, van -1 tot +1.

Beschrijving

ASIN( ) geeft als resultaat de waarde in radialen van de hoek waarvan de sinus gelijk is aan <*Nuitdr*>. ASIN( ) geeft een zwevende waarde van -pi/2 tot pi/2 radialen als resultaat. ASIN( ) heeft nul als resultaat als <*Nuitdr*> gelijk is aan 0. Voor waarden van *x* van -pi/2 tot pi/2, geeft ASIN(Y) *x* als resultaat als SIN(X) de waarde *y* als resultaat geeft.

Met RTOD( ) kunt u de resultaatwaarde in radialen omzetten naar graden. Als het standaardaantal decimalen bijvoorbeeld 2 is, geeft ASIN(.5) 0,52 radialen als resultaat, terwijl RTOD(ASIN(.5)) 30,00 graden oplevert.

Het aantal weergegeven decimalen stelt u in met SET DECIMALS.

De arccosecans van een waarde bepaalt u door de 1 te delen door de arcsinus van de waarde. De arccosecans van 1,54 is bijvoorbeeld ASIN(1/1.54), ofwel 0,71 radialen.

Voorbeeld

In het volgende voorbeeld wordt ASIN( ) gebruikt om de arcsinus te bepalen van een verzameling sinuswaarden:

```
CLEAR
? "Arcsinus" AT 20
?
? "Sinus" AT 9, "Radialen" AT 20,"Graden" AT 33
SET DECIMALS TO 2
FOR sinus = -1 TO 1 STEP .25
```

```
   ? sinus AT 0, ASIN(sinus) AT 13, RTOD(ASIN(sinus)) ;
     AT 26
NEXT
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ACOS( ), ATAN( ), ATN2( ), DTOR( ), RTOD( ), SET DECIMALS, SIN( )

ASORT( ) — Geheugenvariabelen

Sorteert de elementen in een eendimensionale array of de rijen in een tweedimensionale array. Geeft 1 als resultaat als de bewerking is geslaagd en anders een fout.

Syntaxis

ASORT(*<array-naam>*
[, *<eerste element Nuitdr>* [,*<te sorteren elementen Nuitdr>*
[, *<sorteervolgorde Nuitdr>*]]])

<array-naam>

Een gedefinieerde een- of tweedimensionale array.

<eerste element Nuitdr>

In een eendimensionale array is dit het nummer van het eerste element in *<array-naam>* waar met sorteren moet worden begonnen. In een tweedimensionale array is dit het nummer (indexteken) van de kolom waarop moet worden gesorteerd. Zonder *<eerste element Nuitdr>* begint ASORT( ) bij het eerste element of de eerste kolom in de array.

<te sorteren elementen Nuitdr>

In een eendimensionale array is dit het aantal te sorteren elementen. In een tweedimensionale array is dit het aantal te sorteren rijen. Zonder *<te sorteren elementen Nuitdr>* sorteer ASORT( ) de rijen vanaf de rij met het element *<eerste element Nuitdr>* tot de laatste rij. Als u een waarde wilt opgeven voor *<te sorteren elementen Nuitdr>*, moet u ook een waarde opgeven voor *<eerste element Nuitdr>*.

<sorteervolgorde Nuitdr>

De sorteervolgorde:

- 0 sorteert oplopend (de standaardinstelling)
- 1 sorteert aflopend

Als u een waarde wilt opgeven voor *<sorteervolgorde Nuitdr>*, moet u ook waarden opgeven voor *<te sorteren elementen Nuitdr>* en *<eerste element Nuitdr>*.

Beschrijving

ASORT( ) kan alle opgegeven elementen sorteren als ze van hetzelfde gegevenstype zijn. De te sorteren elementen in een eendimensionale array moeten van hetzelfde gegevenstype zijn en dat geldt ook voor de elementen in de kolom waarop rijen moeten worden gesorteerd in een tweedimensionale array.

ASORT( ) rangschikt de elementen alfabetisch, numeriek, chronologisch of logisch op basis van het gegevenstype van *<eerste element Nuitdr>*. In het geval van tekengegevens bepaalt de huidige taalaansturing de sorteervolgorde. Zie Appendix C in *Programmeren* voor meer informatie over de regels voor de sorteervolgorde in de Nederlandse taalaansturing.

Eendimensionale array's

Stel dat u het commando DECLARE aGetal[8] geeft en getallen opslaat in de array zodat de array-elementen deze volgorde hebben:

5 7 3 9 4 1 2 8

Het commando ASORT(aGetal, 1, 5) zorgt dat dBASE de eerste vijf elementen sorteert zodat de array-elementen dan deze volgorde hebben:

3 4 5 7 9 1 2 8

Als u vervolgens het commando ASORT(aGetal, 5, 2) geeft, worden twee elementen vanaf het vijfde element gesorteerd zodat de array dan deze volgorde heeft:

3 4 5 7 1 **9 2** 8

Tweedimensionale array's

Het gebruik van ASORT( ) met een tweedimensionale array komt overeen met het gebruik van het commando SORT met een tabel. De rijen in de array komen overeen met de records in de tabel en de kolommen met de velden.

Als u een tweedimensionale array sorteert, worden volledige rijen gesorteerd en niet alleen de elementen in de kolom waar *<eerste element Nuitdr>*) staat.

Stel dat u het commando DECLARE aInfo[4, 3] geeft en de array vult met de volgende gegevens:

{15-09-65}	7	A
{31-12-65}	4	D
{19-01-45}	8	C
{02-05-72}	2	B

Als u het commando ASORT(aInfo, 1) geeft, worden alle rijen in de array gesorteerd vanaf elementnummer 1. De rijen worden gesorteerd op de datums in de eerste kolom omdat element 1 een datum is.

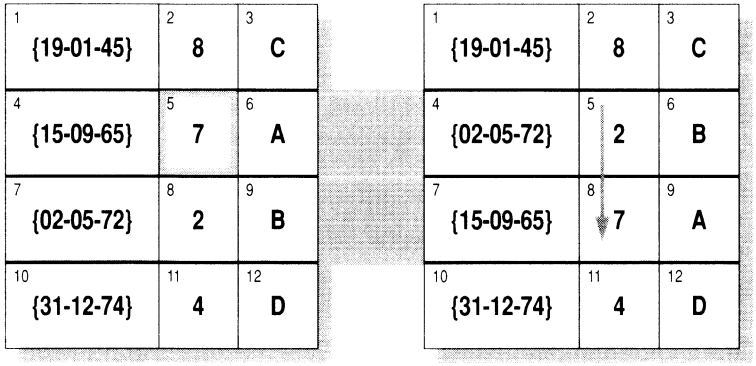
Als u het commando ASORT(aInfo, 5, 2) geeft, worden twee rijen in de array gesorteerd, te beginnen bij elementnummer 5 (met de waarde 7). ASORT( ) sorteert de tweede en derde rijen op basis van de getallen in de tweede kolom. De resultaten ziet u in de volgende afbeelding.

Afbeelding 4.9 Gebruik van ASORT() met een tweedimensionale array

ASORT(aInfo, 5, 2)

1. Twee *rijen* sorteren (ASORT(aInfo, 5, **2**)) te beginnen met de rij met *element 5* (ASORT(aInfo, **5**, 2)).

2. Element 5 bevat een getal, dus worden de rijen gesorteerd op de nummers in de tweede kolom.



Oorspronkelijke inhoud van aInfo

Inhoud na het commando ASORT(aInfo, 5, 2)

Voorbeeld

Zie ASCAN( ) voor een voorbeeld van ASORT( ).

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

AELEMENT( ), ALEN( ), ASCAN( ), ASUBSCRIPT( ), DECLARE

ASUBSCRIPT( )

Geheugenvariabelen

Geeft als resultaat het rijnummer of het kolomnummer van een opgegeven element in een array.

Syntaxis

ASUBSCRIPT(<*array-naam*>, <*element Nuitdr*>, <*rij/kolom Nuitdr*>)

<array-naam>

Een gedefinieerde een- of tweedimensionale array.

<element Nuitdr>
Het elementnummer.

<rij/kolom Nuitdr>
Een getal, 1 of 2, dat aangeeft of u het rij- of het kolomindexteken van een array wilt weten. Als *<rij/kolom Nuitdr>* 1 is, geeft ASUBSCRIPT( ) het nummer van het rij-indexteken. Als *<rij/kolom Nuitdr>* 2 is, levert ASUBSCRIPT( ) het nummer van het kolomindexteken.

Als *<array-naam>* een eendimensionale array is, wordt een fout als resultaat gegeven als *<rij/kolom Nuitdr>* een andere waarde is dan 1.

Beschrijving

Met ASUBSCRIPT( ) kunt u het nummer van een element in een a tweedimensionale array bepalen zodat u naar het element kunt verwijzen door middel van indextekens.

Als u zowel het rij- als het kolomnummer van een element in een tweedimensionale array wilt weten, moet u het commando ASUBSCRIPT( ) twee keer gebruiken, een keer met de waarde 1 voor *<rij/kolom Nuitdr>* en een keer met de waarde 2 voor *<rij/kolom Nuitdr>*. Als het elementnummer 13 is, kunt u bijvoorbeeld de volgende instructies gebruiken om de indextekens te bepalen:

```
? ASUBSCRIPT(eenArray,13,1) && rij-indexteken als resultaat
? ASUBSCRIPT(eenArray,13,2) && kolomindexteken als resultaat
```

In eendimensionale array's is het elementnummer gelijk aan het indexteken, dus heeft het gebruik van ASUBSCRIPT( ) geen zin. ASUBSCRIPT(een1Array,3,1) geeft 3 als resultaat, ASUBSCRIPT(een1Array,5,1) geeft 5 als resultaat, enzovoort.

ASUBSCRIPT( ) is het tegengestelde van de functie AELEMENT( ), die het elementnummer als resultaat geeft als u de indextekens opgeeft.

Voorbeeld

In het volgende voorbeeld wordt ASCAN( ) gebruikt om het elementnummer van een tekenreeks in een array te bepalen. Vervolgens geeft ASUBSCRIPT( ) de rij- en kolomcoördinaten binnen de array als resultaat:

```
CLEAR
DECLARE A_Dir[1]
BestNaam="KLANTEN.DBF"
Bestanden=ADIR(A_Dir,"*.*")  && Array initialiseren met;
                                inhoud van directory
Asort=ASORT(A_Dir)           && Array rangschikken
Element=ASCAN(A_Dir,BestNaam)&& Heeft de plaats van;
                                bestandsnaam als resultaat
IF Element > 0               && ASCAN() heeft bij succes;
                                1 als resultaat
  Rij=ASUBSCRIPT(A_Dir,Element,1)
  Kol=ASUBSCRIPT(A_Dir,Element,2)
  ? "Bestandsnaam   : ", A_Dir[Rij,Kol]
  ? "Bestandsgrootte: ", A_Dir[Rij,Kol+1]
  ? "Bestandsdatum  : ", A_Dir[Rij,Kol+2]
```

```
    ? "Bestandstijd   : ", A_Dir[Rij,Kol+3]
  ENDIF
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ACOPY( ), ADEL( ), AELEMENT( ), AFIELDS( ), AINS( ), ALEN( ), ASCAN( ), ASORT( ), DECLARE

AT( )

Tekenreeksgegevens

Geeft een getal als resultaat dat de positie van een tekenreeks binnen een andere tekenreeks aangeeft.

Syntaxis

AT(<*zoekreeks Tuitdr*>, <*doel Tuitdr*> | <*doelmemoveld*>
[, <*nde overeenkomst Nuitdr*>])

<*zoekreeks Tuitdr*>

De tekenreeks die moet worden gezocht in <*doel Tuitdr*> of <*doelmemoveld*>.

<*doel Tuitdr*> | <*doelmemoveld*>

De tekenreeks of het memoveld waarin <*zoekreeks Tuitdr*> moet worden gezocht.

<*nde overeenkomst Nuitdr*>

Als de zoekreeks meer dan eens voorkomt, kunt u hier opgeven welke u wilt vinden. Standaard wordt gezocht naar de eerste keer dat de tekenreeks voorkomt. U kunt hier een getal (groter dan nul) opgeven als u niet het eerste, maar een volgend exemplaar van de tekenreeks wilt zoeken.

Beschrijving

AT( ) levert de numerieke positie waar een *zoekreeks* begint in een *doelreeks* of in een doelmemoveld. AT( ) zoekt van links naar rechts met één teken tegelijk vanaf het eerste teken van de tekenreeks of het memoveld tot het eind.

Bij het zoeken wordt onderscheid gemaakt tussen hoofdletters en kleine letters. Gebruik UPPER( ) of LOWER( ) om dat onderscheid op te heffen.

Als de zoekreeks meer dan eens voorkomt, kunt u opgeven welke overeenkomst u wilt vinden door een getal op te geven voor <*nde overeenkomst Nuitdr*>. Als u dit argument achterwege laat, levert AT( ) de eerste positie van de eerste overeenkomst.

Onder de volgende omstandigheden geeft AT( ) 0 als resultaat:

- De zoekreeks is niet aangetroffen.
- De zoekreeks, de doelreeks of het doelmemoveld is leeg.
- De zoekreeks is langer dan de doelreeks.
- De *<nde overeenkomst Nuitdr>* bestaat niet.

Als *<nde overeenkomst Nuitdr>* kleiner is dan 1, wordt een fout als resultaat gegeven.

Als AT( ) tekens in een memoveld telt, wordt voor elke combinatie van een regelterugloop en een regeldoorvoer (CR/LF) twee tekens geteld.

Met RAT( ) kunt u de eerste positie van een *<zoekreeks Tuitdr>* zoeken van *rechts naar links*, van het eind tot het begin. Met de subreeksoperator ($) kunt u bepalen of een string voorkomt binnen een andere. Zie Hoofdstuk 1 voor meer informatie over operatoren.

Voorbeeld

In het volgende voorbeeld wordt AT( ) gebruikt om het begin van een opgegeven tekenreeks te vinden binnen een twee tekenreeks:

```
? AT("B","ABC")                    && Geeft 2
? AT("ss","Mississippi")           && Geeft 3
? AT("ss","Mississippi",2)         && Geeft 6
? AT("Z","ABC")                    && Geeft 0
? AT("a","ABC")                    && Geeft 0
? AT("ABC","AB")                   && Geeft 0
? AT("","ABC")                     && Geeft 0
? AT("a","abc",2)                  && Geeft 0
```

In het volgende voorbeeld wordt AT( ) gebruikt om een lijst te tonen of af te drukken van alle records in de tabel Bedrijf die in het memoveld Notities een bepaalde tekenreeks bevatten. Tevens wordt de eerste positie van die reeks binnen het memoveld gegeven.

```
USE Bedrijf
Reeks="Voorkeursbehandeling"
LIST Bedrijf, Notities FOR AT(Reeks, Notities)>0
CLOSE ALL
```

Overdraagbaarheid

In dBASE IV is het zoeken in een memoveld beperkt tot de eerste 64K met gegevens. De argumenten *<memoveld>* en *<nde overeenkomst Nuitdr>* worden niet ondersteund in dBASE III PLUS.

Zie ook

RAT( ), STUFF( ), SUBSTR( )

ATAN( )

Geeft de inverse tangens (arctangens) van een getal als resultaat.

Syntaxis

ATAN(<*Nuitdr*>)

<Nuitdr>

Een positief of negatief getal dat de tangens van een hoek voorstelt.

Beschrijving

ATAN( ) geeft als resultaat de waarde in radialen van de hoek waarvan de tangens gelijk is aan <*Nuitdr*>. het resultaat bestaat uit een zwevende waarde tussen -pi/2 en pi/2 radialen. ATAN( ) levert 0 als <*Nuitdr*> gelijk is aan 0. Voor waarden van *x* van -pi/2 tot pi/2 levert ATAN(Y) de waarde *x* als TAN(X) de waarde *y* als resultaat geeft.

Met RTOD( ) kunt u radialen omzetten in graden. Als het aantal decimalen standaard bijvoorbeeld 2 is, levert ATAN(1) 0,79 radialen als resultaat, terwijl RTOD(ATAN(1)) als resultaat 45,00 graden geeft.

Het aantal decimalen stelt u in met SET DECIMALS.

Het verschil tussen ATAN( ) en ATN2( ) is dat ATAN( ) een tangens als argument krijgt, terwijl ATN2( ) de sinus en de cosinus als argumenten krijgt.

De arccotangens van een waarde bepaalt u door pi/2 af te trekken van de arctangens van de waarde. De arccotangens van pi/3 is bijvoorbeeld pi/2-ATAN(pi/3), ofwel 0,76.

Voorbeeld

In het volgende voorbeeld wordt ATAN( ) gebruikt om de arctangens van een reeks tangenswaarden te bepalen:

```
CLEAR
SET DECIMALS TO 5
? "Arctangens" AT 18
? "Tangens" AT 6, "Radialen" AT 20, "Graden" AT 33
FOR tang = -100 TO 1000 STEP 100
  ? tang AT 0, ATAN(tang) AT 12, ;
  LTRIM(STR(RTOD(ATAN(tang)),8,5)) AT 33
NEXT
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ACOS( ), ASIN( ), ATN2( ), RTOD( ), SET DECIMALS, TAN( )

ATN2( )

Geeft de inverse tangens (arctangens) van een gegeven punt als resultaat.

Syntaxis

ATN2(*<sinus Nuitdr>*, *<cosinus Nuitdr>*)

<sinus Nuitdr>

De sinus van een hoek. Als *<sinus Nuitdr>* gelijk is aan 0, kan *<cosinus Nuitdr>* niet ook gelijk zijn aan 0.

<cosinus Nuitdr>

De cosinus van een hoek. Als *<cosinus Nuitdr>* gelijk is aan 0, kan *<sinus Nuitdr>* niet ook gelijk zijn aan 0. Als *<cosinus Nuitdr>* gelijk is aan 0 en *<sinus Nuitdr>* is een positieve of negatieve waarde (ongelijk aan nul), geeft ATN2( ) respectievelijk +pi/2 of -pi/2 als resultaat.

Beschrijving

ATN2( ) geeft als resultaat de hoek in radialen als u de sinus en de cosinus van die hoek opgeeft. het resultaat is een zwevend getal van -pi tot +pi radialen. ATN2( ) levert 0 als *<sinus Nuitdr>* gelijk is aan 0. Als u voor beide argumenten 0 opgeeft, wordt een fout als resultaat gegeven.

Met RTOD( ) kunt u radialen omzetten in graden. Als het aantal decimalen standaard bijvoorbeeld 2 is, geeft ATN2(1.0) 1,57 radialen en RTOD(ATN2(1.0)) 90,00 graden als resultaat.

Het standaard aantal decimalen stelt u in met SET DECIMALS.

Het verschil tussen ATN2(en ATAN( ) is dat ATN2( ) de sinus en de cosinus als argumenten ontvangt, terwijl het argument bij ATAN( ) de tangens is. Zie ATAN( ) voor meer informatie over het bepalen van de arccotangens.

Voorbeeld

In het volgende voorbeeld ziet u enkele toepassingen van ATN2( ):

```
? ATN2(2.79,-3.2)          && Geeft 2,42 als resultaat
? ATN2(-3,3)               && Geeft -0,79 als resultaat
? RTOD(ATN2(-3,3))         && Geeft -45 als resultaat
? RTOD(ATN2(-1,-SQRT(3)))  && Geeft -150 als resultaat
x=SIN(DTOR(30))
y=COS(DTOR(30))
? RTOD(ATN2(x,y))          && Geeft 30 als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ACOS( ), ASIN( ), ATAN( ), COS( ), RTOD( ), SET DECIMALS, SIN( ), TAN( )

AVERAGE — Tabelindeling

Berekent het rekenkundig gemiddelde van opgegeven numerieke of zwevende velden in de huidige tabel.

Syntaxis

AVERAGE
[<*uitdrukkingenlijst*>]
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[TO <*variabelenlijst*> | TO ARRAY <*array-naam*>]

<*uitdrukkingenlijst*>

De numerieke of zwevende velden, of uitdrukkingen met numerieke of zwevende velden, waarvan u het gemiddelde wilt berekenen.

<*bereik*>

Het bereik van de records waarvan u het gemiddelde wilt berekenen. RECORD <*n*> geeft een enkel record aan door middel van zijn recordnummer. NEXT <*n*> geeft *n* records aan, te beginnen bij het huidige record. ALL specificeert alle records. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR <*voorwaarde1*>
WHILE <*voorwaarde2*>

Bepaalt welke records wordt beïnvloed door AVERAGE. FOR beperkt AVERAGE tot records die voldoen aan <*voorwaarde1*>, te beginnen bij het eerste record in de tabel of weergave. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang <*voorwaarde2*> waar is.

TO <*variabelenlijst*> | TO ARRAY <*array-naam*>

Initialiseert de geheugenvariabelen in <*variabelenlijst*> en slaat de gemiddelden daar op of slaat de gemiddelden op in de bestaande array <*array-naam*>. Als u een array opgeeft, wordt elk veldgemiddelde in de elementen opgeslagen op volgorde van de velden in <*uitdrukkingenlijst*>. Als u geen <*uitdrukkingenlijst*> opgeeft, wordt elk veldgemiddelde opgeslagen op volgorde van de veldnummers. <*array-naam*> kan een een- of meerdimensionale array zijn. U kunt gemiddelden opslaan in een array zonder het sleutelwoord ARRAY te gebruiken door nadrukkelijk te verwijzen naar de indextekens.

Beschrijving

Het commando AVERAGE berekent het rekenkundig gemiddelde van numerieke uitdrukkingen en slaat de resultaten op in opgegeven geheugenvariabelen of array-elementen. Als de waarden worden opgeslagen in geheugenvariabelen, moet het aantal geheugenvariabelen exact gelijk zijn aan het aantal berekende gemiddelden. Als u de waarden opslaat in een array, moet die array al bestaan en het aantal elementen moet tenminste zo groot zijn als het aantal berekende gemiddelden.

Als SET TALK is ingeschakeld (ON), geeft AVERAGE tevens de resultaten weer in het resultatenpaneel van het commandovenster. De instelling SET DECIMALS bepaalt het aantal decimalen dat door AVERAGE wordt weergegeven. Numerieke velden in lege records worden geëvalueerd als nul. Als u lege records wilt uitsluiten, kunt u de functie ISBLANK( ) toepassen in een FOR-voorwaarde. EMPTY( ) sluit records uit waarin de opgegeven uitdrukking 0 of leeg is.

Voorbeeld

In het volgende voorbeeld wordt AVERAGE gebruikt om de gemiddelde jaaromzet tot de dag van vandaag te berekenen voor alle bedrijven in de tabel Bedrijf:

```
USE Bedrijf
AVERAGE VerkTotNu TO JozGem
? "De gemiddelde jaaromzet tot heden is f", ;
   JozGem PICTURE "99,999,999.99"
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CALCULATE, COUNT, SUM, TOTAL

BAR( )

dBASE IV-menu's

Geeft het nummer van de huidige geselecteerde strip of de laatst geselecteerde strip in een dBASE IV popup-menu als resultaat. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows kunt u INSPECT( ) gebruiken om informatie als resultaat te geven over objecten in formulieren.

Zie Help voor meer informatie over de syntaxis van BAR( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

BARCOUNT( )

dBASE IV-menu's

Geeft het aantal strips in een dBASE IV popup-menu als resultaat. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor

Windows kunt u INSPECT( ) gebruiken om informatie als resultaat te geven over objecten in formulieren.

Zie Help voor meer informatie over de syntaxis van BARCOUNT( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

BARPROMPT( ) dBASE IV-menu's

Geeft de informatieregel voor een strip in een dBASE IV popup-menu als resultaat. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows kunt u INSPECT( ) gebruiken om informatie als resultaat te geven over objecten in formulieren.

Zie Help voor meer informatie over de syntaxis van BARPROMPT( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

BEGINTRANS( ) Gedeelde gegevens

Start een transactie en geeft .T. als resultaat als de transactie correct is begonnen.

Syntaxis

BEGINTRANS([*<database-naam Tuitdr>*])

<database-naam Tuitdr>

De naam van de SQL-database waarin de transactie moet beginnen.

- Als *<database-naam Tuitdr>* achterwege is gelaten, maar wel een SET DATABASE-instructie is gegeven, gebruikt BEGINTRANS( ) de database die in de SET DATABASE-instructie is genoemd.
- Als *<database-naam Tuitdr>* achterwege is gelaten en er is geen SET DATABASE-instructie gegeven, gebruikt BEGINTRANS( ) de database die is geopend nadat BEGINTRANS( ) is uitgevoerd.

Opmerking Als u bij BEGINTRANS( ) *<database-naam Tuitdr>* opneemt, moet u dat ook opnemen in volgende instructies met COMMIT( ) of ROLLBACK( ) binnen de transactie. Als u dat niet doet, wordt de instructie met COMMIT( ) of ROLLBACK( ) genegeerd.

Beschrijving

Gebruik BEGINTRANS( ) om een *transactie* te starten tijdens welke de gebruiker wijzigingen kan aanbrengen in een of meer tabellen in een SQL-database die transactieverwerking ondersteunt. Binnen een transactie die is gestart met BEGINTRANS( ) kunt u slechts met één database werken. Verder is het ook niet mogelijk transactie te nesten.

Als u het commando BEGINTRANS( ) geeft voor een SQL-database die transacties niet ondersteunt, of als er een server-fout optreedt, geeft BEGINTRANS( ) .F. als resultaat. Anders wordt .T. als resultaat gegeven. Als BEGINTRANS( ) .F. als resultaat geeft, kunt u SQLERROR( ) of SQLMESSAGE( ) gebruiken om eventueel de aard van de server-fout te bepalen.

U sluit een transactie af met COMMIT( ) of ROLLBACK( ). COMMIT( ) slaat de wijzigingen op die tijdens de transactie zijn gemaakt. ROLLBACK( ) negeert de wijzigingen en herstelt de tabellen in de toestand van voor BEGINTRANS( ).

BEGINTRANS( ) is van toepassing op de volgende commando's:

@...SAY...GET	DELETE	REPLACE
APPEND	EDIT	REPLACE MEMO/BINARY/OLE
APPEND BLANK	FLOCK( )	RLOCK( )
APPEND MEMO	INSERT	
BROWSE	RECALL	

De volgende commando's zijn in transacties niet toegestaan. Er wordt een fout als resultaat gegeven als u deze commando's binnen een transactie tracht te geven.

BEGINTRANS( ) (geneste transacties zijn niet toegestaan)	DELETE TAG
CLEAR ALL	INDEX
CLOSE ALL/DATABASE/INDEX (elk commando waarmee geopende tabellen of indexen worden gesloten)	MODIFY STRUCTURE
CONVERT	PACK
CREATE FROM	USE, als daarmee een geopende tabel wordt gesloten of een tabel in een andere database wordt geopend
ZAP	

Voorbeeld

In het volgende voorbeeld wordt een transactie gestart met BEGINTRANS( ). Er wordt een multi-user-versie geopend van Bedrijf.dbf en vervolgens wordt geprobeerd alle VERKTOTNU 0 te maken. ON ERROR neemt eventuele fouten waar, zoals wanneer een andere gebruiker een record in Bedrijf.dbf heeft vergrendeld. Als een fout optreedt, worden alle waarden hersteld door ROLLBACK( ). Zo niet, dan schrijft COMMIT( ) de wijzigingen naar schijf:

```
CLOSE ALL
SET PROCEDURE TO PROGRAM(1) ADDITIVE
SET EXCLUSIVE OFF

BEGINTRANS()
```

```
TransFt=.f.
ON ERROR DO TransFt      && ON ERROR activeren

USE C:\VEELGEBR\BEDRIJF
REPLACE ALL VerkTotNu WITH 0
ON ERROR                 && ON ERROR uitschakelen

IF TransFt
  ? "Rollback"
  ROLLBACK()             && Gegevens herstellen
ELSE
  ? "Commit"
  COMMIT()               && Gegevens opslaan
ENDIF

PROC TransFt
WAIT "Waarschuwing: Transactie niet gelukt"
TransFt=.t.
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS. BEGINTRANS( ) vervangt de dBASE IV-commando's BEGIN TRANSACTION en END TRANSACTION.

Zie ook

COMMIT( ), FLOCK( ), RLOCK( ), ROLLBACK( ), SET EXCLUSIVE, SQLERROR( ), SQLMESSAGE( )

BINTYPE( )

Velden en records

Geeft het vooringestelde typenummer van een opgegeven binair veld als resultaat.

Syntaxis

BINTYPE([<*veldnaam*>])

<veldnaam>

De naam van een veld in de huidige tabel.

Beschrijving

BINTYPE( ) geeft het vooringestelde binaire typenummer van een binair veld in de huidige tabel als resultaat. Met dit commando kunt u het type van de gegevens in het veld bepalen. BINTYPE( ) geeft de volgende waarden als resultaat:

Vooringestelde binaire typenummers	Omschrijving
1 tot 32K -1 (1 tot 32,767)	Door de gebruiker gedefinieerde bestandstypen

Vooringestelde binaire typenummers	Omschrijving
32K (32,768)	.WAV-bestanden
32K + 1 (32,769)	.BMP- en .PCX-bestanden

BINTYPE( ) geeft een fout als resultaat als een niet-binair veld is opgegeven. Als het binaire veld leeg is, wordt 0 als resultaat gegeven.

Voorbeeld

In het volgende voorbeeld wordt BINTYPE( ) gebruikt om de binaire typecode voor het BMP-veld in de tabel Dieren te bepalen:

```
USE Dieren
GO 5
? BINTYPE(BMP)          && Geeft 32769 als resultaat
```

Zie ook

COPY BINARY, PLAY SOUND, REPLACE BINARY, RESTORE IMAGE

BITAND( )

Programmeren in Windows

Voert een bitsgewijze AND-bewerking uit op twee numerieke waarden en geeft de uitkomst als resultaat.

Syntaxis

BITAND(<*Nuitdr1*>, <*Nuitdr2*>)

Beschrijving

BITAND( ) wordt door gevorderde programmeurs gebruikt om binaire interpretaties te maken van numerieke waarden. Deze waarden zijn vaak het resultaat van met EXTERN gedefinieerde functies die worden gebruikt om toegang te krijgen tot hulpmiddelen in de API van Windows en andere DLL-bestanden. Bij het interpreteren van dergelijke waarden moeten vaak de individuele bits worden geanalyseerd en gemanipuleerd.

BITAND( ) vergelijkt de bits in de numerieke waarde <*Nuitdr1*> met de overeenkomstige bits in de numerieke waarde <*Nuitdr2*>. Als twee bits op dezelfde positie allebei 1 zijn, wordt het overeenkomstige bit in het resultaat ook 1 gemaakt. In alle andere gevallen is dat bit in het resultaat 0.

	Bit 1 = 1	Bit 1 = 0
Bit 2 = 1	1	0
Bit 2 = 0	0	0

Voorbeeld

De volgende procedure ontvangt informatie van de Windows-functie GetVersion( ). Deze functie levert de hoofd- en subversienummers van DOS en Windows in de vorm van een CLONG. De hoge byte van het hoge woord bevat het hoofdversienummer en de lage byte van het hoge woord bevat het subversienummer van DOS. De hoge byte van het lage woord bevat het subversienummer en de lage byte van het lage woord het subversienummer van Windows. Met de bitfuncties BITAND( ) en BITRSHIFT( ) worden de juiste waarden uit het resultaat van GetVersion( ) gehaald.

```
PROCEDURE GVer (verType)
#define HIWORD(x)     (bitand(bitrshift(x,16),2^16-1))
#define LOWORD(x)     (bitand(x,2^16-1))
#define HIBYTE(x)     (bitand(bitrshift(x,8),2^8-1))
#define LOBYTE(x)     (bitand(x,2^8-1))
#define STRHIBYTE(x)  ltrim(str(HIBYTE(x)))
#define STRLOBYTE(x)  ltrim(str(LOBYTE(x)))

* EXTERN CLONG GetVersion(CVOID)  krnl386.exe
local version,retVal
version = GetVersion( )
if upper(verType) = "DOS"
   retVal = ReeksOpmaak("%1.%2",;
      STRHIBYTE(HIWORD(version)), STRLOBYTE(HIWORD(version)))
elseif left(upper(verType),3) = "WIN"
   retVal = ReeksOpmaak("%1.%2",;
      STRLOBYTE(LOWORD(version)), STRHIBYTE(LOWORD(version)))
endif

return retVal
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

BITLSHIFT( ), BITOR( ), BITRSHIFT( ), BITSET( ), BITXOR( ), EXTERN, HTOI( ), ITOH( )

BITLSHIFT( )

Programmeren in Windows

Geeft als resultaat een getal dat wordt verkregen door de bits in een ander getal naar links te verschuiven.

Syntaxis

BITLSHIFT(<*Nuitdr1*>, <*Nuitdr2*>)

Beschrijving

BITLSHIFT( ) wordt door gevorderde programmeurs gebruikt om binaire interpretaties te maken van numerieke waarden. Deze waarden zijn vaak het resultaat van met EXTERN gedefinieerde functies die worden gebruikt om toegang te krijgen tot hulpmiddelen in de API van Windows en andere DLL-bestanden. Bij het interpreteren van dergelijke waarden moeten vaak de individuele bits worden geanalyseerd en gemanipuleerd.

BITLSHIFT( ) schuift elk bit in de numerieke waarde <*Nuitdr1*> naar links. Het aantal keren dat elk bit wordt verschoven, wordt bepaald door <*Nuitdr2*>. Elke keer dat de bits naar links worden opgeschoven, wordt het minst significante bit (het bit uiterst rechts) 0 gemaakt en gaat het meest significante bit (het bit uiterst links) verloren.

Voorbeeld

In het volgende voorbeeld wordt BITLSHIFT( ) gebruikt om de bits in 00001111 binair (15 decimaal) 4 bits naar links op te schuiven. Het resultaat van deze bewerking is 11110000, hetgeen gelijk is aan 240 decimaal:

```
? BITLSHIFT(15,4)      && Geeft 240 als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

BITAND( ), BITOR( ), BITRSHIFT( ), BITSET( ), BITXOR( ), EXTERN, HTOI( ), ITOH( )

BITOR( )

Programmeren in Windows

Voert een bitsgewijze OR-bewerking uit op twee numerieke waarden en geeft de uitkomst als resultaat.

Syntaxis

BITOR(<*Nuitdr1*>, <*Nuitdr2*>)

Beschrijving

BITOR( ) wordt door gevorderde programmeurs gebruikt om binaire interpretaties te maken van numerieke waarden. Deze waarden zijn vaak het resultaat van met EXTERN gedefinieerde functies die worden gebruikt om toegang te krijgen tot hulpmiddelen in de API van Windows en andere DLL-bestanden. Bij het interpreteren van dergelijke waarden moeten vaak de individuele bits worden geanalyseerd en gemanipuleerd.

BITOP( ) vergelijkt de bits in de numerieke waarde <*Nuitdr1*> met de overeenkomstige bits in de numerieke waarde <*Nuitdr2*>. Als één van beide of beide bits gelijk zijn aan 1,

wordt het overeenkomstige bit in de uitkomst ook 1 gemaakt. Als geen van beide bits gelijk is aan 1, wordt dat bit in de uitkomst 0 gemaakt.

	Bit 1 = 1	**Bit 1 = 0**
Bit 2 = 1	1	1
Bit 2 = 0	1	0

Voorbeeld

In het volgende voorbeeld wordt BITOR( ) gebruikt om de bits in de binaire versies van de getallen 64 (1000000 binair) en 48 (110000 binair) te vergelijken. Her resultaat is 1110000 (112 decimaal):

```
? BITOR(64,48)              && Geeft 112 als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

BITAND( ), BITLSHIFT( ), BITRSHIFT( ), BITSET( ), BITXOR( ), EXTERN, HTOI( ), ITOH( )

BITRSHIFT( )

Programmeren in Windows

Geeft als resultaat een getal dat wordt verkregen door de bits in een ander getal naar rechts te verschuiven.

Syntaxis

BITRSHIFT(<*Nuitdr1*>, <*Nuitdr2*>)

Beschrijving

BITRSHIFT( ) wordt door gevorderde programmeurs gebruikt om binaire interpretaties te maken van numerieke waarden. Deze waarden zijn vaak het resultaat van met EXTERN gedefinieerde functies die worden gebruikt om toegang te krijgen tot hulpmiddelen in de API van Windows en andere DLL-bestanden. Bij het interpreteren van dergelijke waarden moeten vaak de individuele bits worden geanalyseerd en gemanipuleerd.

BITRSHIFT( ) schuift elk bit in de numerieke waarde <*Nuitdr1*> naar rechts. Het aantal keren dat elk bit wordt verschoven, wordt bepaald door <*Nuitdr2*>. Elke keer dat de bits naar rechts worden opgeschoven, wordt het meest significante bit (het bit uiterst links) 0 gemaakt en gaat het minst significante bit (het bit uiterst rechts) verloren.

Voorbeeld

In het volgende voorbeeld wordt BITRSHIFT( ) gebruikt om de bits in de binaire waarde 1010000 (80 decimaal) 4 bits naar rechts op te schuiven. Het resultaat van deze bewerking is 00000101, hetgeen gelijk is aan 5 decimaal:

```
? BITRSHIFT(80,4)      && Geeft 5 als resultaat
```

Zie BITAND( ) voor een ander voorbeeld met BITRSHIFT( ).

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

BITAND( ), BITLSHIFT( ), BITOR( ), BITSET( ), BITXOR( ), EXTERN, HTOI( ), ITOH( )

BITSET( )

Programmeren in Windows

Geeft .T. als resultaat als een opgegeven bit in een numerieke waarde gelijk is aan 1.

Syntaxis

BITSET (<*Nuitdr1*>, <*Nuitdr2*>)

<Nuitdr1>

Een numerieke waarde om te evalueren.

<Nuitdr2>

Een decimaal geheel getal van 0 tot 31 dat een positie voorstelt in <*Nuitdr1*>. De bits worden geteld van rechts naar links. De binaire voorstelling van 3 is bijvoorbeeld 00000011; het bit met het nummer 0 heeft de waarde 1 en het bit met het nummer 2 de waarde 0.

Beschrijving

BITSET( ) wordt door gevorderde programmeurs gebruikt om binaire interpretaties te maken van numerieke waarden. Deze waarden zijn vaak het resultaat van met EXTERN gedefinieerde functies die worden gebruikt om toegang te krijgen tot hulpmiddelen in de API van Windows en andere DLL-bestanden. Bij het interpreteren van dergelijke waarden moeten vaak de individuele bits worden geanalyseerd en gemanipuleerd.

BITSET( ) evalueert de waarde <*Nuitdr1*> en geeft .T. als resultaat als het bit op positie <*Nuitdr2*> de waarde 1 heeft en .F. als het de waarde 0 heeft.

Voorbeeld

In de volgende voorbeelden wordt BITSET( ) gebruikt om de waarde te bepalen van verschillende bits in het getal 229 (11100101 binair):

```
mGetal=229
? BITSET(mGetal,0)      && Geeft .T. als resultaat
? BITSET(mGetal,1)      && Geeft .F. als resultaat
? BITSET(mGetal,6)      && Geeft .T. als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

BITAND( ), BITLSHIFT( ), BITOR( ), BITRSHIFT( ), BITXOR( ), EXTERN

BITXOR( )

Programmeren in Windows

Voert een bitsgewijze exclusieve OR-bewerking (XOR-bewerking) uit op twee numerieke waarden en geeft de uitkomst als resultaat.

Syntaxis

BITXOR(<*Nuitdr1*>, <*Nuitdr2*>)

Beschrijving

BITXOR( ) wordt door gevorderde programmeurs gebruikt om binaire interpretaties te maken van numerieke waarden. Deze waarden zijn vaak het resultaat van met EXTERN gedefinieerde functies die worden gebruikt om toegang te krijgen tot hulpmiddelen in de API van Windows en andere DLL-bestanden. Bij het interpreteren van dergelijke waarden moeten vaak de individuele bits worden geanalyseerd en gemanipuleerd.

BITXOR( ) vergelijkt de bits in de numerieke waarde <*Nuitdr1*> met de overeenkomstige bits in de numerieke waarde <*Nuitdr2*>. Als één van beide (en niet meer dan één) van de twee bits op dezelfde positie de waarde 1 heeft, wordt het overeenkomstige bit in de uitkomst ook 1 gemaakt. In elk ander geval wordt het bit in de uitkomst 0 gemaakt.

	Bit 1 = 1	**Bit 1 = 0**
Bit 2 = 1	0	1
Bit 2 = 0	1	0

Deze bewerking wordt een exclusieve OR-bewerking genoemd omdat alleen als één en niet meer dan één bit de waarde 1 heeft, het overeenkomstige bit in de uitkomst 1 wordt gemaakt.

Voorbeeld

In het volgende voorbeeld wordt BITXOR( ) gebruikt om de binaire voorstelling van 90 (01011010) te vergelijken met die van 214 (11010110). Het resultaat is 140 (10001100):

```
mGetal1=90
mGetal2=214
? BITXOR(mGetal1,mGetal2)     && Geeft 140 als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

BITAND( ), BITLSHIFT( ), BITOR( ), BITSET( ), BITRSHIFT( ), HTOI( ), ITOH( )

BLANK

Velden en records

Maakt velden in records leeg.

Syntaxis

BLANK
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[FIELDS
<*veldenlijst*> | [LIKE <*filter1*>] [EXCEPT <*filter2*>]]
[REINDEX]

<bereik>

Het aantal records dat leeggemaakt moet worden. RECORD <*n*> geeft een enkel record aan door middel van zijn recordnummer. NEXT <*n*> geeft *n* records aan, te beginnen bij het huidige record. ALL specificeert alle records. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR *<voorwaarde1>*
WHILE *<voorwaarde2>*

Bepaalt welke records worden beïnvloed door BLANK. FOR beperkt BLANK tot records die voldoen aan <*voorwaarde1*>. WHILE begint bij het huidige record en gaat verder met elk volgend record tot <*voorwaarde2*> onwaar wordt.

FIELDS *<veldenlijst>* | LIKE *<filter1>* | EXCEPT *<filter2>*

De velden die leeggemaakt moeten worden. Zonder FIELDS, vervangt BLANK alle veldwaarden. Als u FIELDS LIKE <*filter1*> opgeeft, worden alleen de velden leeggemaakt die overeenkomen met <*filter1*>. Omgekeerd kunt u FIELDS EXCEPT <filter2> opgeven om alle velden leeg te maken behalve de velden waarvan de naam overeenkomt met <*filter2*>.

REINDEX
Stelt alle geopende indexen opnieuw samen nadat BLANK is uitgevoerd. Zonder REINDEX worden telkens nadat een record is leeggemaakt alle geopende indexen opnieuw samengesteld.

Beschrijving
Gebruik BLANK om velden in de huidige tabel leeg te maken. Als de waarde van een veld met BLANK is vervangen door een lege waarde, geven EMPTY( ) en ISBLANK( ) .T. als resultaat. BLANK vult een bestaand record met dezelfde waarden als APPEND BLANK. Nadat een record of een verzameling records leeg is gemaakt, worden de geopende indexen bijgewerkt.

In de volgende tabel ziet u welke lege waarden voor de verschillende gegevenstypen worden weergegeven:

Gegevenstype	Weergegeven waarde
Teken	Spaties
Datum	" - - "
Zwevend	Lege waarde (niet nul)
Logisch	.F.
Numeriek	Lege waarde (niet nul)
Binair, Memo en OLE	Geen tekst of afbeelding

Gebruik BLANK om logische, zwevende en numerieke waarden te vervangen door echte lege waarden. Met REPLACE kan dat niet. Als u REPLACE gebruikt om een numerieke of zwevende waarde te vervangen door 0, wordt die waarden behandeld als nul en niet als een lege waarde. Het onderscheid tussen lege waarden en de waarde 0 kan van belang zijn als u commando's gebruikt als AVERAGE en CALCULATE.

Voorbeeld
In het volgende voorbeeld wordt BLANK gebruikt om records die in het bestand met de natuurlijke volgorde zijn gemarkeerd voor verwijdering, opnieuw te gebruiken door het record leeg te maken, de verwijderingsmarkering weg te halen en het commando EDIT te geven:

```
USE Klanten
? Hergebruik()
EDIT

FUNCTION Hergebruik
SET DELETED OFF
LOCATE FOR DELETED()
  IF .NOT. FOUND()
    APPEND BLANK
  ELSE
    BLANK            && Veldwaarden leegmaken
    RecNum=RECNO()
    RECALL RECNO()
    GOTO RecNum
```

```
  ENDIF
RETURN .T.
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

APPEND, ISBLANK( ), EMPTY( ), REPLACE

BOF( )

Velden en records

Stelt vast of de recordaanwijzer in een tabel aan het begin van het bestand staat.

Syntaxis

BOF([*<alias>*])

<alias>

Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

BOF( ) geeft .T. als resultaat als de recordaanwijzer voor het eerste logische record in de tabel in het opgegeven werkgebied staat, zo niet, dan wordt .F. als resultaat gegeven. Als u bijvoorbeeld SKIP -1 opgeeft en de recordaanwijzer staat op het eerste record, geeft BOF( ) .T. als resultaat.

Als in het opgegeven werkgebied geen geopende tabel aanwezig is, geeft BOF( ) ook .F. als resultaat.

Voorbeeld

In het volgende voorbeeld wordt een formulier gedefinieerd met twee knoppen om de recordaanwijzer te verplaatsen. BOF( ) en EOF( ) worden gebruikt om foutmeldingen aan het begin en het eind van de tabel te voorkomen:

```
SET PROCEDURE TO PROGRAM(1) ADDITIVE
USE Klanten
DEFINE FORM F1
DEFINE PUSHBUTTON Knop1 OF F1 AT 10,5;
  PROPERTY Text "Vorige",;
    Width 20, OnClick Terug
DEFINE PUSHBUTTON Knop2 OF F1 AT 10,35;
  PROPERTY Text "Volgende",;
    Width 20, OnClick Heen
OPEN FORM F1
```

```
FUNCTION Terug
IF .NOT. BOF()
  SKIP-1
ENDIF
RETURN .T.

FUNCTION Heen
IF .NOT. EOF()
  SKIP
ENDIF
RETURN .T.
```

Zie ook

EOF( ), RECNO( ), SKIP

BOOKMARK( )

Velden en records

Geeft een bladwijzer voor het huidige record als resultaat. Bladwijzers worden in tabellen die geen recordnummers ondersteunen (zoals Paradox- en SQL-tabellen), gebruikt in plaats van recordnummers.

Syntaxis

BOOKMARK( )

Beschrijving

BOOKMARK( ) geeft een waarde als resultaat voor het huidige record in tabellen die geen recordaanwijzers ondersteunen. In combinatie met het commando GO kunt u bladwijzers gebruiken om naar bepaalde gemarkeerde records te gaan.

De waarde die door BOOKMARK( ) als resultaat wordt gegeven, heeft een speciaal, niet-afdrukbaar gegevenstype dat Bookmark (bladwijzer) wordt genoemd. Als in het huidige werkgebied geen geopende tabel aanwezig is, geeft BOOKMARK( ) een lege bladwijzer als resultaat.

Bladwijzerwaarden kunnen worden toegepast in alle commando's en functies die anders een recordnummer gebruiken en bij alle relationele operatoren, =, <, <=, > en >= voor vergelijkingen.

Voorbeeld

In het volgende voorbeeld kan de gebruiker tijdens het bladeren op *F9* drukken om een record te markeren zodat het record kan worden bewerkt nadat het tabelbewerkingsvenster is verlaten:

```
SET DBTYPE TO PARADOX
SET TALK OFF
PUBLIC Mark1
USE Klanten
```

```
ON KEY LABEL F9 DO GaTerug
BROWSE
GOTO Mark1
EDIT
RETURN

PROCEDURE GaTerug
Mark1=BOOKMARK()
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

GO, RECNO( )

BROWSE

Velden en records

Met dit commando kunt u records in een tabelopmaak weergeven en bewerken.

Syntaxis

BROWSE
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[COLOR [<*gewone tekst*>]
[, [<*speciale tekst*>]
[, [<*buitenrandkleur*>]
[, [<*achtergrondkleur*>]]]]]
[FIELDS <*veld1*> [<*veldoptielijst1*>] |
<*rekenveld1*> = <*uitdr1*> [<*rekenveldoptielijst1*>]
[, <*veld2*> [<*veldoptielijst2*>] |
<*rekenveld2*> = <*uitdr2*> [<*rekenveldoptielijst2*>] ...]]
[FORMAT]
[FREEZE <*veld3*>]
[KEY <*uitdr3*>[, <*uitdr4*>] [EXCLUDE]]
[LOCK <*Nuitdr2*>]
[NOAPPEND]
[NODELETE]
[NOEDIT | NOMODIFY]
[NOFOLLOW]
[NOINIT]
[NOORGANIZE]
[NORMAL]
[NOTOGGLE]
[NOWAIT]
[TITLE <*Tuitdr1*>]

[WIDTH <*Nuitdr3*>]
[WINDOW <*vensternaam*>]

<*bereik*>

Het aantal records om doorheen te bladeren. ALL specificeert alle records. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR <*voorwaarde1*>
WHILE <*voorwaarde2*>

Bepaalt welke records worden beïnvloed door BROWSE. FOR beperkt BROWSE tot records die voldoen aan <*voorwaarde1*>. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang <*voorwaarde2*> waar is.

COLOR [<*gew. tekst*>] [, [<*spec. tekst*>] [, [<*buitenrand*>] [, [<*achtergrond*>]]]]

Geeft de kleuren aan van gewone tekst, speciale tekst en de omtrek van het tabelrecordsvenster. Om de kleuren van deze elementen afzonderlijk op te geven gebruikt u de opties <*gewone tekst*>, <*speciale tekst*> en <*buitenrand*>. U kunt ook de optie <*achtergrond*> gebruiken als u een scherm met een uniforme achtergrond hebt.

<*gewone tekst*>	Kleurenattributen voor commandomeldingen en schermuitvoer. De uitvoer van de commando's ? en @ ... SAY verschijnt bijvoorbeeld in normale tekst.
<*speciale tekst*>	Kleurenattributen voor gebieden met speciale tekst, zoals @...GET-velden en gemarkeerde BROWSE-gegevenscellen.
<*buitenrand*>	De kleurattributen van de omtrek van het gebied waar de tekst wordt weergegeven.
<*achtergrond*>	Kleurenattributen voor de achtergrond van schermen met een uniforme achtergrond (zoals monochrome schermen). <*achtergrond*> bestaan uit twee parameters: een achtergrondkleur en een attribuut.

De attributen <*gewone tekst*> en <*speciale tekst*> bestaan uit drie instellingen: een voorgrondkleur, een achtergrondkleuren een optionele kleur waarmee een gemengde (gearceerde) achtergrond wordt gemaakt. Plaats schuine strepen (/) tussen de verschillende instellingen.

Zie SET COLOR TO en SET COLOR OF voor meer informatie over kleureninstellingen.

FIELDS <*veld1*> [<*veldoptielijst1*>] |
<*rekenveld1*> = <*uitdr1*> [<*rekenveldoptielijst1*>]

[, <veld2> [<veldoptielijst2>] |
<rekenveld2> = <uitdr2> [<rekenveldoptielijst2>] ...]]

Geeft in het tabelrecordsvenster de opgegeven velden weer in de volgorde waarin ze staan. De opties voor *<veldoptielijst1>*, *<veldoptielijst2>*, die van toepassing zijn op *<veld1>*, *<veld2>* enzovoort, bepalen de manier waarop deze velden worden weergegeven. Deze opties zijn als volgt:

\<*kolombreedte kolom*>	De breedte van de kolom waarin *<veld1>* wordt weergegeven als *<veld1>* het type teken heeft
\B = *<uitdr1>*, *<uitdr2>* [\F]	Optie RANGE; zorgt dat elke waarde die wordt ingevoerd in *<veld1>* binnen het bereik van *<uitdr1>* tot en met *<uitdr2>* valt
	Optie RANGE REQUIRED; de optie \F voorkomt dat een eerder ingevoerde waarde wordt geaccepteerd als die niet binnen het bereik van *<uitdr1>* tot en met *<uitdr2>* valt
\H = *<Tuitdr>*	Optie HEADER; zorgt dat boven de kolom voor het veld in het tabelrecordsvenster de veldnaam wordt vervangen door *<Tuitdr>*
\P = *<Tuitdr>*	Optie PICTURE; geeft *<veld1>* weer volgens de PICTURE- of FUNCTION-clausule *<Tuitdr>*
\R	Optie READ-ONLY; bepaalt dat *<veld1>* alleen kan worden gelezen en niet bewerkt
\V = *<voorwaarde>* [\F] [\E = *<Tuitdr>*]	Optie VALID; staat alleen toe dat voor *<veld1>* een nieuwe waarde kan worden opgegeven als *<voorwaarde>* .T. als resultaat heeft
	Optie VALID REQUIRED; de optie \F voorkomt dat de cursor *<veld1>* verlaat en het bewerken wordt afgesloten zolang *<voorwaarde>* niet .T. als resultaat geeft
	Optie ERROR MESSAGE; \E = *<Tuitdr>* zorgt dat *<Tuitdr>* wordt weergegeven als *<voorwaarde>* .F. als resultaat geeft
\W = *<voorwaarde>*	Optie WHEN; zorgt dat *<veld1>* alleen kan worden bewerkt als *<voorwaarde>* .T. als resultaat geeft

Opmerking U mag het teken "/" ook gebruiken als u maar één enkele optie opgeeft in een veldoptielijst.

Rekenvelden waarvoor alleen leesrecht is toegekend, bestaan uit een toegewezen veldnaam en een uitdrukking die de waarde van het rekenveld als resultaat geeft, zoals

*Commissie = Percentage * Verkoopprijs*. Opties voor rekenvelden zijn van invloed op de manier waarop deze velden worden weergegeven. Deze opties zijn als volgt:

\<*kolombreedte kolom*>	De breedte van de kolom waarin <*rekenveld1*> wordt weergegeven
\H = <*Tuitdr*>	Zorgt dat <*Tuitdr*> in het tabelrecordsvenster boven de rekenveldkolom verschijnt, waardoor de naam van het rekenveld wordt vervangen.

FORMAT
Zorgt dat BROWSE invoer accepteert en weergeeft aan de hand van de specificaties in een indelingsbestand dat is geopend met SET FORMAT. Ingevoerde gegevens moeten voldoen aan alle PICTURE-, FUNCTION-, RANGE- en VALID-clausules in het indelingsbestand.

FREEZE <*veld3*>
Zorgt dat alleen <*veld3*> kan worden bewerkt, hoewel ook andere velden zichtbaar zijn.

KEY <*uitdr3*> [,<*uitdr4*>] [EXCLUDE]
Als de tabel over een hoofdindex beschikt, worden alleen records weergegeven waarvan de sleutelveldwaarden gelijk zijn aan of komen na <*uitdr3*>, of binnen het bereik van <*uitdr3*> tot en met <*uitdr4*> vallen. EXCLUDE bepaalt dat het opgegeven bereik juist moet worden uitgesloten.

LOCK <*Nuitdr2*>
Zorgt dat de eerste <*Nuitdr2*> velden op het scherm blijven staan, als u de cursor naar rechts gelegen velden verplaatst.

NOAPPEND
Voorkomt dat in het tabelrecordsvenster records worden toegevoegd.

NODELETE
Voorkomt dat in het tabelrecordsvenster records worden gemarkeerd voor verwijdering.

NOEDIT | NOMODIFY
Voorkomt dat in het tabelrecordsvenster records worden gewijzigd.

NOFOLLOW
Als de huidige tabel een hoofdindex heeft, blijft de cursor op zijn plaats als u het sleutelveld in een record verandert, en volgt die niet het record naar de nieuwe plaats in de indexvolgorde. Zonder deze optie volgt de recordaanwijzer het record naar zijn nieuwe plaats.

NOINIT
Zorgt dat BROWSE de opties handhaaft die bij het vorige BROWSE-commando zijn opgegeven. Gebruik NOINIT als BROWSE in een programma meerdere keren wordt gebruikt of als u het commando meer dan eens toepast in het commandovenster, en u wilt de eerder opgegeven opties handhaven. Geef de eerste keer dat u BROWSE gebruikt, de gewenste opties op en gebruik de volgende keren BROWSE NOINIT.

NOORGANIZE
Schakelt de opties uit om records te indexeren, sorteren en verwijderen.

NORMAL
Als BROWSE wordt gegeven vanuit een actief venster, zorgt deze optie dat het tabelrecordsvenster wordt getoond in de normale modus (volledig scherm) met de standaard- of de gedefinieerde kleurenset, waarbij de voor het venster gedefinieerde kleuren worden genegeerd. Als u BROWSE verlaat, wordt teruggekeerd naar het actieve venster. Zonder NORMAL verschijnen de tabelrecords in het actieve venster.

NOTOGGLE
Voorkomt schakelen tussen de modi voor bladeren en bewerken.

NOWAIT
Zet de uitvoering van een programma voort nadat een tabelrecordsvenster is weergegeven; anders wordt de uitvoering opgeschort tot het tabelrecordsvenster is gesloten.

TITLE <*Tuitdr1*>
Zorgt dat <*Tuitdr1*> verschijnt als titel van het tabelrecordsvenster.

WIDTH <*Nuitdr3*>
Geeft de weergavebreedte van tekenvelden in het tabelrecordsvenster aan. Als een veld breder is dan de opgegeven breedte, kunt u het binnen de opgegeven breedte verschuiven. Het argument <*Nuitdr3*> moet een positief getal als resultaat geven.

WINDOW <*vensternaam*>
Activeert het venster <*vensternaam*> en geeft de tabelrecords weer in het venster.

Beschrijving
Het commando BROWSE voorziet in een interactieve, venster-geörienteerde omgeving voor het weergeven en bewerken van meer dan één record tegelijk. Gebruik het commando SET RELATION om velden te bekijken uit records in gekoppelde tabellen. Zolang BROWSE actief is en u hebt de optie NOTOGGLE opgegeven, kunt u op *F2* drukken of **Beeld | Bewerken** kiezen om een enkel record weer te geven in de bewerkingsmodus.

Tijdens het bladeren kunt u tussen records manoeuvreren door op Pijl-omhoog en Pijl-omlaag te drukken om één record omhoog of omlaag te gaan en op *PgUp* en *PgDn* om

met een venster omhoog of omlaag te gaan. U kunt ook de vensterstuurelementen en de muis gebruiken, of een keuze maken uit de beschikbare menu-opdrachten voor het uitvoeren van bewerkingen met BROWSE. Raadpleeg het *Handboek* voor meer informatie over het bewerken van gegevens en het manoeuvreren in het tabelrecordsvenster.

Als u klaar bent met het bewerken van een tabel, drukt u op *Ctrl-W* om het huidige record te verlaten en de wijzigingen op te slaan. U kunt ook **Bestand | Record opslaan** en sluiten kiezen. Druk op *Ctrl-Q*, kies **Bestand | Afbreken en sluiten** of dubbelklik op het Systeemmenu om het record te verlaten zonder de wijzigingen op te slaan. Als u BROWSE of EDIT in een programma gebruikt, wordt de besturing doorgegeven aan de commandoregel direct na de commandoregel met BROWSE of EDIT.

Voorbeeld

In het volgende voorbeeld wordt BROWSE gebruikt om bepaalde velden uit twee verwante tabellen weer te geven, eigen kolomtitels toe te voegen en velden met uitsluitend leesrecht op te geven:

```
USE Contact ORDER CompCode IN SELECT()
USE Bedrijf IN SELECT()
SELECT Bedrijf
SET RELATION TO CompCode INTO Contact
BROWSE FIELDS ;
   Contact->CompCode /R /H="Bedrijfscode", ;
   Bedrijf->Bedrijf /R, ;
   Contact->Contact /R /H="Contactpersoon", ;
   Bedrijf->Straat1 /R, Bedrijf->Straat2 /R, ;
   Bedrijf->Plaats /R, Bedrijf->Regio /R
CLOSE ALL
```

Zie ook

APPEND, EDIT, SET FIELDS, SET FORMAT, SET MEMOWIDTH, SET RELATION, SET WINDOW OF MEMO

CALCULATE

Tabelindeling

Voert financiële en statistische bewerkingen uit op waarden in records in de huidige tabel.

Syntaxis

CALCULATE <*functielijst*>
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[TO <*variabelenlijst*> | TO ARRAY <*array-naam*>]

<functielijst>

U kunt een of meer van de volgende functies gebruiken:

Functie	Doel	Resultaatwaarde
AVG(*<Nuitdr>*)	Berekent het gemiddelde van de opgegeven numerieke of zwevende uitdrukking.	CALCULATE AVG( ) geeft een zwevende waarde als resultaat.
CNT( )	Telt het aantal records in de huidige tabel.	CALCULATE CNT( ) geeft een numerieke waarde als resultaat.
MAX(*<Tuitdr>* | *<Nuitdr>* | *<Duitdr>*)	Bepaalt de maximumwaarde van de opgegeven numerieke, zwevende, teken- of datumuitdrukking.	CALCULATE MAX( ) geeft hetzelfde gegevenstype als resultaat als de opgegeven uitdrukking.
MIN(*<Tuitdr>* | *<Nuitdr>* | *<Duitdr>*)	Bepaalt de minimumwaarde van de opgegeven numerieke, zwevende, teken- of datumuitdrukking.	CALCULATE MIN( ) geeft hetzelfde gegevenstype als resultaat als de opgegeven uitdrukking.
NPV(*<Nuitdr1>*, *<Nuitdr2>*[, *<Nuitdr3>*])	Berekent de netto huidige waarde van de numerieke of zwevende waarden in *<Nuitdr2>*. *<Nuitdr1>* is het periodieke rentepercentage, in de vorm van een decimaal getal. *<Nuitdr3>* is de investering bij aanvang en bestaat in de meeste gevallen uit een negatief getal.	CALCULATE NPV( ) geeft een zwevende waarde als resultaat.
STD(*<Nuitdr>*)	Berekent de standaarddeviatie van de opgegeven numerieke of zwevende uitdrukking.	CALCULATE STD( ) geeft een zwevende waarde als resultaat.
SUM(*<Nuitdr>*)	Berekent de som van de opgegeven numerieke of zwevende uitdrukking.	CALCULATE SUM( ) geeft een zwevende waarde als resultaat.
VAR(*<Nuitdr>*)	Berekent de variantie van het opgegeven numerieke veld.	CALCULATE VAR( ) geef een zwevende waarde als resultaat.

<bereik>

Het aantal records voor de berekening. RECORD *<n>* geeft een enkel record aan door middel van zijn recordnummer. NEXT *<n>* geeft *n* records aan, te beginnen bij het huidige record. ALL specificeert alle records. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR *<voorwaarde1>*
WHILE *<voorwaarde2>*

Bepaalt welke records worden beïnvloed door CALCULATE. FOR beperkt CALCULATE tot records die voldoen aan *<voorwaarde1>*, vanaf het eerste record in de tabel of weergave tot het eind van het bestand. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang *<voorwaarde2>* waar is.

TO *<variabelenlijst>* | TO ARRAY *<array-naam>*

De optie TO *<variabelenlijst>* initialiseert de geheugenvariabelen in *<variabelenlijst>* en slaat de resultaten van CALCULATE op in de variabelen. De optie TO ARRAY *<array-naam>* slaat de resultaten van CALCULATE op in de bestaande array *<array-naam>*.

Beschrijving

Gebruik CALCULATE met een of meer van de acht bijbehorende functies om de som, het maximum, het minimum, het gemiddelde, de variantie, de standaarddeviatie of de netto huidige waarde van opgegeven uitdrukkingen in de huidige tabel te berekenen en op te slaan. CALCULATE kan ook het aantal records in de huidige tabel als resultaat geven. Deze speciale functies (met uitzondering van de functie MAX( )) kunnen alleen worden gebruikt in combinatie met CALCULATE.

Als SET TALK is ingeschakeld (ON), geeft CALCULATE de resultaten weer in het resultaatvenster.

CALCULATE kan dezelfde functie gebruiken voor verschillende uitdrukkingen of verschillende functie voor dezelfde uitdrukking. Als de tabel bijvoorbeeld de velden Salaris en Bonus bevat, kunt u de instructie CALCULATE SUM(Salaris), SUM(Bonus), AVG(Salaris), AVG(12*(Salaris + Bonus)) geven.

U kunt waarden berekenen in een ander werkgebied dan het huidige door een relatie tot stand te brengen tussen de werkgebieden.

CALCULATE slaat de resultaten op in geheugenvariabelen of in een bestaande array. De resultaten worden opgeslagen in de volgorde waarin de functies zijn opgegeven. Als u de resultaten opslaat in geheugenvariabelen, moet het aantal geheugenvariabelen en het aantal functies in instructie met CALCULATE gelijk zijn.

Als u CALCULATE TO ARRAY *<array-naam>* gebruikt, wordt elk resultaat opgeslagen in een array-element. De resultaten van CALCULATE kunnen worden opgeslagen in een multidimensionale array. Als u bijvoorbeeld DECLARE test[3,2] opgeeft, kan CALCULATE de resultaten van maximaal zes functies opslaan in test[1,1], test[1,2], test[2,1], test[2,2], test[3,1], test[3,2], in die volgorde.

U kunt de resultaten ook opslaan in een array zonder nadrukkelijk het sleutelwoord ARRAY te vermelden door te verwijzen naar de indextekens van de array. De instructie CALCULATE AVG(Salaris), MAX(Salaris) TO test[2], test[3] slaat bijvoorbeeld de gemiddelde van en de maximumwaarde in het veld salaris op in de opgegeven twee array-elementen.

Als u CALCULATE...TO ARRAY gebruikt, moet u een array definiëren die minimaal hetzelfde aantal elementen bevat als er functies staan in de instructie. neem in de instructie met CALCULATE geen indextekens van de array op. Als de huidige tabel bijvoorbeeld een numeriek veld, Numveld, bevat en u wilt de minimum- en maximumwaarde van dat veld opslaan in een array *minmax*, kunt u achtereenvolgens DECLARE minmax[2] en CALCULATE MIN(Numveld), MAX(Numveld) TO ARRAY minmax gebruiken.

CALCULATE behandelt een leeg numeriek of zwevend veld als een veld met de waarde 0 en neemt dat veld als zodanig mee in de berekeningen. Als u bijvoorbeeld het gemiddelde berekent van een numeriek veld in een tabel met tien records waarvan er vijf leeg zijn, deelt CALCULATE de som door 10 om het gemiddelde te bepalen. Verder is het zo dat als u van hetzelfde numerieke veld het minimum wilt bepalen, CALCULATE 0 als resultaat geeft. Als u de lege velden wilt uitsluiten als CALCULATE gebruikt, moet u een voorwaarden opgeven als FOR .NOT. ISBLANK(Numveld).

Hoewel u de commando's SUM en AVERAGE kunt gebruiken om de som of het gemiddelde te bepalen, is CALCULATE sneller omdat dit commando de tabel slechts één keer doorloopt om alle opgegeven berekeningen uit te voeren.

Voorbeeld

In het volgende voorbeeld wordt CALCULATE gebruikt om resultaten in geheugenvariabelen te plaatsen:

```
USE Bedrijf
PUBLIC GM,AT,MX,MN,VR,SD
CALCULATE AVG(VerkTotNu),CNT(),MAX(VerkTotNu),;
  MIN(VerkTotNu),SUM(VerkTOtNu),VAR(VerkTOtNu),;
  STD(VerkTOtNu);
  TO  GM,AT,MX,MN,VR,SD
? GM,AT,MX,MN,VR,SD
WAIT
* Geeft Gemiddelde, Aantal records, Maximum,
* Minimum, Variantie, Standaarddeviatie weer
```

In het volgende voorbeeld wordt CALCULATE gebruikt om resultaten in een array te plaatsen:

```
USE Bedrijf
PUBLIC ARRAY Resultaten[10]
CALCULATE AVG(VerkTOtNu),CNT(),MAX(VerkTOtNu),;
  MIN(VerkTOtNu),SUM(VerkTOtNu),VAR(VerkTOtNu),;
  STD(VerkTOtNu);
  TO ARRAY Resultaten
FOR i=1 TO 7
  ? i,Resultaten[i]
NEXT i
* Geeft de resultaten weer
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

AVERAGE, DECLARE, MAX( ), MIN( ), SET FIELDS, SET RELATION, SUM

CANCEL

Programma's

Breekt de uitvoering van een programma af, sluit de programmabestanden, wist alle als PRIVATE gedefinieerde geheugenvariabelen en geeft de besturing terug aan het commandovenster.

Syntaxis

CANCEL

Beschrijving

Gebruik CANCEL om de uitvoering van een programma te annuleren. Als in een programmabestand CANCEL wordt aangetroffen, worden alle volgende regels in het bestand gegenereerd en wordt de uitvoering van het programma afgebroken. U kunt CANCEL ook gebruiken in het commandovenster als een programma is onderbroken (bijvoorbeeld met SUSPEND). Ook in dat geval wordt de uitvoering van het programma geannuleerd. Als een programma wordt geannuleerd, worden alle als PRIVATE gedefinieerde geheugenvariabelen voor dat programma gewist en wordt de besturing teruggegeven aan het commandovenster.

Voorbeeld

In het volgende voorbeeld wordt CANCEL gebruikt om de uitvoering van een programma af te breken als in een formulier op de knop Stoppen is geklikt:

```
DEFINE FORM Hoofd FROM 2,2 TO 20,30;
  PROPERTY MDI .F.
DEFINE PUSHBUTTON Stoppen OF Hoofd AT 13,10;
  PROPERTY;
    TEXT  "Stoppen",;
    OnClick {;Cancel}
READMODAL(Hoofd.Stoppen)
```

Zie ook

DO, QUIT, RESUME, RETRY, RETURN, SUSPEND

CATALOG( ) — Tabellen

Geeft de naam van het huidige catalogusbestand als resultaat.

Syntaxis

CATALOG( )

Beschrijving

CATALOG( ) geeft de naam als resultaat van de huidige catalogus die is geopend met het commando SET CATALOG TO of met Navigator. Als er geen catalogus is geopend, geeft CATALOG( ) een lege tekenreeks ("") als resultaat.

Voorbeeld

In het volgende voorbeeld wordt CATALOG( ) gebruikt om de naam van de huidige, geopende catalogus te bepalen:

```
SET CATALOG TO Leren
catnaam = CATALOG()
USE (catnaam) IN SELECT() NOUPDATE AGAIN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CREATE CATALOG, SELECT, SET( ), SET CATALOG, SET TITLE, USE

CD

Stations- en bestandsfuncties

Met dit commando wijzigt u het standaardstation of de standaarddirectory.

Syntaxis

CD
[<*pad*>]

<*pad*>

Een tekenuitdrukking die het standaardpad aangeeft. Begin <*pad*> met een schuine streep naar links (\) om aan te geven dat het pad begint in de hoofddirectory van het station (zoals C:\). Als <*pad*> niet begint met een schuine streep naar links of met .. (twee punten), begint het pad in de huidige directory.

Beschrijving

CD werkt op dezelfde manier als de DOS-opdrachten CD en CHDIR. U kunt met dit commando de huidige directory veranderen zonder terug te gaan naar DOS. Tevens kunt met CD het actieve station wijzigen. Als u niet zeker weet of een bepaald station geldig is, kunt u VALIDDRIVE( ) gebruiken. De huidige directory verschijnt in Navigator.

CD .. (twee punten) verandert de directory in de directory die zich één niveau boven de oorspronkelijke directory bevindt. CD zonder de optie <*pad*> geeft het huidige station en de huidige directory weer.

U kunt ook toegang krijgen tot bestanden in andere directory's met het commando SET PATH. U kunt een of meer zoekpaden opgeven. Als een bestand niet wordt aangetroffen in de huidige directory, wordt in de directory's in de zoekpaden gezocht. Gebruik SET PATH als de bestanden van een applicatie zich in verschillende directory's bevinden.

CD werkt op min of meer dezelfde wijze als SET DIRECTORY, behalve dat SET DIRECTORY TO (zonder argument) de standaardwerkdirectory herstelt en niet de huidige directory weergeeft. Zie SET DIRECTORY voor meer informatie.

Voorbeeld

In het volgende voorbeeld wordt de huidige directory opgeslagen. Vervolgens wordt enkele keren de directory gewijzigd en tenslotte wordt de oorspronkelijke directory weer hersteld:

```
* Bij dit voorbeeld wordt er van uitgegaan
* dat de directory's zijn gemaakt met:
MD D:\PROJECT
MD D:\PROJECT\PROGRAMS
MD D:\PROJECT\GEGEVENS
MD D:\PROJECT\REKO
CD D:\PROJECT\PROGRAMS
MD C:\EDITOR
Oudedir=SET("DIRECTORY")
? SET("DIRECTORY")      && D:\PROJECT\PROGRAMS
CD ..
? SET("DIRECTORY")      && D:\PROJECT
CD GEGEVENS
? SET("DIRECTORY")      && D:\PROJECT\GEGEVENS
CD ..\REKO
? SET("DIRECTORY")      && D:\PROJECT\REKO
CD C:\EDITOR
? SET("DIRECTORY")      && C: wijzigt naar C:\EDITOR
CD &Oudedir
? SET("DIRECTORY")      && D:\PROJECT\PROGRAMS
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS, maar SET DIRECTORY wordt wel ondersteund in dBASE IV.

Zie ook

MKDIR, SET DEFAULT, SET DIRECTORY, SET PATH, VALIDDRIVE( )

CDOW( )

Datum- en tijdgegevens

Geeft als resultaat de dag van de week die hoort bij een datumuitdrukking die is opgegeven in de vorm van een tekenreeks.

Syntaxis

CDOW(<*Duitdr*>)

<Duitdr>

De datumuitdrukking waarvoor de dag van de week als resultaat moet worden gegeven.

Beschrijving

CDOW( ) geeft een tekenreeks als resultaat die de dag van de week bevat waarop de opgegeven datum valt. Met DOW( ) kunt u de dag van de week in de vorm van een getal van 1 tot en met 7 als resultaat geven.

Geef *<Duitdr>* op in de huidige datumopmaak die wordt bepaald door SET DATE, DBASEWIN.INI of de optie Internationaal in het Configuratiescherm van Windows (in die volgorde). Dat wil zeggen, de instellingen bij SET DATE hebben een hogere prioriteit dan die in DBASEWIN.INI, en de instellingen in DBASEWIN.INI hebben weer een hogere prioriteit dan de instellingen in het Configuratiescherm van Windows. Let er op dat *<Duitdr>* overeenkomt met de datumopmaak die wordt gebruikt op het moment dat het programma wordt uitgevoerd.

Als u een ongeldige datum doorgeeft aan CDOW( ), wordt die omgezet in een geldige datum en wordt de naam van de weekdag als resultaat gegeven. Als u een lege uitdrukking of een andere dan een datumuitdrukking tussen accolades ({ }) doorgeeft aan CDOW( ), wordt "Onbekend" als resultaat gegeven. Als u een andere dan een datumuitdrukking of een uitdrukking die niet tussen accolades staat, doorgeeft aan CDOW( ), wordt een fout als resultaat gegeven.

Voorbeeld

In het volgende voorbeeld wordt CDOW( ) gebruikt om de weekdag voor een geheugenvariabele van het type datum als resultaat te geven.

```
CLEAR
SET TALK OFF
OudeEeuw={31-12-1999}
? CENTER("Welkom in de nieuwe eeuw op")
? CENTER(CDOW(OudeEeuw)+ "nacht,")
? CENTER(LTRIM(STR(DAY(OudeEeuw)) + " " +;
  CMONTH(OudeEeuw) + " " +;
  LTRIM(STR(YEAR(OudeEeuw)))))
```

Deze code geeft het volgende weer:

```
*          Welkom in de nieuwe eeuw op
*                 Vrijdagnacht,
*               31 december 1999
```

CDOW( ) is goed bruikbaar voor filters en voor het manipuleren van gegevens per weekdag. Als u bijvoorbeeld wilt bepalen hoeveel transacties in het weekeinde hebben plaatsgevonden, kunt u het volgende voorbeeld gebruiken.

```
USE Vluchten
SET EXACT OFF
COUNT FOR CDOW(Datum) = "Z" TO Weekeinde
? Weekeinde              && Geeft het aantal weekeindenvluchten;
                            als resultaat
```

Zo produceert u een lijst van alle vluchten op woensdag in de tabel Vluchten:

```
SET FILTER TO CDOW(datum)="Woensdag"
LIST OFF                 && "OFF" onderdrukt de recordnummers
```

Zie ook

CMONTH( ), DATE( ), DAY( ), DOW( ), SET CENTURY, SET DATE, YEAR( )

CEILING( ) — Numerieke gegevens

Geeft als resultaat het dichtstbijzijnde gehele getal dat groter dan of gelijk is aan een opgegeven getal.

Syntaxis

CEILING(<*Nuitdr*>)

<Nuitdr>

Een numeriek of zwevend getal, waarvoor u het gehele getal wilt bepalen dat groter dan of gelijk is aan het oorspronkelijke getal.

Beschrijving

CEILING( ) geeft het dichtstbijzijnde gehele getal als resultaat dat groter is dan of gelijk aan <*Nuitdr*>. Als u getal met een of meer cijfers achter de komma doorgeeft aan CEILING( ), krijgt u het dichtstbijzijnde gehele getal als resultaat dat groter is dan het oorspronkelijke getal. Als u een geheel getal of een getal met uitsluitend nullen achter de komma doorgeeft aan CEILING( ) bestaat het resultaat uit een geheel getal dat gelijk is aan het oorspronkelijke getal.

Enkele voorbeelden waarbij het standaardaantal decimalen gelijk is aan 2:

- CEILING(2.10) geeft 3,00 als resultaat
- CEILING(-2.10) geeft -2,00 als resultaat
- CEILING(2.00) geeft 2,00 als resultaat
- CEILING(2) geeft 2 als resultaat
- CEILING(-2,00) geeft -2,00 als resultaat

Bij de beschrijving van INT( ) vindt u een tabel waarin INT( ), FLOOR( ), CEILING( ) en ROUND( ) met elkaar worden vergeleken.

De waarde die CEILING( ) als resultaat geeft, heeft hetzelfde gegevenstype als <*Nuitdr*>.

Voorbeeld

In het volgende voorbeeld wordt CEILING( ) gebruikt om een doorgegeven waarde af te ronden op het volgende gehele getal:

```
SET TALK OFF
Verkoop = 79.95
Inkoop = 45.50
percentage = inkoop/verkoop * 100
```

```
? "De inkoop is " + STR(CEILING(percentage),2,0) + ;
  "% van de verkoopprijs."
SET TALK ON
```

Zie INT( ) voor een ander voorbeeld met CEILING( ).

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. In dBASE IV geeft CEILING( ) geen decimalen weer, onafhankelijk van de instelling van SET DECIMALS.

Zie ook

FLOOR( ), INT( ), ROUND( )

CENTER( )

Tekenreeksgegevens

Geeft een tekenreeks als resultaat die een reeks bevat die is gecentreerd binnen een regel met een opgegeven lengte.

Syntaxis

CENTER(<*Tuitdr*> | <*memoveld*> [, <*lengte Nuitdr*> [, <*opvulteken Tuitdr*>]])

<Tuitdr> | <memoveld>

De tekst die moet worden gecentreerd.

<lengte Nuitdr>

De lengte van de tekstregel die als resultaat moet worden gegeven. Deze lengte mag niet groter zijn dan 32766, de maximumlengte voor een tekenreeks. Als geen <*lengte Nuitdr*> is opgegeven, wordt een standaardlengte van 80 tekens gebruikt.

<opvulteken Tuitdr>

het enkele teken waarmee de reeks moet worden opgevuld als <*lengte Nuitdr*> groter is dan het aantal tekens in *Tuitdr*> | <*memoveld*>. Als de <*lengte Nuitdr*> gelijk is aan of kleiner is dan het aantal tekens in <*Tuitdr*> | <*memoveld*>, wordt <*opvulteken Tuitdr*> genegeerd.

Als <*opvulteken Tuitdr*> uit meer dan één teken bestaat, wordt alleen het eerste teken gebruikt. Als <*opvulteken Tuitdr*> niet is opgegeven, wordt de reeks opgevuld met spaties.

Beschrijving

CENTER( ) geeft als resultaat een tekenuitdrukking of memoveld. Het resultaat is voorzien van voldoende voorloop- en volgspaties om de opgegeven tekenreeks te centreren binnen een regel die een eveneens opgegeven aantal tekens breed is. Als u een

memoveld opgeeft, wordt de volledige tekst binnen het veld gecentreerd en niet elke regel afzonderlijk.

Als u een opvulteken opgeeft, wordt dat teken gebruikt in plaats van de voorloop- en volspaties. Als u meer dan een teken opgeeft, wordt alleen het eerste teken gebruikt.

De uiteindelijke tekenreeks is het gevolg van deze stappen:

- De lengte van <*Tuitdr*> of <*memoveld*> wordt afgetrokken van <*lengte Nuitdr*>.
- Het verschil wordt gedeeld door twee en eventueel afgerond.
- Vervolgens wordt <*Tuitdr*> of <*memoveld*> aan beide zijden opgevuld met het aldus verkregen aantal spaties (of met het eerste teken in <*opvulteken Tuitdr*>).

Als de lengte van <*Tuitdr*> of <*memoveld*> groter is dan <*lengte Nuitdr*> worden de volgende stappen uitgevoerd:

- <*lengte Nuitdr*> wordt afgetrokken van de lengte van <*Tuitdr*> of <*memoveld*>.
- Het verschil wordt gedeeld door twee en eventueel afgerond.
- <*Tuitdr*> of <*memoveld*> wordt aan beide kanten ingekort met het aldus verkregen aantal tekens.

Als het verschil tussen de lengte van <*Tuitdr*> of <*memoveld*> en <*lengte Nuitdr*> een oneven getal is, wordt links één teken minder toegevoegd als het verschil positief is of één teken minder ingekort als het verschil negatief is.

Voorbeeld

In het volgende voorbeeld wordt CENTER( ) gebruikt om een koptekst te centreren boven een lijst van de tabel Bedrijf. Door middel van geneste lussen worden in het resultatenpaneel van het commandovenster telkens 8 records plus de kopregel getoond:

```
SET TALK OFF
USE Afnemers
CLEAR
DO WHILE .NOT. EOF()
  ? CENTER("Rapport over database Afnemers",80)
  ?
  ? CENTER("Uitgevoerd op " + CDOW(DATE()) + ;
     " " + DTOC(DATE()),80,"-")
  ? "Bedrijf" AT 2,"Telefoon" AT 40,"Balansdatum" AT 60
  ? "********" AT 2,"*********" AT 40,"************" AT 60
  teller=1
  DO WHILE teller<9 .AND. .NOT. EOF()
    ? Bedrijf AT 2, Telefoon AT 40, CDOW(Balnsdatum)+",";
      + DTOC(Balnsdatum) AT 60
    SKIP
    teller=teller+1
  ENDDO
  ?
WAIT
CLEAR
ENDDO
CLOSE ALL
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

LEN( ), REPLICATE( ), SET MEMOWIDTH, TRANSFORM( )

CERROR( )

Foutafhandeling en testen op fouten

Geeft het nummer van de laatste compiler-fout als resultaat.

Syntaxis

CERROR( )

Beschrijving

Gebruik CERROR( ) voordat u een nieuw programma uitvoert om te controleren of de broncode correct is gecompileerd. Als er geen compiler-fout heeft plaatsgevonden, geeft CERROR( ) 0 als resultaat. Iedere keer dat u of dBASE een programma of indelingsbestand compileert, wordt CERROR( ) bijgewerkt. CERROR( ) wordt niet beïnvloed door waarschuwingen die tijdens de compilatie worden gegenereerd.

Gebruik CERROR( ) in een programmabestand. Als u ? CERROR( ) geeft in het commandovenster, wordt 0 als resultaat gegeven. (Op dat moment wordt namelijk de instructie "? CERROR( )" zelf gecompileerd en dat levert geen compiler-fout op.)

Bij de beschrijving van ERROR( ) staat een tabel waarin ERROR( ), MESSAGE( ), DBERROR( ), DBMESSAGE( ), SQLERROR( ), SQLMESSAGE( ) en CERROR( ) met elkaar worden vergeleken.

Zie Help voor een lijst van alle foutmeldingen.

Voorbeeld

In het volgende programmafragment wordt CERROR( ) gebruikt in een DO...WHILE-lus zodat de gebruiker het programma kan bewerken tot het probleemloos wordt gecompileerd:

```
DO WHILE .T.
  Clear
    MODIFY COMMAND GEBR.PRG
    ON ERROR ? ERROR(), MESSAGE(), CERROR()
    COMPILE GEBR.PRG
    IF CERROR()>0
      ?
      WAIT "Het programma is niet gecompileerd. ;
      Druk op een toets om uw .PRG te wijzigen."
      LOOP
    ELSE
      EXIT
```

```
    ENDIF
ENDDO
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

COMPILE, DBERROR( ), DBMESSAGE( ), ERROR( ), MESSAGE( ), ON ERROR

CHANGE

Velden en records

Met dit commando kunt u de gegevens in de huidige tabel per record weergeven en bewerken.

Syntaxis

CHANGE
[<*eerste record Nuitdr1*> | <*bladwijzer*>]
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[COLOR [<*gewone tekst*>]
[, [<*speciale tekst*>]
[, [<*buitenrandkleur*>]
[, <*achtergrondkleur*>]]]]
[COLUMNAR]
[COMPRESS]
[FIELDS <*veld1*> [<*veldoptielijst1*>] |
<*rekenveld1*> = <*uitdr1*> [<*rekenveldoptielijst1*>]
[, <*veld2*> [<*veldoptielijst2*>] |
<*rekenveld2*> = <*uitdr2*> [<*rekenveldoptielijst2*>] ...]]
[FORMAT]
[FREEZE <*veld3*>]
[KEY <*uitdr3*> [, <*uitdr4*>]]
[LOCK <*Nuitdr2*>]
[NOAPPEND]
[NODELETE]
[NOEDIT | NOMODIFY]
[NOFOLLOW]
[NOINIT]
[NORMAL]
[NOTOGGLE]
[NOWAIT]
[TITLE <*Tuitdr1*>]
[WIDTH <*Nuitdr3*>]
[WINDOW <*vensternaam*>]

Beschrijving

CHANGE en EDIT zijn onderling uitwisselbaar. Zie EDIT voor een volledige beschrijving van deze commando's. CHANGE zonder de optie FIELDS geeft alle velden in het huidige record weer.

Voorbeeld

Zie EDIT voor een voorbeeld. Vervang in dit voorbeeld EDIT overal door CHANGE.

Zie ook

BROWSE, EDIT

CHANGE( ) Gedeelde gegevens

Geeft .T. als resultaat als een andere gebruiker een record heeft gewijzigd sinds het uit het tabelbestand is ingelezen.

Syntaxis

CHANGE([<*alias*>])

<alias>

Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan. Als u geen <*alias*> opgeeft, geeft CHANGE( ) informatie over de huidige tabel als resultaat.

Beschrijving

Gebruik CHANGE( ) om te bepalen of een andere gebruiker wijzigingen heeft aangebracht in een record sinds het uit het tabelbestand is ingelezen. Als het record is gewijzigd, kunt u een bericht weergeven voordat u de gebruiker toestaat verder te gaan.

CHANGE( ) kan alleen informatie als resultaat geven als de huidige of opgegeven tabel is voorzien van een _dbaselock-veld. Gebruik CONVERT om een _dbaselock-veld toe te voegen aan de tabel. Als de tabel geen _dbaselock-veld bevat, geeft CHANGE( ) .F. als resultaat.

CHANGE( ) vergelijkt de teller in de geheugenvoorstelling van _dbaselock in het werkstation met de teller die op schijf staat. Als die niet gelijk zijn, is het record gewijzigd en geeft CHANGE( ) .T. als resultaat

U kunt de waarde CHANGE( ) veranderen in .F. door de recordaanwijzer te verplaatsen. GOTO RECNO( ) leest op nieuw het _dbaselock-veld van het huidige record en een volgende CHANGE( ) geeft .F. als resultaat, tenzij een andere gebruiker het record heeft gewijzigd tussen het moment van de verplaatsing en het geven van het commando CHANGE( ) .

Opmerking CHANGE( ) test geen SQL-databases of Paradox-tabellen.

Voorbeeld

In het volgende voorbeeld wordt de tabel Bedrijf geopend en wordt naar een bepaald record gegaan. Vervolgens wordt een formulier geopend waarin de gegevens voor dat jaar tot nu toe kunnen worden bekeken. Binnen het formulier OverzichtTotNu kan de gebruiker die waarde niet wijzigen. (De definitie van het formulier OverzichtTotNu is niet afgebeeld.) Nadat het formulier is gesloten, gebruikt het programma CHANGE( ) om te bepalen of een andere gebruiker het huidige record heeft gewijzigd. Als dat niet het geval is, wordt met REPLACE de waarde van het veld VerkTotNu bijgewerkt:

```
USE BEDRIJF
SEEK mBedrijf
OPEN FORM OverzichtTotNu
IF .NOT. CHANGE()
* Is het record gewijzigd door een andere gebruiker?
  REPLACE  VerkTotNu WITH mJVerk
ELSE
  GOTO RECNO()  && Lees dit record opnieuw
ENDIF
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CONVERT, FLOCK( ), LKSYS( ), RLOCK( ), SET EXCLUSIVE, SET REFRESH

CHARSET( ) — Omgeving

Geeft als resultaat de naam van de tekenset die wordt gebruikt in de huidige of opgegeven tabel. Als geen tabel is geopend en CHARSET( ) wordt gebruikt zonder argument, wordt de naam van de algemene tekenset als resultaat gegeven.

Syntaxis

CHARSET([*<alias>*])

<alias>
Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

Gebruik CHARSET( ) om te bepalen welke tekenset wordt gebruikt in de huidige tabel of een opgegeven tabel. Als u geen argument doorgeeft aan CHARSET( ), wordt de tekenset van de huidige tabel als resultaat gegeven. Als er geen tabel is geopend en u geeft geen argument op, wordt de naam van de algemene tekenset als resultaat gegeven. CHARSET( ) geeft ook informatie als resultaat over Paradox- en SQL-databases.

De tekenset waarin de gegevens van een tabel zijn opgeslagen, is afhankelijk van de taalaansturing die actief was op het moment dat de tabel werd gemaakt. Bij dBASE voor Windows geeft u de taalaansturing voor dBASE-gegevens op in de sectie [CommandSettings] in het bestand DBASEWIN.INI. Zie Appendix C in *Programmeren* voor meer informatie over tekensets en taalaansturingen.

De waarde die door CHRSET( ) als resultaat wordt gegeven, is een onderdeel van de waarde die LDRIVER( ) als resultaat geeft. Zie LDRIVER( ) voor meer informatie.

Voorbeeld

In dit voorbeeld ziet u een toepassing van de functie CHARSET( ) en een voorbeeld van een resultaat:

```
? CHARSET()  && Geeft DOS:437 als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ANSI( ), LDRIVER( ), OEM( ), SET LDCHECK

CHOOSEPRINTER( )

Afdrukken

Opent het dialoogvenster **Printerinstellingen** waarin u een printer kunt kiezen en afdrukopties kunt instellen. Tevens worden de toepasselijke systeemgeheugenvariabelen voor de printer of opgegeven afdrukopties hersteld. Als u het dialoogvenster verlaat door **OK** te kiezen, wordt .T. als resultaat gegeven. Kiest u **Annuleren**, dan wordt .F. als resultaat gegeven.

Syntaxis

CHOOSEPRINTER(<*titel Tuitdr*>)

<titel Tuitdr>

Bepaalt de titel voor het dialoogvenster Printerinstellingen.

Beschrijving

Gebruik CHOOSEPRINTER( ) om een nieuwe printer te kiezen of afdrukopties in te stellen in het dialoogvenster **Printerinstellingen**. U kunt het dialoogvenster **Printerinstellingen** ook openen door **Bestand | Printerinstellingen** te kiezen. In het dialoogvenster **Printerinstellingen** kunt u een andere geïnstalleerde printer kiezen of opties instellen voor papierformaat, papierbron en afdrukrichting (**Staand** of **Liggend**).

CHOOSEPRINTER( ) is van invloed op de waarde van de instelling voor SET PRINTER TO en de geheugenvariabelen _pdriver, _plength en _porientation.

Alleen printers die u eerder hebt geïnstalleerd met Afdrukbeheer van Windows, verschijnt in de lijst bij Specifieke printer. U kunt de printerstuurprogramma's in Windows ook wijzigen met de optie Printers in het Configuratiescherm van Windows.

Tenslotte kunt u ook _pdriver gebruiken om een bepaald printerstuurprogramma te activeren.

Voorbeeld

In dit voorbeeld wordt met CHOOSEPRINTER( ) het dialoogvenster **Printerinstellingen** geopend. NwPrinter geeft waar als resultaat als de printer opnieuw is ingesteld:

```
NwPrinter=CHOOSEPRINTER()
NwPrinter=CHOOSEPRINTER("Kies een matrixprinter")
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

_pdriver, CLOSE..., SET DEVICE, SET PRINTER

CHR( )

Uitdrukkingen en gegevenstypeconversie

Geeft het teken voor een opgegeven ASCII-waarde als resultaat.

Syntaxis

CHR(<*Nuitdr*>)

<Nuitdr>

De numerieke ASCII-waarde, van 0 tot en met 255, waarvoor het teken als resultaat moet worden gegeven.

Beschrijving

CHR( ) is de tegengestelde functie van ASC( ). CHR( ) accepteert een ASCII-waarden en geeft het bijbehorende teken als resultaat, terwijl ASC( ) een teken accepteert en de ASCII-waarde voor dat teken als resultaat geeft.

U kunt CHR( ) en ASC( ) gebruiken om ASCII-waarden om te zetten naar ASCII-tekens en omgekeerd. Raadpleeg de ASCII-tabel in Appendix E voor een volledige lijst van de ASCII-tekens en hun bijbehorende waarden.

CHR(7) geeft een geluidssignaal als resultaat. Met SET BELL TO [<*frequentie Nuitdr*>, <*duur Nuitdr*>] kunt u de frequentie en de duur van het geluidssignaal instellen.

Voorbeeld

In het volgende voorbeeld wordt CHR( ) gebruikt om een speciaal teken op het scherm weer te geven. Speciale tekens zijn tekens die niet beschikbaar zijn op het toetsenbord, maar die wel kunnen worden afgedrukt of weergegeven met de functie CHR( ) (in het voorbeeld wordt de standaard OEM-tekenset gebruikt).

```
? "Het totaal is 50" + CHR(241) + "2"
* De ASCII-waarde 241 geeft het symbool voor plusminus weer
```

Met de commando's ?? en CHR( ) kunnen als volgt printerbesturingscodes naar de printer worden gestuurd:

```
SET PRINT ON
?? CHR(27) + "E"
* Herstelt de standaardinstellingen op een HP LaserJet door
* ESCAPE (CHR(27)) plus een besturingscode naar de printer te sturen.
* Deze instructie is identiek: ?? CHR(27) + CHR(69).
```

Zie ook

ANSI( ), ASC( ), OEM( ), SET BELL

CLASS...ENDCLASS

Objecten

Definieert een eigen klasse en geeft de MEMBER-variabelen en -functies voor die klasse op.

Syntaxis

CLASS <*klassenaam*> [(<*parameters*>)] [CUSTOM]
[PARAMETERS <*parameters*>]
[OF <*superklassenaam*> [(<*parameters*>)]]
[<*constructor-code*>]
[<*verwante functies*>]
ENDCLASS

CUSTOM

Geeft aan dat het nieuwe object een eigen stuurelement is. Zie Hoofdstuk 15 van *Programmeren* voor meer informatie over eigen stuurelementen.

<klassenaam>

De naam voor de nieuwe klasse.

OF *<superklassenaam>*

Geeft aan dat de nieuwe klasse de kenmerken en methoden van <*superklassenaam*> overneemt. U kunt de nieuwe klasse bijvoorbeeld alle kenmerken en methoden laten overnemen van de keuzelijstklasse of van een andere klasse die u hebt gemaakt met CLASS...ENDCLASS.

<constructor-code>
Code die wordt uitgevoerd als u een object van de klasse *<klassenaam>* maakt. Tot de constructor-code behoren alle instructies tussen de sleutelwoorden CLASS en ENDCLASS behalve de code in *<verwante functies>*.

<verwante functies>
Procedures en functies die u definieert tussen de sleutelwoorden CLASS en ENDCLASS. Deze subroutines vormen de methoden van de nieuwe klasse.

Beschrijving

Gebruik CLASS...ENDCLASS om een nieuwe klasse te maken.

Een klasse is een specificatie, of sjabloon, voor een soort object. dBASE voor Windows voorziet in een groot aantal standaardklassen, zoals Formulier en Invoervak. Als u bijvoorbeeld een formulier maakt, maakt u een nieuw formulierobject met de standaardkenmerken en -methoden van de klasse Formulier. Als u echter een eigen klasse definieert met CLASS...ENDCLASS, bepaalt *u* welke kenmerken en methoden objecten die van de nieuwe klasse zijn afgeleid, zullen hebben.

Kenmerken voor de nieuwe klasse maakt u met *<constructor-code>*. De constructor-code wordt uitgevoerd als u een object van die klasse maakt. Hoewel de constructor-code elk willekeurig dBASE-commando mag bevatten, bestaat deze in de meeste gevallen uitsluitend uit instructies voor het toekennen van kenmerken en methoden.

Als u in een klassedefinitie een nieuw kenmerk maakt, moet u de naam van het kenmerk vooraf laten gaan door het sleutelwoord *This*. *This* verwijst naar het object dat u maakt. Het volgende codefragment bevat bijvoorbeeld een klassedefinitie. In de definitie wordt This gebruikt om aan te geven dat Labelnaam, een nieuw kenmerk, een MEMBER is van de nieuwe klasse Tabelbestand.

```
xBestand = NEW Tabelbestand()
? xBestand.Labelnaam
? xBestand.BestNaamId()
CLASS Tabelbestand
   This.Labelnaam = "XORDER"
   FUNCTION BestNaamId && Eigen methode.
   RETURN DBF()
ENDCLASS
```

Eigen methoden voor de klasse maakt u met *<verwante functies>*. Deze code kan bestaan uit definities van procedures of van eigen functies. FUNCTION BestNaamId is een voorbeeld van een eigen methode.

Voorbeeld

In het volgende voorbeeld wordt CLASS...ENDCLASS gebruikt om een objectklasse te definiëren binnen een formulier dat afbeeldingen weergeeft uit de tabel AFBEELD in de directory VOORBD. Dit voorbeeld is een fragment uit AFBEELD.WFM in de directory VOORBD:

```
LOCAL f
f = NEW AFBEELDFORM()
f.Open()
```

```
CLASS AFBEELDFORM OF FORM
   Set Procedure To KNOPPEN.CC additive
   this.Top =          0
   this.Left =          0
   this.Height =         19.0586
   this.Minimize = .F.
   this.Maximize = .F.
   this.HelpFile = ""
   this.Text = "Afbeeldingenformulier"
   this.HelpId = ""
   this.MousePointer =          1
   this.View = "AFBEELD.QBE"
   this.ColorNormal = "BG/B"
   this.Width =         91.5

   DEFINE PUSHBUTTON GELUID OF THIS;
       PROPERTY;
         Top         11.2793,;
         Left          1.3184,;
         Height          1.4854,;
         OnClick {;play sound binary afbeeld->geluid},;
         Group .T.,;
         Text "&Geluid",;
         Default .T.,;
         ColorNormal "N/W",;
         Width         19.8477

   DEFINE LISTBOX DINGEN OF THIS;
       PROPERTY;
         Top          4.5479,;
         Left          1.1592,;
         Height          5.5107,;
         ID        800,;
         DataSource "FIELD NAAM",;
         ColorHighLight "RG+/B",;
         ColorNormal "N/W+",;
         Width         20.1738

   DEFINE OLE AFBEELDING OF THIS;
       PROPERTY;
         Top          3.2979,;
         Left         23.0977,;
         Border .T.,;
         Height         15.5254,;
         ID         88,;
         DataLink "AFBEELD->BITMAPOLE",;
         Width         67.0684

   DEFINE TEXT TITEL OF THIS;
       PROPERTY;
         FontSize         28,;
         Top          0,;
         FontName "Times New Roman",;
         Left         -9,;
```

```
        Border .F.,;
        Height          3,;
        Alignment          4,;
        Text "OOG " + '&' + '&' + " OOR",;
        ColorNormal "GR+/B",;
        FontBold .F.,;
        Width        100

    DEFINE TEXT KOPTEKST OF THIS;
        PROPERTY;
        Top          3.2979,;
        Left          1.3184,;
        Border .F.,;
        Height          0.9365,;
        Text "Afbeelding:",;
        ColorNormal "RG+/B",;
        Width         17.8477

ENDCLASS
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DEFINE, REDEFINE

CLEAR — Omgeving

Wist de inhoud van het resultatenpaneel van het commandovenster of het huidige dBASE IV-venster en tevens alle nog niet behandelde instructies met @...GET.

Syntaxis

CLEAR
  [CHARACTER <*Tuitdr*>]

CHARACTER <*Tuitdr*>

Vult het resultatenpaneel van het commandovenster of het huidige dBASE IV-venster met het eerste teken van de uitdrukking <*Tuitdr*>. Als u geen teken opgeeft, wordt het venster gevuld met het laatste teken dat voor <*Tuitdr*> is opgegeven. Is er nog geen teken opgegeven voor <*Tuitdr*>, dan worden spaties gebruikt.

Beschrijving

Gebruik CLEAR om de inhoud van het resultatenpaneel van het commandovenster of het huidige dBASE IV-venster te wissen. Gebruik bij CLEAR de optie CHARACTER <*Tuitdr*> om een teken op te geven waarmee het venster moet worden gevuld. Gebruik @...CLEAR om een *deel* van het venster te wissen.

Als u CLEAR gebruikt zonder optie nadat u eerder al een keer CLEAR CHARACTER <*Tuitdr*> hebt gebruikt, wordt <*Tuitdr*> opnieuw gebruikt. Als u het ingestelde teken niet langer wilt gebruiken, geeft u de instructie CLEAR CHARACTER " " (met een spatie of een lege tekenreeks voor <*Tuitdr*>).

Als u geen nog openstaande instructies met GET wilt sluiten en verwijderen, moet u READ SAVE gebruiken voordat u CLEAR geeft.

CLEAR sluit het venster niet. Verder is CLEAR in dBASE voor Windows ook niet van invloed op formulieren, die u maakt met DEFINE FORM.

Overdraagbaarheid

De optie CHARACTER <*Tuitdr*> wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

@...CLEAR, @...SAY...GET, DEFINE FORM, READ

CLEAR ALL

Omgeving

Wist alle geheugenvariabelen (behalve systeemgeheugenvariabelen), sluit een catalogus als er een open is, sluit alle geopende tabellen en stelt werkgebied 1 in.

Syntaxis

CLEAR ALL

Beschrijving

Gebruik CLEAR ALL om alle geopende tabellen en databases in de huidige sessie te sluiten en wist het werkgebied zonder van invloed te zijn op geopende PROCEDURE- of LIBRARY-bestanden. CLEAR ALL voert de volgende stappen uit:

- Alle geheugenvariabelen (behalve systeemgeheugenvariabelen) in alle actieve sessies worden gewist.
- Alle geopende dBASE IV-vensters worden gesloten.
- Alle definities van dBASE IV-vensters worden uit het geheugen verwijderd, inclusief vensterelementen, menubalken, afrolmenu's en popup-menu's.
- Een geopend catalogusbestand (.CAT-bestand) wordt gesloten.
- Alle geopende tabellen databases in de huidige sessie en hun bijbehorende index-, memo- en indelingsbestanden worden gesloten.
- Werkgebied 1 wordt geactiveerd.

Opmerking CLEAR ALL wist gewoonlijk geen objecten. Als de enige verwijzing naar een object echter uit een geheugenvariabele bestaat, wordt dat object wel vrijgemaakt door CLEAR ALL.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CLEAR PROGRAM, CLOSE..., RELEASE, SET PROCEDURE, SET LIBRARY

CLEAR AUTOMEM

Velden en records

Initialiseert automatische geheugenvariabelen voor de huidige tabel met lege waarden.

Syntaxis

CLEAR AUTOMEM

Beschrijving

Gebruik CLEAR AUTOMEM om voor de huidige tabel een verzameling automatische geheugenvariabelen te initialiseren met lege waarden. Alle nog niet bestaande automatische geheugenvariabelen worden door CLEAR AUTOMEM gemaakt. Als de variabelen wel bestaan, worden ze door CLEAR AUTOMEM opnieuw geïnitialiseerd. Als er geen tabel wordt gebruikt, maakt CLEAR AUTOMEM geen variabelen.

Automatische geheugenvariabelen hebben dezelfde naam en hetzelfde gegevenstype als de velden in de actieve tabel. U kunt automatisch lege automatische geheugenvariabelen maken voor de huidige tabel met CLEAR AUTOMEM of USE...AUTOMEM, of de variabelen met de hand maken met STORE.

Gebruik CLEAR AUTOMEM als automatische geheugenvariabelen in een programma meer dan eens worden toegepast, zodat ze opnieuw worden geïnitialiseerd met lege waarden en niet de waarden handhaven uit een vorige weergave of uit een vorig record. U kunt CLEAR AUTOMEM bijvoorbeeld gebruiken binnen een lus om gegevens toe te voegen met een formulier, automatische geheugenvariabelen en APPEND AUTOMEM, zodat de ingevoerde waarden tijdens de ene doorgang door de lus tijdens de volgende doorgang niet opnieuw in het invoerformulier verschijnen.

U kunt ook met CLEAR AUTOMEM automatische geheugenvariabelen maken als u dat nog niet eerder hebt gedaan met USE...AUTOMEM. Op die manier kunt u bijvoorbeeld vanuit het commandovenster automatische geheugenvariabelen maken voor een tabel die al in gebruik is.

Voorbeeld

In het volgende voorbeeld wordt CLEAR AUTOMEM gebruikt om de gebruiker in staat te stellen automatische geheugenvariabelen te wijzigen voor een nieuw record. Er wordt alleen een nieuw record toegevoegd als de gebruiker bevestigt dat de gegevens in de automatische geheugenvariabelen juist zijn:

```
SET TALK OFF
CLEAR
USE Bedrijf
GegsToevoegen()
RETURN

FUNCTION GegsToevoegen
@0,0 to 8, 70
@10,20 to 12,45
CLEAR AUTOMEM
DO WHILE .T.
  JaWel = .f.
  @11,22 CLEAR TO 11,40
  @1,1 SAY 'ID' GET m->COMPCODE
  @1,10 SAY 'Bedrijf' GET m->BEDRIJF
  @3,1 SAY 'Adres' GET m->ADRES
  @5,1 SAY 'Postcode' GET POSTCODE
  @5,24 SAY 'Plaats' GET m->PLAATS
  READ
    IF READKEY()  = 12
      EXIT
    ELSE
      @ 11,22 SAY "Gegevens juist? J/N";
      GET JaWel picture 'J'
      READ
        IF JaWel
          APPEND AUTOMEM
          CLEAR AUTOMEM
        ENDIF
      ENDIF
 ENDDO
 RETURN .T.
```

Zie ook

APPEND, STORE, USE

CLEAR FIELDS

Velden en records

Verwijdert de veldenlijst die is gedefinieerd met het commando SET FIELDS TO.

Syntaxis

CLEAR FIELDS

Beschrijving

Gebruik CLEAR FIELDS in alle werkgebieden de instelling voor SET FIELDS TO *<veldenlijst>* te verwijderen en automatisch SET FIELDS uit te schakelen (OFF), zodat alle velden in alle geopende tabellen bereikbaar worden. U kunt CLEAR FIELDS gebruiken voordat u een nieuwe veldenlijst opgeeft met SET FIELDS TO. Verder kunt u CLEAR FIELDS ook gebruiken aan het eind van een programma. CLEAR FIELDS heeft dezelfde resultaten als SET FIELDS TO zonder opties.

Voorbeeld

Zie SET FIELDS voor voorbeelden met CLEAR FIELDS.

Zie ook

SET FIELDS

CLEAR GETS

Invoer/uitvoer

Wist alle huidige @...GET-velden. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE met de klassen Text en EntryField voor het weergeven en accepteren van informatie op een formulier.

Zie Help voor meer informatie over de syntaxis van CLEAR GETS. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het werken met formulieren in dBASE voor Windows.

CLEAR MEMORY

Geheugenvariabelen

Wist alle eigen geheugenvariabelen.

Syntaxis

CLEAR MEMORY

Beschrijving

Met CLEAR MEMORY maakt u alle geheugenvariabelen (behalve systeemgeheugenvariabelen) in alle actieve sessies vrij, inclusief variabelen die zijn gedefinieerd met PUBLIC en STATIC of die in subroutines van een hoger niveau zijn geïnitialiseerd. CLEAR MEMORY is niet van invloed op systeemgeheugenvariabelen.

Opmerking CLEAR MEMORY maakt gewoonlijk geen objecten vrij. Als de enige verwijzing naar een object echter uit een geheugenvariabele bestaat, maakt CLEAR MEMORY dat object wel vrij.

De resultaten van CLEAR MEMORY, gegeven in een programma of in het commandovenster, zijn gelijk aan die van RELEASE ALL in het commandovenster. RELEASE ALL in een programma heeft echter andere resultaten dan CLEAR MEMORY. RELEASE ALL in een programma wist alleen geheugenvariabelen die zijn gemaakt op hetzelfde niveau als de RELEASE ALL-instructie. In tegenstelling tot CLEAR MEMORY heeft RELEASE ALL in een programma geen gevolgen voor variabelen die met PUBLIC of STATIC zijn gedefinieerd of die op een hoger niveau zijn gedefinieerd.

CLEAR MEMORY werkt ook anders dan RELEASE AUTOMEM. Gegeven in een programma of in het commandovenster wist RELEASE AUTOMEM alleen automatische geheugenvariabelen die bij de huidige tabel horen. (Zie USE, CLEAR AUTOMEM en RELEASE AUTOMEM voor meer informatie over automatische geheugenvariabelen.)

Met RELEASE kunt u een of meer opgegeven geheugenvariabelen wissen.

Voorbeeld

In het volgende voorbeeld wordt CLEAR MEMORY gebruikt om variabelen in een array te wissen nadat ze ten behoeve van een reservekopie zijn opgeslagen in een .DBF-bestand:

```
SET SAFETY OFF
USE Klanten EXCLUSIVE
INDEX ON KlantNr TAG KlantNr
DECLARE KopieInfo[RECCOUNT() ,FLDCOUNT() ]
COPY TO ARRAY KopieInfo
*
* Hier bestandsbewerkingen uitvoeren...
*
COPY STRUCTURE TO KlantBak.DBF WITH PRODUCTION
USE KlantBak
APPEND FROM ARRAY KopieInfo
CLEAR MEMORY      && Waarden in array;
                     KopieInfo vrijmaken
* Volgende bewerkingen waarvoor
* veel geheugen nodig is...
CLOSE ALL
```

Zie ook

CLEAR AUTOMEM, PRIVATE, PUBLIC, RELEASE, RELEASE AUTOMEM, STATIC, USE

CLEAR MENUS

dBASE IV-menu's

Verwijdert alle dBASE IV-balkmenu's van het scherm en wist de definities uit het geheugen. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows RELEASE OBJECT om een object te wissen in een formulier.

Zie Help voor meer informatie over de syntaxis van CLEAR MENUS. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

CLEAR POPUPS

dBASE IV-menu's

Verwijdert alle dBASE IV popup-menu's van het scherm en wist hun definities in het geheugen. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows RELEASE OBJECT om een object te verwijderen van een formulier.

Zie Help voor meer informatie over de syntaxis van CLEAR POPUPS. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

CLEAR PROGRAM

Programma's

Wist uit het geheugen alle gecompileerde programmabestanden die niet worden uitgevoerd of zijn geopend met SET FORMAT, SET PROCEDURE of SET LIBRARY.

Syntaxis

CLEAR PROGRAM

Beschrijving

Als u een programma uitvoert met DO, wordt het gecompileerde programma in het geheugen geladen, plus alle gecompileerde programma's, procedures, door gebruiker gedefinieerde functies en indelingsbestanden die door het programma worden aangeroepen met DO, SET FORMAT, SET PROCEDURE en SET LIBRARY. Deze bestanden blijven in het geheugen tot u CLEAR PROGRAM geeft of tot dBASE meer geheugen nodig heeft, als u bijvoorbeeld een grote tabel opent. Het interne dynamische geheugenbeheer van dBASE kan ook meer geheugen beschikbaar maken door alle

programma's uit het geheugen te verwijderen die al een tijd niet meer actief zijn geweest.

CLEAR PROGRAM verwijdert alle inactieve gecompileerde programma's uit het geheugen. programma's die worden uitgevoerd en bestanden die door het huidige programma zijn geopend met SET FORMAT, SET PROCEDURE en SET LIBRARY worden niet uit het geheugen verwijderd. Als u echter een aangeroepen bestand sluit (met CLOSE PROCEDURE of CLOSE FORMAT bijvoorbeeld), zal een volgende instructie met CLEAR PROGRAM het gesloten bestand uit het geheugen verwijderen.

Met CLEAR PROGRAM kunt u geheugen vrijmaken voordat u aan een taak begint die veel geheugen nodig heeft. Gebruik CLEAR PROGRAM bijvoorbeeld voordat u een DOS-opdracht start die veel geheugen nodig heeft of voordat u een groot aantal tabellen opent.

Gebruik CLEAR PROGRAM niet te vaak, want dBASE houdt programma's in het geheugen om de uitvoering te versnellen. Als aangeroepen bestanden telkens opnieuw in het geheugen geladen moeten worden, wordt de verwerkingssnelheid lager.

Voorbeeld

In het volgende voorbeeld wordt CLEAR PROGRAM gebruikt om geheugen vrij te maken dat in beslag wordt genomen door ongebruikte programma's, procedures en indelingsbestanden:

```
** Hoofd.PRG **
CLEAR
DO Inst_Hfd                        && Extern PRG
DO Pak_Film                        && Interne Proc
CLEAR PROGRAM                      && Geheugen vrijmaken;
                                      door programma's uit;
                                      geheugen te verwijderen

DO Film_Rpt

PROCEDURE Pak_Film
USE Films
INDEX ON Regiseur TAG Film_Reg
GOTO rec_no
DO WHILE Director = Film_Reg
   ? Titel
   SKIP
ENDDO WHILE Director = Film_Reg
GOTO rec_no
CLOSE DATABASES
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DO, CLEAR ALL, CLOSE..., SET FORMAT, SET LIBRARY, SET PROCEDURE

CLEAR SCREENS — Invoer/uitvoer

Verwijdert uit het geheugen alle variabelen die zijn gemaakt met SAVE SCREEN en wist de buffer voor het commandovenster. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV.

Zie Help voor meer informatie over de syntaxis van CLEAR SCREENS.

CLEAR TYPEAHEAD — Toetsenbord- en muisacties

Wist de typbuffer. In de typbuffer worden toetsaanslagen opgeslagen die nog niet kunnen worden uitgevoerd omdat het programma nog bezig is met andere taken. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Formulieren in dBASE voor Windows maken geen gebruik van een typbuffer.

Zie Help voor meer informatie over de syntaxis van CLEAR TYPEAHEAD. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het werken met formulieren.

CLEAR WINDOWS — dBASE IV-vensters

Wist alle dBASE IV-vensters van het scherm en verwijdert hun definities uit het geheugen. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows CLOSE FORMS of RELEASE OBJECT om een formulier te sluiten of vrij te maken.

Zie Help voor meer informatie over de syntaxis van CLEAR WINDOWS. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

CLOSE...

Sluit verschillende soorten bestanden.

Syntaxis

CLOSE ALL	**Tabellen**
CLOSE ALTERNATE	**Invoer/uitvoer**
CLOSE DATABASES [<*databasenamenlijst*>]	**Tabellen**
CLOSE FORMAT	**Invoer/uitvoer**
CLOSE FORMS [<*formuliernamenlijst*>	**Formulieren**
CLOSE INDEXES	**Tabelindeling**
CLOSE PRINTER	**Afdrukken**
CLOSE PROCEDURE <*bestandsnaam*>	**Programma's**
CLOSE TABLES	**Tabellen**

CLOSE ALL

Sluit tabellen, databases en alle mogelijke soorten bestanden, inclusief low-level bestanden en maakt werkgebied 1 weer actief. Bestanden die zijn geopend met SET DEVICE TO <*bestandsnaam*> of SET PRINTER TO <*bestandsnaam*> en SQL-databases blijven echter open. Gebruik CLOSE DATABASES om SQL-databases te sluiten.

CLOSE ALTERNATE

Sluit tekstbestanden (.TXT-bestanden) die zijn geopend met het commando SET ALTERNATE.

CLOSE DATABASES [<*databasenamenlijst*>]

Sluit de databases die worden genoemd in een (door komma's gescheiden) lijst of alle geopende databases, inclusief alle tabellen, en bijbehorende indexbestanden (.MDX en .NDX), memobestanden (.DBT) en indelingsbestanden (.FMT) voor elke database.

CLOSE FORMAT

Sluit geopende indelingsbestanden (.FMT) in het huidige werkgebied.

CLOSE FORMS [<*formuliernamenlijst*> | WITH <*uitdr*>]

Sluit formulieren die worden genoemd in een lijst en voert de standaardsluitroutines voor de formulieren en de objecten in die formulieren. Het argument WITH <*uitdr*> bepaalt de waarde die wordt doorgegeven aan de functie READMODAL( ) waarmee het formulier oorspronkelijk is geopend.

CLOSE INDEXES

Sluit indexbestanden (.MDX en .NDX) die in het huidige werkgebied zijn geopend. Deze optie sluit niet het produktie-MDX-bestand.

CLOSE PRINTER
Sluit een bestand dat is geopend met het commando SET PRINTER.

CLOSE PROCEDURE *<bestandsnaam>*
Sluit een geopend procedurebestand.

CLOSE TABLES
Sluit alle tabellen in alle werkgebieden of alle tabellen in de huidige database als er een is geselecteerd.

Beschrijving
Als u het commando CREATE SESSION gebruikt, heeft het commando CLOSE alleen invloed op bestanden in de huidige *sessie*. Het begrip sessie komt overeen met een gebruiker-sessie in een multi-user-omgeving. In elke sessie wordt een eigen groep van werkgebieden bijgehouden. Een exclusief geopend bestand is niet toegankelijk voor andere sessies.

In de interactieve omgeving van dBASE voor Windows wordt standaard gewerkt met CREATE SESSION. Opent u bijvoorbeeld tabellen via de gebruikersinterface (door bijvoorbeeld te dubbelklikken op een tabelpictogram), dan wordt door dBASE elke tabel in een aparte sessie geopend. In het commandovenster ziet u, dat telkens wanneer u een tabel opent, het commando CREATE SESSION wordt uitgevoerd. Als u vervolgens in het commandovenster dde opdracht CLOSE TABLES geeft, blijven de tabellen open, omdat ze immers worden beschermd binnen hun sessies. Net als in een multi-user-omgeving kan een gebruiker niet de geopende bestanden van een andere gebruiker sluiten.

Het commandovenster zelf werkt in een onafhankelijke sessie, hetgeen nog een reden is waarom een commando CLOSE in het commandovenster geen invloed heeft op de bestanden die via de gebruikersinterface zijn geopend. Opent u echter tabellen met de opdrachten USE en BROWSE in het commandovenster, dan maken deze tabellen deel uit van de sessie van het commandovenster. Als u dan het commando CLOSE TABLES in het commandovenster geeft, worden deze tabellen gesloten.

Zie CREATE SESSION voor meer informatie over sessies.

Voorbeeld
In het volgende voorbeeld wordt CLOSE ALL gebruikt om alle geopende tabellen te sluiten nadat een rapport is voltooid:

```
SET SAFETY OFF
SET TALK OFF
USE Contact EXCLUSIVE IN SELECT()
INDEX ON COMPCODE TAG COMPCODE
USE Bedrijf IN SELECT()
SELECT Bedrijf
SET RELATION TO CompCode INTO Contact
GO TOP
DO LystBedr
CLOSE ALL
```

```
PROCEDURE LystBedr
SET ALTERNATE TO BedrLyst
SET ALTERNATE ON
CLEAR
? "Bedrijf" AT 5, "Contactpersoon" AT 45
?
DO WHILE .NOT. EOF()
  ? Bedrijf->Bedrijf AT 5, Contact->Contact AT 45
  ? "In " + REPLICATE("- ",7) + "->" AT 5,;
    TRIM(Bedrijf->Plaats) +", " +;
    TRIM(Bedrijf->Regio) AT 25
   SKIP
ENDDO
RETURN
```

Zie ook

CLEAR ALL, CREATE SESSION

CMONTH( )

Datum- en tijdgegevens

Geeft als resultaat de naam van de maand waarin de opgegeven datum valt.

Syntaxis

CMONTH(<*Duitdr*>)

<Duitdr>

De datumuitdrukking, in de huidige datumopmaak, waarvan u de naam van de bijbehorende maand als resultaat wilt geven.

Beschrijving

CMONTH( ) geeft als resultaat een tekenreeks die de naam bevat van de maand waarin de opgegeven datum valt.

Geef <*Duitdr*> op in de huidige datumopmaak die wordt bepaald door SET DATE, DBASEWIN.INI of de optie Internationaal in het Configuratiescherm van Windows (in die volgorde). Dat wil zeggen, de instellingen bij SET DATE hebben een hogere prioriteit dan die in DBASEWIN.INI, en de instellingen in DBASEWIN.INI hebben weer een hogere prioriteit dan de instellingen in het Configuratiescherm van Windows. Let er op dat <*Duitdr*> overeenkomt met de datumopmaak die wordt gebruikt op het moment dat het programma wordt uitgevoerd.

Als u een ongeldige datum doorgeeft aan CMONTH( ), wordt die omgezet in een geldige datum en wordt de naam van de maand als resultaat gegeven. Als u een lege uitdrukking of een andere dan een datumuitdrukking tussen accolades ({ }) doorgeeft aan CMONTH( ), wordt "Onbekend" als resultaat gegeven. Als u een andere dan een

datumuitdrukking of een uitdrukking die niet tussen accolades staat, doorgeeft aan CMONTH( ), wordt een fout als resultaat gegeven.

Voorbeeld

In het volgende voorbeeld worden CMONTH( ), DAY( ) en YEAR( ) gebruikt om in een tekenreeks de datum als resultaat te geven. De volledige datum bestaat uit de dag van de week, plus dag, maand en jaar.

```
SET TALK OFF
SET CENTURY ON
datum = {01-04-94}
? HeleDatum(datum) && Functie aanroepen. Geeft;
                          "vrijdag, 1 april 1994";
                          als resultaat

FUNCTION HeleDatum
PARAMETERS datum
hele_datum = CDOW(datum) +  ", " + LTRIM(STR(DAY(datum))) + ;
   " "  + LTRIM(CMONTH(datum)) + ;
   " " + LTRIM(STR(YEAR(datum)))
RETURN hele_datum
```

Zie ook

CDOW( ), DAY( ), MONTH( ), SET DATE, YEAR( )

COL( )

Invoer/uitvoer

Geeft als resultaat het nummer van de huidige kolompositie in het resultatenpaneel van het commandovenster of het huidige dBASE IV-venster. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows het kenmerk Left van een klasse om de horizontale positie op een formulier te bepalen.

Zie Help voor meer informatie over de syntaxis van COL( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het werken met formulieren in dBASE voor Windows.

COMMIT( )

Gedeelde gegevens

Beëindigt een transactie die is geïnitieerd met BEGINTRANS( ) en schrijft eventueel alle wijzigingen die tijdens de transactie zijn aangebracht naar de geopende bestanden. Geeft .T. als resultaat als de gegevens correct zijn doorgevoerd.

Syntaxis

COMMIT([<*databasenaam Tuitdr*>])

<databasenaam Tuitdr>

De naam van de database waarin de transactie moet worden uitgevoerd.

- Als u de transactie bent begonnen met BEGINTRANS(*<databasenaam Tuitdr>*), moet u COMMIT(*<databasenaam Tuitdr>*) geven. Als u in dit geval COMMIT( ) gebruikt, wordt de instructie genegeerd.
- Als u de transactie bent begonnen met BEGINTRANS( ), is het argument *<databasenaam Tuitdr>* bij COMMIT( ) optioneel. Als u het argument wel gebruikt, moet het verwijzen naar dezelfde database als de instructie met SET DATABASE TO die aan BEGINTRANS( ) vooraf is gegaan.

Beschrijving

Gebruik COMMIT( ) om de geopende transactie te beëindigen en eventuele wijzigingen op te slaan in de geopende bestanden. Gebruik ROLLBACK( ) om een transactie te beëindigen zonder de wijzigingen weg te schrijven. Zie BEGINTRANS( ) voor meer informatie over transacties.

Voorbeeld

In het volgende voorbeeld wordt een transactie gestart met BEGINTRANS( ). Er wordt een multi-user-versie van Bedrijf.dbf geopend en vervolgens wordt getracht alle waarden in het veld VerkTotnu 0 te maken. Een eventuele fout wordt waargenomen met ON ERROR. Dat gebeurt bijvoorbeeld als een andere gebruiker een record in Bedrijf.dbf heeft vergrendeld. Als er een fout optreedt, zorgt ROLLBACK( ) dat alle waarden worden hersteld. Gaat alles goed, dan worden de wijzigingen met COMMIT( ) naar schijf geschreven:

```
CLOSE ALL
SET PROCEDURE TO PROGRAM(1) ADDITIVE
SET EXCLUSIVE OFF

BEGINTRANS()

TransFt=.f.
ON ERROR DO TransFt      && Onderscheppen van fouten activeren

USE C:\VeelGebr\Bedrijf
REPLACE ALL VerkTotnu WITH 0
ON ERROR                 && ON ERROR uitschakelen

IF TransFt
  ? "Rollback"
  ROLLBACK()             && Gegevens herstellen
ELSE
  ? "Commit"
  COMMIT()               && Wijzigingen opslaan
ENDIF

PROC TransFt
WAIT "Waarschuwing: Transactie niet gelukt"
TransFt=.t.
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS. COMMIT( ) vervangt het commando COMMIT in dBASE IV.

Zie ook

BEGINTRANS( ), ROLLBACK( ), SET EXCLUSIVE

COMPILE — Programma's

Compileert programmabestanden (.PRG, .WFM) en maakt objectcodebestanden (.PRO, .WFO).

Syntaxis

COMPILE <*bestandsnaam*> | <*bestandsnaamfilter*>

<*bestandsnaam*> | <*bestandsnaamfilter*>

Het bestand dat gecompileerd moet worden. De optie <*bestandsnaamfilter*> toont een dialoogvenster waarin een bestand kan worden geselecteerd. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht. Als u een bestand zonder extensie opgeeft, gebruikt dBASE de extensie .PRG.

Beschrijving

Met COMPILE kunt u gecompileerde programmabestanden maken zonder de bestanden uit te voeren of te openen. Ook kunt u met dit commando alleen bepaalde bestanden compileren. U kunt een programma pas uitvoeren nadat het is gecompileerd. Omdat een gecompileerd programma niet kan worden gelezen of gewijzigd, beschermt u op deze manier de broncode tegen wijzigingen door gebruikers van het programma. Standaard worden gecompileerde objectbestanden opgeslagen in dezelfde directory als de broncodebestanden.

COMPILE heeft een aantal voordelen ten opzichte van het compileren van bestanden met DO, SET PROCEDURE of SET FORMAT:

- Door COMPILE worden de opgegeven bestanden niet uitgevoerd of geopend.
- Als u een applicatie schrijft die veel programmabestanden bevat, kunt u COMPILE gebruiken om alleen die programmabestanden te compileren die zijn gewijzigd in plaats van alle programmabestanden van de applicatie. Met FDATE( ) en FTIME( ) kunt u een datum- en of tijdbereik opgeven voor de bestanden die gecompileerd moeten worden.
- Met COMPILE <*bestandsnaamfilter*> kunt u groepen bestanden compileren die al of niet met elkaar te maken hebben.

Als tijdens de compilatie een syntaxisfout wordt waargenomen in het bronbestand, verschijnt een dialoogvenster met een melding over de fout en drie knoppen:

- *Annuleren* annuleert de compilatie (u kunt ook op *Esc* drukken).
- *Negeren* annuleert de compilatie van het programma met de syntaxisfout, maar gaat verder met het compileren van de overige bestanden die voldoen aan *<bestandsnaamfilter>* (mist u een globale bestandsnaam hebt opgegeven).
- *Corrigeren* stelt u in staat de fout te verbeteren door het broncodebestand te openen in een bewerkingsvenster. De cursor wordt in dat geval geplaatst op de plek waar de fout werd waargenomen.

Overdraagbaarheid

Standaard maken zowel dBASE IV als dBASE III PLUS gecompileerde objectbestanden in de huidige directory in plaats van in dezelfde directory als de broncodebestanden.

Zie ook

CLEAR PROGRAM, DO, FDATE( ), FTIME( ), SET COVERAGE, SET DEVELOPMENT, SET FORMAT, SET PROCEDURE

CONTINUE

Tabelindeling

Zet een zoekbewerking voort naar het volgende record dat voldoet aan de voorwaarden die zijn opgegeven in een eerdere instructie met LOCATE.

Syntaxis

CONTINUE

Beschrijving

CONTINUE zet de zoekbewerking voort die is gestart met de laatste instructie met LOCATE in het geselecteerde werkgebied. Nadat u een instructie met LOCATE hebt gegeven, wordt in de huidige tabel sequentieel gezocht naar het eerste record dat voldoet aan de zoekvoorwaarden. Tenzij WHILE *<voorwaarde>* is gebruikt, of voor *<bereik>* NEXT of REST is opgegeven, begint de zoekbewerking bij het eerste record.

Als een record is gevonden, wordt de recordaanwijzer naar het overeenkomende record verplaatst en verschijnt de melding **Record** = *<n>* (als SET TALK is ingeschakeld (ON)). Geef het commando CONTINUE om verder te zoeken. Als een volgend overeenkomend record wordt gevonden, wordt opnieuw de melding **Record** = *<n>* getoond en geeft FOUND( ) .T. als resultaat. Als geen overeenkomend record wordt gevonden, verschijnt de melding Einde van LOCATE-bereik en wordt de recordaanwijzer bij het laatste record in het LOCATE-bereik of aan het eind van het bestand geplaatst. FOUND( ) geeft dan .F. als resultaat.

Als u CONTINUE gebruikt zonder eerst een instructie met LOCATE op te geven voor de huidige tabel, verschijnt een foutmelding.

Voorbeeld

In het volgende voorbeeld wordt CONTINUE gebruikt om het volgende bedrijf te zoeken dat zicht niet in de regio NW bevindt. Het geheel is afhankelijk van het commando LOCATE:

```
USE Bedrijf
LOCATE FOR Regio <> "NW"
* Na LOCATE kan CONTINUE worden gebruikt
IF EOF()
* IF .NOT. FOUND() kan ook worden gebruikt
* om LOCATE/CONTINUE te testen
  ? "Geen bedrijven gevonden"
ENDIF
DO WHILE .NOT. EOF()
   ? Bedrijf, Regio
   CONTINUE
ENDDO
```

Zie ook

EOF( ), FIND, FOUND( ), LOCATE, SEEK, SEEK( )

CONVERT

Gedeelde gegevens

Voegt aan een tabel een _dbaselock-veld toe voor de opslag van multi-user-vergrendelingsinformatie.

Syntaxis

CONVERT
[TO <*Nuitdr*>]

TO <*Nuitdr*>

Geeft de lengte aan van het veld voor multi-user-informatie dat aan de huidige tabel moet worden toegevoegd. Het argument <*Nuitdr*> kan een getal van 8 tot en met 24 zijn. De standaardinstelling is 16.

Beschrijving

Met CONVERT voegt u een tekenveld, _dbaselock, toe aan de structuur van de huidige tabel. Met de optie TO <*Nuitdr*> kunt u de lengte voor het veld opgeven. Als u CONVERT gebruikt zonder de optie TO <*Nuitdr*>, is het veld 16 tekens breed. Als u de lengte van het veld _dbaselock wilt wijzigen nadat u CONVERT hebt gebruikt, gebruikt u CONVERT opnieuw voor dezelfde tabel. Met LKSYS( ) kunt u de inhoud van het veld _dbaselock bekijken.

Opmerking U kunt CONVERT alleen gebruiken als u het exclusieve gebruik hebt van de tabel (USE...EXCLUSIVE). Ook met u SET DELETED uitschakelen (OFF) voordat u CONVERT gebruikt of de tabel opnieuw indexeren (REINDEX) nadat u CONVERT hebt gebruikt.

Als u CONVERT gebruikt, wordt de huidige tabel gekopieerd naar een nieuw bestand met de extensie .CVT. Vervolgens wordt een nieuw .DBF-bestand met het veld _dbaselock gemaakt. Het .CVT-bestand bevat de oorspronkelijke bestandsstructuur.

Het veld _dbaselock bevat de volgende waarden:

Telling	Een hexadecimaal getal van 2 bytes dat wordt gebruikt door CHANGE( )
Tijd	Een hexadecimaal getal van 3 bytes waarin het tijdstip van de vergrendeling wordt opgeslagen
Datum	Een hexadecimaal getal van 3 bytes waarin de datum van de vergrendeling wordt opgeslagen
Naam	Een reeks van 0 tot 16 tekens van de naam van de gebruiker die de vergrendeling heeft geplaatst (als er een vergrendeling actief is)

De delen met de telling, de tijd en datum van het veld _dbaselock vormen altijd de eerste 8 tekens. Als u de standaardbreedte van 16 tekens accepteert, wordt de aanmeldingsnaam ingekort tot 8 tekens. Als u het veld minder dan 16 tekens breed maakt, wordt de aanmeldingsnaam ingekort tot het overgebleven aantal tekens. Als u 8 tekens opgeeft voor de breedte <*Nuitdr*>, verschijnt de aanmeldingsnaam helemaal niet.

Iedere keer dat een record wordt bijgewerkt, wordt het deel met de telling in het veld _dbaselock opnieuw geschreven. Als u een instructie met CHANGE( ) geeft, wordt dat gedeelte van schijf gelezen en vergeleken met de waarde die in het geheugen is opgeslagen op het moment dat het record in eerste instantie werd gelezen. Als die waarden verschillen, heeft een andere gebruiker het record gewijzigd en geeft CHANGE( ) .T. als resultaat. Zie CHANGE( ) voor meer informatie.

LKSYS( ) geeft als resultaat de delen met de aanmeldingsnaam, datum en tijd van het veld _dbaselock. Als u een bestandsvergrendeling instelt voor een tabel met een _dbaselock-veld, bevat het _dbaselock-veld in het eerste record de informatie die door CHANGE( ) en LKSYS( ) wordt gebruikt. Zie LKSYS( ) voor meer informatie.

Opmerking CONVERT is niet van invloed op SQL-databases of Paradox-tabellen.

Voorbeeld

De volgende reeks instructies wordt in het commandovenster gegeven om aan de actieve tabel een veld toe te voegen waarin multi-user-informatie wordt bijgehouden:

```
SET DELETED OFF
USE Bedrijf EXCLUSIVE
DISPLAY STRUCTURE       && Let op structuur
CONVERT TO 24
DISPLAY STRUCTURE       && Let op toegevoegde veld _dbaselock;
                           veld is 24 tekens breed
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CHANGE( ), FLOCK( ), LKSYS( ), LOCK( ), NETWORK( ), REINDEX, RLOCK( ), SET DELETED, SET EXCLUSIVE, SET LOCK, SET REPROCESS, UNLOCK, USE

COPY

Tabellen

Maakt een nieuwe tabel en kopieert uit de huidige tabel naar de nieuwe tabel. COPY stelt u ook in staat gegevens te exporteren naar andere bestanden dan dBASE-bestanden.

Syntaxis

COPY TO <*bestandsnaam*> | ?
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[FIELDS <*veldenlijst*>]
[[TYPE] SDF | DBMEMO3 | PARADOX | DBASE |
DELIMITED [WITH <*teken*> | BLANK]] |
[[WITH] PRODUCTION]

TO <*bestandsnaam*> | ?

Geeft de naam aan van de tabel die u wilt maken. COPY TO ? toont een dialoogvenster waarin u een nieuw doelbestand kunt opgeven. Als u een bestand zonder pad opgeeft, slaat dBASE voor Windows het bestand op in de huidige directory op de huidige schijf. Als u een bestand zonder extensie opgeeft en geen standaardtabeltype definieert met SET DBTYPE en ook geen van de opties voor TYPE opgeeft, gebruikt dBASE voor Windows de extensie .DBF.

U kunt ook records kopiëren naar een tabel in een database (gedefinieerd met het IDAPI-configuratieprogramma) door voor de tabelnaam de naam van de database op te geven (tussen dubbele punten), *:databasenaam:tabelnaam*. Als de database nog niet open is, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

<*bereik*>

Het aantal records dat naar <*bestandsnaam*> moet worden gekopieerd. RECORD <*n*> geeft een enkel record aan door middel van zijn recordnummer. NEXT <*n*> geeft *n* records aan, te beginnen bij het huidige record. ALL specificeert alle records. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR <*voorwaarde1*>
WHILE <*voorwaarde2*>

Bepaalt welke records worden beïnvloed door COPY. FOR beperkt COPY tot records die voldoen aan <*voorwaarde1*>. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang <*voorwaarde2*> waar is.

FIELDS <*veldenlijst*>

Geeft aan welke velden records naar de nieuwe database moeten worden gekopieerd.

[TYPE] SDF | DBMEMO3 | PARADOX | DBASE | DELIMITED [WITH <*teken*> | BLANK]

Geeft de structuur aan van het doelbestand. Het sleutelwoord TYPE is er alleen voor de leesbaarheid; het heeft geen gevolgen voor de werking van het commando. De volgende tabel geeft een overzicht van de ondersteunde bestandsindelingen:

Type	Omschrijving
SDF	SDF is een afkorting van System Data Format. Records in een SDF-bestand hebben een vaste lengte en het eind van een record wordt aangegeven door een regelterugloop en een regeldoorvoer. Als geen extensie opgeeft voor <*bestandsnaam*> wordt de extensie .TXT gebruikt.
DBMEMO3	Een tabelbestand (.DBF) en memobestanden (.DBT) met de indeling van dBASE III PLUS.
PARADOX	Een Paradox-tabel. Records worden gevormd door Paradox-rijen en velden door Paradox-kolommen. Deze bestanden krijgen de extensie .DB.
DBASE	Een dBASE-tabel (de standaardinstelling). Als u geen extensie opgeeft voor <*bestandsnaam*>, wordt de extensie .DBF gebruikt.
DELIMITED	Een tekstbestand met de volgende eigenschappen: Tekenvelden staan tussen aanhalingstekens of tussen de tekens die u opgeeft met WITH <*teken*>. Logische velden bevatten de tekens T of F. Numerieke velden bestaan uit getallen. Een nieuw record wordt aangegeven door de combinatie van een regelterugloop en een regeldoorvoer. Als u geen extensie opgeeft voor <*bestandsnaam*>, wordt de extensie .TXT gebruikt. **WITH <*teken*>** Plaatst tekenvelden in een tekstbestand met scheidingstekens tussen tekens die zijn opgegeven met <*teken*> in plaats van tussen aanhalingstekens. **WITH BLANK** Scheidt de velden in een tekstbestand met scheidingstekens door middel van spaties in plaats van komma's. De tekenvelden worden niet tussen aanhalingstekens of andere tekens geplaatst.

[WITH] PRODUCTION

Geeft aan dat het produktie-MDX-bestand ook moet worden gekopieerd. Deze optie is alleen geldig bij het kopiëren naar een andere dBASE-tabel.

Beschrijving

Met COPY kunt u een volledige tabel of een deel van een tabel kopiëren naar een bestand met hetzelfde of een ander type. Als er een index actief is, rangschikt COPY de records in de nieuwe tabel of in het nieuwe bestand op basis van de volgorde in de

index. Het commando COPY kopieert geen _dbaselock-velden in een tabel die is gemaakt met CONVERT.

Als u een tabel met een memoveld kopieert naar een andere dBASE-tabel, wordt een bestand gemaakt met dezelfde naam als de tabel, maar met de extensie .DBT. De inhoud van het memoveld wordt naar het .DBT-bestand gekopieerd. Als u echter de optie SDF of DELIMITED gebruikt en naar een ASCII-tekstbestand kopieert, wordt het memoveld niet naar een .DBT-bestand gekopieerd.

Ook verwijderde records worden naar het doelbestand gekopieerd (mits dat een dBASE-tabel is) behalve als ze worden uitgesloten met een FOR- of een WHILE-voorwaarde of als SET DELETED is ingeschakeld (ON). Ook in het doelbestand zijn verwijderde records gemarkeerd voor verwijdering.

U kunt met COPY een bestand maken dat velden uit meer dan een tabel bevat. U doet dat door de bronbestanden te openen in verschillende werkgebieden en een relatie tussen de tabellen te definiëren. Met SET FIELDS TO selecteert u de velden in elke tabel die u naar het nieuwe bestand wilt kopiëren. Voordat u de instructie met COPY geeft, moet SET FIELDS zijn ingeschakeld (ON) en moet het werkgebied met de hoofdtabel actief zijn.

Het commando COPY garandeert niet dat de bestanden die u maakt, compatibel zijn met andere programma's. Het kan zijn dat de opgegeven veldlengte of recordlengte of het opgegeven aantal velden of records niet compatibel is met andere software. Raadpleeg de documentatie voor de doelprogramma's voordat u tabellen exporteert met COPY.

Voorbeeld

In het volgende voorbeeld wordt COPY gebruikt om bepaalde velden uit twee gerelateerde tabellen naar een nieuwe Paradox-tabel te kopiëren:

```
SET SAFETY OFF
USE Contact Exclusive
INDEX ON CompCode TAG CompCode
USE Bedrijf IN SELECT()
SELECT Bedrijf
SET RELATION TO CompCode INTO Contact
COPY TO CntctLst FOR Bedrijf->Regio = "NW";
  FIELDS Bedrijf->Bedrijf,;
  Contact->CompCode, Contact->Contact, ;
  Bedrijf->Straat1, Bedrijf->Straat2, ;
  Bedrijf->Plaats, Bedrijf->Regio,;
  Bedrijf->Postcode TYPE PARADOX
CLOSE DATABASES
SET DBTYPE TO PARADOX
USE CntctLst
BROWSE
CLOSE DATABASES
SET DBTYPE TO
```

Zie ook

APPEND FROM, CONVERT, COPY FILE, COPY STRUCTURE, COPY TABLE, COPY TO...STRUCTURE EXTENDED, IMPORT, SET DELETED, SET FIELDS

COPY BINARY

Velden en records

Kopieert de inhoud van opgegeven binaire velden naar een bestand.

Syntaxis

COPY BINARY <*veldnaam*> TO <*bestandsnaam*>
[ADDITIVE]

<*veldnaam*>

Geeft het binaire veld aan dat u wilt kopiëren.

TO <*bestandsnaam*> | ?

Geeft de naam aan van het bestand waar de inhoud van het binaire veld naar toe moet worden gekopieerd. In het geval van vooringestelde binaire bestandstypen wordt de toepasselijke extensie gebruikt, zoals .BMP, .WAV, enzovoort. Voor eigen binaire veldtypen wordt standaard de extensie .TXT gebruikt.

ADDITIVE

Voegt de inhoud van het binaire veld toe aan het eind van een bestand. Zonder deze optie wordt de oorspronkelijke inhoud van het bestand overschreven.

Beschrijving

Met COPY BINARY kunt u gegevens uit een binair veld in het huidige record naar een bestand exporteren. In binaire velden kunt u tekst, afbeeldingen, geluid, videobeelden en andere binaire informatie opslaan.

Als u de optie ADDITIVE opgeeft, wordt de inhoud van het binaire veld toegevoegd aan het eind van het opgegeven bestand. Op die manier kunt u de inhoud van binaire velden in meerdere records combineren. Als u ADDITIVE niet gebruikt, verschijnt een waarschuwing als een bestaand bestand dreigt te worden overschreven (mits SET SAFETY is ingeschakeld (ON)). Let er op dat de gegevens uit velden met een aantal vooringestelde binaire gegevenstypen niet kunnen worden gecombineerd. U kunt bijvoorbeeld niet meer dan een afbeelding in een binair veld of bestand opslaan. In die gevallen kunt u de optie ADDITIVE van COPY BINARY dus niet gebruiken.

Voorbeeld

In het volgende voorbeeld wordt COPY BINARY gebruikt om geluidsgegevens uit binaire velden te kopiëren naar externe bestanden. De nieuwe bestanden krijgen een naam die bestaat uit de inhoud van het veld Naam, het cijfer 2 om het nieuwe bestand te onderscheiden van het oude bestand en de extensie .WAV:

```
USE Afbeeld
DO WHILE .NOT. EOF()
   wav_best = TRIM(Naam) + "2.WAV"
   COPY BINARY Geluid TO &wav_best
   SKIP
ENDDO
CLOSE ALL
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

APPEND MEMO, BINTYPE( ), CLASS IMAGE, COPY, COPY FILE, COPY MEMO, PLAY SOUND, REPLACE BINARY, RESTORE IMAGE

COPY FILE

Stations- en bestandsfuncties

Maakt een kopie van een bestand.

Syntaxis

COPY FILE <*bestandsnaam1*> | ? | <*bestandsnaamfilter1*>
   TO <*bestandsnaam2*> | ? | <*bestandsnaamfilter2*>

<*bestandsnaam1*> | ? | <*bestandsnaamfilter1*>

Geeft de naam van het bestand dat u wilt kopiëren (wordt ook wel het bronbestand genoemd). ? en <*bestandsnaamfilter1*> tonen een dialoogvenster waarin u een bestand kunt kiezen om te kopiëren.

Als u een bronbestand zonder pad opgeeft, wordt het bestand eerst gezocht in de huidige directory en vervolgens in het pad dat u opgeeft met SET PATH. Als u een bronbestand zonder extensie opgeeft, wordt geen extensie gebruikt.

TO <*bestandsnaam2*> | ? | <*bestandsnaamfilter2*>

Geeft het doelbestand aan dat door COPY FILE wordt gemaakt of overschreven. De opties ? en <*bestandsnaamfilter2*> tonen een dialoogvenster waarin u de naam en de directory van het doelbestand kunt opgeven.

Beschrijving

COPY FILE is een dBASE-commando waarmee u een bestaand bestand kunt kopiëren op het niveau van het besturingssysteem. Met COPY FILE maakt u een kopie van een enkel bestand van elk willekeurig type.

COPY FILE verschilt van de DOS-opdracht COPY omdat u geen jokers kunt gebruiken om meer dan een bestand tegelijk te kopiëren. Als u dat wel wilt doen, moet u !, RUN of DOS gebruiken en de DOS-opdracht COPY uitvoeren.

Als SET SAFETY is ingeschakeld (ON) en er bestaat al een bestand met dezelfde naam als het doelbestand, verschijnt een dialoogvenster waarin u wordt gevraagd of u het bestand wilt overschrijven. Als SET SAFETY is uitgeschakeld (OFF), wordt een bestaand bestand met dezelfde naam als het doelbestand zonder waarschuwing overschreven.

COPY FILE kopieert niet automatisch een .DBT- of .MDX-bestand dat bij een .DBF-bestand hoort. Als u bijvoorbeeld een tabelbestand kopieert dat memovelden bevat en niet het bijbehorende .DBT-bestand kopieert, verschijnt een foutmelding als u probeert het bestand te gebruiken. In dergelijke gevallen kunt u beter COPY gebruiken en niet COPY FILE. Verder moet u een dBASE-bestand eerst sluiten voordat u het kunt kopiëren met COPY FILE.

Voorbeeld

In de volgende voorbeelden wordt COPY FILE gebruikt:

```
COPY FILE Tijd.prg TO B:Tijd.prg
* Tijd.prg naar een diskette in B: kopiëren
COPY FILE *.DBF TO B:*.DBF
* Dialoogvenster "Bronbestand openen" verschijnt
```

Overdraagbaarheid

De opties ? en *<bestandsnaamfilter>* worden niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

COPY, COPY INDEXES, COPY MEMO, COPY STRUCTURE, COPY TO ARRAY, COPY TO...STRUCTURE EXTENDED, RENAME, SET FULLPATH, SET PATH

COPY INDEXES

Tabelindeling

Kopieert een lijst van .NDX-bestanden naar indexlabels in een enkel .MDX-bestand. (Kan alleen worden toegepast voor .NDX-indexen voor dBASE-tabellen.)

Syntaxis

COPY INDEXES *<.ndx-bestandsnamenlijst>*
[TO *<.mdx-bestandsnaam>* | ?]

<.ndx-bestandsnamenlijst>

Geeft een lijst op van .NDX-bestanden (maximaal tien) waarvoor u indexlabels wilt maken.

TO *<.mdx-bestandsnaam>* | ?

Geeft de naam op van het .MDX-bestand waaraan u indexlabels wilt toevoegen. Standaard krijgt het indexbestand dezelfde naam als de huidige tabel plus de extensie

.MDX en wordt het bestand opgeslagen in de huidige directory. De optie ? toont een dialoogvenster, waarin u de naam en de plaats van het .MDX-bestand kunt opgeven.

Beschrijving

Het commando COPY INDEXES converteert een lijst van .NDX-bestanden naar indexlabels in een enkel .MDX-bestand. Als u geen naam opgeeft bij TO <*.mdx-bestandsnaam*>, worden de indexlabels gemaakt in het produktie-MDX-bestand, dat dezelfde naam heeft as de tabel (en dat automatisch wordt geopend als u de bijbehorende tabel opent). Als er geen produktie-MDX-bestand bestaat, wordt het gemaakt met dezelfde naam als de tabel. In het geval van dBASE-tabellen wordt de tabel-header bijgewerkt om de aanwezigheid van een produktie-index aan te geven. Als u de optie TO <*.mdx-bestandsnaam*> gebruikt, worden de indexlabels naar het opgegeven .MDX-bestand geschreven. U kunt in een enkel .MDX-bestand maximaal 47 individuele labels maken.

De .NDX-bestanden die u wilt kopiëren en de bijbehorende tabel moet zijn geopend voordat u het commando COPY INDEXES kunt gebruiken. In een multi-user-omgeving moet de tabel die bij de indexen hoort, exclusief geopend zijn.

Voorbeeld

In het volgende voorbeeld worden eerst twee .NDX-indexen gemaakt voor de tabel Bedrijf. Vervolgens wordt COPY INDEXES gebruikt om voor de huidige tabel labels te maken waarbij de twee bestaande .NDX-bestanden als bron worden gebruikt:

```
USE Bedrijf EXCLUSIVE
INDEX ON Bedrijf TO Bedrijf    && .ndx maken
INDEX ON CompCode TO CompCode  && .ndx maken
* Nu bestaan twee .NDX-indexen
SET INDEX TO Bedrijf, CompCode
COPY INDEXES Bedrijf, CompCode
SET ORDER TO TAG Bedrijf
BROWSE FIELDS Bedrijf, CompCode
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

COPY TAG, DELETE TAG, INDEX, MDX( ), NDX( ), SET INDEX, SET ORDER, TAG( ), TAGNO( ), TAGCOUNT( ), USE

COPY MEMO

Velden en records

Kopieert de inhoud van het opgegeven memoveld naar een bestand.

Syntaxis

COPY MEMO <*memoveld*> TO <*bestandsnaam*> | ?
[ADDITIVE]

<*memoveld*>

Geeft aan welk memoveld u wilt kopiëren.

TO <*bestandsnaam*> | ?

Geeft de naam aan van het tekstbestand waar de tekst naar toe moet worden gekopieerd. Standaard wordt de extensie TXT gebruikt en wordt het bestand opgeslagen in de huidige directory. De optie ? toont een dialoogvenster waarin u de naam opgeeft van het doelbestand en van de directory waar het moet worden opgeslagen.

ADDITIVE

Voegt de inhoud van het memoveld toe aan het eind van een bestaand tekstbestand. Zonder de optie ADDITIVE wordt een bestaand bestand met dezelfde naam als het doelbestand overschreven.

Beschrijving

Met COPY MEMO kunt u de tekst uit het memoveld in het huidige record naar een tekstbestand kopiëren. U kunt COPY MEMO ook gebruiken om afbeeldingen of andere binaire gegevens naar een bestand te kopiëren. Voor de opslag van afbeeldingen, geluid en andere eigen binaire informatie wordt echter het gebruik van binaire velden aanbevolen.

Als u de optie ADDITIVE gebruikt, wordt de inhoud van het memoveld toegevoegd aan het eind van het opgegeven bestand. Op die manier kunt u de inhoud van memovelden uit meer dan een record combineren. Als u ADDITIVE niet gebruikt, wordt een waarschuwing getoond voordat een bestaand bestand wordt overschreven (mits SET SAFETY is ingeschakeld (ON)). U kunt in een memoveld of in een bestand slechts één afbeelding opslaan, dus kunt u de optie ADDITIVE van COPY MEMO niet gebruiken als u een afbeelding kopieert naar een bestand. (Met RESTORE IMAGE kunt u een afbeelding tonen die is opgeslagen in een memoveld of in een tekstbestand.)

Voorbeeld

In het volgende voorbeeld wordt COPY MEMO gebruikt om een tekstbestand te maken met de onopgemaakte inhoud van alle memovelden (het veld Notities) in de tabel Bedrijf:

```
USE Bedrijf
IF .NOT. FILE("Test.Txt")
  COPY MEMO Notities TO Test.Txt
  SKIP
ENDIF
DO WHILE .NOT. EOF()
  COPY MEMO Notities TO Test.TXT ADDITIVE
  SKIP
```

```
ENDDO
MODI COMM Test.TXT      && Alle memo's in .TXT-bestand
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

APPEND MEMO, COPY, COPY BINARY, COPY FILE, REPLACE BINARY, REPLACE OLE

COPY STRUCTURE — Tabellen

Maakt een lege tabel met dezelfde structuur als de huidige tabel.

Syntaxis

COPY STRUCTURE TO <*bestandsnaam*> | ? | <*bestandsnaamfilter*>
[[TYPE] PARADOX | DBASE]
[FIELDS <*veldenlijst*>]
[[WITH] PRODUCTION]

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>

De naam van de tabel die u wilt maken. COPY STRUCTURE TO ? en COPY STRUCTURE TO <*bestandsnaamfilter*> tonen een dialoogvenster, waarin u de naam van de doeltabel kunt opgeven. Als u een tabelnaam zonder pad opgeeft, wordt de tabel opgeslagen in de huidige directory op het huidige station. Als u een tabelnaam zonder extensie opgeeft, een standaardtabeltype definieert met SET DBTYPE of een van de TYPE-opties gebruikt, gebruikt dBASE voor Windows de extensie .DBF.

U kunt ook een structuur kopiëren naar een tabel in een database (gedefinieerd met de IDAPI-configuratieprogramma) door voor de naam van de tabel de database op te geven (tussen dubbele punten), zoals *:databasenaam:tabelnaam*. Als de database nog niet open is, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

[TYPE] PARADOX | DBASE

Geeft het type tabel aan dat u wilt maken. Als u hier iets opgeeft, heeft dat een hogere prioriteit dan de huidige instelling voor DBTYPE. Het sleutelwoord TYPE is alleen beschikbaar voor de leesbaarheid; het heeft geen gevolgen voor de werking van het commando.

Als u PARADOX opgeeft, wordt een Paradox-tabel gemaakt met de extensie .DB.

Als u DBASE opgeeft, wordt een dBASE-tabel gemaakt (de standaardinstelling). Als u geen extensie opgeeft voor *<bestandsnaam>*, gebruikt dBASE voor Windows de extensie .DBF.

FIELDS *<veldenlijst>*

Geeft aan welke velden in de structuur van de nieuwe tabel moeten worden opgenomen. De volgorde van de velden wordt bepaald door *<veldenlijst>*.

[WITH] PRODUCTION

Maakt een produktie-MDX-bestand voor de nieuwe tabel. Het nieuwe indexbestand bevat dezelfde indexlabels als het produktie-indexbestand dat bij de oorspronkelijke tabel hoort.

Beschrijving

Het commando COPY STRUCTURE kopieert de structuur van de huidige tabel, maar kopieert geen records. Als SET SAFETY is uitgeschakeld (OFF), worden bestaande tabellen met dezelfde naam zonder waarschuwing overschreven.

Het commando COPY STRUCTURE kopieert de volledige tabelstructuur, tenzij de optie FIELDS of het commando SET FIELDS is gebruikt. Als u COPY STRUCTURE gebruikt zonder de optie FIELDS *<veldenlijst>*, worden de velden die zijn ingesteld met SET FIELDS TO naar de nieuwe tabel gekopieerd. Het veld _dbaselock dat met het commando CONVERT is gemaakt, wordt niet naar nieuwe tabellen gekopieerd.

U kunt met COPY STRUCTURE een lege tabelstructuur maken met velden uit meerdere tabellen. Dat gaat als volgt:

1 Open de brontabellen in verschillende werkgebieden.

2 Geef bij de optie FIELDS *<veldenlijst>* alle gewenste velden op. Denk er aan bij velden die niet in de huidige tabel staan ook het tabelalias op te geven.

U kunt ook met het commando SET RELATION een relatie instellen tussen tabellen en vervolgens met COPY STRUCTURE velden uit de gerelateerde tabellen kopiëren.

Voorbeeld

In het volgende voorbeeld wordt COPY STRUCTURE gebruikt om bepaalde velden uit twee geopende tabellen naar een nieuwe tabel te kopiëren:

```
USE Contact
USE Bedrijf IN SELECT()
COPY STRUCTURE TO CntctLst ;
  FIELDS Bedrijf->Bedrijf,;
  Contact->CompCode, Contact->Contact, ;
  Bedrijf->Straat1, Bedrijf->Straat2, ;
  Bedrijf->Plaats, Bedrijf->Regio, ;
  Bedrijf->Postcode
USE CntctLst
APPEND
CLOSE ALL
```

Zie ook

APPEND, APPEND FROM, COPY, COPY TO...STRUCTURE EXTENDED, DISPLAY STRUCTURE, MODIFY STRUCTURE, SET FIELDS, SET SAFETY

COPY TABLE — Tabellen

Kopieert een opgegeven tabel.

Syntaxis

COPY TABLE <*brontabelnaam*> TO <*doeltabelnaam*>
[[TYPE] DBASE | PARADOX]

<*brontabelnaam*>

De naam van de tabel die u wilt kopiëren. U kunt ook een tabel in een database (gedefinieerd met het IDAPI-configuratieprogramma) kopiëren door voor de naam van de tabel de naam van de database op te geven (tussen dubbele punten), zoals *:databasenaam:tabelnaam*. Als de database nog niet open is, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

<*doeltabelnaam*>

De naam van de tabel die u wilt maken. Het tabeltype is gelijk aan dat van de brontabel. Als u een tabel in een database kopieert, moet u voor de doeltabel dezelfde database opgeven.

[TYPE] PARADOX | DBASE

Geeft aan welk type tabel u wilt kopiëren: een Paradox- of een dBASE-tabel. Dit heeft een hogere prioriteit dan de instelling voor DBTYPE.

Beschrijving

Met het commando COPY TABLE kunt u tabellen en hun bijbehorende .NDX- en .MDX-indexbestanden kopiëren. In het geval van Paradox-tabellen worden ook de bijbehorende indexen, BLOB- en .VAL-bestanden gekopieerd. Let er op dat de tabel niet in gebruik is als u die tracht te kopiëren.

Voorbeeld

In het volgende voorbeeld wordt COPY TABLE gebruikt om een kopie te maken van Klanten.DBF zonder die tabel eerst te openen:

```
COPY TABLE Klanten TO Regio1
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

COPY, COPY FILE, DELETE FILE, DELETE TABLE, ERASE,

COPY TAG

Tabelindeling

Kopieert .MDX-indexlabels naar .NDX-bestanden. (Dit commando is uitsluitend geldig voor dBASE-tabellen.)

Syntaxis

COPY TAG *<labelnaam>*
  [OF *.MDX-bestandsnaam*]
  TO *<.NDX-bestandsnaam>*

<labelnaam>

Geeft een .MDX-indexlabel aan dat u wilt kopiëren.

OF *<.MDX-bestandsnaam>*

Het .MDX-bestand dat de opgegeven indexlabel bevat. Als u een bestand zonder pad opgeeft, wordt het bestand eerst in de huidige directory gezocht en vervolgens in het pad dat is ingesteld met SET PATH. Als u een bestand zonder extensie opgeeft, gebruikt dBASE de extensie .MDX.

TO *<.NDX-bestandsnaam>*

De naam van het .NDX-bestand waar u de indexlabel naar toe wilt kopiëren. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gemaakt. Als u een bestand zonder extensie opgeeft, wordt de extensie .NDX gebruikt.

Beschrijving

COPY TAG maakt afzonderlijke .NDX-bestanden van de indexlabels in de produktie-index of in een opgegeven .MDX-bestand. U kunt dit commando bijvoorbeeld gebruiken als tabellen en indexen die u maakt, ook worden gebruikt in eerder versies van dBASE.

Voordat u het commando COPY TAG geeft, moet het .MDX-bestand en de bijbehorende tabel zijn geopend. Per keer kan slechts één label worden gekopieerd. Een FOR-clausule van een.MDX-bestandslabel wordt genegeerd daar deze niet door .NDX-bestanden worden ondersteund.

Voorbeeld

In het volgende voorbeeld wordt COPY TAG gebruikt om een .NDX-indexbestand te maken op basis van een geopende labelindex:

```
DELETE FILE Bedrijf.ndx && Bestaande index verwijderen
USE Bedrijf EXCLUSIVE
INDEX ON Bedrijf TAG Bedrijf
```

```
* TAG-index bestaat nu
COPY TAG Bedrijf TO Bedrijf
* COPY TAG maakt een .NDX-index
SET INDEX TO Bedrijf
BROWSE FIELDS Bedrijf, CompCode
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

COPY INDEXES, FOR( ), INDEX, MDX( ), NDX( ), SET INDEX, SET ORDER, TAG( ), TAGCOUNT( ), TAGNO( )

COPY TO ARRAY

Velden en records

Kopieert gegevens uit niet-memovelden in de huidige tabel, overschrijft elementen in een bestaande array en verplaatst de recordaanwijzer naar het laatst gekopieerde record.

Syntaxis

COPY TO ARRAY <*array-naam*>
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[FIELDS <*veldenlijst*>]

<*bereik*>

Het aantal records dat naar de opgegeven array moet worden gekopieerd. RECORD <*n*> geeft een enkel record aan door middel van zijn recordnummer. (In het geval van tabellen die niet over recordnummers beschikken, kunt u ook een bladwijzer opgeven.) NEXT <*n*> geeft *n* records aan, te beginnen bij het huidige record. ALL specificeert alle records. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR <*voorwaarde1*>
WHILE <*voorwaarde2*>

Bepaalt welke records worden beïnvloed door COPY TO ARRAY. Met FOR beperkt u COPY TO ARRAY tot records die voldoen aan <*voorwaarde1*>. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang <*voorwaarde2*> waar is.

FIELDS <*veldenlijst*>

Kopieert gegevens uit de velden in <*veldenlijst*> in de opgegeven volgorde. Zonder FIELDS worden alle velden gekopieerd die in de array passen en wel in de volgorde waarin ze in de huidige tabel staan.

Beschrijving

Met COPY TO ARRAY kunt u records uit de huidige tabel naar een bestaande array kopiëren. Wilt u bepaalde records en velden kopiëren, dan moet u DECLARE Voorbld[*<variabele1>*,*<variabele2>*] gebruiken, waarbij *<variabele1>* het maximumaantal records is dat u wilt kopiëren en *<variabele2>* het maximumaantal velden dat u wilt kopiëren. Als u DECLARE Voorbld[2,3] gebruikt, kunt u maximaal drie velden uit twee records kopiëren. Als u gegevens uit alle velden (met uitzondering van memovelden) in alle records in een tabel wilt kopiëren, moet u DECLARE Voorbld[RECCOUNT( ), FCOUNT( )] gebruiken.

Het eerste indexteken in een tweedimensionale array is het aantal records dat moet worden gekopieerd en het tweede indexteken het aantal velden. Als u bijvoorbeeld DECLARE Test[2,3] gebruikt, kopieert de instructie COPY TO ARRAY Test drie velden uit twee records.

COPY TO ARRAY kan naar array's met meer dan twee dimensies (meerdimensionale array's) kopiëren, maar alleen de laatste twee indextekens worden gebruikt. Het op een na laatste indexteken in *<array-naam>* bepaalt hoeveel records de array kan bevatten en het laatste indexteken bepaalt het aantal velden uit elk record dat in de array kan worden opgeslagen. De instructies DECLARE Test[4,5,2,3] en DECLARE Test[2,3] maken beide array waar met COPY TO ARRAY drie velden uit twee records naar toe kunnen worden gekopieerd.

Als de array eendimensionaal is, kan met COPY TO ARRAY slechts één record worden gekopieerd. In dat geval bepaalt het aantal elementen in de array het aantal velden dat kan worden gekopieerd. Als u bijvoorbeeld DECLARE Test[5] gebruikt, kunt u met COPY TO ARRAY de eerste vijf velden uit het huidige record naar de array kopiëren.

COPY TO ARRAY kopieert op volgorde van recordnummer of index en, binnen elk record, op volgorde van veldnummer, tenzij u de optie FIELDS hebt gebruikt om de volgorde van de velden op te geven.

Als het laatste indexteken in de definitie van de array groter is dan het aantal velden dat vanuit de tabel wordt gekopieerd, blijven de overige array-elementen geïnitialiseerd met .F. of met de eerder opgeslagen waarde. Op dezelfde manier geldt dat als het op een na laatste indexteken in de definitie van de array groter is dan het aantal records dat uit de tabel wordt gekopieerd, de overige array-elementen geïnitialiseerd blijven met .F. of met de eerder opgeslagen waarde.

Voorbeeld

In het volgende voorbeeld wordt COPY TO ARRAY gebruikt om uit een weergave die is gemaakt met SET RELATION, vijf records te kopiëren naar een array van vijf rijen en drie kolommen:

```
CLEAR
SET TALK OFF
USE Contact ORDER CompCode IN SELECT()
USE Bedrijf IN SELECT()
SELECT Bedrijf
SET RELATION TO CompCode INTO Contact;
                              && Relatie op Compcode;
                                 instellen
```

```
haal_rec = 5
DECLARE BedrCtc[haal_rec,3]     && Array maken van;
                                   5 records met 3;
                                   velden per record
COPY TO ARRAY BedrCtc ;
   NEXT haal_rec FIELDS Bedrijf->Bedrijf, ;
   Contact->Contact, Bedrijf-> VerkTotnu
Record = 1
? "Bedrijf" AT 10, "Contact" AT 40, ;
  "Omzet" AT 60
?
DO WHILE Record <= haal_rec
 ? BedrCtc[Record,1] AT 10, BedrCtc[Record,2] AT 40,;
   LTRIM(STR(BedrCtc[Record,3])) AT 60
   Record = Record + 1
ENDDO
CLOSE ALL
```

Zie ook

APPEND FROM ARRAY, DECLARE, REPLACE FROM ARRAY, SET FIELDS, STORE MEMO, COPY TO...STRUCTURE EXTENDED

COPY TO...STRUCTURE EXTENDED

Tabellen

Maakt een nieuwe tabel waarvan de records de structuur van de huidige tabel bevatten.

Syntaxis

COPY TO *<bestandsnaam>* | ?
   STRUCTURE EXTENDED
   [[TYPE] PARADOX | DBASE]

of

COPY STRUCTURE EXTENDED TO *<bestandsnaam>* | ?
   [[TYPE] PARADOX | DBASE]

<bestandsnaam> | ?

De naam van de tabel die u wilt maken en die de structuur van de huidige tabel moet bevatten. COPY TO ? en COPY STRUCTURE EXTENDED TO ? tonen een dialoogvenster waarin u de naam van de doeltabel kunt opgeven. Als u een tabelnaam zonder pad opgeeft, wordt de tabel opgeslagen in de huidige directory op het huidige station. Als u een tabelnaam zonder extensie opgeeft, een standaardtabeltype definieert met SET DBTYPE of een van de TYPE-opties gebruikt, gebruikt dBASE voor Windows de extensie .DBF.

U kunt ook een tabel maken in een database (gedefinieerd met de IDAPI-configuratieprogramma) door voor de naam van de tabel de naam van de database op te geven (tussen dubbele punten), zoals *:databasenaam:tabelnaam*. Als de database nog niet open is, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een

aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

[TYPE] PARADOX | DBASE

Geeft aan welk type tabel moet worden gemaakt. Het sleutelwoord TYPE is er alleen voor de leesbaarheid; het heeft geen gevolgen voor de werking van het commando.

Als u PARADOX opgeeft, wordt een Paradox-tabel gemaakt met de extensie .DB.

Als u DBASE opgeeft, wordt een dBASE-tabel gemaakt (de standaardinstelling). Als u geen extensie opgeeft voor *<bestandsnaam>*, gebruikt dBASE voor Windows de extensie .DBF.

Beschrijving

COPY TO...STRUCTURE EXTENDED kopieert de structuur van de huidige tabel naar records in een nieuwe tabel.

COPY TO...STRUCTURE EXTENDED definieert eerst een tabel (een zogenaamde structuurtabel) met vijf velden met vaste namen, typen en lengte. Als de structuurtabel is gedefinieerd, voegt COPY TO...STRUCTURE EXTENDED daar records aan toe die informatie verstrekken over elk veld in de huidige tabel. De velden in de structuurtabel bevatten de volgende informatie over de velden in de huidige tabel:

Veld	Inhoud
FIELD_NAME	Tekenveld dat de naam van het veld bevat.
FIELD_TYPE	Tekenveld dat het gegevenstype van het veld bevat.
FIELD_LEN	Numeriek veld dat de lengte van het veld bevat.
FIELD_DEC	Numeriek veld dat het aantal decimalen voor numerieke en zwevende gegevens bevat.
FIELD_IDX	Tekenveld (wordt in dBASE III PLUS niet gemaakt) dat aangeeft of voor bepaalde velden indexlabels zijn gemaakt toen de huidige tabel werd gemaakt.

Als het proces is voltooid, bevat de structuurtabel evenveel velden als de huidige tabel records bevat. U kunt vervolgens CREATE...FROM gebruiken om een nieuwe tabel te maken aan de hand van de informatie in de structuurtabel.

Het veld _dbaselock dat is gemaakt met het commando CONVERT, wordt niet naar structuurtabellen gekopieerd.

Voorbeeld

In het volgende voorbeeld wordt COPY TO ... STRUCTURE EXTENDED gebruikt om de tabel Naam te maken die als records de structuur van de tabel Klanten bevat. De structuur wordt vervolgens gewijzigd door de velden voorbij het veld Postcode te verwijderen. Daarna wordt met CREATE een nieuwe tabel gemaakt met de verkorte structuur en worden de records uit Klanten toegevoegd. Een nieuwe tabel maken met COPY TO .. STRUCTURE EXTENDED is ook bruikbaar om een beschadigd .DBF-bestand te repareren:

```
SET TALK OFF
SET SAFETY OFF
```

```
USE Klanten
COPY TO Naam STRUCTURE EXTENDED
USE Naam EXCLUSIVE
DELETE FOR RECNO() >7   && Alle velden voorbij Postcode uitsluiten
PACK
CREATE Naam2 FROM Naam
DISPLAY STRUCTURE       && Naam2 gebruiken
? ALIAS() +" bevat nu " + LTRIM(STR(RECCOUNT() )) +;
  " records."
WAIT
APPEND FROM Klanten
?
? ALIAS() +" bevat nu " + LTRIM(STR(RECCOUNT() )) +;
  " records."
WAIT
GO TOP
BROWSE
```

Zie ook

COPY, COPY STRUCTURE, CREATE, CREATE...FROM, CREATE...STRUCTURE EXTENDED, DISPLAY STRUCTURE, LIST STRUCTURE, MODIFY STRUCTURE, SET SAFETY

COS( )

Numerieke gegevens

Geeft als resultaat de cosinus van een hoek.

Syntaxis

COS(<*Nuitdr*>)

<Nuitdr>

De grootte van de hoek in radialen. Gebruik DTOR( ) om de grootte van een hoek in graden om te zetten in radialen. De cosinus van een hoek van 30 graden bepaalt u bijvoorbeeld met COS(DTOR(30)).

Beschrijving

COS( ) berekent de verhouding tussen de aanliggende rechthoekszijde van een hoek en de hypotenusa van een rechthoekige driehoek. COS( ) geeft een zwevend getal van -1 tot 1 als resultaat. Als <*Nuitdr*> pi/2 of 3*pi/2 radialen is, geeft COS( ) 0 als resultaat.

Met SET DECIMALS kunt u het aantal decimalen instellen dat door COS( ) wordt weergegeven.

De secans van een hoek is het omgekeerde van de cosinus van een hoek. De secans berekent u met 1/COS( ).

Voorbeeld

In het volgende voorbeeld wordt COS( ) gebruikt om de lengte van een dakspant te berekenen nadat de gebruiker de breedte van de ruimte en de hoek van het dak heeft ingevoerd:

```
SET PROCEDURE TO PROGRAM(1) ADDITIVE
LOCAL f
f = NEW PFORM()
f.Open()

CLASS PFORM OF FORM
   this.HelpFile = ""
   this.Width =          43.00
   this.Height =          10.47
   this.Left =          46.60
   this.Text = "Spantberekening"
   this.Top =           6.94
   this.HelpId = ""

   DEFINE ENTRYFIELD KAMER OF THIS;
       PROPERTY;
         Width           4.00,;
         Picture "999",;
         Height           1.00,;
         Left          32.00,;
         Top           2.00,;
         Border .T.,;
         Value           0

   DEFINE ENTRYFIELD HOEK OF THIS;
       PROPERTY;
         Width           4.00,;
         Picture "99",;
         Height           1.00,;
         Left          32.00,;
         Top           4.00,;
         Border .T.,;
         Value           0

   DEFINE ENTRYFIELD SPANTLENGTE OF THIS;
       PROPERTY;
         Width           6.00,;
         Height           1.00,;
         Left          32.00,;
         Top           6.00,;
         Border .T.,;
         Value           0.00,;
         OnGotFocus RESULTAAT

   DEFINE TEXT TX1 OF THIS;
       PROPERTY;
         Width          26.00,;
         Height           1.00,;
```

```
        Left          3.00,;
        Text "Geef kamerbreedte op:",;
        Top          2.00,;
        ColorNormal "N/W",;
        Border .F.

    DEFINE TEXT TX2 OF THIS;
        PROPERTY;
        Width         27.00,;
        Height          1.00,;
        Left          3.00,;
        Text "Geef dakhoek op:",;
        Top          4.00,;
        ColorNormal "N/W",;
        Border .F.

    DEFINE TEXT TX3 OF THIS;
        PROPERTY;
        Width         29.00,;
        Height          1.00,;
        Left          3.00,;
        Text "Spant afkorten tot lengte:",;
        Top          6.00,;
        ColorNormal "R/W",;
        Border .F.

    DEFINE PUSHBUTTON SLUITEN OF THIS;
        PROPERTY;
        Width         15.60,;
        Default .T.,;
        OnClick {;Form.Close()},;
        Height          1.12,;
        Left         14.00,;
        Text "Sluiten",;
        Top          8.00,;
        ColorNormal "N/W"

ENDCLASS

FUNCTION Resultaat
Form.Spantlengte.Value=LTRIM(STR((Form.Kamer.Value/2);
  /COS(DTOR(Form.Hoek.Value)),6,2))
RETURN .T.
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ACOS( ), DTOR( ), PI( ), RTOD( ), SET DECIMALS, SIN( ), TAN( )

COUNT

Bepaalt het aantal tabelrecords dat voldoet aan opgegeven voorwaarden.

Syntaxis

COUNT
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[TO <*variabele*>]

<*bereik*>

Het aantal records waarin de records die voldoen aan de voorwaarde moeten worden geteld. RECORD <*n*> geeft een enkel record aan door middel van zijn recordnummer. NEXT <*n*> geeft *n* records aan, te beginnen bij het huidige record. ALL specificeert alle records. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR <*voorwaarde1*>
WHILE <*voorwaarde2*>

Bepaalt welke records worden beïnvloed door COUNT. FOR beperkt COUNT tot records die voldoen aan <*voorwaarde1*>, vanaf het eerste record in de tabel of het bereik tot het eind van de tabel of het bereik. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang <*voorwaarde2*> waar is.

TO <*variabele*>

Slaat het resultaat van COUNT, een getal, op in de opgegeven geheugenvariabele.

Beschrijving

Met COUNT kunt u het aantal records bepalen dat voldoet aan een opgegeven voorwaarde. Als SET TALK is ingeschakeld (ON), geeft het commando COUNT ook het totaal aantal records weer. Als SET DELETED is ingeschakeld (ON), worden geen records meegeteld die zijn gemarkeerd voor verwijdering.

Als SET LOCK in a multi-user-omgeving is ingeschakeld (ON, de standaardinstelling), wordt de tabel tijdens de bewerking automatisch vergrendeld. Als de bewerking is voltooid, wordt de vergrendeling weer opgeheven. U kunt de bewerking ook uitvoeren als SET LOCK is uitgeschakeld (OFF), maar het resultaat kan afwijken als een andere gebruiker de tabel wijzigt.

U kunt het totaal aantal records in een tabel ook bepalen met de functie RECCOUNT( ). In tegenstelling tot COUNT kunt u bij RECCOUNT( ) echter geen voorwaarden opgeven om alleen bepaalde records te tellen.

Voorbeeld

In het volgende voorbeeld wordt COUNT gebruikt om het aantal records te bepalen voor bedrijven in de regio BN waarvoor in de tabel Bedrijf een waarde is ingevuld in het veld VerkTotnu. Dit aantal wordt gebruikt om een gemiddeld verkoopcijfer te bepalen voor de betreffende bedrijven:

```
SET TALK OFF
CLEAR
USE Bedrijf
Voorwaarde = "Regio = 'BN'"
CALCULATE SUM( VerkTotnu) TO Verkoop FOR &Voorwaarde
COUNT TO Aant FOR &Voorwaarde .AND. VerkTotNu<>0
? "Gemiddelde verkoop in regio BN: "+LTRIM(STR(Verkoop/Aant))
CLOSE ALL
```

Zie ook

AVERAGE, CALCULATE, RECCOUNT( ), SUM, TOTAL

CREATE — Tabellen

Opent Tabelontwerp waarin u interactief een tabel kunt maken of wijzigen.

Syntaxis

CREATE
[<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]
[[TYPE] PARADOX | DBASE]

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>

De naam van de tabel die u wilt maken. CREATE ? en CREATE | <*bestandsnaamfilter*> tonen een dialoogvenster waarin u de naam voor de nieuwe tabel kunt opgeven. Als u een tabelnaam zonder pad opgeeft, wordt de tabel opgeslagen in de huidige directory op het huidige station. Als u een tabelnaam zonder extensie opgeeft, wordt de extensie .DBF gebruikt of de extensie voor het bestandstype dat is ingesteld met SET DBTYPE. Als u geen naam opgeeft, blijft de tabel naamloos tot u het bestand opslaat. Als u een bestaande tabelnaam opgeeft, wordt gevraagd of u de bestaande tabel wilt wijzigen of overschrijven.

U kunt ook een tabel maken in een database (gedefinieerd met de IDAPI-configuratieprogramma) door voor de naam van de tabel de naam van de database op te geven (tussen dubbele punten), zoals *:databasenaam:tabelnaam*. Als de database nog niet open is, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

[TYPE] PARADOX | DBASE

Geeft aan welk type tabel moet worden gemaakt. Het sleutelwoord TYPE is er alleen voor de leesbaarheid; het heeft geen gevolgen voor de werking van het commando.

Als u PARADOX opgeeft, wordt een Paradox-tabel gemaakt met de extensie .DB.

Als u DBASE opgeeft, wordt een dBASE-tabel gemaakt (de standaardinstelling). Als u geen extensie opgeeft voor *<bestandsnaam>*, wordt de extensie .DBF gebruikt.

Beschrijving

CREATE opent Tabelontwerp, een interactieve omgeving waarin u de structuur van een tabel kunt maken of wijzigen. Het type van de tabel die u maakt of wijzigt, wordt bepaald door het tabeltype dat u hebt opgegeven bij het commando CREATE of bij SET DBTYPE.

U maakt de tabel door voor elk veld de naam, het type en de lengte op te geven. Raadpleeg het *Handboek* voor meer informatie over het werken met Tabelontwerp.

Voorbeeld

In de volgende voorbeelden worden enkele toepassingen van CREATE in het commandovenster weergegeven:

```
CREATE DagLijst   && Tabelontwerp openen -.DBF-tabel
CREATE DagLijst TYPE PARADOX
                  && Tabelontwerp openen - .DB-tabel
CREATE ?          && Dialoogvenster voor benoemen bestand openen
```

Zie ook

APPEND, APPEND MEMO, COPY STRUCTURE, DISPLAY STRUCTURE, LIST STRUCTURE, MODIFY STRUCTURE, REPLACE

CREATE APPLICATION

Formulieren

Opent Formulierontwerp waarin u interactief een formulier kunt maken of wijzigen.

Syntaxis

CREATE APPLICATION
[*<bestandsnaam>* | ? | *<bestandsnaamfilter>*]

<bestandsnaam> | ? | *<bestandsnaamfilter>*

Het formulier dat u wilt maken of wijzigen. CREATE APPLICATION ? en CREATE APPLICATION *<bestandsnaamfilter>* tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, gebruikt dBASE de extensie .WFM.

Beschrijving

CREATE SCREEN, CREATE APPLICATION en CREATE FORM zijn onderling uitwisselbaar; in alle gevallen verschijnt Formulierontwerp. Zie de beschrijving van CREATE FORM voor meer informatie.

Voorbeeld

Zie het voorbeeld bij CREATE FORM. In dat voorbeeld kunt u CREATE FORM vervangen door CREATE APPLICATION.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. In dBASE IV start CREATE APPLICATION de dBASE IV-applicatiegenerator.

Zie ook

CREATE FORM, CREATE SCREEN, MODIFY APPLICATION, MODIFY FORM, MODIFY SCREEN, OPEN FORM

CREATE CATALOG

Tabellen

Maakt een catalogusbestand.

Syntaxis

CREATE CATALOG
[<*bestandsnaam*> | ? | <*bestandsnaamfilter*>

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Het catalogusbestand dat u wilt maken. CREATE CATALOG ? en CREATE CATALOG <*bestandsnaamfilter*> tonen een dialoogvenster waarin u de naam van een catalogusbestand kunt opgeven.

Beschrijving

Met CREATE CATALOG kunt u een nieuwe catalogus maken. Voor de namen van catalogusbestanden gelden dezelfde regels als voor bestandsnamen onder DOS: ze mogen uit maximaal acht tekens bestaan (zonder spaties). In dBASE voor Windows krijgen catalogusbestanden altijd de extensie .CAT.

Als u TITLE ON gebruikt wanneer u een nieuwe catalogus maken, wordt u gevraagd een uit een regel bestaande beschrijving van het catalogusbestand op te geven. De namen van de catalogusbestanden en de beschrijvingen worden opgeslagen in een hoofdcatalogus, CATALOG.CAT. De beschrijving die u voor elke catalogus opgeeft, verschijnt in het catalogusvenster als u SET CATALOG TO ? gebruikt om een catalogusnaam te kiezen.

Catalogi zijn dBASE-tabellen met een vooringestelde tabelstructuur. Als u een catalogusbestand maakt of kiest, wordt dat automatisch geopend in een eigen werkgebiedbuffer en wordt CATALOG ingeschakeld (ON). Vanaf dat moment worden alle tabellen en bijbehorende bestanden die u gebruikt, zoals index-, query-, indelings-, rapport- en etiketbestanden, en nieuwe bestanden die u maakt, toegevoegd aan de catalogus.

Catalogusstructuur

Alle catalogi hebben dezelfde structuur. In de volgende tabel worden de velden in een catalogus beschreven.

Veld	Veldnaam	Type	Lengte	Beschrijving
1	Path	Teken	70	Bevat het volledige pad naar de tabel of het bestand als dat niet in de huidige directory staat.
2	File_name	Teken	12	De naam van het bestand, inclusief de extensie.
3	Alias	Teken	8	Toegekende tabelalias. Als u geen aliasnaam toekent, wordt de tabelnaam gebruikt als aliasnaam. Voor alle overige bestandstypen blijft dit veld leeg.
4	Type	Teken	3	Dit veld bevat de standaardextensie voor dit type bestand. Zelfs als u bij het maken van het bestand een afwijkende extensie hebt opgegeven, verschijnt de standaardextensie in dit veld. De opgegeven extensie verschijnt echter in het veld Bestand_naam.
5	Title	Teken	80	Dit is een optioneel veld. Als SET TITLE is ingeschakeld (ON), wordt u gevraagd een beschrijving van maximaal 80 tekens op te geven voor de catalogus die u maakt.
6	Code	Numeriek	3	Als u een catalogus hebt geopend, bevat dit veld het nummer dat aan elke geopende tabel is toegekend. Programmabestanden krijgen de waarde 0. Een tabel krijgt een nummer op het moment dat die wordt gemaakt. Elke nieuwe tabel krijgt het volgende beschikbare nummer dat hoger is dan de vorige tabel. Bestanden die bij een tabel horen, zoals index-, indelings-, etiket-, query-, rapport-, scherm- en weergavebestanden krijgen hetzelfde codenummer als de tabel waarnaar ze verwijzen.
7	Tag	Teken	4	Dit veld wordt op dit moment niet gebruikt.

Met uitzondering van de velden Titel en Label wordt de inhoud van deze velden automatisch ingevuld op het moment dat een nieuw bestand aan de catalogus wordt toegevoegd.

Als u de inhoud van de catalogustabel wilt wijzigen, moet u die eerst openen in een toegankelijk werkgebied en vervolgens EDIT of BROWSE gebruiken. Zie SET CATALOG voor meer informatie.

Voorbeeld

In het volgende voorbeeld wordt CREATE CATALOG gebruikt om een catalogus voor uw bestanden te maken.

```
CREATE CATALOG Post  && Catalogus met de naam Post;
                        maken. U wordt gevraagd;
                        een beschrijving van deze;
                        catalogus op te geven
CREATE CATALOG ?     && Dialoogvenster;
                        "Catalogus openen" tonen
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CATALOG( ), SELECT( ), SET( ), SET CATALOG, SET TITLE, USE

CREATE COMMAND

Programma's

Toont het opgegeven bestand zodat u het kunt bewerken, of toont een leeg bewerkingsvenster.

Syntaxis

CREATE COMMAND
  [<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]
  [WINDOW <*vensternaam*>]

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Het bestand dat moet worden getoond om te bewerken. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, gebruikt dBASE de extensie .PRG. Als u CREATE COMMAND zonder optie gebruikt, wordt een leeg bewerkingsvenster geopend.

[WINDOW <*vensternaam*>]

Opgenomen vanwege compatibiliteit met dBASE IV. Toont het bestand in <*vensternaam*>, dat eerder is geopend met ACTIVATE WINDOW.

Standaardinstelling

Standaard start CREATE COMMAND de interne editor van dBASE. U kunt een andere editor opgeven door de instelling van EDITOR in DBASEWIN.INI te wijzigen. U kunt de instelling interactief wijzigen met het commando SET of de parameter EDITOR rechtsreeks wijzigen in het bestand DBASEWIN.INI.

Beschrijving

Met CREATE COMMAND kunt nieuwe programmabestanden maken of bestaande programmabestanden wijzigen. Gebruik DO om programmabestanden uit te voeren.

Opmerking Programma's worden in dBASE gecompileerd voordat ze worden uitgevoerd. De gecompileerde bestanden krijgen dezelfde naam als het oorspronkelijke bestand, maar de laatste letter van de extensie wordt gewijzigd in een "O". De gecompileerde versie van SALESRPT.PRG heet bijvoorbeeld SALESRPT.PRO. Als SALESPRT.PRO al bestaat, wordt dat bestand overschreven. Vandaar dat het verstandig is in directory's met gecompileerde programma's geen bestanden op te slaan waarvan de extensie eindigt op een "O".

Als u een nieuwe programmabestand maakt, toont CREATE COMMAND een bewerkingsvenster zonder bestand. Als u een bestaand bestand opgeeft, toont CREATE COMMAND het programma in een bewerkingsvenster met de cursor aan het begin van regel 1.

Het dBASE-menu verandert als een bewerkingsvenster actief is. Zie het *Handboek* voor meer informatie over de interne tekst-editor, zoals het menu en de sneltoetsen.

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS, maar MODIFY COMMAND wordt in dBASE IV wel ondersteund.

Zie ook

DO, CREATE FILE, SET DEVELOPMENT, SET EDITOR

CREATE FILE

Stations- en bestandsfuncties

Toont het opgegeven tekstbestand zodat u het kunt wijzigen, of toont een leeg bewerkingsvenster.

Syntaxis

CREATE FILE
[<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]
[WINDOW <*vensternaam*>]

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Het tekstbestand dat moet worden getoond om te bewerken. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, gebruikt dBASE de extensie .PRG. Als u CREATE FILE zonder een optie gebruikt, wordt een leeg bewerkingsvenster getoond.

[WINDOW <*vensternaam*>]
Opgenomen vanwege compatibiliteit met dBASE IV. Toont het bestand in <*vensternaam*>, dat eerder is geopend met ACTIVATE WINDOW.

Beschrijving

Met CREATE FILE kunt nieuwe tekstbestanden maken of bestaande tekstbestanden wijzigen met de interne editor van dBASE of met de editor die is opgegeven met SET EDITOR.

CREATE FILE, CREATE COMMAND, MODIFY COMMAND en MODIFY FILE zijn onderling uitwisselbaar. Zie CREATE COMMAND voor meer informatie.

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS, maar MODIFY FILE wordt in dBASE IV wel ondersteund.

Zie ook

CREATE COMMAND, SET EDITOR

CREATE FORM — Formulieren

Opent Formulierontwerp waarin u een formulier kunt maken of wijzigen.

Syntaxis

CREATE FORM
[<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>
Het formulier dat u wilt maken of wijzigen. CREATE FORM ? en CREATE FORM <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, wordt de extensie .WFM gebruikt.

Beschrijving

Gebruik CREATE FORM om Formulierontwerp te openen en interactief een formulier te maken of wijzigen. Formulierontwerp genereert automatisch dBASE-programmacode die de inhoud en de indeling van een formulier definieert. Deze code wordt opgeslagen in een tekstbestand (.WFM-bestand) dat gewijzigd kan worden.

CREATE SCREEN, CREATE APPLICATION en CREATE FORM zijn onderling uitwisselbaar. Voor al deze commando's geldt dat de aanwezigheid van een formulierbestand bepaalt of een formulier wordt gemaakt of een formulier wordt

bewerkt. Als het .WFM-bestand bestaat, kunt u het met deze commando's wijzigen in Formulierontwerp. Als het bestand niet bestaat, zorgen deze commando's dat een nieuw bestand wordt gemaakt.

Omdat een .WFM-bestand een programmabestand is, kunt u het wijzigen met MODIFY COMMAND.

Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het gebruik van Formulierontwerp.

Opmerking Formulierontwerp is een tweeweghulpmiddel. U kunt zelfs een formulier openen in Formulierontwerp als u de code in het .WFM-bestand hebt gewijzigd.

Voorbeeld

In de volgende voorbeelden wordt het dialoogvenster **Bestand opslaan** geopend met de cursor in het blok **Bestandsnaam** zodat het nieuwe formulier kan worden benoemd. De keuzelijst met .WFM-bestanden in de huidige directory is beschikbaar als u een bestaande formuliernaam wilt gebruiken voor een nieuw formulier. Met de instructie MODIFY FORM ? daarentegen kunt u een bestaand formulier wijzigen:

```
CREATE FORM ?
CREATE FORM *.WFM
```

Als u in het commandovenster CREATE FORM zonder meer gebruikt, verschijnt een naamloos formulierontwerpvenster:

```
CREATE FORM
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CREATE APPLICATION, CREATE SCREEN, MODIFY APPLICATION, MODIFY FORM, MODIFY SCREEN, OPEN FORM

CREATE LABEL

Invoer/uitvoer

Opent Rapportontwerp waarin u een etiketbestand kunt maken of wijzigen.

Syntaxis

CREATE LABEL
[<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Het etiketbestand dat u wilt maken of wijzigen. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad

dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, gebruikt dBASE de extensie .RPL.

Als *<bestandsnaam>* niet wordt gevonden, wordt het bestand gemaakt. Standaard krijgt *<bestandsnaam>* de extensie .RPL en wordt het bestand opgeslagen in de huidige directory.

CREATE LABEL zonder een optie opent een leeg rapportontwerpvenster, zodat u een nieuw etiketbestand kunt maken.

Beschrijving

Gebruik CREATE LABEL om Rapportontwerp te openen en een etiketbestand te maken of wijzigen. Een etiketbestand bevat informatie over de indeling en opmaak van etiketten. Zie de Crystal Reports documentatie voor meer informatie over Rapportontwerp. De commando's CREATE LABEL en MODIFY LABEL zijn onderling uitwisselbaar.

Voor dat u CREATE LABEL kunt gebruiken, moet een standaardprinter zijn ingesteld. Nadat u het etiketbestand hebt gemaakt of gewijzigd, kunt u LABEL FORM gebruiken om de etiketten af te drukken.

Voorbeeld

In het volgende voorbeeld wordt een databasetabel geopend en wordt vervolgens CREATE LABEL gebruikt om een etiketformulier te maken:

```
CLOSE DATABASE
USE Bedrijf
CREATE LABEL BEDRETK1
* Als BEDRETK1.LBL al bestaat, wordt het
* geopend voor wijzigen. Zo niet, dan wordt
* nieuw etiketformulier gemaakt.
```

Overdraagbaarheid

De optie *<bestandsnaamfilter>* wordt niet ondersteund in dBASE IV of dBASE III PLUS. In dBASE IV en dBASE III PLUS is .LBL de standaardextensie van een etiketbestand.

Zie ook

CREATE REPORT, LABEL FORM

CREATE MENU

Formulieren

Opent Menu-ontwerp waar u een menubestand kunt maken of wijzigen.

Syntaxis

CREATE MENU
[*<bestandsnaam>* | ? | *<bestandsnaamfilter>*]

[<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]
De naam van het menu dat u maakt of wijzigt. CREATE MENU ? en CREATE MENU <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een bestand kunt kiezen. Geeft u een bestand zonder pad op, dan wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, gebruikt dBASE de extensie .MNU.

Beschrijving

Gebruik CREATE MENU om een menu voor een formulier te ontwerpen.

Het menu dat u ontwerpt, wordt opgeslagen in een menudefinitiebestand (.MNU). Dat bestand bevat dBASE-programmacode. Met het kenmerk Menu van een formulier voegt u het menu toe aan een formulier.

Zie Hoofdstuk 10 in het *Handboek* voor meer informatie over het venster Menu-ontwerp.

CREATE QUERY

Tabelindeling

Opent Query-ontwerp om een query te maken of wijzigen.

Syntaxis

CREATE QUERY <*bestandsnaam*> | ? | <*bestandsnaamfilter*>

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>
Geeft de naam aan van het bestand dat u wilt maken. CREATE QUERY ? en CREATE QUERY <*bestandsnaamfilter*> tonen een dialoogvenster, waarin u de naam van een van een query-bestand opgeeft. Als u een bestand zonder pad opgeeft, slaat dBASE voor Windows het bestand op in de huidige directory op de huidige schijf. Als u geen extensie opgeeft, wordt de extensie .QBE gebruikt.

Beschrijving

CREATE QUERY voert dezelfde bewerking uit als CREATE VIEW. Beide commando's openen Query-ontwerp, een interactieve omgeving waarin u een query kunt maken of wijzigen. In Query-ontwerp maakt u een .QBE-bestand dat alleen de records en velden toont die aan opgegeven voorwaarden voldoen. Met het commando SET VIEW kunt u een query-bestand activeren. Raadpleeg het *Handboek* voor meer informatie over het werken met Query-ontwerp.

Voorbeeld

CREATE QUERY en CREATE VIEW zonder enige argumenten openen beide Query-ontwerp. In dat venster wordt een naamloze query gegenereerd. De gebruiker kan de query later een naam geven:

```
CREATE QUERY
```

CREATE QUERY gevolgd door een bestandsnaam maakt een bestand met die naam als het bestand wordt opgeslagen:

```
CREATE QUERY Mynquery
```

Als MYNQUERY.QBE al bestaat, wordt gevraagd of u het wilt overschrijven.

Zie ook

MODIFY, SET VIEW

CREATE REPORT

Invoer/uitvoer

Opent Rapportontwerp, waarin u een rapportbestand kunt maken of wijzigen.

Syntaxis

CREATE REPORT
[CROSSTAB]
[<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]

CROSSTAB

Opent Rapportontwerp waarin het dialoogvenster **Kruistabulatie** wordt getoond.

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Het rapportbestand dat u wilt maken of wijzigen. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH.

Als u een bestand zonder extensie opgeeft, wordt .RPT gebruikt als u niet CROSSTAB hebt opgegeven, en .RPC als u dat wel hebt gedaan. Als <*bestandsnaam*> niet wordt gevonden, wordt een bestand gemaakt met de toepasselijke extensie. Dat bestand wordt opgeslagen in de huidige directory.

CREATE REPORT zonder enige argumenten opent een leeg rapportontwerpvenster, zodat u een nieuw rapport kunt maken.

Beschrijving

Met CREATE REPORT opent u Rapportontwerp waarin u een rapportbestand kunt maken of wijzigen. Een rapportbestand bevat informatie over de opmaak van een rapport. Zie de Crystal Reports documentatie voor meer informatie over Rapportontwerp. De commando's CREATE REPORT en MODIFY REPORT zijn onderling uitwisselbaar.

Voor dat u CREATE REPORT kunt gebruiken, moet een standaardprinter zijn ingesteld. Nadat u het etiketbestand hebt gemaakt of gewijzigd, kunt u LABEL FORM gebruiken om de etiketten af te drukken.

Voorbeeld

In het volgende voorbeeld wordt een databasetabel geopend en vervolgens wordt het commando CREATE REPORT gebruikt om een rapportformulier te maken:

```
CLOSE DATABASE
USE Bedrijf
CREATE REPORT Bedrrap1
* Als Bedrrap1 al bestaat, wordt het
* geopend voor wijzigen. Zo niet, dan wordt
* nieuw rapportformulier gemaakt.
```

Overdraagbaarheid

De optie *<bestandsnaamfilter>* wordt niet ondersteund in dBASE IV of dBASE III PLUS. In dBASE IV en dBASE III PLUS heeft een rapportbestand standaard de extensie .FRM.

Zie ook

REPORT FORM

CREATE SCREEN

Formulieren

Opent Formulierontwerp waarin u een formulier kunt maken of wijzigen.

Syntaxis

CREATE SCREEN
[*<bestandsnaam>* | ? | *<bestandsnaamfilter>*]

<bestandsnaam> | ? | *<bestandsnaamfilter>*

Het formulier dat u wilt maken of wijzigen. De opties CREATE SCREEN ? CREATE SCREEN *<bestandsnaamfilter>* tonen het dialoogvenster **Bestand opslaan** waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, gebruikt dBASE de extensie .WFM.

Beschrijving

CREATE SCREEN, CREATE APPLICATION en CREATE FORM zijn onderling uitwisselbaar. Alle drie openen zij Formulierontwerp. Zie de beschrijving van CREATE FORM voor meer informatie.

Voorbeeld

Zie het voorbeeld bij CREATE FORM. In dat voorbeeld kunt u CREATE FORM vervangen door CREATE SCREEN.

Overdraagbaarheid

In dBASE III PLUS en dBASE IV opent CREATE SCREEN Formulierontwerp van respectievelijk dBASE III PLUS of dBASE IV. Hier worden twee bestanden gemaakt, .SCR- en .FMT-bestanden. U kunt .FMT-bestanden gebruiken in dBASE voor Windows, maar u kunt ze niet wijzigen met Formulierontwerp van dBASE voor Windows.

Zie ook

CREATE APPLICATION, CREATE FORM, MODIFY APPLICATION, MODIFY FORM, MODIFY SCREEN, OPEN FORM

CREATE SESSION — Omgeving

Maakt een nieuwe *sessie* en maakt die nieuwe sessie tevens actief. Alle volgende commando's die van toepassing zijn op de huidige sessie, vinden hier plaats.

Syntaxis

CREATE SESSION

Beschrijving

Gebruik CREATE SESSION om een schone werksessie binnen dBASE te maken. Een sessie kan worden vergeleken met een multi-user-omgeving. Elke sessie beheert zijn eigen verzameling werkgebieden. Het instellen van een relatie tussen een formulier of applicatie en een sessie zorgt dat geen ander formulier of andere applicatie invloed kan uitoefenen op de omgeving in de sessie. Als een applicatie in een andere sessie bijvoorbeeld het commando CLOSE ALL geeft, is dat niet van invloed op de bestanden in de huidige sessie.

CREATE SESSION voert de volgende stappen uit:

- Alle werkgebieden worden toegankelijk gemaakt, zelfs als andere programma's in uitvoering tabellen hebben geopend. Net als in een multi-user-omgeving kan dezelfde tabel in verschillende sessies zijn geopend, maar de handelingen van recordaanwijzer zijn in elke sessie onafhankelijk van andere sessies. Als een tabel echter in een sessie wordt bijgewerkt, worden die wijzigingen weergegeven in dezelfde tabel die is geopend in een andere sessie.
- Voor de waarden van de meeste SET-commando's wordt de standaardinstelling hersteld (verderop in deze sectie vindt u een volledige lijst).

Opmerking CREATE SESSION oefent niet rechtstreeks invloed uit op de toegang tot geheugenvariabelen. De toegang tot variabelen wordt bepaald door hun bereik (ingesteld met PUBLIC, PRIVATE, LOCAL of STATIC) en niet door de sessie waar ze zijn gedefinieerd.

In een applicatie zou u op de volgende plaatsen CREATE SESSION kunnen toepassen:

- Aan het begin van een programma of procedure waarvoor een standaardomgeving vereist is of als het programma of de procedure moet worden beschermd tegen commando's die in een andere sessie worden gegeven.
- Aan het begin van het eerste programma in een reeks met elkaar samenhangende programma's. Op die manier kunt u zorgen dat het programma wordt gestart in een bekende omgeving en is beschermd tegen handelingen in andere sessies.
- Aan het begin van code (.PRG of .WFM) die een formulier definieert dat op zijn beurt is gekoppeld aan een tabel. Op die manier zorgt u dat het formulier toegang heeft tot de tabel, zelfs als die tabel in een andere sessie wordt gebruikt.
- Voordat u een tabel opent in een blader- of bewerkingsvenster. Op die manier kunt u filters, indexen en andere voorwaarden instellen die van belang zijn voor de huidige weergave, zelfs als de tabel in een andere sessie wordt gebruikt.
- Voordat u een reeks transacties start met BEGINTRANS( ). Dat garandeert dat de transactie kan worden gestart, zelfs als in een andere sessie een andere BEGINTRANS( ) actief is.

U kunt geen commando geven om een bepaalde sessie te kiezen of te beëindigen. U kunt op de volgende manieren een sessie kiezen:

- Als u CREATE SESSION gebruikt, wordt die sessie gekozen.
- Als een formulier wordt geactiveerd, wordt de sessie gekozen die actief was toen het formulier werd gemaakt.
- Als u een blader- of een query-venster kiest, wordt de sessie gekozen die actief was op het moment u het desbetreffende venster opende.

Een sessie wordt automatisch beëindigd als alle vensters en formulieren die bij de sessie horen, ophouden te bestaan.

De volgende SET-commando's zijn alleen van invloed op de huidige sessie:

SET AUTOSAVE	SET BLOCKSIZE	SET CARRY
SET CENTURY	SET CONFIRM	SET CUAENTER
SET CURRENCY	SET DATABASE	SET DATE
SET DBTYPE	SET DECIMALS	SET DEFAULT
SET DELETED	SET DELIMITERS	SET DIRECTORY
SET EXACT	SET EXCLUSIVE	SET FIELDS
SET FILTER	SET IBLOCK	SET INDEX
SET KEY TO	SET LOCK	SET MARK
SET MBLOCK	SET MEMOWIDTH	SET NEAR
SET ORDER	SET PATH	SET POINT
SET PRECISION	SET REFRESH	SET RELATION
SET REPROCESS	SET SAFETY	SET SEPARATOR
SET SKIP	SET TALK	SET UNIQUE

Zie Hoofdstuk 21 in *Programmeren* voor meer informatie over sessies.

Voorbeeld

In het volgende voorbeeld worden in twee verschillende sessies twee bladervensters geopend met verschillende indexen:

```
SET EXCLUSIVE OFF
CLOSE ALL
USE Bedrijf ORDER Bedrijf
BROWSE
CREATE SESSION
USE Bedrijf ORDER CompCode

BROWSE
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DO, DEFINE, LOCAL, PRIVATE, PUBLIC, STATIC

CREATE VIEW — Tabelindeling

Opent Query-ontwerp om een query-bestand te maken of wijzigen.

Syntaxis

CREATE VIEW *<bestandsnaam>* | ? | *<bestandsnaamfilter>*

<bestandsnaam> | ? | *<bestandsnaamfilter>*

Geeft de naam aan van het bestand dat u wilt maken. CREATE VIEW ? en CREATE VIEW *<bestandsnaamfilter>* tonen een dialoogvenster, waarin u de naam van het query-bestand kunt opgeven. Als u een bestand zonder pad opgeeft, slaat dBASE voor Windows het bestand op in de huidige directory op de huidige schijf. Als u een bestandsnaam zonder extensie opgeeft, wordt de extensie .QBE gebruikt.

Beschrijving

CREATE VIEW voert dezelfde bewerking uit als CREATE QUERY. Beide commando's openen Query-ontwerp, een interactieve omgeving waarin u een query kunt maken of wijzigen. In Query-ontwerp maakt u een .QBE-bestand dat alleen die records en velden toont die voldoen aan opgegeven voorwaarden. Met het commando SET VIEW kunt u een query-bestand activeren. Raadpleeg het *Handboek* voor meer informatie over het werken met Query-ontwerp.

Voorbeeld

CREATE VIEW en CREATE QUERY zonder enige argumenten openen beide Query-ontwerp. In dat venster wordt een naamloze query gegenereerd. De gebruiker kan de query later een naam geven:

```
CREATE VIEW
```

CREATE VIEW gevolgd door een bestandsnaam maakt een bestand met die naam als het bestand wordt opgeslagen:

```
CREATE VIEW Mynquery
```

Als MYNQUERY.QBE al bestaat, wordt u gevraagd of u het wilt overschrijven.

Na CREATE VIEW wordt het commando CLOSE DATABASES gegeven.

Zie ook

SET VIEW, CREATE VIEW...FROM ENVIRONMENT

CREATE VIEW...FROM ENVIRONMENT

Tabelindeling

Maakt een met dBASE III PLUS compatibel weergavebestand op basis van de huidige werkomgeving.

Syntaxis

CREATE VIEW *<bestandsnaam>* FROM ENVIRONMENT

<bestandsnaam>

Het weergavebestand met de gegevens over de huidige werkomgeving. Standaard wordt aan *<bestandsnaam>* de extensie .VUE toegevoegd en wordt het bestand opgeslagen in de huidige directory.

Beschrijving

Met dit commando wordt de huidige werkomgeving opgeslagen in een weergavebestand (.VUE). Het weergavebestand kan later worden geopend met het commando SET VIEW TO *<bestandsnaam>*. De opgeslagen instellingen worden dan hersteld. De huidige werkomgeving omvat alle geopende tabellen en indexbestanden, plus hun werkgebiednummers, alle relaties, de actieve veldenlijst, filtervoorwaarden en alle geopende indelingsbestanden. De instellingen van de opties SET RELATION...INTEGRITY en SET KEY maken hier echter geen deel van uit.

Voorbeeld

In het volgende voorbeeld worden twee tabellen geopend en een relatie, filters en veldinstructies ingesteld. Vervolgens wordt met CREATE VIEW een .VUE-bestand gemaakt. Het bestand WERKOMG.VUE bevat al deze instructies. De omgeving kan worden hersteld met SET VIEW TO:

```
CLOSE DATABASE
USE Bedrijf EXCLUSIVE
SELECT 2
USE CONTACT EXCLUSIVE
INDEX ON CompCode TAG CompCode
SET FIELDS TO Contact
SELECT Bedrijf
SET RELATION TO CompCode INTO Contact
SET FIELDS TO Bedrijf, Regio
SET FILTER TO Regio="BN"
CREATE VIEW Werkomg FROM ENVIRONMENT
```

Deze hele reeks instructies wordt opnieuw uitgevoerd als de gebruiker het volgende commando geeft:

```
SET VIEW TO Werkomg
```

Zie ook

CREATE QUERY, CREATE VIEW, SET FIELDS, SET FILTER, SET FORMAT, SET INDEX, SET RELATION, SET VIEW, USE

CREATE...FROM

Tabellen

Maakt een tabel met de structuur die is gedefinieerd met het commando COPY TO...STRUCTURE EXTENDED of het commando CREATE...STRUCTURE EXTENDED.

Syntaxis

CREATE *<bestandsnaam1>* | ? | *<bestandsnaamfilter1>*
[[TYPE] PARADOX | DBASE]
FROM *<bestandsnaam2>* | ? | *<bestandsnaamfilter2>*
[[TYPE] PARADOX | DBASE]

<bestandsnaam1> | ? | *<bestandsnaamfilter1>*

De naam van de tabel die u wilt maken. CREATE ? en CREATE *<bestandsnaamfilter>* tonen een dialoogvenster, waarin u de naam van de doeltabel kunt opgeven. Als u een tabelnaam zonder pad opgeeft, wordt de tabel opgeslagen in de huidige directory op het huidige station. Als u een tabelnaam zonder extensie opgeeft, een standaardtabeltype definieert met SET DBTYPE of een van de TYPE-opties gebruikt, gebruikt dBASE voor Windows de extensie .DBF. Als u geen tabelnaam opgeeft, blijft de tabel naamloos tot u die opslaat. Als u een bestaande tabelnaam opgeeft, toont dBASE voor Windows een dialoogvenster, waarin u kunt aangeven of u de bestaande tabel wilt wijzigen.

U kunt ook een tabel maken in een database (gedefinieerd met de IDAPI-configuratieprogramma) door voor de naam van de tabel de naam van de database op te geven (tussen dubbele punten), zoals *:databasenaam:tabelnaam*. Als de database nog niet open is, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een

aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

[TYPE] PARADOX | DBASE

Geeft aan welk type tabel moet worden gemaakt. Het sleutelwoord TYPE is er alleen voor de leesbaarheid; het heeft geen gevolgen voor de werking van het commando.

Als u PARADOX opgeeft, wordt een Paradox-tabel met de extensie .DB gemaakt.

Als u DBASE opgeeft, wordt een dBASE-tabel gemaakt (de standaardinstelling). Als u geen extensie opgeeft voor *<bestandsnaam>*, gebruikt dBASE voor Windows de extensie .DBF.

FROM *<bestandsnaam2>* | ? | *<bestandsnaamfilter2>* [TYPE] PARADOX | DBASE

Geeft aan van welke tabel u de structuur wilt opslaan in een nieuwe tabel. FROM ? en FROM *<bestandsnaamfilter>* tonen een dialoogvenster waarin u de naam van een bestaande tabel kunt kiezen. Als u een tabel opgeeft zonder een pad, wordt de tabel gezocht in de huidige directory en vervolgens in het pad dat u opgeeft met SET PATH. De opties Type en Database zijn gelijk aan die zijn beschreven in de vorige alinea.

U kunt ook de structuur kopiëren van een tabel in een database (gedefinieerd met het IDAPI-configuratieprogramma) door voor de naam van de tabel de naam van de database op te geven (tussen dubbele punten) zoals *:databasenaam:tabelnaam*. Als de database nog niet open is, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

Beschrijving

Het commando CREATE...FROM wordt in een programma het meest gebruikt in combinatie met het commando COPY TO...STRUCTURE EXTENDED om een nieuwe tabel te maken op basis van de structuur in een andere tabel inplaats van interactief met de commando's CREATE of MODIFY STRUCTURE. U doet dat als volgt:

1. Gebruik COPY TO...STRUCTURE EXTENDED om een tabel te maken met records die informatie bevatten over elk veld in de oorspronkelijke tabel.
2. U kunt eventueel de structuurinformatie in de nieuwe tabel aanpassen met elk commando waarmee u gegevens in tabellen kunt manipuleren, zoals REPLACE.
3. Gebruik CREATE...FROM om een nieuwe tabel maken op basis van de structuurinformatie in het structuurbestand. Nadat u CREATE...FROM hebt gebruikt, is de nieuwe tabel actief.

De tabel die met CREATE...FROM is gemaakt, wordt de huidige tabel in het actieve werkgebied. Als CREATE...FROM om de een of andere reden niet in staat is de tabel te maken, blijft in het huidige werkgebied geen tabel geopend.

Als indexvlaggen zijn ingesteld voor velden in de tabel die is gemaakt met COPY TO...STRUCTURE EXTENDED, levert CREATE...FROM ook een produktie-MDX-bestand met de opgegeven indexlabels.

Voorbeeld

In het volgende voorbeeld wordt COPY TO ... STRUCTURE EXTENDED gebruikt om met de structuur van de tabel Klanten een tijdelijke tabel te maken die kan worden gewijzigd. Vervolgens wordt CREATE...FROM gebruikt om een nieuwe tabel met de aangepaste structuur te maken:

```
SET SAFETY OFF
USE Klanten
COPY TO Klanten2 STRUCTURE EXTENDED
* Klanten2.DBF bevat nu records die de
* structuur van Klanten definiëren.
USE Klanten2 EXCLUSIVE
REPLACE Field_Name WITH "CtDatum" FOR RECNO() =9
CREATE NwKlant FROM Klanten2
APPEND
CLOSE ALL
```

Zie COPY TO ... STRUCTURE EXTENDED voor een ander voorbeeld met CREATE ... FROM.

Zie ook

COPY STRUCTURE, COPY TO...STRUCTURE EXTENDED, CREATE, DISPLAY STRUCTURE, LIST STRUCTURE, MODIFY STRUCTURE, CREATE...STRUCTURE EXTENDED

CREATE...STRUCTURE EXTENDED

Tabellen

Maakt en opent een tabel die u kunt gebruiken om de structuur van een nieuwe tabel te ontwerpen.

Syntaxis

CREATE *<tabelnaam>* | ?
  STRUCTURE EXTENDED
  [[TYPE] PARADOX | DBASE]

<tabelnaam> | ?

De naam van de tabel die u wilt maken. CREATE ? STRUCTURE EXTENDED toont een dialoogvenster, waarin u de naam van de doeltabel kunt opgeven. Als u een tabel zonder pad opgeeft, wordt de tabel opgeslagen in de huidige directory op het huidige station. Als u een tabelnaam zonder extensie opgeeft, een standaardtabeltype definieert met SET DBTYPE of een van de TYPE-opties gebruikt, gebruikt dBASE voor Windows de extensie .DBF.

U kunt ook een tabel maken in een database (gedefinieerd met de IDAPI-configuratieprogramma) door voor de naam van de tabel de naam van de database op te geven (tussen dubbele punten), zoals *:databasenaam:tabelnaam*. Als de database nog niet open is, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een

aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

[TYPE] PARADOX | DBASE

Geeft aan welk type tabel moet worden gemaakt. Het sleutelwoord TYPE is er alleen voor de leesbaarheid; het heeft geen gevolgen voor de werking van het commando.

Als u PARADOX opgeeft, wordt een Paradox-tabel gemaakt met de extensie .DB.

Als u DBASE opgeeft, wordt een dBASE-tabel gemaakt (de standaardinstelling). Als u geen extensie opgeeft voor *<bestandsnaam>*, gebruikt dBASE voor Windows de extensie .DBF.

Beschrijving

CREATE...STRUCTURE EXTENDED maakt een lege tabel, een zogenaamde *structuurtabel* met vijf velden met een vaste naam, lengte en type. De velden komen overeen met kenmerken die de velden beschrijven in de tabel die u wilt maken:

Veld	Inhoud
FIELD_NAME	Tekenveld dat de naam van het veld bevat.
FIELD_TYPE	Tekenveld dat het gegevenstype van het veld bevat.
FIELD_LEN	Numeriek veld dat de lengte van het veld bevat.
FIELD_DEC	Numeriek veld dat het aantal decimalen voor numerieke en zwevende gegevens bevat.
FIELD_IDX	Tekenveld (wordt in dBASE III PLUS niet gemaakt) dat aangeeft of voor bepaalde velden indexlabels zijn gemaakt toen de huidige tabel werd gemaakt.

CREATE...STRUCTURE EXTENDED en COPY TO...STRUCTURE EXTENDED zijn overeenkomstige commando's. COPY TO...STRUCTURE EXTENDED maakt echter een tabel met records die informatie verschaffen over de velden in de huidige tabel, terwijl CREATE...STRUCTURE een lege tabel maakt. Nadat u met CREATE...STRUCTURE EXTENDED een nieuwe tabel hebt gemaakt, kunt u records toevoegen om de structuur van de nieuwe tabel te definiëren. Vervolgens gebruikt u het commando CREATE...FROM om een nieuwe tabel te maken op basis van de velddefinities in de structuurtabel.

Voorbeeld

In het volgende voorbeeld wordt CREATE ... STRUCTURE EXTENDED gebruikt om een tabel te maken met velden voor tabelontwerpwaarden. Nadat een leeg bestand is gemaakt, worden lege records toegevoegd en worden de velden gevuld met structuurwaarden. DBBasis wordt tenslotte gebruikt om een nieuwe dBASE-tabel met de naam namen te maken:

```
SET SAFETY OFF
CLOSE ALL
CLEAR ALL
CREATE DBBasis STRUCTURE EXTENDED
APPEND BLANK
REPLACE Field_Name WITH "VNAAM"
REPLACE Field_Type WITH "C"
```

```
REPLACE Field_Len  WITH 15
REPLACE Field_IDX  WITH "N"
APPEND BLANK
REPLACE Field_Name WITH "ANAAM"
REPLACE Field_Type WITH "C"
REPLACE Field_Len  WITH 15
REPLACE Field_IDX  WITH "N"
APPEND BLANK
REPLACE Field_Name WITH "ADRES"
REPLACE Field_Type WITH "C"
REPLACE Field_Len  WITH 20
REPLACE Field_IDX  WITH "N"
CREATE TempNaam FROM DBBasis TYPE DBASE
APPEND
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

COPY TO...STRUCTURE EXTENDED, CREATE, CREATE...FROM

CTOD( )

Uitdrukkingen en gegevenstypeconversie

Geeft een datumuitdrukking als resultaat voor een opgegeven tekenuitdrukking.

Syntaxis

CTOD(<*Tuitdr*>)

<Tuitdr>

De tekenuitdrukking, in de huidige datumopmaak, die moet worden omgezet in een datumuitdrukking.

Beschrijving

Gebruik CTOD( ) om een tekenuitdrukking om te zetten in een datumuitdrukking. Nadat u tekengegevens hebt omgezet in datumgegevens, kunt u die manipuleren met datumfuncties.

U moet <*Tuitdr*> opgeven in de huidige datumopmaak die wordt bepaald door SET DATE, DBASEWIN.INI of het Configuratiescherm van Windows (in die volgorde). Dat wil zeggen, de instellingen bij SET DATE hebben een hogere prioriteit dan die in DBASEWIN.INI, en de instellingen in DBASEWIN.INI hebben weer een hogere prioriteit dan de instellingen in het Configuratiescherm van Windows. Let er op dat <*Duitdr*> overeenkomt met de datumopmaak die wordt gebruikt op het moment dat het programma wordt uitgevoerd.

CTOD( ) geeft een datum in de huidige datumopmaak als resultaat.

Als u een ongeldige datum doorgeeft aan CTOD( ), wordt die eerst omgezet in een geldige datum en vervolgens wordt die als resultaat gegeven in de vorm van een datumuitdrukking. Als u een lege tekenreeks of een tekenreeks die geen datum bevat doorgeeft aan CTOD( ), wordt een lege datumuitdrukking in de huidige opmaak als resultaat gegeven. Als bij SET DATE bijvoorbeeld AMERICAN is ingesteld, geeft CTOD("") / / als resultaat.

Voor het scheidingsteken in <*Tuitdr*> mag u elk teken behalve een cijfer gebruiken. U mag voor <*Tuitdr*> dus een reeks opgeven als "DD*MM*JJ" of "DD!MM!JJ". In het resultaat wordt het standaardscheidingsteken gebruikt.

Een datum voor het jaar 0 kunt opgeven door achter de datum gevolgd door een spatie en "bc" in kleine letters of hoofdletters te typen, zoals CTOD("4/4/92 BC").

Voorbeeld

In de volgende voorbeelden wordt CTOD( ) gebruikt om enkele tekenuitdrukkingen om te zetten in datumuitdrukkingen:

```
SET DATE AMERICAN
? CTOD("4/1/94")            && Geeft 04/01/94 als resultaat
? CTOD("1/14/94")           && Geeft 01/14/94 als resultaat
? CTOD("5/2/94")            && Geeft 05/02/94 als resultaat
? CTOD("00/00/00")          && Geeft   /  /   als resultaat
? CTOD("")                  && Geeft   /  /   als resultaat
? CTOD("X/X/X")             && Geeft   /  /   als resultaat
? CTOD(4/1/94)              && Geeft een fout als resultaat
x = "4/1/94"
? CTOD(x)                   && Geeft 04/01/94 als resultaat
? CTOD("1/1/91 BC")         && Geeft 01/01/91 BC als resultaat
```

Zie ook

DTOC( ), DTOS( ), SET DATE, SET CENTURY, SET MARK

DATABASE( )

Tabellen

Geeft als resultaat de naam van de huidige database waaruit tabellen worden gebruikt.

Syntaxis

DATABASE( )

Beschrijving

DATABASE( ) geeft de naam van de huidige database als resultaat. De huidige database is ingesteld met het commando SET DATABASE. (Databases worden gedefinieerd met het IDAPI-configuratieprogramma. Raadpleeg *Aan de slag* voor meer informatie.) Als geen database is geopend, geeft de functie DATABASE( ) een lege tekenreeks ("") als resultaat.

Voorbeeld

In het volgende voorbeeld worden OPEN DATABASE en SET DATABASE TO gebruikt om een verbinding tot stand te brengen met databases op een database-server en DATABASE( ) om te bepalen welke database is geopend:

```
CLEAR
SET DBTYPE TO              && Standaardinstelling is DBASE
OPEN DATABASE CAKlant LOGIN gast/gast
                           && Verbinding tot stand brengen met database-server;
                              en gebruikersnaam / wachtwoord opgeven
OPEN DATABASE FLKlant LOGIN gast/gast
                           && Verbinding tot stand brengen met database-server;
                              en gebruikersnaam / wachtwoord opgeven
SET DATABASE TO CAKlant    && Database actief maken
USE admin.belgie
? DATABASE()               && Geeft CAKLANT als resultaat
? ALIAS()                  && Geeft 1 als resultaat
? DBF()                    && Geeft ADMIN.BELGIE als resultaat
? WORKAREA()               && Geeft 1 als resultaat
SET DATABASE TO FLClients && Database actief maken
? DATABASE()               && Geeft FLKLANT als resultaat
CLOSE DATABASES CAKlant, FLKlant ;
                           && Verbinding met;
                              database-server verbreken
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CLOSE..., OPEN DATABASE, SET DATABASE, SET DBTYPE

DATE( )

Datum- en tijdgegevens

Geeft de systeemdatum als resultaat.

Syntaxis

DATE( )

Beschrijving

DATE( ) geeft een datumuitdrukking met de huidige systeemdatum als resultaat.

De huidige datumopmaak wordt bepaald door de instelling voor SET DATE, DBASEWIN.INI of de optie **Internationaal** in het Configuratiescherm van Windows (in die volgorde). Dat wil zeggen, de instellingen bij SET DATE hebben een hogere prioriteit dan die in DBASEWIN.INI, en de instellingen in DBASEWIN.INI hebben weer een hogere prioriteit dan de instellingen in het Configuratiescherm van Windows. DATE( ) geeft de datum in de huidige datumopmaak als resultaat.

Als SET CENTURY is ingeschakeld (ON), geeft DATE( ) een datum met een jaartal van vier cijfers als resultaat.

De systeemdatum wijzigt u met SET DATE TO. Met DTOC( ) of DTOS( ) kunt u de datumuitdrukking omzetten in een tekenreeks.

Voorbeeld

In het volgende voorbeeld wordt DATE( ) gebruikt om het aantal dagen tussen de inhoud van een datumveld en de huidige systeemdatum en een datum 30 dagen in de toekomst te bereken.

```
USE Afnemers
? DTOC(BalnsDatum) + ", " + ;
  LTRIM(STR(DATE()  - BalnsDatum)) + ;
  " dagen geleden is de balans voor deze account" + ;
  " berekend."
CLOSE DATABASE
?
?  "Uw rekening dient over dertig (30) dagen betaald te zijn - "
?? DATE()  + 30
```

Zie ook

DTOC( ), DTOS( ), SET CENTURY, SET DATE, SET DATE TO, SET MARK

DAY( )

Datum- en tijdgegevens

Geeft als resultaat de numerieke waarde van de dag in de maand voor de opgegeven datumuitdrukking.

Syntaxis

DAY(<*Duitdr*>)

<Duitdr>

De datumuitdrukking waarvan u het nummer van de dag in de maand als resultaat wilt geven.

Beschrijving

DAY( ) geeft het nummer van de dag in de maand als resultaat: een getal van 1 tot 31.

Geef <*Duitdr*> op in de huidige datumopmaak die wordt bepaald door SET DATE, DBASEWIN.INI of de optie Internationaal in het Configuratiescherm van Windows (in die volgorde). Dat wil zeggen, de instellingen bij SET DATE hebben een hogere prioriteit dan die in DBASEWIN.INI, en de instellingen in DBASEWIN.INI hebben weer een hogere prioriteit dan de instellingen in het Configuratiescherm van Windows. Let er op dat <*Duitdr*> overeenkomt met de datumopmaak die wordt gebruikt op het moment dat het programma wordt uitgevoerd.

Als u aan DAY( ) een ongeldige datum doorgeeft, wordt die eerst omgezet in een geldige datum en vervolgens wordt het nummer van de dag in de maand als resultaat gegeven. Als u een lege datumuitdrukking of een andere dan een datumuitdrukking tussen accolades ({ }) doorgeeft aan DAY( ), wordt 0 als resultaat gegeven. Als u een andere uitdrukking dan een datumuitdrukking of een uitdrukking die niet tussen accolades staat, doorgeeft aan DAY( ), wordt een fout als resultaat gegeven.

Voorbeeld

In de volgende voorbeelden wordt DAY( ) gebruikt om het nummer van de dag in de maand als resultaat te geven. De functie krijgt gegevens van het type datum doorgegeven.

```
SET DATE AMERICAN
? DAY({1/4/94})              && Geeft 1 als resultaat
? DAY({32/4/93})             && Geeft 2 als resultaat want;
                                April heeft 30 dagen
? DAY({00/00/00})            && Geeft 0 als resultaat
? DAY({})                    && Geeft 0 als resultaat
? DAY(X)                     && Geeft foutmelding als resultaat
? DAY(1/4/94)                && Geeft foutmelding als resultaat
X = {1/4/94}
? DAY(X)                     && Geeft 1 als resultaat
```

Zie CMONTH( ) en CDOW( ) voor meer voorbeelden met DAY( ).

Zie ook

DOW( ), MONTH( ), SET CENTURY, SET DATE

DBERROR( )

Foutafhandeling en testen op fouten

Geeft het nummer van de laatste IDAPI-fout als resultaat.

Syntaxis

DBERROR( )

Beschrijving

DBERROR( ) geeft als resultaat het IDAPI-foutnummer van de laatste IDAPI-fout die is gegenereerd door de huidige tabel. De IDAPI-foutmelding zelf kunt u weergeven met DBMESSAGE( ).

In de tabel bij de beschrijving van ERROR( ) worden ERROR( ), MESSAGE( ), DBERROR( ), DBMESSAGE( ), SQLERROR( ), SQLMESSAGE( ) en CERROR( ) met elkaar vergeleken.

Zie Help voor een lijst van alle foutmeldingen.

DBF( )

Tabellen

Geeft de naam van de tabel als resultaat die is geopend in het huidige of het opgegeven werkgebied.

Syntaxis

DBF([<*alias*>])

<alias>

Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

DBF( ) geeft de naam als resultaat van de tabel die is geopend in een opgegeven werkgebied. Als SET FULLPATH is ingeschakeld (ON), geeft de functie tevens het station en de directory voor de tabel als resultaat. Als u geen werkgebied opgeeft, wordt de naam van de tabel in het huidige werkgebied als resultaat gegeven.

Als in het huidige of opgegeven werkgebied geen tabel is geopend, geeft DBF( ) een lege tekenreeks ("") als resultaat.

Voorbeeld

In het volgende voorbeeld worden de verschillen tussen DBF( ), SELECT( ) en WORKAREA( ) gedemonstreerd als meerdere tabellen zijn geopend in meer dan één werkgebied:

```
CLOSE ALL
USE Vlucht IN 1
USE Vliegtg IN 2
USE Bedrijf IN 3
SELECT 3
? DBF()              && Geeft C:BEDRIJF.DBF als resultaat
? SELECT()           && Geeft 4 als resultaat
? WORKAREA()         && Geeft 3 als resultaat
```

Zie ook

ALIAS( ), MDX( ), NDX( ), SET FULLPATH, TAG( ), WORKAREA( ), USE

DBMESSAGE( )

Foutafhandeling en testen op fouten

Geeft de foutmelding van de laatste IDAPI-fout als resultaat.

Syntaxis

DBMESSAGE( )

Beschrijving

DBMESSAGE( ) geeft de foutmelding van de laatste IDAPI-fout als resultaat.

In de tabel bij de beschrijving van ERROR( ) worden ERROR( ), MESSAGE( ), DBERROR( ), DBMESSAGE( ), SQLERROR( ), SQLMESSAGE( ) en CERROR( ) met elkaar vergeleken.

Zie Help voor een lijst van alle foutmeldingen.

DEACTIVATE MENU — dBASE IV-menu's

Wist een actieve dBASE IV-menubalk en schakelt die uit zonder de definitie uit het geheugen te verwijderen. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows RELEASE OBJECT om een object van een formulier te halen.

Zie Help voor meer informatie over de syntaxis van DEACTIVATE MENU. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

DEACTIVATE POPUP — dBASE IV-menu's

Wist een actief dBASE IV popup-menu en schakelt het uit zonder de definitie van het menu uit het geheugen te verwijderen. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows RELEASE OBJECT om een object van een formulier te halen.

Zie Help voor meer informatie over de syntaxis van DEACTIVATE POPUP. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

DEACTIVATE WINDOW — dBASE IV-vensters

Wist van het scherm alle vensters die zijn gemaakt en ingeschakeld met het dBASE IV-commando ACTIVATE WINDOW zonder de definities van die vensters uit het geheugen te verwijderen. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows CLOSE FORMS of RELEASE OBJECT om een formulier te sluiten of vrij te maken.

Zie Help voor meer informatie over de syntaxis van DEACTIVATE WINDOW. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

DEBUG

Foutafhandeling en testen op fouten

Opent de Debugger van dBASE.

Syntaxis

DEBUG
[<*bestandsnaam*> | ? | <*bestandsnaamfilter*> |
<*procedurenaam*> | <*Naam door gebruiker gedefinieerde functie*>
[WITH <*parameterlijst*>]]

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Het bestand dat u op fouten wilt testen. DEBUG ? en DEBUG <*bestandsnaamfilter*> tonen het dialoogvenster **Bronbestand openen** waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, gebruikt dBASE de extensie .PRG.

<*procedurenaam*> | <*Naam door gebruiker gedefinieerde functie*>

De procedure of door de gebruiker gedefinieerde functie in een geopend programmabestand waarin u fouten wilt opsporen. De procedure of door gebruiker gedefinieerde functie moet in een programmabestand staan dat het commando DEBUG bevat en dat commando aanroept, of in een afzonderlijk geopend procedurebestand in het *zoekpad*. (U kunt een procedurebestand openen met SET PROCEDURE TO.) Zie de beschrijving van DO voor een uitleg van het zoekpad en de volgorde waarin dat wordt doorzocht als een programma, procedure of door gebruiker gedefinieerde functie wordt aangeroepen.

WITH <*parameterlijst*>

Geeft aan welke uitdrukkingen moeten worden doorgegeven aan een programma, procedure of door gebruiker gedefinieerde functie die op de eerste uitvoerbare regel PARAMETERS <*parameterlijst*> bevat. Zie de beschrijving bij PARAMETERS voor meer informatie over het doorgeven van parameters.

U kunt fouten opsporen in een door de gebruiker gedefinieerde functie met DEBUG...WITH door de door de gebruiker gedefinieerde functie rechtstreeks aan te roepen in een uitdrukking. In dat geval plaatst u de parameters voor de functie tussen de haakjes achter de naam van de door de gebruiker gedefinieerde functie. Raadpleeg de laatste alinea van de volgende sectie voor meer informatie.

Als u een parameterlijst opgeeft, moet u ook een van de voorgaande opties gebruiken. Een instructie als DEBUG WITH <*parameterlijst*> geeft een fout als resultaat.

Beschrijving

Met DEBUG schakelt u de Debugger in. U kunt de uitvoering van het programma dan interactief volgen en besturen. De Debugger toont tijdens de uitvoering informatie over de toestand van een programma, procedure of door de gebruiker gedefinieerde functie

(plus alle door dat programma of die subroutine geopende procedurebestanden). Zie *Programmeren* voor meer informatie. Daar wordt de Debugger volledig beschreven.

U kunt DEBUG zonder opties gebruiken in het commandovenster of in een programmabestand. Als u in het commandovenster DEBUG zonder opties gebruikt, wordt de Debugger geopend zonder dat een programma of subroutine wordt geladen. (U kunt vanuit de Debugger een bestand laden waarin u fouten wilt opsporen.) Als u in een programmabestand DEBUG zonder opties opgeeft, wordt het huidige programmabestand in de Debugger geladen. Als u in een programmabestand DEBUG *<bestandsnaam>* gebruikt, moet *<bestandsnaam>* een ander programmabestand zijn dan het bestand dat de instructie bevat, want anders krijgt u een fout.

U kunt in een programmabestand DEBUG *<procedurenaam>* en DEBUG *<Naam door gebruiker gedefinieerde functie>* gebruiken als de procedure of door de gebruiker gedefinieerde functie vanuit dat bestand wordt aangeroepen. U kunt beide commando's ook opgeven in het commandovenster of in een ander programmabestand dan het bestand dat de procedure of functie aanroept, maar het bestand moet dan wel in het zoekpad staan.

In de Debugger kunt u een programma regel voor regel uitvoeren en aangeven bij welke regel moet worden gestopt. Als u de uitvoering van een subroutine niet wilt onderbreken, kunt u over de regel stappen die de subroutine aanroept. De aanroep naar de subroutine wordt wel uitgevoerd en de subroutine zelf ook. Alleen wordt de uitvoering van de subroutine dan niet regel voor regel getoond. De Debugger stopt bij de eerste regel met een instructie na de subroutine.

Als u fouten opspoort in een door de gebruiker gedefinieerde functie met DEBUG *<Naam door gebruiker gedefinieerde functie>* WITH *<parameterlijst>*, wordt de resultaatwaarde van de functie genegeerd. Als u een door de gebruiker gedefinieerde functie wilt aanroepen in de context van een uitdrukking zodat de functie de correct waarde en het juiste gegevenstype als resultaat geeft, kunt u in een programmabestand DEBUG zonder parameters opgeven en vervolgens de instructie die de uitdrukking met de door de gebruiker gedefinieerde functie bevat. DEBUG opent het programmabestand in de Debugger. U kunt dan vervolgens naar de volgende regel gaan zodat de functie-aanroep wordt uitgevoerd.

In de Debugger kunt u als volgt over een regel met een aanroep naar een subroutine stappen:

- Klik op **Overheen stappen** op de knoppenbalk van de Debugger.
- Kies **Uitvoeren | Overheen stappen** in het menu van de Debugger.
- Druk op *F8*.

Voorbeeld

In de volgende voorbeelden wordt de Debugger aangeroepen met drie verschillende opties. Deze instructies geeft u in het commandovenster:

```
DEBUG ?       && Dialoogvenster "Bronbestand openen" verschijnt
DEBUG *.WFM   && Dialoogvenster "Bronbestand openen" verschijnt;
                 waarin .WFM-bestand kan worden gekozen
```

```
DEBUG DBKLOK  && Debugger starten met de broncode van;
                 Doklok.PRG in venster linksboven
```

Als u de volgende regel opneemt in een programma, wordt de Debugger automatisch gestart. U kunt dat ook doen met SET ECHO ON:

```
ON ERROR DEBUG PROGRAM( )
```

Zie ook

DISPLAY COVERAGE, GENERATE, ON ERROR, RESUME, SET COVERAGE, SET PATH, SUSPEND

DECLARE

Geheugenvariabelen

Definieert een of meer vaste array's.

Syntaxis

DECLARE <*arraynaam1*>"["<*Nuitdrukkingenlijst1*>"]"
[,<*arraynaam2*>"["<*Nuitdrukkingenlijst2*>"]" ...]

Teksthaakjes ([ ]) tussen aanhalingstekens vormen verplichte onderdelen van de syntaxis.

<arraynaam1>[,<arraynaam2> ...]

De geheugenvariabelen met de namen van de array's.

"["<Nuitdrukkingenlijst1>"]"[,..."["<Nuitdrukkingenlijst2>"]"][,...]

Numerieke of zwevende uitdrukkingen (van 1 tot en met 254). Het aantal uitdrukkingen dat u opgeeft, bepaalt het aantal dimensies van de array. Elke uitdrukking bepaalt hoeveel waarden (gegevenselementen) die dimensie bevat. Als u voor [<*Nuitdrukkingenlijst1*>] bijvoorbeeld [3,4] opgeeft, wordt een tweedimensionale array met drie rijen en twee kolommen gedefinieerd.

Beschrijving

Met DECLARE definieert u een array met een opgegeven grootte als een geheugenvariabele. Array-elementen kunnen van elk willekeurige gegevenstype zijn. (Een array-element kan ook de naam van een andere array zijn.) Een enkele array kan meerdere gegevenstypen bevatten. Als u DECLARE gebruikt, worden alle array-elementen geïnitialiseerd met de logische waarde .F.

Het aantal elementen in een array wordt alleen beperkt door het geheugen. U kunt array's met meer dan twee dimensies maken, maar de meeste array-functies in dBASE werken alleen op array's met een of twee dimensies.

U kunt op twee manieren naar de afzonderlijke elementen in een array verwijzen. U kunt de *indextekens* van het element gebruiken of het *elementnummer*. De indextekens geven aan in welke rij en kolom elk element zich bevindt. Elementnummers geven de

sequentiële plaats van het element in de array aan, te beginnen bij de eerste rij en de eerste kolom van de array. Het aantal elementen, rijen en kolommen in een array kunt u bepalen met ALEN( ).

Sommige dBASE-functies vereisen een elementnummer, terwijl andere juiste weer indextekens vereisen. Als u een- of tweedimensionale array's gebruikt, kunt u met AELEMENT( ) het elementnummer bepalen als u de indextekens weet en met ASUBSCRIPT( ) de indextekens als u het elementnummer weet.

Nadat u een array hebt gemaakt, kunt u waarden plaatsen in de cellen van de array met STORE. U kunt ook = gebruiken. Verder kunt u een reeks (bereik) cellen in de array vullen met AFILL( ). Als u elementen aan een array wilt toevoegen of uit een array wilt verwijderen, gebruikt u ADEL( ) en AINS( ). Met AGROW( ) en ARESIZE( ) kunt u de grootte van een array wijzigen of een eendimensionale array tweedimensionaal maken. Zie Hoofdstuk 5 in *Programmeren* voor meer informatie over het werken met array's.

Aan een programma of procedure kunt u array-elementen doorgeven als parameters. Ook kunt u een volledige array doorgeven als parameter door alleen maar de array-naam op te geven. Zie Hoofdstuk 4 in *Programmeren* voor meer informatie over het doorgeven van array's als parameters.

Voorbeeld

In het volgende voorbeeld wordt een relatie tussen twee tabellen tot stand gebracht om een weergave te maken met twee velden uit elke tabel. Vervolgens wordt met DECLARE de array BedrOvrz gemaakt en worden de gegevens naar de array gekopieerd. De inhoud van de array wordt met een DO WHILE-lus weergegeven in het resultatenpaneel van het commandovenster:

```
CLOSE ALL
CLEAR
USE Contact IN SELECT()   ORDER CompCode
USE Bedrijf IN SELECT()
SELECT Bedrijf
SET RELATION TO CompCode INTO Contact

DECLARE BedrOvrz[5,4]          && Array maken met;
                                  5 rijen en 4 kolommen
COPY TO ARRAY BedrOvrz NEXT 5;
   FIELDS Bedrijf->Bedrijf, ;
   Bedrijf->Plaats, Contact->CompCode, ;
   Contact->Contact
Teller=1
DO WHILE Teller<=5
  ? BedrOvrz[Teller,1], BedrOvrz[Teller,2]
  ?? BedrOvrz[Teller,3], BedrOvrz[Teller,4]
  Teller=Teller+1
ENDDO
CLOSE ALL
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ADEL( ), AELEMENT( ), AFILL( ), AGROW( ), AINS( ), ALEN( ), APPEND FROM ARRAY, ARESIZE( ), ASUBSCRIPT( ), COPY TO ARRAY, REPLACE FROM ARRAY, STORE

DEFINE

Objecten

Maakt een object van een klasse.

Syntaxis

DEFINE *<klassenaam> <objectnaam>*
[OF *<container>*]
[FROM *<rij, kolom>* TO *<rij, kolom>* | AT *<rij, kolom>*]
[PROPERTY *<standaardkenmerkenlijst>*]
[CUSTOM *<lijst eigen kenmerken>*]
[WITH *<parameterlijst>*]

<klassenaam>

De klasse van het object dat u maakt. Met DEFINE kunt u objecten van twintig standaardklassen maken:

Browse	Checkbox	Combobox
DDELink	DDETopic	Editor
Entryfield	Form	Image
Line	Listbox	Menu
Object	Ole	Pushbutton
Radiobutton	Rectangle	Scrollbar
Spinbox		
Text		

U kunt ook een eigen klasse opgeven die u hebt gedefinieerd met het commando CLASS...ENDCLASS.

<objectnaam>

Het identificatiesymbool van het object dat u maakt. *<objectnaam>* is de objectverwijzingsvariabele, die u gebruikt om toegang te verkrijgen tot de kenmerken van het object en die te wijzigen. Zie Hoofdstuk 10 in *Programmeren* voor meer informatie over objectverwijzingsvariabelen.

OF *<container>*

Geeft het object aan dat het object bevat dat u definieert. (Voor de meeste UI-objecten is de container een formulier.) OF *<container>* is verplicht als u het object aanduidt met *<indexoperator Nuitdr>*.

FROM <*rij*>, <*kolom*> TO <*rij*>, <*kolom*> | AT <*rij*>, <*kolom*>
Geeft de plaats en grootte bij aanvang aan van het object binnen het bevattende formulier. FROM en TO bepalen respectievelijk de coördinaten linksboven en rechtsonder van het object. AT bepaalt de positie van de linkerbovenhoek.

PROPERTY <*standaardkenmerkenlijst*>
Geeft de waarden aan die u toekent aan de ingebouwde kenmerken van het object.

CUSTOM <*lijst eigen kenmerken*>
Geeft nieuwe kenmerken aan die u voor het object maakt en kent daaraan waarden toe. Zie Hoofdstuk 10 in *Programmeren* voor meer informatie over eigen kenmerken.

WITH <*parameterlijst*>
Geeft de parameters aan die u aan het object doorgeeft. Definieer deze parameters met de clausule PARAMETERS van de commando CLASS...ENDCLASS.

Beschrijving

Met DEFINE maakt u de definitie van een object in het geheugen.

Een object bevat geheugenvariabelen die *kenmerken* worden genoemd. Bepaalde kenmerken hebben waarden die het object zelf aanpassen, terwijl andere verwijzen naar subroutines die worden uitgevoerd na bepaalde acties. Als bijvoorbeeld op een knop wordt geklikt, wordt de subroutine OnClick voor dat object uitgevoerd en de subroutine OnSelection van het bevattende formulier.

Elk object behoort tot een klasse. Een klasse is een specificatie, of sjabloon, voor een bepaald soort object. dBASE voor Windows beschikt over een groot aantal ingebouwde klassen waarmee u Windows-objecten (of stuurelementen) zoals keuzerondjes, knoppen en invoervakken kunt maken.

Met het commando CLASS...ENDCLASS kunt u een *eigen klasse* definiëren. U kunt dan gewone dBASE-code gebruiken om kenmerken en methoden te definiëren voor de objecten van die klasse. Zie Hoofdstuk 11 in *Programmeren* voor meer informatie over klassen.

Voorbeeld

In het volgende voorbeeld wordt DEFINE gebruikt om eigen klasse voor dialoogvensters te maken met verschillende handige functies:

```
rDialoog = NEW Dialog()
rDialoog.attentie("Attentie, attentie")
? rDialoog.JaNee("Vindt u dit goed?")
? rDialoog.HaalReeks("Hoe heet u?",;
  SPACE(20),"@!", "Achternaam eerst")
RETURN

CLASS Dialog
   this.Top = 20
   this.Left = 12
   this.Height = 7
```

```
    this.Width = 12
    this.Value = .f.
    this.Color = 'n/w'
    SET CUAENTER OFF

FUNCTION Attentie
PARAMETER tReeks, tTitel
PRIVATE nKnopKol
  this.Height = 8
  this.Value = ""
  this.Left = 39 - (this.Width/2)
    IF TYPE( 'tTitel') <> 'C'
      this.Text = ""
      this.Height = 7
    ELSE
      this.Text = tTitel
      this.Height = 9
    ENDIF
      this.Width = MAX(LEN(tReeks),LEN(this.Text))+7
DEFINE FORM F1;
  PROPERTY Top this.Top,;
    Left this.Left, ;
    Height this.Height, ;
    Width this.Width, ;
    MDI .F., ;
    Text this.Text
  DEFINE TEXT T1 of F1;
    PROPERTY Top 1, Left 3, ;
      Width LEN(tReeks), ;
      Text tReeks
      nKnopKol = (this.width/2)-5
  DEFINE PUSHBUTTON B1 of F1;
    PROPERTY Top 4, Left nKnopKol, ;
      Text "OK", OnClick {; Form.CLOSE() }
F1.READMODAL()
F1.RELEASE()
RETURN .T.

FUNCTION HaalReeks
PARAMETER tPrompt, tReeks, TSjabloon, tTitel
RETURN this.HaalWaarde(tPrompt,tReeks,LEN(tReeks),;
  TSjabloon, tTitel)

FUNCTION HaalGetal
PARAMETER tPrompt, nGetal, TSjabloon, tTitel
RETURN this.HaalWaarde(tPrompt,nGetal,;
  LEN(STR(nGetal)), TSjabloon, tTitel )

FUNCTION HaalDatum
PARAMETER tPrompt, dDatum, TSjabloon, tTitel
RETURN CTOD(this.HaalWaarde(tPrompt,DTOC(dDatum),8,;
  TSjabloon, tTitel))

FUNCTION JaNee
PARAMETER tReeks, tTitel, lStartwaarde
```

```
PRIVATE nKnopKol1, lReswaarde
   this.Height = 8
   lReswaarde = lStartwaarde
   this.Value = .t.
   this.Left = 39 - (this.width/2)
     IF TYPE( 'tTitel') <> 'C'
        this.Text = ""
        this.Height = 7
     ELSE
        this.Text = tTitel
        this.Height = 9
     ENDIF
   this.Width=MAX(18,MAX(LEN(tReeks),;
     LEN(this.Text))+ 7 )
DEFINE FORM F1;
  PROPERTY Top this.Top, ;
    Left this.Left, ;
    Height this.Height, ;
    Width this.Width, ;
    MDI .F., ;
    Text this.Text
  DEFINE TEXT T1 OF F1;
    PROPERTY Top 1, left 3, ;
      Width LEN(tReeks), ;
      Text tReeks
      nKnopKol1 = (this.width/2) - 9
  DEFINE PUSHBUTTON Yes OF F1;
    PROPERTY Top 4, Left nKnopKol1, ;
      Width 5, Text "Ja", OnClick this.JaInst
  DEFINE PUSHBUTTON No OF F1;
    PROPERTY Top 4, Left nKnopKol1 + 12,;
      Width 5, Text "Nee", OnClick this.NeeInst
F1.READMODAL()
F1.RELEASE()
RETURN lReswaarde

FUNCTION HaalWaarde
PARAMETER tPrompt, tReeks, nLengte, TSjabloon, tTitel
PRIVATE nKnopKol
  this.Width = nLengte + 8 + LEN(tPrompt)
  this.Value = ""
  this.Left = 39 - (this.width/2)
    IF TYPE( 'tTitel') <> 'C'
      this.Text = ""
      this.Height = 8
    ELSE
      this.Text = tTitel
      this.Height = 10
    ENDIF
      this.Width=MAX(LEN(tReeks)+nLengte,;
        LEN(this.Text)) + 8
DEFINE FORM F1;
  PROPERTY Top this.Top, ;
    Left this.Left, ;
    Height this.Height, ;
```

```
    Width this.Width, ;
    MDI .F., ;
    Text this.Text, ;
    ColorNormal this.Color

  DEFINE TEXT T1 OF F1;
    PROPERTY Top 1, Left 3, ;
      WIDTH LEN(tPrompt), ;
      Text tPrompt
        IF TYPE('TSjabloon') = 'C'
          DEFINE ENTRYFIELD IV1 OF F1;
            PROPERTY Top 1, Left 4+LEN(tPrompt),;
              Width nLengte, ;
              Value tReeks, ;
              Datalink "tReeks", ;
              Picture TSjabloon
        ELSE
          DEFINE ENTRYFIELD IV1 OF F1;
            PROPERTY Top 1, Left 4 + LEN(tPrompt), ;
              Width nLengte, ;
              Value tReeks, ;
              Datalink "tReeks"
        ENDIF
        nKnopKol = (this.width/2) - 5
  DEFINE PUSHBUTTON B1 OF F1;
    PROPERTY Top 4, Left nKnopKol, ;
      Text "OK", OnClick {; Form.Close() }
F1.READMODAL()
F1.RELEASE()
RETURN tReeks

FUNCTION JaInst
lReswaarde = .T.
Form.CLOSE()
RETURN .T.

FUNCTION NeeInst
lReswaarde = .F.
Form.CLOSE()
RETURN .T.
ENDCLASS
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CLASS...ENDCLASS, REDEFINE

DEFINE BAR

dBASE IV-menu's

Maakt een optie en de bijbehorende informatieregel in een dBASE IV popup-menu. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) om menu's bij formulieren te maken en te activeren.

Zie Help voor meer informatie over de syntaxis van DEFINE BAR. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

DEFINE BOX

Afdrukken

Tekent een uit tekens bestaand kader om uitvoer van het commando ?. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows gebruikt u DEFINE om rechthoeken te maken en op formulieren te plaatsen.

Zie Help voor volledige informatie over de syntaxis van DEFINE BOX. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor informatie over het maken van rechthoeken en formulieren.

Syntaxis

DEFINE BOX FROM <*linkerkolom*> TO <*rechterkolom*> HEIGHT <*Nuitdr1*>
[AT LINE <*Nuitdr2*>]
[SINGLE | DOUBLE | <*kaderdefinitielijst*>]

<*linkerkolom*>

De linkerkolom van het kader. Als u een getal kleiner dan 0 opgeeft voor <*linkerkolom*>, wordt een foutmelding getoond. De bovengrens voor <*linkerkolom*> is 255.

TO <*rechterkolom*>

De rechterkolom van het kader. Als u een getal kleiner dan 0 opgeeft voor <*rechterkolom*>, wordt een foutmelding getoond. De bovengrens voor <*rechterkolom*> is 255.

HEIGHT <*Nuitdr1*>

De hoogte van het kader in een regels, <*Nuitdr1*>.

AT LINE <*Nuitdr2*>

De bovenste regel van het kader, <*Nuitdr2*>. Als u geen regel opgeeft bij AT LINE, verschijnt de bovenster regel van het kader op de huidige regel.

SINGLE

Tekent een kader dat uit een enkele lijn bestaat. Het standaardkader is ingesteld met SET BORDER.

DOUBLE
Tekent een kader dat uit een dubbele lijn bestaat. Het standaardkader is ingesteld met SET BORDER.

<kaderdefinitielijst>
Een lijst van tekens waaruit het kader bestaat. Zie SET BORDER voor meer informatie over het definiëren van kaders.

Beschrijving
Met DEFINE BOX tekent u een uit tekens bestaand dBASE IV-kader om de uitvoer van het commando ?. U schakelt DEFINE BOX in door .T. toe te kennen aan de systeemgeheugenvariabele _box. Zie Hoofdstuk 24 in *Programmeren* voor meer informatie over continue en onderbroken uitvoer.

Voorbeeld
In het volgende voorbeeld wordt een kader getekend om de kop boven een pagina te benadrukken. Het kader is vijf rijen hoog en op de derde rij wordt "Dit is een kop" afgedrukt:

```
SET PRINTER ON
SET TALK OFF
EJECT
_box=.t.                    && _box moet waarde .t. hebben
DEFINE BOX FROM 10 TO 70 HEIGHT 5
?
?
?
?? "Dit is een kop" at 30
?
?
SET PRINTER OFF
CLOSE PRINTER
```

Overdraagbaarheid
Wordt niet ondersteund in dBASE III PLUS.

Zie ook
@...TO, DEFINE

DEFINE COLOR

Kleuren en fonts

Maakt en benoemt een eigen kleur.

Syntaxis

DEFINE COLOR *<kleurnaam>*
<rood Nuitdr>, *<groen Nuitdr>*, *<blauw Nuitdr>*

<rood Nuitdr>, *<groen Nuitdr>*, *<blauw Nuitdr>*

Geeft de hoeveelheden rood, groen en blauw aan waaruit de nieuwe kleur bestaat. Elk getal bepaalt de intensiteit van de kleurcomponent. Het bereik voor de waarden loopt van 0 (laagste intensiteit) tot 255 (hoogste intensiteit).

Beschrijving

Met DEFINE COLOR kunt u een eigen kleur maken. Nadat u *<kleurnaam>* hebt gedefinieerd, kunt u die naam gebruiken in plaats van de standaardkleuren zoals R, W, BG enzovoort.

De kleur die u met DEFINE COLOR maakt, wordt gebaseerd op drie getallen, *<rood Nuitdr>*, *<groen Nuitdr>* en *<blauw Nuitdr>*. Als u deze waarden wijzigt, verandert de uiteindelijke kleur. Als u bijvoorbeeld *<groen Nuitdr>* groter of kleiner maakt, wordt de hoeveelheid groen in de nieuwe kleur meer of minder.

Met de functie GETCOLOR( ) opent u een dialoogvenster waarin u een eigen kleur kunt maken of kunt kiezen uit een palet met beschikbare kleuren. Vervolgens definieert u de nieuwe kleur door bij DEFINE COLOR de resultaatwaarden van GETCOLOR( ) op te geven.

U mag voor *<kleurnaam>* geen standaardkleurnamen (zoals blue, red, white of black) gebruiken. Als u dat doet, wordt een fout als resultaat gegeven als u later SET COLOR TO *<kleurnaam>* gebruikt. Verder is het niet mogelijk een standaardkleur opnieuw te definiëren met een instructie als DEFINE COLOR R *<rood Nuitdr>*, *<groen Nuitdr>*, *<blauw Nuitdr>*. Als u op een later tijdstip R gebruikt in een instructie met SET COLOR TO, wordt gewoon de interne waarde voor R (rood) gebruikt en niet de kleur die u hebt gedefinieerd.

Voorbeeld

In het volgende voorbeeld worden met DEFINE COLOR de kleuren wit, zwart, rood, groen, blauw en geel gedefinieerd. Vervolgens worden deze kleuren gebruikt met SET COLOR TO:

```
OudeKleuren=SET("ATTRIBUTE")
DEFINE COLOR Kleur_wit     255, 255,  255
* Komt overeen met W
DEFINE COLOR Kleur_zwart     0,   0,    0
* Komt overeen met N
DEFINE COLOR Kleur_rood    255,   0,    0
* Komt overeen met R
DEFINE COLOR Kleur_groen     0, 255,    0
* Komt overeen met G
DEFINE COLOR Kleur_blauw     0,   0,  255
* Komt overeen met B
DEFINE COLOR Kleur_geel    255, 255,    0
*
```

```
SET COLOR TO Kleur_wit/Kleur_groen
? "Ik zit mij"
?? "Kleur_wit/Kleur_groen" AT 40
SET COLOR TO Kleur_rood/Kleur_blauw
? "voor het vensterglas"
?? "Kleur_rood/Kleur_blauw" AT 40
SET COLOR TO Kleur_blauw/Kleur_rood
? "onnoemlijk te vervelen"
?? "Kleur_blauw/Kleur_rood" AT 40
SET COLOR TO Kleur_blauw/Kleur_groen
? "Ik wou dat ik"
?? "Kleur_blauw/Kleur_groen" AT 40
SET COLOR TO Kleur_rood/Kleur_blauw
? "twee hondjes was"
?? "Kleur_rood/Kleur_blauw" AT 40
SET COLOR TO Kleur_geel/Kleur_groen
? "Dan kon ik samen spelen"
?? "Kleur_geel/Kleur_groen" AT 40
WAIT
SET COLOR TO &OudeKleuren  && Oorspronkelijke kleuren herstellen
CLEAR
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ColorHighlight, ColorNormal, GETCOLOR( ), SET COLOR TO, SET COLOR OF

DEFINE MENU

dBASE IV-menu's

Benoemt een dBASE IV-menubalk en begint de definitie in het geheugen. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) om menu's te maken en te activeren voor formulieren.

Zie Help voor meer informatie over de syntaxis van DEFINE MENU. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

DEFINE PAD

dBASE IV-menu's

Maakt en benoemt een strip in een bestaande dBASE IV-menubalk. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van DEFINE PAD. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

DEFINE POPUP

dBASE IV-menu's

Maakt een dBASE IV popup-menu en slaat de definitie op in het geheugen. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van DEFINE POPUP. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

DEFINE WINDOW

dBASE IV-vensters

Maakt een dBASE IV-venster, dat wil zeggen een rechthoekig gebied waarin menu's en invoergebieden kunnen worden getoond en ingeschakeld. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE om *formulieren* te maken in plaats van dBASE IV-vensters.

Zie Help voor meer informatie over de syntaxis van DEFINE WINDOW. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

DELETE

Velden en records

Markeert records in dBASE-tabellen voor verwijdering. In het geval van Paradox- of SQL-tabellen verwijdert DELETE records uit de tabel.

Syntaxis

DELETE
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]

<*bereik*>

Het aantal records dat moet worden verwijderd. RECORD <*n*> geeft een enkel record aan door middel van zijn recordnummer. NEXT <*n*> geeft *n* records aan, te beginnen bij het huidige record. ALL specificeert alle records. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR <*voorwaarde1*>
WHILE <*voorwaarde2*>
Bepaalt welke records worden beïnvloed door DELETE. FOR beperkt DELETE tot records die voldoen aan <*voorwaarde1*>. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang <*voorwaarde2*> waar is.

Beschrijving

Als u DELETE gebruikt om records in een dBASE-tabel te markeren voor verwijdering, worden die records niet feitelijk uit de tabel verwijderd. De gemarkeerde records worden pas verwijderd als u PACK gebruikt. DELETE zonder opties markeert alleen het huidige record. Als u werkt met Paradox- of SQL-tabellen, worden de records onmiddellijk uit de tabel verwijderd.

In een lijst of weergave verschijnt een asterisk naast records die zijn gemarkeerd voor verwijdering.

Als u met een tabel wilt werken alsof u het commando PACK hebt gegeven, maar zonder gemarkeerde records onherstelbaar te verwijderen, moet u SET DELETED inschakelen (ON, de standaardinstelling). Met RECALL kunt u de verwijderde records dan herroepen (de wismarkering ervan verwijderen). Met ZAP verwijdert u echter alle records.

Voorbeeld

In het volgende voorbeeld wordt DELETE gebruikt om alle records te verwijderen met een openingsbalans die kleiner is dan F 750. Met SET DELETED ON worden de gemarkeerde records uitgesloten van de volgende bladeropdracht:

```
USE Afnemers
DELETE FOR Openbalans < 750
SET DELETED ON
GO TOP
BROWSE FIELDS Bedrijf, Plaats, Areacode, Zipcode,;
   Openbalans NOMODIFY NOAPPEND NODELETE
RECALL ALL                && Alle verwijderingsmarkeringen weghalen
CLOSE ALL
```

In het volgende voorbeeld wordt DELETE gebruikt om vijf willekeurig getrokken winnaars te markeren en op het scherm te tonen. U kunt ook op een soortgelijke manier LABEL FORM gebruiken om etiketten af te drukken:

```
SET TALK OFF
SET DELETED OFF
USE Bedrijf
RECALL ALL
SET DECIMALS TO 0
Aantal=1
DO WHILE Aantal<=5
   STORE (INT(RAND() *RECCOUNT() )+1) TO Mark
   GOTO Mark
   IF .NOT. DELETED()
       DELETE
```

```
        Aantal=Aantal+1
    ENDIF
ENDDO
SCAN FOR DELETED()
  ? Bedrijf
ENDSCAN
```

Hetzelfde principe is van toepassing als u een nieuwe tabel wilt maken met deze willekeurig getrokken records:

```
SET SAFETY OFF
COPY TO Winnaars FOR DELETED()
RECALL ALL
USE Winnaars
BROWSE
```

Zie ook

PACK, RECALL, SET DELETED, ZAP

DELETE FILE

Stations- en bestandsfuncties

Verwijdert een bestand van schijf.

Syntaxis

DELETE FILE <*bestandsnaam*> | ? | <*bestandsnaamfilter*>

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Geeft het bestand aan dat moet worden verwijderd. ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een bestand kunt kiezen.

Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, wordt geen extensie gebruikt.

Beschrijving

DELETE FILE en ERASE zijn onderling uitwisselbare commando's. Zie ERASE voor meer informatie.

Voorbeeld

In de volgende voorbeelden wordt DELETE FILE gebruikt:

```
DELETE FILE Tijd.prg
DELETE FILE ?
* Dialoogvenster "Bronbestand openen" verschijnt
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS, maar ERASE wel. Het argument *<bestandsnaamfilter>* wordt niet ondersteund in dBASE IV.

Zie ook

ERASE, RENAME, SET PATH

DELETE TABLE

Tabellen

Verwijdert een opgegeven tabel.

Syntaxis

DELETE TABLE *<tabelnaam>* | ? | *<bestandsnaamfilter>*
[[TYPE] PARADOX | DBASE]

<tabelnaam> | ?*<bestandsnaamfilter>*

De naam van de tabel die u wilt verwijderen. DELETE TABLE ? en DELETE *<bestandsnaamfilter>* tonen een dialoogvenster waarin u de tabel kunt kiezen die u wilt verwijderen.

U kunt ook een tabel verwijderen uit een database (gedefinieerd met het IDAPI-configuratieprogramma) door voor de naam van de tabel de naam van de database op te geven (tussen dubbele punten) zoals *:databasenaam:tabelnaam*. Als de database nog niet open is, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

[TYPE] PARADOX | DBASE

Geeft het type aan van de tabel die u wilt verwijderen: een Paradox- of een dBASE-tabel.

Beschrijving

Met het commando DELETE TABLE verwijdert u tabellen en de bijbehorende .NDX- en .MDX-indexbestanden. Let er op dat de tabel niet in gebruik is als u die wilt verwijderen.

Voorbeeld

In het volgende voorbeeld wordt DELETE TABLE gebruikt om twee tijdelijke tabellen te verwijderen die zijn gebruikt voor een alternatieve weergave van de tabel Klanten:

```
SET SAFETY OFF
USE Klanten
COPY TO NwNamen STRUCTURE EXTENDED
USE NwNamen EXCLUSIVE
BROWSE          && Ongewenste velden verwijderen;
                   of ontwerp van de tabel wijzigen
```

```
PACK            && Nieuwe structuur opslaan
CREATE NwKlant FROM NwNamen
APPEND FROM Klanten FOR Regio = "NW"
GO TOP
BROWSE
CLOSE DATABASES
DELETE TABLE NwNamen TYPE DBASE
DELETE TABLE NwKlant TYPE DBASE
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DELETE FILE, DELETE TAG, ERASE

DELETE TAG

Tabelindeling

Verwijdert opgegeven labels uit .MDX-bestanden. U kunt met dit commando ook afzonderlijke labels verwijderen die zijn gedefinieerd voor Paradox- en SQL-tabellen.

Syntaxis

DELETE TAG *<labelnaam1>*
    [OF *<bestandsnaam1>* | ? | *<bestandsnaamfilter1>*]
  [, *<labelnaam2>*
    [OF *<bestandsnaam2>* | ? | *<bestandsnaamfilter2>*] ...]

<labelnaam1>, <labelnaam2>, ... <labelnaam n>

De namen van de indexlabels die uit .MDX-bestanden moeten worden verwijderd.

OF *<bestandsnaam1>* | ? | *<bestandsnaamfilter1>*

Geeft de naam aan van het .MDX-bestand met de label(s) die u wilt verwijderen. OF ? en OF *<bestandsnaamfilter1>* geven een dialoogvenster weer waarin u meervoudig indexbestand kiest. Als u een bestand zonder pad opgeeft, zoekt dBASE voor Windows het bestand in de huidige directory en vervolgens in het pad dat is ingesteld met SET PATH. Als u een bestand zonder extensie opgeeft, gebruikt dBASE de extensie .MDX. Als u geen indexbestand opgeeft, wordt aangenomen dat de indexlabel die u wilt verwijderen, zich bevindt in een indexbestand dat dezelfde naam heeft als de huidige tabel.

Beschrijving

Met DELETE TAG kunt u indexlabels verwijderen uit .MDX-bestanden voor dBASE-tabellen en secundaire indexen voor Paradox-tabellen. In een enkel .MDX-bestand mogen maximaal 47 indexlabels staan, dus kan het verwijderen van overbodige labels

handig zijn voor het vrijmaken van nieuwe plaatsen en het beperken van de ruimte die een .MDX-bestand in het geheugen en op schijf inneemt.

In het geval van dBASE-tabellen moet het .MDX-bestand zijn geopend voordat u labels kunt verwijderen. Als u alle labels in een .MDX-bestand verwijderd, wordt het .MDX-bestand zelf ook verwijderd. Als u het produktie-MDX-bestand verwijdert door alle indexlabels te verwijderen, wordt de header van het tabelbestand gewijzigd om aan te geven dat er voor de tabel geen produktie-index meer bestaat.

In een multi-user-omgeving moet de tabel die hoort bij de indexen die u wilt verwijderen, exclusief zijn geopend. Als u DELETE TAG zonder argumenten gebruikt voor een Paradox-tabel, wordt de primaire index verwijderd.

Voorbeeld

In het volgende voorbeeld wordt een tijdelijke index gemaakt. Die index wordt vervolgens gebruikt in een instructie met BROWSE en daarna verwijderd met DELETE TAG:

```
USE Bedrijf EXCLUSIVE
INDEX ON Btwnr TAG Btwnr
BROWSE FIELDS Btwnr, Bedrijf
* Index Btwnr is niet meer nodig, dus verwijderen
DELETE TAG Btwnr
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. Verder is de optie *<bestandsnaamfilter>* niet beschikbaar in dBASE IV.

Zie ook

CLOSE INDEXES, COPY INDEXES, SET INDEX, TAG( )

DELETED( )

Velden en records

Hiermee bepaalt u of in de huidige of de opgegeven tabel records zijn gemarkeerd voor verwijdering.

Syntaxis

DELETED([*<alias>*])

<alias>

Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

DELETED( ) geeft .T. als resultaat als het huidige record in het opgegeven werkgebied is gemarkeerd voor verwijdering. Zo niet, dan geeft DELETED( ) .F. als resultaat. Als u geen werkgebied opgeeft, wordt het huidige record in het actieve werkgebied gecontroleerd.

Als in het actieve of opgegeven werkgebied geen tabel is geopend, geeft DELETED( ) ook .F. als resultaat.

Voorbeeld

In het volgende voorbeeld wordt DELETE gebruikt om een deelverzameling van de volledige tabel te markeren. DELETED( ) wordt vervolgens gebruikt om een bladerbewerking uit te voeren die alleen die deelverzameling van de records weergeeft en om een nieuwe tabel (Nrdwst) te maken die alleen records bevat met NW in het veld Regio:

```
SET TALK OFF
SET SAFETY OFF
USE Klanten
SET DELETED OFF
DELETE FOR Regio = "NW"
GO TOP
BROWSE FIELDS Naam, Plaats, Regio FOR DELETED()
COPY TO Nrdwst FOR DELETED()
USE Nrdwst
BROWSE
CLOSE ALL
```

In het volgende voorbeeld wordt DELETED( ) gebruikt in een instructie met INDEX om alle voor verwijdering gemarkeerde records aan het begin van een geordende tabel te plaatsen:

```
USE Klanten EXCLUSIVE
DELETE FOR Regio = "NW"
INDEX ON IIF(DELETED() ,"AA"+Regio,"Z" ;
  +Regio)TAG VerwRegio
BROWSE FIELDS Naam, Plaats, Regio
* NW records komen nu bovenaan te staan
* andere regio's alfabetisch daar onder.
RECALL ALL
CLOSE ALL
```

Zie ook

DELETE, PACK, RECALL, SET DELETED

DESCENDING( ) — Tabelindeling

Geeft aan of de opgegeven index is gemaakt met het sleutelwoord DESCENDING (aflopend).

Syntaxis

DESCENDING([<.*MDX-bestandsnaam Nuitdr*>,] <*indexpositie Nuitdr*> [, <*alias*>])

<.*MDX-bestandsnaam Nuitdr*>

Geeft een meervoudig indexbestand aan dat de indexlabel bevat dat u wilt controleren.

<*indexpositie Nuitdr*>

Geeft een indexlabel aan door zijn positie in een .MDX-bestand of de positie van de indexlabel in de lijst van geopende bestanden in het huidige of opgegeven werkgebied.

<*alias*>

Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

De functie DESCENDING( ) geeft .T. als resultaat als de indexlabel die is aangegeven door <*indexpositie Nuitdr*>, is gemaakt met het sleutelwoord DESCENDING. Anders wordt .F. als resultaat gegeven. De optionele parameter <*alias*> geeft het werkgebied aan waar u de indexlabel wilt controleren. Als u geen werkgebied opgeeft, wordt het huidige werkgebied gebruikt.

Als u niet de naam van een .MDX-bestand opgeeft, controleert de functie DESCENDING( ) de indexlabel in alle geopende .MDX-bestanden. Als een produktie-MDX-bestand en andere .MDX-bestanden zijn geopend, controleert de functie DESCENDING( ) alleen indexlabels in het produktie-MDX-bestand.

Als u geen indexlabel opgeeft, bepaalt DESCENDING( ) of de hoofdindex is gemaakt met het sleutelwoord DESCENDING. DESCENDING( ) geeft .F. als resultaat als de index een .NDX-bestand is of als er geen hoofdindex is.

Als het opgegeven .MDX-bestand of de opgegeven indexlabel niet bestaat, geeft de functie DESCENDING( ) een foutmelding.

Voorbeeld

In het volgende voorbeeld wordt DESCENDING( ) gebruikt om te bepalen of de sleutel voor een opgegeven index is gemaakt op basis van een aflopende index:

```
USE Bedrijf EXCLUSIVE
INDEX ON Bedrijf TAG Bedrijf
SET ORDER TO TAG Bedrijf
? TAG() , "Aflopend = ", DESCENDING()

INDEX ON Bedrijf TAG Aflpnd DESCENDING
? TAG() , "Aflopend = ", DESCENDING()
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FOR( ), INDEX, KEY( ), MDX( ), ORDER( ), TAGCOUNT( ), TAGNO( ), UNIQUE( )

DIFFERENCE( ) Tekenreeksgegevens

Geeft als resultaat een getal dat het fonetische verschil tussen twee tekenreeksen voorstelt.

Syntaxis

DIFFERENCE(<*Tuitdr1*> | <*memoveld1*>, <*Tuitdr2*> | <*memoveld2*>)

<*Tuitdr1*> | <*memoveld1*>

De eerste tekenuitdrukking of het eerste memoveld dat met SOUNDEX( ) moet worden geëvalueerd en moet worden vergeleken met de tweede waarde.

<*Tuitdr2*> | <*memoveld2*>

De tweede tekenuitdrukking of het tweede memoveld dat met SOUNDEX( ) moet worden geëvalueerd en moet worden vergeleken met de eerste waarde.

Beschrijving

SOUNDEX( ) geeft als resultaat een uit vier tekens bestaande code die de fonetische waarde van een tekenuitdrukking of memoveld voorstelt. DIFFERENCE( ) vergelijkt deze codes van twee tekenuitdrukkingen of memovelden en geeft een geheel getal van 0 tot 4 als resultaat. dat getal stelt het verschil tussen beide codes voor.

Een resultaatwaarde van 0 duidt op het grootste verschil tussen beide SOUNDEX( )-codes: de twee uitdrukkingen hebben geen SOUNDEX( )-tekens gemeenschappelijk. Een resultaatwaarde van 4 duidt op het kleinste verschil: de twee uitdrukkingen hebben alle vier de SOUNDEX( )-tekens gemeenschappelijk. Als DIFFERENCE( ) echter wordt toegepast op korte tekenreeksen, kan de functie onverwachte resultaten opleveren:

```
? SOUNDEX("Mam")           && Geeft M500 als resultaat
? SOUNDEX("Pap")           && Geeft P100 als resultaat
? DIFFERENCE("Mam","Pap") && Geeft 2 als resultaat
```

Met LIKE( ) kunt u overeenkomsten per teken controleren in plaats van fonetische overeenkomsten.

Voorbeeld

In het volgende voorbeeld wordt DIFFERENCE( ) gebruikt om de fonetische overeenkomsten tussen twee tekenreeksen te bepalen:

```
? SOUNDEX("Lepels")                && Geeft L142 als resultaat
? SOUNDEX("Labels")                && Geeft L142 als resultaat
? DIFFERENCE("Lepels","Labels")    && Geeft 4 als resultaat
? DIFFERENCE("Copy","Kopie")       && Geeft 3 als resultaat
? DIFFERENCE("Daniel","Damien")    && Geeft 2 als resultaat
```

```
? DIFFERENCE("Mand","Maand")        && Geeft 4 als resultaat
? DIFFERENCE("dBASE","C")           && Geeft 1 als resultaat
? DIFFERENCE("Motorolie","Koekjes") && Geeft 0 als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. In dBASE IV worden geen memovelden als argument ondersteund.

Zie ook

LIKE( ), SOUNDEX( )

DIR/DIRECTORY

Stations- en bestandsfuncties

Levert een directorylijst.

Syntaxis

DIR (of DIRECTORY)
[[ON]<*station*>:]
[[LIKE] [<*pad*>\] [<*bestandsnaam*> | <*bestandsnaamfilter*>]]

[ON] <*station*>:

Geeft het station aan waarvan u een directorylijst wilt. Achter ON <*station*> is een dubbele punt (:) optioneel. Als u echter <*station*> zonder ON gebruikt, moet u een dubbele punt gebruiken.

ON is optioneel en heeft geen gevolgen. Het sleutelwoord is alleen beschikbaar vanwege de compatibiliteit met dBASE IV.

[LIKE] [<*pad*>\] <*bestandsnaam*> | <*bestandsnaamfilter*>

Geeft een pad en/of bestandsnaamspecificatie aan. Gebruik de optie <pad> om een directorylijst te maken van een andere dan de huidige directory op het huidige station of op <*station*>. met de optie <bestandsnaam> kunt u een enkele bestandsnaam tonen. Met de optie <*bestandsnaamfilter*> kunt u een lijst van alleen bepaalde bestanden tonen.

LIKE is optioneel en heeft geen gevolgen voor de werking van het commando. Het sleutelwoord is alleen beschikbaar vanwege de compatibiliteit met dBASE IV.

Beschrijving

DIR (of DIRECTORY) is een commando waarmee u een directorylijst kunt weergeven. Van elk bestand worden de naam, de grootte in bytes en de datum van de laatste wijziging getoond. DIR laat tevens het totaal aantal bytes zien dat door de getoonde bestanden in beslag wordt genomen, plus het aantal bytes dat nog op schijf beschikbaar is en de totale schijfruimte.

DIR zonder argumenten geeft alleen informatie over bestanden met de extensie .DBF in de huidige directory. Naast de informatie die gewoonlijk wordt weergegeven, wordt dan tevens het aantal records in elke database getoond.

U mag jokertekens (* en ?) gebruiken om meerdere bestanden met overeenkomstige namen op te geven. DIR zonder argumenten, DIR *.DBF en DIR * zijn allemaal onderling uitwisselbare instructies. DIR N* toont alle .DBF-bestanden waarvan de naam begint met N. DIR N???? toont alle .DBF-bestanden waarvan de naam uit vijf tekens bestaat en begint met N. DIR N*.* toont alle bestanden die beginnen met een N en met elke mogelijke extensie. DIR N*. toont alle bestanden die beginnen met een N en geen extensie hebben.

Als u een station en een pad of een bestandsspecificatie opgeeft zonder ON of LIKE te gebruiken, moet u achter de naam van het station een dubbele punt zetten. DIR ON C LIKE *.BAT en DIR C:*.BAT zijn onderling uitwisselbare instructies.

Een pad moet eindigen op een schuine streep naar links (\), zoals in dit voorbeeld:

```
* Deze instructie toont alle namen van .DBF-bestanden
DIR C:\DBASEWIN\VOORBD\
* Deze instructie toont "0 bytes gebruikt in 0 bestanden"
DIR C:\DBASEWIN\VOORBD
```

Als u de toets *F4* geen andere functie hebt gegeven met ON KEY of SET FUNCTION, kunt u op *F4* drukken om DIR uit te voeren.

De uitvoer van DIR wordt beïnvloed door de instellingen van SET DATABASE en SET DBTYPE.

Voorbeeld

In de volgende voorbeelden wordt DIR gebruikt:

```
DIR         && Alle tabellen in huidige subdirectory
DIR ON B:   && Alle tabellen op station B
DIR A:      && Alle tabellen op station A
DIR *.prg   && Alle programmabestanden
DIR LIKE *.* ON C:\DBASE
* Alle bestanden in C:\DBASE
```

Overdraagbaarheid

De opties ON en LIKE worden niet ondersteund in dBASE III PLUS.

Zie ook

DISPLAY FILES, FILE( ), LIST FILES, FSIZE( ), ON KEY, SET DATABASE, SET DEFAULT, SET DBTYPE TO, SET FUNCTION

DISKSPACE( ) — Stations- en bestandsfuncties

Geeft als resultaat het aantal bytes dat beschikbaar is op het huidige of opgegeven station.

Syntaxis

DISKSPACE([<*station Nuitdr*>])

<station Nuitdr>

Een stationsnummer van 1 tot 26. De nummers 1 en 2 komen bijvoorbeeld overeen met de stations A en B.

Zonder <*station Nuitdr*> of als <*station Nuitdr*> gelijk is aan 0, geeft DISKSPACE( ) het aantal beschikbare bytes op het huidige station als resultaat.

Als <*station Nuitdr*> kleiner is dan 0 of groter dan 26, geeft DISKSPACE( ) het beschikbare aantal bytes op het station van waar dBASE is geladen als resultaat.

Beschrijving

Met DISKSPACE( ) kunt u bepalen hoeveel ruimte u nog beschikbaar hebt op een station.

DISKSPACE( ) kan samen met RECCOUNT( ) en RECSIZE( ) worden toegepast in applicaties die automatisch reservekopieën maken van databasebestanden. Met deze functie kunt u vaststellen of er nog voldoende ruimte is op een schijf.

Voorbeeld

In de volgende voorbeelden wordt DISKSPACE( ) gebruikt:

```
? DISKSPACE()   && Huidige station
? DISKSPACE(2)  && Station B
? DISKSPACE(-1) && dBASE-station
```

Overdraagbaarheid

Het optionele argument <*station Nuitdr*> wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

HOME( ), RECCOUNT( ), RECSIZE( ), SET DEFAULT

DISPLAY

Tabelindeling

Toont records uit de huidige tabel in het resultatenpaneel van het commandovenster.

Syntaxis

DISPLAY
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[[FIELDS] <*uitdrukkingenlijst*>]

[OFF]
[TO FILE <*bestandsnaam*> | ? | <*bestandsnaamfilter*>] | [TO PRINTER]

<*bereik*>
Het aantal records dat moet worden weergegeven. RECORD <*n*> geeft een enkel record aan door middel van zijn recordnummer. NEXT <*n*> geeft *n* records aan, te beginnen bij het huidige record. ALL specificeert alle records. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR <*voorwaarde1*>
WHILE <*voorwaarde2*>
Bepaalt welke records worden weergegeven. FOR beperkt DISPLAY tot records die voldoen aan <*voorwaarde1*>, vanaf het eerste record in de tabel of het bereik tot het eind van de tabel of het bereik. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang <*voorwaarde2*> waar is.

FIELDS <*uitdrukkingenlijst*>
Veldwaarden of uitdrukkingen waarvan u de inhoud (waarden) wilt tonen. De namen van de velden worden in de lijst gescheiden door komma's. Als u hier niets opgeeft, worden de waarden van alle velden in het huidige record of de opgegeven records weergegeven. Het sleutelwoord FIELDS is alleen aanwezig voor de leesbaarheid. Het heeft geen gevolgen voor de werking van het commando.

OFF
Onderdrukt de weergave van het recordnummer. In het geval van Paradox- of SQL-tabellen worden geen recordnummers getoond.

[TO FILE <*bestandsnaam*> | ? | <*bestandsnaamfilter*>] | [TO PRINTER]
Stuurt uitvoer naar een bestand of een printer. TO FILE stuurt uitvoer zowel naar het tekstbestand <*bestandsnaam*>, als naar het commandovenster (tenzij SET CONSOLE is uitgeschakeld (OFF)). Standaard wordt aan <*bestandsnaam*> de extensie .TXT toegekend en wordt het bestand opgeslagen in de huidige directory. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u de naam opgeeft van het doelbestand en van de directory waar het moet worden opgeslagen. TO PRINTER stuurt uitvoer naar de printer en tevens naar het resultatenpaneel van het commandovenster.

Beschrijving
Met DISPLAY kunt u een of meer records uit de huidige tabel tonen in het resultatenpaneel van het commandovenster. Als u DISPLAY geeft zonder argumenten, wordt alleen het huidige record getoond, terwijl LIST alle records toont. Als SET HEADINGS is uitgeschakeld (OFF), worden geen veldnamen getoond als u DISPLAY gebruikt.

DISPLAY pauzeert als het resultatenpaneel vol is en toont dan een dialoogvenster waarin u wordt gevraagd of u verder wilt gaan. U kunt op de volgende manieren door de uitvoer bladeren:

- Druk op *Enter* of *PgDn* om een venster omhoog te gaan.
- Klik onder aan de verticale schuifbalk om een venster omlaag te gaan.
- Druk op *PgUp* om een venster omlaag te gaan.
- Klik boven aan de verticale schuifbalk om een venster omhoog te gaan.
- Druk op de schuifpijltjes op de verticale schuifbalk om één regel omhoog of omlaag te gaan.
- Druk op Pijl-omhoog of Pijl-omlaag om één regel omhoog of omlaag te gaan.
- Druk op de *Spatiebalk* om één regel omhoog te gaan.
- Typ *<Nuitdr>* en druk op *Enter* om het opgegeven aantal regels omhoog te gaan.

DISPLAY komt overeen met LIST, maar LIST pauzeert niet na elk volle venster met informatie. List toont alle informatie achter elkaar.

Als de informatie in het resultatenpaneel meer is dan de buffer van dBASE kan bevatten, kunt u niet meer omhoog bladeren naar de eerste regel. Gebruik de opties TO FILE of TO PRINTER om de informatie op te slaan in een bestand of af te drukken.

Voorbeeld

In het volgende voorbeeld wordt DISPLAY gebruikt om bepaalde gegevens uit de tabel Afnemers te tonen:

```
USE Afnemers
DISPLAY
* Alle velden in het huidige record weergeven

DISPLAY ALL
* Alle velden in alle records weergeven. Na voltooiing
* staat de recordaanwijzer aan het eind van de tabel.

GOTO 2      && Naar het tweede record in de tabel
DISPLAY REST FIELDS Bedrijf, Contact
* Velden Bedrijf en Contact in de rest van de tabel weergeven

SET FIELDS TO Bedrijf, Contact, Telefoon
DISPLAY all
* Commando's SET FIELDS en DISPLAY combineren
* om velden Bedrijf, Contact en Telefoon in
* alle records te tonen.
```

Zie ook

DISPLAY STRUCTURE, LIST, SET HEADINGS, SET FIELDS

DISPLAY COVERAGE

Foutafhandeling en testen op fouten

Toont in het resultatenpaneel van het commandovenster de inhoud van een dekkingsbestand (.COV-bestand).

Syntaxis

DISPLAY COVERAGE
  <.COV-*bestandsnaam*> | ? | <*bestandsnaamfilter*>
  [ALL]
  [SUMMARY]
  [TO FILE <*bestandsnaam*> | ?] | <*bestandsnaamfilter*> | [TO PRINTER]

<.COV-*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Het dekkingsbestand dat dBASE maakt of wijzigt als SET COVERAGE is ingeschakeld (ON), en u een programma aanroept of een procedure of door de gebruiker gedefinieerde functie aanroept in een procedurebestand. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een dekkingsbestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, wordt de extensie .COV gebruikt.

ALL

Toont de volgende informatie in het resultatenpaneel van het commandovenster:

- De inhoud van het dekkingsbestand voor het programma van <*bestandsnaam1*> en de dekkingsbestanden voor elk uitgevoerd programma- of procedurebestand dat door het programma van <*bestandsnaam1*> is aangeroepen
- Het totaal aantal logische blokken dat in alle programmabestanden is uitgevoerd
- Het percentage logische blokken dat in alle programmabestanden is uitgevoerd

SUMMARY

Als ALL niet is opgegeven, wordt over het programma van <*bestandsnaam1*>' de volgende informatie getoond. Als ALL wel is opgegeven, wordt de volgende informatie getoond voor het programma van <*bestandsnaam1*> en alle programma's die door het programma van <*bestandsnaam1*> zijn aangeroepen:

- De logische blokken die *niet* zijn uitgevoerd
- Het totaal aantal logische blokken dat is uitgevoerd
- Het percentage logische blokken dat is uitgevoerd

TO FILE <*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Stuurt de uitvoer naar <*bestandsnaam2*> en naar het resultatenpaneel van het commandovenster. Standaard wordt aan <*bestandsnaam2*> de extensie .TXT toegekend en wordt het bestand opgeslagen in de huidige directory. De opties ? en

<bestandsnaamfilter> tonen een dialoogvenster waarin u de naam opgeeft van het doelbestand en van de directory waar het moet worden opgeslagen.

TO PRINTER

Stuurt uitvoer naar de printer en naar het resultatenpaneel van het commandovenster.

Beschrijving

Een dekkingsbestand bevat de resultaten van een dekkingsanalyse van een programma, procedure of door de gebruiker gedefinieerde functie. Met SET COVERAGE ON kunt u een programma, procedure of door de gebruiker gedefinieerde functie laten analyseren en de resultaten van die analyse opslaan in een dekkingsbestand. Het commando DISPLAY COVERAGE kunt u gebruiken in een programmabestand of in het commandovenster.

DISPLAY COVERAGE toont de inhoud van *<bestandsnaam1>* en enkele percentages die het resultaat zijn van berekeningen op de inhoud van *<bestandsnaam1>*. De uitvoer, die begint bij het dekkingsbestand van het hoofdprogramma, verschijnt in het resultatenpaneel van het commandovenster. DISPLAY COVERAGE toont het eerste venster vol informatie en pauzeert dan. Zie de beschrijving van DISPLAY voor informatie over manoeuvreren in het venster.

De beschikbare hoeveelheid geheugen bepaalt de grootte van de buffer voor het resultatenpaneel van het commandovenster. Als er meer uitvoer is dan in de buffer past, kunt u niet meer terug bladeren naar de eerste regels. Gebruik de opties TO FILE of TO PRINTER om de informatie op te slaan in een bestand of af te drukken.

DISPLAY COVERAGE komt overeen met LIST COVERAGE, maar LIST COVERAGE pauzeert niet na elk venster informatie. U kunt echter wel, net als bij DISPLAY COVERAGE, door de uitvoer van LIST COVERAGE bladeren.

Voorbeeld

Zie SET COVERAGE voor een voorbeeld met DISPLAY COVERAGE.

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

#pragma, DISPLAY, LIST, SET COVERAGE

DISPLAY FILES

Stations- en bestandsfuncties

Toont in het resultatenpaneel van het commandovenster informatie over bestanden op schijf.

Syntaxis

DISPLAY FILES
[[LIKE] <*bestandsnaam1*> | <*bestandsnaamfilter*>]
[ON <*station*>]
[TO FILE <*bestandsnaam2*> | ? | <*bestandsnaamfilter*>] |
[TO PRINTER]

[LIKE] <*bestandsnaam1*> | <*bestandsnaamfilter*>

Het bestand waarover u informatie wilt weergeven. DISPLAY FILES <*bestandsnaamfilter*> toont de namen van alle bestanden die overeenkomen met een globale bestandsnaam met jokertekens (* en ?). Als u een bestand opgeeft en dat bestand staat niet in de huidige directory of in het pad dat is ingesteld met SET PATH, moet <*bestandsnaam*> een station en een pad bevatten. LIKE is optioneel en heeft geen gevolgen voor de werking van het commando. Het sleutelwoord is alleen beschikbaar voor de leesbaarheid.

ON <*station*>

Het station met de bestanden waarover u informatie wilt tonen. Standaard wordt het bestand of de bestanden gezocht zoekt op het actieve station.

TO FILE <*bestandsnaam2*> | ? | <*bestandsnaamfilter*>

Stuurt uitvoer naar het tekstbestand <*bestandsnaam2*> en naar het resultatenpaneel van het commandovenster. Standaard wordt aan <*bestandsnaam2*> de extensie .TXT toegekend en wordt het bestand opgeslagen in de huidige directory. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u de naam en directory voor het doelbestand kunt opgeven.

TO PRINTER

Stuurt uitvoer naar de printer en naar het resultatenpaneel van het commandovenster.

Beschrijving

DISPLAY FILES is een commando waarmee u een directorylijst kunt maken. De informatie over elk bestand bestaat uit de naam, de grootte in bytes en de datum van de laatste wijziging. DIR laat tevens het totaal aantal bytes zien dat door de getoonde bestanden in beslag wordt genomen, plus het aantal bytes dat nog op schijf beschikbaar is en de totale schijfruimte.

DISPLAY FILES zonder argumenten geeft alleen informatie over tabelbestanden (met de extensie .DBF) in de huidige directory. Naast de informatie die gewoonlijk wordt weergegeven, wordt dan tevens het aantal records in elke database getoond.

U mag jokertekens (* en ?) gebruiken om meerdere bestanden met overeenkomstige namen op te geven. Hoewel DISPLAY FILES * onderling uitwisselbaar is met DISPLAY FILES zonder argumenten (er worden alle .DBF-bestanden getoond), toont DISPLAY FILES N* alle .DBF-bestanden waarvan de naam begint met N. DISPLAY FILES N*.* toont alle bestanden waarvan de naam begint met N, onafhankelijk van de extensie. DISPLAY FILES N???? toont alle .DBF-bestanden waarvan de naam uit vijf tekens bestaat en begint met N.

DISPLAY FILES pauzeert nadat een vol venster met informatie is getoond. Zie de beschrijving van DISPLAY voor informatie over het manoeuvreren in het venster.

Als er meer uitvoer is dan in de buffer past, kunt u niet meer terug bladeren naar de eerste regels. Gebruik de opties TO FILE of TO PRINTER om de informatie op te slaan in een bestand of af te drukken.

DISPLAY FILES komt overeen met LIST FILES, maar LIST FILES pauzeert niet na elk vol venster met informatie. U kunt echter wel, net als bij DISPLAY FILES, door de uitvoer van LIST FILES bladeren.

De instellingen voor SET DATABASE en SET DBTYPE zijn van invloed op de uitvoer van DISPLAY FILES en LIST FILES.

Voorbeeld

In de volgende voorbeelden wordt DISPLAY FILES gebruikt:

```
DISPLAY FILES
* Alle tabellen in huidige subdirectory
DISPLAY FILES ON B: && Alle tabellen op station B
DISPLAY FILES A:    && Alle tabellen op station B
DISPLAY FILES *.prg && Alle programmabestanden
DISPLAY FILES LIKE *.* ON C:\DBASEWIN
* Alle bestanden in C:\DBASEWIN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. De opties ON <*station*> en TO FILE ? worden in dBASE IV niet ondersteund.

Als u in dBASE IV TO FILE <*bestandsnaam2*> gebruikt en geen extensie opgeeft, wordt de extensie .PRT gebruikt. In dBASE voor Windows wordt echter de extensie .TXT gebruikt.

Zie ook

DIR, DISPLAY, FILE( ), LIST, SET DATABASE, SET DBTYPE, SET DEFAULT

DISPLAY MEMORY

Omgeving

Toont in het resultatenpaneel van het commandovenster informatie over geheugenvariabelen.

Syntaxis

DISPLAY MEMORY
[TO FILE <*bestandsnaam*> | ? | <*bestandsnaamfilter*>] | [TO PRINTER]

TO FILE <*bestandsnaam*> | ? | <*bestandsnaamfilter*>
Stuurt uitvoer naar het tekstbestand <*bestandsnaam*> (het zogenaamde doelbestand) en naar het resultatenpaneel van het commandovenster. Standaard wordt voor <*bestandsnaam*> de extensie .TXT gebruikt en wordt het bestand opgeslagen in de huidige directory. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u de naam opgeeft van het doelbestand en van de directory waar het moet worden opgeslagen.

TO PRINTER
Stuurt uitvoer naar de printer en naar het resultatenpaneel van het commandovenster.

Beschrijving

Gebruik DISPLAY MEMORY om een lijst te tonen met de inhoud en grootte van geheugenvariabelen. Als u de toets *F7* geen andere functie hebt gegeven met ON KEY of SET FUNCTION, kunt u ook op *F7* drukken om DISPLAY MEMORY uit te voeren.

DISPLAY MEMORY toont informatie over eigen geheugenvariabelen en systeemgeheugenvariabelen. Over eigen (door de gebruiker gedefinieerde) geheugenvariabelen wordt de volgende informatie getoond:

- Naam
- Bereik (gedefinieerd met PUBLIC, PRIVATE, LOCAL of STATIC of verborgen geheugenvariabelen)
- Gegevenstype
- Waarde
- Aantal actieve geheugenvariabelen
- Aantal beschikbare geheugenvariabelen
- Hoeveelheid geheugen (in bytes) dat door tekenvariabelen in beslag wordt genomen
- Hoeveelheid geheugen (in bytes) dat nog voor tekenvariabelen beschikbaar is
- Naam van het programma dat geheugenvariabelen heeft geïnitialiseerd met PRIVATE

Over systeemgeheugenvariabelen wordt de volgende informatie getoond:

- Naam
- Bereik (gedefinieerd met PUBLIC of PRIVATE of verborgen geheugenvariabelen)
- Gegevenstype
- Huidige waarde

DISPLAY MEMORY toont een venster vol informatie en pauzeert dan. Zie de beschrijving van DISPLAY voor informatie over het manoeuvreren in het venster.

De beschikbare hoeveelheid geheugen bepaalt de grootte van de buffer voor het resultatenpaneel van het commandovenster. Als er meer uitvoer is dan in de buffer past,

kunt u niet meer terug bladeren naar de eerste regels. Gebruik de opties TO FILE of TO PRINTER om de informatie op te slaan in een bestand of af te drukken.

Als SET SAFETY is ingeschakeld (ON) en er bestaat een bestand met dezelfde naam als het doelbestand, verschijnt een dialoogvenster waarin wordt gevraagd of het bestand moet worden overschreven. Als SET SAFETY is uitgeschakeld (OFF), wordt een bestand met dezelfde naam zonder waarschuwing overschreven.

DISPLAY MEMORY komt overeen met LIST MEMORY, maar LIST MEMORY pauzeert niet na elk vol scherm. U kunt echter wel, net als bij DISPLAY MEMORY, door de uitvoer van LIST MEMORY bladeren.

Overdraagbaarheid

In dBASE III PLUS wordt alleen de optie TO PRINT ondersteund. De opties ? en *<bestandsnaamfilter>* worden niet ondersteund in dBASE IV. Als u bij de optie TO FILE *<bestandsnaam>* geen extensie opgeeft, wordt in dBASE IV de extensie .PRT toegevoegd in plaats van .TXT.

Zie ook

LIST, LOCAL, PRIVATE, PUBLIC, SET FUNCTION, SET SAFETY, ON KEY, STATIC, STORE

DISPLAY STATUS

Omgeving

Toont informatie over de huidige dBASE-omgeving in het resultatenpaneel van het commandovenster.

Syntaxis

DISPLAY STATUS
[TO FILE *<bestandsnaam>* | ? | *<bestandsnaamfilter>*] | [TO PRINTER]

TO FILE *<bestandsnaam>* | ? | *<bestandsnaamfilter>*

Stuurt uitvoer naar het tekstbestand *<bestandsnaam>* (het zogenaamde doelbestand) en naar het resultatenpaneel van het commandovenster. dBASE kent standaard de extensie .TXT toe aan *<bestandsnaam>* en slaat het bestand op in de huidige directory. De opties ? en *<bestandsnaamfilter>* tonen een dialoogvenster waarin u de naam opgeeft van het doelbestand en van de directory waar het moet worden opgeslagen.

TO PRINTER

stuurt uitvoer naar de printer en naar het resultatenpaneel van het commandovenster.

Beschrijving

Met DISPLAY STATUS kunt u bepalen welke tabellen en indexbestanden zijn geopend en de instellingen van SET-commando's bepalen. DISPLAY STATUS toont alleen informatie die betrekking heeft op de huidige sessie.

Als u de toets *F6* geen andere functie hebt gegeven met ON KEY of SET FUNCTION, kunt u ook op *F6* drukken om DISPLAY STATUS uit te voeren.

DISPLAY STATUS toont de volgende informatie:

- Naam en alias van geopende tabellen in elk werkgebied
- Namen van alle geopende memobestanden in elk werkgebied
- Naam van een indelingsbestand in elk werkgebied
- Namen van alle geopende indexen en hun indexsleuteluitdrukkingen in elk werkgebied
- Instelling van SET ORDER in elk werkgebied
- Hoofdindex, als die er is, in elk werkgebied
- Relaties tussen databases in elk werkgebied
- Filtervoorwaarden in elk werkgebied
- Instelling van SET PATH (zoekpad voor bestanden)
- Instelling van SET DEFAULT
- Huidige werkgebied
- Instelling van SET PRINTER of SET DEVICE
- Instelling van DBTYPE
- Numerieke instellingen van SET MARGIN, SET DECIMALS, SET MEMOWIDTH, SET HISTORY, SET TYPEAHEAD, SET ODOMETER, SET REFRESH en SET REPROCESS
- Instellingen van ON KEY, ON ESCAPE en ON ERROR
- ON/OFF-instellingen van SET-commando's
- Instellingen van programmeerbare functietoetsen en SET FUNCTION

DISPLAY STATUS toont een venster vol informatie en pauzeert dan. Zie de beschrijving van DISPLAY voor informatie over het manoeuvreren in het venster.

De hoeveelheid beschikbaar geheugen bepaalt de grootte van de buffer voor het resultatenpaneel van het commandovenster. Als er meer uitvoer is dan in de buffer past, kunt u niet meer terug bladeren naar de eerste regels. Gebruik de opties TO FILE of TO PRINTER om de informatie op te slaan in een bestand of af te drukken.

Als SET SAFETY is ingeschakeld (ON) en er bestaat een bestand met dezelfde naam als het doelbestand, verschijnt een dialoogvenster waarin wordt gevraagd of u het bestand wilt overschrijven. Als SET SAFETY is uitgeschakeld (OFF), wordt een bestand met dezelfde naam zonder waarschuwing overschreven.

DISPLAY STATUS komt overeen met LIST STATUS, maar LIST STATUS pauzeert niet na elk venster vol informatie. U kunt echter wel, net als bij DISPLAY STATUS, door de uitvoer van LIST STATUS bladeren.

Overdraagbaarheid

In dBASE III PLUS wordt alleen de optie TO PRINT ondersteund. De opties ? en *<bestandsnaamfilter>* worden niet ondersteund in dBASE IV. Als u bij de optie TO FILE *<bestandsnaam>* geen extensie opgeeft, wordt in dBASE IV de extensie .PRT toegevoegd in plaats van .TXT.

Zie ook

LIST, SET( ), SETTO( ), SET SAFETY

DISPLAY STRUCTURE

Tabellen

Toont de velddefinities voor de opgegeven tabel.

Syntaxis

DISPLAY STRUCTURE
[IN *<alias>*]
[TO FILE *<bestandsnaam>* | ? | *<bestandsnaamfilter>*] | [TO PRINTER]

IN *<alias>*

Geeft het werkgebied aan van de open tabel waarvan u de structuur wilt tonen. U kunt een werkgebiednummer (1 tot en met 255), -letter (A tot en met J) of een aliasnaam opgeven. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

[TO FILE *<bestandsnaam>* | ? | *<bestandsnaamfilter>*] | [TO PRINTER]

Stuurt uitvoer naar een bestand of een printer. TO FILE stuurt uitvoer naar het tekstbestand *<bestandsnaam>* en naar het commandovenster. Standaard wordt aan *<bestandsnaam>* de extensie .TXT toegewezen en wordt het bestand opgeslagen in de huidige directory. De opties ? en *<bestandsnaamfilter>* tonen een dialoogvenster waarin u de naam opgeeft van het doelbestand en van de directory waar het moet worden opgeslagen. TO PRINTER stuurt uitvoer naar de printer en tevens naar het resultatenpaneel van het commandovenster.

Beschrijving

Met DISPLAY STRUCTURE kunt u de structuur van de huidige of een opgegeven tabel tonen in het resultatenpaneel van het commandovenster. DISPLAY STRUCTURE toont de volgende informatie over de huidige of opgegeven tabel:

- Naam van de tabel
- Type van de tabel (Paradox, dBASE of SQL)

- Aantal records
- Datum van laatste wijziging
- Velden
- Veldnummer
- Veldnaam (als SET FIELDS is ingeschakeld (ON), verschijnt naast elk veld dat is opgegeven met het commando SET FIELDS TO het groter-dan-symbool, >)
- Type
- Lengte
- Aantal bytes per record (de som van de lengte van alle velden plus één byte die is gereserveerd voor de asterisk waarmee een record kan worden gemarkeerd voor verwijdering)

De grootte van een dBASE-tabel (exclusief de header van het tabelbestand) bepaalt u door het aantal bytes per record te vermenigvuldigen met het aantal records in de tabel.

DISPLAY STRUCTURE pauzeert als het resultatenpaneel van het commandovenster vol is en toont dan een dialoogvenster met de vraag of u de weergave wilt voortzetten. Zie de beschrijving van DISPLAY voor informatie over het manoeuvreren in het venster.

De hoeveelheid beschikbaar geheugen bepaalt de grootte van de buffer voor het resultatenpaneel van het commandovenster. Als er meer uitvoer is dan in de buffer past, kunt u niet meer terug bladeren naar de eerste regels. Gebruik de opties TO FILE of TO PRINTER om de informatie op te slaan in een bestand of af te drukken. U kunt de uitvoer naar een bestand of naar de printer sturen, maar niet naar beide tegelijk.

DISPLAY STRUCTURE komt overeen met LIST STRUCTURE, maar LIST STRUCTURE pauzeert niet na elk venster vol informatie.

DISPLAY STRUCTURE en LIST STRUCTURE staan geen van beide wijziging van een bestaande tabel toe. De structuur van een tabel kunt u wijzigen met MODIFY STRUCTURE.

Voorbeeld

In het volgende voorbeeld worden DISPLAY STRUCTURE en LIST STRUCTURE gebruikt om de structuur van een tabel te tonen:

```
USE Bedrijf
LIST STRUCTURE TO FILE Bedrijf.TXT
```

De volgende tekst wordt opgeslagen in een tekstbestand BEDRIJF.TXT:

```
Structuur voor tabel  C:\DBASEWIN\VOORBD\BEDRIJF.DBF
Type tabel            DBASE
Aantal records        9
Laatst bijgewerkt     29-10-94
---------------------------------------------------------------
Veld   Veldnaam                 Type          Lengte  Dec  Index
    1  BEDRIJF                  CHARACTER         35          J
    2  STRAAT1                  CHARACTER         35          N
```

```
  3  STRAAT2                  CHARACTER        35          N
  4  PLAATS                   CHARACTER        20          N
  5  REGIO                    CHARACTER        15          N
  6  POSTCODE                 CHARACTER        10          N
  7  TELEFOON                 CHARACTER        14          N
  8  TYPE                     CHARACTER         3          N
  9  COMPCODE                 CHARACTER         4          N
 10  NOTITIES                 MEMO             10          N
 11  VERKTOTNU                NUMERIC          14    2     N
 12  ORDOPBR                  NUMERIC           8          N
 13  TERMKLAS                 CHARACTER         3          N
 14  HOOFDKANT                LOGICAL           1          N
 15  BTWNUMMER                CHARACTER        14          N
** Totaal **                                   222
```

Zie ook

DISPLAY, LIST, MODIFY STRUCTURE, SET FIELDS

DMY( )

Datum- en tijdgegevens

Geeft voor een opgegeven datumuitdrukking een tekenreeks met de opmaak DD MAAND JJ of DD MAAND JJJJ als resultaat.

Syntaxis

DMY(<*Duitdr*>)

<Duitdr>

De datumuitdrukking die u wilt omzetten naar een tekenreeks met de opmaak DD MAAND JJ of DD MAAND JJJJ.

Beschrijving

DMY( ) geeft een tekenreeks met de datum als resultaat. De datum heeft de opmaak DD MAAND JJ of DD MAAND JJJJ, waarbij DD het nummer van de dag is, MAAND de volledige naam van de maand en JJ het nummer van het jaar. Als SET CENTURY is uitgeschakeld (OFF, de standaardinstelling), geeft DMY( ) het jaar in de vorm van twee cijfers. Als SET CENTURY is ingeschakeld (ON), wordt het jaar in de vorm van vier cijfers als resultaat gegeven.

Geef <*Duitdr*> op in de huidige datumopmaak die wordt bepaald door SET DATE, DBASEWIN.INI of de optie Internationaal in het Configuratiescherm van Windows (in die volgorde). Dat wil zeggen, de instellingen bij SET DATE hebben een hogere prioriteit dan die in DBASEWIN.INI, en de instellingen in DBASEWIN.INI hebben weer een hogere prioriteit dan de instellingen in het Configuratiescherm van Windows. Let er op dat <*Duitdr*> overeenkomt met de datumopmaak die wordt gebruikt op het moment dat het programma wordt uitgevoerd.

Als u een ongeldige datum doorgeeft aan DMY( ), wordt die eerst omgezet in een geldige datum en vervolgens in een tekenreeks met de opmaak DD MAAND JJ of DD MAAND JJJJ. Als u een lege datumuitdrukking of een andere uitdrukking dan een datumuitdrukking tussen accolades ({ }) doorgeeft aan DMY( ), wordt "0 Onbekend 00" of "0 Onbekend 0000" als resultaat gegeven. Als u een andere dan een datumuitdrukking of een uitdrukking die niet tussen accolades staat doorgeeft aan DMY( ), wordt een fout als resultaat gegeven.

Voorbeeld

In het volgende voorbeeld wordt DMY( ) gebruikt om de waarde van een datumveld weer te geven in de vorm: nummer van de dag, volledige maand in letters, jaar in cijfers volgens de instelling van SET CENTURY:

```
SET TALK OFF
USE Afnemers
Teller=1
DO WHILE .NOT. EOF()
CLEAR
? CENTER("Rapport over database Afnemers",80)
? CENTER("Gemaakt op " + CDOW(DATE() ) + ;
      ", " + DMY(DATE() ),80,"-")
?
? "Bedrijf" AT 2, "Telefoon" AT 40, ;
    "Eerste contact" AT 55
DO WHILE Teller <= 10 .AND. .NOT. EOF()
  ? Bedrijf AT 2, Telefoon AT 40, CDOW(Balnsdatum)+", ";
    +DMY(Balnsdatum) AT 55
  SKIP
  Teller=Teller+1
ENDDO
Teller=1
WAIT
ENDDO
CLOSE ALL
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CDOW( ), CMONTH( ), DAY( ), DOW( ), MDY( ), MONTH( ), SET CENTURY, SET DATE, YEAR( )

DO — Programma's

Voert een programma, procedure of door de gebruiker gedefinieerde functie uit.

Syntaxis

DO <*bestandsnaam*> | ? | <*bestandsnaamfilter*> |
<*procedurenaam*> |
<*Naam door gebruiker gedefinieerde functie*>
[WITH <*parameterlijst*>]

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Het programmabestand dat u wilt uitvoeren. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand eerst in de huidige directory gezocht en dan volgens de *zoekvolgorde* in het *zoekpad*. Zie "Zoekpad en -volgorde" verderop in deze sectie voor meer informatie.

Als u een bestand zonder extensie opgeeft, wordt de extensie .PRO gebruikt (een gecompileerd objectbestand). Als het bestand niet wordt aangetroffen, wordt naar een .PRG-bestand gezocht (een bronbestand). Als dat wel wordt gevonden, wordt het eerste gecompileerd. Standaard wordt het .PRO-bestand gemaakt in dezelfde directory als het .PRG-bestand. Dat hoeft niet de huidige directory te zijn.

<*procedurenaam*> | <*Naam door gebruiker gedefinieerde functie*>

De procedure of door de gebruiker gedefinieerde functie in een geopend programmabestand die u wilt uitvoeren. De procedure of door de gebruiker gedefinieerde functie moet staan in een programmabestand dat de procedure of functie aanroept via het commando DO, of in een afzonderlijk geopend bestand in het zoekpad. Het zoekpad wordt verderop in deze sectie behandeld.

WITH <*parameterlijst*>

Geeft de geheugenvariabelewaarden, veldwaarden of andere geldige uitdrukkingen die als parameters moeten worden doorgegeven aan een programma, procedure of door de gebruiker gedefinieerde functie. Zie de beschrijving van PROCEDURE voor meer informatie over het doorgeven van parameters.

Beschrijving

Met DO voert u programma's uit vanuit het commandovenster of procedures en andere programma's vanuit een programma. Als u in het commandovenster DO geeft, keert de besturing daar terug na voltooiing van het programma, de procedure of de door de gebruiker gedefinieerde functie. Als u DO toepast in een programma om een ander programma of een procedure of door de gebruiker gedefinieerde functie uit te voeren, wordt de besturing na voltooiing teruggegeven aan de regel volgend op de regel met de DO-instructie.

De grens aan het aantal geneste DO-aanroepen is afhankelijk van de beschikbare hoeveelheid geheugen en het aantal geopende bestanden (een DO-instructie kan een bestand openen). Voorkom recursieve aanroepen met DO, want dan bereikt u snel het maximale aantal geneste DO-aanroepen.

Als in een programmabestand een DO-instructie wordt aangetroffen, wordt de procedure of door de gebruiker gedefinieerde functie met de opgegeven naam in dat

bestand gezocht. Als het huidige programmabestand zowel een procedure als een door de gebruiker gedefinieerde functie met de opgegeven naam bevat, wordt de procedure of functie uitgevoerd die als eerste is gedefinieerd. Als in het programmabestand geen PROCEDURE- of FUNCTION-definitie met de opgegeven naam wordt aangetroffen, wordt naar een programma, procedure of door de gebruiker gedefinieerde functie met de opgegeven naam gezocht in het *zoekpad* en wel in de *zoekvolgorde*.

Zoekpad en -volgorde

Als de naam die u bij DO hebt opgegeven geen pad of extensie bevat, kan die naam betrekking hebben op een programma, een procedure, een door de gebruiker gedefinieerde functie of een bestand. Dit probleem wordt opgelost door de naam te zoeken op een bepaalde plaats (het zoekpad) en in een bepaalde volgorde (de zoekvolgorde). Vervolgens wordt het eerste programma of de eerste subroutine met de opgegeven naam uitgevoerd. Het zoekpad en de zoekvolgorde zijn als volgt:

1. Het objectbestand (.PRO) van het programma dat wordt uitgevoerd
2. Andere geopende objectbestanden (.PRO) in de *aanroepreeks*, het laatste geopende bestand eerst
3. Het bestand dat is opgegeven met SYSPROC = <*bestandsnaam*> in DBASEWIN.INI
4. Alle bestanden die zijn geopend met SET PROCEDURE, SET PROCEDURE...ADDITIVE of SET LIBRARY, in de volgorde waarin ze zijn geopend
5. Het objectbestand (.PRO) met de opgegeven naam in het zoekpad
6. Het programmabestand (.PRG) met de opgegeven naam in het zoekpad. Dat wordt dan automatisch gecompileerd.

Voorbeeld

In het volgende voorbeeld wordt DO gebruikt om procedures uit te voeren:

```
DO Klant_Rpt
DO Verkoop_Rpt WITH "NW"
DO RecToevoegen WITH "van Herewaarden","Pieter",25

PROC Klant_Rpt
* ...
RETURN

PROC Verkoop_Rpt
Parameters Regio
* ...
RETURN

PROC RecToevoegen
PARAMETERS Achternaam, Voornaam, Leeftijd
* ...
RETURN
```

Overdraagbaarheid

In dBASE IV of dBASE III PLUS worden geen aanroepen met DO naar door de gebruiker gedefinieerde functies ondersteund. Het zoekpad en de zoekvolgorde werken in dBASE IV en dBASE III PLUS anders dan in dBASE voor Windows. Zowel dBASE IV als dBASE III PLUS maken standaard gecompileerde objectbestanden met de extensie .DBO en slaan die op in de huidige directory.

Zie ook

!, CLEAR PROGRAM, COMPILE, DOS, PROCEDURE, RETURN, RUN, RUN( ), SET DEVELOPMENT, SET ESCAPE, SET LIBRARY, SET PROCEDURE

DO CASE

Programma's

Voert voorwaardelijk instructies uit door een of meer voorwaarden te evalueren en de instructies uit te voeren die volgen op de eerste voorwaarde die waar is.

Syntaxis

```
DO CASE
      CASE <voorwaarde Luitdr1>
         <instructies>
      [CASE <voorwaarde Luitdr2>
         <instructies>...]
      [OTHERWISE
         <instructies>]
   ENDCASE
```

CASE *<voorwaarde Luitdr>*

Als de voorwaarde waar is, wordt de reeks instructies uitgevoerd die staan tussen CASE en het volgende CASE-commando of het commando OTHERWISE of ENDCASE. Als die instructies zijn uitgevoerd, wordt de besturing doorgegeven aan de regel na ENDCASE. Als de voorwaarde onwaar is, wordt de besturing doorgegeven aan het volgende CASE-commando, aan OTHERWISE of aan ENDCASE.

<instructies>

Eén of meer programmaregels met een willekeurige combinatie van commando's, functies en door de gebruiker gedefinieerde functies.

OTHERWISE

Voert een reeks instructies uit als alle CASE-voorwaarden onwaar opleveren.

ENDCASE

Een verplicht commando dat het eind aangeeft van de DO CASE-structuur.

Beschrijving

DO CASE komt overeen met IF...ELSE...ENDIF. Net als IF-voorwaarden worden DO CASE-voorwaarden geëvalueerd in de volgorde waarin ze binnen de structuur zijn opgenomen. In het geval van DO CASE worden echter alleen de instructies uitgevoerd die volgen op de eerste voorwaarde die waar is, zelfs als meer voorwaarden evalueren tot waar. Gebruik DO CASE in plaats van een reeks IF-instructies als u alleen de instructies bij één voorwaarde wilt uitvoeren.

Verder kunt u DO CASE gebruiken als u een reeks uitzonderingen op een voorwaarde wilt programmeren. De instructies met CASE *<voorwaarde>* stellen de uitzonderingen voor en de instructie met OTHERWISE de algemene situatie.

Een reeks instructies met DO CASE wordt als volgt uitgevoerd:

- Elke CASE-voorwaarde wordt geëvalueerd tot er een waar is.
- De instructies tussen de eerste CASE-voorwaarde die waar is en de volgende instructie met CASE, OTHERWISE of ENDCASE worden uitgevoerd.
- De DO CASE-structuur wordt verlaten zonder de overige CASE-voorwaarden te evalueren.
- De besturing wordt doorgegeven aan de eerste regel na het commando ENDCASE.

Als geen van de voorwaarden waar is, worden de instructies na OTHERWISE uitgevoerd. Als er geen instructie met OTHERWISE is opgenomen, wordt de structuur verlaten zonder dat er verder instructies binnen de structuur worden uitgevoerd en wordt de besturing doorgegeven aan de eerste regel na het commando ENDCASE.

U kunt structuren als IF...ENDIF en DO WHILE...ENDDO alleen opnemen binnen afzonderlijke DO CASE-instructies. Verder is het mogelijk DO CASE-structuren te nesten.

Voorbeeld

Vergelijk dit voorbeeld met de voorbeelden bij IF. De volgende CASE-constructie bepaalt de orde van grootte van een variabele en toont een toepasselijke melding:

```
nWaarde = 225
DO CASE
  CASE nWaarde > 1000
    ? "Waarde is groter dan 1000."
  CASE nWaarde > 100
    ? "Waarde is groter dan 100."
  CASE nWaarde > 10
    ? "Waarde is groter dan 10."
  CASE nWaarde > 1
    ? "Waarde is groter dan 1."
  OTHERWISE
    ? "Waarde is kleiner dan of gelijk aan 1."
ENDCASE
```

Zie ook

DO WHILE, DO...UNTIL, FOR...NEXT, IF, IIF( ), SCAN

DO WHILE

Voert de instructies tussen DO WHILE en ENDDO uit zolang de opgegeven voorwaarde waar is of tot het commando EXIT wordt aangetroffen.

Syntaxis

```
DO WHILE <voorwaarde Luitdr>
      <instructies>
      [LOOP]
      [EXIT]
   ENDDO
```

<voorwaarde Luitdr>

Een logische uitdrukking die bepaalt of de instructies na DO WHILE moeten worden uitgevoerd. Als de uitdrukking evalueert tot .F., worden de instructies na DO WHILE overgeslagen en wordt de besturing doorgegeven aan de regel na ENDDO. Als de uitdrukking evalueert tot .T., worden de instructies binnen de DO WHILE-lus uitgevoerd.

<instructies>

Programmaregels met een willekeurige combinatie van commando's, functies, door de gebruiker gedefinieerde functies en LOOP- en EXIT-opties.

LOOP

Geeft de besturing opnieuw door aan het begin van de DO WHILE-lus zonder de instructies tussen LOOP en ENDDO uit te voeren.

EXIT

Geeft de besturing door aan de regel na ENDDO zonder de instructies tussen EXIT en ENDDO uit te voeren.

ENDDO

Een verplicht commando dat het eind aangeeft van de DO WHILE-lus.

Beschrijving

Met DO WHILE...ENDDO kunt u de opgegeven instructies onbeperkt uitvoeren zolang de opgegeven voorwaarde waar is. Als een instructie met DO WHILE wordt aangetroffen, wordt eerst bepaald of *<voorwaarde Luitdr>* waar of onwaar is. Als *<voorwaarde Luitdr>* waar is, worden de instructies binnen de DO WHILE-lus een voor een uitgevoerd tot ENDDO, LOOP of EXIT wordt aangetroffen. Als *<voorwaarde Luitdr>* onwaar is, wordt de lus overgeslagen en wordt de besturing doorgegeven aan de regel na ENDDO.

ENDDO en LOOP gegeven de besturing terug aan de instructie met DO WHILE. In dat geval wordt *<voorwaarde Luitdr>* opnieuw geëvalueerd. EXIT zorgt dat de DO WHILE-lus wordt verlaten en de besturing wordt doorgegeven aan de regel na ENDDO.

U kunt andere lussen en structuren nesten binnen een DO WHILE-lus, inclusief andere DO WHILE-lussen. Maximaal mogen 125 DO WHILE-lussen in een procedure of functie worden genest. Het maximum geldt voor alle actieve lussen.

DO WHILE...ENDDO en DO...UNTIL zijn tegengestelde constructies. DO WHILE...ENDDO voert instructies uit zolang een voorwaarde waar is (tot de voorwaarde onwaar wordt), terwijl DO...UNTIL instructies uitvoert zolang een voorwaarde onwaar is (tot de voorwaarde waar wordt). Omdat DO...UNTIL de instructies eerst uitvoert en daarna de voorwaarde evalueert, worden de instructies tussen DO en UNTIL tenminste één keer uitgevoerd. Bij DO WHILE...ENDDO wordt de voorwaarde echter geëvalueerd voordat de instructies worden uitgevoerd en dat betekent dat de instructies helemaal niet worden uitgevoerd als de voorwaarde al bij aanvang onwaar is.

Voorbeeld

In het volgende voorbeeld wordt DO WHILE gebruikt om record voor record door een tabel te bladeren en een bewerkingsvenster te tonen als de postcode ontbreekt in een record. Dit proces wordt voortgezet tot de EOF( )-markering wordt aangetroffen of de gebruiker op "N" drukt als wordt gevraagd of hij verder wil gaan:

```
SET ESCAPE OFF
USE Klanten
CLEAR
DO WHILE .NOT. EOF()
  IF ISBLANK(Postcode)
    mRec = Recno()
    EDIT NOAPPEND NODELETE RECORD MREC
    SKIP
    ACCEPT "Verder? (J/N): " TO mVerder
      IF UPPER(mVerder) = "N"
        EXIT
      ENDIF
      CLEAR
  ELSE
    SKIP
ENDIF
ENDDO
RETURN
```

Zie ook

DO CASE, DO...UNTIL, FOR...NEXT, IF, SCAN

DO...UNTIL

Programma's

Voert de instructies uit tussen DO en UNTIL zolang een opgegeven voorwaarde onwaar is of totdat het commando EXIT wordt aangetroffen.

Syntaxis

```
DO
      <instructies>
      [LOOP]
      [EXIT]
   UNTIL <voorwaarde Luitdr>
```

<instructies>

Programmaregels met een willekeurige combinatie van commando's, functies, door de gebruiker gedefinieerde functies en LOOP- en EXIT-opties.

LOOP

Geeft de besturing door aan het eind van de DO...UNTIL-lus waar *<voorwaarde Luitdr>* opnieuw wordt geëvalueerd. De instructies tussen LOOP en UNTIL worden niet uitgevoerd.

EXIT

Geeft de besturing door aan de regel na UNTIL *<voorwaarde Luitdr>* zonder *<voorwaarde Luitdr>* opnieuw te evalueren. De instructies tussen EXIT en UNTIL worden niet uitgevoerd.

UNTIL *<voorwaarde Luitdr>*

Een verplicht commando dat het eind aangeeft van de DO...UNTIL-lus. Het argument *<voorwaarde Luitdr>* is een logische uitdrukking die bepaalt of de instructies in DO...UNTIL-lus opnieuw worden uitgevoerd. Als de uitdrukking evalueert tot .F., wordt de besturing doorgegeven aan het commando DO en worden de instructies in de lus opnieuw uitgevoerd. Als de uitdrukking evalueert tot .T., wordt de besturing doorgegeven aan de regel na UNTIL.

Beschrijving

Met DO...UNTIL kunt u een opgegeven reeks instructies uitvoeren zolang een opgegeven voorwaarde onwaar is. Bij een DO...UNTIL-lus worden eerst de instructies tussen DO en UNTIL uitgevoerd tot LOOP, EXIT of UNTIL wordt aangetroffen. Als UNTIL wordt bereikt, wordt de voorwaarde geëvalueerd om te bepalen of de lus opnieuw moet worden uitgevoerd. Omdat de voorwaarde pas aan het eind van de lus wordt geëvalueerd, worden de instructies binnen een DO...UNTIL-lus minimaal één keer uitgevoerd, zelfs als de voorwaarde al bij aanvang waar is.

U kunt andere lussen en structuren nesten binnen een DO...UNTIL-lus, inclusief andere DO...UNTIL-lussen. Maximaal mogen 116 DO...UNTIL-lussen in een procedure of functie worden genest. Het maximum geldt voor alle actieve lussen.

DO...UNTIL en DO WHILE...ENDDO zijn tegengestelde constructies. DO...UNTIL voert instructies uit zolang een voorwaarde onwaar is (tot de voorwaarde waar wordt), terwijl DO WHILE...ENDDO instructies uitvoert zolang een voorwaarde waar is (tot de voorwaarde onwaar wordt). Bij DO WHILE...ENDDO wordt de voorwaarde echter geëvalueerd voordat de instructies worden uitgevoerd en dat betekent dat de instructies helemaal niet worden uitgevoerd als de voorwaarde al bij aanvang onwaar is.

Gebruik DO...UNTIL als u wilt dat de instructies tenminste één keer worden uitgevoerd, zoals in programma's voor gegevensinvoer waar tenminste een keer, maar misschien vaker iets moet worden ingevoerd. In dit soort programma wordt vaak een antwoord van de gebruiker gebruikt om te bepalen of de DO...UNTIL-lus opnieuw moet worden uitgevoerd. Het programma evalueert deze antwoorden met UNTIL *<voorwaarde Luitdr>*.

In andere gevallen wordt de keuze tussen DO...UNTIL en DO WHILE bepaald door het commando dat de minste programmaregels vereist of dat het programma beter leesbaar maakt. In veel gevallen kunt u van een negatieve voorwaarde een positieve voorwaarde maken door het geheel iets anders te programmeren.

Voorbeeld

In het volgende voorbeeld wordt DO...UNTIL gebruikt om een scherm vol informatie over vluchten te tonen. De DO ... UNTIL-lus stopt als rij 20 of het eind van het bestand wordt bereikt:

```
SET TALK OFF
USE Vluchten EXCLUSIVE
CLEAR
? CENTER("Vluchtinformatie")
? "Vluchtnummer","Van","Naar","Vertrek","Aankomst"
DO
  ? Vluchtnr, Van, Naar, Vertrek, Aankomst
  SKIP
UNTIL row() >= 20 .OR. EOF()
CLOSE DATABASE
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DO WHILE, IF, FOR...NEXT

DOS

Stations- en bestandsfuncties

Onderbreekt tijdelijk de uitvoering van dBASE en toont de DOS-opdrachtregel. Er kunnen dan DOS-opdrachten worden uitgevoerd. Nadat de DOS-opdracht EXIT is uitgevoerd, neemt dBASE de draad weer op.

Syntaxis

DOS

Beschrijving

Gebruik het commando DOS om dBASE tijdelijk te verlaten en opdrachten uit te voeren in DOS. Geef in DOS de gewenste opdrachten en vervolgens de opdracht EXIT om terug te keren naar dBASE.

Als u opdrachten geeft in een DOS-venster, wordt met behulp van DBASEWIN.PIF in de directory _dbwinhome COMMAND.COM geladen. U kunt de PIF-editor van Windows gebruiken om de instellingen in DBASEWIN.PIF te wijzigen. Raadpleeg het handboek van Windows voor meer informatie over .PIF-bestanden.

Als u een enkele DOS-opdracht wilt uitvoeren zonder dBASE te verlaten, kunt u ! of RUN gebruiken. Met RUN( ) kunt u Windows-toepassingen uitvoeren.

Voorbeeld

DOS opent een DOS-venster:

```
DOS
```

Typ op de DOS-opdrachtregel EXIT om terug te keren naar dBASE.

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

_dbwinhome, RUN, RUN( )

DOW( )

Datum- en tijdgegevens

Geeft voor een opgegeven datumuitdrukking de dag van de week als resultaat in de vorm van een getal van 1 tot 7.

Syntaxis

DOW(<*Duitdr*>)

<Duitdr>

De datumuitdrukking, in de huidige datumopmaak, waarvan u het nummer van de dag van de week als resultaat wilt krijgen.

Beschrijving

DOW( ) geeft het nummer als resultaat van de dag van de week waarop de datum valt. Zondag is gelijk aan 1, maandag aan 2, enzovoort tot zaterdag (gelijk aan 7). Als u de *naam* van de dag van de week als resultaat wilt hebben, moet u CDOW( ) gebruiken.

U moet <*Duitdr*> opgeven in de huidige datumopmaak die wordt bepaald door SET DATE, DBASEWIN.INI of de optie Internationaal in het Configuratiescherm van

Windows (in die volgorde). Dat wil zeggen, de instellingen bij SET DATE hebben een hogere prioriteit dan die in DBASEWIN.INI, en de instellingen in DBASEWIN.INI hebben weer een hogere prioriteit dan de instellingen in het Configuratiescherm van Windows. Let er op dat *<Duitdr>* overeenkomt met de datumopmaak die wordt gebruikt op het moment dat het programma wordt uitgevoerd.

Als u een ongeldige datum doorgeeft aan DOW( ), wordt die eerst omgezet in een geldige datum. Daarna wordt het nummer van de dag van de week als resultaat gegeven. Als u een lege uitdrukking of een andere dan een datumuitdrukking tussen accolades ({ }) doorgeeft aan DOW( ), wordt 0 als resultaat gegeven. Als u een andere dan een datumuitdrukking of een uitdrukking die niet tussen accolades staat, doorgeeft aan DOW( ), wordt een fout als resultaat gegeven.

Voorbeeld

In het volgende voorbeeld wordt bepaald hoeveel records in de tabel Vluchten een datumveld bevatten met een datum die op Zaterdag (waarde van 7) of Zondag (waarde van 1) valt:

```
CLEAR
USE Vluchten
COUNT FOR DOW(Datum)=1 .OR. DOW(Datum)=7 TO Weekend
? Weekend
RETURN
```

Zie ook

CDOW( ), CMONTH( ), SET CENTURY, SET DATE

DTOC( )

Uitdrukkingen en gegevenstypeconversie

Geeft een opgegeven datumuitdrukking als resultaat in de vorm van een tekenreeks.

Syntaxis

DTOC(*<Duitdr>*)

<Duitdr>

De datumuitdrukking, in de huidige datumopmaak, die u wilt omzetten naar een tekenreeks.

Beschrijving

Met DTOC( ) kunt u een datumuitdrukking omzetten in een tekenuitdrukking. Als u datumgegevens omzet naar tekengegevens, kunt u die manipuleren als elke andere tekenreeks. Als u bijvoorbeeld de huidige datum wilt afdrukken in een rapport, kunt u DTOC(DATE( )) gebruiken. Met DTOS( ) kunt u een datumuitdrukking omzetten in een tekenreeks die geschikt is om op te indexeren of te sorteren.

Geef <*Duitdr*> op in de datumopmaak die wordt bepaald door de instelling van SET DATE, DBASEWIN.INI of de optie Internationaal in het Configuratiescherm van Windows (in die volgorde). Dat wil zeggen, de instellingen bij SET DATE hebben een hogere prioriteit dan die in DBASEWIN.INI, en de instellingen in DBASEWIN.INI hebben weer een hogere prioriteit dan de instellingen in het Configuratiescherm van Windows. Let er op dat <*Duitdr*> overeenkomt met de datumopmaak die wordt gebruikt op het moment dat het programma wordt uitgevoerd. DTOC( ) geeft een tekenuitdrukking in de huidige datumopmaak als resultaat.

Als u een ongeldige datum doorgeeft aan DTOC( ), wordt die eerst omgezet in een geldige datum en wordt die datum vervolgens als resultaat gegeven in de vorm van een tekenuitdrukking. Als u een lege datum ({ / / } of { }) doorgeeft aan DTOC( ), wordt een lege tekenreeks met de huidige datumopmaak als resultaat gegeven. Als DMY is ingesteld met SET DATE, geeft DTOC({ }) " / / " als resultaat. Als u een andere dan een datumuitdrukking of een uitdrukking die niet tussen accolades staat, doorgeeft aan DTOC( ), wordt een fout als resultaat gegeven.

Voorbeeld

In het volgende voorbeeld wordt DTOC( ) gebruikt om datumwaarden uit het veld BalnsDatum om te zetten naar tekenreeksen zodat ze in een array-elementvariabele kunnen worden samengevoegd met een tekenveld (AfnemerNr). Let verder op de conversie van een numerieke waarde (OpenBalans) naar een tekenreeks met STR( ). In het voorbeeld wordt voor elk record een uit drie velden bestaande reeks opgeslagen in de array Reeks. Vervolgens wordt de inhoud van de array weergegeven in het resultatenpaneel van het commandovenster:

```
SET TALK OFF
USE Afnemers
DECLARE Reeks[RECCOUNT() ]
Teller = 1
SCAN
  REEKS[Teller]=AfnemerNr+"*"+DTOC(BalnsDatum)+"*"+;
    LTRIM(STR(OpenBalans,7,2))
  Teller=Teller+1
ENDSCAN
* Volgende FOR...NEXT-lus geeft
* inhoud van de array weer.
FOR i=1 TO RECCOUNT()
  ? REEKS[i] AT 10
NEXT i
```

Zie ook

CTOD( ), DATE( ), DTOS( ), SET DATE, SET CENTURY

DTOR( )

Numerieke gegevens

Zet een waarde in graden om in een waarde in radialen.

Syntaxis

DTOR(<*Nuitdr*>)

<Nuitdr>

Een negatief of positief geheel getal dat gelijk is aan de grootte van de hoek in graden.

Beschrijving

DTOR( ) zet de grootte van een hoek in graden om in radialen. DTOR( ) geeft een zwevende waarde als resultaat. Dit proces verloopt als volgt:

- Het aantal graden wordt vermenigvuldigd met pi.
- Het resultaat wordt gedeeld door 180.
- Het quotiënt wordt als resultaat gegeven.

Een hoek van 180 graden is gelijk aan pi radialen.

U kunt DTOR( ) gebruiken in de trigonometrische functies SIN( ), COS( ) en TAN( ), want deze functies vereisen de grootte van de hoek in radialen als argument. Als u bijvoorbeeld de sinus wilt bepalen van een hoek van 45 graden, gebruikt u SIN(DTOR(45)). Als het standaardaantal decimalen 2 is, levert deze functie 0,71 op.

Het aantal decimalen stelt u in met SET DECIMALS.

Voorbeeld

In de volgende voorbeelden worden enkele toepassingen van DTOR( ) getoond:

```
? DTOR(180)     && Geeft 3,14 (pi) als resultaat
? DTOR(0)       && Geeft 0 als resultaat
? DTOR(360)     && Geeft 6,28 (2*pi) als resultaat
? DTOR(90)      && Geeft 1,57 (pi/2) als resultaat
? DTOR(-90)     && Geeft -1,57 als resultaat
```

In het volgende voorbeeld wordt 60 graden 30 minuten 15 seconden omgezet in radialen:

```
? DTOR(60.525)  && Geeft 1,06 als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ACOS( ), ASIN( ), ATAN( ), ATN2( ), COS( ), PI( ), RTOD( ), SET DECIMALS, SIN( ), TAN( )

DTOS( )

Geeft een opgegeven datumuitdrukking als resultaat in de vorm van een tekenreeks met de opmaak JJJJMMDD.

Syntaxis

DTOS(<*Duitdr*>)

<Duitdr>

De datumuitdrukking, in de huidige datumopmaak, die moet worden omgezet naar een tekenreeks met de opmaak JJJJMMDD.

Beschrijving

Met DTOS( ) kunt u een datumuitdrukking omzetten in een tekenreeks die u kunt gebruiken voor indexeren en sorteren. U kunt DTOS( ) bijvoorbeeld gebruiken als u wilt indexeren op de combinatie van een datumveld en een veld van een ander type. DTOS( ) geeft altijd een tekenreeks met de opmaak JJJJMMDD als resultaat, zelfs als SET CENTURY is uitgeschakeld (OFF).

U moet <*Duitdr*> opgeven in de huidige datumopmaak die wordt bepaald door SET DATE, DBASEWIN.INI of de optie Internationaal in het Configuratiescherm van Windows (in die volgorde). Dat wil zeggen, de instellingen bij SET DATE hebben een hogere prioriteit dan die in DBASEWIN.INI, en de instellingen in DBASEWIN.INI hebben weer een hogere prioriteit dan de instellingen in het Configuratiescherm van Windows. Let er op dat <*Duitdr*> overeenkomt met de datumopmaak die wordt gebruikt op het moment dat het programma wordt uitgevoerd.

Als u de records in een tabel wilt sorteren op datum, onafhankelijk van de huidige datumopmaak, of het datumveld wilt combineren met een ander veld in de tabel, kunt u DTOS( ) gebruiken in plaats van DTOC( ). DTOS( ) geeft altijd de opmaak JJJJMMDD als resultaat, terwijl DTOC( ) altijd de datum in de huidige datumopmaak als resultaat geeft. Dat betekent dat met DTOS( ), de gewijzigde records in een tabel altijd op dezelfde manier worden gesorteerd, onafhankelijk van de geldende datumopmaak. Als u DTOC( ) gebruikt, moet altijd dezelfde datumopmaak zijn ingesteld als de index wordt bijgewerkt. Als de datumopmaak is gewijzigd; zal de index niet correct worden bijgewerkt.

Als u een ongeldige datum doorgeeft aan DTOS( ), wordt die eerst omgezet in een geldige datum en vervolgens wordt de datum in de vorm van een tekenreeks als resultaat gegeven. Als u een lege datumuitdrukking of een andere dan een datumuitdrukking tussen accolades doorgeeft aan DTOS( ), wordt een lege tekenreeks ("") als resultaat gegeven. Als u aan DTOS( ) een uitdrukking doorgeeft die niet tussen accolades staat, wordt een fout als resultaat gegeven.

Voorbeeld

In de volgende voorbeelden wordt het verschil tussen DTOS( ) en DTOC( ) getoond.

```
SET CENTURY OFF
SET DATE DMY
datum1 = {1/4/94}
? DTOC(datum1)              && Geeft 01/04/94 als resultaat
? DTOS(datum1)              && Geeft 19940401 als resultaat
SET CENTURY ON
? DTOC(datum1)              && Geeft 01/04/1994 als resultaat
? DTOS(datum1)              && Geeft 19940401 als resultaat
SET CENTURY OFF
```

In het volgende voorbeeld wordt DTOS( ) gebruikt om gegevens in een datumveld te converteren naar een tekenreeks dia begint met het jaar, zodat eenvoudig naar een bepaald jaar kan worden gezocht.

```
SET TALK OFF
USE Afnemers EXCLUSIVE
INDEX ON DTOS(BalnsDatum) TAG Bal_datum
jr = "1992"
? CENTER("Rapport over: "+jr+" - Uitgevoerd op: " +  DTOC(DATE() ))
?                         && Witregel
  SCAN FOR SUBSTR(DTOS(BalnsDatum),1,4) = jr
    ? Bedrijf + DTOC(BalnsDatum) AT 5
  ENDSCAN
SET TALK ON
CLOSE ALL
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

DTOC( ), INDEX, SET CENTURY, SET DATE

EDIT

Velden en records

Toont velden in de huidige tabel zodat die kunnen worden gewijzigd.

Syntaxis

EDIT
[<*eerste record Nuitdr*> | <*bladwijzer*>]
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[COLOR [<*gewone tekst*>] [, [<*speciale tekst*>] [, <*kader*>]]]
[COLUMNAR]
[COMPRESS]
[FIELDS <*veld1*> [<*veldoptielijst1*>] |
    <*rekenveld1*> = <*uitdr1*> [<*rekenveldoptielijst1*>]
    [, <*veld2*> [<*veldoptielijst2*>] |
    <*rekenveld2*> = <*uitdr2*> [<*rekenveldoptielijst2*>] ...]]

[FORMAT]
[FREEZE <*veld3*>]
[KEY <*uitdr3*> [, <*uitdr4*>]]
[LOCK <*Nuitdr*>]
[NOAPPEND]
[NODELETE]
[NOEDIT | NOMODIFY]
[NOFOLLOW]
[NOINIT]
[NORMAL]
[NOTOGGLE]
[NOWAIT]
[TITLE <*Tuitdr1*>]
[WIDTH <*Nuitdr*>]
[WINDOW <*vensternaam*>]

<*eerste record Nuitdr*> | <*bladwijzer*>

Het recordnummer of de bladwijzer (voor tabellen die geen recordnummers gebruiken) van het record dat u wilt wijzigen.

<*bereik*>

Het aantal records dat u wilt wijzigen. RECORD <*n*> geeft een enkel record aan door middel van zijn recordnummer. NEXT <*n*> geeft *n* records aan, te beginnen bij het huidige record. ALL specificeert alle records. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR <*voorwaarde1*>
WHILE <*voorwaarde2*>

Bepaalt welke records worden beïnvloed door EDIT. FOR beperkt EDIT tot records die voldoen aan <*voorwaarde1*>. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang <*voorwaarde2*> waar is.

COLOR

Geeft de kleuren van gewone tekst, speciale tekst en de omtrek van het tabelrecordsvenster aan. Om de kleuren van deze elementen afzonderlijk op te geven gebruikt u de opties <*gewone tekst*>, <*speciale tekst*> en <*buitenrand*>. U kunt ook de optie <achtergrond> gebruiken als u een scherm met een uniforme achtergrond hebt.

<*gewone tekst*>	Kleurattributen voor commandomeldingen en schermuitvoer. De uitvoer van de commando's ? en @ ... SAY verschijnt bijvoorbeeld in normale tekst.
<*speciale tekst*>	Kleurattributen voor gebieden met speciale tekst, zoals @...GET-velden en gemarkeerde BROWSE-gegevenscellen.
<*buitenrand*>	Kleurattributen voor de omtrek van het gebied waar op het scherm tekst wordt weergegeven.
<*achtergrond*>	Kleurattributen voor de achtergrond van schermen met een uniforme achtergrond. <*achtergrond*> omvat twee parameters: een achtergrondkleur en een attribuut.

De attributen <*gewone tekst*> en <*speciale tekst*> bestaan uit drie instellingen: een voorgrondkleur, een achtergrondkleur en een optionele kleur waarmee een gemengde

(gearceerde) achtergrond wordt gemaakt. Plaats schuine strepen (/) tussen de verschillende instellingen.

Zie SET COLOR TO en SET COLOR OF voor meer informatie over kleureninstellingen.

COLUMNAR
Rangschikt de velden op het bewerkingsformulier in kolommen.

COMPRESS
Beperkt het aantal rijen dat in het tabelrecordsvenster wordt gebruikt voor het weergeven van veldnamen.

FIELDS <*veld1*> [<*veldoptielijst1*>] | <*rekenveld1*> = <*uitdr1*> [<*rekenveldoptielijst1*>] [, <*veld2*> [<*veldoptielijst2*>] | <*rekenveld2*> = <*uitdr2*> [<*rekenveldoptielijst2*>] ...]]
Toont de opgegeven velden in de opgegeven volgorde. Opties voor <*veldoptielijst1*>, <*veldoptielijst2*>, die van toepassing zijn op <*veld1*>, <*veld2*>, enzovoort, bepalen hoe deze velden worden weergegeven. Deze opties zijn als volgt:

\B = <*uitdr1*>, <*uitdr2*> [\F]	RANGE-optie; zorgt dat voor <*veld1*> alleen waarden van <*uitdr1*> tot en met <*uitdr2*> worden geaccepteerd
	RANGE REQUIRED-optie; de optie \F zorgt dat ook een eerder ingevoerde waarde alleen wordt geaccepteerd als die tussen <*uitdr1*> en <*uitdr2*> ligt
\H = <*Tuitdr*>	HEADER-optie; zorgt dat in het tabelrecordsvenster de veldnaam boven de kolom wordt vervangen door <*Tuitdr*>
\P = <*Tuitdr*>	PICTURE-optie; geeft <*veld1*> weer volgens de PICTURE- of FUNCTION-clausule <*Tuitdr*>
\R	READ-ONLY-optie; bepaalt dat <*veld1*> alleen kan worden gelezen en niet gewijzigd
\V = <*voorwaarde*> [\F] [\E = <*Tuitdr*>]	VALID-optie; staat alleen toe dat een nieuwe veldwaarde wordt opgegeven als <*voorwaarde*> waar (.T.) is
	VALID REQUIRED-optie; de optie \F voorkomt dat de cursor het veld mag verlaten en het wijzigen mag worden beëindigd voordat <*voorwaarde*> waar (.T.) is
	ERROR MESSAGE-optie; \E = <*Tuitdr*> zorgt dat <*Tuitdr*> wordt getoond als <*voorwaarde*> onwaar (.F.) is
\W = <*voorwaarde*>	WHEN-optie; zorgt dat <*veld1*> alleen kan worden gewijzigd als <*voorwaarde*> waar (.T.) is

Opmerking U mag het teken "/" ook gebruiken als u maar één enkele optie opgeeft in een veldoptielijst.

Rekenvelden die alleen leesbaar zijn, bestaan uit een toegewezen veldnaam en een uitdrukking die de waarde van het rekenveld als resultaat geeft, zoals *Commissie* =

Percentage * *Verkoopprijs*. Opties voor rekenvelden zijn van invloed op de manier waarop deze velden worden weergegeven. Deze opties zijn als volgt:

\<*kolombreedte kolom*>	De breedte van de kolom waarin <*rekenveld1*> wordt weergegeven
\H = <*Tuitdr*>	Zorgt dat de naam van het rekenveld boven de rekenveldkolom in het tabelrecordsvenster wordt vervangen door <*Tuitdr*>

FORMAT
Zorgt dat invoer wordt geaccepteerd en weergegeven volgens de instellingen in het indelingsbestand dat is geopend met SET FORMAT. Ingevoerde gegevens moeten voldoen aan alle PICTURE-, FUNCTION-, RANGE- en VALID- clausules in het indelingsbestand.

FREEZE <*veld3*>
Zorgt dat alleen <*veld3*> kan worden gewijzigd, hoewel de andere velden wel zichtbaar zijn als u de bladermodus activeert.

KEY <*uitdr3*> [,<*uitdr4*>]
Als de tabel over een hoofdindex beschikt, worden alleen velden getoond waarvan de waarde in het sleutelveld overeenkomt met of komt na <*uitdr3*>, of ligt tussen <*uitdr3*> en <*uitdr4*>.

LOCK <*Nuitdr*>
Zorgt dat de eerste <*Nuitdr2*> velden in de tabel, of de eerste <*Nuitdr*> velden in de veldenlijst op het scherm blijven staan als u in het tabelrecordsvenster naar rechts bladert.

NOAPPEND
Voorkomt dat u tijdens bladeren of wijzigen records toevoegt.

NODELETE
Voorkomt dat in het tabelrecordsvenster records worden gemarkeerd voor verwijdering.

NOEDIT | NOMODIFY
Voorkomt dat in het tabelrecordsvenster records worden gewijzigd.

NOFOLLOW
Als de huidige tabel een hoofdindex heeft, blijft de cursor op zijn plaats als u het sleutelveld in een record verandert, en volgt die niet het record naar diens nieuwe plaats in de indexvolgorde. Als de huidige tabel een hoofdindex heeft en u laat NOFOLLOW achterwege, wordt de tabel opnieuw geïndexeerd zodra u naar een ander record gaat.

NOINIT
Zorgt dat de opties worden gebruikt die zijn ingesteld toen EDIT de vorige keer werd gebruikt. Gebruik NOINIT als een programma enkele malen EDIT aanroept of als u enkele malen achtereen het commando EDIT geeft in het commandovenster, en u wilt

dezelfde opties gebruiken als de vorige keer. U stelt de opties voor EDIT alleen de eerste keer in en gebruikt vervolgens EDIT NOINIT.

NORMAL
Als EDIT vanuit een actief venster is gegeven, wordt het tabelrecordsvenster in de normale schermmodus getoond met de standaard- of de gedefinieerde kleuren. De gedefinieerde kleuren voor het venster worden genegeerd. Als u stopt met wijzigen, keert u terug naar het actieve venster. Zonder NORMAL verschijnen de tabelrecords in het actieve venster.

NOTOGGLE
Voorkomt overschakelen tussen de wijzig- en bladermodus.

NOWAIT
Zet de uitvoering van een programma voort nadat een tabelrecordsvenster is weergegeven; anders wordt de uitvoering opgeschort tot het tabelrecordsvenster is gesloten.

TITLE <*Tuitdr1*>
Toont <*Tuitdr*> als titel van het tabelrecordsvenster.

WIDTH <*Nuitdr*>
Geeft de weergavebreedte van tekenvelden in het tabelrecordsvenster aan. Als een veld breder is dan de opgegeven breedte, kunt u het binnen de opgegeven breedte verschuiven. Het argument <*Nuitdr*> moet een positief getal opleveren.

WINDOW <*vensternaam*>
Activeert het venster <*vensternaam*> en toont tabelrecords in het venster.

Beschrijving
Gebruik EDIT om de inhoud van een of meer tabelrecord te tonen en te wijzigen. EDIT zonder de optie FIELDS toont alle velden van het huidige record. Met het commando SET RELATION kunt u velden uit records in gekoppelde tabellen weergeven.

Druk op *PgUp* en *PgDn* om in het tabelrecordsvenster met één record tegelijk omhoog of omlaag te gaan. U kunt ook de vensterstuurelementen en de muis gebruiken en menuopdrachten kiezen om de bewerken met EDIT te regelen. Raadpleeg het *Handboek* voor meer informatie over het uitvoeren van bewerkingen en het manoeuvreren in het tabelrecordsvenster.

Als u klaar bent met het tonen en/of wijzigen van een tabel in het tabelrecordsvenster, kunt u op *Ctrl-W* drukken of **Bestand | Opslaan en sluiten** kiezen om met wijzigen te stoppen. Als u wilt stoppen met wijzigen zonder de wijzigingen aan het huidige record op te slaan, drukt u op *Ctrl-Q*, kiest u **Bestand | Afbreken en sluiten** of dubbelklikt u op het systeemmenu. Als u in een programma EDIT of BROWSE gebruikt en u stopt met wijzigen of bladeren, wordt de besturing doorgegeven aan de programmaregel na de regel met het commando EDIT.

Als u een memoveld wilt tonen en wijzigen, verplaatst u de cursor naar het memoveld en drukt u op *Ctrl-Home* of dubbelklikt u op het memoveld. Druk op *Ctrl-W* om het gewijzigde memoveld op te slaan.

Als u een record wilt toevoegen, verplaatst u de cursor naar het laatste veld in de tabel en drukt u op *PgDn*. Dat kan alleen als u niet de optie EDIT...NOAPPEND hebt gebruikt.

Een groot aantal van de opties bij EDIT zijn gelijk aan die bij BROWSE. (Druk op *F2* om tussen beide modi te schakelen.)

Gebruik FIELDS gevolgd door een lijst van namen van tabelvelden of rekenvelden om te bepalen welke velden door EDIT en BROWSE worden weergegeven, en of gegevens in een bepaald veld kunnen worden gewijzigd.

Een rekenveld is een uitsluitend leesbaar veld dat de waarde van een uitdrukking weergeeft. In de uitdrukking worden meestal een of meer velden uit de huidige tabel gebruikt, zoals:

```
Naam = Voornaam + " " + Achternaam
Totaal = Prijs + ((Prijs/100) * BTWPerc)
BetaalDatum = CMONTH(NotaDatum + 30)
```

De uitdrukking aan de linkerkant van het gelijkteken (=) geeft de naam van het rekenveld en de uitdrukking aan de rechterkant geeft de inhoud van het rekenveld als resultaat.

In het tabelrecordsvenster worden velden weergegeven volgend de @...SAY...GET-instructies van een geopend indelingsbestand, onafhankelijk van het gebruik van de optie FORMAT. U opent een indelingsbestand met SET FORMAT TO en de naam van het indelingsbestand. Als u een indelingsbestand opent, heeft de optie FIELDS van EDIT geen gevolgen. EDIT en BROWSE tonen dan alleen de velden die zijn opgegeven in het indelingsbestand. Hoewel het indelingsbestand geen invloed heeft op de opmaak van het tabelrecordsvenster, moeten gegevens die worden ingevoerd in het tabelrecordsvenster, voldoen aan alle PICTURE-, FUNCTION-, RANGE- en VALID-clausules in het indelingsbestand.

Met EDIT kunt u alleen de *inhoud* van een tabel wijzigen. De *structuur* van een tabel wijzigt u met MODIFY STRUCTURE. EDIT en CHANGE zijn onderling uitwisselbare commando's.

Voorbeeld

In het volgende voorbeeld wordt EDIT gebruikt om een tabelrecordsvenster te openen met een deelverzameling van de velden en met beperkte mogelijkheden voor wijzigen:

```
USE Bedrijf IN SELECT()  EXCLUSIVE
INDEX ON CompCode TAG CompCode
EDIT FIELDS CompCode, Bedrijf, ;
   Straat1, Plaats, Regio, Postcode ;
   NOFOLLOW NOAPPEND NODELETE
CLOSE ALL
```

Zie ook

APPEND, BROWSE, MODIFY STRUCTURE, SET FORMAT, SET RELATION

EJECT

Afdrukken

Voert een vel uit de printer zodat op een nieuwe pagina wordt begonnen.

Syntaxis

EJECT

Beschrijving

Met EJECT kunt u afdrukuitvoer aan het begin van de pagina plaatsen. Als de printer werkt met kettingvellen (zoals een matrixprinter) en het papier is correct door de printer gevoerd, heeft EJECT tot gevolg dat de printer het papier doorvoert tot het begin van het volgende vel. Als de printer met losse vellen werkt (zoals een laserprinter), drukt EJECT alle gegevens in de printerwachtrij af en wordt de pagina uit de printer gevoerd. Denk er aan de printer aan te sluiten en in te schakelen voordat u met afdrukken begint of EJECT uitvoert.

EJECT werkt in combinatie met _padvance, _plength en _plineno. Als voor _padvance "FORMFEED" (de standaardinstelling) is ingesteld, heeft het commando EJECT dezelfde gevolgen als het drukken op de formfeed-knop op de printer of het sturen van het regeldoorvoer-teken (ASCII 12) naar de printer. Als voor _padvance "LINEFEEDS" is ingesteld, heeft het commando EJECT tot gevolg dat regeldoorvoeren naar de printer worden gestuurd tot _plineno gelijk si _plength. Daarna wordt _plineno weer 0 gemaakt en wordt _pageno verhoogd met 1. Zie _padvance voor meer informatie.

EJECT wordt veel gebruikt bij het afdrukken van rapporten. Als PROW( ) bijvoorbeeld een waarde als resultaat geeft die dicht bij de onderkant van de pagina ligt, kunt u EJECT geven om het rapport boven aan de volgende pagina te vervolgen. EJECT zet automatisch de printkop links boven aan de nieuwe pagina. PROW( ) en PCOL( ) zijn daar beide 0.

EJECT komt overeen met EJECT PAGE, maar EJECT PAGE voert eventueel ook de pagina-afhandelingsroutine uit die u hebt gedefinieerd met ON PAGE.

Voorbeeld

In het volgende voorbeeld wordt één regel naar de printer gestuurd en vervolgens EJECT gebruikt om te zorgen dat volgende uitvoer op een nieuwe pagina komt:

```
SET PRINTER ON
? "Deze pagina blijft verder leeg."
EJECT
```

Zie ook

_padvance, _plength, _plineno, EJECT PAGE, ON PAGE, PCOL( ), PROW( ), SET PRINTER, SET PROW

EJECT PAGE

Voert een vel uit de printer en voert eventueel een ON PAGE-commando uit.

Syntaxis

EJECT PAGE

Beschrijving

Gebruik EJECT PAGE in combinatie met ON PAGE om de doorvoer van pagina's in de printer te besturen. Als u met ON PAGE AT LINE *<Nuitdr>* een pagina-afhandelingsroutine definieert en vervolgens het commando EJECT PAGE geeft, wordt gecontroleerd of het huidige regelnummer (_plineno) groter is dan het regelnummer dat is opgegeven voor *<Nuitdr>*. Als _plineno kleiner is dan *<Nuitdr>*, geeft EJECT PAGE voldoende regeldoorvoeren door om pagina-afhandelingsroutine te starten.

Als _plineno groter is dan *<Nuitdr>*, of als er geen pagina-afhandelingsroutine is gedefinieerd met ON PAGE, wordt de pagina als volgt doorgevoerd:

- Als voor _padvance "FORMFEED" is ingesteld en SET PRINTER is ingeschakeld (ON), wordt een paginadoorvoer (ASCII-code 12) naar de printer gestuurd.
- Als voor _padvance "LINEFEEDS" is ingesteld en SET PRINTER is ingeschakeld (ON), worden voldoende regeldoorvoeren (ASCII-code 10) naar de printer gestuurd om naar het begin van de volgende pagina te gaan. Het benodigde aantal regeldoorvoeren wordt berekend met de formule _plength – _plineno.
- Als u de uitvoer naar een andere bestemming dan de printer stuurt (met SET ALTERNATE of SET DEVICE bijvoorbeeld), wordt de formule _plength – _plineno gebruikt om het aantal regeldoorvoeren te berekenen.

Na een paginadoorvoer met EJECT PAGE wordt _pageno verhoogd met 1 en wordt _plineno weer gelijk aan 0 gemaakt.

Voorbeeld

Zie ON PAGE voor een voorbeeld met EJECT PAGE.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

?, ??, _padvance, _pageno, _plength, _plineno, EJECT, ON PAGE, PRINTJOB...ENDPRINTJOB, SET ALTERNATE, SET DEVICE, SET PRINTER, SET PROW

ELAPSED( )

Datum- en tijdgegevens

Geeft het aantal seconden tussen twee opgegeven tijdstippen als resultaat.

Syntaxis

ELAPSED(<*eindtijd Tuitdr*>, <*begintijd Tuitdr*>)

<*eindtijd Tuitdr*>

De tijduitdrukking, in de notatie UU:MM:SS, waarop wordt gestopt met het bepalen van het verstreken aantal seconden. Het argument <*eindtijd Tuitdr*> moet een later tijdstip aanduiden dan <*begintijd Tuitdr*>. Als dat niet het geval is, wordt een negatieve waarde als resultaat gegeven.

<*begintijd Tuitdr*>

De tijduitdrukking, in de notatie UU:MM:SS, wordt begonnen met het bepalen van het verstreken aantal seconden. Het argument <*begintijd Tuitdr*> moet een eerder tijdstip aanduiden dan <*eindtijd Tuitdr*>. Als dat niet het geval is, wordt een negatieve waarde als resultaat gegeven.

Beschrijving

Gebruik ELAPSED( ) in combinatie met TIME( ) om de snelheid van een programma te bepalen. dat wil zeggen, om te bepalen hoe lang het programma over een bepaald onderdeel doet. U kunt ELAPSED( ) ook gebruiken om de snelheid van een bepaalde computer te testen. U kunt het resultaat van die meting dan gebruiken om de snelheid van uitvoer naar het scherm aan te passen. Verder kunt u ELAPSED( ) gebruiken om de uitvoering tijdelijk stil te leggen in geautomatiseerde demonstraties en lesprogramma's.

ELAPSED( ) trekt de waarde van <*begintijd Tuitdr*> af van <*eindtijd Tuitdr*>. Als <*begintijd Tuitdr*> een later tijdstip aanduidt dan <*eindtijd Tuitdr*>, geeft ELAPSED( ) een negatief geheel getal als resultaat. Zowel <*eindtijd Tuitdr*> als <*begintijd Tuitdr*> moeten worden opgegeven in de notatie UU:MM:SS. UU staat hier voor de uren, MM voor de minuten en SS voor de seconden. U hoeft echter geen minuten en/of seconden op te geven. Als u dat niet doet, wordt voor beide waarden 0 gebruikt.

Voorbeeld

In het volgende voorbeeld wordt ELAPSED( ) gebruikt om de verstreken tijd in uren tussen twee opgegeven tijdstippen te berekenen:

```
LOCAL f
f = NEW GFORM()
f.Open()

CLASS GFORM OF FORM
   this.Left =          53.00
   this.Height =         15.00
   this.Width =          46.00
   this.Text = "Werktijden"
```

```
this.Top =          5.59

DEFINE TEXT T1 OF THIS;
    PROPERTY;
      Left          4.00,;
      Height          1.00,;
      ColorNormal "N/W",;
      Width         28.00,;
      Border .F.,;
      Text "Begintijd (24 uurs notatie):",;
      Top          3.00

DEFINE ENTRYFIELD V1 OF THIS;
    PROPERTY;
      Left         36.00,;
      Height          1.00,;
      Width          6.00,;
      Value "  :  ",;
      Picture "99:99",;
      Border .T.,;
      Top          3.00

DEFINE TEXT T2 OF THIS;
    PROPERTY;
      Left          4.00,;
      Height          1.00,;
      ColorNormal "N/W",;
      Width         28.00,;
      Border .F.,;
      Text "Eindtijd (24 uurs notatie):",;
      Top          5.00

DEFINE ENTRYFIELD V2 OF THIS;
    PROPERTY;
      Left         36.00,;
      Height          1.00,;
      Width          6.00,;
      Value "  :  ",;
      Picture "99:99",;
      Border .T.,;
      Top          5.00

DEFINE PUSHBUTTON BEREKEN OF THIS;
    PROPERTY;
      Left         8.00,;
      Height          2.00,;
      ColorNormal "N/W",;
      Width         30.00,;
      OnClick {;Resultaat="Vandaag gewerkte tijd: ";
      + LTRIM(STR(ELAPSED(Form.V2.Value, Form.V1.Value)/3600,6,2));.
      +" uur"; Form.Werktijd.Text=Resultaat},;
      Text "Gewerkte tijd berekenen",;
      Default .T.,;
      Top          9.00
```

```
    DEFINE TEXT WERKTIJD OF THIS;
        PROPERTY;
          Left          4.00,;
          Height          1.00,;
          ColorNormal "R+/W",;
          Width         44.00,;
          Border .F.,;
          Text "Vandaag gewerkte tijd: ",;
          Top          7.00

ENDCLASS
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

SECONDS( ), TIME( )

EMPTY( ) — Uitdrukkingen en gegevenstypeconversie

Geeft .T. als resultaat als een opgegeven uitdrukking of veld 0 of leeg is en anders .F.

Syntaxis

EMPTY(<*uitdr*>)

<uitdr>
Een uitdrukking van elk willekeurig gegevenstype.

Beschrijving

EMPTY( ) geeft .T. als resultaat als een opgegeven uitdrukking leeg of 0 is en .F. als de uitdrukking gegevens of een numerieke waarde ongelijk aan 0 bevat.

EMPTY( ) komt overeen met ISBLANK( ), maar ISBLANK( ) maakt onderscheid tussen leeg en 0 in numerieke velden, terwijl EMPTY( ) dat niet doet. Zie ISBLANK( ) voor meer informatie.

Voorbeeld

In het volgende voorbeeld wordt het gebruik van AVERAGE zonder voorwaarden vergeleken met AVERAGE in combinatie met ISBLANK( ) en EMPTY( ).

```
*   Het bestand Leegtest.DBF bevat 4 records
*   met de volgende waarden:
*   Record #       NUMVELD
*          1            0
*          2           10
```

```
*          3          20
*          4         (leeg)
```

De volgende instructie berekent het gemiddelde van alle records. Omdat lege velden ook in de berekening worden meegenomen, is dit geen echt gemiddelde.

```
AVERAGE numveld TO nGem1        && Geeft 7,5 (30/4) als resultaat
```

Bij de volgende berekening worden lege records genegeerd, maar worden velden met een waarde van 0 wel meegenomen. Als waarden van 0 geldig zijn, levert dit het meest nauwkeurige gemiddelde op.

```
AVERAGE numveld TO nGem2 ;
  FOR .NOT. ISBLANK(numveld)    && Geeft 10 (30/3) als resultaat
```

Bij de volgende berekening worden zowel lege velden als velden die de waarde 0 bevatten, genegeerd. Gebruik deze instructie als u records wilt uitsluiten die geen waarde of de waarde 0 hebben.

```
AVERAGE numveld TO nGem3 ;
  FOR .NOT. EMPTY(numveld)      && Geeft 15 (30/2) als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

BLANK, ISBLANK( ), SPACE( ), TYPE( )

EOF( )

Velden en records

Geeft aan of de recordaanwijzer bij het eind van het bestand staat.

Syntaxis

EOF([<*alias*>])

<alias>

Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

EOF( ) geeft .T. als resultaat als de recordaanwijzer in de huidige of opgegeven tabel voorbij het laatste record staat of als de recordaanwijzer voorbij het laatste record in de hoofdindex staat. In alle andere gevallen geeft EOF( ) .F. als resultaat. Ook als in het opgegeven werkgebied geen tabel is geopend, geeft EOF( ) .F. als resultaat.

EOF( ) geeft in de volgende situaties .T. als resultaat: nadat met SCAN het laatste record in een tabel is verwerkt, als u SKIP gebruikt om voorbij het laatste record in een tabel of indexbestand te gaan, als u LIST zonder opties gebruikt of als een van de commando's

CONTINUE, FIND, GO, LOCATE, LOOKUP( ), SEEK( ) en SEEK het opgegeven record niet kan vinden (en SET NEAR is uitgeschakeld (OFF)).

Voorbeeld

In het volgende voorbeeld wordt EOF( ) gebruikt om in een DO...UNTIL-lus te testen op het einde van een bestand. Als de recordaanwijzer voorbij het laatste record komt te staan, wordt EOF( ) waar en wordt de DO...UNTIL-lus beëindigd:

```
USE BEDRIJF
DO
  * ...
  SKIP                  && Naar volgende record
UNTIL EOF()             && Test op EOF()
```

In het volgende voorbeeld wordt een formulier met twee knoppen voor het verplaatsen van de recordaanwijzer gedefinieerd. BOF( ) en EOF( ) worden gebruikt om te voorkomen dat aan een van beide einden van de tabel een fout optreedt:

```
SET PROCEDURE TO PROGRAM(1) ADDITIVE
USE Klanten
DEFINE FORM F1
DEFINE PUSHBUTTON Knop1 OF F1 AT 10,10;
  PROPERTY Text "Vorige",;
    Width 12, OnClick Terug
DEFINE PUSHBUTTON Knop2 OF F1 AT 10,22;
  PROPERTY Text "Volgende",;
    Width 12, OnClick Verder
OPEN FORM F1

FUNCTION Terug
IF .NOT. BOF()
  SKIP-1
ENDIF
RETURN .T.

FUNCTION Verder
IF .NOT. EOF()
  SKIP
ENDIF
RETURN .T.
```

Zie ook

BOF( ), FIND, FOUND( ), LOCATE, RECNO( ), SEEK, SEEK( )

ERASE

Stations- en bestandsfuncties

Verwijdert een bestand van schijf.

Syntaxis

ERASE *<bestandsnaam>* | ? | *<bestandsnaamfilter>*

<bestandsnaam> | ? | <bestandsnaamfilter>
Geeft het bestand aan dat moet worden verwijderd. De opties ? en *<bestandsnaamfilter>* tonen een dialoogvenster waarin u een bestand kunt kiezen.

Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, wordt geen extensie gebruikt.

Beschrijving

Met ERASE kunt u een bestand van schijf verwijderen. ERASE en DELETE FILE zijn onderling uitwisselbare commando's.

ERASE komt vrijwel overeen met de DOS-opdracht DEL (of ERASE). In tegenstelling tot de DOS-opdracht heeft het gebruik van de jokertekens * en ? bij ERASE tot gevolg dat een dialoogvenster verschijnt waarin u kunt aangeven welke bestanden u wilt verwijderen. Verder kunt u met het commando ERASE maar één bestand tegelijk verwijderen. Alleen gesloten bestanden kunnen worden verwijderd met ERASE.

Als het bestand dat u wilt verwijderen een extensie heeft, moet u die extensie opnemen in *<bestandsnaam>*. Als het bestand zich niet op het standaardstation bevindt, moet u het station opgeven. Bevindt het bestand zich niet in de huidige directory of in het pad dat u instelt met SET PATH, dan moet u het directorypad opgeven.

Als *<bestandsnaam>* in de huidige directory staat en tevens in een directory aan het pad dat is ingesteld met SET PATH, verwijdert ERASE *<bestandsnaam>* zonder pad alleen het bestand in de huidige directory. SET SAFETY heeft geen gevolgen voor ERASE.

ERASE verwijdert niet automatisch een .DBT-bestand als het bijbehorende .DBF-bestand wordt verwijderd. Gebruik DELETE TABLE om alle bestanden te verwijderen die bij een tabel horen.

Voorbeeld

In de volgende voorbeelden wordt ERASE gebruikt:

```
ERASE Tijd.prg
ERASE ?
* Dialoogvenster "Bronbestand openen" verschijnt
```

Overdraagbaarheid

Het *<bestandsnaamfilter>* argument wordt in dBASE IV of dBASE III PLUS niet ondersteund.

Zie ook

DELETE TABLE, RENAME, SET DIRECTORY, SET PATH, SET SAFETY

ERROR( )

Foutafhandeling en testen op fouten

Geeft het nummer van de laatste dBASE-fout als resultaat.

Syntaxis

ERROR( )

Beschrijving

Gebruik ERROR( ) om het foutnummer te bepalen als zich een fout voordoet. ERROR( ) geeft in eerste instantie 0 als resultaat. Als zich een fout voordoet, geeft ERROR( ) een foutnummer als resultaat. ERROR( ) blijft dat nummer als resultaat geven tot een van de volgende gebeurtenissen plaatsvindt:

- Er vindt een andere fout plaats.
- Het commando RETRY wordt gegeven.
- De subroutine waarin de fout zich voordeed, wordt beëindigd.

In de volgende tabel worden de functies CERROR( ), DBERROR( ), DBMESSAGE( ), ERROR( ), MESSAGE( ), SQLERROR( ) en SQLMESSAGE( ) met elkaar vergeleken.

Functie	Resultaat
CERROR( )	Nummer van compiler-fout
DBERROR( )	Nummer van IDAPI-fout
DBMESSAGE( )	IDAPI-foutmelding
ERROR( )	Nummer van dBASE-fout
MESSAGE( )	dBASE-foutmelding
SQLERROR( )	Nummer van server-fout
SQLMESSAGE( )	Server-foutmelding

Zie Help voor een lijst van alle foutcodes.

Voorbeeld

Zie ON ERROR( ) voor een voorbeeld van het gebruik van ERROR( ).

Overdraagbaarheid

De codes van sommige fouten in dBASE IV en dBASE III PLUS wijken af van de codes voor dezelfde fouten in dBASE voor Windows. Zie Help voor meer informatie.

EXP( )

Numerieke gegevens

Geeft *e* tot een opgegeven macht als resultaat.

Syntaxis

EXP(<*Nuitdr*>)

<Nuitdr>

De exponent voor het getal *e*. Kan een positief getal, een negatief getal of 0 zijn.

Beschrijving

EXP( ) geeft een zwevend getal als resultaat dat gelijk is aan *e* (het grondtal van de natuurlijke logaritme) tot de macht <*Nuitdr*>. EXP(2) geeft bijvoorbeeld 7,39 als resultaat omdat *e*^2=7,39.

EXP( ) is de tegenhanger van LOG( ). Als Y bijvoorbeeld gelijk is aan EXP(X), dan is LOG(Y) gelijk aan X.

Het aantal decimalen stelt u in met SET DECIMALS.

Voorbeeld

In het volgende voorbeeld wordt EXP( ) gebruikt om her resultaat te bepalen van *e* tot een bepaalde macht:

```
SET DECIMALS TO 6
? EXP(1)            && Geeft 2,718282 als resultaat;
                       de waarde van e
? EXP(0)            && Geeft 1,000000 als resultaat
? EXP(-1)           && Geeft 0,367879 als resultaat
? EXP(-10)          && Geeft 0,000045 als resultaat
? EXP(-50)          && Geeft 0,000000 als resultaat
```

In het volgende voorbeeld wordt 25*25 berekend met gebruik van *e* en natuurlijke logaritmen:

```
dec = SET("DECIMALS")
SET DECIMALS TO 2
x = 25
y = LOG(x) + LOG(x) && y = 6,44
? EXP(y)            && Geeft 625,00 als resultaat
SET DECIMALS TO dec
```

Zie ook

LOG( ), LOG10( ), SET DECIMALS

EXTERN

Programmeren in Windows

Definieert een prototype van een externe functie in een DLL-bestand.

Syntaxis

EXTERN [CDECL] *<resultaattype>* *<functienaam>*
([*<lijst van parametertypes>*])
[*<pad>*] *<bestandsnaam>*

of

EXTERN [CDECL] *<resultaattype>* *<naam door gebruiker gedefinieerde functie>*
([*<lijst van parametertypes>*])
[*<pad>*] *<bestandsnaam>*
FROM *<exportfunctienaam>* | *<rangnummer>*

Omdat u met EXTERN het prototype van een functie maakt, zijn haakjes verplicht. Haakjes zijn van invloed op de manier waarop gegevenstypen worden gebruikt en geconverteerd.

[CDECL]

Zorgt dat de aanroepconventies van de taal C worden gebruikt. Als u CDECL achterwege laat, worden de aanroepconventies van Pascal gebruikt. (Zie de volgende tabel.)

<functienaam>

De exportnaam van de functie. De exportnaam van een externe functie staat in het .REF-bestand dat hoort bij het .DLL-bestand waarin de functie is opgeslagen.

<resultaattype> en *<parametertype>*

Bepaalt respectievelijk het gegevenstype van de waarde die door de functie als resultaat wordt gegeven en het gegevenstype van elk argument dat u doorgeeft aan de functie. De volgende tabel geeft een overzicht van de beschikbare sleutelwoorden.

	Sleutelwoord	dBASE-gegevenstype	C-gegevenstype	Pascal-gegevenstype	ASM-gegevenstype
Parameters of resultaatwaarden	CDOUBLE	Numeriek	long double (80 bit)	Double	Niet beschikbaar
	CHANDLE	Numeriek	Handles, zoals HANDLE, HWND, HFONT, HDC	Handles, zoals Hwnd, HFont, HDC	dw
	CINT	Numeriek	int	Integer	dw (16 bit)
	CLOGICAL	Logisch	short Int	Integer	dw (16 bit)
	CLONG	Numeriek	long int (32 bit)	Long Int	dd (32 bit)
	CSTRING	Teken	char far * (eindigt op 0)	PChar	dw (16 bit)
	CVOID	Niet beschikbaar	void	Procedure	Niet beschikbaar
	CWORD	Numeriek	short int (16 bit)	WORD	dw (16 bit)
Alleen parameters	CPTR	Niet beschikbaar	void *	Pointer	dd (32 bit)

Een C-functie kan bijvoorbeeld een 32-bit long int zonder teken accepteren als parameter en als resultaat een char far * (tekenreeks) geven. In de instructie met

EXTERN geeft u CLONG op als de parameter in *<parametertypelijst>* en als het *<resultaattype>*. Als u de functie aanroept, geeft u een numerieke dBASE-variabele door en slaat u het resultaat op in een tekenvariabele.

<naam door gebruiker gedefinieerde functie>

De naam die u aan de externe functie geeft (ter vervanging van de exportnaam). Als u *<naam door gebruiker gedefinieerde functie>* opgeeft (in plaats van *<functienaam>*), moet u de clausule FROM *<Tuitdr>* | *<Nuitdr>* gebruiken om de functie in het .DLL-bestand aan te duiden.

FROM *<exportfunctienaam>* | *<rangnummer>*

Geeft de functie aan in het .DLL-bestand dat wordt aangegeven door *<bestandsnaam>*. *<exportfunctienaam>* geeft de functie aan door middel van zijn naam. Die naam is opgeslagen in het .DEF-bestand dat bij het .DLL-bestand hoort. *<rangnummer>* duidt de functie aan door middel van een nummer dat ook in het .DEF-bestand staat

Als de functie die u aanroept, geen waarde als resultaat geeft, moet u voor *<resultaattype>* CVOID opgeven.

<bestandsnaam>

De naam van het .DLL-bestand waarin de externe functie is opgeslagen. Als het bestand nog niet in het geheugen aanwezig is, moet u de extensie (.DLL) opgeven. De bestandsnaam van een .DLL-bestand dat u in het geheugen laadt, moet uniek zijn. Het is bijvoorbeeld niet mogelijk zowel SCRIPT.DLL als SCRIPT.FON in het geheugen te laden, ondanks dat de bestanden verschillende extensies hebben.

Als het .DLL-bestand nog niet in het geheugen aanwezig is, wordt het automatisch geladen door EXTERN. Als het .DLL-bestand al in het geheugen staat, wordt de verwijzingsteller met 1 verhoogd. Het is dus niet nodig om eerst LOAD DLL uit te voeren voordat u EXTERN gebruikt.

De verwijzingsteller wordt per dBASE-sessie slechts één keer verhoogd, ongeacht het aantal toepassingen van de commando's LOAD DLL en EXTERN.

<pad>

Het directorypad naar het .DLL-bestand waarin de externe functie is opgeslagen. Als u *<pad>* achterwege laat, wordt het .DLL-bestand standaard in de volgende directory's gezocht:

1. De huidige directory.
2. De Windows-directory (zoals C:\WINDOWS).
3. De subdirectory SYSTEM van Windows (zoals C:\WINDOWS\SYSTEM).
4. De directory met DBASEWIN.EXE, of de directory met het .EXE-bestand van het gecompileerde programma.
5. De directory's in het huidige DOS-pad.
6. De directory's die voor zoeken zijn ingesteld op het netwerk.

Het argument <*pad*> is alleen nodig als het .DLL-bestand niet in een van deze directory's staat.

Beschrijving

Met EXTERN definieert u een prototype voor een externe functie die is geschreven in een andere taal dan de eigen programmeertaal van dBASE. Het prototype bepaalt hoe de argumenten worden geconverteerd naar gegevenstypen die door de externe functie worden gehanteerd en hoe het resultaat kan worden omgezet naar een gegevenstype dat door dBASE wordt begrepen.

Als u een externe DLL-functie wilt aanroepen, moet u eerst het prototype definiëren met EXTERN. Vervolgens kunt u de bij EXTERN opgegeven functienaam gebruiken en de functie aanroepen als elke andere dBASE-functie.

De externe functie bevindt zich in een C-bibliotheek (C library) zoals de API van Windows of een eigen .DLL-bestand dat u maakt in C, Pascal of ASM. (Zie Hoofdstuk 25 in *Programmeren* voor meer informatie over het werken met EXTERN en .DLL-bestanden.) Hoewel de meeste externe code is opgeslagen in bestanden met de extensie .DLL, kan dergelijke code ook staan in .EXE-bestanden of in .DRV- en .FON-bestanden.

Voorbeeld

In het volgende voorbeeld wordt een dBASE-programma getoond dat EXTERN gebruikt om een C-programma (cVrbld1.PRG) aan te roepen. De verklaring van de functies in het bestand is als volgt:

```
* Omkeren() :
*   gebruikt de optie FROM om een nieuwe naam te geven
*   aan de functie (StrRevC) met rangnummer 2 in
*   het .DLL-bestand. In dit voorbeeld wordt de variabele
*   MynTReeks doorgegeven als referentieparameter, en
*   vervolgens door de functie StrRevC() in het DLL-bestand
*   gewijzigd. De gewijzigde tekenreeks wordt dan weergegeven
*   in dBASE.
* LeesDOSOmg() :
*   Dit voorbeeld laat zien hoe het adres van
*   een tekenreeks wordt doorgegeven vanuit een DLL.
*   De tekenreeks, een reeks uit de DOS-omgeving, wordt
*   weergegeven in dBASE.
* LeesDOSOmgAant() :
*   Dit voorbeeld laat zien hoe een getal wordt doorgegeven
*   vanuit een DLL. Het getal stelt het aantal tekenreeksen in
*   de DOS-omgeving voor. Gebruik deze functie om te bepalen
*   hoeveel tekenreeksen moeten worden ingelezen met LeesDOSOmg().
* LeesDOSOmgLen() :
*   Dit voorbeeld laat zien hoe een getal wordt doorgegeven
*   vanuit een DLL. Het getal stelt de totale lengte voor
*   van alle tekenreeksen in de DOS-omgeving.
************************************************
CLEAR
EXTERN CVOID   Omkeren(CSTRING) cVrbld1.DLL FROM 2
* 2 is het ORDINAL-getal in de DLL cVrbld1.DLL
EXTERN CSTRING LeesDOSOmg(CWORD)  cVrbld1.DLL
```

```
EXTERN CWORD   LeesDOSOmgLen()    cVrbld1.DLL
EXTERN CWORD   LeesDOSOmgAant()   cVrbld1.DLL
MynTReeks = "AbCdE"
? "Oorspronkelijke tekenreeks: ",MynTReeks
Omkeren(MynTReeks)
? "Na verwerking met StrRevC: ",MynTReeks
nGrootte=LeesDOSOmgLen()
? "De grootte van de DOS-omgeving is: ",nGrootte
?
? "De DOS-omgeving bestaat uit:"
FOR i = 1 to LeesDOSOmgAant()   && voor 1 tot aantal;
                                   tekenreeksen in DOS-omg
   ? LeesDOSOmg(i)              && toon elke reeks;
                                   uit DOS-omgeving
NEXT
```

De C-code voor het maken van het .DLL-bestand (cVrbld1.C) luidt als volgt:

```
#include <windows.h>
#include <string.h>
#pragma argsused
int FAR PASCAL LibMain(HINSTANCE hInstance,
  WORD wDataSeg, WORD wHeapSize, LPSTR lpszCmdline){
         if (wHeapSize > 0) UnlockData (0) ;
         return 1;}
#pragma argsused
int FAR PASCAL WEP(int wParameter) {
         return 1;}
/* ########################################
   ##  Functie StrRevC()                 ##
   ########################################*/
#pragma argsused
void FAR PASCAL StrRevC(LPSTR lpstrString) {
          strrev(lpstrString);}
/* ########################################
   ##  Functies voor LeesDOSOmg()        ##
   ########################################*/
#pragma argsused
LPSTR FAR PASCAL LeesDOSOmg(UINT index) {
        LPSTR p;
        UINT  i;
          for(i=1,p=GetDOSEnvironment() ;
            i<index && *p != '\0';
            p += (lstrlen(p)+1),++i);
          return p;}
#pragma argsused
UINT FAR PASCAL LeesDOSOmgLen(void) {
        return (UINT)
        (LeesDOSOmg((UINT)-1) - LeesDOSOmg(1)) + 1;}
#pragma argsused
UINT FAR PASCAL LeesDOSOmgAant(void) {
        LPSTR p;
        UINT  i;
        for(i=0,p=GetDOSEnvironment() ; *p != '\0';
          p += (lstrlen(p)+1),++i);
        return i;}
```

De C-broncode voor het maken van het .DEF-bestand (cVrbld1.DEF) luidt als volgt:

```
LIBRARY        CVRBLD1.
DESCRIPTION    'Een voorbeeld-DLL voor dBASE voor Windows'
EXETYPE        WINDOWS
CODE           PRELOAD MOVEABLE DISCARDABLE
DATA           PRELOAD SINGLE
HEAPSIZE       1400

EXPORTS
               WEP              @1
               STRREVC          @2
               LEESDOSOMG       @3
               LEESDOSOMGLEN    @4
               LEESDOSOMGNUM    @5
```

In de volgende sectie wordt de optie FROM van EXTERN gedemonstreerd (met de naam van de functie in de .DLL). Het codesegment toont een berichtvenster met het pictogram 'i', de titel 'MijnVenster, de melding 'Hallo Wereld' en twee knoppen, 'OK' en 'Annuleren'.

```
EXTERN CWORD MijnVenster(CHANDLE,CSTRING,CSTRING,CWORD);
  user.exe FROM "MessageBox"
? MijnVenster(0,"Hallo Wereld","MijnVenster",65)
```

Voorbeeld 2: een PASCAL-DLL aanroepen. In dit voorbeeld wordt de variabele MynTReeks doorgegeven als referentieparameter. De variabele wordt vervolgens gewijzigd door de functie StrRevP( ) in de DLL en de gewijzigde tekenreeks wordt als resultaat gegeven.

```
* dBASE-programma (pStrRev.PRG)
EXTERN CVOID StrRevP(CSTRING) pStrRev.DLL
MynTReeks = "ABCD"
StrRevP(MynTReeks)
? MynTReeks
```

DLL-broncode (pStrRev.PAS):

```
Library pStrRev;
Uses
  Strings;
Function StrRevP(str : PChar) : PChar; Export;
Var
  endstr : PChar;
  ch     : Char;
Begin
  StrRevP := str;
  If Assigned(str) And (str^ <> #0) Then
    Begin
      endstr := StrEnd(str);
      Dec(endstr);
      While endstr > str Do
        Begin
          ch := str^;
          str^ := endstr^;
          endstr^ := ch;
          Inc(str);
          Dec(endstr)
        End
```

```
    End
End;
Exports
  StrRevP Index 1;
Begin
End.
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

LOAD DLL, RELEASE DLL

FCLOSE( )

Toegang op laag niveau

Sluit een bestand dat eerder is geopend met FCREATE( ) of FOPEN( ). Geeft .T. als resultaat als de bewerking met succes is uitgevoerd en anders .F.

Syntaxis

FCLOSE(<*bestands-handle Nuitdr*>)

<*bestands-handle Nuitdr*>

Het nummer van de bestands-handle voor het bestand dat u wilt sluiten. Als u een bestand opent met FCREATE( ) of FOPEN( ), geven deze functies een bestands-handle-nummer als resultaat. Gebruik dit getal als <*bestands-handle Nuitdr*>. Als u een bestands-handle-nummer opgeeft dat niet eerder als resultaat is gegeven door FCREATE( ) of FOPEN( ), wordt een foutmelding getoond.

Beschrijving

FCLOSE( ) sluit een bestand dat u hebt geopend met FCREATE( ) of FOPEN( ). FCLOSE( ) geeft .T. als resultaat als het bestand is gesloten. Als het bestand niet langer beschikbaar is (bijvoorbeeld omdat de diskette met het bestand is vervangen door een andere) en de buffer bevat nog gegevens die niet naar het bestand zijn geschreven, geeft FCLOSE( ) .F. als resultaat.

FCLOSE( ) sluit geen bestanden die op een andere manier zijn geopend dan met FCREATE( ) of FOPEN( ).

Met CLOSE ALL of CLEAR ALL kunt u alle geopende bestanden sluiten, inclusief bestanden die zijn geopend met FCREATE( ) en FOPEN( ). Gebruik FFLUSH( ) om een bestand op schijf op te slaan zonder het te sluiten.

Voorbeeld

In het volgende voorbeeld wordt FCLOSE( ) gebruikt om het bestand LEESMIJ.TXT te sluiten nadat er enige tekst aan is toegevoegd:

```
IF FILE("C:\DBASEWIN\LEESMIJ.TXT")
Handle=FOPEN("C:\DBASEWIN\LEESMIJ.TXT","RW")
* Bestand openen voor lezen en schrijven
FSEEK(Handle,0,2)                && Naar EOF
FPUTS(Handle,"Einde")            && Naar bestand schrijven
  IF .NOT. FCLOSE(Handle)        && Bestand proberen te sluiten
    ? "Bestand is niet gesloten"
    ? "Het DOS-foutnummer is:", FERROR()
  ENDIF
ENDIF
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CLEAR ALL, CLOSE..., FCREATE( ), FERROR( ), FFLUSH( ), FOPEN( )

FCREATE( )

Toegang op laag niveau

Maakt en opent een opgegeven bestand. Geeft het nummer van de bestands-handle als resultaat als het bestand is geopend en anders -1.

Syntaxis

FCREATE(<*bestandsnaam Tuitdr*>[, <*toegang Tuitdr*>])

<*bestandsnaam Tuitdr*>

De naam van het bestand dat u wilt maken en openen. Standaard wordt het bestand gemaakt in de huidige directory. Als u het bestand in een andere directory wilt maken, moet u ook een pad opgeven bij <*bestandsnaam Tuitdr*>.

<*toegang Tuitdr*>

Het toegangsniveau van het bestand dat u maakt (de mogelijkheden staan in de volgende tabel). *Schrijven* betekent dat u gegevens in het bestand kunt wijzigen (overschrijven) en *toevoegen* betekent dat u gegevens aan het eind van het bestand kunt toevoegen. Als u probeert gegevens te overschrijven in een bestand waarvoor u alleen toestemming hebt voor toevoegen en niet voor schrijven, worden de gegevens toegevoegd aan het eind van het bestand.

<*toegang Tuitdr*>	**Toegangsniveau**
niet opgegeven	Lezen, schrijven en toevoegen

"R"	Alleen lezen
"W"	Alleen schrijven
"A"	Alleen toevoegen
"RW" of "WR"	Lezen en schrijven
"AR" of "RA"	Lezen en toevoegen
"AW" of "WA"	Schrijven en toevoegen

Beschrijving

Gebruik FCREATE( ) om een bestand te maken met de opgegeven naam, een toegangsniveau in te stellen voor het bestand, het bestand te openen en het nummer van de DOS-bestands-handle voor het bestand als resultaat te geven. Als het bestand niet kan worden gemaakt (omdat het bijvoorbeeld al is geopend), geeft FCREATE( ) -1 als resultaat.

SET SAFETY is niet van invloed op FCREATE( ). Als *<bestandsnaam Tuitdr>* al bestaat, wordt het zonder waarschuwing overschreven. Gebruik eerst FILE( ) om te controleren of er al een bestand met de gewenste naam bestaat.

Voordat u andere low-level-functies, zoals FREAD( ) en FWRITE( ), kunt gebruiken, moet u eerst het bestand openen met FCREATE( ) of FOPEN( ). Zowel FCREATE( ) als FOPEN( ) geven het nummer van de bestands-handle als resultaat. Dat nummer moet u doorgeven aan andere low-level-functies.

Als u een bestand opent met FCREATE( ), wordt de bestandsaanwijzer bij het eerste teken in het bestand geplaatst. Gebruik FSEEK( ) om de bestandsaanwijzer op de gewenste positie te plaatsen voordat u uit een bestand leest of naar een bestand schrijft.

Voorbeeld

In het volgende voorbeeld wordt FCREATE( ) gebruikt om een bestand met de naam JACK.TXT en de toegangsniveaus lezen en schrijven te maken:

```
Bestandsnaam="Jack.TXT"
? FILE(Bestandsnaam)    && Geeft .F. als resultaat;
                           bestand bestaat nog niet
Handle=FCREATE(Bestandsnaam,"RA")
IF Handle>0
  FPUTS(Handle,"Dit is een testbestand.")
  FCLOSE(Handle)
  FOPEN(Bestandsnaam)
  ? FGETS(Handle)        && "Dit is een testbestand." weergeven
ENDIF
WAIT
CLEAR ALL
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FCLOSE( ), FERROR( ), FGETS( ), FILE( ), FOPEN( ), FREAD( ), FSEEK( ), SET ALTERNATE, SET DEVICE, SET SAFETY

FDATE( )

Stations- en bestandsfuncties

Geeft het datumstempel voor een opgegeven bestand als resultaat.

Syntaxis

FDATE(*<bestandsnaam Tuitdr>*)

<bestandsnaam Tuitdr>

De naam van het bestand waarvan u het datumstempel wilt weten. Jokertekens worden niet ondersteund.

Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, wordt geen extensie gebruikt.

Beschrijving

Met FDATE( ) kunt u de datum van de laatste wijziging aan het bestand op schijf bepalen.

Als u een bestand wijzigt, krijgt het bestand op schijf het datumstempel dat wordt bepaald door de huidige systeemdatum. Als de gebruiker bijvoorbeeld een tabel wijzigt, wordt het datumstempel van het tabelbestand gewijzigd als het bestand wordt gesloten. FDATE( ) geeft het datumstempel als resultaat.

Als het bestand een extensie heeft, moet u die ook opgeven in *<bestandsnaam Tuitdr>*. Als het bestand niet op het standaardstation staat, moet u een station opgeven. Staat het bestand niet in de huidige directory of in het met SET PATH opgegeven pad staat, dan moet u de directory opgeven.

Als het bestand niet wordt aangetroffen, wordt een fout als resultaat gegeven. U kunt eerste de aanwezigheid van het bestand testen met FILE( ) voordat u FDATE( ) gebruikt. FLUSH is niet van invloed op het datumstempel.

Als *<bestandsnaam Tuitdr>* in de huidige directory staat, maar ook voorkomt in een directory die is ingesteld met SET PATH, geeft FDATE(*<bestandsnaam Tuitdr>*) zonder pad informatie over het bestand in de huidige directory als resultaat.

Voorbeeld

In het volgende voorbeeld worden de datum- en tijdstempels vergeleken van Bedrijf.DBF en de reservekopie van dat bestand op station B:. Als de reservekopie een oudere datum heeft, of als de datums gelijk zijn maar de reservekopie heeft een oudere tijd, dan wordt geadviseerd een reservekopie te maken:

```
DO CASE
CASE FDATE("B:Bedrijf.dbf") = FDATE("Bedrijf.dbf");
 .AND.;
    FTIME("B:Bedrijf.dbf") < FTIME("Bedrijf.dbf")
  * dezelfde dag, maar verschillende tijden
  ? "Maak een reservekopie!"
  WAIT
CASE FDATE("B:Bedrijf.dbf") < FDATE("Bedrijf.dbf")
  * reservekopie is van eerdere datum dan huidige tabel
  Verschil= ;
    FDATE("Bedrijf.dbf") - FDATE("B:Bedrijf.dbf")
  ? Verschil, " dagen sinds laatste reservekopie."
  ? "Maak een reservekopie!"
  WAIT
ENDCASE
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FILE( ), FLUSH, FSIZE( ), FTIME( ), SET DIRECTORY, SET PATH

FDECIMAL( )

Velden en records

Geeft het aantal decimalen in een opgegeven veld van een tabel als resultaat.

Syntaxis

FDECIMAL(<*veldnummer Nuitdr*> [, <*alias*>])

<veldnummer Nuitdr>

De positie van het veld waarvan u het aantal decimalen wilt weten.

<alias>

Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

FDECIMAL( ) geeft het aantal decimalen (cijfers achter de komma) in een opgegeven veld in een tabel als resultaat. FDECIMAL( ) geeft nul als resultaat als het veld geen decimalen heeft of als de tabel geen veld op de opgegeven positie bevat. FDECIMAL( ) geeft een fout als resultaat als u een niet-numeriek veld opgeeft.

Velden in de tabel zijn genummerd van 1 tot 1024 op basis van hun positie in de tabelstructuur. Als u geen werkgebied opgeeft, geeft FDECIMAL( ) het aantal decimalen van de tabel in het huidige werkgebied als resultaat.

Voorbeeld

In het volgende voorbeeld wordt FDECIMAL( ) gebruikt om het aantal decimalen te bepalen in het numerieke veld OpenBalans van de tabel Afnemers:

```
CLEAR
SET TALK OFF
USE Afnemers
DISPLAY STRUCTURE              && OpenBalans heeft veldnummer 11
STORE FDECIMAL(11,1) TO m_decimalen
? FIELD(11) + " heeft " + LTRIM(STR(m_decimalen)) + ;
  " cijfers achter de komma."   && Geeft 2 als resultaat
SET TALK ON
```

Zie FLENGTH( ) voor een ander voorbeeld met FDECIMAL( ).

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

FIELD( ), FLENGTH( )

FEOF( )

Toegang op laag niveau

Geeft .T. als resultaat als de bestandsaanwijzer aan het eind staat van een bestand dat is geopend met FCREATE( ) of FOPEN( ).

Syntaxis

FEOF(<*bestands-handle Nuitdr*>)

<*bestands-handle Nuitdr*>

Het nummer van de bestands-handle van het bestand waarvan u de positie van de bestandsaanwijzer wilt bepalen. Als u een bestand opent met FCREATE( ) of FOPEN( ), geven deze functies een bestands-handle-nummer als resultaat. Gebruik dit getal als <*bestands-handle Nuitdr*>. Als u een bestands-handle-nummer opgeeft dat niet eerder als resultaat is gegeven door FCREATE( ) of FOPEN( ), wordt een foutmelding getoond.

Beschrijving

FEOF( ) bepaalt of de bestandsaanwijzer van het opgegeven bestand aan het eind van het bestand staat (EOF, end of file: einde van bestand). In dat geval wordt .T. als resultaat gegeven. De bestandsaanwijzer staat aan het eind van het bestand als die bij de byte na het laatste teken in het bestand staat.

U kunt de bestandsaanwijzer naar het eind van het bestand verplaatsen met FSEEK( ). De instructie FSEEK(bestnum,0,2) verplaatst de bestandsaanwijzer bijvoorbeeld naar het eind van het bestand waarvan het nummer van de bestands-handle is opgeslagen in de variabele bestnum.

Voorbeeld

In het volgende voorbeeld wordt het bestand LEESMIJ.TXT geopend en wordt het bestand regel voor regel weergegeven of afgedrukt tot het eind van het bestand is bereikt:

```
SET PATH TO C:\DBASEWIN
BestHandle=FOPEN("Leesmij.TXT","R") && Bestand openen, alleen lezen
SET PRINTER ON                      && optioneel
DO WHILE .NOT. FEOF(BestHandle)     && Lus uitvoeren tot EOF
  ? FGETS(BestHandle)               && 1 regel naar scherm;
                                       of printer sturen
ENDDO
SET PRINTER OFF
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FCLOSE( ), FCREATE( ), FERROR( ), FOPEN( ), FSEEK( )

FERROR( )

Toegang op laag niveau

Test of de laatste low-level I/O-functie met succes is uitgevoerd. Geeft het foutnummer als resultaat als dat niet het geval is en anders 0.

Syntaxis

FERROR( )

Beschrijving

U kunt het nummer dat FERROR( ) als resultaat geeft, gebruiken in een foutverwerkingsroutine voor low-level-fouten. In de volgende tabel staan de foutnummers die door FERROR( ) als resultaat worden gegeven.

Foutnummer	Oorzaak
2	Bestand of directory niet gevonden
3	Pad onjuist
4	Geen bestands-handles meer beschikbaar
5	Geen toegang tot bestand
6	Bestands-handle onjuist
8	Directory vol
9	Fout bij verplaatsen van bestandsaanwijzer
13	Geen ruimte meer op schijf
14	Einde van bestand

Voorbeeld

In het volgende voorbeeld wordt geprobeerd een bestand te openen met FOPEN( ). De naam van het bestand is echter verkeerd gespeld. De instructie met IF zorgt dat FERROR( ) het foutnummer als resultaat geeft, als het bestand niet is geopend:

```
SET PATH TO C:\DBASEWIN
Bestnaam="Lesmij.TXT"
* Bestandsnaam verkeerd gespeld
IF FOPEN(Bestnaam) = -1
* Geeft -1 als resultaat als bestand al open was;
  of als openen niet is gelukt
  ? "Het DOS-foutnummer is: ", LTRIM(STR(FERROR() ))
* FERROR() geeft 2 als resultaat - Bestand of directory niet gevonden
ENDIF
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FCLOSE( ), FCREATE( ), FEOF( ), FFLUSH( ), FGETS( ), FOPEN( ), FPUTS( ), FREAD( ), FSEEK( ), FWRITE( ), ON ERROR

FFLUSH( )

Toegang op laag niveau

Schrijft een bestand naar schijf dat eerder is geopend met FCREATE( ) of FOPEN( ). Het bestand wordt niet gesloten. Als de bewerking is geslaagd, wordt .T. als resultaat gegeven, anders .F.

Syntaxis

FFLUSH(<*bestands-handle Nuitdr*>)

<*bestands-handle Nuitdr*>

Het nummer van de bestands-handle van het bestand dat naar schijf moet worden geschreven. Als u een bestand opent met FCREATE( ) of FOPEN( ), geven deze functies een bestands-handle-nummer als resultaat. Gebruik dit getal als <*bestands-handle Nuitdr*>. Als u een bestands-handle-nummer opgeeft dat niet eerder als resultaat is gegeven door FCREATE( ) of FOPEN( ), wordt een foutmelding getoond.

Beschrijving

Gebruik FFLUSH( ) om een bestand in de bestandsbuffer op te slaan op schijf, de bestandsbuffer te legen en het bestand open te houden. Als FFLUSH( ) is geslaagd, wordt .T. als resultaat gegeven.

Het wegschrijven van een buffer naar schijf komt neer op het opslaan van een bestand zonder het te sluiten. Tot de buffer van een geopend bestand naar schijf is geschreven, zijn gewijzigde gegevens alleen aanwezig in het RAM-geheugen. Als de stroom uitvalt of als dBASE op een abnormale manier wordt beëindigd, gaan de gegevens in RAM verloren. Als u echter FFLUSH( ) hebt gebruikt om de bestandsbuffer naar schijf te schrijven, gaan alleen de gegevens verloren die u hebt toegevoegd of gewijzigd tussen de laatste keer dat u FFLUSH( ) hebt gebruikt en het moment dat het systeem uitvalt.

Met FCLOSE( ) wordt een bestand opgeslagen en vervolgens gesloten.

Voorbeeld

In het volgende voorbeeld wordt een bestand met de naam JACK.TXT gemaakt. Vervolgens wordt een tekenreeks aan het bestand toegevoegd en wordt het bestand naar schijf geschreven met FFLUSH( ). Het bestand blijft open. Met FPUTS( ) wordt vervolgens meer tekst aan het bestand toegevoegd. De tekst wordt weergegeven met de lus DO WHILE .NOT. FEOF( ):

```
Handle=FCREATE("Jack.TXT","RW")
Bestnaam="Jack.Txt"
FPUTS(Handle,"Borland betekent")  && Tekst in bestand
FFLUSH(Handle)                    && Bestand opslaan op schijf;
                                     bestand blijft open
FSEEK(Handle,0,0)                 && Aanwijzer verplaatsen;
                                     naar begin van bestand
? FGETS(Handle)                   && Tekst weergeven
WAIT                              && Programma pauzeert
CLEAR                             && Scherm wissen
FPUTS(Handle,"kwaliteit in software");
                                  && Tekst toevoegen
FCLOSE(Handle)                    && Jack.TXT sluiten
Handle=FOPEN(Bestnaam,"R")        && Bestand opnieuw openen
DO WHILE .NOT. FEOF(Handle)
  ? FGETS(Handle)                 && Inhoud weergeven
ENDDO
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FCLOSE( ), FCREATE( ), FEOF( )

FGETS( )

Toegang op laag niveau

Geeft een tekenreeks als resultaat uit een bestand dat eerder is geopend met FCREATE( ) of FOPEN( ).

Syntaxis

FGETS(*<bestands-handle Nuitdr>* [, *<tekens Nuitdr>*] [, *<einde-regel uitdr>*])

<bestands-handle Nuitdr>

Het nummer van de bestands-handle van het bestand waaruit tekens moeten worden gelezen. Als u een bestand opent met FCREATE( ) of FOPEN( ), geven deze functies een bestands-handle-nummer als resultaat. Gebruik dit getal als *<bestands-handle Nuitdr>*. Als u een bestands-handle-nummer opgeeft dat niet eerder als resultaat is gegeven door FCREATE( ) of FOPEN( ), wordt een foutmelding getoond.

<tekens Nuitdr>

Het aantal tekens dat moet worden gelezen en als resultaat gegeven voordat een regelterugloop wordt bereikt. Waarden van 0 tot 32766 zijn toegestaan; als *<tekens Nuitdr>* kleiner is dan 0 of groter dan 32766, gebruikt dBASE respectievelijk 0 of 32766.

<einde-regel uitdr>

De einde-regelindicatie, die uit één of twee tekens kan bestaan. De volgende tabel geeft een overzicht van de standaardcodes voor einde-regelindicaties. Plaats deze codes tussen aanhalingstekens. U kunt twee tekens combineren met een plusteken (+), zoals CHR(141) + CHR(138).

<einde-regel uitdr>	**Betekenis**
niet opgegeven	Harde regelterugloop/regeldoorvoer (0D0A Hex)
CHR(141)	Zachte regelterugloop (V.S.) (8D Hex)
CHR(255)	Zachte regelterugloop (Europa) (FF Hex)
CHR(138)	Zachte regeldoorvoer (V.S.) (8A Hex)
CHR(0)	Zachte regeldoorvoer (Europa) (00 Hex)
CHR(13)	Harde regelterugloop (0D Hex)
CHR(10)	Harde regeldoorvoer (0A Hex)

Het is niet mogelijk rechtstreeks hexadecimale getallen op te geven voor *<einde-regel uitdr>*, maar u kunt wel decimale voorstellingen combineren. 0D0A Hex is bijvoorbeeld gelijk aan CHR(13) + CHR(10); CHR(13) is 0D Hex en CHR(10) is 0A Hex. Gebruik HTOI( ) om een hexadecimaal getal te converteren naar zijn decimale waarde.

Beschrijving

FGETS( ) lees een tekenreeks uit het opgegeven bestand en geeft die als resultaat. De tekenreeks begint bij de positie van bestandsaanwijzer en loopt tot het eerstvolgende einde-regelteken. Het einde-regelteken maakt geen deel uit van de tekenreeks. Als u geen aantal tekens opgeeft met *<tekens Nuitdr>*, worden maximaal 32766 tekens als resultaat gegeven. Als FGETS( ) het einde-regelteken aantreft voordat het opgegeven aantal tekens is gelezen, worden de tekens tot aan het einde-regelteken als resultaat gegeven.

Als FGETS( ) het hexadecimale teken 00, CHR(0), aantreft, wordt dat beschouwd als een einde-regelteken. Als u denkt dat een bestand deze tekens bevat, kunt u FREAD( ) gebruiken om één byte tegelijk te lezen zodat u ze kunt opsporen.

Als FGETS( ) niet met succes kan worden uitgevoerd, wordt een lege tekenreeks ("") als resultaat gegeven. Dat kan bijvoorbeeld gebeuren als de bestandsaanwijzer aan het eind van het bestand staat. Gebruik FEOF( ) en FERROR( ) om te bepalen of de bestandsaanwijzer aan het eind van het bestand staat of dat er sprake is van een andere situatie die tot gevolg heeft dat FGETS( ) geen reeks als resultaat kan geven.

Als FGETS( ) een einde-regelindicatie aantreft, wordt de bestandsaanwijzer bij het eerste teken voorbij de einde-regelindicatie geplaatst. Is dat niet geval, dan plaatst FGETS( ) de bestandsaanwijzer direct na het laatste teken dat als resultaat wordt gegeven. Met FSEEK( ) kunt u de bestandsaanwijzer verplaatsen voordat of nadat u FGETS( ) gebruikt.

FREAD( ) en FGETS( ) zijn onderling uitwisselbaar met uitzondering van één eigenschap. FREAD( ) geeft ook einde-regeltekens als resultaat, terwijl FGETS( ) dat niet doet. Als FGETS( ) een einde-regelteken aantreft, stopt de functie daar, ook als nog geen *<tekens Nuitdr>* tekens zijn gelezen. FREAD( ) neemt einde-regeltekens daarentegen gewoon op in de tekenreeks die als resultaat wordt gegeven.

Voorbeeld

In het volgende voorbeeld wordt een tekstbestand gemaakt met de naam TEST.TXT. Aan dat bestand worden negen regels met de tekenreeks "123456789" toegevoegd. Vervolgens wordt FGETS( ) gebruikt in combinatie met een teller voor tekens (Teller) en een teller voor regels (Teller2) om negen regels weer te geven. Elke regel is telkens één teken langer dan de voorgaande (de eerste regel bestaat uit 1 teken en de laatste uit 9). FSEEK( ) wordt in combinatie met Teller2 gebruikt om de bestandsaanwijzer voor elke volgende regel 11 bytes verder te plaatsen:

```
nHandle=FCREATE("Test.TXT","RW")  && Bestand maken
* Invoer van tekstgegevens
FOR i=1 TO 9                      && Lus voor 9 regels
  FPUTS(nHandle,"123456789")      && Tekst toevoegen
NEXT i                            && i verhogen
* Output text data
FSEEK(nHandle,0,0)                && Aanwijzer verplaatsen;
                                     naar begin van bestand
CLEAR                             && Resultatenpaneel van het;
                                     commandovenster legen
Teller=1
Teller2=11                        && 11 bytes/regel
DO WHILE .NOT. FEOF(nHandle)
  ? FGETS(nHandle,Teller)         && Tekenreeks met lengte;
                                     Teller weergeven
  FSEEK(nHandle,Teller2)          && Aanwijzer verplaatsen
  Teller=Teller+1                 && Teller
  Teller2=Teller2+11              && Teller2 verhogen
ENDDO
FCLOSE(nHandle)                   && Test.TXT sluiten
```

Overdraagbaarheid
Wordt niet ondersteund in dBASE III PLUS.

Zie ook
FCREATE( ), FEOF( ), FERROR( ), FOPEN( ), FPUTS( ), FREAD( ), FSEEK( ), HTOI( )

FIELD( ) — Velden en records

Geeft de naam van een veld op een opgegeven positie in een tabel als resultaat.

Syntaxis
FIELD(<*veldnummer Nuitdr*> [, <*alias*>])

<*veldnummer Nuitdr*>
De positie van het veld waarvan u de naam wilt weten.

<*alias*>
Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving
FIELD( ) geeft de naam van een veld als resultaat op basis van de parameter <*veldnummer Nuitdr*>. De velden in een tabel zijn volgens hun positie in de tabelstructuur genummerd van 1 tot 1024.

Als u geen werkgebied opgeeft, geeft FIELD( ) de naam van een veld in de huidige tabel als resultaat. Als de tabel geen veld bevat op de opgegeven positie, geeft FIELD( ) een lege tekenreeks ("") als resultaat.

Voorbeeld
In het volgende voorbeeld wordt FIELD( ) gebruikt om de namen van velden in een tabel als resultaat te geven. Die namen worden gebruikt om een formulier voor gegevensinvoer te genereren. Deze code kan worden gebruikt als een algemene procedure voor gegevensinvoer als de structuur van de tabel niet op voorhand bekend is:

```
SET TALK OFF
CLEAR
USE Klanten                  && Kan elke tabel zijn
APPEND BLANK
AantalVelden=1
DO WHILE LEN(FIELD(AantalVelden))<>0
     mVeld=FIELD(AantalVelden)
     @ ROW() +1,0 SAY mVeld+;
     REPLICATE(".",25-LEN(mVeld)) GET &mVeld
```

```
    AantalVelden = AantalVelden + 1
ENDDO
READ
CLEAR
SET TALK ON
```

Zie ook

DBF( ), FLENGTH( )

FILE( )

Stations- en bestandsfuncties

Test op de aanwezigheid van een bestand. Geeft .T. als resultaat als het bestand bestaat en anders .F.

Syntaxis

FILE(<*bestandsnaam Tuitdr*>)

<*bestandsnaam Tuitdr*>

De naam van het bestand dat u zoekt. Jokertekens worden niet ondersteund.

Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, wordt geen extensie gebruikt.

Beschrijving

Met FILE( ) kunt u bepalen of een bestand wel of niet bestaat.

Als het bestand dat u zoekt een extensie heeft, moet u die opnemen in <*bestandsnaam Tuitdr*>. Als het bestand niet op het standaardstation staat, moet u een station opgeven. Staat het bestand niet in de huidige directory of in het pad dat is ingesteld met SET PATH, dan moet u een directorypad opgeven.

Voorbeeld

In de volgende voorbeelden wordt FILE( ) gebruikt:

```
? FILE("C:\Command.com") && .T.
? FILE("Bedrijf.dbf")    && .T.
? FILE("Bak\Nietniet")   && .F.
```

FILE( ) wordt vaak gebruikt om te testen of een bestand bestaat voordat het wordt geopend:

```
DO WHILE .NOT. FILE("B:Bedrijf.dbf")
  ? "Plaats diskette met reservekopie in station B:"
  WAIT
ENDDO
USE B:Bedrijf
```

Zie ook

DIR, DISPLAY FILES, GETFILE( ), PUTFILE( ), SET DIRECTORY, SET PATH

FIND

Tabelindeling

Plaatst de recordaanwijzer bij het eerste record in een geïndexeerde tabel waarvan de sleutelwaarde overeenkomt met een opgegeven sleutelwaarde.

Syntaxis

FIND *<zoeksleutel>*

<zoeksleutel>

De volledige of een deel van de sleutelwaarde van het record dat u zoekt. Plaats *<zoeksleutel>* niet tussen enkele of dubbele aanhalingstekens of teksthaakjes.

Beschrijving

Het commando FIND plaatst de recordaanwijzer bij het eerste record in een geïndexeerde tabel dat overeenkomt met een opgegeven vaste sleutelwaarde. U kunt met FIND alleen zoeken naar tekens of een zwevende of numerieke waarde in het indexsleutelveld. Als SET EXACT is ingeschakeld (ON), moet de indexsleutel exact overeenkomen met *<zoeksleutel>*.

De instelling voor SET EXACT bepaalt in combinatie met de ingestelde taalaansturing hoe speciale tekens (tekens met accenten) worden verwerkt bij zoekbewerkingen. Zie Appendix C in *Programmeren* voor informatie over de verwerking van speciale tekens bij zoekbewerkingen.

Geïndexeerd zoeken lijkt op het opzoeken van een onderwerp in de index van een boek, waarbij direct naar de pagina met het betreffende onderwerp wordt gegaan. Als de index eenmaal is gemaakt, kunt u met FIND en SEEK snel naar een bepaald record gaan.

FIND begint met zoeken aan het begin van de index en zoekt door tot een overeenkomst is gevonden of tot het eind van de index is bereikt. Als een overeenkomst is aangetroffen (FOUND( ) geeft .T. als resultaat), wordt de recordaanwijzer in de bijbehorende tabel bij het record met de overeenkomst geplaatst.

Met SKIP kunt u naar andere records gaan die in het sleutelveld de opgegeven tekens of het opgegeven getal bevatten. SKIP verplaatst de recordaanwijzer naar het volgende geïndexeerde record, zoals andere records met overeenkomende sleutelvelden (als die bestaan).

In tegenstelling tot het commando CONTINUE na het commando LOCATE, zoekt SKIP niet naar een overeenkomst. Het commando heeft alleen tot gevolg dat de aanwijzer één record verder wordt geplaatst. Het maakt niet uit of u dan bij een record terecht komt dat overeenkomt met het argument bij FIND of niet.

Als FIND niets heeft gevonden, wordt de melding Niet gevonden als resultaat gegeven en wordt de recordaanwijzer voorbij het laatste record in de index geplaatst. (EOF( ) is .T.; FOUND( ) is .F.)

Tenzij het sleutelveld voorloopspaties bevat, hoeft het argument niet tussen aanhalingstekens of teksthaakjes te staan. De opgegeven uitdrukking wordt gebruikt vanaf het eerste teken dat geen spatie is. Als u echter zoekt naar een sleutel die voorloopspaties bevat, moet u wel scheidingstekens gebruiken. Tussen de scheidingstekens neemt u de exacte tekenreeks op, inclusief voorloopspaties. U kunt ook een tekenreeks of een numerieke waarde opslaan in een geheugenvariabele en vervolgens die geheugenvariabele opgeven als argument bij het commando FIND.

De zoekcommando's FIND, SEEK en LOCATE zijn bedoeld voor verschillende situaties. FIND en SEEK zijn sneller dan LOCATE, maar kunnen alleen onder bepaalde omstandigheden worden toegepast. Voor beide is een geïndexeerd bestand nodig en beide kunnen alleen zoeken naar waarden in sleutelvelden. SEEK is flexibeler dan FIND omdat SEEK elke willekeurige uitdrukking accepteert.

Als de informatie die u zoekt in een niet-geïndexeerd bestand of niet in het sleutelveld van een geïndexeerd bestand staat, moet u LOCATE gebruiken. LOCATE accepteert een uitdrukking van elk willekeurig gegevenstype als invoer en kan in elk veld van de tabel zoeken. Bij het zoeken in grote tabellen kan LOCATE behoorlijk traag zijn. In die gevallen kunt u INDEX gebruiken om een nieuwe index te maken en vervolgens zoeken met SEEK of FIND.

Voorbeeld

In het volgende voorbeeld wordt FIND gebruikt om het eerste record te zoeken met een klant in het noordwesten:

```
USE Klanten EXCLUSIVE
INDEX ON Regio TAG Regio
FIND NW                  && Eerste klant in;
                            noordwesten zoeken
IF FOUND()
   ? "Alle klanten in regio Noordwest"
   LIST FIELDS Naam, Regio ;
     WHILE Regio="NW"
ELSE
   ? "Regio Noordwest niet gevonden"
ENDIF
```

Zie ook

CONTINUE, EOF( ), FOUND( ), LOCATE, SEEK, SEEK( ), SET EXACT, SKIP

FKLABEL( )

Toetsenbord- en muisacties

Geeft de naam van een programmeerbare functietoets als resultaat.

Syntaxis

FKLABEL(<*Nuitdr*>)

<Nuitdr>

De programmeerbare functietoets waarvan u de naam wilt weten. FKLABEL(0) geeft de naam van de eerste functietoets als resultaat, FKLABEL(1) van de tweede enzovoort.

Beschrijving

Met FKLABEL( ) kunt u de naam als resultaat geven die door het systeem is toegewezen aan een functietoets. U kunt het resultaat dan gebruiken bij SET FUNCTION, SET KEY of ON KEY LABEL om een instructie of commando toe te wijzen aan de functietoets.

Omdat verschillende computerfabrikanten verschillende namen toewijzen aan functietoetsen, is een applicatie waarin naar functietoetsen naar standaardnamen van functietoetsen wordt verwezen niet volledig overdraagbaar. FKLABEL( ) haalt de naam van de functietoets op uit het huidige systeem, zodat de naam altijd correct is als u de applicatie overbrengt naar andere systemen.

Op de meeste IBM-compatibele computers zijn de volgende toetsen programmeerbare functietoetsen *F1* tot en met *F10*, *Ctrl-F1* tot en met *Ctrl-F10* en *Shift-F1* tot en met *Shift-F10*. In dBASE voor Windows kunt u met ON KEY en SET KEY ook *Alt-F1* tot en met *Alt-F10* programmeren. dBASE voor Windows ondersteunt niet het toewijzen van waarden aan *F11* en *F12*.

Voorbeeld

In de volgende voorbeelden wordt FKLABEL( ) toegepast in het commando SET FUNCTION:

```
SET FUNCTION FKLABEL(1) TO "Goedemorgen allemaal"
SET FUNCTION FKLABEL(2) TO "DO MijnSub;"
* Op F2 drukken is gelijk aan
* "Goedemorgen allemaal" typen
* Als u F3 wordt gedrukt, wordt subroutine MijnSub aangeroepen
```

Let op het volgende:

```
? FKLABEL(0)        &&  Geeft F1 als resultaat
? FKLABEL(9)        &&  Geeft F10 als resultaat
? FKLABEL(10)       &&  Geeft CTRL-F1 als resultaat
? FKLABEL(19)       &&  Geeft CTRL-F10 als resultaat
? FKLABEL(20)       &&  Geeft SHIFT-F1 als resultaat
? FKLABEL(29)       &&  Geeft SHIFT-F10 als resultaat
```

Zie ook

FKMAX( ), ON KEY, SET FUNCTION, SET KEY

FKMAX( )

Toetsenbord- en muisacties

Geeft het nummer van een programmeerbare functietoets als resultaat.

Syntaxis

FKMAX( )

Beschrijving

Computers van verschillende fabrikanten beschikken over verschillende functietoetsen. Dat is een factor om rekening mee te houden als u applicaties probleemloos wilt laten werken op andere systemen. Met FKMAX( ) bepaalt u hoeveel functietoetsen beschikbaar zijn op het huidige systeem, zodat u alternatieven kunt verzinnen als bepaalde functietoetsen op een systeem niet aanwezig zijn. Met SET FUNCTION, SET KEY of ON KEY LABEL kunt u een instructie of commando toewijzen aan een functietoets.

Op de meeste IBM-compatibele computers zijn de volgende toetsen programmeerbare functietoetsen *F1* tot en met *F10*, *Ctrl-F1* tot en met *Ctrl-F10* en *Shift-F1* tot en met *Shift-F10*. In dBASE voor Windows kunt u met ON KEY en SET KEY ook *Alt-F1* tot en met *Alt-F10* programmeren. dBASE voor Windows ondersteunt niet het toewijzen van waarden aan *F11* en *F12*.

Voorbeeld

STORE FKMAX() TO FtTeller

Zie ook

FKLABEL( ), ON KEY, SET FUNCTION, SET KEY

FLDCOUNT( )

Velden en records

Geeft het aantal velden in een tabel als resultaat.

Syntaxis

FLDCOUNT([<*alias*>])

<*alias*>

Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

FLDCOUNT( ) geeft het aantal velden in de huidige of een opgegeven tabel als resultaat. Als u geen werkgebied opgeeft, wordt het huidige werkgebied gebruikt. FLDCOUNT( ) geeft 0 als resultaat als in het opgegeven werkgebied geen tabel is geopend.

Voorbeeld

In het volgende voorbeeld wordt FLDCOUNT( ) gebruikt om het aantal velden in twee tabellen te bepalen. Het resultaat wordt gebruikt om een eendimensionale array te definiëren waarin alle veldnamen worden opgeslagen:

```
SET TALK OFF
USE Bedrijf IN 1
AantalVelden = FLDCOUNT(1)
USE Contact IN 2
AantalVelden = AantalVelden + FLDCOUNT(2)
DECLARE Veld_Arr[AantalVelden]
x = 1
FOR Tabel = 1 TO 3
   FOR x_veld = 1 TO FLDCOUNT(Tabel)
      Veld_Arr[x] = FIELD(x_veld,Tabel)
      x = x + 1
   NEXT
NEXT
Teller = 1
DO WHILE Teller <= AantalVelden     && Inhoud van array tonen
  ? Veld_Arr[Teller]
  Teller=Teller+1
ENDDO
SET TALK ON
CLOSE ALL
CLEAR ALL
```

Zie ook

FIELD( ), DISPLAY STRUCTURE, LIST STRUCTURE, RECCOUNT( ), TYPE( )

FLDLIST( )

Velden en records

Geeft de velden en rekenvelduitdrukkingen uit een lijst die is ingesteld met SET FIELDS TO als resultaat.

Syntaxis

FLDLIST([<*veldnummer Nuitdr*>])

<veldnummer Nuitdr>
De positie van het veld of de rekenvelduitdrukking in een SET FIELDS TO-lijst waarvan u de naam als resultaat wilt geven. Als u dit argument achterwege laat, geeft FLDLIST( ) de volledige lijst als resultaat (met een maximum van 254 tekens).

Beschrijving

FLDLIST( ) geeft het veld of de rekenvelduitdrukking uit een SET FIELDS TO-lijst als resultaat dat overeenkomt met een opgegeven veldnummer. Als u geen argument opgeeft, geeft FLDLIST( ) de volledige lijst als resultaat. maximaal worden 254 tekens als resultaat gegeven. De veldnamen en rekenvelduitdrukkingen worden in de veldenlijst gescheiden door komma's. FLDLIST( ) geeft altijd volledige veldnamen als resultaat. Dat wil zeggen, met tabel- of aliasnaam. In het geval van velden met alleen toegang voor lezen, voegt FLDLIST( ) "/R" toe aan de veldnaam.

FLDLIST( ) geeft zelfs een waarde als resultaat als SET FIELDS is uitgeschakeld (OFF). Als het opgegeven veldnummer hoger is dan het aantal elementen in de veldenlijst, geeft FLDLIST( ) een lege tekenreeks ("") als resultaat.

Voorbeeld

In het volgende voorbeeld wordt FLDLIST( ) met en zonder een aliasnaam gebruikt:

```
USE Bedrijf IN 1
USE Contact Alias Vrtws IN 2
SELECT 1
SET FIELDS TO Bedrijf, Plaats
? FLDLIST(1)          && Geeft Bedrijf->Bedrijf als resultaat
SELECT 2
SET FIELDS TO CompCode, Contact
? FLDLIST(1)          && Geeft Bedrijf->Bedrijf als resultaat
? FLDLIST(2)          && Geeft Bedrijf->Plaats als resultaat
? FLDLIST(3)          && Geeft Vrtws->CompCode als resultaat
? FLDLIST()           && Geeft de volledige lijst als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

SET FIELDS

FLENGTH( )

Velden en records

Geeft de lengte van een veld op een opgegeven plaats in de tabel als resultaat.

Syntaxis

FLENGTH(<*veldnummer Nuitdr*> [, <*alias*>])

<veldnummer Nuitdr>
De positie van het veld waarvan u de lengte wilt weten. Velden worden genummerd van 1 tot en met 1024, al naar gelang hun positie in de tabelstructuur.

<alias>
Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

FLENGTH( ) geeft de lengte van een veld in een tabel als resultaat. Het veld wordt aangegeven door de parameter *<veldnummer Nuitdr>*. De veldlengte voor numerieke en zwevende velden omvat ook de cijfers achter de komma plus één positie voor de decimaalscheider. FLENGTH( ) geeft 1 als resultaat voor datum- en memovelden, en 0 als op de opgegeven positie geen veld in de tabel staat.

Als u geen werkgebied opgeeft, geeft FLENGTH( ) de lengte van het veld in de huidige tabel als resultaat.

Voorbeeld

In het volgende voorbeeld wordt FLENGTH( ) gebruikt om de lengte van het veld Bedrijf (in de tabelstructuur) te bepalen. Deze waarde wordt gebruikt om het aantal streepjes te berekenen door er het aantal tekens in het veld van het huidige record (berekend met LEN(TRIM(Bedrijf)) van af te trekken. Vervolgens wordt daar 13 bij opgeteld vanwege de marge van 15 spaties tussen de kolommen die wordt bepaald door de uitdrukking AT FLENGTH(2)+15:

```
USE Afnemers
? CENTER("Lijst van bedrijven en contactpersonen")
?
SCAN
? Trim(Bedrijf)+REPLICATE("-",(FLENGTH(2)-LEN(TRIM;
  (Bedrijf))+13)), Contact AT FLENGTH(2)+15
ENDSCAN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

FDECIMAL( ), FIELD( )

FLOAT( )

Uitdrukkingen en gegevenstypeconversie

Geeft voor een opgegeven getal een zwevend getal als resultaat.

Syntaxis

FLOAT(<*Nuitdr*>)

<Nuitdr>

Een numeriek of zwevend getal dat u wilt omzetten in een zwevend getal.

Beschrijving

Gebruik FLOAT( ) om een decimaalscheider en decimalen (als nullen) toe te voegen aan een geheel getal. FLOAT(5) geeft bijvoorbeeld 5,00 als resultaat als het aantal decimalen (ingesteld met SET DECIMAL) 2 is. Als u een zwevend getal doorgeeft aan FLOAT( ), krijgt u hetzelfde getal als resultaat.

Voorbeeld

In het volgende voorbeeld wordt FLOAT( ) gebruikt om te zorgen dat het resultaat wordt weergegeven of afgedrukt met 2 decimalen:

```
USE Landen
SET DECIMALS TO 2
? CENTER("BNP per inwoner per land ")
? CENTER("in guldens (* duizend)")
?
SCAN
  ? Naam AT 20, FLOAT((BNP*1000000)/Bevolking) AT 40
ENDSCAN
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. In dBASE IV converteert FLOAT( ) BCD-getallen (binary coded decimals, binair gecodeerde decimale getallen) naar zwevende getallen. Deze functie is in dBASE voor Windows niet nodig omdat er geen getallen in de BCD-notatie worden opgeslagen.

Zie ook

INT( ), SET DECIMALS

FLOCK( )

Gedeelde gegevens

Vergrendelt de huidige tabel of een tabel die wordt aangeduid door een alias. Als de vergrendeling slaagt, wordt .T. als resultaat gegeven.

Syntaxis

FLOCK([<*alias*>])

<alias>
Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan. Als u geen *<alias>* opgeeft, vergrendelt FLOCK( ) de huidige tabel.

Beschrijving

Gebruik FLOCK( ) om de huidige tabel of een tabel die wordt aangeduid door een alias te vergrendelen.

Als u een tabel vergrendelt met FLOCK( ), kunt u alleen de tabel wijzigen. In tegenstelling tot USE...EXCLUSIVE en SET EXCLUSIVE ON staat FLOCK( ) andere gebruikers echter wel toe de vergrendelde tabel te bekijken terwijl u bezig bent. Als u een tabel vergrendelt met FLOCK( ), blijft die vergrendeld tot u UNLOCK gebruikt of de tabel sluit.

FLOCK( ) komt overeen met RLOCK( ), maar FLOCK( ) vergrendelt de volledige tabel terwijl RLOCK( ) alleen bepaalde records van de tabel vergrendelt. FLOCK( ) kunt u dus gebruiken als het nodig is dat alleen u toegang hebt tot een volledige tabel of gerelateerde tabellen. Dat kan bijvoorbeeld gebeuren als u door middel van een gemeenschappelijke sleutel meerdere tabellen wijzigt die aan elkaar zijn gerelateerd.

Alle commando's waarmee gegevens in een tabel kunnen worden gewijzigd, hebben tot gevolg dat automatisch wordt geprobeerd een record- of bestandsvergrendeling aan te brengen. Als de record- of bestandsvergrendeling niet lukt, wordt een fout als resultaat gegeven. U kunt FLOCK( ) gebruiken voor het onderscheppen van acties. U kunt dan het resultaat van de functie testen.

FLOCK( ) kan een gemeenschappelijke tabel zelfs vergrendelen als een andere gebruiker gegevens in de tabel bekijkt. FLOCK( ) slaagt er alleen niet in een vergrendeling aan te brengen als een andere gebruiker nadrukkelijk de tabel of een record in de tabel heeft vergrendeld, of een commando heeft gegeven dat automatisch de tabel of een record in de tabel vergrendelt.

Als bij SET REPROCESS de waarde 0 is ingeschakeld (de standaardinstelling) en FLOCK( ) is niet onmiddellijk in staat de tabel te vergrendelen, wordt u gevraagd of nogmaals een poging moet worden ondernomen of dat het commando moet worden geannuleerd. Tot u het commando annuleert, blijft FLOCK( ) proberen de tabel te vergrendelen. Met SET REPROCESS kunt u deze vraag uitschakelen of het aantal pogingen instellen. Als FLOCK( ) .F. als resultaat geeft, kunt u opnieuw het commando FLOCK( ) geven.

Als u een relatie heeft ingesteld met een hoofdtabel (met SET RELATION) en vervolgens tracht de tabel te vergrendelen met FLOCK( ), wordt geprobeerd alle subtabellen te vergrendelen. Zie SET RELATION voor meer informatie over het tot stand brengen van een relatie tussen tabellen.

Voorbeeld

In het volgende voorbeeld wordt een lus gebruikt om het bestand te vergrendelen met FLOCK( ).Als dat niet lukt, kan de gebruiker aangeven of een volgende poging moet worden ondernomen:

```
RecordGelezen=.f.
Nogmaals=.t.
SET REPROCESS TO 10
USE Bedrijf SHARED  && Niet exclusief
DO WHILE Nogmaals
  IF FLOCK()        && Kan bestand worden vergrendeld?
    DO BewerkMod      && Ja: Bedrijf.DBF wijzigen
    RecordGelezen=.t.
    Nogmaals=.f.
    UNLOCK          && Andere gebruikers mogen/kunnen
                    && bestand wijzigen.
  ELSE              && FLOCK() geeft .F. als resultaat
    CLEAR
    ? "Bedrijf kan niet worden vergrendeld."
    Wait "Nog een poging? " TO Antw
    IF UPPER(Antw)="J"
      Nogmaals = .t.
    ELSE
      Nogmaals = .f.
    ENDIF
  ENDIF
ENDDO
USE
SET REPROCESS TO 0  && standaardinstelling
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

BEGINTRANS( ), LOCK( ), RLOCK( ), SET EXCLUSIVE, SET LOCK, SET RELATION, SET REPROCESS, UNLOCK, USE

FLOOR( )

Numerieke gegevens

Rondt naar beneden af op het dichtstbijzijnde gehele getal.

Syntaxis

FLOOR(<*Nuitdr*>)

<*Nuitdr*>

Een numeriek of zwevend getal dat u naar beneden wilt afronden op een geheel getal.

Beschrijving

FLOOR( ) geeft als resultaat het dichtstbijzijnde gehele getal dat kleiner is dan of gelijk aan <*Nuitdr*>. Als u een getal doorgeeft met een of meer andere cijfers dan 0, geeft FLOOR( ) het dichtstbijzijnde gehele getal als resultaat dat kleiner is dan het opgegeven

getal. Als u een geheel getal doorgeeft aan FLOOR( ), of een getal met alleen maar nullen achter de komma, wordt het gehele getal als resultaat gegeven.

Enkele voorbeelden (het standaardaantal decimalen is 2):

- FLOOR(2.10) geeft 2,00 als resultaat.
- FLOOR(-2.10) geeft -3,00 als resultaat.
- FLOOR(2.00) geeft 2,00 als resultaat.
- FLOOR(2) geeft 2 als resultaat.
- FLOOR(-2.00) geeft -2,00 als resultaat.

Het aantal decimalen stelt u in met SET DECIMALS.

Als u een positief getal doorgeeft aan FLOOR( ), geeft de functie hetzelfde resultaat als INT( ). Bij de beschrijving van INT( ) staat een tabel waarin INT( ), FLOOR( ), CEILING( ) en ROUND( ) met elkaar worden vergeleken.

De waarde die door FLOOR( ) als resultaat wordt gegeven, heeft hetzelfde gegevenstype als <*Nuitdr*>.

Voorbeeld

In het volgende voorbeeld wordt FLOOR( ) gebruikt om een doorgegeven waarde naar beneden af te ronden:

```
SET DECIMALS TO 2
prijs  = 129.95
BTW    = .175
totaal = prijs + (prijs*BTW)
? " De nieuwe versie kost (ex. BTW) : " + STR(prijs,6,2)
? "(Het BTW-percentage is " + ;
   STR(BTW*100,5,2) + "%)"
? "              Prijs inclusief BTW : " + ;
   STR(totaal,6,2)
?
? "Verkoopprijs 'ZONDER WISSELGELD' : " + ;
   STR(FLOOR(totaal),6,2)
```

Zie INT( ) voor een ander voorbeeld met FLOOR( ).

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. In dBASE IV geeft FLOOR( ) geen decimalen weer, onafhankelijk van de waarde die is ingesteld met SET DECIMALS.

Zie ook

CEILING( ), INT( ), ROUND( ), SET DECIMALS

FLUSH

Schrijft gegevensbuffers naar schijf en maakt niet-toegewezen geheugen vrij.

Syntaxis

FLUSH

Beschrijving

Gebruik FLUSH om de betrouwbaarheid van de gegevens te garanderen en het beschikbare geheugen te optimaliseren.

Als u een tabel en de bijbehorende index- en memobestanden opent, wordt een bepaald aantal records uit het bestand in het geheugen geladen. Tevens wordt van elk geopend indexbestand het deel geladen dat betrekking heeft op de records. Als de buffer vol is of als u tabellen of indexen sluit (met CLOSE DATABASES, USE, CLOSE INDEXES, CLOSE ALL of SET INDEX TO), worden de records terug naar schijf geschreven zodat de wijzigingen worden opgeslagen. Met FLUSH kunt u informatie in de gegevensbuffer opslaan op schijf zonder eerst tabellen of indexen te sluiten. FLUSH slaat informatie op in tabellen en bijbehorende bestanden die zijn geopend in andere werkgebieden dan het huidige werkgebied.

Met FLUSH kunt u belangrijke informatie opslaan op schijf om te voorkomen dat die verloren gaat. Gebruik FLUSH echter niet te vaak, want het vertraagt de uitvoering. In een applicatie voor de invoer van bestellingen waarin slechts enkele bestellingen per uur worden ingevoerd, kunt u FLUSH gebruiken om de gegevens op te slaan voor het geval de stroom per ongeluk wordt uitgeschakeld. Omdat er niet veel bestellingen worden ingevoerd, is de tijd die de uitvoering van FLUSH kost, niet van belang.

Voorbeeld

In het volgende voorbeeld wordt FLUSH gebruikt na twee verschillende invoerprocedures om de buffers naar schijf te schrijven als AUTOSAVE is uitgeschakeld (OFF):

```
USE Bedrijf IN SELECT()
SELECT Bedrijf
SET AUTOSAVE OFF
APPEND BLANK
@  2, 2 SAY "Bedrijfscode" GET Bedrijf->CompCode
@  3, 2 SAY "Bedrijfsnaam" GET Bedrijf->Bedrijf
@  5, 9 SAY "Adres" GET Bedrijf->Straat1
@  6,15 GET Bedrijf->Straat2
@  7, 6 SAY "Postcode" GET Bedrijf->Postcode
@  7,26 SAY "Plaats" GET Bedrijf->Plaats
@  8, 9 SAY "Regio" GET Bedrijf->Regio
READ
FLUSH
CLEAR
APPEND
```

```
FLUSH
CLOSE ALLOverdraagbaarheid
```

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CLEAR, CLOSE..., MEMORY( )

FOPEN( )

Toegang op laag niveau

Opent een opgegeven bestand. Als het bestand is geopend, wordt het nummer van de bestands-handle als resultaat gegeven en anders –1.

Syntaxis

FOPEN(<*bestandsnaam Tuitdr*> [, <*toegang Tuitdr*>])

<*bestandsnaam Tuitdr*>

De naam van het bestand dat u wilt openen. Standaard wordt het bestand in de huidige directory gezocht, wen vervolgens in de directory's die zijn opgegeven met SET PATH. Als u een bestand in een andere directory wilt openen, moet een pad opgeven bij <*bestandsnaam Tuitdr*>.

<*toegang Tuitdr*>

Het toegangsniveau van het bestand dat wordt geopend (zie volgende tabel). *Schrijven* betekent dat u gegevens in het bestand kunt wijzigen (overschrijven) en *toevoegen* betekent dat u gegevens aan het eind van het bestand kunt toevoegen. Als u probeert gegevens te overschrijven in een bestand waarvoor u alleen toestemming hebt voor toevoegen en niet voor schrijven, worden de gegevens toegevoegd aan het eind van het bestand.

<*toegang Tuitdr*>	**Toegangsniveau**
niet opgegeven	Alleen lezen
"R"	Alleen lezen
"W"	Alleen schrijven
"A"	Alleen toevoegen
"RW" of "WR"	Lezen en schrijven
"AR" of "RA"	Lezen en toevoegen
"AW" of "WA"	Schrijven en toevoegen

Beschrijving

Met FOPEN( ) kunt u een opgegeven bestand openen en een toegangsniveau voor dat bestand instellen. Als het bestand wordt geopend, geeft de functie het nummer van de DOS-bestands-handle als resultaat. Als het bestand niet kan worden geopend (omdat het bijvoorbeeld al is geopend), geeft FOPEN( ) -1 als resultaat.

Voordat u low-level-functies als FREAD( ) en FWRITE( ) kunt gebruiken, moet u een bestand eerst openen met FOPEN( ) of FCREATE( ). Beide functies geven de bestandshandle als resultaat die u nodig hebt om door te geven aan andere low-level-functies.

Als u een bestand opent met FOPEN( ), wordt de bestandsaanwijzer bij het eerste teken in het bestand geplaatst. Met FSEEK( ) kunt u de bestandsaanwijzer verplaatsen voordat u leest uit of schrijft naar een bestand.

Voorbeeld

In het volgende voorbeeld wordt FOPEN( ) gebruikt om het LEESMIJ-BESTAND te openen dat in de installatiedirectory staat. Het bestand wordt vervolgens naar de printer gestuurd met SET PRINTER ON, een lus met FEOF( ) en FGETS( ):

```
SET PATH TO C:\DBASEWIN           && Hoofddirectory
IF FILE("Leesmij.TXT")            && Aanwezigheid bestand testen
  nHandle=FOPEN("Leesmij.TXT","R")&& Openen voor alleen lezen
ENDIF
IF nHandle>0                      && Is bestand geopend?
  SET PRINTER TO LPT1             && Uitvoeren naar printer
  DO WHILE .NOT. FEOF(nHandle)
    ? FGETS(nHandle)              && Regel afdrukken
  ENDDO
  SET PRINTER TO
ELSE
  ? "Bestand niet geopend. Bestandsfout #:",;
    LTRIM(STR(FERROR() ))
ENDIF
IF nHandle=-1
  ? "Bestand niet gesloten.  Bestandsfout #:",;
    LTRIM(STR(FERROR() ))
ELSE
  IF FCLOSE(nHandle)
    ?
    ? "Bestand gesloten"
  ELSE
    ? "Bestand niet gesloten. Bestandsfout #:",;
      LTRIM(STR(FERROR() ))
  ENDIF
ENDIF
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FCLOSE( ), FCREATE( ), FERROR( ), FILE( ), GETFILE( ), SET PATH

FOR( )

Tabelindeling

Geeft de FOR-clausule als resultaat waarmee het opgegeven indexbestand of de opgegeven indexlabel is gemaakt.

Syntaxis

FOR([[<*.MDX-bestandsnaam Tuitdr*>,] <*indexpositie Nuitdr*> [,<*alias*>]])

<.MDX-bestandsnaam Tuitdr>

Het .MDX-bestand met de indexlabel die u wilt controleren.

<indexpositie Nuitdr>

De positie van een indexlabel in een .MDX-bestand of de positie van een indexbestand in de lijst met geopende indexbestanden voor de huidige of een opgegeven tabel.

<alias>

Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

FOR( ) geeft de FOR-clausule als resultaat die is gebruikt bij het commando INDEX om een opgegeven .MDX-label te maken voor de huidige of opgegeven tabel. De FOR( ) evalueert de FOR-clausule voor de hoofdindex van de huidige tabel, de indexlabel op een opgegeven positie in de lijst met geopende indexen voor een tabel of de indexlabel in een opgegeven .MDX-bestand.

Als geen naam van een .MDX-bestand is opgegeven, evalueert FOR( ) de index op een opgegeven positie in de lijst van alle geopende indexen in hetzelfde werkgebied. De FOR( ) controleert eerst .NDX- en produktie-indexlabels, en dan indexlabels in andere .MDX-bestanden.

FOR( ) geeft een lege tekenreeks ("") als resultaat als er geen indexlabel staat op de opgegeven positie, als de index op de opgegeven positie een .NDX-bestand is of als de opgegeven indexlabel niet is gemaakt met een FOR-clausule. FOR( ) geeft ook een lege tekenreeks als resultaat als u geen indexlabelpositie opgeeft en de tabel niet over een hoofdindex beschikt.

Met TAGNO( ) kunt u het indexnummer van een opgegeven index of indexlabel bepalen. De volgorde van de geopende indexen voor een opgegeven tabel blijft hetzelfde tot u een andere volgorde opgeeft met USE, SET INDEX of INDEX.

Voorbeeld

In het volgende voorbeeld wordt FOR( ) gebruikt om de voorwaarden in de FOR-clausule van de index te bepalen:

```
DELETE FILE Bedrtijd.mdx
* Maak een tijdelijk .MDX-bestand
```

```
USE Bedrijf EXCLUSIVE
INDEX ON CompCode ;
  TAG CompCode OF Bedrtijd FOR CompCode = "D"
INDEX ON Bedrijf  ;
  TAG Bedrijf  OF Bedrtijd
INDEX ON Plaats     ;
  TAG Plaats     OF Bedrtijd FOR Postcode = "2"
?
? "Label en FOR-clausule"
?
? TAG(1),FOR(1)
* TAG(1) en FOR(1) geven de label en FOR-clausule
* voor de eerste index in het produktie-indexbestand,
* Bedrijf.mdx, als resultaat
? "1", TAG("Bedrtijd",1), FOR("Bedrtijd",1)
* FOR-clausule van COMPCODE-index is CompCode = "D"
? "2", TAG("Bedrtijd",2), FOR("Bedrtijd",2)
* FOR-clausule van BEDRIJF-index is leeg
? "3", TAG("Bedrtijd",3), FOR("Bedrtijd",3)
* FOR-clausule van PLAATS-index is Postcode ="2"
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

INDEX, SET INDEX, SET ORDER, TAG( ), TAGCOUNT( ), TAGNO( ), USE

FOR...NEXT

Programma's

Voert de instructies tussen FOR en NEXT een bij FOR opgegeven aantal malen uit.

Syntaxis

FOR <*variabele*> = <*begin Nuitdr*> TO <*einde Nuitdr*> [STEP <*stap Nuitdr*>]
    <*instructies*>
    [LOOP]
    [EXIT]
  NEXT

<variabele>

De geheugenvariabele die bij elke doorgang door de lus wordt verhoogd of verlaagd en vervolgens getest. Als de lus op een normale manier wordt beëindigd (dus niet met het commando EXIT), is de waarde van <*variabele*> de hoeveelheid <*stap Nuitdr*> groter of kleiner dan <*einde Nuitdr*>.

<begin Nuitdr>

De beginwaarde van <*variabele*>.

<einde Nuitdr>

De uiteindelijke waarde van *<variabele>*. Als u deze waarde wijzigt in de FOR-lus, wordt dat niet opgemerkt en blijft de uitvoering doorgaan tot de oorspronkelijk opgegeven waarde wordt bereikt.

STEP *<stap Nuitdr>*

Definieert de grootte van elke verhoging of verlaging, de stap *<stap Nuitdr>*. Met dit getal wordt *<variabele>* na elke doorgang door de lus verhoogd of verlaagd. Standaard wordt *<variabele>* telkens verhoogd met 1. Als u deze waarde wijzigt in de FOR-lus, wordt dat niet opgemerkt en blijft de oorspronkelijke waarde voor *<stap Nuitdr>* van kracht.

Als *<stap Nuitdr>* positief is, wordt *<variabele>* verhoogd tot deze waarde groter is dan *<einde Nuitdr>*. Als *<stap Nuitdr>* negatief is, wordt *<variabele>* verlaagd tot deze waarde kleiner is dan *<einde Nuitdr>*.

Het argument *<stap Nuitdr>* kan een geheel getal, een breuk of een zwevend getal zijn. STEP *<stap Nuitdr>* moet op dezelfde regel staan als het commando FOR.

<instructies>

Programmaregels met een willekeurige combinatie van commando's, functies, door de gebruiker gedefinieerde functies en LOOP- en EXIT-opties.

LOOP

Geeft de besturing terug aan het begin van de FOR-lus. *<variabele>* wordt verhoogd of verlaagd met *<stap Nuitdr>* zonder de overige instructies tussen LOOP en NEXT uit te voeren.

EXIT

Geeft de besturing door aan de regel die volgt op NEXT zonder *<variabele>* te evalueren of de instructies tussen EXIT en NEXT uit te voeren. De waarde van *<variabele>* verandert niet meer nadat EXIT is uitgevoerd.

NEXT

Een verplicht commando dat het eind aangeeft van de FOR-lus. Als NEXT wordt bereikt, wordt *<variabele>* verhoogd of verlaagd aan de hand van de bijbehorende FOR-instructie. Daarna wordt de besturing teruggegeven aan de FOR-instructie als de waarde van *<variabele>* nog niet zijn boven- of ondergrens heeft bereikt. Is die waarde wel bereik, dan wordt de besturing doorgegeven aan de volgende regel (die na het commando NEXT).

Beschrijving

Gebruik FOR...NEXT om een blok instructies een opgegeven aantal malen uit te voeren. Als een FOR-lus wordt aangetroffen, wordt *<variabele>* gelijk gemaakt aan *<begin Nuitdr>* en worden de volgende stappen uitgevoerd:

- Als *<variabele>* ligt binnen het bereik van *<begin Nuitdr>* tot *<einde Nuitdr>*, worden de instructies tussen FOR en NEXT een voor een uitgevoerd tot LOOP, EXIT of NEXT wordt bereikt.
- Als LOOP of NEXT wordt bereikt, wordt *<variabele>* verhoogd of verlaagd met *<stap Nuitdr>* (of verhoogd met 1 als u de optie STEP niet hebt gebruikt). De besturing wordt daarna teruggegeven aan de regel met FOR.
- Als *<variabele>* nog steeds binnen het bereik van *<begin Nuitdr>* tot *<einde Nuitdr>* valt, worden de instructies in de lus opnieuw uitgevoerd.
- Als *<variabele>* groter is dan *<einde Nuitdr>* (of kleiner dan *<einde Nuitdr>* in het geval *<stap Nuitdr>* negatief is), wordt de FOR-lus verlaten en wordt de instructie na NEXT uitgevoerd.
- Als tijdens de uitvoering van de lus EXIT wordt aangetroffen, wordt de besturing onmiddellijk doorgegeven aan de regel na NEXT. *<variabele>* wordt niet verhoogd of verlaagd en de instructies tussen EXIT en NEXT worden niet uitgevoerd.

In een FOR-lus kunt u lussen en andere structuren nesten, inclusief andere FOR...NEXT-lussen. Binnen een functie of procedure mogen maximaal 20 FOR-lussen worden genest.

U kunt FOR-lussen onder andere gebruiken voor het verplaatsen van een weergave in een venster, voor het verwerken van tekens in een tekenreeks of voor het sequentieel benaderen van tabelrecords of array-cellen.

Voorbeeld

In het volgende voorbeeld wordt ALEN( ) gebruikt in een FOR...NEXT-lus om de inhoud van een array af te drukken:

```
USE Klanten
DECLARE acContact[RECCOUNT() ,1]
COPY TO ARRAY acContact
USE
ToonArray(acContact)
RETURN

FUNCTION ToonArray
PARAMETER avArray
FOR i = 1 to ALEN(avArray)
  ? STR(i,3,0)+" - "+avArray[i,1]
NEXT
RETURN .T.
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DO...UNTIL, DO WHILE, IF, SCAN

FOUND( )

Tabelindeling

Geeft aan of de laatste instructie met FIND, LOCATE, CONTINUE, SEEK, LOOKUP( ) of SEEK( ) een overeenkomst heeft gevonden.

Syntaxis

FOUND([*<alias>*])

<alias>
Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

FOUND( ) geeft .T. als resultaat als LOCATE, CONTINUE, LOOKUP( ), FIND, SEEK of SEEK( ) een overeenkomst heeft gevonden in de huidige of opgegeven tabel. FOUND( ) geeft .F. als resultaat als in hetzelfde werkgebied nog geen zoekcommando is gebruikt of als de zoekbewerking niets heeft opgeleverd. U kunt zoekbewerkingen uitvoeren in verschillende werkgebieden en in elk de status van de bewerking afzonderlijk testen met FOUND( ).

Als tabellen zijn gerelateerd met behulp van het commando SET RELATION TO, zoekt dBASE voor Windows in de gerelateerde tabellen zodra u de recordaanwijzer in de actieve tabel verplaatst door middel van de commando's FIND, LOCATE, SEEK of CONTINUE of de functies SEEK( ) of LOOKUP( ). Als u de recordaanwijzer verplaatst met een ander commando of een andere functie, geeft FOUND( ) .F. als resultaat.

Als SET NEAR is ingeschakeld (ON), geeft FOUND( ) de volgende resultaten:

- .T., als er een exacte overeenkomst wordt aangetroffen.
- .F., als de overeenkomst niet exact is en de recordaanwijzer is verplaatst naar het record waarvan de sleutel onmiddellijk volgt op de waarde waarop werd gezocht.

Als SET NEAR is uitgeschakeld (OFF), geeft FOUND( ) .F. als resultaat als geen overeenkomst wordt aangetroffen.

Voorbeeld

Zie de voorbeelden bij SEEK, FIND en LOCATE voor een voorbeeld met FOUND( ).

Zie ook

CONTINUE, EOF( ), FIND, LOCATE, LOOKUP( ), SEEK, SEEK( ), SET NEAR, SET RELATION

FPUTS( )

Schrijft een tekenuitdrukking en een of twee einde-regeltekens naar een bestand dat eerder is geopend met FCREATE( ) of FOPEN( ). De bestandsaanwijzer wordt na het laatste geschreven teken geplaatst. Als de functie is geslaagd, wordt het geschreven aantal tekens als resultaat gegeven. Als de functie niet lukt, is het resultaat 0 en als er een fout is opgetreden is het resultaat -1.

Syntaxis

FPUTS(*<bestands-handle Nuitdr>*, *<tekenreeks Tuitdr>*
[, *<tekens Nuitdr>*] [, *<einde-regel uitdr>*])

<bestands-handle Nuitdr>

Het nummer van de bestands-handle van het bestand waar de opgeven tekens en de einde-regeltekens naar toe moeten worden geschreven. Als u een bestand opent met FCREATE( ) of FOPEN( ), geven deze functies een bestands-handle-nummer als resultaat. Gebruik dit getal als *<bestands-handle Nuitdr>*. Als u een bestands-handle-nummer opgeeft dat niet eerder als resultaat is gegeven door FCREATE( ) of FOPEN( ), wordt een foutmelding getoond.

<tekenreeks Tuitdr>

De tekenuitdrukking om naar het opgegeven bestand te schrijven. Als u slechts een deel van *<tekenreeks Tuitdr>* naar het bestand wilt schrijven, moet u het argument *<tekens Nuitdr>* gebruiken.

<tekens Nuitdr>

Het aantal tekens van de opgegeven tekenuitdrukking *<tekenreeks Tuitdr>* dat naar het opgegeven bestand moet worden geschreven, te beginnen bij het eerste teken in *<tekenreeks Tuitdr>*. Het geldige bereik loopt van 0 tot en met 32766.

<einde-regel uitdr>

De uit een of twee tekens bestaande einde-regelindicatie die na de opgegeven tekenreeks naar het bestand moet worden geschreven. De volgende tabel geeft een overzicht van de standaardcodes voor einde-regelindicaties. Zet ze niet tussen aanhalingstekens. U kunt twee tekens combineren met een plusteken (+), zoals CHR(141) + CHR(138).

<einde-regel uitdr>	**Betekenis**
niet geleverd	Harde regelterugloop/regeldoorvoer (0D0A Hex)
CHR(141)	Zachte regelterugloop (V.S.) (8D Hex)
CHR(255)	Zachte regelterugloop (Europa) (FF Hex)
CHR(138)	Zachte regeldoorvoer (V.S.) (8A Hex)
CHR(0)	Zachte regeldoorvoer (Europa) (00 Hex)
CHR(13)	Harde regelterugloop (0D Hex)
CHR(10)	Harde regeldoorvoer (0A Hex)

Het is niet mogelijk rechtstreeks hexadecimale getallen op te geven voor *<einde-regel uitdr>*, maar u kunt wel decimale voorstellingen combineren. 0D0A Hex is bijvoorbeeld gelijk aan CHR(13) + CHR(10); CHR(13) is 0D Hex en CHR(10) is 0A Hex. Gebruik HTOI( ) om een hexadecimaal getal te converteren naar zijn decimale waarde.

Beschrijving

FPUTS( ) schrijft een tekenreeks en een of twee einde-regeltekens naar een bestand. Als de bewerking is geslaagd, wordt het aantal geschreven bytes als resultaat gegeven (inclusief de einde-regeltekens). Als u geen toestemming hebt om naar het bestand te schrijven of aan het bestand toe te voegen, geeft FPUTS( ) 0 als resultaat. Gebruik u vervolgens FERROR( ), dan geeft die functie 5 als resultaat. Als er een fout optreedt, geeft FPUTS( ) -1 als resultaat. Als FPUTS( ) is uitgevoerd, staat de bestandsaanwijzer bij het teken dat onmiddellijk volgt op het teken dat als laatste naar het bestand is geschreven.

Met uitzondering van een eigenschap zijn FPUTS( ) en FWRITE( ) onderling uitwisselbaar. FPUTS( ) schrijft na de tekenreeks altijd een einde-regelindicatie naar het bestand, terwijl FWRITE( ) dat niet doet.

FPUTS( ) schrijft naar het bestand vanaf de huidige positie van bestandsaanwijzer. Met FSEEK( ) kunt u de bestandsaanwijzer verplaatsen voordat of nadat u FPUTS( ) gebruikt.

Als de bestandsaanwijzer aan het eind van het bestand staat, voegt FPUTS( ) de opgegeven tekenreeks en de einde-regelindicatie toe aan het eind van bestand. Als de bestandsaanwijzer niet aan het eind van het bestand staat en u hebt toestemming tot schrijven in het bestand, overschrijft FPUTS( ) bestaande tekst bij de positie van de bestandsaanwijzer. Als u toestemming hebt tot toevoegen aan het bestand (maar niet tot schrijven), voegt FPUTS( ) de tekst toe aan het eind van het bestand, onafhankelijk van de positie van de bestandsaanwijzer. In dat geval wordt geen fout als resultaat gegeven.

Voorbeeld

In het volgende voorbeeld wordt een tekstbestand gemaakt met de naam TEST.TXT. Aan dit bestand wordt een telkens groter aantal tekens uit de geheugenvariabele "Reeks" toegevoegd. Bij de eerste doorgang door de FOR...NEXT-lus wordt het teken "1" uit "Reeks" plus een zachte regelterugloop en een zachte regeldoorvoer toegevoegd. Bij elke volgende doorgang door de lus wordt telkens één teken meer uit "reeks" toegevoegd tot i gelijk is aan 9. Het programma gebruikt vervolgens FGETS in een .NOT. FEOF( )-lus om de inhoud van TEST.TXT weer te geven:

```
nHandle=FCREATE("Test.TXT","RW")  && Bestand maken
* Invoeren tekstgegevens
Reeks="123456789"
FOR i=1 TO 9                      && Lus voor 9 regels
  FPUTS(nHandle,Reeks,i,CHR(13)+CHR(10))
* i tekens uit Reeks gevolgd door zachte
* regelterugloop/regeldoorvoer toevoegen
NEXT i                            && i verhogen
* Weergeven tekstgegevens
FSEEK(nHandle,0,0)                && Aanwijzer verplaatsen naar;
                                     begin van bestand
```

```
CLEAR                              && Resultatenpaneel van;
                                      commandovenster legen

DO WHILE .NOT. FEOF(nHandle)
  ? FGETS(nHandle)                 && Regel weergeven
ENDDO
FCLOSE(nHandle)                    && Test.TXT sluiten
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. Als in dBASE IV geen toestemming tot schrijven of toevoegen is gegeven voor een bestand, slaagt FPUTS( ) niet en wordt een fout als resultaat gegeven.

Zie ook

FCREATE( ), FEOF( ), FERROR( ), FGETS( ), FOPEN( ), FSEEK( ), FWRITE( ), HTOI( ), SUBSTR( )

FREAD( )

Toegang op laag niveau

Geeft een opgegeven aantal tekens als resultaat uit een bestand dat eerder is geopend met FCREATE( ) of FOPEN( ). De bestandsaanwijzer wordt na het laatste teken geplaatst dat als resultaat is gegeven.

Syntaxis

FREAD(*<bestands-handle Nuitdr>*, *<tekens Nuitdr>*)

<bestands-handle Nuitdr>

Het nummer van de bestands-handle van het bestand waaruit tekens moeten worden gelezen en als resultaat gegeven. Als u een bestand opent met FCREATE( ) of FOPEN( ), geven deze functies een bestands-handle-nummer als resultaat. Gebruik dit getal als *<bestands-handle Nuitdr>*. Als u een bestands-handle-nummer opgeeft dat niet eerder als resultaat is gegeven door FCREATE( ) of FOPEN( ), wordt een foutmelding getoond.

<tekens Nuitdr>

Het aantal tekens dat uit het opgegeven bestand als resultaat moet worden gegeven. Waarden van 0 tot 32766 zijn toegestaan; als *<tekens Nuitdr>* kleiner dan 0 of groter dan 32766 is, wordt respectievelijk 0 of 32766 gebruikt.

Beschrijving

FREAD( ) geeft het opgegeven aantal tekens als resultaat uit het opgegeven bestand. FREAD( ) begint met lezen bij de huidige positie van bestandsaanwijzer en plaatst de bestandsaanwijzer vervolgens direct na het laatste gelezen teken. Met FSEEK( ) kunt u de bestandsaanwijzer verplaatsen voordat of nadat u FREAD( ) gebruikt.

Met uitzondering van een eigenschap zijn FREAD( ) en FGETS( ) onderling uitwisselbaar. FREAD( ) geeft ook de einde-regelindicatie als resultaat, terwijl FGETS( ) dat niet doet. Zie FGETS( ) voor meer informatie.

Voorbeeld

In het volgende voorbeeld wordt FREAD( ) gebruikt om per keer 25 tekens uit het bestand LEESMIJ.TXT weer te geven. Het commando WAIT binnen de lus zorgt dat de gebruiker het bestand met een pagina tegelijk kan lezen:

```
SET PATH TO C:\DBASEWIN          && Hoofddirectory
nHandle=FOPEN("Leesmij.TXT","R") && Openen voor alleen lezen
DO WHILE .NOT. FEOF(nHandle)
  ? FREAD(nHandle,25)            && Volgende 25 tekens;
                                    weergeven
  ?                              && Lege regel boven Wait
  WAIT "Blader op uw gemak. ;
    Druk op een toets om door te gaan."
  ?                              && Lege regel onder Wait
ENDDO
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FCREATE( ), FEOF( ), FERROR( ), FGETS( ), FOPEN( ), FSEEK( ), FWRITE( )

FSEEK( )

Toegang op laag niveau

Verplaatst de bestandsaanwijzer een opgegeven aantal bytes verder of terug in een bestand dat eerder is geopend met FCREATE( ) of FOPEN( ). De functie geeft het aantal bytes vanaf het begin van het bestand tot de bestandsaanwijzer als resultaat.

Syntaxis

FSEEK(*<bestands-handle Nuitdr>*, *<bytes Nuitdr>* [, *<positie Nuitdr>*])

<bestands-handle Nuitdr>

Het nummer van de bestands-handle van het bestand waarin u de bestandsaanwijzer wilt verplaatsen. Als u een bestand opent met FCREATE( ) of FOPEN( ), geven deze functies een bestands-handle-nummer als resultaat. Gebruik dit getal als *<bestands-handle Nuitdr>*. Als u een bestands-handle-nummer opgeeft dat niet eerder als resultaat is gegeven door FCREATE( ) of FOPEN( ), wordt een foutmelding getoond.

<bytes Nuitdr>
Het aantal bytes waarmee u de bestandsaanwijzer in het opgegeven bestand wilt verplaatsen. Als *<bytes Nuitdr>* negatief is, wordt de bestandsaanwijzer in de richting van het begin van het bestand verplaatst. Als *<bytes Nuitdr>* 0 is, wordt de bestandsaanwijzer verplaatst naar de positie die u opgeeft met *<positie Nuitdr>*. Als *<bytes Nuitdr>* positief is, wordt de bestandsaanwijzer in de richting van het eind van het bestand verplaatst (of voorbij het eind van het bestand).

<positie Nuitdr>
Het cijfer 0, 1 of 2. Deze cijfers geven respectievelijk een positie aan ten opzichte van het begin van het bestand (0), de huidige positie van de bestandsaanwijzer (1) of het eind van het bestand (2). De standaardinstelling is 0.

Beschrijving

FSEEK( ) verplaatst de bestandsaanwijzer in het opgegeven bestand. De functie geeft het aantal bytes vanaf het begin van het bestand tot de nieuwe positie van de bestandsaanwijzer als resultaat. Als er een fout optreedt, geeft FSEEK( ) -1 als resultaat.

De verplaatsing is relatief ten opzichte van het begin van het bestand tenzij u iets anders opgeeft met *<positie Nuitdr>*. De instructie FSEEK(bestnum, 5) verplaatst de bestandsaanwijzer vijf tekens voorbij het begin van het bestand terwijl FSEEK(bestnum, vijf, 1) de aanwijzer vijf tekens voorbij de huidige positie verplaatst. U kunt de bestandsaanwijzer wel voorbij het eind van het bestand plaatsen, maar niet vóór het begin van het bestand.

Met FSEEK(*<bestands-handle Nuitdr>*, 0) verplaatst u de bestandsaanwijzer naar het begin van het bestand en met FSEEK(*<bestands-handle Nuitdr>*, 0, 2) naar het eind van het bestand.

Ook FGETS( ), FPUTS( ), FREAD( ) en FWRITE( ) verplaatsen de bestandsaanwijzer als ze lezen uit of schrijven naar het bestand.

Voorbeeld

Zie FGETS( ) voor een voorbeeld met FSEEK( ).

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FCREATE( ), FEOF( ), FOPEN( ), FREAD( ), FWRITE( )

FSIZE( )

Stations- en bestandsfuncties

Geeft de grootte van een bestand in bytes als resultaat.

Syntaxis

FSIZE(<*bestandsnaam Tuitdr*>)

<*bestandsnaam Tuitdr*>

De naam van het bestand om te evalueren. Jokertekens worden niet ondersteund.

Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, wordt geen extensie gebruikt.

Beschrijving

Met FSIZE( ) kunt u de grootte van een bestand op schijf bepalen.

Als u een bestand wijzigt, heeft dat meestal gevolgen voor de lengte van het bestand. De lengte van een bestand verschijnt naast de bestandsnaam in een directorylijst. Als de gebruiker een tabel wijzigt en het tabelbestand wordt gesloten, heeft het bestand in de meeste gevallen een andere lengte. FSIZE( ) geeft de huidige bestandslengte als resultaat.

Als het bestand een extensie heeft, moet u die ook opgeven bij <*bestandsnaam Tuitdr*>. Als het bestand niet op het huidige station staat, moet u de naam van een station opgeven. Als het bestand niet in de huidige directory of in het met SET PATH ingestelde pad staat, moet u ook de directory opgeven.

Als het bestand niet wordt gevonden, wordt een fout als resultaat gegeven. U kunt met FILE( ) de aanwezigheid van een bestand testen voordat u FSIZE( ) gebruikt. FLUSH is niet van invloed op de lengte van een bestand.

Als <*bestandsnaam Tuitdr*> in de huidige directory staat en tevens in een directory die is ingesteld met SET PATH, geeft FSIZE(<*bestandsnaam Tuitdr*>) zonder pad informatie over het bestand in de huidige directory.

Voorbeeld

In het volgende voorbeeld wordt de bestandslengte van de tabel Bedrijf bepaald. Als op de diskette in station B: nog voldoende ruimte is, wordt een kopie van de tabel op die diskette gezet:

```
Nodig1=FSIZE("Bedrijf.dbf")
Nodig2=FSIZE("Bedrijf.dbt")  && het memobestand
Nodig=Nodig1+Nodig2
Beschikbaar=DISKSPACE(2)
IF Nodig <= Beschikbaar
  RUN COPY Bedrijf.dbf B:
  RUN COPY Bedrijf.dbt B:
ELSE
  ? "Beschikbaar", Beschikbaar
  ? "Nodig      ", Nodig
  WAIT "WAARSCHUWING. Onvoldoende ruimte op diskette in B:"
ENDIF
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FDATE( ), FILE( ), FLUSH, FTIME( ), SET DIRECTORY, SET PATH

FTIME( )

Stations- en bestandsfuncties

Geeft het tijdstempel van een opgegeven bestand als resultaat.

Syntaxis

FTIME(<*bestandsnaam Tuitdr*>)

<*bestandsnaam Tuitdr*>

De naam van het bestand om te evalueren. Jokertekens worden niet ondersteund.

Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, wordt geen extensie gebruikt.

Beschrijving

Met FTIME( ) kunt u bepalen op welke moment op de dag de laatste wijziging is aangebracht in een bestand op schijf.

Als u een bestand wijzigt, krijgt het bestand op schijf het tijdstempel dat wordt bepaald door de huidige systeemtijd. Als de gebruiker bijvoorbeeld een tabel wijzigt, wordt het tijdstempel van het tabelbestand gewijzigd als het bestand wordt gesloten. FTIME( ) geeft het tijdstempel als resultaat.

Als het bestand een extensie heeft, moet u die ook opgeven in <*bestandsnaam Tuitdr*>. Als het bestand niet op het standaardstation staat, moet u een station opgeven. Staat het bestand niet in de huidige directory of in het met SET PATH opgegeven pad staat, dan moet u de directory opgeven.

Als het bestand niet wordt gevonden, wordt een fout als resultaat gegeven. U kunt eerste de aanwezigheid van het bestand testen met FILE( ) voordat u FTIME( ) gebruikt. FLUSH is niet van invloed op het tijdstempel.

Als <*bestandsnaam Tuitdr*> in de huidige directory staat, maar ook voorkomt in een directory die is ingesteld met SET PATH, geeft FDATE(<*bestandsnaam Tuitdr*>) zonder pad informatie over het bestand in de huidige directory als resultaat.

Voorbeeld

In het volgende voorbeeld worden de datum- en tijdstempels vergeleken van BEDRIJF.DBF en de reservekopie van dat bestand op station B:. Als de reservekopie van

dezelfde dag is als de huidige tabel maar van een vroeger tijdstip, wordt geadviseerd een reservekopie te maken:

```
DO CASE
CASE FDATE("B:Bedrijf.dbf") = FDATE("Bedrijf.dbf");
 .AND.;
    FTIME("B:Bedrijf.dbf") < FTIME("Bedrijf.dbf")
  * dezelfde dag, maar verschillende tijden
  ? "Maak een reservekopie!"
  WAIT
CASE FDATE("B:Bedrijf.dbf") < FDATE("Bedrijf.dbf")
  * reservekopie is van eerdere datum dan huidige tabel
  Verschil= ;
    FDATE("Bedrijf.dbf") - FDATE("B:Bedrijf.dbf")
  ? Verschil, " dagen sinds laatste reservekopie."
  ? "Maak een reservekopie!"
  WAIT
ENDCASE
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FILE( ), FLUSH, FSIZE( ), FDATE( ), SET DIRECTORY, SET PATH

FUNCTION

Programma's

Definieert een door de gebruiker gedefinieerde functie in een programmabestand. Dit commando wordt hoofdzakelijk ondersteund vanwege de compatibiliteit met dBASE IV. In die versie bestonden belangrijke syntactische en functionele verschillen tussen door de gebruiker gedefinieerde functies en procedures (die worden gedefinieerd met het commando PROCEDURE). In dBASE voor Windows geldt dezelfde syntaxis voor procedures en door de gebruiker gedefinieerde functies en kunnen beide op dezelfde manier worden aangeroepen en toegepast. Vandaar dat het niet langer nodig is onderscheid te maken tussen procedures en functies. In dBASE voor Windows wordt u geadviseerd gebruik te maken van het commando PROCEDURE. Zie PROCEDURE voor meer informatie.

Overdraagbaarheid

In dBASE IV was het mogelijk dat een procedure en een door de gebruiker gedefinieerde functie met dezelfde naam naast elkaar bestonden omdat ze op een andere manier werden aangeroepen. Als in dBASE voor Windows een procedure en een door de gebruiker gedefinieerde functie met dezelfde naam bestaan, wordt alleen de procedure of functie gebruikt die als eerste is gedefinieerd. Zie PROCEDURE voor meer informatie over overdraagbaarheid.

FUNIQUE( )

Maakt een unieke bestandsnaam.

Syntaxis

FUNIQUE(<*Tuitdr*>)

<*Tuitdr*>

Een bestandsnaamschema dat mag bestaan uit jokertekens en alle geldige tekens voor bestandsnamen.

Beschrijving

Gebruik FUNIQUE( ) om een unieke bestandsnaam te maken met willekeurige getallen en opgegeven letters. U kunt met FUNIQUE( ) bijvoorbeeld tijdelijke bestanden maken zonder het risico te lopen bestaande bestanden te overschrijven.

Met het jokerteken ? kunt u aangeven hoe lang de bestandsnaam moet zijn en welke tekens in de bestandsnaam uit willekeurige getallen moeten bestaan.

FUNIQUE( ) genereert een nieuwe bestandsnaam door elk jokerteken te vervangen door een willekeurig cijfer. Vervolgens wordt in de huidige of opgegeven directory gekeken of er al een bestand met die naam bestaat. Als er geen gelijke bestandsnaam wordt aangetroffen, maakt FUNIQUE( ) het bestand en wordt de naam als resultaat gegeven. Als er wel een gelijke bestandsnaam wordt aangetroffen, probeert FUNIQUE( ) het opnieuw tot een unieke bestandsnaam is gevonden. Als geen enkele unieke combinatie van letters en willekeurige getallen uniek is, geeft FUNIQUE( ) een lege tekenreeks als resultaat.

Als u <*Tuitdr*> achterwege laat, maakt FUNIQUE( ) een bestandsnaam van acht tekens zonder extensie die volledig uit een willekeurig getal bestaat.

Voorbeeld

In het volgende voorbeeld wordt FUNIQUE( ) gebruikt om een unieke bestandsnaam te bepalen voor een tijdelijke tabel waarin totalen worden berekend. Na afloop wordt het tijdelijke bestand verwijderd:

```
TijdelTabel=FUNIQUE("Tijd????.dbf")
* Instructie geeft bestandsnaam als resultaat die bestaat
* uit Tijd gevolgd door 4 cijfers en .dbf, zoals Tijd7990.dbf

USE Orders EXCLUSIVE
INDEX ON Klantnr TAG Klantnr
TOTAL ON Klantnr TO &TijdelTabel
* TijdelTabel bevat nu totalen
* voor orders per klant

USE &TijdelTabel
BROWSE FIELDS Klantnr, BetBedrag, Totaal
USE
```

```
ERASE &TijdelTabel
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

FCREATE( ), FERROR( ), FILE( ), FOPEN( )

FV( )

Numerieke gegevens

Geeft een zwevende waarde als resultaat die de eindwaarde van een investering voorstelt.

Syntaxis

FV(<*betaling Nuitdr*>, <*rente Nuitdr*>, <*termijnen Nuitdr*>)

<*betaling Nuitdr*>

Het bedrag voor de termijnbetaling. Geef het bedrag op voor dezelfde periode als de rente en de termijnen. De betaling kan negatief of positief zijn.

<*rente Nuitdr*>

De rente per periode als een positief decimaal getal. Geef de rente op voor dezelfde periode als de betaling en de termijnen.

<*termijnen Nuitdr*>

Het aantal betalingen. Geef de termijn op voor dezelfde periode als de betaling en de rente.

Beschrijving

Met FV( ) kunt u de eindwaarde berekenen op basis van gelijke periodieke betalingen (stortingen) en een vast rentepercentage. FV( ) geeft een zwevende waarde als resultaat die het totaal van alle betalingen plus de samengestelde rente voorstelt.

Geef het rentepercentage op in de vorm van een breuk. Als de rente op jaarbasis bijvoorbeeld 9,5% is, is <*rente Nuitdr*> gelijk aan ,095 (9,5/100) voor jaarlijkse betalingen.

gebruik dezelfde periode voor <*betaling Nuitdr*>, <*rente Nuitdr*> en <*termijnen Nuitdr*>. Als de betalingen bijvoorbeeld maandelijks geschieden, moet u ook de rente per maand en het aantal maandelijkse termijnen opgeven. Een rente op jaarbasis van 9,5% wordt dan bijvoorbeeld ,095/12 per maand. Dat is 9,5/100 gedeeld door 12 maanden.

FV( ) wordt berekend aan de hand van de volgende formule:

$$(1 + \text{rente})^{\text{termijnen}} - 1$$

$$fv = betaling * \frac{(rente + 1)^{perioden} - 1}{rente}$$

In deze formule staat rente voor rentepercentage / 100.

De eindwaarde van een maandelijkse investering van 350 gulden gedurende vijf jaar met een rente van 9% per jaar berekent u als volgt:

```
? FV(350,.09/12,60)              && Geeft 26398,45 als resultaat
? 350*((1+.09/12)^60-1)/(.09/12) && Geeft 26398,45 als resultaat
```

Met andere woorden, als u de volgende vijf jaar 350 gulden per maand stort op een rekening die per jaar 9% rente oplevert, kunt u na afloop beschikken over 26398,45 gulden.

Met SET DECIMALS stelt u het aantal decimalen in.

Voorbeeld

In het volgende voorbeeld wordt FV( ) gebruikt om de eindwaarde te berekenen van een reeks maandelijkse betalingen over een aantal maanden en met een gegeven rentepercentage:

```
LOCAL f
f = NEW TWFORM()
f.Open()

CLASS TWFORM OF FORM
   this.Text = "Eindwaarde"
   this.Width =         50
   this.Left =         49
   this.Height =         15
   this.ColorNormal = "BG+/BG"

   DEFINE TEXT TXT1 OF THIS;
       PROPERTY;
         Top          3,;
         Border .F.,;
         Text "Maandelijkse inleg?",;
         Width         26,;
         Left          5,;
         Height          1,;
         ColorNormal "BG+/BG"

   DEFINE ENTRYFIELD INLEG OF THIS;
       PROPERTY;
         Top          3,;
         Border .T.,;
         Value          0,;
         Picture "9999",;
         Width          10,;
         Left         30,;
         Height          1

   DEFINE TEXT TXT2 OF THIS;
       PROPERTY;
```

```
      Top          5,;
      Border .F.,;
      Text "Verwachte rente?",;
      Width         26,;
      Left          5,;
      Height          1,;
      ColorNormal "BG+/BG"

DEFINE ENTRYFIELD RENTE OF THIS;
    PROPERTY;
      Top          5,;
      Border .T.,;
      Value          0,;
      Picture "99.99",;
      Width          10,;
      Left         30,;
      Height          1

DEFINE TEXT TXT3 OF THIS;
    PROPERTY;
      Top          7,;
      Border .F.,;
      Text "Hoeveel maanden?",;
      Width         34,;
      Left          5,;
      Height          1,;
      ColorNormal "BG+/BG"

DEFINE ENTRYFIELD MAANDEN OF THIS;
    PROPERTY;
      Top          7,;
      Border .T.,;
      Value          0,;
      Picture "999",;
      Width          10,;
      Left         30,;
      Height          1

DEFINE PUSHBUTTON RESULTS OF THIS;
    PROPERTY;
      Top         11,;
      Default .T.,;
      OnClick {;Resultaat="Eindwaarde: Fl. " +;
              LTRIM(STR(FV(Form.Inleg.Value,+;
              (Form.Rente.Value/100)/12,Form.Maanden.Value),13,2));
              ; Form.TW.Text=Resultaat},;
      Text "Eindwaarde berekenen",;
      Width         40,;
      Left          5,;
      Height          2,;
      ColorNormal "N/W"

DEFINE TEXT TW OF THIS;
    PROPERTY;
      Top          9,;
```

```
        Border .F.,;
        Text "Eindwaarde: ",;
        Width         33,;
        Left           5,;
        Height           1,;
        ColorNormal "BG+/BG"

ENDCLASS
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

PAYMENT( ), PV( ), SET DECIMALS

FWRITE( )

Toegang op laag niveau

Schrijft een tekenuitdrukking naar een opgegeven bestand en plaatst de bestandsaanwijzer direct na het laatste geschreven teken. Als de functie slaagt, wordt het aantal geschreven tekens als resultaat gegeven. Als de functie niet lukt, wordt 0 als resultaat gegeven en als er een fout optreedt wordt -1 als resultaat gegeven.

Syntaxis

FWRITE(<*bestands-handle Nuitdr*>, <*tekenreeks Tuitdr*> [, <*tekens Nuitdr*>])

<*bestands-handle Nuitdr*>

Het nummer van de bestands-handle van het bestand waar de opgeven tekens en het einde-regelteken naar toe moeten worden geschreven. Als u een bestand opent met FCREATE( ) of FOPEN( ), geven deze functies een bestands-handle-nummer als resultaat. Gebruik dit getal als <*bestands-handle Nuitdr*>. Als u een bestands-handle-nummer opgeeft dat niet eerder als resultaat is gegeven door FCREATE( ) of FOPEN( ), wordt een foutmelding getoond.

<*tekenreeks Tuitdr*>

De tekenuitdrukking om naar het opgegeven bestand te schrijven. Als u slechts een deel van <*tekenreeks Tuitdr*> naar het bestand wilt schrijven, moet u het argument <*tekens Nuitdr*> gebruiken.

<*tekens Nuitdr*>

Het aantal tekens van de opgegeven tekenuitdrukking <*tekenreeks Tuitdr*> dat naar het opgegeven bestand moet worden geschreven, te beginnen bij het eerste teken in <*tekenreeks Tuitdr*>. Het geldige bereik loopt van 0 tot en met 254.

Beschrijving

FWRITE( ) schrijft een tekenreeks naar een bestand. met uitzondering van een eigenschap zijn FWRITE( ) en FPUTS( ) onderling uitwisselbaar. FPUTS( ) schrijft na de tekenreeks altijd een einde-regelindicatie naar het bestand, terwijl FWRITE( ) dat niet doet. Zie FPUTS( ) voor meer informatie.

Voorbeeld

In het volgende voorbeeld wordt FWRITE( ) gebruikt binnen een SCAN-lus om de inhoud van het veld Naam in elk record in de tabel Dieren naar een tekstbestand te schrijven. Het programma gebruikt FGETS( ) binnen een DO WHILE .NOT. FEOF( )-lus om de inhoud van het tekstbestand weer te geven in het commandovenster.

```
nBestand="Dieren.TXT"
nHandle=FCREATE(nBestand,"RW")   && Tekstbestand maken
SET PATH TO C:\DBASEWIN\VOORBD   && Directory Voorbd;
                                    actief maken
USE DIEREN
* Gegevens invoeren
SCAN
  FWRITE(nHandle,Trim(Naam)+CHR(13)+CHR(10))
* Inhoud veld Naam plus regelterugloop/regeldoorvoer
* toevoegen aan tekstbestand.
ENDSCAN

* Uitvoer van gegevens
FSEEK(nHandle,0,0)               && Bestandsaanwijzer naar;
                                    begin van bestand
CLEAR                            && Resultatenpaneel van het;
                                    commandovenster legen
DO WHILE .NOT. FEOF(nHandle)
  ? FGETS(nHandle)               && Regel weergeven
ENDDO
FCLOSE(nHandle)                  && Dieren.TXT sluiten
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FCREATE( ), FEOF( ), FERROR( ), FOPEN( ), FPUTS( ), FREAD( ), FSEEK( )

GENERATE

Voegt willekeurige records toe aan de huidige tabel.

Syntaxis

GENERATE [<*Nuitdr*>]

<Nuitdr>

Het aantal records met willekeurige gegevens dat u wilt toevoegen aan de huidige tabel. Het argument <*Nuitdr*> moet liggen tussen 1 en 1.000.000.000 en mag in totaal niet meer dan 2.000.000.000 bytes (2.000MB) opleveren, de maximumgrootte van een dBASE-tabel. Als u voor <*Nuitdr*> een waarde opgeeft van 0 of kleiner dan 0, worden geen records gegenereerd. Als u geen waarde voor <*Nuitdr*> opgeeft, wordt u gevraagd een aantal op te geven.

Beschrijving

GENERATE vult een tabel met testgegevens, zodat u een programma grondig kunt testen en de fouten in het programma kunt opsporen. Als een tabel bestaande records bevat, blijven die staan en worden <*Nuitdr*> records aan de tabel toegevoegd.

Als u met GENERATE gegevens toevoegt aan een tabel die een memoveld bevat, wordt voor elk record wel een memoveld gemaakt, maar wordt het bijbehorende memobestand niet gevuld met gegevens.

Voorbeeld

In het volgende voorbeeld wordt GENERATE gebruikt om tien nieuwe records toe te voegen aan een tijdelijke tabel op basis van KLANTEN.DBF:

```
USE Klanten
COUNT       && Geeft aantal records als resultaat
COPY TO TempX
USE TempX
GENERATE 10
COUNT       && Geeft oorspronkelijke aantal +10 als resultaat
BROWSE      && Tien willekeurige records toegevoegd
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

APPEND, BROWSE, CHANGE, DEBUG, DISPLAY COVERAGE, EDIT, SET COVERAGE

GETCOLOR( )

Kleuren en fonts

Roept een dialoogvenster aan waarin u een eigen kleur kunt definiëren of een kleur kunt kiezen uit het kleurpalet. Geeft een tekenreeks als resultaat met de waarden voor rood, groen en blauw voor de gekozen kleur.

Syntaxis

GETCOLOR([<*titel Tuitdr*>])

<*titel Tuitdr*>

Een tekenreeks van maximaal 78 tekens die als titel van het dialoogvenster wordt gebruikt.

Beschrijving

Gebruik GETCOLOR( ) om een dialoogvenster te openen waarin u een kleur kunt kiezen uit een palet met vooringestelde kleuren of een eigen kleur kunt maken. In dit dialoogvenster kiest en maakt u kleuren op dezelfde wijze als in het dialoogvenster **Kleur** van het **Configuratiescherm** van Windows.

GETCOLOR( ) geeft een tekenreeks als resultaat met de opmaak "roodwaarde, groenwaarde, blauwwaarde", zoals in het volgende voorbeeld:

```
mRood = GETCOLOR()          && Kies volledig rode kleur
? mRood                     && Geeft "255,0,0" als resultaat
mBlauw = GETCOLOR()         && Kies lichtblauwe kleur
? mBlauw                    && Geeft "164,200,240" als resultaat
```

De tekenreeks die door GETCOLOR( ) als resultaat wordt gegeven, kunt u toepassen in het commando DEFINE COLOR.

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DEFINE COLOR

GETDIRECTORY( )

Stations- en bestandsfuncties

Toont een dialoogvenster waarin u een directory kunt instellen die in volgende instructies zal worden gebruikt.

Syntaxis

GETDIRECTORY([<*directory Tuitdr*>])

<directory Tuitdr>
De startdirectory die in het dialoogvenster verschijnt. Als *<directory Tuitdr>* wordt weggelaten, wordt de huidige directory gebruikt als startdirectory.

Beschrijving

Gebruik GETDIRECTORY( ) om een directorynaam als resultaat te geven. Die naam kan da worden gebruikt in volgende instructies met bijvoorbeeld FCREATE( ), SET DIRECTORY en SET PATH.

De reeks die door GETDIRECTORY( ) als resultaat wordt gegeven, eindigt niet op een backslash. Als u een volledig pad wilt maken door het resultaat van GETDIRECTORY( ) te combineren met een bestandsnaam, moet u in het pad een backslash opnemen. Dat kan op de volgende manieren:

```
* Backslash toevoegen aan resultaat
mTpad = GETDIRECTORY() + "\"
* Backslash toevoegen tijdens aaneenschakelen van bestandsnaam
mTBestnaam = "abc.txt"
mTHeleNaam = GETDIRECTORY() + "\" + mTBestnaam
```

Voorbeeld

Hier volgen drie voorbeelden met GETDIR( ):

```
Nwdir=GETDIR()
* directorydialoogvenster openen
* Nwdir is leeg als gebruiker Annuleren kiest
Nwdir=GETDIR("C:\ADMIN")
USE Bedrijf
HeleNaam=GETDIR()+"\"+SUBSTR(DBF(),3)
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

FCREATE( ), GETFILE( ), SET DIRECTORY, SET PATH

GETENV( )

Stations- en bestandsfuncties

Geeft de waarde van een DOS-omgevingsvariabele als resultaat.

Syntaxis

GETENV(<*Tuitdr*>)

<Tuitdr>
De naam van de DOS-omgevingsvariabele waarvan u de waarde wilt weten.

Beschrijving

Met GETENV( ) kunt u de huidige waarde van een DOS-omgevingsvariabele als resultaat geven.

DOS-omgevingsvariabelen komen overeen met dBASE-geheugenvariabelen. Deze variabelen worden meestal gemaakt met DOS-opdrachten als SET en PATH. De opdracht SET NAAM=BART maakt een omgevingsvariabele genaamd NAAM die de waarde BART heeft, en PATH=C:\DOS wijzigt de DOS-variabele PATH.

Als de door *<Tuitdr>* aangeduide omgevingsvariabele niet wordt aangetroffen, wordt een null-reeks als resultaat gegeven.

Voorbeeld

Hier volgende enkele voorbeelden met GETENV( ):

```
? GETENV("Comspec") && Plaats van command.com
? GETENV("Path")    && Het huidige pad
? GETENV("Prompt")  && De huidige prompt (bijv $p$g)
```

Zie ook

OS( ), SET PATH

GETEXPR( )

Uitdrukkingen en gegevenstypeconversie

Toont een dialoogvenster waarin u een uitdrukking kunt maken of wijzigen. De opgegeven uitdrukking wordt als resultaat gegeven.

Syntaxis

GETEXPR([*<uitdrukking Tuitdr>* [, *<titel Tuitdr>*[, *<gegevenstype Tuitdr>*]]])

<uitdrukking Tuitdr>

Een tekenuitdrukking om te wijzigen. Als u *<uitdrukking Tuitdr>* opgeeft, verschijnt een dialoogvenster met *<uitdrukking Tuitdr>* in het veld **Uitdrukking**. Als u geen *<uitdrukking Tuitdr>* opgeeft, blijft het veld **Uitdrukking** in het dialoogvenster leeg.

<titel Tuitdr>

Een tekenreeks die als titel voor het dialoogvenster wordt gebruikt. Als u geen *<titel Tuitdr>* opgeeft, wordt de standaardtitel van het dialoogvenster gebruikt. Als u een waarde wilt opgeven voor *<titel Tuitdr>*, moet u ook een waarde of lege reeks ("") opgeven voor *<uitdrukking Tuitdr>*.

<gegevenstype Tuitdr>

Een enkel teken dat het gegevenstype van het resultaat van de uitdrukking aangeeft. Het gegevenstype verschijnt in het dialoogvenster bij **Resultaattype**. Als u een waarde wilt opgeven voor *<gegevenstype Tuitdr>*, moet u ook een waarde of lege reeks ("") opgeven voor *<uitdrukking Tuitdr>* en *<titel Tuitdr>*.

Gebruik voor <*gegevenstype Tuitdr*> de volgende tekens:

- C voor tekengegevens
- D voor datumgegevens
- L voor logische gegevens
- N voor numerieke gegevens
- F voor zwevende gegevens
- X voor elke soort gegevens

Beschrijving

GETEXPR( ) toont het venster **Uitdrukking samenstellen**. met dit hulpmiddel kunt u geldige dBASE-uitdrukkingen maken die u kunt invoegen in de Editor, het commandovenster en in bepaalde velden in dialoogvensters. Een gebruiker kan bijvoorbeeld het venster **Uitdrukking samenstellen** gebruiken om een voorwaarde te maken die u vervolgens op een tabel toepast met een SET FILTER-instructie.

Met het venster **Uitdrukking samenstellen** kunt u alleen uitdrukkingen maken. Ze worden niet toegewezen. U kunt GETEXPR( ) toepassen in een programma of in het commandovenster, en u kunt een variabele toewijzen aan GETEXPR( ) zodat die variabele de waarde krijgt die door de functie als resultaat wordt gegeven.

U kunt het venster **Uitdrukking samenstellen** ook op de volgende manieren openen:

- Kies **Bewerken | Uitdrukking samenstellen**
- Druk op *Ctrl-E*

Voorbeeld

In het volgende voorbeeld wordt GETEXPR( ) gebruikt om naar het dialoogvenster **Uitdrukking bewerken** te springen en de regio op te geven waarnaar moet worden gezocht:

```
USE Klanten
tRegio       = SPACE( LEN(Klanten->Regio) )
tUitdrukking = "UPPER(Regio)"
tTitel       = " Maak een uitdrukking "
tType        = "L"
tSleutel     = ""
CLEAR
@ 1,1 SAY "Geef regio op: " ;
      GET tRegio            ;
      PICTURE "@!"
READ
IF .NOT. ISBLANK( tRegio )
  tUitdrukking = tUitdrukking + " = " + "'" + tRegio;
  + "'"
  tSleutel = GETEXPR(tUitdrukking,tTitel,tType)
  LOCATE FOR &tSleutel.
    IF FOUND()
      ? Naam, Plaats
    ELSE
```

```
      CLEAR
      ? "Helaas! Niet gevonden: " + tRegio
    ENDIF
ENDIF
USE
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

SET FILTER

GETFILE( )

Stations- en bestandsfuncties

Toont een dialoogvenster waarin de gebruiker een bestaande bestandsnaam kan opgeven of kiezen. De bestandsnaam wordt als resultaat gegeven. Als de gebruiker het dialoogvenster op een andere wijze verlaat dan door te dubbelklikken op een bestandsnaam of *OK* te kiezen, wordt een lege tekenreeks als resultaat gegeven.

Syntaxis

GETFILE([<*bestandsnaamfilter Tuitdr*>
  [, <*titel Tuitdr*>
  [, <*bestandstype Luitdr*>
  [, <*verander bestandstype Luitdr*>]]]])

<*bestandsnaamfilter Tuitdr*>

Een tekenreeks die een globale bestandsnaam met jokertekens (? en *) voorstelt. In het dialoogvenster worden alleen de bestandsnamen getoond die voldoen aan de opgegeven globale bestandsnaam. Zonder <*bestandsnaamfilter Tuitdr*> worden in het dialoogvenster alle bestandsnamen getoond.

<*titel Tuitdr*>

Een titel voor het dialoogvenster. <*titel Tuitdr*> mag maximaal uit 60 tekens bestaan. Zonder <*titel Tuitdr*> wordt de standaardtitel van het dialoogvenster getoond. Als u een waarde wilt opgeven voor <*titel Tuitdr*>, moet u ook een waarde of lege reeks ("") opgeven voor <*bestandsnaamfilter*>.

<*bestandstype Luitdr*>

Een logische waarde die bepaalt of in het dialoogvenster een lijst verschijnt van alle bestandstypen (.T.) of van tabellen in een database (.F.). De standaardinstelling is .T. Als u een waarde wilt opgeven voor <*bestandstype Luitdr*>, moet u ook een waarde of lege reeks ("") opgeven voor <*bestandsnaamfilter*> en <*titel Tuitdr*>.

<verander bestandstype Luitdr>
Een logische waarde die bepaalt of de gebruiker in het dialoogvenster kan schakelen tussen tabelbestanden en andere bestandstypen (.T.) of niet tussen bestandstypen kan schakelen (.F.). De standaardinstelling is .F. Als u een waarde wilt opgeven voor *<verander bestandstype Luitdr>*, moet u ook een waarde of lege reeks ("") opgeven voor *<bestandsnaamfilter>* en voor *<titel Tuitdr>*, en u moet een waarde opgeven voor *<bestandstype Luitdr>*.

Beschrijving

Gebruik GETFILE( ) om een bestandsnaam te kiezen in een dialoogvenster. De bestandsnaam wordt als resultaat gegeven en kan vervolgens worden toegepast in andere commando's en functie-aanroepen. U kunt GETFILE( ) bijvoorbeeld gebruiken om een tabelnaam als resultaat te geven zodat u de tabel kunt openen met USE, of om de naam van een tekstbestand als resultaat te geven om in een tekstobject te plaatsen. GETFILE( ) opent geen bestanden.

In het dialoogvenster dat verschijnt als u GETFILE( ) gebruikt, worden de namen van alle bestanden getoond, ook de geopende bestanden. Onafhankelijk van de instelling van SET FULLPATH wordt altijd het volledige pad als resultaat gegeven.

Standaard worden in het dialoogvenster dat verschijnt als u GETFILE( ) voor het eerst gebruikt, de bestandsnamen uit de huidige directory getoond. De volgende keer dat u GETFILE( ) gebruikt, wordt de directory getoond die u als laatste hebt ingesteld.

Voorbeeld

Enkele voorbeelden met GETFILE( ):

```
B1=GETFILE()        && Dialoogvenster wordt geopend
B2=GETFILE("*.prg") && Alleen programmabestanden worden getoond
B3=GETFILE("*.dbf","Tabel kiezen")
* Alleen tabellen worden getoond. Dialoogvenster krijgt
* titel "Tabel kiezen"
B4=GETFILE("*.dbf","Aleen tabellen",.f.)
* Gebruiker kan alleen tabellen kiezen

? "B1",B1
? "B2",B2
? "B3",B3
? "B4",B4
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

FILE( ), PUTFILE( ), SET FULLPATH

GETFONT( )

Kleuren en fonts

Toont een dialoogvenster waarin u een tekenfont kunt kiezen. Geeft een tekenreeks als resultaat die de volgende informatie bevat: fontnaam, puntgrootte, fontstijl (als u een andere stijl kiest dan **Standaard**) en fontfamilie.

Syntaxis

GETFONT([<*titel Tuitdr*>])

<titel Tuitdr>

Een tekenreeks van maximaal 78 tekens die als titel van het dialoogvenster wordt gebruikt.

Beschrijving

Met GETFONT( ) kunt u de waarden met de instellingen voor een opgegeven font in een tekenreeks plaatsen (zie de volgende voorbeelden). Als u een font wilt toevoegen aan de sectie [Fonts] in DBASEWIN.INI, maar niet de exacte naam of familie weet, kunt u GETFONT( ) gebruiken. De informatie die door GETFONT( ) als resultaat wordt gegeven, kunt u dan in DBASEWIN.INI invoegen.

```
 mNormaal = GETFONT()  && Arial, Standaard, 10-pt
? mNormal              && Geeft "Arial,10,Swiss" als resultaat
mVet = GETFONT()       && Kies Helvetica bold, 12-pt
? mVet                 && Geeft "Helvetica,12,B,Swiss" als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

?

GO

Velden en records

Verplaatst de recordaanwijzer naar de opgegeven positie in een tabel. In het geval van Paradox- en SQL-tabellen kunt u de recordaanwijzer naar een bepaald record verplaatsen met behulp van bladwijzers.

Syntaxis

GO[TO]
BOTTOM | TOP | <*bladwijzer*> | [RECORD] <*Nuitdr*>
[IN <*alias*>]

TO
Alleen beschikbaar vanwege de leesbaarheid; TO heeft geen gevolgen voor de werking van het commando.

BOTTOM | TOP | <*bladwijzer*> | RECORD <*Nuitdr*>
Geeft aan waar de recordaanwijzer naar toe moet worden verplaatst. De betekenis van de verschillende sleutelwoorden en opties wordt beschreven in de volgende tabel.

Optie	Beschrijving
BOTTOM	Als de opgegeven tabel geen hoofdindex heeft, wordt de recordaanwijzer naar het laatste record in de tabel verplaatst. Als de opgegeven tabel een hoofdindex heeft, wordt de recordaanwijzer naar het laatste record in de index verplaatst.
TOP	Als de opgegeven tabel geen hoofdindex heeft, wordt de recordaanwijzer naar het eerste record in de tabel verplaatst. Als de opgegeven tabel een hoofdindex heeft, wordt de recordaanwijzer naar het eerste record in de index verplaatst.
<*bladwijzer*>	Een markeringsteken voor een bepaalde rij voor tabellen die geen recordnummers ondersteunen (lijkt op een recordaanwijzer).
RECORD <*Nuitdr*>	Het recordnummer waar de recordaanwijzer naar toe moet worden verplaatst. GO <*Nuitdr*> komt overeen met het opgeven van een nummer in het commandovenster. Het sleutelwoord RECORD is alleen beschikbaar vanwege de leesbaarheid. Het heeft geen gevolgen voor de werking van het commando.

IN <*alias*>
Geeft een andere geopende tabel aan dan de huidige tabel. U kunt een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam opgeven. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving
GO positioneert de recordaanwijzer in de huidige of opgegeven tabel of in het huidige of opgegeven indexbestand. GO <*Nuitdr*> en GO RECORD <*Nuitdr*> verplaatsen de recordaanwijzer naar een bepaald record, onafhankelijk van een eventueel geopende hoofdindex of van de positie van het recordnummer in de indexvolgorde.

Voor tabellen die geen recordnummers ondersteunen (voor Paradox- en SQL-tabellen dus), geeft GO <*Nuitdr*> een fout als resultaat. In het geval van deze tabellen geeft u in plaats van een waarde voor de recordaanwijzer een <*bladwijzer*> op. Bladwijzers zijn een speciaal gegevenstype dat niet rechtstreeks kan worden weergegeven of afgedrukt. Een bladwijzer voor een bepaald record kunt bijvoorbeeld als resultaat geven met de functies RECNO( ) en BOOKMARK( ). U kunt een bladwijzer ook opslaan in een geheugenvariabele en bij het commando GO die variabele opgeven voor <*bladwijzer*>.

Als er geen index actief is, verwijzen TOP en BOTTOM naar het eerste en het laatste record in de tabel. Als er wel een index actief is voor de tabel, verwijzen TOP en BOTTOM naar het eerste en het laatste record in het indexbestand.

Als SET DELETED is ingeschakeld (ON) kunt u de recordaanwijzer verplaatsen naar een record dat is gemarkeerd voor verwijdering door het recordnummer op te geven. U kunt de recordaanwijzer met GOTO ook verplaatsen naar records die beperkt zijn met SET FILTER, hoewel u met EDIT geen toegang kunt krijgen tot dergelijke records.

Als er een relatie is ingesteld tussen verschillende tabellen en u verplaatst de recordaanwijzer in de hoofdtabel, dan wordt de recordaanwijzer in een subtabel naar een verwant record verplaatst. Als er geen verwant record is, wordt de recordaanwijzer in de subtabel aan het eind van het bestand geplaatst. Als u de recordaanwijzer verplaatst in een subtabel, heeft dat echter niet tot gevolg dat de recordaanwijzer in de hoofdtabel automatisch wordt verplaatst.

Voorbeeld

In het volgende voorbeeld wordt de recordaanwijzer in een geopende tabel verplaatst met GO:

```
SET TALK OFF
USE Klanten EXCLUSIVE
INDEX ON Klantnr TAG Klantnr
SEEK "1984"
? RECNO(), Klantnr, Naam, Postcode
recnum = RECNO()
GO TOP
? RECNO(), Klantnr, Naam, Postcode
GO BOTTOM
? RECNO(), Klantnr, Naam, Postcode
rec_blw = BOOKMARK()
GO recnum
? RECNO(), Klantnr, Naam, Postcode
GO rec_blw
? RECNO(), Klantnr, Naam, Postcode
CLOSE ALL
```

Zie ook

EOF( ), RECNO( ), SELECT, SET DELETED, SET FILTER, SET RELATION, SKIP

HELP

Programmeren in Windows

Activeert het Helpsysteem van dBASE.

Syntaxis

HELP
[<*helponderwerp*>]

<*helponderwerp*>

Het Helponderwerp dat u wilt tonen.

Beschrijving

Met het commando HELP kunt u informatie over dBASE opvragen.

De optie <*helponderwerp*> is een tekenreeks die bestaat uit een enkele letter of een reeks letters. dBASE zoekt naar het eerste onderwerp waarvan de naam begint met deze

tekenreeks en opent het dialoogvenster **Zoeken** waarin het betreffende onderwerp is geselecteerd.

Als u de toets *F1* geen andere functie hebt gegeven met ON KEY of SET FUNCTION, kunt u op *F1* drukken om het commando HELP uit te voeren.

Zie Hoofdstuk 14 in *Programmeren* voor meer informatie over het maken van een eigen Helpsysteem.

Voorbeeld

Het commando HELP in een programma of in het commandovenster geeft een beschrijving van commando's en bijzondere eigenschappen en een overzicht geven van Weergaven en hulpmiddelen, de Debugger en de taal. De volgende instructie geeft toegang tot het Helpvenster **Weergaven en hulpmiddelen**:

```
HELP Weergaven en hulpmiddelen
```

Helpinformatie over een bepaald commando krijgt u bijvoorbeeld met

```
HELP GETFONT()
```

Helpinformatie over een bepaalde eigenschappen krijgt u bijvoorbeeld met

```
HELP KNOPPENBALKEN
```

Zie ook

HelpFile, HelpID, SET HELP TO, SET TOPIC

HOME( ) — Stations- en bestandsfuncties

Geeft het pad naar de directory waar DBASEWIN.EXE staat, als resultaat.

Syntaxis

HOME( )

Beschrijving

Gebruik HOME( ) om te bepalen in welke directory de actieve kopie van DBASEWIN.EXE staat. Als u dBASE installeert, installeert het installatieprogramma DBASEWIN.EXE standaard in de directory \DBASEWIN\BIN. HOME( ) geeft deze directory als resultaat. HOME( ) geeft altijd het volledige pad als resultaat, onafhankelijk van de instelling van SET FULLPATH.

U kunt de hoofddirectory van dBASE aangeven met _dbwinhome.

Voorbeeld

HOME( ) geeft de dBASE-directory van waaruit DBASEWIN.EXE is gestart:

```
? HOME()      && C:\DBASEWIN\BIN\
* Dit is niet hetzelfde als _dbwinhome
? _dbwinhome  && \DBASEWIN\
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

_dbwinhome, CD, MKDIR, SET DIRECTORY, SET FULLPATH, SET PATH

HTOI( )

Uitdrukkingen en gegevenstypeconversie

Geeft de decimale notatie van een hexadecimaal getal als resultaat.

Syntaxis

HTOI(<*Tuitdr*>)

<Tuitdr>

Het hexadecimale getal waarvan u de decimale notatie als resultaat wilt geven.

Beschrijving

Met HTOI( ) kunt u een hexadecimaal getal omzetten in een decimaal getal van het type zwevend. HTOI( ) is de tegengestelde functie van ITOH( ), waarmee u numerieke en zwevende getallen kunt omzetten in een tekenreeks met de hexadecimale notatie van die getallen. U kunt beide functies gebruiken voor aanroepen naar de API (Application Programming Interface) van Windows waarvoor hexadecimale waarden verplicht zijn.

Voorbeeld

In de volgende voorbeelden wordt HTOI( ) gebruikt om de numerieke waarden te bepalen van opgegeven hexadecimale waarden:

```
? HTOI("FAFA")          && Geeft 64250,0000 als resultaat
? HTOI("40FA")          && Geeft 16634,0000 als resultaat
? ITOH(12345,8)         && Geeft 00003039 als resultaat
? HTOI("    3013")      && Geeft 12307,0000 als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ITOH( )

ID( ) — Gedeelde gegevens

Geeft de naam van de huidige gebruiker op een *local area network* (LAN, lokaal netwerk) of een ander multi-user-systeem als resultaat.

Syntaxis

ID( )

Beschrijving

ID( ) accepteert geen argumenten en geeft de naam van de huidige gebruiker als resultaat in de vorm van een tekenreeks. Als u ID( ) aanroept op een systeem met één gebruiker of als de gebruikersnaam niet is aangemeld op een multi-user-systeem, geeft de functie een lege tekenreeks als resultaat.

Voorbeeld

In het volgende voorbeeld wordt bijgehouden welke netwerkgebruiker als laatste een record heeft gewijzigd. Er wordt een netwerkdatabase bijgewerkt en de huidige gebruiker wordt met ID( ) in het veld GEBRUIKER geplaatst:

```
PROCEDURE WijzigenOK       && Netwerkdatabase wordt bijgewerkt en;
                           gebruikersnaam aangemeld
IF ID() <> ""
   REPLACE NAAM WITH tNaam, GEBRUIKER with ID()
ELSE
   CLEAR
   ? "Verbinding met netwerk verbroken. Gegevens niet opgeslagen."
   WAIT
ENDIF
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CONVERT, LKSYS( ), NETWORK( )

IF — Programma's

Verwerkt voorwaardelijk instructies door een of meer voorwaarden te evalueren. De instructies na de eerste voorwaarde waaraan is voldaan, worden uitgevoerd.

Syntaxis

IF *<voorwaarde Luitdr1>*
    <instructies>
[ELSEIF *<voorwaarde Luitdr2>*
    <instructies>
[ELSEIF *<voorwaarde Luitdr3>*
    <instructies>...]]
[ELSE
    <instructies>]
ENDIF

<voorwaarde Luitdr>

Een logische uitdrukking die bepaalt of de reeks instructies tussen IF en de volgende ELSE, ELSEIF, IF of ENDIF worden uitgevoerd. Als aan de voorwaarde is voldaan, worden de instructies uitgevoerd. Als niet aan de voorwaarde is voldaan, wordt de besturing doorgegeven aan de volgende ELSE, ELSEIF of ENDIF.

<instructies>

Eén of meer programmaregels met een willekeurige combinatie van commando's, functies en preprocessor-instructies.

ELSEIF *<voorwaarde Luitdr>* *<instructies>*

Als niet de voorwaarde bij de vorige IF of ELSEIF is voldaan, wordt de besturing doorgegeven aan deze ELSEIF *<voorwaarde Luitdr>*. Net als bij IF geldt dat als aan de voorwaarde is voldaan, alleen de instructies tussen deze ELSEIF en de volgende ELSEIF, ELSE of ENDIF worden uitgevoerd. Als niet aan de voorwaarde is voldaan, wordt de besturing doorgegeven aan de volgende ELSE, ELSEIF of ENDIF.

U mag deze optie opgeven als ELSEIF of ELSE IF. De puntjes (...) in de syntaxis geven aan dat u meer dan één ELSEIF-instructie mag gebruiken.

ELSE *<instructies>*

Geeft aan welke instructies moeten worden uitgevoerd als niet aan alle voorgaande voorwaarden is voldaan.

ENDIF

Een verplicht commando dat het eind aangeeft van de IF-structuur.

Beschrijving

Gebruik IF...ELSEIF...ENDIF om een of meer voorwaarden te evalueren en alleen de instructies uit te voeren na de eerste voorwaarde waaraan is voldaan. De instructies tussen de eerste voorwaarde waaraan is voldaan en de volgende ELSEIF, ELSE of ENDIF worden uitgevoerd. Vervolgens wordt de rest van de IF-structuur overgeslagen en wordt de besturing doorgegeven aan de regel na ENDIF. Als aan geen enkele voorwaarde is voldaan en de structuur een regel met ELSE bevat, worden de instructies tussen ELSE en ENDIF uitgevoerd.

Gebruik IF...ENDIF om op één voorwaarde te testen en IF...ELSEIF...ENDIF om op twee of meer voorwaarden te testen. As u meer dan drie voorwaarden wilt testen, kunt u overwegen DO CASE te gebruiken in plaats van IF. Vergelijk het voorbeeld in deze sectie met het voorbeeld bij DO CASE.

U kunt meerdere IF-instructies nesten om verschillende voorwaarden te testen, maar in dat geval kunt u beter de optie ELSEIF gebruiken. Als u ELSEIF gebruikt, hoeft u niet bij te houden welke ELSE bij welke IF hoort en hoeveel keer u ENDIF moet gebruiken.

Elke reeks commando's kan uit vele commando's bestaan. Als het aantal commando's in een reeks de code moeilijk leesbaar maakt, kunt u ze opnemen in een procedure en die procedure aanroepen vanuit de IF-instructie.

Voorbeeld

In het volgende voorbeeld met geneste IF-constructies wordt de grootte van een eerder gedefinieerde geheugenvariabele bepaald. Vervolgens wordt een melding daaromtrent weergegeven. Vergelijk deze code met de eenvoudiger constructie met ELSEIF in het tweede voorbeeld:

```
nG_waarde=523
IF nG_waarde > 1000
   ? "Waarde is groter dan 1000."
ELSE
   IF nG_waarde > 100
      ? "Waarde is groter dan 100."
   ELSE
      IF nG_waarde > 10
         ? "Waarde is groter dan 10."
      ELSE
         IF nG_waarde > 1
            ? "Waarde is groter dan 1."
         ELSE
            ? "Waarde is 1 of kleiner dan 1."
         ENDIF
      ENDIF
   ENDIF
ENDIF
```

In het volgende voorbeeld van een IF-constructie wordt ELSEIF gebruikt om de grootte van een eerder gedefinieerde geheugenvariabele te bepalen en een melding weer te geven:

```
IF nG_waarde > 1000
   ? "Waarde is groter dan 1000."
ELSEIF nG_waarde > 100
   ? "Waarde is groter dan 100."
ELSEIF nG_waarde > 10
   ? "Waarde is groter dan 10."
ELSEIF nG_waarde > 1
   ? "Waarde is groter dan 1."
ELSE
   ? "Waarde is 1 of kleiner dan 1."
ENDIF
```

Overdraagbaarheid

ELSEIF wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DO CASE, DO...UNTIL, DO WHILE, FOR...NEXT, IIF( ), SCAN

IIF( )

Programma's

Geeft een van twee waarden als resultaat afhankelijk van de uitkomst van een opgegeven logische uitdrukking.

Syntaxis

IIF(<*Luitdr*>, <*uitdr1*>, <*uitdr2*>)

<*Luitdr*>

De logische uitdrukking die bepaalt of <*uitdr1*> of <*uitdr2*> als resultaat wordt gegeven.

<*uitdr1*>

De teken-, datum-, logische, numerieke of zwevende uitdrukking die als resultaat wordt gegeven als <*Luitdr*> waar (.T.) is.

<*uitdr2*>

De teken-, datum-, logische, numerieke of zwevende uitdrukking die als resultaat wordt gegeven als <*Luitdr*> onwaar (.F.) is. Het gegevenstype van <*uitdr2*> hoeft niet gelijk te zijn aan dat van <*uitdr1*>.

Beschrijving

IIF( ) is een afkorting van "immediate IF", hetgeen onmiddellijke IF (OF) betekent. Deze constructie is een kort alternatief voor IF...ELSE...ENDIF. Gebruik IIF( ) op plaatsen waar uitdrukkingen wel, maar programmeerconstructies niet zijn toegestaan, zoals in rapporten en etiketten. Tevens kunt u met dit commando een waarde beperken tot twee mogelijkheden.

Voorbeeld

In het volgende voorbeeld wordt IIF( ) gebruikt om een van twee mogelijke resultaten door te geven aan de variabele of de instructie waarin IIF( ) wordt gebruikt. Als de IDAPI-taalaansturing op een Nederlandse aansturing is ingesteld, wordt de datumnotatie op ITALIAN (DD-MM-JJ) ingesteld, zo niet, dan wordt de datumnotatie op AMERICAN (MM/DD/JJ) ingesteld:

```
x = IIF(SUBSTR(LDRIVER(),6,2)="NL", "ITALIAN", ;
   "AMERICAN")
SET DATE &x
? DATE()
```

In het volgende voorbeeld wordt IIF( ) gebruikt om een tabel te indexeren en wel zo dat een opgegeven regio boven in de geordende database verschijnt in plaats van op zijn gebruikelijke positie op basis van de teken- of numerieke volgorde.

```
USE Bedrijf EXCLUSIVE
INDEX ON IIF(Regio="NW","A","Z") + REGIO ;
   TAG Regio1
LIST FIELDS Bedrijf, Plaats, Regio OFF NEXT 30
CLOSE ALL
```

Zie ook

IF

IMPORT

Tabellen

Maakt een dBASE-tabel uit gegevens in bestanden met een andere indeling.

Syntaxis

IMPORT FROM *<bestandsnaam>* | ?
[[TYPE] WB1 | WK1]
[HEADING]

<bestandsnaam> | ?

De naam van het bestand dat u wilt importeren. IMPORT ? toont een dialoogvenster waarin u het bestand kunt kiezen. Als u een bestand zonder pad opgeeft, zoekt dBASE voor Windows het bestand in de huidige directory en vervolgens in het pad dat is ingesteld met SET PATH.

[TYPE] WB1 | WK1

Geeft de indeling aan van de gegevens die u wilt importeren. Het sleutelwoord TYPE is er alleen voor de leesbaarheid; het heeft geen gevolgen voor de werking van het commando. De volgende tabel geeft een overzicht van de ondersteunde bestandsindelingen:

Type	Beschrijving
WB1	Een werkboekbestand uit Quattro Pro voor Windows. De rijen in het werkblad worden omgezet in records en de kolommen in velden. Als u geen extensie opgeeft, gebruikt dBASE voor Windows .WB1.
WK1	Een werkboekbestand uit Lotus 1-2-3. De rijen in het werkblad worden omgezet in records en de kolommen in velden. Als u geen extensie opgeeft, gebruikt dBASE voor Windows .WK1.

[HEADING]

Geeft aan dat de koppen boven de kolommen in het werkblad gebruikt moeten worden voor de veldnamen in de tabel die met IMPORT wordt gemaakt. De veldnamen worden ingekort tot tien tekens. Spaties worden omgezet in onderstrepingstekens en andere tekens worden indien nodig omgezet in geldige tekens voor dBASE-veldnamen.

Beschrijving

Met het commando IMPORT kunt u gegevens importeren uit bestanden met een andere dan de dBASE-indeling. Als het bestand een andere dan de standaardextensie heeft, moet u de extensie opgeven.

IMPORT maakt een tabel op hetzelfde station en in dezelfde directory als het oorspronkelijke bestand. De tabel wordt geopend in het actieve werkgebied. Met DISPLAY STATUS en DISPLAY STRUCTURE kunt u informatie over het geïmporteerde bestand weergeven.

Voorbeeld

In het volgende voorbeeld wordt IMPORT FROM gebruikt om een nieuw .DBF-bestand te maken aan de hand van een WB1-bestand (van Quattro Pro):

```
SET DBTYPE TO DBASE
IMPORT FROM C:\QPW\VOORBD\LENING.WB1 TYPE WB1
CD C:\QPW\VOORBD
USE LENING
BROWSE
CLOSE ALL
```

Zie ook

APPEND, DISPLAY STATUS, DISPLAY STRUCTURE, USE

INDEX

Tabelindeling

Maakt een index voor de huidige tabel. In het geval van Paradox- en SQL-tabellen kunt u indexlabels definiëren, maar geeft u geen naam op voor een indexbestand.

Syntaxis

INDEX ON *<sleuteluitdrukking>*
  TO *<.ndx-bestandsnaam>* | ? | *<.ndx-bestandsnaamfilter>*
  [UNIQUE]
    of
INDEX ON *<sleuteluitdrukking>* | *<veldenlijst>*
  TAG *<labelnaam>*
    [OF *<.mdx-bestandsnaam>* | ? | *<.mdx-bestandsnaamfilter>*]
    [FOR *<voorwaarde>*]
  [DESCENDING]
  [UNIQUE]

Met een speciale vorm van dit commando, INDEX ON *<veldenlijst>* PRIMARY, kunt u een primaire index voor een Paradox-tabel maken.

<sleuteluitdrukking> | <veldenlijst>

Voor dBASE-tabellen geeft *<sleuteluitdrukking>* de naam aan van een teken-, numeriek, zwevend of datumveld, of een dBASE-uitdrukking van maximaal 220 tekens inclusief

operatoren en functies waarmee veldwaarden worden gemanipuleerd, of een combinatie van veldnamen en uitdrukkingen. De maximumlengte van de sleutel, het resultaat van de geëvalueerde *<sleuteluitdrukking>*, is 100 tekens. Als voor *<sleuteluitdrukking>* meer dan een veld wordt gebruikt, moeten alle elementen van de uitdrukking het type teken als resultaat geven. U kunt meerdere velden en uitdrukkingen koppelen met de aaneenschakelingsoperatoren voor tekenreeksen (+ of -).

In het geval van Paradox- en SQL-tabellen mogen indexen geen uitdrukkingen bevatten. U kunt echter wel indexen maken die uit meerdere velden bestaan. In dat geval geeft u de indexsleutel op als een *<veldenlijst>*. De namen in de veldenlijst worden van elkaar gescheiden door komma's.

TAG *<labelnaam>*

Geeft de naam aan van de indexlabel die is toegevoegd aan een .MDX-bestand. Als u geen .MDX-bestand opgeeft, worden de indexlabels toegevoegd aan het produktie-MDX-bestand.

OF *<.mdx-bestandsnaam2>* | ? | *<.mdx-bestandsnaamfilter>*

Geeft de naam aan van het .MDX-bestand waaraan de nieuwe indexlabels moeten worden toegevoegd. OF ? en OF *<bestandsnaamfilter>* tonen een dialoogvenster waarin u een bestaand .MDX-bestand kunt kiezen. Als u een bestand opgeeft dat niet bestaat, wordt het gemaakt en wordt de naam van de indexlabel daar aan toegevoegd. Standaard krijgt het bestand de extensie .MDX en wordt het opgeslagen in de huidige directory.

TO *<.ndx-bestandsnaam1>* | ? | *<.ndx-bestandsnaamfilter>*

Geeft de naam aan van een .NDX-bestand. Standaard wordt de extensie .NDX gebruikt en wordt het bestand opgeslagen in de huidige directory. De opties ? en | *<.ndx-bestandsnaamfilter>* tonen een dialoogvenster waarin u de naam opgeeft van het doelbestand en van de directory waar het moet worden opgeslagen.

FOR *<voorwaarde>*

Beperkt de records die in de index worden opgenomen tot de records die voldoen aan de opgegeven *<voorwaarde>*. Zonder de opties FOR *<voorwaarde>* of UNIQUE worden alle records in de index opgenomen.

DESCENDING

Maakt een aflopende index (Z tot A, 9 tot 1, recente datums tot oudere datums). Zonder DESCENDING wordt een oplopende index gemaakt.

UNIQUE

Voorkomt dat in de index records met dezelfde waarde voor *<sleutel Tuitdr>* worden opgenomen. Alleen het eerste record met de waarde wordt in de index opgenomen. Zonder de opties FOR *<voorwaarde>* of UNIQUE worden alle records in de index opgenomen. In het geval van SQL-tabellen wordt een unieke index gemaakt waarin dubbele indexsleutels in de tabel worden voorkomen.

Beschrijving

Met INDEX kunt u gegevens organiseren zodat deze snel kunnen worden opgehaald en weergegeven. INDEX is niet van invloed op de feitelijke volgorde van de records in de tabel, maar maakt een indexbestand waarin de records zijn gerangschikt op basis van de waarde (numeriek, zwevend, teken of datum) van een sleuteluitdrukking. Net als de index in een boek met alfabetische gerangschikte ingangen en paginanummers, bevat een indexbestand gerangschikte indexuitdrukkingen met de bijbehorende recordnummers. Als de tabel wordt gebruikt met een hoofdindex, wordt de inhoud van de tabel weergegeven in de volgorde die wordt bepaald door de index.

Na voltooiing van de indexbewerking wordt het nieuwe indexbestand de hoofdindex en wordt de recordaanwijzer bij het eerste record in de geïndexeerde tabel geplaatst. Alle overige indexen worden gesloten, behalve de indexen waarvan de labelnamen voorkomen in het produktie-MDX-bestand met dezelfde naam als de tabel (mits dat bestaat).

Gewoonlijk zijn records in een index oplopend gerangschikt, met de laagste sleutelwaarde aan het begin van de index. Tekensleutels worden op basis van DOS-codetabel 437 (U.S.) gerangschikt op volgorde van de ASCII-waarden (van A tot Z en van dan van a tot z). Zie ook de volgende opmerking. Numerieke en zwevende sleutels worden gerangschikt van de laagste waarden naar de hoogste waarde en datumsleutels van de vroegste datums naar de laatste datum. Neem de functie UPPER( ) op in de sleuteluitdrukking om alle kleine letters om te zetten naar hoofdletters zodat tekenvelden geheel alfabetisch worden gerangschikt.

Opmerking De meeste andere tekensets leveren een andere sorteervolgorde voor tekens dan de tekenset die wordt bepaald door DOS-codetabel 437. De verwerking van speciale tekens (tekens met accenten) is tevens afhankelijk van de ingestelde taalaansturing. Zie Appendix C in *Programmeren* voor meer informatie over taalaansturingen en een volledige beschrijving van de sorteervolgorde in de Nederlandse taalaansturing.

U kunt de volgorde van een index omdraaien, zodat de records aflopend worden gerangschikt, door het sleutelwoord DESCENDING te gebruiken. (U kunt DESCENDING alleen gebruiken als u .MDX-labels maakt.)

Als in een sleuteluitdrukking een functie wordt gebruikt, moet u onthouden dat de index wordt gerangschikt aan de hand van de uitvoer van die functie. Als u dus FIND of SEEK gebruikt, of op een andere maner toegang zoekt tot de sleutelwaarde van een record, moet u de volledige sleuteluitdrukking opnemen. De instructie INDEX ON SOUNDEX(Naam) TO Namen maakt bijvoorbeeld een index die is geordend op de waarden die door SOUNDEX( ) als resultaat worden gegeven. Als u probeert gegevens op sleutelwaarde te zoeken, moet u de volledige uitdrukking opnemen, zoals SEEK SOUNDEX("Jansen") en niet SEEK "Jansen". Gebruik geen functies als CHR( ), LTRIM( ), RTRIM( ), TRIM( ) en IIF( ) die de veldlengte in de sleuteluitdrukking variëren.

In sommige tabellen kunnen meerdere records dezelfde <*sleutelwaarde*> hebben. Met de optie UNIQUE neemt u alleen het eerste van deze records op in de index. INDEX met de optie UNIQUE heeft dezelfde gevolgen als het indexeren van een tabel met SET UNIQUE ON.

FOR *<voorwaarde>* beperkt de records waarvan de waarden van de sleuteluitdrukking in de index worden opgenomen tot die records die voldoen aan de opgegeven voorwaarde. Als u bijvoorbeeld INDEX ON Achternaam + Voornaam TO Loonkosten FOR Salaris > 58000 gebruikt, worden alleen records in de index opgenomen voor werknemers met een salaris dat hoger is dan F 58000. In de FOR-voorwaarde kunnen geen rekenvelden worden opgenomen.

Als de tabel eenmaal is geïndexeerd, kunt u gegevens ophalen met LOOKUP( ), SEEK, SEEK( ) en FIND. De structuur van een indexbestand stelt deze commando's in staat snel waarden van de sleuteluitdrukking op te zoeken.

Als u APPEND, BROWSE, CHANGE, EDIT, INSERT, PACK, REPLACE, @...GET of UPDATE uitvoert, worden alle geopende indexbestanden automatisch bijgewerkt. Indexbestanden die gesloten zijn op het moment dat er wijzigingen worden aangebracht in de tabel, kunnen worden geopend en vervolgens bijgewerkt met REINDEX.

.MDX-bestanden maken het bijwerken van indexen eenvoudiger, omdat alle indexen worden bijgewerkt waarvan de labelnamen voorkomen in de .MDX-bestanden die zijn opgegeven met USE...ORDER of SET ORDER. Als een produktie-MDX-bestand bestaat, wordt dat automatisch geopend als u de bijbehorende tabel gebruikt.

INDEX...TAG maakt een index en voegt de labelnaam toe aan het .MDX-bestand. Als u geen OF *<bestandsnaam>* opneemt, voegt INDEX...TAG de labelnaam toe aan het produktie-MDX-bestand. Als het produktie-MDX-bestand of het opgegeven bestand nog niet bestaat, wordt dat gemaakt.

INDEX...TAG sluit alle geopende indexen behalve die met labelnamen die in het produktie-MDX-bestand of in het opgegeven .MDX-bestand staan. Als indexen met labelnamen in het opgegeven .MDX-bestand nog niet geopend zijn, worden die geopend.

INDEX komt overeen met SORT, een ander commando voor het ordenen van een tabel. In tegenstelling tot INDEX maakt SORT echter een nieuwe fysieke rangschikking van de tabelrecords en dat kan veel tijd kosten in het geval van grote bestanden. Om de gesorteerde volgorde te handhaven moeten nieuwe records met INSERT op de juiste plaats worden gezet of moet de volledige tabel opnieuw worden gesorteerd. Verder biedt SORT geen ondersteuning voor SEEK, SEEK( ) en FIND, en dat maakt het zoeken naar gegevens in een gesorteerde tabel trager.

Voorbeeld

In het volgende voorbeeld wordt INDEX gebruikt om indexlabels te maken voor de huidige tabel:

```
USE Bedrijf EXCLUSIVE
INDEX ON Bedrijf TAG Bedrijf
* Index op Bedrijf maken
BROWSE TITLE "Geindexeerd op Bedrijf"
INDEX ON Regio+Plaats ;
   TAG RegPlaats DESCENDING
* Gecombineerde index op Regio en Plaats
BROWSE TITLE "Aflopend geindexeerd op Regio & Plaats"
INDEX ON Plaats;
```

```
   TAG BelgNoord ;
   FOR Regio = "BN" && alleen plaatsen in regio BN
BROWSE TITLE "Geïndexeerd op Plaats, alleen België-Noord"
```

Overdraagbaarheid

De optie voor bestandsnaamschema's is niet beschikbaar in dBASE IV. De opties TAG, FOR en .MDX zijn niet beschikbaar in dBASE III PLUS.

Zie ook

FIND, KEY( ), LOOKUP( ), ORDER( ), REINDEX, SEEK, SEEK( ), SET INDEX, SET ORDER, SET UNIQUE, SORT, TAG( ), USE

INKEY( )

Toetsenbord- en muisacties

Geeft een decimale waarde als resultaat die de eerste toets of toetscombinatie in de typbuffer voorstelt, en verwijdert die toetsaanslag uit de buffer. Deze functie kan ook worden gebruikt om op een toetsaanslag te wachten en de waarde van de toetsaanslag als resultaat te geven.

Syntaxis

INKEY([<*seconden Nuitdr*>] [, <*muis Tuitdr*>])

<seconden Nuitdr>

Het aantal seconden dat op een toetsaanslag moet worden gewacht. Als <*Nuitdr*> gelijk is aan 0, wordt onbeperkt op een toetsaanslag gewacht.

<muis Tuitdr>

Bepaalt of ook een waarde als resultaat wordt gegeven als met de muis wordt geklikt. Als voor <*Tuitdr*> M of m is opgegeven, geeft INKEY( ) -100 als resultaat. Als voor <*Tuitdr*> iets anders is opgegeven, worden muisklikken genegeerd en wacht de functie op een toetsaanslag.

Beschrijving

Als het programma bezig is met het verwerken van andere gegevens, worden toetsaanslagen van de gebruiker opgeslagen in de typbuffer. Met INKEY( ) kunt u die toetsaanslagen opvragen en uit de typbuffer verwijderen. Als de typbuffer leeg is, geeft INKEY( ) 0 als resultaat.

Als u bijvoorbeeld op *C* en vervolgens op *Alt-P* drukt, worden in de typbuffer de waarden 67 en -420 opgeslagen. INKEY( ) geeft 67 als resultaat en een tweede INKEY( ) geeft -420 als resultaat. Zie Appendix D voor een volledige lijst van alle mogelijke waarden die door INKEY( ) als resultaat worden gegeven. Zie Appendix E voor een lijst van decimale waarden voor toetsen.

Met NEXTKEY( ) kunt u een waarde uitlezen op een andere positie in de typbuffer, of een waarde uit de typbuffer ophalen zonder die waarde uit de typbuffer te verwijderen.

Voorbeeld

In het volgende voorbeeld wordt een lus uitgevoerd die doorlopend het getypte teken en de resultaatwaarde van INKEY( ) weergeeft. De lus wordt beëindigd als op *Esc* (ASCII 27) wordt gedrukt:

```
CLEAR
SET ESCAPE OFF
* ESCAPE ON onderbreekt het programma en zorgt dat
* de toets Esc niet wordt onderschept door INKEY()
? "Druk op Esc om te stoppen"
t=0
DO WHILE t <> 27
  t=INKEY()
  ? t
  IF t>0
    ?? CHR(t)  && Toets en ascii-teken tonen
  ENDIF
ENDDO
SET ESCAPE ON
```

In het volgende voorbeeld wordt een melding getoond en wordt maximaal 10 seconden gewacht op een toetsaanslag of een muisklik:

```
? "Deze melding blijft maximaal 10 seconden staan"
Pauze=inkey(10,"m")
IF Pauze=0
  ? "U hebt 10 seconden gewacht"
ENDIF
```

Overdraagbaarheid

De optie *<seconden Nuitdr>* wordt niet ondersteund in dBASE III PLUS. De optie *<muis Tuitdr>* wordt niet ondersteund in dBASE IV en dBASE III PLUS. In dBASE voor Windows worden voor sommige toetsen andere waarden als resultaat gegeven dan in eerdere versies van dBASE. Zie Appendix D voor een volledige lijst van de waarden die als resultaat worden gegeven.

Zie ook

CLEAR TYPEAHEAD, KEYBOARD, LASTKEY( ), NEXTKEY( ), ON KEY, READKEY( ), SET TYPEAHEAD

INPUT

Invoer/uitvoer

Accepteert een door de gebruiker opgegeven uitdrukking en slaat die op in een geheugenvariabele. INPUT kan teken-, numerieke, zwevende, datum- en logische gegevens accepteren. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE met de

klassen Text en EntryField voor het weergeven en accepteren van informatie met een formulier.

Zie Help voor meer informatie over de syntaxis van INPUT. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het werken met formulieren in dBASE voor Windows.

INSERT

Velden en records

Voegt in de huidige tabel een nieuw record in op de plaats van het huidige record.

Syntaxis

INSERT
[BLANK]
[BEFORE]
[NOWAIT]

BLANK

Voegt een leeg record in, maar toont geen venster waarin de veldwaarden in het nieuwe record kunnen worden gewijzigd.

BEFORE

Als er geen hoofdindex in gebruik is, wordt het nieuwe record ingevoegd voor in plaats van na het huidige record.

NOWAIT

Roept het formulier aan om een nieuw record te bewerken, maar verplaatst de focus niet naar het formulier. In een programma wordt de besturing doorgegeven aan de instructie na het commando INSERT NOWAIT.

Beschrijving

Gebruik INSERT om een enkel record in te voegen in een bestaande tabel. In het geval van dBASE-tabellen die zijn gekoppeld met het commando SET RELATION, bepalen de opties CONSTRAIN en INTEGRITY tevens de handelingen die nodig zijn om nieuwe records toe te voegen aan de sub- en hoofdtabellen. Zie het commando SET RELATION voor meer informatie.

Als een record wordt toegevoegd aan een geïndexeerde tabel, wordt het nieuwe record toegevoegd aan het eind van de tabel (met een recordnummer dat een hoger is dan het oorspronkelijk hoogste recordnummer) en wordt het record tevens ingevoegd in alle geopende indexen. Indexbestanden die zijn gesloten op het moment dat het nieuwe record wordt toegevoegd kunnen worden bijgewerkt met REINDEX.

Als u een record invoegt aan het eind van een bestand, heeft INSERT (maar niet INSERT BLANK) dezelfde gevolgen als APPEND.

Als SET CARRY is uitgeschakeld (OFF, de standaardinstelling), worden lege records ingevoegd. Als SET CARRY is ingeschakeld (ON), wordt elk nieuwe record gevuld met de inhoud van het voorgaande record.

INSERT BLANK voegt een record toe aan de huidige tabel en verplaatst de recordaanwijzer naar het nieuwe record, net als INSERT, maar er verschijnt geen venster waarin waarden kunnen worden ingevoerd in het lege record.

In een tabel zonder een hoofdindex voegt het commando INSERT zonder de optie BEFORE een leeg record in direct na het huidige record. Tevens worden alle geopende indexen bijgewerkt. In een tabel met een hoofdindex, voegt INSERT het record toe aan het eind van de tabel en worden alle geopende indexen bijgewerkt als u het nieuwe record opslaat.

Voorbeeld

In het volgende voorbeeld wordt INSERT BLANK gebruikt om een leeg record in te voegen na het huidige record:

```
SET TALK OFF
USE Afnemers EXCLUSIVE
SKIP 3
? RECNO(), Bedrijf      && Let op recordnummer en bedrijf
SKIP-1                  && Recordaanwijzer 1 terug
INSERT BLANK
SKIP                    && Recordaanwijzer 1 verder
? RECNO(), Bedrijf         && Nummer oorspronkelijke record;
                           is nu 1 hoger
CLOSE ALL
```

In het volgende voorbeeld wordt INSERT BEFORE gebruikt om een leeg record in te voegen voor het huidige record. Er verschijnt een venster waarin de gebruiker waarden kan invoeren:

```
USE Afnemers EXCLUSIVE
SKIP 4
? RECNO(), Bedrijf
INSERT BEFORE           && Invoervenster verschijnt
? RECNO(), Bedrijf      && Toegevoegde record op;
                           positie oude record
SKIP                    && Recordaanwijzer 1 verder
? RECNO(), Bedrijf      && Oude record
CLEAR
CLOSE ALL
```

Zie ook

APPEND, INSERT AUTOMEM, REINDEX, REPLACE, SET CARRY, SET FIELDS, SET RELATION, STORE, USE

INSERT AUTOMEM

Voegt een nieuw record toe aan de huidige tabel en gebruikt daarbij waarden die zijn opgeslagen in automatische geheugenvariabelen.

Syntaxis

INSERT AUTOMEM
[BEFORE]

BEFORE

Voegt het nieuwe record in voor het huidige record.

Beschrijving

INSERT AUTOMEM voegt aan een tabel een nieuw record toe met waarden die zijn opgeslagen in automatische geheugenvariabelen. Als er geen hoofdindex bestaat, voegt INSERT AUTOMEM zonder de optie BEFORE het record in onmiddellijk na het huidige record. Als er een hoofdindex bestaat, wordt het record ingevoegd aan het eind van de tabel en worden alle geopende indexen bijgewerkt.

Bij APPEND AUTOMEM wordt het gebruik van automatische geheugenvariabelen voor het toevoegen van gegevens aan een tabel uitgebreid besproken.

Het enige verschil tussen INSERT AUTOMEM en APPEND AUTOMEM is dat INSERT AUTOMEM het record in een niet-geïndexeerde tabel toevoegt onmiddellijk na (of voor) het huidige record, terwijl APPEND AUTOMEM het record in dat geval aan het eind van de tabel invoegt. INSERT AUTOMEM wijst aan het record een recordnummer toe dat 1 hoger (of lager) is dan het nummer van het huidige record en wijzigt tevens de recordnummers van alle volgende records. APPEND AUTOMEM wijst aan het nieuwe record een nummer toe dat 1 hoger is dan het nummer van het laatste record in de tabel.

Als er een hoofdindex is geopend, doen APPEND AUTOMEM en INSERT AUTOMEM hetzelfde. Beide wijzen aan het nieuwe record een recordnummer toe dat 1 hoger is dan het nummer van het laatste record in de tabel en werken alle geopende indexen bij.

Het voordeel van INSERT AUTOMEM ten opzichte van INSERT BLANK is dat INSERT AUTOMEM een tabel slechts een keer bijwerkt, terwijl INSERT BLANK de tabel eerst bijwerkt nadat een leeg record is toegevoegd en vervolgens opnieuw als u REPLACE gebruikt om de lege waarden bij te werken.

Voorbeeld

In het volgende voorbeeld wordt een bepaald record gezocht, waarna STORE AUTOMEM wordt gebruikt om de veldnamen en -waarden voor het huidige record op te slaan. Vervolgens wordt de aanwezigheid van een tabel, Verzamel, gecontroleerd en wordt INSERT AUTOMEM gebruikt om de gegevens uit het gevonden record over te brengen naar de tabel Verzamel:

```
CLEAR
SET TALK OFF
```

```
USE Afnemers ORDER Afnemernr
Zoekop = "A3367"
SEEK Zoekop
IF FOUND()
  STORE AUTOMEM
ELSE
  RETURN
ENDIF
IF .NOT. FILE("Verzamel.DBF")
  COPY STRUCTURE TO Verzamel
ENDIF
USE Verzamel EXCLUSIVE
INSERT AUTOMEM
BROWSE
CLOSE ALL
SET TALK ON
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

APPEND AUTOMEM, INSERT

INSPECT( ) — Objecten

Opent het kenmerkenvenster, waarin de kenmerken van een object worden getoond. U kunt de kenmerken ook wijzigen in dit venster.

Syntaxis

INSPECT(*<objectverwijzing>*)

<objectverwijzing>

Een verwijzing naar het object waarvan u de kenmerken wilt wijzigen. Als u een object maakt met de operator NEW of het commando DEFINE, wordt automatisch een variabele gegenereerd met de objectverwijzing. Deze variabele heeft dezelfde naam als u aan het object hebt gegeven.

Beschrijving

Met INSPECT( ) kunt u de objectkenmerken weergeven en rechtstreeks wijzigen. Tijdens de ontwikkelingsfase van een programma kunt u bijvoorbeeld INSPECT( ) gebruiken om objecten te onderzoeken en experimenteren met verschillende kenmerkinstellingen.

Het kenmerkenvenster is niet-modaal en heeft geen invloed op de uitvoering van een programma.

Opmerking U kunt het objectkenmerkenvenster openen vanuit het venster **Formulierontwerp** door met de rechtermuisknop op het formulier of een van zijn objecten te klikken en vervolgens **Kenmerken** te kiezen in het snelmenu.

Kies een kenmerk in het objectkenmerkenvenster en druk op *F1* om Help te tonen voor het kenmerk.

Voorbeeld

In het volgende voorbeeld wordt INSPECT( ) gebruikt om een dialoogvenster te openen dat de kenmerken toont van het ringvak Ringveld1 nadat de waarde van het ringveld is gewijzigd:

```
PUBLIC F1
USE LANDEN
SET PROCEDURE TO PROGRAM(1) ADDITIVE
DEFINE FORM F1 FROM 2,2 TO 15,40;
  PROPERTY Text "Ringvelddemo"
DEFINE SPINBOX Ringveld1 OF F1;
  PROPERTY;
    Datalink "Landen->BNP",;
    Top 2,;
    Left 4,;
    Height 2,;
    Step 1000
DEFINE PUSHBUTTON Verder OF F1;
  PROPERTY;
    OnClick Kijken,;
    Top 5,;
    Left 6,;
    Width 30,;
    TEXT  "Kenmerken bekijken"
OPEN FORM F1

PROCEDURE Kijken
  ? INSPECT(F1.Ringveld1)
  CLOSE FORMS F1
  RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DISPLAY MEMORY, DISPLAY STATUS

INT( )

Numerieke gegevens

Geeft het deel voor de decimaalscheider van een opgegeven getal als resultaat.

Syntaxis

INT(<*Nuitdr*>)

<Nuitdr>

Een numeriek of zwevend getal waarvan u het deel voor de decimaalscheider als resultaat wilt geven.

Beschrijving

Met INT( ) kunt u de decimalen (cijfers achter de komma) van een numeriek of zwevend getal verwijderen en alleen het gehele deel (de cijfers voor de komma) behouden. INT( ) geeft een geheel getal van het type numerieke als resultaat.

Als u een getal met cijfers achter de komma doorgeeft aan een commando of functie, terwijl dat commando of die functie als argument een geheel getal verwacht, zoals _pageno of SLEEP, wordt dat getal automatisch ingekort en hebt u INT( ) niet nodig.

In de volgende tabel worden INT( ), FLOOR( ), CEILING( ) n ROUND( ) met elkaar vergeleken. (In deze voorbeelden is 0 als tweede argument bij ROUND( ) gebruikt.)

<Nuitdr>	INT( )	FLOOR( )	CEILING( )	ROUND( )
2,56	2	2,00	3,00	2,60
-2,56	-2	-3,00	-2,00	-2,60
2,54	2	2,00	3,00	2,50
-2,54	-2	-3,00	-2,00	-2,50

Voorbeeld

In het volgende voorbeeld wordt INT( ) gebruikt om de inhoud van een veld weer te geven:

```
SET TALK OFF
CLEAR
STORE 0 TO uurtarief, minuten, totaal
STORE "H" TO omh_oml
STORE "F    " TO valuta
@  4, 8 SAY "Welke valuta wilt u gebruiken " + ;
   "(F, $, DM, FR, YEN)" GET valuta FUNCTION "@!"
@  5,32 SAY "Wat is het uurtarief?" ;
   GET uurtarief PICTURE "999.99"
@  6,19 SAY "Welke tijdseenheid factureren (in minuten)?";
   GET minuten PICTURE "9999"
@  7,25 SAY "Wilt u omHoog of omLaag afronden? (H/L)" ;
   GET omh_oml PICTURE "!" VALID omh_oml $ "HL"
@  8,12 SAY ;
   "Totaal te factureren tijd (in minuten)?";
   GET totaal PICTURE "9999"
READ

tijd = INT(totaal/60) + (MOD(totaal,60)/60)
fact = IIF(omh_oml = "H", ;
   CEILING(totaal/minuten), ;
```

```
   FLOOR(totaal/minuten))
@ 10,12 SAY "De totaal te factureren tijd is " + ;
   LTRIM(STR(tijd,7,2)) + " uren"
dec = SET("DECIMALS")
SET DECIMALS TO 2
valu = SET("CURRENCY")
SET CURRENCY TO TRIM(valuta)
IF valuta = "YEN"
   SET CURRENCY RIGHT
ENDIF
IF valuta = "F" .OR. valuta = "DM" .OR. valuta = "FR"
   sep   = SET("SEPARATOR")
   point = SET("POINT")
   SET SEPARATOR TO "."
   SET POINT TO ","
ENDIF
@ 11,12 SAY "Het totaal is "
@ 11,27 SAY fact * uurtarief PICTURE "$$999,999.99"
SET DECIMALS TO dec
SET CURRENCY TO valu
IF valuta = "YEN"
   SET CURRENCY LEFT
ENDIF
IF valuta = "F" .OR. valuta = "DM" .OR. valuta = "FR"
   SET SEPARATOR TO sep
   SET POINT TO point
ENDIF
```

Zie ook

ABS( ), CEILING( ), FLOOR( ), ROUND( )

ISALPHA( )

Tekenreeksgegevens

Geeft .T. als resultaat als het eerste teken in een tekenreeks alfabetisch is.

Syntaxis

ISALPHA(<*Tuitdr*> | <*memoveld*>)

<*Tuitdr*> | <*memoveld*>

De tekenreeks of het memoveld om te testen.

Beschrijving

ISALPHA( ) test het eerste teken van een tekenuitdrukking of memoveld en geeft .T. als resultaat als het teken alfabetisch is. ISALPHA( ) geeft .F. als resultaat als het teken niet-alfabetisch is of als de tekenuitdrukking of memoveld leeg is.

De huidige taalaansturing bepaalt welke waarden worden beschouwd als alfabetisch. In een Amerikaanse taalaansturing zijn a tot z (kleine letters) en A tot Z (hoofdletters)

alfabetische tekens. Zie Appendix C in *Programmeren* voor meer informatie over taalaansturing.

Voorbeeld

In het volgende voorbeeld wordt ISALPHA( ) gebruikt om te bepalen of het eerste teken van elke tekenreeks een alfabetisch teken is:

```
? ISALPHA("Borland")                 && Geeft .T. als resultaat
? ISALPHA(" Borland")                && Geeft .F. als resultaat
? ISALPHA("")                        && Geeft .F. als resultaat
? ISALPHA('1e onder de databases') && Geeft .F. als resultaat
```

In het volgende voorbeeld wordt ISALPHA( ) gebruikt om te bepalen of een teken op een positie die wordt aangegeven met SUBSTR( ), alfabetisch is. De variabele Is_Alfa wordt geïnitialiseerd op .F., de procedure Straat bekijkt de inhoud van het veld Adres (met Amerikaanse adressen) totdat een alfabetisch teken wordt gevonden en geeft dan de eropvolgende reeks als resultaat:

```
CLEAR
SET TALK OFF
USE Afnemers
DO WHILE .NOT. EOF()
*Afnemers zijn Amerikaans, dus Adres = huisnummer + straat
  ? Straat(Adres)  && Alleen straatnaam
  SKIP
ENDDO

FUNCTION Straat
Parameter Voll_Adr
Voll  = TRIM(Voll_Adr)
Len   = LEN(Voll)
Start = 0
IF Len > 0                    && Controleer of langer;
                                 dan 0
   Is_Alfa = .F.
   Tek_Pos =  1
   DO WHILE .NOT. Is_Alfa     && Controleer totdat alfa
      Is_Alfa = ISALPHA(SUBSTR(Voll,Tek_Pos,1))
      Start   = Start + 1
      Tek_Pos = Tek_Pos + 1
   ENDDO
   Straat = SUBSTR(Voll,Start,(Len+1-Start))
ELSE
   Straat =  ""
ENDIF
RETURN Straat                 && Geef alleen;
                                 straatnaam
```

Overdraagbaarheid

Het argument <*memoveld*> wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CHARSET( ), ISLOWER( ), ISUPPER( ), LDRIVER( ), LOWER( ), UPPER( )

ISBLANK( )

Uitdrukkingen en gegevenstypeconversie

Bepaalt of een opgegeven veld of uitdrukking leeg is.

Syntaxis

ISBLANK(<*uitdr*>)

<uitdr>
Een uitdrukking van elk willekeurig gegevenstype.

Beschrijving

ISBLANK( ) geeft .T. als resultaat als de opgegeven uitdrukking leeg is en .F. als die gegevens bevat. een veld is leeg als het nooit gegevens heeft bevat of als het is bewerkt met het commando BLANK. ISBLANK( ) geeft in het geval van numerieke uitdrukkingen een ander resultaat dan EMPTY( ). ISBLANK( ) maakt onderscheid tussen leeg en 0, terwijl EMPTY( ) dat niet doet.

ISBLANK( ) is goed bruikbaar bij bewerkingen zoals het berekenen van gemiddelden, omdat de functie zorgt dat lege waarden niet in de berekening worden meegenomen. Als het onderscheid tussen 0 en leeg in numerieke velden niet van belang is, u zowel ISBLANK( ) als EMPTY( ) gebruiken.

Voorbeeld

De volgende interactieve instructies in het commandovenster laten de resultaten van ISBLANK( ) goed zien:

```
Leeg = SPACE(20)
? ISBLANK(Leeg)             && Geeft .T. als resultaat
Leeg = "        "
? ISBLANK(Leeg)             && Geeft .T. als resultaat
mDatum = {  -  -  }         && of  { }
? ISBLANK(mDatum)           && Geeft .T. als resulaat
USE Bedrijf
APPEND BLANK                && Nieuw leeg record invoegen
? ISBLANK(VerkTotNu)        && Geeft .T. als resulaat
REPLACE VerkTotNu WITH 0
? ISBLANK(VerkTotNu)        && Geeft .F. als resulaat
BLANK FIELDS VerkTotNu
? ISBLANK(VerkTotNu)        && Geeft .T. als resulaat
? ISBLANK(Notities)         && Geeft .T. voor memoveld
REPLACE Notities WITH "Iets voor het memoveld"
? ISBLANK(Notities)         && Geeft .F. voor memoveld
```

U kunt ISBLANK( ) ook gebruiken om lege velden uit te sluiten bij berekeningen.

```
CALCULATE AVG(VerkTotNu) FOR .NOT. ISBLANK(VerkTotNu)
```

In het volgende voorbeeld wordt ISBLANK( ) gebruikt om voor een rapport alleen die records in de tabel Afnemers te selecteren die een niet-lege waarde bevatten in het veld OpenBalans:

```
SET SAFETY OFF
SET TALK OFF
USE Afnemers EXCLUSIVE
INDEX ON Bedrijf TAG Bedrijf
BLANK FIELDS OpenBalans FOR OpenBalans = 0
? CENTER("Balansrapport")
?
SCAN
  IF .NOT. ISBLANK(OpenBalans)
    ? BEDRIJF + "Nog te goed:  " + ;
      TRANSFORM(OpenBalans,"@$999,999,999.99")
  ENDIF
ENDSCAN
RETURN
```

Zie EMPTY( ) voor meer voorbeelden met ISBLANK( ).

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

APPEND, BLANK, EMPTY( ), SPACE( ), TYPE( )

ISCOLOR( )

Kleuren en fonts

Geeft .T. als resultaat als het systeem over een kleurenmonitor beschikt en .F. als de monitor monochroom is . ISCOLOR( ) wordt hoofdzakelijk ondersteund vanwege de compatibiliteit met dBASE IV. dBASE voor Windows bepaalt het kleurpalet op basis van de instellingen voor **Kleur** in het **Configuratiescherm** van Windows.

Zie Help voor meer informatie over ISCOLOR( ). In het handboek van Windows vindt u meer informatie over het gebruik van kleuren onder Windows.

ISLOWER( )

Tekenreeksgegevens

Geeft .T. als resultaat als het eerste teken in een tekenreeks een kleine letter is.

Syntaxis

ISLOWER(<*Tuitdr*> | <*memoveld*>)

<Tuitdr> | <memoveld>

De tekenreeks of het memoveld om te testen.

Beschrijving

ISLOWER( ) test het eerste teken van een tekenuitdrukking of memoveld en geeft .T. als resultaat als dat een kleine letter is. ISLOWER( ) geeft .F. als resultaat als het teken geen kleine letter is of als de tekenuitdrukking of het memoveld leeg is.

De huidige taalaansturing bepaalt welke tekens worden beschouwd als kleine letters. In een Amerikaanse taalaansturing zijn a tot z kleine letters en A tot Z hoofdletters. Zie Appendix C in *Programmeren* voor meer informatie over taalaansturing.

Voorbeeld

In het volgende voorbeeld wordt ISLOWER( ) gebruikt om te bepalen of het eerste teken van elke tekenreeks een kleine letter is:

```
? ISLOWER("Borland")                  && Geeft .F. als resultaat
? ISLOWER(" Borland")                 && Geeft .F. als resultaat
? ISLOWER("")                         && Geeft .F. als resultaat
? ISLOWER('le onder de databases')    && Geeft .F. als resultaat
? ISLOWER("software voor kenners")    && Geeft .T. als resultaat
```

Overdraagbaarheid

Het argument *<memoveld>* wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CHARSET( ), ISALPHA( ), ISUPPER( ), LDRIVER( ), LOWER( ), UPPER( )

ISMOUSE( )

Toetsenbord- en muisacties

Geeft waar (.T.) als resultaat als op het huidige systeem een stuurprogramma voor de muis is geïnstalleerd en onwaar (.F.) als dat niet het geval is.

Syntaxis

ISMOUSE( )

Beschrijving

Met ISMOUSE( ) kunt u vaststellen of op het huidige systeem een stuurprogramma voor de muis is geïnstalleerd. U kunt het resultaat van ISMOUSE( ) bijvoorbeeld gebruiken om in het programma een keuze te maken tussen meldingen als "Klik op OK" of "Kies OK".

Voorbeeld

In het volgende voorbeeld wordt ISMOUSE( ) gebruikt om te controleren of de muis kan worden gebruikt of niet. In het resultatenpaneel van het commandovenster wordt een toepasselijke melding getoond:

```
IF ISMOUSE()
  ? "U kunt de muis gebruiken" AT 5
ELSE
  ? "U kunt de muis niet gebruiken" AT 5
ENDIF
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

INKEY( ), MDOWN( ), MCOL( ), MROW( ), ON MOUSE

ISTABLE( ) Tabellen

Test op de aanwezigheid van een tabel in een opgegeven database en geeft .T. als resultaat als de tabel bestaat en anders .F.

Syntaxis

ISTABLE(<*tabelnaam*>)

<tabelnaam>

De naam van de tabel die u zoekt. U moet het volledige pad opgeven als de tabel niet in de huidige directory of in een met SET PATH ingestelde directory staat. U kunt ook een pad opgeven dat in de huidige directory begint.

U kunt ook testen op de aanwezigheid van een tabel in een database (gedefinieerd met het IDAPI-configuratieprogramma) door voor de naam van de tabel de naam van de database op te geven (tussen dubbelepunten) zoals *:databasenaam:tabelnaam*. Als de database nog niet is geopend, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

Beschrijving

Met ISTABLE( ) kunt u controleren of een tabel van het met SET DBTYPE opgegeven type bestaat. Als de tabel niet in de huidige directory staat, moet u een volledig pad opgeven. U kunt ook een database opgeven als u zoekt naar een tabel die niet in de huidige database staat.

Voorbeeld

In het volgende voorbeeld wordt ISTABLE( ) gebruikt om te controleren of de opgegeven Paradox-tabel in de directory VOORBD aanwezig is. Als dat het geval is, wordt de tabel geopend met BROWSE:

```
CLOSE ALL
CLEAR
```

```
IF ISTABLE("C:\DBASEWIN\VOORBD\Klanten.DB")
  USE Klanten
  BROWSE
ELSE
  ? "Tabel bestaat niet"
ENDIF
RETURN
```

Zie ook

DIR, DISPLAY FILES, FILE( ), GETFILE( ), PUTFILE( ), SET DEFAULT, SET DATABASE, SET DBTYPE, SET DIRECTORY, SET PATH

ISUPPER( ) — Tekenreeksgegevens

Geeft .T. als resultaat als het eerste teken van een tekenreeks een hoofdletter is.

Syntaxis

ISUPPER(<*Tuitdr*> | <*memoveld*>)

<*Tuitdr*> | <*memoveld*>

De tekenreeks of het memoveld om te testen.

Beschrijving

ISUPPER( ) test het eerste teken van een tekenuitdrukking of memoveld en geeft .T. als resultaat als dat een hoofdletter is. ISUPPER( ) geeft .F. als resultaat als het teken geen hoofdletter is of als de tekenuitdrukking of het memoveld leeg is.

De huidige taalaansturing bepaalt welke tekens worden beschouwd als hoofdletters. In een Amerikaanse taalaansturing zijn a tot z kleine letters en A tot Z hoofdletters. Zie Appendix C in *Programmeren* voor meer informatie over taalaansturing.

Voorbeeld

In het volgende voorbeeld wordt ISUPPER( ) gebruikt om te bepalen of het eerste teken van elke tekenreeks een hoofdletter is:

```
? ISUPPER("Borland")                 && Geeft .T. als resultaat
? ISUPPER(" Borland")                && Geeft .F. als resultaat
? ISUPPER("")                        && Geeft .F. als resultaat
? ISUPPER('1e onder de databases')   && Geeft .F. als resultaat
? ISUPPER("Software voor kenners")   && Geeft .T. als resultaat
```

Overdraagbaarheid

Het argument <*memoveld*> wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CHARSET( ), ISALPHA( ), ISLOWER( ), LDRIVER( ), LOWER( ), UPPER( )

ITOH( )

Uitdrukkingen en gegevenstypeconversie

Geeft een tekenreeks met de hexadecimale notatie van een opgegeven decimaal getal als resultaat.

Syntaxis

ITOH(<*Nuitdr1*>[, <*Nuitdr2*>])

<*Nuitdr1*>

Het decimale getal dat u wilt omzetten naar hexadecimaal.

<*Nuitdr2*>

Het aantal tekens dat als resultaat moet worden gegeven. Als <*Nuitdr2*> groter is dan het aantal hexadecimale cijfers, wordt de reeks aangevuld met voorloopnullen. ITOH(21) geeft bijvoorbeeld de tekenreeks "15" als resultaat, terwijl ITOH(21,4) "0015" als resultaat geeft.

Als <*Nuitdr2*> kleiner is dan het aantal hexadecimale cijfers in het resultaat, wordt dit argument genegeerd.

Beschrijving

Met ITOH( ) kunt u een decimaal getal omzetten naar een tekenreeks met de hexadecimale notatie van het getal. ITOH( ) is het tegengestelde van HTOI( ), waarmee tekenreeksen met een hexadecimaal getal worden omgezet in een decimaal getal. U kunt beide functies gebruiken voor aanroepen naar de API (Application Programming Interface) van Windows waarvoor hexadecimale waarden verplicht zijn.

Voorbeeld

```
? ITOH(13824,4)  && Geeft 3600 als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

HTOI( )

JOIN

Tabelindeling

Combineert records in de huidige tabel met records uit een opgegeven tabel om zo een nieuwe tabel te maken.

Syntaxis

JOIN WITH *<alias>* TO *<bestandsnaam>* | ? | *<bestandsnaamfilter>*
[[TYPE] [PARADOX | DBASE]]
FOR *<voorwaarde>*
[FIELDS *<veldenlijst>*]

<alias>

De alias van de tabel waaruit records moeten worden gecombineerd met records uit de huidige tabel.

TO *<bestandsnaam>* | ? | *<bestandsnaamfilter>*

Maakt het tabelbestand *<bestandsnaam>*. Standaard wordt de extensie .DBF gebruikt voor *<bestandsnaam>* en wordt het bestand opgeslagen in de huidige directory. De opties ? en *<bestandsnaamfilter>* tonen een dialoogvenster waarin u de naam opgeeft van het doelbestand en van de directory waar het moet worden opgeslagen.

U kunt ook een tabel maken in een database (gedefinieerd met de IDAPI-configuratieprogramma) door voor de naam van de tabel de naam van de database op te geven (tussen dubbelepunten), zoals *:databasenaam:tabelnaam*. Als de database nog is geopend, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

[TYPE] PARADOX | DBASE

Geeft aan welk type tabel moet worden gemaakt. Het sleutelwoord TYPE is er alleen voor de leesbaarheid; het heeft geen gevolgen voor de werking van het commando.

Als u DBASE opgeeft, wordt een dBASE-tabel gemaakt (de standaardinstelling). Als u geen extensie opgeeft voor *<bestandsnaam>*, wordt de extensie .DBF gebruikt.

Als u PARADOX opgeeft, wordt een Paradox-tabel gemaakt met de extensie .DB.

FOR *<voorwaarde>*

Beperkt JOIN tot records in de huidige tabel die voldoen aan de opgegeven *<voorwaarde>*.

FIELDS *<veldenlijst>*

Beperkt de velden in de nieuwe tabel tot de velden die zijn genoemd in *<veldenlijst>*. Namen van velden in de huidige tabel geeft u rechtstreeks op en velden uit de aliastabel in de notatie *alias->veld*. Zonder FIELDS bevat de nieuwe tabel alle velden uit beide tabellen.

Beschrijving

Met JOIN kunt u een nieuwe tabel maken door de records uit twee bestaande tabellen te combineren. In de veldenlijst kan elk type veld uit beide tabellen worden opgenomen, behalve binaire, memo- en OLE-velden.

In plaats van een veldenlijst op te geven, kunt u ook het commando SET FIELDS gebruiken voordat u JOIN gebruikt. In dat geval worden in de nieuwe tabel alleen de velden opgenomen die zijn opgegeven bij SET FIELDS. Als u geen veldenlijst opgeeft, worden eerst de velden uit de huidige tabel en dan die uit de tweede tabel opgenomen. Als beide tabellen een veld met dezelfde naam bevatten, wordt alleen het veld uit de eerste tabel aan de nieuwe tabel toegevoegd. Het veld uit de tweede tabel wordt dan genegeerd.

De FOR-voorwaarde wordt gewoonlijk gebruikt om de waarde van velden in de ene tabel te vergelijken met de waarde van velden in de andere tabel. In de eenvoudigste situatie wordt de gelijkheidsoperator (=) gebruikt om te bepalen of waarden in de ene tabel gelijk zijn aan waarden in de andere tabel (of de veldnamen nu gelijk zijn of niet). Wees voorzichtig bij het samenstellen van een FOR-voorwaarde en let er op dat alleen de gewenste records in de nieuwe tabel worden opgenomen, want het commando JOIN kan zeer grote tabellen opleveren. Het aantal records in de nieuwe tabel is dan gelijk aan het aantal records in de eerste tabel maal het aantal records in de tweede tabel.

Vanwege het aantal records dat wordt vergeleken, kan de uitvoering van het commando JOIN een heleboel tijd kosten. U kunt ook tijdelijk gegevens uit meer dan een tabel combineren met het commando SET RELATION. Dat kan veel tijd besparen.

Voorbeeld

In het volgende voorbeeld wordt JOIN gebruikt om een nieuwe tabel te maken met opgegeven velden uit beide oorspronkelijke tabellen op basis van een opgegeven relatie tussen de twee tabellen:

```
CLOSE DATABASE
USE Bedrijf IN SELECT() && Werkgebied 1
USE Contact IN SELECT() && Werkgebied 2
SELECT Contact
JOIN WITH Bedrijf ;
   FOR Contact->CompCode = Bedrijf->CompCode ;
   FIELDS Contact->CompCode, Bedrijf->Bedrijf,;
     Contact->Contact;
   TO CntcVerg
```

De tabel CntcVerg is gemaakt met een veldenstructuur op basis van de velden in de clausule FIELDS. De nieuwe tabel bevat de velden CompCode, Bedrijf en Contact. De FOR-clausule was noodzakelijk om te zorgen dat elke bedrijf is gekoppeld aan de contactpersonen bij dat bedrijf.

```
SELECT 3
USE CntcVerg
LIST OFF
```

Zie ook

SELECT, SET RELATION, USE

KEY( ) Tabelindeling

Geeft de sleuteluitdrukking als resultaat die is gebruikt toen de opgegeven index werd gemaakt.

Syntaxis

KEY([<*.mdx-bestandsnaam Tuitdr*>,] <*indexpositie Nuitdr*> [, <*alias*>])

<*.mdx-bestandsnaam Tuitdr*>

Geeft een meervoudig indexbestand aan dat de indexlabel bevat dat u wilt controleren.

<*indexpositie Nuitdr*>

Kiest een indexbestand of -label op basis van de positie van een indexlabel in een .MDX-bestand of de positie van een indexbestand in de lijst van geopende indexen voor het huidige of een opgegeven werkgebied.

<*alias*>

Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

De functie KEY( ) geeft de sleuteluitdrukking als resultaat die is gebruikt toen de opgegeven index werd gemaakt. Als er meerdere .NDX-indexen of meerdere indexen met labelnamen in .MDX-bestanden zijn geopend, kunt u KEY( ) gebruiken om de sleuteluitdrukking voor elke index te bepalen.

Als in het huidige of opgegeven werkgebied geen index is geopend, of als <*indexpositie Nuitdr*> geen positie van een index in de indexlijst oplevert, geeft KEY( ) een lege tekenreeks string ("") als resultaat.

Voorbeeld

In het volgende voorbeeld wordt KEY( ) gebruikt om de sleuteluitdrukking van indexen op te halen:

```
USE Bedrijf  EXCLUSIVE
INDEX ON CompCode TAG CompCode
INDEX ON Postcode+Bedrijf TAG PCBedrijf

LabelCompCode=TAGNO("CompCode")
* Labelnummer van CompCode ophalen
LabelPCBedrijf=TAGNO("PCBedrijf")
* Labelnummer van PCBedrijf ophalen

? LabelCompCode,  TAG(LabelCompCode), KEY(LabelCompCode)
? LabelPCBedrijf, TAG(LabelPCBedrijf),;
                  KEY(LabelPCBedrijf)
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

INDEX, NDX( ), ORDER( ), SET INDEX, SET ORDER, TAG( ), TAGCOUNT( ), TAGNO( ), USE

KEYBOARD

Toetsenbord- en muisacties

Plaatst de waarde van <*Tuitdr*> in de typbuffer. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows maken formulieren geen gebruik van de typbuffer.

Zie Help voor meer informatie over de syntaxis van KEYBOARD. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het maken van formulieren.

KEYMATCH( )

Tabelindeling

Geeft aan of de opgegeven uitdrukking voorkomt in een index.

Syntaxis

KEYMATCH (<*uitdrukking*> [,<*indexpositie Nuitdr*> |
[<*.mdx-bestandsnaam Tuitdr*>,]<*label Nuitdr*>]
[,<*alias*>]])

<uitdrukking>

Geeft de uitdrukking van elk willekeurig gegevenstype aan die u wilt zoeken. In het geval van Paradox- en SQL-tabellen kunt u een of meer waarden (gescheiden door komma's) opgeven die overeenkomen met enkelvoudige of samengestelde indexsleutelvelden.

<indexpositie Nuitdr>

Geeft een .NDX-bestand aan door middel van de positie van de index in de lijst met geopende indexen voor de huidige of een opgegeven tabel.

<.mdx-bestandsnaam Tuitdr>

Geeft een meervoudig indexbestand aan dat de indexlabel bevat dat u wilt controleren.

<label Nuitdr>

Geeft een indexlabel aan door middel van de positie van een indexlabel in een .MDX-bestand voor de huidige of een opgegeven tabel.

<alias>
Definieert het werkgebied waarin de opgegeven .NDX of .MDX is geopend. U kunt een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam opgeven. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

De functie KEYMATCH( ) controleert of opgegeven sleuteluitdrukkingen voorkomen in een bepaalde index. KEYMATCH( ) geeft .T. of .F. als resultaat om aan te geven of de opgegeven uitdrukking is gevonden. SET EXACT bepaalt of de overeenkomst exact moet zijn.

De functie KEYMATCH( ) wordt veel gebruikt om te controleren op dubbele waarden tijdens een bewerking met APPEND.

KEYMATCH( ) zoekt alleen in het opgegeven indexbestand of de opgegeven indexlabel. De instellingen voor SET DELETED, SET FILTER en SET KEY worden genegeerd, zodat de betrouwbaarheid van de gegevens in een tabel gewaarborgd blijft, zelfs als u met een deelverzameling van de tabelrecord werkt.

Als u alleen de uitdrukking (*exp*) opgeeft als te zoeken waarde, wordt in de huidige hoofdindex gezocht naar een indexsleutel met dezelfde waarde. Als een overeenkomstige indexsleutel wordt gevonden, geeft KEYMATCH( ) .T. als resultaat

Voorbeeld

In het volgende voorbeeld wordt KEYMATCH( ) gebruikt om te bepalen of een doorgegeven waarde voorkomt in de index:

```
SET EXACT OFF       && KEYMATCH wordt beinvloed door SET EXACT
USE Bedrijf EXCLUSIVE
INDEX ON Bedrijf  TAG BedrijfNW;
  FOR Regio = "NW"
INDEX ON Plaats TAG Plaats OF Lokatie
? KEYMATCH("VOF",TAGNO("BedrijfNW"))
? KEYMATCH("Maarssen","Lokatie",TAGNO("Plaats"))
```

In beide gevallen wordt .T. als resultaat gegeven.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

INDEX, KEY( ), MDX( ), NDX( ), ORDER( ), SET INDEX, SET ORDER, USE

LABEL FORM

Invoer/uitvoer

Genereert etiketten op basis van een etiketopmaak in een opgegeven etiketbestand en informatie uit records in de huidige tabel. De etiketten worden weergegeven of afgedrukt.

Syntaxis

LABEL FORM *<bestandsnaam1>* | ? | *<bestandsnaamfilter1>*
[*<bereik>*] [FOR *<voorwaarde1>*] [WHILE *<voorwaarde2>*]
[SAMPLE]
[TO FILE *<bestandsnaam2>* | ?] | *<bestandsnaamfilter2>*
[TO PRINTER]

<bestandsnaam1> | ? | *<bestandsnaamfilter>*

Het bestand met de etiketopmaak. De opties ? en *<bestandsnaamfilter>* tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, wordt gezocht naar een bestand met de extensie .RPL, .LBG of .LBL en wel in die volgorde.

<bereik>

Het aantal records in de huidige tabel dat moet worden gebruikt voor etiketten. RECORD *<n>* geeft een enkel record aan door middel van zijn recordnummer. NEXT *<n>* geeft *n* records aan, te beginnen bij het huidige record. ALL specificeert alle records. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR *<voorwaarde1>*
WHILE *<voorwaarde2>*

Bepaalt welke records worden gebruikt door LABEL FORM. FOR beperkt LABEL FORM tot records die voldoen aan *<voorwaarde1>*. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang *<voorwaarde2>* waar is.

SAMPLE

Toont een etiket met asterisken in plaats van tekst of drukt het af, en vraagt vervolgens om meer voorbeelden.

TO FILE *<bestandsnaam2>* | ? | *<bestandsnaamfilter>*

Stuurt uitvoer naar het tekstbestand *<bestandsnaam>*. Standaard wordt de extensie .TXT toegekend aan *<bestandsnaam>* en wordt het bestand opgeslagen in de huidige directory. De opties ? en *<bestandsnaamfilter>* tonen een dialoogvenster waarin u de naam en de directory van het doelbestand kunt opgeven.

TO PRINTER

Stuurt uitvoer naar de printer.

Beschrijving

Met LABEL FORM kunt u etiketten afdrukken of weergeven met de opmaak die u in Rapportontwerp hebt gedefinieerd met CREATE LABEL of MODIFY LABEL. Zie de Crystal Reports documentatie voor meer informatie over werken met Rapportontwerp. Als u geen *<bereik>*, WHILE *<voorwaarde1>* of FOR *<voorwaarde2>* opgeeft, worden voor elk record de etiketspecificaties afgedrukt, op volgorde van recordnummer of aan de hand van de indexvolgorde.

Als u etiketten afdrukt of weergeeft waarin gegroepeerde gegevens of groepssubtotalen zijn opgenomen, moet de huidige tabel gesorteerd zijn of moet de hoofdindex zijn geopend. het gesorteerde bestand of de index moet gerangschikt zijn op het veld waarop de gegevens zijn gegroepeerd.

LABEL FORM zonder de optie TO FILE of TO PRINTER geeft de etiketten weer in het resultatenpaneel van het commandovenster of in het huidige door de gebruiker gedefinieerde venster.

Voorbeeld

In het volgende voorbeeld wordt de tabel Bedrijf geopend en worden etiketten gegenereerd op basis van het etikettenformulier Bdretik1:

```
CLOSE DATABASE
USE Bedrijf
LABEL FORM Bdretik1 TO PRINT
* Dit etiketformaat heeft 6 regels per etiket
* met Bedrijf, Straat, Postcode, Plaats en Regio:
*
* Akkerman BV
* Linge 297-307
* 3430 AA Nieuwegein NW
*
*
*
* Merckx & Cie.
* Chaussée Paul Verhaegen 4
* 1932 Zaventem BW
*
*
*
```

Overdraagbaarheid

De optie *<bestandsnaamfilter>* wordt niet ondersteund in dBASE IV en dBASE III PLUS.

Zie ook

CREATE LABEL

LASTKEY( )

Toetsenbord- en muisacties

Geeft als resultaat de waarde van de laatste toets of toetscombinatie die is gebruikt om de uitvoering te beëindigen van een commando dat het volledige scherm in beslag neemt. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows kenmerken als OnClick om handelingen uit te voeren aan de hand van de manier waarop de gebruiker het formulier sluit.

Zie Help voor meer informatie over de syntaxis van LASTKEY( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het maken van formulieren.

LDRIVER( )

Omgeving

Geeft de naam als resultaat van de taalaansturing die wordt gebruikt door de huidige of een opgegeven tabel. Als geen tabel is geopend en u gebruikt LDRIVER( ) zonder argument, wordt de globale taalaansturing als resultaat gegeven.

Syntaxis

LDRIVER([<*alias*>])

<alias>

Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

Met LDRIVER( ) kunt u bepalen welke taalaansturing de huidige of een opgegeven tabel gebruikt. Als u geen argument doorgeeft aan LDRIVER( ), wordt de naam van de taalaansturing van de huidige tabel als resultaat gegeven, of als er geen tabel is geopend, de naam van de globale taalaansturing. LDRIVER( ) geeft ook informatie als resultaat over Paradox- en SQL-databases.

Zie Appendix C in *Programmeren* voor meer informatie over taalaansturingen.

De taalaansturing voor een tabel is afhankelijk van de DOS-codetabel of de IDAPI-taalaansturing die van kracht was toen de tabel werd gemaakt. In het geval van dBASE voor Windows kunt u de taalaansturing voor uw dBASE-gegevens instellen in de sectie onder [CommandSettings] in het bestand DBASEWIN.INI. U kunt bijvoorbeeld een Duitse taalaansturing laden als u moet werken met een tabel die is gemaakt toen die aansturing actief was.

Voorbeeld

In het volgende voorbeeld wordt de functie LDRIVER( ) en een voorbeeld van een resultaat getoond:

```
? LDRIVER()   && DB437NL0
```

In het volgende voorbeeld worden alle tabellen gesloten en wordt de globale taalaansturing opgevraagd. Vervolgens wordt een tabel geopend en wordt gecontroleerd of die tabel is gemaakt toen de globale taalaansturing actief was. Als dat niet zo is, wordt een waarschuwing weergegeven:

```
CLOSE ALL
SET LDCHECK OFF
* programma vervangt waarschuwing van LDCHECK
GlobTaalaanst=LDRIVER()
USE c:\dbasewin\voorbd\music\configs.dbf
TabelTaalaanst=LDRIVER()
SET EXACT ON
IF GlobTaalaanst<>TabelTaalaanst
  ? "Waarschuwing: deze tabel is gemaakt"+;
    "met een andere taalaansturing"
  ? "Globale taalaansturing: "+GlobTaalaanst
  ? "Taalaansturing voor "+DBF()+": "+TabelTaalaanst
  WAIT
ENDIF
SET EXACT OFF
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ANSI( ), CHARSET( ), OEM( ), SET LDCHECK

LEFT( ) — Tekenreeksgegevens

Geeft een opgegeven aantal tekens vanaf de linkerkant van een tekenreeks of memoveld als resultaat.

Syntaxis

LEFT(<*Tuitdr*> | <*memoveld*>, <*lengte Nuitdr*>)

<Tuitdr> | <memoveld>

De tekenreeks of het memoveld waaruit tekens moeten worden opgehaald.

<lengte Nuitdr>

Het aantal tekens dat vanaf het begin van de tekenreeks of het memoveld moet worden opgehaald.

Beschrijving

LEFT( ) geeft een opgegeven aantal, *<lengte Nuitdr>*, tekens vanaf de linkerkant van een tekenreeks of memoveld als resultaat. De functie geeft maximaal 32.766 tekens als resultaat (de maximumlengte van een tekenreeks).

Als <*lengte Nuitdr*> groter is dan het aantal tekens in de opgegeven tekenreeks of het opgegeven memoveld, geeft LEFT( ) de reeks als resultaat zonder tekens toe te voegen om de opgegeven lengte te maken. Met LEN( ) kunt u de werkelijke lengte van de als resultaat gegeven tekenreeks bepalen.

Als <*lengte Nuitdr*> nul of kleiner dan nul is, geeft LEFT( ) een lege tekenreeks als resultaat. Als <*lengte Nuitdr*> groter dan nul of gelijk aan nul is, geeft LEFT(<*Tuitdr*>, <*lengte Nuitdr*>) hetzelfde resultaat als SUBSTR(<*Tuitdr*>, 1, <*lengte Nuitdr*>).

Als LEFT( ) tekens als resultaat geeft uit een memoveld, worden de twee tekens voor elke wagenterugloop/regeldoorvoer meegeteld.

Voorbeeld

In het volgende voorbeeld wordt LEFT( ) gebruikt om een deel van een tekenreeks vanaf het begin van de reeks als resultaat te geven:

```
? LEFT("dBASE",1)              && Geeft "d" als resultaat
? LEFT("dBASE",3)              && Geeft "dBA" als resultaat
? LEFT("dBASE",9)              && Geeft "dBASE" als resultaat
? LEFT("dBASE",0)              && Geeft "" als resultaat
```

In het volgende voorbeeld worden ISALPHA( ) en SUBSTR( ) gebruikt om de eerste letter in een tekenreeks (Adres) als resultaat te geven. De op die manier verkregen variabele (Bij) wordt vervolgens gebruikt als lengteparameter voor LEFT( ) om alleen het deel met het straatnummer uit het veld Adres te tonen:

```
CLOSE DATABASES
SET TALK OFF
CLEAR
USE Afnemers
DO WHILE .NOT. EOF()
? Huis_Nr(Adres)               && Geeft alleen
                               && huisnummer
SKIP
ENDDO
CLOSE DATABASES

FUNCTION Huis_Nr
Parameter Voll_Adr
Voll  = TRIM(Voll_Adr)
Len   = LEN(Voll)
Bij   = 0
IF Len > 0                     && Controleer of langer
   Is_Alfa = .F.               && dan 0
   Tek_Pos = 1
   DO WHILE .NOT. Is_Alfa      && Controleer tot alfa
      Is_Alfa = ISALPHA(SUBSTR(Voll,Tek_Pos,1))
      IF .NOT. Is_Alfa         && Optellen bij variabele
         Bij = Bij + 1         && Bij als ISALPHA() = .F.
      ENDIF
      Tek_Pos = Tek_Pos + 1
   ENDDO
   Straat = LEFT(Voll,Bij-1)
ELSE
   Straat = ""
```

```
ENDIF                          && Geeft
RETURN Straat                  && huisnummer
```

Overdraagbaarheid

Het argument <*memoveld*> wordt niet ondersteund in dBASE III PLUS. In zowel dBASE III PLUS als dBASE IV is de resultaatwaarde van LEFT( ) beperkt tot 254 tekens.

Zie ook

AT( ), LEN( ), RIGHT( ), SUBSTR( )

LEN( )

Tekenreeksgegevens

Geeft het aantal tekens in een opgegeven tekenreeks of memoveld als resultaat.

Syntaxis

LEN(<*Tuitdr*> | <*memoveld*>)

<*Tuitdr*> | <*memoveld*>

De tekenreeks of het memoveld waarvan de lengte moet worden bepaald.

Beschrijving

LEN( ) geeft het aantal tekens (de *lengte*) van een tekenreeks of memoveld als resultaat. De lengte van een lege tekenreeks of een leeg memoveld is nul. LEN( ) telt null-bytes, CHR(0), mee. Als LEN( ) de lengte van een memoveld berekent, worden voor elke regeldoorvoer twee tekens geteld.

De lengte van numerieke of zwevende gegevens bepaalt u met LENNUM( ).

Met MEMLINES( ) kunt u het aantal regels in een memoveld bepalen. De lengte van een bepaalde regel in een memoveld bepaalt u met LEN( ) in combinatie met MLINE( ). De instructie LEN(MLINE(Beschr,3)) geeft bijvoorbeeld de lengte van de derde regel in het Beschr als resultaat.

Voorbeeld

In de volgende voorbeelden wordt LEN( ) gebruikt om de lengte van enkele tekenreeksen te bepalen:

```
? LEN("Hoi!")                  && Geeft 6 als resultaat
? LEN("")                      && Geeft 0 als resultaat
? LEN("Tot" + " Straks")       && Geeft 10 als resultaat
```

In het volgende voorbeeld wordt LEN( ) gebruikt om het aantal tekens in het memoveld Notities te bepalen. Als Notities leeg is, geeft LEN( ) 0 als resultaat:

```
USE Klanten
SCAN
```

```
    ? IIF(LEN(Notities)>0,Notities,"Record ";
    + LTRIM(STR(RECNO()))+" bevat geen notities")
ENDSCAN
CLOSE DATABASES
```

Overdraagbaarheid

Het argument <*memoveld*> wordt niet ondersteund in dBASE III PLUS, en dBASE III PLUS en dBASE IV tellen null-bytes niet mee.

Zie ook

LENNUM( ), MEMLINES( ), MLINE( ), TRIM( )

LENNUM( )

Numerieke gegevens

Geeft de weergegeven lengte van een opgegeven getal als resultaat, inclusief voorloopspaties.

Syntaxis

LENNUM(<*Nuitdr*>)

<Nuitdr>

Het numerieke of zwevende getal waarvan u de weergegeven lengte wilt bepalen.

Beschrijving

Gebruik LENNUM( ) voordat u numerieke waarden met verschillende lengtes weergeeft.

Als u aan LENNUM( ) de naam van een numeriek veld doorgeeft, wordt de veldlengte als resultaat gegeven.

Als een getal uit acht of minder cijfers voor de komma bestaat en geen decimaalscheider bevat, is het een getal van het type numeriek. De standaard weergegeven lengte van numerieke getallen is 10. De instructie LENNUM(123) geeft bijvoorbeeld 10 als resultaat.

Als een getal meer dan acht cijfers voor de komma en/of een decimaalscheider bevat, is het een zwevend getal. De standaard weergegeven lengte van zwevende getallen is 10 voor de cijfers voor de decimaalscheider, één positie voor de decimaalscheider, plus het aantal decimalen dat is ingesteld met SET DECIMALS. Als voor SET DECIMALS bijvoorbeeld 5 is ingesteld, geeft LENNUM(3,14159) 16 als resultaat.

Voorbeeld

In het volgende voorbeeld wordt LENNUM( ) gebruikt om de tekenlengte (weergegeven lengte) van een getal te bepalen:

```
CLEAR
dec = SET("DECIMALS")
FOR x = 3 TO 9 STEP 1
   SET DECIMALS TO x
   getal = 3.2 * 1.53
   ? "De uitkomst van 3,2 * 1,53 - " + ;
      STR(getal,7 + x,x) + " - is " + ;
      LTRIM(STR(LENNUM(getal),2,0)) + ;
      " tekenposities lang"
   ? "  als voor SET DECIMALS " + ;
      LTRIM(STR(x,1,0))+ " is ingesteld"
   ?
NEXT
SET DECIMALS TO dec
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS. In dBASE III PLUS en dBASE IV is een getal per definitie van het type numeriek, onafhankelijk van de lengte en de eventuele aanwezigheid van een decimaalscheider en decimalen. In dBASE IV moet u FLOAT( ) gebruiken om een getal het type zwevend te geven.

Zie ook

FLOAT( ), LEN( ), SET DECIMALS, STR( )

LIKE( )

Tekenreeksgegevens

Geeft .T. als resultaat als een opgegeven tekenreeks overeenkomt met een opgegeven patroonreeks.

Syntaxis

LIKE(<*filter Tuitdr*>, <*Tuitdr*> | <*memoveld*>)

<filter Tuitdr>

Een tekenreeks die bestaat uit tekens en jokers. De jokers zijn ? en *.

<Tuitdr> | <memoveld>

De tekenreeks of het memoveld om met de patroonreeks te vergelijken.

Beschrijving

Met LIKE( ) kunt u een tekenreeks vergelijken met een andere tekenreeks. Het argument <*filter Tuitdr*> bevat jokers en stelt een patroon voor. Het argument <*Tuitdr*> of <*memoveld*> wordt vergeleken met dit patroon. LIKE( ) geeft .T. als resultaat als <*Tuitdr*> of <*memoveld*> een tekenreeks oplevert die overeenkomt met <*filter Tuitdr*>. Als u twee tekenreeksen fonetisch wilt vergelijken in plaats van letterlijk, kunt u DIFFERENCE( ) gebruiken.

Het patroon voor *<filter Tuitdr>* maakt u met de jokers ? en *. Een asterisk (*) stelt nul, een of meer tekens voor. Het vraagteken (?) staat voor één enkele teken. beide jokers mogen op elke positie en meer dan eens voorkomen in *<filterreeks>*. De jokers in *<filter Tuitdr>* kunnen hoofdletters en/of kleine letters voorstellen.

Als in *<Tuitdr>* of *<memoveld>* ? of * voorkomt, worden die geïnterpreteerd als vaste tekens en niet als jokers, zoals u kunt zien in de volgende voorbeelden:

```
LIKE("a*d","abcd") && Geeft .T. als resultaat
LIKE("a*d","aBCd") && Geeft .T. als resultaat
LIKE("abcd","a*d") && Geeft .F. als resultaat
```

LIKE( ) maakt onderscheid tussen hoofdletters en kleine letters. Gebruik UPPER( ) of LOWER( ) om met LIKE( ) vergelijkingen uit te voeren zonder onderscheid tussen hoofdletters en kleine letters, zoals in de volgende voorbeelden:

```
LIKE("*xyz","uvwxyz")                 && Geeft .T. als resulaat
LIKE("*xyz","UVWXYZ")                 && Geeft .F. als resulaat
LIKE(LOWER("*xyz"),LOWER("UVWXYZ"))   && Geeft .T. als resulaat
```

LIKE( ) geeft .T. als resultaat als beide argumenten uit lege tekenreeksen bestaan. LIKE( ) geeft .F. als resultaat een van beide argumenten leeg is en het andere niet.

LIKE( ) wordt niet beïnvloed door SET EXACT of door de huidige taalaansturing.

Voorbeeld

In het volgende voorbeeld wordt LIKE( ) gebruikt om te bepalen welke twee tekenreeksen overeenkomen:

```
? LIKE("abc","abc")                 && Geeft .T. als resulaat
? LIKE("abc","Abc")                 && Geeft .F. als resulaat
? LIKE("a?c","abc")                 && Geeft .T. als resulaat
? LIKE("a*c","abc")                 && Geeft .T. als resulaat
? LIKE("a*","abc")                  && Geeft .T. als resulaat
? LIKE("*bc","abc")                 && Geeft .T. als resulaat
? LIKE("?abc","abc")                && Geeft .F. als resulaat
```

In het volgende voorbeeld wordt LIKE( ) gebruikt om alleen die records te selecteren die in het veld Bedrijf het woord "COMPUTER" bevatten:

```
USE Afnemers
LIST FIELDS Bedrijf, Contact ;
  FOR LIKE("*COMPUTER",UPPER(Bedrijf))
CLOSE DATABASES
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. Het argument *<memoveld>* wordt niet ondersteund in dBASE IV.

Zie ook

AT( ), DIFFERENCE( ), LDRIVER( ), LOWER( ), RAT( ), SET EXACT, SUBSTR( ), UPPER( )

LINENO( )

Geeft het nummer als resultaat van de huidige programmaregel in het huidige programma, de huidige procedure of de huidige door de gebruiker gedefinieerde functie.

Syntaxis

LINENO( )

Beschrijving

Met LINENO( ) kunt u de uitvoering van een programma volgen. Gebruik deze functie samen met PROGRAM( ) om te bepalen wanneer een programma een bepaalde coderegel uitvoert. U kunt LINENO( ) combineren met ON ERROR om vast te stellen welke regel een fout veroorzaakt.

De functie LINENO( ) heeft alleen zin in een programma, procedure of door de gebruiker gedefinieerde functie. Als u in het commandovenster LINENO( ) gebruikt, krijgt u 0 als resultaat.

LINENO( ) geeft altijd het werkelijke regelnummer in het programma als resultaat. De volgorde waarin de regels worden uitgevoerd is niet van invloed op het resultaat van de functie.

Voorbeeld

Zie ON ERROR voor een voorbeeld met LINENO( ).

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ERROR( ), MESSAGE( ), PROGRAM( ), RESUME, SUSPEND

LIST

Tabelindeling

Syntaxis

Tabelindeling

LIST
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[[FIELDS] <*uitdrukkingenlijst*>]
[OFF]
[TO FILE <*bestandsnaam*> | ? | <*bestandsnaamfilter*>] | [TO PRINTER]

Foutafhandeling en testen op fouten

LIST COVERAGE
[<*.COV-bestandnaam*> | ? | <*bestandsnaamfilterr*>]
[ALL]
[SUMMARY]
[TO FILE <*bestandsnaam*> | ? | <*bestandsnaamfilter*>] | [TO PRINTER]

Stations- en bestandsbeheer

LIST FILES
[LIKE <*bestandsnaam1*> | <*bestandsnaamfilter*>]
[ON <*station*>]
[TO FILE <*bestandsnaam*> | ? | <*bestandsnaamfilter*>] | [TO PRINTER]

Omgeving

LIST MEMORY
[TO FILE <*bestandsnaam*> | ? | <*bestandsnaamfilter*>] | [TO PRINTER]

Omgeving

LIST STATUS
[TO FILE <*bestandsnaam*> | ? | <*bestandsnaamfilter*>] | [TO PRINTER]

Tabelfuncties

LIST STRUCTURE
[IN <*alias*>]
[TO FILE <*bestandsnaam*> | ? | <*bestandsnaamfilter*>] | [TO PRINTER]

Beschrijving

Alle LIST-commando's komen overeen met DISPLAY-commando's die dezelfde informatie tonen. Zowel LIST als DISPLAY sturen hun uitvoer naar het resultatenpaneel van het commandovenster. Het enige verschil tussen beide commando's is dat LIST alle uitvoer in een keer weergeeft, terwijl DISPLAY na elk venster met informatie pauzeert. Bij DISPLAY kunt u lezen hoe u door de uitvoer van beide commando's in het resultatenpaneel kunt manoeuvreren. Zie de overeenkomstige DISPLAY-commando's voor een beschrijving van de informatie die de verschillende versies van de DISPLAY- en LIST-commando's leveren.

Als de informatie naar het resultatenpaneel meer is dan in de buffer van dBASE voor Windows past, kunt u misschien niet terug bladeren naar het begin van de informatie. Gebruik de opties TO FILE of TO PRINTER om de informatie op te slaan in een bestand of af te drukken op de printer.

Voorbeeld

In het volgende voorbeeld wordt LIST gebruikt om bepaalde gegevens uit de tabel Afnemers te tonen:

```
USE Afnemers EXCLUSIVE
LIST ALL
* Alle velden in alle records tonen

GO TOP       && Bij eerste record beginnen
LIST NEXT 10 FIELDS Bedrijf, Contact OFF
* Alleen velden Bedrijf en Contact in de
* volgende 10 records tonen
* OFF schakelt recordnummer uit

INDEX ON BEDRIJF TAG BEDRIJF
* Index maken op Bedrijf
SEEK "C"     && Eerste bedrijf dat begint met "C" zoeken
LIST BEDRIJF WHILE BEDRIJF="C" OFF
* Alle namen van bedrijven die beginnen met "C"
```

Zie ook

DISPLAY, DISPLAY COVERAGE, DISPLAY FILES, DISPLAY MEMORY, DISPLAY STATUS, DISPLAY STRUCTURE

LISTCOUNT( )

Objecten

Geeft het aantal keuzemogelijkheden in een keuzelijst als resultaat.

Syntaxis

LISTCOUNT(<*formulierverwijzing*>.<*keuzelijstverwijzing*>)

<*formulierverwijzing*>.<*keuzelijstverwijzing*>

<*formulierverwijzing*> is een objectverwijzingsvariabele die wijst naar het formulier waarin de keuzelijst is geplaatst. <*keuzelijstverwijzing*> is een objectverwijzingsvariabele die wijst naar het keuzevak dat u evalueert.

U kunt <*formulierverwijzing*> en <*keuzelijstverwijzing*> maken met het commando DEFINE:

```
* objectverwijzingsvariabele, MijnForm, maken.
DEFINE FORM MijnForm
* objectverwijzingsvariabele, xKeuze, maken.
DEFINE LISTBOX xKeuze OF MijnForm
```

U kunt <*formulierverwijzing*> en <*keuzelijstverwijzing*> ook maken met de operator NEW:

```
MijnForm = NEW FORM()
xKeuze = NEW LISTBOX("MijnForm")
```

Beschrijving

Gebruik LISTCOUNT( ) als u niet tevoren weet hoeveel mogelijkheden een keuzelijst bevat tijdens de uitvoering van het programma. Als u bijvoorbeeld "FILE *.*" opgeeft voor het kenmerk DataSource, is het aantal keuzemogelijkheden afhankelijk van het aantal bestanden in de standaarddirectory.

U kunt LISTCOUNT( ) ook gebruiken om lussen te besturen waarmee de keuzen van de gebruiker in een meerkeuzelijst worden geëvalueerd. U kunt dan bijvoorbeeld bepalen welke mogelijkheden zijn gekozen door in een DO...WHILE-lus elke keuzemogelijkheid te evalueren met de functie LISTSELECTED( ).

U maakt een meerkeuzelijst van een keuzelijst door voor het kenmerk Multiple waar (.T.) op te geven.

Voorbeeld

In het volgende voorbeeld wordt een formulier gedefinieerd met een keuzelijst waarin namen uit de tabel DIEREN.DBF worden getoond. Omdat voor het kenmerk Multiple .T. is ingesteld, kan de gebruiker meer dan een naam selecteren in het lijstvak. Als de gebruiker op het formulier rechtsdubbelklikt, roept het kenmerk OnRightMouseDown de procedure Geselecteerd aan. In deze procedure worden LISTCOUNT( ) en LISTSELECTED( ) gebruikt om de selecties naar het resultatenpaneel van het commandovenster te sturen:

```
USE Dieren.DBF
SET PROCEDURE TO PROGRAM(1) ADDITIVE
f = NEW TempForm()
f.Open()

CLASS TempForm OF FORM
   this.Left =         58.60
   this.Height =         10.12
   this.Width =         41.00
   this.Text = "Dierenwereld"
   this.OnRightMouseDown = CLASS::Geselecteerd
   this.Top =          9.35

   DEFINE LISTBOX KeuzeLijst1 OF THIS;
       PROPERTY;
         Left          9.00,;
         Height          4.00,;
         ColorNormal "N/W*",;
         Width         20.00,;
         DataSource "FIELD DIEREN->NAAM",;
         ColorHighLight "W+/B",;
         Multiple .T.,;
         Top          3.00,;
         Name "KeuzeLijst1"

   PROCEDURE geselecteerd
   FOR i=1 TO LISTCOUNT(Form.KeuzeLijst1)
       ? LISTSELECTED(Form.KeuzeLijst1,i)
   NEXT i
   RETURN

ENDCLASS
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DO...WHILE, FOR...NEXT, LISTSELECTED( ), Multiple

LISTSELECTED( ) Objecten

Geeft een keuzemogelijkheid uit een keuzelijst als resultaat.

Syntaxis

LISTSELECTED(<*formulierverwijzing*>.<*keuzelijstverwijzing*> [, *expN*])

<*formulierverwijzing*>.<*keuzelijstverwijzing*>

<*formulierverwijzing*> is een objectverwijzingsvariabele die wijst naar het formulier waarin de keuzelijst is geplaatst. <*keuzelijstverwijzing*> is een objectverwijzingsvariabele die wijst naar het keuzevak dat u evalueert.

U kunt <*formulierverwijzing*> en <*keuzelijstverwijzing*> maken met het commando DEFINE:

```
* objectverwijzingsvariabele, MijnForm, maken.
DEFINE FORM MijnForm
* objectverwijzingsvariabele, xKeuze, maken.
DEFINE LISTBOX xKeuze OF MijnForm
```

U kunt <*formulierverwijzing*> en <*keuzelijstverwijzing*> ook maken met de operator NEW:

```
MijnForm = NEW FORM()
xKeuze = NEW LISTBOX("MijnForm")
```

<*Nuitdr*>

Het nummer van de keuzemogelijkheid om te evalueren. Als u <*Nuitdr*> gebruikt, geeft LISTSELECTED( ) alleen de tekst van de opgegeven keuzemogelijkheid als resultaat als die is geselecteerd. Als u <*Nuitdr*> weglaat, geeft LISTSELECTED( ) de geselecteerde keuzemogelijkheid als resultaat in het geval van een gewone keuzelijst of de laatste geselecteerde keuzemogelijkheid in het geval van een meerkeuzelijst.

Beschrijving

Met LISTSELECTED( ) kunt u bepalen welke keuzemogelijkheid of keuzemogelijkheden in een keuzelijst zijn geselecteerd.

Gebruik LISTSELECTED( ) samen met LISTCOUNT( ) om de keuzen van de gebruiker in een meerkeuzelijst te evalueren. U kunt dan bijvoorbeeld bepalen welke mogelijkheden zijn gekozen door in een DO...WHILE-lus elke keuzemogelijkheid te evalueren met de functie LISTSELECTED( ). (Als de keuzelijst een onbepaald aantal keuzemogelijkheden bevat, kunt u LISTCOUNT( ) gebruiken om te bepalen hoeveel keer de lus moet worden uitgevoerd.)

U maakt een meerkeuzelijst van een keuzelijst door voor het kenmerk Multiple waar (.T.) op te geven.

Voorbeeld

In het volgende voorbeeld wordt een formulier gedefinieerd met een keuzelijst waarin namen uit de tabel DIEREN.DBF worden getoond. Omdat voor het kenmerk Multiple .T. is ingesteld, kan de gebruiker meer dan een naam selecteren in het lijstvak. Als de gebruiker op het formulier rechtsdubbelklikt, roept het kenmerk OnRightMouseDown de procedure Geselecteerd aan. In deze procedure worden LISTCOUNT( ) en LISTSELECTED( ) gebruikt om de selecties naar het resultatenpaneel van het commandovenster te sturen:

```
USE Dieren.DBF
SET PROCEDURE TO PROGRAM(1) ADDITIVE
f = NEW TempForm()
f.Open()

CLASS TempForm OF FORM
   this.Left =         58.60
   this.Height =         10.12
   this.Width =         41.00
   this.Text = "Dierenwereld"
   this.OnRightMouseDown = CLASS::Geselecteerd
   this.Top =          9.35

   DEFINE LISTBOX KeuzeLijst1 OF THIS;
       PROPERTY;
         Left          9.00,;
         Height          4.00,;
         ColorNormal "N/W*",;
         Width         20.00,;
         DataSource "FIELD DIEREN->NAAM",;
         ColorHighLight "W+/B",;
         Multiple .T.,;
         Top          3.00,;
         Name "KeuzeLijst1"

   PROCEDURE geselecteerd
   FOR i=1 TO LISTCOUNT(Form.KeuzeLijst1)
       ? LISTSELECTED(Form.KeuzeLijst1,i)
   NEXT i
   RETURN

ENDCLASS
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DO...WHILE, FOR...NEXT, LISTCOUNT( )

LKSYS( )

Gedeelde gegevens

Geeft als resultaat informatie over een vergrendeld record of bestand.

Syntaxis

LKSYS(<*Nuitdr*>)

<*Nuitdr*>

Een cijfer dat aangeeft welke informatie LKSYS( ) als resultaat moet geven:

<Nuitdr>	Resultaat
0	Tijd waarop vergrendeling is aangebracht
1	Datum waarop vergrendeling is aangebracht
2	Aanmeldingsnaam van gebruiker die record of bestand heeft vergrendeld
3	Tijd van laatste wijziging of vergrendeling
4	Datum van laatste wijziging of vergrendeling
5	Aanmeldingsnaam van gebruiker die record of bestand als laatste heeft gewijzigd of vergrendeld

Beschrijving

LKSYS( ) geeft multi-user-informatie als resultaat die is opgeslagen in een _dbaselock-veld in de tabel. LKSYS( ) kan alleen informatie als resultaat geven als de huidige tabel een _dbaselock-veld bevat. Met CONVERT kunt u een _dbaselock-veld toevoegen aan een tabel. Als de huidige tabel geen a _dbaselock-veld bevat, geeft LKSYS( ) voor elke waarde van <*Nuitdr*> een lege tekenreeks als resultaat.

Als een record is vergrendeld, nadrukkelijk of automatisch, wordt in het _dbaselock-veld van het record de volgende informatie opgeslagen: de tijd, de datum en de aanmeldingsnaam van de gebruiker die de vergrendeling aanbracht. Als een bestand wordt vergrendeld wordt deze informatie opgeslagen in het _dbaselock-veld van het eerste record in de tabel.

Als u aan LKSYS( ) het argument 0, 1 of 2 doorgeeft, worden alleen een waarde als resultaat gegeven als de vergrendeling van een record of bestand niet is gelukt. Als een bestands- of recordvergrendeling op een geconverteerde tabel niet lukt, wordt de informatie door LKSYS( ) met het argument 0, 1 of 2 als resultaat wordt gegeven, vanuit het _dbaselock-veld van de tabel naar een buffer geschreven. Als u vervolgens 0, 1 of 2 doorgeeft aan LKSYS( ), wordt de informatie uit de buffer gelezen. De buffer wordt pas overschreven nadat u een volgende vergrendeling niet lukt. Vandaar dat de argumenten 0, 1 en 2 altijd de informatie als resultaat geven die actueel was op het moment dat de laatste vergrendeling niet lukte.

Voor de argumenten 3, 4 en 5 bij LKSYS( ) maakt het niet uit of het huidige record of bestand vergrendeld is of niet. Deze argumenten geven informatie over de laatste record- of bestandsvergrendeling die wel lukte. Als u een van deze argumenten doorgeeft aan LKSYS( ), wordt de informatie rechtstreeks uit _dbaselock-veld gehaald en niet uit een interne buffer.

Als u de aanmeldingsnaam van een gebruiker opvraagt met de argumenten 2 of 5 en het _dbaselock-veld is slechts 8 tekens lang, wordt een lege tekenreeks als resultaat gegeven. De eerste 8 tekens van een _dbaselock-veld bevatten informatie over aantal, tijd en datum van de laatste wijziging of vergrendeling, dus het veld moet meer dan 8 tekens lang zijn om ook informatie over de aanmeldingsnaam van de gebruiker te kunnen bevatten. De lengte van het veld stelt u in met CONVERT.

Opmerking LKSYS( ) geeft geen vergrendelingsinformatie over SQL-databases of Paradox-tabellen.

Voorbeeld

In het volgende voorbeeld wordt RLOCK( ) gebruikt om een record in de tabel Bedrijf te vergrendelen. Als de vergrendeling niet lukt, geeft LKSYS( ) informatie als resultaat over tijd en datum van de vergrendeling en over de netwerkgebruiker die het record heeft vergrendeld. Als de vergrendeling is gelukt, wordt naar een procedure voor gegevensinvoer gesprongen:

```
CLEAR
USE Bedrijf SHARED
GoTo 2
mNogmaals="J"
DO WHILE UPPER(mNogmaals)="J"
  IF .NOT. RLOCK()
    TijdVergr = LKSYS(0)
    DatumVergr = LKSYS(1)
    GebrVergr = LKSYS(2)
      ? "Vergrendelingsinfo: ",TijdVergr, DatumVergr, GebrVergr
      ?
      Wait "Nogmaals proberen? (J/N)" to mNogmaals
  ELSE
    ? "Vergrendeling aangebracht. Voer nu gegevens in."
    mNogmaals=""
  ENDIF
ENDDO
?
WAIT
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CHANGE( ), CONVERT, FLOCK( ), RLOCK( ), SET LOCK, UNLOCK

LOAD DLL

Programmeren in Windows

Initialiseert een DLL-bestand.

Syntaxis

LOAD DLL [*<pad>*] *<DLL-bestandsnaam>*

[*<pad>*] *<DLL-naam>*

De naam van het .DLL-bestand. *<pad>* is het directorypad naar het DLL-bestand waarin de externe functie is opgeslagen.

Beschrijving

Gebruik LOAD DLL om de hulpmiddelen in een DLL-bestand beschikbaar te maken in een applicatie.

U kunt LOAD DLL ook gebruiken om de aanwezigheid van een DLL-bestand te controleren. U kunt bijvoorbeeld het commando ON ERROR gebruiken om een routine voor het onderscheppen van fouten te starten als het commando LOAD DLL een opgegeven DLL-bestand niet kan vinden.

LOAD DLL zoekt de DLL-bestanden niet in het dBASE-pad, maar in de volgende directories:

1 De huidige standaarddirectory

2 De Windows-directory (de directory die WIN.COM bevat)

3 De Windows-systeemdirectory (de directory die GDI.EXE bevat)

4 De directory die DBASEWIN.EXE bevat

5 De directory's die in de DOS-omgevingsvariabele PATH worden genoemd

6 Netwerkdirectory's (indien aanwezig)

Een DLL-bestand is een gecompileerde bibliotheek met externe routines die geschreven zijn in een gewone programmeertaal, zoals C en Pascal. Een DLL-bestand kan elke willekeurige extensie hebben, maar de meeste hebben de extensie .DLL.

Als u met LOAD DLL een DLL-bestand initialiseert, krijgt dBASE toegang tot de hulpmiddelen in het bestand. Het bestand wordt echter pas in het geheugen geladen als het programma of een andere Windows-toepassing gebruik maakt van de hulpmiddelen.

Voordat u een functie in een DLL-bestand kunt aanroepen, moet u eerst een functieprototype maken met EXTERN. Vervolgens kunt u de bij EXTERN opgegeven naam gebruiken om de routine aan te roepen als elke andere dBASE-functie.

Met LOAD DLL kunt u VBX-stuurelementen laden en registreren.

Voorbeeld

In het volgende voorbeeld wordt LOAD DLL gebruikt om een afbeeldingshulpmiddel in een .DLL-bestand te initialiseren:

```
LOAD DLL MijnAfb.DLL
DEFINE FORM Afbeeldingen FROM 2,2 TO 20,40
DEFINE Image MijnAfb OF Afbeeldingen;
```

```
    PROPERTY;
    DataSource "Resource MijnAfb.DLL 1001",;
    Top 5, Left 5
OPEN FORM Afbeeldingen
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

EXTERN, RELEASE DLL

LOCAL

Geheugenvariabelen

Definieert geheugenvariabelen die alleen zichtbaar zijn binnen de subroutine waarin ze zijn gedefinieerd.

Syntaxis

LOCAL <*variabelenlijst*>

<variabelenlijst>

De lijst van geheugenvariabelen die u met LOCAL wilt definiëren.

Beschrijving

Met LOCAL kunt u een lijst van geheugenvariabelen declareren die alleen beschikbaar zijn in de subroutine waarin het commando is gebruikt. Er bestaan twee soorten lokale variabelen. De ene soort wordt gedefinieerd met LOCAL en de andere met PRIVATE. Lokale variabelen die met PRIVATE zijn gedefinieerd , zijn ook beschikbaar in routines op lagere niveaus dan de routine waarin ze zijn gedefinieerd, terwijl variabelen die met LOCAL zijn gedefinieerd, *alleen* beschikbaar zijn in de routine (het programma, de subroutine of de door de gebruiker gedefinieerde functie) waarin ze zijn gedefinieerd.

Een variabele die is gedefinieerd met LOCAL is in feite een andere variabele dan een variabele met dezelfde naam in een routine op een hoger niveau of in een onafhankelijke routine. Als een lokale variabele eenmaal een waarde heeft gekregen, geven DISPLAY MEMORY en LIST MEMORY een variabele met dezelfde naam in een routine op een hoger niveau aan als verborgen.

U moet een variabele eerst definiëren voordat u daar een waarde aan kunt toewijzen. Als u een variabele definieert met LOCAL, heeft dat niet onmiddellijk tot gevolg dat die variabele wordt gemaakt. Als de variabele eenmaal is gedefinieerd, kunt u die maken en initialiseren met STORE of =. Het is niet mogelijk array's te definiëren met LOCAL. Lokale variabelen worden uit het geheugen verwijderd op het moment dat de subroutine waarin ze zijn gedefinieerd, wordt verlaten.

Bij PUBLIC vindt u een tabel waarin het bereik en de beschikbaarheid van variabelen die zijn gedefinieerd met PUBLIC, PRIVATE, LOCAL en STATIC, worden vergeleken.

Voorbeeld

In het volgende voorbeeld wordt LOCAL gebruikt om de variabele Vandaag te definiëren als een geheugenvariabele die alleen beschikbaar is in de procedure Resultaten:

```
SET PROCEDURE TO PROGRAM(1) ADDITIVE
DEFINE FORM Senior FROM 0,0 TO 15,50
DEFINE TEXT T1 OF Senior AT 3,4;
PROPERTY Text "In dienst per (DD-MM-JJ):",;
  Width 35
DEFINE Entryfield IV1 OF Senior AT 3,38;
  PROPERTY Value {}, Picture "99-99-99",;
  Width 9
DEFINE EntryField IV2 OF Senior AT 5,38;
  PROPERTY OnGotFocus Resultaten,;
  Value " ", Width 9
DEFINE TEXT T2 OF Senior AT  5,4;
  PROPERTY TEXT "Aantal dienstjaren:",;
  Width 22
OPEN FORM Senior

PROCEDURE Resultaten
Local Vandaag
Vandaag = Date()
Form.IV2.Value = (Vandaag-Form.IV1.Value)/365
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CLEAR MEMORY, PRIVATE, PUBLIC, RELEASE, STATIC, STORE

LOCATE

Tabelindeling

Zoekt in een tabel naar het eerste record dat voldoet aan een opgegeven voorwaarde.

Syntaxis

LOCATE
  [<*bereik*>]
  [FOR <*voorwaarde1*>]
  [WHILE <*voorwaarde2*>]

<bereik>
Het aantal records dat moet worden gezocht. RECORD <*n*> geeft een enkel record aan door middel van zijn recordnummer. NEXT <*n*> geeft *n* records aan, te beginnen bij het huidige record. ALL specificeert alle records. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR *<voorwaarde1>*
WHILE *<voorwaarde2>*
Bepaalt welke records worden beïnvloed door LOCATE. FOR beperkt LOCATE tot records die voldoen aan <*voorwaarde1*>. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang <*voorwaarde2*> waar is.

Beschrijving

LOCATE zoekt sequentieel in een tabel en test elk record om te controleren of het voldoet aan de opgegeven voorwaarden. Als een overeenkomst wordt aangetroffen (FOUND( ) geeft .T. als resultaat), wordt de recordaanwijzer van de tabel bij dat record gezet. Met het commando CONTINUE kan de zoekbewerking worden voortgezet, zodat ook volgende records die aan de voorwaarden voldoen, kunnen worden gezocht.

Als geen records aan de opgegeven voorwaarden voldoen of als het eind van de tabel is bereikt, verschijnt de melding Einde van LOCATE-bereik (als SET TALK is ingeschakeld (ON)) en wordt de recordaanwijzer aan het eind van het bestand geplaatst (EOF( ) geeft .T. als resultaat en FOUND( ) geeft .F. als resultaat). Deze melding verschijnt ook als CONTINUE is gegeven en er wordt geen record meer aangetroffen dat aan de bij LOCATE opgegeven voorwaarde voldoet.

LOCATE ALL en LOCATE FOR <voorwaarde> beginnen met zoeken aan het begin van de tabel, onafhankelijk van de positie van de recordaanwijzer. LOCATE WHILE <voorwaarde>, LOCATE NEXT <n> en LOCATE REST beginnen echter bij het huidige record en niet bij het eerste record.

LOCATE vereist geen geïndexeerde tabel. Als echter een index in gebruik is, houdt LOCATE zich aan de indexvolgorde. LOCATE maakt gebruik van de regels die zijn ingesteld met SET EXACT om te bepalen of een record aan de opgegeven voorwaarden voldoet. Als SET EXACT is uitgeschakeld (OFF, de standaardinstelling), hoeven alleen de eerste tekens van de tekenreeks rechts van het gelijkteken overeen te komen met de tekenreeks links van het gelijkteken. Als SET EXACT is ingeschakeld (ON), moeten de tekenreeksen volkomen gelijk zijn om aan de voorwaarde te voldoen.

De ingestelde taalaansturing bepaalt in combinatie met de instelling voor SET EXACT hoe speciale tekens (tekens met accenten) worden verwerkt bij zoekbewerkingen. Zie Appendix C in *Programmeren* voor meer informatie over taalaansturingen.

De zoekcommando's LOCATE, SEEK en FIND zijn elk ontwikkeld voor bepaalde omstandigheden. LOCATE is het meest flexibel. Dit commando accepteert uitdrukkingen van elk willekeurig gegevenstype en zoekt in elk veld van een tabel. Voor grote tabellen is sequentieel zoeken met LOCATE echter vrij traag.

Gebruik FIND of SEEK voor meer snelheid. Beide voeren een geïndexeerde zoekbewerking uit. dat lijkt op het opzoeken van een onderwerp in de index van een boek waarna direct naar de betreffende pagina wordt gegaan. De informatie wordt bijna onmiddellijk gevonden. Nadat u met het commando INDEX een index voor een tabel hebt gemaakt, gebruiken FIND en SEEK deze index om snel een toepasselijke record te zoeken. SEEK is ets flexibeler dan FIND omdat SEEK niet alleen numerieke en tekeninvoer accepteert, maar ook uitdrukkingen.

Voorbeeld

In het volgende voorbeeld wordt LOCATE gebruikt om in de tabel Bedrijf het eerste bedrijf in de regio NW:

```
USE Bedrijf EXCLUSIVE
LOCATE FOR Regio="NW"
* locate is niet afhankelijk van de index
IF FOUND()
   DISPLAY FIELDS Bedrijf, Plaats, Regio
ELSE
   ? "Geen bedrijven in regio NW"
ENDIF
```

Zie ook

CONTINUE, FIND, FOUND( ), LOOKUP( ), SEEK, SEEK( ), SET EXACT

LOCK( )

Gedeelde gegevens

Vergrendeld het huidige record of een opgegeven lijst van records in de huidige tabel of een opgegeven aliastabel, en geeft .T. als resultaat als de vergrendeling slaagt.

Syntaxis

LOCK([<*recordlijst Tuitdr*>] | [<*bladwijzerlijst Tuitdr*>][,<*alias*>])

<recordlijst Tuitdr>

De door komma's gescheiden lijst van te vergrendelen recordnummers.

<bladwijzerlijst>

De lijst van bladwijzers (recordindicaties). Een bladwijzer wordt door BOOKMARK( ) als resultaat gegeven om een record aan te duiden in een andere dan een dBASE-tabel, zoals een Paradox-tabel, waarin geen recordnummers in de natuurlijke volgorde worden gebruikt. Plaats komma's tussen de bladwijzers.

<alias>

Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan. Als u geen <*alias*> opneemt, voert LOCK( ) de bewerking uit in de huidige tabel.

U hoeft geen recordnummers of bladwijzers op te geven als u een waarde wilt opgeven voor <*alias*>. Als u echter recordnummers of bladwijzers hebt opgegeven, moet u voor <*alias*> een komma (,) plaatsen.

Beschrijving

LOCK( ) is onderling uitwisselbaar met RLOCK( ). Zie RLOCK( ) voor meer informatie.

Voorbeeld

Zie het voorbeeld met RLOCK( ). Vervang RLOCK( ) overal door LOCK( ).

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. De optie <*bladwijzerlijst*> wordt niet ondersteund in dBASE IV.

Zie ook

FLOCK( ), RLOCK( ), SET LOCK, SET RELATION, SET REPROCESS, UNLOCK

LOG( ) — Numerieke gegevens

Geeft de natuurlijk logaritme van een opgegeven getal als resultaat. De natuurlijke logaritme gebruikt *e* als grondtal.

Syntaxis

LOG(<*Nuitdr*>)

<Nuitdr>

Een positief getal ongelijk aan nul. Als u 0 of een negatief getal opgeeft voor <*Nuitdr*>, wordt een fout als resultaat gegeven.

Beschrijving

LOG( ) geeft de natuurlijke logaritme van <*Nuitdr*> als resultaat in de vorm van een zwevend getal. De natuurlijke logaritme is de macht (exponent) waartoe de rekenkundige constante *e* moet worden verheven om <*Nuitdr*> te krijgen. LOG(5) geeft bijvoorbeeld 1,61 als resultaat want *e*^1,61=5.

LOG( ) is de tegengestelde functie van EXP( ). Als LOG(Y) bijvoorbeeld gelijk is aan X, dan is Y gelijk aan EXP(X).

Het aantal decimalen stelt u in met SET DECIMALS.

Voorbeeld

In het volgende voorbeeld wordt LOG( ) gebruikt om het grondtal als resultaat te geven en EXP( ) om de derde-machts-wortels van een verzameling doorgegeven waarde te bepalen:

```
CLEAR
SET DECIMALS TO 2
? "Waarden" AT 7,"Log" AT 20, ;
    "Exponenten van derde-machts-" AT 29
? "wortel van Log-waarden" AT 29
FOR Waarde = 45 TO 270 STEP 15
? Waarde AT 3, LOG(Waarde) AT 12, ;
    EXP(LOG(Waarde)/3) AT 28
NEXT
```

Zie ook

EXP( ), LOG10( ), SET DECIMALS

LOG10( )

Numerieke gegevens

Geeft de logaritme met grondtal 10 van een opgegeven getal als resultaat.

Syntaxis

LOG10(<*Nuitdr*>)

<*Nuitdr*>

Een positief getal ongelijk aan nul. Als u 0 of een negatief getal opgeeft voor <*Nuitdr*>, wordt een fout als resultaat gegeven.

Beschrijving

LOG10( ) geeft de gewone logaritme van <*Nuitdr*> als resultaat in de vorm van een zwevend getal. De gewone logaritme is de macht (exponent) waartoe 10 moet worden verheven om <*Nuitdr*> te krijgen. LOG10(100) geeft bijvoorbeeld 2 als resultaat want 10^2=100.

Het aantal decimalen stelt u in met SET DECIMALS.

Voorbeeld

In het volgende voorbeeld wordt LOG10( ) gebruikt om exponenten voor het grondtal 10 als resultaat te geven:

```
SET DECIMAL TO 2
?
? "Waarden" AT 7, "Grondtal 10" AT 19,;
  "Exponenten van waarden met grondtal 10" AT 32
FOR waarde = 50 TO 200 STEP 10
   @ ROW() + 1, 3 SAY waarde
```

```
    @ ROW(),13 SAY LOG10(waarde)
    @ ROW(),26 SAY LOG10(EXP(waarde))
NEXT
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

EXP( ), LOG( ), SET DECIMALS

LOOKUP( )

Tabelindeling

Zoekt in een veld naar een opgegeven uitdrukking en geeft de waarde in een veld in hetzelfde record als resultaat als de gezochte waarde is gevonden.

Syntaxis

LOOKUP(<*resultaatveld*>, <*uitdr*>, <*zoekveld*>)

<resultaatveld1>

Het veld in de huidige of opgegeven tabel waarvan u de waarde als resultaat wilt geven als een overeenkomst is gevonden.

<uitdr>

De uitdrukking die moet worden gezocht in <*zoekveld*>. Geef een alias op als u verwijst naar velden buiten het huidige werkgebied.

<zoekveld1>

Het veld in de huidige of een opgegeven tabel waarvan de waarde overeen moet komen met <*uitdr*>. Als <*resultaatveld*> een memoveld is, moet <*uitdr*> een tekenuitdrukking zijn.

Beschrijving

De functie LOOKUP( ) zoekt naar waarden in <*zoekveld*> die overeenkomen met de opgegeven uitdrukking <*uitdr*>. Als een overeenkomst wordt aangetroffen, wordt de recordaanwijzer bij het betreffende record geplaatst en wordt de waarde van <*resultaatveld1*> als resultaat gegeven.

Als geen overeenkomst wordt aangetroffen, geeft LOOKUP( ) afhankelijk van het gegevenstype van <*zoekveld*> een lege tekenreeks (""), 0, een lege datum of .F. als resultaat. Tevens wordt de recordaanwijzer aan het eind van het bestand geplaatst (EOF( ) geeft .T. als resultaat) en geeft FOUND( ) .F. als resultaat.

LOOKUP( ) voert een sequentiële zoekbewerking uit, tenzij een index waarvan de sleutel overeenkomt met *<resultaatveld>*, is geopend als hoofdindex. Om de tijd die LOOKUP( ) nodig heeft te beperken, kunt u een hoofdindex opgeven.

Als u LOOKUP( ) gebruikt na een instructie met SET RELATION te hebben uitgevoerd, kunnen *<uitdr>* en *<zoekveld>* in de hoofdtabel en *<resultaatveld>* in een subtabel staan.

Voorbeeld

In het volgende voorbeeld wordt LOOKUP( ) gebruikt om het eerste bedrijf in Rotterdam te zoeken:

```
USE Bedrijf
Antwoord=LOOKUP(Bedrijf,"Rotterdam",Plaats)
* Antwoord bevat nu de naam van een bedrijf
* Rotterdam of is leeg
IF .NOT. EMPTY(Antwoord)
  ? "In Rotterdam:"+ Antwoord
ELSE
  ? "Geen bedrijven in Rotterdam"
ENDIF
```

In het volgende voorbeeld wordt voor elk record in de tabel Bedrijf LOOKUP( ) gebruikt om de naam van een contactpersoon voor dat bedrijf te zoeken in de tabel Contact:

```
CLOSE DATA
USE CONTACT
SELECT 2
USE Bedrijf EXCLUSIVE
SCAN
  ? Bedrijf,LOOKUP(Contact->Contact,;
                   CompCode,Contact->CompCode)
ENDSCAN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

EOF( ), FOUND( ), LOCATE, SEEK, SEEK( ), SET EXACT, SET INDEX, SET ORDER

LOWER( ) — Tekenreeksgegevens

Zet alle hoofdletters in een tekenreeks om in hoofdletters en geeft de geconverteerde tekenreeks als resultaat.

Syntaxis

LOWER(*<Tuitdr>* | *<memoveld>*)

<Tuitdr> | <memoveld>
De tekenreeks of het memoveld om naar kleine letters te converteren.

Beschrijving

LOWER( ) zet alle hoofdletters in een tekenuitdrukking of memoveld om in kleine letters. Cijfers en andere tekens worden genegeerd. De functie geeft maximaal 32.766 tekens als resultaat (de maximumlengte van een tekenreeks).

De huidige taalaansturing bepaalt welke tekens worden beschouwd als hoofdletters en kleine letters. Zie Appendix C in *Programmeren* voor meer informatie over de verwerking van hoofd- en kleine letters in de Nederlandse taalaansturing.

Voorbeeld

In het volgende voorbeeld wordt LOWER( ) gebruikt mom tekst in hoofdletters om te zetten naar kleine letters:

```
? LOWER("Technisch")          && Geeft "technisch" als resultaat
? LOWER("")                   && Geeft "" als resultaat
? LOWER("12 APPELS")          && Geeft "12 appels" als resultaat
```

In een veld met namen in de opmaak VOORNAAM (spatie) ACHTERNAAM in hoofdletters, kunnen UPPER( ) en LOWER( ) worden gebruikt om de naamreeks te wijzigen in hoofdletters en kleine letters:

```
USE Contact
REPLACE ALL Contact WITH UPPER(SUBSTR(Contact,1,1))+;
  LOWER(SUBSTR(Contact,2,AT(" ",Contact)-2))+ " " +;
  UPPER(SUBSTR(Contact,AT(" ",Contact)+1,1))+;
  LOWER(SUBSTR(Contact,AT(" ",Contact)+2,16))
```

Hoewel in het vorige voorbeeld de tekenreeksen worden gemanipuleerd met UPPER( ), LOWER( ) en SUBSTR( ), kan dezelfde taak ook worden uitgevoerd met:

```
REPLACE ALL Contact with PROPER(Contact)
```

Overdraagbaarheid

Het argument <*memoveld*> wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CHARSET( ), ISALPHA( ), ISLOWER( ), ISUPPER( ), LDRIVER( ), PROPER( ), UPPER( )

LTRIM( ) — Tekenreeksgegevens

Geeft een tekenreeks zonder voorloopspaties als resultaat.

Syntaxis

LTRIM(<*Tuitdr*> | <*memoveld*>)

<Tuitdr> | <memoveld>
De tekenreeks of het memoveld waaruit de voorloopspaties moeten worden verwijderd.

Beschrijving

LTRIM( ) geeft een tekenuitdrukking of memoveld zonder spaties aan het begin als resultaat. LTRIM( ) geeft maximaal 32.766 tekens als resultaat, de maximumlengte van een tekenreeks.

Als u LTRIM( ) gebruikt voor een memoveld worden alleen de voorloopspaties aan het begin van de eerste regel verwijderd. Gebruik LTRIM( ) samen met MLINE( ) om voorloopspaties op een bepaalde regel te verwijderen.

Volgspaties verwijdert u uit een tekenreeks of memoveld met RTRIM( ) of TRIM( ).

Voorbeeld

In het volgende voorbeeld wordt LTRIM( ) gebruikt om de voorloopspaties uit tekst in een tekenveld te verwijderen:

```
? STR(11.95,8,2)            && Geeft 11,95 als resultaat
? LTRIM(STR(11.95,8,2))     && Geeft 11,95 als resultaat
```

Numerieke gegevens worden standaard rechts uitgelijnd in een tabel en tekengegevens links. Als numerieke gegevens worden opgeslagen in een geheugenvariabele, is die variabele standaard 10 posities lang. In het volgende voorbeeld worden LTRIM( ) en STR( ) gebruikt om numerieke gegevens links uit te lijnen:

```
USE Afnemers
96                && Recordaanwijzer bij;
                     recordnummer 96 plaatsen
X=OpenBalans      && Inhoud van OpenBalans opslaan in X
? X               && Geeft 56,36 als resultaat
Y=LTRIM(STR(X,13,2))
? Y               && Geeft 56,36 als resultaat
```

Overdraagbaarheid

Het argument *<memoveld>* wordt niet ondersteund in dBASE IV of dBASE III PLUS. Verder beperken dBASE IV en dBASE III PLUS de resultaatwaarde van LTRIM( ) tot 254 tekens.

Zie ook

LEFT( ), MLINE( ), RIGHT( ), RTRIM( ), STR( ), SUBSTR( ), TRIM( )

LUPDATE( )

Velden en records

Geeft de datum van de laatste wijziging aan de huidige of een opgegeven tabel als resultaat.

Syntaxis

LUPDATE([<*alias*>])

<alias>

Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

LUPDATE( ) toont de datum van de laatste wijziging aan de opgegeven tabel. Als geen tabel is geopend, geeft LUPDATE( ) een lege datum als resultaat. LUPDATE( ) geeft de datum als resultaat in de opmaak dd-mm-jj. Deze opmaak kan worden gewijzigd met SET CENTURY, SET DATE en SET MARK.

Voorbeeld

In het volgende voorbeeld wordt LUPDATE( ) gebruikt om de datum van de laatste wijziging aan een opgegeven tabel als resultaat te geven:

```
USE Bedrijf IN SELECT()
USE Contact IN SELECT()
? "De tabel Bedrijf is het laatst gewijzigd op " + ;
   DTOC(LUPDATE("Bedrijf"))
? "De tabel Contact is het laatst gewijzigd op " + ;
   DTOC(LUPDATE("Contact"))
CLOSE ALL
```

Zie ook

DTOC( ), SET CENTURY, SET DATE

MAX( )

Uitdrukkingen en gegevenstypeconversie

Vergelijkt twee uitdrukkingen van hetzelfde type en geeft de grootste van de twee als resultaat. Als twee logische uitdrukkingen worden vergeleken, geeft de functie .T. als resultaat als een van beide of beide uitdrukkingen waar zijn.

Syntaxis

MAX(<*uitdr1*>, <*uitdr2*>)

<uitdr1>

Een uitdrukking (teken, datum, logisch, zwevend of numeriek) die wordt vergeleken met een tweede uitdrukking van hetzelfde type. U kunt zwevende getallen vergelijken met getallen van het type numeriek.

<uitdr2>
Een uitdrukking van hetzelfde gegevenstype als *<uitdr1>* om te vergelijken met *<uitdr1>*. U kunt zwevende getallen vergelijken met getallen van het type numeriek.

Beschrijving

Met MAX( ) kunt u bepalen welke van twee getallen, twee tekenreeksen, twee datums of twee logische waarden de grootste is. Afhankelijk van de argumenten geeft MAX( ) het volgende als resultaat:

- De grootste van twee getallen
- De tekenreeks met de hoogste *sorteerwaarde* van de twee reeksen. Sorteerwaarden worden bepaald door de huidige taalaansturing. Zie Appendix C in *Programmeren* voor meer informatie over taalaansturing.
- De laatste van de twee datums
- Waar (.T.) als een va beide of beide logische uitdrukkingen waar zijn

Als *<uitdr1>* en *<uitdr2>* gelijk zijn, geeft MAX( ) hun waarde als resultaat. Als u bijvoorbeeld STORE 100 TO gvar1 en STORE 50+50 TO gvar2 opgeeft, geeft MAX(gvar1,gvar2) 100 als resultaat.

Als u MAX( ) gebruikt voor tekenreeksen, moet u eerst SET EXACT ON opgeven om te zorgen dat de uitkomst nauwkeurig is. Bij het vergelijken van tekenreeksen maakt MAX( ) onderscheid tussen hoofdletters en kleine letters.

Voorbeeld

In het volgende voorbeeld wordt MAX( ) gebruikt om een waarde uit het veld VerkTotNu te vergelijken met een geheugenvariabele waarin een berekend gemiddelde is opgeslagen. De hoogste van beide waarden wordt als resultaat gegeven:

```
CLEAR
SET TALK OFF
USE Bedrijf
AVERAGE  VerkTotNu FOR  VerkTotNu<>0 TO Gemiddeld
GO TOP
? CENTER("Filialen met een hoger dan gemiddelde verkoop")
?
? "Bedrijf", "Filiaalgem." AT 27, "Verkoop tot nu" AT 40, ;
  "Verschil" AT 58
? REPLICATE("*",7) AT 0, REPLICATE("*",11) AT 27, ;
  REPLICATE("*",14) AT 40, REPLICATE("*",8) AT 58
DO WHILE .NOT. EOF()
  IF MAX( VerkTotNu,Gemiddeld)=VerkTotnu
    ? Bedrijf, Gemiddeld AT 23,  VerkTotNu, ;
       VerkTotNu - Gemiddeld
  ENDIF
  SKIP
ENDDO
RETURN
CLOSE ALL
```

Overdraagbaarheid

In dBASE III PLUS ondersteunt MAX( ) alleen numerieke uitdrukkingen. In dBASE IV ondersteunt MAX( ) geen logische uitdrukkingen.

Zie ook

CALCULATE, IIF( ), MIN( ), LDRIVER( ), SET EXACT, SET LDCHECK

MCOL( )

Toetsenbord- en muisacties

Geeft de huidige kolompositie van de muisaanwijzer als resultaat. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows kenmerken als OnLeftMouseDown voor het besturen van muishandelingen in formulieren.

Zie Help voor meer informatie over de syntaxis van MCOL( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het maken van formulieren.

MD

Stations- en bestandsfuncties

Maakt een nieuwe DOS-directory.

Syntaxis

MD *<directory>*

<directory>
De directory die u wilt maken.

Beschrijving

MD en MKDIR zijn onderling uitwisselbare commando's. Zie de beschrijving van MKDIR voor meer informatie.

Voorbeeld

Zie MKDIR voor een voorbeeld met MD.

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CD, MKDIR, SET DIRECTORY

MDOWN( )

Toetsenbord- en muisacties

Geeft .T. als resultaat als de muisknop is ingedrukt en .F. als dat niet het geval is. Dit commando wordt niet gebruikt in actiegestuurde omgevingen. Gebruik in dBASE voor Windows kenmerken als OnLeftMouseDown voor het besturen van muishandelingen in formulieren.

Zie Help voor meer informatie over de syntaxis van MDOWN( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het maken van formulieren.

MDX( )

Tabelindeling

Geeft de naam van een geopend .MDX-bestand in het huidige of een opgegeven werkgebied als resultaat.

Syntaxis

MDX(<*indexlabel Nuitdr*> [, <*alias*>])

<indexlabel Nuitdr>
De positie van een indexlabel in het geopende .MDX-bestand waarvan u de naam wilt weten.

<alias>
Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

MDX( ) geeft de naam van een geopend .MDX-bestand in het huidige of een opgegeven werkgebied als resultaat dat een opgegeven indexlabel bevat. De positie van de indexlabel komt overeen met de positie van het .MDX-bestand in de indexlijst die is ingesteld met SET INDEX TO, SET ORDER TO of USE. Als SET FULLPATH is ingeschakeld (ON), geeft MDX( ) ook het station en de directory van het .MDX-bestand als resultaat.

Als u geen nummer in de indexvolgorde opgeeft, geeft MDX( ) de naam van het .MDX-bestand als resultaat dat de hoofdindexlabel bevat. Als u geen nummer in de indexvolgorde opgeeft en de hoofdindex is een .NDX-bestand, geeft MDX( ) een lege tekenreeks ("") als resultaat. MDX( ) geeft ook een lege tekenreeks als resultaat als er geen geopend .MDX-bestand is of als op de opgegeven positie geen indexlabel staat.

Voorbeeld

In het volgende voorbeeld wordt MDX( ) gebruikt om te bepalen welke .MDX-bestanden voor de huidige tabel zijn geopend:

```
USE Bedrijf EXCLUSIVE
? 1,MDX(1)
? 2,MDX(2)
? " Slechts één MDX-bestand geopend"
INDEX ON Plaats     TAG Plaats     OF Lokatie
? 1,MDX(1)
? 2,MDX(2)
? "Nu zijn er twee MDX-bestanden geopend"
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FOR( ), INDEX, NDX( ), SET FULLPATH, SET INDEX, SET ORDER, TAG( ), TAGCOUNT( ), TAGNO( ), USE

MDY( )

Datum- en tijdgegevens

Geeft voor een opgegeven datum een tekenreeks met de opmaak MAAND DD, JJ als resultaat.

Syntaxis

MDY(<*Duitdr*>)

<Duitdr>

De datumuitdrukking die u wilt omzetten naar een tekenreeks met de opmaak MAAND DD, JJ.

Beschrijving

MDY( ) zet een datum om in een tekenreeks met de opmaak MAAND DD, JJ of MAAND DD, JJJJ, waarbij MAAND de volledige naam van de maand voorstelt. Als SET CENTURY is uitgeschakeld (OFF, de standaardinstelling), geeft MDY( ) voor het jaar twee cijfers als resultaat. Als SET CENTURY is ingeschakeld (ON), bestaat het jaar in het resultaat uit vier cijfers. Voor de dag geeft MDY( ) altijd twee cijfers als resultaat.

Geef <*Duitdr*> op in de huidige datumopmaak die wordt bepaald door SET DATE, DBASEWIN.INI of de optie **Internationaal** in het **Configuratiescherm** van Windows (in die volgorde). Dat wil zeggen, de instellingen bij SET DATE hebben een hogere prioriteit dan die in DBASEWIN.INI, en de instellingen in DBASEWIN.INI hebben weer een hogere prioriteit dan de instellingen in het **Configuratiescherm** van Windows. Let er op dat <*Duitdr*> overeenkomt met de datumopmaak die wordt gebruikt op het moment dat het programma wordt uitgevoerd.

Als u een ongeldige datum doorgeeft aan MDY( ), wordt die eerst omgezet in een geldige datum en wordt de datum omgezet in een tekenreeks met de opmaak MAAND

DD, JJ of MAAND DD, JJJJ. Als u aan MDY( ) een lege of een andere dan een datumuitdrukking tussen accolades ({ }) doorgeeft, wordt "Onbekend 00, 00" of "Onbekend 00, 0000" als resultaat gegeven. Als u een andere dan een datumuitdrukking of een uitdrukking die niet tussen accolades staat, doorgeeft, geeft MDY( ) een fout als resultaat.

Voorbeeld

Zie DMY( ) voor een voorbeeld met MDY( ). Vervang DMY( ) door DMY( ).

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CDOW( ), CMONTH( ), DAY( ), DMY( ), DOW( ), MONTH( ), SET CENTURY, YEAR( )

MEMLINES( )

Velden en records

Geeft het aantal regels in een memoveld als resultaat.

Syntaxis

MEMLINES(<*memoveld*> [,<*regellengte Nuitdr*>])

<memoveld>

Het memoveld waarvan u het aantal regels wilt bepalen.

<regellengte Nuitdr>

Geeft aan welke regellengte moet worden gebruikt om het aantal regels in het memoveld te bepalen. <*Nuitdr*> kan elk getal van 8 tot 255 zijn. Als <*Nuitdr*> niet is opgegeven, wordt het aantal regels berekend op basis van de memobreedte die is ingesteld met het commando SET MEMOWIDTH.

Beschrijving

De functie MEMLINES( ) geeft het aantal regels in een memoveld als resultaat op basis van de memobreedte die wordt aangegeven door de lengteparameter. Als u geen regellengte opgeeft, behandelt MEMLINES( ) de tekst in het memoveld als een veld met regelovergang van 50 tekens breed (tenzij u een andere breedte hebt opgegeven met het commando SET MEMOWIDTH).

Als een woord niet meer op de rest van de regel past, neemt MEMLINES( ) dat woord op aan het begin van de volgende regel. Als het aantal tekens in een woord langer is dan de opgegeven veldlengte, wordt het woord aan het eind van de regel afgekapt en wordt de rest aan het begin van de volgende regel geplaatst.

Voorbeeld

In het volgende voorbeeld wordt MEMLINES( ) gebruikt om het aantal regels in een memoveld als resultaat te geven. De breedte van het memoveld wordt door de gebruiker opgegeven. Met behulp van een regelteller en MEMLINE( ) wordt bepaald of het volgende memo op de huidige pagina past:

```
SET TALK OFF
CLEAR
AantKols = 10
@ 6,12 SAY "Breedte van rapport in kolommen? " ;
  GET AantKols PICTURE "99" ;
   VALID AantKols > 9 .AND. AantKols < 68
READ
mbreedte = SET("MEMOWIDTH")
SET MEMOWIDTH TO AantKols
USE Bedrijf
uitvoer_naar = "S"
IF uitvoer_naar = "S"
   paglengte   = 15
ELSE
   paglengte   = 58
ENDIF
regelteller = 1
memoregel = 1
DO WHILE .NOT. EOF()
   CLEAR
   regelteller = 1
   DO WHILE regelteller + MEMLINES(Notities) <= ;
      paglengte .AND. .NOT. EOF()
      ? Bedrijf AT 2
      regelteller  = regelteller + 1
      memoregel = 1
      DO WHILE memoregel <= MEMLINES(Notities) ;
         .AND. MEMLINES(Notities) <> 0
         ? MLINE(Notities,memoregel)AT 12
         regelteller  = regelteller + 1
         memoregel = memoregel + 1
      ENDDO
      SKIP
   ENDDO
   WAIT "Druk op een toets"
ENDDO
CLOSE ALL
SET MEMOWIDTH TO mbreedte
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. Het argument voor regellengte wordt niet ondersteund in dBASE IV.

Zie ook

MLINE( ), SET MEMOWIDTH, STORE MEMO

MEMORY( )

Omgeving

Geeft het beschikbare aantal kilobytes in het RAM-geheugen als resultaat.

Syntaxis

MEMORY([<*Nuitdr*>])

<Nuitdr>

dBASE voor Windows negeert deze parameter; deze is alleen aanwezig vanwege de compatibiliteit met dBASE IV.

Beschrijving

Met MEMORY( ) kunt u bepalen hoeveel geheugenruimte op een bepaald moment in het systeem beschikbaar is. De waarde die MEMORY( ) als resultaat geeft, omvat ook de hoeveelheid ruimte in het wisselbestand van Windows.

U kunt de hoeveelheid beschikbaar geheugen bijvoorbeeld controleren voordat u vanuit een applicatie een commando gebruikt als RUN of RUN( ). U kunt dan bepalen of de aangeroepen applicatie correct kan worden geladen en uitgevoerd.

Voorbeeld

In het volgende voorbeeld wordt de gebruiker gewaarschuwd als minder dan een megabyte RAM-geheugen beschikbaar is:

```
IF MEMORY()>=1000
  ? "Er is meer dan een megabyte RAM-geheugen beschikbaar"
ELSE
 ?? "Pas op! Er is minder dan een megabyte RAM-geheugen beschikbaar:"
 ? MEMORY(),"Kb"
ENDIF
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. In dBASE IV kan <*Nuitdr*> een waarde van 0 tot en met 7 zijn. Deze waarden geven elk informatie over het geheugen in een DOS-omgeving als resultaat.

Zie ook

DIR, DISPLAY MEMORY, FLUSH, LIST, RUN, RUN( )

MENU( )

dBASE IV-menu's

Geeft de naam van de huidige dBASE IV-menubalk als resultaat in de vorm van een tekenreeks die geheel uit hoofdletters bestaat. Dit commando wordt hoofdzakelijk

ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows kunt u INSPECT( ) gebruiken om informatie als resultaat te geven over objecten in formulieren.

Zie Help voor meer informatie over de syntaxis van MENU( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

MESSAGE( )

Foutafhandeling en testen op fouten

Geeft de foutmelding voor de laatste dBASE-fout als resultaat.

Syntaxis

MESSAGE( )

Beschrijving

Gebruik MESSAGE( ) in combinatie met andere commando's en functies voor het onderscheppen van fouten zoals ON ERROR, RETRY en ERROR( ), om de standaardreactie van dBASE te vervangen door andere reacties.

Bij aanvang geeft MESSAGE( ) een lege tekenreeks als resultaat. Als een fout optreedt, geeft MESSAGE( ) een foutmelding als resultaat. MESSAGE( ) blijft deze foutmelding als resultaat geven tot een van de volgende gebeurtenissen plaatsvindt:

- Er treedt een andere fout op.
- RETRY wordt gebruikt.
- De subroutine waarin de fout optrad, wordt beëindigd.

Met DBMESSAGE( ) kunt u de IDAPI-foutmelding van de laatste IDAPI-fout in de huidige tabel als resultaat geven.

Raadpleeg de tabel bij ERROR( ). In die tabel worden CERROR( ), ERROR( ), MESSAGE( ), DBERROR( ), DBMESSAGE( ), SQLERROR( ) en SQLMESSAGE( ) met elkaar vergeleken.

Zie Help voor een lijst van alle dBASE-foutmeldingen.

Voorbeeld

Zie ON ERROR voor een voorbeeld met MESSAGE( ).

Overdraagbaarheid

De foutmeldingen voor sommige fouten in dBASE IV en dBASE III PLUS verschillen van de foutmeldingen voor dezelfde fouten in dBASE voor Windows. Zie Help voor meer informatie.

Zie ook

CERROR( ), DBERROR( ), DBMESSAGE( ), ERROR( ), ON ERROR, RETRY, SQLERROR( ), SQLMESSAGE( )

MIN( )

Uitdrukkingen en gegevenstypeconversie

Vergelijkt twee uitdrukkingen van hetzelfde type en geeft de kleinste van de twee als resultaat. Als twee logische uitdrukkingen worden vergeleken, geeft de functie als resultaat .F. als een van beide of beide uitdrukkingen onwaar zijn.

Syntaxis

MIN(<*uitdr1*>, <*uitdr2*>)

<*uitdr1*>

Een uitdrukking (teken, datum, logisch, zwevend of numeriek) die wordt vergeleken met een tweede uitdrukking van hetzelfde type. U kunt zwevende getallen vergelijken met getallen van het type numeriek.

<*uitdr2*>

Een uitdrukking van hetzelfde gegevenstype als <*uitdr1*> om te vergelijken met <*uitdr1*>. U kunt zwevende getallen vergelijken met getallen van het type numeriek.

Beschrijving

Met MIN( ) kunt u bepalen welke van twee getallen, twee tekenreeksen, twee datums of twee logische waarden de kleinste is. Afhankelijk van de argumenten geeft MIN( ) het volgende als resultaat:

- De kleinste van twee getallen
- De tekenreeks met de laagste *sorteerwaarde* van de twee reeksen. Sorteerwaarden worden bepaald door de huidige taalaansturing. Zie Appendix C in *Programmeren* voor meer informatie over taalaansturing.
- De eerste van de twee datums
- Onwaar (.F.) als een va beide of beide logische uitdrukkingen onwaar zijn

Als <*uitdr1*> en <*uitdr2*> gelijk zijn, geeft MIN( ) hun waarde als resultaat. Als u bijvoorbeeld STORE 100 TO gvar1 en STORE 50+50 TO gvar2 opgeeft, geeft MIN(gvar1,gvar2) 100 als resultaat.

Als u MIN( ) gebruikt voor tekenreeksen, moet u eerst SET EXACT ON opgeven om te zorgen dat de uitkomst nauwkeurig is. Bij het vergelijken van tekenreeksen maakt MIN( ) onderscheid tussen hoofdletters en kleine letters.

Voorbeeld

In het volgende voorbeeld wordt MIN( ) gebruikt om een waarde uit het veld VerkTotNu te vergelijken met een geheugenvariabele waarin een berekend gemiddelde is opgeslagen. De laagste van beide waarden wordt als resultaat gegeven:

```
CLEAR
SET TALK OFF
USE Bedrijf
AVERAGE  VerkTotNu FOR  VerkTotNu<>0 TO Gemiddeld
GO TOP
? CENTER("Filialen met een hoger dan gemiddelde verkoop")
?
? "Bedrijf", "Filiaalgem." AT 27, "Verkoop tot nu" AT 40, ;
  "Verschil" AT 58
? REPLICATE("*",7) AT 0, REPLICATE("*",11) AT 27, ;
  REPLICATE("*",14) AT 40, REPLICATE("*",8) AT 58
DO WHILE .NOT. EOF()
  IF MAX( VerkTotNu,Gemiddeld)=VerkTotnu
    ? Bedrijf, Gemiddeld AT 23,  VerkTotNu, ;
       VerkTotNu - Gemiddeld
  ENDIF
  SKIP
ENDDO
RETURN
CLOSE ALL
```

Overdraagbaarheid

In dBASE III PLUS ondersteunt MIN( ) alleen numerieke uitdrukkingen. In dBASE IV ondersteunt MIN( ) geen logische uitdrukkingen.

Zie ook

CALCULATE, IIF( ), MAX( ), SET EXACT

MKDIR

Stations- en bestandsfuncties

Maakt een nieuwe DOS-directory.

Syntaxis

MKDIR *<directory>*

<directory>

De directory die u wilt maken.

Beschrijving

Met MKDIR kunt u een nieuwe directory maken zonder dBASE te verlaten. MD en MKDIR zijn onderling uitwisselbare commando's.

MKDIR komt overeen met de DOS-opdrachten MKDIR en MD. Als *<directory>* uit een enkele naam bestaat, wordt de nieuwe directory een subdirectory van de huidige directory. Als *<directory>* een pad bevat, wordt de nieuwe directory aan het eind van het opgegeven pad gemaakt. Het pad moet al bestaan en moet eindigen op een backslash gevolgd door de naam van de nieuwe directory. Als *<directory>* begint met een backslash, begint het opgegeven pad in de hoofddirectory van de schijf.

Voor de nieuwe directorynaam mogen alleen tekens worden gebruikt die geldig zijn voor de namen van DOS-directory's.

Nadat u een nieuwe directory hebt gemaakt, kunt u die directory de standaarddirectory maken met CD of SET DIRECTORY. U kunt de nieuwe directory ook in het zoekpad van dBASE opnemen met SET PATH.

Voorbeeld

In de volgende voorbeelden wordt MKDIR of gebruikt om op station D: een subdirectory Project, drie subdirectory's onder Project en een subdirectory op station C: te maken:

```
MKDIR D:\Project
MKDIR D:\Project\Programs
MKDIR D:\Project\Gegevens
MD    D:\Project\Reserve
MD    C:\Editor
```

Als u probeert een directory te maken die al bestaat, of een pad opgeeft dat niet bestaat, verschijnt een foutmelding.

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CD, SET DIRECTORY, SET PATH

MLINE( )

Velden en records

Haalt een opgegeven tekstregel op uit een memoveld in het huidige record.

Syntaxis

MLINE(*<memoveld>* [, *<regelnummer Nuitdr>* [, *<regellengte Nuitdr>*]])

<memoveld>

Het memoveld waar de regel uit moet worden opgehaald.

<regelnummer Nuitdr>
Het nummer van de regel die als resultaat moet worden gegeven. De standaardinstelling voor *<regelnummer Nuitdr>* is 1.

<regellengte Nuitdr>
Dit getal bepaalt de lengte van een regel in het memoveld. *<regellengte Nuitdr>* kan elk getal van 8 tot 255 zijn. Als *<regellengte Nuitdr>* niet is opgegeven, Als *<Nuitdr>* niet is opgegeven, wordt het aantal regels bepaald op basis van de memobreedte die is ingesteld met het commando SET MEMOWIDTH. Als *<regelnummer Nuitdr>* kleiner is dan *<regellengte Nuitdr>* tekens, voegt MLINE( ) tekens toe van de regel die volgt op de door *<regelnummer Nuitdr>* aangegeven regel.

Beschrijving
MLINE( ) geeft een opgegeven tekstregel uit een memoveld als resultaat. MLINE( ) behandeld de tekst in het memoveld als tekst met woordomslag binnen breedte die is ingesteld met SET MEMOWIDTH setting of wordt aangegeven door *<regellengte Nuitdr>*. Als een woord niet geheel binnen de opgegeven regellengte past, wordt dat naar het begin van de volgende regel verplaatst en maakt het geen deel uit van de tekenreeks die door LINE( ) als resultaat wordt gegeven. De tekst op een regel wordt dus bepaald door de lengte van de regel en de woorden die geheel op de regel passen.

Voorbeeld
Zie MEMLINES( ) voor een voorbeeld met MLINE( ).

Overdraagbaarheid
Wordt niet ondersteund in dBASE III PLUS. Het argument voor de regellengte wordt niet ondersteund in dBASE IV.

Zie ook
MEMLINES( ), REPLACE MEMO, SET MEMOWIDTH, STORE MEMO

MOD( ) — Numerieke gegevens

Geeft de modulus (rest) van een quotiënt als resultaat.

Syntaxis
MOD(*<deeltal Nuitdr>*, *<deler Nuitdr>*)

<deeltal Nuitdr>
Het deeltal

<deler Nuitdr>
De deler

Beschrijving

MOD( ) deelt *<deeltal Nuitdr>* door *<deler Nuitdr>* en geeft de rest als resultaat in de vorm van een geheel getal. De instructie MOD(X,Y) geeft bijvoorbeeld de rest van x/y als resultaat.

MOD( ) geeft een positief getal als resultaat als *<deler Nuitdr>* positie is, en een negatief getal als *<deler Nuitdr>* negatief is.

Het resultaat van de functie wordt berekend aan de hand van de volgende formule:

```
<deeltal>-FLOOR(<deeltal>/<deler>)*<deler>
```

Hierbij geeft FLOOR( ) het grootste gehele getal dat kleiner dan of gelijk aan het argument is.

Voorbeeld

Zie INT( ) voor een voorbeeld van MOD( ).

Zie ook

CEILING( ), FLOOR( ), INT( )

MODIFY... Informatie

Wijzigt het overeenkomstige object of bestand.

Syntaxis

MODIFY APPLICATION [<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]	**Formulieren**
MODIFY COMMAND [<*bestandsnaam*> | ? | <*bestandsnaamfilter*>] [WINDOW <*vensternaam*>]	**Programma's**
MODIFY FILE [<*bestandsnaam*> | ? | <*bestandsnaamfilter*>] [WINDOW <*vensternaam*>]	**Stations- en bestandsfuncties**
MODIFY FORM [<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]	**Formulieren**
MODIFY LABEL [<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]	**Invoer/uitvoer**
MODIFY MENU [<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]	**Formulieren**
MODIFY QUERY [<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]	**Tabelindeling**
MODIFY REPORT [<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]	**Invoer/uitvoer**
MODIFY SCREEN [<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]	**Formulieren**
MODIFY VIEW [<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]	**Tabelindeling**

Beschrijving

De hier genoemde MODIFY-commando's werken op min of meer dezelfde manier als de overeenkomstige CREATE-commando's. Zie de overeenkomstige CREATE-commando's voor meer informatie.

MODIFY STRUCTURE — Tabellen

laat u en de structuur van de huidige tabel wijzigen. U kunt de structuur wijzigen van .DBF-bestanden en .DB-bestanden, maar niet van SQL-tabellen.

Syntaxis

MODIFY STRUCTURE

Beschrijving

Met MODIFY STRUCTURE kunt u de structuur van de huidige tabel wijzigen door velden toe te voegen en te verwijderen of door een veldnaam, veldlengte of gegevenstype te wijzigen. Als u het commando MODIFY STRUCTURE geeft, verschijnt Tabelontwerp, een interactieve omgeving waarin u de structuur van een tabel kunt maken of wijzigen. Raadpleeg het *Handboek* voor meer informatie over werken met Tabelontwerp.

Voordat u wijzigingen kunt aanbrengen aan de structuur van een dBASE-tabel, wordt een reservekopie van de oorspronkelijke tabel met de extensie .DBK gemaakt. Vervolgens wordt een nieuw tabelbestand met de extensie .DBF gemaakt en wordt de gewijzigde tabelstructuur naar dat bestand gekopieerd. Als u klaar bent met het wijzigen van de tabelstructuur, wordt de inhoud van het bestand met reservekopie naar de nieuwe structuur gekopieerd. Als er per ongeluk gegevens zijn ingekort of verloren gegaan, kunt u de oorspronkelijke gegevens herstellen vanuit het .DBK-bestand. Controleer voordat u de structuur van een tabel gaat wijzigen of op schijf voldoende ruimte beschikbaar is voor de reservekopie en eventueel tijdelijke opslagruimte voor het kopiëren van de records (ongeveer twee maal de grootte van de oorspronkelijke tabel).

Als een tabel een memoveld bevat, maakt MODIFY STRUCTURE ook een reservekopie van het memobestand met de oorspronkelijke memogegevens. Dit bestand heeft dezelfde naam als de tabel, maar krijgt de extensie .TBK.

Denk er aan niet tegelijk de naam en de lengte of het type van een veld te wijzigen. Als u dat doet, kunnen de gegevens uit het oude veld niet worden toegevoegd en blijft het nieuwe veld leeg. Verander de veldnaam, sla het bestand op en gebruik vervolgens opnieuw MODIFY STRUCTURE om de veldlengte of het type van het veld te wijzigen.

Verder moet u niet tegelijk veldnamen wijzigen en velden toevoegen aan of verwijderen uit een tabel. Als u veldnamen wijzigt, worden gegevens uit het oude bestand toegevoegd aan de hand van de positie van het veld binnen het bestand. Als u tegelijk velden invoegt of verwijdert en veldnamen wijzigt, kan dat tot gevolg hebben dat de posities van de velden veranderen en dat er gegevens verloren gaan. U kunt wel tegelijk de veldlengte of het type van een veld wijzigen en velden toevoegen of verwijderen. Omdat in dat geval de gegevens worden toegevoegd op basis van de veldnamen, zullen de gegevens dan correct worden toegevoegd.

Het is mogelijk gegevens naar verschillende veldtypen te converteren. Als u het type van een veld wijzigt, doet u er echter goed aan een reservekopie van het oorspronkelijke bestand achter de hand te houden, en in het nieuwe bestand te controleren of de conversie helemaal goed is gegaan.

Als u numerieke velden converteert naar tekenvelden, worden de nieuwe tekenreeksen rechts uitgelijnd. Als u een tekenveld converteert naar een numeriek veld, worden de numerieke tekens in elk record geconverteerd naar cijfers tot het eerste niet-numerieke teken wordt aangetroffen. Als het eerste teken in een tekenveld een niet-numeriek teken is, krijgt het geconverteerde numerieke veld de waarde nul.

Het is mogelijk logische velden te converteren naar tekenvelden en omgekeerd. Het is ook mogelijk tekenreeksen die zijn opgemaakt als datums (zoals dd-mm-jj of dd/mm/jjjj) te converteren naar een datumveld, of datumvelden te converteren naar tekenvelden. Het is niet mogelijk logische velden te converteren naar numerieke velden.

dBASE voor Windows zal elke gewenste conversie trachten uit te voeren, maar de conversie moet wel zinnig zijn, want anders kunnen gegevens verloren gaan. Numerieke gegevens kunnen bijvoorbeeld gemakkelijk worden omgezet naar tekens, maar logische gegevens kunnen bijvoorbeeld niet worden geconverteerd naar numerieke gegevens. Als incompatibele gegevenstypen (zoals logisch naar numeriek) wilt converteren, kunt u betere eerst een nieuw veld toevoegen aan het bestand en de gegevens converteren met REPLACE. Daarna kunt u het oude veld dan verwijderen.

Als u de naam, de lengte of het type wijzigt van een veld dat over een gekoppelde label in een produktie-MDX-bestand beschikt, wordt de label automatisch opnieuw gemaakt. Als indexen zijn geopend op het moment dat u een tabelstructuur wijzigt, worden de indexen automatisch gesloten als u de gewijzigde tabel opslaat. Nadat u de structuur van een tabel hebt gewijzigd, moet u de indexen opnieuw samenstellen.

Voorbeeld

In het volgende voorbeeld wordt het commandovenster MODIFY STRUCTURE gebruikt om de structuur van een tabel te wijzigen:

```
USE Klanten IN SELECT() EXCLUSIVE
MODIFY STRUCTURE
CLOSE DATABASES
```

Met de volgende commando's in het commandovenster is het mogelijk de structuur van een tabel te tonen zonder die te wijzigen:

```
USE KLANTEN IN SELECT() NOUPDATE
DISPLAY STRUCTURE
CLOSE DATABASES
```

Zie ook

APPEND, APPEND MEMO, COPY STRUCTURE, CREATE, DISPLAY STRUCTURE, LIST STRUCTURE, REPLACE

MONTH( ) — Datum- en tijdgegevens

Geeft het nummer van de maand uit een opgegeven datumuitdrukking als resultaat.

Syntaxis

MONTH(<*Duitdr*>)

<*Duitdr*>

De datumuitdrukking waarvan u het nummer van de maand als resultaat wilt geven.

Beschrijving

MONTH( ) geeft het nummer van een maand als resultaat: een waarde van 1 tot en met 12.

Geef <*Duitdr*> op in de huidige datumopmaak die wordt bepaald door SET DATE, DBASEWIN.INI of de optie **Internationaal** in het **Configuratiescherm** van Windows (in die volgorde). Dat wil zeggen, de instellingen bij SET DATE hebben een hogere prioriteit dan die in DBASEWIN.INI, en de instellingen in DBASEWIN.INI hebben weer een hogere prioriteit dan de instellingen in het **Configuratiescherm** van Windows. Let er op dat <*Duitdr*> overeenkomt met de datumopmaak die wordt gebruikt op het moment dat het programma wordt uitgevoerd.

Als u een ongeldige datum doorgeeft aan MONTH( ), wordt die eerst omgezet in een geldige datum en wordt vervolgens het nummer van de datum als resultaat gegeven. Als u aan MONTH( ) een lege of een andere dan een datumuitdrukking tussen accolades ({ }) doorgeeft, wordt 0 als resultaat gegeven. Als u een andere dan een datumuitdrukking of een uitdrukking die niet tussen accolades staat, doorgeeft, geeft MONTH( ) een fout als resultaat.

Voorbeeld

In het volgende voorbeeld wordt MONTH( ) gebruikt om het nummer van de maand in de systeemdatum te bepalen en dat nummer te combineren met een tekenreeks die het juiste achtervoegsel bevat:

```
IF SET("CENTURY") <> "OFF"
   SET CENTURY OFF
ENDIF
IF SET("DATE") <> "DMY"
   SET DATE DMY
ENDIF
maand = MONTH(DATE())
mnd_rks = LTRIM(STR(maand))
DO CASE
   CASE maand =  1 .OR. maand= 8
      mnd_rks = mnd_rks + "ste"
   CASE maand =  2 .OR. maand = 3 .OR. maand = 4 ;
      .OR. maand = 5 .OR. maand = 6 .OR. maand = 7 ;
      .OR. maand = 9 .OR. maand = 10 ;
      .OR. maand = 11 .OR. maand = 12
      mnd_rks = mnd_rks + "de"
ENDCASE
? "Dit is de " + mnd_rks + " maand van " + ;
   LTRIM(STR( YEAR(DATE()))) + " " + DTOC(DATE())
```

Zie ook

CMONTH( ), DAY( ), DATE( ), SET CENTURY, SET DATE

MOVE WINDOW

dBASE IV-vensters

Definieert de rij- en kolomcoördinaten voor een bestaand dBASE IV-venster opnieuw. De grootte, vorm en inhoud van het venster worden niet beïnvloed. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows het kenmerk Moveable van een formulier om te bepalen of de gebruiker het formulier kan verplaatsen.

Zie Help voor meer informatie over de syntaxis van MOVE WINDOW. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

MROW( ) Toetsenbord- en muisacties

Geeft de huidige rijpositie van de muisaanwijzer als resultaat. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows kenmerken als OnLeftMouseDown voor het besturen van muishandelingen in formulieren.

Zie Help voor meer informatie over de syntaxis van MROW( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het maken van formulieren.

MSGBOX( ) Formulieren

Opent een dialoogvenster met een melding en knoppen en geeft als resultaat een numerieke waarde die overeenkomt met de knop waarop de gebruiker klikt.

Syntaxis

MSGBOX(<*melding Tuitdr*>, [<*naam Tuitdr*>, [<*venstertype Nuitdr*>]])

<*melding Tuitdr*>

De melding die in het venster wordt getoond.

<*naam Tuitdr*>

De naam die in de titelbalk van het dialoogvenster staat.

<*venstertype Nuitdr*>

Een numerieke waarde die bepaalt welk pictogram en welke knoppen het venster bevat. Voor een dialoogvenster met knoppen zonder pictogram gebruikt u de volgende waarden:

<*venstertype Nuitdr*>	**Knoppen**
0	OK
1	OK, Annuleren
2	Afbreken, Opnieuw, Negeren
3	Ja, Nee, Annuleren
4	Ja, Nee
5	Opnieuw, Annuleren

Het volgende commando opent bijvoorbeeld een dialoogvenster zonder pictogram met de knoppen Ja, Nee en Annuleren:

```
Oordeel = MSGBOX("Klik hier", "Doei!", 3)
```

Voor een dialoogvenster met knoppen en een pictogram telt u een van de volgende waarden op bij *<venstertype Nuitdr>*:

Waarde	Pictogram
16	
32	
48	
64	

Het volgende commando opent bijvoorbeeld hetzelfde dialoogvenster, maar ditmaal met een blauwe i:

```
? MSGBOX("Klik hier", "Doei!", 67) && 3 + 64
```

Als een dialoogvenster meerdere knoppen heeft, heeft de linkerknop de focus. Als u echter 256 optelt bij *<venstertype Nuitdr>*, krijgt de tweede knop focus, en als u 512 erbij optelt, de derde. Het volgende commando opent bijvoorbeeld hetzelfde dialoogvenster als in het vorige voorbeeld, maar nu heeft de knop Annuleren de focus:

```
? MSGBOX("Klik hier", "Doei!", 579) && 3 + 64 + 512
```

Als u *<venstertype Nuitdr>* weglaat, is venstertype 0 standaard.

Beschrijving

Gebruik MSGBOX( ) om de gebruiker te dwingen een keuze te maken of een melding te bevestigen door op een knop in een modaal dialoogvenster te klikken.

Zolang het dialoogvenster open is, wordt de uitvoering onderbroken en kan de gebruiker de focus niet verplaatsen. Zodra de gebruiker op een knop klikt, verdwijnt het dialoogvenster, wordt de uitvoering voortgezet en geeft MSGBOX( ) als resultaat een numerieke waarde die aangeeft op welke knop is geklikt.

Knop	Resultaat
OK	1
Annuleren	2

Knop	Resultaat
Afbreken	3
Opnieuw	4
Negeren	5
Ja	6
Nee	7

U kunt het resultaat bijvoorbeeld gebruiken voor vertakkingen in een foutafhandelingsroutine. Telkens wanneer een fout optreedt, gebruikt de routine MSGBOX( ) om een dialoogvenster te openen met de knoppen Afbreken, Opnieuw en Annuleren. De routine evalueert vervolgens het resultaat van MSGBOX( ) en voert de noodzakelijke handeling uit.

Voorbeeld

Het volgende voorbeeld gebruikt een foutafhandelingsroutine om een dialoogvenster met de knoppen Afbreken, Opnieuw en Annuleren te openen zodra een fout optreedt. Klikt de gebruiker op Afbreken, dan wordt de uitvoering beëindigd. Klikt de gebruiker op Opnieuw, dan wordt de besturing terugggeven aan de hoofdroutine. Klikt de gebruiker op Negeren, dan wordt de uitvoering voortgezet na het commando dat niet kon worden uitgevoerd.

```
* De hoofdroutine
SET TALK OFF
FOR i = 1 to 5
   DO MaakFout
   ? "Probeer opnieuw" && Dit commando wordt alleen uitgevoerd
                       && als gebruiker op Opnieuw klikt.
NEXT i
RETURN

PROCEDURE MaakFout
  ON ERROR DO Herstel WITH;
     PROGRAM(), MESSAGE(), LINENO()

  ? COMPANY()       && Deze functie bestaat niet
                    && en veroorzaakt dus een fout.
  DIR               && Dit commando alleen uitgevoerd
                    && als gebruiker op Negeren klikt.
RETURN

PROCEDURE Herstel(PRO, MES, LIN)
  Oordeel = MSGBOX("Probleem op regel "+;
            LTRIM(STR(LIN))+;
            ", "+MES, "Routinenaam: "+PRO, 18)
  DO CASE
     CASE Oordeel = 3  && Afbreken (Uitvoering stoppen)
          CANCEL
     CASE Oordeel = 4  && Opnieuw (= proberen)
          RETURN TO MASTER
     CASE Oordeel = 5  && Negeren (Ga verder)
          RETURN
```

```
    ENDCASE
RETURN
```

Overdraagbaarheid

Word niet ondersteund dBASE IV of dBASE III PLUS.

Zie ook

ACCEPT, READMODAL( ), ReadModal( ), WAIT

NDX( )

Tabelindeling

Geeft de naam van een .NDX-bestand als resultaat.

Syntaxis

NDX([<*indexpositie Nuitdr*> [, <*alias*>]])

<indexpositie Nuitdr>

Geeft een .NDX-bestand aan door middel van de positie van een indexbestand in de lijst met geopende indexen voor de huidige of een opgegeven tabel.

<alias>

Een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of -aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

De functie NDX( ) geeft de naam van een .NDX-bestand als resultaat aan de hand van de positie van een indexbestand in de lijst met geopende indexen voor de huidige of een opgegeven tabel. De positie van indexen in de indexlijst wordt ingesteld met de commando's SET INDEX, SET ORDER en USE.

Als u geen indexpositie opgeeft, geeft NDX( ) de naam van de huidige hoofdindex als resultaat, of een lege tekenreeks als de hoofdindex een .MDX-bestand is. NDX( ) geeft ook een lege tekenreeks als resultaat als er geen index op de opgegeven positie in de lijst staat.

Als SET FULLPATH is ingeschakeld (ON), geeft NDX( ) tevens het station en het pad naar het .NDX-bestand als resultaat.

Voorbeeld

In het volgende voorbeeld worden drie indexen gemaakt. Vervolgens wordt NDX( ) gebruikt om de namen van geopende .NDX-bestanden voor de huidige tabel te bepalen:

```
USE Bedrijf EXCLUSIVE
INDEX ON CompCode TO CompCode
INDEX ON Plaats  TO Plaats
INDEX ON Postcode      TO PCode
SET INDEX TO PCode , CompCode, Plaats
* Nu zijn 3 indexen geopend
? "Index 1 ",NDX(1)
? "Index 2 ",NDX(2)
? "Index 3 ",NDX(3)
```

Zie ook

DBF( ), FIELD( ), KEY( ), MDX( ), ORDER( ), SET FULLPATH, SET INDEX, SET ORDER, TAG( ), USE

NETWORK( )

Gedeelde gegevens

Geeft .T. als resultaat als dBASE wordt uitgevoerd p een systeem waarin een LAN-kaart (*local area network*) of een andere netwerkkaart is geïnstalleerd.

Syntaxis

NETWORK( )

Beschrijving

Met NETWORK( ) kunt u vaststellen of een programma wordt uitgevoerd in een netwerkomgeving. Het kan bijvoorbeeld zijn dat uw programma in een netwerkomgeving bepaalde handelingen moet uitvoeren die op een zelfstandig systeem niet nodig zijn, zoals bij USE de optie EXCLUSIVE opgeven.

NETWORK( ) geeft .T. als resultaat als een netwerkkaart is geïnstalleerd. De functie controleert niet of dBASE ook werkelijk in een netwerkomgeving wordt uitgevoerd. Gebruik ID( ) om vast te stellen of de gebruiker ook echt in een netwerkomgeving actief is.

Voorbeeld

In het volgende voorbeeld wordt NETWORK( ) gebruikt om te testen of op het huidige systeem een netwerkkaart is geïnstalleerd. De gebruiker kan SET EXCLUSIVE dan alleen inschakelen (ON) als het echt nodig is. Zonder netwerkkaart kan SET EXCLUSIVE permanent zijn ingeschakeld:

```
IF NETWORK()
  SET EXCLUSIVE OFF && Inschakelen indien nodig
ELSE
  SET EXCLUSIVE ON  && Geen netwerk
ENDIF
```

Zie ook

GETENV( ), OS( ), USE

NEXTKEY( )

Toetsenbord- en muisacties

Geef de decimale waarde voor een toets of toetscombinatie in de typbuffer als resultaat. De toetsaanslag wordt niet uit de typbuffer verwijderd.

Syntaxis

NEXTKEY([<*Nuitdr*>])

<Nuitdr>

De positie van de toets of toetscombinatie in de typbuffer. Als <*Nuitdr*> wordt weggelaten, geeft NEXTKEY( ) de waarde van de eerste toetsaanslag in de buffer als resultaat. Als <*Nuitdr*> groter is dan het aantal toetsaanslagen in de typbuffer, geeft NEXTKEY( ) 0 als resultaat.

Beschrijving

In de typbuffer worden toetsaanslagen opgeslagen die de gebruiker invoert terwijl het programma bezig is met de verwerking van andere gegevens. Gebruik NEXTKEY( ) om te bepalen welke toetsaanslag in de buffer staat zonder deze te verwijderen. Als de buffer leeg is, geeft NEXTKEY( ) 0 als resultaat.

Als u bijvoorbeeld op *C* en vervolgens op *Alt-P* drukt, worden de waarden 67 en -420 opgeslagen in de typbuffer. NEXTKEY(1) geeft 67 als resultaat en NEXTKEY(2) geeft -420 als resultaat. Zie Appendix D voor een volledige lijst van de waarden die door NEXTKEY( ) als resultaat worden gegeven. Zie Appendix E voor een volledige lijst van decimale waarden voor toetsen.

Met INKEY( ) kunt u de waarde van de eerste toetsaanslag in de typbuffer ophalen en die waarde uit de typbuffer verwijderen.

Voorbeeld

In het volgende voorbeeld wordt gecontroleerd of onmiddellijk na F6 op Spatie is gedrukt. Als dat zo is, wordt de routine Spatiebalk aangeroepen; zo niet, dan wordt de typbuffer leeg gemaakt:

```
WAIT
IF LASTKEY() = -5       && F6
  IF NEXTKEY() = 32     && Spatie
    DO Spatiebalk
  ELSE
    CLEAR TYPEAHEAD
  ENDIF
ELSE
  DO AndereToets         && Geen F6
ENDIF
```

```
PROCEDURE Spatiebalk
? "Spatiebalk"

PROCEDURE AndereToets
? "Andere Toets"
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

BROWSE, CLEAR TYPEAHEAD, EDIT, INKEY( ), KEYBOARD, LASTKEY( ), ON KEY, READKEY( ), SET TYPEAHEAD.

NOTE

Programma's

Markeert een volledige programmaregel als een commentaarregel in plaats van een regel met uitvoerbare code.

Syntaxis

NOTE *<commentaar>*

<commentaar>

Een reeks van maximaal 1019 tekens op dezelfde regel. Als het laatste teken een puntkomma is, wordt de volgende regel als een voortzetting van het commentaar beschouwd.

Beschrijving

NOTE en * zijn onderling uitwisselbaar. Zie * voor een beschrijving van het commando.

Voorbeeld

Zie * voor een voorbeeld met NOTE.

Zie ook

*, &&

OEM( )

Tekenreeksgegevens

Resulteert in een tekenreeks die de OEM-versie voorstelt van een ANSI-tekenuitdrukking. OEM is een afkorting van Original Equipment Manufacturer en ANSI van American National Standards Institute.

Syntaxis

OEM(<*Tuitdr*>)

<Tuitdr>

De ANSI-tekenuitdrukking waarvan de OEM-versie als resultaat moet worden gegeven.

Beschrijving

OEM( ) is de tegengestelde functie van ANSI( ). Zie ANSI( ) voor meer informatie.

Voorbeeld

In het volgende voorbeeld worden de 255 tekens getoond die mogelijk zijn in de ASCII-, ANSI- en OEM-opmaak:

```
FOR i=1 to 255
  ASCII=CHR(i)
  ? i,ASCII,ANSI(ASCII),OEM(ASCII)
  * De volgende 3 commando's zorgen
  * elke 10 regels voor een pauze
  IF MOD(i,10)=0
    WAIT
  ENDIF
NEXT i
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ANSI( )

ON BAR

dBASE IV-menu's

Voert een commando uit als de gebruiker een strip selecteert in een dBASE IV-popup-menu. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van ON BAR. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ON ERROR — Foutafhandeling en testen op fouten

Voert een opgegeven commando uit als een fout optreedt.

Syntaxis

ON ERROR
[<*commando*>]

<*commando*>

Het commando dat moet worden uitgevoerd als een fout optreedt in het programma, de procedure of de door de gebruiker gedefinieerde functie. Als in het geval van een fout meerdere commando's moeten worden uitgevoerd, kunt u het commando ON ERROR DO <*bestandsnaam*> gebruiken. In dat geval verwijst <*bestandsnaam*> naar een programma- of procedurebestand met de opdrachten die moeten worden uitgevoerd. ON ERROR zonder het argument <*commando*> schakelt een eerder gegeven instructie met ON ERROR <*commando*> uit.

Beschrijving

Gebruik ON ERROR om de reactie van het programma op fouten tijdens de uitvoering (*runtime*-fouten) te besturen. U kunt met <*commando*> bijvoorbeeld een andere melding dan de standaardfoutmelding opgeven. Als ON ERROR actief is, worden geen standaardfoutmeldingen getoond.

Tijdens de uitvoering van een commando ON ERROR is de ON ERROR <*commando*>-instructie uitgeschakeld. Als zich tijdens de uitvoering van <*commando*> opnieuw een fout voordoet, wordt de standaardfoutmelding getoond. U kunt echter een volgende instructie met ON ERROR opgeven in de subroutine die in eerste instantie met ON ERROR is aangeroepen.

ON ERROR <*commando*> is niet van invloed op de volgende commando's:

- APPEND
- CREATE/MODIFY APPLICATION/LABEL/REPORT/SCREEN/STRUCTURE/VIEW
- MODIFY COMMAND/FILE

ON ERROR is vergelijkbaar met ON ESCAPE en ON KEY. ON ESCAPE geeft een commando aan dat moet worden uitgevoerd als SET ESCAPE is ingeschakeld (ON) en op *Esc* wordt gedrukt. Met ON KEY kunt u een commando opgeven dat moet worden uitgevoerd als op een opgegeven toets wordt gedrukt.

Voorkom recursieve aanroepen met ON ERROR.

In de tabel bij ERROR( ) worden CERROR( ), ERROR( ), MESSAGE( ), DBERROR( ), DBMESSAGE( ), SQLERROR( ) en SQLMESSAGE( ) met elkaar vergeleken.

Voorbeeld

In het volgende voorbeeld wordt ON ERROR gebruikt om een door de gebruiker gedefinieerd venster te openen als zich een fout voordoet. In dit voorbeeld veroorzaakt de instructie ? Bedrijf( ) een fout omdat er geen functie Bedrijf( ) bestaat:

```
ON ERROR DO FoutVerwerking WITH ERROR(), MESSAGE(),;
  PROGRAM(), LINENO()
USE Afnemers
? Bedrijf()
RETURN

PROCEDURE FoutVerwerking
PARAMETERS nFoutnr, tFoutmelding, tProgramma, nRegelnr
DEFINE FORM Attentie from 10,25 TO 20,80
DEFINE TEXT Regel1 OF Attentie AT 2,2 ;
  PROPERTY Text "Er is een fout opgetreden",;
  Width 60
DEFINE TEXT Regel2 OF Attentie AT 4,2;
  PROPERTY Text "Fout: " + tFoutmelding,;
  Width 60
DEFINE TEXT Regel3 OF Attentie AT 5,2;
  PROPERTY Text "Nummer: " + STR(nFoutnr),;
  Width 60
DEFINE TEXT Regel4 OF Attentie AT 6,2;
  PROPERTY Text "Programma: "+ tProgramma,;
  Width 60
DEFINE TEXT Regel5 OF Attentie AT 7,2;
  PROPERTY Text "Regelnummer: " + STR(nRegelnr),;
  Width 60
OPEN FORM Attentie
```

Zie ook

CERROR( ), ERROR( ), DBERROR( ), DBMESSAGE( ), MESSAGE( ), ON ESCAPE, ON KEY, ON READERROR, RETRY, RETURN, SET ERROR, SET ESCAPE, SQLERROR( ), SQLMESSAGE( )

ON ESCAPE

Toetsenbord- en muisacties

Voert een opgegeven commando uit als SET ESCAPE is ingeschakeld (ON) en tijdens de uitvoering van een programma of commando op *Esc* wordt gedrukt.

Syntaxis

ON ESCAPE
  [<*commando*>]

<commando>

Het commando dat moet worden uitgevoerd onder de volgende omstandigheden:

- SET ESCAPE is ingeschakeld (ON)
- De gebruiker drukt tijdens de uitvoering van een programma of commando op *Esc*

Als meerdere commando's moeten worden uitgevoerd wanneer op *Esc* is gedrukt, kunt u ON ESCAPE DO <*bestandsnaam*> gebruiken. In dit geval verwijst <*bestandsnaam*> naar een programma of procedurebestand met de reeks commando's die moeten worden uitgevoerd. ON ESCAPE zonder argument schakelt een voorgaande instructie met ON ESCAPE uit.

Als u in een programma ON ESCAPE <*commando*> gebruikt, moet u die instructie weer uitschakelen voordat het programma wordt beëindigd. U doet dat met ON ESCAPE (zonder argument). Als u dat niet doet, blijft de instructie van kracht totdat dBASE wordt afgesloten.

Beschrijving

Als SET ESCAPE is ingeschakeld (ON, de standaardinstelling), kan de gebruiker op *Esc* drukken om de uitvoering van het programma te annuleren. Als het commando ON ESCAPE <*commando*> is opgegeven, wordt het opgegeven commando uitgevoerd nadat op *Esc* is gedrukt en wordt het programma vervolgens voortgezet. Als ON ESCAPE <*commando*> niet is opgegeven, wordt het programma afgebroken en verschijnt het venster **Programma onderbroken** als op *Esc* is gedrukt.

Als u ON ESCAPE <*commando*> gebruikt in het commandovenster, wordt <*commando*> uitgevoerd als op *Esc* wordt gedrukt. In dat geval wordt de besturing teruggegeven aan het commandovenster nadat <*commando*> is uitgevoerd.

Als op *Esc* wordt gedrukt terwijl een door de gebruiker gedefinieerd venster, menubalk, popup-menu of afrolmenu actief is, wordt het object gedeactiveerd en wordt het commando dat is opgegeven met ON ESCAPE *niet* uitgevoerd. ON ESCAPE <*commando*> heeft ook geen effect als op *Esc* wordt gedrukt tijdens de uitvoering van commando's als CHANGE, EDIT en READ.

ON ESCAPE is vergelijkbaar met ON ERROR en ON KEY. ON ERROR geeft een commando aan dat moet worden uitgevoerd als tijdens de uitvoering van een programma, procedure of door de gebruiker gedefinieerde functie een fout optreedt. Met ON KEY kunt u aangeven welk commando moet worden uitgevoerd als de gebruiker op een opgegeven toets drukt.

Als SET ESCAPE is ingeschakeld (ON) en zowel ON KEY (voor de Esc-toets) als ON ESCAPE zijn actief, heeft ON ESCAPE een hogere prioriteit dan ON KEY. Bij ON KEY vindt u een tabel waarin de verhoudingen worden beschreven tussen SET ESCAPE, ON ESCAPE en ON KEY als op *Esc* wordt gedrukt.

Voorbeeld

In het volgende voorbeeld wordt ON ESCAPE gebruikt om een klein programma, PrgEscape, uit te voeren als op *Esc* wordt gedrukt. In dit voorbeeld heeft de

programmeur een oneindige lus geprogrammeerd die alleen kan worden onderbroken met *Esc*. Als op *Esc* wordt gedrukt, worden de programmanaam en -regel getoond:

```
SET PROCEDURE TO PROGRAM(1) ADDITIVE
* Andere programmabestanden kunnen over PrgEcape beschikken
SET ESCAPE ON
ON ESCAPE DO PrgEscape WITH PROGRAM(1),LINENO()
DO WHILE .t.   && oneindige lus instellen
  x="Zorg dat u altijd uit een"
  y="do...while-lus kunt ontsnappen!"
  z="Druk op Escape om te stoppen"
  ? x
  ? y
  ? z
ENDDO

Procedure PrgEscape
PARAMETERS prg,regel
? "Er is op Esc gedrukt:",prg,regel
SUSPEND
```

Zie ook

ON ERROR, ON KEY, SET ESCAPE, SET KEY

ON EXIT BAR — dBASE IV-menu's

Voert een commando uit als de gebruiker een element in een popup-menu verlaat. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van ON EXIT BAR. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ON EXIT MENU — dBASE IV-menu's

Voert een commando uit als de gebruiker een strip in een dBASE IV-menubalk verlaat, als aan die strip geen commando is toegewezen met ON EXIT PAD. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van ON EXIT MENU. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ON EXIT PAD

dBASE IV-menu's

Voert een commando uit als de gebruiker een bepaalde strip in een dBASE IV-menubalk verlaat. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van ON EXIT PAD. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ON EXIT POPUP

dBASE IV-menu's

Voert een commando uit als de gebruiker een balk in een dBASE IV popup-menu verlaat, als aan die balk geen commando is toegewezen met ON EXIT BAR. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van ON EXIT POPUP. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ON KEY

Toetsenbord- en muisacties

Voert een commando uit als tijdens de uitvoering van een commando of programma op een opgegeven toets of toetscombinatie wordt gedrukt.

Syntaxis

ON KEY
  [LABEL <*toetslabel*>]
  [<*commando*>]

LABEL <*toetslabel*>

Geeft aan welke toets of toetscombinatie tot gevolg heeft dat <*commando*> wordt uitgevoerd. Zonder LABEL <*toetslabel*>, wordt na elke toetsaanslag <*commando*> uitgevoerd. Bij ON KEY LABEL wordt geen onderscheid gemaakt tussen hoofdletters en kleine letters.

<*commando*>

Het commando dat wordt uitgevoerd als de gebruiker op de toets of toetscombinatie drukt. Als u <*commando*> achterwege laat, wordt het commando dat eerder is toegewezen met ON KEY uitgeschakeld.

Beschrijving

Met ON KEY kunt u een commando opgeven dat moet worden uitgevoerd als de gebruiker op een toets of toetscombinatie drukt. Als op de toets of toetscombinatie wordt gedrukt, wordt de uitvoering van het huidige commando onderbroken en wordt *<commando>* uitgevoerd. Als *<commando>* voltooid is, wordt het onderbroken commando verder uitgevoerd.

De meeste commando's worden met ON KEY onderbroken. Commando's die ononderbroken moeten worden uitgevoerd om de betrouwbaarheid van gegevens te garanderen, zoals SORT, INDEX en PACK worden echter niet onderbroken. ON KEY heeft ook geen invloed op enkele commando's die invoer van de gebruiker accepteren, zoals ACCEPT en WAIT.

Als u zowel ON KEY LABEL *<toetslabel>* *<commando>* als ON KEY *<commando>* gebruikt, heeft de toets of toetscombinatie die u opgeeft bij ON KEY LABEL *<toetslabel>* *<commando>* de hoogste prioriteit en wordt het daarbij opgegeven commando uitgevoerd.

Als u geen LABEL opgeeft, kan slechts één toetstoewijzing actief zijn. Als u LABEL wel gebruikt, kunnen voor verschillende toetsen verschillende commando's actief zijn.

ON KEY zonder argumenten verwijdert alle eerder met ON KEY *<commando>* ingevoerde definities.

SET KEY komt overeen met ON KEY LABEL. Alleen de syntaxis is anders. Als u ON KEY LABEL en SET KEY gebruikt voor dezelfde toets, voert dBASE het programma of de procedure uit dat of die is ingesteld met het laatste commando.

ON KEY is vergelijkbaar met ON ESCAPE. Met ON ESCAPE geeft u een commando op dat moet worden uitgevoerd als SET ESCAPE is ingeschakeld (ON) en de gebruiker op *Esc* drukt. Als SET ESCAPE is ingeschakeld (ON) en zowel ON KEY als ON ESCAPE zijn actief, heeft ON ESCAPE de hoogste prioriteit als op *Esc* wordt gedrukt. De volgende tabel geeft een overzicht van de verhoudingen tussen SET ESCAPE, ON ESCAPE en ON KEY als op *Esc* wordt gedrukt:

SET ESCAPE	ON ESCAPE	ON KEY	Als u op *Esc* drukt
ON		DO prog1	Venster **Programma onderbroken** verschijnt
OFF		DO prog1	Prog1 wordt uitgevoerd
ON	DO prog2	DO prog1	Prog2 wordt uitgevoerd
OFF	DO prog2	DO prog1	Prog1 wordt uitgevoerd

ON KEY LABEL

Gebruik de volgende toetslabelnamen om met ON KEY LABEL *<toetslabel>* toetsen of toetscombinaties toe te wijzen.

Toets	*<toetslabel>*
Letters	A of a, B of b, ...
Cijfers	0, 1, 2 ...
Backspace	Backspace

Toets	***<toetslabel>***
Back Tab (*Shift-Tab*)	Backtab
Delete	Del
End	End
Home	Home
Insert	Ins
Page Up	PgUp
Page Down	PgDn
Tab	Tab
←	Leftarrow
→	Rightarrow
↑	Uparrow
↓	Dnarrow
F1 tot en met *F10*	F1, F2, F3, ...
Ctrl-<toets>	Ctrl-*<toets>*
Shift-<toets>	Shift-*<toets>*
Alt-<toets>	Alt-*<toets>*
Enter	Enter
Escape	Esc
Spatiebalk	Spacebar

Voorbeeld

In het volgende voorbeeld worden uit tien records bepaalde velden getoond, waarna het programma drie seconden pauzeert alvorens meer records te tonen. ON KEY LABEL wordt gebruikt om naar procedures te springen. Deze procedures wijzigen de richting van de recordaanwijzer, openen een bladervenster of stoppen met bladeren:

```
CLEAR
SET TALK OFF
PUBLIC mStoppen
mStoppen=.F.
ON KEY LABEL F8 DO Omhoog
ON KEY LABEL F9 DO Details
ON KEY LABEL F10 DO Stoppen
USE Afnemers
? CENTER("Aanwijzingen")
? CENTER("F8: ga X records omhoog")
? CENTER("F9: bladeren in alle velden")
? CENTER("F10: stoppen")
?
WAIT "Langzaam bladeren - Druk op een toets"
Cnt=1
DO WHILE .NOT. EOF() .AND. .NOT. mStoppen
  DISPLAY NEXT 10 Bedrijf, Contact
  Pauze=INKEY(10)
ENDDO
SET TALK ON
ON KEY
RETURN
```

```
PROCEDURE Omhoog
CLEAR
SKIP -9
RETURN

PROCEDURE Details
SKIP -9
BROWSE
RETURN

PROCEDURE Stoppen
mStoppen=.T.
RETURN
```

Overdraagbaarheid

De optie LABEL <*toetslabel*> wordt niet ondersteund in dBASE III PLUS.

Zie ook

CLEAR TYPEAHEAD, INKEY( ), KEYBOARD, ON ERROR, ON ESCAPE, SET ESCAPE, SET FUNCTION

ON MENU — dBASE IV-menu's

Voert een commando uit als de gebruiker een strip in een dBASE IV-menubalk kiest en als aan die strip geen commando is toegewezen met ON PAD. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van ON MENU. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ON MOUSE — Toetsenbord- en muisacties

Voert een commando uit als de gebruiker linksklikt en de muisknop loslaat. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows kenmerken als OnLeftMouseDown voor het besturen van muishandelingen in formulieren.

Zie Help voor meer informatie over de syntaxis van ON MOUSE. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het maken van formulieren.

ON NETERROR

Voert een opgegeven commando uit als een netwerkfout optreedt.

Syntaxis

ON NETERROR [*<commando>*]

<commando>

Het commando dat moet worden uitgevoerd als een netwerkfout optreedt. Als u meerdere commando's wilt uitvoeren als een dergelijke fout optreedt, moet u ON NETERROR DO *<bestandsnaam>* gebruiken. In dat geval stelt *<bestandsnaam>* een programma- of procedurebestand voor met de reeks commando's die moeten worden uitgevoerd. ON NETERROR zonder argumenten schakelt een eerder met ON NETERROR ingesteld commando uit.

Beschrijving

Met ON NETERROR kunt u de reactie van een programma besturen als een netwerkfout optreedt. Op een LAN (*local area network*) in een multi-user-omgeving kan bijvoorbeeld een fout optreden als twee gebruikers tegelijk proberen hetzelfde record in een gedeelde tabel te wijzigen, of als een gebruiker tracht een gedeelde tabel te openen als een andere gebruiker die al exclusief heeft geopend.

ON NETERROR is vergelijkbaar met ON ERROR, maar ON ERROR reageert op *alle* fouten tijdens de uitvoering van een programma, inclusief typische multi-user-fouten. U kunt alle fouten verwerken met ON ERROR, of ON NETERROR gebruiken om alleen de multi-user-fouten te verwerken. Als u zowel ON ERROR als ON NETERROR gebruikt, reageert ON ERROR op alle gewone fouten, terwijl ON NETERROR de typische multi-user-fouten verwerkt.

Tijdens de uitvoering van een commando ON NETERROR is de ON NETERROR *<commando>*-instructie uitgeschakeld. Als zich tijdens de uitvoering van *<commando>* opnieuw een netwerkfout voordoet, wordt de standaardfoutmelding getoond. U kunt echter een volgende instructie met ON NETERROR opgeven in de subroutine die in eerste instantie met ON NETERROR wordt aangeroepen.

Voorkom recursieve aanroepen met ON NETERROR.

Voorbeeld

In het volgende voorbeeld wordt ON NETERROR gebruikt als de tabel Klanten niet exclusief kan worden geopend:

```
SET PROCEDURE TO PROGRAM(1) ADDITIVE
ON NETERROR Do Netfout
USE Klanten EXCLUSIVE
* Als Klanten niet exclusief kan worden geopend,
* wordt subroutine Netfout aangeroepen.
```

```
PROCEDURE Netfout
WAIT "Multi-user-probleem"
```

Zie ook

DO, ERROR( ), MESSAGE( ), ON ERROR, RETRY, RETURN, SET EXCLUSIVE, SET REPROCESS, SQLERROR( ), SQLMESSAGE( )

ON PAD

dBASE IV-menu's

Voert een commando uit als de gebruiker een strip in een dBASE IV-menubalk selecteert. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van ON PAD. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ON PAGE

Afdrukken

Voert een opgegeven commando uit als afgedrukte uitvoer een bepaalde regel op de huidige pagina bereikt.

Syntaxis

ON PAGE
[AT LINE <*Nuitdr*> <*commando*>]

AT LINE <*Nuitdr*>

Het nummer van de regel waar het commando voor pagina-opmaak moet worden uitgevoerd.

<*commando*>

Het commando dat moet worden uitgevoerd als afgedrukte uitvoer een bepaalde regel op de huidige pagina bereikt. Als u wilt dat meerdere commando's worden uitgevoerd, moet u ON PAGE DO <*bestandsnaam*> gebruiken, waarbij <*bestandsnaam*> een programma- of procedurebestand voorstelt met de reeks commando's die moeten worden uitgevoerd.

Beschrijving

Met ON PAGE kunt u een commando opgeven dat moet worden uitgevoerd als afgedrukte uitvoer een bepaalde regel op de huidige pagina bereikt. ON PAGE zonder argumenten schakelt een eerder met ON PAGE ingesteld commando uit.

De waarde van de systeemvariabele _plineno geeft aan hoeveel regels op de huidige pagina zijn afgedrukt. Zodra de waarde van _plineno gelijk is aan de waarde die u hebt opgegeven voor <*Nuitdr*>, wordt het bij ON PAGE opgegeven commando uitgevoerd.

Met het commando ON PAGE kunt u *kop-* en *voetteksten* afdrukken. Het commando ON PAGE kan bijvoorbeeld een procedure aanroepen als de systeemgeheugenvariabele _plineno een bepaald regelnummer onder aan de pagina bereikt. De procedure kan vervolgens twee procedures aanroepen: de eerste om onder aan de huidige pagina een voettekst af te drukken en de tweede om boven aan de volgende pagina een koptekst af te drukken.

Een routine voor het afdrukken van een koptekst kunt u beginnen met EJECT PAGE om te zorgen dat de koptekst boven aan de volgende pagina wordt afgedrukt. EJECT PAGE maakt tevens de systeemgeheugenvariabele _plineno gelijk aan 0. Met het commando ? kunt u enkele regels overslaan aan het begin van de procedure voor de koptekst. U kunt dit commando ook gebruiken om na de koptekst enkele regels over te slaan.

Een routine voor voetteksten kunt u beginnen met het commando ? om enkele regels over te slaan voordat de voettekst wordt afgedrukt. U kunt het commando ?? gebruiken in combinatie met de systeemgeheugenvariabele _pageno om op dezelfde regel als de voettekst een paginanummer af te drukken.

De juiste positie voor de voettekst berekent u als volgt: tel het aantal regels voor de voettekst en het aantal regels voor de ondermarge bij elkaar op. Trek dat getal af van het totaal aantal regels op een pagina. De uitkomst kunt u gebruiken als getal bij het argument AT LINE. Als de voettekst groter is dan het totaal aantal regels op een pagina, wordt de rest van de voettekst op de volgende pagina afgedrukt.

Voorbeeld

In dit voorbeeld wordt EJECT PAGE gebruikt in combinatie met ON PAGE om op elke pagina een voettekst af te drukken. Er wordt een paginalengte gebruikt van 5 regels (regels 0 tot en met 4). Op vier regels (0,1,2,3) wordt tekst afgedrukt en de voettekst komt op regel 4. In totaal worden acht regels afgedrukt op twee pagina's:

```
SET TALK OFF
CLEAR
SET PRINTER ON
_padvance="LINEFEEDS"
_plength=5
_pageno=1
EJECT
ON PAGE AT LINE 3 do Nw_Pag
PRINTJOB
i=1
DO WHILE  i<=8
  ?? "Tekstregel  ",_pageno,_plineno,i
  ?
  i=i+1
ENDDO
ON PAGE  && ON PAGE uitschakelen
ENDPRINTJOB
CLOSE PRINTER
```

```
PROCEDURE Nw_Pag
??  "   Voettekst",_pageno,_plineno
EJECT PAGE
RETURN
*                        Pagina     Regel        i
* Afgedrukt ziet dit er zo uit:
* Tekstregel               1          0          1
* Tekstregel               1          1          2
* Tekstregel               1          2          3
* Tekstregel               1          3          4
*    Voettekst             1          4
* Tekstregel               2          0          5
* Tekstregel               2          1          6
* Tekstregel               2          2          7
* Tekstregel               2          3          8
*    Voettekst             2          4
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

?, ??, _pageno, _plineno, EJECT PAGE, PRINTJOB...ENDPRINTJOB, SET PCOL, SET PRINTER, SET PROW

ON POPUP

dBASE IV-menu's

Voert een commando uit als de gebruiker een optie in een dBASE IV popup-menu selecteert, als aan die optie geen commando is toegewezen met ON BAR. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van ON POPUP. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ON READERROR

Foutafhandeling en testen op fouten

Voert een opgegeven commando uit als een fout optreedt tijdens het invoeren van gegevens.

Syntaxis

ON READERROR
  [<*commando*>]

<commando>

Het commando dat moet worden uitgevoerd als de gebruiker een ongeldige waarde invoert in een veld. ON READERROR zonder *<commando>* schakelt elk eerder met ON READERROR opgegeven commando uit.

Beschrijving

Met ON READERROR kunt u de volgende fouten tijdens de invoer van gegevens onderscheppen:

- Ongeldige datums
- Invoer die valt buiten het bereikt dat is ingesteld met de optie RANGE van @...SAY...GET
- Invoer die niet voldoet aan de voorwaarde die is ingesteld met de optie VALID van @...SAY...GET

Het commando dat u opgeeft bij ON READERROR kan een aanroep naar een programma, procedure of door de gebruiker gedefinieerde functie zijn. U kunt bijvoorbeeld een door de gebruiker gedefinieerde functie aanroepen die een melding toont.

Ook als ON READERROR niet wordt gebruikt, moet de gebruiker geldige datums invoeren of gegevens die voldoen aan een clausule met VALID of RANGE. Als de gebruiker een ongeldige waarde invoert terwijl met RANGE een bereik is ingesteld, maar geen instructie met ON READERROR is opgegeven, wordt een melding getoond waarin de geldige invoerwaarden worden aangegeven.

Voorbeeld

In het volgende voorbeeld wordt de tabel Klanten getoond in een tabelrecordsvenster en wordt ON READERROR gebruikt om ongeldige invoer te onderscheppen. Als u in BalDatum een ongeldige datum invoert, wordt ON READERROR geactiveerd en toont de procedure FoutVerw een venster met toepasselijke meldingen:

```
ON READERROR DO FoutVerw
USE Afnemers
BROWSE
RETURN

PROCEDURE FoutVerw
DEFINE FORM Attentie FROM 10,25 TO 20,65
DEFINE TEXT Regel1 OF Attentie AT 2,5;
  PROPERTY Text "Er is een fout opgetreden",;
  Width 40, ColorNormal "R/W"
DEFINE TEXT Regel2 OF Attentie AT 4,5;
  PROPERTY Text "Fout: datum buiten geldig bereik",;
  Width 40
DEFINE TEXT Regel3 OF Attentie AT 6,5;
  PROPERTY Text "Voer geldige datum in",;
  Width 40
OPEN FORM Attentie
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

APPEND, CHANGE, EDIT, INSERT, ON ERROR, READ, @...SAY...GET

ON SELECTION BAR

dBASE IV-menu's

Voert een commando uit als de gebruiker een optie kiest in een dBASE IV popup-menu. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van ON SELECTION BAR. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ON SELECTION FORM

Formulieren

Voert een subroutine of een codeblok uit als de gebruiker een formulier voorlegt.

Syntaxis

ON SELECTION FORM <*formuliernaam*>
  [<*FPuitdr*> | <*CBuitdr*>]

<*formuliernaam*>

De naam van het formulier waaraan <*FPuitdr*> of <*CBuitdr*> is toegewezen.

<*FPuitdr*> | <*CBuitdr*>

Een subroutine of codeblok dat wordt uitgevoerd als de gebruiker het formulier voorlegt. <*FPuitdr*> is de naam van een functie of procedure en <*CBuitdr*> is een codeblok.

Als u <*FPuitdr*> | <*CBuitdr*> achterwege laat, wordt een eerdere toewijzing met ON SELECTION FORM uitgeschakeld.

Beschrijving

Met ON SELECTION FORM kunt u automatisch een subroutine of codeblok uitvoeren als de gebruiker een formulier voorlegt.

Een formulier wordt voorgelegd als de gebruiker een van de volgende handelingen uitvoert:

- Op *Enter* drukken terwijl de cursor binnen het formulier staat, maar niet in een bladerobject, een editor-object of een schijfletter of directory in een lijstvak (al of niet met invoervak).
- Op de *spatiebalk* drukken als de cursor op een knop staat.
- De keuzeletter van een knop of *Alt-<keuzeletter>* typen, terwijl de cursor binnen een formulier staat.
- Met de muis op een knop klikken.

ON SELECTION FORM komt overeen met het kenmerk OnSelection van het object Form.

Voorbeeld

In het volgende voorbeeld wordt een formulier gedefinieerd met een lijstvak voor het kiezen van een vliegtuig. ON SELECTION FORM wordt gebruikt om de procedure Foto aan te roepen als de gebruiker op Enter drukt. De procedure Foto toont een afbeelding uit het binaire veld Afbeelding van het geselecteerde record:

```
CLOSE ALL
SET TALK OFF
SET PROCEDURE TO PROGRAM(1) ADDITIVE
USE Vliegtg ORDER VLIEGTUIG IN SELECT()
SELECT Vliegtg
DEFINE FORM VT              ;
   PROPERTY                 ;
      Top 5,                ;
      Left 60,              ;
      Height 13,            ;
      Width 30,             ;
      Text "Vliegtuig",     ;
      Sizeable .T.
DEFINE LISTBOX Model OF VT            ;
   PROPERTY                           ;
      Top 2,                          ;
      Left 6,                         ;
      Height 7,                       ;
      Width 18,                       ;
      DataSource "FIELD Vliegtg->Vliegtuig"
ON SELECTION FORM VT Foto
OPEN FORM VT

FUNCTION Foto
RESTORE IMAGE FROM BINARY Afbeelding
RETURN .T.
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ActiveControl, _curobj, OnClick, OnSelection

ON SELECTION MENU

dBASE IV-menu's

Voert een commando uit als de gebruiker een strip kiest in een dBASE IV-menubalk, als aan die strip niet eerder een commando is toegewezen met ON SELECTION PAD. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van ON SELECTION MENU. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ON SELECTION PAD

dBASE IV-menu's

Geeft aan welk commando moet worden uitgevoerd als de gebruiker een strip kiest in een dBASE IV-menubalk. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van ON SELECTION PAD. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

ON SELECTION POPUP

dBASE IV-menu's

Voert een commando uit als de gebruiker een optie kiest in een dBASE IV popup-menu, als aan die optie niet eerder een commando is toegewezen met ON SELECTION BAR. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van ON SELECTION POPUP. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

OPEN DATABASE

Tabellen

Brengt een verbinding tot stand met een database-server of een database die is gedefinieerd voor een bepaalde directory.

Syntaxis

OPEN DATABASE <*databasenaam*>
  [LOGIN <*gebruikersnaam*>/<*wachtwoord*>]
  [WITH <*optie Tuitdr*>]

<databasenaam>
De naam van de database die u wilt openen. Databases worden gemaakt met het IDAPI-configuratieprogramma (zie *Aan de slag* voor meer informatie).

<gebruikersnaam>/<wachtwoord>
Een tekenreeks die de gebruikersnaam en het wachtwoord aangeeft die nodig zijn om toegang tot de database te verkrijgen.

WITH *<optiereeks>*
Tekenreeks met specifieke informatie voor de server die nodig is om een verbinding met een database-server tot stand te brengen. Raadpleeg de documentatie bij Borland SQL Link voor informatie over het tot stand brengen van verbindingen met database-servers, en neem contact met de netwerk- of databasebeheerder voor specifieke informatie over uw verbindingen.

Beschrijving

Het commando OPEN DATABASE wordt gebruikt om een verbinding tot stand te brengen met een database die is gedefinieerd met het IDAPI-configuratieprogramma. Als u een database opent, moet u bepaalde informatie opgeven, zoals aanmeldingsparameters en specifieke informatie voor de database. In de meeste gevallen kan de netwerk- of systeembeheerder u vertellen welke informatie u precies moet opgeven om verbindingen tot stand te brengen met databases en database-servers op uw werkplek.

Voorbeeld

In het volgende voorbeeld wordt OPEN DATABASE gebruikt om een verbinding tot stand te brengen met een database-server. Een tabel wordt geopend uit de database die eerder is gemaakt met het IDAPI-configuratieprogramma met SET DATABASE TO. Vervolgens wordt een lokale dBASE-tabel geopend en toegevoegd aan de client/server-database:

```
CLEAR
SET DBTYPE TO DBASE
OPEN DATABASE NWKlanten      && Verbinding met database-;
                                server tot stand brengen
USE NWKlanten IN 1           && Server-tabel openen
SELECT 1                     && Werkgebied 1 activeren
APPEND FROM KLANTEN.DBF      && Toevoegen vanuit;
                                lokale dBASE-tabel
CLOSE ALL                    && Server-tabel ;
                                Klanten sluiten
CLOSE DATABASE NWKlanten     && Verbinding met database- ;
                                server verbreken
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CLOSE..., DATABASE( ), SET DATABASE, SET DBTYPE

OPEN FORM — Formulieren

Opent formulieren als niet-modale vensters.

Syntaxis

OPEN FORM <*formuliernaam1*> [ON <*objectnaam1*>]
[, <*formuliernaam2*> [ON <*objectnaam2*>], ...]

<*formuliernaam1*> [, <*formuliernaam2*>, ...]

De namen van de formulieren die moeten worden geopend.

ON <*objectnaam*>

Geeft aan welk object in een formulier in eerste instantie de focus heeft.

Beschrijving

Gebruik OPEN FORM om formulieren te openen en hun objecten weer te geven.

Formulieren die u opent met OPEN FORM, zijn niet-modaal. Een niet modaal formulier heeft de volgende eigenschappen:

- Zolang het formulier is geopend, kan de focus worden verplaatst naar andere formulieren.
- De routine waarmee het formulier is geopend, wordt verder uitgevoerd nadat het formulier is geopend.

Open formulieren als niet-modale vensters als u wilt dat tegelijk meerdere formulieren geopend kunnen zijn. een applicatie die verschillende taken uitvoert, kan bijvoorbeeld voor elke taak een ander formulier gebruiken.

Slechts één formulier kan de focus hebben. Dat formulier wordt het *actieve* formulier genoemd en weergegeven boven op alle andere geopende formulieren. Het formulier dat u als eerste hebt geopend met OPEN FORM, krijgt het eerst de focus. Als het laatste formulier wordt gesloten, is de uitvoering van de applicatie voltooid.

Opmerking Als u een formulier wilt openen als een *modaal venster*, moet u de methode ReadModal( ) of de functie READMODAL( ) gebruiken. Formulieren waarmee dialoogvensters worden gedefinieerd, zijn bijvoorbeeld modaal omdat de uitvoering van het programma wordt onderbroken tot de gebruiker heeft gereageerd.

Het commando OPEN FORM komt overeen met de methode Open( ).

Voorbeeld

In het volgende voorbeeld worden vier formulieren met standaardafmetingen gedefinieerd en wordt de syntaxis van het commando OPEN FORM gebruikt om te bepalen in welke volgorde de formulieren de focus krijgen:

```
DEFINE FORM Opn1
DEFINE FORM Opn2 AT 4,4
DEFINE FORM Opn3 AT 8,8
DEFINE FORM Opn4 AT 12,12
OPEN FORM Opn4, Opn1, Opn3, Opn2
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

Open( ), READMODAL( ), ReadModal( )

ORDER( ) — Tabelindeling

Geeft de naam van het hoofd-NDX-bestand of de hoofd-MDX-indexlabel in het huidige of een opgegeven werkgebied als resultaat.

Syntaxis

ORDER([<*alias*>])

<alias>

Een werkgebiednummer (van 1 tot 255) of -letter (van A tot J) of aliasnaam. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

ORDER( ) geeft de naam van het hoofd-NDX-bestand of de hoofd-MDX-indexlabel in het huidige of een opgegeven werkgebied als resultaat. ORDER( ) geeft een lege tekenreeks ("") als resultaat als in het opgegeven werkgebied geen hoofdindex bestaat.

Met ORDER( ) kunt u de naam van de hoofdindex bepalen en die naam opslaan in een geheugenvariabele. U kunt dan later met het commando SET ORDER de hoofdindex herstellen.

Voorbeeld

In het volgende voorbeeld wordt ORDER( ) gebruikt om de naam van de hoofdindex voor de huidige tabel te tonen:

```
USE Bedrijf EXCLUSIVE
INDEX ON Bedrijf TAG Bedrijf
INDEX ON Plaats  TAG Plaats
```

```
SET ORDER TO Bedrijf
? "Voor de tabel " + DBF() + " is " +;
   ORDER() + " het hoofdindexbestand"
SET ORDER TO       && geen index
? "Voor de tabel " + DBF() + " is " +;
   ORDER() + " het hoofdindexbestand"
SET ORDER TO Plaats
? "Voor de tabel " + DBF() + " is " +;
   ORDER() + " het hoofdindexbestand"
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ALIAS( ), KEY( ), MDX( ), NDX( ), SELECT( ), SET INDEX, SET ORDER, TAG( ), USE

OS( )

Stations- en bestandsfuncties

Geeft de naam en het versienummer van het huidige besturingssysteem of de huidige versie van Windows als resultaat.

Syntaxis

OS([<*Nuitdr*>])

<Nuitdr>

Elk willekeurig nummer. OS( ) zonder <*Nuitdr*> geeft de naam en versie van het huidige besturingssysteem als resultaat. OS(*Nuitdr*) geeft de versie van Windows als resultaat.

Beschrijving

Met OS( ) kunt u bepalen welk besturingssysteem of welke versie van Windows actief is als uw programma's worden uitgevoerd. Met VERSION( ) kunt u bepalen welke versie van dBASE actief is.

Met OS( ) kunt u uw applicaties beter overdraagbaar maken. Als u van plan bent een applicatie op meerdere systemen uit te voeren en de kans bestaat dat op sommige systemen een DOS-versie wordt gebruikt die te oud is voor bepaalde codefragmenten in uw applicatie, kunt u de uitvoering van deze fragmenten afhankelijk maken van het resultaat van de functie OS( ).

Voorbeeld

In dit voorbeeld wordt OS( ) gebruikt om te controleren of de actieve versie van Windows versie 3.10 of nieuwer is. Het aantal cijfers rechts van 'versie' wordt bepaald en de cijfers worden uit de resultaatreeks gehaald. Vervolgens wordt de reeks omgezet in een getal en vergeleken met 3.10:

```
Bs=OS(1) && Geeft "Windows-versie: ??.??" als resultaat
L=Len(Bs)
* Aantal tekens
Bs=UPPER(Bs)
* Omzetten naar hoofdletters
Pos=AT("VERSION",Bs)
* Plaats van 'versie' in Bs bepalen
* Pos geeft positie van 'v' in 'versie' aan
Rechts=L-Pos-5
* Aantal tekens na 'versie' bepalen
VersieRks=Right(Bs,Rechts)
VersieNum=Val(VersieRks)
* Omzetten naar getal
IF VersieNum<3.10
  ? "Het actieve besturingssysteem is",OS()
  ? "De actieve versie van Windows is",OS(1)
  WAIT ;
   "Dit programma vereist Windows versie 3.10 of nieuwer"
ENDIF
```

Overdraagbaarheid

De optie <*Nuitdr*> wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

VERSION( )

PACK

Velden en records

Verwijdert (schoont) uit de huidige tabel alle voor verwijdering gemarkeerde records. Dit commando heeft geen gevolgen voor Paradox- en SQL-tabellen, waarin records onmiddellijk worden verwijderd.

Syntaxis

PACK

Beschrijving

Met PACK kunt u uit de huidige tabel de records verwijderen die eerder met het commando DELETE zijn gemarkeerd voor verwijdering. De tabel moet exclusief zijn geopend als u PACK wilt gebruiken.

Nadat u PACK hebt uitgevoerd, is de schijfruimte die werd ingenomen door de verwijderde records weer beschikbaar. Alle geopende indexbestanden worden automatisch opnieuw samengesteld nadat PACK is uitgevoerd. Gesloten indexbestanden kunt u later opnieuw samenstellen met REINDEX.

Ruimte in .DBT-memobestanden voor verwijderde records wordt echter niet vrijgegeven als u het commando PACK gebruikt. Als u de ruimte in een memobestand weer beschikbaar wilt maken, moet u de oorspronkelijke tabel kopiëren met COPY.

Let goed op als u PACK gebruikt. Records die zijn gemarkeerd voor verwijdering, maar nog niet daadwerkelijk uit het bestand zijn verwijderd met PACK (nog niet zijn geschoond), kunnen worden herroepen met RECALL. Records die daadwerkelijk zijn verwijderd met PACK, zijn permanent verdwenen en kunnen niet meer worden hersteld.

SET DELETED ON biedt een groot aantal van de voordelen van PACK zonder dat feitelijk records uit de tabel worden verwijderd. Als SET DELETED is ingeschakeld (ON), werken de meeste commando's alsof de voor verwijdering gemarkeerde records uit de tabel zijn verwijderd.

Als u in een keer alle records permanent uit een tabel wilt verwijderen, gebruikt u het commando ZAP.

Voorbeeld

In het volgende voorbeeld wordt een formulier gemaakt met een invoervak voor het invoeren van een Regio-code en een knop die een procedure aanroept. De procedure gebruikt PACK om permanent alle voor verwijdering gemarkeerde records te verwijderen uit het bestand TIJDLK.DBF:

```
* Hoofdprogramma
SET SAFETY OFF
USE Klanten
COPY TO Temp
USE Temp EXCLUSIVE
PUBLIC mCode
mCode="   "                  && Drie spaties
DO Opruimen
* Einde hoofdprogramma

PROCEDURE Opruimen
SET PROCEDURE TO PROGRAM(1) ADDITIVE
DEFINE FORM Schonen FROM 2,2 TO 10,40
DEFINE ENTRYFIELD Code OF Schonen AT 2,15;
  PROPERTY Datalink "mCode",;
  Width 10
DEFINE TEXT Code2 OF Schonen AT 4, 6;
  PROPERTY Text "Voer Regio-code in",;
  Width 40
DEFINE PUSHBUTTON Afsluiten OF Schonen AT 6, 13;
  PROPERTY;
    TEXT  "Schonen",;
    OnClick EchtWeg,;
    Width 15,;
    Height 1
OPEN FORM Schonen

* Procedure voor knop
PROCEDURE EchtWeg
CLOSE FORMS Schonen
```

```
DELETE FOR Regio = UPPER(mCode)
PACK
COUNT FOR Regio= UPPER(mCode) TO Contr
IF Contr = 0
  ? "Schonen voltooid."
ENDIF
RETURN
```

Zie ook

DELETE, RECALL, SET DELETED, ZAP

PAD( )

dBASE IV-menu's

Resulteert in de naam van de huidige geselecteerde strip of van de laatst geselecteerde strip in een dBASE IV-menubalk. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows kunt u INSPECT( ) gebruiken om informatie op te halen over objecten in formulieren.

Zie Help voor meer informatie over de syntaxis van PAD( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

PADPROMPT( )

dBASE IV-menu's

Resulteert in de prompt van de huidige, geselecteerde strip of van de laatst geselecteerde strip in een dBASE IV-menubalk. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows kunt u INSPECT( ) gebruiken om informatie op te halen over objecten in formulieren.

Zie Help voor meer informatie over de syntaxis van PADPROMPT( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

PARAMETERS

Programma's

Kent variabelenamen toe aan gegevens die worden doorgegeven vanuit een aanroepend programma. Dit commando wordt hoofdzakelijk ondersteund vanwege de compatibiliteit met eerdere versies van dBASE, waarin alleen variabelenamen konden worden toegewezen in procedures met PARAMETERS. In dBASE voor Windows kunt u variabelenamen toewijzen door deze tussen haakjes op te nemen achter PROCEDURE. Op deze manier wordt aan de variabele een lokaal bereik toegewezen. Zie PROCEDURE voor meer informatie.

PAYMENT( )

Resulteert in de termijnbetaling voor een schuld.

Syntaxis

PAYMENT(*<hoofdsom Nuitdr>*, *<rente Nuitdr>*, *<termijnen Nuitdr>*)

<hoofdsom Nuitdr>
Het oorspronkelijke bedrag van de schuld.

<rente Nuitdr>
De rente per periode als een positief decimaal getal. Denk er aan voor het rentepercentage dezelfde periode te gebruiken als voor de termijnen.

<termijnen Nuitdr>
Het aantal betalingen. Denk er aan voor de termijnen dezelfde periode te gebruiken als voor het rentepercentage.

Beschrijving

Met PAYMENT( ) kunt u de termijnbetaling berekenen voor een lening of investering van *<hoofdsom Nuitdr>* in *<termijnen Nuitdr>* termijnbetalingen. PAYMENT( ) geeft een zwevende waarde als resultaat op basis van samengestelde interest met een vast rentepercentage gedurende een vaste looptijd.

Als *<hoofdsom Nuitdr>* positief is, geeft PAYMENT( ) een positief getal als resultaat. Als *<hoofdsom Nuitdr>* negatief is, geeft PAYMENT( ) een negatief getal als resultaat.

Geef het rentepercentage op in de vorm van een decimaal getal. Als de rente op jaarbasis bijvoorbeeld 9,5% is, is *<rente Nuitdr>* gelijk aan 0,095 (9,5/100) voor jaarbetalingen.

Denk er aan voor *<rente Nuitdr>* en *<termijnen Nuitdr>* dezelfde periode te gebruiken. Als de betalingen bijvoorbeeld maandelijks geschieden, moet u de rente per maand en het aantal maandtermijnen opgeven. Een rente op jaarbasis van 9,5% geeft u bijvoorbeeld op als ,095/12 (9,5/100, gedeeld door 12 maanden).

Het resultaat van PAYMENT( ) wordt als volgt berekend:

$$\text{Betaling} = \text{hoofdsom} * \frac{\text{rente} * (1 + \text{rente})^{\text{termijn}}}{(1 + \text{rente})^{\text{termijn}} - 1}$$

waarbij rente = rentepercentage / 100

U leent bijvoorbeeld een bedrag van F 16860,68 en betaalt dat bedrag terug in vijf jaar tegen een rente van 9% per jaar. De maandelijkse termijnbetaling berekent u dan als volgt:

```
? PAYMENT(16860.68,.09/12,60)       && Resulteert in 350.00
nTijd = (1 + .09/12)^60
? 16860.68*(.09/12*nTijd)/(nTijd-1) && Resulteert in 350.00
```

Het aantal decimalen stelt u in met SET DECIMALS.

Voorbeeld

In het volgende voorbeeld wordt een formulier gemaakt waarop de gebruiker informatie kan invoeren over een hypotheek. De maandelijkse termijnen worden berekend met PAYMENT( ):

```
LOCAL f
f=NEW Hypotheek()
f.OPEN()
CLASS Hypotheek OF FORM
  this.Top=5
  this.Left=50
  this.Width=55
  this.Height=13
  this.Text="Hypotheken goed bekeken"
  DEFINE TEXT Tekst1 OF THIS AT 2,5;
    PROPERTY Text "Wat is de hoofdsom?",;
    Width 26, ColorNormal "RB/W"
  DEFINE ENTRYFIELD Hoofdsom OF THIS AT 2,35;
    PROPERTY Value 0, ;
    Width 8
  DEFINE TEXT Tekst2 OF THIS AT 4,5;
    PROPERTY Text "Hoeveel bedraagt de rente?",;
    Width 35, ColorNormal "RB/W"
  DEFINE ENTRYFIELD Rente OF THIS AT 4,35 ;
    PROPERTY Value 0, Picture "99.99",;
    Width 8
  DEFINE TEXT Tekst3 OF THIS AT 6,5;
    PROPERTY Text "Wat is het aantal termijnen?",;
    Width 50, ColorNormal "RB/W"
  DEFINE ENTRYFIELD Termijn OF THIS AT 6,35 ;
    PROPERTY Value 0, Picture "999",;
    Width 8
  DEFINE PUSHBUTTON Resultaat OF THIS AT 10,18;
    PROPERTY Text "Termijnbetaling",;
    OnClick {;mijnResultaat="De termijnbetaling bedraagt: F " + ;
    LTRIM(STR(PAYMENT(Form.Hoofdsom.Value,;
    (Form.Rente.Value/100)/12,Form.Termijn.Value),13,2));
    ; Form.FW.Text = mijnResultaat},;
    Height 2, ColorNormal "N/W", Width 17
  DEFINE TEXT FW OF THIS AT 8,5;
    PROPERTY Text "Uw betaling: ", Width 40,;
    ColorNormal "R+/W"
ENDCLASS
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FV( ), PV( ), SET DECIMALS

PCOL( )

Afdrukken

Resulteert in de kolompositie van de afdrukkop van een printer. De kolomnummers beginnen bij 0.

Syntaxis

PCOL( )

Beschrijving

Met PCOL( ) kunt u de horizontale afdrukpositie van een printer bepalen. dat wil zeggen, de kolom waar de printer begint met afdrukken. Gebruik PCOL( ) in rekenkundige uitdrukkingen als u met printen wilt beginnen op een positie relatief ten opzichte van de huidige kolompositie. De uitdrukking PCOL( ) + 5 stelt bijvoorbeeld een positie voor die vijf kolommen rechts van de huidige positie ligt en PCOL( ) - 5 een positie vijf kolommen links van de huidige positie.

Als u uitvoer naar de printer stuurt, worden de tekens geplaatst op basis van het *coördinatenvlak*, een tweedimensionaal raster. Het coördinatenvlak bestaat uit tekencellen waarvan de breedte afhankelijk is van de waarde van _ppitch. Raadpleeg de beschrijving bij _ppitch voor een tabel met de waarden van _ppitch. De hoogte van elke tekencel wordt bepaald door het font van het hoofdvenster. Zie Hoofdstuk 16 in *Programmeren* voor meer informatie over het coördinatenvlak.

PCOL( ) geeft een kolomnummer als resultaat dat de huidige waarde van _ppitch voorstelt, of u nu afdrukt met een proportioneel of een niet-proportioneel font. Als u afdrukt met een proportioneel font, kunt u ook niet gehele waarden aftrekken van en optellen bij het resultaat van PCOL( ) om de afdrukpositie nauwkeurig te bepalen. Als u ? gebruikt zonder de optie STYLE en alleen gehele getallen als coördinaten gebruikt, wordt een niet-proportioneel font gebruikt en ziet de uitvoer er net zo uit als in dBASE IV.

PCOL( ) resulteert alleen in de kolompositie als SET PRINTER is ingeschakeld (ON) of als SET DEVICE TO PRINTER is gebruikt, anders wordt 0 als resultaat gegeven.

Voorbeeld

In het volgende voorbeeld wordt "Romeo & Julia" naar de printer gestuurd. Drie keer wordt met PCOL( ) de kolompositie bepaalt: aan het begin, na "Romeo" en na "Julia":

```
SET TALK OFF
SET PRINTER ON
* Commando ? wordt naar de printer gestuurd
?          && Printer wordt ingesteld op kolom 0 van volgende regel
BeginPos=pcol()  && Huidige kolompositie bepalen
?? "Romeo"
```

```
RomeoPos=pcol()
?? " & Julia"
JuliaPos=pcol()
SET PRINTER OFF
CLOSE PRINTER
? BeginPos    &&  0,00
? RomeoPos    &&  5,00
? JuliaPos    && 13,00
SET TALK ON
```

Zie ook

COL( ), PROW( ), SET DEVICE, SET PCOL, SET PRINTER

PCOUNT( )

Programma's

Resulteert in het aantal paramaters dat door een aanroep is doorgegeven aan het aangeroepen programma, de aangeroepen procedure of de aangeroepen door de gebruiker gedefinieerde functie.

Syntaxis

PCOUNT( )

Beschrijving

Omdat u aan een programma, procedure of door de gebruiker gedefinieerde functie meer of minder parameters kunt doorgeven dan zijn opgegeven in de instructie PARAMETERS <*parameterlijst*>, kunt u PCOUNT( ) gebruiken om te bepalen hoeveel parameters precies zijn doorgegeven.

PCOUNT( ) resulteert in 0 als geen parameters zijn doorgegeven.

Voorbeeld

In het volgende voorbeeld wordt PCOUNT( ) gebruikt om te bepalen hoeveel parameters zijn doorgegeven aan een procedure.

```
Param1 = 'Hallo'
Param2 = 100
DO MijnProg WITH Param1, Param2

PROCEDURE MijnProg
PARAMETERS Par1, Par2, Par3, Par4
? "U hebt aan deze procedure " + ;
   LTRIM(STR(PCOUNT())) + " parameters doorgegeven."
RETURN
```

Overdraagbaarheid

PCOUNT( ) wordt niet ondersteund in dBASE III PLUS.

Zie ook

DO, FUNCTION, PARAMETERS, PROCEDURE, SET PROCEDURE

PI( )

Numerieke gegevens

Resulteert in een benadering van de waarde van pi. Pi is de verhouding tussen de straal en de omtrek van een cirkel.

Syntaxis

PI( )

Beschrijving

PI( ) resulteert in een zwevende waarde die ongeveer gelijk is aan 3,141592653589793. Pi is een constante. Met pi kunt u het oppervlak en de omtrek van een cirkel of de inhoud van een kegel of cilinder bepalen. Ook kunt u pi gebruiken in de trigonometrische functies SIN( ), COS( ) en TAN( ). Deze functies vereisen dat de waarde voor de hoek wordt opgegeven in radialen. Pi radialen is gelijk aan 180 graden.

Het aantal decimalen stelt u in met SET DECIMALS.

Voorbeeld

In het volgende voorbeeld wordt PI( ) gebruikt in een programma dat enkele maten van een bubbelbad berekent:

```
SET TALK OFF
CLEAR
diameter = 0
diepte = 0
@ 12,12 SAY "Wat is de diameter van het bubbelbad? " ;
   GET diameter PICTURE "99"
@ 13,12 SAY ;
   "Wat is de diepte van het bubbelbad? " ;
   GET diepte PICTURE "99"
READ
@ 15,12 SAY "De omtrek van het bubbelbad is " + ;
   LTRIM(STR((2 * PI() * diameter/2))) + " meter"
@ 16,12 SAY "De inhoud van het bubbelbad is " + ;
   STR(PI() * ((diameter/2)^2) * diepte,5,2) + ;
   " kubieke meter"
@ 17,12 SAY "Het oppervlak van het bubbelbad is " + ;
   STR(PI() * ((diameter/2)^2),5,2) + ;
   " vierkante meter"
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. In dBASE IV resulteert PI( ) in een waarde die nauwkeurig is tot op 14 cijfers achter de komma, onafhankelijk van de instelling voor SET DECIMALS.

Zie ook

COS( ), DTOR( ), RTOD( ), SET DECIMALS, SIN( ), TAN( )

PLAY SOUND

Objecten

Speelt geluid af dat is opgeslagen in een .WAV-bestand of een binair veld.

Syntaxis

PLAY SOUND
  FILENAME
    <bestandsnaam> | ? | *<bestandsnaamfilter>* |
  BINARY *<binair veld>*

FILENAME *<bestandsnaam>* | ? | *<bestandsnaamfilter>* |
BINARY *<binair veld>*

Geeft het geluidsbestand of binaire veld op. PLAY SOUND FILENAME ? en PLAY SOUND FILENAME *<bestandsnaamfilter>* tonen een venster waarin de gebruiker een geluidsbestand kan kiezen. *<bestandsnaam>* is de naam van het geluidsbestand. Als u geen extensie opgeeft, wordt de extensie .WAV gebruikt. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. PLAY SOUND BINARY *<binair veld>* speelt het geluid af dat is opgeslagen in een binair veld.

Beschrijving

Met PLAY SOUND kunt u geluiden afspelen die u opneemt met de Geluidsrecorder van Windows of met andere geluidstoepassingen voor Windows. Het geluid kan bestaan uit muziek, spraak of andere geluiden die zijn opgeslagen in binaire velden of .WAV-bestanden.

Opmerking U kunt geluid alleen afspelen als een geluidsaansturingsprogramma en een geluidskaart zijn geïnstalleerd.

Voorbeeld

In het volgende voorbeeld wordt PLAY SOUND gebruikt in een codeblok voor het kenmerk OnClick van de knop Geluid om een geluid af te spelen dat is opgeslagen in een binair veld:

```
LOCAL f
f = NEW BFORM()
f.Open()

CLASS BFORM OF FORM
   this.HelpFile = ""
   this.Width =         63.40
   this.Maximize = .F.
   this.Minimize = .F.
   this.Height =        12.76
```

```
this.Left =          35.00
this.Text = "Afbeeldingen"
this.Top =           5.65
this.ColorNormal = "BG/B"
this.OnOpen = {;create session}
this.View = "AFBEELD.QBE"
this.HelpId = "" .

DEFINE PUSHBUTTON SOUND OF THIS;
    PROPERTY;
      Width          18.00,;
      Default .T.,;
      OnClick {;PLAY SOUND Binary Afbeeld->Geluid},;
      Height           3.00,;
      Left          22.00,;
      Text "Geluid",;
      Top           4.00,;
      ColorNormal "N/W",;
      FontName "Courier",;
      FontSize          16.00

ENDCLASS
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

SET PATH TO

POPUP( )

dBASE IV-menu's

Resulteert in de naam van het huidige dBASE IV popup-menu. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows kunt u INSPECT( ) gebruiken om informatie op te halen over objecten in formulieren.

Zie Help voor meer informatie over de syntaxis van POPUP( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

PRINTJOB...ENDPRINTJOB

Afdrukken

Gebruikt de waarden die zijn opgeslagen in systeemgeheugenvariabelen voor het besturen van een afdrukbewerking.

Syntaxis

PRINTJOB
    <opdrachten>
ENDPRINTJOB

<opdrachten>
Elke combinatie van geldige dBASE-instructies.

Beschrijving

Gebruik PRINTJOB...ENDPRINTJOB om een afdrukbewerking te besturen met de waarden van de systeemgeheugenvariabelen _pbpage, _pepage, _pcopies, _peject en _plineno. Als de uitvoering van PRINTJOB begint, worden de volgende handelingen uitgevoerd:

1 Het huidige afdrukdocument (als dat er is) wordt gesloten en er wordt een nieuw afdrukdocument begonnen (alsof CLOSE PRINTER is gebruikt voor PRINTJOB werd gebruikt)

2 Als voor _peject "BEFORE" of "BOTH" is ingesteld, wordt de huidige pagina in de printer doorgevoerd

3 _pcolno wordt gelijk aan 0 gemaakt

Als ENDPRINTJOB wordt bereikt, worden de volgende handelingen uitgevoerd:

1 Als voor _peject "AFTER" of "BOTH" is ingesteld, wordt de huidige pagina in de printer doorgevoerd

2 _pcolno wordt gelijk aan 0 gemaakt

Voordat u PRINTJOB...ENDPRINTJOB gebruikt, moet u de toepasselijke systeemgeheugenvariabelen instellen en de instructie SET PRINTER ON opgeven. Na ENDPRINTJOB moet u CLOSE PRINTER gebruiken om het document te sluiten en af te drukken.

Voorbeeld

In het volgende voorbeeld wordt PRINTJOB gebruikt om drie exemplaren af te drukken van één tekstregel:

```
_pcopies=3          && 3 exemplaren
_peject="none"      && geen paginadoorvoer voor of na
_plineno=0          && beginwaarde is 0
SET PRINTER ON
PRINTJOB
? "Een afdruktaak van één regel"
?
ENDPRINTJOB
CLOSE PRINTER       && Begin afdrukken
* Drukt af:
* Een afdruktaak van één regel
*
* Een afdruktaak van één regel
```

```
*
* Een afdruktaak van één regel
*
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

_pbpage, _pcopies, _peject, _pepage, _plineno, EJECT, EJECT PAGE, ON PAGE, SET PRINTER

PRINTSTATUS( ) — Afdrukken

Resulteert in waar (.T.) als het afdrukapparaat gereed is voor de ontvangst van uitvoer.

Syntaxis

PRINTSTATUS([<*poortnaam Tuitdr*>])

<poortnaam Tuitdr>

Een tekenuitdrukking die aangeeft welke printerpoort moet worden gecontroleerd, bijvoorbeeld "LPT1". Geldige poortnamen omvatten alle toewijzingen die geldig zijn in het Configuratiescherm van Windows.

Beschrijving

Met PRINTSTATUS( ) kunt u bepalen of met SET PRINTER TO <*poortnaam Tuitdr*> een printerpoort is ingesteld als uitvoerapparaat. Het afdrukken wordt bestuurd door Afdrukbeheer van Windows. Afdrukbeheer geeft een melding als de printer niet in staat is uitvoer te ontvangen.

Als u geen <*poortnaam Tuitdr*> doorgeeft aan PRINTSTATUS( ), wordt de standaardpoort gecontroleerd die u hebt opgegeven met SET PRINTER TO. PRINTSTATUS( ) geeft alleen .F. als resultaat als geen printerpoort is opgegeven met SET PRINTER TO of als de poort die u opgeeft, niet is ingesteld met SET PRINTER TO.

Opmerking SET PRINTER TO wordt automatisch uitgevoerd bij het opstarten als het bestand WIN.INI een geldige printerdefinitie bevat. Raadpleeg het handboek van Windows voor meer informatie over de instellingen in WIN.INI.

Voorbeeld

In het volgende voorbeeld wordt eerst de status van de standaardprinter en vervolgens van LPT1, LPT2 en LPT3 weergegeven:

```
? PRINTSTATUS()
? PRINTSTATUS("LPT1")
```

```
? PRINTSTATUS("LPT2")
? PRINTSTATUS("LPT3")
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CLOSE..., SET DEVICE, SET PRINTER

PRIVATE

Geheugenvariabelen

Met PRIVATE gedefinieerde geheugenvariabelen zijn beschikbaar in het programma waarin deze zijn gedefinieerd en in alle subroutines die door dat programma worden aangeroepen. PRIVATE wordt hoofdzakelijk ondersteund vanwege de compatibiliteit met dBASE IV. In dBASE voor Windows wordt u geadviseerd variabelen te definiëren met LOCAL. Met LOCAL gedefinieerde variabelen zijn uitsluitend beschikbaar in de routine (het programma, de subroutine of de door de gebruiker gedefinieerde functie) waarin deze zijn gedefinieerd. Zowel variabelen die met LOCAL zijn gedefinieerd als variabelen die met PRIVATE zijn gedefinieerd, worden lokale variabelen genoemd.

Syntaxis

PRIVATE <*variabelenlijst*> |
ALL
[LIKE <*variabelefilter1*>]
[EXCEPT <*variabelefilter2*>]

<*variabelenlijst*>

De lijst van geheugenvariabelen die u met PRIVATE wilt definiëren. De variabelen worden van elkaar gescheiden met komma's.

ALL

Zorgt dat alle geheugenvariabelen in de subroutine met PRIVATE worden gedefinieerd.

LIKE <*variabelefilter1*>

Definieert alle geheugenvariabelen waarvan de namen voldoen aan het geheugenvariabelenschema dat u opgeeft met <*variabelefilter1*>, met PRIVATE. Gebruik tekens van de namen van de variabelen en de jokers * en ? om <*variabelefilter1*> te definiëren.

EXCEPT <*variabelefilter2*>

Definieert alle geheugenvariabelen met PRIVATE, behalve die waarvan de namen voldoen aan het geheugenvariabelenschema dat u opgeeft met <*variabelefilter2*>. Gebruik tekens van de namen van de variabelen en de jokers * en ? om <*variabelefilter2*>

te definiëren. U kunt LIKE en EXCEPT in dezelfde instructie gebruiken, zoals PRIVATE ALL LIKE gvar* EXCEPT gvarnum*.

Beschrijving

Gebruik PRIVATE om een geheugenvariabele te definiëren die alleen beschikbaar is in de subroutine waarin de variabele wordt gedefinieerd en alle subroutines die door die routine worden aangeroepen. De variabele is dus niet beschikbaar voor routines op een hoger niveau. Als u een variabele definieert met PRIVATE, heeft dat niet onmiddellijk tot gevolg dat die variabele wordt gemaakt. Als de variabele eenmaal is gedefinieerd, kunt u die maken en initialiseren met STORE, = of DECLARE.

Een met PRIVATE gedefinieerde variabele is niet beschikbaar in routines die hoger liggen in de aanroepreeks. Op die manier kan de waarde van de variabele niet worden veranderd door hoger in de aanroepreeks gelegen routines die variabelen met dezelfde naam bevatten. Een met PRIVATE gedefinieerde variabele is een andere variabele dan een variabele met dezelfde naam in een routine op een hoger niveau of een onafhankelijke routine. Als een lokale variabele eenmaal een waarde heeft gekregen, geven DISPLAY MEMORY en LIST MEMORY een variabele met dezelfde naam in een routine op een hoger niveau aan als verborgen.

Variabelen die u definieert in programma's zijn standaard PRIVATE. Als u echter een variabele definieert die dezelfde naam heeft als een met PUBLIC gedefinieerde variabele in een subroutine op een hoger niveau of in het commandovenster en die variabele niet definieert met PRIVATE, wordt die variabele *niet* als een lokale variabele gemaakt. In plaats daarvan gebruikt de subroutine de waarde van de bestaande met PUBLIC gedefinieerde variabele.

Een met PRIVATE gedefinieerde variabele wordt niet beschermd tegen wijziging door een subroutine op een lager niveau die een variabele met dezelfde naam bevat. Als een subroutine op een lager niveau een variabele initialiseert of wijzigt die dezelfde naam heeft als een variabele die op een hoger niveau wordt gebruikt, wordt de waarde van de variabele op het hogere niveau overschreven.

U kunt dit voorkomen door u de variabele in de lagere subroutine te definiëren met PRIVATE (of variabelen te definiëren met LOCAL). De variabelen worden dan ondanks de identieke namen beschouwd als verschillende variabelen. Als u wilt dat een subroutine op een lager niveau toegang heeft tot de waarde van een variabele, maar niet in staat is een gewijzigde waarde weer omhoog door te geven, moet u die variabele tussen haakjes opnemen in een instructie met DO...WITH.

Met PRIVATE gedefinieerde variabelen worden uit het geheugen verwijderd als de subroutine waarin deze zijn gedefinieerd, is voltooid.

Bij PUBLIC vindt u een tabel waarin het bereik en de beschikbaarheid worden vergeleken van variabelen die zijn gedefinieerd met PUBLIC, PRIVATE, LOCAL en STATIC.

Voorbeeld

In het volgende voorbeeld wordt vanuit een hoofdprogramma naar verschillende subroutines op lagere niveaus gesprongen. Het voorbeeld laat zien dat een in Niveau_2

met PRIVATE gedefinieerde variabele beschikbaar is in alle procedures onder het niveau waarop die is gedefinieerd, maar niet meer beschikbaar is als de besturing weer wordt teruggegeven aan het niveau van hoofdprogramma:

```
* **Hoofd.Prg***
CLOSE ALL
CLEAR ALL
CLEAR
SET TALK OFF
? "***Hoofd.PRG**"
STATIC nTotaal
PUBLIC tReeks
nTotaal = 7109.50
tReeks= "Hallo"
ON ERROR ? "Var niet beschikbaar";
                    && Melding tonen bij fout
? nTotaal           && Geeft 7109,50 als resultaat
? tReeks            && Geeft "Hallo" als resultaat
DO Niveau_2         && Spring naar Proc Niveau_2
?
? "**Terug naar Hoofd.PRG**"
? Deadline          && "Var niet beschikbaar" boven ;
                       Proc Niveau_2

PROCEDURE Niveau_2
?
? "**Proc Niveau_2**" && Ter oriëntatie
PRIVATE Deadline    && Var gedefinieerd met PRIVATE ;
                       in Procedure Niveau_2
? Deadline          && "Var niet beschikbaar" ;
                       nog niet geïnitialiseerd
Deadline = {31-12-99} && Variabele initialiseren
? Deadline          && Variabele heeft nu waarde
DO Niveau_3         && Spring naar Niveau_3
?
? "**Terug naar Niveau_2**"
? Deadline          && 31-12-99 nog beschikbaar
RETURN              && Terug naar regel na ;
                       DO Niveau_2 in Hoofd

PROCEDURE Niveau_3
?
? "**Proc Niveau_3**"  && Ter oriëntatie
? tReeks               && "Hallo" gedefinieerd met PUBLIC
? nTotaal              && "Var niet beschikbaar" ;
                          gedefinieerd met STATIC in Hoofd
? Deadline             && 31-12-99 is beschikbaar in;
                          Procs onder Niveau_2
Do Niveau_4            && Spring naar Niveau_4
?
? "**Terug naar Niveau_3**"
? Deadline          && 31-12-99 nog steeds beschikbaar
RETURN              && Terug naar regel na ;
                       DO Niveau_3 in Niveau_2

PROCEDURE Niveau_4
```

```
?
? "**Proc Niveau_4**" && Ter oriëntatie
? nTotaal             && "Var niet beschikbaar" ;
                         gedefinieerd met STATIC in Hoofd
? tReeks              && "Hallo" - PUBLIC variable
? Deadline            && 31-12-99 nog steeds beschikbaar;
                         in Procs onder Niveau_2
RETURN                && Terug naar regel na ;
                         DO Niveau_4 in Niveau_3
```

Overdraagbaarheid

dBASE IV en dBASE III PLUS ondersteunen LIKE en EXCEPT niet in één instructie met PRIVATE.

Zie ook

CLEAR MEMORY, DECLARE, LOCAL, PUBLIC, STATIC, STORE

PROCEDURE

Programma's

Definieert een procedure in een programmabestand en definieert optioneel geheugenvariabelen voor parameters die aan de procedure worden doorgegeven.

Syntaxis

PROCEDURE <*procedurenaam*> [(<*parameterlijst*>)]
  [<*instructies*>]
  [RETURN [<*resultaatwaarde*>]]

<*procedurenaam*>

De naam van de procedure. Hoewel dBASE geen beperkingen oplegt aan de lengte van procedurenamen, worden alleen de eerste 32 tekens werkelijk gebruikt. Procedurenamen mogen letters, cijfers en onderstrepingstekens bevatten.

(<*parameterlijst*>)

De geheugenvariabelenamen die worden toegewezen aan de gegevens (of *parameters*) die aan de procedure worden doorgegeven door de aanroepende instructie. De variabelen in <*parameterlijst*> hebben een lokaal (LOCAL) bereik, zodat ze zijn beschermd tegen wijziging in subroutines op een lager niveau. Zie LOCAL voor meer informatie over lokaal bereik.

Het aantal toegewezen variabelen kan afwijken van het aantal doorgegeven parameters. Met PCOUNT( ) kunt u het aantal parameters bepalen dat aan een procedure is doorgegeven. U kunt maximaal 255 variabelenamen opnemen in <*parameterlijst*>.

Opmerking Procedures die zijn geschreven in eerdere versie van dBASE, kunnen variabelenamen bevatten die zijn gedefinieerd met PARAMETERS. Dit wordt ondersteund in dBASE voor Windows vanwege de compatibiliteit met oudere versies, maar wordt afgeraden

vanwege het feit dat variabelen die op deze manier zijn gedefinieerd hetzelfde bereik hebben als PRIVATE-variabelen en niet als LOCAL-variabelen. Variabelen die met PRIVATE zijn gedefinieerd (en dus ook variabelen die met PARAMETERS zijn gedefinieerd) kunnen worden overschreven als een subroutine die door de procedure wordt aangeroepen, variabelen bevat die dezelfde naam hebben. Zie PRIVATE voor meer informatie over het bereik van dit soort variabelen.

<instructies>

Elke reeks geldige programma-instructies die u door de procedure wilt laten uitvoeren. Dit kunnen toekenningsinstructies, commando's, procedure-aanroepen en functie-aanroepen zijn. Het is toegestaan procedures recursief aan te roepen.

RETURN [*<resultaatwaarde>*]

Geeft de programmabesturing en de door *<resultaatwaarde>* gedefinieerde waarde terug aan de aanroepende instructie. Als u RETURN achterwege laat, wordt de procedure beëindigd als het eind van het bestand wordt bereikt of als een instructie wordt aangetroffen met PROCEDURE, CLASS of FUNCTION. Als u de resultaatwaarde van de procedure wilt gebruiken, zoals in ? MijnProc( ), moet u RETURN *<resultaatwaarde>* in de instructie opnemen.

Beschrijving

Met PROCEDURE kunt u een subroutine definiëren die bepaalde commando's uitvoert en optioneel een waarde als resultaat geeft. Met procedures kunt u de dBASE-taal uitbreiden met de functionaliteit die u in uw applicaties nodig hebt. Ook kunt u met procedures een meer modulaire code maken, die gemakkelijk te lezen en te onderhouden is en waarin u beter fouten kunt opsporen.

Als u bijvoorbeeld op verschillende plaatsen in een applicatie dezelfde taak wilt uitvoeren, kunt u die in een procedure plaatsen, zoals in het volgende voorbeeld. In dit geval kunt u elke keer dat u een record wilt overslaan of naar het begin van het bestand wilt gaan als het eind van het bestand is bereikt, de procedure VolgRecOfTop aanroepen:

```
PROCEDURE VolgRecOfTop
    SKIP       && Indien mogelijk naar volgende record
   IF EOF()    && Einde van bestand?
      GO TOP   && Dan naar eerste record in de tabel
   ENDIF
RETURN
```

Gebruik PROCEDURE alleen in een programmabestand (.PRG of .WFM).Het is niet mogelijk procedures te nesten of een procedure te starten in een lus die is gedefinieerd met IF, SCAN, enzovoort.

U roept een procedure aan met DO of met de aanroep-operator (haakjes):

```
** Twee manieren om dezelfde procedure aan te roepen
x = 100
DO EersteProc WITH x
? x                && geeft 200 als resultaat
x = 100
? EersteProc(x)    && geeft 200 als resultaat
```

```
PROCEDURE EersteProc(gvar)
gvar = gvar * 2
RETURN gvar
```

Werken met parameters

Gebruik parameters om waarden en verwijzingen door te geven tussen een subroutine en de aanroepende subroutine. Als u bijvoorbeeld regelmatig een datum nodig hebt die 30 dagen voorbij een andere datum ligt, kunt u een procedure maken om de gewenste datum te berekenen en als resultaat te geven op basis van de eerste datum die als parameter aan de procedure is doorgegeven. De aangeroepen procedure maakt een naam van een LOCAL-variabele die de waarde van de doorgegeven parameter voorstelt, voert de berekening uit en geeft de waarde voor de berekende datum als resultaat.

Dit wordt gedemonstreerd in het volgende voorbeeld. Telkens wanneer u 30 dagen moet optellen bij een datum, roept u Dertig( ) aan met de datum als parameter. In dit voorbeeld is Begindatum de naam die de aangeroepen procedure aan de parameter toewijst.

```
SET DATE ITALIAN
dStartDatum = {03-12-95}
? Dertig(dStartDatum)  && geeft waarde als resultaat in einddatum
PROCEDURE Dertig(startdatum)
* startdatum is de variabelenaam die wordt toegewezen
* aan de waarde in dStartDatum
einddatum = startdatum + 30
RETURN einddatum
```

U kunt geheugenvariabelen, kenmerken, array-namen en array-elementen, velden, vaste waarden en uitdrukkingen als parameters doorgeven aan procedures.

Geheugenvariabelen en kenmerken doorgeven

Geheugenvariabelen en kenmerken kunt u op twee manieren doorgeven: *als referentieparameter* en *als waarde*. In deze sectie heeft de term "geheugenvariabele" zowel betrekking op geheugenvariabelen als op kenmerken.

- Als u geheugenvariabelen doorgeeft als referentieparameters, heeft de aangeroepen procedure rechtstreeks toegang tot de variabele. De procedure kan de waarde van de variabele wijzigen (overschrijven). Geef variabelen door als referentieparameters als u wilt dat de aangeroepen procedure de waarden kan wijzigen die zijn opgeslagen in de als parameters doorgegeven geheugenvariabelen.
- Als u geheugenvariabelen doorgeeft als waarden, heeft de aangeroepen procedure alleen toegang tot de waarde die in de variabele is opgeslagen. De procedure kan niet de inhoud van de variabele zelf wijzigen. Geef variabelen door als waarden als u wilt dat een procedure de waarde van de variabele wel kan gebruiken, maar niet de waarde van de variabele in de aanroepende procedure kan wijzigen. Als u een geheugenvariabele wilt doorgeven als waarde, plaatst u die tussen haakjes.

In het volgende voorbeeld wordt het verschil gedemonstreerd tussen het doorgeven van geheugenvariabelen als referentieparameter en als waarde:

```
X=10
? X                    && X = 10
Do MijnProc WITH X     && 10 doorgegeven als referentieparameter
? X                    && X is nu 11 (gewijzigd door MijnProc)
DO MijnProc WITH (X) && 11 doorgegeven als waarde
? X                    && X is nog steeds 11
                       && (ongewijzigd door MijnProc)
PROCEDURE MijnProc(N)  && N stelt de waarde in X voor
? N          && eerste aanroep: 10 als resultaat
             && tweede aanroep: 11 als resultaat
N=N+1
? N          && eerste aanroep: 11 als resultaat
             && tweede aanroep: 12 als resultaat
RETURN
```

Veldnamen hebben een hogere prioriteit dan variabelenamen. Als een veld en een variabele dezelfde naam hebben en u wilt de variabele doorgeven als parameter, gebruikt u M-><*variabelenaam*> om aan te geven dat u de geheugenvariabele wilt gebruiken. Als u bijvoorbeeld beschikt over een veld met de naam Klantnr en een variabele met de naam Klantnr, geeft u de variabele als volgt door aan de procedure ToonKlant:

```
DO ToonKlant WITH M->Klantnr
```

Array's doorgeven

Als u een array als parameter wilt gebruiken, kunt u de gehele array doorgeven (door alleen de arraynaam door te geven) of afzonderlijke array-elementen doorgeven.

- Het doorgeven van een volledige array of een array-element werkt hetzelfde als het doorgeven van een geheugenvariabele.
- Als u de array-naam zonder haakjes doorgeeft, wordt de array doorgegeven als referentieparameter. De aangeroepen routine kan waarden in de array wijzigen.
- Als u de array-naam tussen haakjes plaatst, wordt de array doorgegeven als waarde. De aangeroepen routine kan geen waarden in de array wijzigen.
- Als u een array-element doorgeeft, wordt het altijd doorgegeven als referentieparameter, zelfs als u het tussen haakjes plaatst. De aangeroepen routine kan de waarde wijzigen.

Velden doorgeven

In tegenstelling tot geheugenvariabelen, kunnen velden die worden doorgegeven als parameters niet door de subroutine worden gewijzigd. Velden worden altijd doorgegeven als waarde zodat de aangeroepen procedure de inhoud niet kan wijzigen.

Als u een veld wilt wijzigen, moet u de inhoud opslaan in een geheugenvariabele en de subroutine uitvoeren met die variabele (zoals DO NaamWijz WITH tNwNaam). Als de besturing weer wordt teruggegeven aan het aanroepende programma, kunt u de inhoud van het veld vervangen door de inhoud van de geheugenvariabele (zoals REPLACE Vnaam WITH tNwNaam).

De beschikbaarheid van procedures

U kunt een procedure opnemen in het programmabestand waarin de procedure wordt gebruikt, maar u kunt procedures ook in een afzonderlijke programmabestand plaatsen en beschikbaar maken met SET PROCEDURE of SET LIBRARY. Als u een procedure opneemt in het programmabestand waarin die wordt gebruikt, is het verstandig alle procedures bij elkaar aan het eind van het bestand te plaatsen.

Een programmabestand kan maximaal 193 procedures bevatten. Als u toegang wilt hebben tot meer dan 193 procedures, moet u SET PROCEDURE...ADDITIVE gebruiken. De maximumgrootte van een procedure wordt alleen maar beperkt door de maximumgrootte van een programmabestand.

Als u van plan bent een procedure aan te roepen met het commando DO, zoals in DO DezeProcedure, geeft u de procedure niet dezelfde naam als het programmabestand waarin die is opgeslagen. Als u dat doet, wordt het programma uitgevoerd en niet de procedure, wat tot onvoorspelbare resultaten kan leiden.

Als u van plan bent een procedure aan te roepen met de aanroep-operator (haakjes), zoals in DezeProcedure( ), geeft u de procedure niet dezelfde naam als een standaardfunctie van dBASE. Als u dat wel doet, wordt de dBASE-functie uitgevoerd in plaats van de procedure.

Als u een procedure aanroept, wordt die gezocht in het *zoekpad* en wel in de *zoekvolgorde*. Als meerdere procedures met dezelfde naam beschikbaar zijn, wordt de eerste uitgevoerd die wordt aangetroffen. Let er dus op geen dubbele namen te gebruiken voor procedures. Zie de beschrijving van DO voor een beschrijving van het zoekpad en de zoekvolgorde.

Als u een procedure maakt, kunt u de naam gebruiken om het adres te bepalen (het adres is een waarde van het type "functie-aanwijzer"). Codeblokken bij objecten zijn anonieme procedures. Zie Hoofdstuk 4 in *Programmeren* voor meer informatie over het werken met procedures, codeblokken en functie-aanwijzers.

Voorbeeld

In het volgende voorbeeld worden procedures binnen het hoofdprogrammabestand aangeroepen:

```
* Hoofd.PRG
SET TALK OFF
CLEAR
DO A
DO B
DO C
RETURN

PROCEDURE A
@ 10,2 SAY "PROCEDURE A aan het eind van hoofd.PRG"
RETURN

PROCEDURE B
@ 15,2 SAY "PROCEDURE B aan het eind van hoofd.PRG"
RETURN
```

```
PROCEDURE C
@ 20,2 SAY "PROCEDURE C aan het eind van hoofd.PRG"
RETURN
```

In het volgende voorbeeld worden procedures uit een afzonderlijke procedurebestand aangeroepen:

```
* Hoofd.PRG
SET TALK OFF
CLEAR
SET PROCEDURE TO PROCBEST
DO A
DO B
DO C
RETURN

* PROCBEST.PRG
PROCEDURE A
@ 10,2 SAY "PROCEDURE A in PROCBEST.PRG"
RETURN

PROCEDURE B
@ 15,2 SAY "PROCEDURE B in PROCBEST.PRG"
RETURN

PROCEDURE C
@ 20,2 SAY "PROCEDURE C in PROCBEST.PRG"
RETURN
```

Overdraagbaarheid

In dBASE IV kunnen tegelijkertijd een procedure en een door de gebruiker gedefinieerde functie (gedefinieerd met FUNCTION) met dezelfde naam beschikbaar zijn, omdat deze op verschillende manieren worden aangeroepen. Als in dBASE voor Windows een procedure en een door de gebruiker gedefinieerde functie met dezelfde naam zijn gedefinieerd, is alleen de procedure of functie beschikbaar die het eerst is gedefinieerd.

De volgende tabel geeft een overzicht van de overige verschillen bij het gebruik van PROCEDURE in dBASE III PLUS, dBASE IV en dBASE voor Windows.

	dBASE III PLUS	dBASE IV	dBASE voor Windows
Maximumlengte van procedurenaam	8 tekens	Onbeperkt, maar alleen de eerste 9 tekens worden herkend	Onbeperkt, maar alleen de eerste 32 tekens worden herkend
Geldige tekens in een procedurenaam	Letters, cijfers en onderstrepings-tekens; eerste teken moet een letter zijn	Letters, cijfers en onderstreping-stekens; eerste teken moet een letter of een cijfer zijn	Letters, cijfers en onderstreping-stekens
Maximumaantal procedures per programma-bestand	32 (33 en hoger worden niet ondersteund)	963	193; met SET PROCEDURE...ADDITIVE kunt u meer dan 193 gebruiken
Maximumgrootte van procedures	Onbeperkt	65520 bytes gecompileerde code	Gelijk aan maximum-bestandslengte

	dBASE III PLUS	dBASE IV	dBASE voor Windows
Behandeling van naamloze procedures	Komen niet voor in de procedurelijst	Worden genoemd in de procedurelijst; krijgen standaard de naam van het programma-bestand	Hetzelfde als in dBASE IV
Voortzettingsteken voor regels (;)	Mag niet worden gebruikt tussen het commando PROCEDURE en de procedurenaam	Mag worden gebruikt tussen het commando PROCEDURE en de procedurenaam	Hetzelfde als in dBASE IV

Zie ook

CLASS...ENDCLASS, CLOSE..., COMPILE, DEBUG, DO, FUNCTION, LOCAL, PCOUNT( ), PRIVATE, RETRY, RETURN, SET LIBRARY, SET PROCEDURE

PROGRAM( )

Foutafhandeling en testen op fouten

Resulteert in de naam van het actieve programma, de actieve procedure of de actieve door de gebruiker gedefinieerde functie.

Syntaxis

PROGRAM([<*Nuitdr*>])

<Nuitdr>
Elk willekeurig getal.

Beschrijving

PROGRAM( ) resulteert in de naam van de actieve subroutine op het laagste niveau. Dat kan een programma, procedure of door de gebruiker gedefinieerde functie zijn. PROGRAM( ) resulteert in een lege tekenreeks ("") als geen programma of subroutine wordt uitgevoerd.

PROGRAM(*Nuitdr*) resulteert in het volledige pad van het actieve programma. Dat kan afwijken van de naam van de actieve subroutine op het laagste niveau, zoals u in het volgende voorbeeld kunt zien:

```
SET PROCEDURE TO program1
** PROGRAM1.PRG bevat de PROCEDURE procedure1
** Dit gebeurt als procedure1 actief is:
? PROGRAM()         && geeft PROCEDURE1 als resultaat
? PROGRAM(Nuitdr)   && geeft C:\DBASEWIN\PROGRAM1.PRG als resultaat
```

Als een programma is opgeschort met SUSPEND, kunt u in het commandovenster PROGRAM( ) gebruiken. Als Programma A bijvoorbeeld Procedure B aanroept, en Procedure B is opgeschort, geeft PROGRAM( ) in het commandovenster de naam van Procedure B als resultaat. Als u in het commandovenster PROGRAM(*Nuitdr*) opgeeft, wordt het volledige pad als resultaat gegeven van het bestand dat Procedure B bevat.

U kunt PROGRAM( ) ook gebruiken in combinatie met ON ERROR en LINENO( ) om te bepalen in welke subroutine en op welke regel een fout is opgetreden.

PROGRAM( ) geeft de naam van de subroutine als resultaat in hoofdletters. PROGRAM( ) toont geen extensie, zelfs niet als de subroutine in een afzonderlijk bestand staat. Het resultaat van PROGRAM(*Nuitdr*) bevat daarentegen altijd een extensie.

Voorbeeld

Zie het voorbeeld bij ON ERROR voor een voorbeeld met PROGRAM( ).

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

DEBUG, LINENO( ), ON ERROR, PROCEDURE, RESUME, SET PROCEDURE, SUSPEND

PROMPT( )

dBASE IV-menu's

Resulteert in de prompt van het huidige geselecteerde menu-element of het laatste geselecteerde menu-element. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows kunt u INSPECT( ) gebruiken om informatie op te halen over objecten in formulieren.

Zie Help voor meer informatie over de syntaxis van PROMPT( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

PROPER( )

Tekenreeksgegevens

Converteert een tekenreeks naar een reeks met de opmaak voor een eigennaam en geeft de geconverteerde reeks als resultaat.

Syntaxis

PROPER(<*Tuitdr*> | <*memoveld*>)

<*Tuitdr*> | <*memoveld*>

De tekenreeks of het memoveld dat moet worden opgemaakt als een eigennaam.

Beschrijving

PROPER( ) resulteert in een tekenreeks waarvan de eerste letter van elk woord in een tekenuitdrukking of memoveld is omgezet in een hoofdletter en alle overige letters in kleine letters. PROPER( ) wijzigt de eerste letter van een woord alleen als dat een kleine letter (alfabetisch teken) is. PROPER( ) geeft maximaal 32766 tekens als resultaat (de maximumlengte van een tekenreeks).

De huidige taalaansturing bepaalt welke tekens hoofdletters en welke kleine letters zijn. Voor speciale tekens bepaalt de taalaansturing tevens in welke hoofdletters kleine letters met accenten worden omgezet, en omgekeerd. In de Nederlandse taalaansturing wordt á (a-accent grave) bijvoorbeeld wel omgezet in Á, maar â (a-circonflex) in de gewone hoofdletter A. Zie Appendix C in *Programmeren* voor een volledige beschrijving van de omzettingsregels in de Nederlandse taalaansturing.

Voorbeeld

In het volgende voorbeeld wordt PROPER( ) gebruikt om tekenreeksen te voorzien van een consequent gebruik van hoofdletters en kleine letters:

```
? PROPER("p. p. prins")     && Geeft "P. P. Prins" als resultaat
? PROPER("p.p. prins")      && Geeft "P.p. Prins" als resultaat
? PROPER("3-WERF JACHTEN") && Geeft "3-werf Jachten" als resultaat
? PROPER("")                && Geeft "" als resultaat
```

Als gegevens in een tekenveld volledig in hoofdletters worden ingevoerd, kunt u de inhoud van het veld permanent omzetten in hoofdletters en kleine letters. In het volgende voorbeeld worden alle bedrijfsnamen in hoofdletters in de tabel Bedrijf omgezet in woorden met een beginkapitaal en verder kleine letters:

```
USE BEDRIJF
REPLACE ALL Bedrijf with PROPER(Bedrijf)
```

PROPER( ) kan ook worden gebruikt voor memovelden. In het volgende voorbeeld wordt de inhoud van alle memovelden in de tabel Contact weergegeven en wordt PROPER( ) gebruikt om alle woorden te voorzien van een beginkapitaal:

```
USE Contact
SCAN
? PROPER(Notities)
?
ENDSCAN
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

LDRIVER( ), LOWER( ), SET LDCHECK, UPPER( )

PROW( )

Resulteert in de rijpositie waar de printer met afdrukken begint. Het eerste rijnummer is 0.

Syntaxis

PROW( )

Beschrijving

Met PROW( ) kunt u de verticale afdrukpositie van de printer bepalen. Dat wil zeggen, de rij waar met afdrukken wordt begonnen. U kunt PROW( ) gebruiken in rekenkundige uitdrukkingen om met afdrukken te beginnen op een positie relatief ten opzichte van de huidige rijpositie. De uitdrukking PROW( ) + 5 stelt bijvoorbeeld een positie voor die vijf rijen onder de huidige positie ligt en PROW( ) - 5 een positie vijf rijen boven de huidige positie.

Als u uitvoer naar de printer stuurt, worden de tekens geplaatst op basis van het *coördinatenvlak*, een tweedimensionaal raster. Het coördinatenvlak bestaat uit tekencellen waarvan de breedte afhankelijk is van de waarde van _ppitch. Raadpleeg de beschrijving bij _ppitch voor een tabel met de waarden van _ppitch. De hoogte van elke tekencel wordt bepaald door het font van het hoofdvenster. Zie Hoofdstuk 16 in *Programmeren* voor meer informatie over het coördinatenvlak.

Als u afdrukt met een proportioneel font, kunt u ook niet gehele waarden aftrekken van en optellen bij het resultaat van PROW( ) om de afdrukpositie nauwkeurig te bepalen. Als u ? gebruikt zonder de optie STYLE en alleen gehele getallen als coördinaten gebruikt, wordt een niet-propotioneel font gebruikt en ziet de uitvoer er net zo uit als in dBASE IV.

PROW( ) resulteert alleen in de rijpositie als SET PRINTER is ingeschakeld (ON) of als SET DEVICE TO PRINTER is gebruikt, anders wordt 0 als resultaat gegeven.

Voorbeeld

In het volgende voorbeeld wordt met PROW( ) de huidige kolompositie bepaald. Vervolgens worden twee rijen daaronder "Hallo" en op de daarop volgende regel enkele resultaten geschreven:

```
SET TALK OFF
SET DEVICE TO PRINTER
EJECT   && prow() is nu 0
Beginrij=prow()
@ prow()+2,0 SAY "Hallo"
* prow() is nu beginrij+2
*   vanwege het commando @SAY
Hallorij=prow()
@ prow()+1,0 SAY "Beginrij="     && naar volgende rij
@ prow(),pcol() SAY Beginrij     && 0,00
@ prow(),pcol() SAY " Hallorij="
@ prow(),pcol() SAY Hallorij     && 2,00
```

```
@ prow(),pcol() SAY " Prow()="
@ prow(),pcol() SAY prow()       && 3,00
SET DEVICE TO SCREEN  && Uitvoer weer naar scherm
CLOSE PRINTER         && Afdrukken starten
```

Zie ook

PCOL( ), ROW( ), SET PROW

PUBLIC — Geheugenvariabelen

Definieert globale geheugenvariabelen of array's die in elk dBASE-programma, elke subroutine of in het commandovenster kunnen worden gewijzigd. Dit worden publieke variabelen genoemd.

Syntaxis

PUBLIC *<variabelenlijst>* |
ARRAY *<array-naam1>*"["*<Nuitdrukkingenlijst1>*"]"
[, *<array-naam2>*"["*<Nuitdrukkingenlijst2>*"]"...]

De teksthaakjes ([ ]) tussen dubbele aanhalingstekens maken verplicht deel uit van de syntaxis.

<variabelenlijst>

De geheugenvariabelen die u met PUBLIC wilt definiëren.

ARRAY *<array-naam1>*[*<Nuitdrukkingenlijst1>*], *<array-naam2>*[*<Nuitdrukkingenlijst2>*] ...

De array-variabelen (*array-naam1*>, *<array-naam2*>, enzovoort) en de array-elementen van elke array (*<Nuitdrukkingenlijst1*>, *<Nuitdrukkingenlijst2*>, enzovoort) die u met PUIBLIC wilt definiëren.

Beschrijving

Met PUBLIC kunt u in een subroutine geheugenvariabelen, inclusief array-variabelen, definiëren die gebruikt kunnen worden in alle routines (van een lager en van een hoger niveau) in een programma en in het commandovenster.

Als de besturing wordt doorgegeven vanuit een *subroutine* (een programma, procedure of door de gebruiker gedefinieerde functie) aan de aanroepende routine op een hoger niveau (een programma, procedure, door de gebruiker gedefinieerde functie of het commandovenster), worden gewoonlijk alle door de subroutine geïnitialiseerde variabelen uit het geheugen verwijderd. Als u een variabele definieert met PUBLIC, voorkomt u dat de variabele uit het geheugen wordt verwijderd als de besturing wordt teruggegeven aan de routine op een hoger niveau.

U moet een variabele eerst met PUBLIC definiëren en pas dan kunt u een waarde toekennen aan die variabele. Als u een variabele definieert met PUBLIC wordt die gemaakt en geïnitialiseerd met .F.

De variabelen die u initialiseert in het commandovenster zijn standaard PUBLIC-variabelen, terwijl variabelen die u initialiseert in programma's standaard LOCAL zijn. In de volgende tabel worden de eigenschappen vergeleken van variabelen die in een subroutine MaakVar zijn gedefinieerd met PUBLIC, PRIVATE, LOCAL en STATIC.

	PUBLIC	PRIVATE	LOCAL	STATIC
Wordt gemaakt en geïnitialiseerd met de waarde .F. als die wordt gedefinieerd	J	N	N	J
Kan worden gebruikt en gewijzigd in MaakVar	J	J	J	J
Kan worden gebruikt en gewijzigd in subroutines op een lager niveau die worden aangeroepen door MaakVar	J	J	N	N
Kan worden gebruikt en gewijzigd in subroutines op een hoger niveau die MaakVar aanroepen	J	N	N	N
Wordt automatisch vrijgegeven als MaakVar wordt beëindigd	N	J	J	N

Als een programma wordt opgeschort, krijgen geheugenvariabelen (inclusief array's) die u initialiseert in andere programma's en in het commandovenster, automatisch dezelfde eigenschappen als LOCAL-variabelen. Als er geen programma is opgeschort, krijgen variabelen die u initialiseert in het commandovenster, automatisch dezelfde eigenschappen als variabelen die met PUBLIC zijn gedefinieerd.

Als een array-variabele wordt gedefinieerd met PUBLIC ARRAY heeft dat dezelfde gevolgen als de instructie PUBLIC *<arraynaam>* gevolgd door DECLARE *<array>*. De eerste regel in het volgende voorbeeld heeft dezelfde gevolgen als de beide laatste regels:

```
PUBLIC ARRAY Rij[5]
PUBLIC Rij
DECLARE Rij[5]
```

Voorbeeld

In het volgende voorbeeld wordt PUBLIC gebruikt om de variabelen Wrd, Wrd1, Wrd2 te definiëren zodat deze beschikbaar zijn in alle aangeroepen programma's en procedures:

```
RELEASE Wrd,Wrd1,Wrd2
PUBLIC  Wrd,Wrd1,Wrd2
? Wrd,Wrd1,Wrd2  &&  .F.  .F.  .F.
Wrd=10
? Wrd,Wrd1,Wrd2  &&  10   .F.  .F.
DO Proc1
? Wrd,Wrd1,Wrd2  &&   1   11   .F.
DO Proc2
? Wrd,Wrd1,Wrd2  &&   2   11   22

PROCEDURE Proc1
Wrd = 1
```

```
Wrd1 = 11
RETURN

PROC Proc2
Wrd =  2
Wrd2 = 22
RETURN
```

Als de instructie met PUBLIC achterwege wordt gelaten, zijn Wrd1 en Wrd2 niet beschikbaar in het hoofdprogramma omdat deze voor het eerst worden gebruikt in Proc1 en Proc2. Wrd is dan niet beschikbaar in Proc1 en Proc2 omdat die variabele voor het eerst wordt gebruikt in het hoofdprogramma.

Overdraagbaarheid

Het argument met array-naam wordt niet ondersteund in dBASE III PLUS.

Zie ook

CLEAR MEMORY, DECLARE, LOCAL, PARAMETERS, PRIVATE, RELEASE, RESTORE, SAVE, STATIC, STORE

PUTFILE( )

Stations- en bestandsfuncties

Toont een dialoogvenster waarin de gebruiker een nieuwe bestandsnaam kan maken. De gemaakte bestandsnaam wordt als resultaat gegeven. Als de gebruiker het dialoogvenster verlaat door Annuleren te kiezen of op *Esc* te drukken, wordt een lege tekenreeks als resultaat gegeven.

Syntaxis

PUTFILE([<*titel Tuitdr*>
[, <*bestandsnaam Tuitdr*>
[, <*extensie Tuitdr*>
[, <*bestandstype Luitdr*>
[, <*verander bestandstype Luitdr*>]]]]])

<*titel Tuitdr*>

Een titel die boven aan het dialoogvenster wordt weergegeven.

<*bestandsnaam Tuitdr*>

De standaardbestandsnaam die in het invoervak in het dialoogvenster verschijnt. Als <*bestandsnaam Tuitdr*> achterwege wordt gelaten, toont PUTFILE( ) een leeg invoervak.

<*extensie Tuitdr*>

Een standaardextensie voor de bestandsnaam die door PUTFILE( ) als resultaat wordt gegeven.

<bestandstype Luitdr>
Een logische waarde die bepaalt of in het dialoogvenster een lijst verschijnt van alle bestandstypen (.T.) of van tabellen in een database (.F.). De standaardinstelling is .T. Als u een waarde wilt opgeven voor *<bestandstype Luitdr>*, moet u ook een waarde of lege reeks ("") opgeven voor *<bestandsnaamfilter>*, *<titel Tuitdr>* en *<extensie Tuitdr>*.

<verander bestandstype Luitdr>
Een logische waarde die bepaalt of de gebruiker in het dialoogvenster kan schakelen tussen database-tabellen en andere bestandstypen (.T.) of niet tussen bestandstypen kan schakelen (.F.). De standaardinstelling is .F. Als u een waarde wilt opgeven voor *<verander bestandstype Luitdr>*, moet u ook een waarde of lege reeks ("") opgeven voor *<bestandsnaamfilter>*, *<titel Tuitdr>* en *<extensie Tuitdr>*, en u moet een waarde opgeven voor *<bestandstype Luitdr>*.

Beschrijving

Met PUTFILE( ) kunt u een nieuwe bestandsnaam genereren. Als de bestandsnaam eenmaal is gegenereerd, kunt u die gebruiken in andere commando's en functie-aanroepen. U kunt PUTFILE( ) bijvoorbeeld gebruiken om een naam voor een tabelbestand te genereren, en die naam vervolgens met RENAME aan een ander bestand doorgeven.

In het dialoogvenster dat met PUTFILE( ) wordt geopend, verschijnen standaard de bestandsnamen in de directory die het laatst in het dialoogvenster werd getoond. Elke keer dat u het dialoogvenster gebruikt om een andere directory te tonen, wordt die directory de standaarddirectory als het dialoogvenster een volgende keer wordt geopend.

Voorbeeld

In de volgende voorbeelden wordt PUTFILE( ) gebruikt:

```
B1=PUTFILE()  && Dialoogvenster openen
B2=PUTFILE("Kies een bestandsnaam voor het rapport")
* Dialoogvenster openen met titel
B3=PUTFILE("Geef naam op voor rapport","Rapport.txt")
* Gebruiker accepteert standaardnaam of geeft andere naam op
B4=PUTFILE("Geef tabelnaam op","Tijd",".dbf")
* Sandaardnaam is Tijd en standaardextensie is .dbf
? "B1",B1
? "B2",B2
? "B3",B3
? "B4",B4
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

FILE( ), GETFILE( ), RENAME

PV( )

Resulteert in een zwevende waarde die de huidige waarde van een investering voorstelt.

Syntaxis

PV(*<betaling Nuitdr>*, *<rente Nuitdr>*, *<termijnen Nuitdr>*)

<betaling Nuitdr>

Het bedrag voor de termijnbetaling. Denk er aan voor de betalingen dezelfde periode te gebruiken als voor het rentepercentage en de termijnen. De betaling kan negatief of positief zijn.

<rente Nuitdr>

De rente per periode als een positief decimaal getal. Denk er aan voor het rentepercentage dezelfde periode te gebruiken als voor de betalingen en de termijnen.

<termijnen Nuitdr>

Het aantal betalingen. Denk er aan voor de termijnen dezelfde periode te gebruiken als voor de betalingen en het rentepercentage.

Beschrijving

PV( ) is een financiële functie die de huidige waarde van een investering berekent. PV( ) geeft een zwevende waarde als resultaat die het bedrag voorstelt dat moet worden terugbetaald bij gelijke betalingen met een vaste rente gedurende een vaste looptijd. U kunt PV( ) bijvoorbeeld gebruiken als u wilt weten hoeveel u moet investeren om gedurende een vaste looptijd regelmatige betalingen te ontvangen.

Geef het rentepercentage op in de vorm van een decimaal getal. Als de rente op jaarbasis bijvoorbeeld 9,5% is, is *<rente Nuitdr>* gelijk aan ,095 (9,5/100) voor jaarbetalingen.

Denk er aan voor *<betaling Nuitdr>*, *<rente Nuitdr>* en *<termijnen Nuitdr>* dezelfde periode te gebruiken. Als de betalingen bijvoorbeeld maandelijks geschieden, moet u de rente per maand en het aantal maandtermijnen opgeven. Een rente op jaarbasis van 9,5% geeft u bijvoorbeeld op als ,095/12 (dat is 9,5/100 gedeeld door 12 maanden).

Het resultaat van PV( ) wordt als volgt berekend:

$$\text{huidige waarde} = \text{betaling} * \frac{(1 + \text{rente})^{\text{termijn}} - 1}{\text{rente} * (1 + \text{rente})^{\text{termijn}}}$$

waarbij rente = rentepercentage / 100

De huidige waarde van een investering met een rente van 9% die gedurende 5 jaar een maandelijkse betaling oplevert van F 350, berekent u als volgt:

```
? PV(350,.09/12,60)              && Geeft 16860.68 als resultaat
nTijd = (1 + .09/12)^60
? 350*(nTijd-1)/(.09/12*nTijd)   && Geeft 16860.68 als resultaat
```

Met andere woorden, u moet nu F 16.860,68 investeren op een rekening die 9% op jaarbasis oplevert om gedurende de volgende 5 jaar F 350 per maand te ontvangen.

Het aantal decimalen stelt u in met SET DECIMALS.

Voorbeeld

In het volgende voorbeeld wordt PV( ) gebruikt om de grootte te berekenen van een hypotheeklening die de gebruiker zich kan veroorloven op basis van de gewenste maandbetaling, de huidige rentestand en het aantal maandbetalingen (360 voor een hypotheek met een looptijd van 30 jaar):

```
LOCAL f
f=NEW PV()
f.OPEN()
CLASS PV OF FORM
  this.Width=50
  this.Height=15
  this.Text= "Wat kan ik mij veroorloven?"
  this.ColorNormal="BG+/BG"
  DEFINE TEXT Tekst1 OF THIS AT 3,5;
    PROPERTY Text "Gewenste maandbetaling?",;
    Width 26
  DEFINE ENTRYFIELD Bedrag OF THIS AT 3,38 ;
    PROPERTY Value 0, Picture "9999",;
    Width 5
  DEFINE TEXT Tekst2 OF THIS AT 5,5;
    PROPERTY Text "Huidige rentepercentage?",;
    Width 26
  DEFINE ENTRYFIELD Rente OF THIS AT 5,38 ;
    PROPERTY Value 0, Picture "99.99",;
    Width 5
  DEFINE TEXT Tekst3 OF THIS AT 7,5;
    PROPERTY Text "Aantal maandbetalingen?",;
    Width 30
  DEFINE ENTRYFIELD Betalingen OF THIS AT 7,38 ;
    PROPERTY Value 0, Picture "999",;
    Width 4
  DEFINE PUSHBUTTON Resultaat OF THIS AT 11,14;
    PROPERTY Text "Te lenen bedrag", Width 18,;
    OnClick {;MijnResultaat="U kunt lenen: F "+;
    LTRIM(STR(PV(Form.Bedrag.Value,;
    (Form.Rente.Value/100)/12,Form.Betalingen.Value),13,2));
    ; Form.PV.Text=MijnResultaat},;
    Height 2, ColorNormal "N/W"
  DEFINE TEXT PV OF THIS AT 9,8;
    PROPERTY Text "Hoeveel kan ik lenen? ", Width 30
ENDCLASS
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FV( ), PAYMENT( ), SET DECIMALS

QUIT

Programma's

Sluit alle geopende bestanden, wist alle geheugenvariabelen, sluit dBASE af en geeft de besturing terug aan het besturingssysteem.

Syntaxis

QUIT [WITH <*Nuitdr*>]

WITH <*Nuitdr*>

Resulteert in een code, <*Nuitdr*>, aan het besturingssysteem als u dBASE afsluit.

Beschrijving

Met QUIT kunt u het werken met dBASE beëindigen. Als u QUIT opneemt in een programmabestand, wordt de uitvoering van het programma afgebroken en wordt dBASE afgesloten. Als u de uitvoering van een programma wilt beëindigen zonder dBASE af te sluiten, moet u CANCEL of RETURN gebruiken.

Gebruik QUIT WITH <*Nuitdr*> om een code terug te geven aan Windows of aan een andere toepassing.

Voorbeeld

In het volgende voorbeeld wordt een formulier gemaakt met knoppen waarmee de gebruiker dBASE kan afsluiten met het commando QUIT. Als op de knop "Afsluiten" wordt geklikt, wordt een codeblok met het commando QUIT uitgevoerd:

```
DEFINE FORM AfslTest FROM 2,2 TO 15,40;
  PROPERTY Text "Test met Quit"
DEFINE PUSHBUTTON DK1 OF AfslTest AT 8,9;
  PROPERTY Text "dBASE afsluiten", Width 19,;
    OnClick {;QUIT}
OPEN FORM AfslTest
```

Overdraagbaarheid

De optie WITH <*Nuitdr*> wordt niet ondersteund in dBASE III PLUS.

Zie ook

CANCEL, CLEAR, CLOSE..., RELEASE, RETURN, RUN, RUN( )

RANDOM( ) Numerieke gegevens

Resulteert in een decimaal getal tussen 0 en 1.

Syntaxis

RANDOM([<*Nuitdr*>])

<Nuitdr>
Het numerieke of zwevende getal dat als startwaarde wordt gebruikt.

Beschrijving

Met RANDOM( ) kunt u een reeks willekeurige getallen genereren. Als u voor <*Nuitdr*> een positieve waarde opgeeft, gebruikt RANDOM( ) <*Nuitdr*> als startwaarde. Een startwaarde is een waarde die door de functie wordt gebruikt om een waarde als resultaat te geven. De startwaarde en het resultaat zijn dus niet gelijk, maar de startwaarde bepaalt wel het resultaat. Als u voor <*Nuitdr*> altijd hetzelfde positieve getal gebruikt, resulteert RANDOM( ) dus altijd in hetzelfde getal. De instructie RANDOM(199) geeft bijvoorbeeld altijd 0,11 als resultaat, RANDOM(399) geeft altijd 0,23 als resultaat en RANDOM(599) geeft altijd 0,35 als resultaat.

Als u <*Nuitdr*> niet opgeeft of 0 gebruikt, gebruikt RANDOM( ) het getal dat als laatste door RANDOM( ) als resultaat is gegeven als startwaarde. Als RANDOM( ) nog niet eerder is gebruikt, wordt 0 als startwaarde gebruikt.

Als voor <*Nuitdr*> een negatieve waarde opgeeft, krijgt u meer werkelijk willekeurige waarden als resultaat. Als u een negatieve <*Nuitdr*> opgeeft bij RANDOM( ), wordt een startwaarde gebruikt die is gebaseerd op het aantal seconden van de systeemklok. Vandaar dat RANDOM( ) met een negatieve waarde voor <*Nuitdr*> meer willekeurige waarden oplevert.

U kunt RANDOM( ) gebruiken om een reeks ogenschijnlijk willekeurige getallen te genereren en toch enige beheersing te houden over welke getallen dat zijn. Als u bijvoorbeeld RANDOM(5) gebruikt en vervolgens drie keer RANDOM( ) gebruikt zonder de optie <Nuitdr>, krijgt u altijd 0,03, 0,49, 0,56, 0,68 als resultaat. Bij de eerste toepassing van RANDOM( ) is 5 de startwaarde; de tweede keer is 0,03 de startwaarde omdat RANDOM( ) de laatste keer 0,03 als resultaat heeft gegeven.

Het aantal decimalen stelt u in met SET DECIMALS.

Voorbeeld

Het volgende programma genereert een lijst met 20 willekeurige vluchten uit de tabel Vluchten:

```
USE Vluchten
SET TALK OFF
CLEAR
Teller=1
DO WHILE Teller <=20
  Vlcht=INT(RANDOM()*RECCOUNT()+1)
```

```
    GOTO Vlcht
    ? Vluchtnr AT 10, Van, Naar
    Teller=Teller+1
  ENDDO
  X=INKEY(5)      && Vertraging van 5 seconden
  CLOSE ALL
```

In het volgende voorbeeld wordt RANDOM( ) gebruikt om een willekeurig getal te genereren dat wordt gebruikt als deel van een unieke tabelnaam:

```
CLEAR
TijdBest = "TMP" + LTRIM(STR(RANDOM()*10000,5,0))+".DBF"
USE Klanten
COPY STRUCTURE TO &TijdBest
USE &TijdBest IN 2
SELECT 2
? "De huidige tabel is: " + TijdBest
X=INKEY(5)      && Vertraging van 5 seconden
CLOSE ALL
ERASE &TijdBest
```

In het volgende voorbeeld wordt RANDOM( ) gebruikt om drie reeksen van zes willekeurige getallen te genereren in het bereik van 1 tot en met 41:

```
CLEAR
SET TALK OFF
Spel=1
Keuze=1
DO WHILE Spel<=3
  DO WHILE Keuze<=6
    lotto = RANDOM()*41
    ? "Keuze"+LTRIM(STR(Keuze))+"  " AT 10, ;
      LTRIM(STR(lotto,5,0))
    Keuze=Keuze+1
  ENDDO
  ?
  Keuze=1
  Spel=Spel+1
ENDDO
SET TALK ON
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. In dBASE IV heet deze functie RAND( ); RANDOM( ) wordt niet herkend.

In dBASE IV kunt u de standaardstartwaarde herstellen met RAND(100001). dBASE voor Windows ondersteunt deze mogelijkheid niet. Verder resulteert RAND(0) in dBASE IV altijd in dezelfde waarde. In dBASE voor Windows is RANDOM(0) gelijk aan RANDOM( ) en in dat geval wordt het resultaat van de laatste toepassing van RANDOM( ) als startwaarde gebruikt.

Zie ook

GENERATE, SET DECIMALS

RAT( ) Tekenreeksgegevens

Resulteert in een getal dat de beginpositie voorstelt van een tekenreeks binnen aan andere tekenreeks of een memoveld. RAT( ) zoekt achterwaarts van rechts naar links in de reeks of het memoveld. De positie die als resultaat wordt gegeven, wordt echter berekend vanaf het begin (de linkerkant) van de reeks.

Syntaxis

RAT(*<zoekreeks Tuitdr>*, *<doel Tuitdr>* | *<doelmemoveld>*
[, *<nde overeenkomst Nuitdr>*])

<zoekreeks Tuitdr>

De tekenreeks die moet worden gezocht in *<doel Tuitdr>* of *<doelmemoveld>*.

<doel Tuitdr> | *<doelmemoveld>*

De tekenreeks of het memoveld waarin naar *<zoekreeks Tuitdr>* moet worden gezocht, de *doelreeks*.

<nde overeenkomst Nuitdr>

Als *<zoekreeks Tuitdr>* meer dan eens voorkomt, kunt u hier opgeven van welke overeenkomst (vanaf de rechterkant) u de positie als resultaat wilt krijgen. Als u *<nde overeenkomst Nuitdr>* achterwege laat, wordt de eerste overeenkomst vanaf rechts gezocht.

Beschrijving

Met RAT( ) kunt u in een tekenreeks of memoveld zoeken naar de eerste of *<nde overeenkomst Nuitdr>* met *<zoekreeks Tuitdr>*. Er wordt gezocht van rechts naar links, van het eind tot het begin. Bij het zoeken wordt onderscheid gemaakt tussen hoofdletters en kleine letters. Gebruik UPPER( ) of LOWER( ) om dat onderscheid te annuleren.

Hoewel wordt gezocht van het eind tot het begin van de reeks of het memoveld, geeft RAT( ) de numerieke positie van *<zoekreeks Tuitdr>* als resultaat vanaf het *begin* van de reeks of het memoveld, zoals wordt gedemonstreerd in dit voorbeeld:

```
? RAT("abc","abcdefabc")  && geeft 7 als resultaat
**                  |
** Het streepje geeft aan waar 'abc' voor het
**   eerst in de tekenreeks wordt aangetroffen
** Dat teken staat op positie 7 vanaf het
**   begin van de tekenreeks
```

Als *<zoekreeks Tuitdr>* slechts een keer voorkomt in de doelreeks, geven RAT( ) en AT( ) beide dezelfde waarde als resultaat. De instructies RAT("abc","abcdef") en AT("abc","abcdef") geven bijvoorbeeld beide 1 als resultaat.

RAT( ) geeft onder de volgende omstandigheden 0 als resultaat:

- De zoekreeks wordt niet aangetroffen
- De zoekreeks is een lege tekenreeks

- De zoekreeks is langer dan de doelreeks
- De *nde overeenkomst* bestaat niet

Met AT( ) kunt u naar het begin van <*zoekreeks Tuitdr*> zoeken van *links naar rechts*, van het begin tot het eind. Met de subreeks-operator kunt u bepalen of een tekenreeks bestaat binnen een andere tekenreeks. Zie Hoofdstuk 1 voor meer informatie over operatoren.

Voorbeeld

In het volgende voorbeeld wordt RAT( ) gebruikt om de beginpositie te bepalen van een doorgegeven tekenreeks in een tweede tekenreeks:

```
? RAT("B","ABC")                  && Geeft 2 als resultaat
? RAT("ij","Bijblijven")          && Geeft 6 als resultaat
? RAT("ij","Bijblijven",2)        && Geeft 2 als resultaat
? RAT("Z","ABC")                  && Geeft 0 als resultaat
? RAT("a","ABC")                  && Geeft 0 als resultaat
? RAT("ABC","AB")                 && Geeft 0 als resultaat
? RAT("","ABC")                   && Geeft 0 als resultaat
? RAT("a","abc",2)                && Geeft 0 als resultaat
```

In het volgende voorbeeld wordt RAT( ) gebruikt bij het weergeven van de inhoud van een veld:

```
USE Afnemers
LIST FIELDS Bedrijf, Contact ;
   FOR RAT("COMPUTER",UPPER(Bedrijf)) > 0
CLOSE DATABASES
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. In dBASE IV wordt maximaal ongeveer 64K van een memoveld onderzocht.

Zie ook

AT( ), LOWER( ), STUFF( ), SUBSTR( ), UPPER( )

READ

Invoer/uitvoer

Activeert alle @...GET-velden en geheugenvariabelen in het resultatenpaneel van het commandovenster of het huidige dBASE IV-venster. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) om formulieren te maken en te activeren.

Zie Help voor meer informatie over de syntaxis van READ. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

READKEY( )

Toetsenbord- en muisacties

Resulteert in een geheel getal dat de toets of toetscombinatie voorstelt die is gebruikt om een commando voor het volledige scherm te beëindigen. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows kenmerken als OnClick om handelingen te starten op basis van de manier waarop de gebruiker een formulier heeft afgesloten.

Zie Help voor meer informatie over de syntaxis van READKEY( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het maken van formulieren.

READMODAL( )

Formulieren

Opent een formulier als een modaal venster en resulteert in de naam van het object dat de focus heeft als de gebruiker het formulier voorlegt.

Syntaxis

READMODAL(<*objectverwijzing*> [, <*Luitdr*>])

<objectverwijzing>
Verwijst naar het object dat in eerste instantie de focus heeft als het formulier wordt geopend.

<Luitdr>
Een logische uitdrukking die bepaalt of het formulier kan worden gesloten met *Esc*. De standaardinstelling is waar (.T.).

Beschrijving

Met READMODAL( ) kunt u een formulier openen als een modaal venster.

Een formulier dat u opent als een modaal venster, heeft de volgende eigenschappen:

- Zolang het formulier geopend is, kan de focus niet worden verplaatst naar andere formulieren.
- De uitvoering van de routine waarmee het formulier is geopend, wordt onderbroken tot het formulier is gesloten. Nadat het formulier is gesloten, wordt de besturing doorgegeven aan de commandoregel na de regel waarmee het formulier werd geopend.

In veel applicaties worden modale vensters gebruikt als dialoogvensters die de een handeling van de gebruiker vereisen voordat het dialoogvenster kan worden gesloten.

Standaard resulteert READMODAL( ) in de naam van het object dat de focus heeft als de gebruiker het formulier voorlegt. U kunt echter een eigen resultaatwaarde voor READMODAL( ) opgeven met het argument WITH bij CLOSE FORMS.

READMODAL( ) komt overeen met de methode ReadModal( ), die ook een formulier opent als een modaal venster.

Met de methode Open( ) of het commando OPEN FORM kunt u een formulier openen als een *niet-modaal* venster.

Voorbeeld

In het volgende voorbeeld wordt READMODAL( ) gebruikt om een eerder gedefinieerd formulier te tonen en activeren:

```
SET CUAENTER OFF
DEFINE FORM Hypotheek FROM 2,2 TO 15,60;
  PROPERTY Text "Hypotheken goed bekeken",MDI .F.,;
    OnClose {;SET CUAENTER ON}
  DEFINE TEXT Tekst1 OF Hypotheek AT 2,5;
    PROPERTY Text "Wat is de hoofdsom?",;
    Width 26, ColorNormal "RB/W"
  DEFINE ENTRYFIELD Hoofdsom OF Hypotheek AT 2,45;
    PROPERTY Picture "999999",;
    Width 7, Value 0
  DEFINE TEXT Tekst2 OF Hypotheek AT 4,5;
    PROPERTY Text "Wat is het rentepercentage?",;
    Width 30, ColorNormal "RB/W"
  DEFINE ENTRYFIELD Rente OF Hypotheek AT 4,45 ;
    PROPERTY Picture "99.99",;
    Width 5, Value 0
  DEFINE TEXT Tekst3 OF Hypotheek AT 6,5;
    PROPERTY Text "Wat is het aantal betalingen?", ;
    Width 39,ColorNormal "RB/W"
  DEFINE ENTRYFIELD Betalingen OF Hypotheek AT 6,45 ;
    PROPERTY Picture "999",;
    Width 4, Value 0
  DEFINE TEXT Tekst4 OF Hypotheek AT 8,8;
    PROPERTY Text "De termijnbetaling wordt:",;
    Width 30, ColorNormal "R+/W"
  DEFINE ENTRYFIELD Temijnbet OF Hypotheek AT 8,45;
    PROPERTY Width 10, Value 0, Picture "F 9999.99",;
    OnGotFocus {;this.Value=;
      PAYMENT(Form.Hoofdsom.Value,;
      (Form.Rente.Value/100)/12,Form.Betalingen.Value)}
  DEFINE PUSHBUTTON Sluiten OF Hypotheek AT 11,18;
    PROPERTY Width 14, Text "Sluiten", OnClick {;Form.Close()}
READMODAL(Hypotheek.Hoofdsom)
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ACTIVATE WINDOW, CLEAR WINDOWS, CLOSE..., DEACTIVATE WINDOW, DEFINE WINDOW, Open( ), OPEN FORM, ReadModal(), RELEASE WINDOWS

RECALL

Verwijdert de markeringen van records die eerder in de huidige tabel zijn gemarkeerd voor verwijdering. Dit commando heeft geen effect voor Paradox- of SQL-tabellen, omdat in die tabellen verwijderde records onmiddellijk uit het bestand worden gewist.

Syntaxis

RECALL
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]

<bereik>

Het aantal records waarvan u de markering wilt verwijderen. RECORD <*n*> geeft een record aan door middel van het recordnummer. NEXT <*n*> geeft *n* records aan, te beginnen bij het huidige record. ALL geeft alle records aan. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR *<voorwaarde1>*
WHILE *<voorwaarde2>*

Bepaalt welke records worden beïnvloed door RECALL. FOR beperkt RECALL tot records die voldoen aan <*voorwaarde1*>. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang <*voorwaarde2*> waar is.

Beschrijving

Met RECALL kunt u de verwijdering ongedaan maken van records die met DELETE zijn gemarkeerd voor verwijdering in de huidige tabel, maar nog niet fysiek zijn verwijderd (geschoond) met PACK. DELETE heeft niet tot gevolg dat de records feitelijk worden verwijderd, maar markeert deze alleen als verwijderd. Met RECALL kunt u dit proces herroepen en de records weer volledig in de tabel opnemen. RECALL zonder opties herroept alleen de verwijdering van het huidige record.

Records die uit het bestand zijn verwijderd met PACK of ZAP zijn permanent verwijderd en kunnen niet worden teruggehaald met RECALL.

Als SET DELETED is ingeschakeld (ON), hebben de opties <*bereik*> en FOR geen effect en is de optie WHILE alleen van invloed op het huidige record. RECALL zonder opties wordt niet beïnvloed door SET DELETED. In dat geval wordt altijd alleen het huidige record hersteld.

In de meeste gevallen heeft SET DELETED ON tot gevolg dat alle verwijderde records in de actieve tabel worden uitgefilterd. Het is echter mogelijk toegang te krijgen tot een voor verwijdering gemarkeerd record door middel van het recordnummer. Als het record met recordnummer 4 is verwijderd en SET DELETED is ingeschakeld (ON), plaatst GO 4 de recordaanwijzer bij het verwijderde record.

Voorbeeld

Zie DELETE voor een voorbeeld met RECALL.

Zie ook

DELETE, PACK, SET DELETED, ZAP

RECCOUNT( )

Velden en records

Resulteert in het aantal records in de huidige of een opgegeven tabel.

Syntaxis

RECCOUNT([<*alias*>])

<alias>

Een werkgebiednummer (van 1 tot 255) of -letter (van A tot J) of aliasnaam. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens.

Beschrijving

RECCOUNT( ) haalt een telling van het aantal records in een tabel op uit de tabel-header, waarin informatie is opgeslagen over de tabelstructuur. Als u daarentegen COUNT zonder argumenten gebruikt, wordt een telling van het aantal records als resultaat gegeven die wordt verkregen door de records in de tabel te tellen.

Als geen tabel actief is, geeft RECCOUNT( ) 0 als resultaat. RECCOUNT( ) omvat alle records, zelfs als SET DELETED is ingeschakeld (ON) of als SET FILTER actief is.

U kunt RECSIZE( ) gebruiken in combinatie met RECCOUNT( ) om de omvang van de tabel in bytes bij benadering te bepalen. Het commando DIR toont het aantal bytes dat door DOS is toegewezen aan een tabel. Dat aantal hoeft niet exact gelijk te zijn aan het werkelijke aantal bytes in het bestand.

Voorbeeld

In het volgende voorbeeld wordt RECCOUNT( ) gebruikt om het aantal records als resultaat te geven in elk van de opgegeven tabellen:

```
USE Bedrijf IN SELECT()
USE Contact IN SELECT()
? "De tabel Bedrijf bevat " + ;
  LTRIM(STR(RECCOUNT("Bedrijf"),4,0)) + " records."
? "De tabel Contact bevat " + ;
  LTRIM(STR(RECCOUNT("Contact"),4,0)) + " records."
CLOSE ALL
```

In het volgende voorbeeld wordt RECCOUNT( ) gebruikt om een array te initialiseren met de afmetingen van de tabel Klanten. Vervolgens wordt de inhoud van de tabel naar de array gekopieerd:

```
SET TALK OFF
USE Klanten
DECLARE Klnt2[RECCOUNT(),FLDCOUNT()]
COPY TO ARRAY Klnt2
Aant = 1
DO WHILE Aant <= RECCOUNT()
  ? Klnt2[Aant,2]       && Bedrijfsnaam tonen
  Aant=Aant+1
ENDDO
CLOSE ALL
CLEAR ALL
SET TALK ON
```

Zie ook

DIR, DBF( ), DISKSPACE( ), DISPLAY STRUCTURE, RECNO( ), RECSIZE( )

RECNO( )

Velden en records

Resulteert in het huidige recordnummer in de huidige of een opgegeven tabel. Voor Paradox- en SQL-tabellen resulteert deze functie in een bladwijzer voor de huidige positie in een tabel.

Syntaxis

RECNO([*<alias>*])

<alias>

Een werkgebiednummer (van 1 tot 255) of -letter (van A tot J) of aliasnaam. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens.

Beschrijving

RECNO( ) resulteert in het nummer van het huidige record van de tabel in het huidige of een opgegeven werkgebied. RECNO( ) houdt rekening met alle records, zelfs als SET DELETED is ingeschakeld (ON) of SET FILTER actief is. Als geen tabel is geopend in het opgegeven werkgebied, resulteert RECNO( ) in 0.

Als de recordaanwijzer voorbij het laatste record (EOF) in de tabel is geplaatst, resulteert RECNO( ) in een waarde die 1 hoger is dan het totaal aantal records in de tabel. Als de recordaanwijzer voor het eerste record (BOF) in de tabel staat, geeft RECNO( ) 1 als resultaat. RECNO( ) geeft ook 1 als resultaat als de tabel leeg is (geen records bevat).

Voorbeeld

In het volgende voorbeeld wordt RECNO( ) gebruikt om het recordnummer te tonen in een SCAN-lus en nadat de recordaanwijzer van het eind van het bestand is verplaatst naar het begin van het bestand:

```
SET TALK OFF
CLEAR
USE Bedrijf IN SELECT() EXCLUSIVE
? "Recordnummer" AT 4, "Bedrijf" AT 18
?
SCAN
   ? LTRIM(STR(RECNO())) AT 4, Bedrijf AT 18
ENDSCAN
IF EOF()
? "Het eind van het bestand is bereikt ;
bij recordnummer " +  LTRIM(STR(RECNO(),3,0))
ENDIF
GO TOP
SKIP -1
IF BOF()
? "Het begin van het bestand is bereikt ;
bij recordnummer " +  LTRIM(STR(RECNO(),3,0))
ENDIF
CLOSE ALL
SET TALK ON
```

Zie ook

BOF( ), EOF( ), RECCOUNT( )

RECSIZE( )

Velden en records

Resulteert in het aantal bytes in een record in de huidige of opgegeven tabel.

Syntaxis

RECSIZE([<*alias*>])

<alias>

Een werkgebiednummer (van 1 tot 255) of -letter (van A tot J) of aliasnaam. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens.

Beschrijving

RECSIZE( ) resulteert in het aantal bytes in een record van een tabel in het huidige of een opgegeven werkgebied. Als in het opgegeven werkgebied geen tabel is geopend, resulteert RECSIZE( ) in 0.

LIST STRUCTURE en DISPLAY STRUCTURE geven ook informatie over de omvang van de records in een tabel.

Voorbeeld

In het volgende voorbeeld wordt RECSIZE( ) gebruikt om de recordlengte in bytes oor elke opgegeven tabel als resultaat te geven:

```
USE Bedrijf IN SELECT()
USE Contact IN SELECT()
? "De omvang van de records in tabel Bedrijf is " + ;
   LTRIM(STR(RECSIZE("Bedrijf"),4,0)) + " bytes"
? "De omvang van de records in tabel Contact is " + ;
   LTRIM(STR(RECSIZE("Contact"),4,0)) + " bytes"
CLOSE ALL
```

Zie ook

DBF( ), DIR, DISPLAY STRUCTURE, LIST STRUCTURE, RECCOUNT( ), RECNO( )

REDEFINE

Objecten

Wijzigt de definitie van een object in het geheugen.

Syntaxis

REDEFINE *<klassenaam> <objectnaam Tuitdr>*
[OF *<container-object>*]
[FROM *<rij, kolom>* TO *<rij, kolom>* | <AT *<rij, kolom>*]
[PROPERTY *<standaardkenmerkenlijst>*]
[CUSTOM *<lijst eigen kenmerken>*]
[WITH *<parameterlijst>*]

<klassenaam>

De klasse van het object dat u wijzigt. Zie DEFINE voor een lijst van standaard objectklassen:

<objectnaam Tuitdr>

Het identificatiesymbool van het object dat u wijzigt.

OF *<container-object>*

Geeft het object aan dat het object bevat dat u definieert. (Voor de meeste UI-objecten is het container-object een formulier.)

FROM *<rij>, <kolom>* TO *<rij>, <kolom>* | AT *<rij>, <kolom>*

Geeft de plaats en grootte op van het object bij aanvang aan binnen het hoofdformulier of -venster. FROM en TO bepalen respectievelijk de coördinaten linksboven en rechtstonder van het object. AT bepaalt de positie van de linkerbovenhoek.

PROPERTY *<standaardkenmerkenlijst>*

Geeft waarden aan die u toekent aan de standaardkenmerken van het object.

CUSTOM *<lijst eigen kenmerken>*

Geeft nieuwe kenmerken aan die u voor het object maakt en kent daaraan waarden toe. Zie Hoofdstuk 10 in *Programmeren* voor meer informatie over eigen kenmerken.

WITH *<parameterlijst>*
Geeft de parameters aan die u aan het object doorgeeft. Definieer deze parameters met de clausule PARAMETERS van het commando CLASS...ENDCLASS.

Beschrijving

Met REDEFINE kunt u een bestaand object wijzigen.

De eigenschappen van een object bestuurt u met *kenmerken*. Dit zijn geheugenvariabelen die deel uitmaken van het object. Een formulier heeft bijvoorbeeld een hoogte en een breedte die worden voorgesteld door de formulierkenmerken Height en Width. Met het commando REDEFINE kunt u nieuwe waarden toekennen aan deze en andere kenmerken nadat u een object hebt gemaakt met het commando DEFINE of de operator NEW.

Elk REDEFINE-commando beschikt over dezelfde opties als het overeenkomstige DEFINE-commando, en met beide commando's kunt u dezelfde kenmerken instellen.

U kunt ook kenmerken wijzigen met de punt-operator. Zie Hoofdstuk 10 in *Programmeren* voor meer informatie over de punt-operator en de bijbehorende syntaxis.

Voorbeeld

In het volgende voorbeeld worden vier formulieren gemaakt op het scherm. Het formulier Vierde bevat een knop waarmee de gebruiker de kenmerken kan wijzigen van de formulieren Eerste, Tweede en Derde om de formulieren een verticale oriëntatie te geven:

```
SET PROCEDURE TO PROGRAM(1) ADDITIVE
PUBLIC EERSTE, TWEEDE, DERDE, VIERDE
DEFINE FORM Eerste FROM 0,0 TO 10,29
DEFINE FORM Tweede FROM 0,31 TO 10,58
DEFINE FORM Derde FROM 13,0 TO 28,58
DEFINE FORM Vierde FROM 0,60 TO 28,74
  DEFINE PUSHBUTTON KenmWijz OF Vierde AT 20,1;
    PROPERTY;
      TEXT "Wijzigen", Width 15,;
      OnClick Wijzigen
  DEFINE PUSHBUTTON Sluiten OF Vierde AT 23,1;
    PROPERTY;
      TEXT "Sluiten", Width 15,;
      OnClick Einde
OPEN FORM Eerste, Tweede, Derde, Vierde

PROCEDURE Einde
CLOSE FORMS Eerste, Tweede, Derde, Vierde
RETURN

PROCEDURE Wijzigen
CLOSE FORMS Eerste, Tweede, Derde
REDEFINE FORM Eerste FROM 0,0 TO 28,17
REDEFINE FORM Tweede FROM 0,19 TO 28,37
REDEFINE FORM Derde FROM 0,39 TO 28,58
```

```
OPEN FORM Eerste, Tweede, Derde, Vierde
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS en dBASE IV.

Zie ook

CLASS...ENDCLASS, DEFINE

REFRESH

Tabellen

Werkt de gegevensbuffers voor het huidige of een opgegeven werkgebied bij zodat deze in overeenstemming zijn met de laatste wijzigingen in de gegevens.

Syntaxis

REFRESH [<*alias*>]

<alias>

Een werkgebiednummer (van 1 tot 255) of -letter (van A tot J) of aliasnaam. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens.

Beschrijving

Gebruik REFRESH om de gegevensbuffers van het opgegeven werkgebied bij te werken zodat in de weergegeven gegevens de wijzigingen worden overgenomen die zijn aangebracht door andere gebruikers in het netwerk. Dit commando wordt hoofdzakelijk gebruikt voor gegevens in tabellen die zijn opgeslagen op een database-server. U kunt het commando echter ook gebruiken om de gegevensbuffers voor geopende Paradox- en dBASE-tabellen bij te werken.

Voorbeeld

In het volgende voorbeeld worden de tabellen Klanten.DBF en Orders.DBF geopend. Omdat beide tabellen niet exclusief worden geopend, kan een andere gebruiker een record in Bedrijf of Orders wijzigen nadat de lokale gebruiker het record heeft gelezen vanaf de server. Met REFRESH worden de records opnieuw gelezen zodat de meest actuele gegevens worden weergegeven:

```
CLOSE ALL
SET EXCLUSIVE ON
USE S:\Dbasewin\Voorbd\Orders
INDEX ON Klantnr TAG Klantnr
* index samenstellen
CLOSE ALL
SET EXCLUSIVE OFF
USE S:\Dbasewin\Voorbd\Klanten
SELECT 2
```

```
USE S:\Dbasewin\Voorbd\Orders
SET RELATION TO Klantnr INTO Klanten
* Iedereen heeft toegang tot Klanten
SCAN
  REFRESH("Klanten")
  REFRESH("Orders")
  EDIT
ENDSCAN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

SET REFRESH

REINDEX

Tabelindeling

Werkt alle geopende indexbestanden in het geselecteerde werkgebied bij op basis van de wijzigingen in de huidige tabel.

Syntaxis

REINDEX

Beschrijving

Met REINDEX kunt u alle in de huidige tabel aangebrachte wijzigingen verwerken in indexbestanden die niet actief waren toen de tabel werd bijgewerkt, maar nu wel zijn geopend. Alle .NDX- en .MDX-bestanden die zijn geopend met USE of SET INDEX worden bijgewerkt.

Als een indexbestand is gemaakt met de optie UNIQUE van het commando INDEX, of als SET UNIQUE was ingeschakeld (ON) toen het indexbestand werd gemaakt, wordt de index opnieuw samengesteld als een unieke index. Tevens geldt dat als een aflopende index is gemaakt met de optie DESCENDING van het commando INDEX, het commando REINDEX de index opnieuw samenstelt in aflopende volgorde.

Voorbeeld

In het volgende voorbeeld wordt REINDEX gebruikt om een geopend indexbestand opnieuw te indexeren nadat een nieuw record is toegevoegd terwijl het indexbestand geopend, maar niet actief was:

```
USE Bedrijf EXCLUSIVE
INDEX ON VerkTotNu TAG Vtn OF Bedrijf1
* Bedrijf1.mdx met label Vtn maken
USE Bedrijf EXCLUSIVE
* Bedrijf1.mdx wordt niet bijgewerkt
APPEND BLANK
```

```
REPLACE Bedrijf WITH "Niet in index"
* Bedrijf1.mdx is nu niet actueel
USE Bedrijf INDEX Bedrijf1 EXCLUSIVE
SET ORDER TO TAG Vtn OF Bedrijf1
*
BROWSE TITLE "Nieuw record "+ltrim(str(reccount()))+" niet in index"
* Vtn is nu de index, maar is verouderd
* Het laatste record wordt niet gevonden
REINDEX
* Vtn is nu bijgewerkt
BROWSE TITLE "Laatste record nu ook in index"
```

Zie ook

INDEX, SET INDEX, SET ORDER, SET UNIQUE, USE

RELATION( ) Tabelindeling

Resulteert in de sleuteluitdrukking die voor het huidige of opgegeven werkgebied is gedefinieerd met het commando SET RELATION .

Syntaxis

RELATION(<*Nuitdr*> [,<*alias*>])

<Nuitdr>

Het nummer van de relatie die u als resultaat wilt geven.

<alias>

Een werkgebiednummer (van 1 tot 255) of -letter (van A tot J) of aliasnaam. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens.

Beschrijving

RELATION( ) resulteert in de uitdrukking die is gebruikt om met het commando SET RELATION tabellen te koppelen in het huidige of opgegeven werkgebied. Als de tabel aan meerdere andere tabellen is gekoppeld, kunt u het nummer opgeven van de relatie die u als resultaat wilt geven. RELATION( ) resulteert in een lege tekenreeks ("") als op de positie <*Nuitdr*> geen relatie is ingesteld.

Gebruik RELATION( ) om alle met SET RELATION ingestelde sleuteluitdrukkingen op te slaan zodat u deze op een later tijdstip opnieuw kunt gebruiken. Met de functie TARGET( ) kunt u de doeltabel opslaan. Dat is de tabel waarnaar u met SET een relatie hebt ingesteld.

Voorbeeld

In het volgende voorbeeld wordt RELATION( ) gebruikt om de sleuteluitdrukking te bepalen die is gebruikt om een relatie tot stand te brengen tussen twee subtabellen, Orders en Regels, en de hoofdtabel Klanten:

```
CLOSE ALL
CLEAR
SELECT 1
USE Klanten EXCLUSIVE
INDEX ON Klantnr TAG Klantnr
SELECT 2
USE Orders EXCLUSIVE
SELECT 3
USE Regels EXCLUSIVE
INDEX ON Ordernr TAG Ordernr
SELECT 2
SET RELATION TO Ordernr INTO Regels
SET RELATION TO Klantnr INTO Klanten ADDITIVE
IF LEN(RELATION(1)) > 0
   @ 9,0 SAY "Orders.dbf is gekoppeld aan " + ;
      TARGET(1)+ " met sleuteluitdrukking: "+RELATION(1)
   IF LEN(RELATION(2)) > 0
      @ 11,0 SAY "Orders.dbf is ook gekoppeld aan " + ;
      TARGET(2)+ " met sleuteluitdrukking: "+RELATION(2)
   ENDIF
ENDIF
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ALIAS( ), CREATE QUERY, CREATE VIEW, CREATE VIEW...FROM ENVIRONMENT, SELECT( ), SET RELATION, SET VIEW, SET( ), TARGET( )

RELEASE — Geheugenvariabelen

Verwijdert opgegeven geheugenvariabelen en array's, zodat geheugenruimte vrijkomt voor andere variabelen.

Syntaxis

RELEASE <*variabelenlijst*> |
   ALL
      [LIKE <*variabelefilter1*>]
      [EXCEPT <*variabelefilter2*>]

<variabelenlijst>
De geheugenvariabelen die u uit het geheugen wilt verwijderen, gescheiden door komma's.

ALL
Verwijdert alle geheugenvariabelen uit het geheugen.

LIKE *<variabelefilter1>*
Verwijdert alle geheugenvariabelen uit het geheugen waarvan de namen overeenkomen met het geheugenvariabelenschema dat u opgeeft voor *<variabelefilter1>*. Stel *<variabelefilter1>* samen met tekens die voorkomen in de variabelenamen en de jokers * en ?.

EXCEPT *<variabelefilter2>*
Verwijdert alle geheugenvariabelen uit het geheugen behalve die waarvan de namen overeenkomen met het geheugenvariabelenschema dat u opgeeft voor *<variabelefilter2>*. Stel *<variabelefilter2>* samen met tekens die voorkomen in de variabelenamen en de jokers * en ?. U kunt LIKE en EXCEPT opnemen in dezelfde instructie, bijvoorbeeld RELEASE ALL LIKE gvar* EXCEPT gvarnum*.

Beschrijving
Met RELEASE kunt u geheugenvariabelen wissen, zodat ruimte vrijkomt voor nieuwe variabelen. Als u grote groepen variabelen wilt verwijderen, kunt u de optie ALL [LIKE *<variabelefilter1>*] [EXCEPT *<variabelefilter2>*] gebruiken.

Als u in een subroutine (een programma, procedure of door de gebruiker gedefinieerde functie) RELEASE ALL [LIKE *<variabelefilter1>*] [EXCEPT *<variabelefilter2>*] gebruikt worden alleen de lokale, in die subroutine gedefinieerde, geheugenvariabelen gewist. Geheugenvariabelen die zijn gedefinieerd in routines op een hoger niveau, worden niet gewist.

Als de besturing vanuit een subroutine wordt teruggegeven naar de aanroepende routine, worden alle in die subroutine gedefinieerde variabelen uit het geheugen verwijderd, behalve variabelen die zijn gedefinieerd met PUBLIC of STATIC. Het is dus niet nodig de lokale variabelen in een subroutine nadrukkelijk te wissen met RELEASE.

Voorbeeld
In het volgende voorbeeld wordt een reeks geheugenvariabelen geïnitialiseerd met namen die bestaan uit een aaneenschakeling van een tekenreeks en een recordnummer (op basis van de natuurlijke volgorde). Na deze bewerkingen zijn de veldwaarden van Klantnr en Bedrijf opgeslagen in de variabelen Knx en Bedrx. Met DISPLAY MEMORY wordt de aanwezigheid van deze geheugenvariabelen bevestigd. Vervolgens worden deze geheugenvariabelen met RELEASE ALL LIKE uit het geheugen verwijderd, hetgeen weer wordt bevestigd met twee instructies met DISPLAY MEMORY:

```
SET TALK OFF
SET SAFETY OFF
USE Klanten EXCLUSIVE
```

```
INDEX ON Klantnr TAG Klantnr
DO WHILE .NOT. EOF()
  VAR1="Klntnr"+LTRIM(STR(RECNO()))
  VAR2="Naam"+LTRIM(STR(RECNO()))
  STORE Klantnr TO &VAR1
  STORE Naam TO &VAR2
  SKIP
ENDDO
CLEAR
DISPLAY MEMORY          && Geheugenvars Knx en Bedrx
CLEAR
RELEASE ALL LIKE BEDR*
DISPLAY MEMORY          && Alleen Knx nog over
CLEAR
RELEASE ALL LIKE KN*
DISPLAY MEMORY          && Allemaal gewist
CLEAR
SET TALK ON
SET SAFETY ON
```

Overdraagbaarheid

In dBASE IV en dBASE III PLUS wordt het gebruik van LIKE en EXCEPT in dezelfde instructie niet ondersteund.

Zie ook

CLEAR, LOCAL, PRIVATE, PUBLIC, QUIT, RESTORE, RETURN, SAVE, STATIC

RELEASE AUTOMEM

Velden en records

Verwijdert alle automatische geheugenvariabelen uit het geheugen.

Syntaxis

RELEASE AUTOMEM

Beschrijving

Als u met STORE AUTOMEM, CLEAR AUTOMEM of USE...AUTOMEM een verzameling automatische geheugenvariabelen maakt voor een tabel, wordt voor elk veld één variabele geïnitialiseerd. Elke variabele krijgt dezelfde naam en hetzelfde gegevenstype als het bijbehorende veld. Als u de automatische geheugenvariabelen niet langer nodig hebt, kunt u RELEASE AUTOMEM gebruiken om deze uit het geheugen te verwijderen, zodat ruimte beschikbaar komt voor andere variabelen. RELEASE AUTOMEM geeft alle geheugenvariabelen vrij die dezelfde naam hebben als velden in de huidige tabel, zelfs variabelen die niet zijn gemaakt met een AUTOMEM-commando.

Als u een tabel sluit of naar een ander werkgebied gaat, heeft dat niet automatisch tot gevolg dat de automatische geheugenvariabelen voor die tabel worden gewist. In dBASE voor Windows wordt een variabele die niet dezelfde naam heeft als een veld in de huidige tabel, niet herkend als een automatische geheugenvariabele, zelfs niet als die is gemaakt als een automatische geheugenvariabele. Als u een tabel sluit of naar een ander werkgebied gaat, blijven de bijbehorende automatische geheugenvariabelen dan ook in het geheugen staan. Nadat u een tabel hebt gesloten of naar een ander werkgebied bent gegaan, kunt u deze variabelen niet uit het geheugen verwijderen met RELEASE AUTOMEM. In dat geval moet u dat doen met RELEASE of CLEAR ALL.

Voorbeeld

In het volgende voorbeeld wordt CLEAR AUTOMEM gebruikt om de gebruiker in staat te stellen automatische geheugenvariabelen voor een nieuw record te wijzigen. Het nieuwe record wordt echter alleen toegevoegd als de gebruiker bevestigt dat de gegevens juist zijn. RELEASE AUTOMEM wordt gebruikt in combinatie met ON ESCAPE en READKEY( ) om alle automatische geheugenvariabelen uit het geheugen te verwijderen als de gebruiker stopt met het invoeren van gegevens door op *Esc* te drukken:

```
SET TALK OFF
CLEAR
USE Afnemers
ON ESCAPE RELEASE AUTOMEM
MeerGegsInv()
RETURN

FUNCTION MeerGegsInv
@0,0 to 8, 70
@10,20 to 12,52
CLEAR AUTOMEM
DO WHILE .T.
  lBevestiging = .F.
  @11,22 CLEAR TO 11,50
  @1,1 SAY 'Afnemernr' GET m->AFNEMERNR
  @1,24 SAY 'Bedrijf' GET m->BEDRIJF
  @3,1 SAY 'Contactpersoon' GET m->CONTACT
  @4,1 SAY 'Adres' GET m->ADRES
  @6,1 SAY 'Plaats' GET m->PLAATS
  @6,23 SAY 'Regio' GET m->AREACODE
  @6,40 SAY 'Postcode' GET m->ZIPCODE
  READ
    IF READKEY() = 12
      RELEASE AUTOMEM
      EXIT
    ELSE
      @11,22 SAY "Gegevens juist? (J/N)";
      GET lBevestiging PICTURE 'J'
      READ
        IF lBevestiging
          APPEND AUTOMEM
          CLEAR AUTOMEM
        ENDIF
    ENDIF
```

```
ENDDO
RETURN .T.
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CLEAR AUTOMEM, RELEASE, STORE AUTOMEM, USE

RELEASE DLL

Programmeren in Windows

Maakt DLL-bestanden inactief.

Syntaxis

RELEASE DLL *<DLL-bestandsnamenlijst>*

Beschrijving

Gebruik RELEASE DLL als u fouten opspoort in een DLL-bestand of een dBASE-applicatie. Als u bijvoorbeeld een routine in een DLL-bestand wijzigt, moet u het bestand inactief maken en vervolgens weer actief maken.

Een DLL-bestand is een gecompileerde bibliotheek met externe routines die in een andere taal zoals C of Pascal zijn geschreven. Een DLL-bestand kan elke willekeurige extensie hebben, maar meestal wordt de extensie .DLL gebruikt. U activeert een DLL-bestand met het commando LOAD DLL.

Voorbeeld

In het volgende voorbeeld wordt de reeks commando's gebruikt waarmee RELEASE DLL kan worden toegepast als hulpmiddel voor het opsporen van problemen:

```
LOAD DLL mijnDLL.DLL
* ... test werking van DLL
RELEASE DLL mijnDLL.DLL
* ... wijzig .DLL of C-programma
LOAD DLL mijnDLL.DLL
* ... test opnieuw
RELEASE DLL mijnDLL.DLL
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

EXTERN, LOAD DLL

RELEASE MENUS

dBASE IV-menu's

Verwijdert inactieve menu's uit het geheugen. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows RELEASE OBJECT om een object van een formulier te verwijderen.

Zie Help voor meer informatie over de syntaxis van RELEASE MENUS. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

RELEASE OBJECT

Objecten

Verwijdert objectdefinities uit het geheugen.

Syntaxis

RELEASE OBJECT *<bevattende-objectverwijzing>.<objectverwijzing>*

<bevattende-objectverwijzing>.<objectverwijzing>

<bevattende-objectverwijzing> is een objectverwijzingsvariabele die verwijst naar het object (meestal een formulier) dat het object bevat. *<objectverwijzing>* is een objectverwijzingsvariabele die naar het object zelf verwijst.

Beschrijving

Met RELEASE OBJECT kunt u een object uit het geheugen verwijderen als u het niet langer nodig hebt en op die manier geheugen besparen.

Als een formulier uit het geheugen wordt verwijderd, worden tevens alle objecten op dat formulier verwijderd. Op dezelfde manier worden ook alle menu's die deel uitmaken van een menu verwijderd als u dat menu uit het geheugen verwijdert, .

RELEASE OBJECT komt overeen met de methode Release( ).

Voorbeeld

In het volgende voorbeeld worden op het formulier Kruis_aan drie aankruisvakjes gedefinieerd en wordt RELEASE OBJECT gebruikt om het aankruisvakje "Keuze 3" van het formulier te verwijderen als Keuze 1 wordt geselecteerd:

```
PUBLIC Kruis_aan
SET PROCEDURE TO PROGRAM(1) ADDITIVE
DEFINE FORM Kruis_aan FROM 0,0 TO 10,24
DEFINE CHECKBOX Av1 OF Kruis_aan AT 2,4;
  PROPERTY Text "Keuze 1",;
  OnChange Weghalen, Value .F.
DEFINE CHECKBOX Av2 OF Kruis_aan AT 4,4;
  PROPERTY Text "Keuze 2", Value .F.
DEFINE CHECKBOX Av3 OF Kruis_aan AT 6,4;
  PROPERTY Text "Keuze 3", Value .F.
```

```
OPEN FORM Kruis_aan

PROCEDURE Weghalen
IF TYPE("Kruis_aan.Av3")<>"U"
  RELEASE OBJECT Kruis_aan.Av3
ENDIF
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CLEAR ALL, CLOSE..., Release( )

RELEASE POPUPS — dBASE IV-menu's

Verwijdert dBASE IV-popup-menu's van het scherm en uit het geheugen. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows RELEASE OBJECT om een object van een formulier te verwijderen.

Zie Help voor meer informatie over de syntaxis van RELEASE POPUPS. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

RELEASE SCREENS — Invoer/uitvoer

Verwijdert alle of opgegeven variabelen uit het geheugen die zijn gemaakt met SAVE SCREEN en maakt de buffer van het commandovenster leeg. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV.

Zie Help voor meer informatie over de syntaxis van CLEAR SCREENS.

RELEASE WINDOWS — dBASE IV Windows

Verwijdert opgegeven definities van dBASE IV-vensters uit het geheugen. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows CLOSE FORMS of RELEASE OBJECT om een formulier te sluiten of vrij te geven.

Zie Help voor meer informatie over de syntaxis van RELEASE WINDOWS. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

RENAME

Geeft een bestand op schijf een andere naam.

Syntaxis

RENAME <*bestandsnaam1*> | ? | <*bestandsnaamfilter1*>
TO <*bestandsnaam2*> | ? | <*bestandsnaamfilter2*>

<*bestandsnaam1*> | ? | <*bestandsnaamfilter1*>

Geeft het oorspronkelijke bestand aan (het bronbestand). RENAME ? en RENAME <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een bestand kunt kiezen waaraan u een andere naam wilt geven. Als u een bronbestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bronbestand zonder extensie opgeeft, wordt geen extensie gebruikt.

TO <*bestandsnaam2*> | ? | <*bestandsnaamfilter2*>

Geeft de nieuwe naam voor het bronbestand aan (het doelbestand). De opties ? en <*bestandsnaamfilter*> openen een dialoogvenster waarin u de nieuwe naam en een andere directory kunt opgeven voor het doelbestand.

Beschrijving

Met RENAME kunt u de naam van een bestand wijzigen op het niveau van het besturingssysteem.

Als het bronbestand een extensie heeft, geeft u deze op in de instructie. Als het bronbestand niet in de huidige directory of in het met SET PATH ingestelde pad staat, geeft u een pad op.

RENAME verschilt van de DOS-versie van dit commando omdat in dBASE geen jokers zijn toegestaan. Als u met behulp van jokers meerdere bestanden van een andere naam wilt voorzien, moet u !, RUN of DOS gebruiken om de DOS-opdracht RENAME uit te voeren.

Als SET SAFETY is ingeschakeld (ON) en er bestaat een bestand met dezelfde naam als het doelbestand, verschijnt een dialoogvenster waarin wordt gevraagd of de gebruiker het bestaande bestand wilt overschrijven. Als SET SAFETY is uitgeschakeld (OFF) en er bestaat een bestand met dezelfde naam als het doelbestand, verschijnt een foutmelding en wordt het bestaande bestand niet overschreven.

Als u voor het doelbestand een nieuwe directory opgeeft, wordt het bronbestand naar die directory gekopieerd. Als u een ander station opgeeft voor het doelbestand, verschijnt een foutmelding en wordt het bronbestand niet gekopieerd of van een andere naam voorzien.

Als u met RENAME een .DBF-bestand een andere naam geeft, worden de bijbehorende .DBT- en .MDX-bestanden niet automatisch ook voorzien van een andere naam. Als u bijvoorbeeld een andere naam geeft aan een tabelbestand dat memovelden bevat en u

vergeet het bijbehorende .DBT-bestand dezelfde naam te geven, verschijnt een foutmelding als u probeert het bestand te gebruiken. In dergelijke gevallen kunt u beter een nieuwe kopie van het bestand maken met COPY TO.

Voorbeeld

In de volgende voorbeelden wordt RENAME gebruikt:

```
CLOSE DATABASES
RENAME Tijd.dbf TO SlaOp.dbf
* Tijd.dbf mag niet open zijn
RENAME Tijd.dbt TO SlaOp.dbt
RENAME ?
* Dialoogvenster verschijnt
RENAME *.qbe
* Dialoogvenster met alleen query-bestanden verschijnt
```

Zie ook

!, COPY FILE, DOS, RUN, SET DEFAULT, SET DIRECTORY, SET PATH, SET SAFETY

RENAME TABLE

Tabellen

Wijzigt de naam van de opgegeven tabel.

Syntaxis

RENAME TABLE *<oude-tabelnaam>* | ? | *<bestandsnaamfilter1>*
TO *<nieuwe-tabelnaam>* | ? | *<bestandsnaamfilter2>*
[[TYPE] PARADOX | DBASE]

<oude-tabelnaam> | ? | *<bestandsnaamfilter1>*

De tabel die u van een andere naam wilt voorzien. RENAME TABLE ? en RENAME TABLE *<bestandsnaamfilter>* tonen een dialoogvenster waarin u een tabelbestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand gezocht in de huidige directory en vervolgens in het pad dat is ingesteld met SET PATH. Als u een bestand opgeeft zonder een extensie of een type op te geven, wordt het bestandstype gebruikt dat is ingesteld met het commando SET DBTYPE.

U kunt ook een tabel in een database (gedefinieerd met het IDAPI-configuratieprogramma) een andere naam geven door voor de naam van het bestand de naam van de database op te geven (tussen dubbelepunten) in de notatie *:databasenaam:tabelnaam*. Als de database nog niet open is, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

<nieuwe-tabelnaam> | ? | *<bestandsnaamfilter2>*

De nieuwe naam voor de tabel. Als u een tabel in een database een andere naam geeft, moet u als doel voor de nieuwe tabel dezelfde database opgeven. Ook moet de nieuwe tabelnaam hetzelfde type hebben als de oorspronkelijke tabel. De opties ? en

<bestandsnaamfilter> tonen een dialoogvenster, waarin u de naam en directory voor de doeltabel kunt opgeven.

[TYPE] PARADOX | DBASE

Geeft het type aan van de tabel die u een andere naam wilt geven. Dat kan een Paradox- of een dBASE-tabel zijn.

Beschrijving

Met het commando RENAME TABLE kunt u de naam van een tabel wijzigen zonder terug te gaan naar het besturingssysteem. Het is niet mogelijk een geopende tabel een andere naam te geven, en de nieuwe naam mag niet al aanwezig zijn in dezelfde directory of database.

Als u een andere naam geeft aan een tabel met bijbehorende bestanden (zoals memo- en indexbestanden), wordt aan die bestanden automatisch ook de nieuwe naam toegewezen.

Voorbeeld

In het volgende voorbeeld wordt RENAME TABLE gebruikt om de naam van een gesloten tabel te wijzigen:

```
dbf_best = "VLUCHTEN.DBF"
IF SELECT() > 0
   area = 1
   verder = .T.
   DO WHILE area < SELECT() .AND. verder
      SELECT (area)
      IF dbf_best = DBF()
         verder = .F.
      ENDIF
      area = area + 1
   ENDDO
ENDIF
IF verder = .T.
   RENAME TABLE &dbf_best TO AlleVluc.DBF
ENDIF
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CLOSE..., COPY, COPY FILE, USE

REPLACE

Vervangt de inhoud van opgegeven velden in de huidige tabel door gegevens uit opgegeven uitdrukkingen.

Syntaxis

REPLACE
<*veld1*> WITH <*uitdr1*> [ADDITIVE]
[, <*veld2*> WITH <*uitdr2*> [ADDITIVE] ...]
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[REINDEX]

<*veld1*> WITH <*uitdr1*>

Geeft aan welke velden moeten worden vervangen door gegevens uit opgegeven uitdrukkingen. In een record kunnen meerdere velden worden vervangen door voor elk veld de uitdrukking <*veld n*> WITH <*uitdr n*> op te geven. De uitdrukkingen worden van elkaar gescheiden door komma's.

ADDITIVE

Voegt tekst toe aan het eind van een memoveld in plaats van de bestaande tekst te vervangen. U kunt ADDITIVE alleen gebruiken als het opgegeven veld een memoveld in een dBASE-tabel is.

<*bereik*>

Het aantal records dat moet worden vervangen. RECORD <*n*> geeft een record aan door middel van het recordnummer. NEXT <*n*> geeft *n* records aan, te beginnen bij het huidige record. ALL geeft alle records aan. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR <*voorwaarde1*>
WHILE <*voorwaarde2*>

Bepaalt welke records worden beïnvloed door het commando REPLACE. Een FOR-clausule beperkt REPLACE tot records die voldoen aan <*voorwaarde1*>. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang <*voorwaarde2*> waar is.

REINDEX

Geeft aan dat alle beïnvloede indexen opnieuw moeten worden samengesteld als REPLACE is voltooid.

Beschrijving

Het commando REPLACE overschrijft bestaande gegevens in een opgegeven veld door nieuwe gegevens. Het opgegeven veld kan ook een memoveld zijn. (Wilt u gegevens in

binaire of OLE-velden vervangen, dan gebruikt u REPLACE BINARY of REPLACE OLE). Het veld en de uitdrukking die zijn opgegeven bij WITH moeten echter wel hetzelfde gegevenstype hebben. Bij numerieke velden kan de bij WITH opgegeven uitdrukking groter zijn dan de veldlengte. In dat geval wordt het getal weergegeven in wetenschappelijke notatie. Bij het converteren van memovelden naar tekenvelden kort REPLACE de gegevens in zodat deze binnen de toegewezen veldlengte passen.

Als u meerdere velden wilt wijzigen, moet u voor elk veld dat u wilt vervangen de uitdrukking *<veld n>* WITH *<uitdr n>* opgeven. Als de opties *<bereik>*, WHILE of FOR zijn gebruikt, worden gegevens vervangen in alle records binnen het bereik en in alle records die voldoen aan de opgegeven voorwaarden.

Met de optie ADDITIVE kunt u een tekenreeks toevoegen aan het eind van de bestaande tekst in een memoveld. U kunt aan het begin van de tekenreeks een spatie plaatsen, zodat niet onbedoeld woorden worden aaneengeschakeld.

Wees voorzichtig met het vervangen van gegevens in een tabel met een geopende hoofdindex als u ook de opties *<bereik>*, WHILE of FOR gebruikt. Alle geopende indexbestanden worden automatisch opnieuw samengesteld nadat REPLACE is voltooid. Na het vervangen van gegevens die de waarde van een sleutelveld in de hoofdindex wijzigen, worden het record en de recordaanwijzer onmiddellijk verplaatst naar de positie in de index op basis van de nieuwe waarde. Als de vervanging in een sleutelveld tot gevolg heeft dat een record en de recordaanwijzer voorbij andere records worden verplaatst die binnen het bereik vallen of voldoen aan de opgegeven voorwaarden, worden die records niet vervangen. Als u gegevens wilt vervangen in het sleutelveld van een geïndexeerde tabel, moet u eerst de index sluiten. Vervolgens wijzigt u de gegevens en opent u de index weer met SET INDEX. Tenslotte werkt u de index bij met REINDEX.

Vervangingen in andere velden dan het sleutelveld van de hoofdindex zijn niet van invloed op de volgorde van de index en kunnen zonder problemen worden uitgevoerd met *<bereik>*, WHILE of FOR.

Bij het vervangen van een numerieke of zwevende waarde mag de lengte van de nieuwe waarde niet groter zijn dan de veldlengte, anders geeft dBASE een numerieke foutmelding betreffende de overloop. De veldinhoud wordt vervangen door een benadering van de oorspronkelijke numerieke waarde in exponentiële notatie als dit in het veld past. Is ook dat laatste niet het geval, dan wordt de inhoud vervangen door asterisken en gaan de gegevens verloren.

REPLACE wijzigt gegevens in velden in de huidige tabel tenzij u *alias–>veld* gebruikt om een veld in een aliastabel op te geven.

Als geen relatie is ingesteld tussen de huidige tabel en de aliastabel, kunt u alleen het huidige record in de aliastabel vervangen. Als er wel een relatie is ingesteld, kunt u *<bereik>*, FOR <voorwaarde1> of WHILE *<voorwaarde2>* gebruiken om meerdere records in de aliastabel te vervangen.

REPLACE zonder opties vervangt alleen de inhoud van *<veld1>* in het huidige record door *<uitdr1>*.

Een opgegeven uitdrukking moet hetzelfde gegevenstype hebben als de inhoud van het veld dat door de uitdrukking moet worden vervangen, behalve in het geval van een memoveld, want dat kan worden vervangen door een tekenuitdrukking.

Voorbeeld

In het volgende voorbeeld wordt REPLACE gebruikt om gegevens in TEMP.DBF in te voeren of te wijzigen. TEMP.DBF is een kopie van de tabel Klanten:

```
SET SAFETY OFF
USE Klanten EXCLUSIVE
INDEX ON Klantnr TAG Klantnr
COPY TO Tijd.DBF
USE Tijd
REPLACE ALL Naam with PROPER(Naam)
* Alle namen worden omgezet in een
* beginkapitaal gevolgd door kleine letters.
REPLACE ALL Regio with UPPER(Regio)
* Zet alle regiocodes om in hoofdletters
REPLACE ALL Telefoon with STUFF(Telefoon,AT("-",Telefoon),1,;
    "*")
* Vervangt het eerste streepje in het veld Telefoon
* door een asterisk.
REPLACE ALL NOTES with " Valt onder kantoor Amsterdam.";
    ADDITIVE  FOR Regio="NW"
* Voegt tekst toe in het memoveld Notities in alle
* records met NW in het veld Regio.
GO TOP
BROWSE
CLOSE ALL
SET SAFETY ON
```

Zie ook

APPEND, BLANK, BROWSE, CHANGE, EDIT, REINDEX, REPLACE AUTOMEM, REPLACE BINARY, REPLACE MEMO, REPLACE OLE, SET RELATION, UPDATE

REPLACE AUTOMEM

Velden en records

Plaatst de inhoud van geheugenvariabelen in de overeenkomstige velden van het huidige record in de huidige tabel.

Syntaxis

REPLACE AUTOMEM

Beschrijving

Automatische geheugenvariabelen zijn geheugenvariabelen die dezelfde naam, hetzelfde gegevenstype en dezelfde lengte hebben als de overeenkomstige velden in de huidige tabel. Automatische geheugenvariabelen bevatten gegevens die worden

opgeslagen in recordvelden. U kunt de gegevens in automatische geheugenvariabelen bewerken als geheugenvariabelen in plaats van als veldwaarden, en u kunt de geldigheid van de gegevens controleren voordat deze in de velden worden opgeslagen.

Automatische geheugenvariabelen voor velden in een tabel maakt u met USE...AUTOMEM, CLEAR AUTOMEM en STORE AUTOMEM. Als u nieuwe records wilt toevoegen aan een tabel en de velden wilt vullen met waarden uit overeenkomstige automatische geheugenvariabelen, kunt u APPEND AUTOMEM of INSERT AUTOMEM gebruiken. Met REPLACE AUTOMEM kunt u de velden van bestaande records wijzigen met waarden uit overeenkomstige automatische geheugenvariabelen.

Gebruik REPLACE AUTOMEM om alle velden van een record te wijzigen zonder de naam van de velden op te geven. Als u het commando REPLACE gebruikt, moet u wel de naam opgeven van elk veld dat u wilt wijzigen.

Denk er aan dat een automatische geheugenvariabele en het bijbehorende veld dezelfde naam hebben. Als een commando beschikt over een argument dat zowel een geheugenvariabele als een veld kan zijn, wordt aangenomen dat het argument verwijst naar een veld. Als u de geheugenvariabele wilt gebruiken in plaats van het veld, moet u voor de naam m-> plaatsen.

REPLACE AUTOMEM wijzigt het huidige record. Het is niet mogelijk met dit commando alle records binnen een opgegeven bereik of alle records die voldoen aan een voorwaarde te wijzigen. Dit kunt u doen met het commando REPLACE in combinatie met de opties *<bereik>*, FOR *<voorwaarde>* en WHILE *<voorwaarde>*.

REPLACE AUTOMEM vervangt geen veldgegevens door gegevens uit een geheugenvariabele met dezelfde naam maar een ander gegevenstype. Als u een dergelijke vervanging probeert uit te voeren, verschijnt een foutmelding.

Voorbeeld

In het eerste deel van dit voorbeeld wordt een kopie gemaakt van de tabel Klanten. De kopie wordt doorgegeven aan een filiaal of bijkantoor zodat wijzigingen kunnen worden aangebracht:

```
USE Klanten EXCLUSIVE
INDEX ON SUBSTR(Klantnr,1,1)+ ;
  SUBSTR(Klantnr,2,4) TAG Klantnr
COPY TO filiaal1 FOR Regio = "NW"
! COPY filiaal1.DBF B:   && naar diskette kopiëren
```

Nadat Filiaal1.DBF is bijgewerkt of gewijzigd, kunt u de wijzigingen doorvoeren in de hoofdtabel Klanten. Het volgende programma gebruikt REPLACE AUTOMEM om de huidige veldwaarden uit Filiaal1.DBF over te brengen naar de tabel Klanten:

```
SET TALK OFF
SET EXACT OFF
USE Klanten EXCLUSIVE
INDEX ON SUBSTR(Klantnr,1,1)+ ;
  SUBSTR(Klantnr,2,4) TAG Klantnr
USE Filiaal1 IN 2
SELECT 2
GO TOP
DO WHILE .NOT. EOF()
```

```
  CLEAR AUTOMEM
  STORE AUTOMEM
* Veldwaarden in huidige record (Filiaal1)
* opslaan in automatische geheugenvariabelen
  Zoek = SUBSTR(Klantnr,1,1)+;
    SUBSTR(Klantnr,2,4)
  SELECT 1              && naar tabel Klanten
  SEEK Zoek             && overeenkomstige Klantnr zoeken
  IF FOUND()
    REPLACE AUTOMEM     && Klanten bijwerken
  ENDIF
  SELECT 2              && terug naar Filiaal1.DBF
  SKIP                  && recordaanwijzer verhogen
ENDDO
SELECT 1                && door Klanten bladeren
GO TOP                  && ter controle
BROWSE
CLOSE ALL
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

APPEND AUTOMEM, CLEAR AUTOMEM, INSERT AUTOMEM, REPLACE, STORE AUTOMEM, USE

REPLACE BINARY

Velden en records

Vervangt de inhoud van een binair veld door de inhoud van een ander binair bestand.

Syntaxis

REPLACE BINARY *<naam binair veld>*
  FROM *<bestandsnaam>* | ? | *<bestandsnaamfilter>*
  [TYPE *<binair-typenummer>*]

<naam binair veld>

Het binaire veld in de huidige tabel dat wordt vervangen door de inhoud van *<bestandsnaam>*.

FROM *<bestandsnaam>* | ? | *<bestandsnaamfilter>*

Het bestand dat in het binaire veld in het huidige record moet worden geplaatst. Als u een bestand zonder pad opgeeft, wordt het bestand gezocht in de huidige directory en vervolgens in het pad dat is ingesteld met SET PATH. Als u een bestand zonder extensie opgeeft, wordt de extensie .BMP gebruikt. Extensies als .PCX en .WAV zijn echter ook

toegestaan. De opties ? en *<bestandsnaamfilter>* tonen een dialoogvenster waarin u het bestand opgeeft dat u wilt kopiëren.

TYPE *<binair-typenummer>*

Geeft een nummer aan dat kan worden gebruikt om het type binaire gegevens aan te duiden. Het typenummer kan worden bepaald met de functie BINTYPE( ). Het bereik loopt van 1 tot 32K-1 voor eigen bestandstypen en van 32K tot 64K -1 voor vooringestelde typen (hoewel elk nummer binnen het toegestane bereik kan worden gebruikt).

Binaire-typenummers	Omschrijving
1 tot 32K -1 (32.767)	Eigen bestandstypen
32K (32.768)	.WAV-bestanden
32K + 1 (32.769)	.BMP- en .PCX-bestanden

Beschrijving

Met REPLACE BINARY kunt u een binair bestand naar een binair veld in het huidige record kopiëren. U kunt één binair bestand kopiëren naar elk binair veld in elk record in de tabel.

Hoewel een dBASE-memoveld informatie kan bevatten van een ander type dan tekst, wordt voor de opslag van afbeeldingen, geluid en andere binaire of BLOB-gegevens het gebruik van binaire velden geadviseerd.

Voorbeeld

In het volgende voorbeeld wordt COPY BINARY gebruikt om de bitmap Boa uit DIEREN.DBF te kopiëren naar een bestand dat BOA.BMP wordt genoemd. Met COPY STRUCTURE wordt een nieuwe tabel gemaakt die dezelfde structuur heeft, maar geen records bevat. Vervolgens wordt met APPEND BLANK een nieuw leeg record toegevoegd en wordt met REPLACE BINARY de inhoud van BOA.BMP naar het veld Bmp van het eerste record in DIEREN2.DBF gekopieerd:

```
CLOSE ALL
USE Dieren
GOTO 2
COPY BINARY Bmp TO Boa.BMP
COPY STRUCTURE TO Dieren2
USE Dieren2
b=2**15+1   && 32k+1 voor type .BMP
APPEND BLANK
REPLACE BINARY Bmp FROM "Boa.BMP" TYPE b
EDIT
* Gebruiker kan nu op Bmp klikken en Boa bekijken
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

APPEND MEMO, BINTYPE( ), COPY BINARY, REPLACE MEMO, REPLACE MEMO...FROM, REPLACE OLE, RESTORE IMAGE

REPLACE FROM ARRAY

Velden en records

Verplaatst gegevens uit een array naar de velden in het huidige record van een tabel.

Syntaxis

REPLACE FROM ARRAY *<array-naam>*
[*<bereik>*]
[FOR *<voorwaarde1>*]
[WHILE *<voorwaarde2>*]
[FIELDS *<veldenlijst>*]
[REINDEX]

<array-naam>

De naam van de array waaruit de gegevens afkomstig zijn.

<bereik>

Het aantal records waarin u gegevens wilt vervangen door gegevens uit de opgegeven array. RECORD *<n>* geeft een record aan door middel van het recordnummer. NEXT *<n>* geeft *n* records aan, te beginnen bij het huidige record. ALL geeft alle records aan. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR *<voorwaarde1>*
WHILE *<voorwaarde2>*

Bepaalt welke records worden beïnvloed door REPLACE FROM ARRAY. FOR beperkt REPLACE FROM ARRAY tot records die voldoen aan *<voorwaarde1>*. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang *<voorwaarde2>* waar is.

FIELDS *<veldenlijst>*

Beperkt de bewerking tot velden die worden aangegeven door *<veldenlijst>*.

REINDEX

Bepaalt dat alle niet-hoofdindexen opnieuw worden samengesteld nadat REPLACE FROM ARRAY is voltooid.

Beschrijving

Met REPLACE FROM ARRAY kunt u waarden uit een array verplaatsen naar velden in de huidige tabel. Het nummer van het laatste indexteken van de array bepaalt het aantal velden dat u kunt vervangen. Het nummer van het op een na laatste indexteken van de array bepaalt het aantal records dat u kunt vervangen.

Als u REPLACE FROM ARRAY zonder opties of alleen met de optie FIELDS gebruikt, wordt met vervangen van veldwaarden begonnen in het huidige record. Als meer records zijn opgegeven dan het op een na laatste indexteken van een multidimensionale array, worden recordgegevens vervangen tot geen arraygegevens meer beschikbaar zijn.

Voor het vervangen van veldwaarden in één record kunt u een eendimensionale array gebruiken. Als u bijvoorbeeld DECLARE voorbeeld[3] (een eendimensionale array) gebruikt, worden met de instructie REPLACE FROM ARRAY voorbeeld maximaal drie velden in een record vervangen.

Gebruik een tweedimensionale array als u veldwaarden in meerdere records wilt vervangen. Een tweedimensionale array is geordend als een tabel met rijen voor de records en kolommen voor de velden. Als u bijvoorbeeld DECLARE voorbeeld[2,3] gebruikt, worden met de instructie REPLACE FROM ARRAY voorbeeld maximaal drie velden in twee records vervangen.

De gegevenstypen van de array moeten gelijk zijn aan die van de overeenkomstige velden in de tabel. Als de gegevenstypen van een array-element en een overeenkomstig veld niet overeenkomen, wordt een fout als resultaat gegeven.

Voorbeeld

In het volgende voorbeeld wordt een array gemaakt met drie elementen en worden drie veldwaarden naar die array gekopieerd zodat deze getoond en eventueel kunnen worden gewijzigd. Als de gebruiker de array-waarden wijzigt, wordt REPLACE FROM ARRAY gebruikt om de gewijzigde waarden weer naar het oorspronkelijke record in de tabel te kopiëren:

```
DECLARE BedrArray[3]
* Array met drie elementen maken
CLOSE DATABASES
ON ESCAPE RETURN
USE Bedrijf
SET FIELDS TO Bedrijf, Telefoon, CompCode
DO WHILE .NOT. EOF()
  COPY TO ARRAY BedrArray FIELDS Bedrijf, Telefoon,;
    CompCode NEXT 1
  CLEAR
  ? "Bedrijf", "Telefoon" AT 30, "CompCode" AT 50
  ? "********", "*********" AT 30, "*********" AT 50
  ? BedrArray[1], BedrArray[2] AT 30, BedrArray[3];
    AT 50
  ?
  ?
  ACCEPT "Gegevens juist? (J/N) " TO Antw
  IF UPPER(Antw)="N"
    @ 10,5 SAY "Geef Bedrijf op " GET BedrArray[1]
    @ 11,5 SAY "Geef Telefoon op" GET BedrArray[2]
    @ 12,5 SAY "Geef CompCode op" GET BedrArray[3]
    READ
    REPLACE FROM ARRAY BedrArray
  ENDIF
  SKIP
```

```
ENDDO
CLOSE DATABASES
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

APPEND FROM ARRAY, COPY TO ARRAY, DECLARE

REPLACE MEMO

Velden en records

Vervangt de tekst van een memoveld door de inhoud van een array.

Syntaxis

REPLACE MEMO *<memoveld>* WITH ARRAY *<array-naam>*
[ADDITIVE]

<memoveld>

Het memoveld waar de tekst uit de array moet worden opgeslagen.

WITH ARRAY *<array-naam>*

De array waarvan de inhoud in het memoveld moet worden opgeslagen.

ADDITIVE

Zorgt dat nieuwe tekst wordt toegevoegd aan bestaande tekst. REPLACE MEMO zonder de optie ADDITIVE zorgt dat dBASE voor Windows alle in het memoveld aanwezige tekst overschrijft.

Beschrijving

Met het commando REPLACE MEMO kunt u de tekst in een memoveld vervangen door de inhoud van een array. Elk element van *<array-naam>* bevat de gegevens voor een regel in het opgegeven memoveld. Nadat u tekst hebt opgeslagen in de elementen van de array, kunt u die gegevens met REPLACE MEMO naar een memoveld kopiëren.

Met REPLACE MEMO in combinatie met STORE MEMO kunt u de tekst in een memoveld met behulp van automatische geheugenvariabelen weergeven op een invoerscherm en vervolgens terugplaatsen in het memoveld nadat u klaar bent met wijzigen.

Als u tekst wilt toevoegen aan bestaande tekst in een memoveld of opslaan in een leeg memoveld, kunt u lege tekenwaarden opslaan in de array. Voeg vervolgens tekst toe aan de array en gebruik REPLACE MEMO, eventueel met ADDITIVE als u niet wilt dat bestaande tekst wordt overschreven.

Voorbeeld

In het volgende voorbeeld wordt een array gedefinieerd met een element voor elk veld in de tabel Klanten. Daarna wordt de inhoud van recordnummer 4 naar de array gekopieerd en wordt REPLACE MEMO gebruikt om de inhoud van de array Verplaatsen in het memoveld van recordnummer 10 te plaatsen:

```
SET TALK OFF
USE Klanten
DECLARE Verplaatsen[FLDCOUNT()]
GOTO 4
COPY TO ARRAY Verplaatsen
GOTO 10
REPLACE Notities WITH CHR(13) + ;
  "Kruisverwijzing: " && Gegevens worden overschreven
REPLACE MEMO Notities WITH ARRAY Verplaatsen ADDITIVE
? Notities            && Inhoud van Notities weergeven
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

APPEND MEMO, DECLARE, REPLACE, REPLACE FROM ARRAY, REPLACE MEMO...FROM, STORE MEMO

REPLACE MEMO...FROM

Velden en records

Voegt een tekstbestand in in een memoveld.

Syntaxis

REPLACE MEMO <*memoveld*> FROM <*bestandsnaam*> | ? | <*bestandsnaamfilter*>
[ADDITIVE]

<*memoveld*>

Het memoveld waar het tekstbestand moet worden ingevoegd.

FROM <*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Het tekstbestand dat moet worden ingevoegd. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u opgeeft van welk bestand u de inhoud naar het memoveld wilt kopiëren.

ADDITIVE

Zorgt dat nieuwe tekst wordt toegevoegd aan bestaande tekst. REPLACE MEMO zonder de optie ADDITIVE zorgt dat alle in het memoveld aanwezige tekst wordt overschreven.

Beschrijving

Met het commando REPLACE MEMO...FROM kunt u een tekstbestand invoegen in een memoveld. U kunt in elk memoveld in de tabel één tekstbestand invoegen.

Hoewel een dBASE-memoveld informatie kan bevatten van een ander type dan tekst, wordt voor de opslag van afbeeldingen, geluid en andere binaire gegevens het gebruik van binaire velden geadviseerd. Gebruik OLE-velden voor het koppelen van OLE-documenten uit andere Windows-toepassingen.

Voorbeeld

In dit voorbeeld wordt het bestand Klanten onderzocht. Als er een bestand bestaat met de naam KLNT????.DOC, waar ???? de uit vier tekens bestaande klantcode voorstelt, wordt REPLACE MEMO...FROM gebruikt om het bestand naar het memoveld Notities te kopiëren:

```
CLOSE ALL
USE Klanten
SCAN
  BN="KLNT"+KlantNr+".Doc"
  IF FILE("&BN")
    REPLACE MEMO Notities FROM &BN
  ELSE
    BLANK FIELD Notities
  ENDIF
ENDSCAN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

COPY MEMO, REPLACE BINARY, REPLACE MEMO, REPLACE OLE

REPLACE OLE

Velden en records

Voegt een OLE-document in in een OLE-veld.

Syntaxis

REPLACE OLE <*OLE-veldnaam*>
  FROM <*bestandsnaam*> | ? | <*bestandsnaamfilter*>
  [LINK]

<OLE-veldnaam>

Het veld waar een OLE-document moet worden ingevoegd.

FROM <*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Het bestand met het OLE-document. De extensie moet ook worden opgegeven. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u het bestand opgeeft.

LINK

LINK levert een aanwijzer naar het OLE-document. Standaard wordt het OLE-document in het opgegeven OLE-veld ingesloten.

Beschrijving

Met REPLACE OLE kunt u de inhoud van een OLE-veld vervangen door de inhoud van een OLE-document. U kunt het OLE-document insluiten in het OLE-veld (de standaardinstelling) of een koppeling tussen het OLE-document en het OLE-veld tot stand brengen.

Als u een koppeling tot stand brengt, bevat het OLE-veld alleen een verwijzing naar het OLE-document. Zolang het OLE-dcument is opgeslagen in dezelfde lokatie, bevat het OLE-veld de meest recente versie van het document.

Als u het OLE-document insluit, bevat het OLE-veld een kopie van het document. Er zijn geen koppelingen tussen het veld en het document, zodat een wijziging in het oorspronkelijke OLE-document niet wordt gereflecteerd in het ingesloten document in het OLE-veld.

Voorbeeld

In het volgende voorbeeld wordt het gebruik van REPLACE OLE gedemonstreerd. Er wordt een nieuw record toegevoegd aan de tabel Afbeeld. Het veld Naam wordt vervangen door "AirBrlnd" en het bitmap-bestand AIRBRLND.BMP wordt in het veld BitMapOle geplaatst:

```
USE Afbeeld
APPEND BLANK
REPLACE Naam WITH "AirBrlnd"
REPLACE OLE BitMapOle FROM "AirBrlnd.BMP"
```

Als AIRBRLND.BMP bestaat, kunt u de afbeelding bekijken door het record te tonen met EDIT of BROWSE en op het veld BitMapOle te klikken.

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CLASS OLE, DEFINE

REPLICATE( ) Tekenreeksgegevens

Resulteert in een tekenreeks die bestaat uit een opgegeven aantal kopieën van een andere tekenreeks.

Syntaxis

REPLICATE(<*Tuitdr*> | <*memoveld*>, <*Nuitdr*>)

<*Tuitdr*> | <*memoveld*>

De tekenreeks of het memoveld dat u wilt herhalen.

<*Nuitdr*>

Het aantal keren dat de tekenreeks of het memoveld moet worden herhaald.

Beschrijving

REPLICATE( ) resulteert in een tekenreeks die bestaat uit een tekenuitdrukking of memoveld dat een opgegeven aantal malen is herhaald. Als de tekenreeks die als resultaat wordt gegeven, langer is dan 32.766 tekens, wordt een fout als resultaat gegeven. Vandaar dat <*Nuitdr*> kleiner moet zijn dan 32.766 gedeeld door het aantal tekens in de tekenuitdrukking of het memoveld.

Als de tekenuitdrukking een lege tekenreeks is of als het memoveld leeg is, resulteert REPLICATE( ) in een lege tekenreeks. Ook als u voor <*Nuitdr*> 0 opgeeft, resulteert REPLICATE( ) in een lege tekenreeks. Als <*Nuitdr*> kleiner is dan 0, wordt een foutmelding getoond.

Een reeks spaties kunt u samenstellen met SPACE( ).

Voorbeeld

In het volgende voorbeeld wordt REPLICATE( ) gebruikt om enkele verschillende tekenreeksen te maken:

```
? REPLICATE("+",10)          && Geeft "++++++++++" als resultaat
? REPLICATE("",10)           && Geeft "" als resultaat
? REPLICATE("-",10)          && Geeft "----------" als resultaat
? REPLICATE("dBASE",2)       && Geeft "dBASEdBASE" als resultaat
maxi = REPLICATE("A",32766)
? LEN(maxi)                  && Geeft 32766 als resultaat
? REPLICATE("dBASE voor Windows!",1724)
* 32766/19 tekens in de reeks = 1724
```

Overdraagbaarheid

Het argument <*memoveld*> wordt niet ondersteund in dBASE IV of dBASE III PLUS. Zowel dBASE IV als dBASE III PLUS beperken het resultaat van REPLICATE( ) tot 254 tekens.

Zie ook

SPACE( )

REPORT FORM

Invoer/uitvoer

Genereert een rapport en geeft het weer of drukt het af. Het rapport wordt gemaakt met de rapportindeling die is opgeslagen in een opgegeven rapportbestand, en met informatie uit records in de huidige tabel.

Syntaxis

REPORT FORM <*bestandsnaam1*> | ? | <*bestandsnaamfilter1*>
[<*bereik*>] [FOR <*voorwaarde1*>] [WHILE <*voorwaarde2*>]
[CROSSTAB]
[HEADING <*Tuitdr*>]
[NOEJECT]
[PLAIN]
[SUMMARY]
[TO FILE <*bestandsnaam2*> | ? | <*bestandsnaamfilter2*>] | [TO PRINTER]

<bestandsnaam1> | ? | <bestandsnaamfilter>

Het rapportindelingsbestand dat moet worden gebruikt. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, wordt gezocht naar een bestand met een van de volgende extensies: .RPT, .FRG of .FRM (in deze volgorde). Als u CROSSTAB opgeeft, wordt gezocht naar een bestand met een van de volgende extensies: .RPC, .FRG of .FRM (in deze volgorde).

<bereik>

Het aantal records dat voor het rapport moet worden gebruikt. RECORD <*n*> geeft een record aan door middel van het recordnummer. NEXT <*n*> geeft *n* records aan, te beginnen bij het huidige record. ALL geeft alle records aan. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR *<voorwaarde1>*
WHILE *<voorwaarde2>*

Bepaalt welke records worden beïnvloed door REPORT FORM. FOR beperkt REPORT FORM tot records die voldoen aan <*voorwaarde1*>. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang <*voorwaarde2*> waar is.

CROSSTAB

Geeft aan dat het rapport is gemaakt met het dialoogvenster **Kruistabulatie**.

HEADING <*Tuitdr*>
Neemt op elke pagina een tekenuitdrukking, <*Tuitdr*>, op als koptekst. HEADING heeft geen gevolgen in combinatie met PLAIN.

NOEJECT
Voorkomt een paginadoorvoer voordat met afdrukken wordt begonnen.

PLAIN
Onderdrukt paginanummers en datums op de pagina's van het rapport. Een eventuele koptekst verschijnt alleen op de eerste pagina.

SUMMARY
Neemt alleen de totalen van groepen en de subtotalen van subgroepen op in een rapport. De individuele inhoud van records binnen elke groep en subgroep wordt niet afgedrukt. De groepen en subgroepen worden gedefinieerd door de rapportindeling.

TO FILE <*bestandsnaam2*> | ? | <*bestandsnaamfilter*>
Stuurt uitvoer naar het tekstbestand <*bestandsnaam2*>. dBASE kent standaard de extensie .TXT toe aan <*bestandsnaam2*> en slaat het bestand op in de huidige directory. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u de naam en directory van het doelbestand kunt opgeven.

TO PRINTER
Stuurt uitvoer naar de printer.

Beschrijving
Met REPORT FORM kunt u rapporten afdrukken of weergeven in een indeling die u met CREATE REPORT of MODIFY REPORT hebt gedefinieerd in Rapportontwerp. Zie de Crystal Reports documentatie voor meer informatie over werken met Rapportontwerp. Als u geen <*bereik*>, WHILE <*voorwaarde1*> of FOR <*voorwaarde2*> opgeeft, worden de rapportspecificaties voor elk record afgedrukt (op volgorde van recordnummer of index).

Als u een rapport afdrukt of weergeeft dat gegroepeerde gegevens of groepssubtotalen bevat, moet de huidige tabel zijn gesorteerd of moet de hoofdindex actief zijn. Het gesorteerde bestand of de index moet zijn gerangschikt op basis van de waarde in het veld waarop de gegevens zijn gegroepeerd.

REPORT FORM zonder de optie TO FILE of TO PRINTER toont het rapport in het commandovenster of in het huidige door de gebruiker gedefinieerde venster.

Voorbeeld
In het volgende voorbeeld wordt de database Bedrijf geopend en wordt een rapport gegenereerd op basis van de rapportdefinitie Bedrrap1:

```
CLOSE DATABASES
USE Bedrijf
REPORT FORM Bedrrap1 TO PRINT
```

Overdraagbaarheid

De optie *<bestandsnaamfilter>* wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CREATE REPORT

RESOURCE( ) — Programmeren in Windows

Resulteert in een tekenreeks uit een DLL-bestand.

Syntaxis

RESOURCE(*<bronidentificatie>*, *<DLL-bestandsnaam Tuitdr>*)

<bronidentificatie>

Een numerieke waarde die de tekenreeks aangeeft.

<DLL-bestandsnaam>

De naam van het .DLL-bestand.

Beschrijving

Met RESOURCE( ) kunt u een tekenreeks genereren uit een resource in een DLL-bestand. De tekenreeks moet kleiner zijn dan 32K. Als de tekenreeks langer is, wordt die ingekort.

RESOURCE( ) kan worden gebruikt om applicaties geschikt te maken voor meerdere talen zonder de programmacode te wijzigen. U kunt bijvoorbeeld alle tekenreeksen in een applicatie opslaan in een DLL-bestand. De applicatie kan de tekenreeksen tijdens de uitvoering uit het bestand halen met de functie RESOURCE( ). Als de applicatie dan moet worden vertaald, hoeven alleen de tekenreeksen in het DLL-bestand te worden vertaald. De vertaling moet vervolgens worden opgeslagen in een nieuw DLL-bestand en wel in dezelfde volgorde en met dezelfde resource-identificaties als in het oorspronkelijke DLL-bestand. Zie EXTERN, LOAD DLL en Hoofdstuk 25 in *Programmeren* voor meer informatie over DLL-bestanden.

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Voorbeeld

In het volgende voorbeeld wordt gedemonstreerd hoe RESOURCE( ) kan worden gebruikt voor het veranderen van taal in een applicatie:

```
#define NEDERLANDS 1
#define ENGELS 2
```

```
ApplTaal = ENGELS
MijnGroet = RESOURCE(ENGELS,"groeten.dll")
CLEAR
? MijnGroet
```

Zie ook

EXTERN, LOAD DLL, RELEASE DLL

RESTORE — Geheugenvariabelen

Kopieert de geheugenvariabelen uit een opgegeven bestand op schijf naar het geheugen.

Syntaxis

RESTORE FROM *<bestandsnaam>* | ? | *<bestandsnaamfilter>*
[ADDITIVE]

<bestandsnaam> | ? | *<bestandsnaamfilter>*

Het bestand met geheugenvariabelen dat moet worden geladen. RESTORE FROM ? en RESTORE FROM *<bestandsnaamfilter>* tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, wordt de extensie .MEM gebruikt.

ADDITIVE

Zorgt dat bestaande geheugenvariabelen behouden blijven als RESTORE wordt uitgevoerd.

Beschrijving

Gebruik RESTORE en SAVE om belangrijke geheugenvariabelen te lezen en op te slaan. Standaard worden alle lokale variabelen gewist als het programma is afgelopen waarin deze zijn gedefinieerd. Publieke variabelen worden daarentegen gewist als dBASE wordt afgesloten. Als u variabelen op een later tijdstip opnieuw wilt gebruiken, kunt u deze met SAVE opslaan in een geheugenbestand. U kunt deze variabelen weer in het geheugen laden met RESTORE.

Zonder de optie ADDITIVE wist RESTORE alle bestaande door de gebruiker ingestelde geheugenvariabelen voordat de waarden uit het geheugenbestand worden ingelezen. Met ADDITIVE kunt u de variabelen behouden die al in het geheugen aanwezig zijn.

Opmerking Als u ADDITIVE gebruikt en een ingelezen variabele heeft dezelfde naam als een bestaande variabele, wordt de bestaande variabele overschreven.

Als u RESTORE gebruikt in het commandovenster, worden alle ingelezen variabelen publiek gemaakt. Als RESTORE wordt aangetroffen in een programmabestand, zijn alle ingelezen variabelen lokaal binnen dat programma.

Voorbeeld

In het volgende voorbeeld wordt de naam van een bedrijf opgeslagen in de variabele Start als de gebruiker een bladervenster verlaat. De waarde in Start wordt vervolgens met het commando SAVE opgeslagen in een extern .MEM-bestand. Als het programma opnieuw wordt uitgevoerd, wordt RESTORE FROM gebruikt om de variabele Start te lezen. Met SEEK wordt de recordaanwijzer daarna bij het betreffende bedrijf geplaatst zodat het bladeren kan worden voortgezet vanaf dezelfde positie:

```
SET TALK OFF
SET SAFETY OFF
USE Bedrijf EXCLUSIVE
INDEX ON Bedrijf TAG Bedrijf
IF FILE("Lst_Wrgv.MEM")
  RESTORE FROM Lst_Wrgv       && Start lezen
  SEEK Start
  BROWSE
ELSE
  GO TOP
  BROWSE
ENDIF
STORE Bedrijf TO Start
SAVE TO Lst_Wrgv.MEM
CLOSE ALL
SET TALK ON
SET SAFETY OFF
```

Overdraagbaarheid

De opties ? en <*bestandsnaamfilter*> worden niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CLEAR MEMORY, RELEASE, SAVE, SET PATH, STORE

RESTORE IMAGE

Objecten

Geeft een afbeelding weer die is opgeslagen in een bestand of een binair veld.

Syntaxis

RESTORE IMAGE FROM
  <*bestandsnaam*> | ? | <*bestandsnaamfilter*> | BINARY <*binair veld*>
  [TIMEOUT <*Nuitdr*>]
  [TO PRINTER]
  [[TYPE] PCX]

FROM <*bestandsnaam*> | ? | <*bestandsnaamfilter*> | BINARY <*binair veld*>

Geeft het bestand of binaire veld aan waarin de afbeelding is opgeslagen. RESTORE IMAGE FROM ? en RESTORE IMAGE FROM <*bestandsnaamfilter*> tonen het

dialoogvenster **Bronbestand openen**, waarin de gebruiker een bestand kan kiezen. *<bestandsnaam>* is de naam van een afbeeldingsbestand. Tenzij u een andere waarde opgeeft, gebruikt RESTORE IMAGE standaard de extensie .BMP. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. RESTORE IMAGE FROM BINARY *<binair veld>* geeft een afbeelding weer die is opgeslagen in een binair veld. Met het commando REPLACE BINARY slaat u een afbeelding op in een binair veld.

TIMEOUT *<Nuitdr>*
Geeft aan hoeveel seconden de afbeelding op het scherm wordt weergegeven.

TO PRINTER
Stuurt de afbeelding niet alleen naar het scherm, maar ook naar de printer.

[TYPE] PCX
Geeft aan dat de afbeelding de PCX-indeling heeft. In dat geval wordt de extensie .PCX gebruikt als geen extensie is opgegeven. Het woord TYPE is optioneel.

Beschrijving
Met RESTORE IMAGE kunt u een bitmap- of .PCX-afbeelding op het scherm tonen. De afbeelding wordt weergegeven in een venster.

Opmerking Afbeeldingen kunnen alleen worden getoond als in de computer een grafische kaart is geïnstalleerd.

Om met de optie TO PRINTER een afbeelding te kunnen afdrukken, moet u over een printer beschikken die grafische gegevens kan verwerken en afdrukken.

Voorbeeld
In het volgende voorbeeld wordt een formulier gedefinieerd met een lijstvak waarin een vliegtuigmodel kan worden gekozen. Met RESTORE IMAGE wordt de afbeelding uit het memoveld Afbeelding in het geselecteerde record getoond:

```
CLOSE ALL
SET TALK OFF
SET PROCEDURE TO PROGRAM(1) ADDITIVE
USE Vliegtg ORDER VLIEGTUIG IN SELECT()
SELECT Vliegtg
DEFINE FORM VT              ;
   PROPERTY                 ;
      Top 5,                ;
      Left 60,              ;
      Height 13,            ;
      Width 30,             ;
      Text "Vliegtuig",     ;
      Sizeable .T.
DEFINE LISTBOX Model OF VT          ;
   PROPERTY                         ;
      Top 2,                        ;
      Left 6,                       ;
```

```
        Height 7,                          ;
        Width 18,                          ;
        DataSource "FIELD Vliegtg->Vliegtuig"
ON SELECTION FORM VT Foto
OPEN FORM VT

FUNCTION Foto
RESTORE IMAGE FROM BINARY Afbeelding
RETURN .T.
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DEFINE, REPLACE BINARY

RESTORE SCREEN

Invoer/uitvoer

Herstelt in het resultatenpaneel van het commandovenster de eerder met SAVE SCREEN opgeslagen inhoud van het resultatenpaneel. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV.

Zie Help voor meer informatie over de syntaxis van CLEAR SCREENS.

RESTORE WINDOW

dBASE IV-vensters

Haalt vensterdefinities op uit een vensterbestand en laadt deze in het geheugen. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows start u het programma, de procedure of het bibliotheekbestand dat opgeslagen formulierdefinities bevat.

Zie Help voor meer informatie over de syntaxis van RESTORE WINDOW. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

RESUME

Foutafhandeling en testen op fouten

Gaat verder met de uitvoering van een programma. De uitvoering wordt hervat met de commandoregel die volgt op de regel waar de uitvoering werd onderbroken.

Syntaxis

RESUME

Beschrijving

RESUME zorgt dat een opgeschort programma verder wordt uitgevoerd. U kunt de uitvoering van een programma opschorten met SUSPEND. Als u geen waarde hebt toegewezen aan ON ERROR, kunt u een programma ook opschorten als een fout optreedt.

Als u de uitvoering van een programma wilt hervatten, geeft u in het commandovenster RESUME op. De uitvoering van het programmabestand wordt voortgezet op de regel die volgt op de regel waar het programma werd opgeschort. Als u de regel die de fout veroorzaakte, opnieuw wilt uitvoeren, bijvoorbeeld omdat u de oorzaak van de fout hebt weggenomen, kunt u de programmaregel typen voordat u het commando RESUME geeft.

Voorbeeld

Zie SUSPEND voor een voorbeeld van het gebruik van RESUME nadat in het programma het commando SUSPEND is uitgevoerd.

Zie ook

CANCEL, ON ERROR, RETRY, RUN, SUSPEND

RETRY

Foutafhandeling en testen op fouten

Geeft de besturing vanuit een subroutine terug aan de regel in de aanroepende routine of de regel in het commandovenster waarmee de subroutine werd aangeroepen.

Syntaxis

RETRY

Beschrijving

Met RETRY kunt u een instructie opnieuw uitvoeren, bijvoorbeeld een regel die een fout heeft veroorzaakt. RETRY geeft de besturing terug aan de aanroepende instructie. RETRY wist de geheugenvariabelen die door de subroutine zijn gemaakt.

RETRY is alleen geldig in programmabestanden.

U kunt RETRY gebruiken in combinatie met ON ERROR om de gebruiker de kans te geven de oorzaak van een fout weg te nemen. Als u RETRY in combinatie met ON ERROR opgeeft, wordt ERROR( ) opnieuw wordt ingesteld op 0.

Voorbeeld

In het volgende voorbeeld wordt de procedure Herstel gebruikt als met ON ERROR een fout is waargenomen. Als de instructie USE Klanten wordt gebruikt terwijl de tabel Klanten al is geopend in een ander werkgebied, wordt een fout als resultaat gegeven. In de procedure Herstel wordt CLOSE DATABASES gebruikt om te zorgen dat alle

tabellen worden gesloten. Vervolgens wordt de besturing met RETRY teruggegeven aan de instructie met USE:

```
ON ERROR DO Herstel
USE Klanten EXCLUSIVE
BROWSE
ON ERROR
CLOSE DATABASES

PROCEDURE Herstel
WAIT "Er is een fout opgetreden.;
  Druk op een toets om het nog eens te proberen.."
CLOSE DATABASES
RETRY
```

Zie ook

ERROR( ), MESSAGE( ), ON ERROR, RESUME, RETURN

RETURN

Programma's

Beëindigt de uitvoering van een programma, procedure of door de gebruiker gedefinieerde functie. De besturing wordt teruggegeven aan de aanroepende routine (programma, procedure of door de gebruiker gedefinieerde functie) of aan het commandovenster.

Syntaxis

RETURN
[<*resultaatwaarde*> | TO MASTER | TO <*routinenaam*>]

<resultaatwaarde>

De waarde die door een procedure of door de gebruiker gedefinieerde functie als resultaat wordt gegeven aan de aanroepende routine of aan het commandovenster. RETURN <*resultaatwaarde*> moet de laatste regel in de definitie van een procedure of door de gebruiker gedefinieerde functie zijn. In het geval van een door de gebruiker gedefinieerde functie is <*resultaatwaarde*> verplicht.

TO MASTER

Geeft de besturing terug aan de routine van het hoogste niveau of aan het commandovenster. Als een routine een andere routine aanroept, wordt elke aanroep op de *aanroep-stack* geplaatst. Dat is een lijst van routines en subroutines die nog moeten worden voortgezet. Met TO MASTER kunt u de besturing teruggeven aan de eerste routine op de stack (de hoofdprocedure). Alle subroutines in de aanroep-stack worden beëindigd en alle niet-publieke variabelen op elk niveau worden vrijgegeven.

Als RETURN TO MASTER wordt uitgevoerd in de hoofdprocedure, wordt de besturing teruggegeven aan het commandovenster. Als de oorsprong van de aanroep-stack een opgeschorte prompt in het commandovenster is, wordt de besturing aan die prompt teruggeven.

TO <*routinenaam*>

Geeft de besturing terug aan de opgegeven aanroepende routine. Alle tussenliggende subroutines op de aanroep-stack worden beëindigd en alle niet-publieke variabelen van die subroutines worden vrijgegeven. (Zie de beschrijving van TO MASTER voor een uitleg van de *aanroep-stack*.)

Bijzondere gevallen worden als volgt verwerkt:

- Als <*routinenaam*> de naam van de huidige routine is, wordt een gewone RETURN uitgevoerd.
- Als <*routinenaam*> meer dan eens voorkomt in de aanroep-stack, wordt de besturing teruggegeven aan de routine met die naam die als laatste op de stack is geplaatst.
- Als <*routinenaam*> overeenkomt met *master*, heeft de optie TO MASTER de hoogste prioriteit en wordt de besturing teruggegeven aan de aanroepende routine op het hoogste niveau.
- Als <*routinenaam*> niet op de aanroep-stack voorkomt, veroorzaakt RETURN een runtime-fout.

Beschrijving

Als een procedure wordt beëindigd, wordt de besturing automatisch teruggegeven aan de aanroepende routine. In procedures kunt u RETURN gebruiken om de routine vanaf een ander punt te verlaten dan het eind van de routine. Aan het eind van een door de gebruiker gedefinieerde functie moet u RETURN gebruiken om het resultaat van de functie te kunnen doorgeven.

Gebruik in programma's RETURN om de besturing terug te geven aan aanroepende routines of aan het commandovenster. De verwerking van commando's wordt gewoonlijk als volgt voortgezet:

- Als de routine is aangeroepen met een DO-instructie, wordt de besturing doorgegeven aan de regel na de DO-instructie.
- Als de routine een door de gebruiker gedefinieerde functie is, die rechtstreeks is aangeroepen als een commando of als deel van een uitdrukking (niet met DO), wordt de uitvoering van de instructie met de aanroep voortgezet.
- Als u de optie TO MASTER of TO <*routinenaam*> van het commando RETURN hebt gebruikt, wordt de besturing teruggeven aan de regel na de regel die als laatste is uitgevoerd in de hoofdroutine of in <*routinenaam*>.

Het commando RETURN heeft tot gevolg dat de volgende handelingen worden verricht:

- Alle niet-publieke geheugenvariabelen die zijn gedefinieerd in de routine waaruit wordt teruggekeerd (en eventueel in alle tussenliggende routines op de aanroep-stack) worden vrijgegeven.
- De functie ERROR( ) wordt opnieuw ingesteld op 0.

- De status en waarde van variabelen wordt aangepast aan de routine waarnaar wordt teruggekeerd. Lokale variabelen binnen die routine worden opnieuw ingesteld op de oorspronkelijke waarden.

U kunt de uitvoering van een routine ook beëindigen met CANCEL, maar in dat geval wordt de besturing altijd teruggegeven aan het commandovenster.

Voorbeeld

In de volgende voorbeelden worden verschillende toepassingen van RETURN gedemonstreerd. In GaTerug kunt u terugkeren naar het hoofdprogramma of naar een geopende procedure. De instructie met RETURN in IsKlaar( ) resulteert in een waarde.

```
DO AfdrukkenNaar With "LPT1:"
DO GaTerug With .t.
DO GaTerug With .f.
Antwoord= IsKlaar()

PROCEDURE AfdrukkenNaar
PARAMETER UitvoerApp
* ...
RETURN

PROCEDURE GaTerug
Parameter Klaar
IF Klaar
   RETURN TO MASTER
ELSE
   RETURN TO Rapportmenu
ENDIF

FUNCTION IsKlaar
RETURN .T.
```

Overdraagbaarheid

De opties <*resultaatwaarde*> en TO <*routinenaam*> worden niet ondersteund in dBASE III PLUS.

Zie ook

CANCEL, DO, ERROR( ), FUNCTION, LOCAL, PRIVATE, PROCEDURE, PUBLIC, QUIT, RETRY, STATIC

RIGHT( )

Tekenreeksgegevens

Resulteert in tekens vanaf het eind van een tekenreeks of memoveld.

Syntaxis

RIGHT(<*Tuitdr*> | <*memoveld*>, <*Nuitdr*>)

<Tuitdr> | <memoveld>
De tekenreeks of het memoveld waaruit tekens moeten worden opgehaald.

<Nuitdr>
Het aantal tekens dat uit de tekenreeks of het memoveld moet worden opgehaald.

Beschrijving

Met RIGHT( ) kunt u een opgegeven aantal tekens vanaf de rechterkant van een tekenuitdrukking of memoveld ophalen. De maximumlengte van de opgehaalde tekenreeks is 32.766 tekens (de maximumlengte van een tekenreeks).

Als het opgegeven aantal tekens (<*Nuitdr*>) groter is dan het aantal tekens in de opgegeven tekenreeks of het opgegeven memoveld, resulteert RIGHT( ) in de volledige tekenreeks zonder spaties toe te voegen om de opgegeven lengte vol te maken. Als <*Nuitdr*> kleiner dan of gelijk aan 0 is, resulteert RIGHT( ) in een lege tekenreeks.

Als met RIGHT( ) tekens uit een memoveld worden opgehaald, worden de tekens voor de combinatie van regelterugloop en regeldoorvoer (CR/LF) geteld als twee tekens.

Voorbeeld

In het volgende voorbeeld wordt RIGHT( ) gebruikt om een deel van een tekenreeks als resultaat te geven:

```
? RIGHT("dBASE",1)          && Geeft "E" als resultaat
? RIGHT("dBASE",3)          && Geeft "ASE" als resultaat
? RIGHT("dBASE",9)          && Geeft "dBASE" als resultaat
? RIGHT("dBASE",0)          && Geeft "" als resultaat
```

In het volgende voorbeeld wordt RIGHT( ) gebruikt om een bestand te rangschikken op de laatste vier tekens in het veld Postcode:

```
USE Klanten EXCLUSIVE
INDEX ON RIGHT(Postcode,4) TAG PoCo
LIST FIELDS Naam, Postcode
CLOSE ALL
```

Overdraagbaarheid

Het argument <*memoveld*> wordt niet ondersteund in dBASE IV of dBASE III PLUS. Zowel dBASE IV en dBASE III PLUS beperken de waarde die door RIGHT( ) wordt opgehaald tot 254 tekens.

Zie ook

AT( ), LEFT( ), RAT( ), SUBSTR( )

RLOCK( )

Vergrendelt het huidige record of een opgegeven lijst van records in de huidige tabel of een opgegeven aliastabel, en resulteert in .T. (waar) als de bewerking slaagt.

Syntaxis

RLOCK([<*lijst Tuitdr*>] | [<*bladwijzerlijst*>]
[, <*alias*>])

<*lijst Tuitdr*>

De door komma's gescheiden lijst van te vergrendelen recordnummers.

<*bladwijzerlijst*>

De lijst van bladwijzers (recordindicaties) die zijn opgehaald met BOOKMARK( ). Met bladwijzers kunt u records aanduiden in niet-dBASE-tabellen, zoals Paradox-tabellen, die niet beschikken over recordnummers in de natuurlijke volgorde. Plaats komma's tussen de bladwijzers.

<*alias*>

Een werkgebiednummer (van 1 tot 255) of -letter (van A tot J) of aliasnaam. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens. Als u geen <*alias*> opneemt, gebruikt RLOCK( ) de huidige tabel.

U hoeft geen recordnummers of bladwijzers op te geven als u een waarde wilt opgeven voor <*alias*>. Als u echter recordnummers of bladwijzers hebt opgegeven, moet u voor <*alias*> een komma (,) plaatsen.

Beschrijving

Met RLOCK( ) kunt u het huidige record of een lijst van records in de huidige tabel of een aliastabel vergrendelen. Als u geen argumenten doorgeeft aan RLOCK( ), wordt het huidige record in de huidige tabel vergrendeld. Als u alleen een <*alias*> doorgeeft aan RLOCK( ), wordt het huidige record in de aliastabel vergrendeld. Als met RLOCK( ) *alle* opgegeven records zijn vergrendeld, wordt .T. als resultaat gegeven. U kunt met RLOCK( ) maximaal 100 records in elke geopende tabel op het werkstation vergrendelen.

U kunt een met RLOCK( ) vergrendeld record bekijken en wijzigen. Andere gebruikers kunnen het record wel bekijken, maar niet wijzigen. Als u een record hebt vergrendeld met RLOCK( ), blijft dit vergrendeld tot u een van de volgende handelingen verricht:

- U geeft het commando UNLOCK.
- U drukt op *Ctrl-L* (de toetscombinatie voor schakelen tussen vergrendelen en vergrendeling opheffen) in een blader- of wijzigvenster terwijl de recordaanwijzer op een record staat dat u hebt vergrendeld met RLOCK( ).
- U sluit de tabel.

RLOCK( ) komt overeen met FLOCK( ), maar met FLOCK( ) vergrendeld u de volledige tabel. Gebruik FLOCK( ) als u exclusief toegang moet hebben tot een volledige tabel of gerelateerde tabellen (als u bijvoorbeeld meerdere door een gemeenschappelijke sleutel gerelateerde tabellen wilt wijzigen) of als u meer dan 100 records tegelijk wilt wijzigen.

Bij alle commando's waarmee gegevens in een tabel worden gewijzigd, wordt geprobeerd een automatische record- of bestandsvergrendeling aan te brengen. Als een automatische record- of bestandsvergrendeling niet slaagt, wordt een fout als resultaat gegeven. U kunt RLOCK( ) gebruiken voor het onderscheppen van acties door te testen op de waarde van het resultaat in plaats van op een foutconditie.

Onder de volgende omstandigheden slaagt RLOCK( ) er niet in de opgegeven records te vergrendelen:

- Een andere gebruiker heeft (expliciet of automatisch) het huidige record of een van de records in *<lijst Tuitdr>* of *<bladwijzerlijst>* vergrendeld
- Een andere gebruiker heeft (expliciet of automatisch) de huidige tabel of de opgegeven aliastabel vergrendeld

Als RLOCK( ) er niet onmiddellijk in slaagt de opgegeven records te vergrendelen, verschijnt standaard de vraag of u het nog een keer wilt proberen of de bewerking wilt annuleren. Met SET REPROCESS kunt u opgeven hoeveel nieuwe pogingen moeten worden ondernomen. Als u annuleren kiest, resulteert RLOCK( ) in .F.

Als u met SET RELATION een relatie instelt met een *hoofdtabel* en vervolgens met RLOCK( ) een record in de tabel vergrendelt, wordt geprobeerd alle *subrecords* in *subtabellen* te vergrendelen. Zie SET RELATION voor meer informatie over het instellen van relaties tussen tabellen.

RLOCK( ) is onderling uitwisselbaar met LOCK( ).

Voorbeeld

In het volgende voorbeeld wordt een lus doorlopen tot het tiende record kan worden vergrendeld met RLOCK( ) of tot de gebruiker besluit de pogingen te annuleren. Als de vergrendeling is aangebracht, wordt een subroutine, BedrWijz, aangeroepen om het record te wijzigen:

```
RecordGelezen = .t.
SET REPROCESS TO 20  && 20 vergrendelingspogingen
USE Bedrijf
GO 10
Nogmaals = .t.
DO WHILE Nogmaals
  IF RLOCK()       && Bestand vergrendeld?
    DO BedrWijz    && Record wijzigen
    Nogmaals = .f.
    UNLOCK         && Andere gebruikers mogen
                   && bestand weer wijzigen.
  ELSE             && RLOCK() geeft .F. als resultaat
    CLEAR
    Wait "Vergrendeling niet geslaagd. Nieuwe poging? (J/N) ";
      to mOpnieuw
    IF UPPER(mOpnieuw)="N"
```

```
        Nogmaals = .f.
      ENDIF
      RecordGelezen = .f.
    ENDIF
  ENDDO
  USE
  SET REPROCESS TO 0  && Standaardinstelling
```

Met LOCK( ) kunt u meerdere records vergrendelen:

```
USE Spellen ALIAS Pret
* ...
Pythagoras= LOCK("8,15,17","Pret")
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. De optie *<bladwijzerlijst>* wordt niet ondersteund in dBASE IV.

Zie ook

FLOCK( ), SET LOCK, SET RELATION, SET REPROCESS, UNLOCK

ROLLBACK( )

Gedeelde gegevens

Beëindigt een transactie die is gestart met BEGINTRANS( ) zonder wijzigingen op te slaan in de geopende bestanden. Resulteert in .T. als de transactie met succes is teruggedraaid.

Syntaxis

ROLLBACK([*<databasenaam Tuitdr>*])

<databasenaam Tuitdr>

De naam van de database waarin de transactie moet worden geannuleerd.

- Als de transactie is begonnen met BEGINTRANS(*<databasenaam Tuitdr>*), moet u de instructie ROLLBACK(*<databasenaam Tuitdr>*) gebruiken. Als u alleen maar ROLLBACK( ) gebruikt, wordt de instructie met ROLLBACK( ) genegeerd.
- Als de transactie is begonnen met BEGINTRANS( ), is *<databasenaam Tuitdr>* een optioneel argument bij ROLLBACK( ). Als u het argument opneemt, moet dit naar dezelfde database verwijzen als de instructie met SET DATABASE TO die aan BEGINTRANS( ) voorafging.

Beschrijving

Met ROLLBACK( ) kunt u de open transactie beëindigen en alle geopende bestanden herstellen naar de oorspronkelijke staat (voor BEGINTRANS( ) werd uitgevoerd). Gebruik COMMIT( ) om een transactie te beëindigen en de wijzigingen naar de bestanden te schrijven. Zie BEGINTRANS voor meer informatie over transacties.

Voorbeeld

In het volgende voorbeeld wordt een transactie gestart met BEGINTRANS( ). Er wordt een multi-user-versie van Bedrijf.dbf geopend en er wordt een poging ondernomen de waarden in het veld Verk_Totnu gelijk aan 0 te maken. Eventuele fouten worden onderschept met ON ERROR. Fouten kunnen met name voorkomen als een gebruiker een record in Bedrijf.dbf heeft vergrendeld. Als een fout optreedt, wordt ROLLBACK( ) gebruikt om de oorspronkelijke waarden te herstellen. Als er geen fouten optreden, worden de wijzigingen met COMMIT( ) naar schijf geschreven:

```
CLOSE ALL
SET PROCEDURE TO PROGRAM(1) ADDITIVE
SET EXCLUSIVE OFF

BEGINTRANS()

TransFt=.f.
ON ERROR DO TransFt    && Fouten onderscheppen

USE L:\VeelGebr\Bedrijf
REPLACE ALL VerkTotnu WITH 0
ON ERROR               && ON ERROR uitschakelen

IF TransFt
  ? "Wijzigingen annuleren (Rollback)"
  ROLLBACK()           && Gegevens herstellen
ELSE
  ? "Doorvoeren (Commit)"
  COMMIT()             && Wijzigingen opslaan
ENDIF

PROC TransFt
WAIT "Waarschuwing: Transactie niet geslaagd"
TransFt=.t.
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS. ROLLBACK( ) vervangt het commando ROLLBACK in dBASE IV.

Zie ook

BEGINTRANS( ), COMMIT( ), SET EXCLUSIVE

ROUND( ) — Numerieke gegevens

Rondt een opgegeven getal af tot op een opgegeven aantal decimalen en geeft de uitkomst als resultaat.

Syntaxis

ROUND(<*Nuitdr1*>, <*Nuitdr2*>)

<Nuitdr1>
Het numerieke of zwevende getal dat moet worden afgerond.

<Nuitdr2>
Een positieve <*Nuitdr2*> geeft het aantal decimalen aan waarop <*Nuitdr1*> moet worden afgerond. Als <*Nuitdr2*> negatief is, wordt <*Nuitdr1*> afgerond op tientallen, honderdtallen, duizendtallen, enzovoort.

Beschrijving

Met ROUND( ) kunt u een getal afronden op een opgegeven aantal decimalen of op tientallen, honderdtallen, duizendtallen, enzovoort. Gebruik ROUND( ) samen met SET DECIMALS om een getal af te ronden *en* volgnullen te verwijderen.

Als op de positie <*Nuitdr2*> + 1 een cijfer van 0 tot en met 4 staat, blijft <*Nuitdr1*> (met <*Nuitdr2*> decimalen) onveranderd. Als op de positie <*Nuitdr2*> + 1 een cijfer van 5 tot en met 9 staat, wordt het cijfer op positie <*Nuitdr2*> met 1 verhoogd.

Gebruik 0 voor <*Nuitdr2*> om een getal af te ronden op het dichtstbijzijnde gehele getal. Als u voor <*Nuitdr2*> -1 gebruikt, wordt het getal afgerond op het dichtstbijzijnde tiental, -2 levert het dichtstbijzijnde honderdtal op enzovoort. De uitdrukking ROUND(14932,-2) geeft bijvoorbeeld 14900 als resultaat en ROUND(14932,-3) geeft 15000 als resultaat.

Bij INT( ) is een tabel opgenomen waarin INT( ), FLOOR( ), CEILING( ) en ROUND( ) worden vergeleken

Voorbeeld

In het volgende voorbeeld wordt ROUND( ) gebruikt. De parameter voor het aantal decimalen wordt telkens met 1 verhoogd:

```
SET DECIMALS TO 6
SET TALK OFF
x = 14.746321
? "    x = 14,746321"
FOR y = -5 TO 5 STEP 1
   ? "Als y = " + STR(y,2,0)
   ?? "   is ROUND(x,y) gelijk aan "
   ?? ROUND(x,y)
NEXT
SET TALK ON
```

Zie ook

ABS( ), CEILING( ), FLOOR( ), INT( )

ROW( )

Invoer/uitvoer

Resulteert in het nummer van de huidige rijpositie in het resultatenpaneel van het commandovenster of in het huidige dBASE IV-venster. Dit commando wordt

hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows het kenmerk Top van een klasse om de verticale positie op een formulier te bepalen.

Zie Help voor meer informatie over de syntaxis van ROW( ). Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het werken met formulieren in dBASE voor Windows.

RTOD( ) — Numerieke gegevens

Zet een waarde in radialen om in een waarde in graden.

Syntaxis

RTOD(<*Nuitdr*>)

<Nuitdr>
Een negatief of positief geheel getal dat gelijk is aan de grootte van de hoek in radialen.

Beschrijving

RTOD( ) zet de grootte van een hoek om van radialen in graden. RTOD( ) geeft resulteert in een zwevend getal.

Voor de conversie wordt de volgende berekening gebruikt:

- Het aantal radialen wordt vermenigvuldigd met 180
- Het verkregen produkt wordt gedeeld door pi
- Het quotiënt wordt als resultaat gegeven

Een hoek van pi radialen is gelijk aan 180 graden.

Gebruik RTOD( ) in combinatie met de trigonometrische functies ACOS( ), ASIN( ), ATAN( ) en ATN2( ) om het resultaat in radialen van deze functies om te zetten in graden. Als het standaard aantal decimalen bijvoorbeeld 2 is, resulteert ATAN(1) in de waarde van de hoek in radialen, 0,79, terwijl RTOD(ATAN(1)) resulteer in de waarde van de hoek in graden, 45,00.

Het aantal decimalen stelt u in met SET DECIMALS.

Voorbeeld

In het volgende voorbeeld wordt RTOD( ) gebruikt om radialen om te zetten in graden. Een pi radialen is gelijk aan 180 graden:

```
? RTOD(pi())          && Geeft 180 als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ACOS( ), ASIN( ), ATAN( ), ATN2( ), COS( ), DTOR( ), PI( ), SET DECIMALS, SIN( ), TAN( )

RTRIM( )

Tekenreeksgegevens

Resulteert in een tekenreeks zonder volgspaties.

Syntaxis

RTRIM(<*Tuitdr*> | <*memoveld*>)

<Tuitdr> | <memoveld>

De tekenreeks of het memoveld waaruit de volgspaties moeten worden verwijderd.

Beschrijving

RTRIM( ) is equivalent met TRIM( ). Zie de beschrijving van TRIM( ) voor meer informatie.

Voorbeeld

Zie het voorbeeld met TRIM( ); vervang TRIM( ) overal door RTRIM( ).

Overdraagbaarheid

Het argument <*memoveld*> wordt niet ondersteund in dBASE IV of dBASE III PLUS. In zowel dBASE IV als dBASE III PLUS is het resultaat van RTRIM( ) beperkt tot 254 tekens.

Zie ook

LEFT( ), LTRIM( ), RIGHT( ), TRIM( )

RUN( )

Stations- en bestandsfuncties

Voert een enkele DOS-opdracht, een DOS-toepassing of een Windows-toepassing uit vanuit dBASE en resulteert in een afsluitcode van de opdracht of toepassing.

Syntaxis

RUN([<*Luitdr1*>,] <*DOS-opdracht Tuitdr*> [,<*Luitdr2*>])

<Luitdr1>

Bepaalt of RUN( ) een Windows-programma (waarde = .T.) of een DOS-programma (waarde = .F.) start. Als u *<Luitdr1>* achterwege laat, wordt de waarde .F. gebruikt.

<DOS-opdracht Tuitdr>

Een opdracht die wordt herkend door het besturingssysteem (DOS). Anders dan bij het commando RUN, moet deze opdracht als *<Tuitdr>* worden opgegeven.

<Luitdr2>

dBASE voor Windows negeert deze parameter. De parameter is alleen aanwezig vanwege de compatibiliteit met dBASE IV.

Beschrijving

Met RUN( ) kunt u een enkele DOS-opdracht of een externe DOS- of Windows-toepassing uitvoeren zonder dBASE af te sluiten. Gebruik het commando DOS als u interactief meerdere DOS-opdrachten wilt uitvoeren zonder dBASE af te sluiten.

Als de uitvoering van de opdracht of het programmabestand is voltooid, wordt de besturing teruggegeven aan het commandovenster of aan het programma, de procedure of de door de gebruiker gedefinieerde functie waarin RUN( ) is gebruikt. Tevens wordt de afsluitcode als resultaat gegeven.

Als toegang tot DOS moet worden verkregen (*<Luitdr1>* = .F. of *<Luitdr1>* is achterwege gelaten), wordt COMMAND.COM geladen vanuit DBASEWIN.PIF in de directory waar DBASEWIN.EXE staat. (Met HOME( ) kunt u bepalen welke directory dat is.) U kunt DBASEWIN.PIF wijzigen met de PIF Editor van Windows. Als u bijvoorbeeld niet wilt dat het DOS-venster automatisch wordt gesloten nadat de opdracht is voltooid, kunt u het aankruisvakje **Venster dicht na afsluiten** in de PIF Editor uitschakelen. Zie het handboek bij Windows voor meer informatie over .PIF-bestanden.

Als een DOS-opdracht of -toepassing is uitgevoerd, geeft RUN( ) 0 als resultaat als de opdracht of toepassing correct is uitgevoerd. Als een Windows-toepassing wordt uitgevoerd met RUN( ), wordt de instance handle van de toepassing (als de uitvoering correct is verlopen) of een foutnummer als resultaat gegeven (als tijdens de uitvoering een fout is opgetreden). Een instance handle is een 16-bit geheel getal dat een actieve Windows-toepassing aanduidt en een foutnummer heeft betrekking op een foutconditie in Windows.

Voorbeeld

In het volgende voorbeeld wordt geprobeerd de DOS-opdracht Scandisk uit te voeren. Als de opdracht juist wordt uitgevoerd, is Resultaat gelijk aan 0 en anders is Resultaat ongelijk aan 0:

```
Resultaat=RUN(.f.,"Scandisk")
```

In het volgende voorbeeld wordt geprobeerd Windows-toepassingen uit te voeren:

```
Resultaat=RUN(.t.,"winfile.exe")
* geeft bijvoorbeeld 21006 als resultaat
Resultaat=RUN(.t.,"foutprog.exe")
* geeft bijvoorbeeld 2 als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. In dBASE IV bepaalt *<Luitdr1>* of het opdrachtverwerkingsprogramma van het besturingssysteem wordt geladen en *<Luitdr2>* of alle beschikbare extended geheugen wordt vrijgegeven voordat *<DOS-opdracht>* wordt gestart.

Zie ook

DOS, HOME( ), RUN

RUN

Stations- en bestandsfuncties

Voert een enkele DOS-opdracht of -toepassing uit vanuit dBASE.

Syntaxis

RUN *<DOS-opdracht>*

<DOS-opdracht>

Een commando dat door het besturingssysteem (DOS) wordt herkend.

Beschrijving

Gebruik RUN om een enkele DOS-opdracht of een enkel DOS-programmabestand uit te voeren zonder dBASE af te sluiten. Geef de opdrachten en bestandsnamen net zo op als onder DOS. Plaats deze niet tussen aanhalingstekens. ! is onderling uitwisselbaar met RUN.

Als de uitvoering van de opdracht of het programmabestand is voltooid, wordt de besturing teruggegeven aan het commandovenster of aan het programma, de procedure of de door de gebruiker gedefinieerde functie waarin RUN is gebruikt.

Als u opdrachten geeft die worden uitgevoerd in een DOS-venster, kunt u de PIF Editor van Windows gebruiken om de instellingen in DBASEWIN.PIF te wijzigen. Als u bijvoorbeeld niet wilt dat het DOS-venster automatisch wordt gesloten nadat de opdracht is voltooid, kunt u het aankruisvakje **Venster dicht na afsluiten** in de PIF Editor uitschakelen. Zie het handboek bij Windows voor meer informatie over .PIF-bestanden.

Gebruik het commando DOS als u interactief meerdere DOS-opdrachten wilt uitvoeren zonder dBASE af te sluiten. Met RUN( ) kunt u een Windows-toepassing uitvoeren zonder dBASE af te sluiten.

Als u de huidige directory die door dBASE wordt gebruikt, wilt wijzigen, gebruikt u CD in plaats van RUN. Met RUN CD wijzigt u alleen de directory die in het DOS-venster wordt gebruikt en niet de actieve directory in dBASE.

Voorbeeld

Bij RUN moet een argument worden opgegeven:

```
RUN dir
* DOS-opdracht DIR uitvoeren
RUN  && geeft een fout als resultaat
```

Zie ook

CD, DOS, HOME( ), RUN( ), SET DIRECTORY, SET PATH

SAVE — Geheugenvariabelen

Slaat alle of opgegeven geheugenvariabelen op in een geheugenbestand.

Syntaxis

SAVE TO <*bestandsnaam*> | ? | <*bestandsnaamfilter*>
[ALL]
[LIKE <*variabelefilter1*>]
[EXCEPT <*variabelefilter2*>]

TO <*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Geeft het doelbestand <*bestandsnaam*> op waarin u de geheugenvariabelen wilt opslaan. Standaard wordt de extensie .MEM toegewezen aan <*bestandsnaam*> en wordt het bestand opgeslagen in de huidige directory. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u de naam opgeeft van het doelbestand en van de directory waarin dit moet worden opgeslagen.

ALL

Slaat alle geheugenvariabelen op in het geheugenbestand. Ook als u SAVE TO <*bestandsnaam*> zonder opties opgeeft, worden alle geheugenvariabelen in het geheugenbestand opgeslagen.

LIKE <*variabelefilter1*>

Slaat in het doelbestand de geheugenvariabelen op waarvan de namen overeenkomen met het geheugenvariabelenschema dat u opgeeft voor <*variabelefilter1*>. Stel <*variabelefilter1*> samen uit tekens van de variabelenamen en de jokers * en ?.

EXCEPT <*variabelefilter2*>

Slaat in het doelbestand alle geheugenvariabelen op behalve die waarvan de namen overeenkomen met het geheugenvariabelenschema dat u opgeeft voor <*variabelefilter2*>. Stel <*variabelefilter2*> samen uit tekens van de variabelenamen en de jokers * en ?.

Beschrijving

Gebruik SAVE en RESTORE om belangrijke geheugenvariabelen op te slaan en opnieuw op te halen. Lokale variabelen worden gewist als het programma waarin deze zijn

gedefinieerd, wordt beëindigd, terwijl publieke variabelen worden gewist als u dBASE afsluit. U kunt deze waarden behouden door deze met SAVE op te slaan in een geheugenbestand. Met RESTORE kunt u de variabelen op een later tijdstip opnieuw ophalen.

Als SET SAFETY is ingeschakeld (ON) en er bestaat een bestand met dezelfde naam als het doelbestand, verschijnt een dialoogvenster waarin wordt gevraagd of het bestand moet worden overschreven. Als SET SAFETY is uitgeschakeld (OFF), wordt een bestand met dezelfde naam zonder waarschuwing overschreven.

Opmerking Met SAVE worden geen objectverwijzingen, functie-aanwijzers en systeemgeheugenvariabelen opgeslagen.

Voorbeeld

Zie RESTORE voor een voorbeeld met SAVE.

Overdraagbaarheid

De opties ? en *<bestandsnaamfilter>* worden niet ondersteund in dBASE IV of dBASE III PLUS. In dBASE IV en dBASE III PLUS wordt het gebruik van zowel LIKE als EXCEPT in één instructie met SAVE niet ondersteund.

In dBASE IV en dBASE III PLUS moet u ALL opgeven als u LIKE of EXCEPT gebruikt. In dBASE voor Windows is bijvoorbeeld de instructie SAVE TO gehbest LIKE gvar* geldig zonder de optie ALL.

Zie ook

RELEASE, RESTORE, SET SAFETY, STORE

SAVE SCREEN

Invoer/uitvoer

Slaat de inhoud van het resultatenpaneel van het commandovenster op. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows worden formulierdefinities opgeslagen in programma-, procedure- of bibliotheekbestanden.

Zie Help voor meer informatie over de syntaxis van SAVE SCREEN. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

SAVE WINDOW

dBASE IV-vensters

Slaat vensterdefinities op in een vensterbestand. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows worden formulierdefinities opgeslagen in programma-, procedure- of bibliotheekbestanden.

Zie Help voor meer informatie over de syntaxis van SAVE WINDOW. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

SCAN — Programma's

Doorloopt alle records in de huidige tabel en voert opgegeven instructies uit voor elk record dat aan bepaalde voorwaarden voldoet.

Syntaxis

```
SCAN
      [<bereik>] [FOR <voorwaarde1>] [WHILE <voorwaarde2>]
       [<statements>]
       [LOOP]
       [EXIT]
ENDSCAN
```

<bereik>

Het aantal records waarin moet worden gezocht. RECORD *<n>* geeft een record aan door middel van het recordnummer. NEXT *<n>* geeft *n* records aan, te beginnen bij het huidige record. ALL geeft alle records aan. REST geeft alle records vanaf het huidige record tot het eind van de tabel aan.

FOR *<voorwaarde1>*
WHILE *<voorwaarde2>*

Bepaalt welke records worden beïnvloed door SCAN. FOR beperkt SCAN tot records die voldoen aan *<voorwaarde1>*. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang *<voorwaarde2>* waar is.

<instructies>

Programmaregels met een willekeurige combinatie van commando's, functies, door de gebruiker gedefinieerde functies en LOOP- en EXIT-opties. Omdat SCAN de tabel per record doorloopt, bevatten deze instructies waarschijnlijk tabelcommando's en -functies als ISBLANK( ) en REPLACE.

LOOP

Plaatst de recordaanwijzer één record verder en geeft de besturing terug aan het begin van de lus zonder de instructies tussen LOOP en ENDSCAN uit te voeren.

EXIT

Geeft de besturing door aan de instructie die volgt op ENDSCAN zonder de instructies tussen EXIT en ENDSCAN uit te voeren. Verdere records worden niet verwerkt.

ENDSCAN

Een verplicht commando dat het eind aangeeft van de SCAN-lus. Als ENDSCAN wordt aangetroffen, wordt de recordaanwijzer één record verder geplaatst en keert de besturing terug naar het begin van de lus.

Beschrijving

Met SCAN kunt u de huidige tabel per record verwerken, te beginnen bij het eerste record in de tabel of hoofdindex of bij het eerste record dat voldoet aan een FOR-voorwaarde. Voor elk record dat binnen *<bereik>* valt of voldoet aan een FOR-voorwaarde of een WHILE-voorwaarde of elke willekeurige combinatie van deze drie, worden alle instructies tussen SCAN en LOOP, EXIT of ENDSCAN uitgevoerd. De SCAN-lus gaat door tot het eind van de tabel (of het laatste record in de index) tenzij het *<bereik>*, de FOR- of WHILE-voorwaarde of de optie EXIT de lus eerder afbreekt. De recordaanwijzer wordt niet meer verplaatst als de verwerking wordt beëindigd.

Aan het eind van elke doorgang door de lus wordt de recordaanwijzer automatisch één record verder geplaatst in de tabel voordat wordt teruggekeerd naar het begin van de lus. Denk er dus aan geen instructie met SKIP in de lus op te nemen.

U kunt lussen en andere structuren nesten in een SCAN-lus, inclusief andere SCAN-lussen. Elke ENDSCAN in een groep geneste SCAN-lussen verplaatst de recordaanwijzer één record verder. De code in de geneste lussen moet dus rekening houden met deze verplaatsing van de recordaanwijzer. Anders kunnen onbedoeld records worden overgeslagen.

SCAN functioneert als een DO WHILE .NOT. EOF( )...SKIP...ENDDO-constructie. Met SCAN kunt u echter voorwaarden opgeven met FOR, WHILE en *<bereik>*. Met SCAN hebt u ook minder code nodig dan met DO WHILE.

Als u SCAN gebruikt met een geïndexeerde tabel, mag u niet de waarde veranderen van het veld dat deel uitmaakt van de hoofdindexsleutel. Als u de waarde in een dergelijk veld wijzigt, wordt de positie van dat record in het indexbestand bijgewerkt en dat kan onbedoelde gevolgen hebben. Als u bijvoorbeeld een sleutelveld zo wijzigt dat het record naar het eind van de index wordt verplaatst, worden de SCAN...ENDSCAN-instructies misschien een tweede keer op dat record uitgevoerd.

Als u binnen een SCAN-lus van werkgebied verandert, moet u terugkeren naar het werkgebied met de oorspronkelijke tabel voordat de besturing wordt teruggegeven aan het begin van de lus.

Voorbeeld

In het volgende voorbeeld wordt SCAN gebruikt om een tabel te doorlopen. Als een record wordt aangetroffen met een positieve waarde in het veld OpenBalans, wordt een factuurprocedure aangeroepen:

```
CLEAR
USE Afnemers
SCAN FOR OpenBalans > 0
  DO Factuur WITH Bedrijf, OpenBalans, BalnsDatum
ENDSCAN
```

```
PROCEDURE Factuur
PARAMETERS Bedrijf, OpenBalans, BalnsDatum
? Bedrijf, OpenBalans, BalnsDatum
RETURN
```

De volgende code komt overeen met de SCAN-lus in het voorgaande voorbeeld, maar hier worden DO WHILE en IF/ENDIF gebruikt:

```
SET TALK OFF
USE Afnemers
DO WHILE .NOT. EOF()
  If OpenBalans > 0
    DO Factuur WITH Bedrijf, OpenBalans, BalnsDatum
  ENDIF
  SKIP
ENDDO

PROCEDURE Factuur
PARAMETERS Bedrijf, OpenBalans, BalnsDatum
? Bedrijf, OpenBalans, BalnsDatum
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. In dBASE IV worden geen geneste SCAN-lussen ondersteund.

Zie ook

DO WHILE, DO...UNTIL, FOR...NEXT, INDEX, LOCATE, SEEK, SKIP

SECONDS( ) — Datum- en tijdgegevens

Resulteert in het aantal seconden dat is verstreken sinds middernacht.

Syntaxis

SECONDS( )

Beschrijving

SECONDS( ) resulteert in het aantal seconden tot op honderdsten nauwkeurig dat is verstreken sinds middernacht. Het resultaat heeft de opmaak SS.hh, waarbij SS de seconden en hh de honderdsten van seconden voorstellen.

Met SECONDS( ) kunt u berekenen hoeveel tijd de uitvoering van bepaalde delen van een programma kost. SECONDS( ) is daarvoor beter geschikt dan TIME( ), omdat SECONDS( ) resulteert in een getal, niet in een tekenreeks.

U kunt SECONDS( ) ook gebruiken in plaats van ELAPSED( ) om de verstreken tijd nauwkeuriger te bepalen.

Voorbeeld

In de eerste regel van het volgende voorbeeld wordt de tekenreeks waarin TIME( ) resulteert, gesplitst in de delen voor de uren, de minuten en de seconden. De waarden van die delen worden bepaald en gebruikt om het aantal seconden sinds middernacht te bepalen. De tweede regel gebruikt SECONDS( ) om duidelijk te maken dat die functie in hetzelfde getal resulteert als de berekening in de eerste instructie:

```
? TIME(),(VAL(SUBSTR(TIME(),1,2))*3600) + ;
   (VAL(SUBSTR(TIME(),4,2))*60) + ;
   VAL(SUBSTR(TIME(),7,2))
? SECONDS()
```

Zie ook

ELAPSED( ), SET TIME, TIME( )

SEEK

Tabelindeling

Zoekt het eerste record in een geïndexeerde tabel waarvan het sleutelveld overeenkomt met de opgegeven uitdrukking.

Syntaxis

SEEK *<Tuitdrukkingenlijst> | <Nuitdrukkingenlijst>*

<Tuitdrukkingenlijst> | <Nuitdrukkingenlijst>

De tekenreeks of het getal die of dat moet worden gezocht in de sleutelvelden van de hoofdindex. Voor dBASE-tabellen kunt u een dBASE-uitdrukking opgeven die overeen moet komen met de indexsleuteluitdrukking. Voor Paradox- en SQL-tabellen kunt u waarden opgeven (gescheiden door komma's) die overeenkomen met enkelvoudige of samengestelde indexsleutelvelden.

Beschrijving

Met dBASE voor Windows kunt u in een tabel sequentieel zoeken naar bepaalde informatie of geïndexeerd zoeken in de hoofdindex van de tabel. Een sequentiële zoekbewerking komt min of meer overeen met zoeken naar informatie in een boek door eerst de eerste pagina te lezen, dan de tweede, enzovoort, tot de informatie is gevonden of tot alle pagina's zijn gelezen. Deze methode wordt gebruikt door LOCATE: elk record wordt onderzocht tot de informatie is gevonden of tot alle records zijn onderzocht.

Een geïndexeerde zoekbewerking is vergelijkbaar met het opzoeken van een onderwerp in een boek waarbij u onmiddellijk op de juiste pagina terechtkomt. Als een tabel eenmaal is geïndexeerd, kan met SEEK (of FIND) de index worden gebruikt om snel het juiste record te zoeken.

SEEK begint met zoeken aan het begin van de index en stopt als een overeenkomst is aangetroffen of als het eind van de index is bereikt. Als een overeenkomst is

aangetroffen (FOUND( ) geeft .T. als resultaat), wordt de recordaanwijzer voor de betreffende tabel bij het record met de overeenkomst geplaatst.

Met SKIP kunt u toegang krijgen tot andere records waarvan de sleutelvelden overeenkomen met de indexsleutelvelden of de uitdrukking. SKIP verplaatst de recordaanwijzer één record verder. Vanwege de geïndexeerde volgorde komen eventuele volgende overeenkomsten onmiddellijk na de eerste. Denk er echter aan dat SKIP na SEEK (in tegenstelling tot CONTINUE na LOCATE) niet zoekt naar een overeenkomst. SKIP verplaatst alleen maar de recordaanwijzer naar het volgende record, of dat record nu een overeenkomst bevat of niet.

De instelling voor SET NEAR bepaalt waar de recordaanwijzer wordt geplaatst als SEEK geen overeenkomst aantreft: aan het eind van het bestand of bij het record in de geïndexeerde tabel onmiddellijk na de positie waar de gezochte waarde zou moeten staan. Als SET NEAR is uitgeschakeld (OFF, de standaardinstelling) en SEEK vindt geen overeenkomst, resulteert EOF( ) in .T. en FOUND( ) in .F.. Als SET NEAR is ingeschakeld (ON) en SEEK vindt geen overeenkomst, resulteren EOF( ) (tenzij de gezochte waarde in het laatste record van de index staat) en FOUND( ) beide in .F.

Met SEEK kunt u zoeken naar elk geldig sleutelveld of elke geldige uitdrukking van het type teken, numeriek, zwevend of datum. Een tekenuitdrukking moet tussen (enkele of dubbele) aanhalingstekens of teksthaakjes staan. Datumuitdrukkingen moeten tussen accolades ({ }) staan of worden geconverteerd met de functie CTOD( ).

De uitdrukking die u zoekt, moet overeenkomen met de sleuteluitdrukking of -velden van de hoofdindex. Als de hoofdindexsleutel bijvoorbeeld SUBSTR(Klantnr, 2, 5) is, kunt u SEEK SUBSTR(Klantnr, 2, 5) gebruiken.

Als u een sleuteluitdrukking van het type teken zoekt, bepalen de regels die zijn ingesteld met SET EXACT of een overeenkomst wordt aangetroffen. Als SET EXACT is uitgeschakeld (OFF, de standaardinstelling), hoeft u alleen de eerste tekens in het sleutelveld op te geven om een overeenkomst te vinden. Als SET EXACT is ingeschakeld (ON), moet de opgegeven uitdrukking volledig gelijk zijn aan het sleutelveld.

De instelling voor SET EXACT bepaalt in combinatie met de ingestelde taalaansturing hoe speciale tekens (tekens met accenten) worden verwerkt bij zoekbewerkingen. Zie Appendix C in *Programmeren* voor meer informatie.

De functie SEEK( ) werkt als SEEK gevolgd door FOUND( ), behalve dat SEEK in het huidige werkgebied zoekt, terwijl u met SEEK( ) in het huidige of een opgegeven werkgebied kunt zoeken. Afhankelijk van de uitkomst van de zoekbewerking geeft SEEK( ) .T. of .F. als resultaat.

FIND, SEEK en LOCATE bieden elk hun eigen voordelen. FIND en SEEK zijn het snelst, maar beide vereisen een geïndexeerd bestand en kunnen alleen zoeken naar waarden van de sleuteluitdrukking. SEEK biedt meer flexibiliteit dan FIND omdat SEEK zowel dBASE-uitdrukkingen als numerieke en tekeninvoer accepteert. Als u met FIND naar de waarde van een geheugenvariabele wilt zoeken, moet u de operator voor macrosubstitutie & gebruiken. Tekeninvoer voor SEEK moet tussen scheidingstekens staan.

Als de informatie die u zoekt, in een niet-geïndexeerd bestand staat of niet in een sleutelveld van een geïndexeerd bestand, kunt u LOCATE gebruiken. LOCATE accepteert een uitdrukking van elk willekeurig gegevenstype als invoer en kan in elk veld van de tabel zoeken naar die waarde. Bij grote tabellen kan een sequentiële zoekbewerking met LOCATE echter veel tijd kosten. U kunt dan beter eerst een nieuwe index maken met INDEX en vervolgens SEEK, SEEK( ) of FIND gebruiken.

Voorbeeld

In het volgende voorbeeld wordt SEEK gebruikt om het eerste bedrijf in het noordwesten te zoeken:

```
USE Bedrijf EXCLUSIVE
INDEX ON Regio TAG Regio
Sleutelwaarde="NW"
SEEK Sleutelwaarde     && inhoud van een variabele zoeken
IF FOUND()
   ? "Alle bedrijven in het noordwesten"
   LIST FIELDS Bedrijf, Regio;
     WHILE Regio=Sleutelwaarde
ELSE
   ? "Geen bedrijven in het noordwesten"
ENDIF
```

Zie ook

DTOS( ), EOF( ), FIND, FOUND( ), INDEX, LOCATE, SEEK( ), SET EXACT, SET NEAR

SEEK( )

Tabelindeling

Zoekt het eerste record in een geïndexeerde tabel waarvan het sleutelveld overeenkomt met de opgegeven uitdrukking.

Syntaxis

SEEK(<*Tuitdr*> | <*Nuitdr*> [,<*alias*>])

<*Tuitdr*> | <*Nuitdr*>

De tekenreeks of het getal waarnaar moet worden gezocht in het sleutelveld van de geïndexeerde tabel in het huidige werkgebied of het door <*alias*> aangeduide werkgebied. Tekeninvoer moet tussen enkele of dubbele aanhalingstekens of teksthaakjes staan. Datumuitdrukkingen moeten tussen accolades ({ }) staan of worden geconverteerd met de functie CTOD( ).

<*alias*>

Een werkgebiednummer (van 1 tot 255) of -letter (van A tot J) of aliasnaam. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens.

Beschrijving

SEEK( ) evalueert de uitdrukking <*Tuitdr*> of <*Nuitdr*> en zoekt de waarde van de uitdrukking in de hoofdindex van de tabel die is geopend in het huidige of een opgegeven werkgebied. SEEK( ) geeft .T. als resultaat als een overeenkomst met de sleuteluitdrukking in de hoofdindex wordt aangetroffen en anders .F.

SEEK( ) kan alleen zoeken naar een uitdrukking in het sleutelveld van de hoofdindex. Als u een tekenuitdrukking, <*Tuitdr*>, opgeeft en SET EXACT is ingeschakeld (ON), moet <*Tuitdr*> overeenkomen met elk teken in het sleutelveld. Als SET EXACT is uitgeschakeld (OFF), wordt <*Tuitdr*> alleen vergeleken met de eerste tekens in het sleutelveld.

De instelling voor SET EXACT bepaalt in combinatie met de ingestelde taalaansturing hoe speciale tekens (tekens met accenten) worden verwerkt bij zoekbewerkingen. Zie Appendix C in *Programmeren* voor meer informatie.

SEEK( ) begint met zoeken aan het begin van een geïndexeerde tabel en verplaatst de recordaanwijzer naar het eerste record waarvan het sleutelveld overeenkomt met <*Tuitdr*> of <*Nuitdr*>. Als <*Tuitdr*> of <*Nuitdr*> niet wordt aangetroffen, verplaatst SEEK( ) de recordaanwijzer naar het eind van het bestand. SEEK( ) kan elke geldige sleuteluitdrukking van het type teken, numeriek, zwevend of datum zoeken. Als SET NEAR is ingeschakeld (ON), wordt de recordaanwijzer geplaatst bij het record onmiddellijk na de positie waar de gezochte waarde zou moeten staan.

Met SKIP kunt u toegang krijgen tot andere records waarvan de sleutelveldwaarden voldoen aan de zoekuitdrukking. SKIP verplaatst de recordaanwijzer één record verder. Vanwege de geïndexeerde volgorde komen eventuele volgende overeenkomsten onmiddellijk na de eerste. Denk er echter aan dat SKIP na SEEK (in tegenstelling tot CONTINUE na LOCATE) niet zoekt naar een overeenkomst. SKIP verplaatst alleen maar de recordaanwijzer naar het volgende record, of dat record nu een overeenkomst bevat of niet.

SEEK( ) is vergelijkbaar met SEEK gevolgd door FOUND( ). De resultaatwaarde van FOUND( ) is gelijk aan de resultaatwaarde van SEEK( ).

Voorbeeld

In het volgende voorbeeld wordt SEEK( ) gebruikt om het eerste overeenkomende record in een geïndexeerde tabel te zoeken:

```
CLOSE DATABASE
USE Bedrijf EXCLUSIVE        && werkgebied 1
INDEX ON Regio TAG Regio
SELECT 2
Gevonden=SEEK("NW","Bedrijf")
```

Werkgebied 2 is actief en SEEK( ) wordt uitgevoerd in werkgebied 1.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

EOF( ), FIND, FOUND( ), INDEX, LOOKUP( ), SEEK, SET EXACT, SET NEAR, SET RELATION, SET SKIP

SELECT

Tabellen

Stelt het huidige werkgebied in.

Syntaxis

SELECT *<alias>*

<alias>

Geeft een werkgebied aan. U kunt een werkgebiednummer (1 tot en met 255) of -letter (A tot en met J) of een aliasnaam opgeven. De werkgebiedletter of aliasnaam kan tussen aanhalingstekens staan.

Beschrijving

Met SELECT stelt u het werkgebied in waarin een tabel wordt geopend, of een werkgebied waarin al een tabel is geopend. Gebruik SELECT om in elk werkgebied een tabel en alle bijbehorende bestanden te openen, zoals index-, query- en indelingsbestanden.

Elk werkgebied ondersteunt een eigen waarde van FOUND( ) en een onafhankelijke recordaanwijzer. Als de recordaanwijzer in het actieve werkgebied wordt verplaatst, heeft dat geen gevolgen voor recordaanwijzers in andere werkgebieden, tenzij u met het commando SET RELATION een relatie hebt ingesteld tussen werkgebieden.

Gebruik de macro-operator (&) om een werkgebied in te stellen door middel van een variabele, of plaats de variabele voor het werkgebied tussen haakjes. Als N=1 kunt u bijvoorbeeld SELECT (N) gebruiken om werkgebied 1 te selecteren.

Voorbeeld

In het volgende voorbeeld wordt het commando SELECT gebruikt om van de huidige tabel naar een eerder geopende tabel te gaan:

```
PUBLIC tabel1, tabel2
USE Contact EXCLUSIVE IN SELECT()
INDEX ON CompCode TAG CompCode
tabel1 = ALIAS()

USE Bedrijf IN SELECT()
tabel2 = ALIAS()

SELECT Bedrijf
COPY STRUCTURE TO CntctLst;
   FIELDS Contact->CompCode, ;
   Bedrijf->Bedrijf, Contact->Contact, ;
```

```
    Bedrijf->Straat1, Bedrijf->Straat2, ;
    Bedrijf->Plaats, Bedrijf->Regio,;
    Bedrijf->Postcode
USE CntctLst IN SELECT()
tabel3 = ALIAS()
APPEND
SELECT &tabel1
? ALIAS(),DBF(),WORKAREA()
SELECT &tabel3
? ALIAS(),DBF(),WORKAREA()
SELECT &tabel2
? ALIAS(),DBF(),WORKAREA()
SELECT 3
? ALIAS(),DBF(),WORKAREA()
SELECT 1
? ALIAS(),DBF(),WORKAREA()
SELECT 2
? ALIAS(),DBF(),WORKAREA()
CLOSE ALL
```

Zie ook

SELECT( ), SET RELATION, USE, WORKAREA( )

SELECT( ) Tabellen

Resulteert in het nummer van een beschikbaar werkgebied of het werkgebiednummer voor een opgegeven alias.

Syntaxis

SELECT([<*alias*>])

<alias>

Een werkgebiednummer (van 1 tot 255) of -letter (van A tot J) of een aliasnaam. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens.

Beschrijving

Als u geen alias opgeeft, resulteert SELECT( ) in het nummer van het volgende beschikbare werkgebied. Dat is een nummer van 1 tot en met 225. Als u een alias opgeeft, bepaalt SELECT( ) of de opgegeven aliasnaam al in gebruik is.

Met SELECT( ) kunt u bepalen welk werkgebied beschikbaar is om een tabel te openen zonder eerst een tabel te hoeven sluiten. SELECT( ) resulteert in 0 als er geen werkgebieden meer beschikbaar zijn of als een opgegeven <*alias*> nog niet in gebruik is.

Voorbeeld

In het volgende voorbeeld wordt SELECT( ) gebruikt om het volgende beschikbare werkgebied en het werkgebied voor een alias op te halen:

```
CLOSE DATABASES
USE Klanten IN SELECT()
USE Bedrijf IN SELECT()
? SELECT()    && Resulteert in 3. Het volgende;
                 beschikbare werkgebied is 3
? ALIAS(1)    && Resulteert in KLANTEN
? ALIAS(2)    && Resulteert in BEDRIJF
? DBF(1)      && Resulteert in C:KLANTEN.DBF
? SELECT()    && Resulteert in 3. Werkgebied 3 ;
                 blijft het huidige werkgebied
SELECT 2      && Werkgebied 2 actief
? ALIAS()     && Resulteert in BEDRIJF
? WORKAREA()  && Resulteert in 2
? SELECT()    && Resulteert in 3
? SELECT("Klanten");
              && Resulteert in 1: werkgebied 1
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ALIAS( ), DBF( ), SELECT, WORKAREA( )

SET

Omgeving

Toont het dialoogvenster **Kenmerken bureaublad** waarin de instellingen van een groot aantal SET-commando's kunnen worden bekeken en gewijzigd. De gewijzigde waarden worden opgeslagen in het bestand DBASEWIN.INI.

Syntaxis

SET

Beschrijving

Gebruik SET om instellingen te bekijken en interactief te wijzigen. U hoeft dan geen afzonderlijke SET-commando's te typen in het commandovenster (zoals SET TALK ON).

Opmerking Alle instellingen die u wijzigt met SET, worden automatisch opgeslagen in DBASEWIN.INI. Dat betekent dat de instellingen van kracht worden als u dBASE een volgende keer start. Als u de instelling van een SET-commando tijdelijk wilt wijzigen, moet u het betreffende SET-commando opgeven in het commandovenster of in een

programma. Zie Appendix C in het *Handboek* voor meer informatie over DBASEWIN.INI.

U kunt het dialoogvenster ook openen door **Kenmerken | Bureaublad** te kiezen.

Overdraagbaarheid

In dBASE IV en dBASE III PLUS zijn wijzigingen die u aanbrengt met SET van kracht tot u dBASE afsluit. Wijzigingen worden niet opgeslagen in het bestand CONFIG.DB.

Zie ook

DISPLAY STATUS, SET( ), SETTO( ), afzonderlijke SET-commando's

SET ALTERNATE

Invoer/uitvoer

Bestuurt het vastleggen van invoer en uitvoer in een alternatief tekstbestand.

Syntaxis

```
SET ALTERNATE on | OFF
SET ALTERNATE TO
   [<bestandsnaam> | ? | <bestandsnaamfilter>
      [ADDITIVE]]
```

<bestandsnaam> | ? | <bestandsnaamfilter>

Het alternatieve tekstbestand, of doelbestand, dat moet worden gemaakt of geopend. De opties ? en *<bestandsnaamfilter>* tonen een dialoogvenster waarin u een nieuw bestand kunt opgeven of een bestaand bestand kunt selecteren. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, wordt de extensie .TXT gebruikt.

ADDITIVE

Voegt uitvoer naar het resultatenpaneel van het commandovenster toe aan het opgegeven bestaande alternatieve bestand. Als het bestand niet bestaat, verschijnt een foutmelding.

Standaardinstelling

De standaardinstelling voor SET ALTERNATE is OFF. Als u de standaardinstelling wilt wijzigen, wijzigt u de parameter voor ALTERNATE in de sectie [OnOffCommandSettings] in DBASEWIN.INI. Als u een standaardbestandsnaam voor SET ALTERNATE wilt opgeven, geeft u een parameter voor ALTERNATE op in de sectie [CommandSettings] in DBASEWIN.INI.

Beschrijving

Met SET ALTERNATE TO kunt u een rapport vastleggen van de dBASE-uitvoer en -commando's. U kunt de inhoud van dit bestand bekijken of wijzigen met de Teksteditor of een andere editor. U kunt reeksen commando's vastleggen, wijzigen en opnemen in nieuwe programma's. (Als u de resultaten van @...SAY-commando's naar een tekstbestand wilt sturen, gebruikt u SET DEVICE TO FILE.)

SET ALTERNATE TO *<bestandsnaam>* opent het alternatieve bestand alleen, terwijl SET ALTERNATE ON | OFF de opslag van in- en uitvoer in het bestand bestuurt. Er kan niet meer dan één alternatief bestand tegelijk zijn geopend. Als u SET ALTERNATE TO *<bestandsnaam>* gebruikt om een nieuw bestand te openen, wordt een eventueel eerder geopend alternatief bestand gesloten.

Als SET ALTERNATE is ingeschakeld (ON), wordt uitvoer naar het resultatenpaneel van het commandovenster opgeslagen in het tekstbestand dat u hebt eerder geopend met SET ALTERNATE TO *<bestandsnaam>*. SET ALTERNATE ON heeft alleen effect als een alternatief tekstbestand is geopend. SET ALTERNATE is niet van invloed op de uitvoer van een programma. Het commando bepaalt alleen of uitvoer in een alternatief bestand wordt opgeslagen. (Toetsenbordinvoer in het commandovenster wordt niet opgeslagen in het alternatieve bestand.)

Voorkom dat het tekstbestand begint met een lege regel door twee vraagtekens (??) op te geven voordat u het eerste woord naar het alternatieve bestand stuurt.

SET ALTERNATE OFF heeft niet tot gevolg dat het alternatieve bestand wordt gesloten. Voordat u toegang kunt krijgen tot de inhoud van een alternatief bestand, moet u het eerst sluiten met CLOSE ALTERNATE of SET ALTERNATE TO (zonder bestandsnaam). Hiermee worden alle vastgelegde gegevens opgeslagen in het alternatieve bestand op schijf en wordt SET ALTERNATE automatisch uitgeschakeld (OFF).

Als SET SAFETY is ingeschakeld (ON), u niet de optie ADDITIVE gebruikt en er een bestand bestaat met dezelfde naam als het doelbestand, verschijnt een dialoogvenster waarin wordt gevraagd of u het bestand wilt overschrijven. Als SET SAFETY is uitgeschakeld (OFF) en u gebruikt niet de optie ADDITIVE, wordt elk bestaand bestand met dezelfde naam als het doelbestand zonder waarschuwing overschreven.

Voorbeeld

In dit voorbeeld wordt SET ALTERNATE gebruikt om tekst naar het scherm en naar een ASCII-bestand te schrijven:

```
SET ALTERNATE TO Roos
* Roos.txt openen voor tekstuitvoer
? "Alternatief bestand wordt geopend" && alleen naar scherm
SET ALTERNATE ON
* commando's ? en ?? ook naar Roos.txt
? "Een roos "
SET ALTERNATE OFF     && Niet meer naar Roos.txt.
?? "in de tuin "
SET ALTERNATE ON&& Toevoegen aan Roos.txt.
?? "is een roos "
?? "is een roos "
```

```
? "waar u trots op kunt zijn"
CLOSE ALTERNATE        && Roos.txt sluiten
* Roos.txt bevat:
* Een roos is een roos is een roos
* waar u trots op kunt zijn
```

Overdraagbaarheid

De optie ADDITIVE wordt niet ondersteund in dBASE III PLUS. De opties ? en *<bestandsnaamfilter>* worden niet ondersteund in dBASE III PLUS of dBASE IV.

Zie ook

CLOSE..., SET DEVICE

SET AUTOSAVE

Velden en records

Bepaalt of gegevens automatisch onmiddellijk naar schijf worden geschreven als een record wordt gewijzigd of toegevoegd.

Syntaxis

SET AUTOSAVE on | OFF

Standaardinstelling

De standaardinstelling voor SET AUTOSAVE is OFF. U kunt de standaardinstelling voor AUTOSAVE wijzigen in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor AUTOSAVE rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Gebruik SET AUTOSAVE ON om de kans op gegevensverlies te verkleinen. Als SET AUTOSAVE is ingeschakeld (ON) en u wijzigt een record of voegt een record toe, worden tabellen en indexbestanden op schijf bijgewerkt als u de recordaanwijzer verplaatst. Als SET AUTOSAVE is uitgeschakeld (OFF), worden wijzigingen op schijf opgeslagen als de recordbuffer vol is.

Omdat wijzigingen in een tabel regelmatig op schijf worden opgeslagen, is het in de meeste gevallen niet nodig SET AUTOSAVE in te schakelen. Als SET AUTOSAVE is uitgeschakeld, gaat de verwerking van gegevens sneller omdat wijzigingen minder vaak naar schijf worden geschreven.

Voorbeeld

In het volgende voorbeeld wordt SET AUTOSAVE ON gebruikt om nieuw ingevoerde gegevens naar schijf te schrijven als de recordaanwijzer wordt verplaatst:

```
SET TALK OFF
CLOSE DATABASES
USE Bedrijf
auto_opsl = SET("AUTOSAVE")
carry = SET("CARRY")
SET AUTOSAVE ON
SET CARRY ON
APPEND
SET CARRY &carry
SET AUTOSAVE &auto_opsl
FLUSH
CLOSE ALL
CLEAR
SET TALK ON
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CLOSE..., FLUSH

SET BELL

Omgeving

Stelt de frequentie en duur van het geluidssignaal van de computer in.

Syntaxis

SET BELL ON | off
SET BELL TO
[<*frequentie Nuitdr*>, <*duur Nuitdr*>]

<frequentie Nuitdr>

De frequentie van het geluidssignaal in trillingen per seconde. Deze waarde moet een geheel getal tussen 19 en 10.000 zijn.

<duur Nuitdr>

De duur van het geluidssignaal in tikken (18den van een seconde). Deze waarde moet een geheel getal van 1 tot en met 19 zijn.

Standaardinstelling

De standaardinstelling voor SET BELL is ON. De standaardfrequentie is 512 Hertz (trillingen per seconde), en de standaardduur is 2 tikken. (Een tik is een 18de van een seconde.)

Als u de standaardinstelling voor aan/uit wilt wijzigen, wijzigt u de parameters voor BELL in de sectie [OnOffCommandSettings] in DBASEWIN.INI. Als u de

standaardinstellingen voor frequentie en duur wilt wijzigen, wijzigt u de parameters voor BELL in de sectie [CommandSettings] in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameters rechtstreeks wijzigen in DBASEWIN.INI.

Beschrijving

Als SET BELL is ingeschakeld (ON), klinkt een geluidssignaal als u een invoervak voor gegevens hebt gevuld of als u ongeldige gegevens invoert. SET BELL TO bepaalt de frequentie en duur van het geluidssignaal.

CHR(7) geeft altijd een geluidssignaal, onafhankelijk van de instelling van SET BELL.

SET BELL TO zonder argumenten stelt de standaardwaarden voor frequentie en duur in (512 en 2). SET BELL heeft alleen effect op een systeem dat over een interne luidspreker of een ander geluidssysteem beschikt.

Voorbeeld

In de volgende voorbeelden wordt het geluidssignaal ingesteld op een hoge en een lage toon met respectieveljik een korte en een lange duur:

```
SET BELL ON
SET BELL TO 50,19
* BROWSE, APPEND, enz. geven geluidssignaal
* Lang, laag geluidssignaal
? CHR(7)  && geluidssignaal klinkt
SET BELL TO 10000,1
* Kort, hoog geluidssignaal
? CHR(7)  && geluidssignaal klinkt
SET BELL OFF
? CHR(7)  && geluidssignaal klinkt toch
* BROWSE, APPEND, enz. geven geen geluidssignaal
```

Overdraagbaarheid

SET BELL TO *<frequentie Nuitdr>*, *<duur Nuitdr>* wordt niet ondersteund in dBASE III PLUS.

Zie ook

CHR( ), SET CONFIRM

SET BLOCKSIZE

Velden en records

Wijzigt de standaardblokgrootte van memobestanden en .MDX-indexbestanden.

Syntaxis

SET BLOCKSIZE TO *<Nuitdr>*

<Nuitdr>
Een getal van 1 tot en met 63 dat de grootte bepaalt van blokken in memobestanden en .MDX-indexbestanden. (De werkelijke grootte in bytes is het opgegeven getal vermenigvuldigd met 512.)

Standaardinstelling

De standaardinstelling voor SET BLOCKSIZE is 1 (vanwege compatibiliteit met dBASE III PLUS). De standaardinstelling voor BLOCKSIZE kunt u wijzigen in DBASEWIN.INI.

Beschrijving

Met SET BLOCKSIZE kunt u de grootte wijzigen van de blokken waarin memobestanden en .MDX-indexbestanden op schijf worden opgeslagen. De werkelijke grootte van de blokken is *<Nuitdr>* maal 512 bytes. In plaats van met SET BLOCKSIZE kunt u de blokgrootte van memobestanden en .MDX-indexbestanden ook afzonderlijk instellen met SET MBLOCK en SET IBLOCK.

Nadat de blokgrootte is veranderd, hebben memovelden die zijn gemaakt met COPY, CREATE en MODIFY STRUCTURE de nieuwe blokgrootte. Als u de blokgrootte van een bestaand memobestand wilt wijzigen, moet u eerst de blokgrootte wijzigen met SET BLOCKSIZE en vervolgens de tabel met het betreffende memoveld naar een nieuw bestand kopiëren. Het nieuwe bestand heeft dan de nieuwe blokgrootte.

Voorbeeld

In het volgende voorbeeld wordt SET BLOCKSIZE gebruikt om een kopie van de tabel Klanten te maken, De kopie heeft een memoblokgrootte van 1024 bytes in plaats van de standaardgrootte van 512 bytes:

```
USE Klanten
? SET("Blocksize")          && Geeft standaardinstelling;
                               (1) als resultaat. Elk;
                               memoblok = 512 bytes

SET BLOCKSIZE TO 2
COPY TO Klanten2
USE Klanten2
? SET("Blocksize")          && Geeft 2 als resultaat
LIST FILES LIKE *.DBT       && Bestand Klanten2.DBT is;
                               groter dan Klanten.DBT
CLOSE DATABASES
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

COPY, COPY INDEXES, CREATE, MODIFY STRUCTURE, INDEX, REINDEX, REPLACE, SET( ), SET IBLOCK, SET MBLOCK

SET BORDER — Omgeving

Definieert het standaardkader dat wordt gebruikt in alle volgende definities van dBASE IV-vensters en -popup-menu's en @...TO-kaders. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE om *formulieren* te maken in plaats van dBASE IV-vensters.

Zie Help voor meer informatie over SET BORDER. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het werken met formulieren.

SET CARRY — Velden en records

Bepaalt vanuit welke velden gegevens worden gekopieerd naar nieuwe records die worden gemaakt met APPEND, BROWSE, EDIT of INSERT.

Syntaxis

SET CARRY on | OFF
SET CARRY TO [<*veldenlijst*> [ADDITIVE]]

<*veldenlijst*>

De lijst van velden waaruit waarden naar nieuwe records moeten worden gekopieerd.

ADDITIVE

Voegt de velden in <*veldenlijst*> toe aan een eerder met SET CARRY TO gedefinieerde veldenlijst. Zonder ADDITIVE wordt de vorige <*veldenlijst*> overschreven door de nieuwe lijst.

Standaardinstelling

De standaardinstelling voor SET CARRY is OFF.

Beschrijving

Als SET CARRY is ingeschakeld (ON), worden records die worden toegevoegd met APPEND, BROWSE, EDIT of INSERT, gevuld met de inhoud van het record dat zich onmiddellijk voor het nieuwe record aan het logische eind van de tabel bevindt. (Met het argument DEFAULT van @...SAY...GET kunt u ook een standaardinhoud opgeven voor de records die u toevoegt met INSERT of APPEND.)

Als SET CARRY is uitgeschakeld (OFF), zijn nieuwe records in eerste instantie leeg. SET CARRY is niet van invloed op INSERT AUTOMEM, APPEND AUTOMEM, INSERT BLANK of APPEND BLANK. Deze commando's voegen altijd een record met automatische geheugenvariabelen of een leeg record toe.

Als u een veldenlijst opgeeft bij het commando SET CARRY TO, beperkt u de velden die naar het nieuwe record worden gekopieerd. Als u SET CARRY TO gebruikt, wordt

CARRY automatisch ingeschakeld (ON). Geef het sleutelwoord ADDITIVE op om velden toe te voegen aan een bestaande veldenlijst.

SET CARRY TO zonder veldenlijst stelt de standaardinstelling opnieuw in waarbij alle velden naar het nieuwe record worden gekopieerd.

U kunt ook een veldenlijst opgeven met SET FIELDS TO. De veldenlijst die u opgeeft bij SET CARRY TO, is alleen van invloed op SET CARRY, terwijl een met SET FIELDS TO ingestelde veldenlijst geldt voor alle commando's die bewerkingen uitvoeren op tabellen. Een met SET FIELDS TO ingestelde veldenlijst heeft een hogere prioriteit dan een met SET CARRY TO ingestelde veldenlijst.

Voorbeeld

In het volgende voorbeeld wordt SET CARRY TO gebruikt om gegevens uit het vorige veld te kopiëren. In dit voorbeeld wil de gebruiker records toevoegen voor nieuwe bedrijven die allemaal in hetzelfde gebied liggen en dus dezelfde Regio en Postcode hebben en die allemaal vallen onder hetzelfde hoofdkantoor:

```
USE Bedrijf
SET CARRY TO Regio, Postcode, Hoofdkant
APPEND
* Record verschijnt in bladervenster met
* gegevens in de 3 opgegeven velden.
SET CARRY OFF
CLOSE ALL
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

APPEND, INSERT, SET FIELDS, @...SAY...GET

SET CATALOG

Tabellen

Opent een catalogusbestand.

Syntaxis

SET CATALOG on | OFF
SET CATALOG TO [<*bestandsnaam*> | ?]

TO <*bestandsnaam*> | ?

Geeft de naam aan van de catalogus die u wilt openen. SET CATALOG TO ? toont een dialoogvenster, waarin u een bestaand catalogusbestand kunt selecteren.

Standaardinstelling

De standaardinstelling voor SET CATALOG is OFF. U kunt de standaardinstelling voor CATALOG wijzigen in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor CATALOG rechtstreeks in DBASEWIN.INI wijzigen.

Beschrijving

Met SET CATALOG TO *<bestandsnaam>* opent u een bestaande catalogus.

SET CATALOG on | OFF bepaalt of nieuwe bestanden worden toegevoegd aan het geopende catalogusbestand. Nadat CATALOG is ingeschakeld (ON), worden alle tabellen en bijbehorende bestanden zoals index-, query-, indelings-, rapport- en etiketbestanden die u gebruikt, of nieuwe bestanden die u maakt, toegevoegd aan de catalogus.

Gebruik SET CATALOG OFF om niet langer bestanden toe te voegen aan de catalogus. Als u weer bestanden aan de catalogus wilt toevoegen, gebruikt u opnieuw SET CATALOG ON. Met SET CATALOG TO zonder een bestandsnaam sluit u een catalogusbestand.

De namen van catalogusbestanden worden samen met beschrijvingen opgeslagen in een hoofdcatalogus, CATALOG.CAT. De beschrijvingen die u opgeeft voor elke catalogus, verschijnen in het catalogusvenster als u later SET CATALOG TO ? gebruikt om een catalogusnaam te kiezen.

Als u een catalogusbestand kiest, wordt het automatisch geopend in een eigen werkgebiedbuffer. Als u de catalogustabel wilt bewerken, moet u deze eerst openen in een voor de gebruiker toegankelijk werkgebied:

```
Catnaam=CATALOG()
USE(Catnaam) IN SELECT() AGAIN ALIAS Catalogus
```

Als u de catalogus wilt bijwerken vanuit een programma, schakelt u de catalogus tijdelijk uit:

```
Catnaam = CATALOG()
CatOpslag = SET("CATALOG")
SET CATALOG OFF
USE (Catnaam) IN SELECT() ALIAS Catalogus
* <catalogus bijwerken>
USE IN Catalogus
SET CATALOG TO (CatOpslag)
```

Als u een catalogus opent, wordt de inhoud vergeleken met het bestand op schijf. Als u bestanden hebt verwijderd terwijl SET CATALOG was uitgeschakeld, worden de bijbehorende vermeldingen uit de catalogus verwijderd.

Catalogusvermeldingen toevoegen

Als SET CATALOG is ingeschakeld (ON), worden nieuwe elementen automatisch aan de catalogus toegevoegd als u een van de volgende commando's gebruikt:

COPY STRUCTUREINDEX
COPY STRUCTURE EXTENDEDJOIN

CREATE SET FILTER
CREATE FROM SET FORMAT
CREATE | MODIFY FORMSET VIEW
CREATE | MODIFY LABELSORT
CREATE | MODIFY REPORTTOTAL
CREATE | MODIFY VIEWUSE
IMPORT FROM

Als de bestandsnaam al bestaat in de catalogus, wordt u om een bestandstitel gevraagd (als SET TITLE is ingeschakeld (ON)). Met het commando SET TITLE OFF kunt u deze vraag om bestandstitels uitschakelen.

Voorbeeld

In het volgende voorbeeld wordt eerst een catalogus gemaakt die alleen de tabellen Bedrijf en Orders bevat. Daarna wordt de catalogus geopend met SET CATALOG TO zodat de gebruiker niet met andere tabellen te maken krijgt. De gebruiker kan de catalogusweergave gebruiken in plaats van de bestandsweergave:

```
CLOSE ALL
CREATE CATALOG BedrCat
USE Bedrijf
USE Orders IN Select()
SET CATALOG TO
* BedrCat is gemaakt
SET CATALOG TO BedrCat
```

De gebruiker kan nu de catalogusweergave gebruiken en hoeft zich niet bezig te houden met andere tabellen in de subdirectory.

Zie ook

CATALOG( ), CREATE CATALOG, SELECT( ), SET( ), SET TITLE, USE

SET CENTURY

Datum- en tijdgegevens

Bestuurt de opmaak van jaartallen in datums.

Syntaxis

SET CENTURY on | off

Standaardinstelling

De standaardinstelling voor SET CENTURY wordt bepaald door de instelling bij de optie **Internationaal** in het Configuratiescherm van Windows. U kunt de standaardinstelling voor CENTURY wijzigen in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor CENTURY rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Als SET CENTURY is ingeschakeld (ON), worden jaartallen in datums in de huidige opmaak weergegeven met vier cijfers. Als SET CENTURY is uitgeschakeld (OFF), worden jaartallen in datums in de huidige opmaak weergegeven met twee cijfers.

Onafhankelijk van de instelling voor SET CENTURY kunt u jaartallen opgeven met 2, 3 of 4 cijfers. Als een jaartal uit twee cijfers bestaat, wordt aangenomen dat het een jaar in de twintigste eeuw betreft (19*xx*). Als SET CENTURY is uitgeschakeld (OFF), wordt het jaartal ingekort tot de twee laatste cijfers als datums worden weergegeven. Intern wordt echter de juiste waarde van de datum opgeslagen.

De volgende tabel geeft een overzicht van de weergave en opslag van datums bij de verschillende instellingen voor SET CENTURY. (Aangenomen wordt dat ITALIAN is ingesteld voor SET DATE.)

Datum ingevoerd als	Datum opgeslagen als	Weergave met SET CENTURY ON	Weergave met SET CENTURY OFF
{13-10-94}	13-10-1994	13-10-1994	13-10-94
{13-10-994}	13-10-0994	13-10-0994	13-10-94
{13-10-1994}	13-10-1994	13-10-1994	13-10-94
{13-10-2094}	13-10-2094	13-10-2094	13-10-94

Zoals u ziet heeft SET CENTURY geen invloed op de verhouding tussen de manier waarop u een datum invoert en de manier waarop deze wordt verwerkt en opgeslagen. SET CENTURY is alleen van invloed op de *weergave* van het jaartal in een datum.

Voorbeeld

Zie het voorbeeld bij SET DATE voor een voorbeeld met SET CENTURY.

Zie ook

DATE( ), SET DATE

SET COLOR TO

Kleuren en fonts

Definieert de weergavekleuren van dBASE IV-vensters en tekst die verschijnt in het resultatenpaneel van het commandovenster. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows definieert u kleuren voor objecten met de kenmerken ColorNormal en ColorHighlight.

Zie Help voor meer informatie over SET COLOR TO. Zie Hoofdstuk 10 in *Programmeren* voor meer informatie over het werken met objecten in dBASE voor Windows.

SET COLOR OF

Kleuren en fonts

Definieert de kleureninstelling voor opgegeven schermgebieden. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows definieert u kleuren voor objecten met de kenmerken ColorNormal en ColorHighlight. SET COLOR OF heeft geen invloed op formulieren.

Zie Help voor meer informatie over SET COLOR OF. Zie Hoofdstuk 10 in *Programmeren* voor meer informatie over het werken met objecten in dBASE voor Windows.

SET CONFIRM

Omgeving

Bestuurt de verplaatsing van de cursor van het ene invoervak naar het volgende tijdens gegevensinvoer in het resultatenpaneel van het commandovenster en in dBASE IV-vensters. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Het heeft geen invloed op dBASE voor Windows-formulieren.

Syntaxis

SET CONFIRM on | OFF

Standaardinstelling

De standaardinstelling voor SET CONFIRM is OFF. U kunt de standaardinstelling voor CONFIRM wijzigen in DBASEWIN.INI.

Beschrijving

Als SET CONFIRM is uitgeschakeld (OFF), wordt de cursor onmiddellijk naar het volgende invoergebied verplaatst als het huidige invoergebied is gevuld. Als SET CONFIRM is ingeschakeld (ON), wordt de cursor pas naar het volgende invoergebied verplaatst als u op *Enter* of een cursorbesturingstoets drukt, of in een ander invoergebied klikt met de muis.

Gebruik SET CONFIRM ON om te vermijden dat de cursor automatisch naar het volgende invoergebied wordt verplaatst. Op die manier voorkomt u invoerfouten, bijvoorbeeld dat gegevens van het ene invoergebied overlopen in het volgende. Gebruik SET CONFIRM OFF als snelheid van belang is bij invoer.

Voorbeeld

Er zijn twee instellingen voor SET CONFIRM:

```
SET CONFIRM OFF
SET CONFIRM ON
```

SET CONFIRM ON zorgt dat de cursor automatisch naar het volgende invoergebied wordt verplaatst als het huidige invoergebied is voltooid.

Als de volgende code wordt uitgevoerd terwijl SET CONFIRM is ingeschakeld (ON), moet de gebruiker op Enter drukken om van tMijnVar naar nMijnVar te gaan. Als CONFIRM is uitgeschakeld (OFF), wordt de cursor automatisch naar nMijnVar verplaatst nadat de gebruiker drie tekens heeft ingevoerd in tMijnVar.

```
SET CONFIRM ON                    && of SET CONFIRM OFF
tMijnVar = space(3)
nMijnVar = 0
CLEAR
@3,5 SAY "Instelling van CONFIRM is: "+SET("CONFIRM")
@5,5 SAY "Typ 3 letters: "GET tMijnVar PICTURE "AAA"
@6,5 SAY "Typ 3 cijfers: "GET nMijnVar PICTURE "999"
READ
```

Zie ook

SET BELL, SET CUAENTER

SET CONSOLE — Omgeving

Bestuurt de weergave van invoer en uitvoer in het resultatenpaneel van het commandovenster tijdens de uitvoering van een programma.

Syntaxis

SET CONSOLE ON | off

Standaardinstelling

De standaardinstelling voor SET CONSOLE is ON.

Beschrijving

Als SET CONSOLE is ingeschakeld (ON), verschijnt alle standaardinvoer en -uitvoer in het resultatenpaneel van het commandovenster. Gebruik SET CONSOLE OFF om deze weergave van invoer en uitvoer te voorkomen.

SET CONSOLE kan alleen worden gebruikt in een programma, niet in het invoerpaneel van het commandovenster. SET CONSOLE is niet van invloed op de weergave van foutmeldingen en waarschuwingen en ook niet op de weergave van invoer in het invoerpaneel van het commandovenster.

Een gebruiker kan wel invoer opgeven die door het programma wordt verlangd (bijvoorbeeld met WAIT of ACCEPT) terwijl SET CONSOLE is uitgeschakeld (OFF), maar de vraag om invoer en de invoer zelf worden niet weergegeven.

@...SAY...GET-instructies blijven zichtbaar, ook als SET CONSOLE is uitgeschakeld.

Voorbeeld

In dit voorbeeld wordt SET CONSOLE OFF gebruikt om de weergave op het scherm uit te schakelen tijdens het afdrukken van een rapport. Als het afdrukken is voltooid, wordt de uitvoer naar het scherm weer ingeschakeld:

```
SET CONSOLE OFF
REPORT FORM Rapport1 TO PRINTER
SET CONSOLE ON
```

Gebruik SET CONSOLE OFF alleen als u daar een goede reden voor hebt.

Zie ook

ACCEPT, ON ERROR, SET TALK, WAIT

SET COVERAGE

Foutafhandeling en testen op fouten

Bepaalt of een dekkingsbestand (.COV-bestand) wordt gemaakt en/of bijgewerkt.

Syntaxis

SET COVERAGE on | OFF

Standaardinstelling

De standaardinstelling voor SET COVERAGE is OFF. U kunt de standaardinstelling voor COVERAGE wijzigen in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor COVERAGE rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Een dekkingsbestand is een binair bestand waarin informatie wordt verzameld over het aantal keren dat elk logisch programmablok volledig wordt uitgevoerd (wordt begonnen en beëindigd). Gebruik SET COVERAGE als hulpmiddel bij het ontwikkelen van programma's om te bepalen welke programmaregels telkens worden uitgevoerd als u het programma start en welke niet.

Als SET COVERAGE is ingeschakeld (ON), en u roept een programma, procedure door de gebruiker gedefinieerde functie in een afzonderlijk programmabestand aan, wordt een nieuw dekkingsbestand gemaakt of een bestaand dekkingsbestand bijgewerkt. Als een dekkingsbestand wordt gemaakt, krijgt het de naam van het programma of de procedure en de extensie .COV.

De inhoud van een dekkingsbestand kunt u bekijken met DISPLAY COVERAGE of LIST COVERAGE. Als het dekkingsbestand onthult dat bepaalde regels niet worden uitgevoerd, kunt u het programma of de invoer voor het programma wijzigen zodat de regels wel worden uitgevoerd. Op die manier kunt u alle code in een programma testen.

U kunt het commando SET COVERAGE ON opgeven in een programmabestand of in het commandovenster. Als u SET COVERAGE ON opgeeft in een programmabestand, worden dekkingsbestanden gemaakt voor dat programma en voor elke subroutine die wordt beëindigd in een afzonderlijk programma- of procedurebestand dat door het hoofdprogramma wordt aangeroepen. U kunt in een programmabestand ook de instructie #pragma COVERAGE(ON) opnemen om een dekkingsbestand te maken voor het programma en voor alle programmabestanden die er door worden aangeroepen.

Als u het commando SET COVERAGE ON in het commandovenster opgeeft, worden dekkingsbestanden gemaakt voor alle programma's, procedures en door de gebruiker gedefinieerde functies in geopende procedurebestanden die vanuit het commandovenster worden aangeroepen tot u het commando SET COVERAGE OFF opgeeft.

Een logisch blok omvat geen commentaarregels en regels met programmeerconstructies zoals IF en ENDIF, dit in tegenstelling tot de programmaregels binnen een programmeerconstructie. Als het programma geen programmeerconstructies (IF, DO WHILE, FOR...NEXT, SCAN...ENDSCAN, LOOP, DO CASE, DO...UNTIL) bevat, bestaat het programma slechts uit één logisch blok dat alle programmaregels bevat (zonder de commentaarregels).

In het dekkingsbestand wordt een logisch blok aangeduid door de bijbehorende regelnummers:

```
Regel 1   * GEWIJZGD.PRG
Regel 2   SET TALK OFFBlok 1 (Regels 2-3)
Regel 3   USE Klanten INDEX Vertgnwrd
Regel 4   SCAN
Regel 5      DO CASE
Regel 6      CASE Vertgnwrd = "S-12"
Regel 7         SELECT 2Blok 2 (Regels 7-8)
Regel 8         USE S12
Regel 9      CASE Vertgnwrd = "L-5"
Regel 10        SELECT 2Blok 3 (Regels 10-11)
Regel 11        USE L5
Regel 12     CASE Vertgnwrd = "J-25"
Regel 13        SELECT 2Blok 4 (Regels 13-14)
Regel 14        USE J25
Regel 15     ENDCASE
Regel 16     DO WijzigenBlok 5 (Regels 16-17)
Regel 17     SELECT 1
Regel 18  ENDSCAN
Regel 19  CLOSE ALLBlok 6 (Regels 19-20)
Regel 20  SET TALK ON
```

Voordat dekkingsbestanden kunnen worden gemaakt, moeten het programmabestand en alle aangeroepen bestanden die u wilt analyseren, worden gecompileerd. Als u een programma compileert terwijl SET COVERAGE is uitgeschakeld (OFF), en vervolgens SET COVERAGE inschakelt en het programma start met DO, wordt geen dekkingsbestand gemaakt.

dBASE schrijft het dekkingsbestand naar schijf als het uit het geheugen wordt verwijderd of als u de opdracht LIST COVERAGE of DISPLAY COVERAGE geeft. Om een programma uit het geheugen te verwijderen, gebruikt u de opdracht CLEAR PROGRAM.

De preprocessor-instructie #pragma COVERAGE(ON) kan alleen worden gebruikt in een programmabestand. Als #pragma COVERAGE(ON) gebruikt, hoeft u niet het commando SET COVERAGE ON op te geven voordat u het programma compileert. Als u in een programma wijzigingen aanbrengt en het vervolgens opnieuw compileert, wordt een bestaand dekkingsbestand bijgewerkt.

Voorbeeld

In het volgende voorbeeld wordt SET COVERAGE ON gebruikt om de werking van een programma te testen:

```
* Werkbij.PRG
USE Klanten
SET COVERAGE ON
SCAN
  IF DELETED()
    DO Verwijdr
  ELSE
    DO Bijwerkn
  ENDIF
ENDSCAN
DISPLAY COVERAGE Werkbij
CLOSE ALL

PROCEDURE Bijwerkn
    * code voor bijwerken komt hier

PROCEDURE Verwijdr
    * code voor verwijderen komt hier
```

DISPLAY COVERAGE opent een .COV-bestand waarin u kunt zien welke regels in het programma zijn uitgevoerd. Als geen records zijn gemarkeerd voor verwijderen, zal het dekkingsbestand aantonen dat een logisch blok niet is uitgevoerd: de regel DO Verwijdr. De aanroep naar de procedure Verwijdr kan worden getest door het programma opnieuw te starten en een testrecord te verwijderen. Als het goed is, is regel 5 dan wel uitgevoerd.

Als SET COVERAGE is ingeschakeld (ON) en elke regel in het bestand met het programma, de procedure of de door de gebruiker gedefinieerde functie wordt uitgevoerd, vermeldt het dekkingsbestand een dekkingspercentage van 100%.

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

#pragma, CLEAR PROGRAM, COMPILE, DEBUG, DISPLAY COVERAGE, SET DEVELOPMENT

SET CUAENTER

Bepaalt de werking van *Enter*.

Syntaxis

SET CUAENTER ON | off

Standaardinstelling

De standaardinstelling voor SET CUAENTER is ON. U kunt de standaardinstelling voor CUAENTER wijzigen in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor CUAENTER rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Met SET CUAENTER bestuurt u de functie van *Enter*.

In dBASE voor Windows drukt u op *Enter* om het huidige formulier voor te leggen. Dat is normaal in Windows-toepassingen. In dBASE DOS verplaatst u met *Enter* echter de cursor van het ene GET-veld naar het volgende. In dBASE voor Windows kunt u de focus naar een ander object verplaatsen met de muis of door op *Tab* of *Shift-Tab* te drukken in plaats van op *Enter*.

Als u de focus net zo wilt verplaatsen als onder DOS, moet u het commando SET CUAENTER OFF geven. U kunt dat doen als uw applicaties zoveel mogelijk gelijk moeten blijven aan uw DOS-applicaties.

Voorbeeld

In het volgende voorbeeld wordt een formulier gemaakt met drie invoervakken voor de tabel Contact en twee knoppen om de recordaanwijzer vooruit of achteruit te verplaatsen. SET CUAENTER OFF wordt gebruikt om de cursor op dezelfde wijze te verplaatsen als in een DOS-applicatie. Dat wil zeggen dat de focus wordt verplaatst als de gebruiker op *Enter* drukt. Als u het commando wijzigt in SET CUAENTER ON, reageert de cursor alleen nog maar wanneer de gebruiker op *Tab* drukt of met de muis klikt:

```
CLOSE ALL
USE Bedrijf.DBF
SET CUAENTER OFF
f=NEW INVOER()
f.OPEN()
CLASS Invoer OF FORM
  this.Top=2
  this.Left=2
  this.Width=38
  this.Height=13
  this.Text= "Wijzigen naar behoefte"
  this.OnClose={;SET CUAENTER ON}
```

```
* syntaxis met operator NEW:
    CompCode=NEW ENTRYFIELD(this)
    CompCode.DataLink="Bedrijf->CompCode"
    CompCode.Top=5
    CompCode.Left=2
    CompCode.Width=6
    CompCode.Height=1.5
* Gecombineerde syntaxis:
    DEFINE ENTRYFIELD Type OF THIS
      this.Type.Width=6
      this.Type.Top=5
      this.Type.Left=9
      this.Type.Height=1.5
      this.Type.DataLink="Bedrijf->Type"
* syntaxis met DEFINE object:
      DEFINE ENTRYFIELD Bedrijf OF THIS;
        PROPERTY;
          Width 20,;
          Top 5,;
          Left 16,;
          Height 1.5,;
          DataLink "Bedrijf->Bedrijf"
  DEFINE TEXT Tekst1 OF THIS;
    PROPERTY;
      Text "Bedrijfsinfo",;
      Width 34,;
      Top 1,;
      Left 2,;
      Alignment 4,;
      Height 2.50,;
      FontSize 12.00,;
      Border .T.
  DEFINE PUSHBUTTON Terug OF THIS AT 10,10;
    PROPERTY Text "<<", Height 2,;
      OnClick {;SKIP-1}
  DEFINE PUSHBUTTON Verder OF THIS AT 10,20;
    PROPERTY TEXT ">>", Height 2,;
      OnClick {;SKIP}
ENDCLASS
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

READ, @...SAY...GET

SET CURRENCY

Met SET CURRENCY left | right kunt u het valutasymbool links of rechts van een bedrag plaatsen als u het symbool "$" of de functie TRANSFORM( ) gebruikt. SET CURRENCY TO bepaalt welk teken of welke tekens voor het valutasymbool worden gebruikt bij de clausules PICTURE en FUNCTION.

Syntaxis

SET CURRENCY left | right
SET CURRENCY TO
  [<*Tuitdr*>]

LEFT

Plaatst het valutasymbool links van het bedrag.

RIGHT

Plaatst het valutasymbool rechts van het bedrag

<*Tuitdr*>

Het teken of de tekens die als valutasymbool worden gebruikt. Hoewel dBASE geen beperkingen oplegt aan de lengte van <*Tuitdr*>, worden alleen de eerste negen tekens herkend. U mag geen cijfers gebruiken in <*Tuitdr*>.

Standaardinstelling

De standaardinstelling van SET CURRENCY wordt bepaald door de optie **Internationaal** in het Configuratiescherm van Windows. U kunt de standaardinstellingen interactief wijzigen met het commando SET of door de parameters rechtstreeks in DBASEWIN.INI op te geven.

- De standaardpositie van het valutasymbool kunt u wijzigen door de parameter voor CURRENCY in de sectie [OnOffCommandSettings] in DBASEWIN.INI te wijzigen.
- De standaardtekens voor het valutasymbool kunt u wijzigen door de parameter voor CURRENCY in de sectie [CommandSettings] in DBASEWIN.INI te wijzigen.

Beschrijving

Met SET CURRENCY left | right stelt u de positie in van valutasymbolen in bedragen. Met SET CURRENCY TO kunt u een ander valutasymbool dan het standaardsymbool instellen.

Als LEFT is ingesteld voor SET CURRENCY, worden slechts zoveel valutasymbolen weergegeven als samen met de cijfers binnen tien posities links van het decimaalscheidingsteken passen.

SET CURRENCY TO zonder de optie <Tuitdr> stelt opnieuw de oorspronkelijke instelling voor het valutasymbool in (de instelling van de optie **Internationaal** in het Configuratiescherm van Windows).

Voorbeeld

Zie het voorbeeld bij INT( ) voor een voorbeeld met SET CURRENCY LEFT | RIGHT.

Overdraagbaarheid

SET CURRENCY left | right wordt niet ondersteund in dBASE III PLUS. SET CURRENCY TO *<Tuitdr>* wordt niet ondersteund in dBASE III PLUS of dBASE IV.

Zie ook

SET POINT, SET SEPARATOR, TRANSFORM( )

SET CURSOR

Toetsenbord- en muisacties

Bepaalt of de cursor zichtbaar of onzichtbaar (verborgen) is.

Syntaxis

SET CURSOR ON | off

Standaardinstelling

De standaardinstelling voor SET CURSOR is ON. U kunt de standaardinstelling voor CURSOR wijzigen in DBASEWIN.INI.

Beschrijving

Met SET CURSOR kunt u aangeven of de cursor zichtbaar is op het scherm of niet. Als SET CURSOR is uitgeschakeld (OFF), is de cursor onzichtbaar. Geef het commando SET CURSOR ON om de cursor weer te tonen.

Voorbeeld

In het volgende programma wordt gezorgd dat de cursor zichtbaar is tijdens de uitvoering van het programma. Na voltooiing van het programma wordt de oorspronkelijke weergave (al of niet zichtbaar) van de cursor opnieuw ingesteld:

```
CursorInst=SET("CURSOR")    && huidige instelling opslaan
SET CURSOR ON               && cursor aan zetten
* ...
SET CURSOR &CursorInst      && oorspronkelijke instelling herstellen
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

SET DATABASE — Tabellen

Stelt de standaarddatabase voor toegang tot tabellen in.

Syntaxis

SET DATABASE TO [*<databasenaam>*]

<databasenaam>

Geeft de naam aan van de database die u de huidige database wilt maken.

Beschrijving

Met SET DATABASE stelt u de huidige database in. De huidige database bepaalt de standaardplaats voor tabellen waarop commando's betrekking hebben. Met dit commando kunt u een van de databases kiezen die eerder zijn geopend met het commando OPEN DATABASE. Databases worden gedefinieerd met het IDAPI-configuratieprogramma. (Zie *Aan de slag* voor meer informatie over het werken met dit programma.)

Als u SET DATABASE TO zonder een databasenaam gebruikt, hebben commando's verder betrekking op tabellen in de huidige directory (of in de directory die is ingesteld met SET PATH).

Voorbeeld

Zie OPEN DATABASE en DATABASE( ) voor voorbeelden met SET DATABASE.

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CLOSE..., DATABASE( ), OPEN DATABASE, SET DBTYPE

SET DATE — Datum- en tijdgegevens

Bepaalt de opmaak die wordt gebruikt voor de weergave en invoer van datums. Zie SET DATE TO voor meer informatie over het *wijzigen* van de systeemdatum.

Syntaxis

SET DATE [TO]
AMERICAN | ANSI | BRITISH | FRENCH | GERMAN | ITALIAN | JAPAN | USA |
MDY | DMY | YMD

TO

Alleen opgenomen vanwege de leesbaarheid; TO heeft geen gevolgen voor de werking van het commando.

AMERICAN | ANSI | BRITISH | FRENCH | GERMAN | ITALIAN | JAPAN | USA | MDY | DMY | YMD

In de volgende tabel ziet u welke opmaak bij de verschillende opties hoort:

Optie	Opmaak
AMERICAN	MM/DD/JJ
ANSI	JJ.MM.DD
BRITISH	DD/MM/JJ
FRENCH	DD/MM/JJ
GERMAN	DD.MM.JJ
ITALIAN	DD-MM-JJ
JAPAN	JJ/MM/DD
USA	MM-DD-JJ
MDJ	MM/DD/JJ
DMJ	DD/MM/JJ
JMD	JJ/MM/DD

Standaardinstelling

De standaardinstelling voor SET DATE wordt bepaald door de optie **Internationaal** in het Configuratiescherm van Windows. De standaardinstelling in Nederland is ITALIAN. U kunt de standaardinstelling voor DATE wijzigen in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor DATE rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

SET DATE bepaalt hoe velden en geheugenvariabelen van het type datum worden weergegeven. Als SET CENTURY is ingeschakeld (ON), wordt het jaartal in elke opmaak weergegeven door vier cijfers.

De instelling van SET DATE heeft een hogere prioriteit dan een eerdere instelling met SET MARK. U kunt SET MARK echter gebruiken na SET DATE om het scheidingsteken in datums te wijzigen.

Voorbeeld

In de volgende voorbeelden worden SET DATE en SET CENTURY gebruikt om de weergave van datumgegevens te besturen:

```
datum = {01-04-94}          && of datum = {1-4-94}
SET CENTURY OFF
SET DATE AMERICAN
? datum                     && Geeft 04/01/94 als resultaat
SET CENTURY ON
? datum                     && Geeft 04/01/1994 als resultaat
SET DATE JAPAN
? datum                     && Geeft 1994/04/01 als resultaat
SET DATE GERMAN
? datum                     && Geeft 01.04.1994 als resultaat
SET DATE BRITISH
? datum                     && Geeft 01/04/1994 als resultaat
SET DATE ITALIAN
? datum                     && Geeft 01-04-1994 als resultaat
SET DATE FRENCH
? datum                     && Geeft 01/04/1994 als resultaat
SET DATE USA
? datum                     && Geeft 04-01-1994 als resultaat
SET DATE ITALIAN            && Nederlandse instellingen herstellen
SET CENTURY OFF
? datum                     && Geeft 01-04-94 als resultaat
```

Overdraagbaarheid

De opmaakopties JAPAN, USA, MDY, DMY en YMD worden niet ondersteund in dBASE III PLUS.

Zie ook

DATE( ), DMY( ), MDY( ), SET CENTURY, SET DATE TO, SET MARK

SET DATE TO

Datum- en tijdgegevens

Stelt de systeemdatum in. Zie SET DATE voor informatie over het wijzigen van de datumopmaak.

Syntaxis

SET DATE TO <*Tuitdr*>

<Tuitdr>

De tekenuitdrukking, in de huidige datumopmaak, die moet worden ingesteld als systeemdatum.

Standaardinstelling

De standaardinstelling voor de systeemdatum wordt bepaald door de optie **Datum/tijd** in het Configuratiescherm van Windows.

Beschrijving

Met SET DATE TO kunt u de systeemdatum wijzigen. Alle volgende resultaatwaarden van DATE( ) en datumstempels voor bestanden die u opslaat, zijn gebaseerd op de nieuwe datum.

Geef *<Tuitdr>* op in de huidige datumopmaak die wordt bepaald door SET DATE, DBASEWIN.INI of de optie **Internationaal** in het Configuratiescherm van Windows (in die volgorde). Dat wil zeggen, de instellingen bij SET DATE hebben een hogere prioriteit dan die in DBASEWIN.INI, en de instellingen in DBASEWIN.INI hebben weer een hogere prioriteit dan de instellingen in het Configuratiescherm van Windows. Let er op dat *<Duitdr>* overeenkomt met de datumopmaak die wordt gebruikt op het moment dat het programma wordt uitgevoerd.

De datum moet binnen het bereik van 1 januari 1980 tot en met 31 december 2099 liggen. Een jaartal van twee cijfers wordt beschouwd als een jaartal in de twintigste eeuw (19*xx*). Als u een jaar in de eenentwintigste eeuw wilt opgeven, typt u dus een jaartal van vier cijfers.

Voorbeeld

In het volgende voorbeeld worden SET DATE TO en SET TIME TO gebruikt om de systeemtijd te wijzigen op basis van invoer van de gebruiker:

```
SET PROCEDURE TO PROGRAM(1) ADDITIVE
DEFINE FORM Hoofd FROM 0,0 to 13,40;
   PROPERTY ColorNormal "BG+/BG",;
   Text "Systeem bijwerken"
DEFINE TEXT T1 OF Hoofd AT 3,5 ;
   PROPERTY TEXT "Huidige tijd (24 u.):",;
   Width 30
DEFINE ENTRYFIELD V1 OF Hoofd AT 3,28 ;
   PROPERTY Value SPACE(8), Picture "99:99:99",;
   Width 8
DEFINE TEXT T2 OF Hoofd AT 5,5 ;
   PROPERTY TEXT "Huidige datum:",;
   Width 20
DEFINE ENTRYFIELD V2 OF Hoofd AT 5,28 ;
   PROPERTY Value {}, Picture "99-99-99",;
   Width 8
DEFINE PUSHBUTTON Bijwerken OF Hoofd AT 9,7;
  PROPERTY TEXT "Systeemtijd- en datum bijwerken",;
  Height 2, Width 26, OnClick Bijwerken
OPEN FORM Hoofd

PROCEDURE Bijwerken
SET DATE TO DTOC(Form.V2.Value)
SET TIME TO Form.V1.Value
? "Systeem bijgewerkt"
```

```
CLOSE FORM Hoofd
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DATE( ), SET CENTURY, SET DATE, SET TIME

SET DBTYPE

Tabellen

Stelt het standaardtabeltype in (Paradox of dBASE).

Syntaxis

SET DBTYPE TO [PARADOX | DBASE]

PARADOX | DBASE

Stelt Paradox of dBASE in als standaardtabeltype.

Standaardinstelling

De standaardinstelling voor SET DBTYPE TO is DBASE. U kunt de standaardinstelling voor DBTYPE wijzigen in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor DBTYPE rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Met SET DBTYPE stelt u het standaardtabeltype in dat wordt gebruikt door commando's die een tabel openen of maken. U kunt deze instelling negeren door een extensie op te geven: .DBF voor een dBASE-tabel of .DB voor een Paradox-tabel.

SET DBTYPE TO zonder argument stelt de standaardinstelling (dBASE) opnieuw in.

Voorbeeld

In het volgende voorbeeld wordt SET DBTYPE gebruikt om een relatie tussen twee .DBF-bestanden voor te bereiden. Vervolgens worden bepaalde velden naar een Paradox-tabel gekopieerd en wordt met SET DBTYPE de omgevingsvariabele voor het tabeltype gewijzigd in Paradox, zodat de nieuwe tabel kan worden gebruikt en bekeken. Tenslotte wordt SET DBTYPE gebruikt om de omgevingsvariabele weer te wijzigen in DBASE:

```
CLOSE DATABASES
SET SAFETY OFF
SET DBTYPE TO DBASE
```

```
USE Bedrijf ORDER CompCode IN SELECT()
USE Contact IN SELECT()
SELECT Contact
SET RELATION TO CompCode INTO Bedrijf
SELECT Contact
COPY TO CntctLst TYPE PARADOX;
   FIELDS Bedrijf->Bedrijf, ;
   Contact->CompCode, Contact->Contact, ;
   Bedrijf->Straat1, Bedrijf->Straat2, ;
   Bedrijf->Plaats, Bedrijf->Regio, ;
   Bedrijf->Postcode
CLOSE DATABASES
SET DBTYPE TO PARADOX
USE CntctLst
BROWSE
CLOSE DATABASES
SET DBTYPE TO DBASE
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CLOSE..., COPY TABLE, CREATE, DATABASE( ), DELETE TABLE, MODIFY STRUCTURE, OPEN DATABASE, RENAME TABLE, SET DATABASE, USE

SET DECIMALS

Numerieke gegevens

Bepaalt het aantal decimalen (cijfers achter de komma) dat wordt weergegeven in getallen.

Syntaxis

SET DECIMALS TO
  [<*Nuitdr*>]

<Nuitdr>

Het aantal decimalen, van 0 tot en met 18.

Standaardinstelling

De standaardinstelling voor SET DECIMALS is 2. U kunt de standaardinstelling wijzigen met de parameter voor DECIMALS in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor DECIMALS rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Met SET DECIMALS kunt u opgeven hoeveel decimalen worden weergegeven in getallen. SET DECIMALS is van invloed op de *weergave* van de meeste berekeningen, maar niet op de manier waarop getallen worden opgeslagen op schijf of in het geheugen.

SET DECIMALS TO zonder <*Nuitdr*> stelt het standaardaantal decimalen (2) opnieuw in.

Voorbeeld

In het volgende voorbeeld wordt SET DECIMALS gebruikt om de weergave van het aantal decimalen te regelen:

```
SET DECIMALS TO 2
? 2.22 * 3.3333              && Resulteert in 7,40
SET DECIMALS TO 0
? 2.22 * 3.3333              && Resulteert in 7
SET DECIMALS TO 7
? 2.22 * 3.3333              && Resulteert in 7,3999260
```

Zie de voorbeelden bij INT( ) en RANDOM( ) voor meer voorbeelden met SET DECIMALS.

Zie ook

INT( ), RANDOM( ), ROUND( ), SET PRECISION, VAL( )

SET DEFAULT

Stations- en bestandsfuncties

Bepaalt het standaardstation dat wordt gebruikt om bestanden te zoeken en op te slaan. SET DEFAULT wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE III PLUS. In dBASE voor Windows kunt u SET DIRECTORY gebruiken om in een keer een standaardstation en/of een standaarddirectory in te stellen.

Zie Help voor meer informatie over SET DEFAULT.

SET DELETED

Velden en records

Bepaalt of records worden verwerkt die zijn gemarkeerd voor verwijdering.

Syntaxis

SET DELETED ON | off

Standaardinstelling

De standaardinstelling voor SET DELETED is ON. U kunt de standaardinstelling voor DELETED wijzigen in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het

commando SET of de parameter voor DELETED rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Met SET DELETED geeft u aan of records in een tabel die zijn gemarkeerd voor verwijdering, wel of niet moeten worden verwerkt. Als SET DELETED is uitgeschakeld (OFF), verschijnen alle records in een tabel. Als SET DELETED is ingeschakeld (ON), worden records die zijn gemarkeerd voor verwijdering, uitgesloten van verwerking door commando's als LOCATE en LIST. De voor verwijdering gemarkeerde records bijven echter wel in de tabel aanwezig.

GO <*Nuitdr*>, INDEX, REINDEX, RECCOUNT( ) en elk ander commando dat wordt uitgevoerd met de optie RECORD <*Nuitdr*>, worden niet beïnvloed door SET DELETED. Als echter SET DELETED is ingeschakeld (ON) terwijl de records worden getoond, verplaatst GO <*Nuitdr*> de recordaanwijzer niet naar een record dat voor verwijdering is gemarkeerd.

Als met SET RELATION een relatie is ingesteld tussen twee tabellen, onderdrukt SET DELETED ON de weergave van verwijderde records in de subtabel. Het bijbehorende record in de hoofdtabel verschijnt echter wel, tenzij het hoofdrecord ook is verwijderd.

Voorbeeld

In het volgende voorbeeld wordt SET DELETED gebruikt om te voorkomen dat records die zijn gemarkeerd voor verwijdering, naar een andere tabel worden gekopieerd:

```
SET SAFETY OFF
USE Contact EXCLUSIVE IN SELECT()
INDEX ON CompCode TAG CompCode
USE Bedrijf IN SELECT()
SELECT Bedrijf
SET RELATION TO CompCode INTO Contact
DELETE FOR Bedrijf->Regio = "NW"
SET DELETED ON
COPY FIELDS Bedrijf->Bedrijf, Contact->Contact, ;
   Bedrijf->Straat1, Bedrijf->Straat2, ;
   Bedrijf->Plaats, Bedrijf->Regio,;
   Bedrijf->Postcode TO CntctLst TYPE PARADOX
SET DBTYPE TO PARADOX
USE CntctLst
BROWSE
CLOSE ALL
SET DBTYPE TO DBASE
SET SAFETY ON
```

Overdraagbaarheid

In dBASE IV is SET DELETED standaard uitgeschakeld (OFF). Als u in dBASE IV de recordaanwijzer verplaatst met het commando GO, wordt het record altijd getoond, onafhankelijk van de instelling van SET DELETED. In dBASE voor Windows echter wordt het record niet getoond als SET DELETED is ingeschakeld (ON) en het voor verwijdering is gemarkeerd.

Zie ook

DELETE, DELETED( ), PACK, RECALL, SET( ), SET FILTER, SET RELATION

SET DELIMITERS

Invoer/uitvoer

Bepaalt of het begin en eind van gegevensinvoervakken worden aangegeven door bepaalde, opgegeven scheidingstekens. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV en is niet van invloed op formulieren in dBASE voor Windows.

Zie Help voor meer informatie over de syntaxis van SET DELIMITERS.

SET DESIGN

Omgeving

Bepaalt of de commando's CREATE en MODIFY kunnen worden uitgevoerd.

Syntaxis

SET DESIGN ON | off

Standaardinstelling

De standaardinstelling voor SET DESIGN is ON. U kunt de standaardinstelling wijzigen met de parameter voor DESIGN in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor DESIGN rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Als SET DESIGN is ingeschakeld (ON), kunt u de commando's CREATE en MODIFY gebruiken om tabellen, formulieren, etiketten, rapporten, tekst en query's te maken en te wijzigen. Om te voorkomen dat gebruikers dit soort bestanden kunnen maken en/of wijzigen, kunt u SET DESIGN OFF in uw programma's uitschakelen.

Als u SET DESIGN in- of uitschakelt in een subroutine, is die instelling alleen van kracht tijdens de uitvoering van die subroutine.

Voorbeeld

De standaardinstelling is gewoonlijk ON. In dit voorbeeld is de standaardinstelling in DBASEWIN.INI gewijzigd in OFF zodat de gebruiker CREATE en MODIFY niet kan gebruiken:

```
* In DBASEWIN.INI
[OnOffCommandSettings]
design =OFF
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CREATE, CREATE FORM, CREATE LABEL, CREATE REPORT, MODIFY COMMAND, MODIFY FILE, MODIFY STRUCTURE

SET DEVELOPMENT

Programma's

Bepaalt of een programma-, procedure- of indelingsbestand automatisch wordt gecompileerd als u het bestand wijzigt en uitvoert of opent om het uit te voeren.

Syntaxis

SET DEVELOPMENT ON | off

Standaardinstelling

De standaardinstelling voor SET DEVELOPMENT is ON. U kunt de standaardinstelling wijzigen met de parameter voor DEVELOPMENT in DBASEWIN.INI. U kunt het commando SET gebruiken en de instelling voor Compilatie interactief wijzigen of de parameter voor DEVELOPMENT rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Als SET DEVELOPMENT is ingeschakeld (ON) en u voert een programmabestand uit met DO of u opent een procedure- of indelingsbestand, wordt het datum- en tijdstempel van het bronbestand vergeleken met dat van het gecompileerde bestand. Als het bronbestand van een later tijdstip is dan het gecompileerde bestand, wordt het bestand opnieuw gecompileerd.

Als SET DEVELOPMENT is ingeschakeld (ON) en u wijzigt een programma-, procedure- of indelingsbestand met MODIFY COMMAND, wordt het bijbehorende gecompileerde bestand gewist. Als u het programma uitvoert of het procedure- of indelingsbestand opent, wordt het opnieuw gecompileerd.

Als SET DEVELOPMENT is uitgeschakeld (OFF), worden geen datum- en tijdstempels vergeleken, en worden bestaande gecompileerde programma, procedure en indelingsbestanden uitgevoerd en geopend. Als u een bronbestand wijzigt en het vervolgens opent en uitvoert, wordt eerst in het geheugen gezocht naar het gecompileerde bestand en als dit wordt gevonden, wordt het uitgevoerd. Als het gecompileerde bestand niet in het geheugen wordt aangetroffen, wordt op schijf gezocht naar het gecompileerde bestand en als dit wordt gevonden, wordt het uitgevoerd. Als geen gecompileerd bestand wordt aangetroffen, wordt het bronbestand gecompileerd.

Als u een programma uitvoert met DO of een procedurebestand opent met SET PROCEDURE of een indelingsbestand opent met SET FORMAT, wordt altijd gezocht

naar een gecompileerd bestand. Als geen gecompileerde versie van het bronbestand wordt gevonden, wordt het gecompileerd, ongeacht de instelling van SET DEVELOPMENT.

Tijdens het ontwikkelen van een programma worden bestanden regelmatig gewijzigd en doet u er goed aan SET DEVELOPMENT in te schakelen (ON). U kunt er dan zeker van zijn dat altijd de meest actuele versie van het bestand wordt uitgevoerd.

Schakel SET DEVELOPMENT uit (OFF) als u klaar bent met het wijzigen van de broncode. Als SET DEVELOPMENT is uitgeschakeld, worden programma's sneller uitgevoerd omdat geen datum- en tijdstempels worden vergeleken. U kunt een bestand DBASEWIN.INI meeleveren met de gecompileerde code waarin u de parameter voor DEVELOPMENT instelt op OFF.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CLEAR PROGRAM, COMPILE, DO, SET PROCEDURE

SET DEVICE — Invoer/uitvoer

Stuurt de uitvoer van @...SAY...GET-commando's, of *non-streaming output* (niet-doorlopende uitvoer) naar het resultatenpaneel van het commandovenster of het huidige dBASE IV-venster, de printer of een bestand. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows SET PRINTER om *streaming output* (doorlopende uitvoer) naar de printer of een bestand te sturen.

Zie Help voor meer informatie over de syntaxis van SET DEVICE. Zie Hoofdstuk 24 in *Programmeren* voor meer informatie over streaming output, ofwel doorlopende uitvoer.

SET DIRECTORY — Stations- en bestandsfuncties

Wijzigt het huidige station en/of de huidige werkdirectory.

Syntaxis

SET DIRECTORY TO
  [<*pad*>]

<*pad*>

Een tekenuitdrukking die het standaardpad aangeeft. U begint <*pad*> met een backslash (\) om aan te geven dat het pad begint in de hoofddirectory van het station (zoals C:\). Als <*pad*> niet begint met een backslash of met .. (twee punten), begint het pad in de huidige directory.

Standaardinstelling

De standaardwerkdirectory is de directory die door HOME( ) als resultaat wordt gegeven. Dat is dus de directory met de programmabestanden van dBASE. U kunt de standaarddirectory wijzigen in Windows of in dBASE. In Windows kunt u een werkdirectory opgeven als u de programmagegevens van dBASE opgeeft of wijzigt. Als u hier een werkdirectory opgeeft, gebruikt dBASE die directory als standaarddirectory.

U kunt de instelling voor DIRECTORY in DBASEWIN.INI wijzigen. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor DIRECTORY rechtstreeks invoeren in DBASEWIN.INI.

Een in DBASEWIN.INI opgegeven directory heeft een hogere prioriteit dan de directory die u hebt opgegeven bij **Werkdirectory** in het dialoogvenster **Programmagegevens** van Windows.

Beschrijving

Met SET DIRECTORY kunt u het huidige werkstation en de huidige werkdirectory wijzigen. U kunt elk geldig station en/of pad opgeven. De huidige directory verschijnt in het vak **Huidige directory** in Navigator.

Er verschijnt een foutmelding als u probeert een station op te geven dat niet bestaat of dat niet gereed is. Als u niet zeker weet of een bepaald station geldig en gereed is, kunt u het station testen met de functie VALIDDRIVE( ) voordat u SET DIRECTORY gebruikt.

SET DIRECTORY TO .. (twee punten) wijzigt de directory in de directory één niveau hoger. SET DIRECTORY TO zonder de optie *<pad>* stelt opnieuw de directory in die is opgegeven bij **Werkdirectory** in het dialoogvenster **Programmagegevens** van Windows. Als daar geen directory is opgegeven, wordt de hoofddirectory van dBASE gebruikt.

U kunt ook het commando SET PATH gebruiken om toegang te krijgen tot bestanden in andere directory's. U kunt een of meer zoekpaden instellen. Als een bestand niet in de huidige directory wordt aangetroffen, wordt in deze zoekpaden gezocht. Gebruik SET PATH als de bestanden van een applicatie in meerdere directory's staan. Verder kunt u nog SET DEFAULT gebruiken om een standaardstation in te stellen waar naar bestanden wordt gezocht en waarin bestanden worden opgeslagen.

Als U met SET DIRECTORY het station wijzigt, wordt het standaardstation (SET DEFAULT) ook ingesteld op dat station.

SET DIRECTORY is vergelijkbaar met CD, alleen toont CD zonder argumenten het huidige station en de huidige directory. Zie CD voor meer informatie over paden.

Voorbeeld

In het volgende voorbeeld wordt eerst de huidige directory opgeslagen en vervolgens wordt SET DIRECTORY enkele malen gebruikt om een andere directory in te stellen zodat de bestanden in die directory kunnen worden getoond. Tenslotte wordt de oorspronkelijke directory opnieuw ingesteld:

```
* De directory's in dit
* voorbeeld zijn gemaakt met:
* MD D:\Project
* MD D:\Project\Progs
```

```
* MD D:\Project\Gegevens
* MD C:\Editor
OudeDir=SET("DIRECTORY") && Oorspronkelijke directory
SET DIRECTORY TO D:\Project\Gegevens
DIR                      && DBF-bestanden tonen
SET DIRECTORY TO D:\Project\Progs
DIR *.prg                && Programma's tonen
SET DIRECTORY TO ..
* niveau omhoog van D:\Project\Progs naar D:\Project
DIR *.*
SET DIRECTORY TO C:\Editor && C:\Editor
DIR *.doc
SET DIRECTORY TO &OudeDir && Oorspronkelijke directory herstellen
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CD, HOME( ), MKDIR, SET( ), SET PATH, VALIDDRIVE( )

SET DISPLAY — Omgeving

Past het resultatenpaneel van het commandovenster aan aan de opgegeven DOS-weergavemodus. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. U hoeft dit commando niet te gebruiken in nieuwe applicaties die alleen onder Windows worden uitgevoerd.

Zie Help voor meer informatie over SET DISPLAY.

SET ECHO — Foutafhandeling en testen op fouten

Met SET ECHO opent u de debugger van dBASE. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEBUG om de debugger te openen.

Zie Help voor meer informatie over de syntaxis van SET ECHO.

SET EDITOR — Omgeving

Geeft aan welke tekst-editor moet worden gebruikt voor het maken en wijzigen van programma- en tekstbestanden.

Syntaxis

SET EDITOR TO
[<*Tuitdr*>]

<*Tuitdr*>

De uitdrukking waarmee u de editor start op de opdrachtregel van DOS. Meestal is dat de naam van het uitvoerbare .EXE-bestand van het programma. U kunt ook de naam van een .PIF-bestand opgeven. Als <*Tuitdr*> niet het volledige pad naar het bestand bevat, wordt het bestand in de huidige directory en vervolgens in het DOS-pad gezocht.

Standaardinstelling

De standaardinstelling voor SET EDITOR is de interne Tekst-editor van dBASE voor Windows. U kunt een andere standaard-editor opgeven met de parameter voor EDITOR in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor EDITOR rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Met SET EDITOR kunt u een andere editor opgeven dan de standaard Tekst-editor van dBASE voor het maken en wijzigen van tekstbestanden. U kunt elke tekst-editor gebruiken die niet-opgemaakte ASCII-tekstbestanden maakt. De opgegeven editor wordt gestart als u het commando CREATE/MODIFY FILE of CREATE/MODIFY COMMAND geeft. Als u SET EDITOR TO gebruikt zonder een bestandsnaam op te geven voor <*Tuitdr*>, wordt de interne dBASE-editor opnieuw ingesteld.

Gebruik SET WP om een editor in te stellen voor het tonen en wijzigen van memovelden.

U kunt bij SET EDITOR een .PIF-bestand opgeven. Dat is een Windows-bestand waarmee u de Windows-omgeving voor een DOS-toepassing of een Windows .EXE-bestand bestuurt. Start de DOS-editor door het .PIF-bestand uit te voeren in plaats van het .EXE-bestand. Als u commando's opgeeft die in een DOS-venster worden uitgevoerd, wordt COMMAND.COM geladen via DBASEWIN.PIF in de _dbwinhome-directory. Met de PIF-editor van Windows kunt u de instellingen in DBASEWIN.PIF wijzigen. Raadpleeg uw Windows-handboek voor meer informatie over .PIF-bestanden. Als er niet genoeg geheugen beschikbaar is om de externe editor te starten, verschijnt de foutmelding "DOS kan niet worden uitgevoerd".

Als de tekst-editor die u hebt opgegeven, al in gebruik is als u een memo of bestand opent voor wijzigen, wordt een tweede versie van de editor gestart.

Voorbeeld

In het volgende voorbeeld wordt de standaard-editor gewijzigd in Brief en vervolgens weer in de interne editor van dBASE:

```
SET EDITOR TO "c:\dos\edit"
* c:\dos\edit hoeft nu niet in een
* zoekpad te staan
```

```
* Met MODIFY COMMAND wordt nu Edit gestart.
MODIFY COMMAND TIJD.PRG
SET EDITOR TO
MODIFY COMMAND TIJD
* Nu wordt de interne editor weer gebruikt
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

_dbwinhome, MEMORY( ), MODIFY COMMAND, MODIFY FILE, SET WP

SET ERROR

Foutafhandeling en testen op fouten

Geeft een tekenuitdrukking aan die voor foutmeldingen moet worden weergegeven en een tekenuitdrukking die op foutmeldingen moet volgen.

Syntaxis

SET ERROR TO
[<*voor Tuitdr*> [, <*na Tuitdr*>]]

<voor Tuitdr>

Een uitdrukking van maximaal 33 tekens die aan foutmeldingen vooraf moet gaan. Alle tekens na de eerste 33 worden genegeerd.

<na Tuitdr>

Een uitdrukking van maximaal 33 tekens die op foutmeldingen moet volgen. Alle tekens na de eerste 33 worden genegeerd. Als u een waarde wilt opgeven voor <*na Tuitdr*>, moet u ook een waarde of lege reeks ("") opgeven voor <*voor Tuitdr*>.

Standaardinstelling

De standaardinstelling die aan een foutmelding vooraf gaat, is "Fout: ". De reeks die op een foutmelding moet volgen, is standaard leeg. U kunt de standaardinstelling wijzigen met de parameter voor ERROR in DBASEWIN.INI. Gebruik daarbij deze opmaak:

```
ERROR = <voor Tuitdr> [, <na Tuitdr>]
```

Beschrijving

Met SET ERROR kunt u het begin en einde van foutmeldingen aanpassen. SET ERROR TO zonder argumenten stelt de standaardinstellingen opnieuw in.

SET ERROR is vergelijkbaar met ON ERROR. Met beide kunt u foutmeldingen aanpassen. Met SET ERROR kunt u echter alleen uitdrukkingen opgeven die voor en na een foutmelding moeten worden weergegeven, terwijl u met ON ERROR de melding

zelf kunt opgeven. Verder kunt u met ON ERROR wel een procedure aanroepen die een reeks instructies uitvoert en met SET ERROR niet.

Voorbeeld

Gebruik SET ERROR om foutmeldingen aan te passen.

```
SET ERROR TO "Niet helemaal juist: "
? "a" = 1          && genereert een runtime-fout
SET ERROR TO
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ERROR( ), MESSAGE( ), ON ERROR

SET ESCAPE

Toetsenbord- en muisacties

Bepaalt of de gebruiker op *Esc* kan drukken om de uitvoering van een programma af te breken.

Syntaxis

SET ESCAPE ON | off

Standaardinstelling

De standaardinstelling voor SET ESCAPE is ON. U kunt de standaardinstelling wijzigen met de parameter voor ESCAPE in DBASEWIN.INI.

Beschrijving

Als SET ESCAPE is ingeschakeld (ON), kan de gebruiker op *Esc* drukken om de uitvoering van een programma af te breken. Gebruik SET ESCAPE OFF in een programma om te voorkomen dat een onervaren gebruiker de uitvoering van het programma afbreekt tijdens de uitvoering van commando's als INDEX, PACK en COPY.

Opmerking Gebruik SET ESCAPE OFF alleen in geteste programma's. Als SET ESCAPE is uitgeschakeld en u hebt niet met ON KEY of SET KEY een andere toets aangewezen om het programma te onderbreken, kan het programma alleen worden afgebroken door de computer opnieuw te starten. Dat kan echter het verlies van gegevens tot gevolg hebben.

Onafhankelijk van de instelling voor SET ESCAPE, kunt u altijd op *Esc* drukken om de verwerking te onderbreken van commando's die wachten op invoer vanaf het toetsenbord, zoals ACCEPT, BROWSE, EDIT, INPUT en READ.

Voorbeeld

Het volgende voorbeeld bevat een fout in de subroutine WerktNiet. De aanwijzing en de test voor de lus zijn verschillend. Als SET ESCAPE is ingeschakeld (ON), kunt u het programma onderbreken door op *Esc* te drukken. U kunt dan het commando SUSPEND gebruiken om de variabele Meer of de uitvoering van de lus te onderzoeken:

```
SET ESCAPE ON
DO WerktNiet

PROCEDURE WerktNiet
Meer = ""
DO WHILE Meer <> "X" && Moet eigenlijk Upper(Meer)<>"X" zijn
  ? "Begin van de lus"
  * ...
  WAIT "Druk op X om lus te beëindigen " TO Meer
ENDDO
```

Zie ook

CLEAR TYPEAHEAD, INKEY( ), ON ERROR, ON ESCAPE, ON KEY, READKEY( ), SET KEY

SET EXACT

Tekenreeksgegevens

Bepaalt op basis van welke regels twee tekenreeksen als gelijk worden beschouwd.

Syntaxis

SET EXACT on | OFF

Standaardinstelling

De standaardinstelling voor SET EXACT is OFF. U kunt de standaardinstelling wijzigen met de parameter voor EXACT in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor EXACT rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Bij het vergelijken van twee tekenreeksen wordt het eerste teken van de reeks links van het is-gelijkteken vergeleken met het eerste teken van de reeks rechts van het is-gelijkteken, vervolgens wordt het tweede teken van de ene reeks vergeleken met het tweede teken van de andere reeks, enzovoort, tot twee tekens niet aan elkaar gelijk zijn of tot alle tekens in de reeks rechts van het is-gelijkteken zijn onderzocht.

Als de reeks aan de rechterkant de meeste tekens bevat, wordt altijd .F. als resultaat gegeven als tekenreeksen worden vergeleken. Als u weet dat de ene reeks langer is dan de andere, moet u de langste dus altijd links van het is-gelijkteken plaatsen.

Als SET EXACT is ingeschakeld (ON), worden twee tekenreeksen alleen als gelijk beschouwd als deze identiek zijn. Als SET EXACT is uitgeschakeld (OFF), worden twee tekenreeksen als gelijk beschouwd als alle tekens in de reeks rechts van het is-gelijkteken gelijk zijn aan de eerste tekens in de reeks links van het gelijkteken.

De instructie ? "abc"="abcdef" resulteert bijvoorbeeld onafhankelijk van de instelling voor SET EXACT in .F., omdat de reeks aan de rechterkant langer is dan die aan de linkerkant.

Als SET EXACT is uitgeschakeld (OFF), resulteert ? "abcdef"="abc" in .T. omdat de tekens in de reeks aan de rechterkant gelijk zijn aan de eerste tekens in de reeks aan de linkerkant. Als SET EXACT is ingeschakeld (ON), resulteert ? "abcdef"="abc" echter in .F., omdat de reeksen niet volkomen identiek zijn.

Als SET EXACT is uitgeschakeld (OFF) en de rechterreeks is leeg (""), wordt .T. als resultaat gegeven als een reeks wordt vergeleken met de lege tekenreeks. Als SET EXACT is ingeschakeld (ON) en de rechterreeks is leeg, wordt alleen .T. als resultaat gegeven als de tekenreeks aan de linkerkant ook leeg is.

Als u een exacte vergelijking wilt forceren (onafhankelijk van de instelling voor SET EXACT), kunt u == gebruiken in plaats van =, zoals in het volgende voorbeeld.

```
SET EXACT OFF
? "abcd" = "abc"    && geeft .T. als resulaat
? "abcd" = "abc"    && geeft .F. als resulaat
```

In taalaansturingen met *primaire* en *secundaire zwaarte* voor tekens, worden tekens vergeleken op basis van de primaire zwaarte als SET EXACT is uitgeschakeld (OFF) en op basis van de secundaire zwaarte als SET EXACT is ingeschakeld (ON). De secundaire zwaarte is met name van belang voor de verwerkig van tekens met accenten. De Nederlandse taalaansturing kent, net als de meeste andere Europese taalaansturingen, primaire en secundaire zwaarte voor tekens, de Amerikaanse taalaansturing niet. Als SET EXACT is uitgeschakeld, en de huidige taalaansturing is Nederlands, worden "geen" en "geënterd" bijvoorbeeld als gelijk beschouwd. Zie Appendix C in *Programmeren* voor meer informatie over taalaansturingen.

SET EXACT is van invloed op alle commando's waarbij tekenreeksen worden vergeleken, inclusief FIND, SEEK, SEEK( ), LOCATE, LOOKUP( ) en elk commando dat wordt uitgevoerd met de optie FOR of WHILE. In het geval van FIND en SEEK, wordt de opgegeven tekenreeks beschouwd als de reeks rechts in de vergelijking en het sleutelveld in het indexbestand als de tekenreeks aan de linkerkant.

Voorbeeld

In het volgende voorbeeld wordt SET EXACT gebruikt om te bepalen hoe nauwkeurig twee tekenreeksen moeten worden vergeleken. In het voorbeeld is de Nederlandse taalaansturing ingesteld.

```
SET EXACT ON
? "Will" = "Willem"             && Geeft .F. als resulaat
? "Willem" = "Will"             && Geeft .F. als resulaat
? "Geën" = "Geen"               && Geeft .F. als resulaat
SET EXACT OFF
? "Will" = "Willem"             && Geeft .F. als resulaat
```

```
? "Willem" = "Will"              && Geeft .T. als resulaat
? "Geënterd" = "Geen"            && Geeft .T. als resulaat
```

In het tweede voorbeeld wordt SET EXACT gebruikt om het verschil te demonstreren tussen het zoeken van een geheugenvariabele met het commando SEEK terwijl SET EXACT is uitgeschakeld en ingeschakeld:

```
SET TALK OFF
SET SAFETY OFF
SET EXACT OFF
USE Afnemers EXCLUSIVE
INDEX ON UPPER(Contact) TAG Contact
Zoekop="Marti"
SEEK UPPER(Zoekop)
IF FOUND()
   DISPLAY FIELDS Bedrijf,Contact
ELSE
   ? "SET EXACT is uitgeschakeld, record gevonden"
ENDIF

SET EXACT ON
SEEK UPPER(Zoekop)
IF FOUND()
   DISPLAY FIELDS Bedrijf,Contact
ELSE
   ? "SET EXACT is ingeschakeld, record niet gevonden"
ENDIF
SET TALK ON
SET SAFETY ON
CLOSE DATABASES
```

Overdraagbaarheid

In dBASE IV worden constanten geëvalueerd tijdens de compilatie en niet tijdens de uitvoering van een programma. Tijdens de compilatie resulteert ? "abc"="a" in .T. als SET EXACT is uitgeschakeld (OFF). Als u later in het commandovenster SET EXACT inschakelt, resulteert ? "abc"="a" in het programma nog steeds in .T.

In dBASE voor Windows worden constanten geëvalueerd tijdens de uitvoering van het programma. De instructie ? "abc"="a" resulteert in .T. of .F., afhankelijk van de instelling voor SET EXACT die geldt als het programma wordt uitgevoerd.

Zie ook

CHARSET( ), FIND, LDRIVER( ), LOCATE, LOOKUP( ), SEEK, SEEK( ), SET LDCHECK

SET EXCLUSIVE

Gedeelde gegevens

Bepaalt of tabellen en de bijbehorende index- en memobestanden exclusief worden geopend of voor gezamenlijk gebruik.

Syntaxis

SET EXCLUSIVE on | OFF

Standaardinstelling

De standaardinstelling voor SET EXCLUSIVE is OFF. U kunt de standaardinstelling wijzigen met de parameter voor EXCLUSIVE in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor EXCLUSIVE rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Als u SET EXCLUSIVE inschakelt (ON), worden tabellen (en bijbehorende index- en memobestanden) die u vervolgens opent, *exclusief* geopend, tenzij u deze expliciet met USE...SHARED opent. Als u een tabel exclusief opent, kunnen andere gebruikers het bestand en de bijbehorende index- en memobestanden niet openen, bekijken of wijzigen. Als u probeert een tabel te openen die door een andere gebruiker exclusief is geopend, wordt een foutmelding getoond.

SET EXCLUSIVE OFF zorgt dat tabellen en bijbehorende index- en memobestanden die u opent , worden geopend voor gezamenlijk gebruik, tenzij u deze opent met USE...EXCLUSIVE. Als een tabel die is geopend voor gezamenlijk gebruik in een gezamenlijke netwerkdirectory is geplaatst, kunnen andere gebruikers van het netwerk met toegang tot die directory het bestand en de bijbehorende index- en memobestanden openen, bekijken en wijzigen.

Als u SET INDEX gebruikt en de tabel is exclusief geopend, wordt de index exclusief geopend. Als de tabel is geopend voor gezamenlijk gebruik (met USE...SHARED), wordt de index geopend aan de hand van de optie in de instructie met USE.

Een index die is gemaakt met INDEX, wordt exclusief geopend. Het maakt dan niet uit of de tabel exclusief of voor gezamenlijk gebruik is geopend en ook de instelling van SET EXCLUSIVE is dan niet van belang. Nadat een index is gemaakt, kunt u de index openen voor gezamenlijk gebruik met USE...INDEX...SHARED of door achtereenvolgens de instructies SET EXCLUSIVE OFF en SET INDEX TO te geven.

De volgende commando's vereisen exclusief gebruik van een tabel met SET EXCLUSIVE ON of met USE...EXCLUSIVE:

- CONVERT
- COPY INDEXES
- DELETE TAG
- INDEX...TAG
- INSERT
- INSERT AUTOMEM
- INSERT BLANK
- MODIFY STRUCTURE

- PACK
- REINDEX
- ZAP

Voorbeeld

In het volgende voorbeeld wordt de tabel Bedrijf gebruikt met SET EXCLUSIVE ON zodat een nieuwe index kan worden gemaakt. Als de index is gemaakt, wordt Bedrijf opnieuw geopend, maar nu voor gezamenlijk gebruik:

```
CLOSE ALL
SET EXCLUSIVE ON
USE Bedrijf
INDEX ON CompCode+Bedrijf TAG BedrBedr
SET EXCLUSIVE OFF
* Volgende USE-commando's zijn niet exclusief,
* maar Bedrijf is nog exclusief geopend
USE Bedrijf ORDER BedrBedr
* Bedrijf is nu niet langer exclusief
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FLOCK( ), INDEX, RLOCK( ), SET INDEX, SET LOCK, USE

SET FIELDS

Velden en records

Definieert een groep velden in een tabel waartoe toegang kan worden verkregen.

Syntaxis

SET FIELDS on | OFF
SET FIELDS TO
[<*veldenlijst*> | ALL [LIKE <*filter1*>] [EXCEPT <*filter2*>]]

<*veldenlijst*> | ALL [LIKE <filter> | EXCEPT <filter2>]

De veldenlijst die in volgende bewerkingen wordt gebruikt. De veldenlijst mag velden bevatten uit geopende tabellen in alle werkgebieden en kan ook rekenvelden bevatten die alleen kunnen worden gelezen. In de volgende tabel worden de opties voor SET FIELDS TO beschreven:

Optie	Beschrijving
ALL	Alle velden in alle werkgebieden zijn beschikbaar
LIKE <*filter1*>	Alle velden in alle werkgebieden zijn beschikbaar waarvan de naam overeenkomt met <*filter1*>

Optie	Beschrijving
EXCEPT <*filter2*>	Alle velden in alle werkgebieden zijn beschikbaar behalve die waarvan de naam overeenkomt met <*filter1*>
LIKE <*filter1*> EXCEPT <*filter2*>	Alle velden in alle werkgebieden waarvan de naam overeenkomt met <*filter1*>, zijn beschikbaar, behalve die waarvan de naam overeenkomt met <*filter2*>

Standaardinstelling

De standaardinstelling voor SET FIELDS is OFF. Als u een veldenlijst instelt met SET FIELDS TO, wordt SET FIELDS echter automatisch ingeschakeld.

Beschrijving

Het commando SET FIELDS definieert een lijst van beschikbare velden in een of meer tabellen. U kunt velden in andere werkgebieden opgeven door voor de veldnaam de aliasnaam van het werkgebied op te geven. De veldenlijst die u opgeeft met SET FIELDS wordt pas actief als u SET FIELDS instelt op ON. Als SET FIELDS is uitgeschakeld (OFF), zijn alle velden in een tabel beschikbaar.

SET FIELDS wordt automatisch ingeschakeld (ON) als u een veldenlijst definieert met SET FIELDS TO <*veldenlijst*>. U kunt echter SET FIELDS OFF gebruiken om onder bepaalde omstandigheden opnieuw toegang te krijgen tot alle velden, waarbij u dan de veldenlijst weer actief kunt maken met SET FIELDS ON. Als u SET FIELDS ON gebruikt zonder eerst SET FIELDS TO <*veldenlijst*> te hebben gebruikt, is geen van de velden beschikbaar.

SET FIELDS TO zonder parameters wist de veldenlijst zodat alle velden weer beschikbaar zijn, en schakelt SET FIELDS uit (OFF). (Het commando CLEAR FIELDS wist ook de veldenlijst.)

SET FIELDS is van invloed op de volgende commando's. Als SET FIELDS is ingeschakeld (ON), kan bij deze commando's alleen tot de velden in <*veldenlijst*> toegang worden verkregen:

APPENDDISPLAY
AVERAGEEDIT
BLANK JOIN
BROWSE LIST
CALCULATE SUM
CHANGE SET CARRY
CREATE/MODIFY VIEWTOTAL
COPY
COPY STRUCTURE
COPY TO ARRAY

SET FIELDS TO is niet van invloed op de volgende commando's:

INDEX SET FILTER
LOCATESET RELATION

De veldenlijst die wordt opgegeven met SET FIELDS TO, kan zowel veldnamen uit tabellen als rekenvelden bevatten. Met de optie /R kunt u tabelvelden alleen leesbaar maken, bijvoorbeeld:

```
Salaris/R, Uren/R
```

Een rekenveld geeft u op door middel van een geldige uitdrukking. Bijvoorbeeld:

```
Bruto_sal = Uurloon * Uren
```

In filters kan het jokerteken * worden gebruikt voor nul, een of meer tekens en het jokerteken ? voor een afzonderlijk teken. ALL LIKE selecteert velden die voldoen aan het filter. ALL EXCEPT selecteert velden die niet voldoen aan het filter. SET FIELDS TO ALL neemt alle velden in de huidige tabel op in de veldenlijst.

U kunt velden toevoegen aan een eerder opgegeven veldenlijst door nogmaals SET FIELDS TO te gebruiken. Als u bijvoorbeeld SET FIELDS TO Veld1 en vervolgens SET FIELDS TO Veld2 opgeeft, worden Veld1 en Veld2 allebei in de veldenlijst opgenomen.

Voorbeeld

In het volgende voorbeeld wordt SET FIELDS in- of uitgeschakeld om te schakelen tussen een veldenlijst die is gedefinieerd met SET FIELDS TO, en een tabel in het huidige werkgebied:

```
SELECT 1
USE Contact ORDER CompCode
SELECT 2
USE Bedrijf
SET RELATION TO CompCode INTO Contact
SET FIELDS TO Contact->CompCode, Bedrijf->Bedrijf, ;
   Contact->Contact, Bedrijf->Telefoon
SET FIELDS OFF
BROWSE              && Bedrijf in bladervenster
SET FIELDS ON
BROWSE              && Veldenlijst weergeven
CLEAR FIELDS
```

Zie ook

CLEAR FIELDS, SET( ), SET CARRY, SET RELATION

SET FILTER

Tabelindeling

Stelt een voorwaarde waarmee wordt bepaald welke records worden verwerkt door dBASE-commando's.

Syntaxis

SET FILTER TO
[*<voorwaarde>*] | [FILE *<bestandsnaam>* | ? | *<bestandsnaamfilter>*]

<voorwaarde>

De voorwaarde waaraan records moeten voldoen om door volgende commando's te worden verwerkt.

FILE *<bestandsnaam>* | ? | *<bestandsnaamfilter>*

Het query-bestand dat de voorwaarde aangeeft. SET FILTER FILE ? en SET FILTER FILE *<bestandsnaamfilter>* tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand gezocht in de huidige directory en vervolgens in het pad dat is ingesteld met SET PATH. Als u een bestand zonder extensie opgeeft, wordt de extensie .QRY gebruikt.

Beschrijving

Gebruik SET FILTER als u slechts een deel van de records in een tabel wilt bekijken of verwerken. SET FILTER TO zonder voorwaarde of bestandsnaam zorgt dat alle records in de tabel weer worden verwerkt. U kunt voor elke geopende tabel in elk werkgebied een afzonderlijke filtervoorwaarde opgeven.

SET FILTER TO *<voorwaarde>* stelt een filter in aan de hand van een geldige uitdrukking. De voorwaarde kan records in de huidige tabel filteren op basis van een uitdrukking waarin alle gegevenstypen kunnen worden gebruikt behalve memo. U kunt bijvoorbeeld SET FILTER TO Achternaam = "Aarts" opgeven om records in een tekenveld te filteren of SET FILTER TO Vertrek > {01-01-94} om records te filteren op basis van een datumveld. U kunt ook voorwaarden combineren. Geef bijvoorbeeld SET FILTER TO Achternaam = "Aarts" .AND. Vertrek > {01-01-94}.

SET FILTER TO FILE *<bestandsnaam>* gebruikt een dBASE III PLUS query-bestand (.QRY-bestand) om een filtervoorwaarde in te stellen. U kunt voor dit commando geen .QBE-bestanden uit dBASE IV of dBASE voor Windows gebruiken.

Filters die zijn opgegeven met het commando SET FILTER, worden pas actief nadat de recordaanwijzer binnen een tabel is verplaatst (bijvoorbeeld met GO TOP of SKIP). Voor alle commando's die met een actieve tabel werken, zoals AVERAGE, BROWSE, EDIT en REPORT, kan gebruik wordne gemaakt van voorwaarden die zijn ingesteld met SET FILTER.

Een filtervoorwaarde is niet van invloed op commando's die naar records verwijzen met een recordnummer, zoals GOTO *<Nuitdr>*. Met GOTO *<Nuitdr>* kunt u de recordaanwijzer verplaatsen naar een record dat niet voldoet aan de filtervoorwaarde. U kunt dat record echter niet weergeven (met bijvoorbeeld het commando EDIT).

Voorbeeld

In het volgende voorbeeld wordt SET FILTER gebruikt om het volgende BROWSE-commando te beperken tot klanten in het noordwesten:

```
USE Afnemers
SET FILTER TO Regio = "NW"
BROWSE FIELDS Naam, Contact, Plaats, ;
  Regio, Keuken
SET FILTER TO          && filtervoorwaarden wissen
```

In het volgende voorbeeld wordt SET FILTER gebruikt om records in een gerelateerde tabel te filteren:

```
CLOSE DATA
USE Bedrijf EXCLUSIVE
INDEX ON CompCode TAG CompCode
SELE 2
USE Contact EXCLUSIVE
INDEX ON Contact TAG CompCode
SET RELATION TO CompCode INTO Bedrijf
SET FILTER TO Bedrijf->Regio="NW"
DISPLAY ALL ;
  Contact, CompCode,;
  Bedrijf->Bedrijf, Bedrijf->Regio
```

Alleen contactpersonen bij bedrijven in het noordwesten worden weergegeven.

Zie ook

CREATE QUERY, MODIFY QUERY, SET( ), SET DELETED, SET KEY

SET FORMAT — Invoer/uitvoer

Opent het opgegeven indelingsbestand in het geselecteerde werkgebied en sluit eventueel een indelingsbestand dat al was geopend. Als het indelingsbestand nog niet is gecompileerd, wordt het gecompileerd. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE om *formulieren* te maken in plaats van dBASE IV-vensters of indelingsbestanden.

Zie Help voor meer informatie over de syntaxis van SET FORMAT. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

SET FULLPATH — Omgeving

Bepaalt of functies die een bestandsnaam als resultaat geven, ook het volledige pad als resultaat geven.

Syntaxis

SET FULLPATH on | OFF

Standaardinstelling

De standaardinstelling voor SET FULLPATH is OFF. U kunt de standaardinstelling wijzigen met de parameter voor FULLPATH in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor FULLPATH rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Gebruik SET FULLPATH ON als u wilt dat functies als CATALOG( ), DBF( ), MDX( ) en NDX( ) resulteren in een bestandsnaam met het volledige pad. Als u bijvoorbeeld werkt met tabellen in SQL-databases, of met tabellen in verschillende directory's, kunt u SET FULLPATH inschakelen om te zorgen dat DBF( ) resulteert in de volledige bestandsnaam voor gebruik in volgende commando's.

Voorbeeld

In het volgende voorbeeld wordt de naam van de huidige tabel bepaald met FULLPATH ON en met FULLPATH OFF:

```
USE BEDRIJF
SET FULLPATH ON
? DBF()
* Resultaat met FULLPATH ON:
*   C:\DBASEWIN\VOORBD\BEDRIJF.DBF
SET FULLPATH OFF
? DBF()
* Resultaat met FULLPATH OFF:
*   C:BEDRIJF.DBF
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. Als u wilt dat bestandsnamen die als resultaat worden gegeven, dezelfde opmaak hebben als in dBASE III PLUS, gebruikt u SET FULLPATH OFF.

Zie ook

CATALOG( ), DBF( ), MDX( ), NDX( ), SET( )

SET FUNCTION

Toetsenbord- en muisacties

Kent een commando of uitdrukking toe aan een functietoets, al of niet in combinatie met Ctrl of Shift.

Syntaxis

SET FUNCTION <*toets*> TO <*Tuitdr*>

<*toets*>

Een functietoetsnummer, functietoetsnaam of tekenuitdrukking voor een functietoetsnaam, zoals `3`, `F3` of `"F3"`. Als u een combinatie van een functietoets met Ctrl of Shift wilt gebruiken, moet u voor <*toets*> een tekenuitdrukking opgeven. Typ `"CTRL-"` of `"SHIFT-"` gevolgd door de naam van de functietoets, zoals `"Shift-F5"` of `"Ctrl-f3"`. Er wordt geen onderscheid gemaakt tussen hoofdletters en kleine letters. Het is niet toegestaan Ctrl en Shift allebei te gebruiken, zoals in `"Ctrl-Shift-F3"`.

<Tuitdr>
Een dBASE-commando, functie, door de gebruiker gedefinieerde functie of een willekeurige tekenreeks. Plaats een puntkomma (;) achter het commando om het onmiddellijk uit te voeren als op *<toets>* is gedrukt. U kunt meerdere commando's uitvoeren door tussen de commando's in de lijst puntkomma's te plaatsen.

Standaardinstelling

Als u dBASE start, gelden de volgende functietoetstoewijzingen:

Toets	Commando	Toets	Commando
F1	HELP;	*F7*	DISPLAY MEMORY;
F3	LIST;	*F8*	DISPLAY;
F4	DIR;	*F9*	APPEND;
F5	DISPLAY STRUCTURE;	*F10*	Activeert het menu
F6	DISPLAY STATUS;		

Beschrijving

Met SET FUNCTION kunt u een commando toewijzen aan een functietoets of een combinatie van een functietoets met *Ctrl* of *Shift*. U kunt ook een tekenuitdrukking toewijzen aan een functietoets. Als u op de toets of toetscombinatie drukt terwijl de cursor in het commandovenster staat, verschijnt *<Tuitdr>* bij de cursor. Als *<Tuitdr>* eindigt op een puntkomma, wordt het commando onmiddellijk uitgevoerd. Als *<Tuitdr>* niet eindigt op een puntkomma, kunt u de uitdrukking wijzigen.

Opmerking *F2* is gereserveerd voor het schakelen tussen weergaven in het bladervenster. U kunt deze toets wel programmeren, maar in een bladervenster heeft dat geen effect. Functietoets *F10* en elke combinatie met *F11* of *F12* kunnen niet worden geprogrammeerd.

In formulieren zijn uitdrukkingen die zijn toegewezen met SET FUNCTION alleen geldig in invoervakken.

Zie ON KEY voor meer informatie over het programmeren van functietoetsen en andere toetsen.

Voorbeeld

In het volgende voorbeeld wordt het commando SET gegeven als de gebruiker op F9 drukt. Dit commando activeert de opdracht **Kenmerken | Bureaublad** waarmee de gebruiker instellingen kan opgeven voor SET-commando's:

```
SET FUNCTION F9 TO "SET;"
```

Met SET FUNCTION kunt u ook meerdere commando's tegelijk programmeren:

```
SET FUNCTION F8 TO "CLOSE ALL;CLEAR;"+;
  "USE Klanten;BROWSE;"
```

Zie ook

DISPLAY STATUS, FKLABEL( ), FKMAX( ), HELP, INKEY( ), ON KEY

SET HEADINGS

Invoer/uitvoer

Bestuurt de weergave van veldnamen in de uitvoer van AVERAGE, DISPLAY, LIST en SUM.

Syntaxis

SET HEADINGS ON | off

Standaardinstelling

De standaardinstelling voor SET HEADINGS is ON. U kunt de standaardinstelling wijzigen met de parameter HEADINGS in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor HEADINGS rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Als SET HEADINGS is ingeschakeld (ON), bevat de uitvoer van AVERAGE, DISPLAY, LIST en SUM kolomtitels die de velden in de tabellen aangeven. Als u de gegevens zonder kolomtitels wilt tonen, gebruikt u SET HEADINGS OFF voordat u AVERAGE, DISPLAY, LIST of SUM gebruikt.

Voorbeeld

In het volgende voorbeeld wordt DISPLAY gebruikt in combinatie met SET HEADING ON om veldnamen te tonen en SET HEADING OFF om veldnamen achterwege te laten:

```
USE Klanten
SET HEADING OFF
? "Zonder kolomtitels"
DISPLAY Klantnr, Naam
SET HEADING ON
? "Met kolomtitels"
DISPLAY Klantnr, Naam
*
* Zonder kolomtitels
*       1  1221 Midsland Duikapparatuur
*
* Met kolomtitels
* Record#  Klantnr   Bedrijf
*       1  1221 Midsland Duikapparatuur
```

Overdraagbaarheid

In dBASE III PLUS luidt dit commando SET HEADING (zonder *s* aan het eind).

In dBASE IV is SET HEADINGS ook van invloed op het commando TYPE Als SET HEADINGS is ingeschakeld wordt een titel getoond met de bestandsnaam en -datum. In dBASE voor Windows wordt deze titel niet getoond, ook als SET HEADINGS is ingesteld op ON.

Zie ook

AVERAGE, DISPLAY, LIST, SUM, TYPE

SET HELP

Programmeren in Windows

Bepaalt welk helpbestand (.HLP-bestand) het Helpsysteem van dBASE gebruikt.

Syntaxis

SET HELP TO
[<*help-bestandsnaam*> | ? | <*help-bestandsnaamfilter*>]

<*help-bestandsnaam*> | ? | <*help-bestandsnaamfilter*>

Geeft de naam aan van het Helpbestand. ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder extensie opgeeft, wordt de extensie .HLP gebruikt.

Beschrijving

Met SET HELP TO kunt u opgeven welk Helpbestand moet worden gebruikt als het dBASE-Helpsysteem wordt geactiveerd.

Het Helpbestand wordt automatisch geopend als u het bestand in de hoofddirectory van dBASE plaatst.

SET HELP TO sluit een eventueel geopend Helpbestand voordat het opgegeven bestand wordt geopend.

Voorbeeld

Met de volgende instructie toont u een dialoogvenster waarin u een keuze kunt maken uit de beschikbare Helpbestanden:

```
SET HELP TO ?    && of
SET HELP TO *.HLP
```

Zo stelt u een eigen Helpbestand in:

```
SET HELP TO MijnHelp.HLP
```

Zo stelt u het standaardhelpbestand van dBASE voor Windows opnieuw in:

```
SET HELP TO \DBASEWIN\BIN\DBASEWIN.HLP
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

HELP, HelpFile, HelpID, SET TOPIC

SET IBLOCK

Tabelindeling

Wijzigt de blokgrootte voor nieuwe .MDX-bestanden.

Syntaxis

SET IBLOCK TO *<Nuitdr>*

<Nuitdr>

Een getal van 1 tot 63 dat de grootte aangeeft van indexblokken in nieuwe .MDX-bestanden. De standaardwaarde is 1. (De werkelijke grootte in bytes is het getal dat u opgeeft, vermenigvuldigd met 512. De minimumgrootte van een blok is echter 1024 bytes.) U kunt de standaardinstelling voor IBLOCK wijzigen in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor IBLOCK rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Met SET IBLOCK kunt u de grootte wijzigen van de blokken die worden gebruikt voor de opslag van .MDX-bestanden op schijf. Op die manier kunt u de prestatie en doelmatigheid van indexen verbeteren. U kunt een blokgrootte opgeven van 1024 bytes tot ongeveer 32K. De instelling voor IBLOCK vervangt een eventueel eerder ingestelde blokgrootte die is gedefinieerd met het commando SET BLOCKSIZE of is opgegeven in DBASEWIN.INI. Nadat de blokgrootte is gewijzigd, worden nieuwe .MDX-indexbestanden gemaakt met de nieuwe blokgrootte.

Meervoudige indexbestanden (.MDX-bestanden) bestaan uit afzonderlijke indexblokken (of *nodes*). Nodes bevatten de waarde van sleutels die bij de afzonderlijke records horen, en geven informatie over de lokatie van het bijbehorende record voor elke sleutelwaarde. Omdat de instelling voor IBLOCK de grootte van de nodes bepaalt, bepaalt deze instelling tevens het aantal sleutelwaarden dat in elke node past. Als een enkele node niet alle sleutelwaarden in een index kan bevatten, worden een of meer tussenliggende blokken (parent nodes) gemaakt. Deze tussenliggende nodes bevatten ook sleutelwaarden. Tussenliggende nodes verwijzen niet naar recordnummers maar naar *leaf nodes* of tussenliggende nodes op een lager niveau. Als u de grootte van indexblokken verhoogt en een nieuw .MDX-bestand maakt, bevatten de nieuwe, grotere leaf nodes meer sleutelwaarden.

Of u de prestatie kunt verhogen door sleutelwaarden op te slaan in grotere of juist kleinere nodes is afhankelijk van verschillende factoren: de verdeling van de gegevens, de vraag of tabellen zijn gekoppeld, de lengte van sleutelwaarden, de waarde van INDEXBYTES en het type bewerking dat moet worden uitgevoerd. Gewoonlijk bevat elk .MDX-bestand meerdere indexlabels. De beste instelling voor een bepaald .MDX-bestand vindt u door experimenteren, omdat de optimale grootte voor de ene indexlabel niet de optimale grootte voor een andere hoeft te zijn.

In de volgende lijst staan de basisprincipes die de prestatie van een index bepalen.

- Omdat nodes niet sequentieel hoeven te zijn, wordt per keer maar één node van schijf gelezen. Meerdere nodes tegelijk lezen is gewoonlijk niet doelmatig omdat de tweede node in de meeste gevallen niet de volgende node in de sequentiële lijst is.
- Als een node eenmaal in het geheugen is gelezen, wordt getracht de node daar op te slaan voor later gebruik. De hoeveelheid ruimte die wordt gebruikt voor het opslaan in cache-geheugen van indexnodes, wordt bepaald door de instelling voor INDEXBYTES.
- Als gebruikers verschillende tabellen koppelen, met SET RELATION bijvoorbeeld, is de prestatie beter als alle relevante nodes voor de tabellen tegelijk in het geheugen aanwezig zijn. Als een grote node voor tabel B bijvoorbeeld een eerder gelezen node voor tabel A uit het geheugen verdringt, moet de node voor tabel A opnieuw op schijf worden gezocht en gelezen als deze opnieuw nodig is. Als beide nodes in het geheugen blijven, komt dat de prestatie ten goede.
- Als tabellen veel identieke sleutelwaarden hebben, moeten die misschien worden opgeslagen in een groot aantal nodes. In dat geval kan de prestatie worden verbeterd door de nodes groter te maken zodat minder nodes van schijf ingelezen hoeven te worden om hetzelfde aantal sleutelwaarden in het geheugen op te slaan.
- Kleinere nodes kunnen een verminderde prestatie tot gevolg hebben. Het programma probeert alle nodes in het geheugen te behouden (caching). Als kleine nodes door later gelezen nodes uit het geheugen worden verdrongen, blijft in het geheugen ruimte open die te klein is om grotere nodes aaneengesloten te lezen. Dat kan fragmentatie van het geheugen tot gevolg hebben en dat is weer slecht voor de prestatie.

Voorbeeld

In het volgende voorbeeld worden twee .MDX-bestanden gemaakt met identieke gegevens maar met verschillende instellingen voor IBLOCK en dus verschillende bestandsgrootten:

```
CLOSE DATA
DELETE FILE Co1.mdx
DELETE FILE Co2.mdx
* Oude .mdx-bestanden verwijderen
* Nieuwe .mdx-bestanden maken
USE Bedrijf EXCLUSIVE
SET IBLOCK TO 2
INDEX ON CompCode TAG CompCode OF Co1
INDEX ON Bedrijf  TAG Bedrijf  OF Co1
INDEX ON Plaats TAG Plaats        OF Co1
SET IBLOCK TO 20
INDEX ON CompCode TAG CompCode OF Co2
INDEX ON Bedrijf  TAG Bedrijf  OF Co2
INDEX ON Plaats TAG Plaats        OF Co2
DIR CO?.MDX
```

De beide MDX-bestanden, Co1.MDX en Co2.MDX, hebben verschillende bestandsgrootten omdat de indexen zijn gemaakt met verschillende instellingen voor IBLOCK.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

COPY, COPY INDEXES, CREATE, MODIFY STRUCTURE, INDEX, REINDEX, REPLACE, SET( ), SET BLOCKSIZE, SET MBLOCK,

SET INDEX — Tabelindeling

Opent indexbestanden voor de huidige tabel. Niet van toepassing op Paradox- en SQL-tabellen.

Syntaxis

SET INDEX TO
[<*bestandsnamenlijst*> | ? | <*bestandsnaamfilter*>]
[ORDER [TAG]
<*bestandsnaam1*> | <*labelnaam*> [OF
<*bestandsnaam2*> | ? | <*bestandsnaamfilter*>]]

<*bestandsnamenlijst*> | ? | <*bestandsnaamfilter*>

Geeft aan welke indexbestanden moeten worden geopend (zowel .NDX als .MDX-bestanden). SET INDEX TO ? en SET INDEX TO <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een indexbestand kunt selecteren. Als u een bestand zonder pad opgeeft, wordt het bestand gezocht in de huidige directory en vervolgens in het pad dat is ingesteld met SET PATH.

Als u een bestand zonder extensie opgeeft, wordt voor enkelvoudige indexbestanden de extensie .NDX en voor meervoudige indexbestanden de extensie .MDX gebruikt. Als een .NDX-bestand en een .MDX-bestand met dezelfde naam bestaan, moet u de extensie .NDX opgeven om het .NDX-bestand te openen.

ORDER [TAG] <*bestandsnaam1*> | <*labelnaam*>

Geeft een hoofdindex aan. Dat kan een .NDX-bestand of een labelnaam in een .MDX-indexbestand zijn.

OF <*bestandsnaam2*> | ? | <*bestandsnaamfilter*>

Geeft het .MDX-bestand aan dat <*labelnaam*> bevat. OF ? en OF <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een meervoudig indexbestand kiest. Als u een bestand zonder pad opgeeft, wordt het gezocht in de huidige directory en vervolgens in het pad dat is ingesteld met SET PATH. Als u een bestand zonder extensie opgeeft, wordt de extensie .MDX gebruikt.

Beschrijving

Gebruik SET INDEX om de opgegeven .NDX- en .MDX-bestanden te openen in het huidige werkgebied. Deze bestanden worden bijgewerkt als wijzigingen worden aangebracht in de bijbehorende tabel. Als u een indexbestand opneemt in een indexlijst bij USE...INDEX, heeft dat hetzelfde effect als wanneer u achtereenvolgens de commando's USE en SET INDEX opgeeft.

Als de eerste index die met SET INDEX is geopend, een .NDX-bestand is, wordt die index de hoofdindex, tenzij u een andere hoofdindex opgeeft met de optie ORDER of het commando SET ORDER. Als de eerste index die met SET INDEX is geopend, een .MDX-bestand is en de clausule ORDER is niet gebruikt, is geen hoofdindex gedefinieerd, en verschijnen records in de tabel op volgorde van recordnummer ofwel in de *natuurlijke* volgorde. Een hoofdindex voor de huidige tabel kunt u opgeven met de optie ORDER of het commando SET ORDER.

Voordat de bij SET INDEX opgegeven indexen worden geopend, worden eerst alle geopende indexbestanden gesloten, behalve indexbestanden met labelnamen die voorkomen in het produktie-MDX-bestand. Dat is het indexbestand met dezelfde naam als de huidige tabel. Als u SET INDEX TO zonder indexlijst gebruikt, worden alle geopende .NDX- en .MDX-bestanden in het huidige werkgebied gesloten, behalve het produktie-indexbestand. U kunt ook het commando CLOSE INDEX gebruiken. Als u de tabel sluit, worden alle indexen, inclusief het produktie-MDX-bestand, gesloten.

De volgorde waarin de indexen worden opgegeven bij het commando SET INDEX hoeft niet gelijk te zijn aan de volgorde die door dBASE voor Windows wordt gebruikt. Geopende indexen voor een opgegeven werkgebied hebben deze volgorde:

1 Alle enkelvoudige indexbestanden hebben dezelfde volgorde als in *<bestandsnamenlijst>*.

2 Indexen in het produktie-MDX-bestand hebben de volgorde waarin deze in het .MDX-bestand zijn opgenomen.

3 Indexen in andere .MDX-bestanden die u opgeeft met USE...INDEX of met het commando SET INDEX hebben de volgorde waarin deze in de afzonderlijke .MDX-bestanden zijn opgenomen.

De volgorde van de geopende indexen blijft hetzelfde tot u een andere volgorde opgeeft met USE...INDEX of SET INDEX, of tot u het commando INDEX gebruikt. Gebruik het volgordenummer als argument bij NDX( ), TAG( ) en KEY( ) om een indexbestandsnaam, labelnaam of sleuteluitdrukking op te halen.

Voorbeeld

In het volgende voorbeeld wordt SET INDEX gebruikt om indexlabels te openen en de index in te stellen:

```
SET SAFETY OFF   && voorkomt waarschuwing bij overschrijven
USE Bedrijf EXCLUSIVE
INDEX ON Bedrijf TAG Bedrijf   && TAG Bedrijf in ;
                                  Bedrijf.mdx
INDEX ON CompCode  TAG CompCode
SET INDEX TO Bedrijf ORDER Bedrijf
```

```
BROWSE FIELDS Bedrijf, CompCode;
  TITLE "Geïndexeerd op Bedrijf"
```

In het volgende voorbeeld wordt SET INDEX gebruikt om .NDX-indexbestanden te openen en de index in te stellen:

```
USE Bedrijf EXCLUSIVE
INDEX ON Bedrijf TO Bedrijf    && Bedrijf.ndx
INDEX ON CompCode  TO CompCode && CompCode.ndx
SET INDEX TO CompCode.ndx, Bedrijf.ndx
* Dit zijn .NDX-indexen, geen .MDX-labels
BROWSE FIELDS CompCode, Bedrijf;
  TITLE "Geïndexeerd op CompCode"
```

Zie ook

CLOSE..., INDEX, KEY( ), MDX( ), NDX( ), ORDER( ), REINDEX, SET ORDER, TAG( ), TAGNO( ), TAGCOUNT( ), USE

SET INTENSITY

Omgeving

Bepaalt of variabele- en veldgegevens tijdens invoerbewerkingen in dBASE IV-vensters worden getoond in omgekeerde video (monochrome monitors) of in de kleuren die zijn ingesteld voor speciale tekst met SET COLOR of SET COLOR OF (kleurenmonitors). SET INTENSITY wordt hoofdzakelijk ondersteund vanwege de compatibiliteit met dBASE IV. Gebruik de kenmerken ColorNormal en ColorHighlight om kleuren te definiëren voor Windows-objecten.

Zie Help voor meer informatie over SET INTENSITY.

SET KEY

Toetsenbord- en muisacties

Kent een programma of procedure toe aan een opgegeven toets of toetscombinatie. (Het dBASE IV-commando SET KEY, waarmee de verwerking van records in een tabel wordt beperkt tot de records waarvan de sleutelveldwaarden binnen een opgegeven bereik vallen, heet nu SET KEY TO.)

Syntaxis

SET KEY <*Nuitdr*> | <*Tuitdr*> TO
  [<*programmanaam*> | <*procedurenaam*>]

<*Nuitdr*> | <*Tuitdr*>

De toets waaraan u het programma of de procedure toekent. <*Nuitdr*> is de waarde waarin INKEY( ) resulteert voor de toets of toetscombinatie. <*Tuitdr*> is de naam van een functietoets (*F1* tot en met *F9*, *Shift-F1* tot en met *Shift-F10* of *Ctrl-F1* tot en met *Ctrl-F10*).

<*programmanaam*> | <*procedurenaam*>

Het programma of de procedure die u aan de toets of toetscombinatie wilt toekennen.

Beschrijving

Met SET KEY kunt u een programma of procedure toekennen aan een toets of toetscombinatie. Als de gebruiker op de opgegeven toets of toetscombinatie drukt, wordt de uitvoering van het huidige programma onderbroken en wordt het opgegeven programma of de opgegeven procedure uitgevoerd. Nadat het programma of de procedure is voltooid, wordt het oorspronkelijke programma hervat.

SET KEY is vrijwel gelijk aan ON KEY LABEL. Alleen de syntaxis is verschillend. Als u dezelfde toets programmeert met ON KEY LABEL en met SET KEY, wordt de meest recente toetsdefinitie gebruikt. Zie ON KEY voor meer informatie.

Voorbeeld

In het volgende voorbeeld wordt SET KEY gebruikt om een functietoets, een Alt-toetscombinatie, een ctrl-toetscombinatie en een alfanumerieke toets te definiëren als procedure-aanroepen:

```
SET PROCEDURE TO PROGRAM(1) ADDITIVE
SET KEY "F2" TO Openen      && Roept Openen aan
SET KEY "ALT-F3" TO Sluiten && Roept Sluiten aan
SET KEY "CTRL-F4" TO Nieuw  && Roept Nieuw aan
SET KEY "A" TO Afsluiten    && Roept Afsluiten aan

KEYBOARD "{F2}{ALT-F3}{ctrl-F4}A"
* Procedures worden een voor een aangeroepen

* Waarschuwing: de letter A roept de
* procedure Afsluiten aan en kan niet worden
* gebruikt voor gewone tekst!
* Normale functie van toetsen weer herstellen:
SET KEY "F2" TO
SET KEY "ALT-F3" TO
SET KEY "CTRL-F4" TO
SET KEY "A" TO
PROCEDURE Openen
? "In procedure Openen"
RETURN

PROCEDURE Sluiten
? "In procedure Sluiten"
RETURN

PROCEDURE Afsluiten
? "In procedure Afsluiten"
RETURN

PROCEDURE Nieuw
? "In procedure Nieuw"
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS. ON KEY wordt echter wel in deze beide versies ondersteund.

Zie ook

ON KEY, SET ESCAPE, SET FUNCTION, SET KEY TO

SET KEY TO

Tabelindeling

Beperkt records die worden verwerkt in de huidige of opgegeven tabel tot de records waarvan de sleutelveldwaarde binnen een opgegeven bereik valt.

Syntaxis

```
SET KEY TO
   [<uitdrukkingenlijst1> |
      RANGE <uitdr2> [,] |
         , <uitdr3> |
         <uitdr2>, <uitdr3>
      [LOW <uitdrukkingenlijst2> [,] ]
      [HIGH [,] <uitdrukkingenlijst3>]
   [EXCLUDE]
   [IN <alias>]
```

<uitdr1> | RANGE <uitdr2> [,] | ,<uitdr3> | <uitdr2>, <uitdr3>

Geeft een voorwaarde aan waarop records worden gefilterd. Voor Paradox- en SQL-tabellen kunt u waarden (gescheiden door komma's) opgeven die overeenkomen met enkelvoudige of samengestelde indexsleutelvelden. De volgende tabel geeft een overzicht van de manier waarop SET KEY records in de hoofdindex filtert:

Optie	Beschrijving
<uitdrukkingenlijst1>	Zoekt records waarvan de indexsleutelwaarden overeenkomen met *<uitdrukkingenlijst1>*
RANGE *<uitdr2>* [,]	Zoekt naar records waarvan de indexwaarden groter dan of gelijk zijn aan *<uitdr2>*
RANGE ,*<uitdr3>*	Zoekt naar records waarvan de indexwaarden kleiner dan of gelijk zijn aan *<uitdr3>*
RANGE <uitdr2>, <uitdr3>	Zoekt naar records waarvan de indexwaarden groter dan of gelijk zijn aan *<uitdr2>* en kleiner dan of gelijk zijn aan *<uitdr3>*

LOW <uitdrukkingenlijst2> [,]

Geeft aan dat records waarvan de indexsleutelwaarden kleiner zijn dan de opgegeven ondergrens niet binnen het gewenste bereik vallen.

HIGH [,] <uitdrukkingenlijst2>

Geeft aan dat records waarvan de indexsleutelwaarden groter zijn dan de opgegeven bovengrens niet binnen het gewenste bereik vallen.

EXCLUDE

Geeft aan dat als een bereik is opgegeven, de indexwaarden die overeenkomen met de boven- of ondergrens niet binnen het gewenste bereik vallen.

IN *<alias>*
Geeft een werkgebied aan. U kunt een werkgebiednummer (1 tot en met 255) of -letter (A tot en met J) of een aliasnaam opgeven. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

Beschrijving

Met het commando SET KEY TO beperkt u de records die in de huidige of opgegeven tabel worden verwerkt tot de records waarvan de veldwaarden binnen het opgegeven bereik vallen. Tenzij u het sleutelwoord EXCLUDE opgeeft, vallen de sleutelveldwaarden die overeenkomen met de onder- of bovengrens ook binnen het opgegeven bereik. SET KEY TO zonder argumenten verwijdert het bereik dat eerder met SET KEY TO voor de huidige tabel is ingesteld.

De waarden die zijn opgegeven voor *<uitdr1>*, *<uitdr2>* en *<uitdr3>* moeten overeenkomen met de sleuteluitdrukking van de hoofdindex. Als de indexsleutel bijvoorbeeld UPPER(Naam) is, moet u hoofdletters gebruiken in de bereikuitdrukkingen. Bij het bepalen van overeenkomsten tussen het opgegeven bereik en de sleutelvelduitdrukkingen, worden de regels gehanteerd die zijn ingesteld met SET EXACT. Het bereik wordt van kracht nadat de recordaanwijzer is verplaatst.

U kunt niet SKIP of LOCATE gebruiken om naar een record te gaan dat buiten het ingestelde bereik valt. U kunt de recordaanwijzer echter wel met GO verplaatsen naar een record dat buiten het met SET KEY TO ingestelde bereik valt. Commando's als REPLACE die meerdere records verwerken, kunnen alleen worden uitgevoerd op de records die voldoen aan het met SET KEY TO ingestelde bereik. Als u een bereik opgeeft dat geen records bevat, worden geen records verwerkt.

Als u voor dezelfde tabel zowel een instructie met SET KEY TO als met SET FILTER gebruikt, worden alleen die records verwerkt die binnen het met SET KEY TO opgegeven bereik vallen *en* voldoen aan de met SET FILTER ingestelde voorwaarde.

Voorbeeld

In het volgende voorbeeld wordt SET KEY vier keer gebruikt om records te selecteren in de tabel:

```
USE Bedrijf EXCLUSIVE
INDEX ON Regio TAG Reg

SET KEY TO "NW"
LIST Regio
WAIT "Alleen het noordwesten"

SET KEY TO RANGE ,"BZ"
LIST Regio
WAIT "Tot en met België-zuid"

SET KEY TO RANGE  "NO",
LIST Regio
WAIT "Noordoost en verder"

SET KEY TO RANGE  "NO","NW"
```

```
LIST Regio
WAIT "Van het noordoorsten tot en met het noordwesten"
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

INDEX, KEY( ), MDX( ), NDX( ), TAG( ), SET FILTER

SET LDCHECK — Omgeving

Bepaalt of de identificatie van taalaansturingen wordt gecontroleerd.

Syntaxis

SET LDCHECK ON | off

Standaardinstelling

De standaardinstelling voor SET LDCHECK is ON. U kunt de standaardinstelling wijzigen door de parameter LDCHECK in te stellen in DBASEWIN.INI.

Beschrijving

Met SET LDCHECK kunt u de controle op de compatibiliteit van de taalaansturing in- of uitschakelen. Deze controle is belangrijk als u werkt met dBASE-tabellen die zijn gemaakt met verschillende dBASE-configuraties of verschillende internationale versies van dBASE. U wordt dan gewaarschuwd als er een conflict tussen verschillende taalaansturingen bestaat.

De taalaansturing bepaalt de tekenset en de sorteerregels, dus als u een dBASE-tabel maakt met de ene taalaansturing en het bestand vervolgens gebruikt terwijl een andere taalaansturing is ingesteld, kunnen sommige gegevens onjuist worden weergegeven en krijgt u misschien onjuiste resultaten als u gegevens opvraagt.

Zie Appendix C in *Programmeren* voor meer informatie over taalaansturingen.

Voorbeeld

Zie LDRIVER( ) voor een alternatief voor SET LDCHECK ON.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CHARSET( ), LDRIVER( )

SET LIBRARY

Programma's

Opent een dBASE-programmabestand (.PRG-bestand), zodat alle procedures en door de gebruiker gedefinieerde functies in het bestand beschikbaar zijn.

Syntaxis

SET LIBRARY TO
[<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Het bibliotheekbestand dat moet worden geopend. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, wordt de extensie .PRO gebruikt (een gecompileerd objectbestand). Als geen .PRO-bestand wordt gevonden, wordt gezocht naar een .PRG-bestand (een bronbestand). Als een .PRG-bestand wordt aangetroffen, wordt dat gecompileerd.

Beschrijving

SET LIBRARY vormt een aanvulling op SET PROCEDURE. Met beide commando's kunt u een bestand openen zodat toegang wordt verkregen tot de procedures en door de gebruiker gedefinieerde functies in het bestand. Met SET LIBRARY TO zonder bestandsnaam sluit u een geopend bibliotheekbestand.

Een procedure of door de gebruiker gedefinieerde functie kan alleen worden uitgevoerd als dBASE toegang heeft tot het bestand met de procedure of functie. Als een aanroep naar een procedure of door de gebruiker gedefinieerde functie wordt aangetroffen, wordt de procedure of functie gezocht in een bepaalde volgorde op bepaalde plaatsen. Een van die plaatsen is een bestand dat is geopend met SET LIBRARY. Zie het commando DO voor meer informatie over het zoekpad en de zoekvolgorde.

Zie Hoofdstuk 4 in *Programmeren* voor meer informatie over het werken met een bibliotheekbestand.

Voorbeeld

In het volgende voorbeeld wordt SET LIBRARY gebruikt om het hoofdprogramma de beschikking te geven over aanvullende procedures en functies:

```
** Rpt_Proc.PRG **
PROCEDURE Rpt_Titel
regelteller=1
DEFINE FORM Hoofdform FROM 0,0 TO 20,70
DEFINE TEXT regel01 OF Hoofdform AT  1, 0 ;
   PROPERTY TEXT CENTER("Rapport Database Klanten"),;
```

```
   WIDTH 50
DEFINE TEXT regel02 OF Hoofdform AT  2,19 ;
   PROPERTY TEXT ;
   CENTER("Uitgevoerd op " + DTOC(DATE()),40,"-"),;
   WIDTH 40
DEFINE TEXT rpt_regel OF Hoofdform ;
   AT regelteller + 3,0 ;
   PROPERTY TEXT CENTER("Bedrijf",40),;
   WIDTH 40
DEFINE TEXT rpt_regel2 OF Hoofdform ;
   AT regelteller + 3,40 ;
   PROPERTY TEXT CENTER("Contact",40),;
   WIDTH 40
DEFINE TEXT regel03 OF Hoofdform AT  3,19 ;
   PROPERTY TEXT CENTER(Volg_Rpt(7),40,"-"),;
   WIDTH 40
OPEN FORM Hoofdform

FUNCTION Volg_Rpt
PARAMETERS dagen
Volg_datum = "Datum voor volgend rapport " + ;
   DTOC(DATE() + dagen)
RETURN Volg_datum
```

Het volgende deel van het voorbeeld is het hoofdprogrammabestand:

```
** Hoofd.PRG **
SET PROCEDURE TO Hoofd
SET LIBRARY TO Rpt_Proc
DO HoofdInst
DO Rpt_Titel
SET LIBRARY TO

PROCEDURE HoofdInst
SET TALK OFF
SET ECHO OFF
CLEAR
SET DEVELOPMENT ON
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. De opties ? en *<bestandsnaamfilter>* worden niet ondersteund in dBASE IV.

In dBASE IV wordt eerst gezocht in een bestand dat is geopend met SET PROCEDURE en dan pas in een bestand dat is geopend met SET LIBRARY. In dBASE voor Windows wordt het eerst gezocht in het bestand dat het eerst is geopend.

Zie ook

DO, FUNCTION, PROCEDURE, SET( ), SET PROCEDURE

SET LOCK

Gedeelde gegevens

Bepaalt of wordt geprobeerd een gemeenschappelijke tabel te vergrendelen tijdens de uitvoering van bepaalde commando's die de tabel lezen, maar niet wijzigen.

Syntaxis

SET LOCK ON | off

Standaardinstelling

De standaardinstelling voor SET LOCK is ON. Als u de standaardinstelling wilt wijzigen, moet u de parameter voor LOCK in DBASEWIN.INI instellen. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor LOCK rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Met SET LOCK OFF schakelt u de automatisch bestandsvergrendeling uit voor bepaalde commando's die de tabel alleen lezen. Andere gebruikers kunnen dan gegevens in het bestand wijzigen terwijl u toegang tot het bestand hebt om de gegevens te lezen. U kunt bijvoorbeeld SET LOCK uitschakelen terwijl u AVERAGE gebruikt als u niet verwacht dat andere gebruikers de gegevens in de tabel aanmerkelijk zullen wijzigen. Ook kunt u SET LOCK uitschakelen voordat u een reeks records verwerkt die niet door andere gebruikers wordt gewijzigd.

De volgende commando's vergrendelen tabellen automatisch als SET LOCK is ingeschakeld (ON), maar niet als SET LOCK is uitgeschakeld (OFF):

- AVERAGE
- CALCULATE
- COPY (bronbestand)
- COPY MEMO
- COPY STRUCTURE
- COPY TO ARRAY
- COPY TO...STRUCTURE [EXTENDED] (bronbestand)
- COUNT
- JOIN (beide bronbestanden)
- LABEL FORM
- REPORT FORM
- SORT (bronbestand)
- SUM

- TOTAL (bronbestand)

Onafhankelijk van de instelling voor SET LOCK worden tabellen en records automatisch vergrendeld als commando's worden gebruikt die de gebruiker in staat stellen gegevens te wijzigen.

Voorbeeld

In het volgende programma wordt het gebruik van SET LOCK gedemonstreerd bij de verwerking van uitsluitend leesbare gegevens.

```
USE Bedrijf
SET LOCK OFF    && Records niet vergrendelen tijdens COUNT
COUNT FOR Bedrijf->Type = "VAR"  TO nTeller
CLEAR
? nTeller
SET LOCK ON
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FLOCK( ), RLOCK( )

SET MARGIN

Afdrukken

Stelt de breedte in van de linkerkantlijn (marge) op een afgedrukte pagina.

Syntaxis

SET MARGIN TO <*Nuitdr*>

<Nuitdr>

Het kolomnummer voor de linkermarge. Het geldige bereik loopt van 0 tot en met 254. U kunt een niet-geheel getal opgeven voor <*Nuitdr*> om bij een proportioneel lettertype de uitvoer nauwkeurig te plaatsen.

Standaardinstelling

De standaardinstelling voor SET MARGIN is 0. U kunt de standaardinstelling wijzigen door de parameter voor MARGIN in te stellen in DBASEWIN.INI.

Beschrijving

Gebruik SET MARGIN om de linkermarge voor alle afgedrukte uitvoer in te stellen. De marge die wordt ingesteld met SET MARGIN wordt kolompositie 0 van de printer. SET

MARGIN bepaalt de waarde van de systeemgeheugenvariabele _ploffset maar is niet van invloed op de waarde van de systeemgeheugenvariabele _lmargin.

Met SET MARGIN past u de positie van tekst op de afgedrukte pagina aan aan het soort papier. Als u bijvoorbeeld afdrukt op papier dat voorzien is van een perforatie voor een ordner, hebt u waarschijnlijk een grotere afstand tussen de linkerkant van het papier en de tekst nodig. U kunt SET MARGIN ook gebruiken om de plaatsing van papier in de printer te compenseren. Als het papier bijvoorbeeld niet helemaal in het midden van de printer ligt, kunt u de linkerrand aanpassen om de tekst correct te plaatsen.

Als u uitvoer naar de printer stuurt, wordt elk teken geplaatst binnen het *coördinatenvlak*. Dat is een tweedimensionaal raster. Het coördinatenvlak is opgesplitst in tekencellen waarvan de breedte afhankelijk is van de waarde van _ppitch. Raadpleeg de tabel bij _ppitch voor de mogelijke waarden van deze geheugenvariabele. De hoogte van elke tekencel wordt bepaald door de grootte van het font. Zie Hoofdstuk 16 in *Programmeren* voor meer informatie over het coördinatenvlak.

Als u de instelling voor SET MARGIN wijzigt, wordt bij de berekening van de celbreedte in het coördinatenvlak rekening gehouden met de huidige waarde van _ppitch. Dat gebeurt altijd, of u nu afdrukt met een proportioneel font of met een niet-proportioneel font. Als u ? zonder de optie STYLE gebruikt en alleen gehele getallen opgeeft voor de coördinaten, wordt een niet-proportioneel font gebruikt, en krijgt alle uitvoer hetzelfde uiterlijk als in dBASE IV.

Voorbeeld

In het volgende voorbeeld worden tien cijfers getoond met de standaardmarge. Vervolgens wordt de marge ingesteld op kolom 10 en worden dezelfde tien cijfers nogmaals afgedrukt, ditmaal ingesprongen:

```
SET PRINTER ON
SET MARGIN TO 0   && Standaardinstelling
? "1234567890"
SET MARGIN TO 10
? "1234567890"
? _ploffset
SET MARGIN TO 0   && Oorspronkelijke marge opnieuw instellen
SET PRINTER OFF
CLOSE PRINTER
* De afdruk ziet er zo uit:
* 1234567890
*          1234567890
*                  10
*
* _ploffset wordt bepaald door SET MARGIN
```

Zie ook

_indent, _lmargin, _ploffset, _rmargin

SET MARK

Bepaalt welk scheidingsteken wordt gebruikt in datums.

Syntaxis

SET MARK TO
[<*Tuitdr*>]

<*Tuitdr*>

Het *scheidingsteken voor datums*. U kunt meer dan één teken opgeven voor <*Tuitdr*>, maar alleen het eerste teken wordt gebruikt.

Standaardinstelling

De standaardinstelling voor SET MARK is het scheidingsteken in de huidige datumopmaak die door het systeem wordt gebruikt. De huidige datumopmaak wordt bepaald door de instelling voor SET DATE, in DBASEWIN.INI of van de optie **Internationaal** in het Configuratiescherm van Windows (in die volgorde). Dat wil zeggen, de instellingen bij SET DATE hebben een hogere prioriteit dan die in DBASEWIN.INI, en de instellingen in DBASEWIN.INI hebben een hogere prioriteit dan de instellingen in het Configuratiescherm van Windows.

U kunt het huidige scheidingsteken voor datums wijzigen met de parameter voor MARK in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor MARK rechtstreeks invoeren in DBASEWIN.INI.

Als u in DBASEWIN.INI instellingen opgeeft voor DATE en voor MARK, worden datums getoond met de opmaak die wordt bepaald door de instelling bij DATE, maar wordt het scheidingsteken gebruikt dat is ingesteld bij MARK. Als u dBASE eenmaal hebt gestart en u wijzigt de instelling voor de datumopmaak met SET DATE, wordt elke eerder instelling voor SET MARK geannuleerd, inclusief de instelling die is opgegeven in DBASEWIN.INI.

Beschrijving

Met SET MARK kunt u het scheidingsteken in datums wijzigen. Het standaardscheidingsteken in Nederland is een "-" die met SET DATE ITALIAN wordt ingesteld. De datumopmaak is dan MM-DD-JJ. Als u vervolgens echter de instructie SET MARK TO "." gebruikt, worden datums als volgt weergegeven: MM.DD.JJ. Geeft u daarna weer de instructie SET DATE ITALIAN op, dan wordt de oorspronkelijke opmaak (MM-DD-JJ) hertseld.

Als u SET MARK TO zonder <*Tuitdr*> gebruikt, wordt het oorspronkelijke scheidingsteken voor datums in de huidige datumopmaak opnieuw ingesteld.

Voorbeeld

In het volgende voorbeeld wordt SET MARK gebruikt om het scheidingsteken voor datums te wijzigen tijdens het weergeven van datumgegevens:

```
SET CENTURY OFF
SET DATE ITALIAN
datum = {30-12-94}
? datum  && Geeft 30-12-94 als resultaat
SET CENTURY ON
? datum  && Geeft 30-12-1994 als resultaat
SET MARK TO ","
? datum  && Geeft 30,12,1994 als resultaat
SET MARK TO "/"
? datum  && Geeft 30/12/1994 als resultaat
SET MARK TO ":"
? datum  && Geeft 30:12:1994 als resultaat
SET MARK TO "-"
? datum  && Geeft 30-12-1994 als resultaat
SET DATE AMERICAN
? datum  && Geeft 12/30/1994 als resultaat
SET DATE ITALIAN   && Nederlandse instellingen
SET CENTURY OFF    && herstellen
? datum  && Geeft 30-12-94 als resultaat
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

DATE( ), DMY( ), MDY( ), SET CENTURY, SET DATE

SET MBLOCK

Velden en records

Wijzigt de blokgrootte voor nieuwe memobestanden (.DBT-bestanden).

Syntaxis

SET MBLOCK TO *<Nuitdr>*

<Nuitdr>

Een getal van 1 tot en met 512 dat de grootte bepaalt van blokken in nieuwe memobestanden (.DBT-bestanden). De werkelijke grootte in bytes is het getal dat u opgeeft, vermenigvuldigd met 64.

Standaardinstelling

De standaardinstelling voor SET MBLOCK is 8 (ofwel 512 bytes). U kunt de standaardinstelling wijzigen met de parameter MBLOCK in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor MBLOCK rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Met SET MBLOCK kunt u de grootte wijzigen van de blokken waarin nieuwe memobestanden op schijf worden opgeslagen. U kunt een blokgrootte opgegeven van 64 bytes tot ongeveer 32K. De instelling voor MBLOCK vervangt een eventueel eerder ingestelde blokgrootte die is gedefinieerd met het commando SET BLOCKSIZE of is opgegeven in DBASEWIN.INI. Nadat de blokgrootte is gewijzigd, worden nieuwe .DBT-bestanden gemaakt met de nieuwe blokgrootte. Voor de opslag van de gegevens in elk memoveld worden zoveel blokken gebruikt als nodig zijn.

Nadat de blokgrootte is gewijzigd, hebben memovelden die zijn gemaakt met de commando's COPY, CREATE en MODIFY STRUCTURE, de nieuwe blokgrootte. Als u de blokgrootte van een bestaand memobestand wilt wijzigen, moet u eerst de blokgrootte wijzigen met het commando SET BLOCKSIZE en vervolgens de tabel met het betreffende memoveld naar een nieuw bestand kopiëren. In het nieuwe bestand wordt dan de nieuwe blokgrootte gebruikt.

Als grote blokken worden gebruikt, terwijl de inhoud van de memovelden beperkt is, bevatten memobestanden (.DBT) veel ongebruikte ruimte en worden deze groter dan nodig. Als u denkt dat de inhoud van de memovelden minder dan 512 bytes (de standaardgrootte) in beslag neemt, kunt u een kleinere blokgrootte instellen om ruimte te sparen. Als u verwacht dat memovelden veel informatie zullen bevatten, kunt u de blokgrootte vergroten.

SET MBLOCK komt overeen met het oudere commando SET BLOCKSIZE, maar heeft twee voordelen:

- U kunt verschillende blokgrootten opgeven voor memo- en indexgegevens, terwijl SET BLOCKSIZE voor beide soorten gegevens dezelfde blokgrootte instelt. De blokgrootte voor indexgegevens stelt u in met SET IBLOCK.
- U kunt met SET MBLOCK kleinere blokken instellen dan met SET BLOCKSIZE. SET BLOCKSIZE maakt blokken in stappen van 512 bytes, terwijl SET MBLOCK stappen van 64 bytes toestaat.

Voorbeeld

In het volgende voorbeeld wordt SET MBLOCK gebruikt om een tabel te maken die een kopie is van de tabel Klanten, maar een blokgrootte voor memo's heeft van 256 bytes, tegen 512 bytes in de oorspronkelijke tabel. Deze techniek is goed bruikbaar als de memo's minder dan 256 bytes groot zijn en u de ruimte die door een memobestand in beslag wordt genomen, zoveel mogelijk wilt beperken:

```
USE Klanten
? SET("MBLOCK")                 && Resulteert in de standaard-;
                                   instelling 8. Dat betekent;
                                   dat elk memoblok = 512 bytes
SET MBLOCK TO 4
COPY  TO Klanten2
USE Klanten2
? SET("MBLOCK")                 && Resulteert in 4
LIST FILES LIKE Klanten*.DBT
* U ziet dat Klanten2.DBT kleiner is dan Klanten.DBT
CLOSE DATABASES
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

CREATE, MODIFY STRUCTURE, REPLACE, SET( ), SET BLOCKSIZE, SET IBLOCK

SET MEMOWIDTH

Velden en records

Bepaalt de breedte van memovelden in weergave of uitvoer.

Syntaxis

SET MEMOWIDTH TO
[<*Nuitdr*>]

<Nuitdr>

Een nummer van 8 tot en met 255 dat de breedte van memovelden in weergave of uitvoer aangeeft.

Standaardinstelling

De standaardbreedte voor memo's is 50.

Beschrijving

Met SET MEMOWIDTH kunt u de kolombreedte van memovelden in weergave en uitvoer wijzigen. Memovelden kunnen worden weergegeven met de commando's DISPLAY, LIST, ? en ??. SET MEMOWIDTH is niet van invloed op de weergavebreedte van memovelden in de Tekst-editor. Als de systeemgeheugenvariabele variable_wrap is ingesteld op waar (.T.), bepalen de systeemgeheugenvariabelen _lmargin en _rmargin de breedte van memovelden.

De sjabloonfunctie @V (vertical stretch, verticaal rekken) zorgt dat memovelden worden weergegeven in een verticale kolom als _wrap waar is. Als @V is opgegeven, wordt de systeemgeheugenvariabele _pcolno verhoogd met de waarde van @V. Op die manier kunt u de afgedrukte uitvoer van de commando's ? en ?? wijzigen met de functie @V. Als @V gelijk is aan nul, worden memovelden voorzien van overgangen binnen de breedte die is ingesteld met SET MEMOWIDTH.

Voorbeeld

In het volgende voorbeeld wordt de relatie tussen SET MEMOWIDTH en het resultaat van de functie MEMLINES( ) gedemonstreerd. Het commando REPLACE plaatst een tekenreeks in het memoveld Notities en SET MEMOWIDTH wordt gebruikt om de breedte van het memoveld in uitvoer te wijzigen:

```
SET MEMOWIDTH TO 30
USE KLANTEN
```

```
REPLACE Notities WITH "Dhr. Appel belde ons vandaag, "+;
  "15-04-94, met betrekking tot een aanrijding die hij "+;
  "op 10-04-94 heeft gehad met een vrachtwagen."
CLEAR
? "MEMOWIDTH", "MEMLINES" AT 20
? "  30", LTRIM(STR(MEMLINES(Notities))) AT 22
?
? Notities
?
SET MEMOWIDTH to 45
? "MEMOWIDTH", "MEMLINES" AT 20
? "  45", LTRIM(STR(MEMLINES(Notities))) AT 22
?
? Notities
```

Zie MEMLINES( ) voor een ander voorbeeld met SET MEMOWIDTH.

Zie ook

?, ??, DISPLAY, LIST, MEMLINES( ), MLINE( ), SET( ), _lmargin, _rmargin, _wrap

SET MESSAGE — Omgeving

Toont een melding op de statusbalk. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows kunt u het kenmerk StatusMessage van een object gebruiken om een melding weer te geven op de statusbalk als de focus naar het object wordt verplaatst.

Zie Help voor meer informatie over SET MESSAGE.

SET MOUSE — Toetsenbord- en muisacties

Met SET MOUSE kunt u de muis in- en uitschakelen. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows-applicaties wordt de muis gewoonlijk nooit uitgeschakeld.

Syntaxis

SET MOUSE ON | off

Standaardinstelling

De standaardinstelling voor SET MOUSE is ON.

Beschrijving

Met SET MOUSE OFF kunt u de muisaanwijzer van het scherm verwijderen. Als de muisaanwijzer van het scherm is verwijderd, worden alle muishandelingen genegeerd. Met SET MOUSE ON kunt u de muis weer inschakelen

Zie ook

ISMOUSE( ), MCOL( ), MROW( ), ON MOUSE

SET NEAR

Tabelindeling

Geeft aan waar de recordaanwijzer moet worden geplaatst nadat een bewerking met FIND, SEEK of SEEK( ) geen exacte overeenkomst heeft opgeleverd.

Syntaxis

SET NEAR on | OFF

Standaardinstelling

Standaard is SET NEAR uitgeschakeld (OFF). U kunt de standaardinstelling wijzigen met de parameter NEAR in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor NEAR rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Gebruik SET NEAR om de recordaanwijzer in een geïndexeerde tabel dicht bij een bepaalde sleutelwaarde te plaatsen als een zoekbewerking geen exacte overeenkomst heeft opgeleverd. Als SET NEAR is ingeschakeld (ON), wordt de recordaanwijzer geplaatst bij het record dat het dichtst staat bij de sleuteluitdrukking die is gezocht, maar niet is gevonden met FIND, SEEK of SEEK( ). Als SET NEAR is uitgeschakeld (OFF) en een zoekbewerking levert niets op, wordt de recordaanwijzer aan het eind van het bestand geplaatst.

Als een zoekbewerking naar een teken-, datum, numerieke of zwevende waarde niets oplevert en SET NEAR is ingeschakeld (ON), wordt de recordaanwijzer geplaatst bij het record waarvan de sleutelwaarde volgt op de gezochte waarde (als de index oplopend is). Als SET DELETED is ingeschakeld (ON) of als met het commando SET FILTER een filter is ingesteld, worden verwijderde of uitgefilterde records genegeerd bij het bepalen welk record de sleutelwaarde-uitdrukking het dichtst benaderd.

Als SET NEAR is ingeschakeld (ON), resulteert FOUND( ) in .T. voor een exacte overeenkomst en in .F. voor een niet-exacte overeenkomst. Als SET NEAR is uitgeschakeld (OFF), resulteert FOUND( ) in .F. als geen overeenkomst wordt aangetroffen.

Voorbeeld

In het volgende voorbeeld wordt SET NEAR gebruikt om de plaatsing van de recordaanwijzer te besturen als in een geïndexeerde tabel geen overeenkomend record wordt aangetroffen:

```
USE Bedrijf EXCLUSIVE
INDEX ON Postcode TAG PoCo
```

```
SET NEAR ON
SEEK "3068"
? EOF(),Postcode
SET NEAR OFF
SEEK "3068"
? EOF(),Postcode
```

De postcode 3068 komt niet in de tabel voor. Als SET NEAR is uitgeschakeld (OFF), wordt de recordaanwijzer aan eind van het bestand geplaatst. Als SET NEAR is ingeschakeld (ON), wordt de recordaanwijzer geplaatst bij het dichtstbijzijnde record, ook als dit is gemarkeerd voor verwijdering.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

EOF( ), FIND, FOUND( ), LOCATE, SEEK, SEEK( ), SET( ), SET DELETED, SET FILTER

SET ODOMETER — Omgeving

Bepaalt hoe regelmatig de recordteller wordt bijgewerkt en getoond op de statusbalk als deze is ingeschakeld. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows wordt een voortgangsindicatie getoond tijdens de uitvoering van bepaalde commando's.

Zie Help voor meer informatie over de syntaxis van SET ODOMETER.

SET ORDER — Tabelindeling

Geeft een geopend indexbestand of indexlabel aan als hoofdindex voor een tabel.

Syntaxis

SET ORDER TO [<*indexpositie Nuitdr*>]

of

SET ORDER TO [<*.ndx-bestandsnaam1*>]
  [NOSAVE]

of

SET ORDER TO
  [TAG] <*labelnaam*>
    [OF <*bestandsnaam2*> | ?]
  [NOSAVE]

<indexpositie Nuitdr>

Een nummer dat de positie aangeeft van een index in de lijst met geopende .NDX-bestanden. Deze optie is beschikbaar vanwege de compatibiliteit met dBASE III PLUS en kan niet worden gebruikt als een of meer .MDX-bestanden zijn geopend. Als *<indexpositie Nuitdr>* resulteert in 0, verschijnt de tabel ongeïndexeerd, op volgorde van de recordnummers ofwel in de *natuurlijke volgorde*.

<.ndx-bestandsnaam1>

Geeft de naam aan van een .NDX-bestand dat is gemaakt voor een dBASE-tabel.

[TAG] *<labelnaam>*

Geeft de naam aan van een indexlabel dat is geopend in een .MDX-bestand (of een enkelvoudige of meervoudige veldindex die is gemaakt voor een Paradox- of SQL-tabel). Het sleutelwoord TAG is alleen beschikbaar voor de leesbaarheid. TAG heeft geen gevolgen voor de werking van het commando. Als u voor Paradox-tabellen SET ORDER TO opgeeft zonder een indexlabelnaam op te nemen, wordt de primaire index gebruikt als hoofdindex, als deze bestaat.

OF *<bestandsnaam2>* | ?

Geeft een meervoudig indexbestand aan met de indexlabel die u de volgorde in de huidige tabel wilt laten bepalen. OF ? toont een dialoogvenster, waarin u een indexbestand kunt selecteren. Als u een bestand zonder pad opgeeft, wordt het gezocht in de huidige directory en vervolgens in het pad dat is ingesteld met SET PATH. Als u een bestand zonder extensie opgeeft, gebruikt dBASE de extensie .MDX.

Als u de optie [TAG] *<.ndx-bestandsnaam1>* | *<labelnaam>* gebruikt, maar niet de naam van een .MDX-indexbestand opgeeft, wordt eerst gezocht naar een .NDX-indexbestand, en vervolgens naar de labelnaam in het produktie-MDX-bestand.

NOSAVE

Wordt gebruikt om een tijdelijke index te verwijderen nadat de bijbehorende tabel is gesloten. Als u deze optie hebt gekozen en vervolgens besluit de index toch te bewaren, kunt u, voordat u de tabel sluit, de index opnieuw openen met SET ORDER zonder de optie NOSAVE.

Beschrijving

Met SET ORDER kunt u de hoofdindex van een tabel wijzigen zonder eerst indexen te sluiten en opnieuw te openen. U kunt de hoofdindex kiezen uit de lijst van .NDX-bestanden of .MDX-indexlabels die zijn geopend met SET INDEX of USE...INDEX.

Als u de volgorde opgeeft met *<indexpositie Nuitdr>*, moet u de indexvolgorde gebruiken die is gedefinieerd met het commando SET INDEX. U kunt deze optie alleen gebruiken als geen .MDX-bestanden zijn geopend. Als u SET ORDER gebruikt zonder een index op te geven of als *<indexpositie Nuitdr>* resulteert in 0, verschijnt de tabel ongeïndexeerd, op volgorde van de recordnummers.

In het geval van een Paradox-tabel heeft SET ORDER TO zonder argument tot gevolg dat voor de tabel weer de primaire indexvolgorde wordt ingesteld, als een primaire index bestaat.

Voorbeeld

In het volgende voorbeeld wordt SET ORDER gebruikt om aan te geven welk indexbestand of -label de actieve index is:

```
USE Bedrijf EXCLUSIVE
INDEX ON CompCode TAG CompCode
INDEX ON Bedrijf TAG Bedrijf
INDEX ON Plaats TAG Plaats
SET INDEX TO CompCode, Bedrijf, Plaats
BROWSE FIELDS CompCode, Bedrijf, Plaats ;
  TITLE "Volgorde CompCode"
SET ORDER TO Plaats
BROWSE FIELDS CompCode, Bedrijf, Plaats ;
  TITLE "Volgorde Plaats"
SET ORDER TO Bedrijf
BROWSE FIELDS CompCode, Bedrijf, Plaats ;
  TITLE "Volgorde Bedrijf"
SET ORDER TO         && Natuurlijke volgorde weer instellen
```

Zie ook

CLOSE..., INDEX, KEY( ), MDX( ), NDX( ), ORDER( ), REINDEX, SET INDEX, TAG( ), TAGCOUNT( ), TAGNO( ), USE

SET PATH

Stations- en bestandsfuncties

Geeft aan in welke directory's moet worden gezocht als een bestand niet in de huidige directory wordt aangetroffen.

Syntaxis

SET PATH TO
  [*<padenlijst>*]

<padenlijst>

Een lijst met stations (optioneel) en directory's die het *zoekpad* vormt. Het zoekpad bepaalt waar wordt gezocht naar bestanden. U kunt de directorypaden van elkaar scheiden met komma's, puntkomma's en spaties. De syntaxis voor dit commando komt overeen met die van de DOS-opdracht PATH. Anders dan in DOS kunt u echter ook spaties en komma's gebruiken om de verschillende directorypaden in de lijst van elkaar te scheiden. De DOS-opdracht accepteert alleen puntkomma's.

Standaardinstelling

De standaardinstelling voor SET PATH is leeg (geen zoekpad). U kunt de standaardinstelling wijzigen met de parameter PATH in DBASEWIN.INI. U kunt zoveel geldige stations en directory's toevoegen als u wilt. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor PATH rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Met SET PATH kunt u een zoekpad instellen voor het openen van bestanden in andere directory's dan de huidige directory. Als voor SET PATH niets is ingesteld en u geeft geen volledige padnaam op als u een bestandsnaam opgeeft, wordt het bestand alleen in de huidige directory gezocht.

De volgorde waarin u de directorypaden opneemt in *<padenlijst>*, bepaalt in welke volgorde de directory's worden doorzocht. U kunt SET PATH gebruiken als de bestanden van een applicatie in verschillende directory's staan.

Een padnaam zonder stationsletter of backslash (\) aan het begin, begint in de huidige directory op het huidige station. Gebruik twee punten (..) om de directory boven de huidige directory in de directorystructuur aan te duiden, en gebruik een backslash aan het begin om een pad op te geven dat in de hoofddirectory van het huidige station begint.

SET PATH TO zonder de optie *<padenlijst>* wist het ingestelde zoekpad.

Voorbeeld

In het volgende voorbeeld wordt SET PATH gebruikt, zodat een tabel in het ingestelde zoekpad kan worden geopend. SET FULLPATH wordt gebruikt om de gebruikte subdirectory weer te geven:

```
* Huidige directory is d:\voorbd
SET PATH TO c:\tijd prg, d:\voorbd\pdox
* 'prg' verwijst naar d:\voorbd\prg
USE Regels
SET FULLPATH ON
? DBF()  && D:\VOORBD\PDOX\REGELS.DBF
SET FULLPATH OFF
```

Zie ook

DISPLAY STATUS, LIST STATUS, SET DIRECTORY

SET PCOL — Afdrukken

Stelt de horizontale afdrukpositie van de printer in. Dat is de waarde waarin PCOL( ) resulteert.

Syntaxis

SET PCOL TO *<Nuitdr>*

<Nuitdr>

Het kolomnummer voor de horizontale afdrukpositie. Het geldige bereik loopt van 0 tot en met 32.767.

Beschrijving

Met SET PCOL kunt u de horizontale afdrukpositie van een printer instellen. De waarde die u hier opgeeft, wordt door de functie PCOL( ) als resultaat gegeven. Alle volgende @...SAY-commando's waarin een kolompositie op de huidige afdrukregel wordt opgegeven, worden afgedrukt op een kolompositie relatief ten opzichte van de nieuwe waarde voor PCOL( ). In de meeste gevallen gebruikt u de instructie SET PCOL TO 0 om de afdrukpositie weer in te stellen bij de linkerkant van de pagina.

SET PCOL is niet van invloed op het afdrukken van @...SAY-commando's die relatieve adressering met PCOL( ) gebruiken. De volgende instructie drukt bijvoorbeeld "Hallo" af op tien posities vanaf de huidige kolompositie, ongeacht de instelling voor PCOL( ).

```
@ 1,PCOL( ) + 10 SAY "Hallo"
```

Als u de afdrukpositie naar een nieuwe regel verplaatst, wordt de waarde die PCOL( ) als resultaat geeft, opnieuw op 0 ingesteld. SET PCOL is dus alleen van invloed op de waarde die PCOL( ) als resultaat geeft voor de huidige regel. Als u uitvoer naar de printer stuurt, wordt voor elk teken dat naar de printer wordt gestuurd, de waarde die PCOL( ) als resultaat geeft, verhoogd met 1. De afdrukpositie schuift een kolom naar rechts voor elk teken dat wordt afgedrukt.

Als u een printerbesturingscode of escape-reeks naar de printer stuurt, wordt de afdrukpositie niet opgeschoven. (Printerbesturingscodes en escape-reeksen zijn tekenreeksen die instructies voor de printer bevatten, zoals onderstrepen, vet of een ander lettertype.) Hoewel besturingscodes en escape-reeksen niet van invloed zijn op de afdrukpositie, wordt de waarde waarin PCOL( ) resulteert toch verhoogd met het aantal tekens dat u naar de printer stuurt. Elk besturingsteken verhoogt de waarde waarin PCOL( ) resulteert met 1, net als elk ander teken. Dat brengt met zich mee dat de waarde waarin PCOL( ) resulteert, niet in overeenstemming hoeft te zijn met de werkelijke afdrukpositie. Gebruik SET PCOL om te zorgen dat dit wel het geval is.

U kunt een besturingscode naar de printer sturen zonder de waarde te wijzigen waarin PCOL( ) resulteert, door het resultaat van PCOL( ) op te slaan in een geheugenvariabele, de besturingscode naar de printer te sturen en vervolgens SET PCOL in te stellen met de waarde van de geheugenvariabele.

Voorbeeld

In het volgende voorbeeld wordt "Jan & Jannie" naar de printer geschreven. De functie PCOL( ) wordt gebruikt om drie keer de kolompositie vast te leggen: aan het begin, na "Jan" en na "Jannie":

```
SET TALK OFF
SET PRINTER ON
* ?-commando's worden nu naar de printer verstuurd
```

```
?               && printer naar kol 0 van volgende regel
beginpos=pcol()  && huidige kolom vastleggen
?? "Jan"
eindjanpos=pcol()
?? " & Jannie"
eindjanniepos=pcol()
* Afgedrukt wordt:
* Jan & Jannie
SET PRINTER OFF
CLOSE PRINTER
? beginpos       &&  0,00
? eindjanpos     &&  3,00
? eindjanniepos  && 12,00
* Kolomposities weergeven
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

@...SAY...GET, PCOL( ), PROW( ), SET PROW

SET POINT

Numerieke gegevens

Geeft het decimaalscheidingsteken in getallen aan.

Syntaxis

SET POINT TO
[<*Tuitdr*>]

<*Tuitdr*>

Het teken dat als decimaalscheidingsteken moet worden gebruikt. U kunt meer dan één teken opgeven, maar dBASE gebruikt alleen het eerste teken. Als u een cijfer opgeeft voor <*Tuitdr*> (zoals "3"), wordt een fout als resultaat gegeven.

Standaardinstelling

De standaardinstelling voor SET POINT wordt bepaald door de optie **Internationaal** in het Configuratiescherm van Windows. U kunt de standaardinstelling wijzigen met de parameter POINT in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor POINT rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

SET POINT is van invloed op numerieke invoer en op weergave met commando's als EDIT. SET POINT is ook van invloed op de numerieke weergave bij commando's als DISPLAY MEMORY, STORE, = en het sjabloonteken "." bij PICTURE. In de optie

PICTURE moet u de punt gebruiken, ongeacht de instelling voor SET POINT. Zie Picture in Hoofdstuk 8 voor meer informatie over de optie PICTURE.

SET POINT is alleen van invloed op de weergave van numerieke uitdrukkingen in de dBASE-syntaxis, niet op hun *invoer* in programmacode. In programmacode is alleen de punt geldig als decimaalscheidingsteken. Als u bijvoorbeeld SET POINT TO "," (komma) gebruikt en de volgende instructie geeft, geeft dBASE een fout als resultaat:

```
? MAX(123,4, 123,5)
```

De juiste syntaxis luidt:

```
? MAX(123.4, 123.5)
```

SET POINT TO zonder de optie <Tuitdr> stelt het standaarddecimaalscheidingsteken opnieuw in dat is opgegeven bij de optie **Internationaal** in het Configuratiescherm van Windows.

Voorbeeld

In het volgende voorbeeld wordt SET POINT gebruikt om de weergave van numerieke gegevens te regelen:

```
SET DECIMALS TO 2            && Standaardinstelling
SET CURRENCY LEFT            && Standaardinstelling
SET SEPARATOR TO "."
SET POINT TO ","
SET CURRENCY TO "FR"
? 23445.95 PICTURE "$999,999.99";
                             && Geeft FR23.445,95  als resultaat
? 2345.95 PICTURE "$999,999.99";
                             && Geeft FFR2.345,95 als resultaat
? 345.95 PICTURE "$999,999.99"
                             && Geeft FFFFR345,95 als resultaat
? 345.95 PICTURE "@$999,999.99"
                             && Geeft FR345,95 als resultaat
```

Bij INT( ) vindt u nog een voorbeeld met SET POINT.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

SET DECIMALS, SET SEPARATOR, STORE

SET PRECISION — Numerieke gegevens

Bepaalt het aantal cijfers dat wordt gebruikt bij het vergelijken van getallen.

Syntaxis

SET PRECISION TO
[<*Nuitdr*>]

<Nuitdr>
Het aantal cijfers van 10 tot en met 19.

Standaardinstelling

De standaardinstelling voor SET PRECISION is 16. U kunt de standaardinstelling wijzigen met de parameter PRECISION in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor PRECISION rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Met SET PRECISION kunt u de nauwkeurigheid, of precisie, van numerieke vergelijkingen instellen. U kunt een precisie instellen van 10 tot en met 19 cijfers.

De precisie die u instelt voor numerieke waarden, is van invloed op de manier waarop deze waarden onderling worden vergeleken, maar niet op de manier waarop deze worden berekend of weergegeven. In berekeningen bedraagt de precisie altijd negentien cijfers. Het aantal weergegeven decimalen kunt u instellen met SET DECIMALS.

Voorbeeld

In het volgende voorbeeld wordt gedemonstreerd welke invloed de precisie heeft op berekeningen en vergelijkingen:

```
SET DECIMALS TO 18          && Max instelling
SET PRECISION TO 19         && Max instelling

x = 0.12345678901234567
y = 0.12345678901234568
? x = y            && Resulteert in .F.
? x + y            && Resulteert in 0,246913578024691350

SET PRECISION TO 16
? x = y            && Resulteert nu in .T.
? x + y            && Resulteert nog steeds in 0,246913578024691350
SET DECIMALS TO 8
? x + y            && Resulteert in 0,24691358
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. In dBASE IV heeft SET PRECISION invloed op berekeningen, maar niet op getalsvergelijkingen.

Zie ook
SET DECIMALS

SET PRINTER

Afdrukken

De instelling voor SET PRINTER TO geeft een bestand aan waarheen doorlopende uitvoer (streaming output) wordt gestuurd, of een apparaatcode voor een printer die wordt herkend door Afdrukbeheer van Windows. De instelling On/Off bepaalt of doorlopende uitvoer die in het commandovenster verschijnt, ook naar het met SET PRINTER TO ingestelde bestand of apparaat wordt gestuurd.

Syntaxis
SET PRINTER on | OFF
SET PRINTER TO
[[FILE] <*bestandsnaam*> | ? | <*bestandsnaamfilter*>] |
[<*apparaat*>]

FILE
Dit sleutelwoord is alleen voor de leesbaarheid opgenomen. Als u het weglaat, heeft dit geen invloed op de werking van het commando.

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>
Het tekstbestand waarheen de uitvoer wordt gestuurd in plaats van naar de printer. Standaard wordt aan <*bestandsnaam*> de extensie . PRT toegekend en wordt het bestand opgeslagen in de huidige directory. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u de naam opgeeft van het doelbestand en van de directory waarin het wordt opgeslagen.

<*apparaat*>
De printerpoort waarheen de uitvoer wordt gestuurd. Printers en printerpoorten stelt u in in het Configuratiescherm van Windows.

Standaardinstelling
De standaardinstelling voor SET PRINTER is uitgeschakeld (OFF). U kunt de standaardinstelling wijzigen met de parameter PRINT in de sectie [OnOffCommandSettings] in DBASEWIN.INI. De standaardinstelling voor SET PRINTER TO is de standaardprinter die is ingesteld in het Configuratiescherm van Windows.

Beschrijving
Met SET PRINTER TO kunt u *doorlopende uitvoer* (streaming output) van commando's als ?, ?? en LIST naar een printer of een tekstbestand sturen. SET PRINTER TO zonder optie stuurt deze uitvoer naar de standaardprinter. Zie Hoofdstuk 24 in *Programmeren*

voor meer informatie over doorlopende (streaming) en niet-doorlopende (non-streaming) output.

Met SET PRINTER ON/OFF kunt u de met SET PRINTER TO ingestelde printer in- en uitschakelen.

Als u doorlopende uitvoer naar een bestand wilt sturen in plaats van naar de printer, gebruikt u SET PRINTER TO FILE <*bestandsnaam*>. Als u SET PRINTER TO FILE <*bestandsnaam*> gebruikt, heeft SET PRINTER ON tot gevolg dat doorlopende uitvoer naar het tekstbestand <*bestandsnaam*> wordt gestuurd in plaats van naar de printer. Standaard krijgt het bestand de extensie .PRT.

Als SET PRINTER is uitgeschakeld (OFF), wordt doorlopende uitvoer alleen naar het resultatenpaneel van het commandovenster gestuurd. SET PRINTER moet zijn ingeschakeld (ON) om gegevens uit te voeren naar een tekstbestand tenzij u een commando hebt gegeven met de bijbehorende optie TO PRINTER. Dit wordt gedemonstreerd in het volgende voorbeeld:

```
SET PRINTER OFF
SET PRINTER TO FILE test.prt
TYPE bestand.txt          && Alleen op scherm
TYPE bestand.txt TO PRINT && Naar scherm en TEST.PRT
```

Niet-doorlopende uitvoer kunt u naar de printer sturen met SET DEVICE TO PRINTER. Als u niet-doorlopende uitvoer naar een tekstbestand wilt sturen, gebruikt u SET DEVICE TO FILE.

Voorbeeld

In het volgende voorbeeld wordt SET PRINTER in- en uitgeschakeld:

```
SET PRINTER TO  && Standaardinstelling
SET PRINTER ON
? "Hallo"       && Naar printer
SET PRINTER OFF
? prow(),pcol() && Alleen op scherm
CLOSE PRINTER   && Afdrukken starten
SET PRINTER TO       && Terug naar standaardinstelling
SET PRINTER TO PRN   && DOS-uitvoerapparaat instellen
SET PRINTER TO LPT1
SET PRINTER TO NUL
SET PRINTER TO FILE Test
* Alle doorlopende uitvoer wordt naar Test.prt
* gestuurd, inclusief eventuele besturingscodes
* die voor de printer zijn bestemd.
```

Overdraagbaarheid

In dBASE III PLUS wordt SET PRINT gebruikt.

Zie ook

SET ALTERNATE, SET CONSOLE, SET DEVICE

SET PROCEDURE

Opent een dBASE programmabestand (.PRG), zodat alle procedures en door de gebruiker gedefinieerde functies in het bestand beschikbaar komen.

Syntaxis

SET PROCEDURE TO
[<*bestandsnaam*> | ? | <*bestandsnaamfilter*>]
[ADDITIVE]

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Het procedurebestand dat geopend moet worden. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. Als u een bestand zonder extensie opgeeft, wordt de extensie .PRO gebruikt (een gecompileerd objectbestand). Als geen .PRO-bestand wordt aangetroffen, wordt gezocht naar een .PRG-bestand (een bronbestand). Als een .PRG-bestand wordt gevonden, wordt dat gecompileerd.

ADDITIVE

Opent het procedurebestand zonder procedurebestanden te sluiten die eerder met SET PROCEDURE zijn geopend. SET PROCEDURE TO <*bestandsnaam*> zonder de optie ADDITIVE sluit alle eerder met SET PROCEDURE geopende procedurebestanden.

Beschrijving

Een procedure of door de gebruiker gedefinieerde functie kan alleen worden uitgevoerd als dBASE toegang heeft tot het bestand waarin de procedure of functie is opgeslagen. Als een aanroep naar een procedure of door de gebruiker gedefinieerde functie wordt aangetroffen, wordt die procedure of functie in een bepaalde volgorde gezocht op bepaalde lokaties. Een van die plaatsen is het bestand dat is geopend met SET PROCEDURE. Zie het commando DO voor een uitleg over het zoekpad en de zoekvolgorde.

Als u de procedures en door de gebruiker gedefinieerde functies in een programma beschikbaar wilt stellen aan andere programma's, neemt u in het programma de volgende instructie:

```
SET PROCEDURE TO PROGRAM(1) ADDITIVE
```

Als u SET PROCEDURE TO zonder opties gebruikt, worden alle met SET PROCEDURE geopende procedurebestanden gesloten. Als u alleen bepaalde procedurebestanden wilt sluiten, gebruikt u CLOSE PROCEDURE. Het maximumaantal geopende procedurebestanden is afhankelijk van het beschikbare geheugen.

Zie Hoofdstuk 4 in *Programmeren* voor meer informatie over werken met procedurebestanden.

Voorbeeld

In de volgende programmaregels wordt SET PROCEDURE gebruikt om verschillende procedurebestanden in te stellen en toe te voegen zodat alle programma's toegang hebben tot de procedures in de procedurebestanden. Tenslotte worden alle procedurebestanden weer gesloten:

```
SET PROC TO Geluid
* Alle programma's hebben toegang tot alle
* procedures in Geluid.PRG
SET PROCEDURE TO Afb_bibl ADDITIVE
* Alle programma's hebben nu toegang tot alle
* procedures in Geluid.PRG en Afb_bibl.PRG
SET PROCEDURE TO Afb_bibl
* Alleen toegang tot Afb_bibl
SET PROCEDURE TO PROGRAM(1) ADDITIVE
* Het huidige bestand, ongeacht de naam,
* is nu ook een procedurebestand
SET PROCEDURE TO
* Alle procedurebestanden worden gesloten
* en zijn niet langer beschikbaar
```

In het volgende voorbeeld wordt een invoerformulier gemaakt met drie keuzerondjes waarmee een conversiefactor wordt ingesteld. SET PROCEDURE TO CONVERS.PRG maakt de functie Metrisch in het procedurebestand CONVERS beschikbaar als de gebruiker op het invoervak Antw klikt:

```
** Metrisch conversieprogramma **
SET PROCEDURE TO Convers.PRG
f=NEW Conversform()
f.OPEN()
CLASS Conversform OF FORM
  this.Top=2
  this.Left=2
  this.Width=42
  this.Height=18
  this.Text= "Conversiehulpmiddel"
  DEFINE ENTRYFIELD Hoevlhd OF THIS AT 4,15;
    PROPERTY Value 0, Width 8
  DEFINE TEXT Regel1 OF THIS AT 2,3;
    PROPERTY;
    Text "Geef getal op en kies keuzerondje",;
    Width 43
  DEFINE RADIOBUTTON Inches OF THIS AT 6,5;
    PROPERTY Text "Inches naar centimeters",;
    Width 34, Value .F.
  DEFINE RADIOBUTTON Ponden OF THIS AT 8,5;
    PROPERTY Text "Amerikaanse ponden naar kilo's",;
    Width 34, Value .F.
  DEFINE RADIOBUTTON Graden OF THIS AT 10,5;
    PROPERTY Text "Graden F naar C",;
    Width 34, Value .F.
  DEFINE TEXT Regel2 OF THIS AT 12,5;
    PROPERTY Text "Resultaat:",;
    Width 10
  DEFINE ENTRYFIELD Antw OF THIS AT 12,15;
    PROPERTY Value 0, Width 17,;
```

```
    OnGotFocus Metrisch
  DEFINE PUSHBUTTON Sluiten OF THIS AT 15,15;
    PROPERTY TEXT "Sluiten", Width 14, OnClick {;Form.Close()}
ENDCLASS
```

De volgende conversiefunctie is opgeslagen in het .PRG-bestand CONVERS.PRG dat in het hoofdprogramma beschikbaar is gemaakt met de instructie SET PROCEDURE TO Convers.PRG

```
***Convers.PRG***
FUNCTION Metrisch
DO CASE
   CASE Form.Inches.Value
      Form.Antw.Value = LTRIM(STR(Form.Hoevlhd.Value;
      * 2.54,10,2))+" centimeter"
   CASE Form.Ponden.Value
      Form.Antw.Value = LTRIM(STR(Form.Hoevlhd.Value;
      * .454,10,2))+" kilo"
   CASE Form.Graden.Value
      Form.Antw.Value = LTRIM(STR((Form.Hoevlhd.Value;
      -32)* (5/9),10,2))+" graden C"
ENDCASE
RETURN .T.
```

Overdraagbaarheid

De opties ?, *<bestandsnaamfilter>* en ADDITIVE worden niet ondersteund in dBASE III PLUS en dBASE IV. In dBASE IV wordt altijd eerst gezocht in een bestand dat is geopend met SET PROCEDURE en dan pas in een bestand dat is geopend met SET LIBRARY. In dBASE voor Windows wordt het bestand dat als eerste is geopend, ook als eerste doorzocht.

Zie ook

CLOSE..., COMPILE, DO, FUNCTION, PARAMETERS, PROCEDURE, RETURN, SET( ), SET LIBRARY

SET PROW

Afdrukken

Stelt de verticale afdrukpositie van de printer in. Dit is de waarde waarin PROW( ) resulteert.

Syntaxis

SET PROW TO *<Nuitdr>*

<Nuitdr>

Het rijnummer van de verticale afdrukpositie. Het geldige bereik loopt van 0 tot en met 32.767.

Beschrijving

Met SET PROW kunt u de verticale afdrukpositie van een printer instellen. De waarde die u hier opgeeft, wordt door de functie PROW( ) als resultaat gegeven. Alle volgende @...SAY-commando's waarin een rijpositie wordt opgegeven, worden afgedrukt op een rijpositie relatief ten opzichte van de nieuwe waarde voor PROW( ). In de meeste gevallen zult u de instructie SET PROW TO 0 gebruiken om de afdrukpositie weer in te stellen bij de bovenkant van de pagina.

SET PROW is niet van invloed op het afdrukken van @...SAY-commando's die relatieve adressering met PROW( ) gebruiken. De volgende instructie drukt bijvoorbeeld "Hallo" af op twee rijen onder de huidige rij, ongeacht de instelling voor PROW( ).

```
@ PROW() + 2,0 SAY "Hallo"
```

Voorbeeld

In het volgende voorbeeld wordt SET PROW gebruikt om iets relatief ten opzichte van de huidige regel af te drukken:

```
SET DEVICE TO PRINTER
@ 10,0 say "Tiende rij"
SET PROW TO 0
* oorspronkelijke rij 10 is nu rij 0
@ 2,0 say "Tweede rij ten opzichte van nieuwe instelling"
SET DEVICE TO SCREEN
CLOSE PRINTER
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

@...SAY...GET, PCOL( ), PROW( ), SET PCOL

SET REFRESH — Gedeelde gegevens

Bepaalt hoe vaak het scherm van het werkstation wordt ververst met tabelinformatie van de server.

Syntaxis

SET REFRESH TO *<Nuitdr>*

<Nuitdr>

Een periode in seconden van 0 tot en met 3600 seconden (1 uur).

Standaardinstelling

De standaardinstelling voor SET REFRESH is 0, hetgeen inhoudt dat het scherm niet wordt ververst. U kunt de standaardinstelling wijzigen met de parameter REFRESH in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor REFRESH rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Met SET REFRESH kunt u instellen om de hoeveel tijd het scherm moet worden ververst als u met gemeenschappelijke tabellen werkt op een netwerk. Als u dan BROWSE, EDIT of CHANGE gebruikt om gemeenschappelijke tabellen te wijzigen, wordt het scherm regelmatig ververst zodat u de wijzigingen kunt zien die door andere gebruikers in het netwerk zijn aangebracht in dezelfde tabellen.

Als een andere gebruiker het bestand of het record dat u bekijkt, heeft vergrendeld, wordt het scherm pas ververst nadat de gebruiker de vergrendeling heeft opgeheven.

Voorbeeld

```
USE Werknemr
CONVERT
SET REFRESH TO 5
BROWSE
```

Zie ook

BROWSE, CHANGE, EDIT, FLOCK( ), RLOCK( )

SET RELATION

Tabelindeling

Koppelt twee of meer geopende tabellen door middel van gemeenschappelijke velden of uitdrukkingen.

Syntaxis

```
SET RELATION TO
   [<sleuteluitdrukkingenlijst1> | <Nuitdr1>
      INTO <subtabelalias1>
      [, <sleuteluitdrukkingenlijst2> | <Nuitdr2>
         INTO <subtabelalias2>] ... ]
   [ADDITIVE]]
```

Voor dBASE-tabellen kunt u tevens opties instellen om de verwerking te beperken van records in niet-gekoppelde subtabellen en andere regels voor gegevensintegriteit opgeven:

```
SET RELATION TO
   [<sleuteluitdrukkingenlijst1> | <Nuitdr1>
      INTO <subtabelalias1>
      [CONSTRAIN]
```

```
    [INTEGRITY
        [CASCADE | RESTRICTED] ]
    [, <sleuteluitdrukkingenlijst2> | <Nuitdr2>
        INTO <subtabelalias2>]
        [CONSTRAIN]
        [INTEGRITY
            [CASCADE | RESTRICTED] ] ... ]
[ADDITIVE]
```

<sleuteluitdrukkingenlijst1>

De sleuteluitdrukking of veldenlijst die de huidige tabel en een subtabel gemeenschappelijk hebben en die beide tabellen met elkaar verbindt. Als u de optie INTEGRITY gebruikt, kunt u een sleutelveld opgeven, maar geen uitdrukking. De subtabel moet geïndexeerd zijn op het sleutelveld en die index moet de hoofdindex voor de subtabel zijn. Voor de hoofdtabel moet een index zijn gedefinieerd op hetzelfde sleutelveld, maar dat hoeft niet de hoofdindex voor die tabel te zijn.

<Nuitdr1> INTO *<subtabelalias>*

Uitsluitend voor dBASE-tabellen kunt u *<Nuitdr>* opgeven om een record in een subtabel te koppelen. Als RECNO( ) wordt gebruikt voor *<Nuitdr>*, wordt de huidige tabel aan de subtabel gekoppeld door middel van overeenkomstige recordnummers. In dat geval hoeft de subtabel niet geïndexeerd te zijn.

INTO *<subtabelalias>*

<alias> geeft de subtabel aan die aan de huidige tabel is gekoppeld. U kunt een werkgebiednummer (1 tot en met 255) of -letter (A tot en met J) of een aliasnaam opgeven. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan.

<sleuteluitdrukkingenlijst2> | *<Nuitdr2>* INTO *<alias2>* ...]

Geeft aanvullende relaties aan vanuit de huidige tabel naar andere tabellen.

CONSTRAIN

Beperkt records die worden verwerkt in de subtabel tot die records die overeenkomen met de sleuteluitdrukking in de hoofdtabel.

INTEGRITY [CASCADE | RESTRICTED]

Geeft regels aan die het toevoegen of verwijderen van records in de subtabel besturen. Als INTEGRITY is opgegeven, krijgen sleutelvelden van records die met APPEND, APPEND BLANK, BROWSE, EDIT of INSERT worden toegevoegd aan subtabellen, waarden die overeenkomen met die in de hoofdtabel. Ook verschijnt een dialoogvenster waarin u kunt aangeven of alle gerelateerde subrecords moeten worden verwijderd (een trapsgewijze verwijdering), als u records verwijdert uit de hoofdtabel of de waarde wijzigt van sleutelvelden in de hoofdtabel.

De opties CASCADE en RESTRICTED geven regels aan voor het toevoegen of verwijderen van records in een programma (waarbij het dialoogvenster niet verschijnt). De optie CASCADE bepaalt dat alle subtabelrecords met overeenkomstige sleutelveldwaarden worden verwijderd als u een record verwijderd uit de hoofdtabel of

de sleutelwaarde van een record in de hoofdtabel wijzigt. De optie RESTRICTED voorkomt dat in de hoofdtabel records worden verwijderd of gewijzigd als de subtabel records bevat met overeenkomstige sleutelveldwaarden.

ADDITIVE

Voegt een nieuwe relatie toe. Zonder ADDITIVE worden bestaande relaties gewist voordat een nieuwe relatie wordt ingesteld.

Beschrijving

Met SET RELATION kunt u een koppeling tot stand brengen tussen geopende tabellen op basis van gemeenschappelijke velden of uitdrukkingen. Denk er aan dat u ook automatisch relaties kunt definiëren en tabellen kunt koppelen met de commando's CREATE QUERY/VIEW en MODIFY QUERY/VIEW. De relaties kunnen dan worden opgeslagen in een .QBE-bestand.

Voordat u een relatie instelt, moet u elke tabel openen in een afzonderlijk werkgebied. Als de relatie is ingesteld, wordt de tabel in het huidige werkgebied de hoofdtabel genoemd. Een tabel die aan de hoofdtabel is gekoppeld door middel van een opgegeven sleutel, wordt een *subtabel* genoemd. De subtabel moet geïndexeerd zijn op de velden of uitdrukkingen waarop de tabellen worden gekoppeld en die index moet de actieve hoofdindex voor de subtabel zijn. Ook voor de hoofdtabel moet een index zijn gedefinieerd met een sleuteluitdrukking die overeenkomt met *<sleuteluitdrukkingenlijst1>*, maar dat hoeft niet de hoofdindex voor die tabel te zijn.

Gewoonlijk wordt een relatie tussen tabellen ingesteld via gemeenschappelijke sleutels die worden aangegeven door *<sleuteluitdrukkingenlijst>*. De sleuteluitdrukking kan elke van de hoofdtabel afgeleide uitdrukking zijn die overeenkomt met sleutels in de hoofdindex van de subtabel. De sleutels kunnen bestaan uit een enkel veld of een reeks aaneengeschakelde velden in elke tabel. De velden kunnen verschillende namen hebben in elke tabel, maar moeten van hetzelfde gegevenstype zijn. Voor Paradox- en SQL-tabellen kunt u enkele of samengestelde indexsleutelvelden opgeven.

Uitsluitend voor dBASE-tabellen geldt dat als u tabellen koppelt door een numerieke uitdrukking op te geven, de hoofdtabel altijd met een numerieke uitdrukking (meestal de functie RECNO( )) is gekoppeld aan recordnummers in een subtabel. Record 1 in de hoofdtabel is dan gekoppeld aan record 1 in de subtabel, record 2 in de hoofdtabel aan record 2 in de subtabel, enzovoort.

Als u SET RELATION gebruikt, worden bestaande relaties gewist voordat een nieuwe relatie wordt ingesteld, tenzij u de optie ADDITIVE gebruikt. SET RELATION TO zonder argumenten wist alle bestaande relaties voor de huidige tabel zonder dat nieuwe relaties worden ingesteld.

Voor dBASE-tabellen besturen de opties CONSTRAIN en INTEGRITY de verwerking van gekoppelde records. Deze beide opties vereisen dat de hoofdtabel geen records bevat met dubbele sleutelwaarden. Als een hoofdtabel dubbele records bevat, kunt u de tabel naar een tijdelijke tabel kopiëren, de tijdelijke tabel indexeren met de optie UNIQUE en vervolgens met COPY de tabel terugkopiëren naar de oorspronkelijke tabelnaam. Nadat u een relatie hebt ingesteld met CONSTRAIN of met INTEGRITY, wordt strikt de hand gehouden aan unieke sleutelwaarden in de hoofdtabel.

De optie CONSTRAIN beperkt toegang in de subtabel tot die records waarvan de sleutelwaarden overeenkomen met records in een hoofdtabel. U kunt ook SET KEY TO gebruiken voor het sleutelveld in de subtabel. U kunt echter niet tegelijk SET KEY TO en CONSTRAIN gebruiken. Als SET KEY TO is gebruikt voor de subtabel en u geeft CONSTRAIN op bij SET RELATION, verschijnt de melding "Actieve SET KEY in alias". Als de optie CONSTRAIN is ingesteld en SET KEY TO wordt opgegeven, verschijnt een foutmelding dat CONSTRAIN reeds actief is. Als u aanvullende voorwaarden wilt opgeven voor records in een subtabel, kunt u SET FILTER combineren met de optie CONSTRAIN.

De optie INTEGRITY geeft regels aan die het toevoegen of verwijderen van records in de subtabel regelen. Als INTEGRITY is opgegeven en u voegt records toe met APPEND, APPEND BLANK, BROWSE, EDIT of INSERT, krijgen de sleutelvelden van records die zijn toegevoegd aan subtabellen, waarden die overeenkomen met sleutelvelden in de hoofdtabel. Als u nieuwe records toevoegt aan de hoofdtabel, moeten de waarden in het sleutelveld uniek zijn. Anders verschijnt een foutmelding dat de sleutel reeds bestaat. Als u records wijzigt in een subtabel, kunnen de sleutelvelden in de subtabel alleen worden gelezen.

Uitgangspunt bij de integriteitsregels is dat SET DELETED is ingeschakeld (ON). De instelling SET DELETED OFF in combinatie met de optie INTEGRITY wordt afgeraden, omdat dan ook gekoppelde records verschijnen die zijn gemarkeerd voor verwijdering.

Als u records verwijdert uit de hoofdtabel of de waarde van sleutelvelden in de hoofdtabel wijzigt terwijl de optie INTEGRITY is gebruikt zonder CASCADE of RESTRICTED, verschijnt een dialoogvenster waarin u kunt opgeven of alle gerelateerde subrecords ook moeten worden verwijderd (een zogenaamde trapsgewijze verwijdering).

Als u INTEGRITY met de optie CASCADE gebruikt, worden automatisch alle records in de subtabel verwijderd die overeenkomen met de sleutelwaarde van records die zijn verwijderd uit of gewijzigd in de hoofdtabel. Als een sleutelwaarde van een record in de hoofdtabel is gewijzigd in een waarde die al bestaat, verschijnt een foutmelding dat de sleutel reeds bestaat.

Als u INTEGRITY met de optie RESTRICTED gebruikt, wordt voorkomen dat uit de hoofdtabel records worden verwijderd die zijn gekoppeld aan records in subtabellen. Als u probeert records te verwijderen of de sleutelwaarde van records in de hoofdtabel te wijzigen, terwijl in de subtabel nog records met overeenkomende sleutelwaarden bestaan, verschijnt de foutmelding "In alias komen nog gerelateerde records voor" en wordt het hoofdtabelrecord niet verwijderd of gewijzigd. U kunt records uit een subtabel verwijderen, tenzij die tabel zelf een hoofdtabel is en een integriteitsregel is gedefinieerd die het verwijderen van records voorkomt.

Voor een tabel kunnen meerdere relaties zijn gedefinieerd. Verder kunnen voor dezelfde hoofdtabel meerdere relaties worden ingesteld door de optie ADDITIVE te gebruiken of door meerdere relaties op te geven met dezelfde SET RELATION-instructie. U kunt ook aanvullende relaties instellen vanuit een subtabel, zodat een serie relaties ontstaat. Cyclische relaties zijn niet toegestaan. Dat wil zeggen dat u een foutmelding krijgt als u probeert een relatie te definiëren vanuit een subtabel naar de bijbehorende hoofdtabel.

Als tussen een hoofdtabel en een subtabel een relatie is ingesteld, is die relatie alleen toegankelijk vanuit het werkgebied met de hoofdtabel. Als u vanuit het huidige werkgebied toegang wilt krijgen tot velden in de subtabel, moet u de aliasaanwijzer (->) gebruiken en voor de naam van de velden in de subtabel de aliasnaam opgeven.

Als in een gekoppelde tabel geen overeenkomend record wordt aangetroffen, wordt de recordaanwijzer in de gekoppelde tabel aan het eind van het bestand geplaatst (EOF( ) resulteert in .T.). De instelling voor SET NEAR is niet van invloed op de plaatsing van de recordaanwijzer in subtabellen. Als de optie INTEGRITY is gebruikt en de recordaanwijzer in de hoofdtabel is aan het eind van het bestand geplaatst, verschijnt de foutmelding "Geen overeenkomstig record in hoofdelement".

Als een lijst is ingesteld met SET SKIP, wordt de recordaanwijzer in elke tabel opgeschoven, te beginnen in het laatste werkgebied in de relatiereeks en vervolgens stapsgewijs omhoog naar de hoofdtabel.

Voorbeeld

In het volgende voorbeeld wordt SET RELATION gebruikt om een relatie te maken op basis van het sleutelveld, CompCode, tussen de hoofdtabel, Bedrijf, en de subtabel, Contact:

```
CLOSE ALL
USE Contact EXCLUSIVE
INDEX ON CompCode TAG CompCode
SELECT 2
USE Bedrijf EXCLUSIVE
INDEX ON Bedrijf TAG Bedrijf
SET RELATION TO CompCode INTO Contact
```

Er is nu een relatie ingesteld tussen de contactpersonen en het bedrijf waarvoor zij contactpersonen zijn. Contact, de subtabel, moet geïndexeerd zijn op CompCode. Bedrijf hoeft niet op CompCode te zijn geïndexeerd.

Als u een tweede relatie wilt toevoegen voor Bedrijf, kunt u de clausule ADDITIVE gebruiken. In dit voorbeeld wordt een relatie met de tabel Bedrtype toegevoegd. Bedrtype is een tabel met een code die het veld TYPE uit de tabel Bedrijf (bijvoorbeeld "ISV") en een beschrijving in het veld BESCHR (bijvoorbeeld "Independent Software Vendor") bevat:

```
SELECT SELECT()  && Volgende geopende werkgebied kiezen
USE Bedrtype EXCLUSIVE  && Bevat een beschrijving
                        && van het soort bedrijf
INDEX ON TYPE TAG TYPE
SELECT Bedrijf
SET RELATION TO Type INTO Bedrtype ADDITIVE
```

De tabel Bedrijf is nu tegelijk gekoppeld aan Contact via het veld CompCode en aan Bedrtype via het veld Type.

```
DISPLAY ALL Bedrijf,Contact->Contact,;
  Bedrtype->Beschr
* Alle drie tabellen worden gebruikt
```

Zie ook

CREATE VIEW FROM ENVIRONMENT, JOIN, SELECT, SET FIELDS, SET FILTER, SET SKIP, SET VIEW, SKIP

SET REPROCESS

Gedeelde gegevens

Geeft aan hoeveel keer moet worden geprobeerd een tabel of een of meer records te vergrendelen voordat een fout wordt gegenereerd of .F. als resultaat wordt gegeven.

Syntaxis

SET REPROCESS TO <*Nuitdr*>

<Nuitdr>

Een getal van -1 tot en met 32.000. Dit getal geeft aan hoeveel keer moet worden geprobeerd de vergrendeling aan te brengen.

Standaardinstelling

De standaardinstelling voor SET REPROCESS is 0. Dat betekent dat standaard niet opnieuw wordt geprobeerd de tabel te vergrendelen als de oorspronkelijke poging niet is gelukt. U kunt de standaardinstelling wijzigen met de parameter REPROCESS in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor REPROCESS rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Met SET REPROCESS kunt u aangeven hoeveel pogingen moeten worden ondernomen om een tabel of een of meer records te vergrendelen voordat een fout wordt gegenereerd of .F. als resultaat wordt gegeven. SET REPROCESS is van invloed op RLOCK( ), LOCK( ) en FLOCK( ), en op alle commando's en functies die automatisch een bestand of records vergrendelen.

Als u SET REPROCESS TO 0 gebruikt, wordt u gevraagd of u de vergrendeling opnieuw keer wilt proberen of de bewerking wilt annuleren. Als u annuleert, resulteren RLOCK( ), LOCK( ) en FLOCK( ) in .F.. Een commando dat automatisch probeert een bestand of records te vergrendelen, resulteert dan in een fout.

Als u voor SET REPROCESS een getal opgeeft dat groter is dan 0, wordt het opgegeven aantal pogingen ondernomen om de vergrendeling aan te brengen. U wordt dan niet gevraagd of u de bewerking opnieuw wilt proberen of wilt annuleren.

Als u SET REPROCESS TO -1 opgeeft, wordt u niet gevraagd of u de bewerking opnieuw wilt proberen of wilt annuleren, maar wordt de bewerking telkens opnieuw geprobeerd tot de vergrendeling is aangebracht.

Voorbeeld

Zie FLOCK( ) voor een voorbeeld met SET REPROCESS.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FLOCK( ), ON ERROR, ON NETERROR. RETRY, RLOCK( ), SET LOCK

SET SAFETY

Omgeving

Bepaalt of om bevestiging wordt gevraagd voordat een bestand wordt overschreven of alle records in een tabel worden geleegd met ZAP.

Syntaxis

SET SAFETY ON | off

Standaardinstelling

De standaardinstelling voor SET SAFETY is ON. U kunt de standaardinstelling wijzigen met de parameter SAFETY in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor SAFETY rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Als SET SAFETY is ingeschakeld (ON), wordt om bevestiging gevraagd voordat een bestand wordt overschreven of voordat alle records in een tabel worden geleegd als u ZAP gebruikt. Als u wilt dat het programma de interactie tussen dBASE en de gebruiker bestuurt met betrekking tot het overschrijven van bestanden, moet u SET SAFETY in het programma uitschakelen.

SET SAFETY is van invloed op de volgende commando's:

- Commando's waarbij de optie TO FILE is gebruikt
- COPY
- COPY FILE
- COPY TO...STRUCTURE EXTENDED
- CREATE/MODIFY-commando's
- INDEX
- JOIN

- SAVE
- SET ALTERNATE TO
- SORT
- TOTAL
- UPDATE
- ZAP

Opmerking SET TALK OFF schakelt de waarschuwingen niet uit die met SET SAFETY ON zijn ingesteld.

Voorbeeld

In het volgende voorbeeld worden alle records uit een tabel geleegd. Eerst is SET SAFETY ingeschakeld en vervolgens uitgeschakeld:

```
USE BEDRIJF
COPY TO TIJD
* Tijdelijke tabel maken
USE TIJD EXCLUSIVE
SET SAFETY ON
ZAP
* Het venster Records legen verschijnt met de melding
* "Alle records uit TIJD.DBF verwijderen". De records
* in TIJD.DBF worden pas verwijderd als op OK is geklikt.
SET SAFETY OFF
ZAP
* De records worden onmiddellijk verwijderd
SET SAFETY ON
* Vraag om bevestiging verschijnt weer
USE
DELETE FILE TIJD.DBF
DELETE FILE TIJD.DBT
* Tijdelijke tabel verwijderen
```

Zie ook

SET TALK

SET SEPARATOR — Numerieke gegevens

Geeft het teken aan dat wordt gebruikt om groepen van drie cijfers rechts van het decimaalscheidingsteken te scheiden in de weergave van getallen die groter zijn dan 1000. Dit teken wordt wel de duizendscheider genoemd.

Syntaxis

SET SEPARATOR TO
[<*Tuitdr*>]

<Tuitdr>
Het scheidingsteken dat u wilt gebruiken. U kunt meerdere tekens opgeven, maar dBASE gebruikt alleen het eerste teken. Als u voor *<Tuitdr>* een cijfer opgeeft, wordt een fout als resultaat gegeven.

Standaardinstelling

De standaardinstelling voor SET SEPARATOR wordt bepaald door de optie **Internationaal** in het Configuratiescherm van Windows. U kunt de standaardinstelling wijzigen met de parameter SEPARATOR in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor SEPARATOR rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

SET SEPARATOR is alleen van invloed op het sjabloonteken "," (in een PICTURE-clausule) en de numerieke weergave van aantal bytes in de uitvoer van de commando's DIR, DISPLAY FILES en LIST FILES. Als u bijvoorbeeld SET SEPARATOR TO "." (punt) gebruikt en de volgende instructie geeft, wordt 123456 getoond als 123.456:

```
? 123456 PICTURE "999,999"
```

In de optie PICTURE moet u de komma gebruiken, ongeacht de instelling voor SET SEPARATOR. Zie Picture in Hoofdstuk 8 voor meer informatie over de optie PICTURE.

SET SEPARATOR TO zonder de optie <Tuitdr> stelt het scheidingsteken opnieuw in dat is opgegeven bij de optie **Internationaal** in het Configuratiescherm van Windows.

De instelling van een duizendscheider met SET SEPARATOR is niet van invloed op de waarde van getallen, alleen op de weergave.

Voorbeeld

Zie SET POINT en INT( ) voor voorbeelden met SET SEPARATOR.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

SET POINT

SET SKIP

Tabelindeling

Bepaalt hoe de recordaanwijzer wordt verplaatst door records in gekoppelde tabellen.

Syntaxis

SET SKIP TO
[<*alias1*> [, <*alias2*>] ...]

[<*alias1*>[, <*alias2*>] ...]

De aliastabellen die zijn gedefinieerd in een relatie. U kunt een werkgebiednummer (1 tot en met 255) of -letter (A tot en met J) of een aliasnaam opgeven. De werkgebiedletter of aliasnaam moet tussen aanhalingstekens staan. SET SKIP TO zonder opties annuleert een eventueel bestaande instelling voor SET SKIP.

Beschrijving

U kunt SET SKIP alleen opgeven voor tabellen die zijn gekoppeld met het commando SET RELATION. De commando's SET RELATION en SET SKIP bepalen samen de manier waarop de recordaanwijzer door hoofd- en subtabellen wordt verplaatst.

Gebruik SET SKIP als u een relatie instelt tussen een hoofdtabel met unieke sleutelwaarden en subtabellen met dubbele sleutelwaarden. Dat heet een een-op-meer-relatie. SET SKIP zorgt dat commando's die de recordaanwijzer verplaatsen, de recordaanwijzer eerst naar elk record met een overeenkomstige sleutelwaarde in een subtabel verplaatsen, voordat de recordaanwijzer in de hoofdtabel wordt verplaatst. Als u een serie relaties definieert en SET SKIP gebruikt om van de ene tabel naar de andere te gaan, wordt de recordaanwijzer naar elk record in de laatste subtabel verplaatst voordat de aanwijzer naar de hoofdtabel wordt verplaatst.

Voorbeeld

In het volgende voorbeeld wordt SET RELATION gebruikt om een relatie te definiëren op basis van het sleutelveld Klantnr tussen de hoofdtabel Klanten en de subtabel Orders. Vervolgens wordt SET SKIP op twee verschillende manier gebruikt. Eerst wordt toegang verkregen tot alle orders voor elke klanten vervolgens slechts tot één order per klant:

```
CLOSE DATABASE
USE Orders EXCLUSIVE
INDEX ON Klantnr+Ordernr TAG KlOrders
* Klantnr+Ordernr wordt gebruikt in plaats van
* alleen Ordernr omdat de orders nu op
* volgorde per klant worden geplaatst
SELECT 2
USE Klanten EXCLUSIVE
INDEX ON Naam TAG Naam
SET RELATION TO Klantnr INTO Orders
* de tabellen en relaties zijn nu ingesteld
SET SKIP TO Orders
DISPLAY ALL Klanten->Naam,Orders->Ordernr
* alle orders voor alle klanten worden getoond
SET SKIP TO
DISPLAY ALL Klanten->Naam,Orders->Ordernr
* per klant wordt slechts 1 order getoond
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

SET RELATION, SKIP

SET SPACE

Invoer/uitvoer

Bepaalt of een spatie wordt ingevoegd tussen uitdrukkingen die worden weergegeven of afgedrukt met een enkele instructie met ? of ??.

Syntaxis

SET SPACE ON | off

Standaardinstelling

De standaardinstelling voor SET SPACE is ON. U kunt de standaardinstelling wijzigen met de parameter SPACE in DBASEWIN.INI.

Beschrijving

Gebruik SET SPACE OFF als u wilt dat de commando's ? en ?? een lijst van uitdrukkingen weergeven of afdrukken zonder spaties tussen de uitdrukkingen. Als u wel spaties tussen de uitdrukkingen wilt, gebruikt u SET SPACE ON.

SET SPACE heeft geen gevolgen voor opeenvolgende commando's met ? en ??. Als u bijvoorbeeld twee keer de instructie ?? *<uitdr>* geeft, verschijnt de tweede *<uitdr>* direct naast de eerste, ongeacht de instelling voor SET SPACE. Als SET SPACE echter is ingeschakeld (ON) en u geeft twee uitdrukkingen weer met één enkele instructie ?? *<uitdr>*, *<uitdr>*, wordt tussen de twee uitdrukkingen een spatie ingevoegd.

Voorbeeld

In het volgende voorbeeld worden een voornaam en een achternaam weergegeven. De eerste keer met SET SPACE ON en vervolgens met SET SPACE OFF:

```
Voornaam="Jozef"
Achternaam ="Parket"
SET SPACE ON  && standaardinstelling
? Voornaam,Achternaam
* Jozef Parket
SET SPACE OFF
? Voornaam,Achternaam
* JozefParket
* Geen spatie tussen de twee variabelen
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS, waar altijd een spatie wordt ingevoegd tussen twee uitdrukkingen (alsof SET SPACE is ingeschakeld).

Zie ook

?, ??, LTRIM( ), RTRIM( ), TRIM( )

SET STEP

Foutafhandeling en testen op fouten

SET STEP ON opent de Debugger van dBASE. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows geeft u DEBUG op om de Debugger te openen.

Zie Help voor meer informatie over de syntaxis van SET STEP.

SET TALK

Omgeving

Bepaalt of meldingen op de statusbalk en toekenningen aan geheugenvariabelen in het resultatenpaneel van het commandovenster worden weergegeven.

Syntaxis

SET TALK ON | off

Standaardinstelling

De standaardinstelling voor SET TALK is ON. U kunt de standaardinstelling wijzigen met de parameter TALK in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor TALK rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Als SET TALK is ingeschakeld (ON), wordt de huidige instelling voor SET ODOMETER gebruikt om op de statusbalk een indicatie te tonen als commando's als COUNT en SORT worden uitgevoerd. Tevens worden de resultaten van toekenningen aan geheugenvariabelen (met STORE of =) weergegeven in het resultatenpaneel van het commandovenster.

Afhankelijk van de hoeveelheid geheugen in uw systeem en de hoeveelheid geheugen die nodig is voor bepaalde bewerkingen, kunt u SET TALK OFF uitschakelen om de prestatie van sommige bewerkingen te verbeteren.

Gebruik SET TALK in combinatie met SET ALTERNATE of SET DEVICE om de uitvoer van SET TALK naar een bestand of naar de printer te sturen in plaats van naar het resultatenpaneel van het commandovenster.

Voorbeeld

In het volgende worden het effect gedemonstreerd van SET TALK ON en SET TALK OFF bij het maken van een geheugenvariabele:

```
OudeInstTalk=SET("TALK")
SET TALK ON
Voornaam="Sandra"
Achternaam="Olieveld"
Naam=Achternaam+", "+Voornaam
SET TALK OFF
Voornaam="Tom"
Achternaam="Franssen"
Naam=Achternaam+", "+Voornaam
? Naam
* Naam wordt "Franssen, Tom"
* maar dat wordt niet getoond op de statusbalk
SET TALK &OudeInstTalk
```

Als TALK is uitgeschakeld (OFF), worden de toekenningen aan Voornaam, Achternaam en naam niet weergegeven.

In het volgende voorbeeld wordt de voortgang van Count weergegeven op de statusbalk als SET TALK is ingeschakeld (ON):

```
CLOSE ALL
USE Klanten
SET TALK ON       && Inschakelen
COUNT TO Recs     && Voortgang verschijnt op statusbalk
SET TALK OFF      && Uitschakelen
COUNT TO Recs     && Voortgang wordt niet weergegeven
```

Zie ook

SET ALTERNATE, SET CONSOLE, SET DEVICE, SET ODOMETER, STORE

SET TIME

Datum- en tijdgegevens

Stelt de systeemtijd in.

Syntaxis

SET TIME TO *<Tuitdr>*

<Tuitdr>

De tijd die u opgeeft, moet de volgende opmaak hebben:

- UU
- UU:MM of UU.MM
- UU:MM:SS of UU.MM.SS

Standaardinstelling

De standaardinstelling voor de systeemtijd wordt bepaald door de instelling in het dialoogvenster **Datum & tijd** in het Configuratiescherm van Windows.

Beschrijving

Met SET TIME kunt u de systeemtijd wijzigen. Voor het resultaat van de functie TIME( ) en het tijdstempel van bestanden die u opslaat, wordt de nieuwe tijd gebruikt.

Voorbeeld

Zie SET DATE TO voor een voorbeeld met SET TIME.

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

SET DATE TO, TIME( )

SET TITLE — Tabellen

Schakelt de vraag om een beschrijving voor het catalogusbestand in of uit.

Syntaxis

SET TITLE ON | off

Standaardinstelling

De standaardinstelling voor SET TITLE is ON. U kunt de standaardinstelling wijzigen met de parameter TITLE in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor TITLE rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Als SET CATALOG is ingeschakeld (ON), worden bestanden die worden gemaakt, gebruikt of opgeslagen, automatisch toegevoegd aan de catalogus. Als SET TITLE is ingeschakeld (ON), wordt u gevraagd een titel of beschrijving op te geven voor het bestand dat wordt toegevoegd. Als SET TITLE is uitgeschakeld (OFF), verschijnt die vraag niet.

Als u bestanden maakt vanuit het commandovenster en er is een catalogus geopend, worden die bestanden wel toegevoegd aan de catalogus, maar wordt u niet gevraagd om een beschrijving.

Met commando's als BROWSE, EDIT of REPLACE kunt u een titel invoeren of andere informatie in een catalogusbestand wijzigen. U kunt een catalogusbestand openen en wijzigen in elk werkgebied, maar een catalogus die is geopend met SET CATALOG kan niet worden gewijzigd.

Voorbeeld

In het volgende voorbeeld wordt SET TITLE OFF gebruikt om de vraag om een beschrijving voor het catalogusrecord uit te schakelen als een nieuw gemaakt bestand wordt toegevoegd aan de geopende catalogus:

```
SET CATALOG ON
SET CATALOG TO Leren
SET TITLE OFF
CREATE NwNamen TYPE DBASE FROM Klanten TYPE PARADOX
USE NwNamen TYPE DBASE
BROWSE
```

Zie ook

CATALOG( ), CREATE CATALOG, SELECT( ), SET CATALOG, USE

SET TOPIC

Programmeren in Windows

Geeft aan welk helponderwerp in eerste instantie moet worden getoond.

Syntaxis

SET TOPIC TO [<*Tuitdr*>]

<*Tuitdr*>

Het sleutelwoord voor het Helponderwerp. Als u <*Tuitdr*> achterwege laat, wordt standaard de index van Help getoond.

Standaardinstelling

De standaardinstelling voor SET TOPIC is een lege tekenreeks. U kunt de standaardinstelling wijzigen met de parameter TOPIC in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor TOPIC rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Met SET TOPIC kunt opgeven welk onderwerp moet worden getoond onder de volgende omstandigheden:

- Het commando HELP wordt uitgevoerd.
- De gebruiker drukt op *F1* terwijl de cursor in het commandovenster staat.

U geeft het onderwerp aan door voor <*Tuitdr*> het sleutelwoord voor het onderwerp op te geven. Als u het volledige sleutelwoord opgeeft, wordt het onderwerp onmiddellijk getoond. Als u alleen maar de eerste tekens opgeeft, toont het Helpsysteem het dialoogvenster **Zoeken**, waarin de gebruiker een onderwerp kan kiezen uit een lijst. Het eerste onderwerp waarvan de eerste letters overeenkomen met <*Tuitdr*> wordt geselecteerd weergegeven. Als u bijvoorbeeld SET TOPIC TO "SET A" gebruikt, wordt in eerste instantie het dialoogvenster **Zoeken** van Help getoond waarin het onderwerp SET ALTERNATE is geselecteerd.

Als de gebruiker op *F1* drukt terwijl de cursor niet in het commandovenster staat, is het Helpsysteem contextgevoelig en wordt de instelling voor SET TOPIC TO genegeerd. Als u bijvoorbeeld SET TOPIC TO "SET A" opgeeft, vervolgens BROWSE uitvoert, en op *F1* drukt, verschijnt informatie over het venster **Tabelrecords** en niet het dialoogvenster **Zoeken**.

Voorbeeld

In het volgende voorbeeld is het helponderwerp afhankelijk van een menukeuze:

```
DO CASE
CASE MenuKeuze = 1
    SET TOPIC TO "BROWSE"
    BROWSE
CASE MenuKeuze = 2
    SET TOPIC TO "EDIT"
    EDIT
CASE MenuKeuze = 3
    SET TOPIC TO "SET"
    SetInst         && Eigen procedure voor SET-instellingen;
                       F1 heeft tot gevolg dat dialoogvenster;
                       Zoeken verschijnt met de onderwerpen;
                       die beginnen met SET...
CASE MenuKeuze = 4
    QUIT
ENDCASE
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

HELP, HelpFile, HelpID, SET HELP TO

SET TYPEAHEAD

Toetsenbord- en muisacties

Bepaalt de grootte van de typbuffer. In deze buffer worden toetsaanslagen opgeslagen als dBASE nog bezig is met de verwerking van andere gegevens. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Formulieren in dBASE voor Windows maken geen gebruik van de typbuffer.

Zie Help voor meer informatie over de syntaxis van SET TYPEAHEAD. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het maken van formulieren.

SET UNIQUE

Tabelindeling

Bepaalt of records met dubbele sleutels in een indexbestand worden opgenomen.

Syntaxis

SET UNIQUE on | OFF

Standaardinstelling

De standaardinstelling voor SET UNIQUE is OFF. U kunt de standaardinstelling wijzigen met de parameter UNIQUE in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor UNIQUE rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Gebruik SET UNIQUE ON om alleen het eerste record met een bepaalde sleutelwaarde op te nemen in .MDX- en .NDX-indexen die zijn gemaakt met het commando INDEX. Als SET UNIQUE is uitgeschakeld (OFF), kunnen indexen records met gelijke sleutelwaarden bevatten. Deze records worden in de index gerangschikt op recordnummer. Als u een indexbestand opnieuw samenstelt, worden de instellingen gehandhaafd waarmee het bestand oorspronkelijk is gemaakt.

dBASE voor Windows verwerkt unieke indexen slechts een keer. Vandaar dat een sleutelwaarde die eerder verborgen was, niet onmiddellijk wordt bijgewerkt als deze wordt gewijzigd. Verder geldt dat als u een record toevoegt met een indexsleutel die al in het indexbestand aanwezig is, het nieuwe record niet wordt toegevoegd aan het indexbestand, hoewel het tabelbestand wel wordt bijgewerkt met het nieuwe record. Met REINDEX kunt u nadrukkelijk alle sleutelwaarden in een unieke index bijwerken.

Het commando SET UNIQUE ON gevolgd door het commando INDEX komt overeen met het commando INDEX...UNIQUE.

Voorbeeld

In het volgende voorbeeld wordt SET UNIQUE gebruikt om een index te maken met slechts één record per regio:

```
USE Bedrijf EXCLUSIVE
INDEX ON Regio TAG Reg
SET UNIQUE ON
INDEX ON Regio TAG UniekRegio
SET UNIQUE OFF    && UNIQUE opnieuw instellen
SET ORDER TO TAG Reg
BROWSE FIELDS Regio, Bedrijf ;
```

```
  TITLE "Alle records"
SET ORDER TO TAG UniekRegio
BROWSE FIELDS Regio, Bedrijf  ;
  TITLE "Een record per regio"
COUNT TO aantregs
WAIT "Er bestaan bedrijven in ";
  +ltrim(str(aantregs)) + " regio's"
```

Zie ook

INDEX, REINDEX, SET( ), SET INDEX, SET ORDER, UNIQUE( ), USE

SET VIEW — Tabelindeling

Opent een eerder gedefinieerd query- of weergavebestand.

Syntaxis

SET VIEW TO *<bestandsnaam>* | ? | *<bestandsnaamfilter>*

<bestandsnaam> | ? | *<bestandsnaamfilter>*

Het query- of weergavebestand met de instellingen die de huidige werkomgeving of weergave definiëren. SET VIEW TO ? en SET VIEW TO *<bestandsnaamfilter>* tonen een dialoogvenster waarin u een weergavebestand kunt selecteren. Als u een bestand zonder pad opgeeft, wordt het gezocht in de huidige directory en vervolgens in het pad dat is ingesteld met SET PATH. Als u een bestand zonder bestandsnaamextensie opgeeft, wordt eerst naar een bestand met de extensie .QBE en vervolgens met de extensie .VUE gezocht.

Beschrijving

Met SET VIEW kunt u de werkomgeving wijzigen in een werkomgeving die eerder is gedefinieerd met CREATE QUERY, CREATE VIEW of CREATE VIEW...FROM ENVIRONMENT. De werkomgeving omvat alle geopende tabellen en indexbestanden, alle relaties, de actieve veldenlijst en filtervoorwaarden.

Voorbeeld

Zie CREATE VIEW ... FROM ENVIRONMENT voor een voorbeeld met SET VIEW.

Zie ook

CREATE QUERY, CREATE VIEW, CREATE VIEW...FROM ENVIRONMENT

SET WINDOW OF MEMO

Geeft het venster aan dat moet worden gebruikt tijdens het wijzigen van de inhoud van een memoveld.

Syntaxis

SET WINDOW OF MEMO TO [<*vensternaam*>]

TO <*vensternaam*>

Geeft de naam aan van een eerder gedefinieerd venster dat moet worden gebruikt tijdens het wijzigen van de inhoud van een memoveld.

Beschrijving

Met SET WINDOW OF MEMO kunt u aangeven welk eerder gedefinieerd venster moet worden gebruikt voor het wijzigen van memovelden met commando's als APPEND, BROWSE, CHANGE, EDIT of READ. De clausule WINDOW bij het commando @...SAY...GET heeft een hogere prioriteit dan het venster dat met dit commando is opgegeven.

Als u SET WINDOW OF MEMO TO gebruikt zonder de naam van een venster op te geven, wordt een eerder ingestelde vensternaam gewist.

Voorbeeld

In het volgende voorbeeld wordt SET WINDOW OF MEMO gebruikt om aan te geven welk venster moet worden gebruikt tijdens het wijzigen van een memoveld:

```
DEFINE WINDOW Hoofdvenster FROM  1, 1 TO 15,70
DEFINE WINDOW Memovenster  FROM  6,30 TO 14,68
USE Bedrijf  EXCLUSIVE
ACTIVATE WINDOW Hoofdvenster
SET WINDOW OF MEMO TO Memovenster
@  2, 1 GET Notities OPEN WINDOW Memovenster
READ
CLOSE DATABASES
DEACTIVATE WINDOW Hoofdvenster
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ACTIVATE WINDOW, CLEAR WINDOW, DEFINE WINDOW, MOVE WINDOW

SET WP

Omgeving

Geeft aan welke tekst-editor moet worden gebruikt voor het bekijken en wijzigen van memovelden.

Syntaxis

SET WP TO
[<*Tuitdr*>]

<Tuitdr>

De uitdrukking die u op de DOS-opdrachtregel opgeeft om de editor te starten. Meestal is dat de naam van het uitvoerbare .EXE-bestand van het programma. U kunt ook de naam van een .PIF-bestand opgeven. Als <*Tuitdr*> niet het volledige pad naar het bestand bevat, wordt het bestand in de huidige directory en vervolgens in het DOS-pad gezocht.

Standaardinstelling

De standaardinstelling voor SET EDITOR is de interne Tekst-editor van dBASE voor Windows. U kunt een andere standaard-editor opgeven met de parameter EDITOR in DBASEWIN.INI. U kunt de instelling interactief wijzigen met het commando SET of de parameter voor EDITOR rechtstreeks invoeren in DBASEWIN.INI.

Beschrijving

Met SET WP kunt u een andere dan de standaard-editor opgeven voor het bekijken en wijzigen van memovelden. De opgegeven editor verschijnt als u een memoveld opent om het te bekijken of te wijzigen. Als u SET WP TO zonder <*Tuitdr*> gebruikt, wordt de standaard-editor opnieuw ingesteld.

Met SET EDITOR kunt u opgeven welke editor moet worden gebruikt voor het bekijken en wijzigen van programma- en tekstbestanden.

U kunt bij SET WP een .PIF-bestand opgeven. Dat is een Windows-bestand waarmee u de Windows-omgeving voor een DOS-toepassing of een Windows .EXE-bestand bestuurt. Start de DOS-editor door het .PIF-bestand uit te voeren in plaats van het .EXE-bestand. Als u commando's opgeeft die in een DOS-venster worden uitgevoerd, wordt COMMAND.COM geladen via DBASEWIN.PIF in de _dbwinhome-directory. Met de PIF-editor van Windows kunt u de instellingen in DBASEWIN.PIF wijzigen. Raadpleeg uw Windows handboek voor meer informatie over .PIF-bestanden.

Als de editor al in gebruik is als u een memo opent om te wijzigen, wordt een tweede versie van de editor gestart.

Voorbeeld

In het volgende voorbeeld staan twee instructies. De eerste stelt Word voor Windows in als standaard-editor en tweede stelt de interne editor van dBASE opnieuw in.

```
SET WP TO "C:\WINWORD\Winword.exe"
SET WP TO   && interne editor
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

_dbwinhome, CREATE COMMAND, CREATE FILE, MODIFY COMMAND, MODIFY FILE, SET EDITOR

SET( )

Omgeving

Resulteert in de huidige instelling van een SET-commando of functietoets.

Syntaxis

SET(<*Tuitdr*>)

<Tuitdr>

Een tekenuitdrukking die het SET-commando of de functietoets voorstelt, waarvan u de waarde als resultaat wilt geven.

Beschrijving

Met SET( ) kunt u de instelling opvragen voor een SET-commando of functietoets zodat u de instelling kunt opslaan of wijzigen. U kunt bijvoorbeeld aan het begin van een programma SET( ) gebruiken om de huidige instellingen op te vragen. U kunt die instellingen dan opslaan in geheugenvariabelen, de instellingen wijzigen en aan het eind van het programma de oorspronkelijke instellingen weer herstellen.

Als zowel een SET- als een SET...TO-commando bestaat voor hetzelfde sleutelwoord, geeft SET( ) de instelling van het SET-commando en SETTO( ) die van het SET...TO-commando als resultaat. Als u bijvoorbeeld SET CARRY ON en SET CARRY TO <*veldenlijst*> gebruikt, geeft SET("CARRY") "ON" als resultaat en geeft SETTO("CARRY") de veldenlijst als resultaat.

Als wel een SET...TO-commando, maar geen overeenkomstig SET-commando bestaat, geven SET( ) en SETTO( ) allebei de waarde als resultaat die is ingesteld met SET...TO. SET("BLOCKSIZE") en SETTO("BLOCKSIZE") geven bijvoorbeeld beide dezelfde waarde als resultaat.

Als <*Tuitdr*> de naam van een functietoets is, zoals "F4", geeft SET( ) de instelling voor die toets als resultaat. Als u de waarde van een combinatie van Ctrl en een functietoets wilt opvragen, moet u 10 optellen bij het nummer van de functietoets. Voor een combinatie van Shift en een functietoets moet u 20 optellen bij het nummer van de functietoets. Als u de instelling voor *Ctrl-F4* en *Shift-F4* wilt weten, moet u SET("F14") en SET("F24") gebruiken.

Als een procedurebestand is geopend, geeft SET("PROCEDURE") de naam van het procedurebestand als resultaat. Als meerdere procedurebestanden zijn geopend, geeft SET("PROCEDURE") de naam als resultaat van het bestand dat als eerste is geladen. Wilt u de naam van een ander procedurebestand als resultaat, geeft u als tweede argument het getal dat aangeeft in welke volgorde het werd geladen. Bijvoorbeeld, SET("PROCEDURE",2) geeft de naam van het procedurebestand dat als tweede is geladen. Als geen procedurebestanden zijn geopend, geeft SET("PROCEDURE") een lege tekenreeks ("") als resultaat.

In de meeste gevallen kan het commando dat u opgeeft voor <*Tuitdr*> worden afgekort tot vier letters. Hiervoor gelden dezelfde regels als voor het afkorten van sleutelwoorden. SET("DECI") en SET("DECIMALS") hebben bijvoorbeeld dezelfde betekenis. Voor het argument <*Tuitdr*> wordt geen onderscheid gemaakt tussen hoofdletters en kleine letters.

Voorbeeld

In het volgende voorbeeld ziet u een typische toepassing van SET( ). Met SET( ) worden de huidige kleurinstellingen opgevraagd en opgeslagen. De kleuren worden gewijzigd, er worden enkele taken uitgevoerd (in dit geval wordt alleen het scherm gewist) en vervolgens worden de oorspronkelijke instellingen hersteld:

```
OudeKleuren=SET("ATTRIBUTE")
SET COLOR TO G/B,R/B
CLEAR
SET COLOR TO &OudeKleuren
* Oude kleuren opnieuw instellen
```

In het volgende voorbeeld wordt een groot aantal argumenten van SET( ) getoond, samen met een mogelijk resultaat:

```
? SET("F1")          && help;
? SET("ALTERNATE")   && ON
? SET("ATTRIBUTE")   && RGB+/B,N/W,N/G && N/G,W/B,RGB+/B,B/W,N/W
? SET("AUTO")        && OFF
? SET("BELL")        && ON
? SET("CARRY")       && OFF
? SET("CATALOG")     && OFF
? SET("CENTURY")     && OFF
? SET("CONFIRM")     && OFF
? SET("CONSOLE")     && ON
? SET("COVER")       && OFF
? SET("CUAENTER")    && ON
? SET("CURRENCY")    && LEFT
? SET("DELIMITER")   && OFF
? SET("DELETED")     && OFF
? SET("DESIGN")      && ON
? SET("ECHO")        && OFF
? SET("ENCRYPTION")  && OFF
? SET("ESCAPE")      && ON
? SET("EXACT")       && OFF
? SET("EXCLUSIVE")   && OFF
? SET("FIELDS")      && OFF
? SET("FORMAT")      && ""
? SET("FULLPATH")    && OFF
```

```
? SET("HEADING")    && ON
? SET("HELP")       && ON
? SET("INTENSITY")  && ON
? SET("LIBRARY")    && ""
? SET("LOCK")       && ON
? SET("MARGIN")     &&          0
? SET("MARK")       && -
? SET("MEMOWIDTH")  &&         50
? SET("MESSAGE")    && ""
? SET("NEAR")       && OFF
? SET("ODOMETER")   &&        100
? SET("ORDER")      && ""
? SET("PATH")       && ""
? SET("PCOL")       &&          0
? SET("POINT")      && ,
? SET("PRECISION")  &&         16
? SET("PRINTER")    && OFF
? SET("RELATION")   && ""
? SET("REPROCESS")  &&          0
? SET("SAFETY")     && OFF
? SET("SEPARATOR")  && .
? SET("SKIP")       && ""
? SET("SPACE")      && ON
? SET("STEP")       && OFF
? SET("TALK")       && ON
? SET("TOPIC")      && ""
? SET("TYPE")       &&         50
? SET("UNIQUE")     && OFF
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

DISPLAY STATUS, SET, SETTO( )

SETTO( )

Omgeving

Resulteert in de huidige instelling van een SET...TO-commando of functietoets.

Syntaxis

SETTO(<*Tuitdr*>)

<Tuitdr>

Een tekenuitdrukking die het SET...TO-commando of de functietoets voorstelt, waarvan u de waarde als resultaat wilt geven.

Beschrijving

Met SETTO( ) kunt u de instelling opvragen voor een SET...TO-commando of functietoets zodat u de instelling kunt opslaan of wijzigen. U kunt bijvoorbeeld aan het begin van een programma SETTO( ) gebruiken om de huidige instellingen op te vragen. U kunt die instellingen dan opslaan in geheugenvariabelen, de instellingen wijzigen en aan het eind van het programma de oorspronkelijke instellingen weer herstellen.

Als zowel een SET- als een SET...TO-commando bestaat voor hetzelfde sleutelwoord, resulteert SET( ) in de instelling van het SET-commando en SETTO( ) in die van het SET...TO-commando. Als u bijvoorbeeld SET CARRY ON en SET CARRY TO *<veldenlijst>* gebruikt, geeft SET("CARRY") "ON" als resultaat en geeft SETTO("CARRY") de veldenlijst als resultaat.

SETTO( ) is bijna gelijk aan SET( ). Zie SET( ) voor meer informatie.

Voorbeeld

In het volgende voorbeeld wordt SETTO( ) gebruikt bij het afdrukken van een speciaal rapport naar een bepaalde printer. De huidige instelling voor SET PRINTER TO wordt opgeslagen. Vervolgens wordt SET PRINTER TO LPT2 gebruikt en wordt het rapport afgedrukt. Daarna wordt de oorspronkelijke instelling weer hersteld:

```
OorsprPrnt=SETTO("PRINTER")  && bijvoorbeeld LPT1
SET PRINTER TO LPT2
REPORT FORM Mijnrapport TO PRINTER
SET PRINTER TO &OorsprPrnt
```

In het volgende voorbeeld wordt een groot aantal argumenten van SETTO( ) getoond, samen met een mogelijk resultaat:

```
? SETTO("ALTERNATE")  && C:\DBASEWIN\VOORBD\SETTO().RES
? SETTO("BORDER")     && SINGLE
? SETTO("BELL")       && 512,2
? SETTO("BLOCK")      &&          1
? SETTO("CATALOG")    &&
? SETTO("CURRENCY")   && F
? SETTO("DATE")       && MDY
? SETTO("DECIMALS")   &&          2
? SETTO("DELIMITER")  && ::
? SETTO("DEVICE")     && SCREEN
? SETTO("DIRECTORY")  && C:\DBASEWIN\VOORBD
? SETTO("DISPLAY")    && VGA25
? SETTO("EDITOR")     &&
? SETTO("FILTER")     &&
? SETTO("FIELDS")     &&
? SETTO("FORMAT")     &&
? SETTO("HELP")       && DBASEWIN.HLP
? SETTO("IBLOCK")     &&          1
? SETTO("LIBRARY")    &&
? SETTO("MARGIN")     &&          0
? SETTO("MARK")       && -
? SETTO("MEMOWIDTH")  &&         50
? SETTO("MESSAGE")    &&
? SETTO("ORDER")      &&
? SETTO("PATH")       &&
```

```
? SETTO("PCOL")        &&          0
? SETTO("POINT")       && ,
? SETTO("PRECISION")   &&         16
? SETTO("PROCEDURE")   &&
? SETTO("PROW")        &&          0
? SETTO("PRINTER")     && LPT1
? SETTO("RELATION")    &&
? SETTO("REFRESH")     &&          0
? SETTO("RETRACE")     && OFF
? SETTO("SEPARATOR")   && .
? SETTO("SKIP")        &&
? SETTO("TOPIC")       &&
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DISPLAY STATUS, SET, SET( )

SHELL( )

Omgeving

Hiermee kunt u de interactieve dBASE-omgeving uitschakelen of herstellen als een formulier is geopend. Resulteert in een logische waarde die de vorige toestand van SHELL( ) aangeeft. De interactieve dBASE-omgeving ("shell") bestaat uit de interactieve componenten van de gebruikers-interface zoals Navigator, het commandovenster, het toepassingsmenu, de knoppenbalk en het toepassingsvenster. Standaard is de interactieve dBASE-omgeving ingeschakeld en resulteert SHELL() in .T.

Syntaxis

SHELL([<*Luitdr 1*>])([<*Luitdr 2*>])

<Luitdr 1>

Bepaalt of de dBASE-omgeving moet worden verborgen of getoond. Als de waarde .F. is, wordt de omgeving verborgen.

<Luitdr 2>

Bepaalt of het toepassingsvenster van dBASE zichtbaar blijft als de dBASE-omgeving is uitgeschakeld. Als de uitdrukking resulteert in .T., blijft het toepassingsvenster zichtbaar. Als <*Luitdr 1*> resulteert in .T., is de volledige omgeving ingeschakeld en wordt <*Luitdr 2*> genegeerd. Als de gebruiker een MDI-formulier opent, blijft het toepassingsvenster zichtbaar (het bevat het formulier), ongeacht de waarde van <*Luitdr 2*>.

Beschrijving

Als u in een applicatie werkt met formulieren, kunt u voorkomen dat de gebruikers toegang hebben tot de dBASE-omgeving. Daarmee zorgt u ervoor dat zij alleen de taken uitvoeren waarvoor het programma is bestemd. Als u de dBASE-omgeving uitschakelt, voorkomt u onder meer dat gebruikers tabellen openen met Navigator en commando's opgeven in het commandovenster. Gebruik SHELL(.F.) als u de dBASE-omgeving uit wilt schakelen, een formulier eruit wilt laten zien als een zelfstandige applicatie of onbedoelde wijzigingen in gegevens wilt voorkomen.

Vaak wordt SHELL(.F.) opgenomen in de procedure OnOpen van een formulier. In dat geval wordt de dBASE-omgeving automatisch opnieuw ingeschakeld als alle formulieren in een programma worden gesloten. U kunt de dBASE-omgeving ook expliciet opnieuw inschakelen door SHELL(.T.) op te geven.

In de volgende tabel worden de effecten van de verschillende instellingen van beide parameters van SHELL en van het formulierkenmerk MDI opgesomd.

<Luitdr 1>	<Luitdr 2>	MDI	Resultaat
.F.	Moet .T. zijn	.T.	Navigator en het commandovenster zijn onzichtbaar. Het toepassingsvenster van dBASE is zichtbaar. Voor MDI-vensters is een hoofdvenster vereist, daarom moet <Luitdr 2> .T. zijn als MDI .T. is. Het dBASE-toepassingsmenu en de knoppenbalk zijn zichtbaar, tenzij voor het formulier een eigen menu is gedefinieerd. Als voor het formulier een menu is gedefinieerd, vervangt dit menu het dBASE-toepassingsmenu en de knoppenbalk in het dBASE-toepassingsvenster.
.F.	.F.	.F.	Het toepassingsvenster van dBASE is onzichtbaar. Alleen het formulier verschijnt. In **Taakoverzicht** van Windows vervangt de naam van het formulier **dBASE voor Windows**.
.F.	.T.	.F.	Het toepassingsvenster van dBASE is zichtbaar en het formulier verschijnt in een afzonderlijk venster. Het toepassingsmenu van dBASE en de knoppenbalk zijn zichtbaar, Navigator en het commandovenster niet.
.T.	Genegeerd	.T. of .F.	De dBASE-omgeving is ingeschakeld. Als het formulier wordt geopend met Open(), heeft de gebruiker toegang tot Navigator, het commandovenster, de knoppenbalk en het toepassingsmenu terwijl het formulier is geopend. Als het formulier wordt geopend met ReadModal(), is de dBASE-omgeving wel zichtbaar maar niet toegankelijk totdat het formulier wordt gesloten.

SHELL( ) heeft in een programma hetzelfde effect als het wijzigen van het kenmerk Visible van het object FrameWin van _app, zoals wordt weergegeven in het volgende voorbeeld. Zie _app voor meer informatie.

```
SHELL(.F.) && hetzelfde effect als de volgende regel
_app.FrameWin.Visible = .F.
```

Als u SHELL(.F.) in het commandovenster opgeeft, wordt even naar Windows geschakeld en keert u daarna terug naar dBASE.

Voorbeeld

In het volgende voorbeeld ziet u de code die is gegenereerd in het venster Formulierontwerp voor een formulier waarmee Shell wordt getest. In het formulier is

SHELL(.T.) voor dubbellikken met de linkermuisknop en SHELL(.F.) voor dubbelklikken met de rechtermuisknop.

```
LOCAL f
f = NEW SHELL()
f.Open()

CLASS SHELL OF FORM
   this.OnRightDblClick = {shell(.t.)}
   this.OnLeftDblClick = {shell(.f.)}
   this.EscExit = .T.
   this.mdi = .f.
   * maak mdi=.t. om effect op mdi te zien
   this.Text = "Test Shell .t."
   this.Width =         48.00
   this.Top =          2.00
   this.Left =          2.00
   this.Height =         15.00
   this.Minimize = .F.
   this.Maximize = .F.
ENDCLASS
```

Als het formulier wordt geactiveerd met bijvoorbeeld DO Shell.wfm, verschijnt het formulier op het scherm (met MDI=.F.). Dubbelklik met de linkermuisknop om andere vensters in het dBASE-venster te laten verdwijnen. Dubbelklik met de rechtermuisknop om deze vensters opnieuw te tonen. * Als u voor MDI .T. instelt, kunt u toegang krijgen tot het formulier door op Alt-Tab te drukken waarmee u de actieve vensters in Windows toont.

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

_app, DEFINE, QUIT, SET DESIGN

SHOW MENU

dBASE IV-menu's

Toont een eerder gedefinieerde dBASE IV-menubalk, maar schakelt deze niet in. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van SHOW MENU. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

SHOW OBJECT

Objecten

Werkt de weergave van een object bij op basis van de meeste actuele kenmerkinstellingen.

Syntaxis

SHOW OBJECT *<bevattende-objectverwijzing>.<objectverwijzing>*

<bevattende-objectverwijzing>.<objectverwijzing>

<bevattende-objectverwijzing> is een objectverwijzingsvariabele die verwijst naar het object (meestal een formulier) dat het object bevat. *<objectverwijzing>* is een objectverwijzingsvariabele die naar het object zelf verwijst.

Beschrijving

Met SHOW OBJECT kunt u de weergave van een object op het scherm bijwerken. Als u bijvoorbeeld een bitmap-afbeelding op een knop wijzigt met het kenmerk UpBitMap, kunt u SHOW OBJECT gebruiken om de nieuwe afbeelding weer te geven.

Voorbeeld

In het volgende voorbeeld wordt een formulier gedefinieerd met twee knoppen waarmee de recordaanwijzer wordt verplaatst. Met het kenmerk OnClick wordt een procedure aangeroepen die controleert of de recordaanwijzer aan het begin of aan het eind van de tabel staat. Als dat het geval is (EOF( )=.T. of BOF( )=.T.), wijzigt de procedure de kenmerken van de knoppen en wordt SHOW OBJECT gebruikt om de weergave van de knoppen bij te werken. Met de ELSE-clausule worden de oorspronkelijke kenmerken van de knop opnieuw ingesteld en wordt de recordaanwijzer verplaatst:

```
USE DIEREN
SET PROCEDURE TO PROGRAM(1) ADDITIVE
DEFINE FORM F1FORM;
  PROPERTY Text "Demo van SHOW OBJECT"
DEFINE PUSHBUTTON Opkn1 OF F1FORM AT 9,15;
  PROPERTY Text "Vorige",;
  Width 12, Height 2,;
  OnClick Terug
DEFINE PUSHBUTTON Opkn2 OF F1FORM AT 12,15;
  PROPERTY Text "Volgende",;
  Width 12, Height 2,;
  OnClick Verder
OPEN FORM F1FORM

PROCEDURE Terug
  IF BOF()
    Form.Opkn1.Text = "Begin van tabel"
    Form.Opkn1.Width = 16
    Form.Opkn1.Left = 12
    SHOW OBJECT form.Opkn1
```

```
    ELSE
      Form.Opkn1.Text = "Vorige"
      Form.Opkn1.Width = 10
      Form.Opkn1.Left = 15
      SHOW OBJECT form.Opkn1
      SKIP-1
    ENDIF
  RETURN

  PROCEDURE Verder
    IF EOF()
      Form.Opkn2.Text = "Eind van tabel"
      Form.Opkn2.Width = 16
      Form.Opkn2.Left = 12
      SHOW OBJECT form.Opkn2
    ELSE
      Form.Opkn2.Text = "Volgende"
      Form.Opkn2.Width = 10
      Form.Opkn2.Left = 15
      SHOW OBJECT form.Opkn2
      SKIP
    ENDIF
  RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

UpBitMap

SHOW POPUP

dBASE IV-menu's

Toont een eerder gedefinieerd dBASE IV popup-menu maar schakelt het menu niet in. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE, OPEN FORM en READMODAL( ) voor het maken en activeren van menu's voor formulieren.

Zie Help voor meer informatie over de syntaxis van SHOW POPUP. Zie het Hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

SIGN( )

Numerieke gegevens

Resulteert in een geheel getal dat aangeeft of een opgegeven getal positief, negatief of nul (0) is.

Syntaxis

SIGN(<*Nuitdr*>)

<Nuitdr>

Het numerieke of zwevende getal waarvan u het teken wilt bepalen.

Beschrijving

Met SIGN( ) kunt u bepalen of een numerieke of zwevende uitdrukking positief, negatief of nul (0) is. SIGN( ) geeft 1 als resultaat als het opgegeven getal positief is, -1 als het getal negatief is en 0 als het getal nul is.

SIGN( ) geeft altijd een geheel getal als resultaat, ongeacht de instelling voor SET DECIMALS.

Voorbeeld

In de volgende voorbeelden worden de drie mogelijke waarden van SIGN( ) als resultaat gegeven:

```
x = -1234.45
y = 1234.45
z = 0
? SIGN(x)                    && Geeft -1 als resultaat
? SIGN(y)                    && Geeft 1 als resultaat
? SIGN(z)                    && Geeft 0 als resultaat
```

In het volgende voorbeeld wordt SIGN( ) gebruikt om de records in de tabel Afnemers te tonen op basis van de waarde in het veld OpenBalans:

```
USE Afnemers
? "Bedrijven met een beginBalans van F 00,00"
?
SCAN FOR SIGN(OpenBalans)=0
? Bedrijf, OpenBalans
ENDSCAN
WAIT
?
? "Bedrijven met een beginbalans die groter " + ;
   "is dan F 00,00"
?
SCAN FOR SIGN(OpenBalans)=1
? Bedrijf, OpenBalans
ENDSCAN
CLOSE DATABASES
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ABS( ), MAX( ), MIN( ), SET DECIMALS

SIN( )

Geeft de trigonometrische sinus van een hoek als resultaat.

Syntaxis

SIN(<*Nuitdr*>)

<Nuitdr>

De grootte van de hoek in radialen. Gebruik DTOR( ) om de grootte van een hoek in graden om te zetten in radialen. Om bijvoorbeeld de sinus te bepalen van een hoek van 30 graden, gebruikt u SIN(DTOR(30)).

Beschrijving

SIN( ) berekent de verhouding tussen de overliggende rechthoekszijde en de hypotenusa van een rechthoekige driehoek. SIN( ) geeft een zwevende waarde van -1 tot en met +1 als resultaat. SIN( ) geeft nul als resultaat als <*Nuitdr*> gelijk is aan 0, pi of 2pi radialen.

Het aantal decimalen stelt u in met SET DECIMALS.

De cosecans van een hoek is het complement van de sinus van de hoek. Als u de cosecans van een hoek wilt opvragen, gebruikt u 1/SIN( ).

Voorbeeld

Hier volgen enkele toepassingen van SIN( ):

```
? SIN(PI())       && Geeft 0 als resultaat
? SIN(PI()/2)     && Geeft 1 als resultaat
? SIN(DTOR(30))   && Geeft 0,5 als resultaat
? SIN(-3*PI()/2)  && Geeft 1 als resultaat
? -SIN(3*PI()/2)  && Geeft 1 als resultaat
```

Het volgende programma tekent een sinusgolf. Eerst worden de x- en de y-as getekend en daarna worden de resultaten van SIN( ) uitgezet:

```
* Sinus.prg
SET TALK OFF
CLEAR
* X- en y-as tekenen
@ 11,0 SAY REPLICATE("-", 78)
FOR i = 0 TO 23
        @ i,40 SAY "|"
NEXT
@ 11,40 SAY "."

FOR i = 0 to 79
        y = 11 - 11 * SIN(i * PI()/20)
        @ y,i SAY "."
NEXT
@ 23,0      && Plaatst cursor onder grafiek
SET TALK ON
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ASIN( ), COS( ), DTOR( ), PI( ), RTOD( ), SET DECIMALS, TAN( )

SKIP

Velden en records

Verplaatst de recordaanwijzer in het huidige of een opgegeven werkgebied.

Syntaxis

SKIP
[<*Nuitdr*>]
[IN <*alias*>]

<*Nuitdr*>

Geeft aan met welk aantal records de recordaanwijzer voorwaarts of achterwaarts moet worden verplaatst in de tabel die is geopend in het huidige of een opgegeven werkgebied. Als u voor <*Nuitdr*> een negatieve waarde opgeeft, wordt de recordaanwijzer achterwaarts verplaatst. SKIP zonder <*Nuitdr*> verplaatst de recordaanwijzer een record verder.

IN <*alias*>

Geeft aan in welk werkgebied de recordaanwijzer moet worden verplaatst. U kunt een werkgebiednummer (1 tot en met 255) of -letter (A tot en met J) of een aliasnaam opgeven. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens. SKIP zonder IN <*alias*> verplaatst de recordaanwijzer in het huidige werkgebied.

Beschrijving

Met SKIP kunt u de recordaanwijzer verplaatsen ten opzichte van de huidige positie. In een niet-geïndexeerde tabel wordt de recordaanwijzer het opgegeven aantal records verder of terug verplaatst. In een geïndexeerde tabel slaat de recordaanwijzer het opgegeven aantal records in de indexvolgorde over.

Als u SKIP gebruikt terwijl de recordaanwijzer bij het laatste record in een tabel staat, resulteert EOF( ) in .T. Elke volgende SKIP resulteert in een fout. Op dezelfde manier resulteert BOF( ) in .T. als u de instructie SKIP -1 gebruikt terwijl de recordaanwijzer bij het eerste record in een bestand staat. In dat geval resulteert elke volgende SKIP met een negatief argument in een fout.

Met SKIP IN <*alias*> kunt u de recordaanwijzer verplaatsen in een ander werkgebied dan het huidige werkgebied zonder eerst dat werkgebied te activeren met het commando SELECT.

Voorbeeld

In het volgende voorbeeld wordt SKIP gebruikt om de recordaanwijzer door de records in een tabel te verplaatsen om een lijst van geselecteerde velden te tonen of af te drukken:

```
SET SAFETY OFF
SET TALK OFF
USE Landen EXCLUSIVE
INDEX ON BNP TAG BNP
? CENTER("Landenlijst - Laagste BNP eerst")
?
DO WHILE .NOT. EOF()
    ? Naam AT 2, BNP AT 20, Hoofdstad AT 40
    SKIP
ENDDO
CLOSE ALL
```

Met SKIP in combinatie met een negatief argument kan het omgekeerde van het vorige voorbeeld worden bereikt:

```
USE Landen EXCLUSIVE
INDEX ON Bevolking TAG Pop
? CENTER("Landenlijst - Hoogste bevolkingsaantal eerst")
?
GO BOTTOM
DO WHILE .NOT. BOF()
    ? Naam AT 2, Bevolking AT 20, Hoofdstad AT 40
    SKIP-1
ENDDO
CLOSE ALL
SET TALK ON
SET SAFETY ON
RETURN
```

Zie ook

ALIAS( ), BOF( ), EOF( ), GO, SCAN

SLEEP

Programma's

Pauzeert het programma gedurende een opgegeven periode of tot een opgegeven tijdstip.

Syntaxis

SLEEP <*seconden Nuitdr*> |
UNTIL <*tijd Tuitdr*> [,<*datum Tuitdr*>]

<seconden Nuitdr>

Het aantal seconden gedurende welke het programma moet worden gepauzeerd. De numerieke uitdrukking moet een geheel getal van 1 tot 65.535 opleveren (van 1

seconden tot ongeveer 18 uur). De tijd wordt gerekend vanaf het moment dat het commando SLEEP wordt opgegeven.

UNTIL *<tijd Tuitdr>*

Zorgt dat het programma wordt gepauzeerd tot een opgegeven tijdstip (*<tijd Tuitdr>*) op de huidige dag. Als u tevens *<datum Tuitdr>* opgeeft, pauzeert het programma tot het opgegeven tijdstip op de opgegeven dag. De tijd en datum worden bepaald op basis van de systeemklok en -datum. U kunt de systeemtijd instellen met SET TIME en de systeemdatum met SET DATE TO. Als het tijdstip al voorbij is, heeft SLEEP UNTIL *<tijd Tuitdr>* geen effect.

Het argument *<tijd Tuitdr>* is een tekenuitdrukking die een tijd moet opleveren met de opmaak UU*<scheidingsteken>*MM*<scheidingsteken>*SS (24-uursklok). De gewone opmaak voor *<tijd Tuitdr>* is "UU:MM:SS". Het scheidingsteken is meestal een dubbelepunt, maar elk toetsenbordteken met uitzondering van cijfers is geldig. UU is een getal van 1 of 2 cijfers voor de uren, MM is een getal van 1 of 2 cijfers voor de minuten en SS is een getal van 1 of 2 cijfers voor de seconden.

<datum Tuitdr>

Een optionele datum die aangeeft tot welke dag het programma moet pauzeren. Het argument *<datum Tuitdr>* is een tekenuitdrukking (*geen* datumuitdrukking) die een datum moet opleveren met de volgende opmaak MM*<scheidingsteken>*DD*<scheidingsteken>*JJ als voor SET DATE AMERICAN is ingesteld. (De gewone opmaak is "DD-MM-JJ".) Het scheidingsteken voor *<datum Tuitdr>* is gewoonlijk een streepje, maar elk toetsenbordteken met uitzondering van cijfers is geldig. MM is een getal van 1 of 2 cijfers voor de maanden, DD is een getal van 1 of 2 cijfers voor de dagen en JJ is een getal van 1 of 2 cijfers voor de jaren. Als de datum al voorbij is, heeft SLEEP UNTIL *<tijd Tuitdr>* ,*<datum Tuitdr>* geen effect. Als u een waarde wilt opgeven voor *<datum Tuitdr>*, moet u ook een waarde opgeven voor *<tijd Tuitdr>*.

Beschrijving

Met SLEEP kunt u een programma pauzeren tot *<seconden Nuitdr>* seconden zijn verstreken of tot een opgegeven tijdstip (*<tijd Tuitdr>*). Het opgegeven tijdstip is op dezelfde dag, tenzij u een datum opgeeft met *<datum Tuitdr>*. Als SET ESCAPE is ingeschakeld (ON), kunt u de pauze afbreken met *Esc*.

Opmerking Als SET ESCAPE is uitgeschakeld (OFF), kunt u SLEEP op geen enkele manier onderbreken. U kunt echter nog wel *Ctrl-Esc* of *Alt-Tab* gebruiken om naar een andere toepassing te gaan of dBASE afsluiten met *Alt-F4*.

Hoewel SLEEP een pauze kan genereren vanuit het commandovenster, wordt het commando hoofdzakelijk gebruikt in programma's. U kunt SLEEP bijvoorbeeld gebruiken om een pauze te genereren tussen de weergave van verschillende vensters of om de gebruiker in de gelegenheid te stellen een melding te lezen op het scherm. U kunt ook een pauze inlassen om het programma te onderbreken tot een bepaald tijdstip.

SLEEP is een alternatief voor het genereren van pauzes in een programma met een DO WHILE-lus, een FOR...NEXT-lus of WAIT. SLEEP heeft een nauwkeuriger werkingdan

lussen omdat het commando onafhankelijk is van de snelheid van het systeem. U kunt ook INKEY(<*Nuitdr*>) gebruiken als u wilt dat de gebruiker de pauze kan onderbreken om verder te gaan met de werkzaamheden.

Voorbeeld

In het volgende voorbeeld wordt SLEEP gebruikt om de uitvoering gedurende vijf seconden te onderbreken:

```
SET ESCAPE ON
* Maakt onderbreken van de pauze mogelijk
SLEEP 5
```

In het volgende voorbeeld wordt SLEEP gebruikt om te pauzeren tot 19:30 op dezelfde dag:

```
SET ESCAPE ON
* Maakt onderbreken van de pauze mogelijk
SLEEP UNTIL "19:30:00"
```

In het laatste voorbeeld wordt SLEEP gebruikt om te pauzeren tot 17:30 op 31 mei 1997 (de instelling voor SET DATE is ITALIAN (standaard voor Nederland)):

```
SET ESCAPE ON
* Maakt onderbreken van de pauze mogelijk
SET DATE ITALIAN
SLEEP UNTIL "17:30:00", "31-5-97"
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DO WHILE, FOR...NEXT, INKEY( ), SET DATE TO, SET TIME, WAIT

SORT

Tabelindeling

Kopieert de huidige tabel naar een nieuwe tabel en sorteert de records in de opgegeven volgorde.

Syntaxis

SORT TO <*bestandsnaam*> | ?
[[TYPE] PARADOX | DBASE]
ON <*veld1*> [/A | /D [/C]]
[,<*veld2*> [/A | /D [/C]] ...]
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[ASCENDING | DESCENDING]

<bestandsnaam> | ?

Het nieuwe tabelbestand waarnaar de records uit de huidige tabel moeten worden gekopieerd in de opgegeven volgorde. Standaard wordt voor *<bestandsnaam>* de extensie .DBF gebruikt en wordt het bestand opgeslagen in de huidige directory. De optie ? toont een dialoogvenster waarin u de naam opgeeft van het doelbestand en van de directory waarin dit moet worden opgeslagen.

U kunt ook een tabel maken in een database (gedefinieerd met de IDAPI-configuratieprogramma) door voor de naam van de tabel de naam van de database op te geven (tussen dubbelepunten), zoals *:databasenaam:tabelnaam*. Als de database nog niet open is, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

[TYPE] PARADOX | DBASE

Geeft aan welk type tabel moet worden gemaakt. Het sleutelwoord TYPE is er alleen voor de leesbaarheid; het heeft geen gevolgen voor de werking van het commando.

Als u DBASE opgeeft, wordt een dBASE-tabel gemaakt (de standaardinstelling). Als u geen extensie opgeeft voor *<bestandsnaam>*, wordt de extensie .DBF gebruikt.

Als u PARADOX opgeeft, wordt een Paradox-tabel gemaakt met de extensie .DB.

ON *<veld1>*

Maakt *<veld1>* het eerste veld in *<bestandsnaam>* en sorteert de records in *<bestandsnaam>* op de waarden in *<veld1>*. Dit veld kan van elk type zijn, behalve binair, memo of OLE.

/A

Sorteert de records oplopend (A tot en met Z; 1 tot en met 9; verleden tot toekomst; onwaar en dan waar). Omdat dit de standaardvolgorde is, hoeft u /A alleen maar op te nemen voor de leesbaarheid.

/D

Sorteert de records aflopend.

/C

Maakt geen onderscheid tussen hoofdletters en kleine letters. Als u zowel A als C, of zowel D als C opgeeft, gebruikt u slechts één slash (bijvoorbeeld /AC).

<veld2> [</A | /D> [/C]] ...

Stelt een tweede of volgende sorteersleutel in voor de nieuwe tabel, zodat eerst wordt gesorteerd op *<veld1>*, en vervolgens voor gelijke waarden in *<veld1>*, op veld *<veld2>*. Als een derde veld is opgegeven, worden records met gelijke waarden in *<veld1>* en *<veld2>* gesorteerd op *<veld3>*. U kunt zoveel velden opgeven als er velden zijn in de tabel.

<bereik>

Het aantal records dat uit de huidige tabel moet worden gekopieerd naar *<bestandsnaam>* en vervolgens gesorteerd. RECORD *<n>* geeft een record aan door middel van het recordnummer. NEXT *<n>* geeft *n* records aan, te beginnen bij het huidige record. ALL geeft alle records aan. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR *<voorwaarde1>*
WHILE *<voorwaarde2>*

Bepaalt welke records worden beïnvloed door SORT. FOR beperkt SORT tot records die voldoen aan *<voorwaarde1>*. WHILE begint bij het huidige record en gaat door met elk volgend record tot *<voorwaarde2>* onwaar wordt.

ASCENDING

Sorteert alle velden waarvoor u geen sorteervolgorde hebt opgegeven in de oplopende volgorde. Omdat dit de standaardinstelling is, hoeft u ASCENDING alleen op te nemen voor de leesbaarheid.

DESCENDING

Sorteert alle velden waarvoor u geen sorteervolgorde hebt opgegeven in de aflopende volgorde.

Beschrijving

Het commando SORT maakt een nieuwe tabel waarin de records uit de huidige tabel worden opgenomen op volgorde van de opgegeven sleutelvelden. Als SET DELETED is ingeschakeld (ON), worden records die zijn gemarkeerd voor verwijdering, genegeerd.

Als u SORT gebruikt, wordt een tijdelijk indexbestand gemaakt. Denk er aan dat op schijf voldoende ruimte moet zijn voor dit tijdelijke indexbestand en de nieuwe tabel.

SORT verschilt van INDEX omdat een nieuwe tabel wordt gemaakt in plaats van een nieuwe index voor de oorspronkelijke tabel. Hoewel SORT in de meeste gevallen minder doelmatig is dan het maken van een index voor het organiseren van tabellen, is SORT goed bruikbaar onder de volgende omstandigheden:

- U wilt een verouderde tabel gesorteerd opslaan in een archief
- U wilt een tabel maken die een deelverzameling bevat van een oorspronkelijke tabel
- U wilt een kleine tabel bijhouden die slechts op een manier gesorteerd hoeft te zijn
- U wilt een gesorteerde tabel maken met sequentiële en doorlopende recordnummers

U kunt ook een tabel met gegevens die weinig worden gewijzigd, sorteren op basis van het meest gebruikte indexbestand, want bewerkingen op een tabel die is gesorteerd op volgorde van de index, zijn sneller dan bewerkingen op een ongesorteerde tabel. In een tabel die is gesorteerd op volgorde van de index, staan de sequentiële records naast elkaar op schijf.

De resultaten van sorteerbewerkingen worden beïnvloed door de taalaansturing die is ingesteld. Zie Appendix C in *Programmeren* voor meer informatie over de regels voor de sorteervolgorde in de Nederlandse taalaansturing.

Voorbeeld

In het volgende voorbeeld wordt SORT gebruikt om nieuwe tabellen te maken:

```
USE Afnemers
SET DELETED ON
* Verwijderde records negeren
SORT TO AfnemerC ON Bedrijf /C
USE AfnemerC
BROWSE FIELDS Bedrijf;
  TITLE "Gesorteerd op Bedrijf, geen onderscheid hoofd/kleine letters"
```

AfnemerC is gesorteerd op Bedrijf en er wordt geen onderscheid gemaakt tussen hoofdletters en kleine letters.

```
USE Afnemers
SORT TO AfnemerP ;
  ON Staat , Plaats /D, Bedrijf /DC;
  FOR Zipcode = "9" TYPE PARADOX
* Sorteervolgorde:
*       Staat standaard oplopend
*       Plaats aflopend
*       Bedrijf aflopend, geen onderscheid hoofd/kleine letters
*       Alleen Zipcodes die beginnen met 9
*       Maak een Paradox-tabel
SET DBTYPE TO PARADOX
USE AfnemerP
BROWSE;
  FIELDS Bedrijf, Plaats, Staat, Zipcode ;
  TITLE "Paradoxbestand"
SET DBTYPE TO  && Standaard opnieuw instellen (dBASE)
```

Zie ook

INDEX

SOUNDEX( )

Tekenreeksgegevens

Resulteert in een uit vier tekens bestaande reeks die de soundex ("klinkt als")-code voor een andere tekenreeks voorstelt. Deze code geeft een vergelijkbare klank als de oorspronkelijke tekenreeks aan.

Syntaxis

SOUNDEX(<*Tuitdr*> | <*memoveld*>)

<Tuitdr> | <memoveld>

De tekenreeks of het memoveld waarvoor de code moet worden bepaald. De tekenreeks of het memoveld kan een woord zijn, maar dat hoeft niet, zolang er maar een klank wordt geproduceerd als de reeks wordt uitgesproken. De tekenreeks of het memoveld kan ook uit meerdere woorden met spaties tussen de woorden bestaan.

Beschrijving

SOUNDEX( ) geeft een uit vier tekens bestaande code als resultaat die de fonetische waarde van een tekenuitdrukking of memoveld voorstelt. De code heeft de vorm "letter cijfer cijfer cijfer", waarbij "letter" de eerste letter is in de uitdrukking. Hoe groter de klankovereenkomst tussen twee reeksen, hoe meer de beide codes op elkaar lijken.

Met SOUNDEX( ) kunt u woorden zoeken die gelijk klinken, of een overeenkomstige spelling hebben, zoals "Jansen", "Janssen" en "Jansens". In de Nederlandse taalaansturing leveren deze drie namen allemaal J525 op. U kunt een tabel indexeren op de soundex-waarde van een veld, en vervolgens FIND of SEEK in combinatie met SOUNDEX( ) gebruiken om te zoeken naar namen. Als de gebruiker bijvoorbeeld wil zoeken naar "Jansen", kunt u "Jansen" omzetten in SOUNDEX("Jansen"), en vervolgens naar die code zoeken in een tabel met een hoofdindex die is gebaseerd op SOUNDEX(achternaam). U kunt SOUNDEX( ) ook gebruiken met LOCATE.

SOUNDEX( ) geeft "0000" als resultaat als de tekenuitdrukking of het memoveld een lege reeks is of als het eerste niet-spatieteken geen letter is. SOUNDEX( ) geeft dan nullen als resultaat voor het eerste cijfer en alle volgende tekens, ongeacht of die volgende tekens cijfers of letters zijn.

Als u de soundex-waarden van twee tekenuitdrukkingen of memovelden wilt vergelijken, kunt u DIFFERENCE( ) gebruiken. Als u twee reeksen teken voor teken wilt vergelijken in plaats van op klank, kunt u LIKE( ) gebruiken.

SOUNDEX( ) is afhankelijk van de taalaansturing. Zie Appendix C in *Programmeren* voor meer informatie over taalaansturing.

Als de Nederlandse taalaansturing is ingesteld, bepaalt SOUNDEX( ) als volgt de fonetische waarde van een tekenreeks:

- Voorloopspaties worden genegeerd.
- De letters A, E, I, O, U, Y, H en W worden genegeerd.
- Er wordt geen onderscheid gemaakt tussen hoofdletters en kleine letters.
- Het eerste niet-spatieteken wordt omgezet in een hoofdletter en dat wordt het eerste teken in de soundex-code.
- B, F, P en V worden omgezet naar 1.
- C, G, J, K, Q, S, X en Z worden omgezet naar 2.
- D en T worden omgezet naar 3.
- L wordt omgezet naar 4.
- M en N worden omgezet naar 5.

- R wordt omgezet naar 6.
- Als twee letters naast elkaar dezelfde fonetische waarde hebben, wordt de tweede genegeerd.
- Indien nodig wordt de reeks aangevuld met nullen.
- Indien nodig wordt de reeks ingekort tot een letter en drie cijfers.

Voorbeeld

In het volgende voorbeeld wordt SOUNDEX( ) gebruikt om een codewaarde te bepalen voor de klant van elke tekenreeks:

```
? SOUNDEX("lachen")                && Geeft L250 als resultaat
? SOUNDEX("lagen")                 && Geeft L250 als resultaat
? SOUNDEX("laken")                 && Geeft L250 als resultaat
? SOUNDEX("lachen") = SOUNDEX("lagen") && Geeft .T. als resultaat
? SOUNDEX("Victor")                && Geeft V236 als resultaat
? SOUNDEX("")                      && Geeft 0000 als resultaat
? SOUNDEX("4endelen")              && Geeft 0000 als resultaat
? SOUNDEX("Gou5denregen")          && Geeft G000 als resultaat
? SOUNDEX("Goud5enregen")          && Geeft G300 als resultaat
```

In het volgende voorbeeld wordt SOUNDEX( ) gebruikt bij de weergave van de inhoud van een veld:

```
USE Klanten
? TRIM(Plaats) + " geeft " +;
  SOUNDEX(Plaats) + " als resultaat";
        && Toont "Midsland geeft M324 als resultaat"
CLOSE DATABASES
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. Het argument *<memoveld>* wordt niet ondersteund in dBASE IV.

Zie ook

DIFFERENCE( ), FIND, INDEX, LIKE( ), LOCATE, SEEK, SET EXACT

SPACE( )

Tekenreeksgegevens

Resulteert in een opgegeven aantal spaties.

Syntaxis

SPACE(*<Nuitdr>*)

<Nuitdr>

Het gewenste aantal spaties.

Beschrijving

SPACE( ) resulteert in een tekenreeks die bestaat uit het opgegeven aantal spaties. De ASCII-code voor spatie is 32. Het grootste aantal spaties dat u kunt opgeven, is 32.766, de maximumlengte voor een tekenreeks.

Als u voor <*Nuitdr*> 0 opgeeft, resulteert SPACE( ) in een lege tekenreeks. Als <*Nuitdr*> kleiner is dan 0, wordt een fout als resultaat gegeven.

Als u een reeks wilt maken die uit een ander teken dan spaties bestaat, kunt u REPLICATE( ) gebruiken.

Voorbeeld

In het volgende voorbeeld wordt SPACE( ) gebruikt om tekengeheugenvariabelen van een bepaalde lengte te initialiseren.

```
gBedrijf=SPACE(20)          && gBedrijf bevat 20 spaties
STORE SPACE(20) TO gBedrijf  && gelijk aan vorige instructie
```

In het volgende voorbeeld wordt SPACE( ) gebruikt om extra ruimte in te voegen in uitgevoerde tekst. Het voorbeeld geeft het volgende als resultaat: plaatsnaam (zonder spaties), een komma, een spatie, een regiocode (zonder spaties), vijf spaties en de postcode.

```
USE Afnemers
SCAN
? Bedrijf
? Contact
? Adres
? TRIM(Plaats) + ", " + TRIM(Staat) + SPACE(5);
+ Zipcode
?
ENDSCAN
```

Overdraagbaarheid

In dBASE IV en dBASE III PLUS is het resultaat van SPACE( ) beperkt tot 254 spaties.

Zie ook

ASC( ), CHR( ), REPLICATE( )

SQLERROR( )

Foutafhandeling en testen op fouten

Resulteert in het nummer van de laatste server-fout.

Syntaxis

SQLERROR( )

Beschrijving

Met SQLERROR( ) kunt u het nummer van de laatste server-fout bepalen. Gebruik SQLMESSAGE( ) als u de tekst van de foutmelding wilt opvragen.

In de tabel bij ERROR( ) worden ERROR( ), MESSAGE( ), DBERROR( ), DBMESSAGE( ), SQLERROR( ), SQLMESSAGE( ) en CERROR( ) met elkaar vergeleken.

Zie Help voor een lijst van alle foutmeldingen.

Voorbeeld

In het volgende voorbeeld worden SQLERROR( ) en SQLMESSAGE( ) gebruikt om het nummer en de melding van een SQL-fout door te geven aan een ON ERROR-routine die een MDI-formulier met een foutrapport toont:

```
ON ERROR DO FtBehRout WITH ERROR(), MESSAGE(), ;
  SQLERROR(), SQLMESSAGE(), PROGRAM(), LINENO()
SET DBTYPE TO DBASE
OPEN DATABASE NWBedrijf
SET DATABASE TO NWBedrijf
foutcode = SQLEXEC("SELECT Bedrijf, Plaats FROM ;
  Bedrijf WHERE Regio='NW'", "Regio_NW.DBF")
IF foutcode = 0
  SET DATABASE TO
  USE Regio_NW
  LIST
ENDIF
RETURN

PROCEDURE FtBehRout
PARAMETERS nFoutnr, tFoutmeld, nSQLFoutnr, ;
  tSQLFoutmeld, tProgramma, nRegelnr
DEFINE FORM Attentieform FROM 10,20 TO 20,55;
  PROPERTY Text "Attentie"
DEFINE TEXT Regel1 OF Attentieform AT 2,10 ;
  PROPERTY Text "Er is iets fout gegaan",;
  Width 35, ColorNormal "R+/W"
DEFINE TEXT Regel2 OF Attentieform AT 4,2;
  PROPERTY Text ;
  IIF(ERROR()=240,tSQLFoutmeld,tFoutmeld),;
  Width 35
DEFINE TEXT Regel3 OF Attentieform AT 5,2;
  PROPERTY Text "Nummer: " + ;
  IIF(ERROR()=240,STR(nSQLFoutnr),STR(nFoutnr)),;
  Width 35
DEFINE TEXT Regel4 OF Attentieform AT 6,2;
  PROPERTY Text "Programma: "+ tProgramma,;
  Width 35
DEFINE TEXT Regel5 OF Attentieform AT 7,2;
  PROPERTY Text "Regelnummer: " + STR(nRegelnr),;
  Width 35
OPEN FORM Attentieform
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CERROR( ), DBERROR( ), DBMESSAGE( ), ERROR( ), MESSAGE( ), ON ERROR, RETRY, SQLMESSAGE( )

SQLEXEC( ) Tabellen

Voert een SQL-instructie uit in de huidige database of op opgegeven dBASE- of Paradox-tabellen.

Syntaxis

SQLEXEC(<*SQL-opdracht Tuitdr*> [,<*Antwoordtabel Tuitdr*>])

<SQL-opdracht Tuitdr>

Een tekenreeks die een SQL-instructie bevat. De instructie moet zich houden aan de regels voor het server-dialect voor de huidige database en moet tussen aanhalingstekens staan. Voor Paradox- en dBASE-tabellen is dat dialect gelijk aan het dialect dat wordt gebruikt door de Borland InterBase database-server, die de ANSI-regels volgt. Tekenreeksen en gereserveerde SQL- of IDAPI-woorden binnen de SQL-instructie moeten ook tussen enkele of dubbele aanhalingstekens staan. (Gewoonlijk worden enkele aanhalingstekens gebruikt.)

<Antwoordtabel Tuitdr>

Paradox- of dBASE-tabel waarin gegevens worden opgeslagen die het resultaat zijn van een SQL SELECT-instructie. Ook deze naam moet tussen aanhalingstekens staan. Als u een bestand zonder pad opgeeft, wordt het in de huidige directory gemaakt. Als u een bestand zonder extensie opgeeft, wordt de standaardextensie gebruikt voor het tabeltype dat is opgegeven met het commando SET DBTYPE. Als u geen tabelnaam opgeeft, wordt een tabel gemaakt met de naam Answer met een extensie op basis van de instelling voor SET DBTYPE.

U kunt ook voor de naam van de antwoordtabel tussen dubbelepunten de naam van een reeds geopende database (alleen gedefinieerd voor een bestandsdirectory als lokatie) opgeven, zoals *:databasenaam:tabelnaam*. U kunt voor de antwoordtabel geen lokatie op een database-server opgeven.

Beschrijving

SQLEXEC( ) voert een SQL-instructie uit op de huidige database die is ingesteld met SET DATABASE, of, als geen database is ingesteld, op tabellen in de huidige of een opgegeven directory. (U kunt voor de naam van een tabel de directory opgeven of een reeds geopende database opgeven door de databasenaam tussen dubbelepunten te plaatsen, zoals *:databasenaam:tabelnaam*.) Als u Borland SQL Link gebruikt om een

verbinding met een database-server tot stand te brengen, wordt de opgegeven SQL-instructie rechtsreeks doorgegeven aan de database-server waar de met SET DATABASE ingestelde database zich bevindt.

Als een SQL-instructie gereserveerde SQL- of IDAPI-woorden bevat en u voert de instructie uit op dBASE- of Paradox-tabellen, moet u de gereserveerde woorden tussen enkele (') of dubbele (") aanhalingstekens plaatsen en SQL-tabelaliassen (die anders zijn dan de aliassen voor dBASE-tabellen) gebruiken om de velden aan te duiden, bijvoorbeeld:

```
SELECT * FROM bedrijf.dbf b WHERE b.'CHAR' = 'element'
```

U kunt tabelaliassen gebruiken om velden aan te geven in de SELECT-, WHERE-, GROUP BY- of ORDER BY-clausules in instructies met SELECT. Dat is met name handig als u gegevens uit meerdere tabellen aan een query onderwerpt. SQLEXEC( ) resulteert in foutcodes met dezelfde waarden als die door ERROR( ) en MESSAGE( ) als resultaat worden gegeven. Als nul als resultaat wordt gegeven, wilt dat zeggen dat tijdens de uitvoering van de instructie geen fout heeft plaatsgevonden. Als een fout optreedt, kunt u de functies DBERROR( ) en DBMESSAGE( ) gebruiken om IDAPI-fouten als resultaat te geven en de functies SQLERROR( ) en SQLMESSAGE( ) om rechtstreeks bij de database-server informatie op te vragen over de oorzaak van de fout. (De functie ERROR( ) resulteert in de foutcode 240 als een server-fout heeft plaatsgevonden.)

Voorbeeld

In het volgende voorbeeld wordt een SQL SELECT-instructie uitgevoerd op de server-tabel:

```
SET DBTYPE TO DBASE
OPEN DATABASE NWBedrijf
SET DATABASE TO NWBedrijf
foutcode = SQLEXEC("SELECT Bedrijf, Plaats FROM ;
  Bedrijf WHERE Regio='NW'", "RegioNW.DBF")
IF foutcode = 0
  SET DATABASE TO
  USE RegioNW
  LIST
ENDIF
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DBERROR( ), DBMESSAGE( ), ERROR( ), MESSAGE( ), OPEN DATABASE, SET DATABASE, SET DBTYPE, SET PATH, SQLERROR( ), SQLMESSAGE( )

SQLMESSAGE( )

Foutafhandeling en testen op fouten

Resulteert in de melding voor de laatste server-fout.

Syntaxis

SQLMESSAGE( )

Beschrijving

Gebruik SQLMESSAGE( ) om de foutmelding voor de laatste server-fout te achterhalen. De foutcode kunt u opvragen met SQLERROR( ).

Zie Help voor een lijst van alle foutmeldingen.

Voorbeeld

Zie SQLERROR( ) voor een voorbeeld met SQLMESSAGE( ).

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CERROR( ), DBERROR( ), DBMESSAGE( ), ERROR( ), MESSAGE( ), ON ERROR, RETRY, SQLERROR( )

SQRT( )

Numerieke gegevens

Resulteert in de vierkantswortel van een getal.

Syntaxis

SQRT(<*Nuitdr*>)

<*Nuitdr*>

Een positief getal waarvan de vierkantswortel als resultaat moet worden gegeven. Als <*Nuitdr*> een negatief getal is, wordt een fout als resultaat gegeven.

Beschrijving

SQRT( ) resulteert in een zwevende waarde die de positieve vierkantswortel van een niet-negatief getal is. SQRT(36) geeft bijvoorbeeld 6 als resultaat omdat 6^2=36. De vierkantswortel van 0 is 0.

U kunt de vierkantswortel ook bepalen door het getal te verheffen tot de macht 0,5. De volgende twee instructies geven beide hetzelfde resultaat:

```
? SQRT(36)      && geeft 6,00 als resultaat
? 36^.5         && geeft 6,00 als resultaat
```

Het aantal decimalen stelt u in met SET DECIMALS.

Voorbeeld

In het volgende voorbeeld wordt SQRT( ) gebruikt om de lengte te berekenen van een dakbalk op basis van de breedte van de kamer en de hoogte van de nok:

```
SET PROCEDURE TO PROGRAM(1) ADDITIVE
LOCAL f
f=NEW Dakbalkform()
f.OPEN()
CLASS Dakbalkform OF FORM
  this.Top=2
  this.Left=2
  this.Width=40
  this.Height=12
  this.Text = "Dakbalk berekenen"
DEFINE ENTRYFIELD Kamer OF THIS AT 2,32;
  PROPERTY Picture "999",Value 0, Width 4
DEFINE ENTRYFIELD Nok OF THIS at 4,32;
  PROPERTY Picture "99",Value 0, Width 4
DEFINE ENTRYFIELD Balklengte OF THIS AT 6,32;
  PROPERTY Width 6, Value 0,;
    OnGotFocus Resultaat
DEFINE TEXT Ln1 OF THIS AT 2,3;
  PROPERTY Text "Geef kamerbreedte op:",;
    Width 25
DEFINE TEXT Ln2 OF THIS AT 4,3;
  PROPERTY Text "Geef nokhoogte op:",;
    Width 25
DEFINE TEXT Ln3 OF THIS AT 6,3;
  PROPERTY Text "Lengte van dakbalk wordt:",;
    Width 25, ColorNormal "R/W"
DEFINE PUSHBUTTON Sluiten OF THIS AT 9,14;
  PROPERTY TEXT "Sluiten", WIDTH 14, OnClick {;Form.Close()}
ENDCLASS

FUNCTION Resultaat
Form.Balklengte.Value=;
STR(SQRT(((Form.Kamer.Value/2)^2)+;
(Form.Nok.Value^2)),5,2)
RETURN .T.
```

Zie ook

EXP( ), LOG( ), LOG(10), SET DECIMALS

STATIC

Definieert lokale geheugenvariabelen die alleen beschikbaar zijn in de subroutine waarin deze zijn gedefinieerd, maar die in het geheugen bewaard blijven tot dBASE wordt afgesloten.

Syntaxis

STATIC *<variabele1>* [<= *waarde1>*] [,*<variabele2>* [<= *waarde2>*] ,...]

<variabele>
De variabele die moet worden gedefinieerd als STATIC.

<waarde>
De waarde die aan de variabele moet worden toegekend.

Beschrijving

Met STATIC kunt u geheugenvariabelen definiëren die alleen beschikbaar zijn in een bepaalde subroutine maar wel in het geheugen bewaard blijven. STATIC-variabelen kennen twee belangrijke verschillen met andere soorten geheugenvariabelen:

- U kunt in een enkele instructie een STATIC-variabele definiëren en een waarde toekennen aan die variabele.
- Aan STATIC-variabelen die in een enkele instructie worden geïnitialiseerd, hebben de waarde die in de instructie is toegekend alleen de eerste keer dat de subroutine wordt uitgevoerd.

Als u een STATIC-variabele definieert in een subroutine (een programma, procedure of door de gebruiker gedefinieerde functie) kan die variabele alleen binnen die subroutine worden aangeroepen. De variabele is niet zichtbaar in subroutines op hogere of lagere niveaus. Als de subroutine wordt beëindigd, blijft de STATIC-variabele echter in het geheugen bewaard met de waarde die de variabele had op het moment dat de subroutine werd beëindigd. Als de subroutine opnieuw wordt aangeroepen, heeft de variabele nog de waarde waarop de variabele was ingesteld op het moment dat de subroutine het laatst werd beëindigd. De waarde blijft gehandhaafd tot de subroutine deze wijzigt.

Als u een variabele definieert met STATIC zonder een waarde toe te kennen, wordt de variabele gemaakt met de waarde .F.

Omdat STATIC-variabelen niet worden vrijgegeven op het moment dat de subroutine waarin deze zijn gemaakt, wordt beëindigd, kunt u deze variabelen gebruiken om waarden te behouden voor een volgende keer dat de subroutine wordt uitgevoerd. U moet de variabele definiëren en initialiseren in een enkele instructie, bijvoorbeeld:

```
** subroutine gtest
STATIC gvar = 100
```

De eerste keer dat deze instructie wordt aangetroffen, wordt gvar geïnitialiseerd met de waarde 100. Als de subroutine gtest opnieuw wordt uitgevoerd, wordt gvar niet

opnieuw geïnitialiseerd. In plaats daarvan behoudt gvar de waarde die de variabele had op het moment dat de subroutine de laatste keer werd beëindigd.

Bij PUBLIC vindt u een tabel waarin het bereik en de beschikbaarheid van variabelen die zijn gedefinieerd met PUBLIC, PRIVATE, LOCAL en STATIC, worden vergeleken. Zie Hoofdstuk 5 in *Programmeren* voor meer informatie over het initialiseren en behouden van waarden van statische variabelen.

Voorbeeld

In het volgende voorbeeld wordt vanuit een hoofdprogramma naar verschillende procedures op lagere niveaus gesprongen om het gebruik van STATIC-variabelen te demonstreren. Deze variabelen zijn alleen beschikbaar binnen de subroutine waarin deze zijn gedefinieerd en behouden de waarde, zelfs als op een ander niveau een variabele met dezelfde naam wordt gewijzigd:

```
***Hoofd.Prg***
CLOSE ALL
CLEAR ALL
CLEAR
SET TALK OFF
? "**Hoofd.PRG**"
STATIC nTotaal
PUBLIC tReeks
nTotaal = 7109.50
tReeks= "Hallo"
ON ERROR ? "Variabele niet beschikbaar";
                    && Melding bij fout
? nTotaal           && Resulteert in 7109,50
? tReeks            && Resulteert in "Hallo"
DO Eerste           && Naar proc Eerste springen
?
? "**Terug naar Hoofd.PRG**"
? nTotaal           && 7109,50 - waarde behouden;
                       na terugkeer uit routine;
                       op lager niveau

PROCEDURE Eerste
?
? "**Proc Eerste**"  && Ter oriëntatie
STATIC Deadline      && Var gedefinieerd als STATIC;
                       in procedure Eerste
? Deadline           && .F. omdat geen waarde;
                       is toegekend
Deadline = {31-12-99} && Variabele initialiseren
? Deadline           && var heeft nu waarde
DO Proc1             && Naar Proc1 springen
RETURN               && Terug naar regel na;
                       DO Eerste in Hoofd

PROCEDURE Proc1
?
? "**Proc Proc1**"   && Ter oriëntatie
? tReeks             && "Hallo" is PUBLIC
? nTotaal            && "Variabele niet beschikbaar" ;
```

```
                          STATIC in Hoofd
? Deadline              && "Variabele niet beschikbaar" ;
                          STATIC in Eerste
Do Sub1                 && Naar Sub1 springen
?
? "**Terug naar Proc1**"
? nTotaal               && "Variabele niet beschikbaar" ;
                          STATIC in andere procedure
RETURN                  && Terug naar laatste regel ;
                          in Eerste

PROCEDURE Sub1
?
? "**Proc Sub1**"       && Ter oriëntatie
nTotaal = 8801.11       && nTotaal geinitialiseerd met;
                          nieuwe waarde, maar nog steeds ;
                          STATIC-variabele
? nTotaal               && 8801,11 - gedefinieerd in ;
                          huidige proc
? tReeks                && "Hallo" - PUBLIC-variabele
RETURN                  && Terug naar regel na ;
                          DO Sub1 in Proc1
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CLEAR MEMORY, DECLARE, LOCAL, PRIVATE, PUBLIC, RELEASE, STORE

STORE

Geheugenvariabelen

Slaat een uitdrukking op in opgegeven geheugenvariabelen of array-elementen.

Syntaxis

STORE *<uitdr>* TO *<variabelenlijst>* | *<array-elementenlijst>*
of
<variabele> | *<array-element>* = *<uitdr>*

<uitdr>

De uitdrukking die moet worden opgeslagen.

TO *<variabelenlijst>* | *<array-elementenlijst>*

Slaat *<uitdr>* op in de geheugenvariabelen in *<variabelenlijst>* of in de array-elementen in *<array-elementenlijst>*. U moet de array definiëren met DECLARE voordat u STORE kunt gebruiken.

<variabele> | <array-element> = <uitdr>
De enkele geheugenvariabele of het enkele array-element waarin <*uitdr*> wordt opgeslagen met de toewijzingsoperator (=). U moet de array definiëren met DECLARE voordat u = kunt gebruiken.

Beschrijving

Met STORE kunt u elke geldige uitdrukking opslaan in een enkele geheugenvariabele, in meerdere geheugenvariabelen of in een of meer array-elementen. Gebruik = om een geldige uitdrukking op te slaan in een enkele geheugenvariabele of een enkel array-element. Als <*uitdr*> een veldnaam is, wordt de inhoud van dat veld gebruikt. U kunt een veld van elk type opgeven, inclusief memo.

Voordat u een waarde toekent aan een variabele kunt u het *bereik* van die variabele instellen met LOCAL, PRIVATE, PUBLIC of STATIC. Geheugenvariabelen zijn niet gebonden aan een bepaalde *sessie*, de beschikbaarheid van een variabele binnen een applicatie wordt alleen bepaald door het bereik ervan.

Als u STORE opgeeft, wordt een van de volgende bewerkingen uitgevoerd:

- Er wordt een nieuwe geheugenvariabele gemaakt met de waarde <*uitdr*> of een bestaande geheugenvariabele met dezelfde naam wordt overschreven met de waarde <*uitdr*>. (U kunt TYPE( ) gebruiken om te bepalen of een geheugenvariabele bestaat.) Aan de geheugenvariabele wordt het gegevenstype van <*uitdr*> toegekend. SET SAFETY heeft geen effect op instructies met STORE.
- De waarde van <*uitdr*> wordt opgeslagen in een bestaand array-element dat u opgeeft. Als het gegevenstype van het array-element afwijkt van dat van <*uitdr*>, wordt het gegevenstype van het array-element in overeenstemming gebracht met het gegevenstype van <*uitdr*>. U kunt ook AFILL( ) gebruiken om dezelfde waarde toe te kennen aan meerdere array-elementen.

Als u STORE gebruikt om een variabele te maken met dezelfde naam als een veld in de huidige tabel, laat u de naam van die variabele in instructies vooraf gaan door m->, zoals in het volgende voorbeeld:

```
** Huidige tabel bevat veld genaamd "auteur"
STORE "Hugo" to auteur
? auteur      && Toont inhoud van veld
? m->auteur   && Toont "Hugo"
```

Als u een memoveld wilt opslaan in een geheugenvariabele, kunt u STORE of STORE MEMO gebruiken. STORE slaat het memoveld uit een record, inclusief regelterugloop- en regeldoorvoertekenes, op in een enkele geheugenvariabele of een enkel array-element. STORE MEMO slaat elke afzonderlijke regel van een memoveld op in een element van een bestaande array.

Voorbeeld

In het volgende voorbeeld wordt STORE gebruikt om vier afkortingen op te slaan in vier elementen van een eendimensionale array met de naam Staten. Vervolgens wordt een DO WHILE-lus met een teller gebruikt om voor de in het geheugen opgeslagen staten records op te halen uit de tabel Afnemers:

```
SET TALK OFF
SET SAFETY OFF
DECLARE Staten[4]
STORE "TX" TO Staten[1]
STORE "GA" TO Staten[2]
STORE "WA" TO Staten[3]
STORE "MN" TO Staten[4]
USE Afnemers EXCLUSIVE
INDEX ON Bedrijf Tag Bedrijf
CLEAR
Teller = 1
? CENTER("Bedrijvenlijst voor "+Staten[1] ;
  +", "+Staten[2]+ ", "+Staten[3]+" en "  ;
  +Staten[4])
?
DO WHILE Teller < 5
  SCAN FOR Staat = Staten[Teller]
  ? Bedrijf AT 5, Contact, Staat
  ENDSCAN
  Teller=Teller+1
ENDDO
CLOSE ALL
SET TALK ON
SET SAFETY ON
```

Overdraagbaarheid

In dBASE III PLUS worden geen array-elementenlijsten ondersteund.

Zie ook

ACCEPT, AFILL( ), DECLARE, INPUT, LOCAL, PRIVATE, PUBLIC, RESTORE, SAVE, SET SAFETY, STATIC, STORE MEMO, WAIT

STORE AUTOMEM

Velden en records

Slaat de inhoud van alle velden in het huidige record op in een reeks geheugenvariabelen.

Syntaxis

STORE AUTOMEM

Beschrijving

STORE AUTOMEM kopieert alle velden in het huidige record naar een reeks overeenkomstige automatische geheugenvariabelen (ook wel AUTOMEM-variabelen genoemd). Elke geheugenvariabele heeft dezelfde naam, dezelfde lengte en hetzelfde gegevenstype als het bijbehorende veld. Als de geheugenvariabelen nog niet bestaan, worden deze gemaakt.

In automatische geheugenvariabelen kunt u tijdelijk gegevens uit tabelrecords opslaan. U kunt de gegevens dan manipuleren als geheugenvariabelen in plaats van veldwaarden en vervolgens de waarden weer terug in het record plaatsen (met REPLACE AUTOMEM, APPEND AUTOMEM of INSERT AUTOMEM).

STORE AUTOMEM is een van de drie commando's waarmee automatische geheugenvariabelen kunnen worden gemaakt. De andere twee, USE *<bestandsnaam>* AUTOMEM en CLEAR AUTOMEM, initialiseren lege automatische geheugenvariabelen voor de velden in de huidige tabel. Deze twee commando's kopiëren geen gegevens naar de automatische geheugenvariabelen en de variabelen blijven leeg tot een ander commando wordt gebruikt om gegevens in de variabelen op te slaan. De lege variabelen die worden gemaakt met USE...AUTOMEM en CLEAR AUTOMEM worden meestal gebruikt om gegevens van buiten het programma, bijvoorbeeld uit een gegevensinvoerformulier, toe te voegen aan, in te voegen in of te vervangen in een tabel. STORE AUTOMEM kopieert gegevens uit een tabel naar automatische geheugenvariabelen.

Het gebruik van geheugenvariabelen voor het wijzigen van gegevens in tabellen biedt meer beheersing over de gegevens dan het rechtstreekse gebruik van de veldwaarden. Als u een veld wilt opgeven als een GET-argument in een @...SAY...GET-instructie, wordt het veld nadat het is gewijzigd met READ, rechtsreeks terug in de tabel geplaatst. Als u een geheugenvariabele gebruikt als een GET-argument, kunt u nog andere bewerkingen uitvoeren op de gewijzigde variabele, bijvoorbeeld de geldigheid controleren, voordat de waarde met REPLACE AUTOMEM weer terug in de tabel wordt geplaatst.

Voorbeeld

In het volgende voorbeeld wordt STORE AUTOMEM gebruikt om voor records die zijn gemarkeerd voor verwijdering, veldwaarden naar een reeks automatische geheugenvariabelen te kopiëren. Het commando LOCATE start de zoekbewerking naar het eerste gemarkeerde record en een DO WHILE .NOT. EOF( )-lus met een CONTINUE-instructie kopieert alle overige gemarkeerde records naar Tijd.DBF:

```
SET SAFETY OFF
SET DELETED OFF
CLOSE DATABASES
USE Klanten IN 1
SELECT 1
COPY STRUCTURE TO Tijd
USE Tijd IN 2
LOCATE FOR DELETED()
  IF FOUND()
    DO WHILE .NOT. EOF()
      STORE AUTOMEM
      SELECT 2
      APPEND AUTOMEM
      SELECT 1
      CONTINUE
    ENDDO
    SELECT 2
    BROWSE
  ELSE
```

```
    ? "Geen voor verwijdering gemarkeerde records"
  ENDIF
RETURN
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CLEAR AUTOMEM, REPLACE, USE

STORE MEMO

Velden en records

Slaat de tekst in een memoveld op in een array-geheugenvariabele.

Syntaxis

STORE MEMO *<memoveld>* TO ARRAY *<array-naam>*

<memoveld>

Het memoveld waarvan de tekst moet worden opgeslagen in een array.

TO ARRAY *<arraynaam>*

De array waarin de tekst uit het memoveld wordt opgeslagen. In elk array-element wordt één regel uit het memoveld opgeslagen. De lengte van elk element is maximaal het aantal tekens dat is ingesteld met SET MEMOWIDTH.

Beschrijving

Met STORE MEMO kunt u de tekst uit memovelden opslaan in array-geheugenvariabelen. (Met STORE AUTOMEM kunt u de inhoud van alle soorten velden in het huidige record opslaan in automatische geheugenvariabelen, behalve van memovelden.) Nadat u de inhoud van de array-elementen hebt gewijzigd, kunt u REPLACE MEMO gebruiken om de nieuwe inhoud terug te plaatsen in het memoveld.

Initialiseer de array met het commando DECLARE voordat u STORE MEMO gebruikt. <array-naam> moet een eendimensionale array zijn.

Initialiseer de array met voldoende elementen voor alle memoveldtekst. Als het memoveld teveel tekst bevat voor de array, wordt de tekst ingekort en gaan de regels aan het eind van het memoveld verloren als u REPLACE MEMO gebruikt. U moet dus zorgen dat de array voldoende elementen bevat voor het grootste memoveld dat u in de array wilt opslaan. U kunt de functie MEMLINES( ) gebruiken om het aantal elementen voor een memoveld te bepalen.

Aan elk array-element dat tekst uit een memoveld bevat, wordt het type teken toegewezen. Als de tekst in het memoveld niet de gehele array vult, behouden de overige elementen hun oorspronkelijke waarde (standaard de logische waarde .F.).

Voorbeeld

In het volgende voorbeeld wordt STORE MEMO gebruikt om tekst uit een memoveld in een array-geheugenvariabele te plaatsen:

```
SET TALK OFF
Kolbreedte = 50
Mbreedte = SET("MEMOWIDTH")
SET MEMOWIDTH TO Kolbreedte
USE Bedrijf
DO WHILE .NOT. EOF()
CLEAR
AantRegs = 1
IF .NOT. ISBLANK(Notities)
  DECLARE Memoregel[MEMLINES(Notities)]
  STORE MEMO Notities TO ARRAY Memoregel
  DO WHILE AantRegs <= MEMLINES(Notities)
    @ AantRegs,12 GET Memoregel[AantRegs]
     AantRegs = AantRegs + 1
  ENDDO
  READ
  vervangen_jn = "A"
  @ AantRegs + 2,2 SAY "Wilt u de tekst Vervangen, " + ;
  "tekst Toevoegen of Annuleren ? (V/T/A)";
  GET vervangen_jn PICTURE "!" ;
     VALID vervangen_jn $ "VTA"
  READ
  DO CASE
    CASE vervangen_jn = "V"
      REPLACE MEMO Notities WITH ARRAY Memoregel
    CASE vervangen_jn = "T"
      REPLACE MEMO Notities ;
        WITH ARRAY Memoregel ADDITIVE
    CASE vervangen_jn = "A"
      CLEAR
      EXIT
  ENDCASE
ENDIF
SKIP
ENDDO
CLOSE ALL
SET MEMOWIDTH TO Mbreedte
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

DECLARE, MEMLINES( ), MLINE( ), REPLACE MEMO, STORE

STR( )

Resulteert in een tekenreeks die overeenkomt met een opgegeven numerieke uitdrukking.

Syntaxis

STR(<*Nuitdr*> [, <*lengte Nuitdr*> [, <*decimalen Nuitdr*> [, <*Tuitdr*>]]])

<*Nuitdr*>

De numerieke of zwevende uitdrukking die u wilt omzetten in een tekenreeks.

<*lengte Nuitdr*>

De lengte van de resulterende tekenreeks. Het geldige bereik loopt van 1 tot en met 20, inclusief decimaalscheidingsteken, decimalen en minteken. De standaardinstelling is 10. Als <*lengte Nuitdr*> kleiner is dan het aantal cijfers voor het decimaalscheidingsteken in <*Nuitdr*>, resulteert STR( ) in een reeks asterisken (*).

<*decimalen Nuitdr*>

Het aantal tekens dat moet worden gereserveerd voor cijfers achter het decimaalscheidingsteken. De standaardwaarde en tevens laagste geldige waarde is 0. Als u geen waarde opgeeft voor <*decimalen Nuitdr*>, wordt <*Nuitdr*> afgerond op het dichtstbijzijnde gehele getal. Als u een waarde wilt opgeven voor <*decimalen Nuitdr*>, moet u ook een waarde opgeven voor <*lengte Nuitdr*>.

<*Tuitdr*>

Het teken waarmee het begin van de tekenreeks moet worden opgevuld als de tekenreeks die als resultaat wordt gegeven, kleiner is dan <*lengte Nuitdr*>. Het standaardteken is een spatie. Als u een waarde wilt opgeven voor <*Tuitdr*>, moet u ook waarden opgeven voor <*lengte Nuitdr*> *en* <*decimalen Nuitdr*>. Als u meerdere tekens opgeeft voor <*Tuitdr*>, wordt alleen het eerste teken gebruikt.

Beschrijving

Met STR( ) kunt u numerieke gegevens omzetten in tekengegevens, zodat u de waarde kunt manipuleren als een tekenreeks. U kunt STR( ) bijvoorbeeld gebruiken om het numerieke veld om te zetten in tekens als u wilt indexeren op een numeriek veld in combinatie met een tekenveld.

De waarde wordt afgerond en de tekenreeks wordt opgevuld op basis van de parameters die u instelt met <*lengte Nuitdr*> en <*decimalen Nuitdr*>. Daarbij gelden de volgende regels:

- Als <*decimalen Nuitdr*> kleiner is dan het aantal decimalen in <*Nuitdr*>, wordt de waarde afgerond op het meest nauwkeurige getal dat past binnen <*lengte Nuitdr*>. De instructie STR(10.765,5,1) resulteert bijvoorbeeld in " 10,8" (met één voorloopspatie) en STR(10.765,5,2) in "10,77".

- Als *<lengte Nuitdr>* niet lang genoeg is voor het aantal decimalen dat is ingesteld met *<decimalen Nuitdr>*, wordt *<Nuitdr>* afgerond op het meest nauwkeurige getal dat past binnen *<lengte Nuitdr>*. De instructie STR(10.765,4,3) resulteert bijvoorbeeld in "10,8".
- Als *<decimalen Nuitdr>* groter is dan het aantal decimalen in *<Nuitdr>* en *<lengte Nuitdr>* groter dan de reeks die als resultaat wordt gegeven, worden aan het eind van de reeks nullen toegevoegd. Er worden niet meer nullen toegevoegd dan mogelijk zijn op basis van *<decimalen Nuitdr>*.
- Als de resultaatreeks nog steeds korter is dan *<lengte Nuitdr>*, wordt de reeks aan de linkerkant opgevuld tot *<lengte Nuitdr>* is bereikt. De instructie STR(10.765,8,6) resulteert bijvoorbeeld in "10,76500" (*<lengte Nuitdr>* is 8), STR(10.765,7,6) in "10,7650" (*<lengte Nuitdr>* is 7) en STR(10.765,12,6) in " 10,765000" (*<lengte Nuitdr>* is 12, dus worden drie voorloopspaties ingevoegd).

Voorbeeld

In het volgende voorbeeld wordt STR( ) gebruikt om de inhoud van een numeriek veld in combinatie met een tekenreeks te tonen:

```
SET TALK OFF
USE Afnemers
SET FILTER TO OpenBalans > 0
* Alleen records met positieve OpenBalans
GO TOP
? OpenBalans                        && Resulteert in 456,00
? "De beginbalans was F " + ;
   STR(OpenBalans,8,2)              && Resulteert in F   456,00
? "De beginbalans kan ook zo worden opgemaakt: F " ;
   + LTRIM(STR(OpenBalans,8,2))     && Resulteert in F 456,00
? "Of zo: " + ;
   LTRIM(STR(OpenBalans,8,2,":")) && Resulteert in ::456,00
```

Overdraagbaarheid

Het argument *<uitdr C>* wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

SET POINT, SET SEPARATOR, SUBSTR( ), VAL( )

STUFF( )

Tekenreeksgegevens

Resulteert in een tekenreeks waarin opgegeven tekens zijn vervangen door andere tekens.

Syntaxis

STUFF(*<doel Tuitdr>* | *<doelmemoveld>*,
<begin Nuitdr>, *<aantal Nuitdr>*, *<vervanging Tuitdr>*)

<doel Tuitdr> | <doelmemoveld>
De tekenreeks of het memoveld waarin tekens moeten worden vervangen.

<begin Nuitdr>
De tekenpositie in de tekenreeks of het memoveld waarop tekens moeten worden vervangen.

<aantal Nuitdr>
Het aantal tekens dat uit de tekenreeks of het memoveld moet worden verwijderd.

<vervanging Tuitdr>
De tekens die in de tekenreeks of het memoveld moeten worden ingevoegd.

Beschrijving

STUFF( ) resulteert in een doeltekenuitdrukking of -memoveld waarin op een opgegeven positie tekens zijn vervangen. Vanaf de positie die u opgeeft met *<begin Nuitdr>*, worden *<aantal Nuitdr>* tekens uit de oorspronkelijke reeks verwijderd. STUFF( ) resulteert in een reeks van maximaal 32.766 tekens (de maximumlengte van een tekenreeks).

Als de oorspronkelijke tekenuitdrukking een lege tekenreeks is of als het oorspronkelijke memoveld leeg is, resulteert STUFF( ) in de vervangende reeks.

Als *<begin Nuitdr>* kleiner dan of gelijk is aan 0, gebruikt STUFF( ) 1 voor *<begin Nuitdr>*. Als *<aantal Nuitdr>* kleiner dan of gelijk is aan 0, wordt de vervangreeks ingevoegd op positie *<begin Nuitdr>* zonder dat tekens uit de oorspronkelijke reeks worden verwijderd.

Als *<begin Nuitdr>* groter is dan de lengte van de oorspronkelijke reeks, worden geen tekens verwijderd en wordt de vervangreeks aan het eind van de oorspronkelijke reeks toegevoegd.

Als de vervangende reeks leeg is, worden *<aantal Nuitdr>* tekens verwijderd vanaf positie *<begin Nuitdr>*, maar worden geen tekens ingevoegd.

Voorbeeld

In het volgende voorbeeld wordt STUFF( ) gebruikt om de tekst in een aantal tekenreeksen te vervangen:

```
? STUFF("Janson",5,1,"e")      && Geeft "Jansen" als resultaat
? STUFF("Jansen",4,0,"s")      && Geeft "Janssen" als resultaat
? STUFF("",2,5,"vader")        && Geeft "vader" als resultaat
? STUFF("roos",0,1,"b")        && Geeft "boos" als resultaat
? STUFF("roos",5,2,"jes")      && Geeft "roosjes" als resultaat
? STUFF("roosjes",5,3,"")      && Geeft "roos" als resultaat
```

In het volgende voorbeeld wordt STUFF( ) gebruikt om overal in het veld Naam van Tijd.DBF "&" te vervangen door "en". Tijd.DBF is een kopie van de tabel Klanten.

```
USE Klanten
COPY TO Tijd
```

```
USE Tijd
SET TALK OFF
SCAN
  IF "&" $ Naam
      REPLACE Naam with ;
      STUFF(Naam,AT("&",Naam),1,"en")
  ENDIF
ENDSCAN
SET TALK ON
RETURN
```

Overdraagbaarheid

Het argument <*memoveld*> wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

AT( ), LEFT( ), RAT( ), REPLICATE( ), RIGHT( ), SPACE( ), SUBSTR( )

SUBSTR( ) — Tekenreeksgegevens

Resulteert in een deelreeks uit een opgegeven tekenreeks of memoveld.

Syntaxis

SUBSTR(<*Tuitdr*> | <*memoveld*>, <*begin Nuitdr*> [, <*lengte Nuitdr*>])

<*Tuitdr*> | <*memoveld*>

De tekenreeks of het memoveld waaruit u tekens wilt ophalen.

<*begin Nuitdr*>

De tekenpositie in de tekenreeks of het memoveld waarop de deelreeks begint.

<*lengte Nuitdr*>

Het aantal tekens dat uit de tekenreeks of het memoveld moet worden opgehaald.

Beschrijving

SUBSTR( ) haalt uit een tekenuitdrukking of memoveld <*lengte Nuitdr*> tekens op vanaf de positie die u opgeeft met <*begin Nuitdr*>. Maximaal worden 32.766 tekens opgehaald (de maximumlengte van een tekenreeks). Als <*lengte Nuitdr*> kleiner is dan of gelijk aan 0, resulteert SUBSTR( ) in een lege tekenreeks.

Als u geen <*lengte Nuitdr*> opgeeft, resulteert SUBSTR( ) in alle tekens vanaf positie <*begin Nuitdr*> tot het eind van de reeks. Als <*lengte Nuitdr*> groter is dan het aantal tekens vanaf <*begin Nuitdr*> tot het eind van de tekenreeks, resulteert SUBSTR( ) alleen in het aantal tekens in de reeks zonder de resultaatreeks op te vullen met spaties of andere tekens. U kunt LEN( ) gebruiken om de lengte van de resultaatreeks te bepalen.

Onder de volgende omstandigheden wordt een fout als resultaat gegeven:

- *<Tuitdr>* is een lege tekenreeks
- *<memoveld>* is leeg
- *<begin Nuitdr>* is nul
- *<begin Nuitdr>* is een negatief getal
- *<begin Nuitdr>* is groter dan het aantal tekens in *<Tuitdr>* of *<memoveld>*

Als SUBSTR( ) tekens uit een memoveld ophaalt, wordt voor elke combinatie van een regelterugloop en een regeldoorvoer (CR/LF) in het memoveld twee tekens gerekend.

Met de deelreeks-operator ($) kunt u vaststellen of een tekenreeks bestaat binnen een andere tekenreeks. Zie Hoofdstuk 1 voor meer informatie over operatoren.

Voorbeeld

In de volgende voorbeelden wordt SUBSTR( ) gebruikt om een deel van een tekenreeks op te halen:

```
? SUBSTR("Dame",1)          && Geeft "Dame" als resultaat
? SUBSTR("boerderij",7,3)   && Geeft "rij" als resultaat
? SUBSTR("is",1,1)          && Geeft "i" als resultaat
? SUBSTR("dame",3,1)        && Geeft "m" als resultaat
```

SUBSTR( ) is goed bruikbaar bij het sorteren en manipuleren van gegevens. U kunt bijvoorbeeld een tabel sorteren op de lettercode van een postcode als dit veld de opmaak "nnnn xx" heeft. Maak als volgt een indexlabel met SUBSTR( ).

```
USE Namen EXCLUSIVE
INDEX ON SUBSTR(Postcode,6,2) TAG LetCode
```

Als volgt kunt u met SUBSTR( ) veldgegevens in hoofdletters omzetten naar hoofdletters en kleine letters (veldlengte = 20).

```
USE Namen
REPLACE ALL Anaam WITH;
  UPPER(SUBSTR(Anaam,1,1))+LOWER(SUBSTR(Anaam,2,19))
```

Het voorgaande voorbeeld is alleen zinvol als in het veld slechts één enkele naam of tekenreeks staat. Als u alle eerste letters van woorden in een veld wilt omzetten in hoofdletters, kunt u PROPER( ) gebruiken:

```
REPLACE ALL Anaam with PROPER(Anaam)
```

Overdraagbaarheid

Het argument *<memoveld>* wordt niet ondersteund in dBASE III PLUS. In dBASE IV en dBASE III PLUS is de lengte van het resultaat van SUBSTR( ) beperkt tot 254 tekens, en in beide versies resulteert SUBSTR( ) in een fout als *<begin Nuitdr>* of *<lengte Nuitdr>* gelijk is aan nul of als *<lengte Nuitdr>* een negatief getal is.

Zie ook

AT( ), LEFT( ), LEN( ), PROPER( ), RAT( ), RIGHT( ), STUFF( )

SUM

Berekent een totaal voor opgegeven numerieke en zwevende velden in de huidige tabel en slaat de uitkomsten op in geheugenvariabelen of in een array.

Syntaxis

SUM
[<*uitdrukkingenlijst*>]
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[TO <*variabelenlijst*> | TO ARRAY <*arraynaam*>]

<*uitdrukkingenlijst*>

De numerieke of zwevende velden, uitdrukkingen met numerieke of zwevende velden of uitdrukkingen waarmee tekenvelden worden omgezet in numerieke of zwevende waarden, waarvan u de totalen wilt berekenen.

<*bereik*>

Het aantal records dat u wilt optellen. RECORD <*n*> geeft een record aan door middel van het recordnummer. NEXT <*n*> geeft *n* records aan, te beginnen bij het huidige record. ALL geeft alle records aan. REST geeft alle records aan vanaf het huidige record tot het eind van het bestand.

FOR <*voorwaarde1*>
WHILE <*voorwaarde2*>

Bepaalt welke records worden verwerkt met SUM. FOR beperkt SUM tot records die voldoen aan <*voorwaarde1*>. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang <*voorwaarde2*> waar is.

TO <*variabelenlijst*> | TO ARRAY <*array-naam*>

TO <*variabelenlijst*> geeft een lijst van geheugenvariabelen aan waarin de totalen moeten worden opgeslagen. Als <*variabelenlijst*> array-indextekens bevat, moeten die array's al bestaan. TO ARRAY <array-naam> geeft een array aan waarin de totalen moeten worden opgeslagen.

Beschrijving

Het commando SUM berekent de som van de waarden van numerieke uitdrukkingen en slaat de uitkomsten op in geheugenvariabelen of afzonderlijke array-elementen. Als SET TALK is ingeschakeld (ON), worden de resultaten ook weergegeven in het resultatenpaneel van het commandovenster.

Het opgegeven aantal geheugenvariabelen moet exact overeenkomen met het aantal berekende totalen. De positie van de velden in <*uitdrukkingenlijst*> bepaalt in welke volgorde de waarden worden doorgegeven aan <*variabelenlijst*>. Als u een array

opgeeft, moet dat een eendimensionale array zijn en moeten de afzonderlijke elementen zijn geïnitialiseerd voordat u uitkomsten van SUM in de array kunt opslaan.

SUM is vergelijkbaar met TOTAL, waarmee op basis van een geïndexeerde of gesorteerde tabel een tweede tabel wordt gemaakt die de totalen bevat van numerieke en zwevende velden in records die zijn gegroepeerd op een sleuteluitdrukking.

Voorbeeld

In het volgende voorbeeld wordt SUM gebruikt om het totaal van het veld VerkTotNu voor alle bedrijven te berekenen:

```
USE Bedrijf
SUM VerkTotNu TO Vtn_som
? "De totale verkoop tot heden is F ", ;
   Vtn_som PICTURE "99,999,999.99"
```

In dit voorbeeld hoeft de tabel niet te worden gesorteerd omdat alle records worden gebruikt.

Zie ook

AVERAGE, CALCULATE, COUNT, TOTAL

SUSPEND

Foutafhandeling en testen op fouten

Onderbreekt de uitvoering van een programma tijdelijk en geeft de besturing over aan het commandovenster.

Syntaxis

SUSPEND

Beschrijving

Met SUSPEND kunt u de uitvoering van een programma op een bepaald punt, een *onderbrekingspunt*, tijdelijk onderbreken of opschorten. Het programma blijft onderbroken tot u het commando RESUME of CANCEL gebruikt, of tot u dBASE afsluit. Als u het commando RESUME geeft, gaat het programma verder vanaf het onderbrekingspunt. Als u het commando CANCEL geeft, wordt de uitvoering van het programma geannuleerd en wordt het programma uit het geheugen verwijderd. (CANCEL annuleert alle bestanden die zijn aangeroepen met DO, inclusief bestanden die zijn onderbroken. Alle procedurebestanden moet u sluiten met CLOSE PROCEDURE.)

Zolang een programma is opgeschort, kunt u commando's opgeven in het commandovenster. U kunt dan de toestand van onder meer bestanden, geheugenvariabelen en SET-commando's onderzoeken en wijzigen. Wijzigingen in het opgeschorte programma worden echter genegeerd. Als u een opgeschort programma

wilt wijzigen, moet u CANCEL gebruiken, het programma wijzigen en vervolgens opnieuw starten.

Als u in het commandovenster geheugenvariabelen initialiseert terwijl een programma is opgeschort, worden deze PRIVATE gedefinieerd op het niveau waar het programma werd opgeschort.

Denk er aan niet terug te keren naar een opgeschort programma door in het commandovenster DO <*bestandsnaam*> op te geven. Als u dat doet, raakt op een gegeven moment het geheugen uitgeput. Ook krijgt u te maken met "geneste" SUSPEND-instructies en wordt het op een gegeven moment onduidelijjk welk programma is opgeschort. Als u een opgeschort programma opnieuw vanaf het begin wilt uitvoeren, geeft u eerst CANCEL en dan DO <*bestandsnaam*> op.

Voorbeeld

In het volgende voorbeeld wordt gevraagd om een uit twee letters bestaande afkorting voor een staat en worden vervolgens de klanten binnen die staat getoond. Als het programma geen lijst van klanten toont, kan de programmeur het commando SUSPEND invoegen na de tweede CLEAR-instructie om het programma te onderbreken. In het commandovenster kunnen dan commando's worden opgegeven om de fout op te sporen, bijvoorbeeld ? gStaat om de waarde van de variabele gStaat op te vragen, LIST FOR STAAT= "CA", DISPLAY MEMORY of DISPLAY STATUS. Als het programma weer kan worden hervat, moet het commando RESUME worden opgegeven:

```
CLEAR
SET TALK OFF
USE Afnemers
ACCEPT "Geef afkorting voor staat (2 letters): ";
  TO gStaat
CLEAR
SUSPEND                  && Kan worden verwijderd als ;
                            programma foutloos werkt
? CENTER("Afnemers in "+UPPER(gSTaat))
?
SCAN FOR Staat = UPPER(gStaat)
  ? Bedrijf, Contact, Openbalans
ENDSCAN
RETURN
```

Zie ook

CANCEL, DO, RESUME, QUIT

TAG() — Tabelindeling

Resulteert in de naam van een .NDX-bestand of de label in een .MDX-bestand.

Syntaxis

TAG([<*.mdx-bestandsnaam Tuitdr*>,] <*indexnummer Nuitdr*> [,<*alias*>])

<*.mdx-bestandsnaam Tuitdr*>

De naam van een .MDX-bestand waarin een label moet worden gezocht op positie <*indexnummer Nuitdr*>.

<*indexnummer Nuitdr*>

Het nummer van het geopende .NDX-bestand of de indexlabel in het .MDX-bestand waarvan u de naam wilt ophalen.

<*alias*>

Een werkgebiednummer (van 1 tot 255) of -letter (van A tot J) of een aliasnaam. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens.

Beschrijving

TAG() resulteert in de naam van een .NDX-bestand of .MDX-label voor de index die wordt aangegeven met <*indexnummer Nuitdr*>. Het indexnummer geeft de positie aan van een index in de lijst met geopende indexen in het huidige of een opgegeven werkgebied, die zijn geopend met USE of SET INDEX. Als geen indexnummer is opgegeven, resulteert TAG() in de naam van de hoofdindex.

Als u de naam van een .MDX-bestand opgeeft, resulteert TAG() in labelnamen uit het opgegeven .MDX-bestand. De volgorde waarin de labelnamen in het .MDX-bestand zijn opgenomen, bepaalt de volgorde waarin deze door TAG() worden opgehaald.

Als u niet de naam van een .MDX-bestand opgeeft, resulteert TAG() in de labelnaam in de geopende indexlijst en worden eerst .NDX-bestanden gecontroleerd. Vervolgens worden de indexlabels in het produktie-MDX-bestand gecontroleerd en tenslotte de geopende .MDX-bestanden in de volgorde waarin deze zijn geopend.

Als zich op de opgegeven positie geen index of label bevindt, resulteert TAG() in een lege tekenreeks ("").

Voorbeeld

In het volgende voorbeeld wordt TAG() gebruikt om de namen op te vragen van indexbestanden die zijn geopend voor de huidige tabel:

```
USE Bedrijf EXCLUSIVE
INDEX ON CompCode   TAG CompCode
INDEX ON Bedrijf    TAG Bedrijf
INDEX ON Postcode TAG PoCo
* Er zijn nu tenminste 3 indexen voor Bedrijf.mdx.
FOR i=1 TO TAGCOUNT()
  * TAGCOUNT() is het totale aantal
  * indexen voor Bedrijf.mdx
  ? "Tag",i, TAG(i)
NEXT i
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

DBF(), DISPLAY STATUS, KEY(), MDX(), NDX(), ORDER(), SET INDEX, SET ORDER, TAGCOUNT(), TAGNO(), USE

TAGCOUNT() Tabelindeling

Resulteert in het aantal actieve indexen in een opgegeven werkgebied of .MDX-indexbestand.

Syntaxis

TAGCOUNT([<*.mdx-bestandsnaam Tuitdr*> [,<*alias*>]])

<*.mdx-bestandsnaam Tuitdr*>

Geeft het .MDX-bestand aan met de indexlabel die u wilt controleren. Als u een bestand zonder pad opgeeft, wordt dit gezocht in de huidige directory en vervolgens in het pad dat is ingesteld met SET PATH. Als u een bestand zonder extensie opgeeft, wordt de extensie .MDX gebruikt.

<*alias*>

Een werkgebiednummer (van 1 tot 255) of -letter (van A tot J) of een aliasnaam. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens.

Beschrijving

TAGCOUNT() resulteert in het totale aantal geopende indexen of het aantal indexlabels in een opgegeven .MDX-bestand. TAGCOUNT() resulteert in 0 als voor het huidige of opgegeven werkgebied geen indexen of indexlabels zijn geopend, of als het met <*bestandsnaam*> aangeduide .MDX-bestand niet bestaat. Als u niet de naam van een .MDX-bestand opgeeft, resulteert TAGCOUNT() in het totaal aantal indexen in het opgegeven werkgebied (inclusief .NDX-bestanden). Als u geen alias opgeeft, resulteert TAGCOUNT() in het totaal aantal indexen in het huidige werkgebied.

Voorbeeld

In het volgende voorbeeld wordt TAGCOUNT() gebruikt om het aantal geopende indexen voor de huidige tabel te bepalen:

```
CLOSE ALL
USE Bedrijf IN SELECT() EXCLUSIVE
SELECT Bedrijf
INDEX ON CompCode TAG CompCode
INDEX ON Bedrijf  TAG Bedrijf
* CompCode en Bedrijf opnemen in produktie-mdx
```

```
INDEX ON Postcode      TAG PoCo      OF Lokatie
* PoCo opnemen in Lokatie.mdx
INDEX ON Plaats     TO Plaats
* Plaats opnemen in Plaats.ndx
SET INDEX TO Plaats, Lokatie ORDER PoCo
* Bedrijf.mdx bevat nu tenminste 2 labels
* Lokatie.mdx bevat er een
* In Plaats.ndx is nog een index geopend
? "Er zijn " + LTRIM(STR(TAGCOUNT())) + ;
   " actieve indexen in werkgebied "  + ;
   LTRIM(STR(WORKAREA()))
WAIT
DISPLAY STATUS
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

DBF(), DISPLAY STATUS, KEY(), MDX(), NDX(), ORDER(), SET INDEX, SET ORDER, TAG(), TAGNO(), USE, WORKAREA()

TAGNO()

Tabelindeling

Resulteert in het indexnummer voor een opgegeven index als resultaat.

Syntaxis

TAGNO([<*labelnaam Tuitdr*> [,<*.mdx-bestandsnaam Tuitdr*> [,<*alias*>]]])

<labelnaam Tuitdr>

De naam van de indexlabel waarvan u de positie wilt ophalen. Als u geen labelnaam opgeeft, resulteert TAGNO() in de positie van de hoofdindex.

<.mdx-bestandsnaam Tuitdr>

De naam van het .MDX-bestand dat de opgegeven indexlabel bevat. Als u niet de naam van een .MDX-bestand opgeeft, resulteert TAGNO() in de positie van de labelnaam voor alle geopende indexbestanden in hetzelfde werkgebied, inclusief .NDX-bestanden boven aan de indexlijst.

<alias>

Een werkgebiednummer (van 1 tot 255) of -letter (van A tot J) of een aliasnaam. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens.

Beschrijving

TAGNO() rsulteert in een nummer dat de positie aangeeft van de opgegeven indexnaam in de lijst van geopende indexen in het huidige of een opgegeven werkgebied. De volgorde van de indexen wordt bepaald door de volgorde waarin deze zijn geopend met USE of SET INDEX.

Als u geen labelnaam opgeeft, resulteert TAGNO() in het nummer van de hoofdindex. Als u niet de naam van een .MDX-bestand opgeeft, wordt gezocht in de lijst van geopende indexbestanden in het opgegeven werkgebied, inclusief .NDX-bestanden. Als u geen alias opgeeft, wordt de lijst van geopende indexen in het huidige werkgebied gebruikt.

TAGNO() resulteert in een fout als de opgegeven indexlabel of het opgegeven .MDX-bestand niet bestaat.

Voorbeeld

In het volgende voorbeeld wordt TAGNO() gebruikt om het nummer van de opgegeven indexbestanden (.NDX-bestanden) op te halen:

```
USE Bedrijf EXCLUSIVE
INDEX ON CompCode   TO CompCode
INDEX ON Bedrijf    TO Bedrijf
INDEX ON Postcode   TO PoCo
INDEX ON Regio      TO Regio
INDEX ON Plaats     TO Plaats
SET INDEX TO ;
  CompCode, Bedrijf, Plaats, Regio, PoCo
? TAG(),TAGNO()
* Bijvoorbeeld "COMPCODE"   8
? TAGNO("CompCode")  && bijvoorbeeld 8
? TAGNO("Bedrijf")   && bijvoorbeeld 1
? TAGNO("Plaats")    && bijvoorbeeld 6
? TAGNO("Regio")     && bijvoorbeeld 3
? TAGNO("PoCo")      && bijvoorbeeld 11
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

DBF(), DISPLAY STATUS, KEY(), MDX(), NDX(), ORDER(), SET INDEX, SET ORDER, TAG(), TAGCOUNT(), USE, WORKAREA()

TAN() — Numerieke gegevens

Resulteert in de trigonometrische tangens van een hoek.

Syntaxis

TAN(<*Nuitdr*>)

<Nuitdr>

De grootte van de hoek in radialen. Gebruik DTOR( ) om de grootte van een hoek in graden om te zetten in radialen. De tangens van een hoek van 30 graden bepaalt u bijvoorbeeld met TAN(DTOR(30)).

Beschrijving

Met TAN() berekent u de verhouding tussen de overliggende rechthoekszijde en de aanliggende rechthoekszijde van een rechthoekige driehoek. TAN() resulteert in een zwevende waarde van 0 tot plus of min oneindig. TAN() resulteert in nul als <*Nuitdr*> gelijk is aan 0, pi of 2*pi radialen. TAN() is ongedefinieerd (oneindig) als <*Nuitdr*> gelijk is aan is pi/2 of 3*pi/2 radialen. Oneindigheid wordt aangegeven met een reeks asterisken.

Het aantal decimalen stelt u in met SET DECIMALS.

De cotangens van een hoek is het complement van de tangens van de hoek. De cotangens van een hoek berekent u met 1/TAN().

Voorbeeld

In de volgende voorbeelden wordt TAN() gebruikt om de tangens van een opgegeven hoek (in radialen) op te halen:

```
SET DECIMALS TO 6
? TAN(PI())              && Resulteert in  0,000000
? TAN(PI()/2)            && Resulteert in oneindig
? TAN(PI()/4)            && Resulteert in  1,000000
? TAN(DTOR(150))         && Resulteert in  -0,577350
? TAN(DTOR(-150))        && Resulteert in  0,577350
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ATAN(), ATN2(), COS(), DTOR(), PI(), RTOD(), SET DECIMALS, SIN()

TARGET()

Tabelindeling

Resulteert in de naam van een tabel die is gekoppeld aan het huidige of opgegeven werkgebied.

Syntaxis

TARGET(<*Nuitdr*> [,<*alias*>])

<*Nuitdr*>

Geeft de positie aan van de relatie in de SET RELATION-lijst voor de huidige of opgegeven tabel.

<*alias*>

Geeft het werkgebied aan vanwaar u een relatie hebt gedefinieerd met het commando SET RELATION. Als u geen werkgebiedalias opgeeft, gebruikt TARGET() het huidige werkgebied. U kunt een werkgebiednummer (1 tot en met 255) of -letter (A tot en met J) of een aliasnaam opgeven. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens.

Beschrijving

TARGET() resulteert in de naam van een tabel die door middel van een met SET RELATION gedefinieerde relatie is gekoppeld aan de tabel in het huidige of een opgegeven werkgebied. Als u geen alias opgeeft, wordt aangenomen dat de relatie is ingesteld vanuit het huidige werkgebied. TARGET() resulteert in een lege tekenreeks ("") als geen relatie is ingesteld op positie <*Nuitdr*> in de SET RELATION-lijst.

Voorbeeld

In het volgende voorbeeld wordt TARGET() gebruikt om de namen te bepalen van de subtabellen die zijn gekoppeld aan Bedrijf:

```
CLOSE DATABASE
USE Contact EXCLUSIVE
INDEX ON CompCode TAG CompCode
SELECT 2
USE Bedrtype EXCLUSIVE
SELECT Bedrtype
INDEX ON Type TAG Type
SELECT 3
USE Bedrijf
SET RELATION TO CompCode INTO Contact
SET RELATION TO Type     INTO Bedrtype ADDITIVE
?  "RELATION:",RELATION(1) , "TARGET:",TARGET(1)
* CompCode CONTACT tonen
?  "RELATION:",RELATION(2) , "TARGET:",TARGET(2)
* CompCode TYPECO tonen
select 20
? "TARGET:",TARGET(1,"Bedrijf")   && geldig vanuit elk werkgebied
* CONTACT tonen
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CREATE, CREATE VIEW...FROM ENVIRONMENT, DISPLAY STATUS, RELATION(), SET(), SET RELATION, SET VIEW

TEXT

Invoer/uitvoer

Toont de regels van TEXT tot ENDTEXT als een blok tekst. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows DEFINE met de klasse Text om tekst weer te geven in formulieren.

Zie Help voor meer informatie over de syntaxis van TEXT. Zie het hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het maken van formulieren.

TIME()

Datum- en tijdgegevens

Resulteert in de systeemtijd als tekenreeks met de opmaak UU:MM:SS of UU:MM:SS.hh format.

Syntaxis

TIME([<*uitdr*>])

<uitdr>

Een willekeurige geldige uitdrukking. De uitdrukking zorgt dat TIME() resulteert in de huidige tijd tot op honderdsten van een seconde nauwkeurig.

Beschrijving

TIME() resulteert in de systeemtijd in de vorm van een tekenuitdrukking. Als u geen uitdrukking doorgeeft aan TIME(), wordt de systeemtijd als resultaat gegeven met de opmaak UU:MM:SS, waarbij UU voor de uren staat, MM voor de minuten en SS voor de seconden.

Als u een uitdrukking doorgeeft aan TIME(), wordt de huidige systeemtijd als resultaat gegeven met de opmaak UU:MM:SS.hh, waarbij .hh de honderdsten van een seconden voorstelt. De waarde van de uitdrukking die u doorgeeft aan TIME() heeft alleen tot gevolg dat ook de honderdsten van een seconde als resultaat worden gegeven.

Voorbeeld

Zie het voorbeeld bij SECONDS() voor een voorbeeld met TIME().

Zie ook

ELAPSED(), SET TIME

TOTAL

Tabelindeling

Maakt een tabel waarin totalen worden opgeslagen voor opgegeven numerieke en zwevende velden in records die zijn gegroepeerd op gemeenschappelijke sleutelwaarden.

Syntaxis

TOTAL ON <*sleutel Tuitdr*> TO <*bestandsnaam*> | ? | <*bestandsnaamfilter*>
[[TYPE] PARADOX | DBASE]
[<*bereik*>]
[FOR <*voorwaarde1*>]
[WHILE <*voorwaarde2*>]
[FIELDS <*veldenlijst*>]

<sleutel Tuitdr>

De sleuteluitdrukking van de hoofdindex of de naam van het veld waarop de huidige tabel is gesorteerd.

TO *<bestandsnaam>* | ? | *<bestandsnaamfilter>*

Stuurt de uitvoer naar de dBASE-tabel met de naam <*bestandsnaam*>. Standaard wordt voor <*bestandsnaam*> de extensie .DBF gebruikt en wordt het bestand opgeslagen in de huidige directory. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u de naam opgeeft van het doelbestand en van de directory waar het moet worden opgeslagen.

U kunt ook een tabel maken in een database (gedefinieerd met de IDAPI-configuratieprogramma) door voor de naam van de tabel de naam van de database op te geven (tussen dubbele punten), in de notatie:*databasenaam:tabelnaam*. Als de database nog niet is geopend, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

[TYPE] PARADOX | DBASE

Geeft aan welk type tabel moet worden gemaakt. Het sleutelwoord TYPE is er alleen voor de leesbaarheid; het heeft geen gevolgen voor de werking van het commando.

Als u PARADOX opgeeft, wordt een Paradox-tabel gemaakt met de extensie .DB.

Als u DBASE opgeeft, wordt een dBASE-tabel gemaakt (de standaardinstelling). Als u geen extensie opgeeft voor <*bestandsnaam*>, wordt de extensie .DBF gebruikt.

<bereik>

Het aantal records dat moet worden opgeteld. RECORD <*n*> geeft een record aan door middel van het recordnummer. NEXT <*n*> geeft *n* records aan, te beginnen bij het huidige record. ALL geeft alle records aan. REST geeft alle records vanaf het huidige record tot het eind van het bestand aan.

FOR <*voorwaarde1*>
WHILE <*voorwaarde2*>

Bepaalt welke records worden beïnvloed door TOTAL. FOR beperkt TOTAL tot records die voldoen aan <*voorwaarde1*>. WHILE begint met het verwerken van het huidige record en gaat telkens door met een volgend record zolang <*voorwaarde2*> waar is.

FIELDS <*veldenlijst*>

Geeft aan van welke numerieke en zwevende velden het totaal moet worden berekend. Als de optie FIELDS achterwege laat, wordt het totaal van alle numerieke en zwevende velden berekend.

Beschrijving

Met TOTAL kunt u de waarden van numerieke velden in een tabel optellen en een tweede tabel maken waarin de uitkomsten worden opgeslagen. De numerieke velden in de tabel met de uitkomsten bevatten totalen voor alle records met dezelfde sleutelwaarde in de oorspronkelijke tabel.

De huidige tabel moet zijn geïndexeerd of gesorteerd op het sleutelveld. Alle records met hetzelfde sleutelveld worden opgeteld tot één record in de tabel met de uitkomsten. Alle numerieke velden in de veldenlijst bevatten totalen. Alle overige velden bevatten gegevens uit het eerste record van een reeks records met dezelfde sleutels.

TOTAL is vergelijkbaar met SUM, maar SUM wordt gebruikt voor geïndexeerde en niet-geïndexeerde tabellen en geeft van elk numeriek veld de som voor alle records als resultaat. SUM maakt geen tweede tabel, maar slaat de uitkomsten op in geheugenvariabelen of in een array.

Voorbeeld

In het volgende voorbeeld wordt TOTAL gebruikt om voor elke regio in de tabel Klanten het totaal te berekenen van de waarden in het veld VerkTotNu:

```
CLOSE DATABASE
USE Klanten EXCLUSIVE
INDEX ON Regio TAG Reg
* Indexeren op Regio
* Tabel moet exclusief zijn geopend
TOTAL ON Regio TO RegioTot
SELECT 2
USE RegioTot
BROWSE FIELDS Regio, VerkTotNu;
  TITLE "Verkooptotalen per regio"
```

RegioTot bevat één record per regio.

Zie ook

AVERAGE, CALCULATE, COUNT, SUM

TRANSFORM( ) Tekenreeksgegevens

Resulteert in een tekenreeks met gegevens in een opgegeven opmaak.

Syntaxis

TRANSFORM(<*uitdr*>, <*veldopmaak Tuitdr*>)

<*uitdr*>

De teken-, numerieke, logische of datumuitdrukking die moet worden opgemaakt.

<*veldopmaak Tuitdr*>

De tekenreeks met de sjabloontekens op basis waarvan <*uitdr*> wordt opgemaakt. De sjabloontekens zijn gelijk aan de tekens die worden gebruikt in de optie PICTURE of FUNCTION van ? en ??, bij het kenmerk Function van de klasse Entryfield, enzovoort. Laat FUNCTION-sjabloontekens die worden gebruikt bij TRANSFORM(), nadrukkelijk vooraf gaan door een @-symbool. Zie Function in Hoofdstuk 8 voor meer informatie over de optie FUNCTION.

Beschrijving

TRANSFORM() resulteert in een uitdrukking in de PICTURE- of FUNCTION-opmaak die u aangeeft met <*veldopmaak Tuitdr*>. Met TRANSFORM() kunt u gegevens opmaken als het commando dat u gebruikt, wel uitdrukkingen accepteert maar niet over een PICTURE- of FUNCTION-optie beschikt. U kunt TRANSFORM() bijvoorbeeld gebruiken om gegevens op te maken in rapport- en etiketformulieren en om de uitvoer van DISPLAY en LIST op te maken. U kunt met deze opmaak tekst uitlijnen en getallen weergeven in wetenschappelijke notatie.

Voorbeeld

In het volgende voorbeeld wordt TRANSFORM() gebruikt om bedragen uit het veld OpenBalans te tonen of af te drukken met komma's en om Balnsdatum op te maken met de Europese datumopmaak (dag-maand-jaar):

```
CLEAR
USE Afnemers EXCLUSIVE
INDEX ON Bedrijf TO Bedr
SET FIELDS TO Bedrijf, Openbalans, Balnsdatum
SCAN
  ? Bedrijf, TRANSFORM(Openbalans,"999,999.99"),;
    TRANSFORM(Balnsdatum,"@E")
* Toont balans met komma's en datums in Europese opmaak
ENDSCAN
CLOSE DATABASES
```

Zie ook

?, ??, CLASS ENTRYFIELD

TRIM( )

Tekenreeksgegevens

Resulteert in een tekenreeks zonder volgspaties.

Syntaxis

TRIM(<*Tuitdr*> | <*memoveld*>)

<Tuitdr> | <memoveld>
De tekenreeks of het memoveld waaruit de volgspaties moeten worden verwijderd.

Beschrijving

TRIM() resulteert in een tekenuitdrukking of memoveld zonder volgspaties. TRIM() resulteert in een reeks van maximaal 32.766 tekens (de maximumlengte van een tekenreeks). TRIM() is equivalent met RTRIM().

Als u TRIM() gebruikt met een memoveld, worden alleen volgspaties aan het eind van de laatste regel verwijderd. Gebruik TRIM() in combinatie met MLINE() om spaties aan het eind van een bepaalde regel te verwijderen.

Met LTRIM() kunt u *voorloopspaties* uit een tekenreeks of memoveld verwijderen.

Voorbeeld

In het volgende voorbeeld wordt TRIM() gebruikt om volgspaties te verwijderen uit tekst in een tekenveld:

```
USE Bedrijf
X = Plaats + Regio + Postcode
? X
* Resulteert in "Nieuwegein          NW            3430 AA"
X=TRIM(Plaats)+", "+TRIM(Regio)+SPACE(2)+Postcode
? X
* Resulteert in "Nieuwegein, NW  3430 AA"
```

Overdraagbaarheid

Het argument <*memoveld*> wordt niet ondersteund in dBASE IV en dBASE III PLUS. In dBASE IV en dBASE III PLUS is het resultaat van TRIM() beperkt tot 254 tekens.

Zie ook

LEFT(), LTRIM(), MLINE(), RIGHT(), STR(), SUBSTR()

TYPE

Stations- en bestandsfuncties

Geeft de inhoud van een ASCII-bestand weer.

Syntaxis

TYPE <*bestandsnaam1*> | ? | <*bestandsnaamfilter1*>
[MORE]
[NUMBER]
[TO FILE <*bestandsnaam2*> | ? | <*bestandsnaamfilter2*>] | [TO PRINTER]

<*bestandsnaam*> | ? | <*bestandsnaamfilter*>

Het bestand waarvan u de inhoud wilt weergeven. Wordt ook wel het bronbestand genoemd. TYPE ? en TYPE <*bestandsnaamfilter*> tonen een dialoogvenster waarin u een bestand kunt kiezen. Als u een bestand zonder pad opgeeft, wordt het bestand in de huidige directory gezocht en vervolgens in het pad dat is opgegeven met SET PATH. U moet de extensie van het bestand opgeven.

MORE

Pauzeert de uitvoer als het commandovenster vol is. Zonder deze optie schuift de inhoud van het commandovenster op tot het eind van het bestand is bereikt.

NUMBER

Plaats voor elke regel in de uitvoer een regelnummer.

TO FILE <*bestandsnaam2*> | ? | <*bestandsnaamfilter*>

Stuurt uitvoer naar het resultatenpaneel van het commandovenster en naar het tekstbestand <*bestandsnaam2*>. Dit bestand wordt wel het doelbestand genoemd. Standaard wordt aan <*bestandsnaam2*> de extensie .TXT toegekend en wordt het bestand opgeslagen in de huidige directory. De opties ? en <*bestandsnaamfilter*> tonen een dialoogvenster waarin u de naam opgeeft van het doelbestand en van de directory waarin het moet worden opgeslagen.

TO PRINTER

Stuurt uitvoer naar de printer en naar het resultatenpaneel van het commandovenster.

Beschrijving

Met TYPE kunt u de inhoud van ASCII-bestanden weergeven, kopiëren en afdrukken.

Als u TYPE TO FILE of TYPE TO PRINTER gebruikt, worden aan het begin van de opgeslagen of afgedrukte uitvoer twee regels toegevoegd. De eerste regel is een lege regel en de tweede bevat het volledige pad naar het bronbestand en de datumstempel van het bronbestand. Als u NUMBER gebruikt, worden deze twee regels niet genummerd. Het nummeren begint met 1 bij de eerste regel uit het bronbestand. Als u MORE opgeeft en de uitvoer annuleert voordat het eind van het bestand is bereikt, verschijnt in het resultatenpaneel van het commandovenster de tekst *** ONDERBROKEN ***. Deze tekst verschijnt niet in uitvoer die naar een bestand of naar de printer wordt gestuurd.

Als SET SAFETY is ingeschakeld (ON) en er bestaat een bestand met dezelfde naam als het doelbestand, verschijnt een dialoogvenster waarin wordt gevraagd of het bestand

moet worden overschreven. Als SET SAFETY is uitgeschakeld (OFF), wordt een bestand met dezelfde naam zonder waarschuwing overschreven.

Voorbeeld

In de volgende voorbeelden wordt TYPE gebruikt:

```
TYPE Resltaat.txt
* Naar het scherm
TYPE Resltaat.txt MORE
* Pauzeren als het scherm vol is
TYPE Mijnbest.prg NUMBER
* Regelnummers weergeven
TYPE Mijnbest.prg NUMBER TO FILE Mijnbst1.prg
* Met regelnummers naar nieuw bestand
TYPE Mijnbest.prg NUMBER TO FILE ?
* Dialoogvenster openen om bestandsnaam op te geven
TYPE Mijnbest.prg TO PRINTER
* Mijnbest.prg afdrukken
```

Overdraagbaarheid

In dBASE III PLUS wordt alleen de optie TO PRINT ondersteund. De opties MORE, ? en *<bestandsnaamfilter>* worden niet ondersteund in dBASE IV. Als u in dBASE IV geen extensie opgeeft voor het doelbestand, wordt niet .TXT gebruikt, maar .PRT.TXT.

Zie ook

COPY FILE, EJECT, SET ALTERNATE, SET PRINTER, SET SAFETY

TYPE()

Uitdrukkingen en gegevenstypeconversie

Resulteert in een tekenreeks die het gegevenstype van een uitdrukking voorstelt.

Syntaxis

TYPE(*<uitdr>* | *<Tuitdr>*)

<uitdr>* | *<Tuitdr>

De uitdrukking waarvan u het gegevenstype als resultaat wilt geven.

Beschrijving

TYPE() kan alleen tekenuitdrukkingen verwerken. U kunt *<uitdr>* (zonder aanhalingstekens) alleen gebruiken als u een geheugenvariabele evalueert die de naam bevat van een databaseveld of een tekenreeks die naar een uitdrukking verwijst. Verder kunt u met *<Tuitdr>* een vaste uitdrukking, een veldnaam of een geheugenvariabele evalueren.

Als in de database bijvoorbeeld een veld "START" voorkomt, kunt u op twee manieren het type van dat veld bepalen. U kunt TYPE("START") gebruiken, of de veldnaam opslaan in een geheugenvariabele (STORE "START" TO gvar) en dan TYPE(gvar) gebruiken.

Een soortgelijk principe geldt voor geheugenvariabelen die geen veldnamen bevatten:

- Als u de instructie STORE "1+2=5" TO gvar gebruikt, resulteert TYPE(gvar) in L. TYPE(gvar) wordt in dit geval geïnterpreteerd als "Welk type uitdrukking stelt de tekenreeks in gvar voor?" De tekenreeks in gvar is "1+2=5" en dat is een logische uitdrukking, dus wordt L als resultaat gegeven.
- Als u de instructie STORE "1+2=5" TO gvar gebruikt, resulteert TYPE("gvar") echter in C. In dit geval wordt TYPE("gvar") geïnterpreteerd als "Welk gegevenstype bevat de variabele gvar?" Omdat "1+2=5" een tekenreeks is, wordt nu C als resultaat gegeven.

Geheugenvariabelen kunnen ook andere geheugenvariabelen opleveren. Als u bijvoorbeeld de instructies STORE 10 TO gnum en STORE "gnum" TO gvar geeft, resulteert TYPE(gvar) in N. Ook hier wordt TYPE(gvar) geïnterpreteerd als "Welk type uitdrukking stelt de tekenreeks in gvar voor?" De tekenreeks in gvar is "gnum". Die tekenreeks stelt een numerieke variabele voor en dus wordt N als resultaat gegeven.

Als u een geheugenvariabele wilt evalueren die geen tekenreeks voorstelt, moet u *<Tuitdr>* gebruiken, zoals in het volgende voorbeeld:

```
STORE DATE() TO gvar
? TYPE(gvar)   && resulteert in een fout omdat
               && gvar geen tekenreeks bevat.
? TYPE("gvar") && resulteert in D
```

TYPE() resulteert altijd in een tekenreeks. De volgende tabel geeft een overzicht van de resultaatwaarden van TYPE().

Gegevenstype van *<uitdr>* | *<Tuitdr>*	**Resultaat van TYPE()**
Array-variabele (dBASE)	A
Binair veld (dBASE of Paradox, BLOB)	B
Bladwijzervariabele (dBASE)	BM
Tekenveld of tekenreeksvariabele (dBASE), alfanumeriek veld (Paradox)	C
Codeblokvariabele (dBASE)	CB
Datumveld of -variabele (dBASE), datumveld (Paradox)	D
Zwevend veld (dBASE), numeriek of valutaveld (Paradox)	F
Functie-aanwijzervariabele (dBASE)	FP
OLE (algemeen)	G
Logisch veld of logische variabele (dBASE)	L
Memoveld (dBASE of Paradox)	M
Numeriek veld of numerieke variabele (dBASE)	N
Objectvariabele (dBASE)	O
Schermvariabele (dBASE)	S
Ongedefinieerde variabele, ongedefinieerd veld of ongeldige uitdrukking (dBASE of Paradox)	U

Voorbeeld

In de volgende voorbeelden wordt TYPE() gebruikt om het gegevenstype van een opgegeven uitdrukking te bepalen.

```
USE Afnemers
? TYPE("Bedrijf")          && Resulteert in C
? TYPE("BalnsDatum")       && Resulteert in D
? TYPE("Openbalans")       && Resulteert in N
gDatum={31-12-99}          && Variabele van type datum
? TYPE("gDatum")           && Resulteert in D
? TYPE(gDatum)             && Resulteert in foutmelding
gVeld = "Openbalans"       && Naam van een veld
? TYPE(gVeld)              && Resulteert in N
gStartbal=Openbalans       && Inhoud van een veld
? TYPE(gStartbal)          && Resulteert in foutmelding
? TYPE("gStartbal")        && Resulteert in F
```

Zie ook

EMPTY(), ISBLANK(), STORE

UNIQUE() — Tabelindeling

Geeft aan of de opgegeven index is gemaakt met het sleutelwoord UNIQUE (of met SET UNIQUE ON).

Syntaxis

UNIQUE([[*<.mdx-bestandsnaam Tuitdr>*,] *<indexpositie Nuitdr>* [,*<alias>*]])

<.mdx-bestandsnaam Tuitdr>

Geeft een meervoudig indexbestand aan dat de indexlabel bevat die u wilt controleren.

<indexpositie Nuitdr>

Selecteert een indexbestand of -label door middel van de positie van een indexlabel in een .MDX-bestand of de positie van een indexbestand in de lijst met geopende indexen voor de huidige of een opgegeven tabel. Het positienummer van de index geeft de positie van de index aan in de lijst met indexen die zijn geopend met SET INDEX of USE.

<alias>

Een werkgebiednummer (van 1 tot 255) of -letter (van A tot J) of een aliasnaam. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens.

Beschrijving

De functie UNIQUE() resulteert in .T. als de index die wordt aangegeven door de optionele parameter *<indexpositie Nuitdr>* is gemaakt met de optie INDEX...UNIQUE, of met INDEX terwijl SET UNIQUE was ingeschakeld (ON). Als u geen indexnummer

opgeeft, wordt de hoofdindex of -indexlabel voor het huidige of opgegeven werkgebied gecontroleerd.

De functie UNIQUE() resulteert in de volgende situaties in .F.:

- De hoofdindex of het opgegeven indexnummer of indexlabelnummer is niet gemaakt met de optie INDEX...UNIQUE of met INDEX terwijl SET UNIQUE was ingeschakeld (ON).
- U hebt geen indexnummer opgegeven en de huidige tabel beschikt niet over een hoofdindex.
- In het huidige of een opgegeven werkgebied bestaat geen geopende index of indexlabel met het opgegeven indexnummer.
- UNIQUE() resulteert in een fout als de opgegeven index of bestandsnaam niet bestaat.

Voorbeeld

In het volgende voorbeeld wordt UNIQUE() gebruikt om te bepalen of een opgegeven index is gemaakt met UNIQUE:

```
USE Bedrijf EXCLUSIVE
INDEX ON CompCode TAG CompCode
SET ORDER TO TAG CompCode
? TAG(), "Unieke index = ", UNIQUE()
INDEX ON Regio TAG Reg OF Lokatie UNIQUE
SET ORDER TO TAG Reg OF Lokatie
? TAG(), "Unieke index = ", UNIQUE()
? TAG(1), "Unieke index = ", UNIQUE(1), MDX(1)
* UNIQUE kan naar een index verwijzen dmv positie
* Dit is de eerste index van Bedrijf.mdx
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

DESCENDING(), FOR(), INDEX, KEY(), MDX(), NDX(), ORDER(), SET UNIQUE, TAG(), TAGCOUNT(), TAGNO(), WORKAREA()

UNLOCK

Gedeelde gegevens

Heft de vergrendeling van de huidige tabel op als u die hebt vergrendeld met FLOCK(). Heft de vergrendeling op van alle records in de huidige tabel die u hebt vergrendeld met RLOCK() of LOCK().

Syntaxis

UNLOCK
[ALL | IN <*alias*>]

ALL

Heft in alle werkgebieden de vergrendelingen op van tabellen die u hebt vergrendeld met FLOCK(), en van alle records die u hebt vergrendeld met RLOCK() of LOCK().

IN <*alias*>

Heft de vergrendeling op van de aliastabel <*alias*> als u die hebt vergrendeld met FLOCK(), of heft de vergrendeling op van alle records in die tabel die u hebt vergrendeld met RLOCK() of LOCK(). <*alias*> is een werkgebiednummer (van 1 tot 255), -letter (van A tot J) of een aliasnaam. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens. Als u geen <*alias*> opgeeft, heft UNLOCK de vergrendeling op van de huidige tabel.

Beschrijving

Met UNLOCK kunt u bestandsvergrendelingen opheffen die zijn ingesteld met FLOCK(), of recordvergrendelingen opheffen die zijn ingesteld met RLOCK() of LOCK(). UNLOCK is alleen geldig op het werkstation waar ook de instructies met FLOCK(), RLOCK() en LOCK() zijn gegeven. UNLOCK kan geen vergrendelingen opheffen die zijn aangebracht door andere werkstations.

Als u door een tabel bladert of een tabel wijzigt, en de recordaanwijzer staat bij een record of in een invoerveld dat een veld in de tabel voorstelt, kunt u op *Ctrl-O* drukken om de vergrendeling voor het huidige record in te schakelen of op te heffen. Dat wil zeggen, als het record vergrendeld is, kunt u de vergrendeling opheffen met *Ctrl-O* en als het record niet vergrendeld is, kunt u het vergrendelen met *Ctrl-O*.

Als u met SET RELATION een relatie instelt met een hoofdtabel en vervolgens de vergrendeling van de hoofdtabel of van records in de hoofdtabel opheft met UNLOCK, wordt ook de vergrendeling van gerelateerde tabellen of records opgeheven. Zie SET RELATION voor meer informatie over het instellen van relaties tussen tabellen.

Voorbeeld

Zie FLOCK() voor een voorbeeld met UNLOCK.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

FLOCK(), RLOCK(), SET RELATION

UPDATE

Tabelindeling

Vervangt gegevens in opgegeven velden in de huidige tabel door gegevens uit een andere tabel.

Syntaxis

UPDATE ON <*sleuteluitdrukking*> FROM <*alias*>
  REPLACE <*veld1*> WITH <*uitdr1*>
    [, <*veld2*> WITH <*uitdr2*> ...]
  [RANDOM]
  [REINDEX]

<sleuteluitdrukking>

De sleuteluitdrukking die de huidige tabel en een tabel in een ander werkgebied gemeenschappelijk hebben.

FROM *<alias>*

Geeft een tabel aan in een ander werkgebied die de nieuwe gegevens voor de huidige tabel bevat.

REPLACE *<veld1>* WITH *<uitdr1>*

Geeft het veld in de huidige tabel aan dat moet worden bijgewerkt met gegevens uit de tabel die wordt aangegeven door FROM <*alias*>. Als een bij WITH <*uitdr1*> opgegeven veld zich in een ander werkgebied dan het werkgebied van de huidige tabel bevindt, kunt u dat veld aanduiden met de alias, in de notatie *alias->veld*.

[,*<veld n>* WITH *<uitdr n>* ...]

Aanvullende velden om bij te werken.

RANDOM

Geeft aan dat de tabel waaruit de gegevens afkomstig zijn, niet is geïndexeerd of gesorteerd. (De huidige tabel moet geïndexeerd zijn op de sleuteluitdrukking die beide tabellen gemeenschappelijk hebben.)

REINDEX

Geeft aan dat alle beïnvloede niet-hoofdindexen opnieuw moeten worden samengesteld als de bewerking is voltooid.

Beschrijving

Het commando UPDATE gebruikt gegevens uit een opgegeven tabel om waarden in de huidige tabel te vervangen. De wijzigingen worden aangebracht door te zoeken naar overeenkomstige records op basis van een gemeenschappelijk sleutelveld.

De huidige tabel moet zijn geïndexeerd op het veld in de sleuteluitdrukking. Tenzij de optie RANDOM is gebruikt, moet ook de tabel in het opgegeven werkgebied op

hetzelfde veld zijn geïndexeerd of gesorteerd. Velden die in de sleuteluitdrukking zijn opgenomen moeten in beide tabellen dezelfde naam hebben.

Voorbeeld

In het volgende voorbeeld wordt UPDATE gebruikt om de totalen voor elke order in de tabel Orders opnieuw te berekenen op basis van de tabel Regels. UPDATE kan niet meerdere records tegelijk zoeken, dus worden totalen berekend van de tabel Regels op basis van Ordernr. Deze totalen worden opgeslagen in een nieuwe tabel, RegelTot, en Orders worden bijgewerkt vanuit deze tabel:

```
SET TALK ON
* Aantallen records worden getoond
CLOSE ALL
USE ORDERS EXCLUSIVE
SELECT 2
USE Regels EXCLUSIVE
INDEX ON Ordernr TAG Orders
TOTAL ON Ordernr TO RegelTot
USE RegelTot
SELECT Orders
UPDATE ON Ordernr FROM RegelTot;
  REPLACE BetBedrag WITH RegelTot->Aantal*Regeltot->Verkprijs
* Alleen orders met regels worden bijgewerkt.
* Vanwege SET TALK ON kunt u zien hoeveel
* records zijn bijgewerkt.
```

Zie ook

APPEND FROM, JOIN, REPLACE, SELECT, SET RELATION

UPDATED()

Invoer/uitvoer

Resulteert in .T. (waar) als u de inhoud van @...GET-velden of geheugenvariabelen in het resultatenpaneel van het commandovenster of het huidige dBASE IV-venster hebt gewijzigd. Dit commando wordt hoofdzakelijk ondersteund ten behoeve van dBASE IV-vensters. Gebruik in dBASE voor Windows kenmerken als OnChange om gewijzigde informatie in formulieren te verwerken.

Zie Help voor meer informatie over de syntaxis van UPDATED(). Zie het hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het maken van formulieren.

UPPER()

Tekenreeksgegevens

Zet alle kleine letters in een tekenreeks om in hoofdletters en resulteert in de geconverteerde tekenreeks.

Syntaxis

UPPER(<*Tuitdr*> | <*memoveld*>)

<*Tuitdr*> | <*memoveld*>

De tekenreeks of het memoveld dat moet worden omgezet in hoofdletters.

Beschrijving

UPPER() zet kleine letters in een tekenuitdrukking of memoveld om in hoofdletters. Cijfers en andere tekens worden genegeerd. UPPER() resulteert in een reeks van maximaal 32.766 tekens (de maximumlengte van een tekenreeks).

De huidige taalaansturing bepaalt welke tekens worden beschouwd als hoofdletters en welke als kleine letters. Tevens bepaalt de taalaansturing in welke hoofdletters kleine letters met accenten worden omgezet, en omgekeerd. In de Nederlandse taalaansturing wordt á (a-accent grave) bijvoorbeeld omgezet in Á, maar â (a-circonflex) in de gewone hoofdletter A. Zie Appendix C in *Programmeren* voor een volledige beschrijving van de omzettigsregels die gelden in de Nederlandse taalaansturing.

Voorbeeld

In het volgende voorbeeld wordt UPPER() gebruikt om bij het vergelijken van twee tekenreeksen tekst in kleine letters om te zetten in hoofdletters:

```
? UPPER("Techniek")          && Resulteert in "TECHNIEK"
? UPPER("Techniek") = "Techniek";
                             && Resulteert in .F.
? UPPER("")                  && Resulteert in ""
? UPPER("12 appels")         && Resulteert in "12 APPELS"
```

UPPER() wordt veel gebruikt bij het vergelijken van tekengegevens. In het volgende voorbeeld wordt UPPER() gebruikt om te zorgen dat bij een bewerking met SEEK een overeenkomst wordt gevonden:

```
SET EXACT OFF
USE Dieren EXCLUSIVE
INDEX ON UPPER(Naam) TAG Naam
gZoeken="Boa"
SEEK UPPER(gZoeken)
IF FOUND()
    EDIT
ENDIF
RETURN
```

Overdraagbaarheid

Het argument <*memoveld*> wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CHARSET(), ISALPHA(), ISLOWER(), ISUPPER(), LDRIVER(), LOWER(), PROPER(), SET LDCHECK

USE

Opent de opgegeven tabel plus alle bijbehorende memo- (.DBT) en produktie-MDX-bestanden, indien aanwezig.

Syntaxis

USE
[<*bestandsnaam1*> | ? | <*bestandsnaamfilter1*>
[[TYPE] PARADOX | DBASE]
[IN <*alias*>]
[INDEX <*indexnamenlijst*> |
<*? lijst*> | <*indexnaamfilterslijst*>]
[ORDER [TAG] <*.ndx-bestandsnaam*> |
<*labelnaam*> [OF <*.mdx-bestandsnaam*>]]
[AGAIN]
[ALIAS <*alias*>]
[AUTOMEM]
[EXCLUSIVE | SHARED]
[NOSAVE]
[NOUPDATE]]

<*bestandsnaam1*> | ? | <*bestandsnaamfilter1*>

Het tabelbestand dat u wilt openen. USE ? en USE <*bestandsnaamfilter1*> tonen een dialoogvenster waarin u een tabelbestand kunt selecteren. Als u een bestand zonder pad opgeeft, wordt dit gezocht in de huidige directory en vervolgens in het pad dat is ingesteld met SET PATH. Als u een bestand zonder extensie opgeeft en geen type opgeeft, wordt het bestandstype gebruikt dat is ingesteld met het commando SET DBTYPE.

U kunt ook een tabel openen in een database (gedefinieerd met de IDAPI-configuratieprogramma) door voor de naam van de tabel de naam van de database op te geven (tussen dubbele punten), in de notatie *:databasenaam:tabelnaam*. Als de database nog niet open is, verschijnt een dialoogvenster waarin u de parameters opgeeft, zoals een aanmeldingsnaam en wachtwoord, die nodig zijn om een verbinding met de database tot stand te brengen.

[[TYPE] PARADOX | DBASE]

Geeft het type van de tabel aan. Het sleutelwoord TYPE is er alleen voor de leesbaarheid; het heeft geen gevolgen voor de werking van het commando.

Als u PARADOX opgeeft, wordt een Paradox-tabel met de extensie .DB geopend.

Als u DBASE opgeeft, wordt een dBASE-tabel (de standaardinstelling) geopend. Als u geen extensie opgeeft voor <*bestandsnaam*>, wordt de extensie .DBF gebruikt.

IN <*alias*>
Geeft een werkgebied aan. U kunt een werkgebiednummer (1 tot en met 255) of -letter (A tot en met J) of een aliasnaam opgeven. Plaats de werkgebiedletter of aliasnaam tussen aanhalingstekens.

INDEX <*indexnamenlijst*> | <*? lijst*> | <*indexnaamfilterslijst*>
Is alleen geldig voor dBASE-indexen. (Indexen voor Paradox- en SQL-tabellen worden opgegeven in de ORDER-clausule.) Opent maximaal 100 afzonderlijke indexbestanden voor de opgegeven tabel. <*indexnamenlijst*> kan de namen van .NDX- en .MDX-indexbestanden bevatten. Als het eerste bestand in <*indexnamenlijst*> een .NDX-bestand is, en u maakt geen gebruik van de optie ORDER, wordt dat indexbestand de hoofdindex. INDEX <*? lijst*> en INDEX <*indexnaamfilterslijst*> tonen een dialoogvenster waarin u een bestaand indexbestand kunt selecteren.

ORDER [TAG] <*labelnaam*>
Maakt het indexbestand <*labelnaam*> de hoofdindex. Als u een Paradox-tabel opent, kunt u een secundaire index opgeven als hoofdindex. Als u dit niet doet, wordt de PRIMARY-index gebruikt. Als u een SQL-tabel opent, kunt u ook een index opgeven als hoofdindex.

Als u geen ORDER-clausule gebruikt en de eerste bestandsnaam na INDEX is een .NDX-bestand, wordt dat bestand de hoofdindex. Als u geen ORDER-clausule gebruikt en de eerste bestandsnaam na INDEX is een .MDX-bestand, wordt geen hoofdindex gebruikt voor de tabel.

OF <*.mdx-bestandsnaam*>
Het .MDX-bestand waarin <*labelnaam*> staat. Zonder OF <*bestandsnaam*> wordt <*labelnaam*> gezocht in een .MDX-bestand met dezelfde bestandsnaam als de tabel.

ORDER [TAG] <*.ndx-bestandsnaam*>
Maakt <*.ndx-bestandsnaam*> de hoofdindex.

AGAIN
Opent een tabel en de bijbehorende indexbestanden in het huidige of een opgegeven werkgebied, en laat de tabel open in een of meer andere werkgebieden.

ALIAS <*alias*>
Geeft een alternatieve aliasnaam aan voor het huidige werkgebied. De standaardaliasnaam is de tabelnaam, tenzij u de optie AGAIN hebt gebruikt.

AUTOMEM
Initialiseert voor elk veld in de opgegeven tabel een geheugenvariabele (maar niet voor memo-, binaire en OLE-velden). De geheugenvariabelen krijgen dezelfde naam en hetzelfde type als de velden.

EXCLUSIVE | SHARED

Met EXCLUSIVE opent u de tabel exclusief zodat andere gebruikers geen toegang tot de tabel kunnen krijgen. Met SHARED hebben andere gebruikers wel toegang tot de tabel. De standaardinstelling is afhankelijk van het feit of u werkt in een gemeenschappelijke omgeving, of van de instelling voor LOCALSHARE in een omgeving voor individueel gebruik.

NOSAVE

Wordt gebruikt om een .DBF-bestand te openen als een tijdelijke tabel. Als u een bestand sluit dat is geopend met NOSAVE, wordt het samen met het bijbehorende memobestand verwijderd. Als u per ongeluk een tabel hebt geopend met de optie NOSAVE, kunt u de gegevens alsnog opslaan met COPY TO.

NOUPDATE

Voorkomt dat gebruikers records in de tabel wijzigen, verwijderen of herstellen.

Beschrijving

Met het commando USE opent u een bestaande tabel, produktie-MDX-bestand en bijbehorend .DBT-bestand (als de tabel binaire, memo- of OLE-velden bevat). U kunt met USE eventueel ook bijbehorende .NDX-bestanden of andere .MDX-indexbestanden dan het produktie-MDX-bestand openen. U kunt alleen toegang krijgen tot gegevens in een tabel als de tabel is geopend.

USE zonder argumenten sluit de geopende tabel, indexen en het indelingsbestand in het huidige werkgebied. USE IN <alias> zonder bestandsnaamargument sluit de tabel en alle bijbehorende bestanden in het opgegeven werkgebied. CLOSE TABLES sluit alle bestanden in alle werkgebieden.

U kunt een tabel openen in elk beschikbaar werkgebied. Als u een tabel opent, kunt u in de USE-instructie het werkgebied een naam geven met de optie ALIAS. Voor aliasnamen gelden dezelfde regels als voor bestandsnamen. Aliassen worden gebruikt in verwijzingen naar tabellen in andere werkgebieden.

USE...INDEX geeft indexbestanden aan die geopend en bijgehouden moeten worden voor een bepaalde tabel. Met de optie ORDER geeft u de hoofdindex aan uit een lijst van indexen die zijn geopend met de optie INDEX en de produktie-MDX-index. USE...INDEX is gelijk aan USE gevolgd door SET INDEX. Zie SET INDEX en SET ORDER voor een uitleg van de volgorde van de geopende indexen en het opgeven van een hoofdindex.

U kunt bij de optie INDEX zowel .NDX- als .MDX-indexbestanden opgeven. Als een tabel beschikt over een .NDX-bestand en een .MDX-bestand met dezelfde naam, worden de indexen in het .MDX-indexbestand geopend. Als u de .NDX-index wilt openen, geeft u de volledige naam (inclusief extensie) op.

Met de optie NOSAVE kunt u een tabel openen als een tijdelijk bestand. De tabel en de bijbehorende memo- en indexbestanden worden automatisch gewist als u de tabel sluit.

Met de optie AUTOMEM kunt u geheugenvariabelen maken voor alle velden in de tabel (met uitzondering van binaire, memo- en OLE-velden). Deze geheugenvariabelen worden als volgt aan de hand van het gegevenstype in de velden geïnitialiseerd:

Teken	Spaties
Zwevend	0
Numeriek	0
Logisch	.F.
Datum	/ /

Met USE...AUTOMEM kunt u geheugenvariabelen maken die overeenkomen met velden in een tabel zodat u in een programma gegevenswaarden kunt bijwerken door middel van variabelen in plaats van rechtstreeks de veldwaarden te bewerken. Zie APPEND AUTOMEM en STORE AUTOMEM voor meer informatie over het werken met automatische geheugenvariabelen.

Voorbeeld

In het volgende voorbeeld wordt USE gebruikt om de tabellen Vluchten en Vliegtg te openen. De tabellen worden gesorteerd op een .MDX-label en geopend in de volgende twee beschikbare werkgebieden:

```
CLOSE DATABASES
USE Vluchten ORDER Vliegtuig EXCLUSIVE IN SELECT()
USE Vliegtg IN SELECT()
SELECT Vliegtg
SET RELATION TO Vliegtuig INTO Vluchten
```

In het volgende voorbeeld wordt de tabel Klanten geopend en worden automatisch geheugenvariabelen gemaakt met de veldwaarden:

```
SELECT 1
USE Klanten ORDER Naam AUTOMEM EXCLUSIVE
DISPLAY MEMORY TO PRINT
* Toont geheugenvariabelen gemaakt met AUTOMEM
* uit tabel Klanten.
CLOSE DATABASES
```

In het volgende voorbeeld wordt een Paradox-tabel geopend:

```
USE Klanten TYPE PARADOX
BROWSE
CLOSE DATABASES
```

Zie ook

ALIAS(), APPEND AUTOMEM, CLEAR AUTOMEM, CLOSE..., SELECT, SELECT(), SET INDEX, SET ORDER, STORE AUTOMEM

VAL() — Uitdrukkingen en gegevenstypeconversie

Resulteert in een tekenreeks in de vorm van een numerieke of zwevende waarde.

Syntaxis

VAL(<*Tuitdr*>)

<Tuitdr>

De tekenuitdrukking die u wilt omzetten in een numerieke of zwevende waarde, of de tekenuitdrukking die *begint* met het getal dat u wilt omzetten in een numerieke of zwevende waarde.

Beschrijving

Met VAL() kunt u tekenuitdrukkingen omzetten in getallen van het type numeriek of zwevend. Als u tekengegevens hebt omgezet in numerieke of zwevende gegevens, kunt u berekeningen uitvoeren met de waarden.

Als de tekenreeks die u opgeeft zowel letters als cijfers bevat, resulteert VAL() in de waarde van het volledige getal dat links van het eerste niet-numerieke teken staat. Als de tekenreeks begint met een ander niet-numeriek teken dan een spatie, resulteert VAL() in 0. De instructie VAL("ABC123ABC456") resulteert bijvoorbeeld in 0, VAL("123ABC456ABC") in 123 en VAL(" 123") ook in 123.

Voorbeeld

In de volgende voorbeelden wordt VAL() gebruikt om de numerieke waarde van een tekenreeks, tekenvariabele en tekenveld op te halen:

```
SET TALK OFF
? VAL("123AB34")              && Resulteert in 123
? VAL("-32")                  && Resulteert in -32
Treeks1 = "LON672"
? VAL(Treeks1)                 && Resulteert in 0
Treeks2 = "8 bits in een byte"
? VAL(Treeks2)                 && Resulteert in 8

**VldDemo.Prg**
Treeks2 = "4358 LW"
USE Klanten ORDER Klantnr
SEEK "1560"
IF FOUND()
   REPLACE POSTCODE WITH Treeks2
ENDIF
? VAL(Postcode)               && Resulteert in 4358
SET TALK ON
CLOSE ALL
** End VldDemo.PRG**
```

In het volgende voorbeeld wordt VAL() gebruikt om de tekengegevens in het veld Zipcode om te zetten naar een getal zodat in de tabel Afnemers alle postcodes boven 80000 kunnen worden uitgefilterd:

```
SET TALK OFF
USE Afnemers EXCLUSIVE
INDEX ON Zipcode TAG ZpCo
SET FILTER TO VAL(Zipcode)>80000
GO TOP
```

```
? CENTER("Zipcodelijst")
? CENTER("Zipcode boven 80000")
?
SCAN
  ? Bedrijf, Contact, Telefoon, Zipcode
ENDSCAN
CLOSE ALL
SET TALK ON
RETURN
```

Zie ook

SET DECIMALS, SET POINT, SET SEPARATOR, STR()

VALIDDRIVE( )

Stations- en bestandsfuncties

Resulteert in .T. als resultaat als het opgegeven station bestaat en kan worden gelezen. Resulteert in .F. als het station niet bestaat of niet kan worden gelezen.

Syntaxis

VALIDDRIVE(<*station Tuitdr*>)

<station Tuitdr>

De letter voor het station dat u wilt controleren, eventueel gevolgd door een dubbele punt. Als u meerdere letters opgeeft voor <*station Tuitdr*>, wordt alleen het station gecontroleerd dat wordt voorgesteld door de eerste letter.

Beschrijving

Met VALIDDRIVE() kunt u controleren of een opgegeven station bestaat en beschikbaar is voordat u CD, SET DEFAULT, SET DIRECTORY of SET PATH gebruikt. VALIDDRIVE() is ook goed bruikbaar als een programma bestanden kopieert van of naar een station, of directorypaden inclusief stationsletters bevat.

VALIDDRIVE() kan elk opgegeven station controleren, inclusief netwerkstations.

Voorbeeld

In de volgende voorbeelden wordt VALIDDRIVE() gebruikt:

```
? VALIDDRIVE("C:")  && .T.
? VALIDDRIVE("A:")  && .T. als station A: een diskette bevat
? VALIDDRIVE("Z:")  && .T. als station Z: bestaat
```

In het volgende voorbeeld controleert VALIDDRIVE() het diskettestation B:. Als VALIDDRIVE() resulteert in .F., wordt de gebruiker gevraagd een diskette in het station te plaatsen. Het station wordt maximaal drie keer gecontroleerd:

```
K=1                              && teller instellen
DO WHILE .NOT. VALIDDRIVE("B:") .AND. K<=3
  WAIT "Plaats diskette in B: en druk op een toets"
```

```
  K=K+1                          && teller verhogen
ENDDO
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CD, SET DEFAULT, SET DIRECTORY, SET PATH

VARREAD() Invoer/uitvoer

Resulteert in de naam van de huidige geheugenvariabele of het huidige veld als u READ opgeeft na @...GET. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows het kenmerk Name om een object in een formulier te identificeren.

Zie Help voor meer informatie over de syntaxis van VARREAD(). Zie het hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over werken met formulieren.

VERSION() Omgeving

Geeft een tekenreeks als resultaat die de naam en het versienummer van de huidige versie van dBASE bevat.

Syntaxis

VERSION()

Beschrijving

U kunt VERSION() gebruiken in programma's die mogelijkheden van verschillende versies van dBASE gebruiken. U kunt met VERSION() vaststellen welke versie van dBASE wordt uitgevoerd en op basis van het resultaat bepaalde delen van de code wel of niet uitvoeren.

Voorbeeld

In het volgende voorbeeld wordt getest of de gebruiker met dBASE voor Windows werkt:

```
IF "WINDOW"$UPPER(VERSION())
* Maakt "WINDOW" deel uit van de versienaam?
  ? "dBASE voor Windows actief"  && Ja
ELSE
  * Geen Windows-versie van dBASE
  WAIT "Dit programma kan alleen in "+;
```

```
    "dBASE voor Windows worden uitgevoerd"
ENDIF
```

Zie ook

OS()

WAIT

Invoer/uitvoer

Pauzeert het huidige programma en zet de uitvoering voort als op een toets wordt gedrukt. Optioneel accepteert het commando één teken als invoer. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. In dBASE voor Windows kunt u WAIT alleen gebruiken als u uitvoer naar het resultatenpaneel van het commandovenster stuurt en op bepaalde punten even wilt pauzeren.

Syntaxis

WAIT
[<*prompt Tuitdr*>]
[TO <*variabele*>]

<*prompt Tuitdr*>

Een tekenuitdrukking (aanwijzing) die de gebruiker om invoer vraagt.

TO <*variabele*>

Kent één teken toe aan de geheugenvariabele <*variabele*>. Als <*variabele*> niet bestaat, wordt deze door dBASE gemaakt. Als <*variabele*> bestaat, wordt deze overschreven.

Standaardinstelling

Als u geen <*prompt Tuitdr*> opgeeft, verschijnt de aanwijzing "Willekeurige toets indrukken om door te gaan" als u WAIT gebruikt.

Beschrijving

Met WAIT kunt u de uitvoering van een programma tijdelijk onderbreken (bijvoorbeeld om gegevens te tonen zonder de gebruiker toe te staan deze te wijzigen). Druk op een willekeurige toets om WAIT te beëindigen en het programma verder uit te voeren.

Opmerking Als SET ESCAPE is ingeschakeld (ON), wordt de uitvoering van het programma geannuleerd als de gebruiker op *Esc* drukt tijdens een WAIT-pauze. Als SET ESCAPE is uitgeschakeld (OFF), wordt de uitvoering van het programma voortgezet als de gebruiker tijdens een WAIT-pauze op *Esc* of een andere toets drukt.

Hoewel WAIT hoofdzakelijk wordt gebruikt om een programma tijdelijk te onderbreken tot de gebruiker op een toets drukt, kunt u het commando ook gebruiken om de toetsaanslag op te slaan in een tekengeheugenvariabele. Als de gebruiker op *Enter*

drukt zonder andere tekens te typen, wordt aan *<variabele>* een lege tekenreeks ("") toegekend.

Voorbeeld

WAIT kan worden gebruikt zonder aanwijzing of invoervariabele:

```
WAIT
```

De standaardaanwijzing verschijnt op de volgende regel in kolom 0:

```
* Druk op een toets om door te gaan...
```

In het volgende voorbeeld wordt WAIT met een aanwijzing gebruikt:

```
WAIT "Druk op een toets om naar het volgende scherm te gaan"
```

In dit voorbeeld wordt WAIT gebruikt met een aanwijzing en een invoervariabele:

```
WAIT;
  "Wilt u het rapport afdrukken? (J/N) ";
   to Antwoord
```

Zie ook

SET ESCAPE

WINDOW()

dBASE IV-vensters

Resulteert in de naam van het actieve dBASE IV-venster. Dit commando wordt hoofdzakelijk ondersteund vanwege compatibiliteit met dBASE IV. Gebruik in dBASE voor Windows INSPECT( ) om informatie over formulieren op te halen.

Zie Help voor meer informatie over de syntaxis van WINDOW(). Zie het hoofdstuk "Formulieren maken" in het *Handboek* voor meer informatie over het definiëren van formulieren.

WORKAREA()

Tabellen

Resulteert in het nummer van het huidige actieve werkgebied.

Syntaxis

WORKAREA()

Beschrijving

De functie WORKAREA() resulteert in het nummer van het huidige actieve werkgebied. In een programma kunt u WORKAREA() gebruiken om het nummer van het huidige werkgebied op te slaan zodat u op een later tijdstip naar dat werkgebied kunt terugkeren met het commando SELECT.

Voorbeeld
Zie DBF() en SELECT() voor voorbeelden met WORKAREA().

Overdraagbaarheid
Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook
DBF(), SELECT, SELECT()

YEAR() — Datum- en tijdgegevens

Resulteert in het jaar in een opgegeven datumuitdrukking in de vorm van een getal van vier cijfers.

Syntaxis
YEAR(<*Duitdr*>)

<*Duitdr*>
De datumuitdrukking, in de huidige datumopmaak, waarvan u het jaartal wilt weten.

Beschrijving
YEAR() resulteert in het jaartal van een datum in de vorm van vier cijfers. De instelling voor SET CENTURY is niet van invloed op YEAR().

Geef <*Duitdr*> op in de huidige datumopmaak die wordt bepaald door SET DATE, DBASEWIN.INI of de optie **Internationaal** in het Configuratiescherm van Windows (in die volgorde). Dat wil zeggen, de instellingen bij SET DATE hebben een hogere prioriteit dan die in DBASEWIN.INI, en de instellingen in DBASEWIN.INI hebben weer een hogere prioriteit dan de instellingen in het Configuratiescherm van Windows. Let er op dat <*Duitdr*> overeenkomt met de datumopmaak die wordt gebruikt op het moment dat het programma wordt uitgevoerd.

Als u een ongeldige datum doorgeeft aan YEAR(), wordt die eerst omgezet in een geldige datum en wordt vervolgens het uit vier cijfers bestaande jaartal als resultaat gegeven. Als u een lege of een andere dan een datumuitdrukking tussen accolades ({ }) doorgeeft aan YEAR(), wordt 0 als resultaat gegeven. Als u een andere dan een datumuitdrukking of een uitdrukking die niet tussen accolades staat, doorgeeft aan YEAR(), wordt een fout als resultaat gegeven.

Met STR(YEAR()) kunt u een index maken op een datumveld in combinatie met een tekenveld.

Voorbeeld

Zie CMONTH() voor een voorbeeld met YEAR().

Zie ook

DATE(), DAY(), DOW(), MONTH(), SET CENTURY, STR()

ZAP

Velden en records

Verwijdert alle records uit de huidige tabel.

Syntaxis

ZAP

Beschrijving

ZAP is de snelste manier om alle records uit een tabel te verwijderen. U kunt ook alle records uit een tabel verwijderen met DELETE ALL, gevolgd door PACK. De tabel moet exclusief zijn geopend voordat u ZAP kunt gebruiken.

Als SET SAFETY is ingeschakeld (ON) en u gebruikt ZAP, wordt een waarschuwing getoond. U krijgt dan de gelegenheid de bewerking te bevestigen voordat alle records worden verwijderd.

Voorbeeld

In het volgende voorbeeld wordt ZAP gebruikt om *permanent* alle records uit de huidige tabel te verwijderen:

```
USE Klanten
SET TALK OFF
SET SAFETY ON
COPY TO TIJD
USE TIJD EXCLUSIVE
zap_jn = "N"
ACCEPT "Wilt u alle records verwijderen? (J/N) " ;
  TO zap_jn
READ
IF UPPER(zap_jn) = "J"
   ZAP
? "Alle records zijn verwijderd uit " + ;
   ALIAS()
ENDIF
CLOSE ALL
SET TALK ON
```

Zie ook

DELETE, PACK, SET SAFETY

Systeemgeheugenvariabelen

Hoofdstuk

5

Systeemgeheugenvariabelen

_alignment — Afdrukken

Lijnt de uitvoer van de commando's ? en ?? links, rechts of gecentreerd uit binnen de marges die worden aangegeven door _lmargin en _rmargin, als _wrap waar is.

Syntaxis

_alignment = <*Tuitdr*>

<Tuitdr>

De tekenuitdrukking "LEFT", "CENTER" of "RIGHT". U kunt <*Tuitdr*> opgeven in elke combinatie van hoofdletters en kleine letters.

Standaardinstelling

De standaardinstelling voor _alignment is "LEFT".

Beschrijving

Met _alignment kunt u de uitvoer van de commando's ? en ?? links, rechts of gecentreerd uitlijnen binnen de marges die worden aangegeven door _lmargin en _rmargin. De instelling voor _alignment is alleen van kracht als _wrap waar (.T.) is.

Tekst in een veld kunt u uitlijnen met de opmaakopties "B," "I" en "J" van de PICTURE- of FUNCTION-clausule van een @...SAY-instructie. Zie Picture en Function in Hoofdstuk 8 voor meer informatie over opmaakopties.

Voorbeeld

In het volgende voorbeeld wordt _wrap ingesteld op waar en wordt een reeks afgedrukt met drie verschillende instellingen voor uitlijning: links, gecentreerd en rechts:

```
WrapOpsl=_wrap       && instelling van _wrap opslaan
_wrap=.t.            && moet .t. zijn
UitlOpsl=_alignment  && instelling voor uitlijning opslaan
_alignment="LEFT"
? "Hallo LINKS"
_alignment="RIGHT"
? "Hallo RECHTS"
_alignment="CENTER"
? "Hallo GECENTREERD"
_alignment=UitlOpsl  && instelling herstellen
_wrap=WrapOpsl       && instelling herstellen
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

?, ??, @...SAY...GET, _lmargin, _rmargin, _wrap, SET MARGIN

_app

Objecten

Bevat een objectverwijzing naar het applicatie-object. Dat is de actieve dBASE-sessie.

Syntaxis

_app.<*kenmerknaam*> = <*nieuwe waarde*>

<kenmerknaam>

Het kenmerk van het applicatie-object dat u wilt wijzigen.

<nieuwe waarde>

De waarde die u aan het kenmerk van het applicatie-object wilt geven.

Beschrijving

Met _app kunt u toegang krijgen tot het applicatie-object en het object besturen.

Het applicatie-object is de huidige dBASE-sessie. Dit object heeft vier kenmerken: DdeServiceName, ClassName, Insert en OnInitiate. Het applicatie-object bevat tevens een object met de naam FrameWin. Dat is het dBASE-applicatievenster.

DdeServiceName

Met het kenmerk DdeServiceName kunt u een client-applicatie een DDE-koppeling laten instellen met een bepaalde dBASE-sessie als meerdere dBASE-sessies actief zijn.

De waarde die u opgeeft voor DdeServiceName is de naam van een dBASE-applicatie-object. Elke actieve dBASE-sessie is een applicatie-object. Als u een dBASE-sessie start, start u eigenlijk een instantie van een dBASE-applicatie-object.

Als u bijvoorbeeld twee dBASE-sessies start, maakt u twee dBASE-applicatie-objecten. Door voor het kenmerk DdeServiceName van een van beide applicatie-objecten "NWSERVER" (of een andere unieke naam) op te geven, kunt u een externe toepassing zoals Quattro Pro onderscheid laten maken tussen de twee sessies. U kunt de systeemgeheugenvariabele _app gebruiken om naar het kenmerk DdeServiceName van een van beide sessies te verwijzen:

```
_app.DdeServiceName = "NWSERVER"
```

Quattro Pro zou als volgt met {INITIATE} een DDE-koppeling tot stand kunnen brengen met de sessie NWSERVER:

```
{INITIATE  "NWSERVER", "MijnTopic", CHANNEL}
```

Een DDE-koppeling met de anders sessie (waarvan het kenmerk DdeServiceName nog steeds de standaardnaam "DBASEWIN" bevat) zou als volgt kunnen worden ingesteld:

```
{INITIATE  "DBASEWIN", "MijnTopic", CHANNEL}
```

ClassName

ClassName geeft de klasse van het applicatie-object aan (APPLICATION). Dit kenmerk is alleen leesbaar.

Insert

Bepaalt of *Insert* is in- of uitgeschakeld. Als u Insert instelt op waar (.T.), is *Insert* ingeschakeld.

OnInitiate

Voert een initialisatieroutine uit. Deze routine maakt DDETopic-objecten, die DDE-server-acties behandelen. OnInitiate voert de subroutine uit als een client-applicatie om een DDE-koppeling met een dBASE-sessie verzoekt. Zie CLASS DDETopic en Hoofdstuk 26 in *Programmeren* voor meer informatie.

Het FrameWin-object

Het dBASE-applicatievenster. Dit object bevat vijf kenmerken: Visible, ClassName, hWnd, Text en WindowState.

Visible

Bepaalt of het dBASE-applicatievenster zichtbaar of verborgen is. Stel Visible in op onwaar (.F.) als u niet-MDI-formulieren wilt weergeven zonder de dBASE-omgeving te tonen.

ClassName
ClassName geeft de klasse van het applicatie-object aan (FRAMEWINDOW).

hWnd
Resulteert in de object-handle van het FrameWindow-object van dBASE. Externe functies die zijn geschreven in andere talen als C, Pascal of ASM, gebruiken deze handle om het object aan te duiden. Deze externe functies worden meestal opgeslagen in DLL-bestanden (DLL is een afkorting van Dynamic Link Library). U kunt de object-handle van het FrameWindow-object bijvoorbeeld als parameter doorgeven aan een externe functie zodat de functie het venster kan openen of sluiten.

hWnd is uitsluitend leesbaar.

Zie EXTERN en Hoofdstuk 25 in *Programmeren* voor meer informatie over DLL-bestanden en externe functies.

Text
Geeft een tekenreeks aan die in de titelbalk van het dBASE-applicatievenster verschijnt. Zie de beschrijving van het kenmerk Text voor meer informatie.

WindowState
Bepaalt of het dBASE-applicatievenster wordt weergegeven als maximumvenster, als pictogram of in het oorspronkelijke formaat. Zie WindowState voor meer informatie.

Voorbeeld
De volgende instructie kan aan het begin van een programma worden gebruikt om op de titelbalk van het dBASE-applicatievenster een bedrijfsnaam weer te geven:

```
_app.FrameWin.Text = "De Bunniksche Bank"
```

Zo verwijdert u de dBASE-omgeving van het scherm:

```
_app.FrameWin.Visible = .F.
```

Zo schakelt u Insert uit in uw applicatie:

```
_app.Insert = .F.
```

Het kenmerk ClassName is uitsluitend leesbaar. U kunt het dus alleen gebruiken in instructies als deze:

```
IF _app.ClassName = "APPLICATION"      && Resulteert in .T.
  * bewerking uitvoeren
ENDIF
  * of
? _app.ClassName  && Resulteert in tekenreeks "APPLICATION"
  * of
gVar = _app.ClassName         && Initialiseert variabele
```

Zie ook
CLASS DDETopic, EXTERN, LOAD DLL, RESOURCE( )

_box — Afdrukken

Bepaalt of bij de uitvoer van het commando ? dBASE IV-kaders worden weergegeven die zijn gedefinieerd met DEFINE BOX.

Syntaxis

_box = <*Luitdr*>

<Luitdr>
De logische uitdrukking .T. of .F.

Standaardinstelling

De standaardinstelling voor _box is .F.

Beschrijving

Met _box bepaalt u of kaders moeten worden getoond die zijn gedefinieerd met het dBASE IV-commando DEFINE BOX. Als u _box instelt op waar (.T.), worden kaders getoond op de achtergrond van de doorlopende uitvoer (streaming output) van het commando ?.

Met _box kunt u exact bepalen wanneer het kader moet worden weergegeven of afgedrukt. U kunt het afdrukken van een kader bijvoorbeeld onderbreken door _box in te stellen op onwaar (.F.) voordat het gehele kader is afgedrukt. Als u later _box weer instelt op .T., wordt de rest van het kader afgedrukt.

Voorbeeld

_box wordt alleen gebruikt in combinatie met DEFINE BOX. Zie het voorbeeld met DEFINE BOX.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

ACTIVATE WINDOW, DEFINE BOX

_curobj — Formulieren

Geeft de positie in de tabvolgorde aan van het huidige geselecteerde object.

Syntaxis

_curobj = <*Nuitdr*>

Beschrijving

_curobj bevat een getal dat de positie van het geselecteerde object aangeeft.

Een ander formulierkenmerk, ActiveControl, geeft de naam als resultaat van het object dat de focus heeft.

Voorbeeld

In het volgende voorbeeld wordt een formulier gedefinieerd met vier objecten. Het kenmerk OnGotFocus wordt gebruikt om in het resultatenpaneel van het commandovenster het huidige objectnummer te tonen als een object de focus krijgt:

```
USE Contact.DBF
CLEAR
LOCAL f
f=NEW ENTRY()
f.OPEN()
CLASS Entry OF FORM
  this.Top=2
  this.Left=2
  this.Width=36
  this.Height=8
  this.Text= "Contact.DBF"
  this.StatusMessage="_curobj in commandovenster"
  DEFINE ENTRYFIELD CompCode OF THIS AT 2,2;
    PROPERTY Width 7, Height 1.5,;
      DataLink "COMPCODE", OnGotFocus {;?_curobj}
  DEFINE ENTRYFIELD Contact OF THIS AT 2,12;
    PROPERTY Width 22, Height 1.5,;
    DataLink "CONTACT", OnGotFocus {;?_curobj}
  DEFINE PUSHBUTTON Vorige OF THIS AT 5,9;
    PROPERTY Text "<<", Height 2,;
      OnClick {;SKIP-1},;
      OnGotFocus {;?_curobj}
  DEFINE PUSHBUTTON Volgende OF THIS AT 5,19;
    PROPERTY TEXT ">>", Height 2,;
      OnClick {;SKIP},;
      OnGotFocus {;?_curobj}
ENDCLASS
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

ActiveControl

_dbwinhome

Bevat het pad naar de hoofddirectory van dBASE.

Syntaxis

? _dbwinhome

Beschrijving

Met _dbwinhome kunt u de hoofddirectory van dBASE aangeven. Dat is de directory waar de dBASE-bestanden zijn geïnstalleerd. Het installatieprogramma installeert de dBASE-bestanden (standaard) in de directory \DBASEWIN en in onder meer de subdirectory's \BIN en \VOORBD van \DBASEWIN. _dbwinhome bevat de naam van de dBASE-hoofddirectory (in dit geval DBASEWIN.) _dbwinhome resulteert in het volledige pad, ongeacht de instelling voor SET FULLPATH.

Met HOME( ) kunt u bepalen in welke directory de huidige actieve versie van DBASEWIN.EXE zich bevindt.

Voorbeeld

_DBWINHOME verandert alleen als dBASE opnieuw wordt geïnstalleerd in een andere directory:

```
? _dBWINHOME && zoals D:\DBASEWIN
* dBASE is geinstalleerd in de
* subdirectory DBASEWIN op station D:
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

CD, HOME( ), MKDIR, SET DIRECTORY, SET FULLPATH, SET PATH

_indent

Geeft aan met welk aantal kolommen de eerste regel van een alinea in de uitvoer van het commando ? inspringt (als _wrap waar is).

Syntaxis

_indent = <*Nuitdr*>

<Nuitdr>

Het kolomnummer ten opzichte van de linkermarge waar de eerste regel van een alinea begint. U kunt voor <*Nuitdr*> een niet-geheel getal opgeven om de uitvoer nauwkeurig te plaatsen bij een proportioneel lettertype.

Standaardinstelling

De standaardinstelling voor _indent is 0.

Beschrijving

Met _indent kunt u aangeven in welke kolom ten opzichte van de linkermarge de eerste regel van een nieuwe alinea begint. (De linkermarge stelt u in met _lmargin.) De instelling voor _indent is alleen van kracht als _wrap waar (.T.) is.

Als u uitvoer van het commando ? naar de printer stuurt, wordt elk teken geplaatst aan de hand van het *coördinatenvlak*, een tweedimensionaal raster. Het coördinatenvlak bestaat uit tekencellen waarvan de breedte wordt bepaald door de waarde van _ppitch. Zie de beschrijving van _ppitch voor een tabel met de mogelijke waarden van _ppitch. De hoogte van elke tekencel wordt bepaald door de grootte van het font. Zie Hoofdstuk 16 in *Programmeren* voor meer informatie over het coördinatenvlak.

Als u de waarde van _indent wijzigt, wordt bij de berekening van de celbreedte binnen het coördinatenvlak de huidige waarde van _ppitch gebruikt, ongeacht of u afdrukt met een proportioneel of niet-proportioneel font. Als u ? zonder de optie STYLE gebruikt en alleen gehele getallen opgeeft voor de coördinaten, wordt een niet-proportioneel font gebruikt, en ziet de uitvoer er net zo uit als in dBASE IV.

Als u de eerste regel van een alinea wilt laten inspringen, moet u een waarde groter dan 0 opgeven. Als u de eerste regel van een alinea bijvoorbeeld vijf kolommen rechts van de linkermarge wilt laten beginnen, moet u voor _indent 5 opgeven. Een hangende alinea kunt u maken door een negatieve waarde op te geven. Als u de eerste regel van een alinea bijvoorbeeld vijf kolommen links van de linkermarge wilt laten beginnen, moet u voor _indent -5 opgeven. De standaardwaarde 0 zorgt dat alle regels in een alinea links worden uitgelijnd tegen de linkermarge. De som van _lmargin en _indent moet groter zijn dan 0 en kleiner dan _rmargin.

Voorbeeld

In het volgende voorbeeld worden regelovergangen ingesteld (_wrap is waar) en wordt de eerste regel van tekst met regelovergangen ingesprongen:

```
_indent=3              && inspringen instellen
WrapOpsl=_wrap         && instelling voor _wrap opslaan
_wrap=.t.              && verplicht voor uitlijning
LMargeOpsl=_lmargin    && instelling voor linkermarge opslaan
_lmargin=5
RMargeOpsl=_rmargin   && instelling voor rechtermarge opslaan
_rmargin=20
? "Amsterdam, Rotterdam en Den Haag "+;
  "zijn grote steden."
* Tekst wordt nu met regelovergangen
```

```
* weergegeven tussen de kolommen 5 en 20
*
*       Amsterdam,
*     Rotterdam en
*     Den Haag zijn
*     grote steden.
*
_rmargin=RMargeOpsl   && oorspronkelijke
_lmargin=LMargeOpsl   &&   instellingen
_wrap=WrapOpsl        &&     herstellen
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

?, ??, _alignment, _lmargin, _ploffset, _rmargin, _wrap, SET MARGIN

_lmargin

Afdrukken

Definieert de linkermarge voor de uitvoer van de commando's ? en ?? als _wrap waar (.T.) is.

Syntaxis

_lmargin = <*Nuitdr*>

<Nuitdr>

Het kolomnummer van de linkermarge. Het geldige bereik loopt van 0 tot en met 254. U kunt voor <*Nuitdr*> een niet-geheel getal opgeven om bij een proportioneel font de uitvoer nauwkeurig te plaatsen.

Standaardinstelling

De standaardinstelling voor _lmargin is 0.

Beschrijving

Met _lmargin stelt u de linkermarge in voor de uitvoer van de commando's ? en ??. Als u uitvoer naar de printer stuurt, bepaalt _lmargin de linkermarge ten opzichte van _ploffset (de linkerpagina-offset). Als _ploffset bijvoorbeeld 10 is en _lmargin is 5, wordt afgedrukt vanaf de vijftiende kolom. De instelling voor _lmargin is alleen van kracht als _wrap waar (.T.) is.

Als u uitvoer naar de printer stuurt, wordt elk teken geplaatst aan de hand van het *coördinatenvlak*, een tweedimensionaal raster. Het coördinatenvlak bestaat uit tekencellen waarvan de breedte wordt bepaald door de waarde van _ppitch. Zie de

beschrijving van _ppitch voor een tabel met de mogelijke waarden van _ppitch. De hoogte van elke tekencel wordt bepaald door de grootte van het font. Zie Hoofdstuk 16 in *Programmeren* voor meer informatie over het coördinatenvlak.

Als u de waarde van _indent wijzigt, wordt bij de berekening van de celbreedte binnen het coördinatenvlak rekening gehouden met de huidige waarde van _ppitch, ongeacht of u afdrukt met een proportioneel of niet-proportioneel font. Als u ? zonder de optie STYLE gebruikt en alleen gehele getallen opgeeft voor de coördinaten, wordt een niet-proportioneel font gebruikt, en ziet de uitvoer er net zo uit als in dBASE IV.

Als u _indent gebruikt om de eerste regel van elke alinea te laten inspringen, moet de som van de waarde voor _lmargin en _indent kleiner zijn dan de waarde voor _rmargin.

Voorbeeld

In het volgende voorbeeld wordt _lmargin gebruikt. Eerst worden regelovergangen ingesteld en vervolgens wordt de linkermarge gewijzigd:

```
_wrap=.t.              && verplicht voor _lmargin
_lmargin=0
? "01234567890"
_lmargin=5
? "Marge gewijzigd"
* Uitvoer:
* 01234567890
*      Marge gewijzigd
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

?, ??, _alignment, _indent, _ploffset, _rmargin, _wrap, SET MARGIN, Style

_padvance

Afdrukken

Bepaalt of de printer na een afdruktaak het papier doorvoert met een paginadoorvoer of met regeldoorvoertekens.

Syntaxis

_padvance = <*Tuitdr*>

<Tuitdr>

De tekenuitdrukking "FORMFEED" of "LINEFEEDS".

Standaardinstelling

De standaardinstelling voor _padvance is "FORMFEED".

Beschrijving

Met _padvance kunt u instellen of het printerpapier naar het begin van de volgende pagina moet worden doorgevoerd met een paginadoorvoerteken of met regeldoorvoertekens. Als u de standaardinstelling "FORMFEED" handhaaft, wordt het papier doorgevoerd aan de hand van de standaardinstelling voor de paginalengte van de printer.

Tractor-feedprinters (zoals de meeste matrixprinters) werken meestal met regeldoorvoertekens ("LINEFEEDS"), terwijl laserprinters en dergelijke gewoonlijk een paginadoorvoerteken ("FORMFEED") gebruiken.

Opmerking Als u CHR(12) naar de printer stuurt, heeft dat altijd een paginadoorvoer tot gevolg, ongeacht de instelling voor _padvance.

Gebruik de instelling "LINEFEEDS" als u de paginalengte wilt wijzigen of als u een ander aantal regels wilt afdrukken dan het standaardaantal zonder de instelling van de printer te wijzigen. Als u bijvoorbeeld korte pagina's afdrukt, zoals verkoopbonnen van 20 regels, kunt u voor _plength de lengte van de uitvoer (20 in dit voorbeeld) en voor _padvance "LINEFEEDS" opgeven.

Het aantal regeldoorvoertekens dat nodig is om naar het begin van de volgende pagina te gaan, is afhankelijk van het feit of u een paginadoorvoer geeft bij doorlopende (streaming) of niet-doorlopende (non-streaming) uitvoer. Zie Hoofdstuk 24 in *Programmeren* voor meer informatie over doorlopende en niet-doorlopende uitvoer.

Bij doorlopende uitvoer vindt een paginadoorvoer plaats als u de volgende commando's gebruikt:

- EJECT PAGE zonder een behandelingsroutine voor ON PAGE
- EJECT PAGE met een behandelingsroutine voor ON PAGE als de huidige regelpositie zich bevindt voorbij de regel die is opgegeven bij ON PAGE
- PRINTJOB of ENDPRINTJOB terwijl _peject een paginadoorvoer tot gevolg heeft

In deze gevallen wordt het aantal regeldoorvoertekens dat aan de printer moet worden doorgegeven, berekend met de formule `_plength - _plineno`.

Bij niet-doorlopende uitvoer vindt een paginadoorvoer plaats als u de volgende commando's gebruikt:

- EJECT
- SET DEVICE TO PRINTER met een gedwongen paginadoorvoer met het commando @

In deze gevallen wordt het aantal regeldoorvoertekens dat aan de printer moet worden doorgegeven, berekend met de formule `_plength - MOD(PROW( ), _plength)`.

Voorbeeld

Er zijn twee instellingen voor _padvance:

```
_padvance="FORMFEED"
_padvance="LINEFEEDS"
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

_peject, _plength, EJECT, EJECT PAGE, PRINTJOB...ENDPRINTJOB, ON PAGE, SET DEVICE

_pageno

Afdrukken

Stelt het huidige paginanummer in.

Syntaxis

_pageno = <*Nuitdr*>

<Nuitdr>

Een geheel getal van 1 tot en met 32.767.

Beschrijving

Met _pageno kunt u pagina's nummeren in *doorlopende uitvoer* (streaming output) van commando's als ?, ?? en LIST. Zie Hoofdstuk 24 in *Programmeren* voor meer informatie over doorlopende en niet-doorlopende uitvoer.

Met _pageno kunt u het huidige paginanummer bepalen of een ander paginanummer instellen. U kunt het gebruiken om paginanummers af te drukken in een rapport. Als u documenten samenvoegt, kunt u met _pageno een nieuw paginanummer instellen voor de eerste pagina van het tweede document.

Het einde van een pagina wordt bereikt als de waarde van _plineno (het aantal regels) groter wordt dan de waarde van _plength (de huidige paginalengte in regels). Tekens wanneer in doorlopende uitvoer het einde van een pagina wordt bereikt, wordt de waarde _pageno automatisch verhoogd.

Voorbeeld

In het volgende voorbeeld worden 100 regels afgedrukt en wordt op regel 1 van elke pagina een koptekst afgedrukt:

```
SET TALK OFF
SET PRINTER ON
_pageno=1
FOR i=1 TO 100
  IF _plineno=1  && Bij eerste regel op pagina
    ? "Begin van pagina ",_pageno
  ENDIF
  ? "Regel ",i
NEXT i
SET PRINTER TO
```

```
CLOSE PRINTER
SET TALK ON
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

?, ??, _pbpage, _pepage, _plength, _plineno, LIST, ON PAGE

_pbpage — Afdrukken

Geeft het paginanummer aan van de eerste pagina die u wilt afdrukken.

Syntaxis

_pbpage = *<Nuitdr>*

<Nuitdr>

Het nummer van de pagina die als eerste moet worden afgedrukt. Het geldige bereik loopt van 1 tot en met 32.767. Geef een positief geheel getal op voor *<Nuitdr>*.

Standaardinstelling

De standaardinstelling voor _pbpage is 1.

Beschrijving

Met _pbpage kunt u beginnen met afdrukken op een bepaalde pagina. Pagina's met paginanummers die kleiner zijn dan _pbpage, worden niet afgedrukt. Met _pepage kunt u opgeven tot welke pagina moet worden afgedrukt.

Als u voor _pbpage een hogere waarde opgeeft dan voor _pepage, wordt een fout als resultaat gegeven.

Voorbeeld

In dit voorbeeld wordt _pbpage gebruikt om een pagina uit een rapport weg te laten. Er worden 100 regels naar de printer gestuurd en op elke regel wordt het pagina- en regelnummer afgedrukt. Het nummer van de eerste pagina is 2, dus pagina 1 wordt niet afgedrukt:

```
_pageno=1
_pbpage=2       && beginnen met pagina 2
SET PRINTER ON
PRINTJOB
FOR i=1 TO 100
  ?? "Pagina",_pageno," Regel",_plineno
  ?                   && gedwongen regeldoorvoer
```

```
NEXT i
ENDPRINTJOB
SET PRINTER OFF
CLOSE PRINTER          && beginnen met afdrukken
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

_pageno, _pepage, PRINTJOB...ENDPRINTJOB

_pcolno

Afdrukken

Geeft de huidige kolom aan voor doorlopende uitvoer (streaming output) of stelt die in.

Syntaxis

_pcolno = <*Nuitdr*>

<*Nuitdr*>

Het kolomnummer waar met afdrukken moet worden begonnen. Het geldige bereik loopt van 0 tot en met 255. U mag voor <*Nuitdr*> een niet-geheel getal opgeven om de uitvoer bij een proportioneel lettertype nauwkeurig te plaatsen.

Standaardinstelling

De standaardinstelling voor _pcolno is 0.

Beschrijving

Met _pcolno kunt u de kolompositie van doorlopende uitvoer (streaming output) van commando's als ?, ?? en LIST instellen. Zie Hoofdstuk 24 in *Programmeren* voor meer informatie over doorlopende uitvoer.

Als u uitvoer naar de printer stuurt, wordt elk teken geplaatst aan de hand van het *coördinatenvlak*, een tweedimensionaal raster. Het coördinatenvlak bestaat uit tekencellen waarvan de breedte wordt bepaald door de waarde van _ppitch. Zie de beschrijving van _ppitch voor een tabel met de mogelijke waarden van _ppitch. De hoogte van elke tekencel wordt bepaald door de grootte van het font. Zie Hoofdstuk 16 in *Programmeren* voor meer informatie over het coördinatenvlak.

Als u de waarde van _pcolno wijzigt, wordt bij de berekening van de celbreedte binnen het coördinatenvlak rekening gehouden met de huidige waarde van _ppitch, ongeacht of u afdrukt met een proportioneel of niet-proportioneel font. Als u ? zonder de optie STYLE gebruikt en alleen gehele getallen opgeeft voor de coördinaten, wordt een niet-proportioneel font gebruikt, en ziet de uitvoer er net zo uit als in dBASE IV.

De functie PCOL( ) geeft ook de huidige kolompositie van de printer als resultaat, maar als SET PRINTER is uitgeschakeld (OFF), verandert de waarde die PCOL( ) als resultaat geeft niet. _pcolno geeft daarentegen altijd de huidige kolompositie in doorlopende uitvoer als resultaat, ongeacht de instelling voor SET PRINTER.

Voorbeeld

In het volgende voorbeeld worden de cijfers 1 tot en met 5 weergegeven. Met _pcolno wordt elk cijfer zo geplaatst dat het op de eigen positie begint:

```
SET TALK OFF
FOR i=1 to 5
  _pcolno=i      && kolompositie instellen
  reeks=ltrim(str(i))
  * i omzetten in een enkel teken
  ?? reeks
  ?
NEXT I
SET TALK ON
* Uitvoer:
* 1
*  2
*   3
*    4
*     5
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

? , _plineno, _rmargin, PCOL( ), SET DEVICE, SET PRINTER

_pcopies — Afdrukken

Geeft het aantal exemplaren aan dat moet worden afgedrukt met PRINTJOB.

Syntaxis

_pcopies = <*Nuitdr*>

<*Nuitdr*>

Het aantal af te drukken exemplaren. Het geldige bereik loopt van 1 tot en met 32.767. Geef een positief geheel getal op voor <*Nuitdr*>.

Standaardinstelling

De standaardinstelling voor _pcopies is 1.

Beschrijving

Met _pcopies kunt u opgeven hoeveel exemplaren van een bepaalde afdruktaak moeten worden afgedrukt. U kunt een waarde toekennen aan _pcopies in het commandovenster of in een programma. De waarde van _pcopies heeft alleen effect als u een afdruktaak naar de printer stuurt met het commando PRINTJOB. In een programma moet u eerst een waarde toekennen aan _pcopies en dan het commando PRINTJOB opgeven.

Voorbeeld

In dit voorbeeld wordt _pcopies gebruikt om drie exemplaren af te drukken van een afdruktaak:

```
_pcopies=3          && Drie exemplaren
SET PRINTER ON
PRINTJOB
? "Een heel klein rapportje"
ENDPRINTJOB
SET PRINTER OFF
CLOSE PRINTER       && beginnen met afdrukken
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

PRINTJOB...ENDPRINTJOB

_pdriver

Afdrukken

Geeft het actieve stuurprogramma voor de printer aan of activeert een nieuw stuurprogramma.

Syntaxis

_pdriver = <*Tuitdr*>

<*Tuitdr*>

De naam van het nieuwe stuurprogramma voor de printer.

Standaardinstelling

De standaardinstelling voor _pdriver is het stuurprogramma voor de printer dat u hebt opgegeven in het Configuratiescherm van Windows. Als u in het Configuratiescherm geen stuurprogramma voor de printer hebt opgegeven, bevat _pdriver een lege tekenreeks ("").

Beschrijving

Met _pdriver kunt u het huidige stuurprogramma voor de printer bepalen of een nieuw, geïnstalleerd stuurprogramma activeren. (In het Configuratiescherm van Windows kunt u een nieuw stuurprogramma voor de printer installeren.)

De waarde van _pdriver bestaat uit twee elementen (gescheiden door een komma): de naam van het bestand van het stuurprogramma en de naam waaronder de printer wordt vermeld in WIN.INI. Als voor de huidige printeraansturing geen printernaam is opgegeven, bevat _pdriver alleen de bestandsnaam van de printeraansturing. Als bijvoorbeeld een stuurprogramma is ingesteld voor de Laserjet IIISi PostScript printer, kan _pdriver "pscript,HP LaserJet IIISi PostScript" bevatten. U kunt dit stuurprogramma activeren met het commando _pdriver = "pscript,HP LaserJet IIISi PostScript".

Vanuit dBASE kunt u gemakkelijker een stuurprogramma activeren met CHOOSEPRINTER( ). In Windows kunt u een stuurprogramma activeren met de optie **Printers** in het Configuratiescherm van Windows. Met CHOOSEPRINTER( ) opent u het dialoogvenster **Printerinstellingen**. In dat dialoogvenster kunt u tevens opties instellen voor papierformaat, bron en afdrukstand (liggend of staand). In het dialoogvenster **Printers** in het Configuratiescherm van Windows kunt u **Instellen** kiezen om deze opties in te stellen.

Voorbeeld

In het volgende voorbeeld wordt _pdriver gebruikt om het huidige stuurprogramma voor de printer te bepalen:

```
? _pdriver
* In het geval van een Epson FX 80 is het resultaat:
*       EPSON9,Epson FX-80
* In het geval van een HP LaserJet met PostScript
* is het resultaat:
*       pscript,HP LaserJet IIISi PostScript
```

Zo stelt u met _pdriver het stuurprogramma voor de printer in:

```
_pdriver="pscript"
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. Alleen Windows-namen voor printerstuurprogramma's worden ondersteund; dBASE IV-namen voor printerstuurprogramma's worden niet ondersteund.

Zie ook

_pform, _ppitch, _pquality, CHOOSEPRINTER( ), SET PRINTER

_peject — Afdrukken

Bepaalt of voor en na een afdruktaak een pagina wordt doorgevoerd.

Syntaxis

_peject= <*Tuitdr*>

<Tuitdr>

De tekenuitdrukking "before", "after", "both" of "none".

Standaardinstelling

De standaardinstelling voor _peject is "before". De printer voert een paginadoorvoer uit voordat een afdruktaak wordt begonnen.

Beschrijving

Met _peject kunt u opgeven of en wanneer de printer een paginadoorvoer moet uitvoeren. Als u een nieuwe waarde wilt toekennen aan _peject, moet u dat doen voordat u het commando PRINTJOB opgeeft.

De volgende tabel geeft een overzicht van de betekenis van de opties voor _peject.

<Tuitdr>	Gevolg
"before"	Paginadoorvoer voordat de eerste pagina wordt afgedrukt
"after"	Paginadoorvoer nadat de laatste pagina is afgedrukt
"both"	Paginadoorvoer voor en na de afdruktaak
"none"	Geen paginadoorvoer voor of na de afdruktaak

Opmerking De systeemgeheugenvariabele _peject heeft een andere werking dan het commando EJECT, waarmee u de printer opdracht geeft naar het begin van de volgende pagina te gaan.

Voorbeeld

In het volgende voorbeeld worden de vier mogelijkheden voor _peject gedemonstreerd. De laatste instelling is actief als het commando PRINTJOB wordt gegeven:

```
_peject="before"
_peject="after"
_peject="both"
_peject="none"
* _peject moet zijn ingesteld voordat PRINTJOB wordt gegeven
PRINTJOB
 ? "Hallo allemaal"
ENDPRINTJOB
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

_padvance, EJECT, EJECT PAGE, PRINTJOB...ENDPRINTJOB

_pepage

Afdrukken

Bepaalt het nummer van de laatste pagina voor een afdruktaak.

Syntaxis

_pepage = <*Nuitdr*>

<Nuitdr>

Het nummer van de laatste pagina die u wilt afdrukken. Het geldige bereik loopt van 1 tot en met 32.767. U moet een positief geheel getal opgeven voor <*Nuitdr*>.

Standaardinstelling

De standaardinstelling voor _pepage is 32,767.

Beschrijving

Gebruik _pepage om met het afdrukken van een afdruktaak te stoppen bij een bepaalde pagina. Pagina's met een paginanummer dat groter is dan _pepage, worden niet afgedrukt. Met _pbpage kunt u opgeven welke pagina als eerste moet worden afgedrukt.

Als u voor _pepage een kleinere waarde opgeeft dan voor _pbpage, wordt een fout als resultaat gegeven.

Voorbeeld

In het volgende voorbeeld worden twee pagina's afgedrukt. Het programma drukt 500 regels af en op elke regel wordt het paginanummer en het regelnummer afgedrukt. Pagina 2 wordt als laatste pagina ingesteld, dus alleen de pagina's 1 en 2 worden afgedrukt:

```
_pageno=1
_pbpage=1       && standaardinstelling voor pbpage
_pepage=2       && pagina 2 is laatste pagina
SET PRINTER ON
PRINTJOB
FOR i=1 TO 500
  ?? "Pagina",_pageno," Regel",_plineno
  ?                     && gedwongen regeldoorvoer
NEXT i
ENDPRINTJOB
SET PRINTER OFF
CLOSE PRINTER           && beginnen met afdrukken
```

In dit geval geeft het commando ? een regeldoorvoer op voordat consequent wordt verwerkt. Het juiste regelnummer wordt opgehaald met ??

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

_pageno, _pbpage, PRINTJOB...ENDPRINTJOB

_pform

Afdrukken

Geeft het huidige afdrukformulier aan of activeert een ander afdrukformulier.

Syntaxis

_pform = <*bestandsnaam*>

<*bestandsnaam*>

De naam van het afdrukformulier (.PRF-bestand).

Standaardinstelling

De standaardinstelling voor _pform is een lege tekenreeks ("").

Beschrijving

Met _pform kunt u de naam van het huidige afdrukformulier bepalen of een nieuw afdrukformulier instellen. Een afdrukformulier (.PRF-bestand) is een binair dBASE-bestand dat instellingen voor een afdruktaak bevat. Het afdrukformulier bevat de volgende systeemgeheugenvariabelen:

Variabele	Betekenis
_padvance	Bepaalt of de printer naar een nieuwe pagina gaat met een paginadoorvoer of met regeldoorvoertekens.
_pageno	Het huidige paginanummer.
_pbpage	Het paginanummer waar met afdrukken moet worden begonnen.
_pcopies	Het aantal exemplaren dat moet worden afgedrukt.
_pdriver	Activeert een opgegeven stuurprogramma voor de printer. (Als het afdrukformulier afkomstig is van dBASE IV, wordt deze waarde genegeerd.)
_peject	Bestuurt opdrachten voor paginadoorvoer voor en na een afdruktaak.
_pepage	Het paginanummer van de laatste pagina in een afdruktaak.
_plength	Het aantal regels per pagina voor doorlopende uitvoer.
_ploffset	De breedte van de linkerkantlijn op een afgedrukte pagina.
_ppitch	Het aantal tekens per inch dat wordt afgedrukt.
_pquality	Geeft aan of de printer in kwaliteit- of kladmodus afdrukt.
_pspacing	De regelafstand voor doorlopende uitvoer.
_pwait	Bepaalt of de printer na elke pagina pauzeert.

Als u een afdrukformulier opgeeft door de naam van het formulier toe te kennen aan _pform, worden de waarden in het bestand toegekend aan de betreffende variabelen. Elke volgende afdruktaak wordt afgedrukt aan de hand van deze instellingen.

Voorbeeld

In het volgende voorbeeld worden twee rapporten gebruikt, Rapport1 en Rapport2. Het afdrukformulier van Rapport1, Rapport1.PRF, wordt gebruikt om Rapport2 af te drukken:

```
_pform= "Rapport1"
REPORT FORM Rapport2
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

PRINTJOB...ENDPRINTJOB

_plength — Afdrukken

Geeft het aantal regels per pagina in doorlopende uitvoer (streaming output) aan.

Syntaxis

_plength = <*Nuitdr*>

<Nuitdr>

Het aantal regels per pagina. Het geldige bereik loopt van 1 tot en met 32.767. U kunt voor <*Nuitdr*> een niet-geheel getal opgeven om de uitvoer bij een proportioneel lettertype nauwkeurig te plaatsen.

Standaardinstelling

De standaardpaginalengte wordt bepaald door het standaardpaginaformaat van de huidige printeraansturing en de afdrukstand (staand of liggend).

Beschrijving

Met _plength kunt u een andere paginalengte opgeven dan die is ingesteld in de huidige printeraansturing. Als u bijvoorbeeld korte pagina's afdrukt, zoals verkoopbonnen van 20 regels, kunt u voor _plength de lengte van de uitvoer (20 in dit voorbeeld) opgeven en voor _padvance "LINEFEEDS" instellen om naar het begin van de volgende pagina te gaan.

Vanuit dBASE kunt u een ander stuurprogramma activeren met CHOOSEPRINTER( ). In Windows kunt u een stuurprogramma activeren met de optie **Printers** in het

Configuratiescherm van Windows. Met CHOOSEPRINTER( ) opent u het dialoogvenster **Printerinstellingen**. In dat dialoogvenster kunt u tevens opties instellen voor papierformaat, bron en afdrukstand (liggend of staand). In het dialoogvenster **Printers** in Configuratiescherm van Windows kunt u **Instellen** kiezen om deze opties in te stellen.

Als u een andere printeraansturing instelt of een andere afdrukstand opgeeft, wordt de waarde van _plength automatisch aangepast. In dBASE kunt u een andere printeraansturing instellen met CHOOSEPRINTER( ) of door een waarde toe te kennen aan _pdriver. In Windows kunt u een stuurprogramma activeren met de optie **Printers** in het Configuratiescherm van Windows (de gewijzigde instelling wordt echter pas van kracht als u dBASE afsluit en een nieuwe dBASE-sessie start). U kunt op deze manieren ook de afdrukstand wijzigen, maar in dBASE kunt u dat ook doen door de waarde van _porientation te veranderen.

Voorbeeld

In het volgende voorbeeld wordt een paginalengte ingesteld van 10 regels en worden 25 regels afgedrukt. Op elke regel wordt een regelnummer en een teller (van 1 tot en met 25) afgedrukt. Boven aan elke pagina wordt een koptekst van 3 regels afgedrukt:

```
_plength=10
_pageno=1
_plineno=0
SET PRINTER ON
FOR i=1 TO 25
  IF _plineno=0  && Bij eerste regel op elke pagina
    ?
    ? "Begin van pagina ",_pageno
    ?
  ENDIF
  ? "Regel",_plineno,"i=",i
NEXT i
SET PRINTER OFF
CLOSE PRINTER
* De eerste twee pagina's zien er zo uit:
*
* Begin van pagina          1
*
* Regel         3.00 i=          1
* Regel         4.00 i=          2
* Regel         5.00 i=          3
* Regel         6.00 i=          4
* Regel         7.00 i=          5
* Regel         8.00 i=          6
* Regel         9.00 i=          7
*
* Begin van pagina          2
*
* Regel         3.00 i=          8
* Regel         4.00 i=          9
* Regel         5.00 i=         10
* Regel         6.00 i=         11
* Regel         7.00 i=         12
```

```
* Regel          8.00 i=          13
* Regel          9.00 i=          14
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. De standaardpaginalengte in dBASE IV is 66 regels.

Zie ook

_padvance, _pdriver, _porientation, CHOOSEPRINTER( ), EJECT, EJECT PAGE

_plineno — Afdrukken

Geeft het huidige regelnummer aan in doorlopende uitvoer (streaming output) of wijzigt het.

Syntaxis

_plineno = <*Nuitdr*>

<Nuitdr>

Het regelnummer waar met afdrukken moet worden begonnen. Het geldige bereik loopt van 0 tot _plength - 1. U kunt voor <*Nuitdr*> een niet-geheel getal opgeven om de uitvoer bij een proportioneel lettertype nauwkeurig te plaatsen.

Standaardinstelling

De standaardinstelling voor _plineno is 0.

Beschrijving

Met _plineno kunt u afgedrukte doorlopende uitvoer (streaming output) van commando's als ?, ?? en LIST plaatsen. Zie Hoofdstuk 24 in *Programmeren* voor meer informatie over doorlopende uitvoer.

De functie PROW( ) geeft ook de huidige positie van de afdrukkop als resultaat, maar als SET PRINTER is uitgeschakeld (OFF), blijft de waarde die PROW( ) als resultaat geeft, onveranderd. De waarde van _plineno komt daarentegen altijd overeen met de huidige positie in doorlopende uitvoer, ongeacht de instelling voor SET PRINTER.

Voorbeeld

In het volgende voorbeeld worden 32 regels afgedrukt op vier pagina's. De paginalengte is 10 regels per pagina en op elke regel worden het paginanummer en het regelnummer afgedrukt:

```
_pbpage=1       && standaardinstelling voor _pbpage
_pepage=32767   && standaardinstelling voor _pepage
```

```
_plength=10     && paginalengte is 10 regels
_pageno=1
SET PRINTER ON
PRINTJOB        && _plineno opnieuw instellen op 0
FOR i=1 TO 32
  IF _plineno=0  && bij eerste regel op pagina
    *?
    ? "Begin van pagina ",_pageno
    ?
  ENDIF
  ?? "Pagina",_pageno," Regel",_plineno,i
  ?                    && gedwongen regeldoorvoer
NEXT i
ENDPRINTJOB
SET PRINTER OFF
CLOSE PRINTER
* De eerste twee afgedrukte pagina's zien er zo uit:
*
* Begin van pagina          1
* Pagina        1  Regel         2.00          1
* Pagina        1  Regel         3.00          2
* Pagina        1  Regel         4.00          3
* Pagina        1  Regel         5.00          4
* Pagina        1  Regel         6.00          5
* Pagina        1  Regel         7.00          6
* Pagina        1  Regel         8.00          7
* Pagina        1  Regel         9.00          8
*
* Begin van pagina          2
* Pagina        2  Regel         2.00          9
* Pagina        2  Regel         3.00         10
* Pagina        2  Regel         4.00         11
* Pagina        2  Regel         5.00         12
* Pagina        2  Regel         6.00         13
* Pagina        2  Regel         7.00         14
* Pagina        2  Regel         8.00         15
* Pagina        2  Regel         9.00         16
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

_pcolno, _plength, _ppitch, EJECT PAGE, ON PAGE, PCOL( ), PROW( )

_ploffset

Afdrukken

Geeft de breedte aan van de linkerkantlijn op een afgedrukte pagina of wijzigt deze.

Syntaxis

_ploffset = <*Nuitdr*>

<Nuitdr>

Het kolomnummer voor de linkermarge. Het geldige bereik loopt van 0 tot en met 254. U kunt voor <*Nuitdr*> een niet-geheel getal opgeven om de uitvoer bij een proportioneel lettertype nauwkeurig te plaatsen.

Standaardinstelling

De standaardinstelling voor _ploffset is 0. U kunt de standaardinstelling wijzigen met de parameter MARGIN in DBASEWIN.INI.

Beschrijving

Met _ploffset (linker pagina-offset) kunt u de afstand opgeven tussen de linkerkant van het papier en de linkerkant van het afdrukgebied. Met _lmargin kunt u de afstand tussen de linkerkant van het afdrukgebied en de linkermarge instellen. Als _ploffset bijvoorbeeld gelijk is aan 10 en _lmargin aan 5, wordt de uitvoer afgedrukt vanaf de vijftiende kolom.

De systeemgeheugenvariabele _ploffset is gelijk aan de waarde voor SET MARGIN. Als u de ene waarde wijzigt, geldt dat ook voor de andere waarde. Zie SET MARGIN voor meer informatie.

Voorbeeld

Zie het voorbeeld bij SET MARGIN. _ploffset komt overeen met SET MARGIN.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS, maar SET MARGIN wel.

Zie ook

_indent, _lmargin, SET MARGIN

_porientation — Afdrukken

Bepaalt de afdrukstand (staand of liggend).

Syntaxis

_porientation = <*Tuitdr*>

<Tuitdr>

De tekenuitdrukking "PORTRAIT" (staand) of "LANDSCAPE" (liggend).

Standaardinstelling

De standaardinstelling voor _porientation is de afdrukstand die is ingesteld met de optie **Printers** in het Configuratiescherm van Windows of in dBASE met de functie CHOOSEPRINTER( ). De standaardafdrukstand is staand.

Beschrijving

Met _porientation kunt u opgeven of liggend of staand moet worden afgedrukt. Staand betekent gewoonlijk dat de pagina langer is dan breed (het papierformaat A4 is bijvoorbeeld 297 mm hoog en 210 mm breed). Als u liggend afdrukt, is de pagina 90 graden gedraaid (breder dan lang).

De meeste printerstuurprogramma's ondersteunen liggend afdrukken. Geeft u echter liggend op voor een printer die alleen staand afdrukken ondersteunt, wordt gewoon staand afgedrukt en dat kan tot gevolg hebben dat delen van regels niet op papier verschijnen.

Het wijzigen van de afdrukstand brengt automatisch met zich mee dat _plength wordt aangepast.

Voorbeeld

Voor _porientation zijn twee mogelijkheden:

```
_porientation="PORTRAIT"     && staand
_porientation="LANDSCAPE"    && liggend
```

Als u de instelling wijzigt, gaat de nieuwe instelling in na het eerstvolgende pagina-einde.

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

_pdriver, _plength, _ppitch, SET PRINTER

_ppitch

Afdrukken

Bepaalt het aantal tekens per inch ("pitch") dat wordt afgedrukt.

Syntaxis

_ppitch = <*Tuitdr*>

<Tuitdr>

De tekenuitdrukking "pica", "elite", "condensed" of "default".

Standaardinstelling

De standaardinstelling voor _ppitch is "default". Dat is het aantal tekens per inch dat wordt bepaald door de instellingen van de printer. "Default" (standaard) betekent dat dBASE geen besturingscodes naar de printer heeft gestuurd om het aantal tekens per inch te wijzigen.

Beschrijving

Met _ppitch stelt u het aantal tekens per inch voor de printer in. De instelling voor _ppitch heeft tot gevolg dat een besturingscode aan de printer wordt doorgegeven. De printeraansturing kunt u instellen in het Configuratiescherm van Windows of met CHOOSEPRINTER( ).

Als u uitvoer naar de printer stuurt, wordt elk teken geplaatst aan de hand van het *coördinatenvlak*, een tweedimensionaal raster. Het coördinatenvlak bestaat uit tekencellen waarvan de breedte wordt bepaald door de waarde van _ppitch. De hoogte van elke tekencel wordt bepaald door de grootte van het font. Zie Hoofdstuk 16 in *Programmeren* voor meer informatie over het coördinatenvlak.

De volgende tabel geeft een overzicht van mogelijke waarden voor _ppitch.

Waarde voor _ppitch	Breedte van de tekencel
"pica"	2,54 mm (10 tekens/inch)
"elite"	2,11 mm (12 tekens/inch)
"condensed"	1,49 mm (17 tekens/inch)

Als u de waarde van andere systeemgeheugenvariabelen (zoals _lmargin, _rmargin en _ploffset) wijzigt, wordt bij de berekening van de celbreedte binnen het coördinatenvlak rekening gehouden met de huidige waarde van _ppitch, ongeacht of u afdrukt met een proportioneel of niet-proportioneel font.

Voorbeeld

In het volgende voorbeeld worden drie instellingen voor _ppitch gebruikt:

```
TpiOpsl=_ppitch  && huidige aantal tekens per inch opslaan
SET PRINTER ON
_ppitch="pica"
? "Het leven van Multatuli: 10 tekens per inch"
_ppitch="elite"
? "Het leven van Multatuli: 12 tekens per inch"
_ppitch="condensed"
? "Het leven van Multatuli: 17 tekens per inch"
_ppitch=TpiOpsl        && oorspronkelijke instelling herstellen
CLOSE PRINTER
```

_ppitch is niet geldig voor alle printers.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

_pdriver, _pquality, CHOOSEPRINTER( )

_pquality

Afdrukken

Bepaalt of de printer afdrukt in kwaliteit- of kladmodus. Wordt hoofdzakelijk gebruikt voor matrixprinters. De instelling voor _pquality heeft gewoonlijk geen effect op printers die geen kladmodus ondersteunen, zoals laser- en Postscript-printers.

Syntaxis

_pquality = <*Luitdr*>

<Luitdr>

De logische uitdrukking .T. voor kwaliteitmodus en .F. voor kladmodus.

Standaardinstelling

De standaardinstelling voor _pquality is .T. voor kwaliteitsmodus (dus onwaar (.F.) voor kladmodus).

Beschrijving

Met _pquality kunt u aangeven of de printer moet afdrukken in kwaliteit- of kladmodus. De kwaliteitmodus levert afdrukken van een hogere kwaliteit (een hogere resolutie) dan de kladmodus. Afhankelijk van de printer gaat het afdrukken in de kladmodus gewoonlijk echter wel sneller.

Voorbeeld

In het volgende voorbeeld worden de twee instellingen voor de afdrukkwaliteit gebruikt. De afdrukkwaliteit kan niet halverwege een pagina worden gewijzigd:

```
CLOSE PRINTER
set printer on
_pquality= .f.     && kladmodus
? "Het leven van Multatuli"
CLOSE PRINTER
_pquality= .t.     && kwaliteitmodus
? "Het leven van Multatuli"
CLOSE PRINTER
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

_pdriver, _ppitch

_pspacing

Afdrukken

Stelt de regelafstand in voor doorlopende uitvoer (streaming output).

Syntaxis

_pspacing = <*Nuitdr*>

<Nuitdr>

De waarde voor de regelafstand. Het geldige bereik loopt van 1 tot en met 3:

- 1 is enkele regelafstand.
- 2 is dubbele regelafstand. Tussen twee afgedrukte regels is een lege regel.
- 3 is driedubbele regelafstand. Tussen twee afgedrukte regels zijn twee lege regels.

De afstand tussen alinea's is een veelvoud van de hoogte van de zojuist afgedrukte regel en die hoogte is afhankelijk van het hoogste font in de regel. U kunt een niet-geheel getal opgeven voor <*Nuitdr*>.

Standaardinstelling

De standaardinstelling voor _pspacing is 1, ofwel een enkele regelafstand.

Beschrijving

Met _pspacing kunt u de regelafstand instellen in doorlopende uitvoer (streaming output) van commando's als ?, ?? en LIST. Zie Hoofdstuk 24 in *Programmeren* voor meer informatie over doorlopende en niet-doorlopende uitvoer. Met het commando ? kunt u een enkele lege regel invoegen in de uitvoer.

De waarde van _pspacing is ook van invloed op de kaders die worden weergegeven met het dBASE IV-commando DEFINE BOX. Als u bijvoorbeeld een kader definieert met een hoogte van 10 en voor _pspacing de waarde 2 opgeeft, wordt het kader afgedrukt met een hoogte van 20 regels.

Voorbeeld

In dit voorbeeld wordt _pspacing gebruikt om de regelafstand in afgedrukte tekst eerst 1, vervolgens 2 en tenslotte weer 1 te maken:

```
_pspacing=1
? "Jan Klaassen 1"
? "Katrijn 1"
_pspacing=2
? "Jan Klaassen 2"
? "Katrijn 2"
_pspacing=1
? "Jan Klaassen 1"
? "Katrijn 1"
* Uitvoer:
*         Jan Klaassen 1
```

```
*         Katrijn 1
*
*         Jan Klaassen 2
*
*         Katrijn 2
*         Jan Klaassen 1
*         Katrijn 1
```

U ziet dat _pspacing onmiddellijk van kracht wordt, zodat de dubbele regelafstand al verschijnt voor Jan Klaassen 2 en voor Katrijn 2.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

?, ??, _ppitch, DISPLAY, LIST

_rmargin

Afdrukken

Definieert de rechtermarge in uitvoer van de commando's ? en ?? (als _wrap waar is).

Syntaxis

_rmargin = <*Nuitdr*>

<Nuitdr>

Het kolomnummer van de rechtermarge. Het geldige bereik loopt van 0 tot en met 255. U kunt voor <*Nuitdr*> een niet-geheel getal opgeven om de uitvoer bij een proportioneel lettertype nauwkeurig te plaatsen.

Standaardinstelling

De standaardinstelling voor _rmargin is 79.

Beschrijving

Met _rmargin kunt u de rechtermarge instellen voor uitvoer van de commando's ? en ??. De waarde van _rmargin moet groter zijn dan de waarde van _lmargin of van _lmargin + _indent. Als _lmargin en _indent bijvoorbeeld beide 5 zijn, moet _rmargin groter zijn dan 10 om tenminste één kolom te tonen.

Als u uitvoer naar de printer stuurt, wordt elk teken geplaatst aan de hand van het *coördinatenvlak*, een tweedimensionaal raster. Het coördinatenvlak bestaat uit tekencellen waarvan de breedte wordt bepaald door de waarde van _ppitch. Zie de beschrijving van _ppitch voor een tabel met de mogelijke waarden van _ppitch. De hoogte van elke tekencel wordt bepaald door de grootte van het font. Zie Hoofdstuk 16 in *Programmeren* voor meer informatie over het coördinatenvlak.

Als u de waarde van _rmargin wijzigt, wordt bij de berekening van de celbreedte binnen het coördinatenvlak rekening gehouden met de huidige waarde van _ppitch, ongeacht of u afdrukt met een proportioneel of niet-proportioneel font. Als u ? zonder de optie STYLE gebruikt en alleen gehele getallen opgeeft voor de coördinaten, wordt een niet-proportioneel font gebruikt, en ziet de uitvoer er net zo uit als in dBASE IV.

Voorbeeld

In het volgende voorbeeld wordt _wrap ingesteld op waar en vervolgens worden de beide marges gewijzigd. Binnen de nieuwe marges wordt tekst weergegeven:

```
WrapOpsl=_wrap          && instelling voor _wrap opslaan
_wrap=.t.               && moet .t. zijn
LmargeOpsl=_lmargin     && linkermarge opslaan
_lmargin=5
RmargeOpsl=_rmargin     && rechtermarge opslaan
_rmargin=20
? "Amsterdam, die grote stad, die is gebouwd op palen."
* Tekst wordt nu met regelovergangen
* tussen de kolommen 5 en 20 geplaatst.
_rmargin=RmargeOpsl     && oorspronkelijke
_lmargin=LmargeOpsl     && instellingen
_wrap=WrapOpsl          && herstellen
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

?, ??, _alignment, _indent, _lmargin, _ploffset, _wrap, SET MARGIN

_tabs

Afdrukken

Stelt een of meer tabs in voor de uitvoer van de commando's ? en ??.

Syntaxis

_tabs = <*Tuitdr*>

<*Tuitdr*>

De lijst met kolomnummers voor tabs. Als u meerdere tabs instelt, moeten de kolomnummers oplopend zijn en van elkaar worden gescheiden door komma's. Plaats de volledige lijst tussen aanhalingstekens. U kunt voor <*Nuitdr*> niet-gehele waarden opgeven om de uitvoer bij een proportioneel lettertype nauwkeurig te plaatsen.

Standaardinstelling

De standaardinstelling voor _tabs is een lege tekenreeks ("").

Beschrijving

Met _tabs kunt u een of meer tabstops definiëren. Als _wrap waar is, worden tabstops op posities gelijk aan of groter dan _rmargin genegeerd.

Als u uitvoer naar de printer stuurt, wordt elk teken geplaatst aan de hand van het *coördinatenvlak*, een tweedimensionaal raster. Het coördinatenvlak bestaat uit tekencellen waarvan de breedte wordt bepaald door de waarde van _ppitch. Zie de beschrijving van _ppitch voor een tabel met de mogelijke waarden van _ppitch. De hoogte van elke tekencel wordt bepaald door de grootte van het font. Zie Hoofdstuk 16 in *Programmeren* voor meer informatie over het coördinatenvlak.

Als u de waarde van _tabs wijzigt, wordt bij de berekening van de celbreedte binnen het coördinatenvlak rekening gehouden met de huidige waarde van _ppitch, ongeacht of u afdrukt met een proportioneel of niet-proportioneel font. Als u ? zonder de optie STYLE gebruikt en alleen gehele getallen opgeeft voor de coördinaten, wordt een niet-proportioneel font gebruikt, en ziet de uitvoer er net zo uit als in dBASE IV.

Als u in de uitvoer van ? of ?? een tabteken, CHR(9), opneemt, gaat de uitvoer verder bij de volgende tabstop. Als het tabteken wordt doorgegeven op het moment dat de uitvoer al voorbij de laatste tabstop is, wordt de tab genegeerd en gaat de uitvoer gewoon verder in de volgende kolom.

Voorbeeld

In het volgende voorbeeld worden twee tabstops ingesteld op de posities 5 en 20. De array A wordt weergegeven vanaf de eerste tabstop en de positie wordt weergegeven bij de tweede tabstop:

```
_tabs="5,20"   && tabstops in kolommen 5 en 20
DECLARE A[2]
A[1]="Een"
A[2]="Twee"
FOR i=1 to 2
  ? chr(9),A[i],chr(9),ltrim(str(i))
NEXT i
* chr(9) is code voor tab.
* Uitvoer:
*     Een           1
*     Twee          2
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. In dBASE IV is de waarde van _tabs ook van invloed op de tekst-editor. In dBASE voor Windows is dat niet het geval. De tabstops voor de tekst-editor stelt u in met de waarde voor **Automatisch inspringen** in het kenmerkenvenster van het tekst-editorvenster.

Zie ook

?, ??, _indent, _lmargin, _rmargin, _wrap, CHR( ), MODIFY COMMAND

_wrap

Bepaalt of doorlopende uitvoer (streaming output) tussen de marges (_lmargin en _rmargin) en met regelovergangen wordt weergegeven.

Syntaxis

_wrap = <*Luitdr*>

<Luitdr>
De logische uitdrukking waar (.T.) of onwaar (.F.).

Standaardinstelling

De standaardinstelling voor _wrap is .F. (geen regelovergangen).

Beschrijving

Stel _wrap in op .T. om doorlopende uitvoer (streaming output) van commando's als ?, ?? en LIST tussen de marges weer te geven met regelovergangen. De marges stelt u in met _lmargin en _rmargin. Zie Hoofdstuk 24 in *Programmeren* voor meer informatie over doorlopende uitvoer.

Als u regelovergangen inschakelt, worden woorden (of getallen) die niet meer op de regel passen, aan het begin van de volgende regel geplaatst. Als u regelovergangen uitschakelt, loopt de tekst door voorbij de rechtermarge en wordt pas naar een nieuwe regel gegaan als in de tekst een regelterugloop/regeldoorvoer (CR/LF) wordt aangetroffen.

De instellingen voor _alignment, _indent, _lmargin en _rmargin hebben alleen effect als _wrap waar (.T.) is.

Als _wrap gelijk is aan .T., wordt doorlopende uitvoer (streaming output) opgeslagen in een buffer tot de huidige regel is afgedrukt of weergegeven. Als u uitvoer genereert met het commando ?, moet u daarna nog een commando ? opgeven, omda anders de laatste regel niet wordt weergegeven of afgedrukt.

Voorbeeld

In het volgende voorbeeld wordt een lange tekenreeks weergegeven met en zonder regelovergangen:

```
_lmargin=5
_rmargin=15
Treeks="Op een goede dag gingen Hans en Grietje naar het bos."
_wrap=.f.
? Treeks
_wrap=.t.
? Treeks
* Weergave zonder regelovergangen (_wrap = .F.):
* Op een goede dag gingen Hans en Grietje naar het bos.
* Weergave met regelovergangen (_wrap = .T.):
```

```
*       Op een
*       goede dag
*       gingen
*       Hans en
*       Grietje
*       naar het
*       bos.
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS. In dBASE IV is de waarde van _wrap ook van invloed op de tekst-editor. In dBASE voor Windows is dat niet het geval. Schakel regelovergangen in de tekst-editor in of uit met het aankruisvakje **Regelovergang** in het kenmerkenvenster van het tekst-editorvenster.

Zie ook

?, ??, _alignment, _indent, _lmargin, _ploffset, _rmargin, PRINTJOB...ENDPRINTJOB

Preprocessor-instructies

Hoofdstuk

Preprocessor-instructies

#define

Preprocessor-instructies

Definieert een identificatie (naam) die u kunt gebruiken bij het besturen van de compilatie van het programma, het definiëren van constanten en het maken van inline-functies.

Syntaxis

#define <*identificatie*> [<*vervangtekst*>]
#define <*identificatie*>(<*parameterlijst*>) <*vervangtekst met parameters*>

<*identificatie*>

Een naam. Geeft de tekst aan die moet worden vervangen als <*vervangtekst*> is opgegeven. De naam moet beginnen met een alfabetisch teken en mag verder bestaan uit elke willekeurige combinatie van alfabetische en numerieke tekens, hoofdletters en kleine letters. Er wordt geen onderscheid gemaakt tussen hoofdletters en kleine letters.

(<*parameterlijst*>)

Parameternamen die overeenkomen met argumenten die worden doorgegeven aan een inline-functie (pseudo-functie) die u maakt met #define <*identificatie*> (<*parameterlijst*>) <*vervangtekst*>. Als u meerdere parameters opgeeft, moet u ze van elkaar scheiden door middel van komma's.

<*vervangtekst*>

De tekst die overal <*identificatie*> moet vervangen. Als u <*vervangtekst*> opgeeft, controleert de preprocessor elke regel met broncode en worden identificaties overal vervangen door de opgegeven vervangtekst. <*vervangtekst*> kan elke willekeurige tekst zijn die deel uitmaakt van een dBASE-programma, zoals een tekenreeks, variabelenaam of een reeks commando's.

Beschrijving

De instructie #define definieert een identificatie en stelt u optioneel in staat voor de compilatie tekst in een programma te vervangen. Elke definitie met #define moet op een nieuwe regel beginnen en mag uit maximaal 4096 tekens bestaan.

Identificaties zijn alleen beschikbaar in het programma waarin ze zijn gedefinieerd. Met #include kunt u identificaties definiëren die in meerdere programma's beschikbaar zijn. Met #undef kunt u een identificatie verwijderen die niet langer als preprocessor-instructie wordt gebruikt.

Gewoonlijk wordt de instructie #define gebruikt voor de volgende doeleinden :

- Om een identificatie zonder vervangtekst te definiëren, zodat u die kunt gebruiken in combinatie met de instructie #ifdef of #ifndef.
- Om een identificatie te definiëren en daar vervangtekst aan toe te wijzen. De vervangtekst bestaat dan uit een constante waarde of een samengestelde uitdrukking die u kunt gebruiken in combinatie met de instructie #if.
- Om een inline-functie (pseudo-functie) te maken.

Instructies met #define hebben een hogere prioriteit dan geheugenvariabelen, ingebouwde commando's en functies en elk ander element dat dezelfde naam heeft als <*identificatie*>. Dit wordt gedemonstreerd in de volgende voorbeelden.

```
* een geheugenvariabele negeren
#define mWaarde 10
mWaarde = 25                          && Geeft fout als resultaat
                                      && omdat mWaarde constante is

mVariabele = 25
#define mVariabele 10
? mVariabele                          && Geeft 10 weer
* ingebouwde functie negeren
#define upper(x) "? 'slecht idee' "   && UPPER( ) is
                                      && ingebouwde functie
? upper("mVariabele")                 && Geeft 'slecht idee' als
                                      && resultaat
```

Zie Hoofdstuk 7 in *Programmeren* voor meer informatie over werken met de instructie #define.

Identificaties definiëren

Als u een identificatie definieert zonder vervangtekst, kunt u het symbool gebruiken in combinatie met de instructie #ifdef of #ifndef om op het bestaan van de identificatie te testen. Op die manier kunt u voorwaarden instellen voor het al of niet opnemen van code tijdens de compilatie, zoals in het volgende voorbeeld :

```
#define eersteversie
...
#ifdef eersteversie
   <compileer deze instructies voor het instellen van het programma>
   <bijvoorbeeld om instructies voor foutopsporing op te nemen>
#endif
```

Als het programma na enige tijd zonder fouten werkt en de instructies voor het opsporen van fouten zijn niet langer nodig, kunt u de instructie met #define verwijderen of omzetten in een commentaar.

Identificaties definiëren die constanten voorstellen

U kunt een constante waarde of uitdrukking toewijzen aan een identificatie als u wilt dat de preprocessor de identificatie overal vervangt door de opgegeven waarde of uitdrukking. Dit wordt gedemonstreerd in het volgende voorbeeld.

```
#define tBedrijf "dOplossingen BV."
...
? tBedrijf                    && "dOplossingen BV." tonen
```

Deze zoek-en-vervang-mogelijkheid, die ook wel *macro-uitbreiding* wordt genoemd, kan de code stroomlijnen en beter leesbaar maken, omdat u een regelmatig gebruikte constante of een samengestelde uitdrukking kunt vervangen door een enkele identificatie. Verder kunt u op deze manier zeer eenvoudig de waarde van een constante in het programma wijzigen, namelijk door alleen de definitie van de constante te wijzigen, in plaats van de constante overal in het programma te veranderen.

Als u een identificatie slechts in bepaalde delen van een programma wilt vervangen, voegt u op de plaats in het programma waar u niet verder wilt gaan met vervangen #undef <*identificatie*> in.

Inline-functies maken

Als de preprocessor een functie-aanroep aantreft die overeenkomt met een functiedefinitie, wordt de aanroep vervangen door de functietekst en worden de argumenten voor de functie-aanroep in de vervangtekst ingevoegd. Het volgende voorbeeld laat dit zien.

```
#define Gem(get1,get2) (get1+get2)/2
...
nGetal1=20
nGetal2=40
? Gem(nGetal1,nGetal2)   && Geeft 30 weer
```

In tegenstelling tot gewone door de gebruiker gedefinieerde functies, moet het aantal argumenten dat uit de functie-aanroep wordt doorgegeven gelijk zijn aan het aantal argumenten dat is gedefinieerd in de instructie met #define.

Opmerking Er bestaan belangrijke verschillen tussen de manier waarop dBASE inline-functies en door de gebruiker gedefinieerde functie evalueert. Als u deze verschillen niet goed begrijpt, geven uw inline-functies misschien niet altijd de waarden als resultaat die u verwacht. Let er in elk geval op dat de vervangtekst altijd tussen haakjes staat, zodat bij de evaluatie de juiste volgorde wordt gehanteerd. Zie Hoofdstuk 7 in *Programmeren* voor meer informatie.

Preprocessor-macro's nesten

U kunt preprocessor-macro's nesten. Dit houdt in dat een macro kan worden uitgebreid binnen een andere macro. In het volgende voorbeeld ziet u hoe de derde macrodefinitie afhankelijk is van de eerste twee:

```
#define K_CR CHR(13)
#define K_LF CHR(10)
#define K_CRLF K_CR + K_LF
```

Voorbeeld

In de volgende voorbeelden wordt #define gebruikt om font-parameters op te geven zodat het gewenste font kan worden ingesteld met een enkele identificatie :

```
#define GROTELETTERS fontname "Roman",fontheight 20,;
  fontwidth 22
#define TEKSTFONT fontname "Roman",fontheight 11,;
  fontwidth 6
#define INVOERFONT fontname "Roman",fontheight 16,;
  fontitalic .t.
```

In de volgende voorbeelden wordt #define gebruikt om een functie te definiëren die alle als parameters doorgegeven waarden inkort, en om een vooringestelde veldenlijst te definiëren.

```
#define LRINKORTEN(x) LTRIM(RTRIM(x))
#define VELDENLIJST "Vluchtnr,Van,Naar,Datum,;
  Vertrek,Aankomst,Prijs"
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

#if, #ifdef, #ifndef, #include, #undef

#if

Preprocessor-instructies

Bestuurt de voorwaardelijke compilatie van code op basis van de waarde van een met #define toegewezen identificatie.

Syntaxis

#if <*voorwaarde*>
    <*instructies1*>
[#else
    <*instructies2*>]
#endif

<voorwaarde>

Een logische uitdrukking, met een gedefinieerde identificatie, die waar of onwaar als resultaat geeft.

<instructies1>
Eén of meer programmaregels met een willekeurige combinatie van commando's, functies en preprocessor-instructies. Deze regels worden gecompileerd als *<voorwaarde>* waar is.

#else <instructies2>
Geeft aan welke programmaregels moeten worden gecompileerd als <voorwaarde> onwaar is.

Beschrijving

Gebruik de instructie #if om voorwaardelijk delen van de broncode te compileren op basis van de waarde van *<identificatie>*. Ook de twee instructies #ifdef en #ifndef worden gebruikt voor het voorwaardelijke compileren van delen van de code. In tegenstelling tot de instructie #if testen deze laatste twee alleen op de aanwezigheid van een identificatie en niet op diens waarde.

Voorwaardelijke compilatie is handig als u over verschillende versies van hetzelfde programma wilt beschikken en voor het opsporen van fouten. Het volgende voorbeeld laat dit zien.

```
#define mInvmod = "onaf"
...
#if mInvmod = "af"
   <Instructies voor Inventarismodule komen hier>
#else
   mMelding = "Inventarismodule is nog niet af"
#endif
```

Zie Hoofdstuk 7 in *Programmeren* voor meer informatie over het werken met de instructie #if.

Voorbeeld

In het volgende voorbeeld wordt #if/#else gebruikt om te bepalen of een preprocessor-instructie is gedefinieerd die een veldenlijst bevat. Als dat niet het geval is, wordt #define gebruikt om een veldenlijst op te geven :

```
#define VELDENLIJST "Bedrijf, Contact, Telefoon"
*
*
*  rest van het programma
*
CLEAR
USE Afnemers
#if VELDENLIJST = "Bedrijf, Contact, Telefoon"
  LIST TO PRINT
#else
  #define VELDENLIJST "Bedrijf, Contact, Telefoon"
  LIST TO PRINT
#endif
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS en dBASE IV, maar wel in de dBASE Compiler voor DOS.

Zie ook

#define, #ifdef, #ifndef

#ifdef — Preprocessor-instructies

Bestuurt voorwaardelijke compilatie van code op basis van het bestaan van een identificatie die is toegewezen met #define of het bestaan van de identificatie __dbasewin__.

Syntaxis

#ifdef *<identificatie>*
    <instructies1>
  [#else
    <instructies2>]
  #endif

<identificatie>

De identificatie waarvan u het bestaan wilt testen. *<identificatie>* is gedefinieerd met de instructie #define.

<instructies1>

Eén of meer programmaregels met een willekeurige combinatie van commando's, functies en preprocessor-instructies. Deze regels worden gecompileerd als *<identificatie>* is gedefinieerd.

#else <instructies2>

Geeft aan welke programmaregels moeten worden gecompileerd als <identificatie> niet is gedefinieerd.

Beschrijving

Gebruik de instructie #ifdef om voorwaardelijke delen van de broncode te compileren. Als *<identificatie>* is gedefinieerd met #define, wordt de code die u opgeeft voor *<instructies1>* gecompileerd. Zo niet, dan wordt de code achter #else gecompileerd.

Voorwaardelijke compilatie is handig als u over verschillende versies van hetzelfde programma wilt beschikken en voor het opsporen van fouten. Het volgende voorbeeld laat dit zien.

```
#define InvMod
#ifdef InvMod
   <Instructies voor Inventarismodule komen hier>
```

```
#else
   mMelding = "Inventarismodule is nog niet af"
#endif
```

Met #ifdef in combinatie met de identificatie __dbasewin__ kunt u bepalen of dBASE voor Windows of een oudere versie van dBASE wordt uitgevoerd. Het volgende voorbeeld laat dit zien.

```
#ifdef __dbasewin__
   <code die alleen geldig is in dBASE voor Windows>
#else
   <code die geldig is in oudere versies van dBASE>
#endif
```

Zie Hoofdstuk 7 in *Programmeren* voor meer informatie over werken met de instructie #ifdef.

Voorbeeld

In het volgende voorbeeld wordt #ifdef gebruikt om te controleren of een bepaald font is gedefinieerd voordat met afdrukken wordt begonnen. Als dat niet zo is, wordt het font gedefinieerd en wordt het afdrukken gestart.

```
#ifdef GROTELETTERS
  LIST TO PRINT
#else
  #define GROTELETTERS fontname "Roman",;
    fontheight 20,fontwidth 22
  LIST TO PRINT
#endif
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

#define, #if, #ifndef

#ifndef

Preprocessor-instructies

Bestuurt voorwaardelijke compilatie van code op basis van het bestaan van een identificatie die is toegewezen met #define.

Syntaxis

#ifndef *<identificatie>*
    <instructies1>
[#else
    <instructies2>]
#endif

<identificatie>

De identificatie waarvan u het bestaan wilt testen. *<identificatie>* is gedefinieerd met de instructie #define.

<instructies1>

Eén of meer programmaregels met een willekeurige combinatie van commando's, functies en preprocessor-instructies. Deze regels worden gecompileerd als *<identificatie>* niet is gedefinieerd.

#else *<instructies2>*

Geeft aan welke programmaregels moeten worden gecompileerd als <identificatie> is gedefinieerd.

Beschrijving

Gebruik de instructie #ifndef om voorwaardelijk delen van de broncode te compileren. Als <identificatie> niet is gedefinieerd met #define, wordt de code gecompileerd die u opgeeft bij *<instructies1>*, anders wordt de code na #else gecompileerd.

Voorwaardelijke compilatie is handig als u over verschillende versies van hetzelfde programma wilt beschikken en voor het opsporen van fouten. Het volgende voorbeeld laat dit zien.

```
#ifndef InvMod
   mMelding = "Inventarismodule is nog niet klaar"
#else
   <Instructies voor Inventarismodule komen hier>
#endif
```

Zie Hoofdstuk 7 in *Programmeren* voor meer informatie over het gebruik van de instructie #ifndef.

Voorbeeld

In het volgende voorbeeld wordt #ifndef gebruikt om te controleren of een bepaald font is gedefinieerd. Als dat niet zo is, wordt het gedefinieerd met #define. Het deel na #else dient om te zorgen dat het font juist is gedefinieerd door de definitie eerst te annuleren en vervolgens "GROTELETTERS" te definiëren:

```
#ifndef GROTELETTERS
  #define GROTELETTERS fontname "Roman",;
    fontheight 20,fontwidth 22
#else
  #undef GROTELETTERS
  #define GROTELETTERS fontname "Roman",;
    fontheight 20,fontwidth 22
#endif
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

#define, #if, #ifdef

#include

Preprocessor-instructies

Voegt de inhoud van een opgegeven bronbestand (een zogenaamd include- of header-bestand) in in het huidige programmabestand op de plaats van de #include-instructie.

Syntaxis

#include <*bestandsnaam*>| "<*bestandsnaam*>"

<*bestandsnaam*> | "<*bestandsnaam*>"

De naam van het bestand (eventueel inclusief pad) waarvan de inhoud moet worden ingevoegd in het huidige programmabestand. De bestandsnaam mag tussen aanhalingstekens staan, maar dat hoeft niet. Een include-bestand heeft meestal de extensie .h.

Als u <*bestandsnaam*> zonder pad opgeeft, gebruikt de preprocessor de volgende zoekvolgorde :

1. In de huidige directory wordt gezocht naar het bestand zoals u het hebt opgegeven.
2. Als u geen extensie hebt opgegeven, wordt de extensie .h gebruikt en wordt in de huidige directory gezocht.
3. Als het bestand niet wordt aangetroffen in de huidige directory, wordt gezocht in de directory <*hoofddirectory*>\ INCLUDE. (De hoofddirectory wordt aangegeven door de systeemgeheugenvariabele _dbwinhome.)
4. Als het bestand niet wordt aangetroffen in de huidige directory of in <*hoofddirectory*>\INCLUDE, wordt gezocht in de directory die wordt aangegeven door de DOS-omgevingsvariabele INCLUDE.

Beschrijving

Identificaties zijn meestal alleen beschikbaar in het programma waarin ze zijn gedefinieerd. Als u een verzameling identificaties in meerdere programma's wilt gebruiken, kunt u de #define-instructies opslaan in een bestand en dat bestand (en dus de identificaties) in de andere programma's opnemen met de instructie #include. Het bestand met de #define-instructies wordt een include-bestand genoemd. Een include-bestand met de naam IDENT.H zou bijvoorbeeld de volgende instructies kunnen bevatten :

```
#define tBedrijf="dOplossingen BV."
#define nVerkoop=40000
#define tEigenRegio="ZW"
```

Een programma dat van deze identificaties gebruik maakt, zou dan de volgende regel moeten bevatten :

```
#include ident.h
```

Als alle #define-instructies in één bestand staan, zijn die gemakkelijk te onderhouden. Als u een van de instructies wilt wijzigen, hoeft u alleen het include-bestand te wijzigen. De programmabestanden waarin de #define-instructies worden gebruikt, blijven ongewijzigd. Nadat u het include-bestand hebt gewijzigd, compileert u het programmabestand opnieuw om de wijzigingen door te voeren.

Als u bijvoorbeeld in de volledige applicatie de waarde voor de eigen regio wilt wijzigen van "ZW" in "ZO", hoeft u alleen maar de #define-instructie in IDENT.H te wijzigen.

Zie Hoofdstuk 7 in *Programmeren* voor meer informatie over include-bestanden.

Voorbeeld

In het volgende voorbeeld wordt de inhoud van een door de gebruiker gedefinieerd instellingenbestand ingevoegd in een actief dBASE-programma. Het bestand "instell.h" kan definities bevatten van #define-identificaties die in verschillende programma's worden toegepast :

```
#include "c:\dbasewin\setup\instell.h"
```

Overdraagbaarheid

Wordt niet ondersteund dBASE III PLUS en dBASE IV, maar wordt wel ondersteund door de dBASE Compiler voor DOS.

Zie ook

#define, GETFILE()

#pragma

Preprocessor-instructies

Stelt opties voor de compiler in.

Syntaxis

#pragma <*coverage* (*on*|*off*)>

#pragma <*coverage* (*on*|*off*)>

Schakelt dekkingsanalyse in of uit. Dekkingsanalyse levert informatie over welke programmaregels worden uitgevoerd en welke niet.

Beschrijving

Gebruik in het programma #pragma om telkens wanneer het programma wordt gestart, een dekkingsanalyse uit te voeren. Dit is een alternatief voor het commando SET COVERAGE ON in het Commandovenster voordat u het programma compileert en uitvoert.

Dit is tevens de enige mogelijkheid om vanuit een programma dekkingsanalyse te starten. Als u in een programma het commando SET COVERAGE ON gebruikt, wordt voor dat programma geen dekkingsbestand gemaakt of bijgewerkt. Zie SET COVERAGE voor meer informatie over dekkingsbestanden.

Zie Hoofdstuk 7 in *Programmeren* voor meer informatie over preprocessor-instructies.

Voorbeeld

In het volgende voorbeeld wordt #pragma gebruikt om dekkingsanalyse in te schakelen als naar fouten wordt gezocht :

```
#ifdef DEBUG
  #pragma coverage(on)
#else
  #pragma coverage(off)
#endif
```

Overdraagbaarheid

Wordt niet ondersteund in dBASE IV of dBASE III PLUS.

Zie ook

#define, SET COVERAGE

#undef

Preprocessor-instructies

Verwijdert de huidige definitie van de opgegeven identificatie die eerder is gedefinieerd met #define.

Syntaxis

#undef *<identificatie>*

<identificatie>

De identificatie waarvan de definitie moet worden verwijderd.

Beschrijving

De instructie #undef verwijdert de definitie van een identificatie die eerder is gedefinieerd met de instructie #define. Als u #define gebruikt met *<vervangtekst>*, vervangt de preprocessor de identificatie overal door de vervangtekst tot een instructie met #undef wordt aangetroffen voor dezelfde identificatie. Als u een identificatie dus alleen in een deel van een programma wilt vervangen, kunt u op de plaats waar u met vervangen wilt stoppen de instructie #undef *<identificatie>* invoegen.

Voorbeeld

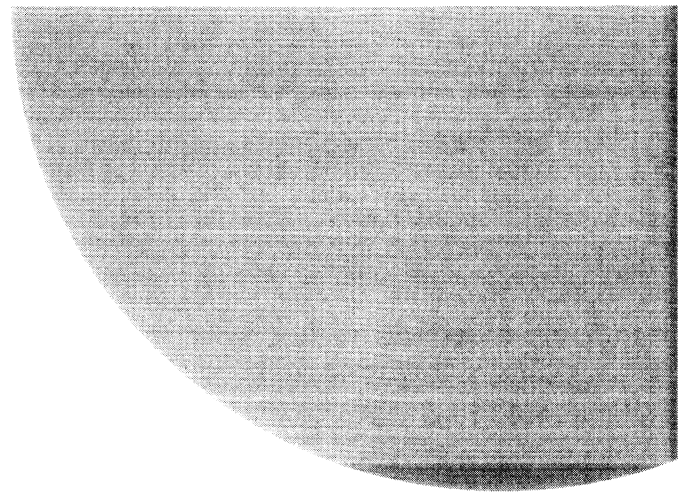
Zie #ifndef voor een voorbeeld met #undef.

Overdraagbaarheid

Wordt niet ondersteund in dBASE III PLUS.

Zie ook

#define



Klassen

Hoofdstuk

7

Klassen

CLASS ARRAY

Een object waarin meerdere geheugenvariabelen kunnen worden opgeslagen. De geheugenvariabelen kunnen individueel worden benaderd.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Array. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Add( )	Geen	Voegt elementen toe aan een eendimensionaal array-object.
ClassName	ARRAY	Geeft de klasse van de array aan.
Delete( )	Geen	Verwijdert een element uit een eendimensionaal array-object, of verwijdert een rij of kolom uit een tweedimensionaal array-object.
Dimensions	0	Geeft het aantal dimensies van het array-object aan.
Dir( )	Geen	Slaat de naam, de lengte, het datumstempel, het tijdstempel en de DOS-attributen van bestanden op in een array-object.
Element( )	Geen	Geeft het nummer van een opgegeven element in het array-object als resultaat.
Fields( )	Geen	Slaat structuurinformatie over de huidige tabel op in het array-object.
Fill( )	Geen	Voegt een opgegeven waarde in op een of meer plaatsen in een array-object.
Grow( )	Geen	Voegt elementen toe aan het array-object.
Insert( )	Geen	Voegt een element in in een eendimensionaal array-object, of voegt een rij of een kolom met elementen in in een tweedimensionaal array-object.
Resize( )	Geen	Wijzigt het aantal elementen in het array-object.
Scan( )	Geen	Zoekt in een array-object naar een opgegeven waarde.
Size	Geen	Wijzigt het aantal elementen in een array-object.

Kenmerk	Standaardinstelling	Beschrijving
Sort( )	Geen	Sorteert de elementen in een eendimensionaal array-object, of sorteert rijen in een tweedimensionaal array-object.
Subscript( )	Geen	Geeft het rijnummer of het kolomnummer van een opgegeven element in een array-object als resultaat.

Beschrijving

In een array-object kunt u meerdere, met elkaar samenhangende elementen opslaan in het geheugen. Zie Hoofdstuk 10 in *Programmeren* voor meer informatie over het werken met array-objecten.

Als u een array-object maakt met de operator NEW, kunt u twee parameters opgeven:

- *<rijen Nuitdr>* — Het aantal elementen als het array-object eendimensionaal is, of het aantal rijen als het array-object tweedimensionaal is.
- *<kolommen Nuitdr>* — Het aantal elementen in elke rij. Het argument *<kolommen Nuitdr>* wordt alleen gebruikt als u een tweedimensionaal array-object maakt.

De volgende instructie maakt bijvoorbeeld een tweedimensionaal array-object met tien rijen en twintig kolommen in elke rij:

```
MijnArray = NEW ARRAY(10, 20)
```

Opmerking Het is niet mogelijk een array-object te maken met DEFINE. In plaats daarvan kunt u de operator NEW gebruiken. In het volgende voorbeeld wordt een array-object gemaakt met de operator NEW:

```
Array1 = NEW ARRAY(10, 20)   && Maak array-object
DEFINE ARRAY Array1          && Geeft fout als resultaat
```

U kunt een array-object ook maken met het commando DECLARE:

```
DECLARE Array1[10]           && Maak array-object Array1
```

Voorbeeld

In het volgende voorbeeld wordt een formulier gemaakt met één opdrachtknop voor elk veld in een tabel. De veldnamen worden opgeslagen in een array-object met de naam AKnoppen:

```
LOCAL loVelden
loVelden = NEW VeldenForm()
loVelden.OPEN()
CLASS VeldenForm OF FORM
  USE ?
  * Maak een knop voor elk veld
  AKnoppen = NEW ARRAY(1)
    FOR i = 1 TO FCOUNT()
      AKnoppen.ADD(1)
      AKnoppen[i]          = NEW PUSHBUTTON(THIS)
      AKnoppen[i].text     = FIELD(i)
      AKnoppen[i].top      = i * 2
      AKnoppen[i].left     =  5
      AKnoppen[i].width    = 20
    NEXT
ENDCLASS
```

Zie ook

DECLARE, PUBLIC

CLASS BROWSE

Een hulpmiddel voor het wijzigen van gegevens. Meerdere records worden weergegeven in een opmaak die uit rijen en kolommen bestaat en enkele records in een opmaak die uit kolommen bestaat. Een browse-object is een bladerobject.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Browse. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Alias	Lege tekenreeks	Bepaalt het tabelbestand waartoe toegang moet worden verkregen.
Append	.T.	Bepaalt of records kunnen worden toegevoegd.
Before	Geen	Geeft aan welk object voor het bladerobject komt in de tabvolgorde van het hoofdformulier.
ClassName	BROWSE	Geeft de klasse van het bladerobject aan.
ColorNormal	N/W	Bepaalt de kleur van het bladerobject.
Delete	.T.	Bepaalt of records kunnen worden gemarkeerd voor verwijdering.
Enabled	.T.	Bepaalt of het bladerobject kan worden geselecteerd.
Fields	Lege tekenreeks	Bepaalt de velden die worden weergegeven en de veldopties die voor elk veld van kracht zijn.
FieldWidth	Geen	Bepaalt de breedte van een tekenveld in een bladerobject.
Follow	.T.	Bepaalt of de weergave een record volgt naar zijn nieuwe positie in de indexvolgorde als een sleutelveldwaarde is gewijzigd.
FontBold	.F.	Bepaalt of tekens vet worden weergegeven.
FontItalic	.F.	Bepaalt of tekens cursief worden weergegeven.
FontName	MS Sans Serif	Bepaalt het font voor de weergegeven tekens.
FontSize	8.00	Bepaalt de grootte van het weergegeven font in punten.
Height	8.00	Bepaalt de hoogte.
HelpFile	Lege tekenreeks	Geeft een Windows-Helpbestand (.HLP-bestand) aan dat contèxtgevoelige Help bevat.
HelpID	Lege tekenreeks	Geeft de contextreeks of het contextnummer aan van een Help-onderwerp in een Windows-Helpbestand (.HLP-bestand).
hWnd	Geen	Geeft de object-handle van het bladerobject als resultaat.
ID	-1	Geeft het bladerobject aan door middel van een numerieke waarde.
Left	0	Geeft de positie van de linkerkant van het kader aan.

Kenmerk	Standaardinstelling	Beschrijving
Mode	0	Geeft de weergavemodus aan.
Modify	.T.	Bepaalt of de gebruiker records kan wijzigen.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op het bladerobject staat.
Move( )	Geen	Verplaatst het bladerobject of wijzigt de grootte van het bladerobject.
Name	BLADEREN1	Geeft de naam aan van het bladerobject.
OnAppend	Geen	Voert een subroutine uit als een *record* wordt toegevoegd aan de tabel.
OnChange	Geen	Voert een subroutine uit als de gebruiker een waarde wijzigt.
OnGotFocus	Geen	Voert een subroutine uit als het bladerobject de focus krijgt.
OnHelp	Geen	Voert een subroutine uit als de gebruiker op *F1* drukt.
OnLeftDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het bladerobject.
OnLeftMouseDown	Geen	Voert een subroutine uit als de gebruiker klikt op het bladerobject.
OnLeftMouseUp	Geen	Voert een subroutine uit als de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op het bladerobject staat.
OnLostFocus	Geen	Voert een subroutine uit als de focus het object verlaat.
OnMiddleDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het bladerobject met de middelste muisknop.
OnMiddleMouseDown	Geen	Voert een subroutine uit als de gebruiker op het bladerobject klikt met de middelste muisknop.
OnMiddleMouseUp	Geen	Voert een subroutine uit als de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op het bladerobject staat.
OnMouseMove	Geen	Voert een subroutine uit als de gebruiker de aanwijzer over het bladerobject beweegt.
OnNavigate	Geen	Voert een subroutine uit als de gebruiker naar een ander record gaat.
OnOpen	Geen	Voert een subroutine uit als het hoofdformulier wordt geopend.
OnRightDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het bladerobject met de rechtermuisknop.
OnRightMouseDown	Geen	Voert een subroutine uit als de gebruiker rechtsklikt op het bladerobject.
OnRightMouseUp	Geen	Voert een subroutine uit als de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op het bladerobject staat.
Parent	Geen	Een objectverwijzing die naar het hoofdformulier wijst.
Release( )	Geen	Verwijdert de definitie van het bladerobject uit het geheugen.
SetFocus( )	Geen	Verplaatst de focus naar het bladerobject.
ShowDeleted	.T.	Bepaalt of de kolom met wismarkeringen van het bladerobject wordt weergegeven.

Kenmerk	Standaardinstelling	Beschrijving
ShowHeading	.T.	Bepaalt of boven aan elke kolom in bladerobject de veldnamen worden weergegeven.
ShowRecNo	.T.	Bepaalt of de kolom met recordnummers van het bladerobject wordt weergegeven.
StatusMessage	Lege tekenreeks	Geeft een melding aan om te tonen op de statusbalk.
TabStop	.T.	Bepaalt of de gebruiker de focus naar het bladerobject kan verplaatsen door op *Tab* of *Shift+Tab* te drukken.
Text	Geen	Geeft een tekenreeks aan die wordt weergegeven op de titelbalk.
Toggle	.T.	Bepaalt of de gebruiker kan schakelen tussen weergavemodi.
Top	Geen	Stelt de positie van de bovenkant van het kader in.
Visible	.T.	Bepaalt of het bladerobject zichtbaar of verborgen is.
When	.T.	Geeft een voorwaarde aan die waar moet zijn voordat de gebruiker de focus naar het bladerobject kan verplaatsen.
Width	Geen	Stelt de breedte in.

Beschrijving

Gebruik een bladerobject om het mogelijk te maken records uit een tabel weer te geven en te wijzigen. Een bladerobject biedt de meeste mogelijkheden en opties die de commando's BROWSE en EDIT bieden.

Als u één record tegelijk wilt weergeven, moet u voor het kenmerk Mode 1 of 2 instellen. Gebruik Mode 1 voor formulieropmaak en 2 voor opmaak in kolommen. Als u records uit meerdere records tegelijk wilt weergeven, moet u voor Mode 0 instellen. Als u .T. opgeeft voor het kenmerk Toggle, kan de gebruiker tussen de drie verschillende modi schakelen door op F2 te drukken.

Twee kenmerken bepalen welke tabel wordt weergegeven in het bladerobject.

- Het kenmerk View van het hoofdformulier
- Het kenmerk Alias van het bladerobject

Het kenmerk View gebruikt een query als basis voor het formulier. De query wordt automatisch aangeroepen als u het formulier opent. Het kenmerk Alias bepaalt welke door de query geopende tabel wordt weergegeven. Een *alias* is een alternatieve naam voor een geopend tabelbestand en kan bestaan uit de volgende onderdelen:

- Een naam die u opgeeft met de optie ALIAS van het commando USE.
- De naam van het tabelbestand (als u geen alias aan de tabel hebt toegewezen).
- De letter voor het werkgebied van de tabel.
- Het nummer voor het werkgebied van de tabel, voorafgegaan door een onderstrepingsteken (_).

Een bladerobject wordt vaak gebruikt om subrecords weer te geven in een formulier voor meerdere tabellen. Als het hoofdformulier bijvoorbeeld is gebaseerd op een query die twee of meer bestanden opent in hoofd-subrelatie, kunt u de subtabel opgeven met

het kenmerk Alias. Zie Alias, SELECT en USE voor meer informatie over aliassen en werkgebieden.

Met het kenmerk Fields kunt u opgeven welke individuele velden u wilt weergeven. Als het formulier van het bladerobject bijvoorbeeld is gebaseerd op een query, kunt u Fields gebruiken om velden weer te geven uit elke tabel die in de query wordt gebruikt. (U moet met Alias een bestand opgeven voordat u Fields kunt gebruiken.)

De grootte van een bladerobject stelt u in met de clausule FROM...TO van de DEFINE-instructie of met de kenmerken Height en Width.

Als u een bladerobject maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing naar het hoofdformulier.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van het nieuwe bladerobject. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een bladerobject om in het formulier weer te geven:

```
MijnForm = NEW FORM()
MijnBlader = NEW BROWSE(MijnForm, "Bladerobject")
```

Het kenmerk Name van het nieuwe bladerobject bevat "Bladerobject".

Opmerking U kunt een waarde opgeven voor het kenmerk View in het dialoogvenster **Weergave kiezen**, waarin u een query of een tabel kunt kiezen. Klik op de hulpmiddelenknop naast het weergave-element in het kenmerkenvenster om toegang te krijgen tot het dialoogvenster **Weergave kiezen**.

U kunt een waarde opgeven voor Alias met het dialoogvenster **Alias kiezen**, waarin een lijst wordt getoond van alle geopende tabellen. Klik op de hulpmiddelenknop naast het Alias-element in het kenmerkenvenster om toegang te krijgen tot het dialoogvenster **Alias kiezen**.

U kunt een waarde opgeven voor Fields met het dialoogvenster **Veld kiezen**, waarin een lijst wordt getoond van alle velden in de geopende tabellen. Klik op de hulpmiddelenknop naast het Fields-element in het kenmerkenvenster om toegang te krijgen tot het dialoogvenster **Veld kiezen**.

Als u het kenmerkenvenster gebruikt, moet u de waarde voor View, Alias of Fields niet tussen aanhalingstekens plaatsen. Als u dat wel doet, wordt een foutmelding getoond.

Voorbeeld

In het volgende voorbeeld wordt een formulier gedefinieerd dat een bladerobject van de tabel Contact bevat. De opdrachtknop Sluiten geeft de gebruiker een alternatieve manier om het formulier te sluiten:

```
LOCAL f
f=NEW InvoerForm()
f.OPEN()
CLASS InvoerForm OF FORM
  this.Top=2
  this.Left=2
  this.Width=50
```

```
this.Height=20
this.View= "Contact.DBF"
this.Text= "Wijzigen naar behoefte"
DEFINE BROWSE Bladerl OF THIS;
  PROPERTY;
    Alias "Contact",;
    Fields "CompCode, Contact",;
    Top 4,;
    Left 3,;
    Width 32,;
    Height 12,;
    Delete .T.,;
    StatusMessage "Bladeren in tabel Contact",;
    Toggle .F.  && Schakelen tussen weergavemodi onmogelijk maken
DEFINE TEXT Tekstl OF THIS;
  PROPERTY;
    Text "Contactpersonen in tabel Contact",;
    Width 46,;
    Top 1,;
    Left 0,;
    Alignment 1,;
    Height 2.50,;
    FontSize 12.00,;
    FontBold .T.
DEFINE PUSHBUTTON Sluiten OF THIS;
  PROPERTY Text "Sluiten", Height 2,;
    Widt 14, Top 17, Left 14,;
    OnClick {;Form.Close()}
ENDCLASS
```

Zie ook

DEFINE, BROWSE, EDIT, OPEN FORM, RELEASE OBJECT

CLASS CHECKBOX

Een object waarmee de gebruiker logische waarden (waar, .T. en onwaar, .F.) kan wijzigen. Een *checkbox* is een aankruisvakje.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Checkbox. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Before	Geen	Bepaalt welk object voor het aankruisvakje komt in de tabvolgorde van het hoofdformulier.
ClassName	CHECKBOX	Geeft de klasse van het aankruisvakje aan.
ColorNormal	N/W	Bepaalt de kleur van het aankruisvakje.
DataLink	Lege tekenreeks	Koppelt het aankruisvakje aan een veld.

Kenmerk	Standaardinstelling	Beschrijving
Enabled	.T.	Bepaalt of het aankruisvakje kan worden geselecteerd.
FontBold	.F.	Geeft aan dat de tekens in de beschrijving bij het aankruisvakje vet worden weergegeven.
FontItalic	.F.	Bepaalt of tekens cursief worden weergegeven.
FontName	MS Sans Serif	Bepaalt het font voor de weergegeven tekens.
FontSize	Geen	Bepaalt de grootte van het weergegeven font in punten.
FontStrikeOut	.F.	Bepaalt of tekens doorgehaald worden weergegeven.
FontUnderline	.F.	Bepaalt of tekens onderstreept worden weergegeven.
Group	.F.	Begint een objectgroep in het hoofdformulier.
Height	Geen	Bepaalt de hoogte.
HelpFile	Lege tekenreeks	Geeft een Windows-Helpbestand (.HLP-bestand) aan dat contextgevoelige Help bevat.
HelpID	Lege tekenreeks	Geeft de contextreeks of het contextnummer aan van een Help-onderwerp in een Windows-Helpbestand (.HLP-bestand).
hWnd	Geen	Geeft de object-handle van het aankruisvakje-object als resultaat.
ID	-1	Geeft het aankruisvakje aan door middel van een numerieke waarde.
Left	Geen	Geeft de positie van de linkerkant van het kader aan.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op het aankruisvakje staat.
Move( )	Geen	Verplaats het aankruisvakje of wijzigt de grootte van het aankruisvakje.
Name	AANKRUISVAKJE1	Geeft de naam aan van het aankruisvakje.
OldStyle	.F.	Bepaalt of het aankruisvakje wordt weergegeven in de standaardstijl van Windows of in de stijl van Borland.
OnChange	Geen	Voert een subroutine uit als de gebruiker de waarde wijzigt.
OnGotFocus	Geen	Voert een subroutine uit als het aankruisvakje de focus krijgt.
OnHelp	Geen	Voert een subroutine uit als de gebruiker op *F1* drukt.
OnLeftDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het aankruisvakje.
OnLeftMouseDown	Geen	Voert een subroutine uit als de gebruiker klikt op het aankruisvakje.
OnLeftMouseUp	Geen	Voert een subroutine uit als de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op het aankruisvakje staat.
OnLostFocus	Geen	Voert een subroutine uit als de focus het object verlaat.
OnMiddleDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het aankruisvakje met de middelste muisknop.
OnMiddleMouseDown	Geen	Voert een subroutine uit als de gebruiker op het aankruisvakje klikt met de middelste muisknop.
OnMiddleMouseUp	Geen	Voert een subroutine uit als de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op het aankruisvakje staat.

Kenmerk	Standaardinstelling	Beschrijving
OnMouseMove	Geen	Voert een subroutine uit als de gebruiker de aanwijzer over het aankruisvakje beweegt.
OnOpen	Geen	Voert een subroutine uit als het hoofdformulier wordt geopend.
OnRightDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het aankruisvakje met de rechtermuisknop.
OnRightMouseDown	Geen	Voert een subroutine uit als de gebruiker rechtsklikt op het aankruisvakje.
OnRightMouseUp	Geen	Voert een subroutine uit als de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op het aankruisvakje staat.
Parent	Geen	Een objectverwijzing die naar het hoofdformulier wijst.
Release( )	Geen	Verwijdert de definitie van het aankruisvakje uit het geheugen.
SetFocus( )	Geen	Verplaatst de focus naar het aankruisvakje.
StatusMessage	Lege tekenreeks	Geeft een melding aan om te tonen op de statusbalk als het aankruisvakje de focus heeft.
TabStop	.T.	Bepaalt of de gebruiker de focus naar het aankruisvakje kan verplaatsen met *Tab* of *Shift+Tab*.
Text	Aankruisvakje	Geeft een tekenreeks aan om naast het aankruisvakje te tonen.
Top	Geen	Stelt de positie van de bovenkant van het kader in.
Value	Geen	Stelt de waarde voor het aankruisvakje in.
Visible	.T.	Bepaalt of het aankruisvakje zichtbaar of verborgen is.
When	Geen	Geeft een voorwaarde aan die waar moet zijn voordat de gebruiker de focus naar het aankruisvakje kan verplaatsen.
Width	Geen	Stelt de breedte in.

Beschrijving

Een aankruisvakje stelt de gebruiker in staat te schakelen tussen waar en onwaar, ja en nee of aan en uit. Deze waarde kan zijn opgeslagen in een logisch veld dat u opgeeft met het kenmerk DataLink.

Als het aankruisvakje een kruisje (X) bevat, heeft het de waarde waar, ja of aan. Bevat het aankruisvakje geen kruisje, dan heeft het de waarde onwaar, nee of uit.

Als u een aankruisvakje koppelt aan een veld en de gebruiker gaat van veld naar veld, verandert de weergave van het aankruisvakje aan de hand van de waarde in het huidige record.

Als u een aankruisvakje maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing die naar het hoofdformulier wijst.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van het nieuwe aankruisvakje. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een aankruisvakje dat op het formulier wordt weergegeven:

```
MijnForm = NEW FORM()
MijnAv = NEW CHECKBOX(MijnForm, "Aankruisvakje")
```

Het kenmerk Name van het nieuwe aankruisvakje bevat "Aankruisvakje".

Opmerking U kunt een veld opgeven voor het kenmerk DataLink door op de hulpmiddelenknop naast het element DataLink in het kenmerkenvenster te klikken.

Voorbeeld

In het volgende voorbeeld worden aankruisvakjes gebruikt om opties te kiezen voor het commando LIST. GrootVak is een subklasse van Checkbox met een groot font:

```
USE ?
DEFINE FORM LijstForm PROPERTY MDI .F.
DEFINE PUSHBUTTON Lijstmaker OF LijstForm ;
       PROPERTY ;
       Text "Lijst", ;
       Top  10, ;
       Left 15, ;
       OnClick  LijstKlik
NEW GrootVak(LijstForm,"Inclusief recordnummers")
NEW GrootVak(LijstForm,"Afdrukken")
LijstForm.Aankruisvakje1.Top = 2
LijstForm.Aankruisvakje2.Top = 6
READMODAL(LijstForm)

CLASS GrootVak(f,gtTekst) OF CHECKBOX(f)
      this.FontName = "Arial"
      this.FontSize = 16
      this.Text     = SPACE(2) + gtTekst
      this.Width    = LEN(gtTekst) * 3
      this.Left     = 2
      this.Value    = .F.
ENDCLASS

FUNCTION LijstKlik
   IF Form.Aankruisvakje1.Value    && Met recordnummers
      IF Form.Aankruisvakje2.Value && afdrukken
         LIST TO PRINTER
      ELSE
         LIST
      ENDIF
   ELSE                            && Geen recordnummers
      IF Form.Aankruisvakje2.Value && afdrukken
         LIST OFF TO PRINTER
      ELSE
         LIST OFF
      ENDIF
   ENDIF
   FORM.CLOSE()
RETURN 0
```

Zie ook

CLASS RADIOBUTTON, DEFINE

CLASS COMBOBOX

Een object waarmee de gebruiker een waarde kan kiezen uit een lijst door tekens in te voeren in een invoervak of door een waarde te kiezen uit een lijst. Een combobox is een keuzelijst met invoervak.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Combobox. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Before	Geen	Bepaalt welk object voor de keuzelijst met invoervak komt in de tabvolgorde van het hoofdformulier.
ClassName	COMBOBOX	Geeft de klasse van de keuzelijst met invoervak aan.
DataLink	Lege tekenreeks	Koppelt de keuzelijst met invoervak aan een veld.
DataSource	Lege tekenreeks	Geeft aan welke tekenreeks moet worden getoond in de keuzelijst met invoervak.
Enabled	.T.	Bepaalt of de keuzelijst met invoervak kan worden geselecteerd.
FontBold	.T.	Bepaalt of tekens vet worden weergegeven.
FontItalic	.F.	Bepaalt of tekens cursief worden weergegeven.
FontName	MS Sans Serif	Bepaalt het font voor de weergegeven tekens.
FontSize	Geen	Bepaalt de grootte van het weergegeven font in punten.
FontStrikeOut	.F.	Bepaalt of tekens doorgehaald worden weergegeven.
FontUnderline	.F.	Bepaalt of tekens onderstreept worden weergegeven.
Height	Geen	Bepaalt de hoogte.
HelpFile	Lege tekenreeks	Geeft een Windows-Helpbestand (.HLP-bestand) aan dat contextgevoelige Help bevat.
HelpID	Lege tekenreeks	Geeft de contextreeks of het contextnummer aan van een Help-onderwerp in een Windows-Helpbestand (.HLP-bestand).
hWnd	Geen	Geeft de object-handle van het keuzelijst met invoervak-object als resultaat.

Kenmerk	Standaardinstelling	Beschrijving
ID	-1	Geeft de keuzelijst met invoervak aan door middel van een numerieke waarde.
Left	Geen	Geeft de positie van de linkerkant van het kader aan.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op de keuzelijst met invoervak staat.
Move( )	Geen	Verplaatst de keuzelijst met invoervak of wijzigt de grootte van de keuzelijst met invoervak.
Name	KEUZELIJST_MET_INVOERVAK1	Geeft de naam aan van de keuzelijst met invoervak.
OnChange	Geen	Voert een subroutine uit als de gebruiker een ander element uit de lijst kiest.
OnGotFocus	Geen	Voert een subroutine uit als de keuzelijst met invoervak de focus krijgt.
OnHelp	Geen	Voert een subroutine uit als de gebruiker op *F1* drukt.
OnLeftDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op de keuzelijst met invoervak.
OnLeftMouseDown	Geen	Voert een subroutine uit als de gebruiker klikt op de keuzelijst met invoervak.
OnLeftMouseUp	Geen	Voert een subroutine uit als de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op de keuzelijst met invoervak staat.
OnLostFocus	Geen	Voert een subroutine uit als de focus het object verlaat.
OnMiddleDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op de keuzelijst met invoervak met de middelste muisknop.
OnMiddleMouseDown	Geen	Voert een subroutine uit als de gebruiker met de middelste muisknop op de keuzelijst met invoervak klikt.
OnMiddleMouseUp	Geen	Voert een subroutine uit als de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op de keuzelijst met invoervak staat.
OnMouseMove	Geen	Voert een subroutine uit als de gebruiker de muis over de keuzelijst met invoervak beweegt.
OnOpen	Geen	Voert een subroutine uit als het hoofdformulier wordt geopend.
OnRightDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op de keuzelijst met invoervak met de rechtermuisknop.
OnRightMouseDown	Geen	Voert een subroutine uit als de gebruiker met de rechtermuisknop op de keuzelijst met invoervak klikt.

Kenmerk	Standaardinstelling	Beschrijving
OnRightMouseUp	Geen	Voert een subroutine uit als de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op de keuzelijst met invoervak staat.
Parent	Geen	Een objectverwijzing die naar het hoofdformulier wijst.
Release( )	Geen	Verwijdert de definitie van de keuzelijst met invoervak uit het geheugen.
SetFocus( )	Geen	Verplaatst de focus naar de keuzelijst met invoervak.
Sorted	.F.	Bepaalt of de keuzemogelijkheden in de lijst alfabetisch zijn gesorteerd.
StatusMessage	Lege tekenreeks	Geeft een melding aan om te tonen op de statusbalk.
Style	0 (Eenvoudig)	Bepaalt of de gebruiker het invoervak, de vervolgkeuzelijst of beide kan gebruiken.
TabStop	.T.	Bepaalt of de gebruiker de focus naar de keuzelijst met invoervak kan verplaatsen met *Tab* of *Shift+Tab*.
Top	Geen	Stelt de positie van de bovenkant van het kader in.
Value	Lege tekenreeks	Stelt de waarde voor de keuzelijst met invoervak in.
Visible	.T.	Bepaalt of de keuzelijst met invoervak zichtbaar of verborgen is.
When	Geen	Bepaalt een voorwaarde die waar moet zijn voordat de gebruiker de focus naar de keuzelijst met invoervak kan verplaatsen.
Width	Geen	Stelt de breedte in.

Beschrijving

Gebruik een keuzelijst met invoervak om een lijst te tonen met keuzemogelijkheden waaruit de gebruiker snel een mogelijkheid kan kiezen.

De keuzemogelijkheden in een keuzelijst met invoervak kunnen bijvoorbeeld bestaan uit een lijst namen uit een geïndexeerd veld in een tabel. De gebruiker kan het eerste record met een bepaalde naam opzoeken door de eerste letters van de naam in het invoervak te typen. Als de gebruiker de letters typt, komt de naam op een gegeven moment vanzelf boven in het lijstvak te staan. De gebruiker kan ook naar de naam bladeren met behulp van de schuifbalken en die vervolgens selecteren met de muis.

In een keuzelijst met invoervak kunnen vijf soorten keuzemogelijkheden worden weergegeven:

1. Veldnamen.
2. Waarden uit een tabelveld.
3. Veldnamen uit de structuur van een tabel.

4 Elementen in een array-object.

5 De namen van alle tabellen in de huidige, geopende database. (Zie OPEN DATABASE voor meer informatie over databases.)

U geeft de keuzemogelijkheden op met het kenmerk DataSource.

De grootte van de keuzelijst met invoervak bepaalt u met de clausule FROM...TO van het commando DEFINE of met de kenmerken Height en Width. Als de opgegeven hoogte onvoldoende is voor alle keuzemogelijkheden, kan de gebruiker door de keuzemogelijkheden bladeren. Als de hoogte groter is dan noodzakelijk, wordt die automatisch teruggebracht.

Als u een keuzelijst met invoervak maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing die naar het hoofdformulier wijst.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van de nieuwe keuzelijst met invoervak. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een keuzelijst met invoervak die op het formulier wordt weergegeven:

```
MijnForm = NEW FORM()
MijnKl = NEW COMBOBOX(MijnForm, "Keuzelijst")
```

Het kenmerk Name van de nieuwe keuzelijst met invoervak bevat "Keuzelijst".

Opmerking U kunt een waarde opgeven voor DataSource met het dialoogvenster **Gegevensbron kiezen** in het formulierontwerpvenster. Klik op de hulpmiddelenknop naast het element DataSource in het kenmerkenvenster om toegang te krijgen tot het dialoogvenster **Gegevensbron kiezen**.

In het dialoogvenster **Veld kiezen** kunt u een veld opgeven voor het kenmerk DataLink. Klik op de hulpmiddelenknop naast het element DataLink in het kenmerkenvenster om toegang te krijgen tot het dialoogvenster **Veld kiezen**.

Voorbeeld

In het volgende voorbeeld wordt een formulier gedefinieerd dat een keuzelijst met invoervak bevat waarin namen uit de tabel Dieren.DBF worden getoond. Als u dubbelklikt op een van deze keuzemogelijkheden, wordt het formulier gesloten:

```
LOCAL f
f=NEW ToonKlmI()
f.OPEN()
CLASS ToonKlmI OF FORM
  this.View = "DIEREN.DBF"
  DEFINE COMBOBOX Keuzelijst_Met_Invoervak1 OF THIS;
    PROPERTY;
      DataSource "FIELD Dieren->Naam",;
      FontBold .T.,;
      Top 4,;
      Left 6,;
      Width 20, ;
      OnLeftDblClick (; ? 'Uw keuze = ',;
```

```
      THIS.value, ; FORM.CLOSE()}
  DEFINE TEXT Tekst1 OF THIS;
    PROPERTY;
      Text "Kies uw favoriete dier",;
      FontBold .T., Width 40,;
      Top 1, Left 3
ENDCLASS
```

Zie ook

CLASS LISTBOX, DEFINE

CLASS DDELINK

Initieert een DDE-koppeling tussen dBASE en een server-toepassing. Via deze koppeling kan dBASE opdrachten en verzoeken om gegevensuitwisseling doorgeven aan de server. Een DDELink is een DDE-koppeling.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse DDELink. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Advise( )	Geen	Geeft aan dat de server de client op de hoogte moet stellen als een element in het server-document is gewijzigd.
ClassName	DDELINK	Geeft de klasse van de DDE-koppeling aan.
Execute( )	Geen	Geeft opdrachten door aan de server in diens eigen taal.
Initiate( )	Geen	Start een conversatie met een DDE-server-toepassing.
OnNewValue	Geen	Voert een subroutine uit als een element in de server-toepassing verandert.
Peek( )	Geen	Haalt een gegevenselement op uit een server-document.
Poke( )	Geen	Voegt een gegevenselement in in een server-document.
Reconnect( )	Geen	Herstelt een DDE-koppeling die is beëindigd met Terminate( ).
Release( )	Geen	Verwijdert de definitie van een DDELink-object uit het geheugen.
Server	Geen	Bevat de naam van de server die u opgeeft met de methode Initiate( ).
Terminate( )	Geen	Beëindigt de koppeling met de server-toepassing.
TimeOut	10.00	Bepaalt de hoe lang dBASE wacht tot een transactie wordt uitgevoerd voordat een fout als resultaat wordt gegeven.
Topic	Lege tekenreeks	Bevat de naam van het server-object dat u hebt opgegeven met de methode Initiate( ).
Unadvise( )	Geen	Geeft de server opdracht niet langer aan het DDELink-object door te geven dat een element in het server-document is gewijzigd.

Beschrijving

Met een DDELink-object kunt u een communicatiekanaal (een zogenaamde *DDE-koppeling*) openen tussen dBASE en een externe Windows-toepassing (die de *server* wordt genoemd). Via deze koppeling kunt u gegevens en opdrachten uitwisselen, zodat beide toepassingen kunnen samenwerken. U kunt bijvoorbeeld via een DDElink-object een koppeling tot stand brengen met Quattro Pro voor Windows, een werkboekbestand openen en de gegevens uit het werkboek naar een dBASE-tabel kopiëren.

Een DDE-koppeling wordt tot stand gebracht met het kenmerk Initiate( ). Als de server nog niet is gestart, zal Initiate( ) eerst trachten de server te starten voordat de koppeling tot stand wordt gebracht. Als dat niet lukt, geeft Initiate( ) de waarde onwaar als resultaat.

Zie Hoofdstuk 26 in *Programmeren* voor meer informatie over DDE. Zie CLASS DDETOPIC voor meer informatie over het gebruik van dBASE als server-toepassing.

Voorbeeld

In het volgende voorbeeld wordt een DDELINK-object gemaakt met de operator NEW. Vervolgens wordt geprobeerd met de methode Initiate een server-sessie met Quattro Pro voor Windows tot stand te brengen. Daarna wordt de methode Peek gebruikt om een waarde uit cel Gasprijs:B5 naar een dBASE-geheugenvariabele te kopiëren en de methode Poke om een nieuwe waarde in cel Gasprijs:B6 van het QPW-werkblad te plaatsen:

```
LOCAL KoppObj
KoppObj = NEW DDELINK()
IF KoppObj.Initiate("QPW","Les.WB1")
  ? "Koppeling met QPW tot stand gebracht"
ELSE
  ? "Niet gelukt koppeling met QPW tot stand te brengen"
ENDIF
mWaarde1=KoppObj.Peek("Gasprijs:B5")
? mWaarde1
KoppObj.Poke("Gasprijs:B6","198")
mWaarde2=KoppObj.Peek("Gasprijs:B6")
? mWaarde2
KoppObj.RELEASE()
```

Zie ook

CLASS DDETOPIC, CLASS OLE, DEFINE

CLASS DDETOPIC

Bepaalt welke handelingen moeten worden uitgevoerd als dBASE voor Windows verzoeken ontvangt van een DDE-client.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse DDETopic. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
ClassName	DDETOPIC	Geeft de klasse van de DDETopic aan.
Notify( )	Geen	Stelt alle client-toepassingen op de hoogte als een dBASE-element is gewijzigd.
OnAdvise	Geen	Voert een subroutine uit als een externe toepassing een dynamische koppeling maakt.
OnExecute	Geen	Voert een subroutine uit als client-toepassing een instructie doorgeeft aan dBASE.
OnPeek	Geen	Voert een subroutine uit als de client om een waarde uit dBASE verzoekt.
OnPoke	Geen	Voert een subroutine uit als de client een nieuwe waarde invoegt in een dBASE-element.
OnUnadvise	Geen	Voert een subroutine uit als een client een dynamische koppeling met een bepaald element verwijdert.
Release( )	Geen	Verwijdert de definitie van het DDETopic-object uit het geheugen.
Topic	Geen	Het onderwerp van het DDETopic-object.

Beschrijving

Met een DDETopic-object kunt u bepalen wat dBASE doet voor een client-toepassing als dBASE de server in een DDE-koppeling is.

Als server-toepassing accepteert dBASE voor Windows instructies van een client-toepassing, kunnen gegevens van en naar de server-toepassing worden doorgegeven en worden de handelingen uitgevoerd die u opgeeft met de kenmerken van het DDETopic-object. Een OnPoke-subroutine kan bijvoorbeeld een veldnaam en een veldwaarde ontvangen van een client-toepassing, de waarde invoegen in het opgegeven veld en vervolgens de handeling bevestigen met de methode Notify( ).

Een DDETopic-object wordt meestal gemaakt in een initiatieroutine, die wordt toegewezen aan het kenmerk OnInitiate van _app. Een initiatieroutine wordt uitgevoerd op het moment dat een client-toepassing om een DDE-koppeling met dBASE verzoekt. Zie Hoofdstuk 26 in *Programmeren* voor meer informatie over initiatieroutines.

Zie CLASS DDELINK voor meer informatie over dBASE als client-toepassing.

Als u een DDETopic-object maakt met de operator NEW, geeft u de parameter *<onderwerp>* op. Deze waarde wordt automatisch in het kenmerk Topic van het nieuwe DDETopic-object geplaatst. Externe toepassingen gebruiken dit kenmerk om een object aan te duiden en een DDE-koppeling tot stand te brengen. De volgende instructie maakt bijvoorbeeld een DDETopic-object en plaatst "Onderwerp" in het kenmerk Topic:

```
OnsOndrw = NEW DDETopic("Onderwerp")
```

Een Quattro Pro-werkblad kan de volgende instructie uitvoeren om het nieuwe object aan te roepen:

```
{INITIATE "DBASEWIN", "Onderwerp"}
```

Voorbeeld

In het volgende voorbeeld wordt een DDE-server voor het verwerken van beursinformatie gemaakt. Om het eenvoudig te houden, wordt maar één dynamische koppeling tegelijk ondersteund en hebben alle aandelen dezelfde waarde. Als een klant een aandeel koopt, gaat de prijs omhoog en als een klant een aandeel verkoopt, gaat de prijs omlaag.

```
SET PROCEDURE TO PROGRAM(1) ADDITIVE
PUBLIC Aandeel, Koppeling, Waarde
Waarde = 100.0
_app.DDEServiceName = "Aandeel"
_app.OnInitiate = INITROUTINE

FUNCTION InitRoutine
PARAMETER Topic
IF ".STK" $ Topic
  x = NEW AandeelOnd()
  ELSE
  x = .F.
ENDIF
RETURN x

CLASS AandeelOnd OF DDETOPIC
Aandeel.OnAdvise = KoppAanRoutine
Aandeel.OnExecute = UitvRoutine
Aandeel.OnPeek = PeekRoutine
Aandeel.OnPoke = PokeRoutine
Aandeel.OnUnadvise = KoppUitRoutine

FUNCTION KoppAanRoutine
PARAMETER Item
Koppeling = Item
RETURN .T.

FUNCTION UitvRoutine
PARAMETER Cmd
IF Cmd = "SELL"
  Waarde = Waarde - 10.0
ELSE
  IF Cmd = "BUY"
    Waarde = Waarde + 10.0
  ENDIF
ENDIF
THIS.Notify(Koppeling)
RETURN .T.

FUNCTION PeekRoutine
PARAMETER Item
RETURN Waarde

FUNCTION PokeRoutine
PARAM Item, Val
```

```
? "POKE: ", Item, Val
RETURN .T.

FUNCTION KoppUitRoutine
PARAMETER Item
IF(Koppeling = item)
  Koppeling = .F.
ENDIF
RETURN .T.

ENDCLASS
```

Zie ook

CLASS DDELINK

CLASS EDITOR

Een hulpmiddel waarmee de gebruiker een tekstbestand of memoveld kan bekijken en wijzigen.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Editor. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Before	Geen	Bepaalt welk object voor het editor-object komt in de tabvolgorde van het hoofdformulier.
ClassName	EDITOR	Geeft de klasse van het editor-object aan.
ColorNormal	N/W	Bepaalt de kleur van het editor-object als het niet de focus heeft.
DataLink	Lege tekenreeks	Koppelt het editor-object aan een tekstbestand, een memoveld of een tekenveld.
Enabled	.T.	Bepaalt of het editor-object kan worden geselecteerd.
FontBold	.T.	Bepaalt of tekens vet worden weergegeven.
FontItalic	.F.	Bepaalt of tekens cursief worden weergegeven.
FontName	MS Sans Serif	Bepaalt het lettertype van de weergegeven tekens.
FontSize	Geen	Bepaalt de grootte van het font in punten.
FontStrikeOut	.F.	Bepaalt of tekens doorgehaald worden weergegeven.
FontUnderline	.F.	Bepaalt of tekens onderstreept worden weergegeven.
Height	Geen	Bepaalt de hoogte.
HelpFile	Lege tekenreeks	Geeft een Windows-Helpbestand (.HLP-bestand) aan dat contextgevoelige Help bevat.
HelpID	Lege tekenreeks	Geeft de contextreeks of het contextnummer aan van een Help-onderwerp in een Windows-Helpbestand (.HLP-bestand).

Kenmerk	Standaardinstelling	Beschrijving
hWnd	Geen	Geeft de object-handle van het editor-object als resultaat.
ID	-1	Geeft het editor-object aan door middel van een numerieke waarde.
Left	Geen	Geeft de positie van de linkerkant van het kader aan.
Modify	.T.	Bepaalt of de gebruiker in het editor-object gegevens kan wijzigen.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op het editor-object staat.
Move( )	Geen	Verplaatst het editor-object of wijzigt de grootte van het editor-object.
Name	EDITOR1	Geeft de naam aan van het editor-object.
OnChange	Geen	Voert een subroutine uit als de gebruiker tekst in het editor-object wijzigt.
OnGotFocus	Geen	Voert een subroutine uit als het editor-object de focus krijgt.
OnHelp	Geen	Voert een subroutine uit als de gebruiker op *F1* drukt.
OnLeftDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het editor-object.
OnLeftMouseDown	Geen	Voert een subroutine uit als de gebruiker klikt op het editor-object.
OnLeftMouseUp	Geen	Voert een subroutine uit als de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op het editor-object staat.
OnLostFocus	Geen	Voert een subroutine uit als de focus het object verlaat.
OnMiddleDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het editor-object met de middelste muisknop.
OnMiddleMouseDown	Geen	Voert een subroutine uit als de gebruiker met de middelste muisknop op het editor-object klikt.
OnMiddleMouseUp	Geen	Voert een subroutine uit als de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op het editor-object staat.
OnMouseMove	Geen	Voert een subroutine uit als de gebruiker de muis over het editor-object beweegt.
OnOpen	Geen	Voert een subroutine uit als het hoofdformulier wordt geopend.
OnRightDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het editor-object met de rechtermuisknop.
OnRightMouseDown	Geen	Voert een subroutine uit als de gebruiker rechtsklikt op het editor-object.
OnRightMouseUp	Geen	Voert een subroutine uit als de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op het editor-object staat.
Parent	Geen	Een objectverwijzing die naar het hoofdformulier wijst.
Release( )	Geen	Verwijdert de definitie van het editor-object uit het geheugen.
Scrollbar	1 (Aan)	Bepaalt of het editor-object over een schuifbalk beschikt.
SetFocus( )	Geen	Verplaatst de focus naar het editor-object.

Kenmerk	Standaardinstelling	Beschrijving
StatusMessage	Lege tekenreeks	Geeft een melding aan om te tonen op de statusbalk als het editor-object de focus krijgt.
TabStop	.T.	Bepaalt of de gebruiker de focus naar het editor-object kan verplaatsen met *Tab* of *Shift+Tab*.
Top	Geen	Stelt de positie van de bovenkant van het kader in.
Valid	Geen	Geeft een voorwaarde aan die waar (.T.) moet zijn voordat de gebruiker de focus van het editor-object verplaatsen.
Value	Lege tekenreeks	Bepaalt de inhoud van het editor-object.
Visible	.T.	Bepaalt of het editor-object zichtbaar of verborgen is.
When	Geen	Geeft een voorwaarde aan die waar moet zijn voordat de gebruiker de focus naar het editor-object kan verplaatsen.
Width	Geen	Stelt de breedte in.
Wrap	.T.	Bepaalt of tekst in het editor-object automatisch wordt voorzien van regelovergangen.

Beschrijving

Met een editor-object kunt u een formulier voorzien van mogelijkheden voor het wijzigen van tekst, zodat gebruikers de inhoud van tekstbestanden, memovelden en tekenvelden kunnen bekijken en wijzigen.

Met het kenmerk DataLink geeft u aan tot welk tekstbestand of memoveld toegang wordt verkregen. Om toegang te krijgen tot een tekstbestand moet u het sleutelwoord FILE gebruiken, zoals:

"FILE MIJNTXT.TXT"

Gebruik de veldnaam en de alias van de tabel die het veld bevat om toegang te krijgen tot een memoveld of een tekenveld.

De afmetingen van het editor-object stelt in met de clausule FROM...TO van het commando DEFINE of met de kenmerken Height en Width van het object. Schuifbalken stellen de gebruiker in staat horizontaal of verticaal binnen de weergave te bladeren als de invoer de afmetingen van het editor-object overschrijdt.

Met de kenmerken Top en Left plaatst u het editor-object op het formulier.

Als u een editor-object maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing die naar het hoofdformulier wijst.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van het nieuwe editor-object. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een editor-object om in het formulier weer te geven:

```
MijnForm = NEW FORM()
MijnEditor = NEW EDITOR(MijnForm, "Editor-object")
```

Het kenmerk Name van het nieuwe editor-object bevat "Editor-object".

Voorbeeld

In het volgende voorbeeld wordt een formulier gedefinieerd dat een bladerobject bevat met informatie uit de velden CompCode en Contact uit de tabel Contact en dat de inhoud van het bijbehorende memoveld toont in een editor-object rechts van het bladerobject. De recordaanwijzer wordt verplaatst met de opdrachtknoppen Terug en Verder en met de knop Sluiten kan de gebruiker het formulier sluiten:

```
LOCAL f
f=NEW InvoerForm()
f.OPEN()
CLASS InvoerForm OF FORM
  this.Top=2
  this.Left=2
  this.Width=72
  this.Height=20
  this.View = "Contact.DBF"
  this.Text= "Wijzigen naar behoefte"
  DEFINE BROWSE BO1 OF THIS;
    PROPERTY;
      Top 4,;
      Left 3,;
      Width 32,;
      Height 12
  DEFINE TEXT Text1 OF THIS;
    PROPERTY;
      Text "Contactpersonen uit tabel Contact",;
      Width 72,;
      Top 1,;
      Left 0,;
      Alignment 1,;
      Height 2.50,;
      FontBold .T.,;
      FontSize 14.00,;
      ColorNormal "R/W"
  DEFINE EDITOR ED1 OF THIS;
    PROPERTY;
      Top 4,;
      Left 37,;
      Width 32,;
      Height 12,;
      DataSource "MEMO Contact->Notities"
  DEFINE PUSHBUTTON Back OF THIS;
    PROPERTY Text "Terug", Height 2,;
      Top 17, Left 22,;
      OnClick {;SKIP-1}, FontBold .T.
  DEFINE PUSHBUTTON Next OF THIS;
    PROPERTY TEXT "Verder", Height 2,;
      Top 17, Left 32,;
      OnClick {;SKIP}, FontBold .T.
  DEFINE PUSHBUTTON Exit OF THIS;
    PROPERTY Text "Sluiten", Height 2,;
      Top 17, Left 42,;
      OnClick {;Form.Close()}, FontBold .T.
ENDCLASS
```

Zie ook

MODIFY COMMAND

CLASS ENTRYFIELD

Een invoervak is een gebied waar de gebruiker een enkele waarde kan invoeren of wijzigen.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Entryfield. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Before	Geen	Bepaalt welk object voor het invoervak komt in de tabvolgorde van het hoofdformulier.
Border	.T.	Bepaalt of het invoervak wordt omgeven door een kader.
ClassName	ENTRYFIELD	Geeft de klasse van het invoervak aan.
ColorHighLight	N/W	Bepaalt de kleur van het invoervak als het is geselecteerd.
ColorNormal	N/W	Bepaalt de kleur van het invoervak als het niet is geselecteerd.
DataLink	Lege tekenreeks	Koppelt het invoervak aan een tabelveld.
Enabled	.T.	Bepaalt of het invoervak kan worden geselecteerd.
FontBold	.T.	Bepaalt of tekens in het invoervak vet worden weergegeven.
FontItalic	.F.	Bepaalt of tekens cursief worden weergegeven.
FontName	MS Sans Serif	Bepaalt het font voor de weergegeven tekens.
FontSize	Geen	Bepaalt de grootte van het weergegeven font in punten.
FontStrikeOut	.F.	Bepaalt of tekens doorgehaald worden weergegeven.
FontUnderline	.F.	Bepaalt of tekens onderstreept worden weergegeven.
Function	Lege tekenreeks	Bepaalt opmaak voor weergegeven tekst.
Height	Geen	Bepaalt de hoogte.
HelpFile	Lege tekenreeks	Geeft een Windows-Helpbestand (.HLP-bestand) aan dat contextgevoelige Help bevat.
HelpID	Lege tekenreeks	Geeft de contextreeks of het contextnummer aan van een Help-onderwerp in een Windows-Helpbestand (.HLP-bestand).
hWnd	Geen	Geeft de object-handle van het invoervak-object als resultaat.
ID	-1	Geeft het invoervak aan door middel van een numerieke waarde.
Key	Geen	Voert een subroutine uit als de gebruiker op een toets drukt.
Left	Geen	Geeft de positie van de linkerkant van het kader aan.

Kenmerk	Standaardinstelling	Beschrijving
MaxLength	Geen	Geeft de bladerbreedte aan.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op invoervak staat.
Move( )	Geen	Verplaatst het invoervak of wijzigt de grootte van het invoervak.
Name	INVOERVAK11	Geeft de naam aan van het invoervak.
OldStyle	Geen	Bepaalt of het invoervak wordt weergegeven in de standaardstijl van Windows of in de stijl van Borland.
OnChange	Geen	Voert een subroutine uit als de gebruiker een waarde wijzigt.
OnGotFocus	Geen	Voert een subroutine uit als het invoervak de focus krijgt.
OnHelp	Geen	Voert een subroutine uit als de gebruiker op *F1* drukt.
OnLeftDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het invoervak.
OnLeftMouseDown	Geen	Voert een subroutine uit als de gebruiker klikt op het invoervak.
OnLeftMouseUp	Geen	Voert een subroutine uit als de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op het invoervak staat.
OnLostFocus	Geen	Voert een subroutine uit als de focus het object verlaat.
OnMiddleDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het invoervak met de middelste muisknop.
OnMiddleMouseDown	Geen	Voert een subroutine uit als de gebruiker op het invoervak klikt met de middelste muisknop.
OnMiddleMouseUp	Geen	Voert een subroutine uit als de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op het invoervak staat.
OnMouseMove	Geen	Voert een subroutine uit als de gebruiker de aanwijzer over het invoervak beweegt.
OnOpen	Geen	Voert een subroutine uit als het hoofdformulier wordt geopend.
OnRightDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het invoervak met de rechtermuisknop.
OnRightMouseDown	Geen	Voert een subroutine uit als de gebruiker rechtsklikt op het invoervak.
OnRightMouseUp	Geen	Voert een subroutine uit als de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op het invoervak staat.
Parent	Geen	Een objectverwijzing die naar het hoofdformulier wijst.
Picture	Lege tekenreeks	Bepaalt de opmaak van tekst.
Release( )	Geen	Verwijdert de definitie van het invoervak uit het geheugen.
SelectAll	.F.	Bepaalt of de startwaarde geselecteerd in het invoervak verschijnt.
SetFocus( )	Geen	Verplaatst de focus naar het invoervak.
StatusMessage	Lege tekenreeks	Geeft een melding aan om te tonen op de statusbalk als het invoervak de focus heeft.

Kenmerk	Standaardinstelling	Beschrijving
TabStop	.T.	Bepaalt of de gebruiker de focus naar het invoervak kan verplaatsen met *Tab* of *Shift+Tab*.
Top	Geen	Stelt de positie van de bovenkant van het kader in.
Valid	Geen	Geeft een voorwaarde aan die waar (.T.) moet zijn voordat de gebruiker de focus van het invoervak kan verplaatsen.
ValidErrorMsg	Lege tekenreeks	Geeft een tekenreeks aan die op de statusbalk moet worden getoond als het kenmerk Valid .F. (Geen) als resultaat geeft.
ValidRequired	.F.	Bepaalt of het kenmerk Valid voor alle gegevens of alleen voor nieuwe gegevens geldt.
Value	Lege tekenreeks	Stelt de waarde in het invoervak in.
Visible	.T.	Bepaalt of het invoervak zichtbaar of verborgen is.
When	Geen	Geeft een voorwaarde aan die waar moet zijn voordat de gebruiker de focus naar het invoervak kan verplaatsen.
Width	Geen	Stelt de breedte in.

Beschrijving

Gebruik een invoervak om gebruikers gegevens te laten invoeren en wijzigen in een enkel veld. Op een formulier dat toegang geeft tot slechts één tabelveld kan bijvoorbeeld een invoervak worden gebruikt in plaats van een bladerobject.

Een invoervak lijkt uiterlijk op een invoergebied dat is gemaakt met het dBASE IV-commando @...GET. Voor het invoeren van gegevens in een invoervak is echter geen READ-commando nodig.

Met het kenmerk DataLink kunt u opgeven welk veld moet worden gewijzigd. U kunt een veld voor het kenmerk DataLink opgeven in het formulierontwerpvenster, door op de hulpmiddelenknop naast het element DataLink in het kenmerkenvenster te klikken.

De tekst bij een invoervak maakt u met een tekstobject. Zo kan naast een invoervak voor het opgeven van een wachtwoord bijvoorbeeld een tekstobject worden geplaatst met de tekst "Geef wachtwoord". Zie CLASS TEXT voor meer informatie over tekstobjecten.

Als u een invoervak maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing die naar het hoofdformulier wijst.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van het nieuwe invoervak. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een invoervak om op het formulier weer te geven:

```
MijnForm = NEW FORM()
MijnVeld = NEW ENTRYFIELD(MijnForm, "Invoervak")
```

Het kenmerk Name van het nieuwe invoervak bevat "Invoervak".

Voorbeeld

In het volgende voorbeeld wordt een formulier met drie invoervakken gedefinieerd. Elk ENTRYFIELD maakt gebruik van een andere objectgeoriënteerde syntaxis voor de definitie. De gebruiker kan de recordaanwijzer verplaatsen met de opdrachtknoppen Terug en Verder:

```
LOCAL F1
F1=NEW InvoerForm()
F1.OPEN()
CLASS InvoerForm OF FORM
  this.View="Bedrijf.DBF"
  this.Top=2
  this.Left=2
  this.Width=42
  this.Height=13
  this.Text= "Wijzigen naar behoefte"
* Syntaxis met operator NEW:
    BedrCode=NEW ENTRYFIELD(this)
    BedrCode.DataLink="Bedrijf->BedrCode"
    BedrCode.Top=5
    BedrCode.Left=2
    BedrCode.Width=7
    BedrCode.Height=1.5
* Gecombineerde syntaxis:
    DEFINE ENTRYFIELD Type OF THIS
      this.Type.Width=5
      this.Type.Top=5
      this.Type.Left=9
      this.Type.Height=1.5
      this.Type.DataLink="Bedrijf->Type"
* Syntaxis met DEFINE object:
      DEFINE ENTRYFIELD Bedrijf OF THIS;
        PROPERTY;
          Width 20,;
          Top 5,;
          Left 16,;
          Height 1.5,;
          DataLink "Bedrijf->Bedrijf"
  DEFINE TEXT Text1 OF THIS;
    PROPERTY;
      Text "Bedrijfsinformatie",;
      Width 34,;
      Top 1,;
      Left 2,;
      Alignment 7,;
      Height 2.50,;
      FontSize 12.00,;
      Border .T.
  DEFINE PUSHBUTTON Terug OF THIS;
    PROPERTY Text "Terug", Height 2,;
      Top 10, Left 10,;
      OnClick {;SKIP-1}
  DEFINE PUSHBUTTON Verder OF THIS;
    PROPERTY TEXT "Verder", Height 2,;
      Top 10, Left 20,;
```

```
        OnClick {;SKIP}
ENDCLASS
```

Zie ook

@...SAY...GET, DEFINE

CLASS FORM

Een formulier is een aangepast venster met objecten voor invoer en uitvoer.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Form. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
ActiveControl	Geen	Bevat een verwijzing naar het object dat de focus heeft.
AutoSize	.T.	Bepaalt of het formulier automatisch wordt aangepast aan zijn objecten als het formulier wordt geopend.
ClassName	FORM	Geeft de klasse van het formulier aan.
Close( )	Geen	Sluit het formulier.
ColorNormal	N/W	Bepaalt de kleur van het formulier.
Enabled	.T.	Bepaalt of het formulier de focus kan krijgen.
EscExit	.T.	Bepaalt of de gebruiker het formulier kan sluiten door op Esc te drukken.
First	Geen	Bevat een objectverwijzing naar het eerste object op het formulier.
Height	15.00	Bepaalt de hoogte.
HelpFile	Lege tekenreeks	Geeft een Windows-Helpbestand (.HLP-bestand) aan dat contextgevoelige Help bevat.
HelpID	Lege tekenreeks	Geeft de contextreeks of het contextnummer aan van een Help-onderwerp in een Windows-Helpbestand (.HLP-bestand).
hWnd	Geen	Geeft de object-handle van het formulier-object als resultaat.
Left	2.00	Geeft de positie van de linkerkant van het kader aan.
Maximize	.F.	Bepaalt of het formulier kan worden vergroot tot maximumvenster.
MDI	.T.	Bepaalt of het formulier zich houdt aan de MDI-standaard van Windows (MDI is een afkorting van Multiple Document Interface).
MenuFile	Lege tekenreeks	Wijst een vooringesteld menusysteem toe aan het formulier.
Minimize	.F.	Bepaalt of het formulier kan worden verkleind tot een pictogram.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op het formulier staat.

Kenmerk	Standaardinstelling	Beschrijving
Moveable	.T.	Bepaalt of het formulier kan worden verplaatst met de muis.
NextCol( )	Geen	De volgende kolompositie.
NextObj	Geen	Geeft een verwijzing naar het volgende object in de tabvolgorde van het formulier als resultaat.
NextRow( )	Geen	De volgende rijpositie.
OnChange	Geen	Voert een subroutine uit als een waarde in een object wordt gewijzigd.
OnClose	Geen	Voert een subroutine uit als het formulier wordt gesloten.
OnGotFocus	Geen	Voert een subroutine uit als het formulier de focus krijgt.
OnHelp	Geen	Voert een subroutine uit als de gebruiker op *F1* drukt.
OnLeftDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het formulier.
OnLeftMouseDown	Geen	Voert een subroutine uit als de gebruiker klikt op het formulier.
OnLeftMouseUp	Geen	Voert een subroutine uit als de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op het formulier staat.
OnLostFocus	Geen	Voert een subroutine uit als de focus het object verlaat.
OnMiddleDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het formulier met de middelste muisknop.
OnMiddleMouseDown	Geen	Voert een subroutine uit als de gebruiker op het formulier klikt met de middelste muisknop.
OnMiddleMouseUp	Geen	Voert een subroutine uit als de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op het formulier staat.
OnMouseMove	Geen	Voert een subroutine uit als de gebruiker de muis over het formulier beweegt.
OnMove	Geen	Voert een subroutine uit nadat de gebruiker het formulier heeft verplaatst.
OnNavigate	Geen	Voert een subroutine uit als de gebruiker naar een ander record gaat.
OnOpen	Geen	Voert een subroutine uit als het formulier wordt geopend.
OnRightDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het formulier met de rechtermuisknop.
OnRightMouseDown	Geen	Voert een subroutine uit als de gebruiker rechtsklikt op het formulier.
OnRightMouseUp	Geen	Voert een subroutine uit als de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op het formulier staat.
OnSelection	Geen	Voert een subroutine uit als de gebruiker het formulier voorlegt.
OnSize	Geen	Voert een subroutine uit nadat de gebruiker de grootte van het formulier heeft gewijzigd.
Open( )	Geen	Opent het formulier als een niet-modaal venster.
Print( )	Geen	Drukt het formulier en zijn objecten af.
ReadModal( )	Geen	Opent het formulier als een modaal venster.

Kenmerk	Standaardinstelling	Beschrijving
Release( )	Geen	Verwijdert de definitie van het formulier uit het geheugen.
ScaleFontName	MS Sans Serif	Bepaalt op welk font het coördinatenvlak van het formulier is gebaseerd.
ScaleFontSize	8.00	Bepaalt de hoogte van elke rij en de breedte van elk kolom in het coördinatenvlak van het formulier.
Scrollbar	0 (Uit)	Bepaalt of het formulier over een schuifbalk beschikt.
SetFocus( )	Geen	Verplaatst de focus naar het formulier.
Sizeable	.F.	Bepaalt of de gebruiker de grootte van het formulier kan wijzigen.
StatusMessage	Lege tekenreeks	Geeft een melding aan om te tonen op de statusbalk zolang het formulier de focus heeft.
SysMenu	.F.	Bepaalt of het formulier over een Systeemmenu beschikt.
Text	Formulier	Geeft een tekenreeks aan om te tonen op de titelbalk.
Top	2	Stelt de positie van de bovenkant van het kader in.
View	Lege tekenreeks	Geeft de query of tabel aan waarop het formulier is gebaseerd.
Visible	.T.	Bepaalt of het formulier zichtbaar of verborgen is.
Width	Geen	Stelt de breedte in.
WindowState	0 (Normaal)	Bepaalt of het formulier als pictogram, als maximumvenster of normaal wordt getoond.

Beschrijving

Een formulier is een venster dat u zelf ontwerpt. Het kan standaard interface-elementen (of *stuurelementen*) van Windows bevatten waarmee de gebruiker toegang krijgt tot gegevens en die kan invoeren of wijzigen. Een formulier kan bijvoorbeeld de volgende objecten bevatten: een bladerobject (voor het wijzigen van records), invoervakken (voor de invoer van individuele waarden) en aankruisvakjes (om logische velden mee in te stellen, waar of onwaar). Een formulier is een *bevattend object* voor de objecten die er op worden weergegeven. Als u de definitie van een formulier uit het geheugen verwijderd, heeft dat dus tevens tot gevolg dat de definities van de objecten die het formulier bevat, worden vrijgegeven.

De meeste formulier zijn gebaseerd op een query (QBE), die u opgeeft met het kenmerk View. Als u een formulier bijvoorbeeld wilt baseren op een query met de naam CONTACT.QBE, stelt u "CONTACT.QBE" in voor het kenmerk View. Vervolgens gebruikt u de kenmerken DataLink en DataSource om objecten in het formulier te koppelen aan velden in de query-tabellen.

U kunt twee soorten formulieren maken: *modale* en *niet-modale*. In de Windows-omgeving is een modaal formuliervenster als een dBASE IV-venster. Een modaal formulier onderbreekt de routine die het formulier heeft geopend tot het weer wordt gesloten. Als het actief is, neemt het de volledige gebruikersinterface over. De gebruiker kan niet naar een ander venster gaan zonder het formulier te sluiten. Een dialoogvenster is een voorbeeld van een modaal formulier. Als het wordt geopend, wordt de uitvoering van het programma onderbroken en kan de focus niet worden verplaatst naar een ander venster tot de gebruiker het dialoogvenster weer sluit.

Een niet-modaal formuliervenster stelt de gebruiker daarentegen in staat over te schakelen naar andere vensters in de applicatie. De meeste formulieren die u maakt voor een Windows-applicatie zijn niet-modaal. Een niet-modaal formuliervenster gedraagt zich volgens het MDI-protocol. Dit protocol staat meerdere geopende documentvensters binnen een applicatievenster toe.

Een niet-modaal formulier maakt en gebruikt u door het kenmerk MDI waar (.T.) te maken en het formulier te openen met de methode Open( ) of met het commando OPEN FORM. Als u een modaal formulier wilt maken en gebruiken, moet u MDI onwaar (.F.) maken en het formulier openen met de methode ReadModal( ) method of de functie READMODAL( ).

U kunt ook formuliervensters maken die er uitzien als applicatievensters. U doet dat door het kenmerk MDI onwaar te maken en SHELL(.F.) te gebruiken. SHELL(.F.) verbergt de standaardomgeving van dBASE voor Windows en zorgt dat het formulier de volledige gebruikersinterface overneemt. Het toepassingsvenster van dBASE verdwijnt en in het Taakoverzicht van Windows verschijnt de formuliernaam in plaats van "dBASE voor Windows".

Als u een formulier maakt met de operator NEW, kunt u met de parameter *<tekst>* een waarde opgeven voor het kenmerk Text. De volgende instructie maakt bijvoorbeeld een formulier:

```
MijnForm = NEW FORM("Dit is mijn formulier")
```

Als dit formulier wordt geopend, verschijnt op de titelbalk "Dit is mijn formulier"

Voorbeeld

In het volgende voorbeeld worden DEFINE FORM, NEW FORM( ) en een subklasse van Form gebruikt om overeenkomstige formulieren te maken:

```
DEFINE FORM Winter;
  PROPERTY;
  Text "Sneeuw",;
  Top 5,;
  Left 5
OPEN FORM Winter

Zomer = NEW FORM()
Zomer.Text = "Zon"
Zomer.Top  = 10
Zomer.Left = 10
Zomer.OPEN()

Herfst = NEW Storm()
Herfst.OPEN()
CLASS Storm OF FORM
      this.Text = "Storm"
      this.Top  = 15
      this.Left = 15
ENDCLASS
```

Zie ook

DEFINE, MDI, Open( ), OPEN FORM, ReadModal( ), READMODAL( ), SHELL( )

CLASS IMAGE

Een afbeeldingsobject is een gebied waar bitmap-afbeeldingen worden weergegeven.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Image. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Alignment	0 (Linksboven)	Plaatst een grafische afbeelding in het afbeeldingsobject.
ClassName	IMAGE	Geeft de klasse van het afbeeldingsobject aan.
DataSource	Lege tekenreeks	Geeft de bron van de grafische afbeelding aan: resource-DLL, bestand of binair veld.
Height	Geen	Bepaalt de hoogte.
hWnd	Geen	Geeft de object-handle van het afbeeldingsobject als resultaat.
ID	-1	Geeft het afbeeldingsobject aan door middel van een numerieke waarde.
Left	Geen	Geeft de positie van de linkerkant van het kader aan.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op het afbeeldingsobject staat.
Move( )	Geen	Verplaatst het afbeeldingsobject of wijzigt de grootte van het afbeeldingsobject.
Name	AFBEELDING1	Geeft de naam aan van het afbeeldingsobject.
OnLeftDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het afbeeldingsobject.
OnLeftMouseDown	Geen	Voert een subroutine uit als de gebruiker klikt op het afbeeldingsobject.
OnLeftMouseUp	Geen	Voert een subroutine uit als de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op het afbeeldingsobject staat.
OnMiddleDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het afbeeldingsobject met de middelste muisknop.
OnMiddleMouseDown	Geen	Voert een subroutine uit als de gebruiker met de middelste muisknop op het afbeeldingsobject klikt.
OnMiddleMouseUp	Geen	Voert een subroutine uit als de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op het afbeeldingsobject staat.
OnMouseMove	Geen	Voert een subroutine uit als de gebruiker de muis over het afbeeldingsobject beweegt.
OnOpen	Geen	Voert een subroutine uit als het hoofdformulier wordt geopend.
OnRightDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het afbeeldingsobject met de rechtermuisknop.

Kenmerk	Standaardinstelling	Beschrijving
OnRightMouseDown	Geen	Voert een subroutine uit als de gebruiker rechtsklikt op het afbeeldingsobject.
OnRightMouseUp	Geen	Voert een subroutine uit als de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op het afbeeldingsobject staat.
Parent	Geen	Een objectverwijzing die naar het hoofdformulier wijst.
Release( )	Geen	Verwijdert de definitie van het afbeeldingsobject uit het geheugen.
Top	Geen	Stelt de positie van de bovenkant van het kader in.
Visible	.T.	Bepaalt of het afbeeldingsobject zichtbaar of verborgen is.
Width	Geen	Stelt de breedte in.

Beschrijving

Met een afbeeldingsobject kunt u afbeeldingen weergeven op een formulier. Een afbeeldingsobject kan bijvoorbeeld afbeeldingen tonen die zijn opgeslagen in een binair veld.

Met het kenmerk DataSource kunt u een afbeeldingsobject aan een van de volgende drie bronnen koppelen:

- Een bestand met een bitmap-afbeelding (.BMP of .PCX)
- Een binair veld met bitmap-afbeeldingen
- Een bitmap-resource in een DLL-bestand

Als u een afbeeldingsobject koppelt aan een binair veld en de gebruiker bladert van record naar record, wordt telkens de afbeelding uit het actieve record getoond.

De afbeelding in een afbeeldingsobject kan niet worden gewijzigd.

Als u een afbeeldingsobject maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing die naar het hoofdformulier wijst.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van het nieuwe afbeeldingsobject. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een afbeeldingsobject dat op het formulier wordt weergegeven:

```
MijnForm = NEW FORM()
MijnAfb = NEW IMAGE(MijnForm, "Afbeelding")
```

Het kenmerk Name van het nieuwe afbeeldingsobject bevat "Afbeelding".

Opmerking U kunt een bitmap-afbeelding kiezen met het dialoogvenster **Bitmap kiezen**. Klik op de hulpmiddelenknop naast het element DataSource in het kenmerkenvenster om toegang te krijgen tot het dialoogvenster **Bitmap kiezen**.

Voorbeeld

In het volgende voorbeeld wordt een subklasse van Form gemaakt die een subklasse van Image bevat. De subklasse van Image geeft een afbeelding weer in het formulier:

```
LOCAL ToonVliegtuig
ToonVliegtuig = NEW VlgtgForm()
ToonVliegtuig.OPEN()

CLASS VlgtgForm OF FORM
      NEW VliegtuigBMP(this)
ENDCLASS

CLASS VliegtuigBMP(form) OF IMAGE(form)
      this.DataSource = "FILE Fokker50.BMP"
ENDCLASS
```

Zie CLASS LISTBOX voor een ander voorbeeld met CLASS IMAGE.

Zie ook

DEFINE, RESTORE IMAGE

CLASS LINE

Een lijnobject geeft een lijn weer op een opgegeven positie op een formulier.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Line. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Bottom	Geen	Stelt de rijpositie in van de onderkant van het lijnobject.
ClassName	LINE	Geeft de klasse van het lijnobject aan.
ColorNormal	N	Bepaalt de kleur van het lijnobject.
Left	Geen	Stelt de horizontale positie van de linkerkant van het lijnobject in.
Name	LIJN1	Geeft de naam aan van het lijnobject.
OnOpen	.F.	Voert een subroutine uit als het hoofdformulier wordt geopend.
Parent	Geen	Een objectverwijzing die naar het hoofdformulier wijst.
Pen	0 (Effen)	Bepaalt de stijl van het lijnobject.
Release( )	Geen	Verwijdert de definitie van het lijnobject uit het geheugen.
Right	Geen	Stelt de kolompositie in van de rechterkant van het lijnobject.
Top	Geen	Stelt de rijpositie in van de bovenkant van het lijnobject.
Visible	.T.	Bepaalt of het lijnobject zichtbaar of verborgen is.
Width	Geen	Stelt de dikte van het lijnobject in.

Beschrijving

Gebruik een lijnobject om een lijn te tekenen op een formulier. Een lijnobject kan een ander object onderstrepen of een scheiding aangeven tussen twee gebieden op een formulier.

De gebruiker kan de focus niet naar een lijnobject verplaatsen.

Als u een lijnobject maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing die naar het hoofdformulier wijst.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van het nieuwe lijnobject. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een lijnobject om op het formulier weer te geven:

```
MijnForm = NEW FORM()
MijnLijn = NEW LINE(MijnForm, "Lijn")
```

Het kenmerk Name van het nieuwe lijnobject bevat "Lijn".

Voorbeeld

In het volgende voorbeeld wordt DEFINE LINE gebruikt, binnen een Class-definitie, om op het formulier twee magenta lijnen te maken (een verticale en een horizontale lijn):

```
LOCAL f
f=NEW InvoerForm()
f.OPEN()
CLASS InvoerForm OF FORM
  this.Top=2
  this.Left=2
  this.Width=36
  this.Height=13
  this.Text= "Demo van klasse Line"
*
* Overige objectdefinities
*
  DEFINE LINE Ln1 OF THIS;
    PROPERTY;
      Left 10,;
      Top 3,;
      Width 4,;
      Bottom 8,;
      ColorNormal "RB"
  DEFINE LINE Ln2 OF THIS;
    PROPERTY;
      Left 3,;
      Top 8,;
      Width 4,;
      Bottom 8,;
      Right 33,;
      ColorNormal "RB"
ENDCLASS
```

Zie ook

CLASS RECTANGLE , DEFINE

CLASS LISTBOX

Een keuzelijst is een object waarmee de gebruiker een keuze kan maken uit een of meer waarden in een lijst.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Listbox. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Before	Geen	Bepaalt welk object voor de keuzelijst komt in de tabvolgorde van het hoofdformulier.
ClassName	LISTBOX	Geeft de klasse van het keuzelijstobject aan.
ColorHighLight	W+/B	Bepaalt de kleur van de keuzelijst als het object is geselecteerd.
ColorNormal	N/W*	Bepaalt de kleur van de keuzelijst als het object niet is geselecteerd.
Count( )	Geen	Geeft het aantal keuzemogelijkheden in de keuzelijst als resultaat.
CurSel	0	Bepaalt de huidige, geselecteerde keuzemogelijkheid in de keuzelijst.
DataSource	Lege tekenreeks	Bepaalt welke gegevens in de keuzelijst worden weergegeven.
Enabled	.T.	Bepaalt of de keuzelijst kan worden geselecteerd.
FontBold	.T.	Bepaalt of tekens vet worden weergegeven.
FontItalic	.F.	Bepaalt of tekens cursief worden weergegeven.
FontName	MS Sans Serif	Bepaalt het font voor de weergegeven tekens.
FontSize	Geen	Bepaalt de grootte van het font in punten.
FontStrikeOut	.F.	Bepaalt of tekens doorgehaald worden weergegeven.
FontUnderline	.F.	Bepaalt of tekens onderstreept worden weergegeven.
Height	Geen	Bepaalt de hoogte.
HelpFile	Lege tekenreeks	Geeft een Windows-Helpbestand (.HLP-bestand) aan dat contextgevoelige Help bevat.
HelpID	Lege tekenreeks	Geeft de contextreeks of het contextnummer aan van een Help-onderwerp in een Windows-Helpbestand (.HLP-bestand).
hWnd	Geen	Geeft de object-handle van de het keuzelijst-object als resultaat.
ID	-1	Geeft de keuzelijst aan door middel van een numerieke waarde.
Left	Geen	Geeft de positie van de linkerkant van het kader aan.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op de keuzelijst staat .

Kenmerk	Standaardinstelling	Beschrijving
Move( )	Geen	Verplaatst, vergroot of verkleint de keuzelijst.
Multiple	.F.	Bepaalt of meer dan één element in de keuzelijst kan worden geselecteerd.
Name	KEUZELIJST1	Bepaalt de naam van de keuzelijst.
OldStyle	Geen	Bepaalt of het lijstvak wordt weergegeven in de standaardstijl van Windows of in de stijl van Borland.
OnGotFocus	Geen	Voert een subroutine uit als de keuzelijst de focus krijgt.
OnHelp	Geen	Voert een subroutine uit als de gebruiker op *F1* drukt.
OnLeftDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op de keuzelijst.
OnLeftMouseDown	Geen	Voert een subroutine uit als de gebruiker klikt op de keuzelijst.
OnLeftMouseUp	Geen	Voert een subroutine uit als de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op de keuzelijst staat.
OnLostFocus	Geen	Voert een subroutine uit als de focus het object verlaat.
OnMiddleDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op de keuzelijst met de middelste muisknop.
OnMiddleMouseDown	Geen	Voert een subroutine uit als de gebruiker met de middelste muisknop op de keuzelijst klikt.
OnMiddleMouseUp	Geen	Voert een subroutine uit als de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op de keuzelijst staat.
OnMouseMove	Geen	Voert een subroutine uit als de gebruiker de muis over de keuzelijst beweegt.
OnOpen	Geen	Voert een subroutine uit als het hoofdformulier wordt geopend.
OnRightDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op de keuzelijst met de rechtermuisknop.
OnRightMouseDown	Geen	Voert een subroutine uit als de gebruiker rechtsklikt op de keuzelijst.
OnRightMouseUp	Geen	Voert een subroutine uit als de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op de keuzelijst staat.
OnSelChange	Geen	Voert een subroutine uit als de selectiebalk naar een andere keuzemogelijkheid in de keuzelijst wordt verplaatst.
Parent	Geen	Een objectverwijzing die naar het hoofdformulier wijst.
Release( )	Geen	Verwijdert de definitie van de keuzelijst uit het geheugen.
Selected( )	Geen	Geeft de geselecteerde keuzemogelijkheid als resultaat.
SetFocus( )	Geen	Verplaatst de focus naar de keuzelijst.
Sorted	.F.	Bepaalt of de keuzemogelijkheden in de keuzelijst de sorteervolgorde of de natuurlijke volgorde hebben.
StatusMessage	Lege tekenreeks	Geeft een melding aan om te tonen op de statusbalk.
TabStop	.T.	Bepaalt of de gebruiker de focus naar de keuzelijst kan verplaatsen met *Tab* of *Shift+Tab*.
Top	Geen	Stelt de positie van de bovenkant van het kader in.
Value	Lege tekenreeks	De geselecteerde keuzemogelijkheid in de keuzelijst.
Visible	.T.	Bepaalt of de keuzelijst zichtbaar of verborgen is.

Kenmerk	Standaardinstelling	Beschrijving
When	Geen	Geeft een voorwaarde aan die waar moet zijn voordat de gebruiker de focus naar de keuzelijst kan verplaatsen.
Width	Geen	Stelt de breedte in.

Beschrijving

Met een keuzelijst kunt u de gebruiker een keuze laten maken uit een lijst met mogelijkheden. Een applicatie kan bijvoorbeeld een keuzelijst met bestandsnamen weergeven, zodat de gebruiker kan aangeven welk bestand moet worden geopend.

De keuzemogelijkheden in de keuzelijst geeft u op met het kenmerk DataSource. U kunt vijf soorten keuzemogelijkheden weergeven in een keuzelijst:

1 Bestandsnamen en subdirectory's.

2 De inhoud van een tabelveld.

3 Veldnamen uit een tabel.

4 Elementen in een array-object.

5 De namen van alle tabellen in de huidige database. (Zie OPEN DATABASE voor meer informatie over databases.)

De afmetingen van een keuzelijst geeft u op met de clausule FROM...TO van het commando DEFINE of met de kenmerken Height en Width. Als de opgegeven hoogte onvoldoende is voor alle keuzemogelijkheden, kan de gebruiker door de keuzemogelijkheden bladeren. Als de hoogte groter is dan noodzakelijk, wordt deze automatisch teruggebracht.

Als een letter wordt getypt, wordt de eerste keuzemogelijkheid geselecteerd die met die letter begint. Als meer dan een mogelijkheid met dezelfde letter begint, kan nogmaals op dezelfde letter worden getypt om het de volgende keuzemogelijkheid te selecteren die met die letter begint.

Stel waar (.T.) in voor het kenmerk Multiple om de gebruiker in staat te stellen nul, een of meer keuzemogelijkheden te selecteren. Elke gekozen mogelijkheid wordt voorzien van een vinkje. In dit geval spreken we van een *meerkeuzelijst*.

U kunt op twee manier bepalen welke keuzemogelijkheid of -mogelijkheden de gebruiker heeft geselecteerd:

- Test de inhoud van het kenmerk Value. Dit kenmerk bevat de huidige keuze in een gewone keuzelijst of de laatste keuze in een meerkeuzelijst.
- Controleer de keuzemogelijkheden met de functies LISTSELECTED( ) en LISTCOUNT( ) in een DO...WHILE-lus. Zie de beschrijvingen bij LISTSELECTED( ) en LISTCOUNT( ) voor meer informatie.

Als u een keuzelijst maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing die naar het hoofdformulier wijst.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van de nieuwe keuzelijst. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een keuzelijst om op het formulier weer te geven:

```
MijnForm = NEW FORM()
MijnLijst = NEW LISTBOX(MijnForm, "Keuzelijst")
```

Het kenmerk Name van de nieuwe keuzelijst bevat "Keuzelijst".

Voorbeeld

In het volgende voorbeeld wordt een formulier gedefinieerd dat een keuzelijst bevat waarin de namen worden getoond uit de tabel Dieren.DBF. Tevens bevat het formulier een afbeeldingsobject waarin de bijbehorende .BMP-afbeelding van het dier wordt getoond:

```
LOCAL ToonPlaatjes
ToonPlaatjes=NEW KeuzelijstForm()
ToonPlaatjes.OPEN()
CLASS KeuzelijstForm OF FORM
  this.View="Dieren.DBF"
  this.Top=2
  this.Left=2
  this.Width=60
  this.Height=20
  this.Text= "Dieren van de wereld"
  DEFINE LISTBOX KL1 OF THIS;
    PROPERTY;
      DataSource "FIELD Dieren->Naam",;
      Top 4,;
      Left 6,;
      Width 20,;
      Height 12
  DEFINE TEXT Tekst1 OF THIS;
    PROPERTY;
      Text "Kies uw favoriete dier",;
      FontBold .T.,;
      Width 40,;
      Top 1,;
      Left 3,;
      Height 2.50,;
      FontSize 12.00,;
      ColorNormal "RB/W"
  DEFINE IMAGE Afb1 OF THIS;
    PROPERTY;
      DataSource "BINARY Dieren->BMP",;
      Top 2,;
      Left 32,;
      Width 25,;
      Height 15
ENDCLASS
```

Zie ook

DO WHILE, DEFINE, LISTCOUNT( ), LISTSELECTED( ), ON SELECTION FORM, OPEN FORM

CLASS MENU

Een Windows-menusysteem dat wordt toegewezen aan een formulier.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Menu. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Before	Geen	Bepaalt welke andere menu-object voor dit menu-object komt.
Checked	.F.	Bepaalt of een vinkje verschijnt naast een menu-opdracht.
ClassName	MENU	Geeft de klasse van het menu-object aan.
Enabled	.T.	Bepaalt of het menu kan worden geselecteerd.
ID	-1	Geeft het menu aan door middel van een numerieke waarde.
Name	MENU1	Geeft de naam aan van het menu-element.
OnClick	.F.	Voert een subroutine uit als de gebruiker het menu kiest.
Parent	Lege tekenreeks	Een objectverwijzing die naar het hoofdformulier wijst.
Release( )	Geen	Verwijdert de definitie van het menu-object uit het geheugen.
Separator	.F.	Bepaalt of de menu-opdracht een menu-element is dat niet door de gebruiker kan worden geselecteerd.
Shortcut	Lege tekenreeks	Geeft de toetscombinatie aan die de subroutine OnClick start.
StatusMessage	Lege tekenreeks	Geeft een melding aan om te tonen op de statusbalk.
Text	Lege tekenreeks	Geeft een tekenreeks aan om te tonen als de menu-opdracht.

Beschrijving

Met menu-objecten kunt u een menusysteem maken voor een formulier.

Een menusysteem bestaat uit twee elementen:

- De objectverwijzingsvariabele die het gehele menusysteem aangeeft. U moet deze variabele maken voordat u het menusysteem maakt. (De naam van de variabele wordt nergens weergegeven.)
- Menu-elementen, de opdrachten in het menusysteem. U kunt menu-elementen op vier plaatsen weergeven:
- De hoofdmenubalk van dBASE die langs de bovenkant van het applicatievenster wordt getoond. (Dit is alleen het geval als voor het kenmerk MDI waar (.T.) is ingesteld.)
- De menubalk, een ongemarkeerde rij langs de bovenkant van het formulier. Dit is alleen het geval als voor het kenmerk MDI onwaar (.F.) is ingesteld. Het eerste menu-element dat u maakt, wordt automatisch links op de menubalk geplaatst.
- Een afrolmenu dat wordt geopend als de gebruiker een menu-element kiest op de menubalk.

- Een vervolgmenu dat verschijnt als de gebruiker een menu-opdracht kiest in een afrolmenu of in een ander vervolgmenu.

De objectverwijzingsvariabele en de menu-elementen maakt u met het commando DEFINE MENU. De volgende instructies maken bijvoorbeeld een formulier en definiëren een naam voor het bijbehorende menusysteem:

```
MijnForm = NEW FORM()
DEFINE MENU Hoofd OF MijnForm
```

De volgende instructie maakt een menu-element (Bestand) op de applicatiemenubalk:

```
DEFINE MENU Bestand OF MijnForm.Hoofd PROPERTY Text "Bestand"
```

De volgende instructie verplaatst het menu-element (Bestand) naar de menubalk van het formulier:

```
MijnForm.MDI = .F.
```

De volgende instructies maken twee menu-elementen (Openen en Sluiten) in een afrolmenu:

```
DEFINE MENU xOpenen OF;
   MijnForm.Hoofd.Bestand PROPERTY Text "Openen"
DEFINE MENU xSluiten OF;
   MijnForm.Hoofd.Bestand PROPERTY Text "Sluiten"
```

De gebruiker opent dit (afrol)menu door Bestand te kiezen op de menubalk.

De volgende instructies maken een vervolgmenu met twee menu-elementen (Sluiten zonder opslaan en Sluiten en opslaan):

```
DEFINE MENU NietOpslaan OF MijnForm.Hoofd.Bestand.xSluiten;
   PROPERTY Text "Sluiten zonder opslaan"
DEFINE MENU WelOpslaan OF MijnForm.Hoofd.Bestand.xSluiten;
   PROPERTY Text "Sluiten en opslaan"
```

De gebruiker opent dit vervolgmenu door het menu-element Sluiten in het menu Bestand te kiezen.

Kiestekens maken

Als u wilt dat de gebruiker een menu-opdracht kan kiezen door op een toets te drukken, moet u een kiesteken opgeven. U doet dat door in de menu-opdracht links van het kiesteken een en-teken (&) te plaatsen. In de laatste opdracht had u bijvoorbeeld als volgt de "o" als kiesteken kunnen definiëren:

```
DEFINE MENU WelOpslaan OF MijnForm.Hoofd.Bestand.xSluiten;
   PROPERTY Text "Sluiten en &opslaan"
```

De methode voor het gebruiken van een kiesteken is afhankelijk van het niveau van het menu-element. Als het kiesteken een menu-element op de menubalk selecteert, moet de gebruiker het kiesteken samen met <Alt> indrukken. Elke andere menu-opdracht kan worden gekozen door alleen op de toets met het teken te drukken.

Handelingen toewijzen aan menu-elementen

Met de subroutine OnClick wijzigt u een handeling toe aan een menu-element. De volgende instructie wijst bijvoorbeeld de een subroutine genaamd OpslSluiten toe aan het menu-element Opslaan en sluiten:

```
MijnForm.Hoofd.Bestand.xSluiten.WelOpslaan.OnClick = OpslSluiten
```

Opmerking U kunt een menu-element naar een ander formulier verplaatsen door het kenmerk Parent van het menu-object te wijzigen. Het kenmerk Parent is voor alle klassen alleen-lezen.

Een menu kunt u ontwerpen met het menu-ontwerpvenster, een hulpmiddel voor het maken van een menubestand (.MNU-bestand). Het menubestand bevat dBASE-code die het ontworpen menu voor u genereert. Klik op de hulpmiddelenknop naast het element MenuFile in het kenmerkenvenster om toegang te krijgen tot het menu-ontwerpvenster.

Voorbeeld

In het volgende voorbeeld wordt een hoofdmenu gedefinieerd met twee afrolmenu's: Bestand en Toestellen. Bestand heeft de menu-keuze Afsluiten en Toestellen heeft de menukeuze Vlucht kiezen. Deze menudefinitie komt uit MATRIEEL.MNU. Dit menubestand wordt aangeroepen in MATRIEEL.WFM in de directory DBASEWIN\VOORBD:

```
Parameter FormObj
NEW MATRIEELMENU(FormObj,"Basis")
CLASS MATRIEELMENU(FormObj,Naam) OF MENU(FormObj,Naam)
   this.Text = ""

   DEFINE MENU BESTAND OF THIS;
       PROPERTY;
         Text "&Bestand"

         DEFINE MENU SLUITEN OF THIS.BESTAND;
             PROPERTY;
               OnClick {;form.Release()},;
               Text "&Sluiten",;
               Shortcut "CTRL-Q"

   DEFINE MENU MATERIEEL OF THIS;
       PROPERTY;
         Text "&Materieel"

         DEFINE MENU VLUCHTENSELECTEREN OF THIS.MATERIEEL;
             PROPERTY;
               OnClick CLASS::VLUCHTINFO,;
               Text "&Vlucht kiezen",;
               Shortcut "CTRL-K"

   procedure VluchtInfo
   local vlchtinfForm,selected
   set procedure to Vlchtinf.wfm additive
   getFltsForm = new VlchtinfForm()
   getFltsForm.mdi = .f.
   getFltsForm.Readmodal()
   getFltsForm.Release()
   show object form.vluchtBladeren
   show object form.vliegtuigType
```

```
ENDCLASS
```

Zie ook

_app, DEFINE, MDI

CLASS OBJECT

Hiermee kunt u een geheel nieuw object zonder kenmerken ontwerpen.

Kenmerken

De klasse Object heeft geen ingebouwde kenmerken.

Beschrijving

Met de klasse Object kunt u een eigen object ontwerpen. Een object van de klasse Object is leeg: het bevat geen kenmerken of methoden. U past het object aan door zelf de gewenste kenmerken toe te wijzen.

Voorbeeld

In het volgende voorbeeld wordt een eigen object gemaakt dat de naam en het adres van een klant bevat:

```
oKlant=NEW OBJECT()
oKlant.Voornaam = "Pieter"
oKlant.Achternaam = "Van Hemelen"
oKlant.Adres = "Voorweg 127"
oKlant.Plaats = "Alkmaar"
oKlant.Regio= "NW"
```

Zie ook

CLASS...ENDCLASS

CLASS OLE

Een OLE-object toont een OLE-document dat is opgeslagen in een OLE-veld. Met het OLE-object kan de gebruiker een handeling initiëren in de server-toepassing waarmee het document is gemaakt.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse OLE. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Before	Geen	Bepaalt welk object voor het OLE-object komt in de tabvolgorde van het hoofdformulier.
Border	.T.	Bepaalt of het OLE-object wordt omgeven door een kader.
ClassName	OLEITEM	Geeft de klasse van het OLE-object aan.
DataLink	Lege tekenreeks	Koppelt het OLE-object aan een veld.
DoVerb( )	0	Start een OLE-server-sessie en bepaalt het type.
Enabled	.T.	Bepaalt of het OLE-object kan worden geselecteerd.
Height	Geen	Bepaalt de hoogte.
hWnd	Geen	Geeft de object-handle van het OLE-object als resultaat.
ID	-1	Geeft het OLE-object aan door middel van een numerieke waarde.
Left	Geen	Geeft de positie van de linkerkant van het kader aan.
LinkFileName	Lege tekenreeks	Geeft het OLE-documentbestand (als dat bestaat) dat is gekoppeld aan het huidige OLE-veld.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op het OLE-object staat.
Name	OLE1	Geeft de naam aan van het OLE-object.
OleType	0	Geeft een getal als resultaat dat aangeeft of het OLE-veld leeg is of een ingesloten document of een koppeling met een documentbestand bevat.
OnChange	Geen	Voert een subroutine uit als de gebruiker een document wijzigt.
OnClose	Geen	Voert een subroutine uit als de OLE-serversessie wordt beëindigd.
OnGotFocus	Geen	Voert een subroutine uit als het OLE-object de focus krijgt.
OnLostFocus	Geen	Voert een subroutine uit als de focus het object verlaat.
OnOpen	Geen	Voert een subroutine uit als het hoofdformulier wordt geopend.
Parent	Geen	Een objectverwijzing die naar het hoofdformulier wijst.
Release( )	Geen	Verwijdert de OLE-objectdefinitie uit het geheugen.
SetFocus( )	Geen	Geeft de focus aan het OLE-object.
ServerName	Lege tekenreeks	Geeft de server-toepassing aan die wordt aangeroepen als de gebruiker dubbelklikt op een OLE-object.
StatusMessage	Lege tekenreeks	Geeft een melding aan om te tonen op de statusbalk.
TabStop	.T.	Bepaalt of de gebruiker de focus naar het OLE-object kan verplaatsen met *Tab* of *Shift+Tab*.
Top	Geen	Stelt de positie van de bovenkant van het kader in.
Valid	Geen	Geeft een voorwaarde aan die waar (.T.) moet zijn voordat de gebruiker de focus van het OLE-object kan verplaatsen.
Visible	.T.	Bepaalt of OLE-object zichtbaar of verborgen is.
Width	Geen	Stelt de breedte in.

Beschrijving

Neem een OLE-object op in een formulier om een in een OLE-veld opgeslagen document te bekijken en wijzigen. Als een OLE-veld bijvoorbeeld een bitmap-afbeelding bevat die is gemaakt met Paintbrush, kunt u dubbelklikken op het OLE-object dat gekoppeld is aan het veld om een sessie met Paintbrush te starten. De afbeelding wordt dan in het Paintbrush-werkgebied geplaatst.

OLE is een afkorting van "Object Linking and Embedding", hetgeen koppelen en insluiten van objecten betekent. Als u een document *koppelt* aan een OLE-object, bevat het OLE-veld niet het document zelf, maar een koppeling naar het bestand dat het document bevat. Als u een document insluit in een OLE-veld, wordt in het OLE-veld een kopie van het document geplaatst en wordt geen verbinding gelegd met een documentbestand.

De gebruiker kan dubbelklikken op het OLE-object om de toepassing aan te roepen waarmee het OLE-document is gemaakt. Als een afbeelding is gemaakt met Paintbrush en vervolgens gekoppeld aan of ingesloten in het OLE-veld, hoeft de gebruiker alleen maar te dubbelklikken op het veld om een sessie met Paintbrush te starten. De afbeelding wordt weergegeven in het tekenvenster van Paintbrush zodat die kan worden gewijzigd. Als het document is gekoppeld, worden aangebrachte wijzigingen opgeslagen in het documentbestand. Als het een ingesloten object betreft, worden de wijzigingen alleen opgeslagen in het OLE-veld.

Een OLE-projectorvensterobject toont de inhoud van een OLE-veld. (Met het kenmerk DataLink kunt u dit veld aanduiden met zijn naam.) Als de recordaanwijzer wordt verplaatst, wordt de inhoud van het projectorvenster aangepast aan het OLE-veld in het huidige record.

Als u een OLE-object maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing die naar het hoofdformulier wijst.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van het nieuwe OLE-object. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een OLE-object om op het formulier weer te geven:

```
MijnForm = NEW FORM()
MijnOLE = NEW OLE(MijnForm, "Torrero")
```

Het kenmerk Name van het nieuwe OLE-object bevat "Torrero".

Voorbeeld

Het volgende voorbeeld is een fragment uit AFBEELD.WFM in de directory VOORBD. In het voorbeeld wordt met een OLE-object een OLE-veld uit de tabel Afbeeld weergegeven op een formulier:

```
LOCAL f
f = NEW AFBEELDFORM()
f.Open()

CLASS AFBEELDFORM OF FORM
```

```
Set Procedure To KNOPPEN.CC additive
this.Top =          0
this.Left =          0
this.Height =        19.0586
this.Minimize = .F.
this.Maximize = .F.
this.HelpFile = ""
this.Text = "Afbeeldingenformulier"
this.HelpId = ""
this.MousePointer =          1
this.View = "AFBEELD.QBE"
this.ColorNormal = "BG/B"
this.Width =         91.5

DEFINE PUSHBUTTON GELUID OF THIS;
    PROPERTY;
      Top         11.2793,;
      Left          1.3184,;
      Height          1.4854,;
      OnClick {;play sound binary afbeeld->geluid},;
      Group .T.,;
      Text "&Geluid",;
      Default .T.,;
      ColorNormal "N/W",;
      Width         19.8477

DEFINE LISTBOX DINGEN OF THIS;
    PROPERTY;
      Top          4.5479,;
      Left          1.1592,;
      Height          5.5107,;
      ID        800,;
      DataSource "FIELD NAAM",;
      ColorHighLight "RG+/B",;
      ColorNormal "N/W+",;
      Width         20.1738

DEFINE OLE AFBEELDING OF THIS;
    PROPERTY;
      Top          3.2979,;
      Left         23.0977,;
      Border .T.,;
      Height         15.5254,;
      ID         88,;
      DataLink "AFBEELD->BITMAPOLE",;
      Width         67.0684

DEFINE TEXT TITEL OF THIS;
    PROPERTY;
      FontSize         28,;
      Top          0,;
      FontName "Times New Roman",;
      Left         -9,;
      Border .F.,;
      Height          3,;
```

```
        Alignment          4,;
        Text "OOG " + '&' + '&' + " OOR",;
        ColorNormal "GR+/B",;
        FontBold .F.,;
        Width        100

    DEFINE TEXT KOPTEKST OF THIS;
        PROPERTY;
        Top          3.2979,;
        Left          1.3184,;
        Border .F.,;
        Height          0.9365,;
        Text "Afbeelding:",;
        ColorNormal "RG+/B",;
        Width         17.8477

ENDCLASS

PROCEDURE Geluid_OnClick
PLAY SOUND BINARY Afbeeld->Geluid

PROCEDURE SluitAfbeeld
USE IN AFBEELD
FORM.CLOSE()
```

Zie ook

CLASS DDELINK, CLASS DDETOPIC, CLASS IMAGE, DEFINE, DoVerb( )

CLASS PUSHBUTTON

Een opdrachtknop is een knop die ervoor zorgt dat een opdracht wordt uitgevoerd als de gebruiker op de knop klikt.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Pushbutton. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Before	Geen	Bepaalt welk object voor de opdrachtknop komt in de tabvolgorde van het hoofdformulier.
ClassName	PUSHBUTTON	Geeft de klasse van de opdrachtknop aan.
ColorNormal	N/W	Bepaalt de kleur van de opdrachtknop.
Default	.F.	Bepaalt of de opdrachtknop de standaardopdrachtknop is.
DisabledBitmap	Lege tekenreeks	Geeft de grafische afbeelding aan die op de opdrachtknop wordt weergegeven als de opdrachtknop is uitgeschakeld.

Kenmerk	Standaardinstelling	Beschrijving
DownBitmap	Lege tekenreeks	Geeft de grafische afbeelding aan die op de opdrachtknop wordt weergegeven als de gebruiker op de opdrachtknop klikt, of als de gebruiker op de *Spatiebalk* drukt als de opdrachtknop de focus heeft.
Enabled	.T.	Bepaalt of de opdrachtknop kan worden geselecteerd.
FocusBitmap	Lege tekenreeks	Geeft de grafische afbeelding aan die op de opdrachtknop wordt weergegeven als de opdrachtknop de focus heeft.
FontBold	.T.	Bepaalt of tekens vet worden weergegeven.
FontItalic	.F.	Bepaalt of tekens cursief worden weergegeven.
FontName	MS Sans Serif	Bepaalt het font voor de weergegeven tekens.
FontSize	Geen	Bepaalt de grootte van het weergegeven font in punten.
FontStrikeOut	.F.	Bepaalt of tekens doorgehaald worden weergegeven.
FontUnderline	.F.	Bepaalt of tekens onderstreept worden weergegeven.
Group	.F.	Begint een objectgroep in het hoofdformulier.
Height	Geen	Bepaalt de hoogte.
HelpFile	Lege tekenreeks	Geeft een Windows-Helpbestand (.HLP-bestand) aan dat contextgevoelige Help bevat.
HelpID	Lege tekenreeks	Geeft de contextreeks of het contextnummer aan van een Help-onderwerp in een Windows-Helpbestand (.HLP-bestand).
hWnd	Geen	Geeft de object-handle van het opdrachtknop-object als resultaat.
ID	-1	Geeft de opdrachtknop aan door middel van een numerieke waarde.
Left	Geen	Geeft de positie van de linkerkant van het kader aan.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op de opdrachtknop staat.
Move( )	Geen	Verplaatst de opdrachtknop of wijzigt de grootte van de opdrachtknop.
Name	KNOP1	Geeft de naam aan van de opdrachtknop.
OnClick	Geen	Voert een subroutine uit als de gebruiker de opdrachtknop kiest.
OnGotFocus	Geen	Voert een subroutine uit als de opdrachtknop de focus krijgt.
OnHelp	Geen	Voert een subroutine uit als de gebruiker op *F1* drukt.
OnLeftDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op de opdrachtknop.
OnLeftMouseDown	Geen	Voert een subroutine uit als de gebruiker klikt op de opdrachtknop.
OnLeftMouseUp	Geen	Voert een subroutine uit als de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op de opdrachtknop staat.
OnLostFocus	Geen	Voert een subroutine uit als de focus het object verlaat.
OnMiddleDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op de opdrachtknop met de middelste muisknop.
OnMiddleMouseDown	Geen	Voert een subroutine uit als de gebruiker met de middelste muisknop op de opdrachtknop klikt.

Kenmerk	Standaardinstelling	Beschrijving
OnMiddleMouseUp	Geen	Voert een subroutine uit als de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op de opdrachtknop staat.
OnMouseMove	Geen	Voert een subroutine uit als de gebruiker de muis over de opdrachtknop beweegt.
OnOpen	Geen	Voert een subroutine uit als het hoofdformulier wordt geopend.
OnRightDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op de opdrachtknop met de rechtermuisknop.
OnRightMouseDown	Geen	Voert een subroutine uit als de gebruiker rechtsklikt op de opdrachtknop.
OnRightMouseUp	Geen	Voert een subroutine uit als de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op de opdrachtknop staat.
Parent	Geen	Een objectverwijzing die naar het hoofdformulier wijst.
Release( )	Geen	Verwijdert de definitie van de opdrachtknop uit het geheugen.
SetFocus( )	Geen	Verplaatst de focus naar de opdrachtknop.
SpeedBar	.F.	Bepaalt of de opdrachtknop zich gedraagt als een knop op de knoppenbalk of als een standaardopdrachtknop.
StatusMessage	Lege tekenreeks	Geeft een melding aan om te tonen op de statusbalk.
TabStop	.T.	Bepaalt of de gebruiker de focus naar de opdrachtknop kan verplaatsen met *Tab* of *Shift+Tab*.
Text	Knop1	Geeft een tekenreeks aan om op de opdrachtknop te tonen.
Top	Geen	Stelt de positie van de bovenkant van het kader in.
UpBitmap	Lege tekenreeks	Geeft de grafische afbeelding aan die op de opdrachtknop wordt weergegeven als de opdrachtknop niet is geselecteerd.
Visible	.T.	Bepaalt of de opdrachtknop zichtbaar of verborgen is.
When	Geen	Geeft een voorwaarde aan die waar moet zijn voordat de gebruiker de focus naar de opdrachtknop kan verplaatsen.
Width	Geen	Stelt de breedte in.

Beschrijving

Gebruik een opdrachtknop om een bepaalde handeling uit te voeren als de gebruiker op de knop klikt (of op de *Spatiebalk* drukt als de knop de focus heeft).

Als de gebruiker op een opdrachtknop klikt, gebeuren er drie dingen:

- De procedure of het codeblok dat u hebt toegewezen aan het kenmerk OnGotFocus, wordt uitgevoerd.
- De procedure of het codeblok dat u hebt toegewezen aan het kenmerk OnClick, wordt uitgevoerd.

- Het hoofdformulier wordt *voorgelegd*, hetgeen tot gevolg heeft dat de procedure of het codeblok wordt uitgevoerd dat u hebt toegewezen aan het kenmerk OnSelection van het hoofdformulier.

In de procedure of het codeblok dat u opgeeft bij OnSelection, wordt de opdrachtknop aangegeven met het kenmerk ID. Als de gebruiker op de opdrachtknop klikt, wordt deze waarde doorgegeven aan de procedure of het code blok dat u hebt toegewezen aan het kenmerk OnSelection. De procedure of het codeblok kan Id gebruiken om te bepalen welk object het laatst is geselecteerd.

Als u een opdrachtknop maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing die naar het hoofdformulier wijst.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van de nieuwe opdrachtknop. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een opdrachtknop om op het formulier weer te geven:

```
MijnForm = NEW FORM()
MijnKnop = NEW PUSHBUTTON(MijnForm, "Opdrachtknop")
```

Het kenmerk Name van de nieuwe opdrachtknop bevat "Opdrachtknop".

Opmerking Een OnClick- of OnSelection-procedure kunt u schrijven met de Procedure-editor. Dat is een venster waarin u programmacode kunt invoeren. Klik op de hulpmiddelenknop naast het element OnClick of het element OnSelection in het kenmerkenvenster om toegang te krijgen tot de Procedure-editor.

Voorbeeld

In het volgende voorbeeld wordt een formulier met twee velden uit de tabel Contact gedefinieerd. Op drie verschillende wijzen worden opdrachtknoppen gedefinieerd voor het vooruit en achteruit verplaatsen van de recordaanwijzer en voor het afsluiten van het formulier:

```
SET PROCEDURE TO KNOPPEN.CC ADDITIVE
* Eigen objecten VORIGEKNOP, VOLGENDEKNOP
* en ANNULEERKNOP beschikbaar maken
LOCAL f
f=NEW InvoerForm()
f.OPEN()
CLASS InvoerForm OF FORM
  this.Top=2
  this.Left=2
  this.Width=44
  this.Height=27
  this.View = "Contact.DBF"
  this.Text= "Demo van klasse Pushbutton"
  DEFINE ENTRYFIELD CompCode OF THIS;
    PROPERTY;
      Width 7,;
      Top 1,;
      Left 2,;
      Height 1.5,;
      FontBold .T.,;
```

```
    DataLink "COMPCODE"
DEFINE ENTRYFIELD Contact OF THIS;
  PROPERTY;
    Width 22,;
    Top 1,;
    Left 12,;
    Height 1.5,;
    FontBold .T.,;
    DataLink "CONTACT"
DEFINE PUSHBUTTON Vorigel OF THIS;
  PROPERTY;
    OnClick {;SKIP-1},;
    Text "V&orige",;
    Top  4,;
    Left 6,;
    Width 15,;
    Height 1.5,;
    Group .T.,;
    FontSize 8,;
    FocusBitmap "Resource #104 DBAS0009.DLL",;
    DownBitmap "Resource #104 DBAS0009.DLL",;
    UpBitmap "Resource #104 DBAS0009.DLL"
DEFINE PUSHBUTTON Volgendel OF THIS;
  PROPERTY;
    OnClick {;SKIP},;
    Text "Vo&lgende",;
    Top  4,;
    Left 21,;
    Width 15,;
    Height 1.5,;
    Group .T.,;
    FontSize 8,;
    FocusBitmap "Resource #100 DBAS0009.DLL",;
    DownBitmap "Resource #100 DBAS0009.DLL",;
    UpBitmap "Resource #100 DBAS0009.DLL"
DEFINE PUSHBUTTON Annuleren1 OF THIS;
  PROPERTY;
    OnClick {;Form.Close()},;
    Text "&Annuleren",;
    Top  7,;
    Left 13,;
    Height 1.5,;
    Width 15,;
    Group .T.,;
    FontSize 8,;
    FocusBitmap "Resource #28 DBAS0009.DLL",;
    DownBitmap "Resource #29 DBAS0009.DLL",;
    UpBitmap "Resource #28 DBAS0009.DLL"
DEFINE LINE Lijn1 OF THIS;
  PROPERTY Left 6, Top 11, Width 4,;
    Bottom 11, Right 36, ColorNormal "RB/W"
DEFINE PUSHBUTTON Vorige2 OF THIS AT 13,6;
  PROPERTY Text "V&orige", Width 15, Height 1.5,;
    OnClick {;SKIP-1}, FontBold .T.
DEFINE PUSHBUTTON Volgende2 OF THIS AT 13,21;
```

```
    PROPERTY TEXT "Vo&lgende", Width 15, Height 1.5,;
      OnClick {;SKIP}, FontBold .T.
  DEFINE PUSHBUTTON Annuleren2 OF THIS AT 16,13;
    PROPERTY Text "&Annuleren", Width 15, Height 1.5,;
      OnClick {;Form.Close()}, FontBold .T.
  DEFINE LINE Lijn2 OF THIS;
    PROPERTY Left 6, Top 19, Width 4,;
      Bottom 19, Right 36, ColorNormal "RB/W"
* Volgende drie knoppen zijn reeds gedefinieerd in
* Knoppen.CC
  DEFINE VORIGEKNOP Vorige3 OF THIS;
    PROPERTY;
      Top 21 ,;
      Left 6
  DEFINE VOLGENDEKNOP Volgende3 OF THIS;
    PROPERTY;
      Top 21,;
      Left 21
  DEFINE ANNULEERKNOP Annuleren3 OF THIS;
    PROPERTY;
      Top  24,;
      Left 13
ENDCLASS
```

Zie ook

CLASS FORM, DEFINE, ON SELECTION FORM

CLASS RADIOBUTTON

Een keuzerondje is een object dat één keuzemogelijkheid binnen een groep elkaar wederzijds uitsluitende keuzemogelijkheden voorstelt.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Radiobutton. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Before	Geen	Bepaalt welk object voor het keuzerondje komt in de tabvolgorde van het hoofdformulier.
ClassName	RADIOBUTTON	Geeft de klasse van het keuzerondje aan.
ColorNormal	N/W	Bepaalt de kleur van het keuzerondje.
DataLink	Lege tekenreeks	Koppelt het keuzerondje aan een veld.
Enabled	.T.	Bepaalt of het keuzerondje kan worden geselecteerd.
FontBold	.T.	Bepaalt of tekens in het keuzerondje vet worden weergegeven.
FontItalic	.F.	Bepaalt of tekens cursief worden weergegeven.
FontName	MS Sans Serif	Bepaalt het font voor de weergegeven tekens.

Kenmerk	Standaardinstelling	Beschrijving
FontSize	Geen	Bepaalt de grootte van het font in punten.
FontStrikeOut	.F.	Bepaalt of tekens doorgehaald worden weergegeven.
FontUnderline	.F.	Bepaalt of tekens onderstreept worden weergegeven.
Group	.F.	Begint een objectgroep in het hoofdformulier.
Height	Geen	Bepaalt de hoogte.
HelpFile	Lege tekenreeks	Geeft een Windows-Helpbestand (.HLP-bestand) aan dat contextgevoelige Help bevat.
HelpID	Lege tekenreeks	Geeft de contextreeks of het contextnummer aan van een Help-onderwerp in een Windows-Helpbestand (.HLP-bestand).
hWnd	Geen	Geeft de object-handle van het keuzerondje-object als resultaat.
ID	-1	Geeft het keuzerondje aan door middel van een numerieke waarde.
Left	Geen	Geeft de positie van de linkerkant van het kader aan.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op het keuzerondje staat.
Move( )	Geen	Verplaatst het keuzerondje of wijzigt de grootte van het keuzerondje.
Name	KEUZERONDJE1	Geeft de naam aan van het keuzerondje.
OldStyle	.F.	Bepaalt of het keuzerondje wordt weergegeven in de standaardstijl van Windows of in de stijl van Borland.
OnChange	Geen	Voert een subroutine uit als de gebruiker een ander keuzerondje kiest.
OnGotFocus	Geen	Voert een subroutine uit als het keuzerondje de focus krijgt.
OnHelp	Geen	Voert een subroutine uit als de gebruiker op F1 drukt.
OnLeftDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het keuzerondje.
OnLeftMouseDown	Geen	Voert een subroutine uit als de gebruiker klikt op het keuzerondje.
OnLeftMouseUp	Geen	Voert een subroutine uit als de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op het keuzerondje staat.
OnLostFocus	Geen	Voert een subroutine uit als de focus het object verlaat.
OnMiddleDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het keuzerondje met de middelste muisknop.
OnMiddleMouseDown	Geen	Voert een subroutine uit als de gebruiker met de middelste muisknop op het keuzerondje klikt.
OnMiddleMouseUp	Geen	Voert een subroutine uit als de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op het keuzerondje staat.
OnMouseMove	Geen	Voert een subroutine uit als de gebruiker de muis over het keuzerondje beweegt.
OnOpen	Geen	Voert een subroutine uit als het hoofdformulier wordt geopend.
OnRightDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het keuzerondje met de rechtermuisknop.

Kenmerk	Standaardinstelling	Beschrijving
OnRightMouseDown	Geen	Voert een subroutine uit als de gebruiker rechtsklikt op het keuzerondje.
OnRightMouseUp	Geen	Voert een subroutine uit als de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op het keuzerondje staat.
Parent	Geen	Een objectverwijzing die naar het hoofdformulier wijst.
Release( )	Geen	Verwijdert de definitie van het keuzerondje uit het geheugen.
SetFocus( )	Geen	Verplaatst de focus naar het keuzerondje.
StatusMessage	Lege tekenreeks	Geeft een melding aan om te tonen op de statusbalk.
TabStop	.T.	Bepaalt of de gebruiker de focus naar het keuzerondje kan verplaatsen met *Tab* of *Shift+Tab*.
Text	Keuzerondje1	Geeft een tekenreeks aan die naast het keuzerondje moet worden getoond.
Top	N/A	Stelt de positie van de bovenkant van het kader in.
Value	Lege tekenreeks	Stelt de waarde van het keuzerondje in.
Visible	.T.	Bepaalt of het keuzerondje zichtbaar of verborgen is.
When	Geen	Geeft een voorwaarde aan die waar moet zijn voordat de gebruiker de focus naar het keuzerondje kan verplaatsen.
Width	N/A	Stelt de breedte in.

Beschrijving

Met een groep keuzerondjes kunt u de gebruiker een keuze laten maken uit een groep mogelijkheden die elkaar uitsluiten. In de groep kan maar één keuzerondje zijn geselecteerd. Een toepassing voor het afdrukken van een rapport kan bijvoorbeeld een groep van drie keuzerondjes bevatten zodat de gebruiker kan kiezen tussen "Printer", "Scherm" en "Bestand".

Met het kenmerk Group groepeert u twee of meer keuzerondjes. Als u bijvoorbeeld zeven keuzerondjes maakt en u stel het kenmerk Group van het eerste en het vierde keuzerondje in op waar (.T.), vormen de eerste drie keuzerondjes een groep en de laatste vier keuzerondjes een andere groep. De beide groepen zijn onafhankelijk van elkaar. De gebruiker kan een keuzerondje kiezen in de eerste groep en één in de tweede groep.

Met het kenmerk DataLink koppelt u het keuzerondje aan een veld. Gebruik het kenmerk Text om een waarde op te geven die in het veld wordt ingevoegd als het keuzerondje wordt geselecteerd. De waarde tekst wordt automatisch naast het keuzerondje weergegeven.

Als u een keuzerondje maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing die naar het hoofdformulier wijst.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van het nieuwe keuzerondje. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een keuzerondje om op het formulier weer te geven:

```
MijnForm = NEW FORM()
MijnKRondje = NEW RADIOBUTTON(MijnForm, "Keuzerondje")
```

Het kenmerk Name van het nieuwe keuzerondje bevat "Keuzerondje".

Opmerking Met het dialoogvenster **Veld kiezen** kunt u een veld opgeven voor het kenmerk DataLink. Klik op de hulpmiddelenknop naast het element DataLink in het kenmerkenvenster om toegang te krijgen tot het dialoogvenster **Veld kiezen**.

Voorbeeld

In het volgende voorbeeld wordt een invoerformulier gemaakt met drie keuzerondjes voor conversiefactoren. Nadat op de opdrachtknop Berekenen is geklikt, worden numerieke waarden die in het invoervak zijn ingevoerd, omgezet naar de gekozen maateenheid:

```
** Programma voor getalconversie **
SET PROCEDURE TO PROGRAM(1) ADDITIVE
LOCAL f
f=NEW Conversie()
f.OPEN()
CLASS Conversie OF FORM
this.Top=2
  this.Left=2
  this.Width=42
  this.Height=18
  this.Text= "Conversiehulpmiddel"
  DEFINE ENTRYFIELD Aantal OF THIS AT 4,15;
    PROPERTY Value 0, Width 8
  DEFINE TEXT Regel1 OF THIS AT 2,3;
    PROPERTY;
    Text "Geef getal op en kies keuzerondje",;
    Width 43
  DEFINE RADIOBUTTON Inches OF THIS AT 6,5;
    PROPERTY Text "Inches naar centimeters",;
    Width 34, Value .F.
  DEFINE RADIOBUTTON Ponden OF THIS AT 8,5;
    PROPERTY Text "Amerikaanse ponden naar kilo's",;
    Width 34, Value .F.
  DEFINE RADIOBUTTON Graden OF THIS AT 10,5;
    PROPERTY Text "Graden F naar C",;
    Width 34, Value .F.
  DEFINE TEXT Ln2 OF THIS;
    PROPERTY Text "Resultaat:",;
    Width 30, Top 12, Left 8,;
    ColorNormal "R/W"
  DEFINE PUSHBUTTON Resultaat OF THIS;
    PROPERTY TEXT "&Berekenen",;
    Top 15, Width 20, Left 15,;
    OnClick {;mijnResultaat=Metric(form);
    ;Form.Ln2.Text="Results: " + MijnResultaat}
ENDCLASS

FUNCTION Metric(pForm)
DO CASE
   CASE pForm.Inches.Value
```

```
      mijnResultaat = LTRIM(STR(pForm.Aantal.Value;
      * 2.54,10,2))+" centimeters"
   CASE pForm.Ponden.Value
      mijnResultaat = LTRIM(STR(pForm.Aantal.Value;
      * .454,10,2))+" kilo's"
   CASE pForm.Graden.Value
      mijnResultaat = LTRIM(STR((pForm.Aantal.Value;
      -32)* (5/9),10,2))+" graden C"
ENDCASE
RETURN mijnResultaat
```

Zie ook

DEFINE

CLASS RECTANGLE

Een kader is een object dat u kunt tonen op een opgegeven plaats op een formulier.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Rectangle. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Border	.T.	Bepaalt of het kader wordt omgeven door een kader.
BorderStyle	Normaal	Bepaalt het type rechthoek: Normaal, Omhoog (in haut-relief) of Omlaag (in bas-relief).
ClassName	RECTANGLE	Geeft de klasse van het kader aan.
ColorNormal	N/W	Bepaalt de kleur van het kaderobject.
FontBold	.F.	Bepaalt of tekens in de label van het kaderobject vet worden weergegeven.
FontItalic	.F.	Bepaalt of tekens in de label van het kaderobject cursief worden weergegeven.
FontName	MS Sans Serif	Bepaalt het font voor de weergegeven tekens.
FontSize	Geen	Bepaalt de grootte van het weergegeven font in punten.
FontStrikeOut	.F.	Bepaalt of tekens in de label van het kaderobject doorgehaald worden weergegeven.
FontUnderline	.F.	Bepaalt of tekens in de label van het kaderobject onderstreept worden weergegeven.
Height	Geen	Bepaalt de hoogte.
hWnd	Geen	Geeft de object-handle van het rechthoek-object als resultaat.
Left	Geen	Geeft de positie van de linkerkant van het kader aan.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op het kaderobject staat.
Move( )	Geen	Verplaatst het kaderobject of wijzigt de grootte van het kaderobject.

Kenmerk	Standaardinstelling	Beschrijving
Name	KADER1	Geeft de naam aan van het kaderobject.
OldStyle	.F.	Bepaalt of het kaderobject wordt weergegeven in de standaardstijl van Windows of in de stijl van Borland.
OnLeftDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het kaderobject.
OnLeftMouseDown	Geen	Voert een subroutine uit als de gebruiker klikt op het kaderobject.
OnLeftMouseUp	Geen	Voert een subroutine uit als de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op het kaderobject staat.
OnMiddleDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het kaderobject met de middelste muisknop.
OnMiddleMouseDown	Geen	Voert een subroutine uit als de gebruiker met de middelste muisknop op het kaderobject klikt.
OnMiddleMouseUp	Geen	Voert een subroutine uit als de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op het kaderobject staat.
OnMouseMove	Geen	Voert een subroutine uit als de gebruiker de muis over het kaderobject beweegt.
OnOpen	Geen	Voert een subroutine uit als het hoofdformulier wordt geopend.
OnRightDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het kaderobject met de rechtermuisknop.
OnRightMouseDown	Geen	Voert een subroutine uit als de gebruiker rechtsklikt op het kaderobject.
OnRightMouseUp	Geen	Voert een subroutine uit als de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op het kaderobject staat.
Parent	Lege tekenreeks	Een objectverwijzing die naar het hoofdformulier wijst.
PatternStyle	Effen	Geeft een vooringesteld gearceerd Windows-achtergrondpatroon aan.
Release( )	Geen	Verwijdert de definitie van het kaderobject uit het geheugen.
Text	Kader1	Geeft een tekenreeks aan op te tonen in de label van het kaderobject.
Top	Geen	Stelt de positie van de bovenkant van het kader in.
Visible	.T.	Bepaalt of het kaderobject zichtbaar of verborgen is.
Width	Geen	Stelt de breedte in.

Beschrijving

Met een kaderobject kunt u een gebied op een formulier omsluiten. U kunt een kaderobject bijvoorbeeld gebruiken om een kader te tekenen rond een groep bij elkaar horende objecten, zoals een groep keuzerondjes.

Met het kenmerk Text kunt u een label toewijzen om de groep objecten te beschrijven. De label verschijnt in de linkerbovenhoek van het kader.

Een kaderobject heeft geen invloed op andere objecten. Ook is het niet mogelijk de focus naar een kaderobject te verplaatsen en is het niet mogelijk gegevens in een kaderobject te tonen of te wijzigen.

Als u een kaderobject maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing die naar het hoofdformulier wijst.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van het nieuwe kaderobject. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een kaderobject om op het formulier weer te geven:

```
MijnForm = NEW FORM()
MijnKader = NEW RECTANGLE(MijnForm, "Kader")
```

Het kenmerk Name van het nieuwe kaderobject bevat "Kader".

Voorbeeld

In het volgende voorbeeld wordt DEFINE RECTANGLE gebruikt, binnen een Class-definitie, om in het midden van een formulier een kader met een verlaagde rand te maken. Het visuele effect dat hiermee wordt bereikt, is dat het formulier voorzien lijkt te zijn van een verhoogd kader:

```
LOCAL f
f=NEW DISPLAY()
f.OPEN()
CLASS DISPLAY OF FORM
  this.Top=2
  this.Left=2
  this.Width=50
  this.Height=13
  this.Text= "Demo van klasse Rectangle"
  DEFINE RECTANGLE Kader1 OF THIS;
    PROPERTY;
      Left 5,;
      Top 2,;
      Width 40,;
      Height 9,;
      ColorNormal "RB/W",;
      BorderStyle 2        && Stijl met verlaagde rand
  DEFINE TEXT Tekst1 OF THIS;
    PROPERTY;
      Text "dBASE voor Windows",;
      Top 4, Left 6,;
      FontItalic .T., FontSize 14,;
      Width 38, Height 3, FontBold .T.,;
      ColorNormal "RB/W"
  DEFINE TEXT Tekst2 OF THIS;
    PROPERTY;
      Text "is nu op de markt",;
      Top 7, Left 8,;
      FontItalic .T., FontSize 14,;
      Width 30, Height 3, FontBold .T.
ENDCLASS
```

Zie ook

CLASS LINE

CLASS SCROLLBAR

Een schuifbalk is een object waarmee de gebruiker een waarde kan verhogen of verlagen door een schuifblokje te verplaatsen.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Scrollbar. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Before	Geen	Bepaalt welk object voor de schuifbalk komt in de tabvolgorde van het hoofdformulier.
ClassName	SCROLLBAR	Geeft de klasse van de schuifbalk aan.
DataLink	Lege tekenreeks	Koppelt de schuifbalk aan een veld.
Enabled	.T.	Bepaalt of schuifbalk kan worden geselecteerd.
Height	Geen	Bepaalt de hoogte.
HelpFile	Lege tekenreeks	Geeft een Windows-Helpbestand (.HLP-bestand) aan dat contextgevoelige Help bevat.
HelpID	Lege tekenreeks	Geeft de contextreeks of het contextnummer aan van een Help-onderwerp in een Windows-Helpbestand (.HLP-bestand).
hWnd	Geen	Geeft de object-handle van het schuifbalk-object als resultaat .
ID	-1	Geeft de schuifbalk aan door middel van een numerieke waarde.
Left	Geen	Geeft de positie van de linkerkant van het kader aan.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op de schuifbalk staat.
Move( )	Geen	Verplaatst de schuifbalk of wijzigt de grootte van de schuifbalk.
Name	SCHUIFBALK1	Geeft de naam van de schuifbalk aan.
OnChange	Geen	Voert een subroutine uit als de gebruiker het schuifblokje verplaatst.
OnGotFocus	Geen	Voert een subroutine uit als de schuifbalk de focus krijgt.
OnHelp	Geen	Voert een subroutine uit als de gebruiker op F1 drukt.
OnLeftDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op de schuifbalk.
OnLeftMouseDown	Geen	Voert een subroutine uit als de gebruiker klikt op de schuifbalk.
OnLeftMouseUp	Geen	Voert een subroutine uit als de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op de schuifbalk staat.
OnLostFocus	Geen	Voert een subroutine uit als de focus het object verlaat.

Kenmerk	Standaardinstelling	Beschrijving
OnMiddleDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op de schuifbalk met de middelste muisknop.
OnMiddleMouseDown	Geen	Voert een subroutine uit als de gebruiker met de middelste muisknop op de schuifbalk klikt.
OnMiddleMouseUp	Geen	Voert een subroutine uit als de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op de schuifbalk staat.
OnMouseMove	Geen	Voert een subroutine uit als de gebruiker de muis over de schuifbalk beweegt.
OnOpen	Geen	Voert een subroutine uit als het hoofdformulier wordt geopend.
OnRightDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op de schuifbalk met de rechtermuisknop.
OnRightMouseDown	Geen	Voert een subroutine uit als de gebruiker rechtsklikt op de schuifbalk.
OnRightMouseUp	Geen	Voert een subroutine uit als de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op de schuifbalk staat.
Parent	Geen	Een objectverwijzing die naar het hoofdformulier wijst.
RangeMax	100.00	Bepaalt de bovengrens voor de waarde die aan de schuifbalk is gekoppeld.
RangeMin	1.00	Bepaalt de ondergrens voor de waarde die aan de schuifbalk is gekoppeld.
Release( )	Geen	Verwijdert de definitie van de schuifbalk uit het geheugen.
SetFocus( )	Geen	Verplaatst de focus naar de schuifbalk.
StatusMessage	Lege tekenreeks	Geeft een melding aan om te tonen op de statusbalk.
TabStop	.T.	Bepaalt of de gebruiker de focus naar de schuifbalk kan verplaatsen met *Tab* of *Shift+Tab*.
Top	Geen	Stelt de positie van de bovenkant van het kader in.
Value	1	Stelt de waarde in van de schuifbalk.
Vertical	.T.	Bepaalt of het een verticale of horizontale schuifbalk is.
Visible	.T.	Bepaalt of schuifbalk zichtbaar of verborgen is.
When	Geen	Geeft een voorwaarde aan die waar moet zijn voordat de gebruiker de focus naar de schuifbalk kan verplaatsen.
Width	Geen	Stelt de breedte in.

Beschrijving

Met een schuifbalk kan de gebruiker snel een numerieke waarde wijzigen. In tegenstelling tot ringvelden accepteren schuifbalken geen invoer vanuit het toetsenbord en werken ze niet met een stapwaarde (Step). De gebruiker wijzigt de waarde door het schuifblokje te slepen.

Als de gebruiker het schuifblokje versleept, wordt de waarde doorlopend bijgewerkt om de positie van het schuifblokje aan te geven. Als de numerieke waarde van een

schuifbalk varieert van 1 tot 100 en de startwaarde wordt 50 gemaakt, bevindt het schuifblokje zich in het midden van de schuifbalk.

Met het kenmerk DataLink koppelt u een schuifbalk aan een numeriek veld.

Het bereik voor de schuifbalk stelt u in met de kenmerken RangeMin (ondergrens) en RangeMax (bovengrens).

U kunt een schuifbalk combineren met een invoervak, zodat de waarden die met de schuifbalk worden gewijzigd, worden weergegeven in het invoervak. U doet dat door het kenmerk DataLink van het invoervak dezelfde waarde te geven als het kenmerk DataLink van de schuifbalk.

Als u een schuifbalk maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing die naar het hoofdformulier wijst.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van de nieuwe schuifbalk. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een schuifbalk om op het formulier weer te geven:

```
MijnForm = NEW FORM()
MijnSbalk = NEW SCROLLBAR(MijnForm, "Schuifbalk")
```

Het kenmerk Name van de nieuwe schuifbalk bevat "Schuifbalk".

Voorbeeld

In het volgende voorbeeld wordt een formulier gemaakt met een invoervakobject dat de waarde weergeeft van het veld BNP in Landen.DBF. De waarde in het veld kan met een schuifbalk worden gewijzigd binnen een opgegeven bereik:

```
LOCAL f
f=NEW DISPLAY()
f.OPEN()
CLASS DISPLAY OF FORM
  this.Top=2
  this.Left=2
  this.Width=36
  this.Height=13
  this.View = "Landen.DBF"
  this.Text= "Demo van klasse Scrollbar"
  DEFINE ENTRYFIELD Invoervak1 OF THIS;
    PROPERTY Datalink "Landen->BNP",;
    Width 9, Top 4, Left 13
  DEFINE SCROLLBAR Sb1 OF THIS;
    PROPERTY Vertical .F., Height 1,;
    Top 6, Left 5,;
    Width 25, DataLink "Landen->BNP",;
    RangeMin 100, RangeMax 9999
  DEFINE TEXT Txt1 OF THIS;
    PROPERTY Text "***Bereik: 100 tot 9999***",;
      Top 9, Left 0, Width 36, Alignment 7
ENDCLASS
```

Zie ook

CLASS SPINBOX

CLASS SPINBOX

Een ringveld is een object waarmee de gebruiker waarden kan invoeren in een tekstvak of waarden kan wijzigen door op pijltjes te klikken.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Spinbox. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Before	Geen	Bepaalt welk object voor het ringveld komt in de tabvolgorde van het hoofdformulier.
Border	.T.	Bepaalt of het ringveld wordt omgeven door een kader.
ClassName	SPINBOX	Geeft de klasse aan van het ringveld.
ColorHighLight	N/W	Bepaalt de kleur van het ringveld als het is geselecteerd.
ColorNormal	N/W	Bepaalt de kleur van het ringveld als het niet is geselecteerd.
DataLink	Lege tekenreeks	Koppelt het ringveld aan een veld.
Enabled	.T.	Bepaalt of het ringveld kan worden geselecteerd.
FontBold	.T.	Bepaalt of tekens vet worden weergegeven.
FontItalic	.F.	Bepaalt of tekens cursief worden weergegeven.
FontName	MS Sans Serif	Bepaalt het font voor de weergegeven tekens.
FontSize	Geen	Bepaalt de grootte van het weergegeven font in punten.
FontStrikeOut	.F.	Bepaalt of tekens doorgehaald worden weergegeven.
FontUnderline	.F.	Bepaalt of tekens onderstreept worden weergegeven.
Function	Lege tekenreeks	Maakt weergegeven tekst op.
Height	Geen	Bepaalt de hoogte.
HelpFile	Lege tekenreeks	Geeft een Windows-Helpbestand (.HLP-bestand) aan dat contextgevoelige Help bevat.
HelpID	Lege tekenreeks	Geeft de contextreeks of het contextnummer aan van een Help-onderwerp in een Windows-Helpbestand (.HLP-bestand).
hWnd	Geen	Geeft de object-handle van het ringveld-object als resultaat.
ID	-1	Geeft het ringveld aan door middel van een numerieke waarde.
Left	Geen	Geeft de positie van de linkerkant van het kader aan.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op het ringveld staat.

Kenmerk	Standaardinstelling	Beschrijving
Move( )	.F.	Verplaatst het ringveld of wijzigt de grootte van het ringveld.
Name	RINGVELD1	Geeft de naam aan van het ringveld.
OldStyle	.F.	Bepaalt of het ringveld wordt weergegeven in de standaardstijl van Windows of in de stijl van Borland.
OnChange	Geen	Voert een subroutine uit als de gebruiker een waarde wijzigt.
OnGotFocus	Geen	Voert een subroutine uit als het ringveld de focus krijgt.
OnHelp	Geen	Voert een subroutine uit als de gebruiker op *F1* drukt.
OnLeftDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het ringveld.
OnLeftMouseDown	Geen	Voert een subroutine uit als de gebruiker klikt op het ringveld.
OnLeftMouseUp	Geen	Voert een subroutine uit als de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op het ringveld staat.
OnLostFocus	Geen	Voert een subroutine uit als de focus het object verlaat.
OnMiddleDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het ringveld met de middelste muisknop.
OnMiddleMouseDown	Geen	Voert een subroutine uit als de gebruiker met de middelste muisknop op het ringveld klikt.
OnMiddleMouseUp	Geen	Voert een subroutine uit als de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op het ringveld staat.
OnMouseMove	Geen	Voert een subroutine uit als de gebruiker de muis over het ringveld beweegt.
OnOpen	Geen	Voert een subroutine uit als het hoofdformulier wordt geopend.
OnRightDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het ringveld met de rechtermuisknop.
OnRightMouseDown	Geen	Voert een subroutine uit als de gebruiker rechtsklikt op het ringveld.
OnRightMouseUp	Geen	Voert een subroutine uit als de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op het ringveld staat.
Parent	Lege tekenreeks	Een objectverwijzing die naar het hoofdformulier wijst.
Picture	Lege tekenreeks	Maakt tekst op.
RangeMax	100.00	Bepaalt de bovengrens voor de waarde die aan het ringveld is gekoppeld.
RangeMin	1.00	Bepaalt de ondergrens voor de waarde die aan het ringveld is gekoppeld.
RangeRequired	.F.	Bepaalt of het bereik dat is ingesteld met RangeMax en RangeMin voor alle gegevens geldt of alleen voor nieuwe gegevens.
Release( )	Geen	Verwijdert de definitie van het ringveld uit het geheugen.
SelectAll	.F.	Bepaalt of de startwaarde is geselecteerd.
SetFocus( )	Geen	Verplaatst de focus naar het ringveld.

Kenmerk	Standaardinstelling	Beschrijving
SpinOnly	.F.	Bepaalt of de waarde in het tekstvak van het ringvak kan worden gewijzigd.
StatusMessage	Lege tekenreeks	Geeft een melding aan om te tonen op de statusbalk.
Step	1	Bepaalt met hoeveel de gebruiker een waarde kan verhogen of verlagen door op de pijlknopjes van het ringveld te klikken.
TabStop	.T.	Bepaalt of de gebruiker de focus naar het ringveld kan verplaatsen met *Tab* of *Shift+Tab*.
Top	Geen	Stelt de positie van de bovenkant van het kader in.
Valid	Geen	Geeft een voorwaarde aan die waar (.T.) moet zijn voordat de gebruiker de focus van het ringveld kan verplaatsen.
ValidErrorMsg	Lege tekenreeks	Geeft een tekenreeks aan die op de statusbalk moet worden getoond als het kenmerk Valid onwaar (.F.) als resultaat geeft.
ValidRequired	.F.	Bepaalt of het kenmerk Valid voor alle gegevens of alleen voor nieuwe gegevens geldt.
Value	1	Stelt de waarde in voor het ringveld.
Visible	.T.	Bepaalt of het ringveld zichtbaar of verborgen is.
When	Geen	Geeft een voorwaarde aan die waar moet zijn voordat de gebruiker de focus naar het ringveld kan verplaatsen.
Width	Geen	Stelt de breedte in.

Beschrijving

Een ringveld stelt de gebruiker in staat een waarde in te voeren door die te typen of door de huidige waarde te verhogen of te verlagen met de pijlknopjes.

U kunt instellen met welke stap de knopjes de waarde verhogen of verlagen. In het geval van een ringveld voor een rentepercentage kan die stap eenhonderste zijn, terwijl de stap voor een datumveld een jaar kan zijn. U stelt de stap in met het kenmerk Step. Als u voor Step bijvoorbeeld 5 opgeeft, kunt u op een pijltje klikken om een numerieke waarde te verhogen of te verlagen met 5 of een datumwaarde met 5 dagen.

Met het kenmerk DataLink koppelt u een ringveld aan een numeriek, zwevend of datumveld. Het kenmerk Value bevat de huidige waarde van het veld.

Als u de waarden in het ringveld wilt beperken, kunt u met de kenmerken RangeMin (ondergrens) en RangeMax (bovengrens) een bereik instellen. Als u wilt dat de gebruiker de waarde alleen kan veranderen met de pijlknopjes, moet u voor het kenmerk SpinOnly waar (.T.) instellen.

Als u een ringveld maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing die naar het hoofdformulier wijst.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van het nieuwe ringveld. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een ringveld om op het formulier weer te geven:

```
MijnForm = NEW FORM()
MijnRveld = NEW SPINBOX(MijnForm, "Ringveld")
```

Het kenmerk Name van het nieuwe ringveld bevat "Ringveld".

Voorbeeld

In het volgende voorbeeld wordt een formulier gemaakt met een bladerobject waarin gegevens over de bevolking van een land wordt getoond en een ringveld waarmee het inwonersaantal in stappen van 1000 kan worden verhoogd:

```
LOCAL f
f=NEW DISPLAY()
f.OPEN()
CLASS DISPLAY OF FORM
  this.Top=2
  this.Left=2
  this.Width=36
  this.Height=13
  this.View = "Landen.DBF"
  this.Text= "Demo van klasse SpinBox"
  DEFINE BROWSE BR1 OF THIS;
    PROPERTY Fields "Naam, Bevolking, BNP",;
    Top 1, Left 1.5, Width 32, Height 6
DEFINE TEXT Tekst1 OF THIS;
    PROPERTY Top 9, Left 2, Width 24,;
      Text "Wijzig aantal inwoners:", ColorNormal "R/W"
DEFINE SPINBOX Ringveld1 OF THIS;
    PROPERTY;
      Datalink "Landen->Bevolking",;
      Top 10,;
      Left 2,;
      Height 2.0,;
      Width 22,;
      TabStop .F.,;
      Step 1000
ENDCLASS
```

Zie ook

CLASS SCROLLBAR

CLASS TEXT

Een tekstobject bestaat uit een reeks tekens.

Kenmerken

In de volgende tabel staan de kenmerken van de klasse Text. Zie Hoofdstuk 8 voor meer informatie over elk kenmerk.

Kenmerk	Standaardinstelling	Beschrijving
Alignment	Top Left	Bepaalt de positie van tekst in het tekstobject.
Before	Geen	Bepaalt welk object voor het tekstobject komt in de tabvolgorde van het hoofdformulier.
Border	.T.	Bepaalt of het tekstobject wordt omgeven door een kader.
ClassName	TEXT	Geeft de klasse van het object aan.
ColorNormal	N/W	Bepaalt de kleur van de tekst.
FontBold	.T.	Bepaalt of tekens vet worden weergegeven.
FontItalic	.F.	Bepaalt of tekens cursief worden weergegeven.
FontName	MS Sans Serif	Bepaalt het font voor de weergegeven tekens.
FontSize	Geen	Bepaalt de grootte van het weergegeven font in punten.
FontStrikeOut	.F.	Bepaalt of tekens doorgehaald worden weergegeven.
FontUnderline	.F.	Bepaalt of tekens onderstreept worden weergegeven.
Function	Lege tekenreeks	Maakt de weergegeven tekst op.
GetTextExtent( )	Geen	Geeft de lengte van het tekstobject als resultaat op basis van een vergelijking tussen het font van het tekstobject en het font van het formulier.
Height	Geen	Bepaalt de hoogte.
hWnd	Geen	Geeft de object-handle van het tekstobject als resultaat .
ID	-1	Geeft het tekstobject aan door middel van een numerieke waarde.
Left	Geen	Geeft de positie van de linkerkant van het kader aan.
MousePointer	0	Geeft het type muisaanwijzer aan als de aanwijzer op het tekstobject staat.
Move( )	.F.	Verplaatst het tekstobject of wijzigt de grootte van het tekstobject.
Name	TEKST1	Geeft de naam van het tekstobject aan.
OldStyle	.F.	Bepaalt of het tekstobject wordt weergegeven in de standaardstijl van Windows of in de stijl van Borland.
OnLeftDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het tekstobject.
OnLeftMouseDown	Geen	Voert een subroutine uit als de gebruiker klikt op het tekstobject.
OnLeftMouseUp	Geen	Voert een subroutine uit als de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op het tekstobject staat.
OnMiddleDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het tekstobject met de middelste muisknop.
OnMiddleMouseDown	Geen	Voert een subroutine uit als de gebruiker met de middelste muisknop op het tekstobject klikt.

Kenmerk	Standaardinstelling	Beschrijving
OnMiddleMouseUp	Geen	Voert een subroutine uit als de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op het tekstobject staat.
OnMouseMove	Geen	Voert een subroutine uit als de gebruiker de muis over het tekstobject beweegt.
OnOpen	Geen	Voert een subroutine uit als het hoofdformulier wordt geopend.
OnRightDblClick	Geen	Voert een subroutine uit als de gebruiker dubbelklikt op het tekstobject met de rechtermuisknop.
OnRightMouseDown	Geen	Voert een subroutine uit als de gebruiker rechtsklikt op het tekstobject.
OnRightMouseUp	Geen	Voert een subroutine uit als de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op het tekstobject staat.
Parent	Geen	Een objectverwijzing die naar het hoofdformulier wijst.
Picture	Lege tekenreeks	Bepaalt de opmaak van de tekst.
Release( )	Geen	Verwijdert de definitie van het tekstobject uit het geheugen.
Text	Tekst	Geeft de weergegeven tekenreeks aan.
Top	Geen	Stelt de positie van de bovenkant van het kader in.
Visible	.T.	Bepaalt of het tekstobject zichtbaar of verborgen is.
Width	Geen	Stelt de breedte in.

Beschrijving

U kunt een tekstobject gebruiken voor verschillende doeleinden, namelijk als vraag, informatie, titel, label of geheugensteuntje. Een tekstobject kan bijvoorbeeld worden gebruikt om de gebruiker te vragen waarden in te voeren in een invoervak of om aanwijzingen te geven over het gebruik van een schuifbalk.

Met het kenmerk Text geeft u de tekenreeks op die moet worden weergegeven. De weergegeven tekst is alleen-lezen. De gebruiker kan de focus niet naar een tekstobject verplaatsen en de tekst niet rechtstreeks wijzigen.

Als u een tekstobject maakt met de operator NEW, kunt u twee parameters opgeven:

- *<hoofdformulierverwijzing>* — Een objectverwijzing die naar het hoofdformulier wijst.
- *<objectnaam Tuitdr>* — Een tekenreeks die is toegewezen aan het kenmerk Name van het nieuwe tekstobject. Deze waarde is optioneel.

De volgende instructies maken bijvoorbeeld een formulier en een tekstobject om op het formulier weer te geven:

```
MijnForm = NEW FORM()
MijnTekst = NEW TEXT(MijnForm, "Tekstobject")
```

Het kenmerk Name van het nieuwe tekstobject bevat "Tekstobject".

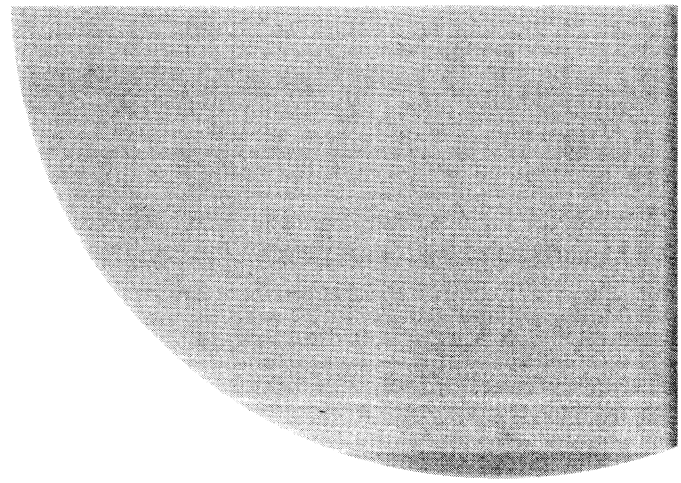
Voorbeeld

In het volgende voorbeeld worden drie tekstregels op het formulier geplaatst. De eerste maakt gebruik van een font dat is gedefinieerd met #define; de tweede en derde laten alternatieve parameteropties zien:

```
#define GROTELETTERS    FontName "Arial", FontSize 20
LOCAL f
f=NEW Demo()
f.OPEN()
CLASS Demo OF FORM
  this.Top=2
  this.Left=2
  this.Width=50
  this.Height=13
  this.Text= "Demo van klasse TEXT"
  DEFINE TEXT Tekst1 OF THIS;
    PROPERTY;
      Top 1,;
      Left 0,;
      Alignment 4,;
      Text "dBASE voor Windows",;
      Height 3,;
      Width 50,;
      GROTELETTERS,;
      ColorNormal "R/W"
  DEFINE TEXT Tekst2 OF THIS;
    PROPERTY;
      Top 5,;
      Left 0,;
      Alignment 4,;
      Text "van BORLAND",;
      Height 2,;
      Width 50,;
      FontBold .T.,;
      FontItalic .T.,;
      FontSize 16
  DEFINE TEXT Tekst3 OF THIS;
    PROPERTY;
      Top 8,;
      Left 5,;
      Alignment 4,;
      Text "is nu verkrijgbaar!",;
      Height 3,;
      Width 40,;
      FontBold .T.,;
      FontSize 22,;
      Border .T.,;
      ColorNormal "GB/W"
ENDCLASS
```

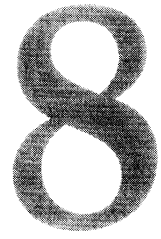
Zie ook

DEFINE



Kenmerken

Hoofdstuk



Kenmerken

ActiveControl

Het kenmerk ActiveControl bevat een verwijzing naar het object met focus.

Kenmerk uit klasse
FORM

Gegevenstype
Objectverwijzing

Gebruik
Gebruik het kenmerk ActiveControl voor verwijzingen naar het object met focus.

Een object kan op de volgende drie manieren focus krijgen:

- De gebruiker gaat met Tab naar het object.
- De gebruiker klikt op het object.
- De methode SetFocus( ) van het object wordt uitgevoerd.

Gebruik het kenmerk ActiveControl om na te gaan welk object focus heeft en op basis daarvan sprongopdrachten te geven. Met het volgende commando wordt bijvoorbeeld het kenmerk Name van het huidige object gecontroleerd:

```
? MijnForm.ActiveControl.Name
```

U kunt het kenmerk ActiveControl ook gebruiken om het kenmerk van het object met focus op te vragen of te wijzigen. Met het volgende commando wordt bijvoorbeeld het object met focus uitgeschakeld:

```
MijnForm.ActiveControl.Enabled = .F.
```

Voorbeeld

In het volgende voorbeeld worden de kenmerken ActiveControl en NextObj gebruikt om de naam van het huidige en het volgende invoervakobject op te vragen door op de rechtermuisknop in een formulier te klikken. Dit is bijvoorbeeld handig bij een formulier waarin invoervakken niet voorzien zijn van een aanduiding:

```
LOCAL f
f=NEW InvoerForm()
f.OPEN()
CLASS Invoerform OF FORM
  this.View="Bedrijf.DBF"
  this.OnRightMouseDown={;Fcs="Huidig veld: " + ;
    Form.ActiveControl.Name; Form.Txt1.Text=Fcs}
  this.OnRightMouseUp={;Fcs2="Volgend veld:     " + ;
    Form.NextObj.Name; Form.Txt2.Text=Fcs2}
DEFINE ENTRYFIELD Bedrijf OF THIS;
  PROPERTY Datalink "Bedrijf->Bedrijf",;
  Top 3, Left 1, Width 20, Name "Bedrijf"
DEFINE ENTRYFIELD Regio OF THIS;
  PROPERTY Datalink "Bedrijf->Regio",;
  Top 5, Left 1, Name "Regio"
DEFINE TEXT Txt1 OF THIS;
  PROPERTY Text " ",;
  Top 8, Left 1, Width 25
DEFINE TEXT Txt2 OF THIS;
  PROPERTY Text " ",;
  Top 9, Left 1, Width 25
ENDCLASS
```

Zie ook

_curobj, First, ID, NextObj

Add( )

Met de methode Add( ) wordt een element toegevoegd aan een ééndimensionaal array-object.

Kenmerk uit klasse

ARRAY

Gebruik

Gebruik de methode Add( ) om één element toe te voegen aan een ééndimensionaal array-object.

Bij de methode Add( ) moet de parameter <*Nuitdr*> worden opgegeven. Dit is een parameter waarmee u een waarde toewijst aan het nieuwe element. Met het volgende commando wordt bijvoorbeeld aan een array-object met 10 elementen een elfde element

toegevoegd. Aan het elfde element word met de parameter <*Nuitdr*> de waarde 100 toegewezen:

```
MijnArray = NEW ARRAY(10)
MijnArray.Add(100)
```

Zie Insert( ) voor een andere manier om elementen aan een arrray-object toe te voegen.

Voorbeeld

```
USE Klanten.DBF
* Array-object initialiseren.
ArrObj=NEW ARRAY(RECCOUNT())
* Eéndimensionale array vullen met waarden
* uit veld Naam van Klanten.DBF.
GO TOP
FOR i=1 TO RECCOUNT()
  ArrObj[i]= Klanten->Naam
  SKIP
NEXT i
* Add() gebruiken om extra element toe te voegen
* aan ééndimensionale array en naam van
* klant aan element toe te wijzen.
ArrObj.ADD(1)
Cnt=ArrObj.SIZE     && Resultaat is nieuwe aantal elementen
ArrObj[Cnt] = "Dirk Rooker Instituut"
* Inhoud array weergeven.
FOR i=1 TO Cnt
? ArrObj[i] AT 10
NEXT i
```

Zie ook

Grow( ), Insert( ) , Resize( )

Advise( )

Met de methode Advise( ) wordt aangegeven dat een wijziging in een onderdeel van een server-object in een server-toepassing moet worden doorgevoerd in dBASE.

Kenmerk uit klasse

DDELINK

Gebruik

Gebruik de methode Advise( ) om een *automatische koppeling* te maken met een onderdeel in een server-object. In geval van een automatische koppeling wordt een wijziging in een object die in de server-toepassing wordt gemaakt ook doorgevoerd in dBASE.

Een server-object kan alles zijn wat door de server-toepassing wordt ondersteund maar is meestal een document dat u in de externe toepassing hebt geopend. Met behulp van een programma voor gegevensuitwisseling kunt u bijvoorbeeld Quattro Pro voor Windows starten, een spreadsheet-bestand openen (het object) en de methode Advise( ) gebruiken om een automatische koppeling te maken met een van de cellen in de spreadsheet.

Bij de methode Advise( ) moet de parameter *<element>* worden opgegeven. Met deze parameter geeft u aan met welk onderdeel van het server-object een automatische koppeling is gemaakt. Dit onderdeel kan elk willekeurig element zijn zoals een veld in een tabel of een cel in een spreadsheet. U kunt bijvoorbeeld cel C2 op pagina A van een spreadsheet-bestand uit Quattro Pro opgeven door middel van de parameter "A:C2". Wanneer de inhoud van de cel wordt gewijzigd, wordt deze wijziging ook doorgevoerd in dBASE en wordt automatisch een subroutine gestart, die u opgeeft met het kenmerk OnNewValue.

Voorbeeld

```
PUBLIC KoppObj
KoppObj = NEW DDELINK()
KoppObj.OnNewValue = Waarde-afhandeling;
                && Codeblok of functie-aanwijzer
KoppObj.Initiate("QPW","Demo.WB1")
KoppObj.Advise("A:A1");
                && Wijzigingen in cel A:A1 worden in dBASE doorgevoerd
```

Zie ook

Execute( ), Initiate( ), OnNewValue, Peek( ), Poke( ), Server, Terminate( ), TimeOut, Topic, Unadvise( )

Alias

Met het kenmerk Alias bepaalt u welke tabel wordt weergegeven in een bladerobject.

Kenmerk uit klasse

BROWSE

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling voor het kenmerk Alias is een lege tekenreeks.

Gebruik

Gebruik het kenmerk Alias om aan te geven welke tabel in een bladerobject moet worden getoond.

Een *alias* is een alternatieve naam die aan een geopend tabelbestand is toegekend. Dit kan zijn:

- De willekeurige naam die u met de optie ALIAS van het commando USE hebt gedefinieerd.
- De bestandsnaam van de tabel (als u geen alias aan de tabel hebt toegewezen).
- De letter die overeenkomt met het werkgebied van de tabel. Deze alias is gelijk aan een van de letters A tot met J.

Als het hoofdformulier bijvoorbeeld is gebaseerd op een query waarmee twee tabellen worden geopend in een hoofdtabel-subtabel-structuur, kunt u het kenmerk Alias gebruiken om aan te geven welke tabel in het bladerobject moet worden getoond.

Zie SELECT en USE voor meer informatie over aliassen en werkgebieden.

Voorbeeld

Syntaxis operator NEW:

```
this.Alias = "Dieren"
```

Syntaxis commando DEFINE:

```
DEFINE BROWSE Blader1 OF THIS;
   PROPERTY Alias "Dieren"
```

Zie ook

DataLink, DEFINE

Alignment

Met het kenmerk Alignment wordt de positie van een grafische voorstelling in een afbeeldingsobject of van tekst in een tekstobject bepaald.

Kenmerk uit klasse

IMAGE, TEXT

Gegevenstype

Numeriek

Standaardinstelling

De standaardinstelling van Alignment is 0 (linksboven).

Gebruik

Gebruik het kenmerk Alignment als een afbeeldingsobject of een tekstobject groter is dan de afbeelding of de tekst in het object.

Als u bijvoorbeeld afbeeldingen pslaat in een binair veld en dat veld aan een afbeeldingsobject koppelt met het kenmerk DataSource, kan de gebruiker elke afbeelding weergeven door van record naar record te gaan. Als de afbeeldingen niet allemaal even groot zijn, kan het voorkomen dat de kleinere afbeeldingen niet het hele object vullen. Met het kenmerk Alignment kunt u de positie van de afbeeldingen in het object opgeven.

U kunt de volgende instellingen gebruiken voor het kenmerk Alignment van een tekstobject:

Instelling	Omschrijving
0 (linksboven)	Tegen de boven- en linkerrand
1 (middenboven)	Tegen de bovenrand, horizontaal gecentreerd
2 (rechtsboven)	Tegen de boven- en rechterrand
3 (linksmidden)	Verticaal gecentreerd en tegen de linkerrand
4 (midden)	Horizontaal en verticaal gecentreerd
5 (rechtsmidden)	Verticaal gecentreerd en tegen de rechterrand
6 (linksonder)	Tegen de onder- en linkerrand
7 (middenonder)	Horizontaal gecentreerd en tegen de onderrand
8 (rechtsonder)	Tegen de onder- en rechterrand
9 (regelovergang)	Gebruik van regelovergang bij tekenreeksen die langer zijn dan het tekstobject
0 (uitrekken)	Vergroten zodat het gehele afbeeldingsobject wordt gevuld

U kunt de volgende instellingen gebruiken voor het kenmerk Alignment van een afbeeldingsobject:

Instelling	Omschrijving
0 (uitrekken)	Vergroten zodat het gehele afbeeldingsobject wordt gevuld
1 (linksboven)	Tegen de linker- en bovenrand
2 (midden)	Gecentreerd

Opmerking Definieer de afmetingen van een tekst- of afbeeldingsobject met de kenmerken Height en Width.

Voorbeeld

Syntaxis operator NEW:

```
Kop = NEW Text(this)
  Kop.Text = "Dieren van de wereld "
  Kop.Alignment = 7
  Kop.Border = .T.
```

Syntaxis commando DEFINE:

```
DEFINE TEXT Kop OF THIS;
  PROPERTY;
    Text "Dieren van de wereld",;
    Alignment 7,;
    Border .T.
```

Zie ook

Height, Width

Append

Met het kenmerk Append wordt aangegeven of records kunnen worden toegevoegd aan een tabel in een bladerobject.

Kenmerk uit klasse

BROWSE

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Append is waar (.T.).

Gebruik

Stel het kenmerk Append in op (.F.), als u wilt voorkomen dat gebruikers records aan een tabel toevoegen. U kunt bijvoorbeeld een applicatie maken waarmee managers wel klantgegevens kunnen bekijken of zelfs wijzigen maar geen nieuwe klanten kunnen toevoegen.

Voorbeeld

Syntaxis operator NEW:

```
BedrijfBladeren = New BROWSE(this)
  BedrijfBladeren.Top = 3
  BedrijfBladeren.Left = 1
  BedrijfBladeren.Width = 60
  BedrijfBladeren.Alias = "Bedrijf"
  BedrijfBladeren.Append = .F.
  BedrijfBladeren.Delete = .F.
  BedrijfBladeren.Modify = .F.
```

Syntaxis commando DEFINE:

```
DEFINE BROWSE BedrijfBladeren OF THIS;
   FROM 3,1 TO 13,40;
     Property;
      Alias "Bedrijf",;
      Append .F. ,;
      Delete .F. ,;
      Modify .F.
```

Zie ook

Delete, Modify

AutoSize

Met het kenmerk AutoSize wordt aangegeven of de afmetingen en verhoudingen van een formulier bij het openen automatisch worden aangepast, zodat alle objecten in het formulier passen.

Kenmerk uit klasse

FORM

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van AutoSize is onwaar (.F.).

Gebruik

Gebruik het kenmerk AutoSize om te aan te geven hoe de afmetingen van een formulier worden bepaald.

Als u het kenmerk AutoSize van een formulier instelt op waar (.T.) en het formulier wordt geopend, worden de afmetingen automatisch aangepast, zodat de objecten in het formulier passen. Als u dit kenmerk op onwaar (.F.) instelt en het formulier wordt geopend, krijgt het formulier de standaardafmetingen.

De standaardafmetingen van een formulier worden als volgt bepaald:

- De instellingen die u definieert voor de kenmerken Height en Width
- De FROM...TO-clausule van het commando DEFINE FORM, als u het formulier met dat commando hebt gemaakt
- De afmetingen en de verhoudingen die de gebruiker de laatste keer met de muis of met de opdracht **Formaat wijzigen** uit het systeemmenu heeft ingesteld voor het formulier

Als u het kenmerk AutoSize van een formulier op waar instelt, worden de standaardafmetingen genegeerd. De gebruiker kan het formaat van het formulier dan nog wel wijzen en het formulier ook verplaatsen, maar als het formulier wordt gesloten en weer geopend, worden de afmetingen weer zo aangepast dat alle objecten in het formulier passen.

Voorbeeld

Syntaxis operator NEW:

```
InvoerForm = NEW Form()
  InvoerForm.Autosize = .F.
```

Syntaxis commando DEFINE:

```
DEFINE FORM InvoerForm ;
  Property;
      Autosize .F.
```

Zie ook

Height, Left, Width

Before

Met het kenmerk Before wordt aangegeven welk ander object voorafgaat aan een object in de tabvolgorde van een formulier.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Objectverwijzing

Gebruik

Gebruik het kenmerk Before om de tabvolgorde van de objecten in een formulier te bepalen.

Als een formulier twee of meer objecten bevat die een gebruiker kan selecteren, kan de gebruiker van object naar object gaan door op *Tab* of *Shift-Tab* te drukken. De volgorde waarin de focus van object naar object wordt verplaatst, wordt de *tabvolgorde* genoemd.

Als u een objectverwijzing naar een object (object A) opgeeft bij het kenmerk Before van een ander object (object B), verandert de tabvolgorde als volgt:

- Als de gebruiker op *Tab* drukt om van object naar object te gaan, komt object B voor object A.

- Als de gebruiker op *Shift-Tab* drukt om van object naar object te gaan, komt object A voor object B.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL f
f=NEW InvoerForm()
f.OPEN()
CLASS Invoerform OF FORM
  this.View="Bedrijf.DBF"
  this.Vld1 = NEW Entryfield(this)
  this.Vld1.Top = 3
  this.Vld1.Left = 1
  this.Vld1.Width=20
  this.Vld1.Datalink = "Bedrijf->Bedrijf"
  this.Vld2 = NEW Entryfield(this)
  this.Vld2.Top = 5
  this.Vld2.Left = 1
  this.Vld2.Datalink = "Bedrijf->Regio"
  this.Vld2.Before = this.Vld1
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Bedrijf OF THIS;
  Property Datalink "Bedrijf->Bedrijf",;
  Top 3, Left 1, Width 20
DEFINE ENTRYFIELD Regio OF THIS;
  Property Datalink "Bedrijf->Regio",;
  Top 5, Left 1, Before this.Bedrijf
```

Zie ook

_curobj

Border

Met het kenmerk Border wordt aangegeven of rond een object een kader wordt getekend.

Kenmerk uit klasse

EDITOR, ENTRYFIELD, IMAGE, OLE, RECTANGLE, SPINBOX, TEXT

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Border is waar (.T.).

Gebruik

U kunt voorkomen dat rond een kaderobject een rand wordt getekend door het kenmerk Border in te stellen op (.F.). Als u de rand weer wilt herstellen, stelt u het kenmerk Border in op waar.

Standaard wordt een enkele lijn rond een object getekend. Als u een verwant kenmerk met de naam BorderStyle op 1 (hoger) instelt, lijkt het object hoger te liggen dan de omgeving en als u het kenmerk BorderStyle instelt op 2 (lager), lijkt het object lager te liggen dan de omgeving.

Voorbeeld

Syntaxis operator NEW:

```
Kop = NEW TEXT(this)
  Text = "Invoerscherm"
  Top  = 1
  Left = 5
  Border = .T.
```

Syntaxis commando DEFINE:

```
DEFINE TEXT Kop OF EntryForm;
  Property;
      Text "Invoerscherm",;
      Top  0,;
      Left  5,;
      Border .T.
```

Zie ook

CLASS RECTANGLE, BorderStyle

BorderStyle

Met het kenmerk BorderStyle worden de karakteristieken van het kader rond een kaderobject gedefinieerd.

Kenmerk uit klasse

RECTANGLE

Gegevenstype

Numeriek

Standaardinstelling

De standaardinstelling van BorderStyle is 0 (normaal).

Gebruik

Gebruik het kenmerk BorderStyle om de weergave van een kaderobject te definiëren.

BorderStyle	Resultaat
0 (normaal)	Kaderobject is omkaderd door een driedimensionale lijn met een titel
1 (hoger)	Kader lijkt hoger dan omgeving
2 (lager)	Kader lijkt lager dan omgeving

Als u BorderStyle instelt op 1 of 2, verdwijnt het label dat in de linkerbovenhoek van het kader wordt weergegeven.

Voorbeeld

Syntaxis operator NEW:

```
Kdr1 = NEW RECTANGLE(this)
  Top  = 1
  Left = 5
  Width = 30
  Height = 15
  BorderStyle = 2    && Lager dan omgeving
```

Syntaxis commando DEFINE:

```
DEFINE RECTANGLE Kdr1 OF THIS;
  Property;
    Top  1,;
    Left  5,;
    Width 30,;
    Height 15,;
    BorderStyle 1    && Hoger dan omgeving
```

Zie ook

Border, CLASS RECTANGLE

Bottom

Met het kenmerk Bottom wordt ingesteld in welke rij het onderste uiteinde van een lijnobject wordt geplaatst.

Kenmerk uit klasse

LINE

Gegevenstype
Numeriek

Gebruik
Gebruik het kenmerk Bottom in combinatie met de kenmerken Right, Left en Top om de positie en de lengte van een lijnobject te definiëren.

Als maateenheid wordt bij het kenmerk Bottom de gemiddelde hoogte van de tekens gebruikt van het font dat in het hoofdformulier actief is. Als u het kenmerk Bottom van een lijnobject bijvoorbeeld instelt op 17,5, wordt het onderste uiteinde van de lijn 17,5 tekens omlaag geplaatst.

Voorbeeld
```
DEFINE LINE Ln1 OF THIS;
  PROPERTY;
    Left 10,;          && Verticale lijn van 3,10;
    Top 3,;            && tot 8,10
    Width 4,;
    Bottom 8,;
    ColorNormal "RB/W"
DEFINE LINE Ln2 OF THIS;
  PROPERTY;
    Left 3,;           && Horizontale lijn van 8,3
    Top 8,;            && tot 8,33
    Width 4,;
    Bottom 8,;
    Right 33,;
    ColorNormal "RB/W"
```

Zie ook
CLASS FORM, Right, ScaleFontName, ScaleFontSize, Top, Width

Checked

Met het kenmerk Check wordt aangegeven of een vinkje naast een menucommando wordt weergegeven.

Kenmerk uit klasse
MENU

Gegevenstype
Logisch

Standaardinstelling

De standaardinstelling van Checked is onwaar (.F.).

Gebruik

Gebruik het kenmerk Checked om aan te geven dat een voorwaarde of een proces is in- of uitgeschakeld.

Het vinkje wordt links van het menucommando getoond als u het kenmerk Checked instelt op waar (.T.). Het vinkje wordt verwijderd als u het kenmerk Checked instelt op onwaar. Een gebruiker voegt een vinkje toe of verwijdert een vinkje door op de menuopdracht te klikken.

U kunt de huidige instelling van het kenmerk Checked met een query opvragen en op basis daarvan sprongopdrachten geven. Als bijvoorbeeld het kenmerk Checked is ingesteld op waar, kunt u een bepaalde procedure of een bepaald codeblok uitvoeren en als dit kenmerk op onwaar is ingesteld een andere procedure of een ander codeblok (of helemaal geen procedure of codeblok).

Voorbeeld

Syntaxis operator NEW:

```
Bewerken = NEW MENU(this.Hoofd.View)
  Bewerken.Text = "Bewerken"
  Bewerken.Checked = .T.
```

Syntaxis commando DEFINE:

```
DEFINE MENU Bewerken OF THIS;
  PROPERTY ;
  Text "Bewerken",;
  Checked .T.
```

Zie ook

DEFINE

ClassName

Het kenmerk ClassName kan alleen worden gelezen en geeft aan op welke klasse een object is gebaseerd.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, DDELINK, DDETOPIC, EDITOR, ENTRYFIELD, FORM, IMAGE, LINE, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype
Teken

Standaardinstelling
De naam van de klasse waarop het object is gebaseerd.

Gebruik
Gebruik het kenmerk ClassName om na te gaan op basis van welke klasse een object is gemaakt. De waarde van het kenmerk ClassName wordt ingesteld als u het object maakt en kan alleen worden gelezen.

Voorbeeld
Syntaxis operator NEW:

```
this.Vld1 = NEW ENTRYFIELD(this)
  this.Vld1.OnLostFocus = Controle
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Vld1 OF THIS;
  PROPERTY OnLostFocus Controle
PROCEDURE Controle
IF This.Classname = "ENTRYFIELD"
   ? this.Value        && Inhoud naar commandovenster,
ENDIF                  && resultatenpaneel of printer
```

Zie ook
Name

Close( )

Met de methode Close wordt een formulier gesloten.

Kenmerk uit klasse
FORM

Gebruik
Gebruik Close( ) om een open formulier te sluiten.

Als u een formulier sluit, gebeurt het volgende in dBASE:

1 De subroutine of het codeblok die zijn opgegeven bij het kenmerk Valid van het huidige object, wordt uitgevoerd. Als het resultaat onwaar (.F.) is, wordt het formulier niet gesloten.

2 De subroutine of het codeblok die zijn opgegeven bij het kenmerk OnLostFocus van het huidige object, wordt uitgevoerd.

3 De subroutine of het codeblok die zijn opgegeven bij het kenmerk OnLostFocus van het formulier, wordt uitgevoerd.

4 Het formulier en de objecten in dat formulier worden niet langer op het scherm weergegeven.

5 De subroutine of het codeblok die zijn opgegeven bij het kenmerk OnClose van het formulier, wordt uitgevoerd.

6 De formulierdefinitie wordt uit het geheugen verwijderd, tenzij er objectverwijzingen naar het formulier verwijzen.

Als de formulierdefinitie niet uit het geheugen is verwijderd, kunt u het formulier weer openen met Open( ) of OPEN FORM.

Voorbeeld

Syntaxis operator NEW:

```
this.Afsluiten = NEW PUSHBUTTON(this)
  this.Top=16
  this.Left=14
  this.Width = 20
  this.Text="Afsluiten"
  this.Height=2
  this.OnClick {;Form.Close()}
```

Syntaxis commando DEFINE:

```
DEFINE PUSHBUTTON Afsluiten OF THIS;
  PROPERTY Text "Afsluiten", Height 2,;
  Width 20, Top 16, Left 14,;
  OnClick {;Form.Close()}
```

Zie ook

CLOSE FORMS, Open( ), OPEN FORM, ReadModal( ), READMODAL( )

ColorHighlight

Met het kenmerk ColorHighlight wordt de kleur van het object met focus ingesteld.

Kenmerk uit klasse

ENTRYFIELD, LISTBOX, OLE, SPINBOX

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van ColorHighlight is N/W.

Gebruik

Met het kenmerk ColorHighlight kunt u twee kleurinstellingen definiëren: een voorgrondkleur (voor tekst) en een achtergrondkleur. Kleurinstellingen worden gedefinieerd door middel van een kleurcode en de instellingen worden onderling gescheiden door een schuine streep (/). Zie ColorNormal voor een lijst met kleurcodes.

Gebruik de kenmerken ColorHighlight en ColorNormal van een object zodanig dat gebruikers het verschil kunnen zien tussen een object met focus en een object zonder focus.

Opmerking U kunt in het dialoogvenster **Kleur kiezen** een instelling opgeven voor het kenmerk ColorHighlight. In dit dialoogvenster kunt u een vooringestelde kleur selecteren maar ook een zelfgedefinieerde kleur. U opent het dialoogvenster **Kleur kiezen** door op de hulpmiddelenknop naast de optie **ColorHighlight** in het kenmerkenvenster te klikken.

Voorbeeld

Zie Datalink voor een voorbeeld van het gebruik van ColorHighlight.

Zie ook

ColorNormal, DEFINE

ColorNormal

Met het kenmerk ColorNormal wordt de kleur van een niet-geselecteerd object ingesteld.

Kenmerk uit klasse

BROWSE, CHECKBOX, EDITOR, ENTRYFIELD, FORM, LISTBOX, OLE, SPINBOX, TEXT

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van ColorNormal is N/W (zwart op een witte achtergrond).

Gebruik

Met het kenmerk ColorNormal kunt u twee kleurinstellingen opgeven: een voorgrondkleur (voor tekst) en een achtergrondkleur. U geeft kleuren op door middel van een kleurcode en instellingen moeten worden gescheiden met een schuine streep (/). Voor kleurenschermen zijn de kleurcodes als volgt:

Blauw	B	Bruin	RG
Cyaan	GB	Groen	G
Magenta	RB	Rood	R
Wit	W	Zwart	N
Geen kleur	X		

Deze instellingen werken niet bij monochrome schermen.

U kunt ook attributen voor de intensiteit (helderheid) instellen. Bij zowel monochrome schermen als kleurenschermen is de attribuutcode voor een voorgrond met hoge intensiteit +, en de attribuutcode voor een achtergrond met een hoge intensiteit *. Als u bijvoorbeeld de tekst in een invoervak helderblauw wilt weergeven op een witte achtergrond, stelt u het kenmerk ColorNormal in op B+/W.

Bij monochrome schermen is I een alternatieve attribuutcode voor een hoge intensiteit.

U kunt de kenmerken ColorNormal en ColorHighlight van een object zo gebruiken dat een gebruiker kan zien of een object wel of geen focus heeft.

Opmerking U kunt in het dialoogvenster **Kleur kiezen** een instelling opgeven voor het kenmerk ColorNormal. In dit dialoogvenster kunt u een vooringestelde kleur selecteren maar ook een door u gedefinieerde kleur. U opent het dialoogvenster **Kleur kiezen** door op de hulpmiddelenknop naast de optie **ColorNormal** in het kenmerkenvenster te klikken.

Voorbeeld

Zie Datalink voor een voorbeeld van het gebruik van ColorNormal.

Zie ook

ColorHighlight, DEFINE

Count( )

Met methode Count( ) wordt het aantal onderdelen in een keuzelijst opgevraagd.

Kenmerk uit klasse

LISTBOX

Gebruik

Gebruik de methode Count( ) als u het aantal onderdelen in een keuzelijst in runtime niet kunt voorspellen. Als u bijvoorbeeld FILE *.* opgeeft bij het kenmerk DataSource, varieert het aantal onderdelen omdat bestanden aan de standaarddirectory kunnen worden toegevoegd of uit deze directory worden verwijderd.

In een keuzelijst waarin meerdere onderdelen tegelijk geselecteerd kunnen worden, gebruikt u de methode Count( ) om te bepalen hoe vaak lussen worden uitgevoerd waarmee de keuzes worden geëvalueerd die een gebruiker maakt. U kunt bijvoorbeeld zien welke onderdelen zijn gekozen door elk onderdeel in een DO...WHILE-lus te evalueren met de methode Selected( ).

Een keuzelijst waarin meerdere onderdelen tegelijkertijd kunnen worden geselecteerd, maakt u door het kenmerk Multiple op waar (.T.) in te stellen.

Voorbeeld

In het volgende voorbeeld wordt een formulier gedefinieerd dat een keuzelijst bevat waarin de namen uit de tabel DIEREN.DBF worden opgenomen. Door het kenmerk Multiple in te stellen op .T. kan de gebruiker meer dan één onderdeel in de keuzelijst selecteren. Met het kenmerk OnRightMouseDown wordt de procedure Gevinkt aangeroepen. In deze procedure wordt gebruik gemaakt van de methoden Count( ) en Selected( ) om de geselecteerde onderdelen in het resultatenpaneel van het commandovenster weer te geven. Dit gebeurt telkens als met het kenmerk OnRightMouseDown de procedure Gevinkt wordt aangeroepen:

```
LOCAL f
f = NEW XFORM()
f.Open()

CLASS XFORM OF FORM
   this.Left =          52.80
   this.Width =          40.60
   this.Text = "Dierenwereld"
   this.HelpId = ""
   this.OnRightMouseDown = GEVINKT
   this.HelpFile = ""
   this.Top =           3.12
   this.Height =          20.00

   DEFINE LISTBOX LV1 OF THIS;
       PROPERTY;
         Left          10.00,;
         ColorNormal "N/W*",;
         Width          20.00,;
         DataSource "FIELD DIEREN->NAAM",;
         Multiple .T.,;
         ColorHighLight "W+/B",;
         Top          4.00,;
         Height          12.00

ENDCLASS

PROCEDURE Gevinkt
```

```
FOR i=1 TO Form.LV1.Count()
  ? Form.LV1.Selected(i)
NEXT i
RETURN
```

Zie ook

FOR...NEXT, LISTCOUNT( ), LISTSELECTED( ), Selected( ), Multiple

CurSel

Met het kenmerk CurSel wordt bepaald, welke prompt in de keuzelijst is geselecteerd.

Kenmerk uit klasse

LISTBOX

Gegevenstype

Numeriek

Standaardinstelling

De standaardinstelling van CurSel is 0.

Gebruik

Gebruik het kenmerk CurSel om op te geven welke prompt in de keuzelijst is geselecteerd.

Met het kenmerk CurSel kunt u een andere prompt selecteren dan de prompt die als laatste door de gebruiker is geselecteerd. In de subroutine van het kenmerk OnLostFocus kunt u bijvoorbeeld het kenmerk CurSel gebruiken om steeds de bovenste prompt te selecteren als de gebruiker de focus naar een ander object verplaatst.

Voorbeeld

Syntaxis operator NEW:

```
this.KL1 = NEW LISTBOX(this)
  this.Kl1.Top=4
  this.Kl1.Left=6
  this.Kl1.Width=20
  this.Kl1.Height=12
  this.Kl1.DataSource="FIELD Dieren->Naam"
  this.Kl1.CurSel= 3
```

Syntaxis commando DEFINE:

```
DEFINE LISTBOX KL1 OF THIS;
  PROPERTY;
  Top 4, Left 6, Width 20, Height 12,;
```

```
DataSource "FIELD Dieren->Naam",;
CurSel 3
```

Zie ook

Count( ), Selected( )

DataLink

Met het kenmerk DataLink wordt een object in een formulier aan een veld in een tabel gekoppeld.

Kenmerk uit klasse

CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, OLE, RADIOBUTTON, SPINBOX

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van DataLink is een lege tekenreeks.

Gebruik

Als u een aankruisvakje, invoervak, keuzerondje of ringveld maakt om gegevens op te kunnen vragen uit een tabel, moet u met het kenmerk DataLink een koppeling maken tussen het object en een veld in de betreffende tabel. Wanneer u deze koppeling hebt gemaakt, worden wijzigingen die via het object worden ingevoerd, automatisch in het veld ingevuld.

Als de gebruiker van record naar record gaat, wordt steeds de waarde uit het betreffende record in het object getoond. Als bijvoorbeeld een invoervak aan een veld in een tabel is gekoppeld en de gebruiker naar een ander record gaat, wordt in het invoervak de waarde getoond die in het huidige veld van de tabel staat.

Het kenmerk DataLink lijkt op het kenmerk DataSource. De gegevens die echter met het kenmerk DataLink worden getoond, kunnen worden gewijzigd terwijl de gegevens die worden weergegeven met behulp van het kenmerk DataSource alleen kunnen worden gelezen. Met het kenmerk DataSource van een keuzelijst met invoervak wordt bijvoorbeeld bepaald welke prompts (waarden die alleen kunnen worden gelezen) worden getoond. Daarentegen wordt met het kenmerk DataLink bepaald welk veld wordt gewijzigd wanneer de gebruiker gegevens invoert.

Opmerking U kunt een veld selecteren in het dialoogvenster **Veld kiezen**. U opent het dialoogvenster **Veld kiezen** door op de hulpmiddelenknop naast de optie **DataLink** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
Bedrijf = NEW ENTRYFIELD(this)
Bedrijf.Top = 3
Bedrijf.Left = 1
Bedrijf.Datalink = "Bedrijf"
Bedrijf.ColorHighlight = "B/W"
Bedrijf.ColorNormal   = "W/B"
Bedrijf.Function = "@!"
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Bedrijf OF THIS;
  AT 3,1;
  Property;
     Datalink "Bedrijf",;
     ColorHighlight "B/W",;
     ColorNormal   "W/B",;
     Function "@!"
```

Zie ook

Alias, DataSource, DEFINE, Fields

DataSource

Met het kenmerk DataSource wordt bepaald, welke gegevens in een keuzelijst, keuzelijst met invoervak of een afbeeldingsobject wordt weergegeven.

Kenmerk uit klasse

COMBOBOX, IMAGE, LISTBOX

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van DataSource is een lege tekenreeks.

Gebruik

Als u met het kenmerk DataSource een afbeelding in een afbeeldingsobject of prompts in een keuzelijst of keuzelijst met invoervak wilt weergeven, moet u geldige waarden opgeven.

Het kenmerk DataSource lijkt op het kenmerk DataLink. De gegevens die worden getoond met het kenmerk DataLink kunnen echter worden gewijzigd terwijl de

gegevens die worden weergegeven met het kenmerk DataSource alleen kunnen worden gelezen.

Voor een keuzelijst of een keuzelijst met invoervak kunt u een van de volgende vijf waarden als DataSource opgeven:

1 FILE <*bestandsnaamschema Tuitdr*> Hiermee worden de bestandsnamen in de huidige standaarddirectory gebruikt als prompts.

2 FIELD <*veldnaam*> Hiermee worden alle waarden in een bepaald veld in een tabelbestand gebruikt als prompt. Elke prompt staat voor een record en u kunt door de records bladeren door steeds een andere prompts te selecteren. Als u prompts wilt maken waarmee niet door records kan worden gebladerd, kopieert u het veld naar een array-object met COPY TO ARRAY en gebruikt u vervolgens de optie ARRAY <*array-naam*> bij het kenmerk DataSource (deze optie wordt in deze lijst beschreven).

3 STRUCTURE Hiermee worden alle veldnamen uit een tabel als prompt gebruikt.

4 ARRAY <*array-naam*> Hiermee worden elementen uit een array als prompt gebruikt.

5 TABLES Hiermee worden de namen van alle tabellen in de geopende database als prompt gebruikt. Zie OPEN DATABASE voor informatie over databases.

Als u wilt weten welke prompts in een keuzelijst zijn geselecteerd, gebruikt u LISTSELECTED( ) of Selected( ).

Opmerking U kunt bij het kenmerk DataSource in geval van een afbeeldingsobject een van de volgende drie waarden opgeven:

- RESOURCE <*bronidentificatie*> <*DLL-naam*> Hiermee wordt een DLL-bestand en een bron in dat bestand opgegeven. De parameter <*DLL-naam*> staat voor de naam van het DLL-bestand. Als dit DLL-bestand zich nog niet in het geheugen bevindt, moet u ook de extensie DLL opgeven. De parameter <*bronidentificatie*> staat voor een cijfer waarmee een bitmap-afbeelding in het DLL-bestand wordt aangeduid.
- FILENAME <*bestandsnaam*> Hiermee wordt de naam van een bestand met een bitmap-afbeelding opgegeven.
- BINARY <*binair veld*> Hiermee wordt de naam van een binair veld met bitmap-afbeeldingen opgegeven.

Voorbeeld

Syntaxis operator NEW:

```
this.Klminvoervak1 = NEW COMBOBOX(this)  && Keuzelijst;
                                            met invoervak
this.Klminvoervak11.DataLink = "Bedrijf"
this.Klminvoervak11.Value = "De Volharding BV"
    DataSource = "FIELD BEDRIJF"
* of
*   DataSource = "STRUCTURE"
* of
*   DataSource = "FILE '*.PRG'"
* of
*   DataSource = "TABLES"
```

```
* of
*   DataSource = "Array 'COMBOBOX'"
```

Syntaxis commando DEFINE:

```
DEFINE COMBOBOX KLMINVOERVAK1 OF THIS;
  PROPERTY;
    DataLink "Bedrijf",;
    Value "De Volharding BV",;
    DataSource "FIELD BEDRIJF"
* of
*   DataSource "STRUCTURE"
* of
*   DataSource "FILE '*.PRG'"
* of
*   DataSource "TABLES"
* of
*   DataSource "Array 'COMBOBOX'"
```

Zie ook

DEFINE

Default

Met het kenmerk Default wordt bepaald of een knop een standaardknop is.

Kenmerk uit klasse

PUSHBUTTON

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Default is onwaar (.F.).

Gebruik

Gebruik het kenmerk Default om van een knop de standaardknop te maken zodra de gebruiker een formulier voorlegt door op *Enter* te drukken. Als het kenmerk Default van een knop op waar is ingesteld, wordt deze knop zichtbaar gemarkeerd, om aan te geven dat dit de standaardknop is.

Als het kenmerk Default is ingesteld op waar (.T.), gebeuren er twee dingen wanneer de gebruiker op *Enter* drukt:

- De aan het kenmerk OnClick van de standaardknop toegewezen subroutine wordt uitgevoerd.

- Het kenmerk ID van de standaardknop wordt doorgegeven aan de subroutine die bij het kenmerk OnSelection van het hoofdformulier is opgegeven.

Bij de subroutine van het kenmerk OnSelection kan de waarde van het kenmerk ID worden gebruikt voor sprongopdrachten.

Als de gebruiker echter op een willekeurige knop klikt, wordt de subroutine uitgevoerd die bij het kenmerk OnClose van de betreffende knop is opgegeven. De waarde van het kenmerk ID van die knop wordt dan doorgegeven aan de subroutine van het kenmerk OnSelection, ook als het kenmerk Default van een andere knop op waar ingesteld.

Als u bij meerdere knoppen het kenmerk Default op waar instelt, wordt de laatste knop waarbij u dat doet beschouwd als de standaardknop. Als bij alle knoppen het kenmerk Default op waar is ingesteld, wordt de knop die als eerste is gemaakt, beschouwd als de standaardknop.

Voorbeeld

Syntaxis operator NEW:

```
KNOP1 = New PUSHBUTTON(this)
KNOP1.Text = "KNOP1"
KNOP1.Top = 3.00
KNOP1.Onclick = Actie1
KNOP2 = New PUSHBUTTON(this)
KNOP2.Text = "KNOP2"
KNOP2.Top = 8.00
KNOP2.Onclick = Actie2
KNOP2.Default = .T.
```

Syntaxis commando DEFINE:

```
DEFINE PUSHBUTTON KNOP1 OF THIS;
   PROPERTY;
      Text "KNOP1",;
      Top  3.00,;
      Onclick Actie1
DEFINE PUSHBUTTON KNOP2 OF THIS;
   PROPERTY;
      Text "KNOP2",;
      Top 8.00,;
      Onclick Actie2,;
      Default .T.
```

Zie ook

DEFINE, OnClick

Delete

Met het kenmerk Delete wordt bepaald of records in een bladerobject kunnen worden gemarkeerd om te worden verwijderd.

Kenmerk uit klasse
BROWSE

Gegevenstype
Logisch

Standaardinstelling
De standaardinstelling van Delete is waar (.T.).

Gebruik
Stel het kenmerk Delete in op onwaar om te voorkomen dat gebruikers een record markeren om het te verwijderen. Het is bijvoorbeeld mogelijk dat gebruikers in een bepaalde applicatie wel records mogen bekijken maar om veiligheidsredenen niet mogen verwijderen.

Opmerking Een record wordt niet uit een tabel verwijderd wanneer het wordt gemarkeerd om te worden verwijderd. Het gemarkeerde record wordt pas verwijderd als u het commando PACK uitvoert of wanneer de gebruiker **Tabel | Tabelhulpmiddelen | Records schonen** kiest.

Voorbeeld
Syntaxis operator NEW:

```
DEFINE BROWSE Blader1 OF this
  this.Blader1.Delete=.F.
  this.Blader1.Top=4
  this.Blader1.Left=3
  this.Blader1.Width=32
  this.Blader1.Height=12
```

Syntaxis commando DEFINE:

```
DEFINE BROWSE Blader1 OF THIS FROM 4,3 TO 16,35;
    PROPERTY;
      Delete .F.
```

Zie ook
Append, Modify

Delete( )

Met de methode Delete( ) wordt een element uit een ééndimensionaal array-object verwijderd of een rij of kolom uit een tweedimensionaal array-object.

Kenmerk uit klasse
ARRAY

Gebruik

Gebruik de methode Delete( ) om geselecteerde elementen uit een array te verwijderen zonder de grootte van de array te wijzigen. Als u de methode Delete( ) gebruikt, gebeurt het volgende:

- Uit een ééndimensionale array wordt een element verwijderd of uit een tweedimensionale array wordt een rij of kolom verwijderd.
- Alle resterende elementen worden naar het begin van de array verplaatst (omhoog als een rij is verwijderd of naar links als een element of kolom is verwijderd).
- Op de laatste positie(s) wordt de waarde .F. ingevoegd.

Zie Insert( ) voor informatie over elementen verwijderen en de resterende elementen naar het *einde* van de array te verplaatsen. Zie Fill( ) voor informatie over elementen vervangen zonder de overige elementen te verplaatsen. Gebruik Grow( ) of Resize( ) om de grootte van een array te wijzigen.

Voorbeeld

```
USE Dieren.DBF
* Array-object initialiseren.
ArrObj=NEW ARRAY(RECCOUNT(),3)
* Array-object vullen met waarden uit bestand Dieren.DBF
COPY TO ARRAY ArrObj FIELDS Naam, Gebied, Gewicht
* DELETE() gebruiken om gewicht te verwijderen
* uit kolom 3 en te vervangen door waarde onwaar (.F.)
ArrObj.DELETE(3,2)
* Resultaat DELETE() weergeven.
FOR i=1 TO RECCOUNT()
? ArrObj[i,1],ArrObj[i,2],ArrObj[i,3]
NEXT i
```

Zie ook

ADEL( ), Add( ), Insert( )

Dimensions

Met het kenmerk Dimensions wordt opgevraagd hoeveel dimensies een array-object heeft.

Kenmerk uit klasse

ARRAY

Gegevenstype

Numeriek

Gebruik

Gebruik het kenmerk Dimensions om na te gaan of een object ééndimensionaal of meerdimensionaal is.

De waarde van het kenmerk Dimensions kan alleen worden gelezen.

Voorbeeld

```
USE Klanten.DBF
* Array-object initialiseren.
ArrObj=NEW ARRAY(RECCOUNT())
* Eéndimensionale array vullen met waarden
* uit veld Naam in Klanten.DBF
GO TOP
FOR i=1 TO RECCOUNT()
  ArrObj[i] = Klanten->Naam
  SKIP
NEXT i
* Dimensions gebruiken om aantal dimensies van array
* op te vragen en GROW() om tweede kolom aan bestaande array
* toe te voegen en vervolgens waarden uit veld Verktotnu
* in tweede kolom in te vullen.
IF ArrObj.Dimensions=1
  ArrObj.GROW(2)
  GO TOP
  FOR i=1 TO RECCOUNT()
    ArrObj[i,2]=Klanten->Verktotnu
    SKIP
  NEXT i
ENDIF
* Inhoud van tweedimensionale array tonen.
FOR i=1 TO RECCOUNT()
? ArrObj[i,1], ArrObj[i,2]
NEXT i
```

Zie ook

Add( ), ALEN( ), Delete( ), Fields( ), Insert( ), Resize( )

Dir( )

Met de methode Dir( ) worden de naam, de grootte, de datum, de tijd en de DOS-attributen van bestanden in een array-object opgeslagen.

Kenmerk uit klasse

ARRAY

Gebruik

Gebruik de methode Dir( ) om informatie over een bestand in een array-object op te slaan. De methode Dir( ) lijkt op de functie ADIR( ).

Bij de methode Dir( ) kunt u een van de volgende twee optionele parameters worden opgegeven:

- *<bestandsnaamschema Tuitdr>* Hiermee geeft u bestandsnamen met een jokerteken op. Als u bijvoorbeeld alleen informatie wilt over tabellen geeft u voor de parameter *<bestandsnaamschema Tuitdr>* "*.DBF" op. Als u de parameter *<bestandsnaamschema Tuitdr>* weglaat, wordt met de methode Dir( ) informatie opgeslagen over alle bestanden in de huidige directory, behalve verborgen - en systeembestanden.
- *<bestandskenmerkenlijst Tuitdr>* Hiermee geeft u een of meer DOS-bestandsattributen op. Hieronder vindt u een omschrijving van elk attribuut.

Teken	Omschrijving
D	Inclusief directories
H	Inclusief verborgen bestanden
S	Inclusief systeembestanden
V	Alleen volumelabel

Als u bij de parameter *<bestandskenmerkenlijst Tuitdr>* meer dan een letter opgeeft, moet u alle letters tussen aanhalingstekens plaatsen. Als u een waarde opgeeft voor de parameter *<bestandskenmerkenlijst Tuitdr>*, moet u ook een waarde opgeven voor de parameter *<bestandsnaamschema Tuitdr>*.

Als u de letter V opgeeft bij de parameter *<bestandskenmerkenlijst Tuitdr>*, worden bij de methode Dir( ) de parameter *<bestandsnaamschema Tuitdr>* en alle andere tekens in de attributenlijst genegeerd en wordt het volumelabel opgeslagen als een ééndimensionaal array-object.

De grootte van het array-object wordt dynamisch aangepast aan de hoeveelheid informatie.

Voorbeeld

In het volgende voorbeeld wordt Array( ) gebruikt om een array-object te initialiseren en DIR( ) om de array te vullen met informatie over bestanden in een directory onder DOS. Met de lus FOR...NEXT wordt de inhoud van de array gelezen en wordt het resultaat van de methode Dir( ) getoond. U ziet dat Size( ) wordt gebruikt om het aantal elementen in de array te bepalen. Door het aantal elementen te delen door het aantal bestandsattributen dat met Dir( ) is gekopieerd, wordt het aantal rijen in de array berekend.

```
ArrObj=NEW ARRAY()
ArrObj.DIR("*.PRG","D")
FOR i=1 TO ArrObj.Size/5
? ArrObj[i,1],ArrObj[i,2] AT 20,;
  ArrObj[i,3] AT 35,ArrObj[i,4] AT 45,;
  ArrObj[i,5] AT 55
NEXT i
```

Zie ook
ADIR( ), Fields( )

DisabledBitmap

Met het kenmerk DisabledBitmap wordt de grafische afbeelding opgegeven die moet worden getoond op een knop als deze knop is uitgeschakeld.

Kenmerk uit klasse
PUSHBUTTON

Gegevenstype
Teken

Standaardinstelling
De standaardinstelling van DisabledBitmap is een lege tekenreeks.

Gebruik
Gebruik het kenmerk DisabledBitmap om zichtbaar te maken dat een knop niet beschikbaar is.

Een knop is in de volgende gevallen uitgeschakeld:

- Het kenmerk Enabled is op onwaar (.F.) ingesteld.
- Het kenmerk When resulteert in onwaar.

Het kenmerk DisabledBitmap kan op een van de volgende twee manieren worden ingesteld:

1 RESOURCE <*bronidentificatie*> <*DLL-naam*> Hiermee wordt de bron van de bitmap opgegeven evenals het DLL-bestand waarin de bitmap staat (een DLL-bestand is een vooraf gecompileerde bibliotheek met externe routines en bronnen die niet in de dBASE-taal zijn geschreven, maar bijvoorbeeld in C of Pascal). U kunt de bronidentificatie opvragen met een resource-editor zoals Resource Workshop.

2 FILENAME <*bestandsnaam*> Hiermee geeft u een bitmap-bestand op (een dergelijk bestand heeft meestal de extensie .BMP).

Als u een tekenreeks opgeeft met het kenmerk Text en een afbeelding met het kenmerk DisabledBitmap, worden zowel de afbeelding als de tekst getoond wanneer de knop is uitgeschakeld. Als u geen afbeelding opgeeft met DisabledBitmap, wordt de bij het kenmerk Text gedefinieerde tekenreeks met een lage intensiteit weergegeven wanneer de knop is uitgeschakeld.

Opmerking U kunt een bitmap-bestand selecteren in het dialoogvenster **Bitmap kiezen**. U opent het dialoogvenster **Bitmap kiezen** door op de hulpmiddelenknop naast de optie **DisabledBitmap** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
Start = NEW PUSHBUTTON(this)
Start.DisabledBitmap = "RESOURCE 20 EntryIcn.dll"
* of
* Start.DisabledBitmap = "FILENAME PushDis.bmp"
* of
* Start.DisabledBitmap = "BINARY PushDis"
```

Syntaxis commando DEFINE:

```
DEFINE PUSHBUTTON Start OF THIS;
  Property;
    DisableBitmap "RESOURCE 20 EntryIcn.dll"
* of
*   DisableBitmap "FILENAME PushDis.bmp"
* of
*   DisableBitmap "BINARY PushDis"
```

Zie ook

DEFINE, DownBitmap, Enabled, FocusBitmap, UpBitmap

DoVerb( )

Met DoVerb wordt een handeling gestart in een OLE server-toepassing.

Kenmerk uit klasse

OLE

Gebruik

Gebruik DoVerb( ) om een handeling te starten vanuit een OLE-document dat in een OLE-veld is opgeslagen en om aan te geven welke handeling moet worden gestart.

Bij elke OLE-object kunnen een of meer werkwoorden worden gebruikt. Elk werkwoord bepaalt welke handelingen worden verricht en elk werkwoord wordt aangegeven met een cijfer. Bij de meeste geluidstoepassingen kan bijvoorbeeld een van de volgende twee werkwoorden worden gebruikt:

- 0 (afspelen) Hiermee wordt een geluid afgespeeld dat in een OLE-veld is opgeslagen.
- 1 (bewerken) Hiermee wordt Geluidsrecorder geopend zodat het geluid kan worden bewerkt.

Sommige OLE-documenten hebben maar één werkwoord. Quattro Pro voor Windows kent bijvoorbeeld alleen 0 (bewerken).

Bij DoVerb kunnen de volgende twee parameters worden opgegeven:

- *<OLE-werkwoord>* De numerieke waarde van het werkwoord.
- *<titel>* Een tekenreeks die wordt weergegeven in de titelbalk van het server-venster (als via de parameter *<OLE-werkwoord>* een venster wordt geopend).

Opmerking U kunt als volgt bepalen welke werkwoorden in een OLE-documenten kunnen worden gebruikt:

1. In een bladerobject dubbelklikt u op het OLE-veld met het document. Hierdoor wordt de OLE-weergave geopend.
2. Open het menu **Bewerken**. De laatste menu-opdracht is het OLE-object. Als het object bijvoorbeeld is gemaakt met Geluidsrecorder van Windows, is de laatste menu-opdracht **Geluidsobject**.
3. Klik één keer op deze menu-opdracht. Als bij het OLE-document meerdere werkwoorden kunnen worden gebruikt, worden deze in een submenu als opdrachten weergegeven. Als maar één werkwoord kan worden gebruikt bij het document, wordt het betreffende werkwoord uitgevoerd.

U kunt OLE-werkwoorden ook bekijken via de registratiedatabase van Windows. Deze kunt u openen met de Registratie info-editor. U start deze editor door een Windows-sessie te starten in *'verbose mode'* en de parameter *regedit* op te geven:

```
win regedit /v
```

Voorbeeld

Het volgende programma maakt een formulier met een OLE-object en een knop. De subroutine OnClick van de knop gebruikt DoVerb om een handeling te starten in een OLE-server.

```
LOCAL f
f = NEW XFORM()
f.Open()

CLASS XFORM OF FORM
   this.Left =         22.40
   this.Width =        103.60
   this.MousePointer =          1
   this.Text = "Formulier"
   this.HelpId = ""
   this.HelpFile = ""
   this.View = "AFBEELD.QBE"
   this.Top =          1.06
   this.Height =         24.53

   DEFINE OLE OLE1 OF THIS;
       PROPERTY;
         Left          2.00,;
         Width       100.00,;
         DataLink "AFBEELD->BITMAPOLE",;
```

```
      Border .T.,;
      Top          1.00,;
      Height         20.00

  DEFINE PUSHBUTTON KNOP1 OF THIS;
      PROPERTY;
      Left         39.00,;
      ColorNormal "N/W",;
      Width         24.00,;
      OnClick CLASS::KNOP1_ONCLICK,;
      Text "Klik hier om afbeelding te bewerken",;
      Default .T.,;
      Top         22.00,;
      Height          2.00

  Procedure KNOP1_OnClick
  form.OLE1.DoVerb(0, "Mijn formulier ")

ENDCLASS
```

Zie ook

CLASS OLE

DownBitmap

Met het kenmerk DownBitmap wordt aangegeven, welke grafische afbeelding op een knop moet worden weergegeven als de gebruiker op de knop klikt.

Kenmerk uit klasse

PUSHBUTTON

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van DownBitmap is een lege tekenreeks.

Gebruik

Gebruik het kenmerk DownBitmap om zichtbaar te bevestigen dat de gebruiker op een knop heeft geklikt. Wanneer de gebruiker de muisknop loslaat of de aanwijzer naar een positie naast de knop verplaatst, wordt de afbeelding of tekst weergegeven die bij het kenmerk UpBitmap of Text is opgegeven.

Het kenmerk DownBitmap kan op een van de volgende twee manieren worden ingesteld:

1 RESOURCE <*bronidentificatie*> <*DLL-naam*> Hiermee wordt de bron van de bitmap opgegeven, evenals het DLL-bestand waarin de bitmap staat (een DLL-bestand is een vooraf gecompileerde bibliotheek met externe routines en bronnen die niet in de dBASE-taal is geschreven, maar bijvoorbeeld in C of Pascal). U kunt de bronidentificatie opvragen met een resource-editor zoals Resource Workshop.

2 FILENAME <*bestandsnaam*> Hiermee geeft u een bitmap-bestand op (een dergelijk bestand heeft meestal de extensie .BMP).

Als u een tekenreeks opgeeft met het kenmerk Text en een afbeelding met het kenmerk DownBitmap, worden zowel de afbeelding als de tekst getoond wanneer de gebruiker op de knop klikt.

Opmerking U kunt een bitmap-bestand selecteren in het dialoogvenster **Bitmap kiezen**. U opent het dialoogvenster **Bitmap kiezen** door op de hulpmiddelenknop naast de optie **DownBitmap** in het kenmerkenvenster te klikken.

Voorbeeld

Zie UpBitmap voor een voorbeeld van het gebruik van DownBitmap.

Zie ook

DEFINE, DisabledBitmap, Enabled, FocusBitmap, Text, UpBitmap

Element( )

Het resultaat van de methode Element( ) is het nummer van een bepaald element in een array-object.

Kenmerk uit klasse

ARRAY

Gebruik

Gebruik de methode Element( ) als u de indextekens van een element in een tweedimensionaal array-object kent en het nummer van het element nodig hebt voor een andere methode, bijvoorbeeld voor Scan( ). De methode Element( ) lijkt op de functie AELEMENT( ).

Een elementnummer is een volgnummer dat de positie van een element in de array aangeeft. In ééndimensionale array's is het elementnummer gelijk aan het indexteken en hoeft u de methode Element( ) niet te gebruiken. Het resultaat van Element(3) is bijvoorbeeld 3 en van Element(5) 5 enzovoorts.

Bij de methode Element( ) kunt u de volgende twee parameters opgeven:

- <*indexteken1 Nuitdr*> Hiermee geeft u het eerste indexteken aan van het element. In een ééndimensionale array is de parameter <*indexteken1 Nuitdr*> gelijk aan het

elementnummer. In een tweedimensionale array geeft de parameter *<indexteken1 Nuitdr>* een rij aan.

- *<indexteken2 Nuitdr>* Hiermee verkrijgt u de kolom waarin het element in een tweedimensionale array staat. Als u de parameter *<indexteken2 Nuitdr>* niet opgeeft, wordt uitgegaan van de waarde 1. Als het array-object ééndimensionaal is, wordt de parameter *<indexteken2 Nuitdr>* genegeerd.

De methode Element( ) is tegengesteld aan Subscript( ) waarbij u het elementnummer opgeeft en het resultaat bestaat uit het kolom- of rij-indexteken van een element.

Voorbeeld

```
USE Dieren.DBF
* Array-object initialiseren
ArrObj=NEW ARRAY(RECCOUNT(),3)
* Array-object vullen met waarden uit Dieren.DBF
COPY TO ARRAY ArrObj FIELDS Naam, Gebied, Gewicht
* ELEMENT() gebruiken om elementnummer te verkrijgen van alle waarden
* uit veld Naam die in eerste kolom van array staan
FOR i=1 TO RECCOUNT()
? ArrObj[i,1], ArrObj.Element(i,1)
NEXT i
```

Zie ook

AELEMENT( ), Scan( ), Subscript( )

Enabled

Met het kenmerk Enabled wordt bepaald of een object kan worden geselecteerd.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, SCROLLBAR, SPINBOX

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Enabled is waar (.T.).

Gebruik

Als u het kenmerk Enabled van een object op waar instelt, kan de gebruiker dat object selecteren en gebruiken. Als u het kenmerk op onwaar (.F.) instelt, wordt het object grijs weergegeven en kan de gebruiker het object niet selecteren of gebruiken.

Gewoonlijk stelt u het kenmerk Enabled in op onwaar (.F.) als niet wordt voldaan aan een bepaalde voorwaarde. Met de subroutine van het kenmerk Valid van een invoervak kan bijvoorbeeld een OK-knop worden uitgeschakeld als de gebruiker geen of onjuiste gegevens in een invoervak heeft ingevoerd.

Voorbeeld

Syntaxis operator NEW:

```
CompCode = NEW ENTRYFIELD(this)
CompCode.Top = 3
CompCode.Left = 1
CompCode.Datalink = "CompCode"
CompCode.Enabled = .F.
* Gebruiker kan CompCode niet wijzigen
```

Syntaxis commando DEFINE:

```
DEFINE Entryfield CompCode OF THIS;
  AT 3,1;
  Property;
     Datalink "CompCode",;
     Enabled .F.
```

Zie ook

Visible

EscExit

Met het kenmerk EscExit wordt bepaald of een gebruiker een formulier kan sluiten door op *Esc* te drukken.

Kenmerk uit klasse

FORM

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van EscExit is waar (.T.).

Gebruik

Stel het kenmerk EscExit in op onwaar (.F.) als u niet wilt dat een gebruiker een formulier kan sluiten door op *Esc* te drukken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL f
f=NEW InvoerForm()
f.OPEN()
CLASS InvoerForm OF FORM
  this.Top=2
  this.Left=2
  this.Height=10
  this.Width = 20
  this.EscExit=.F.
```

Syntaxis commando DEFINE:

```
DEFINE FORM InvoerForm ;
  PROPERTY;
  Top 2, Left 2, Height 10, Width 20,;
  EscExit .F.
```

Zie ook

ON ESCAPE, SET ESCAPE

Execute( )

Met de methode Execute( ) wordt een commandoreeks naar een DDE server-toepassing gestuurd in de taal van de betreffende toepassing.

Kenmerk uit klasse

DDELINK

Gebruik

Gebruik de methode Execute( ) om macro-instructies naar een server-toepassing te sturen.

U moet bij de methode Execute( ) de parameter <*commando*> opgeven. Deze parameter bestaat uit een reeks met ten minste één commando in de taal van de server-toepassing. Plaats elk commando tussen de scheidingstekens die door de server-toepassing worden vereist. Elk commando in Quattro Pro moet bijvoorbeeld tussen accolades ({}) staan. Bij sommige toepassingen kunnen meerdere commando's worden opgegeven die dan tussen vierkante haakjes moeten staan. Raadpleeg de documentatie bij de DDE-server voor meer informatie.

Voordat u een commandoreeks naar een server stuurt, moet u eerst de server-toepassing starten, het gewenste document opvragen en een DDE-koppeling maken. Zie Initiate( ) en Hoofdstuk 26 in *Programmeren* voor meer informatie over DDE-koppelingen maken.

Voorbeeld

```
PUBLIC KoppObj
KoppObj = NEW DDELINK()
KoppObj.Initiate("QPW","Demo.WB1")
? KoppObj.Execute('{OPEN "C:\INFO\Info.TXT",W}');
            && Tekstbestand maken met naam Info.TXT.
? KoppObj.Execute('{WRITELN +A1}');
            && Inhoud van cel A1 in QPW naar
            && Info.TXT sturen.
? KoppObj.Execute('{WRITELN +A2}');
            && Inhoud van cel A2 in QPW naar
            && Info.TXT sturen.
? KoppObj.Execute('{CLOSE}');
            && Bestand Info.TXT sluiten.
```

Zie ook

Advise( ), Initiate( ), OnNewValue, Peek( ), Poke( ), Server, Terminate( ), TimeOut, Topic, Unadvise( )

Fields

Met het kenmerk Fields worden de velden opgegeven die in een bladerobject moeten worden weergegeven, evenals de veldopties die voor elk veld gelden.

Kenmerk uit klasse

BROWSE

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van Fields is een lege tekenreeks.

Gebruik

De waarde die u aan het kenmerk Fields toekent moet tussen aanhalingstekens staan en op de volgende manier worden opgegeven:

"*<veld1>* [*<veldoptieslijst1>*] |
<rekenveld1> = *<uitdr1>* [*<rekenveldoptieslijst1>*]
[, *<veld2>* [*<veldoptieslijst2>*] |
<rekenveld2> = *<uitdr1>* [*<rekenveldoptieslijst2>*], ...]"

De velden worden in de aangegeven volgorde weergegeven. Opties voor veldenlijsten zijn van invloed op de manier waarop de velden worden getoond. Hieronder ziet u een beschrijving van deze opties:

/<*kolombreedte kolom*>	De breedte van de kolom waarin <veld1> wordt getoond, als <veld1> een tekentype is.
/B = <*uitdr1*>, <*uitdr2*> [/F]	RANGE-optie; de waarde die in <veld1> is ingevoerd moet in het bereik vallen van <uitdr1> tot en met <uitdr2>.
	RANGE REQUIRED-optie; met de optie /F voorkomt dat een eerder ingevoerde waarde wordt geaccepteerd wanneer deze niet in het bereik valt van <uitdr1> tot en met <uitdr2>.
/H = <*Tuitdr*>	HEADER-optie; in plaats van de veldnaam wordt <Tuitdr> boven de veldkolom in het tabelrecordsvenster weergegeven.
/P = <*Tuitdr*>	PICTURE-optie; <veld1> wordt overeenkomstig de PICTURE- of FUNCTION-clausule <Tuitdr> weergegeven.
/R	READ-ONLY-optie; geeft aan dat <veld1> alleen kan worden gelezen en niet kan worden bewerkt.
/V = <*voorwaarde*> [/F] [/E = <*Tuitdr*>]	VALID-optie; er kan alleen een nieuwe waarde worden ingevoerd voor <veld1> als <voorwaarde> resulteert in .T..
	VALID REQUIRED-optie; met de optie /F wordt voorkomen dat de gebruiker <veld1> verlaat en de bewerksessie afsluit als het resultaat van <voorwaarde> niet .T. is.
	ERROR MESSAGE-optie; als /E = <Tuitdr> is opgegeven, verschijnt <Tuitdr> als het resultaat van <voorwaarde>.F. is.
/W = <*voorwaarde*>	WHEN-optie; veld <veld1> kan alleen worden bewerkt als het resultaat van <voorwaarde>.T. is.

Rekenvelden die alleen kunnen worden gelezen, bestaan uit een toegewezen veldnaam en een uitdrukking die resulteert in de waarde van het rekenveld, bijvoorbeeld: *Commissie = Tarief * Verkoopprijs*. Opties voor rekenvelden zijn van invloed op de manier waarop deze velden worden weergegeven. Hieronder ziet u een beschrijving van deze opties:

/<*kolombreedte kolom*>	De breedte van de kolom waarin <veld1> wordt getoond.
/H = <*Tuitdr*>	De parameter <Tuitdr> wordt boven de rekenveldkolom in het tabelrecordvenster weergegeven, in plaats van de naam van het rekenveld.

Berekende velden kunnen alleen worden gelezen.

Opmerking U kunt een veld selecteren in het dialoogvenster **Bladerveld kiezen**. U opent dit dialoogvenster door op de hulpmiddelenknop naast de optie **Fields** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
this.Blader1 = NEW BROWSE(this)
  this.Blader1.Alias="Contact"
  this.Blader1.Fields="CompCode /R,Contact"
```

```
this.Blader1.Top=4
this.Blader1.Left=3
this.Blader1.Width=32
this.Blader1.Height=12
```

Syntaxis commando DEFINE:

```
DEFINE BROWSE Blader1 OF THIS;
  PROPERTY;
  Top 4, Left 3, Width 32, Height 12,;
  Alias "Contact",;
  Fields "CompCode /R, Contact"
```

Zie ook

Alias, DataLink, DataSource, SET FIELDS

Fields( )

Met de methode Fields( ) wordt informatie over de structuur van de huidige tabel in een array-object opgeslagen. Het resultaat van deze methode is het aantal velden waarvan de karakteristieken zijn opgeslagen.

Kenmerk uit klasse

ARRAY

Gebruik

Gebruik de methode Fields( ) om informatie over de structuur van de huidige tabel in een array-object op te slaan. Elke rij in het array-object bevat informatie over één veld uit in de huidige tabel. De methode Fields( ) lijkt op de functie AFIELDS( ).

Bij de methode Fields( ) wordt de grootte van het array-object dynamisch aangepast zodat het aantal rijen ten minste even groot is als het aantal velden in de huidige tabel. Het aantal kolommen is minimaal vier. Als de array meer elementen heeft dan nodig zijn om de tabelstructuur op te slaan, blijven de extra elementen ongewijzigd.

In de volgende tabel ziet u welke veldkarakteristieken met de methode Fields( ) worden opgeslagen en in welke kolom de informatie wordt geplaatst:

Kolom 1	Kolom 2	Kolom 3	Kolom 4
Veldnaam (gegevenstype: teken)	Veldtype (gegevenstype: teken)	Veldlengte (gegevenstype: numeriek)	Decimale plaatsen (gegevenstype: numeriek)

In dBASE worden de volgende codes gebruikt voor de verschillende veldtypen: C—teken, D—datum, L—logisch, M—memo, N—numeriek, F—zwevend, G—OLE, B—binair.

Met de methode Fields( ) slaat u dezelfde informatie in een array-object op als u met COPY TO...STRUCTURE EXTENDED in een tabel opslaat. Bij de methode Fields( ) wordt echter geen rij met FIELD_IDX-informatie gemaakt.

Voorbeeld

```
USE Klanten.DBF
* Array-object initialiseren waarin
* structuur van Klanten.DBF wordt opgeslagen.
ArrObj=NEW ARRAY(FLDCOUNT(),4)
* Array vullen met informatie over structuur
* van .DBF-bestand.
ArrObj.Fields()
* Inhoud array tonen in resultatenpaneel
* van commandovenster.
FOR i=1 TO FLDCOUNT()
? ArrObj[i,1],ArrObj[i,2] AT 20,;
  ArrObj[i,3] AT 25, ArrObj[i,4] AT 35
NEXT i
```

Zie ook

AFIELDS( ), COPY TO...STRUCTURE EXTENDED , Dir( ), Element( ), Scan( ), Sort( ), Subscript( )

FieldWidth

Met het kenmerk FieldWidth wordt de breedte van een tekenveld in een bladerobject gedefinieerd.

Kenmerk uit klasse

BROWSE

Gegevenstype

Numeriek

Standaardinstelling

De standaardinstelling van FieldWidth is de lengte van een veld in een tabel of de lengte van de veldnaam, maar altijd de langste van de twee.

Gebruik

Gebruik het kenmerk FieldWidth om de weergegeven lengte van een tekenveld in een bladerobject te beperken.

Als de tekst in een tekenveld langer is dan het veld zelf, kan de gebruiker door de inhoud van het veld bladeren.

Voorbeeld

Syntaxis operator NEW:

```
USE Dieren.DBF
LOCAL f
f=NEW Blader()
f.OPEN()
CLASS Blader OF FORM
  this.Width = 45
  this.Blader1=NEW BROWSE(this)
  this.Blader1.Top=2
  this.Blader1.Left=1
  this.Blader1.Width=40
  this.Blader1.Height=10
  this.Blader1.FieldWidth=10
ENDCLASS
```

Syntaxis commando DEFINE:

```
* Formulierdefinitie
DEFINE BROWSE Blader1 OF THIS;
  PROPERTY;
  Top 2, Left 1, Width 40,;
  Height 10, FieldWidth 10
```

Zie ook

ShowHeading, ShowRecNo

Fill( )

Met de methode Fill( ) wordt een bepaalde waarde op een of meer locaties in een array-object ingevoegd.

Kenmerk uit klasse

ARRAY

Gebruik

Gebruik de methode Fill( ) om een waarde toe te wijzen aan alle of enkele elementen in een array-object. De methode Fill( ) lijkt op de functie AFILL( ).

U kunt de volgende drie parameters opgeven bij de methode Fill( ):

- *<uitdr>* Dit is de waarde die u in het array-object wilt invoegen.
- *<begin Nuitdr>* Dit is het elementnummer van waaraf u *<uitdr>* wilt gaan invoegen (als u alleen de indextekens van het element weet, kunt u met de methode Element( )

het elementnummer opvragen). Als u de parameter *<begin Nuitdr>* niet gebruikt, wordt bij het eerste element van het array-object begonnen.

- *<aantal Nuitdr>* Dit is het aantal elementen waaraan *<uitdr>* moet worden toegewezen, te beginnen bij element *<begin Nuitdr>*. Als u de parameter *<aantal Nuitdr>* niet gebruikt, wordt bij de methode Fill( ) *<uitdr>* toegewezen vanaf *<begin Nuitdr>* tot het laatste element van de array. Als u een waarde opgeeft voor de parameter *<aantal Nuitdr>*, moet u ook een waarde opgeven voor de parameter *<begin Nuitdr>*.

Als een van de parameters *<begin Nuitdr> of <aantal Nuitdr>* niet opgeeft, wordt bij de methode Fill( ) *<uitdr>* toegewezen aan alle elementen van de array.

Voorbeeld

```
Use Dieren.Dbf
* Array-object initialiseren.
ArrObj=NEW ARRAY(RECCOUNT(),3)
* Tekenreeks "Test" aan alle elementen toewijzen en
* inhoud tonen in resultatenpaneel van commandovenster.
ArrObj.Fill("Test")
FOR i=1 TO RECCOUNT()
? ArrObj[i,1],ArrObj[i,2], ArrObj[i,3]
NEXT i
```

Zie ook

AFILL( ), Dir( ), Element( ), Fields( ), Subscript( )

First

Met het kenmerk First wordt aangegeven welke object als eerste focus krijgt wanneer het bijbehorende hoofdformulier wordt geopend.

Kenmerk uit klasse

FORM

Gegevenstype

Objectverwijzing

Gebruik

Gebruik het kenmerk First om aan te geven welk object in eerste instantie focus krijgt wanneer u een formulier opent. De instelling van het kenmerk First kan alleen worden gelezen.

Als een formulier twee of meer objecten bevat die de gebruiker kan selecteren, kan de gebruiker van object naar object gaan door op *Tab* of *Shift-Tab* te drukken. De volgorde

waarin de focus van object naar object verspringt wordt de *tabvolgorde* genoemd. De objectverwijzingsvariabele in het kenmerk First verwijst naar het object waarmee de tabvolgorde begint.

Zie Hoofdstuk 10 in *Programmeren* voor meer informatie over objectverwijzingsvariabelen.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL f
f=NEW InvoerForm()
f.OPEN()
CLASS InvoerForm OF FORM
  this.view="Bedrijf.DBF"
  this.Vld1 = NEW Entryfield(this)
  this.Vld1.Top = 3
  this.Vld1.Left = 1
  this.Vld1.Width=20
  this.Vld1.Datalink = "Bedrijf->Bedrijf"
  this.Vld2 = NEW Entryfield(this)
  this.Vld2.Top = 5
  this.Vld2.Left = 1
  this.Vld2.Datalink = "Bedrijf->Regio"
  this.Vld2.Before = this.First
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE Entryfield Bedrijf OF THIS;
  Property Datalink "Bedrijf->Bedrijf",;
  WIDTH 20, Top 3, Left 1
DEFINE Entryfield Regio OF THIS;
  Property Datalink "Bedrijf->Regio",;
  Before this.First, Top 5, Left 1
```

Zie ook

ActiveControl, _curobj, BEFORE, NextObj, SetFocus( )

FocusBitmap

Met het kenmerk FocusBitmap wordt de grafische afbeelding opgegeven die op een knop moet worden weergegeven wanneer deze knop focus heeft.

Kenmerk uit klasse

PUSHBUTTON

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van FocusBitmap is een lege tekenreeks.

Gebruik

Gebruik het kenmerk FocusBitmap om zichtbaar te maken dat een knop is geselecteerd.

Het kenmerk FocusBitmap kan op een van de volgende twee manieren worden ingesteld:

1 RESOURCE <*bronidentificatie*> <*DLL-naam*> Hiermee wordt de bron van de bitmap opgegeven, evenals het DLL-bestand waarin de bitmap staat (een DLL-bestand is een vooraf gecompileerde bibliotheek met externe routines en bronnen die niet in de dBASE-taal is geschreven, maar bijvoorbeeld in C of Pascal). U kunt de bronidentificatie opvragen met een resource-editor zoals Resource Workshop.

2 FILENAME <*bestandsnaam*> Hiermee geeft u een bitmap-bestand op (een dergelijk bestand heeft meestal de extensie .BMP).

Als u een tekenreeks opgeeft met het kenmerk Text en een afbeelding met het kenmerk FocusBitmap, worden zowel de afbeelding als de tekst getoond wanneer de knop focus heeft.

Opmerking U kunt een bitmap-bestand selecteren in het dialoogvenster **Bitmap kiezen**. U opent het dialoogvenster **Bitmap kiezen** door op de hulpmiddelenknop naast de optie **FocusBitmap** in het kenmerkenvenster te klikken.

Voorbeeld

Zie DownBitmap voor een voorbeeld van het gebruik van FocusBitmap.

Zie ook

DEFINE, DisabledBitmap, DownBitmap, Enabled, UpBitmap

Follow

Met het kenmerk Follow wordt aangegeven of een record in een bladerobject zichtbaar blijft als de waarde in een sleutelveld wordt gewijzigd en de indexvolgorde van de records verandert.

Kenmerk uit klasse

BROWSE

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Follow is waar (.T.).

Gebruik

Stel het kenmerk Follow in op waar (.T.) om te voorkomen dat een record in een bladerobject lijkt te verdwijnen als de records in de volgorde van de index staan en de gebruiker een waarde in een sleutelveld wijzigt.

Als records in de volgorde van de index staan en de waarde in een sleutelveld van een record wordt gewijzigd, verandert de positie van dat record in de tabel. In een bladerobject is het daarom mogelijk dat het gewijzigde record niet langer wordt weergegeven. Door het kenmerk Follow in te stellen op waar, wordt dit voorkomen.

Voorbeeld

Syntaxis operator NEW:

```
this.Blader1 = NEW BROWSE(this)
  this.Blader1.Alias="Contact"
  this.Blader1.Fields="CompCode,Contact"
  this.Follow=.F.
  this.Blader1.Top=4
  this.Blader1.Left=3
  this.Blader1.Width=32
  this.Blader1.Height=12
```

Syntaxis commando DEFINE:

```
DEFINE BROWSE Blader1 OF THIS;
  PROPERTY;
  Top 4, Left 3, Width 32, Height 12,;
  Alias "Contact", Follow .F.,;
  Fields "CompCode, Contact"
```

Zie ook

Modify

FontBold

Met het kenmerk FontBold wordt aangegeven of bij een object tekens vet worden weergegeven.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, LISTBOX, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SPINBOX, TEXT

Gegevenstype
Logisch

Standaardinstelling
De standaardinstelling van FontBold is onwaar (.F.) bij alle klassen behalve EDITOR, ENTRYFIELD, PUSHBUTTON, RADIOBUTTON, SPINBOX en TEXT.

Gebruik
Gebruik het kenmerk FontBold als u nadruk wilt leggen op tekens in een object of tekst in een invoergebied.

Als het kenmerk FontBold van bijvoorbeeld een aankruisvakje op waar (.T.) is ingesteld, wordt de tekst bij het aankruisvakje vet weergegeven. Als u het kenmerk FontBold van een invoervak op waar instelt, wordt de tekst in het invoervak vet weergegeven.

Voorbeeld
Zie FontName voor een voorbeeld van het gebruik van FontBold.

Zie ook
FontItalic, FontName, FontSize, FontStrikeOut, FontUnderline

FontItalic

Met het kenmerk FontItalic wordt aangegeven of bij een object tekens cursief worden weergegeven.

Kenmerk uit klasse
BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, LISTBOX, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SPINBOX, TEXT

Gegevenstype
Logisch

Standaardinstelling
De standaardinstelling van FontItalic is onwaar (.F.).

Gebruik
Gebruik het kenmerk FontItalic als u nadruk wilt leggen op tekens in een object of tekst in een invoergebied.

Als het kenmerk FontItalic van bijvoorbeeld een aankruisvakje op waar (.T.) is ingesteld, wordt de tekst bij het aankruisvakje cursief weergegeven. Als u het kenmerk FontItalic van een invoervak op waar instelt, wordt de tekst in het invoervak cursief weergegeven.

Voorbeeld

Zie FontName voor een voorbeeld van het gebruik van FontItalic.

Zie ook

FontBold, FontName, FontSize, FontStrikeOut, FontUnderline

FontName

Met het kenmerk FontName wordt bepaald in welk font de tekens in een object worden getoond.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, LISTBOX, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SPINBOX, TEXT

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van FontName is MS Sans Serif.

Gebruik

Gebruik het kenmerk FontName om het uiterlijk van de tekens in een object te bepalen. U kunt bijvoorbeeld tekens weergeven in het font Arial door de instelling "Arial" to te wijzen aan het kenmerk Name van een tekstobject.

Als het font dat u bij het kenmerk FontName opgeeft niet in Windows is geïnstalleerd, wordt de standaardinstelling MS Sans Serif gebruikt.

Opmerking U kunt een font selecteren in het dialoogvenster **Font** waarin u alle geïnstalleerde fonts vindt. U opent het dialoogvenster **Font** door op de hulpmiddelenknop naast de optie **FontName** in het kenmerkenvenster te klikken. U kunt het dialoogvenster **Font** ook openen met de functie GETFONT( ).

Voorbeeld

Syntaxis operator NEW:

```
Code = NEW ENTRYFIELD(this)
Code.Top = 5
```

```
Code.Left = 20
Code.Datalink = "Contact->CompCode"
Code.Height = 2
Code.FontBold = .T.
Code.FontSize = 12.00
Code.FontName = "Arial"
Code.FontItalic  = .T
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Code OF THIS;
  Property;
     Top 5, Left 20
     Datalink "Contact->CompCode",;
     Height 2,;
     FontBold .T.,;
     FontSize 12.00,;
     FontName "Arial",;
     FontItalic .T.
```

Zie ook

FontBold, FontItalic, FontSize, FontStrikeOut, FontUnderline, ScaleFontName

FontSize

Met het kenmerk FontSize wordt de grootte van de tekst in een object opgegeven.

.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, LISTBOX, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SPINBOX, TEXT

Gegevenstype

Numeriek

Gebruik

Gebruik het kenmerk FontSize om de grootte van de tekens die in een object worden weergegeven te bepalen. De grootte van een teken wordt aangegeven in punten. Hoe meer punten, hoe groter het teken.

Voorbeeld

Syntaxis operator NEW:

```
this.Tkst2 =NEW Text(this)
   this.Tkst2.Top = 5
   this.Tkst2.Left = 0
   this.Tkst2.Alignment = 4
   this.Tkst2.Text = "is hier in '94"
```

```
this.Tkst2.Height = 2
this.Tkst2.Width = 50
this.Tkst2.FontSize = 16
```

Syntaxis commando DEFINE:

```
DEFINE TEXT Tkst2 OF THIS AT 5,4;
   PROPERTY;
     Alignment 4,;
     Text "is hier in '94",;
     Height 2,;
     Width 50,;
     FontSize 16
```

Zie ook

FontBold, FontItalic, FontName, FontStrikeOut, FontUnderline, ScaleFontSize

FontStrikeOut

Met het kenmerk FontStrikeOut wordt bepaald of de tekens in een object doorgehaald worden weergegeven.

Kenmerk uit klasse

CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, LISTBOX, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SPINBOX, TEXT

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van FontStrikeOut is onwaar (.F.).

Gebruik

Gebruik het kenmerk FontStrikeOut als u een streep wilt trekken door de tekst in een object.

Als u bijvoorbeeld het kenmerk FontStrikeOut van een aankruisvakje op waar (.T.) instelt, wordt de tekst bij het aankruisvakje doorgehaald. Als u het kenmerk FontStrikeOut van een invoervak op waar instelt, wordt de tekst in het invoervak doorgehaald weergegeven.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL f
f=NEW Doorhln()
```

```
f.OPEN()
CLASS Doorhln OF FORM
   this.IVV1 = NEW ENTRYFIELD(this)
   this.IVV1.Top = 2
   this.IVV1.Left = 16
   this.IVV1.Width=6
   this.IVV1.Value = SPACE(6)
   this.IVV1.FontStrikeOut=.T.
ENDCLASS
```

Syntaxis commando DEFINE:

```
* Formulierdefinitie
DEFINE ENTRYFIELD IVV1 OF THIS;
  PROPERTY Top 2, Left 16, Width 6,;
  Value SPACE(6), FontStrikeOut .T.
```

Zie ook

FontBold, FontItalic, FontName, FontSize, FontUnderline

FontUnderline

Met het kenmerk FontUnderline wordt bepaald of tekens in een object worden onderstreept.

Kenmerk uit klasse

CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, LISTBOX, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SPINBOX, TEXT

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van FontUnderline is onwaar (.F.).

Gebruik

Gebruik het kenmerk FontUnderline als u tekst in een object of in een invoergebied wilt onderstrepen.

Als u het kenmerk FontUnderline van bijvoorbeeld een aankruisvakje op waar (.T.) is ingesteld, wordt de tekst bij het aankruisvakje onderstreept weergegeven. Als u het kenmerk FontUnderline van een invoervak op waar instelt, wordt de tekst in het invoervak onderstreept.

Voorbeeld

Zie FontName voor een voorbeeld van het gebruik van FontUnderline.

Zie ook

FontBold, FontItalic, FontName, FontSize, FontStrikeOut

Function

Met het kenmerk Function wordt de weergegeven tekst in een object opgemaakt.

Kenmerk uit klasse

ENTRYFIELD, SPINBOX, TEXT

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van Function is een lege tekenreeks.

Gebruik

Het argument dat u bij het kenmerk Function opgeeft bestaat uit een of meer functiesymbolen die onderling niet mogen worden gescheiden door spaties, komma's of andere tekens.

In dBASE worden de volgende functiesymbolen herkend:

(	Hiermee worden negatieve getallen tussen haakjes geplaatst.
!	Hiermee worden kleine letters in hoofdletters omgezet.
^	Hiermee worden getallen in de exponentiële notatie weergegeven.
$	Hiermee wordt een guldenteken of het symbool dat met SET CURRENCY TO is opgegeven, gebruikt in plaats van voorloopspaties.
A	Hiermee wordt de invoer beperkt tot letters.
B	Hiermee worden cijfers links uitgelijnd.
C	Hiermee wordt CR (credit) achter een positief getal geplaatst.
D	Hiermee worden ingevoerde datums weergegeven en geaccepteerd in de met SET DATE gedefinieerde notatie.
E	Hiermee worden ingevoerde datums weergegeven en geaccepteerd in de Europese notatie van (DD/MD/JJ).
I	Hiermee worden de ingevoerde gegevens gecentreerd.
J	Hiermee worden de ingevoerde gegevens rechts uitgelijnd.
L	Hiermee worden getallen getoond met voorloopnullen.

M	Hiermee worden vooraf gedefinieerde opties achter elkaar getoond, steeds wanneer de gebruiker op de *SpatieBalk* drukt.. U geeft deze opties (gescheiden door komma's) na de parameter @M op.
R	Hiermee worden vaste tekens in de weergave ingevoegd zonder deze in het veld op te nemen.
S<*n*>	Hiermee wordt de breedte van de weergave beperkt tot <*n*> tekens en worden de tekens in de weergave horizontaal over <*n*> kolommen verdeeld. De parameter <*n*> moet een positief geheel getal zijn.
T	Hiermee worden spaties voor en achter tekenreeksen verwijderd.
V<*n*>	Hiermee wordt een regelovergang ingesteld voor tekenreeksen langer dan <*n*>. De breedte wordt opgegeven in inches maal 10 (een waarde van bijvoorbeeld 15,5 voor <*n*> is gelijk aan 1,55 inch (3,94 cm)).
X	Hiermee wordt DB (debet) achter een negatief getal geplaatst.
Z	Hiermee worden nullen als spaties getoond.

Voorbeeld

Zie DataLink voor een voorbeeld van het gebruik van Function.

Zie ook

DEFINE, Picture

GetTextExtent( )

Het resultaat van de methode GetTextExtent( ) is de lengte van een tekenreeks, uitgedrukt in eenheden die zijn gebaseerd op de fontgrootte van een hoofdformulier.

Kenmerk uit klasse

TEXT

Gebruik

Gebruik de methode GetTextExtent( ) om een waarde te berekenen voor het kenmerk Width van een tekstobject. De instelling van het kenmerk Width moet bijvoorbeeld worden aangepast telkens wanneer een nieuwe tekenreeks wordt getoond die langer is dan het tekstobject.

Bij de methode GetTextExtent( ) wordt het font van het tekstobject vergeleken met het font van het hoofdformulier. Het resultaat is de lengte van de tekenreeks, uitgedrukt in *tekeneenheden*. De grootte van een tekeneenheid is gelijk aan de breedte van een kolom in het coördinatenvlak van het hoofdformulier. Als u de lengte van de reeks in tekeneenheden opgeeft bij het kenmerk Width van het tekstobject, wordt het object aangepast aan de lengte van de tekenreeks.

Bij de methode GetTextExtent( ) wordt de parameter <*tekenreeks*> geaccepteerd.

Voorbeeld

In het volgende voorbeeld wordt een formulier gemaakt met twee lange invoervakken. Wanneer het tweede veld wordt verlaten, wordt met het kenmerk OnLostFocus een codeblok aangeroepen waarmee een tekenreeksvariabele wordt geïnitialiseerd. Deze variabele is een combinatie van de reeksen in beide invoervakken. Vervolgens maakt het codeblok gebruik van de methode GetTextExtent( ) om de lengte van de reeks te bepalen (op basis van het huidige font) zodat het kenmerk Width van tekstobject T1 kan worden aangepast aan de lengte van de gecombineerde tekenreeksen:

```
LOCAL f
f = NEW XFORM()
f.Open()

CLASS XFORM OF FORM
   this.HelpFile = ""
   this.Top =          2.76
   this.Height =         11.71
   this.Text = "Formulier"
   this.Left =         53.00
   this.Width =         41.80
   this.HelpId = ""

   DEFINE ENTRYFIELD VNAAM OF THIS;
       PROPERTY;
         Top          3.00,;
         Height          1.00,;
         Value "                    ",;
         Border .T.,;
         Left          6.00,;
         Width         20.00

   DEFINE ENTRYFIELD ANAAM OF THIS;
       PROPERTY;
         OnLostFocus CLASS::XLENGTE,;
         Top          5.00,;
         Height          1.00,;
         Value "                    ",;
         Border .T.,;
         Left          6.00,;
         Width         20.00

   DEFINE TEXT T1 OF THIS;
       PROPERTY;
         Top          7.00,;
         Height          1.00,;
         Border .F.,;
         Text "Voor- en achternaam",;
         ColorNormal "N/W",;
         Left          6.00,;
         Width         25.00

         PROCEDURE xLengte
           ExtReeks =;
           TRIM(form.Vnaam.Value)+" "+form.Anaam.Value
           Form.T1.Width =;
```

```
        form.T1.GetTextExtent(ExtReeks)
        Form.T1.Text = ExtReeks

ENDCLASS
```

Zie ook

FontName, FontSize, LEN( ), ScaleFontName, ScaleFontSize

Group

Met het kenmerk Group wordt een objectgroep gestart in de tabvolgorde van het hoofdformulier.

Kenmerk uit klasse

CHECKBOX, PUSHBUTTON, RADIOBUTTON

Gegevenstype

Logisch

Gebruik

Gebruik het kenmerk Group om te bepalen of een object deel uitmaakt van een groep waarbinnen de gebruiker de focus kan verplaatsen met de pijltoetsen.

Als u het kenmerk Group gebruikt om twee of meer keuzerondjes te groeperen, kan de gebruiker steeds maar één keuzerondje van de groep selecteren. Dit is handig als u de gebruiker één mogelijkheid uit een aantal waarden wilt laten selecteren.

U maakt een groep door het kenmerk Group van het eerste object op waar (.T.) in te stellen. Bij alle andere objecten die tot dezelfde groep behoren als dit object, zet u het kenmerk Group op onwaar (.F.). Met het volgende object waarbij het kenmerk Group op waar staat, wordt een nieuwe groep begonnen.

Voorbeeld

Syntaxis operator NEW:

```
Antwoord=""
Ja = New RADIOBUTTON(this)
Ja.Text = "Ja"
Ja.DataLink = "Antwoord"
Ja.Group = .F.
Nee = New RADIOBUTTON(this)
Nee.Top = 5
Nee.Text = "Nee"
Nee.DataLink = "Antwoord"
```

Syntaxis commando DEFINE:

```
Antwoord=""
DEFINE RADIOBUTTON Ja OF THIS;
   PROPERTY;
      Text "Ja",;
      DataLink "Antwoord",;
      Group .F.
DEFINE RADIOBUTTON Nee OF THIS;
   PROPERTY;
      Top 5,;
      Text "Nee",;
      DataLink "Antwoord"
```

Zie ook

_curobj, ID, TabStop

Grow( )

Met de methode Grow( ) worden elementen toegevoegd aan een array-object.

Kenmerk uit klasse

ARRAY

Gebruik

Gebruik de methode Grow( ) om een element, rij of kolom aan een array-object toe te voegen. Alle nieuwe elementen hebben de logische waarde .F. De methode Grow( ) lijkt op de functie AGROW( ).

Bij de methode Grow( ) moet één parameter, <*Nuitdr*>, worden opgegeven, deze kan de waarde 1 of 2 hebben. Als u een andere waarde probeert toe te kennen, wordt de foutmelding "Ongeldig functie-argument" weergegeven.

- Als u voor de parameter <*Nuitdr*> de waarde 1 gebruikt, wordt met de methode Grow( ) één element toegevoegd aan een ééndimensionaal array-object of een rij aan een tweedimensionaal array-object.
- Als u voor de parameter <*Nuitdr*> de waarde 2 gebruikt, wordt met de methode Grow( ) een kolom aan een array-object toegevoegd. Als het array-object ééndimensionaal is, wordt het tweedimensionaal gemaakt: bestaande elementen worden naar de eerste kolom verplaatst en de tweede kolom wordt gevuld met de waarde .F.

  Als het array-object tweedimensionaal is, wordt aan het einde van de array een kolom toegevoegd die met nieuwe elementen wordt gevuld die alle de waarde .F. krijgen.

Zie Insert( ) voor een andere manier om elementen aan een array-object toe te voegen. Als u meerdere rijen of kolommen aan een array-object wilt toevoegen, gebruikt u de methode Resize( ).

Voorbeeld

```
USE Klanten.DBF
* Array-object initialiseren.
ArrObj=NEW ARRAY(RECCOUNT())
* Eéndimensionale array vullen met waarden
* uit veld Naam van Klanten.DBF
GO TOP
FOR i=1 TO RECCOUNT()
  ArrObj[i]=Klanten->Naam
  SKIP
NEXT i
* GROW() gebruiken om tweede kolom toe te voegen aan
* bestaande array en gegevens uit veld Verktotnu in
* die kolom in te vullen.
ArrObj.GROW(2)
GO TOP
FOR i=1 TO RECCOUNT()
  ArrObj[i,2]=Klanten->Verktotnu
  SKIP
NEXT i
* Inhoud tweedimensionale array tonen.
FOR i=1 TO RECCOUNT()
? ArrObj[i,1], ArrObj[i,2]
NEXT i
```

Zie ook

AGROW( ), Insert( ) , Resize( )

Height

Met het kenmerk Height wordt de hoogte van een object bepaald.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Numeriek

Gebruik

Gebruik het kenmerk Height in combinatie met het kenmerk Width om de afmetingen van een object aan te passen. De instellingen die u definieert met het kenmerk Height bepaalt de afstand tussen de boven- en onderzijde van een object.

Elke eenheid van de waarde die u aan het kenmerk Height toewijst, is gelijk aan de gemiddelde hoogte van de tekens in het actieve font van het hoofdformulier. Als u het kenmerk Height bijvoorbeeld instelt op 17,5, wordt het editor-object 17,5 tekens hoog.

Aangezien bij het kenmerk zowel gehele als gebroken getallen kunnen worden opgegeven, kunt u de hoogte van een object zeer precies instellen.

Voorbeeld

Syntaxis operator NEW:

```
Code = NEW ENTRYFIELD(this)
Code.Top = 5
Code.Left = 1
Code.Datalink = "Contact->CompCode"
Code.Height = 2
Code.Left = 20
* Standaardhoogte van invoervak is 1,17
* daarom wordt met instelling 2 voor hoogte
* in dit voorbeeld alleen extra ruimte rond waarde
* in veld gecreëerd.
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Code OF THIS AT 5,1;
  Property;
     Datalink "CompCode",;
     Height 2, Left 20
```

Zie ook

Left, Top, Width

HelpFile

Met het kenmerk HelpFile wordt een Windows-helpbestand (.HLP) opgegeven met contextgevoelige Help-onderwerpen.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, SCROLLBAR, SPINBOX

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van HelpFile is een lege tekenreeks.

Gebruik

Gebruik het kenmerk HelpFile in combinatie met het kenmerk HelpID om een object aan een onderwerp in een Windows-helpbestand te koppelen. Het kenmerk HelpID kan bijvoorbeeld verwijzen naar een Help-onderwerp met instructies voor het gebruik van keuzelijsten met invoervakken.

Geef de naam van het Help-bestand op bij het kenmerk HelpFile en stel het kenmerk HelpID in op het Help-sleutelwoord van het onderwerp. Wanneer de gebruiker de focus naar een object verplaatst en op *F1* drukt, wordt het Help-bestand geopend en het Help-onderwerp weergegeven.

Zie de documentatie bij Windows voor informatie over het maken van Help-bestanden en Help-onderwerpen.

Opmerking Subroutines die u bij het kenmerk OnHelp opgeeft, hebben prioriteit boven de waarden die u toekent aan de kenmerken HelpFile en HelpID.

Voorbeeld

Syntaxis operator NEW:

```
Bedrijf = NEW ENTRYFIELD(this)
Bedrijf.Top = 3
Bedrijf.Left = 1
Bedrijf.Datalink = "Bedrijf"
Bedrijf.HelpFile = "GegInv"
Bedrijf.HelpId = 100
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Bedrijf OF THIS AT 3,1;
  Property;
     Datalink "Bedrijf",;
     HelpFile "GegInv",;
     HelpId 100
```

Zie ook

HELP, HelpID, OnHelp, SET HELP

HelpID

Met het kenmerk HelpID wordt een Help-sleutelwoord van een onderwerp een Help-bestand opgegeven.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, SCROLLBAR, SPINBOX

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van HelpID is een lege tekenreeks.

Gebruik

Gebruik het kenmerk HelpID in combinatie met het kenmerk HelpFile om een object aan een onderwerp in een Help-bestand van Windows (.HLP) te koppelen. Het bestand dat u bij het kenmerk HelpFile opgeeft kan bijvoorbeeld een Help-onderwerp bevatten met instructies over het gebruik van objecten in een formulier.

Stel het kenmerk HelpID in op het Help-sleutelwoord van het onderwerp en geef de naam van het Help-bestand op bij het kenmerk HelpFile. Als de gebruiker focus geeft aan het object en op *F1* drukt, wordt het Help-bestand geopend en het Help-onderwerp weergegeven.

Opmerking Subroutines die u bij het kenmerk OnHelp opgeeft, hebben prioriteit boven de waarden die u bij de kenmerken HelpID en HelpFile specificeert.

Voorbeeld

Zie HelpFile voor een voorbeeld van het gebruik van HelpId.

Zie ook

HELP, HelpFile, OnHelp, SET HELP

hWnd

Met het kenmerk hWnd wordt een object-handle opgegeven.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX

Gegevenstype

Numeriek

Standaardinstelling

Het kenmerk hWnd kent geen standaardinstelling; de waarde wordt onder Windows automatisch gegenereerd.

Gebruik

Een object-handle is een unieke numerieke waarde die in Windows automatisch wordt gegenereerd voor elk object dat u maakt. Bij externe functies die geschreven zijn in talen als C, Pascal of ASM wordt deze handle gebruikt om een object aan te duiden. U kunt bijvoorbeeld de object-handle van een formulier als parameter doorgeven aan een externe functie, wellicht om met deze functie het fomulier te openen of te sluiten.

Dergelijke externe functies staan in externe C-bibliotheken zoals Windows API of een door uzelf gemaakt DLL-bestand. Zie EXTERN, LOAD DLL en Hoofdstuk 25 in *Programmeren* voor meer informatie over externe functies en DLL-bestanden.

De instelling van het kenmerk hWnd kan alleen worden gelezen.

Voorbeeld

```
DEFINE FORM Invoer FROM 2,2 TO 30,70
DEFINE ENTRYFIELD Bedrijf OF THIS;
  PROPERTY;
    Datalink "Bedrijf"
    FormHandle=Invoer.hwnd            && Bijvoorbeeld 14260
    BedrijfHandle=Invoer.Bedrijf.hwnd  && Bijvoorbeeld 14348
```

Zie ook

EXTERN, LOAD DLL

ID

Het kenmerk ID is een numerieke aanduiding van een object.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, MENU, PUSHBUTTON, RADIOBUTTON, SCROLLBAR, SPINBOX

Gegevenstype

Numeriek

Gebruik

Gebruik het kenmerk ID om een object een unieke identificatie te geven.

Bijvoorbeeld, wanneer de gebruiker een knop kiest, wordt het formulier voorgelegd en wordt de subroutine of het codeblok uitgevoerd dat u hebt opgegeven bij het kenmerk OnSelection of bij het commando ON SELECTION FORM. De waarde van het kenmerk ID wordt doorgegeven aan de subroutine of het codeblok, zodat de knop wordt aangeduid waarmee het formulier is voorgelegd.

Desgewenst kunt u de waarde van het kenmerk ID gebruiken bij sprongopdrachten die zijn gebaseerd op het object waarmee de subroutine van het kenmerk OnSelection wordt uitgevoerd. Om er zeker van te zijn dat met het kenmerk ID het juiste object wort aangeduid, moet u het bij elk object een unieke waarde geven.

De ID-waarden van menu-objecten worden door dBASE gegenereerd en kunnen niet worden gewijzigd.

Voorbeeld

Syntaxis operator NEW:

```
Datum = NEW ENTRYFIELD(this)
Datum.Datalink = "Afnemers->BalnsDatum"
Datum.Id = 10
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Datum OF THIS;
  PROPERTY Datalink "Afnemers->BalnsDatum",;
  ID 10
```

Zie ook

OnSelection , ON SELECTION FORM, PARAMETERS

Initiate( )

Met de methode Initiate( ) wordt een verbinding gemaakt met een externe toepassing.

Kenmerk uit klasse

DDELINK

Gebruik

Gebruik de methode Initiate( ) om een verbinding tot stand te brengen (een zogenaamde *DDE-koppeling*) tussen dBASE en een externe Windows-toepassing.

Gebruik een DDE-koppeling om gegevens en instructies uit te wisselen met een andere toepassing. In een programma voor gegevensuitwisseling kan de methode Initiate( ) bijvoorbeeld worden gebruikt om een verbinding te maken met Quattro Pro voor Windows, daarin een spreadsheet te openen en vervolgens gegevens vanuit de spreadsheet naar een dBASE-tabel te kopiëren.

Als de externe toepassing nog niet is gestart wanneer u de methode Initiate( ) uitvoert, wordt geprobeerd de toepassing te starten voordat de DDE-koppeling tot stand wordt gebracht. Als dat niet lukt, is het resultaat van de methode Initiate( ) onwaar (.F.).

Bij de methode Initiate( ) moet u twee parameters opgeven:

- *<server>* Dit is het primaire programmabestand van de server-toepassing.
- *<onderwerp>* Dit is de naam van het gegevensbestand dat in de server-toepassing moet worden geopend.

Als u de DDE-koppeling wilt beëindigen, gebruikt u de methode Terminate( ).

Voorbeeld

```
PUBLIC KoppObj
KoppObj = NEW DDELINK()
KoppObj.Initiate("QPW","Demo.WB1");
              && Koppeling maken met een
              && QPW-server
```

Zie ook

Advise( ), Execute( ), OnNewValue, Peek( ), Poke( ), Server, Terminate( ), TimeOut, Topic, Unadvise( )

Insert( )

Met de methode Insert( ) wordt de waarde .F. ingevoegd in een element in een ééndimensionaal array-object of in een rij of kolom in een tweedimensionaal array-object.

Kenmerk uit klasse

ARRAY

Gebruik

Gebruik de methode Insert( ) om de waarde .F. in te voegen in geselecteerde elementen in een array-object zonder de grootte daarvan te wijzigen. De methode Insert( ) doet het volgende:

- Er wordt een element in een ééndimensionale array ingevoegd of een rij of kolom in een tweedimensionale array
- Alle overige elementen worden naar het einde van de array verplaatst (naar beneden als een rij wordt ingevoegd en naar rechts als een element of kolom wordt ingevoegd).
- De waarde .F. wordt ingevoegd in het nieuwe element of in de nieuwe elementen.

U kunt twee parameters opgeven bij de methode Insert( ):

- *<positie Nuitdr>* Hiermee wordt het nummer van het element opgegeven waarbij het nieuwe element in een ééndimensionale array moet worden ingevoegd. Met deze parameter wordt in geval van een tweedimensionale array een rij of kolom opgegeven.
- Met de waarde 2 wordt bepaald of met de methode Insert( ) een rij of een kolom in een tweedimensionale array moet worden ingevoegd. Als u de waarde 2 opgeeft, wordt een rij ingevoegd en anders een kolom. In geval van een ééndimensionale array wordt deze parameter genegeerd.

Zie Delete( ) voor informatie over het invoegen van elementen en de overige elementen in de richting van het *begin* van de array verplaatsen. Zie Fill( ) voor informatie over het vervangen van elementen zonder de overige elementen te verplaatsen. Zie Grow( ) of Resize( ) over het omzetten van een ééndimensionale array in een tweedimensionale array.

Voorbeeld

```
USE Dieren.DBF
* Array-object initialiseren.
ArrObj=NEW ARRAY(RECCOUNT(),3)
* Array-object vullen met waarden uit Dieren.DBF
COPY TO ARRAY ArrObj FIELDS Naam, Gebied, Gewicht
* Insert() gebruiken om gewichten in
* kolom 3 te vervangen met waarde .F.
ArrObj.Insert(3,2)
* Resultaten tonen in resultatenpaneel van commandovenster.
FOR i=1 TO RECCOUNT()
? ArrObj[i,1],ArrObj[i,2],ArrObj[i,3]
NEXT i
```

Zie ook

Add( ), Delete( ) , Resize( )

Key

Met het kenmerk Key wordt een subroutine uitgevoerd wanneer de gebruiker in een invoervak of bladerobject op een toets drukt.

Kenmerk uit klasse

BROWSE, ENTRYFIELD

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk Key om elk teken dat een gebruiker in een invoervak of bladerobject invoert te evalueren en aanpassen. Bij een invoervak voor wachtwoorden kan elk teken dat de gebruiker typt in asterisken worden omgezet.

Net zoals bij andere actiekenmerken, kunt u bij het kenmerk Key het volgende opgegeven:

- Functies
- Procedures
- Codeblokken

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Met het kenmerk Key worden twee parameters naar de bijbehorende subroutine doorgestuurd:

- *<tekennummer>* Dit is de ASCII-code van het nieuwe teken dat de gebruiker typt.
- *<positie>* Dit is de positie van het nieuwe teken in de tekenreeks.

Als bijvoorbeeld het derde teken een "a" is, is de parameter *<tekennummer>* gelijk aan 97 en de parameter *<positie>* aan 3.

Uw subroutine voor Key moet resulteren in een numerieke of een logische waarde. Een numerieke waarde wordt geïnterpreteerd als de decimale waarde van een ASCII-teken, dat automatisch het door de gebruiker ingevoerde teken vervangt. Een logische waarde wordt geïnterpreteerd als beslissing om een ingevoerd teken al dan niet te accepteren.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **Key** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
Bedrijf = NEW ENTRYFIELD(this)
Bedrijf.Datalink = "Bedrijf->Bedrijf"
Bedrijf.Key = EenToets
* Functie EenToets aanroepen.
FUNCTION EENTOETS(teken, positie)
* Toetsaanslag verwerken.
RETURN teken + 1
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Bedrijf OF THIS;
  PROPERTY;
     Datalink "Bedrijf->Bedrijf",;
     Key EenToets
```

Zie ook
OnChange, OnGotFocus

Left

Met het kenmerk Left wordt de positie van de linkerrand van een object opgegeven ten opzichte van het hoofdformulier of de positie van een formulier ten opzichte van het bureaublad.

Kenmerk uit klasse
BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LINE, LISTBOX, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX

Gegevenstype
Numeriek

Gebruik
Gebruik het kenmerk Left in combinatie met het kenmerk Top om een object in een formulier op de gewenste plaats te zetten of een formulier op de gewenste plaats op het bureaublad te zetten (met het kenmerk Top wordt de positie van de bovenste rand gedefinieerd). De kenmerken Left en Top kunnen op gehele en gebroken getallen worden ingesteld.

Als u de clausules FROM of AT bij het commando DEFINE opgeeft *en* een waarde voor het kenmerk Left definieert, wordt de instelling van het kenmerk Left gebruikt.

Elke eenheid van de waarde die u aan het kenmerk Left toewijst, is gelijk aan de gemiddelde breedte van de tekens in het actieve font van het hoofdformulier. Als u het kenmerk Left bijvoorbeeld instelt op 20, wordt het linkeruiteinde van de lijn op een afstand van 20 tekens van de linkerrand van het formulier geplaatst.

Voorbeeld
Zie Height voor een voorbeeld van het gebruik van Left.

Zie ook
CLASS FORM, Bottom, Height, Right, ScaleFontName, ScaleFontSize, Top, Width

LineNo

Met het kenmerk LineNo wordt de huidige regel in een editor-object gedefinieerd.

Kenmerk uit klasse
EDITOR

Gegevenstype
Numeriek

Standaardinstelling
De standaardinstelling van LineNo is 1.

Gebruik
Gebruik het kenmerk LineNo om de cursor naar een bepaalde regel in een editor-object te verplaatsen.

Wanneer u de cursor naar een regel verplaatst, blijft deze in dezelfde kolom staan als de regel lang genoeg is. Als dat niet het geval is, komt de cursor aan het einde van de regel te staan.

Voorbeeld
```
LOCAL f
f=NEW Aanpassen()
f.OPEN()
CLASS Aanpassen OF EDITOR
   this.LineNo = 15 && Cursor naar 15de regel verplaatsen.
ENDCLASS
```

Zie ook
Wrap

LinkFileName

Met het kenmerk LinkFileName wordt eventueel aangegeven welk OLE-document aan het huidige OLE-veld is gekoppeld wanneer dat veld in een OLE-weergave wordt getoond.

Kenmerk uit klasse
OLE

Gegevenstype
Teken

Standaardinstelling

De standaardinstelling van LinkFileName is een lege tekenreeks.

Gebruik

Gebruik het kenmerk LinkFileName om aan te geven welk OLE-document aan het huidige OLE-veld is gekoppeld wanneer dat veld in een OLE-weergave wordt getoond.

In een OLE-weergave wordt een OLE-document (ook wel OLE-object genoemd) getoond. Een OLE-document kan een grafische afbeelding zijn, een document dat in een tekstverwerker is gemaakt of elk ander gegevensobject dat met een externe toepassing is gemaakt. Deze externe toepassing wordt de *OLE-server* genoemd. Een bitmap-bestand dat bijvoorbeeld in Paintbrush is gemaakt kan een OLE-server zijn als u dat bestand met **Knippen** en **Koppeling plakken** uit het menu **Bewerken** aan een OLE-veld koppelt. Het kenmerk LinkFileName bevat de naam van een dergelijk bestand.

De instelling van het kenmerk LinkFileName kan alleen worden gelezen.

Voorbeeld

Het volgende programma maakt een formulier met een OLE-object en een knop. De routine OnClick van de knop gebruikt LinkFileName om de bestandsnaam van het document in een meldingvenster te tonen.

```
LOCAL f
f = NEW XFORM()
f.Open()

CLASS XFORM OF FORM
   this.HelpFile = ""
   this.Top =          0.47
   this.Height =         26.00
   this.Text = "Formulier"
   this.View = "AFBEELD.QBE"
   this.MousePointer =          1
   this.Left =         23.00
   this.Width =        101.20
   this.HelpId = ""

   DEFINE OLE OLE1 OF THIS;
       PROPERTY;
         DataLink "AFBEELD->BITMAPOLE",;
         Top          1.00,;
         Height         21.00,;
         Border .T.,;
         Left          1.00,;
         Width         99.00

   DEFINE PUSHBUTTON KNOP1 OF THIS;
       PROPERTY;
         Top         23.00,;
         Height          2.00,;
         OnClick CLASS::KNOP1_ONCLICK,;
         Text "Klik hier voor bitmap-naam",;
```

```
        ColorNormal "N/W",;
        Default .T.,;
        Left         36.00,;
        Width         31.00

    Procedure KNOP1_OnClick
    MSGBOX(form.OLE1.LinkFileName, "Zo heet het bestand...")

ENDCLASS
```

Zie ook

OleType, REPLACE OLE, ServerName

Maximize

Met het kenmerk Maximize kan worden bepaald of een formulier tot een maximumvenster kan worden vergroot.

Kenmerk uit klasse

FORM

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Maximize is waar (.T.).

Gebruik

Net zoals bij andere vensters ziet u bij een formulier normaalgesproken de knoppen Pictogram en Maximumvenster in de rechterbovenhoek. Als u op de knop Pictogram klikt, wordt het formulier verkleind tot een pictogram en als u op de knop Maximumvenster klikt, wordt het formulier net zo groot als het werkgebied van het bureaublad.

Als het kenmerk Maximize op onwaar (.F.) is ingesteld, wordt de knop Maximumvenster verwijderd, zodat de gebruiker het formulier niet meer kan vergroten tot een maximumvenster. Dit is handig als bepaalde delen van het werkgebied steeds zichtbaar moeten zijn, bijvoorbeeld wanneer andere formulieren zijn geopend of wanneer de gebruiker toegang moet hebben tot het commandovenster.

Opmerking Als u het kenmerk MDI van een formulier instelt op waar (.T.), wordt de instelling van het kenmerk Maximize genegeerd en kan de gebruiker het formulier steeds tot een maximumvenster vergroten.

Voorbeeld

Syntaxis operator NEW:

```
f=NEW InvoerForm()
f.OPEN()
CLASS InvoerForm OF FORM
  this.MDI = .F.
  this.Maximize = .F.
  this.Minimize = .F.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM Invoer FROM 1,1 TO 30,35;
  Property;
      MDI = .F.,;
      Maximize .F.,;
      Minimize .F.
```

Zie ook

MDI, Moveable, Minimize, Sizeable

MaxLength

Met het kenmerk MaxLength wordt opgegeven hoever tekst in een invoervak kan worden verschoven.

Kenmerk uit klasse

ENTRYFIELD

Gegevenstype

Numeriek

Gebruik

Gebruik het kenmerk MaxLength om het aantal kolommen dat een gebruiker tekst in een invoervak naar links kan schuiven te beperken.

Een gebruiker kan bijvoorbeeld gegevens in een invoervak typen, ook als dat niet aan een veld is gekoppeld. Het kan nodig zijn het aantal tekens dat een gebruiker kan typen te beperken, zodat de inhoud van het invoervak later in een tekenveld kan worden geplaatst. Als u het kenmerk MaxLength instelt op een waarde die gelijk is aan of kleiner is dan de lengte van het veld, voorkomt u dat een gebruiker een te lange tekenreeks invoert.

De kolommen zijn net zo breed als de gemiddelde breedte van de tekens in het actieve font.

Voorbeeld

Syntaxis operator NEW:

```
Bedrijf = NEW ENTRYFIELD(this)
Bedrijf.Top = 3
Bedrijf.Left = 1
Bedrijf.Datalink = "Bedrijf"
Bedrijf.MaxLength  = 10
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Bedrijf OF THIS;
  AT 3,1;
  Property;
     Datalink "Bedrijf",;
     MaxLength  10
```

Zie ook

Function, Height, Picture, ScaleFontName, ScaleFontSize, Width

MDI

Met het kenmerk MDI wordt bepaald of een formulier voldoet aan de standaard van de Windows Multiple Document Interface (MDI).

Kenmerk uit klasse

FORM

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van MDI is waar (.T.).

Gebruik

MDI (Multiple Documents Interface) is een voorziening van Windows die het mogelijk maakt meerdere *documentvensters* in een toepassingsvenster te openen. Een toepassing zoals dBASE maakt gebruik van MDI om meerdere documenten of meerdere weergaven van hetzelfde document in het hoofdtoepassingsvenster te beheren. Deze weergaven of documenten worden in aparte documentvensters getoond. Als u uw formulier in een documentvenster wilt bekijken, stelt u het kenmerk MDI in op waar (.T.).

Hieronder vindt u enkele eigenschappen van documentvensters:

- Documentvensters kunnen net als toepassingsvensters worden verplaats, vergroot en verkleind.
- Documentvensters worden als opdracht getoond in het menu **Venster**, ook als het venster actief is.
- Documentvensters hebben een systeemmenu, de knoppen Pictogram en Maximumvenster en een titelbalk met de naam van het venster.
- Als een documentvenster actief is, vervangen de bijbehorende menu's de primaire menubalk (als het kenmerk MDI op onwaar (.F.) is ingesteld, worden de menu's in de menubalk van het formulier getoond).

Als u het kenmerk MDI op waar (.T.) instelt, gebeurt het volgende:

- Het formulier wordt een element van het toepassingsvenster.
- De instelling van het kenmerk Minimize wordt genegeerd, zodat de gebruiker het formulier altijd tot een pictogram kan verkleinen.
- De instelling van het kenmerk Maximize wordt genegeerd, zodat de gebruiker het formulier altijd tot een maximumvenster kan vergroten.
- De instelling van het kenmerk Sizeable wordt genegeerd, zodat de gebruiker de afmetingen van het formulier steeds kan wijzigen.
- De instelling van het kenmerk Moveable wordt genegeerd, zodat de gebruiker het formulier steeds kan verplaatsen.
- De instelling van het kenmerk SysMenu wordt genegeerd, zodat het systeemmenu steeds beschikbaar is met het *Symbool Systeemmenu*, de knop in de linkerbovenhoek van het formulier.
- De sneltoets waarmee het formulier wordt gesloten is *Ctrl-F4* in plaats van *Alt-F4*, dat wil zeggen: met *Alt-F4* wordt het toepassingsvenster gesloten en met *Ctrl-F4* het formulier.

Een MDI-formulier kan per definitie niet als modaal worden gedefinieerd. De focus kan immers worden verwijderd. Daarom kunt u een MDI-formulier ook niet openen met de functie READMODAL( ) of de methode ReadModal( ).

Zie de documentatie bij Windows voor meer informatie over MDI.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL f
f = NEW InvoerForm()
f.MDI  = .F.
f.Open()
```

Syntaxis commando DEFINE:

```
DEFINE FORM InvoerForm ;
  Property MDI  .F.
OPEN FORM InvoerForm
```

Zie ook

CLASS FORM, Maximize, MenuFile, Minimize, Moveable, Sizeable, SysMenu

MenuFile

Met het kenmerk MenuFile wordt een vooraf gedefinieerd menusysteem aan een formulier toegewezen.

Kenmerk uit klasse

FORM

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van MenuFile is een lege tekenreeks.

Gebruik

Gebruik het kenmerk MenuFile om een menudefinitiebestand (.MNU) voor een formulier op te geven.

Een menudefinitiebestand is een tekstbestand dat dBASE-code bevat waarmee menu's kunnen worden gegenereerd. In dBASE worden menu's in het formulier of het toepassingsvenster weergegeven (afhankelijk van de instelling van het kenmerk MDI) wanneer u het formulier opent.

Opmerking In Menu-ontwerp kunt u een menudefinitiebestand maken. U start Menu-ontwerp door op de hulpmiddelenknop naast de optie **MenuFile** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL f
f = NEW MATRIEELFORM()
f.Open()

CLASS MATRIEELFORM OF FORM
   Set Procedure To KNOPPEN.CC additive
   this.MenuFile = "Matrieel.MNU"
   this.Text = "Beheer vluchtmaterieel"
   this.View = "Matrieel.QBE"
   this.Width =         99.5
   this.Height =         26
   this.MDI = .F.
* Zie Matrieeel.WFM en Matrieel.MNU in directory
* DBASEWIN\VOORBD.
```

Syntaxis commando DEFINE:

```
DEFINE FORM MatrieelForm;
  PROPERTY;
  MDI .F.,;
  Width 99.5, Height 26,;
  View "Matrieel.QBE",;
  MenuFile "Matrieel.MNU"
```

Zie ook

CLASS MENU

Minimize

Met het kenmerk Minimize wordt bepaald of een fomulier tot een pictogram kan worden verkleind.

Kenmerk uit klasse

FORM

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Minimize is waar (.T.).

Gebruik

Zoals bij andere vensters ziet u bij een formulier normaal gesproken de knoppen Pictogram en Maximumvenster in de rechterbovenhoek. Als u op de knop Pictogram klikt, wordt het formulier verkleind tot een pictogram en als u op de knop Maximumvenster klikt, wordt het formulier net zo groot als het werkgebied van het bureaublad.

Als het kenmerk Minimize op onwaar (.F.) is ingesteld, wordt de knop Pictogram verwijderd zodat de gebruiker het formulier niet meer tot pictogram kan verkleinen. Dit is handig als het formulier altijd zichtbaar moet zijn omdat het objecten of gegevens bevat die meteen aandacht nodig hebben.

Opmerking Als u het kenmerk MDI van een formulier instelt op waar (.T.), wordt de instelling van het kenmerk Minimize genegeerd en kan de gebruiker het formulier steeds tot pictogram verkleinen.

Voorbeeld

Zie Maximize voor een voorbeeld van het gebruik van Minimize.

Zie ook
MDI, Moveable, Maximize, Sizeable

Mode

Met het kenmerk Mode wordt de opmaak van een bladerobject opgegeven.

Kenmerk uit klasse
BROWSE

Gegevenstype
Numeriek

Standaardinstelling
De standaardinstelling van Mode is 0 (bladeren).

Gebruik
Gebruik het kenmerk Mode om te bepalen hoe gegevens in een bladerobject worden getoond.

Het kenmerk Mode kent de volgende drie instellingen:

Instelling	Opmaak	Zichtbaar
0 (bladeren)	Rijen en kolommen	Meerdere records
1 (formulier bewerken)	Formulier	Eén record
2 (kolom bewerken)	Kolom	Eén record

Als het kenmerk Toggle op waar (.T.) is ingesteld, kan de gebruiker tussen deze verschillende soorten opmaak heen en weer schakelen door op F2 te drukken.

Voorbeeld
Syntaxis operator NEW:

```
BedrijfBladeren = NEW BROWSE(this)
BedrijfBladeren.Top = 3
BedrijfBladeren.Left = 1
BedrijfBladeren.Height = 10
BedrijfBladeren.Width = 59
BedrijfBladeren.Alias = "Bedrijf"
BedrijfBladeren.Mode = 2
BedrijfBladeren.Fields="Bedrijf,Verktotnu;
  :H='Verkoop,Totnu'"
```

Syntaxis commando DEFINE:

```
DEFINE BROWSE BedrijfBladeren OF THIS;
  FROM 3,1 TO 13,60;
    PROPERTY;
      Alias "Bedrijf", Mode 2,;
      Fields "Bedrijf,Verktotnu:H='Verkoop,Totnu'"
```

Zie ook

BROWSE, CLASS BROWSE, EDIT

Modify

Met het kenmerk Modify wordt bepaald of een gebruiker records in een blader- of editor-object kan wijzigen.

Kenmerk uit klasse

BROWSE, EDITOR

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Modify is waar (.T.).

Gebruik

Stel het kenmerk Modify in op onwaar (.F.) als u wilt voorkomen dat gebruikers records in een bladerobject wijzigen. In een elektronische kaartenbak worden bijvoorbeeld gegevens over boeken in een bibliotheek bijgehouden en elk record bevat een memoveld met een boekbeschrijving. Gebruikers kunnen in het programma bijvoorbeeld wel de beschrijvingen in een bladerobject bekijken maar niet wijzigen.

Voorbeeld

Syntaxis operator NEW:

```
this.Blader1=NEW BROWSE(this)
  this.Blader1.Alias="Contact"
  this.Blader1.Modify=.F.
  this.Blader1.Top=4
  this.Blader1.Left=3
  this.Blader1.Width=32
  this.Blader1.Height=12
```

Syntaxis commando DEFINE:

```
DEFINE BROWSE Blader1 OF THIS;
  PROPERTY;
    Alias "Contact", Modify .F.,;
    Top 4, Left 3, Width 32, Height 12
```

Zie ook

Append, Delete

MousePointer

Met het kenmerk MousePointer kan het uiterlijk van de muisaanwijzer worden gewijzigd.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Numeriek

Standaardinstelling

De standaardinstelling van MousePointer is 0 (standaard).

Gebruik

Gebruik het kenmerk MousePointer om zichtbaar te maken dat een gebruiker de muisaanwijzer over een bepaald object beweegt. Het ene soort aanwijzer geeft bijvoorbeeld aan dat een object is uitgeschakeld terwijl het andere soort aanwijzer betekent dat het object wacht op invoer van gegevens.

Hieronder ziet u de mogelijke instellingen voor het kenmerk MousePointer:

0 - Standaard	n.v.t.	6 - Omvang NOZW	
1 - Pijl		7 - Omvang Z	
2 - Kruis		8 - Omvang NWZO	
3 - Invoegsymbool		9 - Omvang O	
4 - Pictogram		10 - Pijl omhoog	
5 - Omvang		11 - Wachten	

Voorbeeld

Syntaxis operator NEW:

```
DEFINE BROWSE Blader1 OF THIS
  this.MousePointer=1
  this.Blader1.Top=4
  this.Blader1.Left=3
  this.Blader1.Width=32
  this.Blader1.Height=12
```

Syntaxis commando DEFINE:

```
DEFINE BROWSE Blader1 OF THIS FROM 4,3 TO 16,35;
    PROPERTY MousePointer 1
```

Zie ook

OnMouseMove

Move( )

Met de methode Move( ) wordt een object in het hoofdformulier verplaatst, vergroot en verkleind.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, OLE, RADIOBUTTON, PUSHBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gebruik

Gebruik de methode Move( ) om de positie en de afmetingen van een object te wijzigen.

De methode Move( ) kent de volgende vier parameters:

- *<linkerkolom Nuitdr>* Hiermee wordt de positie van de linkerrand ingesteld, de instelling van het kenmerk Left wordt gewijzigd.
- *<bovenste rij Nuitdr>* Hiermee wordt de positie van de bovenrand ingesteld, de instelling van het kenmerk Top wordt gewijzigd.
- *<breedte Nuitdr>* Hiermee wordt de breedte ingesteld, de instelling van het kenmerk Width wordt gewijzigd.
- *<hoogte Nuitdr>* Hiermee wordt de hoogte ingesteld, de instelling van het kenmerk Height wordt gewijzigd.

Elke eenheid van de waarde die u opgeeft bij de methode Move( ), is gebaseerd op de gemiddelde breedte en hoogte van de tekens in het actieve font.

Voorbeeld

Syntaxis operator NEW:

```
this.KL1 = NEW LISTBOX(THIS)
  this.KL1.DataSource="FIELD Dieren->Naam"
  this.KL1.Top=4
  this.KL1.Left=6
  this.KL1.Width=20
  this.KL1.Height=12
  this.KL1.OnRightMouseDown {;Form.Move(6,10,15,5)}
```

Syntaxis commando DEFINE:

```
DEFINE LISTBOX KL1 OF THIS;
    PROPERTY;
      DataSource "FIELD Dieren->Naam",;
      Top 4,;
      Left 6,;
      Width 20,;
      Height 12,;
      OnRightMouseDown {;Form.Move(6,10,15,5)}
```

Zie ook

Height, Left, ScaleFontName, ScaleFontSize, Top, Width

Moveable

Met het kenmerk Moveable wordt bepaald of een formulier met de muis kan worden verplaatst.

Kenmerk uit klasse

FORM

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Moveable is waar (.T.).

Gebruik

Als u het kenmerk Moveable van een formulier op waar (.T.) instelt, wordt bovenaan het formulier een titelbalk weergegeven. De gebruiker kan in dat geval de muisaanwijzer naar de titelbalk verplaatsen, de linkermuisknop ingedrukt houden en het formulier vervolgens naar een andere positie slepen. Een formulier kan alleen worden verplaatst als het een titelbalk heeft.

Als u het kenmerk MDI van een formulier op waar (.T.) instelt, wordt de instelling van het kenmerk Moveable genegeerd en kan de gebruiker het formulier altijd verplaatsen.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL f
f=NEW InvoerForm()
f.OPEN()
CLASS InvoerForm OF FORM
  this.Moveable = .F.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM InvoerForm;
  Property Moveable .F.
OPEN FORM InvoerForm
```

Zie ook

Left, Sizeable, Top

Multiple

Met het kenmerk Multiple wordt bepaald of meer dan een onderdeel van een keuzelijst kan worden geselecteerd.

Kenmerk uit klasse

LISTBOX

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Multiple is onwaar (.F.).

Gebruik

Als u het kenmerk Multiple op waar (.(.T.).) instelt, kan de gebruiker een willekeurig aantal onderdelen (of geen enkel onderdeel) in de keuzelijst selecteren. Een dergelijke keuzelijst wordt ook wel een *meerkeuzelijst* genoemd. Als u het kenmerk Multiple op onwaar (.F.) instelt, kan de gebruiker slechts één onderdeel uit de keuzelijst selecteren.

Meerkeuzelijsten zijn handig omdat de gebruiker verschillende onderdelen in één keer kan selecteren. In een programma dat voor de personeelsafdeling is geschreven, kan de gebruiker bijvoorbeeld verschillende sollicitanten selecteren voor een gesprek. In het

programma kan de gebruiker dan een of meer sollicitanten selecteren in een meerkeuzelijst (elke geselecteerde prompt wordt voorzien van een vinkje).

Opmerking Gebruik LISTSELECTED( ) of Selected( ) om te evalueren welke prompts zijn gekozen.

Voorbeeld

Syntaxis operator NEW:

```
KL1 = NEW LISTBOX(this)
KL1.DataSource = "FIELD BEDRIJF->BEDRIJF"
KL1.Multiple = .T.
KL1.OnRightMouseDown = Gevinkt
* Meerdere bedrijven kunnen worden gekozen.
```

Syntaxis commando DEFINE:

```
DEFINE LISTBOX KL1 OF THIS;
  PROPERTY;
    DataSource "FIELD BEDRIJF->BEDRIJF,;
    Multiple .T., OnRightMouseDown Gevinkt
PROCEDURE Gevinkt
FOR i=1 TO Form.KL1.Count()
  ? Form.KL1.Selected(i)
NEXT i
RETURN
```

Zie ook

Count( ), LISTSELECTED( ), Selected( )

Name

De instelling van het kenmerk Name kan alleen worden gelezen en wordt gebruikt om de naam van een object op te geven.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, MENU, PUSHBUTTON, RADIOBUTTON, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van Name is de naam die in dBASE aan het object wordt toegewezen wanneer u het maakt. In dBASE wordt bijvoorbeeld automatisch de naam BUTTON1 toegewezen aan het kenmerk Name van de eerste knop die u in Formulierontwerp maakt.

Gebruik

Gebruik het kenmerk Name om een object een naam te geven.

Als u een object maakt met het commando DEFINE, geeft u een naam op het voor object met de parameter <*objectnaam*>. Zie DEFINE voor meer informatie.

Als u een object maakt met de operator NEW, kunt u een naam voor het object opgeven met de tweede (optionele) parameter. Met de volgende commando's wordt bijvoorbeeld een formulier gemaakt en een afbeeldingsobject dat in het formulier moet worden weergegeven:

```
MijnForm = NEW FORM()
MijnAfbeelding = NEW IMAGE(MijnForm, "Afbeelding")
```

Het kenmerk Name van het nieuwe afbeeldingsobject is ingesteld op "Afbeelding".

De instelling van het kenmerk Name kan alleen worden gelezen.

Voorbeeld

Syntaxis operator NEW:

```
XXX = NEW LISTBOX(this)
XXX.DataSource = "FIELD BEDRIJF->Postcode"
XXX.Name = "PC_Lijst"
* PC_Lijst heeft als naam prioriteit over XXX.
```

Syntaxis commando DEFINE:

```
DEFINE LISTBOX XXX OF THIS;
  PROPERTY;
      DataSource "FIELD BEDRIJF->Postcode",;
      Name "PC_Lijst"
```

Zie ook

DEFINE, ID

NextCol( )

Het resultaat van de methode NextCol is het getal van de volgende hoogste kolompositie waarop een object in een formulier kan worden geplaatst.

Kenmerk uit klasse

FORM

Gebruik

Gebruik de methode NextCol( ) in combinatie met de methode NextRow( ) om te voorkomen dat objecten in een formulier elkaar overlappen. Als u bijvoorbeeld een invoervak maakt, is het resultaat van de methode NextCol( ) de coördinaat van de volgende beschikbare kolom en van de methode NextRow( ) de coördinaat van de

volgende beschikbare rij. U kunt op deze coördinaten een nieuw object plaatsen zonder dat dit het invoervak overlapt.

De ruimte tussen de kolomcoördinaten die het resultaat is van de methode NextCol( ), is gelijk aan de gemiddelde breedte van de tekens in het actieve font van het formulier.

Voorbeeld

```
LOCAL f
f=NEW InvoerForm()
f.OPEN()
CLASS InvoerForm of FORM
DEFINE ENTRYFIELD Vnaam OF this;
  PROPERTY;
  Top 3, Left 2, Width 20,;
  Value SPACE(20)
DEFINE ENTRYFIELD Anaam OF this;
  PROPERTY;
  Top 5, Left 2, Width 20,;
  Value SPACE(20)
* NextRow() en NextCol() gebruiken om knop
* te definiëren die vier kolommen rechts van
* invoervak Anaam wordt geplaatst.
DEFINE PUSHBUTTON OpKnp1 OF this;
  PROPERTY Text "Doorgaan",;
  Top this.NextRow()-2,;
  Left this.NextCol()+4,;
  Width 10
ENDCLASS
```

Zie ook

COL( ), NextRow( ), ROW( ), ScaleFontName, ScaleFontSize

NextObj

Het kenmerk NextObj bevat een verwijzing naar het object dat in de tabvolgorde van een hoofdformulier volgt op het huidige object.

Kenmerk uit klasse

FORM

Gegevenstype

Objectverwijzing

Gebruik

Gebruik het kenmerk NextObj om te verwijzen naar het volgende object dat focus krijgt wanneer de gebruiker op *Tab* drukt.

Gebruik het kenmerk NextObj in subroutines van het kenmerk Valid om te controleren of validatie nodig is voordat wordt verdergegaan naar het volgende object. Bij de volgende commando's wordt bijvoorbeeld gecontroleerd of het volgende object een knop met de naam Annuleren is:

```
IF (MijnForm.NextObj.Name = "Annuleren")
    RETURN .T.          && Geen validatie vereist.
ELSE
   RETURN Validatie()  && Validatieroutine
ENDIF
```

Voorbeeld

In het volgende voorbeeld worden de kenmerken ActiveControl en NextObj gebruikt om de naam van het huidige en het volgende invoervak op te vragen wanneer een gebruiker op een formulier rechtsklikt:

```
LOCAL f
f=NEW InvoerForm()
f.OPEN()
CLASS Invoerform OF FORM
  this.View="Bedrijf.DBF"
  this.OnRightMouseDown={;Fcs="Huidig veld: " + ;
    Form.ActiveControl.Name; Form.Txt1.Text=Fcs}
  this.OnRightMouseUp={;Fcs2="Volgend veld: " + ;
    Form.NextObj.Name; Form.Txt2.Text=Fcs2}
DEFINE ENTRYFIELD Bedrijf OF THIS;
  PROPERTY Datalink "Bedrijf->Bedrijf",;
  Top 3, Left 1, Width 20, Name "Bedrijf"
DEFINE ENTRYFIELD Regio OF THIS;
  PROPERTY Datalink "Bedrijf->Regio",;
  Top 5, Left 1, Name "Regio"
DEFINE TEXT Txt1 OF THIS;
  PROPERTY Text " ",;
  Top 8, Left 1, Width 25
DEFINE TEXT Txt2 OF THIS;
  PROPERTY Text " ",;
  Top 9, Left 1, Width 25
ENDCLASS
```

Zie ook

ActiveControl, _curobj, First, SetFocus( )

NextRow( )

Het resultaat van de methode NextRow is het nummer van de volgende hoogste rijpositie waarop een object in een formulier kan worden geplaatst.

Kenmerk uit klasse

Form

Gebruik

Gebruik de methode NextRow( ) in combinatie met de methode NextCol( ) om te voorkomen dat objecten in een formulier elkaar overlappen. Als u bijvoorbeeld een invoervak maakt, is het resultaat van de methode NextRow( ) de coördinaat van de volgende beschikbare rij en van de methode NextCol( ) de coördinaat van de volgende beschikbare kolom. U kunt op deze coördinaten een nieuw object plaatsen zonder dat dit het invoervak overlapt.

De ruimte tussen de kolomcoördinaten die het resultaat is van de methode NextCol( ) is gelijk aan de gemiddelde breedte van de tekens in het actieve font van het formulier.

Voorbeeld

Zie NextCol( ) voor een voorbeeld van het gebruik van NextRow( ).

Zie ook

COL( ), NextRow( ), ROW( )

Notify( )

Met de methode Notify( ) wordt aan een client-toepassing doorgegeven dat in dBASE een onderdeel is gewijzigd.

Kenmerk uit klasse

DDETOPIC

Gebruik

Gebruik Notify( ) in een DDE server-programma om aan de client-toepassing door te geven dat een onderdeel in een server-programma van dBASE is gewijzigd.

In de subroutine OnPoke wordt de methode Notify( ) vaak uitgevoerd wanneer een externe toepassing een Poke-verzoek naar dBASE stuurt. In een applicatie voor gegevensuitwisseling in Quattro Pro kan bijvoorbeeld het commando {POKE} worden gebruikt om een waarde naar dBASE te sturen waarmee de bij het kenmerk OnPoke opgegeven subroutine wordt uitgevoerd. Met deze subroutine kan de waarde in een veld worden ingevoerd waarna de methode Notify( ) wordt uitgevoerd om aan Quattro Pro door te geven dat het veld is gewijzigd.

Bij de methode Notify( ) kan één parameter, *<element>*, worden opgegeven. Gebruik deze parameter om aan de client-toepassing door te geven welk veld, welke variabele of welk array-element is gewijzigd. Als bijvoorbeeld met de subroutine die bij het kenmerk

OnPoke is opgegeven, de waarde in het veld ACHTERNAAM is gewijzigd, kan met de subroutine het volgende commando worden uitgevoerd:

```
MijnDDETopic.Notify("ACHTERNAAM")
```

Met de client-toepassing moet een automatische koppeling tot stand zijn gebracht, anders werkt de methode Notify( ) niet. Zie OnAdvise( ) voor informatie over automatische koppelingen.

Voorbeeld

Zie CLASS DDETOPIC voor een voorbeeld van het gebruik van Notify( ).

Zie ook

Advise( ), Execute( ), Initiate( ), OnNewValue, Peek( ), Poke( ), Server, Terminate( ), TimeOut, Topic, Unadvise( )

OldStyle

Met het kenmerk OldStyle wordt bepaald of een aankruisvakje of een keuzerondje op de standaardwijze van Windows worden getoond of driedimensionaal.

Kenmerk uit klasse

CHECKBOX, ENTRYFIELD, LISTBOX, RADIOBUTTON, RECTANGLE, SPINBOX, TEXT

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van OldStyle is onwaar (.F.).

Gebruik

Gebruik het kenmerk OldStyle om de weergave van een object in te stellen. Als u het kenmerk OldStyle op waar (.T.) instelt, wordt het object weergegeven op de manier zoals in Windows gebruikelijk is. Als het kenmerk OldStyle wordt ingesteld op onwaar (.F.), wordt het object driedimensionaal weergegeven.

Voorbeeld

Syntaxis operator NEW:

```
KruisOud = NEW CHECKBOX(this)
KruisOud.Oldstyle = .T.
```

Syntaxis commando DEFINE:

```
DEFINE CHECKBOX KruisOud OF THIS;
  Property;
    Oldstyle .T.
```

Zie ook

PatternStyle, BorderStyle

OleType

Het resultaat van het kenmerk OleType is een getal dat aangeeft of een OLE-veld leeg is, een ingesloten document bevat of een koppeling met een documentbestand bevat.

Kenmerk uit klasse

OLE

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van OleType is 0 (leeg).

Gebruik

Gebruik het kenmerk OleType om na te gaan of een OLE-veld leeg is, een koppeling met een documentbestand bevat of een ingesloten bestand bevat.

In een OLE-weergave wordt een OLE-document getoond (ook wel een OLE-object genoemd). Een OLE-document kan een grafische afbeelding zijn maar ook een document dat in een tekstverwerker is gemaakt of elk ander gegevensobject dat in een externe toepassing is gemaakt. Een dergelijke externe toepassing wordt een *OLE-server* genoemd (de OLE-server moet een OLE-voorziening hebben).

Een grafische afbeelding die bijvoorbeeld in Paintbrush is gemaakt kan een OLE-document zijn en Paintbrush een OLE-server wanneer u het volgende doet:

- Het document (of een deel daarvan) insluit in een OLE-veld met **Knippen** en **Plakken** uit het menu **Bewerken**.
- Het bestand met het document koppelt aan een OLE-veld met **Knippen** en **Koppeling plakken** uit het menu **Bewerken**.

Als het huidige OLE-veld leeg is, dat wil zeggen, als het geen ingesloten OLE-document en geen koppeling met een OLE-document bevat, is het kenmerk OleType ingesteld op 0. Als het OLE-veld een ingesloten document bevat, is het kenmerk OleType ingesteld

op 1. Als het OLE-veld een koppeling met een documentbestand bevat, is het kenmerk OleType ingesteld op 2.

De instelling van het kenmerk OleType kan alleen worden gelezen.

Voorbeeld

Zie DoVerb( ) voor een voorbeeld van het gebruik van OleType.

Zie ook

LinkFileName, ServerName

OnAdvise

Met het kenmerk OnAdvise word een subroutine uitgevoerd wanneer via een externe toepassing een automatische DDE-koppeling wordt gemaakt met een onderdeel in een onderwerp van een server in dBASE.

Kenmerk uit klasse

DDETOPIC

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnAdvise in een DDE server-programma om te reageren op een poging een automatische koppeling tot stand te brengen en om na te gaan met welke gegevens in dBASE de koppeling tot stand moet worden gebracht (bij een automatische koppeling kunt u met de methode Notify( ) een verandering in de gegevens doorgeven aan de client-toepassing).

Bij het kenmerk OnAdvise moet de parameter *<element>* worden opgegeven, waarmee de gegevens in dBASE worden aangeduid. Het server-programma kan voor deze gegevens elke willekeurige informatie gebruiken, bijvoorbeeld een veld, een variabele of een array-element.

De namen van gegevens worden vaak opgeslagen in tabellen of array-objecten als meerdere automatische koppelingen gemaakt zijn. Met een client-toepassing kan bijvoorbeeld een automatische koppeling worden gemaakt met een veld waarbij de veldnaam wordt doorgegeven door middel van de parameter *<element>*. Elke keer kan met het kenmerk OnAdvise de veldnaam in een array-object worden geplaatst.

Met de subroutine OnPoke kan dit array-object telkens worden doorzocht wanneer een veld wordt gewijzigd. Als de naam van het gewijzigde veld in het array-object wordt gevonden, kan de methode Notify( ) worden uitgevoerd.

Voorbeeld

Zie CLASS DDETOPIC voor een voorbeeld van het gebruik van OnAdvise.

Zie ook

Advise( ), Execute( ), Initiate( ), OnNewValue, Peek( ), Poke( ), Server, Terminate( ), TimeOut, Topic, Unadvise( )

OnAppend

Met het kenmerk OnAppend wordt een subroutine uitgevoerd wanneer een record aan een tabel wordt toegevoegd.

Kenmerk uit klasse

BROWSE, FORM

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnAppend om steeds een handeling te verrichten wanneer een nieuw record wordt toegevoegd.

Met het kenmerk OnAppend kunt u uw applicatie laten reageren wanneer nieuwe informatie aan een tabel wordt toegevoegd. In een bladerobject kan het kenmerk OnAppend bijvoorbeeld worden gebruikt om een logisch veld met de naam NIEUWVELD op waar (.T.) in te stellen wanneer een gebruiker een nieuw record toevoegt.

Voorbeeld

Syntaxis operator NEW:

```
Blader = NEW BROWSE(this)
Blader.text = "Voorbeeld voor OnAppend"
Blader.OnAppend = DataCheck
```

Syntaxis commando DEFINE:

```
DEFINE BROWSE Blader OF THIS;
  PROPERTY;
  Text  "Voorbeeld voor OnAppend",;
  OnAppend DataCheck
PROCEDURE DataCheck
IF LEN(Klanten->Naam) = 0
  ? "U moet een naam opgeven"
ENDIF
```

Zie ook

OnChange, OnNavigate

OnChange

Met het kenmerk OnChange wordt een subroutine uitgevoerd wanneer de gebruiker de waarde wijzigt die in een object wordt weergegeven.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, ENTRYFIELD, LISTBOX, OLE, RADIOBUTTON, SCROLLBAR, SPINBOX

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnChange om een subroutine uit te voeren wanneer de gebruiker een van de volgende handelingen verricht:

- Een waarde in een veld wijzigen en naar een andere rij in een bladerobject gaan
- Een kruis in een aankruisvakje plaatsen of uit het aankruisvakje verwijderen
- Een waarde in een invoervak wijzigen
- Een waarde in het tekstdeel van een keuzelijst met invoervak of een ringveld wijzigen
- Een ander keuzerondje selecteren
- Het schuifblokje in een schuifbalk verplaatsen

De subroutine OnChange van een OLE-object wordt telkens uitgevoerd wanneer de recordaanwijzer wordt verplaatst.

In een applicatie kan een gebruiker bijvoorbeeld het SOFI-nummer van een klant wijzigen met een invoervak. Om de wijziging te controleren kunt u een functie toewijzen aan het kenmerk OnChange waarmee een dialoogvenster wordt geopend waarin de gebruiker wordt gevraagd de wijziging te bevestigen.

Net zoals bij de andere actiekenmerken kunt u bij het kenmerk OnChange het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnChange** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
Bedrijf = NEW EntryField(this)
Bedrijf.Top = 3
Bedrijf.Left = 1
Bedrijf.Datalink = "Bedrijf->Bedrijf"
Bedrijf.OnChange = CntrWzg
```

Syntaxis commando DEFINE:

```
DEFINE EntryField Bedrijf OF THIS;
  PROPERTY Top 3, Left 1, ;
  DataLink "Bedrijf->Bedrijf",;
  OnChange CntrWzg
```

Zie ook

OnAppend, OnNavigate

OnClick

Met het kenmerk OnClick kan een subroutine worden uitgevoerd wanneer een gebruiker op een knop of een menu-opdracht klikt.

Kenmerk uit klasse

MENU, PUSHBUTTON

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnClick om een handeling toe te wijzen aan een knop of een menu-opdracht. Bij een Sluiten-knop kan bijvoorbeeld een functie aan het kenmerk OnClick zijn toegewezen waarmee een programma wordt beëindigd.

Net zoals bij de andere actiekenmerken kunt u bij het kenmerk OnClick het volgende opgeven:

- Functie
- Procedure

- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnClick** in het kenmerkenvenster te klikken.

De subroutine die u bij het kenmerk OnClick opgeeft, wordt alleen in de volgende gevallen uitgevoerd:

- Het kenmerk Enabled van de knop is ingesteld op waar (.T.)
- Het kenmerk When van de knop resulteert in waar (.T.)

Voorbeeld

Syntaxis operator NEW:

```
Vlg = NEW PUSHBUTTON(this)
Vlg.Text = "Volgende invoer "
Vlg.OnClick = {;SKIP}
```

Syntaxis commando DEFINE:

```
DEFINE PUSHBUTTON Vlg OF THIS;
  Property;
    Text "Volgende invoer",;
    Onclick {;SKIP}
```

Zie ook

OnLeftMouseDblClick, OnLeftMouseDown, OnLeftMouseUp, OnMiddleDblClick, OnMiddleMouseDown, OnMiddleMouseUp, OnRightMouseDblClick, OnRightMouseDown, OnRightMouseUp, OnSelection

OnClose

Met het kenmerk OnClose wordt een subroutine uitgevoerd wanneer een formulier wordt gesloten.

Kenmerk uit klasse

FORM, OLE

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnClose om automatisch een handeling uit te voeren wanneer een formulier wordt gesloten. Met het kenmerk OnClose kunt u bijvoorbeeld het bureaublad opschonen door procedurebestanden te sluiten, instellingen te herstellen, tabellen te sluiten en objecten vrij te geven.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnClose het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Voordat de subroutine die u bij het kenmerk OnClose hebt opgegeven, wordt uitgevoerd, gebeurt het volgende:

1 De subroutine van het kenmerk Valid (indien gedefinieerd) van het object met invoerfocus wordt uitgevoerd. Als het resultaat de waarde onwaar (.F.) is, wordt het formulier niet gesloten.

2 De subroutine van het kenmerk OnLostFocus (indien gedefinieerd) van het object met invoerfocus wordt uitgevoerd.

3 De subroutine van het kenmerk OnLostFocus (indien gedefinieerd) van het formulier wordt uitgevoerd.

Nadat de bij het kenmerk OnClose opgegeven subroutine is uitgevoerd, worden het formulier, de objecten en alle subformulieren gesloten.

De subroutine OnClose van een OLE-object wordt uitgevoerd zodra het hoofdformulier wordt gesloten.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnClose** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL f
f=NEW InvoerForm()
f.OPEN()
CLASS InvoerForm OF FORM
   this.OnClose SluitAlles  && Functie waarmee tabellen
                            && en andere programma-onderdelen
                            && worden gesloten.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM InvoerForm;
  PROPERTY OnClose SluitAlles
```

Zie ook

OnChange, OnClick, OnGotFocus, OnHelp, OnLostFocus, OnMove, OnOpen, OnSize

OnExecute

Met het kenmerk OnExecute wordt een subroutine uitgevoerd wanneer een client-toepassing een commandoreeks naar een DDE server-programma wordt gestuurd.

Kenmerk uit klasse

DDETOPIC

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnExecute om een handeling uit te voeren wanneer vanuit een externe toepassing (de *client-toepassing*) een instructie naar dBASE voor Windows wordt gestuurd. Deze instructie kan bestaan uit een willekeurige tekenreeks.

De parameter <*cmd*> wordt naar het kenmerk OnExecute gestuurd. Deze parameter bevat de instructie die van de client-toepassing afkomstig is. Als de instructie een dBASE-commando is, kan deze met de subroutine worden uitgevoerd. De client-toepassing kan bijvoorbeeld de tekenreeks "CLOSE DATABASES" sturen. In de subroutine wordt dit commando uitgevoerd door middel van de macrovervangingsfunctie:

```
&Cmd
```

In dit geval zorgt de macrovervanging (&) ervoor, dat de inhoud van de parameter *Cmd* in dBASE wordt beschouwd als een commando en niet als een tekenreeks.

Als de instructie geen commando is, kan deze in de subroutine bij sprongopdrachten worden gebruikt. In een routine voor de effectenhandel kunnen bijvoorbeeld twee tekenreeksen worden ontvangen: "BUY" en "SELL". In de routine kan door middel van een IF...ELSE...ENDIF-opdracht in elk geval een andere procedure worden gestart.

Zie CLASS DDETOPIC en Hoofdstuk 26 in *Programmeren* voor meer informatie over de objectklasse DDETopic.

Voorbeeld

Zie CLASS DDETOPIC voor een voorbeeld van het gebruik van OnExecute.

Zie ook

Advise( ), Execute( ), Initiate( ), OnNewValue, OnPeek, OnPoke, Peek( ), Poke( ), Server, Terminate( ), TimeOut, Topic, Unadvise( )

OnGotFocus

Met het kenmerk OnGotFocus wordt een subroutine uitgevoerd wanneer een object focus krijgt.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, LISTBOX, PUSHBUTTON, RADIOBUTTON, SCROLLBAR, SPINBOX

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnGotFocus om automatisch een handeling uit te voeren wanneer een object wordt geselecteerd. Bij een bladerobject waarin geheime informatie wordt getoond, kan het kenmerk OnGotFocus bijvoorbeeld worden gebruikt om een dialoogvenster weer te geven waarin een wachtwoord moet worden ingevoerd. Als de gebruiker niet het juiste wachtwoord typt, kan vervolgens de focus worden verplaatst naar een ander object.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnGotFocus het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnGotFocus** in het kenmerkenvenster te klikken.

Het kenmerk OnGotFocus verschilt van het kenmerk When, waarbij een voorwaarde wordt opgegeven die moet resulteren in waar (.T.) voordat een object focus krijgt.

Voorbeeld

Syntaxis operator NEW:

```
Bedrijf = NEW ENTRYFIELD(this)
Bedrijf.Top = 3
Bedrijf.Left = 1
Bedrijf.Datalink = "Bedrijf"
Bedrijf.OnGotFocus = KlaarStaan
```

Syntaxis commando DEFINE:

```
DEFINE Entryfield Bedrijf OF THIS;
  Property;
  Top 3, Left 1,;
  Datalink "Bedrijf",;
  OnGotFocus KlaarStaan
```

Zie ook

OnClick, OnClose, OnHelp, OnLostFocus, OnMove, OnOpen, OnSize

OnHelp

Met het kenmerk OnHelp wordt een subroutine uitgevoerd wanneer de gebruiker op *F1* drukt terwijl een object focus heeft.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, LISTBOX, MENU, PUSHBUTTON, RADIOBUTTON, SCROLLBAR, SPINBOX

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnHelp om een dBASE-routine te starten (in plaats van een Help-onderwerp in een Help-bestand van Windows op te vragen) wanneer de gebruiker op *F1* drukt. In een programma voor personeelszaken kunnen gebruikers bijvoorbeeld namen van werknemers opzoeken in een keuzelijst met invoervak. Met het kenmerk OnHelp kan dan een dialoogvenster worden geopend waarin wordt uitgelegd dat namen snel kunnen worden gevonden door letters te typen in plaatsen van door de keuzelijst met invoervak te bladeren.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnHelp het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnHelp** in het kenmerkenvenster te klikken.

De subroutine die u bij het kenmerk OnHelp opgeeft, heeft prioriteit boven het gecompileerde Help-bestand dat is opgegeven bij de kenmerken HelpID en HelpFile.

Voorbeeld

Syntaxis operator NEW:

```
Hervat = New EDITOR(this)
Hervat.Top = 3
Hervat.Left = 1
Hervat.OnHelp = BewerkenHelp
```

Syntaxis commando DEFINE:

```
DEFINE EDITOR Hervat OF THIS;
  AT 3,1;
  Property;
    OnHelp BewerkenHelp
```

Zie ook

HELP, HelpFile, OnChange, OnClick, OnClose, OnGotFocus, OnLostFocus, OnMove, OnOpen, OnSize

OnLeftDblClick

Met het kenmerk OnLeftDblClick wordt een subroutine uitgevoerd wanneer de gebruiker dubbelklikt op een formulier of object.

Property of class

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Data type

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnLeftDblClick om een handeling uit te voeren wanneer de gebruiker met de linkermuisknop dubbelklikt. Met het kenmerk OnLeftDblClick kan ook worden geconstateerd of de gebruiker *Shift, Ctrl,* de middelste muisknop of de rechtermuisknop ingedrukt houdt wanneer hij of zij dubbelklikt.

Met het kenmerk OnLeftDblClick worden de volgende drie parameters aan de bijbehorende subroutine doorgegeven:

- *<Vlaggen>* Dit is een uit één byte bestaande waarde waaraan u kunt zien welke andere toetsen en muisknoppen werden ingedrukt toen de gebruiker dubbelklikte.

- <*Kolom*> Dit is de horizontale positie van de muis op het moment dat de gebruiker dubbelklikte.
- <*Rij*> Dit is de verticale positie van de muis op het moment dat de gebruiker dubbelklikte.

Parameter <*vlaggen*>

Desgewenst kunt u de parameter *Vlaggen* gebruiken in combinatie met de functie BITSET( ) om verschillende combinaties van toetsen en muisknoppen te herkennen. Bij de functie BITSET( ) wordt een numerieke waarde geëvalueerd en het resultaat is de logische waarde waar (.T.) of onwaar (.F.), afhankelijk van het feit of een bepaald bit in de waarde aan (1) of uit (0) is. U moet de volgende twee parameters opgeven bij de functie BITSET( ):

- De waarde van de parameter <*Vlaggen*> zelf.
- Het bit in de parameter <*Vlaggen*> dat moet worden geëvalueerd.

Hieronder vindt u de waarden die u via de tweede parameter bij de functie BITSET( ) kunt opgeven:

Tegelijkertijd ingedrukte knop of /toets	Tweede parameter
Rechtermuisknop	1
Shift	2
Ctrl	3
Middelste muisknop	4

Als u bijvoorbeeld wilt controleren of de gebruiker *Shift* ingedrukt heeft gehouden terwijl hij of zij op de linkmuisknop drukte en de aanwijzer op een object stond, gebruikt u BITSET(Vlaggen, 2). Als dat het geval is, resulteert de functie BITSET( ) in waar (.T.). De functies BITSET(Vlaggen, 3) en BITSET(Vlaggen, 1) resulteren beide in waar (.T.), als *Ctrl* en de rechtermuisknop werden ingedrukt toen de gebruiker dubbelklikte. U gebruikt deze informatie in opdrachten als IF...ELSE...ENDIF of DO CASE...ENDCASE om verschillende handelingen toe te wijzen aan verschillende combinaties van toetsen en muisknoppen.

Parameters <*Kolom*> en <*Rij*>

Met de parameters <*Kolom*> en <*Rij*> wordt aangegeven waar de gebruiker heeft geklikt. Op basis daarvan kunt u verschillende handelingen uitvoeren. Met het kenmerk OnLeftDblClick van een formulier kunnen bijvoorbeeld verschillende handelingen worden uitgevoerd, afhankelijk van de plaats waar de gebruiker dubbelklikt. De parameters <*Kolom*> en <*Rij*> worden uitgedrukt in tekeneenheden.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnLeftDblClick het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Prodecure-editor door op de hulpmiddelenknop naast de optie **OnLeftDblClick** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL f
f=NEW InvoerForm
f.OPEN()
CLASS InvoerForm OF FORM
   this.OnLeftDblClick = SelecteerInvoer
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM InvoerForm;
  Property OnLeftDblClick SelecteerInvoer
```

Zie ook

BITSET( ), OnLeftMouseDown, OnLeftMouseUp, OnMiddleDblClick, OnMiddleMouseDown, OnMiddleMouseUp, OnRightMouseDblClick, OnRightMouseDown, OnRightMouseUp

OnLeftMouseDown

Met het kenmerk OnLeftMouseDown wordt een subroutine uitgevoerd wanneer de gebruiker op de linkermuisknop drukt terwijl de aanwijzer op een formulier of object staat.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnLeftMouseDown om een handeling uit te voeren wanneer de gebruiker de linkermuisknop indrukt. Met het kenmerk OnLeftMouseDown kan ook worden geconstateerd of de gebruiker *Shift, Ctrl,* de middelste muisknop of de rechtermuisknop ingedrukt houdt wanneer hij of zij op de linkermuisknop drukt.

Met het kenmerk OnLeftMouseDown worden de volgende drie parameters aan de bijbehorende subroutine doorgegeven:

- <*Vlaggen*> Dit is een uit één byte bestaande waarde waaraan u kunt zien welke andere toetsen en muisknoppen werden ingedrukt toen de gebruiker de linkermuisknop indrukte.
- <*Kolom*> Dit is de horizontale positie van de muis op het moment dat de gebruiker op de linkermuisknop indrukte.
- <*Rij*> Dit is de verticale positie van de muis op het moment dat de gebruiker op de linkermuisknop drukte.

Parameter <*Vlaggen*>

Desgewenst kunt u de parameter *Vlaggen* gebruiken in combinatie met de functie BITSET( ) om verschillende combinaties van toetsen en muisknoppen te herkennen. Bij de functie BITSET( ) wordt een numerieke waarde geëvalueerd en het resultaat is de logische waarde waar (.T.) of onwaar (.F.), afhankelijk van het feit of een bepaald bit in de waarde aan (1) of uit (0) is. U moet de volgende twee parameters opgeven bij de functie BITSET( ):

- De waarde van de parameter <*Vlaggen*> zelf.
- Het bit in de parameter <*Vlaggen*> dat moet worden geëvalueerd.

Hieronder vindt u de waarden die u via de tweede parameter bij de functie BITSET( ) kunt opgeven:

Tegelijkertijd ingedrukte knop of toets	Tweede parameter
Rechtermuisknop	1
Shift	2
Ctrl	3
Middelste muisknop	4

Als u bijvoorbeeld wilt controleren of de gebruiker *Shift* ingedrukt heeft gehouden terwijl hij of zij op de linkermuisknop drukte en de aanwijzer op een object stond, gebruikt u BITSET(Vlaggen, 2). Als dat het geval is, resulteert de functie BITSET( ) in waar (.T.). De functies BITSET(Vlaggen, 3) en BITSET(Vlaggen, 1) resulteren beide in waar (.T.), als *Ctrl* en de rechtermuisknop werden ingedrukt toen de gebruiker op de linkermuisknop drukte. U gebruikt deze informatie in opdrachten als IF...ELSE...ENDIF of DO CASE...ENDCASE om verschillende handelingen toe te wijzen aan verschillende combinaties van toetsen en muisknoppen.

Parameters <*Kolom*> en <*Rij*>

Met de parameters <*Kolom*> en <*Rij*> wordt aangegeven waar de gebruiker heeft geklikt. Op basis daarvan kunt u verschillende handelingen uitvoeren. Met het kenmerk OnLeftMouseDown van een formulier kunnen bijvoorbeeld afhankelijk van de plaats waar de gebruiker klikt verschillende handelingen worden uitgevoerd. De parameters <*Kolom*> en <*Rij*> worden uitgedrukt in tekeneenheden.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnLeftMouseDown het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnLeftMousDown** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL f
f=NEW InvoerForm()
f.OPEN()
CLASS InvoerForm OF FORM
  this.OnLeftMouseDown = VerplaatsOk
  this.OnLeftMouseUp = VerplaatsStop
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM InvoerForm;
  PROPERTY OnLeftMouseDown VerplaatsOk,;
    OnLeftMouseUp VerplaatsStop
OPEN FORM InvoerForm
```

Zie ook

BITSET( ), OnLeftMouseDblClick, OnLeftMouseUp, OnMiddleDblClick, OnMiddleMouseDown, OnMiddleMouseUp, OnRightMouseDblClick, OnRightMouseDown, OnRightMouseUp

OnLeftMouseUp

Met het kenmerk OnLeftMouseUp wordt een subroutine uitgevoerd wanneer de gebruiker de linkermuisknop loslaat terwijl de aanwijzer op een formulier of object staat.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnLeftMouseUp om een handeling uit te voeren wanneer de gebruiker de linkermuisknop loslaat. Met het kenmerk OnLeftMouseUp kan ook worden geconstateerd of de gebruiker *Shift, Ctrl,* de middelste muisknop of de rechtermuisknop ingedrukt houdt wanneer hij of zij de linkermuisknop loslaat.

Met het kenmerk OnLeftMouseUp worden de volgende drie parameters aan de bijbehorende subroutine doorgegeven:

- *<Vlaggen>* Dit is een uit één byte bestaande waarde waaraan u kunt zien welke andere toetsen en muisknoppen werden ingedrukt toen de gebruiker de linkermuisknop losliet.
- *<Kolom>* Dit is de horizontale positie van de muis op het moment dat de gebruiker op de linkermuisknop losliet.
- *<Rij>* Dit is de verticale positie van de muis op het moment dat de gebruiker de linkermuisknop losliet.

Parameter *<Vlaggen>*

Desgewenst kunt u de parameter *Vlaggen* gebruiken in combinatie met de functie BITSET( ) om verschillende combinaties van toetsen en muisknoppen te herkennen. Bij de functie BITSET( ) wordt een numerieke waarde geëvalueerd en het resultaat is de logische waarde waar (.T.) of onwaar (.F.), afhankelijk van het feit of een bepaald bit in de waarde aan (1) of uit (0) is. U moet de volgende twee parameters opgeven bij de functie BITSET( ):

- De waarde van de parameter *<Vlaggen>* zelf.
- Het bit in de parameter *<Vlaggen>* dat moet worden geëvalueerd.

Hieronder vindt u de waarden die u via de tweede parameter bij de functie BITSET( ) kunt opgeven:

Tegelijkertijd ingedrukte knop of toets	Tweede parameter
Rechtermuisknop	1
Shift	2
Ctrl	3
Middelste muisknop	4

Als u bijvoorbeeld wilt controleren of de gebruiker *Shift* ingedrukt heeft gehouden terwijl hij of zij de linkermuisknop losliet, gebruikt u BITSET(Vlaggen, 2). Als dat het geval is, resulteert de functie BITSET( ) in waar (.T.). De functies BITSET(Vlaggen, 3) en BITSET(Vlaggen, 1) resulteren beide in waar (.T.), als *Ctrl* en de rechtermuisknop werden ingedrukt toen de gebruiker de linkermuisknop losliet. U gebruikt deze informatie in opdrachten als IF...ELSE...ENDIF of DO CASE...ENDCASE om verschillende handelingen toe te wijzen aan verschillende combinaties van toetsen en muisknoppen.

Parameters *<Kolom>* en *<Rij>*

Met de parameters <*Kolom*> en <*Rij*> wordt aangegeven waar de gebruiker de linkermuisknop heeft losgelaten. Op basis daarvan kunt u verschillende handelingen uitvoeren. Met het kenmerk OnLeftMouseUp van een formulier kunnen bijvoorbeeld afhankelijk van de plaats waar de gebruiker de linkermuisknop loslaat, verschillende handelingen worden uitgevoerd. De parameters <*Kolom*> en <*Rij*> worden uitgedrukt in tekeneenheden.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnLeftMouseUp het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnLeftMouseUp** in het kenmerkenvenster te klikken.

Voorbeeld

Zie OnLeftMouseDown voor een voorbeeld van het gebruik van OnLeftMouseUp.

Zie ook

BITSET( ), OnLeftMouseDblClick, OnLeftMouseDown, OnMiddleDblClick, OnMiddleMouseDown, OnMiddleMouseUp, OnRightMouseDblClick, OnRightMouseDown, OnRightMouseUp

OnLostFocus

Met het kenmerk OnLostFocus wordt een subroutine uitgevoerd wanneer de focus van een object wordt verwijderd.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, LISTBOX, OLE, PUSHBUTTON, RADIOBUTTON, SCROLLBAR, SPINBOX

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnLostFocus om een handeling uit te voeren wanneer de focus van een object wordt verwijderd. Een invoervak kan bijvoorbeeld toegang geven tot een belangrijk tabelveld waarin geldige gegevens moeten worden ingevoerd. Wanneer de gebruiker de focus wil verplaatsen naar een ander object, kan met het kenmerk OnLostFocus een dialoogvenster worden geopend met de vraag "Weet u het zeker?" en twee knoppen met het opschrift "Ja" en "Nogmaals".

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnLostFocus het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnLostFocus** in het kenmerkenvenster te klikken.

Het kenmerk OnLostFocus verschilt van het kenmerk Valid, waarbij een voorwaarde wordt opgegeven die in waar (.T.) moet resulteren voordat de focus van het object kan worden verwijderd.

Voorbeeld

Syntaxis operator NEW:

```
Ring1 = NEW SPINBOX(this)
Ring1.Datalink = "BTW-tarief"
Ring1.Top = 2
Ring1.Left = 4
Ring1.Height = 2
Ring1.OnLostFocus = Herberekening
* Herberekening is FUNCTIE waarmee handeling wordt uitgevoerd
* en die resulteert in .T. of waarde.
```

Syntaxis commando DEFINE:

```
DEFINE SPINBOX Ring1 OF THIS;
  PROPERTY Datalink "BTW-tarief",;
  Top 2, Left 4, Height 2,;
  OnLostFocus Herberekening
```

Zie ook

OnChange, OnClick, OnClose, OnGotFocus, OnHelp, OnMove, OnOpen, OnSize

OnMiddleDblClick

Met het kenmerk OnMiddleDblClick wordt een subroutine uitgevoerd wanneer de gebruiker met de middelste muisknop dubbelklikt terwijl de aanwijzer op een formulier of object staat.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnMiddleDblClick om een handeling uit te voeren wanneer de gebruiker met de middelste muisknop dubbelklikt. Met het kenmerk OnMiddleDblClick kan ook worden geconstateerd of de gebruiker bij het dubbelklikken *Shift, Ctrl,* de linker- of de rechtermuisknop ingedrukt houdt.

Met het kenmerk OnMiddleDblClick worden de volgende drie parameters aan de bijbehorende subroutine doorgegeven:

- *<Vlaggen>* Dit is een uit één byte bestaande waarde waaraan u kunt zien welke andere toetsen en muisknoppen werden ingedrukt toen de gebruiker dubbelklikte.
- *<Kolom>* Dit is de horizontale positie van de muis op het moment dat de gebruiker dubbelklikte.
- *<Rij>* Dit is de verticale positie van de muis op het moment dat de gebruiker dubbelklikte.

Parameter *<Vlaggen>*

Desgewenst kunt u de parameter *Vlaggen* gebruiken in combinatie met de functie BITSET( ) om verschillende combinaties van toetsen en muisknoppen te herkennen. Bij de functie BITSET( ) wordt een numerieke waarde geëvalueerd en het resultaat is de logische waarde waar (.T.) of onwaar (.F.), afhankelijk van het feit of een bepaald bit in de waarde aan (1) of uit (0) is. U moet de volgende twee parameters opgeven bij de functie BITSET( ):

- De waarde van de parameter *<Vlaggen>* zelf.
- Het bit in de parameter *<Vlaggen>* dat moet worden geëvalueerd.

Hieronder vindt u de waarden die u via de tweede parameter bij de functie BITSET( ) kunt opgeven:

Tegelijkertijd ingedrukte knop of toets	Tweede parameter
Linkermuisknop	0
Rechtermuisknop	1
Shift	2
Ctrl	3

Als u bijvoorbeeld wilt controleren of de gebruiker *Shift* ingedrukt heeft gehouden terwijl hij of zij op een object dubbelklikte, gebruikt u BITSET(Vlaggen, 2). Als dat het geval is, resulteert de functie BITSET( ) in waar (.T.). De functies BITSET(Vlaggen, 3) en BITSET(Vlaggen, 0) resulteren beide in waar (.T.), als *Ctrl* en de linkermuisknop werden ingedrukt toen de gebruiker met de middelste muisknop dubbelklikte. U gebruikt deze informatie in opdrachten als IF...ELSE...ENDIF of DO CASE...ENDCASE om verschillende handelingen toe te wijzen aan verschillende combinaties van toetsen en muisknoppen.

Parameters <*Kolom*> en <*Rij*>

Met de parameters <*Kolom*> en <*Rij*> wordt aangegeven waar de gebruiker met de middenmuisknop dubbelklikte. Op basis daarvan kunt u verschillende handelingen uitvoeren. Met het kenmerk OnMiddleDblClick van een formulier kunnen bijvoorbeeld afhankelijk van de plaats waar de gebruiker dubbelklikt loslaat, verschillende handelingen worden uitgevoerd. De parameters <*Kolom*> en <*Rij*> worden uitgedrukt in tekeneenheden.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnMiddleDblClick het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnMiddleDblClick** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL F1
F1=NEW Reizen()
F1.OPEN()
CLASS Reizen OF FORM
  this.Text = "Reisschema "
  this.Top = 2
```

```
    this.Left = 2
    this.Width = 38
    this.Height = 18
    this.OnMiddleDblClick = Dierenrijk
  * Dierenrijk is FUNCTIE waarmee handeling wordt uitgevoerd
  * en die resulteert in .T. of waarde.
  ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM Reizen;
  PROPERTY Text "Reisschema ",;
  Top 2, Left 2, Width 38, Height 18
  OnMiddleDblClick Dierenrijk
OPEN FORM Reizen
```

Zie ook

BITSET( ), OnLeftMouseDblClick, OnLeftMouseDown, OnLeftMouseUp, OnMiddleMouseDown, OnMiddleMouseUp, OnRightMouseDblClick, OnRightMouseDown, OnRightMouseUp

OnMiddleMouseDown

Met het kenmerk OnMiddleMouseDown wordt een subroutine uitgevoerd wanneer de gebruiker op de middelste muisknop drukt terwijl de aanwijzer op een formulier of object staat.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnMiddleMouseDown om een handeling uit te voeren wanneer de gebruiker op de middelste muisknop drukt. Met het kenmerk OnMiddleMouseDown kan ook worden geconstateerd of de gebruiker bij het indrukken *Shift, Ctrl,* de linker- of de rechtermuisknop ingedrukt houdt.

Met het kenmerk OnMiddleMouseDown worden de volgende drie parameters aan de bijbehorende subroutine doorgegeven:

- *<Vlaggen>* Dit is een uit één byte bestaande waarde waaraan u kunt zien welke andere toetsen en muisknoppen werden ingedrukt toen de gebruiker de middelste muisknop indrukte.

- <*Kolom*> Dit is de horizontale positie van de muis op het moment dat de gebruiker de middelste muisknop indrukte.
- <*Rij*> Dit is de verticale positie van de muis op het moment dat de gebruiker de middelste muisknop indrukte.

Parameter <*Vlaggen*>

Desgewenst kunt u de parameter *Vlaggen* gebruiken in combinatie met de functie BITSET( ) om verschillende combinaties van toetsen en muisknoppen te herkennen. Bij de functie BITSET( ) wordt een numerieke waarde geëvalueerd en het resultaat is de logische waarde waar (.T.) of onwaar (.F.), afhankelijk van het feit of een bepaald bit in de waarde aan (1) of uit (0) is. U moet de volgende twee parameters opgeven bij de functie BITSET( ):

- De waarde van de parameter <*Vlaggen*> zelf.
- Het bit in de parameter <*Vlaggen*> dat moet worden geëvalueerd.

Hieronder vindt u de waarden die u via de tweede parameter bij de functie BITSET( ) kunt opgeven:

Tegelijkertijd ingedrukte knop of toets	Tweede parameter
Linkermuisknop	0
Rechtermuisknop	1
Shift	2
Ctrl	3

Als u bijvoorbeeld wilt controleren of de gebruiker *Shift* ingedrukt heeft gehouden terwijl hij of zij de middelste muisknop ingedrukt hield, gebruikt u BITSET(Vlaggen, 2). Als dat het geval is, resulteert de functie BITSET( ) in waar (.T.). De functies BITSET(Vlaggen, 3) en BITSET(Vlaggen, 0) resulteren beide in waar (.T.), als *Ctrl* en de linkermuisknop werden ingedrukt toen de gebruiker op de middelste muisknop drukte. U gebruikt deze informatie in opdrachten als IF...ELSE...ENDIF of DO CASE...ENDCASE om verschillende handelingen toe te wijzen aan verschillende combinaties van toetsen en muisknoppen.

Parameters <*Kolom*> en <*Rij*>

Met de parameters <*Kolom*> en <*Rij*> wordt aangegeven waar de gebruiker op de middelste muisknop drukte. Op basis daarvan kunt u verschillende handelingen uitvoeren. Met het kenmerk OnMiddleMouseDown van een formulier kunnen bijvoorbeeld afhankelijk van de plaats waar de gebruiker dubbelklikt verschillende handelingen worden uitgevoerd. De parameters <*Kolom*> en <*Rij*> worden uitgedrukt in tekeneenheden.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnMiddleMouseDown het volgende opgeven:

- Functie
- Procedure

- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnMiddleMouseDown** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL F1
F1=NEW Reizen()
F1.OPEN()
CLASS Reizen OF FORM
  this.Text = "Reisschema"
  this.Top = 2
  this.Left = 2
  this.Width = 38
  this.Height = 18
  this.OnMiddleMouseDown = Dierenrijk
* Dierenrijk is FUNCTIE waarmee handeling wordt uitgevoerd
* en die resulteert in .T. of waarde.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM Reizen FROM 2,2 TO 20,40;
  PROPERTY Text "Reisschema ",;
  OnMiddleMouseDown Dierenrijk
OPEN FORM Reizen
```

Zie ook

BITSET( ), OnLeftMouseDblClick, OnLeftMouseDown, OnLeftMouseUp, OnMiddleDblClick, OnMiddleMouseUp, OnRightMouseDblClick, OnRightMouseDown, OnRightMouseUp

OnMiddleMouseUp

Met het kenmerk OnMiddleMouseUp wordt een subroutine uitgevoerd wanneer de gebruiker de middelste muisknop loslaat terwijl de aanwijzer op een formulier of object staat.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnMiddleMouseUp om een handeling uit te voeren wanneer de gebruiker de middelste muisknop loslaat. Met het kenmerk OnMiddleMouseUp kan ook worden geconstateerd of de gebruiker bij het loslaten *Shift*, *Ctrl*, de linker- of de rechtermuisknop ingedrukt houdt.

Met het kenmerk OnMiddleMouseUp worden de volgende drie parameters aan de bijbehorende subroutine doorgegeven:

- <*Vlaggen*> Dit is een uit één byte bestaande waarde waaraan u kunt zien welke andere toetsen en muisknoppen werden ingedrukt toen de gebruiker de middelste muisknop losliet.
- <*Kolom*> Dit is de horizontale positie van de muis op het moment dat de gebruiker de middelste muisknop losliet.
- <*Rij*> Dit is de verticale positie van de muis op het moment dat de gebruiker de middelste muisknop losliet.

Parameter <*Vlaggen*>

Desgewenst kunt u de parameter *Vlaggen* gebruiken in combinatie met de functie BITSET( ) om verschillende combinaties van toetsen en muisknoppen te herkennen. Bij de functie BITSET( ) wordt een numerieke waarde geëvalueerd en het resultaat is de logische waarde waar (.T.) of onwaar (.F.), afhankelijk van het feit of een bepaald bit in de waarde aan (1) of uit (0) is. U moet de volgende twee parameters opgeven bij de functie BITSET( ):

- De waarde van de parameter <*Vlaggen*> zelf.
- Het bit in de parameter <*Vlaggen*> dat moet worden geëvalueerd.

Hieronder vindt u de waarden die u via de tweede parameter bij de functie BITSET( ) kunt opgeven:

Tegelijkertijd ingedrukte knop of toets	Tweede parameter
Linkermuisknop	0
Rechtermuisknop	1
Shift	2
Ctrl	3

Als u bijvoorbeeld wilt controleren of de gebruiker *Shift* ingedrukt heeft gehouden terwijl hij of zij de middelste muisknop losliet, gebruikt u BITSET(Vlaggen, 2). Als dat het geval is, resulteert de functie BITSET( ) in waar (.T.). De functies BITSET(Vlaggen, 3) en BITSET(Vlaggen, 0) resulteren beide in waar (.T.), als *Ctrl* en de linkermuisknop werden ingedrukt toen de gebruiker de middelste muisknop losliet. U gebruikt deze informatie in opdrachten als IF...ELSE...ENDIF of DO CASE...ENDCASE om

verschillende handelingen toe te wijzen aan verschillende combinaties van toetsen en muisknoppen.

Parameters <*Kolom*> en <*Rij*>

Met de parameters <*Kolom*> en <*Rij*> wordt aangegeven waar de gebruiker de middelste muisknop los heeft gelaten. Op basis daarvan kunt u verschillende handelingen uitvoeren. Met het kenmerk OnMiddleMouseUp van een formulier kunnen bijvoorbeeld afhankelijk van de plaats waar de gebruiker de muisknop heeft losgelaten verschillende handelingen worden uitgevoerd. De parameters <*Kolom*> en <*Rij*> worden uitgedrukt in tekeneenheden.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnMiddleMouseUp het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnMiddleMouseUp** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL F1
F1=NEW Reizen()
F1.OPEN()
CLASS Reizen OF FORM
  this.Text = "Reisschema"
  this.Top = 2
  this.Left = 2
  this.Width = 38
  this.Height = 18
  this.OnMiddleMouseUp = KostSchema
* KostSchema is FUNCTIE waarmee handeling wordt uitgevoerd
* en die resulteert in .T. of waarde.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM Reizen;
  PROPERTY Text "Reisschema ",;
  Top 2, Left 2, Width 38, Height 18,;
  OnMiddleMouseUp KostSchema
OPEN FORM Reizen
```

Zie ook

BITSET( ), OnLeftMouseDblClick, OnLeftMouseDown, OnLeftMouseUp, OnMiddleDblClick, OnMiddleMouseDown, OnRightMouseDblClick, OnRightMouseDown, OnRightMouseUp

OnMouseMove

Met het kenmerk OnMouseMove wordt een subroutine uitgevoerd wanneer de gebruiker de muisaanwijzer in een formulier verplaatst.

Kenmerk uit klasse

FORM

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnMouseMove om automatisch handelingen uit te voeren wanneer de gebruiker de muis beweegt.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnMouseMove het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Met het kenmerk OnMouseMove worden de volgende drie parameters aan de bijbehorende subroutine doorgegeven:

- <*Vlaggen*> Dit is een uit één byte bestaande waarde waaraan u kunt zien welke andere toetsen en muisknoppen werden ingedrukt toen de gebruiker de muisaanwijzer verplaatste. U kunt deze waarde interpreteren met de functie BITSET( ) waarmee individuele bits in numerieke waarden kunnen worden gecontroleerd.
- <*Kolom*> Dit is de horizontale positie van de muis nadat de gebruiker de muisaanwijzer heeft verplaatst.
- <*Rij*> Dit is de verticale positie van de muis nadat de gebruiker de muisaanwijzer heeft verplaatst.

De parameters <*Kolom*> en <*Rij*> geven de nieuwe positie aan op basis waarvan u handelingen kunt uitvoeren. Wanneer de gebruiker de muisaanwijzer bijvoorbeeld

verplaatst naar een deel van het formulier waar dat niet geoorloofd is, kan deze handeling worden opgemerkt via de subroutine van het kenmerk OnMouseMove en kan een dialoogvenster worden getoond met een waarschuwing.

Zie de muisactiekenmerken (zoals OnLeftMouseDown en OnRightMouseDown) voor meer informatie over de parameter *<Vlaggen>*.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnMouseMove** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL F1
F1=NEW Reizen()
F1.OPEN()
CLASS Reizen OF FORM
  this.Text = "Reisschema"
  this.Top = 2
  this.Left = 2
  this.Width = 38
  this.Height = 18
  this.OnMouseMove = KostLijst
* KostLijst is FUNCTIE waarmee handeling wordt uitgevoerd
* en die resulteert in .T. of waarde.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM Reizen;
  PROPERTY Text "Reisschema",;
  Top 2, Left 2, Width 38, Height 18,;
  OnMouseMove KostLijst
OPEN FORM Reizen
```

Zie ook

OnClick, OnLeftMouseDblClick, OnLeftMouseDown, OnLeftMouseUp, OnMiddleDblClick, OnMiddleMouseDown, OnMiddleMouseUp, OnRightMouseDblClick, OnRightMouseDown, OnRightMouseUp

OnMove

Met het kenmerk OnMove wordt een subroutine uitgevoerd nadat een formulier is geopend en de gebruiker dat formulier verplaatst.

Kenmerk uit klasse

FORM

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnMove om automatisch handelingen uit te voeren nadat een formulier is geopend en wordt verplaatst.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnMove het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Via OnMove worden twee parameters doorgegeven aan de bijbehorende subroutine:

- *<Links>* Dit is de nieuwe horizontale positie van de linkerbovenhoek van het formulier.
- *<Boven>* Dit is de nieuwe verticale positie van de rechterbovenhoek van het formulier.

Gebruiker de parameter Links en Boven voor sprongopdrachten. Wanneer de gebruiker bijvoorbeeld een formulier op een ander formulier plaatst, kan met de subroutine bij het kenmerk OnMove van het verplaatste formulier op basis van de parameters Links en Boven een nieuwe positie voor het onderste formulier worden berekend.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnMove** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL F1
F1=NEW Reizen()
F1.OPEN()
CLASS Reizen OF FORM
  this.Text = "Reisschema"
  this.Top = 2
  this.Left = 2
  this.Width = 38
  this.Height = 18
  this.OnRightClick = {;Form.Move(10,10,20,10)}
  this.OnMove = ToonKost
* ToonKost is FUNCTIE waarmee handeling wordt uitgevoerd
* en die resulteert in .T. of waarde.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM Reizen;
  PROPERTY Text "Reisschema",;
  Top 2, Left 2, Width 38, Height 18,;
  OnRightClick {;Form.Move(10,10,20,10)},;
  OnMove ToonKost
OPEN FORM Reizen
```

Zie ook

OnChange, OnClick, OnClose OnGotFocus, OnHelp, OnLeftMouseDown, OnLostFocus, OnOpen, OnSize

OnNavigate

Met het kenmerk OnNavigate wordt een subroutine uitgevoerd wanneer de recordaanwijzer in een tabel wordt verplaatst.

Kenmerk uit klasse

BROWSE, FORM

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnNavigate om uw applicatie te laten reageren wanneer de gebruiker van het ene record naar het andere gaat.

Met het kenmerk OnNavigate kunt u vaststellen wanneer de gebruiker de recordaanwijzer door de tabel verplaatst. In een bladerobject kunnen door middel van een bij het kenmerk OnNavigate opgegeven subroutine de activiteiten van een gebruiker worden vastgelegd. Steeds wanneer de gebruiker naar een ander record gaat, wordt het recordnummer in een array-object of een andere tabel opgeslagen.

Voorbeeld

Syntaxis operator NEW:

```
USE Contact
LOCAL f
f = NEW XFORM()
f.Open()

CLASS XFORM OF FORM
   this.HelpFile = ""
   this.Top =        5
   this.Height =    15
   this.Text = "Formulier"
```

```
this.Left =      39
this.Width =     72
this.HelpId = ""

DEFINE BROWSE BLADER1 OF THIS;
    PROPERTY;
      OnNavigate {;?Compcode+SPACE(5)+Contact},;
      Top          3,;
      Height       10,;
      FontBold .F.,;
      ColorNormal "N/W",;
      Fields "CompCode, Contact",;
      Left         7,;
      Width        59

ENDCLASS
```

Syntaxis commando DEFINE:

```
USE Contact
DEFINE FORM Navigatie FROM 1,1 TO 15,40
DEFINE BROWSE Blader1 OF Navigatie;
  PROPERTY Alias "Contact",;
    Height 15, Fields "CompCode, Contact",;
    Top 1, Left 1, Width 40, Height 15,;
    OnNavigate {;? CompCode+SPACE(5)+Contact}
OPEN FORM Navigatie
```

Zie ook

OnAppend, OnChange

OnNewValue

Met het kenmerk OnNewValue wordt een subroutine uitgevoerd wanneer een onderdeel met een automatische koppeling in een document van een DDE-server wordt gewijzigd.

Kenmerk uit klasse

DDELINK

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnNewValue om een handeling uit te voeren wanneer een onderdeel via een automatische koppeling in een server-toepassing wordt gewijzigd. Via een automatische koppeling die u met de methode Advise( ) maakt, wordt de server

duidelijk gemaakt dat veranderingen in het gekoppelde onderdeel aan dBASE moeten worden doorgegeven.

Een server-document is een bestand dat u in een externe toepassing opent. Bij een applicatie voor de uitwisseling van gegevens kan bijvoorbeeld Quattro Pro voor Windows worden gestart waarna een spreadsheet-bestand (een server-document) wordt geopend. U kunt met de methode Advise( ) automatische koppelingen maken met een of meer cellen in de spreadsheet en vervolgens het kenmerk OnNewValue gebruiken om een codeblok op te geven dat steeds moet worden uitgevoerd wanneer een van de cellen wordt gewijzigd.

U kunt de volgende twee parameters opgeven bij het kenmerk OnNewValue:

- *<element>* Dit staat voor het onderdeel in de server-toepassing waarnaar wordt weggeschreven. Hiermee kan het onderdeel worden aangeduid met de automatische koppeling zoals een veld in een tabel of een cel in een spreadsheet. Met "A:H3" wordt bijvoorbeeld cel H3 op pagina A van een spreadsheet in Quattro Pro opgegeven.
- *<waarde>* Dit is de nieuwe waarde van het server-onderdeel met de automatische koppeling.

In het codeblok kunnen deze parameters worden gebruikt om de wijziging te evalueren en op basis daarvan beslissingen te nemen.

Voorbeeld

```
PUBLIC KoppObj
KoppObj = NEW DDELINK()
KoppObj.OnNewValue = VerwerkWaarde;
                && Codeblok of functie-aanwijzer
KoppObj.Initiate("QPW","Demo.WB1")
KoppObj.Advise("A:A1");
                && Doorgegeven wanneer inhoud cel A:A1 verandert
```

Zie ook

Advise( ), Execute( ), Initiate( ), Peek( ), Poke( ), Server, Terminate( ), TimeOut, Topic, Unadvise( )

OnOpen

Met het kenmerk OnOpen wordt een subroutine uitgevoerd wanneer een formulier wordt geopend.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnOpen om een handeling uit te voeren wanneer een formulier wordt geopend.

De subroutine die is opgegeven bij het kenmerk OnOpen van een object, wordt uitgevoerd wanneer het hoofdformulier wordt geopend. De subroutine die is opgegeven bij het kenmerk OnOpen van een formulier, wordt uitgevoerd wanneer het formulier zelf wordt geopend.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnOpen het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnOpen** in het kenmerkenvenster te klikken.

Subroutines die u toewijst aan het kenmerk OnMove van het formulier worden ook uitgevoerd wanneer het formulier wordt geopend.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL F1
F1=NEW Reizen()
F1.OPEN()
CLASS Reizen OF FORM
  this.Text = "Reisschema"
  this.Top = 2
  this.Left = 2
  this.Width = 38
  this.Height = 18
  this.OnOpen = VltSchema
* VltSchema is FUNCTIE waarmee handeling wordt uitgevoerd
* en die resulteert in .T. of waarde.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM Reizen FROM 2,2 TO 18,38;
  PROPERTY Text "Reisschema",;
  OnOpen VltSchema
OPEN FORM Reizen
```

Zie ook

OnClose, OnGotFocus, OnLostFocus, OnMove, OnSize

OnPeek

Met het kenmerk OnPeek wordt een subroutine uitgevoerd wanneer een client-toepassing een onderdeel in een DDE server-toepassing probeert te lezen.

Kenmerk uit klasse

DDETOPIC

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnPeek om een waarde naar een client-toepassing te sturen wanneer de client-toepassing een Peek-verzoek doet.

Het kenmerk OnPeek kent de parameter *<element>* die aangeeft welke gegevens in dBASE door de client-toepassing gelezen willen worden.

Gebruik het commando RETURN om de gevraagde gegevens naar de client-toepassing te sturen. In een applicatie in Quattro Pro voor Windows kan bijvoorbeeld het commando {PEEK} worden uitgevoerd waardoor via de parameter *<element>* een veldnaam wordt verzonden. In de bij het kenmerk OnPoke opgegeven subroutine kan het commando RETURN worden gebruikt om de inhoud van het veld naar de client-toepassing te sturen:

```
RETURN &Gegevens
```

In dit geval wordt door de macrovervangings-functie (&) de inhoud van *Gegevens* in dBASE beschouwd als een veldnaam in plaats van een tekenreeks, zodat de inhoud van het veld naar de client-toepassing wordt gestuurd.

Zie CLASS DDETOPIC en Hoofdstuk 26 in *Programmeren* voor meer informatie over de objectklasse DDETopic.

Voorbeeld

Zie CLASS DDETOPIC voor een voorbeeld van het gebruik van OnPeek.

Zie ook

Advise( ), Execute( ), Initiate( ), OnNewValue, OnPoke, Peek( ), Poke( ), Server, Terminate( ), TimeOut, Topic, Unadvise( )

OnPoke

Met het kenmerk OnPoke wordt een subroutine uitgevoerd wanneer een client-toepassing een waarde probeert in te voegen in gegevens in een DDE-server.

Kenmerk uit klasse

DDETOPIC

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnPoke om een waarde te ontvangen van een client-toepassing en deze waarde in te voegen in een gegevenselement, wanneer de client-toepassing een Poke-verzoek doet.

Het kenmerk OnPoke kent de volgende twee parameters:

- *<element>* Hiermee worden de gegevens aangeduid waarin de waarde wordt ingevoegd. Deze kan elke willekeurige tekenreeks zijn.
- *<waarde>* Hiermee wordt de waarde aangeduid die in de gegevens moet worden ingevoegd.

Een client-toepassing kan bijvoorbeeld een veldnaam doorgeven en een waarde die in dat veld moet worden ingevuld. Via de subroutine die met het kenmerk OnPoke is opgegeven, kan de waarde in het veld worden ingevoerd waarbij het commando REPLACE en de macrovervangings-functie als volgt worden gebruikt:

```
REPLACE &Gegevens WITH Waarde
```

In dit geval wordt door de macrovervangings-functie (&) de inhoud van *Gegevens* in dBASE beschouwd als een veldnaam in plaats van een tekenreeks.

Zie CLASS DDETOPIC en Hoofdstuk 26 in *Programmeren* voor meer informatie over de objectklasse DDETopic.

Opmerking Als een automatische koppeling is gemaakt met een client-toepassing voordat de gegevens zijn verzonden, kunt u de methode Notify( ) uitvoeren in de subroutine bij het kenmerk OnPoke, waarmee u aan de client-toepassing doorgeeft dat een verandering heeft plaatsgevonden. Zie OnAdvise voor informatie over automatische koppelingen.

Voorbeeld

Zie CLASS DDETOPIC voor een voorbeeld van het gebruik van het kenmerk OnPoke.

Zie ook

Advise( ), Execute( ), Initiate( ), OnNewValue, OnPeek, Peek( ), Poke( ), Server, Terminate( ), TimeOut, Topic, Unadvise( )

OnRightDblClick

Met het kenmerk OnRightDblClick wordt een subroutine uitgevoerd wanneer de gebruiker met de rechtermuisknop dubbelklikt als de aanwijzer op een formulier of object staat.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnRightDblClick om een handeling uit te voeren wanneer de gebruiker met de rechtermuisknop dubbelklikt. Met het kenmerk OnRightDblClick kan ook worden geconstateerd of de gebruiker bij het dubbelklikken *Shift, Ctrl,* de middelste muisknop of de linkermuisknop ingedrukt houdt.

Met het kenmerk OnRightDblClick worden de volgende drie parameters aan de bijbehorende subroutine doorgegeven:

- <*Vlaggen*> Dit is een uit één byte bestaande waarde waaraan u kunt zien welke andere toetsen en muisknoppen werden ingedrukt toen de gebruiker dubbelklikte.
- <*Kolom*> Dit is de horizontale positie van de muis op het moment dat de gebruiker dubbelklikte.
- <*Rij*> Dit is de verticale positie van de muis op het moment dat de gebruiker dubbelklikte.

Parameter <*Vlaggen*>

Desgewenst kunt u de parameter *Vlaggen* gebruiken in combinatie met de functie BITSET( ) om verschillende combinaties van toetsen en muisknoppen te herkennen. Bij de functie BITSET( ) wordt een numerieke waarde geëvalueerd en het resultaat is de logische waarde waar (.T.) of onwaar (.F.), afhankelijk van het feit of een bepaald bit in de waarde aan (1) of uit (0) is. U moet de volgende twee parameters opgeven bij de functie BITSET( ):

- De waarde van de parameter <*Vlaggen*> zelf.
- Het bit in de parameter <*Vlaggen*> dat moet worden geëvalueerd.

Hieronder vindt u de waarden die u via de tweede parameter bij de functie BITSET( ) kunt opgeven:

Tegelijkertijd ingedrukte knop of toets	Tweede parameter
Linkermuisknop	0
Shift	2
Ctrl	3
Middelste muisknop	4

Als u bijvoorbeeld wilt controleren of de gebruiker *Shift* ingedrukt heeft gehouden terwijl hij of zij op een object dubbelklikte, gebruikt u BITSET(Vlaggen, 2). Als dat het geval is, resulteert de functie BITSET( ) in waar (.T.). De functies BITSET(Vlaggen, 3) en BITSET(Vlaggen, 0) resulteren beide in waar (.T.), als *Ctrl* en de linkermuisknop werden ingedrukt toen de gebruiker dubbelklikte. U gebruikt deze informatie in opdrachten als IF...ELSE...ENDIF of DO CASE...ENDCASE om verschillende handelingen toe te wijzen aan verschillende combinaties van toetsen en muisknoppen.

Parameters <*Kolom*> en <*Rij*>

Met de parameters <*Kolom*> en <*Rij*> wordt aangegeven waar de gebruiker met de rechtermuisknop dubbelklikte. Op basis daarvan kunt u verschillende handelingen uitvoeren. Met het kenmerk OnRightDblClick van een formulier kunnen bijvoorbeeld afhankelijk van de plaats waar de gebruiker dubbelklikt verschillende handelingen worden uitgevoerd. De parameters <*Kolom*> en <*Rij*> worden uitgedrukt in tekeneenheden.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnRightDblClick het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnRightDblClick** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL F1
F1=NEW Reizen()
F1.OPEN()
CLASS Reizen OF FORM
  this.Text = "Reisschema"
  this.Top = 2
  this.Left = 2
  this.Width = 38
```

```
    this.Height = 18
    this.OnRightDblClick = Dierenrijk
  * Dierenrijk is FUNCTIE waarmee handeling wordt uitgevoerd
  * en die resulteert in .T. of waarde.
  ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM Reizen;
  PROPERTY Text "Reisschema",;
  Top 2, Left 2, Width 38, Height 18,;
  OnRightDblClick Dierenrijk
OPEN FORM Reizen
```

Zie ook

BITSET( ), OnLeftMouseDblClick, OnLeftMouseDown, OnLeftMouseUp, OnMiddleDblClick, OnMiddleMouseDown, OnMiddleMouseUp, OnRightMouseDown, OnRightMouseUp

OnRightMouseDown

Met het kenmerk OnRightMouseDown wordt een subroutine uitgevoerd wanneer de gebruiker op de rechtermuisknop drukt terwijl de aanwijzer op een formulier of object staat.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnRightMouseDown om een handeling uit te voeren wanneer de gebruiker de rechtermuisknop indrukt. Met het kenmerk OnRightMouseDown kan ook worden geconstateerd of de gebruiker *Shift, Ctrl,* de middelste muisknop of de linkermuisknop ingedrukt houdt wanneer hij of zij op de rechtermuisknop drukt.

Met het kenmerk OnRightMouseDown worden de volgende drie parameters aan de bijbehorende subroutine doorgegeven:

- *<Vlaggen>* Dit is een uit één byte bestaande waarde waaraan u kunt zien welke andere toetsen en muisknoppen werden ingedrukt toen de gebruiker de rechtermuisknop indrukte.

- <*Kolom*> Dit is de horizontale positie van de muis op het moment dat de gebruiker de rechtermuisknop indrukte.
- <*Rij*> Dit is de verticale positie van de muis op het moment dat de gebruiker op de rechtermuisknop drukte.

Parameter <*Vlaggen*>

Desgewenst kunt u de parameter *Vlaggen* gebruiken in combinatie met de functie BITSET( ) om verschillende combinaties van toetsen en muisknoppen te herkennen. Bij de functie BITSET( ) wordt een numerieke waarde geëvalueerd en het resultaat is de logische waarde waar (.T.) of onwaar (.F.), afhankelijk van het feit of een bepaald bit in de waarde aan (1) of uit (0) is. U moet de volgende twee parameters opgeven bij de functie BITSET( ):

- De waarde van de parameter <*Vlaggen*> zelf.
- Het bit in de parameter <*Vlaggen*> dat moet worden geëvalueerd.

Hieronder vindt u de waarden die u via de tweede parameter bij de functie BITSET( ) kunt opgeven:

Tegelijkertijd ingedrukte knop of toets	Tweede parameter
Linkermuisknop	0
Shift	2
Ctrl	3
Middelste muisknop	4

Als u bijvoorbeeld wilt controleren of de gebruiker *Shift* ingedrukt heeft gehouden toen hij of zij op de rechtermuisknop drukte terwijl de muisaanwijzer op een object stond, gebruikt u BITSET(Vlaggen, 2). Als dat het geval is, resulteert de functie BITSET( ) in waar (.T.). De functies BITSET(Vlaggen, 3) en BITSET(Vlaggen, 0) resulteren beide in waar (.T.), als *Ctrl* en de linkermuisknop werden ingedrukt toen de gebruiker op de rechtermuisknop drukte. U gebruikt deze informatie in opdrachten als IF...ELSE...ENDIF of DO CASE...ENDCASE om verschillende handelingen toe te wijzen aan verschillende combinaties van toetsen en muisknoppen.

Parameters <*Kolom*> en <*Rij*>

Met de parameters <*Kolom*> en <*Rij*> wordt aangegeven waar de gebruiker heeft geklikt. Op basis daarvan kunt u verschillende handelingen uitvoeren. Met het kenmerk OnRightMouseDown van een formulier kunnen bijvoorbeeld afhankelijk van de plaats waar de gebruiker klikt verschillende handelingen worden uitgevoerd. De parameters <*Kolom*> en <*Rij*> worden uitgedrukt in tekeneenheden.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnRightMouseDown het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnRightMouseDown** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL F1
F1=NEW Reizen()
F1.OPEN()
CLASS Reizen OF FORM
  this.Text = "Reisschema"
  this.Top = 2
  this.Left = 2
  this.Width = 38
  this.Height = 18
  this.OnRightMouseDown = Dierenrijk
* Dierenrijk is FUNCTIE waarmee handeling wordt uitgevoerd
* en die resulteert in .T. of waarde.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM Reizen ;
  PROPERTY Text "Reisschema",;
  Top 2, Left 2, Width 38, Height 18,;
  OnRightMouseDown Dierenrijk
OPEN FORM Reizen
```

Zie ook

BITSET( ), OnLeftMouseDblClick, OnLeftMouseDown, OnLeftMouseUp, OnMiddleDblClick, OnMiddleMouseDown, OnMiddleMouseUp, OnRightMouseDblClick, OnRightMouseUp

OnRightMouseUp

Met het kenmerk OnRightMouseUp wordt een subroutine uitgevoerd wanneer de gebruiker de rechtermuisknop loslaat terwijl de aanwijzer op een formulier of object staat.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnRightMouseUp om een handeling uit te voeren wanneer de gebruiker de rechtermuisknop loslaat. Met het kenmerk OnRightMouseUp kan ook worden geconstateerd of de gebruiker *Shift, Ctrl,* de middelste muisknop of de linkermuisknop ingedrukt houdt wanneer hij of zij de rechtermuisknop loslaat.

Met het kenmerk OnRightMouseUp worden de volgende drie parameters aan de bijbehorende subroutine doorgegeven:

- <*Vlaggen*> Dit is een uit één byte bestaande waarde waaraan u kunt zien welke andere toetsen en muisknoppen werden ingedrukt toen de gebruiker de rechtermuisknop losliet.
- <*Kolom*> Dit is de horizontale positie van de muis op het moment dat de gebruiker op de rechtermuisknop losliet.
- <*Rij*> Dit is de verticale positie van de muis op het moment dat de gebruiker de rechtermuisknop losliet.

Parameter <*Vlaggen*>

Desgewenst kunt u de parameter *Vlaggen* gebruiken in combinatie met de functie BITSET( ) om verschillende combinaties van toetsen en muisknoppen te herkennen. Bij de functie BITSET( ) wordt een numerieke waarde geëvalueerd en het resultaat is de logische waarde waar (.T.) of onwaar (.F.), afhankelijk van het feit of een bepaald bit in de waarde aan (1) of uit (0) is. U moet de volgende twee parameters opgeven bij de functie BITSET( ):

- De waarde van de parameter <*Vlaggen*> zelf.
- Het bit in de parameter <*Vlaggen*> dat moet worden geëvalueerd.

Hieronder vindt u de waarden die u via de tweede parameter bij de functie BITSET( ) kunt opgeven:

Tegelijkertijd ingedrukte knop of toets	Tweede parameter
Linkermuisknop	0
Shift	2
Ctrl	3
Middelste muisknop	4

Als u bijvoorbeeld wilt controleren of de gebruiker *Shift* ingedrukt heeft gehouden toen hij of zij de linkermuisknop losliet, gebruikt u BITSET(Vlaggen, 2). Als dat het geval is, resulteert de functie BITSET( ) in waar (.T.). De functies BITSET(Vlaggen, 3) en BITSET(Vlaggen, 0) resulteren beide in waar (.T.), als *Ctrl* en de linkermuisknop werden ingedrukt toen de gebruiker de rechtermuisknop losliet. U gebruikt deze informatie in opdrachten als IF...ELSE...ENDIF of DO CASE...ENDCASE om verschillende handelingen toe te wijzen aan verschillende combinaties van toetsen en muisknoppen.

Parameters <*Kolom*> en <*Rij*>

Met de parameters <*Kolom*> en <*Rij*> wordt aangegeven waar de gebruiker de rechtermuisknop heeft losgelaten. Op basis daarvan kunt u verschillende handelingen uitvoeren. Met het kenmerk OnRightMouseUp van een formulier kunnen bijvoorbeeld afhankelijk van de plaats waar de gebruiker de rechtermuisknop loslaat, verschillende handelingen worden uitgevoerd. De parameters <*Kolom*> en <*Rij*> worden uitgedrukt in tekeneenheden.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnRightMouseUp het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnRightMouseUp** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL F1
F1=NEW Reizen()
F1.OPEN()
CLASS Reizen OF FORM
  this.Text = "Reisschema"
  this.Top = 2
  this.Left = 2
  this.Width = 38
  this.Height = 18
  this.OnRightMouseUp = Dierenrijk
* Dierenrijk is FUNCTIE waarmee handeling wordt uitgevoerd
* en die resulteert in .T. of waarde.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM Reizen ;
  PROPERTY Text "Reisschema",;
  Top 2, Left 2, Width 38, Height 18,;
  OnRightMouseUp Dierenrijk
OPEN FORM Reizen
```

Zie ook

BITSET( ), OnLeftMouseDblClick, OnLeftMouseDown, OnLeftMouseUp, OnMiddleDblClick, OnMiddleMouseDown, OnMiddleMouseUp, OnRightMouseDblClick, OnRightMouseDown

OnSelChange

Met het kenmerk OnSelChange wordt een subroutine uitgevoerd wanneer de selectie van een prompt in een keuzelijst ongedaan wordt gemaakt en een andere prompt wordt geselecteerd.

Kenmerk uit klasse

LISTBOX

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnSelChange om steeds wanneer een ander prompt in een keuzelijst wordt geselecteerd een subroutine uit te voeren..

Met het kenmerk OnSelChange kunt u vastleggen hoe de gebruiker door een keuzelijst loopt. Bij een keuzelijst kan bijvoorbeeld een subroutine worden toegewezen aan het kenmerk OnSelChange waarmee de activiteiten van de gebruiker worden opgeslagen door het nummer van elke geselecteerde prompt (de waarde van het kenmerk CurSel) in een tabel op te slaan wanneer de gebruiker een andere prompt selecteert.

Voorbeeld

Syntaxis operator NEW:

```
USE Contact
SET PRINTER TO FILE Record.Txt
* NIEUWE formulierdefinitie (Lst)
Lst1=NEW LISTBOX(this)
Lst1.Width=40
Lst1.Height=15
Lst1.DataSource="FIELD Contact->Contact"
Lst1.OnSelChange={;? CompCode+SPACE(3)+Contact}
```

Syntaxis commando DEFINE:

```
USE Contact
SET PRINTER TO FILE Record.Txt
DEFINE FORM Lst FROM 1,1 TO 15,40
DEFINE LISTBOX Lst1 OF Lst;
  PROPERTY Width 40, Height 15,;
    DataSource "FIELD Contact->Contact",;
    OnSelChange {;? CompCode+SPACE(3)+Contact}
OPEN FORM Lst
```

Zie ook

Selected( ), Count( ), CurSel

OnSelection

Met het kenmerk OnSelection wordt een subroutine uitgevoerd wanneer de gebruiker een formulier voorlegt.

Kenmerk uit klasse

FORM

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnSelection om subroutines of codeblokken op te geven die moeten worden uitgevoerd wanneer een gebruiker een formulier voorlegt. Wanneer een gebruiker het formulier selecteert, wordt het kenmerk ID van het laatste object met invoerfocus doorgegeven aan de subroutine.

In de volgende gevallen wordt een formulier voorgelegd:

- Als de gebruiker op *Enter* drukt wanneer het formulier focus heeft zonder dat een bladerobject of editor-object focus hebben.
- Als de gebruiker op de *Spatiebalk* drukt wanneer een knop focus heeft.
- Als de gebruiker op een knop klikt.

Wanneer de bij het kenmerk OnSelection of het commando ON SELECTION FORM opgegeven subroutine is uitgevoerd, wordt het formulier weer actief. Als in de subroutine echter het commando CLOSE FORM wordt uitgevoerd, wordt het formulier gesloten.

Met het kenmerk OnSelection wordt alleen de parameter *stuur_id* aan de subroutine doorgegeven. Deze parameter geeft aan via welke knop het formulier is voorgelegd. De waarde van de parameter *stuur_id* is gelijk aan de waarde van het kenmerk ID van de knop.

Opmerking U kunt in plaats van het kenmerk OnSelection ook het commando ON SELECTION FORM gebruiken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL F1
F1=NEW Reizen()
F1.OPEN()
CLASS Reizen OF FORM
  this.Text = "Reisschema"
  this.Top = 2
  this.Left = 2
```

```
    this.Width = 38
    this.Height = 18
    this.OnSelection = Dierenrijk
* Dierenrijk is FUNCTIE waarmee handeling wordt uitgevoerd
* en die resulteert in .T. of waarde.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM Reizen ;
  PROPERTY Text "Reisschema",;
  Top 2, Left 2, Width 38, Height 18,;
  OnSelection Dierenrijk
```

Zie ook

OnGotFocus, ON SELECTION FORM

OnSize

Met het kenmerk OnSize wordt een subroutine uitgevoerd nadat de gebruiker de afmetingen van een formulier heeft gewijzigd.

Kenmerk uit klasse

FORM

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk OnSize om automatisch handelingen uit te voeren wanneer de gebruiker de afmetingen van een formulier wijzigt.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk OnSize het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

Met het kenmerk OnSize worden de volgende drie parameters aan de bijbehorende subroutine doorgegeven:

- *<nType>* Dit is een numerieke waarde die aangeeft hoe de gebruiker de afmetingen van het formulier heeft gewijzigd. De parameter nType heeft de volgende drie mogelijke waarden:
- 0: De gebruiker heeft de afmetingen van het formulier met de muis gewijzigd of de vorige afmetingen hersteld nadat het formulier tot een maximumvenster was vergroot of tot een pictogram was verkleind.
- 1: De gebruiker heeft het formulier tot een pictogram verkleind.
- 2: De gebruiker heeft het formulier tot een maximumvenster verkleind.
- *<breedte>* Dit is de nieuwe breedte van het object.
- *<hoogte>* Dit is de nieuwe hoogte van het object.

De parameters Hoogte en Breedte geven de nieuwe afmetingen aan op basis waarvan u beslissingen kunt nemen.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **OnSize** in het kenmerkenvenster te klikken.

Wanneer de gebruiker de afmetingen van een formulier wijzigt dat in eerste instantie geen invoerfocus heeft, worden eerst de subroutines van de kenmerken When en OnGotFocus uitgevoerd.

De gebruiker kan de afmetingen van een formulier alleen handmatig wijzigen als het kenmerk Sizeable is ingesteld op waar (.T.).

Voorbeeld

Syntaxis operator NEW:

```
LOCAL F1
F1=NEW Reizen()
F1.OPEN()
CLASS Reizen OF FORM
  this.Text = "Reisschema"
  this.Top = 2
  this.Left = 2
  this.Width = 38
  this.Height = 18
  this.OnSize = Schuiven
* Schuiven is FUNCTIE waarmee handeling wordt uitgevoerd
* en die resulteert in .T. of waarde.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM Reizen FROM 2,2 TO 20,40;
  PROPERTY Text "Reisschema",;
  Top 2, Left 2, Width 38, Height 18,;
  OnSize Schuiven
OPEN FORM Reizen
```

Zie ook

OnChange, OnClick, OnClose, OnGotFocus, OnHelp, OnLeftMouseDown, OnLostFocus, OnMove, OnOpen, Sizeable

OnUnadvise

Met het kenmerk OnUnAdvise wordt een subroutine uitgevoerd wanneer dBASE niet langer aan een client-toepassing hoeft door te geven dat er gegevens in dBASE worden gewijzigd.

Kenmerk uit klasse

DDETOPIC

Gegevenstype

Logisch

Gebruik

Gebruik het kenmerk OnUnAdvise om te kunnen reageren op de beëindiging van een automatische koppeling (bij een automatische koppeling kunt u de methode Notify( ) gebruiken om een wijziging aan de client-toepassing door te geven).

De parameter *<element>* wordt doorgegeven aan het kenmerk OnUnAdvise. Dit is de naam van de gegevens in dBASE die aan de client-toepassing zijn gekoppeld. Dit kan elk veld, elke variabele en elk array-element in dBASE zijn.

Wanneer automatische koppelingen worden gemaakt, worden namen van gegevens vaak opgeslagen in tabellen of array-objecten. Wanneer bijvoorbeeld via een client-toepassing een automatische koppeling wordt gemaakt met een veld, wordt de naam van dat veld doorgegeven aan de subroutine die bij het kenmerk OnAdvise is gedefinieerd. De veldnaam kan elke keer in een array-object worden opgeslagen via de subroutine van het kenmerk OnAdvise.

Een bij het kenmerk OnPoke opgegeven subroutine kan dit array-object steeds doorzoeken wanneer een veld wordt gewijzigd. Als de naam van het gewijzigde veld in het array-object wordt gevonden, kan de methode Notify( ) worden uitgevoerd. Met het kenmerk OnUnadvise wordt de veldnaam uit de array verwijderd, zodat wijzigingen in dat veld niet langer met de methode Notify( ) worden doorgegeven.

Voorbeeld

Zie CLASS DDETOPIC voor een voorbeeld van het gebruik van het kenmerk OnUnAdvise.

Zie ook

Advise( ), Execute( ), Initiate( ), OnNewValue, Peek( ), Poke( ), Server, Terminate( ), TimeOut, Topic, Unadvise( )

Open( )

Met de methode Open( ) wordt een formulier geopend als een niet-modaal venster.

Kenmerk uit klasse

FORM

Gebruik

Gebruik de methode Open( ) om een formulier te openen en de bijbehorende objecten weer te geven.

Het formulier dat u met de methode Open( ) opent is *niet-modaal* en heeft de volgende karakteristieken:

1 Als het formulier is geopend, kan de focus naar andere formulieren worden verplaatst.

2 De routine waarmee het formulier is geopend stopt niet als het formulier is geopend en actief is.

U opent formulieren als niet-modale vensters als u meer dan een formulier tegelijkertijd geopend wilt hebben.

Opmerkingen Om een formulier te openen als een *modaal venster*, gebruikt u de methode ReadModal( ) of de functie READMODAL( ). Formulieren die als dialoogvenster worden gebruikt zijn bijvoorbeeld modaal, omdat door deze formulieren de uitvoering van het programma wordt onderbroken totdat de gebruiker reageert.

De methode Open( ) is identiek aan het commando OPEN FORM.

Voorbeeld

```
LOCAL f
f = NEW XFORM()
f.Open()

CLASS XFORM OF FORM
   this.HelpFile = ""
   this.Top =          2.00
   this.Height =        20.00
   this.Text = "Kijk...open!"
   this.Left =         35.00
   this.Width =         73.00
   this.HelpId = ""

ENDCLASS
```

Zie ook

CLOSE..., Close( ), OPEN FORM, ReadModal( ), READMODAL( )

Parent

Dit kenmerk is een objectverwijzing naar het hoofdformulier van een object of een menu.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, IMAGE, LISTBOX, LINE, MENU, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Objectverwijzing

Gebruik

Gebruik het kenmerk Parent voor verwijzingen naar het hoofdformulier of een object.

Stel het kenmerk Parent in met de clausule OF *<formuliernaam>* van het commando DEFINE of het commando REDEFINE wanneer u het object maakt. U kunt het kenmerk Parent ook instellen met de parameter *<formuliernaam>* die u voor het object opgeeft als u het maakt met de operator NEW.

U kunt een menu-object naar een ander formulier verplaatsen door de instelling van het kenmerk Parent van het menu-object te wijzigen. Bij alle andere klassen kan de instelling van het kenmerk Parent alleen worden gelezen.

Voorbeeld

In het volgende voorbeeld wordt een formulier gemaakt met een hoofdmenu met daarin de tekst "Bestand". Het menu Bestand heeft als submenu "Hoofdtekst wijzigen " waarmee het kenmerk Parent wordt gebruikt om de tekst "Bestand" te wijzigen in "Nieuwe tekst":

```
DEFINE FORM f1
DEFINE MENU Hoofd OF f1
DEFINE MENU mFile OF f1.Hoofd;
  PROPERTY;
  Text "Bestand"
DEFINE MENU mCp OF f1.Hoofd.mFile;
  PROPERTY;
  Text "Hoofdtekst wijzigen",;
  OnClick {;this.Parent.Text = "Nieuwe Tekst"}
OPEN FORM f1
```

Zie ook
ActiveControl, Before, First

PatternStyle

Met het kenmerk PatternStyle wordt een vooraf in Windows gedefinieerd arceerpatroon voor een achtergrond opgegeven.

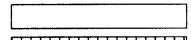
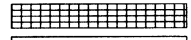
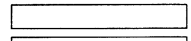
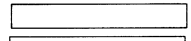
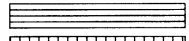
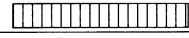
Kenmerk uit klasse
RECTANGLE

Gegevenstype
Numeriek

Standaardinstelling
De standaardinstelling van PatternStyle is 0.

Gebruik
Gebruik het kenmerk PatternStyle om een vooraf in Windows gedefinieerd arceerpatroon voor een achtergrond te selecteren voor een kaderobject.

U kunt de volgende instellingen kiezen voor het kenmerk PatternStyle:

Getal	Beschrijving	Voorbeeld
0	Effen	
1	Diagonaal	
2	Dubbele arcering	
3	Diagkruis	
4	Diagonaal 2	
5	Horizontaal	
6	Verticaal	

Voorbeeld
Syntaxis operator NEW:

```
Bedrijf = NEW ENTRYFIELD(this)
Bedrijf.Datalink = "Klanten->Bedrijf"
Bedrijf.PatternStyle = 3
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Bedrijf OF THIS;
  PROPERTY Datalink "Klanten->Bedrijf",;
  PatternStyle 3
```

Zie ook

ColorHighlight, ColorNormal

Peek( )

Via de methode Peek( ) worden gegevens opgevraagd uit een DDE-server.

Kenmerk uit klasse

DDELINK

Gebruik

Gebruik de methode Peek( ) om gegevens in een DDE server-onderwerp te lezen.

Een server-onderwerp is meestal een bestand dat u opent in een externe toepassing. In bijvoorbeeld een programma voor het uitwisselen van gegevens, kan Quattro Pro voor Windows worden gestart, een spreadsheet-bestand (onderwerp) worden geopend en met de methode Peek( ) de gegevens in een van de cellen worden gelezen.

Bij de methode Peek( ) moet de parameter *<element>* worden opgegeven waarmee gegevens in een server-onderwerp worden aangeduid. Hierbij kan het gaan om één element zoals een veld in een tabel of een cel in een spreadsheet. U kunt bijvoorbeeld de inhoud lezen van cel C2 op pagina A van een spreadsheet in Quattro Pro door voor de parameter *<element>* "A:C2" op te geven.

Voordat u gegevens kunt opvragen uit een server-onderwerp, moet u een DDE-koppeling met dat onderwerp maken. Zie Initiate( ) en Hoofdstuk 26 in *Programmeren* voor informatie over het maken van DDE-koppelingen.

Voorbeeld

```
PUBLIC KoppObj
KoppObj = NEW DDELINK()
IF KoppObj.Initiate("QPW","Demo.WB1")
  ? "Koppeling met QPW geinitialiseerd"
ELSE
  ? "Koppeling met QPW niet tot stand gebracht"
ENDIF
mWaarde1=KoppObj.Peek("A:A1")
? mWaarde1
```

Zie ook

Advise( ), Execute( ), Initiate( ), OnNewValue, Poke( ), Server, Terminate( ), TimeOut, Topic, Unadvise( )

Pen

Met het kenmerk Pen wordt het patroon van een lijnobject opgegeven.

Kenmerk uit klasse
LINE

Gegevenstype
Numeriek

Standaardinstelling
De standaardinstelling van Pen is 0 (effen).

Gebruik
Gebruik het kenmerk Pen om de weergave van een lijnobject in te stellen.

Het kenmerk Pen kent de volgende vijf instellingen:

Getal	Beschrijving	Voorbeeld
0	Effen	————
1	Streepjes	— — —
2	Stippeltjes	. . . . .
3	Streepje/stip	— . — . —
4	Streepje/stip/stip	— . . — . . —

Voorbeeld
Syntaxis operator NEW:

```
Ln2=NEW LINE(this)
Ln2.Left=3
Ln2.Top=8
Ln2.Bottom=8
Ln2.Right=33
Ln2.Pen = 3              && Streepje/stip
```

Syntaxis commando DEFINE:

```
DEFINE LINE Ln2 OF THIS;
  PROPERTY Left 3, Top 8, Bottom 8,;
    Right 33, Pen 3      && Streepje/stip
```

Zie ook
PatternStyle

Picture

Met het kenmerk Picture wordt tekst in een invoervak, ringveld of een tekstobject opgemaakt.

Kenmerk uit klasse

ENTRYFIELD, SPINBOX, TEXT

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van Picture is een lege tekenreeks.

Gebruik

Stel het kenmerk Picture in met een tekenreeks die een *sjabloon* wordt genoemd. Een sjabloon kan bestaan uit:

1 Sjabloontekens voor veldopmaak, deze staan voor individuele tekens in een tekenreeks en die deze kunnen wijzigen.

2 Functiesymbolen, hiermee wordt gewoonlijk de hele tekenreeks gewijzigd (zie het kenmerk Function voor informatie over functiesymbolen).

3 Vaste tekens, deze worden in de tekenreeks ingevoegd.

Hieronder vindt u de sjabloontekens voor veldopmaak:

9	Hiermee wordt de invoer van tekens beperkt tot cijfers en de invoer van numerieke gegevens tot cijfers en het plus- en minteken (+ en -)
#	Hiermee wordt de invoer beperkt tot cijfers, spaties, punten en symbolen
!	Hiermee worden letters omgezet in hoofdletters
$	Hiermee wordt een guldenteken of het symbool dat met het commando SET CURRENCY TO is gedefinieerd, ingevoegd in plaats van voorloopspaties
*	Hiermee worden asterisken ingevoegd in plaats van voorloopspaties
,	Hiermee wordt de positie van het decimaalteken aangegeven
.	Hiermee worden de duizendtallen gescheiden (of met een ander teken dat is gedefinieerd met het commando SET SEPARATOR)
A	Hiermee wordt de invoer beperkt tot letters
L	Hiermee wordt de invoer beperkt tot T, t, F, f, J, j, N of n. Deze letters worden bovendien omgezet in hoofdletters
N	Hiermee wordt de invoer beperkt tot letters en cijfers
X	Hiermee zijn alle tekens toegestaan
Y	Hiermee wordt de invoer beperkt tot J, j, N of n, alleen J en N worden weergegeven

Als u functiesymbolen gebruikt in een sjabloon, moet u deze vooraf laten gaan door het symbool @. Als u sjabloontekens en functiesymbolen in één sjabloon gebruikt, moet u eerst de functiesymbolen opgeven en deze met een spatie van de sjabloontekens scheiden.

Opmerking U kunt een veldopmaaksjabloon opgeven met het dialoogvenster **Sjabloon kiezen** waarin u veldopmaaksjablonen kunt invoeren. U opent het dialoogvenster **Sjabloon kiezen** door op de hulpmiddelenknop naast de optie **Picture** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
VLD1.Picture = "@!"
* of
VLD1.Picture = "F999,999,999.99"
* of
VLD1.Picture = "99:99:99"
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD VLD1 OF THIS;
  PROPERTY Datalink "<teken-, numeriek of datumveld>",;
  Picture "@!"
* of
  Picture "F999,999,999.99"
* of
  Picture "99:99:99"
```

Zie ook

@...SAY...GET, DEFINE, Function, TRANSFORM( )

Poke( )

Met de methode Poke( ) worden gegevens in een server-document ingevoegd.

Kenmerk uit klasse

DDELINK

Gebruik

Gebruik de methode Poke( ) om gegevens naar een server-document te schrijven.

Een server-document is een bestand dat u in een externe toepassing opent. In een programma voor het uitwisselen van gegevens kan bijvoorbeeld Quattro Pro voor Windows worden gestart en een spreadsheet-bestand worden geopend en de methode Poke( ) worden gebruikt om waarden naar een van de cellen van dat spreadsheet-bestand te schrijven.

Bij de methode Poke( ) moeten de volgende twee parameters worden opgegeven:

- *<element>* Dit is het element dat naar het server-document wordt geschreven. Dat kan bijvoorbeeld een veld in een tabel of een cel in een spreadsheet zijn. Met "B:G2" geeft u bijvoorbeeld cel G2 op pagina B van een spreadsheet-bestand in Quattro Pro aan.
- *<waarde>* Dit is de waarde die naar het server-document wordt geschreven. U kunt bijvoorbeeld een vaste waarde doorgeven maar ook de waarde die in een veld is opgeslagen.

Voordat u gegevens naar een server-document kunt sturen, moet u de server-toepassing starten, het document openen en een DDE-koppeling maken. Zie Initiate( ) en Hoofdstuk 26 in *Programmeren* voor informatie over het maken van DDE-koppelingen.

Voorbeeld

```
PUBLIC KoppObj
KoppObj = NEW DDELINK()
IF KoppObj.Initiate("QPW","Demo.WB1")
  ? "Koppeling met QPW geïnitialiseerd."
ELSE
  ? "Koppeling met QPW niet tot stand gebracht."
ENDIF
KoppObj.Poke("A:A3","198")
mWaarde2=KoppObj.Peek("A:A3")
? mWaarde2          && Succesvol uitvoeren POKE() bevestigen.
```

Zie ook

Advise( ), Execute( ), Initiate( ), OnNewValue, Peek( ), Server, Terminate( ), TimeOut, Topic, Unadvise( )

Print( )

Met de methode Print( ) worden een formulier en de objecten in het formulier afgedrukt.

Kenmerk uit klasse

FORM

Gebruik

Gebruik de methode Print( ) om een formulier af te drukken op een geselecteerde printer.

Wanneer de methode Print( ) wordt uitgevoerd, wordt het dialoogvenster **Afdrukken** geopend waarin de gebruiker de volgende instellingen kunnen definiëren:

- Het aantal pagina's dat moet worden afgedrukt
- De afdrukkwaliteit

- Het aantal exemplaren dat moet worden afgedrukt
- Of de uitvoer naar een bestand wordt weggeschreven of wordt afgedrukt
- Of de uitvoer wordt verzameld

De gebruiker klikt op **OK** om het formulier af te drukken.

Het uitvoeren van de methode Print( ) heeft hetzelfde resultaat als **Bestand | Afdrukken** selecteren. Om aan te geven op welke printer het formulier moet worden afgedrukt, voert u de functie CHOOSEPRINTER( ) uit.

Voorbeeld

In het volgende voorbeeld wordt een formulier gedefinieerd met daarin een bladerobject. Wanneer de gebruiker een of meer records ziet die hij of zij wil afdrukken, rechtsklikt hij of zij waardoor een codeblok wordt aangeroepen waarin de methode Print( ) wordt gebruikt om de weergegeven records naar de printer te sturen:

```
LOCAL f
f = NEW XFORM()
f.Open()

CLASS XFORM OF FORM
   this.HelpFile = ""
   this.Top =          5
   this.Height =       14
   this.Text = "Formulier"
   this.View = "DIEREN.DBF"
   this.OnRightMouseDown = {;this.Print()}
   this.Left =         34
   this.Width =        86
   this.HelpId = ""

   DEFINE BROWSE Blader1 OF THIS;
       PROPERTY;
         Top          3.00,;
         Height          8.00,;
         FontBold .F.,;
         ColorNormal "N/W",;
         Left         8.00,;
         Width        69.00

ENDCLASS
```

Zie ook

CHOOSEPRINTER( ), PRINTJOB...ENDPRINTJOB, SET PRINTER

RangeMax

Met het kenmerk RangeMax wordt de bovengrens gedefinieerd van de waarden die de gebruiker in een ringveld kan invoeren.

Kenmerk uit klasse

SCROLLBAR, SPINBOX

Gegevenstype

Datum of numeriek

Standaardinstelling

De standaardinstelling van RangeMax is 100,00.

Gebruik

Gebruik het kenmerk RangeMax in combinatie met het kenmerk RangeMin om het aantal waarden te beperken die een gebruiker in een schuifbalk of ringveld kan invoeren (met het kenmerk RangeMax stelt u de bovengrens en met het kenmerk RangeMin de ondergrens in). In een applicatie waar de gebruiker een percentage moet invoeren, kan bijvoorbeeld worden voorkomen dat de gebruiker waarden invoert die kleiner zijn dan 0 of groter dan 100.

Gebruiker moeten een waarde invoeren die tussen de gedefinieerde grenzen valt voordat zij de focus naar een ander object kunnen verplaatsen.

Opmerking Grenswaarden gelden alleen als het kenmerk RangeRequired op waar (.T.) is ingesteld.

Voorbeeld

Syntaxis operator NEW:

```
RNGVLD1.RangeMax = 100
* of
RNGVLD1.RangeMax = {31-12-94}
```

Syntaxis commando DEFINE:

```
DEFINE RingVeld Rng1 OF THIS;
  PROPERTY DataLink "<numeriek of datumveld>",;
  RangeMax 100
  * of
  RangeMax {31-12-94}
```

Zie ook

RangeMin, RangeRequired, Valid, ValidErrorMsg, ValidRequired

RangeMin

Met het kenmerk RangeMin wordt de ondergrens gedefinieerd voor de waarden die een gebruiker in een ringveld kan invoeren.

Kenmerk uit klasse
SCROLLBAR, SPINBOX

Gegevenstype
Datum of numeriek

Standaardinstelling
De standaardinstelling van RangeMin is 1.

Gebruik
Gebruik het kenmerk RangeMin in combinatie met het kenmerk RangeMax om het aantal waarden te beperken die een gebruiker in een schuifbalk of ringveld kan invoeren (met het kenmerk RangeMin stelt u de ondergrens en met het kenmerk RangeMax de bovengrens in). In een applicatie waarin de gebruiker een percentage moet invoeren, kan bijvoorbeeld worden voorkomen dat de gebruiker waarden invoert die kleiner zijn dan 0 of groter dan 100.

Gebruiker moeten een waarde invoeren die tussen de gedefinieerde grenzen valt voordat zij de focus naar een ander object kunnen verplaatsen.

Opmerking Grenswaarden gelden alleen als het kenmerk RangeRequired op waar (.T.) is ingesteld.

Voorbeeld
Syntaxis operator NEW:

```
RNGVLD1.RangeMin = 1
* of
RNGVLD1.RangeMin = {01-01-94}
```

Syntaxis commando DEFINE:

```
DEFINE RingVeld Rng1 OF THIS;
  PROPERTY DataLink "<numeriek of datumveld>",;
  RangeMin 1
  * of
  RangeMin {01-01-94}
```

Zie ook
RangeMax, RangeRequired, Valid, ValidErrorMsg, ValidRequired

RangeRequired

Met het kenmerk RangeRequired wordt bepaald of het bereik dat u instelt met de kenmerken RangeMax en RangeMin, altijd vereist is.

Kenmerk uit klasse
SPINBOX

Gegevenstype
Logisch

Standaardinstelling
De standaardinstelling van RangeRequired is onwaar (.F.).

Gebruik
Stel het kenmerk RangeRequired in op waar (.T.) als u de onder- en bovengrens wilt gebruiken die met de kenmerken RangeMax en RangeMin zijn gedefinieerd.

Als bijvoorbeeld het kenmerk RangeRequired van een ringveld op waar (.T.) is ingesteld, wordt in dBASE een fout geconstateerd als de ingevoerde waarde buiten het toegestane bereik valt en moet de gebruiker deze fout herstellen. Gebruikers moeten een waarde invoeren die binnen het gedefinieerde bereik valt voordat zij de focus naar een ander object kunnen verplaatsen.

Voorbeeld
Syntaxis operator NEW:

```
Datum = NEW ENTRYFIELD(this)
Datum.Datalink = "Afnemers->BalnsDatum"
Datum.RangeRequired = .T.
Datum.RangeMax = {31-12-95}
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Datum OF THIS;
  PROPERTY Datalink "Afnemers->BalnsDatum",;
  RangeRequired .T., RangeMax {31-12-95}
```

Zie ook
RangeMax, RangeMin, Valid, ValidErrorMsg

ReadModal( )

Met de methode ReadModal( ) wordt een formulier als een modaal venster geopend.

Kenmerk uit klasse
FORM

Gebruik

Gebruik de methode ReadModal( ) om een formulier als een modaal venster te openen. Een modaal venster heeft de volgende karakteristieken:

1 Als het formulier is geopend, kan de focus niet naar andere formulieren worden verplaatst.

2 De routine waarmee het formulier is geopend, wordt onderbroken totdat het formulier weer wordt gesloten. Wanneer het formulier wordt gesloten, wordt verdergegaan met het commando dat volgt op het commando waarmee het formulier is geopend.

In veel applicaties worden modale formulieren gebruikt als dialoogvensters waarbij gebruikers meestal een handeling moeten verrichten voordat het dialoogvenster kan worden gesloten.

Als het kenmerk MDI is ingesteld op waar (.T.), kunt u geen formulier openen met de methode ReadModal( ) of de functie READMODAL( ).

Als u een *niet-modaal* venster wilt openen, gebruikt u de methode Open( ) of het commando OPEN FORM.

De methode ReadModal( ) is identiek aan de functie READMODAL( ).

Voorbeeld

```
LOCAL f
f=NEW InvoerForm()
f.readmodal()
CLASS InvoerForm OF FORM
  this.MDI=.F.
  this.Top=2
  this.Left=2
  this.Width=38
  this.Height=13
* Volgende objectdefinities
ENDCLASS
```

Zie ook

Close( ), CLOSE..., OPEN FORM, Open( ), READMODAL( )

Reconnect( )

Met de methode Reconnect( ) wordt een beëindigde koppeling met een DDE server-toepassing opnieuw gemaakt. Als dat lukt, is het resultaat waar (.T.).

Kenmerk uit klasse

DDELINK

Gebruik

Gebruik de methode Reconnect( ) om een DDE-koppeling te herstellen die met de methode Terminate( ) is beëindigd.

Gebruik een DDE-koppeling om gegevens en instructies uit te wisselen met een andere toepassing. In een programma voor gegevensuitwisseling kan bijvoorbeeld een koppeling worden gemaakt met Quattro Pro voor Windows en een van de spreadsheet-bestanden worden geopend om vervolgens gegevens uit de spreadsheet naar een tabel in dBASE te kopiëren.

Wanneer u een DDE-koppeling beëindigt met de methode Terminate( ), kunt u deze weer herstellen met de methode Reconnect( ). Als u de koppeling beëindigt met de methode Release( ) kan deze niet meer worden hersteld en moet u het DDELink-object opnieuw maken.

Voorbeeld

```
PUBLIC KoppObj
KoppObj = NEW DDELINK()
KoppObj.Initiate("QPW","Demo.WB1")
* Volgende programmabewerkingen afgerond; koppeling niet meer nodig
KoppObj.Terminate()
* Later in programma DDE-koppeling weer nodig
KoppObj.Reconnect()
```

Zie ook

Initiate( ), Release( ), Terminate( )

Release( )

Met de methode Release( ) wordt een objectdefinitie uit het geheugen verwijderd.

Kenmerk uit klasse

BROWSE, DDELINK, DDETOPIC, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LINE, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gebruik

Gebruik de methode Release( ) zodra een object niet meer nodig is, zodat er geheugen vrijkomt. Als in een applicatie een formulier bijvoorbeeld niet meer nodig is, kan het formulier uit het geheugen worden verwijderd zodat dit vrijkomt voor andere doeleinden.

Opmerking Zodra een formulier wordt gesloten, geeft dBASE het daarvoor gereserveerde geheugen vrij, tenzij er nog een objectverwijzing naar verwijst.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL f
f=NEW InvoerForm()
f.OPEN()
  CLASS InvoerForm OF FORM
    this.Top=2
    this.Left=2
    this.Width=38
    this.Height=13
    this.OnClose={;Form.Release()}
* Volgende objectdefinities
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM InvoerForm;
  PROPERTY Top 2, Left 2, Width 38, Height 13,;
  OnClose {;Form.Release()}
* Andere stuurelementdefinities
OPEN FORM InvoerForm
```

Zie ook

RELEASE OBJECT

Resize( )

Met de methode Resize( ) wordt het aantal elementen in een array-object vergroot of verkleind.

Kenmerk uit klasse

ARRAY

Gebruik

Gebruik de methode Resize( ) om rijen en kolommen uit een array-object te verwijderen of aan een array-object toe te voegen. De methode Resize( ) lijkt op de functie ARESIZE( ).

U kunt bij de methode Resize( ) de volgende drie parameters opgeven:

- *<nieuwe rijen Nuitdr>*—Dit is het aantal rijen in het aangepaste array-object. Deze parameter moet groter zijn dan nul.
- *<nieuwe kolommen Nuitdr>*—Dit is het aantal kolommen in het aangepaste array-object. Deze parameter moet 0 of een positieve waarde zijn. Als u deze parameter niet gebruikt, wordt het aantal rijen gewijzigd en blijft het aantal kolommen gelijk.

- *<waarden behouden Nuitdr>*—Hiermee wordt bepaald hoe de array-elementen worden gerangschikt wanneer rijen worden verwijderd of toegevoegd. Zie ARESIZE( ) voor het gebruik van de parameter *<waarden behouden Nuitdr>*.

Voorbeeld

```
USE Klanten.DBF
* Array-object initialiseren.
ArrObj=NEW ARRAY(RECCOUNT())
* Eéndimensionale array vullen met waarden
* uit veld Naam in Klanten.DBF
GO TOP
FOR i=1 TO RECCOUNT()
  ArrObj[i]=Klanten->Naam
  SKIP
Next i
* RESIZE() gebruiken om twee extra kolommen toe te voegen
* aan bestaande array en Verktotnu en telefoonnummer in
* nieuwe kolommen invoeren
ArrObj.RESIZE(RECCOUNT(),3,1)
GO TOP
FOR i=1 TO RECCOUNT()
  ArrObj[i,2]=Klanten->Verktotnu
  ArrObj[i,3]=Klanten->Telefoon
  SKIP
Next i
* Inhoud tonen van driedimensionale array
FOR i = 1 TO RECCOUNT()
? ArrObj[i,1], ArrObj[i,2], ArrObj[i,3]
Next i
```

Zie ook

Add( ), Insert( ), GROW( ), ARESIZE( ), INSERT( )

Right

Met het kenmerk Right wordt de positie gedefinieerd van het rechteruiteinde van een lijnobject ten opzichte van het hoofdformulier.

Kenmerk uit klasse

LINE

Gegevenstype

Numeriek

Gebruik

Gebruik het kenmerk Right in combinatie met de kenmerken Bottom, Left en Top om de positie en lengte van een lijnobject te bepalen.

Elke eenheid van de waarde die u aan het kenmerk Right toewijst, is gelijk aan de gemiddelde breedte van de tekens in het actieve font van het hoofdformulier. Als u het kenmerk Right bijvoorbeeld instelt op 20, wordt het rechteruiteinde van de lijn op een afstand van 20 tekens naar rechts geplaatst.

Voorbeeld

```
DEFINE LINE Ln1 OF THIS;
  PROPERTY;
    Left 10,;          && Verticale lijn van 3,10
    Top 3,;            && tot 8,10
    Width 4,;
    Bottom 8,;
    ColorNormal "RB"
DEFINE LINE Ln2 OF THIS;
  PROPERTY;
    Left 3,;           && Horizontale lijn van 8,3
    Top 8,;            && tot 8,33
    Width 4,;
    Bottom 8,;
    Right 33,;
    ColorNormal "RB"
```

Zie ook

Height, ScaleFontName, ScaleFontSize, Top, Width

ScaleFontName

Met het kenmerk ScaleFontName wordt bepaald op welk font het coördinatenvlak van het formulier is gebaseerd.

Kenmerk uit klasse

FORM

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van FontName is MS Sans Serif.

Gebruik

Gebruik het kenmerk ScaleFontName in combinatie met het kenmerk ScaleFontSize om de hoogte van de rijen en de breedte van de kolommen in het *coördinatenvlak* van een formulier te bepalen.

Het coördinatenvlak is een tweedimensionaal raster van rij- en kolomcoördinaten. De hoogte van de rijen en de breedte van de kolommen hangen voornamelijk af van de volgende instellingen:

- Het actieve font van het formulier dat u opgeeft met het kenmerk ScaleFontName. De tekens van elk font hebben een andere breedte en hoogte zodat een andere instelling van het kenmerk ScaleFontName kan leiden tot een andere rijhoogte en kolombreedte.
- De grootte van het font dat u instelt met het kenmerk ScaleFontSize (de waarde die u bij het kenmerk ScaleFontSize opgeeft, moet zijn uitgedrukt in punten).

De rijhoogte en kolombreedte van een coördinatenvlak vormen samen een *tekeneenheid*. Bij kenmerken als Height en Width worden tekeneenheden gebruikt om de afmetingen en positie van een object te bepalen. Bij het kenmerk Height wordt bijvoorbeeld de hoogte in tekeneenheden uitgedrukt:

```
MijnForm.MijnObj.Height = 2.4   && 2,4 tekeneenheden
```

De daadwerkelijke hoogte van het object is afhankelijk van het aantal tekeneenheden *en* van de grootte van een tekeneenheid.

Zie Hoofdstuk 16 in *Programmeren* voor meer informatie over het coördinatenvlak.

Opmerking U kunt een font selecteren in het dialoogvenster **Font** waarin alle geïnstalleerde fonts worden getoond. U opent het dialoogvenster **Font** door op de hulpmiddelenknop naast de optie **ScaleFontName** in het kenmerkenvenster te klikken. U kunt het dialoogvenster **Font** ook openen met de functie GETFONT( ).

Voorbeeld

```
LOCAL f
f=NEW Scale()
f.OPEN()
CLASS Scale OF FORM
  this.ScaleFontName = "Courier"
  this.ScaleFontSize = 15
   DEFINE TEXT Txt1 OF THIS;
     PROPERTY Text "dBASE voor Windows",;
     Width 40, Top 5, Alignment 4
ENDCLASS
```

Zie ook

Bottom, GetTextExtent( ), Height, Right, Top, Width

ScaleFontSize

Met het kenmerk ScaleFontSize wordt de hoogte van elke rij en de breedte van elke kolom in het coördinatenvlak van een formulier bepaald.

Kenmerk uit klasse

FORM

Gegevenstype

Numeriek

Standaardinstelling

De standaardinstelling van ScaleFontSize is 8.00.

Gebruik

Gebruik het kenmerk ScaleFontSize in combinatie met het kenmerk ScaleFontName om de hoogte van de rijen en de breedte van de kolommen in het *coördinatenvlak* van een formulier te bepalen.

Het coördinatenvlak is een tweedimensionaal raster van rij- en kolomcoördinaten. De hoogte van de rijen en de breedte van de kolommen hangen voornamelijk af van de volgende instellingen:

- Het actieve font van het formulier dat u opgeeft met het kenmerk ScaleFontName. De tekens van elk font hebben een andere gemiddelde breedte en hoogte zodat een andere instelling van het kenmerk ScaleFontName kan leiden tot een andere rijhoogte en kolombreedte.
- De grootte van het font dat u instelt met het kenmerk ScaleFontSize (de waarde die u bij het kenmerk ScaleFontSize opgeeft, moet zijn uitgedrukt in punten).

De rijhoogte en kolombreedte van een coördinatenvlak vormen samen een *tekeneenheid*. Bij kenmerken als Height en Width worden tekeneenheden gebruikt om de afmetingen en positie van een object te bepalen. Bij het kenmerk Height wordt bijvoorbeeld de hoogte in tekeneenheden uitgedrukt:

```
MijnForm.MijnObj.Height = 4.5   && 4,5 tekeneenheden
```

De daadwerkelijke hoogte van het object is afhankelijk van het aantal tekeneenheden *en* van de grootte van een tekeneenheid.

Zie Hoofdstuk 16 in *Programmeren* voor meer informatie over het coördinatenvlak.

Voorbeeld

```
LOCAL f
f=NEW Scale()
f.OPEN()
CLASS Scale OF FORM
```

```
    this.ScaleFontName = "Courier"
    this.ScaleFontSize = 15
     DEFINE TEXT Txt1 OF THIS;
       PROPERTY Text "dBASE voor Windows",;
       Width 40, Top 5, Alignment 4
ENDCLASS
```

Zie ook

Bottom, Height, Right, Top, Width

Scan( )

Met de methode Scan( ) wordt in een array-object naar een bepaalde waarde gezocht.

Kenmerk uit klasse

ARRAY

Gebruik

Gebruik de methode Scan( ) om een waarde te zoeken in een array-object. Als een array-object bijvoorbeeld namen van klanten bevat, gebruikt u de methode Scan( ) om de locatie van een bepaalde naam op te zoeken. De methode Scan( ) lijkt op de functie ASCAN( ).

Het resultaat van de methode Scan( ) is het nummer van het eerste element dat overeenkomt met de uitdrukking, of het resultaat is 0 als de uitdrukking niet is gevonden. U kunt de methode Subscript( ) zo nodig gebruiken om het indextekennummer van het element op te zoeken. Zie Element( ) en Subscript( ) voor meer informatie over elementnummers en indextekens.

U kunt de volgende drie parameters bij de methode Scan( ) opgeven:

- *<uitdr>* Dit is de uitdrukking die moet worden gezocht.
- *<eerste element Nuitdr>* Dit is het nummer van het element waarbij met zoeken moet worden begonnen. Als u de parameter *<eerste element Nuitdr>* niet opgeeft wordt bij het eerste element begonnen met zoeken.
- *<elementen Nuitdr>* Dit is het aantal elementen dat met de methode Scan( ) wordt doorzocht. Als u de parameter *<elementen Nuitdr>* niet opgeeft, wordt de array doorzocht van het bij de parameter *<eerste element Nuitdr>* opgegeven element tot aan het laatste element. Als u een waarde opgeeft voor de parameter *<elementen Nuitdr>*, moet u ook een waarde opgeven voor de parameter *<eerste element Nuitdr>*.

Als voor de parameter *<uitdr>* een uitdrukking met een tekenreeks is opgegeven, wordt bij het zoeken onderscheid gemaakt tussen kleine en hoofdletters. U kunt derhalve gebruik maken van de functies UPPER( ), LOWER( ) of PROPER( ) om in de uitdrukking dezelfde hoofdletters en kleine letters te gebruiken als in het array-object.

Als voor de parameter <*uitdr*> een uitdrukking met een tekenreeks is opgegeven, wordt met de methode Scan( ) naar een uitdrukking gezocht op basis van de instelling van de functie SET EXACT. Als SET EXACT is ingesteld op ON, resulteert de methode Scan( ) in 0 als de voor de parameter <*uitdr*> opgegeven uitdrukking niet identiek is aan een element in het array-object. Als de functie SET EXACT is ingesteld op OFF, resulteert de methode Scan( ) in 0 als de tekens die bij de parameter <*Nuitdr*> zijn opgegeven niet overeenkomen met begintekens van de gegevens in een element van het array-object. Zie SET EXACT voor meer informatie.

Voorbeeld

```
USE Dieren.DBF
* Array-object initialiseren.
ArrObj=NEW ARRAY(RECCOUNT(),3)
* Array vullen met waarden uit bestand Dieren.DBF.
COPY TO ARRAY ArrObj FIELDS Naam, Gebied, Gewicht
* SCAN() gebruiken om in array te zoeken naar reeks
* "Papegaai" en elementnummer te verkrijgen.
* SUBSCRIPT() gebruiken om rij en kolom
* van door SCAN() gevonden element te verkrijgen:
String = "Papegaai"
aElement = ArrObj.Scan(String)
aRow = ArrObj.SUBSCRIPT(aElement,1)
aCol = ArrObj.SUBSCRIPT(aElement,2)
? String +" bevindt zich in rij " + ;
  LTRIM(STR(aRow)) + ", kolom " + ;
  LTRIM(STR(aCol))
```

Zie ook

ASCAN( ), Element( ), LOWER( ), PROPER( ), SET EXACT, Sort( ), Subscript( ), UPPER( )

ScrollBar

Met het kenmerk ScrollBar wordt bepaald of een formulier of een editor-object een schuifbalk krijgt.

Kenmerk uit klasse

EDITOR, FORM

Gegevenstype

Numeriek

Standaardinstelling

De standaardinstelling van ScrollBar is 1 (aan) bij editor-objecten en 0 (uit) bij formulieren.

Gebruik

Maak een schuifbalk zodat de gebruiker door een formulier of editor-object kan bladeren als de inhoud groter is dan het formulier of het editor-object zelf. Als een editor-object bijvoorbeeld tien regels hoog is en de tekst uit twintig regels bestaat, kan de gebruiker met een schuifbalk door de tekst bladeren.

U kunt de volgende vier instellingen bij het kenmerk ScrollBar opgeven:

ScrollBar waarde	Omschrijving
0 - Uit	Het object heeft geen schuifbalk.
1 - Aan	Het object heeft een schuifbalk.
2 - Automatisch	De schuifbalk wordt alleen weergegeven wanneer dat nodig is.
3 - Uitgeschakeld	De schuifbalk is zichtbaar maar kan niet worden gebruikt.

Voorbeeld

```
LOCAL f
f = NEW XFORM()
f.Open()

CLASS XFORM OF FORM
   this.Top = 2
   this.Height = 20
   this.Text = "Formulier"
   this.ScrollBar = 1
   this.Left = 33
   this.Width = 72

   DEFINE EDITOR ED1 OF THIS;
       PROPERTY;
         DataLink "CONTACT->NOTITIES",;
         Top          5.00,;
         Height         10.00,;
         Border .T.,;
         ColorNormal "N/W*",;
         ScrollBar          0,;
         Left         13.00,;
         Width         50.00

ENDCLASS
```

Zie ook

CLASS SCROLLBAR

SelectAll

Met het kenmerk SelectAll wordt aangegeven of de waarde in een invoervak of ringveld in eerste instantie geselecteerd (gemarkeerd) is.

Kenmerk uit klasse

ENTRYFIELD, SPINBOX

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van SelectAll is onwaar (.F.).

Gebruik

Stel het kenmerk SelectAll in op waar (.T.) zodat de gebruiker snel de beginwaarde in een invoervak of ringveld kan verwijderen of vervangen. De waarde (die is gedefinieerd met de kenmerken DataLink of Value) wordt geselecteerd zodra de gebruiker het object focus geeft. Wanneer de gebruiker een teken typt, wordt de beginwaarde verwijderd. Wanneer de gebruiker op *Del* of *Backspace* drukt, wordt de waarde verwijderd. Wanneer de gebruiker op een pijltoets drukt (bijvoorbeeld *Pijl links* of *Pijl rechts*) wordt de markering verwijderd maar blijft de waarde staan.

Voorbeeld

Syntaxis operator NEW:

```
Datum = NEW ENTRYFIELD(this)
Datum.Datalink = "Afnemers->BalnsDatum"
Datum.SelectAll = .T.
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Datum OF THIS;
  PROPERTY Datalink "Afnemers->BalnsDatum",;
  SelectAll .T.
```

Zie ook

Mode, Style

Selected( )

Het resultaat van de methode Selected( ) is de prompt die in een keuzelijst is geselecteerd.

Kenmerk uit klasse
LISTBOX

Gebruik
Gebruik de methode Selected( ) in combinatie met de methode Count( ) om de keuzes te evalueren die een gebruiker heeft gemaakt in een meerkeuzelijst. U kunt bijvoorbeeld in een FOR...NEXT-lus met behulp van de methode Selected( ) elke prompt evalueren om na te gaan welke prompts zijn geselecteerd (als het aantal prompts in de keuzelijst niet altijd gelijk is, gebruikt u de methode Count( ) om vooraf te bepalen hoe vaak de lus wordt uitgevoerd).

Bij de methode Selected( ) kan één optionele parameter worden opgegeven: *<element>*. Dit is een numerieke waarde die een prompt aangeeft op basis van de relatieve positie in de keuzelijst.

U maakt een meerkeuzelijst door het kenmerk Multiple op waar (.T.) in te stellen.

Voorbeeld
Zie Count( ) voor een voorbeeld van het gebruik van Selected( ).

Zie ook
FOR...NEXT, Count( ), Multiple

Separator

Met het kenmerk Separator wordt aangegeven of een menu-onderdeel een lijn is die de gebruiker niet kan selecteren.

Kenmerk uit klasse
MENU

Gegevenstype
Logisch

Standaardinstelling
De standaardinstelling van Separator is onwaar (.F.).

Gebruik
Stel het kenmerk in op waar (.T.) als u een menu-onderdeel wilt gebruiken als een scheiding tussen twee groepen met menucommando's. In een menu met de naam

Boekhouding kunt u bijvoorbeeld een scheidingslijn gebruiken om Credit-opdrachten te onderscheiden van Debet-opdrachten.

Voorbeeld

```
DEFINE FORM f1
DEFINE MENU Hoofd OF f1
DEFINE MENU mOpt1 OF f1.Hoofd;
  PROPERTY;
  Text "Opdracht 1"
DEFINE MENU mSlct1 OF f1.Hoofd.mOpt1;
  PROPERTY;
  Text "Selecteer 1"
DEFINE MENU mLijn1 OF f1.Hoofd.mOpt1;
  PROPERTY;
  Separator .T.
DEFINE MENU mSlct2 OF f1.Hoofd.mOpt1;
  PROPERTY;
  Text "Selecteer 2"
DEFINE MENU mLijn2 OF f1.Hoofd.mOpt1;
  PROPERTY;
  Separator .T.
DEFINE MENU mSlct3 OF f1.Hoofd.mOpt1;
  PROPERTY;
  Text "Selecteer 3"
OPEN FORM f1
```

Zie ook

CLASS MENU

Server

Het kenmerk Server is ingesteld op de naam van een DDE server-toepassing.

Kenmerk uit klasse

DDELINK

Gegevenstype

Teken

Gebruik

Gebruik het kenmerk Server voor de server-toepassing waarmee u een DDE-koppeling hebt gemaakt.

U maakt een DDE-koppeling met de methode Initiate( ), bijvoorbeeld een koppeling met Quattro Pro voor Windows. Vervolgens opent u een van de spreadsheet-bestanden en kopieert u gegevens uit de cellen naar een dBASE-tabel.

Het kenmerk Server is ingesteld op de waarde van de eerste parameter die u aan de methode Initiate( ) hebt doorgegeven en is gelijk aan de naam van het hoofdprogrammabestand van de server-toepassing.

Voorbeeld

```
PUBLIC KoppObj
KoppObj = NEW DDELINK()
KoppObj.Initiate("QPW","Demo.WB1")
* Volgende programmabewerkingen
IF KoppObj.Server="QPW" .AND. ;
   KoppObj.Topic = "Demo.WB1"
      mWaarde1=KoppObj.Peek("A:A1")
ENDIF
```

Zie ook

Advise( ), Execute( ), Initiate( ), OnNewValue, Peek( ), Poke( ), Terminate( ), TimeOut, Topic, Unadvise( )

ServerName

Het kenmerk ServerName geeft de naam aan die wordt aangeroepen wanneer de gebruiker op een OLE-weergave-object dubbelklikt.

Kenmerk uit klasse

OLE

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van ServerName is een lege tekenreeks.

Gebruik

Gebruik het kenmerk ServerName zodat u weet welke server-toepassing wordt gestart als de gebruiker dubbelklikt op het huidige OLE-weergave-object.

In een OLE-weergave-object wordt een OLE-document getoond. Een OLE-document kan een grafische afbeelding zijn maar ook een document dat in een tekstverwerker is gemaakt of andere gegevens die in een externe toepassing zijn gemaakt. Deze externe

toepassing wordt de *OLE-server* genoemd. Een grafische afbeelding die in Paintbrush is gemaakt kan bijvoorbeeld een OLE-document zijn, en Paintbrush een OLE-server, als u de grafische afbeelding (of een deel daarvan) met **Knippen** en **Plakken** uit het menu **Bewerken** insluit in een OLE-veld.

De instelling van het kenmerk ServerName kan alleen worden gelezen.

Voorbeeld

Zie DoVerb( ) voor een voorbeeld van het gebruik van ServerName.

Zie ook

LinkFileName, OleType

SetFocus( )

Met de methode SetFocus( ) wordt focus gegeven aan een formulier of een object in een formulier.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LISTBOX, OLE, PUSHBUTTON, RADIOBUTTON, SCROLLBAR, SPINBOX

Gebruik

Gebruik de methode SetFocus( ) om te zorgen dat een gebruiker gegevens kan invoeren in een object of formulier.

Als een formulier focus heeft, wordt de titelbalk helderder weergegevens. Als een object in een formulier focus heeft, wordt de rand helderder weergegeven. Een formulier of object met focus wordt het *huidige* formulier of object genoemd.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL f
f=NEW InvoerForm()
f.OPEN()
CLASS Invoerform OF FORM
  this.View="Bedrijf.DBF"
  this.Vld1 = NEW Entryfield(this)
  this.Vld1.Top = 3
  this.Vld1.Left = 1
  this.Vld1.Width=20
  this.Vld1.Datalink = "Bedrijf->Bedrijf"
  this.Vld2 = NEW Entryfield(this)
  this.Vld2.Top = 5
  this.Vld2.Left = 1
```

```
  this.Vld2.Datalink = "Bedrijf->Regio"
* Focus verplaatsen naar Vld2 wanneer muis
* over tweede veld wordt bewogen:
  this.Vld2.OnMouseMove = {;this.SetFocus()}
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Bedrijf OF THIS;
  Property Datalink "Bedrijf->Bedrijf",;
  Top 3, Left 1, Width 20
DEFINE ENTRYFIELD Regio OF THIS;
  Property Datalink "Bedrijf->Regio",;
  Top 5, Left 1,;
* Focus verplaatsen naar Vld2 wanneer muis
* over tweede veld wordt bewogen:
  OnMouseMove {;this.SETFOCUS()}
```

Zie ook

ActiveControl, _curobj, Nextobj, SET CUAENTER

ShortCut

Met het kenmerk ShortCut wordt een toetscombinatie gedefinieerd waarmee de bij het kenmerk OnClick van een menu-object opgegeven subroutine kan worden uitgevoerd.

Kenmerk uit klasse

MENU

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van ShortCut is een lege tekenreeks.

Gebruik

Gebruik het kenmerk ShortCut om een menucommando snel te kunnen uitvoeren vanaf het toetsenbord. Als u bijvoorbeeld de tekenreeks "CTRL-S" aan het kenmerk ShortCut toewijst, kan de gebruiker de bij het kenmerk OnClick gedefinieerde subroutine uitvoeren door op *Ctrl-S* of *Ctrl-s* te drukken.

De waarde die u bij het kenmerk ShortCut opgeeft, wordt naast de prompt weergegeven die met het kenmerk Text is gedefinieerd.

Voorbeeld

Syntaxis operator NEW:

```
BstBhr= NEW MENU ITEM(this)
BstBhr.ShortCut = "Alt-B"
```

Syntaxis commando DEFINE:

```
DEFINE MENU ITEM BstBhr OF THIS;
  PROPERTY ShortCut "Alt-B"
```

Zie ook

OnClick

ShowDeleted

Met het kenmerk ShowDeleted wordt bepaald of de wismarkeringen in een bladerobject worden weergegeven.

Kenmerk uit klasse

BROWSE

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van ShowDeleted is waar (.T.).

Gebruik

Gebruik het kenmerk ShowDeleted om wismarkeringen links van elk record in een bladerobject weer te geven of weg te laten.

De instelling van het kenmerk ShowDeleted wordt alleen gebruikt als de functie SET DELETED is ingesteld op OFF. Als de functie SET DELETED op ON staat, worden de wismarkeringen nooit weergegeven.

Voorbeeld

Syntaxis operator NEW:

```
Blader1=NEW BROWSE(this)
Blader1.Fields="CompCode,Contact"
Blader1.ShowDeleted=.T.
```

Syntaxis commando DEFINE:

```
DEFINE BROWSE Blader1 OF THIS;
    PROPERTY Fields "CompCode, Contact",;
      ShowDeleted .T.
```

Zie ook

ShowHeading, ShowRecno

ShowHeading

Met het kenmerk ShowHeading wordt bepaald of boven de kolommen in een bladerobject de veldnamen worden weergegeven.

Kenmerk uit klasse

BROWSE

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van ShowHeading is waar (.T.).

Gebruik

Gebruik het kenmerk ShowHeading om te bepalen of op de bovenste regel van een bladerobject een record of een titel wordt weergegeven.

Voorbeeld

Syntaxis operator NEW:

```
Blader1 NEW BROWSE(this)
Blader1.Fields = "CompCode,Contact"
Blader1.Width = 38
Blader1.Height = 17
Blader1.Showheading = .F.
```

Syntaxis commando DEFINE:

```
DEFINE BROWSE Blader1 OF THIS;
  PROPERTY;
    Fields "CompCode, Contact",;
    Width 38,;
    Height 17,;
    ShowHeading .F.
```

Zie ook
ShowDeleted, ShowRecno

ShowRecNo

Met het kenmerk ShowRecordNo wordt bepaald of in een bladerobject een kolom met recordnummers wordt weergegeven.

Kenmerk uit klasse
BROWSE

Gegevenstype
Logisch

Standaardinstelling
De standaardinstelling van ShowRecNo is waar (.T.)

Gebruik
Gebruik het kenmerk ShowRecNo om links van elke record in een bladerobject de recordnummers weer te geven of weg te laten.

De instelling van het kenmerk ShowRecNo is van invloed op het gebruik van de ruimte in het bladerobject. Als u het kenmerk ShowRecNo bijvoorbeeld op onwaar (.F.) instelt, wordt de kolom met recordnummers verwijderd waardoor meer ruimte overblijft voor de weergave van velden.

Voorbeeld
Syntaxis operator NEW:

```
Blader1=NEW BROWSE(this)
Blader1.Fields="CompCode,Contact"
Blader1.ShowRecNo=.F.
```

Syntaxis commando DEFINE:

```
DEFINE BROWSE Blader1 OF THIS;
    PROPERTY Fields "CompCode, Contact",;
      ShowRecNo .F.
```

Zie ook
ShowDeleted, ShowHeading

Size

Het kenmerk Size is ingesteld op het aantal elementen in een array-object.

Kenmerk uit klasse
ARRAY

Gegevenstype
Numeriek

Gebruik
Gebruik het kenmerk Size om na te gaan hoeveel elementen in een array-object zitten.

Als u het aantal elementen in een array-object weet, kunt u door de array lopen door een FOR...NEXT-lus uit te voeren en elk element te lezen of aan elk element een waarde toe te kennen. Zie Dir( ) voor meer informatie over door array's lopen.

Voorbeeld
Zie Dir( ) voor een voorbeeld van het gebruik van Size in combinatie met array-objecten.

Zie ook
ALEN( ), Delete( ), Dimensions, Grow( ), Insert( ), Resize( )

Sizeable

Met het kenmerk Sizeable wordt bepaald of de gebruiker de afmetingen van een formulier kan wijzigen.

Kenmerk uit klasse
FORM

Gegevenstype
Logisch

Standaardinstelling
De standaardinstelling van Sizeable is waar (.T.).

Gebruik

Stel het kenmerk Sizeable in op onwaar (.F.) om te voorkomen dat de gebruiker de afmetingen van een formulier wijzigt.

Als u het kenmerk Sizeable instelt op waar (.T.), wordt een dikke rand om het formulier weergegeven. Deze rand geeft aan dat de gebruiker de afmetingen met de muis kan veranderen. Als u het kenmerk Sizeable instelt op onwaar (.F.), wordt een dunne rand om het formulier getoond. De muisaanwijzer verandert in dit geval niet van vorm wanneer de gebruiker deze over de rand beweegt en het formulier kan niet groter of kleiner worden gemaakt.

Opmerking Als u het kenmerk MDI van een formulier op waar (.T.) instelt, wordt de instelling van het kenmerk Sizeable genegeerd en kan de gebruiker de afmetingen van het formulier altijd aanpassen.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL F1
F1=NEW Reizen()
F1.OPEN()
CLASS Reizen OF FORM
  this.MDI = .F.
  this.Text = "Reisschema"
  this.Top = 2
  this.Left = 2
  this.Width = 38
  this.Height = 18
  this.Sizeable = .F.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM Reizen ;
  PROPERTY Text "Reisschema",;
  Top 2, Left 2, Width 38, Height 18,;
  Sizeable .F., MDI .F.
```

Zie ook

Moveable, Maximize, Minimize

Sort( )

Met de methode Sort( ) worden de elementen in een ééndimensionaal array-object gesorteerd of de rijen in een tweedimensionaal array-object.

Kenmerk uit klasse

ARRAY

Gebruik

Gebruik de methode Sort( ) om elementen in een array-object in een alfabetische, numerieke, chronologische of logische volgorde te rangschikken.

De methode Sort( ) lijkt op de functie ASORT( ).

De te sorteren elementen in een ééndimensionaal array-object moeten van hetzelfde gegevenstype zijn. Dit geldt ook voor de elementen in de kolom op basis waarvan de rijen moeten worden gesorteerd.

U kunt de volgende drie parameters opgeven bij de methode Sort( ):

- *<eerste element Nuitdr>* Dit is het nummer van het element waarbij moet worden begonnen met sorteren in een ééndimensionaal array-object of het nummer (indexteken) van de kolom op basis waarvan een tweedimensionaal array-object moet worden gesorteerd. Als u de parameter *<eerste element Nuitdr>* niet gebruikt, wordt met sorteren begonnen bij het eerste element of de eerste kolom in de array.
- *<te sorteren elementen Nuitdr>* Dit is het aantal elementen dat moet worden gesorteerd in een ééndimensionaal array-object of het aantal te sorteren kolommen in een tweedimensionaal array-object. Als u de parameter *<te sorteren elementen Nuitdr>* niet gebruikt, worden de rijen gesorteerd vanaf de rij met het element dat met de parameter *<eerste element Nuitdr>* is opgegeven tot en met de laatste rij. Als u een waarde opgeeft voor de parameter *<te sorteren elementen Nuitdr>*, moet u ook een waarde definiëren voor de parameter *<eerste element Nuitdr>*.
- *<sorteervolgorde Nuitdr2>* Dit is de sorteervolgorde. De waarde 0 staat voor een oplopende volgorde (standaardinstelling) en de waarde 1 voor een aflopende volgorde.

Zie ASORT( ) voor meer informatie over het sorteren van elementen in een array-object.

Voorbeeld

```
USE Dieren.DBF
* Array-object initialiseren.
ArrObj=NEW ARRAY(RECCOUNT(),3)
* Array vullen met waarden uit Dieren.DBF
COPY TO ARRAY ArrObj FIELDS Naam, Gebied, Gewicht
* SORT() gebruiken om te sorteren op waarden gebied
* in kolom 2 van array.
ArrObj.Sort(2)
* Weergave gesorteerd op inhoud array.
FOR i=1 TO RECCOUNT()
  ? ArrObj[i,1], ArrObj[i,2], ArrObj[i,3]
NEXT i
```

Zie ook

ASORT( ), Dir( ), Fields( ), Scan( )

Sorted

Met het kenmerk Sorted wordt bepaald of de prompts in een keuzelijst of keuzelijst met invoervak worden gesorteerd of niet.

Kenmerk uit klasse

COMBOBOX, LISTBOX

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Sorted is onwaar (.F.).

Gebruik

Stel het kenmerk Sorted in op waar (.T.), als u de prompt in een keuzelijst of keuzelijst met invoervak wilt sorteren (op alfabet, nummer of in een chronologische volgorde). Een lijst met namen is bijvoorbeeld duidelijker als deze op alfabet is gesorteerd.

Als de prompts in een keuzelijst of keuzelijst met invoervak niet zijn gesorteerd, is de volgorde afhankelijk van de volgorde waarin de prompts zijn gemaakt. Als u het kenmerk DataSource van een keuzelijst instelt op "FILE *.*", bestaan de prompts uit de bestandsnamen in de standaarddirectory. De prompts worden gemaakt in de volgorde waarin de bestanden in de directory staan. Deze prompts hoeven dus niet op alfabet gesorteerd te zijn als u het kenmerk Sorted op onwaar (.F.) instelt.

Sorted werkt niet als bij het kenmerk DataSource van het lijstvak of de keuzelijst met invoervak "FIELD" plus een veldnaam is opgegeven. In dat geval is de volgorde van de prompts afhankelijk van de volgorde van de records in de tabel met het opgegeven veld.

Voorbeeld

Syntaxis operator NEW:

```
LST1 = NEW LISTBOX(this)
LST1.DataSource = "File"
LST1.Sorted = .T.
```

Syntaxis commando DEFINE:

```
DEFINE LISTBOX LST1 OF FORM THIS;
  PROPERTY DataSource "File",;
  Sorted .T.
```

Zie ook

Multiple

SpeedBar

Met het kenmerk SpeedBar wordt bepaald of een knop werkt als een knop op een knoppenbalk of als een standaardknop.

Kenmerk uit klasse

PUSHBUTTON

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van SpeedBar is onwaar (.F.).

Gebruik

Stel het kenmerk SpeedBar in op waar (.T.), als u van een knop een knoppenbalk-knop wilt maken. Een knoppenbalk-knop wordt bij de tabvolgorde van een formulier buiten beschouwing gelaten en kan dus ook geen focus krijgen door op *Tab* of *Shift-Tab* te drukken.

In Windows-toepassingen zijn knoppen op een knoppenbalk een snel alternatief voor menucommando's. Meestal worden deze knoppen gebruikt voor veel voorkomende taken als knippen, kopiëren en afdrukken.

Voorbeeld

Syntaxis operator NEW:

```
Vooruit = NEW PUSHBUTTON(this)
Vooruit.Onclick = WzgRecno
Vooruit.Text = "Vooruit"
Vooruit.SpeedBar = .T.
```

Syntaxis commando DEFINE:

```
DEFINE PUSHBUTTON Vooruit OF THIS;
  PROPERTY  Onclick WzgRecno,;
  Text "Vooruit", SpeedBar .T.
```

Zie ook

DEFINE, OnClick

SpinOnly

Met het kenmerk SpinOnly wordt bepaald of gebruikers een waarde in het tekstvak van een ringveld kunnen invoeren.

Kenmerk uit klasse
SPINBOX

Gegevenstype
Logisch

Standaardinstelling
De standaardinstelling van SpinOnly is onwaar (.F.).

Gebruik
Bij een ringveld kan een gebruiker waarden in het tekstvak typen of waarden selecteren met de pijlknoppen. Als u het kenmerk SpinOnly op onwaar (.F.) instelt, kan een waarde worden ingevoerd in het tekstvak. Als u het kenmerk SpinOnly op waar (.T.) instelt, kunnen alleen waarden worden ingevoerd die met de pijlknoppen kunnen worden geselecteerd.

Voorbeeld
Syntaxis operator NEW:

```
RV1 = NEW SPINBOX(this)
RV1.DataLink = "Landen->BNP"
RV1.SpinOnly = .T.,;
RV1.Height = 2
```

Syntaxis commando DEFINE:

```
DEFINE SPINBOX RV1 OF THIS;
  PROPERTY;
    DataLink "Landen->BNP",;
    SpinOnly .T., Height 2
```

Zie ook
CLASS SPINBOX

StatusMessage

Met het kenmerk StatusMessage wordt een melding gedefinieerd die op de statusbalk wordt weergegeven wanneer een object focus heeft.

Kenmerk uit klasse
BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, LISTBOX, MENU, OLE, PUSHBUTTON, RADIOBUTTON, SCROLLBAR, SPINBOX

Gegevenstype
Teken

Standaardinstelling
De standaardinstelling van StatusMessage is een lege tekenreeks.

Gebruik
Gebruik het kenmerk StatusMessage om instructies te geven aan de gebruiker wanneer deze een object selecteert.

Voorbeeld
Syntaxis operator NEW:

```
Afsluiten = NEW PUSHBUTTON(this)
Afsluiten.Onclick = Berekenen
Afsluiten.Text = "Afsluiten"
Afsluiten.StatusMessage = "Klik op Afsluiten om resultaat;
  te compileren"
* Berekenen is FUNCTIE waarmee handeling wordt uitgevoerd
* en die resulteert in .T. of waarde.
```

Syntaxis commando DEFINE:

```
DEFINE PUSHBUTTON Afsluiten OF THIS;
  PROPERTY  Onclick Berekenen,;
  Text "Afsluiten",;
  StatusMessage "Klik op Afsluiten om resultaat te compileren"
```

Zie ook
Text, ValidErrorMsg

Step

Met het kenmerk Step wordt bepaald hoe groot de stap is waarmee een waarde in een ringveld toeneemt of afneemt wanneer de gebruiker op een pijl klikt.

Kenmerk uit klasse
SPINBOX

Gegevenstype
Numeriek

Standaardinstelling
De standaardinstelling van Step is 1.

Gebruik

Gebruik het kenmerk Step om de stapwaarde te bepalen waarmee een gebruiker een numerieke waarde of datumwaarde kan verhogen of verlagen. Bij een programma waarin bijvoorbeeld met grote bedragen wordt gewerkt die worden uitgedrukt in veelvouden van *f*500,00, stelt u het kenmerk Step van een ringveld in op 500.

Voorbeeld

Syntaxis operator NEW:

```
Ring1 = NEW SPINBOX(this)
Ring1.Datalink = "Landen->BNP"
Ring1.Top = 2
Ring1.Left = 4
Ring1.Height = 2
Ring1.Step = 10
```

Syntaxis commando DEFINE:

```
DEFINE SPINBOX Ring1 OF THIS;
  PROPERTY Datalink "Landen->BNP",;
  Top 2, Left 4, Height 2, Step 10
```

Zie ook

RangeMax, RangeMin

Style

Met het kenmerk Style wordt gedefinieerd welke delen van een keuzelijst met invoervak kunnen worden gebruikt en welke delen automatisch worden weergegeven.

Kenmerk uit klasse

COMBOBOX

Gegevenstype

Numeriek

Standaardinstelling

De standaardinstelling van Style is 0 (eenvoudig).

Gebruik

Gebruik het kenmerk Style om te bepalen hoe de gebruiker waarden kan selecteren in een keuzelijst met invoervak.

De gebruiker selecteert een waarde in een keuzelijst met invoervak door de begintekens in een tekstvak te typen of door de waarde direct in de keuzelijst te selecteren. De

instelling van het kenmerk Style bepaalt of het tekstvak kan worden gebruikt en of de keuzelijst automatisch wordt weergegeven.

U kunt bij het kenmerk Style de volgende drie instellingen gebruiken:

Instelling	Omschrijving
0 - Eenvoudig	De keuzelijst wordt automatisch weergegeven.
1 - Vervolgkeuze	De gebruiker moet op de pijl klikken om de keuzelijst weer te geven.
2 - Vervolgkeuzelijst	De gebruiker moet op de pijl klikken om de keuzelijst weer te geven en kan geen waarden in het tekstvak typen.

Voorbeeld

Syntaxis operator NEW:

```
DEFINE COMBOBOX KLMIV OF THIS
  this.KLMIV.Style=1
  this.KLMIV.Datasource="FIELD Dieren->Naam"
  this.KLMIV.Top=4
  this.KLMIV.Left=6
  this.Width=20
  this.Height=12
```

Syntaxis commando DEFINE:

```
DEFINE COMBOBOX KLMIV OF THIS FROM 4,6 TO 26,16;
    PROPERTY;
      DataSource "FIELD Dieren->Naam",;
      Style 1
```

Zie ook

Enabled

Subscript( )

Het resultaat van de methode Subscript( ) is het rijnummer of het kolomnummer van een bepaald element in een array-object.

Kenmerk uit klasse

ARRAY

Gebruik

Gebruik de methode Subscript( ) als u het nummer van een element in een tweedimensionaal array-object kent en met de indextekens naar het element wilt verwijzen. De methode Subscript( ) lijkt op de functie ASUBSCRIPT( ).

U kunt bij de methode Subscript( ) de volgende twee parameters gebruiken:

- *<element Nuitdr>* Dit is het nummer van het element.

- *<rij/kolom Nuitdr>* Dit is het getal 1 of 2. Met dit getal bepaalt u of u als resultaat het indexteken van de rij of van de kolom in het array-object wilt. Als u voor de parameter *<rij/kolom Nuitdr>* het getal 1 opgeeft, resulteert de methode Subscript( ) in het indextekennummer van de rij. Als voor de parameter *<rij/kolom Nuitdr>* het getal 2 is gedefinieerd, resulteert de methode Subscript( ) in het indextekennummer van de kolom.

Als u zowel het rij- als het kolomnummer van een element in een tweedimensionaal array-object wilt bepalen, gebruikt u de methode Subscript( ) tweemaal, één keer met de waarde 1 voor de parameter *<rij/kolom Nuitdr>* en één keer met de waarde 2 voor de parameter *<rij/kolom Nuitdr>*. Als het elementnummer bijvoorbeeld 13 is, kunt u op de volgende manier de indextekens bepalen:

```
Subscript(Form.eenArray,13,1)   && Resultaat is indexteken van rij
Subscript(Form.eenArray,13,2)   && Resultaat is indexteken van kolom
```

In ééndimensionale array-objecten is het elementnummer gelijk aan het indexteken en hoeft u de methode Subscript( ) niet te gebruiken. Dat wil zeggen dat het resultaat van Subscript(3) gelijk is aan 3, van Subscript(5) gelijk aan 5 enzovoorts.

De methode Subscript( ) is tegengesteld aan de methode Element( ), bij deze laatste methode geeft u de indextekens van het element op en verkrijgt u het elementnummer.

Voorbeeld

Zie Scan( ) voor een voorbeeld van het gebruik van Subscript( ) in combinatie met array-objecten.

Zie ook

ASUBSCRIPT( ), Element( )

SysMenu

Met het kenmerk SysMenu wordt bepaald of een formulier een systeemmenu heeft.

Kenmerk uit klasse

FORM

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van SysMenu is waar (.T.).

Gebruik

Stel het kenmerk SysMenu in op waar (.T.) om in een formulier een systeemmenu te maken. Het systeemmenu is in Windows een standaardvoorziening en kan worden geopend via het *Symbool Systeemmenu*, de knop in de linkerbovenhoek van een formulier.

In het systeemmenu vindt u de volgende opdrachten:

- **Vorig formaat**. Hiermee worden de afmetingen van het formulier hersteld die het had voordat het formulier tot een maximumvenster werd vergroot.
- **Verplaatsen**. Hiermee kan de gebruiker het formulier verplaatsen met de pijltoetsen.
- **Formaat wijzigen**. Hiermee kan de gebruiker de afmetingen van het formulier wijzigen met de pijltoetsen..
- **Pictogram**. Hiermee wordt het formulier tot een pictogram verkleind.
- **Maximumvenster**. Hiermee wordt het formulier vergroot tot een maximumvenster.
- **Sluiten**. Hiermee wordt het formulier gesloten.

Als u een formulier opent met de methode ReadModal( ) of de functie READMODAL( ), bevat het systeemmenu slechts de opdrachten **Verplaatsen** en **Sluiten**.

Het is gebruikelijk om het kenmerk SysMenu op onwaar (.F.) in te stellen als u een dialoogvenster in een Windows-applicatie maakt.

Opmerking Als het kenmerk MDI van een formulier op waar (.T.) is ingesteld, wordt de instelling van het kenmerk SysMenu genegeerd en heeft dat formulier altijd een systeemmenu.

Voorbeeld

Syntaxis operator NEW:

```
F1=NEW InvoerForm()
CLASS InvoerForm OF FORM
  this.MDI = .F.
  this.Text = "Invoeren"
  this.Top = 2
  this.Left = 2
  this.Width = 38
  this.Height = 18
  this.SysMenu = .F.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM InvoerForm ;
  Property SysMenu .F., MDI .F.,;
  Top 2, Left 2, Width 38, Height 18
```

Zie ook

Moveable, Sizeable

TabStop

Met het kenmerk TabStop wordt bepaald of de gebruiker een object kan selecteren door op *Tab* of *Shift-Tab* te drukken.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, LISTBOX, OLE, PUSHBUTTON, RADIOBUTTON, SCROLLBAR, SPINBOX

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van TabStop is waar (.T.).

Gebruik

Stel het kenmerk TabStop in op onwaar (.F.) als u een object wilt uitsluiten van de tabvolgorde van het hoofdformulier. Een formulier kan bijvoorbeeld een knop bevatten waarmee routine wordt gestart die slechts zelden wordt gebruiken. Als u het kenmerk TabStop van die knop op onwaar (.F.) instelt, wordt deze niet geselecteerd wanneer de gebruiker op *Tab* of *Shift-Tab* drukt.

Voorbeeld

Syntaxis operator NEW:

```
Bedrijf = NEW ENTRYFIELD(this)
Bedrijf.Datalink = "Klanten->Bedrijf"
Bedrijf.TabStop = .F.
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Bedrijf OF THIS;
  PROPERTY Datalink "Klanten->Bedrijf",;
  TabStop .F.
```

Zie ook

Before, _curobj, Group

Terminate( )

Met de methode Terminate( ) wordt een koppeling met een DDE server-toepassing beëindigd.

Kenmerk uit klasse

DDELINK

Gebruik

Gebruik de methode Terminate( ) om een DDE-koppeling tussen dBASE en een server-toepassing te verbreken.

Door middel van de methode Terminate( ) wordt een koppeling tussen dBASE en de server-toepassing verbroken maar wordt de server-toepassing zelf niet afgesloten.

Als u een DDE-koppeling beëindigt met de methode Terminate( ), kunt u deze weer herstellen met Reconnect. Als u de koppeling beëindigt met de methode Release( ), kan deze niet meer worden hersteld en moet u het DDELink-object opnieuw maken.

Voorbeeld

```
PUBLIC KoppObj
KoppObj = NEW DDELINK()
KoppObj.Initiate("QPW","Demo.WB1")
* Volgende programmabewerkingen gereed; koppeling
* niet langer nodig
KoppObj.Terminate()
```

Zie ook

Advise( ), Execute( ), Initiate( ), OnNewValue, Peek( ), Poke( ), Reconnect( ), Server, TimeOut, Topic, Unadvise( )

Text

Met het kenmerk Text wordt een tekenreeks gedefinieerd die in of naast een object moet worden getoond.

Kenmerk uit klasse

BROWSE, CHECKBOX, FORM, MENU, PUSHBUTTON, RADIOBUTTON, RECTANGLE, TEXT

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van Text is een tekenreeks die in dBASE aan het object wordt toegekend wanneer u dat maakt. In dBASE wordt bijvoorbeeld automatisch de tekst KNOP1 aan het kenmerk Text toegekend van de eerste knop die u maakt.

Gebruik

Gebruik het kenmerk Text om objecten te voorzien van een aanduiding of om opdrachten op te nemen in menu's. Als met een knop bijvoorbeeld een formulier wordt gesloten, kunt u het kenmerk Text instellen op "Sluiten" of "Annuleren". Om een tekenreeks op een formulier weer te geven, maat u een tekstobject en stelt u het kenmerk Text in op de gewenste tekenreeks.

Gebruik een *opdrachtletter* om de gebruiker een object focus te laten geven of een menu-opdracht te laten selecteren door op een letter te drukken. U definieert een teken als een opdrachtletter door deze vooraf te laten gaan door een ampersand (&). In het volgende voorbeeld wort de letter S gedefinieerd als de opdrachtletter:

```
MijnForm.MijnMenu.EersteOpdracht.Text = "&Sluiten"
```

Wanneer het object wordt getoond, is de opdrachtletter onderstreept.

Voorbeeld

Syntaxis operator NEW:

```
Afsluiten = NEW PUSHBUTTON(this)
Afsluiten.Text = "&Afsluiten"
Afsluiten.Width = 15
Afsluiten.Onclick = {;Form.Close()}
```

Syntaxis commando DEFINE:

```
DEFINE PUSHBUTTON Afsluiten OF THIS AT 11,13 ;
  PROPERTY Text "&Afsluiten", Width 15,;
  OnClick {;Form.Close()}
```

Zie ook

StatusMessage

TimeOut

Met het kenmerk TimeOut wordt de tijd in seconden gedefinieerd die in dBASE op een transactie wordt gewacht voordat een fout wordt geconstateerd.

Kenmerk uit klasse

DDELINK

Gegevenstype

Numeriek

Standaardinstelling

De standaardinstelling van TimeOut is 10.00.

Gebruik

Gebruik het kenmerk TimeOut om de tijd te beperken die in dBASE wordt gewacht op het succesvol afronden van een DDE-transactie.

Soms treden fouten op wanneer vanuit dBASE gegevens worden uitgewisseld met een server-toepassing of wanneer instructies naar die toepassing worden gestuurd. Steeds wanneer geprobeerd wordt gegevens uit te wisselen of instructies te versturen, wordt het aantal seconden dat u bij het kenmerk TimeOut hebt opgegeven gewacht. Als de transactie niet wordt afgerond binnen de opgegeven tijd, wordt een foutmelding weergegeven.

Als u DDE-koppelingen in een netwerk gebruikt, kan het nodig zijn het kenmerk TimeOut op een hogere waarde in te stellen.

Voorbeeld

```
PUBLIC KoppObj
KoppObj = NEW DDELINK()
KoppObj.Timeout = 30
KoppObj.Initiate("QPW","Demo.WB1")
```

Zie ook

Advise( ), Execute( ), Initiate( ), OnNewValue, Peek( ), Poke( ), Server, Terminate( ), Topic, Unadvise( )

Toggle

Met het kenmerk Toggle wordt bepaald of de gebruiker in een bladerobject kan schakelen tussen twee weergavemodi.

Kenmerk uit klasse

BROWSE

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Toggle is waar (.T.).

Gebruik

Stel het kenmerk Toggle in op onwaar (.F.) om in een bladerobject maar één weergavemodus mogelijk te maken.

In een bladerobject kunnen meerdere records (bladermodus) maar ook slechts één record (bewerkmodus) worden weergegeven. Door op *F2* te drukken kan de gebruiker tussen deze twee weergavemodi schakelen. Als u het kenmerk Mode op 0 instelt, is de standaardinstelling de bladermodus. Als u het kenmerk Mode op 1 of 2 instelt, is de standaardinstelling de bewerkmodus. Als u het kenmerk Toggle op onwaar (.T.) instelt, kan alleen in standaardmodus worden gebruikt.

Voorbeeld

Syntaxis operator NEW:

```
BedrijfBladeren = New BROWSE(this)
  BedrijfBladeren.Top = 3
  BedrijfBladeren.Left = 1
  BedrijfBladeren.Width = 60
  BedrijfBladeren.Alias = "Bedrijf"
  BedrijfBladeren.Toggle = .F.
```

Syntaxis commando DEFINE:

```
DEFINE BROWSE BedrijfBladeren OF THIS;
   FROM 3,1 TO 13,40;
     Property;
      Alias "Bedrijf",;
      Toggle .F.
```

Zie ook

Browse, Append, Delete

Top

Met het kenmerk Top wordt de positie gedefinieerd van de bovenste rand van een object ten opzichte van het hoofdformulier of ten opzichte van het bureaublad van Windows (als het object geen hoofdformulier heeft).

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LINE, LISTBOX, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX

Gegevenstype

Numeriek

Gebruik

Gebruik het kenmerk Top in combinatie met het kenmerk Left om de positie van een object in een formulier of op het bureaublad van Windows te definiëren (het kenmerk

Left geeft de positie aan van de linkerrand). U kunt de kenmerken Top en Left instellen op gehele en gebroken getallen.

Als u de clausule met FORM of AT van het commando DEFINE hebt gebruiker *en* een waarde hebt toegewezen aan het kenmerk Top, wordt de instelling van het kenmerk Top gebruikt.

Elke eenheid van de waarde die u aan het kenmerk Top toewijst, is gelijk aan de gemiddelde hoogte van de tekens in het actieve font van het hoofdformulier. Als u de instelling van het kenmerk Top van een invoervak bijvoorbeeld verhoogt met 5,5, komt het invoervak 5,5 tekens hoger te staan.

Voorbeeld

Syntaxis operator NEW:

```
Afsluiten=NEW PUSHBUTTON(this)
Afsluiten.Top = 5
Afsluiten.Left = 2
Afsluiten.Width = 17
Afsluiten.Text = "Programma afsluiten"
Afsluiten.OnClick = {;Form.Close()}
```

Syntaxis commando DEFINE:

```
DEFINE PUSHBUTTON Afsluiten OF Invoer;
  PROPERTY Top 5, Left 2, Width 17,;
  Text "Programma afsluiten", OnClick {;Form.Close()}
```

Zie ook

Bottom, Height, Left, Right, ScaleFontName, ScaleFontSize, Width

Topic

Het kenmerk Topic is ingesteld op de naam van het server-document dat is geopend via een DDELink-object of op het onderwerp van een DDETopic-object.

Kenmerk uit klasse

DDELINK, DDETOPIC

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van Topic is een lege tekenreeks.

Gebruik

U gebruikt het kenmerk Topic om het volgende te doen:

- De naam van het server-document op te vragen die is doorgegeven aan de methode Initiate( ) van een DDELink-object.
- Het onderwerp van een DDETopic-object op te vragen.

Het kenmerk Topic gebruiken bij DDELink-objecten

Met de methode Initiate( ) van een DDELink-object wordt een *DDE-koppeling* gemaakt tussen dBASE en een externe Windows-toepassing (de *server*). Vervolgens wordt een van de gegevensbestanden (het *server-document)* in die toepassing geopend.

U moet de volgende twee parameters opgeven bij de methode Initiate( ):

- *<server>* Dit is het hoofdprogrammabestand van de server-toepassing.
- *<onderwerp>* Dit is de bestandsnaam van het server-document.

Het kenmerk Topic is ingesteld op de naam die opgeeft voor de parameter *<onderwerp>*.

Het kenmerk Topic gebruiken bij DDETopic-objecten

Door middel van kenmerk Topic van een DDETopic-object kan dit object worden onderscheiden van andere DDETopic-objecten. In een dBASE server-applicatie kunnen bijvoorbeeld twee DDETopic-objecten worden gemaakt, een met als onderwerp NASDAQ en de ander met als onderwerp AMEX:

```
xServer1 = NEW DDETOPIC("NASDAQ")
xServer2 = NEW DDETOPIC("AMEX")
```

Voorbeeld

```
PUBLIC KoppObj
KoppObj = NEW DDELINK()
KoppObj.Initiate("QPW","Demo.WB1")
* Volgende programmabewerkingen
IF KoppObj.Server="QPW" .AND. ;
   KoppObj.Topic = "Demo.WB1"
      mWaarde1=KoppObj.Peek("A:A1")
ENDIF
```

Zie ook

Advise( ), Execute( ), Initiate( ), OnNewValue, Peek( ), Poke( ), Server, Terminate( ), TimeOut, Unadvise( )

Unadvise( )

Met de methode UnAdvise( ) wordt de server-toepassing gevraagd wijzigingen in het server-document niet langer door te geven aan de client-toepassing.

Kenmerk uit klasse

DDELINK

Gebruik

Gebruik de methode Unadvise( ) om een automatische koppeling met gegevens in een server-document te beëindigen. Bij een automatische koppeling, die u maakt met de methode Advise( ), geeft de server-toepassing wijzigingen aan dBASE door.

Een server-document is een bestand dat u opent in een externe toepassing. Bij een programma voor het uitwisselen van gegevens kan bijvoorbeeld Quattro Pro voor Windows worden gestart, een van de spreadsheet-bestanden worden geopend en vervolgens een koppeling worden gemaakt met een van de cellen in die spreadsheet. Als de koppeling niet meer wordt gebruikt, kan deze worden beëindigd met de methode Unadvise( ).

U moet bij de methode Unadvise( ) de parameter *<element>* opgeven. Met deze parameter worden de gegevens in het server-document aangeduid. Deze gegevens kunnen bestaan uit één element zoals een veld in een tabel of een cel in een spreadsheet. U kunt bijvoorbeeld een automatische koppeling met cel C2 op pagina A van een spreadsheet in Quattro Pro beëindigen door de parameter "A:C2" op te geven.

Voorbeeld

```
PUBLIC KoppObj
KoppObj = NEW DDELINK()
KoppObj.OnNewValue = VerwerkWaarde;
                && Codeblok of functie-aanwijzer
KoppObj.Initiate("QPW","Demo.WB1")
KoppObj.Advise("A:A1");
                && Wijzigingen in cel A:A1 doorgegeven.
* Volgende programmabewerkingen afgerond.
KoppObj.UnAdvise("A:A1")
```

Zie ook

Advise( ), Execute( ), Initiate( ), OnNewValue, Peek( ), Poke( ), Server, Terminate( ), TimeOut, Topic

UpBitmap

Met het kenmerk UpBitmap wordt een grafische afbeelding opgegeven die op een knop moet worden weergegeven wanneer deze wordt geselecteerd.

Kenmerk uit klasse

PUSHBUTTON

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van UpBitmap is een lege tekenreeks.

Gebruik

Gebruik het kenmerk UpBitmap om zichtbaar te maken dat een knop is ingeschakeld en de gebruiker niet op de knop klikt.

Het kenmerk UpBitmap kan op een van de volgende drie manieren worden ingesteld:

1. RESOURCE *<bronidentificatie> <DLL-naam>* Hiermee wordt de bron van de bitmap opgegeven evenals het DLL-bestand waarin de bitmap staat (een DLL-bestand is een vooraf gecompileerde bibliotheek met externe routines en bronnen die niet in de dBASE-taal is geschreven, maar bijvoorbeeld in C of Pascal). U kunt de bronidentificatie opvragen met een resource-editor zoals Resource Workshop.
2. FILENAME *<bestandsnaam>* Hiermee geeft u een bitmap-bestand op (een dergelijk bestand heeft meestal de extensie .BMP).
3. BINARY *<binair veld>* Hiermee wordt de naam van een binair veld met bitmap-afbeeldingen opgegeven. Als u deze optie gebruikt, moet het hoofdformulier zijn gebaseerd op een tabel of query die het opgegeven veld bevat. U geeft deze tabel of query op met het kenmerk View van het formulier. De afbeelding die wordt weergegeven is steeds afkomstig uit het huidige record, als de gebruiker dus naar een ander record gaat, verandert de afbeelding.

Als u een tekenreeks opgeeft met het kenmerk Text en een afbeelding met het kenmerk UpBitmap, worden zowel de afbeelding als de tekst getoond.

Opmerking U kunt een bitmap-bestand selecteren in het dialoogvenster **Bitmap kiezen**. U opent het dialoogvenster **Bitmap kiezen** door op de hulpmiddelenknop naast de optie **UpBitmap** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
Vooruit = NEW PUSHBUTTON(this)
Vooruit.Onclick = ShowInfo
Vooruit.Text = "Database"
Vooruit.UpBitmap = "Resource #1904 Bwcc.dll"
Vooruit.DownBitmap = "Resource Empty Resource.dll"
Vooruit.StatusMessage = "Klik op 'Database' voor;
  landgegevens"
```

Syntaxis commando DEFINE:

```
DEFINE PUSHBUTTON Vooruit OF THIS;
  PROPERTY  Onclick ShowInfo,;
  Text "Database",;
  UpBitmap "Resource #1904 Bwcc.dll",;
```

```
DownBitmap "Resource Empty Resource.dll",;
StatusMessage "Klik op 'Database' voor;
landgegevens"
```

Zie ook

DEFINE, DisabledBitmap, DownBitmap, Enabled, FocusBitmap

Valid

Met het kenmerk Valid wordt een voorwaarde opgegeven die in (.T.) moet resulteren, voordat de gebruiker focus van een object kan verwijderen.

Kenmerk uit klasse

ENTRYFIELD, SPINBOX

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk Valid om gegevens te valideren. Bij een applicatie kan het kenmerk Valid van een invoervak bijvoorbeeld worden gebruikt om te voorkomen dat een gebruiker ongeldige rekeningnummers invoert. Als de gebruiker een onjuiste waarde typt, kan de focus niet naar een ander object worden verplaatst. Eerst moet de gebruiker de gegevens corrigeren.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk Valid het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

De subroutine die u bij het kenmerk Valid opgeeft moet resulteren in een logische waarde (waar of onwaar).

Het kenmerk Valid lijkt op de kenmerken RangeMax en RangeMin. Met de kenmerken RangeMax en RangeMin stelt u echter alleen een boven- en ondergrens in. Met het kenmerk Valid kunt u andere voorwaarden opgeven. Met een onder- en bovengrens kunt u bijvoorbeeld geen waarden uitsluiten die niet op elkaar volgen zoals 2, 7 en 9. Daarvoor moet u een subroutine opgeven bij het kenmerk Valid.

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **Valid** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
Datum = NEW ENTRYFIELD(this)
Datum.Datalink = "Afnemers->BalnsDatum"
Datum.Valid = Wrdecntr
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Datum OF THIS;
  PROPERTY Datalink "Afnemers->BalnsDatum",;
  Valid Wrdecntr
```

Zie ook

RangeMax, RangeMin, RangeRequired, ValidErrorMsg, ValidRequired, When

ValidErrorMsg

Met het kenmerk ValidErrorMsg wordt een tekenreeks opgegeven die op de statusbalk wordt getoond als het kenmerk Valid van een invoervak resulteert in onwaar (.F.).

Kenmerk uit klasse

ENTRYFIELD, SPINBOX

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van ValidErrorMsg is "Ongeldige invoer".

Gebruik

Gebruik het kenmerk ValidErrorMsg om instructies of een waarschuwing te tonen als de gebruiker ongeldige gegevens invoert.

Het kenmerk Valid wordt bijvoorbeeld gebruikt om te voorkomen dat ongeldige rekeningnummers worden ingevoerd. Als dat gebeurt, resulteert het kenmerk Valid in onwaar (.F.) en wordt de bij het kenmerk ValidErrorMsg opgegeven melding op de statusbalk weergegeven.

Voorbeeld

Syntaxis operator NEW:

```
Datum = NEW ENTRYFIELD(this)
Datum.Datalink = "Afnemers->BalnsDatum"
Datum.Valid = Wrdecntr
Datum.ValidErrorMsg = "Ongeldige datum"
* Wrdecntr is FUNCTIE waarmee handeling wordt uitgevoerd
* en die resulteert in .T. of waarde.
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Datum OF THIS;
  PROPERTY Datalink "Afnemers->BalnsDatum",;
  Valid Wrdecntr,;
  ValidErrorMsg "Ongeldige datum"
```

Zie ook

StatusMessage, Valid, ValidRequired

ValidRequired

Met het kenmerk ValidRequired wordt bepaald of de instelling van het kenmerk Valid geldt voor alle gegevens of alleen voor nieuwe gegevens.

Kenmerk uit klasse

ENTRYFIELD, SPINBOX

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van ValidRequired is onwaar (.F.).

Gebruik

Stel het kenmerk ValidRequired in op waar (.T.) om zowel bestaande als nieuwe gegevens te controleren. Het kenmerk ValidRequired werkt alleen als u met het kenmerk Valid een voorwaarde definieert.

Gewoonlijk stelt u het kenmerk ValidRequired in op waar (.T.) als u een validatievoorwaarde verandert en bestaande gegevens moet controleren en aanpassen. In een boekhouding van een bedrijf kan bijvoorbeeld een cijfer worden toegevoegd aan de rekeningnummers en de instelling van het kenmerk Valid van een invoervak worden gewijzigd zodat het extra cijfer kan worden ingevoerd. Als het kenmerk ValidRequired is ingesteld op waar (.T.), worden ook bestaande rekeningnummers gecontroleerd en als het extra cijfer ontbreekt, moet de gebruiker dat toevoegen.

Het kenmerk Valid lijkt op de kenmerken RangeMax en RangeMin. Met de kenmerken RangeMax en RangeMin stelt u echter alleen een boven- en ondergrens in. Met het kenmerk Valid kunt u andere voorwaarden opgeven. Met een onder- en bovengrens kunt u bijvoorbeeld geen waarden uitsluiten die niet op elkaar volgen zoals 2, 7 en 9. Daarvoor moet u een subroutine opgeven bij het kenmerk Valid.

Voorbeeld

Syntaxis operator NEW:

```
Datum = NEW ENTRYFIELD(this)
Datum.Datalink = "Afnemers->BalnsDatum"
Datum.Valid = Wrdecntr
Datum.ValidRequired = .T.
Datum.ValidErrorMsg = "Ongeldige datum"
* Wrdecntr is FUNCTIE waarmee handeling wordt uitgevoerd
* en die resulteert in .T. of waarde.
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD Datum OF THIS;
  PROPERTY Datalink "Afnemers->BalnsDatum",;
  Valid Wrdecntr, ValidRequired .T.,;
  ValidErrorMsg "Ongeldige datum"
```

Zie ook

RangeMax, RangeMin, RangeRequired, Valid, ValidErrorMsg

Value

Het kenmerk Value is ingesteld op de waarde van een object.

Kenmerk uit klasse

CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, LISTBOX, RADIOBUTTON, SCROLLBAR, SPINBOX

Gegevenstype

Numeriek, zwevend, teken of datum

Standaardinstelling

De standaardinstelling van Value verschilt per object. De standaardinstelling bij een invoervak is bijvoorbeeld de waarde die in het tabelveld (waaraan het invoervak is gekoppeld) van het eerste record is ingevoerd. In geval van een aankruisvakje, is de standaardinstelling waar (.T.) (een aankruisvakje is standaard aangekruist).

Gebruik

Gebruik het kenmerk Value om een beginwaarde voor een object op te geven of een query te maken van de door de gebruiker ingevoerde gegevens. Met het kenmerk Value kunt u bij een ringveld bijvoorbeeld de beginwaarde instellen die in het tekstvak wordt weergegeven. U kunt ook de instelling van het kenmerk Value van een invoervak opvragen om de waarde te verkrijgen die de gebruiker heeft ingevoerd.

Voorbeeld

Syntaxis operator NEW:

```
BDatum = NEW ENTRYFIELD(this)
BDatum.Datalink = "Afnemers->BalnsDatum"
BDatum.Value = DATE()
```

Syntaxis commando DEFINE:

```
DEFINE ENTRYFIELD BDatum OF THIS;
  PROPERTY Datalink "Afnemers->BalnsDatum",;
  Value DATE()
```

Zie ook

DataLink, DataSource, LISTSELECTED( ), Selected( ), STORE

Vertical

Het kenmerk Vertical geeft aan of een schuifbalk horizontaal of verticaal wordt weergegeven.

Kenmerk uit klasse

SCROLLBAR

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Vertical is waar (.T.).

Gebruik

Gebruik het kenmerk Vertical om een schuifbalk aan te passen aan andere objecten in een formulier of om de opmaak van het formulier te verbeteren. In een formulier kan bijvoorbeeld niet voldoende ruimte zijn voor een horizontale schuifbalk, of een verticale schuifbalk naast een ringveld kan een weinig fraai formulier opleveren.

Voorbeeld

Syntaxis operator NEW:

```
IncNum = NEW SCROLLBAR(this)
IncNum.Top = 7
IncNum.Left = 3
IncNum.Datalink = "Num"
IncNum.Vertical = .F.
```

Syntaxis commando DEFINE:

```
DEFINE SCROLLBAR IncNum OF THIS;
  PROPERTY DataLink "Num", Vertical .F.,;
  Top 7, Left 3
```

Zie ook

DataLink, Height, Left, Top, Width

View

Met het kenmerk View wordt de naam van een query of een tabel opgegeven op basis waarvan een formulier is gemaakt.

Kenmerk uit klasse

FORM

Gegevenstype

Teken

Standaardinstelling

De standaardinstelling van View is een lege tekenreeks.

Gebruik

Gebruik het kenmerk View om te bepalen welke tabellen automatisch worden geopend wanneer het formulier wordt geopend. U kunt bij een query (.QBE, .VUE) meerdere tabellen toewijzen maar ook slechts één tabel (.DBF). Hoewel u het kenmerk View niet hoeft in te stellen, moet u dat wel doen als in het formulier gebruik wordt gemaakt van gegevens in een tabel. Anders moet u steeds als u het formulier opent, eerst de tabel of query openen.

Als u het formulier aan een of meer tabellen hebt gekoppeld, moet u objecten als invoervakken en aankruisvakjes aan bepaalde velden in de tabel koppelen.

Opmerking In het dialoogvenster **Weergave kiezen** kunt u een waarde opgeven voor het kenmerk View. In dit dialoogvenster kunt u een query of een tabel selecteren. U opent het

dialoogvenster **Weergave kiezen** door op de hulpmiddelenknop naast de optie **View** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL
F1=NEW InvoerForm()
CLASS InvoerForm OF FORM
this.Text = "Invoerformulier"
this.Top = 2
this.Left = 2
this.Width = 38
this.Height = 18
this.View = "Bedr94.QBE"
* Bedr94.QBE is querybestand waarop
* invulformulier is gebaseerd.
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM InvoerForm;
  Property Text "Invoerformulier",;
  View "Bedr94.QBE",;
  Top 2, Left 2, Width 38, Height 18
```

Zie ook

Alias, DataLink, DataSource

Visible

Met het kenmerk Visible wordt bepaald of een object zichtbaar is of is verborgen.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LINE, LISTBOX, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Visible is waar (.T.).

Gebruik

Gebruik het kenmerk Visible als u een object alleen wilt weergeven indien aan een bepaalde voorwaarde is voldaan of een bepaalde handeling is uitgevoerd. U kunt bijvoorbeeld voorwaarden opgeven bij actiekenmerken (zoals OnClick of Valid) waardoor het kenmerk Visible op waar wordt ingesteld als de gebruiker op een object klikt of een bepaalde waarde invoert.

Voorbeeld

Syntaxis operator NEW:

```
Ring1 = NEW SPINBOX(this)
Ring1.Datalink = "Waarde1"
Ring1.Top = 2
Ring1.Left = 4
Ring1.Height = 2
Ring1.Visible = .F.
```

Syntaxis commando DEFINE:

```
DEFINE SPINBOX Ring1 OF THIS;
  PROPERTY Datalink "Waarde1",;
  Top 2, Left 4, Height 2, Visible .F.
```

Zie ook

Enabled

When

Met het kenmerk When wordt een voorwaarde opgegeven die in (.T.) moet resulteren voordat de gebruikt een object focus kan geven.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, LISTBOX, PUSHBUTTON, RADIOBUTTON, SCROLLBAR, SPINBOX

Gegevenstype

Functie-aanwijzer of codeblok

Gebruik

Gebruik het kenmerk When om te bepalen wanneer een object zichtbaar moet zijn voor de gebruiker. In een applicatie waarmee vertrouwelijke klantgegevens worden beheerd kan een bladerobject bijvoorbeeld pas focus krijgen, wanneer de gebruiker het juiste wachtwoord in een invoervak typt.

Net zoals bij de andere actiekenmerken, kunt u bij het kenmerk When het volgende opgeven:

- Functie
- Procedure
- Codeblok

Zie Hoofdstuk 4 in *Programmeren* voor informatie over deze subroutines. Zie Hoofdstuk 14 in *Programmeren* voor informatie over het schrijven van actie-afhandelingsroutines.

De subroutine die u bij het kenmerk When opgeeft moet resulteren in een logische waarde (waar of onwaar).

Opmerking U kunt een subroutine schrijven met de Procedure-editor, een venster waarin u programmacode kunt invoeren. U opent de Procedure-editor door op de hulpmiddelenknop naast de optie **When** in het kenmerkenvenster te klikken.

Voorbeeld

Syntaxis operator NEW:

```
USE Bedrijf
* Formulierdefinitie
  VlgRec = NEW PUSHBUTTON(this)
  VlgRec.Text = "Volgende"
  VlgRec.When = {;.NOT. EOF()}
  VlgRec.OnClick = {;SKIP}
```

Syntaxis commando DEFINE:

```
DEFINE PUSHBUTTON VlgRec OF THIS;
  PROPERTY;
    Text "Volgende",;
    When {;.NOT. EOF()},;
    OnClick {;SKIP}
```

Zie ook

Valid

Width

Met het kenmerk Width wordt de breedte van een object gedefinieerd.

Kenmerk uit klasse

BROWSE, CHECKBOX, COMBOBOX, EDITOR, ENTRYFIELD, FORM, IMAGE, LINE, LISTBOX, OLE, PUSHBUTTON, RADIOBUTTON, RECTANGLE, SCROLLBAR, SPINBOX, TEXT

Gegevenstype

Numeriek

Gebruik

Gebruik het kenmerk Width in combinatie met het kenmerk Height om de afmetingen van een object aan te passen.

Met het kenmerk Width definieert u de afstand tussen de linkerrand en de rechterrand. U kunt zowel gehele als gebroken getallen opgeven.

Elke eenheid van de waarde die u aan het kenmerk Width toewijst, is gelijk aan de gemiddelde breedte van de tekens in het actieve font van het hoofdformulier. Als u het kenmerk Width bijvoorbeeld instelt op 15.5, wordt het invoervak 15,5 tekens breed.

Voorbeeld

Syntaxis operator NEW:

```
Handl = NEW PUSHBUTTON(this)
Handl.Text = "Openstaande rekeningen zoeken"
Handl.Onclick = Zoeken
Handl.Width = 40.3
Handl.Top = 11
Handl.Left = 13
```

Syntaxis commando DEFINE:

```
DEFINE PUSHBUTTON Handl OF THIS ;
  PROPERTY Text "Openstaande rekeningen zoeken",;
  Top 11, Left 13, Width 40.3
  OnClick Zoeken
```

Zie ook

Left, Height, ScaleFontName, ScaleFontSize, Top

WindowState

Met het kenmerk WindowState wordt bepaald of een formulier als pictogram, maximumvenster of met de oorspronkelijke afmetingen wordt weergegeven.

Kenmerk uit klasse

FORM

Gegevenstype

Numeriek

Standaardinstelling

De standaardinstelling van WindowState is 0 (normaal).

Gebruik

Gebruik het kenmerk WindowState om een formulier tot een maximumvenster te vergroten, tot een pictogram te verkleinen of om de oorspronkelijke afmetingen te herstellen.

U kunt het kenmerk WindowState op de volgend waarden instellen:

Instelling	Omschrijving
0 - Normaal	De oorspronkelijke afmetingen van het formulier worden hersteld.
1 - Pictogram	Het formulier wordt verkleind tot een pictogram.
2 - Vergroot	Het formulier wordt net zo groot als het werkgebied van het bureaublad.

Als u het kenmerk WindowState op 0 instelt, wordt het zogenaamde *vorige formaat* hersteld.

Voorbeeld

Syntaxis operator NEW:

```
LOCAL f
f=NEW Reizen()
f.OPEN()
CLASS Reizen OF FORM
  this.Text = "Reisschema"
  this.Top = 2
  this.Left = 2
  this.Width = 38
  this.Height = 18
  this.WindowState = 2
ENDCLASS
```

Syntaxis commando DEFINE:

```
DEFINE FORM Reizen FROM 2,2 TO 20,40;
  PROPERTY Text "Reisschema",;
  Top 2, Left 2, Width 38, Height 18.;
  WindowState 2
OPEN FORM REIZEN
```

Zie ook

Moveable, Sizeable

Wrap

Het kenmerk Wrap geeft aan of in een editor-object bij het invoeren van tekst met automatische regelovergangen wordt gewerkt.

Kenmerk uit klasse

EDITOR

Gegevenstype

Logisch

Standaardinstelling

De standaardinstelling van Wrap is onwaar (.F.).

Gebruik

Stel het kenmerk Wrap in op waar (.T.) als u in dBASE met automatische regelovergangen wilt werken als een tekst langer is dan een editor-object. Stel het kenmerk Wrap in op onwaar (.F.) als u de lengte van elke tekstregel wilt beperken.

Voorbeeld

Syntaxis operator NEW:

```
Ed1=NEW EDITOR(this)
  Ed1.Wrap = .T.
  Ed1.Top = 4
  Ed1.Left = 37
  Ed1.Height = 12
  Ed1, Width = 32
  Ed1.DataSource = "MEMO Contact->Notities"
```

Syntaxis commando DEFINE:

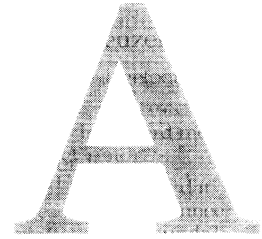
```
DEFINE EDITOR Ed1 OF THIS;
  PROPERTY Top 4, Left 37, Height 12, Width 32,;
    Wrap .T.,;
    DataSource "MEMO Contact->Notities"
```

Zie ook

Width, Height

Appendices

Appendix



Verschillen met dBASE IV 2.0

In deze appendix worden de verschillen beschreven tussen de dBASE-taal van dBASE IV versie 2.0 en die van dBASE voor Windows. Deze appendix bevat een overzicht van nieuwe en uitgebreide taalelementen, alsmede van de taalelementen die niet langer worden ondersteund.

Nieuwe taalelementen

Element	Omschrijving
ACOPY( )	Elementen kopiëren van de ene arrray naar de andere.
ADEL( )	Een element, rij of kolom uit een array verwijderen.
ADIR( )	Bestandskenmerken of bestanden uit een directory (naam, grootte, datum, tijd en DOS-attributen) opslaan in een array.
AELEMENT( )	Het resultaat is het nummer van het opgegeven element in een één- of tweedimensionale array.
AFIELDS( )	Informatie over de structuur van een tabel opslaan in een array.
AFILL( )	Een waarde invoegen in een of meer elementen van een array.
AGROW( )	Een element, rij of kolom toevoegen aan een array.
AINS( )	De waarde .F. invoegen in een array.
ALEN( )	Het resultaat is het aantal elementen, rijen of kolommen in een array.
ANSI( )	Het resultaat is de ANSI-waarde die overeenkomt met een teken uit de opgegeven codetabel van DOS.
ARESIZE( )	De grootte van een array vergoten of verkleinen.
ASCAN( )	Een uitdrukking zoeken in een array. Het resultaat is het nummer van het eerste element dat overeenkomt met de uitdrukking.
ASORT( )	De elementen in een array sorteren.
ASUBSCRIPT( )	Het resultaat is het nummer van de rij of kolom van een array waarin zich het opgegeven element bevindt.

Nieuwe taalelementen (vervolg)

Element	Omschrijving
BEGINTRANS( )	Vervanging voor BEGIN TRANSACTION en END TRANSACTION.
BINTYPE( )	Het resultaat is het vooraf gedefinieerde nummer van een opgegeven binair veld.
BITAND( )	Een AND-vergelijking maken van de bits in twee opgegeven uitdrukkingen.
BITLSHIFT( )	Het resultaat is een getal dat wordt gegenereerd door de bits in de opgegeven uitdrukking naar links te verschuiven.
BITRSHIFT( )	Het resultaat is een getal dat wordt gegenereerd door de bits in de opgegeven uitdrukking naar rechts te verschuiven.
BITOR( )	Een OR-vergelijking maken van de bits in twee opgegeven uitdrukkingen.
BITSET( )	Controleren of een bit in de opgegeven uitdrukking de waarde 1 heeft.
BITXOR( )	Een exclusieve OR-vergelijking maken (XOR) van de bits in twee opgegeven uitdrukkingen.
BOOKMARK( )	Het resultaat is een bladwijzer (lijkt op een recordaanwijzer) voor het huidige record in een dBASE-, Paradox- of SQL-tabel.
CD	Het standaardstation of de standaarddirectory wijzigen
CENTER( )	Het resultaat is een tekenreeks, die een reeks bevat die op een regel met de opgegeven lengte is gecentreerd.
CHARSET( )	Het resultaat is de naam van de tekenset die in de huidige of opgegeven tabel wordt gebruikt.
CHOOSEPRINTER( )	Een printer kiezen of afdrukopties instellen; de desbetreffende systeemgeheugenvariabelen opnieuw instellen.
CLASS...ENDCLASS	Een objectklasse en optioneel de kenmerken en methoden definiëren die gelden voor de elementen in die klasse.
CLEAR AUTOMEM	Een set automatische geheugenvariabelen initaliseren die bij velden in de huidige tabel horen.
CLEAR PROGRAM	Alle gecompileerde programmabestanden uit het geheugen verwijderen die niet worden uitgevoerd en die niet zijn geopend met een van de commando's SET FORMAT, SET PROCEDURE of SET LIBRARY.
CLOSE FORMS	Formulieren van het scherm verwijderen zonder de definities van die formulieren uit het geheugen te verwijderen.
CLOSE TABLES	Alle tabellen in werkgebieden sluiten of alle tabellen in de huidige database sluiten als een database is geselecteerd.
COMMIT()	Een transactie beëindigen en de wijzigingen die tijdens de transactie zijn gemaakt naar de geopende bestanden schrijven.
COPY BINARY	De inhoud van het opgegeven binaire veld kopiëren naar een bestand.
COPY TABLE	De opgegeven tabel kopiëren naar een bestand.
CREATE CATALOG	Een nieuwe catalogus maken.
CREATE COMMAND	Het opgegeven programmabestand weergeven, zodat het kan worden bewerkt, of een leeg bewerkscherm weergeven.
CREATE FILE	Het opgegeven tekstbestand weergeven zodat het kan worden bewerkt, of een leeg bewerkscherm weergeven.
CREATE FORM	Formulierontwerp starten om een fomulierbestand (.WFM) te maken of te wijzigen.

Nieuwe taalelementen (vervolg)

Element	Omschrijving
CREATE MENU	Menu-ontwerp starten om een menubestand (.MNU) te maken of te wijzigen.
CREATE SESSION	Een nieuwe sessie maken waarin u dezelfde tabel nogmaals kunt openen net alsof u in een multi-user-omgeving werkt.
CREATE...STRUCTURE EXTENDED	Een tabel maken en openen die u als sjabloon voor een nieuwe tabel kunt gebruiken.
DATABASE( )	Het resultaat is de naam van de huidige database van waaruit toegang wordt verkregen tot tabellen.
DBERROR( )	Het resultaat is het nummer van de laatst opgetreden IDAPI-fout.
DBMESSAGE( )	Het resultaat is de foutmelding die hoort bij de laatste of opgegeven IDAPI-fout.
DEFINE	Een object maken op basis van een ingebouwde of eigen klasse.
DEFINE COLOR	Een eigen kleur maken en daar een naam aan toekennen.
DELETE TABLE	De opgegeven tabel verwijderen.
DISPLAY/LIST COVERAGE	De inhoud van een dekkingsbestand (.COV) weergeven.
DO...UNTIL	De opdrachten uitvoeren die tussen de commando's DO en UNTIL staan zolang de opgegeven voorwaarde onwaar is of totdat het commando EXIT wordt gegeven.
DOS	Tijdelijk overschakelen naar de DOS-prompt zodat DOS-commando's kunnen worden uitgevoerd .
ELAPSED( )	Het resultaat is het aantal seconden dat is verlopen tussen de twee opgegeven tijden.
EMPTY( )	Het resultaat is .T. als de opgegeven uitdrukking of het opgegeven veld blanco is, of .F. als de uitdrukking of het veld gegevens bevat.
EXTERN	Een prototype definiëren voor een niet-dBASE-functie uit een DLL-bestand. Een prototype laat dBASE argumenten converteren in gegevenstypen die door de externe functie kunnen worden gebruikt. Vervolgens wordt het resultaat van de externe functie geconverteerd in een gegevenstype dat in dBASE kan worden gebruikt.
FDECIMAL( )	Het resultaat is het aantal decimale plaatsen in het opgegeven veld of de opgegeven tabel.
FLENGTH( )	Het resultaat is de lengte van het veld op de opgegeven positie van de tabel.
FOR...NEXT	De opdrachten uitvoeren die tussen de opdrachten FOR en NEXT staan, zo vaak uit als is opgegeven bij de opdracht FOR.
FUNIQUE( )	Een unieke bestandsnaam maken.
GENERATE	Records met willekeurige gegevens toevoegen aan de huidige tabel.
GETCOLOR( )	Een eigen kleur definiëren of een kleur selecteren van een kleurenpalet.
GETDIRECTORY( )	Een dialoogvenster weergeven waarin u een directory kunt selecteren die bij de volgende commando's wordt gebruikt.
GETEXPR( )	Het dialoogvenster **Uitdrukking samenstellen** of **Uitdrukking bewerken** weergeven. Het resultaat is de uitdrukking die u opgeeft.
GETFILE( )	Een dialoogvenster weergeven waarin u een bestandsnaam kunt kiezen.
GETFONT( )	Een dialoogvenster weergeven waarin u een font kunt selecteren.

Nieuwe taalelementen (vervolg)

Element	Omschrijving
HTOI( )	Het resultaat is het decimale equivalent van het opgegeven hexadecimale getal.
INSPECT( )	Een dialoogvenster weergeven met daarin alle kenmerken van het opgegeven object, evenals de huidige waarde van die kenmerken.
ISTABLE( )	Controleren of een tabel in de opgegeven database voorkomt.
ITOH( )	Het resultaat is het hexadecimale equivalent van het opgegeven decimale getal in de vorm van een tekenreeks.
LDRIVER( )	Het resultaat is de naam van de taalaansturing die voor de huidige tabel of de opgegeven tabel wordt gebruikt.
LENNUM( )	Het resultaat is het aantal cijfers waarin het opgegeven getal wordt weergegeven, met inbegrip van de spaties aan het begin.
LISTCOUNT( )	Het resultaat is het aantal prompts in een keuzelijst.
LISTSELECTED( )	Het resultaat is de prompt van een onderdeel uit een keuzelijst of keuzelijst met invoervak dat door de gebruiker is geselecteerd.
LOAD DLL	Een DLL-bestand initialiseren. Een DLL-bestand is een vooraf gecompileerde bibliotheek met externe routines die in een andere dan de dBASE-taal zijn geschreven, bijvoorbeeld in C of Pascal.
LOCAL	Geheugenvariabelen definiëren die alleen zichtbaar zijn in de subroutine waarin deze worden gedefinieerd.
MD	Een nieuwe DOS-directory maken.
MKDIR	Een nieuwe DOS-directory maken.
MODIFY FORM	Formulierontwerp starten om formulierbestanden te maken of wijzigen.
MODIFY MENU	Menu-ontwerp starten om een menubestand (.MNU) te wijzigen.
NEXTKEY( )	Het resultaat is de decimale waarde van een toets of toetscombinatie die is opgeslagen in de typbuffer.
OEM( )	Het resultaat is een tekenreeks die gelijk is aan de waarde van de opgegeven uitdrukking met ANSI-tekens in de codetabel.
ON NETERROR	Het opgegeven commando uitvoeren als er een fout optreedt die te maken heeft met multi-user-toepassingen.
ON SELECTION FORM	Een subroutine of een codeblok uitvoeren wanneer een gebruiker een formulier voorlegt.
OPEN DATABASE	Een verbinding met een database-server maken.
OPEN FORM	Formulieren en de objecten daarin weergeven en activeren.
PLAY SOUND	Een geluid uit een .WAV-bestand of memoveld herstellen en afspelen.
PROPER( )	Een tekenreeks in een eigennaam converteren.
PUTFILE( )	Een dialoogvenster weergeven waarin u een nieuwe bestandsnaam kunt maken.
RANDOM( )	Het resultaat is een willekeurige decimale waarde tussen 0 en 1
READMODAL( )	Een formulier openen als modaal venster. Het resultaat is de naam van het object dat de invoerfocus heeft wanneer de gebruiker het formulier voorlegt.
REDEFINE	De instellingen wijzigen die oorspronkelijk zijn gedefinieerd met het commando DEFINE.
REFRESH	De gegevensbuffers van het huidige of het opgegeven werkgebied aanpassen aan de laatste wijzigingen in de gegevens.

Nieuwe taalelementen (vervolg)

Element	Omschrijving
RELATION( )	Het resultaat is de indexsleuel die met het commando SET RELATION is gedefinieerd voor het huidige of het opgegeven werkgebied.
RELEASE AUTOMEM	Alle opgeslagen automatische geheugenvariabelen uit het geheugen verwijderen.
RELEASE DLL	Alle DLL-bestanden deactiveren. DLL-bestanden zijn vooraf gecompileerde bibliotheken met externe routines die eerst zijn geïnitialiseerd met het commando LOAD DLL.
RELEASE OBJECT	Objectdefinities uit het geheugen verwijderen.
RENAME TABLE	De naam van de opgegeven tabel wijzigen.
REPLACE AUTOMEM	De inhoud van geheugenvariabelen plaatsen in de juiste velden van het huidige record in de huidige tabel.
REPLACE BINARY	De inhoud van een binair veld vervangen door de inhoud van een binair bestand.
REPLACE MEMO	De tekst in een memoveld vervangen door de inhoud van een array.
REPLACE MEMO...FROM	Een tekstbestand plaatsen in een memoveld.
REPLACE OLE	Een OLE-document plaatsen in een OLE-veld.
RESOURCE( )	Het resultaat is een Windows resource-nummer uit een DLL-bestand.
RESTORE IMAGE	Een afbeelding weergeven die is opgeslagen in een bestand of memoveld.
ROLLBACK( )	Vervanging voor het commando ROLLBACK.
SECONDS( )	Het resultaat is het aantal seconden dat sinds middernacht is verlopen.
SET COVERAGE	Bepalen of dBASE een dekkingsbestand (.COV) maakt of wijzigt .
SET CUAENTER	Bepalen of de Enter-toets werkt als in Windows of als in dBASE voor DOS.
SET DATABASE	De standaarddatabase definiëren van waaruit toegang wordt verkregen tot tabellen.
SET DATE TO	De systeemdatum definiëren.
SET DBTYPE	Het standaardtype tabel instellen op Paradox of dBASE.
SET DEVICE	De opties ? en *<bestandsnaamfilter>* zijn toegevoegd.
SET EDITOR	Bepalen welke teksteditor moet worden gebruikt om programma- en tekstbestanden te maken en te wijzigen.
SET ERROR	Bepalen welke tekenuitdrukking aan een foutmelding voorafgaat en welke tekenuitdrukking daarop volgt.
SET HELP	Bepalen welk helpbestand (.HLP) het helpsysteem van dBASE gebruikt.
SET KEY	Een programma of procedure toewijzen aan de opgegeven toets of toetscombinatie.
SET PCOL	De kolom definiëren waarin de printkop moet worden geplaatst. Dit is de waarde van de functie PCOL( ).
SET PROW	De rij definiëren waarin de printkop moet worden geplaatst. Dit is de waarde van de functie PROW( ).
SET TIME	De systeemtijd definiëren.

Nieuwe taalelementen (vervolg)

Element	Omschrijving
SET TOPIC	Bepalen welk onderwerp in eerste instantie in het helpvenster wordt weergegeven als u het commando HELP gebruikt of op F1 drukt.
SET WP	Bepalen welke teksteditor wordt gebruikt om memovelden te bekijken of te bewerken.
SETTO( )	Het resultaat is de huidige instellen van het commando SET...TO of een functietoets.
SHELL( )	De interactieve omgeving van dBASE verwijderen of herstellen.
SHOW OBJECT	De weergave bijwerken van een object op het scherm.
SLEEP	Een programma onderbreken voor de duur van het opgegeven interval of tot aan de opgegeven tijd.
SQLERROR( )	Het resultaat is het nummer van de laatste server-fout.
SQLMESSAGE( )	Het resultaat is de meest recente melding betreffende een server-fout.
SQLEXEC()	Een SQL-opdracht uitvoeren in de huidige database.
STATIC	Geheugenvariabelen definiëren die alleen actief zijn in de subroutine waarin deze zijn gedefinieerd maar die tot aan het einde van de dBASE-sessie in het geheugen blijven staan.
STORE AUTOMEM	De inhoud van alle velden in het huidige record opslaan in een aantal geheugenvariabelen.
STORE MEMO	De tekst in een geheugenveld opslaan in een geheugenvariabele van het type array.
TARGET( )	Het resultaat is de naam van een tabel die aan het huidige of opgegeven werkgebied is gekoppeld.
UPDATED( )	Het resultaat geeft aan of u de inhoud van een @...GET-veld of van geheugenvariabelen hebt gewijzigd in het commandovenster.
VALIDDRIVE( )	Het resultaat geeft aan of een opgegeven station bestaat en gegevens kan lezen.

dBASE kent ook standaardklassen waarmee u veel gebruikte besturingselementen in vensters kunt maken zoals opdrachtknoppen, keuzerondjes en invoervakken (zie Hoofdstuk 7, "Klassen" voor een beschrijving van elke klasse).

Uitgebreide taalelementen

Element	Wijziging
ACTIVATE MENU	De optie RETRACTED is toegevoegd. Hierdoor wordt een popup-menu dat bij een strip hoort alleen weergegeven als de gebruiker op Enter drukt of op een geselecteerde strip klikt. Als de optie RETRACTED niet is opgegeven, wordt het popup-menu weergegeven zodra de bijbehorende strip is geselecteerd.
ACTIVATE SCREEN	De optie SAVE is toegevoegd. Dit voorkomt dat vensters in dBASE IV-stijl kunnen worden verschoven buiten het resultaatpaneel van het commandovenster.
APPEND	De opties NOWAIT en COLUMNAR zijn toegevoegd.

Uitgebreide taalelementen (vervolg)

Element	Wijziging
APPEND FROM	De opties <*bestandsnaamfilter*> en POSITION zijn toegevoegd. De typen PARADOX en DBASE zijn toegevoegd. De opties dBASE II, RPD, FW2, SYLK, DIF en WKS zijn vervallen.
APPEND MEMO	De opties ? en <*bestandsnaamfilter*> zijn toegevoegd.
AT( )	Zoeken in memovelden niet langer beperkt tot de eerste 64K van de gegevens.
BROWSE	De optie <*bladerobjectnaam*> is toegevoegd. De opties FOR en WHILE zijn toegevoegd. De optie COLOR is toegevoegd. De optie KEY...[EXCLUDE] is toegevoegd. De opties NOFOLLOW, NOINIT, NOMODIFY, NORMAL, NOTOGGLE en NOWAIT zijn toegevoegd. De optie TITLE is toegevoegd.
CHANGE	Identiek aan EDIT.
CLEAR	De optie CHARACTER <*Tuitdr*> toegevoegd. Hiermee in wordt het resultaatpaneel van het commandovenster of van het huidige dBASE IV-achtige venster het eerste teken van de uitdrukking <*Tuitdr*> ingevuld.
CLOSE DATABASES	Sluit niet alleen geopende tabellen maar ook geopende databases.
COPY	De typen PARADOX en DBASE zijn toegevoegd. De opties dBASE II, RPD, FW2, SYLK, DIF en WKS zijn vervallen.
COPY FILE	De opties ? en <*bestandsnaamfilter*> zijn toegevoegd.
COPY INDEXES	De optie ? is toegevoegd.
COPY MEMO	De optie ? is toegevoegd.
COPY STRUCTURE	De optie [TYPE] PARADOX | DBASE is toegevoegd.
COPY TO...STRUCTURE EXTENDED	De optie [TYPE] PARADOX | DBASE is toegevoegd.
CREATE	De optie ? is toegevoegd. De optie [TYPE] PARADOX | DBASE is toegevoegd.
CREATE...FROM	De opties ? en <*bestandsnaamfilter*> toegevoegd. De optie [TYPE] PARADOX | DBASE is toegevoegd.
CREATE LABLE	De optie <*bestandsnaamfilter*> is toegevoegd.
CREATE QUERY	Query-ontwerp starten. Mutatie -query's (.UPD) worden niet meer ondersteund.
CREATE REPORT	De optie <*bestandsnaamfilter*> is toegevoegd.
CREATE VIEW	Query-ontwerp starten. Mutatie-query's (.UPD) worden niet meer ondersteund.
DEFINE BAR	De optie SKIP [FOR <*voorwaarde Luitdr*>] is toegevoegd. Hierbij is een opdracht in een popup-menu niet beschikbaar als aan de voorwaarde is voldaan.
DEFINE PAD	De optie SKIP [FOR <*voorwaarde Luitdr*>] is toegevoegd. Hierbij is de menustrip niet beschikbaar als aan de voorwaarde is voldaan.
DEFINE POPUP	De optie PROMPT ARRAY is toegevoegd. Met PROMPT ARRAY <*naam array*> wordt de waarde van elk element uit de opgegeven array in het popup-menu getoond.
DELETE FILE	De optie <*bestandsnaamfilter*> is toegevoegd.

Uitgebreide taalelementen (vervolg)

Element	Wijziging
DIFFERENCE( )	De optie *<memoveld>* is toegevoegd.
DISKSPACE( )	De optie *<station Nuitdr>* is toegevoegd. Hiermee geeft u een stationnummer op van 1 tot en met 26. Nummer 1 en 2 komen bijvoorbeeld overeen met respectievelijk station A en B.
DISPLAY FILES	De opties ON *<station>* en TO FILE ? zijn toegevoegd.
DISPLAY MEMORY	De opties ? en *<bestandsnaamfilter>* zijn toegevoegd.
DISPLAY STRUCTURE	De opties ? en IN *<alias>* zijn toegevoegd.
DISPLAY STATUS	De opties ? en *<bestandsnaamfilter>* zijn toegevoegd.
DO	Het aanroepen van een door de gebruiker gedefinieerde functie met het commando DO wordt ondersteund.
EDIT	De opties FOR en WHILE zijn toegevoegd.
	De optie *<bladwijzer>* is toegevoegd.
	De optie COLOR is toegevoegd.
	De opties COLUMNAR, NOFOLLOW, NOINIT, NOMODIFY, NORMAL, NOTOGGLE en NOWAIT zijn toegevoegd.
	De optie TITLE is toegevoegd.
ERASE	De optie *<bestandsnaamfilter>* is toegevoegd.
GO/GOTO	De optie *<bladwijzer>* is toegevoegd.
IF	De optie ELSEIF is toegevoegd (ELSE IF is ook toegestaan).
IMPORT	De optie WB1 is toegevoegd.
	De opties dBASE II, RPD, FW2, FW3, PFS en WKS zijn vervallen.
INDEX	De opties ? en *<bestandsnaamfilter>* zijn toegevoegd.
	Sleutel PRIMARY is toegevoegd om het maken van primaire indexen bij Paradox-tabellen mogelijk te maken.
INKEY( )	De optie *<muis Tuitdr>* is toegevoegd.
ISALPHA( )	De optie *<memoveld>* is toegevoegd.
ISLOWER()	De optie *<memoveld>* is toegevoegd.
ISUPPER( )	De optie *<memoveld>* is toegevoegd.
JOIN	De optie ? is toegevoegd.
	De optie [TYPE] PARADOX | DBASE is toegevoegd.
LABEL FORM	De optie *<bestandsnaamfilter>* is toegevoegd.
LEFT( )	De lengte van het resultaat is niet langer beperkt tot 254 tekens.
	De optie *<memoveld>* is toegevoegd.
LEN( )	Null-waarden worden meegeteld.
LIKE( )	De optie *<memoveld>* is toegevoegd.
LIST STRUCTURE	De opties ? en IN *<alias>* zijn toegevoegd.
LOCK( )	De optie *<bladwijzer>* is toegevoegd.
LOWER( )	De optie *<memoveld>* is toegevoegd.
LTRIM( )	De lengte van het resultaat is niet langer beperkt tot 254 tekens.
	De optie *<memoveld>* is toegevoegd.
MAX( )	Logische uitdrukkingen worden ondersteund.
MEMLINES()	De optie *<regellengte Nuitdr>* is toegevoegd.
MIN( )	Logische uitdrukkingen worden ondersteund.
MLINE()	De optie *<regellengte Nuitdr>* is toegevoegd.

Uitgebreide taalelementen (vervolg)

Element	Wijziging
MODIFY COMMAND	De opties ? en <*bestandsnaamfilter*> zijn toegevoegd.
MODIFY QUERY	Query-ontwerp starten. Mutatie-query's (.UPD) worden niet meer ondersteund.
MODIFY VIEW	Start Query-ontwerp starten. Mutatie-query's (.UPD) worden niet meer niet ondersteund.
OS( )	Optioneel argument <*Nuitdr*> heeft als resultaat de versie van Windows die wordt gebruikt.
PRIVATE	LIKE en EXCEPT kunnen in dezelfde opdracht worden gebruikt.
QUIT	Mogelijke waarde <*Nuitdr*> nu 1 tot 254 in plaats van 0 tot 255.
RELEASE	LIKE en EXCEPT kunnen in dezelfde opdracht worden gebruikt.
REPLICATE( )	De lengte van het resultaat is niet langer beperkt tot 254 tekens. De optie <*memoveld*> is toegevoegd. Resultaat niet langer beperkt tot 254 tekens
REPORT FORM	De optie <*bestandsnaamfilter*> is toegevoegd.
RESTORE	De opties ? en <*bestandsnaamfilter*> zijn toegevoegd.
RESTORE WINDOW	De opties ? en <*bestandsnaamfilter*> zijn toegevoegd.
RIGHT( )	De lengte van het resultaat is niet langer beperkt tot 254 tekens. De optie <*memoveld*> is toegevoegd. Resultaat niet langer beperkt tot 254 tekens
RLOCK( )	De optie <*bladwijzer*> is toegevoegd.
RTRIM( )	De lengte van het resultaat is niet langer beperkt tot 254 tekens. De optie <*memoveld*> is toegevoegd.*memoveld> toegevoegd* Resultaat niet langer beperkt tot 254 tekens
SAVE	De opties ? en <*bestandsnaamfilter*> zijn toegevoegd. LIKE en EXCEPT kunnen beide bij dezelfde opdracht SAVE worden gebruikt
SAVE SCREEN	De optie TO FILE toegevoegd, optie TO <*gehvar*> is niet meer meer vereist.
SCAN	Geneste SCAN-lussen.
SEEK	<*uitdrukkingenlijst*> om op samengestelde veldsleutels te zoeken.
SET	Schrijft wijzigingen automatisch naar het bestand DBASEWIN.INI.
SET ALTERNATE	De opties ? en <*bestandsnaamfilter*> zijn toegevoegd.
SET CURRENCY	De optie <*Tuitdr*> is toegevoegd. Deze optie zet weergegeven tekens om in valutasymbool.
SET DEVICE	De opties ? en <*bestandsnaamfilter*> zijn toegevoegd.
SET FILTER	De optie <*bestandsnaamfilter*> is toegevoegd.
SET INDEX	De optie <*bestandsnaamfilter*> is toegevoegd. Sleutelwoord TAG is toegevoegd.
SET KEY TO	Sleutelwoorden LOW, HIGH en EXCLUDE zijn toegevoegd.
SET LIBRARY	De opties ? en <*bestandsnaamfilter*> zijn toegevoegd.
SET MESSAGE	De optie AT wordt niet ondersteund.
SET PRECISION	In dBASE zijn nu zowel numerieke als zwevende gegevens tot op 19 cijfers nauwkeurig.

Uitgebreide taalelementen (vervolg)

Element	Wijziging
SET PROCEDURE	De opties ?, *<bestandsnaamfilter>* en ADDITIVE worden ondersteund. Met de optie ADDITIVE worden een of meer procedurebestanden geopend zonder bestanden te sluiten die eerst met de opdracht SET PROCEDURE zijn geopend.
SET RELATION	De optie *<uitdrukkingenlijst>* is toegevoegd. om relatie te kunnen instellen bij samengestelde veldsleutels.
	De opties CONSTRAIN en INTEGRITY[CASCADE] zijn toegevoegd om de integriteit van gegevens in gekoppelde tabellen te garanderen.
SET VIEW	Activeert een query die met Query-ontwerp is gemaakt.
SPACE( )	De lengte van het resultaat is niet langer beperkt tot 254 tekens.
STUFF( )	De optie *<memoveld>* is toegevoegd.
SORT	De optie ? is toegevoegd.
	De optie [TYPE] PARADOX | DBASE is toegevoegd.
SOUNDEX( )	De optie *<memoveld>* is toegevoegd.
STR( )	De optie *<uit C>* is toegevoegd. Hiermee wordt het teken gedefinieerd waarmee de resulterende tekenreeks aan het begin moet worden opgevuld
STUFF( )	De optie *<memoveld>* is toegevoegd.
SUBSTR( )	De optie *<memoveld>* is toegevoegd.
	Resultaat is niet langer beperkt tot 254 tekens, als de argumenten *<begin Nuitdr>* of *<lengte Nuitdr>* gelijk zijn aan nul of als het argument *<lengte Nuitdr>* een negatief getal is, is de werking van de functie anders.
TRIM( )	De lengte van het resultaat is niet langer beperkt tot 254 tekens.
	De optie *<memoveld>* is toegevoegd.
	Resultaat is niet langer beperkt tot 254 tekens
TYPE	De opties MORE, ? en *<bestandsnaamfilter>* zijn toegevoegd.
UPPER()	De optie *<memoveld>* is toegevoegd.
USE	De opties *<bestandsnaamfilter>* en SHARED zijn toegevoegd.
	De optie [TYPE] PARADOX | DBASE is toegevoegd.
	De optie NOLOG is vervallen.

Niet-ondersteunde taalelementen

Element	Resultaat in dBASE
_pecode	Null-waarde.
_pscode	Null-waarde.
_pwait	.F.
ACCESS( )	0
ASSIST	Waarschuwing.
BEGIN...END TRANSACTION	Waarschuwing.
CALL	Waarschuwing.

Niet-ondersteunde taalelementen (vervolg)

Element	Resultaat in dBASE
CALL( )	Waarschuwing.
COMPLETED( )	.T.
DEXPORT	Waarschuwing.
DGEN( )	0
DISPLAY HISTORY	Commando wordt genegeerd.
DISPLAY USERS	Commando wordt genegeerd.
EXPORT	Waarschuwing.
FIXED( )	De waarde die aan de functie is doorgegeven.
ISMARKED( )	.F.
LIST HISTORY	Commando wordt genegeerd.
LIST USERS	Commando wordt genegeerd.
LOAD	Waarschuwing.
LOGOUT	Commando wordt genegeerd.
PLAY MACRO	Waarschuwing.
PROTECT	Commando wordt genegeerd.
RELEASE MODULE	Waarschuwing.
RESET	Waarschuwing.
RESTORE MACROS	Waarschuwing.
ROLLBACK	Waarschuwing.
SAVE MACROS	Waarschuwing.
SET CLOCK	Waarschuwing.
SET COLOR ON | OFF	Waarschuwing.
SET DBTRAP	Waarschuwing.
SET DEBUG	Waarschuwing.
SET ENCRYPTION	Waarschuwing.
SET HELP ON | OFF	Waarschuwing.
SET HISTORY	Waarschuwing.
SET HOURS	Waarschuwing.
SET INSTRUCT	Waarschuwing.
SET PAUSE	Waarschuwing.
SET SCOREBOARD	Waarschuwing.
SET SQL	Waarschuwing.
SET STATUS	Waarschuwing.
SET TRAP	Waarschuwing.
USER( )	Null-waarde.

Appendix

B

Technische specificaties van dBASE voor Windows

Deze appendix bevat de technische specificaties van bestanden en handelingen in dBASE voor Windows.

.DBF-tabellen

Tabel B.1

Omschrijving	Limiet in dBASE IV	Limiet in dBASE voor Windows
Maximaal aantal records	1 miljard	1 miljard
Maximale grootte van .DBF-tabelbestand (in bytes)	2 miljard	2 miljard
Maximale recordgrootte (in bytes)	4000 (exclusief _DBASELOCK-veld)	32.767 (inclusief _DBASELOCK-veld)
Maximaal aantal velden in .DBF-tabelbestand	255	1024

Indexen

Tabel B.2

Omschrijving	Limiet in dBASE IV	Limiet in dBASE voor Windows
Maximale lengte uitdrukking indexsleutel (in bytes)	220	220
Maximale lengte uitdrukking index FOR (in bytes)	220	220
Maximale lengte geëvalueerde indexsleutel (in bytes)	100	100
Maximale blokgrootte in .MDX-indexbestand (in bytes)	16.384	32.256 (standaard 1024)
Maximaal aantal index-labels in .MDX-indexbestand	47	47
Vaste blokgrootte van .NDX-indexbestand (in bytes)	512	512

Velden

Tabel B.3

Omschrijving	Limiet in dBASE IV	Limiet in dBASE voor Windows
Maximale grootte tekenveld (in bytes)	254	254
Grootte van datumvelden (in bytes)	8	8
Grootte van logische velden (in bytes)	1	1
Maximale grootte van numerieke velden (in cijfers)	20	20
Maximale grootte van memovelden (in bytes); inclusief OLE-velden en binaire velden	Memovelden alleen beperkt door beschikbaar geheugen	Alleen beperkt door beschikbaar geheugen
Maximaal aantal tekens in veldnamen	10	10
Precisie van zwevende getallen (in cijfers)	15,9	19
Grootste zwevende getal	0,9xE+308	0,9xE+308
Kleinste zwevende getal	0,1xE-307	0,1xE-307
Precisie van numerieke waarden (in cijfers)	20	19
Grootste numerieke waarde	0,9xE+308	0,9xE+308
Kleinste numerieke waarde	0,1xE-307	0,1xE-307

Multi-user

Tabel B.4

Omschrijving	Limiet in dBASE IV	Limiet in dBASE voor Windows
Maximaal aantal vergrendelingen (bestanden en records)	200 totaal	100 per werkgebied
Maximaal aantal herhalingen	32.000	32.000
Maximaal aantal seconden voor bijwerken	3600	3600

Procedures

Tabel B.5

Omschrijving	Limiet in dBASE IV	Limiet in dBASE voor Windows
Maximale lengte van commandoregel in programma's (in bytes)	1024	4096
Maximaal aantal actieve .PRO-bestanden	32	147
Maximale grootte van een procedure (in bytes)	65.520	Alleen beperkt door beschikbaar geheugen
Maximaal aantal procedures per bestand	963	193[1]
Maximaal aantal geopende procedurebestanden	1	Alleen beperkt door beschikbaar geheugen

1. Omdat u meerdere procedurebestanden kunt openen met behulp van de optie ADDITIVE, wordt het aantal beschikbare procedures alleen beperkt door het beschikbare geheugen.

Bestanden

Tabel B.6

Omschrijving	Limiet in dBASE IV	Limiet in dBASE voor Windows
Maximaal aantal geopende bestanden (alle typen)	99	250
Maximaal aantal geopende tabellen	40	225
Maximaal aantal geopende .DBT-memobestanden per actieve .DBF-tabel	1	1
Maximaal aantal geopende .MDX-indexbestanden per actieve .DBF-tabel	1 .MDX-productie-index plus extra non-productie-indexen toegestaan	1 .MDX-productie-index plus extra non-productie-indexen toegestaan
Maximaal aantal geopende .NDX-indexbestanden per actieve .DBF-tabel	10	10
Maximaal aantal geopende .FMT-indelingsbestanden per actieve .DBF-tabel	1	1
Maximale breedte van een rapport	255 tekens	57,7 centimeter
Maximaal aantal tabellen per rapport	9	225
Maximaal aantal rapporten per tabel	Onbeperkt	Onbeperkt
Maximaal aantal pagina's in een rapport	32.767	Onbeperkt
Maximaal aantal geneste rapportgroepzone's	44	Onbeperkt
Maximaal aantal velden in een rapport	Alleen beperkt door beschikbaar geheugen	Onbeperkt

Diversen

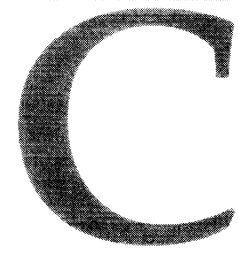
Tabel B.7

Omschrijving	Limiet in dBASE IV	Limiet in dBASE voor Windows
Maximaal aantal regels in de tekst-editor	32.000	Onbeperkt
Maximale regellengte in de tekst-editor	1024	32.767
Maximale breedte van een formulier	80 tekens	32767 pixels (het aantal tekens hangt af van de pixelbreedte van het font)
Maximaal aantal rijen in een formulier	32.767	32.767 pixels
Maximale lengte commandoregel in commandovenster	255	4096
Maximaal aantal regels opgeslagen in invoerpaneel van commandovenster	N/A	1000 regels
Maximaal aantal simultane sorteerniveau's	16	16
Maximaal aantal printer-aansturingen	4	Onbeperkt

Tabel B.7 (vervolg)

Omschrijving	Limiet in dBASE IV	Limiet in dBASE voor Windows
Maximaal aantal fonts per printeraansturing	5	9 (maximaal; niet per aansturing)
Maximaal aantal werkgebieden	40	225
Aantal programmeerbare functietoetsen	29	29
Maximale grootte geheugenvariabele	32.767	32.767
Maximaal aantal array-dimensies	2	255
Maximaal aantal elementen per dimensie	65.525	Onbeperkt
Maximale array-grootte	Alleen beperkt door beschikbaar geheugen	Alleen beperkt door beschikbaar geheugen

Appendix



Bestandsstructuren

Deze appendix biedt informatie over de structuur van dBASE-tabelbestanden (.DBF) en -memobestanden (.DBT).

Tabel-header en records

Een dBASE-tabelbestand (.DBF) bestaat uit een header, gegevensrecords, verwijdervlaggen en een einde-bestandsmarkering (EOF). De header bevat gegevens over de structuur van het bestand, de records de feitelijke gegevens. In elk record is een byte gereserveerd voor de verwijdervlag.

Structuur tabel-header

De structuur van de header biedt informatie waarmee het tabelbestand wordt onderhouden. Zie de tabellen C-1 en C-2 voor de details.

Tabel C.1 Header dBASE-tabelbestand

Byte	Inhoud	Omschrijving
0	1 byte	Geldig dBASE voor Windows-tabelbestand; bits 0–2 geven versienummer aan; bit 3 geeft aanwezigheid dBASE IV- of dBASE voor Windows-memobestand aan; bits 4–6 geven aanwezigheid dBASE IV SQL-tabel aan; bit 7 geeft de aanwezigheid van elk type .DBT-memobestand aan (dBASE III PLUS-type of een dBASE IV- of dBASE voor Windows-memobestand).
1–3	3 bytes	Datum laatst bijgewerkt, in notatie JJMMDD.
4–7	32-bits getal	Aantal records in tabel.
8–9	16-bits getal	Aantal bytes in header.
10–11	16-bits getal	Aantal bytes in record.
12–13	2 bytes	Gereserveerd. Opgevuld met nullen.
14	1 byte	Vlag, geeft onvoltooide dBASE IV-transactie aan[1].
15	1 byte	Vlag voor dBASE IV-codering[2].

Tabel C.1 Header dBASE-tabelbestand (vervolg)

Byte	Inhoud	Omschrijving
16–27	12 bytes	Gereserveerd voor multi-user-verwerking.
28	1 byte	.MDX-produktievlag. 01H wordt opgeslagen in deze byte als voor de tabel een .MDX-produktiebestand bestaat. 00H wordt opgeslagen als geen .MDX-bestand bestaat.
29	1 byte	Taalaansturings-ID.
30–31	2 bytes	Gereserveerd en opgevuld met nullen.
$32 - n^3$	32 bytes per veld	Veldbeschrijvings-array (zie tabel C-2 voor de structuur van de array).
$n + 1$	1 byte	0DH. Deze byte wordt hier opgeslagen als einde-veldteken (field terminator).

1. Deze vlag wordt niet gebruikt in dBASE voor Windows. In dBASE IV stelt BEGIN TRANSACTION deze vlag in op 01H en stellen END TRANSACTION en ROLLBACK de vlag opnieuw in op 00H.
2. Codering wordt niet ondersteund in dBASE voor Windows. Als deze vlag in dBASE IV is ingesteld op 01H, is de tabel gecodeerd.
3. n is de laatste byte in de veldbeschrijvings-array. De omvang van de array wordt bepaald door het aantal velden in het tabelbestand.

Tabel C.2 Veldbeschrijvings-bytes voor tabellen

Byte	Inhoud	Omschrijving
0–10	11 bytes	Veldnaam in ASCII (opgevuld met nullen).
11	1 byte	Veldtype in ASCII (B, C, D, F, G, L, M, of N).
12–15	4 bytes	Gereserveerd.
16	1 byte	Veldlengte (binair).
17	1 byte	Decimale veldtelling (binair).
18–19	2 bytes	Gereserveerd.
20	1 byte	Werkgebied-ID.
21–30	10 bytes	Gereserveerd.
31	1 byte	Vlag voor .MDX-produktieveld. 01H als het .MDX-produktiebestand een indexlabel voor het veld bevat. 00H als het veld niet is geïndexeerd.

Tabelrecords

In het tabelbestand volgen de records op de header. Gegevensrecords worden voorafgegaan door één byte. Deze byte bevat een spatie (20H) als het record niet is verwijderd en een asterisk (2AH) als het record is verwijderd. Velden worden in records verpakt zonder veldscheidingstekens of einde-recordtekens. Het einde van het bestand wordt gemarkeerd met één byte, de einde-bestandsmarkering (EOF). Hiervoor

wordt tekenwaarde 26 (1AH) uit de OEM-codetabel gebruikt. De tekens uit de OEM-codetabel die u kunt gebruiken voor gegevensinvoer, worden aangegeven in Tabel C-3.

Tabel C.3 Toegestane invoer voor dBASE-gegevenstypen

Gegevenstype	Gegevensinvoer
B (Binair)	Alle tekens uit de OEM-codetabel (intern opgeslagen als 10 cijfers die een .DBT-bloknummer aangeven).
C (Teken)	Alle tekens uit de OEM-codetabel.
D (Datum)	Cijfers en een teken waarmee jaar, maand en dag worden gescheiden (intern opgeslagen als 8 cijfers in de notatie JJJJMMDD).
G (Algemeen of OLE)	Alle tekens uit de OEM-codetabel (intern opgeslagen als 10 cijfers die een .DBT-bloknummer aangeven).
N (Numeriek) en F (Zwevend)	- ###,### 0 1 2 3 4 5 6 7 8 9
L (Logisch)	? J j N n T t F f (? als niet geïnitialiseerd)
M (Memo)	Alle tekens uit de OEM-codetabel (intern opgeslagen als 10 cijfers die een .DBT-bloknummer aangeven).

Binaire, memo- en OLE-velden en .DBT-bestanden

Voor binaire, memo- en OLE-velden worden de gegevens opgeslagen in .DBT-bestanden. Deze bestanden bestaan uit blokken die opeenvolgend worden genummerd (0, 1, 2 enzovoort). De omvang van elk blok wordt bepaald door SET BLOCKSIZE. Het eerste blok in het .DBT-bestand, blok 0, is de header van het .DBT-bestand.

Elk binair , memo- of OLE-veld van elk record in het .DBF-bestand bevat het nummer van het blok (in OEM-codetabelwaarden) waar de feitelijke gegevens in het record beginnen. Als een veld geen gegevens bevat, bevat het .DBF-bestand spaties (20H) in plaats van een nummer.

Als in een veld gegevens worden gewijzigd, kunnen ook de bloknummers wijzigen. In dat geval wordt mogelijk ook het nummer in het .DBF-bestand gewijzigd om de nieuwe plaats aan te geven.

Als u tekst verwijdert uit een memoveld (of een binair of OLE-veld), kan, anders dan in dBASE III PLUS, in dBASE voor Windows (net als in dBASE IV), de ruimte die werd ingenomen door de verwijderde tekst opnieuw worden gebruikt als u nieuwe tekst typt. In dBASE III PLUS wordt het .DBT-bestand altijd groter als nieuwe tekst wordt toegevoegd, ook als andere tekst wordt verwijderd.

Appendix

D

Waarden voor INKEY() en READKEY()

In deze bijlage vindt u een overzicht van de waarden die resulteren uit de functies INKEY( ) en READKEY( ).

Tabel D.1 Waarden voor INKEY( )

Ingedrukte toets	Waarde	Shift-toets waarde	Ctrl-toets waarde	Alt-toets[1] waarde
0	48	Afhankelijk van toetsenbord	-404	-452
1	49	Afhankelijk van toetsenbord	-404	-451
2	50	Afhankelijk van toetsenbord	-404	-450
3	51	Afhankelijk van toetsenbord	-404	-449
4	52	Afhankelijk van toetsenbord	-404	-448
5	53	Afhankelijk van toetsenbord	0	-447
6	54	Afhankelijk van toetsenbord	-30	-446
7	55	Afhankelijk van toetsenbord	-404	-445
8	56	Afhankelijk van toetsenbord	-404	-444
9	57	Afhankelijk van toetsenbord	-404	-443
a	97	65	1	-435
b	98	66	2	-434
c	99	67	3	-433

Tabel D.1 Waarden voor INKEY( ) (vervolg)

Ingedrukte toets	Waarde	Shift-toets waarde	Ctrl-toets waarde	Alt-toets[1] waarde
d	100	68	4	-432
e	101	69	5	-431
f	102	70	6	-430
g	103	71	7	-429
h	104	72	8	-428
i	105	73	9	-427
j	106	74	10	-426
k	107	75	11	-425
l	108	76	12	-424
m	109	77	13	-423
n	110	78	14	-422
o	111	79	15	-421
p	112	80	16	-420
q	113	81	17	-419
r	114	82	18	-418
s	115	83	19	-417
t	116	84	20	-416
u	117	85	21	-415
v	118	86	22	-414
w	119	87	23	-413
x	120	88	24	-412
y	121	89	25	-411
z	122	90	26	-410
F1 (Ctrl-\)	28	-20	-10	-30
F2	-1	-21	-11	-31
F3	-2	-22	-12	-32
F4	-3	-23	-13	-33
F5	-4	-24	-14	-34
F6	-5	-25	-15	-35
F7	-6	-26	-16	-36
F8	-7	-27	-17	-37
F9	-8	-28	-18	-38
F10	-9	-29	-19	-39
F11	-544	-546	-548	-550
F12	-545	-547	-549	-551
Pijl-links	19	-500	1	0
Pijl-rechts	4	-501	6	0
Pijl-omhoog	5	5	5	0
Pijl-omlaag	24	24	24	0
Home (Ctrl-])	26	26	29	0
End	2	2	23	0

Tabel D.1 Waarden voor INKEY( ) (vervolg)

Ingedrukte toets	Waarde	Shift-toets waarde	Ctrl-toets waarde	Alt-toets[1] waarde
Tab	9	-400	0	0
Enter	13	0	-402	0
Esc (Ctrl-[)	27	27	-	-
Ins	22	0	0	0
Del	7	-502	7	7
Backspace	127	127	-401	-403
PgUp	18	18	31	0
PgDn	3	3	30	0

1. De toetsenbordcode die wordt weergegeven voor combinaties van Alt met een alfanumerieke toets (behalve de kleine letters a-z), is de code minus 500. De codes voor toetscombinaties met Alt zijn hetzelfde voor hoofdletters en kleine letters.

Tabel D.2 Waarden voor READKEY( )

Niet-bijgewerkte code	Bijgewerkte code	Ingedrukte toets	Functie van toets
0	256	Ctrl-S, Pijl-links, Ctrl-H	Eén teken achteruit
—	256	Backspace	Eén teken achteruit
1	257	Ctrl-D, Pijl-rechts, Ctrl-L	Eén teken vooruit
2	258	Ctrl-A, Ctrl-Pijl-links	Naar begin voorafgaand woord
3	259	Ctrl-F, Ctrl-Pijl-rechts	Naar begin volgend woord
4	260	Ctrl-E, Ctrl-K, Pijl-omhoog	Eén veld achteruit
5	261	Ctrl-J, Ctrl-X, Pijl-omlaag	Eén veld vooruit
6	262	Ctrl-R, PgUp	Eén scherm achteruit
7	263	Ctrl-C, PgDn	Eén scherm vooruit
12	—	Ctrl-Q, Esc	Beëindigen zonder opslaan
—	270	Ctrl-W, Ctrl-End	Beëindigen met opslaan
15	271	Enter, Ctrl-M	RETURN of vul laatste record
16	—	Enter, Ctrl-M	Naar begin record in een venster **Toevoegen**
33	289	Ctrl-Home	Menu aan- en uitzetten

Tabel D.2 Waarden voor READKEY( ) (vervolg)

Niet-bijgewerkte code	Bijgewerkte code	Ingedrukte toets	Functie van toets
34	290	Ctrl-PgUp	Uitzoomen
35	291	Ctrl-PgDn	Inzoomen
36	292	F1	Help-functietoets

Appendix

E

ASCII-tekentabel (codepagina 437)

Dec	Hex	Teken	Dec	Hex	Teken	Dec	Hex	Teken	Dec	Hex	Teken
0	00	<null>	32	20	<space>	64	40	@	96	60	`
1	01	☺	33	21	!	65	41	A	97	61	a
2	02	☻	34	22	"	66	42	B	98	62	b
3	03	♥	35	23	#	67	43	C	99	63	c
4	04	♦	36	24	$	68	44	D	100	64	d
5	05	♣	37	25	%	69	45	E	101	65	e
6	06	♠	38	26	&	70	46	F	102	66	f
7	07	•	39	27	'	71	47	G	103	67	g
8	08	◘	40	28	(	72	48	H	104	68	h
9	09	○	41	29	)	73	49	I	105	69	i
10	0A	◙	42	2A	*	74	4A	J	106	6A	j
11	0B	♂	43	2B	+	75	4B	K	107	6B	k
12	0C	♀	44	2C	,	76	4C	L	108	6C	l
13	0D	♪	45	2D	-	77	4D	M	109	6D	m
14	0E	♫	46	2E	.	78	4E	N	110	6E	n
15	0F	☼	47	2F	/	79	4F	O	111	6F	o
16	10	►	48	30	0	80	50	P	112	70	p
17	11	◄	49	31	1	81	51	Q	113	71	q
18	12	↕	50	32	2	82	52	R	114	72	r
19	13	‼	51	33	3	83	53	S	115	73	s
20	14	¶	52	34	4	84	54	T	116	74	t
21	15	§	53	35	5	85	55	U	117	75	u
22	16	▬	54	36	6	86	56	V	118	76	v
23	17	↨	55	37	7	87	57	W	119	77	w
24	18	↑	56	38	8	88	58	X	120	78	x
25	19	↓	57	39	9	89	59	Y	121	79	y
26	1A	→	58	3A	:	90	5A	Z	122	7A	z
27	1B	←	59	3B	;	91	5B	[	123	7B	{
28	1C	∟	60	3C	<	92	5C	\	124	7C	⌂
29	1D	↔	61	3D	=	93	5D	]	125	7D	}
30	1E	▲	62	3E	>	94	5E	^	126	7E	~
31	1F	▼	63	3F	?	95	5F	_	127	7F	

Dec	Hex	Teken	Dec	Hex	Teken	Dec	Hex	Teken	Dec	Hex	Teken
128	80	Ç	160	A0	á	192	C0	└	224	E0	α
129	81	ü	161	A1	í	193	C1	┴	225	E1	ß
130	82	é	162	A2	ó	194	C2	┬	226	E2	Γ
131	83	â	163	A3	ú	195	C3	├	227	E3	π
132	84	ä	164	A4	ñ	196	C4	─	228	E4	Σ
133	85	à	165	A5	Ñ	197	C5	┼	229	E5	σ
134	86	å	166	A6	ª	198	C6	╞	230	E6	µ
135	87	ç	167	A7	º	199	C7	╟	231	E7	τ
136	88	ê	168	A8	¿	200	C8	╚	232	E8	Φ
137	89	ë	169	A9	⌐	201	C9	╔	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	╩	234	EA	Ω
139	8B	ï	171	AB	½	203	CB	╦	235	EB	δ
140	8C	î	172	AC	¼	204	CC	╠	236	EC	∞
141	8D	ì	173	AD	¡	205	CD	═	237	ED	ø
142	8E	Ä	174	AE	«	206	CE	╬	238	EE	ϵ
143	8F	Å	175	AF	»	207	CF	╧	239	EF	∩
144	90	É	176	B0	░	208	D0	╨	240	F0	≡
145	91	æ	177	B1	▒	209	D1	╤	241	F1	±
146	92	Æ	178	B2	▓	210	D2	╥	242	F2	≥
147	93	ô	179	B3	│	211	D3	╙	243	F3	≤
148	94	ö	180	B4	┤	212	D4	╘	244	F4	⌠
149	95	ò	181	B5	╡	213	D5	╒	245	F5	⌡
150	96	û	182	B6	╢	214	D6	╓	246	F6	÷
151	97	ù	183	B7	╖	215	D7	╫	247	F7	≈
152	98	ÿ	184	B8	╕	216	D8	╪	248	F8	°
153	99	Ö	185	B9	╣	217	D9	┘	249	F9	∙
154	9A	Ü	186	BA	║	218	DA	┌	250	FA	·
155	9B	¢	187	BB	╗	219	DB	█	251	FB	√
156	9C	£	188	BC	╝	220	DC	▄	252	FC	^{n}
157	9D	¥	189	BD	╜	221	DD	▌	253	FD	^{2}
158	9E	₧	190	BE	╛	222	DE	▐	254	FE	■
159	9F	ƒ	191	BF	┐	223	DF	▀	255	FF	

Index

A

B

C

D

E

F

G

H

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

Borland

Borland, dat haar hoofdkantoor heeft in Amerika, heeft verder kantoren in Australië, België, Canada, Denemarken, Duitsland, Engeland, Frankrijk, Hong Kong, Italië, Japan, Korea, Maleisië, Nederland, Nieuw-Zeeland, Singapore, Spanje, Taiwan en Zweden.
Nederland: Borland Benelux B.V., Postbus 71876, 1008 EB Amsterdam. België: Borland Belgium N.V., Boechoutlaan 55, Bus 1, 1853 Strombeek-Bever. • Part # DBS1150NL2177E • BOR 7120