Quick Reference Guide

Adaptive Server® Enterprise

15.0

DOCUMENT ID: DC70202-01-1500-02

LAST REVISED: November 2005

Sybase, Inc., One Sybase Drive, Dublin, CA 94568.

Quick Reference Guide

Datatypes

See *Reference Manual: Building Blocks* for more information about datatypes.

Data-types by category	Synonyms	Range	Bytes of storage
Exact numeric: integers			
bigint		Whole numbers between 2^{63} and -2^{63} -1 (from -9,223,372,036,854,775,808 to +9,223,372,036,854,775,807, inclusive.	8
int	integer	2^{31} -1 (2,147,483,647) to -2^{31} (-2,147,483,648)	4
smallint		2^{15} -1 (32,767) to -2^{15} (-32,768)	2
tinyint		0 to 255 (Negative numbers are not permitted)	1

Data-types by category	Synonyms	Range	Bytes of storage
unsigned bigint		Whole numbers between 0 and 18,446,744,073,709,551,615	8
unsigned int		Whole numbers between 0 and 4,294,967,295	4
unsigned smallint		Whole numbers between 0 and 65535	2
Exact numeric: decimals			
numeric (p, s)		10^{38} -1 to -10^{38}	2 to 17
decimal (p, s)	dec	10^{38} -1 to -10^{38}	2 to 17
Approximate numeric			
float (precision)		machine dependent	4 for default precision < 16, 8 for default precision >= 16
double precision		machine dependent	8
real		machine dependent	4
Money			
smallmoney		214,748.3647 to -214,748.3648	4
money		922,337,203,685,477.5807 to -922,337,203,685,477.5808	8
Date/time			
smalldatetime		January 1, 1900 to June 6, 2079	4
datetime		January 1, 1753 to December 31, 9999	8
date		January 1, 0001 to December 31, 9999	4
time		12:00:00AM to 11:59:59:999PM	4
Character			
char(n)	character	pagesize	n
varchar(n)	character varying, char varying	pagesize	actual entry length
unichar	Unicode character	pagesize	*n* * *@@unicharsize* (*@@unicharsize* equals 2)

Data-types by category	Synonyms	Range	Bytes of storage
univarchar	Unicode character varying, char varying	pagesize	actual number of characters * *@@unicharsize*
nchar(n)	national character, national char	pagesize	*n* * *@@ncharsize*
nvarchar(n)	nchar varying, national char varying, national character varying	pagesize	*@@ncharsize* * number of characters
text		2^{31} -1 (2,147,483,647) bytes or fewer	0 when uninitialized; multiple of 2K after initialization
unitext		1 – 1,073,741,823	0 when uninitialized; multiple of 2K after initialization
Binary			
binary(n)		pagesize	*n*
varbinary(n)		pagesize	actual entry length
image		2^{31} -1 (2,147,483,647) bytes or fewer	0 when uninitialized; multiple of 2K after initialization
Bit			
bit		0 or 1	1 (one byte holds up to 8 bit columns)

Standards and compliance

This table lists the ANSI SQL standards and compliance levels for Transact-SQL datatypes. See *Reference Manual: Building Blocks* for more information about standards and compliance.

Transact-SQL – ANSI SQL datatypes	Transact-SQL extensions – User-defined datatypes
• char • varchar • smallint • int • bigint • decimal • numeric • float • real • date • time • double precision	• binary • varbinary • bit • nchar • datetime • smalldatetime • tinyint • unsigned smallint • unsigned int • unsigned bigint • money • smallmoney • text • unitext • image • nvarchar • unichar • univarchar • sysname • longsysname • timestamp

Global variables

This section lists the Adaptive Server global variables and their brief descriptions. See *Reference Manual: Building Blocks* for more information.

Global variable	Definition
@@authmech	A read-only variable that indicates the mechanism used to authenticate the user.
@@bootcount	Returns the number of times an Adaptive Server installation has been booted.
@@boottime	Returns the date and time Adaptive Server was last booted.
@@bulkarraysize	*Component Integration Services* – returns the number of rows to be buffered in local server memory before being transferred using the bulk copy interface.

Global variable	Definition
@@bulkbatchsize	*Component Integration Services* – returns the number of rows transferred to a remote server via select into *proxy_table* using the bulk interface.
@@char_convert	Returns 0 if character set conversion is not in effect. Returns 1 if character set conversion is in effect.
@@cis_rpc_handling	*Component Integration Services* – returns 0 if cis rpc handling is off. Returns 1 if cis rpc handling is on.
@@cis_version	Returns the date and version of Component Integration Services.
@@client_csexpansion	Returns the expansion factor used when converting from the server character set to the client character set.
@@client_csid	Returns -1 if the client character set has never been initialized. Returns the client character set ID from syscharsets for the connection if the client character set has been initialized.
@@client_csname	Returns NULL if client character set has never been initialized; returns the name of the character set for the connection if the client character set has been initialized.
@@cmpstate	Returns the current mode of Adaptive Server in a high availability environment.
@@connections	Returns the number of user logins attempted.
@@cpu_busy	Returns the number of seconds, in CPU time, that Adaptive Server's CPU was performing Adaptive Server work.
@@cursor_rows	A global variable designed specifically for scrollable cursors. Displays the total number of rows in the cursor result set.
@@curloid	Either no cursors are open, no rows qualify for the last opened cursor, or the last open cursor is closed or deallocated.
@@datefirst	Set using set datefirst *n* where *n* is a value between 1 and 7.
@@dbts	Returns the timestamp of the current database.
@@error	Returns the error number most recently generated by the system.
@@errorlog	Returns the full path to the directory in which the Adaptive Server errorlog is kept, relative to *$SYBASE* directory (*%SYBASE%* on NT).
@@failedoverconn	Returns a value greater than 0 if the connection to the primary companion has failed over and is executing on the secondary companion server.

Global variable	Definition
@@fetch_status	Returns values: 0: fetch operation successful; –1: fetch operation unsuccessful; –2: value reserved for future use.
@@guestuserid	Returns the ID of the guest user.
@@hacmpservername	Returns the name of the companion server in a high availability setup.
@@haconnection	Returns a value greater than 0 if the connection has the failover property enabled.
@@heapmemsize	Returns the size of the heap memory pool, in bytes.
@@identity	Returns the most recently generated IDENTITY column value.
@@idle	Returns the number of seconds, in CPU time, that Adaptive Server has been idle.
@@invaliduserid	Returns a value of -1 for an invalid user ID.
@@io_busy	Returns the number of seconds in CPU time that Adaptive Server has spent doing input and output operations.
@@isolation	Returns the value of the session-specific isolation level (0, 1, or 3) of the current Transact-SQL program.
@@kernel_addr	Returns the starting address of the first shared memory region that contains the kernel region.
@@kernel_size	Returns the size of the kernel region that is part of the first shared memory region.
@@langid	Returns the server-wide language ID of the language in use, as specified in syslanguages.langid.
@@language	Returns the name of the language in use, as specified in syslanguages.name.
@@lock_timeout	Set using set lock wait n. Returns the current *lock_timeout* setting, in milliseconds.
@@maxcharlen	Returns the maximum length, in bytes, of a character in Adaptive Server's default character set.
@@max_connections	Returns the maximum number of simultaneous connections that can be made with Adaptive Server in the current computer environment.
@@maxgroupid	Returns the highest group user ID. The highest value is 1048576.
@@maxpagesize	Returns the server's logical page size.
@@max_precision	Returns the precision level used by decimal and numeric datatypes set by the server.
@@maxspid	Returns maximum valid value for the spid.
@@maxsuid	Returns the highest server user ID. The default value is 2147483647.

Global variable	Definition
@@maxuserid	Returns the highest user ID. The highest value is 2147483647.
@@mempool_addr	Returns the global memory pool table address.
@@min_poolsize	Returns the minimum size of a named cache pool, in kilobytes
@@mingroupid	Returns the lowest group user ID. The lowest value is 16384.
@@minspid	Returns 1, which is the lowest value for spid.
@@minsuid	Returns the minimum server user ID. The lowest value is -32768.
@@minuserid	Returns the lowest user ID. The lowest value is -32768.
@@monitors_active	Reduces the number of messages displayed by sp_sysmon.
@@ncharsize	Returns the maximum length, in bytes, of a character set in the current server default character set.
@@nestlevel	Returns the current nesting level.
@@nodeid	Returns the current installation's 48-bit node identifier.
@@optgoal	Returns the current optimization goal setting for query optimization
@@options	Returns a hexadecimal representation of the session's set options.
@@opttimeout	Returns the current optimization timeout limit setting for query optimization
@@pack_received	Retruns the number of input packets read by Adaptive Server.
@@pack_sent	Returns the number of output packets written by Adaptive Server.
@@packet_errors	Returns the number of errors detected by Adaptive Server while reading and writing packets.
@@pagesize	Returns the server's virtual page size.
@@parallel_degree	Returns the current maximum parallel degree setting.
@@probesuid	Returns a value of 2 for the probe user ID.
@@procid	Returns the stored procedure ID of the currently executing procedure.
@@recovery_state	Indicates whether Adaptive Server is in recovery.
@@repartition_degree	Returns the current dynamic repartitioning degree setting.
@@resource_granularity	Returns the maximum resource usage hint setting for query optimization
@@rowcount	Returns the number of rows affected by the last query.
@@scan_parallel_degree	Returns the current maximum parallel degree setting for nonclustered index scans.

Global variable	Definition
@@servername	Returns the name of Adaptive Server.
@@setrowcount	Returns the current value for set rowcount.
@@shmem_flags	Returns the shared memory region properties. This variable is for internal use.
@@spid	Returns the server process ID of the current process.
@@sqlstatus	Returns status information (warning exceptions) resulting from the execution of a fetch statement.
@@ssl_ciphersuite	Returns NULL if SSL is not used on the current connection; otherwise, it returns the name of the cipher suite you chose during the SSL handshake on the current connection.
@@stringsize	Returns the amount of character data returned from a toString() method.
@@tempdbid	Returns a valid temporary database ID (dbid) of the session's assigned temporary database.
@@textcolid	Returns the column ID of the column referenced by *@@textptr*.
@@textdataptnid	Returns the partition ID of a text partition containing the column referenced by *@@textptr*.
@@textdbid	Returns the database ID of a database containing an object with the column referenced by *@@textptr*.
@@textobjid	Returns the object ID of an object containing the column referenced by *@@textptr*.
@@textptnid	Returns the partition ID of a data partition containing the column referenced by *@@textptr*.
@@textptr	Returns the text pointer of the last text, unitext, or image column inserted or updated by a process (Not the same as the textptr function).
@@textptr_parameters	Returns 0 if the current status of the textptr_parameters configuration parameter is off. Returns 1 if the current status of the textptr_parameters if on.
@@textsize	Returns the limit on the number of bytes of text, unitext, or image data a select returns.
@@textts	Returns the text timestamp of the column referenced by *@@textptr*.
@@thresh_hysteresis	Returns the decrease in free space required to activate a threshold.
@@timeticks	Returns the number of microseconds per tick. The amount of time per tick is machine-dependent.
@@total_errors	Returns the number of errors detected by Adaptive Server while reading and writing.
@@total_read	Returns the number of disk reads by Adaptive Server.
@@total_write	Returns the number of disk writes by Adaptive Server.

Global variable	Definition
@@tranchained	Returns 0 if the current transaction mode of the Transact-SQL program is unchained. Returns 1 if the current transaction mode of the Transact-SQL program is chained.
@@trancount	Returns the nesting level of transactions in the current user session.
@@transactional_rpc	Returns 0 if RPCs to remote servers are transactional. Returns 1 if RPCs to remote servers are not transactional.
@@transtate	Returns the current state of a transaction after a statement executes in the current user session.
@@unicharsize	Returns 2, the size of a character in unichar.
@@version	Returns the date, version string, and so on of the current release of Adaptive Server.
@@version_number	Returns the whole version of the current release of Adaptive Server as an integer
@@version_as_integer	Returns the number of the last upgrade version of the current release of Adaptive Server as an integer.

Transact-SQL reserved words

This table lists words that are are reserved by Adaptive Server as keywords (part of SQL command syntax). See *Reference Manual: Building Blocks* for more information about reserved words.

	Words
A	add, all, alter, and, any, arith_overflow, as, asc, at, authorization, avg
B	begin, between, break, browse, bulk, by
C	cascade, case, char_convert, check, checkpoint, close, clustered, coalesce, commit, compute, confirm, connect, constraint, continue, controlrow, convert, count, count_big, create, current, cursor
D	database, dbcc, deallocate, declare, decrypt, default, delete, desc, deterministic, disk, distinct, drop, dummy, dump
E	else, encrypt, end, endtran, errlvl, errordata, errorexit, escape, except, exclusive, exec, execute, exists, exit, exp_row_size, external
F	fetch, fillfactor, for, foreign, from
G	goto, grant, group
H	having, holdlock
I	identity, identity_gap, identity_start, if, in, index, inout, insensitive, insert, install, intersect, into, is, isolation
J	jar, join
K	key, kill
L	level, like, lineno, load, lock

	Words
M	materialized, max, max_rows_per_page, min, mirror, mirrorexit, modify
N	national, new, noholdlock, nonclustered, nonscrollable, non_sensitive, not, null, nullif, numeric_truncation
O	of, off, offsets, on, once, online, only, open, option, or, order, out, output, over
P	partition, perm, permanent, plan, prepare, primary, print, privileges, proc, procedure, processexit, proxy_table, public
Q	quiesce
R	raiserror, read, readpast, readtext, reconfigure, references, remove, reorg, replace, replication, reservepagegap, return, returns, revoke, role, rollback, rowcount, rows, rule
S	save, schema, scroll, scrollable, select, semi_sensitive, set, setuser, shared, shutdown, some, statistics, stringsize, stripe, sum, syb_identity, syb_restree, syb_terminate
T	table, temp, temporary, textsize, to, tracefile, tran, transaction, trigger, truncate, tsequal
U	union, unique, unpartition, update, use, user, user_option, using
V	values, varying, view
W	waitfor, when, where, while, with, work, writetext
X	xmlextract, xmlparse, xmltest, xmlvalidate

ANSI SQL reserved words

This table lists words that are are ANSI SQL keywords that are not reserved by Adaptive Server. See *Reference Manual: Building Blocks* for more information about reserved words.

	Words
A	absolute, action, allocate, are, assertion
B	bit, bit_length, both
C	cascaded, case, cast, catalog, char, char_length, character, character_length, coalesce, collate, collation, column, connection, constraints, corresponding, cross, current_date, current_time, current_timestamp, current_user
D	date, day, dec, decimal, deferrable, deferred, describe, descriptor, diagnostics, disconnect, domain
E	end-exec, exception, extract
F	false, first, float, found, full
G	get, global, go
H	hour
I	immediate, indicator, initially, inner, input, insensitive, int, integer, interval
J	join
L	language, last, leading, left, local, lower
M	match, minute, module, month

	Words
N	names, natural, nchar, next, no, nullif, numeric
O	octet_length, outer, output, overlaps
P	pad, partial, position, preserve, prior
R	real, relative, restrict, right
S	scroll, second, section, semi_sensitive, session_user , size , smallint, space, sql, sqlcode, sqlerror, sqlstate, substring, system_user
T	then, time, timestamp, timezone_hour, timezone_minute, trailing, translate, translation, trim, true
U	unknown, upper, usage
V	value, varchar
W	when, whenever, write, year
Z	zone

Potential ANSI SQL reserved words

If you are using the ISO/IEC 9075:1989 standard, avoid using the words shown in this list because they may become ANSI SQL reserved words in the future. See *Reference Manual: Building Blocks* for more information about reserved words.

	Words
A	after, alias, async
B	before, boolean, breadth
C	call, completion, cycle
D	data, depth, dictionary
E	each, elseif, equals
G	general
I	ignore
L	leave, less, limit, loop
M	modify
N	new, none
O	object, oid, old, operation, operators, others
P	parameters, pendant, preorder, private, protected
R	recursive, ref, referencing, resignal, return, returns, routine, row
S	savepoint, search, sensitive, sequence, signal, similar, sqlexception, structure
T	test, there, type
U	under
V	variable, virtual, visible
W	wait, without

Functions

This section provides brief descriptions and syntax for built-in functions. See *Reference Manual: Building Blocks* for complete descriptions, examples, and usage information.

abs

Returns the absolute value of an expression.

abs(*numeric_expression*)

acos

Returns the angle (in radians) with a specified cosine.

acos(*cosine*)

ascii

Returns the ASCII code for the first character in an expression.

ascii(*char_expr* | *uchar_expr*)

asin

Returns the angle (in radians) with a specified sine.

asin(*sine*)

atan

Returns the angle (in radians) with a specified tangent.

atan(*tangent*)

atn2

Returns the angle (in radians) with specified sine and cosine.

atn2(*sine*, *cosine*)

avg

Returns the numeric average of all (distinct) values.

avg([all | distinct] *expression*)

audit_event_name

Returns a description of an audit event.

audit_event_name(*event_id*)

biginttohex

Returns the platform-independent 8-byte hexadecimal equivalent of the specified integer expression.

biginttohex (*integer_expression*)

case

Supports conditional SQL expressions; can be used anywhere a value expression can be used.

```
case
   when search_condition then expression
   [when search_condition then expression]...
```

[else *expression*]
end

case and values syntax:

case *expression*
when *expression* then *expression*
[when *expression* then *expression*]...
[else *expression*]
end

cast

Returns the specified value, converted to another datatype.

cast (*expression* as *datatype* [(*length* | *precision*[, *scale*])])

ceiling

Returns the smallest integer greater than or equal to the specified value.

ceiling(*value*)

char

Returns the character equivalent of an integer.

char(*integer_expr*)

char_length

Returns the number of characters in an expression.

char_length(*char_expr* | *uchar_expr*)

charindex

Returns an integer representing the starting position of an expression.

charindex(*expression1*, *expression2*)

coalesce

Supports conditional SQL expressions; can be used anywhere a value expression can be used; alternative for a case expression.

coalesce(*expression*, *expression* [, *expression*]...)

col_length

Returns the defined length of a column.

col_length(*object_name*, *column_name*)

col_name

Returns the name of the column where the table and column IDs are specified, and can be up to 255 bytes in length.

col_name(*object_id*, *column_id* [, *database_id*])

compare

Allows you to directly compare two character strings based on alternate collation rules.

compare ({*char_expression1*|*uchar_expression1*},
{*char_expression2*|*uchar_expression2*}),
[{*collation_name* | *collation_ID*}]

convert

Returns the specified value, converted to another datatype or a different datetime display format.

convert (*datatype* [(*length*) | (*precision*[, *scale*])]
[null | not null], *expression* [, *style*])

cos

Returns the cosine of the specified angle.

cos(*angle*)

cot

Returns the cotangent of the specified angle.

cot(*angle*)

count

Returns the number of (distinct) non-null values, or the number of selected rows as an integer.

count([all | distinct] *expression*)

count_big

Returns the number of (distinct) non-null values or the number of selected rows as a bigint.

count_big([all | distinct] *expression*)

current_date

Returns the current date.

current_date()

current_time

Returns the current time.

current_time()

curunreservedpgs

Returns the number of free pages in the specified disk piece.

curunreservedpgs (*dbid*, *lstart*, *unreservedpgs*)

data_pages

Returns the number of pages used by the specified table, index, or a specific partition.

data_pages(*dbid*, *object_id* [, *indid* [, *ptnid*]])

datachange

Measures the amount of change in the data distribution since update statistics last ran.

datachange(*object_name*, *partition_name*, *column_name*)

datalength

Returns the actual length, in bytes, of the specified column or string.

datalength(*expression*)

dateadd

Returns the date produced by adding or subtracting a given number of years, quarters, hours, or other date parts to the specified date.

dateadd(*date_part*, *integer*, *date expression*)

datediff

Returns the difference between two dates.

datediff(*datepart*, *date expression1*, *date expression2*)

datename

Returns the specified datepart (the first argument) of the specified date or time (the second argument) as a character string.

datename (*datepart*, *date expression*)

datepart

Returns the specified datepart in the first argument of the specified date (the second argument) as an integer.

datepart(*date_part*, *date expression*)

day

Returns an integer that represents the day in the datepart of a specified date.

day(*date_expression*)

db_id

Returns the ID number of the specified database.

db_id(*database_name*)

db_name

Returns the name of the database where the ID number is specified.

db_name([*database_id*])

degrees

Returns the size, in degrees, of an angle with the specified number of radians.

degrees(*numeric*)

derived_stat

Returns derived statistics for the specified object and index.

derived_stat(*object_name* | *object_id*,
index_name | *index_id*,
[*partition_name* | *partition_id*,]"*statistic*")

difference

Returns the difference between two soundex values.

difference(*expr1*,*expr2*)

exp

Returns the value that results from raising the constant to the specified power.

exp(*approx_numeric*)

floor

Returns the largest integer that is less than or equal to the specified value.

floor(*numeric*)

get_appcontext

Returns the value of the attribute in a specified context. get_appcontext is a built-in function provided by the Application Context Facility (ACF).

get_appcontext ("*context_name*", "*attribute_name*")

getdate

Returns the current system date and time.

getdate()

getutcdate

Returns a date and time where the value is in Universal Coordinated Time (UTC).

insert t1 (*c1, c2, c3*) select c1, getutcdate(), getdate() from t2)

has_role

Returns information about whether the user has been granted the specified role.

has_role ("*role_name*"[, 0])

hextobigint

Returns the bigint value equivalent of a hexadecimal string

hextobigint (*hexadecimal_string*)

hextoint

Returns the platform-independent integer equivalent of a hexadecimal string.

hextoint (*hexadecimal_string*)

host_id

Returns the client computer's operating system process ID for the current Adaptive Server client.

host_id()

host_name

Returns the current host computer name of the client process.

host_name()

identity_burn_max

Tracks the identity burn max value for a given table.

identity_burn_max(*table_name*)

index_col

Returns the name of the indexed column in the specified table or view, and can be up to 255 bytes in length

index_col (*object_name*, *index_id*, *key_#* [, *user_id*])

index_colorder

Returns the column order.

index_colorder (*object_name*, *index_id*, *key_#*
[, *user_id*])

inttohex

Returns the platform-independent hexadecimal equivalent of the specified integer.

inttohex (*integer_expression*)

is_quiesced

Indicates whether a database is in quiesce database mode.

is_quiesced(*dbid*)

is_sec_service_on

Returns 1 if the security service is active and 0 if it is not.

is_sec_service_on(*security_service_nm*)

isnull

Substitutes the value specified in *expression2* when *expression1* evaluates to NULL.

isnull(*expression1*, *expression2*)

lct_admin

Manages the last-chance threshold, returns the current value of the last-chance threshold (LCT), and aborts transactions in a transaction log that has reached its LCT.

lct_admin({{"lastchance" | "logfull" | |"reserved_for_rollbacks"},
database_id |"reserve", {*log_pages* | 0 }
| "abort", *process-id* [, *database-id*]})

left

Returns a specified number of characters on the left end of a character string.

left(*character_expression*, *integer_expression*)

len

Returns the number of characters, not the number of bytes, of a specified string expression, excluding trailing blanks.

len(*string_expression*)

license_enabled

Returns 1 if a feature's license is enabled, 0 if the license is not enabled, or NULL if you specify an invalid license name.

license_enabled("ase_server" | "ase_ha" |
"ase_dtm" | "ase_java" | "ase_asm")

list_appcontext

Lists all the attributes of all the contexts in the current session.

list_appcontext (["*context_name*"])

lockscheme

Returns the locking scheme of the specified object as a string.

lockscheme(*object_name*)

lockscheme(*object_id* [, *db_id*])

log

Returns the natural logarithm of the specified number.

log(*approx_numeric*)

log10

Returns the base 10 logarithm of the specified number.

log10(*approx_numeric*)

lower

Returns the lowercase equivalent of the specified expression.

lower(*char_expr* | *uchar_expr*)

ltrim

Returns the specified expression, trimmed of leading blanks.

ltrim(*char_expr* | *uchar_expr*)

max

Returns the highest value in an expression.

max(*expression*)

min

Returns the lowest value in a column.

min(*expression*)

month

Returns an integer that represents the month in the datepart of a specified date.

month(*date_expression*)

mut_excl_roles

Returns information about the mutual exclusivity between two roles.

mut_excl_roles (*role1*, *role2* [membership | activation])

newid

Generates human-readable, globally unique IDs (GUIDs) in two different formats, based on arguments you provide.

newid([*optionflag*])

next_identity

Retrieves the next identity value that is available for the next insert.

next_identity(*table_name*)

nullif

Supports conditional SQL expressions.

nullif(*expression*, *expression*)

object_id

Returns the object ID of the specified object.

object_id(*object_name*)

object_name

Returns the name of the object with the object ID you specify.

object_name(*object_id*[, *database_id*])

pagesize

Returns the page size, in bytes, for the specified object.

pagesize(*object_name* [, *index_name*])
pagesize(*object_id* [,*db_id* [, *index_id*]])

partition_id

Returns the partition ID of the specified data or index partition name.

partition_id(*table_name*, *partition_name* [,*index_name*])

partition_name

The explicit name of a new partition.

partition_name(*indid*, *ptnid* [, *dbid*])

patindex

Returns the starting position of the first occurrence of a specified pattern.

patindex("%*pattern*%", *char_expr*|*uchar_expr*
[, using {bytes | characters | chars}])

pi

Returns the constant value 3.1415926535897936.

pi()

power

Returns the value that results from raising the specified number to a given power.

power(*value*, *power*)

proc_role

Returns information about whether the user has been granted the specified role.

proc_role ("*role_name*")

radians

Returns the size, in radians, of an angle with the specified number of degrees.

radians(*numeric*)

rand

Returns a random value between 0 and 1, which is generated using the specified seed value.

rand([*integer*])

replicate

Returns a string consisting of the specified expression repeated a given number of times.

replicate (*char_expr* | *uchar_expr*, *integer_expr*)

reserved_pages

Reports the number of pages reserved to a table, index or a specific partition.

reserved_pages(*dbid*, *object_id* [, *indid* [, *ptnid*]])

reverse

Returns the specified string with characters listed in reverse order.

reverse(*expression* | *uchar_expr*)

right

The rightmost part of the expression with the specified number of characters.

right(*expression*, *integer_expr*)

rm_appcontext

Removes a specific application context, or all application contexts.

rm_appcontext (“*context_name*”, “*attribute_name*”)

role_contain

Returns 1 if *role2* contains *role1*.

role_contain("*role1*", "*role2*")

role_id

Returns the system role ID of the name you specify.

role_id("*role_name*")

role_name

Returns the name of a system role ID you specify.

role_name(*role_id*)

round

Returns the value of the specified number, rounded to a specified number of decimal places.

round(*number*, *decimal_places*)

row_count

Returns an estimate of the number of rows in the specified table.

row_count(*dbid*, *object_id* [,*ptnid*])

rtrim

Returns the specified expression, trimmed of trailing blanks.

rtrim(*char_expr* | *uchar_expr*)

set_appcontext

Sets an application context name, attribute name, and attribute value for a user session, defined by the attributes of a specified application. set_appcontext is a built-in function that the Application Context Facility (ACF) provides.

set_appcontext (“*context_name*, “*attribute_name*”, “*attribute_value*”)

show_role

Shows the login’s currently active system-defined roles.

show_role()

show_sec_services

Lists the security services that are active for the session.

show_sec_services()

sign

Returns the sign (1 for positive, 0, or -1 for negative) of the specified value.

sign(*numeric*)

sin

Returns the sine of the specified angle (in radians).

sin(*approx_numeric*)

sortkey

Generates values that can be used to order results based on collation behavior.

sortkey (*char_expression* | *uchar_expression*)
[, {*collation_name* | *collation_ID*}])

soundex

Returns a four-character code representing the way an expression sounds.

soundex(*char_expr* | *uchar_expr*)

space

Returns a string consisting of the specified number of single-byte spaces.

space(*integer_expr*)

square

Returns the square of a specified value expressed as a float.

square(*numeric_expression*)

sqrt

Returns the square root of the specified number.

sqrt(*approx_numeric*)

str

Returns the character equivalent of the specified number.

str(*approx_numeric* [, *length* [, *decimal*]])

str_replace

Replaces any instances of the second string expression (*string_expression2*) that occur within the first string expression (*string_expression1*) with a third expression (*string_expression3*).

replace("*string_expression1*", "*string_expression2*",
"*string_expression3*")

stuff

Returns the string formed by deleting a specified number of characters from one string and replacing them with another string.

stuff(*char_expr1* | *uchar_expr1*, *start*, *length*,
char_expr2 | *uchar_expr2*)

substring

Returns the string formed by extracting the specified number of characters from another string.

substring(*expression*, *start, length*)

sum

Returns the total of the values.

sum([all | distinct] *expression*)

suser_id

Returns the server user's ID number from the syslogins table.

suser_id([*server_user_name]*)

suser_name

Returns the name of the current server user or the user whose server ID is specified.

suser_name([*server_user_id*])

syb_quit

Terminates the connection.

syb_quit()

syb_sendmsg

UNIX only – sends a message to a User Datagram Protocol (UDP) port.

syb_sendmsg *ip_address*, *port_number*, *message*

tan

Returns the tangent of the specified angle (in radians).

tan(*angle*)

tempdb_id

Reports the temporary database to which a given session is assigned.

tempdb_id()

textptr

Returns a pointer to the first page of a text, image, or unitext column.

textptr(*column_name*)

textvalid

Returns 1 if the pointer to the specified text or unitext column is valid; 0 if it is not.

textvalid("*table_name.column_name*", *textpointer*)

to_unichar

Returns a unichar expression having the value of the integer expression.

to_unichar (*integer_expr*)

tran_dumptable_status

Returns a true/false indication of whether dump transaction is allowed.

tran_dumpable_status("*database_name*")

tsequal

Compares timestamp values to prevent update on a row that has been modified since it was selected for browsing.

tsequal(*browsed_row_timestamp*, *stored_row_timestamp*)

uhighsurr

Returns 1 if the Unicode value at position *start* is the high half of a surrogate pair (which should appear first in the pair). Returns 0 otherwise.

uhighsurr(*uchar_expr*, start)

ulowsurr

Returns 1 if the Unicode value at position *start* is the low half of a surrogate pair (which should appear second in the pair). Returns 0 otherwise.

ulowsurr(*uchar_expr*, start)

upper

Returns the uppercase equivalent of the specified string.

upper(*char_expr*)

uscalar

Returns the Unicode scalar value for the first Unicode character in an expression.

uscalar(*uchar_expr*)

used_pages

Reports the number of pages used by a table, an index, or a specific partition.

used_pages(*dbid*, *object_id* [, *indid* [, *ptnid*]])

user

Returns the name of the current user.

user

user_id

Returns the ID number of the specified user or of the current user in the database.

user_id([*user_name*])

user_name

Returns the name within the database of the specified user or of the current user.

user_name([*user_id*])

valid_name

Returns 0 if the specified string is not a valid identifier or a number other than 0 if the string is a valid identifier, and can be up to 255 bytes in length.

valid_name(*character_expression* [, *maximum_length*])

valid_user

Returns 1 if the specified ID is a valid user or alias in at least one database on this Adaptive Server.

valid_user(*server_user_id*)

year

Returns an integer that represents the year in the datepart of a specified date.

year(*date_expression*)

Commands

This section provides brief descriptions and syntax for Adaptive Server commands. See *Reference Manual: Commands* for complete descriptions, examples, and usage information.

alter database

Increases the amount of space allocated to a database.

```
alter database database_name
    [on {default | database_device } [= size]
        [, database_device [= size]]...]
    [log on { default | database_device } [ = size ]
        [ , database_device [= size]]...]
    [with override] [for load] [for proxy_update]
```

alter role

Defines mutually exclusive relationships between roles; adds, drops, and changes passwords for roles; specifies the password expiration interval, the minimum password length, and the maximum number of failed logins allowed for a specified role.

```
alter role role1 { add | drop } exclusive
    { membership | activation } role2
alter role role_name [add passwd "password" |
    drop passwd] [lock | unlock]
alter role { role_name | "all overrides" }
    set { passwd expiration | min passwd length |
    max failed_logins } option_value
```

alter table

Adds new columns to a table; drops or modifies existing columns; adds, changes, or drops constraints; changes properties of an existing table; enables or disables triggers on a table.

```
alter table [[database.][owner].table_name
    { add column_name datatype}
        [default {constant_expression | user | null}]
        {identity | null | not null} [off row | in row]
        [ [constraint constraint_name]
        { { unique | primary key } [clustered | nonclustered] [asc | desc]
            [with { fillfactor = pct, max_rows_per_page = num_rows,
                reservepagegap = num_pages }]
            [on segment_name]
        | references [[database.]owner.]ref_table
            [(ref_column)] [match full]
        | check (search_condition) ] ... }
        [, next_column]...
    | add {[constraint constraint_name]
    {  unique | primary key}
            [clustered | nonclustered] (column_name [asc | desc]
        [, column_name [asc | desc]...])
        [with { fillfactor = pct, max_rows_per_page = num_rows,
                reservepagegap = num_pages}]
        [on segment_name]
    | foreign key (column_name [{, column_name}...])
```

```
        references [[database.]owner.]ref_table
        [(ref_column [{, ref_column}...])] [match full]
    | check (search_condition)} | drop {column_name [, column_name]...
        | constraint constraint_name }
    | modify column_name datatype [null | not null] [, next_column]...
    | replace column_name default { constant_expression | user | null}
        | { enable | disable } trigger
    | lock {allpages | datarows | datapages } }
    | with exp_row_size=num_bytes | partition number_of_partitions
    | unpartition | partition_clause | add_partition_clause
```

For partitions:

```
partition_clause::=
    partition by range ( column_name[, column_name ]...)
        ( [ partition_name ] values <= ( { constant | MAX }
            [, { constant | MAX } ] ...) [ on segment_name ]
            [, [ partition_name ] values <= ( { constant | MAX }
                [, { constant | MAX } ] ...) [ on segment_name ] ]...)

    | partition by hash (column_name[, column_name ]...)
        { ( partition_name [ on segment_name ]
            [, partition_name [ on segment_name ] ]...)
        | number_of_partitions [ on (segment_name
            [, segment_name ] ...) ] }

    | partition by list (column_name)
        ( [ partition_name ] values ( constant[, constant ] ...)
            [ on segment_name ]
            [, [ partition_name ] values ( constant[, constant ] ...)
                [ on segment_name ] ] ...)

    | partition by roundrobin
        { ( partition_name [ on segment_name ]
            [, partition_name [ on segment_name ] ]...)
        | number_of_partitions [ on ( segment_name [, segment_name ]...) ]
}

add_partition_clause::=
    add partition
        { ( [ partition_name ] values <= ( { constant | MAX }
            [, { constant | MAX } ]...) [ on segment_name ]
            [, [ partition_name ] values <= ( { constant | MAX }
                [, { constant | MAX } ] ...) [ on segment_name ] ]... )

        | ( [ partition_name ] values ( constant[, constant ] ...)
            [ on segment_name ]
            [, [ partition_name ] values ( constant[, constant ] ...)
                [ on segment_name ] ] ...) }
```

For computed columns:

```
alter table
    add column_name {compute | as}
        computed_column_expression... [materialized | not materialized]
    drop column_name
    modify column_name {null | not null |
```

{materialized | not materialized} [null | not null] |
{compute | as} *computed_column_expression*
[materialized | not materialized] [null | not null]}

begin...end

Encloses a series of SQL statements so that control-of-flow language, such as if...else, can affect the performance of the whole group.

```
begin
    statement block
end
```

begin transaction

Marks the starting point of a user-defined transaction.

```
begin tran[saction] [transaction_name]
```

break

Causes an exit from a while loop. break is often activated by an if test.

```
while logical_expression
    statement
break
    statement
continue
```

checkpoint

Writes all dirty pages (pages that have been updated since they were last written) to the database device.

```
checkpoint [all | [dbname[, dbname, dbname, .....]]
```

close

Deactivates a cursor.

```
close cursor_name
```

commit

Marks the ending point of a user-defined transaction.

```
commit [tran | transaction | work] [transaction_name]
```

compute clause

Generates summary values that appear as additional rows in the query results.

```
start_of_select_statement
    compute row_aggregate (column_name)
        [, row_aggregate(column_name)]...
    [by column_name [, column_name]...]
```

connect to...disconnect

Component Integration Services only – connects to the specified server and disconnects the connected server.

```
connect to server_name
disconnect [from ASE [all] [connection_name]
```

Creates a passthru to a different server:

```
connect
    [to ASE engine_name]
    [database database_name]
    [as connection_name]
    [user user_id]
    [identified by password]]]
```

Opens a new JDBC-level connection to Adaptive Server, and does not use CIS:

```
connect using connect_string
```

continue

Restarts the while loop. continue is often activated by an if test.

```
while boolean_expression
        statement
    break
        statement
continue
```

create database

Creates a new database.

```
create [temporary] database database_name
    [on {default | database_device} [= size]
        [, database_device [= size]]...]
    [log on database_device [= size]
        [, database_device [= size]]...]
    [with {override | default_location = "pathname"}]
    [for {load | proxy_update}]
```

create default

Specifies a value to insert in a column (or in all columns of a user-defined datatype) if no value is explicitly supplied at insert time.

```
create default [owner.]default_name as constant_expression
```

create existing table

Component Integration Services only– Creates a proxy table, then retrieves and stores metadata from a remote table and places the data into the proxy table. Allows you to map the proxy table to a table, view, or procedure at a remote location.

```
create existing table table_name (column_list)
    [ on segment_name ]
    [ [ external {table | procedure | file} ] at pathname
    [column delimiter "string"]]
```

create function (SQLJ)

Creates a user-defined function by adding a SQL wrapper to a Java static method. Can return a value defined by the method.

```
create function [owner.]sql_function_name
        ( [ sql_parameter_name sql_datatype
            [( length)| (precision[, scale ]) ]
        [ [, sql_parameter_name sql_datatype
            [( length )| ( precision[, scale ]) ]]
        ... ] ] )
    returns sql_datatype [ ( length)| (precision[, scale ]) ]
```

```
[modifies sql data]
[returns null on null input |
   called on null input]
[deterministic | not deterministic]
[exportable]
language java
parameter style java
external name 'java_method_name
   [ ( [java_datatype[, java_datatype
   ...] ] ) ] '
```

create index

Creates an index on one or more columns in a table, computed or non-computed. Creates partitioned indexes.

```
create [unique] [clustered | nonclustered] index index_name
   on [[database.]owner.]table_name
      (column_expression [asc | desc]
         [, column_expression [asc | desc]]...)
   [with { fillfactor = pct, max_rows_per_page = num_rows,
         reservepagegap = num_pages,
         consumers = x, ignore_dup_key,
   sorted_data, [ignore_dup_row | allow_dup_row],
         statistics using num_steps values  } ]
   [on segment_name] [ index_partition_clause ]
index_partition_clause::=
   [ local index [partition_name [on segment_name ]
   [, partition_name [ on segment_name ] ... ] ] ]

create [unique | nonclustered] index index_name
   on [[ database.] owner.] table_name
   (column_expression [asc | desc] [, column_expression [asc | desc]]...
```

create plan

Creates an abstract plan.

```
create plan query plan [into group_name] [and set @new_id]
```

create procedure

Creates a stored procedure or an extended stored procedure (ESP) that can take one or more user-supplied parameters.

```
create procedure [owner.]procedure_name[;number]
   [[()@parameter_name datatype [(length ) | (precision  [, scale ])]
      [= default][output]
   [, @parameter_name datatype [(length ) | (precision  [, scale ])]
      [= default][output]]...[()]]
   [with recompile]
   as {SQL_statements | external name dll_name}
```

create procedure (SQLJ)

Creates a SQLJ stored procedure by adding a SQL wrapper to a Java static method.

```
create procedure [owner.]sql_procedure_name
   ([ [ in | out | inout ] sql_parameter_name
      sql_datatype [( length) | (precision[, scale]) ] [=default]...])
```

```
[, [ in | out | inout ] sql_parameter_name
    sql_datatype [( length) | (precision[, scale ]) ] ] [=default]...])
[modifies sql data ]
[dynamic result sets integer]
[deterministic | not deterministic]
language java
parameter style java
external name 'java_method_name
    [ ( [java_datatype[, java_datatype
    ...]] ) ]'
```

create proxy_table

Component Integration Services only– Creates a proxy table without specifying a column list. Component Integration Services derives the column list from the metadata it obtains from the remote table.

```
create proxy_table table_name
    [on segment_name] [ external [ table | directory | file ] ]
    at pathname [column delimiter “<string>”]
```

create role

Creates a user-defined role; specifies the password expiration interval, the minimum password length, and the maximum number of failed logins allowed for a specified role at creation. You can also associate a password with the role at the time that the role is created.

```
create role role_name [ with passwd "password"
    [, {passwd expiration | min passwd length |
    max failed_logins } option_value ] ]
```

create rule

Specifies the domain of acceptable values for a particular column or for any column of a user-defined datatype, and creates access rules.

```
create [ [ and | or ] access]] rule
    [owner.]rule_name as condition_expression
```

create schema

Creates a new collection of tables, views, and permissions for a database user.

```
create schema authorization authorization_name
    create_oject_statement [create_object_statement ... ]
    [permission_statement ... ]
```

create table

Creates new tables and optional integrity constraints, defines computed columns when a table is created, and defines the table’s partition property when the table is created.

```
create table [database .[owner ].]table_name (column_name datatype
    [default {constant_expression  | user | null}]
    {[{identity | null | not null}] [off row | [ in row [ (size_in_bytes) ] ]
    [[constraint constraint_name ]
        {{unique | primary key}
        [clustered | nonclustered] [asc | desc]
        [with { fillfactor = pct, max_rows_per_page = num_rows, }
              reservepagegap = num_pages }]
        [on segment_name] | references [[database .]owner .]ref_table
```

```
            [(ref_column )] [match full] | check (search_condition)}]}
            [match full]...
        | [constraint constraint_name]
            {{unique | primary key}
            [clustered | nonclustered]
            (column_name [asc | desc] [{, column_name [asc | desc]}...])
            [with { fillfactor = pct
                max_rows_per_page = num_rows ,
                reservepagegap = num_pages } ]
            [on segment_name]
            |foreign key (column_name [{,column_name}...])
            references [[database.]owner.]ref_table
                [(ref_column [{, ref_column}...])] [match full]
            | check (search_condition) ... }
        [{, {next_column | next_constraint}}...])
        [lock {datarows | datapages | allpages }]
        [with { max_rows_per_page = num_rows,
            exp_row_size = num_bytes, reservepagegap = num_pages,
            identity_gap = value } ]
        [on segment_name ]
        [ partition_clause ]
        [ [ external table ] at pathname ]
    partition_clause::=
        partition by range ( column_name[, column_name ]...)
            ( [ partition_name ] values <= ( { constant | MAX }
                [, { constant | MAX } ] ...) [ on segment_name ]
                [, [ partition_name ] values <= ( { constant | MAX }
                    [, { constant | MAX } ] ...) [ on segment_name ] ]...)

        | partition by hash (column_name[, column_name ]...)
            { ( partition_name [ on segment_name ]
                [, partition_name [ on segment_name ] ]...)
            | number_of_partitions
                [ on (segment_name[, segment_name ] ...) ] }

        | partition by list (column_name)
            ( [ partition_name ] values ( constant[, constant ] ...)
                [ on segment_name ]
                [, [ partition_name ] values ( constant[, constant ] ...)
                    [ on segment_name ] ] ...)

        | partition by roundrobin
            { ( partition_name [ on segment_name ]
                [, partition_name [ on segment_name ] ]...)
            | number_of_partitions
                [ on ( segment_name [, segment_name ]...) ] }
    create table [database.[owner].] table_name
        (column_name {compute | as}
            computed_column_expression [materialized | not materialized] }
```

create trigger

Creates a trigger, a type of stored procedure that is often used for enforcing integrity constraints.

```
create trigger [owner .]trigger_name
    on [owner .]table_name for {insert , update , delete}
    as SQL_statements
```

Or, using the if update clause:

```
create trigger [owner .]trigger_name
    on [owner .]table_name for {insert , update}
    as [if update (column_name )
            [{and | or} update (column_name )]...] SQL_statements
       [if update (column_name )
            [{and | or} update (column_name )]... SQL_statements ]...
```

create view

Creates a view.

```
create view [owner .]view_name
    [(column_name [, column_name ]...)]
    as select [distinct] select_statement [with check option]
```

dbcc

Checks the logical and physical consistency of a database and provides statistics, planning, and repair functionality.

```
dbcc addtempdb( dbid |database_name )
dbcc checkalloc [(database_name [, fix | nofix])]
dbcc checkcatalog [(database_name[, fix])
dbcc checkdb [(database_name [, skip_ncindex])]
dbcc checkindex({table_name | table_id}, index_id
    [, bottom_up [, partition_name | partition_id]])
dbcc checkstorage [(database_name)]
dbcc checktable(table_name | table_id
    [, skip_ncindex | fix_spacebits | "check spacebits" |
    bottom_up | NULL [, partition_name | partition_id)
dbcc checkverify(dbname[,tblname [,ignore_exclusions]])
dbcc complete_xact (xid, {["commit", “1pc”] | "rollback"})
dbcc forget_xact (xid)
dbcc dbrepair (database_name, dropdb)
dbcc engine( {offline , [enginenum] | "online" })
dbcc fix_text ({table_name | table_id})
dbcc indexalloc(table_name | table_id, index_id
    [,optimized | fast | NULL [, fix | nofix | NULL
    [, partition_name | partition_id]]])
dbcc monitor (increment, <group name>)
dbcc monitor (decrement, <group name>)
dbcc monitor (reset, <group name>)
dbcc pravailabletempdbs
dbcc rebuild_text(table_name | table_id | "all" [, column [, text_page
    [, data_partition_name | data_partition_id]]])
dbcc reindex ({table_name | table_id})
dbcc serverlimits
dbcc stackused
```

dbcc tablealloc(*table_name* | *table_id* [, full | optimized | fast | NULL
[, fix | nofix | NULL [, *data_partition_name* | *data_partition_id*]]])

dbcc { traceon | traceoff } (*flag* [, flag ...])

dbcc tune ({ ascinserts, {0 | 1 } , *table_name* |
cleanup, {0 | 1 } |
cpuaffinity, *start_cpu* {, on| off } |
des_greedyalloc, *dbid*, *object_name*,
" { on | off }" |
deviochar vdevno, "*batch_size*" |
doneinproc { 0 | 1 }})

dbcc upgrade_object([*dbid* | *dbname*
[, *object_name* | *object_type* [, force | check]]))

deallocate cursor

Makes a cursor inaccessible and releases all memory resources committed to that cursor.

deallocate [cursor] *cursor_name*

declare

Declares the name and type of local variables for a batch or procedure. Variable declaration:

declare @*variable_name datatype* [, @*variable_name datatype*]...

Variable assignment:

select @*variable* = {*expression* | *select_statement*}
[, @*variable* = {*expression* | *select_statement*} ...]
[from *table_list*] [where *search_conditions*]
[group by *group_by_list*] [having *search_conditions*]
[order by *order_by_list*] [compute *function_list* [by *by_list*]]

declare cursor

Defines a cursor, by associating a select statement with a cursor name.

declare *cursor_name*
[semi_sensitive | insensitive] [scroll | no scroll]
cursor for *select_statement*
[for {read only | update [of *column_name_list*]}]

delete

Removes rows from a table.

delete
[top *unsigned_integer*]
[from] [[*database*.]*owner*.]{*view_name*|*table_name*}
[where *search_conditions*]
[plan "*abstract plan*"]

delete [[database.]owner.]{table_name | view_name}
[from [[database.]owner.]{view_name [readpast]|
table_name [readpast] [(index {index_name | table_name }
[prefetch *size*][lru|mru])]}
[, [[database.]owner.]{view_name [readpast]|
table_name [readpast] [(index {index_name | table_name }
[prefetch size][lru|mru])]} ...]
[where search_conditions]]
[plan "*abstract plan*"]

```
delete [from] [[database.]owner.]{table_name|view_name}
    where current of cursor_name
```

delete statistics

Removes statistics from the sysstatistics system table.

```
delete [ shared ] statistics table_name
    [ partition data_partition_name ]
    [( column_name[, column_name ] ...)]
```

disk init

Makes a physical device or file usable by Adaptive Server.

```
disk init
    name = "device_name" ,
    physname = "physicalname" ,
    [vdevno = virtual_device_number ,]
    size = number_of_blocks
    [, vstart = virtual_address, cntrltype = controller_number ]
        [, dsync = { true | false } ]
    [, directio = {true | false} ]
```

disk mirror

Creates a software mirror that immediately takes over when the primary device fails.

```
disk mirror
    name = "device_name" , mirror = "physicalname"
    [, writes = { serial | noserial }]
```

disk refit

Rebuilds the master database's sysusages and sysdatabases system tables from information contained in sysdevices.

```
disk refit
```

disk reinit

Rebuilds the master database's sysdevices system table. Use disk reinit as part of the procedure to restore the master database.

```
disk reinit
    name = "device_name", physname = "physicalname" ,
    [vdevno = virtual_device_number ,]size = number_of_blocks
    [, vstart = virtual_address, cntrltype = controller_number]
        [, dsync = { true | false } ] [, directio = {true | false}]
```

disk remirror

Restarts disk mirroring after it is stopped by failure of a mirrored device or temporarily disabled by the disk unmirror command.

```
disk remirror name = "device_name"
```

disk resize

Dynamically increases the size of the device used by Adaptive Server.

```
disk resize name = "device_name", size = additional_space
```

disk unmirror

Suspends disk mirroring initiated with the disk mirror command to allow hardware maintenance or the changing of a hardware device.

```
disk unmirror
    name = "device_name" [ ,side = { "primary" | secondary }]
    [ ,mode = { retain | remove }]
```

drop database

Removes one or more databases from Adaptive Server.

```
drop database database_name [, database_name] ...
```

drop default

Removes a user-defined default.

```
drop default [owner.]default_name[, [owner.]default_name] ...
```

drop function (SQLJ)

Removes a SQLJ function.

```
drop func[tion] [owner.]function_name[, [owner.]function_name ] ...
```

drop index

Removes an index from a table in the current database.

```
drop index table_name.index_name[, table_name.index_name] ...
```

drop procedure

Removes a procedure.

```
drop proc[edure] [owner.]procedure_name
    [, [owner.]procedure_name] ...
```

drop role

Drops a user-defined role.

```
drop role role_name [with override]
```

drop rule

Removes a user-defined rule.

```
drop rule [owner.]rule_name [, [owner.]rule_name] ...
```

drop table

Removes a table definition and all of its data, indexes, partition properties, triggers, and permissions from the database.

```
drop table [[database.]owner.]table_name
    [, [[database.]owner.]table_name ] ...
```

drop trigger

Removes a trigger.

```
drop trigger [owner.]trigger_name[, [owner.]trigger_name] ...
```

drop view

Removes one or more views from the current database.

```
drop view [owner.]view_name [, [owner.]view_name] ...
```

dump database

Makes a backup copy of the entire database, including the transaction log, in a form that can be read in with load database.

dump database *database_name*
to [compress::[*compression_level*::]]*stripe_device*
[at *backup_server_name*]
[density = *density_value*,
blocksize = *number_bytes*,
capacity = *number_kilobytes*,
dumpvolume = *volume_name*,
file = *file_name*]
with verify[= header | full]
[stripe on [compress::[*compression_level*::]]*stripe_device*
[at *backup_server_name*]
[density = *density_value*,
blocksize = *number_bytes*,
capacity = *number_kilobytes*,
dumpvolume = *volume_name*,
file = *file_name*]]
[[stripe on [compress::[*compression_level*::]]*stripe_device*
[at *backup_server_name*]
[density = *density_value*,
blocksize = *number_bytes*,
capacity = *number_kilobytes*,
dumpvolume = *volume_name*,
file = *file_name*]]...]
[with {
density = *density_value*,
blocksize = *number_bytes*,
capacity = *number_kilobytes*,
compression = *compress_level*
dumpvolume = *volume_name*,
file = *file_name*,
[dismount | nodismount],
[nounload | unload],
passwd = *password*,
retaindays = *number_days*,
[noinit | init],
notify = {client | operator_console}
}]

dump transaction

Makes a copy of a transaction log and removes the inactive portion. To make a routine log dump:

dump tran[saction] *database_name*
to [compress::[*compression_level*::]]*stripe_device*
[at *backup_server_name*]
[density = *density_value*,
blocksize = *number_bytes*,
capacity = *number_kilobytes*,
dumpvolume = *volume_name*,
file = *file_name*]
[stripe on [compress::[*compression_level*::]]*stripe_device*
[at *backup_server_name*]
[density = *density_value*,
blocksize = *number_bytes*,
capacity = *number_kilobytes*,

```
        dumpvolume = volume_name,
        file = file_name]]
    [[stripe on [compress::[compression_level::]]stripe_device
        [at backup_server_name]
        [density = density_value,
        blocksize = number_bytes,
        capacity = number_kilobytes,
        dumpvolume = volume_name,
        file = file_name] ]...]
    [with {
        density = density_value,
        blocksize = number_bytes,
        capacity = number_kilobytes,
        compression = compress_level,
        dumpvolume = volume_name,
        file = file_name,
        [dismount | nodismount],
        [nounload | unload],
        retaindays = number_days,
        [noinit | init],
        notify = {client | operator_console},
        standby_access }]
```

To truncate the log without making a backup copy:

```
dump tran[saction] database_name with truncate_only
```

To truncate a log that is filled to capacity. *Use only as a last resort*:

```
dump tran[saction] database_name with no_log
```

To back up the log after a database device fails:

```
dump tran[saction] database_name
    to [compress::[compression_level::]]stripe_device
        [at backup_server_name]
        [density = density_value,
        blocksize = number_bytes,
        capacity = number_kilobytes,
        dumpvolume = volume_name,
        file = file_name]
    [stripe on [compress::[compression_level::]]stripe_device
        [at backup_server_name]
        [density = density_value,
        blocksize = number_bytes,
        capacity = number_kilobytes,
        dumpvolume = volume_name,
        file = file_name]]
    [[stripe on [compress::[compression_level::]]stripe_device
        [at backup_server_name]
        [density = density_value,
        blocksize = number_bytes,
        capacity = number_kilobytes,
        dumpvolume = volume_name,
        file = file_name] ]...]
    [with {
        density = density_value,
        blocksize = number_bytes,
```

```
    capacity = number_kilobytes,
    compression = compress_level
    dumpvolume = volume_name,
    file = file_name,
    [dismount | nodismount],
    [nounload | unload],
    retaindays = number_days,
    [noinit | init],
    no_truncate,
    notify = {client | operator_console}}]
```

execute

Runs a procedure or dynamically executes Transact-SQL commands.

```
[exec[ute]] [@return_status  = ]
   [[[server .]database.]owner.]procedure_name[;number]
      [[@parameter_name =] value  |
         [@parameter_name =] @variable [output]
      [,[@parameter_name =] value  |
         [@parameter_name =] @variable [output]...]]
   [with recompile]
exec[ute] ("string" | char_variable [+ "string" | char_variable]...)
```

fetch

Returns a row or a set of rows from a cursor result set.

```
fetch [next |prior | first | last | absolute
   fetch_offset | relative fetch_offset ]
   [ from ] cursor_name[ into fetch_target_list ]
```

goto label

Branches to a user-defined label.

```
label: goto label
```

grant

Assigns permissions to individual users, groups of users, and roles. Assigns roles to users or system or user-defined roles. To grant permission to access database objects:

```
grant {all [privileges]| permission_list}
   on { table_name [(column_list)]
      | view_name[(column_list)]
      | stored_procedure_name}
   to {public | name_list | role_list}
   [with grant option]
```

To grant permission to use built-in functions:

```
grant select on [builtin] builtin to { name_list | role_list }
```

To grant permission to execute certain commands:

```
grant {all [privileges] | command_list}
   to {public | name_list | role_list}
```

To grant access on certain dbcc commands:

```
grant dbcc {dbcc_command [on {all | database }]
      [, dbcc_command [on {all | database }], ...]}
   to { user_list | role_list }
```

To grant the default permissions for specific system tables:

```
grant default permissions on system tables
```

To grant a role to a user or a role:

```
grant {role role_granted [, role_granted ...]}
    to grantee [, grantee...]
```

To switch your server user identity to any other server login and limit its use based on the target login roles:

```
grant set proxy to role_list
    [restricted role role_list | all | system]
```

group by and having clauses

Used in select statements to divide a table into groups and to return only groups that match conditions in the having clause.

```
Start of select statement
[group by [all] aggregate_free_expression
    [, aggregate_free_expression]...]
[having search_conditions]
End of select statement
```

if...else

Imposes conditions on the execution of a SQL statement.

```
if logical_expression [plan "abstract plan"]
    statements
[else
    [if logical_expression] [plan "abstract plan"]
        statement]
```

insert

Adds new rows to a table or view.

```
insert [into] [database.[owner.]]{table_name|view_name}
    [(column_list)]
    {values (expression [, expression]...)
        |select_statement [plan "abstract plan"] }
```

kill

Kills a process.

```
kill spid with statusonly
```

load database

Loads a backup copy of a user database, including its transaction log, that was created with dump database. To make a routine database load:

```
load database database_name
    from [compression=]stripe_device
        [at backup_server_name ]
        [density = density_value,
        blocksize = number_bytes,
        dumpvolume = volume_name,
        file = file_name]
        with verify only [ = header | full]
```

```
    [stripe on [compression=]stripe_device
        [at backup_server_name ]
        [density = density_value,
        blocksize = number_bytes,
        dumpvolume = volume_name,
        file = file_name]
    [[stripe on [compression=]stripe_device
        [at backup_server_name ]
        [density = density_value, blocksize = number_bytes,
        dumpvolume = volume_name, file = file_name]]...]
    [with {
        density = density_value, blocksize = number_bytes,
        compression, dumpvolume = volume_name, file = file_name,
        [dismount | nodismount], [nounload | unload],
        passwd = password, notify = {client | operator_console}
        }]]
```

To return header or file information without loading the backup:

```
load database database_name
    from [compress::]stripe_device
        [at backup_server_name ]
        [density = density_value, blocksize = number_bytes,
        dumpvolume = volume_name, file = file_name]
    [stripe on [compress::]stripe_device
        [at backup_server_name ]
        [density = density_value, blocksize = number_bytes,
        dumpvolume = volume_name, file = file_name]
    [[stripe on [compress::]stripe_device
        [at backup_server_name ] [density = density_value,
        blocksize = number_bytes, dumpvolume = volume_name,
        file = file_name]]...]
    [with {
        density = density_value,,
        blocksize = number_bytes,
        compression, dumpvolume = volume_name,
        file = file_name,
        [dismount | nodismount], [nounload | unload],
        passwd = password,
        listonly [= full], headeronly,
        notify = {client | operator_console}
        }]]
```

load transaction

Loads a backup copy of the transaction log that was created with dump transaction. To make a routine log load:

```
load tran[saction] database_name
    from [compress::]stripe_device
        [at backup_server_name ]
        [density = density_value,
        blocksize = number_bytes,
        dumpvolume = volume_name,
        file = file_name]
    [stripe on [compress::]stripe_device
        [at backup_server_name ]
```

```
        [density = density_value,
        blocksize = number_bytes,
        dumpvolume = volume_name,
        file = file_name]
    [[stripe on [compress::]stripe_device
        [at backup_server_name ]
        [density = density_value,
        blocksize = number_bytes,
        dumpvolume = volume_name,
        file = file_name]]...]
    [with {
        density = density_value,
        blocksize = number_bytes,
        compression,
        dumpvolume = volume_name,
        file = file_name,
        [dismount | nodismount],
        [nounload | unload],
        notify = {client | operator_console}
        }]]
```

To return header or file information without loading the backup log:

```
load tran[saction] database_name
    from [compress::]stripe_device
        [at backup_server_name ]
        [density = density_value,
        blocksize = number_bytes,
        dumpvolume = volume_name,
        file = file_name]
    [stripe on [compress::]stripe_device
        [at backup_server_name ]
        [density = density_value,
        blocksize = number_bytes,
        dumpvolume = volume_name,
        file = file_name]
    [[stripe on [compress::]stripe_device
        [at backup_server_name ]
        [density = density_value,
        blocksize = number_bytes,
        dumpvolume = volume_name,
        file = file_name]]...]
    [with {
        density = density_value,
        blocksize = number_bytes,
        compression,
        dumpvolume = volume_name,
        file = file_name,
        [dismount | nodismount],
        [nounload | unload],
        listonly [= full],
        headeronly,
        notify = {client | operator_console}
        until_time = datetime}]]
```

lock table

Explicitly locks a table within a transaction.

```
lock table table_name in {share | exclusive } mode
  [ wait [ numsecs ] | nowait ]
```

mount

Attaches a database to a destination or secondary Adaptive Server. You cannot execute mount without first performing unmount on your database.

```
mount database all from manifest_file [with listonly]
```

online database

Marks a database available for public use after a normal load sequence; if needed, upgrades a loaded database to the current version of Adaptive Server; brings a database online after loading a transaction log dumped with the for standby_access option.

```
online database database_name [for standby_access]
```

open

Opens a cursor for processing.

```
open cursor_name
```

order by clause

Returns query results in the specified columns in sorted order.

```
[Start of select statement]
[order by {[table_name.| view_name.]column_name
      | select_list_number | expression} [asc | desc]
  [,{[table_name.| view_name.] column_name
      select_list_number|expression} [asc|desc]]...]
[End of select statement]
```

prepare transaction

Used by DB-Library in a two-phase commit application to see if a server is prepared to commit a transaction.

```
prepare tran[saction]
```

print

Prints a user-defined message on the user's screen.

```
print
  {format_string | @local_variable | @@global_variable} [, arg_list]
```

quiesce database

Suspends and resumes updates to a specified list of databases.

```
quiesce database tag_name hold database_list [for external dump]
  [to manifest_file [with override]]
quiesce database tag_name release
```

raiserror

Prints a user-defined error message on the user's screen and sets a system flag to record that an error condition has occurred.

```
raiserror error_number
    [{format_string | @local_variable}] [, arg_list]
    [with errordata restricted_select_list]
```

readtext

Reads text, unitext, and image values, starting from a specified offset and reading a specified number of bytes or characters.

```
readtext [[database.]owner.]table_name.column_name
    text_pointer offset size
    [holdlock | noholdlock] [readpast]
    [using {bytes | chars | characters}]
    [at isolation {
        [ read uncommitted | 0 ] |
        [ read committed | 1 ] |
        [ repeatable read | 2 ]|
        [ serializable | 3 ] } ]
```

reconfigure

The reconfigure command currently has no effect; it is included to allow existing scripts to run without modification.

```
reconfigure
```

remove java

Removes one or more Java-SQL classes, packages, or JARs from a database.

```
remove java
    class class_name [, class_name]...
        | package package_name [, package_name]...
        | jar jar_name [, jar_name]...[retain classes]
```

reorg

Reclaims unused space on pages, removes row forwarding, or rewrites all rows in the table to new pages, depending on the option used.

```
reorg forwarded_rows table_name [partition partition_name]
    [with {resume, time = no_of_minutes}]

reorg reclaim_space table_name [index_name] [partition partition_name]
    [with {resume, time = no_of_minutes}]

reorg compact table_name [partition partition_name]
    [with {resume, time = no_of_minutes}]
reorg rebuild table_name [index_name  [partition index_partition_name]]
```

return

Exits from a batch or procedure unconditionally and provides an optional return status. Statements following return are not executed.

```
return [integer_expression] [plan "abstract plan"]
```

revoke

Revokes permissions or roles from users, groups, or roles. To revoke permission to access database objects:

```
revoke [grant option for]
    {all [privileges] | permission_list}
```

```
    on { table_name [(column_list)]
        | view_name [(column_list)]
        | stored_procedure_name}
    from {public | name_list | role_list}
    [cascade]
```

To revoke permission to select built-in functions:

```
revoke select
    on [builtin] builtin
    to { name_list | role_list }
```

To revoke permission to create database objects, execute set proxy, or execute set session authorization:

```
revoke {all [privileges] | command_list }
    from {public | name_list | role_list}
```

To revoke a role from a user or another role:

```
revoke role {role_name [, role_list ...]} from
    {grantee [, grantee ...]}
```

To revoke access on some dbcc commands:

```
revoke dbcc {dbcc_command [on {all | database }]
        [, dbcc_command [on {all | database }], ...]}
    from { user_list | role_list }
```

To revoke the default permissions from public:

```
revoke default permissions on system tables
```

rollback

Rolls back a user-defined transaction to the named savepoint in the transaction or to the beginning of the transaction.

```
rollback [tran | transaction | work]
    [transaction_name | savepoint_name]
```

rollback trigger

Rolls back the work done in a trigger, including the data modification that caused the trigger to fire, and issues an optional raiserror statement.

```
rollback trigger [with raiserror_statement]
```

save transaction

Sets a savepoint within a transaction.

```
save transaction savepoint_name
```

select

Retrieves rows from database objects.

```
select ::=
    select [ all | distinct ]
    [top unsigned_integer]
    select_list
    [into_clause ]
    [from_clause ]
    [where_clause ]
    [group_by_clause]
    [having_clause ]
```

```
    [order_by_clause ]
    [compute_clause ]
    [read_only_clause ]
    [isolation_clause ]
    [browse_clause ]
    [plan_clause ]
select_list ::=
```

Note For details on *select_list*, see the "Parameters" section for select in *Reference Manual: Commands*.

```
into_clause ::=
    into [ [ database.] owner.] table_name
        [ { [ external table at ]
            'server_name.[database].[owner].object_name'
            | external directory at 'pathname'
            | external file at 'pathname' [column deliminter 'string' ] } ]
        [ on segment_name ]
        [ partition_clause ]
        [ lock { datarows | datapages | allpages } ]
        [ with [, into_option[, into_option] ...] ] ]

    | into existing table table_name

    partition_clause ::=
        partition by range ( column_name[, column_name ]...)
            ( [ partition_name ] values <= ( { constant | MAX }
                [, { constant | MAX } ] ...) [ on segment_name ]
                [, [ partition_name ] values <= ( { constant | MAX }
                    [, { constant | MAX } ] ...) [ on segment_name ] ]...)

        | partition by hash (column_name[, column_name ]...)
            { ( partition_name [ on segment_name ]
                [, partition_name [ on segment_name ] ]...)
            | number_of_partitions
                [ on (segment_name[, segment_name ] ...) ] }

        | partition by list (column_name)
            ( [ partition_name ] values ( constant[, constant ] ...)
                [ on segment_name ]
                [, [ partition_name ] values ( constant[, constant ] ...)
                    [ on segment_name ] ] ...)

        | partition by roundrobin
            { ( partition_name [ on segment_name ]
                [, partition_name [ on segment_name ] ]...)
            | number_of_partitions
                [ on ( segment_name [, segment_name ]...) ] }
    into_option ::=
        | max_rows_per_page = num_rows
        | exp_row_size = num_bytes
        | reservepagegap = num_pages
        | identity_gap = gap
```

from_clause ::=
from *table_reference* [,*table_reference*]...
table_reference ::=
table_view_name | *ANSI_join*
table_view_name ::=
[[*database*.]*owner*.] {{*table_name* | *view_name*}
[as] [*correlation_name*]
[index {*index_name* | *table_name* }]
[parallel [*degree_of_parallelism*]]
[prefetch size][lru | mru]}
[holdlock | noholdlock]
[readpast]
[shared]
ANSI_join ::=
table_reference join_type join *table_reference*
join_conditions
join_type ::= inner | left [outer] | right [outer]
join_conditions ::= on *search_conditions*
where_clause ::= where *search_conditions*
group_by_clause ::=
group by [all] *aggregate_free_expression*
[, *aggregate_free_expression*]...
having_clause ::= having *search_conditions*
order_by_clause ::= order by *sort_clause* [, *sort_clause*]...
sort_clause ::=
{ [[[*database*.]*owner*.]{*table_name*.|*view_name*.}]*column_name*
| *select_list_number*
| *expression* }
[asc | desc]
compute_clause ::=
compute *row_aggregate*(*column_name*)
[, *row_aggregate*(*column_name*)]...
[by *column_name* [, *column_name*]...]
read_only_clause ::=
for {read only | update [of *column_name_list*]}
isolation_clause ::=
at isolation
{ read uncommitted | 0 }
| { read committed | 1 }
| { repeatable read | 2 }
| { serializable | 3 }
browse_clause ::= for browse
plan_clause ::= plan "abstract plan"

set

Sets Adaptive Server query-processing options for the duration of the user's work session; sets some options inside a trigger or stored procedure.

set *@variable* = *expression* [, *@variable* = *expression*...]
set ansinull {on | off}
set ansi_permissions {on | off}

set arithabort [arith_overflow | numeric_truncation]
{on | off}

set arithignore [arith_overflow] {on | off}

set bulk array size *number*

set bulk batch size *number*

set {chained, close on endtran, nocount, noexec, parseonly, procid,
self_recursion, showplan, sort_resources} {on | off}

set char_convert {off | on [with {error | no_error}] |
charset [with {error | no_error}]}

set cis_rpc_handling {on | off}

set [clientname *client_name* | clienthostname
host_name | clientapplname *application_name*]

set cursor rows *number* for *cursor_name*

set {datefirst *number*, dateformat *format*,
language *language*}

set delayed_commit { on | off | default }

set fipsflagger {on | off}

set flushmessage {on | off}

set forceplan {on | off}

set identity_insert [*database*.[*owner*.]]*table_name* {on | off}
set identity_update *table_name* {on | off}

set lock { wait [numsecs] | nowait }

set metrics_capture *on/off*

set offsets {select, from, order, compute, table,
procedure, statement, param, execute} {on | off}

set option *show*

set parallel_degree *number*

set plan {dump | load } [*group_name*] {on | off}

set plan exists check {on | off}

set plan for *show*

set plan optgoal allrows_mix | allrows_dss

set plan opttimeoutlimit *number*

set plan replace {on | off}

set prefetch [on|off]

set proc_output_params on | off

set proc_return_status on | off

set process_limit_action {abort | quiet | warning}

set proxy login_name

set quoted_identifier {on | off}

set repartition_degree *number*

set resource_granularity *number*

set role {"sa_role" | "sso_role" | "oper_role" |
role_name [with passwd "*password*"]} {on | off}

set {rowcount *number*, textsize *number*}

set scan_parallel_degree *number*

set session authorization login_name
set statistics {io, subquerycache, time, plancost} {on | off}
set statistics simulate { on | off }
set strict_dtm_enforcement {on | off}
set string_rtruncation {on | off}
set textsize {*number*}
set transaction isolation level {
[read uncommitted | 0] |
[read committed | 1] |
[repeatable read | 2]|
[serializable | 3] }
set transactional_rpc {on | off}

setuser

Allows a Database Owner to impersonate another user.

setuser ["*user_name*"]

shutdown

Shuts down the Adaptive Server from which the command is issued, its local Backup Server, or a remote Backup Server.

shutdown [*srvname*] [with {wait | nowait}]

truncate table

Removes all rows from a table or partition.

truncate table [[*database*.]*owner*.]*table_name*
[partition *partition_name]*

union operator

Returns a single result set that combines the results of two or more queries.

select [top *unsigned_integer*] *select_list*
[into *clause*] [from *clause*] [where *clause*]
[group by *clause*] [having *clause*]
[union [all]
select [top *unsigned_integer*] *select_list*
[from *clause*] [where *clause*]
[group by *clause*] [having *clause*]]...
[order by *clause*]
[compute *clause*]

unmount

Shuts down the database and drops it from the Adaptive Server. Do not use this command without first reading the full description in *Reference Manual: Commands*.

unmount database *dbname_list* to *manifest_file*

update

Changes data in existing rows, either by adding data or by modifying existing data.

update [top *unsigned_integer*]
[[*database*.]*owner*.]{*table_name* | *view_name*}
set [[[*database*.]*owner*.]{*table_name*.|*view_name*.}]

```
    column_name1 =
    {expression1 | NULL | (select_statement)} |
    variable_name1 =
    {expression1 | NULL | (select_statement)}
    [, column_name2 =
    {expression2 | NULL | (select_statement)}]... |
    [, variable_name2 =
    {expression2 | NULL | (select_statement)}]...

    [from [[database.]owner.]{view_name [readpast]|
        table_name [readpast]
            [(index {index_name | table_name}
            [ prefetch size ][lru|mru])]}
        [,[[database.]owner.]{view_name [readpast]|
        table_name [readpast]
            [(index {index_name | table_name }
            [ prefetch size ][lru|mru])]}]
    ...]
    [where search_conditions] [plan "abstract plan"]
update [[database.]owner.]{table_name | view_name}
    set [[[database.]owner.]{table_name.|view_name.}]
        column_name1 =
            {expression1 | NULL | (select_statement)} |
        variable_name1 =
            {expression1 | NULL | (select_statement)}
        [, column_name2 =
            {expression2 | NULL | (select_statement)}]... |
        [, variable_name2 =
            {expression2 | NULL | (select_statement)}]...
    where current of cursor_name
```

update all statistics

Updates all statistics information for a given table.

```
update all statistics table_name [ partition data_partition_name ]
```

update index statistics

Updates the statistics for all columns in an index.

```
update index statistics
    table_name [[ partition data_partition_name ] |
    [ index_name [ partition index_partition_name ] ] ]
    [ using step values ]
    [ with consumers = consumers ] [, sampling=N percent]
```

update statistics

Updates information about the distribution of key values in specified indexes, for all columns in an index, table, or partition.

```
update statistics table_name
    [[ partition data_partition_name ] [ (column_list ) ] |
    index_name [ partition index_partition_name ] ]
    [ using step values ]
    [ with consumers = consumers ][, sampling=N percent ]
```

update table statistics

Updates statistics that are stored in systabstats table, such as rowcount, cluster ratios, and so on.

update table statistics *table_name*
[partition *data_partition_name*]

use

Specifies the database with which you want to work.

use *database_name*

waitfor

Specifies a specific time, a time interval, or an event for the execution of a statement block, stored procedure, or transaction.

waitfor { delay *time* | time *time* | errorexit | processexit | mirrorexit }

where clause

Sets the search conditions in a select, insert, update, or delete statement.

where [not] *expression comparison_operator expression*

where {[not] *expression comparison_operator expression*} | { ... }

where [not] *expression* [not] like "*match_string*"
[escape "*escape_character* "]

where [not] *expression* is [not] null

where [not] expression [not] between *expression* and *expression*

where [not]*expression* [not] in ({*value_list* | *subquery*})

where [not] exists (*subquery*)

where [not] *expression comparison_operator* {any | all} (*subquery*)

where [not] *column_name join_operator column_name*

where [not] *logical_expression*

where [not] *expression* {and | or} [not] *expression*

while

Sets a condition for the repeated execution of a statement or statement block.

while *logical_expression* [plan "*abstract plan*"]
statement

writetext

Permits minimally logged, interactive updating of an existing text, unitext or image column.

writetext [[*database*.]*owner*.]*table_name*.*column_name*
text_pointer [readpast] [with log] *data*

Interactive dbsql commands

This section provides the syntax and brief descriptions for interactive dbsql commands for Adaptive Server. See *Reference Manual: Commands* for more information.

clear

Clears the Interactive SQL panes.

clear

configure

Opens the Interactive SQL Options dialog.

configure

connect

Establishes a connection to a database.

connect
[to *engine_name*]
[database *database_name*]
[as *connection_name*]
[user] *user_id* identified by password
engine_name, *database_name*, *connection_name*, *user_id* ,
password : { identifier | string | hostvar }

connect using *connect_string* : { identifier | string | hostvar }

disconnect

Drops the current connection to a database.

disconnect [*connection_name* | current | all]
connection_name : *identifier* , *string*, or *hostvar*

exit

Leaves Interactive SQL.

{ exit | quit | bye } [*return_code*]
return_code: *number* | *connection_variable*

input

Imports data into a database table from an external file or from the keyboard.

input into [*owner*.]*table-name*
[from *filename* | prompt]
[format *input-format*]
[escape character *character*]
[escapes { on | off }
[by order | by name]
[delimited by string]
[column widths (integer , . . .)]
[nostrip]
[(*column-name*, . . .)]
[encoding encoding]
input-format :
ascii | dbase | dbasell| dbaselll
| excel | fixed | foxpro | lotus

encoding : identifier or string

output

Imports data into a database table from an external file or from the keyboard.

output to *filename*
[append]
[verbose]
[format *output-format*]
[escape character *character*]
[escapes { on | off}
[delimited by string]
[quote string [all]]
[column widths (integer , . . .)]
[hexidecimal { on | off | asis }]
[encoding encoding]

output-format :
ascii | dbase | dbasell| dbaselll
| excel | fixed | foxpro | lotus | sql | xml

encoding : string or identifier

parameters

Specifies parameters to an Interactive SQL command file.

parameters parameter1, parameter2, . . .

read

Reads Interactive SQL statements from a file.

read [encoding encoding] *file_name* [*parameters*]
encoding : *identifier* or *string*

set connection

Changes the current database connection to another server.

set *connection_name* :*identifier* , *string*, or *hostvar*

set option

Changes the values of Interactive SQL options.

set [temporary] option
[*user_id*. | public.]*option_name* = [*option_value*]
user_id : *identifier* , *string* or *hostvar*
option_name : *identifier* , *string* or *hostvar*

set permanent

set

start logging

Starts logging executed SQL statements to a log file.

start logging *file_name*

stop logging

Stops logging of executed SQL statements in the current session.

stop logging

system

Launches an executable file from within Interactive SQL.

system '[*path*] *file_name*'

System procedures

This section provides the syntax and brief descriptions for Adaptive Server system stored procedures. See *Reference Manual: Procedures* for more information.

sp_activeroles

Displays all active roles.

sp_activeroles [expand_down]

sp_add_qpgroup

Adds an abstract plan group.

sp_add_qpgroup *new_name*

sp_add_resource_limit

Creates a limit on the number of server resources that can be used by an Adaptive Server login and/or an application to execute a query, query batch, or transaction.

sp_add_resource_limit *name*, *appname*, *rangename, limittype*, *limitvalue* [, *enforced* [, *action* [, *scope*]]]

sp_add_time_range

Adds a named time range to an Adaptive Server.

sp_add_time_range *name*, *startday*, *endday*, *starttime*, *endtime*

sp_addalias

Allows an Adaptive Server user to be known in a database as another user.

sp_addalias *loginame*, *name_in_db*

sp_addauditrecord

Allows users to enter user-defined audit records (comments) into the audit trail.

sp_addauditrecord [*text* [, *db_name* [, *obj_name* [, *owner_name* [, *dbid* [, *objid*]]]]]]

sp_addaudittable

Adds another system audit table after auditing is installed.

sp_addaudittable *devname*

sp_addengine

Adds an engine to an existing engine group or, if the group does not exist, creates an engine group and adds the engine.

sp_addengine *engine_number*, *engine_group*

sp_addexeclass

Creates or updates a user-defined execution class that you can bind to client applications, logins, and stored procedures.

sp_addexeclass *classname*, *priority*, *timeslice*, *engine_group*

sp_addextendedproc

Creates an extended stored procedure (ESP) in the master database.

sp_addextendedproc *esp_name*, *dll_name*

sp_addexternlogin

Component Integration Services only– creates an alternate login account and password to use when communicating with a remote server through Component Integration Services.

sp_addexternlogin *server*, *loginame*, *externname*
[, *externpasswd*] [*rolename*]

sp_addgroup

Adds a group to a database. Groups are used as collective names in granting and revoking privileges.

sp_addgroup *grpname*

sp_addlanguage

Defines the names of the months and days for an alternate language and its date format.

sp_addlanguage *language*, *alias*, *months*, *shortmons*,
days, *datefmt*, *datefirst*

sp_addlogin

Adds a new user account to Adaptive Server; specifies the password expiration interval, the minimum password length, and the maximum number of failed logins allowed for a specified login at creation.

sp_addlogin *loginame*, *passwd* [, *defdb*] [, *deflanguage*] [, *fullname*]
[, *passwdexp*] [, *minpwdlen*] [, *maxfailedlogins*] [, *auth_mech*]

sp_addmessage

Adds user-defined messages to sysusermessages for use by stored procedure print and raiserror calls and by sp_bindmsg.

sp_addmessage *message_num*, *message_text*
[, *language* [, *with_log* [, replace]]]

sp_addobjectdef

Component Integration Services only – specifies the mapping between a local table and an external storage location.

sp_addobjectdef *tablename*, *objectdef* [,"*objecttype*"]

sp_addremotelogin

Authorizes a new remote server user by adding an entry to master.dbo.sysremotelogins.

sp_addremotelogin *remoteserver* [, *loginame* [, *remotename*]]

sp_addsegment

Defines a segment on a database device in a database.

sp_addsegment *segname*, *dbname*, *devname*

sp_addserver

Defines a remote server, or defines the name of the local server.

sp_addserver *lname* [, *class* [, *pname*]]

sp_addthreshold

Creates a threshold to monitor space on a database segment.

sp_addthreshold *dbname*, *segname*, *free_space*, *proc_name*

sp_addtype

Creates a user-defined datatype.

sp_addtype *typename*,
phystype [(*length*) | (*precision* [, *scale*])] [, "identity" | *nulltype*]

sp_addumpdevice

Adds a dump device to Adaptive Server.

sp_addumpdevice {"tape" | "disk"}, *logicalname*,
physicalname [, *tapesize*]

sp_adduser

Adds a new user to the current database.

sp_adduser *loginame* [, *name_in_db* [, *grpname*]]

sp_altermessage

Enables and disables the logging of a system-defined or user-defined message in the Adaptive Server error log.

sp_altermessage *message_id*, *parameter, parameter_value*

sp_audit

Allows a System Security Officer to configure auditing options.

sp_audit *option*, *login_name*, *object_name* [,*setting*]
sp_audit 'restart'

sp_autoconnect

Component Integration Services only – defines a passthrough connection to a remote server for a specific user, which allows the named user to enter passthrough mode automatically at login.

sp_autoconnect *server*, {true|false} [, *loginame*]

sp_autoformat

Reformats the width of variable-length character data to display only non-blank characters.

sp_autoformat *fulltabname* [, *selectlist* , *whereclause* , *orderby*]

sp_bindcache

Binds a database, table, index, text object, or image object to a data cache.

sp_bindcache *cachename*, *dbname*
[, [*ownername*.]*tablename*
[, *indexname* | "text only"]]

sp_bindefault

Binds a user-defined default to a column or user-defined datatype.

sp_bindefault *defname*, *objname* [, futureonly]

sp_bindexeclass

Associates an execution class with a client application, login, or stored procedure.

sp_bindexeclass "*object_name*", "*object_type*", "*scope*", "*classname*"

sp_bindmsg

Binds a user message to a referential integrity constraint or check constraint.

```
sp_bindmsg constrname, msgid
```

sp_bindrule

Binds a rule to a column or user-defined datatype.

```
sp_bindrule rulename, objname [, futureonly]
```

sp_cacheconfig

Creates, configures, reconfigures, and drops data caches, and provides information about them.

```
sp_cacheconfig [cachename [ ,"cache_size[P|K|M|G]" ]
    [,logonly | mixed ] [,strict | relaxed ] ]
    [, "cache_partition=[1|2|4|8|16|32|64]"]
```

sp_cachestrategy

Enables or disables prefetching (large I/O) and MRU cache replacement strategy for a table, index, text object, or image object.

```
sp_cachestrategy dbname, [ownername.]tablename
    [, indexname | "text only" | "table only"
    [, { prefetch | mru }, { "on" | "off"}]]
```

sp_changedbowner

Changes the owner of a user database.

```
sp_changedbowner loginame [, true ]
```

sp_changegroup

Changes a user's group.

```
sp_changegroup grpname, username
```

sp_checknames

Checks the current database for names that contain characters not in the 7-bit ASCII set.

```
sp_checknames [help | silent]
```

sp_checkreswords

Detects and displays identifiers that are Transact-SQL reserved words. Checks server names, device names, database names, segment names, user-defined datatypes, object names, column names, user names, login names, and remote login names.

```
sp_checkreswords [user_name_param]
```

sp_checksource

Checks for the existence of the source text of the compiled object, and for the existence of computed column source text.

```
sp_checksource [objname [, tabname [, username]]]
```

sp_chgattribute

Changes the max_rows_per_page, fillfactor, reservepagegap, or exp_row_size value for future space allocations of a table or an index; sets the concurrency_opt_threshold for a table. Provides the user interface for optimistic index locking.

sp_chgattribute *objname*, {"max_rows_per_page" | "fillfactor" |
"reservepagegap" | "exp_row_size"
concurrency_opt_threshold | “optimistic_index_lock” |
“identity_burn_max”}, *value*, *optvalue*

sp_chgattribute *objname*, {"identity_gap", *set_number* |
“dealloc_first_txtpg”, *value*}

sp_clearpsexe

Clears the execution attributes of an Adaptive Server session that was set by ssp_setpsexe.

sp_clearpsexe *spid*, *exeattr*

sp_clearstats

Initiates a new accounting period for all server users or for a specified user. Prints statistics for the previous period by executing sp_reportstats.

sp_clearstats [*loginame*]

sp_client_addr

Displays the IP (Internet Protocol) address of every Adaptive Server task with an attached client application, including the spid and the client host name.

sp_client addr[“*spid*”]

sp_cmp_all_qplans

Compares all abstract plans in two abstract plan groups.

sp_cmp_all_qplans *group1*, *group2* [, *mode*]

sp_cmp_qplans

Compares two abstract plans.

sp_cmp_qplans *id1*, *id2*

sp_commonkey

Defines a common key—columns that are frequently joined—between two tables or views.

sp_commonkey *tabaname*, *tabbname*, *col1a*, *col1b*
[, *col2a*, *col2b*, ..., *col8a*, *col8b*]

sp_companion

Performs cluster operations such as configuring Adaptive Server as a secondary companion in a high availability system and moving a companion server from one failover mode to another.

sp_companion
[*server_name*
{, configure
[, {with_proxydb | NULL}]
[, *srvlogin*]
[, *server_password*]
[, *cluster_login*]
[, *cluspassword*]]
| drop
| suspend
| resume

| prepare_failback
| do_advisory}
{, all
| help
| group attribute_name
| base attribute_name}

sp_configure

Displays configuration parameters by group, their current values, their default values, the value to which they have most recently been set, and the amount of memory used by this setting.

sp_configure [*configname* [, *configvalue*] | *group_name* | *non_unique_parameter_fragment*]

sp_configure "configuration file", 0, {"write" | "read" | "verify" | "restore"} "*file_name*"

sp_copy_all_qplans

Copies all plans for one abstract plan group to another group.

sp_copy_all_qplans *src_group*, *dest_group*

sp_copy_qplan

Copies one abstract plan to an abstract plan group.

sp_copy_qplan *src_id*, *dest_group*

sp_countmetadata

Displays the number of indexes, objects, or databases in Adaptive Server.

sp_countmetadata "*configname*" [, *dbname*]

sp_cursorinfo

Reports information about a specific cursor or all execute cursors that are active for your session.

sp_cursorinfo [{*cursor_level* | null}] [, *cursor_name*]

sp_dbextend

Allows you to install automatic database expansion procedures on database/segment pairs and devices, define site-specific policies for individual segments and devices, and simulate execution of the database expansion machinery, to study the operation before engaging large volume loads.

sp_dbextend 'help'[, <*command* >]

sp_dbextend [['set', ['threshold', *dbname*, *segmentname*, *freespace* | 'database', *dbname*, *segmentname* {[[, *growby*] [, *maxsize*]]} | 'device', *devicename* { [[, *growby*] [, *maxsize*]] }] | 'clear', 'threshold', *dbname*, *segmentname*

sp_dbextend 'clear', 'database' [, *dbname* [, *segmentname*]]

sp_dbextend 'clear', 'device' [, *devicename*]

sp_dbextend 'modify', 'database', *dbname*, *segmentname*, { 'growby' | 'maxsize' }, *newvalue*

sp_dbextend 'modify', 'device', *devicename*, { 'growby' | ' maxsize ' }, *newvalue*

sp_dbextend { 'list' | 'listfull' } [, 'database' [, *dbname* [, *segmentname* [, *order_by_clause*]]]]

sp_dbextend { 'list' | 'listfull' } [, 'device' [, *devicename* [, *order_by_clause*]]]

sp_dbextend { 'list' | 'listfull' }, ['threshold' [, *@dbname* [, *@segmentname*]]]

sp_dbextend 'check', 'database' [, *dbname* [, *segmentname*]]

sp_dbextend { 'simulate' | 'execute' }, *dbname*, *segmentname*[, *iterations*]

sp_dbextend 'trace', {'on' | 'off' }

sp_dbextend 'reload [defaults]'

sp_dbextend { 'enable' | 'disable' }, 'database' [, *dbname* [, *segmentname*]]

sp_dbextend 'who' [, '<spid>' | 'block' | 'all']

sp_dboption

Displays or changes database options, and enables the asynchronous log service feature.

sp_dboption [*dbname*, *optname*, *optvalue* [, *dockpt*]]

sp_dbrecovery_order

Specifies the order in which user databases are recovered and lists the user-defined recovery order of a database or all databases.

sp_dbrecovery_order
[*database_name* [, *rec_order* [, force]]]

sp_dbremap

Forces Adaptive Server to recognize changes made by alter database. Run this procedure only when instructed to do so by an Adaptive Server message.

sp_dbremap *dbname*

sp_defaultloc

Component Integration Services only – defines a default storage location for objects in a local database.

sp_defaultloc *dbname*, *defaultloc*, *defaulttype*

sp_depends

Displays information about database object dependencies.

sp_depends *objname*[, *column_name*]

sp_deviceattr

UNIX platforms only – changes the device parameter settings of an existing database device file.

sp_deviceattr *logicalname*, *optname*, *optvalue*

sp_diskdefault

Specifies whether or not a database device can be used for database storage if the user does not specify a database device or specifies default with the create database or alter database commands.

sp_diskdefault *logicalname,* {defaulton | defaultoff}

sp_displayaudit

Displays the status of audit options.

sp_displayaudit ["procedure" | "object" | "login" | "database" | "global" | "default_object" | "default_procedure" [, "*name*"]]

sp_displaylevel

Sets or shows which Adaptive Server configuration parameters appear in sp_configure output.

sp_displaylevel [*loginame* [, *level*]]

sp_displaylogin

Displays information about a login account. Also displays information about the hierarchy tree above or below the login account when you so specify.

sp_displaylogin [*loginame* [, expand_up | expand_down]]

sp_displayroles

Displays all roles granted to another role, or displays the entire hierarchy tree of roles in table format.

sp_displayroles [*grantee_name* [, *mode*]]

sp_dropalias

Removes the alias user name identity established with sp_addalias.

sp_dropalias *loginame* [, force]

sp_drop_all_qplans

Deletes all abstract plans in an abstract plan group.

sp_drop_all_qplans *name*

sp_drop_qpgroup

Drops an abstract plan group.

sp_drop_qpgroup *group*

sp_drop_qplan

Drops an abstract plan.

sp_drop_qplan *id*

sp_drop_resource_limit

Removes one or more resource limits from Adaptive Server.

sp_drop_resource_limit { *name*, *appname* }
[, *rangename*, *limittype*, *enforced*, *action*, *scope*]

sp_drop_time_range

Removes a user-defined time range from Adaptive Server.

sp_drop_time_range *name*

sp_dropdevice

Drops an Adaptive Server database device or dump device.

sp_dropdevice *logicalname*

sp_dropengine

Drops an engine from a specified engine group or, if the engine is the last one in the group, drops the engine group.

sp_dropengine *engine_number*, *engine_group*

sp_dropexeclass

Drops a user-defined execution class.

sp_dropexeclass *classname*

sp_dropextendedproc

Removes an extended stored procedure (ESP).

sp_dropextendedproc *esp_name*

sp_dropexternlogin

Component Integration Services only – Drops the definition of a remote login previously defined by sp_addexternlogin.

sp_dropexternlogin *server* [, *loginame* [, *rolename*]]

sp_dropglockpromote

Removes lock promotion values from a table or database.

sp_dropglockpromote {"database" | "table"}, *objname*

sp_dropgroup

Drops a group from a database.

sp_dropgroup *grpname*

sp_dropkey

Removes from the syskeys table a key that had been defined using sp_primarykey, sp_foreignkey, or sp_commonkey.

sp_dropkey *keytype*, *tabname* [, *deptabname*]

sp_droplanguage

Drops an alternate language from the server and removes its row from master.dbo.syslanguages.

sp_droplanguage *language* [, dropmessages]

sp_droplogin

Drops an Adaptive Server user login by deleting the user's entry from master.dbo.syslogins.

sp_droplogin *loginame*

sp_dropmessage

Drops user-defined messages from sysusermessages.

sp_dropmessage *message_num* [, *language*]

sp_dropobjectdef

Component Integration Services only – deletes the external storage mapping provided for a local object.

sp_dropobjectdef *tablename*

sp_dropremotelogin

Drops a remote user login.

sp_dropremotelogin *remoteserver* [, *loginame* [, *remotename*]]

sp_droprowlockpromote

Removes row lock promotion threshold values from a database or table.

sp_droprowlockpromote {"database" | "table"}, *objname*

sp_dropsegment

Drops a segment from a database or unmaps a segment from a particular database device.

sp_dropsegment *segname*, *dbname* [, *device*]

sp_dropserver

Drops a server from the list of known servers or drops remote logins and external logins in the same operation.

sp_dropserver *server* [, droplogins]

sp_dropthreshold

Removes a free-space threshold from a segment.

sp_dropthreshold *dbname*, *segname*, *free_space*

sp_droptype

Drops a user-defined datatype.

sp_droptype *typename*

sp_dropuser

Drops a user from the current database.

sp_dropuser *name_in_db*

sp_dumpoptimize

Specifies the amount of data dumped by Backup Server during the dump database operation.

sp_dumpoptimize ['archive_space = {maximum | minimum | default }']
sp_dumpoptimize ['reserved_threshold = {*nnn* | default }']
sp_dumpoptimize ['allocation_threshold = {*nnn* | default }']

sp_engine

Enables you to bring an engine online or offline.

sp_engine {"online" | [offline | can_offline] [, *engine_id*] |
["shutdown", *engine_id*]}

sp_estspace

Estimates the amount of space required for a table and its indexes, and the time needed to create the index.

sp_estspace *table_name*, *no_of_rows*, *fill_factor*,
cols_to_max, *textbin_len*, *iosec*, *page_size*

sp_export_qpgroup

Exports all plans for a specified user and abstract plan group to a user table.

sp_export_qpgroup *usr*, *group*, *tab*

sp_extendsegment

Extends the range of a segment to another database device.

sp_extendsegment *segname*, *dbname*, *devname*

sp_extengine

Starts and stops EJB Server. Displays status information about EJB Server.

sp_extengine '*ejb_server*', '{ start | stop | status }'

sp_familylock

Reports information about all the locks held by a family (coordinating process and its worker processes) executing a statement in parallel.

sp_familylock [*fpid1* [, *fpid2*]]

sp_find_qplan

Finds an abstract plan, given a pattern from the query text or plan text.

sp_find_qplan *pattern* [, group]

sp_fixindex

Repairs the index on one of your system tables when it has been corrupted.

sp_fixindex *dbname*, *tabname*, *indid*

sp_flushstats

Flushes statistics from in-memory storage to the systabstats and sysstatistics system tables.

sp_flushstats *objname*

sp_forceonline_db

Provides access to all the pages in a database that were previously marked suspect by recovery.

sp_forceonline_db *dbname*, {"sa_on" | "sa_off" | "all_users"}

sp_forceonline_object

Provides access to an index previously marked suspect by recovery.

sp_forceonline_object *dbname*, *objname*, *indid*,
{sa_on | sa_off | all_users} [, no_print]

sp_forceonline_page

Provides access to pages previously marked suspect by recovery.

sp_forceonline_page *dbname*, *pgid*, {"sa_on" | "sa_off" | "all_users"}

sp_foreignkey

Defines a foreign key on a table or view in the current database.

sp_foreignkey *tabname*, *pktabname*, *col1* [, *col2*] ... [, *col8*]

sp_freedll

Unloads a dynamic link library (DLL) that was previously loaded into XP Server memory to support the execution of an extended stored procedure (ESP).

sp_freedll *dll_name*

sp_getmessage

Retrieves stored message strings from sysmessages and sysusermessages for print and raiserror statements.

sp_getmessage *message_num*, *result* output [, *language*]

sp_grantlogin

Windows NT only – assigns Adaptive Server roles or default permissions to Windows NT users and groups when Integrated Security mode or Mixed mode (with Named Pipes) is active.

sp_grantlogin {*login_name* | *group_name*} ["*role_list*" | default]

sp_ha_admin

Performs administrative tasks on Adaptive Servers configured with Sybase Failover in a high availability system.

sp_ha_admnin [cleansessions | help]

sp_help

Reports information about a database object (any object listed in sysobjects) and about system or user-defined datatypes, as well as computed columns and function-based indexes.

sp_help [*objname*]

sp_help_resource_limit

Reports on resource limits.

sp_help_resource_limit [*name* [, *appname* [, *limittime* [, *limitday* [, *scope* [, *action*[, verbose]]]]]]]

sp_help_qpgroup

Reports information on an abstract plan group.

sp_help_qpgroup [*group* [, *mode*]]

sp_help_qplan

Reports information about an abstract plan.

sp_help_qplan *id* [, *mode*]

sp_helpartition

Lists partition-related information of a table or index.

sp_helpartition [*tabname* [, { null | *indexname* | 'all' }[, *partitionname*]]]

sp_helpcache

Displays information about the objects that are bound to a data cache or the amount of overhead required for a specified cache size.

sp_helpcache {*cache_name* | "*cache_size*[P|K|M|G]"}

sp_helpcomputedcolumn

Reports information on the computed columns in a specified table.

sp_helpcomputedcolumn {tabname}

sp_helpconfig

Reports help information on configuration parameters.

sp_helpconfig "*configname*", ["*size*"]

sp_helpconstraint

Reports information about integrity constraints used in the specified tables.

sp_helpconstraint [*objname*] [, detail]

sp_helpdb

Reports information about a particular database or about all databases.

sp_helpdb [*dbname* [, *order*]]

sp_helpdevice

Reports information about a particular device or about all Adaptive Server database devices and dump devices.

sp_helpdevice [*devname*]

sp_helpextendedproc

Displays extended stored procedures (ESPs) in the current database, along with their associated DLL files.

sp_helpextendedproc [*esp_name*]

sp_helpexternlogin

Component Integration Services only – reports information about external login names.

sp_helpexternlogin [*server* [, *loginame* [, *rolename*]]]

sp_helpgroup

Reports information about a particular group or about all groups in the current database.

sp_helpgroup [*grpname*]

sp_helpindex

Reports information about the indexes created on a table, and on computed column indexes and function-based indexes.

sp_helpindex *objname*

sp_helpjava

Displays information about Java classes and associated JARs that are installed in the database.

sp_helpjava ["class" [, *java_class_name* [, “detail” | “depends”]] |
"jar" [, *jar_name* [, “depends”]]]

sp_helpjoins

Lists the columns in two tables or views that are likely join candidates.

sp_helpjoins *lefttab*, *righttab*

sp_helpkey

Reports information about a primary, foreign, or common key of a particular table or view, or about all keys in the current database.

sp_helpkey [*tabname*]

sp_helplanguage

Reports information about a particular alternate language or about all languages.

```
sp_helplanguage [language]
```

sp_helplog

Reports the name of the device that contains the first page of the transaction log.

```
sp_helplog
```

sp_helpobjectdef

Component Integration Services only – reports owners, objects, and type information for remote object definitions.

```
sp_helpobjectdef [objname]
```

sp_helpremotelogin

Reports information about a particular remote server's logins or about all remote server logins.

```
sp_helpremotelogin [remoteserver [, remotename]]
```

sp_helprotect

Reports on permissions for database objects, users, groups, or roles.

```
sp_helprotect [name [, username [, "grant"
    [,"none"|"granted"|"enabled"|role_name]]]]
```

sp_helpsegment

Reports information about a particular segment or about all segments in the current database.

```
sp_helpsegment [segname]
```

sp_helpserver

Reports information about a particular remote server or about all remote servers.

```
sp_helpserver [server]
```

sp_helpsort

Displays Adaptive Server's default sort order and character set.

```
sp_helpsort
```

sp_helptext

Displays the source text of a compiled object. Displays the source text of computed columns or function-based index definitions.

```
sp_helptext objname [,number]
```

sp_helpthreshold

Reports the segment, free-space value, status, and stored procedure associated with all thresholds in the current database or all thresholds for a particular segment.

```
sp_helpthreshold [segname]
```

sp_helpuser

Reports information about a particular user, group, or alias, or about all users, in the current database.

```
sp_helpuser [name_in_db]
```

sp_hidetext

Hides the source text for the specified compiled object. Hides the text of computed columns and function-based index keys.

sp_hidetext [*objname* [, *tabname* [, *username*]]]

sp_import_qpgroup

Imports abstract plans from a user table into an abstract plan group.

sp_import_qpgroup *tab*, *usr*, *group*

sp_indsuspect

Checks user tables for indexes marked as suspect during recovery following a sort order change.

sp_indsuspect [*tab_name*]

sp_ldapadmin

Creates an LDAP URL search string, lists an LDAP URL search string, or verifies an LDAP URL search string or login.

```
sp_ldapadmin { set_primary_url, 'ldapurl' |
    set_secondary_url, { 'ldapurl' | null } |
    set_access_acct, account_distinguished_name, account_password |
    set_dn_lookup_url, distinguished_name_url |
    list_urls | check_url, 'ldapurl' |
    check_login, 'login_name' }
```

sp_listener

Dynamically starts and stops listeners on Adaptive Server on any given port on a per-engine basis.

sp_listener "*command*", "*server_name*", *engine* | *remaining*

sp_listener "*command*", "[*protocol*:]*machine*:*port*", *engine*

sp_listsuspect_db

Lists all databases that currently have offline pages because of corruption detected on recovery.

sp_listsuspect_db

sp_listsuspect_object

Lists all indexes in a database that are currently offline because of corruption detected on recovery.

sp_listsuspect_object [*dbname*]

sp_listsuspect_page

Lists all pages in a database that are currently offline because of corruption detected on recovery.

sp_listsuspect_page [*dbname*]

sp_lmconfig

Configures license management-related information on Adaptive Server.

```
sp_lmconfig "edition", edition_type,
    "license type", license_type_name,
    "smtp host", smtp_host_name,
```

"smtp port", *smtp_port_number*,
"email sender", *sender_email_address*,
"email recipients", *email_recipients*,
"email severity", *email_severity*

sp_lock

Reports information about processes that currently hold locks.

sp_lock [*spid1* [, *spid2*]]

sp_locklogin

Locks an Adaptive Server account so that the user cannot log in, or displays a list of all locked accounts.

sp_locklogin *login* | all | NULL | *wildcard_string* , "lock" |"unlock",
[*except_login_name* | *except_role_name*]

sp_locklogin

sp_logdevice

Moves the transaction log of a database with log and data on the same device to a separate database device.

sp_logdevice *dbname*, *devname*

sp_loginconfig

Windows NT only – displays the value of one or all integrated security parameters.

sp_loginconfig ["*parameter_name*"]

sp_logininfo

Windows NT only – displays all roles granted to Windows NT users and groups with sp_grantlogin.

sp_logininfo ["*login_name*" | "*group_name*"]

sp_logiosize

Changes the log I/O size used by Adaptive Server to a different memory pool when doing I/O for the transaction log of the current database.

sp_logiosize ["default" | "*size*" | "all"]

sp_maplogin

Maps external users to Adaptive Server logins.

sp_maplogin (*authentication_mech* | null), (*client_username* | null),
(*action* | *login_name* | null)

sp_metrics

Backs up, drops, and flushes QP metrics—always captured in the default running group, which is group 1 in each respective database—and their statistics on queries.

sp_metrics ['backup' *backup_group_ID* | 'drop', '*gid*' [, '*id*'] |
'flush' | 'help', '*command*']

sp_modify_resource_limit

Changes a resource limit by specifying a new limit value, or the action to take when the limit is exceeded, or both.

sp_modify_resource_limit {*name*, *appname* }
rangename , *limittype* , *limitvalue* , *enforced* , *action* , *scope*

sp_modify_time_range

Changes the start day, start time, end day, and/or end time associated with a named time range.

sp_modify_time_range *name, startday*, *endday*, *starttime*, *endtime*

sp_modifylogin

Modifies the default database, default language, default role activation, login script, full name, the password expiration interval, the minimum password length, and the maximum number of failed logins allowed for a specified Adaptive Server login account.

sp_modifylogin {*loginame* | "all overrides"}, *option*, *value*

sp_modifystats

Allows the System Administrator to modify the density values of a column—or columns—in sysstatistics

sp_modifystats [*database*].[*owner*].*table_name*,
{"*column_group*" | "all"},
MODIFY_DENSITY,
{range | total},
{absolute | factor},
"*value*"

sp_modifystats [*database*].[*owner*].*table_name*,
column_name,
REMOVE_SKEW_FROM_DENSITY

sp_modifythreshold

Modifies a threshold by associating it with a different threshold procedure, free-space level, or segment name.

sp_modifythreshold *dbname*, *segname*, *free_space*
[, *new_proc_name*] [, *new_free_space*] [, *new_segname*]

sp_monitor

Displays statistics about Adaptive Server.

sp_monitor [connection, [cpu | diskio | elapsed time]]
[event, [*spid*]] [*procedure*, [*dbname*, [*procname*,
[, summary | detail]]]] [enable] [disable]
[statement, [cpu | diskio | elapsed time]] [help],
[connection | statement | procedure | event]]

sp_monitorconfig

Displays cache usage statistics regarding metadata descriptors for indexes, objects, and databases, and reports statistics on auxiliary scan descriptors used for referential integrity queries and usage statistics for transaction descriptors and DTX participants.

sp_monitorconfig "*configname*" [, "*result_tbl_name*"] [, "full"]

sp_object_stats

Shows lock contention, lock wait-time, and deadlock statistics for tables and indexes.

sp_object_stats *interval* [, *top_n*[, *dbname*, *objname* [, *rpt_option*]]]

sp_passthru

Component Integration Services only – allows the user to pass a SQL command buffer to a remote server.

sp_passthru *server*, *command*, *errcode*, *errmsg*, *rowcount*
[, *arg1*, *arg2*, ... *argn*]

sp_password

Adds or changes a password for an Adaptive Server login account.

sp_password *caller_passwd*, *new_passwd* [, *loginame*, *immediate*]

sp_placeobject

Puts future space allocations for a table or index on a particular segment.

sp_placeobject *segname*, *objname*

sp_plan_dbccdb

Recommends suitable sizes for new dbccdb and dbccalt databases, lists suitable devices for dbccdb and dbccalt, and suggests a cache size and a suitable number of worker processes for the target database.

sp_plan_dbccdb [*dbname*]

sp_poolconfig

Creates, drops, resizes, and provides information about memory pools within data caches.

sp_poolconfig *cache_name* [, "*mem_size* [P|K|M|G]", "*config_pool*K"
[, "*affected_pool*K"]]

sp_poolconfig *cache_name*, "*io_size* ", "wash=*size*[P|K|M|G]"

sp_poolconfig *cache_name*, "*io_size* ",
"local async prefetch limit=*percent* "

sp_post_xpload

Checks and rebuilds indexes after a cross-platform load database where the endian types are different.

sp_post_xpload

sp_primarykey

Defines a primary key on a table or view.

sp_primarykey *tabname*, *col1* [, *col2*, *col3*, ..., *col8*]

sp_processmail

Windows NT only – reads, processes, sends, and deletes messages in the Adaptive Server message inbox, using the sp_findnextmsg, xp_readmail, xp_sendmail, and xp_deletemail system extended stored procedures (ESPs).

sp_processmail [*subject*] [, *originator* [, *dbuser*
[, *dbname* [, *filetype* [, *separator*]]]]]

sp_procxmode

Displays or changes the execution modes associated with stored procedures.

sp_procxmode [*procname* [, *tranmode*]]

sp_recompile

Causes each stored procedure and trigger that uses the named table to be recompiled the next time it runs.

sp_recompile *objname*

sp_remap

Remaps a stored procedure, trigger, rule, default, or view from releases later than 4.8 and prior to 10.0 to be compatible with releases 10.0 and later. Use sp_remap on pre-existing objects that the upgrade procedure failed to remap.

sp_remap *objname*

sp_remoteoption

Displays or changes remote login options.

sp_remoteoption [*remoteserver* [, *loginame*
[, *remotename* [, *optname* [, *optvalue*]]]]]

sp_remotesql

Component Integration Services only – establishes a connection to a remote server, passes a query buffer to the remote server from the client, and relays the results back to the client.

sp_remotesql *server*, *query*[, *query2*, ... , *query254*]

sp_rename

Changes the name of a user-created object or user-defined datatype in the current database.

sp_rename *objname*, *newname* [,"*index*" | "*column*"]

sp_rename_qpgroup

Renames an abstract plan group.

sp_rename_qpgroup *old_name*, *new_name*

sp_renamedb

Changes the name of a user database.

sp_renamedb *dbname*, *newname*

sp_reportstats

Reports statistics on system usage.

sp_reportstats [*loginame*]

sp_revokelogin

Windows NT only – revokes Adaptive Server roles and default permissions from Windows NT users and groups when Integrated Security mode or Mixed mode (with Named Pipes) is active.

sp_revokelogin {*login_name* | *group_name*}

sp_role

Grants or revokes roles to an Adaptive Server login account.

sp_role {"grant" | "revoke"}, *rolename*, *loginame*

sp_sendmsg

Sends a message to a User Datagram Protocol (UDP) port.

sp_sendmsg *ip_address*, *port_number*, *message*

sp_serveroption

Displays or changes remote server options.

sp_serveroption [*server*, *optname*, *optvalue*]

sp_set_qplan

Changes the text of the abstract plan of an existing plan without changing the associated query.

sp_set_qplan *id*, *plan*

sp_setlangalias

Assigns or changes the alias for an alternate language.

sp_setlangalias *language*, *alias*

sp_setpglockpromote

Sets or changes the lock promotion thresholds for a database, for a table, or for Adaptive Server.

sp_setpglockpromote {"database" | "table"}, *objname*, *new_lwm*, *new_hwm*, *new_pct*

sp_setpglockpromote server, NULL, *new_lwm*, *new_hwm*, *new_pct*

sp_setpsexe

Sets custom execution attributes for a session while the session is active.

sp_setpsexe *spid*, *exeattr*, *value*

sp_setrowlockpromote

Sets or changes row-lock promotion thresholds for a datarows-locked table, for all datarows-locked tables in a database, or for all datarows-locked tables on a server.

sp_setrowlockpromote "server", NULL, *new_lwm*, *new_hwm*, *new_pct*

sp_setrowlockpromote {"database" | "table"}, *objname*, *new_lwm*, *new_hwm*, *new_pct*

sp_setsuspect_granularity

Displays or sets the recovery fault isolation mode for a user database, which governs how recovery behaves when it detects data corruption.

sp_setsuspect_granularity [*dbname* [, "database" | "page" [, "read_only"]]]

sp_setsuspect_threshold

Displays or sets the maximum number of suspect pages that Adaptive Server allows in a database before marking the entire database suspect.

sp_setsuspect_threshold [*dbname* [, *threshold*]]

sp_showcontrolinfo

Displays information about engine group assignments, bound client applications, logins, and stored procedures.

sp_showcontrolinfo [*object_type*, *object_name*, *spid*]

sp_showexeclass

Displays the execution class attributes and the engines in any engine group associated with the specified execution class.

sp_showexeclass [*execlassname*]

sp_showplan

Displays the showplan output for any user connection for the current SQL statement or for a previous statement in the same batch.

sp_showplan *spid*, *batch_id* output, *context_id* output, *stmt_num* output

sp_showpsexe

Displays execution class, current priority, and affinity for all client sessions running on Adaptive Server.

sp_showpsexe [*spid*]

sp_spaceused

Displays estimates of the number of rows, the number of data pages, the size of indexes, and the space used by a specified table or by all tables in the current database.

sp_spaceused [*objname* [,1]]

sp_ssladmin

Adds, deletes, or displays a list of server certificates for Adaptive Server.

```
sp_ssladmin {[addcert, certificate_path [, password|NULL]]
    [dropcert, certificate_path]
    [lscert]
    [help]}
    [lsciphers]
    [setciphers, {"FIPS" | "Strong" | "Weak" | "All"
        | quoted_list_of_ciphersuites}]
```

sp_syntax

Displays the syntax of Transact-SQL statements, system procedures, utilities, and other routines for Adaptive Server, depending on which products and corresponding sp_syntax scripts exist on your server.

sp_syntax *word* [, *mod*][, *language*]

sp_sysmon

Displays performance information.

sp_sysmon begin_sample

sp_sysmon interval [, noclear,[,section [, applmon]]]

sp_sysmon { end_sample | *interval* } [, *section* [, *applmon*]]

sp_sysmon { end_sample | *interval* } [, *applmon*]

sp_tempdb

Allows users to create the default temporary database group, bind temporary databases to the default temporary database group, and bind users and applications to the default temporary database group or to specific temporary databases

```
sp_tempdb [
    [ { “create” | “drop” } , “groupname” ] |
    [ { “add” | “remove” } , “tempdbname”, “groupname” ] |
    [ { “bind”, ““objtype”, “objname”, “bindtype”, “bindobj”
        [, “scope”, “hardness” ] } |
        { “unbind”, “objtype”, “objname” [, “scope” ] } ] |
    [ “unbindall_db”, “tempdbname” ] |
    [ show [, "all" | "gr" | "db" | "login" | "app" [, “name” ] ] |
    [ who, “dbname” ] [ help ] ]
```

sp_thresholdaction

Executes automatically when the number of free pages on the log segment falls below the last-chance threshold, unless the threshold is associated with a different procedure. Sybase does not provide this procedure.

```
sp_thresholdaction @dbname, @segment_name,
    @space_left, @status
```

sp_transactions

Reports information about active transactions.

```
sp_tranactions ["xid", xid_value] |
    ["state", {"heuristic_commit" | "heuristic_abort"
    | "prepared" | "indoubt"} [, "xactname"]] |
    ["gtrid", gtrid_value]
```

sp_unbindcache

Unbinds a database, table, index, text object, or image object from a data cache.

```
sp_unbindcache dbname [,[owner.]tablename
    [, indexname | "text only"]]
```

sp_unbindcache_all

Unbinds all objects that are bound to a cache.

```
sp_unbindcache_all cache_name
```

sp_unbindefault

Unbinds a created default value from a column or from a user-defined datatype.

```
sp_unbindefault objname [, futureonly]
```

sp_unbindexeclass

Removes the execution class attribute previously associated with an client application, login, or stored procedure for the specified scope.

```
sp_unbindexeclass object_name, object_type, scope
```

sp_unbindmsg

Unbinds a user-defined message from a constraint.

```
sp_unbindmsg constrname
```

sp_unbindrule

Unbinds a rule from a column or from a user-defined datatype.

```
sp_unbindrule objname [, futureonly [, “accessrule” | “all”]]
```

sp_version

Returns the version information of the installation scripts that was last run and whether it was successful.

```
sp_version [script_file, [all]]
```

sp_volchanged

Notifies the Backup Server that the operator performed the requested volume handling during a dump or load.

```
sp_volchanged session_id, devname, action[, fname [, vname]]
```

sp_who

Reports information about all current Adaptive Server users and processes or about a particular user or process.

```
sp_who [loginame | "spid"]
```

Catalog stored procedures

This section provides the syntax and brief descriptions for Adaptive Server catalog stored procedures. See *Reference Manual: Procedures* for more information.

sp_column_privileges

Returns permissions information for one or more columns in a table or view.

```
sp_column_privileges table_name [, table_owner
    [, table_qualifier [, column_name]]]
```

sp_columns

Returns information about the type of data that can be stored in one or more columns.

```
sp_columns table_name [, table_owner ]
    [, table_qualifier] [, column_name]
```

sp_databases

Returns a list of databases in Adaptive Server.

```
sp_databases
```

sp_datatype_info

Returns information about a particular ODBC datatype or about all ODBC datatypes.

```
sp_datatype_info [data_type]
```

sp_fkeys

Returns information about foreign key constraints created with the create table or alter table command in the current database.

```
sp_fkeys pktable_name [, pktable_owner]
    [, pktable_qualifier] [, fktable_name]
    [, fktable_owner] [, fktable_qualifier]
```

sp_pkeys

Returns information about primary key constraints created with the create table or alter table command for a single table.

```
sp_pkeys table_name [, table_owner] [, table_qualifier]
```

sp_server_info

Returns a list of Adaptive Server attribute names and current values.

sp_server_info [*attribute_id*]

sp_special_columns

Returns the optimal set of columns that uniquely identify a row in a table or view; can also return a list of timestamp columns whose values are automatically generated when any value in the row is updated by a transaction.

sp_special_columns *table_name* [, *table_owner*]
[, *table_qualifier*] [, *col_type*]

sp_sproc_columns

Returns information about a stored procedure's input and return parameters.

sp_sproc_columns *procedure_name*[, *procedure_owner*]
[, *procedure_qualifier*] [, *column_name*]

sp_statistics

Returns a list of indexes on a single table.

sp_statistics *table_name* [, *table_owner*]
[, *table_qualifier*] [, *index_name*] [, *is_unique*]

sp_stored_procedures

Returns information about one or more stored procedures.

sp_stored_procedures [*sp_name* [, *sp_owner*[, *sp_qualifier*]]]

sp_table_privileges

Returns privilege information for all columns in a table or view.

sp_table_privileges *table_name* [, *table_owner*[, *table_qualifier*]]

sp_tables

Returns a list of objects that can appear in a from clause.

sp_tables [*table_name*][, *table_owner*][, *table_qualifier*][, *table_type*]

Extended stored procedures

This section provides the syntax and brief descriptions for Adaptive Server extended stored procedures. See *Reference Manual: Procedures* for more information.

xp_cmdshell

Executes a native operating system command on the host system running Adaptive Server.

xp_cmdshell *command* [, no_output] [return_status | no_wait]

xp_deletemail

Windows NT only – deletes a message from the Adaptive Server message inbox.

xp_deletemail [*msg_id*]

xp_enumgroups

Windows NT only – displays groups for a specified Windows NT domain.

xp_enumgroups [*domain_name*]

xp_findnextmsg

Windows NT only – retrieves the next message identifier from the Adaptive Server message inbox.

xp_findnextmsg @*msg_id* = @*msg_id* output [, *type*]
[, unread_only = {true | false}]

xp_logevent

Windows NT only – provides for logging a user-defined event in the Windows NT Event Log from within Adaptive Server.

xp_logevent *error_number*, *message* [, *type*]

xp_readmail

Windows NT only – reads a message from the Adaptive Server message inbox.

xp_readmail [*msg_id*]
[, *recipients* output]
[, *sender* output]
[, *date_received* output]
[, *subject* output]
[, *cc* output]
[, *message* output]
[, *attachments* output]
[, suppress_attach = {true | false}]
[, peek = {true | false}]
[, unread = {true | false}]
[, *msg_length* output]
[, *bytes_to_skip* [output]]
[, *type* [output]]

xp_sendmail

Windows NT only – sends a message to the specified recipients. The message is either text or the results of a Transact-SQL query.

xp_sendmail *recipient* [; *recipient*] . . .[, *subject*][, *cc_recipient*] . . .
[, *bcc_recipient*] . . .
[, {query | message}]
[, *attachname*]
[, *attach_result* = {true | false}]
[, *echo_error* = {true | false}]
[, *include_file* [, include_file] . . .]
[, *no_column_header* = {true | false}]
[, *no_output* = {true | false}]
[, *width*][, *separator*][, *dbuser*][, *dbname*][, *type*]
[, *include_query* = {true | false}]

xp_startmail

Windows NT only – starts an Adaptive Server mail session.

xp_startmail [*mail_user*] [, *mail_password*]

xp_stopmail

Windows NT only – stops an Adaptive Server mail session.

xp_stopmail

dbcc stored procedures

This section provides the syntax and brief descriptions for Adaptive Server dbcc stored procedures. See *Reference Manual: Procedures* for more information.

sp_dbcc_alterws

Changes the size of the specified workspace to a specified value, and initializes the workspace.

sp_dbcc_alterws *dbname*, *wsname*, "*wssize*[K|M]"

sp_dbcc_configreport

Generates a report that describes the configuration information used by the dbcc checkstorage operation for the specified database.

sp_dbcc_configreport [*dbname*]

sp_dbcc_createws

Creates a workspace of the specified type and size on the specified segment and database.

sp_dbcc_createws *dbname*, *segname*, [*wsname*], *wstype*, "wssize[K|M]"

sp_dbcc_deletedb

Deletes from dbccdb all the information related to the specified target database.

sp_dbcc_deletedb [*dbname* | *dbid*]

sp_dbcc_deletehistory

Deletes the results of dbcc checkstorage operations performed on the target database before the specified date and time.

sp_dbcc_deletehistory [*cutoffdate* [, *dbname* | *dbid*]]

sp_dbcc_differentialreport

Generates a report that highlights the changes in I/O statistics and faults that took place between two dbcc operations.

sp_dbcc_differentialreport [*dbname* [, *objectname*]],
[*db_op*] [, "*date1"* [, "*date2"*]]

sp_dbcc_evaluatedb

Recomputes configuration information for the target database and compares it to the current configuration information.

sp_dbcc_evaluatedb [*dbname*]

sp_dbcc_exclusions

Allows the user to create and manage persistent exclusion lists for use by checkverify and sp_dbcc_faultreport.

sp_dbcc_exclusions *dbname*, *op*, *type*,*exclusion_list*

sp_dbcc_faultreport

Generates a report covering fault statistics for the dbcc checkstorage operations performed for the specified object in the target database on the specified date.

sp_dbcc_faultreport [*report_type* [, *dbname*
[, *objectname*[, *date* [, *hard_only*

[, *exclusion_mode*[, *exclusion_faults*
[, *exclusion_tables* [, *exclusion_combo*
[, *display_recommendations*
[, *opid* [,*fault_type_in*]]]]]]]]]]]

sp_dbcc_fullreport

Runs sp_dbcc_summaryreport, sp_dbcc_configreport, sp_dbcc_statisticsreport, and sp_dbcc_faultreport short for *database..object_name* on or before the specified *date*.

sp_dbcc_fullreport [*dbname* [, *objectname* [, *date*]]]

sp_dbcc_help_fault

Provides a description of the specified fault type and the recommended fix.

sp_dbcc_help_fault [*fault_type*]

sp_dbcc_patch_finishtime

Facilitates reporting on aborted checkverify and checkstorage operations.

sp_dbcc_patch_finishtime *dbname*, *opid* [,*optype*
[,*seq* [,*finishtime*]]]

sp_dbcc_recommendations

Analyzes faults reported by the checkstorage operation corresponding to the specified operation ID, or date, and generates a list of recommended corrective actions for the specified object in the target database.

sp_dbcc_recommendations dbname [,"*date*"[, *opid* [, "*objectname*"]]]

sp_dbcc_runcheck

Runs dbcc checkstorage on the specified database, then runs sp_dbcc_summaryreport or a report you specify.

sp_dbcc_runcheck *dbname* [, *user_proc*]

sp_dbcc_statisticsreport

Generates an allocation statistics report on the specified object in the target database.

sp_dbcc_statisticsreport [*dbname* [, *objectname* [, *date*]]]

sp_dbcc_summaryreport

Generates a summary report on the specified database.

sp_dbcc_summaryreport [dbname [, date [, op_name [,
display_recommendations]]]]

sp_dbcc_updateconfig

Updates the dbcc_config table in dbccdb with the configuration information of the target database.

sp_dbcc_updateconfig *dbname*, *type*, "*str1*" [, "*str2*"]

System tables

This section provides very brief descriptions for Adaptive Server system tables. See *Reference Manual: Tables* for more a full description.

syblicenseslog

master database only – contains one row for each update of the maximum number of licenses used in Adaptive Server per 24-hour period.

sysalternates

All databases – contains one row for each Adaptive Server user that is mapped or aliased to a user of the current database.

sysattributes

All databases – defines properties of objects such as databases, tables, indexes, users, logins, and procedures.

sysauditoptions

sybsecurity database – contains one row for each server-wide audit option and indicates the current setting for that option.

sysaudits_01 – sysaudits_08

sybsecurity database – These system tables contain the audit trail.

syscharsets

master database only – contains one row for each character set and sort order defined for use by Adaptive Server.

syscolumns

All databases – contains one row for every column in every table and view, and a row for each parameter in a procedure.

syscomments

All databases – contains entries for each view, rule, default, trigger, table constraint, partition, procedure, computed column, function-based index key, and other forms of compiled objects.

sysconfigures

master database only – contains one row for each configuration parameter that can be set by the user.

sysconstraints

All databases – Whenever a user declares a new check constraint or referential constraint using create table or alter table, Adaptive Server inserts a row into the sysconstraints table.

syscoordinations

sybsystemdb database only – contains information about remote Adaptive Servers participating in distributed transactions (remote participants) and their coordination states.

syscurconfigs

master database only – is built dynamically when queried. It contains an entry for each of the configuration parameters, as does sysconfigures, but with the current values rather than the default values. In addition, it contains four rows that describe the configuration structure.

sysdatabases

master database only – contains one row for each database in Adaptive Server.

sysdepends

All databases – contains one row for each procedure, view, or table that is referenced by a procedure, view, or trigger.

sysdevices

master database only – contains one row for each tape dump device, disk dump device, disk for databases, and disk partition for databases.

sysencryptkeys

Reserved for future use.

sysengines

master database only – contains one row for each Adaptive Server engine currently online.

sysgams

All databases – stores the global allocation map (GAM) for the database.

sysindexes

All databases – contains one row for each clustered index, one row for each nonclustered index, one row for each table that has no clustered index, and one row for each table that contains text or image columns.This table also contains one row for each function-based index or index created on a computed column.

sysjars

All databases – contains one row for each Java archive (JAR) file that is retained in the database.

syskeys

All databases – contains one row for each primary, foreign, or common key.

syslanguages

master database only – contains one row for each language known to Adaptive Server. us_english is not in syslanguages, but it is always available to Adaptive Server.

syslisteners

master database only – contains a row for each network protocol available for connecting with the current Adaptive Server. Adaptive Server builds syslisteners dynamically when a user or client application queries the table.

syslocks

master database only – contains information about active locks, and built dynamically when queried by a user. No updates to syslocks are allowed.

sysloginroles

master database only – contains a row for each instance of a server login possessing a system role.

syslogins

master database only – contains one row for each valid Adaptive Server user account.

syslogs

All databases – contains the transaction log. It is used by Adaptive Server for recovery and roll forward. It is not useful to users.

syslogshold

master database only – contains information about each database's oldest active transaction (if any) and the Replication Server truncation point (if any) for the transaction log, but it is not a normal table. Rather, it is built dynamically when queried by a user. No updates to syslogshold are allowed.

sysmessages

master database only – contains one row for each system error or warning that can be returned by Adaptive Server. Adaptive Server displays the error description on the user's screen.

sysmonitors

master database only – contains one row for each monitor counter.

sysobjects

All databases – contains one row for each table, view, stored procedure, extended stored procedure, log, rule, default, trigger, check constraint, referential constraint, computed column, function-based index key, and (in tempdb only) temporary object, and other forms of compiled objects. It also contains one row for each partition condition ID when object type is N.

syspartitionkeys

All databases – contains one row for each partition key for hash, range, and list partitioning of a table. All columns are not null.

syspartitions

All databases – contains one row for each data partition and one row for each index partition.

sysprocedures

All databases – contains entries for each view, default, rule, trigger, procedure, declarative default, partition condition, check constraint, comuted column, function-based index key, and other forms of compiled objects.

sysprocesses

master database only – contains information about Adaptive Server processes, but it is not a normal table. It is built dynamically when queried by a user. No updates to sysprocesses are allowed. Use the kill statement to kill a process.

sysprotects

All databases – contains information on permissions that have been granted to, or revoked from, users, groups, and roles.

sysquerymetrics

All databases – presents aggregated historical query processing metrics for individual queries from persistent data. In addition to monitoring tables, use performance metrics information from this catalog.

sysqueryplans

All databases – contains two or more rows for each abstract query plan. Uses datarow locking.

sysreferences

All databases – contains one row for each referential integrity constraint declared on a table or column.

sysremotelogins

master database only – contains one row for each remote user that is allowed to execute remote procedure calls on this Adaptive Server.

sysresourcelimits

master database only – contains a row for each resource limit defined by Adaptive Server.

sysroles

All databases – maps server role IDs to local role IDs.

syssecmechs

master database only – contains information about the security services supported by each security mechanism that is available to Adaptive Server. syssecmechs is not created during installation, rather, it is built dynamically when queried by a user.

syssegments

All databases – contains one row for each segment (named collection of disk pieces).

sysservers

master database only – contains one row for each remote Adaptive Server, Backup Server™, or Open Server™ on which this Adaptive Server can execute remote procedure calls.

syssessions

master database only – is used only when Adaptive Server is configured for Sybase Failover in a high availability system. syssessions contains one row for each client that connects to Adaptive Server with the failover property.

sysslices

All databases – contains one row for each slice (page chain) of a sliced table. sysslices is used only during the Adaptive Server upgrade process. After the upgrade is complete, all the data is removed.

syssrvroles

master database only – contains a row for each system or user-defined role.

sysstatistics

All databases – contains one or more rows for each indexed column on a user table and for each partition. May also contain rows for unindexed column.

systabstats

All databases – contains one row for each clustered index, one row for each nonclustered index, one row for each table that has no clustered index, and one row for each partition.

systhresholds

All databases – contains one row for each threshold defined for the database.

systimeranges

master database only – stores named time ranges, which are used by Adaptive Server to control when a resource limit is active.

systransactions

master database only – contains information about Adaptive Server transactions, but it is not a normal table. Portions of the table are built dynamically when queried by a user, while other portions are stored in the master database. Updates to the dynamically-built columns of systransactions are not allowed.

systypes

All databases – contains one row for each system-supplied and user-defined datatype. Domains (defined by rules) and defaults are given, if they exist.

sysusages

master database only – sysusages contains one row for each disk allocation piece assigned to a database. Each database contains a specified number of database (logical) page numbers.

sysusermessages

All databases – contains one row for each user-defined message that can be returned by Adaptive Server.

sysusers

All databases – contains one row for each user allowed in the database, and one row for each group or role.

sysxtypes

All databases – contains one row for each extended, Java-SQL datatype.

DBCC tables

This section provides brief descriptions for Adaptive Server dbcc tables. See *Reference Manual: Tables* for full descriptions.

dbcc_config

Describes the currently executing or last completed dbcc checkstorage operation.

dbcc_counters

Stores the results of the analysis performed by dbcc checkstorage.

dbcc_exclusions

Stores the faults, tables or a combination of them that should be excluded from processing by checkverify and fault reporting via sp_dbcc_faultreport.

dbcc_fault_params

Provides additional descriptive information for a fault entered in the dbcc_faults table.

dbcc_faults

Provides a description of each fault detected by dbcc checkstorage.

dbcc_operation_log

Records the use of the dbcc checkstorage operations.

dbcc_operation_results

Provides additional descriptive information for an operation recorded in the dbcc_operation_log table.

dbcc_types

Provides the definitions of the datatypes used by dbcc checkstorage.

Utilities

This section provides the syntax and brief descriptions for Adaptive Server utilities. See *Utility Guide* for more information.

backupserver

The executable form of the Backup Server program.

```
backupserver
    [-C server_connections]
    [-S b_servername]
    [-I interfaces_file]
    [-e error_log_file]
    [-M sybmultbuf_binary]
    [-N network_connections]
    [-T trace_value]
    [-L Sybase_language_name]
    [-J Sybase_character_set_name]
    [-c tape_config_file]
    [-D n]
    [-A pathname]
    [-P active_service_threads]
    [-V level_number]
    [-p n]
    [-m max_shared_memory]
backupserver -v
```

bcp

Copies a database table to or from an operating system file in a user-specified format.

```
bcp
[[database_name.]owner.]table_name [:
    [ partition_id | slice_number ] |
    partition partition_name] {in | out} datafile
    [--show-fi]
    [--hide-vcc]
    [-f formatfile]
    [-e errfile]
    [-F firstrow]
    [-L lastrow]
    [-b batchsize]
    [-m maxerrors]
    [-n] [-c]
    [-t field_terminator]
    [-r row_terminator]
    [-U username]
    [-P password]
    [-I interfaces_file]
    [-S server]
    [-a display_charset]
    [-z language]
    [-A packet_size]
    [-J client_charset]
    [-T text_or_image_size]
    [-E]
    [-g id_start_value]
    [-N] [-X]
    [-K keytab_file]
    [-R remote_server_principal]
    [-V [security_options]]
    [-Z security_mechanism]
    [-Q] [-Y]
    [--maxconn maximum_connections
bcp -v
```

buildmaster

Adaptive Server no longer uses buildmaster.

certauth

Converts a server certificate request to a CA- (certificate authority) signed certificate.

```
certauth
    [-r]
    [-C caCert_file]
    [-Q request_filename]
    [-K caKey_filename]
    [-N serial_number
    [-O SignedCert_filename]
    [-P caPassword]
    [-s start_time]
    [-T valid_time]
certauth -v
```

certpk12

Export or import a PKCS #12 file into a certificates file and a private key.

certpk12
{-O *Pkcs12_file* | -I *Pkcs12_file*}
[-C *Cert_file*]
[-K *Key_file*]
[-P *key_password*]
[-E *Pkcs12_password*]

certpk12 -v

certreq

Creates a server certificate request and corresponding private key.

certreq
[-F *input_file*]
[-R *request_filename*]
[-K *PK_filename*]
[-P *password*]

certreq -v

charset

UNIX platforms only – loads the character sets and sort order files in Adaptive Server.

charset
[-P*password*]
[-S*server*]
[-I*interface*]
sort_order
[*charset*]

charset -v

cobpre

Precompiler for COBOL. See Appendix A of the *Open Client and Open Server Programmer's Supplement*.

cpre

Precompiler for C. See Appendix A of the *Open Client and Open Server Programmer's Supplement*.

dataserver

UNIX platforms only – the executable form of the Adaptive Server program.

dataserver [-f] [-g] [-G] [-h] [-H] [-m] [-q] [-v] [-X]
[-a *path_to_CAPs_directive_file*]
-b *master_device_size*
[k | K | m | M | g | G | t | T]
[-c *config_file_for_server*]
[-d *device_name*]
[-e *path_to_error_log*]
[-i *interfaces_file_directory*]
[-K *keytab_file*]
[-L *config_file_name_for_connectivity*]
[-M *shared_memory_repository_directory*]
[-N *licinstant*
[-n *sa_login_name*]
[-p *sa_login_name*]
[-r *mirror_disk_name*]

```
    [-s server_name]
    [-T trace_flag]
    [-u sa/sso_name]
    [-w master | model database]
    [-y [password] ]
    [-z page_size [ k | K ] ]
  dataserver -v
```

ddlgen

A Java-based tool that generates definitions for server- and database-level objects in Adaptive Server.

```
  ddlgen
    -Ulogin
    -Ppassword
    -S[server | host_name : port_number]
    [-I interfaces_file]
    [-Tobject_type]
    [-Nobject_name]
    [-Ddbname]
    [-Xextended_object_type]
    [-Ooutput_file]
    [-Eerror_file]
    [-Lprogress_log_file]
    [-Jclient_charset]
    -F[ % | SGM | GRP | USR | R | D | UDD | U | V |
        P | XP | I | RI | KC | TR | PC ]
  ddlgen -v
```

defncopy

Copies definitions for specified views, rules, defaults, triggers, or procedures from a database to an operating-system file or from an operating-system file to a database.

```
  defncopy
    [-X]
    [-a display_charset]
    [-I interfaces_file]
    [-J [client_charset]]
    [-K keytab_file]
    [-P password]
    [-R remote_server_principal]
    [-S [server_name]]
    [-U username]
    [-V security_options]
    [-Z security_mechanism]
    [-z language]
    { in file_name database_name |
        out file_name database_name
        [owner.]object_name
         [[owner.]object_name...] }
  defncopy -v
```

dscp

UNIX platforms only – allows you to view and edit server entries in the interfaces file from the command line in UNIX platforms.

```
dscp [-p]
dscp -v
```

dsedit

UNIX platforms– the dsedit utility allows you to view and edit server entries in the interfaces file using a GUI based on X11/Motif in UNIX platforms.

```
dsedit
dsedit -v
```

extractjava (extrjava in Windows)

Copies a retained JAR and the classes it contains from an Adaptive Server into a client file.

```
extractjava
   -j jar_name
   -f file_name
   [-S server_name]
   [-U user_name]
   [-P password]
   [-D database_name]
   [-I interfaces_file]
   [-a display_charset]
   [-J client_charset]
   [-z language]
   [-t timeout]
   [-v]
extractjava -v
```

installjava

Installs a JAR from a client file into an Adaptive Server.

```
installjava
   -f file_name
   [ -new | -update ]
   [ -j jar_name ]
   [ -S server_name ]
   [ -U user_name ]
   [ -P password ]
   [ -D database_name ]
   [ -I interfaces_file ]
   [ -a display_charset ]
   [ -J client_charset ]
   [ -z language ]
   [ -t timeout ]
   [-v]
installjava -v
```

isql

Interactive SQL parser to Adaptive Server.

```
isql [-b] [-e] [-F] [-p] [-n] [-v] [-X] [-Y] [-Q]
   [-a display_charset]
   [-A packet_size]
   [-c cmdend]
   [-D database]
```

[-E *editor*]
[-h *header*]
[-H *hostname*]
[-i *inputfile*]
[-I *interfaces_file*]
[-J *client_charset*]
[-K *keytab_file*]
[-l *login_timeout*]
[-m *errorlevel*]
[-o *outputfile*]
[-P *password*]
[-R *remote_server_principal*]
[-s *colseparator*]
[-S *server_name*]
[-t *timeout*]
-U *username*
[-V [*security_options*]]
[-w *columnwidth*]
[-z *locale_name*]
[-Z *security_mechanism*]

langinstall

Installs a new language in an Adaptive Server.

langinstall
[-S *server*]
[-U *user*]
[-P *password*]
[-R *release_number*]
[-I *path*]
language
character_set

langinstall -v

optdiag

Displays optimizer statistics or loads updated statistics into system tables.

optdiag [binary] [simulate] statistics
{ -i *input_file* |
database[.*owner*[.[*table*[.*column*]]]]
[-o *output_file*] }
[-U *user_name*]
[-P *password*]
[-T *trace_value*]
[-I *interfaces_file*]
[-S *server*]
[-v] [-h] [-s]
[-z *language*]
[-J *client_character_set*]
[-a *display_charset*]

preupgrade

Performs tests on an installation or database to determine its readiness for upgrade, and reports found problems.

```
preupgrade [-v] [-h] [-N]
   [-D database_name]
   [-I interfaces_file]
   [-P password]
   [-S server_name]
   [-U user_name]
   [-X option[,option]...]
```

pwdcrypt

Creates and prints an encrypted LDAP password in the *libtcl.cfg* file.

```
pwdcrypt
```

showserver

UNIX platforms only – shows the Adaptive Servers and Backup Servers that are currently running on the local machine, available only in UNIX platforms.

```
showserver
```

sqldbgr

sqldbgr is a command-line utility that debugs stored procedures and triggers.

```
sqldbgr
   -U username
   -P password
   -S host:port
```

sqlloc

UNIX platforms only – installs and modifies languages, character sets, and sort order defaults for Adaptive Server using a GUI based on X11/Motif.

```
sqlloc
   [-S Server]
   [-U User]
   [-P Password]
   [-s Sybase Dir]
   [-l Interfaces file]
   [-r Resource file]
sqlloc -v
```

sqllocres

UNIX platforms only– installs and modifies languages, character sets, and sort order defaults for Adaptive Server, using a resource file.

```
sqllocres
   [-S Server]
   [-U User]
   [-P Password]
   [-s Sybase Dir]
   [-l Interfaces file]
   [-r Resource file]
sqllocres -v
```

sqlsrvr

Windows platforms only – is the executable form of the Adaptive Server program.

```
sqlserver [-f] [-g] [-G] [-h] [-H] [-m] [-P] [-q] [-v] [-X]
   [-a path_to_CAPs_directive_file]
```

[-b *master_device_size*]
[k | K | m | M | g | G | t | T]
[-c *config_file_for_server*]
[-d *device_name*]
[-e *path_to_error_log*]
[-i *interfaces_file_directory*]
[-K *keytab_file*]
[-L *config_file_name_for_connectivity*]
[-M *shared_memory_repository_directory*]
[-p *sa_login_name*]
[-r *mirror_disk_name*]
[-s *server_name*]
[-T *trace_flag*]
[-u *sa/sso_name*]
[-w *master* | *model database*]
[-y [*password*]]
[-z *page_size* [k | K]]

sqlupgrade

UNIX platforms only – upgrades your currently installed version of Adaptive Server to the newest release using a GUI based on X11/Motif.

sqlupgrade
[-s *Sybase Dir*]
[-r *Resource File*]

sqlupgrade -v

sqlupgraderes

UNIX platforms only – upgrades your currently installed release of Adaptive Server to the newest release using resource files.

sqlupgraderes
[-s *Sybase Dir*]
[-r *Resource File*]

sqlupgraderes -v

srvbuild

UNIX platforms only – creates a new Adaptive Server, Backup Server, Monitor Server, or XP Server with default or user-specified values for key configuration attributes.

srvbuild
[-s *sybase_dir*]
[-I *interfaces_file*]
[-r *resource_file*]

srvbuild -v

srvbuildres

UNIX platforms only – creates, using resource files, a new Adaptive Server, Backup Server, Monitor Server, or XP Server with default or user-specified values for key configuration attributes.

srvbuildres
[-s *sybase_dir*]
[-I *interfaces_file*]
[-r *resource_file*]

srvbuildres -v

startserver

UNIX platforms only – starts an Adaptive Server or a Backup Server.

startserver [[-f *runserverfile*] [-m]] ...

sybmigrate

Allows you to convert an Adaptive Server from one page size to another page size, and migrate between platforms.

sybmigrate [-v] [-h] [-f]
[-D 1 | 2 | 3 | 4]
[-I *interfaces file*]
[-r *input resource file*]
[-m setup | migrate | validate | report]
[-rn status | space_est | repl | diff | password]
[-l *log file*]
[-t *output template resource file*]
[-J *client_charset*]
[-z *language*]
[-T *trace_flags*]
[-Tase *trace flags*]
[-f]

xpserver

Starts XP Server manually.

xpserver
-S*XP_Server*
[-I*interfaces_file*]
[-p*priority*]
[-s*stack_size*]
[-u] [-v] [-x]