Connecting to Your Database

VERSION 4.0

December 1994

Contents

About This Manual xiii

1 **Understanding Data Connections** 1
- How to find the information you need 2
- Types of data sources and databases you can access 3
- ODBC data sources 4
  - Watcom SQL ODBC data sources 4
  - Other ODBC data sources 6
  - Using other ODBC drivers 7
- Powersoft database interfaces 8
- Using database profiles 9
- Summary 10
- What to do next 11

2 **Using ODBC Data Sources** 13
- About the Powersoft ODBC interface 15
  - What is ODBC? 15
  - Components of an ODBC connection 16
  - Types of ODBC drivers 18
  - Ensuring the proper ODBC driver conformance levels 19
  - Obtaining ODBC drivers 21
  - Using ODBC drivers with PowerBuilder Desktop 21
- About preparing ODBC data sources 23
- About defining ODBC data sources 24
  - When you define the data source 24
  - How Powersoft products access the data source 25
  - About defining multiple data sources for the same data 28
  - Inheriting ODBC data sources 29
  - Getting help 30
  - Completing the ODBC setup dialog box 32
- What to do next 39

Microsoft Access 40
Supported versions 40
Supported SQL operations 40
Basic software components 41
Preparing to use the data source 42
Defining the data source 42
What to do next 44
INTERSOLV Btrieve 45
Supported versions 45
Supported SQL operations 45
Basic software components 46
Preparing to use the data source 47
Defining the data source 48
What to do next 54
Microsoft Btrieve 55
Supported versions 55
Supported SQL operations 55
Basic software components 56
Preparing to use the data source 57
Defining the data source 59
What to do next 60
INTERSOLV dBASE 61
Supported versions 61
Supported SQL operations 61
Basic software components 62
Preparing to use the data source 63
Defining the data source 64
What to do next 68
Microsoft dBASE 69
Supported versions 69
Supported SQL operations 69
Basic software components 70
Preparing to use the data source 71
Defining the data source 72
What to do next 75
Microsoft Excel 76
Supported versions 76
Supported SQL operations 76
Basic software components 77
Preparing to use the data source 78
Defining the data source 80
What to do next 81

Microsoft FoxPro 82
Supported versions 82
Supported SQL operations 82
Basic software components 83
Preparing to use the data source 84
Defining the data source 85
What to do next 88
INTERSOLV NetWare SQL 89
Supported versions 89
Supported SQL operations 89
Basic software components 90
Preparing to use the data source 91
Defining the data source 92
What to do next 93
INTERSOLV Paradox 4 94
Supported versions 94
Supported SQL operations 94
Basic software components 95
Preparing to use the data source 96
Defining the data source 98
About creating an index for Paradox tables 100
What to do next 100
INTERSOLV Paradox 5 101
Supported versions 101
Supported SQL operations 101
Basic software components 102
Preparing to use the data source 103
Defining the data source 105
About creating an index for Paradox tables 107
What to do next 107
Microsoft Paradox 108
Supported versions 108
Supported SQL operations 108
Basic software components 109
Preparing to use the data source 110
Defining the data source 110
What to do next 112
DEC Rdb 113
Supported versions 113
Supported SQL operations 113
Basic software components 114
Preparing to use the data source 115
Defining the data source 115
What to do next 117

Microsoft Text File 118
Supported versions 118
Supported SQL operations 118
Basic software components 119
Preparing to use the data source 120
Defining the data source 121
What to do next 128
Watcom SQL 129
Supported versions 129
Supported SQL operations 129
Basic software components 130
Preparing to use the data source 131
Defining the data source 131
What to do next 138

3 **Using Powersoft Database Interfaces 139**
About Powersoft database interfaces 141
What is a Powersoft database interface? 141
Components of a database interface connection 142
Using a Powersoft database interface 143
About preparing to use the database 144
About defining Powersoft database interfaces 145
Getting help 145
Creating a database profile 146
What to do next 154
ALLBASE/SQL 155
Supported data types 155
Basic software components 156
Preparing to use the database 156
Defining the database interface 157
What to do next 161
IBM DRDA databases 162
Features of the Powersoft IBM database interface 162
Supported data types 165
Supported functions 165
Preparing to use Database Manager and DB2/2 168
Preparing to use DB2/MVS 171
Preparing to use DB2/6000 175
Binding PowerBuilder or InfoMaker to your databases 178
Defining the IBM DRDA database interface 180
Troubleshooting your IBM DRDA connection 187
What to do next 187

INFORMIX 188
Supported versions 188
Supported data types 189
Basic software components 190
Preparing to use the database 192
Defining the database interface 197
Connecting to INFORMIX in a PowerBuilder script 199
What to do next 200
Micro Decisionware Database Gateway Interface for DB2 201
Supported versions 201
Supported data types 201
Basic software components 202
Preparing to use the database 203
Defining the database interface 204
ORACLE 205
Supported versions 205
Supported data types 205
Basic software components 206
Preparing to use the database 208
Defining the database interface 210
Using ORACLE 7 stored procedures as a data source 214
What to do next 217
SQL Server 218
Supported versions 218
Supported data types 219
Basic software components 221
Preparing to use the database 221
Defining the database interface 224
What to do next 225
SQLBase 226
Supported data types 226
Basic software components 227
Preparing to use the database 227
Defining the database interface 229
What to do next 230
Sybase Net-Gateway Interface for DB2 231
Supported versions 231
Supported data types 231
Basic software components 232
Preparing to use the database 233
Defining the database interface 233
What to do next 234

Sybase SQL Server System 10 235
Supported versions 235
Supported data types 236
Basic software components 237
Preparing to use the database 238
Defining the database interface 240
What to do next 241
XDB 242
Supported versions 242
Supported data types 242
Basic software components 243
Preparing to use the database 244
Defining the database interface 245
What to do next 246
Creating Powersoft system tables in DB2 databases 247
Creating the repository 247
Using the DB2SYSPB.SQL script 248
Installing Powersoft stored procedures in SQL Server databases 250
What are the Powersoft stored procedure scripts? 250
When to run the scripts 253
How to run the scripts 254

4 **Managing Database Connections 261**
About database connections 263
When database connections occur 263
Using database profiles 264
Creating the Powersoft repository 266
Logging on to your database for the first time 266
Displaying the repository 267
Contents of the repository 268
Controlling catalog access 268
Connecting to a database 271
Selecting a database profile 271
Responding to prompts 273
What happens when you connect 277
Maintaining ODBC data source definitions 278
Editing an ODBC data source definition 278
Deleting an ODBC data source definition 283
Maintaining database profiles 286
Editing a database profile 286
Deleting a database profile 289
Creating a profile for an existing ODBC data source 290

Sharing database profiles 294
Editing the PB.INI or IM.INI file 294
Using the Preferences painter 295
What happens when you define shared profiles 297
Maintaining shared database profiles 298

5 **Setting Additional Connection Parameters 301**
Basic steps for setting connection parameters 302
Setting DBParm parameters 303
Editing a database profile 303
Setting DBParm parameters in a PowerBuilder script 307
DBParm parameters and supported DBMSs 309
Descriptions of DBParm parameters 312
AppName 312
Async 314
Block (ORACLE) 315
Block (Sybase System 10) 316
CharSet 317
ConnectString 318
ConversionTable 320
CursorLib 322
CursorLock (ODBC) 322
CursorLock (SQL Server) 323
CursorScroll (ODBC) 326
CursorScroll (SQL Server) 327
CursorUpdate 329
Date 330
DateTime 332
DBAdm 333
DBGetTime 334
DBTextLimit 335
DelimitIdentifier 336
DisableBind 338
GroupID 340
Host 341
HPConnect 342
INET_DBPATH 343
INET_PROTOCOL 344
INET_SERVICE 345
Language 346
Log 347
LoginTimeOut 348
MixedCase 348
MsgTerse 349

PBCatalogOwner 351
PBDBMS 353
Recovery 354
Release (SQL Server) 355
Release (XDB) 356
Request 357
Scroll 358
SQLCache 359
SQLQualifiers 362
SystemOwner 363
TableCriteria (IBM DRDA, Micro Decisionware Gateway, XDB) 365
TableCriteria (ODBC) 366
TableCriteria (ORACLE) 368
TableCriteria (Sybase Net-Gateway) 370
Time 372
Setting database preferences 374
Editing the PB.INI or IM.INI file 374
Using the Preferences painter 376
Database preferences and supported DBMSs 379
Descriptions of database preferences 381
AutoCommit 381
Lock 383
NoCatalog 386
ReadOnly 388
StoredReqOwner 389
TableSpace 390

6 **Troubleshooting Your Connection 393**
Overview of troubleshooting tools 394
About the Database Trace tool 395
How you can use the Database Trace tool 395
Contents of the Database Trace log 396
Format of the Database Trace log 397
Starting the Database Trace tool 398
Starting Database Trace by editing a database profile 398
Starting Database Trace in a PowerBuilder script 401
Stopping the Database Trace tool 404
Using the Database Trace log 405
Viewing the log 405
Annotating the log 405
Deleting or clearing the log 406
Sample Database Trace output 407

Using the ODBC Driver Manager Trace 409
Starting the trace 410
Stopping the trace 411
Viewing the log 412
Sample trace output 412
Troubleshooting your IBM DRDA connection 414

A **Supported Data Sources and Databases 415**
ODBC data sources 416
Using other ODBC drivers 416
Powersoft database interfaces 418

B **Adding Functions to the PBODB040.INI File 419**
About the PBODB040.INI file 420
Adding functions to the PBODB040.INI file 421
Adding functions to an existing section 421
Adding functions to a new section 423

About This Manual

Subject

This manual describes how to connect to a database from PowerBuilder or InfoMaker by using an ODBC data source driver or Powersoft database interface. It gives procedures for preparing, defining, establishing, maintaining, and troubleshooting your database connection.

Audience

This manual is for anyone who uses PowerBuilder or InfoMaker to connect to a database. It assumes that you are familiar with the database you are using and have installed the server and client software required to access the data.

CHAPTER 1

Understanding Data Connections

About this chapter

This chapter provides an overview of the procedure for connecting to a data source or database from PowerBuilder or InfoMaker. It also describes the types of data sources and databases that you can access from these products.

Contents

Topic	Page
How to find the information you need	2
Types of data sources and databases you can access	3
ODBC data sources	4
Powersoft database interfaces	8
Using database profiles	9
Summary	10
What to do next	11

How to find the information you need

This manual describes how to connect to your data source or database in PowerBuilder or InfoMaker. The following table gives an overview of this procedure and tells where you can find more detailed information about each step.

Step	Details	See
1 Prepare to use the data source or database	Outside PowerBuilder or InfoMaker, install the required database server and client software and prepare the data for use	For ODBC data sources: Chapter 2 For Powersoft database interfaces: Chapter 3
2 Install the ODBC driver or Powersoft database interface	Install the ODBC data source driver or Powersoft database interface required to access your data	For PowerBuilder: *Installation and Deployment Guide* For InfoMaker: *Installation Guide*
3 Define the data source or database	Create the required configuration (for an ODBC data source) or database profile (for a Powersoft database interface)	For ODBC data sources: Chapter 2 For Powersoft database interfaces: Chapter 3
4 Connect to the data source or database	Access the data from PowerBuilder or InfoMaker	Chapter 4
5 (Optional) Set additional connection parameters	If necessary, set DBParm parameters and database preferences to fine-tune your database connection	Chapter 5
6 (Optional) Troubleshoot the data connection	If necessary, use the Database Trace and ODBC Driver Manager Trace tools to troubleshoot problems with your data connection	Chapter 6

Types of data sources and databases you can access

Using PowerBuilder or InfoMaker, you can access:

- **ODBC data sources** To access one of the supported Open Database Connectivity (ODBC) data sources, you must install the ODBC driver for that data source. Once the driver is installed, Powersoft's interface to ODBC (PBODB040.DLL) communicates with the driver through the ODBC Driver Manager to access the data you need. For example, you can access a Watcom SQL data source by installing the Watcom SQL ODBC driver.
- **Native databases or DBMSs (non-ODBC)** To access one of the supported native databases or database management systems (DBMSs), you must install the appropriate database software at your site and the corresponding Powersoft database interface. For example, if you have the appropriate SQL Server software installed, you can access a SQL Server database by installing the Powersoft SQL Server interface. PowerBuilder and InfoMaker do *not* go through ODBC to access native databases and DBMSs.

Supported data sources and databases

For a complete list of the data sources and databases supported by the Powersoft Enterprise Series products, see Appendix A, "Supported Data Sources and Databases."

ODBC data sources

An **ODBC data source** consists of the data you want to access and its associated DBMS or file manager, operating system, and, if present, network software that accesses the DBMS. The data source stores and manages the data on behalf of your application.

Examples of ODBC data sources

You can access an ODBC data source that resides locally on your computer or remotely on a network server. The following are examples of ODBC data sources accessible from PowerBuilder or InfoMaker:

- A Microsoft Excel or dBASE file on your computer
- A Watcom SQL relational database installed on a network file server
- A Digital Equipment Corporation (DEC) Rdb database running on the VMS operating system and accessed through DECnet or TCP/IP

Accessing ODBC data sources

You can access an ODBC data source in PowerBuilder or InfoMaker through the Powersoft ODBC interface (PBODB040.DLL). PowerBuilder and InfoMaker provide access to two types of ODBC data sources:

- A Watcom SQL data source using the Watcom SQL ODBC driver
- An ODBC data source using an ODBC driver *other* than the Watcom SQL driver

Watcom SQL ODBC data sources

When you install PowerBuilder or InfoMaker, a Watcom SQL demonstration database is installed by default, unless you deselect this option.

The Watcom SQL demonstration database is an ODBC data source that you access with the Watcom SQL ODBC driver. Initially, you access tables in the demonstration database when you use the PowerBuilder code examples or the InfoMaker tutorial.

You can also create your own Watcom SQL database for use in your application in either of the following ways:

- Use the Database painter in PowerBuilder or InfoMaker when running these products on your computer.
- Create the database some other way, such as with PowerBuilder or InfoMaker running on another user's computer, or with Watcom SQL outside PowerBuilder or InfoMaker.

The method you use to create a Watcom SQL database determines how you connect to it from PowerBuilder or InfoMaker, as follows.

Watcom SQL database created on your computer

When you create a Watcom SQL database in PowerBuilder or InfoMaker on your computer, the software automatically creates the ODBC data source configuration required to connect to the new database. It also creates a database profile so you can easily connect to this database in the future.

For instructions on creating a Watcom SQL database, see the PowerBuilder or InfoMaker *User's Guide*.

For an introduction to database profiles, see "Using database profiles" on page 9.

Watcom SQL database not created on your computer

To access a Watcom SQL database created with PowerBuilder or InfoMaker on *another* computer, or with Watcom SQL outside these products, follow these general steps:

1 Install the Watcom SQL ODBC driver on your computer.

You can install the driver by default with PowerBuilder or InfoMaker, or you install it later when you need it.

2 Prepare to use the Watcom SQL data source.

For instructions, see "Watcom SQL" in Chapter 2, "Using ODBC Data Sources."

3 Define the Watcom SQL data source.

This involves completing the Watcom SQL ODBC Configuration dialog box.

For instructions, see "Watcom SQL" in Chapter 2, "Using ODBC Data Sources."

When you define the data source in PowerBuilder or InfoMaker, a database profile is automatically created. You can select this profile to connect to the data source.

Other ODBC data sources

In addition to Watcom SQL, PowerBuilder and InfoMaker provide access to many other ODBC data sources. To connect to an ODBC data source *other* than Watcom SQL, follow these general steps:

1 Install the ODBC driver for that data source on your computer.

A set of ODBC drivers comes with PowerBuilder or InfoMaker. You can install one or more of these drivers when you first install the products, or later as you need them.

2 Prepare to use the ODBC data source.

For instructions, see the section for your data source in Chapter 2, "Using ODBC Data Sources."

3 Define the ODBC data source.

This involves completing the ODBC Configuration dialog box for the ODBC driver you are using.

For instructions, see the section for your data source in Chapter 2, "Using ODBC Data Sources."

When you define the data source in PowerBuilder or InfoMaker, a database profile is automatically created. You can select this profile to connect to the data source.

For more about how Powersoft products support ODBC, see Chapter 2, "Using ODBC Data Sources."

Using other ODBC drivers

If you are using PowerBuilder Enterprise, PowerBuilder Team/ODBC, or InfoMaker, you can also use ODBC drivers purchased from vendors *other* than Powersoft, such as directly from DBMS vendors.

In most cases, any ODBC driver that is Level 1-compliant or higher will work with these products, However, Powersoft has not tested other ODBC drivers to verify this.

Using ODBC drivers with PowerBuilder Desktop

If you are using PowerBuilder Desktop, you are restricted to the ODBC drivers shipped with the product, plus drivers obtained directly from INTERSOLV for Btrieve, dBASE, NetWare SQL, and Paradox.

To use ODBC drivers from vendors other than Powersoft, you must upgrade to PowerBuilder Team/ODBC.

Powersoft database interfaces

A **Powersoft database interface** is a native (direct) connection to a database or DBMS. If your site uses one of these databases, and if you have the required database server and client software installed, you can access the data by installing the corresponding Powersoft database interface that comes with PowerBuilder or InfoMaker. For example, you can access an ORACLE database by installing one of the Powersoft ORACLE interfaces.

A Powersoft database interface does *not* go through ODBC to access a database. Therefore, you do not complete the ODBC Configuration dialog box to define the data source. Instead, you create a database profile in which you specify connection parameters for accessing the database.

For more about how PowerBuilder and InfoMaker support native database interfaces, see Chapter 3, "Using Powersoft Database Interfaces."

Using database profiles

A **database profile** is a named set of parameters that specifies a connection to a particular data source or database. As described in the preceding sections, database profiles are created automatically for some types of data and by you for others.

Using database profiles is the easiest way for you to manage your data connections. For example, you can:

- Select a database profile to connect to or switch between data sources or databases
- Edit a database profile to customize a connection
- Delete a database profile if you no longer need to access that data

For instructions on using database profiles, see Chapter 4, "Managing Database Connections."

Summary

The type of data you are accessing determines whether PowerBuilder or InfoMaker automatically creates the ODBC configuration (if applicable) and database profile, or you do.

The following table summarizes how an ODBC configuration (if applicable) and database profile are created for each data type:

For this type of data	ODBC configuration is created	Database profile is created
Watcom SQL database created with PowerBuilder or InfoMaker on your computer	Automatically by PowerBuilder or InfoMaker	Automatically by PowerBuilder or InfoMaker
Watcom SQL database created with PowerBuilder or InfoMaker on another computer, or with Watcom SQL outside these products	By you	Automatically by PowerBuilder or InfoMaker
ODBC data source using ODBC driver other than Watcom SQL	By you	Automatically by PowerBuilder or InfoMaker
Powersoft database interface	—	By you

Defining the ODBC data source outside PowerBuilder or InfoMaker

The preceding table assumes that you are using the Configure ODBC dialog box in PowerBuilder or InfoMaker to define the ODBC data source.

If you define the ODBC data source with a tool *other* than PowerBuilder or InfoMaker and then want to access the data source from PowerBuilder or InfoMaker, you must create the database profile yourself.

For instructions, see "Creating a database profile for an existing ODBC data source" in Chapter 4, "Managing Database Connections."

What to do next

For instructions on preparing to use and defining an ODBC data source, see Chapter 2, "Using ODBC Data Sources."

For instructions on preparing to use and defining a Powersoft database interface, see Chapter 3, "Using Powersoft Database Interfaces."

CHAPTER 2

Using ODBC Data Sources

About this chapter

This chapter provides an introduction to the Powersoft ODBC interface. It then describes how to prepare and define each supported ODBC data source in order to connect to it from PowerBuilder or InfoMaker.

Contents

Topic	Page
About the Powersoft ODBC interface	15
About preparing ODBC data sources	23
About defining ODBC data sources	24
What to do next	39
Microsoft Access	40
INTERSOLV Btrieve	45
Microsoft Btrieve	55
INTERSOLV dBASE	61
Microsoft dBASE	69
Microsoft Excel	76
Microsoft FoxPro	82
INTERSOLV NetWare SQL	89
INTERSOLV Paradox 4	94
INTERSOLV Paradox 5	101
Microsoft Paradox	108
DEC Rdb	113
Microsoft Text File	118
Watcom SQL	129

For more information

This chapter gives general information about preparing and defining each ODBC data source. For more detailed information:

- Use the online Help provided by the driver vendor, as described in "Getting help" on page 30. This Help provides important details about using the data source.
- Check for a FaxLine document that describes how to connect to your data source. Many of these FaxLines are available on the Powersoft Infobase CD-ROM. For a complete list of available FaxLines, order the *Technical Information Catalog* from the Powersoft FaxLine system.

About the Powersoft ODBC interface

PowerBuilder and InfoMaker provide an interface to a wide variety of ODBC data sources. This section describes how the Powersoft ODBC interface connects to ODBC data sources.

Supported ODBC data sources

For a complete list of the ODBC data sources supported by the Powersoft Enterprise Series products, see Appendix A, "Supported Data Sources and Databases."

What is ODBC?

Open Database Connectivity (ODBC) is a standard application programming interface (API) developed by Microsoft. It allows a single application to access a variety of data sources for which ODBC-compliant drivers exist. The application uses Structured Query Language (SQL) as the standard data access language.

Applications that provide an ODBC interface, like PowerBuilder or InfoMaker, can access data sources for which an ODBC driver exists. An **ODBC data-source driver** is a dynamic link library (DLL) that implements ODBC function calls. The application invokes the ODBC driver to access a particular data source.

The Powersoft ODBC interface defines the following:

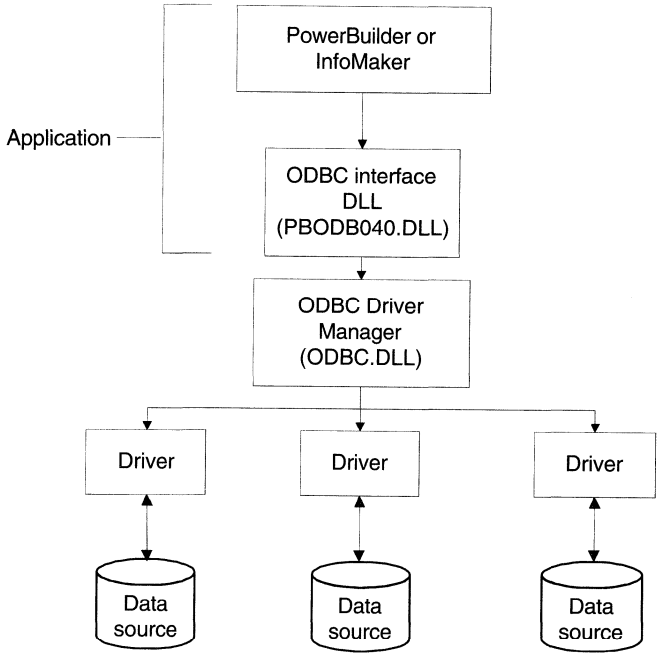
- A library of ODBC function calls that connect to the data source, execute SQL statements, and retrieve results
- A standard way to connect and log on to a data source
- SQL syntax based on the X/Open and SQL Access Group (SAG) CAE specification (1992)
- A standard representation for data types
- A standard set of error codes

Components of an ODBC connection

When you access an ODBC data source from PowerBuilder or InfoMaker, your connection goes through several layers before reaching the data source. It is important to understand that each layer represents a separate component of the connection, and that each component may come from a different vendor.

Because ODBC is a standard interface, PowerBuilder and InfoMaker use the same ODBC interface DLL (PBODB040.DLL) to access every ODBC data source. As long as a driver is ODBC-compliant, PowerBuilder and InfoMaker can access it through the interface to the ODBC Driver Manager (ODBC.DLL).

The following diagram shows the general components of a Powersoft ODBC connection. Notice that PowerBuilder or InfoMaker and the ODBC interface DLL work together as the application component.



The following table gives the provider and a brief description of each ODBC component shown in the diagram.

Component	Provider	What it does
Application	Powersoft	Calls ODBC functions to submit SQL statements and catalog requests to and retrieve results from a data source. PowerBuilder and InfoMaker use the same ODBC interface DLL (PBODB040.DLL) to access all ODBC data sources.
ODBC Driver Manager (ODBC.DLL)	Microsoft	Installs, loads, and unloads drivers for an application.
Driver	Driver vendor	Processes ODBC function calls, submits SQL requests to a particular data source, and returns results to an application. If necessary, translates an application's request to conform to the SQL syntax supported by the backend database. For more information, see "Types of ODBC drivers" next.
Data source	DBMS or database vendor	Stores and manages data for an application. Consists of the data to be accessed and its associated DBMS, operating system, and, if present, network software that accesses the DBMS.

For more information

For diagrams showing the basic components of the ODBC connection for each supported data source, see the section for your data source driver later in this chapter.

Types of ODBC drivers

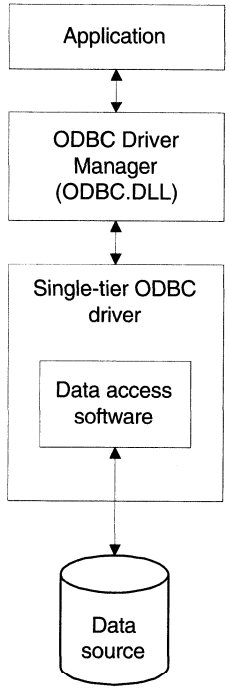
When you are connected to an ODBC data source in PowerBuilder or InfoMaker, you may see messages from the ODBC driver that include the words *single-tier* or *multiple-tier*. These terms refer to the two types of drivers defined by the ODBC standard:

- Single-tier driver
- Multiple-tier driver

Single-tier driver

A single-tier ODBC driver processes both ODBC functions and SQL statements. In other words, a single-tier driver includes the data access software required to manage the data source file and catalog tables.

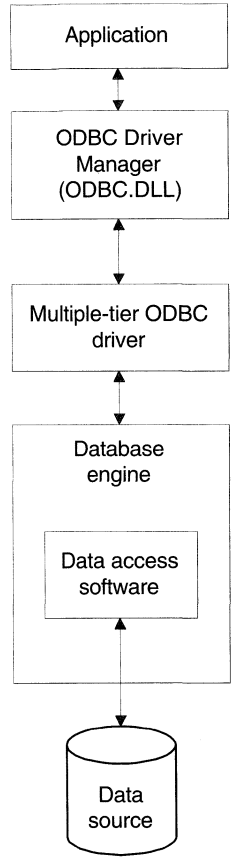
An example of a single-tier ODBC driver is one that accesses Xbase files, such as dBASE.



Multiple-tier driver

A multiple-tier ODBC driver processes ODBC functions, but sends SQL statements to the database engine for processing. Unlike the single-tier driver, a multiple-tier driver does not include the data access software required to manage the data directly.

An example of a multiple-tier ODBC driver is the Watcom SQL driver.



Ensuring the proper ODBC driver conformance levels

If you are using PowerBuilder Enterprise, PowerBuilder Team/ODBC, or InfoMaker, you can access data with ODBC drivers purchased from vendors *other* than Powersoft, such as directly from DBMS vendors. If you purchase an ODBC driver from another vendor, it must meet certain conformance requirements in order to work properly with PowerBuilder or InfoMaker. This section describes how to make sure your driver meets these requirements.

What are ODBC conformance levels?

PowerBuilder and InfoMaker can access many data sources for which ODBC-compliant drivers exist. However, ODBC drivers manufactured by different vendors may vary widely in the functions they provide.

To ensure a standard level of compliance with the ODBC interface, and to provide a means by which application vendors can determine if a specific driver provides the functions they need, ODBC defines conformance levels for drivers in two areas:

- **API** Deals with supported ODBC function calls
- **SQL grammar** Deals with supported SQL statements and SQL data types

API conformance levels

ODBC defines three API conformance levels, in order of increasing functionality:

- **Core** A set of core API functions that corresponds to the functions in the X/Open and SAG Call Level Interface (CLI) specification
- **Level 1** Includes all Core API functions as well as several extended functions
- **Level 2** Includes all Core and Level 1 API functions as well as additional extended functions

❖ **To ensure the proper ODBC driver API conformance level:**

- Make sure the ODBC drivers you use with PowerBuilder or InfoMaker meet *Level 1 or higher* API conformance requirements.

SQL conformance levels

ODBC defines three SQL grammar conformance levels, in order of increasing functionality:

- **Minimum** A set of SQL statements and data types that meets a basic level of ODBC conformance
- **Core** Includes all Minimum SQL grammar as well as additional statements and data types that roughly correspond to the X/Open and SAG CAE specification (1992)
- **Extended** Includes all Minimum and Core SQL grammar as well as an extended set of statements and data types that supports common DBMS extensions to SQL

❖ **To ensure the proper ODBC driver SQL conformance level:**

- Make sure the ODBC drivers you use with PowerBuilder or InfoMaker meet *Core or higher* SQL conformance requirements.

Obtaining ODBC drivers

There are two ways that you can obtain ODBC drivers for use with PowerBuilder or InfoMaker:

- **From Powersoft (recommended)** Install one or more of the ODBC drivers shipped with PowerBuilder or InfoMaker. You can do this when you first install PowerBuilder or InfoMaker, or later. This method ensures that the driver was tested and found to work with PowerBuilder and InfoMaker.
- **From another vendor** PowerBuilder Enterprise, PowerBuilder Team/ODBC, and InfoMaker enable you to access data with Level 1 or higher ODBC-compliant drivers purchased from a vendor other than Powersoft. In most cases, these drivers will work with PowerBuilder or InfoMaker. However, Powersoft has not tested them to verify this.

Getting information about ODBC drivers from other vendors

For more about using PowerBuilder and InfoMaker with an ODBC driver obtained from another vendor, check whether there is a FaxLine document that describes this driver. Many of the FaxLines are available on the Powersoft Infobase CD-ROM. For a complete list of available FaxLines, order the *Technical Information Catalog* from the Powersoft FaxLine system.

Using ODBC drivers with PowerBuilder Desktop

If you are using PowerBuilder Desktop, you are limited to the ODBC drivers shipped with the product, plus the following ODBC drivers obtained directly from INTERSOLV, Inc.:

- Btrieve
- dBASE (also supports Clipper, FoxPro, and FoxBASE files)

- NetWare SQL
- Paradox 4 and Paradox 5

To use *other* ODBC drivers, you should upgrade to PowerBuilder Team/ODBC.

About preparing ODBC data sources

The first step in connecting to an ODBC data source is to prepare to use the data source. Preparing the data source ensures that you will be able to connect to it and use your data in PowerBuilder or InfoMaker.

You prepare a data source *outside* PowerBuilder or InfoMaker *before* you start the product, define the data source, and connect to it. The requirements differ for each data source but in general, preparing a data source involves making sure that:

- The required files and directories are correctly installed on your computer
- Configuration files (such as AUTOEXEC.BAT) are properly set up
- The tables you want to access have appropriate names
- If necessary, the tables you want to access have unique indexes associated with them

For instructions, see "Preparing to use the data source" in the section for your data source driver later in this chapter.

About defining ODBC data sources

Each ODBC data source requires a corresponding ODBC driver to access it. When you define an ODBC data source, you are providing information about the data source that the driver needs to connect to it. (Defining an ODBC data source is often referred to as configuring the data source.)

After you prepare to use the data source as described in the preceding section, you start PowerBuilder or InfoMaker and define the data source. To define a data source, you must complete the ODBC setup dialog box for the corresponding driver. For example, to define a Watcom SQL data source, you must complete the Watcom SQL ODBC Configuration dialog box, as described on page 129.

The contents of the ODBC setup dialog box varies for each driver. However, most setup dialog boxes enable you to provide the following information when defining an ODBC data source:

- Data source name
- Optional description
- Other information about the DBMS as needed

The rest of this section describes what you need to know to define an ODBC data source in order to connect to it from PowerBuilder or InfoMaker.

When you define the data source

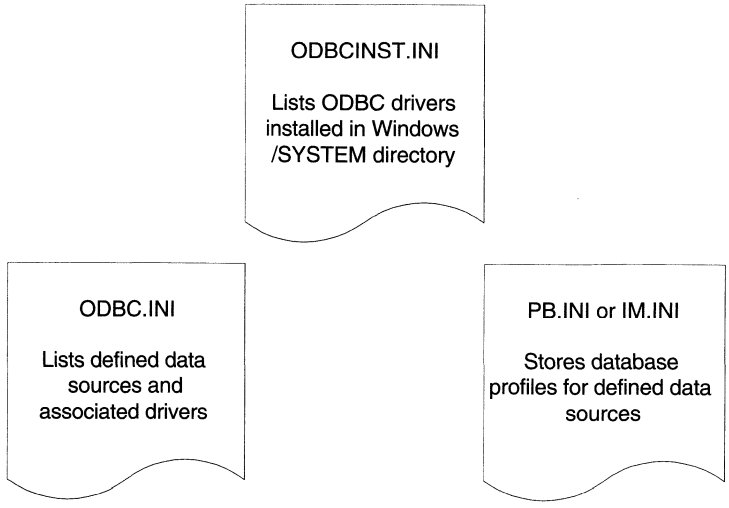
For some ODBC data sources, PowerBuilder or InfoMaker defines the data source and creates the database profile for you. For example, a Watcom SQL database is an ODBC data source. When you create a Watcom SQL database using PowerBuilder or InfoMaker on your computer, PowerBuilder or InfoMaker automatically defines the Watcom SQL data source and creates the database profile.

However, *you* must define the ODBC data source yourself when you want to access:

- A Watcom SQL database that you did *not* create in PowerBuilder or InfoMaker on your computer
- An ODBC data source that uses an ODBC driver *other* than the Watcom SQL driver

How Powersoft products access the data source

When you access an ODBC data source from PowerBuilder or InfoMaker, there are three files on your computer that work with the ODBC interface and driver to make the connection:



ODBCINST.INI file

When you install an ODBC-compliant driver supplied by Powersoft or another vendor, the ODBCINST.INI file in your Windows directory is automatically updated with a description of the driver. This description includes:

- The DBMS or data source associated with the driver.
- The drive and directory of the driver and setup DLLs. For some data sources, the driver and setup DLLs are the same.

For example, the following portion of an ODBCINST.INI file shows two ODBC drivers installed: Watcom SQL and Microsoft dBASE.

```
[ODBC Drivers]
WATCOM SQL 4.0 =Installed
dBase files (*.dbf)=Installed

[WATCOM SQL 4.0]
driver=c:\windows\system\wod40w.dll
setup=c:\windows\system\wod40w.dll
```

```
[dBase files (*.dbf)]
Driver=c:\windows\system\simba.dll
Setup=c:\windows\system\simadmin.dll
```

ODBC.INI file

When you define a data source for a particular ODBC driver, the driver writes the values you specify in the ODBC setup dialog box to the ODBC.INI file in your Windows directory.

The [ODBC Data Sources] section of ODBC.INI lists the name of each defined data source and its associated DBMS. The ODBC.INI file also includes a separate section for each data source. This section contains the values specified for that data source in the ODBC setup dialog box. The values may vary for each data source, but generally include the following:

- Name
- Optional description
- Other information required by the DBMS

For example, the following portion of an ODBC.INI file shows definitions for a Watcom SQL data source named Products and a Microsoft dBASE data source named Sales.

```
[ODBC Data Sources]
Products=WATCOM SQL 4.0
Sales=dBase files (*.dbf)

[Products]
Driver=c:\windows\system\wod40w.dll
UID=dba
PWD=sql
Description=Watcom SQL 4.0 Products DB
Database=C:\WATCOM\PRODUCTS.DB
Start=db32w
DatabaseFile=C:\WATCOM\PRODUCTS.DB
DatabaseName=Products

[Sales]
Driver=c:\windows\system\simba.dll
Description=Microsoft dBASE IV Sales data source
FileType=dBase4
DataDirectory=c:\odbcsrcs\dbase\dbase4
SingleUser=False
```

PB.INI and IM.INI files

When you define an ODBC data source in PowerBuilder or InfoMaker, a database profile is automatically created for that data source. You can then select this database profile to connect to the data source.

Database profiles for all data sources are stored in the initialization (INI) file for PowerBuilder or InfoMaker, as shown in the following table:

Powersoft product	INI file
PowerBuilder	PB.INI
InfoMaker	IM.INI

For example, the following portion of a PB.INI file shows the database profiles for the Products and Sales data sources.

```
[PROFILE Products]
DBMS=ODBC
Database=
UserId=
DatabasePassword=
LogPassword=
ServerName=
LogId=
Lock=
DbParm=Connectstring='DSN=Products'
Prompt=0

[PROFILE Sales]
DBMS=ODBC
Database=
UserId=
DatabasePassword=
LogPassword=
ServerName=
LogId=
Lock=
DbParm=Connectstring='DSN=Sales'
Prompt=0
```

Values used with ODBC data sources

As shown in these examples, the most important values in a database profile for an ODBC data source are the following:

- **DBMS** The DBMS value indicates that you are using the Powersoft ODBC interface (PBODB040.DLL) to connect to the data source.

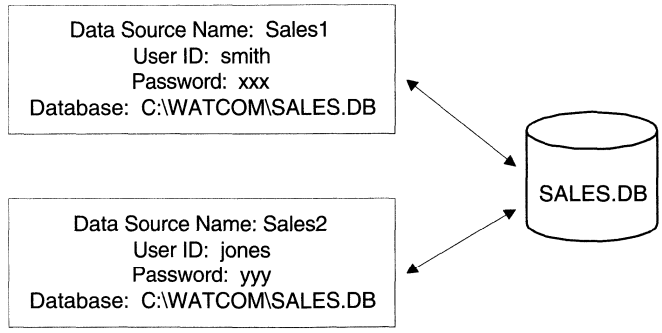
- **DBParm** The ConnectString value in the DBParm field controls your ODBC data source connection. The connect string *must* specify the DSN (data source name) value, which tells ODBC which data source you want to access. When you select a database profile to connect to a data source, ODBC looks in the ODBC.INI file for a section that corresponds to the data source name in your profile. ODBC then uses the information in this section of ODBC.INI to load the programs required to connect to the data source.

About defining multiple data sources for the same data

When defining an ODBC data source in PowerBuilder or InfoMaker, each data source name must be unique. You can, however, define multiple data sources that access the same data, as long as the data sources have unique names.

To illustrate this, assume your data source is a Watcom SQL database that resides in C:\WATCOM\SALES.DB. Depending on your application, you may want to specify different sets of connection parameters for accessing the database—for example, using different passwords and user IDs.

To do this, you can define two ODBC data sources named Sales1 and Sales2 that specify the same database name (C:\WATCOM\SALES.DB) but different user IDs and passwords. Defining these data sources automatically creates corresponding database profiles named Sales1 and Sales2. When you connect to the data source using either of these profiles, you are using different connection parameters to access the same data.



Inheriting ODBC data sources

In addition to defining a new ODBC data source in PowerBuilder or InfoMaker, you can also access an existing ODBC data source that you inherit from another user. For example, you can access a Watcom SQL database created by someone else at your site. To do so, you must:

- Install the required data files for the data source.
- Define the ODBC data source on your computer by completing the ODBC setup dialog box for the driver you are using.

❖ To access an inherited ODBC data source:

1. Make sure the required data files are installed on your computer or on the network.

   For example, if you are accessing a Watcom SQL data source, you need the database (DB) file and the accompanying transaction log (LOG) file. If you are accessing an Excel data source, you need the Excel worksheet (XLS file).

2. Make sure you have prepared the data file for use with PowerBuilder or InfoMaker.

   For instructions, see "Preparing to use the data source" in the section for your data source driver later in this chapter.

3. Make sure the ODBC driver required to access the data source is installed on your computer.

   For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide.*

   For more about using an ODBC driver supplied by another vendor, see "Obtaining ODBC drivers" on page 21.

4. Define the ODBC data source on your computer by completing the ODBC setup dialog box for the driver you are using.

   For example, to define a Watcom SQL data source, you must complete the Watcom SQL ODBC Configuration dialog box.

   For instructions, see the section for your data source driver on pages 40 through 129.

What happens

When you complete the ODBC setup dialog box for your driver, PowerBuilder or InfoMaker automatically does the following:

- Updates the ODBC.INI file on your computer with the information required to connect to the data source
- Creates a database profile so you can easily connect to this data source

Getting help

The ODBC setup dialog boxes shown on pages 40 through 129 reflect the versions of the drivers that shipped with PowerBuilder or InfoMaker when this manual was printed. However, driver vendors frequently update their drivers, and this often results in changes to the driver features and corresponding ODBC setup dialog boxes.

In some cases, the ODBC setup dialog box you use may look different from the one shown in this manual. Therefore, to ensure that you have the most up-to-date and complete information about the ODBC drivers Powersoft supplies, in addition to reading this chapter you should use one or more of the following methods to help you define an ODBC data source:

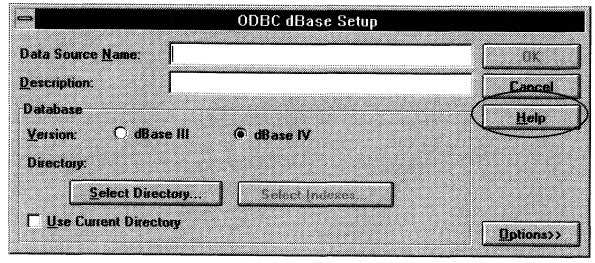
To get help on	Do this
The fields in the Configure ODBC dialog box	Click the Help button in the Configure ODBC dialog box.
Your ODBC driver and data source	Click the Help button in the ODBC setup dialog box for your driver, as described below. Check whether there is a Powersoft FaxLine document that describes how to connect to your data source. For a complete list of available FaxLines, order the *Technical Information Catalog* from the Powersoft FaxLine system.

Displaying Help for ODBC drivers

The online Help for ODBC drivers in PowerBuilder and InfoMaker is provided by the driver vendors. It gives detailed descriptions of the fields in the ODBC setup dialog boxes and information about using the driver to access your data source.

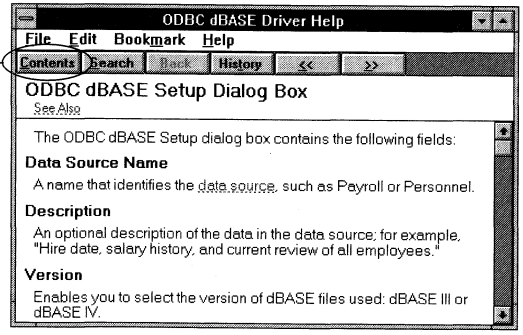
❖ To display online Help for an ODBC driver:

1 Click the Help button in the ODBC setup dialog box for the driver.



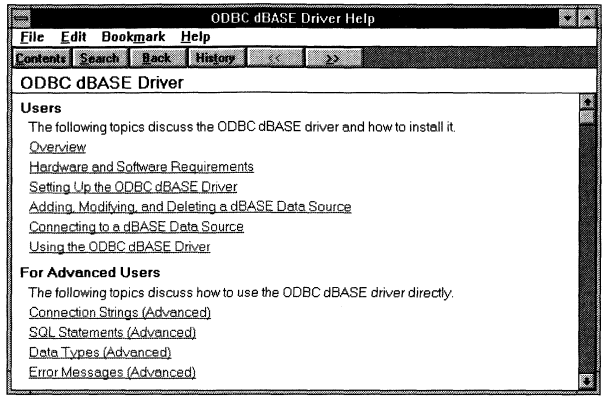
A Help window appears describing the fields in the setup dialog box you are completing.

Click here to display a list of Help topics



2 Click the Contents button in the Help window to display additional Help topics for this driver.

Another Help window appears listing the topics you can view.



3 Click on an underlined topic to display its Help window.

Completing the ODBC setup dialog box

To define an ODBC data source, you must complete the ODBC setup dialog box for the driver you are using to access the data source. There are two ways to complete the ODBC setup dialog box:

- Specify the data source name and location in the ODBC setup dialog box. If the driver requires additional information to connect to the data source, the software prompts you to enter it.
- Specify values for any additional fields in the ODBC setup dialog box.

This section gives general procedures for each of these methods.

Before you start

Before you define an ODBC data source, make sure you've installed the ODBC driver required to access the data source, and prepared the data source for use in PowerBuilder or InfoMaker.

For more about ODBC drivers, see "About the Powersoft ODBC interface" on page 15.

For instructions on preparing an ODBC data source, see "Preparing to use the data source" in the section for your data source driver on pages 40 through 129.

Specifying the data source name and location

❖ To define an ODBC data source by specifying its name and location:

1 Click the Configure ODBC button in the PowerBar.
or
In the PowerBuilder or InfoMaker Database painter, select File➤Configure ODBC from the menu bar.

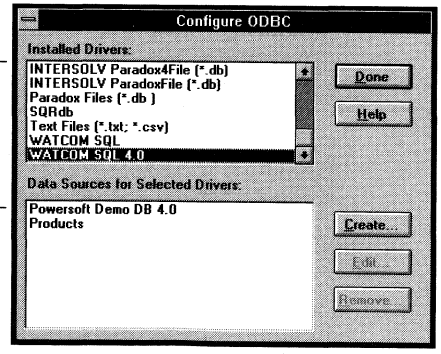
> **Configure ODBC button**
> If your PowerBar does not include the Configure ODBC button, use the customize feature to add the button to the PowerBar.
>
> For information about customizing toolbars, see the PowerBuilder or InfoMaker *User's Guide*.

The Configure ODBC dialog box appears, listing the following:

- The ODBC drivers installed on your computer. The names of the ODBC drivers come from the ODBCINST.INI file in your Windows directory.
- The data sources defined for each selected driver. The names of the data sources come from the ODBC.INI file in your Windows directory.

Installed drivers (from ODBCINST.INI file)

Defined data sources (from ODBC.INI file)



2 Select the ODBC driver for the data source you want to access.

If any data sources are already defined for that driver, their names appear in the data source list.

3 Click the Create button to create a new ODBC data source definition.

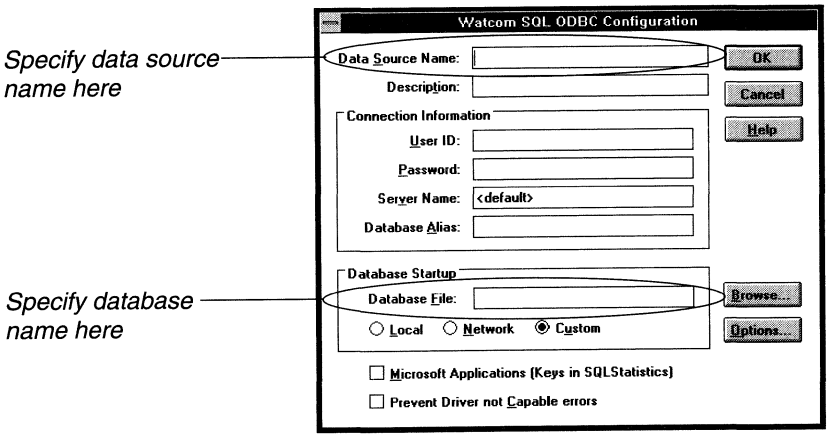
The ODBC setup dialog box for the selected driver appears. The contents of this dialog box varies, depending on the driver you select.

Getting help

For basic information about completing the ODBC setup dialog boxes, see the section for your data source driver on pages 40 through 129.

For detailed information about the fields in the ODBC setup dialog boxes, click the Help button in the ODBC setup dialog box. This Help comes from the driver vendor and provides important information about using the driver to access your data source.

For example, the following dialog box appears if you select the Watcom SQL driver.



4 In the Data Source Name field, type a name for the data source.

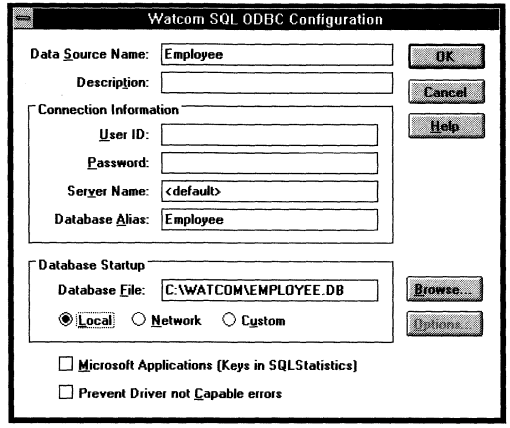
The name you specify becomes the name of the database profile for this data source.

5 Specify the fully qualified database name (such as C:\WATCOM\EMPLOYEE.DB), or the drive and directory where the database resides (such as C:\WATCOM).

Do this in one of the following ways, depending on the contents of the ODBC setup dialog box you are completing:

- Type the name in the appropriate field. In most dialog boxes, this field is labeled Database, Database File, or Database Directory.
- If the ODBC setup dialog box contains a Browse or Select button, you can click this button to select the database name or directory from a list of available databases. The name you select appears in the appropriate field in the setup dialog box.

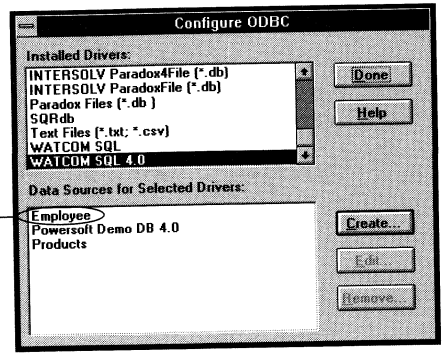
For example, the following shows the Watcom SQL ODBC Configuration dialog box for a data source named Employee.



6 Click OK.

If PowerBuilder or InfoMaker requires additional information to connect to the data source, you will be prompted for it later when you select the database profile to connect to the data source.

When you complete the definition, the Configure ODBC dialog box appears. The data source you defined is listed with any other data sources previously defined for the selected driver.

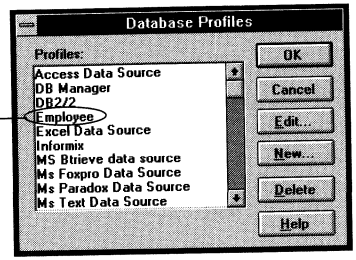


7 Repeat steps 2 through 6 if you want to define another ODBC data source.

8 Click Done when you are finished defining the ODBC data sources you need.

PowerBuilder or InfoMaker updates the ODBC.INI file with each data source definition, and creates a database profile for each data source you defined.

For example, after you define a Watcom SQL data source named Employee, a database profile named Employee appears in the Database Profiles dialog box.



For instructions on using database profiles to connect to a data source, see Chapter 4, "Managing Database Connections."

Specifying values for additional fields

As described in "Specifying the data source name and location" on page 33, the only values you *must* supply in the ODBC setup dialog box when defining a data source are:

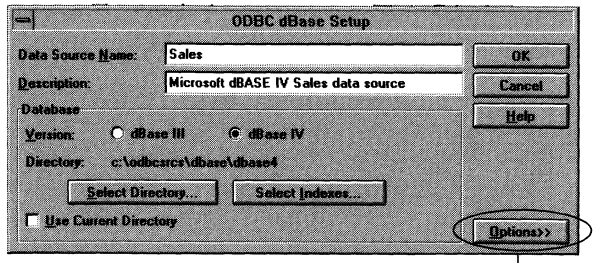
- Data source name
- Database name or directory

If you prefer, however, you can specify values for any additional fields in the ODBC setup dialog box when you define the data source.

❖ To define an ODBC data source by specifying values for additional fields:

1 Follow steps 1 through 5 in "Specifying the data source name and location" on page 33.

For example, here is the ODBC dBASE Setup dialog box after specifying the data source name, description, version, and directory. (The ODBC dBASE Setup dialog box appears when you select the Microsoft dBASE driver.)

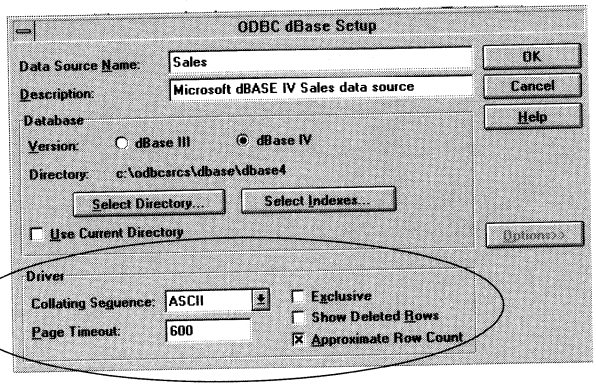


Click here to display additional fields

Notice that in the ODBC setup dialog boxes for some drivers, like Microsoft dBASE, the additional fields are not visible when the dialog box first appears.

2 Click the Options button to display additional fields.

The ODBC setup dialog box expands to reveal additional fields that you can complete.



Additional driver options

3 Specify information in the additional fields as appropriate for your data source connection.

> **Getting help**
>
> For basic information about completing the ODBC setup dialog boxes, see the section for your data source driver on pages 40 through 129.
>
> For detailed information about the fields in the ODBC setup dialog boxes, click the Help button in the ODBC setup dialog box. This Help comes from the driver vendor and provides important information about using the driver to access your data source.

4 Click OK.

The Configure ODBC dialog box appears, listing the data source you just defined.

5 Repeat steps 1 through 4 if you want to define another ODBC data source.

6 Click Done when you are finished defining the ODBC data sources you need.

PowerBuilder or InfoMaker updates the ODBC.INI file with each data source definition, and creates a database profile entry in the PB.INI or IM.INI file for each data source you defined.

What to do next

For step-by-step instructions on how to prepare and define your ODBC data source, go to the section for your data source driver on pages 40 through 129.

Microsoft Access

This section describes how to prepare and define an Access data source in order to connect to it using the Microsoft Access ODBC driver.

Supported versions

The Microsoft Access ODBC driver supports connection to Access Version 1.x data sources.

Supported SQL operations

The Microsoft Access ODBC driver supports the following SQL operations with Access Version 1.x data sources:

CREATE INDEX

CREATE TABLE

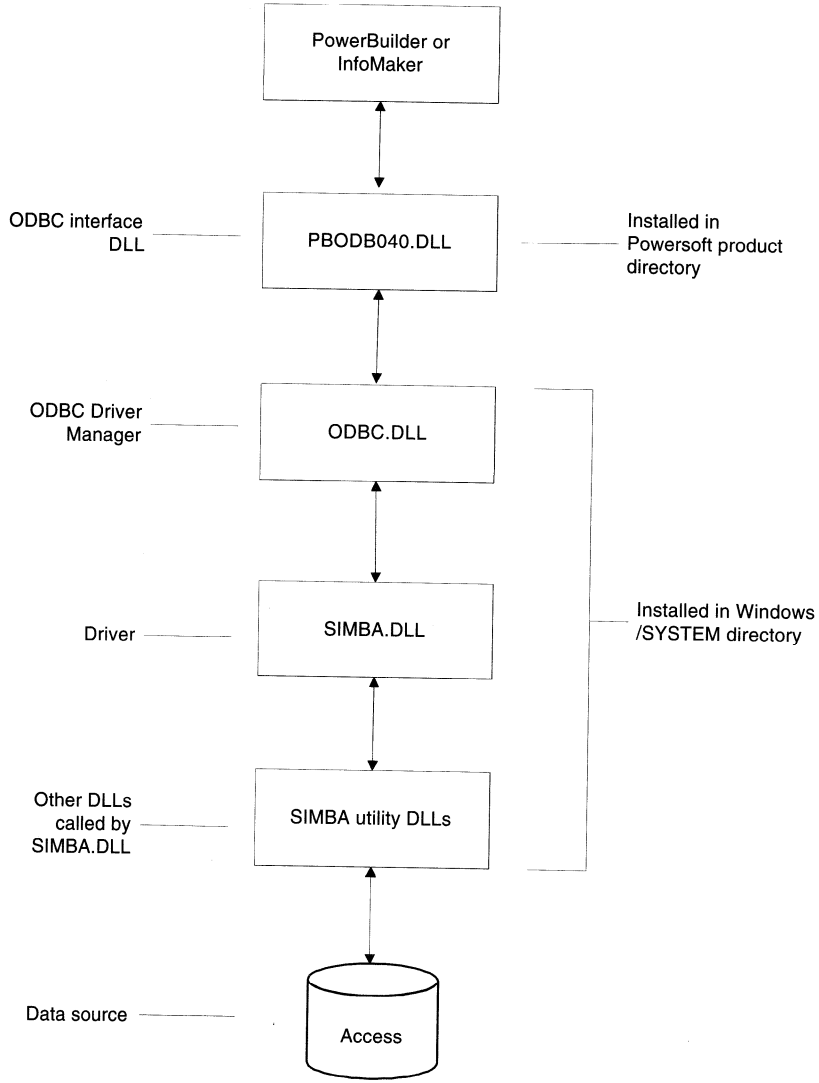
DELETE INDEX

DELETE TABLE

UPDATE

Basic software components

The following diagram shows the basic software components you need to connect to an Access data source using the Microsoft Access ODBC driver. All of these files come with PowerBuilder or InfoMaker.



Preparing to use the data source

Before you define and connect to an Access data source from PowerBuilder or InfoMaker, follow these steps to prepare the data source.

❖ To prepare an Access data source:

1 Issue the MS-DOS SHARE command in either of the following ways:

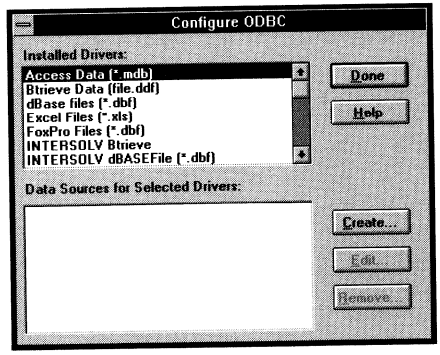
- Type the command from an MS-DOS prompt.
- Include the command line in your AUTOEXEC.BAT file.

2 Start Windows on your computer.

Defining the data source

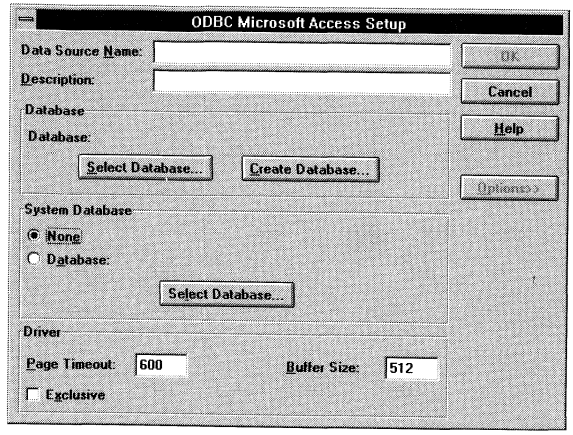
❖ To define an Access data source for the Microsoft Access driver:

1 Select the Microsoft Access driver in the Configure ODBC dialog box.



2 Click the Create button.

The ODBC Microsoft Access Setup dialog box appears. (Click the Options button if necessary to display the entire dialog box.)



3 Specify values as follows:

For more information

For a detailed description of each field in the ODBC Microsoft Access Setup dialog box, click the Help button.

Field	Value
Data Source Name	A short name to identify the data source. This becomes the name of the database profile created for this data source.
Description	(Optional) A description of the data source.
Database	Displays the full name of the Access database to which you are connecting (for example, C:\ACCESS\ORDENTRY.MDB). To display the name of an existing database in the Database field, click the Select Database button and choose a database from the list. To create a new database and display its name in the Database field, click the Create Database button and type the name of the database you want to create.

Field	Value
System Database	(Optional) Specifies whether you want to use a system database with the Access data source. To log in to the database as the Admin user and not use a system database, click the None radio button. (This is the default selection.) To use a system database, click the Database radio button and then click the Select Database button. Select or type the name of a system database (for example, C:\ACCESS\SYSTEM.MDA).
Page Timeout	(Optional) The time in tenths of a second that an unused page remains in the buffer before being removed. The default is 600 (60 seconds).
Exclusive	(Optional) Select this checkbox to open Access files in exclusive mode. Exclusive mode allows file access by only one user at a time, and enhances performance. Leave this checkbox deselected to open Access files in shared mode. Shared mode allows file access by more than one user at a time.
Buffer Size	(Optional) The size of the internal buffer, in kilobytes, that Access uses to transfer data to and from the disk. The default buffer size is 512 kilobytes.

4 Click OK to create the data source definition.

What to do next

For instructions on connecting to the data source, see Chapter 4, "Managing Database Connections."

INTERSOLV Btrieve

This section describes how to prepare and define a Btrieve data source in order to connect to it from PowerBuilder or InfoMaker using the INTERSOLV Btrieve ODBC driver.

Supported versions

The INTERSOLV Btrieve ODBC driver supports connection to Btrieve Version 5.x and 6.x data sources.

Supported SQL operations

The INTERSOLV Btrieve ODBC driver supports the following SQL operations with Btrieve Version 5.x and 6.x data sources:

CREATE INDEX

CREATE TABLE

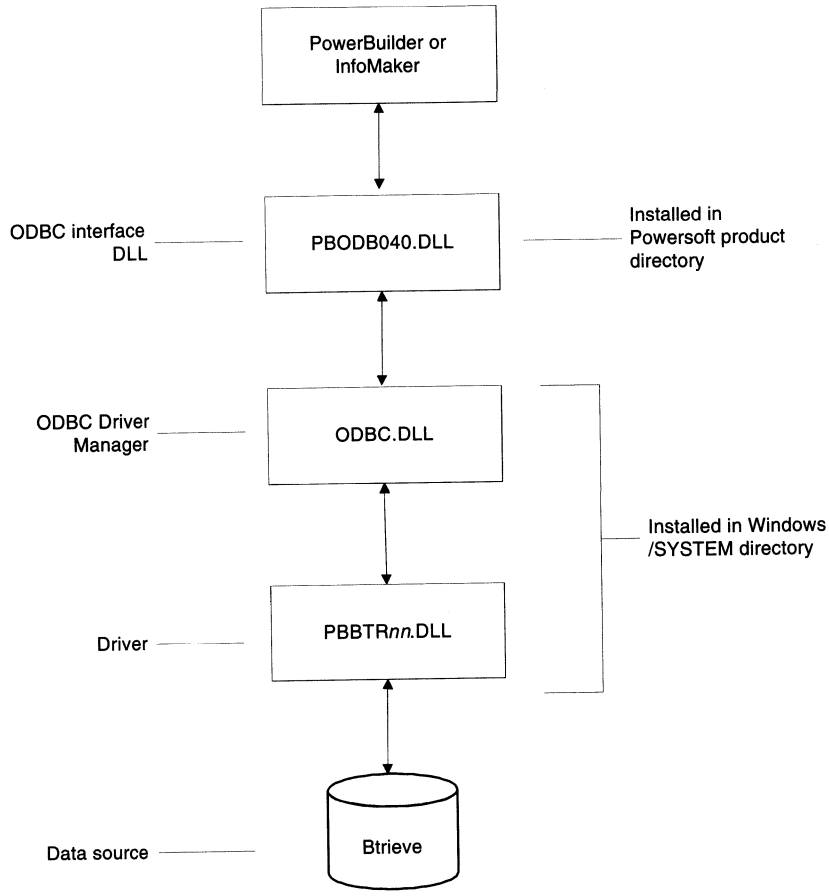
DELETE INDEX

DELETE TABLE

UPDATE

Basic software components

The following diagram shows the basic software components you need to connect to a Btrieve data source using the INTERSOLV Btrieve ODBC driver. All of these files come with PowerBuilder or InfoMaker.



Preparing to use the data source

Before you define and connect to a Btrieve data source from PowerBuilder or InfoMaker, follow these steps to prepare the data source.

❖ To prepare a Btrieve data source when using the INTERSOLV Btrieve driver to connect:

1 In addition to PBODB040.DLL, ODBC.DLL, and PBBTR*nn*.DLL (as shown on page 46), make sure the files listed in the following table are also installed on your computer, and that the directory containing these files appears in your DOS path:

Btrieve version	Required files	Provider
Btrieve 5.x (single-user version installed locally)	WBTRCALL.DLL (non-networked version)	Powersoft or INTERSOLV
	WDDLSVCS.DLL	INTERSOLV
Btrieve 6.x (multi-user version installed on a network)	WBTRCALL.DLL (networked version)	Btrieve Technologies
	WDDLSVCS.DLL	INTERSOLV

2 Make sure you have the necessary NetWare SQL data dictionary files in the same directory as the Btrieve data source you want to access.

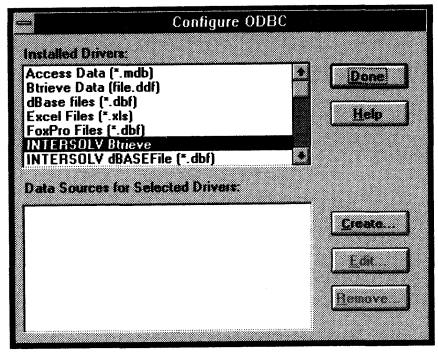
Without the proper data dictionary files, you cannot access Btrieve data through PowerBuilder or InfoMaker.

For instructions, see "Defining the structure of Btrieve tables" on page 50.

Defining the data source

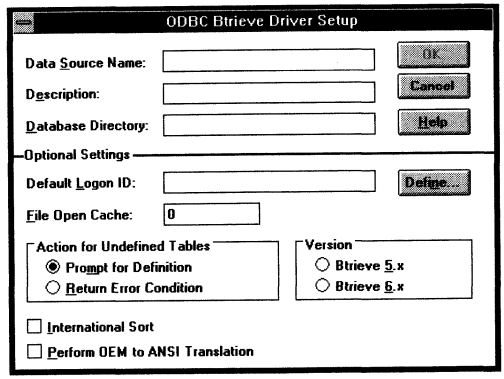
❖ To define a Btrieve data source for the INTERSOLV Btrieve ODBC driver:

1 Select the INTERSOLV Btrieve driver in the Configure ODBC dialog box.



2 Click the Create button.

The ODBC Btrieve Driver Setup dialog box appears.



3 Specify values as follows:

> **For more information**
>
> For more about each field in the ODBC Btrieve Driver Setup dialog box, click the Help button.

Field	Value
Data Source Name	A short name to identify the data source. This becomes the name of the database profile created for this data source.
Description	(Optional) A description of the data source.
Database Directory	The drive and directory containing the Btrieve data source and data dictionary files (for example, C:\BTRIEVE).
Default Logon ID	(Optional) The default logon ID used to connect to your Btrieve database. A logon ID is required only if security is enabled on your database.
File Open Cache	(Optional) The maximum number of unused file opens to cache. For example, the value 3 specifies that when you open and close three tables, the driver keeps all three tables open so it does not have to reopen them if another query uses one of the tables. The default is 0, which means no file opens are cached.
Action for Undefined Tables	(Optional) Click the Prompt for Definition radio button to be prompted when the driver finds a Btrieve table for which it has no structure information. Click the Return Error Condition radio button to return an error when the driver finds a Btrieve table for which it has no structure information.
Version	(Optional) Click Btrieve 5.x to access a Btrieve Version 5.x data source. Click Btrieve 6.x to access a Btrieve Version 6.x data source.
International Sort	(Optional) Select this checkbox to use the international sort order as defined by your operating system. Leave this checkbox deselected to use the ASCII sort order.

Field	Value
Perform OEM to ANSI Translation	(Optional) Select this checkbox if the database file stores data in the IBM PC character set. Leave this checkbox deselected if the database file stores data in the ANSI character set. This is the default.
Define	Displays the Define File dialog box to define the structure of an existing Btrieve table. For more information, see "Defining the structure of Btrieve tables" next.

4 Click OK to save the data source definition.

Defining the structure of Btrieve tables

Unlike some other databases, Btrieve does not store field information with the data file itself. Therefore, in order to access an existing Btrieve table from PowerBuilder or InfoMaker, you *must* have the necessary NetWare SQL data dictionary files in the same directory as this table.

NetWare SQL data dictionary

The INTERSOLV Btrieve ODBC driver uses the NetWare SQL data dictionary to obtain information about the structure of your Btrieve tables. This includes information about table names, column names, data types, precision, scale, nulls, and indexes.

The data dictionary defines the structure of the table, and consists of the files listed in the following table:

File	What it stores
FILE.DDF	Information about each table in the database
FIELD.DDF	Information about each column in each table
INDEX.DDF	Information about each index

Using CDB=1 in the ConnectString

If a complete data dictionary does not exist in your Btrieve database directory, you can set a DBParm value in the database profile for this data source that tells the INTERSOLV driver to create a new data dictionary if one does not already exist. (PowerBuilder and InfoMaker create a database profile automatically when you define an ODBC data source.)

❖ To set the ConnectString DBParm to create a new data dictionary:

- In the DBParm field in the Database Profile Setup dialog box, type the value **CDB=1** in the ConnectString parameter, as follows:

  ConnectString = ' DSN = Btrieve;**CDB=1** '

  For more information, see the description of ConnectString in Chapter 5, "Setting Additional Connection Parameters."

Creating DDF files

The new data dictionary files will contain information about any new tables you create. They will not, however, contain information about existing tables you may want to access. To access an existing Btrieve table, you must have the DDF files created for that table.

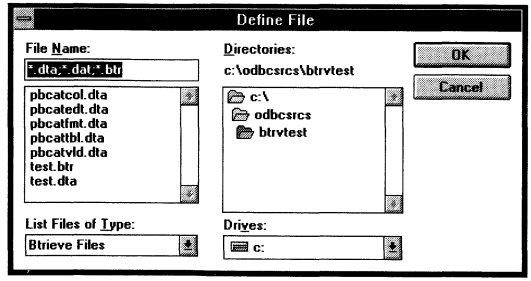
If you do not have the DDF files for an existing Btrieve table, you can use the INTERSOLV Btrieve driver to create the DDF files. This is not an easy task, since you must know the exact layout and internal structure of the table in order to create DDF files for it. Creating DDF files does not change your Btrieve file in any way. You can continue to access the file directly through any existing Btrieve application.

If you know the layout and structure of the Btrieve tables you want to access, use the following procedure to create the DDF files.

❖ To define the structure of an existing Btrieve table:

1 Click the Define button in the ODBC Btrieve Driver Setup dialog box.

The Define File dialog box appears.

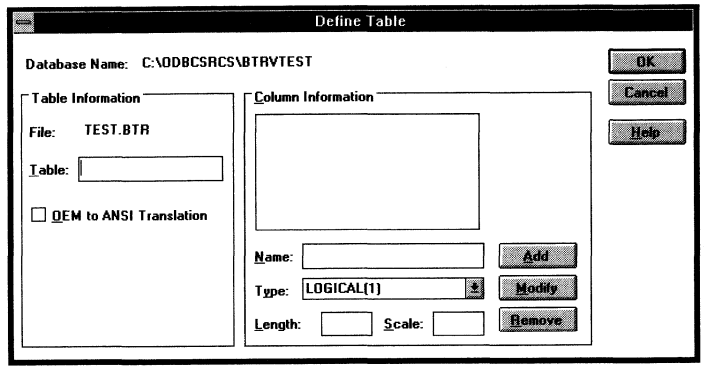


2 Select the BTR file (Btrieve table) to define.

When you access a Btrieve data source from PowerBuilder or InfoMaker, a directory represents the database and each BTR file in the directory represents a table in the database.

3 Click OK.

The Define Table dialog box appears. The Database Name field displays the database directory, and the File field displays the table name you selected.



4 Specify values as follows:

Field	Value
Table	Enter the name of the Btrieve table to be returned by the SQLTables API call. The name can have a maximum of 20 characters and must be unique among all defined tables in the database.
OEM to ANSI Translation	Select this checkbox if the database file stores data in the IBM PC character set. Leave this checkbox deselected if the database file stores data in the ANSI character set. This is the default.
Column Information	Lists the names of any columns defined in the selected table. To modify or remove a column's definition, select its name from the Column Information list.
Name	Displays the name of the selected column. To specify or change this value, type a new name in the field.

Field	Value
Type	Displays the data type of the selected column. To specify or change this value, select another data type from the dropdown listbox.
Length	Displays the length of the selected column. To specify or change this value, type a new length in the field. This field is enabled only if the data type selected in the Type list is one for which length is applicable.
Scale	Displays the scale of the selected column. To specify or change this value, type a new scale in the field. This field is enabled only if the data type selected in the Type list is one for which scale is applicable.
Add	Click this button to add the selected column definition to the data dictionary for this table. The column name is added to the end of the Column Information list.
Modify	Click this button to modify the definition of a selected column. To modify a column's definition, select the column name, edit its type, length, and scale, and click the Modify button.
Remove	Click this button to delete the selected column definition from the data dictionary for this table.

5 Click OK in the Define Table dialog box to save the table definition.

The Define File dialog box appears.

6 If necessary, repeat steps 2 through 5 to define the structure of other tables in the database.

7 Click the Cancel button in the Define File dialog box when you are finished defining the structure of all tables in the database.

You are returned to the ODBC Btrieve Driver Setup dialog box.

What to do next

For instructions on connecting to the data source, see Chapter 4, "Managing Database Connections."

Microsoft Btrieve

This section describes how to prepare and define a Btrieve data source in order to connect to it from PowerBuilder or InfoMaker using the Microsoft Btrieve ODBC driver.

Supported versions

The Microsoft Btrieve ODBC driver supports connection to Btrieve Version 5.x data sources.

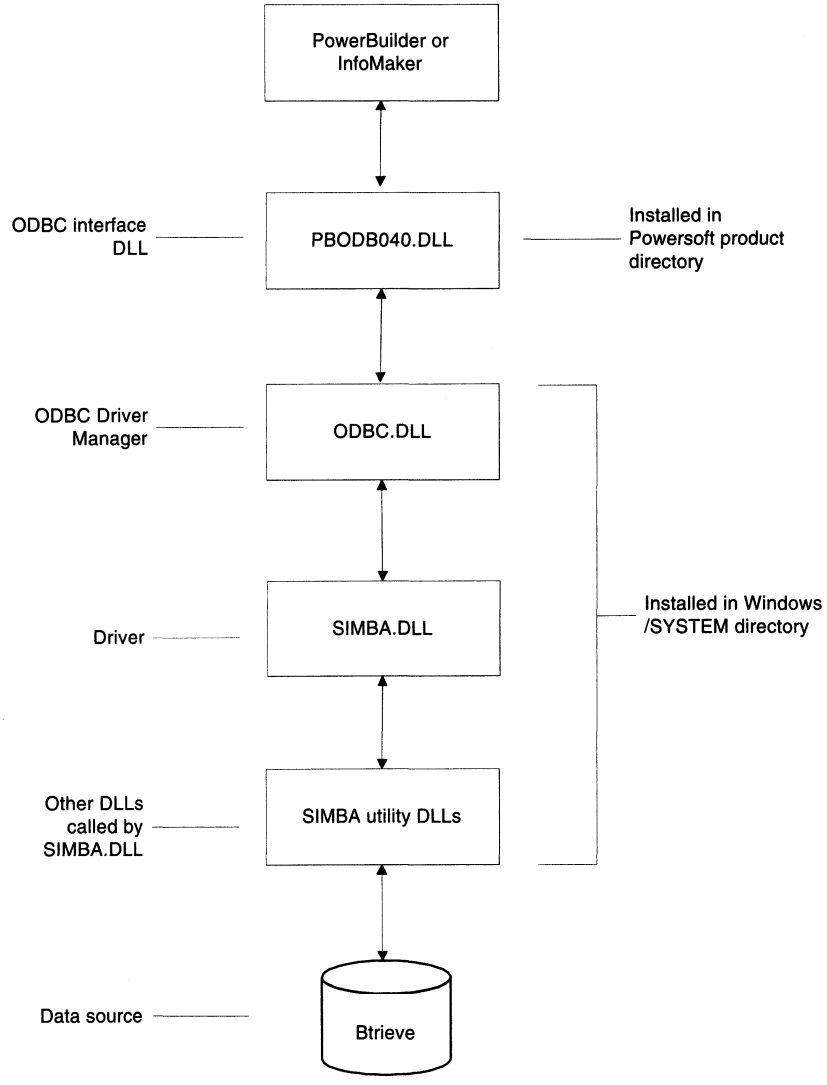
Supported SQL operations

The Microsoft Btrieve ODBC driver supports the following SQL operations with Btrieve Version 5.x data sources:

CREATE INDEX
CREATE TABLE
DELETE INDEX
DELETE TABLE
UPDATE

Basic software components

The following diagram shows the basic software components you need to connect to a Btrieve data source using the Microsoft Btrieve ODBC driver. All of these files come with PowerBuilder or InfoMaker.



Preparing to use the data source

Before you define and connect to a Btrieve data source from PowerBuilder or InfoMaker, follow these steps to prepare the data source.

❖ To prepare a Btrieve data source when using the Microsoft Btrieve driver to connect:

1 Make sure a copy of the file WBTRCALL.DLL is installed in your Windows \SYSTEM directory.

The Microsoft Btrieve ODBC driver requires the file WBTRCALL.DLL to access Btrieve data. WBTRCALL.DLL is the standalone version of the Novell Btrieve for Windows DLL. This file is supplied with PowerBuilder or InfoMaker, or you can obtain it from Novell.

> **Using WBTRCALL.DLL**
> The file WBTRCALL.DLL may be used by other programs on your computer besides Btrieve. Therefore, if you already have a copy of WBTRCALL.DLL in your Windows \SYSTEM directory, contact your system administrator to determine whether this is the correct version to use with Btrieve.

2 Make sure you have the necessary NetWare SQL data dictionary files in the same directory as the Btrieve data source you want to access.

Without the proper data dictionary files, you cannot access Btrieve data from PowerBuilder or InfoMaker.

For instructions, see "Using data dictionary files" next.

Using data dictionary files

In order to connect to a Btrieve data source from PowerBuilder or InfoMaker, you *must* have the necessary NetWare SQL data dictionary files (DDF files) in the same directory as your data source. The Microsoft Btrieve ODBC driver uses the NetWare SQL data dictionary to obtain information about the structure of your Btrieve tables. This includes information about table names, column names, data types, precision, scale, nulls, and indexes.

The data dictionary consists of the following DDF files:

File	What it stores
FILE.DDF	Information about each table in the database
FIELD.DDF	Information about each column in each table
INDEX.DDF	Information about each index

The first time you connect to a Btrieve data source in PowerBuilder or InfoMaker, the Microsoft Btrieve ODBC driver creates the DDF files automatically. These DDF files contain information about any new tables created after the DDF files were created. They do *not* contain information about existing Btrieve tables created before the DDF files.

Therefore, if you want to access existing Btrieve tables, make sure the DDF files for these tables are in the same directory as the tables. This should be the data source directory you specify when defining the Btrieve ODBC data source, as described in "Defining the data source" on page 59.

Creating your own DDF files

If you do not have the required DDF files for an existing Btrieve table, you must create them before you can access the table from PowerBuilder or InfoMaker. This is not an easy task, since you must know the exact layout and internal structure of the table in order to create DDF files for it.

Creating DDF files does not change your Btrieve file in any way. You can continue to access the file directly through any existing Btrieve application.

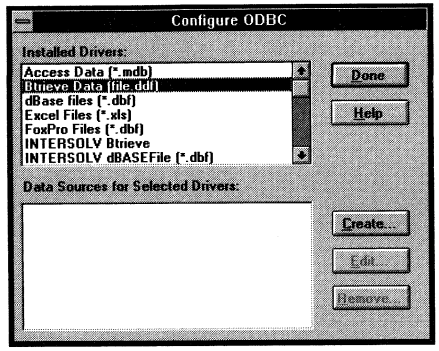
To create your own DDF files, you can use the INTERSOLV Btrieve ODBC driver to access Btrieve. The INTERSOLV driver allows you to define DDF files for existing Btrieve tables.

For information about using the INTERSOLV Btrieve driver to create DDF files, see "Defining the structure of Btrieve tables" on page 50.

Defining the data source

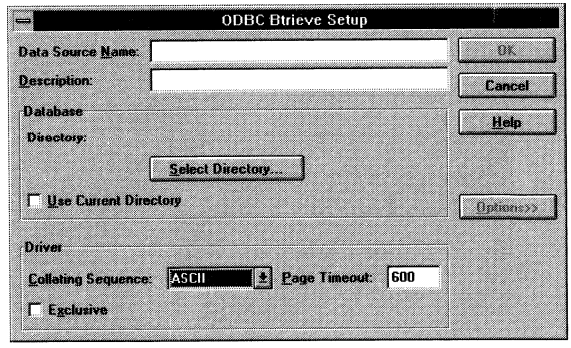
❖ To define a Btrieve data source for the Microsoft Btrieve driver:

1 Select the Microsoft Btrieve driver in the Configure ODBC dialog box.



2 Click the Create button.

The ODBC Btrieve Setup dialog box appears. (Click the Options button if necessary to display the entire dialog box.)



3 Specify values as follows:

> **For more information**
>
> For a detailed description of each field in the ODBC Btrieve Setup dialog box, click the Help button.

Field	Value
Data Source Name	A short name to identify the data source. This becomes the name of the database profile created for this data source.
Description	(Optional) A description of the data source.
Directory	Displays the drive and directory of the Btrieve data source. To select a directory and display it in the directory field, click the Select Directory button and choose a directory from the list. To make the current working directory the data source directory, select the Use Current Directory checkbox. This disables the Select Directory button.
Collating Sequence	(Optional) The sort order: ASCII or International. ASCII is the default.
Page Timeout	(Optional) The time in tenths of a second that an unused page remains in the buffer before being removed. The default is 600 (60 seconds).
Exclusive	(Optional) Select this checkbox to open Btrieve files in exclusive mode. Exclusive mode allows file access by only one user at a time, and enhances performance. Leave this checkbox deselected to open Btrieve files in shared mode. Shared mode allows file access by more than one user at a time.

4 Click OK to create the data source definition.

What to do next

For instructions on connecting to the data source, see Chapter 4, "Managing Database Connections."

INTERSOLV dBASE

This section describes how to prepare and define a data source in order to connect to it using the INTERSOLV dBASE ODBC driver.

Supported versions

The INTERSOLV dBASE ODBC driver supports connection to the following data sources:

- dBASE II, dBASE III, dBASE IV, and dBASE V files
- Clipper files
- FoxPro and FoxBASE files

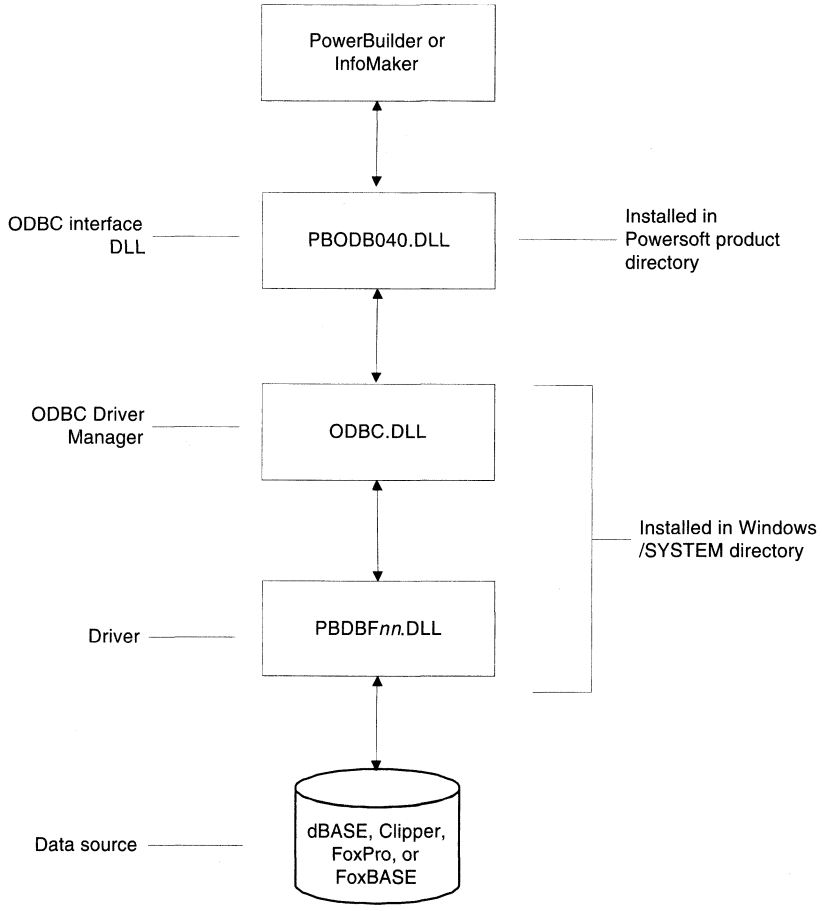
Supported SQL operations

The INTERSOLV dBASE ODBC driver supports the following SQL operations with all data sources to which it connects:

CREATE INDEX
CREATE TABLE
DELETE INDEX
DELETE TABLE
UPDATE

Basic software components

The following diagram shows the basic software components you need to connect to a data source using the INTERSOLV dBASE ODBC driver. All of these files come with PowerBuilder or InfoMaker.



Preparing to use the data source

Before you define and connect to a data source from PowerBuilder or InfoMaker, follow these steps to prepare the data source.

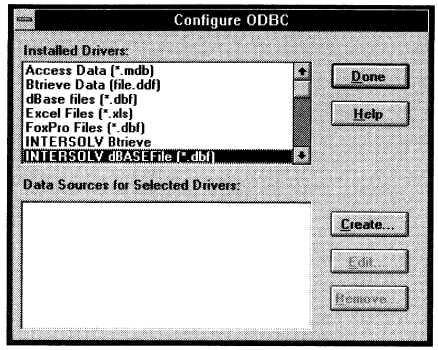
❖ To prepare a data source when using the INTERSOLV dBASE driver to connect:

- In order to update a dBASE or Clipper table, PowerBuilder or InfoMaker requires that the table have a unique index associated with it. Therefore, make sure a unique index is associated with the data source, as follows:
  - **dBASE II, dBASE III, and Clipper** Indexes for dBASE II and dBASE III files have an NDX extension; indexes for Clipper files have an NTX extension. If a dBASE II, III, or Clipper file does not have a unique index, click the Define button in the ODBC dBASE Driver Setup dialog box to assign one. (For instructions, see "Associating dBASE and Clipper files with indexes" on page 66.) You *must* do this in order for the INTERSOLV driver to use and maintain the index.
  - **dBASE IV** Indexes for dBASE IV files have an MDX extension. The INTERSOLV driver automatically associates dBASE IV files with the proper index. However, you cannot mark indexes with an MDX or CDX extension as unique. Instead, you can use the procedure for assigning an index to mark a tag within the index as unique. (For instructions, see "Associating dBASE and Clipper files with indexes" on page 66.)

Defining the data source

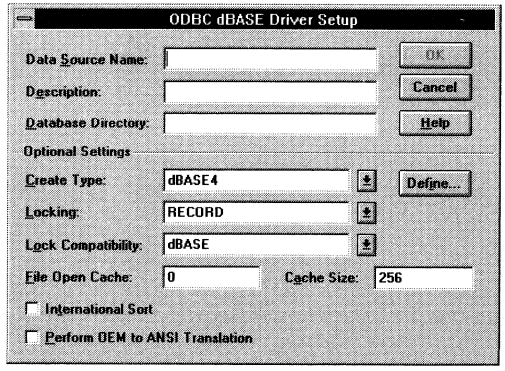
❖ To define a data source for the INTERSOLV dBASE driver:

1 Select the INTERSOLV dBASE and Clipper driver in the Configure ODBC dialog box.



2 Click the Create button.

The ODBC dBASE Driver Setup dialog box appears.



3 Specify values as follows:

For more information

For more about each field in the ODBC dBASE Driver Setup dialog box, click the Help button.

Field	Value
Data Source Name	A short name to identify the data source. This becomes the name of the database profile created for this data source.
Description	(Optional) A description of the data source.
Database Directory	The drive and directory of the dBASE data source (for example, C:\DBASE).
Create Type	(Optional) Select the type from the dropdown listbox that represents the format the driver should use to create new tables or indexes.
Locking	(Optional) Select the value from the dropdown listbox that represents the level of record locking the driver should use. Values are: ♦ **FILE** Locks all records in a table. ♦ **RECORD** Locks only those records affected by the SQL statement. ♦ **NONE** Does no record locking. It is intended for single-user environments.
Lock Compatibility	(Optional) Select the value from the dropdown listbox that represents the locking scheme the driver should use to lock records.
File Open Cache	(Optional) The maximum number of unused file opens to cache. For example, the value 3 specifies that when you open and close three tables, the driver keeps all three tables open so it does not have to reopen them if another query uses one of the tables. The default is 0, which means no file opens are cached.
Cache Size	The amount of memory, in 64K-block increments, that the driver uses to cache database records. The higher the number, the better the performance. The maximum cache size you can set depends on the system memory available.
International Sort	(Optional) Select this checkbox to use the international sort order as defined by your operating system. Leave this checkbox deselected to use the ASCII sort order.

Field	Value
Perform OEM to ANSI Translation	(Optional) Select this checkbox if the database file stores data in the IBM PC character set. Leave this checkbox deselected if the database file stores data in the ANSI character set. This is the default.
Define	Displays the Define File dialog box to associate dBASE and Clipper files with indexes. For more information, see "Associating dBASE and Clipper files with indexes" next.

4 Click OK to create the data source definition.

Associating dBASE and Clipper files with indexes

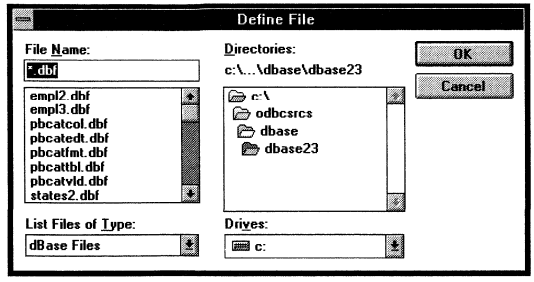
In order to update a dBASE or Clipper table, PowerBuilder or InfoMaker requires that the table have a unique index associated with it, as follows:

- **dBASE II, dBASE III, and Clipper** The INTERSOLV dBASE driver does *not* automatically associate an index with a dBASE II, dBASE III, or Clipper table. Therefore, to access one of these tables that does not have a unique index, you must associate an index with it by following the procedure below.
- **dBASE IV** The INTERSOLV dBASE driver automatically associates a dBASE IV table, which uses an MDX index format, with the proper index. However, if the index you are using has an MDX or CDX extension, you cannot mark it as unique. Therefore, although the following procedure for associating an index is not required for dBASE IV tables, you can use it as a way to mark tags within the index as unique. (For instructions, see the Tag field description on page 68.)

❖ To associate an index with a dBASE or Clipper table:

1 Click the Define button in the ODBC Driver Setup dialog box.

The Define File dialog box appears.

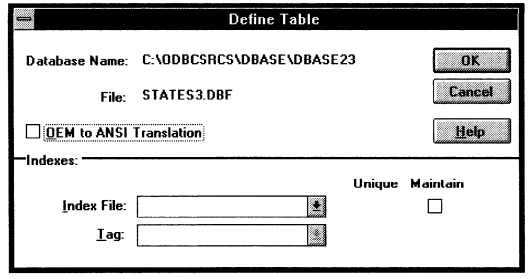


2 Select the DBF file to which you want to associate an index.

When you access a dBASE or Clipper data source from PowerBuilder or InfoMaker, a directory represents the database and each DBF file in the directory represents a table in the database.

3 Click OK.

The Define Table dialog box appears. The top half of the dialog box displays the database directory and file you selected.



4 Specify values as follows:

Field	Value
OEM to ANSI Translation	Select this checkbox if the database file stores data in the IBM PC character set. Leave this checkbox deselected if the database file stores data in the ANSI character set. This is the default.
Index File	Select an index from the dropdown listbox to associate with the selected table.

Field	Value
Unique	Select this checkbox to specify that the selected index is unique. Otherwise, leave it deselected. If this is a dBASE IV index (MDX extension), you cannot mark it as unique. Instead, you can mark tags within a dBASE IV index as unique. (For instructions, see the Tag field description later in this table.)
Maintain	Select this checkbox to specify that the index should be maintained. Otherwise, leave it deselected.
Tag	To mark a dBASE IV index tag as unique, select a tag name from the dropdown listbox. Then select the Unique checkbox that appears to the right of the tag name to specify that the tag is unique. If this is a dBASE II or dBASE III index (NDX extension) or a Clipper index (NTX extension), the Tag field is disabled.

5 Click OK in the Define Table dialog box to save the table/index association.

The Define File dialog box appears.

6 Repeat steps 2 through 5 to associate all files in the database with indexes.

7 Click the Cancel button in the Define File dialog box when you are finished associating all files with indexes.

You are returned to the ODBC dBASE Driver Setup dialog box.

What to do next

For instructions on connecting to the data source, see Chapter 4, "Managing Database Connections."

Microsoft dBASE

This section describes how to prepare and define a dBASE data source in order to connect to it using the Microsoft dBASE ODBC driver.

Supported versions

The Microsoft dBASE ODBC driver supports connection to dBASE Version III and IV data sources.

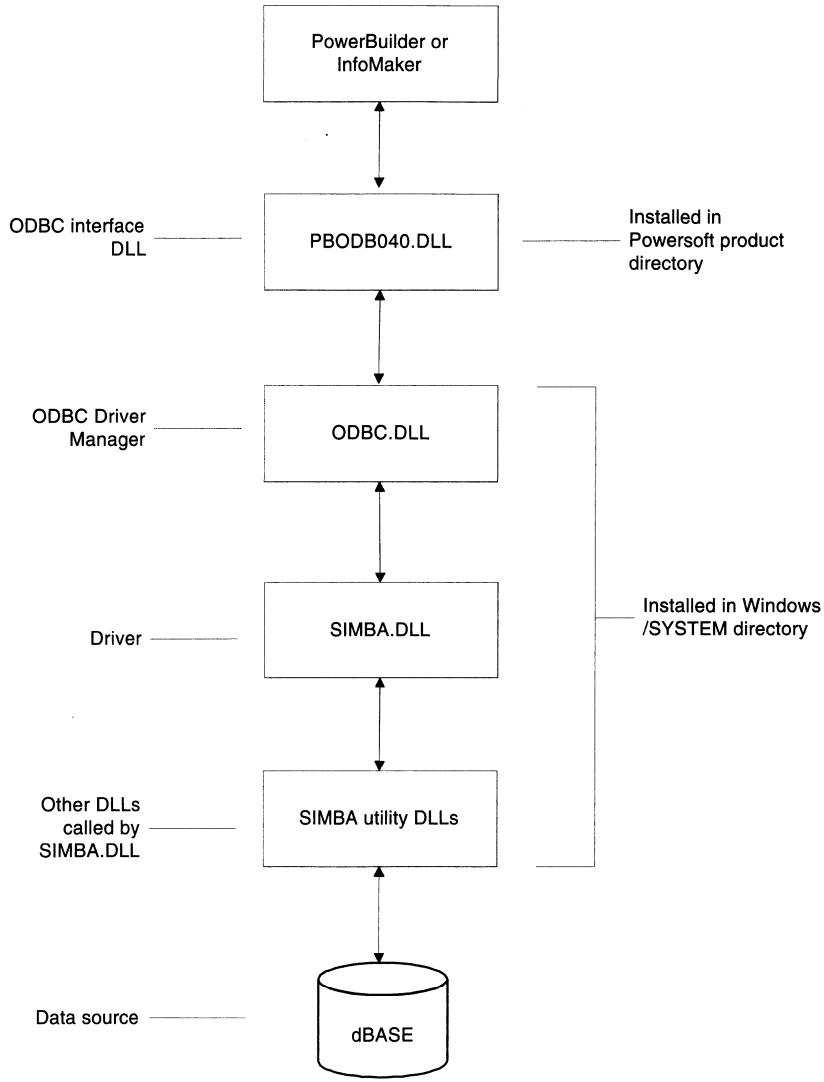
Supported SQL operations

The Microsoft dBASE ODBC driver supports the following SQL operations with dBASE III and IV data sources:

CREATE INDEX
CREATE TABLE
DELETE INDEX
DELETE TABLE
UPDATE

Basic software components

The following diagram shows the basic software components you need to connect to a dBASE data source using the Microsoft dBASE ODBC driver. All of these files come with PowerBuilder or InfoMaker.



Preparing to use the data source

Before you define and connect to a dBASE data source from PowerBuilder or InfoMaker, follow these steps to prepare the data source.

❖ To prepare a dBASE data source when using the Microsoft dBASE driver to connect:

- Make sure each dBASE file is associated with a unique index.

  In order to update a dBASE table, PowerBuilder or InfoMaker requires that the table have a unique index associated with it, as follows:

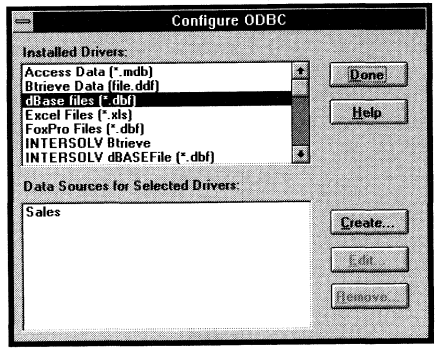
  - **dBASE III** Indexes for dBASE III files have an NDX extension. If a dBASE III file does not have a unique index, click the Select Indexes button in the ODBC dBASE Setup dialog box to assign one. (For instructions, see "Defining the data source" next.) You *must* do this in order for the Microsoft driver to use and maintain the index.
  - **dBASE IV** Indexes for dBASE IV files have an MDX extension. The Microsoft driver automatically associates dBASE IV files with the proper index.

Clipper index files
The Microsoft dBASE driver does *not* support the NTX index structure used by Clipper files. If you need to access a Clipper data source, use the INTERSOLV dBASE driver, described on page 61.

Defining the data source

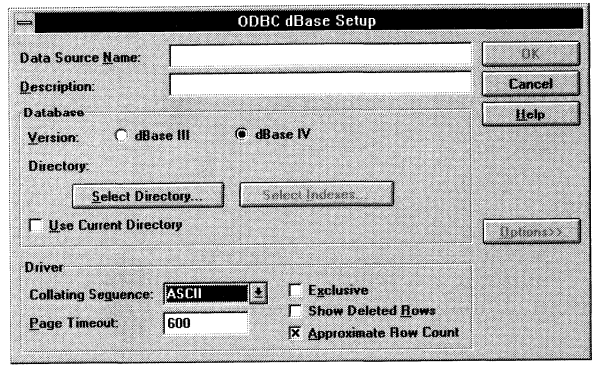
❖ To define a dBASE data source for the Microsoft dBASE driver:

1 Select the Microsoft dBASE driver in the Configure ODBC dialog box.



2 Click the Create button.

The ODBC dBASE Setup dialog box appears. (Click the Options button if necessary to display the entire dialog box.)



3 Specify values as follows:

> **For more information**
>
> For a detailed description of each field in the ODBC dBASE Setup dialog box, click the Help button.

Field	Value
Data Source Name	A short name to identify the data source. This becomes the name of the database profile created for this data source.
Description	(Optional) A description of the data source.
Version	Click the radio button representing the version of your dBASE data source. To help you determine your version of dBASE, keep in mind that dBASE III files use indexes with an NDX extension, and dBASE IV files use indexes with an MDX extension.
Directory	Displays the drive and directory of the dBASE data source. To select a directory and display it in the Directory field, click the Select Directory button and choose a directory from the list. Selecting a directory enables the Select Indexes button. To make the current working directory the data source directory, check Use Current Directory. This disables the Select Directory button.
Select Indexes	Displays the Select Indexes dialog box to associate dBASE files with indexes. The Select Indexes button is enabled after you use the Select Directory button to select a data source directory. For instructions, see "Associating dBASE files with indexes" on page 74.
Collating Sequence	(Optional) The sort order: ASCII or International. ASCII is the default.
Page Timeout	(Optional) The time in tenths of a second that an unused page remains in the buffer before being removed. The default is 600 (60 seconds).
Exclusive	(Optional) Select this checkbox to open dBASE files in exclusive mode. Exclusive mode allows file access by only one user at a time, and enhances performance. Leave this checkbox deselected to open dBASE files in shared mode. Shared mode allows file access by more than one user at a time.

Field	Value
Show Deleted Rows	(Optional) Select this checkbox to display rows marked as deleted. Leave this checkbox deselected if you do not want to display rows marked as deleted.
Approximate Row Count	(Optional) Leave this checkbox selected to approximate table size statistics. Deselect this checkbox if you do not want to approximate table size statistics.

4 Click OK to create the data source definition.

Associating dBASE files with indexes

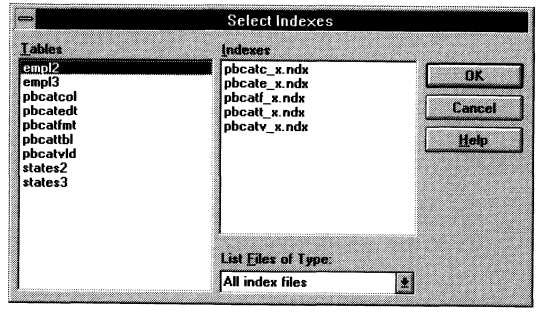
In order to update a dBASE table, PowerBuilder or InfoMaker requires that the table have a unique index associated with it, as follows:

- **dBASE III** The Microsoft dBASE driver does *not* automatically associate an index with a dBASE III table. Therefore, to access a dBASE III table that does not have a unique index, you must associate an index with it by following the procedure below.
- **dBASE IV** The Microsoft dBASE driver automatically associates a dBASE IV table with the proper index. Therefore, the following procedure for associating an index is *not required* for dBASE IV tables.

❖ To associate an index with a dBASE III table:

1 Click the Select Indexes button in the ODBC dBASE Setup dialog box. (The Select Indexes button is enabled after you use the Select Directory button to select a data source directory.)

The Select Indexes dialog box appears. The Tables list displays the files in the currently selected data source directory. The Indexes list displays the indexes assigned to the currently selected file.



2 Select a file from the Tables list.

3 Select an index from the Indexes list to associate with this file.

4 Click OK to save the table/index association.

What to do next

For instructions on connecting to the data source, see Chapter 4, "Managing Database Connections."

Microsoft Excel

This section describes how to prepare and define an Excel data source in order to connect to it using the Microsoft Excel ODBC driver.

Supported versions

The Microsoft Excel ODBC driver supports connection to Excel Version 3.x and 4.x data sources.

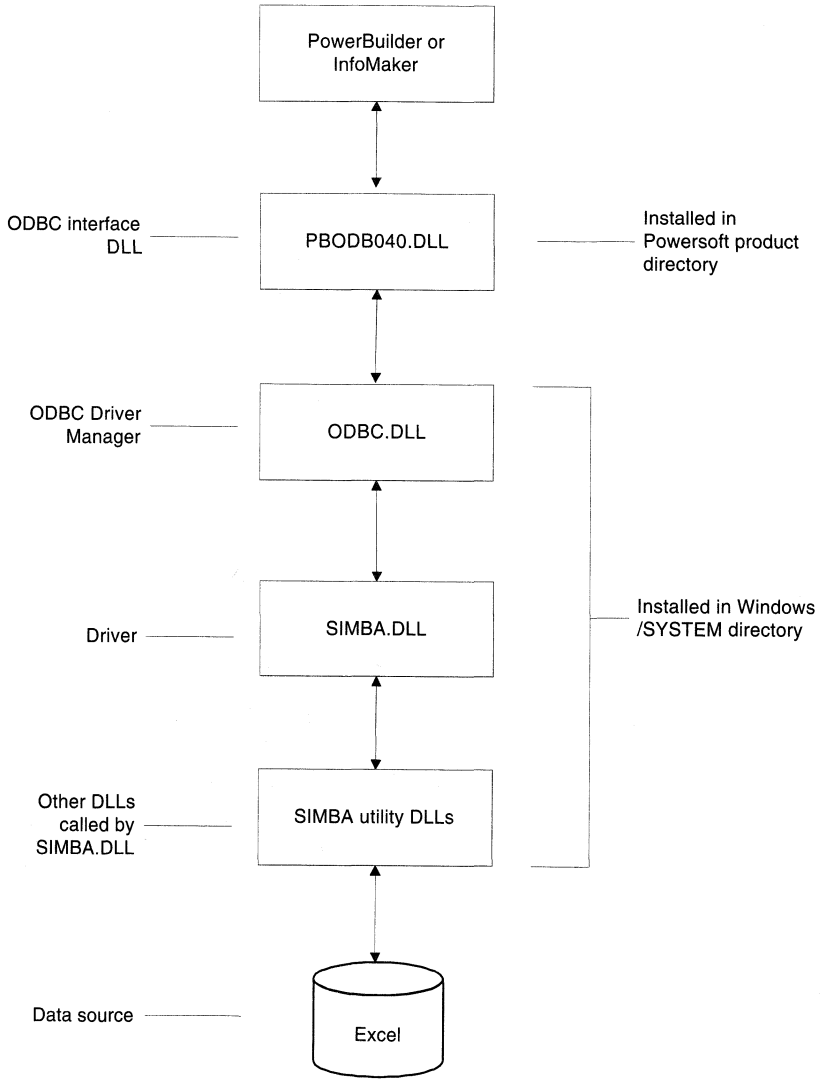
Read-only Excel data
When you access an Excel data source with the Microsoft Excel ODBC driver, Excel data is read-only. You cannot create Excel databases or modify the data in them with PowerBuilder or InfoMaker.

Supported SQL operations

Because Excel data is read-only when you access it with the Microsoft Excel ODBC driver, no SQL operations are supported.

Basic software components

The following diagram shows the basic software components you need to connect to an Excel data source using the Microsoft Excel ODBC driver. All of these files come with PowerBuilder or InfoMaker.



Preparing to use the data source

In order for PowerBuilder or InfoMaker to access data from an Excel Version 3.x or 4.x worksheet, you must first define a database range for the worksheet. In an Excel worksheet, a **database range** is a rectangular range of cells defined as a database. Only those Excel worksheets with a defined database range will appear in the list of tables for the data source in PowerBuilder or InfoMaker.

Before you define and connect to an Excel data source from PowerBuilder or InfoMaker, follow these steps to prepare the data source.

❖ To prepare an Excel Version 3.x or 4.x data source:

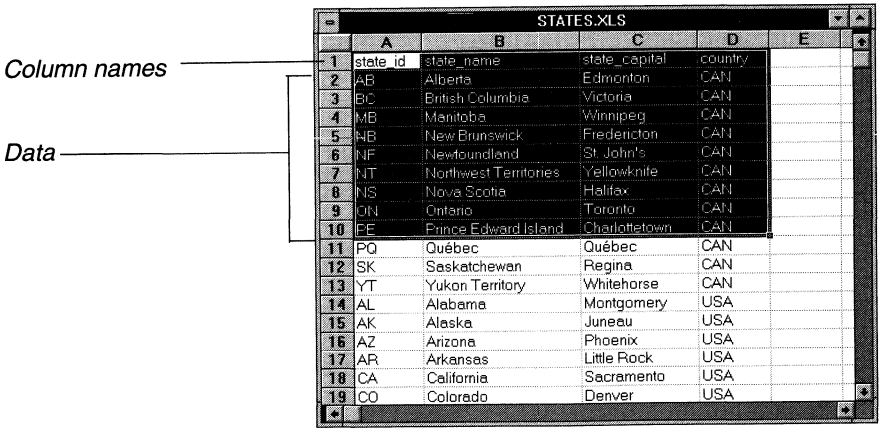
1 Start Excel on your computer.

2 Open the Excel worksheet you want to use as a data source.

3 Select the range of cells you want to include in the database range for this worksheet.

The first row of cells you select contains the column (field) names. Subsequent rows you select contain data for the columns.

In the following example of an Excel worksheet, the database range is the highlighted area including cells A1 through D10. The cells in this range will display when you open this worksheet as a table in PowerBuilder or InfoMaker.



	A	B	C	D	E
1	state_id	state_name	state_capital	country	
2	AB	Alberta	Edmonton	CAN	
3	BC	British Columbia	Victoria	CAN	
4	MB	Manitoba	Winnipeg	CAN	
5	NB	New Brunswick	Fredericton	CAN	
6	NF	Newfoundland	St. John's	CAN	
7	NT	Northwest Territories	Yellowknife	CAN	
8	NS	Nova Scotia	Halifax	CAN	
9	ON	Ontario	Toronto	CAN	
10	PE	Prince Edward Island	Charlottetown	CAN	
11	PQ	Québec	Québec	CAN	
12	SK	Saskatchewan	Regina	CAN	
13	YT	Yukon Territory	Whitehorse	CAN	
14	AL	Alabama	Montgomery	USA	
15	AK	Alaska	Juneau	USA	
16	AZ	Arizona	Phoenix	USA	
17	AR	Arkansas	Little Rock	USA	
18	CA	California	Sacramento	USA	
19	CO	Colorado	Denver	USA	

If any empty cells appear in the database range in place of column names, these cells will appear as columns when you open the worksheet as a table in PowerBuilder or InfoMaker. However, they will have names such as Field1 or Field2 assigned instead of actual column names.

4 Select Data➤Set Database from the menu bar to define the current worksheet selection as a database range.

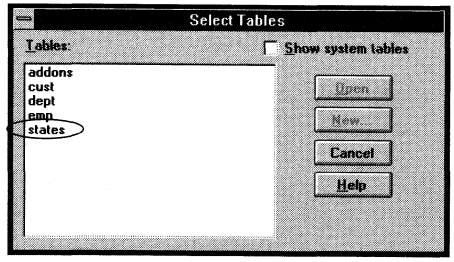
For instructions on using the Set Database command, see your Excel documentation.

5 Save your changes to the Excel worksheet.

What happens

When you connect to the Excel data source containing this worksheet in PowerBuilder or InfoMaker:

- The name of the worksheet (without the XLS extension) appears on the table list for this data source. For example, the Excel file STATES.XLS appears as a table named States in the Select Tables dialog box.

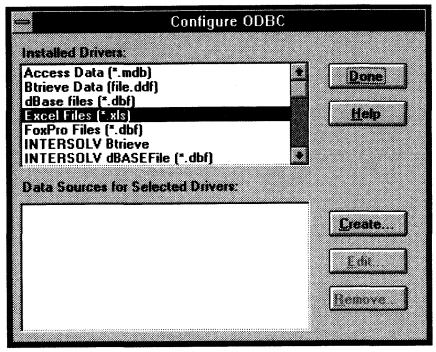


- The rows and columns you included in the database range for this worksheet display when you view the data in the table.

Defining the data source

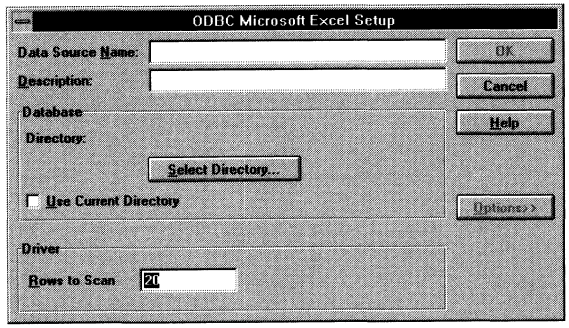
❖ To define an Excel data source for the Microsoft Excel driver:

1 Select the Microsoft Excel driver in the Configure ODBC dialog box.



2 Click the Create button.

The ODBC Microsoft Excel Setup dialog box appears. (Click the Options button if necessary to display the entire dialog box.)



3 Specify values as follows:

For more information

For a detailed description of each field in the ODBC Microsoft Excel Setup dialog box, click the Help button.

Field	Value
Data Source Name	A short name to identify the data source. This becomes the name of the database profile created for this data source.
Description	(Optional) A description of the data source.
Directory	Displays the drive and directory of the Excel data source. To select a directory and display it in the Directory field, click the Select Directory button and choose a directory from the list. To make the current working directory the data source directory, select the Use Current Directory checkbox. This disables the Select Directory button.
Rows to Scan	The number of rows you want the driver to scan to determine the data type of each column. To scan the entire file, enter 0 (zero). The default is 20 rows.

4 Click OK to create the data source definition.

What to do next

For instructions on connecting to the data source, see Chapter 4, "Managing Database Connections."

Microsoft FoxPro

This section describes how to prepare and define a FoxPro data source in order to connect to it using the Microsoft FoxPro ODBC driver.

Supported versions

The Microsoft FoxPro ODBC driver supports connection to the following data sources:

- FoxPro for DOS Version 2.x
- FoxPro for Windows

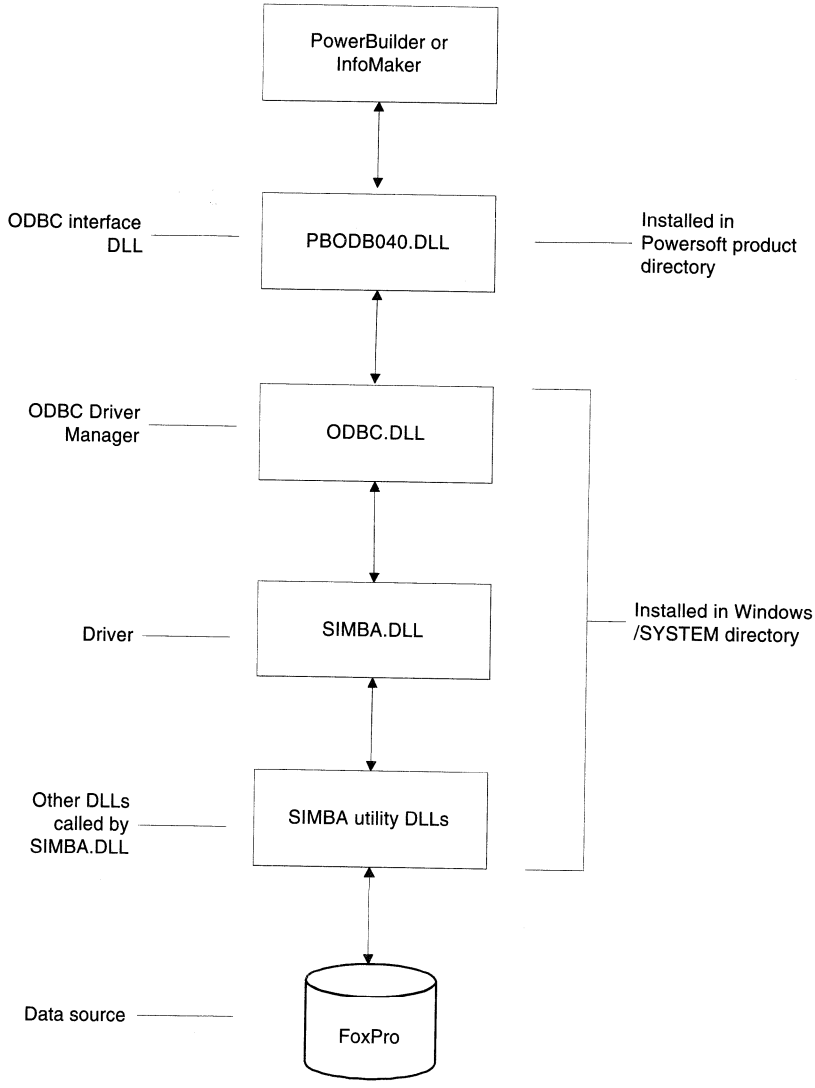
Supported SQL operations

The Microsoft FoxPro ODBC driver supports the following SQL operations with FoxPro for DOS Version 2.x data sources:

CREATE INDEX
CREATE TABLE
DELETE INDEX
DELETE TABLE
UPDATE

Basic software components

The following diagram shows the basic software components you need to connect to a FoxPro data source using the Microsoft FoxPro ODBC driver. All of these files come with PowerBuilder or InfoMaker.



Preparing to use the data source

Before you define and connect to a FoxPro data source from PowerBuilder or InfoMaker, follow these steps to prepare the data source.

❖ To prepare a FoxPro data source:

1 Make sure each FoxPro file is associated with a unique index.

In order to update a FoxPro table, PowerBuilder or InfoMaker requires that the table have a unique index associated with it, as follows:

- **FoxPro Version 2.0** Indexes for FoxPro 2.0 files have an IDX extension. If a FoxPro 2.0 file does not have a unique index, click the Select Indexes button in the ODBC FoxPro Setup dialog box to assign one. (For instructions, see "Defining the data source" on page 85.) You *must* do this in order for the Microsoft driver to use and maintain the index.
- **FoxPro Version 2.5** Indexes for FoxPro 2.5 files have a CDX extension. The Microsoft driver automatically associates FoxPro 2.5 files with the proper index.

2 Issue the MS-DOS SHARE command in either of the following ways:

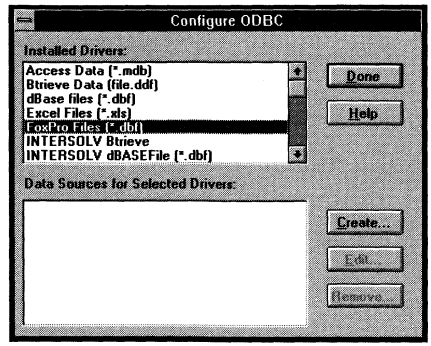
- Type the command from an MS-DOS prompt.
- Include the command line in your AUTOEXEC.BAT file.

3 Start Windows on your computer.

Defining the data source

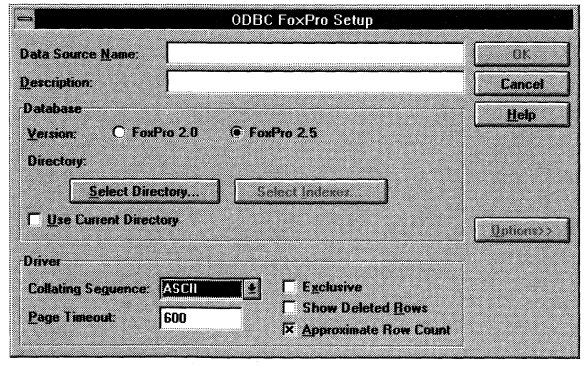
❖ To define a FoxPro data source for the Microsoft FoxPro driver:

1 Select the Microsoft FoxPro driver in the Configure ODBC dialog box.



2 Click the Create button.

The ODBC FoxPro Setup dialog box appears. (Click the Options button if necessary to display the entire dialog box.)



3 Specify values as follows:

For more information

For a detailed description of each field in the ODBC FoxPro Setup dialog box, click the Help button.

Field	Value
Data Source Name	A short name to identify the data source. This becomes the name of the database profile created for this data source.
Description	(Optional) A description of the data source.
Version	Click the radio button representing the version of your FoxPro data source. To help you determine your version of FoxPro, keep in mind that FoxPro 2.0 files use indexes with an IDX extension, and FoxPro 2.5 files use indexes with a CDX extension.
Directory	Displays the drive and directory of the FoxPro data source. To select a directory and display it in the Directory field, click the Select Directory button and choose a directory from the list. Selecting a directory enables the Select Indexes button. To make the current working directory the data source directory, select the Use Current Directory checkbox. This disables the Select Directory button.
Select Indexes	Displays the Select Indexes dialog box to associate FoxPro files with indexes. The Select Indexes button is enabled after you use the Select Directory button to select a data source directory. For instructions, see "Associating FoxPro files with indexes" on page 87.
Collating Sequence	(Optional) The sort order: ASCII or International. ASCII is the default.
Page Timeout	(Optional) The time in tenths of a second that an unused page remains in the buffer before being removed. The default is 600 (60 seconds).
Exclusive	(Optional) Select this checkbox to open FoxPro files in exclusive mode. Exclusive mode allows file access by only one user at a time, and enhances performance. Leave this checkbox deselected to open FoxPro files in shared mode. Shared mode allows file access by more than one user at a time.

Field	Value
Show Deleted Rows	(Optional) Select this checkbox to display rows marked as deleted. Leave this checkbox deselected if you do not want to display rows marked as deleted.
Approximate Row Count	(Optional) Leave this checkbox selected to approximate table size statistics. Deselect this checkbox if you do not want to approximate table size statistics.

4 Click OK to create the data source definition.

Associating FoxPro files with indexes

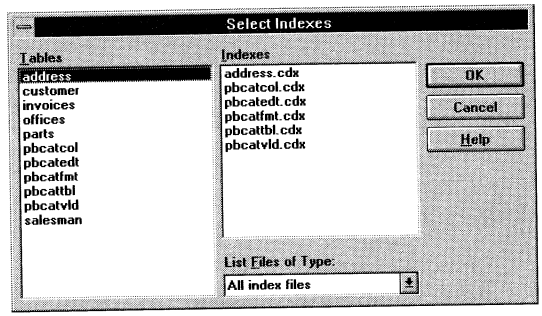
In order to update a FoxPro table, PowerBuilder or InfoMaker requires that the table have a unique index associated with it, as follows:

- **FoxPro 2.0** The Microsoft FoxPro driver does *not* automatically associate an index with a FoxPro 2.0 table. Therefore, to access a FoxPro 2.0 table that does not have a unique index, you must associate an index with it by following the procedure below.
- **FoxPro 2.5** The Microsoft FoxPro driver automatically associates a FoxPro 2.5 table with the proper index. Therefore, the following procedure for associating an index is *not required* for FoxPro 2.5 tables.

❖ To associate an index with a FoxPro 2.0 table:

1 Click the Select Indexes button in the ODBC FoxPro Setup dialog box. (The Select Indexes button is enabled after you use the Select Directory button to select a data source directory.)

The Select Indexes dialog box appears. The Tables list displays the files in the currently selected data source directory. The Indexes list displays the indexes assigned to the currently selected file.



2 Select a file from the Tables list.

3 Select an index from the Indexes list to associate with this file.

4 Click OK to save the table/index association.

What to do next

For instructions on connecting to the data source, see Chapter 4, "Managing Database Connections."

INTERSOLV NetWare SQL

This section describes how to prepare and define a NetWare SQL data source in order to connect to it using the INTERSOLV NetWare SQL ODBC driver.

Supported versions

The INTERSOLV NetWare SQL ODBC driver supports connection to NetWare SQL databases running in a Windows environment.

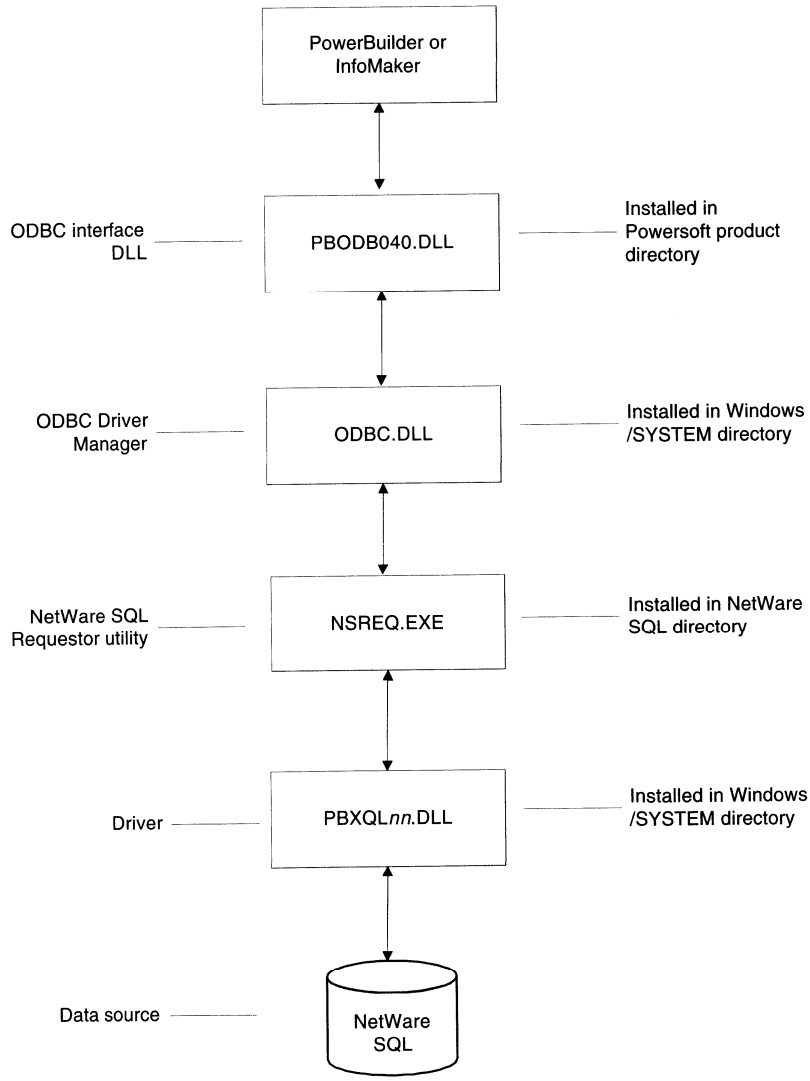
Supported SQL operations

The INTERSOLV NetWare SQL ODBC driver supports the following SQL operations with NetWare SQL Version 3.x data sources:

ALTER TABLE
CREATE INDEX
CREATE TABLE
CREATE VIEW
DELETE INDEX
DELETE TABLE
UPDATE

Basic software components

The following diagram shows the basic software components you need to connect to a NetWare SQL data source using the INTERSOLV NetWare SQL ODBC driver. All of these files come with PowerBuilder or InfoMaker *except* for the NetWare SQL Requestor utility.



NetWare SQL Requestor utility (NSREQ.EXE)
To access a NetWare SQL data source from PowerBuilder or InfoMaker, you must first install and start the NetWare SQL Requestor Utility (NSREQ.EXE) on your computer or network server *before* starting Windows and invoking PowerBuilder or InfoMaker.

Powersoft does not supply the NetWare SQL Requestor utility with its products. You must purchase it from Novell as part of the NetWare SQL Requestor software.

Preparing to use the data source

Before you define and connect to a NetWare SQL data source from PowerBuilder or InfoMaker, follow these steps to prepare the data source.

❖ To prepare a NetWare SQL data source:

1 Install the NetWare SQL Requestor utility (NSREQ.EXE) on your computer or network sever.

2 Start the NetWare SQL Requestor utility with a minimum data size of 4096, which is the default value.

For example, to start NSREQ.EXE with a data size of 32 K, type the following at a DOS prompt:

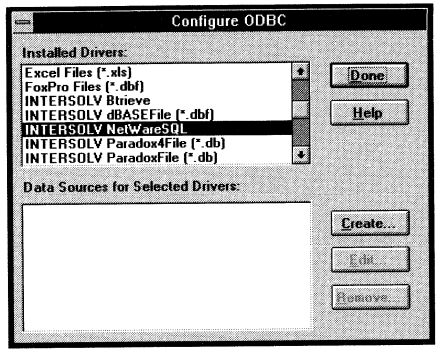
```
NSREQ /D: 32767
```

3 Start Windows on your computer.

Defining the data source

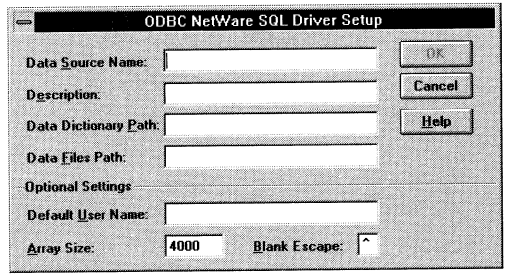
❖ To define a NetWare SQL data source for the INTERSOLV NetWare SQL driver:

1 Select the INTERSOLV NetWare SQL driver in the Configure ODBC dialog box.



2 Click the Create button.

The ODBC NetWare SQL Driver Setup dialog box appears.



3 Specify values as follows:

> **For more information**
>
> For more about each field in the ODBC NetWare SQL Driver Setup dialog box, click the Help button.

Field	Value
Data Source Name	A short name to identify the data source. This becomes the name of the database profile created for this data source.
Description	(Optional) A description of the data source.
Data Dictionary Path	The drive and directory of the NetWare SQL data dictionary files containing the tables you want to access (for example, C:\NETWARE\EXAMPLES). This may be the same as the Data Files Path.
Data Files Path	The drive and directory of the data files you want to access (for example, C:\NETWARE\EXAMPLES). This may be the same as the Data Dictionary Path.
Default User Name	(Optional) The default user name used to connect to your NetWare SQL database. A user name is required only if security is enabled in your database.
Array Size	(Optional) The number of bytes the driver uses to fetch multiple rows. The value can be between 0 and 65,536 bytes.
Blank Escape	(Optional) Specifies the escape character you want to use to replace blanks in table or field names. Valid escape characters are carat (^), tilde (~), or underscore (_).

4 Click OK to create the data source definition.

What to do next

For instructions on connecting to the data source, see Chapter 4, "Managing Database Connections."

INTERSOLV Paradox 4

This section describes how to prepare and define a Paradox data source in order to connect to it using the INTERSOLV Paradox 4 ODBC driver.

Supported versions

The INTERSOLV Paradox 4 ODBC driver supports connection to Paradox Version 3.x and 4.x data sources.

Supported SQL operations

The INTERSOLV Paradox 4 ODBC driver supports the following SQL operations with Paradox Version 3.x and 4.x data sources.

Note that for the CREATE INDEX operation, the unique index *must* be named Primary.

CREATE INDEX

CREATE TABLE

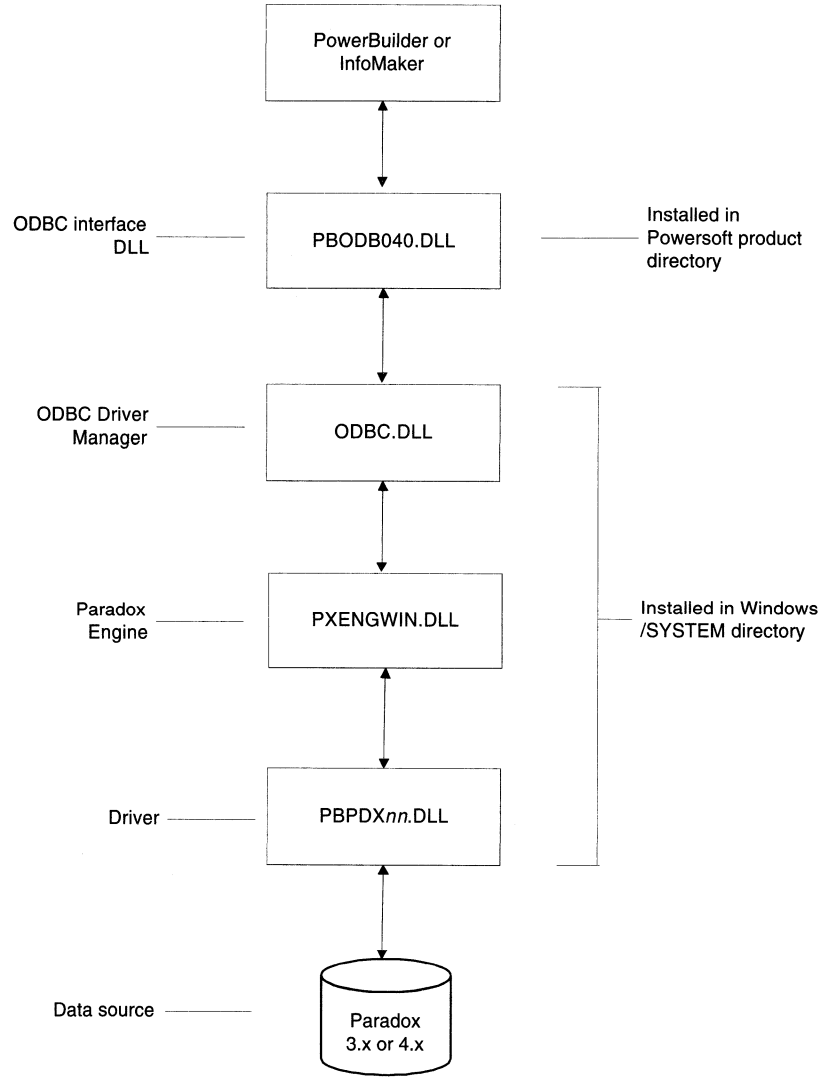
DELETE INDEX

DELETE TABLE

UPDATE

Basic software components

The following diagram shows the basic software components you need to access a Paradox data source using the INTERSOLV Paradox 4 ODBC driver. All of these files come with PowerBuilder or InfoMaker *except* for the Paradox Engine.



Paradox Engine (PXENGWIN.DLL)
To access a Paradox data source with the INTERSOLV Paradox 4 ODBC driver, you *must* first install the Windows version of the Paradox Engine (PXENGWIN.DLL) on your computer or network server. Without this DLL, you will be unable to use this driver to open Paradox tables in PowerBuilder or InfoMaker.

Powersoft does not supply the Paradox Engine with its products. You must purchase Version 3.0 or higher of the Paradox Engine from Borland.

Preparing to use the data source

Before you define and connect to a Paradox data source from PowerBuilder or InfoMaker, follow these steps to prepare the data source.

❖ To prepare a Paradox data source when using the INTERSOLV Paradox 4 driver to connect:

1 Install the Paradox Engine (PXENGWIN.DLL) on your computer or network server.

If you do not install the Paradox Engine, the following message appears when you attempt to define the Paradox data source:

> The setup routines for the INTERSOLV Paradox4File (*.db) ODBC driver could not be loaded. You may be low on memory and need to quit a few applications.

2 Make sure the directory containing the Paradox Engine appears in your program search path.

3 Make sure the column names in your data source do not contain any Paradox reserved words.

If a column name contains a Paradox reserved word, an error message may appear when you try to access this table in PowerBuilder or InfoMaker. If this occurs, rename the column and reconnect.

For more about Paradox reserved words, see your Paradox documentation.

4 Issue the MS-DOS SHARE command in either of the following ways:

- Type the command from an MS-DOS prompt.
- Include the command line in your AUTOEXEC.BAT file.

5 Start Windows on your computer.

Preparing a shared Paradox data source

In addition to accessing a local Paradox data source, you can also access shared Paradox data sources from PowerBuilder or InfoMaker. Examples of shared Paradox data sources are:

- Paradox tables residing in a shared network directory
- Paradox tables residing on your computer and shared among many applications

❖ To access a shared Paradox data source:

1 Set up a shared directory on your network and configure it to give appropriate access to all users.

For instructions, see your system or network administrator.

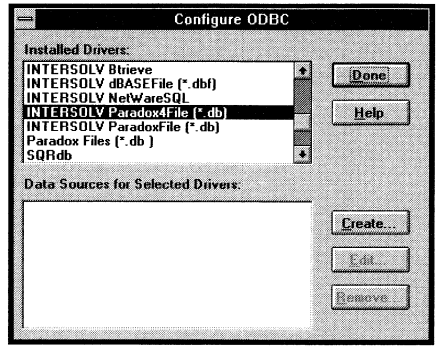
2 Use the Paradox Engine configuration utility (PXENGCFG.EXE) to specify the proper file-sharing characteristics for the tables accessed by Paradox.

For instructions, see your Paradox documentation.

Defining the data source

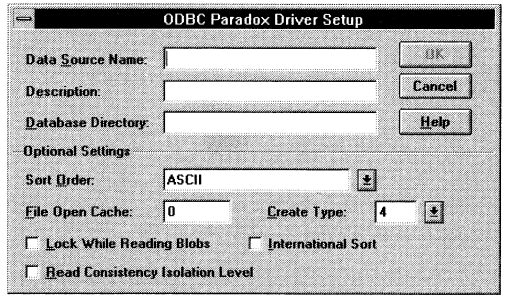
❖ To define a Paradox data source for the INTERSOLV Paradox 4 driver:

1 Select the INTERSOLV Paradox 4 driver in the Configure ODBC dialog box.



2 Click the Create button.

The ODBC Paradox Driver Setup dialog box appears.



3 Specify values as follows:

> **For more information**
> For more about each field in the ODBC Paradox Driver Setup dialog box, click the Help button.

Field	Value
Data Source Name	A short name to identify the data source. This becomes the name of the database profile created for this data source.
Description	(Optional) A description of the data source.
Database Directory	The drive and directory of the Paradox data source (for example, C:\PARADOX).
Sort Order	(Optional) The order in which records are sorted in a Paradox index. ASCII is the default.
File Open Cache	(Optional) The maximum number of unused file opens to cache. For example, the value 3 specifies that when you open and close three tables, the driver keeps all three tables open so it does not have to reopen them if another query uses one of the tables. The default is 0, which means no file opens are cached.
Create Type	(Optional) The version the driver uses to create tables. Select **3** to create Paradox 3.x tables, or **4** to create Paradox 4.x tables.
Lock While Reading Blobs	(Optional) Select this checkbox to lock binary large object (BLOB) data types before they are fetched.
Read Consistency Isolation Level	(Optional) Select this checkbox to lock tables to ensure that all records can be reread within a statement.
International Sort	(Optional) Select this checkbox to use the international sort order as defined by your operating system.

4 Click OK to create the data source definition.

About creating an index for Paradox tables

When you create the primary (unique) index for a Paradox table accessed with the INTERSOLV Paradox 4 ODBC driver, you *must* name the index Primary. You can then create other non primary (duplicate) indexes for the table as needed.

For instructions on creating indexes, see the PowerBuilder or InfoMaker *User's Guide*.

What to do next

For instructions on connecting to the data source, see Chapter 4, "Managing Database Connections."

INTERSOLV Paradox 5

This section describes how to prepare and define a Paradox data source in order to connect to it using the INTERSOLV Paradox 5 ODBC driver.

Supported versions

The INTERSOLV Paradox 5 ODBC driver supports connection to Paradox Version 3.x, 4.x, and 5.x data sources.

Supported SQL operations

The INTERSOLV Paradox 5 ODBC driver supports the following SQL operations with Paradox Version 3.x, 4.x, and 5.x data sources.

Note that for the CREATE INDEX operation, the unique index *must* be named Primary.

CREATE INDEX

CREATE TABLE

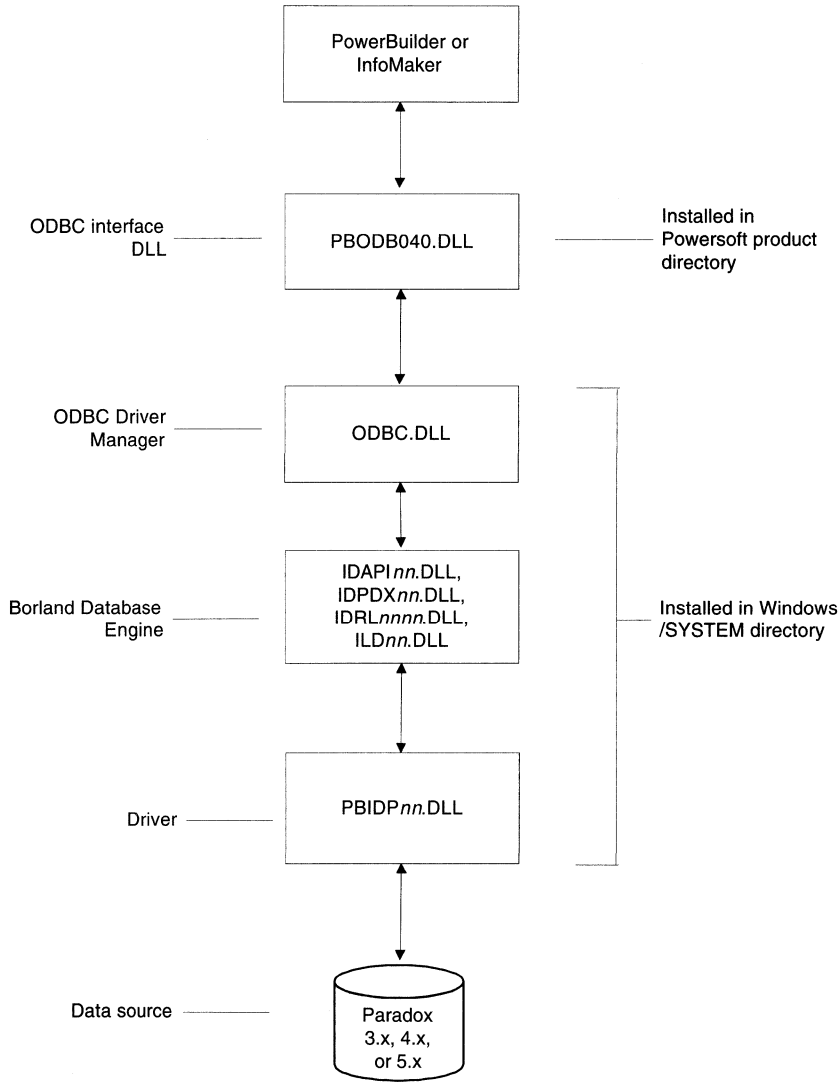
DELETE INDEX

DELETE TABLE

UPDATE

Basic software components

The following diagram shows the basic software components you need to access a Paradox data source using the INTERSOLV Paradox 4 ODBC driver. All of these files come with PowerBuilder or InfoMaker *except* for the Borland Database Engine.



Borland Database Engine

To access a Paradox data source with the INTERSOLV Paradox 5 ODBC driver, you *must* first install the Borland Database Engine, which conforms to the IDAPI programming interface. Without the Borland Database Engine, you will be unable to use this driver to open Paradox tables in PowerBuilder or InfoMaker.

Powersoft does not supply the Borland Database Engine with its products. If you have installed the most recent version of Paradox for Windows or dBASE for Windows, then you should already have the necessary files. These files include:

- IDAPI*nn*.DLL
- IDPDX*nn*.DLL
- IDRL*nnnn*.DLL
- ILD*nn*.DLL

Preparing to use the data source

Before you define and connect to a Paradox data source from PowerBuilder or InfoMaker, follow these steps to prepare the data source.

❖ To prepare a Paradox data source when using the INTERSOLV Paradox 5 driver to connect:

1 Install the Borland Database Engine on your computer or network server. Make sure you have the following files:

- IDAPI*nn*.DLL
- IDPDX*nn*.DLL
- IDRL*nnnn*.DLL
- ILD*nn*.DLL

If you do not install the Borland Database Engine, the following message appears when you attempt to define the Paradox data source:

> The setup routines for the INTERSOLV ParadoxFile (*.db) ODBC driver could not be loaded. You may be low on memory and need to quit a few applications.

2 Make sure the directory containing the Borland Database Engine appears in your program search path.

3 Make sure the PARADOX.NET file and IDAPI configuration file are correctly configured. (These files come with the Paradox software.)

For instructions, see your Paradox documentation.

4 Make sure the column names in your data source do not contain any Paradox reserved words.

If a column name contains a Paradox reserved word, an error message may appear when you try to access this table in PowerBuilder or InfoMaker. If this occurs, rename the column and reconnect.

For more about Paradox reserved words, see your Paradox documentation.

Preparing a shared Paradox data source

In addition to accessing a local Paradox data source, you can also access shared Paradox data sources from PowerBuilder or InfoMaker. Examples of shared Paradox data sources are:

- Paradox tables residing in a shared network directory
- Paradox tables residing on your computer and shared among many applications

❖ To access a shared Paradox data source:

1 Set up a shared directory on your network and configure it to give appropriate access to all users.

For instructions, see your system or network administrator.

2 Make sure you specify the following connection parameters when you define the data source as described in the next section, "Defining the data source."

- **Database Directory** Specifies the directory containing the Paradox tables you want to access.
- **Network Directory** If the database directory you specify is a shared network directory, Paradox also requires a Network Directory (NetDir) setting. This setting specifies the directory containing the PARADOX.NET file for the database you want to access.

3 Make sure every user who accesses the same shared directory of Paradox tables has a Network Directory setting that points to the same PARADOX.NET file.

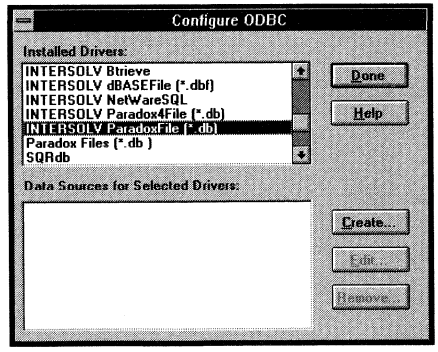
If your connect string does not specify a Network Directory value, Paradox uses the NetDir setting specified in the Paradox section of the IDAPI configuration file. Therefore, make sure the IDAPI configuration file includes the correct NetDir setting.

For instructions, see your Paradox documentation.

Defining the data source

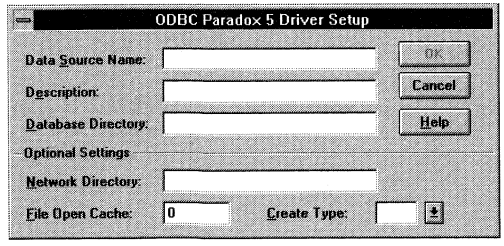
❖ To define a Paradox data source for the INTERSOLV Paradox 5 driver:

1 Select the INTERSOLV Paradox 5 driver in the Configure ODBC dialog box.



2 Click the Create button.

The ODBC Paradox 5 Driver Setup dialog box appears.



3 Specify values as follows:

> **For more information**
> For more about each field in the ODBC Paradox 5 Driver Setup dialog box, click the Help button.

Field	Value
Data Source Name	A short name to identify the data source. This becomes the name of the database profile created for this data source.
Description	(Optional) A description of the data source.
Database Directory	The drive and directory of the Paradox data source (for example, C:\PARADOX).
Network Directory	(Optional) The directory containing the PARADOX.NET file that corresponds to the database you want to access. For instructions on accessing Paradox data sources in a shared network directory, see "Preparing a shared Paradox data source" on page 104.
File Open Cache	(Optional) The maximum number of unused file opens to cache. For example, the value 3 specifies that when you open and close three tables, the driver keeps all three tables open so it does not have to reopen them if another query uses one of the tables. The default is 0, which means no file opens are cached.
Create Type	(Optional) The version the driver uses to create tables. Select **3** to create Paradox 3.x tables, **4** to create Paradox 4.x tables, or **5** to create Paradox 5.x tables. If you leave Create Type blank, the driver uses the type specified in the Level setting in the Paradox section of the IDAPI configuration file.

4 Click OK to create the data source definition.

About creating an index for Paradox tables

When you create the primary (unique) index for a Paradox table accessed with the INTERSOLV Paradox 5 ODBC driver, you *must* name the index Primary. You can then create other non primary (duplicate) indexes for the table as needed.

For instructions on creating indexes, see the PowerBuilder or InfoMaker *User's Guide*.

What to do next

For instructions on connecting to the data source, see Chapter 4, "Managing Database Connections."

Microsoft Paradox

This section describes how to prepare and define a Paradox data source in order to connect to it using the Microsoft Paradox ODBC driver.

Supported versions

The Microsoft Paradox ODBC driver supports connection to Paradox Version 3.x data sources.

> **Accessing Paradox Version 4.x data sources**
> The Microsoft Paradox driver does *not* support connection to Paradox Version 4.x data sources. To access Paradox Version 4.x data sources from PowerBuilder or InfoMaker, you can:
>
> - Use the INTERSOLV Paradox 4 driver (described on page 94) or the INTERSOLV Paradox 5 driver (described on page 101).
> - Use the compatibility mode feature in Paradox to save Paradox Version 4.x files as Version 3.x. You can then use the Microsoft Paradox driver to access the Version 3.x files.

Supported SQL operations

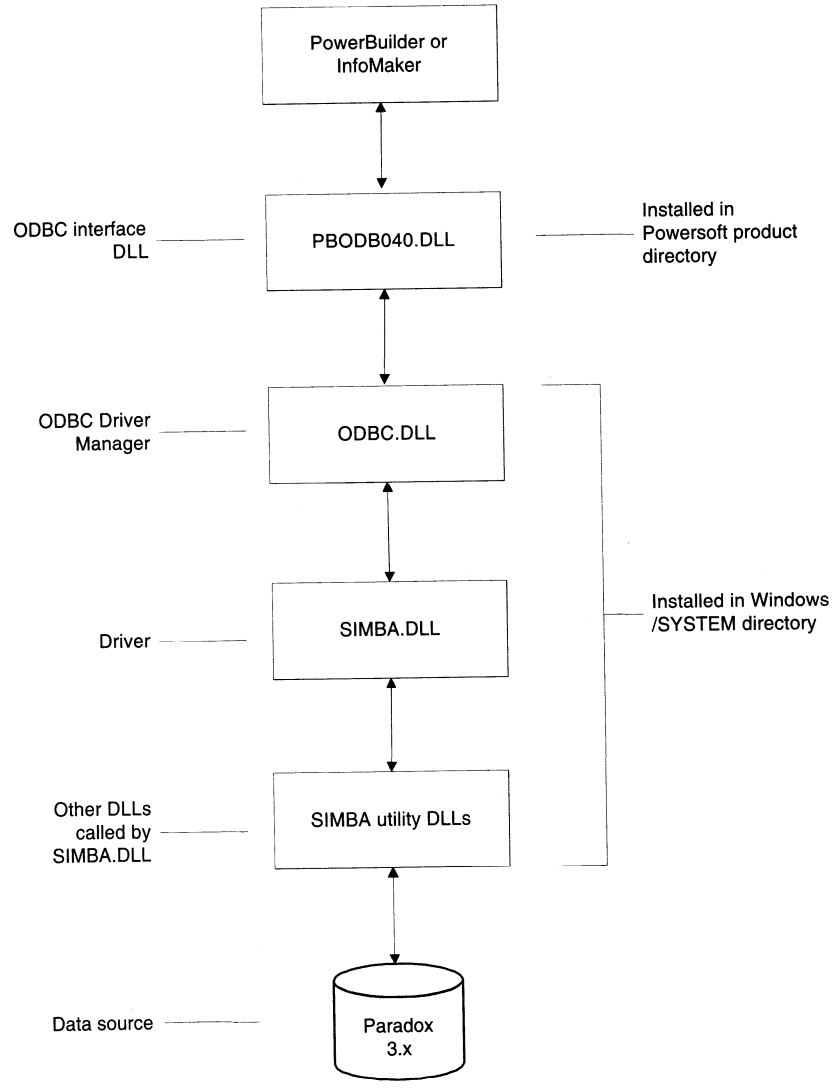
The Microsoft Paradox ODBC driver supports the following SQL operations with Paradox Version 3.x data sources.

Note that for the CREATE INDEX operation, the primary index name *must* match the table name.

CREATE INDEX	DELETE TABLE
CREATE TABLE	UPDATE

Basic software components

The following diagram shows the basic software components you need to connect to a Paradox data source using the Microsoft Paradox ODBC driver. All of these files come with PowerBuilder or InfoMaker. To access a Paradox data source with the Microsoft Paradox driver, *you need not install* the Windows version of the Paradox Engine (like the INTERSOLV Paradox 4 driver described on page 94) or the Borland Database Engine (like the INTERSOLV Paradox 5 driver described on page 101).



Preparing to use the data source

Before you define and connect to a Paradox data source from PowerBuilder or InfoMaker, follow these steps to prepare the data source.

❖ To prepare a Paradox data source when using the Microsoft Paradox driver to connect:

1 Make sure the data source you are accessing was created with Paradox Version 3.x.

For information about accessing Paradox Version 4.x data sources, see "Supported versions" on page 108.

2 Make sure the column names in your data source do not contain any Paradox reserved words.

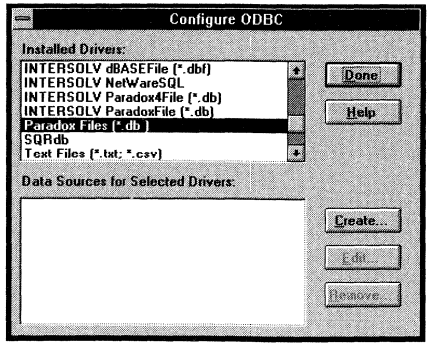
If a column name contains a Paradox reserved word, an error message may appear when you try to access this table in PowerBuilder or InfoMaker. If this occurs, rename the column and reconnect.

For more about Paradox reserved words, see your Paradox documentation.

Defining the data source

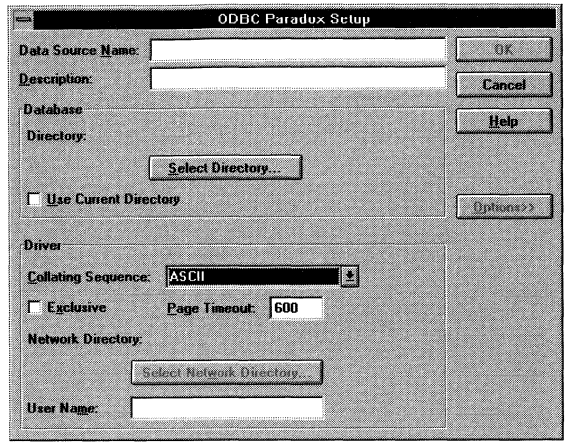
❖ To define a Paradox data source for the Microsoft Paradox driver:

1 Select the Microsoft Paradox driver in the Configure ODBC dialog box.



2 Click the Create button.

The ODBC Paradox Setup dialog box appears. (Click the Options button if necessary to display the entire dialog box.)



3 Specify values as follows:

For more information

For a detailed description of each field in the ODBC Paradox Setup dialog box, click the Help button.

Field	Value
Data Source Name	A short name to identify the data source. This becomes the name of the database profile created for this data source.
Description	(Optional) A description of the data source.
Directory	Displays the drive and directory of the Paradox data source. To select a directory and display it in the Directory field, click the Select Directory button and choose a directory from the list. To make the current working directory the data source directory, select the Use Current Directory checkbox. This disables the Select Directory button.
Collating Sequence	(Optional) The sequence in which characters are sorted. ASCII is the default.

Field	Value
Exclusive	(Optional) Select this checkbox to open Paradox files in exclusive mode. Exclusive mode allows file access by only one user at a time, and enhances performance. Leave this checkbox deselected to open Paradox files in shared mode. Shared mode allows file access by more than one user at a time.
Page Timeout	(Optional) The time in tenths of a second that an unused page remains in the buffer before being removed. The default is 600 (60 seconds).
Network Directory	(Optional) Displays the drive and directory where the PARADOX.NET file resides. To select the network directory and display it in this field, click the Select Network Directory button and choose a directory from the list. The Select Network Directory button is enabled by typing your user name in the User Name field. For more about the PARADOX.NET file, see your Paradox documentation.
User Name	(Optional) Your Paradox user name. Typing a user name in this field enables the Select Network Directory button, described above.

4 Click OK to create the data source definition.

What to do next

For instructions on connecting to the data source, see Chapter 4, "Managing Database Connections."

DEC Rdb

This section describes how to prepare and define an Rdb data source in order to connect to it using the Digital Equipment Corporation (DEC) Rdb ODBC driver.

Supported products
The DEC Rdb ODBC driver ships with PowerBuilder Enterprise and InfoMaker only. It does *not* ship with PowerBuilder Team/ODBC or PowerBuilder Desktop.

Supported versions

The DEC Rdb ODBC driver supports connection to Rdb/VMS databases Version 4.0 or higher running on the VMS operating system and accessed through DECnet or TCP/IP.

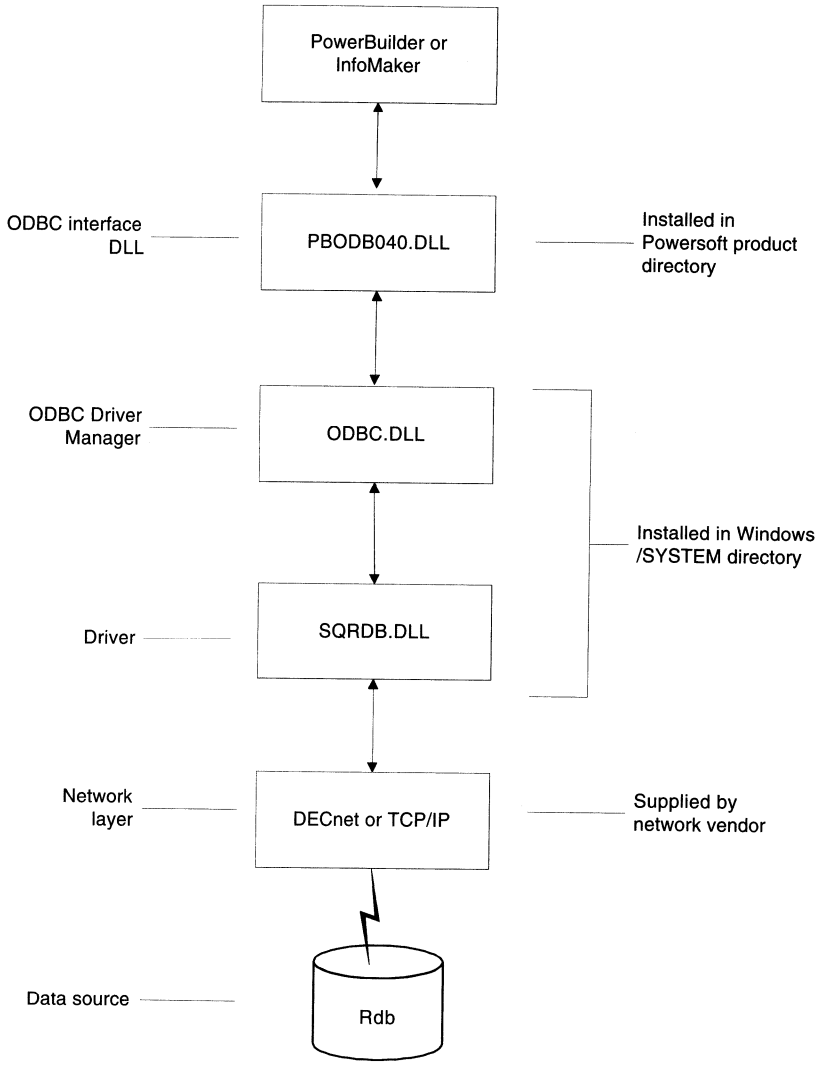
Supported SQL operations

The DEC Rdb ODBC driver supports the following SQL operations with Rdb data sources:

ALTER TABLE (column widths only)
CREATE INDEX
CREATE TABLE
CREATE VIEW
DELETE INDEX
DELETE TABLE
UPDATE

Basic software components

The following diagram shows the basic software components you need to connect to an Rdb data source using the DEC Rdb ODBC driver. All of these files come with PowerBuilder or InfoMaker.



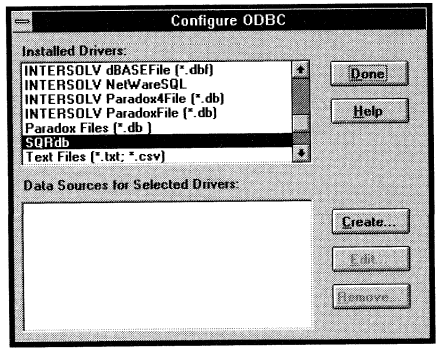
Preparing to use the data source

There is no special preparation required to access an Rdb data source from PowerBuilder or InfoMaker.

Defining the data source

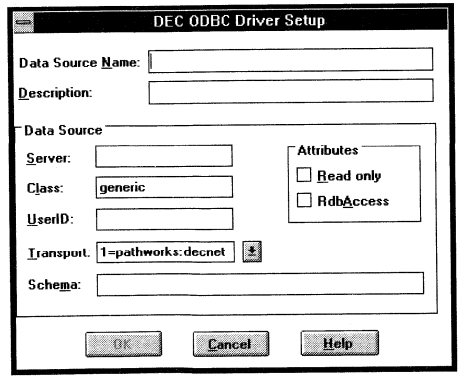
❖ To define an Rdb data source for the DEC Rdb driver:

1 Select the DEC Rdb (SQRdb) driver in the Configure ODBC dialog box.



2 Click the Create button.

The DEC ODBC Driver Setup dialog box appears.



3 Specify values as follows:

For more information

For a detailed description of each field in the DEC ODBC Driver Setup dialog box, click the Help button.

Field	Value
Data Source Name	A short name to identify the data source. This becomes the name of the database profile created for this data source.
Description	(Optional) A description of the data source.
Server	The VMS node running the SQL/Services server used to access the data. For information, contact your network administrator.
Class	A SQL/Services class name that identifies the execution class of the SQL/Services server. Generic is the default. For information, contact your network administrator.
UserID	The VMS user name of the account on the server used to access the data.
Transport	The network transport (DECnet or TCP/IP) used to access the database. For information, contact your network administrator.
Schema	The filename of the Rdb database you want to access. Precede the filename with a SQL ATTACH or SQL DECLARE SCHEMA statement.
Attributes	Select the checkboxes that represent the type of read/write access you need to the data source. You can select neither, one, or both checkboxes.

4 Click OK to create the data source definition.

What to do next

For instructions on connecting to the data source, see Chapter 4, "Managing Database Connections."

Microsoft Text File

This section describes how to prepare and define a text data source in order to connect to it using the Microsoft Text File ODBC driver.

Structure of a text database

The Microsoft Text File ODBC driver supports connection to a text database. A **text database** is a directory that contains one or more text files formatted as tables. Because the data source name references a particular directory, you can access any text tables residing in this directory through a single data source connection.

For example, assume you have a directory named C:\SALES that contains three text tables: Customer, Product, and Vendor. You then define an ODBC text data source named Sales, which references the C:\SALES directory. When you connect to the Sales data source in PowerBuilder or InfoMaker, you can access data from all three tables in the \SALES directory.

SCHEMA.INI file

Within the data source directory, a file named SCHEMA.INI contains a description of the content and format of each text table in the directory. The Microsoft Text File ODBC driver uses the information in SCHEMA.INI to create and open new text tables.

You create the SCHEMA.INI file when you:

- Define the text data source (for instructions, see "Defining the data source" on page 121)
- Create a new text table in PowerBuilder or InfoMaker (for instructions, see the PowerBuilder or InfoMaker *User's Guide*)

Supported versions

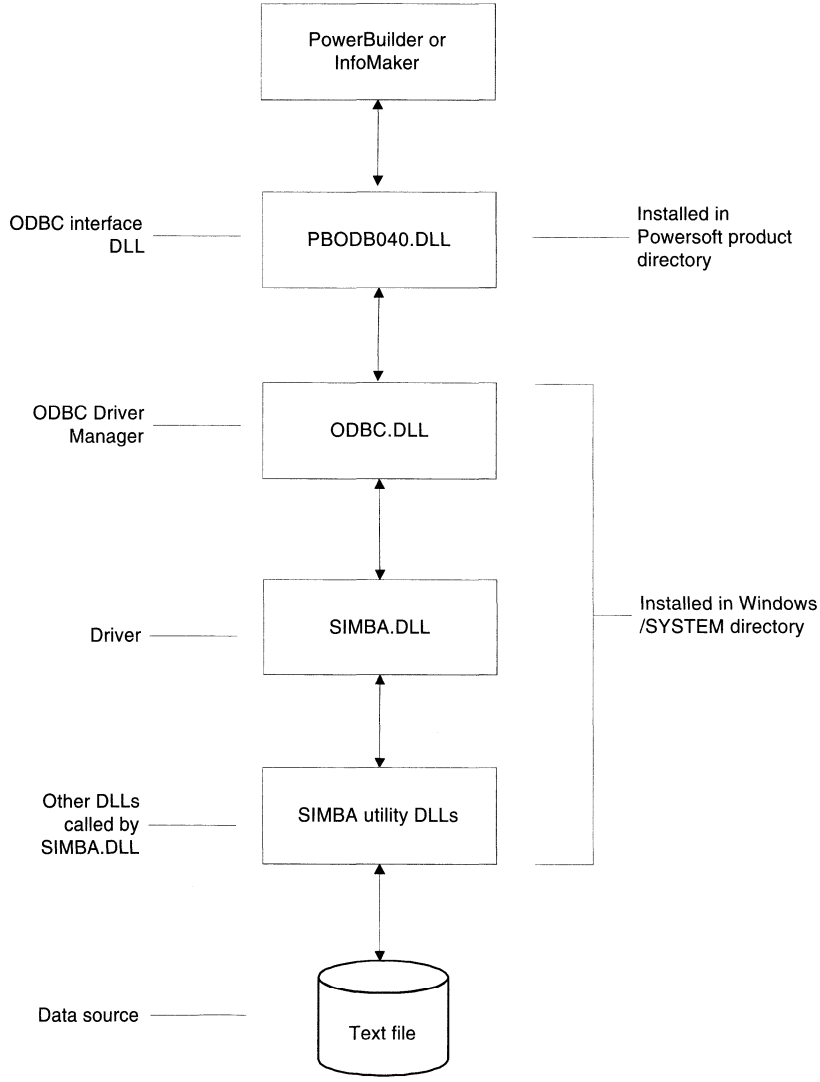
The Microsoft Text File ODBC driver supports connection to any text database.

Supported SQL operations

The Microsoft Text File ODBC driver supports no SQL operations with text data sources.

Basic software components

The following diagram shows the basic software components you need to connect to a text data source using the Microsoft Text File ODBC driver. All of these files come with PowerBuilder or InfoMaker.



Preparing to use the data source

Before you define and connect to a text data source from PowerBuilder or InfoMaker, follow these steps to prepare the data source.

❖ To prepare a text data source:

1 Make sure all text tables that you want to access through a single data source connection reside in the same directory on your computer.

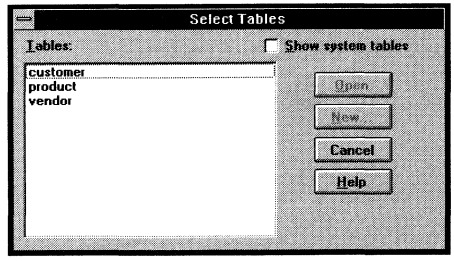
To access text files in a directory *other* than the one referenced by your data source, do either of the following:

- Define a new data source that references this directory.
- Move all text files to the directory referenced by your data source.

2 Rename all text files to remove extensions such as ASC, CSV, TAB, or TXT.

Only those filenames without extensions will appear in the list of tables for the data source in PowerBuilder or InfoMaker.

For example, assume a directory named C:\SALES contains three text files named CUSTOMER.TXT, PRODUCT.CSV, and VENDOR.TAB. By renaming these files CUSTOMER, PRODUCT, and VENDOR, they will appear as follows in the Select Tables dialog box in PowerBuilder or InfoMaker:

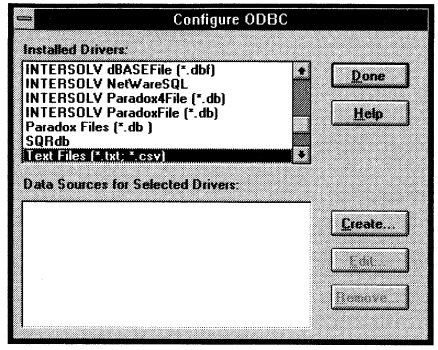


You can then open one or more of these tables to view or manipulate their data.

Defining the data source

❖ To define a text data source for the Microsoft Text File driver:

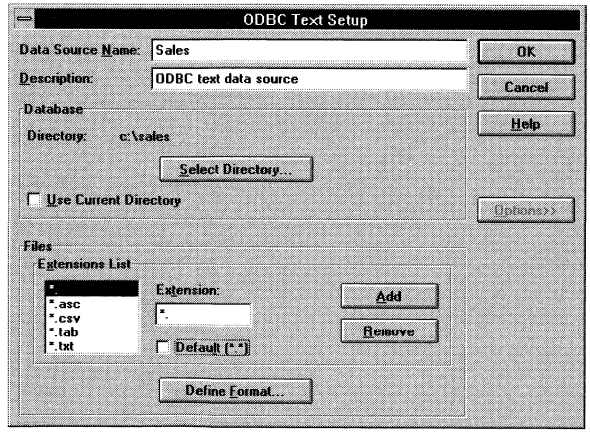
1 Select the Microsoft Text File driver in the Configure ODBC dialog box.



2 Click the Create button.

The ODBC Text Setup dialog box appears. (Click the Options button if necessary to display the entire dialog box.)

By default, the ODBC Text Setup dialog box appears with several fields disabled. In order to enable these fields in the illustration, the following example shows certain fields completed.



3 Specify values as follows:

For more information

For a detailed description of each field in the ODBC Text Setup dialog box, click the Help button.

Field	Value
Data Source Name	A short name to identify the data source. This becomes the name of the database profile created for this data source.
Description	(Optional) A description of the data source.
Directory	Displays the drive and directory of the text data source. To select a directory and display it in the Directory field, click the Select Directory button and choose a directory from the list. Selecting a directory enables the Define Format button. To make the current working directory the data source directory, select the Use Current Directory checkbox. This disables the Select Directory button and enables the Define Format button.
Extensions List	This field is not applicable when accessing a text data source through PowerBuilder or InfoMaker. You can access only those text filenames without extensions. For use with PowerBuilder or InfoMaker, make sure ***.** (the first extension on the list) is selected. This happens automatically when you deselect the Default checkbox, as described later in this table. For more about removing extensions from text data sources, see "Preparing to use the data source" on page 120.
Extension	The file extension. For use with PowerBuilder or InfoMaker, make sure ***.** appears in this field. This happens automatically when you deselect the Default checkbox, as described later in this table.

Field	Value
Default	Deselect this checkbox for use with PowerBuilder or InfoMaker. This places the correct value (*.) in the Extensions List and Extension field. In PowerBuilder or InfoMaker, you can access only those text filenames without extensions.
Add	Not applicable for use with PowerBuilder or InfoMaker.
Remove	Not applicable for use with PowerBuilder or InfoMaker.
Define Format	Displays the Define Text Format dialog box to define the schema of individual tables in the data source directory. The Define Format button is enabled after you specify a data source directory. For instructions, see "Defining the format of text files" next.

4 Click OK to create the data source definition.

Defining the format of text files

In order to access tables in a text database, the Microsoft Text File driver needs information about the content and format of each table in the database. It gets this information from a file named SCHEMA.INI, located in the data source directory. The driver uses the information in SCHEMA.INI to create new and open existing text tables.

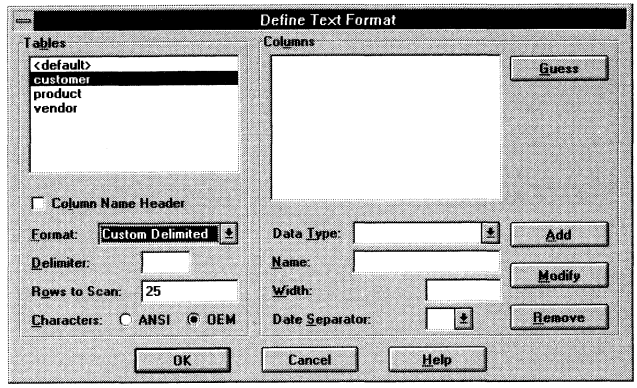
As part of defining a text data source, you can use the Define Text Format dialog box to specify the structure of each table you want to access. This information is stored in the SCHEMA.INI file for the data source directory. There is a separate SCHEMA.INI file for each text data source directory.

❖ To define the format of a text file when using the Microsoft Text File driver:

1 Click the Define Format button in the ODBC Text Setup dialog box. (The Define Format button is enabled after you specify a data source directory.)

The Define Text Format dialog box appears.

By default, the Define Text Format dialog box appears with <default> selected in the Tables list and the Columns fields disabled. In order to enable these fields in the illustration, the following example shows the Customer table and Custom Delimited format selected.



2 Specify values as follows:

Field	Value
Tables	Lists the names of tables (text files) defined in the data source directory. Select the table you want to define. When you select a table other than <default>, the Columns fields become enabled. If an entry exists for this table in the SCHEMA.INI file, the Columns list and associated fields display information about each column in the table. If <default> is selected, you can specify the format for tables not defined in the data source directory. Selecting <default> disables the Columns list and associated fields.
Column Name Header	Select this checkbox if the first row in the table contains column names instead of data.

Field	Value
Format	Select one of the following text formats from the dropdown listbox: ♦ **CSV Delimited** Fields are separated by commas. ♦ **Tab Delimited** Fields are separated by tab characters. ♦ **Custom Delimited** Fields are separated by the character specified in the Delimiter field. ♦ **Fixed Length** Fields are of a fixed length. Selecting Fixed Length changes the Guess button to a Parse button, as described later in this table.
Delimiter	The character used as a column separator in custom delimited text files. This field is enabled only if you select Custom Delimited from the Format dropdown listbox.
Rows to Scan	The number of rows you want the driver to scan to determine the data type of each column. To scan the entire file, enter 0. The default is 25 rows.
Characters	Click the ANSI radio button if the text file uses the ANSI character set. Click the OEM radio button if the text file uses a non-ANSI character set. OEM is the default.
Columns	If an entry exists for the selected table in the SCHEMA.INI file, or if you use the Guess or Parse button to generate column information, this field lists the names of columns in the selected table. To remove a column or to specify its properties, select the column name from the Columns list.
Data Type	Displays the data type of the selected column. To specify or change this value, select another data type from the dropdown listbox.
Name	Displays the name of the selected column. To specify or change this value, type a new name in the field.

Field	Value
Width	Displays the width of the selected column. To specify or change this value, type a new width in the field. If the table format is Fixed Length, width is valid for all data types. However, the only way to change the width is by using the Parse dialog box, as described on page 128. If the table format is CSV Delimited, Tab Delimited, or Custom Delimited, the Width field is enabled only for Char and LongChar data types.
Date Separator	Displays the separator character for columns having a Date data type. To specify a character, select it from the dropdown listbox. This field is enabled only if you select one of the Date data types from the Data Type dropdown listbox.
Guess	Click the Guess button to automatically generate the data type, name, and width values for all columns in the selected table. The driver generates these values by scanning the table's contents according to the format selected in the Format dropdown listbox. When you click Guess, the driver clears any previously defined columns from the Columns table and replaces them with the newly generated columns. The Guess button is available only when you select CSV Delimited, Tab Delimited, or Custom Delimited from the Format dropdown listbox. It changes to a Parse button when you select Fixed Length.
Parse	Displays the Parse dialog box to define the format of fixed length files. The Parse button is available only when you select Fixed Length from the Format dropdown listbox. It changes to a Guess button when you select CSV Delimited, Tab Delimited, or Custom Delimited. For instructions on using the Parse dialog box, see page 128.

Field	Value
Add	Click the Add radio button to add the selected column and its properties to the end of the Columns list and to the SCHEMA.INI file for the data source.
Modify	Click the Modify radio button to change one or more properties for the selected column. To modify a column's properties, select the column name, edit its data type, name, width, or date separator as desired, and click Modify.
Remove	Click the Remove radio button to delete the selected column and its properties from the Columns list and from the SCHEMA.INI file for the data source.

3 Click OK to save the text format definition.

Defining the format of fixed-length files

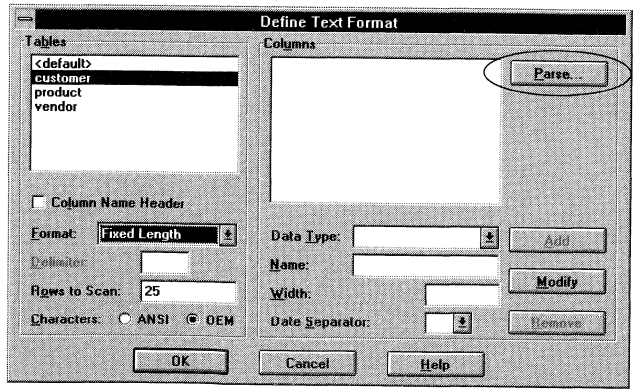
Use the following procedure to define the format of a fixed-length text file.

❖ To define the format of a fixed length text file:

1 In the Define Text Format dialog box, select a table name from the Tables list.

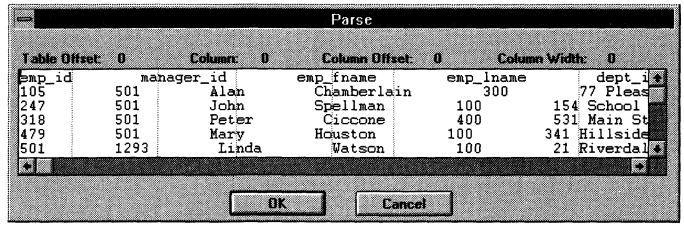
2 Select Fixed Length from the Format dropdown listbox.

This changes the Guess button to a Parse button.



3 Click the Parse button.

The Parse dialog box appears. In the dialog box, vertical lines separate the table columns and a vertical line marks the last column.



4 Define the column format as follows:

- **To separate two columns** Position the cursor after the last character in the first column. Then double-click or press the space bar where you want to add a line separating the two columns.
- **To join two columns** Position the cursor on the separator line that you want to remove. Then double-click the line or press the space bar to remove the line.

5 Click OK.

The Define Text Format dialog box appears. The Columns list displays the column names and the data type is set to Char for each column.

If necessary, you can change columns and their properties by using the Add, Modify, and Remove buttons, as described on page 127.

What to do next

For instructions on connecting to the data source, see Chapter 4, "Managing Database Connections."

Watcom SQL

This section describes how to prepare and define a Watcom SQL data source in order to connect to it from PowerBuilder or InfoMaker using the Watcom SQL ODBC driver.

Supported versions

The Watcom SQL ODBC driver supports connection to Watcom SQL databases created with the following:

- PowerBuilder or InfoMaker running on your computer
- Watcom SQL for Windows Version 3.2 or higher
- Watcom SQL Network Server Edition Version 3.2 or higher

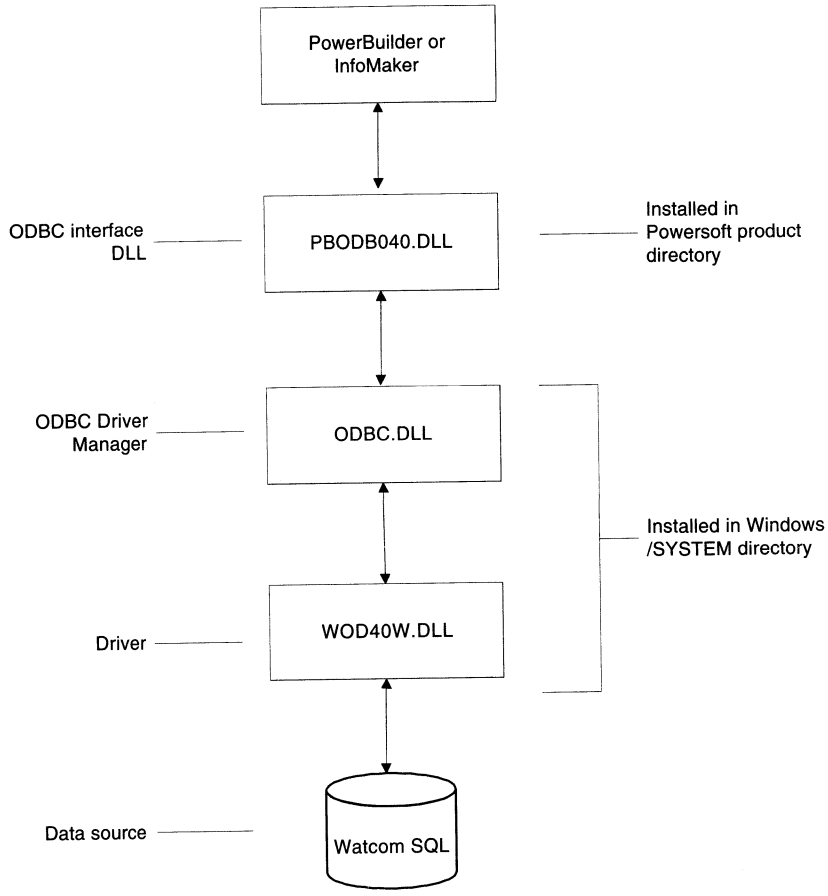
Supported SQL operations

The Watcom SQL ODBC driver supports the following SQL operations with Watcom SQL data sources:

ALTER TABLE
CREATE INDEX
CREATE PRIMARY KEY
CREATE TABLE
CREATE VIEW
DELETE INDEX
DELETE TABLE
UPDATE

Basic software components

The following diagram shows the basic software components you need to connect to a Watcom SQL data source using the Watcom SQL ODBC driver. All of these files come with PowerBuilder or InfoMaker.



Preparing to use the data source

Before you define and connect to a Watcom SQL data source from PowerBuilder or InfoMaker, follow these steps to prepare the data source.

❖ To prepare a Watcom SQL data source:

1 Make sure the database file for the Watcom SQL data source already exists.

To create a new Watcom SQL database, you can:

- Use the Database painter in PowerBuilder or InfoMaker when running these products on your computer. (See the instructions in the PowerBuilder or InfoMaker *User's Guide*.)
- Create the database some other way, such as with PowerBuilder or InfoMaker running on another user's computer, or by using Watcom SQL commands outside PowerBuilder or InfoMaker.

  For more about Watcom SQL commands, see *Watcom SQL* or online Help if you are using PowerBuilder, or online Help if you are using InfoMaker.

2 Make sure you have the LOG file associated with the Watcom SQL database.

If the LOG file for the Watcom SQL database does not exist, the Watcom SQL database engine will create it. However, if you are copying or moving a database from another computer or directory, you should copy or move the LOG file with it.

By keeping the LOG file with the database, you can fully recover the database if it becomes corrupted.

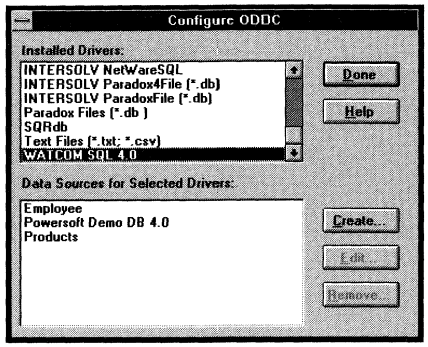
Defining the data source

When you create a Watcom SQL database using PowerBuilder or InfoMaker on your computer, the software automatically defines the data source and creates the database profile for you.

Therefore, you need only use the following procedure to define a Watcom SQL data source when you want to access a Watcom SQL database *not* created using PowerBuilder or InfoMaker on your computer.

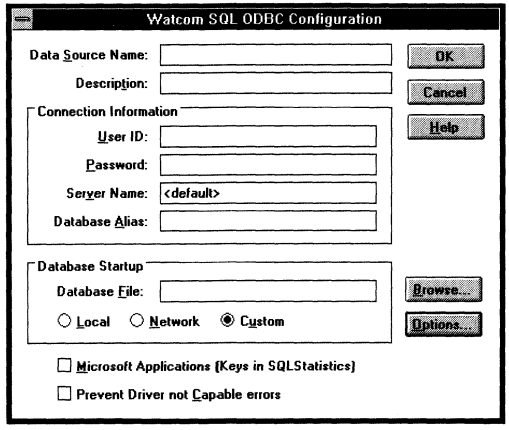
❖ To define a Watcom SQL data source for the Watcom SQL driver:

1 Select the Watcom SQL driver in the Configure ODBC dialog box.



2 Click the Create button.

The Watcom SQL ODBC Configuration dialog box appears.



3 Specify values as follows:

> **For more information**
>
> For a detailed description of each field in the Watcom SQL ODBC Configuration dialog box, click the Help button.

Field	Value
Data Source Name	A short name to identify the data source. This becomes the name of the database profile created for this data source.
Description	(Optional) A description of the data source.
User ID	(Optional) The user name used to connect to the data source. If you omit the user ID, PowerBuilder and InfoMaker prompt you for it when you connect to the data source, unless the UID (user ID) value is already in the ConnectString. (If the ODBC driver returns the UID value when you connect, PowerBuilder or InfoMaker copies this value to the ConnectString DBParm in your database profile.)
Password	(Optional) The password for the specified user ID. The password is stored in the ODBC.INI file. Therefore, specifying a password here may be a security risk. If you omit the password, PowerBuilder or InfoMaker will prompt you for it when you connect to the data source.
Server Name	The name of a Watcom SQL database engine or Watcom SQL network server. If you omit the server name, the default local engine is used. This is the first database engine started.
Database Alias	If specified, corresponds to the name of a database already running on a Watcom SQL database engine or Watcom SQL network server.
Database File	If specified, contains the name of a database file (for example, C:\WSQL\TEST.DB). You can type the file name, or click the Browse button to select the name of an existing database file and display it in this field.

Field	Value
Database Startup	Click one of the following radio buttons to start the Watcom SQL database engine when the named database engine or server is not already running: ♦ **Local** Start the 32-bit version of the database engine with the default command **db32w**. If you click Local, you cannot edit this command to add other database switches or startup options. You should select Local if you are using a single-user Watcom SQL database and want to start the engine with the default settings. ♦ **Network** Start the database engine with the default command **dbclienw**. If you click Network, you cannot edit this command to add other database switches or startup options. You should select Network if you are using the Watcom SQL Network Server Edition and want to start the engine with the default settings. ♦ **Custom** Click Custom and then click the Options button to display the Startup Options dialog box. You should select Custom and complete the Startup Options dialog box if you want to specify nondefault commands, database switches, and options to start the database engine. For instructions, see "Specifying startup options for the database engine" on page 135.)
Microsoft Applications (Keys in SQLStatistics)	Not applicable for use with PowerBuilder or InfoMaker.
Prevent Driver not Capable errors	Not applicable for use with PowerBuilder or InfoMaker.

Field	Value
Browse	Click the Browse button to display the Select Database dialog box. When you select an existing Watcom SQL database from the Select Database dialog box and click OK, its fully qualified name displays in the Database File field (for example, C:\WSQL\TEST.DB). When you use the Browse button, the name of the selected database file (for example, Test) also appears in both the Data Source Name and Database Alias fields. If you want to specify a different name for the data source or database, you can edit one or both of these fields *after* using the Browse button.
Options	Displays the Startup Options dialog box to specify nondefault commands, database switches, and options to start the database engine. The Options button is enabled only when you click the Custom button, described earlier in this table. For instructions on completing the Startup Options dialog box, see "Specifying startup options for the database engine" next.

4 Click OK to create the data source definition.

Specifying startup options for the database engine

In most cases, you should not have to change the default commands that start the Watcom SQL database engine. These commands are **db32w** (associated with the Local radio button) and **dbclienw** (associated with the Network radio button).

Reasons for using nondefault startup options

Sometimes, however, you may need to start the database engine with nondefault commands, database switches, and options. There are several reasons for using nondefault startup options. Some examples are:

- You want to start the database engine with a command that includes one or more engine switches, or with a command other than **db32w** or **dbstartw**.

  For example, you can use the **-n** engine switch to specify a name for the database engine (for example, **db32w -n myengine**). Or, you can use the **dbstartw** command to start the 16-bit version of the database engine.

- You want to specify a database switch.

  For example, you can use the **-n** database switch to specify the name of the database file running on a particular engine (for example, **-n mydatabase**).

  Or, you may want to use the **-d** database switch to start the database engine using normal DOS input and output (I/O) routines instead of fast (direct) I/O routines. With some hard disk drives, Watcom SQL cannot use its fast I/O routines. The symptom for this is that when you start the database engine, such as by opening the Database painter in PowerBuilder or InfoMaker, the MS-DOS prompt appears and you must reboot your computer. To solve this problem, you can start the engine with the -d switch (for example, **db32w -d**).

- You want to use the Autostop Database option to specify whether the database engine or the database running on the engine should stop automatically when you disconnect from the database.

How to specify startup options

Use the following procedure if you need to specify nondefault startup options for the Watcom SQL database engine.

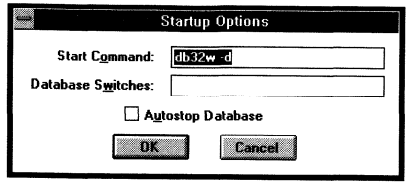
❖ **To specify startup options:**

1. In the Watcom SQL ODBC configuration dialog box, click the Custom radio button.

   The Options button becomes enabled.

2. Click the Options button.

The Startup Options dialog box appears.



3 Specify values as follows:

Field	Value
Start Command	The command you want to use to start the Watcom SQL database engine if it is not already running. For example, enter **db32w -n myengine** to start the 32-bit version of the engine with the name myengine. For more about commands that start the database engine, see the description of DBSTART in *Watcom SQL* or online Help if you are using PowerBuilder, or in online Help if you are using InfoMaker.
Database Switches	The switch or switches you want to use to specify database-specific options. For example, enter **-n mydatabase** to specify mydatabase as the name of your database file. For more about database switches, see the description of DBSTART in *Watcom SQL* or online Help if you are using PowerBuilder, or in online Help if you are using InfoMaker.
Autostop Database	Select the Autostop Database checkbox if you want the database engine or database running on that engine to stop automatically when you disconnect from the database. Deselect the Autostop Database checkbox if you do *not* want the database engine or database running on the engine to stop automatically when you disconnect from the database.

4 Click OK.

You are returned to the Watcom SQL ODBC Configuration dialog box.

Using local and network connections

The Watcom SQL ODBC driver supports connection to Watcom SQL databases that reside:

- **Locally on your computer** You can create a local Watcom SQL database using PowerBuilder, InfoMaker, or Watcom SQL for Windows.
- **Remotely on a Watcom SQL network server** You can create a network Watcom SQL database using the Watcom SQL Network Server Edition.

The following table compares the values you should specify in the Watcom SQL ODBC Configuration dialog box when accessing local and network Watcom SQL databases. (For more about these values, see the table on pages 133 through 135.)

Field	Value for local database connection	Value for network database connection
Data Source Name	Short name for data source	Short name for data source
Description	Description of data source	Description of data source
User ID	Your user ID	Your user ID
Password	Your password	Your password
Server Name	Name of the Watcom SQL database engine	Name of the Watcom SQL network server
Database Alias	Name of a database already running on a Watcom SQL database engine	Name of a database already running on a Watcom SQL network server
Database File	Name of a database file	Name of a database file
Database Startup	Local *or* Custom	Network *or* Custom

What to do next

For instructions on connecting to the data source, see Chapter 4, "Managing Database Connections."

CHAPTER 3

Using Powersoft Database Interfaces

About this chapter

This chapter provides an introduction to Powersoft database interfaces. It then describes how to use each supported Powersoft database interface in order to connect to it from PowerBuilder or InfoMaker.

Contents

Topic	Page
About Powersoft database interfaces	141
About preparing to use the database	144
About defining Powersoft database interfaces	145
What to do next	154
ALLBASE/SQL	155
IBM DRDA databases	162
INFORMIX	188
Micro Decisionware Database Gateway Interface for DB2	201
ORACLE	205
SQL Server	218
SQLBase	226
Sybase Net-Gateway Interface for DB2	231
Sybase SQL Server System 10	235
XDB	242
Creating Powersoft system tables in DB2 databases	247
Installing Powersoft stored procedures in SQL Server databases	250

For more information

This chapter gives general information about using each Powersoft database interface. For more detailed information:

- Check for a FaxLine document that describes how to connect to your database. Many of these FaxLines are available on the Powersoft Infobase CD-ROM. For a complete list of available FaxLines, order the *Technical Information Catalog* from the Powersoft FaxLine system.
- Ask your network or system administrator for assistance when installing and setting up the database server and client software at your site.

About Powersoft database interfaces

The Powersoft database interfaces provide native connections to many databases and DBMSs. This section describes how the Powersoft database interfaces connect to these databases.

Supported Powersoft database interfaces

For a complete list of the Powersoft database interfaces supported in PowerBuilder Enterprise and InfoMaker, see Appendix A, "Supported Data Sources and Databases."

The Powersoft database interfaces are *not* supported in PowerBuilder Team/ODBC and PowerBuilder Desktop. However, you can upgrade to PowerBuilder Enterprise in order to use the Powersoft database interfaces.

What is a Powersoft database interface?

A **Powersoft database interface** is a native (direct) connection to a database or DBMS.

Each Powersoft database interface has its own interface DLL that communicates with a specified database. For example, when you install the Powersoft SQL Server database interface, PBSYB040.DLL is the interface DLL that communicates with the SQL Server database. This is unlike an ODBC connection, in which a single Powersoft interface DLL named PBODB040.DLL communicates through the ODBC Driver Manager and corresponding driver to an ODBC data source.

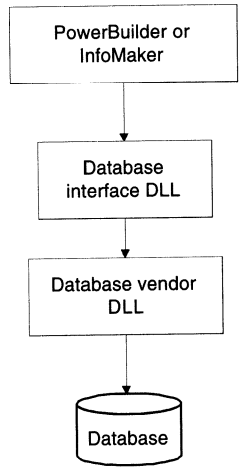
A Powersoft database interface does *not* go through an ODBC driver to access the database. Therefore, you do not complete a driver-specific ODBC setup dialog box to define it. Instead, you create a database profile in which you specify the information that PowerBuilder or InfoMaker needs to connect to the database.

Components of a database interface connection

When you use a Powersoft database interface to access a database, your connection goes through several layers before reaching the data. Each layer represents a separate component of the connection and each component may come from a different vendor.

PowerBuilder and InfoMaker provide a different database interface DLL for each supported database. This is because each interface DLL must connect to the database through a different database vendor API.

The following diagram shows the general components of a Powersoft database interface connection.



The following table gives the provider and a brief description of each component shown in the diagram.

Component	Provider	What it does
PowerBuilder or InfoMaker	Powersoft	Processes the data contained in the database
Database interface DLL	Powersoft	Submits SQL statements to the database vendor API for an application in order to retrieve results from the database PowerBuilder and InfoMaker provide a different database interface DLL for each supported database
Database vendor DLL	Database vendor	Routes SQL requests between the database interface DLL and the database
Database	Database vendor	Stores and manages data for an application

For more information

For diagrams showing the basic components of the connection for each Powersoft database interface, see the section for your Powersoft database interface on pages 155 through 242.

Using a Powersoft database interface

❖ To use a Powersoft database interface to access a database:

1 Prepare the database for use with PowerBuilder or InfoMaker.

For instructions, see the section for your Powersoft database interface on pages 155 through 242.

2 Install the Powersoft database interface that accesses this database.

For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide*.

3 Define the database interface to specify connection parameters.

This involves completing the Database Profile dialog box.

For instructions, see the section for your Powersoft database interface on pages 155 through 242.

About preparing to use the database

The first step in connecting to a database or DBMS through a Powersoft database interface is to prepare to use the database. Preparing the database ensures that you will be able to connect to it and use your data in PowerBuilder or InfoMaker.

You prepare a database *outside* PowerBuilder or InfoMaker *before* you start the product, define the database interface, and connect to it. The requirements differ for each database but in general, preparing a database involves making sure that:

- The required database server software is installed at your site
- The required network software is installed at your site
- The required client software files and directories are correctly installed on your computer
- Configuration files are properly set up

For instructions, see "Preparing to use the database" (or a similarly named section) in the section for your Powersoft database interface on pages 155 through 242.

About defining Powersoft database interfaces

After you prepare to use the database as described in the preceding section, you start PowerBuilder or InfoMaker and define the database interface. To define a database interface, you must complete the Database Profile Setup dialog box to create a database profile.

Getting help

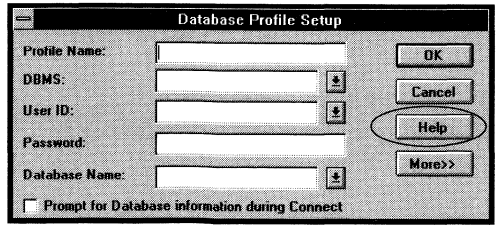
There are several ways that you can get help while creating a database profile for a Powersoft database interface. The following table summarizes the Help that is available.

To get help on	Do this
The fields in the Database Profile Setup dialog box	Click the Help button in the Database Profile Setup dialog box, as described below.
Your Powersoft database interface	See the Database Interfaces Help in PowerBuilder or InfoMaker, as described on page 146. Check whether there is a Powersoft FaxLine document available that describes how to connect to your database. For a complete list of available FaxLines, order the *Technical Information Catalog* from the Powersoft FaxLine system.

Displaying Help for the Database Profile Setup dialog box

❖ **To display Help for the Database Profile Setup dialog box:**

- Click the Help button in the Database Profile Setup dialog box.



A Help window appears describing the fields in the Database Profile Setup dialog box.

Displaying Help for your Powersoft database interface

❖ **To display Help for your Powersoft database interface:**

1. Select Help➤Database Interfaces from the PowerBuilder or InfoMaker menu bar.

   A Help window appears listing the Powersoft database interfaces.

2. Click the name of the Powersoft database interface for which you want to display Help.

   Another window appears listing the Help topics for the Powersoft database interface you selected.

3. Click an underlined topic to display its Help window.

Creating a database profile

When you complete the Database Profile Setup dialog box as described in the following procedure, PowerBuilder or InfoMaker automatically creates a database profile. You can select this database profile at any time to connect to the database with the connection parameters you specified.

For some Powersoft database interfaces, you need not supply values for all fields in the Database Profile Setup dialog box. If you supply the profile name and DBMS name and then click OK, PowerBuilder or InfoMaker displays a series of dialog boxes to prompt you for additional connection information. This information can include:

- User ID or login ID
- Password or login password
- Database name
- Server name

For some databases, supplying the profile name and DBMS name does not give PowerBuilder or InfoMaker enough information to prompt you for additional connection values. For these interfaces, you should supply values for all applicable fields in the Database Profile Setup dialog box, as described in the section for your interface on pages 155 through 242.

Before you start

Before you create a database profile to define a Powersoft database interface, make sure you have:

- Prepared your database for use with PowerBuilder or InfoMaker
- Installed the Powersoft database interface required to access the data

❖ To create a database profile for a Powersoft database interface:

1 Click the Database Profile button in the PowerBar.
or
In any of the painters listed in the following table, select File➤Connect➤Setup from the menu bar.

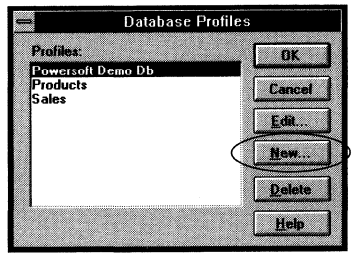
Product	Painters
PowerBuilder	Database painter DataWindow painter Report painter
InfoMaker	Database painter Form painter Report painter

Database Profile button

If your PowerBar does not include the Database Profile button, use the customize feature to add the button to the PowerBar.

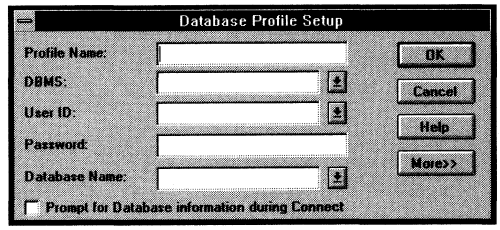
For instructions on customizing toolbars, see the PowerBuilder or InfoMaker *User's Guide*.

The Database Profiles dialog box appears, listing the names of existing profiles.



2 Click the New button to create a new database profile.

The Database Profile Setup dialog box appears.



3 Type a name for the database profile in the Profile Name field.

4 Select the identifier for the DBMS you want to access from the DBMS dropdown listbox.

The DBMS list contains identifiers for ODBC (to access ODBC data sources) and for any Powersoft database interfaces you have installed.

Where the DBMS identifiers come from

When you install a Powersoft database interface, PowerBuilder or InfoMaker updates the Vendors list in the [Database] section of the PB.INI or IM.INI file with the proper identifier for your interface.

The identifiers that appear in the DBMS list in the Database Profile Setup dialog box are the same ones that are in the Vendors list.

The ODBC identifier appears in the Vendors list and DBMS list by default.

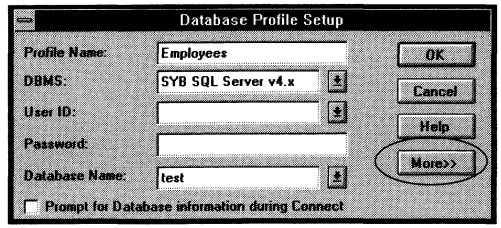
The following table lists the identifier you should select from the DBMS list for each Powersoft database interface.

Powersoft database interface	Identifier in DBMS list
ALLBASE/SQL	HPAllbase
Database Manager and DB2/2	IBM DB2
DB2/MVS	IBM DB2
DB2/6000	IBM DB2
INFORMIX Version 4.x/5.x through INFORMIX-NET Version 4.x	IN4 - I-Net v4.x
INFORMIX Version 5.x/6.x through INFORMIX-NET Version 5.x	IN5 - I-Net v5.x
Micro Decisionware Database Gateway Interface for DB2	MDI Gateway
ORACLE Version 6.x	OR6 - ORACLE v6.x
ORACLE Version 7.x	OR7 - ORACLE v7.x
SQL Server	SYB - SQL Server v4.x
SQLBase	GUPTA
Sybase Net-Gateway Interface for DB2	NETGATEWAY
Sybase SQL Server System 10	SYC - Sybase System 10
XDB	XDB

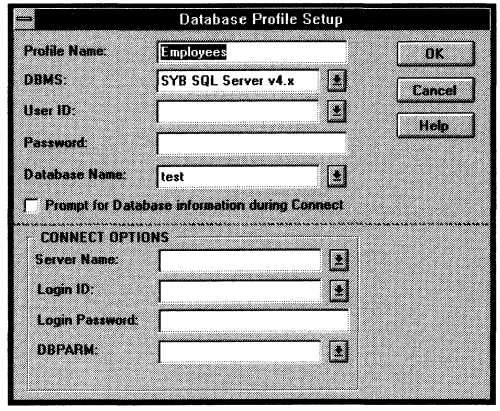
5 Supply values for the other fields in the Database Profile Setup dialog box as required for your database connection.

To display additional connect options in the Database Profile Setup dialog box, click the More button.

For example, here is the Database Profile Setup dialog box after specifying profile, DBMS, and database names for the Powersoft SQL Server database interface.

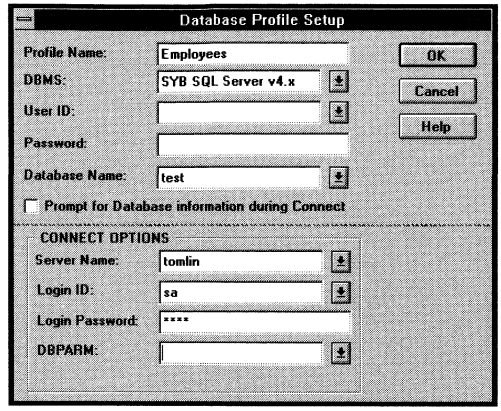


When you click the More button, the Database Profile Setup dialog box expands to reveal additional connect options that you can supply.



For information about the values you should supply in the Database Profile Setup dialog box, see the section for your Powersoft database interface on pages 155 through 242.

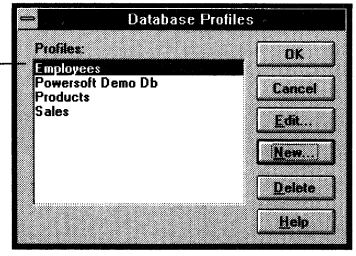
For example, a completed SQL Server database profile named Employees might look like the following:



6 Click OK in the Database Profile Setup dialog box.

The Database Profiles dialog box appears, with the new profile name highlighted.

New database profile



7 Click OK in the Database Profiles dialog box.

If you supplied sufficient information in the Database Profile Setup dialog box, PowerBuilder or InfoMaker connects to the database specified in the selected profile.

If the database profile does not supply enough information to connect to the database, various dialog boxes may display to prompt you for additional connection information.

What happens when you connect

When you create a database profile, the values are stored in the initialization file for PowerBuilder (PB.INI) or InfoMaker (IM.INI).

For example, the SQL Server database profile named Employees shown on page 151 creates the following entry in the PB.INI or IM.INI file:

```
[Profile Employees]
DBMS=SYB SQL Server v4.x
Database=test
UserId=
DatabasePassword=
LogPassword=help
ServerName=tomlin
LogId=sa
Lock=
DbParm=
Prompt=0
```

When you connect to the SQL Server database by selecting the Employees profile, PowerBuilder or InfoMaker copies the profile values for Employees to the [Database] section of the INI file to indicate that this is the current connection.

Specifying passwords in database profiles

As shown in the completed SQL Server profile on page 151, your password does not display when you specify it in the Database Profile Setup dialog box. Small Xs appear in place of the characters you type.

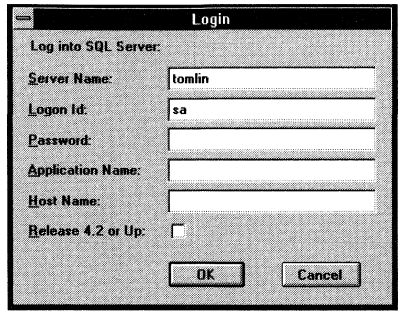
However, when PowerBuilder or InfoMaker stores the profile values in the PB.INI or IM.INI file, as shown on page 152, the actual password *does* display in the DatabasePassword or LogPassword field. If the INI file resides on a network at your site, displaying the password in this file may be a security risk.

❖ To avoid having the password display in the INI file:

- Do *not* type the password in the Database Profile Setup dialog box. Instead, specify the password in the dialog box that appears to prompt you for additional connection information.

  When you specify the password in response to a prompt instead of in the Database Profile Setup dialog box, the password does not display in the INI file entry for this profile.

For example, if you do not supply a password in the Database Profile Setup dialog box when creating a SQL Server database profile, the following Login dialog box appears to prompt you for the missing information:



Specifying the password in the Login dialog box instead of in the Database Profile Setup window creates the following entry in the PB.INI or IM.INI file. Notice that the LogPassword value does not appear.

```
[Profile Employees]
DBMS=SYB SQL Server v4.x
Database=test
UserId=
DatabasePassword=
LogPassword=
ServerName=tomlin
LogId=sa
Lock=
DbParm=appname=,host=,release=
Prompt=1
```

What to do next

For instructions on how to prepare the database and define the Powersoft database interface you are using, go to the section for your database interface on pages 155 through 242.

ALLBASE/SQL

This section describes how to use the Powersoft ALLBASE/SQL database interface in PowerBuilder or InfoMaker.

Supported data types

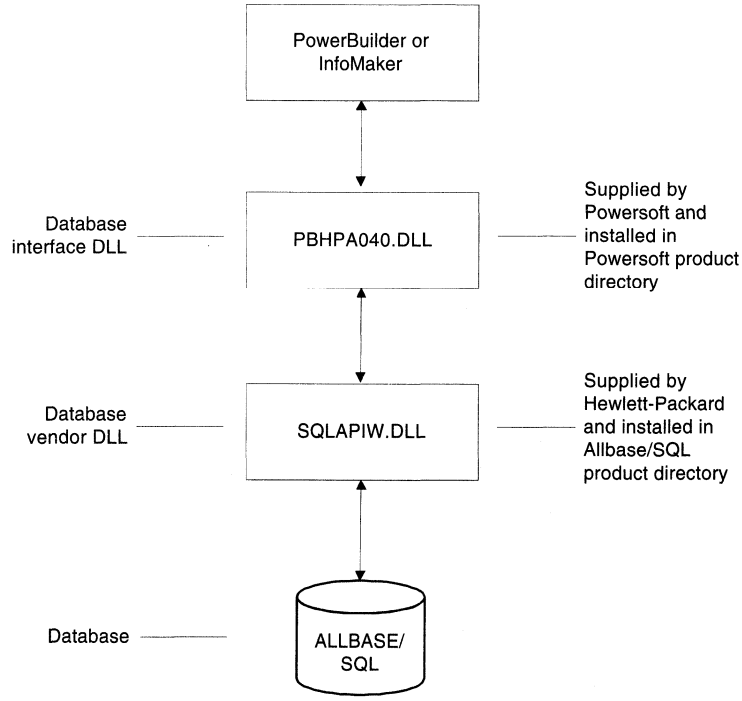
PowerBuilder and InfoMaker support the following ALLBASE/SQL data types in DataWindows, reports, and embedded SQL:

Char	Float
Date	Integer
DateTime	Time
Decimal	VarChar

Data type conversion
When you retrieve or update columns, PowerBuilder or InfoMaker converts data appropriately between the ALLBASE/SQL data type and the Powersoft data type.

Basic software components

The following diagram shows the basic software components you need to access an ALLBASE/SQL database using the ALLBASE/SQL database interface. The diagram also indicates who supplies each DLL and where it is installed.



Preparing to use the database

Before you define the interface and connect to an ALLBASE/SQL database from PowerBuilder or InfoMaker, follow these steps to prepare the database.

❖ To prepare an ALLBASE/SQL database:

1 Install the ALLBASE/SQL database server software, following the instructions in your ALLBASE/SQL documentation.

You must obtain the ALLBASE/SQL database server software from the Hewlett Packard Company.

2 Install the ALLBASE/SQL client software on your computer, following the instructions in your ALLBASE/SQL documentation.

You must obtain the ALLBASE/SQL client software from the Hewlett Packard Company.

3 Install the Powersoft ALLBASE/SQL database interface (PBHPA040.DLL) on your computer.

For instructions, see the *Installation and Deployment Guide*.

4 Make sure the file SQL.INI exists in your ALLBASE/SQL product directory, and that it is properly edited for your environment.

For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide*.

5 Make sure the following files are installed, and that the directory containing these files appears in your program search path:

- SQLAPIW.DLL
- SQLAWIN.DLL

Defining the database interface

The following table lists the values you should supply for each field in the Database Profile Setup dialog box when defining a Powersoft ALLBASE/SQL database interface.

Field	Value
Profile Name	The name of your database profile.
DBMS	**HPAllbase**
User ID	The user ID required to connect to your database. In order to properly create the Powersoft repository tables, make sure the first person to connect to the database has sufficient authority to create tables and grant permissions to public.
Password	The password required to connect to your database. The actual password does not display in this field. Small Xs appear in place of the characters you type.

Field	Value
Database Name	The name of the ALLBASE/SQL database you want to access.
Prompt for Database information during Connect	Select this checkbox if you want to be prompted for connection information when creating or selecting a profile to connect to the database.
Server Name	Not applicable for use with PowerBuilder or InfoMaker.
Login ID	Not applicable for use with PowerBuilder or InfoMaker.
Login Password	Not applicable for use with PowerBuilder or InfoMaker.
DBParm	Specify the HPConnect string and other DBMS-specific connection parameters in this field. For information, see "Specifying the HPConnect string" next. For information about DBParm values that you can specify for ALLBASE/SQL, see Chapter 5, "Setting Additional Connection Parameters."

Specifying the HPConnect string

To connect to an ALLBASE/SQL database, you can specify a valid HPConnect string in the DBParm field of the Database Profile Setup dialog box.

> **HPConnect string and SQL.INI file**
> When you use an HPConnect string to connect to an ALLBASE/SQL database, you do not need an [ALLBASE] section in your SQL.INI file. (SQL.INI is the configuration file for ALLBASE/SQL, and is generally installed in your ALLBASE/SQL product directory.)
>
> For instructions on setting up your SQL.INI file, see your ALLBASE/SQL system administrator.

The format of the HPConnect string is:

HPConnect = ' *connectstring* '

where *connectstring* must conform to the Hewlett-Packard MPE/iX or HP-UX syntax, as described in the following sections.

Getting help
The syntax for the HPConnect string can be complicated. If you need help specifying it in the Database Profile Setup window, see your ALLBASE/SQL system administrator.

Using MPE/iX syntax

The MPE/iX syntax for the HPConnect string is:

'**#mpeix**/*Node*:*DBEnvironment* {,*Network*}{,**Convert**}#*LogonString*'

Parameter	Description
Node	A string containing the node name of the database server. Node names can have a maximum of eight characters.
DBEnvironment	The name of the DBEnvironment containing one or more databases. The syntax for the fully qualified DBEnvironment is: *DBEnvironmentName*{.*Group*{.*Account*}}
Network	(Optional) The networking software on the client computer. For more, see the Hewlett Packard Company PC API documentation.
Convert	(Optional) This keyword converts the ROMAN8 character set to the ANSI character set on the client computer. It then converts the data back to ROMAN8 when returning it to the database server.
LogonString	The logon string required to connect to the database. For instructions, see "Specifying the MPE/iX logon string" next.

Specifying the MPE/iX logon string The MPE/iX syntax for the logon string is:

{*SessionId*},*User*{/*UserPass*}.*Account*{/*AcctPass*}{,*Group*{/*GrpPass*}}

Password security feature
You can use a question mark (?) as a placeholder for a password. Users will be prompted for the actual password when they log on to the database.

Parameter	Description
SessionId	(Optional) The name identifying the current MPE/iX session
User	The MPE/iX user name
UserPass	(Optional) The user password
Account	The MPE/iX user account
AcctPass	(Optional) The account password
Group	(Optional) The group name
GrpPass	(Optional) The group password

MPE/iX examples The following are two examples of valid HPConnect strings using the MPE/iX syntax. Type the HPConnect string *on a single line* in the DBParm field.

```
HPConnect = ' #mpeix/mpesysid:dbe1.pub.acct#user1/
   userpass.acct/acctpass '

HPConnect = ' #mpeix/nova:forecast,sk,an#pc,
  user2.acct/ acctpass,sales '
```

Using HP-UX syntax

The HP-UX syntax for the HPConnect string is:

'**#hpux**/*Node*:*DBEnvironment* {,*Network*}{,**Convert**}#*LogonString*'

Parameter	Description
Node	A string containing the node name of the database server. Node names can have a maximum of eight characters.
DBEnvironment	The name of the DBEnvironment containing one or more databases. The syntax for the fully qualified DBEnvironment is: *{/path/...}DBEnvironmentName*
Network	(Optional) The networking software on the client computer. For more, see the Hewlett Packard Company PC API documentation.

Parameter	Description
Convert	(Optional) This keyword converts the ROMAN8 character set to the ANSI character set on the client computer. It then converts the data back to ROMAN8 when returning it to the database server.
LogonString	The logon string required to connect to the database. For instructions, see "Specifying the HP-UX logon string" next.

Specifying the HP-UX logon string The HP-UX syntax for the logon string is:

User{:*UserPass*}

> **Password security feature**
> You can use a question mark (?) as a placeholder for a password. Users will be prompted for the actual password when they log on to the database.

Parameter	Description
User	The HP-UX user name.
UserPass	(Optional) The user password.

HP-UX examples The following are two examples of valid HPConnect strings using the HP-UX syntax. Type the HPConnect string *on a single line* in the DBParm field.

HPConnect = ' #hpux/mpesysid:/path/dbe/#userid:passwd '

HPConnect = ' #hpux/nova:/widgets/sales/forecast,sk,an#user2 '

What to do next

For instructions on connecting to the database, see Chapter 4, "Managing Database Connections."

IBM DRDA databases

The IBM **Distributed Relational Database Architecture (DRDA)** is the connection protocol for distributed relational database processing used by IBM relational database products. DRDA includes protocols for communication between an application and a remote database, and communication between databases.

The Powersoft IBM database interface enables you to access the following IBM DRDA databases:

- Database Manager
- Database 2 for OS/2 (DB2/2)
- Database 2 for MVS (DB2/MVS)
- Database 2 for RS/6000 (DB2/6000)

The following sections describe what you need to know to use the Powersoft IBM DRDA database interface in PowerBuilder or InfoMaker.

Getting help with IBM databases

If you need help with any of the procedures in this section, you should:

- Check for a FaxLine document that describes how to connect to your IBM database. For a complete list of available FaxLines, order the *Technical Information Catalog* from the Powersoft FaxLine system.
- Ask your IBM account representative for assistance setting up the database server, client, and networking software at your site.

Features of the Powersoft IBM database interface

This section describes various features that PowerBuilder and InfoMaker support when you connect to an IBM DRDA database with the Powersoft IBM database interface.

Running multiple executables

You can run multiple executable programs that connect to the same IBM database. You can also develop new applications while running an executable program that is connected to the same IBM database.

Modifying the table list

When PowerBuilder or InfoMaker builds the list of tables in the Select Tables dialog box, it references both the SYSIBM.SYSTABLES and SYSIBM.SYSTABAUTH tables. They list only PUBLIC tables and tables for which the user has been granted an explicit authorization level.

You can modify the table list by setting the following DBParm parameters:

- DBAdm
- DelimitIdentifier
- GroupID
- TableCriteria

For instructions, see the descriptions of these DBParms in Chapter 5, "Setting Additional Connection Parameters."

Defining primary keys

The way you define primary keys for an IBM DRDA database in PowerBuilder or InfoMaker depends on the database you are accessing.

Database Manager, DB2/2, and DB2/6000

If you are connected to a Database Manager, DB2/2, or DB2/6000 database, you can define primary keys in either the Create Table dialog box or Alter Table dialog box as long as the column specification does not match an existing index.

DB2/MVS

If you are connected to a DB2/MVS database, you must define primary keys in the Alter Table dialog box. The primary key must match the column specification for an existing unique index.

Defining foreign keys

The way you define foreign keys for an IBM DRDA database in PowerBuilder or InfoMaker depends on the database you are accessing.

Database Manager, DB2/2, and DB2/6000

If you are connected to a Database Manager, DB2/2, or DB2/6000 database and create a foreign key, the rows in the dependent table (the table containing the foreign key) are tested immediately for referential integrity. If any row fails this test, the DBMS rejects the foreign key definition.

DB2/MVS

If you are connected to a DB2/MVS database, the DBMS creates the foreign key definition as long as the dependent table never contained data.

If the dependent table ever contained rows of data, the DBMS creates the foreign key but issues a warning message that places the tablespace in CHECK PENDING status. You must run the IBM Check Data utility to verify the data and restore the tablespace to ACTIVE status.

For instructions on running the Check Data utility, see your IBM documentation.

Tip for defining foreign keys in DB2/MVS databases
If you are connected to a DB2/MVS database, define a foreign key before inserting data into tables.

Adding columns to an existing table

The features supported when you add columns to an existing table in an IBM DRDA database depend on the database you are accessing.

Database Manager, DB2/2, and DB2/6000

Database Manager, DB2/2, and DB2/6000 databases let you add multiple columns to an existing table with a single SQL ALTER TABLE statement.

DB2/MVS

DB2/MVS databases require a separate SQL ALTER TABLE statement for each column you add to an existing table.

Tip for adding columns to existing DB2/MVS tables
To avoid syntax errors in the Alter Table dialog box when you are adding columns to an existing DB2/MVS table, add the columns one at a time.

Supported data types

PowerBuilder and InfoMaker support the following DB2 data types in DataWindows, reports, and embedded SQL:

Server data type	Powersoft data type	Minimum length/value	Maximum length/value
Char	String	1	254
Date	Date	01/01/1000	12/31/3000
Decimal	Decimal	-10E31 + 1	10E31 - 1
Float	Double	-1.79E308	1.79E308
Integer	Long	-2,147,483,648	2,147,483,647
Long VarChar	String	1	32,700
SmallInt	Integer	-32,768	32,767
Time	Time	00:00:00	24:00:00
Timestamp	DateTime	01/01/1000-00.00.00.000000	12/31/3000-24.00.00.000000
VarChar	String	1	4000

Data type conversion
When you retrieve or update columns, PowerBuilder or InfoMaker converts data appropriately between the DB2 server data type and the Powersoft data type.

Date values
Database Manager supports date values of 01/01/0001 to 12/31/999. However, PowerBuilder returns a DataWindow validation error if you try to set a date value before the year 1000 or after the year 3000.

Supported functions

PowerBuilder and InfoMaker support the following types of DB2 functions:

- Scalar functions
- Aggregate functions

Scalar functions

You can use a scalar function in SQL syntax wherever an expression can be used. Most commonly, scalar functions are used to create computed columns in the column list or serve as arguments in a WHERE or ORDER BY clause. Functions can be used as arguments of other functions.

For more about DB2 scalar functions, see the SQL documentation for your DB2 server.

PowerBuilder and InfoMaker support the following DB2 scalar functions:

Function	Return value
CHAR(*expression, format*)	The string representation of a DateTime expression in the format you specify. By default, PowerBuilder or InfoMaker returns DateTime values in ISO format, like this: ' *yyy-mm-dd hh:mm:ss.ffffff* '
DATE(*expression*)	A date value from an expression of type date, timestamp, or string.
DAY(*expression*)	The day value (1 – 31) of a date, timestamp, duration, or string expression.
DAYS(*expression*)	A long integer representing 1 plus the number of days from January 1, 0001, to DATE(*expression*).
DECIMAL(*expression, precision, scale*)	A decimal number derived from *expression* with the specified *precision* and *scale*.
DIGITS(*expression*)	The character string representation of its numeric argument. **Note** Available for DB2/MVS only.
FLOAT(*expression*)	A floating-point representation of the numeric *expression*.
HEX(*expression*)	The hexadecimal representation of its argument. **Note** Available for DB2/MVS only.
HOUR(*expression*)	The hour part of a time, timestamp, or string *expression*.
INTEGER(*expression*)	An integer representation of the numeric *expression*.

Function	Return value
LENGTH(*expression*)	A long integer containing the length of *expression.* The length of a varying length string is the actual length, not the maximum length.
MICROSECOND(*expression*)	The microsecond part of a timestamp, duration, or string *expression.*
MINUTE(*expression*)	The minute part (0 – 59) of a time, timestamp, duration, or string *expression.*
MONTH(*expression*)	The month part of a date, timestamp, duration, or string *expression.*
SECOND(*expression*)	The second part of a time, timestamp, duration, or string *expression.*
SUBSTR(*string, start*{,*length*})	A string derived from *string* beginning at position *start* for a length of *length* characters.
TIME(*expression*)	The time value from a time, timestamp, or string *expression.*
TIMESTAMP(*expression1* {,*expression2*})	A timestamp value from a value or a pair of values. If one argument is specified, it must be of type timestamp or string. If two arguments are specified, *expression1* must be of type date or string, and *expression2* must be of type time or string.
TRANSLATE(*char_str_exp, to_str_exp from_str_exp* {,*pad_char*})	A translated string derived from *char_str_exp* where each character in the *from_str_exp* is translated to the corresponding character in the *to_str_exp*. If specified, *pad_char* will be used to pad the *to_str_exp* if it is shorter than *from_str_exp.*
VALUE(*expression1* {,*expression2, . . .* })	The first non-null result in the series of *expressions*. The data types of the arguments must be compatible. **Note** Not available for DB2/2.
VARGRAPHIC(*expression*)	A pure double-byte character string from *expression.*
YEAR(*expression*)	The year portion of a date, timestamp, duration, or string *expression.*

Aggregate functions

You can use aggregate functions in summary-only SQL statements that contain GROUP BY clauses. Their arguments are a collection of numeric values from a group of rows. Database Manager and DB2/2 allow the syntax:

DISTINCT *column-name*

to be used as the *expression* argument of an aggregate function. If the DISTINCT keyword is specified, duplicate values are eliminated from the collection of numeric values upon which the aggregate function operates. DISTINCT has no effect on the MAX or MIN functions.

Function	Return value
AVG(*expression*)	The average value of *expression* for a set of rows
COUNT(*)	The number of rows in a set of rows
MAX(*expression*)	The maximum value for *expression* in a set of rows
MIN(*expression*)	The minimum value for *expression* in a set of rows
SUM(*expression*)	The total value for *expression* in a set of rows

Preparing to use Database Manager and DB2/2

This section describes how to prepare an IBM Database Manager or DB2/2 database for use with PowerBuilder or InfoMaker.

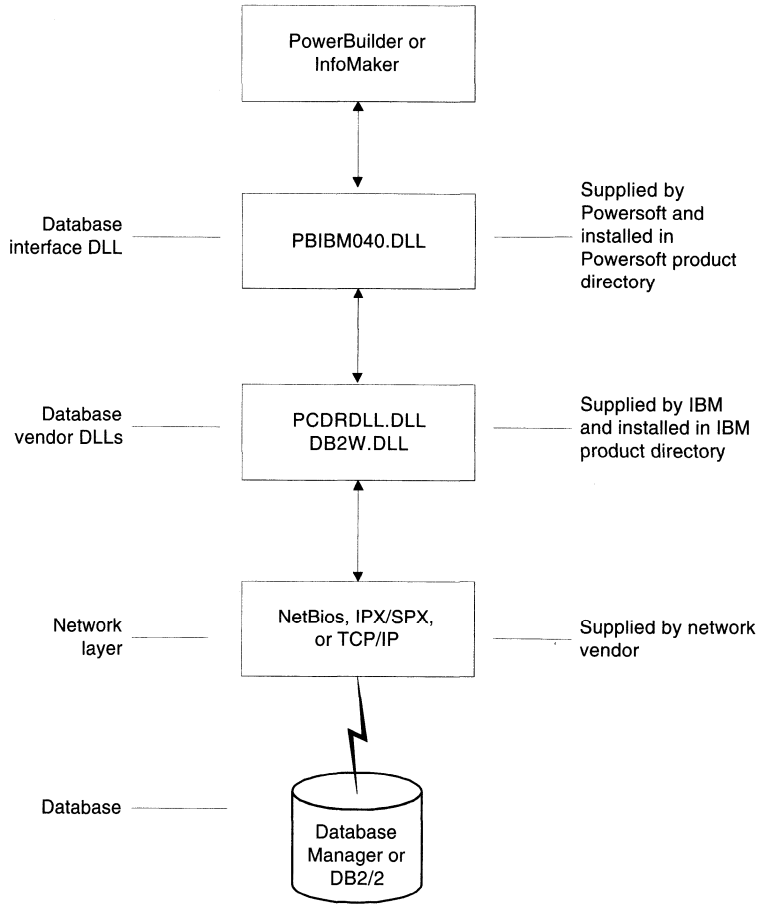
Supported versions

You can access the following from PowerBuilder or InfoMaker:

- All versions of IBM Database Manager
- All versions of IBM DB2/2

Basic software components

The following diagram shows the basic software components you need to access a Database Manager or DB2/2 database using the IBM database interface. The diagram also indicates who supplies each DLL and where it is installed.



Preparing to use the database

Before you define the interface and connect to a Database Manager or DB2/2 database from PowerBuilder or InfoMaker, follow these steps to prepare the database.

❖ To prepare a Database Manager or DB2/2 database:

1 Install the database server software, following the instructions in your Database Manager or DB2/2 documentation.

You must obtain the database server software from IBM.

2 Install and set up the IBM Client Application Enabler (CAE) software on your computer, following the instructions in your CAE documentation and in the appropriate Powersoft FaxLine.

This step includes cataloging the database you want to access on your computer, as follows:

- **CAE 1.0 or 1.1** If you are using CAE Version 1.0 or 1.1, you can use the Catalog Database dialog box in PowerBuilder or InfoMaker to catalog your database. For information, see "Cataloging an IBM database" on page 184.
- **CAE 1.2** If you are using CAE Version 1.2, use the command line interface provided by IBM to catalog your database. You cannot catalog databases for CAE Version 1.2 in PowerBuilder or InfoMaker. For more about the command line interface, see your IBM documentation or the FaxLine document that describes how to connect to an IBM DB2/2 database using CAE Version 1.2.

You must obtain the CAE software from IBM.

3 Edit the SYSTEM.INI file in your Windows product directory as follows to set the NetHeapSize parameter in the [386Enh] section to 76.

```
[386Enh]
NetHeapSize=76
```

4 Install the Powersoft IBM database interface (PBIBM040.DLL) on your computer.

For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide*.

5 Bind PowerBuilder or InfoMaker to each database you want to access.

For instructions, see "Binding PowerBuilder or InfoMaker to your databases" on page 178.

6 If necessary, run the DB2SYSPB.SQL script outside PowerBuilder or InfoMaker to create the PowerBuilder system tables on the database server.

For instructions, see "Creating Powersoft system tables in DB2 databases" on page 247.

7 Start Windows on your computer.

Logging on to the database
When using PowerBuilder or InfoMaker, you can log on to and off from the IBM database automatically. Therefore, if you are upgrading from an earlier version of PowerBuilder or InfoMaker, you no longer need to issue the SQLLOGN2 and SQLLOGF2 commands from a DOS prompt to do this.

Preparing to use DB2/MVS

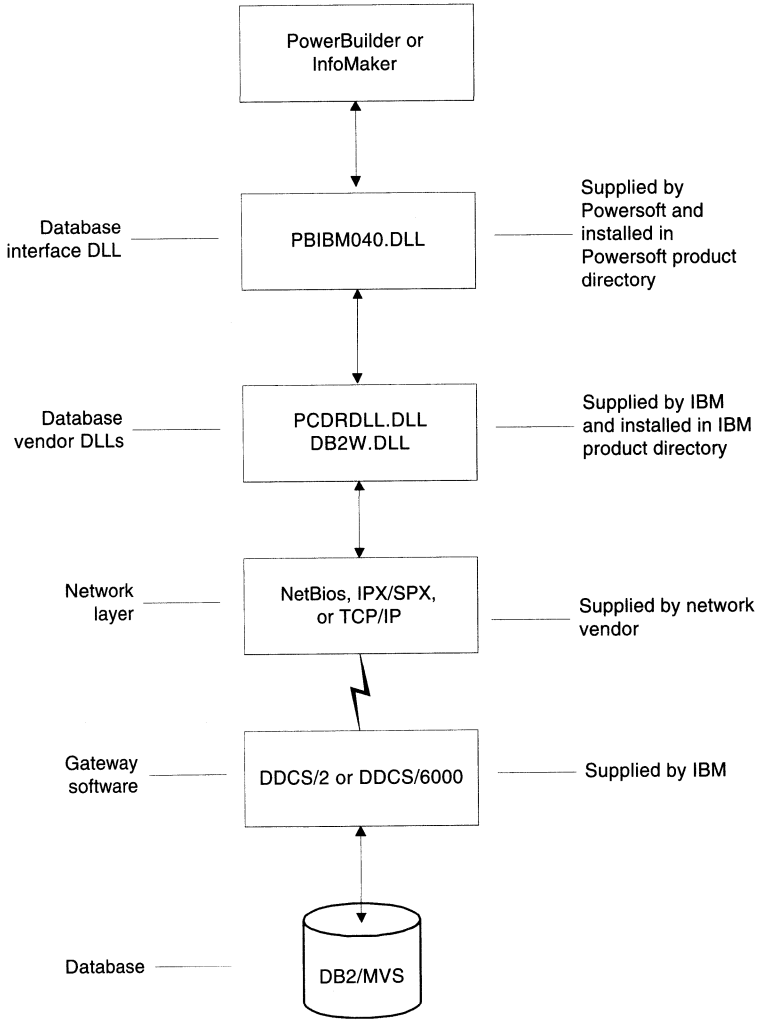
This section describes how to prepare an IBM DB2/MVS database for use with PowerBuilder or InfoMaker.

Supported versions

You can access an IBM DB2/MVS database Version 2 Release 3 or higher from PowerBuilder or InfoMaker.

Basic software components

The following diagram shows the basic software components you need to access a DB2/MVS database using the IBM database interface. The diagram also indicates who supplies each DLL and where it is installed.



Preparing to use the database

Before you define the interface and connect to a DB2/MVS database from PowerBuilder or InfoMaker, follow these steps to prepare the database.

❖ To prepare a DB2/MVS database:

1 Make sure that either the MVS/XA Version 2.2 or MVS/ESA Version 3.1.3 operating system software is installed on the database server.

You must obtain the operating system software from IBM.

2 Install the database server software, following the instructions in your DB2/MVS documentation.

You must obtain the database server software from IBM.

3 Install and set up the following software on the client computer:

- Windows Version 3.1 or OS/2 Version 2.1 operating system
- IBM Client Application Enabler (CAE) software
- Powersoft IBM database interface (PBIBM040.DLL)

This step includes cataloging the database you want to access on your computer, as follows:

- **CAE 1.0 or 1.1** If you are using CAE Version 1.0 or 1.1, you can use the Catalog Database dialog box in PowerBuilder or InfoMaker to catalog your database. For information, see "Cataloging an IBM database" on page 184.
- **CAE 1.2** If you are using CAE Version 1.2, use the command line interface provided by IBM to catalog your database. You cannot catalog databases for CAE Version 1.2 in PowerBuilder or InfoMaker. For more about the command line interface, see your IBM documentation or the FaxLine document that describes how to connect to an IBM DB2/2 database using CAE Version 1.2.

For instructions on installing the Powersoft IBM database interface, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide*.

You must obtain the other client software components from IBM or, for Windows Version 3.1, from Microsoft.

4 Install the software listed in the following table on the database gateway.

You must obtain the networking software from your network vendor, and the other gateway software components from IBM.

If you are using this networking software	Install this software on the database gateway
NetBios	OS/2 Version 2.1 DB2/2 Distributed Database Connection Services/2 (DDCS/2)
TCP/IP	DDCS/6000
Novell IPX/SPX	OS/2 Version 2.1 DB2/2 Version 1.2 DDCS/2 Version 1.2

5 Issue the following command from an OS/2 window on your gateway to verify the connection to the remote DB2/MVS database:

DBM CONNECT TO *remote_database_name*

If the connection succeeds, information similar to the following appears in the OS/2 window:

Database Connection Information

Database Product=DB2 2 3.0
SQL authorization ID=*authorization_ID*
Local database alias=*database_alias_name*

6 Issue the following command to disconnect from the DB2/MVS session after verifying the connection:

DBM CONNECT RESET

7 Bind PowerBuilder or InfoMaker to each database you want to access.

For instructions, see "Binding PowerBuilder or InfoMaker to your databases" on page 178.

8 If necessary, run the DB2SYSPB.SQL script outside PowerBuilder or InfoMaker to create the PowerBuilder system tables on the database server.

For instructions, see "Creating Powersoft system tables in DB2 databases" on page 247.

9 Start Windows on your computer.

Logging on to the database
When using PowerBuilder or InfoMaker, you can log on to and off from the IBM database automatically. Therefore, if you are upgrading from an earlier version of PowerBuilder or InfoMaker, you no longer need to issue the SQLLOGN2 and SQLLOGF2 commands from a DOS prompt to do this.

Preparing to use DB2/6000

This section describes how to prepare an IBM DB2/6000 database for use with PowerBuilder or InfoMaker.

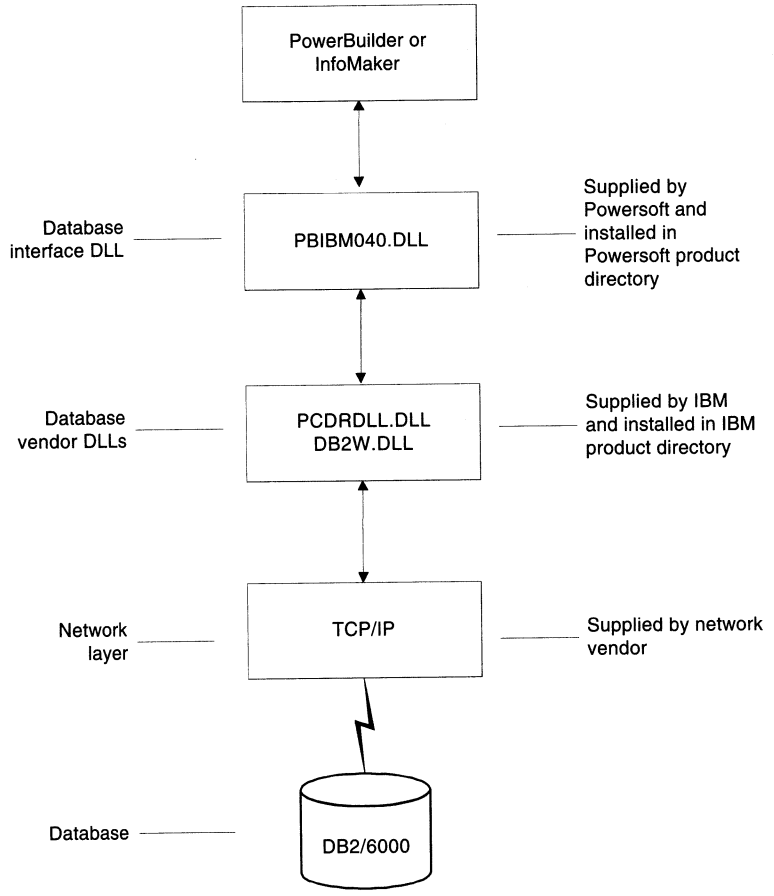
Supported versions

You can access all versions of an IBM DB2/6000 database from PowerBuilder or InfoMaker.

Basic software components

The following diagram shows the basic software components you need to access a DB2/6000 database using the IBM database interface. The diagram also indicates who supplies each DLL and where it is installed.

As shown in this diagram, access to a DB2/6000 database is through TCP/IP. You need Client Application Enabler (CAE) Version 1.2 or CAE for DB2/6000 Version 1.1 to connect to the database through TCP/IP. These versions of CAE contain both PCDRDLL.DLL and DB2W.DLL.



Preparing to use the database

Before you define the interface and connect to a DB2/6000 database from PowerBuilder or InfoMaker, follow these steps to prepare the database.

❖ To prepare a DB2/6000 database:

1 Install the database server software, following the instructions in your DB2/6000 documentation.

You must obtain the database server software from IBM.

2 Install the CAE software on your computer, following the instructions in your CAE documentation and in the appropriate Powersoft FaxLine.

A DB2/6000 connection requires either of the following versions of CAE, which you obtain from IBM:

- CAE Version 1.2
- CAE for DB2/6000 Version 1.1

This step includes cataloging the database you want to access on your computer. Use the command line interface provided by IBM to catalog your database. You *cannot* catalog DB2/6000 databases using the Catalog Database dialog box in PowerBuilder or InfoMaker.

For instructions on using the command line interface, see your IBM documentation or the FaxLine document that describes how to connect to an IBM DB2/2 database using CAE Version 1.2.

3 Install the Powersoft IBM database interface (PBIBM040.DLL) on your computer.

For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide*.

4 Bind PowerBuilder or InfoMaker to each database you want to access.

For instructions, see "Binding PowerBuilder or InfoMaker to your databases" on page 178.

5 If necessary, run the DB2SYSPB.SQL script outside PowerBuilder or InfoMaker to create the PowerBuilder system tables on the database server.

For instructions, see "Creating Powersoft system tables in DB2 databases" on page 247.

6 Start Windows on your computer.

Logging on to the database
When using PowerBuilder or InfoMaker, you can log on to and log off from the IBM database automatically. Therefore, if you are upgrading from an earlier version of PowerBuilder or InfoMaker, you no longer need to issue the SQLLOGN2 and SQLLOGF2 commands from a DOS prompt to do this.

Binding PowerBuilder or InfoMaker to your databases

As part of the setup procedure for connecting to an IBM DRDA database, you must bind PowerBuilder or InfoMaker to each database you want to access. **Binding** is the process by which the DBMS converts a SQL statement into a sequence of internal commands in order to improve data retrieval.

The bind procedure you should follow depends on the version of IBM Client Application Enabler (CAE) software you are using:

- CAE Version 1.0 or 1.1
- CAE Version 1.2 or CAE for DB2/6000 Version 1.1

CAE Version 1.0 or 1.1

When you are running CAE Version 1.0 or 1.1, use the following procedure to bind PowerBuilder or InfoMaker to an IBM database.

❖ To bind PowerBuilder or InfoMaker to a database when using CAE Version 1.0 or 1.1:

1. On the database server, open an OS/2 window and make C: the default drive.
2. Create a new directory to hold the PowerBuilder bind (BND) files and the PBBIND.CMD command file.
3. Make the newly created directory your current working directory.
4. Insert the disk containing the Powersoft IBM database interface into drive A.

If you are using PowerBuilder, the IBM database interface is on Disk 5 of the Deployment Kit. If you are using InfoMaker, it is on Disk 7.

The BND and CMD files are not installed by default when you install the IBM database interface.

5 Copy all of the BND and CMD files from the disk in drive A to the current working directory. For example, type:

```
copy a:*.bnd
copy a:*.cmd
```

6 Issue the PBBIND command as follows to bind PowerBuilder or InfoMaker to the specified IBM database:

PBBIND *database_name*

7 Repeat step 6 for each IBM database you want to access.

CAE Version 1.2 or CAE for DB2/6000 Version 1.1

When you are running CAE Version 1.2 or CAE for DB2/6000 Version 1.1, use the following procedure to bind PowerBuilder or InfoMaker to an IBM database.

❖ To bind PowerBuilder or InfoMaker to a database when using CAE Version 1.2 or CAE for DB2/6000 Version 1.1:

1 Insert the disk containing the Powersoft IBM database interface into drive A of your client workstation.

If you are using PowerBuilder, the IBM database interface is on Disk 5 of the Deployment Kit. If you are using InfoMaker, it is on Disk 7.

2 Change to your PowerBuilder or InfoMaker product directory. For example, type:

```
cd c:\pb040
```

3 Copy all of the PowerBuilder BND files from the disk in drive A to your PowerBuilder or InfoMaker product directory. For example, type:

```
copy a:*.bnd
```

The BND files are not installed by default when you install the IBM database interface.

4 Type the following to bind PowerBuilder or InfoMaker to the specified IBM database.

```
DB2 CONNECT TO database_name
DB2 BIND C:\PB040\@POWERBLD.BND BLOCKING ALL
    GRANT PUBLIC
DB2 CONNECT RESET
```

Change the directory specification (C:\PB040) if the BND files are located in another directory.

5 Repeat step 4 for each IBM database you want to access.

Defining the IBM DRDA database interface

The following table lists the values you should supply for each field in the Database Profile Setup dialog box when defining the Powersoft IBM database interface.

Field	Value
Profile Name	The name of your database profile.
DBMS	**IBM DB2**
User ID	The user ID required to connect to the database. The value you supply depends on the version of IBM Client Application Enabler (CAE) software you are using, as follows: ♦ **CAE 1.0 or 1.1** The user ID must be the same as the authorization ID used in the SQLLOGN2 command to log on to IBM Communication Manager. ♦ **CAE 1.2** The user ID must be the same as the value of the DB2USERID environment variable in the \SQLLIB\SETUP.BAT script. You run this script as part of the CAE installation. (For instructions on using the CAE software, see your IBM CAE documentation or the FaxLine document that describes how to connect to an IBM DB2/2 database using CAE Version 1.2.) In order to properly create the Powersoft repository tables, make sure the first person to connect to the database has sufficient authority to create tables and grant permissions to public.

Field	Value
Password	The password required to connect to your database. The actual password does not display in this field. Small Xs appear in place of the characters you type.
Database Name	The alias name of the IBM database you want to access. The name you specify must match the alias name in the catalog entry for this database on your computer. If you are using CAE Version 1.0 or 1.1 and have not yet cataloged the database on your computer, leave the Database Name field blank. PowerBuilder or InfoMaker will display the Powersoft DRDA Interface dialog box to prompt you for missing information. You can access the Catalog Database dialog box from here to catalog your database. For instructions, see "Cataloging an IBM database" on page 184.
Prompt for Database information during Connect	Select this checkbox if you want to be prompted for connection information when creating or selecting a profile to connect to the database. When you select this checkbox, the Powersoft DRDA Interface dialog box appears in which you can specify the user ID, password, database name, and catalog information. For instructions, see "Being prompted for connection parameters" on page 182.
Server Name	Not applicable for use with PowerBuilder or InfoMaker.
Login ID	Not applicable for use with PowerBuilder or InfoMaker.
Login Password	Not applicable for use with PowerBuilder or InfoMaker.
DBParm	Specify DBMS-specific connection parameters in this field. For information about the DBParm values that you can specify for the IBM DRDA databases, see Chapter 5, "Setting Additional Connection Parameters."

Being prompted for connection parameters

If you omit one or more of the following values from the Database Profile Setup dialog box when defining a Powersoft IBM database interface, PowerBuilder or InfoMaker displays the Powersoft DRDA Interface dialog box to prompt you for the missing information:

- User ID
- Password
- Database name

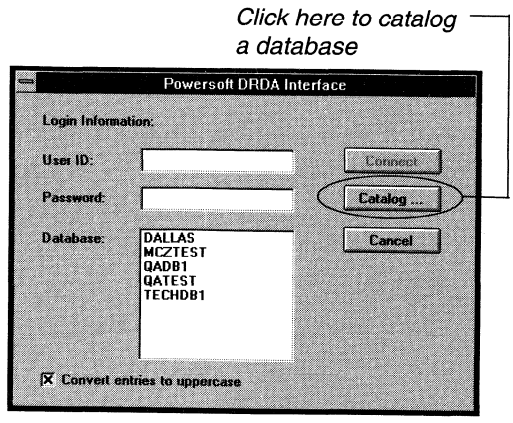
Connecting during application execution
In PowerBuilder or InfoMaker applications that connect to an IBM database, the Powersoft DRDA Interface dialog box appears when the user omits the user ID, password, or database name. This feature enables you to build applications from which multiple users can connect to different IBM databases.

❖ To use the Powersoft DRDA Interface dialog box to supply IBM connection parameters:

1 Select OK in the Database Profile Setup dialog box.

The Powersoft DRDA Interface dialog box appears. If you supplied the user ID, password, or cataloged database name in the Database Profile Setup dialog box, these values appear in the Powersoft DRDA Interface dialog box. If you did not supply these values, these fields are blank.

The Database list contains the alias names of IBM databases cataloged on your computer. If you have not yet cataloged any databases, the Database list will be empty.



2 Specify the user ID and password required to connect to your database.

For instructions on supplying these values, see the table on page 180.

Specifying the password
To connect to an IBM database, you *must* specify a password. Therefore, even if no password is associated with your user ID, specify any non blank value in the Password field. This enables PowerBuilder or InfoMaker to connect to the database while ignoring the invalid password.

3 Specify a database by doing one of the following:

- If you have already cataloged the database you want to access, select its name from the Database list.
- If you have not yet cataloged the database you want to access, click the Catalog button to display the Catalog Database dialog box. This dialog box enables you to catalog a database if you are using CAE software Version 1.0 or 1.1. (For instructions, see "Cataloging an IBM database" on page 184.)

4 If you want to convert the user ID and password values to lowercase, deselect the Convert Entries to Uppercase checkbox.

   By default, PowerBuilder or InfoMaker converts your user ID and password to uppercase. You may need to specify these values as lowercase if, for example, you are accessing a DB2/6000 database. In this situation, deselect the Convert Entries to Uppercase checkbox to make the user ID and password case-sensitive.

5 Click the Connect button. (The Connect button is enabled when you select a database name.)

   PowerBuilder or InfoMaker connects to the specified database and displays the Database Profiles dialog box with the name of your new profile highlighted.

6 Click Cancel to close the Database Profiles dialog box.

Cataloging an IBM database

When you catalog the IBM database you want to access, information about the database is recorded on your computer. This information includes the database name, database alias name, server node name, and database type. PowerBuilder and InfoMaker use this information to connect to the database.

Which method to use

The method you use to catalog the database depends on the version of CAE software you are using, as summarized in the following table:

With this version of CAE	Use this to catalog your database
CAE Version 1.0	Catalog Database dialog box in PowerBuilder or InfoMaker
CAE Version 1.1	Catalog Database dialog box in PowerBuilder or InfoMaker
CAE Version 1.2	IBM command line interface
CAE for DB2/6000 Version 1.1	IBM command line interface

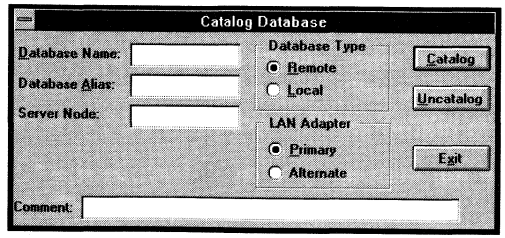
Using the Catalog Database dialog box

If you are using CAE Version 1.0 or 1.1, use the Catalog Database dialog box in PowerBuilder or InfoMaker to catalog your database, as described in the following procedure.

❖ To use the Catalog Database dialog box to catalog an IBM database:

1 Click the Catalog button in the Powersoft DRDA Interface dialog box.

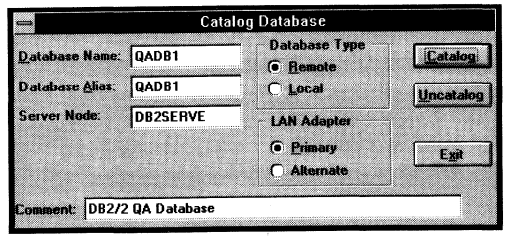
The Catalog Database dialog box appears.



2 Specify values as follows for the database you want to access:

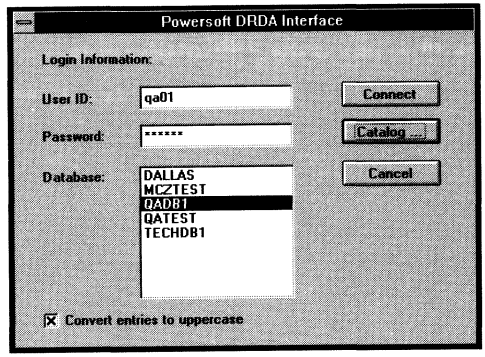
Field	Value
Database Name	The actual name of the database in the directory on the OS/2 server. The maximum length is eight characters.
Database Alias	The name you want to use to access this database on your computer. The alias name will appear in the Database list in the Powersoft DRDA Interface dialog box. If you omit this value, the name you specified in the Database Name field becomes the alias name.
Server Node	In a NetBios environment, the NetBios name of the OS/2 server or DDCS/2 gateway. For information, contact your network administrator.
Comment	A brief description of the database.
Database Type	Click the Remote radio button.
LAN Adapter	Click the Primary radio button.
Catalog	Click this button to catalog the database on your computer with the values you specified.
Uncatalog	Click this button to remove the entry for the specified database from the catalog.
Exit	Click this button to close the Catalog Database dialog box without cataloging the database.

For example, here is a completed Catalog Database dialog box for a DB2/2 database:



3 Click the Catalog button.

The Powersoft DRDA Interface dialog box appears, with the name of the cataloged database highlighted.



4 Click the Connect button in the Powersoft DRDA Interface dialog box.

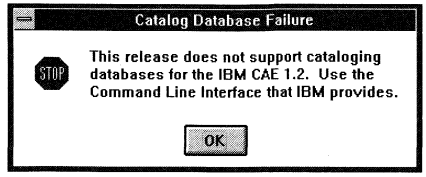
PowerBuilder or InfoMaker connects to the specified database and displays the Database Profiles dialog box with the name of your new profile highlighted.

5 Click Cancel to close the Database Profiles dialog box.

Using the IBM command line interface

If you are using CAE Version 1.2 or CAE for DB2/6000 Version 1.1, you *must* use the command line interface provided by IBM to catalog your database. You *cannot* use the Catalog Database dialog box in PowerBuilder or InfoMaker to catalog the database.

If you have one of these versions of CAE installed and click the Catalog button in the Powersoft DRDA Interface dialog box, the following message box appears:



For instructions on using the command line interface, see your IBM documentation or the FaxLine document that describes how to connect to an IBM DB2/2 database using CAE Version 1.2.

Troubleshooting your IBM DRDA connection

For more about common errors that may occur during an IBM DRDA connection and how to resolve them, see the FaxLine document that describes how to connect to an IBM DB2/2 database using CAE Version 1.2. For a complete list of available FaxLines, order the *Technical Information Catalog* from the Powersoft FaxLine system.

What to do next

For instructions on connecting to the database, see Chapter 4, "Managing Database Connections."

INFORMIX

This section describes how to use the Powersoft INFORMIX IN4 and IN5 database interfaces in PowerBuilder or InfoMaker.

Supported versions

Powersoft supplies *two* INFORMIX database interfaces:

- IN4
- IN5

These interfaces use different Powersoft DLLs and access different versions of the INFORMIX database server software and INFORMIX-NET client software, as listed in the following table:

Powersoft INFORMIX interface	Powersoft interface DLL	INFORMIX database server software	INFORMIX-NET client software
IN4	PBIN4040.DLL	INFORMIX -SE 4.x INFORMIX-SE 5.x INFORMIX-OnLine 4.x INFORMIX-OnLine 5.x	INFORMIX-NET 4.x
IN5	PBIN5040.DLL	INFORMIX-SE 5.x INFORMIX-SE 6.x INFORMIX-OnLine 5.x INFORMIX-OnLine 6.x	INFORMIX-NET 5.x

Supported data types

PowerBuilder and InfoMaker support the following INFORMIX data types in DataWindows, reports, and embedded SQL.

Byte, text, and VarChar data types are *not* supported in INFORMIX SE. Float and real data types are *not* supported in INFORMIX Local.

Byte (a maximum of 2^{31} bytes)	Integer (4 bytes)
Character (1 to 32,511 bytes)	Money
Date	Real
DateTime	Serial
Decimal	SmallInt (2 bytes)
Float	Text (a maximum of 2^{31} bytes)
Interval	VarChar (1 to 255 bytes)

Data type conversion
When you retrieve or update columns, PowerBuilder or InfoMaker converts data appropriately between the INFORMIX data type and the Powersoft data type.

DateTime data type

The DateTime data type is a contiguous sequence of fields. Each field represents a component of time that you want to record. The syntax is:

DATETIME *largest_qualifier* **TO** *smallest_qualifier*

PowerBuilder and InfoMaker default to Year TO Fraction(5).

For a list of qualifiers, see your INFORMIX documentation.

❖ To create your own variation of the DateTime data type:

1 In the Database painter Create Table dialog box, create a table with a DateTime column.

For instructions on creating a table, see the *User's Guide*.

2 Click the Log Only button.

PowerBuilder or InfoMaker writes the CREATE TABLE statement only to the log file; it does *not* submit it to the DBMS.

3 In the Database Administration painter, modify the DateTime syntax and execute the CREATE TABLE statement.

For instructions on using the Database Administration painter, see the *User's Guide*.

Time data type

PowerBuilder and InfoMaker also support a time data type. The time data type is a subset of the DateTime data type. The time data type uses only the time qualifier fields.

Interval data type

The interval data type is one value or a sequence of values that represent a component of time. The syntax is:

INTERVAL *largest_qualifier* **TO** *smallest_qualifier*

PowerBuilder and InfoMaker default to Day(3) TO Day.

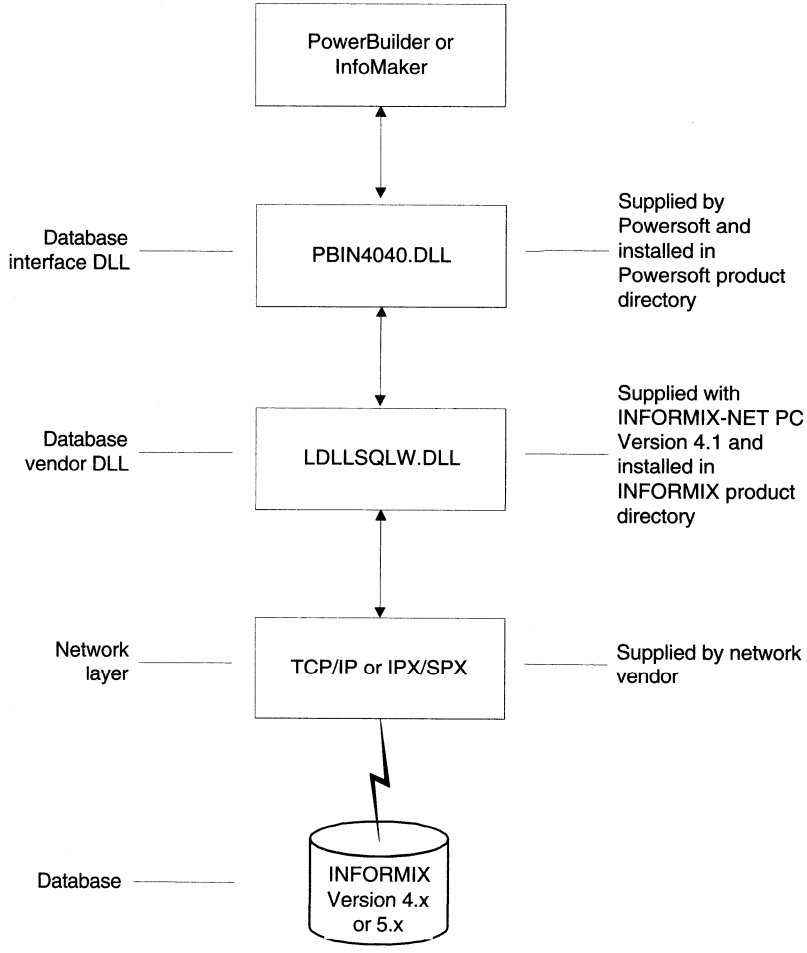
For more about interval data types, see your INFORMIX documentation.

Basic software components

The following diagrams show the basic software components you need to access an INFORMIX database using each INFORMIX database interface. The diagrams also indicate who supplies each DLL and where it is installed.

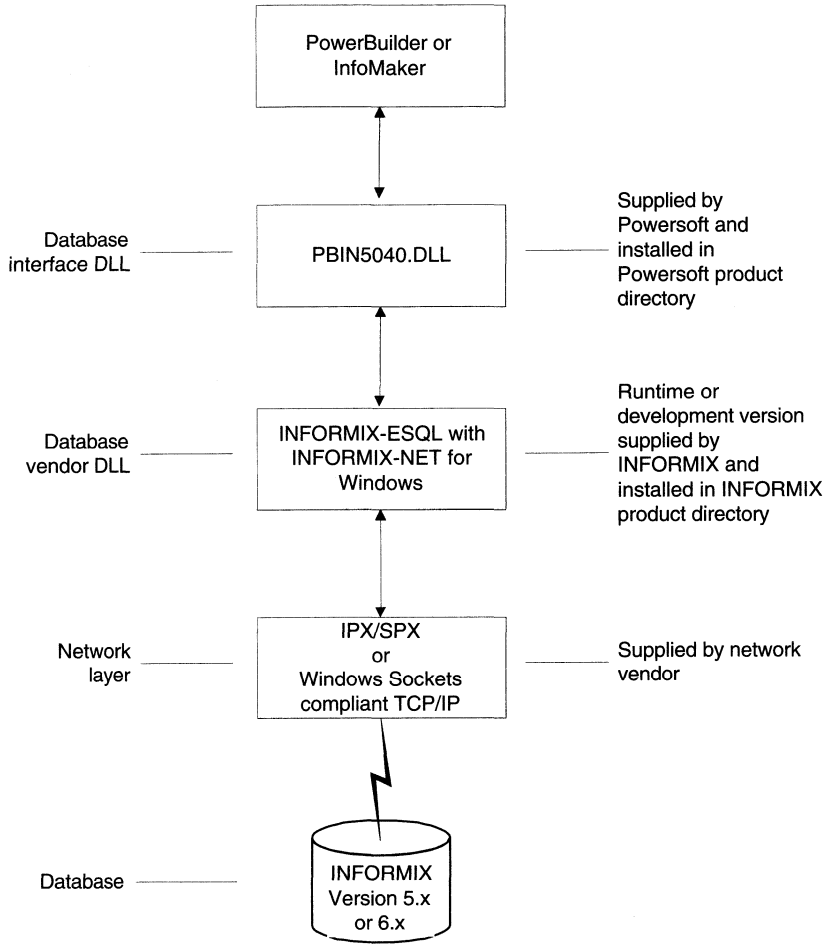
Components of the IN4 interface

To access an INFORMIX Version 4.x or 5.x database using the IN4 interface, you need the software components shown in this diagram.



Components of the IN5 interface

To access an INFORMIX Version 5.x or 6.x database using the IN5 interface, you need the software components shown in this diagram.



Preparing to use the database

The procedure for preparing to use an INFORMIX database depends on whether you access it with the IN4 or IN5 interface.

Preparing to use the database with the IN4 interface

Before you define the interface and connect to an INFORMIX database with the IN4 interface, follow these steps to prepare the database.

❖ To prepare an INFORMIX database that you access with the IN4 interface:

1 Install the database server software and database network software, following the instructions in your INFORMIX documentation.

You must obtain the database server software from INFORMIX.

2 Make sure the appropriate network software is installed at your site. INFORMIX-NET supports the following network protocols:

- Novell IPX/SPX
- Any Windows Sockets compliant TCP/IP protocol

You must obtain the network software from your network vendor.

3 Install the appropriate INFORMIX-NET Version 4.x client software on your computer, following the instructions in your INFORMIX-NET documentation.

You must obtain the INFORMIX-NET client software from INFORMIX.

4 Install the Powersoft IN4 database interface (PBIN4040.DLL) on your computer.

For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide*.

5 Make sure the following files are installed on your computer, and that the directory containing these files appears in your program search path. (These files are usually installed in the INFORMIX \BIN directory.)

- LDLLSQLW.DLL
- SETNET.EXE
- REMSQL.EXE

6 Make sure the following line appears in your AUTOEXEC.BAT file:

```
SET INFORMIXDIR=INFORMIX-NET_directory
```

For example:

```
SET INFORMIXDIR=C:\INFORMIX\BIN
```

7 Load the INFORMIX-NET client software and set connection attributes.

One way to do this is to issue the REMSQL command from DOS. Issuing REMSQL loads the INFORMIX-NET software and sets the attributes in the client software network layer that you will use to communicate with the INFORMIX database server.

To issue the REMSQL command, type the following at a DOS prompt:

remsql -h *hostname* **-u** *username* **-p** *password* **-s** *servicename*

Parameter	Description
hostname	The name of the host computer running the INFORMIX database process
username	Your user ID or name
password	Your password
servicename	The name of the service to which REMSQL should connect

For example:

remsql -h solar -u informix -p informix -s ol4

8 Make sure you can connect to the INFORMIX database server and to the INFORMIX database you want to access from outside PowerBuilder or InfoMaker.

9 Start Windows on your computer.

Preparing to use the database with the IN5 interface

Before you define the interface and connect to an INFORMIX interface with the IN5 interface, follow these steps to prepare the database.

❖ To prepare an INFORMIX database that you access with the IN5 interface:

1 Install the database server software and database network software, following the instructions in your INFORMIX documentation.

- **If you are using INFORMIX Version 5.x** You must install database server software *and* database network software.
- **If you are using INFORMIX Version 6.x or higher** You can install just the database server software. You need *not* install a separate database network component (such as INFORMIX-STAR or INFORMIX-NET) on the server.

You must obtain the database server and database network software from INFORMIX.

2 Install the appropriate INFORMIX-NET for Windows Version 5.x client software on your computer, following the instructions in your INFORMIX-NET documentation.

INFORMIX-NET supports the following network protocols:

- Novell IPX/SPX
- Any Windows Sockets compliant TCP/IP protocol

You must obtain the INFORMIX-NET client software from your network vendor.

3 Install the Powersoft IN5 database interface (PBIN5040.DLL) on your computer.

For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide*.

4 Make sure the following files are installed on your computer, and that the directory containing these files appears in your program search path. (These files are usually installed in the INFORMIX \BIN directory.)

- LDLLSQLW.DLL
- SETNET.EXE

Managing different versions of LDLLSQLW.DLL

LDLLSQLW.DLL is an INFORMIX software module that allows PowerBuilder or InfoMaker to communicate with a remote INFORMIX database server.

Since PowerBuilder and InfoMaker work with Versions 4.x and 5.x of the INFORMIX-NET client software, you may have both the 4.x and 5.x versions of LDLLSQLW.DLL installed on your computer. (Version 4.x of the LDLLSQLW.DLL file is dated before 1994, and Version 5.x is dated 1994 or later.)

To ensure that you use the Version 5.x LDLLSQLW.DLL file with INFORMIX-NET Version 5.x client software, make sure *both* of the following are true:

- Version 5.x of the LDLLSQLW.DLL file appears in your program search path *before* the Version 4.x LDLLSQLW.DLL file.
- Version 4.x of the LDLLSQLW.DLL file does *not* appear in the program search path. If it does, rename or delete it.

5 Make sure the following line appears in your AUTOEXEC.BAT file:

```
SET INFORMIXDIR=INFORMIX-NET_directory
```

For example:

```
SET INFORMIXDIR=C:\INFORMIX\BIN
```

6 Make sure the INFORMIX.INI configuration file in your Windows directory is properly edited for your environment.

When you install INFORMIX-NET Version 5.x client software on your computer, it automatically creates a file named INFORMIX.INI and places it in the Windows directory.

The INFORMIX.INI file contains default parameters that define your network configuration, network protocol, and environment variables. If you omit these values from the database profile when you define the Powersoft INFORMIX database interface, they default to the values specified in your INFORMIX.INI file.

If necessary, you can modify the INFORMIX.INI file by:

- Editing it with any text editor
- Running the Windows-based SETNET utility that comes with INFORMIX-NET for Windows Version 5.x

For instructions on using the INFORMIX.INI file, see your INFORMIX-NET documentation.

7 Make sure you can connect to the INFORMIX database server and to the INFORMIX database you want to access from outside PowerBuilder or InfoMaker.

INFORMIX provides the ILOGIN.EXE demo program for this purpose.

For instructions on using ILOGIN.EXE, see your INFORMIX documentation.

Defining the database interface

This section describes how to define the IN4 and IN5 Powersoft INFORMIX database interfaces.

Defining the IN4 database interface

The following table lists the values you should supply for each field in the Database Profile Setup dialog box when defining the Powersoft IN4 database interface.

Field	Value
Profile Name	The name of your database profile.
DBMS	**IN4 - I-Net v4.x**
User ID	Not applicable for use with PowerBuilder or InfoMaker.
Password	Not applicable for use with PowerBuilder or InfoMaker.
Database Name	The name of the INFORMIX database you want to access.
Prompt for Database information during Connect	Select this checkbox if you want to be prompted for connection information when creating or selecting a profile to connect to the database.
Server Name	Not applicable for use with PowerBuilder or InfoMaker.

Field	Value
Login ID	Not applicable for use with PowerBuilder or InfoMaker.
Login Password	Not applicable for use with PowerBuilder or InfoMaker.
DBParm	Specify DBMS-specific connection parameters in this field. For more about DBParm values that you can specify for INFORMIX, see Chapter 5, "Setting Additional Connection Parameters."

Defining the IN5 database interface

The following table lists the values you should supply for each field in the Database Profile Setup dialog box when defining the Powersoft IN5 database interface.

Field	Value
Profile Name	The name of your database profile.
DBMS	**IN5 - I-Net v5.x**
User ID	The user ID required to connect to your database. This is your user account name on the host computer running the INFORMIX database server. If you do not supply this value, it defaults to the user ID specified in your INFORMIX.INI file. In order to properly create the Powersoft repository tables, make sure the first person to connect to the database has sufficient authority to create tables and grant permissions to public.
Password	The password required to connect to your database. This is your account password on the host computer running the INFORMIX database server. The actual password does not display in this field. Small Xs appear in place of the characters you type. If you do not supply this value, it defaults to the password specified in your INFORMIX.INI file.

Field	Value
Database Name	The name of the INFORMIX database you want to access.
Prompt for Database information during Connect	Select this checkbox if you want to be prompted for connection information when creating or selecting a profile to connect to the database.
Server Name	The name of the host computer running the INFORMIX database server. If you do not supply this value, it defaults to the server (host) name specified in your INFORMIX.INI file.
Login ID	Not applicable for use with PowerBuilder or InfoMaker.
Login Password	Not applicable for use with PowerBuilder or InfoMaker.
DBParm	Specify DBMS-specific connection parameters in this field. For more about DBParm values that you can specify for INFORMIX, see Chapter 5, "Setting Additional Connection Parameters."

Connecting to INFORMIX in a PowerBuilder script

For PowerBuilder developers only
Read this section if you are a PowerBuilder developer connecting to an INFORMIX database in a script.

If you are connecting to an INFORMIX database in a PowerBuilder script, you can obtain the serial number of the row inserted into an INFORMIX table by checking the value of the SQLReturnData attribute of the transaction object.

After a embedded SQL INSERT statement executes, SQLReturnData contains the serial number that uniquely identifies the row inserted into the table.

PowerBuilder updates SQLReturnData following an embedded SQL statement only; it does not update it following a DataWindow operation.

What to do next

For instructions on connecting to the database, see Chapter 4, "Managing Database Connections."

Micro Decisionware Database Gateway Interface for DB2

This section describes how to use the Powersoft Micro Decisionware Database Gateway Interface for DB2 in PowerBuilder or InfoMaker.

Supported versions

You can access a DB2 database using the Micro Decisionware Database Gateway for DB2 Version 2 Release 1 or higher with PowerBuilder or InfoMaker.

Supported data types

PowerBuilder and InfoMaker support the following DB2 data types in DataWindows, reports, and embedded SQL:

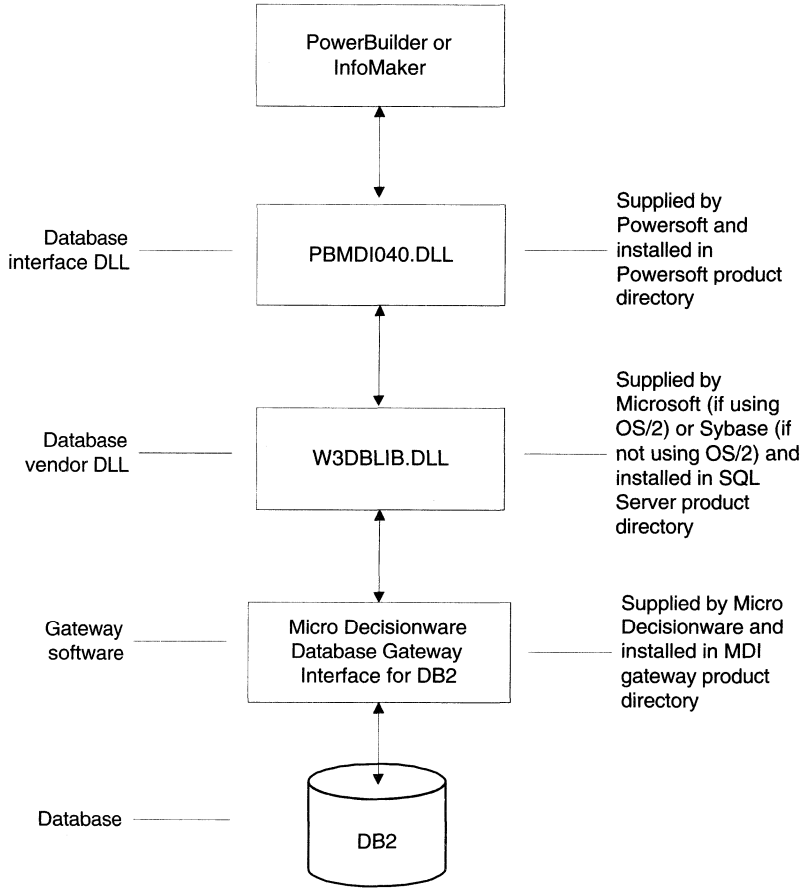
Char (less than 255 characters)	Long VarChar
Date	SmallInt
Decimal	Time
Float	Timestamp
Int	VarChar (less than 255 characters)

Data type conversion

When you retrieve or update columns, the Micro Decisionware Database Gateway converts data appropriately between the DB2 data type and the SQL Server data type. PowerBuilder or InfoMaker then converts the data appropriately between the SQL Server data type and the Powersoft data type.

Basic software components

The following diagram shows the basic software components you need to access a DB2 database using the Micro Decisionware Database Gateway Interface for DB2. The diagram also indicates who supplies each DLL and where it is installed.



Preparing to use the database

Before you define the interface and connect to a DB2 database through the Powersoft Micro Decisionware Database Gateway Interface, follow these steps to prepare the database.

❖ To prepare a DB2 database that you access with the Micro Decisionware Database Gateway Interface:

1 Install the database server software, following the instructions in your DB2 documentation.

You must obtain the DB2 database server software from IBM.

2 Install the Powersoft Micro Decisionware Database Gateway interface (PBMDI040.DLL) on your computer.

For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide.*

3 Install the Micro Decisionware Database Gateway Interface for DB2 software on the gateway machine.

You must obtain the Micro Decisionware Database Gateway Interface for DB2 software from Micro Decisionware, Inc.

4 Make sure only one copy of the file W3DBLIB.DLL resides in the directories listed in your program search path.

For more about the version of W3DBLIB.DLL that you should be using, contact your system administrator or database vendor.

5 If necessary, run the DB2SYSPB.SQL script outside PowerBuilder or InfoMaker to create the PowerBuilder system tables on the database gateway.

For instructions, see "Creating Powersoft system tables in DB2 databases" on page 247.

Defining the database interface

The following table lists the values you should supply for each field in the Database Profile Setup dialog box when defining the Powersoft Micro Decisionware Database Gateway Interface for DB2.

Field	Value
Profile Name	The name of your database profile.
DBMS	**MDI Gateway**
User ID	Not applicable for use with PowerBuilder or InfoMaker.
Password	Not applicable for use with PowerBuilder or InfoMaker.
Database Name	The name of the DB2 database you want to access. Specify this value only if you want to create a table in a database that is *not* the default database.
Prompt for Database information during Connect	Select this checkbox if you want to be prompted for connection information when creating or selecting a profile to connect to the database.
Server Name	The name of the DB2 database server.
Login ID	The login ID of your database server. In order to properly create the Powersoft repository tables, make sure the first person to connect to the database has sufficient authority to create tables and grant permissions to public.
Login Password	The login password of your database server. The actual password does not display in this field. Small Xs appear in place of the characters you type.
DBParm	Specify DBMS-specific connection parameters in this field. For more about DBParm values that you can specify for the Micro Decisionware Database Gateway Interface for DB2, see Chapter 5, "Setting Additional Connection Parameters."

ORACLE

This section describes how to use the Powersoft ORACLE OR6 and OR7 database interfaces in PowerBuilder or InfoMaker.

Supported versions

Powersoft supplies *two* ORACLE database interfaces:

- OR6
- OR7

These interfaces use different Powersoft DLLs and access different versions of the ORACLE database software, as summarized in the following table:

Powersoft ORACLE interface	Powersoft interface DLL	ORACLE database software
OR6	PBOR6040.DLL	ORACLE Version 6
OR7	PBOR7040.DLL	ORACLE Version 7

Supported data types

PowerBuilder and InfoMaker support the following ORACLE data types in DataWindows, reports, and embedded SQL:

Char

Date

Float (ORACLE 7 only)

Long

LongRaw

Number

Raw

VarChar (ORACLE 6 only)

VarChar2 (ORACLE 7 only)

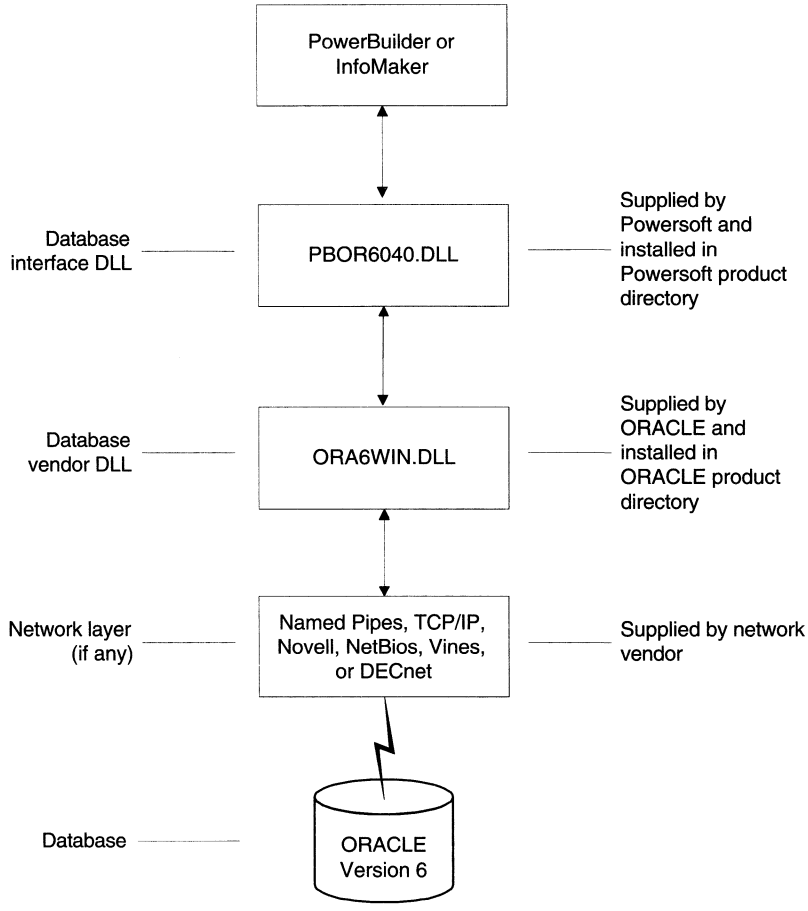
Data type conversion
When you retrieve or update columns, PowerBuilder or InfoMaker converts data appropriately between the ORACLE data type and the Powersoft data type.

Basic software components

The following diagrams show the basic software components you need to access an ORACLE database using the ORACLE database interface. The diagrams also indicate who supplies each DLL and where it is installed.

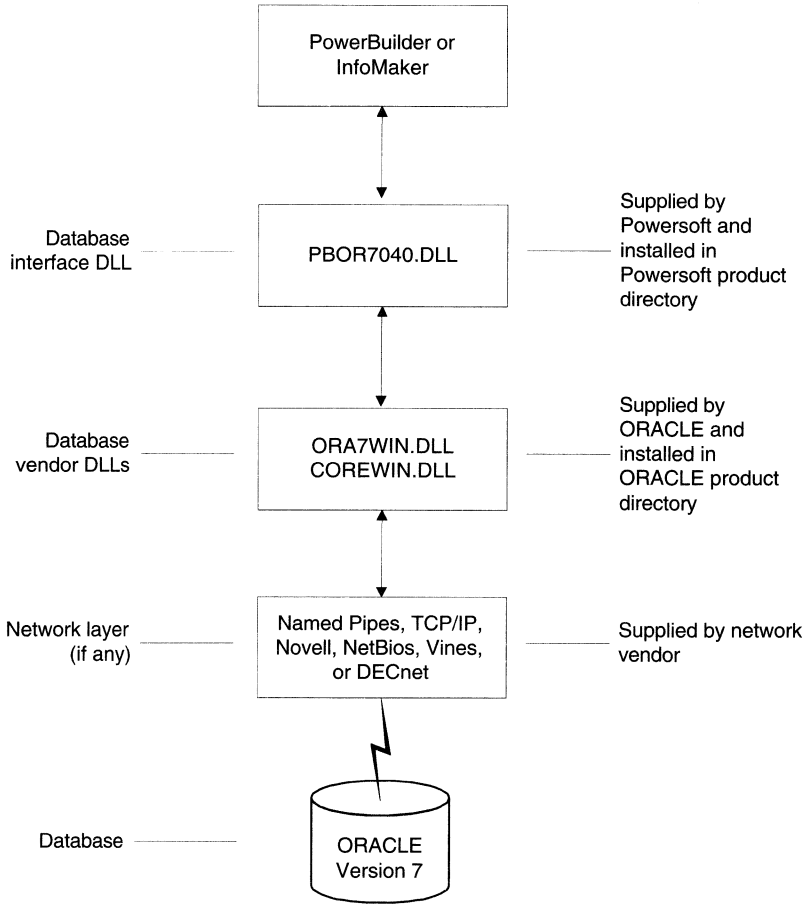
Components of the OR6 interface

To access an ORACLE Version 6 database using the OR6 interface, you need the software components shown in this diagram.



Components of the OR7 interface

To access an ORACLE Version 7 database using the OR7 interface, you need the software components shown in this diagram.



Preparing to use the database

Before you define the interface and connect to an ORACLE database from PowerBuilder or InfoMaker, follow these steps to prepare the database.

This procedure applies to *both* Version 6 and Version 7 ORACLE databases.

❖ To prepare an ORACLE database:

1 Install the ORACLE database server software, following the instructions in your ORACLE documentation.

You must obtain the database server software from Oracle Corporation.

2 Make sure there is only one copy of ORA7WIN.DLL and/or one copy of ORA6WIN.DLL on your computer.

If you want to access both ORACLE Version 6 and ORACLE Version 7 databases, you can have one copy of each file on your computer.

3 Install the SQL*Net client software, following the instructions in your SQL*Net documentation.

You must obtain the SQL*Net client software from Oracle Corporation.

Installing the SQL*Net software places the correct configuration file in the ORACLE directory on your computer. In general, this configuration file is named CONFIG.ORA if you are using SQL*Net for DOS, and ORACLE.INI if you are using SQL*Net for Windows.

4 Install the appropriate Powersoft ORACLE database interface (OR6 or OR7) on your computer.

For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide*.

5 Make sure the files listed in the following table are installed on your computer, and that the directories containing these files appear in your program search path.

ORACLE database	Program search path entries
ORACLE Version 6	PBOR6040.DLL ORA6WIN.DLL SQL*Net client software
ORACLE Version 7	PBOR7040.DLL ORA7WIN.DLL COREWIN.DLL SQL*Net client software

6 Make sure the following line appears in your AUTOEXEC.BAT file to define the SET CONFIG environment variable:

```
SET CONFIG = ORACLE_configuration_file_pathname
```

For example:

```
SET CONFIG = C:\ORACLE\ORACLE.INI
```

7 Determine the network protocol you are using and the SQL*Net driver that it requires.

You need this information to make sure the appropriate SQL*Net driver is loaded on your computer, as described in step 8. You will also need this information when you specify the ORACLE server connect string in the database profile, as described on page 212.

The following table lists frequently used network protocols and their corresponding SQL*Net for DOS and SQL*Net for Windows drivers:

Network protocol	SQL*Net for DOS driver	SQL*Net for Windows driver
DECnet	SQLDNT.EXE	SQLDNT.DLL
Local	SQLPME.EXE	SQLPME.DLL
Named Pipes	SQLNMP.EXE	SQLNMP.DLL
NetBios	SQLNTB.EXE	SQLNTB.DLL
Novell	SQLSPX.EXE	SQLSPX.DLL
TCP/IP	SQLTCP.EXE	SQLTCP.DLL
Vines	SQLVIN.EXE	SQLVIN.DLL

8 Make sure the SQL*Net driver required by your network protocol is loaded on your computer.

9 Make sure you can connect to the ORACLE database server and to the ORACLE database you want to access from outside PowerBuilder or InfoMaker.

10 Start Windows on your computer.

Defining the database interface

The following table lists the values you should supply for each field in the Database Profile Setup dialog box when defining the Powersoft OR6 or OR7 database interface.

Field	Value
Profile Name	The name of your database profile.
DBMS	**OR6 - ORACLE v6.x** *or* **OR7 - ORACLE v7.x**
User ID	Not applicable for use with PowerBuilder or InfoMaker.
Password	Not applicable for use with PowerBuilder or InfoMaker.
Database Name	Not applicable for use with PowerBuilder or InfoMaker.
Prompt for Database information during Connect	Select this checkbox if you want to be prompted for connection information when creating or selecting a profile to connect to the database.
Server Name	The server connect string. This field is required only when using a networked version of the ORACLE database server. If you are using a local version of the ORACLE database server, leave the Server Name field blank. For information, see "Specifying the server connect string" on page 212.
Login ID	The login ID of your database server. In order to properly create the Powersoft repository tables, make sure the first person to connect to the database has sufficient authority to create tables and grant permissions to public.
Login Password	The login password of your database server. The actual password does not display in this field. Small Xs appear in place of the characters you type.
DBParm	Specify DBMS-specific connection parameters in this field. For more about DBParm values that you can specify for ORACLE, see Chapter 5, "Setting Additional Connection Parameters."

Specifying the server connect string

In order to connect to a networked version of ORACLE 6 or ORACLE 7, you must specify the proper connect string or connect descriptor in the Server Name field of the Database Profile Setup window. The connect string or connect descriptor specifies the connection parameters that the ORACLE Windows API uses to access the database.

> **Getting help**
>
> For help determining the proper connect string or connect descriptor for your environment, see your ORACLE documentation or system administrator.

Using a connect string or connect descriptor

The SQL*Net client software you are using determines whether you should specify an ORACLE connect string or connect descriptor in the Server Name field, as summarized in the following table:

If you are using	Specify this in Server Name field
SQL*Net for DOS	ORACLE connect string
SQL*Net for Windows Version 1.0 (SQL*Net V1)	ORACLE connect string
SQL*Net for Windows Version 2.0 (SQL*Net V2)	ORACLE connect descriptor

Specifying a connect string

The syntax of the connect string depends on the ORACLE client software you are using: SQL*Net for DOS or SQL*Net for Windows Version 1.0.

If you are using SQL*Net for DOS, the syntax is:

@ identifier : LogicalServerName

If you are using SQL*Net for Windows Version 1.0, the syntax is:

@ identifier : LogicalServerName : ORACLEInstanceName

Parameter	Description
@	The at (@) sign is required.
identifier	The appropriate SQL*Net communications identifier for your network protocol and driver, as follows: **B** NetBios, SQLNTB driver **D** DECnet, SQLDNT driver **P** Named Pipes, SQLNMP driver **T** TCP/IP, SQLTCP driver **V** Vines, SQLVIN driver **X** Novell, SQLSPX driver
:	The colon (:) is required.
LogicalServerName	The name assigned to the database server.
ORACLEInstanceName	The name assigned to the ORACLE instance you are logging on to.

Example 1 To use TCP/IP with SQL*Net for DOS client software to connect to an ORACLE server named LION, enter the following connect string in the Server Name field of the Database Profile Setup dialog box:

```
@T:LION
```

Example 2 To use NetBios with SQL*Net for Windows Version 1.0 client software to connect to an ORACLE server named TIGER under the default ORACLE instance named ORACLE, enter the following connect string in the Server Name field of the Database Profile Setup dialog box:

```
@B:TIGER:ORACLE
```

Specifying a connect descriptor

If you are using SQL*Net for Windows Version 2.0, you must specify an ORACLE connect descriptor in the Server Name field. The syntax is:

@TNS : *ORACLEServiceName*

Parameter	Description
@	The at (@) sign is required.
TNS	The identifier for the Oracle Transparent Network Substrate (TNS) technology, on which SQL*Net Version 2.0 is based.
:	The colon (:) is required.
ORACLEServiceName	The service name assigned to your server in the TNSNAMES.ORA configuration file. TNSNAMES.ORA is one of the required configuration files for SQL*Net Version 2.0. For instructions on configuring SQL*Net Version 2.0, see your ORACLE documentation.

Example To use SQL*Net for Windows Version 2.0 client software to connect to the service named ORA7UNIX, enter the following connect descriptor in the Server Name field of the Database Profile Setup dialog box:

@TNS:ORA7UNIX

Using ORACLE 7 stored procedures as a data source

You can define DataWindow objects and reports that use an ORACLE 7 PBDBMS stored procedure as their data source. A **stored procedure** is a group of pre compiled and pre optimized SQL statements that performs some database operation. Stored procedures reside on the database server where they can be accessed as needed.

Using an ORACLE 7 stored procedure as a data source involves two general steps:

1 Setting up your ORACLE 7 database server

2 Creating DataWindow objects and reports that use the PBDBMS stored procedure

Setting up the ORACLE 7 database server

The first step in using ORACLE 7 stored procedures is to install special software on the database server and create stored procedures using PBDBMS.Put_Line function calls.

❖ To set up your ORACLE 7 database server:

1 From PowerBuilder or InfoMaker, connect to your ORACLE 7 database as the System user.

2 Change the TerminatorCharacter value in Database preferences to ` (back quote).

If you are using PowerBuilder, you can use the Preferences painter to do this or you can edit the [Database] section of the PB.INI file. If you are using InfoMaker, you must edit the [Database] section IM.INI file to set the TerminatorCharacter value.

For instructions, see the *User's Guide*.

3 In the Database Administration painter, open and execute the PBOR7CAT.SQL script.

The PBOR7CAT.SQL script installs the PBDBMS package on the ORACLE 7 database server. The PBDBMS package enables you to use ORACLE 7 PBDBMS stored procedures as data sources for DataWindow objects and reports.

If you are using PowerBuilder, the PBOR7CAT.SQL file is on Disk 5 of the Deployment Kit. If you are using InfoMaker, it is on Disk 7.

For instructions on executing SQL in the Database Administration painter, see the *User's Guide*.

4 Create the ORACLE 7 stored procedure.

You create the ORACLE 7 stored procedure as you normally do. The only difference is that you must code PBDBMS.Put_Line function calls to build the SQL SELECT statement.

The PBDBMS.Put_Line function takes one parameter, a string. The SELECT statement is a concatenation of the parameters passed to the Put_Line function. You can call Put_Line repeatedly in the procedure (up to 100 times) until you have built the entire SELECT statement.

Example

Here is a sample stored procedure:

```
CREATE PROCEDURE spmdw2 (dbname varchar2, dboth
varchar2)
is mystr varchar2(255);

BEGIN
your code here...
PBDBMS.Put_Line('SELECT Col1,Col2');
PBDBMS.Put_Line('FROM test');
END;
```

The generated SELECT statement will be:

```
SELECT Col1, Col2 FROM test
```

A DataWindow object or report using this procedure will have two retrieval arguments: dbname and dboth.

Creating DataWindow objects and reports using the stored procedure

After you set up the ORACLE 7 database server and create the stored procedure, you create the DataWindow object or report using the stored procedure.

❖ To create a DataWindow object or report using an ORACLE 7 PBDBMS stored procedure as its data source:

1 Set the PBDBMS DBParm parameter to 1 as follows to enable use of an ORACLE 7 PBDBMS stored procedure as a data source:

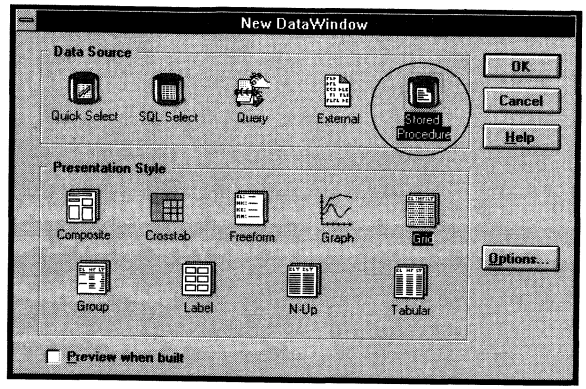
```
DBParm = PBDBMS = 1
```

For instructions, see "Setting DBParm parameters" in Chapter 5, "Setting Additional Connection Parameters."

For more about the PBDBMS parameter, see the description of PBDBMS in Chapter 5, "Setting Additional Connection Parameters."

2 In the DataWindow or Report painter, click the New button in the Select dialog box.

The New DataWindow or New Report dialog box appears. The Stored Procedure icon displays as a data source.



3 Select the Stored Procedure data source and define your DataWindow object or report.

You call a DataWindow Retrieve function as you normally do to get the data. The appropriate SELECT statement is generated by the specified stored procedure and accessed internally by the DataWindow through the PBDBMS package on the database server.

For instructions on defining a DataWindow object or report that uses a stored procedure as its data source, see the *User's Guide*.

Limitations

Using an ORACLE 7 PBDBMS stored procedure as a data source for DataWindow objects and reports has the following limitations:

- The stored procedure must have *no* output parameters.
- The SELECT statement is limited to 255 characters * 100, or 25,500 characters.

What to do next

For instructions on connecting to the database, see Chapter 4, "Managing Database Connections."

SQL Server

This section describes how to use the Powersoft SQL Server database interface in PowerBuilder or InfoMaker.

DB-Library API

The Powersoft SQL Server database interface (PBSYB040.DLL) uses the DB-Library (DBLIB) application programming interface (API) to access the database.

When you connect to a SQL Server database, PowerBuilder or InfoMaker makes the required calls to the API. Therefore, you do not need to know anything about DBLIB to use the database interface.

If you want to use the Sybase Client Library (CT-Lib) API to access a Sybase SQL Server System 10 database, you must install the Powersoft Sybase SQL Server System 10 database interface (PBSYC040.DLL), as described on page 235.

Supported versions

You can access SQL Server Version 1.0 or higher using PowerBuilder or InfoMaker.

Using SQL Server DB-Library cursor processing

To use SQL Server DB-Library cursor processing in PowerBuilder or InfoMaker instead of the default cursor processing, you must set the Release DBParm value to 4.2 as follows when you create the database profile:

```
Release = ' 4.2 '
```

For more about using the Release DBParm parameter with SQL Server, see Chapter 5, "Setting Additional Connection Parameters."

Supported data types

PowerBuilder and InfoMaker support the following SQL Server data types in DataWindows, reports, and embedded SQL:

Binary
Bit
Character (less than 255 characters)
DateTime
Float
Image
Int
Money
SmallInt
Text
TinyInt
VarBinary
VarChar

Updating image and text columns
To update image or text columns within a transaction, AutoCommit must be set to False (the default setting).

For more about using AutoCommit, see Chapter 5, "Setting Additional Connection Parameters."

Short data types

In addition to the data types listed above, PowerBuilder and InfoMaker also support the following short data types in SQL Server Release 4.2. Each of these data types has a storage size of four bytes.

For more about these data types, see your SQL Server documentation.

Real
SmallDateTime
SmallMoney

Data type conversion

When you retrieve or update columns, PowerBuilder or InfoMaker converts data appropriately between the SQL Server data type and the Powersoft data type.

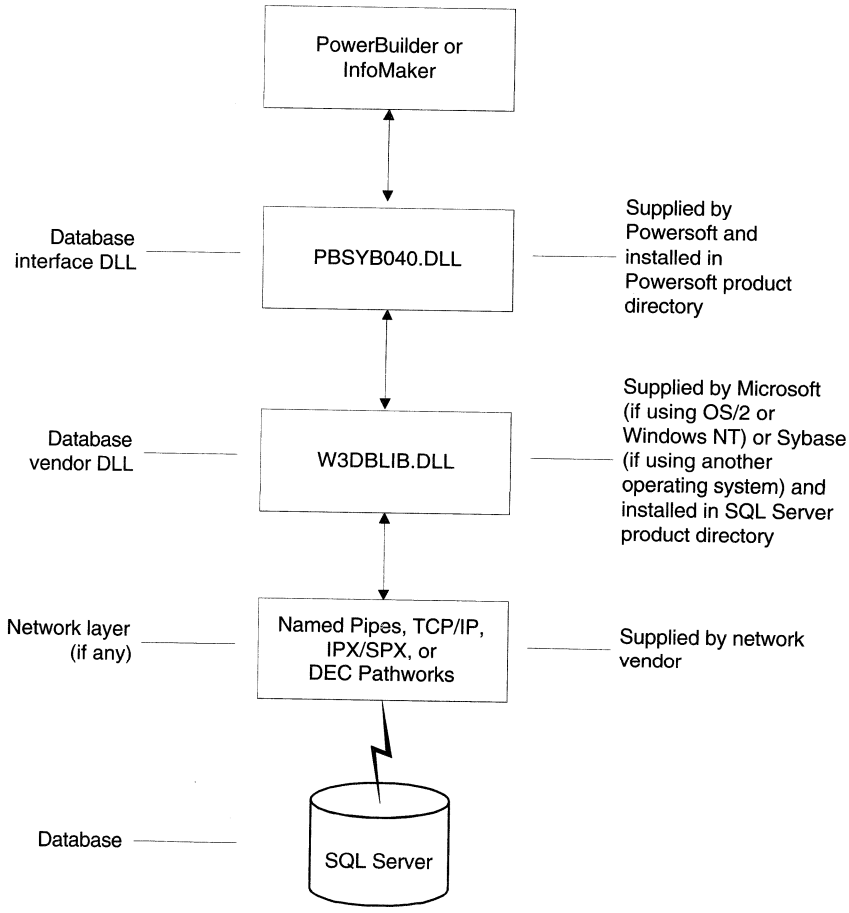
Conversion in PowerBuilder scripts

When you use substitution variables in a PowerBuilder script, PowerBuilder does some special handling when converting from Powersoft data types of double and decimal to substitution strings. A decimal type is always converted to a string that begins with a $. For example, the decimal value 12.34567 is converted to a string value of $12.34567.

A double that has no fractional component is converted to a string with one decimal place if the converted string would cause SQL Server to have an overflow error when parsing the string. For example, the double string 12345678901234 would cause an overflow error so PowerBuilder converts the double to 12345678901234.0.

Basic software components

The following diagram shows the basic software components you need to access a SQL Server database using the SQL Server database interface. The diagram also indicates who supplies each DLL and where it is installed.



Preparing to use the database

Before you define the interface and connect to a SQL Server database from PowerBuilder or InfoMaker, follow these steps to prepare the database.

❖ To prepare a SQL Server database:

1 Install the database server software, following the instructions in your SQL Server documentation.

The operating system on the database server determines which vendor provides the SQL Server database libraries and communication libraries, as follows:

- **Windows NT** If the database server operating system is Windows NT, *Microsoft and Sybase* provide the database libraries and communication libraries.
- **OS/2** If the database server operating system is OS/2, *Microsoft* provides the database libraries and some of the communication libraries.
- **Other** If the database server is running another operating system, *Sybase* provides the database libraries and communication libraries.

2 Make sure the appropriate network software is installed at your site, and that the corresponding network communication DLL is installed on your computer.

The following table lists some frequently used network protocols and their corresponding DLL files. You can obtain DBNMP3.DLL from your SQL Server vendor (Microsoft or Sybase). All other network DLLs listed in the table are available from your network vendor.

Network protocol	Network communication DLLs
DEC Pathworks	WDBDCDE.DLL
Named Pipes	DBNMP3.DLL and NETAPI.DLL
Novell IPX/SPX (Netware 311 NLM)	WDNOVSP.DLL
TCP/IP	WDBFTPTC.DLL (FTP Software) *or* WDBHPTCP.DLL (Hewlett Packard ARPA Services) *or* WDBNOVTC.DLL (Novell LAN Workplace) *or* WDBWOLTC.DLL (Wollongong Group)

3 Install the SQL Server Windows client software on your computer.

The client software includes the file W3DBLIB.DLL, which contains the DB-Library programs. The Powersoft SQL Server interface communicates with the database through W3DBLIB.DLL. W3DBLIB communicates with the network DLL installed on your computer.

You must obtain the client software from Microsoft or Sybase.

4 Install the Powersoft SQL Server database interface (PBSYB040.DLL) on your computer.

For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide*.

5 If you are using a network protocol *other* than Named Pipes, make sure the WIN.INI file in your Windows directory contains a [sqlserver] section that is properly edited for your environment.

The [sqlserver] section in your WIN.INI file should contain an entry for each SQL Server database server that you want to access. Each entry should include the following information:

- Server name
- Network communication DLL name
- Internet address (in TCP/IP environments)

Here is a sample [sqlserver] entry for a database server named Sales in a Novell TCP/IP environment:

```
[sqlserver]
sales = wdbnovtc, 192.1.200.27
```

When you connect to SQL Server, PowerBuilder or InfoMaker passes information from your database profile to W3DBLIB.DLL. This information includes the database server name. W3DBLIB then checks the WIN.INI file to find the corresponding entry for this server name to determine how to connect to it. At this point, either of the following can happen:

- If W3DBLIB.DLL finds an entry for this server in the WIN.INI file, it loads the specified network DLL and connects to the database.
- If W3DBLIB.DLL does *not* find an entry for this server in WIN.INI, or if you are using a Named Pipes network protocol, W3DBLIB loads DBNMP3.DLL and NETAPI.DLL and connects to the database.

6 Make sure only *one* copy of each of the following files is installed on your computer's hard drive:

- PBSYB040.DLL
- W3DBLIB.DLL
- The appropriate network communication DLL listed in the table on page 222

For more about the versions of W3DBLIB.DLL and the network DLLs that you should be using, see your system administrator, database vendor, or network vendor.

7 Run the PBSYB.SQL and, if necessary, the PBSYBRT.SQL scripts outside PowerBuilder or InfoMaker to install required Powersoft stored procedures into the master database and on the deployment machine.

For instructions, see "Installing Powersoft stored procedures in SQL Server databases" on page 250.

8 Make sure you can connect to the database server and to the SQL Server database you want to access from outside PowerBuilder or InfoMaker.

9 Start Windows on your computer.

Defining the database interface

The following table lists the values you should supply for each field in the Database Profile Setup dialog box when defining the Powersoft SQL Server database interface.

Field	Value
Profile Name	The name of your database profile.
DBMS	**SYB - SQL Server v4.x**
User ID	Not applicable for use with PowerBuilder or InfoMaker.
Password	Not applicable for use with PowerBuilder or InfoMaker.
Database Name	The name of the SQL Server database you want to access.

Field	Value
Prompt for Database information during Connect	Select this checkbox if you want to be prompted for connection information when creating or selecting a profile to connect to the database.
Server Name	The name of the SQL Server database server.
Login ID	The login ID of your database server. In order to properly create the Powersoft repository tables, make sure the first person to connect to the database has sufficient authority to create tables and grant permissions to public.
Login Password	The login password of your database server. The actual password does not display in this field. Small Xs appear in place of the characters you type.
DBParm	Specify DBMS-specific connection parameters in this field. For more about DBParm values that you can specify for SQL Server, see Chapter 5, "Setting Additional Connection Parameters."

What to do next

For instructions on connecting to the database, see Chapter 4, "Managing Database Connections."

SQLBase

This section describes how to use the Powersoft SQLBase database interface in PowerBuilder or InfoMaker.

Supported data types

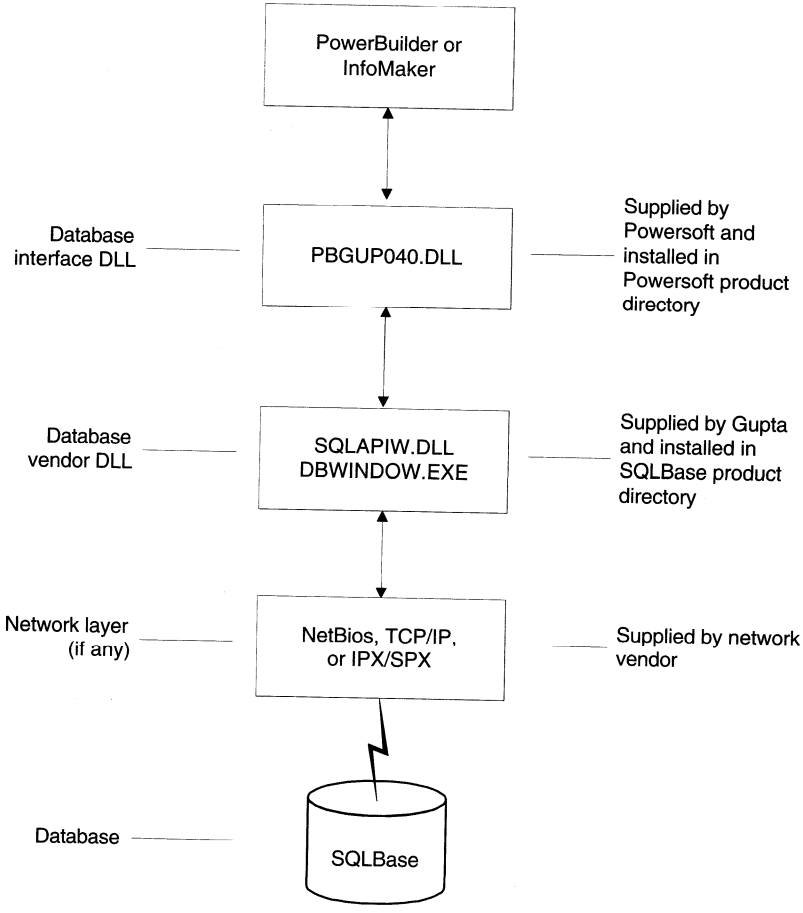
PowerBuilder and InfoMaker support the following SQLBase data types in DataWindows, reports, and embedded SQL:

- Char
- Date
- DateTime (supports microseconds)
- Decimal
- Float
- Integer
- LongVar (long)
- Number
- SmallInt
- Time (supports microseconds)
- VarChar

Data type conversion
When you retrieve or update columns, PowerBuilder or InfoMaker converts data appropriately between the SQLBase data type and the Powersoft data type.

Basic software components

The following diagram shows the basic software components you need to access a SQLBase database using the SQLBase database interface. The diagram also indicates who supplies each DLL and where it is installed.



Preparing to use the database

Before you define the interface and connect to a SQLBase database from PowerBuilder or InfoMaker, follow these steps to prepare the database.

❖ To prepare a SQLBase database:

1 Install the database software on your computer or on a network server, following the instructions in your SQLBase documentation.

The default SQLBase installation places your program files in the \GUPTA directory and your database files in the \SQLBASE directory.

You must obtain the SQLBase software from Gupta Corporation.

2 Install the Powersoft SQLBase database interface (PBGUP040.DLL) on your computer.

For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide*.

3 Make sure the following SQLBase files are installed on your computer, and that the directories containing these files appear in your program search path. (These files usually reside in the \GUPTA and \SQLBASE directories.)

- SQLAPIW.DLL
- DBWINDOW.EXE
- SQL.INI
- ERROR.SQL
- COUNTRY.SQL (SQLBase Version 5.0 or later)
- MESSAGE.SQL (SQLBase Version 5.0 or later)
- Those communication DLLs uncommented in the [winclient.dll] section of your SQL.INI file
- SQLBase database files

4 Make sure the SQL.INI configuration file:

- Is in the same directory as the DBWINDOW.EXE file. (This is generally the \GUPTA directory.)
- Appears only once in the directories in your program search path.
- Is properly edited for your environment. This includes:
  - Setting the DBDIR variable in the [dbwindow] section to the directory containing your SQLBase database. For example:

    ```
    dbdir = c:\sqlbase
    ```

  - Uncommenting the appropriate communication DLLs in the [winclient.dll] section.

For instructions on using and editing the SQL.INI file, see your SQLBase documentation and the comments in the SQL.INI file.

Defining the database interface

The following table lists the values you should supply for each field in the Database Profile Setup dialog box when defining the Powersoft SQLBase database interface.

Field	Value
Profile Name	The name of your database profile.
DBMS	**GUPTA**
User ID	The user ID required to connect to your database. In order to properly create the Powersoft repository tables, make sure the first person to connect to the database has sufficient authority to create tables and grant permissions to public.
Password	The password required to connect to your database. The actual password does not display in this field. Small Xs appear in place of the characters you type.
Database Name	The name of the SQLBase database you want to access.
Prompt for Database information during Connect	Select this checkbox if you want to be prompted for connection information when creating or selecting a profile to connect to the database.
Server Name	Not applicable for use with PowerBuilder or InfoMaker.
Login ID	Not applicable for use with PowerBuilder or InfoMaker.
Login Password	Not applicable for use with PowerBuilder or InfoMaker.
DBParm	Specify DBMS-specific connection parameters in this field. For more about DBParm values that you can specify for SQLBase, see Chapter 5, "Setting Additional Connection Parameters."

What to do next

For instructions on connecting to the database, see Chapter 4, "Managing Database Connections."

Sybase Net-Gateway Interface for DB2

This section describes how to use the Powersoft Sybase Net-Gateway Interface for DB2 with PowerBuilder or InfoMaker.

Supported versions

You can access a DB2 database using the Sybase Net-Gateway Interface for DB2 Version 2.0 or higher with PowerBuilder or InfoMaker.

Supported data types

PowerBuilder and InfoMaker support the following DB2 data types in DataWindows, reports, and embedded SQL:

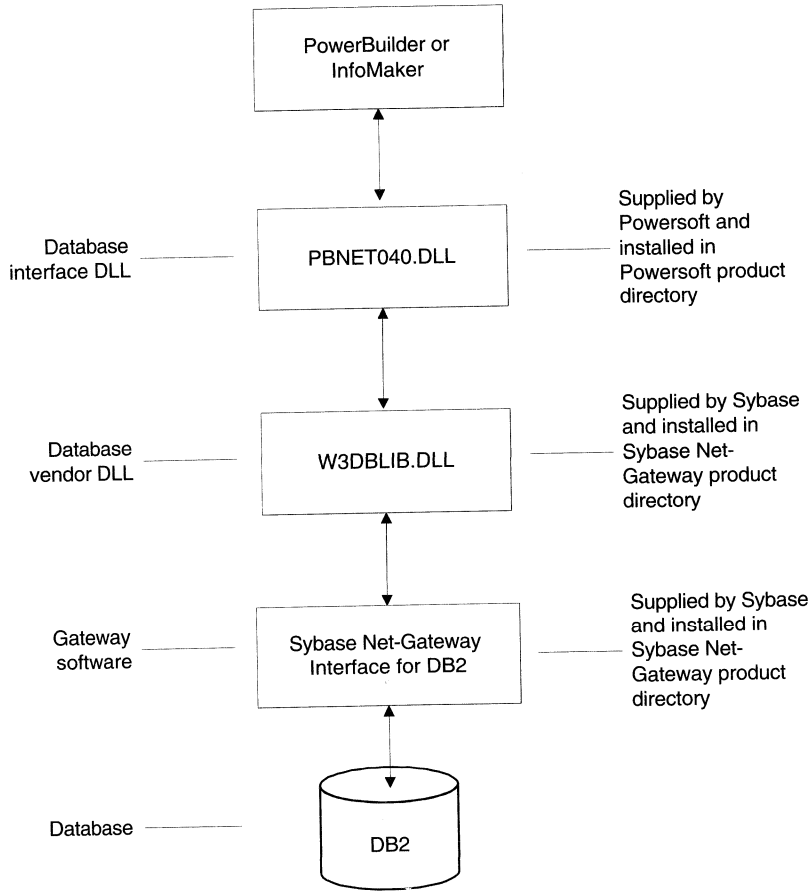
Char (less than 255 characters)	Long VarChar
Date	SmallInt
Decimal	Time
Float	Timestamp
Int	VarChar (less than 255 characters)

Data type conversion

When you retrieve or update columns, the Sybase Net-Gateway converts data appropriately between the DB2 data type and the SQL Server data type. PowerBuilder or InfoMaker then converts the data appropriately between the SQL Server data type and the Powersoft data type.

Basic software components

The following diagram shows the basic software components you need to access a DB2 database using the Sybase Net-Gateway Interface for DB2. The diagram also indicates who supplies each DLL and where it is installed.



Preparing to use the database

Before you define the interface and connect to a DB2 database through the Powersoft Sybase Net-Gateway Interface, follow these steps to prepare the database.

❖ To prepare a DB2 database that you access with the Sybase Net-Gateway Interface:

1 Install the database server software, following the instructions in your DB2 documentation.

You must obtain the DB2 database server software from IBM.

2 Install the Powersoft Sybase Net-Gateway Interface (PBNET040.DLL) on your computer.

For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide*.

3 Install the Sybase Net-Gateway Interface for DB2 software on the gateway machine.

You must obtain the Sybase Net-Gateway software from Sybase Corporation.

4 Make sure only one copy of the file W3DBLIB.DLL resides in the directories listed in your program search path.

For more about the version of W3DBLIB.DLL that you should be using, see your system administrator or database vendor.

5 If necessary, run the DB2SYSPB.SQL script outside PowerBuilder or InfoMaker to create the PowerBuilder system tables on the database gateway.

For instructions, see "Creating Powersoft system tables in DB2 databases" on page 247.

Defining the database interface

The following table lists the values you should supply for each field in the Database Profile Setup dialog box when defining the Powersoft Sybase Net-Gateway Interface for DB2.

Field	Value
Profile Name	The name of your database profile.
DBMS	**NETGATEWAY**
User ID	Not applicable for use with PowerBuilder or InfoMaker.
Password	Not applicable for use with PowerBuilder or InfoMaker.
Database Name	The name of the DB2 database you want to access. Specify this value only if you want to create a table in a database that is *not* the default database.
Prompt for Database information during Connect	Select this checkbox if you want to be prompted for connection information when creating or selecting a profile to connect to the database.
Server Name	The name of the DB2 database server.
Login ID	The login ID of your database server. In order to properly create the Powersoft repository tables, make sure the first person to connect to the database has sufficient authority to create tables and grant permissions to public.
Login Password	The login password of your database server. The actual password does not display in this field. Small Xs appear in place of the characters you type.
DBParm	Specify DBMS-specific connection parameters in this field. For more about DBParm values that you can specify for the Sybase Net-Gateway Interface for DB2, see Chapter 5, "Setting Additional Connection Parameters."

What to do next

For instructions on connecting to the database, see Chapter 4, "Managing Database Connections."

Sybase SQL Server System 10

This section describes how to use the Powersoft Sybase SQL Server System 10 database interface in PowerBuilder or InfoMaker.

Client Library API

The Powersoft Sybase SQL Server System 10 database interface (PBSYC040.DLL) uses the Client Library (CT-Lib) application programming interface (API) to access the database.

When you connect to a Sybase System 10 database, PowerBuilder or InfoMaker makes the required calls to the API. Therefore, you do not need to know anything about CT-Lib to use the database interface.

If you want to use the DB-Library (DBLIB) API to access a SQL Server database, you must install the Powersoft SQL Server database interface (PBSYB040.DLL), as described on page 218.

Supported versions

You can access Sybase SQL Server Release 10.0.1 or higher using PowerBuilder or InfoMaker.

Supported data types

PowerBuilder and InfoMaker support the following Sybase SQL Server System 10 data types in DataWindows, reports, and embedded SQL:

Binary	Money
Bit	Numeric
Char (less than 255 characters)	Real
DateTime	SmallDateTime
Decimal	SmallInt
Double precision	SmallMoney
Float	Text
Identity	TinyInt
Image	VarBinary
Int	VarChar

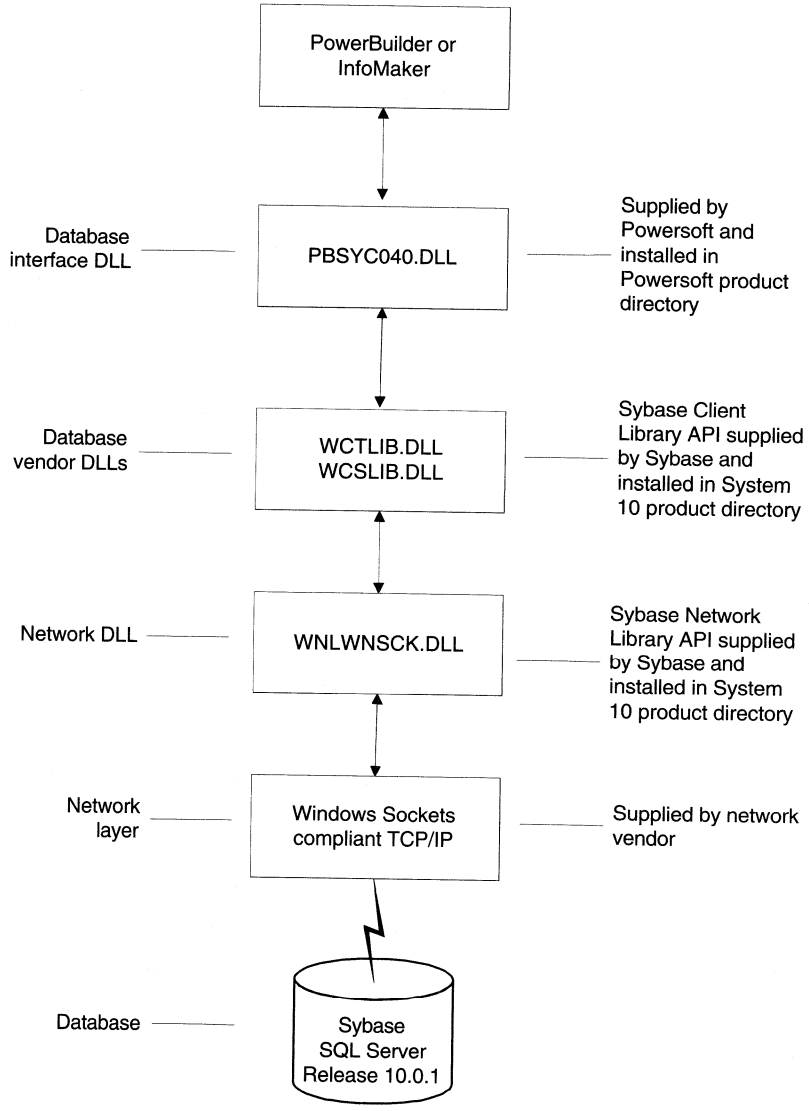
Data type conversion

When you retrieve or update columns, PowerBuilder or InfoMaker converts data appropriately between the Sybase SQL Server System 10 data type and the Powersoft data type.

A double that has no fractional component is converted to a string with one decimal place if the converted string would cause Sybase SQL Server System 10 to have an overflow error when parsing the string. For example, the double string 12345678901234 would cause an overflow error so it is converted to 12345678901234.0.

Basic software components

The following diagram shows the basic software components you need to access a Sybase SQL Server System 10 database. The diagram also indicates who supplies each DLL and where it is installed.



Preparing to use the database

Before you define the interface and connect to a Sybase SQL Server System 10 database from PowerBuilder or InfoMaker, follow these steps to prepare the database.

❖ To prepare a Sybase SQL Server System 10 database:

1 Install the database server software, following the instructions in your Sybase SQL Server System 10 documentation.

You must obtain the database server software from Sybase Corporation.

2 Make sure the appropriate network software is installed at your site and properly configured for your environment.

Sybase SQL Server System 10 requires the use of a Windows Sockets compliant TCP/IP network protocol.

You must obtain the network software from your network vendor.

3 Install the following client software on your computer:

- Sybase Open Client Client-Library for Windows, Release 10.0.1 or higher
- Sybase Net-Library for Windows, Release 10.0.1 or higher, including the Windows Sockets Driver
- Windows Sockets compliant TCP/IP software

The Open Client software includes the file WCTLIB.DLL, which contains the CT-Library programs. The Powersoft Sybase System 10 interface communicates with the database through WCTLIB.DLL. WCTLIB communicates with the Sybase Net-Library DLL (WNLWNSCK.DLL) installed on your computer.

You must obtain the Open Client and Net-Library software from Sybase, and the TCP/IP client software from your network vendor.

4 Install the Powersoft Sybase SQL Server System 10 database interface (PBSYC040.DLL) on your computer.

For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide*.

5 Make sure the SQL.INI configuration file for the Sybase Open Client software is correctly configured for your environment.

The SQL.INI file is installed by default in the \INI subdirectory of your Sybase System 10 product directory. It should contain an entry for each SQL Server database server that you want to access. Each entry should include the following information:

- Server name
- Sybase Net-Library for Windows driver (DLL) name
- Server Internet address or alias name
- TCP/IP port number

The basic format of an entry in the SQL.INI file is as follows:

```
[server_name]
WIN3_QUERY=driver,address,port_number
```

Here is a sample entry in the SQL.INI file for a database server named Sales. WNLWNSCK is the Windows Sockets Net-Library driver, 192.1.200.27 is the server's Internet address, and 5001 is the server's port number.

```
[sales]
WIN3_QUERY=wnlwnsck,192.1.200.27,5001
```

When you connect to Sybase System 10, PowerBuilder or InfoMaker passes information from your database profile to WCTLIB.DLL. This information includes the database server name, which *must* match the server name specified in the SQL.INI file.

WCTLIB then checks the SQL.INI file to find the corresponding entry for this server name to determine how to connect to it. When WCTLIB finds the appropriate entry, it loads the specified Net-Library driver and connects to the database.

For instructions on using the SQL.INI file, see your Sybase Open Client documentation.

6 Run the PBSYB.SQL and PBSYC.SQL scripts outside PowerBuilder or InfoMaker to install required Powersoft stored procedures in the master database.

For instructions, see "Installing Powersoft stored procedures in SQL Server databases" on page 250.

7 Make sure you can connect to the database server and to the SQL Server System 10 database you want to access from outside PowerBuilder or InfoMaker.

To verify the connection, use the WSYBPING utility to check that you can reach the database server from your computer. WSYBPING is a network diagnostic utility included with the Sybase Net-Library for Windows software. The WSYBPING utility detects whether the process you want to access is listening at the specified Internet address.

For instructions on using WSYBPING, see your Sybase Open Client/Server documentation.

8 Start Windows on your computer.

Defining the database interface

The following table lists the values you should supply for each field in the Database Profile Setup dialog box when defining the Powersoft SQL Server System 10 database interface.

Field	Value
Profile Name	The name of your database profile.
DBMS	**SYC - Sybase System 10**
User ID	Not applicable for use with PowerBuilder or InfoMaker.
Password	Not applicable for use with PowerBuilder or InfoMaker.
Database Name	The name of the SQL Server System 10 database you want to access.
Prompt for Database information during Connect	Select this checkbox if you want to be prompted for connection information when creating or selecting a profile to connect to the database.
Server Name	The name of the SQL Server System 10 database server. This name *must* match the server name specified in your SQL.INI configuration file. For more about the SQL.INI file, see "Preparing to use the database" on page 238.

Field	Value
Login ID	The login ID of your database server. In order to properly create the Powersoft repository tables, make sure the first person to connect to the database has sufficient authority to create tables and grant permissions to public.
Login Password	The login password of your database server. The actual password does not display in this field. Small Xs appear in place of the characters you type.
DBParm	Specify DBMS-specific connection parameters in this field. For more about DBParm values that you can specify for Sybase SQL Server System 10, see Chapter 5, "Setting Additional Connection Parameters."

What to do next

For instructions on connecting to the database, see Chapter 4, "Managing Database Connections."

XDB

This section describes how to use the Powersoft XDB database interface in PowerBuilder or InfoMaker.

Supported versions

You can access XDB Versions 2.41 and 3.0 using PowerBuilder or InfoMaker.

Using XDB Version 3.0

To connect to an XDB Version 3.0 database from PowerBuilder or InfoMaker, you must set the Release DBParm value to 3.00 as follows when you create the database profile:

Release = ' 3.00 '

If you are using XDB Version 2.41 you need *not* specify a value for Release, because the default setting is the following:

Release = ' 2.41 '

For more about using the Release DBParm parameter with XDB, see Chapter 5, "Setting Additional Connection Parameters."

Supported data types

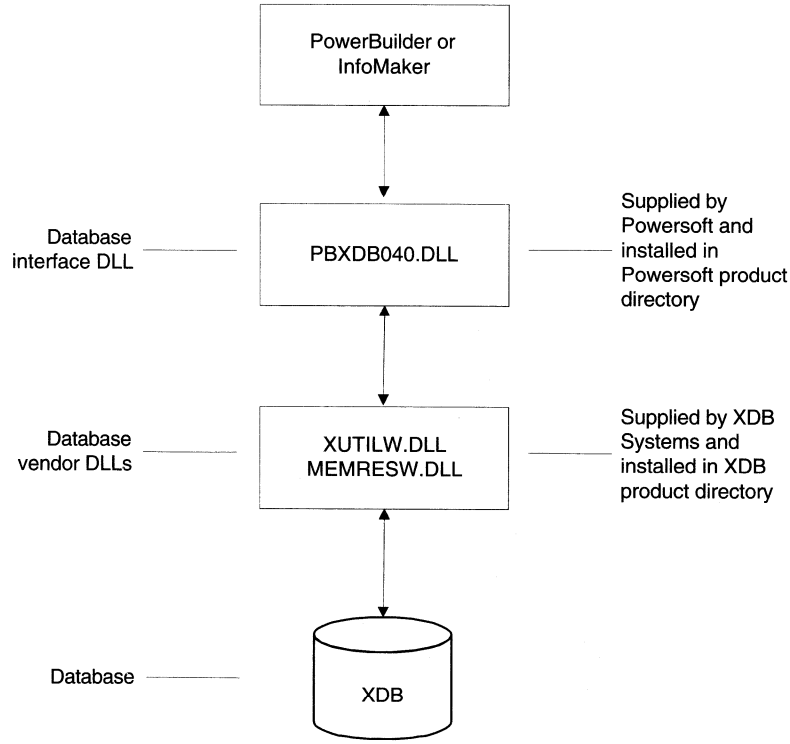
PowerBuilder and InfoMaker support the following XDB data types in DataWindows, reports, and embedded SQL:

Char (less than 1500 characters)	Money
Date	Number
Decimal	SmallInt
Float	Time (supports microseconds)
Integer	Timestamp

Data type conversion
When you retrieve or update columns, PowerBuilder or InfoMaker converts data appropriately between the XDB data type and the Powersoft data type.

Basic software components

The following diagram shows the basic software components you need to access an XDB database using the XDB database interface. The diagram also indicates who supplies each DLL and where it is installed.



Preparing to use the database

Before you define the interface and connect to an XDB database from PowerBuilder or InfoMaker, follow these steps to prepare the database.

❖ To prepare an XDB database:

1 Install the XDB database software on the database server or on your computer, following the instructions in your XDB documentation.

You must obtain the database software from XDB Systems, Inc.

2 Install the Powersoft XDB database interface (PBXDB040.DLL) on your computer.

For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide.*

3 Make sure the directory containing your XDB files appears in your program search path.

4 If you plan to run XDB DOS programs, make sure the following lines appear in your AUTOEXEC.BAT file:

```
set xdbcfg=XDB_directory
set xrsrvcfg=XDB_directory
```

For example:

```
set xdbcfg=c:\xdb
set xrsrvcfg=c:\xdb
```

5 Make sure the [XDB] section of your WIN.INI file contains the following lines:

```
[XDB]
xdbcfg=XDB_directory
xrsrvcfg=XDB_directory
```

For example:

```
[XDB]
xdbcfg=c:\xdb
xrsrvcfg=c:\xdb
```

6 If you are running XDB on a remote DOS or OS/2 database server in your network, make sure the configuration settings are correct for a network connection.

For help, see your system administrator.

7 If necessary, run the DB2SYSPB.SQL script outside PowerBuilder or InfoMaker to create the PowerBuilder system tables on the database gateway.

For instructions, see "Creating Powersoft system tables in DB2 databases" on page 247.

Defining the database interface

The following table lists the values you should supply for each field in the Database Profile Setup dialog box when defining the Powersoft XDB database interface.

Field	Value
Profile Name	The name of your database profile.
DBMS	**XDB**
User ID	Public
Password	Not applicable for use with PowerBuilder or InfoMaker.
Database Name	The full pathname of the XDB database you want to access.
Prompt for Database information during Connect	Select this checkbox if you want to be prompted for connection information when creating or selecting a profile to connect to the database.
Server Name	Not applicable for use with PowerBuilder or InfoMaker.
Login ID	The login ID required to connect to your database. In order to properly create the Powersoft repository tables, make sure the first person to connect to the database has sufficient authority to create tables and grant permissions to public.

Field	Value
Login Password	The login password required to connect to your database. The actual password does not display in this field. Small Xs appear in place of the characters you type.
DBParm	Specify DBMS-specific connection parameters in this field. For more about DBParm values that you can specify for XDB, see Chapter 5, "Setting Additional Connection Parameters."

What to do next

For instructions on connecting to the database, see Chapter 4, "Managing Database Connections."

Creating Powersoft system tables in DB2 databases

This section describes how PowerBuilder and InfoMaker create system tables in your DB2 database to store extended attribute information. It then explains why you may want to use the DB2SYSPB.SQL script to create these system tables outside PowerBuilder or InfoMaker.

You can use the DB2SYSPB.SQL script if you are connecting to a DB2 or DB2-compatible database through any of the following Powersoft database interfaces:

- IBM DRDA databases, including:
  - Database Manager
  - DB2/2
  - DB2/MVS
  - DB2/6000
- Micro Decisionware Database Gateway Interface for DB2
- Sybase Net-Gateway Interface for DB2
- XDB

Creating the repository

When you use the Database painter in PowerBuilder or InfoMaker to create or modify a table, the information you provide is stored in five Powersoft system tables in your database. These system tables, known collectively as the **repository**, contain extended attribute information such as the text to use for labels and column headings, validation rules, display formats, and edit styles. (The Powersoft system tables are different from the system tables provided by your DB2 database.)

By default, the Powersoft repository is created automatically the first time a user connects to the database using PowerBuilder or InfoMaker.

For more about the Powersoft repository and the tables it contains, see "Creating the Powersoft repository" in Chapter 4, "Managing Database Connections."

❖ **To ensure that the Powersoft repository is created with the proper access rights:**

- Make sure the first person to connect to the database with PowerBuilder or InfoMaker has sufficient authority to create tables and grant permissions to public.

  This means that the first person to connect to the database should log on as the database owner, database administrator, system user, or system administrator, as specified by the DBMS.

Using the DB2SYSPB.SQL script

If you are a system administrator at a DB2 site, you may prefer to create the Powersoft system tables outside PowerBuilder or InfoMaker for the following reasons:

- The first user to connect to the DB2 database using PowerBuilder or InfoMaker may not have the proper authority to create tables.
- When PowerBuilder or InfoMaker creates the system tables, it places them in the default table space. This may not be appropriate for your needs.

To create the system tables yourself, you can run the DB2SYSPB.SQL script outside PowerBuilder or InfoMaker. This script contains SQL commands that create and initialize the Powersoft system tables with the table owner and table space you specify.

❖ **To use the DB2SYSPB.SQL script to create Powersoft system tables:**

1. Log on to the database server or gateway as the system administrator.
2. Insert the disk containing your Powersoft database interface into drive A.

   If you are using PowerBuilder, the Powersoft database interfaces are on Disk 5 of the Deployment Kit. If you are using InfoMaker, they are on Disk 7.
3. Copy the file DB2SYSPB.SQL from the disk in drive A to the database server or gateway machine.

   The file DB2SYSPB.SQL is *not* installed by default when you install the Powersoft database interfaces.

4 Use any text editor to modify the DB2SYSPB.SQL script for your environment, if necessary. You can do any of the following:

- Change all instances of pbcatown to another name. (You can also leave the table owner as pbcatown, which is the default.)

Specifying SYSIBM is prohibited
You cannot specify SYSIBM as the table owner. This is prohibited by DB2.

- Change all instances of database.tablespace to the appropriate value.
- Add appropriate statement delimiters for the tool you are using to run the script.
- Remove comments and blank lines if necessary.

PBCatalogOwner
If you changed pbcatown to another name in the DB2SYSPB.SQL script, you must specify the new owner name as the value for the PBCatalogOwner DBParm parameter in your database profile.

For instructions, see the description of PBCatalogOwner in Chapter 5, "Setting Additional Connection Parameters."

5 Save any changes you made to the DB2SYSPB.SQL script.

6 Execute the DB2SYSPB.SQL script from the database server or gateway by using the SQL tool of your choice.

Installing Powersoft stored procedures in SQL Server databases

This section describes how to use the PBSYB.SQL, PBSYBRT.SQL, and PBSYC.SQL scripts to install Powersoft stored procedures in a SQL Server database.

You *must* run these scripts outside PowerBuilder or InfoMaker *before* connecting to a SQL Server database for the first time through either of the following Powersoft database interfaces:

- SQL Server (PBSYB040.DLL)
- Sybase SQL Server System 10 (PBSYC040.DLL)

What are the Powersoft stored procedure scripts?

In order to use PowerBuilder or InfoMaker in a SQL Server environment, you or your system administrator must install certain Powersoft stored procedures in the database *before* you connect to SQL Server from these products. PowerBuilder and InfoMaker use the Powersoft stored procedures to get information about tables and columns from the SQL Server system catalog. (The Powersoft stored procedures are different from the stored procedures you may create in your database.)

You may also need to install a subset of the Powersoft stored procedures on those computers where you plan to deploy PowerBuilder or InfoMaker applications.

A **stored procedure** is a group of pre compiled and pre optimized SQL statements that performs some database operation. Stored procedures reside on the database server where they can be accessed as needed.

Powersoft provides three SQL script files to install the required stored procedures on the SQL Server database server and deployment machines:

- PBSYB.SQL
- PBSYBRT.SQL
- PBSYC.SQL

Where to find the Powersoft stored procedure scripts
If you are using PowerBuilder, the Powersoft stored procedure scripts are on Disk 5 of the Deployment Kit. If you are using InfoMaker, they are on Disk 7.

The Powersoft stored procedure scripts are *not* installed by default when you install the Powersoft database interfaces.

PBSYB.SQL script

The PBSYB.SQL script contains SQL code that drops all existing Powersoft stored procedures in the SQL Server master database and then recreates them.

When to run

Before you connect to a SQL Server database in PowerBuilder or InfoMaker, you must run the PBSYB.SQL script once per database server.

For instructions, see "How to run the scripts" on page 254.

Stored procedures it creates

The following table briefly describes the Powersoft stored procedures created by the PBSYB.SQL script in the SQL Server master database. The procedures are listed in the order in which the script creates them.

Stored procedure	What it retrieves
sp_pbcolumn	Column information from the catalog
sp_pbdb	A database list from the catalog
sp_pbindex	Index information from the catalog
sp_pbproc	Stored procedure information from the catalog
sp_pbhelprotect	Security information from the catalog
sp_pbtable	Table information from the catalog
sp_pbtext	Comment information from the catalog
sp_pbprimarykey	Primary key information from the catalog
sp_pbforeignkey	Foreign key information from the catalog
sp_pbfktable	Table information from foreign keys referencing the current table

PBSYBRT.SQL script

The PBSYBRT.SQL script is a subset of the PBSYB.SQL script. It contains SQL code that drops a subset of the existing Powersoft stored procedures in the SQL Server master database and then recreates them.

When to run

Before you deploy a PowerBuilder or InfoMaker application, you must run the PBSYBRT.SQL script instead of the PBSYB.SQL script on those servers where both of the following are true:

- You plan to deploy PowerBuilder or InfoMaker applications that use SQL Server.
- No PowerBuilder or InfoMaker development takes place.

For instructions, see "How to run the scripts" on page 254.

Exception

You do *not* need to run PBSYBRT.SQL on those computers where PBSYB.SQL has already been run. In fact, if you have no disk space limitations, you can simply run PBSYB.SQL on the database server and on all servers at your site that will deploy PowerBuilder or InfoMaker applications.

The PBSYB.SQL script installs the Powersoft stored procedures required for SQL Server application development and deployment.

Stored procedures it creates

The following table briefly describes the Powersoft stored procedures created by the PBSYBRT.SQL script in the SQL Server master database. The procedures are listed in the order in which the script creates them.

Stored procedure	What it retrieves
sp_pbindex	Index information from the catalog
sp_pbtable	Table information from the catalog
sp_pbprimarykey	Primary key information from the catalog

PBSYC.SQL script

The PBSYC.SQL script contains SQL code that drops all existing Powersoft stored procedures in the Sybase SQL Server System 10 sybsystemprocs database and then recreates them.

The PBSYC.SQL script uses the sybsystemprocs database to hold the Powersoft stored procedures. This database is created when you install Sybase System 10.

When to run

Before you connect to a Sybase System 10 database in PowerBuilder or InfoMaker, you must run the PBSYC.SQL script once per database server.

For instructions, see "How to run the scripts" on page 254.

Stored procedures it creates

The following table briefly describes the Powersoft stored procedures created by the PBSYC.SQL script in the sybsystemprocs database. The procedures are listed in the order in which the script creates them.

Stored procedure	What it does
sp_pb40column	Lists columns in a table
sp_pb40primarykey	Lists columns in the primary key for the current table
sp_pb40pkcheck	Determines whether the table has a primary key
sp_pb40fktable	Lists the tables that reference the current table
sp_pb40foreignkey	Lists all foreign keys associated with the current table
sp_pb40extcat	Checks the status of the PowerBuilder catalog
sp_pb40proc	Lists available stored procedures
sp_pb40text	Retrieves the text of a stored procedure from the syscomments table
sp_pb40db	Retrieves the names of all databases available for this server
sp_pb40helprotect	Retrieves security information
sp_pb40table	Retrieves information about all tables in a database or about a specified table
sp_pb40index	Retrieves information about all indexes for a specified table

When to run the scripts

Before you connect to SQL Server for the first time from PowerBuilder or InfoMaker, you or your system administrator must run the Powersoft stored procedure scripts to install the necessary stored procedures in the database.

The following table lists which scripts you need to run when connecting to SQL Server or Sybase SQL Server System 10.

Powersoft database interface	SQL script	When to run
SQL Server (PBSYB040.DLL)	PBSYB.SQL	Once per database server into the master database
	PBSYBRT.SQL	Once per deployment machine into the master database
Sybase SQL Server System 10 (PBSYC040.DLL)	PBSYC.SQL	Once per database server into the sybsystemprocs database

How to run the scripts

There are various ways to run the Powersoft stored procedure scripts, depending on the SQL tools available at your site. The following table lists some of the tools that you can use to run the scripts outside PowerBuilder or InfoMaker.

You can use this tool	To run these scripts
WISQL	PBSYB.SQL PBSYBRT.SQL PBSYC.SQL
ISQL	PBSYB.SQL PBSYBRT.SQL PBSYC.SQL
PBSYB.BAT with ISQL	PBSYB.SQL

The following sections describe how to run the Powersoft stored procedure scripts using each of these tools.

Using WISQL to run the stored procedure scripts

WISQL is an interactive SQL utility that comes with the Sybase Open Client software and runs on Windows operating systems. If you have WISQL installed, use the following procedure to run the Powersoft stored procedure scripts.

For instructions on using WISQL, see your Sybase Open Client documentation.

❖ To use WISQL to run the Powersoft stored procedure scripts:

1 Start the WISQL utility from Windows.

2 Open a connection to the SQL Server database as the system administrator. The database to which you connect depends on the script you are running, as follows:

To run this script	Connect to this SQL Server database
PBSYB.SQL	master
PBSYBRT.SQL	master
PBSYC.SQL	sybsystemprocs

3 Insert the disk containing the Powersoft stored procedure scripts into drive A. The files you need are:

- PBSYB.SQL
- PBSYBRT.SQL
- PBSYC.SQL

If you are using PowerBuilder, the Powersoft stored procedure scripts are on Disk 5 of the Deployment Kit. If you are using InfoMaker, they are on Disk 7.

4 Open the file containing the SQL script you want to run.

5 Delete the **use master** or **use sybsystemprocs** command and the **go** command that appear at the beginning of each script.

WISQL requires that you issue the **use master** or **use sybsystemprocs** command by itself, with no other SQL commands following it. When you open a connection to the master or sybsystemprocs database in step 2 you are, in effect, issuing the **use master** or **use sybsystemprocs** command. This command should not be issued again as part of the stored procedure script.

Therefore, to successfully install the stored procedures, you *must* delete the lines shown in the following table from the beginning of the Powersoft stored procedure script *before* executing the script.

Before executing this script	Delete these lines
PBSYB.SQL	`use master` `go`
PBSYBRT.SQL	`use master` `go`
PBSYC.SQL	`use sybsystemprocs` `go`

6 Execute all of the statements in the SQL script.

7 Exit the WISQL session.

The first time you run a Powersoft stored procedure script, there are no Powersoft stored procedures to drop in the master or sybsystemprocs database. Therefore, the SQL DROP statements in the script cause error messages to display. You can disregard these messages.

Using ISQL to run the stored procedure scripts

ISQL is an interactive SQL utility that comes with SQL Server and runs on MS-DOS operating systems. If you have ISQL installed, use the following procedure to run the Powersoft stored procedure scripts.

For instructions on using ISQL, see your SQL Server documentation.

❖ To use ISQL to run the Powersoft stored procedure scripts:

1 Connect to the SQL Server database as the system administrator. The database to which you connect depends on the script you are running, as follows:

To run this script	Connect to this SQL Server database
PBSYB.SQL	master
PBSYBRT.SQL	master
PBSYC.SQL	sybsystemprocs

2 Insert the disk containing the Powersoft stored procedure scripts into drive A. The files you need are:

- PBSYB.SQL
- PBSYBRT.SQL
- PBSYC.SQL

If you are using PowerBuilder, the Powersoft stored procedure scripts are on Disk 5 of the Deployment Kit. If you are using InfoMaker, they are on Disk 7.

3 From a DOS prompt, issue the appropriate ISQL command to run the SQL script with the user ID, servername, and, optionally, password you specify. (Specify uppercase and lowercase exactly as shown.)

isql /U sa /S *SERVERNAME* **/i** *pathname* **/P** {*password*}

Parameter	Description
sa	The user ID for the system administrator. Do *not* change this user ID.
SERVERNAME	The name of the computer running the SQL Server database.
pathname	The drive and directory containing the SQL script you want to run.
password	(Optional) The password for the sa (system administrator) user ID. The default SQL Server installation creates the sa user ID without a password. If you changed the password for sa during the installation, replace *password* with your new password.

For example:

```
isql /U sa /S TESTDB /i a:\pbsyb.sql /P
isql /U sa /S SALES /i c:\pbsybrt.sql /P adminpwd
isql /U sa /S EMPLOYEE /i a:\pbsyc.sql /P
```

The first time you run a Powersoft stored procedure script, there are no Powersoft stored procedures to drop in the master or sybsystemprocs database. Therefore, the SQL DROP statements in the script cause error messages to display. You can disregard these messages.

Using the PBSYB.BAT file to run the PBSYB.SQL script

Powersoft provides a batch file named PBSYB.BAT as a convenient way to run the PBSYB.SQL script using the ISQL utility. PBSYB.BAT contains an ISQL command that runs PBSYB.SQL with the user ID, servername, and, optionally, password you specify.

If you have ISQL installed, use the following procedure to run the PBSYB.SQL script with the PBSYB.BAT file.

❖ To use the PBSYB.BAT file to run the PBSYB.SQL script:

1. Connect to the master SQL Server database as the system administrator.
2. Insert the disk containing the Powersoft stored procedure scripts into drive A. The files you need are:
   - PBSYB.BAT
   - PBSYB.SQL

   If you are using PowerBuilder, the Powersoft stored procedure scripts are on Disk 5 of the Deployment Kit. If you are using InfoMaker, they are on Disk 7.
3. Open the PBSYB.BAT file with any text editor.

4 Edit the ISQL command line in the PBSYB.BAT file as follows. Keep the command on a single line, and specify uppercase and lowercase exactly as shown.

isql /U sa /S *SERVERNAME* **/i** *PBSYB.SQL_pathname* **/P** {*password*}

Parameter	Description
sa	The user ID for the system administrator. Do *not* change this user ID.
SERVERNAME	The name of the computer running the SQL Server database.
PBSYB.SQL_pathname	The drive and directory containing the PBSYB.SQL file.
password	(Optional) The password for the sa (system administrator) user ID. The default SQL Server installation creates the sa user ID without a password. If you changed the password for sa during the installation, replace *password* with your new password.

Here are two examples of a properly edited PBSYB.BAT file:

```
isql /U sa /S TESTDB /i a:\pbsyb.sql /P
isql /U sa /S SALES /i c:\pbsyb.sql /P adminpwd
```

5 Save any changes you made to the PBSYB.BAT file.

6 From the directory where the PBSYB.BAT file resides, type the following at a DOS prompt to run the PBSYB.SQL script:

pbsyb

The first time you run the PBSYB.SQL script, there are no Powersoft stored procedures to drop in the master database. Therefore, the SQL DROP statements in the script cause error messages to display. You can disregard these messages.

CHAPTER 4

Managing Database Connections

About this chapter

After you prepare the database and define an ODBC data source or Powersoft database interface, you can connect to the database from PowerBuilder or InfoMaker. Connecting to a database means that you can work with the tables and views stored in that database.

This chapter describes how to connect to a database in PowerBuilder or InfoMaker, maintain ODBC data source definitions and database profiles, and share database profiles.

> **Terminology**
> In this chapter, the term **database** refers to *both* of the following, unless otherwise specified:
>
> - An ODBC data source that you access with the Powersoft ODBC interface and the appropriate ODBC driver
> - A database or DBMS that you access with the appropriate Powersoft database interface

Contents

Topic	Page
About database connections	263
Creating the Powersoft repository	266
Connecting to a database	271
Maintaining ODBC data source definitions	278
Maintaining database profiles	286
Sharing database profiles	294

Before you begin

Before using the procedures in this chapter to connect to your data, make sure you have done both of the following:

- Installed the ODBC driver or Powersoft database interface required to access the data.

  For instructions, see the PowerBuilder *Installation and Deployment Guide* or the InfoMaker *Installation Guide*.

  For more about using an ODBC driver supplied by a vendor other than Powersoft, see "About the Powersoft ODBC interface" in Chapter 2, "Using ODBC Data Sources."

- Prepared the database and defined the ODBC data source or Powersoft database interface.

  For instructions, see Chapter 2, "Using ODBC Data Sources," or Chapter 3, "Using Powersoft Database Interfaces."

About database connections

This section gives an overview of when database connections occur in PowerBuilder and InfoMaker. It also tells why you should use database profiles to manage your database connections.

When database connections occur

Database connections occur at different times in PowerBuilder and InfoMaker, depending on the product you are using and whether you are developing or executing an application.

The following table summarizes the actions you can take that cause PowerBuilder or InfoMaker to connect to your database. A checkmark indicates that a particular action causes a database connection in PowerBuilder or InfoMaker.

When you	PowerBuilder connects to your database	InfoMaker connects to your database
Start the product		✓
Open a painter that accesses the database	✓	✓
Compile or save a PowerBuilder script containing embedded SQL statements	✓	
Execute a PowerBuilder application that accesses the database	✓	

Connecting at startup or from a painter

PowerBuilder or InfoMaker connects to the database you used last when you do either of the following:

- Start InfoMaker
- Open a PowerBuilder or InfoMaker painter that accesses your database

PowerBuilder or InfoMaker determines which database you used last by reading the values in the [Database] section of the PB.INI or IM.INI file. When you connect to a database, PowerBuilder or InfoMaker copies the connection parameters you specified to the [Database] section of the PB.INI or IM.INI file, overwriting the current values in the [Database] section.

For instructions on using a particular painter in PowerBuilder or InfoMaker, see the *User's Guide*.

Connecting during application execution

In PowerBuilder, a database connection can also occur when you:

- Compile or save a script containing embedded SQL statements while developing an application
- Execute a SQL CONNECT statement in a script while executing an application
- Invoke a DataWindow control function that accesses the database while executing an application

This manual describes how to connect to your database when you are using PowerBuilder or InfoMaker to develop an application.

For instructions on connecting to a database from a PowerBuilder application, see *Building Applications*.

Using database profiles

A **database profile** is a named set of parameters that specifies a connection to a particular data source or database in PowerBuilder or InfoMaker. Defining and using database profiles is the easiest way to manage your database connections in PowerBuilder or InfoMaker because you can:

- Select a database profile to establish or change a database connection. You can easily connect to another database anytime during a PowerBuilder or InfoMaker session.
- Edit a database profile to modify or supply additional connection parameters.
- Delete a database profile if you no longer need to access that data.

Database profiles are created as part of the process of defining ODBC data sources and Powersoft database interfaces. When you define an ODBC data source in PowerBuilder or InfoMaker, as described in Chapter 2, a database profile is automatically created for the data source. When you define a Powersoft database interface, as described in Chapter 3, you create the database profile yourself.

Because database profiles are created when you define your data and are stored in the PB.INI or IM.INI file, they have the following benefits:

- They are always available to you.
- Connection parameters supplied in a database profile are saved until you edit or delete the database profile.

Creating the Powersoft repository

By default, the first time you use PowerBuilder or InfoMaker to connect to a database, PowerBuilder or InfoMaker creates five Powersoft system tables. These five tables, known collectively as the Powersoft **repository**, contain default extended attribute information about tables and columns in your database. You can also define extended attributes when you create or modify a table in the Database painter.

This section tells you how to:

- Make sure the Powersoft repository tables are created with the proper access rights
- Display and open a Powersoft repository table, if you want
- Understand the kind of information stored in the Powersoft repository
- Control catalog access by specifying that PowerBuilder or InfoMaker *not* create the Powersoft repository

Logging on to your database for the first time

By default, PowerBuilder or InfoMaker creates the Powersoft repository the first time you connect to a database with PowerBuilder or InfoMaker.

To ensure that PowerBuilder or InfoMaker creates the repository tables with the proper access rights to make them available to all users, the first person to connect to the database with PowerBuilder or InfoMaker must log on with the proper authority.

❖ To ensure proper creation of the Powersoft repository:

- Make sure the first person to connect to the database with PowerBuilder or InfoMaker has sufficient authority to create tables and grant permissions to public.

  This means that the first person to connect to the database should log on as the database owner, database administrator, system user, or system administrator, as specified by the DBMS.

Displaying the repository

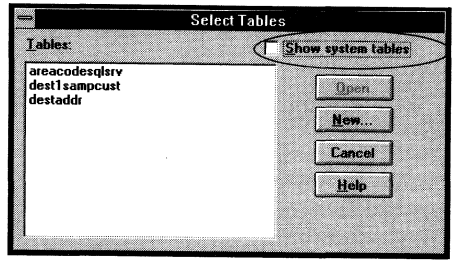
PowerBuilder or InfoMaker maintains the Powersoft repository automatically whenever you change the information for a table or column in the Database painter. The system tables in the repository are different from the system tables provided by your DBMS.

If you want, you can display and open Powersoft repository tables in the Database painter just like other tables.

❖ To display the Powersoft repository tables:

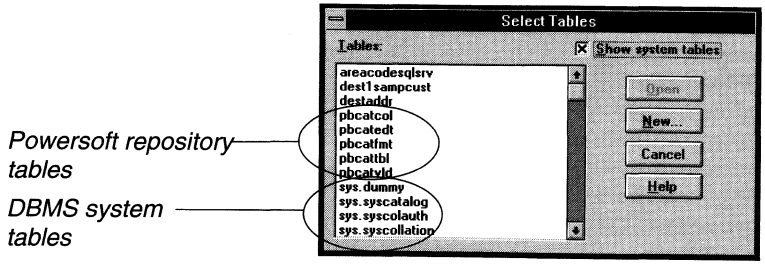
1 Open the Select Tables dialog box in the Database painter.

For instructions, see the PowerBuilder or InfoMaker *User's Guide*.



2 Select the Show System Tables checkbox.

The Powersoft repository and DBMS system tables appear in the Tables list.



3 Open a Powersoft repository table to display its contents.

For instructions, see the PowerBuilder or InfoMaker *User's Guide*.

Contents of the repository

PowerBuilder or InfoMaker stores five types of extended attribute information in the Powersoft repository, as described in the following table. (For more about the Powersoft repository, see the PowerBuilder *User's Guide* or online Help.)

> **Prefixes in repository table names**
> For some databases, PowerBuilder or InfoMaker precedes the name of the repository table with a DBMS-specific prefix by default. For example, the names of Powersoft repository tables have the prefix DBO in a SQL Server database (such as DBO.pbcatcol), SYSTEM in an ORACLE database (such as SYSTEM.pbcatfmt), and PBCATOWN in an IBM DRDA database (such as PBCATOWN.pbcattbl).
>
> The following table lists the base name of each repository table without the DBMS-specific prefix.

This Powersoft repository table	Stores information about	That includes
pbcatcol	Columns	Names, comments, headers, labels, case, initial value, and justification
pbcatedt	Edit styles	Edit style names and definitions
pbcatfmt	Display formats	Display format names and definitions
pbcattbl	Tables	Name, owner, default fonts (for data, headings and labels), and comments
pbcatvld	Validation rules	Validation rule names and definitions

Controlling catalog access

To control access to the catalog (repository) at your site, you can specify that PowerBuilder or InfoMaker not create the repository, that the existing repository is read-only, or that the repository is accessible only to certain users or groups.

You can control catalog access by doing any of the following:

- **Setting the NoCatalog preference** Set the NoCatalog database preference in the PowerBuilder Preferences painter or in the PBODB040.INI file. (NoCatalog is called PBNoCatalog in the PBODB040.INI file.)
- **Setting the ReadOnly preference** Set the ReadOnly database preference in the PowerBuilder Preferences painter or in the PBODB040.INI file.
- **Granting permissions on repository tables** Grant explicit permissions on the repository tables to users or groups at your site.

Setting NoCatalog and ReadOnly to control access

❖ To control catalog access by setting NoCatalog or ReadOnly:

- Set the NoCatalog or ReadOnly database preference to 1 or 'Yes' in the PowerBuilder Preferences painter or the PBODB040.INI file. (The default settings for these preferences are 0 or 'No'.)

Setting this variable to 1 or 'Yes'	Has this effect
NoCatalog	If the Powersoft repository tables *exist*, PowerBuilder or InfoMaker does not use them when you create a new DataWindow object, report, or form. If the Powersoft repository tables *do not exist*, PowerBuilder or InfoMaker does not create them. Instead, the DataWindow, Report, and Form painters use the appropriate default values for extended attributes.
ReadOnly	If the Powersoft repository tables *exist*, PowerBuilder or InfoMaker uses them when you create a new DataWindow object, report, or form, but does not update them. If the Powersoft repository tables *do not exist*, PowerBuilder or InfoMaker does not create them. Instead, the DataWindow, Report, and Form painters use the appropriate default values for extended attributes.

For more about the NoCatalog and ReadOnly database preferences, see their descriptions in Chapter 5, "Setting Additional Connection Parameters."

Granting permissions on the repository tables to control access

If your DBMS supports SQL GRANT and REVOKE statements, you can control access to the Powersoft repository tables. The default authorization for each repository table is:

GRANT SELECT, UPDATE, INSERT, DELETE ON *table* TO PUBLIC

After you create the repository, you could, for example, control access by granting SELECT authority to end users and SELECT, UPDATE, INSERT, and DELETE authority to developers.

This technique offers security and flexibility that is enforced by the DBMS itself.

Connecting to a database

There are two ways to establish or change a database connection in PowerBuilder or InfoMaker:

- Selecting a database profile in the Database Profiles dialog box or from the File➤Connect menu
- Responding to prompts for connection parameters

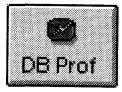
Using database profiles is the easiest way to connect to a database in PowerBuilder or InfoMaker, especially if you often switch connections between different databases.

Selecting a database profile

You can connect to a database by selecting its database profile from:

- The Database Profiles dialog box
- The File➤Connect menu

❖ To connect to a database by using the Database Profiles dialog box:



1 Click the Database Profile button in the PowerBar.
or
In any of the painters listed in the following table, select File➤Connect➤Setup from the menu bar.

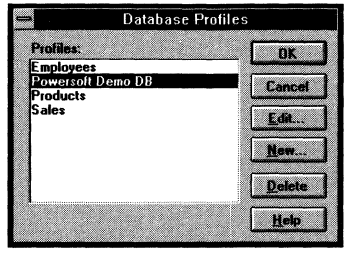
Product	Painters
PowerBuilder	Database painter DataWindow painter Report painter
InfoMaker	Database painter Form painter Report painter

Database Profile button
If your PowerBar does not include the Database Profile button, use the customize feature to add the button to the PowerBar.

Having the Database Profile button on your PowerBar is useful if you frequently switch connections between different databases.

For instructions on customizing toolbars, see the PowerBuilder or InfoMaker *User's Guide*.

The Database Profiles dialog box appears, listing the names of defined profiles. If you are currently connected to a database, the name of its profile is highlighted.



2 Select the name of the database profile to which you want to connect.

3 Click OK.

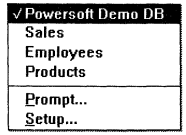
PowerBuilder or InfoMaker connects to the selected database and returns you to the painter workspace.

❖ To connect to a database by using the File➤Connect menu:

1 In any of the painters listed in the following table, select File➤Connect from the menu bar.

Product	Painters
PowerBuilder	Database painter DataWindow painter Report painter
InfoMaker	Database painter Form painter Report painter

A cascading menu appears, listing the names of defined database profiles. If you are currently connected to a database, the name of its profile is checked.



2 Select the name of the database profile to which you want to connect.

PowerBuilder or InfoMaker connects to the selected database and returns you to the painter workspace.

Responding to prompts

You can also connect to a database in PowerBuilder or InfoMaker by responding to dialog boxes that prompt you to supply connection information.

When you connect to a database by responding to prompts, PowerBuilder or InfoMaker writes the connection parameters you specify to the [Database] section of the PB.INI or IM.INI file.

Unlike database profiles, however, connection parameters supplied in response to prompts are *not* permanently stored in the INI file. PowerBuilder or InfoMaker overwrites the existing values in the [Database] section with new values every time you connect to a different database.

Since connection parameters supplied in response to prompts are not permanently stored in the INI file, you may want to use prompts only when connecting to databases you use infrequently.

Connecting to an ODBC data source through prompts
You must define an ODBC data source before you can connect to it by responding to prompts.

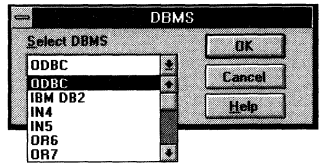
For instructions on defining an ODBC data source, see the section for your data source driver in Chapter 2, "Using ODBC Data Sources."

❖ To connect to a database by responding to prompts:

1 In any of the painters listed in the following table, select File➤Connect➤Prompt from the menu bar.

Product	Painters
PowerBuilder	Database painter DataWindow painter Report painter
InfoMaker	Database painter Form painter Report painter

The DBMS dialog box appears. The Select DBMS dropdown listbox contains the ODBC identifier and identifiers for any other Powersoft database interfaces you have installed. If you are currently connected to a database, its identifier is highlighted.



Where the DBMS identifiers come from

When you install a Powersoft database interface, PowerBuilder or InfoMaker updates the Vendors list in the [Database] section of the PB.INI or IM.INI file with the proper DBMS identifier for your interface. (The ODBC identifier appears in the Vendors list by default.)

The DBMS identifiers that appear in the DBMS dialog box and the Database Profile Setup dialog box are the ones in your Vendors list.

If you have installed only ODBC drivers and have *not* installed any Powersoft database interfaces, the DBMS dialog box does not appear when you select File➤Connect➤Prompt. The first dialog box you see is the SQL Data Sources dialog box, as shown in the example for an ODBC data source on page 275.

2 Select the identifier of the DBMS to which you want to connect.

3 Click OK.

One or more dialog boxes appear to prompt you for information required to connect to the database. This information can include:

- Data source name (for ODBC data sources)
- User ID or login ID
- Password or login password
- Server name
- Database name

For examples of the kind of dialog boxes that can appear when connecting to an ODBC data source or Powersoft database interface, see the Example sections next.

4 In the dialog boxes that appear, supply the connection parameters required to connect to your database.

PowerBuilder or InfoMaker connects to the database with the parameters you specified.

For instructions on supplying connection parameters, see the section for your data source driver in Chapter 2, "Using ODBC Data Sources," or the section for your database interface in Chapter 3, "Using Powersoft Database Interfaces."

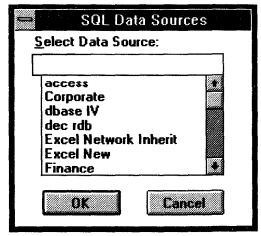
Example for an ODBC data source

If you select ODBC in the DBMS dialog box, the SQL Data Sources dialog box appears, listing all ODBC data sources you have defined.

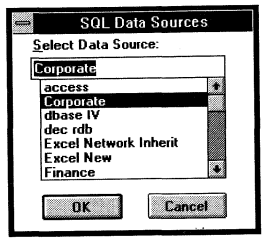
PowerBuilder Desktop

If you are using PowerBuilder Desktop, the SQL Data Sources dialog box lists only the names of *supported* ODBC data sources.

For a list of the ODBC data sources supported in PowerBuilder Desktop, see Appendix A, "Supported Data Sources and Databases."

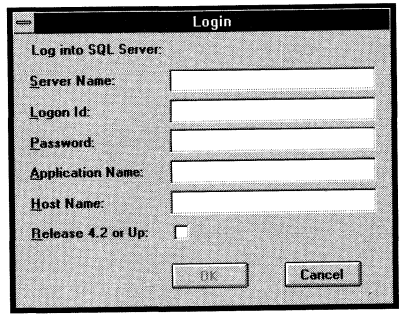


To connect to an ODBC data source, select its name and click OK.

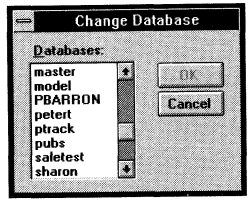


Example for a Powersoft database interface

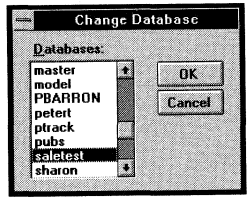
If you select the identifier for the Powersoft SQL Server interface (SYBASE) in the DBMS dialog box, the following Login dialog box appears.



After you supply the necessary connection parameters in the Login dialog box and click OK, the Change Database dialog box appears. The Change Database dialog box lists all available databases for your DBMS.



To connect to a database, select its name and click OK.



What happens when you connect

This section describes what happens when you connect to a database in PowerBuilder or InfoMaker.

Connection parameters

When you connect to a database by selecting its database profile or by responding to prompts, PowerBuilder or InfoMaker writes the connection parameters you specify to the [Database] section of the PB.INI or IM.INI file.

Each time you connect to a different database, PowerBuilder or InfoMaker overwrites the existing parameters in the [Database] section with the parameters for the new database connection.

What you are connected to

When you start InfoMaker or open a PowerBuilder or InfoMaker painter that accesses the database, you are connected to the database you used last. PowerBuilder or InfoMaker determines which database this is by reading the [Database] section of the PB.INI or IM.INI file.

For example, the following portions of the [Database] section show a connection to the Powersoft Demo database, which is a Watcom SQL ODBC data source.

```
[Database]
Vendors=ODBC,IBM DB2,IN5,OR7,SYBASE,GUPTA,MDI
DBMS=ODBC
LogId=
LogPassword=
ServerName=
Database=Powersoft Demo DB
UserId=
DatabasePassword=
TableDir=1
StayConnected=1
AutoCommit=0
.
.
.
DbParm=Connectstring='DSN=Powersoft Demo DB'
.
.
.
```

Maintaining ODBC data source definitions

You can easily edit or delete an existing ODBC data source definition in PowerBuilder or InfoMaker.

Editing an ODBC data source definition

You can edit an ODBC data source definition to change one or more of its connection parameters.

You cannot edit the ODBC data source you are connected to
You *cannot* edit the definition of the ODBC data source to which you are currently connected in PowerBuilder or InfoMaker.

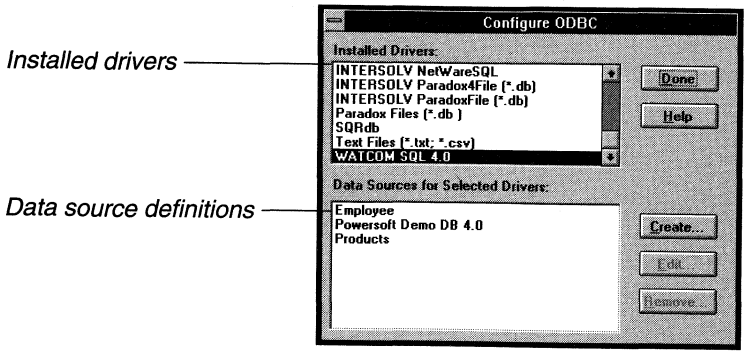
❖ To edit an ODBC data source definition:

1 Click the Configure ODBC button in the PowerBar.
or
In the PowerBuilder or InfoMaker Database painter, select File➤Configure ODBC from the menu bar.

Configure ODBC button
If your PowerBar does not include the Configure ODBC button, use the customize feature to add the button to the PowerBar.

For instructions on customizing toolbars, see the PowerBuilder or InfoMaker *User's Guide*.

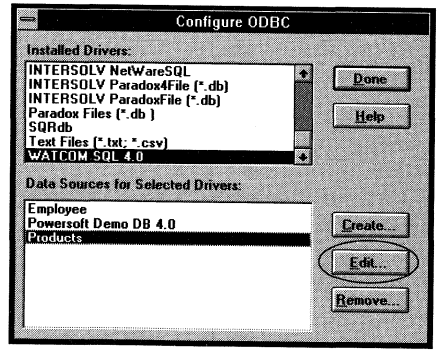
The Configure ODBC dialog box appears, listing the ODBC drivers installed on your computer and the data sources defined for each driver.



2 Select the ODBC driver that accesses the data source you want to edit.

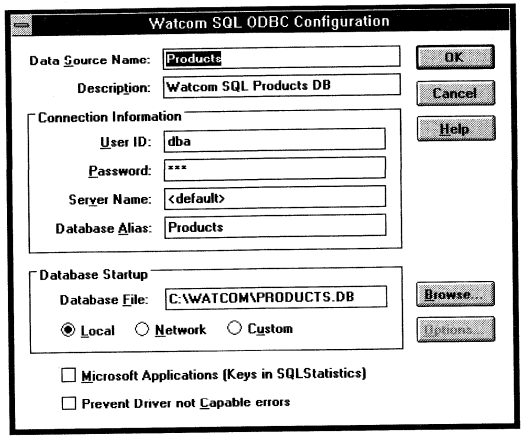
The names of the data source definitions for that driver appear in the Data Sources for Selected Drivers list.

3 Select the name of the data source definition you want to edit.



4 Click the Edit button.

The ODBC setup dialog box for the selected data source appears.

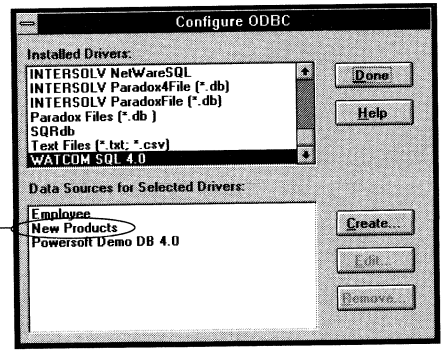


5 Edit the data source definition as required for your connection.

For instructions on completing the ODBC setup dialog box, see the section for your data source driver in Chapter 2, "Using ODBC Data Sources."

6 Click OK.

The Configure ODBC dialog box appears. If you changed the name of this data source, the new name appears in the data source list.



Edited data source definition

7 Repeat steps 2 through 6 if you want to edit another ODBC data source definition.

8 Click Done when you are finished editing ODBC data source definitions.

The Configure ODBC dialog box closes and you are returned to the painter workspace.

9 If you changed the data source name in step 5, edit the database profile for this data source to make sure it contains the correct name in the connect string.

For instructions, see "Changing the ODBC data source name" next.

What happens

When you edit an ODBC data source definition, PowerBuilder or InfoMaker updates the ODBC.INI file with the new values for this data source.

However, PowerBuilder or InfoMaker does *not* update the database profile for this data source in the PB.INI or IM.INI file.

Changing the ODBC data source name

As described in the preceding section, PowerBuilder or InfoMaker does not update the database profile when you edit an ODBC data source. This means that if you change the name of an ODBC data source for which PowerBuilder or InfoMaker has already created a database profile, the connect string in the existing database profile still contains the old data source name. Since the existing database profile does not have the correct data source name, it will no longer connect to the data source.

To update the database profile so it connects to the data source, you *must* edit the profile to supply the correct data source name, as described in the following procedure.

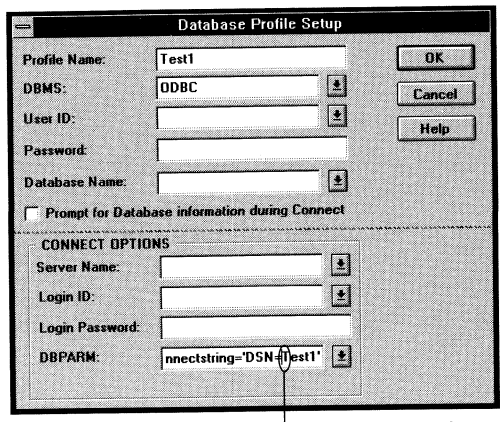
❖ To update an existing database profile with the correct ODBC data source name:

1 Open the Database Profile Setup dialog box for the edited ODBC data source.

For instructions, see "Editing a database profile" on page 286.

2 Click the More button to display the DBParm dropdown listbox.

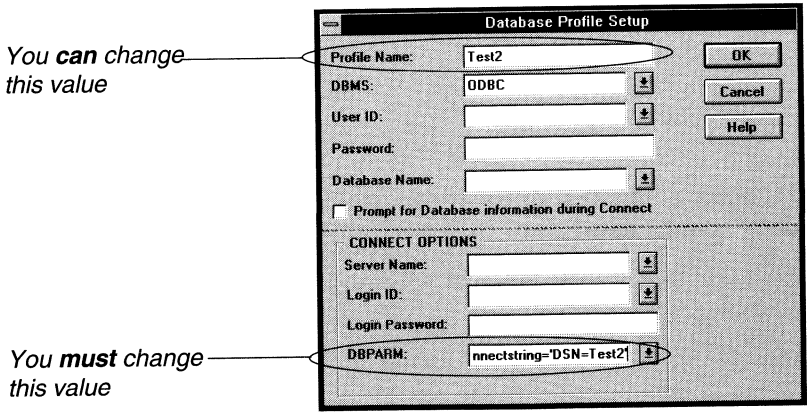
3 Position the insertion point to the right of DSN= in the DBParm field.



4 Edit the DSN (data source name) value in the DBParm field to match the data source name you supplied in the ODBC setup dialog box.

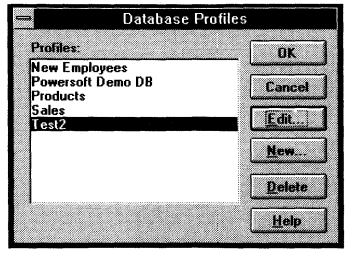
If you want, you can also change the profile name to match the new data source name. However, this is *not* required to connect to the database.

For example, here is the Database Profile Setup window after changing the profile name and data source name from Test1 to Test2.



5 Click OK in the Database Profile Setup dialog box.

The Database Profiles dialog box appears, with the name of the edited profile highlighted.



6 Click OK in the Database Profiles dialog box.

PowerBuilder or InfoMaker connects to the selected data source.

Deleting an ODBC data source definition

You can delete an ODBC data source definition if you no longer need to access its data.

You cannot delete the ODBC data source you are connected to

You *cannot* delete the definition of the ODBC data source to which you are currently connected in PowerBuilder or InfoMaker.

❖ To delete an ODBC data source definition:

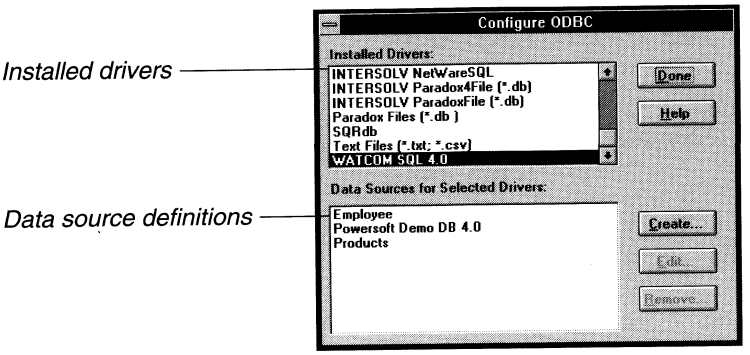
1 Click the Configure ODBC button in the PowerBar.
or
In the PowerBuilder or InfoMaker Database painter, select File➤Configure ODBC from the menu bar.

Configure ODBC button

If your PowerBar does not include the Configure ODBC button, use the customize feature to add the button to the PowerBar.

For instructions on customizing toolbars, see the PowerBuilder or InfoMaker *User's Guide*.

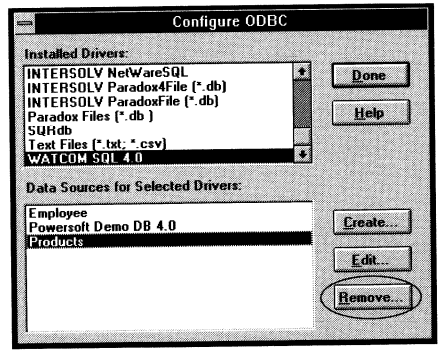
The Configure ODBC dialog box appears, listing the ODBC drivers installed on your computer and the data sources defined for each driver.



2 Select the ODBC driver that accesses the data source you want to delete.

The names of the data source definitions for that driver appear in the Data Sources for Selected Drivers list.

3 Select the name of the data source definition you want to delete.

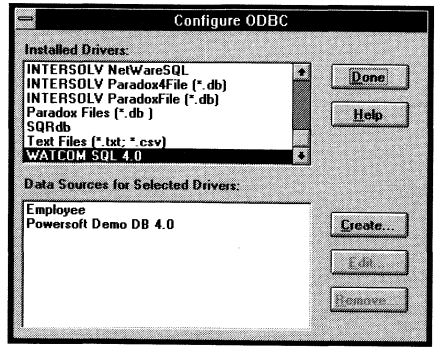


4 Click the Remove button.

A message box asks you to confirm the delete operation.

5 Click OK in the message box to delete the selected data source definition.

The Configure ODBC dialog box appears. The name of the data source definition you deleted disappears from the data source list.



6 Repeat steps 2 through 5 if you want to delete another ODBC data source definition.

7 Click Done when you are finished deleting ODBC data source definitions.

The Configure ODBC dialog box closes and you are returned to the painter workspace.

What happens

When you delete an ODBC data source definition, PowerBuilder or InfoMaker:

- Removes the definition from the ODBC.INI file
- Removes the database profile for this data source from the PB.INI or IM.INI file

Maintaining database profiles

You can easily edit or delete an existing database profile in PowerBuilder or InfoMaker.

Editing a database profile

You can edit a database profile to change one or more of its connection parameters.

❖ To edit a database profile:

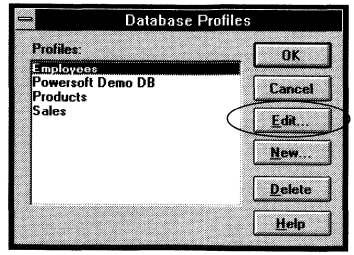
1 Click the Database Profile button in the PowerBar.
or
In any of the painters listed in the following table, select File➤Connect➤Setup from the menu bar.

Product	Painters
PowerBuilder	Database painter DataWindow painter Report painter
InfoMaker	Database painter Form painter Report painter

Database Profile button
If your PowerBar does not include the Database Profile button, use the customize feature to add the button to the PowerBar.

For instructions on customizing toolbars, see the PowerBuilder or InfoMaker *User's Guide*.

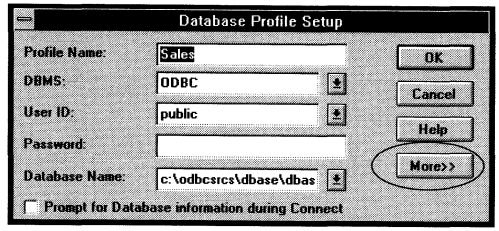
The Database Profiles dialog box appears, listing the names of existing profiles.



2 Select the name of the database profile you want to edit.

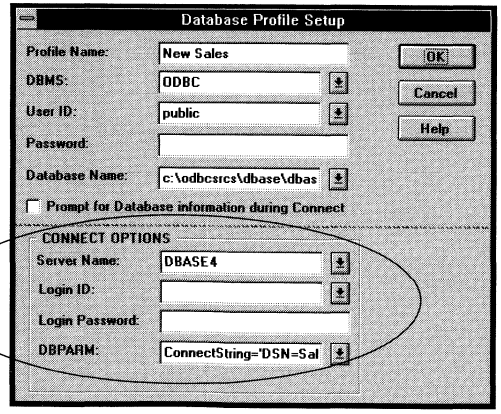
3 Click the Edit button.

The Database Profile Setup dialog box for the selected profile appears.



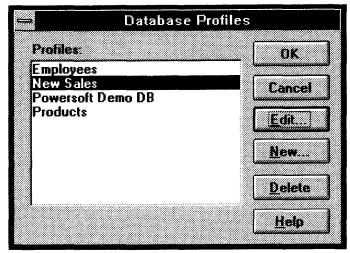
4 Edit the database profile as required for your database connection.

5 If necessary, click the More button to display additional connection options that you can edit.



6 Click OK in the Database Profile Setup dialog box.

The Database Profiles dialog box appears, with the name of the edited profile highlighted.



7 Click OK in the Database Profiles dialog box.

If you supplied sufficient information in the Database Profile Setup dialog box, PowerBuilder or InfoMaker connects to the specified database.

If the database profile does not supply enough information to connect to the database, various dialog boxes appear to prompt you for additional connection information. After you supply this information, PowerBuilder or InfoMaker connects to the database.

What happens

When you edit a database profile, PowerBuilder or InfoMaker updates the database profile in the PB.INI or IM.INI file.

For more information

For more about the values you should supply when editing a database profile, see the sections listed in the following table:

For more about	See
Defining an ODBC data source	Chapter 2, "Using ODBC Data Sources"
Defining a Powersoft database interface	Chapter 3, "Using Powersoft Database Interfaces"
Changing the ODBC data source name in the DBParm connect string	"Changing the ODBC data source name" on page 281
Setting DBParm parameters	Chapter 5, "Setting Additional Connection Parameters"

Deleting a database profile

You can delete a database profile if you no longer need to access its data.

❖ To delete a database profile:

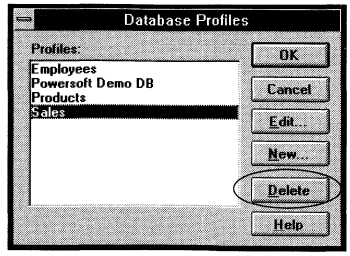
1 Click the Database Profile button in the PowerBar.
or
In any of the painters listed in the following table, select File➤Connect➤Setup from the menu bar.

Product	Painters
PowerBuilder	Database painter DataWindow painter Report painter
InfoMaker	Database painter Form painter Report painter

Database Profile button
If your PowerBar does not include the Database Profile button, use the customize feature to add the button to the PowerBar.

For instructions on customizing toolbars, see the PowerBuilder or InfoMaker *User's Guide*.

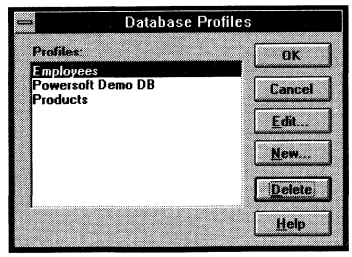
The Database Profiles dialog box appears, listing the names of existing profiles.



2 Select the name of the database profile you want to delete.

3 Click the Delete button.

The selected database profile disappears from the Profiles list.



4 Take one of the following actions:

- Click the Cancel button to close the Database Profiles dialog box. PowerBuilder or InfoMaker remains connected to the current database.
- Select another database profile and click OK to connect to that database.

What happens

When you delete a database profile, PowerBuilder or InfoMaker removes it from the PB.INI or IM.INI file.

Deleting a profile for an ODBC data source

If you delete a database profile that connects to an ODBC data source, PowerBuilder or InfoMaker does *not* delete the data source definition from the ODBC.INI file. This enables you to recreate the database profile at a later time if necessary without having to redefine the data source.

For instructions, see "Creating a profile for an existing ODBC data source" next.

Creating a profile for an existing ODBC data source

In most cases, you do not need to create a database profile for an ODBC data source. When you define the data source in PowerBuilder or InfoMaker, the database profile is created for you automatically.

However, you may want to create or recreate a database profile for an existing ODBC data source for the following reasons:

- You deleted the database profile for this data source and then decide later that you want to access this data.
- You used a tool *other* than PowerBuilder or InfoMaker to define an ODBC data source and then want to access this data from PowerBuilder or InfoMaker. (When you use a tool other than PowerBuilder or InfoMaker to define an ODBC data source, you must create the database profile yourself.)

When you delete a database profile that connects to an ODBC data source, PowerBuilder or InfoMaker does *not* delete the data source definition from the ODBC.INI file. This enables you to recreate the database profile at a later time without having to redefine the ODBC data source.

❖ To create a database profile for an existing ODBC data source:

1 Click the Database Profile button in the PowerBar.
or
In any of the PowerBuilder or InfoMaker painters that access the database, select File➤Connect➤Setup from the menu bar.

Product	Painters
PowerBuilder	Database painter DataWindow painter Report painter
InfoMaker	Database painter Form painter Report painter

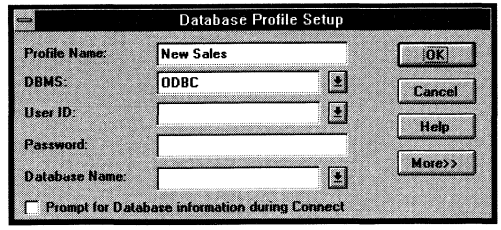
The Database Profiles dialog box appears, listing the names of existing profiles.

2 Click the New button to create a new database profile.

The Database Profile Setup dialog box appears.

3 Specify both of the following values in the Database Profile Setup dialog box:

- Enter the name of the profile in the Profile Name field.
- Enter or select ODBC in the DBMS field.

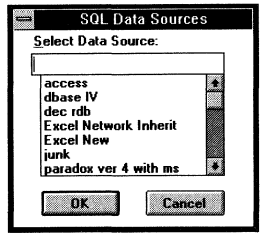


4 Click OK in the Database Profile Setup dialog box.

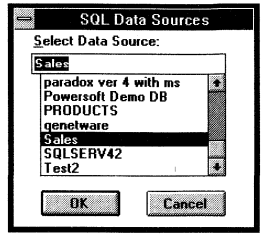
The SQL Data Sources dialog box appears, listing the names of existing ODBC data sources.

PowerBuilder Desktop
If you are using PowerBuilder Desktop, the SQL Data Sources dialog box lists only the names of *supported* ODBC data sources.

For a list of the ODBC data sources supported in PowerBuilder Desktop, see Appendix A, "Supported Data Sources and Databases."



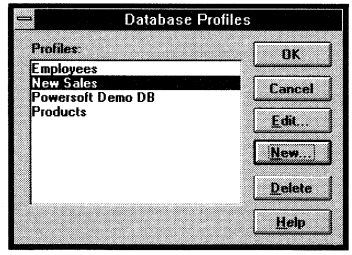
5 Select the name of the ODBC data source you want to access from the SQL Data Sources dialog box.



6 Click OK in the SQL Data Sources dialog box.

If you did not supply enough information in your data source definition for PowerBuilder or InfoMaker to connect, the driver that accesses this data source prompts you for additional connection parameters.

After you supply additional connection parameters as needed, the Database Profiles dialog box appears with the name of the new profile highlighted.



7 Click OK in the Database Profiles dialog box.

PowerBuilder or InfoMaker connects to the selected data source.

Sharing database profiles

You can share database profiles in PowerBuilder or InfoMaker by setting up a shared PB.INI or IM.INI file. The location of the shared INI file determines how you can share the profiles.

To share database profiles	Store the shared INI file
Among all PowerBuilder or InfoMaker users at your site	On a network file server accessible to all users
Between PowerBuilder and InfoMaker when both products are running on your computer	In a directory or folder on your computer

There are two ways to set up shared database profiles:

- In PowerBuilder or InfoMaker, by editing the PB.INI or IM.INI file to define the SharedINI variable
- In PowerBuilder, by using the Preferences painter to define the SharedINI variable

The following sections describe these methods, explain what happens in PowerBuilder or InfoMaker when you define shared profiles, and tell how to maintain shared profiles.

Editing the PB.INI or IM.INI file

In PowerBuilder or InfoMaker, you can define the SharedINI variable by editing the [PB] section of the PB.INI or IM.INI file.

❖ To set up shared database profiles by editing the PB.INI or IM.INI file:

1 Open the PB.INI or IM.INI file in one of the following ways. (The INI file resides in your PowerBuilder or InfoMaker product directory.)

- Use the File Editor in PowerBuilder or InfoMaker. (For instructions, see the *User's Guide*.)
- Use any text editor outside PowerBuilder or InfoMaker.

2 Edit the [PB] section of the INI file as follows:

```
[PB]
SharedIni = pathname_of_shared_INI_file
```

For example, if the shared INI file is I:\SHARE\IM.INI, edit the INI file like this:

```
[PB]
SharedIni = i:\share\im.ini
```

3 Save your changes to the INI file.

4 Exit PowerBuilder or InfoMaker if it is currently running.

You must exit PowerBuilder or InfoMaker for your changes to take effect.

5 Start PowerBuilder or InfoMaker.

When you open the Database Profiles dialog box or use the File➤Connect menu, your shared profiles are displayed, as shown in "What happens when you define shared profiles" on page 297.

Using the Preferences painter

In PowerBuilder, the easiest way to define the SharedINI variable is by using the Preferences painter.

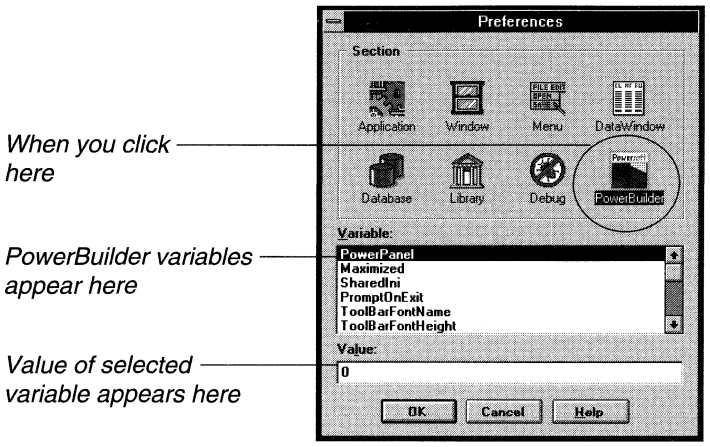
❖ To set up shared database profiles in PowerBuilder by using the Preferences painter:

1 Open the Preferences painter.

For instructions, see the PowerBuilder *User's Guide*.

2 Click the PowerBuilder button in the Preferences painter.

The variables for the [PB] (PowerBuilder) section of the PB.INI file appear in the Variable listbox.

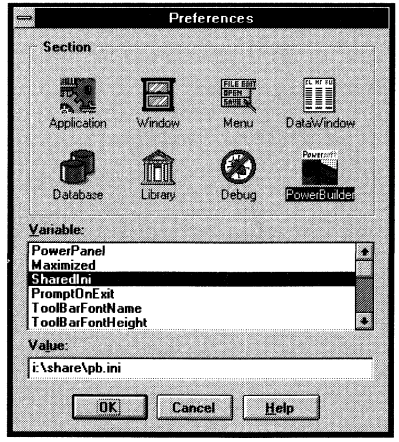


3 Select SharedIni in the Variable list.

The current value of the SharedINI variable appears in the Value box.

4 In the Value box, type the pathname of the shared INI file containing the database profiles.

In the following example, the shared INI file is located on the network in I:\SHARE\PB.INI.



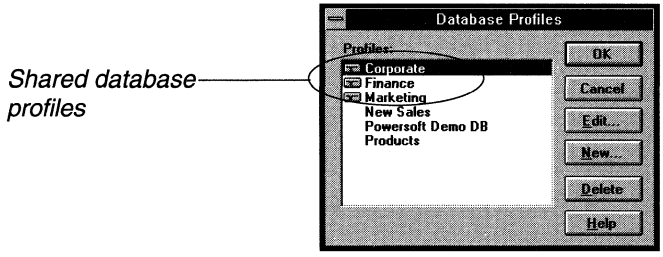
5 Click OK to save your changes.

The Preferences painter closes. The next time you open the Database Profiles dialog box or use the File➤Connect menu, your shared profiles are displayed, as shown in the next section.

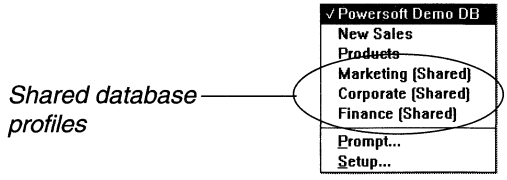
What happens when you define shared profiles

When you define shared database profiles:

- PowerBuilder or InfoMaker saves the SharedINI setting in the [PB] section of the PB.INI or IM.INI file.
- The Database Profiles dialog box displays all profiles defined in your local PB.INI or IM.INI file as well as those defined in the shared INI file. Shared database profiles appear with a network icon, as shown here:



- The File➤Connect menu displays (*Shared*) after the name of a shared database profile to indicate that it is stored in your shared INI file.



Maintaining shared database profiles

> **For administrators**
> If you maintain the shared database profiles at your site, read this section to learn how to update and save profiles in the shared INI file.

You can edit shared database profiles if necessary and save them in the local INI file on your computer. However, you *cannot* save or delete profiles stored in the shared INI file.

If you want to edit, delete, and save profiles in a shared PB.INI or IM.INI file, you must make the shared INI file the *active* INI file for a PowerBuilder or InfoMaker session. The active INI file is the one that PowerBuilder or InfoMaker reads to determine your painter preferences.

To make a shared INI file active, you must define a variable named InitPath040 in your WIN.INI file and assign it the drive and directory of the shared INI file. This tells PowerBuilder or InfoMaker where to find your INI file for the current session.

❖ To make your shared PB.INI or IM.INI file the active INI file:

1. Open the WIN.INI file in one of the following ways. (The WIN.INI file resides in your Windows directory.)
   - Use the File Editor in PowerBuilder or InfoMaker. (For instructions, see the *User's Guide*.)
   - Use any text editor outside PowerBuilder or InfoMaker.
2. Edit the WIN.INI file as follows, depending on whether you are using PowerBuilder or InfoMaker:

When using	Add these lines to WIN.INI
PowerBuilder	`[PowerBuilder]` `InitPath040=`*`drive:directory_of_shared_PB.INI`*
InfoMaker	`[InfoMaker]` `InitPath040=`*`drive:directory_of_shared_IM.INI`*

For example, if you are using PowerBuilder and the shared INI file is I:\SHARE\PB.INI, add these lines to the WIN.INI file:

```
[PowerBuilder]
InitPath040=i:\share
```

If you are using InfoMaker and the shared INI file is C:\IM\IM.INI, add these lines to WIN.INI:

```
[InfoMaker]
InitPath040=c:\im
```

3 Save your changes to the WIN.INI file.

The shared PB.INI or IM.INI file you specified becomes the active INI file. You can now make changes to the profiles and save them in the shared INI file. You can also delete profiles from the shared INI file.

4 Make changes as needed to the shared database profiles.

5 If you want to resume using the PB.INI or IM.INI file in your product directory as the active INI file, comment out the InitPath040 statement in the WIN.INI file by preceding it with a semicolon (;).

For example:

```
[PowerBuilder]
;InitPath040=i:\share
```

The PB.INI or IM.INI file in your product directory becomes the active INI file.

How PowerBuilder or InfoMaker finds the INI file

PowerBuilder or InfoMaker looks for the PB.INI or IM.INI file in the following directories, in this order:

1 The directory specified by the InitPath040 variable.

2 If InitPath040 is not defined, the directory specified by the InitPath variable. This is for compatibility with pre–4.0 versions of PowerBuilder or InfoMaker.

3 If neither InitPath040 nor InitPath is defined, the current PowerBuilder or InfoMaker directory.

CHAPTER 5

Setting Additional Connection Parameters

About this chapter

Chapters 2 and 3 describe the *basic* connection parameters you must supply to define an ODBC data source or Powersoft database interface. To fine-tune your database connection and take advantage of DBMS-specific features that PowerBuilder and InfoMaker support, you can set additional connection parameters anytime. These additional connection parameters include:

- DBParm parameters
- Database preferences

This chapter describes how to set DBParm parameters and database preferences in PowerBuilder or InfoMaker.

Contents

Topic	Page
Basic steps for setting connection parameters	302
Setting DBParm parameters	303
DBParm parameters and supported DBMSs	309
Descriptions of DBParm parameters	312
Setting database preferences	374
Database preferences and supported DBMSs	379
Descriptions of database preferences	381

Basic steps for setting connection parameters

This section gives the basic steps for setting DBParm parameters and database preferences in PowerBuilder or InfoMaker.

❖ To set DBParm parameters:

1 Learn how to set DBParm parameters in a database profile or a PowerBuilder application.

For instructions, see "Setting DBParm parameters" on page 303.

2 Determine the DBParm parameters you can set for your DBMS.

For a table listing DBParm parameters and the DBMSs to which they apply, see "DBParm parameters and supported DBMSs" on page 309.

3 Read the description of the DBParm parameter you want to set.

See the section for your DBParm parameter in "Descriptions of DBParm parameters" on pages 312 through 372. The DBParm parameters are described in alphabetical order.

4 Set the DBParm parameter for your database connection.

❖ To set database preferences:

1 Learn how to set database preferences by editing the PB.INI or IM.INI file or, if you are using PowerBuilder, by using the Preferences painter.

For instructions, see "Setting database preferences" on page 374.

2 Determine the database preferences you can set for your DBMS.

For a table listing database preferences and the DBMSs to which they apply, see "Database preferences and supported DBMSs" on page 379.

3 Read the description of the database preference you want to set.

See the section for your database preference in "Descriptions of database preferences" on pages 381 through 390. The database preferences are described in alphabetical order.

4 Set the database preference for your database connection.

Setting DBParm parameters

There are two ways to set DBParm parameters in PowerBuilder or InfoMaker, as summarized in the following table:

Set DBParm parameters in	If you are using	To do this
A database profile	PowerBuilder or InfoMaker	Connect to a database
A PowerBuilder script	PowerBuilder	Develop an application that connects to a database

The following sections give the steps for using each of these methods.

Setting DBParm parameters in InfoMaker
Editing a database profile is the *only* way to set DBParm parameters in InfoMaker.

Editing a database profile

To set DBParm parameters for a database connection in PowerBuilder or InfoMaker, edit the database profile for that connection as described in the following procedure. A database profile is automatically created when you:

- Define an ODBC data source in PowerBuilder or InfoMaker by completing the ODBC setup window for your data source driver. (For instructions, see Chapter 2, "Using ODBC Data Sources.")
- Define a Powersoft database interface in PowerBuilder or InfoMaker by completing the Database Profile Setup dialog box. (For instructions, see Chapter 3, "Using Powersoft Database Interfaces.")

❖ To set DBParm parameters in a database profile:

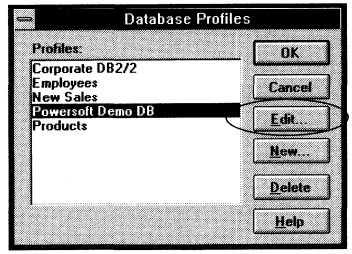
1 Click the Database Profile button in the PowerBar.
or
In any of the painters listed in the following table, select File➤Connect➤Setup from the menu bar.

Product	Painters
PowerBuilder	Database painter DataWindow painter Report painter
InfoMaker	Database painter Form painter Report painter

Database Profile button
If your PowerBar does not include the Database Profile button, use the customize feature to add the button to the PowerBar.

For instructions on customizing toolbars, see the PowerBuilder or InfoMaker *User's Guide*.

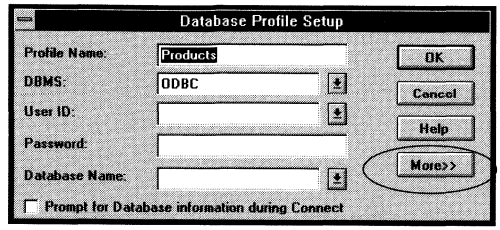
The Database Profiles dialog box appears, listing the names of existing profiles. If you are currently connected to a database profile, its name is highlighted.



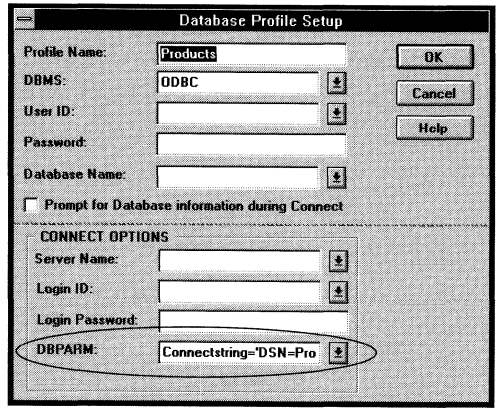
2 Select the name of the database profile you want to edit.

3 Click the Edit button.

The Database Profile Setup dialog box for the selected profile appears.

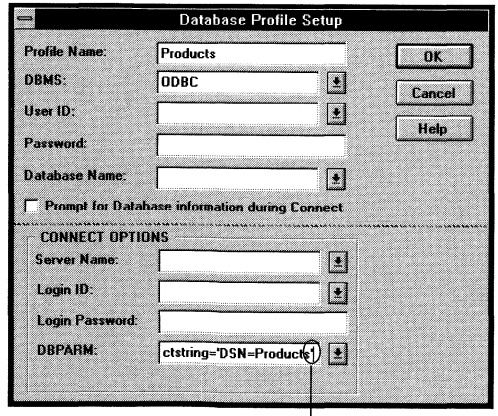


4 Click the More button to display the DBParm dropdown listbox.



5 Position the insertion point in the DBParm field.

If you are supplying a DBParm parameter for an ODBC data source, position the insertion point *after* the Connectstring value enclosed in single quotes.



6 Use the following syntax in the DBParm field to set one or more parameters. Make sure to separate the list of parameters with commas.

parameter_1 = value, parameter_2 = value, parameter_n = value

For examples using the Async DBParm parameter, see the Examples section next.

7 Click OK.

PowerBuilder or InfoMaker connects to the database profile you edited.

Examples

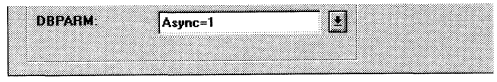
For those DBMSs that support it, you can specify Async=1 in the DBParm field to enable asynchronous operations on the database. This means that you can cancel the current operation or start a new operation in PowerBuilder or InfoMaker before the current one completes.

Example 1 To set the Async DBParm *for an ODBC data source*, type the expression shown in bold in the DBParm field *after* the Connectstring value, as follows:

```
Connectstring='DSN=Test;UID=PB;PWD=xyz',Async=1
```

Example 2 To set the Async DBParm *for a Powersoft database interface*, type the following expression in the DBParm field. (The profile for a Powersoft database interface does not contain a Connectstring value.)

```
Async = 1
```



Setting DBParm parameters in a PowerBuilder script

For PowerBuilder developers only
Read this section if you are a PowerBuilder developer who wants to specify connection information, including DBParm parameters, in a PowerBuilder application script.

For more about using transaction objects to communicate with a database in a PowerBuilder application, see *Building Applications*.

If you are developing a PowerBuilder application that connects to a database, you must specify the required connection parameters in the appropriate script. For example, you might specify connection parameters in the script that opens the application.

One of the connection parameters you may want to specify in a script is DBParm. You can do this in two ways:

- By setting values for the DBParm attribute of the default transaction object
- By reading DBParm values from an external text file

Setting values for the DBParm attribute

One way to specify connection parameters in a script is by assigning values to attributes of the default transaction object. PowerBuilder uses a special, nongraphic object called a **transaction object** to communicate with the database. The default transaction object is named SQLCA, which stands for SQL Communications Area.

SQLCA has 15 attributes, ten of which are used to connect to your database. One of the ten connection attributes is DBParm. DBParm contains DBMS-specific parameters that enable your application to use various features supported by the database.

❖ To set values for the DBParm attribute in a PowerBuilder script:

1 Open the application script in which you want to specify connection parameters.

For instructions, see the PowerBuilder *User's Guide*.

2 Use the following PowerScript syntax to specify DBParm parameters. Make sure you separate the DBParm parameters with commas, and enclose the entire DBParm string in double quotes.

sqlca.dbParm = "*parameter_1,parameter_2,parameter_n***"**

For example, the following statement in a PowerBuilder script sets the DBParm attribute for an ODBC data source named Sales. In this example, the DBParm attribute consists of two parameters: Connectstring and Async.

```
sqlca.dbParm="Connectstring='DSN=Sales;UID=PB;PWD=xyz',Async=1"
```

3 Compile the PowerBuilder script to save your changes.

For instructions, see the PowerBuilder *User's Guide*.

Reading DBParm values from an external text file

You can use the PowerScript ProfileString function to read DBParm values from an external text file. The DBParm values can come from the [Database] section of your PB.INI file, or from a specified section of an application-specific INI file.

❖ To read DBParm values from an external text file:

1 Open the application script in which you want to specify connection parameters.

For instructions, see the PowerBuilder *User's Guide*.

2 Use the following PowerScript syntax to specify the ProfileString function with the SQLCA DBParm attribute:

sqlca.dbParm = ProfileString (*file, section, variable, default_value*)

For example, the following statement in a PowerBuilder script reads the DBParm values from the [Database] section of the PB.INI file:

```
sqlca.dbParm=ProfileString("PB.INI","Database","dbParm","")
```

3 Compile the PowerBuilder script to save your changes.

For instructions, see the PowerBuilder *User's Guide*.

DBParm parameters and supported DBMSs

The table on pages 310 and 311 lists all of the DBParm parameters that you can set in PowerBuilder or InfoMaker and the DBMSs to which each parameter applies. The DBParm parameters are listed in alphabetical order.

A checkmark indicates that you can set a particular DBParm parameter when connecting to this DBMS.

DBParm parameters	ALLBASE/ SQL	IBM DRDA	INFOR-MIX	MDI Gateway	ODBC	ORACLE
AppName						
Async	✓				✓	
Block						✓
CharSet						
ConnectString					✓	
ConversionTable				✓		
CursorLib					✓	
CursorLock					✓	
CursorScroll					✓	
CursorUpdate						
Date						✓
DateTime						✓
DBAdm		✓				
DBGetTime					✓	
DBTextLimit						
DelimitIdentifier		✓		✓	✓	✓
DisableBind			✓		✓	✓
GroupID		✓		✓		
Host						
HPConnect	✓					
INET_DBPATH			✓			
INET_PROTOCOL			✓			
INET_SERVICE			✓			
Language						
Log						
LoginTimeOut					✓	
MixedCase						✓
MsgTerse					✓	
PBCatalogOwner		✓		✓	✓	
PBDBMS						✓
Recovery						
Release						
Request				✓		
Scroll			✓			
SQLCache					✓	✓
SQLQualifiers						
SystemOwner		✓		✓		
TableCriteria		✓		✓	✓	✓
Time						✓

DBParm parameters	SQL Server	SQL Base	Sybase Net-Gateway	Sybase SQL Server System 10	XDB
AppName	✓			✓	
Async	✓	✓		✓	
Block				✓	
CharSet				✓	
ConnectString					
ConversionTable					
CursorLib					
CursorLock	✓				
CursorScroll	✓				
CursorUpdate				✓	
Date					
DateTime					
DBAdm					
DBGetTime	✓			✓	
DBTextLimit	✓				
DelimitIdentifier			✓	✓	
DisableBind		✓			
GroupID			✓		
Host	✓			✓	
HPConnect					
INET_DBPATH					
INET_PROTOCOL					
INET_SERVICE					
Language				✓	
Log	✓			✓	
LoginTimeOut					
MixedCase					
MsgTerse					
PBCatalogOwner			✓		✓
PBDBMS					
Recovery		✓			
Release	✓				✓
Request					
Scroll					
SQLCache					
SQLQualifiers			✓		
SystemOwner			✓		✓
TableCriteria			✓		✓
Time					

Descriptions of DBParm parameters

This section describes the syntax and use of each DBParm parameter listed in the table on pages 310 and 311. The DBParm parameters are described in alphabetical order.

Syntax and examples
Although you can set DBParm parameters in a database profile or in a PowerBuilder script, most of the syntax and examples in this section show only how to set DBParm parameters in a database profile.

For an example showing how to set a DBParm parameter in a script, see "Setting DBParm parameters in a PowerBuilder script" on page 307.

AppName

Description

Specifies the application name you want to use when connecting to a SQL Server or Sybase SQL Server System 10 database in PowerBuilder or InfoMaker.

When to specify AppName
You must specify the AppName DBParm parameter *before* connecting to a SQL Server or Sybase SQL Server System 10 database in PowerBuilder or InfoMaker.

Applies to

SQL Server
Sybase SQL Server System 10

Syntax

AppName = '*application_name*'

Default value

The default value for the AppName parameter depends on the DBMS you are accessing, as summarized in the following table:

DBMS	AppName default value
SQL Server	There is no default value for AppName. PowerBuilder or InfoMaker does not set the AppName parameter unless you specify a value.
Sybase SQL Server System 10	PowerBuilder or InfoMaker sets the Sybase SQL Server System 10 CS_APPNAME connection property to PowerBuilder, as follows: `AppName = 'PowerBuilder'`

Usage

It is useful to specify a different AppName value for each of your SQL Server applications. If you are a SQL Server administrator, you can query the MASTER.DBO.SYSPROCESSES table to determine which applications are running on the database server. The value specified for AppName appears in the program_name field of the SYSPROCESSES table, making it easy to identify the applications.

You can use the AppName and Host parameters in a single DBParm statement to specify both the application name and the host name. (For more, see the description of the Host parameter on page 341.)

Examples

Example 1 This statement sets the application name to Test:

```
AppName = 'Test'
```

Example 2 This statement sets the application name to Sales and the host name to Fran:

```
AppName = 'Sales',Host = 'Fran'
```

See also

Host

Async

Description

Allows you to perform asynchronous operations on your database in PowerBuilder or InfoMaker. Essentially, this means that you can cancel the current operation or start another operation before the current one completes.

By default, PowerBuilder or InfoMaker operates synchronously.

Applies to

ALLBASE/SQL
ODBC
SQL Server
SQLBase
Sybase SQL Server System 10

Syntax

Async = *value*

Parameter	Description
value	A value specifying synchronous or asynchronous operation. Values are: ♦ **0** (Default) Synchronous operation ♦ **1** Asynchronous operation

Default value

```
Async = 0
```

Usage

Setting the Async DBParm parameter to 1 to allow asynchronous operation has an effect only on DataWindow controls, reports, and forms for which you have coded a RetrieveRow event.

Enabling asynchronous operation in PowerBuilder or InfoMaker is useful when you are executing a complex SQL statement that takes several minutes to return results. If the Async parameter is set to 1, you can do either of the following while the SQL statement is executing:

- Work in another window
- Cancel the statement before it retrieves the first row of data

When you set Async to 1 for an ODBC data source, SQL Server database, or Sybase SQL Server System 10 database, you can also set the DBGetTime parameter. DBGetTime specifies the number of seconds you want PowerBuilder or InfoMaker to wait for a response from the DBMS when you retrieve rows in a DataWindow, report, or form. (For more, see the description of DBGetTime on page 334.)

If you are communicating with the database in a PowerBuilder script, you can reset the Async value at any time before or after the transaction object has connected to the database.

Examples

Example 1 This statement enables asynchronous operation:

```
Async = 1
```

Example 2 This statement enables asynchronous operation and sets the DBGetTime parameter to 20 seconds:

```
Async = 1, DBGetTime = 20
```

See also

DBGetTime

Block (ORACLE)

Description

Specifies the cursor blocking factor when connecting to an ORACLE database. The blocking factor determines the number of rows that a DataWindow object or report can fetch from an ORACLE database at one time.

> **Using the default blocking factor**
> In most cases, the default blocking factor used by PowerBuilder and InfoMaker should meet your needs when connecting to an ORACLE database. Therefore, you should not have to set a nondefault value for Block.

Applies to

ORACLE

Syntax

Block = *blocking_factor*

Parameter	Description
blocking_factor	The number of rows you want the DataWindow object to fetch from the database at one time. The blocking factor can be a number from 1 to 1000, inclusive.

Default value

The default value for the Block parameter depends on whether you are accessing an ORACLE 6 or ORACLE 7 database, as summarized in the following table:

ORACLE database interface	Block default value
OR6 (ORACLE Version 6)	PowerBuilder or InfoMaker uses the following formula to determine the number of rows to fetch at one time, up to a maximum of 32 K for *all* columns: 32,000 / *row_size*
OR7 (ORACLE Version 7)	PowerBuilder or InfoMaker sets the blocking factor to the largest value possible to fetch the maximum number of rows at one time, up to a maximum of 32 K *per* column.

Example

This statement sets the blocking factor to 1000 rows:

```
Block = 1000
```

Block (Sybase System 10)

Description

Specifies the internal blocking factor used by the Sybase Client Library (CT-Lib) interface when declaring a cursor. The blocking factor determines the number of rows that are fetched from the database at one time when CT-Lib makes a physical request for data.

When you are connected to a Sybase SQL Server System 10 database, the Block DBParm parameter applies only to declared cursors and not to DataWindow objects.

Applies to

Sybase SQL Server System 10

Syntax

Block = *blocking_factor*

Parameter	Description
blocking_factor	The number of rows that are fetched from the database at one time when CT-Lib makes a physical request for data. The default blocking factor is 100 rows.

Default value

```
Block = 100
```

Examples

Example 1 This statement sets the blocking factor to 1000 rows:

```
Block = 1000
```

Example 2 The following embedded SQL statements show how to set the blocking factor in a PowerBuilder application script and use it to declare a cursor. These statements set the blocking factor to 1000 rows and then declare a cursor that uses this internal blocking factor.

```
sqlca.dbParm = "Block = 1000"
DECLARE dept_cursor CURSOR FOR
   SELECT dept_id, dept_name FROM DEPARTMENT
   USING SQLCA;
OPEN dept_cursor;
```

CharSet

Description

Specifies the character set you want to use when connecting to a Sybase SQL Server System 10 database in PowerBuilder or InfoMaker.

When you specify a value for CharSet, PowerBuilder or InfoMaker does the following:

- Allocates a CS_LOCALE structure for this connection
- Sets the CS_LC_CTYPE value to the character set you specify
- Sets the SQL Server CS_LOC_PROP connection property with the new locale information

When to specify CharSet
You must specify the CharSet DBParm parameter *before* connecting to a Sybase SQL Server System 10 database in PowerBuilder or InfoMaker.

Applies to

Sybase SQL Server System 10

Syntax

CharSet = *'character_set'*

Default value

There is no default value for the CharSet DBParm parameter.

Example

This statement allocates a CS_LOCALE structure for the Sybase System 10 connection and sets the CS_LC_CTYPE value to the iso_1 character set:

```
CharSet = 'iso_1'
```

ConnectString

Description

Specifies the parameters required to connect to an ODBC data source. PowerBuilder and InfoMaker use these parameters to connect to the database.

How ConnectString is specified
PowerBuilder or InfoMaker generates the ConnectString automatically when you define an ODBC data source and copies it to the DBParm field in the Database Profile Setup dialog box. This happens *before* you connect to the data source in PowerBuilder or InfoMaker.

Therefore, you do *not* have to enter the ConnectString yourself when defining an ODBC data source. However, you may need to edit the ConnectString value in the Database Profile Setup dialog box. (For more, see the Usage section below.)

Applies to

ODBC

Syntax

The ConnectString syntax appears on a single line. You must enclose the entire ConnectString in single quotes and separate parameters within the ConnectString with semicolons.

ConnectString = ' **DSN** = *data_source_name*; {**UID** = *user_ID*; **PWD** = *password*; *other_required_parameters*} '

Parameter	Description
data_source_name	A name that identifies the data source
user_ID (optional)	The user ID required to connect to the data source
password (optional)	The password required by *user_ID* to connect to the data source
other_required_parameters (optional)	Any other data source-specific parameters required to connect. For example, some ODBC drivers specify the data source directory here. If you are using the INTERSOLV Btrieve ODBC driver, you can specify the value **CDB = 1** here to create a new NetWare SQL data dictionary if one does not already exist.

Default value

There is no default value for the ConnectString parameter.

Usage

You can change the ConnectString parameter if necessary by editing it in the DBParm field of the Database Profile Setup dialog box. For example, if you change the name of an existing ODBC data source, you should edit its database profile to update the connect string with the new DSN (data source name) value.

For instructions, see "Editing an ODBC data source definition" in Chapter 4, "Managing Database Connections."

Examples

Example 1 This example shows a connect string for an ODBC data source that contains the DSN (data source name), UID (user ID), and PWD (password). The entire connect string is enclosed in single quotes and parameters within the connect string are separated by semicolons.

```
Connectstring = 'DSN=Demo DB;UID=dba;PWD=sql'
```

Example 2 This example shows a connect string for a Btrieve data source accessed with the INTERSOLV ODBC Btrieve driver. The connect string consists of two parameters: DSN and CDB (to create a new NetWare SQL data dictionary). The entire connect string is enclosed in single quotes and parameters within the connect string are separated by semicolons.

```
Connectstring = 'DSN=Btrieve;CDB=1'
```

Example 3 This example shows how to specify a DBParm string that contains a connect string and another DBParm parameter (Async). The connect string is enclosed in single quotes and semicolons separate its three parameters (DSN, UID, and PWD). A comma at the end of the connect string separates it from the Async parameter.

```
Connectstring='DSN=Test;UID=PB;PWD=xyz',Async=1
```

ConversionTable

Description

When you connect to an IBM DB2 database through the Powersoft Micro Decisionware Database Gateway Interface for DB2, the ConversionTable parameter specifies the name of the table that contains the custom data type mapping you want to use.

You can set the ConversionTable parameter when you want PowerBuilder or InfoMaker to use custom data type mapping instead of the standard data type mapping provided by the Micro Decisionware Database Gateway. By default, PowerBuilder or InfoMaker uses the standard mapping.

When to specify ConversionTable
You must specify the ConversionTable DBParm parameter *before* connecting to a DB2 database in PowerBuilder or InfoMaker through the Powersoft Micro Decisionware Database Gateway Interface for DB2.

Applies to

Micro Decisionware Database Gateway Interface for DB2

Syntax

ConversionTable = '*table_name*'

Parameter	Description
table_name	Specifies the name of the table that contains the custom data type mapping you want to use when connecting to a DB2 database through the Powersoft Micro Decisionware Database Gateway Interface for DB2. By default, PowerBuilder or InfoMaker uses the standard data type mapping provided by the Micro Decisionware Database Gateway.

Default value

If you do not specify a value for ConversionTable, PowerBuilder or InfoMaker executes the following statement to use the standard data type mapping provided by the Micro Decisionware Database Gateway:

SET CONVERT ALL = STD

Usage

When you specify a value for ConversionTable, PowerBuilder or InfoMaker executes the following statement to use the custom data type mapping contained in the *table_name* you specify:

SET CONVERT ALL = *table_name*

> **No validation performed**
> When you use the ConversionTable DBParm parameter, PowerBuilder or InfoMaker does *not* validate the custom data type mapping in the table you specify, and ignores any errors or data type mismatches that may occur.

Example

This statement specifies Custom as the name of the table containing the custom data type mapping you want to use when connecting to a DB2 database through the Powersoft Micro Decisionware Database Gateway Interface for DB2:

```
ConversionTable = 'Custom'
```

When you connect to the database after setting the ConversionTable parameter, PowerBuilder or InfoMaker executes the following statement to use the data type mapping in the Custom table:

SET CONVERT ALL = Custom

CursorLib

Description

Specifies the cursor library to use when connecting to an ODBC data source.

Applies to

ODBC

Syntax

CursorLib = '*value*'

Parameter	Description
value	The cursor library to use when connecting to an ODBC data source. Values are: ♦ **ODBC_Cur_Lib** Use the ODBC Version 2.0 cursor library. ♦ **If_Needed** Use the ODBC Version 2.0 cursor library if your ODBC driver does *not* support cursors. ♦ **Driver_Cursors** (Default) Use your data source's native cursor support.

Default value

```
CursorLib = 'Driver_Cursors'
```

Example

This statement specifies use of the ODBC Version 2.0 cursor library when connecting to an ODBC data source:

```
CursorLib = 'ODBC_Cur_Lib'
```

CursorLock (ODBC)

Description

When used with the CursorScroll parameter, specifies locking options for cursors in ODBC data source.

The values you can set for CursorLock control two aspects of cursor locking:

- **Concurrent access** Ensures that users can simultaneously access data that is accurate and current.
- **Collision detection** Detects collisions that occur when multiple users update the same data at the same time.

Applies to

ODBC

Syntax

CursorLock = '*lock_value*'

Parameter	Description
lock_value	Specifies the type of locking you want to use for ODBC cursors. Values are: ♦ **Lock** Uses the lowest level of locking sufficient to allow updates on table rows. ♦ **Opt** Uses optimistic concurrency control. This means that table rows are *not* locked against updates by other users. To detect collisions, compares row versions or timestamps. ♦ **OptValue** Uses optimistic concurrency control. This means that table rows are *not* locked against updates by other users. To detect collisions, compares selected values with their previous values. ♦ **ReadOnly** Prohibits updates on table rows by any user. For more about how the ODBC standard defines lock values, see your ODBC documentation.

Default value

If you do not specify a value for CursorLock, PowerBuilder or InfoMaker defaults to the cursor lock settings specified for your ODBC data source driver.

Example

This statement sets scrolling and locking options for cursors in an ODBC data source:

```
CursorScroll='Dynamic',CursorLock='OptValue'
```

See also

CursorScroll (ODBC)

CursorLock (SQL Server)

Description

When used with the Release and CursorScroll parameters, specifies locking options for cursors that access a SQL Server database Version 4.2 or higher.

The values you can set for CursorLock control two aspects of cursor locking:

- **Concurrent access** Ensures that users can simultaneously access data that is accurate and current.
- **Collision detection** Detects collisions that occur when multiple users update the same data at the same time.

> **Using CursorLock with SQL Server**
> To use the CursorLock DBParm parameter with a SQL Server database, you *must* set the Release DBParm parameter to 4.2 when you create the SQL Server database profile. This allows you to use DB-Library cursor support in SQL Server Version 4.2 or higher.
>
> For a sample DBParm statement that includes the Release, CursorScroll, and CursorLock parameters, see the Example section below.

Applies to

SQL Server

Syntax

CursorLock = '*lock_value*'

Parameter	Description
lock_value	Specifies the type of locking you want to use for SQL Server cursors. Values are: ♦ **Lock** ♦ **Opt** (Default) ♦ **OptVal** ♦ **ReadOnly** For more about how these values control concurrent access and detect collisions, see the table in the Usage section below.

Default value

```
CursorLock = 'Opt'
```

Usage

The following table briefly describes how each CursorLock value controls concurrent access and detects collisions in a SQL Server database. (For more, see your SQL Server documentation.)

CursorLock value	Concurrency control	Collision detection
Lock	Locks table rows when they are fetched inside a SQL transaction. No other user can update or read these rows.	Collision detection is unnecessary because locking the rows ensures that any updates made by the owner of the cursor will succeed.
Opt	Does not lock table rows. Other users can update or read these rows. This is called **optimistic concurrency control**.	Compares timestamps if available. If timestamps are not available, saves and compares the values of all nontext, nonimage columns in the tables with their previous values.
OptVal	Does not lock table rows. Other users can update or read these rows. This is called **optimistic concurrency control**.	Compares selected values whether or not a timestamp is available.
ReadOnly	Prohibits updates on table rows by any user.	Collision detection is unnecessary because no updates on the rows are allowed.

Example

This statement sets scrolling and locking options for cursors in a SQL Server database that is Version 4.2 or higher. Type the statement on a single line.

```
Release='4.2',CursorScroll='Dynamic',
   CursorLock='OptVal'
```

See also

CursorScroll (SQL Server)
Release

CursorScroll (ODBC)

Description

When used with the CursorLock parameter, specifies scrolling options for cursors in an ODBC data source.

The location of a cursor indicates the current position in the result set produced by a SQL statement. **Scrolling** allows a cursor to move through the data in a result set one row at a time.

Applies to

ODBC

Syntax

CursorScroll = '*scroll_value*'

Parameter	Description
scroll_value	Specifies the type of scrolling you want to use for ODBC cursors. Values are: ♦ **Forward** The cursor only scrolls forward through the result set. ♦ **Static** The data in the result set does not change. ♦ **KeySet** The driver saves and uses the keys for a specified number of rows called the keyset. You specify the size of the keyset as an integer value, described in the last bullet below. ♦ **Dynamic** The driver saves and uses only the keys for the rows specified in the rowset. ♦ For cursors that use KeySet scrolling, you can specify a 32-bit integer value that is the number of rows in your keyset. The default keyset size is 0. Specifying an integer greater than zero lets you use a **mixed cursor**. A mixed cursor uses KeySet scrolling within the keyset and Dynamic scrolling outside the keyset.

Default value

If you do not specify a value for CursorScroll, PowerBuilder or InfoMaker defaults to the cursor scroll settings specified for your ODBC data source driver.

Examples

Example 1 This statement sets scrolling and locking options for cursors in an ODBC data source:

```
CursorScroll = 'Dynamic',CursorLock = 'OptValue'
```

Example 2 This statement sets the number of rows in the keyset to 100:

```
CursorScroll = 100
```

See also CursorLock (ODBC)

CursorScroll (SQL Server)

Description

When used with the Release and CursorLock parameters, specifies scrolling options for cursors in a SQL Server database Version 4.2 or higher.

The location of a cursor indicates the current position in the result set produced by a SQL statement. Scrolling allows a cursor to move through the data in a result set one row at a time.

> **Using CursorScroll with SQL Server**
> To use the CursorScroll DBParm parameter with a SQL Server database, you *must* set the Release DBParm parameter to 4.2 when you create the SQL Server database profile. This allows you to use DB-Library cursor support in SQL Server Version 4.2 or higher.
>
> For sample DBParm statements that include the Release, CursorScroll, and CursorLock parameters, see the Examples section below.

Applies to SQL Server

Syntax

CursorScroll = '*scroll_value*'

Parameter	Description
scroll_value	Specifies the type of scrolling you want to use for SQL Server cursors. Values are: ♦ **Forward** The cursor only scrolls forward through the result set. Values, order, and membership in the result set do not change while the cursor is open. ♦ **KeySet** The driver saves and uses the keys for a specified number of rows called the keyset. You specify the size of the keyset as an integer value, described in the last bullet below. Values in the result set can change, but order and membership remain fixed at the moment a cursor is opened. ♦ **Dynamic** (Default) The driver saves and uses only the keys for the rows specified in the rowset. Values, order, and membership in the result set can all change. ♦ For cursors that use KeySet scrolling, you can specify a 32-bit integer value that is the number of rows in your keyset. SQL Server puts this number of rows in the keyset memory buffer. The default keyset size is 0. Specifying an integer greater than zero means that you are using a mixed cursor. A mixed cursor uses KeySet scrolling within the keyset and Dynamic scrolling outside the keyset. Using a mixed cursor may be useful for large result sets, especially if the number of rows in your keyset exceeds the memory capacity of your computer. When you use KeySet scrolling, SQL Server tries to create a buffer on your computer that is large enough to hold all the rows in your keyset. If the number of rows exceeds your computer's memory capacity, you may get an out-of-memory error.

Default value

```
CursorScroll = 'Dynamic'
```

Examples

Example 1 This statement sets scrolling and locking options for cursors in a SQL Server database that is Version 4.2 or higher. Type the statement on a single line.

```
Release = '4.2',CursorScroll = 'Dynamic',
   CursorLock = 'OptVal'
```

Example 2 This statement sets the number of rows in the keyset to 100:

```
Release = '4.2', CursorScroll = 100
```

See also

CursorLock (SQL Server)
Release

CursorUpdate

Description

Specifies whether cursors in a Sybase SQL Server System 10 database are declared read-only or updatable.

By default, cursors are declared read-only.

Applies to

Sybase SQL Server System 10

Syntax

CursorUpdate = *value*

Parameter	Description
value	A number that specifies whether Sybase System 10 cursors are declared read-only or updatable. Values are: ♦ **0** (Default) Cursors are declared read-only. Sybase Client Library cursor declarations include the CS_READ_ONLY option. ♦ **1** Cursors are declared updatable. Sybase Client Library cursor declarations include the CS_FOR_UPDATE option. This option applies to all updatable columns in the table.

Default value

```
CursorUpdate = 0
```

Usage

Set the CursorUpdate parameter to 1 to declare updatable cursors if you plan to use either of the following SQL statements in your application. (In these statements, *table* represents the table name and *cursor* represents the cursor name.)

```
DELETE FROM table WHERE CURRENT OF cursor
UPDATE table SET set_clause WHERE CURRENT OF cursor
```

If you are communicating with the database in a PowerBuilder script, you can reset the CursorUpdate value at any time before or after the transaction object has connected to the database.

Example

This statement specifies that Sybase System 10 cursors are declared updatable:

```
CursorUpdate = 1
```

Date

Description

When you update a data source in the DataWindow painter or Form painter, PowerBuilder or InfoMaker builds a SQL UPDATE statement in the background. If your data source is an ORACLE table, the Date DBParm parameter determines how PowerBuilder or InfoMaker specifies a date data type when it builds the SQL UPDATE statement.

Applies to

ORACLE

Syntax

Date = ' ""*Powersoft_date_format*"" '

Typing single quotes
When PowerBuilder or InfoMaker parses the DBParm Date string, it converts each set of four consecutive single quotes you type to one single quote. Therefore, you *must* type the syntax exactly as described in the following table.

Parameter	Description
' ''''	Type a single quote, followed by one space, followed by four single quotes. There is no space between the four single quotes and the beginning of the Powersoft date format.
Powersoft_date_format	The date format you want PowerBuilder or InfoMaker to use when it builds a SQL UPDATE statement to update a data source in the DataWindow or Form painters. (For more on Powersoft display formats, see the *User's Guide*.)
'''' '	Type four single quotes, followed by one space, followed by a single quote. There is no space between the end of the Powersoft date format and the four single quotes.

Default value

The default format for date data types is the default ORACLE format.

Example

Assume you are using the DataWindow painter to update an ORACLE table named Employee by setting the Startdate column to 1994-10-21. This date is represented by the following Powersoft date format:

```
yyyy-mm-dd
```

To specify that PowerBuilder should use this format for the date data type when it builds the SQL UPDATE statement, use the following Date DBParm statement:

```
Date = ' ''''yyyy-mm-dd'''' '
```

As a result, PowerBuilder builds the following SQL UPDATE statement to update the table:

```
UPDATE EMPLOYEE
SET STARTDATE = '1994-10-21'
```

See also

DateTime
Time

DateTime

Description

When you update a data source in the DataWindow painter or Form painter, PowerBuilder or InfoMaker builds a SQL UPDATE statement in the background. If your data source is an ORACLE table, the DateTime DBParm parameter determines how PowerBuilder or InfoMaker specifies a DateTime data type when it builds the SQL UPDATE statement. (A DateTime data type contains both a date value and a time value.)

Applies to

ORACLE

Syntax

DateTime = ' ""*Powersoft_DateTime_format*"" '

> **Typing single quotes**
> When PowerBuilder or InfoMaker parses the DBParm DateTime string, it converts each set of four consecutive single quotes you type to one single quote. Therefore, you *must* type the syntax exactly as described in the following table.

Parameter	Description
' ""	Type a single quote, followed by one space, followed by four single quotes. There is no space between the four single quotes and the beginning of the Powersoft DateTime format.
Powersoft_DateTime_format	The DateTime format you want PowerBuilder or InfoMaker to use when it builds a SQL UPDATE statement to update a data source in the DataWindow or Form painters. (For more on Powersoft display formats, see the *User's Guide*.)
"" '	Type four single quotes, followed by one space, followed by a single quote. There is no space between the end of the Powersoft DateTime format and the four single quotes.

Default value

The default display format for DateTime data types is the default ORACLE format.

Example

Assume you are using the DataWindow painter to update an ORACLE table named Files by setting the Timestamp column to 5/2/94 3:45 pm. This DateTime is represented by the following Powersoft DateTime format:

```
m/d/yy h:mm am/pm
```

To specify that PowerBuilder should use this format for the DateTime data type when it builds the SQL UPDATE statement, use the following DateTime DBParm statement:

```
DateTime = ' ''''m/d/yy h:mm am/pm'''' '
```

As a result, PowerBuilder builds the following SQL UPDATE statement to update the table:

```
UPDATE FILES
SET TIMESTAMP = '5/2/94 3:45 pm'
```

See also

Date
Time

DBAdm

Description

Specifies whether you have database administrator (DBAdm) authority to access every table in the IBM database you connect to.

If you set DBAdm to 'Yes' and you *do* have database administrator authority, all tables in the database appear in the Select Tables lists in PowerBuilder or InfoMaker.

Applies to

IBM DRDA

Syntax

DBAdm = '*value*'

Parameter	Description
value	Specifies whether you have database administrator (DBAdm) authority to access every table in your IBM database. Values are: ♦ **Yes** You *have* DBAdm authority. ♦ **No** (Default) You *do not have* DBAdm authority.

Default value

```
DBAdm = 'No'
```

Example

This statement specifies that you have database administrator authority to access every table in the IBM database to which you are connected:

```
DBAdm = 'Yes'
```

DBGetTime

Description

When you set the Async parameter to 1 to enable asynchronous operation, you can also set the DBGetTime parameter for those DBMSs that support it. DBGetTime specifies the number of seconds that you want PowerBuilder or InfoMaker to wait for a response from the DBMS when you retrieve rows in a DataWindow object, query, report, or form.

If DBGetTime is set to 0 (the default), PowerBuilder or InfoMaker waits indefinitely for a DBMS response. In other words, the request never times out. If the DBGetTime value expires before the first row is retrieved, your request is automatically canceled.

Applies to

ODBC
SQL Server
Sybase SQL Server System 10

Syntax

DBGetTime = *value*

Parameter	Description
value	The number of seconds you want PowerBuilder or InfoMaker to wait for a DBMS response while waiting to retrieve the first row of a DataWindow object, query, report, or form.

Default value

```
DBGetTime = 0
```

Usage

To use the DBGetTime parameter, you must do both of the following:

- Set the Async parameter to 1 to enable asynchronous operation. You set both the Async and DBGetTime parameters in the same DBParm statement, as shown in the Example below.
- Code a RetrieveRow event for your DataWindow object.

Example

This statement enables asynchronous operation and sets the DBGetTime parameter to 20 seconds:

```
Async = 1, DBGetTime = 20
```

See also

Async

DBTextLimit

Description

When connecting to a SQL Server database, DBTextLimit specifies the maximum length of a text field that DB-Library will return when you include the text field in a SQL SELECT statement.

You can set the DBTextLimit parameter if you want to include a long text string in a DataWindow object or report without treating the text as a Binary Large Object (BLOB) data type.

Applies to

SQL Server

Syntax

DBTextLimit = '*value*'

Parameter	Description
value	The maximum length in bytes of a text field that DB-Library will return when you include the text field in a SQL SELECT statement. The range of valid values is from 0 bytes to 32,763 bytes. When you set DBTextLimit to 0, DB-Library returns the maximum length text field.

Default value

The default value for DBTextLimit is the default specified by SQL Server for the DBTEXTLIMIT DB-Library option. (For information, see your SQL Server documentation.)

Usage

The text field length that DB-Library will return is the *lesser* of the DBTextLimit value and the setting for the SQL Server global variable @@textsize.

If the setting for @@textsize is less than the value you specify for DBTextLimit, DB-Library will return the @@textsize value.

Example

This statement specifies that DB-Library will return a text field that is up to 32,000 bytes long when you include the text field in a SQL SELECT statement:

```
DBTextLimit = '32000'
```

DelimitIdentifier

Description

Specifies whether you want PowerBuilder or InfoMaker to enclose the names of tables, columns, indexes, and constraints in double quotes when it generates SQL statements.

Applies to

IBM DRDA
Micro Decisionware Database Gateway Interface for DB2
ODBC
ORACLE
Sybase Net-Gateway Interface fore DB2
Sybase SQL Server System 10

Syntax

DelimitIdentifier = '*value*'

Parameter	Description
value	Specifies whether you want PowerBuilder or InfoMaker to enclose table and column names in double quotes. Values are: ♦ **Yes** *Enclose* table and column names in double quotes. ♦ **No** *Do not enclose* table and column names in double quotes.

Default value

The default value for the DelimitIdentifier parameter depends on the DBMS you are accessing, as summarized in the following table:

DBMS	DelimitIdentifer default value
IBM DRDA	`DelimitIdentifier = 'Yes'`
Micro Decisionware Database Gateway Interface for DB2	`DelimitIdentifier = 'No'`
ODBC	Depends on the DelimitIdentifer setting in the PBODB040.INI file
ORACLE	`DelimitIdentifier = 'Yes'`
Sybase Net-Gateway Interface for DB2	`DelimitIdentifier = 'No'`
Sybase SQL Server System 10	`DelimitIdentifier = 'No'`

Usage in IBM databases

By default, PowerBuilder or InfoMaker encloses IBM identifiers in double quotes in order to preserve case sensitivity. However, at many DB2/MVS sites, the established convention is to use only uppercase identifier names. At these sites, specifying DelimitIdentifer = 'No' is recommended. The DB2 family of databases automatically converts identifiers to uppercase if they are not enclosed in double quotes.

Example

This statement specifies that PowerBuilder or InfoMaker should not enclose table and column names in double quotes when it generates SQL statements:

```
Delimitidentifier = 'No'
```

DisableBind

Description

For those DBMSs that support bind variables, PowerBuilder or InfoMaker binds input parameters to a compiled SQL statement by default. The DisableBind parameter allows you to specify whether you want to disable this default binding.

When you set DisableBind to 1 to disable the binding, PowerBuilder or InfoMaker replaces the input variable with the value entered by the end user or specified in a script.

Applies to

INFORMIX
ODBC
ORACLE
SQLBase

Syntax

DisableBind = *value*

Parameter	Description
value	Specifies whether you want to disable the default binding of input parameters to a compiled SQL statement. Values are: ♦ **0** (Default) PowerBuilder or InfoMaker binds input parameters to a compiled SQL statement. This value *enables the default binding.* ♦ **1** PowerBuilder or InfoMaker does *not* bind input parameters to a compiled SQL statement. This value *disables the default binding.*

Default value

```
DisableBind = 0
```

Usage

In a SQL statement, a **bind variable** is a placeholder for a column value. By default, PowerBuilder or InfoMaker associates (binds) data from a variable defined in your application to the bind variable each time the SQL statement executes.

Using bind variables in SQL statements For example, the following SQL statement retrieves those rows in the Books table about books written by Hemingway:

```
SELECT * FROM BOOKS WHERE AUTHOR = HEMINGWAY
```

Suppose that you want to execute this statement to get information about books written by other authors. Instead of compiling and executing a new statement for each author, you can define a bind variable that represents the author's name. The user then supplies the author's actual name when the application executes. By using bind variables, the statement is compiled only once and executed repeatedly with new values supplied by the user.

Here is the same SELECT statement using a bind variable named :Author instead of the author's name. To indicate a bind variable in a SQL statement, precede the variable name with a colon (:).

```
SELECT * FROM BOOKS WHERE AUTHOR = :AUTHOR
```

Bind variables and cached statements Using bind variables in conjunction with cached statements can improve the performance of most applications. The amount of improvement depends on the application. In general, applications that perform a large amount of transaction processing benefit the most from using bind variables and cached statements.

In order to use cached statements, make sure that DisableBind is set to 0. This enables the default binding of input variables to SQL statements in PowerBuilder or InfoMaker. (For more about using cached statements, see the description of the SQLCache parameter on page 359.)

Performance improvements For Watcom SQL and ORACLE databases, bind variables improve performance by allowing PowerBuilder or InfoMaker to insert and modify strings that exceed 255 characters.

Example

This statement specifies that PowerBuilder or InfoMaker should disable the default binding of input parameters to a compiled SQL statement:

```
DisableBind = 1
```

See also

SQLCache

GroupID

Description

At some sites, administrators grant permission to access database tables to groups of users instead of or in addition to individual users. The GroupID DBParm allows you to specify an additional group authorization ID that gives a group of users permission to access tables in the database.

If you specify a group ID, tables accessible to this group as well as to individuals appear in the Select Tables lists in PowerBuilder or InfoMaker.

Applies to

IBM DRDA
Micro Decisionware Database Gateway Interface for DB2
Sybase Net-Gateway Interface for DB2

Syntax

GroupID = '*group_authorization_ID*'

Parameter	Description
group_authorization_ID	The ID that grants authority to a group of users to access tables in your database

Default value

There is no default value for the GroupID DBParm parameter.

Examples

Example 1 This statement sets the GroupID to PB1. Tables accessible to this group will appear in the Select Tables list in PowerBuilder or InfoMaker.

```
GroupID = 'PB1'
```

Example 2 You can set the GroupID, PBCatalogOwner, and System Owner parameters in a single DBParm statement. This statement sets the group ID to PB3, PowerBuilder catalog owner to POWERBLD, and system owner to SYSIBM. Type the statement on a single line.

```
GroupID = 'PB3',PBCatalogOwner = 'POWERBLD',
   SystemOwner = 'SYSIBM'
```

See also

PBCatalogOwner
SystemOwner

Host

Description

Specifies the workstation name when connecting to a SQL Server or Sybase SQL Server System 10 database in PowerBuilder or InfoMaker.

When you specify a value for Host, PowerBuilder or InfoMaker sets the SQL Server CS_HOSTNAME connection property to the workstation name you specify.

> **When to specify Host**
> You must specify the Host DBParm parameter *before* connecting to a SQL Server or Sybase SQL Server System 10 database in PowerBuilder or InfoMaker.

Applies to

SQL Server
Sybase SQL Server System 10

Syntax

Host = '*workstation_name*'

Parameter	Description
workstation_name	The name of the workstation in a SQL Server or Sybase SQL Server System 10 connection

Default value

There is no default value for the Host DBParm parameter.

Usage

The value you specify for the Host parameter appears in the Hostname column of the MASTER.DBO.SYSPROCESSES table in a SQL Server database. How you use the Host parameter depends on the design of your PowerBuilder or InfoMaker application.

For example, many sites want to secure their production tables so that updates are possible only through a specific application. To do this, you can grant explicit authority to the PowerBuilder or InfoMaker application but *not* to end users.

The application prompts the end user for an authorization ID and password, verifies it, and then connects to the SQL Server database through a single application logon ID. Only this application logon ID has authorization to update production tables.

In this scenario, you could use the Host DBParm parameter to store the name of the end user running the application.

Examples

Example 1 This statement sets the host name to Alan:

```
Host = 'Alan'
```

Example 2 This statement sets the host name to Jane and the application name to Sales:

```
Host = 'Jane',AppName = 'Sales'
```

See also

AppName

HPConnect

Description

Specifies the parameters required to connect to an ALLBASE/SQL database. You must supply a valid HPConnect string in the Database Profile Setup dialog box when defining the ALLBASE/SQL Powersoft database interface. PowerBuilder and InfoMaker use the parameters in the HPConnect string to connect to an ALLBASE/SQL database.

> **When to specify HPConnect**
> Specify the HPConnect string in your database profile when you define the Powersoft ALLBASE/SQL database interface. You must do this *before* connecting to the database in PowerBuilder or InfoMaker.
>
> For instructions, see "Defining the database interface" in the ALLBASE/SQL section of Chapter 3, "Using Powersoft Database Interfaces."

Applies to

ALLBASE/SQL

Syntax

HPConnect = ' *connectstring* '

Parameter	Description
connectstring	A string that conforms to the Hewlett-Packard MPE/iX or HP-UX syntax. For instructions, see "Defining the database interface" in the ALLBASE/SQL section of Chapter 3, "Using Powersoft Database Interfaces."

Default value

There is no default value for the HPConnect parameter.

Usage

You can change the HPConnect string if necessary by editing it in the DBParm field of the Database Profile Setup dialog box.

Examples

Example 1 This example shows an HPConnect string that uses the MPE/iX syntax. Type the HPConnect string on a single line:

```
HPConnect = ' #mpeix/mpesysid:dbe1.pub.acct#user1/
   userpass.acct/acctpass '
```

Example 2 This example shows an HPConnect string that uses the HP-UX syntax:

```
HPConnect = ' #hpux/mpesysid:/path/dbe/#usrid:pass '
```

INET_DBPATH

Description

Specifies the INFORMIX DBPATH setting. The DBPATH environment variable identifies a list of directories that contain INFORMIX databases.

Applies to

INFORMIX (IN5 interface connected to INFORMIX-SE 5.x or INFORMIX-SE 6.x only)

Syntax

INET_DBPATH = ' *server_db_path* '

Parameter	Description
server_db_path	The name of the directory that contains INFORMIX databases.

Default value

By default, PowerBuilder or InfoMaker uses the value specified for DBPATH in the INFORMIX.INI configuration file.

Usage

You use the INET_DBPATH parameter only when you connect to an INFORMIX-SE Version 5.x or 6.x database through the Powersoft IN5 interface. The INET_PROTOCOL parameter does *not* apply to the IN5 interface connected to an INFORMIX-OnLine Version 5.x or 6.x database, or to the IN4 interface.

You can specify values for INET_DBPATH, INET_PROTOCOL, and INET_SERVICE in the same DBParm statement.

Examples

Example 1 This statement specifies that the directory /HOME/INFORMIX contains INFORMIX databases:

```
INET_DBPATH = '/home/informix'
```

Example 2 This statement specifies that the directory /INFORMIX contains INFORMIX databases, and that you want to connect using the SE5 service and the TCP/IP network protocol. Enter the statement on a single line.

```
INET_DBPATH = '/informix',INET_SERVICE = 'se5',
   INET_PROTOCOL = 'tcp-ip'
```

See also

INET_PROTOCOL
INET_SERVICE

INET_PROTOCOL

Description

Specifies the network protocol that the INFORMIX-NET Version 5.x client software uses to communicate with a remote INFORMIX database server.

Applies to

INFORMIX (IN5 interface only)

Syntax

INET_PROTOCOL = ' *network_protocol* '

Parameter	Description
network_protocol	A null-terminated string that specifies the name of the network protocol used by INFORMIX-NET Version 5.x. For information about the correct network protocol for your site, see your INFORMIX system administrator.

Default value

By default, PowerBuilder or InfoMaker uses the network protocol specified in the INFORMIX.INI configuration file.

Usage

You use the INET_PROTOCOL parameter only when you connect to an INFORMIX database through the Powersoft IN5 interface. The INET_PROTOCOL parameter does *not* apply to the IN4 interface.

You can specify values for INET_DBPATH, INET_PROTOCOL, and INET_SERVICE in the same DBParm statement.

Examples

Example 1 This statement specifies that INFORMIX-NET Version 5.x uses the Novell IPX/SPX network protocol:

```
INET_PROTOCOL = 'ipx'
```

Example 2 This statement specifies that the directory /INFORMIX contains INFORMIX databases, and that you want to connect using the SE5 service and the TCP/IP network protocol. Enter the statement on a single line.

```
INET_DBPATH = '/informix',INET_SERVICE = 'se5',
   INET_PROTOCOL = 'tcp-ip'
```

See also

INET_DBPATH
INET_SERVICE

INET_SERVICE

Description

Specifies the name of the service that a remote INFORMIX database server uses to listen to all incoming requests from client applications.

Applies to

INFORMIX (IN5 interface only)

Syntax

INET_SERVICE = ' *service_name* '

Parameter	Description
service_name	A null-terminated string that specifies the name of the service that a remote INFORMIX database server uses to listen to incoming requests. For information about the correct service name for your site, see your INFORMIX system administrator.

Default value

By default, PowerBuilder or InfoMaker uses the service name specified in the INFORMIX.INI configuration file.

Usage

You use the INET_SERVICE parameter only when you connect to an INFORMIX database through the Powersoft IN5 interface. The INET_SERVICE parameter does *not* apply to the IN4 interface.

You can specify values for INET_DBPATH, INET_PROTOCOL, and INET_SERVICE in the same DBParm statement.

Examples

Example 1 This statement specifies that INFORMIX-NET Version 5.x uses the sqlexec service name:

```
INET_SERVICE = 'sqlexec'
```

Example 2 This statement specifies that the directory /INFORMIX contains INFORMIX databases, and that you want to connect using the SE5 service and the TCP/IP network protocol. Enter the statement on a single line.

```
INET_DBPATH = '/informix',INET_SERVICE = 'se5',
   INET_PROTOCOL = 'tcp-ip'
```

See also

INET_DBPATH
INET_PROTOCOL

Language

Description

Specifies the language you want to use when connecting to a Sybase SQL Server System 10 database in PowerBuilder or InfoMaker.

When you specify a value for Language, PowerBuilder or InfoMaker does the following:

- Allocates a CS_LOCALE structure for this connection
- Sets the CS_SYB_LANG value to the language you specify
- Sets the SQL Server CS_LOC_PROP connection property with the new locale information

> **When to specify Language**
> You must specify the Language DBParm parameter *before* connecting to a Sybase SQL Server System 10 database in PowerBuilder or InfoMaker.

Applies to

Sybase SQL Server System 10

Syntax

Language = *'language_name'*

Default value

There is no default value for the Language DBParm parameter.

Example

This statement allocates a CS_LOCALE structure for the Sybase System 10 connection and sets the CS_SYB_LANG value to the French language:

```
Language = 'French'
```

Log

Description

Specifies whether the database server should log updates of text and image data in the SQL Server transaction log.

By default, the database server logs updates of image and text data in the SQL Server transaction log.

Applies to

SQL Server
Sybase SQL Server System 10

Syntax

Log = *value*

Parameter	Description
value	A value that specifies whether the database server should log updates of text and image data in the SQL Server transaction log. Values are: ♦ **0** *Do not log text and image updates* in the SQL Server transaction log. Specify this value only if your database server allows you to disable logging. ♦ **1** (Default) *Log text and image updates* in the SQL Server transaction log.

Default value

```
Log = 1
```

Usage

You should set the Log parameter to 0 only if your database server allows you to disable logging.

Example

This statement specifies that PowerBuilder or InfoMaker should *not* log text and image updates in the SQL Server transaction log:

```
Log = 0
```

LoginTimeOut

Description

Specifies the number of seconds the ODBC driver should wait for a login request to an ODBC data source. If you set LoginTimeOut to 0, PowerBuilder or InfoMaker waits the number of seconds specified by the ODBC driver's client software.

By default, the ODBC driver waits 15 seconds for a login request.

Applies to

ODBC

Syntax

LoginTimeOut = *value*

Parameter	Description
value	The number of seconds you want the ODBC driver to wait for an ODBC login request. To wait for the number of seconds specified by the ODBC driver's client software, set the value to 0.

Default value

```
LoginTimeOut = 15
```

Example

This statement sets the LoginTimeOut value to wait 60 seconds for an ODBC login request:

```
LoginTimeOut = 60
```

MixedCase

Description

Specifies whether you want connections to an ORACLE database to be case sensitive or case insensitive.

By default, MixedCase is set to 0. This setting specifies a case insensitive connection, and assumes that all identifiers are uppercase. To make the ORACLE connection case sensitive, set the MixedCase parameter to 1.

Applies to

ORACLE

Syntax

MixedCase = *value*

Parameter	Description
value	Specifies whether an ORACLE database connection is case sensitive or case insensitive. Values are: ♦ **0** (Default) The ORACLE database connection is *case insensitive*. It assumes that all identifiers are uppercase. ♦ **1** The ORACLE database connection is *case sensitive*. It supports mixed case, uppercase, and lowercase identifiers.

Default value

```
MixedCase = 0
```

Usage

When you set the MixedCase parameter to 1 and define a primary key for a table in an ORACLE database, *all* of the following must contain only uppercase letters:

- The name of the primary key
- The name of the table containing the primary key
- The names of any foreign keys that reference the primary key in this table

Example

This statement makes an ORACLE database connection case sensitive:

```
MixedCase = 1
```

MsgTerse

Description

Specifies whether PowerBuilder or InfoMaker should display terse error messages for ODBC data source drivers. A terse ODBC error message is one without the SQLSTATE = *nnnn* prefix, where *nnnn* is the number of the error message.

By default, PowerBuilder or InfoMaker displays ODBC error messages with the SQLSTATE prefix. To display ODBC error messages *without* the SQLSTATE prefix, set MsgTerse to 'Yes'.

Applies to

ODBC

Syntax

MsgTerse = *'value'*

Parameter	Description
value	Specifies whether PowerBuilder or InfoMaker should display ODBC error messages without the SQLSTATE prefix. Values are: ♦ **Yes** PowerBuilder or InfoMaker displays ODBC error messages *without* the SQLSTATE prefix. ♦ **No** (Default) PowerBuilder or InfoMaker displays ODBC error messages *with* the SQLSTATE prefix.

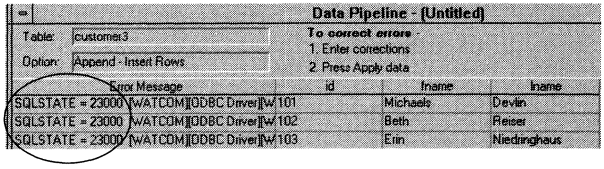
Default value

```
MsgTerse = 'No'
```

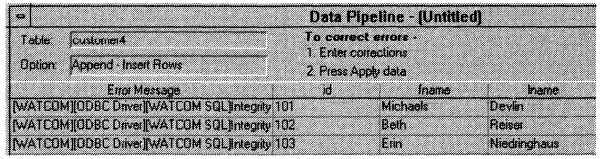
Usage

You can set the MsgTerse DBParm parameter to 'Yes' to display shorter ODBC error messages in PowerBuilder or InfoMaker. This may be useful if space on your screen is limited.

For example, suppose you are using the Data Pipeline in PowerBuilder or InfoMaker to pipe data to a Watcom SQL ODBC database, and errors occur while you are executing the pipeline. If MsgTerse is set to 'No' (the default value), pipeline errors display in an Error dialog box *with* the SQLSTATE prefix, like this:



If you specify MsgTerse = 'Yes' in the database profile of the Watcom SQL destination database, the Data Pipeline displays terse ODBC error messages *without* the SQLSTATE prefix, like this:



For instructions on using the Data Pipeline, see the *User's Guide*.

Example

This statement specifies that PowerBuilder or InfoMaker should display terse ODBC error messages without the SQLSTATE prefix:

```
MsgTerse = 'Yes'
```

PBCatalogOwner

Description

Specifies a nondefault owner for the tables in the Powersoft repository. The Powersoft repository, also known as the Powersoft catalog or Powersoft system tables, consists of five tables that contain default extended attribute information for your database.

When you specify an owner name that is different from the default repository owner for your DBMS, PowerBuilder or InfoMaker creates a new set of repository tables with the owner name you specify.

Applies to

IBM DRDA
Micro Decisionware Database Gateway Interface for DB2
ODBC
Sybase Net-Gateway Interface for DB2
XDB

Syntax

PBCatalogOwner = '*owner_name*'

Parameter	Description
owner_name	Specifies the owner of the tables in the Powersoft repository. If you use the DB2SYSPB.SQL script to create the repository in a DB2 database and you replace all instances of pbcatown in the script with the name of a nondefault table owner, *owner_name* must be the same as the owner specified in the DB2SYSPB.SQL script.

Default value

The default value for PBCatalogOwner depends on the DBMS you are accessing, as summarized in the following table:

DBMS	PBCatalogOwner default value
IBM DRDA	`PBCatalogOwner = 'PBCATOWN'`
Micro Decisionware Database Gateway Interface for DB2	If you do not specify a value for PBCatalogOwner, PowerBuilder or InfoMaker prompts you to supply it.
ODBC	If you do not specify a value for PBCatalogOwner in the database profile or in the PBODB040.INI file, the default value is the user ID specified in the database profile.
Sybase Net-Gateway Interface for DB2	If you do not specify a value for PBCatalogOwner, PowerBuilder or InfoMaker prompts you to supply it.
XDB	There is no default value for PBCatalogOwner.

Usage

By specifying a nondefault owner for the Powersoft repository tables, you are in effect creating an alternative repository. This is useful if you want to test new validation rules or display formats without overwriting the extended attributes currently in the default repository.

ODBC data sources When you connect to an ODBC data source and a value for PBCatalogOwner is set in both the database profile and the PBODB040.INI file, the setting in the profile overrides the setting in the PBODB040.INI file.

DB2 databases When you connect to a DB2 database, you can use the DB2SYSPB.SQL script to create the Powersoft repository (system tables). If you use the DB2SYSPB.SQL script, you can do any of the following:

- Edit the script to change all instances of pbcatown to another name. (You can also leave the table owner as pbcatown in the script, which is the default.)

> **Specifying SYSIBM is prohibited**
> You cannot specify SYSIBM as the table owner. This is prohibited by DB2.

- If you changed pbcatown to another name in the DB2SYSPB.SQL script, set the PBCatalogOwner parameter to the owner you specified in this script.

Example

This statement creates a new set of Powersoft repository tables with the owner TEST. The names of the new tables have the prefix TEST, such as TEST.pbcatcol, TEST.pbcatedt, and so on.

```
PBCatalogOwner = 'TEST'
```

See also

For instructions on using the DB2SYSPB.SQL script, see "Creating Powersoft system tables in DB2 databases" in Chapter 3, "Using Powersoft Database Interfaces."

For more about the Powersoft repository, see "Creating the Powersoft repository" in Chapter 4, "Managing Database Connections."

PBDBMS

Description

Setting the PBDBMS parameter to 1 enables you to create a DataWindow object or report that uses an ORACLE 7 PBDBMS stored procedure as its data source. You must create a special kind of ORACLE 7 stored procedure that uses PBDBMS.Put_Line function calls to build the SQL SELECT statement. (For more information, see the Usage section below.)

Applies to

ORACLE Version 7 (OR7 interface only)

Syntax

PBDBMS = *value*

Parameter	Description
value	Specifies whether you can use an ORACLE 7 PBDBMS stored procedure as a data source for DataWindow objects and reports. Values are: ♦ **0** (Default) You *cannot use* an ORACLE 7 PBDBMS stored procedure as a data source. ♦ **1** You *can use* an ORACLE 7 PBDBMS stored procedure as a data source.

Default value

```
PBDBMS = 0
```

Usage

Using an ORACLE 7 PBDBMS stored procedure as a data source for a DataWindow object or report involves two general steps:

1 From PowerBuilder or InfoMaker, set up your ORACLE 7 database server.

2 Create DataWindow objects and reports that use the stored procedure.

For instructions, see "Using ORACLE 7 stored procedures as a data source" in the ORACLE section of Chapter 3, "Using Powersoft Database Interfaces."

Example

This statement specifies that you can use an ORACLE 7 PBDBMS stored procedure as a data source for DataWindow objects and reports:

```
PBDBMS = 1
```

Recovery

Description

Specifies whether the SQLBase transaction log file is created when you connect to a SQLBase database in PowerBuilder or InfoMaker. The transaction log contains information about the sequence of events that occur while running SQLBase. The data in the transaction log file is used to perform rollbacks, crash recovery, and media recovery.

By default, the SQLBase transaction log file is created.

Applies to

SQLBase

Syntax

Recovery = *value*

Parameter	Description
value	Specifies whether the SQLBase transaction log file is created. Values are: ♦ **0** The SQLBase transaction log file *is not created.* You *cannot* roll back database changes, and in the event of a database crash you *cannot* recover your data. ♦ **1** (Default) The SQLBase transaction log file *is created.*

Default value

```
Recovery = 1
```

Usage

All users connected to a SQLBase database *must* have the same setting for the Recovery parameter. The Recovery setting of the first user to connect to a SQLBase database determines the Recovery setting required for all other users.

In other words, if the first user to connect to a SQLBase database sets the Recovery parameter to 1, only users with Recovery set to 1 are able to connect to the database. If the first user's Recovery setting is 0, then all other users must set the Recovery parameter to 0.

> ***Caution***
> *When you set the Recovery parameter to 0, you cannot roll back database changes, and if there is a database crash you cannot recover your data.*

Example

This statement specifies that the SQLBase transaction log file should not be created:

```
Recovery = 0
```

Release (SQL Server)

Description

You can set the Release DBParm parameter to '4.2' when connecting to a SQL Server Version 4.2 (or higher) database in PowerBuilder or InfoMaker. This specifies that you want to use SQL Server DB-Library cursor processing instead of the default PowerBuilder or InfoMaker cursor processing.

Applies to

SQL Server

Syntax

Release = '4.2'

> **You must specify '4.2'**
> When you set the Release DBParm parameter for SQL Server, '4.2' is the only valid value. No other values (such as '4.9' or '10.0') are permitted.

Default value

By default, the Release parameter is not set, and PowerBuilder or InfoMaker uses its own default cursor processing.

Usage

You *must* set the Release parameter to '4.2' in any of the following situations:

- You are using the CursorLock and CursorScroll DBParm parameters with a SQL Server database.
- You want to use features of SQL Server DB-Library cursor processing, such as the ability to create scroll cursors.
- You want to use SQL UNION statements.
- You want to use any of the following SQL Server data types:
  - Real
  - SmallDateTime
  - SmallMoney

In general, SQL Server cursor processing is slower than PowerBuilder's or InfoMaker's default cursor processing. Therefore, using SQL Server cursor processing may slow the performance of your application.

Examples

Example 1 This statement specifies that you want to use SQL Server DB-Library cursor processing when connecting to a SQL Server Version 4.2 (or higher) database:

```
Release = '4.2'
```

Example 2 This statement sets Dynamic scrolling and OptVal locking options for cursors in a SQL Server database that is Version 4.2 or higher. Type the statement on a single line.

```
Release='4.2',CursorScroll='Dynamic',
   CursorLock='OptVal'
```

See also

CursorLock (SQL Server)
CursorScroll (SQL Server)

Release (XDB)

Description

Specifies that you want to connect to an XDB Version 3.0 database in PowerBuilder or InfoMaker.

By default, the Release parameter is set to '2.41' to connect to an XDB Version 2.41 database.

When to specify Release for an XDB database
You must specify Release = '3.00' in the DBParm field of the Database Profile Setup dialog box *before* connecting to an XDB Version 3.0 database in PowerBuilder or InfoMaker.

Applies to

XDB

Syntax

Release = '3.00'

Default value

```
Release = '2.41'
```

Example

This statement specifies connection to an XDB Version 3.0 database in PowerBuilder or InfoMaker:

```
Release = '3.00'
```

Request

Description

When you connect to an IBM DB2 database through the Powersoft Micro Decisionware Database Gateway Interface for DB2, the Request parameter specifies whether you want to release IBM Customer Information Control System (CICS) transaction resources after each request. To do so, set the value of Request to 1.

By default, Request is set to 0, which maintains the CICS resources for the duration of the request.

Requirements for using the Request parameter
Setting the Request parameter to 1 to release CICS resources has an effect only when you do *both* of the following:

- Set the AutoCommit database preference to 'True' to turn off normal recoverable transaction processing. (See the description of AutoCommit on page 381.)
- Specify the value for Request *before* connecting to a DB2 database through the Powersoft Micro Decisionware Database Gateway Interface for DB2.

Applies to

Micro Decisionware Database Gateway Interface for DB2

Syntax

Request = *value*

Parameter	Description
value	Specifies whether you want to release CICS transaction resources after each request. Values are: ♦ **0** (Default) Maintain CICS resources for the duration of the request. ♦ **1** Release CICS resources after each request.

Default value

```
Request = 0
```

Usage

When you set the Request parameter to 1, a CICS transaction is started for each request. This may slow the performance of your application.

Example

This statement specifies that you want to release CICS transaction resources after each request:

```
Request = 1
```

See also

AutoCommit

Scroll

Description

Specifies whether you want to use a scroll cursor when connecting to an INFORMIX database in PowerBuilder or InfoMaker. When you fetch rows in an INFORMIX table, using a **scroll cursor** enables you to fetch the next row, previous row, first row, or last row.

By default, PowerBuilder or InfoMaker does not use scroll cursors in an INFORMIX database connection.

Applies to

INFORMIX

Syntax

Scroll = *value*

Parameter	Description
value	Specifies whether you want to use a scroll cursor when connecting to an INFORMIX database in PowerBuilder or InfoMaker. Values are: ♦ **0** (Default) *Do not use* a scroll cursor. ♦ **1** *Use* a scroll cursor.

Default value

```
Scroll = 0
```

Example

This statement specifies that you want to use a scroll cursor when connecting to an INFORMIX database in PowerBuilder or InfoMaker:

```
Scroll = 1
```

SQLCache

Description

The SQLCache parameter lets you specify the number of SQL statements that PowerBuilder or InfoMaker should cache. By default, SQLCache is set to 0, indicating an empty SQL cache. PowerBuilder or InfoMaker caches:

- SQL statements generated by a DataWindow object or report
- Embedded SQL statements

Statements in the SQL cache are maintained on a least-recently-used (**LRU**) basis. In other words, if a statement must be removed from the cache to make room for another statement, PowerBuilder or InfoMaker removes the statement that was least recently executed.

PowerBuilder or InfoMaker caches SQL statements for ORACLE databases and for ODBC if the backend database supports it.

Applies to

ODBC
ORACLE

Syntax

The syntax for setting the SQLCache parameter depends on whether you are accessing an ODBC data source or an ORACLE database.

ODBC syntax Use the following syntax to set the SQLCache parameter for an ODBC data source:

SQLCache = *value*

Parameter	Description
value	The number of cursors you want to open in a script, plus the number of DataWindow-generated SELECT statements with retrieval arguments. The default value is 0.

ORACLE syntax Use the following syntax to set the SQLCache parameter for an ORACLE 6 or ORACLE 7 database:

SQLCache = *value*

where *value* is a number that is less than the maximum number of cursors that can be open on the client machine. To determine this number, use the following formula:

value < = *open_cursors* – *reserved* – *declare_cursor_space*

Parameter	Description
open_cursors	The ORACLE server setting (OPEN_CURSORS) that specifies the maximum number of cursors a single process can have open at one time. Values are: ♦ *For ORACLE 6* A number between 5 and 255. The default value is 50. ♦ *For ORACLE 7* A number between 1 and the operating system limit. The default value is 50.
reserved	The number of reserved cursors. You should reserve five cursors for use by the Powersoft ORACLE database interface.
declare_cursor_space	The maximum number of cursors you expect to open per ORACLE connection in PowerBuilder or InfoMaker.

Default value

```
SQLCache = 0
```

Usage

Caching SQL statements that you execute frequently will improve their performance. Statements with bind variables are often the most frequently used. In fact, if your DBMS does not support bind variables, caching statements is of limited value.

> **Setting DisableBind to use cached statements**
> In order to use cached statements, make sure the DisableBind DBParm parameter is set to 0 (the default). This enables the binding of input variables to SQL statements in PowerBuilder or InfoMaker.
>
> For more about using bind variables, see the description of the DisableBind parameter on page 338.

The first time you execute a SQL statement containing bind variables, PowerBuilder or InfoMaker does the following:

1 Parses the statement.

2 For SQL SELECT statements, calls the appropriate database function to get a description of the result set.

3 Allocates memory buffers for the bind variables.

4 Binds the allocated memory buffers to the parsed statement.

When you cache this SQL statement, PowerBuilder or InfoMaker stores the parsed statement, result set description, and memory buffer allocation and binding in the SQL cache. The next time you execute this statement, PowerBuilder or InfoMaker finds it in the cache and avoids the overhead of repeating these steps.

If PowerBuilder or InfoMaker finds an exact match for this statement in the SQL cache, it simply copies the new values supplied for the bind variables to the pre allocated memory buffers and executes the statement. This is much faster than having to process the statement from scratch.

For PowerBuilder developers
To help you determine an appropriate size for your SQL cache, you can check the value of the SQLReturnData attribute of the transaction object. When you disconnect from the database, the number of hits, misses, and entries in the SQL cache is stored in SQLReturnData, as follows:

- **Hits** is the number of times PowerBuilder found a matching statement in the SQL cache.
- **Misses** is the number of times PowerBuilder did *not* find a matching statement in the cache.
- **Entries** is the total number of statements in the SQL cache, which is determined by your SQLCache setting.

Example

This statement sets the SQL cache size to 25 statements:

```
SQLCache = 25
```

See also

DisableBind

SQLQualifiers

Description

Sets the level of qualification for identifiers (table and column names) in SQL statements when you connect to a DB2 database through the Powersoft Sybase Net-Gateway Interface for DB2.

When PowerBuilder or InfoMaker **qualifies** a table or column name, it prefixes it with the name of the owner. For example, if a table named Sales is owned by Fran, the qualified table name is Fran.Sales.

If the name of the table owner is the same as the person logged on to the DB2 database, PowerBuilder or InfoMaker does *not* qualify identifiers with owner names in the SQL statements it generates. If you set the SQLQualifiers parameter to 1, PowerBuilder or InfoMaker qualifies identifiers with an owner name in SQL statements.

You can set the SQLQualifiers parameter if *both* of the following are true:

- DB2 security is off.
- You assume that unqualified tables are owned by the default DB2 table owner.

Applies to

Sybase Net-Gateway Interface for DB2

Syntax

SQLQualifiers = *value*

Parameter	Description
value	Sets the level of qualification for identifiers in SQL statements when you connect to a DB2 database through the Powersoft Sybase Net-Gateway Interface for DB2. Values are: ♦ **0** (Default) *Do not qualify* identifiers with owner names in SQL statements. ♦ **1** *Qualify* identifiers with owner names in SQL statements.

Default value

```
SQLQualifiers = 0
```

Example

When you connect to a DB2 database through the Powersoft Sybase Net-Gateway Interface for DB2, this statement specifies that you want PowerBuilder or InfoMaker to qualify identifiers with owner names in SQL statements:

```
SQLQualifiers = 1
```

SystemOwner

Description

Specifies the owner of the IBM DB2 system tables that you want PowerBuilder or InfoMaker to use. PowerBuilder or InfoMaker accesses the DB2 system tables to get information about the tables and columns in your database.

By default, the owner of the DB2 system tables is SYSIBM. When you use the SystemOwner parameter to specify a nondefault system owner, PowerBuilder or InfoMaker uses the set of system tables belonging to this owner instead of the default system tables owned by SYSIBM.

Applies to

IBM DRDA
Micro Decisionware Database Gateway Interface for DB2
Sybase Net-Gateway Interface for DB2
XDB

Syntax

SystemOwner = '*owner_name*'

Parameter	Description
owner_name	Specifies the owner of the DB2 system tables that you want PowerBuilder or InfoMaker to use. The default owner of the DB2 system tables is SYSIBM.

Default value

```
SystemOwner = 'SYSIBM'
```

Usage

If your site has a large DB2 system catalog, it may be useful to create local copies of the catalog tables and populate them with a subset of the information in the default system catalog. (These local copies are sometimes called **shadow catalogs**.)

You can then set the value of SystemOwner to the owner of the shadow catalogs. This tells PowerBuilder or InfoMaker to access the smaller shadow catalogs instead of the larger default system tables, resulting in faster performance. However, you must make sure to keep the shadow catalogs synchronized with the default system catalog owned by SYSIBM.

For more about creating shadow catalogs, see your DB2 system administrator.

Example

This statement specifies MYAPP as the owner of the system tables that you want PowerBuilder or InfoMaker to use:

```
SystemOwner = 'MYAPP'
```

TableCriteria (IBM DRDA, Micro Decisionware Gateway, XDB)

Description

Enables you to specify criteria (search conditions) to limit the list of tables and views that appears in the Select Tables list in PowerBuilder or InfoMaker. Setting this parameter can be useful if you are working with a very large database.

PowerBuilder or InfoMaker appends the search criteria you specify to the WHERE clause of the SQL SELECT statement it generates to create the table list. This SELECT statement has the following format:

```
SELECT column_list FROM SYSTABLES, SYSTABAUTH
   WHERE join_criteria
   AND table_authorizations
   AND tablecriteria_search_conditions
```

Applies to

IBM DRDA
Micro Decisionware Database Gateway Interface for DB2
XDB

Syntax

TableCriteria = '*search_conditions*'

Parameter	Description
search_conditions	Specifies the criteria you want to use to limit the tables and views displayed in the Select Tables list. The *search_conditions* string can be any valid DB2/MVS WHERE clause syntax. Enclose the entire *search_conditions* string in single quotes. For each literal value in the *search_conditions* string, type two consecutive single quotes, followed by the value, followed by two consecutive single quotes. PowerBuilder or InfoMaker interprets these two consecutive single quotes as an escape quote.

Default value

There is no default value for TableCriteria. If you do not specify a value, the TableCriteria parameter is not used.

Examples

The following examples show different ways to set the TableCriteria DBParm parameter to limit the Select Tables list:

> **Typing single quotes**
> The quotes in all of the following examples are single quotes. Do not type double quotes.

Example 1 This statement produces a lists of tables belonging to either the QR or FS systems:

```
TableCriteria='NAME like ''QR%'' OR NAME like ''FS%'' '
```

Example 2 This statement produces a list of tables created by the specified users:

```
TableCriteria='CREATOR in (''FRANS'',''MIKEC'')'
```

Example 3 This statement produces a list of accessible tables created in October 1994:

```
TableCriteria='MONTH(CTIME)=10 AND YEAR(CTIME)=94'
```

TableCriteria (ODBC)

Description

Enables you to specify criteria (search conditions) to limit the list of tables and views that appears in the Select Tables list in PowerBuilder or InfoMaker. Setting this parameter can be useful if you are working with a very large database.

Applies to

ODBC

Syntax

TableCriteria = '{*table_name_criteria*},{*table_owner_criteria*},{*table_qualifier_criteria*}'

Type the TableCriteria string on a single line. The order of parameters within the TableCriteria string is important. Therefore, when you omit one or more leading parameters, you must include a comma to indicate the omission.

The TableCriteria string can include the following metacharacters:

- Percent (%) matches any sequence of zero or more characters.
- Underscore (_) matches any single character.

To use a metacharacter as a literal, precede it with the escape character for the ODBC data source you are accessing. (See Example 3 below.)

Parameter	Description
table_name_criteria	Specifies the names of tables to display.
table_owner_criteria	Displays only those tables belonging to the specified *table_owner*.
table_qualifier_criteria	Specifies the type of table objects to display.You can specify one or more of the following types: ♦ **TABLE** ♦ **VIEW** ♦ **SYSTEM TABLE** ♦ **ALIAS** ♦ **SYNONYM** ♦ **GLOBAL TEMPORARY** ♦ **LOCAL TEMPORARY** ♦ A data source-specific type identifier. (For information, see the documentation for your ODBC data source.)

Default value

There is no default value for TableCriteria. If you do not specify a value, the TableCriteria parameter is not used.

Examples

The following examples show different ways to set the TableCriteria DBParm parameter to limit the Select Tables list:

Example 1 This statement produces a list of tables and views owned by SDG that begin with cust:

```
TableCriteria = 'cust%,SDG'
```

Example 2 This statement produces a list of tables owned by SDG. The comma before SDG indicates that *table_name_criteria* was intentionally omitted.

```
TableCriteria = ',SDG'
```

Example 3 This statement produces a list of tables owned by Dept_Marketing. To use the underscore character in Dept_Marketing as a literal, precede it with the escape character for your ODBC data source. In this example, the escape character is a backslash (\ ).

```
TableCriteria = ',Dept\_Marketing'
```

TableCriteria (ORACLE)

Description

Enables you to specify criteria (search conditions) to limit the list of tables and views that appears in the Select Tables list in PowerBuilder or InfoMaker. Setting this parameter can be useful if you are working with a very large database.

Applies to

ORACLE

Syntax

TableCriteria =
'{*table_name_criteria*},{*table_owner_criteria*},{*table_qualifier_criteria*}'

Type the TableCriteria string on a single line. The order of parameters within the TableCriteria string is important. Therefore, when you omit one or more parameters, you must include a comma to indicate the omission.

The TableCriteria string can include the percent (%) metacharacter that matches any sequence of zero or more characters.

Parameter	Description
table_name_criteria	Specifies the names of tables to display. PowerBuilder or InfoMaker does not validate the value you supply.
table_owner_criteria	Displays only those tables belonging to the specified *table_owner*. PowerBuilder or InfoMaker does not validate the value you supply.
table_qualifier_criteria	Specifies the type of table objects to display.You can specify one or more of the following types, using only uppercase characters: ♦ **"TABLE"** ♦ **"VIEW"** ♦ **"SYNONYM"** For each table type in the *table_qualifier_criteria* string, type two consecutive single quotes, followed by the table type, followed by two consecutive single quotes. Separate the types with commas.

Default value

If you do not specify a value for the TableCriteria DBParm parameter, all ORACLE tables, views, and synonyms that you have permission to access appear in the Select Tables list.

Examples

The following examples show different ways to set the TableCriteria DBParm parameter to limit the Select Tables list:

> **Typing single quotes**
> The quotes in all of the following examples are single quotes. Do not type double quotes.

Example 1 This statement produces a list of all tables and views owned by FMS that begin with emp. The comma after FMS indicates that *table_qualifier_criteria* was intentionally omitted.

```
TableCriteria = 'emp%,FMS,'
```

Example 2 This statement produces a list of tables only. The two commas before TABLE indicate that *table_name_criteria* and *table_owner_criteria* were intentionally omitted.

```
TableCriteria = ',,''TABLE'' '
```

Example 3 This statement produces a list of synonyms for all tables and views. Only the synonyms for the tables and views appear in the list, not the tables and views themselves. Synonyms for tables and views do not appear in the table list by default; you must explicitly specify the type SYNONYM to list them.

```
TableCriteria = ',,''SYNONYM'' '
```

Example 4 This statement produces a list of all views and all synonyms for views.

```
TableCriteria = ',,''VIEW'',''SYNONYM'' '
```

Example 5 This statement produces a list of all views owned by FMS that start with Prod, such as Product_Quantities_View. It also lists all synonyms owned by FMS for views that start with Prod, such as Product_Quantities_View_Syn.

```
TableCriteria = 'Prod%,FMS,''VIEW'',''SYNONYM'' '
```

Example 6 This statement (an empty TableCriteria string) produces a list of all tables and views, but no synonyms. It is equivalent to typing TableCriteria=',,"TABLE","VIEW" '.

```
TableCriteria = ',,'
```

TableCriteria (Sybase Net-Gateway)

Description

Enables you to specify criteria (search conditions) to limit the list of tables and views that appears in the Select Tables list in PowerBuilder or InfoMaker. Setting this parameter can be useful if you are working with a very large database.

The Powersoft Sybase Net-Gateway interface uses the sp_tables stored procedure to create the table list. The sp_tables procedure takes four optional arguments:

sp_tables {*table_name*},{*table_owner*},{*table_qualifier*},{*table_type*}

PowerBuilder or InfoMaker uses the criteria you specify in the TableCriteria parameter to supply the arguments to sp_tables and build the table list based on your criteria.

Applies to

Sybase Net-Gateway Interface for DB2

Syntax

TableCriteria =
'{*table_name*},{*table_owner*},{*table_qualifier*},{"*table_type*"}'

Type the TableCriteria string on a single line. The order of parameters in the TableCriteria string is important. Therefore, when you omit one or more leading parameters, you must include a comma to indicate the omission.

Parameter	Description
table_name	Specifies the names of tables to display. You can use wildcard characters. If you omit this value, PowerBuilder or InfoMaker displays all tables that you have permission to access.
table_owner	Displays only those tables belonging to the specified *table_owner*. You can use wildcard characters. If you omit this value, PowerBuilder or InfoMaker displays all tables matching *table_name* that you have permission to access.

Parameter	Description
table_qualifier	Not applicable for use with PowerBuilder or InfoMaker. Leave this parameter blank.
"*table_type*"	Specifies the type of table objects to display. You must enclose the entire *table_type* string in double quotes. You *cannot* use wildcard characters. You can specify one or more of the following types. Enclose each type in single quotes, and separate the types with commas. ♦ **'TABLE'** ♦ **'VIEW'** ♦ **'SYSTEM TABLE'** ♦ **'ALIAS'** ♦ **'SYNONYM'**

Default value

There is no default value for TableCriteria. If you do not specify a value, the TableCriteria parameter is not used.

Examples

The following examples show different ways to set the TableCriteria DBParm parameter to limit the Select Tables list:

Example 1 This statement produces a list of tables in the Powersoft repository. These tables satisfy the criteria of having names that start with pb (such as pbcatcol and pbcatedt). You do not need two commas after the percent sign for the missing *table_owner* and *table_qualifier* entries because *table_name* is the first parameter. All quotes are single quotes.

```
TableCriteria = '''pb%'''
```

Example 2 This statement produces a list of tables owned by SYSIBM. The leading comma indicates that *table_name* was intentionally omitted. All quotes are single quotes.

```
TableCriteria = ',''SYSIBM'' '
```

Example 3 This statement produces a list of views only. The three leading commas indicate that *table_name*, *table_owner*, and *table_qualifier* were intentionally omitted. The entire *table_type* string (" 'VIEW' ") is enclosed in double quotes. Within the *table_type* string, the type itself ('VIEW') is enclosed in single quotes.

```
TableCriteria = ', , , " 'VIEW' " '
```

Time

Description

When you update a data source in the DataWindow painter or Form painter, PowerBuilder or InfoMaker builds a SQL UPDATE statement in the background. If your data source is an ORACLE table, the Time DBParm parameter determines how PowerBuilder or InfoMaker specifies a time data type when it builds the SQL UPDATE statement.

Applies to

ORACLE

Syntax

Time = ' ''''*Powersoft_time_format*'''' '

> **Typing single quotes**
> When PowerBuilder or InfoMaker parses the DBParm Time string, it converts each set of four consecutive single quotes you type to one single quote. Therefore, you *must* type the syntax exactly as described in the following table.

Parameter	Description
' ''''	Type a single quote, followed by one space, followed by four single quotes. There is no space between the four single quotes and the beginning of the Powersoft time format.
Powersoft_time_format	The time format you want PowerBuilder or InfoMaker to use when it builds a SQL UPDATE statement to update a data source in the DataWindow or Form painters. (For more on Powersoft display formats, see the *User's Guide*.)
'''' '	Type four single quotes, followed by one space, followed by a single quote. There is no space between the end of the Powersoft time format and the four single quotes.

Default value

The default display format for time data types is the default ORACLE format.

Example

Assume you are using the DataWindow painter to update an ORACLE table named Workhours by setting the Start column to 08:30. This time is represented by the following Powersoft time format:

```
hh:mm
```

To specify that PowerBuilder should use this format for the time data type when it builds the SQL UPDATE statement, use the following Time DBParm statement:

```
Time = ' ''''hh:mm'''' '
```

As a result, PowerBuilder builds the following SQL UPDATE statement to update the table:

```
UPDATE WORKHOURS
SET START = '08:30'
```

See also

Date
DateTime

Setting database preferences

There are two ways to set database preferences in PowerBuilder or InfoMaker, as summarized in the following table:

Set database preferences in	If you are using	To do this
The PB.INI or IM.INI file	PowerBuilder or InfoMaker	Connect to a database
The Preferences painter	PowerBuilder	Connect to a database

The following sections give the steps for each of these methods.

Setting database preferences in InfoMaker
Editing the IM.INI file is the *only* way to set database preferences in InfoMaker.

Editing the PB.INI or IM.INI file

In PowerBuilder or InfoMaker, you can set database preferences by editing the PB.INI or IM.INI file. You can edit the file in either of the following ways:

- **In the [Database] section** Add the preferences to the [Database] section of the INI file. Keep in mind that values in the [Database] section are overwritten with new values each time you connect to a different database.
- **In the database profile** Add the preferences to the database profile for this data source or database. The preference values will be permanently saved with this database connection until you change them. Setting preferences in the database profile is useful for those database preferences that must be set *before* connecting to the database.

❖ To set database preferences by editing the PB.INI or IM.INI file:

1 Open the PB.INI or IM.INI file in one of the following ways. (The INI file is in your PowerBuilder or InfoMaker product directory.)

- Use the File Editor in PowerBuilder or InfoMaker. (For instructions, see the *User's Guide*.)
- Use any text editor outside PowerBuilder or InfoMaker.

2 Edit the INI file to set values for one or more database preferences.

Use the following syntax to specify each database preference:

PreferenceVariable=value

For examples showing how to set preferences in the INI file, see the Examples section next.

3 Save your changes to the INI file.

4 Reconnect to the database for which you set preferences.

You *must* reconnect to the database for the preferences you set to take effect.

When you reconnect to the database, PowerBuilder or InfoMaker applies the preferences you set to the current connection.

Examples

The following examples show different ways to edit your INI file to specify database preferences.

Example 1 To set the NoCatalog preference to Yes and the ReadOnly preference to 1 in the [Database] section of the INI file, add the expressions shown in bold to the [Database] section, like this:

```
[Database]
Vendors=ODBC
DBMS=ODBC
ServerName=
Database=Powersoft Demo DB
UserID=dba
DatabasePassword=
.
.
.
NoCatalog=Yes
ReadOnly=1
.
.
.
```

Example 2 To set the NoCatalog preference to Yes and the ReadOnly preference to 1 in the database profile, add the expressions shown in bold to the database profile, like this:

```
[PROFILE Powersoft Demo DB]
DBMS=ODBC
Database=Powersoft Demo DB
UserId=dba
DatabasePassword=sql
LogPassword=
ServerName=
LogId=
Lock=
DbParm=Connectstring='DSN=Powersoft Demo DB'
Prompt=0
NoCatalog=Yes
ReadOnly=1
```

Example 3 For some database preferences, the preference name may already appear in the INI file. In that case, you need only specify the new value. For example, Lock= already appears in the database profile. To set the Lock value to RU (Read uncommitted) in the database profile, add the RU value (shown in bold) to the database profile, like this:

```
[PROFILE Powersoft Demo DB]
DBMS=ODBC
Database=Powersoft Demo DB
UserId=dba
DatabasePassword=sql
LogPassword=
ServerName=
LogId=
Lock=RU
DbParm=Connectstring='DSN=Powersoft Demo DB'
Prompt=0
```

Using the Preferences painter

In PowerBuilder, you can also set database preferences by using the Preferences painter. PowerBuilder stores the preferences you set in the [Database] section of the PB.INI file.

If you are using PowerBuilder and want to set preferences in the [Database] section of the INI file, it is quicker and easier to use the Preferences painter to do this than to edit your INI file.

❖ To set database preferences in PowerBuilder by using the Preferences painter:

1 Open the Preferences painter.

For instructions, see the PowerBuilder *User's Guide*.

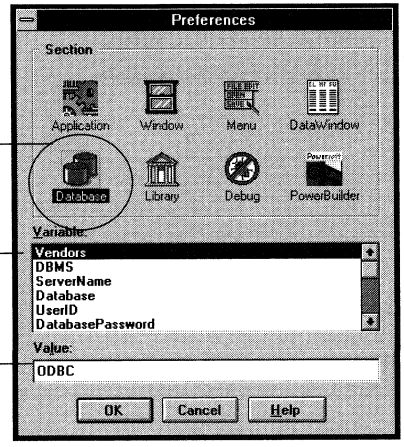
2 Click the Database button in the Preferences painter.

The variables for the [Database] section of the PB.INI file appear in the Variable listbox.

When you click here

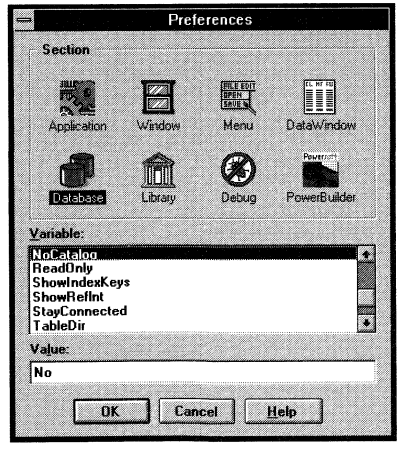
Database variables appear here

Value of selected variable appears here



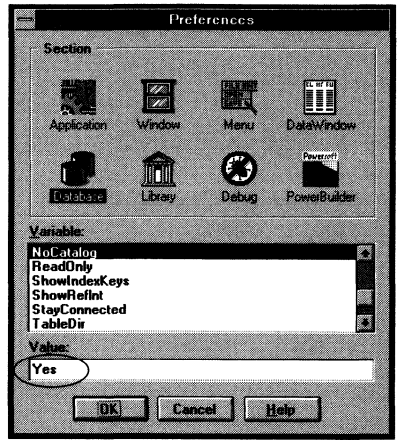
3 Select the name of the database preference you want to set from the Variable list.

The current setting of the selected variable appears in the Value box.



4 In the Value box, type a new value for the variable.

For example, to set the NoCatalog database preference to Yes, type **Yes** in the Value box.



5 Repeat steps 3 and 4 to set other preferences for this database connection.

6 Click the OK button to save your changes.

The Preferences painter closes.

PowerBuilder or InfoMaker saves your preferences in the [Database] section of the PB.INI or IM.INI file.

7 Reconnect to the database for which you set preferences.

You *must* reconnect to the database in order for the preferences you set to take effect.

When you reconnect to the database, PowerBuilder or InfoMaker applies the preferences you set to the current connection.

Database preferences and supported DBMSs

The table on the next page lists some of the database preferences that you can set in PowerBuilder or InfoMaker and the DBMSs to which each preference applies.

A checkmark indicates that you can set a particular database preference when connecting to this DBMS.

Database preferences covered in this table

This table does not list all database preferences that you can set in PowerBuilder or InfoMaker. It lists only those database preferences that:

- Pertain to your database connection, and not to the Database painter itself
- You *cannot* set in the Database Profile Setup dialog box when you define a Powersoft database interface

For complete information about database and other preferences that you can set in PowerBuilder or InfoMaker, see the online Help.

Database preferences	ALLBASE/ SQL	IBM DRDA	INFOR- MIX	MDI Gateway	ODBC	ORACLE
AutoCommit				✓	✓	
Lock	✓		✓		✓	
NoCatalog	✓	✓	✓	✓	✓	✓
ReadOnly	✓	✓	✓	✓	✓	✓
StoredReqOwner				✓		
TableSpace				✓		✓

Database preferences	SQL Server	SQL Base	Sybase Net- Gateway	Sybase SQL Server System 10	XDB
AutoCommit	✓			✓	
Lock		✓			✓
NoCatalog	✓	✓	✓	✓	✓
ReadOnly	✓	✓	✓	✓	✓
StoredReqOwner			✓		
TableSpace			✓		

Descriptions of database preferences

This section describes the syntax and use of each database preference listed on the table on page 380. The database preferences are described in alphabetical order.

AutoCommit

Description

All DBMSs support transaction processing, but not all DBMSs allow you to turn transaction processing on or off. For those DBMSs that allow it, the AutoCommit preference specifies whether you want to turn on or turn off normal recoverable transaction processing.

A **transaction** is one or more SQL statements that forms a **logical unit of work (LUW)**. Within a transaction, all SQL statements must succeed or fail as one logical entity. Changes are made to the database only if all statements in the transaction succeed and a COMMIT is issued. If one or more statements fail, you must issue a ROLLBACK to undo the changes. This ensures the integrity and security of data in your database.

By default, AutoCommit is set to False. This turns on normal recoverable transaction processing when you connect to the database.

Applies to

Micro Decisionware Database Gateway Interface for DB2
ODBC
SQL Server
Sybase SQL Server System 10

Syntax

AutoCommit = *value*

Parameter	Description
value	Specifies whether normal recoverable transaction processing is on or off. Values are: ♦ **True** *Turns off* normal recoverable transaction processing. PowerBuilder or InfoMaker automatically commits each successful statement as it occurs, and not as part of an LUW. When AutoCommit is set to True, you *cannot* issue a ROLLBACK to undo your changes. ♦ **False** (Default) *Turns on* normal recoverable transaction processing. PowerBuilder or InfoMaker issues a BEGIN TRANSACTION statement at the start of the connection. Also, after each COMMIT or ROLLBACK is issued, PowerBuilder or InfoMaker issues another BEGIN TRANSACTION statement.

Default value

```
AutoCommit = False
```

Usage

Some DBMSs require you to execute certain SQL statements outside the context of a transaction. For example, when connected to a SQL Server database, you must execute SQL Data Definition Language (DDL) statements such as CREATE TABLE and DROP TABLE outside a transaction. There are two reasons for this:

- It ensures that the structure of your database cannot change during a transaction
- It improves database performance, because DDL statements are costly operations to recover

Therefore, to execute DDL statements or stored procedures containing DDL statements in a SQL Server database, you must set AutoCommit to True to turn off transaction processing. This allows you to execute the DDL operation outside a transaction. You should, however, set AutoCommit back to False immediately after completing the DDL operation to restore normal recoverable transaction processing.

> ***Caution***
>
> *When you set the AutoCommit preference to True, you cannot roll back database changes. Therefore, use care when you change the setting of AutoCommit.*

Micro Decisionware Database Gateway Interface When you connect to a DB2 database through the Micro Decisionware Database Gateway Interface for DB2, the effect of setting the AutoCommit preference depends on how the gateway is configured at your site, as follows:

- If the Micro Decisionware Database Gateway Interface at your site is configured for *long transactions*, setting AutoCommit has no effect.
- If the Micro Decisionware Database Gateway Interface at your site is configured for *short transactions*:
  - Setting AutoCommit to *False* causes PowerBuilder or InfoMaker to change the gateway configuration to support long transactions.
  - Setting AutoCommit to *True* causes PowerBuilder or InfoMaker to leave the gateway configured for short transactions.

Example

This statement turns off normal recoverable transaction processing:

```
AutoCommit = True
```

Lock

Description

Sets the isolation level to use when you connect to the database.

In multiuser databases, transactions initiated by different users can overlap. If these transactions access common data in the database, they can overwrite each other, or collide.

To prevent concurrent transactions from interfering with each other and compromising the integrity of your database, certain DBMSs allow you to set the isolation level when you connect to the database. **Isolation levels** are defined by your DBMS, and specify the degree to which operations in one transaction are visible to operations in a concurrent transaction. Isolation levels determine how your DBMS isolates or *locks* data from other processes while it is being accessed.

PowerBuilder and InfoMaker use the Lock preference to allow you to set various database lock options. Each lock value you can set corresponds to an isolation level defined by your DBMS.

When to specify the Lock value

You must specify the Lock value *before* you connect to the database in PowerBuilder or InfoMaker. The easiest way to do this is to edit the database profile in your PB.INI or IM.INI file to include the Lock value. (For instructions and an example, see "Editing the PB.INI or IM.INI file" on page 374.)

The Lock value is honored only at the moment the database connection occurs. Changes to the Lock value after the connection occurs have no effect on the connection.

Applies to

ALLBASE/SQL
INFORMIX
ODBC
SQLBase
XDB (multiuser databases only)

Syntax

Lock = *value*

where *value* is the isolation level you want to set. The following table lists the lock values you can set, corresponding isolation levels, and default values for each DBMS that supports locking.

For more information

For more about isolation levels in a Watcom SQL database, see *Watcom SQL* or online Help if you are using PowerBuilder, or online Help if you are using InfoMaker.

For more about the isolation levels for other DBMSs, see your DBMS documentation.

DBMS	Lock values	Isolation levels	Default value
ALLBASE/SQL	CS	Cursor stability	RR
	RL	Release locks	
	RR	Repeatable read	
INFORMIX	Dirty read	Dirty read	Depends on how your database is configured
	Committed read	Committed read	
	Cursor stability	Cursor stability	
	Repeatable read	Repeatable read	
ODBC	RU	Read uncommitted	Depends on how your database is configured
	RC	Read committed	
	RR	Repeatable read	
	TS	Serializable transactions	
	TV	Transaction versioning	
SQLBase	RR	Read repeatability	RR
	CS	Cursor stability	
	RO	Read only	
	RL	Release locks	
XDB (multiuser databases only)	RR	Repeatable read	Depends on how your database is configured
	EU	Exclusive use	
	LC	Lock current	
	DR	Dirty read	
	CS	Cursor stability	

Default value

See the preceding table for the default lock values for each DBMS.

Usage

For ALLBASE/SQL, set the lock value to RL (Release locks) to improve performance when you are developing applications. However, you *cannot* perform cursor updates when Lock is set to RL.

For ODBC, the TV (Transaction versioning) setting does *not* apply to a Watcom SQL database.

Examples

Example 1 This statement sets the Lock value to RC (Read committed) for a Watcom SQL database:

```
Lock = RC
```

Example 2 This statement sets the Lock value to Cursor stability for an INFORMIX database:

```
Lock = Cursor stability
```

NoCatalog

Description

Controls access to the Powersoft repository by specifying whether you want PowerBuilder or InfoMaker to create the Powersoft repository tables. The Powersoft repository, also known as the Powersoft catalog or Powersoft system tables, consists of five tables that contain default extended attribute information for your database.

By default, the NoCatalog parameter is set to 0 (No). This creates the repository tables the first time you connect to a database using PowerBuilder or InfoMaker.

Applies to

All DBMSs

Syntax

NoCatalog = *value*

Parameter	Description
value	Specifies whether you want PowerBuilder or InfoMaker to create the Powersoft repository. Values are: ♦ **1** If the Powersoft repository tables do not exist, PowerBuilder or InfoMaker does not create them. Instead, the DataWindow, Report, and Form painters use the appropriate default values for extended attributes (such as headers, labels, and text color). If the Powersoft repository tables already exist, PowerBuilder or InfoMaker does not use them when you create a new DataWindow, report, or form. You can also specify **Yes** to set this value. ♦ **0** (Default) PowerBuilder or InfoMaker creates the Powersoft repository tables the first time you connect to a database. You can also specify **No** to set this value.

Default value

```
NoCatalog = 0
```

Usage

If you set the NoCatalog parameter to 1 or Yes, PowerBuilder or InfoMaker executes *no* statements that reference the Powersoft repository. PowerBuilder or InfoMaker does *not* create the repository tables; insert, update, or delete rows in them; or select information from them (such as header names).

Example

This statement specifies that PowerBuilder or InfoMaker should not create the Powersoft repository tables when you first connect to a database, or should not use them if they already exist:

```
NoCatalog = 1
```

See also

ReadOnly

ReadOnly

Description

Controls access to your database. ReadOnly specifies whether you want PowerBuilder or InfoMaker to create the Powersoft repository tables and whether you can modify the information in these tables. The Powersoft repository, also known as the Powersoft catalog or Powersoft system tables, consists of five tables that contain default extended attribute information for your database.

By default, the ReadOnly parameter is set to 0. This creates the repository tables the first time you connect to a database using PowerBuilder or InfoMaker. When you modify the information in these tables, PowerBuilder or InfoMaker updates them with your changes.

Applies to

All DBMSs

Syntax

ReadOnly = *value*

Parameter	Description
value	Specifies whether you want PowerBuilder or InfoMaker to create the Powersoft repository. Values are: ♦ **1** If the Powersoft repository tables do not exist, PowerBuilder or InfoMaker does not create them. Instead, the DataWindow, Report, and Form painters use the appropriate default values for extended attributes (such as headers, labels, and text color). If the Powersoft repository tables already exist, PowerBuilder or InfoMaker uses them when you create a new DataWindow, report, or form, but *does not update them*. Therefore, you *cannot* modify information in *any* tables from the DataWindow, Report, or Form painters when ReadOnly is set to 1. ♦ **0** (Default) PowerBuilder or InfoMaker creates the Powersoft repository tables the first time you connect to a database, and updates the tables when you modify them.

Default value

```
ReadOnly = 0
```

Usage

If you set the ReadOnly parameter to 1, you *cannot* modify information in *any* tables from the DataWindow, Report, or Form painters. Therefore, you can use only the following SQL statements in PowerBuilder or InfoMaker:

- SELECT statements in embedded SQL
- SELECT and Retrieve statements in the DataWindow, Report, and Form painters

Example

This statement specifies that PowerBuilder or InfoMaker should not create the Powersoft repository tables when you first connect to a database, or should not update them if they already exist:

```
ReadOnly = 1
```

See also

NoCatalog

StoredReqOwner

Description

Specifies the name of the owner of the table containing database stored requests. **Database stored requests** are SQL statements that reside in the host request library. When you connect to a DB2 database through either the Micro Decisionware Database Gateway Interface or Sybase Net-Gateway Interface, you can use stored requests in embedded SQL statements to perform retrieval or update operations on the database.

For more about using stored requests with embedded SQL, see online Help.

When to specify StoredReqOwner

If you need to use database stored requests, specify a value for StoredReqOwner *before* you connect to a DB2 database through either the Micro Decisionware Database Gateway Interface or Sybase Net-Gateway Interface.

Applies to

Micro Decisionware Database Gateway Interface for DB2
Sybase Net-Gateway Interface for DB2

Syntax

StoredReqOwner = *table_owner*

Parameter	Description
table_owner	The name of the owner of the table containing database stored requests

Default value

There is no default value for the StoredReqOwner parameter. If you do not specify a value for StoredReqOwner, PowerBuilder or InfoMaker prompts you to supply it.

Example

This statement sets FRAN as the owner of the table containing stored requests:

```
StoredReqOwner = FRAN
```

TableSpace

Description

If your database is partitioned with tablespace, specifies the name of the tablespace in which you want PowerBuilder or InfoMaker to create a table. A **tablespace** is a logical storage unit of a database. The data in a database is logically stored in the tablespace and physically stored in data files.

When you are connected to an ORACLE database or to a DB2 database through the Micro Decisionware Database Gateway Interface or Sybase Net-Gateway Interface, the SQL CREATE TABLE statement accepts the TABLESPACE option. This means that you can create a table in a specific tablespace. To do so, you must specify the tablespace as the value for the TableSpace parameter.

When to specify TableSpace

You must specify the TableSpace preference *before* you connect to the database.

If you create a table while connected to a DB2 database through the Micro Decisionware Database Gateway Interface or the Sybase Net-Gateway Interface, PowerBuilder or InfoMaker prompts you to supply the tablespace if you have not already done so.

Applies to

Micro Decisionware Database Gateway Interface for DB2
ORACLE
Sybase Net-Gateway Interface for DB2

Syntax

TableSpace = *tablespace_name*

Parameter	Description
tablespace_name	The name of the tablespace in which you want PowerBuilder or InfoMaker to create the table. For DB2/MVS databases, the default tablespace is the database name. For an ORACLE database, the default tablespace is the one assigned to the user logged on to the database.

Default value

The default value for TableSpace depends on the DBMS you are accessing, as summarized in the following table:

DBMS	TableSpace default value
DB2 (through Micro Decisionware Database Gateway Interface or Sybase Net-Gateway Interface)	PowerBuilder or InfoMaker uses the database name as the default tablespace.
ORACLE	PowerBuilder or InfoMaker uses the default tablespace assigned to the user logged on to the database.

Example

This statement sets Sales as the tablespace in which you want PowerBuilder or InfoMaker to create a table:

```
TableSpace = Sales
```

CHAPTER 6

Troubleshooting Your Connection

About this chapter

This chapter describes how to troubleshoot your database connection in PowerBuilder or InfoMaker by using either of the following tools:

- Database Trace
- ODBC Driver Manager Trace

It also tells where you can find more information about troubleshooting an IBM DRDA database connection.

Contents

Topic	Page
Overview of troubleshooting tools	394
About the Database Trace tool	395
Starting the Database Trace tool	398
Stopping the Database Trace tool	404
Using the Database Trace log	405
Sample Database Trace output	407
Using the ODBC Driver Manager Trace	409
Troubleshooting your IBM DRDA connection	414

Overview of troubleshooting tools

When you use PowerBuilder or InfoMaker, there are two tools available to trace your database connection, as summarized in the following table:

Use this tool	To trace a connection to
Database Trace	Any DBMS that PowerBuilder or InfoMaker supports, including ODBC
ODBC Driver Manager Trace	An ODBC data source only

The rest of this chapter describes how to use these tools to identify and resolve problems with your database connection.

About the Database Trace tool

The Database Trace tool records the internal commands that PowerBuilder or InfoMaker performs while you are connected to a database. You can trace a database connection while you are developing an application in PowerBuilder or InfoMaker or, if you are using PowerBuilder, in a PowerBuilder application that connects to a database.

PowerBuilder or InfoMaker writes the output of the Database Trace tool to a log file named PBTRACE.LOG. When you enable database tracing for the first time, PowerBuilder or InfoMaker creates the PBTRACE.LOG file in your Windows directory. Tracing continues until you disconnect from the database.

Using the Database Trace tool with one connection
You can only use the Database Trace tool for one DBMS vendor at a time and for one database connection at a time.

For example, if your PowerBuilder application connects to both an ODBC data source and a SQL Server database, you can trace either the ODBC connection or the SQL Server connection, but not both connections at the same time.

How you can use the Database Trace tool

You can use the information from the Database Trace tool to help you understand what PowerBuilder or InfoMaker is doing *behind the scenes* when you work with your database. Examining the information in the PBTRACE.LOG file can help you:

- Understand how PowerBuilder or InfoMaker interacts with your database
- Identify and resolve problems with your database connection
- Provide useful information about your database connection to Powersoft Technical Support if you call them for help

If you are familiar with PowerBuilder or InfoMaker and your DBMS, you can use the information in PBTRACE.LOG to help troubleshoot connection problems on your own.

If you are less experienced or need help, run the Database Trace tool *before* you call Powersoft Technical Support. You can then report or send the results of the trace to the Powersoft Technical Support Representative who takes your call.

Contents of the Database Trace log

The Database Trace tool records the following information in the PBTRACE.LOG file when you trace a database connection:

- Parameters used to connect to the database
- Time to perform each database operation (in 55-millisecond increments)
- The internal commands that PowerBuilder or InfoMaker performs to retrieve and display table and column information from your database. Examples of these database commands include:
  - Preparing and executing SQL statements such as SELECT, INSERT, UPDATE, and DELETE
  - Getting column descriptions
  - Fetching table rows
  - Binding user-supplied values to columns (if you are connected to a database that supports bind variables)
  - Committing and rolling back database changes
- Disconnecting from the database
- Shutting down the database interface

Format of the Database Trace log

The specific contents of the PBTRACE.LOG file depends on the database you are accessing and the operations you are performing. However, the PBTRACE.LOG file uses the following basic format to display output for all databases:

```
COMMAND: (time)
   {additional_information}
```

Parameter	Description
COMMAND	The command that PowerBuilder or InfoMaker executes to perform the database operation.
time	The number of milliseconds it takes PowerBuilder or InfoMaker to perform the command, in increments of 55 milliseconds. If you are using PowerBuilder or InfoMaker on a PC running Windows, the Database Trace timer rounds down to the nearest 55-millisecond increment. For example, if an operation takes 54 milliseconds or less to perform, the PBTRACE.LOG file displays the time as 0 milliseconds. If an operation takes between 55 and 109 milliseconds, the time is displayed as 55 milliseconds.
additional_information	Optional additional information about the command. The information provided depends on the database operation.

For example, the following portion of a PBTRACE.LOG file shows the commands PowerBuilder or InfoMaker executes to fetch two rows from a database table:

```
FETCH NEXT: (55 MilliSeconds)
   dept_id= dept_name=Business Services
FETCH NEXT: (0 MilliSeconds)
   dept_id= dept_name=Corporate Management
```

For a more complete example of Database Trace output, see "Sample Database Trace output" on page 407.

Starting the Database Trace tool

To trace your database connection, you must start the Database Trace tool. There are two ways to start the Database Trace tool in PowerBuilder or InfoMaker, as summarized in the following table:

Start Database Trace	If you are using	To do this
By editing a database profile	PowerBuilder or InfoMaker	Connect to a database
In a PowerBuilder script	PowerBuilder	Develop an application that connects to a database

The following sections give the steps for using each of these methods.

Starting Database Trace in InfoMaker
Editing a database profile is the *only* way to start the Database Trace tool in InfoMaker.

Starting Database Trace by editing a database profile

To start the Database Trace tool in PowerBuilder or InfoMaker, edit the database profile for the connection you want to trace, as described in the following procedure. PowerBuilder or InfoMaker automatically creates a database profile when you:

- Define an ODBC data source by completing the ODBC setup window for your data source driver. (For instructions, see Chapter 2, "Using ODBC Data Sources.")
- Define a Powersoft database interface by completing the Database Profile Setup dialog box. (For instructions, see Chapter 3, "Using Powersoft Database Interfaces.")

❖ To start the Database Trace tool by editing a database profile:

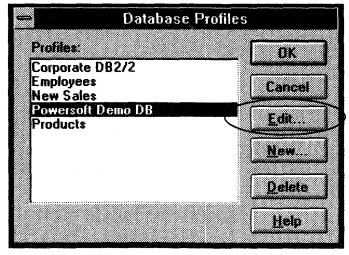
1 Click the Database Profile button in the PowerBar.
or
In any of the painters listed in the following table, select File➤Connect➤Setup from the menu bar.

Product	Painters
PowerBuilder	Database painter DataWindow painter Report painter
InfoMaker	Database painter Form painter Report painter

Database Profile button
If your PowerBar does not include the Database Profile button, use the customize feature to add the button to the PowerBar.

For instructions on customizing toolbars, see the PowerBuilder or InfoMaker *User's Guide*.

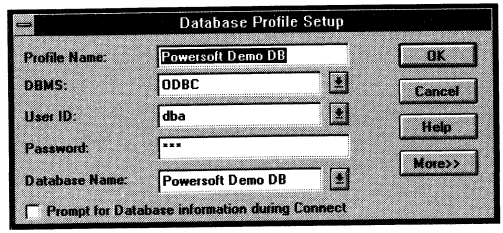
The Database Profiles dialog box appears listing the names of existing profiles. If you are currently connected to a database profile, its name is highlighted.



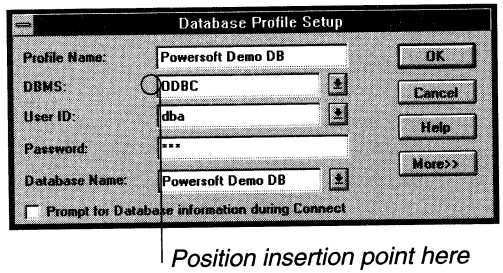
2 Select the name of the profile for the database connection you want to trace.

3 Click the Edit button.

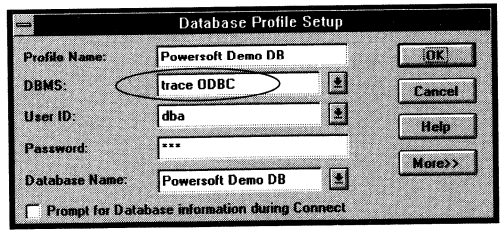
The Database Profile Setup dialog box for the selected profile appears.



4 Position the insertion point to the left of the name in the DBMS field.



5 Type the word **trace** followed by one space to the left of the DBMS name.

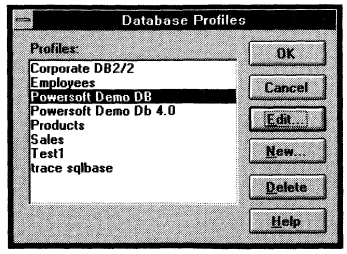


6 Click OK in the Database Profile Setup dialog box.

A message box appears stating that database tracing is enabled and that output will be written to the PBTRACE.LOG file in your Windows directory.

7 Click OK in the message box.

The Database Profiles dialog box appears with the name of the edited profile highlighted.



8 Click OK in the Database Profiles dialog box.

PowerBuilder or InfoMaker connects to the database profile you edited and starts tracing the database connection.

Starting Database Trace in a PowerBuilder script

For PowerBuilder developers only
Read this section if you are a PowerBuilder developer who wants to trace a database connection in a PowerBuilder application script.

For more about using transaction objects to communicate with a database in a PowerBuilder application, see *Building Applications*.

If you are developing a PowerBuilder application that connects to a database, you must specify the required connection parameters in the appropriate script. For example, you might specify connection parameters in the script that opens the application.

To trace a database connection in a PowerBuilder script, you specify the name of the DBMS preceded by the word **trace** and a single space. You can do this in two ways:

- By setting a value for the DBMS attribute of the default transaction object
- By reading the DBMS value from an external text file

Setting a value for the DBMS attribute

One way to start the Database Trace tool in a PowerBuilder script is to specify it as part of the DBMS attribute of the default transaction object. PowerBuilder uses a special, nongraphic object called a **transaction object** to communicate with the database. The default transaction object is named SQLCA, which stands for SQL Communications Area.

SQLCA has 15 attributes, ten of which are used to connect to your database. One of the ten connection attributes is DBMS. The DBMS attribute contains the name of the database to which you want to connect.

❖ To start the Database Trace tool by specifying the DBMS attribute:

1 Open the application script in which you want to specify connection parameters.

For instructions, see the PowerBuilder *User's Guide*.

2 Use the following PowerScript syntax to specify the DBMS attribute. (This syntax assumes you are using the default transaction object SQLCA.)

sqlca.DBMS = "**trace** *DBMS_name*"

For example, the following statements in a PowerBuilder script set the SQLCA attributes required to connect to a SQL Server database named Test. The keyword **trace** in the DBMS attribute indicates that you want to trace the database connection.

```
sqlca.DBMS       =  "trace Sybase"
sqlca.database   =  "Test"
sqlca.logId      =  "Frans"
sqlca.LogPass    =  "xxyyzz"
sqlca.ServerName =  "Tomlin"
```

Reading the DBMS value from an external text file

You can use the PowerScript ProfileString function to read the DBMS value from an external text file. The DBMS value can come from the [Database] section of your PB.INI file, or from a specified section of an application-specific INI file.

The following procedure assumes that the DBMS value read from your INI file uses the following syntax to enable database tracing for your connection:

DBMS = **trace** *DBMS_name*

❖ To start the Database Trace tool by reading the DBMS value from an external text file:

1 Open the application script in which you want to specify connection parameters.

For instructions, see the PowerBuilder *User's Guide*.

2 Use the following PowerScript syntax to specify the ProfileString function with the DBMS attribute:

sqlca.DBMS = ProfileString (*file, section, variable, default_value*)

For example, the following statement in a PowerBuilder script reads the DBMS value from the [Database] section of the PB.INI file:

```
sqlca.DBMS=ProfileString("PB.INI","Database","DBMS","")
```

Stopping the Database Trace tool

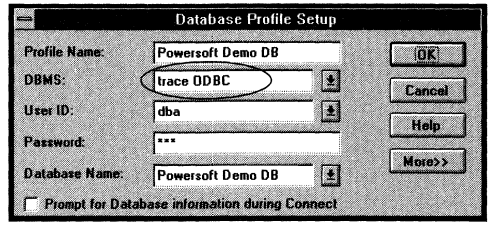
Once you start tracing a particular database connection, PowerBuilder or InfoMaker continues sending trace output to the PBTRACE.LOG file until you do one of the following:

- Reconnect to the same database with tracing stopped
- Connect to another database for which you have not enabled tracing

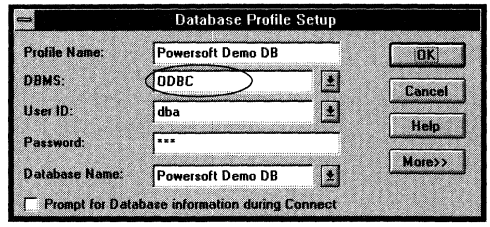
❖ To stop tracing a database connection:

1 In the Database Profile Setup dialog box for the database you are tracing, delete the word **trace** from the DBMS field.

For example, here is the Database Profile Setup dialog box for the Powersoft Demo DB with tracing enabled:



Here is how the Database Profile Setup dialog box should look after you edit it to stop tracing:



2 Click OK in the Database Profile Setup dialog box.

The Database Profiles dialog box appears, with the name of the edited profile highlighted.

3 Click OK in the Database Profiles dialog box.

PowerBuilder or InfoMaker connects to the database profile you edited and stops tracing the database connection.

Using the Database Trace log

PowerBuilder or InfoMaker writes the output of the Database Trace tool to the PBTRACE.LOG file in your Windows directory. To use the trace log, you can do the following anytime:

- View the PBTRACE.LOG file with any text editor
- Annotate the PBTRACE.LOG file with your own comments
- Delete the PBTRACE.LOG file or clear its contents when it becomes too large

Viewing the log

You can display the contents of the PBTRACE.LOG file anytime during a PowerBuilder or InfoMaker session.

❖ **To view the contents of the PBTRACE.LOG file:**

- Open the PBTRACE.LOG file in one of the following ways. The PBTRACE.LOG file is in your Windows directory.
  - Use the File Editor in PowerBuilder or InfoMaker. (For instructions, see the *User's Guide*.)
  - Use any text editor outside PowerBuilder or InfoMaker.

  If you want, you can leave the PBTRACE.LOG file open as you work in PowerBuilder or InfoMaker. However, the Database Trace tool does not update the PBTRACE.LOG file if you leave it open.

Annotating the log

When you use the PBTRACE.LOG file as a troubleshooting tool, it may be helpful to add your own comments or notes to the file. For example, you can specify the date and time of a particular connection, the versions of database software you used, or any other useful information.

❖ To annotate the PBTRACE.LOG file:

1. Open the PBTRACE.LOG file in one of the following ways. The PBTRACE.LOG file is in your Windows directory.
   - Use the File Editor in PowerBuilder or InfoMaker. (For instructions, see the *User's Guide*.)
   - Use any text editor outside PowerBuilder or InfoMaker.
2. Edit the log file with your comments.
3. Save your changes to the log file.

Deleting or clearing the log

Each time you connect to a database with tracing enabled, PowerBuilder or InfoMaker appends the trace output of your connection to the existing PBTRACE.LOG file. As a result, the log file can become very large over time, especially if you frequently enable tracing when connected to a database in PowerBuilder or InfoMaker.

❖ To keep the size of the PBTRACE.LOG file manageable:

- Do one of the following periodically:
  - Open the PBTRACE.LOG file, clear its contents, and save the empty file. PowerBuilder or InfoMaker will write to this file the next time you connect to a database with tracing enabled.
  - Delete the PBTRACE.LOG file. PowerBuilder or InfoMaker will automatically create a new PBTRACE.LOG file the next time you connect to a database with tracing enabled.

Sample Database Trace output

This section provides a typical example of Database Trace output you might see in the PBTRACE.LOG file and briefly explains each portion of the output.

The example traces a connection to a SQL Server database named Sample. The output was generated while running a PowerBuilder application that displays information about employees in each department. The SELECT statement shown retrieves information from the Employee table to display the names of employees in department 100.

Connect to SQL Server database

```
LOGIN: (220 MilliSeconds)
CONNECT TO trace sybase:
DATA=sample
LOGID=sa
DBPARM=release='4.2'
SERVER=oregano (55 MilliSeconds)
BEGIN TRANSACTION: (0 MilliSeconds)
```

Prepare SELECT statement

```
PREPARE: (0 MilliSeconds)
SELECT employee.emp_id, employee.emp_lname,
employee.emp_fname, employee.dept_id FROM employee
WHERE ( employee.dept_id = 100 )
ORDER BY employee.emp_lname ASC (0 MilliSeconds)
```

Get column descriptions

```
DESCRIBE: (0 MilliSeconds)
name=emp_id,len=4,type=????,pbt5,dbt3,ct0,dec0
name=emp_lname,len=21,type=CHAR,pbt1,dbt1,ct0,dec0
name=emp_fname,len=21,type=CHAR,pbt1,dbt1,ct0,dec0
name=dept_id,len=4,type=????,pbt5,dbt3,ct0,dec0
```

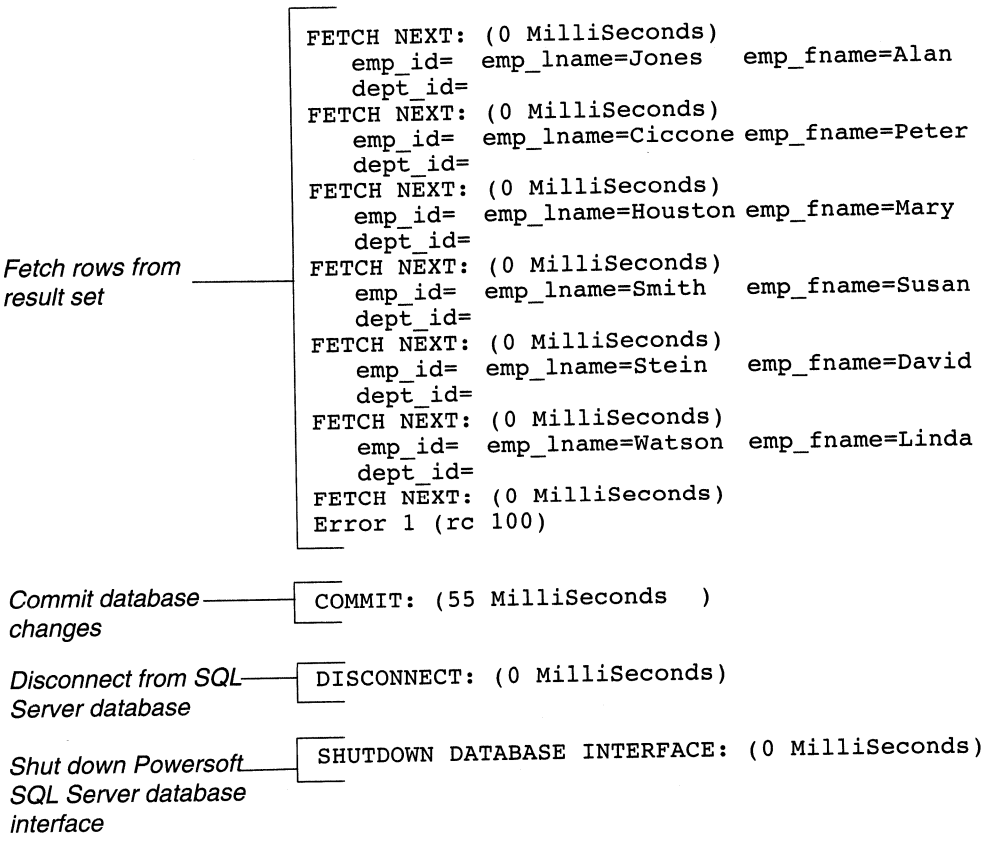
Bind memory buffers to columns

```
BIND SELECT OUTPUT BUFFER (DataWindow):
(0 MilliSeconds)
name=emp_id,len=4,type=FLOAT,pbt3,dbt3,ct0,dec0
name=emp_lname,len=21,type=CHAR,pbt1,dbt1,ct0,dec0
name=emp_fname,len=21,type=CHAR,pbt1,dbt1,ct0,dec0
name=dept_id,len=4,type=FLOAT,pbt3,dbt3,ct0,dec0
```

Execute SELECT statement

```
EXECUTE: (0 MilliSeconds)
```

(continued)


Fetch rows from result set
FETCH NEXT: (0 MilliSeconds)
emp_id= emp_lname=Jones emp_fname=Alan
dept_id=
FETCH NEXT: (0 MilliSeconds)
emp_id= emp_lname=Ciccone emp_fname=Peter
dept_id=
FETCH NEXT: (0 MilliSeconds)
emp_id= emp_lname=Houston emp_fname=Mary
dept_id=
FETCH NEXT: (0 MilliSeconds)
emp_id= emp_lname=Smith emp_fname=Susan
dept_id=
FETCH NEXT: (0 MilliSeconds)
emp_id= emp_lname=Stein emp_fname=David
dept_id=
FETCH NEXT: (0 MilliSeconds)
emp_id= emp_lname=Watson emp_fname=Linda
dept_id=
FETCH NEXT: (0 MilliSeconds)
Error 1 (rc 100)
Commit database changes
COMMIT: (55 MilliSeconds)
Disconnect from SQL Server database
DISCONNECT: (0 MilliSeconds)
Shut down Powersoft SQL Server database interface
SHUTDOWN DATABASE INTERFACE: (0 MilliSeconds)

Using the ODBC Driver Manager Trace

You can use the ODBC Driver Manager Trace to trace a connection to any ODBC data source.

What this tool does

When you are connected to an ODBC data source, the ODBC Driver Manager Trace records information about the ODBC API calls made by PowerBuilder or InfoMaker, such as SQLDriverConnect, SQLGetInfo, SQLFetch, and so on. It writes this information to a default log file named PBSQL.LOG, or to a file that you specify.

What both tools do

The information from both the Database Trace and ODBC Driver Manager Trace tools can help you:

- Understand what PowerBuilder or InfoMaker does *behind the scenes* while you are connected to an ODBC data source
- Identify and resolve problems with your ODBC connection
- Provide useful information about your database connection to Powersoft Technical Support if you call them for help

When to use this tool

You should use the ODBC Driver Manager Trace *instead* of the Database Trace tool if you want more detailed information about the ODBC API calls made by PowerBuilder or InfoMaker.

Performance considerations
Turning on the ODBC Driver Manager Trace can slow your performance while working in PowerBuilder or InfoMaker. Therefore, you should use the ODBC Driver Manager Trace for debugging purposes only and keep it turned off when you are not debugging.

Starting the trace

To start the ODBC Driver Manager Trace in PowerBuilder or InfoMaker, you must edit the PBODB040.INI file, as described in the following procedure:

❖ To start the ODBC Driver Manager Trace:

1. Open the PBODB040.INI file in one of the following ways. The PBODB040.INI file is in your Powersoft product directory.
   - Use the File Editor in PowerBuilder or InfoMaker. (For instructions, see the *User's Guide*.)
   - Use any text editor outside PowerBuilder or InfoMaker.
2. Find the [PBCONNECTOPTIONS] section at the top of the PBODB040.INI file.

   Here is the default [PBCONNECTOPTIONS] section in the PBODB040.INI file. The ODBC Driver Manager Trace is turned off, and the log file is named PBSQL.LOG.

   ```
   [PBCONNECTOPTIONS]
   PBTrace='OFF'
   PBTraceFile='c:\PBSQL.LOG'
   ```
3. Edit the [PBCONNECTOPTIONS] section to turn on the ODBC Driver Manager Trace and, if you want, specify a nondefault log file. Use the following syntax:

   ```
   [PBCONNECTOPTIONS]
   PBTrace='ON'
   PBTraceFile='pathname_of_log_file'
   ```

   For example, the following lines turn on the ODBC Driver Manager Trace and send the output to a log file named C:\PB\MYTRACE.LOG:

   ```
   [PBCONNECTOPTIONS]
   PBTrace='ON'
   PBTraceFile='C:\PB\MYTRACE.LOG'
   ```
4. Save your changes to the PBODB040.INI file.
5. Start PowerBuilder or InfoMaker if you have not already done so.

   For instructions, see the *User's Guide*.

6 Connect to an ODBC data source in PowerBuilder or InfoMaker. (For instructions, see "Connecting to a database" in Chapter 4, "Managing Database Connections.")

PowerBuilder or InfoMaker connects to the ODBC data source and starts tracing the connection.

Stopping the trace

Once you start tracing an ODBC connection with the ODBC Driver Manager Trace, PowerBuilder or InfoMaker continues sending trace output to the log file until you edit the PBODB040.INI file to turn off the trace, as described in the following procedure:

❖ To stop the ODBC Driver Manager Trace:

1 Open the PBODB040.INI file in one of the following ways. The PBODB040.INI file is in your Powersoft product directory.

- Use the File Editor in PowerBuilder or InfoMaker. (For instructions, see the *User's Guide*.)
- Use any text editor outside PowerBuilder or InfoMaker.

2 Edit the [PBCONNECTOPTIONS] section to turn off the ODBC Driver Manager Trace.

To turn off the ODBC Driver Manager Trace, specify PBTrace='OFF', like this:

```
[PBCONNECTOPTIONS]
PBTrace='OFF'
PBTraceFile='C:\PB\MYTRACE.LOG'
```

3 Save your changes to the PBODB040.INI file.

The next time you connect to an ODBC data source, PowerBuilder or InfoMaker does not use the ODBC Driver Manager Trace to trace the connection.

Viewing the log

You can display the contents of the ODBC Driver Manager Trace log file anytime during a PowerBuilder or InfoMaker session.

❖ To view the contents of the log file:

- Open the log file specified as the value for PBTraceFile in the PBODB040.INI file. Do this in one of the following ways:
  - Use the File Editor in PowerBuilder or InfoMaker. (For instructions, see the *User's Guide*.)
  - Use any text editor outside PowerBuilder or InfoMaker.

  If you did not change the value of PBTraceFile in PBODB040.INI, PowerBuilder or InfoMaker sends the output of the ODBC Driver Manager Trace to C:\PBSQL.LOG by default.

Sample trace output

This section shows a partial example of output from the ODBC Driver Manager Trace to give you an idea of the information it provides. The example traces a connection to the Powersoft Demo DB, which is a Watcom SQL ODBC data source.

For more about a particular ODBC API call, see your ODBC documentation.

```
SQLDriverConnect(hdbc5DC70000, hwnd4BB8,
   "DSN=Powersoft Demo DB", -3, szConnStrOut, 513,
   pcbConnStrOut, 1);
SQLGetInfo(hdbc5DC70000, 6, rgbInfoValue, 512,
   pcbInfoValue);
.
.
.
SQLGetConnectOption(hdbc5DC70000, 102, pvParam);
.
.
.
SQLAllocStmt(hdbc5DC70000, phstmt094F0000);
SQLGetTypeInfo(hstmt094F0000, 0);
SQLBindCol(hstmt094F0000, 1, 1, rgbValue, 129,
   pcbValue);
.
.
.
```

```
SQLFetch(hstmt094F0000);
SQLFreeStmt(hstmt094F0000, 1);
SQLAllocStmt(hdbc5DC70000, phstmt094F0000);
SQLTables(hstmt094F0000, "(null)", 0, "dba", 3,
   "pbcattbl", -3, "(null)", 0);
.
.
.
SQLColumns(hstmt5C470000, "(null)", 0, "dba", 3,
   "employee", 8, "(null)", 0);
.
.
.
SQLStatistics(hstmt5C470000, "(null)", 0, "dba", 3,
   "employee", 8, 1, 1);
.
.
.
SQLPrimaryKeys(hstmt5C470000, "(null)", 0, "dba", 3,
   "employee", 8);
.
.
.
SQLForeignKeys(hstmt5C470000, "(null)", 0, "(null)",
   0, "(null)", 0, "(null)", 0, "dba", 3,
"employee",
   8);
.
.
.
SQLExecDirect(hstmt69870000, "select pbd_fhgt,
   pbd_fwgt, pbd_fitl, pbd_funl, pbd_fchr, pbd_fptc,
   pbd_ffce, pbh_fhgt, pbh_fwgt, pbh_fitl, pbh_funl,
   pbh_fchr, pbh_fptc, pbh_ffce, pbl_fhgt, pbl_fwgt,
   pbl_fitl, pbl_funl, pbl_fchr, pbl_fptc, pbl_ffce
   pbt_cmnt from dba.pbcattbl where
   pbt_tnam='employee' and pbt_ownr = 'dba'", -3);
.
.
.
SQLDisconnect(hdbc60270000);
SQLFreeConnect(hdbc60270000);
SQLFreeEnv(henv602F0000);
```

Troubleshooting your IBM DRDA connection

For information about common errors that may occur during an IBM DRDA connection and how to resolve them, see the FaxLine document that describes how to connect to an IBM DB2/2 database using CAE Version 1.2. For a complete list of available FaxLines, order the *Technical Information Catalog* from the Powersoft FaxLine system.

APPENDIX A

Supported Data Sources and Databases

About this appendix

This appendix lists the data sources and databases supported by the Powersoft Enterprise Series products on Windows 3.1. The Powersoft Enterprise Series products include:

- PowerBuilder Enterprise
- PowerBuilder Team/ODBC
- PowerBuilder Desktop
- InfoMaker

As new data sources and databases are supported, the updated information will be available on CompuServe and in the Powersoft BBS.

> **Using operating systems other than Windows 3.1**
> This appendix lists supported data sources and databases for the Windows 3.1 operating system. When you use the Powersoft Enterprise Series products on other operating system platforms, the data sources and databases supported may be different.

Contents

Topic	Page
ODBC data sources	416
Powersoft database interfaces	418

ODBC data sources

The following ODBC data sources are supported in PowerBuilder Enterprise, PowerBuilder Team/ODBC, PowerBuilder Desktop, and InfoMaker, unless noted otherwise.

The vendor in parentheses is the manufacturer of the ODBC driver that accesses this data source. These drivers ship with the Powersoft products and are supported by Powersoft.

Access 1.x (Microsoft)

Btrieve 5.x and 6.x (INTERSOLV)

Btrieve 5.x (Microsoft)

Clipper (INTERSOLV)

dBASE II, III, IV, and V (INTERSOLV)

dBASE III and IV (Microsoft)

Excel 3.x and 4.x (Microsoft)

FoxBASE (INTERSOLV)

FoxPro (INTERSOLV)

FoxPro 2.x for DOS (Microsoft)

FoxPro for Windows (Microsoft)

NetWare SQL (INTERSOLV)

Paradox 3.x and 4.x (INTERSOLV Paradox 4 driver)

Paradox 3.x, 4.x, and 5.x (INTERSOLV Paradox 5 driver)

Paradox 3.x (Microsoft)

Rdb (DEC) *

Text (Microsoft)

Watcom SQL

* The DEC Rdb driver ships with PowerBuilder Enterprise and Info Maker only. It does *not* ship with PowerBuilder Team/ODBC or PowerBuilder Desktop.

Using other ODBC drivers

If you are using PowerBuilder Enterprise, PowerBuilder Team/ODBC, or InfoMaker, you can also use ODBC drivers purchased from vendors *other* than Powersoft, such as directly from DBMS vendors.

In most cases, any ODBC driver that is Level 1-compliant or higher will work with these products. However, Powersoft has not tested other ODBC drivers to verify this.

Using ODBC drivers with PowerBuilder Desktop

If you are using PowerBuilder Desktop, you are restricted to the ODBC drivers shipped with the product, plus drivers obtained directly from INTERSOLV for Btrieve, dBASE, NetWare SQL, and Paradox.

To use ODBC drivers from vendors *other* than Powersoft, you must upgrade to PowerBuilder Team/ODBC.

Powersoft database interfaces

The following Powersoft database interfaces are supported in PowerBuilder Enterprise and InfoMaker (but *not* in PowerBuilder Team/ODBC or PowerBuilder Desktop):

ALLBASE/SQL

IBM DRDA

INFORMIX 4 (IN4 interface)

INFORMIX 5 (IN5 interface)

Micro Decisionware Database Gateway Interface for DB2

ORACLE 6 (OR6 interface)

ORACLE 7 (OR 7 interface)

SQL Server

SQLBase

Sybase Net-Gateway Interface for DB2

Sybase SQL Server System 10

XDB

APPENDIX B

Adding Functions to the PBODB040.INI File

About this appendix

In general, you should *not* need to modify the PBODB040.INI file. In certain situations, however, you may need to add functions to the PBODB040.INI file for your backend DBMS.

This appendix describes how to add functions to the PBODB040.INI file if necessary.

Contents

Topic	Page
About the PBODB040.INI file	420
Adding functions to the PBODB040.INI file	421

About the PBODB040.INI file

The PBODB040.INI file is installed in your Powersoft product directory. PowerBuilder and InfoMaker use the PBODB040.INI file to maintain access to extended functionality in the backend DBMS for which ODBC does not provide an API call. Examples of extended functionality are SQL syntax or function calls specific to a particular DBMS.

> ***Caution***
>
> *In most cases, you should not need to modify the PBODB040.INI file. Changes to this file can adversely affect PowerBuilder or InfoMaker. You should change the PBODB040.INI file only if you are asked to do so by a Powersoft Technical Support representative.*
>
> *If you do modify the PBODB040.INI file, first make a copy of the existing file. Then keep a record of all changes you make.*
>
> *If you call Powersoft Technical Support after modifying the PBODB040.INI file, tell the representative that you changed PBODB040.INI and describe the changes you made.*

Adding functions to the PBODB040.INI file

The PBODB040.INI file lists the functions for certain DBMSs that have ODBC drivers. If you need to add a function to the PBODB040.INI file for use with your backend DBMS, you can do either of the following:

- Add the function to the Functions section for your backend database, if this section exists in the PBODB040.INI file.
- Create new sections for your backend DBMS in the PBODB040.INI file and add the function to the appropriate Functions section.

Adding functions to an existing section

If sections for your backend DBMS already exist in the PBODB040.INI file, use the following procedure to add new functions.

❖ To add functions to an existing section in PBODB040.INI:

1. Open the PBODB040.INI file in one of the following ways. The PBODB040.INI file is in your Powersoft product directory.
   - Use the File Editor in PowerBuilder or InfoMaker. (For instructions, see the *User's Guide*.)
   - Use any text editor outside PowerBuilder or InfoMaker.
2. Locate the entry for your backend DBMS in the DBMS Driver/DBMS Settings section of the PBODB040.INI file.

For example, here is the entry for the Watcom SQL DBMS.

```
;************************************************
;DBMS Driver/DBMS Settings
;************************************************
[WATCOM SQL WOD40W]
[WATCOM SQL]
PBSyntax='WATCOM40_SYNTAX'
PBDateTime='STANDARD_DATETIME'
PBFunctions='WATCOM_FUNCTIONS'
DelimitIdentifier='YES'
PBDateTimeInvalidInSearch='NO'
PBTimeInvalidInSearch='YES'
PBQualifierIsOwner='NO'
PBSpecialDataTypes='WATCOM_SPECIALDATATYPES'
PBSystemOwner='sys'
ForeignKeyDeleteRule='Disallow if Dependent Rows
   Exist (RESTRICT),Delete any Dependent Rows
   (CASCADE),Set Dependent Columns to NULL
   (SET NULL)'
```

3 Find the name of the section in PBODB040.INI that contains function information for your backend DBMS.

To find this section, look for a line similar to the following in the DBMS Driver/DBMS Settings entry:

```
PBFunctions='section_name'
```

For example, the following line in the DBMS Driver/DBMS Settings entry for Watcom SQL indicates that the name of the Functions section is WATCOM_FUNCTIONS:

```
PBFunctions='WATCOM_FUNCTIONS'
```

4 Find the Functions section for your backend DBMS in the PBODB040.INI file.

For example, here is the Functions section for Watcom SQL:

```
;************************************************
;Functions
;************************************************
[WATCOM_FUNCTIONS]
AggrFuncs=avg(),count(),list(),max(),min(),sum()
Functions=length(),similar(),soundex(),substr(),
   string(),date(),dateformat(),datetime(),
   day(),days(),dow(),hour(),hours(),minute(),
   minutes(),second(),seconds(),month(),
   months(),now(*),today(*),weeks(),year(),
   years(),ymd(),abs(),ifnull(),isnull(),
   number(*),remainder(),mod()
```

5 To add a new function, type a comma followed by the function name at the end of the appropriate function list, as follows:

- **Aggregate functions** Add aggregate functions to the end of the AggrFuncs list.
- **All other functions** Add all other functions to the end of the Functions list.

> **Case sensitivity**
> If the backend DBMS you are using is case sensitive, be sure to use the required case when you add the function name.

The following example shows (in bold) a new function for Watcom SQL added at the end of the Functions list:

```
;************************************************
;Functions
;************************************************
[WATCOM_FUNCTIONS]
AggrFuncs=avg(),count(),list(),max(),min(),sum()
Functions=length(),similar(),soundex(),substr(),
   string(),date(),dateformat(),datetime(),
   day(),days(),dow(),hour(),hours(),minute(),
   minutes(),second(),seconds(),month(),
   months(),now(*),today(*),weeks(),year(),
   years(),ymd(),abs(),ifnull(),isnull(),
   number(*),remainder(),mod(),newfunction()
```

6 Save your changes to the PBODB040.INI file.

Adding functions to a new section

If entries for your backend DBMS do not exist in the PBODB040.INI file, use the following procedure to create the required sections and add the appropriate functions.

> **Before you start**
> For more about the settings you should supply for your backend DBMS in the PBODB040.INI file, see the comments at the end of the PBODB040.INI file.

❖ To add functions to a new section in PBODB040.INI:

1 Open the PBODB040.INI file in one of the following ways. The PBODB040.INI file is in your Powersoft product directory.

- Use the File Editor in PowerBuilder or InfoMaker. (For instructions, see the *User's Guide*.)
- Use any text editor outside PowerBuilder or InfoMaker.

2 Edit the DBMS Driver/DBMS Settings section of the PBODB040.INI file to add an entry for your backend DBMS.

> **Tip**
> The name required to identify the entry in the DBMS Driver/DBMS Settings section is in the ODBC.INI file for your backend DBMS.

Make sure that you:

- Follow the instructions in the comments at the end of the PBODB040.INI file.
- Use the same syntax as existing entries in the DBMS Driver/DBMS Settings section of PBODB040.INI.
- Include a section name for PBFunctions.

For example, here is the relevant portion of an entry for a DB2/2 database:

```
;************************************************
;DBMS Driver/DBMS Settings
;************************************************
[DB2/2]
.
.
.
PBFunctions='DB22_FUNCTIONS'
.
.
.
```

3 Edit the Functions section of the PBODB040.INI file to add an entry for your backend DBMS.

Make sure that you:

- Follow the instructions in the comments at the end of the PBODB040.INI file.
- Use the same syntax as existing entries in the Functions section of PBODB040.INI.
- Give the Functions section the name that you specified for PBFunctions in the DBMS Driver/DBMS Settings entry.

For example:

```
;************************************************
;Functions
;************************************************
[DB22_FUNCTIONS]
AggrFuncs=avg(),count(),list(),max(),min(),sum()
Functions=curdate(),curtime(),hour(), ...
```

4 Type a comma followed by the function name at the end of the appropriate function list, as follows:

- **Aggregate functions** Add aggregate functions to the end of the AggrFuncs list.
- **All other functions** Add all other functions to the end of the Functions list.

> **Case sensitivity**
> If the backend DBMS you are using is case sensitive, be sure to use the required case when you add the function name.

The following example shows (in bold) a new DB2/2 function named substr( ) added at the end of the Functions list:

```
;************************************************
;Functions
;************************************************
[DB22_FUNCTIONS]
AggrFuncs=avg(),count(),list(),max(),min(),sum()
Functions=curdate(),curtime(),hour(),substr()
```

5 Save your changes to the PBODB040.INI file.

Index

A

about this manual xiii
Access, connecting through Microsoft Access ODBC driver
  connection components 41
  database preferences 380
  DBParm parameters 310
  defining 42
  ODBC Microsoft Access Setup dialog box, values for 43
  preparing to use 42
  SQL operations supported 40
  versions supported 40
accessing databases
  ODBC data sources 3, 25, 29
  Powersoft database interfaces 3, 143
active INI files, setting up 298
aggregate functions, supported in IBM DRDA interface 168
ALLBASE/SQL database interface
  connection components 156
  data types supported 155
  database preferences 380
  DBParm parameters 310
  defining 157
  HPConnect string, specifying 158, 342
  lock values 385
  preparing the database 156
API conformance levels for ODBC 20
applications
  connecting to databases from 264, 307
  in ODBC connections 16
  in Powersoft database interface connections 142
  setting DBParm parameters 307
  tracing database connections from 401
AppName DBParm parameter 312
associating files with indexes *see* indexes, associating with files
Async DBParm parameter 306, 314
asynchronous processing, enabling 314
audience for this manual xiii
AutoCommit database preference
  about 381
  updating image and text columns 219

B

basic procedures
  defining ODBC data sources 32
  responding to connection prompts 273
  selecting a database profile 271
  setting database preferences 302, 374
  setting DBParm parameters 302, 303
  starting Database Trace 398
  starting ODBC Driver Manager Trace 410
  steps for connecting 2
  stopping Database Trace 404
  stopping ODBC Driver Manager Trace 411
  using Powersoft database interfaces 143
bind variables
  disabling default binding 338
  using in SQL statements 338
binding databases
  about 178
  with CAE 1.0 or 1.1 178
  with CAE 1.2 or 1.1 for DB2/6000 179
Block DBParm parameter
  ORACLE 315
  Sybase SQL Server System 10 316
blocking factor, setting for cursors 315, 316
Borland Database Engine, required for INTERSOLV Paradox 5 driver 102, 103
Btrieve, connecting through INTERSOLV Btrieve ODBC driver
  CDB value in connect string, setting 50
  connection components 46
  creating DDF files 51, 58
  database preferences 380
  DBParm parameters 310
  Define Tables dialog box, values for 52

Btrieve, connecting through INTERSOLV Btrieve ODBC driver (*continued*)
defining 48
NetWare SQL data dictionary 50
ODBC Btrieve Driver Setup dialog box, values for 48
preparing to use 47
required files 47
SQL operations supported 45
versions supported 45
Btrieve, connecting through Microsoft Btrieve ODBC driver
connection components 56
database preferences 380
DBParm parameters 310
defining 59
NetWare SQL data dictionary 57
ODBC Btrieve Setup dialog box, values for 60
preparing to use 57
required files 57
SQL operations supported 55
versions supported 55

C

caching SQL statements
about 359
determining cache size 362
with bind variables 339, 361
CAE, IBM *see* Client Application Enabler (CAE), IBM
case sensitivity
in IBM DRDA databases 337
in ORACLE databases 348
in PBODB040.INI file 423, 425
Catalog Database dialog box
DB2/2 example 186
unsupported with CAE 1.2 or 1.1 for DB2/6000 186
values for 185
catalog, Powersoft *see* repository
cataloging databases
about 184
with Catalog Database dialog box 184
with IBM command line interface 186
catalogs, shadow 364
CDB value, in ODBC connect string 50, 319
Change Database dialog box 276
CharSet DBParm parameter 317
Check Data utility, IBM 164
CICS resources, releasing 357
Client Application Enabler (CAE), IBM
binding with 1.0 or 1.1 178
binding with 1.2 or 1.1 for DB2/6000 179
cataloging databases with 1.0 or 1.1 184
cataloging databases with 1.2 or 1.1 for DB2/6000 186
with Database Manager and DB2/2 170
with DB2/6000 177
with DB2/MVS 173
Clipper, connecting through INTERSOLV dBASE ODBC driver 61
see also INTERSOLV dBASE ODBC driver
columns
adding to IBM DRDA tables 164
enclosing names in double quotes 336
in repository 268
concurrency control, optimistic 323, 325
Configure ODBC button 33, 278
Configure ODBC dialog box
displaying Help 30
editing data source definitions 279
using 33
configuring ODBC data sources *see* defining ODBC data sources
conformance levels for ODBC drivers
API 20
ensuring proper 19
SQL 20
connect descriptor, ORACLE
about 212
syntax and example 214
connect string, ODBC
about 28, 318
CDB value 50, 319
DSN (data source name) value 28, 281, 319
PWD (password) value 319
UID (user ID) value 319
connect string, ORACLE
about 212
syntax and examples 212
connecting to databases
about 263, 277
at startup or from a painter 263

connecting to databases (*continued*)
basic steps for 2
by responding to prompts 273
by selecting a database profile 271
during application execution 264
connection prompts *see* prompts, connection
ConnectString DBParm parameter
about 28, 318
CDB value 50, 319
DSN (data source name) value 28, 281, 319
in ODBC connections 28
PWD (password) value 319
UID (user ID) value 319
ConversionTable DBParm parameter 320
Core API conformance level for ODBC 20
Core SQL conformance level for ODBC 20
CT-Library application programming interface 235
CursorLib DBParm parameter 322
CursorLock DBParm parameter
ODBC 322
SQL Server 323
cursors
blocking factor, ORACLE 315
blocking factor, Sybase System 10 316
library, ODBC 322
locking options, ODBC 322
locking options, SQL Server 323
scrolling options, INFORMIX 358
scrolling options, ODBC 326
scrolling options, SQL Server 327
SQL Server DB-Library support 218, 355
CursorScroll DBParm parameter
ODBC 326
SQL Server 327
CursorUpdate DBParm parameter 329
Custom button, in Watcom SQL ODBC Configuration dialog box 133, 136
Customer Information Control System (CICS), IBM 357

D

Data Definition Language (DDL) statements, SQL 382
data dictionary files *see* DDF files
Data Pipeline painter, displaying terse error messages 350
data source name (DSN) value *see* DSN value
data sources *see* ODBC data sources
data types
ALLBASE/SQL 155
IBM DRDA 165
INFORMIX 189
Micro Decisionware Database Gateway Interface for DB2 201, 320
ORACLE 205
SQL Server 219
SQLBase 226
Sybase Net-Gateway Interface for DB2 231
Sybase SQL Server System 10 236
XDB 242
Database 2 for MVS, IBM *see* DB2/MVS, IBM
Database 2 for OS/2, IBM *see* DB2/2, IBM
Database 2 for RS/6000, IBM *see* DB2/6000, IBM
database interfaces, Powersoft *see* Powersoft database interfaces
Database Manager, IBM
columns, adding to tables 164
connection components 169
database preferences 380
DB2SYSPB.SQL script, using 247
DBParm parameters 310
foreign keys, defining 164
preparing the database 170
primary keys, defining 163
supported in IBM DRDA database interface 162
versions supported 168
see also IBM DRDA database interface
database preferences
about 379
AutoCommit 219, 381
Lock 376, 383
NoCatalog 269, 375, 386
ReadOnly 269, 375, 388
setting in INI files 374
setting in Preferences painter 376
StoredReqOwner 389
TableSpace 390
Database Profile button 147, 271

Database Profile Setup dialog box
about 148
ALLBASE/SQL database interface, values for 157
completing for an existing ODBC data source 292
Database Trace, starting 400
Database Trace, stopping 404
deleting profiles 289
displaying Help 145
editing data source name 281
editing profiles 286
IBM DRDA database interface, values for 180
INFORMIX IN4 database interface, values for 197
INFORMIX IN5 database interface, values for 198
Micro Decisionware Database Gateway Interface for DB2, values for 204
More button 150, 305
ORACLE OR6 and OR7 database interfaces, values for 210
setting DBParm parameters 305
SQL Server database interface, values for 224
SQL Server example 151
SQLBase database interface, values for 229
Sybase Net-Gateway Interface for DB2, values for 233
Sybase SQL Server System 10 database interface, values for 240
XDB database interface, values for 245
database profiles
about 9, 264
ALLBASE/SQL database interface 157
connect string for ODBC data sources 28, 318
creating 146
creating for existing ODBC data sources 290
Database Trace, starting 399
Database Trace, stopping 404
DBMS value for ODBC data sources 27
deleting 289
displaying Help 145
editing 286
editing data source name 281
example in INI file 152
database profiles (*continued*)
IBM DRDA database interface 180
INFORMIX IN4 database interface 197
INFORMIX IN5 database interface 198
Micro Decisionware Database Gateway Interface for DB2 204
ORACLE OR6 and OR7 database interfaces 210
password display, suppressing 152
selecting in Database Profiles dialog box 271
selecting with File menu 272
setting database preferences 374
setting DBParm parameters 304
shared, about 294
shared, displaying 297
shared, maintaining 298
SharedIni variable, setting in INI files 294
SharedIni variable, setting in Preferences painter 295
SQL Server database interface 224
SQLBase database interface 229
stored in PB.INI and IM.INI files 27
Sybase Net-Gateway Interface for DB2 233
Sybase SQL Server System 10 database interface 240
XDB database interface 245
Database Profiles dialog box
about 36, 148, 271
displaying shared profiles 297
database ranges, defining for Excel worksheets 78
Database section in INI files 277, 374
database stored requests 389
Database Trace
about 395
annotating the log 405
deleting or clearing the log 406
log file contents 396
log file format 397
sample output 407
starting in database profiles 399
starting in PowerBuilder scripts 401
stopping in database profiles 404
viewing the log 405
see also PBTRACE.LOG file

databases
accessing 3, 25, 29
basic steps for connecting 2
connecting at startup or from a painter 263
connecting during application execution 264
controlling access 388
creating Watcom SQL 5
in Powersoft database interface connections 142
lock values and isolation levels 383
logging on for the first time 266
native *see* Powersoft database interfaces
responding to connection prompts 273
selecting a database profile to connect 271
text, structure of 118
DataWindow objects, creating with ORACLE 7 stored procedures 216
Date DBParm parameter 330
DateTime data type, INFORMIX 189
DateTime DBParm parameter 332
DB2/2, IBM
adding functions to PBODB040.INI file 424
Catalog Database dialog box example 186
columns, adding to tables 164
connection components 169
database preferences 380
DB2SYSPB.SQL script, using 247
DBParm parameters 310
foreign keys, defining 164
preparing the database 170
primary keys, defining 163
supported in IBM DRDA database interface 162
versions supported 168
see also IBM DRDA database interface
DB2/6000, IBM
columns, adding to tables 164
connection components 175
database preferences 380
DB2SYSPB.SQL script, using 247
DBParm parameters 310
foreign keys, defining 164
preparing the database 176
primary keys, defining 163
supported in IBM DRDA database interface 162
versions supported 175
see also IBM DRDA database interface
DB2/MVS, IBM
columns, adding to tables 164
connection components 172
database preferences 380
DB2SYSPB.SQL script, using 247
DBParm parameters 310
foreign keys, defining 164
preparing the database 173
primary keys, defining 163
supported in IBM DRDA database interface 162
versions supported 171
see also IBM DRDA database interface
DB2SYSPB.SQL file, for creating repository in DB2 databases 248, 352
DB32W Watcom SQL command 135
DBAdm DBParm parameter 333
dBASE, connecting through INTERSOLV dBASE ODBC driver
connection components 62
database preferences 380
DBParm parameters 310
Define Tables dialog box, values for 67
defining 64
ODBC dBASE Driver Setup dialog box, values for 65
preparing to use 63
SQL operations supported 61
versions supported 61
dBASE, connecting through Microsoft dBASE ODBC driver
connection components 70
database preferences 380
DBParm parameters 310
defining 72
ODBC dBASE Setup dialog box, values for 73
preparing to use 71
Select Indexes dialog box, values for 75
SQL operations supported 69
versions supported 69
DBCLIENW Watcom SQL command 135
DBGetTime DBParm parameter 334
DB-Library application programming interface 218
DBMS
backend, adding functions for 421
database preferences supported for each 379

DBMS (*continued*)
DBParm parameters supported for each 309
dialog box 274
identifiers in Database Profile Setup dialog box 148
identifiers in DBMS dialog box 274
lock values and isolation levels 383
system tables, displaying 267
value in database profiles 27
DBParm dropdown listbox
displaying 305
editing 281
DBParm parameters
about 309
AppName 312
Async 306, 314
Block, ORACLE 315
Block, Sybase SQL Server System 10 316
CharSet 317
ConnectString 28, 50, 318
ConversionTable 320
CursorLib 322
CursorLock, ODBC 322
CursorLock, SQL Server 323
CursorScroll, ODBC 326
CursorScroll, SQL Server 327
CursorUpdate 329
Date 330
DateTime 332
DBAdm 333
DBGetTime 334
DBTextLimit 335
DelimitIdentifier 336
DisableBind 338
GroupID 340
Host 341
HPConnect 158, 342
in ODBC connections 28
INET_DBPATH 343
INET_PROTOCOL 344
INET_SERVICE 345
Language 346
Log 347
LoginTimeOut 348
MixedCase 348
MsgTerse 349
PBCatalogOwner 249, 351
PBDBMS 216, 353
DBParm parameters (*continued*)
Recovery 354
Release, SQL Server 218, 355
Release, XDB 242, 356
Request 357
Scroll 358
setting in database profiles 304
setting in PowerBuilder scripts 307
SQLCache 359
SQLQualifiers 362
SystemOwner 363
TableCriteria for IBM DRDA, Micro Decisionware, XDB 365
TableCriteria, ODBC 366
TableCriteria, ORACLE 368
TableCriteria, Sybase Net-Gateway 370
Time 372
DBSTARTW Watcom SQL command 135
DBTextLimit DBParm parameter 335
DDCS/2, IBM 174
DDF files, for Btrieve data sources
about 50, 57
creating 51, 58
DDL statements, SQL 382
DEC ODBC Driver Setup dialog box, values for 116
DEC Rdb ODBC driver
connection components 114
DEC ODBC Driver Setup dialog box, values for 116
defining the data source 115
preparing the data source 115
SQL operations supported 113
versions supported 113
Define Table dialog box, values for
INTERSOLV Btrieve ODBC driver 52
INTERSOLV dBASE ODBC driver 67
Define Text Format dialog box, values for 124
defining ODBC data sources
about 24
Access 42
Btrieve, through INTERSOLV Btrieve ODBC driver 48
Btrieve, through Microsoft Btrieve ODBC driver 59
by specifying additional values 37
by specifying data source name and location 33

defining ODBC data sources (*continued*)
changing data source name 281
creating configurations and database profiles 10
dBASE, through INTERSOLV dBASE ODBC driver 64
dBASE, through Microsoft dBASE ODBC driver 72
deleting database profiles 290
deleting definitions 283
editing database profiles 286
editing definitions 278
Excel 80
existing data sources, creating database profiles for 290
FoxPro, through Microsoft FoxPro ODBC driver 85
inherited from others 29
multiple data sources 28
NetWare SQL, through INTERSOLV NetWare SQL ODBC driver 92
other than Watcom SQL 6
outside PowerBuilder or InfoMaker 10, 291
Paradox, through INTERSOLV Paradox 4 ODBC driver 98
Paradox, through INTERSOLV Paradox 5 ODBC driver 105
Paradox, through Microsoft Paradox ODBC driver 110
Rdb, through DEC Rdb ODBC driver 115
sharing database profiles 294
text, through Microsoft Text File ODBC driver 121
Watcom SQL 4, 131
defining Powersoft database interfaces
about 145
ALLBASE/SQL 157
deleting database profiles 289
editing database profiles 286
IBM DRDA 180
INFORMIX IN4 197
INFORMIX IN5 198
Micro Decisionware Database Gateway Interface for DB2 204
ORACLE OR6 and OR7 210
sharing database profiles 294
SQL Server 224
SQLBase 229
defining Powersoft database interfaces (*continued*)
Sybase Net-Gateway Interface for DB2 233
Sybase SQL Server System 10 240
XDB 245
DelimitIdentifier DBParm parameter 336
demonstration database, Watcom SQL 4
DisableBind DBParm parameter 338
display formats, in repository 268
Distributed Database Connection Services/2 (DDCS/2), IBM 174
Distributed Relational Database Architecture (DRDA), IBM *see* IBM Distributed Relational Database Architecture (DRDA)
DLL files
in ODBC connections 16, 25
in Powersoft database interface connections 142
DRDA, IBM *see* IBM Distributed Relational Database Architecture (DRDA)
drivers, ODBC *see* ODBC drivers
DSN (data source name) value, in ODBC connect string
about 28, 319
changing 281
dynamic link libraries *see* DLL files

E

edit styles, in repository 268
error messages, displaying terse 349
Excel, connecting through Microsoft Excel ODBC driver
connection components 77
database preferences 380
database ranges, defining for worksheets 78
DBParm parameters 310
defining 80
ODBC Microsoft Excel Setup dialog box, values for 81
preparing to use 78
SQL operations not supported 76
versions supported 76
executable programs, running multiple with IBM DRDA interface 163
Extended SQL conformance level for ODBC 20

F

FaxLines, Powersoft
  getting help from 14, 30, 140, 145
  troubleshooting IBM DRDA connections 187
FIELD.DDF file 50, 57
File menu
  displaying shared database profiles 297
  using to connect 272
FILE.DDF file 50, 57
fixed-length text files, defining format 127
foreign keys, defining with IBM DRDA interface 164
FoxBASE, connecting through INTERSOLV dBASE ODBC driver 61
  see also INTERSOLV dBASE ODBC driver
FoxPro, connecting through INTERSOLV dBASE ODBC driver 61
  see also INTERSOLV dBASE ODBC driver
FoxPro, connecting through Microsoft FoxPro ODBC driver
  connection components 83
  database preferences 380
  DBParm parameters 310
  defining 85
  ODBC FoxPro Setup dialog box, values for 86
  preparing to use 84
  Select Indexes dialog box, values for 88
  SQL operations supported 82
  versions supported 82
functions
  adding to existing section in PBODB040.INI file 421
  adding to new section in PBODB040.INI file 423
  aggregate, supported in IBM DRDA interface 168
  scalar, supported in IBM DRDA interface 166

G

granting permissions on repository tables 270
GroupID DBParm parameter 340

H

help
  Database Trace, using 395
  FaxLines, using 14, 30, 140, 145
  for database profiles 145
  for IBM DRDA connections 162, 187
  for ODBC drivers 31
  for Powersoft database interfaces 146
  ODBC Driver Manager Trace, using 409
  online Help, using 14, 30, 145
Host DBParm parameter 341
HPConnect DBParm parameter
  about 158, 342
  HP-UX syntax and examples 160
  MPE/iX syntax and examples 159
HP-UX syntax for HPConnect string 160

I

IBM Check Data utility 164
IBM Database Manager *see* Database Manager, IBM
IBM DB2/2 *see* DB2/2, IBM
IBM DB2/6000 *see* DB2/6000, IBM
IBM DB2/MVS *see* DB2/MVS, IBM
IBM Distributed Relational Database Architecture (DRDA) 162
  see also IBM DRDA database interface
IBM DRDA database interface
  aggregate functions supported 168
  binding PowerBuilder or InfoMaker to the database 178
  Catalog Database dialog box 185
  cataloging databases 184
  columns, adding to tables 164
  data types supported 165
  database administrator authority, specifying 333
  database preferences 380
  databases supported 162
  DB2SYSPB.SQL script, using 247
  DBParm parameters 310
  defining 180
  executable programs, running multiple 163
  foreign keys, defining 164
  Powersoft DRDA Interface dialog box 182

IBM DRDA database interface (*continued*)
preparing Database Manger and DB2/2 168
preparing DB2/6000 175
preparing DB2/MVS 171
primary keys, defining 163
responding to connection prompts 182
scalar functions supported 166
table list, modifying 163, 365
troubleshooting 187, 414
see also Database Manager, IBM; DB2/2, IBM; DB2/6000, IBM; DB2/MVS, IBM
IDAPI files, required for INTERSOLV Paradox 5 driver 102, 103
identifiers, DBMS
list of supported 149
where they come from 148, 274
IM.INI file
about 27
connection parameters 277
database profile example 152
directory search path 299
making shared file active 298
setting database preferences 374
setting up shared database profiles 294
suppressing password display 152
Vendors list, DBMS identifiers in 148, 274
INDEX.DDF file 50, 57
indexes, associating with files
dBASE and Clipper, through INTERSOLV dBASE ODBC driver 63, 66
dBASE, through Microsoft dBASE ODBC driver 71, 74
FoxPro, through Microsoft FoxPro ODBC driver 84, 87
indexes, creating unique
Paradox, through INTERSOLV Paradox 4 ODBC driver 100
Paradox, through INTERSOLV Paradox 5 ODBC driver 107
indexes, enclosing names in double quotes 336
INET_DBPATH DBParm parameter 343
INET_PROTOCOL DBParm parameter 344
INET_SERVICE DBParm parameter 345
InfoMaker
ODBC drivers, using 7, 21, 416
setting database preferences 374
setting DBParm parameters 303
starting Database Trace 398
InfoMaker (*continued*)
supported ODBC data sources 416
supported Powersoft database interfaces 418
INFORMIX IN4 database interface
connecting from PowerBuilder scripts 199
connection components 191
cursor scrolling options, specifying 358
data types supported 189
database preferences 380
DBParm parameters 310
defining 197
lock values 385
preparing the database 193
versions supported 188
INFORMIX IN5 database interface
connecting from PowerBuilder scripts 199
connection components 192
cursor scrolling options, specifying 358
data types supported 189
database preferences 380
DBParm parameters 310
DBPATH, specifying 343
defining 198
lock values 385
network protocol, specifying 344
preparing the database 194
service name, specifying 345
versions supported 188
INFORMIX-NET client software 188, 193, 195
inheriting ODBC data sources 29
INI files
adding functions to PBODB040.INI 420
directory search path 299
in ODBC connections 25
making shared files active 298
setting database preferences 374
setting up shared database profiles 294
storing connection parameters 277
storing database profiles 152
suppressing password display 152
Vendors list, DBMS identifiers in 148, 274
see also IM.INI file; PB.INI file; PBODB040.INI file
initialization files *see* INI files
InitPath040 variable, setting in WIN.INI file 298

installing
ODBC drivers 21
Powersoft database interfaces 143
interface DLLs, in Powersoft database interface connections 142
INTERSOLV Btrieve ODBC driver
CDB value in connect string 50
connection components 46
creating DDF files 51, 58
Define Table dialog box, values for 52
defining the data source 48
NetWare SQL data dictionary 50
ODBC Btrieve Driver Setup dialog box, values for 48
preparing the data source 47
required files 47
SQL operations supported 45
using with PowerBuilder Desktop 21
versions supported 45
INTERSOLV dBASE ODBC driver
connection components 62
Define Table dialog box, values for 67
defining the data source 64
ODBC dBASE Driver Setup dialog box, values for 65
preparing the data source 63
SQL operations supported 61
using with PowerBuilder Desktop 21
versions supported 61
INTERSOLV NetWare SQL ODBC driver
connection components 90
defining the data source 92
ODBC NetWare SQL Driver Setup dialog box, values for 93
preparing the data source 91
SQL operations supported 89
using with PowerBuilder Desktop 21
versions supported 89
INTERSOLV Paradox 4 ODBC driver
connection components 95
defining the data source 98
ODBC Paradox Driver Setup dialog box, values for 99
preparing the data source 96
SQL operations supported 94
using with PowerBuilder Desktop 21
versions supported 94
INTERSOLV Paradox 5 ODBC driver
connection components 102
defining the data source 105
ODBC Paradox 5 Driver Setup dialog box, values for 106
preparing the data source 103
SQL operations supported 101
using with PowerBuilder Desktop 21
versions supported 101
interval data type, INFORMIX 190
isolation levels 383
ISQL utility, for installing Powersoft stored procedures 256

L

Language DBParm parameter 346
LDLLSQLW.DLL file, managing different versions 196
Level 1 API conformance level for ODBC 20
Level 2 API conformance level for ODBC 20
Local button, in Watcom SQL ODBC Configuration dialog box 133
local Watcom SQL connections, compared to network connections 138
Lock database preference 376, 383
locking
and DBMS isolation levels 383
cursors, ODBC 322
cursors, SQL Server 323
Log DBParm parameter 347
LOG files
PBSQL.LOG 409, 412
PBTRACE.LOG 395, 405
SQL Server 347
SQLBase 354
Watcom SQL 29, 131
logging on to databases for the first time 266
logical unit of work (LUW) 381
Login dialog box, SQL Server 276
LoginTimeOut DBParm parameter 348
LUW 381

M

master database, SQL Server 253
Micro Decisionware Database Gateway Interface for DB2
  AutoCommit setting 383
  CICS resources, releasing 357
  connection components 202
  custom data type mapping, specifying 320
  data types supported 201
  database preferences 380
  database stored requests owner, specifying 389
  DB2SYSPB.SQL script, using 247
  DBParm parameters 310
  defining 204
  preparing the database 203
  table list, modifying 365
  tablespace, specifying 390
  versions supported 201
Microsoft Access ODBC driver
  connection components 41
  defining the data source 42
  ODBC Microsoft Access Setup dialog box, values for 43
  preparing the data source 42
  SQL operations supported 40
  versions supported 40
Microsoft Btrieve ODBC driver
  connection components 56
  defining the data source 59
  NetWare SQL data dictionary 57
  ODBC Btrieve Setup dialog box, values for 60
  preparing the data source 57
  required files 57
  SQL operations supported 55
  versions supported 55
Microsoft dBASE ODBC driver
  connection components 70
  defining the data source 72
  ODBC dBASE Setup dialog box, values for 73
  preparing the data source 71
  Select Indexes dialog box, values for 75
  SQL operations supported 69
  versions supported 69
Microsoft Excel ODBC driver
  connection components 77
  defining the data source 80
  ODBC Microsoft Excel Setup dialog box, values for 81
  preparing the data source 78
  SQL operations not supported 76
  versions supported 76
Microsoft FoxPro ODBC driver
  connection components 83
  defining the data source 85
  ODBC FoxPro Setup dialog box, values for 86
  preparing the data source 84
  Select Indexes dialog box, values for 88
  SQL operations supported 82
  versions supported 82
Microsoft Paradox ODBC driver
  connection components 109
  defining the data source 110
  ODBC Paradox Setup dialog box, values for 111
  preparing the data source 110
  SQL operations supported 108
  versions supported 108
Microsoft Text File ODBC driver
  connection components 119
  Define Text Format dialog box, values for 124
  defining the data source 121
  file format, defining 123
  fixed-length files, defining format 127
  ODBC Text Setup dialog box, values for 122
  Parse dialog box 128
  preparing the data source 120
  SQL operations supported 118
  versions supported 118
Minimum SQL conformance level for ODBC 20
MixedCase DBParm parameter 348
More button, in Database Profile Setup dialog box 150, 305
MPE/iX syntax for HPConnect string 159
MsgTerse DBParm parameter 349
multiple ODBC data sources, defining 28
multiple-tier ODBC drivers 18

N

native databases *see* Powersoft database interfaces
NetWare SQL data dictionary
  about 50, 57
  creating DDF files 51, 58
NetWare SQL Requestor Utility (NSREQ.EXE), required for NetWare SQL 90, 91
NetWare SQL, connecting through INTERSOLV NetWare SQL ODBC driver
  connection components 90
  database preferences 380
  DBParm parameters 310
  defining 92
  ODBC NetWare SQL Driver Setup dialog box, values for 93
  preparing to use 91
  SQL operations supported 89
  versions supported 89
Network button, in Watcom SQL ODBC Configuration dialog box 133
network Watcom SQL connections, compared to local connections 138
NoCatalog database preference 269, 375, 386
NSREQ.EXE file, required for NetWare SQL 90, 91

O

ODBC (Open Database Connectivity)
  about 15
  components 16
  data sources, about 4
  driver conformance levels 20
  drivers from other vendors 7, 21
ODBC Btrieve Driver Setup dialog box, values for 48
ODBC Btrieve Setup dialog box, values for 60
ODBC connect string
  about 28, 318
  CDB value 50, 319
  DSN (data source name) value 28, 281, 319
  PWD (password) value 319
  UID (user ID) value 319
ODBC data sources
  about 4, 416
  Access 40
  accessing 3, 25
  Btrieve, through INTERSOLV Btrieve ODBC driver 45
  Btrieve, through Microsoft Btrieve ODBC driver 55
  caching SQL statements 359
  changing data source name 281
  connect string, specifying 318
  connecting through prompts 273
  creating configurations and database profiles 10
  cursor library, specifying 322
  cursor locking options, specifying 322
  cursor scrolling options, specifying 326
  database preferences 380
  dBASE, through INTERSOLV dBASE ODBC driver 61
  dBASE, through Microsoft dBASE ODBC driver 69
  DBParm parameters 310
  DEC Rdb 113
  deleting database profiles 290
  deleting definitions 283
  editing database profiles 286
  editing definitions 278
  error messages, displaying terse 349
  examples 4
  Excel 76
  existing, creating database profiles for 290
  FoxPro, through Microsoft FoxPro ODBC driver 82
  in ODBC connections 16
  in ODBC.INI file 26
  inheriting from others 29
  list of supported 416
  lock values 385
  multiple, defining 28
  NetWare SQL 89
  other than Watcom SQL 6
  Paradox, through INTERSOLV Paradox 4 ODBC driver 94
  Paradox, through INTERSOLV Paradox 5 ODBC driver 101
  Paradox, through Microsoft Paradox ODBC driver 108

ODBC data sources (*continued*)
procedures for defining 32
sharing database profiles 294
table list, modifying 366
text files 118
troubleshooting 395, 409
Watcom SQL 4, 129
see also defining ODBC data sources; ODBC drivers; preparing ODBC data sources
ODBC dBASE Driver Setup dialog box, values for 65
ODBC dBASE Setup dialog box
example 38
values for 73
ODBC Driver Manager (ODBC.DLL) 16
ODBC Driver Manager Trace
about 409
performance considerations 409
sample output 412
starting 410
stopping 411
viewing the log 412
see also PBSQL.LOG file
ODBC drivers
about 15, 416
Access, Microsoft 40
API conformance levels 20
Btrieve, INTERSOLV 45
Btrieve, Microsoft 55
caching SQL statements 359
conformance levels, about 19
connect string, specifying 318
cursor library, specifying 322
cursor locking options, specifying 322
cursor scrolling options, specifying 326
dBASE, INTERSOLV 61
dBASE, Microsoft 69
displaying Help 14, 31
error messages, displaying terse 349
Excel, Microsoft 76
FoxPro, Microsoft 82
from other vendors 7, 21, 416
in ODBC connections 16
in ODBCINST.INI file 25
installing 21
list of supported 416
lock values 385
login timeout, specifying 348
ODBC drivers (*continued*)
multiple-tier type 18
NetWare SQL, INTERSOLV 89
Paradox 4, INTERSOLV 94
Paradox 5, INTERSOLV 101
Paradox, Microsoft 108
PBODB040.INI file 420
Rdb, DEC 113
single-tier type 18
SQL conformance levels 20
table list, modifying 366
Text File, Microsoft 118
troubleshooting 395, 409
Watcom SQL 129
with PowerBuilder Desktop 7, 21, 113, 416
with PowerBuilder Team/ODBC 7, 22, 113, 416
ODBC FoxPro Setup dialog box, values for 86
ODBC interface
about 15
components of connection 16
connect string, specifying 318
database preferences 380
DBParm parameters 310
DLL files required 16, 25
INI files required 25
PBODB040.INI file 420
troubleshooting 409
ODBC Microsoft Access Setup dialog box, values for 43
ODBC Microsoft Excel Setup dialog box, values for 81
ODBC NetWare SQL Driver Setup dialog box, values for 93
ODBC Paradox 5 Driver Setup dialog box, values for 106
ODBC Paradox Driver Setup dialog box, values for 99
ODBC Paradox Setup dialog box, values for 111
ODBC Text Setup dialog box, values for 122
ODBC.DLL file 16
ODBC.INI file
about 26
and Configure ODBC dialog box 33
and PBODB040.INI file 424

ODBCINST.INI file
about 25
and Configure ODBC dialog box 33
online Help *see* help
Open Database Connectivity *see* ODBC
optimistic concurrency control 323, 325
Options button, in Watcom SQL ODBC Configuration dialog box 133, 136
ORACLE 7 stored procedures *see* stored procedures, ORACLE 7
ORACLE OR6 database interface
caching SQL statements 359
case sensitivity, specifying 348
connect string or descriptor, specifying 212
connection components 207
cursor blocking factor, specifying 315
data types supported 205
database preferences 380
date data type 330
DateTime data type 332
DBParm parameters 310
defining 210
preparing the database 208
table list, modifying 368
tablespace, specifying 390
time data type 372
versions supported 205
ORACLE OR7 database interface
caching SQL statements 359
case sensitivity, specifying 348
connect string or descriptor, specifying 212
connection components 208
cursor blocking factor, specifying 315
data types supported 205
database preferences 380
date data type 330
DateTime data type 332
DBParm parameters 310
defining 210
preparing the database 208
stored procedures, using as data source 214, 353
table list, modifying 368
tablespace, specifying 390
time data type 372
versions supported 205

P

painters
connecting to databases from 263
Data Pipeline 350
Preferences 215, 295, 376
Paradox Engine (PXENGWIN.DLL), required for INTERSOLV Paradox 4 driver 95, 96
Paradox, connecting through INTERSOLV Paradox 4 ODBC driver
connection components 95
database preferences 380
DBParm parameters 310
defining 98
ODBC Paradox Driver Setup dialog box, values for 99
preparing to use 96
shared data sources, accessing 97
SQL operations supported 94
unique indexes, creating 100
versions supported 94
Paradox, connecting through INTERSOLV Paradox 5 ODBC driver
connection components 102
database preferences 380
DBParm parameters 310
defining 105
ODBC Paradox 5 Driver Setup dialog box, values for 106
preparing to use 103
shared data sources, accessing 104
SQL operations supported 101
unique indexes, creating 107
versions supported 101
Paradox, connecting through Microsoft Paradox ODBC driver
connection components 109
database preferences 380
DBParm parameters 310
defining 110
ODBC Paradox Setup dialog box, values for 111
preparing to use 110
SQL operations supported 108
versions supported 108
Parse dialog box 128
passwords, suppressing display in database profiles 152

PB.INI file
about 27
connection parameters 277
database profile example 152
directory search path 299
making shared file active 298
setting database preferences 374
setting up shared database profiles 294
suppressing password display 152
Vendors list, DBMS identifiers in 148, 274
PBCatalogOwner DBParm parameter
about 351
and DB2SYSPB.SQL script 249
pbcatcol table 268
pbcatedt table 268
pbcatfmt table 268
pbcattbl table 268
pbcatvld table 268
PBDBMS DBParm parameter 216, 353
PBDBMS package, for using ORACLE 7 stored procedures 214
PBGUP040.DLL file 227
PBHPA040.DLL file 156
PBIBM040.DLL file 169, 172, 175
PBIN4040.DLL file 188, 191
PBIN5040.DLL file 188, 192
PBMDI040.DLL file 202
PBNET040.DLL file 232
PBODB040.DLL file 16
PBODB040.INI file
about 420
adding functions to existing section 421
adding functions to new section 423
case sensitivity 423, 425
finding DBMS names in ODBC.INI file 424
NoCatalog and ReadOnly database preferences 269
starting ODBC Driver Manager Trace 410
stopping ODBC Driver Manager Trace 411
PBOR6040.DLL file 205, 207
PBOR7040.DLL file 205, 208
PBOR7CAT.SQL file, using for ORACLE 7 stored procedures 215
PBSQL.LOG file
about 409
sample output 412
viewing 412
see also ODBC Driver Manager Trace
PBSYB.BAT file, for running PBSYB.SQL 258
PBSYB.SQL file
about 251
running with ISQL utility 256
running with PBSYB.BAT file 258
running with WISQL utility 255
when to run 253
PBSYB040.DLL file 221
PBSYBRT.SQL file
about 252
running with ISQL utility 256
running with WISQL utility 255
when to run 253
PBSYC.SQL file
about 252
running with ISQL utility 256
running with WISQL utility 255
when to run 253
PBSYC040.DLL file 237
PBTRACE.LOG file
about 395
annotating 405
contents 396
deleting or clearing 406
format 397
sample output 407
viewing 405
see also Database Trace
PBXDB040.DLL file 243
permissions, granting on repository tables 270
PowerBuilder Desktop
ODBC drivers, using 7, 21, 113, 275, 416
Powersoft database interfaces, not supported 141, 418
supported ODBC data sources 416
PowerBuilder Enterprise
ODBC drivers, using 7, 21, 416
supported ODBC data sources 416
supported Powersoft database interfaces 418
PowerBuilder scripts *see* scripts, PowerBuilder
PowerBuilder Team/ODBC
ODBC drivers, using 7, 22, 113, 416
Powersoft database interfaces, not supported 141, 418
supported ODBC data sources 416

Powersoft database interfaces
about 8, 141, 418
accessing 3
ALLBASE/SQL 155
components of connection 142
connecting through prompts 276
creating database profiles 10, 146
deleting database profiles 289
displaying Help 146
editing database profiles 286
IBM DRDA 162
INFORMIX 188
installing 143
list of supported 418
Micro Decisionware Database Gateway Interface for DB2 201
not installing 274
ORACLE 205
sharing database profiles 294
SQL Server 218
SQLBase 226
steps for using 143
Sybase Net-Gateway Interface for DB2 231
Sybase SQL Server System 10 235
troubleshooting 395
unsupported in PowerBuilder Team/ODBC and PowerBuilder Desktop 141
XDB 242
see also defining Powersoft database interfaces; preparing databases for use with Powersoft database interfaces
Powersoft DRDA Interface dialog box 182
Powersoft FaxLine system *see* FaxLines, Powersoft
Powersoft repository *see* repository
Powersoft stored procedures *see* stored procedures, Powersoft
Preferences painter
database preferences, setting 376
ORACLE 7 stored procedures, setting up 215
shared database profiles, setting up 295
preparing databases for use with Powersoft database interfaces
about 144
ALLBASE/SQL 156
Database Manager and DB2/2, IBM 168
DB2/6000, IBM 175
preparing databases for use with Powersoft database interfaces (*continued*)
DB2/MVS, IBM 171
INFORMIX, through IN4 interface 193
INFORMIX, through IN5 interface 194
Micro Decisionware Database Gateway Interface for DB2 203
ORACLE 208
SQL Server 221
SQLBase 227
Sybase Net-Gateway Interface for DB2 233
Sybase SQL Server System 10 238
XDB 244
preparing ODBC data sources
about 23
Access 42
Btrieve, through INTERSOLV Btrieve ODBC driver 47
Btrieve, through Microsoft Btrieve ODBC driver 57
dBASE, through INTERSOLV dBASE ODBC driver 63
dBASE, through Microsoft dBASE ODBC driver 71
Excel 78
FoxPro, through Microsoft FoxPro ODBC driver 84
NetWare SQL, through INTERSOLV NetWare SQL ODBC driver 91
Paradox, through INTERSOLV Paradox 4 ODBC driver 96
Paradox, through INTERSOLV Paradox 5 ODBC driver 103
Paradox, through Microsoft Paradox ODBC driver 110
Rdb, through DEC Rdb ODBC driver 115
text, through Microsoft Text File ODBC driver 120
Watcom SQL 131
primary indexes *see* indexes, creating unique
primary keys, defining with IBM DRDA interface 163
procedures, basic
defining ODBC data sources 32
responding to connection prompts 273
selecting a database profile 271
setting database preferences 302, 374
setting DBParm parameters 302, 303

procedures, basic (*continued*)
starting Database Trace 398
starting ODBC Driver Manager Trace 410
steps for connecting 2
stopping Database Trace 404
stopping ODBC Driver Manager Trace 411
using Powersoft database interfaces 143
profiles, database *see* database profiles
ProfileString function
setting DBParm parameters in scripts 308
starting Database Trace in scripts 402
prompts, connection
responding to for all databases 273
responding to for IBM DRDA interface 182
PWD (password) value, in ODBC connect string 319
PXENGWIN.DLL file, required for INTERSOLV Paradox 4 driver 95, 96

R

Rdb, connecting through DEC Rdb ODBC driver
connection components 114
database preferences 380
DBParm parameters 310
DEC ODBC Driver Setup dialog box, values for 116
defining 115
preparing to use 115
SQL operations supported 113
versions supported 113
ReadOnly database preference 269, 375, 388
Recovery DBParm parameter 354
Release DBParm parameter
SQL Server 218, 355
XDB 242, 356
reports, creating with ORACLE 7 stored procedures 216
repository
about 247, 266
access rights, ensuring 248, 266
contents 268
controlling access 268, 386, 388
creating in DB2 databases 248
displaying 267
granting permissions on 270
table owner, specifying 351
Request DBParm parameter 357
requests, stored 389

S

scalar functions, supported in IBM DRDA interface 166
SCHEMA.INI file 118, 123
scripts, PowerBuilder
IBM DRDA, connecting to 182
INFORMIX, connecting to 199
setting DBParm values 307
starting Database Trace 401
Scroll DBParm parameter 358
scrolling options, cursor
INFORMIX 358
ODBC 326
SQL Server 327
Select Indexes dialog box, values for
Microsoft dBASE ODBC driver 75
Microsoft FoxPro ODBC driver 88
Select Tables list, modifying 163, 365
shadow catalogs, creating in DB2 databases 364
shared database profiles
displaying 297
maintaining 298
setting up 294
shared Paradox data sources
accessing through INTERSOLV Paradox 4 ODBC driver 97
accessing through INTERSOLV Paradox 5 ODBC driver 104
SharedIni variable
setting in INI files 294
setting in Preferences painter 295
short data types, SQL Server 219
Show system tables checkbox 267
single-tier ODBC drivers 18
SQL conformance levels for ODBC 20
SQL Data Definition Language (DDL) statements 382
SQL Data Sources dialog box 275
SQL files
DB2SYSPB.SQL 248, 352
PBOR7CAT.SQL 215
PBSYB.SQL 251

SQL files (*continued*)
PBSYBRT.SQL 252
PBSYC.SQL 252
SQL operations, supported in ODBC drivers
Access, Microsoft 40
Btrieve, INTERSOLV 45
Btrieve, Microsoft 55
dBASE, INTERSOLV 61
dBASE, Microsoft 69
FoxPro, Microsoft 82
NetWare SQL, INTERSOLV 89
Paradox 4, INTERSOLV 94
Paradox 5, INTERSOLV 101
Paradox, Microsoft 108
Rdb, DEC 113
Text File, Microsoft 118
Watcom SQL 129
SQL Server database interface
application name, specifying 312
connection components 221
cursor locking options, specifying 323
cursor scrolling options, specifying 327
data types supported 219
database preferences 380
DB-Library cursor processing 218, 355
DBParm parameters 311
defining 224
host name, specifying 341
installing Powersoft stored procedures 250
logging text and image updates 347
Login dialog box 276
preparing the database 221
required files 222
text field, specifying length returned 335
versions supported 218
SQL*Net client software
and ORACLE connect string or descriptor 212
required for ORACLE connection 209
SQLBase database interface
connection components 227
data types supported 226
database preferences 380
DBParm parameters 311
defining 229
lock values 385
preparing the database 227
SQLBase database interface (*continued*)
required files 228
transaction log creation, specifying 354
SQLCA transaction object 307, 402
SQLCache DBParm parameter 359
SQLQualifiers DBParm parameter 362
SQLReturnData attribute
and INFORMIX databases 199
determining SQL cache size 362
SQLSTATE error prefix, suppressing display 349
Startup Options dialog box, values for 137
startup options, specifying for Watcom SQL databases 135
stored procedures, ORACLE 7
about 214
creating 215
creating DataWindows and reports 216, 353
limitations when using 217
setting up the database server 215
see also PBDBMS DBParm parameter; PBDBMS package
stored procedures, Powersoft
about 250
created by PBSYB.SQL script 251
created by PBSYBRT.SQL script 252
created by PBSYC.SQL script 253
installing in SQL Server databases 250
ISQL utility, using to install 256
PBSYB.BAT file, for running PBSYB.SQL 258
WISQL utility, using to install 255
stored procedures, SQL Server 382
StoredReqOwner database preference 389
subject of this manual xiii
Sybase Net-Gateway Interface for DB2
connection components 232
data types supported 231
database preferences 380
database stored requests owner, specifying 389
DB2SYSPB.SQL script, using 247
DBParm parameters 311
defining 233
preparing the database 233
table and column name qualification 362
table list, modifying 370

Sybase Net-Gateway Interface for DB2 (*continued*)
tablespace, specifying 390
versions supported 231
Sybase Net-Library software 238
Sybase Open Client software 238
Sybase SQL Server *see* SQL Server database interface
Sybase SQL Server System 10 database interface
application name, specifying 312
character set, specifying 317
connection components 237
cursor blocking factor, specifying 316
data types supported 236
database preferences 380
DBParm parameters 311
declaring cursors 329
defining 240
host name, specifying 341
installing Powersoft stored procedures 250
language, specifying 346
logging text and image updates 347
preparing the database 238
required software 238
versions supported 235
sybsystemprocs database, Sybase SQL Server System 10 253
SYSIBM, prohibited as DB2 table owner 249, 352
System 10, Sybase *see* Sybase SQL Server System 10 database interface
system tables, DBMS 267, 363
system tables, Powersoft *see* repository
SystemOwner DBParm parameter 363

T

TableCriteria DBParm parameter
IBM DRDA 365
Micro Decisionware Database Gateway Interface for DB2 365
ODBC 366
ORACLE 368
Sybase Net-Gateway 370
XDB 365
tables
controlling access 388
enclosing names in double quotes 336
in repository 268
repository, creating in DB2 databases 247
Select Tables list, modifying 163, 365
system tables, displaying 267
TableSpace database preference 390
TerminatorCharacter, changing for ORACLE 7 stored procedures 215
text, connecting through Microsoft Text File ODBC driver
connection components 119
database preferences 380
DBParm parameters 310
Define Text Format dialog box, values for 124
defining 121
file format, defining 123
fixed-length files, defining format 127
ODBC Text Setup dialog box, values for 122
Parse dialog box 128
preparing to use 120
SCHEMA.INI file 118
SQL operations supported 118
text databases, structure of 118
versions supported 118
time data type, INFORMIX 190
Time DBParm parameter 372
tracing database connections
about 394
Database Trace 395
ODBC Driver Manager Trace 409
sample output, Database Trace 407
sample output, ODBC Driver Manager Trace 412
see also Database Trace; ODBC Driver Manager Trace; PBSQL.LOG file; PBTRACE.LOG file
transaction objects 307, 402
transaction processing 381
troubleshooting database connections
about 394
Database Trace 395
IBM DRDA 187, 414

troubleshooting database connections (*continued*)
ODBC Driver Manager Trace 409
see also Database Trace; ODBC Driver Manager Trace; PBSQL.LOG file; PBTRACE.LOG file

U

UID (user ID) value, in ODBC connect string 319
unique indexes *see* indexes, creating unique

V

validation rules, in repository 268
vendor DLLs, in Powersoft database interface connections 142
Vendors list, in INI files
list of supported DBMS identifiers 149
where DBMS identifiers come from 148, 274

W

Watcom SQL ODBC Configuration dialog box
example 34
values for 133
Watcom SQL, connecting through Watcom SQL ODBC driver
about 4
adding functions to PBODB040.INI file 422
comparing local and network connections 138
connection components 130
creating configurations and database profiles 10
database preferences 380
DBParm parameters 310
defining 131
lock values 385
preparing to use 131
SQL operations supported 129
Startup Options dialog box, values for 137
startup options, specifying 135
Watcom SQL, connecting through Watcom SQL ODBC driver (*continued*)
versions supported 129
Watcom SQL ODBC Configuration dialog box, values for 133
WIN.INI file, setting InitPath040 variable 298
WISQL utility, for installing Powersoft stored procedures 255
worksheets, Excel 29, 78

X

XDB database interface
connection components 243
data types supported 242
database preferences 380
DB2SYSPB.SQL script, using 247
DBParm parameters 311
defining 245
lock values 385
preparing the database 244
table list, modifying 365
versions supported 242, 356