

About Power SQL Monitor

The Power SQL Monitor allows you to oversee and manage the Power SQL server. You can use the Monitor to:

- Check the number of users currently connected
- Disconnect users
- Start and stop the Power SQL server
- Allow or prevent new connections to the Power SQL server
- Set the users allowed to connect to the Power SQL server
- Set up options to improve performance
- Log information about the users connecting to the server
- Set up the encryption and data compression used by Power SQL

For more information about the Power SQL Monitor, see [Power SQL Monitor Overview](#).

Power SQL Monitor Overview

Power SQL allows you to share ODBC data sources across the Internet or an intranet, even if the database is not a Client/Server database. When you create an ODBC Power SQL data source on the client that points to a data source on the server, the client can access the data source just as if it were on their system. You do not need to rewrite or recompile old ODBC applications in order for them to be used across the Internet: whenever an ODBC function is called, it is passed to the server, and the results are passed back.

The Power SQL Monitor also allows you to monitor the number of connections to the Power SQL system, and change settings for Power SQL's security, data compression, caching, and logging.

The Power SQL Monitor allows you to start and stop the Power SQL Server, as needed for maintenance or other reasons. Because the Power SQL Server is a service, by default it will restart automatically when the computer is rebooted, even if no one logs in. You can stop the server, or stop accepting new connections through the File menu, by using the Stop Power SQL and Stop New Connections commands. Once you have made the modifications or upgrades needed (for example, if you were changing the port you were operating on, or updating the hardware), you can use the Start Power SQL command to restart the server, then use the Allow New Connections command to enable connections. These options provide you with a considerate way of taking the Power SQL Server down, by first preventing new connections, waiting until all existing connections are gone, then stopping Power SQL.

From Monitor's main window you can view information about the connections currently made to Power SQL.

The Connection List displays the user name, data source, and start time for each connection. You can disconnect users from Power SQL by selecting a user in the Connection List, opening the Edit menu, and selecting Disconnect User. You can also disconnect all users using the Disconnect All Users command. The [Connection Properties](#) command in the View menu displays information about the currently selected connection, such as how long it has been connected, and how long it has been idle.

The Bar Graph displays the number of active connections and the maximum number of connections currently allowed.

Note: Gathering information about the active connections does make connecting to Power SQL slower when the Monitor is open. We suggest that unless you need to continually monitor user connections, that you use the Monitor only for changing settings, disconnecting users, and monitoring performance.

See Also

Editing the Users List	How to configure the list of users authorized to connect to the Power SQL server.
Settings	How to change Power SQL settings which include connection limits, the port number, encryption and data compression methods, security, caching performance enhancements, and connection logs.
Connection History	How to view the connection log and empty or remove items from the connection log.
Power SQL Database Password	How to change the password used to access the Power SQL Database.

Connection Properties

The Power SQL Monitor displays information about all currently-active connections. To display the Connection Properties, select a connection in the Connection List, open the View menu, and select Connection Properties. The Connection Properties dialog appears, displaying the following information:

User Name

The User Name used to log in to the database, if any was supplied.

IP Address

The IP Address of the Client for the connection. This value is found for every connection attempt, whether the connection is successful or not.

Data Source Name

The ODBC Data Source on the server to which the user connected.

Connect Time

The time and date on which the connection was opened.

Amount of Time to Connect (milliseconds)

The amount of time required to make the connection. This information can be useful when measuring the server performance.

Seconds Idle

The number of seconds that have passed since the last ODBC function call was made. Power SQL server has a setting that allows you to disconnect users that have been idle for a specific amount of time (this is useful for people who go on vacation without closing all of their applications). See [Settings](#) for more information.

Bytes Received

The number of bytes received from the client, including all ODBC calls, updates, and parameters.

Bytes Sent

The number of bytes sent to the client, including returns from ODBC calls and fetched records.

Database ID

The ID associated with the connection in the Database Log table (if logging to the database is enabled in the [Settings](#) dialog).

Event Log ID

The ID associated with the connection in the Event Log (if this option is enabled in the [Settings](#) dialog).

Editing the Users List

If your database does not have password protection, Power SQL can provide password protection for you. Enabling connection passwords in the [Settings](#) dialog requires users to enter their user name and password before they are allowed to connect. To edit the User List, go to the Power SQL Monitor, open the Edit menu, and select the Edit Users command. The Edit Users dialog displays a list of the current users and the following commands:

Add

Pressing the Add button displays the [New User](#) dialog, which allows you to specify a new user.

Edit

Selecting a user from the User List and pressing the Edit button displays the [Edit User](#) dialog, which allows you to edit information about the currently-selected user.

Delete

Pressing the Delete button deletes the currently-selected user from the User List, and invalidates the user name and password.

New User

The New User dialog is called when you select the Add button in the [Edit Users](#) dialog. You're prompted to enter the following information:

User Name

The user name for the account. If your database has password protection enabled, create user names and passwords that match those in your database.

Password

The password for the user name.

Confirm Password

Allows you to check against typographic errors in the password.

Edit User

The Edit User dialog is called when you select the Edit button in the [Edit Users](#) dialog. You can change the following information:

User Name

The user name for the account. If your database has password protection enabled, create user names and passwords that match those in your database.

Password

The password for the user name.

Confirm Password

Allows you to check against typographic errors in the password.

Power SQL Settings

You can configure the entire Power SQL system from the Settings dialog. The settings allow you to limit the number of connections, set timeouts, choose the encryption method, set security levels, and determine caching and logging behavior. You access the Settings dialog by selecting the Settings command in Power SQL Monitor's View menu. You can change the following settings:

Max Connections

You can specify the maximum number of connections permitted at any one time. This is very useful to prevent the server from becoming overloaded.

Max Statements Per Connection

Setting the maximum number of statements allowed on each connection prevents users from accidentally or deliberately bringing down the server by allocating a huge number of statements. If you do not want a limit, set this to zero.

Port Number

This setting changes the port number assigned to the Power SQL server. By default, the port number is 4545. For important information about changing port numbers, see [Port Numbers](#).

Timeout (minutes)

The Timeout is used to prevent resources from being used on inactive connections (which can occur when someone leaves an application connected to the server open for an extended period of time). When a connection is closed due to a timeout, all pending transactions are rolled back, and all statements and connections are closed. Users attempting to use the connection get an error message indicating that the connection was closed. To indicate that you do not want connections to time out, enter zero.

Max Packet Size (bytes)

This setting determines the maximum amount of data that can be transferred for an ODBC function call. It is most useful for preventing someone from accidentally transferring huge amounts of data, such as a two-gigabyte binary column. Typically, you should set the maximum packet size to at least 32k. The minimum size allowed is 512 bytes to ensure that all ODBC calls can go through.

Encryption DLL

The Encryption DLL is used on both the client and server side to encrypt and compress messages. If you use encryption on the server, all clients must use the same encryption DLL or they will be unable to connect. Power SQL comes with a sample Encryption DLL called PSCRYPT.PCR. Also, for better security, any properly constructed DLL can be used to encrypt and compress the messages. See [The Encryption DLL](#) for information about building your own encryption DLL.

Edit Users

Pressing the Edit Users button displays the [Edit Users](#) dialog, which allows you to add and delete users from the Power SQL user list, and edit user information.

Security

If one of your data sources does not provide user authentication, you can use one of the following levels of security provided by Power SQL:

Require Password to Connect	Requires a user name and password in order to connect to the data source. If the data source has its own authentication settings, you must make sure that the user name and password is the same for both the database and Power SQL.
Allow Data Source Listing	Allows data source listings without requiring a password.
Data Source Listing w/ Password	Requires a user name and password from your user list in order to list the data sources available on the server. The data sources are listed from the Setup dialog in the ODBC Administrator, or from the dialog used in SQLDriverConnect.
No Data Source Listing	Prevents users from viewing the data sources on the server.

Caching

The following caching settings determine how data is transferred from the server to the client. If all caching is off, every ODBC function call is sent individually to the server for a result. Turning caching on allows the server to gather commonly-requested information and send it together to reduce the number of messages sent across the network. For more information, see [Data Caching](#).

Column Info	Enabling this setting caches information about the result set for each statement executed.
Get Data	Enabling this setting uses the first set of calls to SQLGetData as a map to indicate which columns to cache. BLOB and MEMO (data over 255) are not cached. See Data Caching for important information about Get Data caching.
Data Dictionary	Enabling this setting uses Get Data caching on Data Dictionary result sets as well. See Data Caching for important information about Data Dictionary caching.
Get Info	This setting retrieves several of the most common information types from SQLGetInfo when

the user connects.

Get Functions

Caches information about the functions supported by the driver when the user connects.

Processes vs. Threads

This setting determines whether to use threads or processes for new connections. Unless your database driver doesn't support threads (most databases, including Access, do), you will want to use threads because processes are much slower and use a lot more memory. The calls to SQLConnect, SQLBrowseConnect, SQLDriverConnect, and SQLDisconnect are wrapped with a critical section to prevent them from being called simultaneously (which can give some database drivers problems).

Logging

These settings determine which data is logged, and where it is logged. Logging can be very useful for determining the amount of traffic on the Power SQL Server, and checking for attempted security violations. You can set the following logging options:

- | | |
|---|--|
| Log To Database | Stores all logged information in the Power SQL database . |
| Log to Event Log | Stores all logged information in the Windows NT Event Log. This can rapidly fill up the event log. |
| Log Successful Connections | Logs all connections. While logging all connections provides some very useful information, it also tends to rapidly fill up the database or the event log. This will also increase the amount of time required for a connection. |
| Log Unsuccessful Logon Attempts | Logs all failed connection attempts. The information logged includes the IP address of the individual attempting to log on, which can be very useful for detecting break-in attempts in high-security databases. |
| Log to Database at disconnect only (faster) | When Log Successful Connections is on, causes connection information to be written to the database when the user disconnects instead of when the user connects, which makes connections faster. |
| Connection Count Logging (Log Interval) | Logs the number of connections made during a time interval, the peak number of connections, and the number of failed connections. |

For more information about logging, see [Logging Connection Information](#).

Port Numbers

To connect to a program on a different computer, you need to know which port is assigned to the program. Generally, well-known programs such as Ping and web pages have a preassigned port that relieves the user from worrying about what port to connect with, or from even being aware that a port exists. For Power SQL, the default, and expected port is 4545. In the Client ODBC Setup dialog, new servers have a default port already set to 4545.

To avoid any possible conflicts with present or future programs, you can change the port used by Power SQL using the [Settings](#) dialog. However, there are some guidelines you should follow when choosing your port. First, port numbers under 1,000 are reserved by the operating system. Unless you are logged on as an administrator, you may not be able to access them. Second, while Windows uses a long integer to hold its port number, ports are usually assumed to be contained in a short integer to assure compatibility with other operating systems, and so are usually less than 32,767. If you change the port number, you need to restart Power SQL in order for it to take effect. You can stop and restart Power SQL from the File menu in the Power SQL Monitor (select Stop Power SQL, then select Start Power SQL).

Finally, after you change the port, you need a way to inform your clients about the new port number. The simplest way to do this is to define your server in the Power SQL Client INI file so that it becomes a name on the server list. See [Installing a Power SQL Client](#) for more information about generating the INI file.

The Encryption DLL

The Encryption DLL allows you to encrypt and compress your data before sending it through the Internet (or intranet). It is a fairly simple DLL that uses two functions, Encrypt and Decrypt. You set the Encryption DLL in the [Settings](#) dialog in the Power SQL Monitor, and in the ODBC Setup dialog on the client. These Encryption DLLs must match for any connection to be made. A sample encryption DLL called PSCRYPT.PCR is included with Power SQL: however for more security Power SQL was designed so that it is quite simple to plug in your own DLL, with any level of security desired.

Designing your own Encryption DLL

Power SQL includes sample code for a simple Encryption DLL which serves as a template to help you create your own. The encryption sample contains three functions in a file called PSCrypt.cpp to perform memory reallocation and freeing. These functions are designed to allocate memory in the manner that Power SQL expects, and thus are required, and should not be modified. The only other requirement for the encryption DLL is two exported functions named 'Encrypt' and 'Decrypt', and that have the same parameters and return value as the sample. These functions present you with a block of binary data that you can encrypt, compress, enlarge, shrink—whatever you require. You can find the sample encryption DLL in the PowerSQL\Samples\Encrypt directory.

Data Caching

The caching settings (set in the [Settings](#) dialog) determine how data is transferred from the server to the client. When all caching is off, each ODBC function call is sent individually to the server for a result. When caching is on, the server can gather commonly-requested information and send it together in a single data packet to reduce the number of messages sent across the network.

ODBC clears a previous function call's error and warning messages when the next ODBC function call is received. For this reason, caching will not occur when there is an error, to avoid erasing error messages. If there is a warning, caching continues, and the warning message is erased. Usually, you don't need to see the warning messages: however, if you do, you need to turn off caching in that area.

For the best performance, turn on all levels of caching. However, one exception to this rule may be Get Data caching. Get Data caching is performed at two levels called Get Data and Data Dictionary caching. Both of these cache the same information, but Data Dictionary only works on result sets from the Data Dictionary functions, while Get Data caching works on everything else. Caching SQLGetData calls significantly improves performance on applications using SQLGetData (many released applications and utilities use SQLGetData instead of SQLBindCol). In some rare cases, Get Data caching may break the application in some way because of the interaction between certain database drivers and some applications. Get Data caching is described below, along with the type of use that may cause problems.

Because SQLGetData is often used to fetch only a subset of columns from a result set, Get Data caching does not automatically fetch all columns. The first record fetched is used as a pattern, and any subsequent calls to SQLGetData are recorded, including the types of all variables, the size of the data retrieved, and which columns are used. On some databases, SQLGetData must be called in column order, meaning that you may retrieve column 1, 3, 6, but not column 2, 1, 6. For this reason, Get Data caching retrieves all columns in column order until the first column containing BLOB or MEMO data (over 255 bytes) is retrieved, or until an error occurs. So, for example, if your table contains Product Name[64], Product Price (double), and Product Picture (BLOB) columns, SQL GetData caches the first two columns, but not the third. If your application were to call SQLGetData on the second column, and give a type of SQL_DATE (which causes an Invalid Conversion error), only the first column could be cached. The other two columns would not be cached, and would get their data from the server whenever called.

If your application gets a consistent set of columns that use consistent types, Get Data caching always works. Caching also works if your application occasionally gets more columns, as long as they are after the last regular column. On most database drivers, calling SQLGetData out of order on columns also works properly, in which case caching will never give you problems.

If you suspect that Get Data caching is causing problems, try turning both Get Data and Data Dictionary caching off. If this works, try turning only Get Data off, then try turning off only Data Dictionary. Because Get Data caching provides a significant performance increase, you should leave it enabled whenever possible. You can turn off Get Data caching from the client or the server. Generally, it is good practice to leave Get Data caching on for the server, and only turn it off for the client data sources where it causes problems.

Logging Connection Information

Power SQL Logging stores information about the connections made to the Power SQL server. From the [Settings](#) dialog, you can specify several different types of logging and decide where to store the logged data. You can view the logs from the Power SQL Monitor by opening the View menu and selecting Connection Log.

Your first decision is where to store the logged information. You can store it in the Event Log, in the [Power SQL database](#), or both. Storing logged information in the Windows NT Event Log places all log entries in the Application Log, which may make it hard for you to find log entries from other applications. Another possible difficulty is that the Event Log will not contain all of the fields that are stored if logging to the database. However, an advantage of using the Event Log is that you can automatically delete log entries after a certain length of time, which helps reduce the storage size. We suggest that you log to the database because of its storage capability and its ability to store more information about the connections. All log entries are placed in one of two tables, either USERLOG or TIMELOG.

You can choose to log information about all connections, about unsuccessful connections only, or log summary information about the number of connections made during a time interval. Logging information about all connections can rapidly fill up your log unless you receive a small number of connections. Logging information about unsuccessful logon attempts is useful because it can help you detect problems with the database, or detect an individual attempting to guess a password. The information logged includes the reason why the logon attempt failed, including the error message from the database, if any. Logging the number of connections at specific time intervals shows you the number of users connecting, or failing to connect.

The Power SQL database places all log information into one of two tables, USERLOG or TIMELOG. The USERLOG table stores all log entries for successful and unsuccessful connections, and contains the following fields:

ID	A unique identifier for the record, which is a long integer.
UserName	The user name used to connect to the data source. This field is empty if no user name is provided.
ConnectTime	The date and time the connection was made.
DisconnectTime	The date and time the user disconnected from the server.
BytesReceived	The number of bytes received from the client, including ODBC function calls, parameters, and updates.
BytesSent	The number of bytes sent to the client, including ODBC return values and records.
IPAddress	The IP Address of the client machine.
DataSourceName	The ODBC data source on the server to which the user attempted to connect. If the client received a list of data sources in the Setup or SQLDriverConnect dialog, this is SQLDataSources.
Status	Indicates the final status of the connection. If the user connected and disconnected successfully, this value is zero. If the connection failed, one of the following values is shown: 1 Unable to connect because the user name and password were not found in the Power SQL user list. 2 Unable to list data sources because the user name and password were not found in the Power SQL user list. 3 Unable to connect because Power SQL server is not running. 4 Unable to connect because the maximum number of connections was reached. 5 Unable to list data sources, data source listing was not allowed in the Settings dialog. 101 Unable to connect because of an ODBC error. The Error message from ODBC is listed in the ErrorMessage column. 201 The connection was closed due to a timeout on the server side. This means that the connection was idle longer than the timeout set in the Settings dialog. 202 The connection was manually closed from the Power SQL Monitor using the Disconnect User command in the Edit menu.
TimeToConnect	The amount of time, in milliseconds, used to connect to the data source.
ErrorMessage	Error messages from the database driver are stored in this field when an ODBC error occurs during a connection attempt.

The TIMELOG table contains log entries made at a specified time interval, with information about the connections made during that time period. The structure of the TIMELOG table is:

ID	A unique identifier for the record, which is a long integer.
Time	The date and time when the timelog entry was made.
MaxConnections	The highest number of connections active at one time during the time interval.

NewConnections The number of new connections made during the time interval.

FailedConnections The number of connection attempts which failed during the time interval.

To view the logs and empty full log tables from the Power SQL Monitor, select Connection Log from the View menu.

Note:

While the original Power SQL database is provided in an Access format, you can create your own database using a different database manager. For more information, see [Power SQL Database](#).

Connection History

From the Power SQL Monitor, you can view the Power SQL log and delete entries from the log as it gets full. To view the Connection History dialog, open the View menu, and select Connection Log. The Connection History dialog appears, with the following tabs:

Connection Log

This log contains information about individual connections. Depending on the values chosen in the [Settings](#) dialog, the connection log records all connections or all failed connections, which is the default setting. For each connection, the Connection Log stores information about the user, the amount of time connected, the amount of data transferred, and any error codes or messages from ODBC.

Connection Count Log

The Connection Count log contains summary information about the number of connections made or attempted during a specific time period (as set in the [Settings](#) dialog). The information includes the maximum number of concurrent connections (the peak load), the total number of new connections, and the number of failed connection attempts during each time period. This information can be very useful for monitoring the server load and attempted security break-ins.

You can delete log entries from both of these logs by selecting the log entry or entries to be deleted and pressing the Delete key. For more information about the Log tables, see [Logging Connection Information](#).

Power SQL Database Password

From the Power SQL Monitor, you can change the user name and password used to connect to the [Power SQL Database](#). To change the password, open the View menu and select Configure Security. The Power SQL Database Password dialog appears, allowing you to set the following options:

User Name

The user name to use to log on to the Power SQL database. This account is used not only by the Monitor, but also by the Power SQL server when getting settings, validating users, and writing to the log.

Password

The password that accompanies the user name.

Confirm Password

Confirms the password you enter to safeguard against typing errors.

Require Login to use the Power SQL Monitor

Checking this option requires you to enter a valid user name and password every time you start the Power SQL Monitor. This option safeguards against unauthorized access to the Power SQL settings.

For the procedure to change the user name and password, see [Changing the User Name and Password in the Power SQL Database](#).

Setting up the Client

When you create an application using Power SQL, you need to distribute the Power SQL Driver to all of your clients as part of the installation process. There is no royalty attached to the Power SQL client side, so you can distribute the Power SQL Driver to as many Power SQL clients as you like. The Power SQL Driver has been designed to be easy to install, and has the following set of requirements for the client computer:

- Must be running Windows 95 or higher, or Windows NT 3.51 or higher
- Must have ODBC installed (this is part of the operating system for Windows 95 and Windows NT 4.0)
- Must have the Internet or intranet connection properly installed, with TCP/IP enabled, and an IP address assigned to the client computer
- Must have about 200K free on the hard drive containing the Windows System directory.

You can install the Power SQL Internet Driver in two short steps:

1. Copy POWERSQL.DLL, POWERSQL.HLP, POWERSQL.CNT, and MSVCRT.DLL to the system directory. The InstallDriver function in the Power SQL dll has been provided to properly install the driver.
2. Execute the install function by using LoadLibrary to load the PowerSQL DLL, getting the address (GetProcAddress) of the "InstallDriver" function, and calling it. This function registers the Power SQL driver with the ODBC Manager and names it "Power SQL Internet Driver."

If you want your server to be easy to find, especially if you have used a port other than the default (see [Port Numbers](#) for more information), you can create an alias for your server. See [The Server List](#) for more information.

Advanced Administrative Details

Following is a list of articles that describe some additional features of Power SQL. While Power SQL works properly and contains a high level of flexibility without these features, you may find them to be useful. Probably the most interesting is the first article, which describes how to create your own Encryption DLL for the Power SQL server.

[Creating an Encryption DLL](#)

How to create your own encryption and/or compression for the Power SQL server, and where to find the Encryption DLL template.

[Creating a new Power SQL database](#)

How to create a new Power SQL database using a different database management system.

[Changing the User Name and Password in the Power SQL database](#)

How to change the user name and password used by the Power SQL server.

[Handling memory leaks in ODBC Database Drivers](#)

How Power SQL handles memory leaks in the ODBC database drivers, and how to configure this feature.

The Power SQL Database

The Power SQL server uses a database to store all of the Power SQL settings, the user list, and the two log tables, User Log and Time Log. The default database management system is Microsoft Access: to use a different database management system (DBMS), follow the steps below to create a new database, and replace the original.

Creating the Database

The first step is to create the database in the DBMS of your choice. If you want to create a new database using a different database management system, the DBMS must support the Auto Number type, a long integer which automatically increments on each new record. You need to create the following four tables, using these exact names and a similar data type:

The **SETTINGS** Table

The SETTINGS table contains most of the configuration settings used by Power SQL. It only contains one record that must contain the initial settings (for more information about the setting values, see [Settings](#)). You can copy the data for this record from your current database.

<u>Field Name</u>	<u>Data Type</u>	<u>Nullable</u>	<u>Default Value</u>
ID	AutoNumber Long Integer (4 bytes)	No	None needed, 1 will work
MaxConnections	Long Integer (4 bytes)	Yes	0
TimeOut	Long Integer (4 bytes)	Yes	0
MaxPacketSize	Long Integer (4 bytes)	Yes	32,767
PortNumber	Long Integer (4 bytes)	Yes	4545
bAutomaticStart	Long Integer (4 bytes)	Yes	0 - Reserved for later use
bAllowDSLlisting	Long Integer (4 bytes)	Yes	1 (TRUE)
Authentication	Long Integer (4 bytes)	Yes	3 (Authentication ON for connections and data source lists)
LogTo	Long Integer (4 bytes)	Yes	1 (Log to the Database)
LogWhat	Long Integer (4 bytes)	Yes	2 (Unsuccessful Connections)
MaxStmtsPerConnection	Long Integer (4 bytes)	Yes	50
CacheSettings	Long Integer (4 bytes)	Yes	31
bProcess	Long Integer (4 bytes)	Yes	0
LogTimeSpan	Long Integer (4 bytes)	Yes	60

The **TIMELOG** Table

The TIMELOG table stores the Time Interval log entries. For this table, you only need to provide the structure: records are not required.

<u>Field Name</u>	<u>Data Type</u>	<u>Nullable</u>
ID	AutoNumber Long Integer (4 bytes)	No
Time	Date/Time	Yes
MaxConnections	Long Integer (4 bytes)	Yes
NewConnections	Long Integer (4 bytes)	Yes
FailedConnections	Long Integer (4 bytes)	Yes

The **USERLIST** Table

The USERLIST table contains a list of Power SQL user names and passwords. If you are not using Power SQL authentication, this table may be empty: otherwise, fill it with your user name list. We suggest that you leave this table empty and fill it from the Power SQL Monitor after you have finished switching databases.

<u>Field Name</u>	<u>Data Type</u>	<u>Nullable</u>
ID	AutoNumber Long Integer (4 bytes)	No
UserName	char[50]	Yes

PassWord	char[50]	Yes
----------	----------	-----

The **USERLOG** Table

The USERLOG table stores connection information. For this table you only need to provide the structure, as no records are needed.

<u>Field Name</u>	<u>Data Type</u>	<u>Nullable</u>
ID	AutoNumber Long Integer (4 bytes)	No
UserName	char[50]	Yes
ConnectTime	Date/Time	Yes
DisconnectTime	Date/Time	Yes
BytesReceived	double	Yes
BytesSent	double	Yes
IPAddress	char[20]	Yes
DataSourceName	char[50]	Yes
Status	Long Integer (4 bytes)	Yes
TimeToConnect	Long Integer (4 bytes)	Yes
ErrorMessage	char[255]	Yes

Installing the Database

The next step is to shut down the Power SQL system for a few minutes and install the database. Do the following steps:

1. Start the Power SQL Monitor.
2. Do one of the following:
 - If no users are currently connected, open the File menu, and select Stop Power SQL.
 - If some users are connected, disconnect them by opening the Edit menu, and selecting Disconnect All Users, then selecting Stop Power SQL from the File menu.
 - If some users are connected and you don't want to disconnect them, open the File menu, select Stop New Connections (this prevents anyone else from connecting), wait for everyone to disconnect, then select Stop Power SQL from the File menu.
3. Close the Power SQL Monitor, and the whole Power SQL system is stopped.

Next, you're ready to switch out the old database.

4. To list the ODBC system data sources, go to the ODBC Administrator in the Control Panel and press the System Data Sources button. You should see a data source called POWERSQL. You can either delete or rename it. We suggest that you rename it in case you need to use it again (press the Setup button, and type a new name into the Data Source Name edit control).
5. Create a new data source by pressing the Add button, selecting your database driver from the list, and filling in the data source description with the information for your new database. Name this data source POWERSQL (all caps, one word).

Testing the Installation

1. Start the Power SQL Monitor.
2. Do one of the following:
 - If the Monitor starts with no errors, skip to the next step.
 - If you get some error messages, or if Power SQL refuses to start up, check the error messages or the Application section in the Event Log for the nature of the error. The error messages should indicate what went wrong, which is most likely a name mismatch or type mismatch. Fix the error and re-start the Power SQL Monitor.
3. Start the Power SQL Server by selecting Start Power SQL in the File menu.
4. Log on from a remote machine to ensure that you can connect to the server.

Note:

If your new database has a password, be sure to [set the new password](#) before restarting Power SQL.

Changing the User Name and Password in the Power SQL Database

The [Power SQL Database](#) contains the settings, logs, and user list for your Power SQL server. To provide extra protection for this information, you can specify a user name and password to be used when accessing the Power SQL database. To add or change the user name and password, do the following steps:

1. Start the Power SQL Monitor.
 2. Do one of the following:
 - If no users are currently connected, select Stop Power SQL from the File menu.
 - If some users are connected, disconnect them by selecting Disconnect All Users from the Edit menu , then selecting Stop Power SQL from the File menu.
 - If some users are connected and you don't want to disconnect them, select Stop New Connections from the File menu (this prevents anyone else from connecting), wait for everyone to disconnect, then select Stop Power SQL from the File menu.
 3. Close the Power SQL Monitor, and the whole Power SQL system is stopped.
 4. In the database manager for the Power SQL Database, add a user name and password, or change the existing one.
 5. Start the Power SQL Monitor.
- If the old password is no longer valid, Power SQL prompts you to enter the new password. If not, go to the View menu and select [Configure Password](#) to set the new password.
6. Restart Power SQL by opening the File menu, and selecting Start Power SQL.

Handling Memory Leaks in ODBC Database Drivers

When working with a server, you expect to be able to leave it running for months on end without running low on resources or rebooting the server. However, some ODBC database drivers do have resource leaks, and even a very small and intermittent leak can build up over time and bring down the server. To prevent problems from resource leaks, PowerSQL accesses the database using one of two executables, and on occasion restarts the executables. PSSERVER.EXE handles all client requests from the database, and PSLOG.EXE handles all [logging](#) to the database and retrieval of the [settings](#).

When a certain number of connections has been made with either executable, a new copy of the executable is created, which begins handling all new connection requests. When the last connection in the old copy of the executable is closed, the old executable closes, causing Windows NT to free all resources associated with it, including any lost resources. If your Power SQL server is busy, or if someone maintains a connection for a long time, this could result in two or possibly three copies of PSSERVER.EXE. The spare executables close after the users log off or get disconnected by a timeout.

Depending on the ODBC database driver you are using, you may have no leaks, a few leaks, or many leaks. If you have a very clean and stable driver, or an abnormally leaky and unstable driver, the executables may be switched too often, or not often enough. There is some overhead involved in switching executables, so you should not do it too frequently. Ordinarily, you will not need to change the default values for the number of connections before switching executables, but there is a registry entry to allow this if you do.

To modify the rate at which either executable is switched, do the following:

1. Start the registry editor by running REGED32.exe (or REGEDIT.exe).
2. In the registry editor, go to the HKEY_LOCAL_MACHINE section, then go to the SOFTWARE\Blue Sky Software\Power SQL Internet Server section. Under this section, you may add or edit the following two keys:

<u>Key Name</u>	<u>Key Type</u>	<u>Default Value</u>	<u>Affect</u>
#PSLog	REG_SZ	1000	This key indicates how many log entries and user authentication verifications to handle before switching executables
#PSServer	REG_SZ	1000	This key indicates how many connections to handle before switching executables

Generally, the default should work well, so you only need to change it if you notice problems, or if you want to experiment with improving performance by increasing the number. The #PSLog key only impacts the log entries and user authentication on your [Power SQL Database](#). The #PSServer key only impacts the database requests submitted from the clients.

About Help Desk

Providing updated technical information over the Internet is just one example of how Power SQL can lower your product support costs by integrating a Help Desk database that can be updated on your Internet server. Blue Sky's Help Desk allows you to provide easy access to dynamic data to your end users—from within your Help system.

Including dynamic data access in your application can have a dramatic impact on your support costs and can ultimately improve user satisfaction. Research has shown that users do not like to search the Web for solutions to their technical problems. An online Help Desk can reduce the number of technical support calls which in turn, results in lower costs for maintaining and supporting your applications. Blue Sky Software makes it possible to add solutions to technical problems right in the application—through a Help Desk database that provides answers to your users' questions where they expect to find them—in the Help system.

Your application's Help Desk database can be maintained on-the-fly. When new problems and solutions arise, your technical support professionals can enter the latest information and maintain the database right on your server. Then, the next time someone has the same problem, the latest solution is available in the application's Help system.

If fact, you can send questions or feedback to "powersql@blue-sky.com" and it will be entered in the Help Desk database on our server. Your questions and our solutions will be in this Help system in a matter of hours.

Blue Sky Software will be releasing a beta version of the Help Desk authoring tool soon. If you are interested in testing beta versions of this product, please send email to "helpdesk@blue-sky.com".

[View the Help Desk](#)

Power SQL Help Desk

```
{ewl actxwh.dll, ActiveWinHelp, CLSID="E9B53FC3-761E-11D0-AF02-00009290C4DB" ALTERNATE="Unable_To_Load"  
CODEBASE="HELPDESK.OCX"  
PROPERTIES="00000100182e0000de1d000000000000a74452495645523d7b506f7765722053514c20496e7465726e6574204  
472697665727d3b5053514c5352563d7777772e7673716c2e636f6d3b5053514c49503d3230372e3133372e302e3233343b5053  
514c504f52543d343534353b5053514c44534e3d506f7765722053514c2048656c70204465736b3b5053514c454e43523d433a5c  
57494e444f57535c53595354454d5c707363727970742e7063723b5053514c544f3d303b207777772e7673716c2e636f6d5c506f  
7765722053514c2048656c70204465736b03000000000083c204e6f6e65203e00"}}
```

Unable_To_Load

Help Desk was unable to load.

[Return](#)

Power SQL

Power SQL allows you to share ODBC data sources across the Internet or an intranet, even if the database is not a Client/Server database. When you create an ODBC Power SQL data source on the client that points to a data source on the server, the client can access the data source just as if it were on their system. You do not need to rewrite or recompile old ODBC applications in order for them to be used across the Internet: whenever an ODBC function is called, it is passed to the server, and the results are passed back.

The Power SQL Monitor allows you to monitor the number of connections to the Power SQL system, and change settings for security, data compression, caching, and logging.

Contents

<u>Power SQL Overview</u>	An overview of Power SQL, and the components that make up the Power SQL Server.
<u>System Requirements</u>	Describes the requirements for the computer on which the Power SQL Server is installed.
<u>Power SQL Data Source Setup</u>	Describes how to set up a Power SQL data source on the client to access a specific server, along with other information useful when setting up your client.
<u>Setting up Power SQL</u>	Describes how to set up Power SQL and the Power SQL data sources.
<u>The Power SQL Monitor</u>	Describes the Power SQL Monitor application, which is your main interface to Power SQL.
<u>Power SQL Settings</u>	Describes the settings available from Power SQL, including custom encryption, data caching, and logging.
<u>Developing Applications with Power SQL</u>	An overview of considerations when creating Internet-enabled applications with Power SQL data sources, along with information useful for simplifying the installation.
<u>Setting up the Client</u>	Describes how to set up an installation and set up initial files for your clients.
<u>Advanced Administrator Details</u>	Describes additional features of Power SQL that are useful to the Power SQL administrator or expert users.

Power SQL Overview

Power SQL allows existing [ODBC](#) applications to connect to databases across the Internet, or an intranet, even if neither the application nor the database was designed to communicate across a network. A developer can write an application using ordinary ODBC calls, test the application against a local database, then simply change the data source to use the Power SQL Internet Driver. The application can then access data anywhere on the Internet.

All you need to do is install Power SQL Client on the client and Power SQL Server on the server. The Power SQL Client makes it easy to define an ODBC Data Source that specifies a remote Power SQL server and any ODBC database on that server. Power SQL handles the complexities of communicating the ODBC function calls and data between the application and the database. Options on the Power SQL Server allow the administrator to configure data caching, security, encryption, logging, and other network and server issues.

Power SQL adds many features beyond simply sharing a database:

- All configuration issues for your database can be managed on one machine and upgrades to the database or database driver now only need to happen on one machine. You can even change your database to use a different database management system (DBMS) without any of your users needing to do anything at all.
- Power SQL adds password protection to databases which do not supply any form of security.
- For additional security, Power SQL allows you to add your own encryption of any type desired, rather than locking you into one type of encryption. You can use any level of encryption you want to. A simple default encryption DLL is included in Power SQL, if you do not need an exceptional level of protection.
- The Power SQL ODBC Driver is thread-safe, and allows you to use non-thread-safe drivers from multiple threads on the client side.
- Your application can use one central repository for information, such as for copy protection or licensing purposes.
- By creating an ActiveX control with Visual SQL or by hand, you can put your database on the Internet. This is particularly useful for information which frequently changes, or lists of information which may grow and shrink frequently. By using a form view or a datasheet, the data can change without needing to update the web page. This has several potential uses:
 - Technical Support Knowledge Base: as new items are found, they can be added to the web page by modifying the database.
 - Decision Support Systems: remote offices can access the information in the central database.
 - Traveling Salespeople: Leads can be entered, sales can be recorded to speed up shipping, and information about accounts can be accessed, with all of the data protection, concurrency support and transactions a true database can provide.
 - And anything else your imagination can dream up...

If you're ready to get started, you can: read below about [Developing Applications with Power SQL](#) and [The Power SQL Components](#); get more information about [ODBC](#); or begin [Setting up Power SQL](#).

See Also

[The Power SQL Monitor](#)

Describes the Power SQL Monitor application, your main interface to Power SQL.

[Power SQL Settings](#)

Describes the settings available from Power SQL, including custom encryption, data caching, and logging.

[Improving Performance](#)

Describes varying options that can improve or impede the speed of the Power SQL system.

[Setting up the Client](#)

Describes how to set up an installation and set up initial files for your clients.

[System Requirements](#)

[Power SQL Internet Driver Setup](#)

[Developing Applications with Power SQL](#)

ODBC

The Open Database Connectivity standard (ODBC) allows developers to use a common set of functions to access a wide array of databases. Over the years, ODBC has become more than a way to connect to different databases: it is now a method of programming that allows the programmer to learn one database language and be able to write applications for almost any database. In addition, once that application is written, it is not locked into one database. It is now relatively simple to develop with one database, then change your mind later, and use a different database. The ODBC database drivers have not only become faster, more powerful, and more stable, but ODBC is now the language of choice for accessing all databases in Microsoft's programming tools, Visual Basic, and the Visual C++ MFC classes.

The Power SQL Components

Following is a quick overview of the Client and Server components in Power SQL.

Client

POWERSQL.DLL	The Power SQL ODBC Driver. This must be registered with ODBC.
POWERSQL.HLP	The Power SQL help file.

Server

PSMONITR.EXE	The Power SQL Monitor .
POWERSQL.HLP	The Power SQL help file.
PowerSQL.mdb	The Power SQL Database .
PSCRYPT.PCR	The default encryption DLL .
PSMAIN.EXE	The Power SQL Service .
PSSOCKET.EXE	Part of the Power SQL server.
PSSERVER.EXE	Part of the Power SQL server.
PSCON.EXE	Part of the Power SQL server.
PSCON.DLL	Part of the Power SQL server.
PSLOG.EXE	Part of the Power SQL server.

System Requirements

The Power SQL Server

The amount of RAM, the CPU speed, and the amount of disk space required for the Power SQL Server depends on how much traffic you expect the server to receive. If you expect heavy traffic, you need to configure the server to handle it. If you expect light traffic, the hardware requirements for Windows NT will suffice.

The requirements for the Power SQL Server are as follows:

- The server must run on the Windows NT operating system version 3.51, 4.0, or later
- Approximately 8MB of hard drive space (less if ODBC does not need to be installed)
- The server must have TCP/IP configured, and an IP address assigned
- When running the installation, some registry keys need to be written. The user running the install should have sufficient permission to write to the Local Machine section of the registry, and to install a service (usually, this means that the user needs to be in the Administrator group).

The Power SQL Client

- Power SQL Client is supported on Windows 95, and Windows NT 3.51 or 4.0.
- Approximately 300 KB of hard drive space must be available.
- ODBC 2.5 or newer must be installed.
- TCP/IP must be configured, and an IP address assigned.

Setting up Power SQL

There are a few steps involved in setting up a Power SQL data source on the server so it can be reached by your clients. The following sections describe how to set up an ODBC data source and a few different options for configuring, starting, and stopping the Power SQL service.

[Improving Performance](#)

[Power SQL Data Sources](#)

How to create the ODBC data sources that your clients will access.

[Power SQL Service](#)

How to configure the service in the Control Panel, and some of the options you can change there.

[Finding Power SQL Setup Errors](#)

Where to find Power SQL error messages.

See Also

[Setting up the Client](#)

[Power SQL Monitor Overview](#)

[Power SQL Settings](#)

Improving Performance

You can affect Power SQL's performance by choosing different caching, logging, and monitoring settings. By default, these settings are intended to provide close to maximum performance. However, you can experiment with the following settings to further improve the performance of the Power SQL system.

Caching

Power SQL allows you to select the kind of information you want to be cached. Because most applications use all of the data cached, you will usually enable all forms of caching. Get Data caching in particular can make an enormous difference in your applications' performance. See [Power SQL Settings](#), and [Data Caching](#) for more information.

Logging

Logging is very useful for tracking the usage, performance, and even security of your Power SQL Server. However, it does increase connection times. To maximize performance, disable some of the information logged, such as the number of successful connections. Often a good compromise is to log information about failed connections, and the number of connections made every 6 to 24 hours. This gives you a reasonable amount of tracking information without sacrificing performance. You change the logging settings in the [Power SQL Settings](#) dialog.

Monitor

Because the [Power SQL Monitor](#) displays information about the current connections, it does slow down connections to Power SQL. Generally, you only want Monitor up to check performance or change settings.

Timeout

The connection Timeout setting for the [Power SQL Internet Driver](#) will have an effect on performance. The overhead required for the operating system to check for an expiring timeout slows down the connection of the client to the server. Also, normally the network software will have the ability to timeout unsuccessful connections. The Timeout should be set to 0 to avoid any decrease in performance.

Power SQL Data Sources

There are two ways to set up the data sources you want to share, depending on your [Power SQL Service](#) setup. If your service is set up as a System Account (which is the default), all shared ODBC data sources must also be system data sources. If you are logging the service in as a user account, then the data sources may be entered into the user account's data source.

The default is to run the Power SQL service under the System Account. The main benefit of a System Account is that only system data sources are listed on the client side, so you can have some private data sources. The only drawback is that some ODBC drivers may install needed components only into the current user account, and those database drivers cannot be used for system data sources without editing the registry.

To add a system data source, do the following steps:

1. Start the ODBC Administrator by opening the Control Panel (in WinNT 3.51, it's in the Main program group; in WinNT 4.0, it's in the Start menu under Settings) and double-clicking on the ODBC or 32-bit ODBC Administrator icon.
2. Click on the System DSN... button to bring up the System Data Sources list.
3. From here, click on the Add button, select the desired database driver, and configure your data source like you normally would.

If you have configured the [Power SQL Service](#) to log in under a user account, you must be logged into that account before you create your data source. There are two benefits to using a user account. First, you can use normal user data sources for any database drivers that cannot be used as system data sources. Second, you can create a separate user account for Power SQL in order to manage the data sources the client is able to see, and the database drivers installed on that account. One of the main drawbacks is that the user account setup for a service is harder to configure, since the user account must have a very high level of permissions to run properly.

To configure the data source as a user account, do the following steps:

1. Log in to the correct user account.
2. Open the ODBC Administrator by opening the Control Panel (in WinNT 3.51, it's in the Main program group; in WinNT 4.0, it's in the Start menu under Settings), and double-clicking on the ODBC or 32-bit ODBC Administrator icon.
3. Click on the Add button, select the desired database driver, and configure your data source.

To change the service from System Account to a user account or back again, see the [Power SQL Service](#) topic.

Power SQL Service

The Power SQL Service starts up the Power SQL Server, and begins listening for messages from the client. You can modify how the Power SQL Service starts up from the Control Panel Service manager. You can also start and stop the Power SQL Server either from the Service manager, or from the [Power SQL Monitor](#).

To start the Service Manager, do the following steps:

1. Open the Control Panel (in WinNT 3.51, it's in the Main program group; in WinNT 4.0, it's in the Start menu under Settings), and double-click on the Services icon.
2. Scroll down the Service list and select Power SQL.
3. Click on the Startup button to change the properties of the Power SQL Service.

You can select from among three Startup options. Automatic startup, the default, means that the service restarts every time the computer is turned on, even if no one logs on. Automatic startup is good for servers that can run without any intervention. If the power goes out and comes back on, Automatic startup causes the Power SQL Server to restart automatically.

Manual startup means that the service must be started from the Service Manager or from the [Power SQL Monitor](#) any time you want to use the Power SQL Server. This option is useful if you only want the Power SQL Server to be active at certain times. Use disabled startup only when you want to ensure that the Power SQL Server does not start.

The Log On As setting allows you to decide how the service should operate. Most services work from the System Account, but sometimes (see [Power SQL Data Sources](#)), you may want the service to work from a particular user account, in which case you need to browse to select the user account and provide the password for that account. Power SQL then runs in that user account using the path setup, and works with the ODBC drivers and data sources set up on that account only. Even if someone logs on to a different account, Power SQL continues working with the options from that account. The user account selected requires a very high level of permissions on the server, usually the user must be an administrator of the machine to work properly. When possible, use the system account instead.

Finding Power SQL Errors

Because server errors can go undetected—sometimes for an extended period of time—Power SQL logs all errors in the Application section of the Event Log so you can monitor your server's performance.

To view the Event Log:

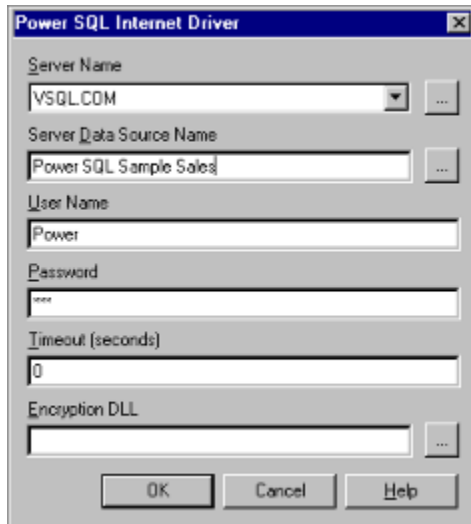
1. Go to the Administrative Tools program group and open the Event Viewer.
2. To view the Application log, select Application from the Log menu. The icons beside the log entries indicate the severity of the message. Errors are indicated by a red stop sign.
3. To get more information about a particular status or error message, double-click on the log entry.

Note:

When a [Power SQL Monitor](#) error occurs, a message box appears notifying you to look in the Event Log.

Power SQL Internet Driver Setup

The Power SQL Internet Driver allows you to connect to another database across the Internet (or an intranet), without requiring any network capabilities from that database engine. The Setup dialog enables you to configure a user data source that will point to a database anywhere on the Internet. You can then access it, modify it, and treat it as if it is on your computer.

The image shows a Windows-style dialog box titled "Power SQL Internet Driver". It contains several input fields and buttons. The "Server Name" field is a combo box with "VSQL.COM" selected and a button to its right. The "Server Data Source Name" field is a text box with "Power SQL Sample Sales" and a button to its right. The "User Name" field is a text box with "Power". The "Password" field is a text box with "none". The "Timeout (seconds)" field is a text box with "0". The "Encryption DLL" field is a text box with an empty space and a button to its right. At the bottom are three buttons: "OK", "Cancel", and "Help".

Server Name

The Server Name indicates the Power SQL Server where your data source is located. This server name is strictly a logical name, it is not necessarily the server domain name. Select one from the list, or use the button beside the combo box to manage the [Power SQL servers](#) (add a new server, edit a server, or delete a server).

Server Data Source Name

The Server Data Source Name is the ODBC Data Source you will connect to on the selected server. You can type the data source name or [browse through the available data sources](#) by pressing the button beside the Server Data Source Name after selecting a server in the Server Name combo box. The Power SQL Server may be configured to require a user name and password before allowing a list of data sources to be found. If you get an error indicating an invalid user name or password, specify the user name and password in the text boxes. Some Power SQL Servers may be configured so that a data source listing is not provided to the client. The client must know the name of the data source.

User Name

The User Name is the user name to be used when logging on to the selected data source. This user name may also be used by the Power SQL Server, if it has been configured to require user names. Typically, Power SQL Server will be setup so that any user names required for logging into the Power SQL Server are the same as those used by the database.

Password

The Password is used with the User Name to log on to Power SQL and the selected data source.

Timeout

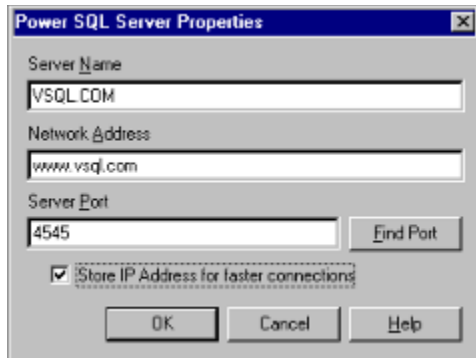
The Timeout is the number of seconds to wait for a connection to be successful. If the Timeout is set to zero (the default), there is no limit to the amount of time to wait. Performance may be slightly better with the timeout is set to zero.

Encryption DLL

The Encryption DLL indicates the DLL used by Power SQL to compress and encrypt the data. If this is left empty, no encryption will be used. You may use either the encryption DLL supplied by Power SQL (PSCRYPT.PCR) or your system administrator may provide you with one.

Power SQL Server Properties

These properties of a Power SQL server are used to define the exact server to connect to. If you select a server that does not exist, you will be notified when exiting this dialog.

The image shows a Windows-style dialog box titled "Power SQL Server Properties". It contains three text input fields: "Server Name" with the value "VSQL.COM", "Network Address" with the value "www.vsql.com", and "Server Port" with the value "4545". To the right of the "Server Port" field is a button labeled "Find Port". Below these fields is a checkbox labeled "Store IP Address for faster connections", which is checked. At the bottom of the dialog are three buttons: "OK", "Cancel", and "Help".

Server Name

The Server Name is strictly a logical name, used when listing the available servers. It does not need to be the same as the domain name, network address, or IP address.

Network Address

The Network Address should contain the domain name of the server, such as www.blue-sky.com. It could also contain a local network name, such as MyServer, or an IP Address, such as 123.45.67.890. This is the server that Power SQL is installed on.

Server Port

The Server Port is the port that Power SQL is monitoring for new connections. The Port number is typically 4545. If not, ask the network administrator for the correct port.

Find Port

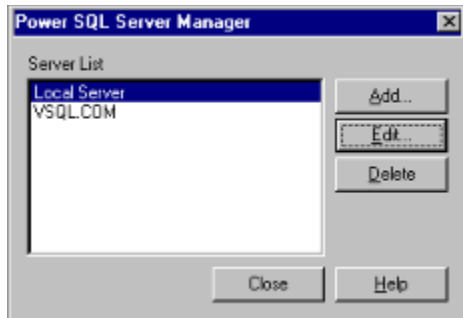
The Find Port button will look through nearby ports to try to find which one a Power SQL Server is monitoring. Due to the time required to check a port, this will only search through 60 ports near the current Server Port value.

Store IP Addresses for faster connections

Unless you expect the IP Address of the server to change frequently, your performance will be much better if this option is on. The IP Address of the server will be found, stored, and used whenever you attempt to connect. If connecting using the IP Address fails (if the server has a new IP address), the Network Address will be used to connect, and the new IP Address will be stored, and used next time you connect.

Power SQL Server Manager

Each Power SQL server represents a set of values which specify one particular server. You may add new servers, edit existing servers, or delete unused servers from this dialog box, or simply select the server that you want to use.



Server List

This lists the Power SQL servers that have been created. They are listed here so it's easy for you to connect several data sources to the same server.

Add

This brings up the [Power SQL Server Properties](#) dialog, to specify the properties for the new server.


Edit

This brings up the [Power SQL Server Properties](#) dialog, which allows you to edit the currently selected Power SQL server.

Delete

This deletes the currently selected server.

Power SQL Server Data Sources

The server Data Sources are displayed by pressing the  next to the “Server Data Source Name” field in the Power SQL Internet Driver dialog. The Data Sources dialog lists all of the data sources that are available on the Power SQL server. Some servers may be configured so that a user name and password is required to view the data sources, and some may not allow you to browse the data sources. This is configured using the Power SQL Monitor located on the Power SQL server.

Developing Applications with Power SQL

Application Development using Power SQL is generally the same as developing an ordinary ODBC application. All of the ODBC functions are supported, and very nearly all of the ODBC functionality. Either MFC CDatabase and CRecordset classes or ODBC SDK functions may be used, or any other method of using ODBC. No special libraries or header files are required, and the application can be an executable, DLL, OCX, or ActiveX control. The Power SQL Internet Driver is thread-safe. The sole feature of ODBC that is not supported is changing the Bind Type, a feature that is rarely used. Power SQL is completely compatible with using the Microsoft Foundation Class Library's database classes, as well as raw ODBC. Creating and deploying your database application using Power SQL has been designed to be as simple as possible.

Note: The Power SQL client can be distributed royalty free, so there are no limits on what the developer can do.

Contents

[Connection String Values](#)

[Adding Driver-Specific Connection Options](#)

[SQLSetStmtOption](#)

[Limitations](#)

[The Server List](#)

[Creating Client Installations](#)

[ODBC Overview](#)

[MFC Overview](#)

[Error Messages](#)

Connection String Values

SQLDriverConnect (used by the MFC CDatabase class) and SQLBrowseConnect use a connection string rather than a data source name, which allows you to specify where and how to connect without creating a data source. Any unknown connection string values are passed through for the ODBC Driver on the server to handle. The following list describes the connection string values for Power SQL-specific information.

PSQLDSN	Power SQL Server DSN - Translated to the DSN keyword on the server side. This is the ODBC data source name to connect to on the server side.
PSQLDRV	Power SQL Server Driver - Translated to the DRIVER keyword on server side. This is the ODBC database driver to use on the server side.
PSQLSRV	Power SQL Server Name - The domain name of the server, such as www.vsql.com. Either this, or the IP address is required
PSQLTO	Power SQL Timeout - Sets the client-side timeout.
PSQLIP	Power SQL Server IP - Used to connect to the server (required if PSQLSRV is not supplied). If the Server IP and PSQLSRV are supplied, the IP is tried first: the domain name in PSQLSRV is tried if the IP fails.
PSQLPORT	Power SQL Server Port - Used to connect to the server (required).
PSQLENCR	Power SQL Encryption DLL - Sets the DLL used to encrypt or compress your data as it is transferred across the Internet. This may be the compression DLL provided with Power SQL (PSCRYPT.PCR), or you may create your own proprietary encryption.
PSQLGETD	Power SQL Get Data Caching - Set this to zero (PSQLGETD=0) only if you need to disable GetData caching. For more information see Data Caching .
PSQLDD	Power SQL Data Dictionary Caching - Set this to zero (PSQLDD=0) only if you need to disable Data Dictionary caching. For more information see Data Caching .

Adding Driver-Specific Connection Options

The SQLSetConnectOption function takes a DWORD or a pointer to a string as one of its arguments. Without knowing which argument is intended, it is impossible to send this function to the server. For standard ODBC options, the data type to use with a specific option is clearly defined, so for all normal cases you will not have any problems.

However, ODBC allows database driver manufacturers to define database-dependent connection options for use with their own drivers. If your database driver uses such values, you need to add the new options to the PSQL.INI file along with an indication of their data type. The section for the new connection options is called PSQL Option Types, and contains a list consisting of the option number and a type for the option represented by that number. Numbers already taken by normal options are not recognized. Use the following format when adding database driver options:[PSQL Option Types]

1045=DWORD

1039=STRING

...

The server list is also contained in the PSQL.INI file. For more information, see [The Server List](#).

Limitations

Because Power SQL allows you to move your ODBC database driver over to the server, some of the functions have to be changed in a few minor details. For example, the dialog box displayed by `SQLDriverConnect` cannot be shipped across the network like other types of data. Also, although Power SQL was designed to prevent changes in database-dependent behavior, it may behave differently in one or two areas (such as when the behavior for one driver would conflict with another driver). The five functions that may behave differently are listed below.

SQLBindParameter - If a parameter is bound with a type of `SQL_PARAM_OUTPUT` or `SQL_PARAM_INPUT_OUTPUT`, the final data is written into these parameters when the statement is freed, even if the ODBC driver on the server has not. If this causes problems for you, you can make the parameter input only, or provide a pointer that will be valid until the bindings are dropped.

For literal pointers (a `DWORD` cast to a pointer for use with `SQLParamData` later), `SQLBindParameter` does not allow the parameter to be an output parameter. For `SQL_PARAM_INPUT_OUTPUT`, a warning is returned, and the parameter is considered to be an input-only parameter. For `SQL_PARAM_OUTPUT`, an error is returned.

SQLDriverConnect - If there is not enough connection information provided, Power SQL prompts the user for additional information by default. However, the ODBC database driver cannot be permitted to display a dialog to prompt the user because there is no practical way to display it from the server. Because of this, you must either have a data source name on the server, or enough information in the connect string to make the connection on the server.

SQLFreeStmt - According to the ODBC SDK, the `SQL_CLOSE` parameter does not cause the server to free any bound columns or parameters. To free those bindings (especially if you are freeing the memory they were bound to), use `SQL_UNBIND`, `SQL_RESET_PARAMS`, or `SQL_DROP`. This behavior is in accordance with the ODBC SDK, but it may conflict with some database drivers that do not fully conform to the ODBC SDK.

SQLSetPos - In `SQLSetPos`, you can update any row in your rowset (fetching bulk data and updating bulk data is supported by Power SQL). In addition, on some drivers, including Power SQL, you can use a spare row in the rowset for adding records. For example, if your rowset size is 10, you can declare the arrays of data to contain 11 elements and use the eleventh for added records. This feature is not supported for rowset sizes of 1 for performance reasons.

SQLSetStmtOption, or SQLSetConnectOption - Row-wise binding is not supported. When `SetStmtOption` is called on this item, it fails with a "Driver not capable" error.

The Server List

The PSQL.INI file contains information for the client that includes a list of Power SQL Servers. You can add your own servers here, or create standard servers by distributing this file with the Power SQL client. For example, if you are sharing a Marketing Statistics database on the www.MyCorp.com server, and using port 4545, you would put the following information into the PSQL.INI file:

```
[PSQL Servers]
Count=1
Server1=Marketing Statistics
```

```
[Marketing Statistics]
IP=www.MyCorp.com
TryIP=Y
Port=4545
Stored IP=123.45.78.9
```

The IP key is the server name, either a domain name or the IP address. The TryIP key indicates if a stored IP address is provided (it is always faster to connect to an IP address). The Port key provides the port number for the server, and the Stored IP key contains the IP address corresponding to the domain name in the IP key. When a client's user adds or modifies a Power SQL data source, the list of servers will contain a server named "Marketing Statistics." Any driver-dependent connection options are also stored here (see [Adding Driver Specific Connection Options](#)).

Creating Client Installations

After you create an application using Power SQL, you need to distribute the Power SQL Driver to all of your clients as part of your installation. There is no royalty attached to the Power SQL client side, so you can distribute the driver to as many Power SQL clients as you like. The Power SQL Driver has been designed to be easy to install, with a minimum set of requirements.

To install the Power SQL Driver, the client computer:

- Must be running Windows 95 or higher, or Windows NT 3.51, or higher
- Have ODBC installed (this is part of the operating system for Windows 95 and Windows NT 4.0)
- Have the Internet or intranet connection properly installed, with TCP/IP enabled, and an IP address assigned to the client computer
- Have about 200K free on the hard drive containing the Windows System directory.

In your installation, the Power SQL Internet Driver can be installed in two short steps.

1. Copy POWERSQL.DLL, POWERSQL.HLP, POWERSQL.CNT, and Msvcr7.dll to the system directory.
2. Execute the install function by using LoadLibrary to load the PowerSQL DLL, getting the address (GetProcAddress) of the InstallDriver function, and calling it.

This function registers the Power SQL Driver with the ODBC Manager, with a name of "Power SQL Internet Driver."

See Also

[The Server List](#)

[Adding Driver Specific Connection Options](#)

ODBC Overview

The Open Database Connectivity standard (ODBC) allows developers to use a common set of functions to access a wide array of databases. Over the years, ODBC has become more than a way to connect to different databases: it is now a method of programming that allows the programmer to learn one database language and be able to write applications for almost any database. In addition, once that application is written, it is not locked into one database. It is now relatively simple to develop with one database, then change your mind later, and use a different database. The ODBC database drivers have not only become faster, more powerful, and more stable, but ODBC is now the language of choice for accessing all databases in Microsoft's programming tools, Visual Basic, and the Visual C++ MFC classes. Power SQL implements all ODBC SDK functions, with very few [limitations](#).

MFC Overview

The Microsoft Foundation Classes from Microsoft Visual C++ contain two classes, CDatabase and CRecordset, which wrap the ODBC SDK functionality. These classes are used to access databases and manipulate data in an object-oriented fashion. Visual SQL also generates code using the MFC classes. Power SQL has been thoroughly tested to work with MFC. For more information about the MFC classes, see the Microsoft Visual C++ documentation.

Error Messages

Power SQL has two classes of error messages: those that come from the client, and those that come from the server. At times, it may be useful to know where the error originated. All errors from the server look something like this:

[Blue Sky][Power SQL **Server**] User Authentication failed, check the user name and password to be sure they are valid.

All errors from the client look something like this:

[Blue Sky][Power SQL **Client**] Unable to connect to server, Power SQL Server is not currently running.

Some of the error messages from the client and the server have “Communication Error:” as a prefix. These indicate that the error had to do with a network problem. For example:

[Blue Sky][Power SQL Client] Communication Error: Connection has been closed by the server.

SQLSetStmtOption

There are two new statement options that have been created for use with Power SQL. They can have a dramatic impact on the performance of your Power SQL / ODBC application. The `#define` statements for these options can be in "POWERSQL.H". The header file is in your Power SQL installation directory. The new statement options are:

- SQL_ENABLE_BATCH_SQLBINDCOL
- SQL_ENABLE_BULK_READ

SQL_ENABLE_BATCH_SQLBINDCOL

The SQL_ENABLE_BATCH_SQLBINDCOL option stores any BindColumn calls up until the first time Fetch is called. At that point, all of the columns are bound in one batch, then fetch is called on the server. This option will shorten the time required to get the first records in a query, especially if there is a large number of columns. The only negative side effect: if any BindColumn call fails, the error will not be noticed until the first time Fetch is called. At that point any error message will be returned. All BindColumn calls will return success.

For Visual SQL Users only: The SQL_ENABLE_BATCH_SQLBINDCOL will work very well with your generated Recordset classes. Just override the CRecordset::OnSetOptions function.

» [SQL_ENABLE_BULK_READ](#)

SQL_ENABLE_BULK_READ

The SQL_ENABLE_BULK_READ option causes an immense (easily 10-20 times faster reading of data) improvement in speed, however it does require a fair number of restrictions. When enabled, this will cause records to be read in blocks, rather than one at a time. This will shorten the time needed to scroll through the results of a query, or the time needed to fetch all of the data returned by some query. However, the result set can only scroll forward, and will not support positioned updates. The requirements for this are:

- » The option must be set immediately after SQLAllocStmt, before any result set has been attached to the statement handle. The easiest way to do this may be to set it as a connection level option.
- » The Statement handle must have the SQL_CURSOR_TYPE option is set to SQL_CURSOR_FORWARD_ONLY, and the SQL_ROWSET_SIZE must be 1.
- » Positioned updates of any kind (either using SQLSetPos, or a WHERE CURRENT OF Cursor statement) are not permitted. Updates to the data may take place by using SQL statements, but no other way.
- » If calling SQLGetData, you must get all columns desired on the first record. This set of columns is used as a pattern to determine what data to fetch, so no other columns will be retrieved (just like the GetData caching).

Note: When this option is used, the SQLExtendedFetch, SQLSetPos, and SQLSetCursorName functions are disabled on that statement handle.

For Visual SQL Users only: The SQL_ENABLE_BULK_READ option is designed for the datasheet and you can use it with both read-only datasheets and datasheets that you can modify. (You can still update your data because Visual SQL's datasheet method for updating the database does not require positioned updates.)

- » [SQL_ENABLE_BATCH_SQLBINDCOL](#)

