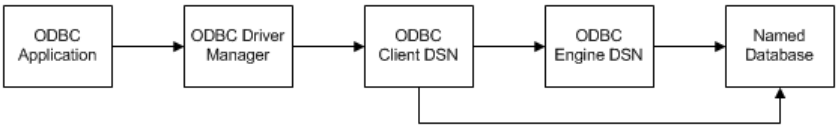
Tech Tip #15: Creating and Using a Client DSN

Overview: How to create a client DSN on a workstation for use with Access or SQLExec

The last few Tech Tips worked with the **Pervasive Control Center**. However, sometimes, you want to access data from another ODBC-compatible application, so let's dig into this a bit further.

PSQL connections over Open DataBase Connectivity (ODBC) go through a number of layers to get to the server. When an application opens up a database connection, it provides a *Data Source Name* (*DSN*) to the ODBC Driver Manager. The ODBC driver manager then uses the DSN to locate an appropriate ODBC driver on the client workstation, known as an *ODBC Client Interface*. The ODBC Client Interface (often called a *Client DSN* for short) contains information about how to locate the server (such as name, protocol, etc.), along with either an *Engine Data Source Name* (*Engine DSN* for short) or a *Named Database* (*DBName* for short) on that server. If the Client DSN contains an Engine DSN reference (optional with PSQLv11), then the ODBC Driver Manager on the server is invoked to locate the appropriate *ODBC Engine Interface* on the server, which is then used to locate the DBName. Alternatively, PSQLv11 supports the use of a DBName directly on the client, which skips this interim step.

Confused yet? Luckily, a picture is worth 1 kiloword:

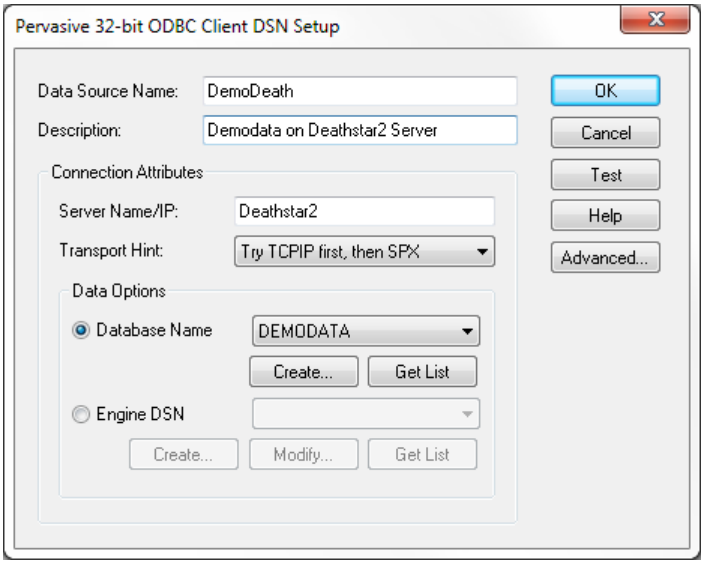


Again, it is possible for the Client DSN to indicate either an Engine DSN or a DBName, so there are two possible paths for PSQLv11. In most cases, when you create a DBName on the server, you ALSO create an Engine DSN of the exact same name. (See Tech Tip #12 for the Create 32-bit Engine DSN option.) Sticking with the same name is not required, but it does simplify things. Creating the DSN on the server is only half the battle – you still need a DSN on each client that will be accessing the database.

Before we can do this, however, there is one more topic to cover: the **ODBC Administrator**. On 32-bit operating systems, there is one ODBC Administrator (the 32-bit version), and it is located in C:\Windows\System32\ODBCAD32.EXE. However, on 64-bit systems, there are *two different ODBC Administrators*. In order to to drive up support revenues, Microsoft opted to place the **64-bit ODBC Administrator** in C:\Windows\System32\ODBCAD32.EXE. Then, to avoid anyone catching on too quickly, they moved the **32-bit ODBC Administrator** to C:\Windows\SysWOW64\ODBCAD32.EXE. Confused yet? In case anyone ever figured THAT out, Microsoft (knowing that users would still have 32-bit applications) made the ODBC Administrator link in the Control Panel point to the 64-bit ODBC administrator, and completely removed any links to the 32-bit ODBC Administrator. Cha-ching!

Luckily, you have the **Pervasive Control Center**. Using the *Tools* menu, you can select the right ODBC Administrator for your needs every time. If you are setting up a link for 32-bit applications, use the 32-bit ODBC Administrator. If you have a 64-bit application (like SQL Server, 64-bit versions of Office, etc.), then you may need the 64-bit ODBC Administrator.

Now that you have the right ODBC Administrator, you can create the Client DSN. If you are making a private DSN (that only you will access), then create a new *User DSN*. Otherwise, use the *System DSN* and click *Add* to get to the *ODBC Client DSN Setup* dialog box.



The *Data Source Name* can be whatever you want, but keep it short for best results. The *Description* can be whatever you want, as it really isn't used anywhere.

Be sure to specify the *Server Name* properly, or use an IP address so that the database engine can be located.

Then, decide if you want to talk directly to the DBName or to the Engine DSN and select the appropriate radio button. (Note: A 64-bit Client DSN can *only* talk to a DBName, so that dialog box does not show the Engine DSN fields.) Once selected, click *Get List* to get a list of objects on the server, and then select the right one from the drop box. Click *OK* and your Client DSN is now created and available for use.

To use the DSN, simply specify that DSN in your ODBC-compatible application. One such application is the command-line utility **SQLExec** from Goldstar Software:

http://www.goldstarsoftware.com/sqlexec.asp

This 32-bit tool allows you to access any DSN from a command line in order to submit ad hoc queries or to script entire processes from a batch file. Here's an example of **SQLExec** in action:

```
C:\>SQLExec DemoDeath "SELECT TOP 1 ID, Last_Name, First_Name FROM PERSON ORDER BY
Last_Name"
SQLEXEC Version 2.54: 01/30 (C)2014 Goldstar Software Inc.
Registered to Goldstar Software Inc. (Site License GS)
"ID","Last_Name","First_Name"
"175053812","Abad","Alicia"
```

SQLExec supports a large number of formatting options, from comma delimited to fixed field to delimiters of other sorts. It also supports text processing, such as eliminating spaces from fields, filtering quotes or control characters, and more.

Another ODBC-compatible application is **Microsoft Office**, but the exact process to connect to an ODBC data source varies depending on your program (**Excel**/**Word**/**Access**) and the exact version. Just remember this key piece of trivia -- if you have a 64-bit version of Office, you need to use a 64-bit DSN. If you have a 32-bit version of Office, you need to use a 32-bit DSN. Another key note for MS **Access** users – the **JET** engine (used by **Access**) doesn't handle all data types supported by PSQL and may have other limitations (such as the number of fields in a table) that differ from PSQL. In these cases, you may have to utilize an *ODBC Pass-Through Query* to access the database from a query, instead of linking directly to the PSQL tables. See the **Access** manuals for information on setting this up.