

# Connecting Excel to MySQL or PostgreSQL via ODBC

**Published by the Open Source Software at Microsoft, April 2007**

Special thanks to Chris Travers, Contributing Author to the Open Source Software Lab

Most current version will be maintained at: <http://port25.technet.com>



## Introduction

In cases where business applications have been built on open source databases, it may be necessary to connect other Windows applications, such as Microsoft Access or Excel, to these databases for reporting or business intelligence purposes.

One potential application of this process is to use Excel as a front-end for data analysis. Data can be pulled from views or tables and then further analyzed, graphed, and the like. Even pivot tables can be used to create even more powerful reporting solutions.

This how-to walks through this process using Excel as an example application. Although in this example, the MySQL and PostgreSQL servers are running on Linux, the steps are no different if the software is running on Windows. These steps are:

1. Setting up authentication
2. Installing the ODBC Drivers
3. Configuring the data source
4. Importing the data.

## Server Configuration: Authentication Requirements

Relational database management systems often store sensitive information and so require some sort of authentication and authorization in order allow data retrieval. Before one can connect, one must make sure that the user account to be used exists and has appropriate permissions. Those who do not run their own servers can skip this section and expect their IT staff to be able to handle the appropriate user rights issues, but these steps are included for those who may not have that luxury.

## MySQL

In MySQL, both the hostname and username are part of the user identifier, allowing for the same username to be used with different passwords and different rights from different client systems. Note that the GRANT command can implicitly create users. The process of creating users via GRANT is not covered here because most databases separate user creation from access permissions management. It is best to learn the most generally applicable processes.

To create the user, connect to the database using the standard command-line client:

```
bash$ mysql -u root
```

Then create the user:

```
mysql=> CREATE USER 'foo'@'%' IDENTIFIED BY 'bar';
```

This creates a user named 'foo' which is allowed to connect from all hosts, and has a password of 'bar.' Obviously, it is a good idea to use a stronger password.

To grant SELECT privileges (necessary for importing data into Excel), the next command would be:

```
GRANT SELECT ON dbname.tablename TO foo;
```

In the above example, dbname should be replaced with the name of your database and tablename should be replaced with the name of the table you want to grant the privileges on.

Note that MySQL supports a non-standard use of GRANT [privilege] ON \* to foo. Although this is common on MySQL because of a traditional lack of role-level security (prior to 5.0), and the fact that it was used primarily on single-application databases. This is not considered to be a good practice because it is easy to inadvertently grant unnecessary permissions this way.

## PostgreSQL

PostgreSQL uses a system where the username is unique to a server instance but where the client connection is passed through a host-based authentication layer in order to determine the appropriate authentication method. To add the necessary access, it is necessary to edit the pg\_hba.conf. This file is found in the data directory when it is installed from source but many Linux distributions place it somewhere in /etc/. Locating the file will be distribution-dependent.

PostgreSQL supports a large number of internal authentication methods. These include:

- **identd** (usually identd sameuser) attempts to map the logged in user to the connection request. This does not work properly on Windows, and is not safe over a network connection.
- **password** passes the password in plain text over the network. It is not recommended.
- **crypt** uses the UNIX crypt hash algorithm for challenge/response authentication. It is considered to be weaker than md5.
- **md5** creates a hash based on the password and log-in. For password-protected access, this is the recommended means of authentication.
- **trust** allows all connections without further asking for credentials. While it is useful for data recovery, it is not recommended for production systems.
- **reject** disallows all matching connections. This is useful for preventing certain systems or networks from connecting.
- **pam** passes the password in plain text over the network but authenticates the user as a system user on the host. It does not work properly on Windows, and is not generally recommended.

Additionally, the following external means of authentication are supported but beyond the scope of this paper:

- **krb5** allows one to authenticate using Kerberos tickets.
- **ldap** allows one to authenticate against an LDAP database. Passwords are passed over the network in plain text.

If one wants to let the user 'foo' connect to all databases from any host but using md5 authentication, you will want to add the following line to the end of the pg\_hba.conf:

```
#TYPE          DATABASE USER CIDR          METHOD
host           all           foo    0.0.0.0/0      md5
```

Once this has been made, the configuration can be reloaded by running the appropriate init script with the reload argument, by using pg\_ctl reload from the command line, or by stopping and starting the service. Novice users will probably want to use this last option because it is the simplest. Each of these methods, though, may vary between distributions.

Then, a user can be created either in a graphical administration tool such as pgAdmin III or from a command-line client such as psql. If you use the command-line to create the user and give permission, the commands are:

```
CREATE USER 'foo' WITH ENCRYPTED PASSWORD 'bar';
GRANT SELECT ON tablename TO 'foo';
```

## Client Configuration I: Downloading/Installing the Drivers

The easiest way to connect the application to the database is to use the ODBC driver for that database. The ODBC framework presents a consistent interface for a program like Excel to access the database, and these drivers connect the ODBC framework to the actual databases (in this case MySQL or PostgreSQL).

The Drivers can be found at:

- MySQL: <http://dev.mysql.com/downloads/connector/odbc/3.51.html>
- PostgreSQL: <http://www.postgresql.org/ftp/odbc/versions/msi/>

In both cases, the installation of these drivers on the Windows client is straight-forward and requires no more explanation.

Vista users will note a pop-up warning from User Account Control but disabling it is not required for this installation.

## Client Configuration II: Creating the Data Source Name (DSN)

In order to pull the data from the database, we need to be able to connect to the database. The Data source name (DSN) provides a way of storing the information necessary to make such a connection in Windows so that any application can use it. This way, when we import the data in Excel, the program knows what server to connect to, what username and password to supply, and any other settings that are needed to make the data available.

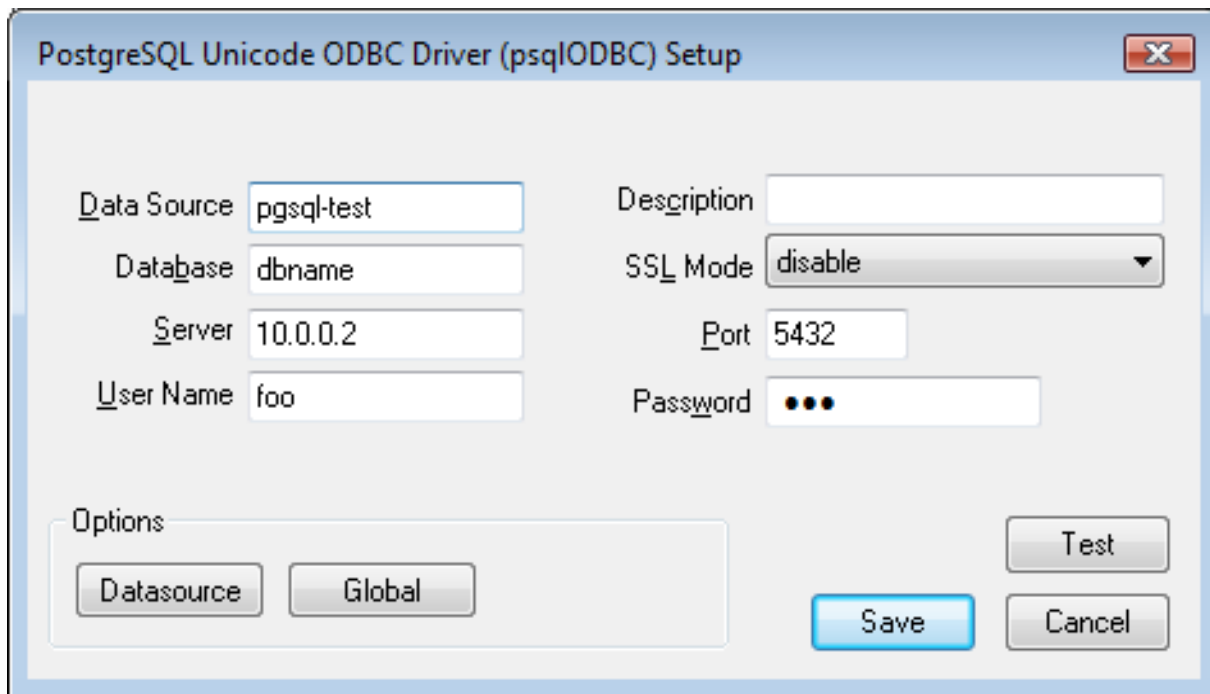
If you are connecting to PostgreSQL, you can skip this stage for now and create your DSN through Microsoft Query. MySQL, however, requires that the DSN be created in the ODBC data source administrator. This tool is found in the control panel,

In Classic View, it is found under "Administrative Tools." In Vista's Category View, this folder is found under "System and Maintenance."

A System DSN is accessible to the entire system, while a User DSN is limited to the user that creates it.

## PostgreSQL

The PostgreSQL DSN creation screen is as follows:



The screenshot shows the "PostgreSQL Unicode ODBC Driver (psqlODBC) Setup" dialog box. It features the following fields and controls:

- Data Source:** Text box containing "pgsql-test".
- Database:** Text box containing "dbname".
- Server:** Text box containing "10.0.0.2".
- User Name:** Text box containing "foo".
- Description:** Empty text box.
- SSL Mode:** Dropdown menu set to "disable".
- Port:** Text box containing "5432".
- Password:** Password field with three dots indicating it is masked.
- Options:** A group box containing "Datasource" and "Global" buttons.
- Test:** Button to test the connection.
- Save:** Button to save the configuration.
- Cancel:** Button to cancel the setup.

The fields are as follows:

- **Data Source** is the name of the data source. Enter a value you can easily remember.
- **Database** is the name of the database you wish to connect to
- **Server** is the ip address or hostname of the server you want to connect to.
- **User Name** is the username you want to authenticate as on the database.
- **Description** can be used for notes to help you remember what a given connection was created for
- **SSL Mode** specifies whether you want to use SSL when connecting to the database. This can be useful in cases where the database connection is carrying confidential information across a public network.

- **Port** is the TCP port to connect to. The default (5432) should be used unless you know otherwise.
- **Password** is the password you want to use to log in.

If you wish to test your credentials (highly recommended), you can click the "Test" button.

## MySQL

Once you select the ODBC driver, a similar screen will pop up to help you through the connection process.

Connector/ODBC 3.51.12 - Add Data Source Name

Connector/ODBC

MySQL

Login | Connect Options | Advanced

Data Source Name: mysql-test

Description:

Server: 10.0.0.2

User: foo

Password: \*\*\*

Database: hermes

Database

The database to be current upon connect.

Optional	Yes
Default	[none]

Test | Diagnostics >> | Ok | Cancel | Help

The fields are as follows:

- **Data Source Name** is the name of the data source. Enter a value you can easily remember.

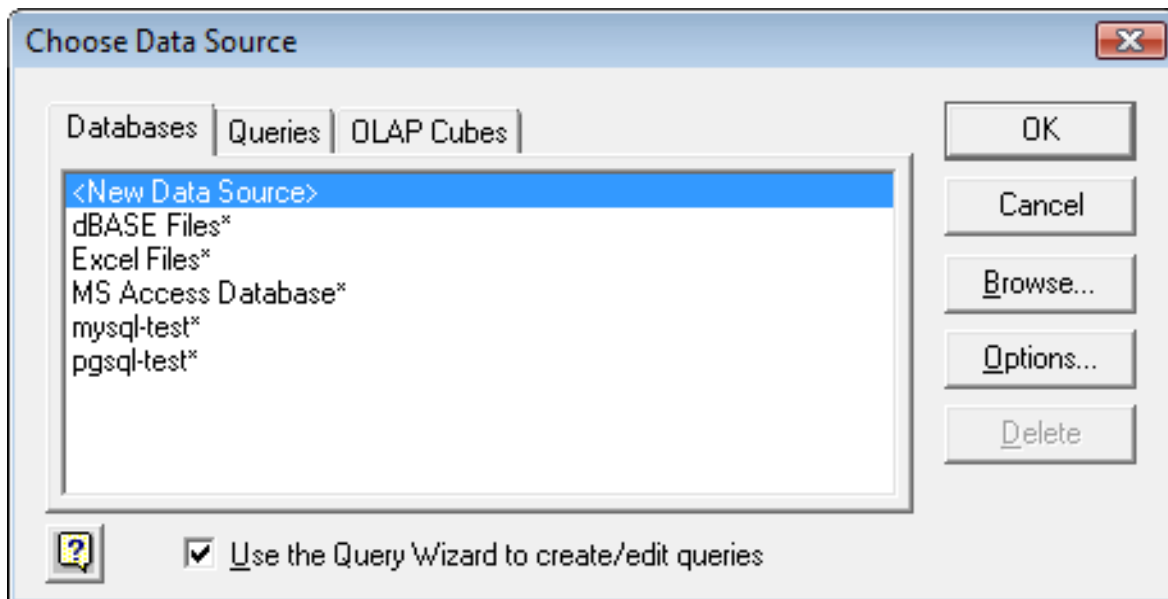
- **Description** can be used for notes to help you remember what a given connection was created for
- **Database** is the name of the database you wish to connect to
- **Server** is the ip address or hostname of the server you want to connect to.
- **User** is the username you want to authenticate as on the database.
- **Password** is the password you want to use to log in.

## Importing the Data into Excel

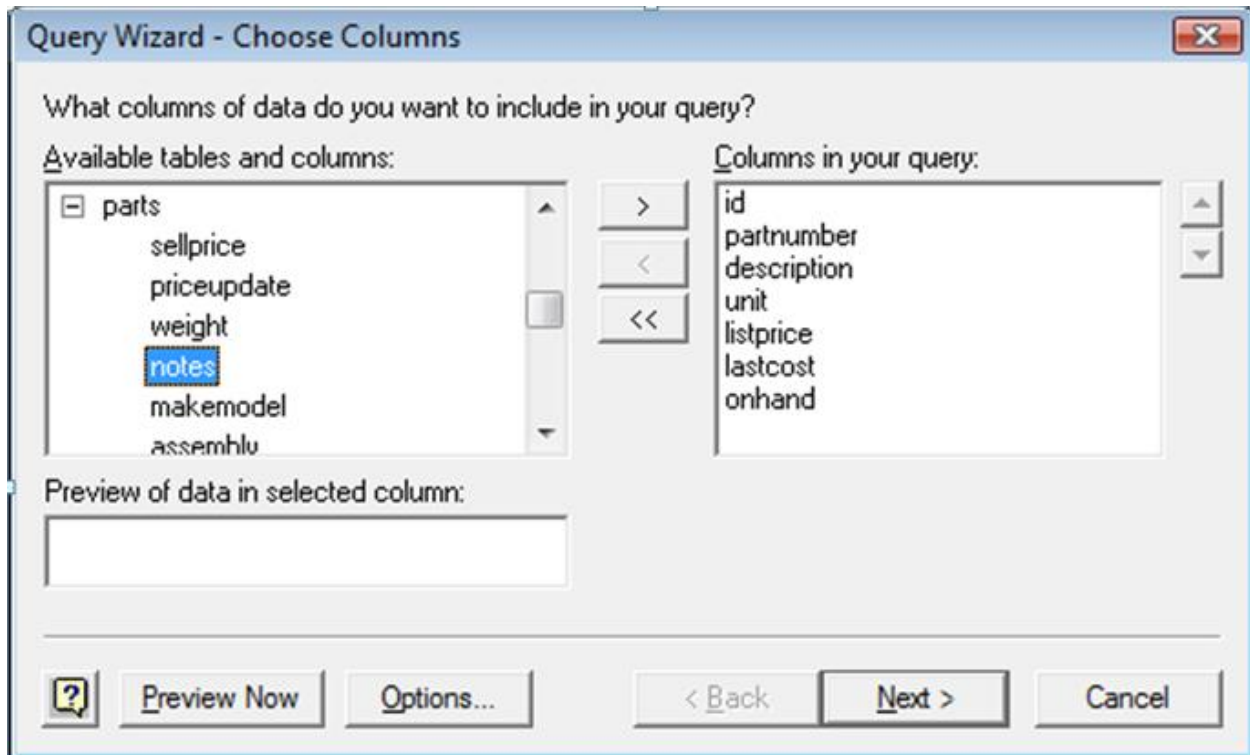
Different versions of Excel have subtly different user interfaces. The directions here are expected to work on all currently supported Windows versions. However, in some cases, the labels may vary slightly from what is noted here.

These steps essentially take the components we have been working with (Excel, ODBC, and the database server) and connect them all together. The final steps are to tell Excel to use the DSN to connect to the database server, what data to retrieve from the database, and what to do with that data.

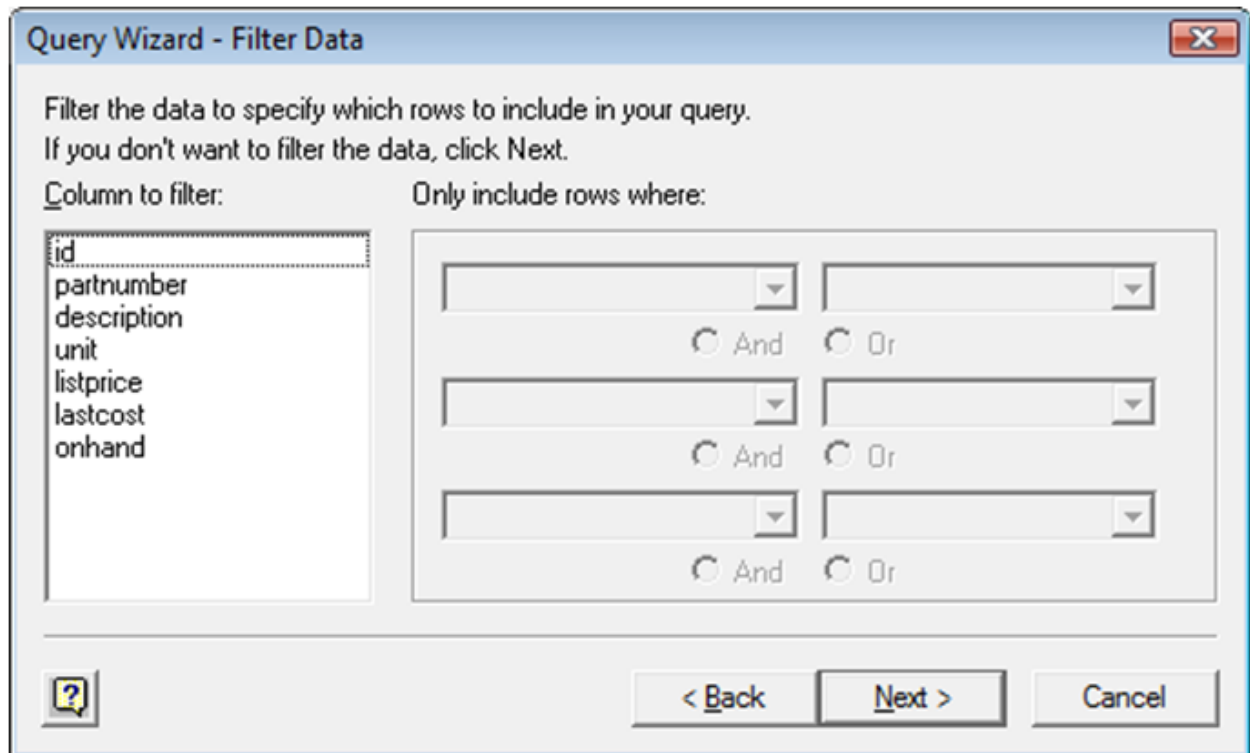
In the data menu, select "Get External Data" (Excel 2003 changes this to "Import External Data"). From the child menu, select "New Database Query." The following screen will appear:



Select the desired DSN and click OK. Note that if you are using PostgreSQL, you can create a new user DSN in by selecting "<New Data Source>." If you go this route, you will need to go through the steps outlined above in creating a new DSN. I am not entirely sure why the MySQL driver does not support this behavior.

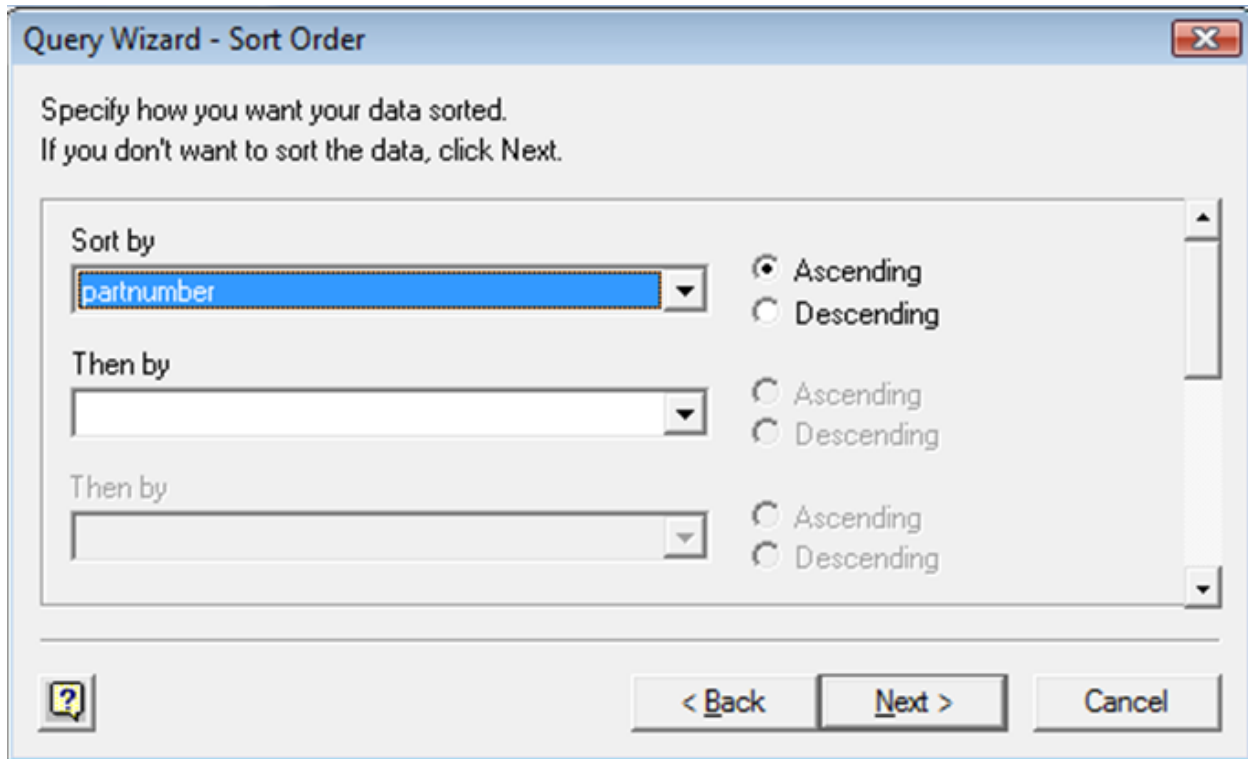


On the next step, you can select the tables and fields you want to return. In this simple example, I only selected fields from one table.



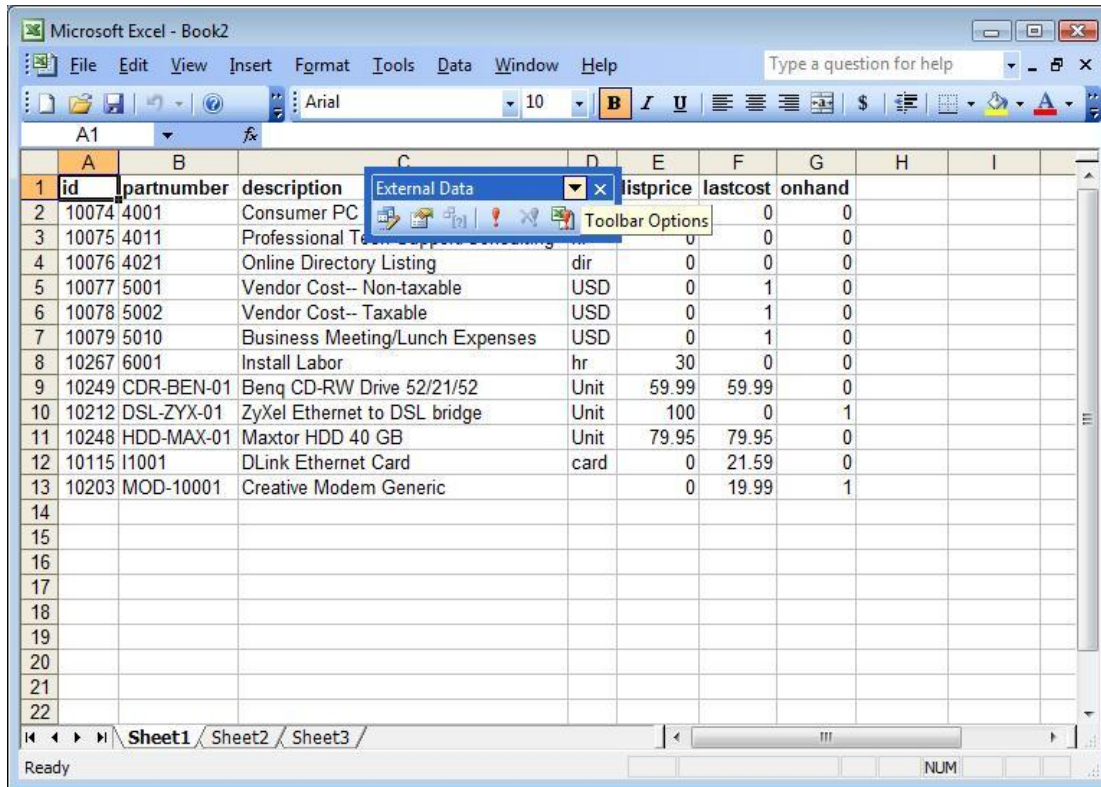


In the next step, you can create filters. This can be a useful way to limit the number of rows returned since Excel can only handle a little over 65,000 rows.



One can also add ordering to the returned results. This can help maintain the order when the data is refreshed.

At the end of the wizard, you have the option of returning the data to Excel. If you select this option, the data will appear on the spreadsheet.



The screenshot shows a Microsoft Excel spreadsheet with the following data:

	A	B	C	D	E	F	G	H	I
1	id	partnumber	description		listprice	lastcost	onhand		
2	10074	4001	Consumer PC			0	0		
3	10075	4011	Professional T			0	0		
4	10076	4021	Online Directory Listing	dir	0	0	0		
5	10077	5001	Vendor Cost-- Non-taxable	USD	0	1	0		
6	10078	5002	Vendor Cost-- Taxable	USD	0	1	0		
7	10079	5010	Business Meeting/Lunch Expenses	USD	0	1	0		
8	10267	6001	Install Labor	hr	30	0	0		
9	10249	CDR-BEN-01	Benq CD-RW Drive 52/21/52	Unit	59.99	59.99	0		
10	10212	DSL-ZYX-01	ZyXel Ethernet to DSL bridge	Unit	100	0	1		
11	10248	HDD-MAX-01	Maxtor HDD 40 GB	Unit	79.95	79.95	0		
12	10115	I1001	DLink Ethernet Card	card	0	21.59	0		
13	10203	MOD-10001	Creative Modem Generic		0	19.99	1		
14									
15									
16									
17									
18									
19									
20									
21									
22									

Note that Excel does not provide access to real-time access to the data. As data is modified in the database, Excel does not update the database. If you want to see the most current data, you must use the **Refresh Data** item on the **Data** menu to pull the most recent information from the database.

## Final Thoughts

The potential for this sort of connection goes well beyond the brief example given here. More advanced uses may include access to database views, allowing greater amounts of data to be aggregated more quickly, and advanced features of Microsoft Query. The combination of these two tools provide a quick reporting interface which is quite useful. Furthermore, the same techniques can be used to use these database management systems as back-ends to Microsoft Access.

## About the Author

Chris Travers is the owner of Metatron Technology Consulting, a business dedicated to helping customers best utilize open source software. He has over ten years IT experience, most of that with open source database management systems and tools.

## Copyright

Information in this document, including URL and other Internet Web site references, is subject to change without notice and is provided for informational purposes only. The entire risk of the use or results from the use of this document remains with the user, and Microsoft Corporation makes no warranties, either express or implied. Unless otherwise noted, the companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in examples herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

© 2007 Microsoft Corporation. This work is licensed under the Microsoft Permissive License. The Microsoft Permissive License is [available here](#).

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Microsoft, Windows, Windows XP, Windows Server, and Windows Vista are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

All other trademarks are property of their respective owners.