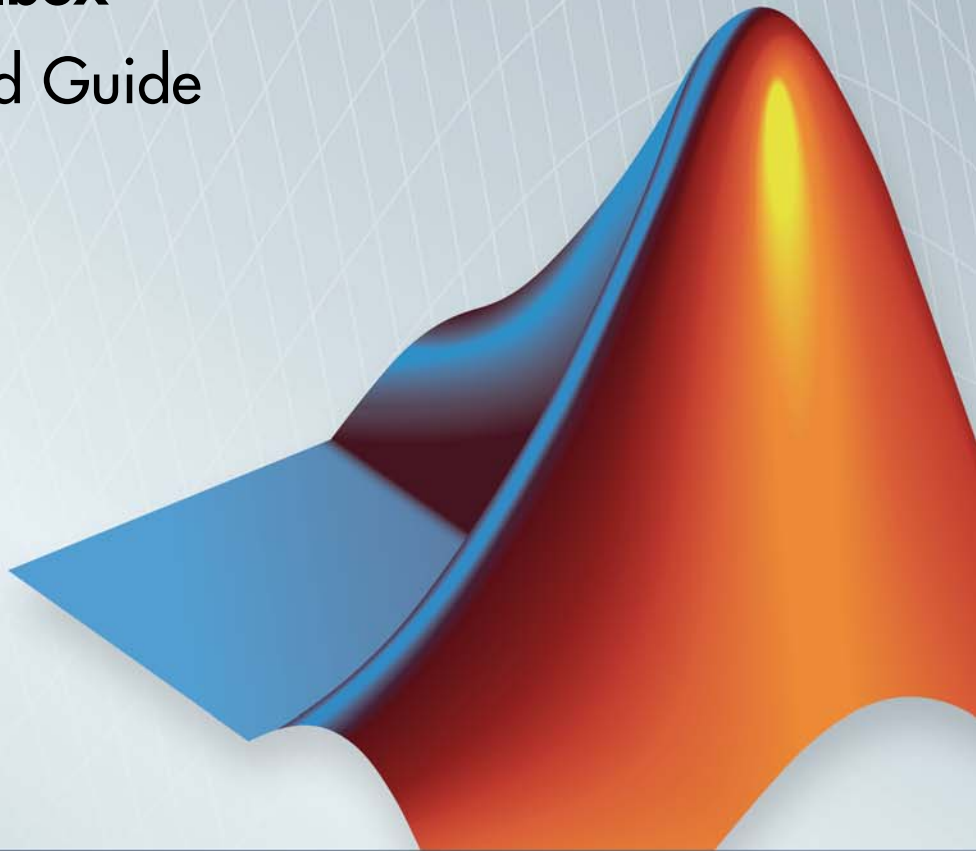


Database Toolbox™

Getting Started Guide

R2013b



MATLAB®



How to Contact MathWorks



www.mathworks.com Web
comp.soft-sys.matlab Newsgroup
www.mathworks.com/contact_TS.html Technical Support



suggest@mathworks.com Product enhancement suggestions
bugs@mathworks.com Bug reports
doc@mathworks.com Documentation error reports
service@mathworks.com Order status, license renewals, passcodes
info@mathworks.com Sales, pricing, and general information



508-647-7000 (Phone)



508-647-7001 (Fax)



The MathWorks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098

For contact information about worldwide offices, see the MathWorks Web site.

Database Toolbox™ Getting Started Guide

© COPYRIGHT 1998–2013 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

Revision History

September 2007	Online only	New for Version 3.4 (Release 2007b)
March 2008	Online only	Revised for Version 3.4.1 (Release 2008a)
October 2008	Online only	Revised for Version 3.5 (Release 2008b)
March 2009	Online only	Revised for Version 3.5.1 (Release 2009a)
September 2009	Online only	Revised for Version 3.6 (Release 2009b)
March 2010	Online only	Revised for Version 3.7 (Release 2010a)
September 2010	Online only	Revised for Version 3.8 (Release 2010b)
April 2011	Online only	Revised for Version 3.9 (Release 2011a)
September 2011	Online only	Revised for Version 3.10 (Release 2011b)
March 2012	Online only	Revised for Version 3.11 (Release 2012a)
September 2012	Online only	Revised for Version 4.0 (Release 2012b)
March 2013	Online only	Revised for Version 4.1 (Release 2013a)
September 2013	Online only	Revised for Version 5.0 (Release 2013b)

Using the Database Toolbox Software

1

Database Toolbox Product Description	1-2
Key Features	1-2
Product Prerequisites	1-3
Software Requirements	1-3
Background Knowledge	1-3
Configuring Your Environment	1-4
About Data Sources and Database Drivers	1-4
Before You Begin	1-5
Set Up the dbtoolboxdemo Data Source	1-6
Configure ODBC Data Sources	1-6
Configure JDBC Data Sources	1-11
Starting the Database Toolbox and Visual Query Builder Software	1-16
Database Toolbox Workflows	1-17
Importing Database Data	1-17
Exporting Data to a Database	1-17
Using Database Explorer to Import Database Data ...	1-19
Database Toolbox Functions Versus Visual Query Builder	1-20
Limitations of Visual Query Builder	1-20
Working with Visual Query Builder	1-22
Using Queries to Import Database Data	1-22
Saving Queries	1-27
Running Saved Queries	1-28
Editing Queries	1-28

Clearing Variables from the VQB Data Area	1-28
Using Queries to Export Data to Databases	1-29
Exiting Visual Query Builder	1-32
Learning More	1-33
Product Help	1-33
MathWorks Online	1-33

Index

Using the Database Toolbox Software

- “Database Toolbox Product Description” on page 1-2
- “Product Prerequisites” on page 1-3
- “Configuring Your Environment” on page 1-4
- “Starting the Database Toolbox and Visual Query Builder Software” on page 1-16
- “Database Toolbox Workflows” on page 1-17
- “Using Database Explorer to Import Database Data” on page 1-19
- “Database Toolbox Functions Versus Visual Query Builder” on page 1-20
- “Working with Visual Query Builder” on page 1-22
- “Learning More” on page 1-33

Database Toolbox Product Description

Exchange data with relational databases

Database Toolbox™ provides an app and functions for exchanging data between relational databases and MATLAB®. You can use SQL commands to read and write data or use the Database Explorer app to interact with a database without using SQL.

The toolbox supports ODBC-compliant and JDBC-compliant databases, including Oracle®, MySQL®, Sybase®, Microsoft® SQL Server®, and Informix®. You can apply simple and advanced conditions to database queries from MATLAB. The toolbox lets you access multiple databases simultaneously within a single MATLAB session and enables segmented import of large data sets.

Key Features

- Database Explorer app for working with databases interactively
- JDBC-compliant database connections
- ODBC-compliant database connections, with the option for fast access via a native ODBC driver
- Functions for executing queries using SQL files and SQL statements
- Data import and export with multiple databases in a single session
- Large data set import via a single transaction or via multiple transactions of segmented data
- Direct data import into numeric, cell, structure, and dataset arrays

Product Prerequisites

In this section...
“Software Requirements” on page 1-3
“Background Knowledge” on page 1-3

Software Requirements

- The Database Toolbox product requires that the MATLAB software be installed.
- This toolbox requires your system to have access to a supported database. For a list of supported databases, see “Database Support”.

Background Knowledge

The following technical knowledge is required to use this toolbox:

- This documentation assumes that you know how to work with MATLAB cell arrays and structures.
- You must be familiar with basic database concepts, such as tables, attributes, and queries.
- You do not need to be familiar with Structured Query Language (SQL) to use Database Toolbox functions and Visual Query Builder. However, you do need to be familiar with SQL to perform complex queries and database operations.

Configuring Your Environment

In this section...
“About Data Sources and Database Drivers” on page 1-4
“Before You Begin” on page 1-5
“Set Up the dbtoolboxdemo Data Source” on page 1-6
“Configure ODBC Data Sources” on page 1-6
“Configure JDBC Data Sources” on page 1-11

About Data Sources and Database Drivers

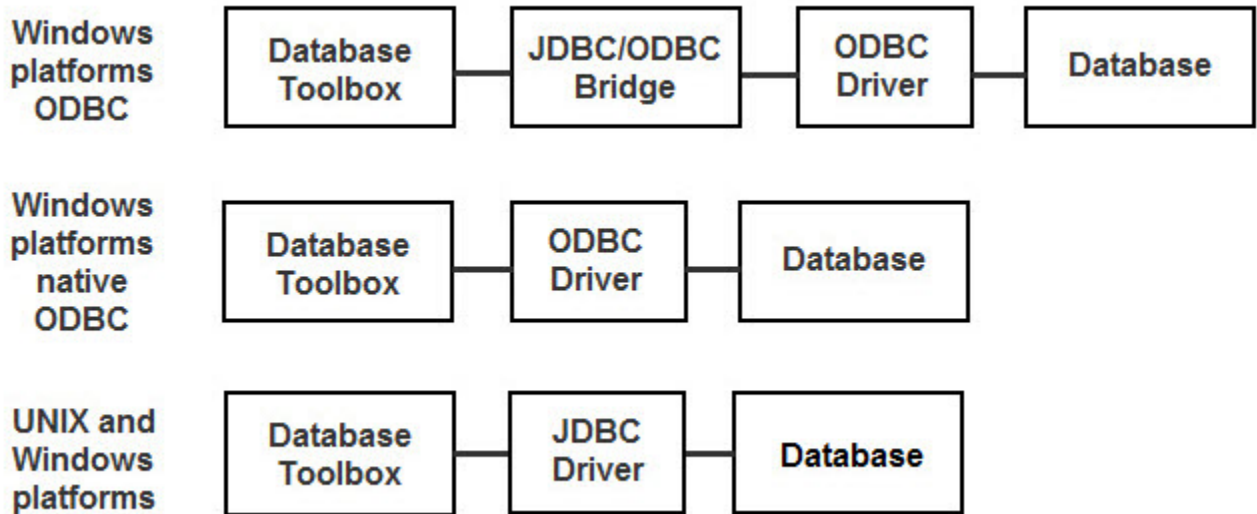
Before you can use the Database Toolbox to connect to a database, you must set up a data source, consisting of:

- Data that the toolbox accesses
- Information required to find the data, such as driver, folder, server, or network names

Data sources interact with *ODBC drivers* or *JDBC drivers*. An ODBC driver is a standard Microsoft Windows® interface that enables communication between database management systems and SQL-based applications. A JDBC driver is a standard interface that enables communication between applications based on Oracle Java® and database management systems.

The Database Toolbox has a Java library that connects directly to a pure JDBC driver or uses the JDBC/ODBC bridge to connect to an ODBC driver. The JDBC/ODBC bridge is automatically installed as part of the MATLAB JVM™. Database Toolbox also has a C++ library that connects natively to an ODBC driver.

The following figure illustrates how drivers interact with Database Toolbox.



Tip On Windows systems that support both ODBC and JDBC drivers, pure JDBC drivers and the native ODBC interface provide better connectivity and performance than the JDBC/ODBC bridge.

Before You Begin

Before you can use Database Toolbox with the examples provided in this documentation, do the following:

- 1 Set up the data sources that are provided with Database Toolbox. For instructions, see “Set Up the dbtoolboxdemo Data Source” on page 1-6.
- 2 Configure the data sources for use with your database driver.
 - If you are using an ODBC driver, see “Configure ODBC Data Sources” on page 1-6.

- If you are using a JDBC driver, see “Configure JDBC Data Sources” on page 1-11.

Set Up the dbtoolboxdemo Data Source

The dbtoolboxdemo data source uses the tutorial database located in `matlabroot/toolbox/database/dbdemos/tutorial.mdb`.

To set up this data source:

- 1 Copy `tutorial.mdb` into a folder to which you have write access.
- 2 Confirm you have write access to `tutorial.mdb`.
- 3 Open `tutorial.mdb` from within the MATLAB Current Folder browser by right-clicking the file and selecting **Open Outside MATLAB**. The file opens in Microsoft Access™.

Note You might need to convert the database to the version of Access you are currently running. For example, beginning in Microsoft Access 2007, you see the option to save as `*.accdb`. For more information, consult your database administrator.

Configure ODBC Data Sources

When setting up a data source for use with an ODBC driver, the target database can be located on a PC running the Windows operating system or on another system to which the PC is networked. These instructions use the Microsoft ODBC Data Source Administrator Version 6.1 for the U.S. English version of Microsoft Access 2010 for Windows systems. If you have a different configuration, you might need to modify these instructions. For more information, consult your database administrator.

- 1 Close open databases, including `tutorial.mdb` in the database program.
- 2 Start Database Explorer by clicking the **Apps** tab on the MATLAB Toolstrip and then selecting **Database Explorer** from the **Database Connectivity and Reporting** section in the apps gallery. Alternatively, at the command line, enter:

dexplore

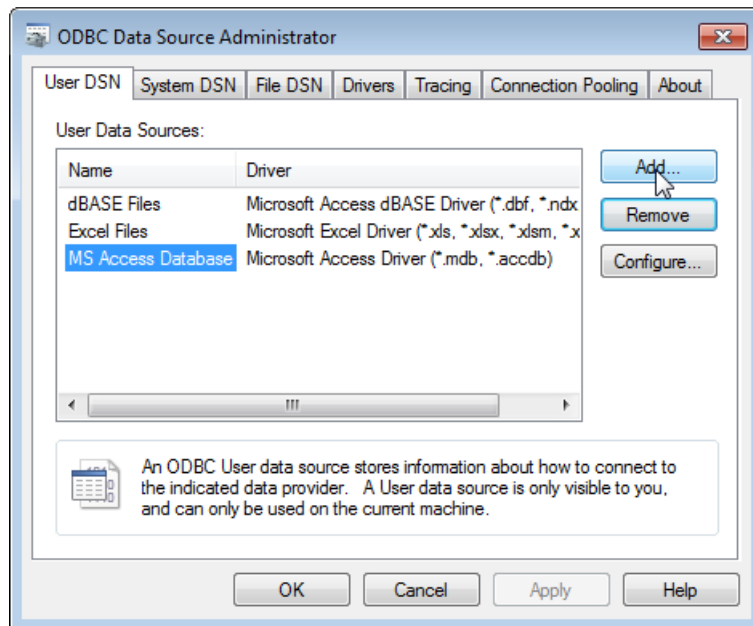
If no data sources are set up, a message box opens. Click **OK** to dismiss the message. Otherwise, the **Connect to a data source** prompt displays. Click **Cancel** to dismiss this prompt.

- 3 Click the **Database Explorer** tab and then select **New > ODBC** to open the ODBC Data Source Administrator dialog box to define the ODBC data source.

Requirement When using a 32-bit version of Microsoft Office, you must also use a 32-bit version of MATLAB to complete the remaining steps.

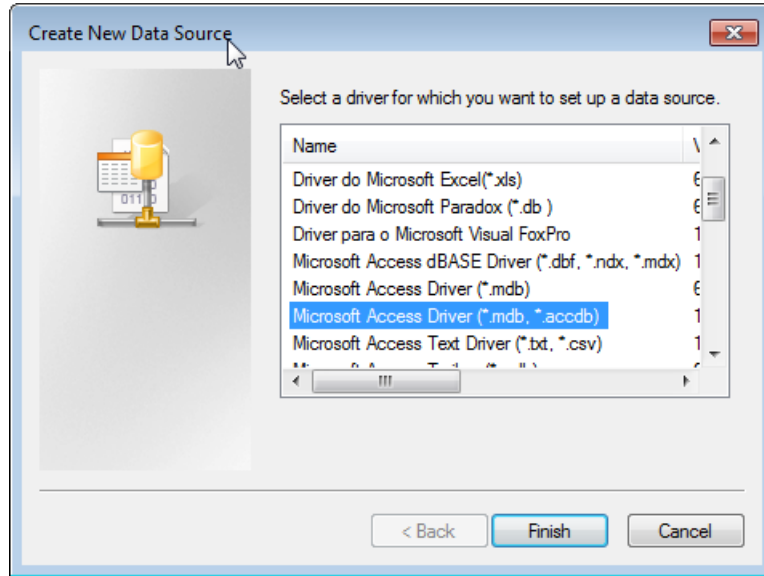
- 4 Click the **User DSN** tab.

- 5 Click **Add**.

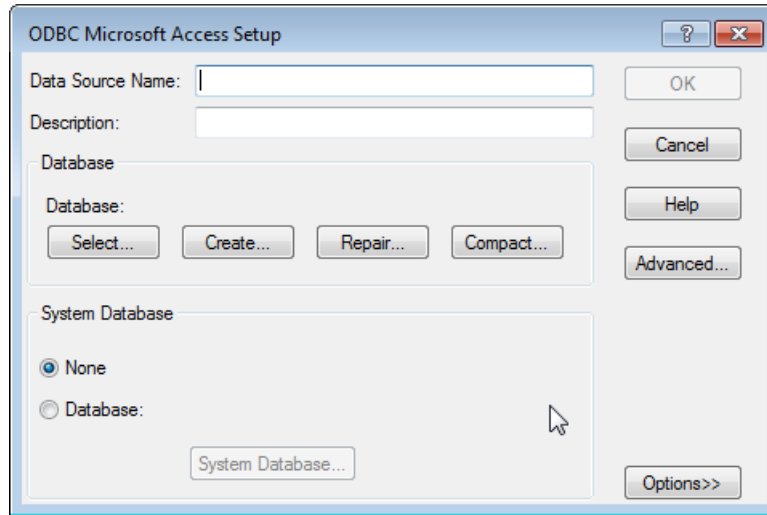


A list of installed ODBC drivers appears in the Create New Data Source dialog box.

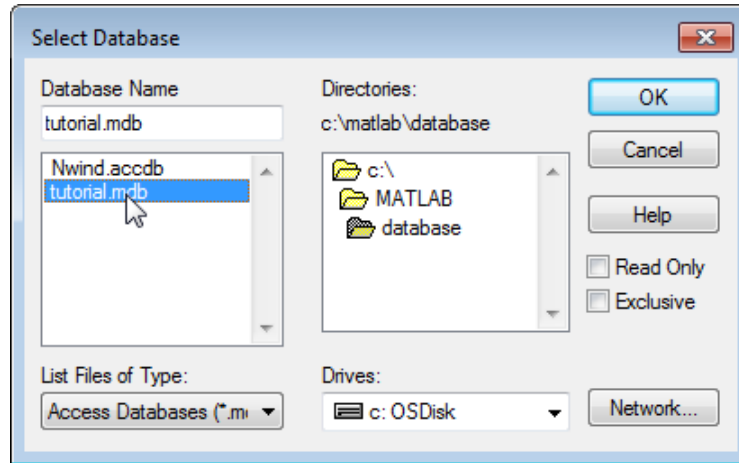
- 6 Select Microsoft Access Driver (*.mdb, *.accdb) and click **Finish**.



The ODBC Microsoft Access Setup dialog box for your driver opens. The dialog box for your driver might differ from the following.

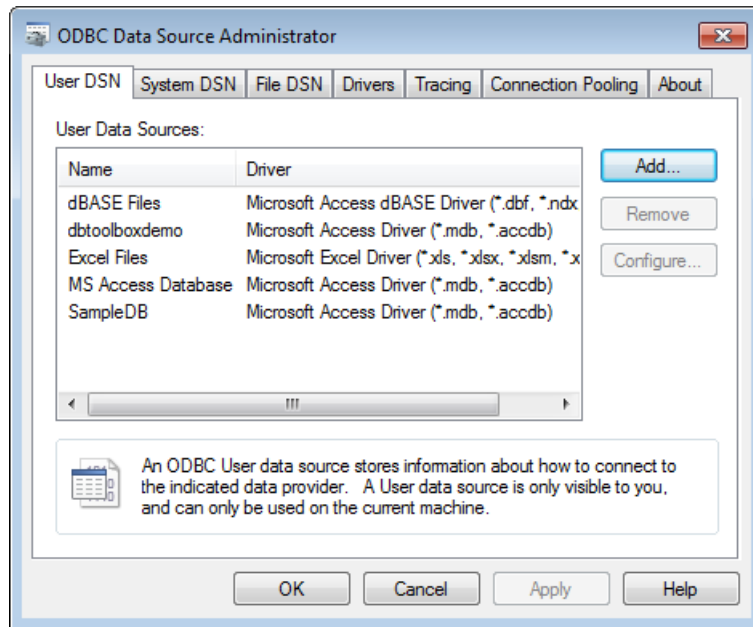


- 7 Enter dbtoolboxdemo as the data source name.
- 8 Enter tutorial database as the description.
- 9 Select the database for this data source to use. For some drivers, you can skip this step. If you are unsure about skipping this step, consult your database administrator.
 - a In the ODBC Microsoft Access Setup dialog box, click **Select** to open the **Select Database** dialog box.



- b** Specify the database you want to use. For the dbtoolboxdemo data source, select `tutorial.mdb`.
 - c** If your database is on a system to which your PC is connected:
 - i** Click **Network**. The Map Network Drive dialog box opens.
 - ii** Specify the folder containing the database you want to use.
 - iii** Click **Finish**.
 - d** Click **OK** to close the Select Database dialog box.
- 10** In the ODBC Microsoft Access Setup dialog box, click **OK**.
 - 11** Repeat steps 7 through 10 with the following changes to define the data source for any additional databases that you want to use.

The ODBC Data Source Administrator dialog box displays the `dbtoolboxdemo` and any additional data sources that you have added in the **User DSN** tab.



12 Click **OK** to close the dialog box.

Configure JDBC Data Sources

- 1** Find the name of the JDBC driver file. This file is provided by your database vendor. The name and location of this file differ for each system. If you do not know the name or location of this file, consult your database administrator.

Caution If you have previously used Visual Query Builder (querybuilder) to access a JDBC data source, before starting Database Explorer for the first time, you must execute the following command:

```
setdbprefs('JDBCDataSourceFile', '')
```

Then follow these instructions to set up the JDBC data source using Database Explorer.

2 Specify the location of the JDBC drivers file in the MATLAB Java class path by adding this file's path to the `javaclasspath.txt` file. MATLAB loads the static class path at the start of each session. The static path offers better class loading performance than the dynamic path. To add folders to the static path, create the file `javaclasspath.txt`, and then restart MATLAB.

- a** Create an ASCII file in your preferences folder named `javaclasspath.txt`. To view the location of this folder, type:

```
prefdir
```

- b** Each line in the file is the path name of a folder or jar file. For example:

```
d:\work\javaclass
```

To simplify the specification of folders in cross-platform environments, use any of these macros: `$matlabroot`, `$arch`, or `$jre_home`. You can also create a `javaclasspath.txt` file in your MATLAB startup folder. Classes specified in this file override classes specified in the `javaclasspath.txt` file in the preferences folder.

Requirement: MATLAB reads the static class path only at startup. If you edit `javaclasspath.txt` or change your `.class` files while MATLAB is running, you must restart MATLAB to put those changes into effect.

If the drivers file is not located where `javaclasspath.txt` indicates, errors do not appear, and Database Explorer does not establish a database connection.

For more information, see “Bringing Java Classes into MATLAB Workspace”.

- 3** Close the open database, `tutorial.mdb`, in the database program.
- 4** Start Database Explorer by clicking the **Apps** tab on the MATLAB Toolstrip and then selecting **Database Explorer** from the **Database Connectivity and Reporting** section in the apps gallery. Alternatively, at the command line, enter:

dexplore

- 5 Click the **Database Explorer** tab and then select **New > JDBC** to open the **Create a New JDBC data source** dialog box.

- 6 Use the following table to set up JDBC drivers for use with Database Explorer.
 - a Using the **Create a New JDBC data source** dialog box, this table describes the fields that you use to define your JDBC data source. For examples of syntax used in these fields, see “JDBC Driver Name and Database Connection URL” on the database function reference page.

Field	Description
Data Source Name	The name you assign to the data source. For some databases, the Name must match the name of the database as recognized by the machine it runs on.
Vendor	<p>The vendor's name for the data source. When using Other:</p> <ul style="list-style-type: none"> • Driver — The JDBC driver name (sometimes referred to as the class that implements the Java SQL driver for your database). • URL — The JDBC URL object, of the form <code>jdbc:subprotocol:subname</code>. <i>subprotocol</i> is a database type. <i>subname</i> can contain other information used by Driver, such as the location of the database and/or a port number. It can take the form <code>//hostname:port/databasename</code>. <hr/> <p>Note When using Other as the Vendor, your driver manufacturer's documentation specifies the Driver and URL formats. You might need to consult your database system administrator for this information.</p> <hr/>
Server Name	Server name.
Port Number	Server port number.
Authentication Type	(Microsoft SQL Server only) Server or Windows authentication.
Driver Type	(Oracle only) Driver type is thin or oci .
Username	User name to access the database.
Password	Password.
Database	Database name.

- b** In the **Create a New JDBC data source** dialog box, click **Save**.
- c** If this is the first time you are creating a data source using Database Explorer, the New file to store JDBC connection parameters dialog box opens. Use this dialog box to create a MAT-file that saves your specified data source information for future Database Explorer sessions.

Navigate to the folder where you want to put the MAT-file, specify a name for it that includes a `.mat` extension, and click **Save**.

- d** Test the connection by clicking **Test**.

If your database requires a user name and password, a dialog box prompting you to supply them opens. Enter values into these fields and click **OK**.

A confirmation dialog box states that the database connection succeeded.

- e** To add more data sources, repeat steps 5 and 6 for each new data source.

Note You can use tabs in Database Explorer to access different data sources. All of the data sources created using Database Explorer are stored in a single MAT-file for easy access. This MAT-file name is stored in `setdbprefs('JDBCDataSourceFile')` and is valid for all MATLAB sessions.

Starting the Database Toolbox and Visual Query Builder Software

To use the Database Toolbox software, enter its functions into the MATLAB Command Window. For example, to start Visual Query Builder, enter the `querybuilder` function into the MATLAB Command Window.

Database Toolbox Workflows

In this section...
“Importing Database Data” on page 1-17
“Exporting Data to a Database” on page 1-17

Importing Database Data

Here is the workflow for importing data from a database into the MATLAB workspace:

- 1 Construct a query to import data from a data source.
- 2 Define a MATLAB variable to store the results of the query.
- 3 Run the query.
- 4 (Optional) Use MATLAB functions to manipulate the data.

To import database data into the MATLAB workspace, you can use

- Visual Query Builder — For more information, see “Using Queries to Import Database Data” on page 1-22.
- Database Explorer — For more information, see “Using Database Explorer to Import Database Data” on page 1-19

Exporting Data to a Database

Here is the workflow for exporting data from the MATLAB workspace to a database:

- 1 Construct a query to export data to a data source.
- 2 Specify the MATLAB variable that contains the data to export.
- 3 Run the query.

For more details about how to use Visual Query Builder to export data from the MATLAB workspace to a database, see “Using Queries to Export Data to Databases” on page 1-29.

Using Database Explorer to Import Database Data

`dexplore` starts Database Explorer, which is the Database Toolbox app for connecting to a database and importing data to the MATLAB Workspace.

Alternatively, you can start Database Explorer by selecting **Database Explorer** from the **Database Connectivity and Reporting** section of the **Apps** tab in the MATLAB Toolstrip.

For more information on Database Explorer, after starting Database Explorer, click **Help** on the Database Explorer Toolstrip.

Database Toolbox Functions Versus Visual Query Builder

The table describes tasks you can perform with Database Toolbox functions and tasks you can perform with Visual Query Builder.

Database Toolbox Functions	Visual Query Builder
Retrieve large data sets or partial data sets in a single <code>fetch</code> command, or in discrete amounts using multiple fetches.	Import data from relational databases into the MATLAB workspace by selecting information from lists to build queries.
Dynamically import data into the MATLAB workspace.	Display retrieved information in relational tables, reports, and charts.
Modify SQL queries in MATLAB statements.	Easily build SQL queries and exchange data between databases and the MATLAB workspace.
Write MATLAB files and applications that access databases.	View and edit SQL statements for queries generated with VQB.
Replace existing records in databases with data from the MATLAB workspace.	Automatically generate a MATLAB file that consists of Database Toolbox functions that perform queries you built using VQB.
Export binary data or other data types.	Export data from the MATLAB workspace into new records in a database.
Access database metadata.	

Limitations of Visual Query Builder

You can use Visual Query Builder for most, but not all, of the tasks you can do with Database Toolbox functions. The limitations of Visual Query Builder are as follows:

- You cannot use Visual Query Builder to replace existing data in a database with data from the MATLAB workspace. Use the `update` function instead.

- You cannot use Visual Query Builder to export binary data. Instead, use the `fastinsert` function.
- You cannot use Visual Query Builder to access database metadata.

Working with Visual Query Builder

In this section...

- “Using Queries to Import Database Data” on page 1-22
- “Saving Queries” on page 1-27
- “Running Saved Queries” on page 1-28
- “Editing Queries” on page 1-28
- “Clearing Variables from the VQB Data Area” on page 1-28
- “Using Queries to Export Data to Databases” on page 1-29
- “Exiting Visual Query Builder” on page 1-32

Using Queries to Import Database Data

You can graphically construct and run SQL queries to import database data using:

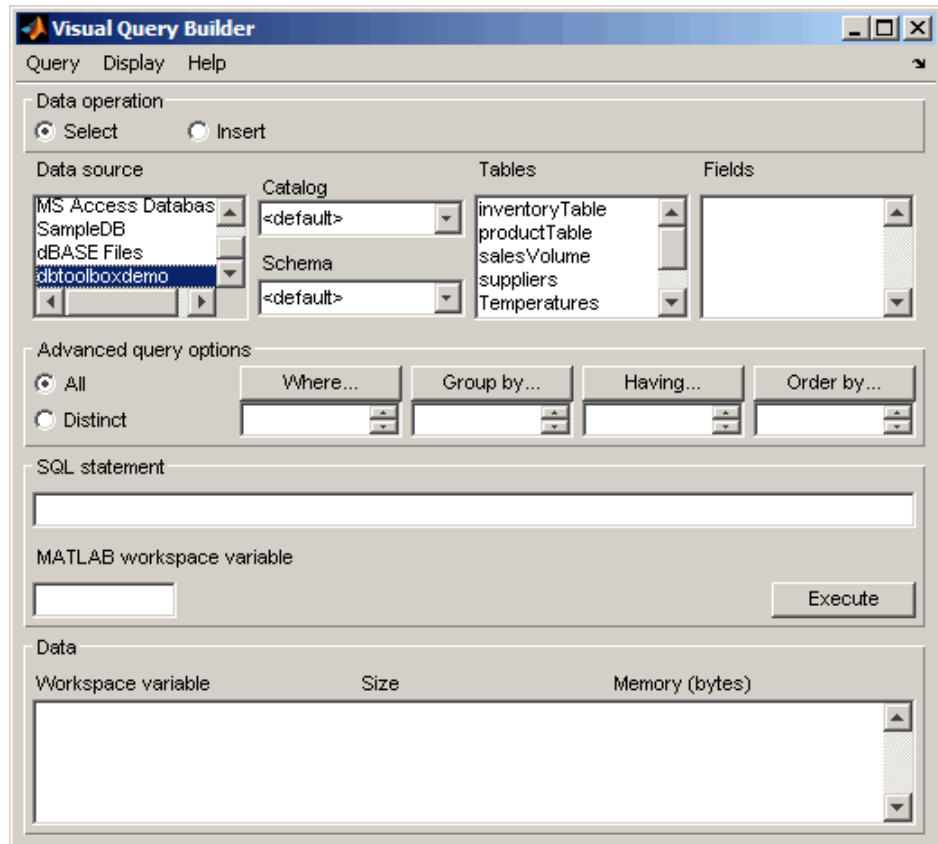
- Visual Query Builder (querybuilder)
- Database Explorer

`dexplore` starts Database Explorer, which is the Database Toolbox app for connecting to a database and importing data to the MATLAB Workspace. Alternatively, you can start Database Explorer by selecting **Database Explorer** from the **Database Connectivity and Reporting** section of the **Apps** tab in the MATLAB Toolstrip. For more information on Database Explorer, after starting Database Explorer, click **Help** on the Database Explorer Toolstrip.

To create and run a query using Visual Query Builder to import data from a database into the MATLAB workspace:

- 1 Select data from a database by clicking the **Select** button under **Data operation**. The data sources that you defined in “Configuring Your Environment” on page 1-4 appear.
- 2 Select `dbtoolboxdemo` as the data source from which to import data.

After you select a data source, the catalog, schema, and tables for your specified data source appear in the **Catalog**, **Schema**, and **Tables** fields.



- 3 Accept the default values <default> for the **Catalog** and **Schema** fields. Setting these fields to the default values indicates that you have not specified a catalog or schema.

Tip To populate the **VQB Schema** and **Catalog** fields, you must associate your user name with schemas or catalogs before starting VQB.

- To specify a **Catalog**, select one from the list, and then select a schema from within that catalog. The **Schema** field updates to reflect your selections.
 - Alternatively, you can select a schema without specifying a catalog; that is, when the **Catalog** field set to <default>. The **Tables** field updates to reflect the schema you selected.
-

- 4** In the **Tables** list, select `salesVolume` as the table that contains the data you want to import.

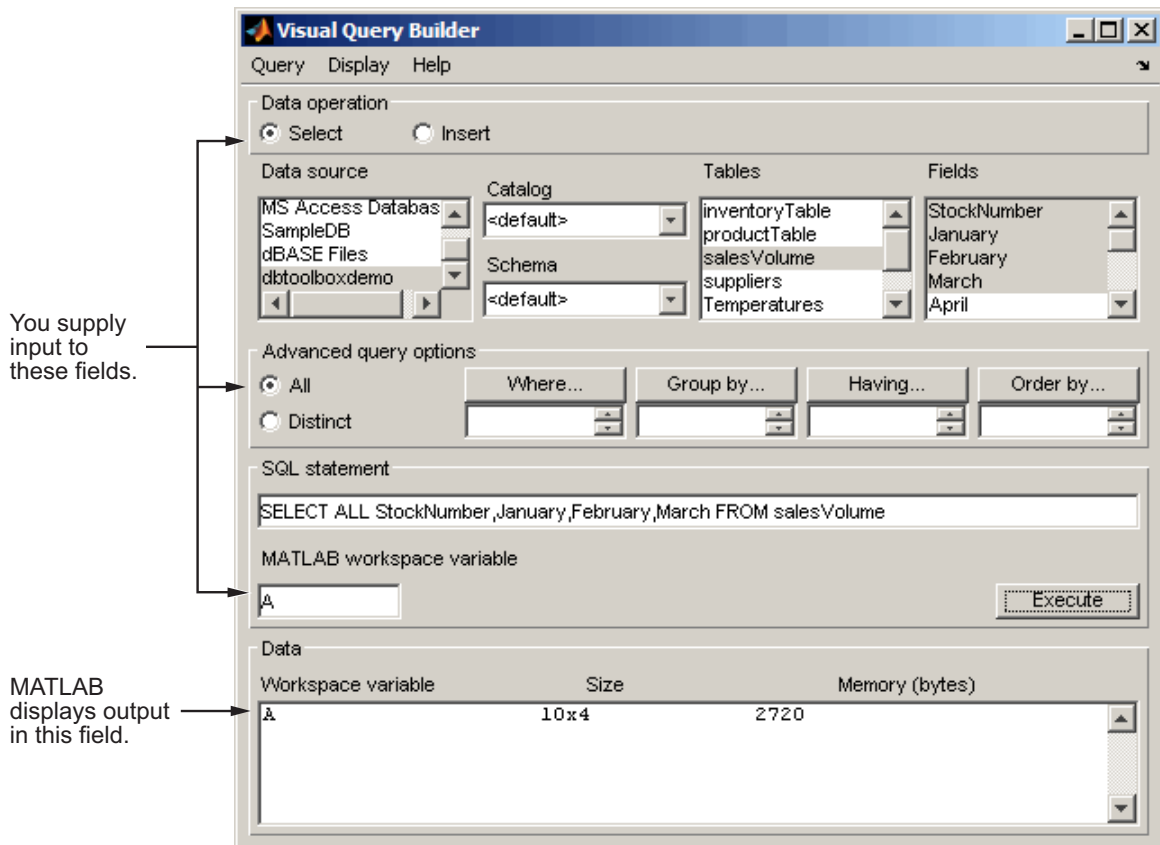
The set of **Fields** (column names) in the table appears.

- 5** In the **Fields** list, select `StockNumber`, `January`, `February`, and `March` as the fields that contain the data you want to import.

Tip To select more than one field, hold down the **Ctrl** or **Shift** key while selecting multiple fields. To clear an entry, use **Ctrl+click**.

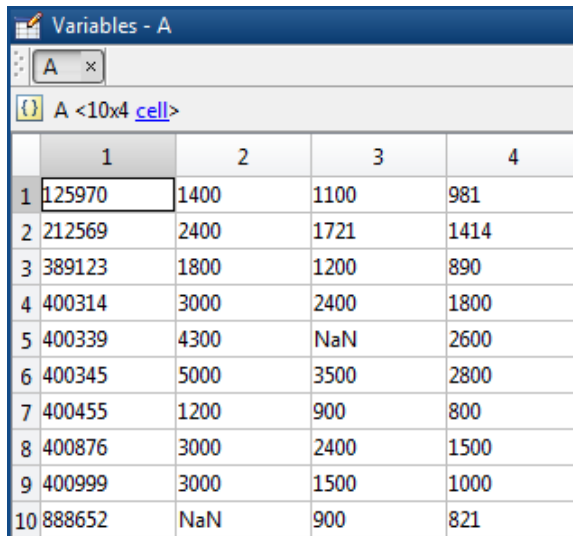
VQB adds each field you select to the query in the **SQL statement** field.

- 6** Enter the name `A` in the **MATLAB workspace variable** field. `A` is a cell array that stores the data that the query returns.
- 7** Click **Execute** to run the query and import the data. The **Data** field displays information about the query result.



- 8** Double-click **A** in the **Data** area. The contents of **A** appear in the Variables editor, where you can view and edit the data. In this example, sales for item 400876 are 3000 in January, 2400 in February, and 1500 in March.

For more information about using the Variables editor, see “View, Edit, and Copy Variables”.



The screenshot shows the MATLAB Variables window for a variable named 'A'. The window title is 'Variables - A'. Below the title bar, there is a tab labeled 'A' with a close button. Below the tab, there is a label 'A <10x4 cell>'. The main area of the window displays a table with 10 rows and 4 columns. The columns are labeled 1, 2, 3, and 4. The rows are labeled 1 through 10. The data in the table is as follows:

	1	2	3	4
1	125970	1400	1100	981
2	212569	2400	1721	1414
3	389123	1800	1200	890
4	400314	3000	2400	1800
5	400339	4300	NaN	2600
6	400345	5000	3500	2800
7	400455	1200	900	800
8	400876	3000	2400	1500
9	400999	3000	1500	1000
10	888652	NaN	900	821

Alternatively, you can view the contents of A by entering A in the MATLAB Command Window.


```

>> A

A =

    [125970]    [1400]    [1100]    [ 981]
    [212569]    [2400]    [1721]    [1414]
    [389123]    [1800]    [1200]    [ 890]
    [400314]    [3000]    [2400]    [1800]
    [400339]    [4300]    [ NaN]    [2600]
    [400345]    [5000]    [3500]    [2800]
    [400455]    [1200]    [ 900]    [ 800]
    [400876]    [3000]    [2400]    [1500]
    [400999]    [3000]    [1500]    [1000]
    [888652]    [ NaN]    [ 900]    [ 821]

>> |

```

Saving Queries

- 1 Select **Query > Save**. The Save SQL Statement dialog box appears.
- 2 Enter a name (without spaces) for the query into the **File name** field and click **Save**. Save the query as `basic.qry`.

Note When you save a **Select** query (a query that imports data), MATLAB does not save your specified preferences or the workspace variable that contains the query results. This prevents you from inadvertently overwriting an existing variable in the MATLAB workspace when you run a saved query.

When you save an **Insert** query (a query that exports data), MATLAB saves the workspace variable whose data you exported, but does not save your preferences.

Running Saved Queries

- 1 Select **Query > Load**. The Load SQL Statement dialog box appears.
- 2 Select the name of the query you want to load and click **Open**. The VQB fields reflect the values for the saved query.
- 3 Run a **Select** query to import data into the MATLAB workspace, or an **Insert** query to export data from the MATLAB workspace.
 - To run a **Select** query, use the **MATLAB workspace variable** field to assign a variable to the data and click **Execute**.
 - For an **Insert** query, the saved query may include a workspace variable, which appears as part of the **MATLAB command** field. Type that variable name or enter a new name in the **MATLAB workspace variable** field. Press **Return** or **Enter** to see the updated **MATLAB command**.
- 4 Click **Execute** to run the query.

Tip You can generate a file that runs the query from the MATLAB Command Window in the future. For more information, see “Saving Queries in Files”.

Editing Queries

Edit a query using one of the following options:

- Changing your selections
- Editing the **SQL statement** field
- Editing the **MATLAB command** field

Clearing Variables from the VQB Data Area

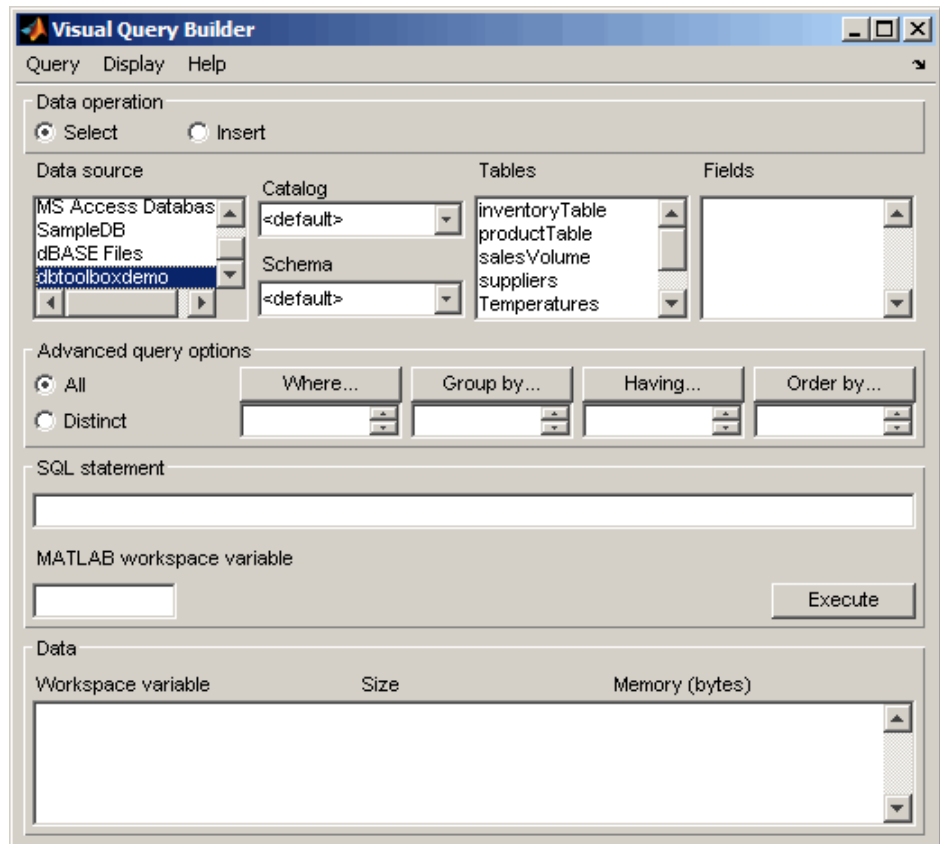
Variables in the **Data** area include those you assigned in the Command Window and those that contain query results. The variables do not appear in the **Data** area until you execute a query. They then remain in the **Data**

area until you clear them. To clear the variables, run the `clear` function in the Command Window.

Using Queries to Export Data to Databases

To build, run, and save a query that exports data from the MATLAB workspace into new rows in a database:

- 1 Select **Data Operation** > **Insert** to select data to export.
- 2 Select `dbtoolboxdemo` as the data source to which to export data from the **Data source** list box. The **Catalog**, **Schema**, and **Tables** fields for `dbtoolboxdemo` appear.



3 Do not specify values for **Catalog** and **Schema**.

4 In the **Tables** list box, select `inventoryTable` as the table to which you want to export data from the MATLAB software.

The set of **Fields** (column names) in your selected table appears.

5 In the **Fields** list box, select `productNumber`, `Quantity`, and `Price` as the fields to which you want to export data from the MATLAB software.

VQB adds each field you select to the query in the **MATLAB command** field.

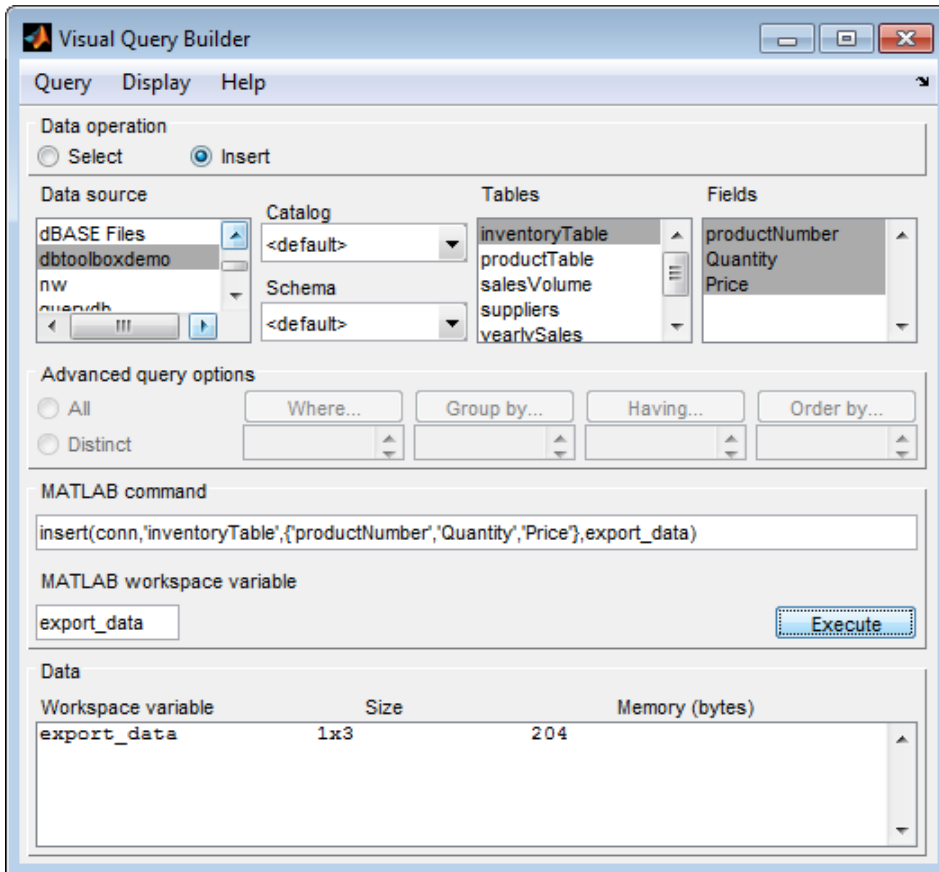
- 6 In the MATLAB workspace, assign the data you want to export to a cell array, `export_data`.

```
export_data = {14,1500,18.50};
```

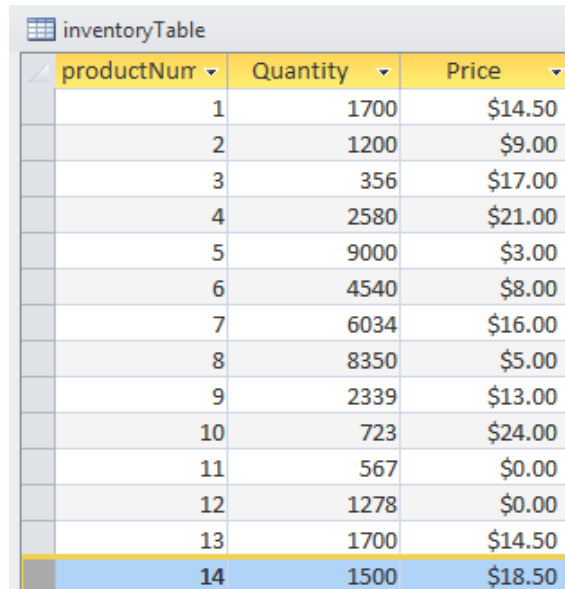
- 7 In the **MATLAB workspace variable** field, enter the name of the variable containing data to export, `export_data`. Press **Enter** or **Return** to view the **MATLAB command** that exports the data.

- 8 Click **Execute** to run the query to export the data.

Information about the exported data appears in the **Data** area.



- 9 View the inventoryTable table in the Microsoft Access database to verify the query results.



productNurr	Quantity	Price
1	1700	\$14.50
2	1200	\$9.00
3	356	\$17.00
4	2580	\$21.00
5	9000	\$3.00
6	4540	\$8.00
7	6034	\$16.00
8	8350	\$5.00
9	2339	\$13.00
10	723	\$24.00
11	567	\$0.00
12	1278	\$0.00
13	1700	\$14.50
14	1500	\$18.50

- 10 To save this query, select **Query > Save** and name it `export.qry`.

Exiting Visual Query Builder

To exit Visual Query Builder, select **Query > Exit**.

Learning More

In this section...
“Product Help” on page 1-33
“MathWorks Online” on page 1-33

Product Help

More information is available with your product installation. In the MATLAB Toolstrip, select **Help > Product Help** for help, and then click **Database Toolbox** in the **Help** window.

MathWorks Online

For additional information and support, go to the MathWorks® Database Toolbox web site at <http://www.mathworks.com/products/database/>.

C

clearing variables from Data area 1-28

D

Data area in VQB

clearing 1-28

example 1-24

data sources

about 1-4

selecting for VQB 1-22

Database Toolbox

starting 1-16

drivers

about 1-4

E

editing queries 1-28

examples

setting up 1-6

executing queries

VQB 1-24

exiting

Visual Query Builder 1-32

exporting data

using VQB 1-29

F

feature 1-16

fields

selecting for VQB 1-24

I

importing data

using VQB 1-22

L

loading saved queries 1-28

M

MATLAB

workspace variables in VQB 1-24

matlabroot 1-6

multiple entries, selecting 1-24

Q

queries

creating with VQB 1-22

editing 1-28

executing 1-24

executing for export 1-31

exporting with VQB 1-29

loading saved queries 1-28

results

VQB 1-24

saving 1-27

R

results

from query 1-24

viewing 1-25

retrieving data

from database 1-22

running queries 1-24

S

saving queries 1-27

selecting multiple entries in VQB 1-24

speed

database access 1-5

starting

Database Toolbox 1-16

T

tables

 selecting for VQB 1-24

V

Visual Query Builder

 exiting 1-32

 limitations 1-20

 starting 1-16

W

workspace variables in VQB 1-24

 clearing from Data area 1-28