8

Structured Query Language

This chapter describes Structured Query Language (SQL), Open Database Connectivity (ODBC), and accessing Citadel data using both SQL and ODBC.

Introduction

The Citadel historical database includes an Open Database Connectivity (ODBC) driver, which enables other applications to directly retrieve data from Citadel using Structured Query Language (SQL) queries.

What is ODBC?

ODBC is a standard developed by Microsoft. It defines the mechanisms for accessing data residing in database management systems (DBMSs). Nearly all Windows-based applications that can retrieve data from a database support ODBC.

What is SQL?

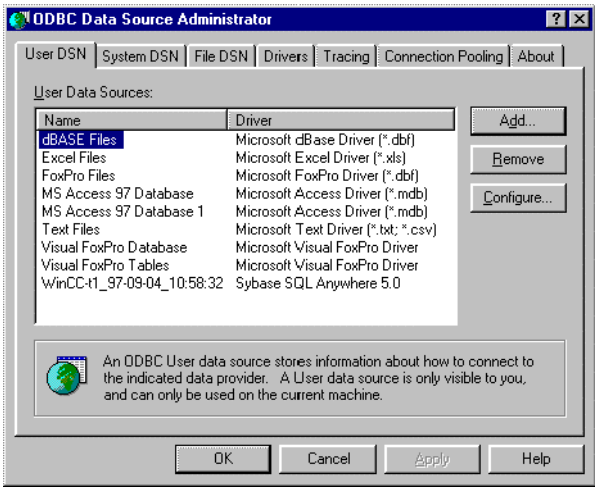
Structured Query Language (SQL) is an industry-standard language used for retrieving, updating, and managing data. In Lookout*Direct*, you can use SQL to build queries to extract data from Citadel. The Citadel ODBC driver also includes many built-in data transforms to simplify statistical analysis of retrieved data.

Creating a Citadel ODBC Data Source

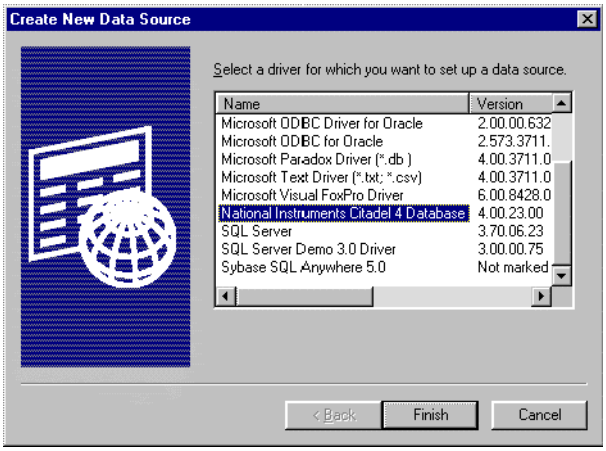
Use the following procedure to create a Citadel ODBC data source for use with Lookout*Direct*.

1. Click the Windows **Start** button and select **Settings»Control Panel**.
2. Run the ODBC applet. It might be called **ODBC** or **ODBC Data Sources** or something similar, depending on your operating system version number.

Note Shut down all ODBC applications, such as databases, spreadsheets, word processors, and Microsoft Query, before you run the ODBC applet.



3. Select the **User DSN** tab or the **System DSN** tab, depending on which type of data source you want to create. User DSNs are only visible to the user who created them on the current machine. System DSNs are available to all users on the current machine.
4. Click on the **Add** button. The following dialog box appears.



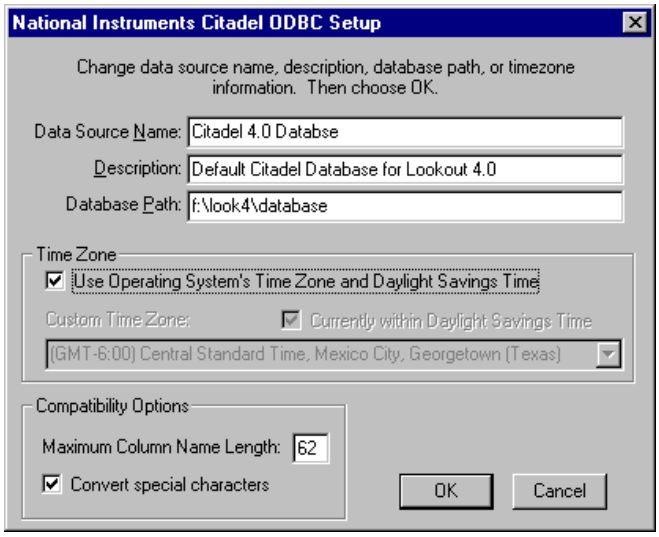
5. Select **National Instruments Citadel 4 Database**, then click **Finish**.

6. In the **National Instruments Citadel ODBC Setup** dialog box, fill in the **Data Source Name**, **Description**, and **Database Path** fields.

   a. The **Data Source Name** is the name that ODBC applications use to select your data source.

   b. **Description** is a free-form text string you can enter to describe the data source.

   c. **Database Path** should match the location of the Citadel database you intend to access. Use a fully qualified path to a remote computer if you are accessing the Citadel database on a remote computer, such as `\\Tripper\c:\lookout\database`.



The **Data Source Name** must be different from any other ODBC data source name. The **Description** is arbitrary. The **Database Path** gives the location of the folder where the data for this source is stored. In Lookout*Direct* 4, it is possible to configure multiple Citadel databases. If you do so, you probably want to configure one ODBC data source for each. If you used the **Options»System** dialog box in Lookout*Direct* to configure a default Citadel database on the local computer, you should create a data source for the location specified in the **Default Path** field in the Lookout*Direct* **System Options** dialog box. If you used **File»Modify Process** in Lookout*Direct* to configure a different database for a process, you probably want to create a separate data source for that process using the location specified in the **Citadel Database Folder** field of the Lookout*Direct* **Modify Process** dialog box.

7. Click on **OK** in the **Setup** dialog box, then click on **OK** in the **ODBC Data Source Administrator** dialog box.

Note Some applications are not completely ODBC compliant. If you plan to use Microsoft Query, Microsoft Access, or Visual Basic, make sure **Maximum Column Name Length** does not exceed 62 characters. These applications cannot handle longer names. Applications that are completely ODBC compliant can handle names up to 126 characters long. All traces whose names exceed the **Maximum Column Name Length** are excluded from queries.

Because Lookout*Direct* objects may include network path information in their names, and because you can arrange Lookout*Direct* objects in hierarchies inside your processes, you may find it easy to exceed the 62 and 126 character limitations of ODBC. Consideration given to naming and organizing objects can minimize the risk of encountering this difficulty.

If you plan to use Microsoft Access or Visual Basic, select **Convert special characters** to force Lookout*Direct* names into an accepted format by replacing characters within the names with the characters in Table 8-1.

The special characters changed in Lookout*Direct* 4. If you are converting Lookout*Direct* processes from a version earlier than Lookout*Direct* 4, you might have to rewrite any SQL queries you set up in your earlier processes.

Table 8-1. Special Access SQL Characters

Special Character	Converted Character
period (.)	at sign (@)

Accessing Citadel Data

Traces Table

The ODBC driver presents Citadel data to other applications as a traces table. The table contains a field or column for each data member logged to the Citadel database and three fields you can use to specify query criteria and to time stamp retrieved data: **Interval**, **LocalTime**, and **UTCTime**.

Interval specifies the query value sample rate. **Interval** can range from 10 ms to several years. **Interval** defaults to 1 (one day). **WEEK** is a standard seven days, but **MONTH** and **YEAR** account for different month lengths and leap years.

Because Citadel is event-driven, it only logs a value when the value changes. Using **Interval**, you can query Citadel for values evenly spaced over a period of time.

LocalTime and **UTCTime** are time-stamps that indicate when values are logged. Citadel stores the time in **UTCTime** format and derives **LocalTime** from the stored time.

The following `where clause` query uses **Interval** and **LocalTime** to select data over a specified time at one-minute intervals. Notice that time and date formats are the same as those used in Lookout*Direct*.

```
SELECT * FROM Traces
WHERE LocalTime>“12/1 10:00”
   AND LocalTime<“12/2 13:00”
   AND Interval=“1:00”
```

Points Table

The Points table is used to retrieve the actual values logged by Lookout*Direct* for a data member and the times they were logged. As Lookout*Direct* logs values for data members asynchronously, there is no correlation between the timestamps for one data member and another. For this reason, when querying the Points table, you may only query one data member at a time.

The where clause using **LocalTime** and **UTCTime** is supported for the Points table; however, **Interval** is not relevant to the Points table. The data transforms are also not relevant to the Points table and are not supported.

An example of a query using the Points table could be

```
SELECT LocalTime, Pot1 FROM Points
WHERE LocalTime > "12/1 10:00" AND
LocalTime < "12/2 10:00"
```

Data Transforms

Your queries can include special commands that perform data transforms to manipulate and analyze historical data. Table 8-2 lists data transform commands.

Table 8-2. Data Transform Commands

Command	Transformation
Min{Datapoint}	Returns the minimum for *`Datapoint`* across the interval.
Max{Datapoint}	Returns the maximum for *`Datapoint`* across the interval.
Avg{Datapoint}	Returns the average for *`Datapoint`* across the interval.
Stdev{Datapoint}	Returns the standard deviation for *`Datapoint`* across the interval.
Starts{Datapoint}	Returns the number of starts (that is, the number of transitions from OFF to ON) for *`Datapoint`* across the interval. For numeric points, 0.0 is interpreted as OFF, and all other numbers are treated as ON.
Stops{Datapoint}	Returns the number of stops (that is, the number of transitions from ON to OFF) for *`Datapoint`* across the interval.
ETM{Datapoint}	Returns the amount of time *`Datapoint`* was in the ON state across the interval.
Qual{Datapoint}	There might be gaps in the historical data traces in Citadel because of machine shutdown, LookoutDirect shutdown, or a similar occurrences. Qual returns the ratio of time for which valid data exists for *`Datapoint`* across the interval to the length of the interval itself. If valid data exists for only one-half of the interval, Qual returns 0.5.

Using these data transforms, you can directly calculate and retrieve complex information from the database such as averages and standard deviations, so you do not need to extract raw data and then manipulate it in another application.

For example, you need to know how many times a compressor motor started in December. You also need to know its total runtime for the month. Use the following query to get your answers:

```
SELECT“Starts{PLC.MotorRun}”,
       “ETM{PLC.MotorRun}”
FROM Traces
WHERE LocalTime>=“12/1/95”
  AND LocalTime<“1/1/96”
  AND Interval=“31”
```

SQL Examples

The following examples are typical query statements; however, your queries might be much more involved, depending on your system requirements.

```
SELECT *
FROM Traces
```

Retrieves the current value of every data member logged to Citadel. Because your query does not occur at the same moment in time as a PLC poll, signals scanned from PLCs are not included in the retrieved data.

```
SELECT *
FROM Traces
WHERE Interval=“0:01”
```

Retrieves the value of every data member logged today in one-second increments. Notice that the interval value is enclosed in quotation marks.

```
SELECT LocalTime, “Pot1”
FROM Traces
WHERE LocalTime>“8:50”
AND Interval=“0:01”
```

Retrieves and time stamps the value of Pot1 in one-second increments from 8:50 this morning to now. Names are enclosed by quotes.

```
SELECT LocalTime, “AB1.I:3”, “Max{AB1.I:3}”
FROM Traces
WHERE LocalTime>“10/1/95”
AND LocalTime<“11/1/95”
AND Interval=“1:00”
```

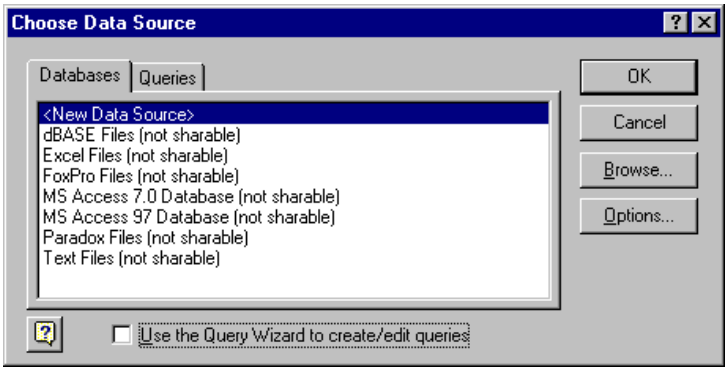
Retrieves and time stamps an Allen-Bradley PLC input in one-minute intervals for the month of October. This query also indicates the highest occurring input value of each minute.

```
SELECT LocalTime, “OVEN1_SP”, “PLC.OVEN1_PV”,
“Max{PLC.OVEN1_PV}”, “Min{PLC.OVEN1_PV}”,
“Avg{PLC.OVEN1_PV}”
FROM Traces
WHERE LocalTime>=“14:00”
   AND LocalTime <“15:00”
   AND Interval=“1:00:00”
```

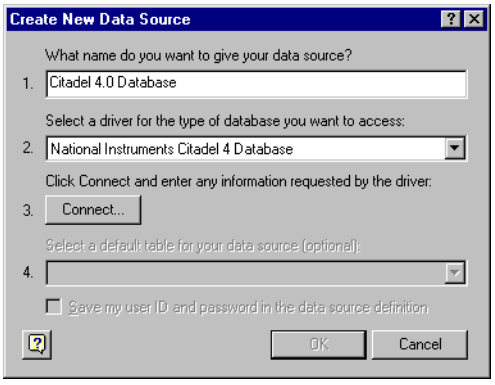
Retrieves an oven temperature at 3:00 p.m. and shows the highest, lowest, and average temperatures between 2 p.m. and 3 p.m.

Accessing Citadel Data with Microsoft Query

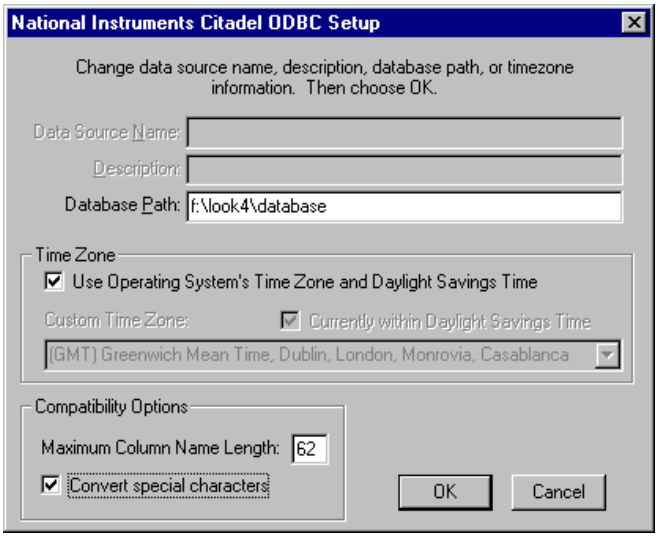
1. Launch Microsoft Query.
2. Choose **File»New.** If the Citadel data source is not visible, you must create it. If it is visible, skip to step 3.
   a. Select the `<New Data Source>` entry in the **Databases** tab. Make sure the **Use the Query Wizard to create/edit queries** option is *not* selected.



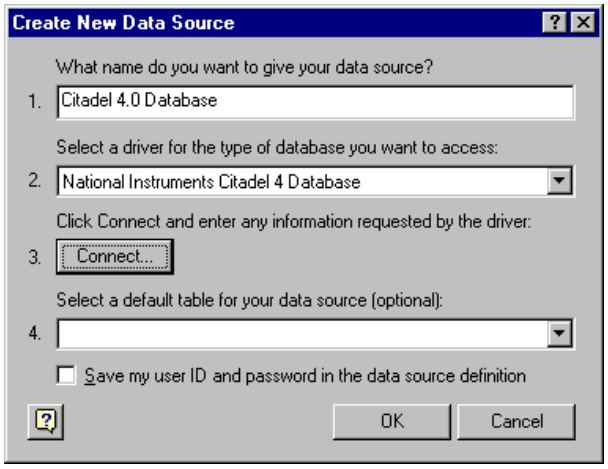
   b. The **Create New Data Source** dialog box appears. Fill in the fields as shown in the following illustration.



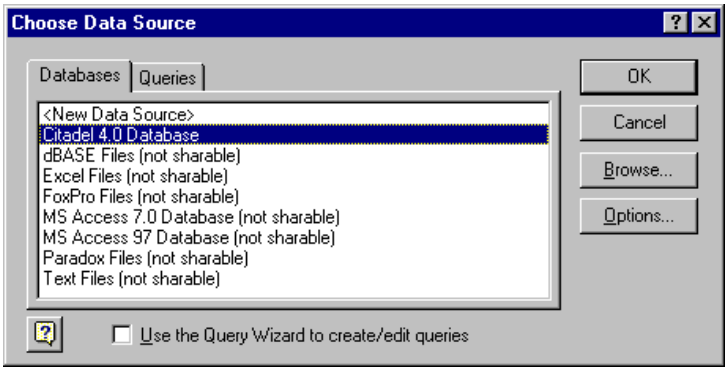
c. Click on the **Connect** button. The **National Instruments Citadel ODBC Setup** dialog box appears. Fill in the fields as shown in the following illustration. Enter the location of your database in the **Database Path** field.



d. Click on **OK**. The **Create New Data Source** dialog box reappears, and should look like the following illustration.



e. Do *not* select a default table or save your ID and password. Click on **OK**. The **Choose Data Source** dialog box appears, this time with the Citadel database as one of your data source choices. Choose Citadel as your data source and click on **OK**.

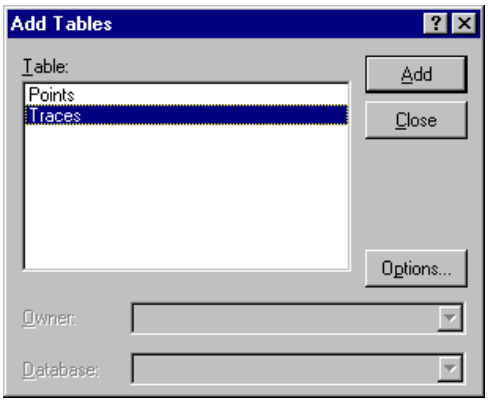


3. Select the `Citadel` data source. Make sure the **Use the Query Wizard to create/edit queries** option is *not* selected.

Note If Microsoft Query is unable to connect to Citadel, make sure you have logged data to Citadel and entered the correct database path in the **ODBC Setup** dialog box.

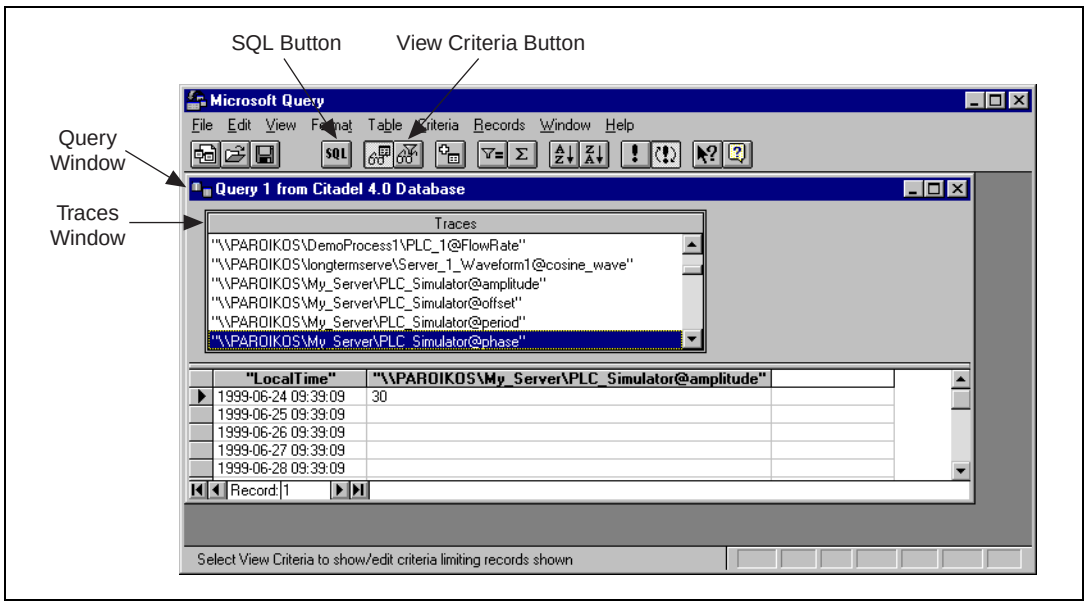
Note If Citadel is not listed in the **Data Source** dialog box, make sure you have created Citadel as a data source. See the *Creating a Citadel ODBC Data Source* section for more information.

4. In the **Add Tables** dialog box, double-click `Traces` or `Points`, depending on whether you want to view raw data (`Points`) or resampled data (`Traces`). In the following figure, `Traces` are used.



5. Close the dialog box.

   Microsoft Query presents the full Query Window with the `Traces` and `Points` tables. The names in these tables are a comprehensive list of all name values that have been logged to Citadel.



6. To view a trace, double-click on or drag the field you want to view to a blank column in the data pane. In the figure above, `LocalTime` and

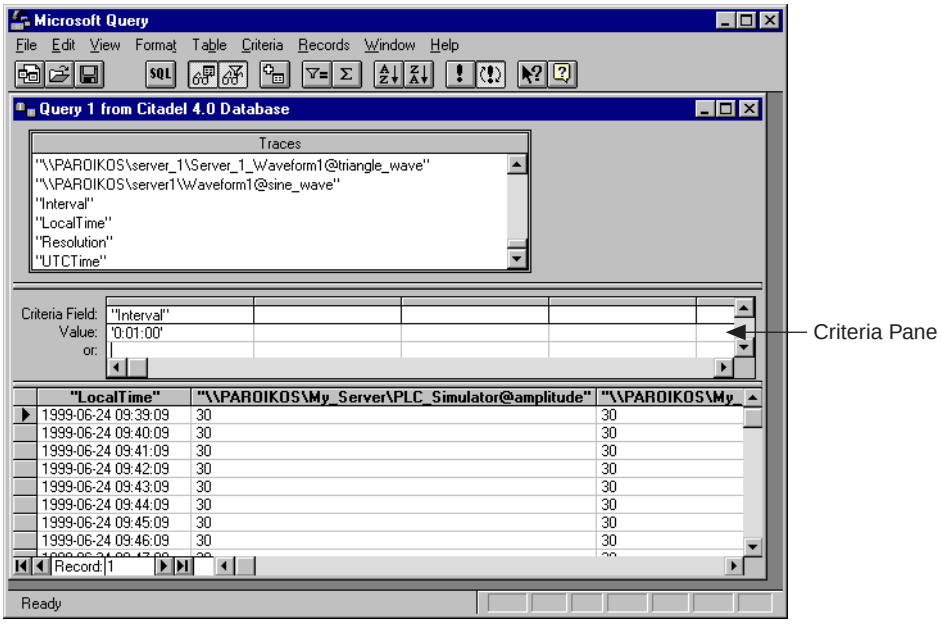
`"\\PAROIKOS\My_Server\PLC_Simulator@amplitude"` have been dragged down.

7. To view a data transform value, enter the function directly into a blank column. For example, to view the minimum value of `PLC_Simulator.amplitude`, you would enter `“min{\\PAROIKOS\My_Server\PLC_Simulator@amplitude}”`. You must include the quotation marks and braces.

   The data set in the figure above was retrieved using no specific criteria, so the ODBC driver used the default. Although there are several ways to specify criteria, this example uses the criteria pane.

8. Click on the **View Criteria** button. The pane appears in the **Query** window.

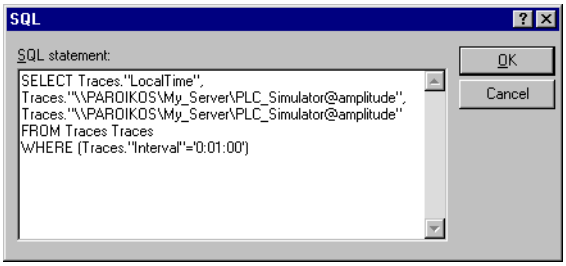
9. Add a field to the criteria pane by double-clicking on the field, or by dragging it to the blank column in the criteria pane. In the following example, an Interval criteria of one minute was chosen.



When you enter qualifying criteria values, use the syntax demonstrated in *SQL Examples* earlier in this chapter.

As soon as you specify criteria, Microsoft Query retrieves the specified data. You can save your query at any stage of development, and as you build your query, the application builds an SQL statement.

10. Click on the **SQL** button to view or edit the query statement.



Accessing Citadel Data with Microsoft Excel

1. Launch Excel.
2. Choose **Data»Get External Data»Create New Query**. This Excel command directly launches Microsoft Query.
3. Use an existing query or create a new one. See the section *Accessing Citadel Data with Microsoft Query* for information on querying.
4. When you finish building your query, return the result set with the **File»Return Data to Microsoft Excel** command. Excel prompts you with the **Get External Data** dialog box, enabling you to change or confirm the destination cells of the result set. If you want to query Citadel later to update the result set, check the **Keep Query Definition** checkbox.
5. Click on **OK** to write the data to an Excel worksheet.
6. To update your result set, select any cell within the worksheet result set, choose **Data»Get External Data**, and click on the **Refresh** button.

Accessing Citadel Data with Microsoft Access

Note The SQL/92 standard states that a delimited identifier is any string of no more than 128 characters enclosed in quotation marks. It further states that characters within a delimited identifier are exempt from SQL syntax checking. Unfortunately, Microsoft Access performs its own syntax checking for ODBC queries using a non-standard SQL syntax—even within delimited identifiers. For this reason, National Instruments provides a Convert Special Characters selection in the Citadel ODBC Setup dialog box. When selected, the ODBC driver converts the special characters to accepted Access characters. Refer to Table 8-1 for a complete list of special characters.

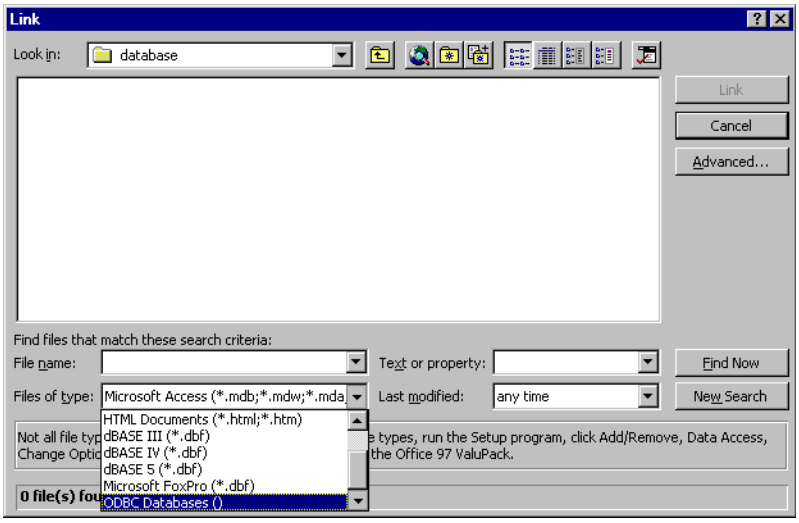
When you query Citadel using Microsoft Access, you must use the Microsoft Access non-standard SQL syntax in your select statement. Remember to consider the following elements:

- Convert special characters for Access compatibility (see Table 8-1)
- Place double quotes around LookoutDirect trace names to identify them as delimited identifiers
- Enclose identifiers in square brackets ([ ])
- Place pound signs (#) around time stamps

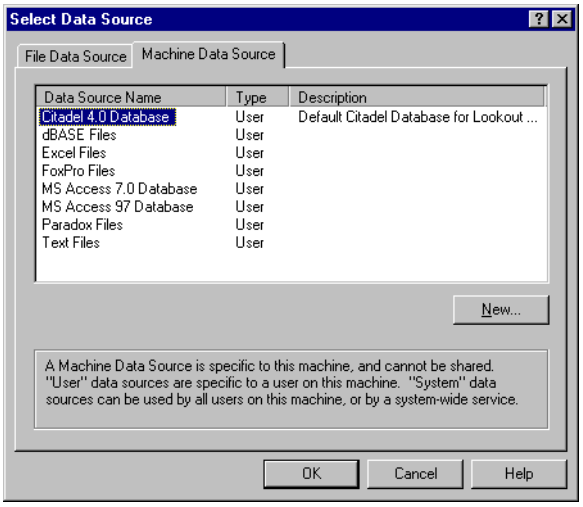
For example, to retrieve `LocalTime`, `Pot1`, and `Tiway1.V101`, where `LocalTime` is greater than 11/20/95 18:00:00, and where `Interval` is one second, enter the following query:

```
SELECT LocalTime, ["Pot1"], ["Tiway1@V101"]
FROM Traces
WHERE LocalTime > #11/20/95 6:00:00 PM#
AND Interval = '0:01'
```

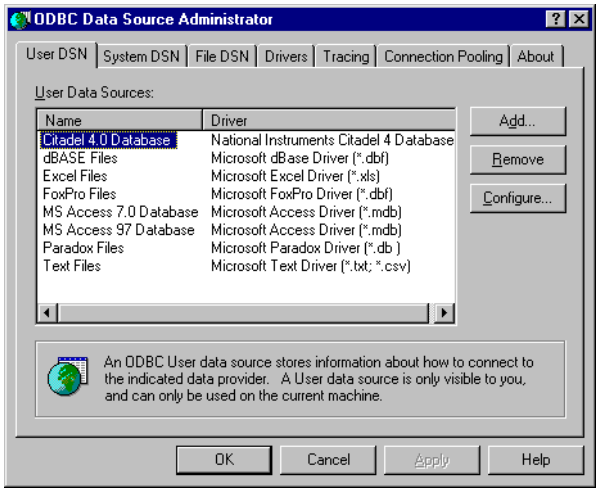
1. To query Citadel from within Microsoft Access, open a database and select **File»Get External Data»Link Table**. The **Link** dialog box appears, as shown in the following illustration.



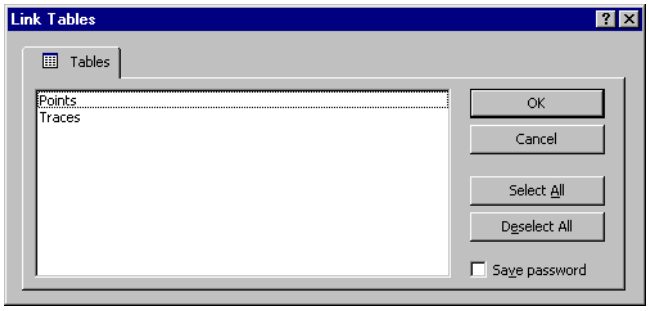
2. Set the **Files of type** field to **ODBC databases()**. The **Select Data Source** dialog box appears.



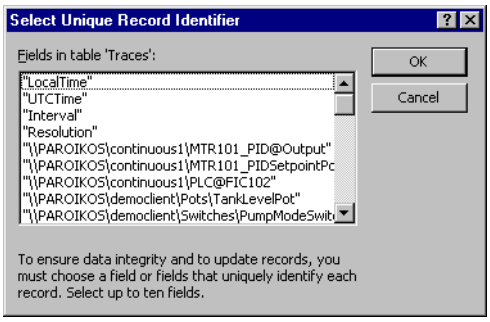
3. Choose the **Machine Data Source** tab.
4. Highlight **Citadel 4.0 Database.dsn**, as shown in following illustration. (This example assumes you have created a Citadel data source as described in the *Creating a Citadel ODBC Data Source* section. If you have not created a Citadel data source, you may not be able to complete this example exercise.)



5. Click on **OK**. The **Link Tables** dialog box appears.



6. Choose **Traces**. The new table links to your database.
7. The **Select Unique Record Identifier** dialog box appears. Click on **Cancel**, because LookoutDirect does not support unique record identifiers.



8. At this point, you can build queries in Access to extract data directly from the Citadel database. See your Microsoft Access documentation for more detailed information on this process.

Accessing Citadel Data with Visual Basic

Note Visual Basic software relies on Microsoft Access DLLs for performing ODBC queries. Because it uses the Access non-standard SQL syntax, be sure that **Convert Special Characters** is selected in the Citadel **ODBC Setup** dialog box. Refer to the note under *Accessing Citadel Data with Microsoft Access.*

You use the Citadel ODBC Driver in Visual Basic the same way you would use any other ODBC driver. To retrieve and view data, you have to create a data control and at least one text control.

1. Place a data control on an open form.
2. Set its **Connect** property to `DSN=Citadel` and double-click on its **Record Source** property to identify `Traces` as its source table.
3. If you want to select all of the data for all traces in the Citadel database, leave the **Record Source** property set to `Traces`. If you want to narrow your query, enter an SQL select statement in the **Record Source** property.

   For example, to retrieve `LocalTime`, `Pot1`, and `AB1.I:3` where `LocalTime` is greater than 11/20/95 18:00:00 and less than 18:30:00, and where `Interval` is one minute, enter the following query:

```
SELECT LocalTime, ["Pot1"], ["AB1@I:3"]
FROM Traces
WHERE LocalTime > #11/20/95 6:00:00 PM#
AND LocalTime < #11/20/95 6:30:00 PM#
AND Interval = '1:0'
```

Note Remember to use the Microsoft Access SQL syntax in the select statement, convert special characters for Access compatibility (see Table 8-1), place double quotes around Lookout*Direct* trace names to identify them as delimited identifiers, enclose identifiers in square brackets ([ ]), and place pound signs (#) around time stamps.

4. Place a text control on the form.
5. Set its **Data Source** property to the name of your data control—for example, `Data1`.
6. Click on the **Data Field** property to highlight it and then use the property sheet drop-down combo box to select the desired field name. All logged data members should be listed including `LocalTime`, `Interval`, and `Pot1`.
7. Repeat steps 4 through 6 for each data member you want to display on your form.

If you are using Visual Basic 4 or later, you might get a Visual Basic warning telling you `Object variable or with block variable not set`.

The following steps should help if this problem arises.

1. Make sure that the Citadel ODBC is properly installed. Select **Control Panel»32-bit ODBC** and choose the **Users DSN** tab. You should see your Citadel data source listed. If you do not, you must create it. See *Creating a Citadel ODBC Data Source* for more information.
2. Make sure you have placed a Data Control on your open form.
3. Set the `Connect` property of the Data Control to `ODBC;DSN=Citadel` (or any other name you defined in the **Users DSN** tab in step 1). You should now be able to see **`Traces`** in the **Record Source** property.