3

RETRIEVING iSERIES DATA WITH MICROSOFT QUERY

Microsoft Query, the helper application included with Microsoft Office, allows Office applications such as Word and Excel to read data from ODBC data sources. While Query retrieves and formats data for inclusion in other Office applications, it isn't a widely known stand-alone application because it is not installed by default by the Office setup program. Even if you opt to install it, Office does not give MS Query an icon in the Office folder.

The primary way for you to access Query is through the Word and Excel menus. For example, in Word, when you access the Mail Merge Helper from the Tools menu and opt to get data from an external source by the selecting the Open Data Source option from the Get Data drop-down list, Query retrieves data from the database. In Excel, when you choose Get External Data from the Data menu and select Create New Query, Query is used to bring data into the worksheet.

Because Query is such a commonly used application, learning it can help you maximize productivity. It is the primary means for bringing iSeries data into Office applications through ODBC.

This chapter helps you become familiar with ODBC and Query terminology while explaining how to install Query. Practical examples show you how to create, save, and use queries, how to join tables, and how to use filtering, grouping, totaling, and sorting. In addition, there are tips on how to speed up the design process and reduce performance issues with your iSeries. Along with the basics, there are advanced tips that specifically address Query's interaction with the iSeries.

The examples in this chapter use the sample data that has been loaded to the MC Press Online Web site (see the Introduction for instructions on how to navigate to this code). To complete these examples as shown, first load the sample data onto your iSeries. These examples cover the basics as well as some of the more advanced applications of Query. Because the sample data is as realistic as possible, you should be able to benefit immediately by using and applying your new skills to your own situation.

Before attempting the examples, make sure that you have an ODBC DSN configured that points to the data you want to retrieve from your iSeries. See chapter 2 for information on how to configure an ODBC DSN. (Note that under earlier releases of ODBC or Office, you might have to configure a "file-type" DSN for Query.)

COMING TO TERMS

ODBC and SQL use some terms that might not be familiar to you. When you are learning these new terms, it is helpful to imagine the data in an iSeries file laid out in a spreadsheet-like format. When viewed in this format, the entire spreadsheet, or file, is referred to as a *table*. Each record in the table is referred to as a *row*. A particular field is referred to as a *column*. A collection of tables is referred to as a *database*. This is generally analogous to an iSeries library.

INSTALLATION

Before you can work with Query, it has to be installed on your system. Unfortunately, it usually isn't as easy as looking for a shortcut to Query, since Microsoft does not expect you to start this program by itself. You can determine if Query is installed on your system by searching for a program called Msqry32.exe. To search your computer for this program, click Windows' **Start** button, select **Find**, and then select **Files or Folders**, as shown in Figure 3.1.
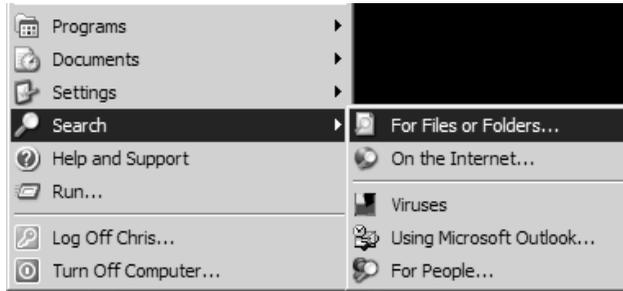
Figure 3.1: From the Start button, you can search for files or folders.

In the Named text box, enter **msqry32.exe**. In the Look In box, select **My Computer**. Click the **Find Now** button to begin the search. If the program is found, it will be listed in the box at the bottom of the Find dialog box, as shown in Figure 3.2. In that case, skip the "Installing Microsoft Query" section on the following pages and continue with the "Creating a Shortcut" section. If the program isn't found, you must install it.
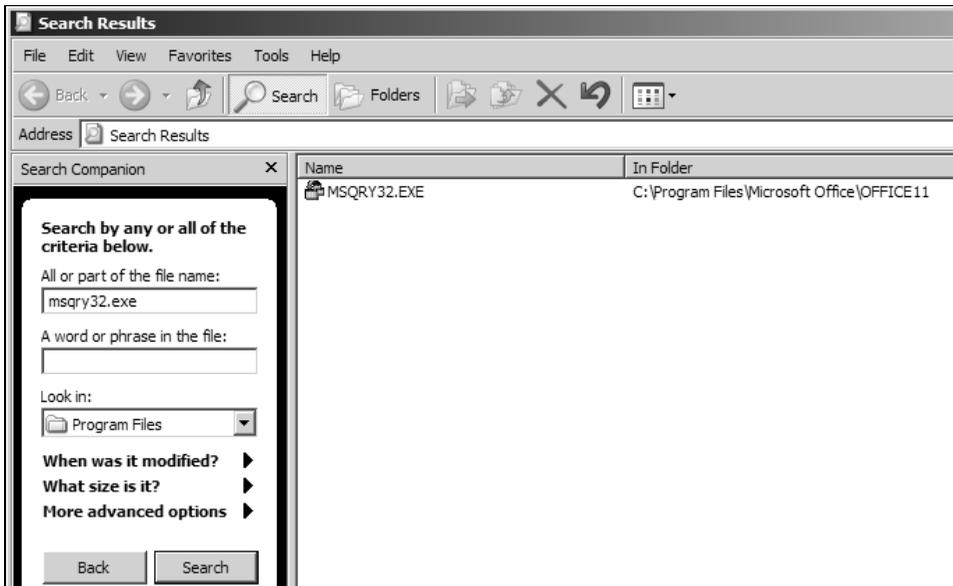
Figure 3.2: Search for Msquery32.exe.

Installing Microsoft Query

If you didn't find Query on your system, install it by running the Office setup program. Query is found under the Office Tools tab (as shown in Figure 3.3 for Office 2000/XP). Ensure that Query is set to run from your local drive by clicking the drop-down box (as shown in Figure 3.4 for Office 2003). Proceed through the setup program to install Query. Once it is installed, the next (optional) step is to create a shortcut to it.
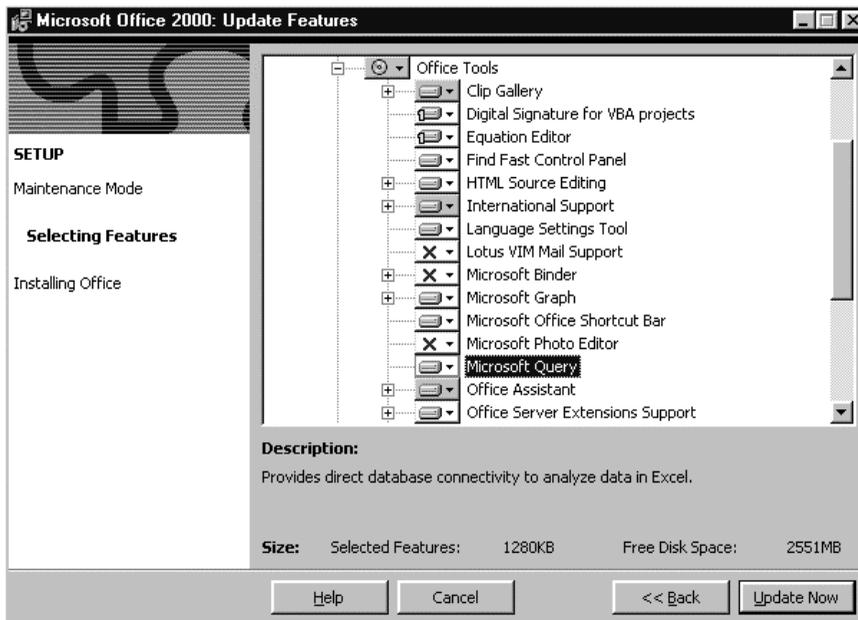
Figure 3.3: Load Query using the Office 2000 setup program.
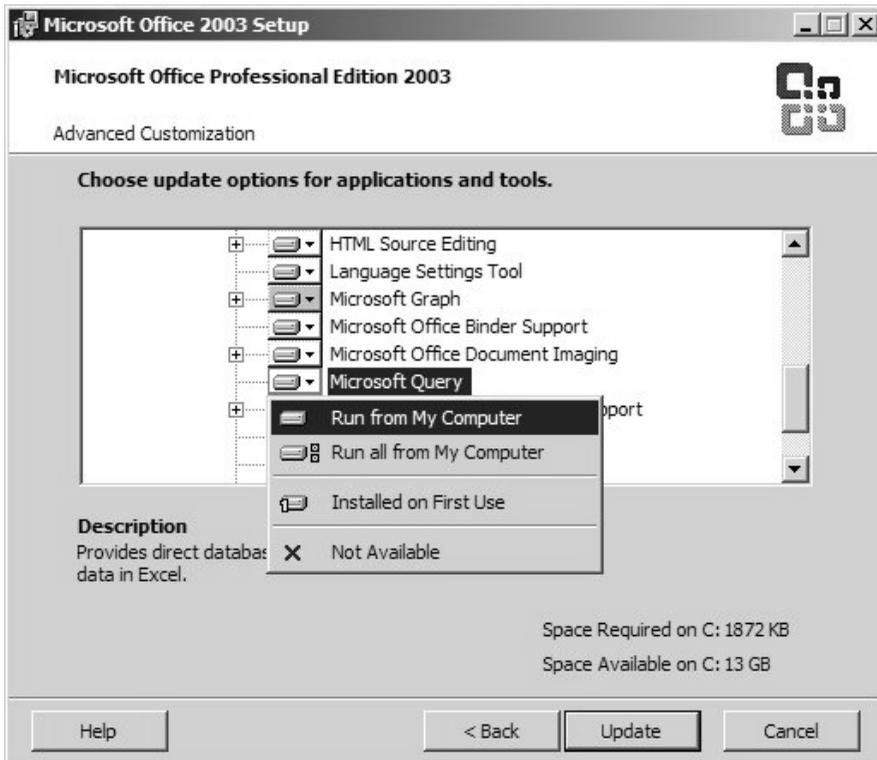
Figure 3.4: Install Query using the Office 2003 setup program.

Creating a Shortcut

Although there is no shortcut to Query by default, creating one on your desktop lets you quickly run the program without having to go into Word or Excel. One technique for creating a shortcut is to drag-and-drop using the right mouse button.

Locate program Msqry32.exe with Explorer, and click and hold the right mouse button. Without releasing the right mouse button, drag the file to the desktop. When you let go of the button, a menu appears. Select **Create Shortcut(s) Here**, as shown in Figure 3.5. You will now have a shortcut on the desktop that you can use to start Query without having to open any other Office applications.
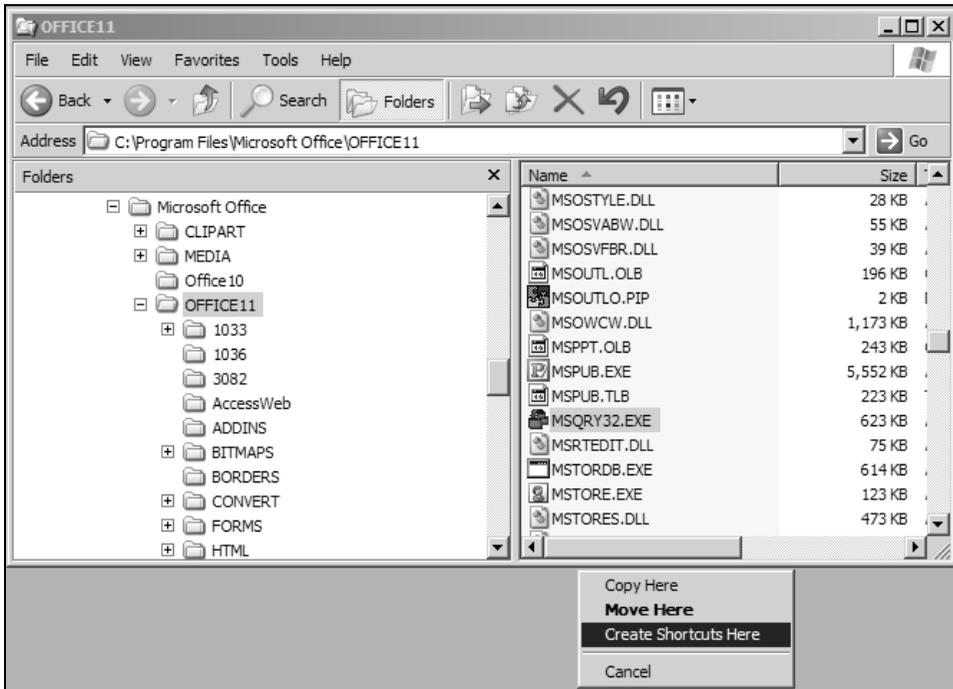
Figure 3.5: Create a shortcut to Query.

A QUICK LESSON

There are two basic methods to create a query within Microsoft Query. The easiest method is to use the Query Wizard, which lets you create a query by answering questions about the data you want to retrieve. However, this method doesn't provide you with all the available options. To access all the facilities available in Query, you need to use the "manual" (for lack of a better term) method of creating a query.

Note that it is possible to use a combination of the two methods, by creating a query with the wizard and then manually adding any additional features you need. So that you can choose the option that works best in your situation, both methods of creating queries are explored in the following sections.

Let's start with the basics, by creating and saving a simple query with the Query Wizard. To start Query, launch Msqry32.exe from its icon on the desktop (created in the previous section). Query loads and presents a blank workspace. To create a new query, click the

New Query button shown in Figure 3.6 (on the left side of the toolbar), or select New Query from the File menu.

 Figure 3.6: The New Query button creates a new query.

The first step in creating a new query is selecting a database. The databases listed in the Choose Data Source dialog box, shown in Figure 3.7, are the file DSNs that you create (as explained in chapter 2). The data source points to the library where the data you want to access resides. If you don't see your ODBC data source on this screen, it's because you either have not created it, or you created a DSN that was not a file DSN (earlier versions of OS/400 only). In either case, review the steps in chapter 2 for creating a DSN.
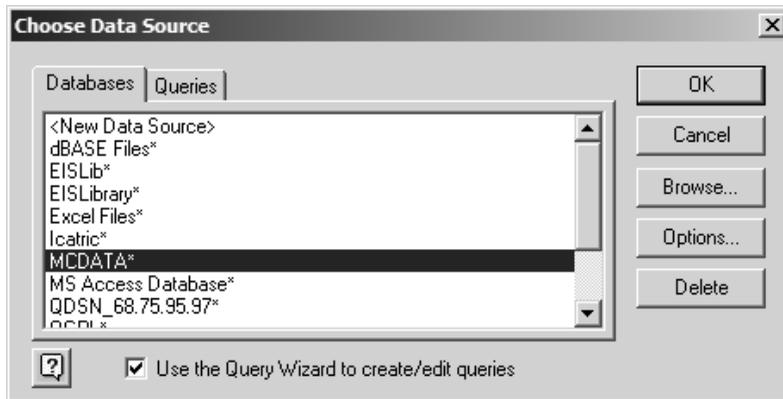
Figure 3.7: The Choose Data Source dialog box displays your ODBC data sources.

The Options button in this dialog box allows you to specify directories to search for file DSNs. Click **Options** to display the Data Source Options dialog box shown in Figure 3.8. The directories currently set on your system are listed in the box at the bottom of this dialog box. The defaults should be sufficient for most installations. If you are an administrator and you want to store your file DSNs in a central location (like a network server), you could point to it here. For now, click the **Cancel** button to return to the Choose Data Source dialog box.
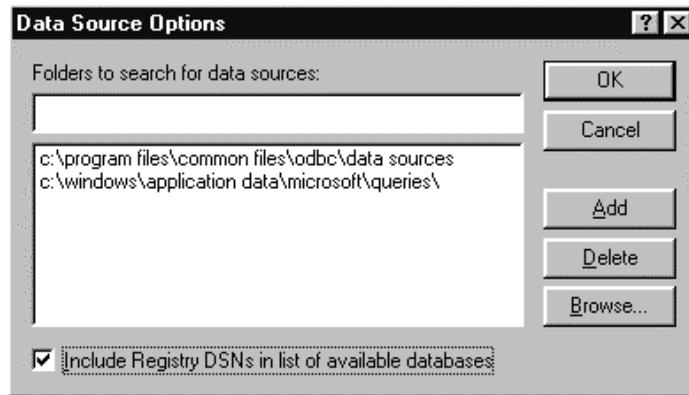
Figure 3.8: You can optionally choose a directory to search for file DSNs.

If you need to create a new DSN, select **New Data Source** from the database list and click **OK**. You can also create queries and save them for later using Query. To load a previously saved query, click the Queries tab at the top of the Choose Data Source dialog box. If you have any saved queries, they will be listed. For now, click the **Databases** tab to return to the previous screen.

CREATING A SIMPLE QUERY USING THE WIZARD

Did you notice the option to use the Query Wizard to create and edit your queries in Figure 3.7? Check this option; the first example exercise will use the Query Wizard. With this option checked, select a DSN, and click **OK** to create a new query. At this point, you might be presented with an ODBC logon screen. If so, enter your user ID and password to log on.

Now that you have logged on, you will see the first screen: Choose Columns. On this screen, you can select the tables and columns to use in your query by clicking them in the list on the left and clicking the top button with the single arrow pointing to the right, as shown in Figure 3.9.
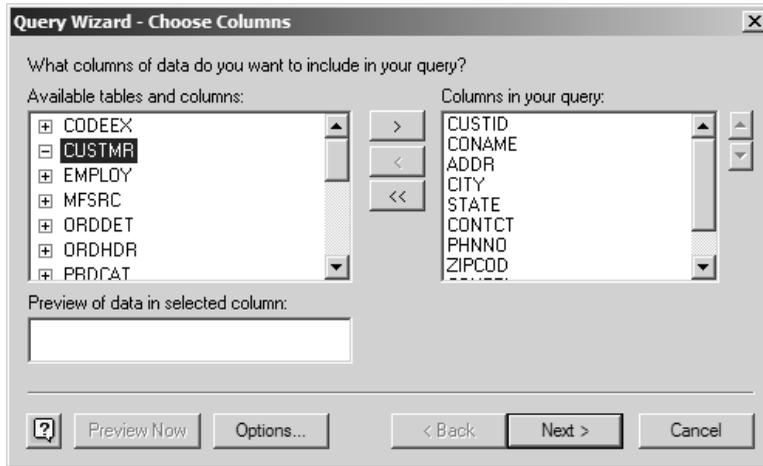
Figure 3.9: Select fields on the Query Wizard's Choose Columns screen.

You can expand the table entries to see a list of columns in the table by clicking the plus sign to the left of the table name. On some systems, however, the plus sign doesn't show up. In that case, click the area to the left of the table name to open its field list.

If you accidentally add a table you don't want, select it in the list on the right, and click the middle button, with the arrow pointing left. To remove all tables from the list on the right, click the bottom button, which has two left-pointing arrows. You may select the fields one at a time to specify the ordering, if you like.

For this example, accept all the columns from your Customer table and click the **Next** button at the bottom of the screen. This brings up the Filter Data screen shown in Figure 3.10.

On this screen, you can filter records based on values in the columns. For example, suppose you want to select only customers from California. To do so, first select **State** in the Column to Filter list. Next, to select an operator for the filter, expand the Only Include Rows Where: State drop-down list and select **Equals**. Notice that this action activates the box on the right side of the screen. Use this box to specify the value for which you want to filter. In this example, the goal is to select only customers in California. Therefore, select **CA** from the drop-down list.
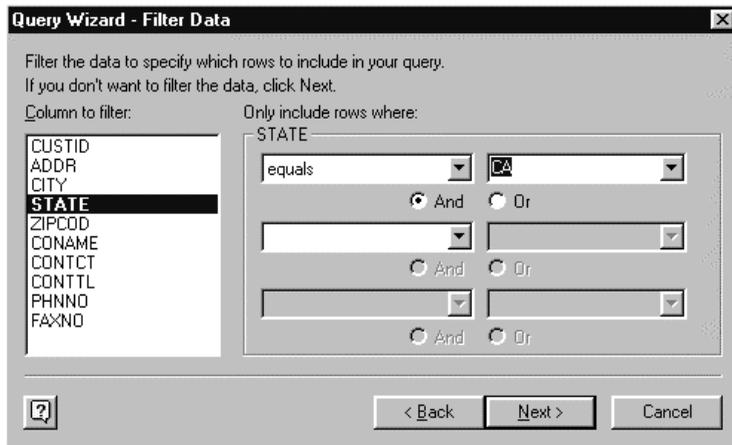
Figure 3.10: You can use the Query Wizard to filter query data.

At this point, you could enter up to two more filters. Because this example uses only one filter, however, click **Next** to go to the Sort Order screen shown in Figure 3.11.
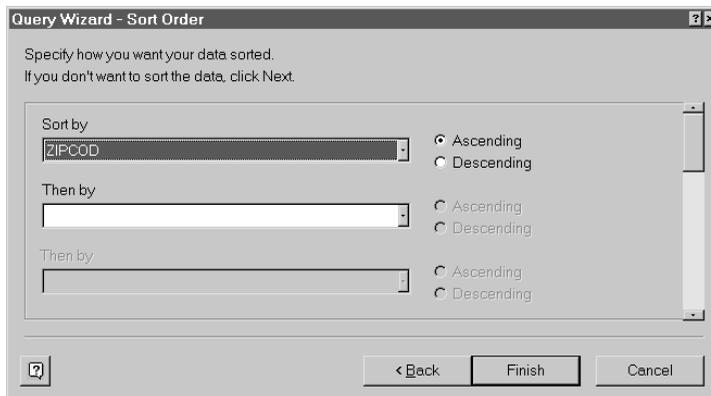
Figure 3.11: Specify a sort on this screen

On this screen, you can specify how you want the data sorted. For this example, you will sort by ZIP code. The option buttons to the right of the Sort By box allow you to choose either ascending or descending sort order. Select **Ascending**. At this point, you can, if you want, sort by multiple fields. When you are done, click **Finish**.

Once you define the query, Query builds it and retrieves the data. The selected data is displayed in the lower window. The tables are displayed in the boxes at the top of the window, and the criteria are displayed in the center, as shown in Figure 3.12.

Figure 3.12: Query results are displayed in the grid.

CREATING A QUERY WITHOUT THE WIZARD

Now let's delve into creating a query without using a wizard. This method sacrifices speed and simplicity for flexibility and power. The sacrifices of speed and simplicity aren't really that large—as you'll see, it's still very easy to create a query.

If it isn't already running, the first step is to start Query. When you are presented with Query's blank workspace, click the **New Query** button. (Alternatively, you can click **File** on the menu bar and select New, as you did for the last query.)

The Choose Data Source screen that was shown in Figure 3.7 appears again. This time, make sure the option to use the Query Wizard is not selected. Choose your iSeries DSN, and click **OK**. When you do this, Query will work for a moment or two, and then present you with a catalog of tables available to your data source. On the Add Tables dialog box shown in Figure 3.13, you will see the tables that you are authorized to use from the databases specified in the DSN setup.
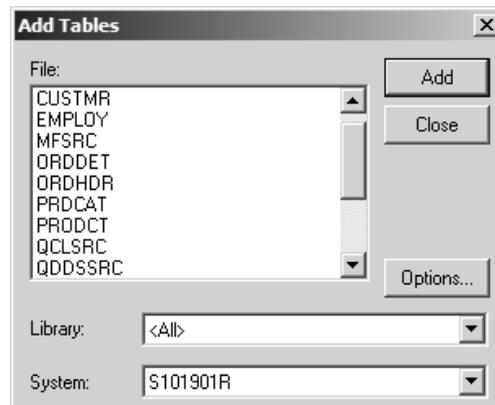
Figure 3.13: Select tables to use in the query from the Add Tables dialog box.

The Options button allows you to specify whether you want to see tables, views, system tables, or synonyms, as shown in Figure 3.14. You usually won't need to make changes to these options, so click **Cancel** to return to Add Tables. Here, you can simply select the tables you want to use for the query and click the Add button. You add tables to the query one at a time. When the tables are added, they appear in the table pane of the query's workspace. When you have the tables you need, you click the Close button.
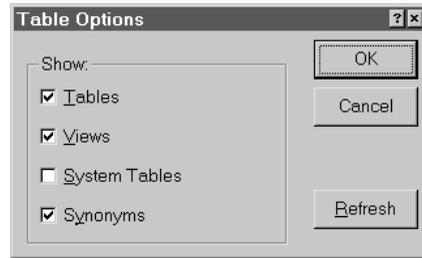
Figure 3.14: Select options using the Table Options dialog box.

For this example, select just one table—the customer (**Custmr**) table. Click **Add**, followed by **Close**.

The remaining screen is the query's workspace. The top section is called the *table pane*. From here, you see representations of the data tables used in your query. The bottom section of the screen is called the *data pane*. This is where the data returned by the query is displayed.

To complete the creation of the query, you add some fields to it. To add a field to the query, select the field's name in the table representation and drag it onto the data pane. The order that you add the fields will determine the layout of the columns in the query. To add all of the fields in the table to your query at once, select the asterisk in the table and drag it to the data pane.

After a delay, Query returns the selected rows and columns from the table, as shown in Figure 3.15. If you don't see the data at this point it may be because the query didn't automatically execute. If not, click the **Query Now** button on the toolbar (the button with the exclamation point).

Figure 3.15: Microsoft Query displays the results of the query.

SAVING THE QUERY

When you're ready, you'll want to save your work. Click the diskette icon on the toolbar, or click **File** on the menu bar and select **Save**. If this is the first time this query has been saved, you will be prompted to enter a file name. Type **myquery** in the File Name field, as shown in Figure 3.16, and click **OK**. The query is now saved for later retrieval. (If you've updated a query that has already been saved and you want to save it with a different name, select the Save As option from the File menu, instead of Save.)
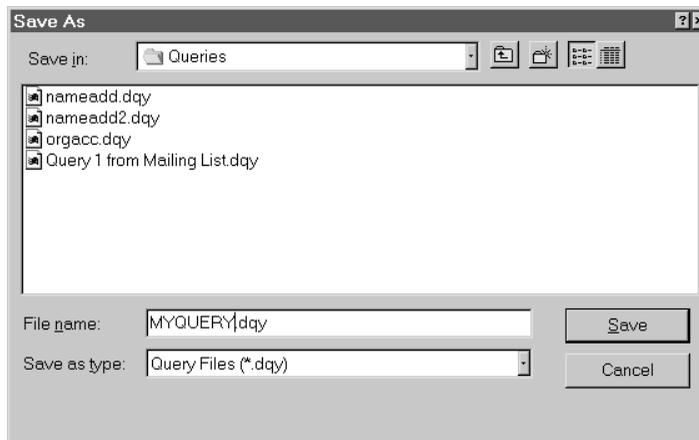
Figure 3.16: You can save the query for later recall.

47

SORTING RECORDS

Now that the basics have been covered, let's move on to the additional capabilities in Query. You've already learned how to select records from a table, but what if you want to sort them? The quickest way to create a sort is to select a column and click the Sort Ascending or Sort Descending button on the toolbar, show in Figure 3.17.

Figure 3.17: The Sort Ascending and Sort Descending buttons determine the sort order.

For example, select a column in the query you just created by clicking the **State** column title in the data pane. The background of the column turns black when it is selected. Then, click the **Sort Ascending** button on the toolbar to sort by that column.

> **Tip:** To determine the name of a button on any toolbar, hold the mouse pointer over the button. A "tool tip" will appear, telling you the button's name.

You can also specify more complex sorts using the Sort option from the Records menu. This allows you to specify exactly which fields should be included in the sort, and which direction they should be sorted. Click **Records** on the menu bar and select **Sort** to display the dialog box shown in Figure 3.18. You can see the sort you've already added in the Sorts in Query drop-down list. Remove that sort by selecting it and clicking **Remove**.
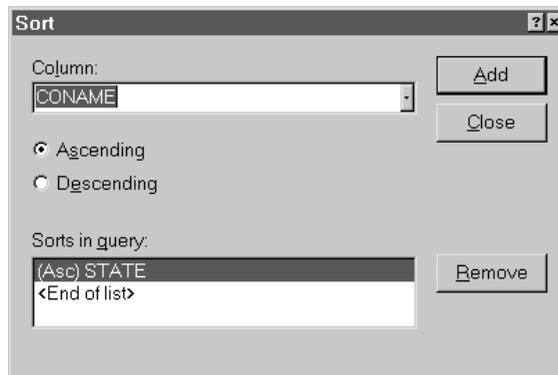
Figure 3.18: The Sort dialog box lets you control a query's sorting.

Add a new column to the sort by selecting it from the Columns drop-down list. Select either **Ascending** or **Descending** to determine the order of the sort, and then click **Add** to add it to your query. You can do this as many times as necessary to get the data sorted in the order you want. To finish, click the **Close** button.

THE SQL STATEMENT GENERATED BY MS QUERY

Through all these changes being made in MS Query, an interpreter is running under the covers to create an SQL statement dynamically. The statement is then passed to the iSeries for execution. This SQL statement requesting data from the iSeries is more-or-less plain English, which you can read in MS Query. Click the **SQL** button on the Query toolbar, and the statement will be displayed, as shown in Figure 3.19. In this example, you can see the basic structure of a data request created for the file: how the fields are selected and how the file is sorted.
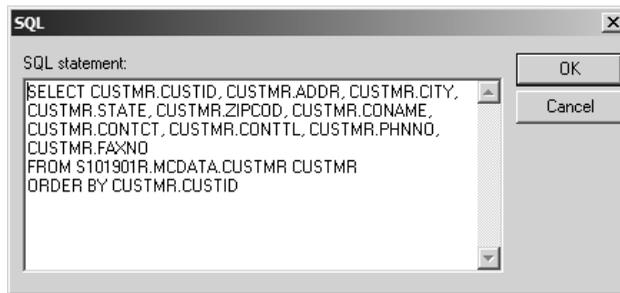
Figure 3.19: This is the SQL statement generated by MS Query.

UPDATING DATA

Query is an easy-to-use tool for updating iSeries database data. It allows you to type changes directly in the fields that you want changed, and *the changes take place immediately* on the iSeries. Consider the ramifications of this. If you are an administrator, you'll want to see the security sections in chapters 2, 6, and 8.

To edit data, you must have the proper authority to the object on the iSeries. Users with read-only capabilities will not be able to select the options to update. Users with full rights to an object can edit field data by creating an updateable query (usually a single file), and then selecting Allow Editing from the Records menu, as shown in Figure 3.20.
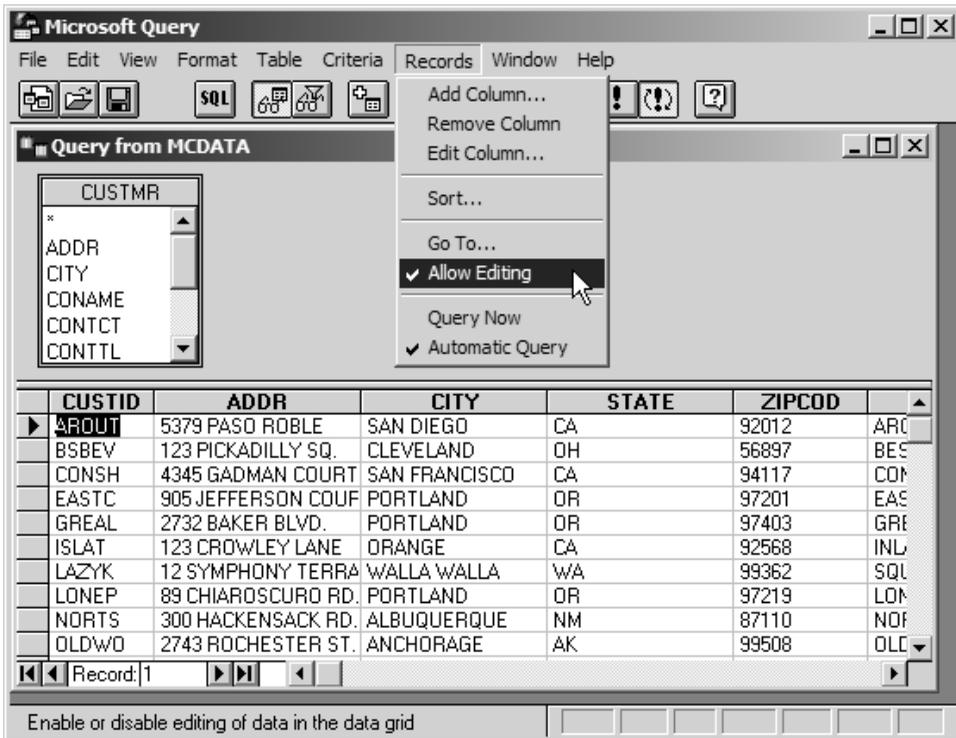
Figure 3.20: You can edit live iSeries data by selecting the Allow Editing option.

> **Warning:** If you are not experienced with all the nuances of relational databases, or if you are not the person at your organization responsible for the data, do not edit data in this manner. Doing so can lead to a broken database and lost business time. There are methods for administrators to detect such edits and determine who is responsible, so be advised.

When editing is allowed, you can change DB2/400 data by simply clicking the field in the query result and typing new data into the cell. As soon as you click an area outside of the cell being edited (i.e., anywhere else on the screen), the changes you have entered are written back to DB2/400. All the rules of DB2/400 are in effect during this process, so all database features such as journals, triggers, referential constraints, and security will be used at the appropriate time.

SPEEDING UP THE DESIGN PROCESS

As you might have noticed, every time you change something in a query, the data automatically gets refreshed, meaning it is read again from the iSeries. This can be a slow process.

To speed up the creation of queries, you can turn off the auto-query option. Do this by clicking the Auto-Query button on the toolbar. This button is a toggle; it changes each time you click it. To turn it off, leave the button in the "up" position. With the auto-query option off, you can click the Query Now button on the toolbar to refresh the data from the iSeries.

> **Note:** Throughout this chapter, whenever you see a reference to refreshing the query, it means pressing the Query Now button on the toolbar.

FILTERING RECORDS

You can also tell Query to retrieve only the records that you want to see. For example, you can create a list of all customers in California by entering a search criterion for the State column, and requesting that only records with CA in that column be returned.

The quickest way to add a criterion is to highlight a value you want by clicking it, and then clicking the Criteria Equals button shown in Figure 3.21. For example, let's find all the customers in California. Locate a cell in the data pane that has the value CA in the state column. Click that cell, and then click the **Criteria Equals** button. This opens up a new pane, called the *criteria pane*, in the window that contains the selection criteria of your query. Refresh the query by clicking the **Query Now** button, and you will see only the customers from California listed, as shown in Figure 3.22.

Figure 3.21: The Criteria Equals button sets criteria based on the cursor location.
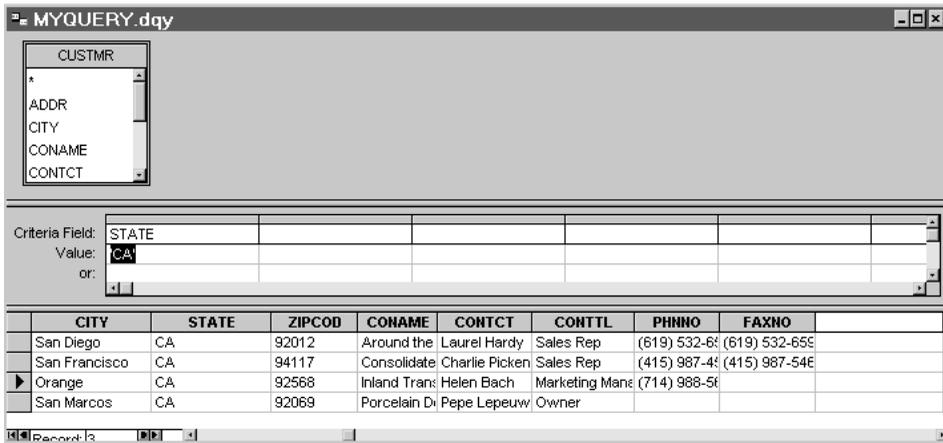
Figure 3.22: The query uses the selection criteria set with the Criteria Equals button.

You can add multiple criteria just as you can add multiple sorts. Another way to add criteria to a query is to select Add Criteria from the Criteria menu. This displays the Add Criteria dialog box shown in Figure 3.23.

To add more criteria, enter the information for the criteria and click the Add button. For example, you will add a criterion to restrict the records to customers in the city of Orange, California. In the field list box, choose **City**. Leave the Operator set at **Equals**, and then click the **Values** button. Clicking this button lists all the current values in the database for the field selected, as shown in Figure 3.24. (Use the Values button carefully. Sometimes, the list can be large and can take a while to be retrieved from the iSeries. We don't often use this button, but it's good to know it's there.)

Figure 3.23: The Add Criteria dialog box helps create data filters.
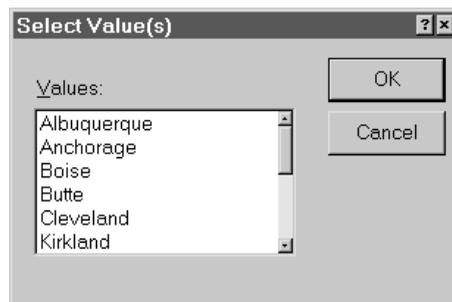
Figure 3.24: You can select from actual values in the database.

Select **Orange** and click **OK** to put it in the Value field. Click the **Add** button in the Add Criteria box, and then click **Close**. Click the **Query Now** button to display only the records from the Customer table with addresses in Orange, California.

That's how easy it is to create criteria to limit your queries to only the records you want to see. One use of this feature would be to select records for a particular timeframe. This allows you to restrict reports with a lot of detail to just the period on which you want to focus. Another use would be to filter out orders for a particular product. There are endless possibilities.

JOINING TABLES

The *relational model* is a method of dividing data into logical, nonrepeating units. This is usually one of the most efficient ways to store data. For example, in a typical relational business system, information about customers is stored in a customer table, while information about products is stored in a product table. Information about orders is stored in two tables, one holding information common to the order, such as the ordering customer and the order date, and the other holding the contents of the order (in other words, the items the customer wants). There may be several items per order and, because it would be inefficient to store the order-header information for each item on the order, separate tables are used. The tables are linked using a special field called a *key field*. If this seems confusing, just follow along with the examples and it will become clearer.

You will create a query that shows all the items ordered from customers in Portland, Oregon. To do that, you need information from three of the tables in your sample database: the Customer table (Custmr), the Order Header table (Ordhdr), and the Order Detail table (Orddet).

First, start a new query by clicking the **New Query** icon on the Query toolbar. Select your iSeries DSN and click **OK**. To add the three tables (Custmr, Ordhdr, and Orddet) to the query, select each one from the list and click **Add** after it. After the last table is added to the query, click **Close**.

Now you can see all three tables in the table pane, as shown in Figure 3.25. To retrieve information from these tables correctly, the relationship between them has to be described to Query. The relationship between the tables is called the *join specification* or the *join criteria*. With some databases, Query does its best to guess the join criteria for you. With others, you have to tell Query how the tables are related by manually adding the join specification.
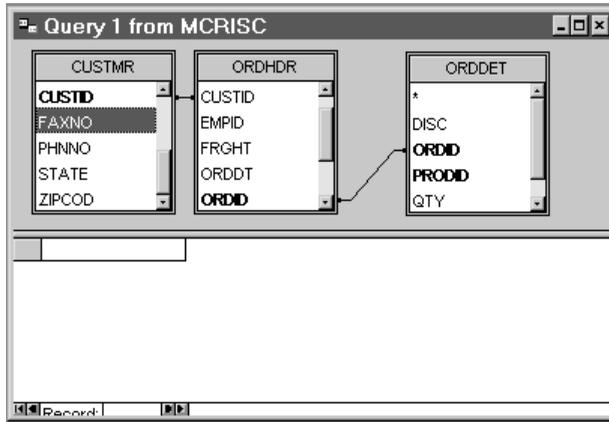
Figure 3.25: Join tables by dragging and dropping key fields.

As you might know, tables in a relational database are joined by common fields. Using Query, the simplest way to specify a join is to drag a common field from one table to its complement in another table. The following sections explain this technique using the Customer and Order Header tables.

> **Note:** Your query will probably detect and present join relationships based on matching field names found in the participating files and iSeries index information.

First, find the Custid field in the Customer table. Then, drag and drop it over the Custid field in the Order Header table. A line appears, joining the two fields. This tells Query that the records in the Order Header table, in effect, belong to the records in the Customer table in a *one-to-many relationship*: one record in the Customer table can have many records in the Order Header table. This makes sense because a customer can have more than one order, but an order would not normally have more than one customer.

There is a similar relationship between the orders in the Order Header table and the items on the order in the Order Detail table. One order can have many items, but one particular order item cannot be on more than one order. The Order Header and Order Detail tables are joined by the Ordid field. Find the Ordid field within the Order Header table, and drag it to the Ordid field in the Order Detail table.

Now there are relationships between all the tables, shown by lines connecting the related fields, as in Figure 3.25.

Let's add some fields to this query to see some data. Rather than looking at all the fields in all three tables, take a look at just a select few. The fields from the customer table are Coname, City, and State. Drag these fields to the data pane to add them to the query. Since you also want to see the order date, drag the Orddt field from the Order Header table to the data pane. From the Order Detail table, take the Qty, Uprice, and Disc fields.

> **Note:** By holding down the Ctrl key, you can select multiple fields from the table and drag them all to the data pane together.

Refresh the query by clicking the **Query Now** button. This returns a lot of records, and many of the values seem to repeat themselves on the screen. Notice that there is one row for every item ordered on every order, as shown in Figure 3.26. That's a lot of information. You can add criteria that help limit the records to just what you want to see: in this example, only customers from Portland, Oregon.

Figure 3.26: The query results reflect a row for every customer order.

First, select **Add Criteria** from the **Criteria** menu. Notice that the fields in the Field drop-down list are now prefixed by the name of the table from which they come. This distinguishes fields with the same name that are in different tables.

Select the **Custmr.State** field and set the Operator text box to **Equals**, set the Value text box to **Or**, and click the **Add** button. This limits the query to customers located in Oregon.

To further limit the query to the city of Portland, select the **Custmr.City** field. Leave the Operator text box set at **Equals**, and type **Portland** in the Value text box. Make sure you use the correct case when entering the city, as shown in Figure 3.27. Now, click **Add**, followed by **Close**.

Figure 3.27: Create new filters using the Add Criteria dialog box.

When the query is refreshed, you see a list of all items ordered by any customers located in Portland, Oregon.

Did you notice that there is no extended price field in the Order Detail table? This is because the extended price needs to be calculated. Fortunately, Query can do that as well. Add a calculated field by selecting **Add Column** or **Insert Column** from the Records menu. (It will say *Add* or *Insert* depending on what is selected at the time.) The example used in Figure 3.28 displays the Add Column dialog box.
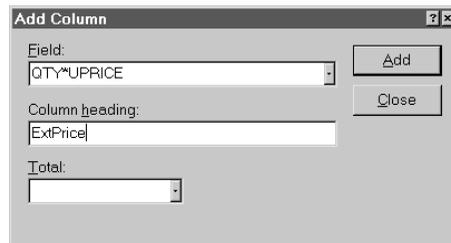
Figure 3.28: The Add Column dialog box lets you add a column heading.

In the Field box, type **qty*uprice**. This is called an *expression*, and expressions are used to create calculated fields. For the column heading, type **ExtPrice**, click the **Add** button, and then click **Close**.

> **Tip:** Once you become familiar with Query, be sure to read its Help file on the subject of expressions. Expressions provide powerful tools for obtaining the information you want.

Refresh the query by clicking the **Query Now** button, and you see the extended price for each item ordered, as shown in Figure 3.29. Expression fields can be used in almost any way that you can use a regular field.

Figure 3.29: The extended price is added to the query.

TOTALING AND GROUPING

So far, you have used Query to produce information in the form of detailed lists. This is useful in manipulating mailing lists and labels, but it is not very useful in analyzing data. To analyze data in a more meaningful way, records must be totaled and grouped based on how you want to see the data. For example, it might be useful to know the total sales by state, or the total sales by customer. Performing these kinds of queries using Query is easy.

To start, create a new query by selecting **New** from the File menu (or by clicking the New Query button). Choose your iSeries data source and click **OK**. From the list of tables, add the **Custmr**, **Ordhdr**, and **Orddet** tables, and then click **Close**. Join the tables as you did for the previous example: the Customer table should be joined with the Order Header table by the Custid field, and the Order Header table should be joined to the Order Detail table with the Ordid field.

To create a grouping query, you have to choose the fields by which you want to group. For this example, drag the **State** field from the Customer table to the data window, to add it to your query. Click the **Query Now** button to run the query. To sort this field in

ascending order, highlight the column by clicking its header, and then click the **Sort Ascending** button on the toolbar.

Now you're ready to create a summary field for the dollar amount of each order. Under the Records menu, select **Insert Column** to bring up the Insert Column dialog box shown in Figure 3.30.

Figure 3.30: Add a summary column using the Insert Column dialog box.

In the Field box, type **qty*uprice**, and in the Column Heading box, type **Sum of Ext.Price**. Then, select **Sum** from the Total drop-down list. Click **Insert**, and then click **Close** to close the Insert Column dialog box.

When you refresh the query, you see the list of states with the total sales for each state, as shown in Figure 3.31. The detailed information is summarized to give a total value.
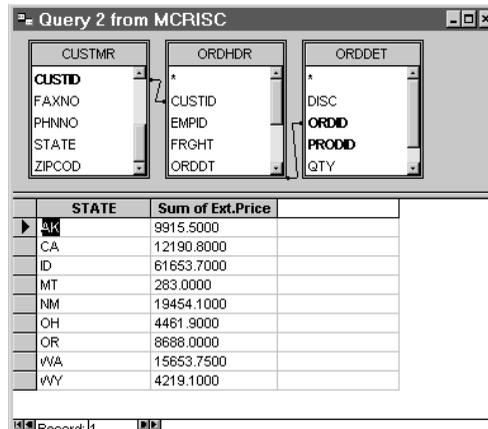
Figure 3.31: A list of states with total sales is displayed.

You can also get averages, counts, and minimum and maximum statistics on summary data. These statistics are useful for creating reports that will help you spot trends in your data. In general, summaries should be to the right side of the grouped data. In fact, that is how Query works. When you create a summary field, the data is grouped by all the fields to the left side of the summary field.

PERFORMANCE CAUTIONS

Microsoft Query can be an excellent tool for exploring data. However, because it is possible to retrieve very large sets of data using Query, you must be careful how you design your requests. You do not want to adversely affect the performance of the iSeries for other users. The following tips will help ensure that you don't get more than you asked for:

- Try to minimize the amount of data that will be retrieved before you refresh the query. This will lower the workload on the iSeries.

- If you do not need to sort data, then do not specify a sort. This will allow the iSeries to process the data in the order that is most efficient for it. Requiring the iSeries to sort a large amount of data can adversely affect the performance, and you should avoid large sorts whenever possible.

- If you do need to sort a large amount of data, consider adding a logical file to the iSeries table that requires sorting. This allows the iSeries to have the data sorted before you request it (meaning the iSeries does not have to perform the sort on the fly). If you do not know how to add a logical file, ask your system administrator if a logical file can be added.

SUMMARY

This chapter explains many ways to use Microsoft Query to retrieve iSeries data. Microsoft Query is a visual query-building tool that offers many powerful options for data retrieval. Because it is used as a "glue" application that shuttles data from the iSeries to Office applications such as Word and Excel, taking the time to understand Query is very important.