chapter

1 Database Basics

IN THIS CHAPTER:

- A Look at Relational Databases
- Introduction to SQL
- To Quote or Not to Quote?
- Project I: Creating an Access Database File
- Setting Up an ODBC Connection
- Project II: Retrieving Data
- SQL Builder
- New Functions
  - SELECT
  - FROM
  - INSERT
  - ORDER BY
  - WHERE
  - DISTINCT
  - INNER JOIN
  - ASC/DESC
  - <CFOUTPUT>
  - <CFSET>
  - <CFQUERY>
- Recap

Let's face it . . . nowadays, just about every major site is using some sort of database on the backend to power their site in one way or another. Even smaller sites have started incorporating databases into their websites due to the amazing flexibility it offers. Not to mention the somewhat short learning curve involved in understanding the basics of what is involved in actually using them. Today, with applications such as ColdFusion, ASP, PHP, and countless others, integrating database functionality into a website is becoming increasingly easier.

But in order to effectively work with databases you need to have some sort of idea of how they actually work. And with a chapter called "Database Basics" I'm sure you are counting on learning exactly that.

There are several types of database applications out on the market, ranging in price from under a hundred dollars to tens of thousands of dollars. The application that most people have available to them—or that doesn't cost an arm and a leg—is Microsoft's Access. Others that you can use if you are going to be doing some heavy-duty database work that needs to span multiple servers are Microsoft SQL server, Oracle, or Sybase. While these cost much more than Access, they offer a lot more in functionality, scalability, and security. They are also designed to handle a lot more simultaneous requests than Microsoft Access.

For the examples in this book we will be using Microsoft Access. You will find the data sources (also known as DSNs which we will be explaining in the next few pages) and all the examples from this book on the *Essential ColdFusion 4.5 for Web Professionals* accompanying website which is located at www.phptr.com/essential/coldfusion45/.

If you do not have a copy of Microsoft Access you can still use the data files to run with ColdFusion and do all the fun stuff like adding, deleting, and editing your data.

◆ A Look at Relational Databases

What is a relational database? In simplest of terms it can be defined as a series of tables that have common fields linking related information. It's a way to relate different information that has a common bond. For instance, a simple online store application may have three tables:

- **Customer table**
  Where all the personal information about the customers is stored.
- **Products table**
  Lists information about the products.
- **Category table**
  Contains the various categories the products will fall under.

The **Customer** table will be a stand-alone table meaning that it will not relate to the other two tables. Only the **Products** and **Category** tables will be related. The relationship between these two tables could look something like this:

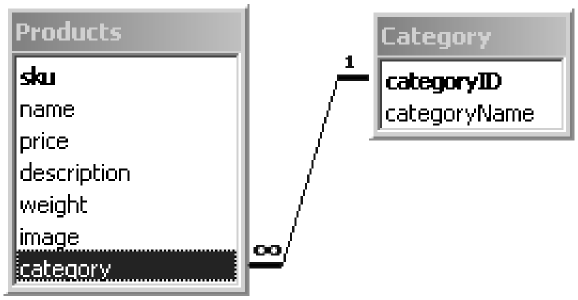


FIGURE 1–1 Relationship of the *category* field to the ***Category*** table

You'll notice the **Products** and **Category** tables have primary keys (shown in bold) associated with them. A primary key is used to uniquely identify a row in a table. For a relational database to function properly, primary keys cannot be duplicated. The relationship shown above demonstrated a "one to one" relationship. In the **Products** table the **sku** field is the primary key, which has the data type of *AutoNumber*. This means that every time you add a new product into the **Products** table the **sku** will *AutoNumber* itself in increments of one.

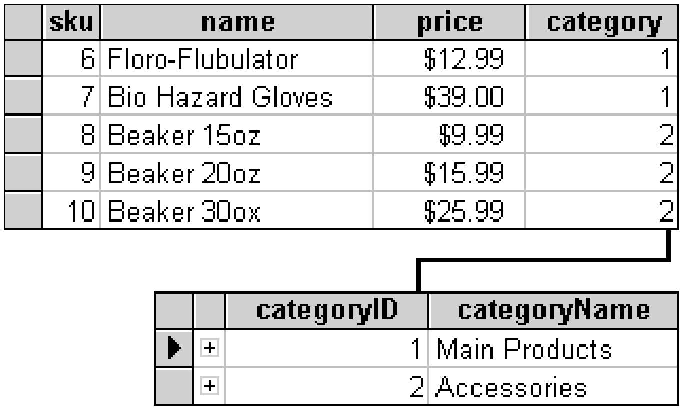
TABLE 1–1 Common Data Types Found in Access

Text	Holds text up to 255 characters.
Memo	Holds text up to 65,535 characters.
Number	Holds numerical characters only. Good to use for mathematical calculations.

TABLE 1–1 Common Data Types Found in Access *(continued)*

Date/Time	Holds date and time values.
Currency	Holds currency (numeric) values.
Autonumber	Incremented value by 1 when a new record is added. Can also be set to randomly autoincrement.
Yes/No	Contains a value of 0 or 1. Used for 'NO/YES', 'OFF/ON', 'TRUE/FALSE'.

The *categoryID* field in the ***Category*** table is the primary key for that table. The *category* field in the **Products** table holds the value of the *categoryID* in the **Category** table and therefore uses the *categoryID* in the **Category** table. This might sound a little confusing at first, so take a look at Figure 1-2 and you'll see how this is working.



	sku	name	price	category
	6	Floro-Flubulator	$12.99	1
	7	Bio Hazard Gloves	$39.00	1
	8	Beaker 15oz	$9.99	2
	9	Beaker 20oz	$15.99	2
	10	Beaker 30ox	$25.99	2

		categoryID	categoryName
▶	+	1	Main Products
	+	2	Accessories

FIGURE 1–2 The values of the *category* field of the **Products** table are related to the values of the *categoryID* field in the **Category** table.

Through the use of SQL you can now interact with the data in your tables.

◆ Introduction to SQL

The Structured Query Language, known as SQL, is the language used to talk to most databases. All the previously mentioned databases understand this language, which allows you to talk to them. SQL is a very powerful language and can be a little complicated the further you dive into it. But for now you need to know only a few frequently used commands.

TABLE 1–2 SQL Basics

Keyword	Definition and Usage
SELECT	Used to select data from the database **Usage** SELECT * FROM table Note: The (*) selects all fields in the specified table.
INSERT	Used to insert new information into the database **Usage** INSERT INTO table (field1,field2) Values (‘Micah’,’Mike’)

DELETE	Used to delete data from a database
	Usage
	DELETE FROM table
	WHERE name = ‘Mike’

UPDATE	Used to update data in the database
	Usage
	UPDATE table
	SET Name1 = ‘Mike’, Name2 = ‘Micah’

TABLE 1–3 SQL Basics—Attributes

Keyword	Definition and Usage

WHERE	Used to select a certain record
	Usage
	SELECT *
	FROM table
	WHERE name = ‘Micah’

ASC	Used to sort results in ascending order
	Usage
	SELECT *
	FROM table
	WHERE name = ‘Micah’ ASC

TABLE 1–3 SQL Basics—Attributes *(continued)*

DESC	Used to sort results in descending order
	Usage
	SELECT *
	FROM table
	WHERE name = ‘Mike’ DESC

ORDER BY	Used to sort results by a field name
	Usage
	SELECT *
	FROM table
	ORDER BY name

DISTINCT	Used to list distinct field names. If multiple names appear in a field column only one is returned
	Usage
	SELECT DISTINCT category
	FROM table

LIKE	Used to search a field for certain text
	Usage
	SELECT *
	FROM table
	WHERE name LIKE ‘%Mike%’
	Note: The ‘%’ is a wildcard character which will return any record that has the string ‘Mike’ in it.

AND	Condition used to ensure two or more fields of information hold true
	Usage
	SELECT *
	FROM table
	WHERE fName = ‘Mike’ AND lName = ‘Fredrick’

OR	Condition used to ensure that any of the two or more fields of information holds true
	Usage
	SELECT *
	FROM table
	WHERE fName = ‘Mike’ OR fName = ‘Micah’

BETWEEN	Condition used to return a result within a certain range
	Usage
	SELECT *
	FROM table
	WHERE birthDate BETWEEN ‘01’ AND ‘10’

TABLE 1–3 SQL Basics—Attributes *(continued)*

IN	Condition used to return any specified results in a field
	Usage
	SELECT *
	FROM table
	WHERE birthDate IN (‘01’,’10’,’23’)

=	Equal to operator, used to find exact matches. Mostly used with WHERE clause
<>	Not equal to operator, used to return results that do not equal. Again used in the WHERE clause
>	Greater than operator
>=	Greater than or equal to operator
<	Less than operator
<=	Less than or equal to operator

◆ To Quote or Not to Quote?

This is by far the largest issue when dealing with SQL. Depending on what datatype you are dealing with (such as a number or character string) you have to interface the database differently. Don’t worry for now if you don’t understand the ColdFusion tags we’re mentioning here, they will be explained a later in this chapter.

Let’s say you have two fields in a table called *userID* and *Password*. *userID* is numeric and *Password* is a character string. Consider the following code.

```
<CFSET userID = 12345>
<CFSET password = “myPass”>
<CFQUERY NAME=”record”>
     UPDATE users
     SET userID = #userID#, password = ‘#password#’
</CFQUERY>
```

Pay close attention to the quotes when setting new variables. *userID* is defined as numeric so the value does not need to be quoted. However, since the *Password* field is a character string (like text or memo in Microsoft Access) it must be enclosed in single quotes. Date formats must also be enclosed in quotes to be inserted or updated in a table.

◆ Project I: Creating an Access Database File

For several of the projects throughout the rest of the book we will be using a database file that you will create in Microsoft Access for the Shelley Biotechnologies website. This is a database containing the three tables we discussed earlier: **Customer**, **Products**, and **Categories**.

While Microsoft Access is great for setting up databases quickly or for smaller database-driven websites, you might want to consider using something like Microsoft SQL server or Oracle to handle the larger-scale websites. These types of databases are designed to handle a large number of simultaneous requests whereas Microsoft Access is not and response times will suffer as a result.

Use the following information to set up your tables:

Customer **Table**

Field Name	Data Type
customerID	AutoNumber—**Primary Key**
firstName	Text
lastName	Text
address	Text
city	Text
state	Text
zip	Text
country	Text
phone	Text
email	Text
mailingList	Text

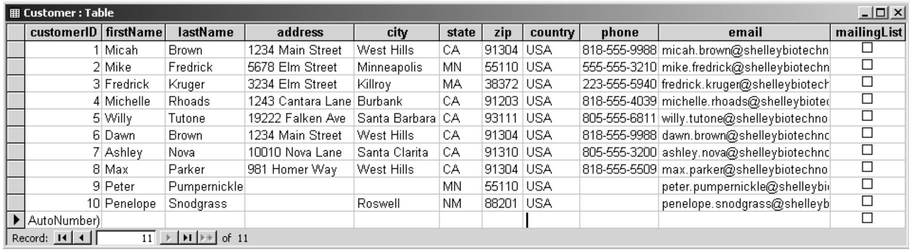
Products **Table**

Field Name	Data Type
sku	AutoNumber—**Primary Key**
name	Text
price	Currency
description	Text
weight	Text
image	Text
category	Number

Category Table	
Field Name	Data Type
categoryID	AutoNumber—**Primary Key**
categoryName	Text

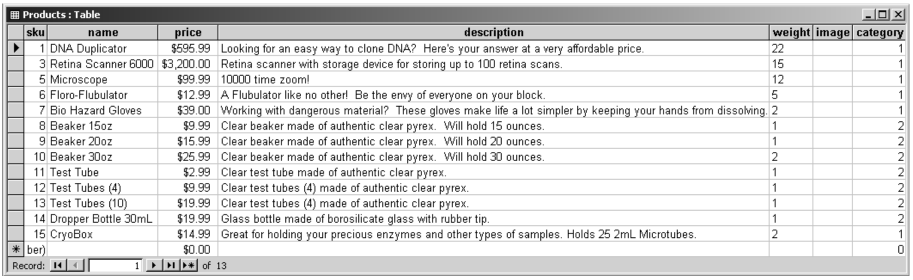
Below is the content we will be using throughout the examples in the following chapters. Some fields were left blank intentionally.

The database file (*shelleyCatalog.mdb*) is also available from our website under chapter 1.



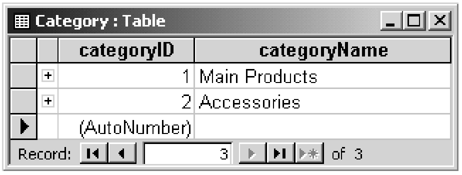
customerID	firstName	lastName	address	city	state	zip	country	phone	email	mailingList
1	Micah	Brown	1234 Main Street	West Hills	CA	91304	USA	818-555-9988	micah.brown@shelleybiotechn	□
2	Mike	Fredrick	5678 Elm Street	Minneapolis	MN	55110	USA	555-555-3210	mike.fredrick@shelleybiotechn	□
3	Fredrick	Kruger	3234 Elm Street	Killroy	MA	38372	USA	223-555-5940	fredrick.kruger@shelleybiotech	□
4	Michelle	Rhoads	1243 Cantara Lane	Burbank	CA	91203	USA	818-555-4039	michelle.rhoads@shelleybiotec	□
5	Willy	Tutone	19222 Falken Ave	Santa Barbara	CA	93111	USA	805-555-6811	willy.tutone@shelleybiotechno	□
6	Dawn	Brown	1234 Main Street	West Hills	CA	91304	USA	818-555-9988	dawn.brown@shelleybiotechnc	□
7	Ashley	Nova	10010 Nova Lane	Santa Clarita	CA	91310	USA	805-555-3200	ashley.nova@shelleybiotechnc	□
8	Max	Parker	981 Homer Way	West Hills	CA	91304	USA	818-555-5509	max.parker@shelleybiotechno	□
9	Peter	Pumpernickle			MN	55110	USA		peter.pumpernickle@shelleybi	□
10	Penelope	Snodgrass		Roswell	NM	88201	USA		penelope.snodgrass@shelleyb	□
AutoNumber)										□

FIGURE 1–3 *Customer* table.



sku	name	price	description	weight	image	category
1	DNA Duplicator	$595.99	Looking for an easy way to clone DNA? Here's your answer at a very affordable price.	22		1
3	Retina Scanner 6000	$3,200.00	Retina scanner with storage device for storing up to 100 retina scans.	15		1
5	Microscope	$99.99	10000 time zoom!	12		1
6	Floro-Flubulator	$12.99	A Flubulator like no other! Be the envy of everyone on your block.	5		1
7	Bio Hazard Gloves	$39.00	Working with dangerous material? These gloves make life a lot simpler by keeping your hands from dissolving.	2		1
8	Beaker 15oz	$9.99	Clear beaker made of authentic clear pyrex. Will hold 15 ounces.	1		2
9	Beaker 20oz	$15.99	Clear beaker made of authentic clear pyrex. Will hold 20 ounces.	1		2
10	Beaker 30oz	$25.99	Clear beaker made of authentic clear pyrex. Will hold 30 ounces.	2		2
11	Test Tube	$2.99	Clear test tube made of authentic clear pyrex.	1		2
12	Test Tubes (4)	$9.99	Clear test tubes (4) made of authentic clear pyrex.	1		2
13	Test Tubes (10)	$19.99	Clear test tubes (4) made of authentic clear pyrex.	1		2
14	Dropper Bottle 30mL	$19.99	Glass bottle made of borosilicate glass with rubber tip.	1		2
15	CryoBox	$14.99	Great for holding your precious enzymes and other types of samples. Holds 25 2mL Microtubes.	2		1
ber)		$0.00				0

FIGURE 1–4 *Products* table.



categoryID	categoryName
1	Main Products
2	Accessories
(AutoNumber)	

FIGURE 1–5 *Category* table.

Obviously this was just a brief overview of what can be done with databases. We suggest you pick up a book specifically on databases if you plan on doing more sophisticated database designs.

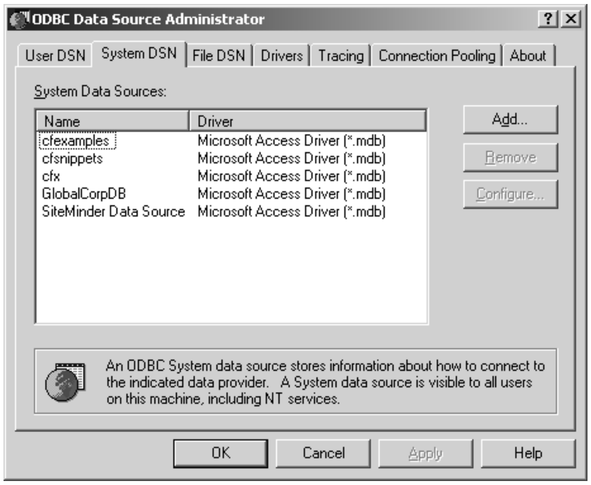
◆ Setting Up an ODBC Connection

In order for your applications, which you will be building soon, to know where the database file is and how to talk to it, you will use open database connectivity also known as ODBC. ODBC usually comes with several database drivers already installed such as Access, SQL Server, dBase, Excel, FoxPro, Oracle, and text. Each of these drivers knows the characteristics of their respective data type and communicates your SQL commands to it. This way you don't have to know exactly how to talk to each individual data type. This is the beauty of SQL!

Once you have a data file saved to an appropriate spot on your machine or on a shared drive on your network, you need to set up the ODBC connection. There are a couple of ways in which to do this.

Windows Control Panel

First open the **control panel** from your **settings** menu and choose the **ODBC** control panel. There are several tabs you can choose from but we will only be paying attention to the tab **System DSN** (Data Source Name).



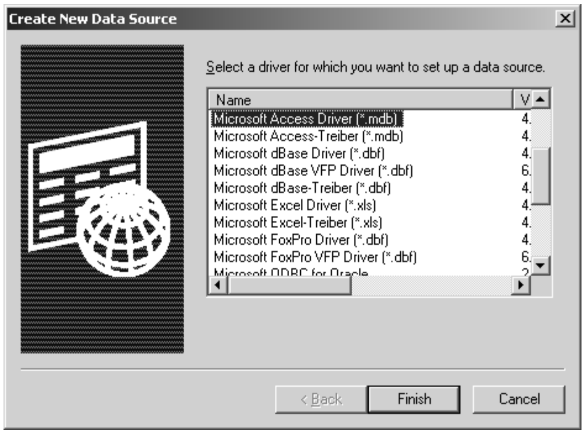
System Data Sources:

Name	Driver
cfexamples	Microsoft Access Driver (*.mdb)
cfsnippets	Microsoft Access Driver (*.mdb)
cfx	Microsoft Access Driver (*.mdb)
GlobalCorpDB	Microsoft Access Driver (*.mdb)
SiteMinder Data Source	Microsoft Access Driver (*.mdb)

An ODBC System data source stores information about how to connect to the indicated data provider. A System data source is visible to all users on this machine, including NT services.

FIGURE 1–6 ODBC Control Panel—System DSN.

Now you need to locate the data file and establish an ODBC connection by clicking the **Add** button, which will bring you to a list of several types of database drivers to choose from. Select the **Microsoft Access Driver**. Once you have done this click **Finish.**



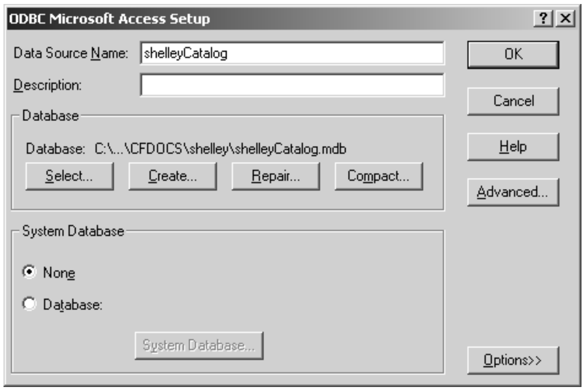
Create New Data Source

Select a driver for which you want to set up a data source.

Name	V
Microsoft Access Driver (*.mdb)	4.
Microsoft Access-Treiber (*.mdb)	4.
Microsoft dBase Driver (*.dbf)	4.
Microsoft dBase VFP Driver (*.dbf)	6.
Microsoft dBase-Treiber (*.dbf)	4.
Microsoft Excel Driver (*.xls)	4.
Microsoft Excel-Treiber (*.xls)	4.
Microsoft FoxPro Driver (*.dbf)	4.
Microsoft FoxPro VFP Driver (*.dbf)	6.
Microsoft ODBC for Oracle	2.

FIGURE 1–7 ODBC control panel—select the Microsoft Access driver.

Now you will tell the system the location of the Access database by clicking the **Select** button and locating the file. Next give the file a name. For our example we will name it shelleyCatalog. Don't use spaces for the name. You can use underscores (such as shelley_catalog) for the sake of clarity if needed. Now hit O**K**.



ODBC Microsoft Access Setup

Data Source Name: shelleyCatalog

Description:

Database

Database: C:\...\CFDOCS\shelley\shelleyCatalog.mdb

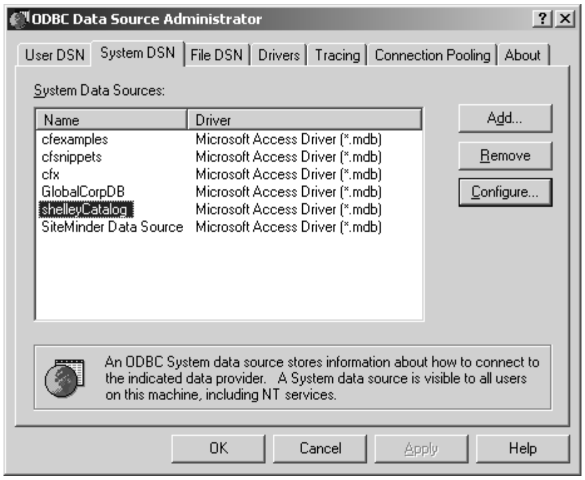
System Database

● None

○ Database:

FIGURE 1–8 ODBC control panel—naming the ODBC connection and defining the location of the data source.

Now you have successfully set up your ODBC connection to the *shelleyCatalog.mdb* file which you will start working with later in this book.



Name	Driver
cfexamples	Microsoft Access Driver (*.mdb)
cfsnippets	Microsoft Access Driver (*.mdb)
cfx	Microsoft Access Driver (*.mdb)
GlobalCorpDB	Microsoft Access Driver (*.mdb)
shelleyCatalog	Microsoft Access Driver (*.mdb)
SiteMinder Data Source	Microsoft Access Driver (*.mdb)

An ODBC System data source stores information about how to connect to the indicated data provider. A System data source is visible to all users on this machine, including NT services.

FIGURE 1–9 ODBC control panel—ODBC connection is now set up.

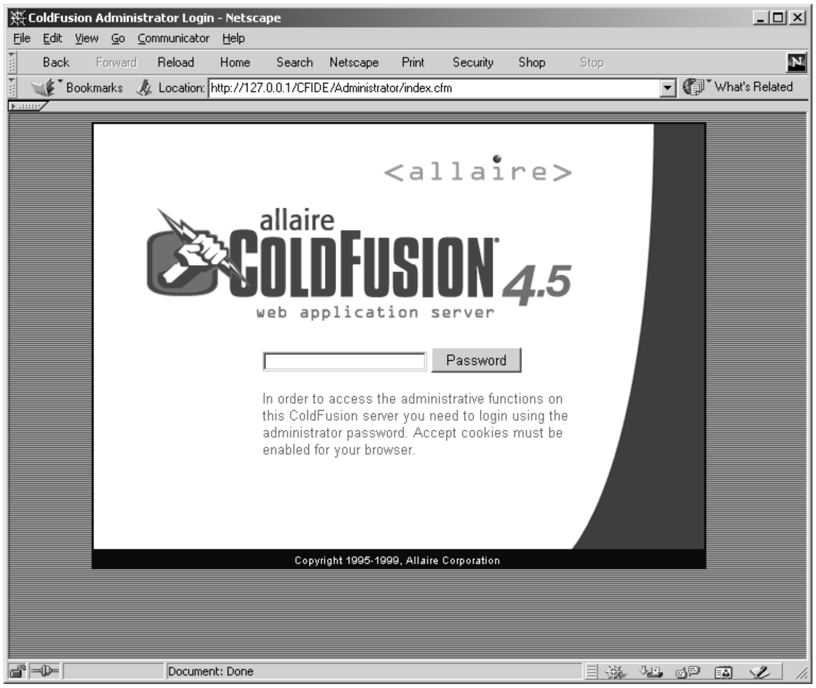
ColdFusion Administrator

The better and far easier way to set up an ODBC connection is through the ColdFusion server's built-in administration screen. Again, if you are going through an ISP that is hosting your website, chances are you cannot use this. The ISP will have to set it up for you.

The ColdFusion server has a built-in administration interface that allows you to control many features of your server either locally or remotely. The server's URL is:

http://127.0.0.1/CFIDE/Administrator/index.cfm

You will be asked to enter a password to log into it. This is the password you entered when you originally set up the ColdFusion server.

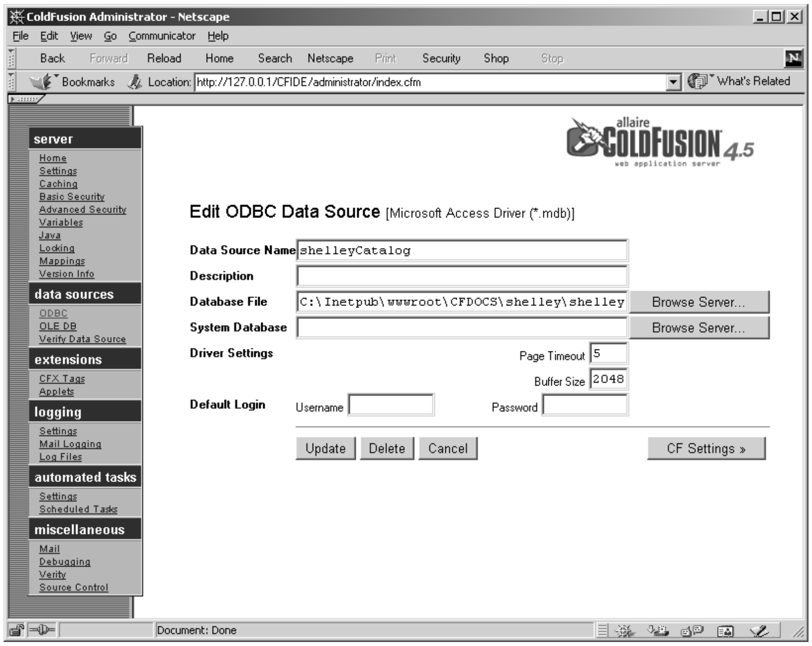

ColdFusion Administrator Login - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Location: http://127.0.0.1/CFIDE/Administrator/index.cfm
What's Related
<allaire>
allaire COLDFUSION 4.5
web application server
Password
Copyright 1995-1999, Allaire Corporation
Document: Done

In order to access the administrative functions on this ColdFusion server you need to login using the administrator password. Accept cookies must be enabled for your browser.

FIGURE 1–10 ColdFusion server administrator.

Under the **Data Sources** heading select **ODBC**. This will list all the ODBC connections currently running on this machine. It tells you the name of the file and the ODBC driver it is using and verifies the connection. If you have made a change to a file and want to make sure the DSN connection is still intact, click the **Verify** link to have the application check it for you. Alternatively you can select the **Verify All** button at the top to have all of your DSNs verified.

Now you want to set up your ODBC connection (assuming you haven't already done so using the Windows ODBC control method). Once you select ODBC under the **Data Sources** heading you will see the list of your current ODBC connections. Select the **ADD** button with the appropriate database type selected in the drop-down menu. Here is where you will add the database name and the location of the database you will be using.



Edit ODBC Data Source [Microsoft Access Driver (*.mdb)]

Data Source Name: shelleyCatalog

Description:

Database File: C:\Inetpub\wwwroot\CFDOCS\shelley\shelley

System Database:

Driver Settings: Page Timeout 5 Buffer Size 2048

Default Login: Username Password

FIGURE 1–11 ColdFusion's ODBC administration screen.

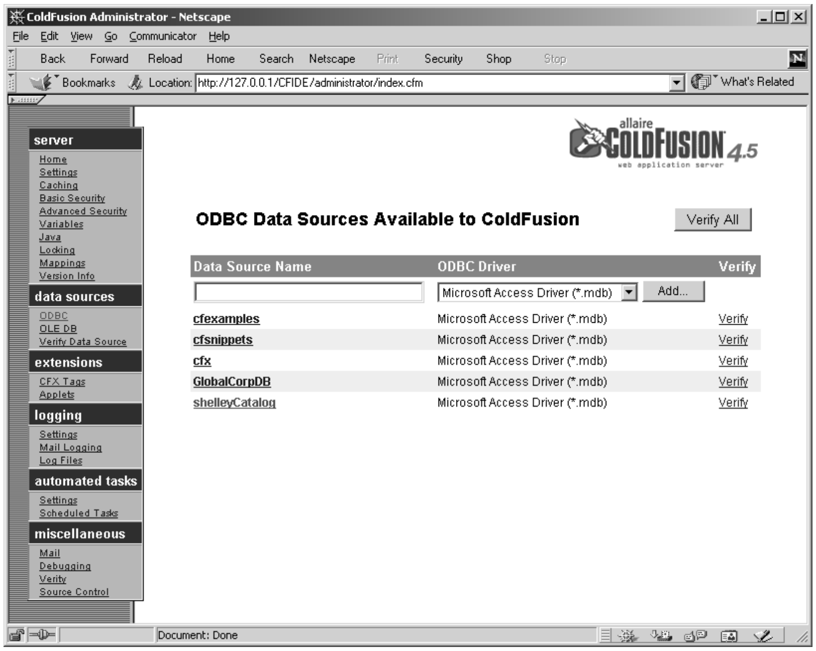
Different database drivers ask for different types of information. All of them ask for three common pieces of information:

- **Database name**—This doesn't necessarily have to be the name of the database file but it is the name you will be using for that connection down the road.
- **Description**—Only used for reference. When you get to the point of running several types or different versions of a database this is a good way to remember which is which.
- **Database File** or **Server**—Using the *Browse Server* button you can locate the file on your hard drive or network. Using SQL server you must type in the location of the machine on your network. If this database file is on the same machine just type in **(local)** and enter the database name in the **Database** field next to **Login Info**.

NOTE
The Browse Server is a Java class application that under Netscape operates fine. However, under Internet Explorer (all versions) it may ask you to connect to the Microsoft site to download the necessary Java class files.

If you have set any password restrictions on a database be sure to enter that in this area. If you are using SQL server connections, select the **CF Settings** to enter this information.

After you have created the ODBC connection, go back to your list of ODBC connections and your new connection will be listed. You are now ready to start learning ColdFusion commands.



ODBC Data Sources Available to ColdFusion

Data Source Name	ODBC Driver	Verify
	Microsoft Access Driver (*.mdb)	
cfexamples	Microsoft Access Driver (*.mdb)	Verify
cfsnippets	Microsoft Access Driver (*.mdb)	Verify
cfx	Microsoft Access Driver (*.mdb)	Verify
GlobalCorpDB	Microsoft Access Driver (*.mdb)	Verify
shelleyCatalog	Microsoft Access Driver (*.mdb)	Verify

FIGURE 1–12 Verification that the ODBC connection was set up correctly.

◆ Project II: Retrieving Data

Now that you have your database set up, it’s time to pull some data from those tables. Here are a few basic elements of ColdFusion you need to know in order to make this happen. These will help you understand how ColdFusion works and give you some hands-on experience right up front.

<CFSET>

The `<CFSET>` tag allows you to set a value to a ColdFusion variable. For example:

```
<CFSET firstName = “Mike”>
<CFSET lastName = “Fredrick”>
```

Now that you have set the variables *firstName* and *lastName*, you can reference them anywhere in the ColdFusion template and they will hold these values. You will learn other ways you can use this as you are introduced to more functions.

Concatenating Variables

You can also very easily concatenate variables by assigning a variable to equal the value of two or more variables:

```
<CFSET firstName=“Mike”>
<CFSET lastName=“Fredrick”>
<CFSET wholeName=”#firstName# #lastName#>
```

Now the variable *`wholeName`* will be: **Mike Fredrick**

<CFOUTPUT>

```
<CFOUTPUT
     QUERY=”name of the query”                    Optional
     GROUP=”query column”                         Optional
     GROUPCASESENSITIVE=”yes | no”                Optional
     STARTROW=”row of query to start from”        Optional
     MAXROWS=”maximum number of rows to return”   Optional
</CFOUTPUT >
```

The `<CFOUTPUT>` tag will allow you to output ColdFusion content requests to your browser, such as the variables that were set above.

TABLE 1–4 <CFOUTPUT> Attributes

Attribute	Description
`QUERY`	The name set in the CFQUERY in which you are querying the database from.
`GROUP`	Defines the column in the query to use when sets of records are grouped.
`GROUPCASESENSITIVE`	Defines whether to group by case sensitivity. Default value is set to YES.
`STARTROW`	Defines the row of the recordset to start the output from.
`MAXROWS`	Defines the number of rows from the recordset you want returned.

#—THE POUND (#) SIGN

Whenever you are going to be displaying a variable to the browser you must enclose the variable within pound characters. This lets the ColdFusion server know at runtime that this is not just ordinary text but it is a variable that must be processed before being displayed to the user's browser.

For example:

```
<CFOUTPUT>Welcome back firstName lastName!</CFOUTPUT>
```

Will output to the screen:

```
Welcome back firstName lastName!
```

If you wrap pound (#) signs around the variables **FirstName** and **LastName**, then ColdFusion will know to process these since they are not just ordinary characters.

For example:

```
<CFOUTPUT> Welcome back #firstName# #lastName#!</CFOUTPUT>
```

Will output to the screen:

```
Welcome back Mike Fredrick!
```

Keeping these basics in mind, we can now grab data from the **Customer** and **Products** table from the Shelley Catalog we built earlier in this chapter.

<CFQUERY>

```
<CFQUERY
     NAME="name of query"                            Required
     DATASOURCE="datasource name"                    Required
     DBTYPE="database type"                          Optional
     DBSERVER="database server"                      Optional
     DBNAME="database name"                          Optional
     USERNAME="username"                             Optional
     PASSWORD="password"                             Optional
     MAXROWS="maximum number of rows"                Optional
     BLOCKFACTOR="block-size"                        Optional
     TIMEOUT="time in milliseconds"                  Optional
     CACHEDAFTER="date"                              Optional
     CACHEDWITHIN="time-span"                        Optional
     PROVIDER="COM provider"                         Optional
     PROVIDERDSN="datasource name"                   Optional
     DEBUG="Yes | No"                                Optional
</CFQUERY>
```

TABLE 1–5 <CFQUERY> Attributes

Attribute	Description
NAME	Defines what you would like to name the query.
DATASOURCE	Defines the name of the data source you set up in your ODBC connection.
DBTYPE	Defines the database driver type. By default, ODBC is set which is perfectly fine for most cases. If using other types of connections like DB2, Informix, OLEDB, Oracle, or Sybase use the following accordingly: - DB2 - Informix73 - OLEDB - Oracle73 - Oracle80 - Sybase11
DBSERVER	Used with native drivers to specify the database server machine.
DBNAME	SQLOLEDB and Sybase drivers specific; specifies the name of the database.
USERNAME	Defines the username set on the data source.
PASSWORD	Defines the password set on the data source.

TABLE 1–5 <CFQUERY> Attributes *(continued)*

Attribute	Description
MAXROWS	Defines the maximum number of rows to return from the recordset.
BLOCKFACTOR	Used to set a maximum number of records an Oracle or ODBC driver will return from a query.
TIMEOUT	Defines the time in milliseconds to timeout from query.
CACHEDAFTER	Returns cached data from the same query that was executed previously. SQL statement and data source must be the same as the original to retrieve cached data.
CACHEDWITHIN	Retrieves the cached data if the query was run within the parameter supplied here. Use the ColdFusion function CreateTimeSpan to create a valid entry.
PROVIDER	Defines the COM provider and is used for OLE-DB only.
PROVIDERDSN	Defines the name of the COM provider and is used for OLE-DB only.
DEBUG	Used for debugging queries.

The <CFQUERY> tag is the method of communication to the ColdFusion application server, which allows you to retrieve data from a data source. <CFQUERY> uses SQL to extract the data you are looking for. For example, let's start the application with a statement to retrieve all the customers.

```
1. <CFQUERY NAME=”getCustomers” DATASOURCE=”shelleyCatalog”>
2. SELECT *
3. FROM Customer
4. </CFQUERY>
```

HOW THIS WORKS

1. NAME defines the name used to reference this query statement. DATASOURCE is the name of the data source you set up in your ODBC connection. In this case we are using *shelleyCatalog*.
2. Using the SQL statement SELECT you will select ALL the fields using the * wildcard character.

3. Using the SQL statement FROM you will SELECT * FROM the Customer table.
4. Close the `<CFQUERY>` statement with `</CFQUERY>`.

Now that you have made your request to the database for the information, you can display it to the browser with `<CFOUTPUT>`.

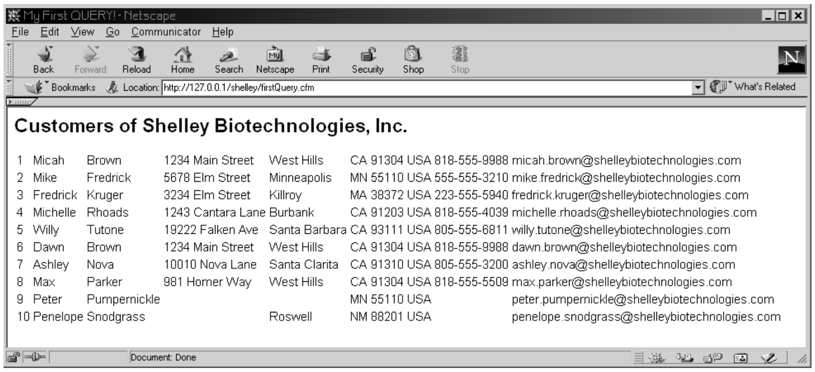
The `<CFOUTPUT>` now has the QUERY attribute attached to it and the name of the query that you specified with `<CFQUERY>` as the NAME attribute. `<CFOUTPUT>` will now loop through all the records pulled from the `<CFQUERY>` and display them. We will throw this into a table to produce a nicely formatted output.

Script 1-1
firstQuery.cfm

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE>My First QUERY!</TITLE>
</HEAD>
<BODY>

<CFQUERY NAME="getCustomers" DATASOURCE="shelleyCatalog">
SELECT *
FROM Customer
</CFQUERY>

<H2>Customers of Shelley Biotechnologies, Inc.</H2>
<TABLE BORDER="0">
<CFOUTPUT QUERY="getCustomers">
<TR>
      <TD>#customerID#</TD>
      <TD>#firstName#</TD>
      <TD>#lastName#</TD>
      <TD>#address#</TD>
      <TD>#city#</TD>
      <TD>#state#</TD>
      <TD>#zip#</TD>
      <TD>#country#</TD>
      <TD>#phone#</TD>
      <TD>#email#</TD>
      </TD>
</TR>
</CFOUTPUT>
</TABLE>
</BODY>
</HTML>
```


My First QUERY! - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Location: http://127.0.0.1/shelley/firstQuery.cfm
What's Related
Document: Done

Customers of Shelley Biotechnologies, Inc.

1	Micah	Brown	1234 Main Street	West Hills	CA	91304	USA	818-555-9988	micah.brown@shelleybiotechnologies.com
2	Mike	Fredrick	5678 Elm Street	Minneapolis	MN	55110	USA	555-555-3210	mike.fredrick@shelleybiotechnologies.com
3	Fredrick	Kruger	3234 Elm Street	Killroy	MA	38372	USA	223-555-5940	fredrick.kruger@shelleybiotechnologies.com
4	Michelle	Rhoads	1243 Cantara Lane	Burbank	CA	91203	USA	818-555-4039	michelle.rhoads@shelleybiotechnologies.com
5	Willy	Tutone	19222 Falken Ave	Santa Barbara	CA	93111	USA	805-555-6811	willy.tutone@shelleybiotechnologies.com
6	Dawn	Brown	1234 Main Street	West Hills	CA	91304	USA	818-555-9988	dawn.brown@shelleybiotechnologies.com
7	Ashley	Nova	10010 Nova Lane	Santa Clarita	CA	91310	USA	805-555-3200	ashley.nova@shelleybiotechnologies.com
8	Max	Parker	981 Homer Way	West Hills	CA	91304	USA	818-555-5509	max.parker@shelleybiotechnologies.com
9	Peter	Pumpernickle			MN	55110	USA		peter.pumpernickle@shelleybiotechnologies.com
10	Penelope	Snodgrass		Roswell	NM	88201	USA		penelope.snodgrass@shelleybiotechnologies.com

FIGURE 1–13 Output of all the fields in the ***Customer*** table.

◆ SQL Builder

A useful built-in feature of ColdFusion Studio is the **SQL Builder** which allows you to view the table layouts and make queries on the fly.

To do this click on **Tools** > **SQL Builder**

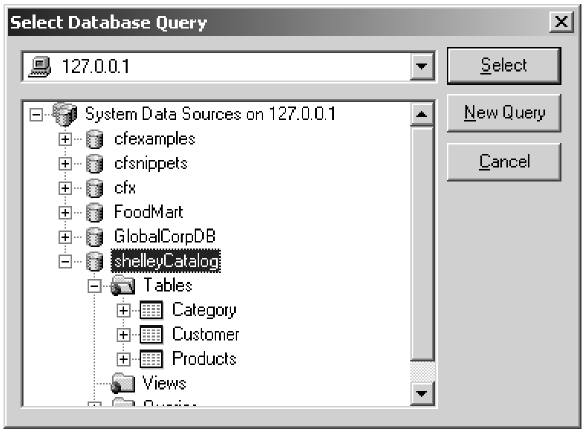
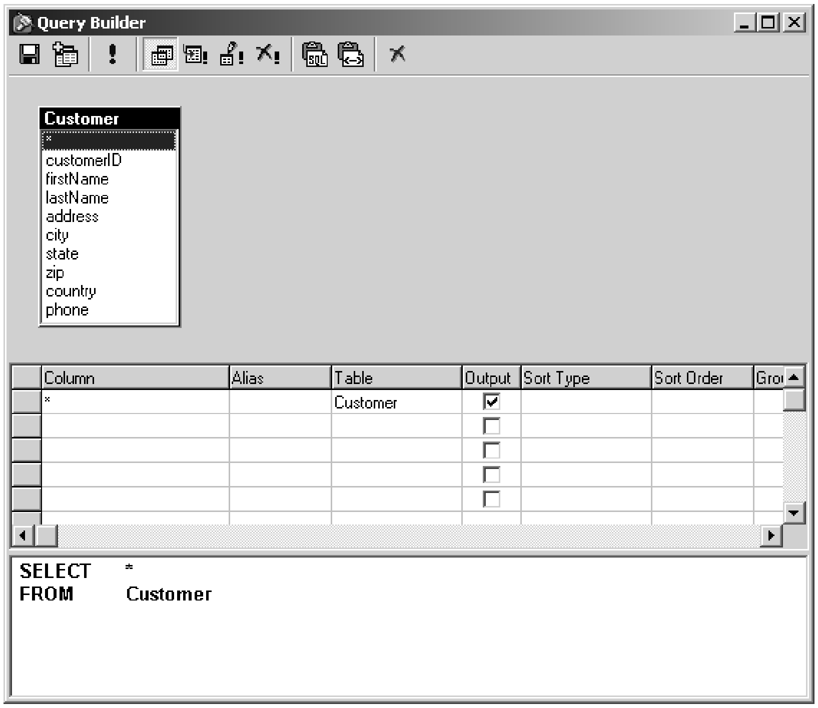


FIGURE 1–14 ColdFusion's built-in SQL builder.

When you see this screen you'll have to have an RDS connection to the source you will be working with. If you are going to be using a data source on your own machine make sure you have

RDS services running. (Check in your ColdFusion Server folder and run the “ColdFusion RDS Services.”) Drill into the **Customer** table and click on **New Query**.

Choose the *shelleyCatalog* data source and select **New Query**. Now you will be able to choose which table from the *shelleyCatalog* you would like to use. If you would like to use another when building a query, right-click and choose **Add Table** where you will be able to select from a list of all the other tables available from the data source you are working in. For this example we will just be using the **Customer** table.



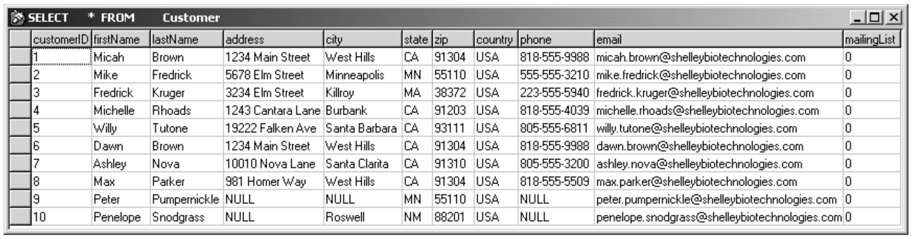
Customer
- *
- customerID
- firstName
- lastName
- address
- city
- state
- zip
- country
- phone

Column	Alias	Table	Output	Sort Type	Sort Order	Grou
*		Customer	☑			
			□			
			□			
			□			
			□			

```
SELECT   *
FROM     Customer
```

FIGURE 1–15 As you choose your fields, the query statement is automatically built for you.

By double clicking on the wildcard * you will choose all fields in the table. As you will notice in the bottom window, the SQL statement is built for you. Once the query statement is built, select the **RUN Query** ! button and the query will execute. This is used to validate the query.



SELECT * FROM Customer

customerID	firstName	lastName	address	city	state	zip	country	phone	email	mailingList
1	Micah	Brown	1234 Main Street	West Hills	CA	91304	USA	818-555-9988	micah.brown@shelleybiotechnologies.com	0
2	Mike	Fredrick	5678 Elm Street	Minneapolis	MN	55110	USA	555-555-3210	mike.fredrick@shelleybiotechnologies.com	0
3	Fredrick	Kruger	3234 Elm Street	Killroy	MA	38372	USA	223-555-5940	fredrick.kruger@shelleybiotechnologies.com	0
4	Michelle	Rhoads	1243 Cantara Lane	Burbank	CA	91203	USA	818-555-4039	michelle.rhoads@shelleybiotechnologies.com	0
5	Willy	Tutone	19222 Falken Ave	Santa Barbara	CA	93111	USA	805-555-6811	willy.tutone@shelleybiotechnologies.com	0
6	Dawn	Brown	1234 Main Street	West Hills	CA	91304	USA	818-555-9988	dawn.brown@shelleybiotechnologies.com	0
7	Ashley	Nova	10010 Nova Lane	Santa Clarita	CA	91310	USA	805-555-3200	ashley.nova@shelleybiotechnologies.com	0
8	Max	Parker	981 Homer Way	West Hills	CA	91304	USA	818-555-5509	max.parker@shelleybiotechnologies.com	0
9	Peter	Pumpernickle	NULL	NULL	MN	55110	USA	NULL	peter.pumpernickle@shelleybiotechnologies.com	0
10	Penelope	Snodgrass	NULL	Roswell	NM	88201	USA	NULL	penelope.snodgrass@shelleybiotechnologies.com	0

FIGURE 1–16 The results from running the SQL builder.

This can also be helpful when working with several tables at once. With a little investigation, it can really save you some time!

◆ Recap

A lot has been covered in this first chapter. By now we're sure you can see some of the possibilities ColdFusion offers. It's very simple to pull data from a database and believe it or not, it's not much harder to add or change data in the database as well. We've only introduced you to some of the basics of SQL and more will be covered in the book as we move along. There is a tremendous amount of flexibility that SQL offers that is good to know when you continue to learn more about web and database development. There are plenty of good SQL books on the market, and if you plan on working with databases you should pick up a copy to learn more.