Creating an Online Address Book

Lesson 9b

In this hands-on lesson, your project will be to create a manageable, online address book. You will learn the methods for creating the relevant database tables, as well as the forms and scripts for adding, deleting, and viewing database records.

In this lesson, you will learn how to

- Create relational tables for an online address book
- Create the forms and scripts for adding and deleting records in the address book

Create the forms and scripts for viewing records Planning and Creating the Database Tables

When you think of an address book, the obvious fields come to mind: name, address, telephone number, email address. However, if you look at your own paper based address book, you may note that you have several entries for one person. Maybe that person has three telephone numbers, or two email addresses, and so forth. In your online address book, a set of related tables will help alleviate the redundancy and repetition of information.

Table 1 shows sample table and field names to use for your online address book. In a minute, you'll use actual SQL statements to create the tables, but first you should look at this information and try to see the relationships appear. Ask yourself which of the fields should be primary or unique keys.

Table 1. Address Book Table and Field Names

Table Name	**Field Names**
`master_name`	`id`, `date_added`, `date_modified`, `f_name`, `l_name`
`address`	`id`, `master_id`, `date_added`, `date_modified`, `address`, `city`, `state`, `zipcode`, `type`
`telephone`	`id`, `master_id`, `date_added`, `date_modified`, `tel_number`, `type`
`fax`	`id`, `master_id`, `date_added`, `date_modified`, `fax_number`, `type`
`email`	`id`, `master_id`, `date_added`, `date_modified`, `email`, `type`
`personal_notes`	`id`, `master_id`, `date_added`, `date_modified`, `note`

Notice the use of date-related fields; each table has a `date_added` and `date_modified` field in it. The fields will help maintain your data; you may at some point want to issue a query that removes all records that are older than a certain number of months or years, or that removes all records that haven't been updated within a certain period of time.

As you can see in the following SQL statements, the `master_name` table has two fields besides the ID and date-related fields: `f_name` and `l_name`, for first name and last name. The `id` field is the primary key. No other keys need to be primary or unique, unless you really want to limit your address book to one John Smith, one Mary Jones, and so forth.

NOTE:

The field lengths for the text fields in the following statements are arbitrary; you can make them as long or as short as you want, within the allowable definition of the field type.

The following SQL statement creates the `master_name` table:

```
mysql> create table master_name (
    -> id int not null primary key auto_increment,
    -> date_added datetime,
    -> date_modified datetime,
    -> f_name varchar (75),
    -> l_name varchar (75)
    -> );
Query OK, 0 rows affected (0.01 sec)
```

Next, you'll create the supplementary tables, which will all relate back to the `master_name` table. For instance, the `address` table has the basic primary key `id` field and the `date_added` and `date_modified` fields, plus the field through which the relationship will be madethe `master_id` field.

The `master_id` will be equal to the `id` field in the `master_name` table, matching the person whose address this is. The `master_id` field is not a unique key because it is a perfectly valid assumption that one person may have several address entries. We see this in the `type` field, which is defined as an enumerated list containing three options: `home`, `work`, or `other`. A person may have one or more of all three types, so no other keys are present in this table besides the primary key `id`. Assuming this particular address book contains only United States addresses, we round out the table with `address`, `city`, `state`, and `zipcode` fields.

```
mysql> create table address (
    -> id int not null primary key auto_increment,
    -> master_id int not null,
    -> date_added datetime,
    -> date_modified datetime,
    -> address varchar (255),
    -> city varchar (30),
    -> state char (2),
    -> zipcode varchar (10),
    -> type enum ('home', 'work', 'other')
    -> );
Query OK, 0 rows affected (0.01 sec)
```

The `telephone`, `fax`, and `email` tables are all variations on the same theme:

```
mysql> create table telephone (
    -> id int not null primary key auto_increment,
    -> master_id int not null,
    -> date_added datetime,
    -> date_modified datetime,
    -> tel_number varchar (25),
    -> type enum ('home', 'work', 'other')
    -> );
Query OK, 0 rows affected (0.01 sec)

mysql> create table fax (
    -> id int not null primary key auto_increment,
    -> master_id int not null,
    -> date_added datetime,
    -> date_modified datetime,
    -> fax_number varchar (25),
    -> type enum ('home', 'work', 'other')
    -> );
Query OK, 0 rows affected (0.00 sec)

mysql> create table email (
    -> id int not null primary key auto_increment,
    -> master_id int not null,
    -> date_added datetime,
    -> date_modified datetime,
    -> email varchar (150),
    -> type enum ('home', 'work', 'other')
    -> );
Query OK, 0 rows affected (0.00 sec)
```

The `personal_notes` table also follows the same sort of pattern, except that `master_id` a unique key and allows only one `notes` record per person:

```
mysql> create table personal_notes (
    -> id int not null primary key auto_increment,
    -> master_id int not null unique,
    -> date_added datetime,
    -> date_modified datetime,
    -> note text
    -> );
Query OK, 0 rows affected (0.00 sec)
```

Now that your tables are created, you can work through the forms and scripts for managing and viewing your records.

Creating a Menu

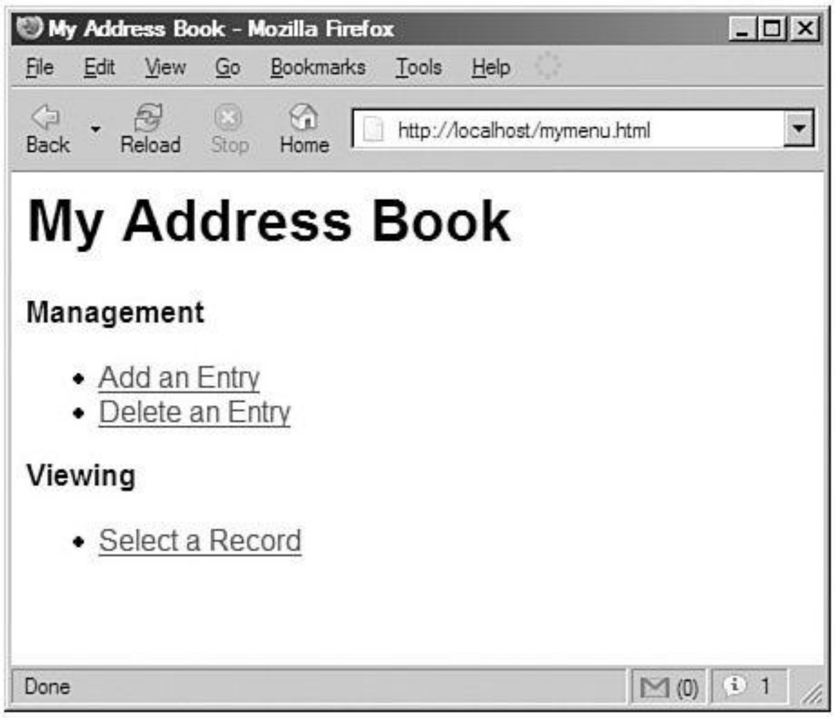
Your online address book will contain several actions, so it makes sense to create a menu for your links. Listing 1 creates a simple menu for all the scripts you will create in this lesson, called `mymenu.html`.

Listing 1. Address Book Menu

```
<html>
<head>
<title>My Address Book</title>
</head>
<body>
<h1>My Address Book</h1>

<P><strong>Management</strong>
<ul>
<li><a href="addentry.php">Add an Entry</a>
<li><a href="delentry.php">Delete an Entry</a>
</ul>

<P><strong>Viewing</strong>
<ul>
<li><a href="selentry.php">Select a Record</a>
</ul>
</body>
</html>
```

Figure 1 shows the output of Listing 1. You'll tackle each of these items in order, starting with "Add an Entry" in the next section.

Figure 1. Address book menu.


My Address Book - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Back Reload Stop Home
http://localhost/mymenu.html
Done
(0)
1

My Address Book

Management

- Add an Entry
- Delete an Entry

Viewing

- Select a Record

Creating the Record Addition Mechanism

Just because you'll potentially be adding information to six different tables doesn't mean your form or script will be monstrous. In fact, your scripts won't look much different from any of the ones you created in previous lessons, and with practice, you will be able to make these verbose scripts much more streamlined and efficient.

In Listing 2, you can see a basic record addition script, called `addentry.php`, that has two parts: what to do if the form should be displayed (lines 2 through 46) and what actions to take if the form is being submitted (lines 48 through 108). Lines 2 through 46 simply place the contents of the HTML form into a string called `$display_block`.

Listing 2. Basic Record Addition Script Called `addentry.php`

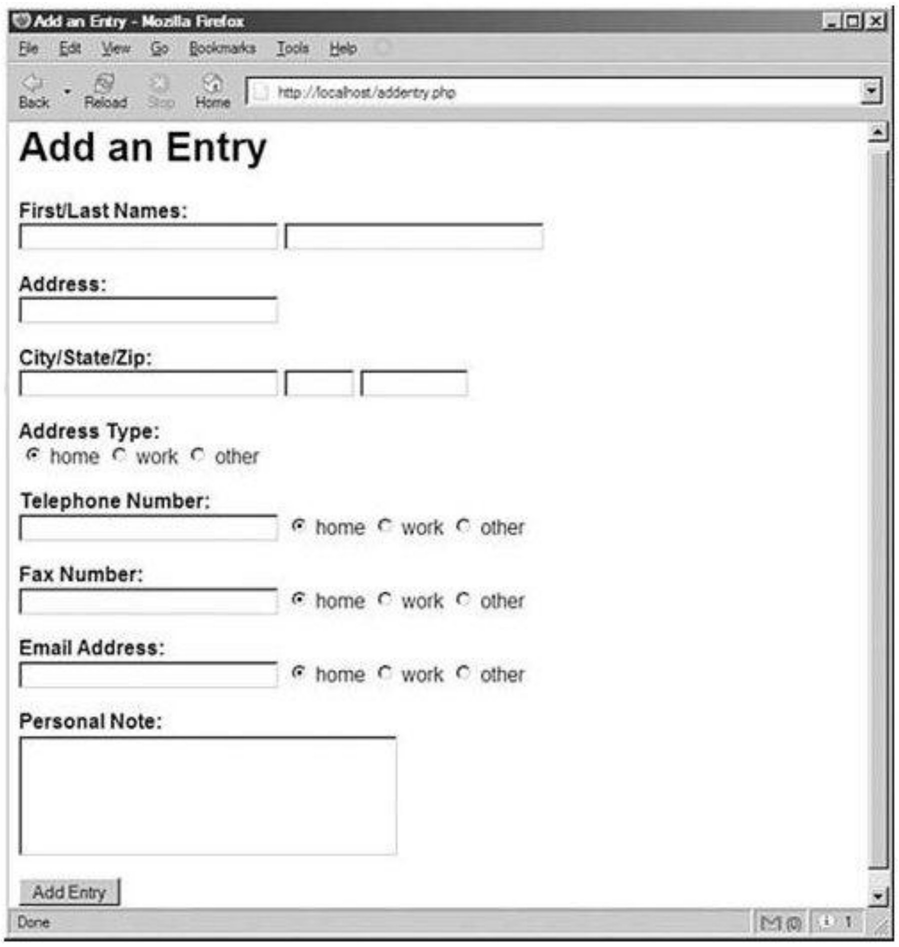
```
  1: <?php
  2: if ($_POST[op] != "add") {
  3:    //haven't seen the form, so show it
  4:    $display_block = "<h1>Add an Entry</h1>
  5:    <form method=\"post\" action=\"$_SERVER[PHP_SELF]\">
  6:    <P><strong>First/Last Names:</strong><br>
  7:    <input type=\"text\" name=\"f_name\" size=30 maxlength=75>
  8:    <input type=\"text\" name=\"l_name\" size=30 maxlength=75>
  9:
 10:    <P><strong>Address:</strong><br>
 11:    <input type=\"text\" name=\"address\" size=30>
 12:
 13:    <P><strong>City/State/Zip:</strong><br>
 14:    <input type=\"text\" name=\"city\" size=30 maxlength=50>
 15:    <input type=\"text\" name=\"state\" size=5 maxlength=2>
 16:    <input type=\"text\" name=\"zipcode\" size=10 maxlength=10>
 17:
 18:    <P><strong>Address Type:</strong><br>
 19:    <input type=\"radio\" name=\"add_type\" value=\"home\"
checked> home
 20:    <input type=\"radio\" name=\"add_type\" value=\"work\"> work
 21:    <input type=\"radio\" name=\"add_type\" value=\"other\">
other
 22:
 23:    <P><strong>Telephone Number:</strong><br>
 24:    <input type=\"text\" name=\"tel_number\" size=30
maxlength=25>
 25:    <input type=\"radio\" name=\"tel_type\" value=\"home\"
checked> home
 26:    <input type=\"radio\" name=\"tel_type\" value=\"work\"> work
 27:    <input type=\"radio\" name=\"tel_type\" value=\"other\">
other
 28:
 29:    <P><strong>Fax Number:</strong><br>
 30:    <input type=\"text\" name=\"fax_number\" size=30
maxlength=25>
 31:    <input type=\"radio\" name=\"fax_type\" value=\"home\"
checked> home
 32:    <input type=\"radio\" name=\"fax_type\" value=\"work\"> work
 33:    <input type=\"radio\" name=\"fax_type\" value=\"other\">
other
 34:
 35:    <P><strong>Email Address:</strong><br>
 36:    <input type=\"text\" name=\"email\" size=30 maxlength=150>
```

```
   <input type=\"radio\" name=\"email_type\" value=\"home\" checked> home
   <input type=\"radio\" name=\"email_type\" value=\"work\"> work
   <input type=\"radio\" name=\"email_type\" value=\"other\"> other

   <P><strong>Personal Note:</strong><br>
   <textarea name=\"note\" cols=35 rows=5 wrap=virtual></textarea>
    <input type=\"hidden\" name=\"op\" value=\"add\">

   <p><input type=\"submit\" name=\"submit\" value=\"Add Entry\"></p>
   </FORM>";

} else if ($_POST[op] == "add") {
   //time to add to tables, so check for required fields
    if (($_POST[f_name] == "") || ($_POST[l_name] == "")) {
       header("Location: addentry.php");
       exit;
    }

   //connect to database
   $conn = mysql_connect("localhost", "joeuser", "somepass")
       or die(mysql_error());
   mysql_select_db("testDB",$conn) or die(mysql_error());

   //add to master_name table
   $add_master = "insert into master_name values ('', now(), now(),
       '$_POST[f_name]', '$_POST[l_name]')";
   mysql_query($add_master) or die(mysql_error());

   //get master_id for use with other tables
   $master_id = mysql_insert_id();

   if (($_POST[address]) || ($_POST[city]) || ($_POST[state]) ||
       ($_POST[zipcode])) {
      //something relevant, so add to address table
       $add_address = "insert into address values ('', $master_id,
            now(), now(), '$_POST[address]', '$_POST[city]',
            '$_POST[state]', '$_POST[zipcode]', '$_POST[add_type]')";
       mysql_query($add_address) or die(mysql_error());
   }

   if ($_POST[tel_number]) {
       //something relevant, so add to telephone table
       $add_tel = "insert into telephone values ('', $master_id,
        now(), now(), '$_POST[tel_number]', '$_POST[tel_type]')";
       mysql_query($add_tel) or die(mysql_error());
   }

   if ($_POST[fax_number]) {
       //something relevant, so add to fax table
       $add_fax = "insert into fax values ('', $master_id, now(),
        now(), '$_POST[fax_number]', '$_POST[fax_type]')";
```

```
 88:        mysql_query($add_fax) or die(mysql_error());
 89:    }
 90:
 91:    if ($_POST[email]) {
 92:        //something relevant, so add to email table
 93:        $add_email = "insert into email values ('', $master_id,
 94:             now(), now(), '$_POST[email]',
'$_POST[email_type]')";
 95:        mysql_query($add_email) or die(mysql_error());
 96:    }
 97:
 98:    if ($_POST[note]) {
 99:        //something relevant, so add to notes table
100:       $add_note = "insert into personal_notes values ('',
$master_id,
101:           now(), now(), '$_POST[note]')";
102:       mysql_query($add_note) or die(mysql_error());
103:    }
104:
105:    $display_block = "<h1>Entry Added</h1>
106:    <P>Your entry has been added. Would you like to
107:     <a href=\"addentry.php\">add another</a>?</p>";
108:  }
109:  ?>
110:  <HTML>
111:  <HEAD>
112:  <TITLE>Add an Entry</TITLE>
113:  </HEAD>
114:  <BODY>
115:  <?php echo $display_block; ?>
116:  </BODY>
117:  </HTML>
```

As we've already noted, this script will perform one of two tasks at any given time: It will either show the record addition form, or it will perform the SQL queries related to adding a new record. The logic that determines the task begins at line 2, with a test for the value of `$_POST[op]`. If the value of `$_POST[op]` is not `"add"`, the user is not coming from the form and therefore needs to see the form. The HTML for the form is placed in a string called `$display_block`, from lines 4 - 55. The script then breaks out of the `if...else` construct and jumps down to line 110, which outputs the HTML and prints the value of `$display_block`, in this case the form. This outcome is shown in Figure 2.

Figure 2. The record addition form.


Add an Entry - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Back Reload Stop Home
http://localhost/addentry.php
Done

Add an Entry

First/Last Names:

Address:

City/State/Zip:

Address Type:
[✓ home] work other

Telephone Number:
[✓ home] work other

Fax Number:
[✓ home] work other

Email Address:
[✓ home] work other

Personal Note:

Add Entry

The `else` condition on Line 48 is invoked if the value of `$_POST[op]` is `"add"`, meaning the user has submitted the form. In this simple example, two fields have been designated as required fields: the first name and last name of the person. So, lines 50 - 53 check for values in `$_POST[f_name]` and `$_POST[l_name]` and redirect the user back to the form if either value is missing.

After making it through the check for required fields, we connect to the database in lines 56 - 59. Next comes the multitude of insertion statements, only one of which is requiredthe insertion of a record into the `master_name` table. This occurs on lines 61 - 63. After the insertion is made, the `id` of this record is extracted using `mysql_insert_id()` on line 66. We use this value, now referred to as `$master_id`, in our remaining SQL queries.

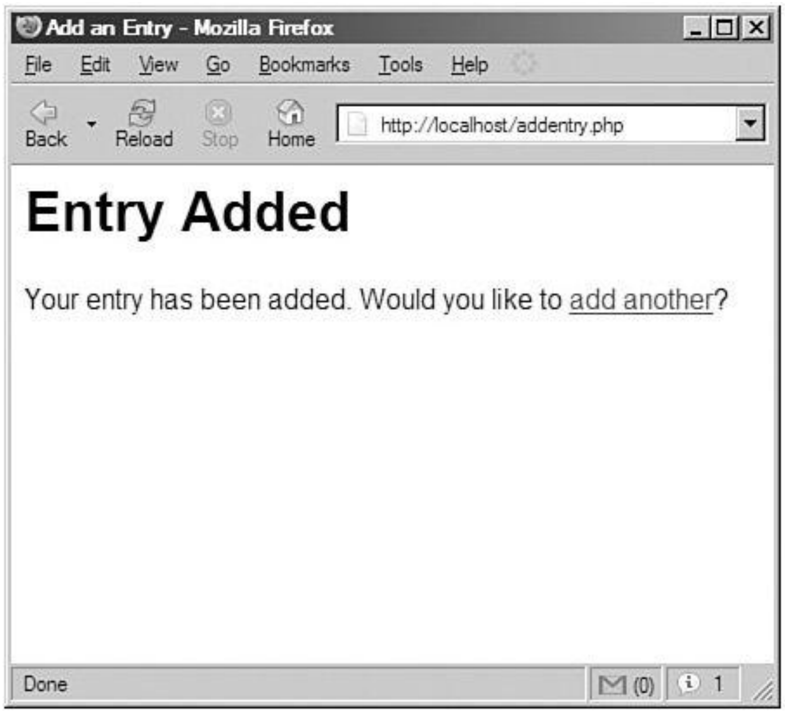
The SQL queries for inserting records into the remaining tables are all conditional, meaning they will occur only if some condition is true. In lines 68 - 69, we see that the condition that must be met is that a value exists for any of the following variables: `$_POST[address]`, `$_POST[city]`, `$_POST[state]`, `$_POST[zipcode]`. Lines 70 - 74 create and issue the query if this condition is met.

The same principle holds true for adding to the `telephone` table (lines 77 - 82), the `fax` table (lines 84 - 89), the `email` table (lines 91 - 96), and the `personal_notes` table (lines 98 - 103). If the conditions are met, records are inserted into those tables.

Once through this set of conditions, the message for the user is placed in the `$display_block` variable, and the script exits this `if...else` construct and prints HTML from lines 110- 117.

An output of the record addition script is shown in Figure 3.

Figure 3. A record has been added.


Add an Entry - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Back Reload Stop Home
http://localhost/addentry.php
Done

Entry Added

Your entry has been added. Would you like to add another?

Add a few records using this form so that you have some values to play with in the following sections. On your own, try to modify this script in such a way that the values entered in the form are printed to the screen after successful record insertion.

Viewing Records

If you verified your work in the preceding section by issuing queries through the MySQL monitor or other interface, you probably became tired of typing `SELECT * FROM...` for every table. In this section, you'll create the two-part script that shows you how to select and view records in your database.

Listing 3 shows the select-and-view script called `selentry.php`, that has two parts: the record selection form (lines 7 through 41) and the code to display the record contents (lines 43 through 155). Because this code is longer than the other code you've seen so far, we'll break it up into smaller chunks for discussion.

Listing 3. Script Called selentry.php for Selecting and Viewing a Record

```
 1: <?php
 2: //connect to database
 3: $conn = mysql_connect("localhost", "joeuser", "somepass")
 4:    or die(mysql_error());
 5: mysql_select_db("testDB",$conn) or die(mysql_error());
 6:
 7: if ($_POST[op] != "view") {
 8:    //haven't seen the selection form, so show it
 9:    $display_block = "<h1>Select an Entry</h1>";
10:
11:   //get parts of records
12:   $get_list = "select id, concat_ws(', ', l_name, f_name) as
display_name
13:       from master_name order by l_name, f_name";
14:   $get_list_res = mysql_query($get_list) or die(mysql_error());
15:
16:   if (mysql_num_rows($get_list_res) < 1) {
17:      //no records
18:      $display_block .= "<p><em>Sorry, no records to
select!</em></p>";
19:
20:   } else {
21:       //has records, so get results and print in a form
22:       $display_block .= "
23:       <form method=\"post\" action=\"$_SERVER[PHP_SELF]\">
24:       <P><strong>Select a Record to View:</strong><br>
25:       <select name=\"sel_id\">
26:       <option value=\"\">-- Select One --</option>";
27:
28:       while ($recs = mysql_fetch_array($get_list_res)) {
29:           $id = $recs['id'];
30:           $display_name = stripslashes($recs['display_name']);
31:
32:           $display_block .= "<option value=\"$id\">
33:                $display_name</option>";
34:       }
35:       $display_block .= "
36:       </select>
37:       <input type=\"hidden\" name=\"op\" value=\"view\">
38:       <p><input type=\"submit\" name=\"submit\"
39:           value=\"View Selected Entry\"></p>
40:       </FORM>";
41:    }
42:
```

As with the `addentry.php` script, the `selentry.php` script will perform one of two tasks at any given time: It either shows the selection form, or it performs all the SQL queries related to viewing the record. No matter which of the two tasks will be performed, the database still comes into play. Given that, we connect to it in lines 35.

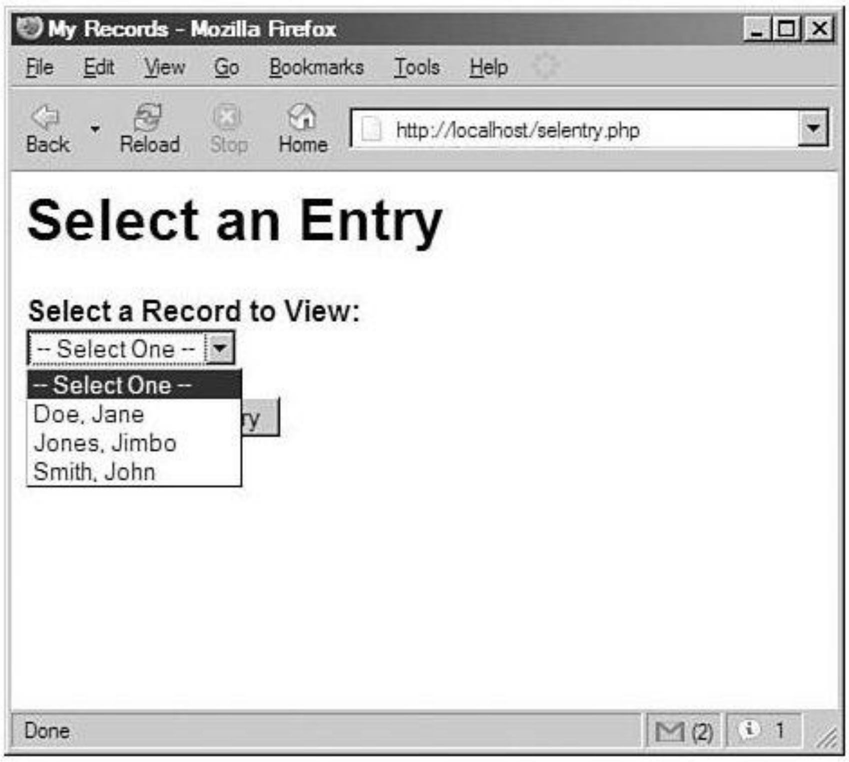
The logic that determines the task begins at line 7, with a test for the value of `$_POST[op]`. If the value of `$_POST[op]` is not `"view"`, the user is not coming from the selection form and therefore needs to see it. A string called `$display_block` is started in line 9, and this string will ultimately hold the HTML that makes up the record selection form.

In lines 12 - 14, we select specific fields from the records in the `master_name` table, to build the selection drop-down options in the form. For this step, you need only the name and ID of the person whose record you want to select. Line 16 tests for results of the query; if the query has no results, you can't build a form. If this were the case, the value of `$display_block` would be filled with an error message and the script would end, printing the resulting HTML to the screen.

However, let's assume you have a few records in the `master_name` table. In this case, you have to extract the information from the query results to be able to build the form. This is done in lines 28 - 33, with form elements written to the `$display_block` string both above and below it.

We've stopped this listing at line 42, but you'll soon see lines 43 through the end of the script. If we were to close up the `if` statement and the PHP block, and print the value of `$display_block` to the screen at this point, you would a form something like that in Figure 4 (with different entries).

Figure 4. The record selection form.



My Records - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Back Reload Stop Home
http://localhost/selentry.php
Done

Select an Entry

Select a Record to View:

-- Select One --

-- Select One --
Doe, Jane
Jones, Jimbo
Smith, John

However, we must finish the `selentry.php` script, so we continue Listing 3 at line 43, which begins the `else` portion of the `if...else` statement:

Listing 3.

```
43:} else if ($_POST[op] == "view") {
44:
45:    //check for required fields
46:     if ($_POST[sel_id] == "") {
47:        header("Location: selentry.php");
48:        exit;
49:    }
50:
51:   //get master_info
52:   $get_master = "select concat_ws(' ', f_name, l_name) as
display_name
53:        from master_name where id = $_POST[sel_id]";
54:   $get_master_res = mysql_query($get_master);
55:   $display_name = stripslashes(mysql_result($get_master_res,
56:        0,'display_name'));
57:   $display_block = "<h1>Showing Record for $display_name</h1>";
58:   //get all addresses
59:   $get_addresses = "select address, city, state, zipcode, type
60:        from address where master_id = $_POST[sel_id]";
61:   $get_addresses_res = mysql_query($get_addresses);
62:
63:   if (mysql_num_rows($get_addresses_res) > 0) {
64:
65:       $display_block .= "<P><strong>Addresses:</strong><br>
66:       <ul>";
67:
68:       while ($add_info = mysql_fetch_array($get_addresses_res)) {
69:           $address = $add_info[address];
70:           $city = $add_info[city];
71:           $state = $add_info[state];
72:           $zipcode = $add_info[zipcode];
73:           $address_type = $add_info[type];
74:
75:           $display_block .= "<li>$address $city $state $zipcode
76:               ($address_type)";
77:       }
78:
79:       $display_block .= "</ul>";
80:   }
81:
```

Line 43 contains the `else` portion of the `if...else` statement, and is invoked if the value of `$_POST[op]` is `"view"`, meaning the user has submitted the form and wants to see a specific record. We first check for a required field, in line 46, in this case the value of `$_POST[sel_id]`. This value matches the ID from the `master_name` table to that of the selection made in the record selection form. If that value does not exist, the user is redirected back to the selection form you can't very well gather information from a set of tables when the primary key isn't present!

Assuming a value was present for `$_POST[sel_id]`, we issue a query in lines 52 - 55 that obtains the name of the user whose record you want to view. This information is placed in the now-familiar `$display_block` string, which will continue to be built as the script continues.

Lines 59 - 80 represent the query against the address table, and the resulting display that is built. If the selected individual has no records in the address table, nothing is added to the `$display_block` string. However, if there are one or more entries, the addresses for this person are added to the `$display_block` string as one or more unordered list elements, as shown in lines 65 - 79.

Lines 82 through 152 of Listing 3 performs the same type of looping and writing to the `$display_block` variable, but the tables are different. For instance, lines 82 through 100 look for information in the `telephone` table and create an appropriate string to be added to `$display_block`, if any information is present. The same structure is repeated in lines 102 through 120 for information from the `fax` table, lines 122 through 140 for information from the `email` table, and lines 142 through 152 for any content present in the `personal_notes` table.

Listing 3.

```
 82:   //get all tel
 83:   $get_tel = "select tel_number, type from telephone where
 84:        master_id = $_POST[sel_id]";
 85:   $get_tel_res = mysql_query($get_tel);
 86:
 87:   if (mysql_num_rows($get_tel_res) > 0) {
 88:
 89:       $display_block .= "<P><strong>Telephone:</strong><br>
 90:       <ul>";
 91:
 92:       while ($tel_info = mysql_fetch_array($get_tel_res)) {
 93:           $tel_number = $tel_info[tel_number];
 94:           $tel_type = $tel_info[type];
 95:
 96:           $display_block .= "<li>$tel_number ($tel_type)";
 97:       }
 98:
 99:       $display_block .= "</ul>";
100:   }
101:
102:   //get all fax
103:   $get_fax = "select fax_number, type from fax where
104:        master_id = $_POST[sel_id]";
105:   $get_fax_res = mysql_query($get_fax);
106:
107:   if (mysql_num_rows($get_fax_res) > 0) {
108:
109:       $display_block .= "<P><strong>Fax:</strong><br>
110:       <ul>";
111:
112:       while ($fax_info = mysql_fetch_array($get_fax_res)) {
113:           $fax_number = $fax_info[fax_number];
114:           $fax_type = $fax_info[type];
115:
116:           $display_block .= "<li>$fax_number ($fax_type)";
117:       }
118:
119:       $display_block .= "</ul>";
120:   }
121:
122:   //get all email
123:   $get_email = "select email, type from email where
124:        master_id = $_POST[sel_id]";
125:   $get_email_res = mysql_query($get_email);
126:
127:   if (mysql_num_rows($get_email_res) > 0) {
128:
129:       $display_block .= "<P><strong>Email:</strong><br>
130:       <ul>";
131:
132:       while ($email_info = mysql_fetch_array($get_email_res)) {
133:           $email = $email_info[email];
134:           $email_type = $email_info[type];
135:
136:           $display_block .= "<li>$email ($email_type)";
137:       }
138:
139:       $display_block .= "</ul>";
140:   }
141:
142:   //get personal note
143:   $get_notes = "select note from personal_notes where
144:        master_id = $_POST[sel_id]";
145:   $get_notes_res = mysql_query($get_notes);
146:
147:   if (mysql_num_rows($get_notes_res) == 1) {
148:       $note =
nl2br(stripslashes(mysql_result($get_notes_res,0,'note')));
149:
150:       $display_block .= "<P><strong>Personal
Notes:</strong><br>$note";
151:   }
152:
```

We still have to do a little housekeeping and finish up the script, as shown in the last portion of Listing 3:

Listing 3.

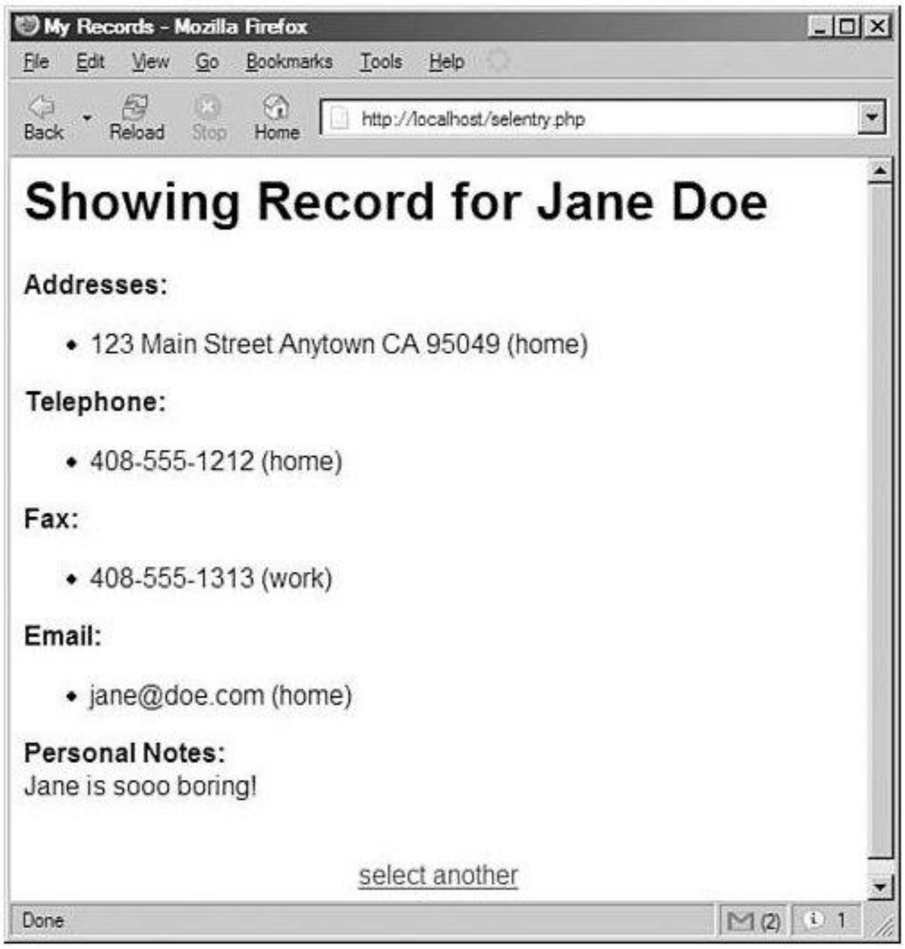
```
153:   $display_block .= "<br><br><P align=center>
154:       <a href=\"$_SERVER[PHP_SELF]\">select another</a></p>";
155: }
156: ?>
157: <HTML>
158: <HEAD>
159: <TITLE>My Records</TITLE>
160: </HEAD>
161: <BODY>
162: <?php echo $display_block; ?>
163: </BODY>
164: </HTML>
```

In lines 153 - 154, we simply print a link back to the selection form before closing up the `if...else` statement in line 155 and the PHP block in the line following. Lines

157 through the end of the script are the generic HTML template that we use to surround the contents of the `$display_block` string.

After selecting a record from the form shown in Figure 4, you will see a result like that shown in Figure 5your data will vary, of course.

Figure 5. An individual's record.


My Records - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Back Reload Stop Home
http://localhost/selentry.php
Done

Showing Record for Jane Doe

Addresses:

- 123 Main Street Anytown CA 95049 (home)

Telephone:

- 408-555-1212 (home)

Fax:

- 408-555-1313 (work)

Email:

- jane@doe.com (home)

Personal Notes:
Jane is sooo boring!

select another

When you try this script for yourself, against your own records, you should see information only for those individuals who have additional data associated with them. For example, if you have an entry for a friend, and all you have is an email address entered in the `email` table, you shouldn't see any text relating to address, telephone, fax, or personal notes no associated records were entered in those tables.

Creating the Record Deletion Mechanism

The record deletion mechanism is virtually identical to the script used to view a record. In fact, you can just take the first 44 lines of Listing 3 and paste them into a new file, called `delentry.php`, and make the following changes:

- In lines 7, 37, and 43, change `"view"` to `"delete"`

- In lines 24 and 39, change `"View"` to `"Delete"`

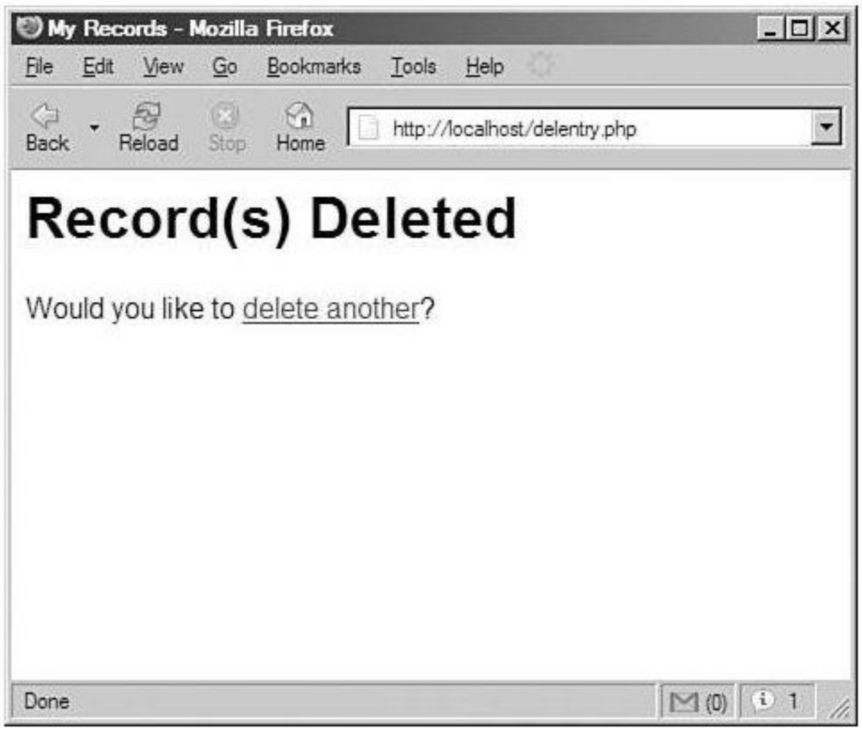
Starting with a new line 45, the remainder of the code for `delentry.php` is shown in Listing 4.

Listing 4. Script Called `delentry.php` for Selecting and Deleting a Record

```
45:   //check for required fields
46:     if ($_POST[sel_id] == "") {
47:       header("Location: delentry.php");
48:       exit;
49:   }
50:
51:   //issue queries
52:   $del_master = "delete from master_name where id =
$_POST[sel_id]";
53:   mysql_query($del_master);
54:
55:   $del_address = "delete from address where id = $_POST[sel_id]";
56:   mysql_query($del_address);
57:
58:   $del_tel = "delete from telephone where id = $_POST[sel_id]";
59:   mysql_query($del_tel);
60:
61:   $del_fax = "delete from fax where id = $_POST[sel_id]";
62:   mysql_query($del_fax);
63:
64:   $del_email = "delete from email where id = $_POST[sel_id]";
65:   mysql_query($del_email);
66:
67:   $del_note = "delete from personal_notes where id =
$_POST[sel_id]";
68:   mysql_query($del_master);
69:
70:   $display_block = "<h1>Record(s) Deleted</h1>
71:   <P>Would you like to
72:   <a href=\"$_SERVER[PHP_SELF]\">delete another</a>?</p>";
73: }
74: ?>
75: <HTML>
76: <HEAD>
77: <TITLE>My Records</TITLE>
78: </HEAD>
79: <BODY>
80: <?php echo $display_block; ?>
81: </BODY>
82: </HTML>
```

Picking up with line 45, the script looks for the required field, `$_POST[sel_id]`, just as it did in the `selentry.php` script. If that required value does not exist, the user is redirected to the selection form. In lines 52 - 68, queries delete all information related to the selected individual, from all tables. Lines 70 - 72 place a nice message in `$display_block`, and the script exits and prints the HTML to the screen. An output of the record deletion script is shown in Figure 6.

__Figure 6. Deleting a record.__


My Records - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Back Reload Stop Home
http://localhost/delentry.php

Record(s) Deleted

Would you like to delete another?

When you go back to the record selection form after deleting a record, you'll note that the individual you deleted is no longer in the selection menuas it should be!

Adding Subentries to a Record

At this point in the lesson, you've learned how to add, remove, and view records. What's missing is adding additional entries to the related tables once you've already entered a master record entries for home versus work telephone number, for example. All you need to do is make a few changes to existing scripts.

In the `selentry.php` script in Listing 3, change lines 153 - 154 to read

```
$display_block .= "<P align=center>
<a href=\"addentry.php?master_id=$_POST[sel_id]\">add info</a> ...
<a href=\"$_SERVER[PHP_SELF]\">select another</a></p>";
```

This change simply adds a link to the `addentry.php` script and also passes it a variable accessible via `$_GET[master_id]`.

Now we need to modify the `addentry.php` script from Listing 2 to account for its dual purposes. Here is a summary of the changes to the original script.

Replace the first 10 lines of the original `addentry.php` script with the following snippet:

```
<?php
if (($_POST[op] != "add") || ($_GET[master_id] != "")) {
    //haven't seen the form, so show it
    $display_block = "
    <h1>Add an Entry</h1>
    <form method=\"post\" action=\"$_SERVER[PHP_SELF]\">";

    if ($_GET[master_id] != "") {
        //connect to database
        $conn = mysql_connect("localhost", "joeuser", "somepass")
                    or die(mysql_error());
        mysql_select_db("testDB",$conn) or die(mysql_error());

        //get first, last names for display/tests validity
        $get_names = "select concat_ws(' ', f_name, l_name) as
                    display_name from master_name where id =
$_GET[master_id]";
        $get_names_res = mysql_query($get_names) or
die(mysql_error());

        if (mysql_num_rows($get_names_res) == 1) {
            $display_name =
mysql_result($get_names_res,0,'display_name');
        }
    }

    if ($display_name != "") {
        $display_block .= "<P>Adding information for
                       <strong>$display_name</strong>:</p>";
    } else {
        $display_block .= " <P><strong>First/Last Names:</strong><br>
        <input type=\"text\" name=\"f_name\" size=30 maxlength=75>
        <input type=\"text\" name=\"l_name\" size=30 maxlength=75>";
    }
    $display_block .= "<P><strong>Address:</strong><br>
```

This snippet simply moves around the form elements, printing the first and last name fields only if they contain a new record. If they contain an addition to a record, the individual's name is extracted from the database for aesthetic purposes as well as for a validity check of the ID.

Next, find this line in the original `addentry.php` script:

```
<input type=\"hidden\" name=\"op\" value=\"add\">
```

Beneath it, add the following:

```
<input type=\"hidden\" name=\"master_id\" value=\"$_GET[master_id]\">
```

This modification ensures the known value of `master_id` is passed along to the next task.

Identify what were lines 49 - 67 of the original script, beginning with the comment `time to add to tables` and ending with obtaining the value of `$master_id`. These lines should be replaced with the following:

```
//time to add to tables, so check for required fields
if ((($_POST[f_name] == "") || ($_POST[l_name] == "")) &&
           ($_POST[master_id] == "")) {
   header("Location: addentry.php");
   exit;
}

//connect to database
$conn = mysql_connect("localhost", "joeuser", "somepass")
           or die(mysql_error());
mysql_select_db("testDB",$conn) or die(mysql_error());

if ($_POST[master_id] == "") {
    //add to master_name table
    $add_master = "insert into master_name values ('', now(),
                  now(), '$_POST[f_name]', '$_POST[l_name]')";
    mysql_query($add_master) or die(mysql_error());
    //get master_id for use with other tables
    $master_id = mysql_insert_id();
} else {
     $master_id = $_POST[master_id];
}
```

These lines modify the check for required fields, allowing the script to continue without values for first and last names, but only if it has a `$_POST[master_id]` value. Then the script connects to the database to perform all the additions we want it to, but it skips the addition to the `master_name` table if a value for `$_POST[master_id]` exists.

Finally, in the section of the script that handles the insertion into the `personal_notes` table, change `INSERT into` to `REPLACE into` to handle an update of the `notes` field.

The new script should look like Listing 5.

Listing 5. New `addentry.php` Script

```
<?php
if (($_POST[op] != "add") || ($_GET[master_id] != "")) {
   //haven't seen the form, so show it
   $display_block = "
   <h1>Add an Entry</h1>
   <form method=\"post\" action=\"$_SERVER[PHP_SELF]\">";

   if ($_GET[master_id] != "") {
      //connect to database
     $conn = mysql_connect("localhost", "joeuser", "somepass")
             or die(mysql_error());
     mysql_select_db("testDB",$conn) or die(mysql_error());

     //get first, last names for display/tests validity
     $get_names = "select concat_ws(' ', f_name, l_name) as
             display_name from master_name where id = $_GET[master_id]";
     $get_names_res = mysql_query($get_names) or die(mysql_error());

     if (mysql_num_rows($get_names_res) == 1) {
```

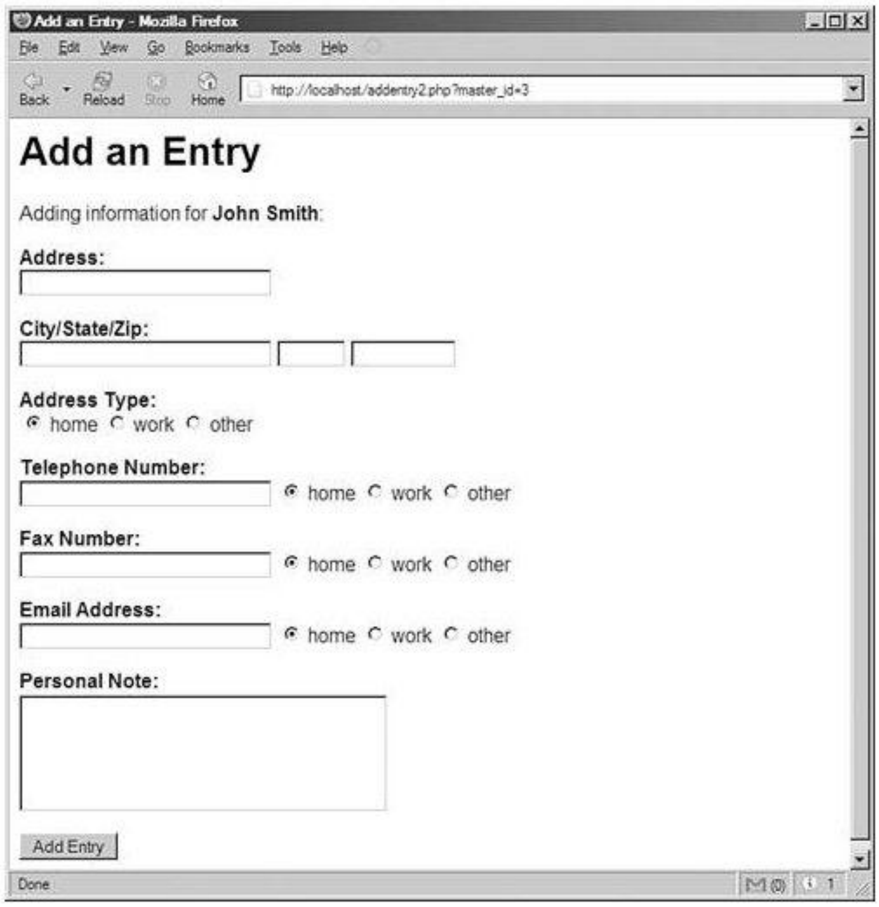
```
              $display_name = mysql_result($get_names_res,0,'display_name');
          }
       }

       if ($display_name != "") {
           $display_block .= "<P>Adding information for
                <strong>$display_name</strong>:</p>";
       } else {
           $display_block .= "
          <P><strong>First/Last Names:</strong><br>
          <input type=\"text\" name=\"f_name\" size=30 maxlength=75>
          <input type=\"text\" name=\"l_name\" size=30 maxlength=75>";
       }
       $display_block .= "<P><strong>Address:</strong><br>
           <input type=\"text\" name=\"address\" size=30>

           <P><strong>City/State/Zip:</strong><br>
           <input type=\"text\" name=\"city\" size=30 maxlength=50>
           <input type=\"text\" name=\"state\" size=5 maxlength=2>
           <input type=\"text\" name=\"zipcode\" size=10 maxlength=10>

           <P><strong>Address Type:</strong><br>
           <input type=\"radio\" name=\"add_type\" value=\"home\" checked> home
           <input type=\"radio\" name=\"add_type\" value=\"work\"> work
           <input type=\"radio\" name=\"add_type\" value=\"other\"> other

           <P><strong>Telephone Number:</strong><br>
           <input type=\"text\" name=\"tel_number\" size=30 maxlength=25>
           <input type=\"radio\" name=\"tel_type\" value=\"home\" checked> home
           <input type=\"radio\" name=\"tel_type\" value=\"work\"> work
           <input type=\"radio\" name=\"tel_type\" value=\"other\"> other

           <P><strong>Fax Number:</strong><br>
           <input type=\"text\" name=\"fax_number\" size=30 maxlength=25>
           <input type=\"radio\" name=\"fax_type\" value=\"home\" checked> home
           <input type=\"radio\" name=\"fax_type\" value=\"work\"> work
           <input type=\"radio\" name=\"fax_type\" value=\"other\"> other

           <P><strong>Email Address:</strong><br>
           <input type=\"text\" name=\"email\" size=30 maxlength=150>
           <input type=\"radio\" name=\"email_type\" value=\"home\" checked> home
           <input type=\"radio\" name=\"email_type\" value=\"work\"> work
           <input type=\"radio\" name=\"email_type\" value=\"other\"> other

```

```
     <P><strong>Personal Note:</strong><br>
     <textarea name=\"note\" cols=35 rows=5 wrap=virtual></textarea>
     <input type=\"hidden\" name=\"op\" value=\"add\">
     <input type=\"hidden\" name=\"master_id\" value=\"$_GET[master_id]\">

     <p><input type=\"submit\" name=\"submit\" value=\"Add Entry\"></p>
     </FORM>";

} else if ($_POST[op] == "add") {
  //time to add to tables, so check for required fields
  if ((($_POST[f_name] == "") || ($_POST[l_name] == "")) &&
  ($_POST[master_id] == "")) {
      header("Location: addentry.php");
      exit;
  }

  //connect to database
 $conn = mysql_connect("localhost", "joeuser", "somepass")
 or die(mysql_error());
 mysql_select_db("testDB",$conn) or die(mysql_error());

  if ($_POST[master_id] == "") {
      //add to master_name table
      $add_master = "insert into master_name values ('', now(),
      now(), '$_POST[f_name]', '$_POST[l_name]')";
      mysql_query($add_master) or die(mysql_error());
      //get master_id for use with other tables
      $master_id = mysql_insert_id();
  } else {
      $master_id = $_POST[master_id];
  }

   if (($_POST[address]) || ($_POST[city]) || ($_POST[state]) ||
        ($_POST[zipcode])) {
       //something relevant, so add to address table
       $add_address = "insert into address values ('', $master_id,
           now(), now(), '$_POST[address]', '$_POST[city]',
           '$_POST[state]', '$_POST[zipcode]', '$_POST[add_type]')";
       mysql_query($add_address) or die(mysql_error());
  }

  if ($_POST[tel_number]) {
      //something relevant, so add to telephone table
      $add_tel = "insert into telephone values ('', $master_id,
          now(), now(), '$_POST[tel_number]', '$_POST[tel_type]')";
      mysql_query($add_tel) or die(mysql_error());
  }

  if ($_POST[fax_number]) {
      //something relevant, so add to fax table
      $add_fax = "insert into fax values ('', $master_id, now(),
          now(), '$_POST[fax_number]', '$_POST[fax_type]')";
      mysql_query($add_fax) or die(mysql_error());
```

```
    }

    if ($_POST[email]) {
        //something relevant, so add to email table
        $add_email = "insert into email values ('', $master_id,
            now(), now(), '$_POST[email]', '$_POST[email_type]')";
        mysql_query($add_email) or die(mysql_error());
    }

    if ($_POST[note]) {
        //something relevant, so add to notes table
        $add_note = "replace into personal_notes values ('', $master_id,
            now(), now(), '$_POST[note]')";
        mysql_query($add_note) or die(mysql_error());
    }

    $display_block = "<h1>Entry Added</h1>
    <P>Your entry has been added. Would you like to
    <a href=\"addentry.php\">add another</a>?</p>";
}
?>
<HTML>
<HEAD>
<TITLE>Add an Entry</TITLE>
</HEAD>
<BODY>
<?php echo $display_block; ?>
</BODY>
</HTML>
```

You can try out this revised script by selecting a record to view and then following the `add info` link. You should see a form like Figure 7.

Figure 7. Adding to a record.


Add an Entry - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Back Reload Stop Home
http://localhost/addentry2.php?master_id=3
Done

Add an Entry

Adding information for **John Smith**:

Address:

City/State/Zip:

Address Type:
[✓ home] work other

Telephone Number:
[✓ home] work other

Fax Number:
[✓ home] work other

Email Address:
[✓ home] work other

Personal Note:

Add Entry

After submitting this form, you can go back through the selection sequence and view the record to verify that your changes have been made.

Workshop

The workshop is designed to help you anticipate possible questions, review what you've learned, and begin learning how to put your knowledge into practice.

Quiz

1. When passing a variable through the query string, which superglobal does it belong in?

2. How many records in the `address`, `email`, `telephone`, and `fax` tables can you have for each individual in your `master_name` table?

Answers

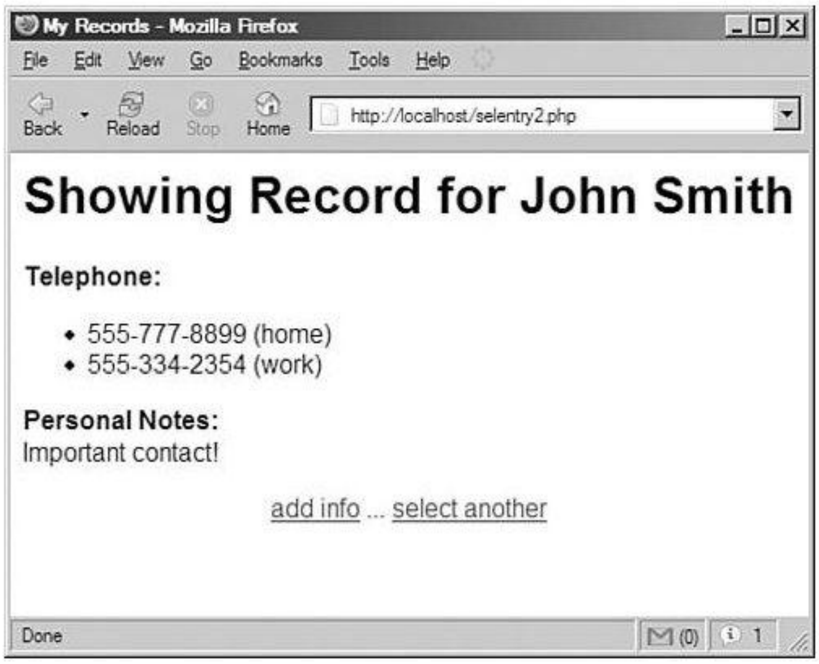
1. The `$_GET` superglobal.

2. As many as you want it's relational!

Activities

1. Go through each of the administration scripts and modify the code so that a link to the menu is printed at the bottom of each screen.
2. Use the second version of the `addentry.php` script to add secondary contact information to records in your database. Figure 8 shows how a record will look after secondary contact information is added to it.

Figure 8. An individual's record with multiple entries in tables.


My Records - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Back Reload Stop Home
http://localhost/selentry2.php
Done

Showing Record for John Smith

Telephone:

- 555-777-8899 (home)
- 555-334-2354 (work)

Personal Notes:
Important contact!

add info ... select another