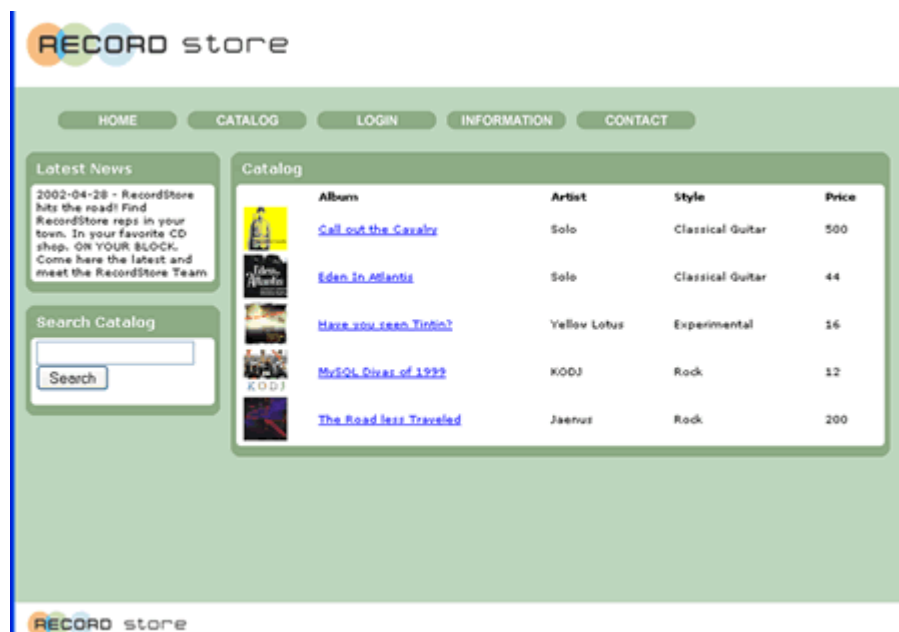


# Creating a simple database search for Record Store



Searching a database is simple and based upon almost everything that you have done in both previous tutorials and previous pages. When you built the **adminlogin.php** page, the behavior that you inserted searched the database for a username and a password. This time, however, you will search for more than one thing at a time.

You are going to create a simple form. You will take your user's search criteria and compare it against several database columns to see if it matches anything in any of the columns. You will use a SQL "OR" operator.

Your user may want to search for all the records by price, by artist, by title, by group, or by style of music. The search you build will let your users find what they need with one search.

## ► PART ONE - Creating the search form in the catalog.php page

First you will create the search form on the catalog.php page. The search form will be right below the latest news box.

First we save it as **catalog\_before\_search.php** in case if we want to start again with our clean catalog.php before adding search functionality.

For this tutorial, you are going to use the **catalog.php** after detaching the template from it. This will simplify the process for you.

1. Open the **catalog.php** page. Click **File, Save As**
2. Type: **catalog\_before\_search.php**
3. Click **File, Close**.
4. Open the **catalog.php** one more time
5. From the menu select **Modify > Templates > Detach** from Templates.
6. Click **File, Save to save your page**.

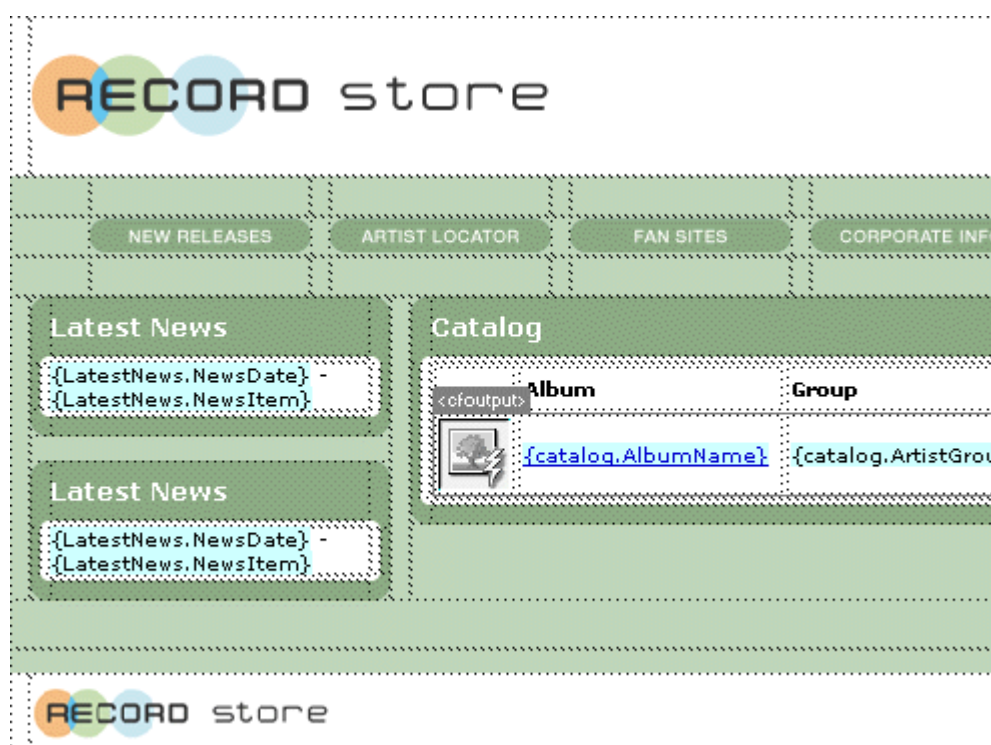
To detach a template in CS3  
Modify, Templates, Detach

## Adding the form to the catalog.php page

You are going to use a form that looks like the latest news area. To use this form, copy the Latest News table and paste it on your page. You will modify it as a search area (described below).

1. Click the **Latest News box** and click the first `<table>` tag on the right side of the tag selector, which is located at the bottom of the Dreamweaver window. **This will select the whole table and its contents.**
2. **Copy** the table and **press the right arrow key** to move the cursor to the right of the existing Latest News table. Then, hold the **Shift** key on the Keyboard and Press **Enter** key (This insert a blank line), and **paste** the copy of the table into the page.

The page should look like this:



3. Replace the "Latest News" text in the copied table with "Search Catalog".
4. Delete the "LatestNews.NewsDate" and "LatestNews.NewsItem" place holders to clear the cell of all contents.
5. From the Insert bar, Click the Form Tab
6. Insert a form in the empty cell, and insert both a text field and a submit button.
7. Rename the text field **searchcatalog**
8. Set the value of the submit button to **Search**.
9. Go to the **Code View** and change the `<Form.....` to `<form name="form1" method="get" action="catalog.php?search=true">`
10. This sets the method to GET, and sets the action of the form to "catalog.php?search=true".
11. Save the page.

## ► PART TWO - Setting up the viewing conditions

In the last section, we added an action to the search form that calls the catalog.php page. The form action goes to the same page as the form itself. I do this to place all the logic for the search and the results in one place. This way, you don't need to maintain multiple pages—one place each for the search form, catalog, and search results. This is better for both the user (for browsing) and the developer (for developing and maintaining).

### Setting up the viewing conditions

1. Put your cursor in the table where the catalog appears and select the table by clicking on the **table** tag farthest to the right in the tag inspector at the bottom of the Dreamweaver window.
2. Click Edit, Copy to copy the entire table.

**3. You are going to paste this in step 17 below.**

4. Press the **left arrow key** to move the cursor before the table.
5. From the PHP tab of the **Insert bar**, click the **IF** icon.
6. Your screen will display the **split mode** when you click the **IF** icon.  
**Switch to the Code View to see only the codes**
7. Click just above the **<?php** line that you have the if statement
8. We want to add a comment, therefore go to the **Top menu, Insert, Comment**
9. Type: **START OF SEARCH**

10. Inside the **IF** tag, add:

```
(isset($_HTTP_GET_VARS["searchcatalog"])) {
```

11. At the end of if statement (above line) right after **?>** press **Enter** key two times
12. Type: **<?php } ?>**

13. **Below** the **<?php } ?>** line, add another comment line as **END OF SEARCH**

```
<!-- START OF SEARCH -->
<?php if (isset($_HTTP_GET_VARS["searchcatalog"])) { ?>
    ←
< ?php } ?>
<!-- END OF SEARCH -->
```

15. Click on **Design View** button
16. Click between the **?php** line – on the blank line
17. Click on the 1<sup>st</sup> blank line, then click **Edit, Paste** (pasting the table)

## ► PART TWO CONTINUE – Replace Catalog with Search

Your duplicated table shows the catalog, except it will only display when a page calls it with "search" parameter attached to the URL.

Also, your search query is called "Search" not "Catalog", so you need to change each reference of "Catalog" to "Search" in your code.

**BE CAREFUL NOT TO GO BEYOND THE <?PHP ?> TAG, BECAUSE THE NEXT VERSION OF THE TABLE IS INDEED FOR THE CATALOG.**

**YOU DON'T WANT TO CHANGE THOSE REFERENCES.**

**Your code will look like this: Save Your File after making the changes**

```
<!-- START OF SEARCH -->
<?php if (isset($_HTTP_GET_VARS["searchcatalog"])) { ?>
  <table width="100%" border="0" cellpadding="2">
    <tr>
      <td> </td>
      <td><strong>Album</strong></td>
      <td><strong>Artist</strong></td>
      <td><strong>Style</strong></td>
      <td><strong>Price</strong></td>
    </tr>
    <?php do { ?>
      <tr>
        <td></td>
        <td><a href="albumdetail.php?AlbumID=<?php echo
          $row_Search['AlbumID']; ?>"><?php echo
          $row_Search['AlbumName']; ?></a></td>
        <td><?php echo $row_Search['ArtistGroupName']; ?></td>
        <td><?php echo $row_Search['ArtistStyle']; ?></td>
        <td><?php echo $row_Search['AlbumPrice']; ?></td>
      </tr>
    <?php } while ($row_Search = mysql_fetch_assoc($Search)); ?>
  </table>
< ?php } ?>
<!-- END OF SEARCH -->
```

## ► PART THREE – Setting up the search query

You will create a query that responds to the search criteria passed from the calling page. In this case, we are not going to use the Dreamweaver query because we need to customize the query. When you search, you can search for a single word or term, or you can search for parts of the word. For instance, you might search for "sea" and get entries containing "sea". More powerful, though, you may search for "sea" and to get "sea", "seasalt", "seaside", "seal" and so on. To do that, you need to use wildcards, which will extend the search for partial words in the database.

In MySQL, the wildcard is the **% character**. The query has to do a few things. It has to accept the value of the 'searchcatalog' text box, which is going to come to the page as part of the URL using the **GET method** of the form.

Once the value is in the query, it has to remove any special characters that are reserved for PHP and it has to fix any quotation marks. Then, the query has to open a connection to the database. Once the query has database access, it needs to compare the value of the searchcatalog text box to all the values of the tables and columns you are going to search. Once it has the query, it has to run the query and then it is done.

## Building the search query

1. Change to the Code view if you are not already there, and scroll to the top of the **catalog.php** page.
2. Click underneath the **\$query catalog** and above the **"?>"** closing tag.

```
30 }
31
32 mysql_select_db($database_connl, $connl);
33 $query_LatestNews = "SELECT * FROM news ORDER BY NewsDate DESC";
34 $LatestNews = mysql_query($query_LatestNews, $connl) or die(mysql_error());
35 $row_LatestNews = mysql_fetch_assoc($LatestNews);
36 $totalRows_LatestNews = mysql_num_rows($LatestNews);
37
38 mysql_select_db($database_connl, $connl);
39 $query_catalog = "SELECT * FROM album, artist WHERE album.ArtistID = art
40 $catalog = mysql_query($query_catalog, $connl) or die(mysql_error());
41 $row_catalog = mysql_fetch_assoc($catalog);
42 $totalRows_catalog = mysql_num_rows($catalog);
43
44 Click Here ←
45
46 ?>
47 <html>
48 <head>
```

3. Press **Enter** key a few times to give yourself some space.
4. Add the block of code to get the value from the 'searchcatalog' text field. Enter the following code.

```
$colname_Search = "1";
    if (isset($_HTTP_GET_VARS['searchcatalog'])) {
        $colname_Search = (get_magic_quotes_gpc()) ?
$_HTTP_GET_VARS['searchcatalog']
        : addslashes($_HTTP_GET_VARS['searchcatalog']);
    }
```

**The first line** creates a new variable and sets the value to one. Always initialize your variables so that there are no surprises later on.

**The second line** starts a conditional statement and checks if there is a URL's GET method parameter called "searchcatalog". If there is not, the application escapes this condition and moves to the next line of code.

If there is a value in the URL's GET method parameter "searchcatalog", it takes that value and it adds slashes to take care of special characters that might be in the string. It also sets it equal to a variable called "\$colname\_Search".

5. Press the Enter key a couple times to give yourself some space, and enter the following text:

```
mysql_select_db($database_conn1, $conn1);
```

This opens the connection to the database that is defined in the connections folder. You are going to use the conn1 connection, which all the other queries in this tutorial use as well.

Press the return key and enter the following text:

```
$query_Search = "SELECT * FROM album, artist WHERE  
(album.AlbumName LIKE '%$colname_Search%'  
OR album.AlbumPrice LIKE '%$colname_Search%'  
OR artist.ArtistGroupName LIKE '%$colname_Search%'  
OR artist.ArtistStyle LIKE '%$colname_Search%')  
AND album.ArtistID = artist.ArtistID";  
$Search = mysql_query($query_Search, $conn1) or die(mysql_error());  
$row_Search = mysql_fetch_assoc($Search);  
$totalRows_Search = mysql_num_rows($Search);
```

This is the query itself in SQL. The formatting is not so important here, developers do format queries to organize ideas and to maintain the code.

The first line is a variable and the results of the query are going to be entered into this variable.

The second line selects all (\*) the columns from the album and artist tables. Best practice is to only select the columns you need to make the query faster and use less memory. Selecting all the columns makes the code more difficult to read though, so select all the columns.

The following lines all place conditions on what to select from the columns.

The third line starts the WHERE clause and then opens a parenthesis. You are going to test all the columns at once and if there is a match in any of them, it will continue. If you set them all to select one at a time, without the parenthesis, the query will stop checking when it finds the first match.

The WHERE clause looks at the AlbumName column from the Album table and compares that with a variable called "\$colname\_Search", which is the contents of the "searchcatalog" text area that was in the GET method of the URL that called the page. The comparison uses the LIKE operator, which means the SQL is looking for things like "\$colname\_Search", not exactly "\$colname\_Search".

The "\$colname\_Search" variable is wrapped in % signs, which indicate that MySQL is supposed to reach out from that word and find things that are larger than

"\$colname\_Search" if "\$colname\_Search" is only part of the word. The trouble here is that you need to escape the % sign. To do so, add another % before each instance so that PHP knows that the % sign actually means a %, and not a wildcard. This technique is called "escaping." When you are done with the tutorial, you may want to remove the second % signs and then save the page and see what happens when you run the page.

The remaining lines do the same comparison, except the last line, which joins the album.ArtistID to the artist.ArtistID so that you can access the names of the bands from the artist table using the ID provided in the album table.

**Let's look at the last 3 lines now:**

```
$Search = mysql_query($query_Search, $conn1) or die(mysql_error());
$row_Search = mysql_fetch_assoc($Search);
$totalRows_Search = mysql_num_rows($Search);
```

**The Search line takes the query that we just entered, forces it to be run on the database, and provides a die command to kill the process if it cannot connect. The next line creates a variable to hold each returned value from the query. The final line counts of the number of records so that a loop can output a table row for each record.**

```
37
38 mysql_select_db($database_connl, $conn1);
39 $query_catalog = "SELECT * FROM album, artist WHERE album.ArtistID = artist.Ar
40 $catalog = mysql_query($query_catalog, $conn1) or die(mysql_error());
41 $row_catalog = mysql_fetch_assoc($catalog);
42 $totalRows_catalog = mysql_num_rows($catalog);
43
44 $colname_Search = "1";
45 if (isset($_HTTP_GET_VARS['searchcatalog'])) {
46 $colname_Search = (get_magic_quotes_gpc()) ? $_HTTP_GET_VARS['searchcatalog']
47 : addslashes($_HTTP_GET_VARS['searchcatalog']);
48 }
49 mysql_select_db($database_connl, $conn1);
50 $query_Search = "SELECT * FROM album, artist WHERE
51 (album.AlbumName LIKE '%$colname_Search%'
52 OR album.AlbumPrice LIKE '%$colname_Search%'
53 OR artist.ArtistGroupName LIKE '%$colname_Search%'
54 OR artist.ArtistStyle LIKE '%$colname_Search%')
55 AND album.ArtistID = artist.ArtistID";
56 $Search = mysql_query($query_Search, $conn1) or die(mysql_error());
57 $row_Search = mysql_fetch_assoc($Search);
58 $totalRows_Search = mysql_num_rows($Search);
59
60 ?>
61 <html>
```

- **Click File, Save to save your File.**



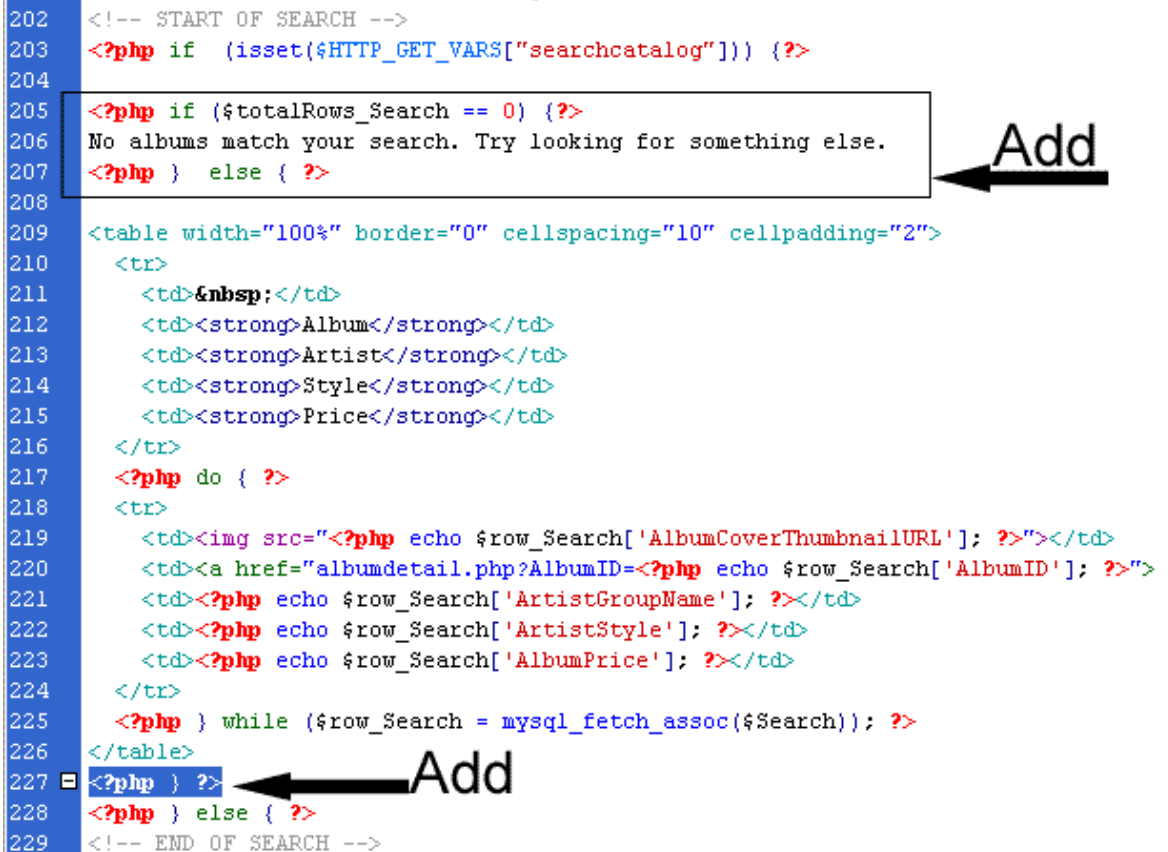


## ► PART 4 Continue – Exceptions to the search

To fix exceptions, you will validate the query to ensure it returns some data, but if it does not display data that it displays a user message. This will be a quick code edit and will make your search page a lot more professional.

5. Find the Comment line `<!-- START OF SEARCH -->`
6. Place your cursor below the tag that checks for the existence of the `HTTP_GET_VARS` and add the following text:

```
202 <!-- START OF SEARCH -->
203 <?php if (isset($_HTTP_GET_VARS["searchcatalog"])) {?>
204
205 <?php if ($totalRows_Search == 0) {?>
206 No albums match your search. Try looking for something else.
207 <?php } else { ?>
208
209 <table width="100%" border="0" cellspacing="10" cellpadding="2">
210 <tr>
211 <td>&nbsp;</td>
212 <td><strong>Album</strong></td>
213 <td><strong>Artist</strong></td>
214 <td><strong>Style</strong></td>
215 <td><strong>Price</strong></td>
216 </tr>
217 <?php do { ?>
218 <tr>
219 <td></td>
220 <td><a href="albumdetail.php?AlbumID=<?php echo $row_Search['AlbumID']; ?>">
221 <td><?php echo $row_Search['ArtistGroupName']; ?></td>
222 <td><?php echo $row_Search['ArtistStyle']; ?></td>
223 <td><?php echo $row_Search['AlbumPrice']; ?></td>
224 </tr>
225 <?php } while ($row_Search = mysql_fetch_assoc($Search)); ?>
226 </table>
227 <?php } ?>
228 <?php } else { ?>
229 <!-- END OF SEARCH -->
```



This block of code adds a test to see if the `$totalRows_Search` variable which we mentioned above is equal to zero. If the value is not there, the first condition executes. If the value is 0, as in there were no records found, then the second condition executes and the line of text will appear instead of the empty table. This makes a good user experience and handles most of the cases that you are going to find with a simple search.

8. Finally, since you added another `ELSE` condition to the page, you need to end it to balance the code. Put your cursor before the `<?php } else { ?>` tag, which is between the search result table and the catalog table and close the `IF` statement by adding: `<?php } ?>`

Save your page and test your `catalog.php`

You can visit my `catalog.php` at <http://www.maxtips.com/dw/3/catalog.php>