

# Accessing data from multiple tables using Adobe GoLive Dynamic Content

By Neill D. Tyler 2003



This work is licensed under a [Creative Commons License](https://creativecommons.org/licenses/by/4.0/).

Out of the box, GoLive does a nice job when it comes to dynamic content. When I began using GoLive 6, I had some difficulty doing anything with dynamic content. I bought the GoLive 6 Magic book... followed the tutorial and was off and running. Many thanks to the authors GoLive 6 Magic for providing documentation that should have shipped with the program. Even after leaping over my first hurdle, I could only access a single table in my MySQL database. From that table, I could display, add, and update the data residing there. I'd worked with other relational databases before, Filemaker Pro, Lotus Approach, and Microsoft Access, and all of them allowed you to link tables together based on common fields and manipulate or display data from each linked table. Through phpMyAdmin, I could create multiple tables in my MySQL database, and according to the documentation, MySQL was relational just like all the other databases I'd worked with. Knowing all this... I still couldn't configure GoLive to act relational.

After banging my head against my desk and having one of my clients tell me I needed to make two tables work, I ran across an article by Derry Thompson that showed how to do two tables. Derry used two content sources and an expression match to pull data from two tables. This worked, and thanks to his tutorial I was able to complete my project. See [www.carriagehousearts.com](http://www.carriagehousearts.com).

In working with SQL recently I ran across some examples of SQL select statements that access data from multiple tables. There are basically two ways, joins and the where statement. Say you have two tables, an artists table and an image table.

Table 1. Artists

```
CREATE TABLE `artist` (  
  `id` tinyint(4) NOT NULL auto_increment,  
  `artist` varchar(255) NOT NULL default '',  
  `artiststtype` varchar(50) default NULL,  
  `email` varchar(255) NOT NULL default '',  
  `url` varchar(255) default NULL,  
  `bio` text,  
  `phone` varchar(50) default NULL,  
  `active` tinyint(1) NOT NULL default '0',  
  PRIMARY KEY (`id`)  
) TYPE=MyISAM COMMENT='Artist infomartion' ;
```

Table 2. Images

```
CREATE TABLE `image` (  
  `image_id` tinyint(4) NOT NULL auto_increment,  
  `imageurl` varchar(255) NOT NULL default '',  
  `imageurlT` varchar(255) NOT NULL default '',  
  `description` varchar(255) NOT NULL default '',  
  `category` varchar(50) default '0',  
  `active` tinyint(1) NOT NULL default '0',  
  `artist_id` tinyint(4) NOT NULL default '0',  
  `smallPrints` char(1) default '1',  
  `largePrints` char(1) default '1',  
  `tShirts` char(1) default '1',  
  `misc` char(1) default '1',
```

```
`code1` int(20) default NULL,  
PRIMARY KEY (`image_id`),  
FULLTEXT KEY `category` (`category`)  
) TYPE=MyISAM COMMENT='Image table for image urls' ;
```

You want to be able to display all of the images for a particular artist. The two ways to do this are with “WHERE” and by performing a join.

Select statement with where.

```
select * from artist, image where artist.id = image.artist_id;
```

The format is:

```
Select * from table1, table2 where table1.field = table2.field;
```

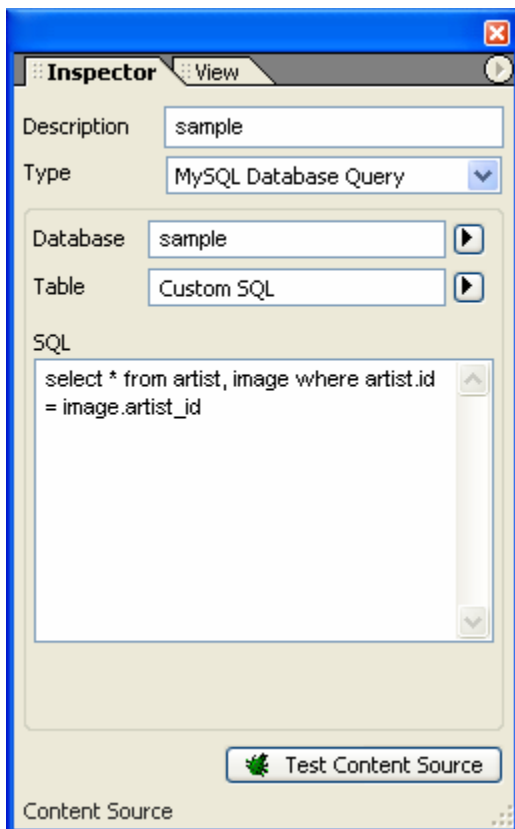
Select statement using a join.

```
select * from artist left join image on artist.id = image.artist_id;
```

The format for the left join is:

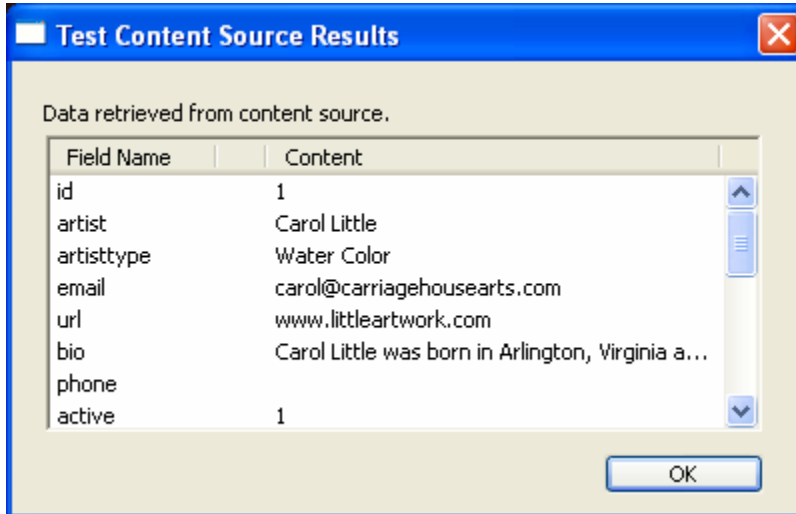
```
Select * from table1 left join on table2 on table1.field = table2.field;
```

So what does this mean for a GoLive user? Well you can select your database and instead of selecting a table select Custom SQL from the dropdown. Then type in your own custom SQL select statement.



After entering your custom statement, click the Test Content Source. This should return the first row in your table (see below).

First half of row:



The screenshot shows a dialog box titled "Test Content Source Results" with a close button in the top right corner. Below the title bar, it says "Data retrieved from content source." Below that is a table with two columns: "Field Name" and "Content". The table contains the following data:

Field Name	Content
id	1
artist	Carol Little
artisttype	Water Color
email	carol@carriagehousearts.com
url	www.littleartwork.com
bio	Carol Little was born in Arlington, Virginia a...
phone	
active	1

An "OK" button is located at the bottom right of the dialog box.

Second half of row:



The screenshot shows a dialog box titled "Test Content Source Results" with a close button in the top right corner. Below the title bar, it says "Data retrieved from content source." Below that is a table with two columns: "Field Name" and "Content". The table contains the following data:

Field Name	Content
phone	
active	1
image_id	2
imageurl	artistImages/1_2.jpg
imageurlT	artistImages/1_2t.jpg
description	Autumn Farm
category	Water Color
14	1

An "OK" button is located at the bottom right of the dialog box.

What this amounts to is that you know have the ability to bind data fields from both tables to you mock content.

The left join will work the same way in the custom SQL statement. The left join and the “where table1.field = table2.field” work for 1 to 1 and a 1 to many relationships.

For more information see the MySql language reference: <http://www.mysql.com/doc/en/JOIN.html>