**SQL Select Statement – Searching Data**

Although in our test database we only have two entries and everything is easy to find, as a database **grows** it is useful to be able to quickly **search** the information. From phpMyAdmin you can do this by selecting your database, and then clicking the **search tab**. Below is an example of how to search for all users under 12 years old.

In our example database this only returned one result, Peggy.

To do this same search from the query window or command line we would type in:

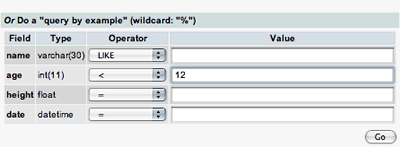
**SELECT \* FROM people WHERE age < 12**

What this does is **SELECT \*(all columns) from the "people" table WHERE the "age" field is a number less than 12.**

If we only wanted to **see the names of people who are under 12 years old** we could run this instead:

**SELECT name FROM people WHERE age < 12**

This may be more helpful if your database contains a lot of fields that are irrelevant to what you are currently searching for.

[](http://0.tqn.com/d/php/1/0/9/-/-/-/MySQL5.gif)

**SQL Delete Statement – Removing Data**

Often you need to **remove old information** from your database. You should be **very careful** when doing this because **once it is gone, it's gone.** That being said, when you are in phpMyAdmin, you can remove information a number of ways. First select the database on the left. One way to remove entries is by then choosing the **browse tab** on the right. Next to each entry you will see a **red X**. **Clicking** the **X will remove the entry, or to delete multiple entries you can check the boxes on the far left and then hit the red X at the bottom of the page.**

Another thing you can do **is click the search tab**. Here you can perform a search. Let's say **the doctor in our example database gets a new partner who is a pediatrician. He will no longer be seeing children, so anyone under 12 needs to be removed from the database. You can perform a search for an age less than 12 from this search screen. All the results are now displayed in the browse format where you can delete individual records with the red X, or check multiple records and click the red X at the bottom of the screen.**

Removing data by searching from a query window or command line is very easy, but being extra **careful**:

**DELETE FROM people WHERE age < 12**

If the table is **no longer needed** you can remove the entire table by clicking on the "**Drop**" tab in phpMyAdmin or running this line:

**DROP TABLE people**

**Views**

In SQL, a view is a **virtual** table based on the **result-set** of an SQL statement.

A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.

You can add SQL functions, WHERE, and JOIN statements to a view and present the data as if the data were coming from one single table.

A view always shows **up-to-date data**! The **database engine recreates and redraws the data**, using the view's SQL statement, **every time a user queries a view**.

**CREATE VIEW view\_name AS  
SELECT column\_name(s)  
FROM table\_name  
WHERE condition**

CREATE VIEW [Current Product List] AS  
SELECT ProductID,ProductName  
FROM Products  
WHERE Discontinued=No

We can query the view above as follows:

SELECT \* FROM [Current Product List]

**Using operators functions**

CREATE VIEW [Products Above Average Price] AS  
SELECT ProductName,UnitPrice  
FROM Products  
WHERE UnitPrice>(SELECT AVG(UnitPrice) FROM Products)

We can query the view above as follows:

SELECT \* FROM [Products Above Average Price]

Note that the view below

selects its data from another view called "Product Sales for 1997":

CREATE VIEW [Category Sales For 1997] AS  
SELECT DISTINCT CategoryName,Sum(ProductSales) AS CategorySales  
FROM [Product Sales for 1997]  
GROUP BY CategoryName

We can query the view above as follows:

SELECT \* FROM [Category Sales For 1997]

We can also add a condition to the query. Now we want to see the total sale only for the category "Beverages":

SELECT \* FROM [Category Sales For 1997]  
WHERE CategoryName='Beverages'

You can update a view by using the following syntax:

### SQL CREATE OR REPLACE VIEW Syntax

**CREATE OR REPLACE VIEW view\_name AS  
SELECT column\_name(s)  
FROM table\_name  
WHERE condition**

Now we want to add the "Category" column to the "Current Product List" view. We will update the view with the following SQL:

CREATE VIEW [Current Product List] AS  
SELECT ProductID,ProductName,Category  
FROM Products  
WHERE Discontinued=No

**Member table**

CREATE TABLE member (

loginName VARCHAR (20) NOT NULL,

createdate DATE NOT NULL,

password VARCHAR (255) NOT NULL,

lastname VARCHAR(50),

firstnmame VARCHAR(40),

street VARCHAR(50),

city VARCHAR(50),

state CHAR(2),

zip CHAR(10),

email CHAR(50),

phone CHAR(15),

fax CHAR(15),

state CHAR(2),

PRIMARY KEY(loginName)  
);