Week 12: SQL Part 2

Agenda

- Overview
  - Final Project Review
  - What's Due?
- SQL Continued
  - SQL Language Types
  - Query Examples
  - Setting Up a Database on 000webhost
  - Combining SQL and PHP
- Database Challenge 2
- Next Week

Overview

Last week, we covered some concepts in SQL including how to select, insert, update, or delete data in a database. This week, we will go over a few more useful commands and keywords that will allow us to create complicated queries. Please make sure you have created a 000webhost account and a database in that account as well (database should have been created last week).

Final Project Review

The Final Project Review deadline has been extended to May 4th, because why not? May the 4th be with you! I'm excited to see your work closer to the finished product in any case. The Final Project Submission is due only a week after the Review, so make sure you are close to finishing it.

The Review portion of the final project is essentially a quick presentation on your work done so far. You can use a video, screen shots, a word document, etc. to talk about your website. Why did you choose this topic? How much progress have you made? What were some obstacles? What features are you proud of? Why did you choose that color scheme? These are some questions you can answer in your review.

Look over the Final Project Details inside of the Final Project folder for more information on the requirements. It doesn't have to be a long presentation, but you should show enough of your website that anyone could understand what it was about and how it works.

I will allow you to present your work during office hours that week as an alternative. You must email me first. The Final Project Review is **due on Thursday, May 4h at midnight.**

What's Due

- Database Challenge 2

SQL Continued

Last week, we learned a bit about databases, and SQL, a language used to manipulate and access databases. While there are many different versions of SQL, the concepts learned in this course should apply to most of them. Since we are using 000webhost, the Database Management System (DBMS) in place is MySQL.

SQL Language Types

If you recall from last week, we used SQL statements to retrieve data from a database or alter it in some way. Those statements served as queries to our database and returned results, changed items inside of the database, or could have changed the database itself. In general, there are five major categories for SQL queries.

- Data Query Language (DQL)
- Data Manipulation Language (DML)
- Data Definition Language (DDL)
- Data Control Language (DCL)
- Transaction Control Language (TCL)

Data Query Language is used to retrieve data from a database. The only command associated with this is SELECT.

```
SELECT Column_Name
FROM Table
WHERE Conditions;
```

In the first line, you chose what columns or data to collect. In the second line starting with FROM, you chose the table or tables you want to query. The third line is where you supply condition to narrow your query. Please note, queries normally have all code appearing on the same line, but I broke them apart for better readability.

Data Manipulation Language is used to modify the data inside of a database. Three important commands that we saw last week were INSERT, UPDATE, and DELETE. Below is an example of an insert statement.

```
INSERT INTO [table] (column1, column2,...) VALUES ('value1',
'value2',…);
```

This tells the database to insert a new row into the table with specified values for each column. Every column provided must match the column in the database exactly and must have an accompanying value to go with it. If you supply 3 columns, 3 values should also be provided. The order of the columns in the query should follow the column order of your table as well.

Data Definition Language is used to define the database structure or schema in terms of datatypes. CREATE and DROP were examples from last week which allow us to manipulate the structure of a database. Below is an example of general CREATE TABLE syntax.

Creating a table:

```
CREATE TABLE [table_name] (
     [Column1] [datatype],
     [Column2] [datatype],
     …
);
```

Data Control Language is used to provide or set permissions for users in a database. GRANT and REVOKE are some examples of commands you would use in this type of language.

Transaction Control Language is used to manage changes made by DML based queries. For example, if a user updated every record in the database, you might use a ROLLBACK command to restore the values.

Query Examples

We are going to review how to query multiple tables which contain similar column names. We'll also look at some key words and commands which will help us narrow our retrieved results. Use the following tables for the examples listed below. OwnerID and PetID are the primary keys and Owner is the foreign key for the Pets Table.

Owners

OwnerID	Name	Age	Email
1	Chris Velez	30	cvelez@albany.edu
2	John Doe	23	jodoe@mail.com
3	Jane Doe	25	Jadoe@mail.com

Pets

PetID	Name	Type	Owner	Age
1	Doggo	Dog	1	4
2	Kitty	Cat	3	10
3	Birdy	Bird	1	11
4	Hamtaro	Hamster	2	2

Multiple Table Queries

In most applications, there are a lot of tables which contribute to the overall database. To use the data in those tables, we will often need to query multiple at the same time. In some instances, those tables might have similarly named columns. Depending on the structures, your syntax may change when trying to perform a query on a database.

This example selects all owners, their pets, and emails from both tables. Since we have similarly named columns in both tables, we must specify the table name followed by a period and then the column name. In the FROM portion, we name both tables. In our WHERE clause, we use the table name and period to specify which column we are referring to in our query.

```
SELECT Owner.Name, Pets.Name, Email
FROM Owners, Pets
WHERE Pets.Owner = Owners.OwnerID
```

If we wanted to get the names and ages of all owners older than 25 with dogs, we could use a query like the one below. We have two columns that have the same name in both tables, so we use the table.columName format to select them. Both tables must be mentioned in the FROM portion and our conditions are in the WHERE clause. In this case, the owner's age must be greater than 25 and the owner's ID must be the same as the pet's ID.

```
SELECT Owner.Name, Owner.Age, Pets.Name, Pets.Age
FROM Owners, Pets
```

```
WHERE Owner.Age > 25 AND (Pets.Owner = Owners.OwnerID)
```

ALTER (DDL)
This command is useful for adding, dropping, or altering the data type of any column in an existing table. Ideally, this isn't something you would allow users to do on your website.

Adding a column to our table:
```
ALTER TABLE [table_name]
ADD [column] [data type];
```

Deleting a column from a table:
```
ALTER TABLE [table_name]
DROP [column];
```

Changing a data type:
```
ALTER TABLE [table_name]
MODIFY COLUMN [column] [data type];
```

TRUNCATE (DDL)
This allows a user to delete the content of a table but not the table itself. It is equivalent to `DELETE * FROM [table_name]`. In the example below we would be deleting all the content inside our Owners table.

```
TRUNCATE TABLE Owners;
```

ORDER BY
The ORDER BY clause allows us to sort our returned results from a query. We can choose the order with ASC (ascending) or DESC (descending).

```
ORDER BY [column1, column2, …] [ASC | DESC]
```

Select all owners and order them by their name
```
SELECT *
FROM Owners
ORDER BY Name ASC;
```

DISTINCT
This allows us to select distinct records from a table that may contain duplicate data. If we wanted to see all the unique pets in our Pets table, distinct would be useful.

```
SELECT DISTINCT Type
FROM Pets;
```

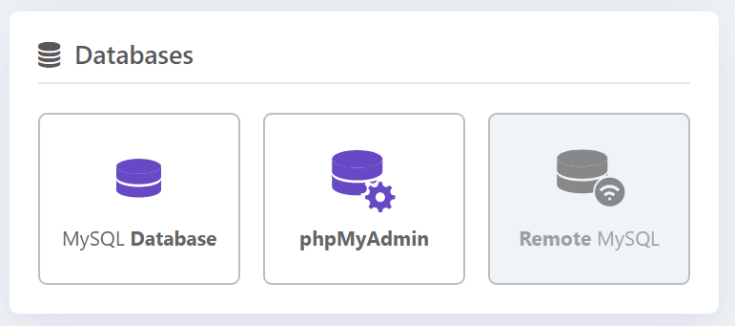
BETWEEN
This allows us to select records from a table between a certain range of values whether they are numbers, text, or dates.

```
SELECT columns
FROM table
WHERE column1 BETWEEN value1 AND value 2;
```

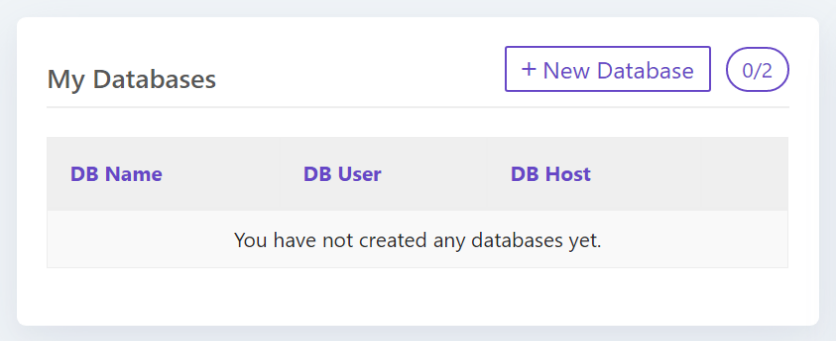
It is similar to saying WHERE column1 >= value1 AND column1 <= value2.

Setting Up a Database on 000webhost (same as last week)

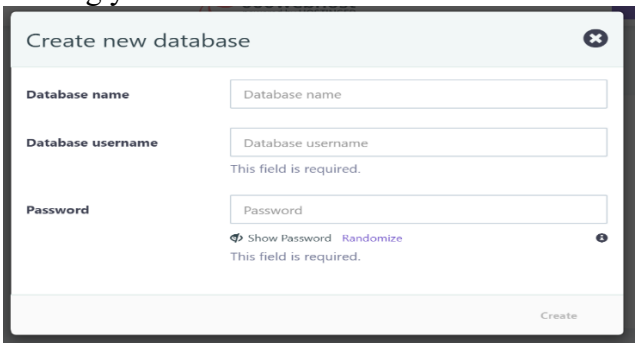
Before we can get fancy with PHP and SQL, we need to set up our database on 000webhost. Creating a database on 000webhost is quick and easy to do. Log on to your 000webhost account. In your website's dashboard area, which contains your File Manager, scroll down to the section titled "Databases" and click the first option, "MySQL Database."



After clicking "MySQL Database" you will be brought to a screen with the following content on it. Click "+ New Database" to create a new database. Another screen will appear asking for credentials for your database.

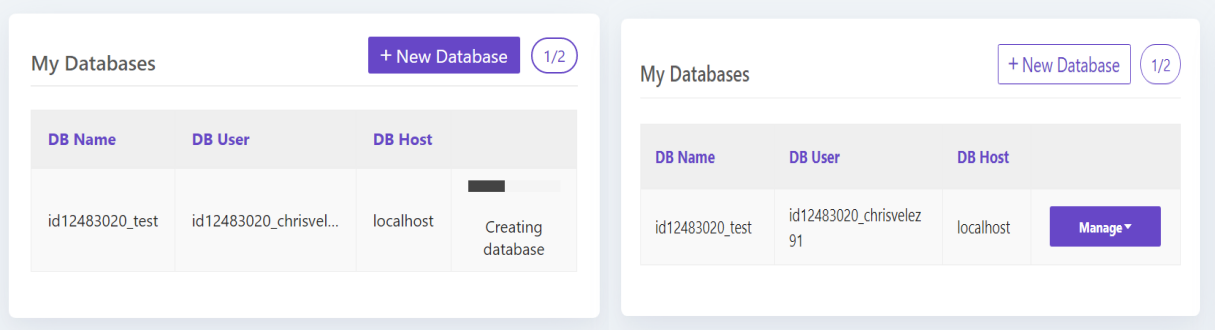


On the next screen called "Create new database," you will add a database name, username, and password. **MAKE SURE YOU STORE THIS INFO SOMEWHERE.** This information is vital for establishing a connection to your database using PHP code. Once your info is typed in, click "Create" to finish creating your database.



After clicking "Create," the screen will close and one like the following will appear. You must wait for the "Creating database" progress bar to be finished before your database is available.

That section will change to a purple "Manage" button with a dropdown arrow. The images below depict a before/after view of when your database is loaded versus when it is finished.



Examples
A new video will be released on Tuesday afternoon. This video will cover PHP and SQL in the context of this week's assignment. When it is available, the link will be added to the assignment description.

My Examples
This week's class example will not be live. Instead, you'll need to place the page on your 000webhost area for it to fully function. It will allow you to insert and update records in your table. You'll need to enter your credentials for your database into the pwd.php file. Place the pwd.php and index.php file in the same folder on your 000webhost account and the page should work for you

PowerPoint documents have also been released under Blackboard → Course Materials → Resources →PHP & SQL. One of them offers a brief introduction to PHP/SQL while the other will serve as a debugging guide.

You won't be able to right-click the page and view the source this week to view my PHP code. Since PHP is processed by the server and rendered as HTML, the user can't see the PHP code you're using. The code for the example below is in the "examples" folder that was included with this week's lecture notes zip folder. You can take all the contents of the examples folder and place them somewhere on your 000webhost account and the pages should work. There are comments in the code to explain what is happening. Feel free to look at the live link to see the page in action.

Here's a link to **all** course examples:
https://cinf362.000webhostapp.com/examples/ (PHP/SQL Examples)
https://www.albany.edu/~cv762525/cinf362/examples/ (HTML, CSS, and JS examples)
https://www.albany.edu/~cv762525/cinf362/videos/ (videos)
https://drive.google.com/drive/folders/13sh0oaUeE9di4aZuTYczqNdpzdb4kzKE?usp=sharing (more videos)

Database Challenge 2

Due Monday, April 25th at midnight
Download the "Database-Challenge-2.zip" folder from Blackboard under this week's Lecture Notes. Inside of the zip folder will be files for this week's assignment. Your task will be to add HTML/PHP to the files provided to complete the challenge described below. Look for the following lines of code to know where you need to start:

// ADD CODE FOR YOUR INSERT STATEMENT HERE

<!-- ADD YOUR INPUTS, LABELS, AND NEW BUTTON TO THIS FORM -->

The Database-Challenge-2.php page provided is almost an exact replica of the class example from last week. The main differences are that I removed some content and added comments to specify where your code for this assignment should go. The page also doesn't have the update feature built into it. You can also refer to this week's example for the syntax necessary for the INSERT statement.

Your task is to add HTML/PHP so that a user can use your form to manually enter a new customer into your database. The syntax for this is inside of the class example. However, you'll need to use string concatenation to place the user's supplied values inside of your INSERT statement. A previous class example entered 3 separate rows at once, but for the assignment, you will only insert one row at a time. You'll need to look at the syntax in the example and alter it a little bit. The case/message portions should be similarly structured (case: 'insert' and a message which says "Person added successfully."

I will check your work by going to your page and entering data/clicking submit. After I have submitted data, I'll click "View Customer Table" to ensure that it's working. Please make sure you are not deleting any code as any deletions could result in the page not working correctly. You should only be adding in them items I mentioned already or in the General Steps section below.

General Tips

- Add 3 inputs and labels for each piece of data (first name, last name, and email)
- Add a button with type and name attributes set to submit. It should also have a value of "insert" to submit this inserted data
- Create a new "case" in the switch statement for insert
- Create a variable for the insert SQL statement ($sql)
- Create a variable for the message ($message) to output
- Use the other SQL statement structures and the class example as a guide
- https://www.tutorialspoint.com/sql/sql-insert-query.htm (Insert Syntax)

Your work should be submitted through 000webhost. If you are experiencing issues with your code, I would review Lu's SQL Debugging Guide on Blackboard → Course Materials → Resources → PHP & SQL → SQL Debugging Guide.

Your webpage and queries are **due on Monday, April 25th at midnight.** To submit the work for this exercise, visit Blackboard → Course Materials → Lectures Notes for this week's class. You can also go into the "Assignments" folder and the submission area will be titled "Database Challenge 2." You should be submitting a link to the Database-Challenge-2.php" page. The work submitted will be evaluated based on the rubric explained below.

Database Challenge 2 Rubric – 10pts

- Live link was provided – 1pt
- Zip file provided – 1pt
- Database Functions – 8pts
    - General functions are working – 2pts

- PHP is easy to read – 2pts
- I can insert content into the database – 2pts
- Correct output is generated when insert is successful or fails – 2pts

Next Week

Next week, we will learn about web security and how we can implement it in our web pages.