CSE417: WEB ENGINEERING

Daffodil International University

Learning Outcomes

You will learn

- SQL
- To access mySQL database
- To create a basic mySQL database
- To use some basic queries
- To use PHP and mySQL

Introduction to SQL

SQL is an ANSI (American National Standards Institute) standard computer language for accessing and manipulating databases.

- SQL stands for Structured Query Language
- using SQL can you can
  - access a database
  - execute queries, and retrieve data
  - insert, delete and update records
- SQL works with database programs like **MS Access, DB2, Informix, MS SQL Server, Oracle, Sybase, mySQL**, etc.

Unfortunately, there are many different versions. But, they must support the same major keywords in a similar manner such as SELECT, UPDATE, DELETE, INSERT, WHERE, etc.

Most of the SQL database programs also have their own proprietary extensions!

The University of Liverpool CS department has a version of mySQL installed on the servers, and it is this system that we use in this course. Most all of the commands discussed here should work with little (or no) change to them on other database systems.

SQL Database Tables

A database most often contains one or more tables. Each table is identified by a name (e.g. "Customers" or "Orders"). Tables contain records (rows) with data.

For example, a table called "Persons":

LastName	FirstName	Address	City
Hansen	Ola	Timoteivn 10	Sandnes
Svendson	Tove	Borgvn 23	Sandnes
Pettersen	Kari	Storgt 20	Stavanger

The table above contains three records (one for each person) and four columns (**LastName, FirstName, Address, and City**).

SQL Queries

With SQL, you can query a database and have a result set returned.

A query like this:

```
SELECT LastName FROM Persons;
```

gives a result set like this:

LastName
Hansen
Svendson
Pettersen

The mySQL database system requires a semicolon at the end of the SQL statement!

SQL Data Languages

The query and update commands together form the Data Manipulation Language (DML) part of SQL:

- SELECT - extracts data from a database table
- UPDATE - updates data in a database table
- DELETE - deletes data from a database table
- INSERT INTO - inserts new data into a database table

The Data Definition Language (DDL) part of SQL permits database tables to be created or deleted:

- CREATE TABLE - creates a new database table
- ALTER TABLE - alters (changes) a database table
- DROP TABLE - deletes a database table
- CREATE INDEX - creates an index (search key)
- DROP INDEX - deletes an index

*Here we will use some of them in mySQL

Logging into mySQL Server

You can log into our mySQL server from Linux by typing in the prompt

```
bash-2.05b$ mysql -h mysql martin -u martin
```

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 209201 to server version: 5.0.22

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

From here you can create, modify, and and drop tables, and modify the data in your tables. But first, you must specify which database on the server you want to use.

```
mysql> use martin;
```

```
Database changed
```

Creating a Table

You can create a table you might use for the upcoming project. For example,

```
mysql> CREATE TABLE students(
    -> num INT NOT NULL AUTO_INCREMENT,
    -> f_name VARCHAR(48),
    -> l_name VARCHAR(48),
    -> student_id INT,
    -> email VARCHAR(48),
    -> PRIMARY KEY(num));
```

Hit **Enter** after each line (if you want). MySQL doesn't try to interpret the command itself until it sees a semicolon (;)

(The "->" characters you see are not typed by you.)

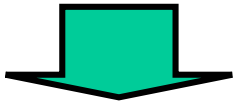
```
Query OK, 0 rows affected (0.02 sec)
```

*If the server gives you a big ERROR, just try again from the top!

Viewing The Table Structure

Use DESCRIBE to see the structure of a table

```
mysql> DESCRIBE students;
```



```
+------------+-------------+------+-----+---------+----------------+
| Field      | Type        | Null | Key | Default | Extra          |
+------------+-------------+------+-----+---------+----------------+
| num        | int(11)     | NO   | PRI | NULL    | auto_increment |
| f_name     | varchar(48) | YES  |     | NULL    |                |
| l_name     | varchar(48) | YES  |     | NULL    |                |
| student_id | int(11)     | YES  |     | NULL    |                |
| email      | varchar(48) | YES  |     | NULL    |                |
+------------+-------------+------+-----+---------+----------------+
```

Inserting Data

Using `INSERT INTO` you can insert a new row into your table. For example,

```
mysql> INSERT INTO students
    -> VALUES(NULL,'Russell','Martin',396310,'martin@csc.liv.ac.uk');
```

```
Query OK, 1 row affected (0.00 sec)
```

Using `SELECT FROM` you select some data from a table.

```
mysql> SELECT * FROM students;
```

```
+-----+---------+--------+------------+----------------------+
| num | f_name  | l_name | student_id | email                |
+-----+---------+--------+------------+----------------------+
|   1 | Russell | Martin |     396310 | martin@csc.liv.ac.uk |
+-----+---------+--------+------------+----------------------+
1 row in set (0.00 sec)
```

Inserting Some More Data

You can repeat inserting until all data is entered into the table.

```
mysql> INSERT INTO students
    -> VALUES(NULL,'James','Bond',007,'bond@csc.liv.ac.uk');
Query OK, 1 row affected (0.01 sec)


mysql> SELECT * FROM students;
+-----+---------+--------+------------+----------------------+
| num | f_name  | l_name | student_id | email                |
+-----+---------+--------+------------+----------------------+
|   1 | Russell | Martin |     396310 | martin@csc.liv.ac.uk |
|   2 | James   | Bond   |          7 | bond@csc.liv.ac.uk   |
+-----+---------+--------+------------+----------------------+
2 rows in set (0.00 sec)
```

Note: The value “NULL” in the “num” field is automatically replaced by the SQL interpreter as the “auto_increment” option was selected when the table was defined.

Getting Data Out of the Table

- The SELECT command is the main way of getting data out of a table, or set of tables.

```
SELECT * FROM students;
```

Here the asterisk means to select (i.e. return the information in) all columns.

You can specify one or more columns of data that you want, such as

```
SELECT  f_name,l_name FROM students;
```

```
+---------+--------+
| f_name  | l_name |
+---------+--------+
| Russell | Martin |
| James   | Bond   |
+---------+--------+
2 rows in set (0.00 sec)
```

Getting Data Out of the Table (cont.)

- You can specify other information that you want in the query using the **WHERE** clause.

```
SELECT * FROM students WHERE l_name='Bond';
```

```
+-----+---------+--------+------------+---------------------+
| num | f_name  | l_name | student_id | email               |
+-----+---------+--------+------------+---------------------+
|   2 | James   | Bond   |          7 | bond@csc.liv.ac.uk  |
+-----+---------+--------+------------+---------------------+
1 row in set (0.00 sec)
```

```
SELECT student_id, email FROM students WHERE l_name='Bond';
```

```
+------------+---------------------+
| student_id | email               |
+------------+---------------------+
|          7 | bond@csc.liv.ac.uk  |
+------------+---------------------+
1 row in set (0.00 sec)
```

Altering the Table

The `ALTER TABLE` statement is used to add or drop columns in an existing table.

```
mysql> ALTER TABLE students ADD date DATE;
```

```
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

```
mysql> SELECT * FROM students;
+-----+---------+--------+------------+----------------------+------+
| num | f_name  | l_name | student_id | email                | date |
+-----+---------+--------+------------+----------------------+------+
|   1 | Russell | Martin |     396310 | martin@csc.liv.ac.uk | NULL |
|   2 | James   | Bond   |          7 | bond@csc.liv.ac.uk   | NULL |
+-----+---------+--------+------------+----------------------+------+
2 rows in set (0.00 sec)
```

Updating the Table

The UPDATE statement is used to modify data in a table.

```
mysql> UPDATE students SET date='2007-11-15' WHERE num=1;
```

```
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

```
mysql> SELECT * FROM students;
+-----+---------+--------+------------+----------------------+------------+
| num | f_name  | l_name | student_id | email                | date       |
+-----+---------+--------+------------+----------------------+------------+
|   1 | Russell | Martin |     396310 | martin@csc.liv.ac.uk | 2007-11-15 |
|   2 | James   | Bond   |          7 | bond@csc.liv.ac.uk   | NULL       |
+-----+---------+--------+------------+----------------------+------------+
2 rows in set (0.00 sec)
```

Note that the default date format is “YYYY-MM-DD” and I don’t believe this default setting can be changed.

Deleting Some Data

The DELETE statement is used to delete rows in a table.

```
mysql> DELETE FROM students WHERE l_name='Bond';
```

```
Query OK, 1 row affected (0.00 sec)
```

```
mysql> SELECT * FROM students;
+-----+---------+--------+------------+----------------------+------------+
| num | f_name  | l_name | student_id | email                | date       |
+-----+---------+--------+------------+----------------------+------------+
|   1 | Russell | Martin |     396310 | martin@csc.liv.ac.uk | 2006-11-15 |
+-----+---------+--------+------------+----------------------+------------+
1 row in set (0.00 sec)
```

The Final Table

We'll first add another column, update the (only) record, then insert more data.

```
mysql> ALTER TABLE students ADD gr INT;
Query OK, 1 row affected (0.01 sec)
Records: 1  Duplicates: 0  Warnings: 0
mysql> SELECT * FROM students;
+-----+---------+--------+------------+----------------------+------------+------+
| num | f_name  | l_name | student_id | email                | date       | gr   |
+-----+---------+--------+------------+----------------------+------------+------+
|   1 | Russell | Martin |     396310 | martin@csc.liv.ac.uk | 2007-11-15 | NULL |
+-----+---------+--------+------------+----------------------+------------+------+
1 row in set (0.00 sec)
mysql> UPDATE students SET gr=3 WHERE num=1;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> SELECT * FROM students;
+-----+---------+--------+------------+----------------------+------------+------+
| num | f_name  | l_name | student_id | email                | date       | gr   |
+-----+---------+--------+------------+----------------------+------------+------+
|   1 | Russell | Martin |     396310 | martin@csc.liv.ac.uk | 2007-11-15 |    3 |
+-----+---------+--------+------------+----------------------+------------+------+
1 row in set (0.00 sec)

mysql> INSERT INTO students VALUES(NULL,'James','Bond',007,'bond@csc.liv.ac.uk','2007-
11-15', 1);
. . .
. . .
```

The Final Table (cont.)

```
. . .
. . .
mysql> INSERT INTO students VALUES(NULL,'Hugh,'Milner',75849789,'hugh@poughkeepsie.ny',
    CURRENT_DATE, 2);
```

Note: **CURRENT_DATE** is a built-in SQL command which (as expected) gives the current (local) date.

```
mysql> SELECT * FROM students;
+-----+---------+----------+------------+----------------------------+------------+------+
| num | f_name  | l_name   | student_id | email                      | date       | gr   |
+-----+---------+----------+------------+----------------------------+------------+------+
|   1 | Russell | Martin   |     396310 | martin@csc.liv.ac.uk       | 2007-11-15 |    3 |
|   5 | Kate    | Ash      |     124309 | kate@ozymandius.co.uk      | 2007-11-16 |    3 |
|   3 | James   | Bond     |          7 | bond@csc.liv.ac.uk         | 2007-11-15 |    1|
|   4 | Bob     | Jones    |      12190 | bob@nowhere.com            | 2007-11-16 |    3 |
|   6 | Pete    | Lofton   |         76 | lofton@iwannabesedated.com | 2007-11-17 |    2 |
|   7 | Polly   | Crackers |       1717 | crackers@polly.org         | 2007-11-17 |    1|
|   8 | Hugh    | Milner   |   75849789 | hugh@poughkeepsie.ny       | 2007-11-17 |    2 |
+-----+---------+----------+------------+----------------------------+------------+------+
7 rows in set (0.00 sec)
mysql> exit
Bye
```

Other SQL Commands

- SHOW tables; gives a list of tables that have been defined in the database
- ALTER TABLE students DROP email; would drop the "email" column from all records
- DROP TABLE students; deletes the entire "students" table, and its definition **(use the DROP command with extreme care!!)**
- DELETE FROM students; removes all rows from the "students" **table (so once again, use the DELETE command with great caution)**, the table definition remains to be used again
- A more useful command is something like

  DELETE FROM students WHERE (num > 5) AND (num <= 10);

  which selectively deletes students based on their "num" values (for example).

- HELP; gives the SQL help
- HELP DROP; gives help on the DROP command, etc.

Backing up/restoring a mySQL database

- You can back up an entire database with a command such as

```
mysqldump –h mysql –u martin martin > backup.sql
```

(Run from the Unix command line.)

- This gives a script containing SQL commands to reconstruct the table structure (of all tables) and all of the data in the table(s).
- To restore the database (from scratch) you can use this type of Unix command:

```
mysql –h mysql –u martin martin < backup.sql
```

- Other commands are possible to backup/restore only certain tables or items in tables, etc. if that is what you desire. For example

```
mysqldump –h mysql –u martin martin books clients> backup.sql
```

stores information about the “books” and “clients” tables in the “martin” database.

PHP: MySQL Database [CRUD Operations]

To do any operations, you need to Connect to your database first!

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
// Create connection
$conn = mysqli_connect($servername, $username, $password,
$dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
?>
```

Inserting Data

```
//Remember, you always connect to your database first
//Assumption: We have a Table named MyGuests
```

```
$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if (mysqli_query($conn, $sql)) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}
```

Reading Data

```
//Remember, you always connect to your database first
//Assumption: We have a Table named MyGuests


$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo "id: " . $row["id"]. " - Name: " .
$row["firstname"]. " " . $row["lastname"]. "<br>";
}
} else {
    echo "0 results";
}
```

More..

- Did you close your connection?

```
mysqli_close($conn);
//put this at the end of operation
```

- Similarly you can update and delete data
-
- More here : PHP: MySQL Database

- There are slides hidden and supposed to help you doing more complex opertions
- These will not be covered in Class Lecture.
- Most of them are already explained to you in DATABASE MANAGEMENT SYSTEM course

Attention!

- There is easier way to do using PHPMyadmin accesses via localhost (after XAMPP installation)
- Remember, you will not always get flexibilty and,
- You love command line tools to write code!
- Cheers!

Exercise

- Show one example of each of CRUD operation
- Do two complex operations
- **READINGS/Practice**
  - M Schafer: Ch. 31
  - W3 schools
  - http://www.php-mysql-tutorial.com/
  - http://www.sitepoint.com/php-security-blunders/
  - http://php.net/manual/en/mysqli.quickstart.prepared-statements.php

Acknowledgement

- This module is designed and created with the help from following sources-
  - https://cgi.csc.liv.ac.uk/~ullrich/COMP519/
  - http://www.csc.liv.ac.uk/~martin/teaching/comp519/
  -