# Welcome to (the Second evening of) ICA40511 Certificate IV in Programming (MySQL studies)

Firstly, I’d like to briefly discuss possible solutions if your find XAMPP is very slow to return MYSQL queries.

## XAMPP ISSUES

xampp-win32-7.0.4-0-VC14-installer.exe

Within the my.ini file:

#----------------------------------------------------

# !!!! Query Cache Config !!!!

#----------------------------------------------------

query-cache-size = 524288000

query-cache-limit = 5242880

query-cache-type = 1

#----------------------------------------------------

# !!!! InnoDB Buffer Config !!!!

#----------------------------------------------------

innodb-buffer-pool-size = 2000M

innodb-additional-mem-pool-size = 400M

innodb-log-files-in-group = 2

innodb-log-buffer-size = 10M

innodb-file-per-table = 1

Have to open Notepad as Administrator – Then update C:\Windows\System32\drivers\etc\hosts

**# localhost name resolution is handled within DNS itself.**

**Try to add these lines to the hosts file:**

**127.0.0.1 127.0.0.1**

**127.0.0.1 localhost**

**#::1 localhost**

Under Settings (Picture of GOG) at top right-hand side, brings up Page related settings.

Click on **Retain Query Box** option

Update PHPAdmin steps:

<http://stackoverflow.com/questions/27131029/how-to-upgrade-phpmyadmin-in-xampp-to-latest>

config.inc.php <== Copy this file to the desktop, and then delete all the files in the c:\xampp\phpAdmin directory

\*\*\*\*\*\*\*\*\*

## IN Windows 7 VIRTUAL MACHINE

Discuss XAMPP installation and its issues and solutions – As required

See Issues and solutions.doc

**Programs such as SKYPE can interfere with the start-up of XAMPP, as they use the same port numbers.**

Firstly, we’ll go through the installation of the XAMPP software (in a virtual machine).

<https://www.youtube.com/watch?v=rjeFuGi47Lk>

Root password:

In regards to the setting of the root password, this doesn’t need to be done for our lessons.

Don’t set the root password

mysqladmin.exe -u root password secret

Now that XAMPP software has been installed and the root password set, we can look at using the XAMPP

## MySQL 5.7 Reference guide

http://dev.mysql.com/doc/refman/5.7/en/show-columns.html

# Fundamental MySQL Database Commands

### Syntax for creating the EMPLOYEES database

CREATE DATABASE EMPLOYEES;

**Click on GO within the MYSQL screen;**

**(Click on Retain query box, to keep the MySQL command window)**

Once the database (EMPLOYEES) has been created, we can now create a **table**, which will hold our data.

Firstly, make sure that you select the database you’re working with by using one of the following methods:

* Using the USE statement

USE students

**OR**

* Selecting the name of the required database from the drop down list of databases, within the GUI.

### Creating the Emp\_Table table with three (3) Columns

Firstly, we’ll create the Emp\_Table, within the Database Employees, with only three (3) columns Emp\_First\_Name , Emp\_Emp\_Last\_Name and Employee\_ID, using the following syntax.

**USE EMPLOYEES / Select EMPLOYEES database**

CREATE TABLE Emp\_Table

(

Emp\_First\_Name varchar(20) not null,

Emp\_Last\_Name varchar(20) not null,

Emp\_ID int not null

);

This command created the Emp\_Table with three (3) columns Emp\_First\_Name , Emp\_Last\_Name and Emp\_ID. The first two are of type varchar, meaning they will accept up to 20 characters and spaces. The last column was created to accept numbers.

Having created this table, we can now easily add other columns to it without the need to delete it and start again, by using the ALTER command.

## ALTER Command

### Adding a 4th Column called “Salary” to Emp\_Table, with the field characteristics of “decimal” (or “numeric”)

Here we are going to add a non-mandatory column to the Emp\_Table, so we’re setting it to allow NULL (or no values), for all or some records as the need requires.

Now we can add a fourth column called **Emp\_Salary** to account for the employee’s payroll. To make this change we will use the **ALTER** keyword as follows:

**USE EMPLOYEES**

ALTER TABLE Emp\_Table ADD Emp\_Salary decimal(10,2) NULL;

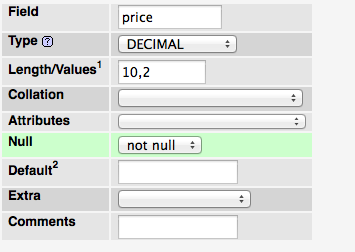
We use “decimal” instead of “float” as accuracy is most important in this field.

Also we have added the keyword NULL to the end to allow for nothing to be added to this field, say in the case of a contractor, or person who has started as yet, etc.

Note: You can use DECIMAL or NUMERIC as both have the same meaning in MySQL:

The DECIMAL and NUMERIC types store exact numeric data values. These types are used when it is important to preserve exact precision, for example with monetary data. In MySQL, NUMERIC is implemented as DECIMAL, so the following remarks about DECIMAL apply equally to NUMERIC. : [**MySQL**](http://dev.mysql.com/doc/refman/5.0/en/fixed-point-types.html)

i.e. DECIMAL(10,2)



### Using the ALTER Command to ADD a 5th column to the Emp\_Table called “Position”, within the EMPLOYEES Database

Having now created a fourth field in the table, we also want to add a fifth field, which is called position, with the following command:

**USE EMPLOYEES**

ALTER TABLE Emp\_Table ADD Emp\_Position varchar(20) null;

# ADDING RECORDS

Having now added a 5th column into the table called **Emp\_Position**, we now want to add records to the table. To accomplish this we will use the INSERT INTO command.

## INSERT INTO

### The INSERT INTO command is used to add a record into the Emp\_Table, which now has five (5) fields

We can use one of three (3) methods for inserting data into the Emp\_Table.

#### Method One:

**Add data values for single records, specifying the column names for every INSERT statement:**

INSERT INTO tbl\_name (col\_A,col\_B,col\_C,col\_D) VALUES (1,2,3,4)

**USE EMPLOYEES**

INSERT INTO Emp\_Table (Emp\_First\_Name , Emp\_Last\_Name , Emp\_ID, Emp\_Salary, Emp\_Position) VALUES ('David', 'Jones', 10045, 45000.00, 'Manager');

INSERT INTO Emp\_Table (Emp\_First\_Name , Emp\_Last\_Name , Emp\_ID, Emp\_Salary, Emp\_Position) VALUES ('Tom', 'Rollins', 10046, 50000.00, 'Programmer');

INSERT INTO Emp\_Table (Emp\_First\_Name , Emp\_Last\_Name , Emp\_ID, Emp\_Salary, Emp\_Position) VALUES ('Bob', 'Hanover', 10047, 37000.00, 'Printer');

#### Method Two:

**Insert the data values for multiple records, without specifying the column names:**

INSERT INTO tbl\_name VALUES (1,2,3), (4,5,6), (7,8,9)

INSERT INTO Emp\_Table VALUES ('David', 'Bowie', 10051, 45200.00,'Entertainer'), ('James', 'Bond', 10052, 55000.00,'Spy Master'), ('Homer', 'Simpson', 10053, 30000.00,'Safety Manager');

#### Method Three:

**Insert the data values for multiple records, specifying the column names:**

INSERT INTO tbl\_name (col\_A,col\_B,col\_C) VALUES (1,2,3), (4,5,6), (7,8,9)

#### INSERT INTO Emp\_Table (Emp\_First\_Name , Emp\_Last\_Name , Emp\_ID, Emp\_Salary, Emp\_Position) VALUES ('Brian', 'Harris', 10054, 44000.00, 'Truck Driver'),('Harry', 'Smith', 10055, 50000.00, 'Printer'), ('Richard', 'Hanson', 10056, 60000.00, 'Senior Manager');

#### Method Four:

**Add data values for single records, without specifying the column names for every INSERT statement:**

INSERT INTO tbl\_name VALUES (1,2,3,4)

**USE EMPLOYEES**

INSERT INTO Emp\_Table VALUES ('Alan', 'Jones', 10048, 36500.00, 'Office Manager');

[INSERT](http://localhost/phpmyadmin/url.php?url=http://dev.mysql.com/doc/refman/5.5/en/insert.html) INTO Emp\_Table [VALUES](http://localhost/phpmyadmin/url.php?url=http://dev.mysql.com/doc/refman/5.5/en/miscellaneous-functions.html#function_values) ('Tom', 'Rollins', 10049, 35000.00, 'Accounts Payable');

INSERT INTO Emp\_Table VALUES ('Bob', 'Hanover', 10050, 42000.00, 'Accounts Manager');

## SELECT STATEMENT – Put on the NOTES

The MySQL SELECT statement has a number of options:

In its simplest form, the syntax for the SELECT statement in MySQL is:

SELECT expressions

FROM tables

[WHERE conditions];

However, the full syntax for the SELECT statement in MySQL is: (Put on NOTE Pad)

SELECT [ ALL | DISTINCT | DISTINCTROW ]

[ HIGH\_PRIORITY ]

[ STRAIGHT\_JOIN ]

[ SQL\_SMALL\_RESULT | SQL\_BIG\_RESULT ] [ SQL\_BUFFER\_RESULT ]

[ SQL\_CACHE | SQL\_NO\_CACHE ]

[ SQL\_CALC\_FOUND\_ROWS ]

expressions

FROM tables

[WHERE conditions]

[GROUP BY expressions]

[HAVING condition]

[ORDER BY expression [ ASC | DESC ]]

[LIMIT [offset\_value] number\_rows | LIMIT number\_rows OFFSET offset\_value]

[PROCEDURE procedure\_name]

[INTO [ OUTFILE 'file\_name' options

| DUMPFILE 'file\_name'

| @variable1, @variable2, ... @variable\_n]

[FOR UPDATE | LOCK IN SHARE MODE];

Now, we will be looking at some examples of using the SELECT clauses, with example Databases and Tables.

**Syntax:**

**USE DATABASE\_NAME**

**SELECT [ALL] \* Column1, Column2 FROM Table1 , [Table 2 ]**

[ INTO new\_table\_name ]

[ORDER BY order\_list [ASC | DESC]];

#### Quoting Column Names and Literal Values

NOTE: When dealing with table names, strings or literal values, you need to put the correct quotes around them.

1. When accessing column information the column name needs to have **grave** accents (Above the TAB key) around it, for example:

USE EMPLOYEES

Select `Emp\_First\_Name ` from Emp\_table;

1. When inserting data into a table:

For example, as in the following INSERT INTO command, the string variables need to be enclosed with a single quote, as shown:

INSERT INTO Emp\_Table (Emp\_First\_Name, Emp\_Last\_Name, Emp\_ID) VALUES ('David', 'Jones', 10045, 45000.00, 'Manager');

The MySql SELECT statement can, firstly, be used to select ALL the data from the table Emp\_Table. To check what records have been loaded, we can run this query:

SELECT \* FROM Emp\_Table;

Also, the MySQL SELECT statement can be used to select data ONLY from particular **named columns** within a Database

**USE EMPLOYEES**

SELECT `Emp\_First\_Name`, `Emp\_Last\_Name` FROM Emp\_Table;

## Adding a SERIAL (Auto-Incrementing) Column to the Employees Table

Interest ONLY

Now we want to add a self incrementing column called **Serial\_Num** to the table,. As above, we add the column to the Emp\_Table, but we use the **serial** attribute.

ALTER TABLE Emp\_Table ADD Serial\_Num serial;

SELECT \* FROM Emp\_Table; (After the Change);

Now to test that this new column will auto-increment we can add another record, by specifying the “Serial\_Num” column with a NULL value, i.e. '':

INSERT INTO Emp\_Table VALUES ('David', 'Bowie', 10057, 7000.00, 'Performer', '');

SELECT \* FROM Emp\_Table; (Show the Updates);

## DELETE FROM `emp\_table` WHERE `serial\_num` = '5';

SELECT \* FROM Emp\_Table; (Show the Updates);

Shows the numbers are not consistent.

One way to correct this is to drop the column and re-create it:

ALTER TABLE Emp\_Table

DROP Serial\_Num;

ALTER TABLE Emp\_Table ADD Serial\_Num serial;

## Modifying Elements of a Table

The *attributes* of a column refer to the rules and behaviour of data in that column. You can modify the attributes of a column with the ALTER command, using the MODIFY clause. The word *attributes* here refers to the following:

* The *data* type of a column
* The length, precision or scale of a column
* Whether the column can contain NULL values

The following example uses the ALTER TABLE command on the Emp\_Table to modify the attributes of the column EMP\_ID:

ALTER TABLE Emp\_Table MODIFY

Emp\_Position varchar (15);

The column EMP\_ID was originally an **integer** value however, now it’s been changed to a **varchar** column with a length of 15, from the original 20.

**Note:** **The space available and the data in the column will be truncated by this command. So, you need to ensure that the data won’t be affected by decreasing the length of the field.**

Otherwise, you will need to undo the change, and rebuild the data.

## Counting the number of records in a Table

To count the number of Records in the table:

SELECT COUNT(\*) FROM Emp\_Table

AS COUNT; (Employees Database)

## Using Column Aliases

To give a particular name to a column of output is done via the use of the AS function.

SELECT COUNT(\*) FROM Emp\_Table AS Count;

## Inserting a NULL value

A NULL can be inserted in the record as long as the Column attribute is set to ALLOW NULL values. In this example we are putting a NULL value into the Emp\_Position, as it allows NULL’s.

INSERT INTO Emp\_Table (Emp\_First\_Name, Emp\_Last\_Name, Emp\_ID, Emp\_Salary, Emp\_Position, Serial\_Num) VALUES ('David', 'Tester', 10051, 37000.00, 'NULL', '');

SELECT \* FROM Emp\_Table;

## Commands to DELETE Databases, Tables, Rows and Columns

CREATE DATABASE Required;

CREATE TABLE Required\_Table

(NR\_First\_Name varchar(20) not null,

NR\_Last\_Name varchar(20) not null,

NR\_ID int not null);

First, we’ll enter a few records into the Required\_table:

INSERT INTO Required\_Table VALUES ('Bob', 'Hope', 10047);

INSERT INTO Required\_Table VALUES ('Jack', 'Jones', 10048);

[INSERT](http://localhost/phpmyadmin/url.php?url=http://dev.mysql.com/doc/refman/5.5/en/insert.html) INTO Required\_Table [VALUES](http://localhost/phpmyadmin/url.php?url=http://dev.mysql.com/doc/refman/5.5/en/miscellaneous-functions.html#function_values) ('Tom', 'Rabbit', 10049);

INSERT INTO Required\_Table VALUES ('Bob', 'Fox', 10050);

### Command to Delete a ROW

Firstly, we’ll look at how to **delete a row**, as required:

Command to DELETE a record (or row) from the DROP TABLE Required\_Table, based on the NR\_ID being equal to “10048”

DELETE FROM Required\_Table WHERE NR\_ID = 10048;

### Command to DROP a COLUMN

ALTER TABLE Required\_Table

DROP NR\_Last\_Name;

### Command to DROP the Emp\_Table Table, from the Required Database

USERequired

DROP TABLE Required\_Table;

## UPDATE COMMAND

The update command does not add new records to a table or remove them, it can only modify existing data.

* A single record field in a column, for a particular record
* All records in a column, in a single table, in a single database
* All columns in a table, at the same time

Note: Test your UPDATE and DELETE Statements

USE extreme caution when using the UPDATE statement **without** a WHERE clause. The target column is updated for ALL rows of data in the table IF conditions are not designated using the WHERE clause. In most situations, the use of the WHERE clause with a DML command is appropriate.

The UPDATE command does not add or delete data from a table it can modify the data in:

* A single record field in a column, for a particular record
* All records in a column, in a single table, in a single database
* All columns in a table, at the same time

### Updating the Value of a Single Column

1. The UPDATE following command will change the Emp\_First\_Name for the Employee\_ID record = 10046, using the SET and WHERE Clauses.

In this case, we are updating Tom Rollins to Michael Rollins, with the command:

[UPDATE](http://localhost/phpmyadmin/url.php?url=http://dev.mysql.com/doc/refman/5.5/en/update.html) Emp\_Table [SET](http://localhost/phpmyadmin/url.php?url=http://dev.mysql.com/doc/refman/5.5/en/set.html) Emp\_First\_Name  = 'Michael' WHERE Emp\_ID = '10046';

1. Now we want to give everyone a pay rise following UPDATE command allows for all salaries to be updated by a set amount.

UPDATE Emp\_Table SET Emp\_Salary = Emp\_Salary + (Emp\_Salary \* .03);

The UPDATE command can be used to modify individual pieces of data, such as, increasing the “salary amount”, in the salary column, by 3%. This is done by taking the original salary amount and then adding the calculated amount of the original salary multiplied by 0.03 or 3 percent, back to it. Once this operation is completed, the larger amount is then placed back into the field, within the table.

1. We can update all the columns in a row for one employee number numbered 10046, at the same time:

[UPDATE](http://localhost/phpmyadmin/url.php?url=http://dev.mysql.com/doc/refman/5.5/en/update.html) Emp\_Table [SET](http://localhost/phpmyadmin/url.php?url=http://dev.mysql.com/doc/refman/5.5/en/set.html) Emp\_First\_Name  = 'James', Emp\_Last\_Name  = 'Kirk', Emp\_ID = 17010, Emp\_Salary = 80000.00, Emp\_Position = 'Star Ship Captain' WHERE Emp\_ID = 10046;

First, we’ll enter a few records into the Required\_Table:

INSERT INTO Required\_Table VALUES ('Bob', 'Hope', 10047);

INSERT INTO Required\_Table VALUES ('Jack', 'Jones', 10048);

[INSERT](http://localhost/phpmyadmin/url.php?url=http://dev.mysql.com/doc/refman/5.5/en/insert.html) INTO Required\_Table [VALUES](http://localhost/phpmyadmin/url.php?url=http://dev.mysql.com/doc/refman/5.5/en/miscellaneous-functions.html#function_values) ('Tom', 'Book', 10049);

INSERT INTO Required\_Table VALUES ('Bob', 'Fox', 10050);

## How to Display the Columns and Attributes of a Table, within a Database

When working with the SHOW command, the Database doesn’t need to be selected within the SQL TAB, for the command to work, as shown:

Syntax:

SHOW [FULL] COLUMNS {FROM | IN} ***tbl\_name*** [{FROM | IN} ***db\_name***]

[LIKE '***pattern***' | WHERE ***expr***]

SHOW COLUMNS FROM Agents FROM Realestate;

AgentNo int(11) NO MUL auto\_increment

AgentFirst varchar(50) YES

AgentLast varchar(50) YES

AgentPhone varchar(50) YES

AgencyNo int(11) YES 0

SHOW COLUMNS FROM Emp\_Table FROM Employees;

### How to copy one Table into another Table name, from with a Database

The process for making a clone copy of a table is possible in two ways by copying the full contents or you can copy over just the columns you want.

Format for copying the entire Agents table to Agents\_Copy table

CREATE TABLE (AgentNo int, AgentFirst varchar(50), AgentLast varchar(50),

AgentPhone varchar(50), AgencyNo int);

CREATE TABLE Agents\_Copy

SELECT ALL INTO Agents\_Full FROM Agents;

SELECT AgentNo, AgentFirst, AgentLast INTO Agents\_Partial FROM Agents

## SELECT INDIVIDUAL FIELDS FROM ONE TABLE

Let's look at how to use a MySQL SELECT query to select all the fields from a table.

SELECT \* FROM AGENTS

WHERE AgentNo <= 6

ORDER BY AgencyNo DESC;

In this MySQL SELECT statement example, we've used \* to signify that we wish to select all fields from the Agents table, within the Realestate database where the AgentNo is less than or equal to 6. The result set is sorted by AgencyNo, in descending order.

## EXAMPLE - SELECT INDIVIDUAL FIELDS FROM ONE TABLE

You can also use the MySQL SELECT statement to select individual fields from the table, as opposed to all fields from the table, and based on certain conditions:

For example:

SELECT AgentNo, Area, Asking

FROM listings

WHERE Asking <= 147900.00

ORDER BY Asking ASC;

This MySQL SELECT example would return only the order\_id, quantity, and unit\_price fields from the order\_details table where the quantity is less than 500. The results are sorted by quantity in ascending order and then unit\_price in descending order.

REFERENCES:

<http://www.techonthenet.com/mysql/index.php>

Creating a table from an existing table

### Quick Fix – Rebuild of Emp\_Table and Records

CREATE TABLE Emp\_Table

(Emp\_First\_Name varchar(20) not null,

Emp\_Last\_Name varchar(20) not null,

Emp\_ID int not null);

ALTER TABLE Emp\_Table ADD Emp\_Salary decimal(10,2) NULL;

ALTER TABLE Emp\_Table ADD Emp\_Position varchar(20) null;

INSERT INTO Emp\_Table (Emp\_First\_Name , Emp\_Last\_Name , Emp\_ID, Emp\_Salary, Emp\_Position)

VALUES ('David', 'Jones', 10045, 45000.00, 'Manager');

INSERT INTO Emp\_Table (Emp\_First\_Name , Emp\_Last\_Name , Emp\_ID, Emp\_Salary, Emp\_Position) VALUES ('Tom', 'Rollins', 10046, 50000.00, 'Programmer');

INSERT INTO Emp\_Table (Emp\_First\_Name , Emp\_Last\_Name , Emp\_ID, Emp\_Salary, Emp\_Position) VALUES ('Bob', 'Hanover', 10047, 37000.00, 'Printer');

INSERT INTO Emp\_Table VALUES ('David', 'Bowie', 10051, 45200.00,'Entertainer'), ('James', 'Bond', 10052, 55000.00,'Spy Master'), ('Homer', 'Simpson', 10053, 30000.00,'Safety Manager');

INSERT INTO Emp\_Table (Emp\_First\_Name , Emp\_Last\_Name , Emp\_ID, Emp\_Salary, Emp\_Position) VALUES ('Brian', 'Harris', 10054, 44000.00, 'Truck Driver'),('Harry', 'Smith', 10055, 50000.00, 'Printer'), ('Richard', 'Hanson', 10056, 60000.00, 'Senior Manager');

INSERT INTO Emp\_Table VALUES ('Alan', 'Jones', 10048, 36500.00, 'Office Manager');

INSERT INTO Emp\_Table VALUES ('Tom', 'Rollins', 10049, 35000.00, 'Accounts Payable');

INSERT INTO Emp\_Table VALUES ('Bob', 'Hanover', 10050, 42000.00, 'Accounts Manager');

ALTER TABLE Emp\_Table ADD Serial\_Num serial;