Tutorial 1 Summary

Following I have summed up the activities and commands discussed in the first tutorial session.

Some Basic SQL syntax

Creating Database, Tables:

To start working with tabular data you need a table, and to make a table you need a database where you can make one. So start by creating a database.

```
CREATE DATABASE Example;
```

Once we have our database, we need our SQL server to know that we now want to work inside this 'Example' database. So we do that.

```
USE DATABASE Example;
```

We are into our database, so now let's go ahead and create a table with a few columns. Each column needs a column name and a datatype of the values that will be entered in the column.

```
CREATE TABLE Students(roll_num INT,
                      s_name VARCHAR,
                      grade VARCHAR(2),
                      p_money INT);
```

(Note the use of '(2)' beside the 'grade' column denoting that it can take at most 2 characters. This is saving us storage memory)

Populating a Table

To add data to your table, i.e. to add a row to your table, you need to 'INSERT' a n-tuple into the specified table, where n is the number of columns and the tuple values are the entry for each column.

```
INSERT INTO Students VALUES('001', 'Prasun', 'A', '5000');
INSERT INTO Students VALUES(002, 'Megha', 'AB', 4500);
INSERT INTO Students VALUES(3, 'Shiuli', 'AB', 3000);
```

(Note the use/not use of the quotes for integers. MySQL can work with both.)

Here you can have a lot of variation while populating the table. You may specifiy which columns you want to input values into, or specify some null values.

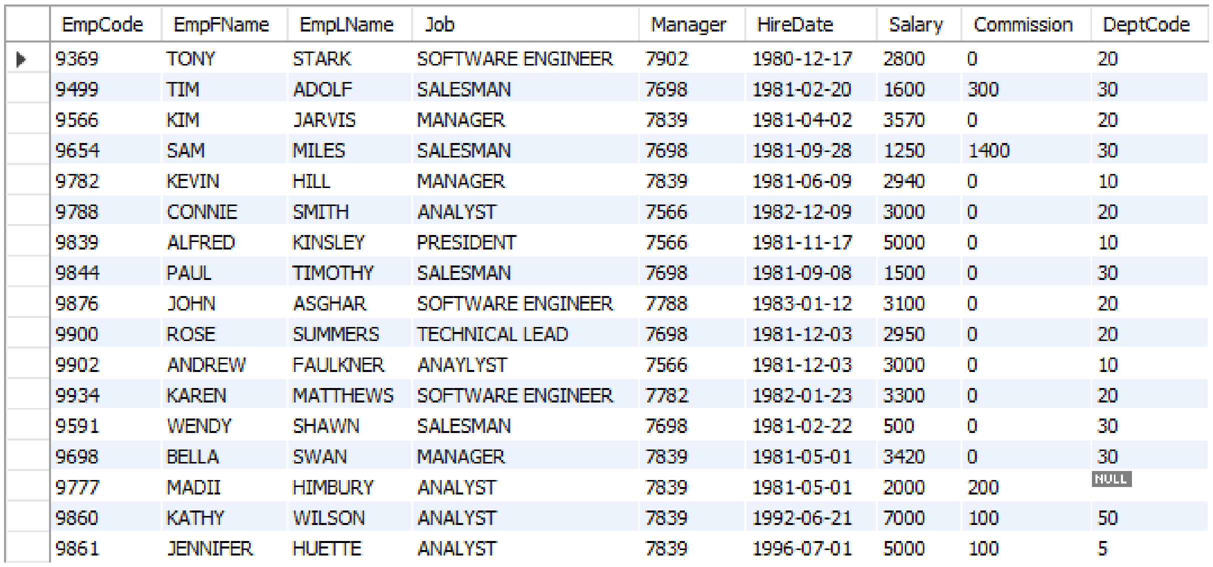
Viewing data from a Table

To view the entire contents of a table you may use the '*' wildcard key in SQL, as such.

SELECT * FROM Students;

The 'SELECT' keyword works like a print statement in SQL and used whenever we need to display something (or make a selection). We may choose to display partial information too. For example, to generate just the Names and Pocket Money of the students you can use,

SELECT s_name, p_money FROM Students



EmpCode	EmpFName	EmpLName	Job	Manager	HireDate	Salary	Commission	DeptCode
9369	TONY	STARK	SOFTWARE ENGINEER	7902	1980-12-17	2800	0	20
9499	TIM	ADOLF	SALESMAN	7698	1981-02-20	1600	300	30
9566	KIM	JARVIS	MANAGER	7839	1981-04-02	3570	0	20
9654	SAM	MILES	SALESMAN	7698	1981-09-28	1250	1400	30
9782	KEVIN	HILL	MANAGER	7839	1981-06-09	2940	0	10
9788	CONNIE	SMITH	ANALYST	7566	1982-12-09	3000	0	20
9839	ALFRED	KINSLEY	PRESIDENT	7566	1981-11-17	5000	0	10
9844	PAUL	TIMOTHY	SALESMAN	7698	1981-09-08	1500	0	30
9876	JOHN	ASGHAR	SOFTWARE ENGINEER	7788	1983-01-12	3100	0	20
9900	ROSE	SUMMERS	TECHNICAL LEAD	7698	1981-12-03	2950	0	20
9902	ANDREW	FAULKNER	ANAYLYST	7566	1981-12-03	3000	0	10
9934	KAREN	MATTHEWS	SOFTWARE ENGINEER	7782	1982-01-23	3300	0	20
9591	WENDY	SHAWN	SALESMAN	7698	1981-02-22	500	0	30
9698	BELLA	SWAN	MANAGER	7839	1981-05-01	3420	0	30
9777	MADII	HIMBURY	ANALYST	7839	1981-05-01	2000	200	NULL
9860	KATHY	WILSON	ANALYST	7839	1992-06-21	7000	100	50
9861	JENNIFER	HUETTE	ANALYST	7839	1996-07-01	5000	100	5

Figure 1: The employee table used as the example for querying commands

Querying a Table

Once we have our data fed to MySQL in tabular form we can start querying it now.

- All information about employees who have a salary higher than 3000.

  SELECT * FROM employee WHERE Salary > 3000;

- Or maybe just the first names of employees who have the salary higher than 3000.

  SELECT EmpFName FROM employee WHERE Salary > 3000;

- Maybe you want to find out the full names of all those Analysts that have earned some commission. We can use the CONCAT keyword to join two strings from different columns and display it under a single column, like 'Name', using the AS keyword.

  SELECT CONCAT(EmpFName, ' ', EmpLName) AS 'Name' FROM employee WHERE Job = 'Analyst' AND Commission > 0;

- Mostly all boolean keywords are SQL keywords too, like AND, OR, NOT etc. You may use these keywords in conjuction with keywords such as IN to refine queries. The queries have similar semantic implications as any query in the English language.

  SELECT * FROM employee WHERE Commission > 0 AND Job != 'Analyst';

  OR

  SELECT * FROM employee WHERE Commission > 0 AND Job NOT IN ('Analyst', 'Manager');

(Note the use of the 'Analyst' in the queries instead of 'ANALYST' as present in the table. MySQL is case-agnostic and treats every query as such, unless specifically specified. You may query case-sensitive query by using the BINARY keyword before the query term)

Modifying data in a Table

Once you understand how you may access a particular row by the using the WHERE keyword you can now update or modify the table as per your wish. Lets say, you want to assign a commission of 1000 to 'KIM JARVIS', who is a 'Manager'.

UPDATE employee SET Commission = 1000 WHERE EmpFName = 'KIM' AND EmpLName = 'JARVIS';

Using the 'UPDATE' and 'SET' keywords we have modified the entry of the table.

Similarly you can DELETE an entry too.

DELETE FROM employee WHERE Commission = 200;

Deleting a Table or Database

You may also DROP(delete) an entire table from your database or the whole database.

DROP TABLE employee;
DROP DATABASE Example;