

MySQL

Lecture 1

by
T.K.Malwatta
Senior Lecturer/HOD –ICT
University of Vocational Technology

What's a database ?

A database is a collection of data organized in a particular way.

Databases can be of many types such as Flat File Databases, Relational Databases, Distributed Databases etc.

So now a days we use relational database management systems (RDBMS) to store and manager huge volume of data. This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as foreign keys.

A Relational Database Management System facilitates the organization, storage, access, security, and integrity of data and eliminates data redundancy. It stores the data in a set of tables each of which contains a unique identifier. Typical examples of RDBMS software include Oracle, Microsoft SQL Server, Sybase, PostgreSQL, MySQL, etc.

RDBMS Terminology:

Before we proceed to explain MySQL database system, let's revise few definitions related to database.

- ✓ Database: A database is a collection of tables, with related data.
- ✓ Table (Entity) : A table is a matrix with data. A table in a database looks like a simple spreadsheet.
- ✓ Column (Attribute) : One column (data element) contains data of one and the same kind, for example the column postcode.
- ✓ Row: A row (tuple, entry or record) is a group of related data, for example the data of one subscription.
- ✓ Redundancy: Storing data twice, redundantly to make the system faster.
- ✓ Primary Key: A primary key is unique. A key value can not occur twice in one table. With a key you can find at most one row.
- ✓ Foreign Key: A foreign key is the linking pin between two tables.
- ✓ Compound Key: A compound key (composite key) is a key that consists of multiple columns, because one column is not sufficiently unique.
- ✓ Referential Integrity: Referential Integrity makes sure that a foreign key value always points to an existing row.

MySQL is a fast, easy-to-use RDBMS used being used for many small and big businesses. MySQL is developed, marketed, and supported by MySQL AB, which is a Swedish company. MySQL is becoming so popular because of many good reasons.

- ✓MySQL is released under an open-source license. So you have nothing to pay to use it.
- ✓MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
- ✓MySQL uses a standard form of the well-known SQL data language.
- ✓MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA etc.
- ✓MySQL works very quickly and works well even with large data sets.
- ✓MySQL is very friendly to PHP, the most appreciated language for web development.
- ✓MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).

MySQL is the most popular Open Source Relational SQL database management system.

Connecting to and Disconnecting from the Server

To connect to the server, you will usually need to provide a MySQL user name when you invoke `mysql` and, most likely, a password. If the server runs on a machine other than the one where you log in, you will also need to specify a host name. Once you know the proper parameters, you should be able to connect like this:

```
mysql -h host -u user -p  
Enter password: *****
```

host and user represent the host name where your MySQL server is running and the user name of your MySQL account. Substitute appropriate values for your setup. The `*****` represents your password; enter it when `mysql` displays the Enter password: prompt.

If that works, you should see some introductory information followed by a `mysql>` prompt:

```
Welcome to the MySQL monitor. Commands end with ; or \g.  
Your MySQL connection id is 4 to server version: 6 .0.4-standard  
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.  
mysql>
```

The `mysql>` prompt tells you that mysql is ready for you to enter commands.

After you have connected successfully, you can disconnect any time by typing QUIT

mysql> QUIT

Entering Queries

Make sure that you are connected to the server. Here is a simple command that asks the server to tell you its version number and the current date. Type it in as shown here following the `mysql>` prompt and press Enter:

mysql> select version(), current_date;

It demonstrates that you can use mysql as a **simple calculator**

mysql> select (4+1)*5, 50-37, 70/7, 56*2;

```
mysql> select (4+1)*5, 50-37, 70/7, 56*2;
+-----+-----+-----+-----+
| (4+1)*5 | 50-37 | 70/7 | 56*2 |
+-----+-----+-----+-----+
|      25 |     13 | 10.0000 |    112 |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> 
```

You can even enter multiple statements on a single line.
Just end each one with a semicolon:

```
mysql> select version(); select now();
```

```
mysql> select  
-> user()  
-> ,  
-> current _date;
```

The following table shows each of the prompts you may see and summarizes what they mean about the state that mysql is in.

Prompt	Meaning
mysql>	Ready for new command.
->	Waiting for next line of multiple-line command.
'>	Waiting for next line, waiting for completion of a string that began with a single quote ("").
">	Waiting for next line, waiting for completion of a string that began with a double quote ("").
`>	Waiting for next line, waiting for completion of an identifier that began with a backtick ("").
/*>	Waiting for next line, waiting for completion of a comment that began with /*.
	Multiple-

Working with MySQL Database

MySQL Creating and Deleting Database

- ✓ Creation and Selection of database
- ✓ Creation of table
- ✓ Load the data into the table

Use the SHOW statement to find out what databases currently exist on the server:

```
mysql> show databases;
```

Creating And Selecting a Database

```
mysql> create database search;
```

Creating a database does not mean that it is select for use, you have to select it explicitly by the USE command.

```
mysql> use search;
```

Creating table

```
mysql> show tables;
```


Emp

fname	lname	city	sex	bod

```
mysql>create table Emp (fname VARCHAR(20), lname VARCHAR(20), city  
VARCHAR(20), sex CHAR(1), bod DATE);
```

MySQL uses many different [data types](#), broken into three categories: numeric, date and time, and string types.

If you want to show the structure list of your table then use DESCRIBE statement. like :

```
mysql> describe Emp;
```

```
mysql> describe Emp;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| fname | varchar(20)   | YES  |     | NULL    |       |
| lname | varchar(20)   | YES  |     | NULL    |       |
| city  | varchar(10)   | YES  |     | NULL    |       |
| sex   | char(1)       | YES  |     | NULL    |       |
| dob   | date          | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.05 sec)

mysql>
```

Load the data into the table

After creating the table you need to load the data in this table, you can do this by using the INSERT statement. Following example is helps you to teach the INSERT statement.

```
mysql> insert into Emp values( 'Nimal','Silva',Ratnapura, 'M', '1980-03-16');
```

After inserting the data into the table, now we discuss about retrieving the data from the table. For retrieving the data we used the SELECT statement. The general syntax of SELECT statement is :

Retrieving Information from a Table

The SELECT statement is used to pull information from a table. The general form of the statement is:

*SELECT what_to_select
FROM which_table
WHERE conditions_to_satisfy;*

```
mysql> select * from Emp;
```

```
mysql> select fname,lname FROM Emp;
```

```
mysql> select * FROM Emp WHERE fname='Amar';
```

If you need to delete the database then you have to use the DROP statement. Example

```
mysql> drop database search;
```

Select Particular row

You can select only particular rows from your table. For example, if you want to verify the change that you made to employers City, select his record like this:

```
mysql > select * from Emp where fname='Kamal';
```

fname	lname	city	sex	DOB
Kamal	Silva	Galle	F	1978-04-04

Update a record

```
Update Emp set city='Dehiwala' where fname='Kamal';
```

fname	lname	city	sex	DOB
Kamal	Silva	Dehiwala	F	1978-04-04

```
mysql > select * from Emp where DOB>'1980-01-01';
```

You can combine conditions with 'and' , for example,

```
mysql >select * from Emp where DOB>'1980-01-01' and sex='M';
```

The preceding query uses the AND logical operator. There is also an OR,NOT operator:

```
mysql>select * from Emp where lname='Perera' or lname='Silva';
```

Selecting Particular Columns

If you do not want to see entire rows from your table, just name the columns in which you are interested, separated by commas. For example,

```
mysql>select fname, DOB from Emp;
```

The query simply retrieves the fname column from each record, and some of them appear more than once. To minimize the output, retrieve each unique output record just once by adding the keyword DISTINCT:

```
mysql>select distinct fname from Emp;
```

You can use a WHERE clause to combine row selection with column selection

```
mysql>select fname,city from Emp where city='Colombo' or city='Galle';
```

Sorting Rows

You may have noticed in the preceding examples that the result rows are displayed in no particular order. It is often easier to examine query output when the rows are sorted in some meaningful way. To sort a result, use an **ORDER BY** clause.

Add columns to existing table

```
Mysql > alter table emp add email varchar(20); // add the column to be last  
Mysql> alter table emp add age int(10) after city; //add column after city column  
Mysql> alter table emp add age int(10) first; //add column to be first
```

Delete columns from existing table

```
Mysql> alter table emp drop email;
```

Delete row from existing table

```
Mysql > delete from emp where fname='ajith';
```

MySQL provides several functions that you can use to perform calculations on dates, for example, to calculate ages or extract parts of dates.

```
> select fname,lname ,curdate(),  
-> (YEAR (curdate()) – YEAR(dob))  
->As age  
->From emp;
```

simply want to extract the month part of the birth column. MySQL provides several functions for extracting YEAR(), MONTH(), and DAYOFMONTH().

```
Mysql >select fname,dob,MONTH(dob) from emp;
```

Finding employees birthdays in the particular month(Eg. October)

```
Mysql>select fname BOD from Emp where MONTH(DOB)=10;
```