Beginning with MySQL

Youtube: Javademystified by SSR

Sadhu Sreenivas

Agenda

- Introduction & features of MySQL
- MySQL and SQL
- SQL Commands- DDL, DML, TCL & DCL
- Data types in MySQL
- Creating Database & Tables
- Inserting, Deleting and modifying records
- Making Simple Queries
- Altering Table Structure and Constraints

Introduction to MySQL

MySQL is an Open Source, Fast and Reliable Relational Database Management System (RDBMS) software like Oracle, Sybase, MS SQL Server etc. It was developed by Michael Widenius and AKA Monty and is alternative to many of the commercial RDBMS.

The main features of MySQL are-

- **Open Source & Free of Cost:**

  It is Open Source and available at free of cost.
- **Portability:**

  It can be installed and run on any types of Hardware and OS like Linux, MS Windows or Mac etc.
- **Security :**

  It creates secured database protected with password.
- **Connectivity**

  It may connect various types of Network client using different protocols and Programming Languages .
- **Query Language**

  It uses SQL (Structured Query Language) for handling database.

MySQL & SQL

In order to access data from the MySQL database, all program and user must use SQL (Structured Query Language). SQL is a set of commands that are recognized by all the RDBMSs and has become a standard language for database handling.

SQL is a language that enables you to create and manage a relational database, in which all the information are kept in tables.

There are numerous version of SQL. The original version was developed at IBM’s San Jose Research Laboratory with a name of Sequel, as a part of System R project in 1970s. It was standardized by ANSI in 1986 by the name of SQL.

SQL is a Standard Query language whereas MySQL is a RDBMS Software based on SQL.

Types of SQL Commands

MySQL follows SQL specifications for its commands . These SQL commands can be categorized as -

- **Data Definition Language (DDL)**

  These SQL commands are used to create, alter and delete database objects like database, table, views, index etc.

  Example: CREATE , ALTER , DROP etc.; works on schema

- **Data Manipulation Language (DML)**

  These commands are used to insert, delete, update and retrieve the stored records from the table; works on data instance

  INSERT DELETE,UPDATE… and the SELECT (Query Command)

- **Transaction Control Language (TCL)**

  These commands are used for the transaction

  management. Ex. COMMIT, ROLLBACK, SAVEPOINT

- **Data Control Language (DCL)**

  These commands are used to manipulate permissions or access rights to the tables etc.

  Ex. GRANT , REVOKE etc.

Data types in MySQL

- **Numeric Data Types:**
  - **INTEGER or INT** – up to 11 digit number without decimal.
  - **SMALLINT** – up to 5 digit number without decimal.
  - **FLOAT (M,D) or DECIMAL(M,D) or NUMERIC(M,D)**

    Stores Real numbers upto **M** digit length (including .) with **D** decimal places.

    e.g. Float (10,2) can store 1234567.89
- **Date & Time Data Types:**
  - **DATE** - Stores date in YYYY-MM-DD format.
  - **TIME** - Stores time in HH:MM:SS format.
- **String or Text Data Type:**
  - **CHAR(Size)**

    A fixed length string up to 255 characters. (default is 1)
  - **VARCHAR(Size)**

    A variable length string up to 255 characters.

Char, **Varchar**, **Date** and **Time** values should be enclosed with single (' ') or double ("") quotes in MySQL.

Database Handling commands in MySQL

- **Creating a Database.**
  The following command will create School database in MySQL.
  ```
  mysql> CREATE DATABASE School;
  ```
- **Opening a database**
  To open an existing database, following command is used.
  ```
  mysql> USE school ;
  ```
- **Getting listings of database and tables**
  ```
  mysql> SHOW DATABASES;
  mysql> SHOW TABLES;
  ```
- **Deleting a Database and Table**
  ```
  mysql> DROP DATABASE School;
  mysql> DROP TABLE Student;
  ```
- **Viewing Table Structure**
  ```
  mysql> DESCRIBE Student;
  ```

```
Select database();
```
Shows the name of currently open database

Creating Tables & Inserting records

- **Creating Simple Tables:**

  CREATE TABLE < Table Name>
  (<Col name1><data type>[(size)],….);

  Data types- INTEGER, NUMERIC(P,D), CHAR(n), VARCHAR(n), DATE etc.

```
mysql> CREATE TABLE Employee
      (empID integer,
       ename char(30),
       city char(25),
       pay decimal(10,2));
```

Employee			
empID	ename	city	pay

- **Inserting Records:**

  INSERT INTO <Table Name> VALUES (value1, vale2, ……);

  String and Date type values must be enclosed in single or double quotes.

```
mysql> INSERT INTO Employee VALUES (1,‘Amitabh’,‘Allahabad’,15000);
mysql> INSERT INTO Employee VALUES (2, ‘Akbar’, ‘Dehradun’,20000);
mysql> INSERT INTO Employee VALUES (3, ‘Anthony’, ‘Mumbai’,10500);
```

Making Simple Queries Using SELECT

The SELECT command of SQL, empower you to make a request (queries) to retrieve stored records from the database.

The syntax of SQL is given below-

SELECT < [Distinct | ALL] *| column name(s)>

FROM <table(s)>

WHERE <condition>

ORDER BY <column name> [ASC | DESC] ;

Consider the table ***Student*** having some records as –

StID	Name	Fname	DOB	City	Class
S1	Amitabh	Harivansh Rai	1948-11-10	Allahabad	12
S2	Sharukh	Firoz	1970-05-10	Delhi	11
S3	Irphan	Akbar	1970-10-05	Jaipur	11
S4	Salman	Salim Javed	1972-04-10	Mumbai	10
S5	Abhishek	Amitabh	1975-03-12	Mumbai	10

Making Simple Queries – Cont..

- **Selecting all columns**

If you want to view all columns of the student table, then you should give the following command-

```
mysql> SELECT * FROM Student ;
```

MySQL will display the all records with all columns in the Student table.

* Is used to represent all columns.

StID	Name	Fname	DOB	City	Class
S1	Amitabh	Harivansh Rai	1948-11-10	Allahabad	12
S2	Sharukh	Firoz	1970-05-10	Delhi	11
S3	Irphan	Akbar	1970-10-05	Jaipur	11
S4	Salman	Salim Javed	1972-04-10	Mumbai	10
S5	Abhishek	Amitabh	1975-03-12	Mumbai	10

Making Simple Queries – Cont..

- **Selecting columns**

If you want to view only **Name** and **City** columns of the student table

mysql> **SELECT Name, City FROM Student ;**

Name	City
Amitabh	Allahabad
Sharukh	Delhi
Irphan	Jaipur
Salman	Mumbai
Abhishek	Mumbai

mysql> **SELECT City, Name FROM Student ;**

City	Name
Allahabad	Amitabh
Delhi	Sharukh
Jaipur	Irphan
Mumbai	Salman
Mumbai	Abhishek

Making Simple Queries – Cont..

- **Eliminating Duplicate values in a column - DISTINCT**

mysql> **SELECT City FROM Student ;**

City
Allahabad
Delhi
Jaipur
Mumbai
Mumbai

Mumbai is repeated

MySQL assumes ALL keyword, if you are not using DISTICT keyword.

mysql> **SELECT DISTINCT City FROM Student ;**

City
Allahabad
Delhi
Jaipur
Mumbai

Only Unique Cities are displayed

Making Simple Queries – Cont..

□ Doing simple calculations

We can also perform simple calculations with SQL Select command. SQL provide a dummy table named DUAL, which can be used for this purpose.

mysql> **SELECT 4*3 ;**

We can also extend this idea with a columns of the existing table.

mysql> **SELECT Name, Sal *12 FROM EMP ;**

□ Using Column Aliases

We can give a different name to a column or expression (Alias) in the output of a query.

Alias for Sal*12

mysql> **SELECT Name, Sal*12 AS 'Annual Salary' FROM EMP;**

mysql> **SELECT Name, DOB AS 'Date of Birth' FROM Student;**

mysql> **SELECT 22/7 AS PI FROM Dual;**

When Alias name is a single word then single quotes is not required.

Selecting Specific Records – WHERE clause

- **WHERE <Condition>**

  We can select specific records by specifying conditions with WHERE clause.

```
mysql> SELECT * FROM Student WHERE City=‘Mumbai’;
```

StID	Name	Fname	DOB	City	Class
S4	Salman	Salim Javed	1972-04-10	Mumbai	10
S5	Abhishek	Amitabh	1975-03-12	Mumbai	10

```
mysql> SELECT Name, Fname, City from Student
       WHERE  Class >10;
```

Condition

Name	Fname	City	Class
Amitabh	Harivansh Rai	Allahabad	12
Sharukh	Firoz	Delhi	11
Irphan	Akbar	Jaipur	11

Selecting Specific Records – WHERE clause

- **Relational Operators**

  We can use the following Relational operators in condition.

  =, > , < , >=, <=, <>, IS , LIKE, IN, BETWEEN

- **Logical Operators**

  We can use the following Logical Operators to connect two conditions.

  OR , AND , NOT (!)

```
mysql> SELECT Name, City from Student
       WHERE City <> 'Mumbai' AND Class>10;

mysql> SELECT * FROM Emp
       WHERE Sal >10000 OR Job ='Manager';

mysql> SELECT * FROM Student
       WHERE NOT Grade='A';
```

Selecting Specific Rows – WHERE clause

- **Specifying Range of Values – BETWEEN Operator**

```
mysql> SELECT * FROM Emp
       WHERE Sal BETWEEN 5000 AND 10000 ;
```

The same query can also be written as -

```
mysql> SELECT * FROM Emp
       WHERE Sal >= 5000 AND Sal<=10000 ;
```

Other Logical operators also can be applied-

```
mysql> SELECT * FROM Emp
        WHERE  NOT Sal BETWEEN 5000 AND 10000 ;
```

- **Specifying List – IN Operator**

```
mysql> SELECT * FROM Emp
       WHERE Sal IN (5000, 10000) ;
```

The same query can also be written as -

```
mysql> SELECT * FROM Emp
       WHERE Sal = 5000 OR Sal =10000 ;

mysql> SELECT * FROM Student
        WHERE  City IN ('Mumbai', 'Delhi','Kanpur') ;
```

Selecting Specific Rows – WHERE clause

□ Pattern Matching – LIKE Operator

A string pattern can be used in SQL using the following wild card

- ❖ **%** **Represents a substring in any length**
- ❖ **_** **Represents a single character**

Example:

'A%'	represents any string starting with 'A' character.
'_ _A'	represents any 3 character string ending with 'A'.
'_B%'	represents any string having second character 'B'
'_ _ _'	represents any 3 letter string.

A pattern is case sensitive and can be used with LIKE operator.

```
mysql> SELECT * FROM Student WHERE Name LIKE 'A%';
mysql> SELECT * FROM Student WHERE Name LIKE '%Singh%';
mysql> SELECT Name, City FROM Student
           WHERE  Class>=9 AND Name LIKE '%Kumar%' ;
```

Selecting Specific Rows – WHERE clause

- **Searching NULL Values – IS Operator**

```
mysql> SELECT * FROM Student WHERE City IS NULL ;
```

The **NOT** Operator can also be applied -

```
mysql> SELECT * FROM Student WHERE City IS NOT NULL;
```

- **Ordering Query Result – ORDER BY Clause**

A query result can be orders in ascending (A-Z) or descending (Z-A) order as per any column. Default is Ascending order.

```
mysql> SELECT * FROM Student ORDER BY City;
```

To get descending order use **DESC** key word.

```
mysql> SELECT * FROM Student ORDER BY City DESC;

mysql> SELECT Name, Fname, City FROM Student
             Where Name LIKE ‘R%’  ORDER BY Class;

mysql> SELECT Name, Basic+DA AS ‘PAY’ FROM Student ORDER BY PAY;
```

You can also use Alias column with ORDER BY clause

Inserting Records in a Table

You can insert record in the table by using by using the following DML command.

INSERT INTO <Table Name> [<Column list>]

VALUES <list of values>

If value is not available for a column, NULL can be used.

Suppose a table <u>STUDENT</u> has been created as per given structure-

StID	NAME	FNAME	DOB	CITY	CLASS

The sequence of values should match with order of columns.

We can insert a record as follows-

```
mysql> INSERT INTO Student VALUES
       ('s1','Amitabh', 'Harivansh','1955-10-25', 'Mumbai', 12);
 mysql> INSERT INTO Student VALUES
       ('s2','Sharukh Khan', NULL,'1972-5-25', 'Delhi', 10);
mysql> INSERT INTO Student (StID, FName, Name, Class)
       VALUES ('s3','Amitabh', 'Abhishek', 10);
```

You can also define order of columns.

Inserting Records from Other Table

You can insert all or selected record(s) in the table from another table by using Select … command in place of Values.

Suppose a table named NEWSTUDENT has been created and records to be inserted from OLDSTUDENT table having the same structure of columns.

```
mysql> INSERT INTO Newstudent VALUES
       (SELECET * FROM Oldstudent);
```

Both tables must have same column structure

```
 mysql>INSERT INTO Newstudent VALUES
       (SELECT * FROM Oldstudent WHERE City=‘Mumbai’);
mysql> INSERT INTO Newstudent (StID, Name, Class)
       VALUES (Select StID, Name,Class FROM Oldstudent
       WHERE Class>=11);
```

You can also select columns from both tables.

Deleting Records from the Table

You can delete all or selected record(s) from the table by using the following DML command.

DELETE FROM <Table Name> [WHERE <Condition>]

```
mysql> DELETE FROM Student ;
```

Caution!!! This command will delete all records…

```
mysql> DELETE FROM Student WHERE  City='Mumbai' ;
mysql> DELETE FROM Student WHERE Class >=11 ;
mysql> DELETE FROM Student WHERE Class <9 AND City='Delhi';
```

- **You can recall (Undelete) records by giving ROLLBACK command.**
  ```
  mysql> ROLLBACK ;
  ```
- **You can issue COMMIT command to record the changes permanently.**
  ```
  mysql> COMMIT;
  ```

Modifying Records –UPDATE Command

You can modify the values of columns of all or selected records in the table by using the following DML command.

UPDATE <Table Name>
SET <Column> = <Expression>
[WHERE <Condition>]

```
mysql> UPDATE Student SET Class =10  ;

mysql> UPDATE Student SET FName= CONACT('Mr.', FName') ;

mysql> UPDATE Emp SET Sal = Sal+(Sal*10/100);

mysql> UPDATE Emp SET Sal = Sal+(Sal*10/100)
       WHERE Sal <=10000;
mysql> UPDATE Emp SET City = 'Dehradun'
       WHERE CITY IS NULL;
```

Working with Tables

- **Creating Tables:**

  CREATE TABLE < Table Name>

  (<Col name><data type>[(size)][Constraints], …..)

- **Data types -** Commonly used data types are-

  INTEGER, DECIMAL(P,D), NUMERIC(P,D), CHAR(n), VARCHAR(n), DATE etc.

Employee (EmpID, Ename, Sex, DOB, Pay)

```
mysql> CREATE TABLE Employee
     (EmpID integer,
      Ename char(20),
      Sex  char(1),
      Dob  Date,
      Pay decimal (8,2));
```

Creating Table with Constraints

One of the major responsibility of a DBMS is to maintain the Integrity of the data i.e. Data being stored in the Database must be correct and valid.

An Integrity Constraints are condition or checks applicable to a column or table which ensures the integrity and validity of data.

The following constraints are available in MySQL.

Constraints	Description
NOT NULL	Ensures that a column cannot have NULL value.
PRIMARY KEY	Used to identify a row uniquely.
DEFAULT*	Provides a default value for a column, if no value is given.
UNIQUE*	Ensures that all values in a column are different.
CHECK*	Ensures that value for a column should satisfy certain condition.
FOREIGN KEY*	Used to ensure Referential Integrity of the data.

* Not included in the syllabus (recommended for advanced learning)

Implementing Constraints in the Table

- **NOT NULL**

  This constraints specifies that column must not contain NULL value i.e. value for the column must be given (**mandatory**)
- **PRIMARY KEY**

  This constraints declare a column as the primary key. Since **Primary key must not have NULL value**, so it is used with NOT NULL constraints.
- **UNIQUE**

  This constraints ensures that the value for the column should be Unique i.e. **no two records have the same** (duplicate) value.

```
mysql> CREATE TABLE Student
      (StCode char(3) NOT NULL PRIMARY KEY,
       Stname char(20)  NOT NULL,
       StAdd  varchar(40),
       AdmNo  char(5) UNIQUE,
       StAge  integer CHECK (StAge>=5) );
```

Generally Constraints are defined with Column definitions i.e. Column level

UNIQUE v/s PRIMARY KEY

- ❖ UNIQUE allows NULL values but PRIMERY KEY does not.
- ❖ A table may have multiple UNIQUE constraints, but there must be only one PRIMERY KEY constraints in a table.

Implementing Primary Key Constraints

❖ Defining Primary Key at Column Level:

```
mysql> CREATE TABLE Student
       ( StCode    char(3)    NOT NULL PRIMARY KEY,
         Stname   char(20)    NOT NULL,
          StAge   int(2) );
```

❖ Defining Primary Key at Table Level:

```
mysql> CREATE TABLE Student
       ( StCode    char(3)    NOT NULL,
         Stname  char(20)     NOT NULL,
         StAge   int(2),
          PRIMARY KEY (StCode) );
```

PRIMARY KEY Constraint is defined after all column definitions (Table Level).

A Composite (multi-column) Primary key can be defined as only a Table level whereas Single-column Primary key can be defined in both way i.e. Column level or Table level.

Handling Tables

- **Viewing Table Structure:**
  You can view structure of any table after using database as-
  DESC[RIBE] <table name>
  ```
  mysql> DESC Student;
  ```
- **Deleting Table:**
  You can delete an existing table as-
  DROP TABLE [IF EXIST] <table name>
  ```
  mysql> DROP TABLE Student;
  ```
- **Creating Table from Existing Table:**
  CREATE TABLE <Table name>
  AS (<Select Query>);

```
mysql> CREATE TABLE Staff
          ( Select empID, ename, sex  From Emp);
```

```
mysql> CREATE TABLE Staff
          ( Select * From Emp);
```

It will create identical table as Emp

Modifying Table Structure

You can alter (modify) the structure of existing table by the using **ALTER TABLE….** Command of MySQL.

You can do the following with the help of ALTER TABLE.. Command.

- **Add a new Column or Constraints**
- **Modifying existing column (name, data type, size etc.)**
- **Delete an existing column or Constraints**
- **Changing Column Name**

ALTER TABLE <Table Name>

ADD|MODIFY|DROP|CHANGE <Column Definition(s)>

You can add/Delete/Modify multiple columns with single ALTER Command.

Modifying Table Structure

- ☐ **Adding new column**

  ALTER TABLE <Table Name>
  ADD <Column>[<data type> <size>][<Constraints>]

```
mysql> ALTER TABLE Student ADD (TelNo Integer);
mysql> ALTER TABLE Student ADD (Age Integer DEFAUL 10);
```

- ☐ **Modifying Existing Column**

  ALTER TABLE <Table Name>
  MODIFY <Column>[<data type> <size>] [<Constraints>]

```
mysql> ALTER TABLE Student MODIFY Name VARCHAR(40);
mysql> ALTER TABLE Employee MODIFY (Pay DECIMAL (10,2));
```

- ☐ **Removing Column & Constraints**

  ALTER TABLE <Table Name>
  DROP <Column name> |<Constraints>

```
mysql> ALTER TABLE Student DROP TelNo;
mysql> ALTER TABLE Emp DROP JOB, DROP Pay;
```