MySQL – Revision Tour

Chapter 8: MySQL Revision (XII-IP)

Informatics Practices
Class XII (CBSE Board)

"Open Teaching-Learning Material"

Visit www.ip4you.blogspot.com for more...

Authored By: Rajesh Kumar Mishra, PGT (Comp.Sc.)
Kendriya Vidyalaya Upper Camp, Dehradun (Uttarakhand)
e-mail : rkmalld@gmail.com

Revised as per CBSE Curriculum 2015

What is the Database?

- A database is an organized collection of interrelated data stored together to serve applications. It works like a container which may contain various database objects.
- Most of the databases store data in the form of Relations (also called Tables). Such databases are known as Relational Database.
- A software used to manage relational database is called RDBMS (Relational Database Management System).

Some commonly used RDBMS software are - Oracle, MySQL, MS SQL Server, SyBase, and Ingres etc.

Why Database System is used? (Advantages)

- Databases reduces Redundancy
  It removes duplication of data because data are kept at one place and all the applications refer to the centrally maintained database.
- Database controls Inconsistency
  When two copies of the same data do not agree to each other, then it is called Inconsistency. By controlling redundancy, the inconsistency is also controlled.
- Database facilitate Sharing of Data
  Data stored in the database can be shared among several users.
- Database ensures Security
  Data are protected against accidental or intentional disclosure to unauthorized person or unauthorized modification.
- Database maintains Integrity
  It enforces certain integrity rules to ensure the validity or correctness of data. For example, a date can’t be like 25/25/2000.

Data Model

Data model describes "How data is organized or stored" in the database. It may be-

- Relational Data Model
  In this model, data is organized into Relations or Tables (i.e. Rows and Columns). A row in a table represents a relationship of data to each other and also called a Tuple or Record. A column is called Attribute or Field.
- Network Data Model
  In this model, data is represented by a collection of records and relationship among data is shown by Links.
- Hierarchical Data Model
  In this model, Records are organized as Trees. Records at top level is called Root record and this may contains multiple directly linked children records.
- Object Oriented Data Model
  In this model, records are represented as objects. The collection of similar types of object is called class.

Data Models

- Attribute (Field)
- Table (Relation)

- Relation (Table)
  A Relation or Table is Two-Dimensional (Matrix) like structure arranged in Rows and Columns. It has the following properties-
  ◆ Column homogeneous - All items in a column are of same data type.
  ◆ Unique name - Each column assigned a unique name and must have atomic (indivisible) value.
  ◆ All rows of a relation are distinct i.e. no two identical rows (records) can exist in the relation.
  ◆ Ordering Rows (Records) or Columns (fields) are immaterial.
- Domain
  It is collection (set) of possible values from which the value for a column is derived.
- Tuple/Entity/Record:
  A Row of a table is called Tuple or Record.
- Attribute/Field:
  Column of a table is called Attribute or Field.
- Degree: Number of columns (attributes) in a table.
- Cardinality: Number of Records in a table.

RDBMS Terminology

- Relation (Table)
  A Relation or Table is Two-Dimensional (Matrix) like structure arranged in Rows and Columns. It has the following properties-
  ◆ Column homogeneous - All items in a column are of same data type.
  ◆ Each column assigned a unique name and must have atomic (indivisible) value.
  ◆ All rows of a relation are distinct i.e. no two identical rows (records) can exist in the relation.
  ◆ Ordering Rows (Records) or Columns (fields) are immaterial.
- Domain
  It is collection (set) of possible values from which the value for a column is derived.
- Tuple/Entity/Record:
  A Row of a table is called Tuple or Record.
- Attribute/Field:
  Column of a table is called Attribute or Field.
- Degree: Number of columns (attributes) in a table.
- Cardinality: Number of Records in a table.

Representation of Records and Relationship in various Data Model
**Concept of Keys**

In a Relation, each record must be unique i.e. no two identical records are allowed in the Database. A column or combination of columns which identifies a record is called Key of the Table. A key attribute must have unique (non-repeatable) value.

- **Primary Key**
  A set of one or more columns that can uniquely identify a record in the relation is called Primary Key.

- **Candidate Key**
  A Column or group of columns which can be used as primary key are called Candidate keys, as they are candidate to become as Primary key.

- **Alternate Key**
  A Candidate Key that is not a Primary key is called Alternate key.

- **Foreign Key**
  A non-key column whose values are derived from the primary key of some other table is called Foreign key.

**Introduction to MySQL**

MySQL is an Open Source, Fast and Reliable Relational Database Management System (RDBMS). It 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 free of cost. It is part of LAMP (Linux, Apache, MySQL, PHP/Perl/Python) Open Source group.

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

- **Security:**
  It offers privilege and password system for authorization.

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

- **Query Language**
  It uses SQL (Structured Query Language) as query language, which is standardized by ANSI.

**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 table, views, index etc.
  Example : CREATE, ALTER, DROP etc.

- **Data Manipulation Language (DML)**
  These commands are used to insert, delete, update and retrieve the stored records from the table.
  Ex. SELECT, INSERT, DELETE, UPDATE etc.

- **Transaction Control Language (TCL)**
  These commands are used to control the transaction.
  Ex. COMMIT, ROLLBACK, SAVEPOINT etc.

- **Data Control Language (DCL)**
  These commands are used to manipulate permissions or access rights to the tables etc.
  Ex. GRANT, REVOKE etc.

**Database Handling commands in MySQL**

- **Creating a Database.**
  The following command will create School database in MySQL.
  ```sql
  mysql> CREATE DATABASE School;
  ```

- **Opening a database**
  To open an existing database, following command is used.
  ```sql
  mysql> USE school;
  ```

- **Getting listings of database and tables**
  ```sql
  mysql> SHOW DATABASES;
  mysql> SHOW TABLES;
  ```

- **Deleting a Database and Table**
  ```sql
  mysql> DROP DATABASE School;
  mysql> DROP TABLE Student;
  ```

- **Viewing Table Structure**
  ```sql
  mysql> DESCRIBE Student;
  ```

**Data type 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 up to 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.

**Creating Tables**

- **Creating Simple Tables:**
  ```sql
  mysql> CREATE TABLE Emp
  (empID integer,
  ename char(30),
  city char(25),
  pay decimal(10,2));
  ```

- **Creating Table from Existing Table**
  ```sql
  CREATE TABLE Staff AS ( Select * From Emp);
  ```

**Select database();**

- **ShowTable();**
  Shows the name of currently open database
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 –

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

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.

  

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

  mysql> SELECT Name, City FROM Student;

- **Eliminating Duplicate values in a column**

  - **DISTINCT**

    mysql> SELECT City FROM Student;

    Only Unique Cities are displayed

    Mumbai is repeated

- **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.

  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 '"' is not required.

- **Selecting Specific Rows**

  - **WHERE clause**

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

    mysql> SELECT * FROM Student WHERE City='Mumbai';

    Name  Fname  City  Class
    ----------------------------------
    Amitabh Harivansh Rai  Allahabad 12
    Sharukh Firoz  Delhi 11
    Irphan Akbar  Jaipur 11
Selecting Specific Rows – 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 (!)

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

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

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

**Logical Operators**

- **OR, AND, NOT (!)**

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

The same query can also be written as:

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

Other Logical operators also can be applied:

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

**Specifying List – IN Operator**

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

The same query can also be written as:

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

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

**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_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.

```sql
mysql> SELECT * FROM Student
WHERE Name LIKE 'A%';
```

```sql
mysql> SELECT * FROM Student
WHERE Name LIKE '%Singh%';
```

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

**Searching NULL Values – IS Operator**

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

The `NOT` Operator can also be applied:

```sql
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.

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

To get descending order use `DESC` key word.

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

```sql
mysql> SELECT Name, Fname, City FROM Student
WHERE Name LIKE 'R%' ORDER BY Class;
```

Inserting Records in a Table

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

```sql
INSERT INTO  <Table Name> <[Column list]>
VALUES <list of values>
```

Suppose a table named STUDENT has been created with the following structure.

<table>
<thead>
<tr>
<th>StID</th>
<th>NAME</th>
<th>FNAME</th>
<th>DOB</th>
<th>CITY</th>
<th>CLASS</th>
</tr>
</thead>
</table>

We can insert a record as follows:

```sql
mysql> INSERT INTO Student VALUES ('s1', 'Amitabh', 'Harivansh', '1955-10-25', 'Mumbai', 12);
```

```sql
mysql> INSERT INTO Student VALUES ('s2','Sharukh Khan', NULL,'1972-5-25', 'Delhi', 10);
```

```sql
mysql> INSERT INTO Student (StID, FName, Name, Class)
VALUES (Select StID, Name,Class FROM Oldstudent WHERE Class>=11);
```

Both tables must have same structure.

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.

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

```sql
mysql> INSERT INTO Newstudent VALUES
(SELECT * FROM Oldstudent WHERE City='Mumbai');
```

```sql
mysql> INSERT INTO Newstudent (StID, Name, Class)
VALUES (Select StID, Name,Class FROM Oldstudent WHERE Class=21);
```
Chapter 8: MySQL Revision (XII-IP)

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;

mysql> DELETE FROM Student WHERE City='Mumbai';

mysql> DELETE FROM Student WHERE Class >=11;

You can recall (Undelete) records by giving ROLLBACK command.

```
ROLLBACK;
```

You can issue COMMIT command to record the changes permanently.

```
COMMIT;
```

Modifying Records in the Table
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= CONCAT('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 Functions

What is Function?
A function is a special types of command that performs some operation and returns a single value as a result. It is similar to method or function in JAVA, which can be called by giving some argument.

Types of Functions:
- Numeric Functions
- String Functions
- Date & Time Function
- Aggregate Functions

Numeric Functions
These functions may accept some numeric values and performing required operation, returns numeric values as result.

<table>
<thead>
<tr>
<th>Name</th>
<th>Purpose</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOD (M, N)</td>
<td>Returns remainder of M divide by N</td>
<td>Select MOD(11,4) ; 3</td>
</tr>
<tr>
<td>POWER (M, N)</td>
<td>Returns M^N</td>
<td>Select POWER(3,2); 9</td>
</tr>
<tr>
<td>ROUND (N, M)</td>
<td>Returns a number rounded off to M place. If M is -1, it rounds nearest 10. If M is not given, the N is rounded to the nearest integer.</td>
<td>Select ROUND(15.193,1); 15.2 Select ROUND(15.193); 15</td>
</tr>
<tr>
<td>SQRT (N)</td>
<td>Returns square root of N</td>
<td>Select SQRT(25); 5</td>
</tr>
<tr>
<td>TRUNCATE(N,M)</td>
<td>Returns number after truncating M decimal place.</td>
<td>Select TRUNCATE(15.79,1); 15.7</td>
</tr>
</tbody>
</table>

String Functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Purpose</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONCAT(str1,str2)</td>
<td>Returns concatenated string i.e. str1+str2.</td>
<td>Select CONCAT(Name, City) from Student;</td>
</tr>
<tr>
<td>LOWER(str) / LCASE(str)</td>
<td>Returns the given string in lower case.</td>
<td>Select LOWER('ABC'); abc</td>
</tr>
<tr>
<td>UPPER(str) / UCASE(str)</td>
<td>Returns the given String in upper case.</td>
<td>Select UPPER('abc'); ABC</td>
</tr>
<tr>
<td>LTRIM(str) / TRIM(leading)</td>
<td>Removes leading/trailing/both spaces from given string.</td>
<td>Select TRIM('    ABC  '); 'ABC'</td>
</tr>
<tr>
<td>LEFT(str, N)</td>
<td>Returns the (N) characters from left of the given string.</td>
<td>Select LEFT('Computer',4); Comp</td>
</tr>
<tr>
<td>RIGHT(str,N)</td>
<td>Returns the (N) characters from right of the given string.</td>
<td>Select RIGHT('Computer',4); rnt</td>
</tr>
<tr>
<td>SUBSTR(str,P,N)</td>
<td>Returns the substring for given position(P) and length (N). If N is -ve then backward position counted.</td>
<td>Select SUBSTR('Computer','3',2); ur</td>
</tr>
<tr>
<td>INSTR(str1,str2)</td>
<td>Returns the index of first occurrence of str2 in str1.</td>
<td></td>
</tr>
<tr>
<td>LENGTH(str)</td>
<td>Returns the length of given string</td>
<td>Select LENGTH('Computer'); 8</td>
</tr>
</tbody>
</table>

Date & Time Functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Purpose</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>CURDATE()</td>
<td>Returns the current date in YYYY-MM-DD format</td>
<td>Select CURDATE(); 2013-10-02</td>
</tr>
<tr>
<td>NOW()</td>
<td>Returns the current date &amp; Time as YYYY-MM-DD HH:MM:SS</td>
<td>Select NOW(); 2013-10-02 11:30:02</td>
</tr>
<tr>
<td>SYSDATE()</td>
<td>Returns the current date &amp; Time as YYYY-MM-DD HH:MM:SS</td>
<td>Select SYSDATE(); 2013-10-02 11:30:10</td>
</tr>
<tr>
<td>DATE()</td>
<td>Returns the date part of a date-time expression.</td>
<td>Select DATE(SYSDATE()); 2013-10-02</td>
</tr>
<tr>
<td>MONTH()</td>
<td>Returns the Month/Year from given date argument.</td>
<td>Select MONTH(SYSDATE()); 2013-10-02</td>
</tr>
<tr>
<td>DAYNAME()</td>
<td>Returns the name of the weekday</td>
<td>Select DAYNAME(CURDATE());</td>
</tr>
<tr>
<td>DAYOFMONTH()</td>
<td>Returns the day of month (1-31)</td>
<td>Select DAYOFMONTH(CURDATE());</td>
</tr>
<tr>
<td>DAYOFWEEK()</td>
<td>Returns the day of week (1-7)</td>
<td>Select DAYOFWEEK(CURDATE());</td>
</tr>
<tr>
<td>DAYOFYEAR()</td>
<td>Returns the day of year (1-366)</td>
<td>Select DAYOFYEAR(CURDATE());</td>
</tr>
</tbody>
</table>
### Aggregate Functions

<table>
<thead>
<tr>
<th>Name</th>
<th>Purpose</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUM()</td>
<td>Returns the sum of given column.</td>
<td>Select SUM(Pay) from Emp;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Select SUM(Pay), SUM(Net) from Emp;</td>
</tr>
<tr>
<td>MIN()</td>
<td>Returns the minimum value in the given column.</td>
<td>Select MIN(Pay) from Emp;</td>
</tr>
<tr>
<td>MAX()</td>
<td>Returns the maximum value in the given column.</td>
<td>Select MAX(Pay) from Emp;</td>
</tr>
<tr>
<td>AVG()</td>
<td>Returns the average value of the given column.</td>
<td>Select AVG(Pay) from Emp;</td>
</tr>
<tr>
<td>COUNT()</td>
<td>Returns the total number of values/records in given column.</td>
<td>Select COUNT(Name) from Emp;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Select COUNT(*) from Emp;</td>
</tr>
</tbody>
</table>

Aggregate Functions should not be used with other columns which may have multiple values in the table. The following query is illogical and wrong. Why? Think yourself...

```
Select sum(pay), name from Employee;
```

### 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  cont..

- **Adding new column**
  ```
  ALTER TABLE <Table Name>
  ADD <Column>[<data type> [<size>][<Constraints>]]
  ```

- **Modifying Existing Column**
  ```
  ALTER TABLE <Table Name>
  MODIFY <Column>[<data type> [<size>][<Constraints>]]
  ```

- **Removing Column & Constraints**
  ```
  ALTER TABLE <Table Name>
  DROP <Column name> |<Constraints>
  ```

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

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

```
mysql> ALTER TABLE Student MODIFY Name VARCHAR(40);
```

```
mysql> ALTER TABLE Emp MODIFY (Sal DECIMAL (10,2));
```

```
mysql> ALTER TABLE Student DROP TelNo;
```

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