# MySQL – Database Fundamentals

## Database:

A database system is computer based record keeping system that contains the data/ information about any entity in systematic way. It also provides various effective methods to retrieve and manipulate the data.

**Some Advantages of database are:**

- It Reduce data Redundancy ( Repetition / Duplication)
- It Controls data Inconsistency
- It facilitate sharing of data
- It enforce standards ( Rules)
- Data security as centralized database
- Integrity can be maintained

## Database Models:

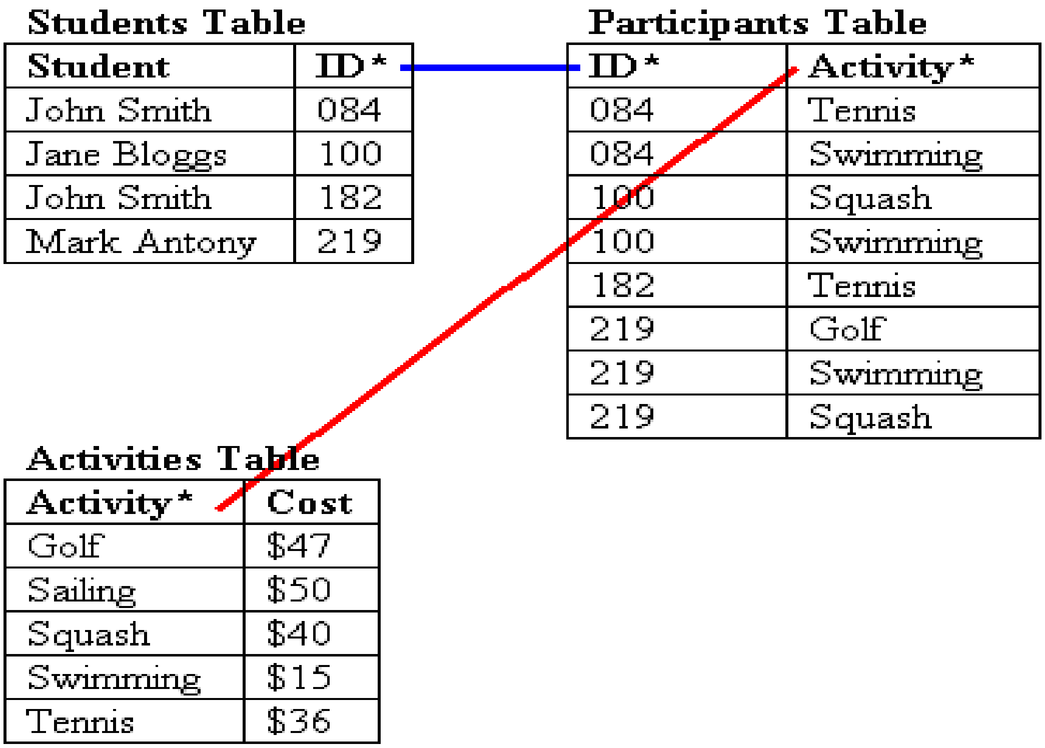
A database model is a collection of concepts that describes the structure of database and also describe some constraints which applied on the database system.

**Various types of database models are:**

- Relational Data Model
- Network Data Model
- Hierarchical Data Model

# Relational Data Model:

**Relational Data Model is mostly used model in database management system. In this model the data stored in tabular form that mean rows and columns. The rows in table represent the relationship among set of values.**



**Students Table**

| Student | ID* |
|---|---|
| John Smith | 084 |
| Jane Bloggs | 100 |
| John Smith | 182 |
| Mark Antony | 219 |

**Participants Table**

| ID* | Activity* |
|---|---|
| 084 | Tennis |
| 084 | Swimming |
| 100 | Squash |
| 100 | Swimming |
| 182 | Tennis |
| 219 | Golf |
| 219 | Swimming |
| 219 | Squash |

**Activities Table**

| Activity* | Cost |
|---|---|
| Golf | $47 |
| Sailing | $50 |
| Squash | $40 |
| Swimming | $15 |
| Tennis | $36 |

## Terminology of Relational Data model:

**Relation:** A "table" is termed as relation in database.

**Attribute:** The "Column" in Relation

**Tuple:** Row in Relation

**Domain:** Pool of values given in an attribute.

**Degree:** Number of Attributes in Relation

**Cardinality:** Number of Tuple in Relation

**Candidate Key:** All the combinations of attributes in relation which uniquely identify the tuples in relation and can serve as Primary Key.

**Primary Key:** Set of one or more attributes from Candidate Keys that uniquely identify the tuples in relation.

**Alternate Key:** Other than Primary key of Candidate keys are known as alternate key.

**Secondary Key:** Other than candidate keys in relation.

**Foreign Key:** It is a non-key attribute (key) whose values can derived from primary key of other table.

**View:** A view is a virtual table in database that not really exists with its own rights but derived from one or more base tables.

## MySQL Datatypes:

Datatypes are used to identify the type of value that an attribute can hold in database. It also helps in performing the various operations associated with type of values.

Some common datatypes are:

| Datatype | Explanation |
|---|---|
| CHAR | Stores Fixed length String type values |
| VARCHAR | Stores Variable length String type values |
| INT | Stores Integer type Values |
| FLOAT | Stores Floating point Values (Precise 23 digits) |
| DOUBLE | Stores Floating point Values (Precise 24-53 digits) |
| DATE | Stores date in YYYY-MM-DD format |
| TIME | Stores time in HH:MM:SS format |

## CHAR vs VARCHAR:

| CHAR | VARCHAR |
|---|---|
| It allowed to store Fixed length String type value | It allowed to store Variable length String type value |
| When a column declared as CHAR(n) then column reserve n-bytes memory for values. If value is shorter than length n byte, then blank bytes are added. In this case the total size remains n- bytes. | When a column declared as VARCHAR(n) then column reserve maximum n-bytes memory for values. If value is shorter than length n byte, then no blank bytes are added. In this case the total size remains exactly equal to size of specified value. |
| If specified size of value is larger than n-byte( Maximum), that time error occurred by Database. | |
| Example:<br>City CHAR(10)<br>City=”Jaipur”<br>Here maximum size is 10 bytes, and size of specified value (Jaipur) is 6. But the size of value remains 10 bytes. | Example:<br>City VARCHAR(10)<br>City=”Jaipur”<br>Here maximum size is 10 bytes, and size of specified value (Jaipur) is 6. The size of value will also remain 6 bytes. |

## Create Database in MySQL:

To create new database in MySQL, use following command.

CREATE DATABASE <Database Name>;

**Example:**

CREATE DATABASE student;

- Query OK, 1 row affected # This message will show, if created.
- Can’t create database Student, database exists. # This message will show, if database already existed.

Here “Student” is new Database where we can create tables and other objects of database. Note: Database Name should not existed.

## Open Database in MySQL:

To open existing database in MySQL, use following command.

```
USE  <Database Name>;
```

Note: Database Name should be existed.

Example:

```
USE student;
```

- Database Changed      # This message will show, if changed.
- Unknown database student # This message will show, if database not found.

## Show List of Database in MySQL:

To show the existing database list in MySQL, use following command. It will show the list in ascending order of their names.

```
SHOW DATABASES;
```

```
-------------------------------
Databases
-------------------------------
Information_schema
Student
------------------------------
```

## Show List of Tables in Database:

To show the existing list of tables in Database, use following command. It will show the list in ascending order of their names.

```
SHOW TABLES;
```

# SQL Constraints

- Constraints are some limitations or rules which can apply for storing data in the table.
- Constraints can be specified when a table is created (with the CREATE TABLE statement) or after the table is created (with the ALTER TABLE statement).
- We will focus on the following constraints:

♦ NOT NULL

♦ UNIQUE

♦ PRIMARY KEY

♦ FOREIGN KEY

♦ CHECK

♦ DEFAULT

## SQL NOT NULL Constraint

- The NOT NULL constraint enforces a column to NOT accept NULL values.
- The NOT NULL constraint enforces a field to always contain a value. This means that you cannot insert a new record, or update a record without adding a value to this field.
- The following SQL enforces the "P_Id" column and the "LastName" column to not accept NULL values:

```
CREATE TABLE Persons
(P_Id  int  NOT NULL,
LastName  varchar(25)  NOT NULL,
FirstName  varchar(25),
Address  varchar(55),
City  varchar(25)
);
```

## SQL UNIQUE Constraint

- The following SQL creates a UNIQUE constraint on the "P_Id" column when the "Persons" table is created:

-
  ```
  CREATE TABLE Persons
  (
  P_Id  int  NOT NULL UNIQUE,
  LastName  varchar(55) NOT NULL,
  FirstName  varchar(55),
  Address  varchar(55),
  City  varchar(55),
  );
  ```

## SQL PRIMARY KEY Constraint

- The PRIMARY KEY constraint uniquely identifies each record in a database table.
- Primary keys must contain unique values.
- A primary key column cannot contain NULL values.
- Each table should have a primary key, and each table can have only ONE primary key.
- The following SQL creates a PRIMARY KEY on the "P_Id" column when the "Persons" table is created:

-
  ```
  CREATE TABLE Persons
  (
  P_Id  int  PRIMARY KEY NOT NULL,
  LastName  varchar(55) NOT NULL,
  FirstName  varchar(55),
  Address  varchar(55),
  City  varchar(55),
  );
  ```

## SQL FOREIGN KEY Constraint

- A FOREIGN KEY in one table points to a PRIMARY KEY in another table.
- Note that the "P_Id" column in the "Orders" table points to the "P_Id" column in the "Persons" table.
- The "P_Id" column in the "Persons" table is the PRIMARY KEY in the "Persons" table.
- The "P_Id" column in the "Orders" table is a FOREIGN KEY in the "Orders" table.
- The FOREIGN KEY constraint is used to prevent actions that would destroy links between tables.
- The FOREIGN KEY constraint also prevents that invalid data form being inserted into the foreign key column, because it has to be one of the values contained in the table it points to.
- The following SQL creates a FOREIGN KEY on the "P_Id" column when the "Orders" table is created:

```
CREATE TABLE Orders
(O_Id int  NOT NULL,
OrderNo  int  NOT NULL,
P_Id  int,
PRIMARY KEY (O_Id),
FOREIGN KEY (P_Id) REFERENCES  Persons(P_Id)
);
```

## SQL CHECK Constraint

- The CHECK constraint is used to limit the value range that can be placed in a column.
- If you define a CHECK constraint on a single column it allows only certain values for this column.
- If you define a CHECK constraint on a table it can limit the values in certain columns based on values in other columns in the row.
- The following SQL creates a CHECK constraint on the "P_Id" column when the "Persons" table is created. The CHECK constraint specifies that the column "P_Id" must only include integers greater than 0.

  ```
  CREATE  TABLE  Persons
  (P_Id  int  NOT NULL,
  LastName  varchar(55) NOT NULL,
  FirstName  varchar(55),
  Address  varchar(55),
  City  varchar(55),
  CHECK (P_Id>0)
  );
  ```

## SQL DEFAULT Constraint

- The following SQL creates a DEFAULT constraint on the "City" column when the "Persons" table is created:

  ```
  CREATE TABLE Persons
  (P_Id  int  NOT NULL,
  LastName  varchar(55)  NOT NULL,
  FirstName  varchar(55),
  Address  varchar(55),
  City varchar(55)  DEFAULT ‘Jaipur'
  );
  ```

# CREATE TABLE in Database:

A table in database is the type of database object. A table in database can be created by using following command.

```
CREATE TABLE < table_name>
(
First_column_name datatype(size) constraint ,
Second_column_name datatype(size) constraint ,
Third_column_name datatype(size) constraint
) ;
```

Notice. A comma used for every new column

Notice. No comma used after last column or

No comma if there is no more column.

Notice. Semi Colon (;) used at end of query

**Example:**

**Create a table "Student" as per given below specification.**

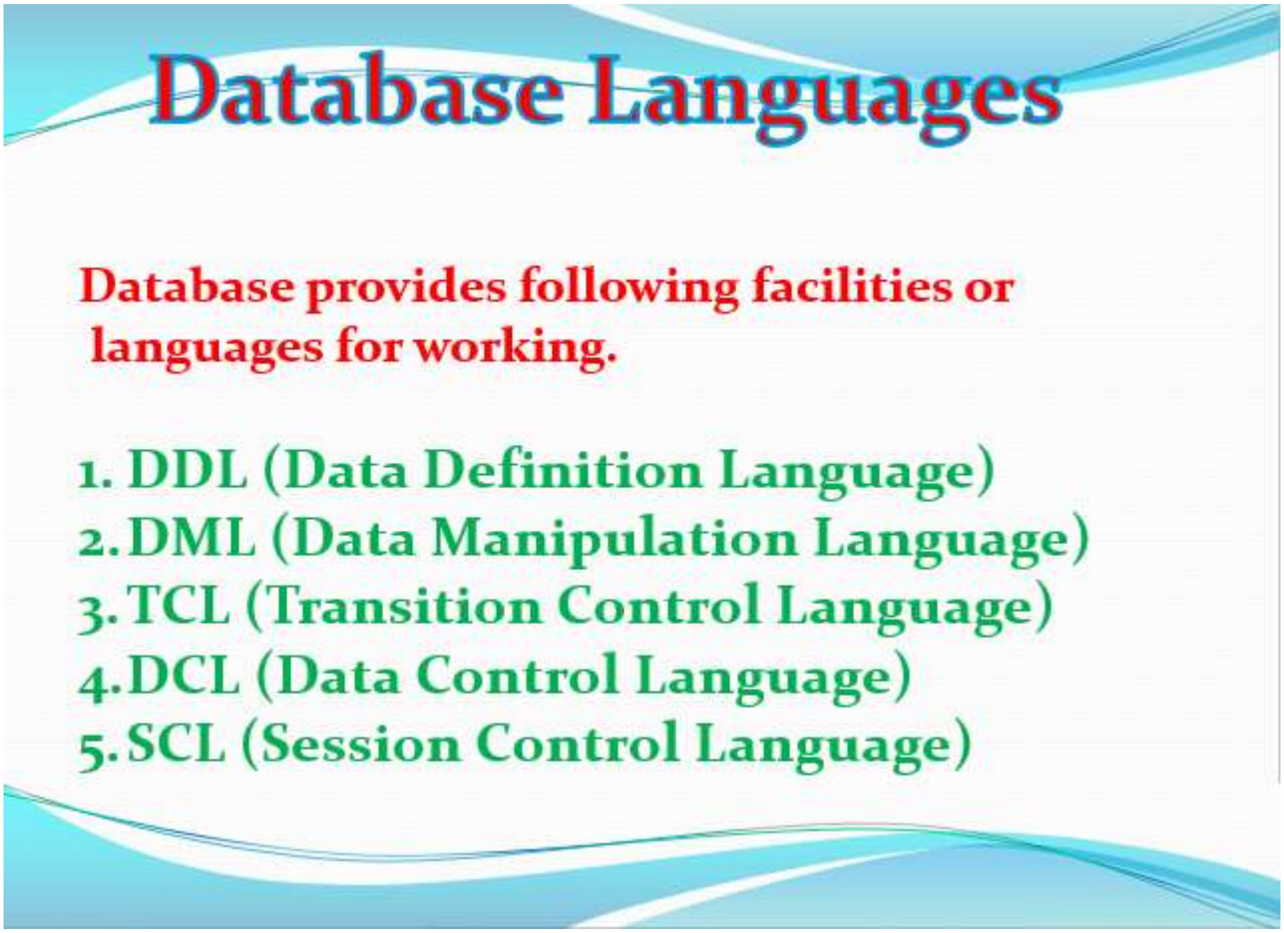
| Field Name | Datatype | Size | Constraint | Description |
|---|---|---|---|---|
| SID | INT | 3 | PRIMARY KEY | Student ID |
| NAME | VARCHAR | 25 | NOT NULL | Name of Student |
| AGE | INT | 2 | AGE > 5 YEAR | Age greater than 5 |
| GENDER | CHAR | 1 | DEFAULT 'M' | Use M / F / T |
| MARKS | FLOAT | ----- | ------------- | --------------- |

```
CREATE TABLE student
(
SID INT(3) PRIMARY KEY ,
NAME VARCHAR(25) NOT NULL ,
AGE INT(2) CHECK (AGE>5) ,
GENDER CHAR(1) DEFAULT 'M' ,
MARKS FLOAT
);
```

## DESCRIBE the Structure of TABLE:

**Use following commands.**
**DESC student;**

| Field | Type | NULL | Key | Default | Extra |
|---|---|---|---|---|---|
| SID | INT(3) | NO | PRI | NULL | |
| NAME | VARCHAR(25) | NO | | NULL | |
| AGE | INT(2) | YES | | NULL | |
| GENDER | CHAR(1) | YES | | M | |
| MARKS | FLOAT | YES | | NULL | |



## Database Languages

Database provides following facilities or languages for working.

1. DDL (Data Definition Language)
2. DML (Data Manipulation Language)
3. TCL (Transition Control Language)
4. DCL (Data Control Language)
5. SCL (Session Control Language)

**:: Finished ::**