# 18 UNDERSTANDING DATA
## (NCERT CLASS 12)

### INTRODUCTION

In general; data is a collection of characters; numbers, and other symbols that represents values of some situations or variables. Data is plural and singular of the word data is "datum". Using computers, data are stored in electronic forms because data processing becomes faster and easier as compared to manual data processing done by people. The Information and Communication Technology (ICT) revolution led by computer, mobile and Internet has resulted in generation of large volume of data and at a very fast pace. The following list contains some examples of data that we often come across.

(I) Name, age, gender, contact details, etc., of a person
(II) Transactions data generated through banking, ticketing, shopping, etc. whether online or offline.
(III) Images, graphics, animations, audio, video .
(IV) Documents and web pages
(V) Online posts, comments and messages
(VI) Signals generated by sensors
(VII) Satellite data including meteorological data, communication data, earth observation data, etc.

### IMPORTANCE OF DATA

Data allows organizations to more effectively determine the cause of problems. Data allows organizations to visualize relationships between what is happening in different locations, departments, and systems.

Besides business, following are some other scenarios where data are also stored and analysed for making decisions:

(I) The electronic voting machines are used for recording the votes cast. Subsequently, the voting data from all the machines are accumulated to declare election results in a short time as compared to manual counting of ballot papers.
(II) Scientists record data while doing experiments to calculate and compare results.
(III) Pharmaceutical companies record data while trying out a new medicine to see its effectiveness.
(IV) Libraries maintain data about books in the library and the membership of the library.
(V) The search engines give us results after analysing large volume of data available on the websites across World Wide Web (www).
(VI) Weather alerts are generated by analysing data received from various satellites.

### TYPES OF DATA

As data come from different sources, they can be in different formats. Two broad categories in which data can be classified on the basis of their format are:

(I) **Structured Data:** Data which is organised and can be recorded in a well defined format is called structured data.Structureddata is usually stored in computer in a tabular (in rows and columns) format where each column represents different data for a particular parameter called attribute/ characteristic/variable and each row represents data of an observation for different attributes.
(II) **Unstructured Data :** Data which are not in the traditional row and column structure is called unstructured data. Examples of unstructured data include web pages consisting of text as well as multimedia contents (image, graphics, audio/ video)'. Other examples include text documents, business reports, books, audio/video files, social media messages.

Unstructured data are sometimes described with the help of some other data called metadata. Metadata is basically data about data. For example, we describe different parts of an email as subject, recipient, main body, attachment, etc. These are the metadata for the email data. Likewise, we can have some metadata for an image file as image size (in KB or MB), image type (for example, JPEG, PNG), image resolution, etc.

### DATA COLLECTION

For processing data, we need to collect or gather data first. We can then store the data in a file or database for later use. Data collection here means identifying already available data or collecting from the appropriate sources.

### DATA STORAGE

Data storage is the process of storing data on storage devices so that data can be retrieved later. There are numerous digital storage devices available in the market like, Hard Disk Drive (HDD), Solid State Drive (SSD), CD/DVD, Tape Drive, Pen Drive, Memory Card, etc.

We store data like images, documents, audios/ videos, etc. as files in our computers. Likewise, school/ hospital data are stored in data files. We use computers to add, modify or delete data in these files or process these data files to get results. However, file processing has certain limitations, which can be overcome through Database Management System (DBMS).

## DATA PROCESSING

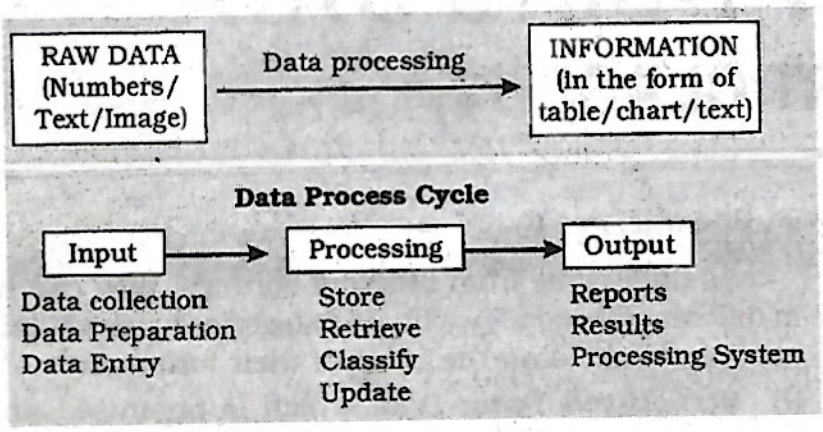



***Fig** - Steps in data processing*

## STATISTICAL TECHNIQUES FOR DATA PROCESSING

Summarisation methods are applied on tabular data for its easy comprehension. Commonly used statistical techniques for data summarisation are given below:

(I) **Measures of Central Tendency:** A measure of central tendency is a single value that gives us some idea about the data. Three most common measures of central tendency are the mean, median, and mode.

(a) **Mean :** Mean is simply the average of numeric values of an attribute. Mean is also called average. Suppose there are data on weight of 60 students in a class. Instead of looking at each of the data values, we can calculate the average to get an idea about the average weight of students in that class.

**Definition:** Given n values $x_1, x_2, x_3,...x_n$, mean is computed as :-

$$\frac{\sum_{i}^{n} xi.}{n}$$

Mean is not a suitable choice if there are outliers in the data. To calculate mean, the outliers or extreme values should be removed from the given data and then calculate mean of the remaining data.

**Note:** An outlier is an exceptionally large or small value, in comparison to other values of the data. Usually, outliers are considered as error since they can influence/affect the average or other statistical calculation based on the data.

(b) **Median :** Median is also computed for a single attribute/variable at a time. When all the values are sorted in ascending or descending order, the middle value is called the Median. When there are odd number of values, then median is the value at the middle position. If the list has even number of values, then median is the average of the two middle values. Median represents the central value at which the given data is equally divided into two parts.

(c) **Mode :** Value that appears most number of times in the given data of an attribute/variable is called Mode. It is computed on the basis of frequency of occurrence of distinct values in the given data. A data set has no mode if each value occurs only once. There may be multiple modes in the data if more than one values have same highest frequency. Mode can be found for numeric as well as non-numeric data.

(II) **Measures of Variability:** The measures of variability refer to the spread or variation of the values around the mean. They are also called measures of dispersion that indicate the degree of diversity in a data set. They also indicate difference within the group. Two different data sets can have the same mean, median or mode but completely different levels of dispersion, or vice versa. Common measures of dispersion or variability are Range and Standard Deviation.

(a) **Range:** It is the difference between maximum and minimum values of the data (the largest value minus the smallest value). Range can be calculated only for numerical data. It is a measure of dispersion and tells about coverage/spread of data values.

Let M be the largest or maximum value and S is the smallest or minimum value in the data, then Range is the difference between two extreme values i.e. M – S or Maximum – Minimum.

(b) **Standard deviation:** It refers to differences within the group or set of data of a variable. Like Range, it also measures the spread of data. However, unlike Range which only uses two extreme values in the data, calculation of standard deviation considers all the given data. It is calculated as the positive square root of the average of squared difference of each value from the mean value of data. Smaller value of standard deviation means data are less spread while a larger value of standard deviation means data are more spread.

Given n values x1, x2, x3,...xn, and their mean x, the standard deviation, represented as ó (greek letter sigma) is computed as:

$$\sigma = \sqrt{\frac{\sum_{i-1}^{n} (X_1 - \overline{X})^2}{n}}$$

It is important to understand statistical techniques so that one can decide which statistical technique to use to arrive at a decision. Different programming tools are available for efficient analysis of large volumes of data. These tools make use of statistical techniques for data analysis. One such programming tool is Python and it has libraries specially built for data processing and analysis.

# 19 DATABASE CONCEPT
## (NCERT CLASS 12)

### INTRODUCTION

A Database is a collection of DATA/INFORMATION that is organized so that it can be easily accessed, managed and updated.

In Database, Data is organized into rows, columns and tables, and it is indexed to make it easier to find relevant information. It works like a container which contains the various object like Tables, Queries, Reports etc. in organized way.

**A database has the following properties:**

(I) A database is a representation of some aspect of the real world also called miniworld. Whenever there are changes in this mini world they are also reflected in the database.

(II) It is designed, built and populated with data for specific purpose.

(III) It can be of any size and complexity.

(IV) It can be maintained manually or it may be computerized.

**Need for a Database :** Data is stored in the form of files. A number of application programs are written by programmers to insert, delete, modify and retrieve data from these files. New application programs will be added to the system as the need arises.

### FILE SYSTEM

A file can be understood as a container to store data in a computer. Files can be stored on the storage device of a computer system. Contents of a file can betexts, computer program code, comma separated values (CSV), etc. Likewise, pictures, audios/videos, web pages are also files.

Files stored on a computer can be accessed directly and searched for desired data. But to access data of a file through software, for example, to display monthly attendance report on school website, one has to write computer programs to access data from files.

**Limitations of a File System :**

Following are some of the limitations of file system:

(I) **Difficulty in Access :** Files themselves do not provide any mechanism to retrieve data. Data maintained in a file system are accessed through application programs. While writing such programs, the developer may not anticipate all the possible ways in which data may be accessed. So, sometimes it is difficult to access data in the required format and one has to write application program to access data.

(II) **Data Redundancy :** Redundancy means same data are duplicated in different places (files). Redundancy leads to excess storage use and may cause data inconsistency also.

(III) **Data Inconsistency :** Data inconsistency occurs when same data maintained in different places do not match.

(IV) **Data Isolation :** In a more complex system where data files are generated by different person at different times, files being created in isolation may be of different formats. In such case, it is difficult to write new application programs to retrieve data from different files maintained at multiple places, as one has to understand the underlying structure of each file as well.

(V) **Data Dependence :** Data are stored in a specific format or structure in a file. If the structure or format itself is changed, all the existing application programs accessing that file also need to be changed. Otherwise, the programs may not work correctly. This is data dependency. Hence, updating the structure of a data file requires modification in all the application programs accessing that file.

(VI) **Controlled Data Sharing :** Controlled access data sharing requires a request for access to the dataset to be approved. The requirements vary, but usually limit data sharing to researchers with a specific, relevant research question.It is very difficult to enforce this kind of access control in a file system while accessing files through application programs.

### DATABASE MANAEMENT SYSTEM

- A DBMS refers to a software that is responsible for storing, maintaining and utilizing database in an efficient way.
- A Database along with DBMS software is called Database System.
- Example of DBMS software are Oracle, MS SQL Server, MS Access, Paradox, DB2 and MySQL etc.
- MySQL is open source and freeware DBMS.

### WHY DO WE NEED DATABASE ?

(I) **To manage large chunks of data:** if size of data increases into thousands of records, it will simply create a problem to manage.Database can manage large amount of data.

(II) **Accuracy:** Through validation rule in database , data accuracy can be maintained.

(III) **Ease of updating data:** With the database, we can flexibly update the data according to our convenience. Moreover, multiple people can also edit data at same time.

(IV) **Security of data:** With databases we have security groups and privileges to restrict access.

(V) **Data integrity:** In databases, we can be assured of accuracy and consistency of data due to the built in integrity checks and access controls.

**Advantages of Database System :**

(I) Databases reduces Redundancy It removes duplication of data because data are kept at one place and all the application refers to the centrally maintained database.

(II) 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.

(III) Database facilitate Sharing of Data Data stored in the database can be shared among several users.

(IV) Database ensures Security Data are protected against accidental or intentional disclosure to unauthorized person or unauthorized modification.

(V) Database maintains Integrity It enforces certain integrity rules to insure the validity or correctness of data. For ex. A date can't be like 31/31/2000.

**Data Model- Way of data representation :**

Data model is a model or presentation which shows How data is organized or stored in the database. A data is modeled by one of the following given-

- **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 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 a objects. The collection of similar types of object is called class.

## RELATIONAL DATABASE

A relational database is a collective set of multiple data sets organized by tables, records and columns. Relational database establish a well-defined relationship between database tables. Tables communicate and share information, which facilitates data searcheability, organization and reporting. A Relational database use Structured Query Language (SQL), which is a standard user application that provides an easy programming interface for database interaction.

## RELATIONAL DATABASE TERMS

(I) **Relation (Table) :** A Relation or Table is Matrix like structure arranged in Rows and Columns. It has the following properties-

- **Atomicity :** Each column assigned a unique name and must have atomic(indivisible) value i.e. a value that can not be further subdivided.
- **No duplicity :** No two rows of relation will be identical i.e. in any two rows value in at least one column must be different. All items in a column are homogeneous i.e.same data type.
  v
- Ordering of rows and column is immaterial.

(II) Domain :It is collection of values from which the value is derived for a column.

(III) Tuple / Entity / Record - Rows of a table is called Tuple or Record.

(IV) Attribute/ Field- Column of a table is called Attribute or Field.

(V) Degree - Number of columns (attributes) in a table.

(VI) Cardinality - Number of rows (Records) in a table.

Three Important Properties of a Relation:-In relational data model, following three properties are observed with respect to a relation which makes a relation different from a data file or a simple table.

**Property 1:** imposes following rules on an attribute of the relation.

- Each attribute in a relation has a unique name.
- Sequence of attributes in a relation is immaterial.

**Property 2:** governs following rules on a tuple of a relation.

- Each tuple in a relation is distinct. For example, data values in no two tuples of relation ATTENDANCE can be identical for all the attributes. Thus, each tuple of a relation must be uniquely identified by its contents.
- Sequence of tuples in a relation is immaterial. The tuples are not considered to be ordered, even though they appear to be in tabular form.

**Property 3:** imposes following rules on the state of a relation.

- All data values in an attribute must be from the same domain (same data type).
- Each data value associated with an attribute must be atomic (cannot be further divisible into meaningful subparts). For example, GPhone of relation GUARDIAN has ten digit numbers which is indivisible.
- No attribute can have many data values in one tuple. For example, Guardian cannot specify multiple contact numbers under GPhone attribute.

- A special value "NULL" is used to represent values that are unknown or non-applicable to certain attributes. For example, if a guardian does not share his or her contact number with the school authorities, then GPhone is set to NULL (data unknown).

## KEYS IN A DATABASE

Key plays an important role in relational database; it is used for identifying unique rows from table & establishes relationship among tables on need.

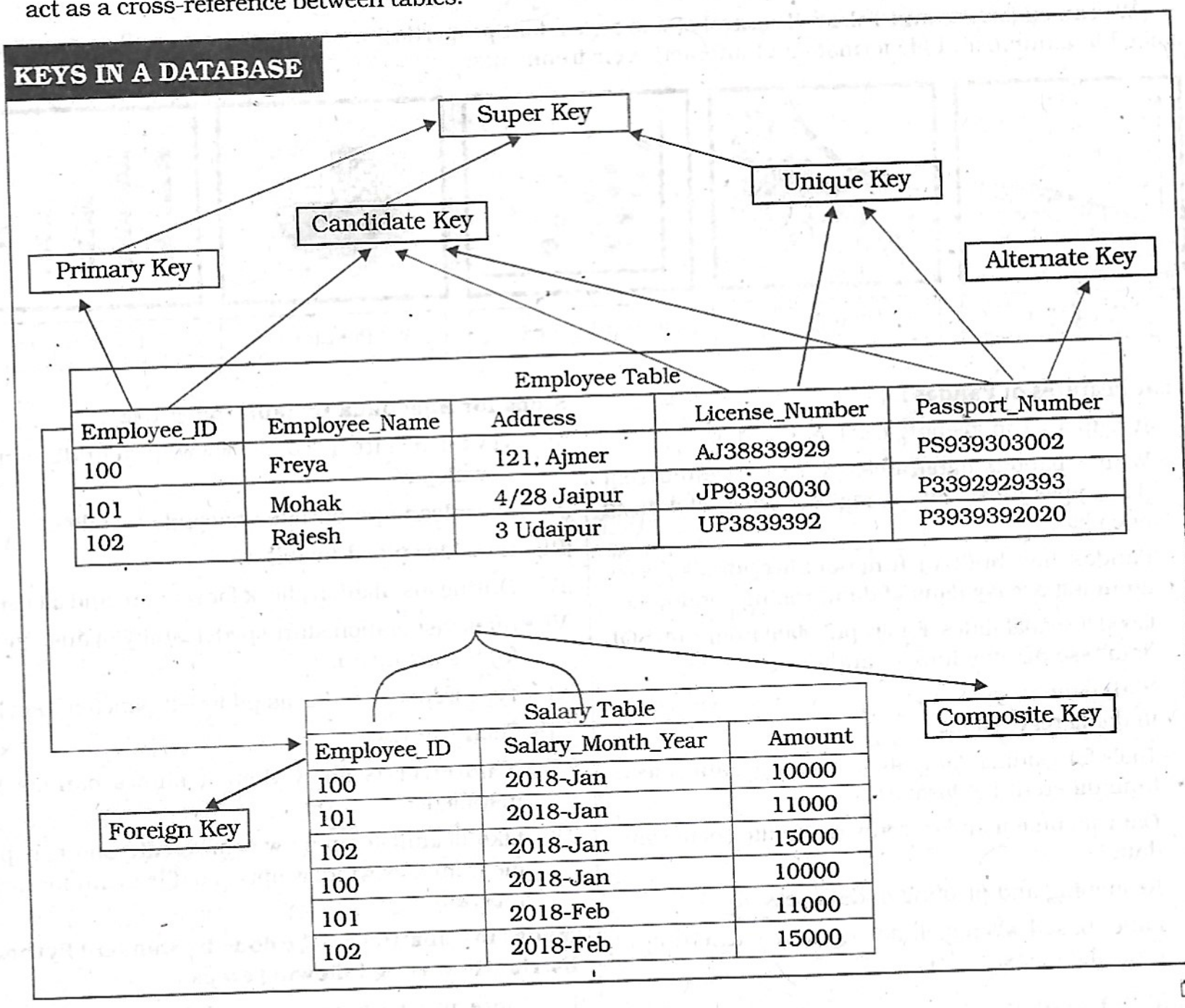
**Types of keys in DBMS :**

(I) **Primary Key** – A primary is a column or set of columns in a table that uniquely identifies tuples (rows) in that table.

(II) **Candidate Key** – It is an attribute or a set of attributes or keys participating for Primary Key, to uniquely identify each record in that table.

(III) **Alternate Key** – Out of all candidate keys, only one gets selected as primary key, remaining keys are known as alternate or secondary keys.

(IV) **Foreign Key** – Foreign keys are the columns of a table that points to the primary key of another table. They act as a cross-reference between tables.




Employee Table

| Employee_ID | Employee_Name | Address | License_Number | Passport_Number |
|---|---|---|---|---|
| 100 | Freya | 121, Ajmer | AJ38839929 | PS939303002 |
| 101 | Mohak | 4/28 Jaipur | JP93930030 | P3392929393 |
| 102 | Rajesh | 3 Udaipur | UP3839392 | P3939392020 |

Salary Table

| Employee_ID | Salary_Month_Year | Amount |
|---|---|---|
| 100 | 2018-Jan | 10000 |
| 101 | 2018-Jan | 11000 |
| 102 | 2018-Jan | 15000 |
| 100 | 2018-Jan | 10000 |
| 101 | 2018-Feb | 11000 |
| 102 | 2018-Feb | 15000 |

❑❑❑

# 21 STRUCTURED QUERY LANGUAGE (SQL) (NCERT CLASS 12)

## INTRODUCTION

There are many Relational Database Management Systems (RDBMS) such as MySQL, Microsoft SQL Server, PostgreSQL, Oracle, etc. that allow us to create a database consisting of relations. These RDBMS also allow us to store, retrieve and manipulate data on that database through queries.Now, we will learn how to create, populate and query databases using MySQL.

## STRUCTURED QUERY LANGUAGE (SQL)

SQL (Structured Query Language) is a standard language for accessing and manipulating databases. SQL commands are used to create, transform and retrieve information from Relational Database Management Systems and also used to create interface between user and database. By using SQL commands, one can search any data in the database and perform other functions like, create tables, add records, modify data, remove rows, drop table etc. SQL commands are used to implement the following;

(I) SQL can retrieve data from a database
(II) SQL can insert records in a database
(III) SQL can update records in a database
(IV) SQL can delete records from a database
(V) SQL can create new databases
(VI) SQL can create new tables in a database
(VII) SQL can create views in a database

- **Installing MySQL**

MySQL is an open source RDBMS software which can be easily downloaded from the official website https://dev.mysql.com/downloads. After installing MySQL, start MySQL service. The appearance of mysql> prompt means that MySQL is ready to accept SQL statements.

**Following are some important points to be kept in mind while using SQL:**

(I) SQL is case insensitive. For example, the column names 'salary' and 'SALARY' are the same for SQL.
(II) Always end SQL statements with a semicolon (;).
(III) To enter multiline SQL statements, we don't write ";" after the first line. We press the Enter key to continue on the next line. The prompt mysql> then changes to "->", indicating that statement is continued to the next line. After the last line, put ";" and press enter.

## DATA TYPES AND CONSTRAINTS IN MySQL

Database consists of one or more relations and each relation (table) is made up of attributes (column). Each attribute has a data type. We can also specify constraints for each attribute of a relation.

- **Data type of Attribute:** Data type of an attribute indicates the type of data value that an attribute can have. It also decides the operations that can be performed on the data of that attribute. For example, arithmetic operations can be performed on numeric data but not on character data. Commonly used data types in MySQL are numeric types, date and time types, and string types as shown in following Table:-

| Data type | Description |
|---|---|
| CHAR(n) | Specifies character type data of length n where n could be any value from 0 to 255. CHAR is of fixed length, means, declaring CHAR (10) implies to reserve spaces for 10 characters. If data does not have 10 characters (e.g., 'city' has four characters), MySQL fills the remaining 6 characters with spaces padded on the right. |
| VARCHAR(n) | Specifies character type data of length where n could be any value from 0 to 65535. But unlike CHAR, VARCHAR(n) is a variable-length data type. That is, declaring VARCHAR (30) means a maximum of 30 characters can be stored but the actual allocated bytes will depend on the length of entered string. So 'city' in VARCHAR (30) will occupy space needed to store 4 characters only. |
| INT | INT specifies an integer value. Each INT value occupies 4 bytes of storage. The range of unsigned values allowed in a 4 byte integer type are 0 to 4,294,967,295. For values larger than that, we have to use BIGINT, which occupies 8 bytes. |
| FLOAT | Holds numbers with decimal points. Each FLOAT value occupies 4 bytes. |
| DATE | The DATE type is used for dates in 'YYYY-MM-DD' format. YYYY is the 4 digit year, MM is the 2 digit month and DD is the 2 digit date. The supported range is '1000-01-01' to '9999-12-31'. |

• **Constraints :** Constraints are the certain types of restrictions on the data values that an attribute can have. Following table lists some of the commonly used constraints in SQL. They are used to ensure correctness of data. However, it is not mandatory to define constraints for each attribute of a table.

| Constraint | Description |
|---|---|
| NOT NULL | Ensures that a column cannot have NULL values where NULL means missing/ unknown/not applicable value. |
| UNIQUE | Ensures that all the values in a column are distinct/unique |
| DEFAULT | A default value specified for the column if no value is provided |
| PRIMARY KEY | The column which can uniquely identify each row/record in a table. |
| FOREGIN KEY | The column which refers to value of an attribute defined as primary key in another table |

## SQL FOR DATA DEFINITION

In order to be able to store data we need to first define the relation schema. Defining a schema includes creating a relation and giving name to a relation, identifying the attributes in a relation, deciding upon the datatype for each attribute and also specify the constraints as per the requirements. Sometimes, we may require to make changes to the relation schema also. SQL allows us to write statements for defining, modifying and deleting relation schemas. These are part of Data Definition Language (DDL).

• **CREATE Database :** To create a database, we use the CREATE DATABASE statement as shown in the following syntax: CREATE DATABASE databasename;

**Note:** In LINUX environment, names for database and tables are case-sensitive whereas in WINDOWS, there is no such differentiation. However, as a good practice, it is suggested to write database/table name in the same letter cases that were used at the time of their creation.

A DBMS can manage multiple databases on one computer. Therefore, we need to select the database that we want to use.

To know the names of existing databases, we use the statement SHOW DATABASES. From the listed databases, we can select the database to be used. Once the database is selected, we can proceed with creating tables or querying data.

• **CREATE Table :** After creating a database, we need to define relations in this database and specify attributes for each relation along with data type and constraint (if any) for each attribute. This is done using the CREATE TABLE statement.

Syntax:

```
CREATE TABLE tablename(
attributename1 datatype constraint,
attributename2 datatype constraint,
:
attributenameN datatype constraint);
```

It is important to observe the following points with respect to the CREATE TABLE statement:

(I) The number of columns in a table defines the degree of that relation, which is denoted by N.

(II) Attribute name specifies the name of the column in the table.

(III) Datatype specifies the type of data that an attribute can hold.

(IV) Constraint indicates the restrictions imposed on the values of an attribute. By default, each attribute can take NULL values except for the primary key.

**Note :** "," is used to separate two attributes and each statement terminates with a semi-colon (;). The arrow (->) is an interactive continuation prompt. If we enter an unfinished statement, the SQL shell will wait for us to enter the rest of the statement.

• **Describe Table :** We can view the structure of an already created table using the DESCRIBE statement or DESC statement.

Syntax:

```
DESCRIBE tablename;
```

• **ALTER Table :** After creating a table, we may realise that we need to add/remove an attribute or to modify the datatype of an existing attribute or to add constraint in attribute. In all such cases, we need to change or alter the structure (schema) of the table by using the alter statement.

(I) **Add primary key to a relation :** The following MySQL statement adds a primary key to the GUARDIAN relation:

```
mysql> ALTER TABLE GUARDIAN ADD PRIMARY
KEY (GUID);
Query OK, 0 rows affected (1.14 sec)
Records: 0 Duplicates: 0 Warnings: 0
```

(II) **Add foreign key to a relation :** Once primary keys are added, the next step is to add foreign keys to the relation (if any). Following points need to be observed while adding foreign key to a relation:

(a) The referenced relation must be already created.

(b) The referenced attribute(s) must be part of the primary key of the referenced relation.

(c) Data types and size of referenced and referencing attributes must be the same.

*Syntax:*

```
ALTER TABLE table_name ADD FOREIGN
KEY(attribute name)
REFERENCES referenced_table_name
(attribute name);
```

(III) **Add constraint UNIQUE to an existing attribute:**

*Syntax:*

ALTER TABLE table_name ADD UNIQUE (attribute name);

(IV) Add an attribute to an existing table :-Sometimes, we may need to add an additional attribute in a table. It can be done using the ADD attribute statement as shown in the following Syntax:

ALTER TABLE table_name ADD attribute name DATATYPE;

(V) **Modify datatype of an attribute :** We can change data types of the existing attributes of a table using the following ALTER statement.

*Syntax:*

ALTER TABLE table_name MODIFY attribute DATATYPE;

(VI) **Modify constraint of an attribute :** When we create a table, by default each attribute takes NULL value except for the attribute defined as primary key. We can change an attribute's constraint from NULL to NOT NULL using an alter statement.

*Syntax:*

ALTER TABLE table_name MODIFY attribute DATATYPE NOT NULL;

**Note:** We have to specify the data type of the attribute along with constraint NOT NULL while using MODIFY.

(VII) **Add default value to an attribute :** If we want to specify default value for an attribute, then use the following syntax:

ALTER TABLE table_name MODIFY attribute DATATYPE DEFAULT default_value;

**Note:** We have to specify the data type of the attribute along with DEFAULT while using MODIFY.

(VIII) **Remove an attribute :** Using ALTER, we can remove attributes from a table, as shown in the following syntax:

ALTER TABLE table_name DROP attribute;

(IX) **Remove primary key from the table :** Sometime there may be a requirement to remove primary key constraint from the table. In that case, Alter table command can be used in the following way:

*Syntax:*

ALTER TABLE table_name DROP PRIMARY KEY;

- **DROP Statement :**Sometimesa table in a database or the database itself needs to be removed. We can use a DROP statement to remove a database or a table permanently from the system. However, one should be very cautious while using this statement as it cannot be undone.

*Syntax to drop a table:*

DROP TABLE table_name;

*Syntax to drop a database:*

DROP DATABASE database_name;

**Note:** Using the DROP statement to remove a database will ultimately remove all the tables within it.

## SQL FOR DATA MANIPULATION:-

Data Manipulation using a database means either insertion of new data, removal of existing data or modification of existing data in the database.

- **INSERTION of Records :** INSERT INTO statement is used to insert new records in a table. Its syntax is:

INSERT INTO tablename

VALUES(value 1, value 2,....);

Here, value 1 corresponds to attribute 1, value 2 corresponds to attribute 2 and so on.

Note that we need not to specify attribute names in the insert statement if there are exactly the same numbers of values in the INSERT statement as the total number of attributes in the table.

**Caution:** While populating records in a table with foreign key, ensure that records in referenced tables are already populated.

## SQL FOR DATA QUERY

SQL provides efficient mechanisms to retrieve data stored in multiple tables in MySQL database (or any other RDBMS). The SQL statement SELECT is used to retrieve data from the tables in a database and is also called a query statement.

- **SELECT Statement :**

The SQL statement SELECT is used to retrieve data from the tables in a database and the output is also displayed in tabular form.

*Syntax:*

SELECT attribute1, attribute2, ...

FROM table_name

WHERE condition;

Here, attribute1, attribute2, ... are the column names of the table table_name from which we want to retrieve data. The FROM clause is always written with SELECT clause as it specifies the name of the table from which data is to be retrieved. The WHERE clause is optional and is used to retrieve data that meet specified condition(s).

To select all the data available in a table, we use the following select statement:

SELECT * FROM table_name;

- **QUERYING using Database OFFICE**

Organisations maintain databases to store data in the form of tables.

(I) **Retrieve selected columns :** The following query selects employee numbers of all the employees:

mysql> SELECT EmpNo FROM EMPLOYEE;

(II) **Renaming of columns :-** In case we want to rename any column while displaying the output, it can be done by using the alias 'AS'. The following query selects Employee name as Name in the output for all the employees:

```
mysql> SELECT EName as Name FROM EMPLOYEE;
```

(III) **Distinct Clause :** By default, SQL shows all the data retrieved through query as output. However, there can be duplicate values. The SELECT statement when combined with DISTINCT clause, returns records without repetition (distinct records). For example, while retrieving a department number from employee relation, there can be duplicate values as many employees are assigned to the same department. To select unique department number for all the employees, we use DISTINCT as shown below:

```
mysql> SELECT DISTINCT DeptId FROM EMPLOYEE;
```

(IV) **WHERE Clause :** The WHERE clause is used to retrieve data that meet some specified conditions. In the OFFICE database, more than one employee can have the same salary.

**Note :-** "=" operator is used in the WHERE clause. Other relational operators (<, <=, >, >=, !=) can be used to specify such conditions. The logical operators AND, OR, and NOT are used to combine multiple conditions.

(V) **Membership operator IN :** The IN operator compares a value with a set of values and returns true if the value belongs to that set. The above query can be rewritten using IN operator as shown below:

```
mysql> SELECT
```

(VI) **ORDER BY Clause :** ORDER BY clause is used to display data in an ordered form with respect to a specified column. By default, ORDER BY displays records in ascending order of the specified column's values. To display the records in descending order, the DESC (means descending) keyword needs to be written with that column.

(VII) **Handling NULL Values :** SQL supports a special value called NULL to represent a missing or unknown value. For example, the Gphone column in the GUARDIAN table can have missing value for certain records. Hence, NULL is used to represent such unknown values. It is important to note that NULL is different from 0 (zero).

Also, any arithmetic operation performed with NULL value gives NULL. For example: 5 + NULL = NULL because NULL is unknown hence the result is also unknown. In order to check for NULL value in a column, we use IS NULL operator.

(VIII) **Substring pattern matching :** Many a times we come across situations where we do not want to query by matching exact text or value. Rather, we are interested to find matching of only a few characters or values in column values.

The LIKE operator makes use of the following two wild card characters:

- % (per cent)- used to represent zero, one, or multiple characters.
- _ (underscore)- used to represent exactly a single character .

## DATA UPDATION AND DELETION

Updation and deletion of data are also part of SQL Data Manipulation Language (DML).

- **Data Updation** :-This command is used to implement modification of the data values.

**Syntax:**

UPDATE <table name>

SET <column name1>=new value, <column name>=new value etc

[WHERE <condition>];

- **Data Deletion** :-DELETE statement is used to delete/remove one or more records from a table.

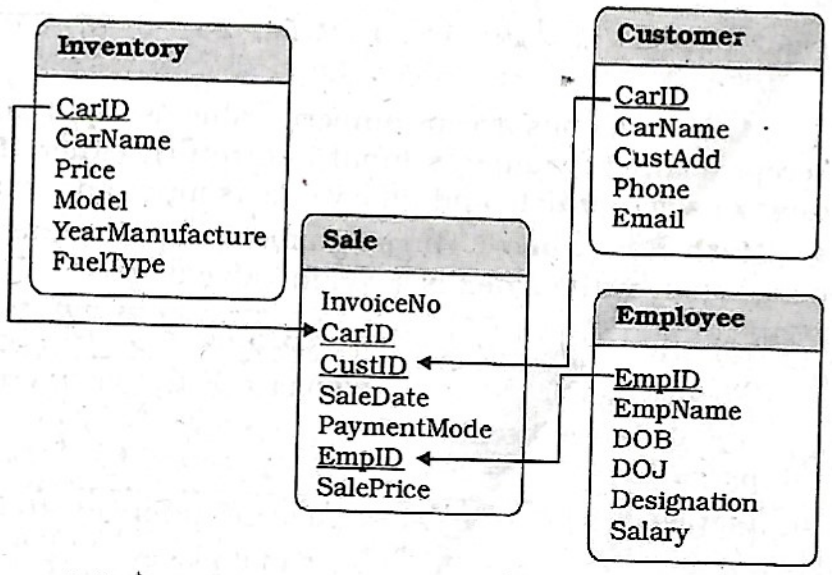
*Syntax:*

DELETE FROM table_name

WHERE condition;

## FUNCTIONS IN SQL

In this section, we will understand how to use single row functions, multiple row functions, group records based on some criteria, and working on multiple tables using SQL.



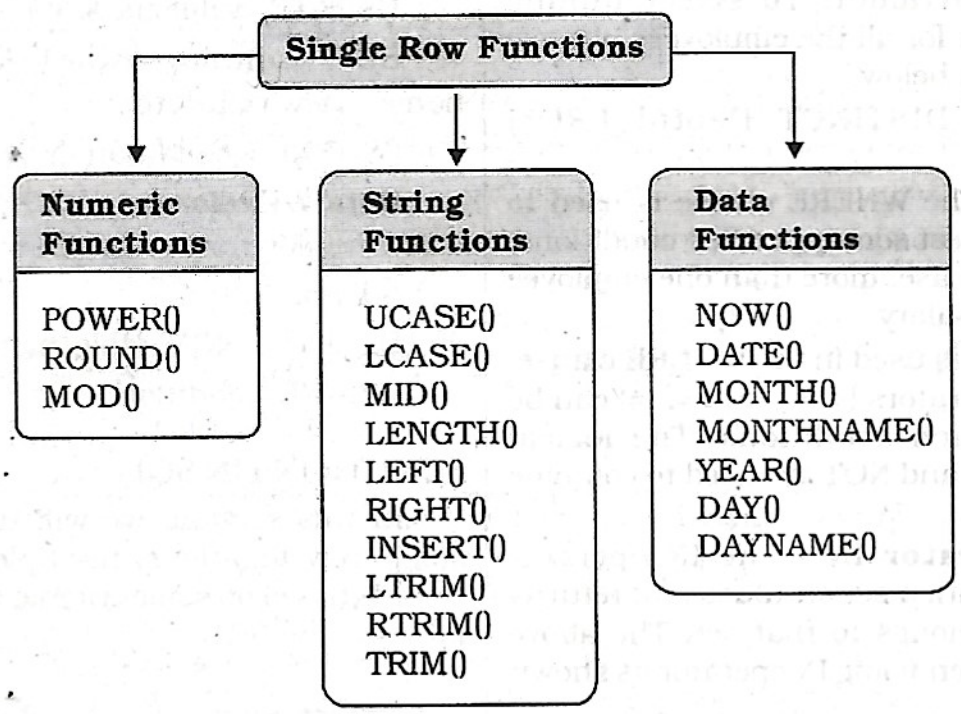

Let us create a database called CARSHOWROOM having the schema as shown in Figure above. It has the following four relations:

1) **INVENTORY :** Stores name, price, model, year of manufacturing, and fuel type for each car in inventory of the showroom,
2) **CUSTOMER :** Stores customer id, name, address, phone number and email for each customer,
3) **SALE :** Stores the invoice number, car id, customer id, sale date, mode of payment, sales person's employee id and selling price of the car sold,
4) **EMPLOYEE :** Stores employee id, name, date of birth, date of joining, designation and salary of each employee in the showroom.

We know that a function is used to perform some particular task and it returns zero or more values as a result. Functions are useful while writing SQL queries also. Functions can be applied to work on single or multiple records (rows) of a table. Depending on their application in one or multiple rows, SQL functions are categorised as Single Row functions and Aggregate functions.

- **Single Row Functions :**These are also known as Scalar functions. Single row functions are applied on a single value and return a single value.

Below Figure lists different single row functions under three categories — Numeric (Math), String, Date and Time.




Math Functions accept numeric value as input and return a numeric value as a result. String Functions accept character value as input and return either character or numeric values as output. Date and Time functions accept date and time value as input and return numeric or string or Date and Time as output.

- **Math Functions :** Three commonly used numeric functions are POWER(), ROUND() and MOD(). Their usage along with syntax is given in Table below:-

| Function | Description | Example with output |
|---|---|---|
| POWER(X,Y)<br>can also be written as<br>POW(X,Y) | Calculates X to the power Y. | mysql> SELECT POWER(2,3);<br>Output:<br>8 |
| ROUND(N,D) | Rounds off number N to D number of decimal places.<br>Note: If D=0, then it rounds off the number to the nearest integer. | mysql>SELECT ROUND(2912.564, 1);<br>Output:<br>2912.6<br>mysql> SELECT ROUND(283.2);<br>Output:<br>283 |
| MOD(A, B) | Returns the remain<br>der after dividing number A by number B. | mysql> SELECT MOD(21, 2);<br>Output:<br>1 |

- **String Functions :** String functions can perform various operations on alphanumeric data which are stored in a table. They can be used to change the case (uppercase to lowercase or vice-versa), extract a substring, calculate the length of a string and so on.String functions and their usage are shown in Table below:-

| Function | Description | Example with output |
|---|---|---|
| UCASE(string)<br>OR<br>UPPER(string) | converts string into uppercase. | mysql> SELECT UCASE<br>("Informatics Practices");<br>Output:<br>INFORMATICS PRACTICES |
| LOWER(string)<br>OR<br>LCASE(string) | converts string into lowercase. | mysql> SELECT LOWER<br>("Informatics Practices");<br>Output:<br>informatics practices |
| MID(string, pos, n)<br>OR<br>SUBSTRING<br>(string, pos, n)<br>OR<br>SUBSTR<br>(string, pos, n) | Returns a substring of size n starting from the specified position (pos) of the string. If n is not specified, it returns the substring from the position pos till end of the string. | mysql> SELECT MID("Informatics", 3, 4);<br>Output: form<br>mysql> SELECT MID('Informatics',7);<br>Output:<br>atics |
| LENGTH(string) | Return the number of characters in the specified string. | mysql> SELECT LENGTH("Informatics");<br>Output:<br>11 |
| LEFT(string, N)<br><br>RIGHT(string, N) | Returns N number of characters from the left side of the string.<br><br>Returns N number of characters from the right side of the string. | mysql> SELECT LEFT("Computer", 4);<br>Output:<br>Comp<br>mysql> SELECT RIGHT("SCIENCE", 3);<br>NCE |
| INSTR<br>(string, substring) | Returns the position of the first occurrence of the substring in the given string. Returns 0, if the substring is not present in the string. | mysql> SELECT INSTR("Informatics", "ma");<br>Output:<br>6 |
| LTRIM(string) | Returns the given string after removing leading white space characters. | mysql> SELECT LENGTH(" DELHI"),<br>LENGTH(LTRIM(" DELHI"));<br>Output:<br>+---+---+<br>\| 7 \| 5 \|<br>+---+---+<br>1 row in set (0.00 sec) |
| RTRIM(string) | Returns the given string after removing trailing white space characters. | mysql>SELECT LENGTH("PEN ")<br>LENGTH(RTRIM("PEN ));<br>Output:<br>+---+---+<br>\| 5 \| 3 \|<br>+---+---+<br>1 row in set (0.00 sec) |
| TRIM(string) | Returns the given string after removing both leading and trailing white space characters. | mysql> SELECT LENGTH(" MADAM "),<br>LENGTH(TRIM(" MADAM "));<br>Output:<br>+---+---+<br>\| 9 \| 5 \|<br>+---+---+<br>1 row in set (0.00 sec) |

- **Date and Time Functions :** There are various functions that are used to perform operations on date and time data. Some of the operations include displaying the current date, extracting each element of a date (day, month and year), displaying day of the week and so on. Below Table explains various date and time functions.

| Function | Description | Example with output |
|---|---|---|
| NOW() | It returns the current system date and time. | mysql> SELECT NOW(); Output: 2019-07-11 19:41:17 |
| DATE()<br><br>MONTH(date) | It returns the date part from the given date/time expression.<br><br>It returns the month in numeric form from the date. | mysql> SELECT DATE(NOW()); Output: 2019-07-11<br>mysql> SELECT MONTH(NOW()); Output: 7 |
| MONTHNAME(date) | It returns the month name from the specified date. | mysql> SELECT MONTHNAME("2003-11-28"); Output: November |
| YEAR(date) | It returns the year from the date. | mysql> SELECT YEAR("2003-10- 03"); Output: 2003 |
| DAY(date) | It returns the day part from the date. | mysql> SELECT DAY("2003-03- 24"); Output: 24 |
| DAYNAME(date) | It returns the name of the day from the date. | mysql> SELECT DAYNAME("2019-07-11"); Output: Thursday |

- **Aggregate functions :** These functions are also called Multiple Row functions. These functions work on a set of records as a whole and return a single value for each column of the records on which the function is applied.

**Differences between Single and Multiple Row Functions :**

| Single Row Function | Multiple Row Function |
|---|---|
| 1. It operates on a single row at a time | 1. It operates on groups of rows. |
| 2. It returns one result per row. | 2. It returns one result for a group of rows. |
| 3. It can be used in Select, Where, and Order by clause. | 3. It can be used in the select clause only. |
| 4. Math, String and Date functions are examples of single row functions. | 4. Max(), Min(), Avg(), Sum(), Count() and Count(*) are examples of multiple row functions. |

**Aggregate Functions in SQL :**

| Function | Description | Example with output |
|---|---|---|
| MAX(column) | Returns the largest value from the specified column. | mysql> SELECT MAX(Price) FROM INVENTORY; Output: 673112.00 |
| MIN(column) | Returns the smallest value from the specified column. | mysql> SELECT MIN(Price) FROM INVENTORY; Output: 355205.00 |
| AVG(column) | Returns the average of the values in the specified column. | mysql> SELECT AVG(Price) FROM INVENTORY; Output: 576091.625000 |

| | | |
|---|---|---|
| SUM(column) | Returns the sum of the values for the specified column. | mysql> SELECT SUM(Price)<br>FROM INVENTORY;<br>Output:<br>4608733.00 |
| COUNT(*) | Returns the number of records in a table.<br>**Note:** In order to display the number of records that matches a particular criteria in the table, we have to use COUNT(*) with WHERE clause. | mysql> SELECT COUNT(*) from MANAGER;<br>+----------+<br>\| count(*) \|<br>+----------+<br>\| 4 \|<br>+----------+<br>1 row in set (0.00 sec) |
| COUNT(column) | Returns the number of values in the specified column ignoring the NULL values.<br>**Note:** In this example, let us consider a MANAGER table having two attributes and four records. | mysql> SELECT * from MANAGER;<br>+------ +----------+<br>\| MNO \| MEMNAME \|<br>+------ +----------+<br>\| 1 \| AMIT \|<br>\| 2 \| KAVREET \|<br>\| 3 \| KAVITA \|<br>\| 4 \| NULL \|<br>+------+----------+<br>4 rows in set (0.00 sec)<br>mysql> SELECT COUNT(MEMNAME)<br>FROM MANAGER;<br>+----------------+<br>\| COUNT(MEMNAME) \|<br>+----------------+<br>\| 3 \|<br>+----------------+<br>1 row in set (0.01 sec) |

## GROUP BY CLAUSE IN SQL

At times we need to fetch a group of rows on the basis of common values in a column. This can be done using a group by clause. It groups the rows together that contains the same values in a specified column. We can use the aggregate functions (COUNT, MAX, MIN, AVG and SUM) to work on the grouped values. HAVING Clause in SQL is used to specify conditions on the rows with Group By clause.

## OPERATIONS ON RELATIONS

We can perform certain operations on relations like Union, Intersection and Set Difference to merge the tuples of two tables. These three operations are binary operations as they work upon two tables.

These operations can only be applied if both the relations have the same number of attributes and corresponding attributes in both tables have the same domain.

- **UNION (∪) :** This operation is used to combine the selected rows of two tables at a time. If some rows are same in both the tables, then result of the Union operation will show those rows only once.
- **INTERSECT (∩) :** Intersect operation is used to get the common tuples from two tables and is represented by symbol ?.
- **MINUS (-) :** This operation is used to get tuples/rows which are in the first table but not in the second table and the operation is represented by the symbol - (minus).
- **Cartesian Product (X) :** Cartesian product operation combines tuples from two relations. It results in all pairs of rows from the two input relations, regardless of whether or not they have the same values on common attributes. It is denoted as 'X'.

The degree of the resulting relation is calculated as the sum of the degrees of both the relations under consideration. The cardinality of the resulting relation is calculated as the product of the cardinality of relations on which cartesian product is applied.

## USING TWO RELATIONS IN QUERY

Till now we have written queries in SQL using a single relation only. In this section, we will learn to write queries using two relations.

- **Cartesian product on two tables :** When more than one table is to be used in a query, then we must specify the table names by separating commas in the FROM clause. On execution of such a query, the DBMS (MySql) will first apply cartesian product on specified tables to have a single table.
- **JOIN on two tables :** JOIN operation combines tuples from two tables on specified conditions. This is unlike cartesian product which make all possible combinations of tuples. While using the JOIN clause of SQL, we specify conditions on the related attributes of two tables within the FROM clause. Usually, such an attribute is the primary key in one table and foreign key in another table.

Following are some of the points to be considered while applying JOIN operations on two or more relations:

(I) If two tables are to be joined on equality condition on the common attribute, then one may use JOIN with ON clause or NATURAL JOIN in FROM clause. If three tables are to be joined on equality condition, then two JOIN or NATURAL JOIN are required.

(II) In general, N-1 joins are needed to combine N tables on equality condition.

(III) With JOIN clause, we may use any relational operators to combine tuples of two tables.

□□□