DATABASE

**Database:** It is a collection of data in a format that can be easily accessed.

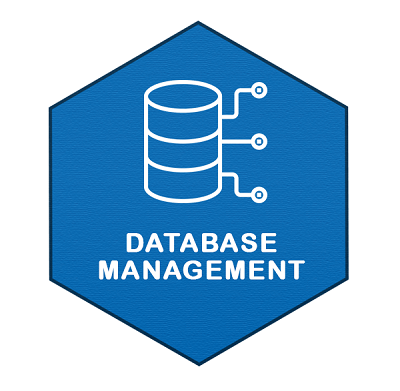
To manage these databases, **Database Management System** (DBMS) are used.



**System**



Database



**DBMS**: It is a software used to create, manage and organize data in a structured manner.

DBMS provide an environment to store and retrieve the data in convenient and efficient manner.

Example: A university database organizes the data about student, faculty, staff etc.

**Key Features of DBMS:**

* **Data Modeling:** Tools for defining the structure and relationships of data within a database.
* **Data Storage and Retrieval:** Manages storing and accessing data, enabling efficient searching and querying.
* **Concurrency Control:** Ensures multiple users can access the database simultaneously without conflicts.
* **Data Integrity and Security:** Enforces rules for data accuracy and restricts access to authorized users only.
* **Backup and Recovery:** Provides mechanisms to back up data and restore it after system failures.

**DBMS can be classified into two types:**

1. Relational Database Management System (RDBMS)
2. Non-Relational Database Management System (NoSQL or Non-SQL)

**RDBMS:** Data is organized in the form of tables and each table has a set of rows and columns. The data are related to each other through primary and foreign keys.

Example: MySQL, Oracle, PostgreSQL

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Class** |
| 101 | Charlie | 9 |
| 102 | Bob | 11th |
| 103 | Alice | 10 |

**NoSQL**: Data is organized in the form of key-value pairs, documents, graphs, or column-based. These are designed to handle large-scale, high-performance scenarios.

Example: MongoDB, Redis

{

“RollNo”: 1,

“Class”: 5th,

“Name”: “Bob”

}

**SQL INTRODUCTION**

SQL stands for Structured Query Language

It is used for accessing and manipulating databases.

**What can SQL do?**

With the help of sql we can execute query, retrieve data, create database and tables, insert records, update records, delete record, create views and set permission on tales, procedure and views.

**SYNTAX**:

Select \* from table\_name;

NOTE: SQL keywords are not case sensitive: Select is same as SELECT.

IMPORTANT Commands

|  |  |  |
| --- | --- | --- |
| **Commands** | **Working** | **Syntax** |
| SELECT | Exact data from database | Select \* from table\_name; |
| UPDATE | Update data in a database | Update table\_name set  col\_1 = value1,… where condition; |
| DELETE | Delete data from a database | Delete from table\_name where condition; |
| INSERT INTO | Insert new record into a database | Insert into table\_name (col\_1, col\_2, col\_3,..) values(value1, value2, value3,…); |
| CREATE DATABASE | Create a new database | Create database database\_name; |
| ALTER DATABASE | Modifies a database. |  |
| CREATE TABLE | Create a new table | Create table table\_name; |
| ALTER TABLE | Modifies a table | Alter table table\_name add col\_name datatype; |
| DROP TABLE | Delete a table | Drop table table\_name; |
| CREATE INDEX | Creates an index | Create index index\_name on table\_name (column1, column2, ...); |
| DROP INDEX | Deletes an index | Drop index table\_name.index\_name; |

* SELECT: It is used to select the data from a database.

Select \* from table\_name;

* CREATE DATABASE: Create database statement is used to create new database.

CREATE DATABASE database\_name;

Once database is created, we can check it in the list of databases with the SHOW DATABASES command.

* CREATE TABLE: Create table statement is used to create a new table in a database.

CREATE TABLE table\_name (column1 datatype, column2 datatype, column3 datatype);

* INSERT INTO: Insert into statement is used to insert new records in a table.

**There are two ways to write INSERT INTO statement:**

1. Specify both the column names and the values to be inserted:

INSERT INTO table\_name (column1, column2, column3,...) VALUES (value1, value2, value3, ...);

1. When we use this method, we make sure the order of the values is in the same order as the columns in the table.

INSERT INTO table\_name VALUES (value1, value2, value3, ...);

* WHERE: it is used to filter records.

SELECT column1, column2, ... FROM table\_name WHERE condition;

**NOTE:** The WHERE clause is not only used in SELECT statements, it is also used in UPDATE, DELETE, etc.

**Operators in the WHERE Clause**

|  |  |  |
| --- | --- | --- |
| **Operator** | **Description** | **Example** |
| **=** | Equal | Select \* from customer where city = “Mumbai” |
| **>** | Greater Than | Select \* from customer where price > 50; |
| **<** | Less Than | Select \* from customer where price < 50; |
| **>=** | Greater Than or Equal | Select \* from customer where price >= 50; |
| **<=** | Less Than or Equal | Select \* from customer where price <= 50; |
| **< >** | Not Equal | Select \* from customer where price < > 100; |
| LIKE | Search for a pattern | Select \* from customer where city Like ‘s%’; |
| IN | To specify multiple possible values for a column | Select \* from customer where city IN (‘paris’, ‘London’); |
| BETWEEN | Between a certain range | Select \* from customer where price between 20 and 30; |

**AND, OR and NOT:**

* The WHERE clause can be combined with AND, OR, and NOT operators.
* The AND and OR operators are used to filter records based on more than one condition:
* The OR operator displays a record if **any** of the conditions are TRUE.

SELECT column1, column2, ... FROM table\_name WHERE condition1 OR condition2 OR condition3 ...;

* The AND operator displays a record if **all** the conditions are TRUE.

SELECT column1, column2, ... FROM table\_name WHERE condition1 AND condition2 AND condition3 ...;

* The NOT operator displays a record if the condition(s) is NOT TRUE.

SELECT column1, column2, ... FROM table\_name WHERE NOT condition;

* SELECT DISTINCT: it is used to return only distinct (different) values.

SELECT DISTINCT column1, column2, .. FROM table\_name;

* LIKE: It is used in a WHERE clause to search for a specified pattern in a column.

There are two wildcards often used in conjunction with the LIKE operator: -

* The percent sign (%) represents zero, one, or multiple characters.
* The underscore sign (\_) represents one, single character

WHERE CustomerName LIKE 'a%' :: Finds any values that start with "a"

WHERE CustomerName LIKE '%a' :: Finds any values that end with "a"

WHERE CustomerName LIKE '%or%' :: Finds any values that have "or" in any position

WHERE CustomerName LIKE '\_r%' :: Finds any values that have "r" in the second position

WHERE CustomerName LIKE 'a\_%' :: Finds any values that start with "a" and are at least 2 characters in length

WHERE CustomerName LIKE 'a\_\_%' :: Finds any values that start with "a" and are at least 3 characters in length

WHERE ContactName LIKE 'a%o' -:: Finds any values that start with "a" and ends with "o"

* IN: Filters results based on a list of values in the WHERE clause.

Select \* from customer where city IN (‘Paris’, ‘London’);

* BETWEEN: Filters results within a specified range in the WHERE clause.

Select \* from customer where price between 20 and 30;

* IS NULL: Checks for NULL values in the WHERE clause.

select \* from customers where email is null;

* AS: Renames columns or expressions in query results.

select first\_name as "First Name", last\_name as "Last Name" from employees;

* ORDER BY: It is used to sort the result in ascending or descending order.

select column1, column2, .. from table\_name order by column1, column2, ... asc | desc;

The ORDER BY keyword sorts the records in ascending order by default. To sort the records in descending order, use the DESC keyword.