# **DBMS Basic Theory :**

### **What is Data ?**

* Data is information that has been translated into a form that is efficient for movement or processing.
* It's like the raw ingredients in cooking, not very useful on their own but essential for making a finished dish.

### **What is a databases ?**

* A database is an organized collection of data, generally stored and accessed electronically from a computer system.
* Think of it as a digital filing cabinet where information is stored in a way that makes it easy to find what you're looking for.

### **What is DBMS ?**

* A Database Management System (DBMS) is the software that interacts with end-users, applications, and the database itself to capture and analyze data.
* It's the software that helps organize, retrieve, and manage data in databases.

### **Relational Database**

This is a type of Database Where data stores in Tabular format known as RDBMS

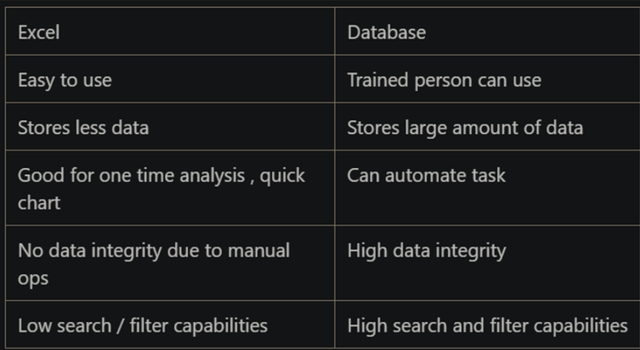
### **Non Relational Database**

This is a type of Database Where data not stores in tabular format. [Documents , Graphs etc , Key value Pairs etc]

### **What is the difference betweeen DBMS vs RDBMS ?**

* The key difference is that RDBMS (Relational Database Management System) stores data in a tabular form, whereas DBMS may not.
* RDBMS also supports a structure that allows for the easy relationship of tables with keys, which isn't necessarily a feature in DBMS.

### **Excel vs Database**

****

### **Tables**

In a relational database, tables are collections of related data entries and they consist of columns and rows.

### **Rows**

A single, implicitly structured data item in a table, also known as a record.

### **Columns**

A vertical entity in a table that contains all information associated with a specific field within the table's database.

### **Data Types**

Categories for the kind of data that can be stored and manipulated within a DBMS, such as integer, float, text, and datetime.

### **Constrains**

Rules applied to table columns to ensure the integrity and validity of the data within the database. Examples include primary keys, foreign keys, unique, not null, and check constraints.

### **Adding Data**

The process of inserting new data or records into a database.

### **Updating Data**

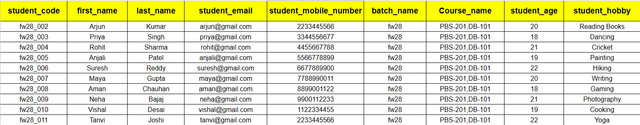
Modifying existing data within a database. This can involve changing existing records, including adding, deleting, or modifying data.

### **Removing Data**

Deleting records or data from a database. This can be done from tables or removing the entire table itself.

### **Filtering Data**

The process of specifying conditions to display only those records that meet those conditions within a database.



Sample data

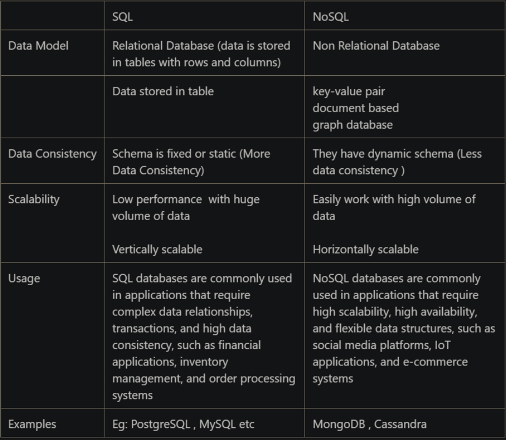
<https://docs.google.com/spreadsheets/d/1bTQnxKRVtmae8Hwm_ul0QUEa_c4XhhaaegAYyihPLUs/edit?gid=0#gid=0>

# **Intro to SQL and MySQL :**

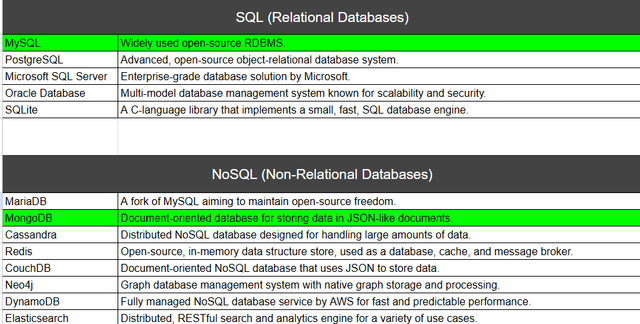
### **What is SQL ?**

* SQL → Structured query Language
* A programming language used to manage and manipulate relational databases
* It is a standard language used to communicate with databases

### **SQL vs NoSQL**

****

### **List of SQL and NoSQL Databases**

****

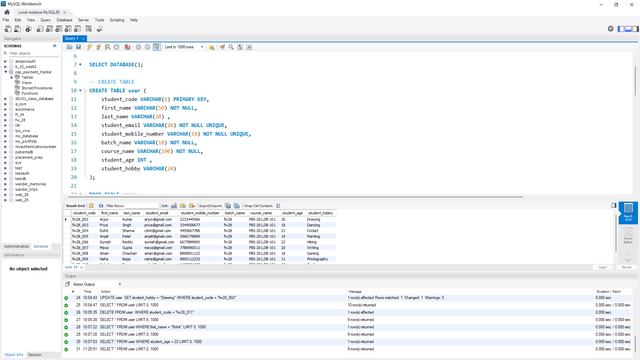
### **MySQL Server**

It is the software component that actually manages the databases and their data.

### **MySQL Workbench (Client)**

* MySQL Clients are the tools or applications that interact with the MySQL Server to manage or manipulate the database.
* It is an GUI Application to write and execute SQL query in MySQL database.

### **My sql work bench work through**

****

### **MySQL Structure :**

* Database → When you create a new sheet in google sheet
* Table → When you create a new tab in google sheet
* Data (Row , Column) → When you add Data in google sheet

### **Relate MySQL with google sheet analogy:**

* Main Google sheet - Database
* Sub-sheet - Table
* Data (row ,column) - Data (row ,column)

### **common operations :**

* CREATE - Database , Table , Data
* READ - Database , Table , Data ( Lets talk little bit more here, E.g. - putting filter in sheet )
* UPDATE - Database ,Table Properties , Table Name , Data
* DELETE - Database , Table , Data
* Others - Lets think here - searching , sorting , filtering etc.

# **Database Level Query**

### **How to see all the database present :**

SHOW DATABASES;

### **How to create a new Database :**

CREATE DATABASE MyFirstDatabase;

#### **Rules for Database Name**

* You can not give any random name to your database
* Database names can be up to 64 characters in length.
* Database names can contain letters (both uppercase and lowercase), numbers, and underscores.
* But, name must start with a letter or an underscore.
* Special characters, spaces, and punctuation marks are not allowed.
* Avoid using MySQL reserved words as database names.
* Database names must be unique within the MySQL instance.
* Here are a few examples of valid database names:
  + mydatabase
  + my\_database
  + mydb123
  + user\_data
* And some examples of invalid database names:
  + 123database (starts with a number)
  + my database (contains a space)
  + my-database (contains a hyphen)
  + SELECT (a reserved word)

### **How to go inside a Database :**

USE database\_name;

### **How to see which database you are in :**

SELECT DATABASE();

# **Table Level Query**

### **How to see all the table present in a database ?**

SHOW TABLES;

### **How to create a new tables**

CREATE TABLE table\_name (

column\_name data\_type constraints ,

column\_name data\_type constraints ,

)

#### **Column Name Rules :**

* Column names should start with a letter (A-Z or a-z) or an underscore (\_).
* After the initial character, column names can consist of letters, numbers (0-9), or underscores.
* Column names can be up to 64 characters long,
* Avoid using reserved words (e.g., keywords used by MySQL) as column names.
* Column names must be unique within a table. You cannot have two columns with the same name in the same tables\*\*.\*\*

#### **Data Types in Table :**

INT : Used to store whole numbers.

CREATE TABLE employees (

employee\_id INT PRIMARY KEY,

age INT

);

VARCHAR(size): Used for variable-length character strings.

CREATE TABLE students (

student\_id INT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50)

);

CHAR(size): Used for fixed-length character strings

CREATE TABLE products (

product\_id INT PRIMARY KEY,

product\_code CHAR(10),

product\_name VARCHAR(100)

);

DATE: Used to store dates in 'YYYY-MM-DD' format.

CREATE TABLE orders (

order\_id INT PRIMARY KEY,

order\_date DATE

);

TIME: Used to store time values in 'HH:MM:SS' format.

CREATE TABLE appointments (

appointment\_id INT PRIMARY KEY,

appointment\_time TIME

);

DATETIME : Used to store both date and time values in 'YYYY-MM-DD HH:MM:SS' format.

CREATE TABLE events (

event\_id INT PRIMARY KEY,

event\_datetime DATETIME

);

DECIMAL(precision, scale): Used to store fixed-point numbers with a specified number of digits (precision) and a specified number of decimal places (scale).

CREATE TABLE accounts (

account\_id INT PRIMARY KEY,

balance DECIMAL(10, 2)

);

BOOLEAN: Used to store true/false or 1/0 values.

CREATE TABLE users (

user\_id INT PRIMARY KEY,

is\_active BOOLEAN

);

ENUM: Used to store one of a predefined list of values.

CREATE TABLE colors (

color\_id INT PRIMARY KEY,

color\_name ENUM('Red', 'Green', 'Blue')

);

### **Constraints**

PRIMARY KEY: Ensures that the values in a column (or a set of columns) are unique and not null

CREATE TABLE students (

student\_id INT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50)

);

UNIQUE: Ensures that values in a column (or a set of columns) are unique but allows null values.

CREATE TABLE employees (

employee\_id INT UNIQUE,

email VARCHAR(100) UNIQUE

);

NOT NULL: Requires that a column cannot have null values. Every row must have a value in this column.

CREATE TABLE orders (

order\_id INT PRIMARY KEY,

order\_date DATE NOT NULL

);

AUTO\_INCREMENT: Used with an integer column to automatically generate a unique value for each new row. Typically used with primary keys.

CREATE TABLE employees (

employee\_id INT AUTO\_INCREMENT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50)

);

CHECK: Defines a condition that values in a column must satisfy. It is used to enforce data integrity rules.

CREATE TABLE products (

product\_id INT PRIMARY KEY,

stock\_quantity INT,

CHECK (stock\_quantity >= 0)

);

DEFAULT: Specifies a default value for a column when no value is provided during an insert operation.

CREATE TABLE users (

user\_id INT PRIMARY KEY,

username VARCHAR(50) NOT NULL,

creation\_date DATE DEFAULT CURRENT\_DATE

);

### **How to delete a table (Whole table)**

DROP TABLE table\_name;

### **How to see schema of a table**

DESCRIBE table\_name;

# **Adding data in table (INSERT INTO)**

### **Syntax :**

Adding single Data :

INSERT INTO table\_name

(column1 , column2 , column3 , column4)

VALUES

(value1 , value2 , value3 , value4);

Adding Many Data :

INSERT INTO table\_name

(column1 , column2 , column3 , column4)

VALUES

(value1 , value1 , value1 , value1),

(value2, value2 , value2 , value2 ),

(value3 , value3 , value3 , value3);

#### **Example :**

Ading One Data :

INSERT INTO user (student\_code, first\_name , last\_name , student\_email , student\_mobile\_number , batch\_name , course\_name , student\_age , student\_hobby )

VALUES

("fw28\_002", "Arjun" , "Kumar" , "arjun@gmail.com" ,"2233445566" ,"fw28" ,"PBS-201,DB-101", 20, "Reading Books");

Adding Many Data:

INSERT INTO user (student\_code, first\_name, last\_name, student\_email, student\_mobile\_number, batch\_name, course\_name, student\_age, student\_hobby) VALUES

('fw28\_003', 'Priya', 'Singh', 'priya@gmail.com', '3344556677', 'fw28', 'PBS-201,DB-101', 18, 'Dancing'),

('fw28\_004', 'Rohit', 'Sharma', 'rohit@gmail.com', '4455667788', 'fw28', 'PBS-201,DB-101', 21, 'Cricket'),

('fw28\_005', 'Anjali', 'Patel', 'anjali@gmail.com', '5566778899', 'fw28', 'PBS-201,DB-101', 19, 'Painting'),

('fw28\_006', 'Suresh', 'Reddy', 'suresh@gmail.com', '6677889900', 'fw28', 'PBS-201,DB-101', 22, 'Hiking'),

('fw28\_007', 'Maya', 'Gupta', 'maya@gmail.com', '7788990011', 'fw28', 'PBS-201,DB-101', 20, 'Writing'),

('fw28\_008', 'Aman', 'Chauhan', 'aman@gmail.com', '8899001122', 'fw28', 'PBS-201,DB-101', 18, 'Gaming'),

('fw28\_009', 'Neha', 'Bajaj', 'neha@gmail.com', '9900112233', 'fw28', 'PBS-201,DB-101', 21, 'Photography'),

('fw28\_010', 'Vishal', 'Desai', 'vishal@gmail.com', '1122334455', 'fw28', 'PBS-201,DB-101', 19, 'Cooking'),

('fw28\_011', 'Tanvi', 'Joshi', 'tanvi@gmail.com', '2233445567', 'fw28', 'PBS-201,DB-101', 22, 'Yoga');

# **Read Data From Table :**

### **Syntax :**

Get All the Data from a Table :

SELECT \* FROM table\_name;

Get only specific colum that you want :

SELECT column\_1 , column\_2 FROM table\_name;

### **Example :**

Getting all column :

SELECT \* FROM user;

Getting Specific Column :

SELECT first\_name , student\_email , course\_name FROM user;

# **Get Filtering Data based on Condition (WHERE)**

### **Syntax**

SELECT \* FROM emp\_data WHERE column\_name = filter\_value;

-- Eg - SELECT \* FROM emp\_data WHERE departments = "Support";

### **Example**

Get all the details where first\_anme is Rohit.

SELECT \* FROM user WHERE first\_name = "Rohit";

Get all the dertails where students\_age is 22

SELECT \* FROM user WHERE student\_age = 22;

Get firat\_name and last\_name where student code is fw28\_011

SELECT first\_name , last\_name FROM user

WHERE student\_code = "fw28\_011";

# **Update Data (UPDATE)**

### **Syntax :**

Update One Details

UPDATE tableName

SET column1 = "value1"

WHERE column = "value"; -- // Choose the primary key that you want to update

Update Many Details

UPDATE tableName

SET column1 = "value1" , column2 = "value2" -- You can update one or more

WHERE column = "value"; -- // Choose the primary key that you want to update

### **Example :**

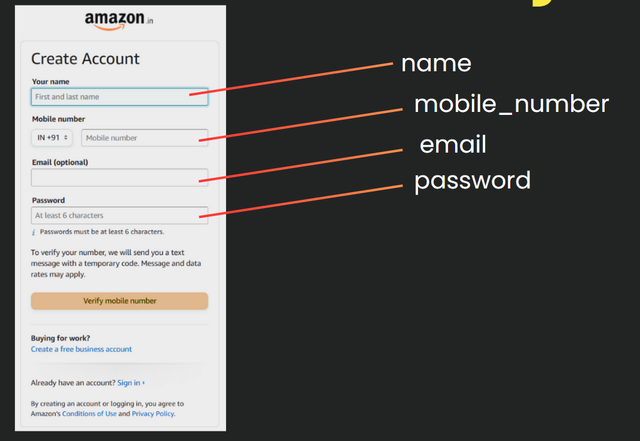
Update student\_hobby to "Drawing" Where student\_code is "fw28\_002"

UPDATE user

SET student\_hobby = "Drawing"

WHERE student\_code = "fw28\_002";

# **Mini Project (Amazon user Details)**

****

### **Creating database and tables**

See all the database

SHOW DATABASES;

Create New database

CREATE DATABASE myAuthenticationSystem;

Use that database

USE myAuthenticationSystem;

Show tables

SHOW TABLES;

Create tables for our user

CREATE TABLE user (

user\_id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL,

mobile\_number VARCHAR(15) NOT NULL,

email VARCHAR(255),

password VARCHAR(255) NOT NULL

);

### **Adding Data → Think this will run when someone registering**

Add One data

INSERT INTO user (name, mobile\_number, email, password)

VALUES ('John Doe', '1234567890', 'john.doe@example.com', 'securepassword');

Add Multiple data

INSERT INTO user (name, mobile\_number, email, password)

VALUES

('Alice Smith', '9876543210', 'alice.smith@example.com', 'strongpassword1'),

('Bob Johnson', '5551112233', NULL, 'secretpass'),

('Eva Green', '7890123456', 'eva.green@example.com', 'myp@ssw0rd'),

('David Miller', '4567890123', NULL, 'davidpass'),

('Sophie Williams', '6543210987', 'sophie.w@example.com', 'secure123');

### **Updating the data : (This will run when you want to update something)**

-- Update Eva Green's email and password

UPDATE user

SET email = 'eva.newemail@example.com', password = 'newpassword'

WHERE name = 'Eva Green';

### **Deleting the data : ( This will run when you are delating your account )**

-- Delete user with mobile number 5551112233 (Bob Johnson)

DELETE FROM user

WHERE mobile\_number = '5551112233';

### **Reading Data**

Filter data where name is jhon doe or name is alice smith

SELECT \* FROM user

WHERE name = "John Doe" OR name = "Alice Smith";

Filter data where name is jhon doe and email is test@gmail.com

SELECT \* FROM user

WHERE name = "John Doe" AND email = "test@gmail.com";

## **Resources - Official Documentation and Other Resources**

* MySQL Reference Manual - <https://dev.mysql.com/doc/refman/8.3/en/>
* W3 School Quick Reference - <https://www.w3schools.com/sql/>
* CREATE DATABASE - <https://www.w3schools.com/sql/sql_create_db.asp>
* CREATE TABLE - <https://www.w3schools.com/sql/sql_create_table.asp>
* DATA TYPES - <https://www.w3schools.com/sql/sql_datatypes.asp>
* CONSTRAINTS - <https://www.w3schools.com/sql/sql_constraints.asp>
* INSERT INTO - <https://www.w3schools.com/sql/sql_insert.asp>
* SELECT - <https://www.w3schools.com/sql/sql_select.asp>
* WHERE - <https://www.w3schools.com/sql/sql_where.asp>
* UPDATE - <https://www.w3schools.com/sql/sql_update.asp>
* DELETE - <https://www.w3schools.com/sql/sql_delete.asp>

Class notes

-- Showing all the databases

SHOW DATABASES ;

-- Creating new database

CREATE DATABASE db101 ;

-- Switching to the database db101

USE db101;

-- Creating a table named 'students'

CREATE TABLE students(

roll\_number INT PRIMARY KEY,

firstName VARCHAR(25),

isActive BOOLEAN,

joinDate DATE

);

-- Creating a table named 'test'

CREATE TABLE test(

id INT PRIMARY KEY

);

-- Dropping the table named 'test'

DROP TABLE test;

-- Displaying all the tables in the current database

SHOW TABLES;

-- Inserting records into the 'students' table

INSERT INTO students(roll\_number, firstName, isActive, joinDate)

VALUES (2, 'Ravi', true, '2022-11-14'),

(3, 'Pooja', false, '2021-12-15');

-- Selecting the columns 'firstName' and 'joinDate' from the 'students' table

SELECT firstName, joinDate FROM students;

-- Selecting all columns from the 'students' table

SELECT \* FROM students;

-- Selecting all columns from the 'students' table where 'firstName' is 'pabl'

SELECT \* FROM students WHERE firstName = 'pabl';

-- Updating the 'firstName' to 'Raj' where 'roll\_number' is 1 in the 'students' table

UPDATE students SET firstName = 'Raj' WHERE roll\_number = 1;

-- Deleting records from the 'students' table where 'roll\_number' is 3

DELETE FROM students WHERE roll\_number = 3;