

SQL

DBMS ---> Data can be stored in the form of Tables (Old Tech)

RDBMS (Relational Database Management Sys) ---> Latest Technology

What is Database? ---> Data Storage Area

In order to update, delete or retrieve the data we need to communicate with the Database, for this we use SQL Language.

Database Components:-

- 1) Client ---> We can send SQL Commands using client software.
- 2) Server ---> Data is actually stored in the Database Server (Remote Machines)

Types of client:-

- 1) Graphical Mode(GUI)
- 2) CLI (Command line Interface)

Example:-

IF we install ORacle Database ----> SQLDeveloper (GUI)

SQLPlus (CLI)

Toad, Squirell,

Aquadata studio (3rd party client)

MySQL -----> MySQLWorkbench(GUI client)

MySQL CommandLineTool (CLI client)

Toad, Squirell, Aquadata (3rd party client)

SQL Command Line command

- show databases;
- use world;
- show tables;
- select * from country;

SQL COMMANDS

1) DDL (Data Definition Language)

- CREATE, ALTER, DROP, TRUNCATE, RENAME

2) DML (Data Manipulation Language)

- INSERT, UPDATE, DELETE

3) DRL/DQL (Data Retrieval/Data Query Language)

- SELECT

4) TCL (Transaction Control Language)

- COMMIT, ROLLBACK, SAVEPOINT

5) DCL (Data Control Language) -- mainly used by the administrators

- GRANT, REVOKE

Step 1:-

The very first thing is to Create a Database and Schemas

```
CREATE DATABASE employee;
```

```
DROP DATABASE employee;
```

```
CREATE SCHEMA employee1;
```

```
DROP DATABASE employee1;
```

```
CREATE DATABASE IF NOT EXISTS indianFood;
```

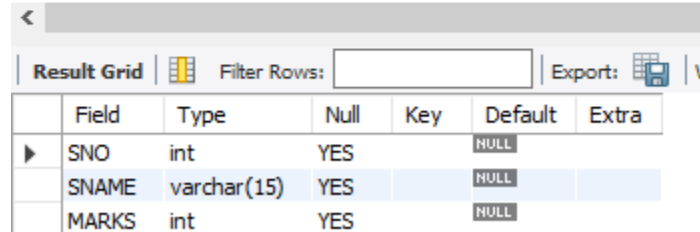
```
DROP DATABASE indianFood;
```

CREATE DATABASE IF NOT EXISTS employee; (this is used when we have lot of databases and check if already the database is there or not, if not it will create)

Step 2:-

The second thing is to Create a Table.

```
1 • use employee;  
2  
3 • CREATE TABLE STUDENT  
4 • (SNO INT(3),  
5   SNAME VARCHAR(15),  
6   MARKS INT(3));  
7  
8 • DESCRIBE STUDENT;
```



	Field	Type	Null	Key	Default	Extra
▶	SNO	int	YES		NULL	
	SNAME	varchar(15)	YES		NULL	
	MARKS	int	YES		NULL	

DESCRIBE STUDENT ---> will show the Grid view of table created with the datatype allotted

Step 3:-

The next thing is to Insert the values into Table.

```
INSERT INTO STUDENT VALUES(101, 'Rohan', 67);  
INSERT INTO STUDENT (SNAME, SNO , MARKS) VALUES('Deepika' ,102,89) ;  
INSERT INTO STUDENT VALUES(103, 'Aman', NULL);
```

Step 4:-

How to select rows from the Table?

```
use sakila;
```

```
SELECT * FROM film;
```

```
SELECT film_id, description, release_year, length, rating FROM film;
```

```
SELECT film_id FID, description DES, release_year REL, length LEN , rating RAT FROM film
```

WHERE Clause

WHERE ---> It is used for selecting the rows based on condition.

DISTINCT ---> It is used to filter out the duplicate entries from the DATABASE

```
SELECT * FROM city;
```

```
SELECT * FROM city WHERE CountryCode ='AFG';
```

```
SELECT * FROM city WHERE Population < 300000;
```

```
SELECT * FROM city WHERE Name = 'Dubai';
```

```
SELECT * FROM city WHERE District is null;
```

DISTINCT Clause

```
SELECT CountryCode FROM city;
```

```
SELECT DISTINCT CountryCode FROM city;
```

```
SELECT DISTINCT * FROM city;
```

Logical Operator

```
SELECT * FROM countrylanguage;
```

```
SELECT * FROM countrylanguage WHERE IsOfficial = 'T' AND Percentage > 5;
```

```
SELECT * FROM countrylanguage WHERE Language = 'Spanish' OR IsOfficial = 'F';
```

```
SELECT * FROM countrylanguage WHERE NOT CountryCode = 'ABW';
```

BETWEEN & IN Operator

BETWEEN ---> Used to display the rows which is following in the range values

IN ----> IN Operator returns the rows when the values are matching in the list

```
SELECT * FROM countrylanguage WHERE Percentage > 25 AND Percentage <= 50;
```

```
SELECT * FROM countrylanguage WHERE Percentage BETWEEN 25 AND 50;
```

```
SELECT * FROM countrylanguage WHERE Percentage NOT BETWEEN 25 AND 50;
```

```
SELECT * FROM countrylanguage WHERE Percentage= 86.2 OR Percentage= 96.8 OR Percentage= 2.6;
```

```
SELECT * FROM countrylanguage WHERE Percentage IN (86.2, 96.8, 2.6);
```

```
SELECT * FROM countrylanguage WHERE Percentage NOT IN (86.2, 96.8, 2.6);
```

Pattern Matching Operators (%, _)

% ---> Many characters

_ ---> Single Characters

```
SELECT * FROM city;
```

```
SELECT * FROM city WHERE Name LIKE 'G%';
```

```
SELECT * FROM city WHERE Name LIKE '%s';
```

```
SELECT * FROM city WHERE Name LIKE 'M%h';
```

```
SELECT * FROM city WHERE Name LIKE '%t%';
```

```
SELECT * FROM city WHERE Name LIKE '%e_';
```

```
SELECT * FROM city WHERE Name LIKE '____';
```

```
SELECT * FROM city WHERE Name NOT LIKE 'T%';
```

```
|
```

DDL Operations

1. CREATE

```
use employee;

CREATE TABLE STUDENT
(SID INT(4), SNAME VARCHAR(13), MARKS INT(4));

DESCRIBE STUDENT;

INSERT INTO STUDENT VALUES (101, 'Brijesh' , 85 );
INSERT INTO STUDENT VALUES (102, 'Monika' , 47 );
INSERT INTO STUDENT VALUES (103, 'Rohan' , 96 );
INSERT INTO STUDENT VALUES (104, 'Anamika' , 80 );

COMMIT;

SELECT * FROM STUDENT;
```

2. ALTER

```
ALTER TABLE STUDENT ADD (GRADE INT(3));
DESCRIBE STUDENT;

ALTER TABLE STUDENT DROP COLUMN MARKS;

ALTER TABLE STUDENT MODIFY COLUMN SNAME VARCHAR(25);

ALTER TABLE STUDENT RENAME COLUMN SNAME TO STUD_NAME;
```

3. DROP , TRUNCATE & DELETE (DML command)


```
SET autocommit=0;
```

```
SET SQL_SAFE_UPDATES = 0;
```

```
SELECT * FROM STUDENT;
```

```
DELETE FROM STUDENT;  
Commit;
```

```
ROLLBACK;
```

```
-----  
INSERT INTO STUDENT VALUES (101, 'Neha', 15);  
INSERT INTO STUDENT VALUES (102, 'Rohgan', 78);  
INSERT INTO STUDENT VALUES (103, 'Pickahcy', 45);  
INSERT INTO STUDENT VALUES (104, 'Qrtyub', 86);  
Commit;
```

```
TRUNCATE TABLE STUDENT;
```

```
-----  
DROP TABLE STUDENT;
```

RENAME command

```
use world;
```

```
SELECT * FROM CITY;
```

```
RENAME TABLE countrylanguage to Languages;
```

BUILT-in Functions in MySQL

1. STRING Functions

```
Use world;
```

```
SELECT * FROM city;
```

```
SELECT UPPER('Kabul');
```

```
SELECT UPPER(Name) FROM city;
```

```
SELECT LOWER(Name) FROM city;
```

```
-----  
SELECT LENGTH('Amsterdam');
```

```
SELECT LENGTH(Name) FROM city;
```

```
3 *****Print name of City who have only 4 characters*****  
  ^
```

```
SELECT * FROM city WHERE LENGTH(Name)=4;
```

```
-----  
• SELECT TRIM('      Welcome      ');
```

```
• SELECT TRIM('z' FROM 'zzzzAbuDhabhizzzz');
```

```
-----  
3 **** INSTR() ---> Returns the position of the character in the string  
  ^
```

```
SELECT INSTR('Welcome','e');
```

```
• SELECT INSTR('Oracle','c');
```

**** SUBSTR()/SUBSTRING() ---->Returns the substring of the string.

```
SELECT SUBSTR('ORACLE' ,2 ,3 )
SELECT SUBSTR('APACHE MAVEN' , 4 , 8)

SELECT SUBSTRING('ORACLE' ,2 ,3 )
SELECT SUBSTRING('APACHE MAVEN' , 4 , 8)

SELECT SUBSTRING(Name,1,3) FROM city;
```

```
-----

SELECT CONCAT( 'ORACLE', 'Java');

SELECT CONCAT(Name,District) FROM city;

-----
```

2. Numeric Functions

```
SELECT ABS(-40);
SELECT ABS(40);

SELECT SQRT(25);

SELECT MOD(8,3);

SELECT POWER(2,5);

SELECT TRUNCATE(40.43231, 3);
SELECT TRUNCATE(78.7663, 2);

SELECT TRUNCATE(6463223, -4);
SELECT TRUNCATE(2434647, -2);
SELECT TRUNCATE(88676656756, -9);
```

3. DATE Functions

```
SELECT CURDATE();  
SELECT CURRENT_DATE();
```

```
SELECT CURTIME();  
SELECT CURRENT_TIME();
```

```
SELECT NOW();
```

```
SELECT SYSDATE();
```

```
SELECT MONTH("2023-08-15");  
SELECT Year("2023-08-15");  
SELECT Day("2023-08-15");
```

4. Aggregate Functions

It works on single column

```
use world;
```

```
SELECT * FROM city;
```

```
SELECT AVG(Population) FROM city;  
SELECT SUM(Population) FROM city;  
SELECT MAX(Population) FROM city;  
SELECT MIN(Population) FROM city;
```

GROUP BY , HAVING & ORDER BY Clause



10

11 • `SELECT CountryCode, SUM(Population) FROM city GROUP BY CountryCode;`

12 • `SELECT CountryCode, AVG(Population) FROM city GROUP BY CountryCode;`

13 • `SELECT CountryCode, COUNT(*) FROM city GROUP BY CountryCode;`

14

<		
Result Grid		
Filter Rows: <input type="text"/>		
Export: 		
Wrap Cell Content: 		
Fetch rows: <input type="text"/>		
	CountryCode	SUM(Population)
▶	ABW	29034
	AFG	2332100
	AGO	2561600
	AIA	1556
	ALB	270000
	AND	21189
	ANT	2345
	ARE	1728336
	ARG	19996563
	ARM	1633100
	ASM	7523

```
SELECT CountryCode, District, COUNT(*) FROM city GROUP BY CountryCode, District HAVING COUNT(*)>5 ;
SELECT CountryCode, District, COUNT(*) FROM city GROUP BY CountryCode, District HAVING COUNT(*) <=5 ;
```

```
SELECT CountryCode, District, COUNT(*) FROM city WHERE CountryCode<> 'AFG'
GROUP BY CountryCode, District HAVING COUNT(*) <=5 ;
```

```
SELECT CountryCode, District, COUNT(*) FROM city WHERE CountryCode = 'NLD'
GROUP BY CountryCode, District HAVING COUNT(*) <=5 ;
```

ORDER BY

```
SELECT * FROM City ORDER BY Name;
```

```
SELECT * FROM City ORDER BY CountryCode;
```

```
SELECT * FROM City ORDER BY Population DESC;
```

```
SELECT CountryCode, District, COUNT(*) FROM city WHERE CountryCode = 'NLD'  
GROUP BY CountryCode, District HAVING COUNT(*) <=5 ORDER BY CountryCode;
```

```
SELECT CountryCode, District, COUNT(*) FROM city WHERE CountryCode = 'NLD'  
GROUP BY CountryCode, District HAVING COUNT(*) <=5 ORDER BY COUNT(*);
```

```
SELECT CountryCode, District, COUNT(*) FROM city  
WHERE CountryCode = 'NLD'  
GROUP BY CountryCode, District  
HAVING COUNT(*) <=5  
ORDER BY CountryCode;
```

```
SELECT CountryCode, District, COUNT(*) FROM city  
WHERE CountryCode = 'NLD'  
GROUP BY CountryCode, District  
HAVING COUNT(*) <=5  
ORDER BY COUNT(*);
```

SET Operators

UNION ---> In UNION it will print only the data that is distinct i.e no duplication

The UNION Operator is used to combine the result-set of two or more SELECT statements

Each SELECT statement with UNION must have the same number of columns

The Columns must also have similar data types

The columns in each SELECT Statement must also be in the same order

```
1 * SELECT * FROM A;  
2 * SELECT * FROM B;  
3  
4 * SELECT NUM FROM A UNION SELECT NUM FROM B;  
5
```

T

NUM
10
11
12
14
13
15

UNION ALL --> In UNION ALL it will print all the data whether it is having duplication or not

```
1 * SELECT * FROM A;  
2 * SELECT * FROM B;  
3  
4 * SELECT NUM FROM A UNION ALL SELECT NUM FROM B;  
5
```

NUM
10
11
12
14
13
15

