1) Create:-

Create keyword is used to create new database objects such as tables, views, indexes, or databases. The most common usage is ~~created~~ CREATE TABLE that defines the table's structure.

i) Table:-

Syntax:-

```
CREATE TABLE TABLE_NAME ( column1 datatype constraint,
                          column2 datatype constraint,
                          :
                          columnN datatype constraint);
```

Ex:-

```
CREATE TABLE Students ( rollno int primary key,
                        name Varchar(30) not null,
                        email Varchar(40));
```

ii) View:-

Syntax:-

```
CREATE VIEW VIEW_NAME AS SELECT Col1, Col2, ... colN
FROM TABLE_NAME WHERE condition;
```

Ex:-

```
CreaCREATE VIEW students_above_avg AS SELECT * FROM
Students WHERE marks >= 70;
```

iii) Index :-

Syntax :-

```
CREATE INDEX Index_name ON Table_name (Column1, Column2);
```

Ex:-

```
CREATE INDEX idx_lastname ON Employee (last_name);
```

iv) Constraints :-

→ There are various types of constraints in SQL and are given as follows,

1) NOT NULL :-

Ex:-

```
CREATE TABLE Employees ( ID INT NOT NULL,
                         Name Varchar(50));
```

2) UNIQUE :-

Ex:-

```
CREATE TABLE Users ( Email Varchar(100) UNIQUE);
```

3) PRIMARY KEY :-

Ex:-

```
CREATE TABLE Student (Rollnum INT Primary key,
                      Name Varchar(50));
```

4) FOREIGN KEY :-

Ex:-

```
CREATE TABLE Student ( Rollno INT Primary key,
                       Name Varchar(30),
                       Teacher Varchar(30),
                       FOREIGN KEY (Teacher) REFERENCES
                       Courses (Teacher));
```

5) CHECK :-

Ex:-

```
CREATE TABLE Products ( Price Decimal(10,2) CHECK(price>0));
```

6) DEFAULT :-

1) ~~INT~~ Ex:-

```
CREATE TABLE Account ( status Varchar(10) Default 'Active');
```

v) Datatypes :-

1) INT :-

Ex:-

```
CREATE TABLE Employees (EMP_id INT Primary Key,
                        Age INT);
```

2) BIGINT :-

Ex:-

```
CREATE TABLE Transactions ( Transaction_id BIGINT Primary Key,
                            Amount BIGINT);
```

3) Decimal / Numeric :-

Ex:-

```
CREATE TABLE Products ( Price Decimal(10, 2));
```

4) float :-

Ex:-

```
CREATE TABLE Measurements ( Length FLOAT,
                            Breadth FLOAT);
```

5) CHAR(n) :-

Ex:-

```
CREATE TABLE Codes ( Countrycode CHAR(3));
```

6) Varchar(n) :-

Ex:-

```
CREATE TABLE Customers ( Name Varchar(20));
```

7) TEXT:-

Ex:-

```
CREATE TABLE Articles ( Content TEXT);
```

8) DATE:-

```
Create table Login_history ( LoginDate DATE);
```

9) TIME:-

Ex:-

```
Create table Uploads ( Upload_time TIME);
```

10) DATETIME:-

Ex:-

```
CREATE TABLE Logs(
        Entry_time DATETIME,
        Exit_time DATETIME);
```

Copy table Data :-

Ex:-

```
CREATE TABLE Students2 AS SELECT * FROM Students;
INSERT INTO Students2 VALUES (SELECT * FROM students);
```

vii) Copy table structure:-

Ex:-

```
CREATE TABLE Students2 LIKE Students;
```

Alter :-

i) Add Column :-

Ex:-

```
ALTER TABLE Students add column1 Varchar(50);
```

ii) Modify Column :-

Ex:-

```
ALTER TABLE Students MODIFY dept Varchar(5);
```

iii) DROP :-

```
ALTER TABLE Students DROP phone_no;
```

iv) Add Constraint :-

Ex:-

```
ALTER TABLE students ADD CONSTRAINT unique_email
UNIQUE (email_id);
```

v) MODIFY constraints :-

Ex :-

```
ALTER TABLE Students MODIFY phone_no INT NOT NULL;
```

vi) DROP Constraints :-

Ex :-

```
ALTER TABLE Students DROP CONSTRAINT PK_student;
```

Insert :-

Ex :-

```
INSERT INTO students values (128, "Gowrish", 18);
```

Update :-

Ex :-

```
UPDATE students set marks = marks + 8;
```

DElete :-

Ex :-

```
DELETE FROM Employees WHERE Dept = "Sales";
```

TRUNCATE :-

Ex :-

```
TRUNCATE TABLE Employees;
```

Show :-

Ex :-

```
SHOW databases;
SHOW tablos;
```

Use :-

Ex :-

```
Use students;
```

WHERE :-

Ex :-

```
SELECT * FROM students WHERE Marks >= 75;
```

LIKE :-

Ex :-

```
SELECT * FROM students WHERE Name LIKE "G%";
```

LIMIT :-

Ex :-

```
SELECT * FROM Employees Limit 5;
```

Between :-

Ex :-

```
SELECT * FROM Employees WHERE Salary between 35000 and
45000;
```

DESC :-

Ex :-

```
DESC students;
```

(or)

```
SELECT * FROM students Order by marks DESC;
```

NOT :-

Ex :-

```
SELECT * FROM Employees WHERE NOT dept='Sales';
```

AND :-

Ex:-

```
SELECT * FROM Employees WHERE dept = 'IT' AND Salary >
                                                500000;
```

OR :-

Ex :-

```
SELECT * FROM EMP Employees WHERE dept = "IT" OR dept = "ECE";
```

Group by :-

Ex :-

```
SELECT dept FROM students Group by dept;
```

Having :-

Ex :-

```
SELECT * FROM students Group by dept having marks > 95;
```

Order by :-

Ex :-

```
SELECT Name, Salary FROM Employees Order by Salary DESC;
```

AS :-

Ex :-

```
SELECT first_name AS name, salary FROM Employees AS emp;
```

IN :-

Ex :-

```
SELECT * FROM students WHERE dept in ('IT', 'CSE', 'ECE');
```

NOT IN :-

Ex :-

```
SELECT * FROM Employee WHERE dept NOT IN("ME", "CIVIL");
```

Exists :-

Ex :-

```
SELECT * FROM dept d WHERE Exists(SELECT 1 FROM Employee e
  WHERE e.dept_id = d.dept_id);
```

NOT Exists :-

Ex :-

```
SELECT Name FROM Customers c WHERE NOT Exists(
 SELECT 1 FROM Orders o WHERE o.customer_id = c.customer_id);
```

Operators :-

Arithmetic operators :-

+, -, *, /, %

Ex :-

```
SELECT marks + 5 AS updated_marks FROM student;
```

ii) Relational operators :-

=, >, <, >=, <=, !=

Ex :-

```
SELECT * FROM Student WHERE marks > 60;
```

iii) Logical operators :-

AND OR NOT

Ex :-

```
SELECT * FROM Studen WHERE marks > 95 AND Age > 15;
```

Functions :-

i) Aggregate functions :-

Count(), Sum(), Avg(), Max(), Min().

Ex :-

```
SELECT count(marks) FROM student;
SELECT Sum(marks) FROM student;
SELECT Avg(marks) FROM student;
SELECT Max(marks) FROM student;
SELECT Min(marks) FROM student;
```

ii) DATE functions :-

Ex :-

```
SELECT curDate();
SELECT Now();
SELECT DATE('2026-3-14 09:03:15');
SELET DAY ('2026-03-14');
```

iii) STRING functions :-

Function	Definition
upper()	converts string to upper case.
lower()	converts string to lower case.
length()	returns length
concat()	joins two strings.
substring	Extracts a part of a string.
trim()	Removes spaces.

iv) Mathematical functions :-

Function	Define
ABS(), CEIL(), FLOOR(), ROUND(), MOD()	

Ex:-

```
SELECT ABS(-12);
SELECT CEIL(4.3);
SELECT FLOOR(4.8);
SELECT Round(4.56, 1);
SELECT Mod(10, 3);
```

JOINS :-

i) Set operators :-

Union, Union all, intersect, set difference(-)

Ex :-

```
SELECT *name, rollno FROM Students
        Union
SELECT  name, rollno FROM Courses;

SELECT  name, rollno  FROM Students
        INTERSECT
SELECT  name, rollno  FROM Courses;

SELECT  name, rollno  FROM Students
        —
SELECT  name, rollno  FROM Courses;
```

Set Quantifiers :-

ALL, DISTINCT,

Ex :-

```
SELECT ALL dept FROM Employees;

SELECT DISTINCT dept FROM employees;
```

———— X ————