**Syntax**

**1] Create Database**

Create database database\_name

**2] Create Table**

Create table xyz ( id int, name varchar(20), age int);

**3] Alter:**

A] Add New Column

ALTER TABLE table\_name ADD column\_name datatype;

B] Rename column

ALTER TABLE table\_name RENAME COLUMN old\_name TO new\_name;

C] Modify Datatype or size

ALTER TABLE emp MODIFY sal DECIMAL(12,2);

D] Drop Column

ALTER TABLE table\_name DROP COLUMN column\_name;

E] Rename Table

ALTER TABLE old\_table\_name RENAME TO new\_table\_name;

F] Add and drop constraint

ALTER TABLE table\_name

ADD CONSTRAINT constraint\_name constraint\_type (column\_name);

ALTER TABLE table\_name DROP CONSTRAINT constraint\_name;

**4] Drop & Truncate**

A] Drop Database database\_name

B] Drop table table\_name

C] Truncate table table\_name

**5] Rename Column/Table/Constraint**

RENAME old\_table\_name students TO new\_table\_name;

**6] Insert**

A] insert into table\_name values(10, ‘name’, ‘name’);

B] insert into dept (deptno, dname, loc) VALUES (50, 'hhh', 'ewww');

**7] Update**

update emp SET sal = 2000 where empno = 7369;

**8] Delete**

Delete from table\_name where condition;

**9] Select**

A] Select \* from table name

**10] where**

Select \* from where id = 1;

**11] Distinct**

SELECT DISTINCT column\_name FROM table\_name;

**12] Order by**

SELECT \* FROM table\_name ORDER BY col\_name ASC;

**13] Limit**

SELECT \* FROM table\_name LIMIT 5;

SELECT \* FROM dept LIMIT 2, 5; skip first 2 rows and take next 5 record

**14] Grant and Revoke**

GRANT SELECT, INSERT ON dept TO 'user1'@'localhost';

REVOKE INSERT ON dept FROM 'user1'@'localhost';

**15] AND OR NOT**

select \* from students where age > 18 AND gender = 'F';

select \* from students where age < 18 OR gender = 'M';

select \* from students where NOT age = 20;

**16] Between**

select \* from students where age BETWEEN 18 AND 25;

**17] IN**

Select \* from students WHERE name IN ('Alice', 'Bob', 'Charlie');

**18] Like**

Select \* from students WHERE name LIKE 'A%';

‘A%’ start with A, ‘%n’ ends with n, ‘%by%’ contain by everywhere

**19] IS NULL / IS NOT NULL**

SELECT \* FROM students WHERE email IS NULL;

SELECT \* FROM students WHERE email IS NOT NULL;

**20] Count Sum Avg Min Max**

SELECT COUNT(\*) FROM students; give all rows count include null also

SELECT COUNT(studname) FROM students; for specific column

SELECT SUM(age) FROM students;

SELECT AVG(age) FROM students;

**21] Upper Lower Length**

SELECT UPPER(name) FROM students;

SELECT UPPER(ename) AS emp\_name\_upper FROM emp;

SELECT LOWER(name) FROM students;

SELECT name, LENGTH(name) FROM students;

**22] Substring**

SELECT SUBSTRING('HelloWorld', 1, 5);

SELECT SUBSTRING(name, 1, 3) FROM students;

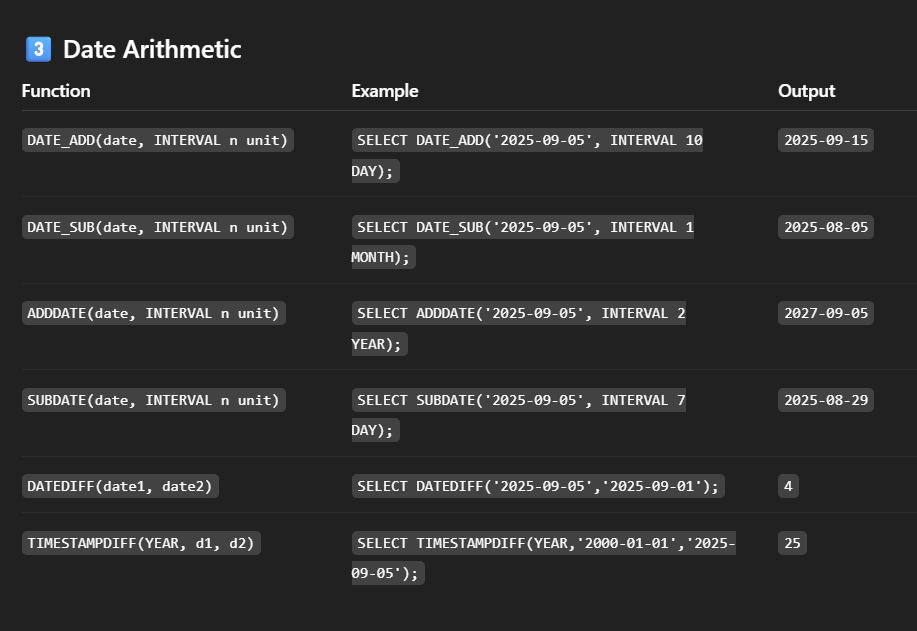
**23] Concat**

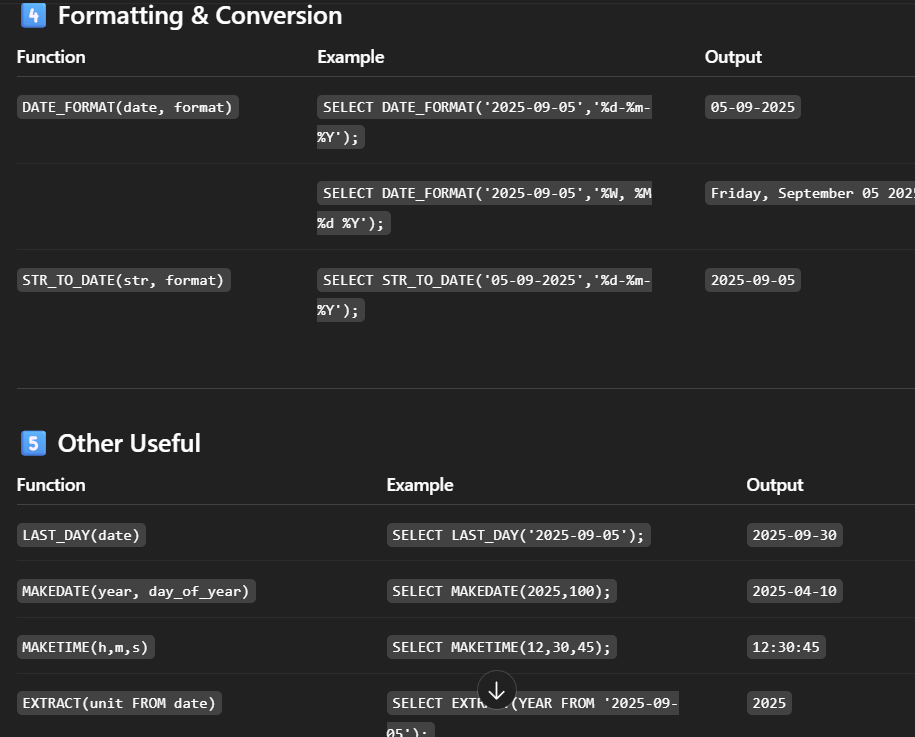
**S**ELECT CONCAT('Hello', ' ', 'World');

SELECT CONCAT(name, ' is ', age, ' years old') FROM students;

**24] Date**





****

**25] Cast Convert**

SELECT CAST('2025-06-12' AS DATE);

SELECT CONVERT('2025-06-12', DATE);

**27] Group By**

Select mode, count(amount) as total

From payment

Group by mode

Having count(amount) >=3

Order by total desc;

**28] Joins (Need atleast 1 same column)**

Inner Join

Select \* from customer AS C

Inner join payment as p on c.customer.id = p.customer id

Left Join

Select \* from customer AS C

left join payment as p on c.customer.id = p.customer id

Full outer join

Self-Join

Select t1.empname as ename, t2.empname as mgrname

From emp as t1

join emp as t2

on t2.empid = t1.manager\_id

**29] Union / Union All**

SELECT deptno FROM emp UNION SELECT deptno FROM dept;

SELECT deptno FROM emp UNION ALL SELECT deptno FROM dept;

**30] Except**

Returns rows from the first query that are not in the second query.

SELECT deptno FROM emp EXCEPT SELECT deptno FROM dept;

**31] Not Null / Unique / Default / Check**

A] CREATE TABLE users (username VARCHAR(50) NOT NULL);

B] CREATE TABLE employees (email VARCHAR(100) UNIQUE);

C] CREATE TABLE orders (status VARCHAR(20) DEFAULT 'Pending');

D] CREATE TABLE accounts (balance INT CHECK (balance >= 0));

**32] Primary Key**

CREATE TABLE students (id INT PRIMARY KEY, name VARCHAR(50));

**33] Foreign Key**

CREATE TABLE enrolments (student\_id INT, FOREIGN KEY (student\_id) REFERENCES students(id));

34]

**38] Indexes**

Cluster index created automatically with primary key

Cluster index create INDEX index\_name on table\_name (col1,col2);

Show index from users;

**39] Virtual Table**

Create VIEW view\_name as Select col1,col2 from table\_name where condition;

Show full tables where table type = ‘view’ ;

CREATE OR REPLACE VIEW female\_students As SELECT name, age, department FROM students WHERE gender = 'F';

**40] User Grant Revoke**

Create user ‘user\_name’@’localhost’ identified by ‘1223’;

Grant select on sample.\* to ‘user\_name’@’localhost’;

REVOKE SELECT ON emp FROM user1;

**41] Being Commit Rollback Savepoint**

BEGIN; -- or START TRANSACTION

START TRANSACTION;

UPDATE emp SET sal = sal + 500 WHERE deptno = 10;

SAVEPOINT sp1;

UPDATE emp SET sal = sal + 1000 WHERE deptno = 20;

ROLLBACK TO sp1; -- undo dept 20 update, keep dept 10 update

COMMIT; -- save dept 10 changes permanently