**SQL:**

**Basic Queries:**

**--TO CREATE A DATABASE**

CREATE DATABASE LOGICFIRST; -- creates a new database

**-- TO DELETE A DATABASE**

DROP DATABASE LOGICFIRST;

DROP SCHEMA LOGICFIRST; -- same as above. u can use DATABASE Or SCHEMA

DROP SCHEMA IF EXISTS LOGICFIRST; -- prevents error if db not found

SHOW DATABASES; -- shows all the databases

SHOW SCHEMAS; -- same as above. shows schemas/db

USE SYS; -- uses this database for all further commands

SHOW TABLES;-- shows all tables in the database being used

**Table - Create,Delete,Alter**

**primary key - uniquely identifies a row in a table**

//creating a table

CREATE TABLE student(

id INT PRIMARY KEY,

name VARCHAR(30),

gpa DECIMAL(3,2)

);

-- ----or-----

CREATE TABLE student(

id INT,

name VARCHAR(30),

gpa DECIMAL(3,2),

PRIMARY KEY(id)

);

DROP TABLE student; -- drops the table

DESCRIBE student; -- describes the columns in the table student

ALTER TABLE student ADD department VARCHAR(5); -- Adds a new column department to the student table

ALTER TABLE student DROP COLUMN department; -- drops the department column from student table

-- ---or---

ALTER TABLE student DROP department; -- same as above

**INSERTING DATE IN A TABLE**

INSERT INTO student VALUES(1,"Aarthi",7.6);

INSERT INTO student VALUES(2,"Anitha",8.5); -- inserts a row. give values in column order

INSERT INTO student VALUES

(3,"Anitha",8.5),

(4,"Arul",8.2),

(5,"Ashwin",7.6); -- inserts more than one row

INSERT INTO student(id,name) VALUES(5,"Balaji"),(6,"Chandru"); -- inserts specific columns.

**SELECT**

SELECT \* FROM student; -- displays all rows and columns in the student table

SELECT id,name FROM student; -- displays specific columns

**Employee table:**

CREATE TABLE employee (

emp\_id INT PRIMARY KEY,

ename VARCHAR(30),

job\_desc VARCHAR(20),

salary INT );

INSERT INTO employee VALUES(1,'Ram','ADMIN',1000000);

INSERT INTO employee VALUES(2,'Harini','MANAGER',2500000);

INSERT INTO employee VALUES(3,'George','SALES',2000000);

INSERT INTO employee VALUES(4,'Ramya','SALES',1300000);

INSERT INTO employee VALUES(5,'Meena','HR',2000000);

INSERT INTO employee VALUES(6,'Ashok','MANAGER',3000000);

INSERT INTO employee VALUES(7,'Abdul','HR',2000000);

INSERT INTO employee VALUES(8,'Ramya','ENGINEER',1000000);

INSERT INTO employee VALUES(9,'Raghu','CEO',8000000);

INSERT INTO employee VALUES(10,'Arvind','MANAGER',2800000);

INSERT INTO employee VALUES(11,'Akshay','ENGINEER',1000000);

INSERT INTO employee VALUES(12,'John','ADMIN',2200000);

INSERT INTO employee VALUES(13,'Abinaya','ENGINEER',2100000);

**Syntax**:

SELECT column1, column2, ...

FROM table\_name

WHERE condition;

**Following can be used within the condition.**

|  |  |
| --- | --- |
| = | Equal |
| > | Greater than |
| < | Less than |
| >= | Greater than or equal |
| <= | Less than or equal |
| <> | Not equal. Note: In some versions of SQL this operator may be written as != |
| BETWEEN | Between a certain range |
| LIKE | Search for a pattern |
| IN | To specify multiple possible values for a column |
| NOT | negation |

**AND/OR can be used to combine the relational operators.**

SELECT \* FROM employee

WHERE ename = 'Ramya';

SELECT emp\_id,ename,salary FROM employee

WHERE salary>2000000;

SELECT emp\_id,ename,salary FROM employee

WHERE salary<2600000 AND job\_desc = 'MANAGER';

SELECT \* FROM employee

WHERE job\_desc='ADMIN' OR job\_desc='HR'; -- though this works next command is a much better choice

SELECT \* FROM employee

WHERE job\_desc IN ('ADMIN','HR');

SELECT \* FROM employee

WHERE job\_desc NOT IN ('MANAGER','CEO');

SELECT \* FROM employee

WHERE salary BETWEEN 2000000 AND 3000000

LIMIT 5; --limits the records shown

SELECT \* FROM employee

LIMIT 5; -- Different syntax in oracle/sql server

**Using Like and wildcards**

LIKE is used with WHERE clause for searching a specific pattern in a column. It is used along with the following wild cards

% represents zero or more characters

\_ represents exactly one character

SELECT \* FROM employee

WHERE ename LIKE 'A%'; -- filters name starting with A

SELECT \* FROM employee

WHERE ename LIKE 'R%a'; -- filters name starting with R and ending with a

SELECT \* FROM employee

WHERE ename LIKE '%I%'; -- filters name containing I

SELECT \* FROM employee

WHERE ename LIKE '\_\_I%'; -- filters name with i as third character

SELECT \* FROM employee

WHERE ename LIKE 'R\%'; -- filters name starting with R%. \ is the escape character.

**UPDATE and DELETE:**

UPDATE employee

SET job\_desc = "Analyst"; -- updates all job\_desc of all rows to "Analyst" when safe update not enabled

UPDATE employee

SET job\_desc = "Analyst"

WHERE job\_desc = "Engineer"; -- changes Engineer to Analyst in all applicable rows

UPDATE employee

SET job\_desc = "Analyst"

WHERE emp\_id=1;

DELETE FROM employee; -- deletes all rows

DELETE from employee

WHERE emp\_id = 12;

**DISTINCT**

SELECT DISTINCT job\_desc

FROM employee; -- shows only distinct values without duplicates

**ORDER BY**

SELECT \* FROM employee

ORDER BY salary; -- order by salary asc

**ORDER BY Ascending and Descending**

SELECT \* FROM employee

ORDER BY salary ASC; -- order by salary asc

SELECT \* FROM employee

ORDER BY salary DESC; -- order by salary desc

SELECT \* FROM employee

WHERE job\_desc="MANAGER"

ORDER BY salary DESC; -- order the manager salaries in desc order

SELECT \* FROM employee

ORDER BY job\_desc,ename; -- first sorts by job\_desc and then by ename

SELECT \* FROM employee

ORDER BY (CASE job\_desc -- specific order

WHEN 'CEO' THEN 1

WHEN 'MANAGER' THEN 2

WHEN 'HR' THEN 3

WHEN 'ANALYST' THEN 4

WHEN 'SALES' THEN 5

ELSE 100 END);

**COPY TABLE**

INSERT INTO first\_table\_name [(column1, column2, ... columnN)]

SELECT column1, column2, ...columnN

FROM second\_table\_name;

**FUNCTIONS**

**Every functions** [**https://www.techonthenet.com/mysql/functions/index.php**](https://www.techonthenet.com/mysql/functions/index.php)

**Aggregate functions** [**https://dev.mysql.com/doc/refman/8.0/en/aggregate-functions.html**](https://dev.mysql.com/doc/refman/8.0/en/aggregate-functions.html)

SELECT COUNT(\*) FROM employee; -- total count of entries in the table

SELECT AVG(salary) FROM employee; -- avg salary of all employees

SELECT AVG(salary)

FROM employee

WHERE job\_desc="MANAGER"; -- avg salary of managers

SELECT SUM(salary)

FROM employee

WHERE job\_desc="ANALYST"; -- total salary given to all analysts

SELECT \* FROM employee

WHERE salary = (SELECT MAX(salary)

FROM employee); -- display the employee with

SELECT MIN(salary) FROM employee;

SELECT UCASE(ename),salary

FROM employee; -- uppercase

SELECT ename,CHAR\_LENGTH(ename)

FROM employee;

SELECT ename,CONCAT("Rs.",salary)

FROM employee; -- adds Rs. to the beginning of salary

SELECT ename,CONCAT("Rs.",FORMAT(salary,0))

FROM employee; -- formats the number to add comma. The second argument(0 here) represents digits to round off after decimal

SELECT ename,LEFT(job\_desc,4)

FROM employee; -- returns only the first 4 characters of the ename

**USING DATE**

ALTER TABLE employee ADD COLUMN Hire\_Date DATE; -- adding hire\_date column

UPDATE employee

SET Hire\_Date="2012-10-05"; -- updating hire\_date

UPDATE employee

SET Hire\_Date="2014-10-05"

WHERE job\_desc = "ANALYST"; -- updating hire\_date

SELECT NOW(); -- Current date and time

SELECT DATE(NOW()); -- current date

SELECT CURDATE(); -- current system date

SELECT DATE\_FORMAT(CURDATE(),'%d/%m/%Y'); -- to change the display format. use %d %m and %y or %Y in required format.

SELECT DATEDIFF(CURDATE(),'2020-01-01') DAYS; -- to calculate date difference

SELECT CURDATE() 'start date',

DATE\_ADD(CURDATE(),INTERVAL 1 DAY) 'one day later',

DATE\_ADD(CURDATE(),INTERVAL 1 WEEK) 'one week later',

DATE\_ADD(CURDATE(),INTERVAL 1 MONTH) 'one month later',x

DATE\_ADD(CURDATE(),INTERVAL 1 YEAR) 'one year later';