-- To create table student --

CREATE TABLE student(

student\_id INT PRIMARY KEY,

name VARCHAR(20),

major VARCHAR(20)

);

-- To check the table description and schema --

DESCRIBE student;

-- To drop/delete whole table we use DROP keyword --

DROP TABLE student;

-- Once table is created we can add additional column using ALTER keyword --

ALTER TABLE student ADD gpa DECIMAL(3,2);

-- If we want to drop specific column from table we use same ALTER keyword to do the same --

ALTER TABLE student DROP COLUMN gpa;

-- Inserting data into table --

INSERT INTO student VALUES(1,'Jack','Biology');

INSERT INTO student VALUES(2,'Kate','Sociology');

INSERT INTO student VALUES(4,'Tom','Sociology');

INSERT INTO student VALUES(5,'Mike','Computer Science');

-- If we do not have coressponding column value then we can insert data by specifying table column name while inserting --

--suppose for claire we do not have major then we can insert using below query--

INSERT INTO student(student\_id,name) VALUES(3,'Claire');

-- To check the table content --

SELECT \*FROM student;

--------------------------------------------------------------------------------------------------------------------------------------------

-- Drop student table --

DROP TABLE student;

-- create student table again --

CREATE TABLE student(

student\_id INT PRIMARY KEY,

name VARCHAR(20) NOT NULL, -- specifying NOT NULL because student must have their name --

major VARCHAR(20) UNIQUE -- UNIQUE keyweord which allows major to be unique for each students--

)

-- Insert values int table --

INSERT INTO student VALUES(1,'Jack','Biology');

INSERT INTO student VALUES(2,'Kate','Sociology');

INSERT INTO student VALUES(3,'Mark','Science');

INSERT INTO student VALUES(4,'Tom','ML');

INSERT INTO student VALUES(5,'Mike','Computer Science');

-- Select table --

SELECT \*FROM student;

-- Let's drop table and create again with major column with constraints DEFAULT --

DROP TABLE student;

-- Create table again --

CREATE TABLE student(

student\_id INT PRIMARY KEY,

name VARCHAR(20) NOT NULL,

major VARCHAR(20) DEFAULT 'Not decided'

);

-- description of table --

DESCRIBE student;

-- Insert value by leaving some student's major as blank --

INSERT INTO student VALUES(1,'Jack','Biology');

INSERT INTO student VALUES(2,'Kate','Sociology');

INSERT INTO student VALUES(3,'Mark','Science');

INSERT INTO student(student\_id,name) VALUES(4,'Tom');

INSERT INTO student VALUES(5,'Mike','Computer Science');

-- select --

SELECT \*FROM student;

-- creating table again with constraint on primary key --

DROP TABLE student;

-- table creation --

CREATE TABLE student(

student\_id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(20) NOT NULL,

major VARCHAR(20) UNIQUE

);

-- INSERT values --

-- We are not inserting id manually --

INSERT INTO student(name,major) VALUES('Jack','Biology');

INSERT INTO student(name,major) VALUES('Kate','Sociology');

INSERT INTO student(name,major) VALUES('Mark','Science');

INSERT INTO student(name,major) VALUES('Mike','Computer Science');

SELECT \*FROM student;

-- Update major as Bio for Biology--

UPDATE student SET major = 'Bio' WHERE major = 'Biology';

UPDATE student SET name = 'Nick',major = 'Data Science' WHERE student\_id = 1;

-- delete row --

DELETE FROM student WHERE student\_id = 4;

-----------------------------------------------------------------------------------------------------------------------------------------

--Select all records from table--

SELECT \*FROM student;

-- select specific column from table

SELECT name FROM student;

SELECT name,major FROM student ORDER BY name;

SELECT name,major FROM student ORDER BY name DESC;

SELECT \*FROM student ORDER BY student\_id DESC;

SELECT \*FROM student ORDER BY major,student\_id DESC;

SELECT \*FROM student LIMIT 1;

SELECT COUNT(\*) FROM student;

SELECT \*FROM student WHERE major = 'Data Science';

SELECT \*FROM student WHERE major <> 'Data Science';

SELECT \*FROM student WHERE major IN('Science','Data Science');

---------- Creating Company Database --------

-- Let's drop student table --

DROP TABLE student;

----- Creating Employee table -----

CREATE TABLE employee(

emp\_id INT PRIMARY KEY,

first\_name VARCHAR(40),

last\_name VARCHAR(40),

birth\_date DATE,

sex VARCHAR(1),

salary INT,

super\_id INT,

branch\_id INT

);

-- Checking created table schema --

DESCRIBE employee;

--- Creating Branch table --

CREATE TABLE branch(

branch\_id INT PRIMARY KEY,

branch\_name VARCHAR(20),

mgr\_id INT,

mgr\_start\_date DATE,

FOREIGN KEY(mgr\_id) REFERENCES employee(emp\_id) ON DELETE SET NULL

);

-- Checking created table schema --

DESCRIBE branch;

-- Modify employee table and add branch\_id and super\_id as foreign key --

ALTER TABLE employee

ADD FOREIGN KEY(branch\_id)

REFERENCES branch(branch\_id) ON DELETE SET NULL;

ALTER TABLE employee

ADD FOREIGN KEY(super\_id)

REFERENCES branch(mgr\_id) ON DELETE SET NULL;

-- Create table Client--

CREATE TABLE client(

client\_id INT PRIMARY KEY,

client\_name VARCHAR(30),

branch\_id INT,

FOREIGN KEY(branch\_id) REFERENCES branch(branch\_id) ON DELETE SET NULL

);

DESCRIBE client;

--- Create table work\_with --

CREATE TABLE works\_with(

emp\_id INT,

client\_id INT,

total\_sales INT,

PRIMARY KEY(emp\_id,client\_id),

FOREIGN KEY(emp\_id) REFERENCES employee(emp\_id) ON DELETE CASCADE,

FOREIGN KEY(client\_id) REFERENCES client(client\_id) ON DELETE CASCADE

);

DESCRIBE works\_with;

CREATE TABLE branch\_supplier(

branch\_id INT,

supplier\_name VARCHAR(30),

supply\_type VARCHAR(30),

PRIMARY KEY(branch\_id,supplier\_name),

FOREIGN KEY(branch\_id) REFERENCES branch(branch\_id) ON DELETE CASCADE

);

--- Insert data into employee table ---

INSERT INTO employee VALUES(100,'David','Wallece','1967-11-17','M',250000,NULL,NULL);

INSERT INTO branch VALUES(1,'corporate',100,'2006-02-09');

UPDATE employee SET branch\_id = 1 WHERE emp\_id = 100;

SELECT \*FROM employee;

INSERT INTO employee VALUES(101, 'Jan', 'Levinson', '1961-05-11', 'F', 110000, 100, 1);

-- Scranton

INSERT INTO employee VALUES(102, 'Michael', 'Scott', '1964-03-15', 'M', 75000, 100, NULL);

INSERT INTO branch VALUES(2, 'Scranton', 102, '1992-04-06');

UPDATE employee

SET branch\_id = 2

WHERE emp\_id = 102;

INSERT INTO employee VALUES(103, 'Angela', 'Martin', '1971-06-25', 'F', 63000, 102, 2);

INSERT INTO employee VALUES(104, 'Kelly', 'Kapoor', '1980-02-05', 'F', 55000, 102, 2);

INSERT INTO employee VALUES(105, 'Stanley', 'Hudson', '1958-02-19', 'M', 69000, 102, 2);

-- Stamford

INSERT INTO employee VALUES(106, 'Josh', 'Porter', '1969-09-05', 'M', 78000, 100, NULL);

INSERT INTO branch VALUES(3, 'Stamford', 106, '1998-02-13');

UPDATE employee

SET branch\_id = 3

WHERE emp\_id = 106;

INSERT INTO employee VALUES(107, 'Andy', 'Bernard', '1973-07-22', 'M', 65000, 106, 3);

INSERT INTO employee VALUES(108, 'Jim', 'Halpert', '1978-10-01', 'M', 71000, 106, 3);

-- BRANCH SUPPLIER

INSERT INTO branch\_supplier VALUES(2, 'Hammer Mill', 'Paper');

INSERT INTO branch\_supplier VALUES(2, 'Uni-ball', 'Writing Utensils');

INSERT INTO branch\_supplier VALUES(3, 'Patriot Paper', 'Paper');

INSERT INTO branch\_supplier VALUES(2, 'J.T. Forms & Labels', 'Custom Forms');

INSERT INTO branch\_supplier VALUES(3, 'Uni-ball', 'Writing Utensils');

INSERT INTO branch\_supplier VALUES(3, 'Hammer Mill', 'Paper');

INSERT INTO branch\_supplier VALUES(3, 'Stamford Lables', 'Custom Forms');

-- CLIENT

INSERT INTO client VALUES(400, 'Dunmore Highschool', 2);

INSERT INTO client VALUES(401, 'Lackawana Country', 2);

INSERT INTO client VALUES(402, 'FedEx', 3);

INSERT INTO client VALUES(403, 'John Daly Law, LLC', 3);

INSERT INTO client VALUES(404, 'Scranton Whitepages', 2);

INSERT INTO client VALUES(405, 'Times Newspaper', 3);

INSERT INTO client VALUES(406, 'FedEx', 2);

-- WORKS\_WITH

INSERT INTO works\_with VALUES(105, 400, 55000);

INSERT INTO works\_with VALUES(102, 401, 267000);

INSERT INTO works\_with VALUES(108, 402, 22500);

INSERT INTO works\_with VALUES(107, 403, 5000);

INSERT INTO works\_with VALUES(108, 403, 12000);

INSERT INTO works\_with VALUES(105, 404, 33000);

INSERT INTO works\_with VALUES(107, 405, 26000);

INSERT INTO works\_with VALUES(102, 406, 15000);

INSERT INTO works\_with VALUES(105, 406, 130000);

SELECT \*FROM employee;

SELECT \*FROM branch;

SELECT \*FROM client;

SELECT \*FROM works\_with;

------------- SELECT QUERIES ---------------

-- Find all employees

SELECT \*FROM employee;

-- Find all clients

SELECT \*FROM client;

-- Find all employees ordered by salary

SELECT \*FROM employee ORDER BY salary;

SELECT \*FROM employee ORDER BY salary DESC;

-- Find all employees order by sex and then name

SELECt \*FROM employee ORDER BY sex,first\_name,last\_name;

-- Find first 5 employees from employee table

SELECT \*FROM employee LIMIT 5;

-- Find first and last name of all employees

SELECT first\_name,last\_name FROM employee;

-- Find forename and surname of all employees

SELECT first\_name as Forename,last\_name as Surname FROM employee;

-- Find out all different genders of employees

SELECT DISTINCT sex FROM employee;

------------------- SQL FUNCTIONS -------------------------------

-- Find the number of employees

SELECT COUNT(emp\_id) FROM employee;

SELECT COUNT(super\_id) FROM employee;

-- Find the number of female employees born after 1970

SELECT COUNT(emp\_id) FROM employee WHERE sex = 'F' AND birth\_date > '1970-01-01';

SELECT \*FROM employee;

-- Find the average of all employee salaries

SELECT AVG(salary) FROM employee;

-- Find the average salary of all male employees

SELECT AVG(salary) FROM employee WHERE sex = 'M';

-- Find the sum of all employee's salary

SELECT SUM(salary) FROM employee;

-- Find how many males and females are there

SELECT COUNT(sex),sex FROM employee GROUP BY sex;

-- Find the how much each client

SELECT \* FROM works\_with;

SELECT SUM(total\_sales),client\_id FROM works\_with GROUP BY client\_id;

-- Wildcards---

-- Find any client who are an LLC

SELECT \*FROM client WHERE client\_name LIKE '%LLC%';

-- FInd any branch supplier who are in business label

SELECt \*FROM branch\_supplier;

SELECT \*FROM branch\_supplier WHERE supplier\_name LIKE '%Labels%';

-- Find any employee who's born in october

SELECT \*FROM employee;

SELECT \*FROM employee WHERE birth\_date LIKE '%-10-%';

SELECT \*FROM employee WHERE birth\_date LIKE '\_\_\_\_-10%';

-- Find any clients who are schools

SELECT \*FROM client WHERE client\_name LIKE '%school';

--------------------- UNIONS in SQL --------------------------

-- Find list of employees and branch name

SELECT first\_name FROM employee UNION SELECT branch\_name FROM branch;

-- FInd the list of all clients and branch suppliers name

SELECT client\_name AS Client\_Supplier\_Names FROM client

UNION

SELECT supplier\_name FROM branch\_supplier;

-- Find the list of all the money spent or earned by company

SELECT salary FROM employee

UNION

SELECT total\_sales FROM works\_with;

INSERT INTO branch VALUES(4,'Buffalo',NULL,NULL);

------------------------ JOINS in SQL ------------------------------

-- Find all branches and the names of their managers

SELECT employee.emp\_id,employee.first\_name,branch.branch\_name

FROM employee JOIN branch ON employee.emp\_id = branch.mgr\_id;

-- Left Join

SELECT employee.emp\_id,employee.first\_name,branch.branch\_name

FROM employee

LEFT JOIN branch

ON

employee.emp\_id = branch.mgr\_id;

-- Right JOIN

SELECT employee.emp\_id,employee.first\_name,branch.branch\_name

FROM employee

RIGHT JOIN branch

ON

employee.emp\_id = branch.mgr\_id;

------------------------- Nested Queries in SQL -------------------------

-- Find the name of all employees who have sold over 30,000 to a single client

SELECT first\_name,last\_name FROM employee WHERE emp\_id IN (SELECT emp\_id FROM works\_with WHERE total\_sales > 30000);

--- Find all clients who are handled by the branch that Michael Scott manages.

SELECT \*FROM branch;

SELECT \*FROM employee;

SELECT client\_name FROM client WHERE branch\_id IN (

SELECT branch\_id FROM branch WHERE mgr\_id IN (

SELECT emp\_id FROM employee WHERE first\_name = 'Michael'));

---------------------- Trigger in SQL -------------------------

CREATE TABLE trigger\_test(

message VARCHAR(100)

);

---- Now create Trigger

DELIMITER $$

CREATE TRIGGER my\_trigger

BEFORE INSERT ON employee

FOR EACH ROW BEGIN

INSERT INTO trigger\_test VALUES('Employee Added');

END$$

DELIMITER;

-- Now let's add new value in employee table

INSERT INTO employee VALUES(109,'Oscar','Mart','1989-01-01','M',69000,106,3);

SELECT \*FROM employee

-- Now let's check whether our trigger gets executed automatically on now by checking trigger test table.

SELECT \*FROM trigger\_test;

-- Create one more trigger that will add firat name of newly inserted values

DELIMITER $$

CREATE TRIGGER my\_trigger\_1

BEFORE INSERT ON employee

FOR EACH ROW BEGIN

INSERT INTO trigger\_test VALUES(NEW.first\_name);

END$$

DELIMITER;

-- Now let's add new value in employee table

INSERT INTO employee VALUES(110,'Nick','Niks','1992-02-21','M',1000000,106,3);

-- Now let's check whether our trigger gets executed automatically on now by checking trigger test table.

SELECT \*FROM trigger\_test;