CREATE TABLE employee (

emp\_id INT PRIMARY KEY,

first\_name VARCHAR(40),

last\_name VARCHAR(40),

birth\_day DATE,

gender VARCHAR(1),

salary INT,

super\_id INT,

branch\_id INT

);

CREATE TABLE branch (

branch\_id INT PRIMARY KEY,

branch\_name VARCHAR(40),

mgr\_id INT,

mgr\_start\_date DATE,

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

);

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 employee(emp\_id)

ON DELETE SET NULL;

CREATE TABLE client (

client\_id INT PRIMARY KEY,

client\_name VARCHAR(40),

branch\_id INT,

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

);

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

);

CREATE TABLE branch\_supplier (

branch\_id INT,

supplier\_name VARCHAR(40),

supply\_type VARCHAR(40),

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

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

);

*-- -----------------------------------------------------------------------------*

*-- Corporate*

INSERT INTO employee VALUES(100, 'David', 'Wallace', '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;

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);

*mysql –u root –p;*

*use company;*

*-- Find all employees*

SELECT \* FROM employee;

*-- Find all clients*

SELECT \* FROM client;

*-- Find all employees ordered by salary*

SELECT \* FROM employee ORDER BY salary ASC/DESC;

*-- Find all employees ordered by gender then name*

SELECT \* FROM employee ORDER BY gender, first\_name;

*-- Find the first 5 employees in the table*

SELECT \* FROM employee LIMIT 5;

*-- Find the first and last names of all employees*

SELECT first\_name, last\_name FROM employee;

SELECT first\_name, employee.last\_name FROM employee;

*-- Find the forename and surnames names of all employees*

SELECT first\_name AS forename, employee.last\_name AS surname FROM employee;

*-- Find out all the different genders*

SELECT DISTINCT gender FROM employee;

*-- Find all male employees*

SELECT \* FROM employee WHERE gender = 'M';

*-- Find all employees at branch 2*

SELECT \* FROM employee WHERE branch\_id = 2;

*-- Find all employee's id's and names who were born after 1969*

MariaDB [company]> SELECT emp\_id, first\_name, last\_name FROM employee WHERE birth\_day >= 1970-01-01;

*-- Find all female employees at branch 2*

SELECT \* FROM employee WHERE branch\_id = 2 AND gender = 'F';

*-- Find all employees who are female & born after 1969 or who make over 80000*

SELECT \* FROM employee WHERE (gender = 'F' AND birth\_day >= '1970-01-01') OR salary>80000;

*-- Find all employees born between 1970 and 1975*

SELECT \* FROM employee WHERE birth\_day BETWEEN '1970-01-01' AND '1975-01-01';

*-- Find all employees named Jim, Michael, Johnny or David*

SELECT \* FROM employee WHERE first\_name IN ('Jim', 'Michael', 'Johnny', 'David');

*-- Find the number of employees*

SELECT COUNT(emp\_id) FROM employee;

*-- Find the average of all employee's salaries*

SELECT AVG(salary) FROM employee;

*-- Find the sum of all employee's salaries*

SELECT SUM(salary) FROM employee;

*-- Find out how many males and females there are*

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

*-- Find the total sales of each salesman*

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

*-- Find the total amount of money spent by each client*

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

*--find any client’s who an LLC*

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

--find any suppliers who r in the lable business

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

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

SELECT \* FROM branch\_supplier WHERE supplier\_name LIKE '\_Labels';**WRONG**

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

--find any employee born on the 10th day of month

SELECT \* FROM employee WHERE birth\_day LIKE '\_10';**WRONG**

SELECT \* FROM employee WHERE birth\_day LIKE '\_\_\_\_\_\_\_\_11';

SELECT \* FROM employee WHERE birth\_day LIKE '\_\_\_\_\_\_\_\_05';

--find any clients who are schools

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

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

--find a list of employee and branch name

SELECT employee.first\_name AS Employee\_Branch\_names FROM employee UNION SELECT branch.branch\_name FROM branch; **WITHOUT USING JOIN**

--find a list of client and branch suppliers name

SELECT client.client\_name AS Client\_Supplier\_names FROM client UNION SELECT branch\_supplier.supplier\_name FROM branch\_supplier;

--find all branches and 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;

SELECT employee.emp\_id, employee.first\_name, branch.branch\_name FROM employee JOIN branch ON employee.emp\_id = branch.mgr\_id; **THEATA JOIN**

SELECT employee.emp\_id, employee.first\_name, branch.branch\_name FROM employee LEFT JOIN branch ON employee.emp\_id = branch.mgr\_id; **LEFT JOIN**

SELECT employee.emp\_id, employee.first\_name, branch.branch\_name FROM employee RIGHT JOIN branch ON employee.emp\_id = branch.mgr\_id; **RIGHT JOIN**

--Find names of all employees sold over 50000

SELECT employee.first\_name, FROM employee JOIN (SELECT works\_with.emp\_id FROM works\_with WHERE works\_with.total\_sales>50000) ON employee.emp\_id = works\_with.emp\_id;

SELECT employee.first\_name, employee.last\_name FROM employee WHERE employee.emp\_id IN (SELECT works\_with.emp\_id FROM works\_with WHERE works\_with.total\_sales>50000);

SELECT employee.first\_name, employee.last\_name FROM employee WHERE employee.emp\_id IN (SELECT works\_with.emp\_id FROM works\_with WHERE SUM(total\_sales)>150000 FROM works\_with GROUP BY emp\_id);

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

--assume you know Michael’s id

SELECT client.client\_id, client.client\_name

-> FROM client

-> WHERE client.branch\_id = (SELECT branch.branch\_id FROM branch WHERE branch.mgr\_id=102);

--don’t know the Michael Scott’s id

SELECT client.client\_id, client.client\_name

-> FROM client

-> WHERE client.branch\_id = (SELECT branch.branch\_id FROM branch WHERE branch.mgr\_id = (SELECT employee.emp\_id

FROM employee WHERE employee.first\_name = ‘Michael’ AND employee.last\_name = ‘Scott’ LIMIT 1));

--find the names of employees who work with clients handled by the scronton branch

SELECT employee.first\_name, employee.last\_name

FROM employee WHERE employee.branch\_id=

(SELECT branch.branch\_id FROM branch WHERE branch.branch\_name = ‘Scronton’);

--find the names of the employee who have spent more than 100,000 dollars

SELECT client.client\_id, client.client\_name

FROM client

WHERE client.client\_id IN (SELECT client\_id

FROM(

SELECT SUM(works\_with.total\_sales) AS totals, client\_id FROM works\_with GROUP BY client\_id) AS total\_client\_sales WHERE totals>100000);

--On delete set null--

DELETE FROM employee WHERE emp\_id = 102;

SELECT \* FROM employee;

CREATE TABLE emps AS (SELECT E.\* FROM EMPLOYEE AS E WHERE E.gender = ‘M’);

create table wc\_info(

-> bname varchar(20),

-> cname varchar(20));

insert into wc\_info(bname, cname)

-> select B.branch\_name, C.client\_name

-> from branch B, client C where B.branch\_id = C.branch\_id;

select E.first\_name as Employee\_name, S.first\_name as Supervisor\_name

-> from employee E, employee S

-> where E.super\_id = S.emp\_id;

MariaDB [company]> select E.first\_name as Employee\_name, S.first\_name as Supervisor\_name

-> from employee E, employee S

-> where E.emp\_id = S.super\_id;

select S.first\_name as Supervisor\_name, E.first\_name as Employee\_name

-> from employee E, employee S

-> where E.emp\_id = S.super\_id;

--find the total sales for each sales man

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

-- create view to find the total sales for each sales man

create view total as SELECT emp\_id, SUM(total\_sales) AS total FROM works\_with GROUP BY emp\_id;

drop view total;

--create view to find the money spent by each client

create view money\_spent\_client as SELECT client\_id, SUM(total\_sales) AS money\_spent FROM works\_with GROUP BY client\_id;

--create a view to display all the client id and clinet name handled by the branch that Hudson manages where Hudson is last name

create view hudson\_client as select client\_id, client\_name from client where branch\_id = (select branch\_id from branch where mgr\_id in (select emp\_id from employee where employee.last\_name = 'Hudson'));

create view scott\_client as select client\_id, client\_name from client where branch\_id = (select branch\_id from branch where mgr\_id in (select emp\_id from employee where employee.last\_name = 'Scott'));

create view porter\_client as select client\_id, client\_name from client where branch\_id = (select branch\_id from branch where mgr\_id = (select emp\_id from employee where employee.last\_name = 'porter'));

drop view porter\_client;**DROPING VIEW**

delimiter $$

--creating function

create function tovote(age integer)

-> RETURNS varchar(20)

-> DETERMINISTIC

-> BEGIN

-> IF age > 18 THEN

-> RETURN ("yes");

-> ELSE

-> RETURN ("No");

-> END IF;

-> end$$

select tovote(40);

-> $$

delimiter &&

create procedure get\_max\_sal()

-> begin

-> select \* from employee where salary > 100000;

-> select count(emp\_id) as total\_employee from employee;

-> end &&

delimiter ;

call get\_max\_sal();

delimiter &&

create procedure uspGetEmployeeList()

-> begin

-> select emp\_id, first\_name, last\_name

-> from employee;

-> end &&

delimiter ;

call uspGetEmployeeList();

delimiter &&

create procedure get\_employee(IN var1 INT)

-> begin

-> select \* from employee limit var1;

-> select count(emp\_id) as total\_employee from employee;

-> end &&

delimiter ;

call get\_employee(5);

--triggers

create table student\_age(age int, name varchar(35));

delimiter //

create Trigger studentage BEFORE INSERT ON student\_age for each row

-> begin

-> if new.age < 0 then set new.age = 0;

-> end if;

-> end//

delimiter ;

insert into student\_age(age, name) values (40, 'ram');

insert into student\_age(age, name) values (-10, 'sam');

select \* from student\_age;

--cursor

delimiter //

create table backup(tutorialID int, tutorialTitle varchar(20), tutorialAuthor varchar(20), tutorialDate date)//

insert into backup values(1, 'DBMS', 'Sanmat', 10-6-2022)//

insert into backup values(2, 'SE', 'Sanjay', 10-5-2022)//

create procedure ExampleProc6()

begin

declare done int default 0;

declare tutorialID INTEGER;

declare tutorialTitle, tutorialAuthor, tutorialDate varchar(20);

declare cur cursor for select \* from backup;

declare continue handler for not found set done = 1;

open cur;

label: LOOP

fetch cur into tutorialID, tutorialTitle, tutorialAuthor, tutorialDate;

if done = 1 then leave label;

end if;

end loop;

close cur;

end//

delimiter ;

call ExampleProc6();

delimiter //

create procedure curconcs()

begin

declare done int default 0;

declare v\_salary INTEGER;

declare v\_first\_name varchar(20);

declare cur cursor for select first\_name, salary from employee;

declare continue handler for not found set done = 1;

open cur;

label: LOOP

fetch cur into v\_first\_name, v\_salary;

if done = 1 then leave label;

end if;

end loop;

close cur;

end//

delimiter ;

call curconcs();

select salary, case(branch\_id)

-> when '1' then salary+2000

-> when '2' then salary+1500

-> end as netsal

-> from employee;