**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

**“JnanaSangama”, Belgaum -590014, Karnataka.**

****

**LAB REPORT**

**on**

**Database Management Systems (23CS3PCDBM)**

***Submitted by***

**Prathith Suresh Rao (1BM23CS247)**

***in partial fulfilment for the award of the degree of***

**BACHELOR OF ENGINEERING**

***in***

**COMPUTER SCIENCE AND ENGINEERING**

****

**B.M.S. COLLEGE OF ENGINEERING**

**(Autonomous Institution under VTU)**

**BENGALURU-560019**

**Sep-2024 to Jan-2025**

**B. M. S. College of Engineering,**

**Bull Temple Road, Bangalore 560019**

(Affiliated To Visvesvaraya Technological University, Belgaum)

**Department of Computer Science and Engineering**

****

**CERTIFICATE**

This is to certify that the Lab work entitled “Database Management Systems (23CS3PCDBM)” carried out by **Prathith Suresh Rao (1BM23CS247),** who is Bonafede student of **B. M. S. College of Engineering.** It is in partial fulfilment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a Database Management Systems (23CS3PCDBM) work prescribed for the said degree.

|  |  |
| --- | --- |
| Lab faculty Incharge Name  Assistant Professor  Department of CSE, BMSCE | Dr. Joythi S Nayak  Professor HOD  Department of CSE, BMSCE |

<#Employee_add>

**Index**

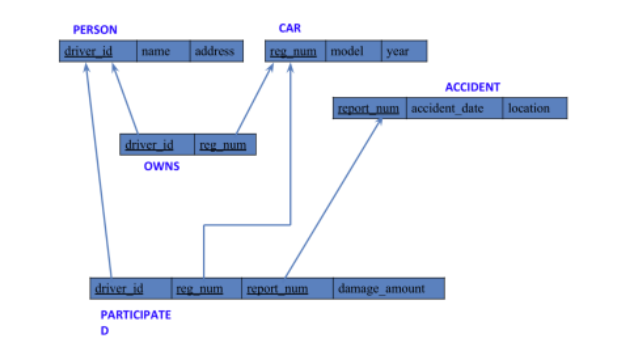
|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.**  **No.** | **Date** | **Experiment Title** | **Page No.** |
| 1 | 01-10-24 | [Insurance Database](#Insurance) | 4 |
| 2 | 08-10-24 | [More Queries on Insurance Database](#insurance_add) | 10 |
| 3 | 15-10-24 | [Bank Database](#Bank) | 12 |
| 4 | 22-10-24 | [More Queries on Bank Database](#Bank_add) | 19 |
| 5 | 29-10-24 | [Employee Database](#Employee) | 21 |
| 6 | 26-10-24 | [More Queries on Employee Database](#Employee_add) | 29 |
| 7 | 03-10-24 | [Supplier Database](#Supplier) | 32 |
| 8 |  | NO SQL - Student Database |  |
| 9 |  | NO SQL - Customer Database |  |
| 10 |  | NO SQL – Restaurant Database |  |

**INSURANCE DATABASE**

**Question (Week 1)**

* person (driver\_id: string, name: string, address: string)
* car (reg\_num: string, model: string, year: int)
* accident (report\_num: int, accident\_date: date, location: string)
* owns (driver\_id: string, reg\_num: string)
* participated (driver\_id: string,reg\_num: string, report\_num: int, damage\_amount: int)
* Create the above tables by properly specifying the primary keys and the foreign keys. - Enter at least five tuples for each relation
* Display Accident date and location
* Update the damage amount to 25000 for the car with a specific reg\_num (example “KA053408”) for which the accident report number was 12.
* Add a new accident to the database.
* To Do
  + Display Accident date and location
  + Display driver id who did accident with damage amount greater than or equal to Rs.25000

**SCHEMA DIAGRAM**

****

**CREATE DATABASES:**

create database insurance;

use INSURANCE;

**CREATE TABLES:**

create table person

(Driver\_id varchar(10),

Name varchar(20),

Address varchar(30),

primary key(Driver\_id));

create table car

(reg\_num varchar(10),

Model varchar(10),

year int,

primary key(reg\_num));

create table accident

(Report\_num int,

Accident\_date date,

location varchar(20),

primary key(Report\_num));

create table owns

(Driver\_id varchar(10),

reg\_num varchar(10),

primary key(Driver\_id, reg\_num),

foreign key(Driver\_id) references person(Driver\_id),

foreign key(reg\_num) references car(reg\_num));

create table participated

(Driver\_id varchar(10),

reg\_num varchar(10),

Report\_num int,

Damage\_amount int,

primary key(Driver\_id, reg\_num, report\_num),

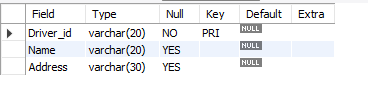
foreign key(Driver\_id) references person(driver\_id),

foreign key(reg\_num) references car(reg\_num),

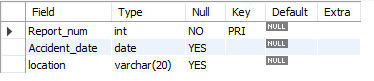
foreign key(report\_num) references accident(report\_num));

**STRUCTURE OF THE TABLE:**

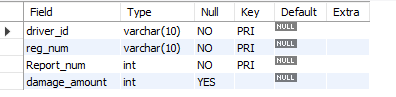
desc person;



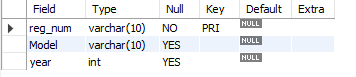
desc accident;

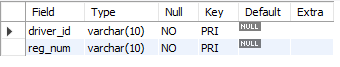


desc participated;



desc car;



desc owns;

**INSERTING VALUES TO THE TABLE:**

insert into person values('A01','Richard','Srinivas Nagar');

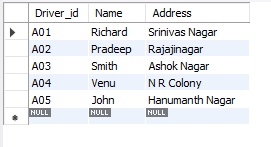
insert into person values('A02','Pradeep','Rajajinagar');

insert into person values('A03','Smith','Ashok Nagar');

insert into person values('A04','Venu','N R Colony');

insert into person values('A05','John','Hanumanth Nagar');

select \* from person;



insert into car values('KA052250','INDICA','1990');

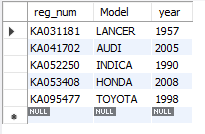
insert into car values('KA031181', 'LANCER','1957');

insert into car values('KA095477','TOYOTA','1998');

insert into car values('KA053408', 'HONDA','2008');

insert into car values('KA041702','AUDI','2005');

select \* from car;



insert into accident values('11','2003-01-01','Mysore Road');

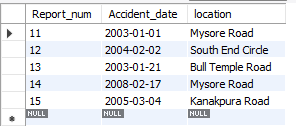
insert into accident values('12', '2004-02-02','South End Circle');

insert into accident values('13','2003-01-21','Bull Temple Road');

insert into accident values('14', '2008-02-17','Mysore Road');

insert into accident values('15','2005-03-04','Kanakpura Road');

select \* from accident;



insert into owns values('A01','KA052250');

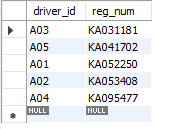
insert into owns values('A02','KA053408');

insert into owns values('A03','KA031181');

insert into owns values('A04','KA095477');

insert into owns values('A05','KA041702');

select \* from owns;



insert into participated values('A01','KA052250','11','10000');

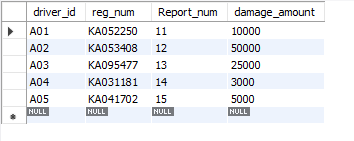
insert into participated values('A02','KA053408','12','50000');

insert into participated values('A03','KA095477','13','25000');

insert into participated values('A04','KA031181','14','3000');

insert into participated values('A05','KA041702','15','5000');

Select \* from participated;



**QUERIES**

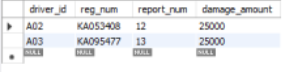
* **Update the damage amount to 25000 for the car with a specific reg-num**

**for which the accident report number was 12.**

update participated

set damage\_amount=25000

where reg\_num=’KA053408’ and report\_num=12;



* **Find the total number of people who owned cars that were involved in accidents in 2008.**

select count(distinct driver\_id) CNT

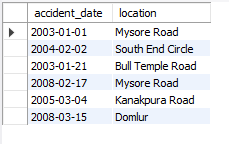
from participated a, accident b

where a.report\_num=b.report\_num and b.accident\_date like ‘2008%’;



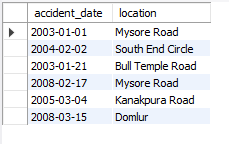
* **Display Accident date and location**

select accident\_date, location from accident;



* **Add a new accident to the database.**

insert into accident values ('16','2008-03-15','Domlur');

select accident\_date, location from accident;

* **Display driver id who did accident with damage amount greater than or equal to Rs.25000**

select driver\_id from participated where damage\_amount >=25000;

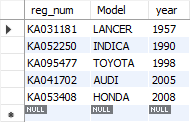


Additional Queries on Insurance Database

**Questions (Week 2)**

* **Display the entire CAR relation in the ascending order of manufacturing year.**

select \*from car order by year asc;



* **Find the number of accidents in which cars belonging to a specific model (example 'Lancer') were involved.**

select count(report\_num) from car c, participated p where c.reg\_num = p.reg\_num and c.model='Lancer';



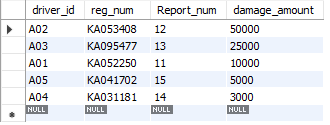
* **Find the total number of people who owned cars that were involved in accidents in 2008.**

select count(distinct driver\_id) CNT from participated a, accident b where a.report\_num=b.report\_num and b.accident\_date like '\_\_08%';



* **List the entire participated relation in the descending order of damage amount.**

select \*from participated order by damage\_amount desc;



* **Find the average damage amount**

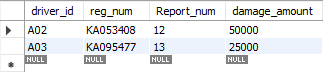
select avg(damage\_amount) from participated;



* **Delete the tuple whose damage amount is below the average damage amount**

delete from participated where damage\_amount<18600;

select \*from participated;



* **List the name of drivers whose damage is greater than the average damage amount.**

select name from person a, participated b

WHERE a.driver\_id = b.driver\_id and

damage\_amount > (select avg(damage\_amount) from participated);



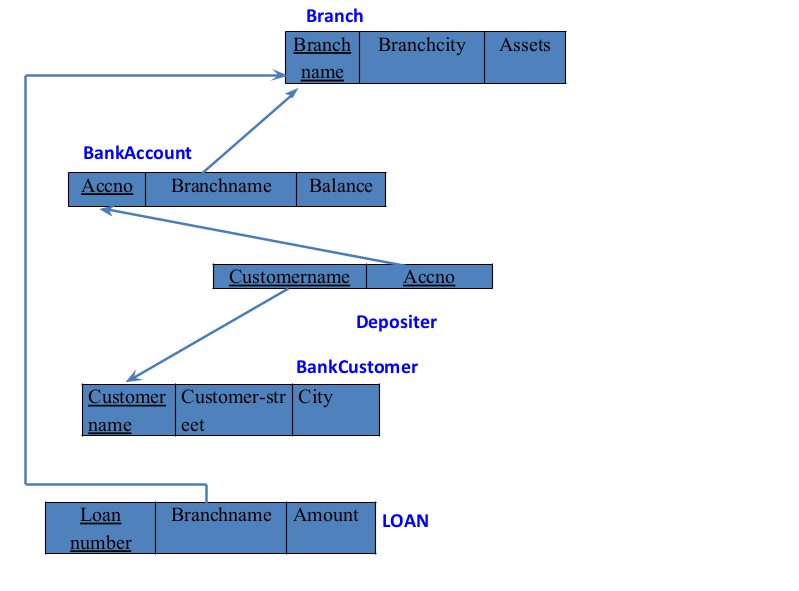
* **Find maximum damage amount.**

select max(damage\_amount) from participated;



**Bank database**

**Schema Diagram:**

****

**Questions(week3)**

* Branch (branch-name: String, branch-city: String, assets: real)
* BankAccount(accno: int, branch-name: String, balance: real)
* BankCustomer (customer-name: String, customer-street: String, customer-city: String)
* Depositer(customer-name: String, accno: int)
* LOAN (loan-number: int, branch-name: String, amount: real)
* Create the above tables by properly specifying the primary keys and the foreign keys.
* Enter at least five tuples for each relation.
* Display the branch name and assets from all branches in lakhs of rupees and rename the assets column to 'assets in lakhs'.
* Find all the customers who have at least two accounts at the same branch (ex. SBI\_ResidencyRoad).
* Create a view which gives each branch the sum of the amount of all the loans at the branch.

**Create Database:**

create database bank\_247;

use bank\_247;

**Create table:**

CREATE TABLE branch (

branch\_name VARCHAR(30),

branch\_city VARCHAR(25),

assets INT,

PRIMARY KEY (branch\_name)

);

CREATE TABLE bankaccount (

accno INT,

branch\_name VARCHAR(30),

balance INT,

PRIMARY KEY (accno),

FOREIGN KEY (branch\_name)

REFERENCES branch (branch\_name)

);

CREATE TABLE bankcustomer (

customername VARCHAR(20),

customer\_street VARCHAR(30),

customer\_city VARCHAR(35),

PRIMARY KEY (customername)

);

CREATE TABLE depositer (

customername VARCHAR(20),

accno INT,

PRIMARY KEY (customername , accno),

FOREIGN KEY (accno)

REFERENCES bankaccount (accno),

FOREIGN KEY (customername)

REFERENCES bankcustomer (customername)

);

CREATE TABLE loan (

loan\_number INT,

branch\_name VARCHAR(30),

amount INT,

PRIMARY KEY (loan\_number),

FOREIGN KEY (branch\_name)

REFERENCES branch (branch\_name));

CREATE TABLE borrower (

loan\_number INT,

customername VARCHAR(20),

PRIMARY KEY (loan\_number),

FOREIGN KEY (loan\_number)

REFERENCES loan (loan\_number),

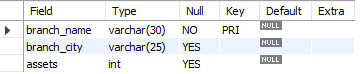
FOREIGN KEY (customername)

REFERENCES bankcustomer (customername)

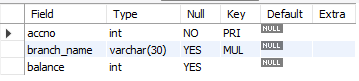
);

**Structure of the table:**

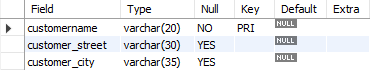
desc branch;



desc bankaccount;



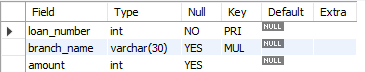
desc bankcustomer;



desc depositer;



desc loan;



desc borrower;



**Inserting values to the table:**

insert into branch values("SBI\_Chamrajpet","Bangalore",50000);

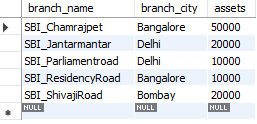
insert into branch values("SBI\_ResidencyRoad","Bangalore",10000);

insert into branch values("SBI\_ShivajiRoad","Bombay",20000);

insert into branch values("SBI\_Parliamentroad","Delhi",10000);

insert into branch values("SBI\_Jantarmantar","Delhi",20000);

select \* from branch;



insert into bankaccount values(1,"SBI\_Chamrajpet",2000);

insert into bankaccount values(2,"SBI\_ResidencyRoad",5000);

insert into bankaccount values(3,"SBI\_ShivajiRoad",6000);

insert into bankaccount values(4,"SBI\_Parliamentroad",9000);

insert into bankaccount values(5,"SBI\_Jantarmantar",8000);

insert into bankaccount values(6,"SBI\_ShivajiRoad",4000);

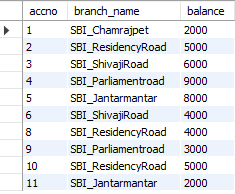
insert into bankaccount values(8,"SBI\_ResidencyRoad",4000);

insert into bankaccount values(9,"SBI\_Parliamentroad",3000);

insert into bankaccount values(10,"SBI\_ResidencyRoad",5000);

insert into bankaccount values(11,"SBI\_Jantarmantar",2000);

select \* from bankaccount;



insert into bankcustomer values("Avinash","BUll\_temple\_Road","Bangalore");

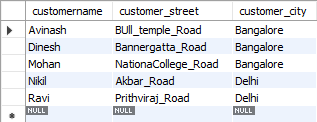
insert into bankcustomer values("Dinesh","Bannergatta\_Road","Bangalore");

insert into bankcustomer values("Mohan","NationaCollege\_Road","Bangalore");

insert into bankcustomer values("Nikil","Akbar\_Road","Delhi");

insert into bankcustomer values("Ravi","Prithviraj\_Road","Delhi");

select \* from bankcustomer;



insert into depositer values("Avinash",1);

insert into depositer values("Dinesh",2);

insert into depositer values("Nikil",4);

insert into depositer values("Ravi",5);

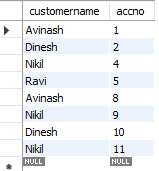
insert into depositer values("Avinash",8);

insert into depositer values("Nikil",9);

insert into depositer values("Dinesh",10);

insert into depositer values("Nikil",11);

select \* from depositer;



insert into loan values(1,"SBI\_Chamrajpet",1000);

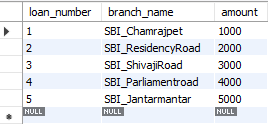
insert into loan values(2,"SBI\_ResidencyRoad",2000);

insert into loan values(3,"SBI\_ShivajiRoad",3000);

insert into loan values(4,"SBI\_Parliamentroad",4000);

insert into loan values(5,"SBI\_Jantarmantar",5000);

select \* from loan;



insert into borrower values(1,"Mohan");

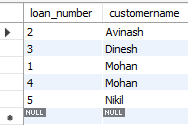
insert into borrower values(2,"Avinash");

insert into borrower values(3,"Dinesh");

insert into borrower values(4,"Mohan");

insert into borrower values(5,"Nikil");

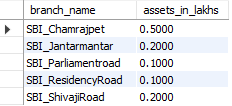
select \* from borrower;



**Queries:**

* **Display the branch name and assets from all branches in lakhs of rupees and rename the assets column to 'assets in lakhs'.**

select branch\_name,(assets/100000) as assets\_in\_lakhs from branch;



* **Find all the customers who have at least two accounts at the same branch (ex. SBI\_ResidencyRoad).**

SELECT d.customername FROM depositer d,bankaccount b

WHERE b.branch\_name = 'SBI\_ResidencyRoad'

AND d.accno = b.accno

GROUP BY d.customername

HAVING COUNT(d.accno) >= 2;



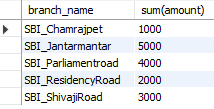
* **Create a view which gives each branch the sum of the amount of all the loans at the branch.**

create view sum\_of\_loan

as select branch\_name,sum(amount) from loan

group by branch\_name;

select \* from sum\_of\_loan;



**Additional Queries on Bank Database**

**Questions (week 4)**

* Branch (branch-name: String, branch-city: String, assets: real)
* Bankaccount(accno: int, branch-name: String, balance: real)
* Bankcustomer (customer-name: String, customer-street: String, customer-city: String)
* Depositer(customer-name: String, accno: int)
* Loan (loan-number: int, branch-name: String, amount: real)
* Find all the customers who have an account at all the branches located in a specific city (Ex. Delhi).
* Find all customers who have a loan at the bank but do not have an account.
* Find all customers who have both an account and a loan at the Bangalore branch
* Find the names of all branches that have greater assets than all branches located in Bangalore.
* Demonstrate how you delete all account tuples at every branch located in a specific city (Ex. Bombay).
* Update the Balance of all accounts by 5%

**Queries**:

* **Find all the customers who have an account at all the branches located in a specific city (Ex. Delhi).**

select distinct s.customername from depositer s

where not exists ((select branch\_name from branch where branch\_city="Delhi") except (select r.branch\_name from depositer t, bankaccount r

where t.accno=r.accno and

s.customername=t.customername));



* Find all customers who have a loan at the bank but do not have an account.

select distinct customername from borrower

where customername not in (select customername from depositer);



* **Find all customers who have both an account and a loan at the Bangalore branch**

SELECT DISTINCT b.customername FROM borrower b, loan l, depositer d, branch br

WHERE b.loan\_number = l.loan\_number

AND l.branch\_name = br.branch\_name

AND br.branch\_city = 'Bangalore'

AND b.customername IN (SELECT customername FROM depositer);



* **Find the names of all branches that have greater assets than all branches located in Bangalore.**

SELECT branch\_name FROM branch

WHERE assets > ALL (SELECT assets FROM branch

WHERE branch\_city = 'Bangalore');

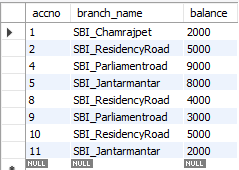


* **Demonstrate how you delete all account tuples at every branch located in a specific city (Ex. Bombay).**

DELETE FROM bankaccount

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

WHERE branch\_city = "Bombay");

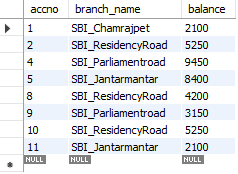


* **Update the Balance of all accounts by 5%**

UPDATE bankaccount

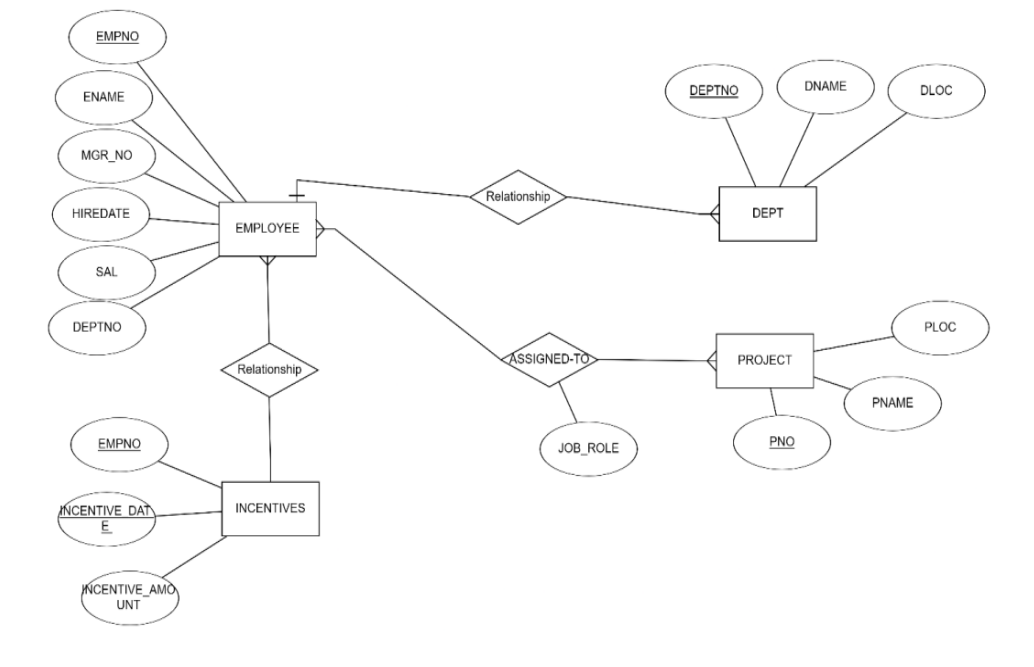
SET balance = balance \* 1.05;

select \* from bankaccount;

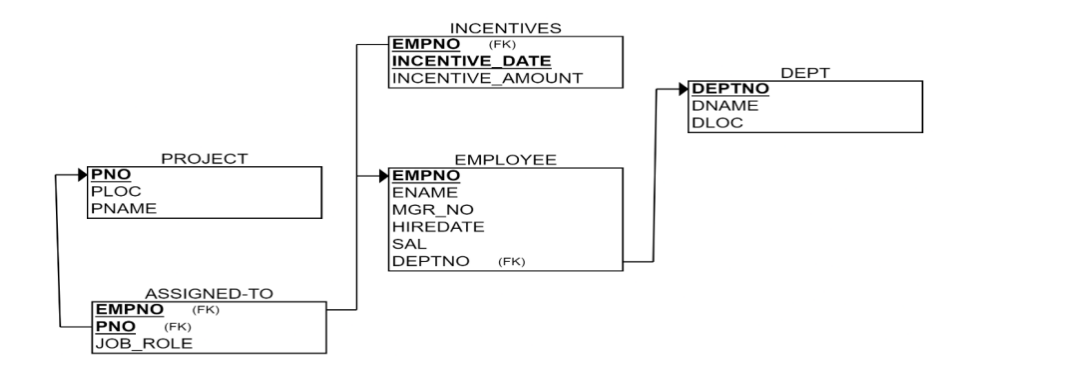


**Employee database**

**ER Diagram:**



**Schema Diagram:**



**Questions (week 5):**

* Using Scheme diagram, create tables by properly specifying the primary keys and the foreign keys.
* Enter greater than five tuples for each table.
* Retrieve the employee numbers of all employees who work on project located in Bengaluru, Hyderabad, or Mysuru
* Get Employee ID’s of those employees who didn’t receive incentives
* Write a SQL query to find the employees name, number, dept, job\_role, department location and project location who are working for a project location same as his/her department location.
* increase income of all employees by 5% in a table.
* Find the names of employee starting with “A”

**Create database:**

create database emp;

use emp;

**Create tables:**

create table dept (

deptno decimal(2 , 0 ) primary key,

dname varchar(14) default null,

loc varchar(13) default null

);

create table emp (

empno decimal(4 , 0 ) primary key,

ename varchar(10) default null,

mgr\_no decimal(4 , 0 ) default null,

hiredate date default null,

sal decimal(7 , 2 ) default null,

deptno decimal(2 , 0 ) references dept (deptno)

on delete cascade on update cascade

);

create table incentives (

empno decimal(4 , 0 ) references emp (empno)

on delete cascade on update cascade,

incentive\_date date,

incentive\_amount decimal(10 , 2 ),

primary key (empno , incentive\_date)

);

create table project (

pno int primary key,

pname varchar(30) not null,

ploc varchar(30)

);

create table assigned\_to (

empno decimal(4 , 0 ) references emp (empno)

on delete cascade on update cascade,

pno int references project (pno)

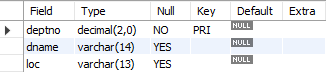
on delete cascade on update cascade,

job\_role varchar(30),

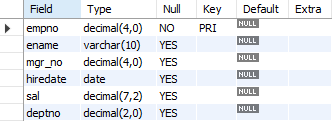
primary key (empno , pno));

**Structure of table:**

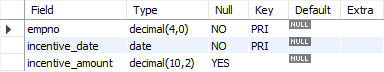
desc dept;



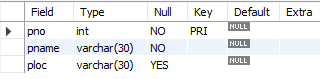
desc emp;



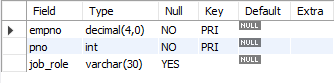
desc incentives;



desc project;



desc assigned\_to;



**Inserting values into tables:**

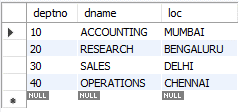
insert into dept values (10, 'accounting', 'mumbai');

insert into dept values (20, 'research', 'bengaluru');

insert into dept values (30, 'sales', 'delhi');

insert into dept values (40, 'operations', 'chennai');

select \* from dept;



insert into emp values (7369, 'adarsh', 7902, '2012-12-17', '80000.00', '20');

insert into emp values (7499, 'shruthi', 7698, '2013-02-20', '16000.00', '30');

insert into emp values (7521, 'anvitha', 7698, '2015-02-22', '12500.00', '30');

insert into emp values (7566, 'tanvir', 7839, '2008-04-02', '29750.00', '20');

insert into emp values (7654, 'ramesh', 7698, '2014-09-28', '12500.00', '30');

insert into emp values (7698, 'kumar', 7839, '2015-05-01', '28500.00', '30');

insert into emp values (7782, 'clark', 7839, '2017-06-09', '24500.00', '10');

insert into emp values (7788, 'scott', 7566, '2010-12-09', '30000.00', '20');

insert into emp values (7839, 'king', null, '2009-11-17', '90000', '10');

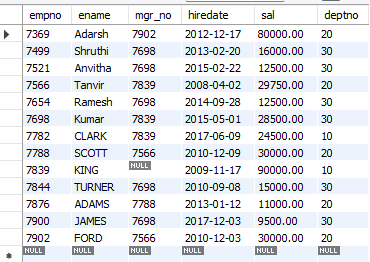
insert into emp values (7844, 'turner', 7698, '2010-09-08', '15000.00', '30');

insert into emp values (7876, 'adams', 7788, '2013-01-12', '11000.00', '20');

insert into emp values (7900, 'james', 7698, '2017-12-03', '9500.00', '30');

insert into emp values (7902, 'ford', 7566, '2010-12-03', '30000.00', '20');

select \* from emp;



insert into incentives values(7499, '2019-02-01', 5000.00);

insert into incentives values(7521, '2019-03-01', 2500.00);

insert into incentives values(7566, '2022-02-01', 5070.00);

insert into incentives values(7654, '2020-02-01', 2000.00);

insert into incentives values(7654, '2022-04-01', 879.00);

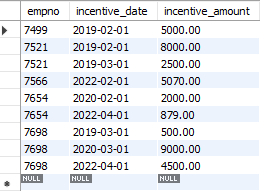
insert into incentives values(7521, '2019-02-01', 8000.00);

insert into incentives values(7698, '2019-03-01', 500.00);

insert into incentives values(7698, '2020-03-01', 9000.00);

insert into incentives values(7698, '2022-04-01', 4500.00);

select \* from incentives;



insert into project values(101, 'ai project', 'bengaluru');

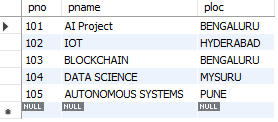
insert into project values(102, 'iot', 'hyderabad');

insert into project values(103, 'blockchain', 'bengaluru');

insert into project values(104, 'data science', 'mysuru');

insert into project values(105, 'autonomous systems', 'pune');

select \* from project;



insert into assigned\_to values(7499, 101, 'software engineer');

insert into assigned\_to values(7521, 101, 'software architect');

insert into assigned\_to values(7566, 101, 'project manager');

insert into assigned\_to values(7654, 102, 'sales');

insert into assigned\_to values(7521, 102, 'software engineer');

insert into assigned\_to values(7499, 102, 'software engineer');

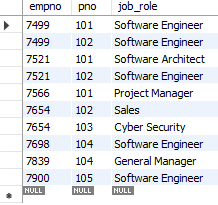
insert into assigned\_to values(7654, 103, 'cyber security');

insert into assigned\_to values(7698, 104, 'software engineer');

insert into assigned\_to values(7900, 105, 'software engineer');

insert into assigned\_to values(7839, 104, 'general manager');

select \* from assigned\_to;



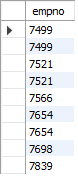
**Queries:**

* **Retrieve the employee numbers of all employees who work on project located in Bengaluru, Hyderabad, or Mysuru**

select e.empno from emp e, project p,assigned\_to a

where e.empno = a.empno AND a.pno = p.pno

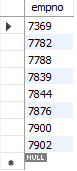
and p.ploc in ('BENGALURU' , 'HYDERABAD', 'MYSURU');



* **Get Employee ID’s of those employees who didn’t receive incentives.**

select empno from emp

where empno not in (select empno from incentives);



* **Write a SQL query to find the employees name, number, dept, job\_role, department location and project location who are working for a project location same as his/her department location.**

select e.empno , e.ename,d.dname,d.loc,a.job\_role,p.ploc

from emp e,dept d,assigned\_to a,project p

where e.deptno=d.deptno and

e.empno=a.empno and

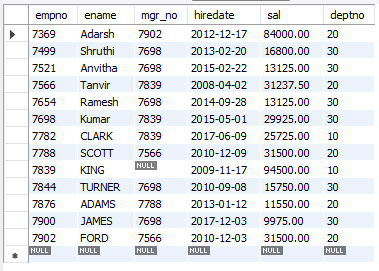
a.pno=p.pno and d.loc=p.ploc;



* **Increase income of all employees by 5% in a table.**

update emp set sal = 1.05\*sal;

select \* from emp;



* **Find the names of employee starting with “A”**

select ename from emp

where ename like 'A%';



**More Queries on Employee Database**

**Questions (Week 6)**

* Using Scheme diagram (under Program-5), Create tables by properly specifying the

primary keys and the foreign keys.

* Enter greater than five tuples for each table.
* List the name of the managers with the maximum employees
* Display those managers name whose salary is more than average salary of his employee.
* Find the name of the second top level managers of each department.
* Find the employee details who got second maximum incentive in January 2019.
* Display those employees who are working in the same department where his manager is

working.

**Queries:**

* **Enter greater than five tuples for each table.**

select e.ename as manager\_name, count(emp.empno) as num\_employees from emp e

join emp as emp on e.empno = emp.mgr\_no

group by e.ename

having count(emp.empno) = ( select max(employee\_count) from (

select count(empno) as employee\_count from emp

where mgr\_no is not null

group by mgr\_no

) as counts);



* **Display those managers name whose salary is more than average salary of his employee.**

select e.empno,e.ename as manager\_name, e.sal as manager\_salary

from emp e

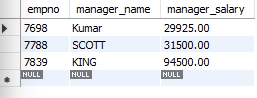
where e.empno in (select mgr\_no from emp

where mgr\_no is not null

group by mgr\_no

having e.sal > avg(sal)

);



* **Find the name of the second top level managers of each department.**

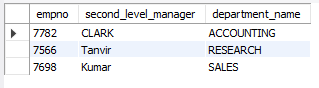
select e2.ename as second\_level\_manager, d.dname as department\_name

from emp e1

join emp e2 on e1.empno = e2.mgr\_no

join dept d on e2.deptno = d.deptno

where e1.mgr\_no is null;



* **SQL Query to find the name of the top-level manager of each department.**

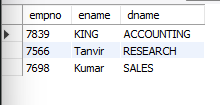
select e.empno, e.ename, d.dname from emp e, dept d

where d.deptno = e.deptno

and e.empno in (select mgr\_no from emp)

and e.deptno not in (select m.deptno from emp m

where e.mgr\_no = m.empno);



* **SQL Query to find the employee details who got second maximum incentive in February 2019.**

select e.ename, i.incentive\_amount from emp e

join incentives i on e.empno = i.empno

where i.incentive\_date like '2019-02%'

and i.incentive\_amount = (

select incentive\_amount

from incentives

where incentive\_date like '2019-02%'

order by incentive\_amount desc

limit 1 offset 1

);



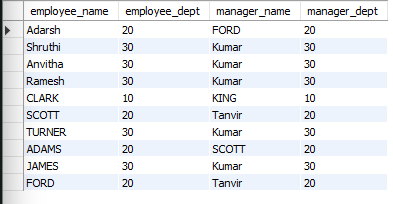
* **Display those employees who are working in the same dept where his manager is working.**

select e.ename as employee\_name, e.deptno as employee\_dept,

m.ename as manager\_name, m.deptno as manager\_dept from emp e

join emp m on e.mgr\_no = m.empno

where e.deptno = m.deptno;

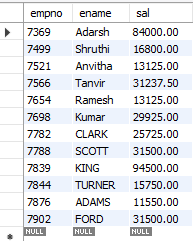


* **Write a SQL query to find those employees whose net pay are higher than or equal to the salary of any other employee in the company.**

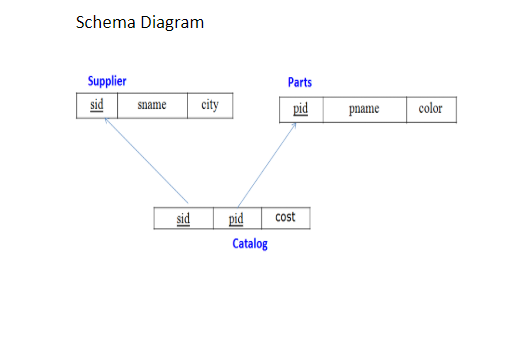
select e.empno, e.ename, e.sal from emp e

where e.sal >= any (select m.sal from emp m

where m.empno not in (select p.empno from emp p where p.empno = e.empno));



**Supplier Database**

****

**Queries (Week 7):**

* Using Scheme diagram, create tables by properly specifying the primary keys and the foreign keys.

Insert appropriate records in each table.

* Find the pnames of parts for which there is some supplier.
* Find the snames of suppliers who supply every part.
* Find the snames of suppliers who supply every red part.
* Find the pnames of parts supplied by Acme Widget Suppliers and by no one else.
* Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over all the suppliers who supply that part).
* For each part, find the sname of the supplier who charges the most for that part.

**Create Database:**

create database supplier;

use supplier;

**Create tables:**

CREATE TABLE Supplier (

sid INT PRIMARY KEY,

sname VARCHAR(50),

city VARCHAR(50)

);

CREATE TABLE Parts (

pid INT PRIMARY KEY,

pname VARCHAR(50),

color VARCHAR(20)

);

CREATE TABLE Catalog (

sid INT,

pid INT,

cost DECIMAL(10, 2),

PRIMARY KEY (sid, pid),

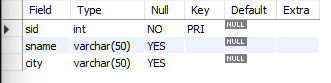
FOREIGN KEY (sid) REFERENCES Supplier(sid),

FOREIGN KEY (pid) REFERENCES Parts(pid)

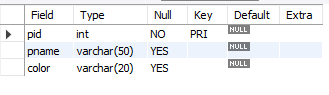
);

**Structure of tables:**

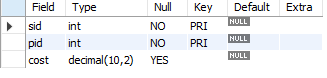
desc supplier;



desc parts;



desc catalog;



**Inserting Values into tables:**

INSERT INTO Supplier VALUES (10001, 'Acme Widget', 'Bangalore');

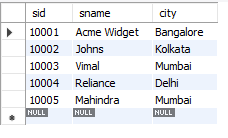
INSERT INTO Supplier VALUES (10002, 'Johns', 'Kolkata');

INSERT INTO Supplier VALUES (10003, 'Vimal', 'Mumbai');

INSERT INTO Supplier VALUES (10004, 'Reliance', 'Delhi');

INSERT INTO Supplier VALUES (10005, 'Mahindra', 'Mumbai');

select \* from supplier;



INSERT INTO Parts VALUES (20001, 'Book', 'Red');

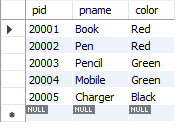
INSERT INTO Parts VALUES (20002, 'Pen', 'Red');

INSERT INTO Parts VALUES (20003, 'Pencil', 'Green');

INSERT INTO Parts VALUES (20004, 'Mobile', 'Green');

INSERT INTO Parts VALUES (20005, 'Charger', 'Black');

select \* from parts;



INSERT INTO Catalog VALUES (10001, 20001, 10);

INSERT INTO Catalog VALUES (10001, 20002, 10);

INSERT INTO Catalog VALUES (10001, 20003, 30);

INSERT INTO Catalog VALUES (10001, 20004, 10);

INSERT INTO Catalog VALUES (10001, 20005, 10);

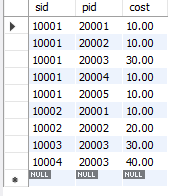
INSERT INTO Catalog VALUES (10002, 20001, 10);

INSERT INTO Catalog VALUES (10002, 20002, 20);

INSERT INTO Catalog VALUES (10003, 20003, 30);

INSERT INTO Catalog VALUES (10004, 20003, 40);

select \* from catalog;

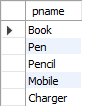


**Queries**:

* **Find the pnames of parts for which there is some supplier.**

select distinct pname from parts

where pid in (select pid from catalog);



* **Find the snames of suppliers who supply every part.**

select sname from supplier

where not exists (select pid from parts

where pid not in (select pid from catalog

where catalog.sid = supplier.sid));



* **Find the snames of suppliers who supply every red part.**

select sname from supplier

where not exists (select pid from parts

where color = 'red' and pid not in (select pid from catalog

where catalog.sid = supplier.sid));



* **Find the pnames of parts supplied by Acme Widget Suppliers and by no one else**

select pname from parts

where pid in ( select pid from catalog

where sid = (select sid from supplier

where sname = 'acme widget'))

and pid not in (select pid from catalog

where sid != (select sid from supplier where sname = 'acme widget'));



* **Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over all the suppliers who supply that part).**

select distinct c.sid from catalog c

join (select pid, avg(cost) as avg\_cost from catalog

group by pid) avg\_table on c.pid = avg\_table.pid

where c.cost > avg\_table.avg\_cost;



* **For each part, find the sname of the supplier who charges the most for that part.**

select p.pname,p.pid, s.sname from parts p

join catalog c on p.pid = c.pid

join supplier s on c.sid = s.sid

where (p.pid, c.cost) in (select pid, max(cost) from catalog

group by pid);

