**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

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

****

**LAB REPORT**

**on**

**Database Management Systems (23CS3PCDBM)**

***Submitted by***

**BHOOMIKA.M(1BM23CS068)**

***in partial fulfillment 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 **BHOOMIKA.M (1BM23CS068),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment 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 KVN Name  Assistant Professor  Department of CSE, BMSCE | Dr. Joythi S Nayak  Professor HOD  Department of CSE, BMSCE |

**Index**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.**  **No.** | **Date** | **Experiment Title** | **Page No.** |
| 1 |  | Insurance Database | 04-09 |
| 2 |  | More Queries on Insurance Database | 10-11 |
| 3 |  | Bank Database | 12-18 |
| 4 |  | More Queries on Bank Database | 19-21 |
| 5 |  | Employee Database | 22-28 |
| 6 |  | More Queries on Employee Database | 29-31 |
| 7 |  | Supplier Database | 32-37 |
| 8 |  | NO SQL - Student Database | 38-40 |
| 9 |  | NO SQL - Customer Database | 4142 |
| 10 |  | NO SQL – Restaurant Database | 43-47 |

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

**-** Update the damage amount to 25000 for the car with a specific reg\_num (example 'K A053408' ) for which the accident report number was 12.

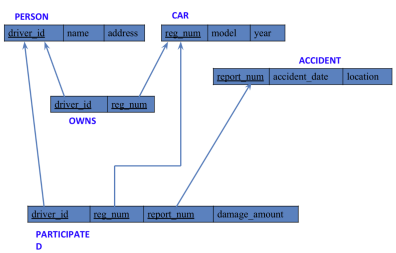
**-** Add a new accident to the database.

**-** 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 database**

create database bhoomika\_cs068;

use bhoomika\_cs068;

**Create table**

create table person (

driver\_id varchar(10),

name varchar(20),

address varchar(20),

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),

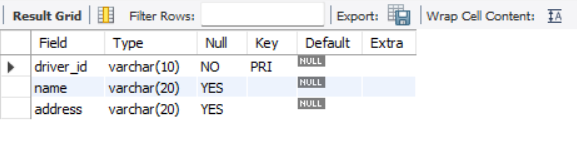
foreign key (reg\_num) references car(reg\_num),

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

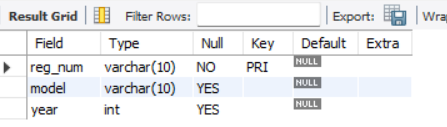
);

**Structure of the table**

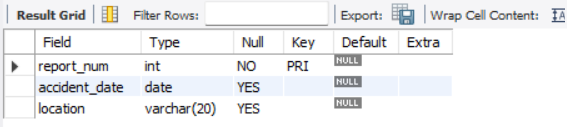
desc person;



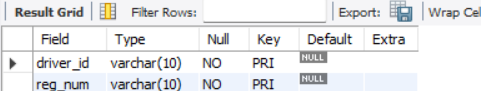
desc car;



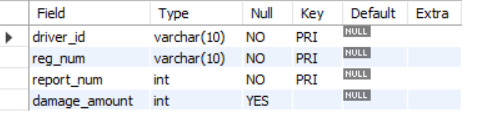
desc accident;



desc owns;



desc participated;



**Inserting Values to the table**

insert into person

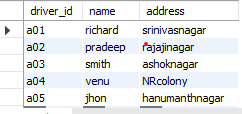
values(‘a01’,’richard’,’srinivasnagar’),

(‘a02’,’pradeep’,’rajajinagar’),

(‘a03’,’smith’,’ ashoknagar’),

(‘a04’,’venu’,’NRcolony’),

(‘a05’,’jhon’,’hanumanthnagar’);



insert into car

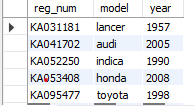
values ('KA052250', 'indica',1990),

('KA031181','lancer',1957),

('KA095477','toyota',1998),

('KA053408','honda',2008),

('KA041702','audi',2005);



insert into accident

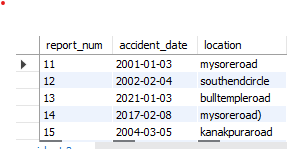
values(11,'2001-01-03','mysoreroad'),

(12,'2002-02-04','southendcircle'),

(13,'2021-01-03','bulltempleroad'),

(14,'2017-02-08','mysoreroad)'),

(15,'2004-03-05','kanakpuraroad');



insert into owns

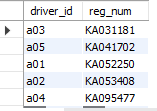
values('a01','KA052250'),

('a02','KA053408'),

('a03','KA031181'),

('a04','KA095477'),

('a05','KA041702');



insert into participated

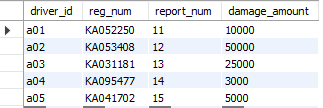
values('a01','KA052250',11,10000),

('a02','KA053408',12,50000),

('a03','KA031181',13,25000),

('a04','KA095477',14,3000),

('a05','KA041702',15,5000);

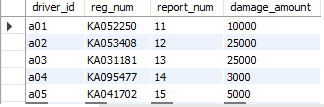


**Queries**

* Update the damage amount to 25000 for the car with a specific reg-num (example 'KA053408' ) for which the accident report number was 12.

update participated set damage\_amount=25000

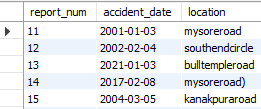
where reg\_num='KA053408' and report\_num=12;



* Add a new accident to the database.

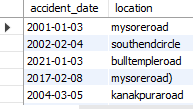
insert into accident

values (16,'2015-03-08','domlur');



* Display Accident date and location

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

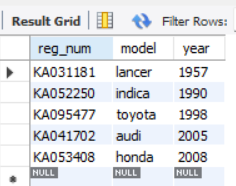
where damage\_amount>=25000;



**WEEK -02**

\*select \* from car

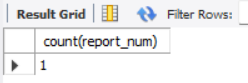
order by year asc;



select count(report\_num)

from car c , participated p

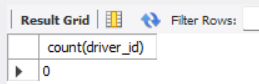
where c.reg\_num=p.reg\_num and c.model='lancer';



select count(driver\_id)

from accident a,participated p

where a.report\_num=p.report\_num and a.accident\_date like '\_\_08%';

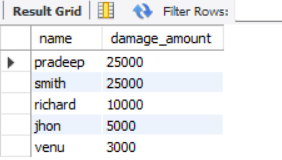


select name,damage\_amount

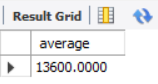
from person a ,participated b

where a.driver\_id =b.driver\_id

order by damage\_amount desc;



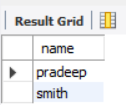
select avg(damage\_amount) as 'average' from participated;



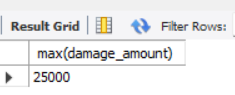
select name

from person a ,participated p

where a.driver\_id=p.driver\_id and p.damage\_amount >(select avg(damage\_amount) from participated);



select max(damage\_amount) from participated;



**BANK ACCOUNT**

**QUESTION**

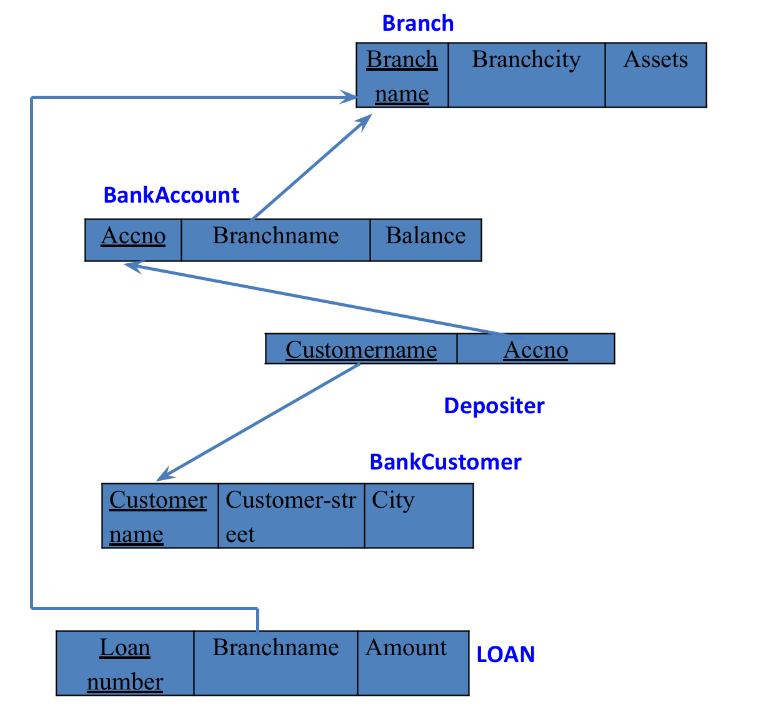
**(WEEK 3)**

* 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.

**SCHEMA DIAGRAM**

****

**CREATE DATABASE**

create database bhoomika\_cs068;

use bhoomika\_cs068;

**CREATE TABLES**

create table branch (

branchname varchar(50),

branchcity varchar(50),

assests int ,

primary key (branchname));

create table bankcustomer(

customername varchar(50),

customer\_street varchar(50),

city varchar(50),

primary key(customername));

create table bankaccount (

accno int,

branchname varchar(50),

balance int,

primary key (accno),

foreign key (branchname) references branch (branchname));

create table depositer(

customername varchar(50),

accno int,

primary key (customername, accno),

foreign key (customername) references bankcustomer(customername),

foreign key (accno) references bankaccount(accno));

create table loan(

loannumber int,

branchname varchar(50),

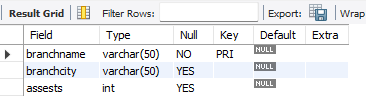
amount int,

primary key (loannumber),

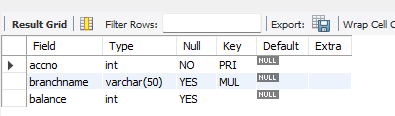
foreign key (branchname) references branch (branchname));

**STRUCTURE OF TABLE**

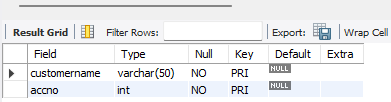
desc branch;



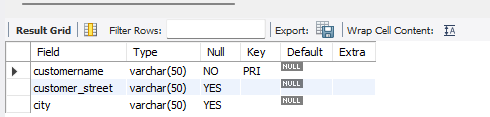
desc bankaccount;



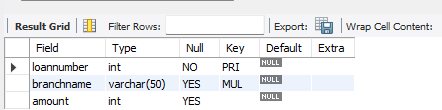
desc depositer;



desc bankcustomer;



desc loan;



**INSERTING VALUES INTO THE TABLE**

insert into branch

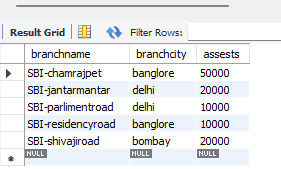
values('SBI-chamrajpet','banglore', 50000),

('SBI-residencyroad','banglore',10000),

('SBI-shivajiroad','bombay',20000),

('SBI-parlimentroad','delhi',10000),

('SBI-jantarmantar','delhi',20000);



insert into bankcustomer

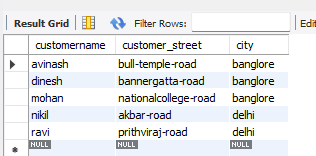
values('avinash','bull-temple-road','banglore'),

('dinesh','bannergatta-road','banglore'),

('mohan','nationalcollege-road','banglore'),

('nikil','akbar-road','delhi'),

('ravi','prithviraj-road','delhi');



insert into bankaccount

values(1,'SBI-chamrajpet',2000),

(2,'SBI-residencyroad',5000),

(3,'SBI-shivajiroad',6000),

(4,'SBI-parlimentroad',9000),

(5,'SBI-jantarmantar',8000),

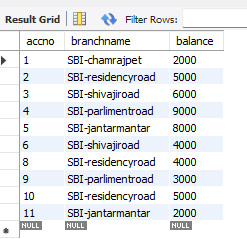
(6,'SBI-shivajiroad',4000),

(8,'SBI-residencyroad',4000),

(9,'SBI-parlimentroad',3000),

(10,'SBI-residencyroad',5000),

(11,'SBI-jantarmantar',2000);



insert into depositer

values('avinash',1),

('dinesh',2),

('nikil',4),

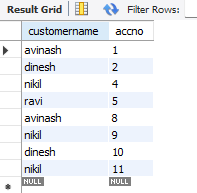
('ravi',5),

('avinash',8),

('nikil',9),

('dinesh',10),

('nikil',11);



insert into loan

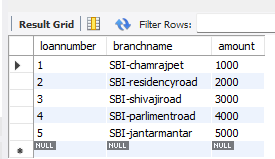
values(1,'SBI-chamrajpet',1000),

(2,'SBI-residencyroad',2000),

(3,'SBI-shivajiroad',3000),

(4,'SBI-parlimentroad',4000),

(5,'SBI-jantarmantar',5000);

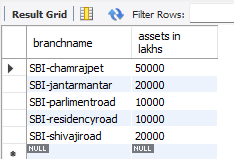


**QUERIES**

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

select branchname,assests as 'assets in lakhs'

from  branch;



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

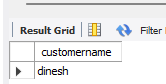
select d.customername

from bankaccount b, depositer d

where b.accno=d.accno and branchname='SBI-residencyroad'

group by customername

having count(\*)>=2;



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

create view loan\_info as

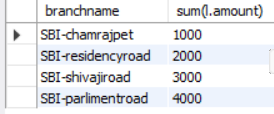
select b.branchname, sum(l.amount)

from branch b , loan l

where b.branchname=l.branchname

group by l.branchname;

select \* from loan\_info;

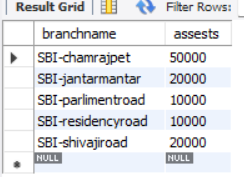


WEEK -04

1. Retrieve all branches and their respective total assets

select branchname, assests

from branch;

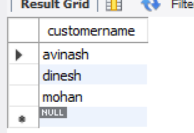


1. List all customers who live in a particular city

select customername

from bankcustomer

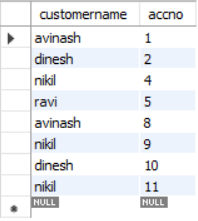
where city='banglore';



1. List all customers with their account numbers

select customername ,accno

from depositer ;



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

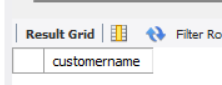
select c.customername

from bankcustomer c, depositer d, bankaccount a, branch b

where c.customername=d.customername and d.accno=a.accno and a.branchname=b.branchname and b.branchname=all(select b.branchname

from branch b

where b.branchcity='delhi');

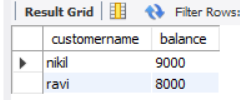


1. Find all customers who have accounts with a balance greater than a specified amount (5000)

select c.customername, b.balance

from bankcustomer c, bankaccount b, depositer d

where d.accno=b.accno and c.customername=d.customername and b.balance>5000;

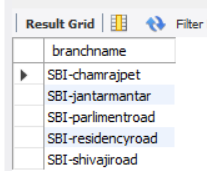


1. List all branch who have both a loan and an account

select distinct(b.branchname)

from branch b, bankaccount a, loan l

where b. branchname=a.branchname and b.branchname=l.branchname;

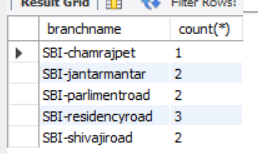


1. Get the number of accounts held at each branch

select branchname , count(\*)

from bankaccount

group by branchname;



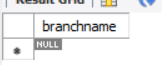
1. Find all branches that have no loans issued

select b.branchname

from branch b

where b.branchname not in(select branchname

from loan);



1. Retrieve the branch with the smallest total loan amount

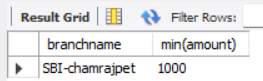
select branchname ,min(amount)

from loan

group by branchname

order by min(amount)

limit 1;



**EMPLOYEE DATABASE**

**(WEEL -05)**

**QUESTION**

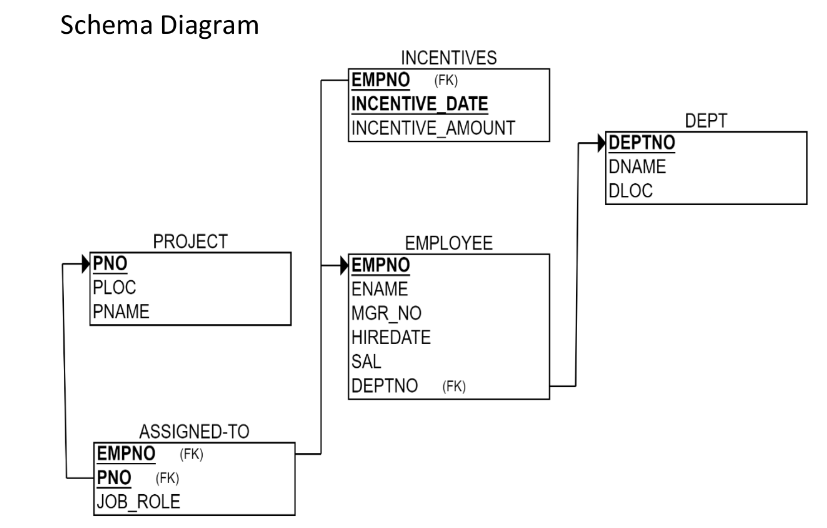
1. Using Scheme diagram, Create tables by properly specifying the primary keys and the foreign keys.

2. Enter greater than five tuples for each table.

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

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

5. 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.



**CREATE DATABASE**

Create database Bhoomika\_cs068;

Use database Bhoomika\_cs068;

**CREATE TABLES**

create table project(

pno int,

ploc varchar(50),

pname varchar(50),

primary key (pno));

create table dept(

deptno int primary key,

dname varchar(50),

dloc varchar(50));

create table employee(

empno int primary key,

empname varchar(50),

mgr\_no int,

hiredate date,

sal int,

deptno int,

foreign key (deptno) references dept (deptno));

create table incentives(

empno int ,

incentive\_date date ,

incentive\_amt int,

primary key(empno,incentive\_date),

foreign key (empno) references employee (empno));

create table assigned\_to(

empno int,

pno int,

job\_role varchar (50),

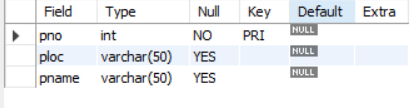
primary key (empno, pno),

foreign key (empno) references employee(empno),

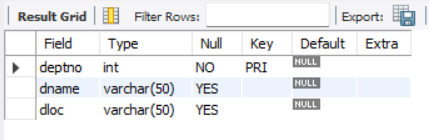
foreign key (pno) references project (pno));

**STRUCTURE OF TABLE**

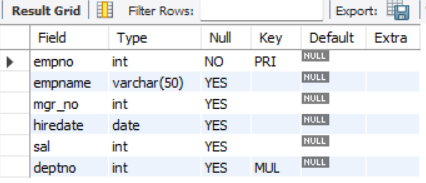
Desc project;

****

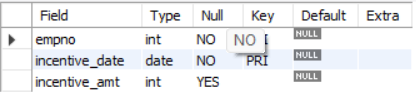
desc dept;



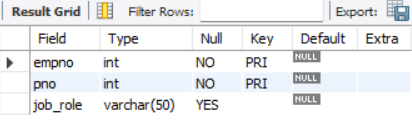
Desc employee;



Desc incentives;



Desc assigned\_to;



**INSERT TO TABLE**

insert into project

values(01,'bengaluru','apx'),

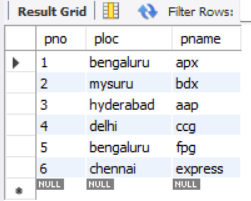
(02,'mysuru','bdx'),

(03,'hyderabad','aap'),

(04,'delhi','ccg'),

(05,'bengaluru','fpg'),

(06,'chennai','express');



insert into dept

values(01,'cse','bengaluru'),

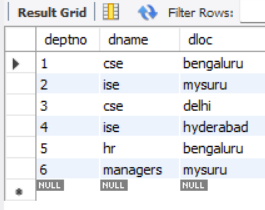
(02,'ise','mysuru'),

(03,'cse','delhi'),

(04,'ise','hyderabad'),

(05,'hr','bengaluru'),

(06,'managers','mysuru');



insert into employee

values (1,'bhoomika',3,'2015-04-19',15000,05),

(2,'bhoomi',7,'2016-07-20',70000,02),

(3,'divya',NULL,'2000-07-22',10000,05),

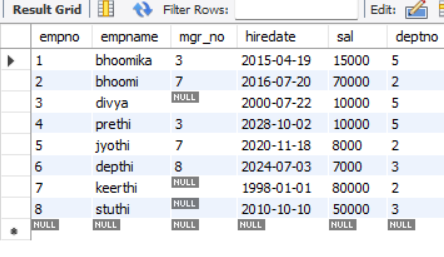
(4,'prethi',3,'2028-10-02',10000,05),

(5,'jyothi',7,'2020-11-18',8000,02),

(6,'depthi',8,'2024-07-03',7000,03),

(7,'keerthi',NULL,'1998-01-01',80000,02),

(8,'stuthi',NULL,'2010-10-10',50000,03);



insert into incentives

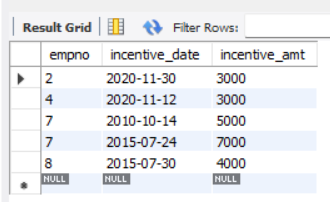
values(4,'2020-11-12',3000),

(8,'2015-07-30',4000),

(7,'2010-10-14',5000),

(7,'2015-07-24',7000),

(2,'2020-11-30',3000);



insert into assigned\_to

values(7,02,'manager'),

(8,01,'data analyst'),

(3,04,'worker'),

(2, 06,'worker'),

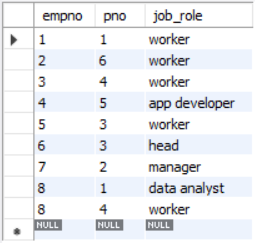
(4,05,'app developer'),

(8,04,'worker'),

(1,01,'worker'),

(5,03,'worker'),

(6,03,'head');



**QUERIES**

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

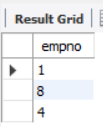
select e.empno

from employee e, assigned\_to a

where e.empno=a.empno and a.pno in(select pno

from project

where ploc in ('bengaluru' ,' hyderabad',' mysuru'));



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

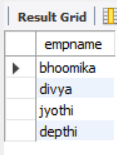
select empname

from employee

where not exists(select 1

from incentives

where empno=employee.empno);

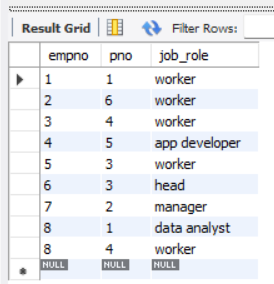


1. Write a SQL query to find the employees name, number, dept, job\_role, department location and project location who are working for a project location same as his/her department location.

select e.empno, e.empname, d.deptno, a.job\_role, d.dloc ,p.ploc

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

where e.empno=a.empno and p.pno=a.pno and e.deptno=d.deptno and d.dloc=p.ploc;



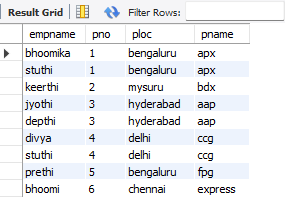
**(WEEK-06)**

1. List all employees along with their project details (if assigned)

select e.empname, p.\*

from employee e , project p, assigned\_to a

where e.empno=a.empno and p.pno=a.pno;



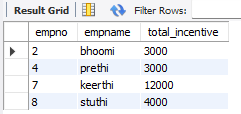
1. Find all employees who received incentives, along with the total incentive amount

select i.empno, e.empname, sum(i.incentive\_amt) as total\_incentive

from incentives i, employee e

where i.empno=e.empno

group by empno;

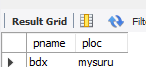


1. Retrieve the project names and locations of projects with employees assigned as 'Manager'

select p.pname, p.ploc

from project p

where pno in (select pno from assigned\_to a where job\_role='manager');



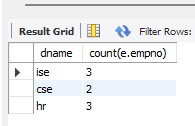
1. List departments along with the number of employees in each department

select d.dname, count(e.empno)

from dept d, employee e

where d.deptno =e. deptno

group by d.dname;

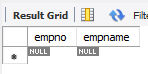


1. Find employees who have not been assigned to any project

select e.empno, e.empname

from employee e

where not exists (select 1 from assigned\_to a where e.empno=a.empno);

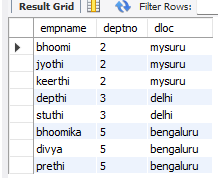


1. List all employees along with their department names and location

select e.empname , d.deptno, d.dloc

from employee e , dept d

where e.deptno=d.deptno;

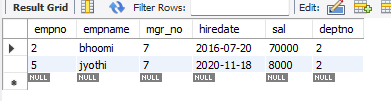


1. Retrieve the details of employees who work under a specific manager

select \*

from employee e

where mgr\_no=7;



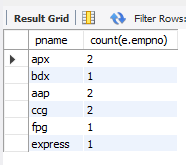
1. List all projects that have employees assigned and the number of employees on each project:

select p.pname, count(e.empno)

from project p, assigned\_to e

where e.pno=p.pno

group by p.pname;

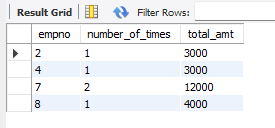


1. List the total number of incentives given to each employee and the sum of incentives for each

select empno, count(incentive\_date) as number\_of\_times, sum(incentive\_amt) as total\_amt

from incentives

group by empno;

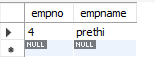


1. Retrieve all employees who have the role of 'Developer' on any project

select e.empno, e.empname

from employee e

where e.empno in (select empno from assigned\_to where empno=e.empno and job\_role='app developer');

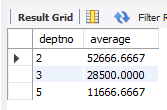


1. Display the department-wise average salary of employees:

select deptno , avg(sal) as average

from employee

group by deptno;



**SUPPLIERS DATABASE**

**(WEEK -07)**

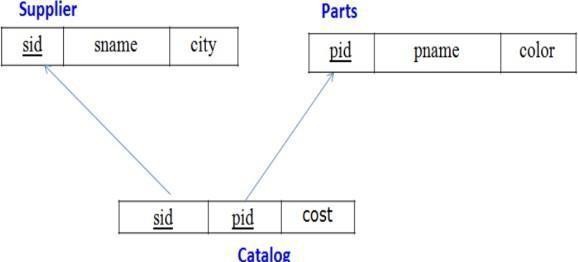
**QUESTION**

1. Using Scheme diagram, Create tables by properly specifying the primary keys and the foreign keys.
2. Insert appropriate records in each table.

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

1. Find the snames of suppliers who supply every part.
2. Find the snames of suppliers who supply every red part.
3. Find the pnames of parts supplied by Acme Widget Suppliers and by no one else
4. 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)
5. For each part, find the sname of the supplier who charges the most for that part

### Schema Diagram:

****

**Create Database:**

create database supp;

use supp;

### Create Tables:

create table Supplier(

s\_id int primary key,

s\_name varchar(30),

city varchar(20));

create table Parts( p\_id int primary key, p\_name varchar(30), color varchar(30));

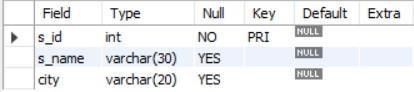
create table Catalog( s\_id int,

p\_id int, cost float,

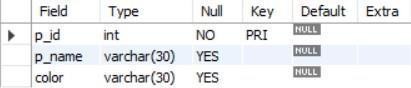
foreign key(s\_id) references Supplier(s\_id), foreign key(p\_id) references Parts(p\_id));

### Structure of the Table:

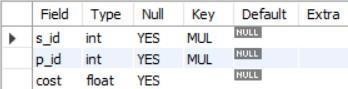
desc Supplier;



desc Parts;



desc Catalog;



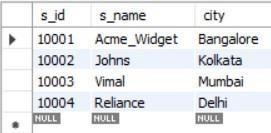
### Inserting Values to the tables:

insert into Supplier values

(10001, 'Acme\_Widget', 'Bangalore'),

(10002, 'Johns', 'Kolkata'),

(10003, 'Vimal', 'Mumbai'),

(10004, 'Reliance', 'Delhi'); select \* from Supplier;

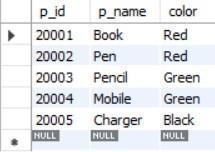
insert into Parts values (20001, 'Book', 'Red'),

(20002, 'Pen', 'Red'),

(20003, 'Pencil', 'Green'),

(20004, 'Mobile', 'Green'),

(20005, 'Charger', 'Black');



insert into Catalog values (10001, 20001, 10),

(10001, 20002, 10),

(10001, 20003, 30),

(10001, 20004, 10),

(10001, 20005, 10),

(10002, 20001, 10),

(10002, 20002, 20),

(10003, 20003, 30),

(10004, 20003, 40);

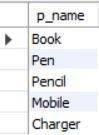


### Queries:

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

select distinct p.p\_name

from Supplier s, Catalog c, Parts p where s.s\_id = c.s\_id and

p.p\_id = c.p\_id and c.s\_id is not null;

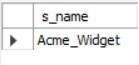
#### Find the snames of suppliers who supply every part.

select distinct s\_name

from Supplier s, Catalog c, Parts p where s.s\_id = c.s\_id

group by s.s\_id, s.s\_name

having count(distinct c.p\_id)=(select count(\*) from Parts p);

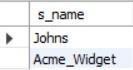


#### Find the snames of suppliers who supply every red part.

select distinct s\_name

from Supplier s, Catalog c, Parts p where s.s\_id = c.s\_id and

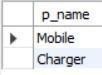
c.p\_id in (select p\_id from Parts p where p.color = 'Red')



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

select distinct p\_name from Supplier s, Parts p, Catalog c where p.p\_id in (select c.p\_id from Catalog c, Supplier s where

s.s\_id = c.s\_id and s.s\_name = 'Acme\_Widget') and

p.p\_id not in (select c.p\_id from Catalog c, Supplier s where s.s\_id = c.s\_id and s.s\_name != '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)

create view Average(p\_id, Average\_Product\_Cost) as select c.p\_id, avg(cost)

from Catalog c group by c.p\_id;

select c.s\_id from Catalog c, Average a where c.p\_id = a.p\_id and c.cost>(a.Average\_Product\_Cost)

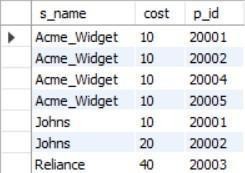
group by c.p\_id, c.s\_id;



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

select distinct s.s\_name, c.cost, c.p\_id from Catalog c, Supplier s where s.s\_id = c.s\_id and

c.cost in (select max(cost) from Catalog c group by c.p\_id);



**NO SQL STUDENT DATABASE**

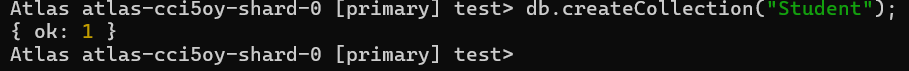
**(WEEK -08)**

**QUESTION**

* Perform the following DB operations using MongoDB.
* Create a database “Student” with the following attributes Rollno, Age, ContactNo, Email-Id.
* Insert appropriate values
* Write query to update Email-Id of a student with rollno 10.
* Replace the student name from “ABC” to “FEM” of rollno 11.

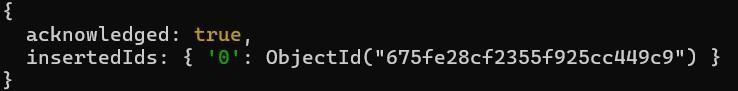
### Create Database:

db.createCollection("Student");

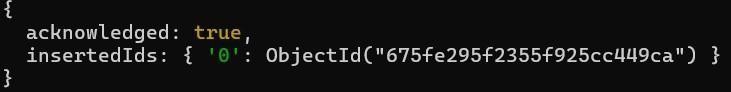


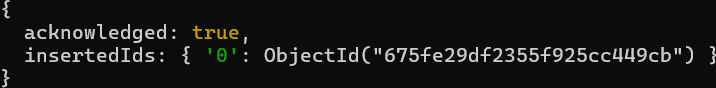
### Inserting Values to the tables:

db.Student.insert({RollNo:1,Age:21,Cont:9876,email:"[antara.de9@gmail.com](mailto:antara.de9@gmail.com)"});

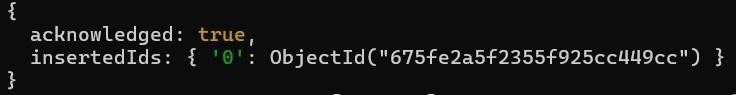


db.Student.insert({RollNo:2,Age:22,Cont:9976,email:"[anushka.de9@gmail.com](mailto:anushka.de9@gmail.com)"});

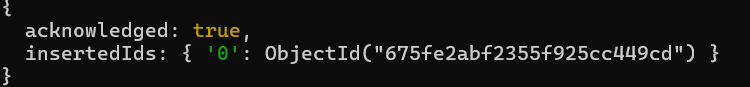


db.Student.insert({RollNo:3,Age:21,Cont:5576,email:"[anubhav.de9@gmail.com](mailto:anubhav.de9@gmail.com)"});

db.Student.insert({RollNo:4,Age:20,Cont:4476,email:"[pani.de9@gmail.com](mailto:pani.de9@gmail.com)"});



db.Student.insert({RollNo:10,Age:23,Cont:2276,email:"[rekha.de9@gmail.com](mailto:rekha.de9@gmail.com)"});



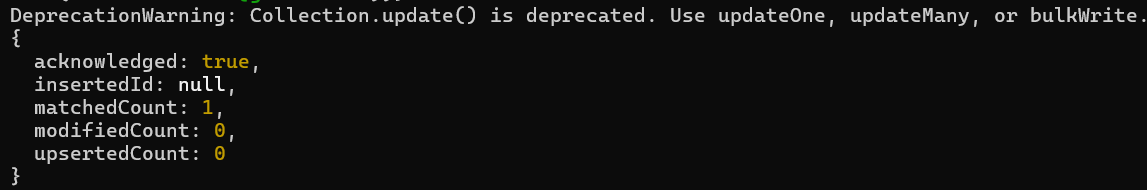
### Queries:

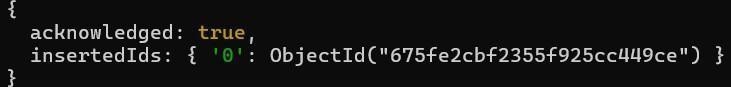
db.Student.find()



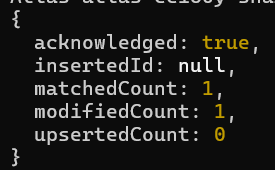
#### Write query to update Email-Id of a student with rollno 10.

db.Student.update({RollNo:10},{$set:{email:"[Abhinav@gmail.com](mailto:Abhinav@gmail.com)"}})



**Replace the student name from “ABC” to “FEM” of rollno 11.** db.Student.insert({RollNo:11,Age:22,Name:"ABC",Cont:2276,email:"[rea.de9@gmail.](mailto:rea.de9@gmail.com) [c](mailto:rea.de9@gmail.com) [om](mailto:rea.de9@gmail.com)"});

db.Student.update({RollNo:11,Name:"ABC"},{$set:{Name:"FEM"}})



# **NO SQL CUSTOMERS DATABASE**

**(WEEK -09)**

**QUESTION**

* + Create a collection by name Customers with the following attributes. Cust\_id, Acc\_Bal, Acc\_Type
  + Insert at least 5 values into the table
  + Write a query to display those records whose total account balance is greater than 1200 of account type ‘Z’ for each customer\_id.
  + Determine Minimum and Maximum account balance for each customer\_id.
  + Export the created collection into local file system
  + Drop the table
  + Import a given csv dataset from local file system into mongodb collection.

### Create Database:

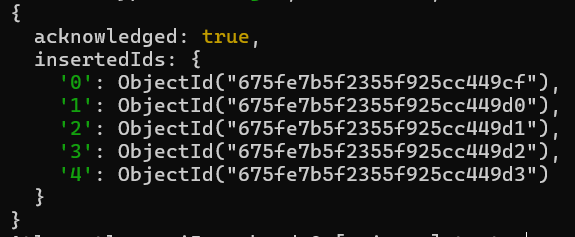
db.createCollection("Customer");



### Inserting Values to the tables:

db.Customer.insertMany([{custid: 1, acc\_bal:10000, acc\_type:"Saving"},

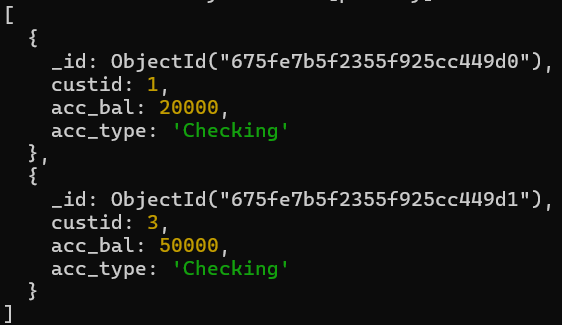
{custid: 1, acc\_bal:20000, acc\_type: "Checking"}, {custid: 3, acc\_bal:50000, acc\_type: "Checking"}, {custid: 4, acc\_bal:10000, acc\_type: "Saving"}, {custid: 5, acc\_bal:2000, acc\_type: "Checking"}]);

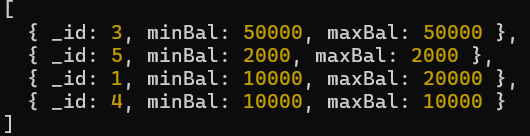


### Queries:

#### Write a query to display those records whose total account balance is greater than 1200 of account type ‘Z’ for each customer\_id.

db.Customer.find({acc\_bal: {$gt: 12000}, acc\_type:"Checking"});



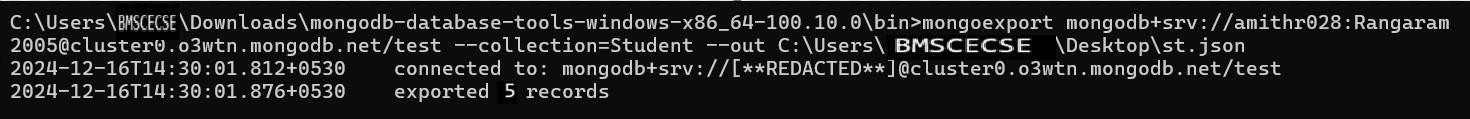
**Determine Minimum and Maximum account balance for each customer\_id.** db.Customer.aggregate([{$group:{\_id:"$custid", minBal:{$min:"$acc\_bal"}, maxBal:{$max:"$acc\_bal"}}}]);

db.Customers.drop()



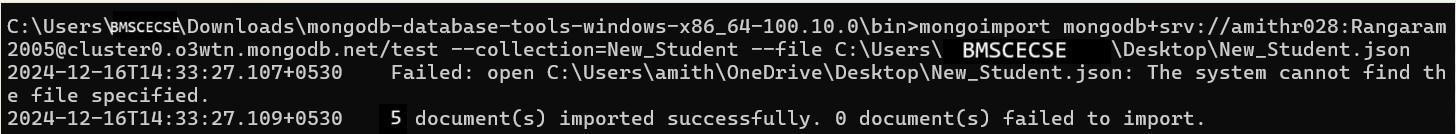
mongoexport mongodb+srv://dbms:@cluster0.xmdk9.mongodb.net/test

--collection=Student --out C:\Users\BMSCECSE\Desktop\st.json



mongoimport mongodb+srv://dbms:@cluster0.xmdk9.mongodb.net/test

--collection=New\_Student --file C:\Users\BMSCECSE\Desktop\New\_Student.json



# **NO SQL RESTAURANTS DATABASE**

**(WEEK-10)**

**QUESTION**

* Write a MongoDB query to display all the documents in the collection restaurants.
* Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns.
* Write a MongoDB query to find the restaurant Id, name, town and cuisine for those restaurants which achieved a score which is not more than 10.
* Write a MongoDB query to find the average score for each restaurant.
* Write a MongoDB query to find the name and address of the restaurants that have a zipcode that starts with '10'.

### Create Database:

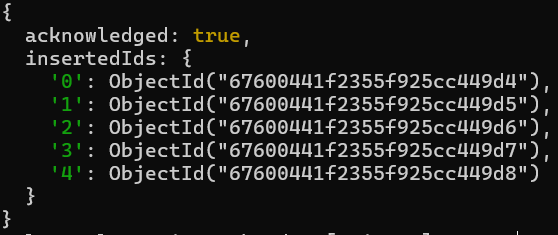
db.createCollection("restaurants");



### Inserting Values to the tables:

db.restaurants.insertMany([{ name: "Meghna Foods", town: "Jayanagar", cuisine: "Indian", score: 8, address: { zipcode: "10001", street: "Jayanagar"}},{ name: "Empire", town: "MG Road", cuisine: "Indian", score: 7, address: { zipcode: "10100", street: "MG Road"}},{ name: "Chinese WOK", town: "Indiranagar", cuisine: "Chinese", score: 12, address: { zipcode: "20000", street: "Indiranagar" }},{ name: "Kyotos", town: "Majestic", cuisine: "Japanese", score: 9, address: { zipcode: "10300", street: "Majestic"

}},{ name: "WOW Momos", town: "Malleshwaram", cuisine: "Indian", score: 5, address: { zipcode: "10400", street: "Malleshwaram"}}])



### Queries:

#### Write a MongoDB query to display all the documents in the collection restaurants.

db.restaurants.find({})



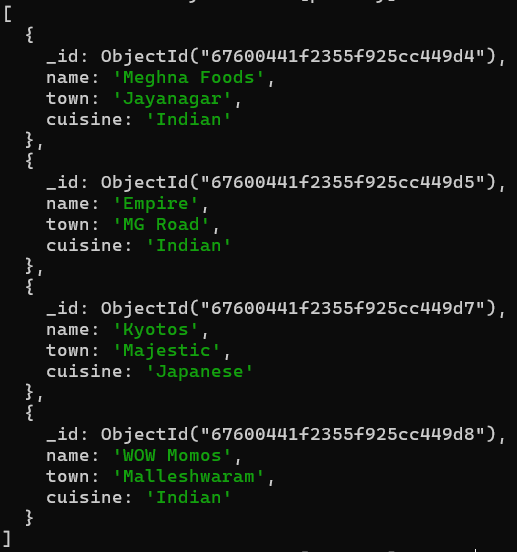
#### Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns

db.restaurants.find({}).sort({ name: -1 })



#### Write a MongoDB query to find the restaurant Id, name, town and cuisine for those restaurants which achieved a score which is not more than 10.

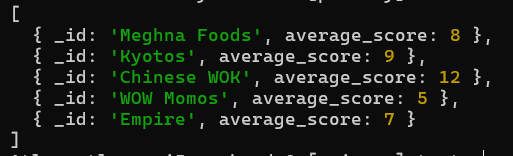
db.restaurants.find({ "score": { $lte: 10 } }, { \_id: 1, name: 1, town: 1, cuisine: 1 })



#### Write a MongoDB query to find the average score for each restaurant.

db.restaurants.aggregate([ { $group: { \_id: "$name", average\_score: {

$avg:"$score"}}}])



#### Write a MongoDB query to find the name and address of the restaurants that have a zipcode that starts with '10'.

db.restaurants.find({ "address.zipcode": /^10/ }, { name: 1, "address.street": 1, \_id: 0 })

