**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

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

****

**LAB REPORT**

**on**

**Database Management Systems (23CS3PCDBM)**

***Submitted by:***

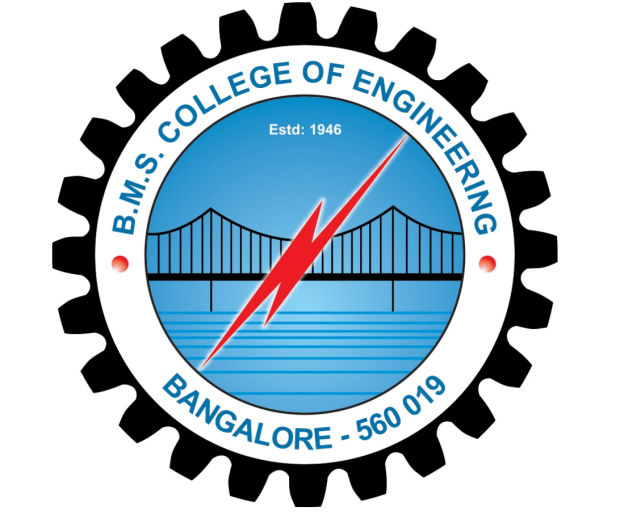
**Supreeth AR (2023BMS02528)**

***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**

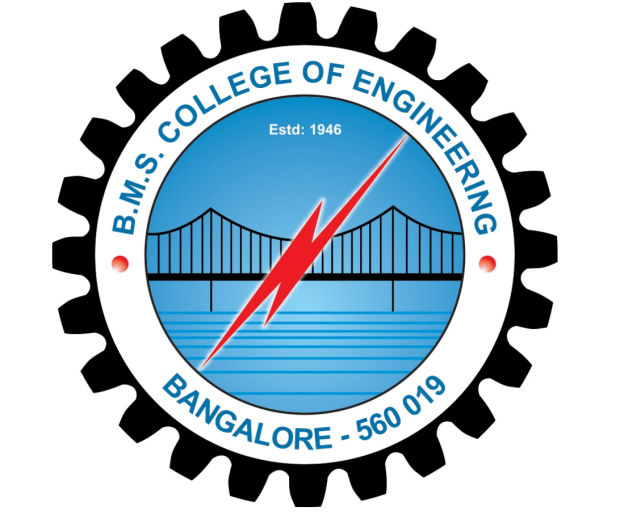
**December-2023 to March-2024**

**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 **Supreeth AR(2023BMS02528),**who is a 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 (22CS3PCDBM) work prescribed for the said degree.

**Prof. Rekha G S**                                                                    **Dr. Jyothi S Nayak**

Assistant Professor                                                          Professor and Head of The Department

Department of CSE,BMSCE,Bengaluru                      Department of CSE BMSCE,Bengaluru

`

**Index**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.**  **No.** | **Date** | **Experiment Title** | **Page No.** |
| 1 | 21-12-2023 | Insurance Database | 4-9 |
| 2 | 28-12-2023 | More Queries on Insurance Database | 10-15 |
| 3 | 11-11-2024 | Bank Database | 16-21 |
| 4 | 18-01-2024 | More Queries on Bank Database | 22-29 |
| 5 | 25-01-2024 | Employee Database | 30-35 |
| 6 | 25-01-2024 | More Queries on Employee Database | 36-41 |
| 7 | 01-02-2024 | Supplier Database | 42-46 |
| 8 | 29-02-2024 | NoSQL Lab 1 | 47-48 |
| 9 | 29-02-2024 | NoSQL Lab 2 | 49-50 |

**INSURANCE DATABASE**

WEEK 1

Consider the Insurance database given below. The data types are specified.

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)

LIST OF OPERATIONS

● Create the above tables by properly specifying the primary keys and the foreign

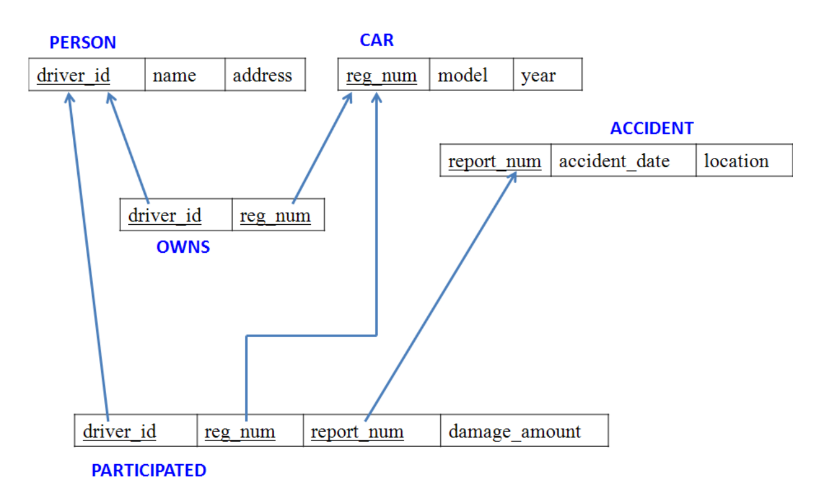
keys.

● Enter at least five tuples for each relation

● Display Accident date and location

● Display driver id who did the accident damage greater than or equal to Rs.25000

SCHEMA DIAGRAM



CREATE DATABASE

create database tanmay\_insurance;

Use tanmay\_insurance;

CREATE TABLES

create table person(

driver\_id varchar(20),

name varchar(30),

address varchar(50),

PRIMARY KEY(driver\_id));

create table car(

reg\_num varchar(15),

model varchar(10),

year int,

PRIMARY KEY(reg\_num));

create table owns(

driver\_id varchar(20),

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 accident(

report\_num int,

accident\_date date,

location varchar(50),

PRIMARY KEY(report\_num));

create table participated(

driver\_id varchar(20),

reg\_num varchar(10),

report\_num int,

damage\_amount int,

PRIMARY KEY(driver\_id,reg\_num,report\_num),

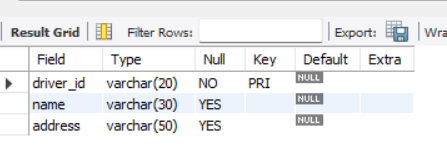
FOREIGN KEY(driver\_id) REFERENCES person(driver\_id),

FOREIGN KEY(reg\_num) REFERENCES car(reg\_num),

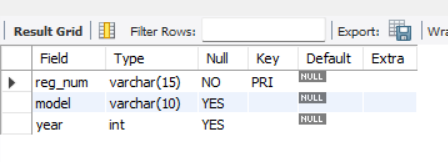
FOREIGN KEY(report\_num) REFERENCES accident(report\_num));

STRUCTURE OF THE TABLES

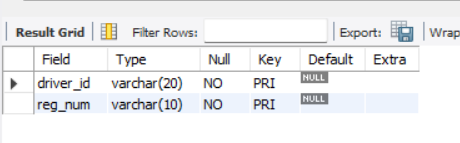
desc person;



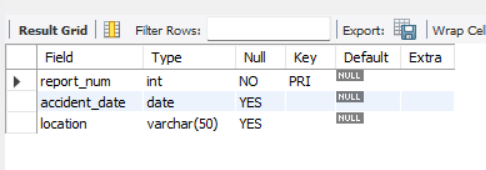
desc car;



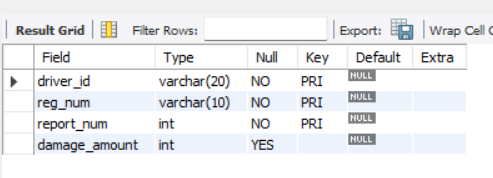
desc owns;



desc accident;



desc participated;



INSERTING VALUES TO THE TABLES

insert into person values

(‘A01’,’Richard’, ‘Srinivas nagar’),

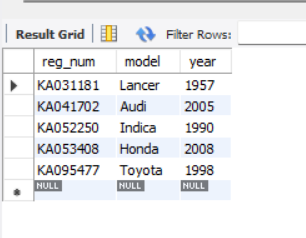
(‘A02’,’Pradeep’, ‘Rajaji Nagar’),

(‘A03’,’Smith’, ‘Ashok Nagar’),

(‘A04’,’Venu’, ‘N R Colony’),

(‘A05’,’John’, ‘Hanumanth Nagar’);

select \* from person;



insert into car values

(“KA052250”,”Indica”,1990),

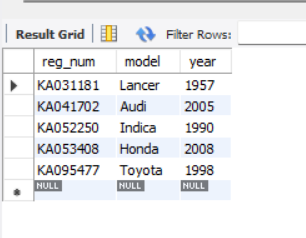
(“KA031181”,”Lancer”,1957),

(“KA095477”,”Toyota”,1998),

(“KA053408”,”Honda”,2008),

(“KA041702”,”Audi”,2005);

select \* from car;



insert into owns values

(“A01”,”KA052250”),

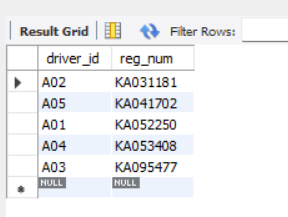
(“A02”,”KA031181”),

(“A03”,”KA095477”),

(“A04”,”KA053408”),

(“A05”,”KA041702”);

select \* from owns;



insert into accident values

(11,’2003-01-01’,”Mysore Road”),

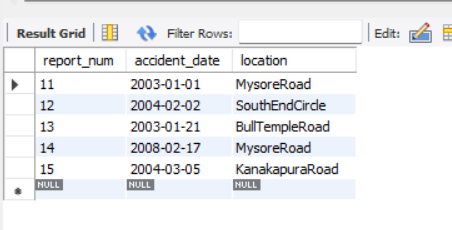
(12,’2004-02-02’,”South end Circle”),

(13,’2003-01-21’,”Bull temple Road”),

(14,’2008-02-17’,”Mysore Road”),

(15,’2004-03-05’,”Kanakapura Road”);

select \* from accident;



insert into participated values

(“A01”,”KA052250”,11,10000);

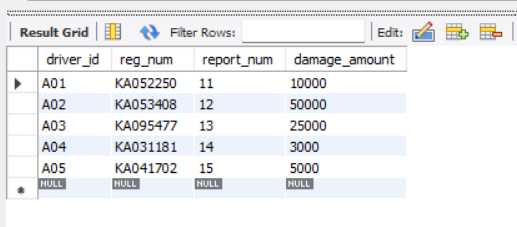
(“A02”,”KA053408”,12,50000);

(“A03”,”KA095477”,13,25000);

(“A04”,”KA031181”,14,3000);

(“A05”,”KA041702”,15,5000);

select \* from participated;



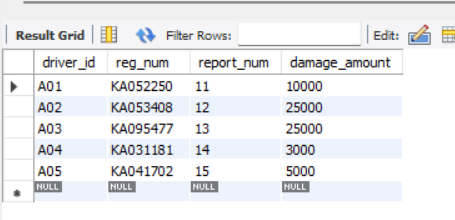
QUERIES

* 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

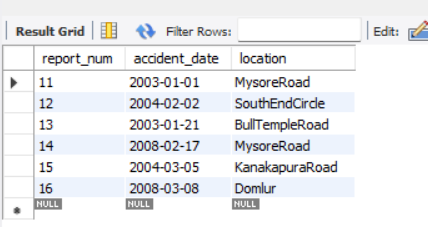
where reg\_num=”KA053408” and report\_num=12;



* Add a new accident to the database.

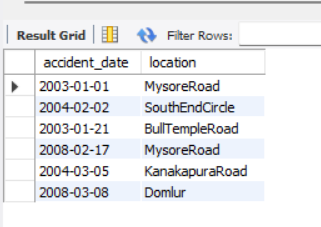
insert into accident values(16,”2008-03-08”,”Domlur”);

select \* from accident;



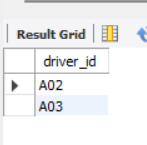
* Display accident date and location.

select accident\_date, location from accident;



* Display Driver ID,whose accident with damage amount greater than or equal to 25000.

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



**MORE QUERIES ON INSURANCE DATABASE**

(WEEK 2)

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)

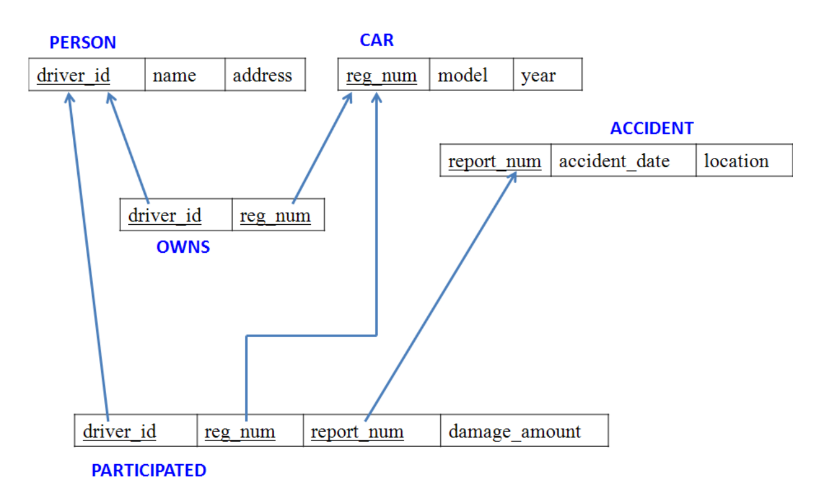
-Display the entire CAR relation in the ascending order of manufacturing year.

- Find the number of accidents in which cars belonging to a specific model,

 (example,”Lancer”) were involved.

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

SCHEMA DIAGRAM



CREATE DATABASE

create database tanmay\_insurance;

Use tanmay\_insurance;

CREATE TABLES

create table person(

driver\_id varchar(20),

name varchar(30),

address varchar(50),

PRIMARY KEY(driver\_id));

create table car(

reg\_num varchar(15),

model varchar(10),

year int,

PRIMARY KEY(reg\_num));

create table owns(

driver\_id varchar(20),

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 accident(

report\_num int,

accident\_date date,

location varchar(50),

PRIMARY KEY(report\_num));

create table participated(

driver\_id varchar(20),

reg\_num varchar(10),

report\_num int,

damage\_amount int,

PRIMARY KEY(driver\_id,reg\_num,report\_num),

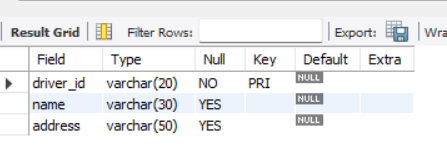
FOREIGN KEY(driver\_id) REFERENCES person(driver\_id),

FOREIGN KEY(reg\_num) REFERENCES car(reg\_num),

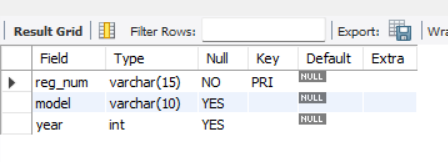
FOREIGN KEY(report\_num) REFERENCES accident(report\_num));

STRUCTURE OF THE TABLES

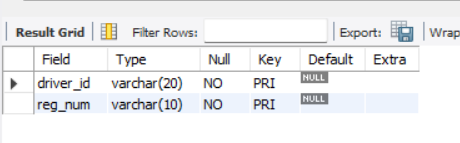
desc person;



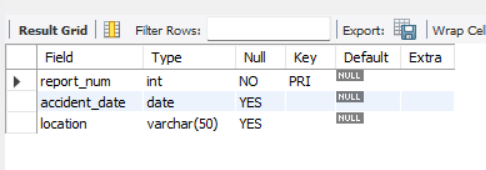
desc car;



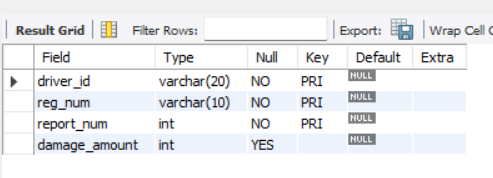
desc owns;



desc accident;



desc participated;



INSERTING VALUES TO THE TABLES

insert into person values

(‘A01’,’Richard’, ‘Srinivas nagar’),

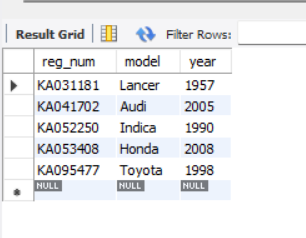
(‘A02’,’Pradeep’, ‘Rajaji Nagar’),

(‘A03’,’Smith’, ‘Ashok Nagar’),

(‘A04’,’Venu’, ‘N R Colony’),

(‘A05’,’John’, ‘Hanumanth Nagar’);

select \* from person;



insert into car values

(“KA052250”,”Indica”,1990),

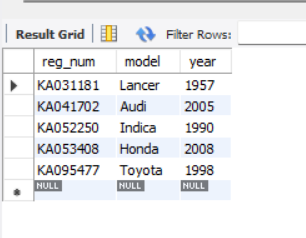
(“KA031181”,”Lancer”,1957),

(“KA095477”,”Toyota”,1998),

(“KA053408”,”Honda”,2008),

(“KA041702”,”Audi”,2005);

select \* from car;



insert into owns values

(“A01”,”KA052250”),

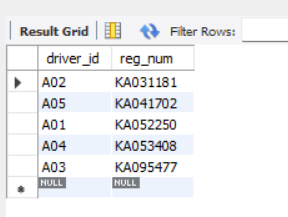
(“A02”,”KA031181”),

(“A03”,”KA095477”),

(“A04”,”KA053408”),

(“A05”,”KA041702”);

select \* from owns;



insert into accident values

(11,’2003-01-01’,”Mysore Road”),

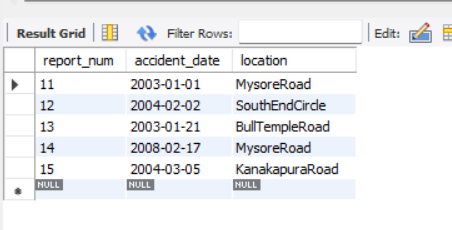
(12,’2004-02-02’,”South end Circle”),

(13,’2003-01-21’,”Bull temple Road”),

(14,’2008-02-17’,”Mysore Road”),

(15,’2004-03-05’,”Kanakapura Road”);

select \* from accident;



insert into participated values

(“A01”,”KA052250”,11,10000);

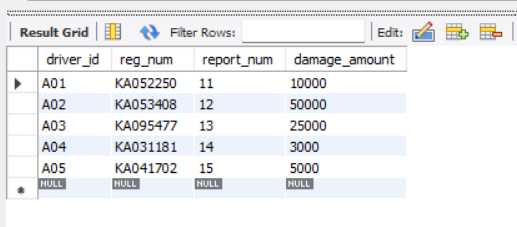
(“A02”,”KA053408”,12,50000);

(“A03”,”KA095477”,13,25000);

(“A04”,”KA031181”,14,3000);

(“A05”,”KA041702”,15,5000);

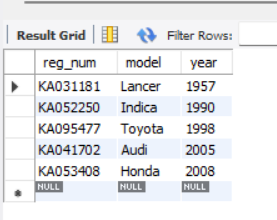
select \* from participated;



QUERIES

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

select \* from car order by year asc;

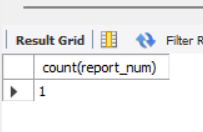


● Find the number of accidents in which cars belonging to a specific model (example,”Lancer”) were involved.

select count(report\_num)

from car c, participated p

where c.reg\_num=p.reg\_num and c.model=”Lancer”;

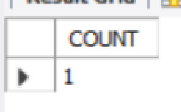


Find the total number of people who owned cars that were involved in accidents in 2008

select count(distinct driver\_id)

from participated a, accident b

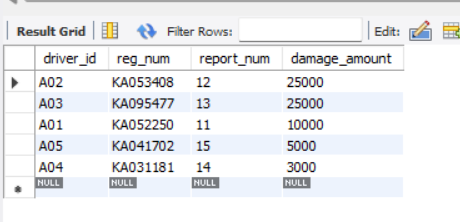
where a.report\_num=b.report\_num and b.accident\_date like “\_ \_08”;



TO DO:

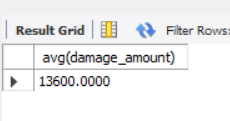
● List the entire participated relation in the descending order of the damage amount,

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



● Find the average damage amount.

select avg(damage\_amount) from participated;

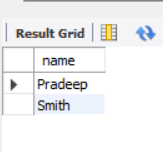


● List the name of drivers whose damage is greater than the average damage amount.

select name from person p, participated part

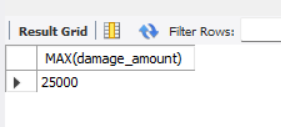
where p.driver\_id=part.driver\_id and

damage\_amount>(select AVG(damage\_amount) FROM participated)



● Find maximum damage amount.

   select MAX(damage\_amount) from participated;



**BANK DATABASE**

(Week 3)

Question

- 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 &#39;assets in lakhs&#39;.

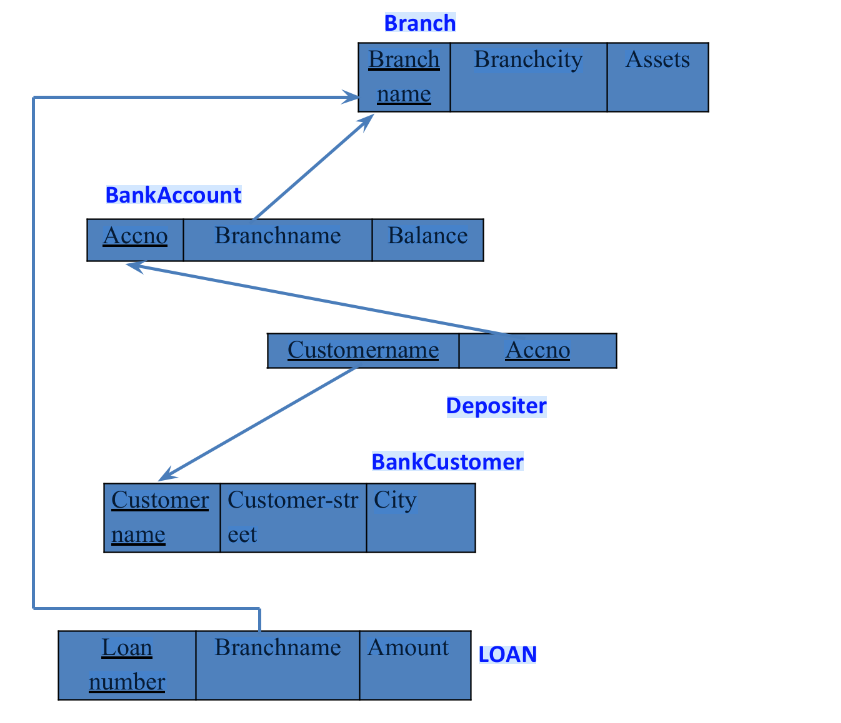
- Find all the customers who have at least two accounts at the same branch (ex.

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 tanmay\_bank;

Use tanmay\_bank;

CREATE TABLES

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),

CustomerCity 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),

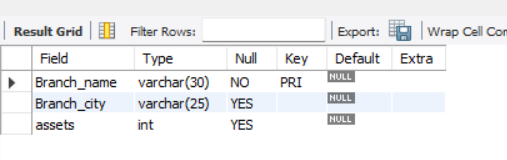
Amount int,

PRIMARY KEY(Loan\_number),

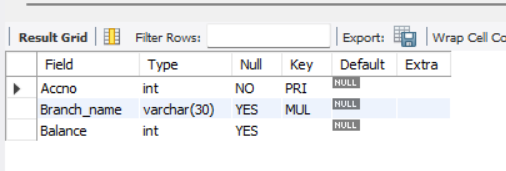
foreign key (Branch\_name) references branch(Branch\_name));

STRUCTURE OF THE TABLES

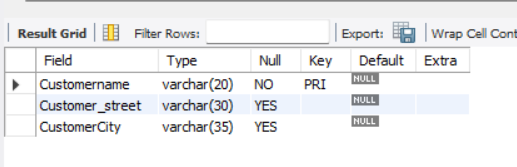
desc Branch;



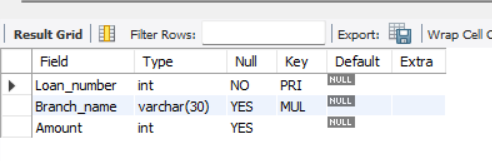
desc BankAccount;



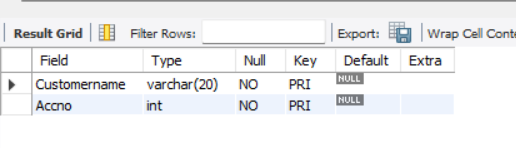
desc BankCustomer;



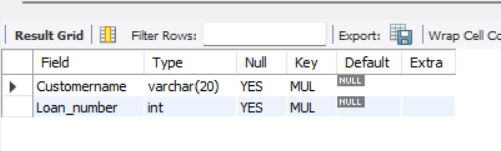
desc Loan;



desc Depositor;



desc Borrower;



INSERTING VALUES TO THE TABLES

insert into branch values

(“SBI\_Chamrajpet”,”Bangalore”,50000),

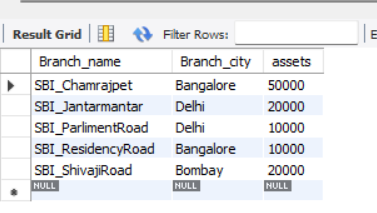
(“SBI\_ResidencyRoad”,”Bangalore”,10000),

(“SBI\_ShivajiRoad”,”Bombay”,20000),

(“SBI\_ParlimentRoad”,”Delhi”,10000),

(“SBI\_Jantarmantar”,”Delhi”,20000);

select \* from branch;



insert into BankAccount values

(1001,”SBI\_Chamrajpet”,2000),

(1002,”SBI\_ResidencyRoad”,5000),

(1003,”SBI\_ShivajiRoad”,6000),

(1004,”SBI\_ParlimentRoad”,9000),

(1005,”SBI\_Jantarmantar”,8000),

(1006,”SBI\_ShivajiRoad”,4000),

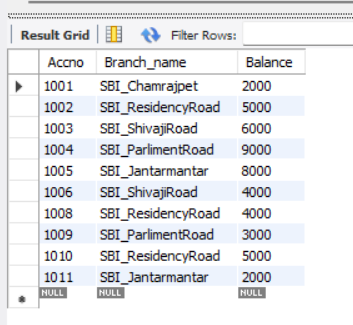
(1008,”SBI\_ResidencyRoad”,4000),

(1009,”SBI\_ParlimentRoad”,3000),

(1010,”SBI\_ResidencyRoad”,5000),

(1011,”SBI\_Jantarmantar”,2000);

select \* from BankAccount;



insert into BankCustomer values

(“Avinash”,”Bull\_Temple\_Road”,”Bangalore”),

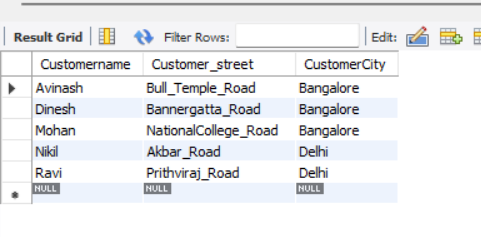
(“Dinesh”,”Bannergatta\_Road”,”Bangalore”),

(“Mohan”,”NationalCollege\_Road”,”Bangalore”),

(“Nikil”,”Akbar\_Road”,”Delhi”),

(“Ravi”,”Prithviraj\_Road”,”Delhi”);

select \* from BankCustomer;



insert into Depositor values

(“Avinash”,1001),

(“Dinesh”,1002),

(“Nikil”,1004),

(“Ravi”,1005),

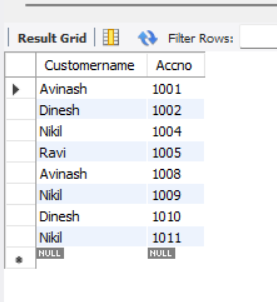
(“Avinash”,1008),

(“Nikil”,1009),

(“Dinesh”,1010),

(“Nikil”,1011);

select \* from Depositor;



insert into Loan values

(1,”SBI\_Chamrajpet”,1000),

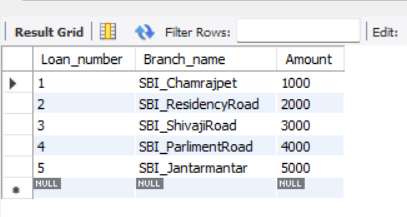
(2,”SBI\_ResidencyRoad”,2000),

(3,”SBI\_ShivajiRoad”,3000),

(4,”SBI\_ParlimentRoad”,4000),

(5,”SBI\_Jantarmantar”,5000);

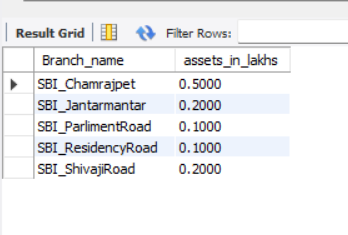
select \* from Loan;



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; 

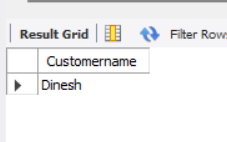
● Find all the customers who have at least two accounts at the same branch (ex.SBI\_ResidencyRoad).

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;



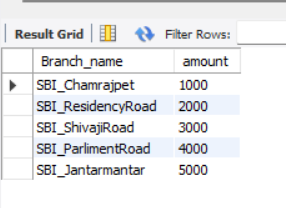
● Create a view which gives each branch the sum of the amount of all the loans at the branch.

create view sum\_of\_loan

as select Branch\_name, amount

from Loan;

select \* from sum\_of\_loan;



**MORE QUERIES ON THE  BANK DATABASE**

Question

(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)

- Depositor(customer-name: String, accno: int)

- Loan (loan-number: int, branch-name: String, amount: real)

- Borrower (customer-name: String, loan-number: int)

- 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

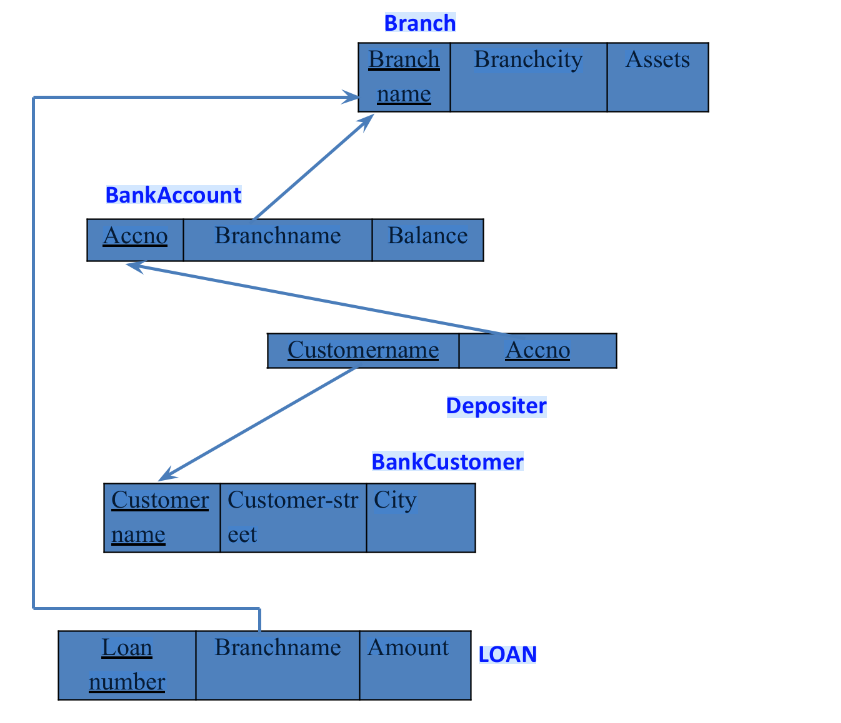
Bangalore.

- 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%

SCHEMA DIAGRAM:



CREATE DATABASE

create database tanmay\_bank;

Use tanmay\_bank;

CREATE TABLES

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),

CustomerCity 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(

Customername varchar(20),

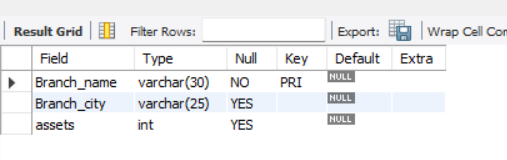
Loan\_number int,

foreign key(Customername) references BankCustomer(Customername),

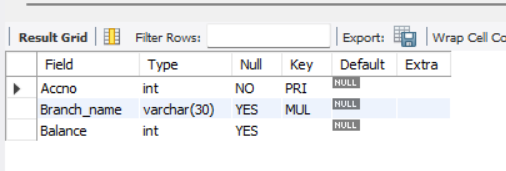
foreign key(Loan\_number) references Loan(Loan\_number));

STRUCTURE OF THE TABLES

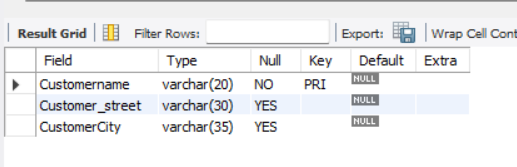
desc Branch;



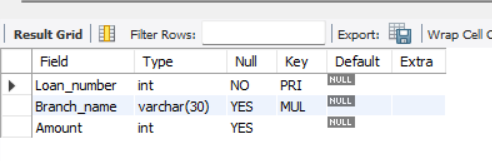
desc BankAccount;



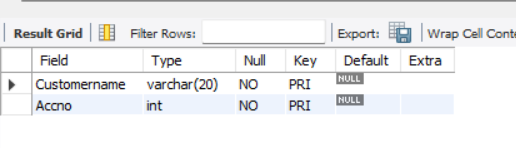
desc BankCustomer;



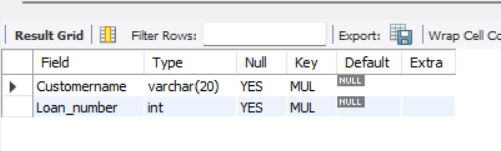
desc Loan;



desc Depositor;



desc Borrower;



INSERTING VALUES TO THE TABLES

insert into branch values

(“SBI\_Chamrajpet”,”Bangalore”,50000),

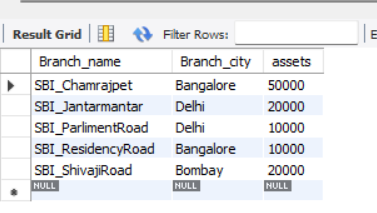
(“SBI\_ResidencyRoad”,”Bangalore”,10000),

(“SBI\_ShivajiRoad”,”Bombay”,20000),

(“SBI\_ParlimentRoad”,”Delhi”,10000),

(“SBI\_Jantarmantar”,”Delhi”,20000);

select \* from branch;



insert into BankAccount values

(1001,”SBI\_Chamrajpet”,2000),

(1002,”SBI\_ResidencyRoad”,5000),

(1003,”SBI\_ShivajiRoad”,6000),

(1004,”SBI\_ParlimentRoad”,9000),

(1005,”SBI\_Jantarmantar”,8000),

(1006,”SBI\_ShivajiRoad”,4000),

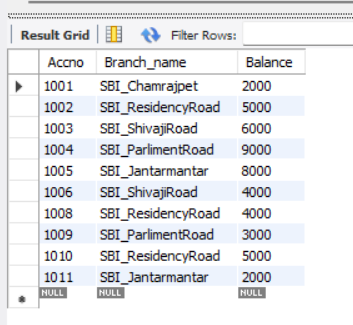
(1008,”SBI\_ResidencyRoad”,4000),

(1009,”SBI\_ParlimentRoad”,3000),

(1010,”SBI\_ResidencyRoad”,5000),

(1011,”SBI\_Jantarmantar”,2000);

select \* from BankAccount;



insert into BankCustomer values

(“Avinash”,”Bull\_Temple\_Road”,”Bangalore”),

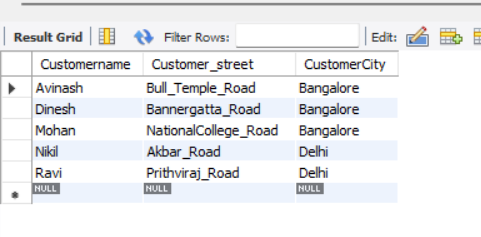
(“Dinesh”,”Bannergatta\_Road”,”Bangalore”),

(“Mohan”,”NationalCollege\_Road”,”Bangalore”),

(“Nikil”,”Akbar\_Road”,”Delhi”),

(“Ravi”,”Prithviraj\_Road”,”Delhi”);

select \* from BankCustomer;



insert into Depositer values

(“Avinash”,1001),

(“Dinesh”,1002),

(“Nikil”,1004),

(“Ravi”,1005),

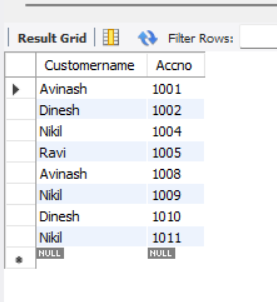
(“Avinash”,1008),

(“Nikil”,1009),

(“Dinesh”,1010),

(“Nikil”,1011);

select \* from Depositor;



insert into Loan values

(1,”SBI\_Chamrajpet”,1000),

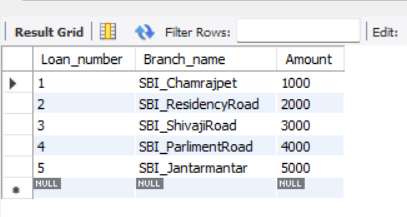
(2,”SBI\_ResidencyRoad”,2000),

(3,”SBI\_ShivajiRoad”,3000),

(4,”SBI\_ParlimentRoad”,4000),

(5,”SBI\_Jantarmantar”,5000);

select \* from Loan;



insert into Borrower values

(“Avinash”,1),

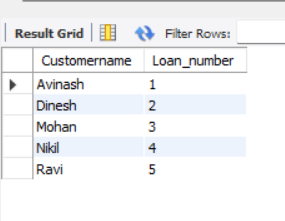
(“Dinesh”,2),

(“Mohan”,3),

(“Nikil”,4),

(“Ravi”,5);

select \* from Borrower ;



QUERIES

● Find all the customers who have an account at all the branches located in a specific

city (Ex. Delhi).

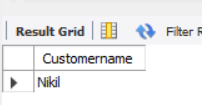
select d.Customername from branch b, Depositer d, BankAccount ba where

b.Branch\_city=”Delhi” and d.Accno=ba.Accno and b.Branch\_name=ba.Branch\_name

group by d.Customername

having count(distinct b.Branch\_name)= (select

count(distinct b.Branch\_name) from branch b where b.Branch\_city=”Delhi”;



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

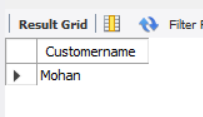
select distinct b.Customername from Borrower b, Depositer d

where b.Customername NOT IN(

select d.Customername from Loan l,Depositer d, Borrower b

where l.Loan\_number=b.Loan\_number and

d.Customername=b.Customername);



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

select distinct d.Customername from Depositer d

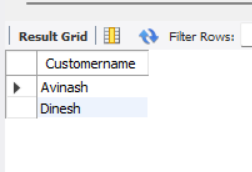
where d.Customername IN(

select d.Customername from branch br,Depositer d, BankAccount ba

where br.Branch\_city=”Bangalore” and

br.Branch\_name=ba.Branch\_name and ba.accno=d.accno and

Customername IN(select Customername from Borrower));



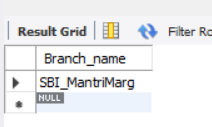
● Find the names of all branches that have greater assets than all branches located in

Bangalore.

select b.Branch\_name from Branch b

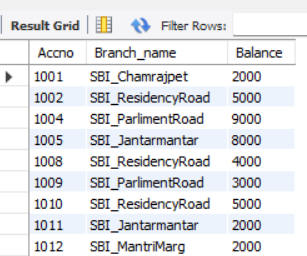
where b.assets>ALL (select SUM(b.assets) from Branch b

where b.Branch\_City=”Bangalore”);



● Update the Balance of all accounts by 5%

UPDATE BankAccount set Balance=(Balance + (Balance\*0.05));



● Demonstrate how you delete all account tuples at every branch located in a specific

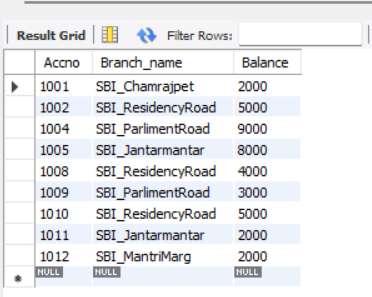
city (Ex. Bombay).

delete ba.\* from BankAccount ba, branch b

where branch\_city=”Bombay” and

ba.Branch\_name=b.Branch\_name;

select \* from BankAccount;

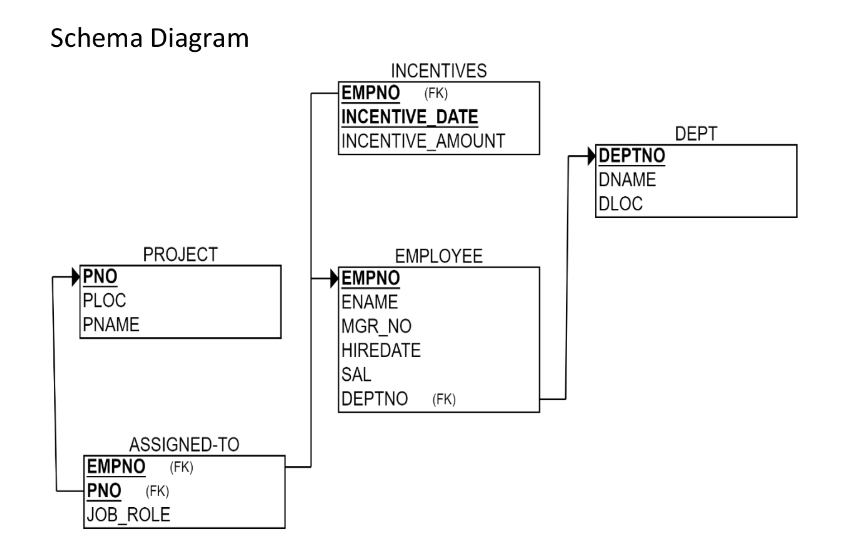


**EMPLOYEE DATABASE**

Question

(Week 5)

SCHEMA DIAGRAM



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 tanmay\_employee;

use tanmay\_employee;

CREATE TABLES

create table project(

pno int,

ploc varchar(40),

pname varchar(40),

PRIMARY KEY(pno));

create table dept(

deptno int,

dname varchar(40),

dloc varchar(40),

PRIMARY KEY(deptno));

create table employee(

empno int,

ename varchar(40),

mgr\_no int,

hiredate date,

sal int,

deptno int,

primary key (empno),

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

create table incentives(

empno int,

incentive\_date date,

incentive\_amount int,

primary key(incentive\_date),

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

create table assigned\_to(

empno int,

pno int,

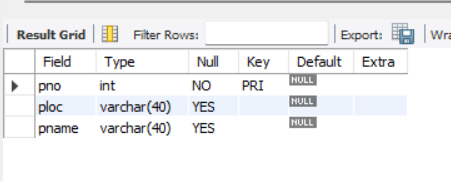
job\_role varchar(50),

foreign key (pno) references project(pno),

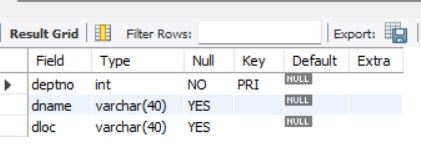
foreign key (empno) references employee(empno));

STRUCTURE OF THE TABLES

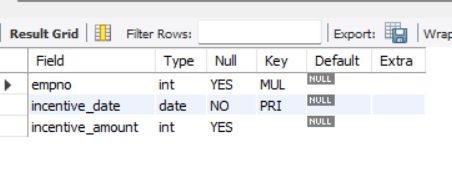
desc project;



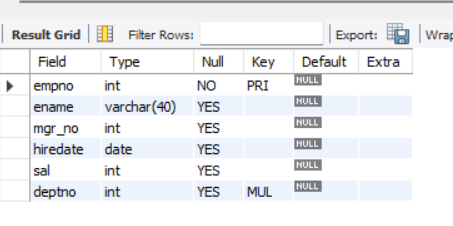
desc dept;



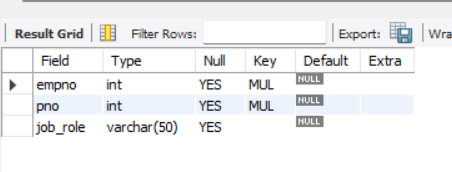
desc incentives;



desc employee;



desc assigned\_to;



INSERTING VALUES TO THE TABLES

insert into project values

(1,”Bengaluru”,”Syntax”),

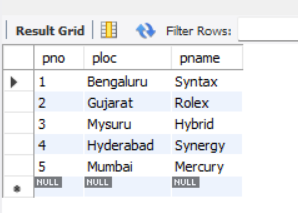
(2,”Gujarat”,”Rolex”),

(3,”Mysuru”,”Hybrid”),

(4,”Hyderabad”,”Synergy”),

(5,”Mumbai”,”Mercury”);

select \* from project;



insert into dept values

(10,”Sales”,”Bengaluru”),

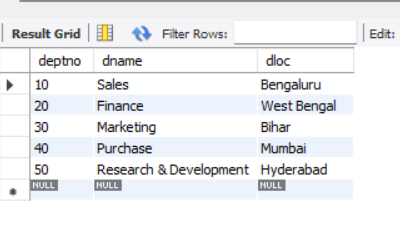
(20,”Finance”,”West Bengal”),

(30,”Marketing”,”Bihar’),

(40,”Purchase”,”Mumbai”),

(50,”Research & Development”,”Hyderabad’);

select \* from dept;



insert into employee values

(100,”Prannay”,400,”2003-01-01”,100000,10),

(200,”Farhaan”,500,”2004-02-02”,100500,50),

(300,”Sanika”,100,”2003-01-21”,200500,30),

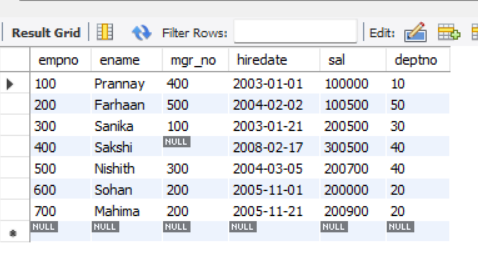
(400,”Sakshi”, NULL ,”2008-02-17”,300500,40),

(500,”Nishith”,300,”2004-03-05”,200700,40),

(600,”Sohan”,200,”2005-11-01”,200000,20),

(700,”Mahima”,200,”2005-11-21”,200900,20);

select \* from employee;



insert into incentives values

(100,”2012-02-17”,6000),

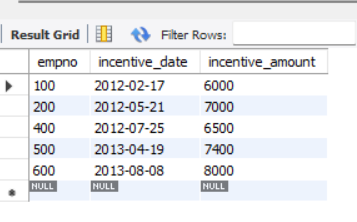
(200,”2012-05-21”,7000),

(400,”2012-07-25”,6500),

(500,”2013-04-19”,7400),

(600,”2013-08-08”,8000);

select \* from incentives;



insert into assigned\_to values

(100,1,”Project Manager”),

(200,1,”Resource Manager”),

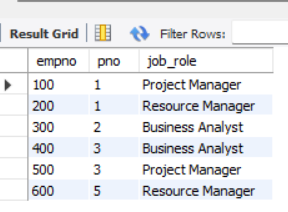
(300,2,”Business Analyst”),

(400,3,”Business Analyst”),

(500,3,”Project Manager”),

(600,5,”Resource Manager”);

select \* from assigned\_to;

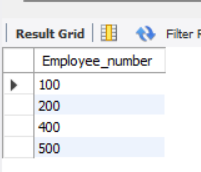


QUERIES

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

select a.empno Employee\_number from project p, assigned\_to a

where p.pno=a.pno and p.ploc in(“Hyderabad”,”Bengaluru”,”Mysuru”);

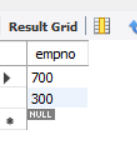


● Get Employee ID’s of those employees who didn’t receive incentives.

select e.empno from employee e

where e.empno NOT IN

(select i.empno from incentives i);



● 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

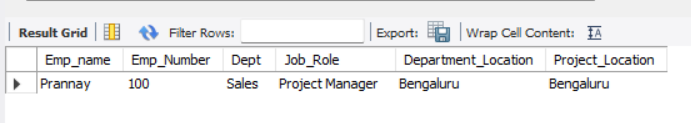
department location.

select e.ename Emp\_name, e.empno Emp\_Number, d.dname Dept, a.job\_role Job\_Role, d.dloc

Department\_Location, p.ploc Project\_Location

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

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



**MORE QUERIES ON THE EMPLOYEE DATABASE**

Question

(Week 6)

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. List the name of the managers with the maximum employees

4. Display the manager's name whose salary is more than the average salary of his employee.

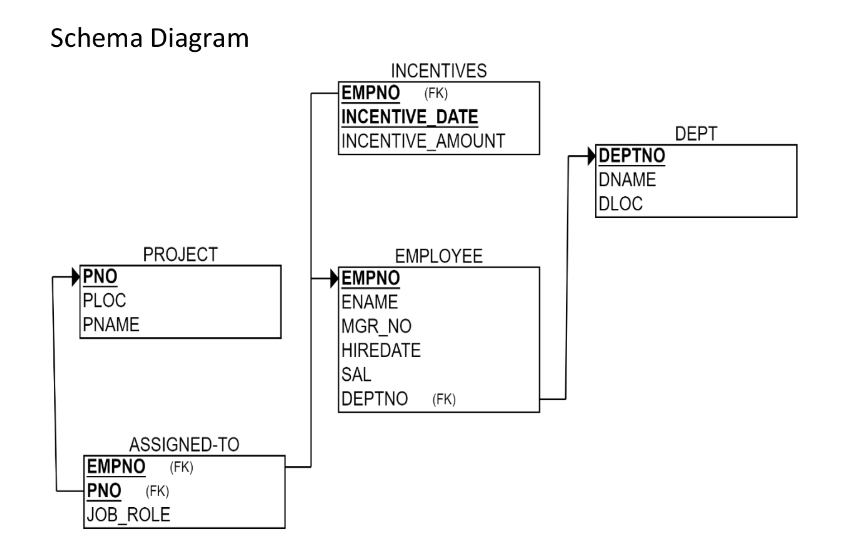
5. Find the name of the second top level managers of each department.

6. Find the employee details who got the second maximum incentive in January 2019.

7. Display those employees who are working in the same department where his the

     manager is working.

SCHEMA DIAGRAM



CREATE DATABASE

create database tanmay\_employee;

use tanmay\_employee;

CREATE TABLES

create table project(

pno int,

ploc varchar(40),

pname varchar(40),

PRIMARY KEY(pno));

create table dept(

deptno int,

dname varchar(40),

dloc varchar(40),

PRIMARY KEY(deptno));

create table employee(

empno int,

ename varchar(40),

mgr\_no int,

hiredate date,

sal int,

deptno int,

primary key (empno),

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

create table incentives(

empno int,

incentive\_date date,

incentive\_amount int,

primary key(incentive\_date),

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

create table assigned\_to(

empno int,

pno int,

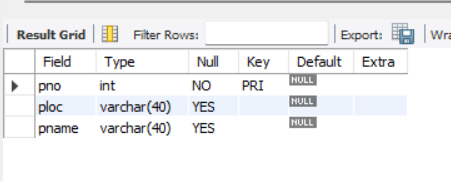
job\_role varchar(50),

foreign key (pno) references project(pno),

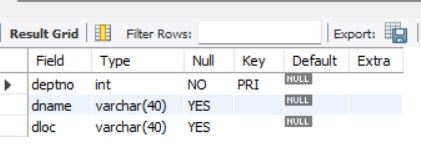
foreign key (empno) references employee(empno));

STRUCTURE OF THE TABLES

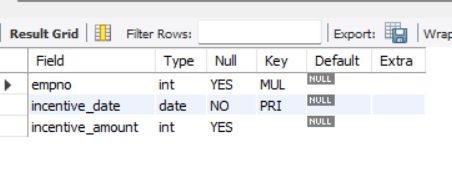
desc project;



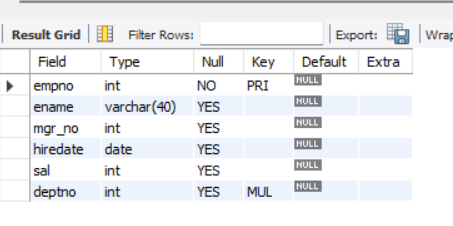
desc dept;



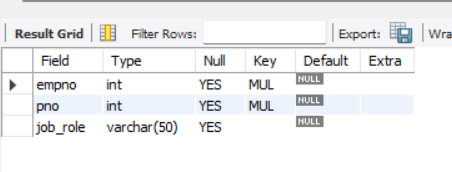
desc incentives;



desc employee;



desc assigned\_to;



INSERTING VALUES TO THE TABLES

insert into project values

(1,”Bengaluru”,”Syntax”),

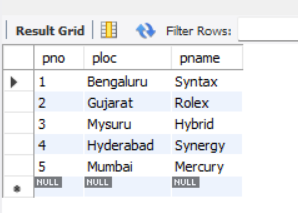
(2,”Gujarat”,”Rolex”),

(3,”Mysuru”,”Hybrid”),

(4,”Hyderabad”,”Synergy”),

(5,”Mumbai”,”Mercury”);

select \* from project;



insert into dept values

(10,”Sales”,”Bengaluru”),

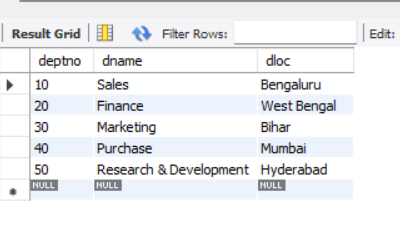
(20,”Finance”,”West Bengal”),

(30,”Marketing”,”Bihar’),

(40,”Purchase”,”Mumbai”),

(50,”Research & Development”,”Hyderabad’);

select \* from dept;



insert into employee values

(100,”Prannay”,400,”2003-01-01”,100000,10),

(200,”Farhaan”,500,”2004-02-02”,100500,50),

(300,”Sanika”,100,”2003-01-21”,200500,30),

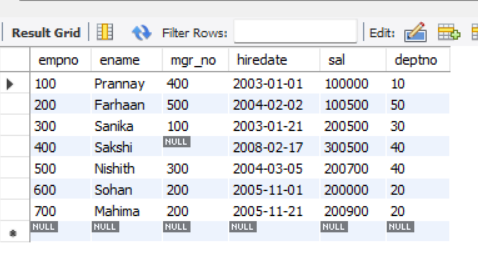
(400,”Sakshi”, NULL ,”2008-02-17”,300500,40),

(500,”Nishith”,300,”2004-03-05”,200700,40),

(600,”Sohan”,200,”2005-11-01”,200000,20),

(700,”Mahima”,200,”2005-11-21”,200900,20);

select \* from employee;



insert into incentives values

(100,”2012-02-17”,6000),

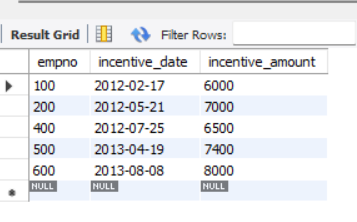
(200,”2012-05-21”,7000),

(400,”2012-07-25”,6500),

(500,”2013-04-19”,7400),

(600,”2013-08-08”,8000);

select \* from incentives;



insert into assigned\_to values

(100,1,”Project Manager”),

(200,1,”Resource Manager”),

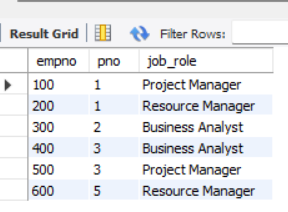
(300,2,”Business Analyst”),

(400,3,”Business Analyst”),

(500,3,”Project Manager”),

(600,5,”Resource Manager”);

select \* from assigned\_to;



QUERIES

● List the name of the managers with the maximum employees

select e1.ename

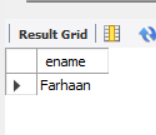
from employee e1, employee e2

where e1.empno=e2.mgr\_no group by e1.ename

having count(e1.mgr\_no)=(select count(e1.ename)

from employee e1, employee e2 where e1.empno=e2.mgr\_no

group by e1.ename order by count(e1.ename) desc limit 1);



● Display those managers name whose salary is more than average salary of his

employee

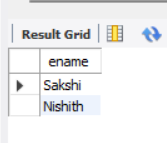
select m.ename from employee m

where m.empno in

(select mgr\_no from employee)

and m.sal > (select avg(n.sal) from employee n

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



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

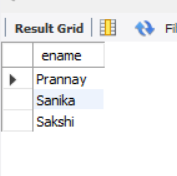
select ename from employee where empno in

(select distinct mgr\_no

from employee where empno in

(select distinct mgr\_no from employee where empno in

(select distinct mgr\_no from employee)));



● Find the employee details who got the second maximum incentive in January 2019.

select \* from employee where empno=

(select i.empno from incentives i

where i.incentive\_amount= (select max(n.incentive\_amount) from incentives n

where n.incentive\_amount < (select max(inc.incentive\_amount) from incentives inc

where inc.incentive\_date between ‘2019-01-01’ and ‘2019-12-31’) and incentive\_date

between ‘2019-01-01’ and ‘2019-12-31’)));

● Display those employees who are working in the same department where his manager is

Working.

select e2.ename

from employee e1, employee e2

where e1.empno=e2.mgr\_no and e1.deptno=e2.deptno;

**SUPPLIER DATABASE**

Question

(Week 7)

1. Using the Schema diagram, create tables by properly specifying the primary keys and the foreign keys.

2. Insert appropriate records in each table.

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

4. Find the names of suppliers who supply every part.

5. Find the names of suppliers who supply every red part.

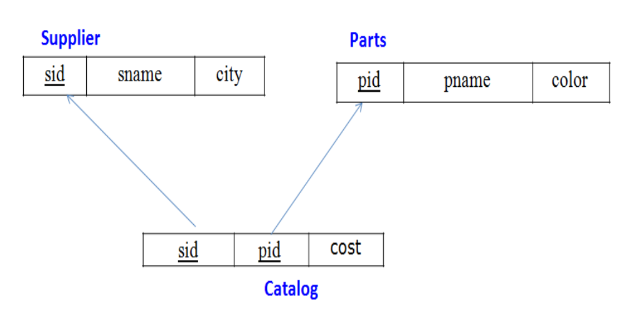
6. Find the names of parts supplied by Acme Widget Suppliers and by no one else.

7. 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).

8. For each part, find the name of the supplier who charges the most for that part.

SCHEMA DIAGRAM



CREATE DATABASE

create database tanmay\_supplier;

use tanmay\_supplier;

CREATE TABLES

create table Supplier(

sid int,

sname varchar(15),

city varchar(10),

PRIMARY KEY(sid));

create table Parts(

pid int,

pname varchar(10),

color varchar(5),

PRIMARY KEY(pid));

create table Catalog(

sid int,

pid int,

cost int,

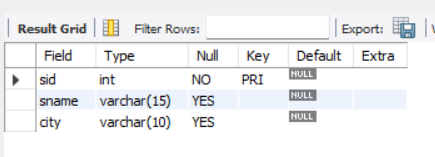
PRIMARY KEY(sid, pid),

FOREIGN KEY(sid) REFERENCES Supplier(sid),

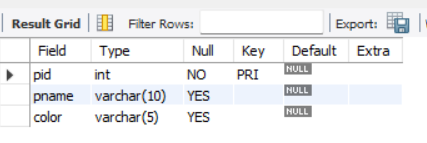
FOREIGN KEY(pid) REFERENCES Parts(pid));

STRUCTURE OF THE TABLES

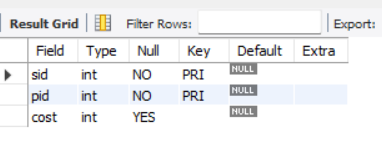
desc Supplier;



desc Parts;



desc Catalog;



INSERTING VALUES TO THE TABLES

insert into Supplier values

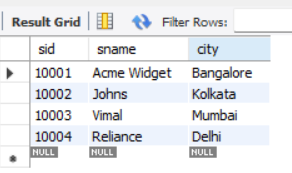
(10001,”Acme Widget”, “Bangalore”),

(10002,”Johns”, “Kolkata”),

(10003,”Vimal”, “Mumbai”),

(10004,”Reliance”, “Delhi”);

select \* from Supplier;



insert into Parts values

(20001,”Book”,”Red”),

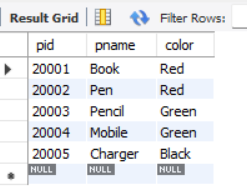
(20002,”Pen”,”Red”),

(20003,”Pencil”,”Green”),

(20004,”Mobile”, “Green”),

(20005,”Charger”, “Black”);

select \* from Parts;



insert into Catalog values

(10001,20001, 10);

(10001,20002, 10);

(10001,20003, 30);

(10001,20004, 10);

(10001,20005, 10);

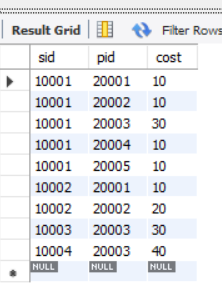
(10002,20001, 10);

(10002,20002, 20);

(10003,20003, 30);

(10004,20003, 40);

select \* from Catalog;



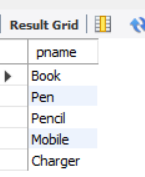
QUERIES

● Find the names of parts for which there is some supplier.

select distinct p.pname

from Parts p, Catalog c

where p.pid = c.pid;



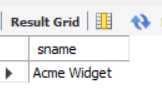
● Find the names of suppliers who supply every part.

select distinct s.sname

from Catalog C, Supplier s WHERE C.sid=s.sid and NOT EXISTS (select P.pid FROM Parts P

where NOT EXISTS (select C1.sid from Catalog C1

where C1.sid = C.sid and C1.pid = P.pid));



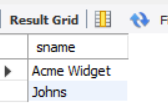
● Find the names of suppliers who supply every red part.

select distinct s.sname

from Catalog C, Supplier s where C.sid=s.sid and NOT EXISTS (select P.pid from Parts P

where P.color=”Red” and NOT EXISTS (select C1.sid from Catalog C1

where C1.sid = C.sid and C1.pid = P.pid and P.color=”Red”));



● Find the names of parts supplied by Acme Widget Suppliers and by no one else.

select P.pname

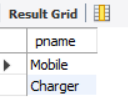
from Parts P, Catalog C, Supplier S

where P.pid = C.pid and C.sid = S.sid and S.sname = “Acme Widget”

and NOT EXISTS (select \* from Catalog C1, Supplier S1

where P.pid = C1.pid and C1.sid = S1.sid and

S1.sname != “Acme Widget”);



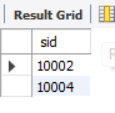
● Find the ids of suppliers who charge more for some part than the average cost of that part.

(averaged over all the suppliers who supply that part).

select distinct C.sid from Catalog C

where C.cost > (select AVG(C1.cost)

from Catalog C1 where C1.pid = C.pid);



● For each part, find the name of the supplier who charges the most for that part.

select P.pid, S.sname

from Parts P, Supplier S, Catalog C

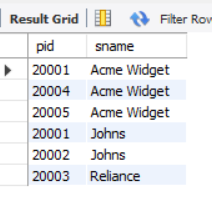
where C.pid = P.pid and

C.sid = S.sid and

C.cost = (select max(C1.cost)

from Catalog C1

where C1.pid = P.pid);



**NoSQL Lab 1**

Question

(Week 8)

Perform the following DB operations using MongoDB.

1. Create a database “Student” with the following attributes Rollno, Age, ContactNo, Email-Id.

2. Insert appropriate values

3. Write query to update Email-Id of a student with rollno 5.

4. Replace the student name from “ABC” to “FEM” of rollno 11.

5. Export the created table into local file system

6. Drop the table

7. Import a given csv dataset from local file system into mongodb collection.

STRUCTURE OF THE COLLECTION

            db.Student.find();

**QUERIES**

**● C**reate a database “Student” with the following attributes Rollno, age, contactNo, Email-Id.

db.createCollection("Student");

* Insert appropriate values

db.Student.insert({rollno:1,age:21,cont:9876,email:"prannay@gmail.com"}); db.Student.insert({rollno:2,age:22,cont:9976,email:"sohan@gmail.com"}); db.Student.insert({rollno:3,age:21,cont:5576,email:"farhaan@gmail.com"}); db.Student.insert({rollno:4,age:20,cont:4476,email:"sakshi@gmail.com"}); db.Student.insert({rollno:5,age:23,cont:2276,email:"sanika@gmail.com"});

**●** Write a query to update the Email-Id of a student with rollno 5.

db.Student.update({rollno:5},{$set:{email:"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:"madhura@gmail.com"}); db.Student.update({rollno:11,name:"ABC"},{$set:{name:"FEM"}})

**●** Export the created table into local file system

mongoexport mongodb+srv://tanmay:tanmay23@cluster0.xbmgopf.mongodb.net/Lab\_9 --collection=Student -- out C:\Users\tanma\Desktop\export\output.json

**●** Drop the table

db.Student.drop();

**●** Import a given csv dataset from local file system into mongodb collection.

mongoimport mongodb+srv://tanmay:tanmay23@cluster0.xbmgopf.mongodb.net/Lab\_9 --collection=new\_Student – type json –file C:\Users\tanma\Desktop\export\output.json

**NoSQL Lab 2**

Perform the following DB operations using MongoDB.

1. Create a collection by name Customers with the following attributes.

Cust\_id, Acc\_Bal, Acc\_Type

2. Insert at least 5 values into the table

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

4. Determine Minimum and Maximum account balance for each customer\_id.

5. Export the created collection into local file system

6. Drop the table

7. Import a given csv dataset from local file system into mongodb collection.

QUERIES:

* Create a collection by name Customers with the following attributes.

               Cust\_id, Acc\_Bal, Acc\_Type.

                 db.createCollection("customers");

* Insert at least 5 values into the table.

  db.customers.insert({cust\_id:1,acc\_bal:25000,acc\_type:”savings”});

  db.customers.insert({cust\_id:2,acc\_bal:45000,acc\_type:”z”});

  db.customers.insert({cust\_id:3,acc\_bal:5000,acc\_type:”x”});

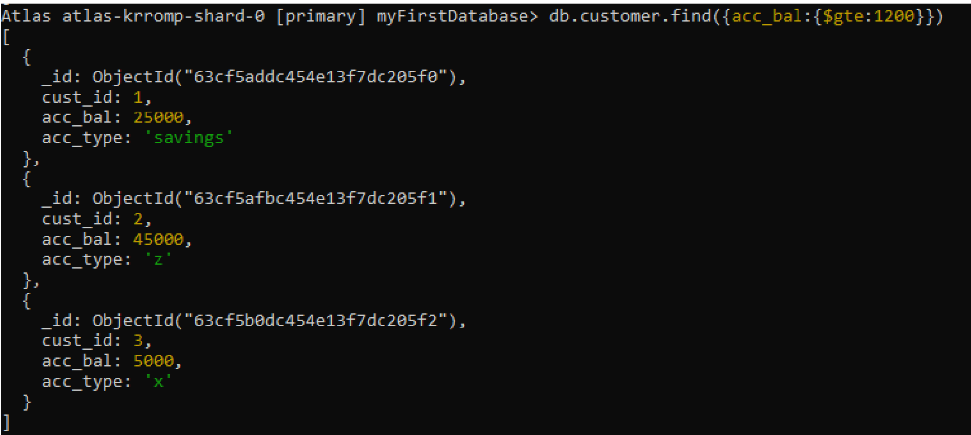
  db.customers.insert({cust\_id:4,acc\_bal:500,acc\_type:”current”});

  db.customers..insert({cust\_id:5,acc\_bal:500,acc\_type:”current"});



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

db.customers.find({acc\_bal:{$gte:1200}})



* Determine Minimum and Maximum account balance for each customer\_id.

                    db.customer.aggregate([{“$group:{“\_id”:”$cust\_id”,”max”:{“$max”:“$acc\_bal”},

                                                                              ”min”:{“$min”:”$acc\_bal”}}}])



* Export the created collection into the local file system.

mongoexport mongodb+srv://tanmay:tanmay23@cluster0.xbmgopf.mongodb.net /Lab\_9 -- collection=customers -- out C:\Users\tanma\Desktop\export\output.json

* Drop the table

db.customers.drop();

* Import a given csv dataset from the local file system into mongodb collection.

mongoimport mongodb+srv://tanmay:tanmay23@cluster0.xbmgopf.mongodb.net/Lab\_9 - -collection=new\_customers – type json –file C:\Users\tanma\Desktop\export\output.json