# VISVESVARAYATECHNOLOGICALUNIVERSITY

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



**LAB REPORT on**

**Database Management Systems (23CS3PCDBM)**

***Submitted by***

**CHITRASHREE K (1BM23CS081)**

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

## BACHELOR OF ENGINEERING

***in***

**COMPUTER SCIENCE AND ENGINEERING**



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

**(Autonomous Institution under VTU)**

**BENGALURU-560019**

**December-2023 to Feb-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 (22CS3PCDBM)” carried out by **CHITRASHREE K (1BM23CS081),** who is a bonafide student of **B. M. S. College of Engineering.** It is in partial fulfilment for the award of **Bachelor of Engineering in Computer** **Science and Engineering** of the Visveshwaraiah Technological University, Belgaum during the year 2024. 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.

|  |  |
| --- | --- |
| Sheetal V A  Associate Professor  Department of CSE, BMSCE | Dr. Kavitha Sooda  Professor HOD  Department of CSE, BMSCE |

`

# Index

|  |  |  |  |
| --- | --- | --- | --- |
| **Sl.**  **No.** | **Date** | **Experiment Title** | **Page No.** |
| 1 | 04-10-2024 | Insurance Database | 1-8 |
| 2 | 11-10-2024 | More Queries on Insurance Database | 9-10 |
| 3 | 18-10-2024 | Bank Database | 11-19 |
| 4 | 25-10-2024 | More Queries on Bank Database | 20-22 |
| 5 | 08-11-2024 | Employee Database | 23-31 |
| 6 | 15-11-2024 | More Queries on Employee Database | 32-36 |
| 7 | 15-12-2024 | Supplier Database | 37-45 |
| 8 | 20-12-2024 | NoSQL Lab 1 | 46-49 |
| 9 | 20-12-2024 | NoSQL Lab 2 | 50-51 |
| 10 | 20-12-2024 | NoSQL Lab 3 | 52-55 |

## Insurance Database

**Question**

### (Week 1)

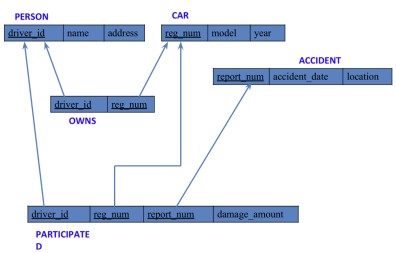
* PERSON (driver\_id: String, name: String, address: String)
* CAR (reg\_num: String, model: String, year: int)
* ACCIDENT (report\_num: int, accident\_date: date, location: String)
* OWNS (driver\_id: String, reg\_num: String)
* PARTICIPATED (driver\_id: String,reg\_num: String, report\_num: int, damage\_amount: int)
* Create the above tables by properly specifying the primary keys and the foreign keys. **-**

Enter at least five tuples for each relation

* Display Accident date and location
* Update the damage amount to 25000 for the car with a specific reg\_num (example 'K A031181' ) 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 insurance\_1BM23CS081;

### use insurance\_1BM23CS081;

### Create table

create table PERSON\_081(

driver\_id varchar(20),

name varchar(30),

address varchar(100),

primary key (driver\_id));

create table CAR\_081(

reg\_num varchar(20),

model varchar (10),

year int,

Primary key (req\_num));

create table ACCIDENT\_081(

report\_num int,

accident\_date date,

location varchar (100),

primary key (report\_num));

create table OWNS\_081(

driver\_id varchar(20),

reg\_num varchar(20),

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

foreign key (driver\_id) references PERSON\_081 (driver\_id),

foreign key (reg\_num) references CAR\_081(reg\_num));

create table PARTICIPATED\_081(

driver\_id varchar (20),

reg\_num varchar(20),

report\_num int,

damage\_amount int,

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

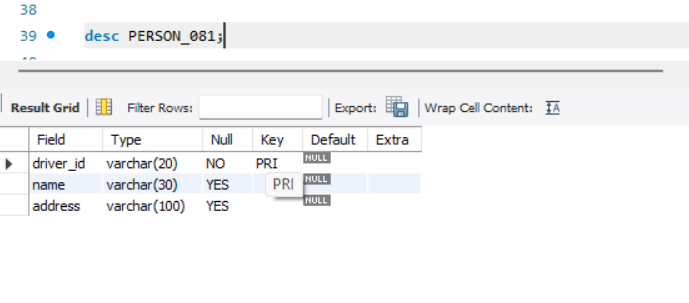
foreign key (driver\_id) references PERSON\_081 (driver\_id),

foreign key (reg\_num) references CAR\_081 (reg\_num),

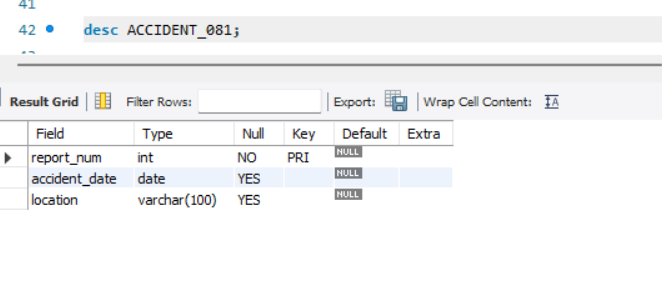
foreign key (report\_num) references ACCIDENT\_081 (report\_num));

### Structure of the table

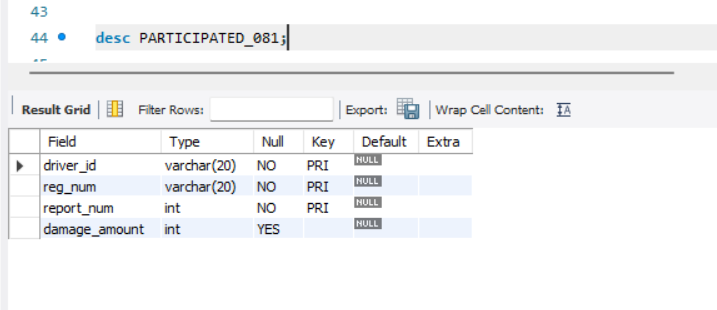
PERSON\_081



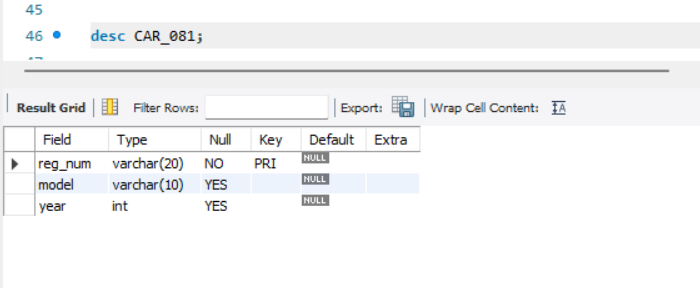
ACCIDENT\_081



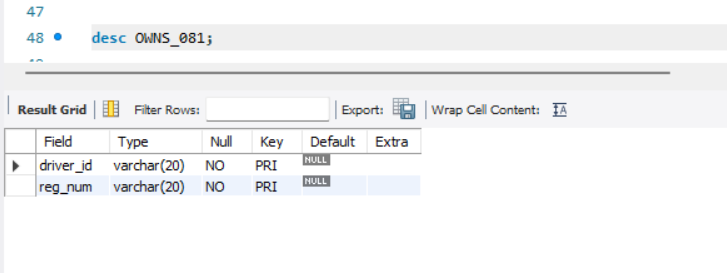
PARTICIPATED\_081



CAR\_081



OWNS\_081



### Inserting Values into the table

insert into PERSON\_081 values

("A01", "Richard", "Sri Nagar"),

("A02", "Pradeep", "Raj Nagar"),

("A03", "Smith", "Ashok Nagar"),

("A04", "Venu", "N R Colony"),

("A05", "John", "Hanu Nagar");

insert into CAR\_081 values

("KA052250", "Indica", 1990),

("KA031181", "Lancer", 1957),

("KA095477", "Toyota", 1998),

("KA053408", "Honda", 2008),

("KA041702", "Audi", 2005);

insert into OWNS\_081 values

("A01", "KA052250"),

("A02", "KA031181"),

("A03", "KA095477"),

("A04", "KA053408"),

("A05", "KA041702");

insert into ACCIDENT\_081 values

(11, "01-01-03", "Mysore Rd"),

(12, "02-02-04", "SE Circle"),

(13, "21-01-03", "Bull Temple Rd"),

(14, "17-02-08", "Mysore Rd"),

(15, "04-03-05", "KR Puram");

insert into PARTICIPATED\_081 values

("A01", "KA052250", 11, 10000),

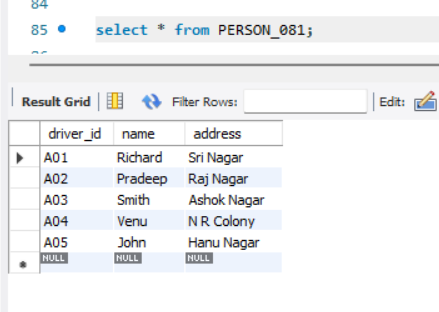
("A02", "KA031181", 12, 50000),

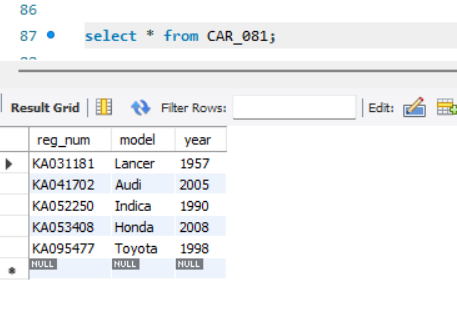
("A03", "KA053408", 13, 25000),

("A04", "KA095477", 14, 3000),

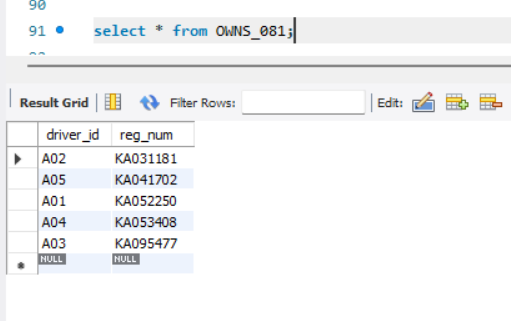
("A05", "KA041702", 15, 5000);

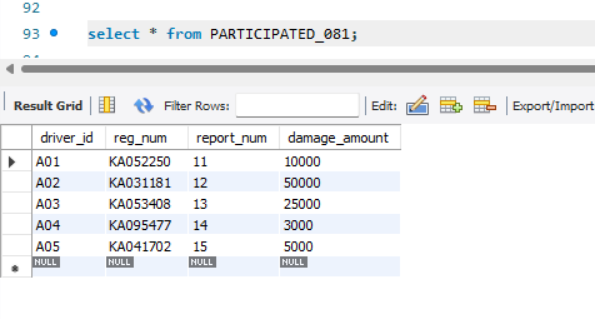
**INSERTED VALUES**

****

****

### 

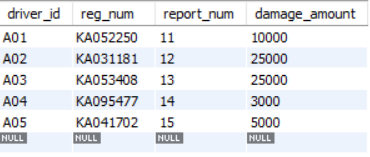




### Queries

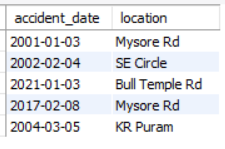
**1. Update the damage amount to 25000 for the car with a specific reg-num (example 'KA031181' ) for which the accident report number was 12.**

Query : update PARTICIPATED\_081 set damage\_amount = 25000 where reg\_num = "KA031181" and report\_num = 12;



**2.** **Display Accident date and location**

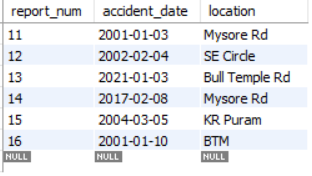
Query : select accident\_date, location from ACCIDENT\_081;



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

Query : insert into ACCIDENT\_081

values (16, "01-01-10", "BTM");

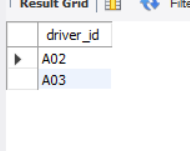


**4.Display driver id who did the accident damage greater than or equal to Rs.25000**

Query : select driver\_id

from PARTICIPATED\_081

where damage\_amount >=25000;

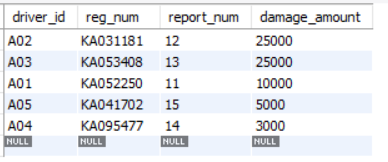


### More Queries On Insurance Database

### (Week 2)

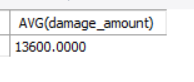
**1.List The Entire Participated Relation In The Descending Order Of Damage Amount.**

**Query :** select \* from PARTICIPATED\_081 order by damage\_amount desc;

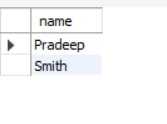


**2.** **Find The Average Damage Amount**

Query : select avg(damage\_amount) from PARTICIPATED\_081;



**3.List The Name Of Driver Whose Damage Is Greater Than The Average Damage Amount.**

Query : select name from PERSON\_081, PARTICIPATED\_081 where PERSON\_081.driver\_id = PARTICIPATED\_081.driver\_id and PARTICIPATED\_081.damage\_amount > (select avg(damage\_amount) from PARTICIPATED\_081);

## Bank Database

**Question**

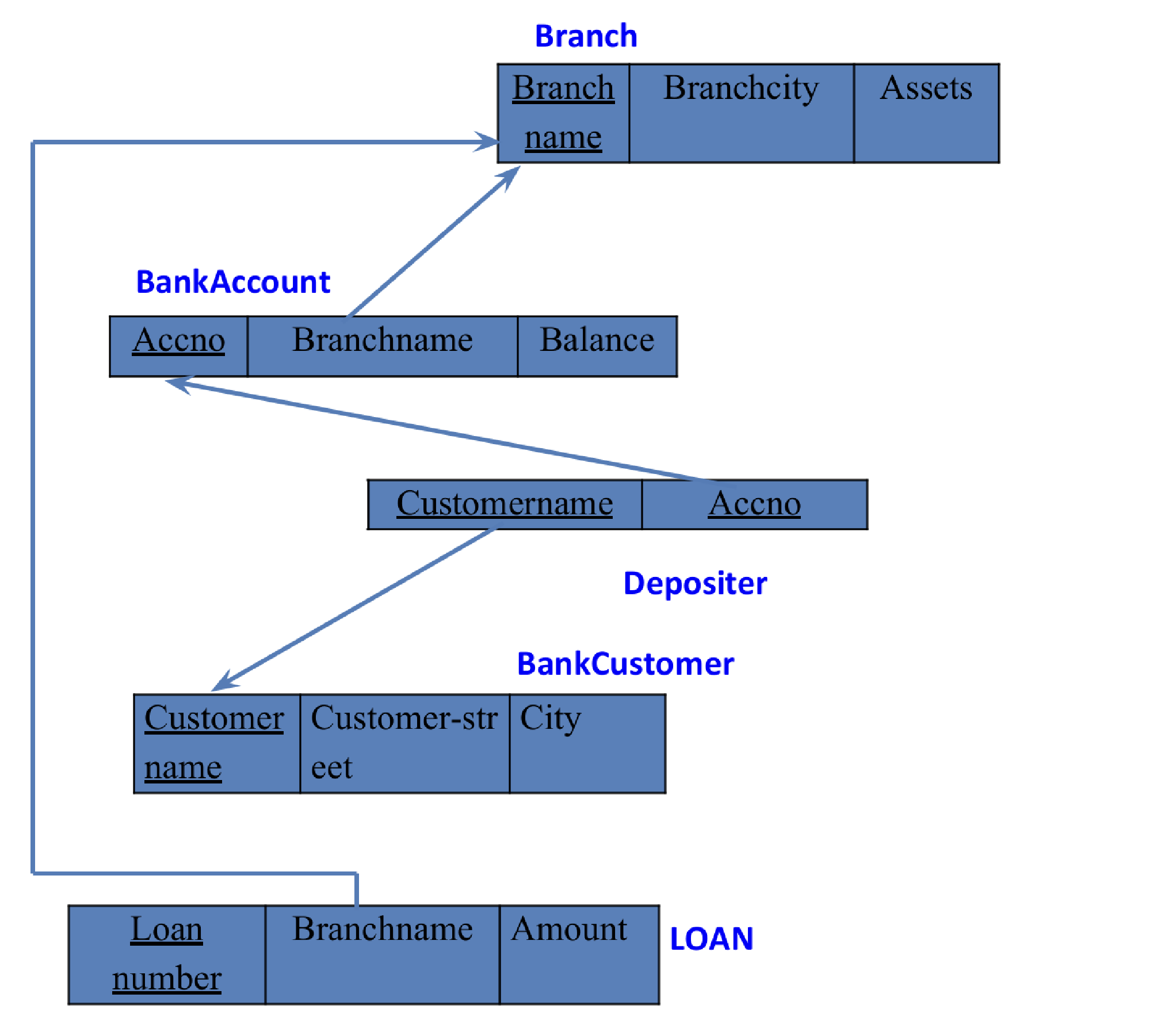
### (Week 3)

* Branch (branch-name: String, branch-city: String, assets: real)
* BankAccount(accno: int, branch-name: String, balance: real)
* BankCustomer (customer-name: String, customer-street: String, customer-city: String) **-**

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

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

### Schema Diagram



### Create database

### create database bank\_database\_1BM23CS081;

### use bank\_database\_1BM23CS081;

### Create table

### create table BRANCH\_081(

### branch\_name varchar(100),

### branch\_city varchar(100),

### assets real,

### primary key (branch\_name));

### create table BANK\_ACCOUNT\_081(

### acc\_no int,

### branch\_name varchar(100),

### balance real,

### primary key(acc\_no),

### foreign key(branch\_name) references BRANCH\_081(branch\_name));

### create table BANK\_CUSTOMER\_081(

### customer\_name varchar(100),

### customer\_street varchar(100),

### customer\_city varchar(100),

### primary key(customer\_name));

### create table DEPOSITOR\_081(

### customer\_name varchar(100),

### acc\_no int,

### primary key(customer\_name, acc\_no),

### foreign key(acc\_no) references BANK\_ACCOUNT\_081(acc\_no),

### foreign key(customer\_name) references BANK\_CUSTOMER\_081(customer\_name));

### create table LOAN\_081(

### loan\_number int,

### branch\_name varchar(100),

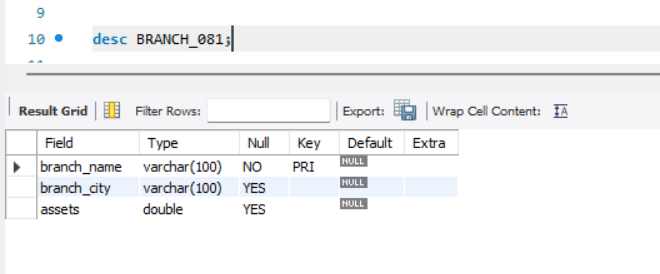
### amount real,

### primary key(loan\_number),

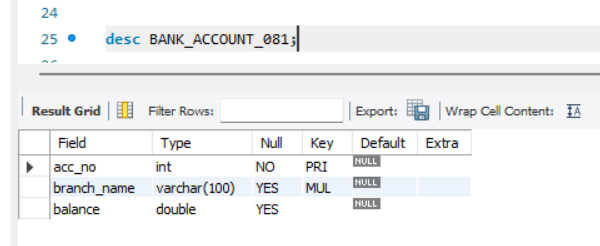
### foreign key(branch\_name) references BRANCH\_081(branch\_name));

### Structure of the table

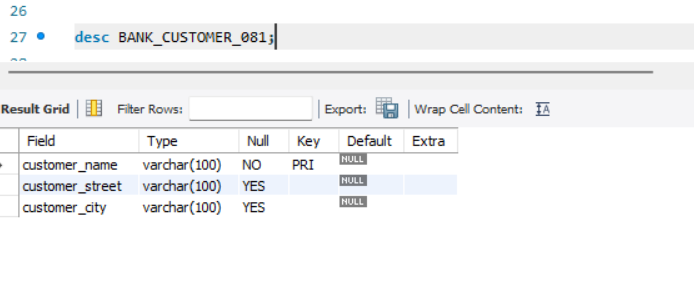
BRANCH\_081



BANK\_ACCOUNT\_081

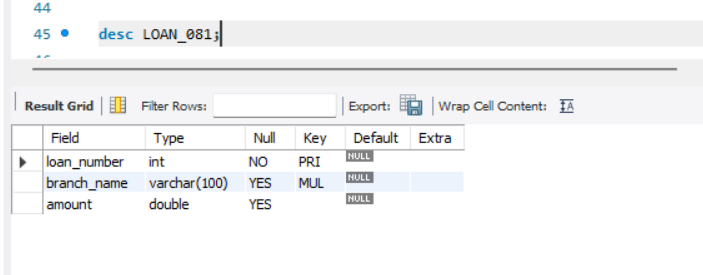


### BANK\_CUSTOMER\_081

DEPOSITOR\_081

### 

LOAN\_081



### Inserting Values to the table

insert into BRANCH\_081(branch\_name, branch\_city, assets)

values ("SBI\_Chamrajpet", "Bangalore", 50000),

("SBI\_ResidencyRoad", "Bangalore", 10000),

("SBI\_Shivaginagar", "Bombay", 20000),

("SBI\_ParlimentRoad", "Delhi", 10000),

("SBI\_Jantarmantar", "Delhi", 20000);

insert into BANK\_ACCOUNT\_081(acc\_no, branch\_name, balance)

values (1, "SBI\_Chamrajpet", 2000),

(2, "SBI\_ResidencyRoad", 5000),

(3, "SBI\_Shivaginagar", 6000),

(4, "SBI\_ParlimentRoad", 9000),

(5, "SBI\_Jantarmantar", 8000),

(6, "SBI\_Shivaginagar", 4000),

(8, "SBI\_ResidencyRoad", 4000),

(9, "SBI\_ParlimentRoad", 3000),

(10, "SBI\_ResidencyRoad", 5000),

(11, "SBI\_Jantarmantar", 2000);

insert into BANK\_CUSTOMER\_081(customer\_name, customer\_street,customer\_city)

values ("Avinash", "Bull Temple Road", "Bangalore"),

("Dinesh", "Bannergatta Road", "Bangalore"),

("Mohan", "National College Road", "Bangalore"),

("Nikil", "Akbar Road", "Delhi"),

("Ravi", "Prithiviraj Road", "Delhi");

insert into DEPOSITOR\_081(customer\_name, acc\_no)

values ("Avinash", 1),

("Dinesh", 2),

("Nikil", 4),

("Ravi", 5),

("Avinash", 8),

("Nikil", 9),

("Dinesh", 10),

("Nikil", 11);

insert into LOAN\_081(loan\_number, branch\_name, amount)

values (1, "SBI\_Chamrajpet", 1000),

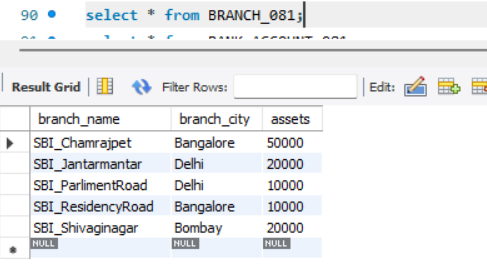
(2, "SBI\_ResidencyRoad", 2000),

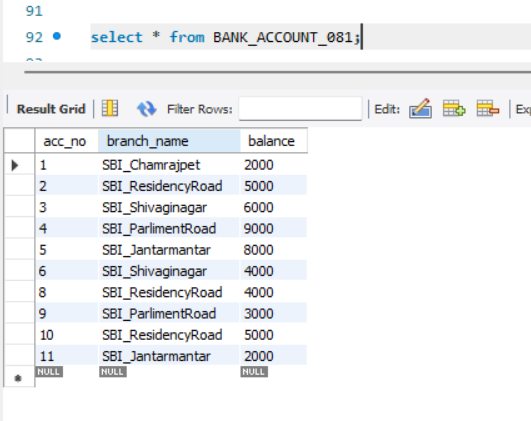
(3, "SBI\_Shivaginagar", 3000),

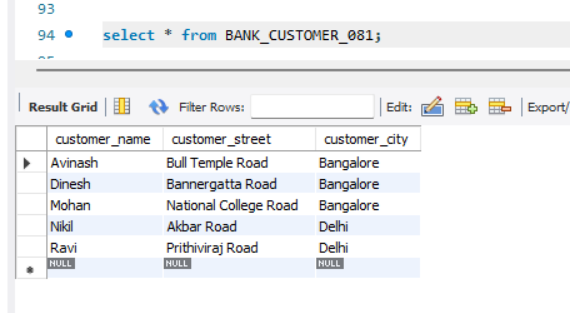
(4, "SBI\_ParlimentRoad", 4000),

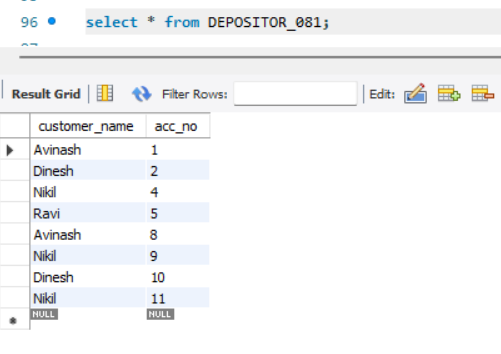
(5, "SBI\_Jantarmantar", 5000);

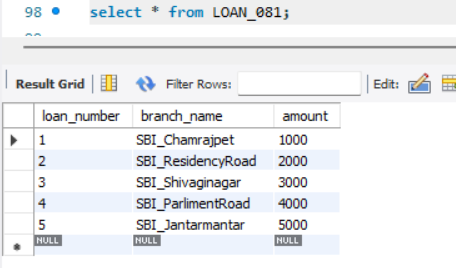
**Inserted Values**







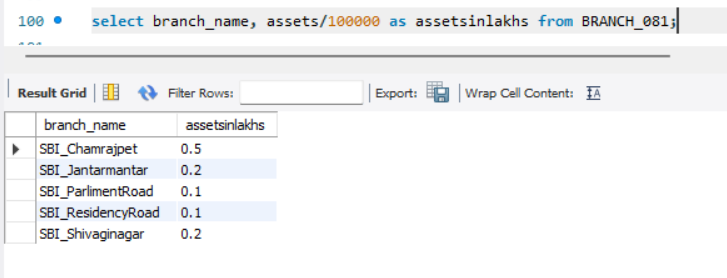




### Queries

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

Query : select branch\_name, assets/100000 as assetsinlakhs from BRANCH\_081;



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

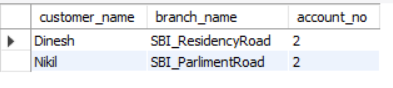
Query : select d.customer\_name, b.branch\_name, count(d.acc\_no) as account\_no

from DEPOSITOR\_081 d, BANK\_ACCOUNT\_081 b

where d.acc\_no=b.acc\_no

group by d.customer\_name, b.branch\_name

having count(d.acc\_no)>=2;



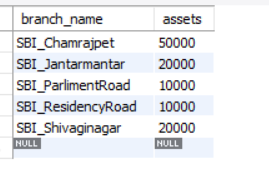
### (Week 4)

### More Queries On Bank Database

**1. Retrieve all branches and their respective total assets**

Query : select branch\_name, assets

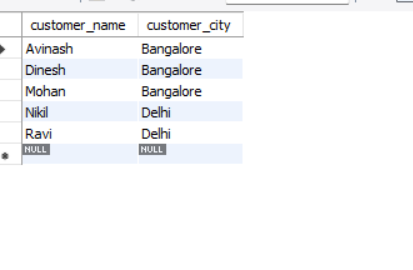
from BRANCH\_081;



**2. List all customers who live in a particular city**

Query : select customer\_name, customer\_city

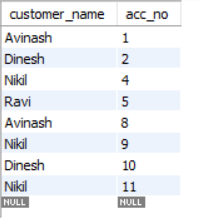
from BANK\_CUSTOMER\_081;



**3. List all customers with their account numbers**

Query : select customer\_name, acc\_no

from DEPOSITOR\_081;

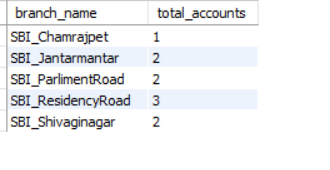


**4. Get the number of accounts held at each branch**

Query : select branch\_name, count(acc\_no) as total\_accounts

from BANK\_ACCOUNT\_081

GROUP BY branch\_name;

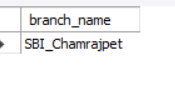


**5. Retrieve the branch with the smallest total loan amount**

Query : select branch\_name

from LOAN\_081

where amount=(select min(amount) from LOAN\_081);

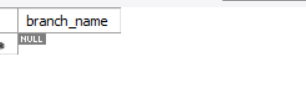


**6. Find all branches that have no loans issued**

Query : select branch\_name

from BRANCH\_081

where branch\_name not in(select branch\_name from LOAN\_081);

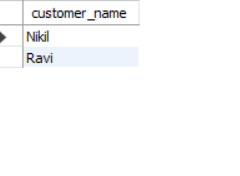


**7.Find all customers who have accounts with a balance greater than a specified amount (100000)**

Query : select d.customer\_name

from DEPOSITOR\_081 d, BANK\_ACCOUNT\_081 b

where d.acc\_no=b.acc\_no and b.balance>5000;

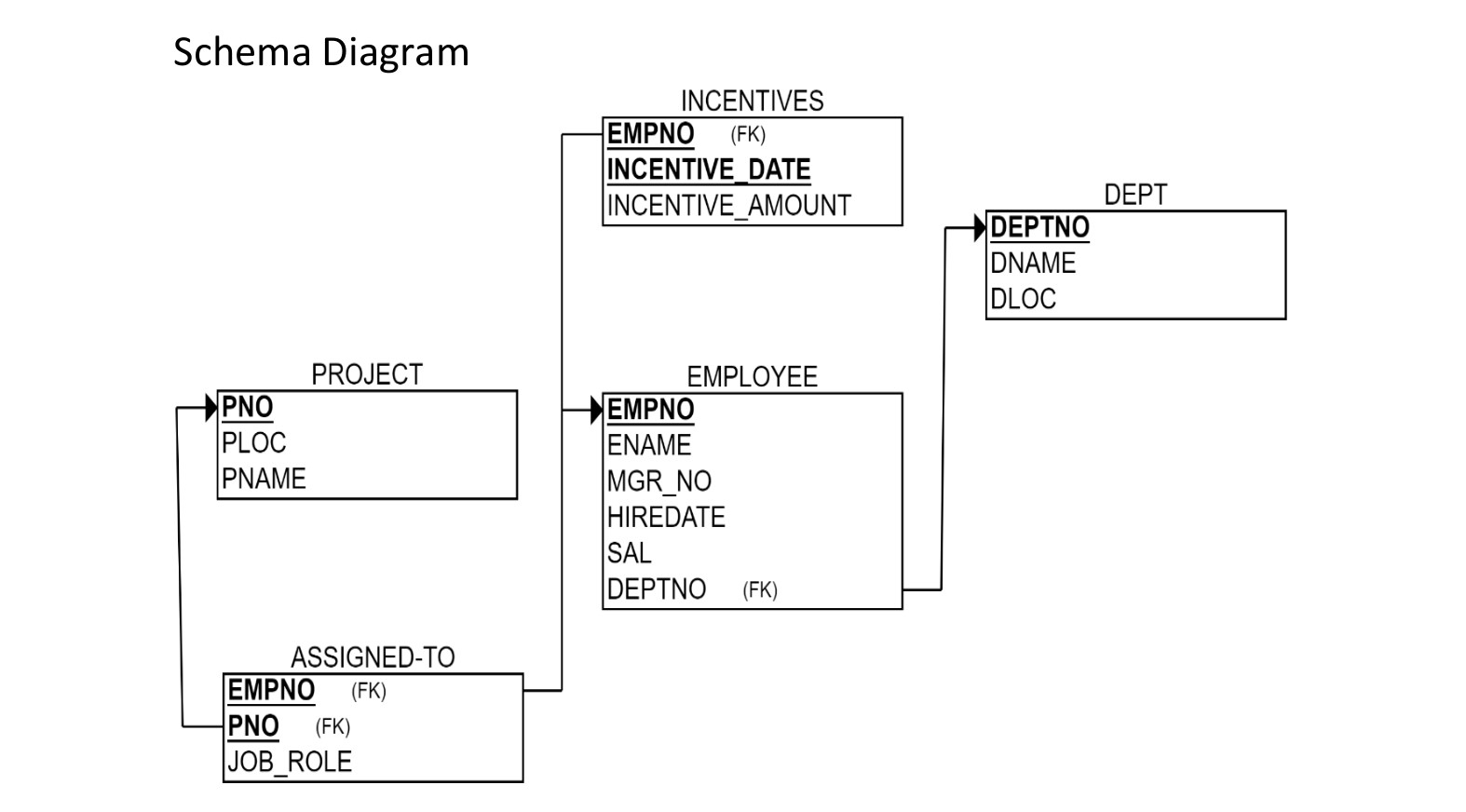


## Employee Database

## (Week 5)

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.

### Schema Diagram



### Create database

### create database employee\_1BM23CS081;

### use employee\_1BM23CS081;

### Create table

### create table DEPT\_081(

### dept\_no int ,

### dname varchar(30),

### dloc varchar(55),

### primary key(dept\_no));

### create table PROJECT\_081(

### pno int ,

### ploc varchar(30),

### pname varchar(55),

### primary key(pno));

### create table EMPLOYEE\_081(

### empno int ,

### ename varchar(55),

### mgr\_no int,

### hiredate date,

### sal int,

### dept\_no int ,

### primary key(empno),

### foreign key(dept\_no) references DEPT\_081(dept\_no));

### create table INCENTIVES\_081(

### empno int ,

### incentive\_date date,

### incentive\_amount int,

### primary key(empno , incentive\_date ),

### foreign key(empno) references EMPLOYEE\_081(empno));

### create table ASSIGNED\_081(

### empno int,

### pno int,

### job\_role varchar(55),

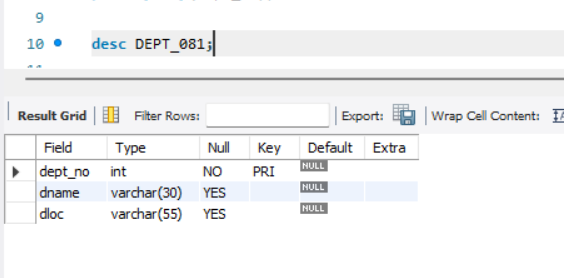
### primary key(empno, pno),

### foreign key(empno) references EMPLOYEE\_081(empno),

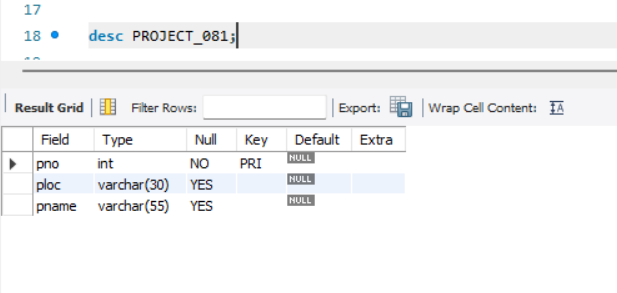
### foreign key(pno) references PROJECT\_081(pno));

### Structure of the table

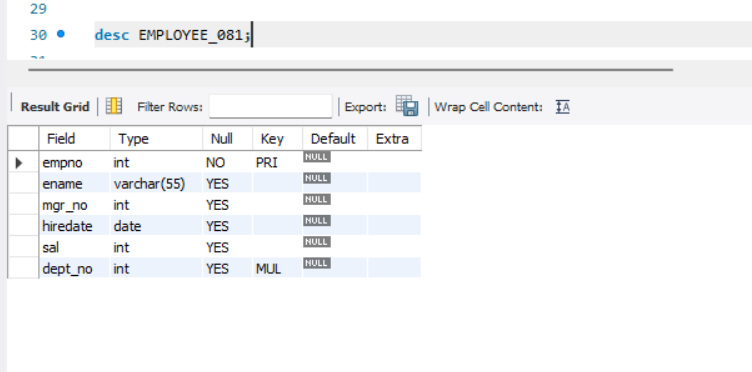
DEPT\_081



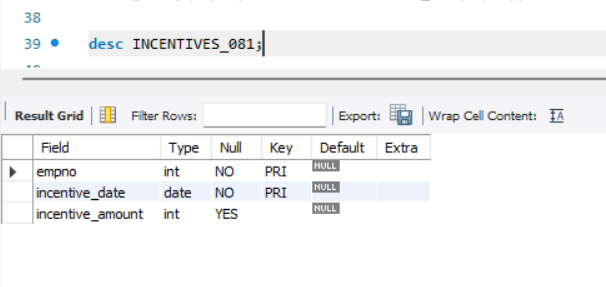
PROJECT\_081



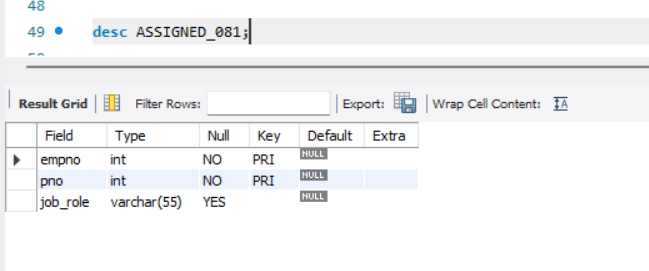
EMPLOYEE\_081



INCENTIVES\_081



ASSIGNED\_081



### Inserting Values to the table

insert into DEPT\_081(dept\_no, dname, dloc)

values (1, "IT", "MUMBAI"),

(2, "AI", "BANGALORE"),

(3, "DATA SCIENCE", "DELHI"),

(4, "ML", "HYDERABAD"),

(5, "ADMINISTRATION", "MYSORE"),

(6, "LEGAL", "MANGALORE");

insert into PROJECT\_081(pno, ploc, pname)

values (1, "BANGALORE", "WEB DEV"),

(2, "MYSORE", "APP DEV"),

(3, "DELHI", "NLP"),

(4, "MUMBAI", "AR GAME"),

(5, "HYDERABAD", "IOT AUTOMATION"),

(6, "BANGALORE", "AI CHATBOT"),

(7, "MUMBAI", "CRYPTO CURRENCY"),

(8, "MYSORE", "TRAFFIC MANAGEMENT");

insert into EMPLOYEE\_081(empno, ename, mgr\_no, hiredate, sal, dept\_no)

values (1, "ABAY", 5, '2024-01-15', 48000, 4),

(2, "JOHN", 1, '2023-03-22', 50000, 1),

(3, "DANIEL", 2, '2022-06-10', 82000, 5),

(4, "DAVID", 8, '2023-09-30', 95000, 6),

(5, "JAMES", 1, '2020-12-05', 55000, 2),

(6, "THOMAS", 5, '2017-11-01', 65000, 1),

(7, "ROBERT", 8, '2019-04-18', 75000, 5),

(8, "LAURA", 2, '2018-08-25', 80000, 3);

insert into INCENTIVES\_081(empno, incentive\_date, incentive\_amount)

values (4, '2024-05-20', 1800),

(3, '2024-06-30', 2500),

(8,'2024-09-18', 1700),

(7, '2024-03-15', 2200),

(1, '2024-02-01', 1950);

insert into ASSIGNED\_081(empno, pno, job\_role)

values (6, 8, "MANAGER"),

(1, 3, "AI RESEACHER"),

(3, 2, "APP DEVELOPER"),

(5, 5, "DEVOPS ENGINEER"),

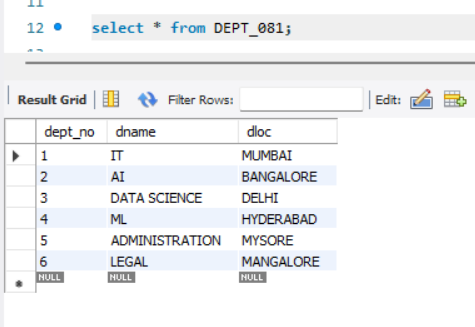
(2, 4, "GAME DEVELOPER"),

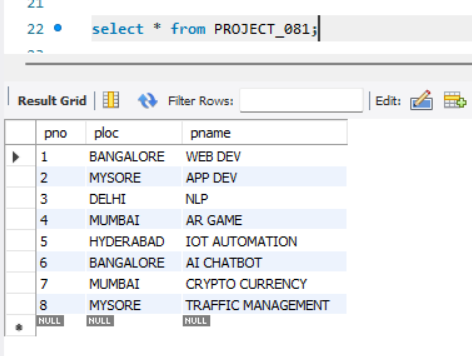
(7, 1, "WEB DEVELOPER"),

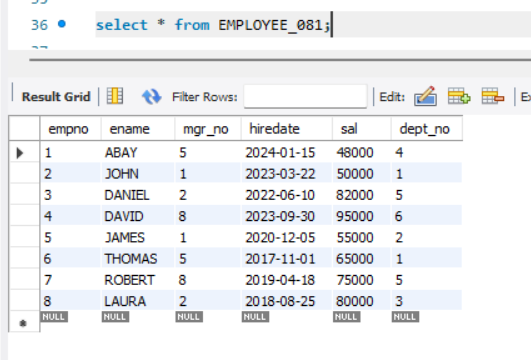
(4, 7, "ANALYST"),

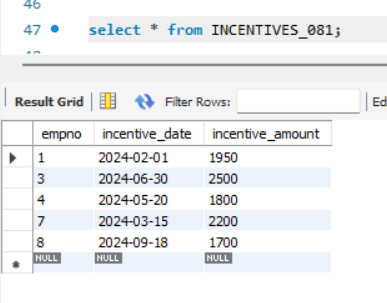
(8, 6, "AIML ENGINEER");

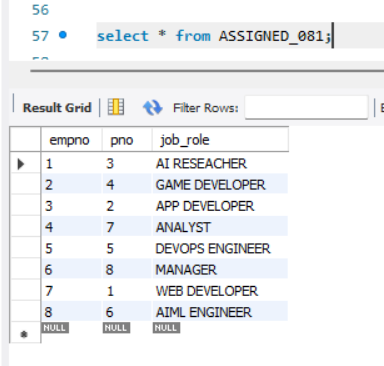
**Inserted Values**

****

****

****

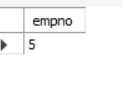
****

****

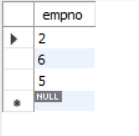
**Queries**

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

**Query :** select ASSIGNED\_081.empno from ASSIGNED\_081, PROJECT\_081 where ASSIGNED\_081.pno = PROJECT\_081.pno and PROJECT\_081.ploc in ("Bengaluru", "Mysuru", "Hyderabad");



**2.Get Employee ID’s of those employees who didn’t receive incentives** Query : select empno from employee\_204 where empno not in (select empno from incentives\_204);

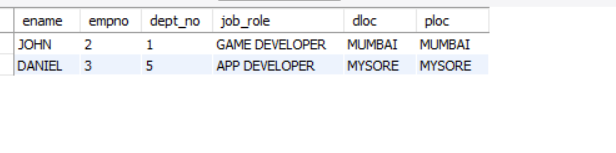


**3.Write a SQL query to find the employees name, number, dept, job\_role, department location and project location who are working for a project location same as his/her department location.**

Query : select e.ename, e.empno, d.dept\_no, a.job\_role, d.dloc, p.ploc

from EMPLOYEE\_081 e, DEPT\_081 d, ASSIGNED\_081 a, PROJECT\_081 p

where e.dept\_no=d.dept\_no and a.empno=e.empno and a.pno=p.pno and p.ploc=d.dloc;



### (Week 6)

**More Queries on Employee Database**

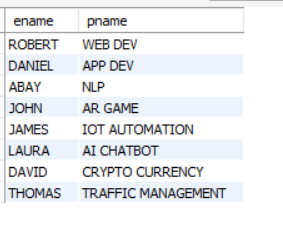
### Queries

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

Query : select e.ename, p.pname

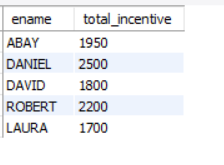
from ASSIGNED\_081 a, EMPLOYEE\_081 e, PROJECT\_081 p

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



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

Query : select e.ename , i.incentive\_amount as total\_incentive from EMPLOYEE\_081 e, INCENTIVES\_081 i where e.empno=i.empno;



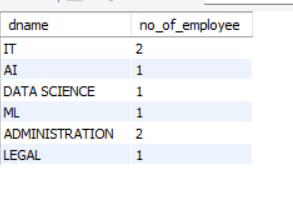
**3.List departments along with the number of employees in each department2019.**

Query : select d.dname, count(e.ename) as no\_of\_employee

from DEPT\_081 d, EMPLOYEE\_081 e

where d.dept\_no=e.dept\_no

group by e.dept\_no;

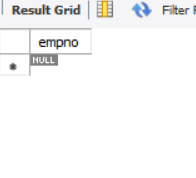


**4.** **Find employees who have not been assigned to any project**

Query : select e.empno

from EMPLOYEE\_081 e

where e.empno not in(select a.empno from ASSIGNED\_081 a);

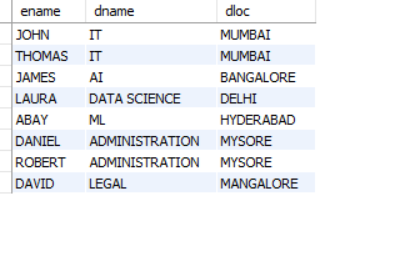


**5.List all employees along with their department names and location**

Query : select e.ename, d.dname, d.dloc

from EMPLOYEE\_081 e, DEPT\_081 d

where e.dept\_no=d.dept\_no;

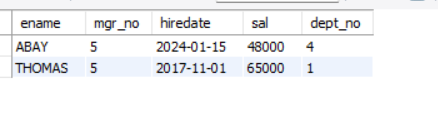
****

**6.Retrieve the details of employees who work under a specific manager (e.g., manager with empno = 5)**

Query : select ename, mgr\_no, hiredate, sal, dept\_no

from EMPLOYEE\_081

where mgr\_no=5;

****

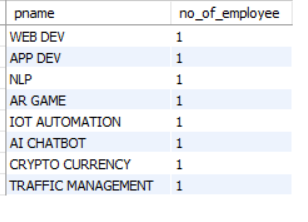
**7.List all projects that have employees assigned and the number of employees on each project:**

Query : select p.pname, count(a.empno) as no\_of\_employee

from ASSIGNED\_081 a, PROJECT\_081 p

where a.pno=p.pno

group by p.pname;

****

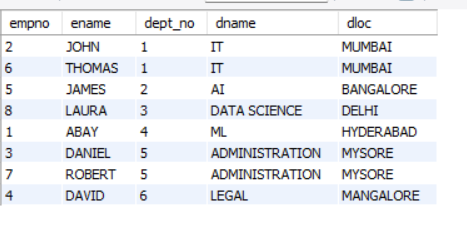
**8.** **Find employees with the same manager and list their department details**

Query : select e1.empno, e1.ename, d.dept\_no, d.dname, d.dloc

from EMPLOYEE\_081 e1

join EMPLOYEE\_081 e2 on e1.mgr\_no=e2.mgr\_no and e1.empno<> e2.empno

join DEPT\_081 d on e1.dept\_no=d.dept\_no;

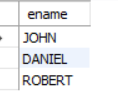


**9. Retrieve all employees who have the role of 'Developer' on any project:**

Query : select e.ename

from EMPLOYEE\_081 e, ASSIGNED\_081 a

where a.empno=e.empno and a.job\_role like "%DEVELOPER";

****

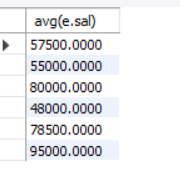
**10.** **Display the department-wise average salary of employees:**

Query : select avg(e.sal)

from EMPLOYEE\_081 e, DEPT\_081 d

where e.dept\_no=d.dept\_no

group by e.dept\_no;

****

**SUPPLIER DATABASE**

**Question**

**(Week 7)**

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.

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

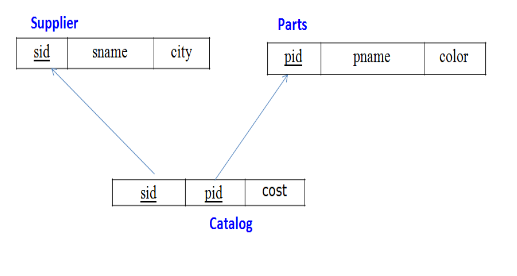
5. Find the snames of suppliers who supply every red part.

6. Find the pnames 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 sname of the supplier who charges the most for that part.

**Schema Diagram**

****

**Create Database**

create database supplier\_database\_142;

use supplier\_database\_142;

**Create Tables**

create table suppliers(

sid int,

sname varchar(100),

city varchar(100),

primary key(sid));

create table parts(

pid int,

pname varchar(100),

color varchar(100),

primary key(pid));

create table catalog(

sid int,

pid int,

cost int,

primary key(sid,pid),

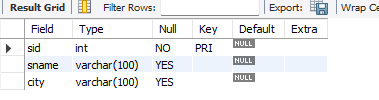
foreign key (pid) references parts(pid),

foreign key (sid) references suppliers(sid)

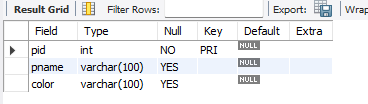
on delete cascade on update cascade);

**Structure Of The Table**

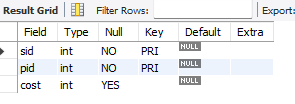
Desc suppliers;



Desc parts;



Desc catalog;



**Inserting Of Values**

insert into suppliers values(

10001,

'Acme widget',

'Bangalore');

insert into suppliers values(

10002,

'Johns',

'Kolkata');

insert into suppliers values(

10003,

'Vimal',

'Mumbai');

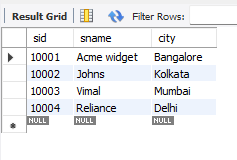
insert into suppliers values(

10004,

'Reliance',

'Delhi');

select \* from suppliers;



insert into parts values(

20001,

'Book',

'Red');

insert into parts values(

20002,

'Pen',

'Red');

insert into parts values(

20003,

'Pencil',

'Green');

insert into parts values(

20004,

'Mobil',

'Green');

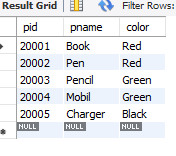
insert into parts values(

20005,

'Charger',

'Black');

select \* from parts;



insert into catalog values(

10001,

20001,

10);

insert into catalog values(

10001,

20002,

10);

insert into catalog values(

10001,

20003,

30);

insert into catalog values(

10001,

20004,

10);

insert into catalog values(

10001,

20005,

10);

insert into catalog values(

10002,

20001,

10);

insert into catalog values(

10002,

20002,

20);

insert into catalog values(

10003,

20003,

30);

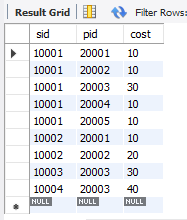
insert into catalog values(

10004,

20003,

40);

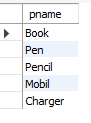
select \* from catalog;



**More Queries On Suppliers Database**

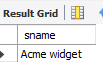
**1. Find the pnames of parts for which there is some supplier.**

select distinct p.pname from parts p join catalog c on p.pid = c.pid;



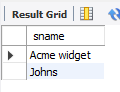
**2. Find the snames of suppliers who supply every part.**

select sname from suppliers s where not exists( select p.pid from parts p where not exists(select c.pid from catalog c where c.pid = p.pid and c.sid = s.sid));



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

select sname from suppliers s where not exists(select p.pid from parts p where p.color = 'Red' and not exists(select c.pid from catalog c where c.pid = p.pid and c.sid = s.sid));



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

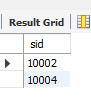
select p.pname from parts p join catalog c on p.pid = c.pid join suppliers s on c.sid = s.sid where s.sname = 'Acme widget'and p.pid not in (select c1.pid from catalog c1 where c1.sid <>

(select sid from suppliers where sname = 'Acme widget'));



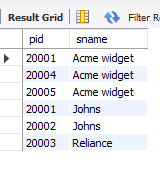
**5. Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over all the suppliers who supply that part).**

select c.sid from catalog c join (select pid,avg(cost) as avg\_cost from catalog group by pid) avg\_cost\_table on c.pid = avg\_cost\_table.pid where c.cost > avg\_cost\_table.avg\_cost;



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

select p.pid,s.sname from parts p join catalog c on p.pid = c.pid join suppliers s on c.sid = s.sid where (c.pid,c.cost) in (select pid,max(cost) from catalog group by pid);



## NoSQL Lab 1

**Question**

### (Week 8)

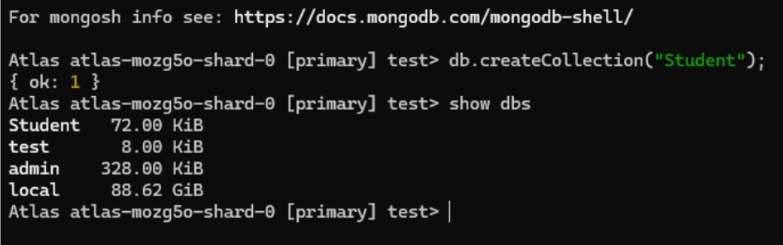
Perform the following DB operations using MongoDB.

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 10.
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.

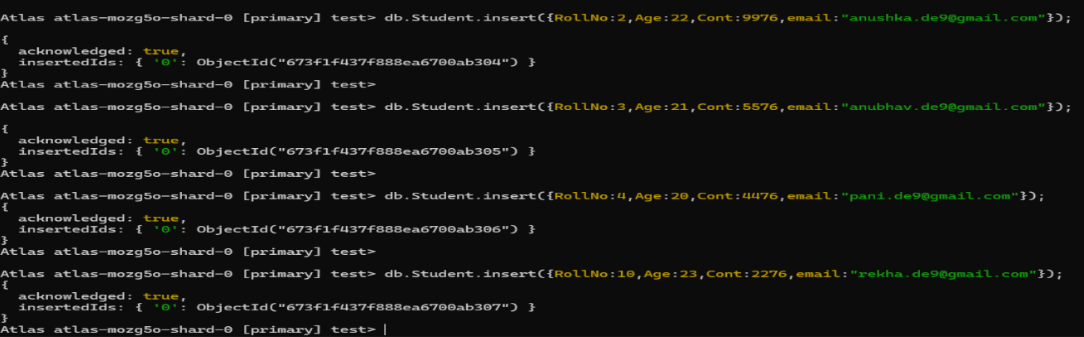
**1. Create a database “Student” with the following attributes Rollno, Age, ContactNo, Email-Id.**

db.createCollection("Student");

show dbs



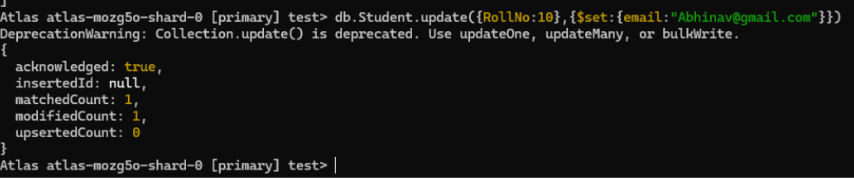
**2. Insert appropriate values**



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

db.Student.find()





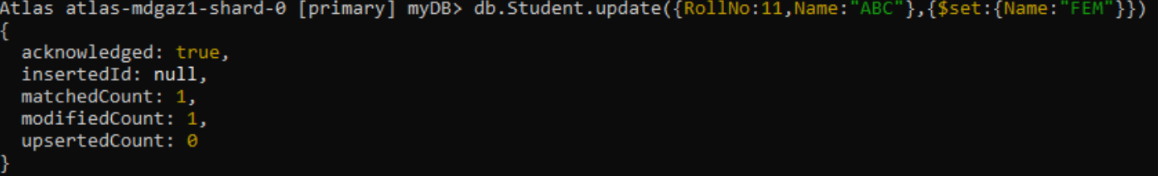
**4. Replace the student name from “ABC” to “FEM” of rollno 11.**

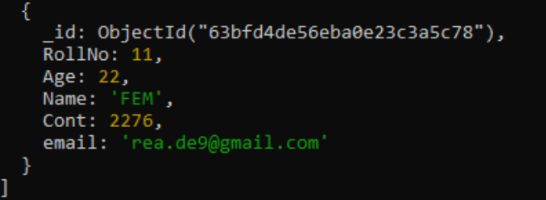
Insert appropriate data:

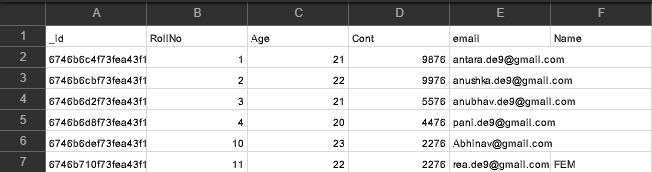
db.Student.insert({RollNo:11,Age:22,Name:

"ABC",Cont:2276,email:"[rea.de9@gmail.com](mailto:rea.de9@gmail.com)"});







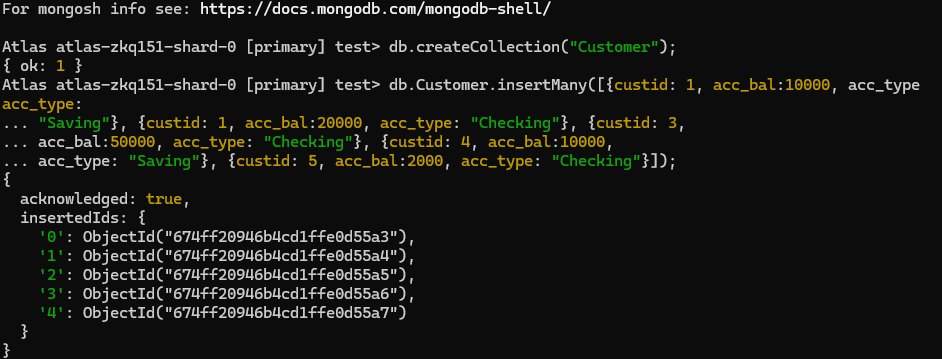


**NoSQL- Customer Database**

(**Week 9)**

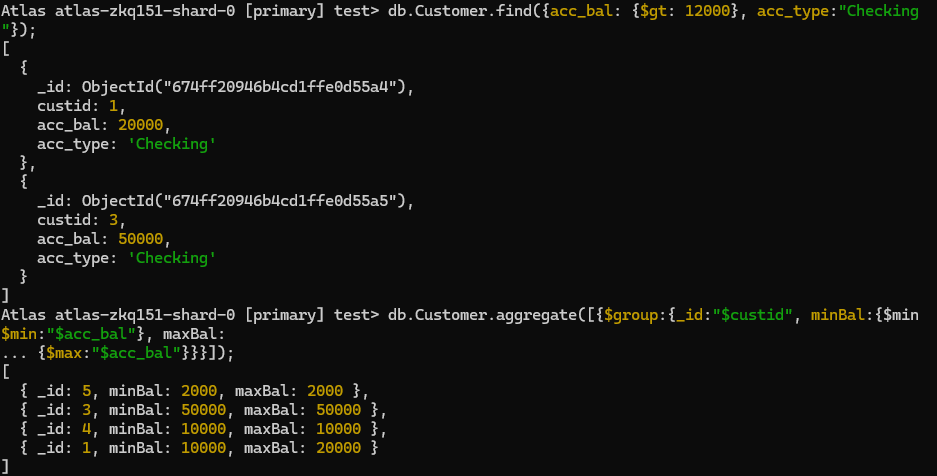
**1. Create a collection by name Customers with the following attributes.**

Cust\_id, Acc\_Bal, Acc\_Type and Insert at least 5 values into the table

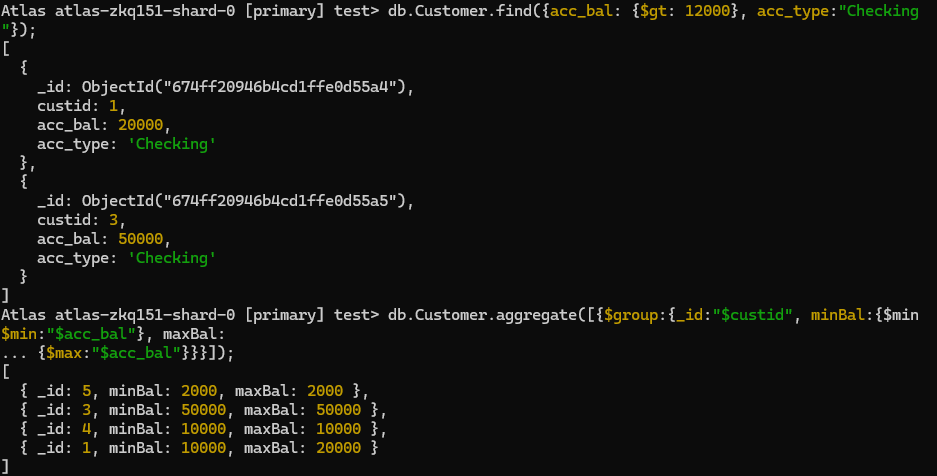


**2. Write a query to display those records whose total account balance is greater than**

12000 of account type ‘Z’ for each customer\_id.



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

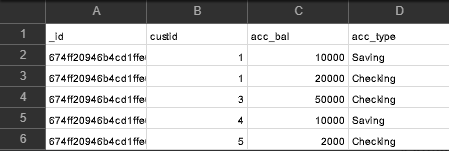


**4. Export the created collection into local file system**

**5. Drop the table**



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

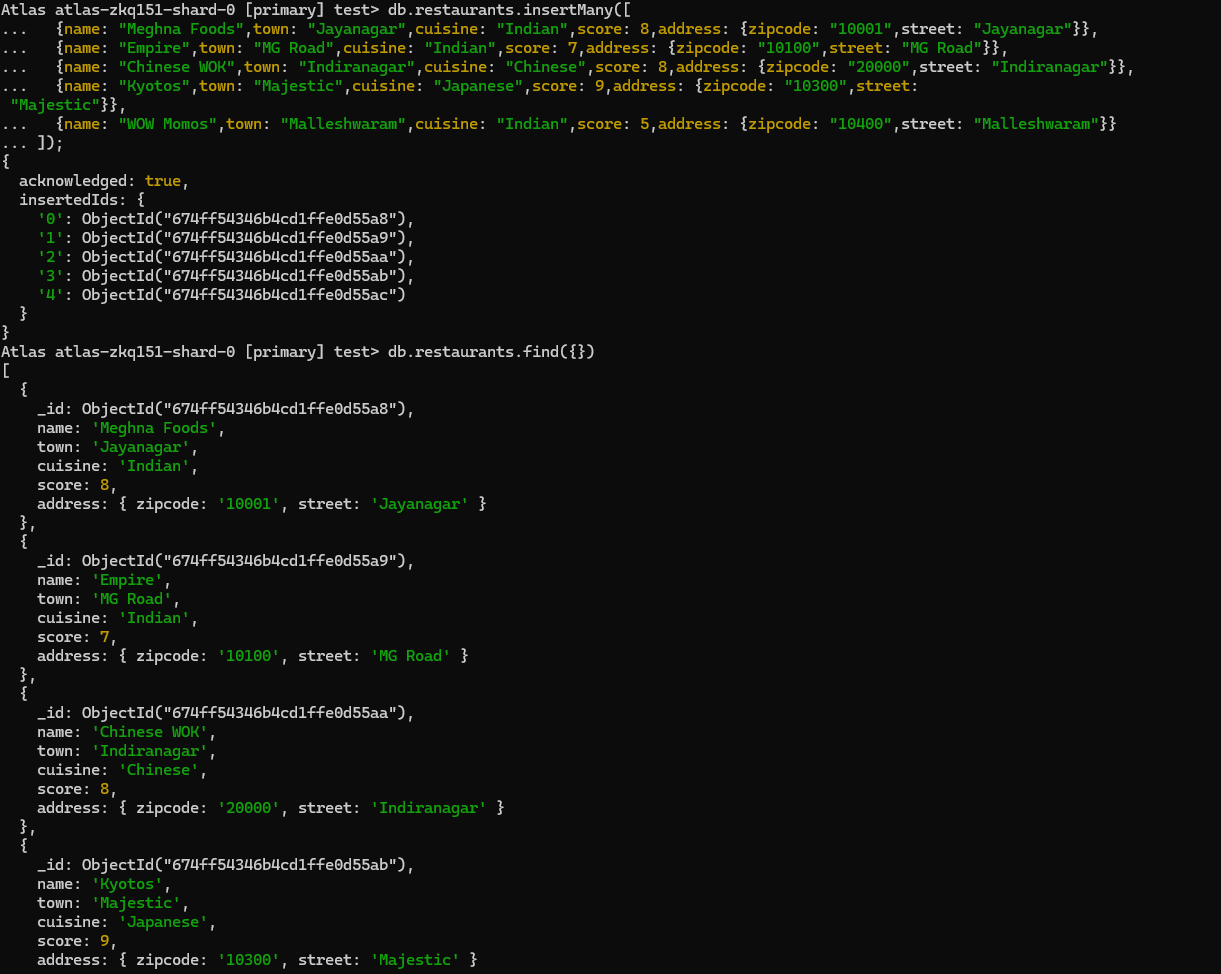


**NoSQL Restaurants Database**

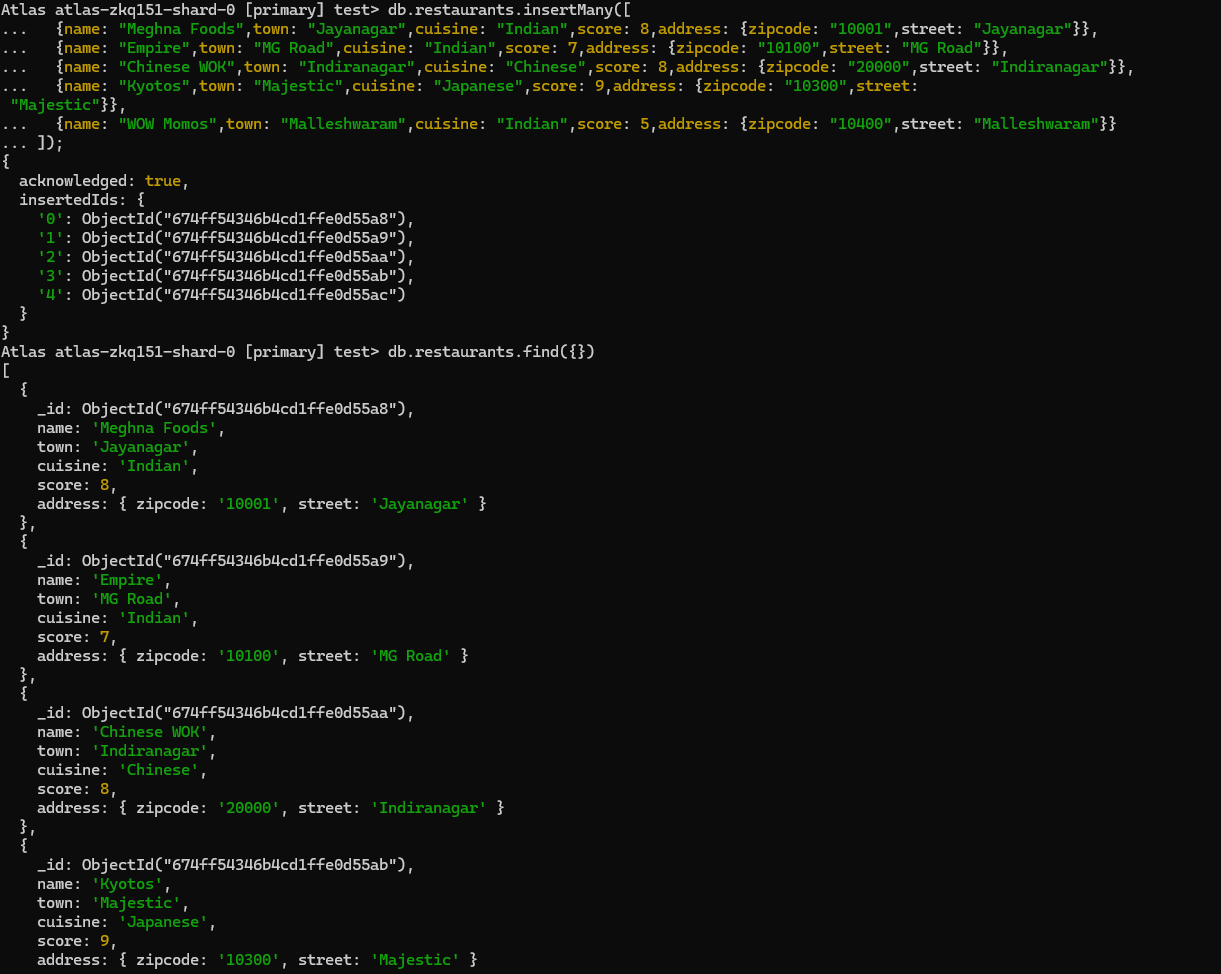
(**Week-10)**

**1.In MongoDB create a collection for “Restaurant” and insert atleast five records**

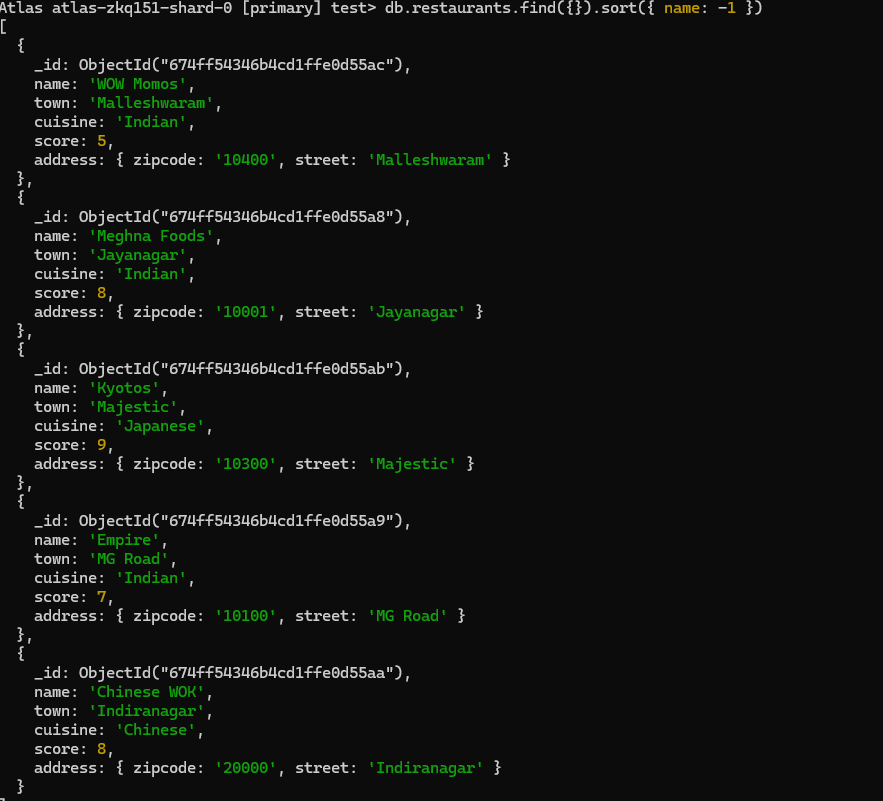




**2. Write a MongoDB query to display all the documents in the collection restaurants.**



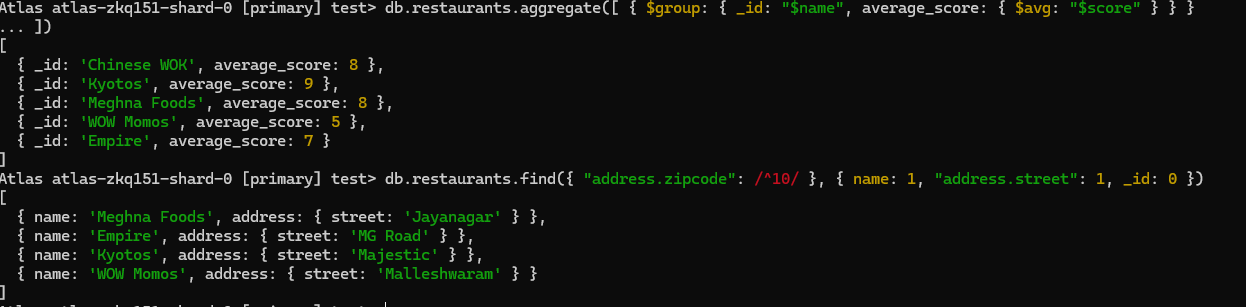
**3.Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns.**



**4.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.**



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



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

