VISVESVARAYATECHNOLOGICALUNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Database Management Systems (23CS3PCDBM)

Submitted by

KAVANA M A (1BM23CS145)

in partial fulfilment for the award of the degree of

BACHELOROFENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
Sep-2024 to Jan-2025

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 **Kavana M A (1BM23CS145),** 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 Visvesvaraya 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.

Dr. Sheetal V A Associate Professor Department of CSE, BMSCE	Dr. Joythi S Nayak Professor HOD Department of CSE, BMSCE

`

Index

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.

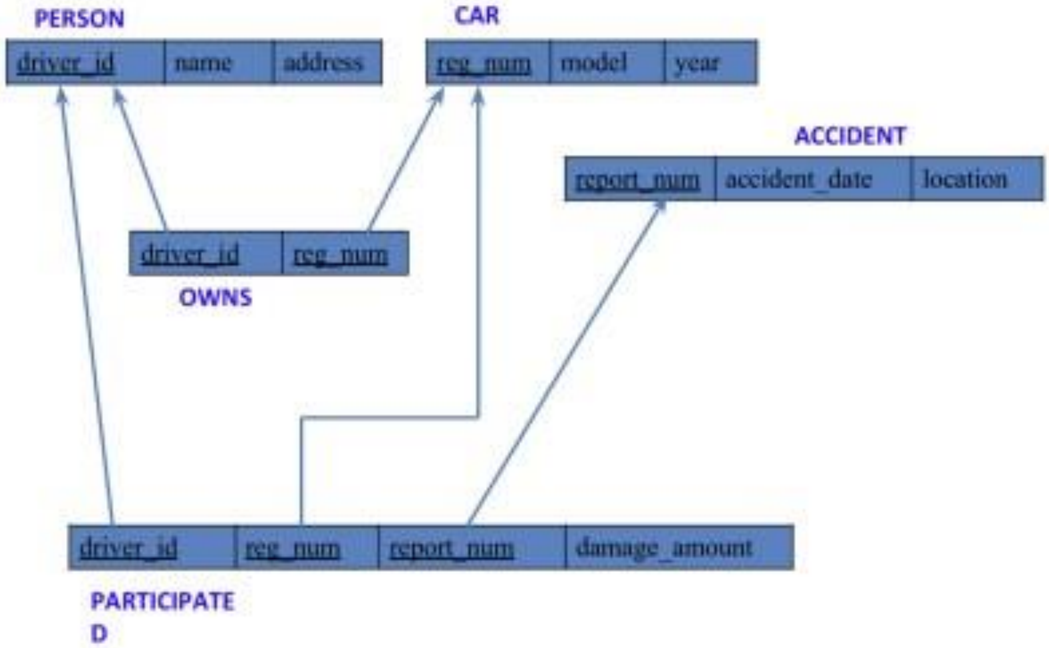
- Add a new accident to the database.

- 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_1BMS23CS145;
```

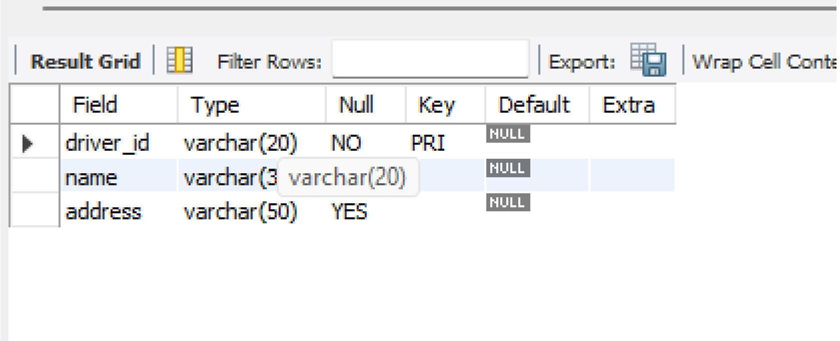
Create table

```
create table person_1BMS23CS145(
driver_id varchar(20),
name varchar(30),
address varchar(50),
PRIMARY KEY(driver_id)
);
create table car_1BMS23CS145(
reg_num varchar(15),
model varchar(10),
year int,
PRIMARY KEY(reg_num)
);
create table owns_1BMS23CS145(
driver_id varchar(20),
reg_num varchar(15),
PRIMARY KEY(driver_id,reg_num),
FOREIGN KEY(driver_id) REFERENCES person_1BMS23CS145(driver_id),
FOREIGN KEY(reg_num) REFERENCES car_1BMS23CS145(reg_num)
);
```

```
create table accident_1BMS23CS145(
report_num int,
accident_date date,
location varchar(50),
PRIMARY KEY(report_num)
);
create table participated_1BMS23CS145(
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_1BMS23CS145(driver_id),
FOREIGN KEY(reg_num) REFERENCES car_1BMS23CS145(reg_num),
FOREIGN KEY(report_num) REFERENCES accident_1BMS23CS145(report_num)
);
```

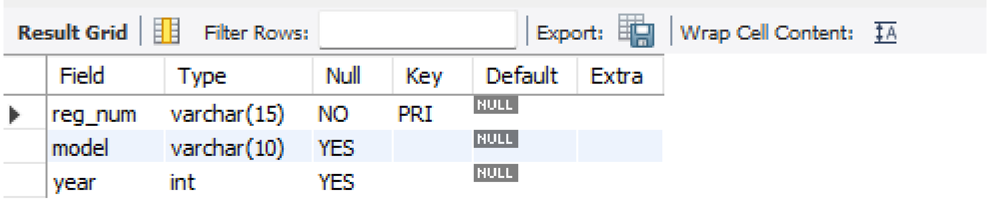
Structure of the table

desc person_1BMS23CS145;



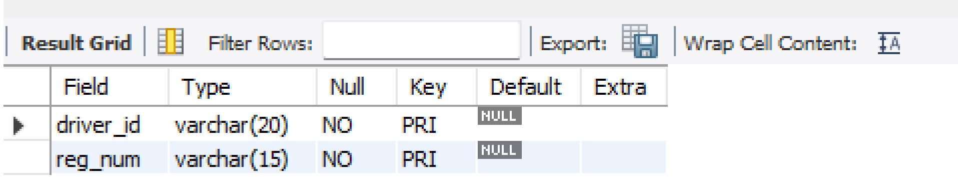
Field	Type	Null	Key	Default	Extra
driver_id	varchar(20)	NO	PRI	NULL	
name	varchar(3 varchar(20)			NULL	
address	varchar(50)	YES		NULL	

desc car_1BMS23CS145;



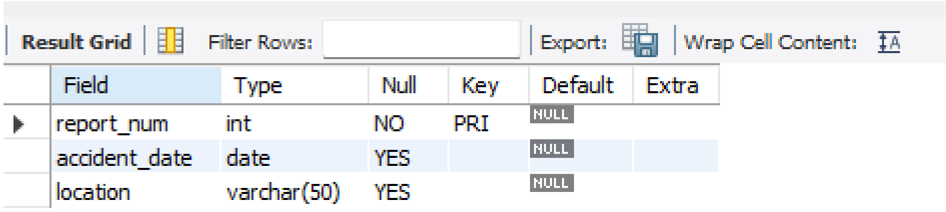
Field	Type	Null	Key	Default	Extra
reg_num	varchar(15)	NO	PRI	NULL	
model	varchar(10)	YES		NULL	
year	int	YES		NULL	

desc owns_1BMS23CS145;



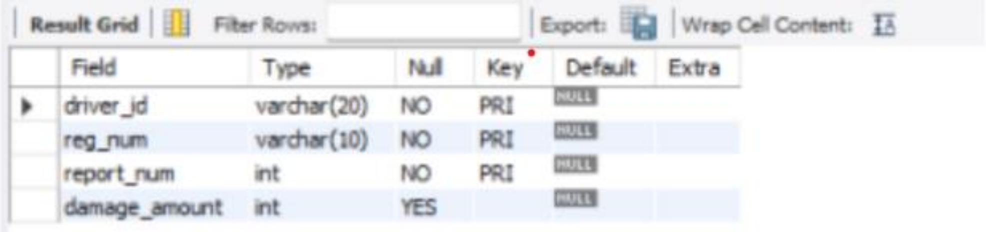
Field	Type	Null	Key	Default	Extra
driver_id	varchar(20)	NO	PRI	NULL	
reg_num	varchar(15)	NO	PRI	NULL	

desc accident_1BMS23CS145;



Field	Type	Null	Key	Default	Extra
report_num	int	NO	PRI	NULL	
accident_date	date	YES		NULL	
location	varchar(50)	YES		NULL	

desc participated_1BMS23CS145;



Field	Type	Null	Key	Default	Extra
driver_id	varchar(20)	NO	PRI	NULL	
reg_num	varchar(10)	NO	PRI	NULL	
report_num	int	NO	PRI	NULL	
damage_amount	int	YES		NULL	

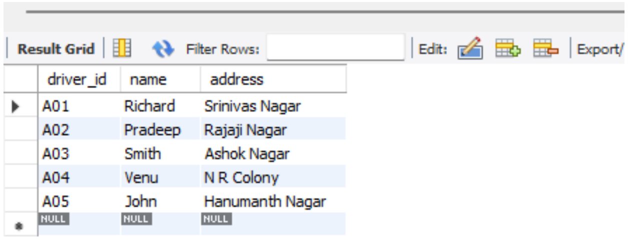
Inserting Values into the table

```
insert into
person_1BMS23CS145
values("A01","Richard",
"Srinivas Nagar");
insert into
person_1BMS23CS145
values("A02","Pradeep",
"Rajaji Nagar");
insert into
person_1BMS23CS145
values("A03","Smith", "Ashok
Nagar");
insert into
person_1BMS23CS145
values("A04","Venu", "N R
Colony");
insert into
person_1BMS23CS145
values("A05","John",
"Hanumanth Nagar");

insert into car_1BMS23CS145
values("KA052250","Indica",
"1990");
insert into car_1BMS23CS145
values("KA031181","Lancer",
"1957");
insert into car_1BMS23CS145
values("KA095477","Toyota",
"1998");
insert into car_1BMS23CS145
values("KA053408","Honda",
"2008");
insert into car_1BMS23CS145
values("KA041702","Audi",
"2005");

insert into
owns_1BMS23CS145
values("A01","KA052250");
insert into
owns_1BMS23CS145
values("A02","KA031181");
```

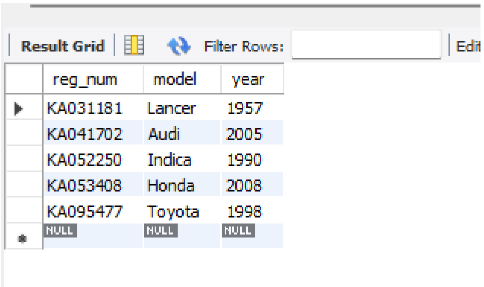
```
insert into
owns_1BMS23CS145
values("A03","KA095477");
insert into
owns_1BMS23CS145
values("A04","KA053408");
insert into
owns_1BMS23CS145
values("A05","KA041702");

insert into
accident_1BMS23CS145
values(11,"2003-01-
01","Mysore Road");
insert into
accident_1BMS23CS145
values(12,"2004-02-02","South
end circle");
insert into
accident_1BMS23CS145
values(13,"2003-01-21","Bull
temple Road");
insert into
accident_1BMS23CS145
values(14,"2008-02-
17","Mysore Road");
insert into
accident_1BMS23CS145
values(15,"2004-03-
05","Kanakpura Road");

insert into
participated_1BMS23CS145
values ("a01", "ka052250", 11,
10000);
insert into
participated_1BMS23CS145
values ("a02", "ka053408", 12,
50000);
insert into
participated_1BMS23CS145
values ("a03", "ka095477", 13,
25000);
insert into
participated_1BMS23CS145
```

```
values ("a04", "ka031181", 14,
3000);
insert into
participated_1BMS23CS145
values ("a05", "ka041702", 15,
5000);
```

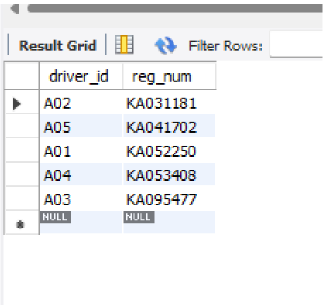
```
select * from person_1BMS23CS145;
select * from car_1BMS23CS145;
select * from owns_1BMS23CS145;
select * from accident_1BMS23CS145;
select * from participated_1BMS23CS145;
```



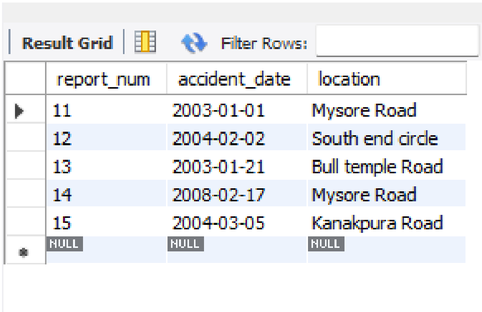
driver_id	name	address
A01	Richard	Srinivas Nagar
A02	Pradeep	Rajaji Nagar
A03	Smith	Ashok Nagar
A04	Venu	N R Colony
A05	John	Hanumanth Nagar
NULL	NULL	NULL



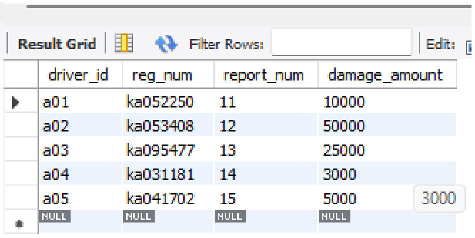
reg_num	model	year
KA031181	Lancer	1957
KA041702	Audi	2005
KA052250	Indica	1990
KA053408	Honda	2008
KA095477	Toyota	1998
NULL	NULL	NULL



driver_id	reg_num
A02	KA031181
A05	KA041702
A01	KA052250
A04	KA053408
A03	KA095477
NULL	NULL



report_num	accident_date	location
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	Kanakpura Road
NULL	NULL	NULL



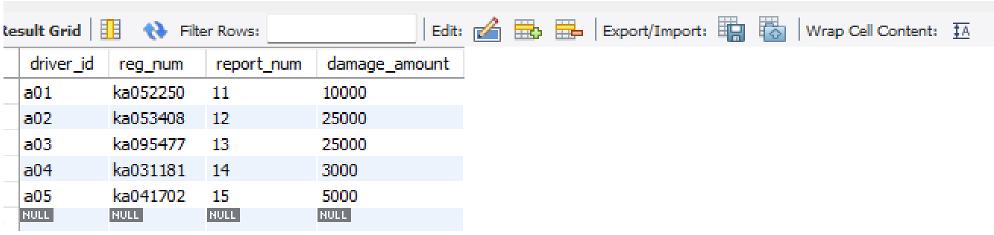
driver_id	reg_num	report_num	damage_amount
a01	ka052250	11	10000
a02	ka053408	12	50000
a03	ka095477	13	25000
a04	ka031181	14	3000
a05	ka041702	15	5000
NULL	NULL	NULL	NULL

3000

Queries

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

```
update participated_1BMS23CS145
set damage_amount=25000
where reg_num='KA053408' and report_num=12;
```



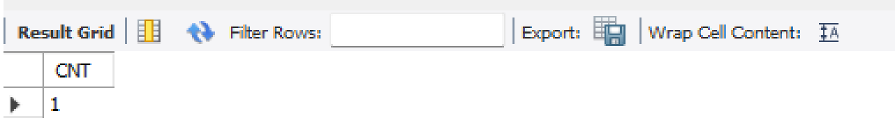
driver_id	reg_num	report_num	damage_amount
a01	ka052250	11	10000
a02	ka053408	12	25000
a03	ka095477	13	25000
a04	ka031181	14	3000
a05	ka041702	15	5000
NULL	NULL	NULL	NULL

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

select count(distinct driver_id) CNT

from participated_1BMS23CS145 a, accident_1BMS23CS145 b

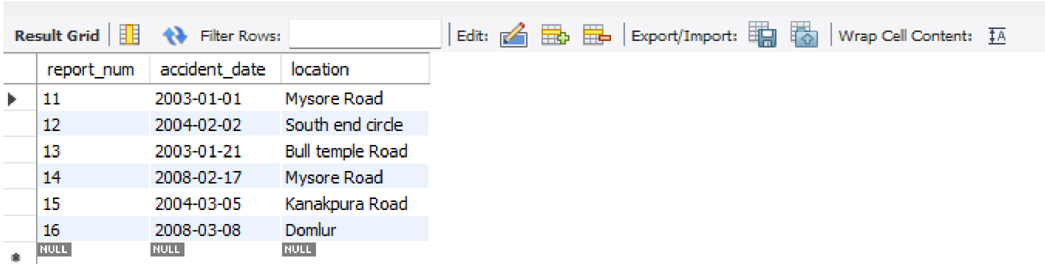
where a.report_num=b.report_num and b.accident_date like '2008%';



CNT
1

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

insert into accident_1BMS23CS145 values(16,'2008-03-08',"Domlur");
select * from accident_1BMS23CS145;



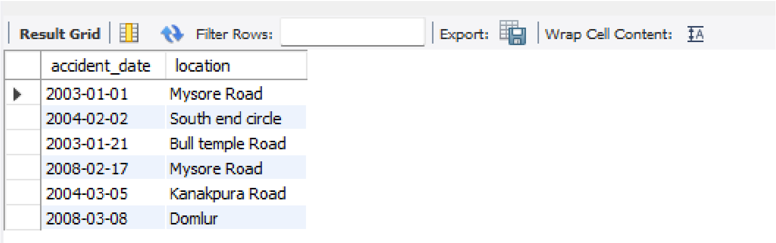
report_num	accident_date	location
11	2003-01-01	Mysore Road
12	2004-02-02	South end circle
13	2003-01-21	Bull temple Road
14	2008-02-17	Mysore Road
15	2004-03-05	Kanakpura Road
16	2008-03-08	Domlur
NULL	NULL	NULL

TO DO:

- **DISPLAY ACCIDENT DATE AND LOCATION**

SELECT accident_date, location

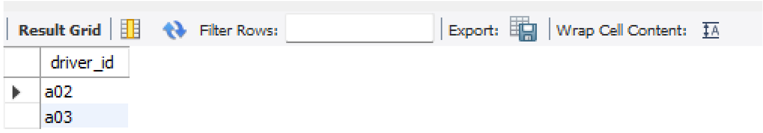
FROM accident_1BMS23CS145;



accident_date	location
2003-01-01	Mysore Road
2004-02-02	South end circle
2003-01-21	Bull temple Road
2008-02-17	Mysore Road
2004-03-05	Kanakpura Road
2008-03-08	Domlur

- **DISPLAY DRIVER ID WHO DID ACCIDENT WITH DAMAGE AMOUNT GREATER THAN OR EQUAL TO RS.25000**

```
SELECT DISTINCT a.driver_id
FROM participated_1BMS23CS145 a
JOIN accident_1BMS23CS145 b ON a.report_num = b.report_num
WHERE a.damage_amount >= 25000;
```



driver_id
a02
a03

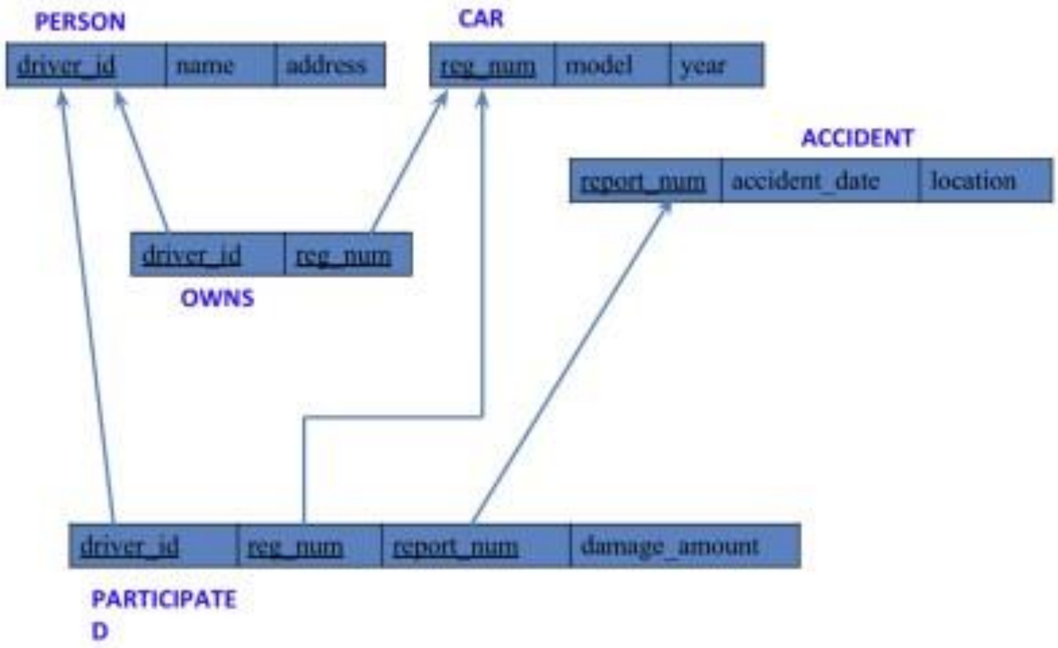
More Queries on Insurance Database

Question

(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.

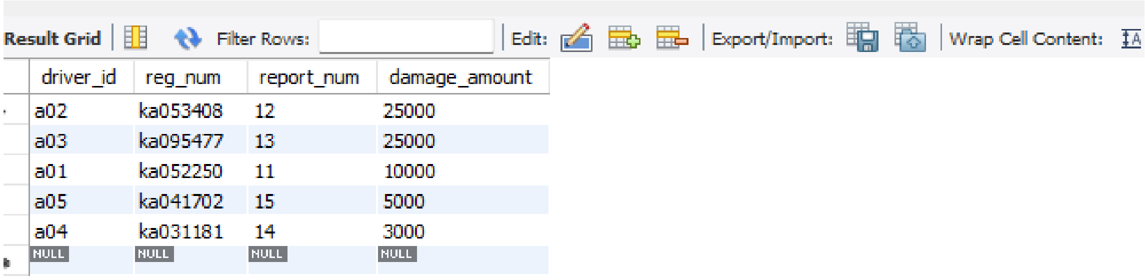
Schema Diagram



Queries

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

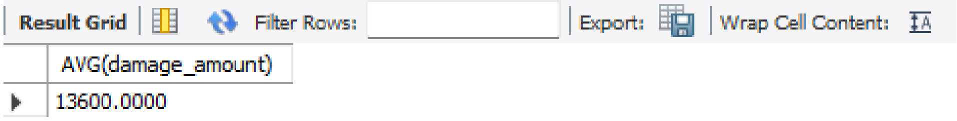
```
SELECT * FROM participated_1BMS23CS145

ORDER BY damage_amount DESC;
```



driver_id	reg_num	report_num	damage_amount
a02	ka053408	12	25000
a03	ka095477	13	25000
a01	ka052250	11	10000
a05	ka041702	15	5000
a04	ka031181	14	3000
NULL	NULL	NULL	NULL

- **Find the average damage amount**

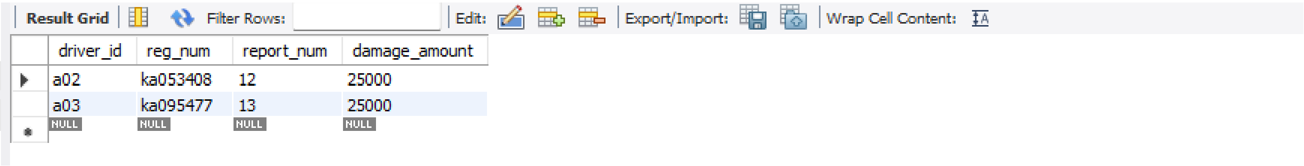
```
SELECT AVG(damage_amount) FROM participated_1BMS23CS145;
```



AVG(damage_amount)
13600.0000

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

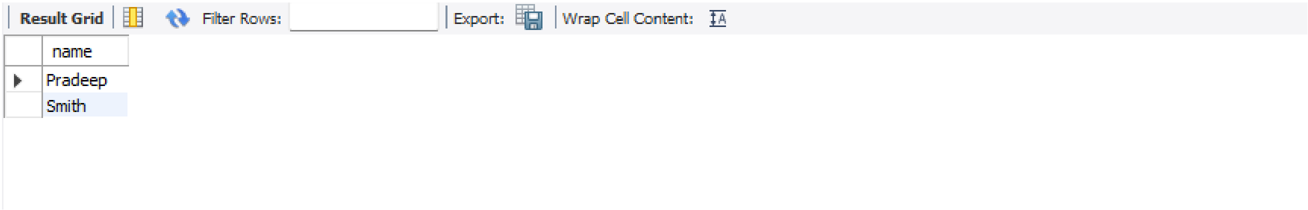
```
DELETE FROM participated_1BMS23CS145
WHERE damage_amount < (
   SELECT avg_damage
   FROM (SELECT AVG(damage_amount) AS avg_damage FROM participated_1BMS23CS145)
AS avg_table
);
```



driver_id	reg_num	report_num	damage_amount
a02	ka053408	12	25000
a03	ka095477	13	25000
NULL	NULL	NULL	NULL

- **LIST THE NAME OF DRIVERS WHOSE DAMAGE IS GREATER THAN THE AVERAGE DAMAGE AMOUNT.**

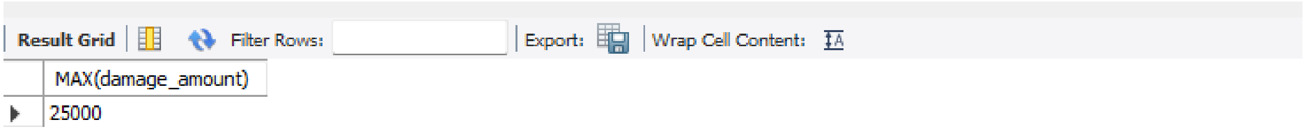
```
SELECT p.name
FROM person_1BMS23CS145 p
JOIN participated_1BMS23CS145 part ON p.driver_id = part.driver_id
WHERE part.damage_amount >= (SELECT AVG(damage_amount) FROM
participated_1BMS23CS145);
```



```
name
Pradeep
Smith
```

- **Find maximum damage amount.**

```
SELECT MAX(damage_amount) FROM participated_1BMS23CS145;
```



```
MAX(damage_amount)
25000
```

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)

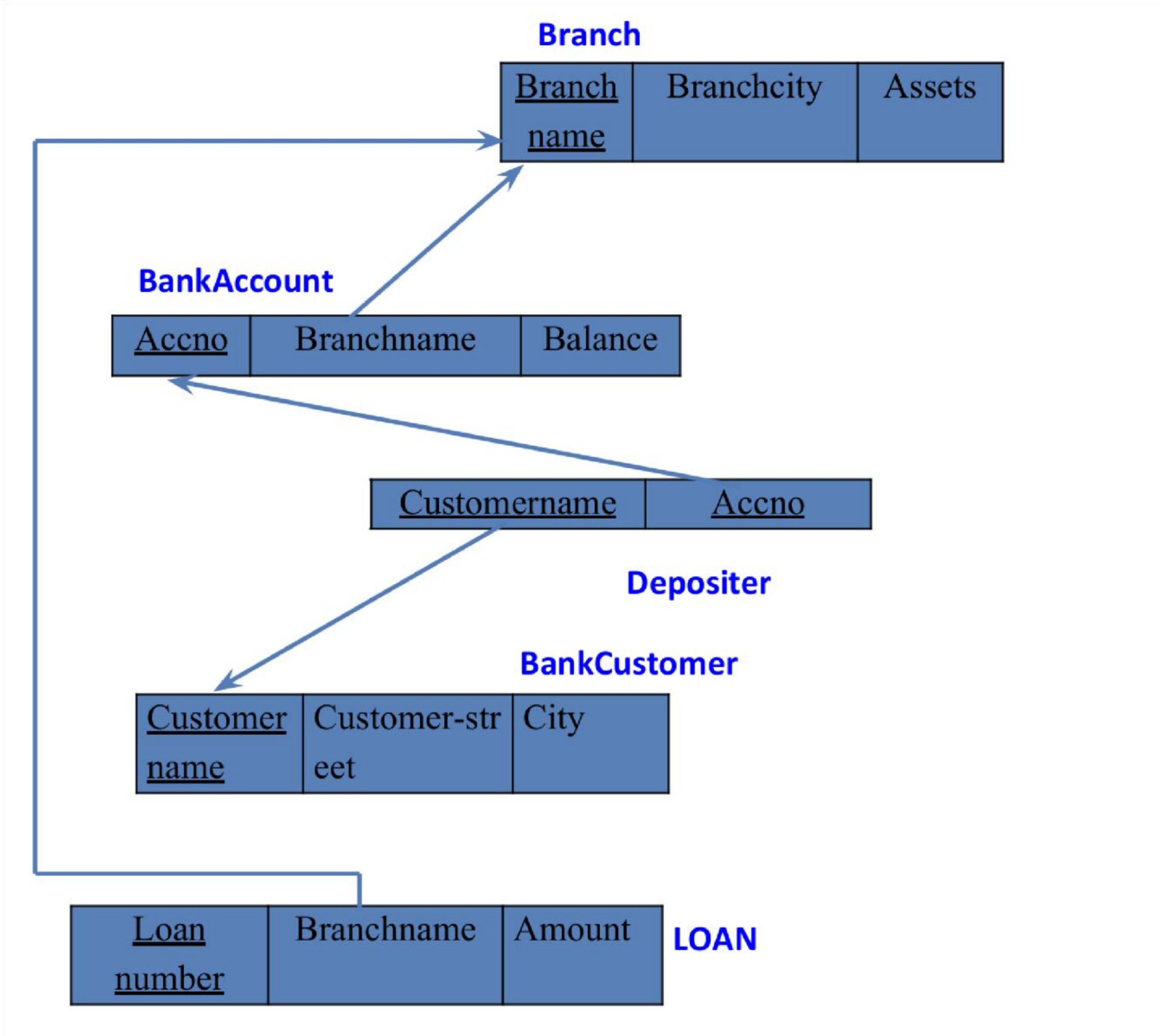
- LOAN (loan-number: int, branch-name: String, amount: real)

- 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 view which gives each branch the sum of the amount of all the loans at the branch.

Schema Diagram



Create database

```
create database bank_1bm23cs145;
use bank_1bm23cs145;
```

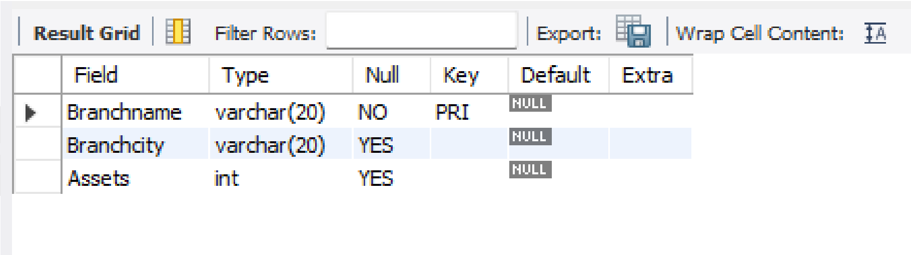
Create table

```
create table Branch_1bm23cs145(
Branchname varchar(20),
Branchcity varchar(20),
Assets int,
PRIMARY KEY(Branchname)
);
```

```
create table BankAccount_1bm23cs145(
Accno int,
Branchname varchar(20),
Balance int,
PRIMARY KEY(Accno),
foreign key(Branchname) references Branch_1bm23cs145(Branchname)
);

create table BankCustomer_1bm23cs145(
Customername varchar(20),
Customerstreet varchar(20),
City varchar(20),
PRIMARY KEY(Customername)
);

create table Depositer_1bm23cs145(
Customername varchar(20),
Accno int,
PRIMARY KEY(Customername,Accno),
foreign key(Customername) references BankCustomer_1bm23cs145(Customername),
foreign key(Accno) references BankAccount_1bm23cs145(Accno)
);

create table Loan_1bm23cs145(
Loannumber int,
Branchname varchar(20),
Amount int,
PRIMARY KEY(Loannumber),
foreign key(Branchname) references Branch_1bm23cs145(Branchname)
);
```

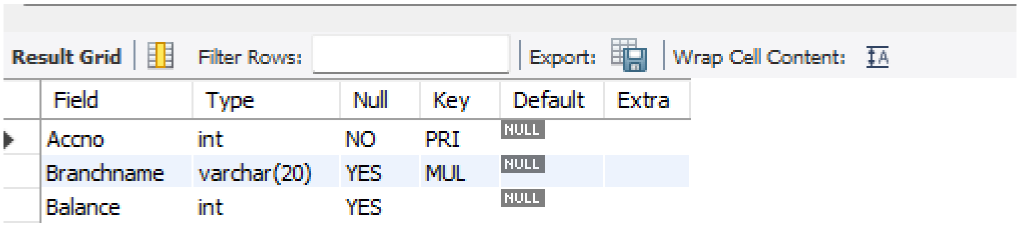
Structure of the table

desc Branch_1bm23cs145 ;



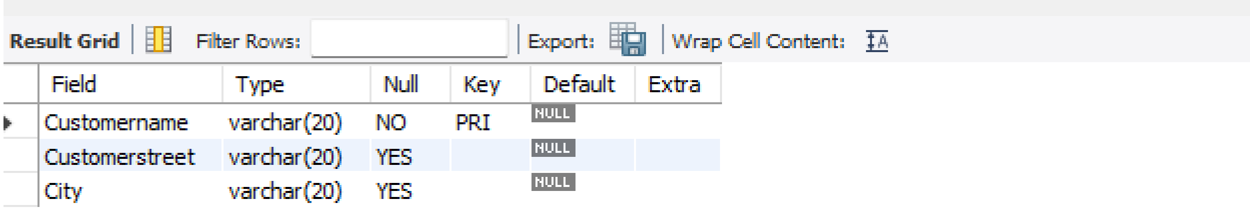
Field	Type	Null	Key	Default	Extra
Branchname	varchar(20)	NO	PRI	NULL	
Branchcity	varchar(20)	YES		NULL	
Assets	int	YES		NULL	

desc BankAccount_1bm23cs145 ;



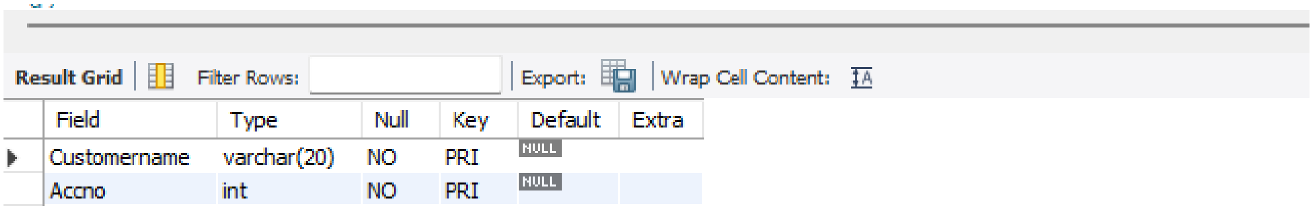
Field	Type	Null	Key	Default	Extra
Accno	int	NO	PRI	NULL	
Branchname	varchar(20)	YES	MUL	NULL	
Balance	int	YES		NULL	

desc BankCustomer_1bm23cs145 ;



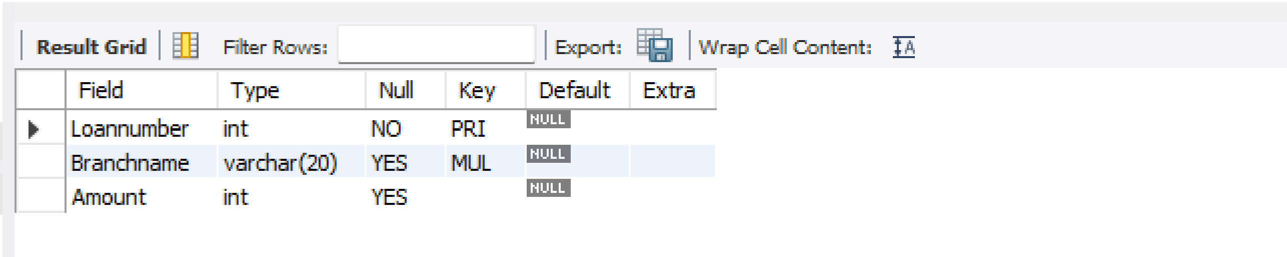
Field	Type	Null	Key	Default	Extra
Customername	varchar(20)	NO	PRI	NULL	
Customerstreet	varchar(20)	YES		NULL	
City	varchar(20)	YES		NULL	

desc Depositer_1bm23cs145 ;



Field	Type	Null	Key	Default	Extra
Customername	varchar(20)	NO	PRI	NULL	
Accno	int	NO	PRI	NULL	

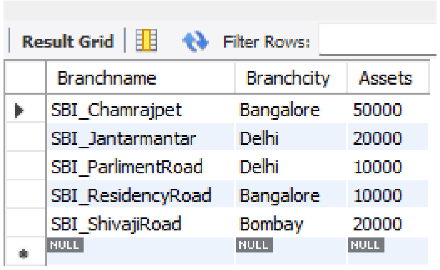
desc Loan_1bm23cs145 ;



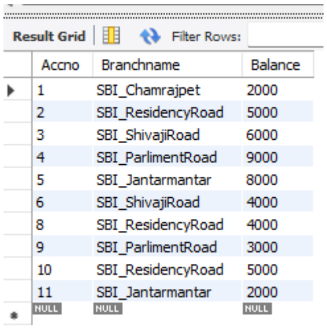
Field	Type	Null	Key	Default	Extra
Loannumber	int	NO	PRI	NULL	
Branchname	varchar(20)	YES	MUL	NULL	
Amount	int	YES		NULL	

Inserting Values to the table

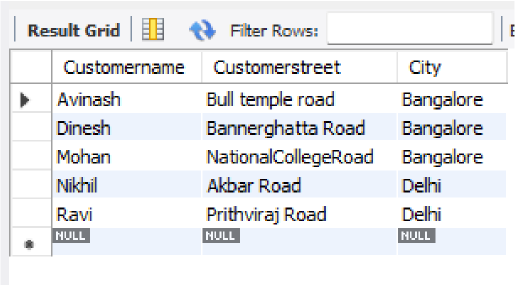
```
insert into Branch_1bm23cs145 values("SBI_Chamrajpet", "Bangalore", 50000);
insert into Branch_1bm23cs145 values("SBI_ResidencyRoad", "Bangalore", 10000);
insert into Branch_1bm23cs145 values("SBI_ShivajiRoad", "Bombay", 20000);
insert into Branch_1bm23cs145 values("SBI_ParlimentRoad", "Delhi", 10000);
insert into Branch_1bm23cs145 values("SBI_Jantarmantar", "Delhi", 20000);

insert into BankAccount_1bm23cs145 values(1, "SBI_Chamrajpet",2000 );
insert into BankAccount_1bm23cs145 values(2, "SBI_ResidencyRoad", 5000);
insert into BankAccount_1bm23cs145 values(3, "SBI_ShivajiRoad", 6000);
insert into BankAccount_1bm23cs145 values(4, "SBI_ParlimentRoad", 9000);
insert into BankAccount_1bm23cs145 values(5, "SBI_Jantarmantar", 8000);
insert into BankAccount_1bm23cs145 values(6, "SBI_ShivajiRoad", 4000);
insert into BankAccount_1bm23cs145 values(8, "SBI_ResidencyRoad", 4000);
insert into BankAccount_1bm23cs145 values(9, "SBI_ParlimentRoad", 3000);
insert into BankAccount_1bm23cs145 values(10, "SBI_ResidencyRoad", 5000);
insert into BankAccount_1bm23cs145 values(11, "SBI_Jantarmantar", 2000);

insert into BankCustomer_1bm23cs145 values("Avinash", "Bull temple road","Bangalore" );
insert into BankCustomer_1bm23cs145 values("Dinesh", "Bannerghatta Road","Bangalore" );
insert into BankCustomer_1bm23cs145 values("Mohan", "NationalCollegeRoad","Bangalore" );
insert into BankCustomer_1bm23cs145 values("Nikhil", "Akbar Road","Delhi" );
insert into BankCustomer_1bm23cs145 values("Ravi", "Prithviraj Road","Delhi" );

insert into Depositer_1bm23cs145 values("Avinash", 1);
insert into Depositer_1bm23cs145 values("Dinesh", 2);
insert into Depositer_1bm23cs145 values("Nikhil", 4);
insert into Depositer_1bm23cs145 values("Ravi", 5);
insert into Depositer_1bm23cs145 values("Avinash", 8);

insert into Depositer_1bm23cs145 values("Nikhil", 9);

insert into Depositer_1bm23cs145 values("Dinesh", 10);

insert into Depositer_1bm23cs145 values("Nikhil", 11);


insert into Loan_1bm23cs145 values(1, "SBI_Chamrajpet", 1000);

insert into Loan_1bm23cs145 values(2, "SBI_ResidencyRoad", 2000);

insert into Loan_1bm23cs145 values(3, "SBI_ShivajiRoad", 3000);

insert into Loan_1bm23cs145 values(4, "SBI_ParlimentRoad", 4000);

insert into Loan_1bm23cs145 values(5, "SBI_Jantarmantar", 5000);

select * from Branch_1bm23cs145 ;

select * from BankAccount_1bm23cs145 ;

select * from BankCustomer_1bm23cs145 ;

select * from Depositer_1bm23cs145 ;

select * from Loan_1bm23cs145 ;
```



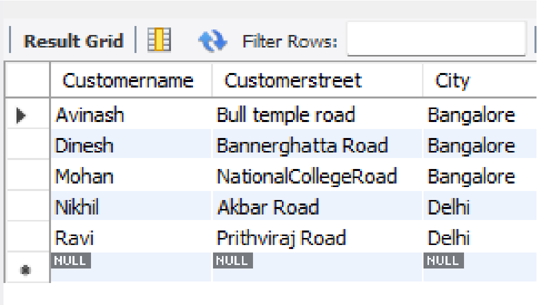
Branchname	Branchcity	Assets
SBI_Chamrajpet	Bangalore	50000
SBI_Jantarmantar	Delhi	20000
SBI_ParlimentRoad	Delhi	10000
SBI_ResidencyRoad	Bangalore	10000
SBI_ShivajiRoad	Bombay	20000
NULL	NULL	NULL



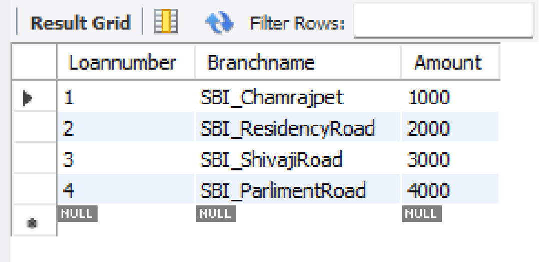
Accno	Branchname	Balance
1	SBI_Chamrajpet	2000
2	SBI_ResidencyRoad	5000
3	SBI_ShivajiRoad	6000
4	SBI_ParlimentRoad	9000
5	SBI_Jantarmantar	8000
6	SBI_ShivajiRoad	4000
8	SBI_ResidencyRoad	4000
9	SBI_ParlimentRoad	3000
10	SBI_ResidencyRoad	5000
11	SBI_Jantarmantar	2000
NULL	NULL	NULL



Customername	Customerstreet	City
Avinash	Bull temple road	Bangalore
Dinesh	Bannerghatta Road	Bangalore
Mohan	NationalCollegeRoad	Bangalore
Nikhil	Akbar Road	Delhi
Ravi	Prithviraj Road	Delhi
NULL	NULL	NULL



Customername	Customerstreet	City
Avinash	Bull temple road	Bangalore
Dinesh	Bannerghatta Road	Bangalore
Mohan	NationalCollegeRoad	Bangalore
Nikhil	Akbar Road	Delhi
Ravi	Prithviraj Road	Delhi
NULL	NULL	NULL

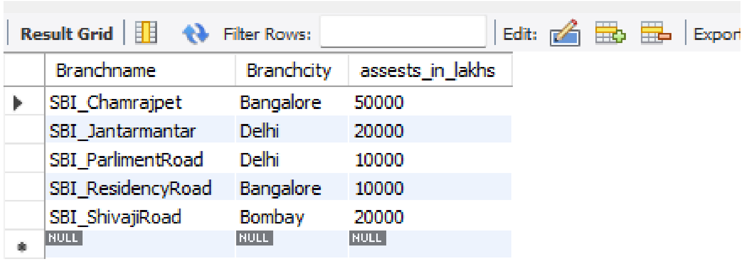


Loannumber	Branchname	Amount
1	SBI_Chamrajpet	1000
2	SBI_ResidencyRoad	2000
3	SBI_ShivajiRoad	3000
4	SBI_ParlimentRoad	4000
NULL	NULL	NULL

Queries

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

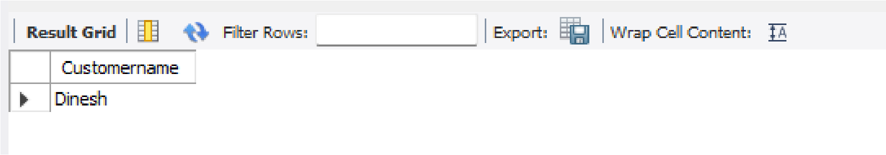
```
alter table Branch_1bm23cs145
change column assets assests_in_lakhs real;
```



Branchname	Branchcity	assests_in_lakhs
SBI_Chamrajpet	Bangalore	50000
SBI_Jantarmantar	Delhi	20000
SBI_ParlimentRoad	Delhi	10000
SBI_ResidencyRoad	Bangalore	10000
SBI_ShivajiRoad	Bombay	20000
NULL	NULL	NULL

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

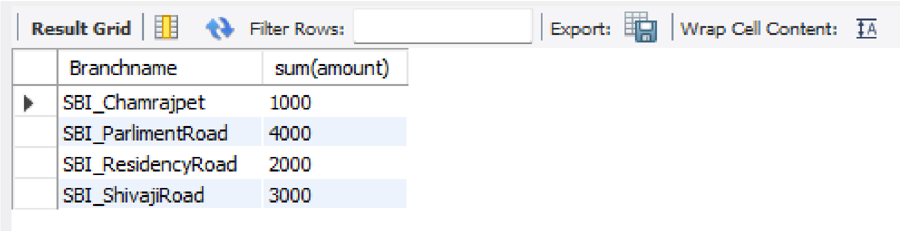
```
select d.Customername
from Depositer_1bm23cs145 d,BankAccount_1bm23cs145 b
where b.Branchname="SBI_ResidencyRoad"
AND d.accno=b.accno GROUP BY d.Customername
having count(d.accno) >=2;
```



Customername
Dinesh

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

```
create view br
as select Branchname , sum(amount)
  from loan_1bm23cs145
  group by Branchname;
select * from br;
```



Branchname	sum(amount)
SBI_Chamrajpet	1000
SBI_ParlimentRoad	4000
SBI_ResidencyRoad	2000
SBI_ShivajiRoad	3000

More Queries on 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) - Depositer(customer-name: String, accno: int)

- LOAN (loan-number: int, branch-name: String, amount: real) - Find all the customers who have

  an account at all the branches

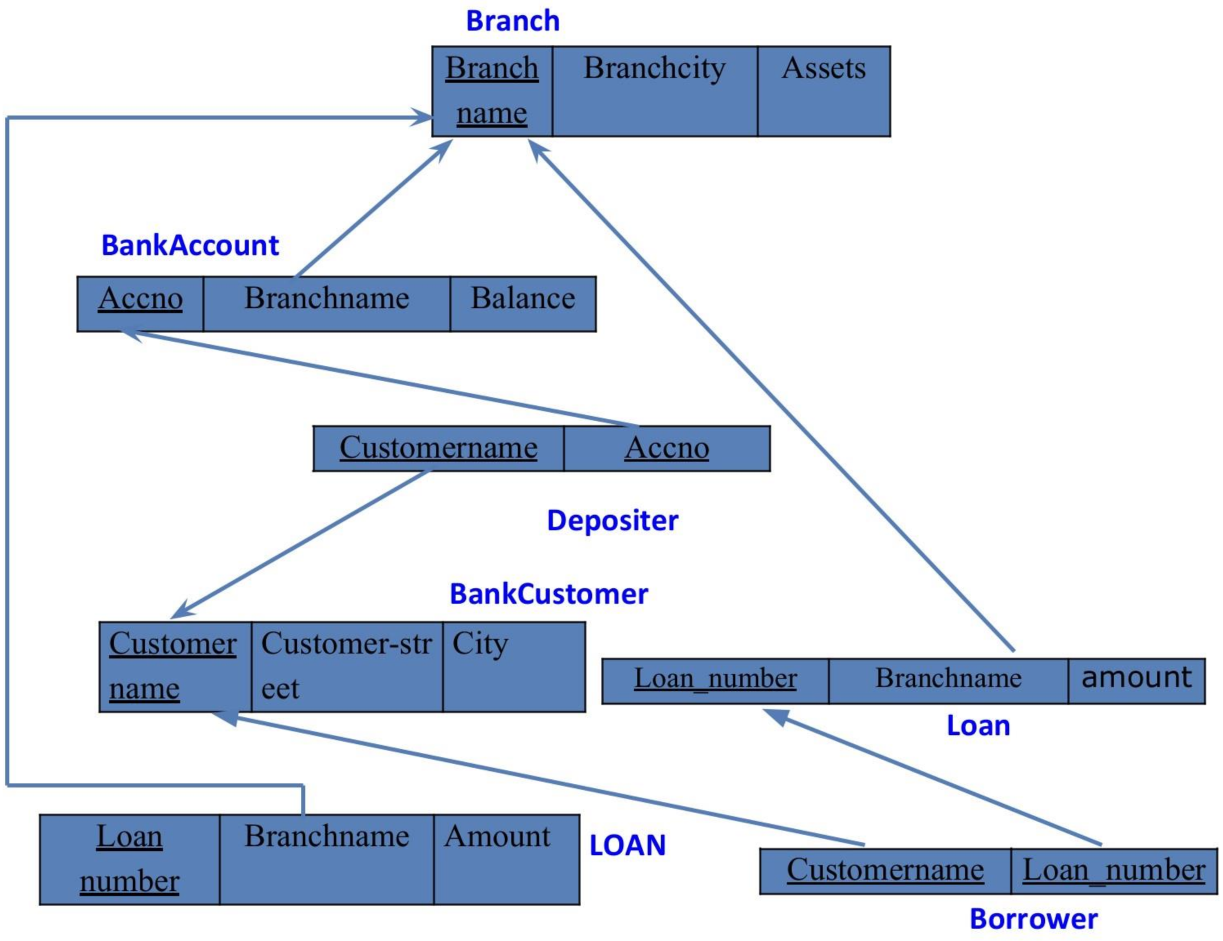
- located in a specific city (Ex. Delhi).

- Find all customers who have a loan at the bank but do not have an account. - Find all customers who have both an account and a loan at the Bangalore branch
- Find the names of all branches that have greater assets than all branches located in Bangalore.

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

- Update the Balance of all accounts by 5%

Schema Diagram



Creating Table:

```
create table Borrower_1bm23cs145(
Customername varchar(20),
LoanNumber int,
PRIMARY KEY(Customername,LoanNumber),
foreign key(Customername) references BankCustomer_1bm23cs145(Customername),
foreign key(LoanNumber) references Loan_1bm23cs145(LoanNumber)
);
```

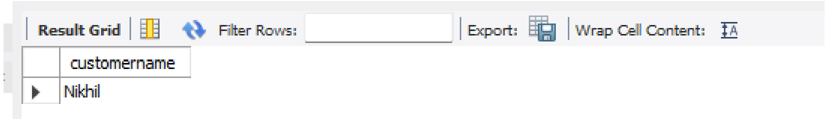
Inserting values:

```
insert into Branch_1bm23cs145 values("SBI_MantriMarg", "Delhi", 200000);
insert into BankAccount_1bm23cs145 values(12, "SBI_MantriMarg", 2000);
insert into Depositer_1bm23cs145 values("Nikhil", 12);

insert into Borrower_1bm23cs145 values("Avinash",1);
insert into Borrower_1bm23cs145 values("Dinesh",2);
insert into Borrower_1bm23cs145 values("Mohan",3);
insert into Borrower_1bm23cs145 values("Nikhil", 4);
insert into Borrower_1bm23cs145 values("Ravi",5);
```

Queries

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

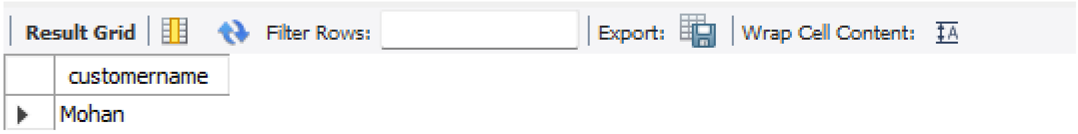
```
select distinct d.customername
from Depositer_1bm23cs145 d,BankAccount_1bm23cs145 ba,Branch_1bm23cs145 b
where d.accno=ba.accno and ba.branchname=b.branchname and b.branchcity="Delhi"
group by d.customername having count(b.branchname)>1;
```



customername
Nikhil

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

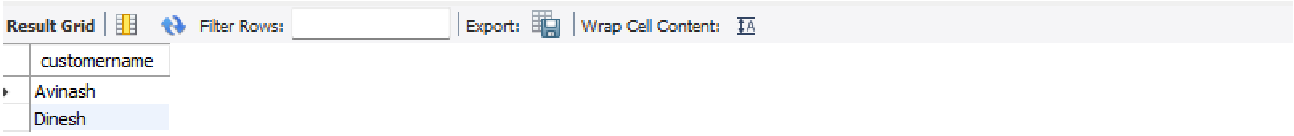
```
select b.customername
from borrower_1bm23cs145 b
where b.loannumber not in (select d.accno
                                        from depositer_1bm23cs145 d
                               where b.loannumber=d.accno);
```



customername
Mohan

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

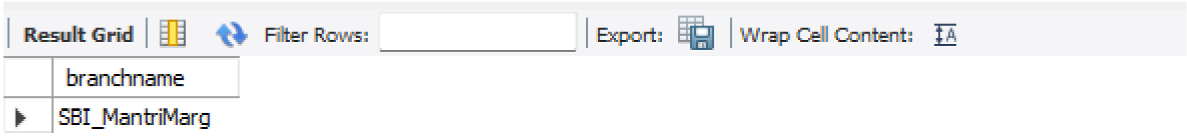
```
select b.customername
from Borrower_1bm23cs145 b
where b.loannumber in ( select d.accno
                         from depositer_1bm23cs145 d,BankAccount_1bm23cs145 ba, Branch_1bm23cs145 b
                           where b.loannumber=d.accno and d.accno=ba.accno a
                           ba.branchname=b.branchname and b.branchcity="Bangalore");
```



customername
Avinash
Dinesh

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

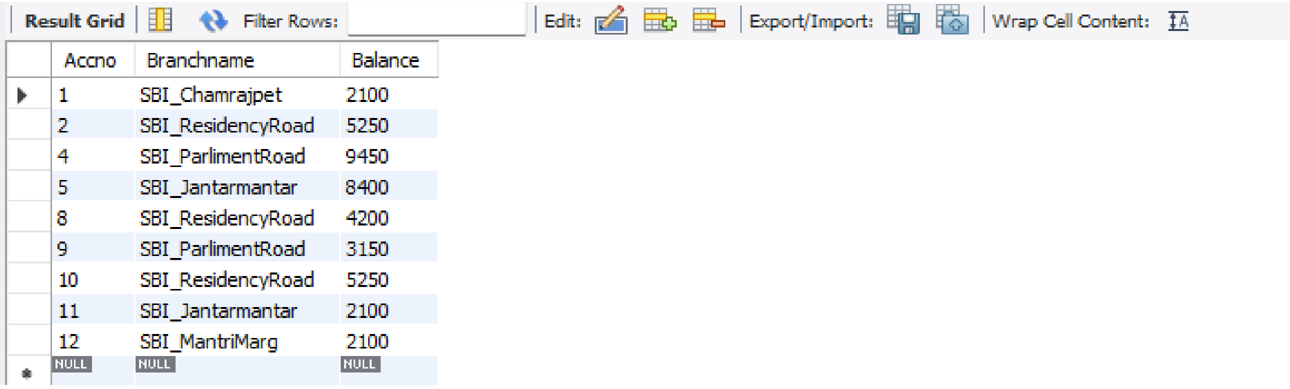
```
select branchname
from Branch_1bm23cs145
where assests_in_lakhs > all (select assests_in_lakhs
                        from Branch_1bm23cs145
                        where branchcity="Bangalore" );
```



branchname
SBI_MantriMarg

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

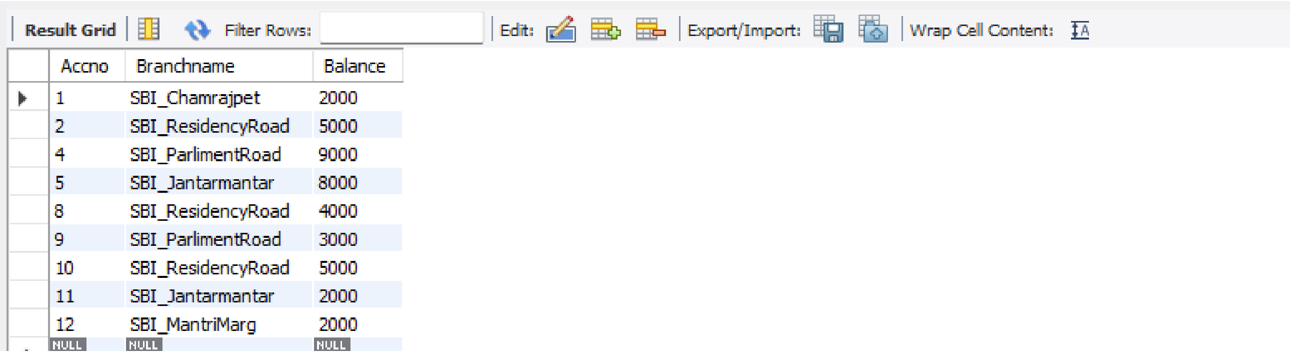
```
update BankAccount_1bm23cs145
set balance=balance+((5*balance)/100)
where accno in (1,2,4,5,8,9,10,11,12);
select * from BankAccount_1bm23cs145;
```



Accno	Branchname	Balance
1	SBI_Chamrajpet	2100
2	SBI_ResidencyRoad	5250
4	SBI_ParlimentRoad	9450
5	SBI_Jantarmantar	8400
8	SBI_ResidencyRoad	4200
9	SBI_ParlimentRoad	3150
10	SBI_ResidencyRoad	5250
11	SBI_Jantarmantar	2100
12	SBI_MantriMarg	2100
NULL	NULL	NULL

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

```
delete from BankAccount_1bm23cs145 ba
where ba.branchname=( select b.branchname
                  from branch_1bm23cs145 b
                  where b.branchcity="Bombay");
select * from BankAccount_1bm23cs145;
```



Accno	Branchname	Balance
1	SBI_Chamrajpet	2000
2	SBI_ResidencyRoad	5000
4	SBI_ParlimentRoad	9000
5	SBI_Jantarmantar	8000
8	SBI_ResidencyRoad	4000
9	SBI_ParlimentRoad	3000
10	SBI_ResidencyRoad	5000
11	SBI_Jantarmantar	2000
12	SBI_MantriMarg	2000
NULL	NULL	NULL

Employee Database

Question

(Week 5)

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

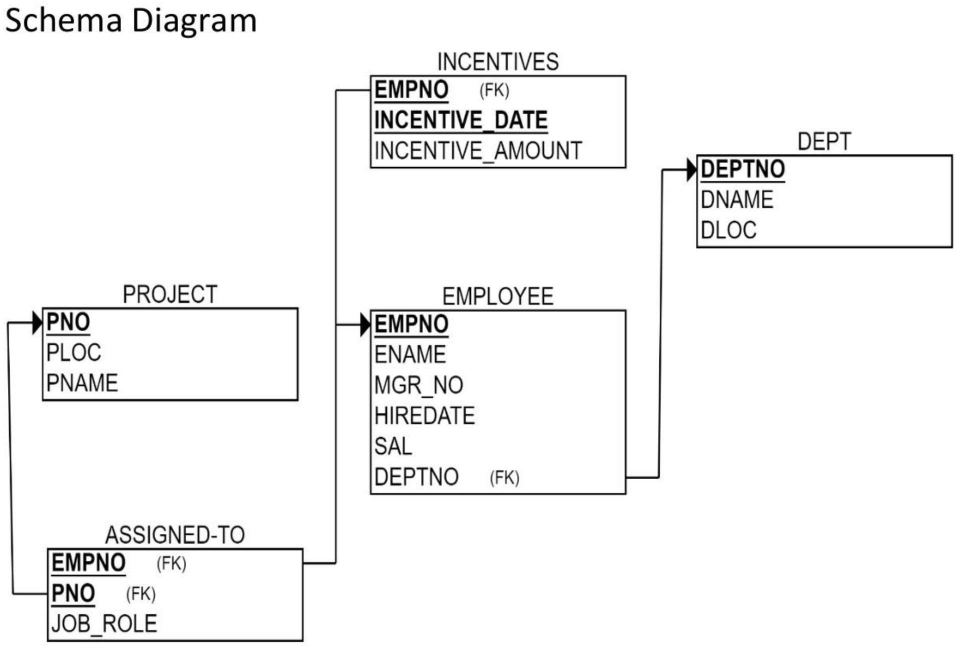
2. Enter greater than five tuples for each table.

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_Database_1bm23cs145;
use Employee_Database_1bm23cs145;
```

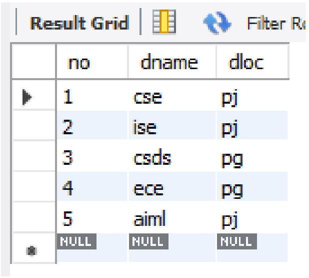
Create table

```
create table dept(
no varchar(20) primary key,
dname varchar(20),
dloc varchar(20));

create table employee(
empno int,
ename varchar(20),
mgr_no int,
hiredate varchar(20),
sal float,
no varchar(20),
primary key(empno,no),
foreign key(no) references dept(no)
);

create table incentives(
empno int,
date VARCHAR(20),
amt float,
primary key(empno,date),
foreign key(empno) references employee(empno)
);

create table project(
pno int primary key,
ploc VARCHAR(20),
pname varchar(20));

create table Assingnedto(
empno int,
pno int,
job_role text,
primary key(empno,pno),
foreign key(empno) references employee(empno),
foreign key(pno) references project(pno));
```

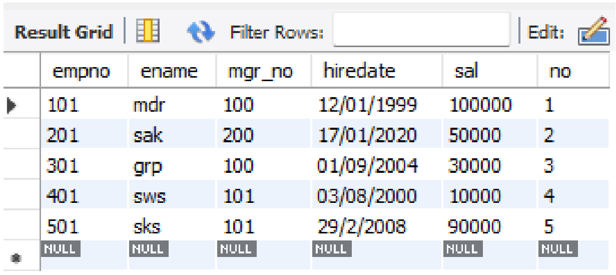
Inserting Values to the table

```
 insert into dept values(1,"cse","pj");
 insert into dept values(2,"ise","pj");
 insert into dept values(3,"csds","pg");
 insert into dept values(4,"ece","pg");
 insert into dept values(5,"aiml","pj");
```

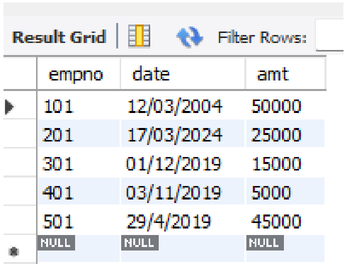
```
insert into employee values(101,"mdr",100,"12/01/1999",100000,1);
insert into employee values(201,"sak",200,"17/01/2020",50000,2);
insert into employee values(301,"grp",100,"01/09/2004",30000,3);
insert into employee values(401,"sws",101,"03/08/2000",10000,4);
insert into employee values(501,"sks",101,"29/2/2008",90000,5);

insert into incentives values(101,"12/03/2004",50000);
insert into incentives values(201,"17/03/2024",25000);
insert into incentives values(301,"01/12/2019",15000);
insert into incentives values(401,"03/11/2019",5000);
insert into incentives values(501,"29/4/2019",45000);

insert into project values(10,"bng","chatbot");
insert into project values(40,"delhi","ml model");
insert into project values(50,"bombay","blockchain");
insert into project values(30,"chennai","stocks");
insert into project values(80,"mysore","android app");

insert into Assingnedto values(101,10,"devops");
insert into Assingnedto values(201,40,"sde");
insert into Assingnedto values(301,50,"manager");
insert into Assingnedto values(401,30,"jpa");
insert into Assingnedto values(501,80,"pa");

select * from dept;
select * from employee;
select * from incentives;
select * from project;
select * from Assingnedto;
```



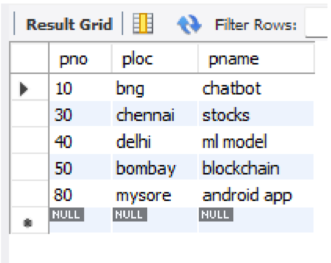
no	dname	dloc
1	cse	pj
2	ise	pj
3	csds	pg
4	ece	pg
5	aiml	pj
NULL	NULL	NULL



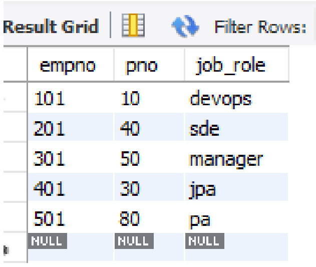
empno	ename	mgr_no	hiredate	sal	no
101	mdr	100	12/01/1999	100000	1
201	sak	200	17/01/2020	50000	2
301	grp	100	01/09/2004	30000	3
401	sws	101	03/08/2000	10000	4
501	sks	101	29/2/2008	90000	5
NULL	NULL	NULL	NULL	NULL	NULL



empno	date	amt
101	12/03/2004	50000
201	17/03/2024	25000
301	01/12/2019	15000
401	03/11/2019	5000
501	29/4/2019	45000
NULL	NULL	NULL



pno	ploc	pname
10	bng	chatbot
30	chennai	stocks
40	delhi	ml model
50	bombay	blockchain
80	mysore	android app
NULL	NULL	NULL

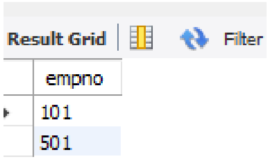


empno	pno	job_role
101	10	devops
201	40	sde
301	50	manager
401	30	jpa
501	80	pa
NULL	NULL	NULL

Queries

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

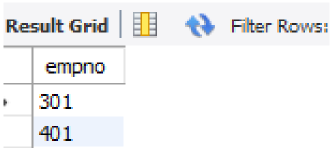
```
select a.empno
from assingnedto a
where a.pno=any(select pno
                    from project
                    where ploc in ('bng','mysore','hyderabad'));
```



empno
101
501

- **Get Employee ID's of those employees who didn't receive incentives**

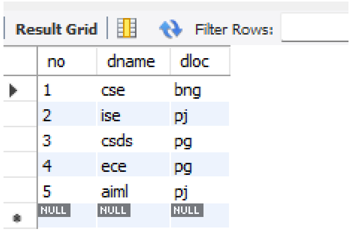
```
select e.empno
from employee e
where e.empno != all(select i.empno from incentives i);
```



empno
301
401

- **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.no,d.dname,d.dloc
from employee e, dept d,assingnedto a,project p
where e.no=d.no
           and e.empno=a.empno
    and a.pno=p.pno
    and d.dloc=p.ploc;
```



no	dname	dloc
1	cse	bng
2	ise	pj
3	csds	pg
4	ece	pg
5	aiml	pj
NULL	NULL	NULL

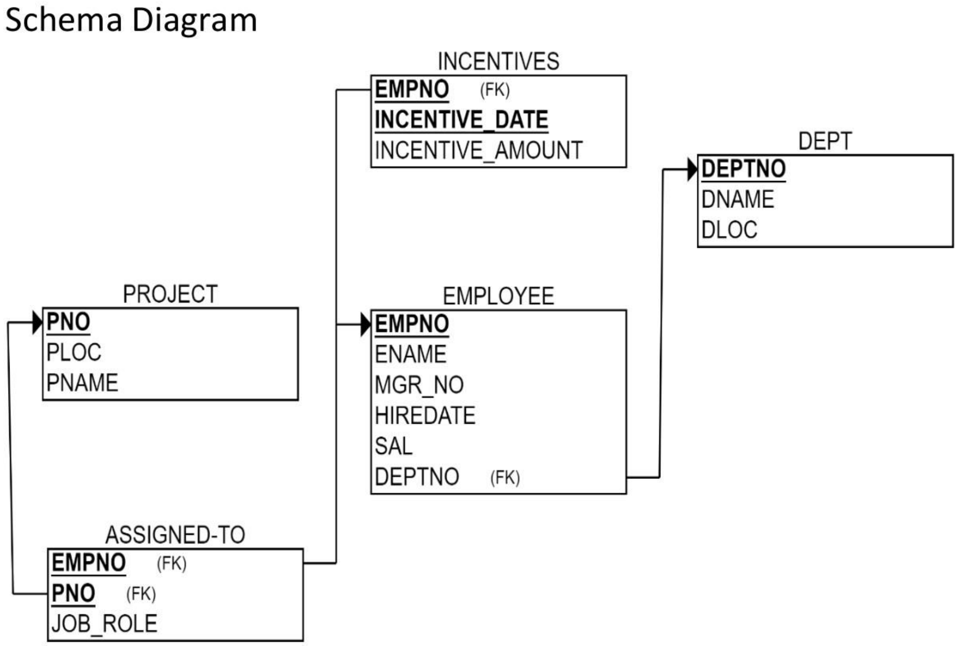
More Queries on Employee Database

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 those managers name whose salary is more than 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.

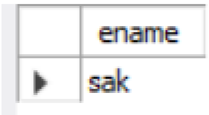
Schema Diagram



Queries

- **List the name of the managers with the maximum employees**

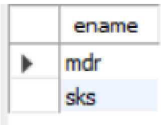
```
select ename from employee where empno = (select mgr_no from employee group by mgr_no
order by count(empno desc limit 1);
```



	ename
▶	sak

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

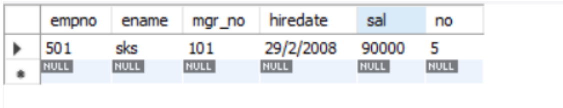
```
select ename from employee where sal > (select avg(sal) from employee);
```



	ename
▶	mdr
	sks

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

```
select * from employee
where empno=(select empno
                          from incentives
          where amt=(select max(amt)
                          from incentives
                          where amt <(select max(amt) from incentives
                                WHERE date like  "%01/2019")
                          )
                             );
```



	empno	ename	mgr_no	hiredate	sal	no
▶	501	sks	101	29/2/2008	90000	5
*	NULL	NULL	NULL	NULL	NULL	NULL

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

```
select emp.ename AS emp_name
from employee emp
join employee mgr where emp.mgr_no=mgr.empno
and emp.no=mgr.no;
```

Supplier Database

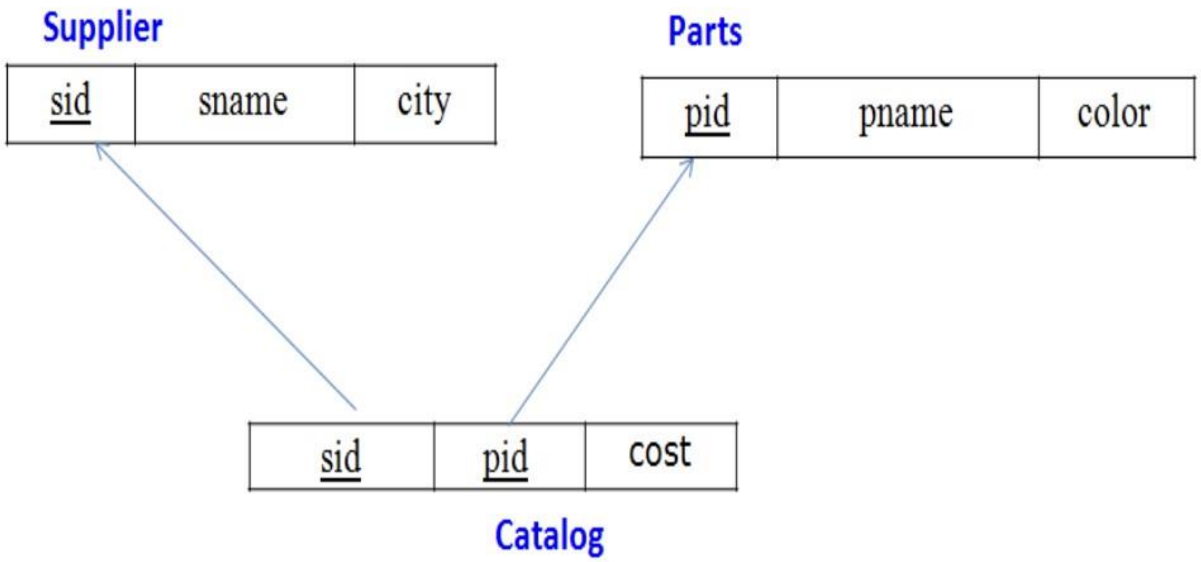
Question

(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 s;
use s;
create table Supplier(
sid int primary key,
sname varchar(20) ,
city varchar(20));
create table Parts(
pid int primary key,
pname varchar(20),
color varchar(20) );
create table Catalog(
sid int,
pid int,
cost int,
foreign key(sid) references Supplier(sid),
foreign key(pid) references Parts(pid));
```

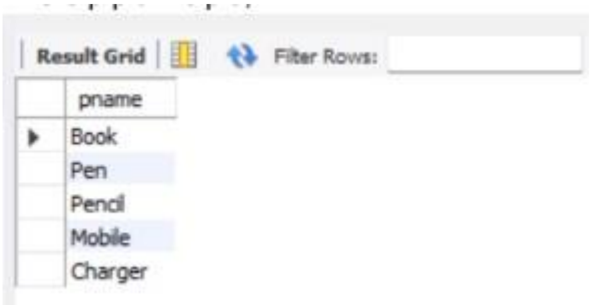
Insert appropriate records in each table.

```
insert into Supplier values (10001, 'Acme Widget','Bangalore');
insert into Supplier values (10002, 'Johns','Kolkata');
insert into Supplier values (10003, 'Vimal','Mumbai');
insert into Supplier values (10004, 'Reliance','Delhi');
insert into Parts values (20001, 'Book','Red');
insert into Parts values (20002, 'Pen','Red');
insert into Parts values (20003, 'Pencil','Green');
insert into Parts values (20004, 'Mobile','Green');
insert into Parts values (20005, 'Charger','Black');
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);
```

Queries

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

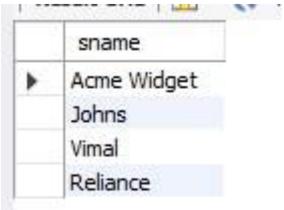
```
select distinct p.pname

from Parts p, Catalog c

where p.pid = c.pid;
```



pname
Book
Pen
Pencil
Mobile
Charger

3. Find the snames 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 =c.pid));
```

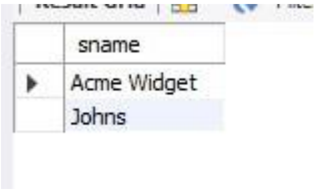


sname
Acme Widget
Johns
Vimal
Reliance

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

```
select distinct s.sname from
Catalog C, Supplier s where
C.sid=s.sid and
```

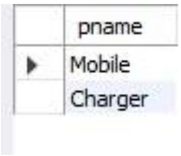
```
NOT EXISTS (select P.pid from Parts P where P.color="Red" and NOT EXISTS
(select C1.sid from Catalog C1 where C1.sid = C.sid and C1.pid = P.pid and
P.color="Red"));
```



	sname
▶	Acme Widget
	Johns

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

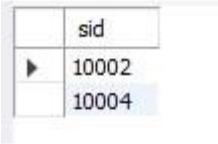
```
select p.pname from Parts p, Catalog c, Supplier s where p.pid=c.pid
and c.sid=s.sid and s.sname="Acme Widget" and NOT EXISTS (select
* from Catalog c1, Supplier s1 where p.pid=c1.pid and c1.sid=s1.sid
and s1.sname != "Acme Widget");
```



	pname
▶	Mobile
	Charger

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

```
select distinct C.sid from Catalog C where
C.cost > (select AVG(C1.cost) from Catalog C1
where C1.pid = C.pid);
```

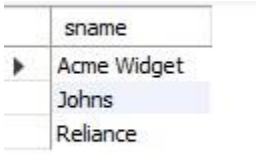


	sid
▶	10002
	10004

7. For each part, find the sname 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);
```



sname
Acme Widget
Johns
Reliance

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.

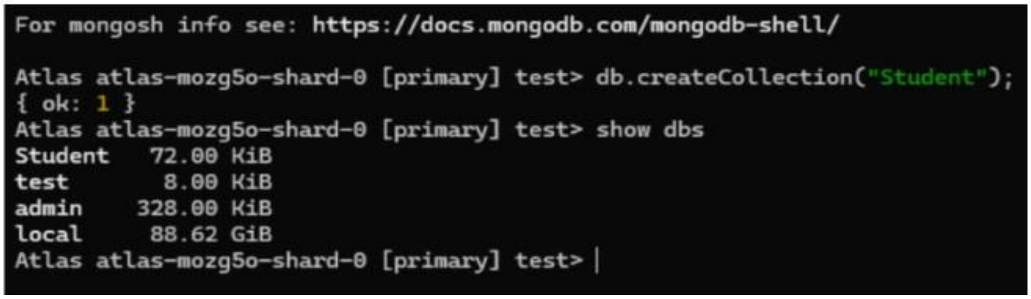
STRUCTURE OF THE COLLECTION

db.Student.find();

QUERIES

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

db.createCollection("Student"); show
dbs



```
For mongosh info see: https://docs.mongodb.com/mongodb-shell/

Atlas atlas-mozg5o-shard-0 [primary] test> db.createCollection("Student");
{ ok: 1 }
Atlas atlas-mozg5o-shard-0 [primary] test> show dbs
Student    72.00 KiB
test        8.00 KiB
admin     328.00 KiB
local      88.62 GiB
Atlas atlas-mozg5o-shard-0 [primary] test> |
```

Insert appropriate values

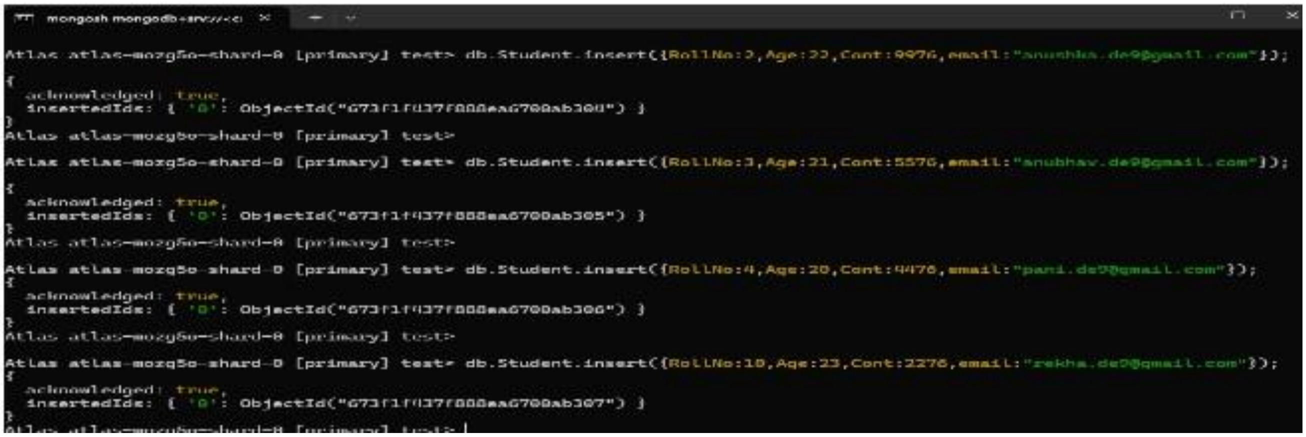
db.Student.insert({RollNo:1,Age:21,Cont:9876,email:"antara.de9@gmail.com"});

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

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

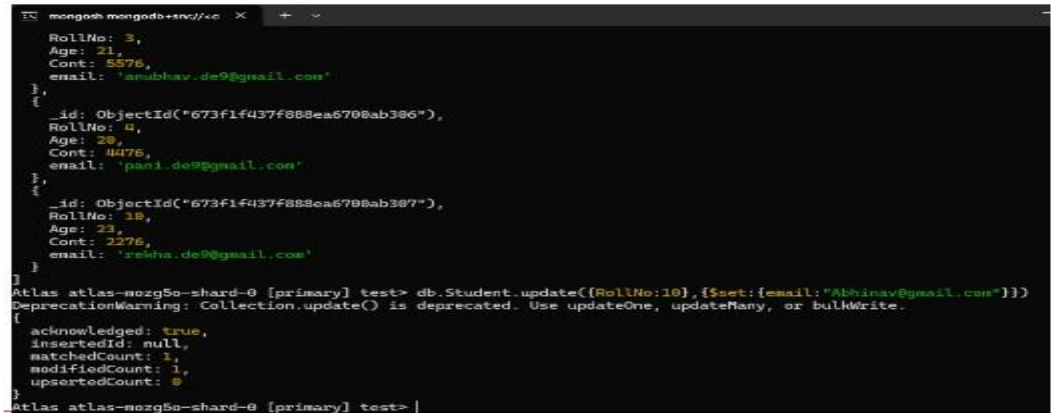
db.Student.insert({RollNo:4,Age:20,Cont:4476,email:"pani.de9@gmail.com"});

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



```
Atlas atlas-mozg5o-shard-0 [primary] test> db.Student.insert({RollNo:2,Age:22,Cont:9976,email:"anushka.de9@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("673f1f437f888ea6708ab304") }
}
Atlas atlas-mozg5o-shard-0 [primary] test>
Atlas atlas-mozg5o-shard-0 [primary] test> db.Student.insert({RollNo:3,Age:21,Cont:5576,email:"anubhav.de9@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("673f1f437f888ea6708ab305") }
}
Atlas atlas-mozg5o-shard-0 [primary] test>
Atlas atlas-mozg5o-shard-0 [primary] test> db.Student.insert({RollNo:4,Age:20,Cont:4476,email:"pani.de9@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("673f1f437f888ea6708ab306") }
}
Atlas atlas-mozg5o-shard-0 [primary] test>
Atlas atlas-mozg5o-shard-0 [primary] test> db.Student.insert({RollNo:10,Age:23,Cont:2276,email:"rekha.de9@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("673f1f437f888ea6708ab307") }
}
Atlas atlas-mozg5o-shard-0 [primary] test> |
```

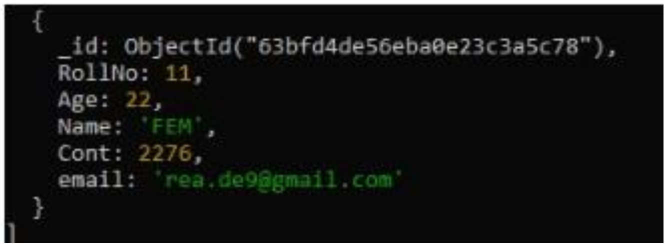
- Write a query to update the Email-Id of a student with rollno 5. db.Student.update({RollNo:10},{$set:{email:"Abhinav@gmail.com"}})



```
    RollNo: 3,
    Age: 21,
    Cont: 5576,
    email: 'anubhav.de9@gmail.com'
  },
  {
    _id: ObjectId("673f1f437f888ea6708ab306"),
    RollNo: 4,
    Age: 20,
    Cont: 4476,
    email: 'pani.de9@gmail.com'
  },
  {
    _id: ObjectId("673f1f437f888ea6708ab307"),
    RollNo: 10,
    Age: 23,
    Cont: 2276,
    email: 'rekha.de9@gmail.com'
  }
]
Atlas atlas-mozg5o-shard-0 [primary] test> db.Student.update({RollNo:10},{$set:{email:"Abhinav@gmail.com"}})
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Atlas atlas-mozg5o-shard-0 [primary] test> |
```

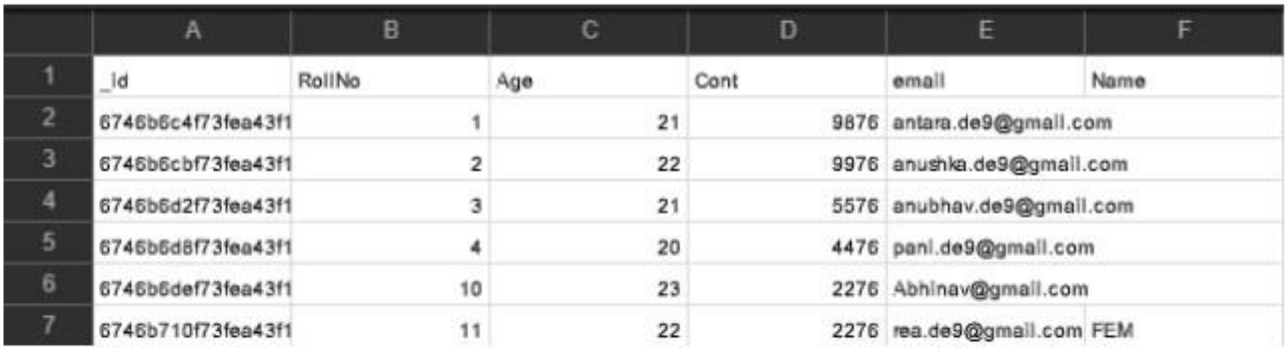
- Replace the student name from “ABC” to “FEM” of rollno 11.

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



```
  {
    _id: ObjectId("63bfd4de56eba0e23c3a5c78"),
    RollNo: 11,
    Age: 22,
    Name: 'FEM',
    Cont: 2276,
    email: 'rea.de9@gmail.com'
  }
]
```

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



	A	B	C	D	E	F
1	_id	RollNo	Age	Cont	email	Name
2	6746b6c4f73fea43f1	1	21	9876	antara.de9@gmail.com	
3	6746b6cbf73fea43f1	2	22	9976	anushka.de9@gmail.com	
4	6746b6d2f73fea43f1	3	21	5576	anubhav.de9@gmail.com	
5	6746b6d8f73fea43f1	4	20	4476	pani.de9@gmail.com	
6	6746b6def73fea43f1	10	23	2276	Abhinav@gmail.com	
7	6746b710f73fea43f1	11	22	2276	rea.de9@gmail.com	FEM

NoSQL Lab 2

Question

(Week 9)

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

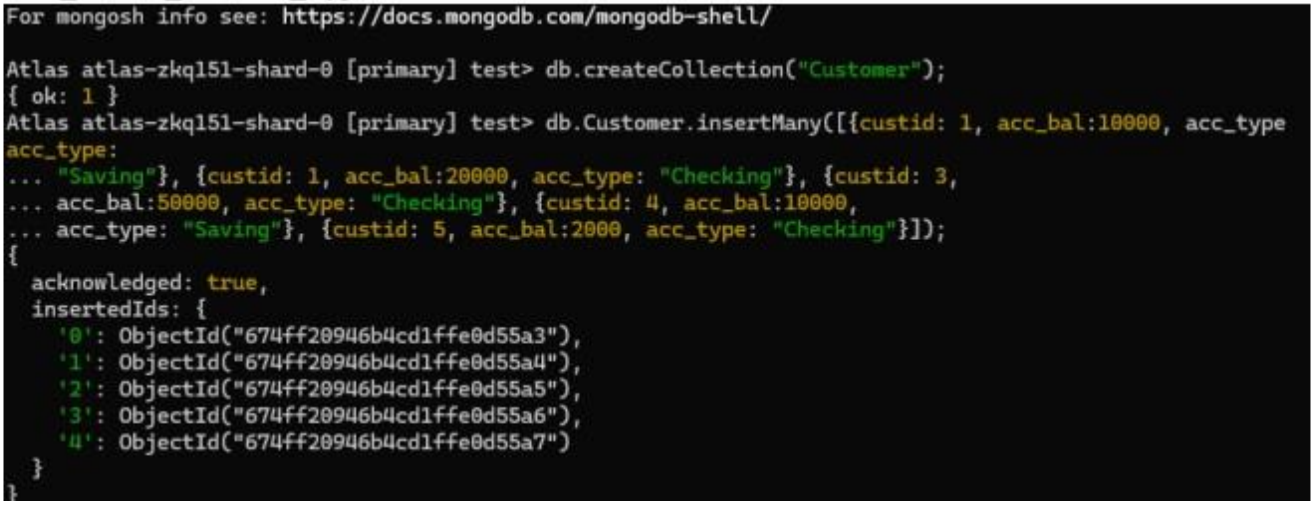
7. Import a given csv dataset from local file system into mongodb collection.

Create Table:

db.createCollection("Customer"); **Inserting**

Values:

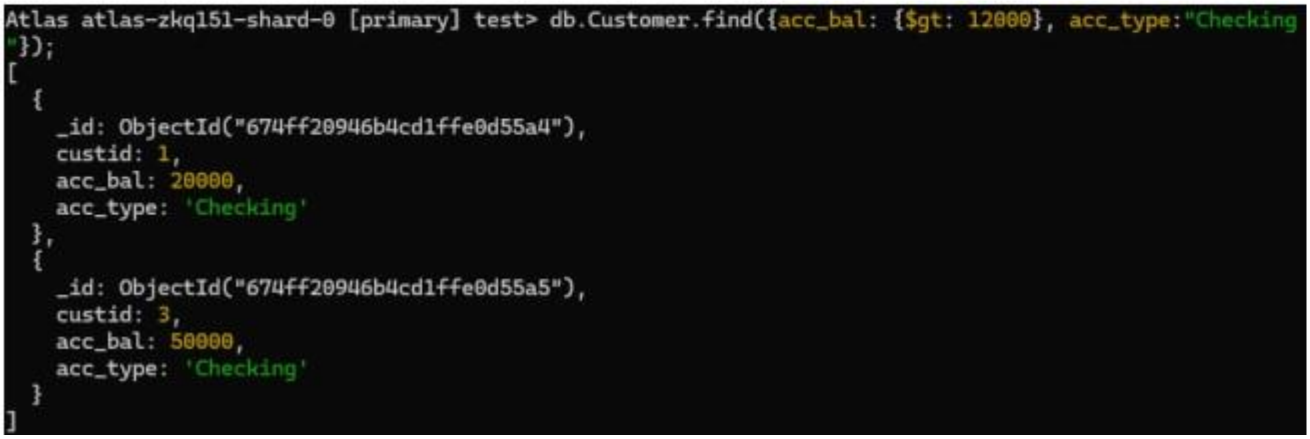
db.Customer.insertMany([{custid: 1, acc_bal:10000, acc_type: "Saving"}, {custid: 1, acc_bal:20000, acc_type: "Checking"}, {custid: 3, acc_bal:50000, acc_type: "Checking"}, {custid: 4, acc_bal:10000, acc_type: "Saving"}, {custid: 5, acc_bal:2000, acc_type: "Checking"}]);



```
For mongosh info see: https://docs.mongodb.com/mongodb-shell/

Atlas atlas-zkq151-shard-0 [primary] test> db.createCollection("Customer");
{ ok: 1 }
Atlas atlas-zkq151-shard-0 [primary] test> db.Customer.insertMany([{custid: 1, acc_bal:10000, acc_type
acc_type:
... "Saving"}, {custid: 1, acc_bal:20000, acc_type: "Checking"}, {custid: 3,
... acc_bal:50000, acc_type: "Checking"}, {custid: 4, acc_bal:10000,
... acc_type: "Saving"}, {custid: 5, acc_bal:2000, acc_type: "Checking"}]);
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("674ff20946b4cd1ffe0d55a3"),
    '1': ObjectId("674ff20946b4cd1ffe0d55a4"),
    '2': ObjectId("674ff20946b4cd1ffe0d55a5"),
    '3': ObjectId("674ff20946b4cd1ffe0d55a6"),
    '4': ObjectId("674ff20946b4cd1ffe0d55a7")
  }
}
```

Finding all checking accounts with balance greater than 12000 db.Customer.find({acc_bal: {$gt:

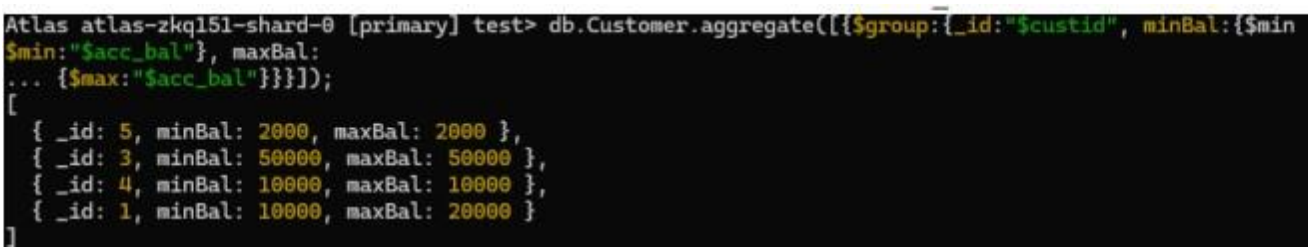
12000}, acc_type:"Checking"});



```
Atlas atlas-zkq151-shard-0 [primary] test> db.Customer.find({acc_bal: {$gt: 12000}, acc_type:"Checking
"});
[
  {
    _id: ObjectId("674ff20946b4cd1ffe0d55a4"),
    custid: 1,
    acc_bal: 20000,
    acc_type: 'Checking'
  },
  {
    _id: ObjectId("674ff20946b4cd1ffe0d55a5"),
    custid: 3,
    acc_bal: 50000,
    acc_type: 'Checking'
  }
]
```

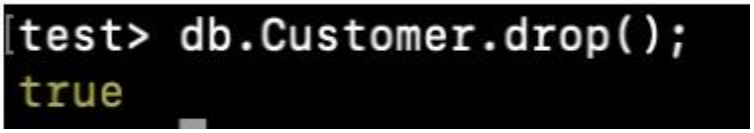
Finding the maximum and minimum balance of each customer

db.Customer.aggregate([{$group:{_id:"$custid", minBal:{$min:"$acc_bal"}, maxBal: {$max:"$acc_bal"}}}]);



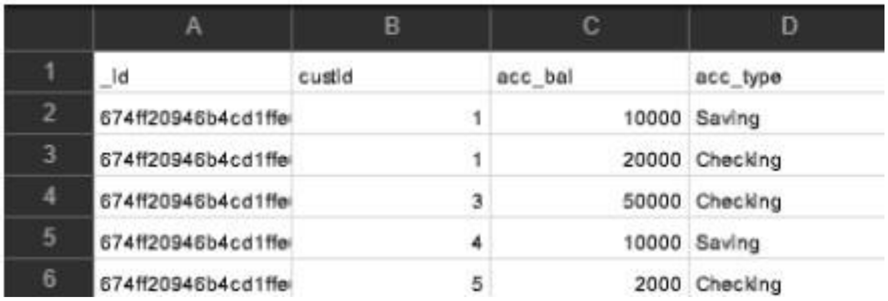
```
Atlas atlas-zkq151-shard-0 [primary] test> db.Customer.aggregate([{$group:{_id:"$custid", minBal:{$min
$min:"$acc_bal"}, maxBal:
... {$max:"$acc_bal"}}}]);
[
  { _id: 5, minBal: 2000, maxBal: 2000 },
  { _id: 3, minBal: 50000, maxBal: 50000 },
  { _id: 4, minBal: 10000, maxBal: 10000 },
  { _id: 1, minBal: 10000, maxBal: 20000 }
]
```

Dropping collection "Customer" db.Customer.drop();



```
test> db.Customer.drop();
true
```

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



	A	B	C	D
1	_id	custid	acc_bal	acc_type
2	674ff20946b4cd1ffe	1	10000	Saving
3	674ff20946b4cd1ffe	1	20000	Checking
4	674ff20946b4cd1ffe	3	50000	Checking
5	674ff20946b4cd1ffe	4	10000	Saving
6	674ff20946b4cd1ffe	5	2000	Checking

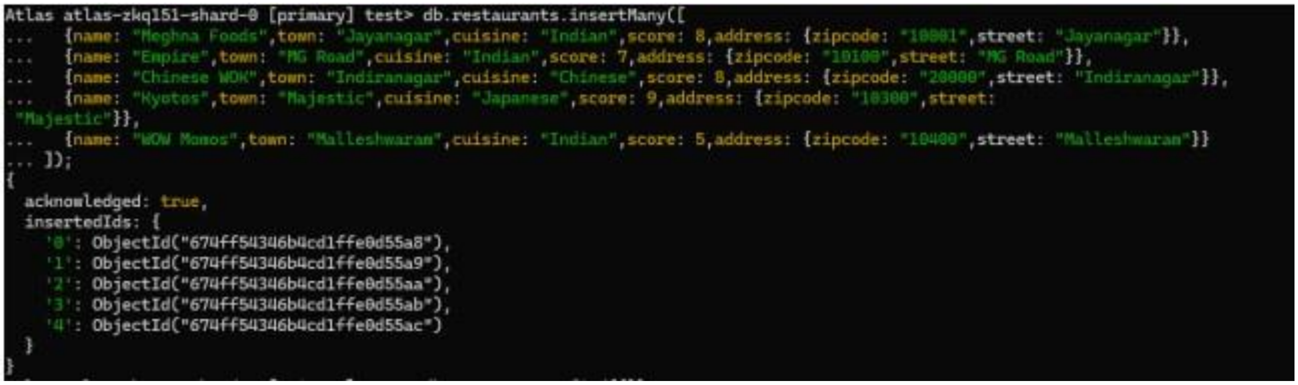
NoSQL Lab 3

Question

(Week 10)

1. Write a MongoDB query to display all the documents in the collection restaurants.
2. Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns.
3. 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.
4. Write a MongoDB query to find the average score for each restaurant.
5. Write a MongoDB query to find the name and address of the restaurants that have a zipcode that starts with '10'.

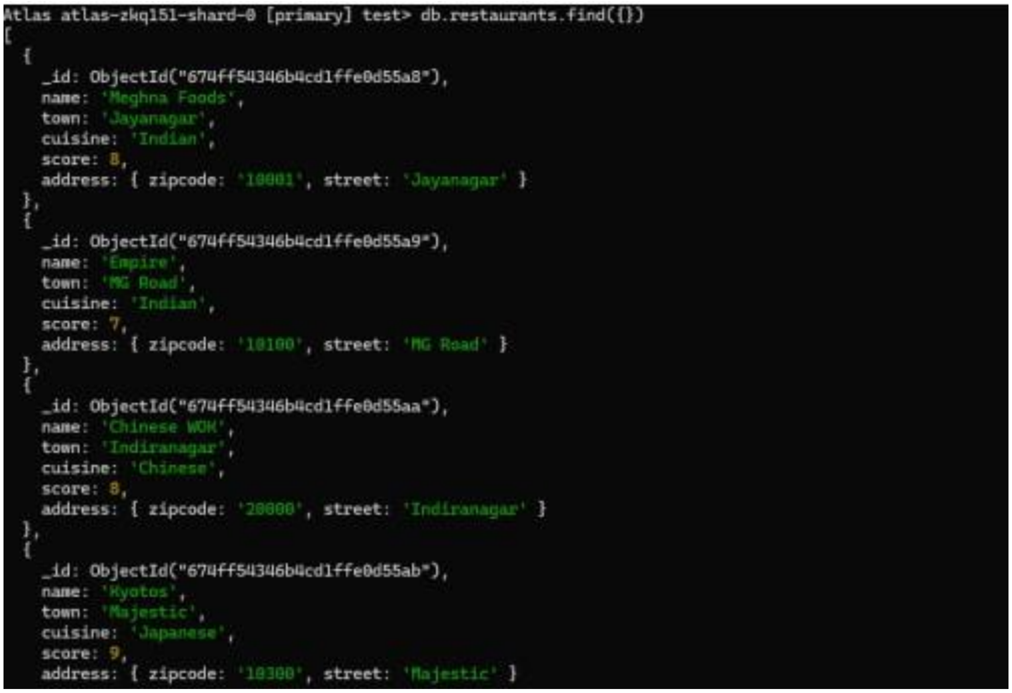
```
db.createCollection("restaurants");

db.restaurants.insertMany([

{ name: "Meghna Foods", town: "Jayanagar", cuisine: "Indian", score: 8, address: { zipcode: "10001",
street: "Jayanagar“

} },

{ name: "Empire", town: "MG Road", cuisine: "Indian", score: 7, address: { zipcode: "10100", street: "MG
Road" } },

{ name: "Chinese WOK", town: "Indiranagar", cuisine: "Chinese", score: 12, address: { zipcode:
"20000", street: "Indiranagar" } },

{ name: "Kyotos", town: "Majestic", cuisine: "Japanese", score: 9, address: { zipcode: "10300", street:
"Majestic" } },

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

} ])
```



```
Atlas atlas-zkq151-shard-0 [primary] test> db.restaurants.insertMany([
...    {name: "Meghna Foods",town: "Jayanagar",cuisine: "Indian",score: 8,address: {zipcode: "10001",street: "Jayanagar"}},
...    {name: "Empire",town: "MG Road",cuisine: "Indian",score: 7,address: {zipcode: "10100",street: "MG Road"}},
...    {name: "Chinese WOK",town: "Indiranagar",cuisine: "Chinese",score: 8,address: {zipcode: "20000",street: "Indiranagar"}},
...    {name: "Kyotos",town: "Majestic",cuisine: "Japanese",score: 9,address: {zipcode: "10300",street:
 "Majestic"}},
...    {name: "WOW Momos",town: "Malleshwaram",cuisine: "Indian",score: 5,address: {zipcode: "10400",street: "Malleshwaram"}}
... ]);
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("674ff54346b4cd1ffe0d55a8"),
    '1': ObjectId("674ff54346b4cd1ffe0d55a9"),
    '2': ObjectId("674ff54346b4cd1ffe0d55aa"),
    '3': ObjectId("674ff54346b4cd1ffe0d55ab"),
    '4': ObjectId("674ff54346b4cd1ffe0d55ac")
  }
}
```

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

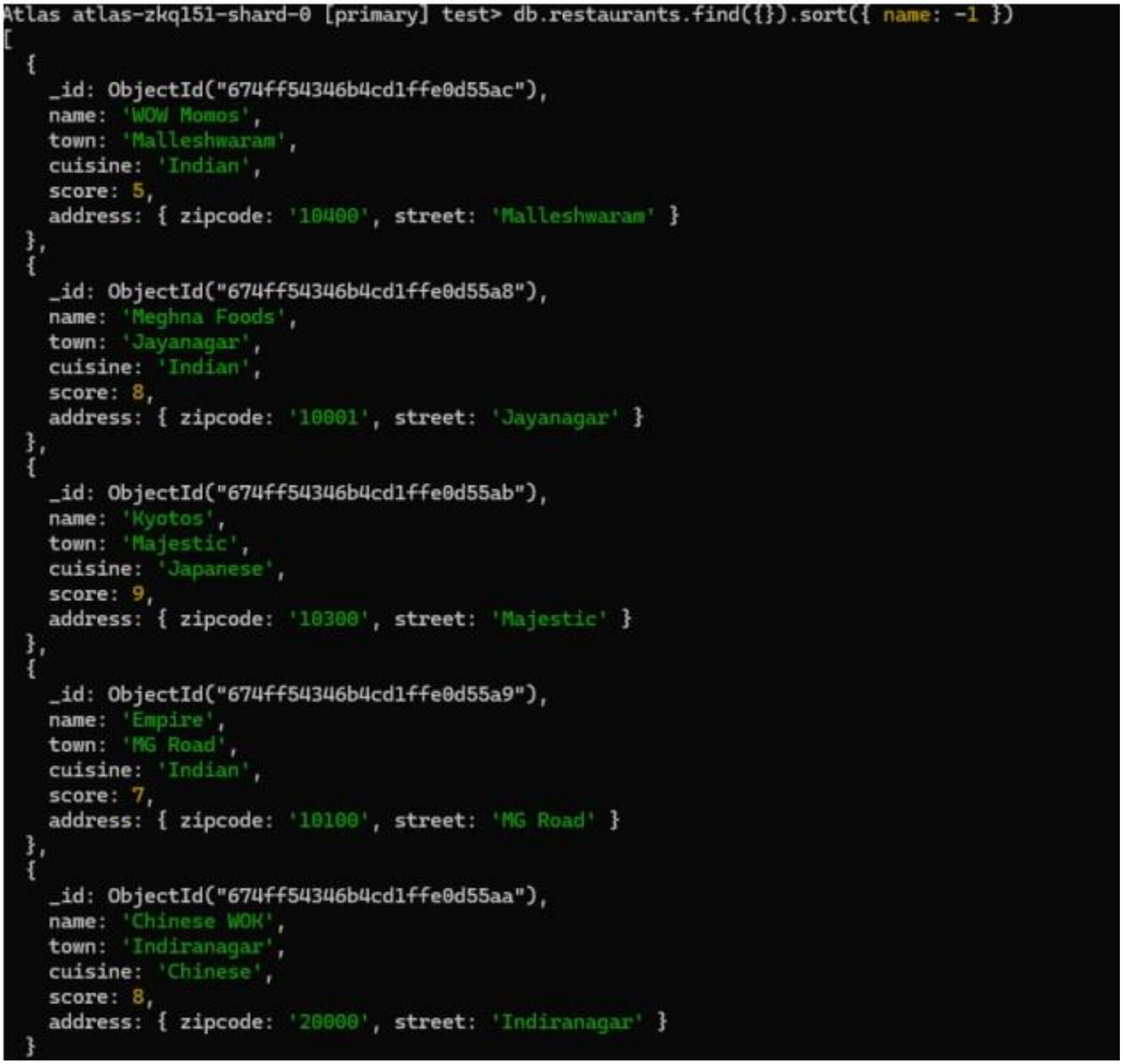
db.restaurants.find({})



```
Atlas atlas-zkq151-shard-0 [primary] test> db.restaurants.find({})
[
  {
    _id: ObjectId("674ff54346b4cd1ffe0d55a8"),
    name: 'Meghna Foods',
    town: 'Jayanagar',
    cuisine: 'Indian',
    score: 8,
    address: { zipcode: '10001', street: 'Jayanagar' }
  },
  {
    _id: ObjectId("674ff54346b4cd1ffe0d55a9"),
    name: 'Empire',
    town: 'MG Road',
    cuisine: 'Indian',
    score: 7,
    address: { zipcode: '10100', street: 'MG Road' }
  },
  {
    _id: ObjectId("674ff54346b4cd1ffe0d55aa"),
    name: 'Chinese WOK',
    town: 'Indiranagar',
    cuisine: 'Chinese',
    score: 8,
    address: { zipcode: '20000', street: 'Indiranagar' }
  },
  {
    _id: ObjectId("674ff54346b4cd1ffe0d55ab"),
    name: 'Kyotos',
    town: 'Majestic',
    cuisine: 'Japanese',
    score: 9,
    address: { zipcode: '10300', street: 'Majestic' }
```

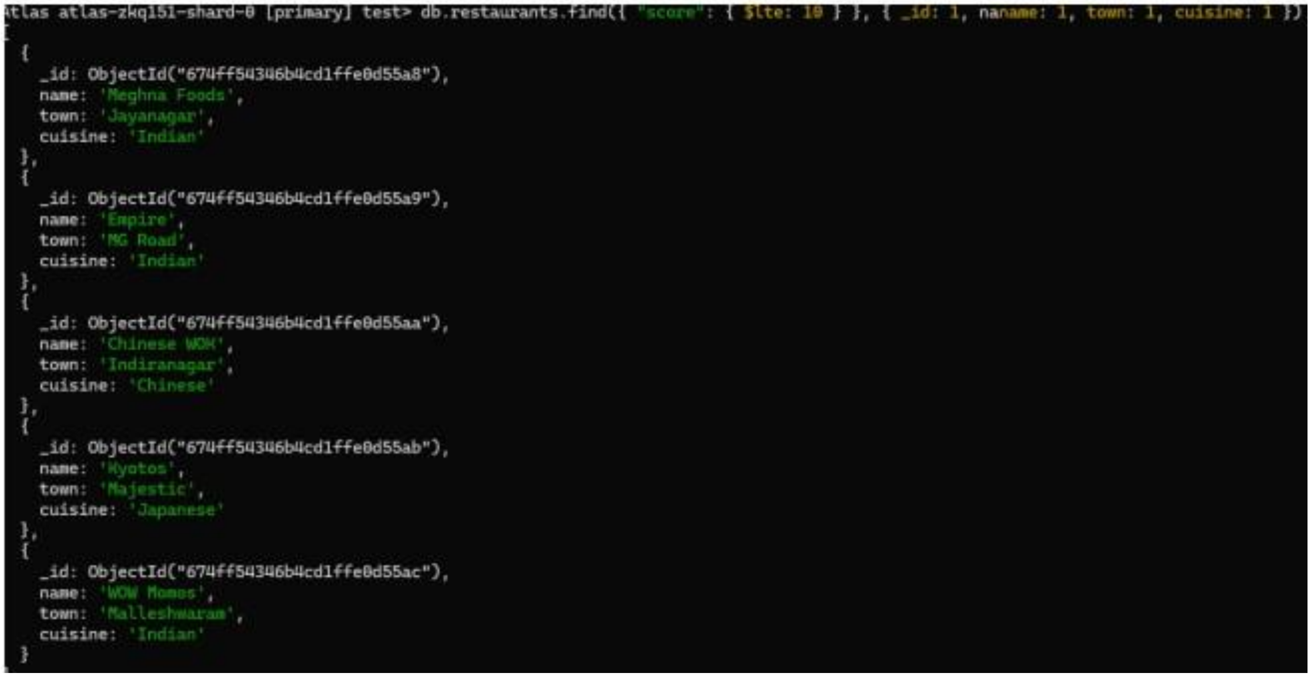
Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns.

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



```
Atlas atlas-zkql51-shard-0 [primary] test> db.restaurants.find({}).sort({ name: -1 })
[
  {
    _id: ObjectId("674ff54346b4cd1ffe0d55ac"),
    name: 'WOW Momos',
    town: 'Malleshwaram',
    cuisine: 'Indian',
    score: 5,
    address: { zipcode: '10400', street: 'Malleshwaram' }
  },
  {
    _id: ObjectId("674ff54346b4cd1ffe0d55a8"),
    name: 'Meghna Foods',
    town: 'Jayanagar',
    cuisine: 'Indian',
    score: 8,
    address: { zipcode: '10001', street: 'Jayanagar' }
  },
  {
    _id: ObjectId("674ff54346b4cd1ffe0d55ab"),
    name: 'Kyotos',
    town: 'Majestic',
    cuisine: 'Japanese',
    score: 9,
    address: { zipcode: '10300', street: 'Majestic' }
  },
  {
    _id: ObjectId("674ff54346b4cd1ffe0d55a9"),
    name: 'Empire',
    town: 'MG Road',
    cuisine: 'Indian',
    score: 7,
    address: { zipcode: '10100', street: 'MG Road' }
  },
  {
    _id: ObjectId("674ff54346b4cd1ffe0d55aa"),
    name: 'Chinese WOK',
    town: 'Indiranagar',
    cuisine: 'Chinese',
    score: 8,
    address: { zipcode: '20000', street: 'Indiranagar' }
  }
```

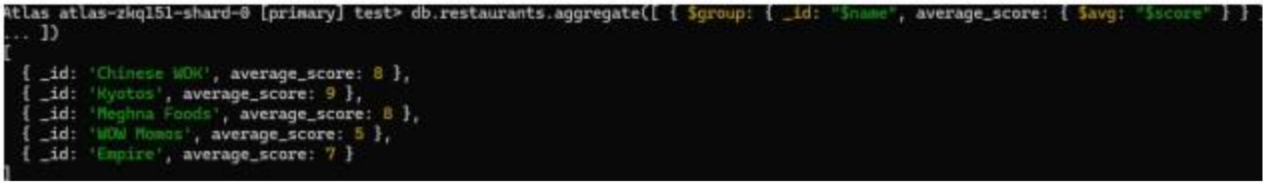
Query to find the restaurant Id, name, town and cuisine for those restaurants which achieved a score which is not more than 10 db.restaurants.find({ "score": { $lte: 10 } }, { _id: 1, name: 1, town: 1, cuisine: 1 })



```
Atlas atlas-zkq151-shard-0 [primary] test> db.restaurants.find({ "score": { $lte: 10 } }, { _id: 1, naname: 1, town: 1, cuisine: 1 })
[
  {
    _id: ObjectId("674ff54346b4cd1ffe0d55a8"),
    name: 'Meghna Foods',
    town: 'Jayanagar',
    cuisine: 'Indian'
  },
  {
    _id: ObjectId("674ff54346b4cd1ffe0d55a9"),
    name: 'Empire',
    town: 'MG Road',
    cuisine: 'Indian'
  },
  {
    _id: ObjectId("674ff54346b4cd1ffe0d55aa"),
    name: 'Chinese WOK',
    town: 'Indiranagar',
    cuisine: 'Chinese'
  },
  {
    _id: ObjectId("674ff54346b4cd1ffe0d55ab"),
    name: 'Kyotos',
    town: 'Majestic',
    cuisine: 'Japanese'
  },
  {
    _id: ObjectId("674ff54346b4cd1ffe0d55ac"),
    name: 'WOW Momos',
    town: 'Malleshwaram',
    cuisine: 'Indian'
  }
```

Query to find the average score for each restaurant

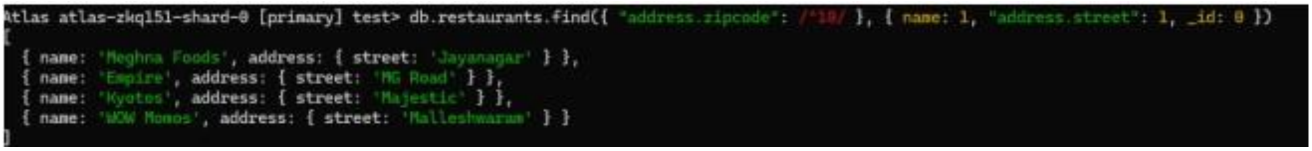
db.restaurants.aggregate([{ $group: { _id: "$name", average_score: { $avg: "$score" } } }
])



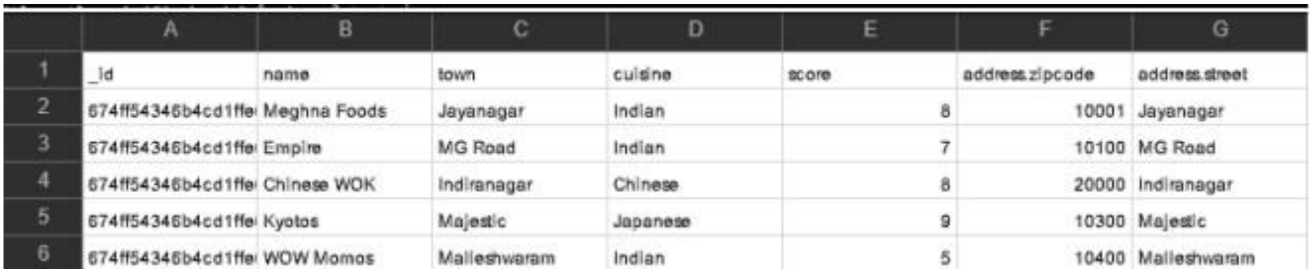
```
Atlas atlas-zkq151-shard-0 [primary] test> db.restaurants.aggregate([ { $group: { _id: "$name", average_score: { $avg: "$score" } } }
... ])
[
  { _id: 'Chinese WOK', average_score: 8 },
  { _id: 'Kyotos', average_score: 9 },
  { _id: 'Meghna Foods', average_score: 8 },
  { _id: 'WOW Momos', average_score: 5 },
  { _id: 'Empire', average_score: 7 }
]
```

Query to find the name and address of the restaurants that have a zipcode that starts with '10'.

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



```
Atlas atlas-zkq151-shard-0 [primary] test> db.restaurants.find({ "address.zipcode": /^10/ }, { name: 1, "address.street": 1, _id: 0 })
[
  { name: 'Meghna Foods', address: { street: 'Jayanagar' } },
  { name: 'Empire', address: { street: 'MG Road' } },
  { name: 'Kyotos', address: { street: 'Majestic' } },
  { name: 'WOW Momos', address: { street: 'Malleshwaram' } }
]
```



	A	B	C	D	E	F	G
1	_id	name	town	cuisine	score	address.zipcode	address.street
2	674ff54346b4cd1ffe	Meghna Foods	Jayanagar	Indian	8	10001	Jayanagar
3	674ff54346b4cd1ffe	Empire	MG Road	Indian	7	10100	MG Road
4	674ff54346b4cd1ffe	Chinese WOK	Indiranagar	Chinese	8	20000	Indiranagar
5	674ff54346b4cd1ffe	Kyotos	Majestic	Japanese	9	10300	Majestic
6	674ff54346b4cd1ffe	WOW Momos	Malleshwaram	Indian	5	10400	Malleshwaram