VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Database Management Systems (23CS3PCDBM)

Submitted by

Bhoomi Udedh (1BM23CS066)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

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

B. M. S. College of Engineering,
Bull Temple Road, Bangalore 560019
(Affiliated To Visvesvaraya Technological University, Belgaum)
Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "Database Management Systems (23CS3PCDBM)" carried out by **Bhoomi Udedh (1BM23CS066),** who is a bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a Database Management Systems (23CS3PCDBM) work prescribed for the said degree.

Lab faculty Incharge Name Assistant Professor Department of CSE, BMSCE	Dr.Kavitha Sooda Professor & HOD Department of CSE, BMSCE

Index

Insurance Database

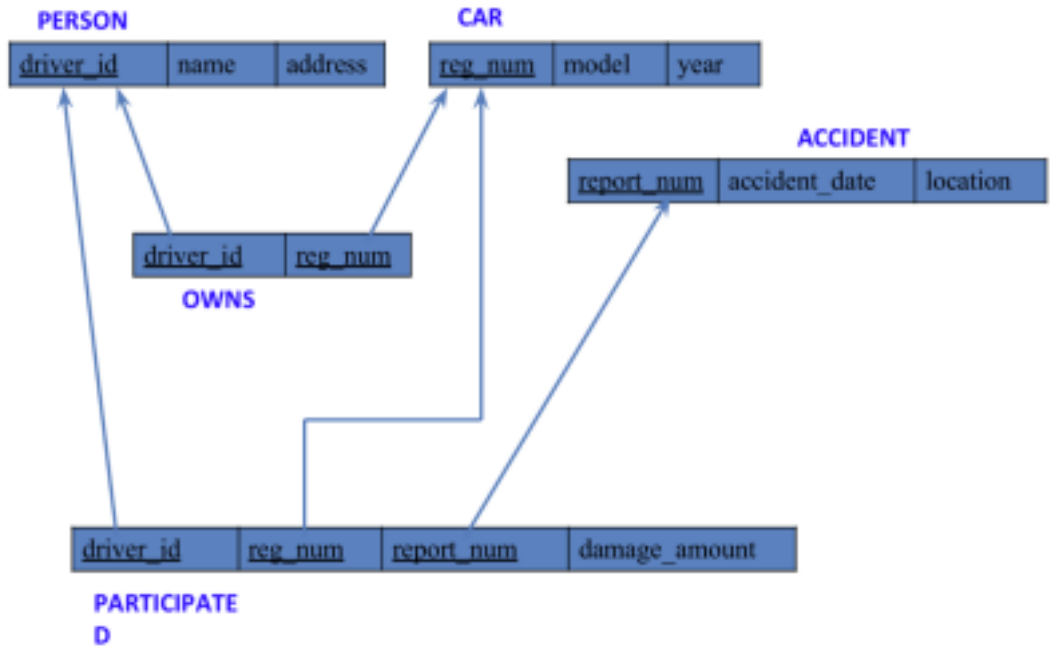
Question

(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'KA053408') for which the accident report number was 12.
- Add a new accident to the database.
- 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 bhoomi;

use bhoomi;
```

Create table

```
create table person(

driver_id varchar(20),

name varchar(20),

address varchar(20),

PRIMARY KEY(driver_id)

);

create table car(

reg_num varchar(20),

model varchar(20),
year int,

PRIMARY KEY(reg_num)

);

create table owns(
```

```
driver_id varchar(20),
reg_num varchar(20),
PRIMARY KEY(driver_id, reg_num),
FOREIGN KEY(driver_id) REFERENCES person(driver_id),
FOREIGN KEY(reg_num) REFERENCES car(reg_num)
);
create table accident(
report_num int,
accident_date date,
location varchar(20),
PRIMARY KEY(report_num)
);
create table participated(
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(driver_id),
FOREIGN KEY(reg_num) REFERENCES car(reg_num),
FOREIGN KEY(report_num) REFERENCES accident(report_num)
);
```

Structure of the table

```
desc person;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

desc accident;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
report_num	int	NO	PRI	NULL	
accident_date	date	YES		NULL	
loaction	varchar(20)	YES		NULL	

desc participated;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

desc car;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
reg_num	varchar(20)	NO	PRI	NULL	
model	varchar(20)	YES		NULL	
year	int	YES		NULL	

desc owns;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

Inserting Values to the table

```
insert into person VALUES('A01','Richard','srinivas nagar');
insert into person VALUES('A02','Pradeep','Rajajinagar');
insert into person VALUES('A03','Smith','Ashok nagar');
insert into person VALUES('A04','Venu','N R colony');
insert into person VALUES('A05','Jhon','Hanumanth nagar');
select * from person;
```

Result Grid | Filter Rows:

driver_id	name	address
A01	Richard	srinivas nagar
A02	Pradeep	Rajajinagar
A03	Smith	Ashok nagar
A04	Venu	N R colony
A05	Jhon	Hanumanth nagar
NULL	NULL	NULL

```
insert into car VALUES('KA052250','Indica',1990);
insert into car VALUES('KA031181','Lancer',1957);
insert into car VALUES('KA095477','Toyota',1998);
insert into car VALUES('KA053408','Honda',2008);
insert into car VALUES('KA041702','Audi',2005);
select * from car;
```

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

```
insert into owns VALUES('A01','KA052250');
insert into owns VALUES('A02','KA053408');
```

```
insert into owns VALUES('A03','KA031181');
insert into owns VALUES('A04','KA095477');
insert into owns VALUES('A05','KA041702');
select * from owns;
```

Result Grid | Filter Rows:

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

```
insert into accident VALUES(11,'01-01-03','Mysore road');
insert into accident VALUES(12,'02-02-04','South end circle');
insert into accident VALUES(13,'21-01-03','Bull temple road');
insert into accident VALUES(14,'17-02-08','Mysore road');
insert into accident VALUES(15,'04-03-05','Kanakpura road');
select * from accident;
```

report_num	accident_date	loaction
11	2001-01-03	Mysore road
12	2002-02-04	South end circle
13	2021-01-03	Bull temple road
14	2017-02-08	Mysore road
15	2004-03-05	Kanakpura road
NULL	NULL	NULL

```
insert into participated VALUES('A01','KA052250',11,10000);
insert into participated VALUES('A02','KA053408',12,50000);
insert into participated VALUES('A03','KA095477',13,25000);
insert into participated VALUES('A04','KA031181',14,3000);
insert into participated VALUES('A05','KA041702',15,5000);
select * from participated;
```

driver_id	reg_num	report_num	damage_amt
A01	KA052250	11	10000
A02	KA053408	12	50000
A03	KA095477	13	25000
A04	KA031181	14	3000
A05	KA041702	15	5000
NULL	NULL	NULL	NULL

Queries

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

```
update participated
set damage_amt=25000
where reg_num="KA053408" and report_num=12;
select * from participated;
```

driver_id	reg_num	report_num	damage_amt
A01	KA052250	11	10000
A02	KA053408	12	25000
A03	KA095477	13	25000
A04	KA031181	14	3000
A05	KA041702	15	5000
NULL	NULL	NULL	NULL

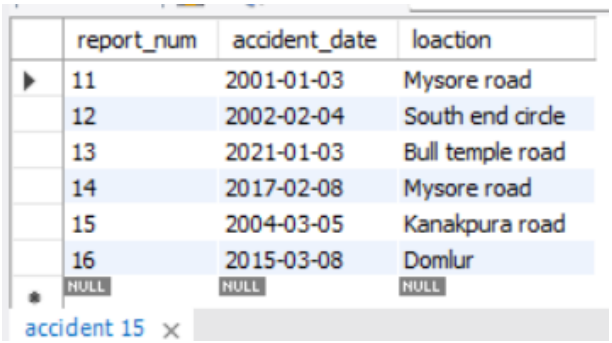
2)Find the total number of people who owned cars that were involved in accidents in 2008.

```
select count(distinct driver_id)
from participated a, accident b
where a.report_num=b.report_num and b.accident_date like '2008%';
```

count(distinct driver_id)
0

3)Add a new accident to the database.

insert into accident values(16,"15-03-08","Domlur");
select * from accident;



report_num	accident_date	loaction
11	2001-01-03	Mysore road
12	2002-02-04	South end circle
13	2021-01-03	Bull temple road
14	2017-02-08	Mysore road
15	2004-03-05	Kanakpura road
16	2015-03-08	Domlur
NULL	NULL	NULL

accident 15

4) Display accident date and location.

select accident_date date,loaction
from accident;

date	loaction
2001-01-03	Mysore road
2002-02-04	South end circle
2021-01-03	Bull temple road
2017-02-08	Mysore road
2004-03-05	Kanakpura road
2015-03-08	Domlur

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

select driver_id
from participated
where damage_amt>=25000;

driver_id
A02
A03

More Queries on Insurance Database

(week 2)

Question

1) List the entire participated relation in descending order of damage_amt.

2) Find the average damage_amt.

3) Delete the tuple from participated whose damage amount is below average damage amount.

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

5) Find maximum damage amount.

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

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

Queries

1) List the entire participated relation in descending order of damage_amt.

SELECT * FROM PARTICIPATED ORDER BY DAMAGE_AMT DESC;

driver_id	reg_num	report_num	damage_amt
A02	KA053408	12	25000
A03	KA095477	13	25000
A01	KA052250	11	10000
A05	KA041702	15	5000
A04	KA031181	14	3000
NULL	NULL	NULL	NULL

2) Find the average damage_amt.

SELECT AVG(DAMAGE_AMT) FROM PARTICIPATED;

	AVG(DAMAGE_AMT)
▶	13600.0000

3) Delete the tuple from participated whose damage amount is below average damage amount.

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

SELECT NAME FROM PERSON A, PARTICIPATED B WHERE A.DRIVER_ID = B.DRIVER_ID AND DAMAGE_AMT>(SELECT AVG(DAMAGE_AMT) FROM PARTICIPATED);

	NAME
▶	Pradeep
	Smith

5) Find maximum damage amount.

SELECT MAX(DAMAGE_AMT) FROM PARTICIPATED;

	MAX(DAMAGE_AMT)
▶	25000

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

select*

from car

order by year asc;

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

7) Find the number of accidents in which cars belonging to a specific model (example 'Lancer')

```
select count(report_num)
from car c, participated p
where c.reg_num=p.reg_num and c.model='Lancer';
```

	count(report_num)
▶	1

Week-3 **Bank Database**

Question

branch (branch-name: String, branch-city: String, assets: real)

bankaccount(accno: int, branch-name: String, balance: real)

bankcustomer (customer-name: String, customer-street: String,

customer-city: String)

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

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

1.Create the above tables by properly specifying the primary keys and the foreign keys.

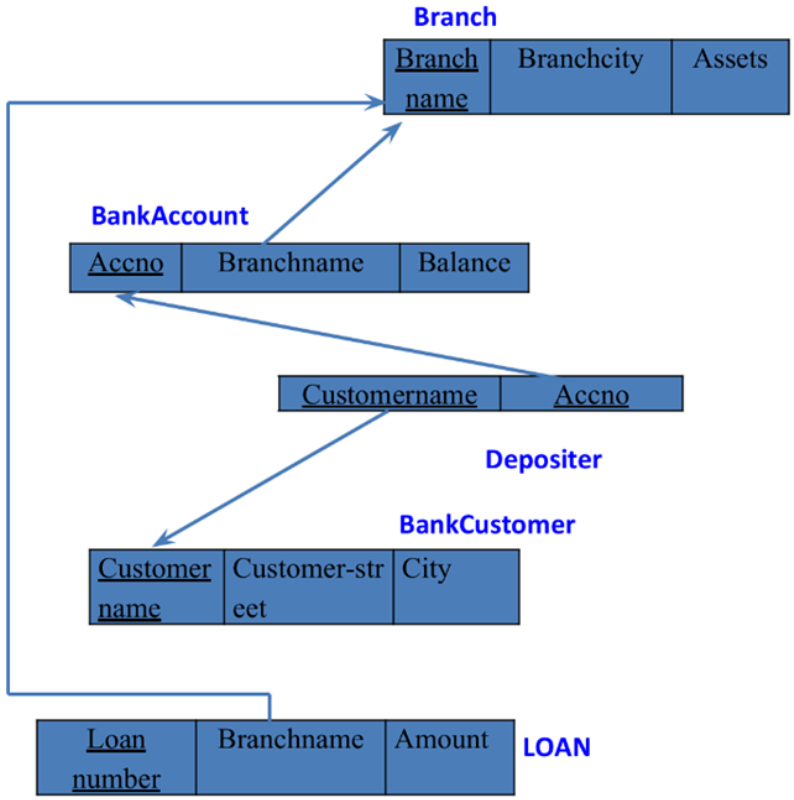
2.Enter at least five tuples for each relation.

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

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

5.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 bank345;
use bank345;
```

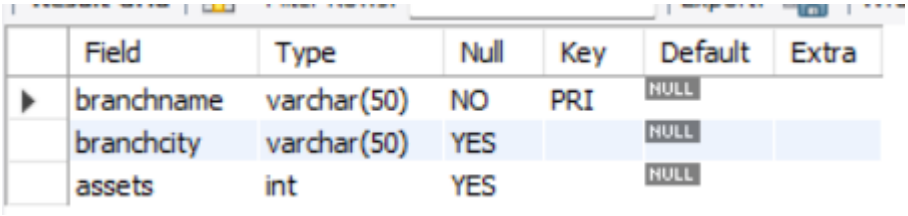
Create Table

```
create table branch (
branchname varchar(50),
branchcity varchar(50),
assets int,
primary key (branchname));

create table bankcustomer(
customername varchar(50),
customer_street varchar(50),
city varchar(50),
primary key(customername));
```

```
create table bankaccount (
accno int,
branchname varchar(50),
balance int,
primary key (accno),

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

create table depositer(
customername varchar(50),
accno int,
primary key (customername, accno),
foreign key (customername) references bankcustomer(customername),
foreign key (accno) references bankaccount(accno));

create table loan(
loannumber int,

branchname varchar(50),
amount int,
primary key (loannumber),
foreign key (branchname) references branch (branchname));
```

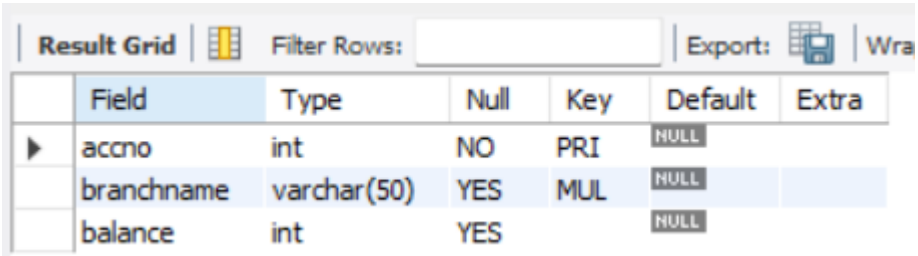
Structure of the Table

desc branch;



Field	Type	Null	Key	Default	Extra
branchname	varchar(50)	NO	PRI	NULL	
branchcity	varchar(50)	YES		NULL	
assets	int	YES		NULL	

desc bankaccount;



Result Grid | Filter Rows: | Export: | Wra

Field	Type	Null	Key	Default	Extra
accno	int	NO	PRI	NULL	
branchname	varchar(50)	YES	MUL	NULL	
balance	int	YES		NULL	

desc bankcustomer;

Result Grid | Filter Rows: | Export: | Wrap Cell

Field	Type	Null	Key	Default	Extra
customername	varchar(50)	NO	PRI	NULL	
customer_street	varchar(50)	YES		NULL	
city	varchar(50)	YES		NULL	

desc depositer;

Result Grid | Filter Rows: | Export: | Wrap Cell

Field	Type	Null	Key	Default	Extra
customername	varchar(50)	NO	PRI	NULL	
accno	int	NO	PRI	NULL	

desc loan;

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
loannumber	int	NO	PRI	NULL	
branchname	varchar(50)	YES	MUL	NULL	
amount	int	YES		NULL	

Inserting Values to the table

insert into branch

values(SBI-chamrajpet,banglore, 50000),

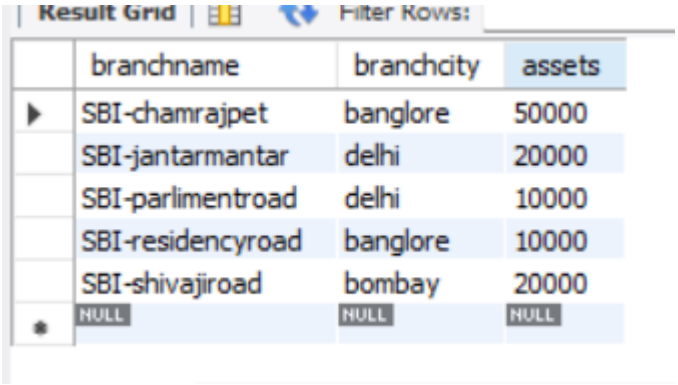
(SBI-residencyroad,banglore,10000),

(SBI-shivajiroad,bombay,20000),

(SBI-parlimentroad,delhi,10000),

(SBI-jantarmantar,delhi,20000);

select * from branch;



insert into bankaccount

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

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

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

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

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

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

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

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

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

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

select * from bankaccount;

Result Grid | Filter Rows:

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

```
insert into bankcustomer
values('avinash','bull-temple-road','banglore'),
('dinesh','bannergatta-road','banglore'),
('mohan','nationalcollege-road','banglore'),
('nikil','akbar-road','delhi'),
('ravi','prithviraj-road','delhi');
select * from bankcustomer;
```

Result Grid | Filter Rows:

customername	customer_street	city
avinash	bull-temple-road	banglore
dinesh	bannergatta-road	banglore
mohan	nationalcollege-road	banglore
nikil	akbar-road	delhi
ravi	prithviraj-road	delhi
NULL	NULL	NULL

```
insert into depositer
values('avinash',1),
```

('dinesh',2),
('nikil',4),
('ravi',5),
('avinash',8),
('nikil',9),
('dinesh',10),
('nikil',11);
select * from depositer;

Result Grid | Filter Rows:

customername	accno
avinash	1
dinesh	2
nikil	4
ravi	5
avinash	8
nikil	9
dinesh	10
nikil	11
NULL	NULL

insert into loan
values(1,'SBI-chamrajpet',1000),
(2,'SBI-residencyroad',2000),
(3,'SBI-shivajiroad',3000),
(4,'SBI-parlimentroad',4000),
(5,'SBI-jantarmantar',5000);
select * from loan;

Result Grid | Filter Rows:

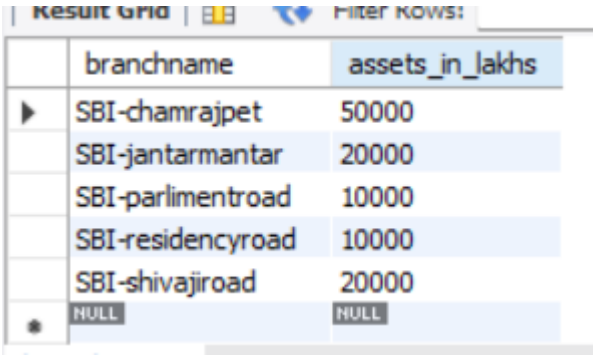
loannumber	branchname	amount
1	SBI-chamrajpet	1000
2	SBI-residencyroad	2000
3	SBI-shivajiroad	3000
4	SBI-parlimentroad	4000
5	SBI-jantarmantar	5000
NULL	NULL	NULL

loan 11

Queries

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

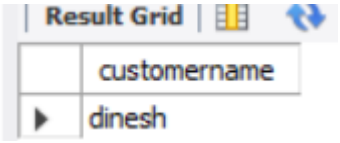
```
select branchname,assets as assets_in_lakhs

from branch;
```



branchname	assets_in_lakhs
SBI-chamrajpet	50000
SBI-jantarmantar	20000
SBI-parlimentroad	10000
SBI-residencyroad	10000
SBI-shivajiroad	20000
NULL	NULL

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

```
select d.customername

from bankaccount b, depositer d

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

group by customername

having count(*)=2;
```



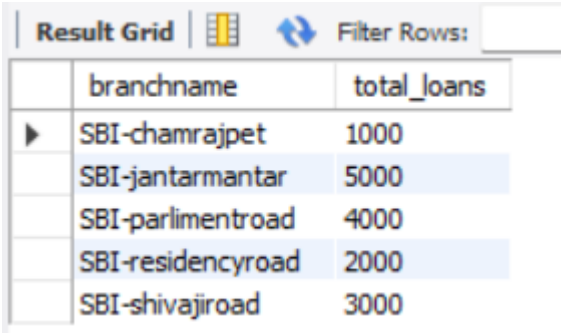
Result Grid

customername
dinesh

3.CREATE A VIEW WHICH GIVES EACH BRANCH THE SUM OF THE AMOUNT OF ALL THE LOANS AT THE BRANCH.

```
create view branch_loan_summary AS

select branchname, SUM(amount) AS total_loans

from loan

GROUP BY branchname;
```

select * from branch_loan_summary;



Result Grid | Filter Rows:

branchname	total_loans
SBI-chamrajpet	1000
SBI-jantarmantar	5000
SBI-parlimentroad	4000
SBI-residencyroad	2000
SBI-shivajiroad	3000

More Queries on Bank Database

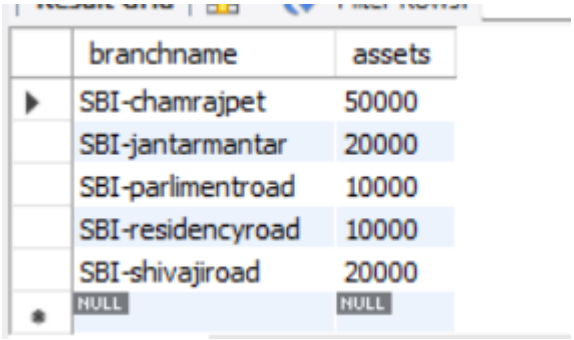
Week 04

Queries

1. Retrieve all branches and their respective total assets.

select branchname,assets

from branch;



branchname	assets
SBI-chamrajpet	50000
SBI-jantarmantar	20000
SBI-parlimentroad	10000
SBI-residencyroad	10000
SBI-shivajiroad	20000
NULL	NULL

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

select customername

from bankcustomer

where city='banglore';

Result Grid

customername
avinash
dinesh
mohan
NULL

3. List all customers with their account numbers

3. List all customers with their loan amounts

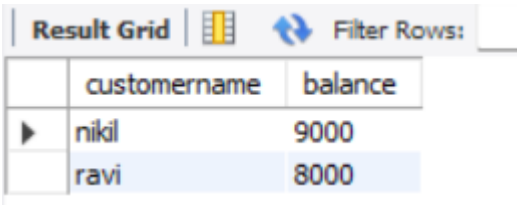
4. Find all the customers who have an account at all the branches located in a specific city (Ex. Delhi).

```
select c.customername

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

where c.customername=d.customername and

d.accno=a.accno and

a.branchname=b.branchname and

b.branchname=all(select b.branchname from branch

where b.branchcity='delhi');
```

customername
avinash
avinash
dinesh
dinesh
nikil
nikil

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

```
select c.customername,b.balance

from bankcustomer c,bankaccount b,depositer d

where d.accno=b.accno and c.customername=d.customername

and b.balance>5000;
```

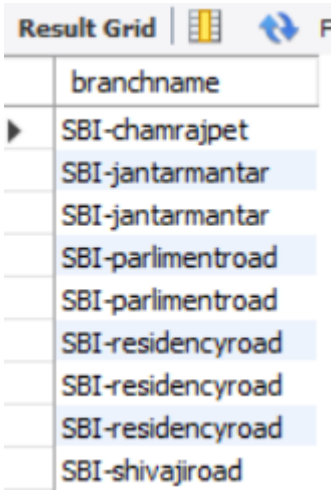


Result Grid | Filter Rows:

	customername	balance
▶	nikil	9000
	ravi	8000

6. List all customers who have both a loan and an account at the same branch.

```
select b.branchname

from branch b,bankaccount a,loan l where b.branchname=a.branchname and
b.branchname=l.branchname;
```



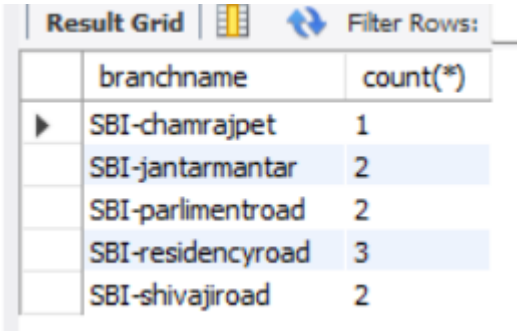
Result Grid

branchname
▶ SBI-chamrajpet
SBI-jantarmantar
SBI-jantarmantar
SBI-parlimentroad
SBI-parlimentroad
SBI-residencyroad
SBI-residencyroad
SBI-residencyroad
SBI-shivajiroad

7. Get the number of accounts held at each branch

```
select branchname,count(*)

from bankaccount
```

group by branchname;



Result Grid | Filter Rows:

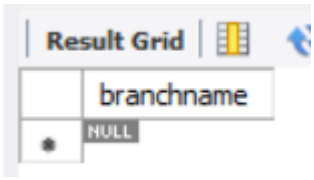
branchname	count(*)
SBI-chamrajpet	1
SBI-jantarmantar	2
SBI-parlimentroad	2
SBI-residencyroad	3
SBI-shivajiroad	2

8. Find all branches that have no loans issued

select b.branchname

from branch b

where b.branchname not in(select branchname from loan);



Result Grid

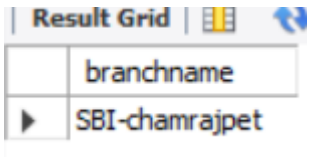
branchname
NULL

9. Retrieve the branch with the smallest total loan amount

select branchname

from loan

where amount=(select min(amount) from loan);



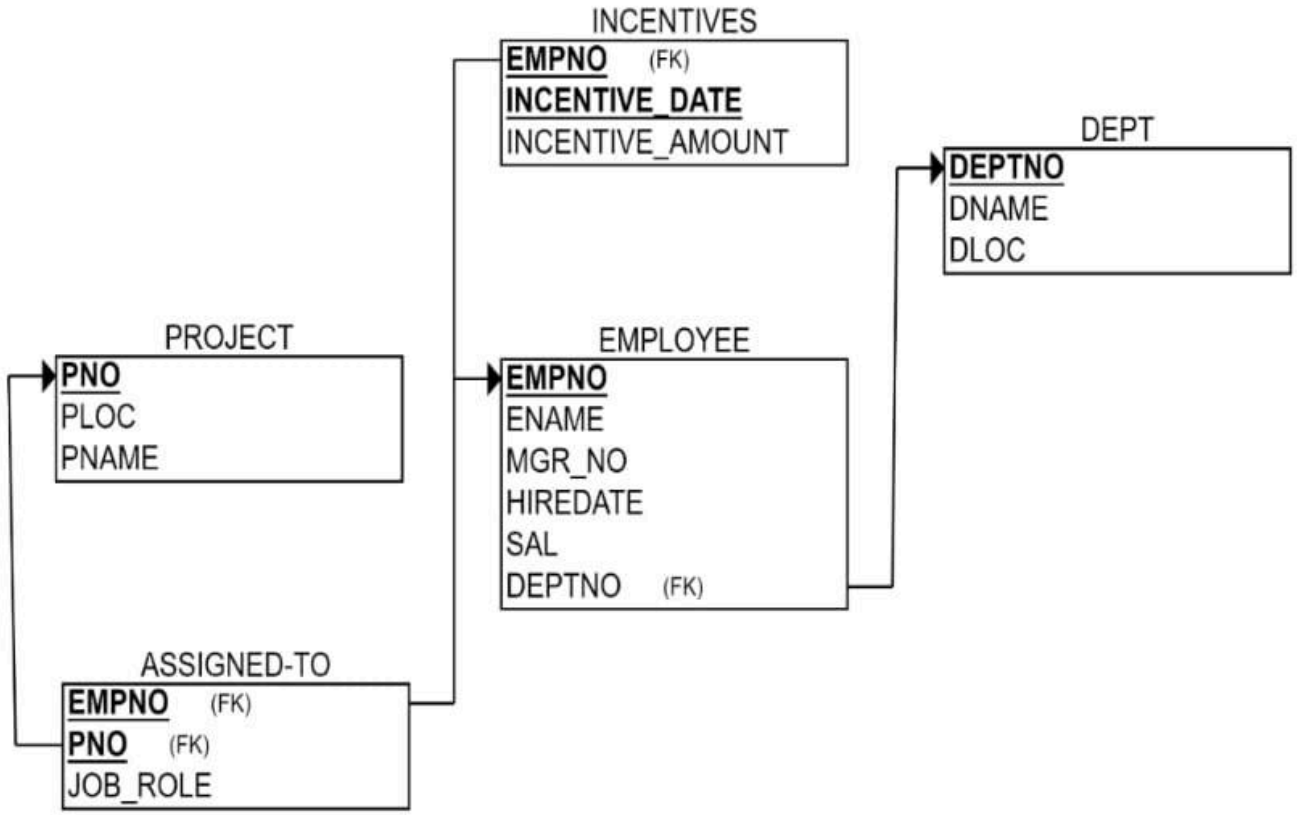
Result Grid

branchname
SBI-chamrajpet

Employee Database

(week 05)

Schema Diagram



Question

Incentives (empno, incentive_date,incentive_amount)

project (pno,ploc,pname)

employee(empno,ename,mgr_no,hiredate,sal,deptno)

dept(deptno,dname,dloc)

assigned-to(empno,pno,job_role)

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

2. Enter greater than five tuples for each table.

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

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

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

Create Database

```
create database bhoomi066;

use bhoomi066;
```

Create Table

```
create table project(

pno int,

ploc varchar(50),

pname varchar(50),

primary key (pno));

create table dept(

deptno int primary key,

dname varchar(50),

dloc varchar(50));

primary key(deptno));

create table employee(
```

```
empno int,
empname varchar(50),
mgr_no int,
hiredate date,
sal int,
deptno int,
primary key(empno),
foreign key (deptno) references dept (deptno));

create table incentives(
empno int,
incentive_date date,
incentive_amt int,
primary key(empno,incentive_date),
foreign key (empno) references employee (empno));
create table assigned_to(
empno int,
pno int,
job_role varchar (50),
primary key (empno, pno),
foreign key (empno) references employee(empno),
foreign key (pno) references project (pno));
```

Struc Table

desc project;

Result Grid | Filter Rows: | Export:

Field	Type	Null	Key	Default	Extra
pno	int	NO	PRI	NULL	
ploc	varchar(50)	YES		NULL	
pname	varchar(50)	YES		NULL	

desc dept;

Result Grid | Filter Rows: | Export: | Wra

Field	Type	Null	Key	Default	Extra
deptno	int	NO	PRI	NULL	
dname	varchar(50)	YES		NULL	
dloc	varchar(50)	YES		NULL	

desc employee;

Field	Type	Null	Key	Default	Extra
empno	int	NO	PRI	NULL	
empname	varchar(50)	YES		NULL	
mgr_no	int	YES		NULL	
hiredate	date	YES		NULL	
sal	int	YES		NULL	
deptno	int	YES	MUL	NULL	

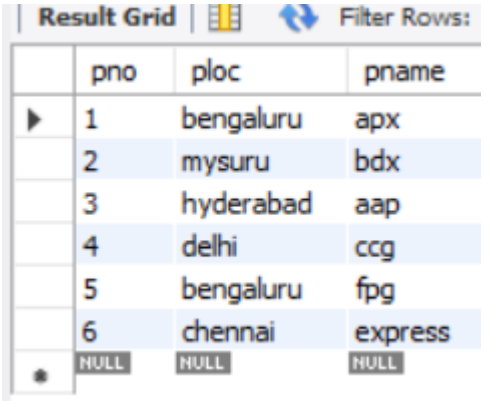
desc incentives;

Result Grid | Filter Rows: | Export: | Wrap

Field	Type	Null	Key	Default	Extra
empno	int	NO	PRI	NULL	
incentive_date	date	NO	PRI	NULL	
incentive_amt	int	YES		NULL	

Inserting Values to Table

```
insert into project
values(1,'bengaluru','apx'),
(2,'mysuru','bdx'),
(3,'hyderabad','aap'),
(4,'delhi','ccg'),
(5,'bengaluru','fpg'),
(6,'chennai','express');
select*from project;
```



Result Grid | Filter Rows:

	pno	ploc	pname
▶	1	bengaluru	apx
	2	mysuru	bdx
	3	hyderabad	aap
	4	delhi	ccg
	5	bengaluru	fpg
	6	chennai	express
*	NULL	NULL	NULL

```
insert into dept
values(1,'cse','bengaluru'),
(2,'ise','mysuru'),
(3,'cse','delhi'),
(4,'ise','hyderabad'),
(5,'hr','bengaluru');
select * from dept;
```

Result Grid | Filter Rows:

deptno	dname	dloc
1	cse	bengaluru
2	ise	mysuru
3	cse	delhi
4	ise	hyderabad
5	hr	bengaluru
6	managers	mysuru
NULL	NULL	NULL

insert into employee

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

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

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

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

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

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

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

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

select*from employee;

Result Grid | Filter Rows: | Edit:

empno	empname	mgr_no	hiredate	sal	deptno
1	bhoomika	3	2015-04-19	15000	5
2	bhoomi	7	2016-07-20	70000	2
3	bhumi	NULL	2000-07-22	10000	5
4	bhavana	3	2028-10-02	10000	5
5	jyothi	7	2020-11-18	8000	2
6	deepthi	8	2024-07-03	7000	3
7	shriya	NULL	1998-01-01	80000	2
8	kavya	NULL	2010-10-10	50000	3

insert into incentives

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

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

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

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

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

select*from incentives;

empno	incentive_date	incentive_amt
2	2020-11-30	3000
4	2020-11-12	3000
7	2010-10-14	5000
7	2015-07-24	7000
8	2015-07-30	4000
NULL	NULL	NULL

insert into assigned_to

values(7,02,'manager'),

(8,01,'data analyst'),

(3,04,'worker'),

(2, 06,'worker'),

(4,05,'app developer'),

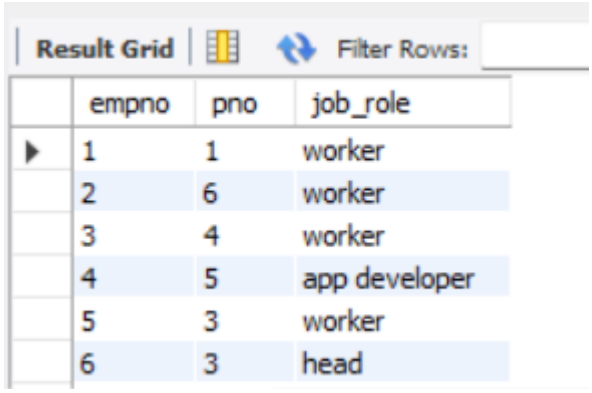
(8,04,'worker'),

(1,01,'worker'),

(5,03,'worker'),

(6,03,'head');

select*from assigned_to;



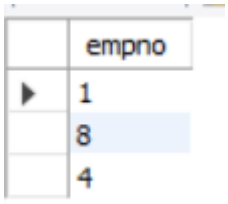
Result Grid | Filter Rows:

empno	pno	job_role
1	1	worker
2	6	worker
3	4	worker
4	5	app developer
5	3	worker
6	3	head

Queries

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

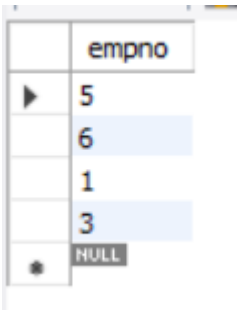
```
select e.empno

from employee e, assigned_to a

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

                                   from project

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



empno
1
8
4

2. Get Employee ID's of those employees who didn't receive incentives

```
select empno

from employee

where not exists(select 1

                 from incentives

                 where empno=employee.empno);
```



empno
5
6
1
3
NULL

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.

```
select e.empno, e.empname, d.deptno, a.job_role, d.dloc ,p.ploc

from employee e, project p,  assigned_to a, dept d

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

Result Grid | Filter Rows: | Export: | Wrap Cell Conten

empno	empname	deptno	job_role	dloc	ploc
1	bhoomika	5	worker	bengaluru	bengaluru
7	shriya	2	manager	mysuru	mysuru
8	kavya	3	worker	delhi	delhi
4	bhavana	5	app developer	bengaluru	bengaluru

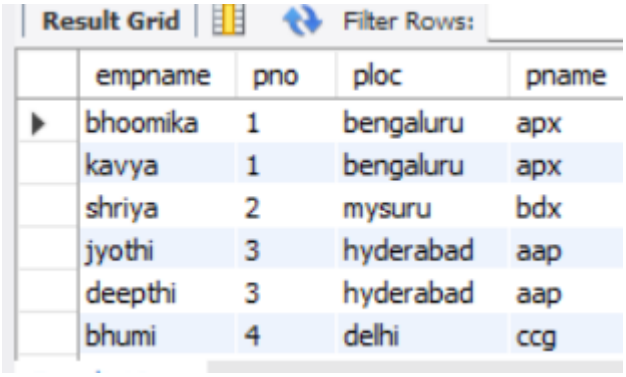
More Queries on Employee Database

Week 06

Queries

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

```
select e.empname,p.*
from employee e, project p, assigned_to a
where e.empno=a.empno and p.pno=a.pno;
```

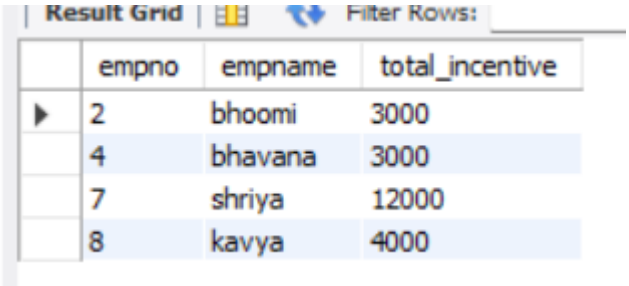


Result Grid | Filter Rows:

empname	pno	ploc	pname
bhoomika	1	bengaluru	apx
kavya	1	bengaluru	apx
shriya	2	mysuru	bdx
jyothi	3	hyderabad	aap
deepthi	3	hyderabad	aap
bhumi	4	delhi	ccg

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

```
select i.empno,e.empname,sum(i.incentive_amt)as total_incentive
from incentives i,employee e
where i.empno=e.empno
group by empno;
```

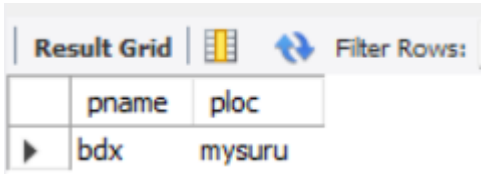


Result Grid | Filter Rows:

empno	empname	total_incentive
2	bhoomi	3000
4	bhavana	3000
7	shriya	12000
8	kavya	4000

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

```
select p.pname,p.ploc

from project p

where pno in(select pno from assigned_to a where job_role='manager');
```

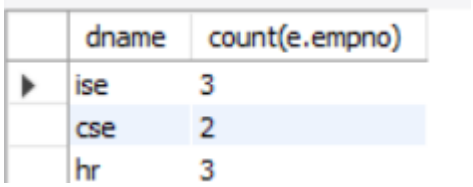


Result Grid | Filter Rows:

	pname	ploc
▶	bdx	mysuru

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

```
select d.dname,count(e.empno)

from dept d,employee e

where d.deptno=e.deptno

group by d.dname;
```



	dname	count(e.empno)
▶	ise	3
	cse	2
	hr	3

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

```
select e.empno,e.empname

from employee e

where not exists(select 1 from assigned_to a where e.empno=a.empno);
```

Result Grid | Filter Rows:

empno	empname
NULL	NULL

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

```
select e.empname,d.deptno,d.dloc
from employee e,dept d
where e.deptno=d.deptno;
```

empname	deptno	dloc
bhoomi	2	mysuru
jyothi	2	mysuru
shriya	2	mysuru
deepthi	3	delhi
kavya	3	delhi
bhoomika	5	bengaluru

Result 23

7. **Retrieve the details of employees who work under a specific manager (e.g., manager with empno = 101)**

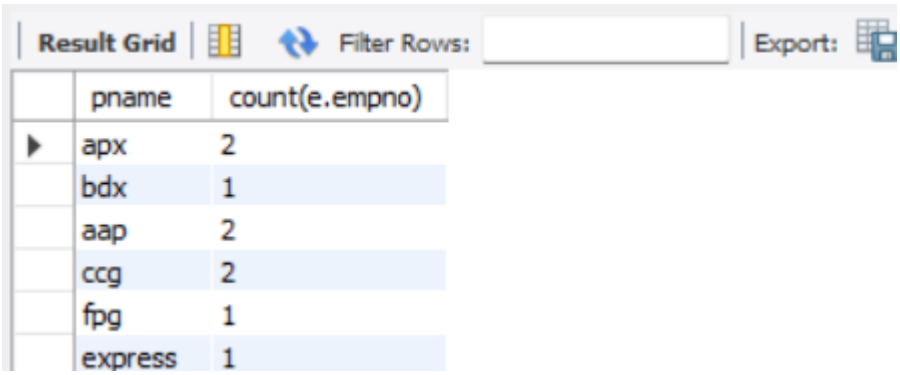
```
select *
from employee e
where mgr_no=7;
```

Result Grid | Filter Rows: | Edit:

empno	empname	mgr_no	hiredate	sal	deptno
2	bhoomi	7	2016-07-20	70000	2
5	jyothi	7	2020-11-18	8000	2
NULL	NULL	NULL	NULL	NULL	NULL

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

```
select p.pname,count(e.empno)
from project p,assigned_to e
where e.pno=p.pno
group by p.pname;
```



Result Grid | Filter Rows: | Export:

pname	count(e.empno)
apx	2
bdx	1
aap	2
ccg	2
fpg	1
express	1

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

```
SELECT e1.empname AS employee_name,
    e1.empno AS employee_number,
    e1.deptno,
    d.dname AS department_name,
    d.dloc AS department_location,
    e2.empname AS manager_name
FROM employee e1
JOIN employee e2 ON e1.mgr_no = e2.empno
JOIN dept d ON e1.deptno = d.deptno
ORDER BY e2.empname, e1.empname;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

employee_name	employee_number	deptno	department_name	department_location	manager_name
bhavana	4	5	hr	bengaluru	bhumi
bhoomika	1	5	hr	bengaluru	bhumi
deepthi	6	3	cse	delhi	kavya
bhoomi	2	2	ise	mysuru	shriya
jyothi	5	2	ise	mysuru	shriya

10. **List the total number of incentives given to each employee and the sum of incentives for each:**

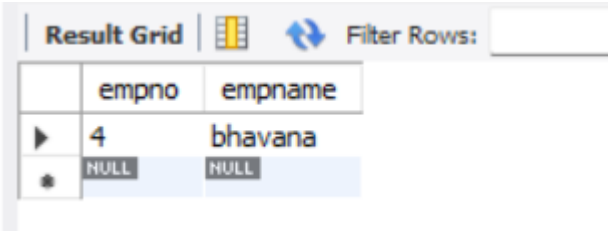
```
select empno,count(incentive_date)as number_of_times,sum(incentive_amt) as total_amt

from incentives

group by empno;
```

Result Grid | Filter Rows: | Export: | Wrap Cell

empno	number_of_times	total_amt
2	1	3000
4	1	3000
7	2	12000
8	1	4000

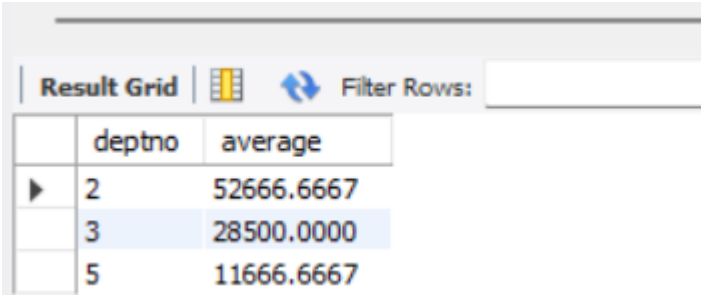
11. **Retrieve all employees who have the role of 'Developer' on any project:**

```
select e.empno,e.empname

from employee e

where e.empno in (select empno from assigned_to where empno=e.empno and job_role='app developer');
```



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

```
select deptno,avg(sal) as average
from employee
group by deptno;
```



WEEK 07

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.

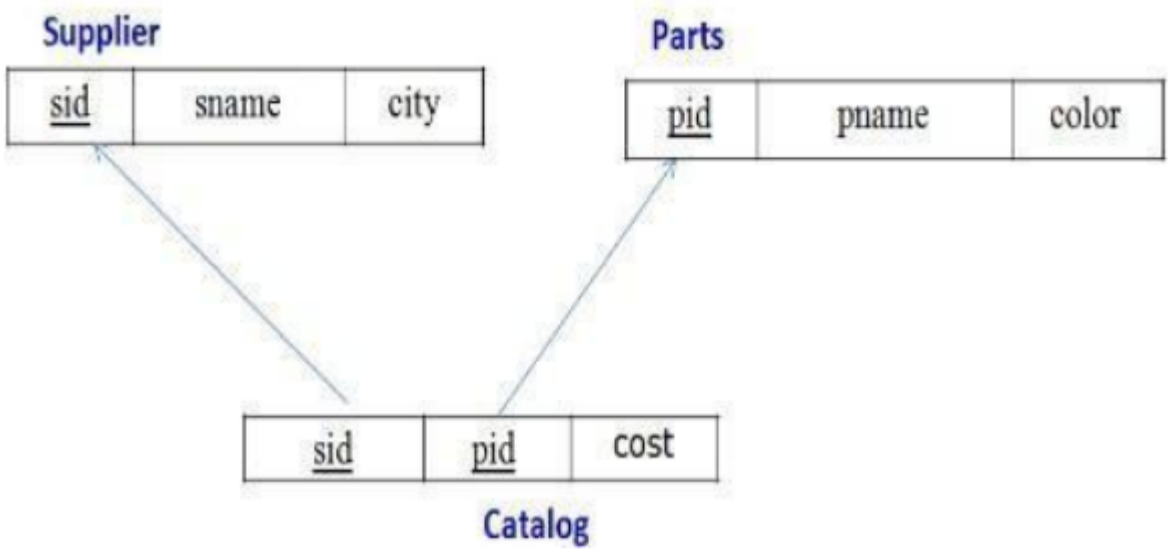
5. Find the snames of suppliers who supply every red part.

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;

use supplier_database;
```

Create Table

```
create table

Supplier (

SID int,

Sname varchar(20),

City varchar(20),

PRIMARY KEY(SID)

);


create table

Parts (

PID int,

Pname varchar(20),

Color varchar(20),

PRIMARY KEY(PID)

);


create table

Catalog (

SID int,

PID int,

Cost int,

PRIMARY KEY(SID,PID),
```

FOREIGN KEY(SID) references Supplier(SID),

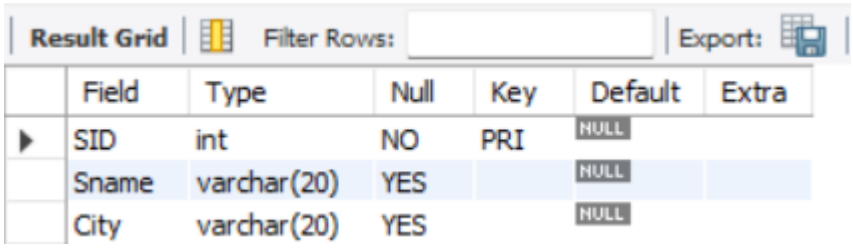
FOREIGN KEY(PID) references Parts(PID)

ON DELETE CASCADE ON UPDATE CASCADE

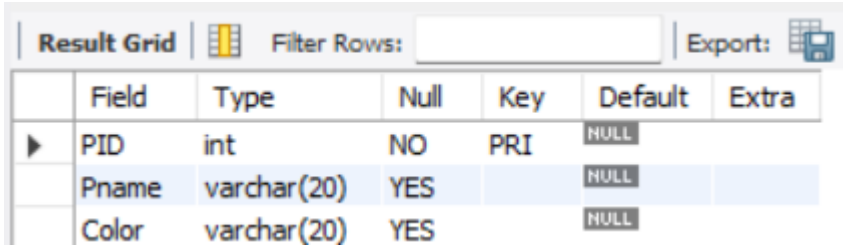
);

Structure of the Table

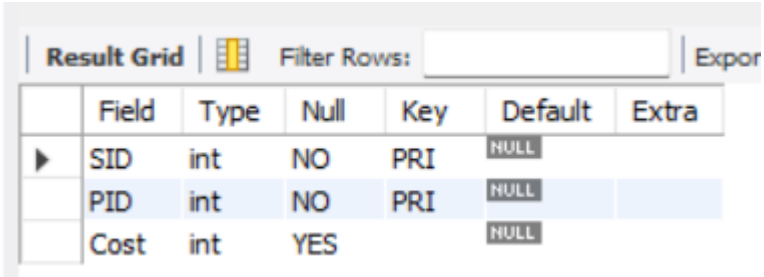
desc Supplier;



desc Parts;



desc Catalog;



Inserting values to the Table

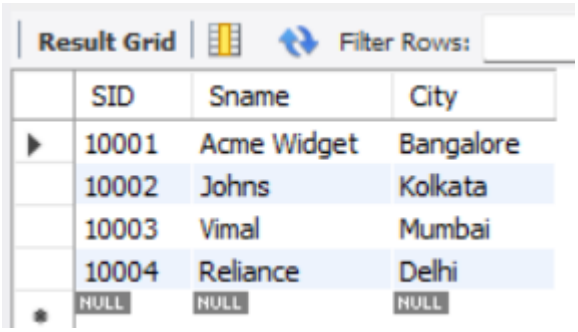
insert into Supplier values(10001,'Acme Widget','Bangalore');

insert into Supplier values(10002,'Johns','Kolkata');

insert into Supplier values(10003,'Vimal','Mumbai');

insert into Supplier values(10004,'Reliance','Delhi');

select * from Supplier;



Result Grid | Filter Rows:

SID	Sname	City
10001	Acme Widget	Bangalore
10002	Johns	Kolkata
10003	Vimal	Mumbai
10004	Reliance	Delhi
NULL	NULL	NULL

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

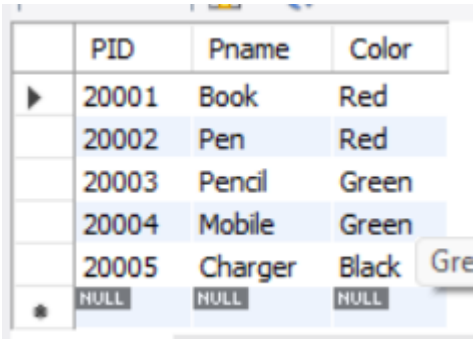
insert into Parts values(20002,'Pen','Red');

insert into Parts values(20003,'Pencil','Green');

insert into Parts values(20004,'Mobile','Green');

insert into Parts values(20005,'Charger','Black');

select * from Parts;



PID	Pname	Color
20001	Book	Red
20002	Pen	Red
20003	Pencil	Green
20004	Mobile	Green
20005	Charger	Black
NULL	NULL	NULL

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);

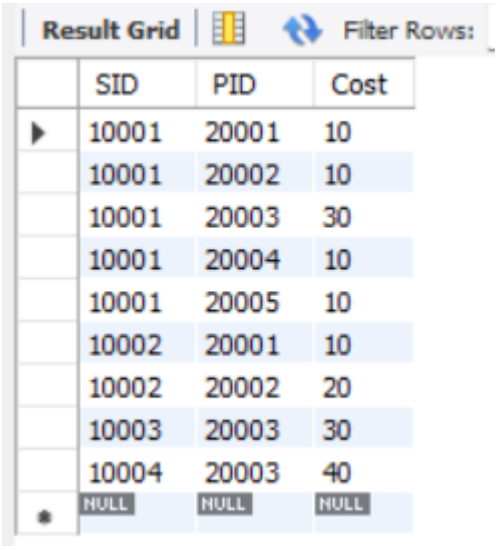
insert into Catalog values(10002,20001,10);

insert into Catalog values(10002,20002,20);

insert into Catalog values(10003,20003,30);

insert into Catalog values(10004,20003,40);

select * from Catalog;

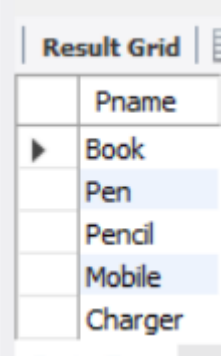


SID	PID	Cost
10001	20001	10
10001	20002	10
10001	20003	30
10001	20004	10
10001	20005	10
10002	20001	10
10002	20002	20
10003	20003	30
10004	20003	40
NULL	NULL	NULL

Queries

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

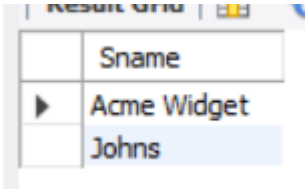
select distinct Pname from Parts where PID in(select PID from Catalog);



Pname
Book
Pen
Pencil
Mobile
Charger

Find the snames of suppliers who supply every part.

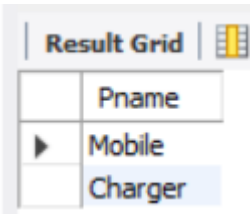
```
select Sname from Supplier where
SID NOT IN( select s.SID from Supplier s , Parts p
where p.Color='Red' and p.PID NOT IN(select c.PID from Catalog c where c.SID=s.SID));
```



Sname
Acme Widget
Johns

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 Supplier s ON c.SID = s.SID
WHERE s.Sname = 'Acme Widget' AND NOT EXISTS (
SELECT 1 FROM Catalog c1
JOIN Supplier s1 ON c1.SID = s1.SID
WHERE c1.PID = p.PID
AND s1.Sname != 'Acme Widget'
);
```



Result Grid

Pname
Mobile
Charger

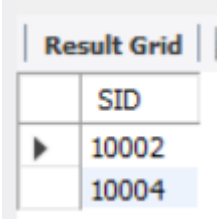
Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over all the suppliers who supply that part).

```
select distinct c.SID from Catalog c join
```

(select PID,avg(Cost) as Avg_Cost from Catalog group by

PID) avg_cost_table on c.PID=avg_Cost_table.PID

where c.Cost>avg_Cost_table.Avg_Cost;



Result Grid

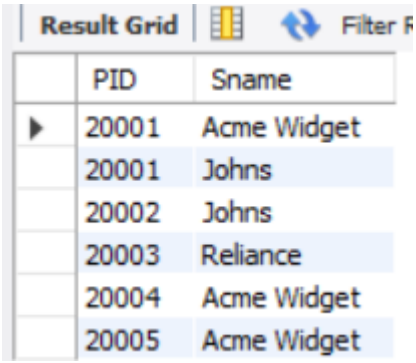
	SID
▶	10002
	10004

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

select p.PID,s.Sname from Supplier s join Catalog c on

s.SID=c.SID join Parts p on c.PID=p.PID

where c.Cost=(select max(c2.Cost) from Catalog c2 where c2.PID=p.PID);



Result Grid Filter

	PID	Sname
▶	20001	Acme Widget
	20001	Johns
	20002	Johns
	20003	Reliance
	20004	Acme Widget
	20005	Acme Widget

NoSQL Student Database

Question

(week 08)

Perform the following DB operations using MongoDB.

1. Create a database “Student” with the following

attributesRollno, 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.

Queries

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

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

```
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>
```

2. 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-okge9d-shard-0 [primary] test> db.Student.insert({RollNo:1,Age:21,Cont:9876,email:"antara.de9@gmail.com"});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6746b7a60ffbfb92d32f8e1a") }
}
Atlas atlas-okge9d-shard-0 [primary] test> db.Student.insert({RollNo:2,Age:22,Cont:9976,email:"anushka.de9@gmail.com"});

{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6746b7fb0ffbfb92d32f8e1b") }
}
Atlas atlas-okge9d-shard-0 [primary] test> db.Student.insert({RollNo:3,Age:21,Cont:5576,email:"anubhav.de9@gmail.com"});

{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6746b8060ffbfb92d32f8e1c") }
}
Atlas atlas-okge9d-shard-0 [primary] test> db.Student.insert({RollNo:4,Age:20,Cont:4476,email:"pani.de9@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6746b8110ffbfb92d32f8e1d") }
}
Atlas atlas-okge9d-shard-0 [primary] test> db.Student.insert({RollNo:10,Age:23,Cont:2276,email:"rekha.de9@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6746b8180ffbfb92d32f8e1e") }
}
```

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

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

```
Atlas atlas-okge9d-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
}
```

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

db.Student.insert({RollNo:11,Age:22,Name:"ABC",Cont:2276,email:"rea.de9@gmail.com"});
db.Student.update({RollNo:11,Name:"ABC"},{$set:{Name:"FEM"}})

```
Atlas atlas-okge9d-shard-0 [primary] test> db.Student.update({RollNo:11,Name:"ABC"},{$set:{Name:"FEM"}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

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

	_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 Customer Database

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

QUERIES

1. Create a collection by name Customers with the following attributes. Cust_id, Acc_Bal, Acc_Type

db.createCollection("Customer");

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")
  }
```

2. Write a query to display those records whose total account balance is greater than 12000 of account type 'Z' for each customer_id.

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'
  }
]
```

3. Determine Minimum and Maximum account balance for each customer_id.

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 }
]
```

4. Export the created collection into local file system

5. Drop the table

db.Customer.drop();

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

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

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 Restaurant Database

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'.

Queries

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

db.createCollection("restaurants");

{ 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.createCollection("restaurants");
{ ok: 1 }
```

```
Atlas atlas-zkql5l-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")
  }
}
```

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

db.restaurants.find({})

```
Atlas atlas-zkql5l-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' }
```

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

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

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.

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

```
Atlas atlas-zkql5i-shard-0 [primary] test> db.restaurants.find({ "score": { $lte: 10 } }, { _id: 1, name: 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'
  }
]
```

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

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

```
Atlas atlas-zkql5i-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 }
]
```

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

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

```
Atlas atlas-zkql5i-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' } }
]
```

	_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