VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Database Management Systems (23CS3PCDBM)

Submitted by

HITESH SHARMA (1BM23CS114)

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 **HITESH SHARMA (1BM23CS114),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a Database Management Systems (23CS3PCDBM) work prescribed for the said degree.

Prof Sheetal V A Assistant Professor Department of CSE, BMSCE	Dr. Kavitha Sooda Professor HOD Department of CSE, BMSCE

Index

Insurance Database

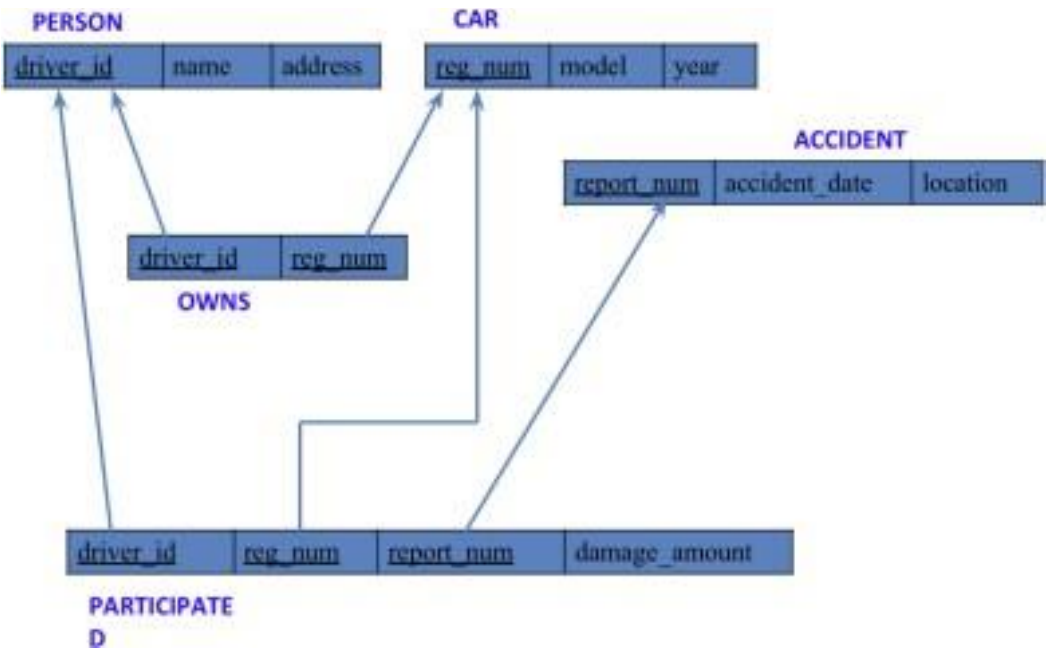
Question

(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
- Update the damage amount to 25000 for the car with a specific reg_num (example 'K A053408') for which the accident report number was 12.
- Add a new accident to the database.
- To Do
- Display Accident date and location

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

Schema Diagram



Create database

```
create database Hitesh_114;
use Hitesh_114;
```

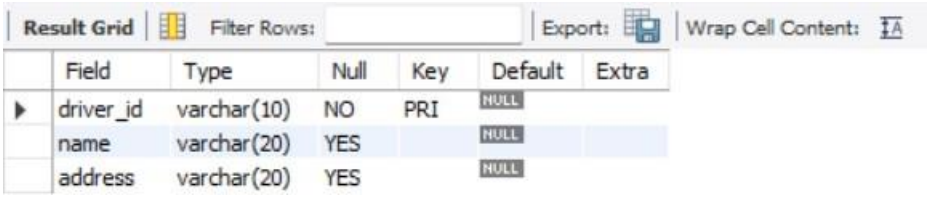
Create table

```
create table person (
driver_id varchar(10),
name varchar(20),
address varchar(20),
primary key(driver_id)
);

create table car (
reg_num varchar(10),
model varchar(10),
year int,
primary key (reg_num)
);

create table accident(
report_num int,
accident_date date,
location varchar(20),
primary key (report_num)
);
```

```
create table owns(
driver_id varchar(10),
reg_num varchar(10),
primary key (driver_id , reg_num),
foreign key(driver_id) references person(driver_id),
foreign key(reg_num) references car(reg_num)
);

create table participated (
driver_id varchar(10),
reg_num varchar(10),
report_num int,
damage_amount int,
primary key (driver_id, reg_num, report_num),
foreign key (driver_id) references person(driver_id),
foreign key (reg_num) references car(reg_num),
foreign key (report_num) references accident(report_num)
);
```

Structure of the table

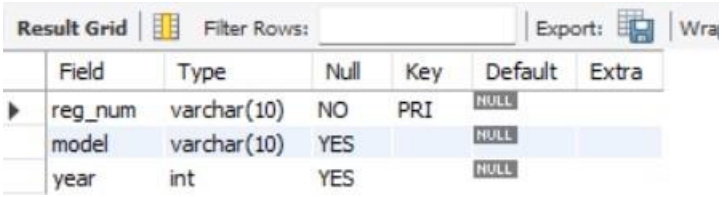
desc person;



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

Field	Type	Null	Key	Default	Extra
driver_id	varchar(10)	NO	PRI	NULL	
name	varchar(20)	YES		NULL	
address	varchar(20)	YES		NULL	

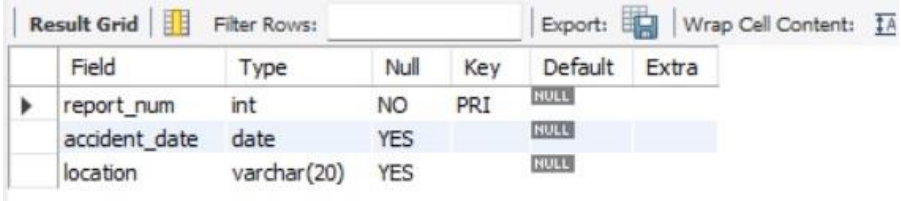
desc car;



Result Grid | Filter Rows: | Export: | Wra

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

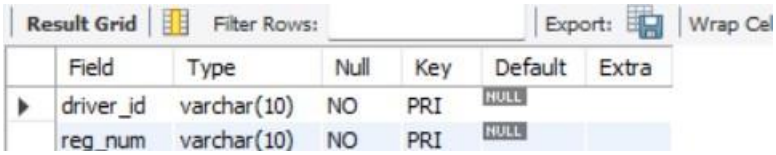
desc accident;



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	
location	varchar(20)	YES		NULL	

desc owns;



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

Field	Type	Null	Key	Default	Extra
driver_id	varchar(10)	NO	PRI	NULL	
reg_num	varchar(10)	NO	PRI	NULL	

desc participated;

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

Inserting Values to the table

insert into person
values(‘a01’,’richard’,’srinivasnagar’),
(‘a02’,’pradeep’,’rajajinagar’),
(‘a03’,’smith’,’ ashoknagar’),
(‘a04’,’venu’,’NRcolony’),
(‘a05’,’jhon’,’hanumanthnagar’);

driver_id	name	address
a01	richard	srinivasnagar
a02	pradeep	rajajinagar
a03	smith	ashoknagar
a04	venu	NRcolony
a05	jhon	hanumanthnagar

insert into car
values ('KA052250', 'indica',1990),
('KA031181','lancer',1957),
('KA095477','toyota',1998),
('KA053408','honda',2008),
('KA041702','audi',2005);

reg_num	model	year
KA031181	lancer	1957
KA041702	audi	2005
KA052250	indica	1990
KA053408	honda	2008
KA095477	toyota	1998

insert into accident
values(11,'2001-01-03','mysoreroad'),
(12,'2002-02-04','southendcircle'),

(13,'2021-01-03','bulltempleroad'),
(14,'2017-02-08','mysoreroad)'),
(15,'2004-03-05','kanakpuraroad');

report_num	accident_date	location
11	2001-01-03	mysoreroad
12	2002-02-04	southendcircle
13	2021-01-03	bulltempleroad
14	2017-02-08	mysoreroad)
15	2004-03-05	kanakpuraroad

insert into owns
values('a01','KA052250'),
('a02','KA053408'),
('a03','KA031181'),
('a04','KA095477'),
('a05','KA041702');

driver_id	reg_num
a03	KA031181
a05	KA041702
a01	KA052250
a02	KA053408
a04	KA095477

insert into participated
values('a01','KA052250',11,10000),
('a02','KA053408',12,50000),
('a03','KA031181',13,25000),
('a04','KA095477',14,3000),
('a05','KA041702',15,5000);

driver_id	reg_num	report_num	damage_amount
a01	KA052250	11	10000
a02	KA053408	12	50000
a03	KA031181	13	25000
a04	KA095477	14	3000
a05	KA041702	15	5000

Queries

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

```
update participated set damage_amount=25000

where reg_num='KA053408' and report_num=12;
```

driver_id	reg_num	report_num	damage_amount
a01	KA052250	11	10000
a02	KA053408	12	25000
a03	KA031181	13	25000
a04	KA095477	14	3000
a05	KA041702	15	5000

- Add a new accident to the database.

```
insert into accident
values (16,'2015-03-08','domlur');
```

report_num	accident_date	location
11	2001-01-03	mysoreroad
12	2002-02-04	southendcircle
13	2021-01-03	bulltempleroad
14	2017-02-08	mysoreroad)
15	2004-03-05	kanakpuraroad

- Display Accident date and location

```
select accident_date ,location from accident;
```

accident_date	location
2001-01-03	mysoreroad
2002-02-04	southendcircle
2021-01-03	bulltempleroad
2017-02-08	mysoreroad)
2004-03-05	kanakpuraroad

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

```
select driver_id from participated
where damage_amount>=25000;
```

driver_id
a02
a03

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

Result Grid | Filter Rows:

count(report_num)
1

```
select count(driver_id)
from accident a,participated p
where a.report_num=p.report_num and a.accident_date like '__08%';
```

Result Grid | Filter Rows:

count(driver_id)
0

```
select avg(damage_amount) as 'average' from participated;
```

Result Grid

average
13600.0000

```
select name
from person a ,participated p
where a.driver_id=p.driver_id and p.damage_amount >(select avg(damage_amount) from participated);
```

Result Grid

name
pradeep
smith

```
select max(damage_amount) from participated;
```

Result Grid | Filter Rows:

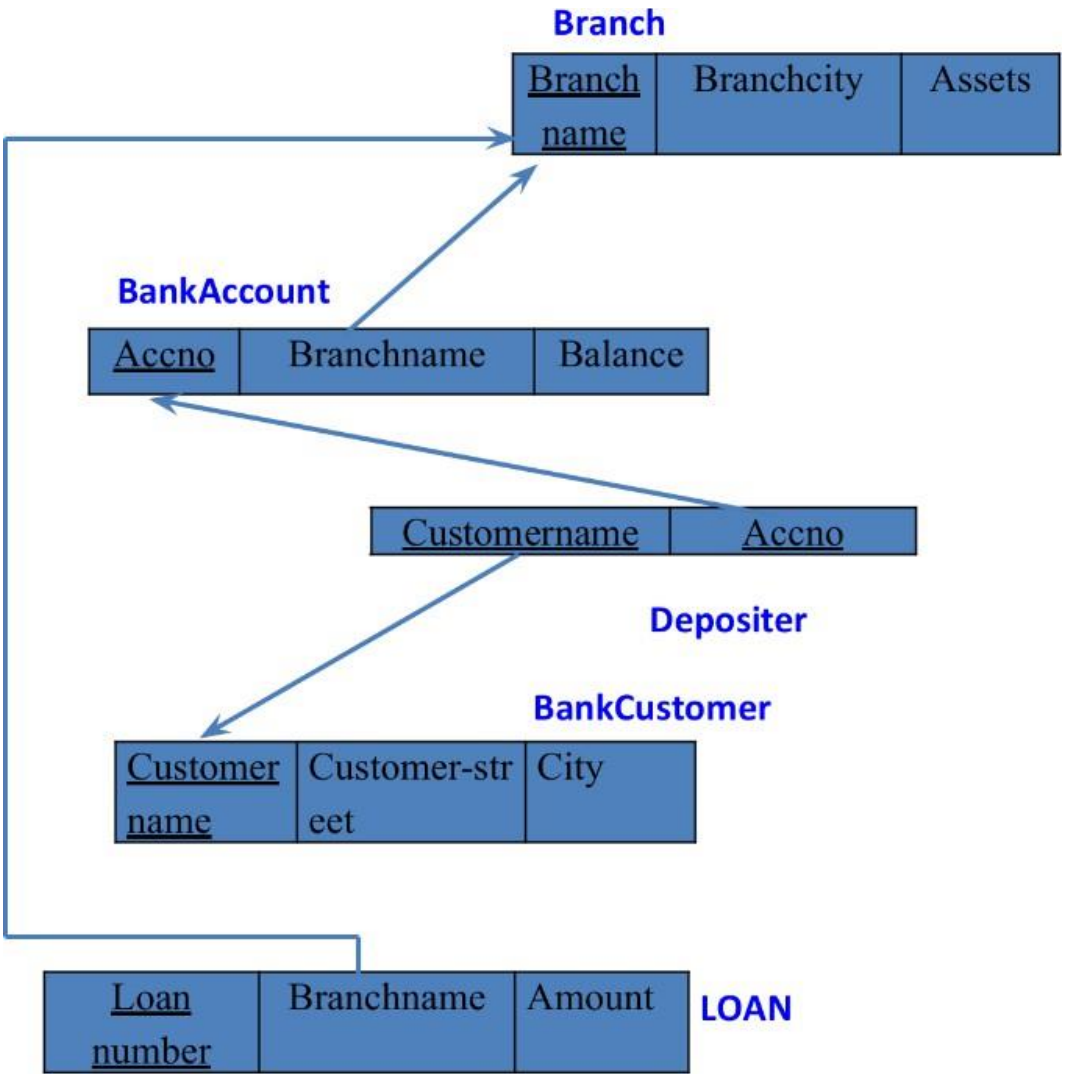
max(damage_amount)
25000

BANK ACCOUNT

QUESTION
(WEEK 3)

- Create the above tables by properly specifying the primary keys and the foreign keys.
- Enter at least five tuples for each relation.
- Display the branch name and assets from all branches in lakhs of rupees and rename The assets column to 'assets in lakhs'.
- Find all the customers who have at least two accounts at the same branch (ex.SBI_residencyroad).
- Create a view which gives each branch the sum of the amount of all the loans at the branch.

SCHEMA DIAGRAM


Branch
Branch name
Branchcity
Assets
BankAccount
Accno
Branchname
Balance
Customername
Accno
Depositer
BankCustomer
Customer name
Customer-street
City
Loan number
Branchname
Amount
LOAN

CREATE DATABASE

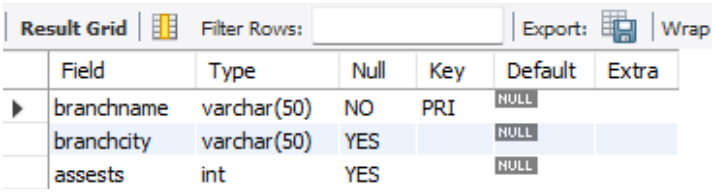
```
create database Hitesh_cs114;
use Hitesh_cs114;
```

CREATE TABLES

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

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

create table bankaccount (
accno int,
branchname varchar(50),
balance int,
primary key (accno),
foreign key (branchname) references branch (branchname));

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

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

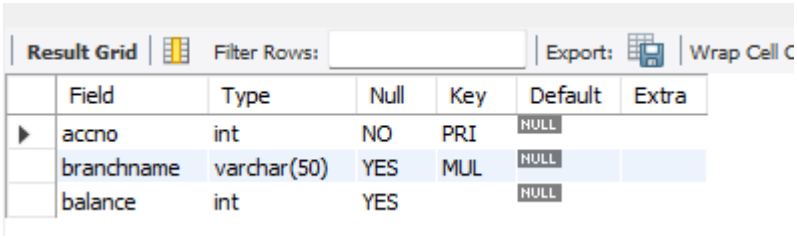
STRUCTURE OF TABLE

```
desc branch;
```



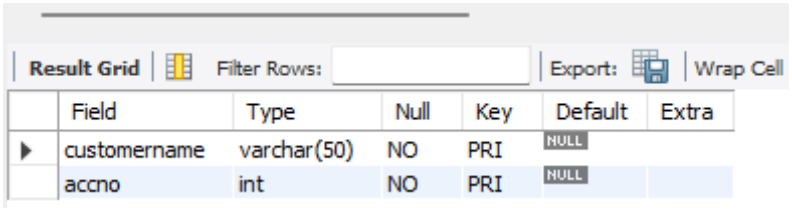
Field	Type	Null	Key	Default	Extra
branchname	varchar(50)	NO	PRI	NULL	
branchcity	varchar(50)	YES		NULL	
assests	int	YES		NULL	

desc bankaccount;



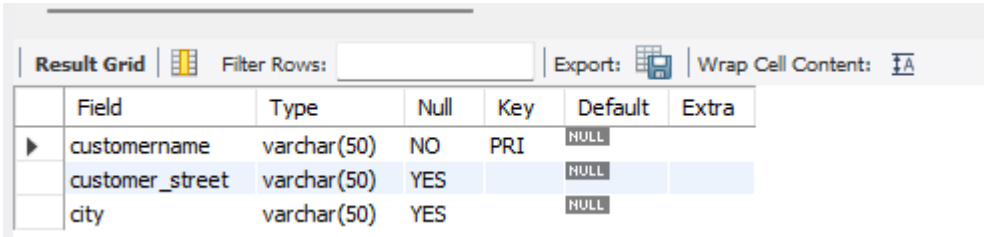
Field	Type	Null	Key	Default	Extra
accno	int	NO	PRI	NULL	
branchname	varchar(50)	YES	MUL	NULL	
balance	int	YES		NULL	

desc depositer;



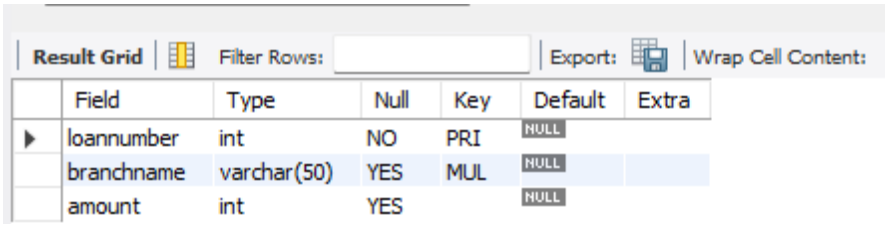
Field	Type	Null	Key	Default	Extra
customername	varchar(50)	NO	PRI	NULL	
accno	int	NO	PRI	NULL	

desc bankcustomer;



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

desc loan;



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

INSERTING VALUES INTO THE TABLE

insert into branch

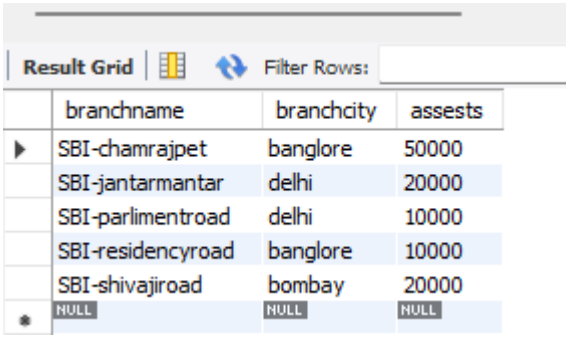
values('SBI-chamrajpet','banglore', 50000),

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

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

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

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



Result Grid | Filter Rows:

branchname	branchcity	assests
SBI-chamrajpet	banglore	50000
SBI-jantarmantar	delhi	20000
SBI-parlimentroad	delhi	10000
SBI-residencyroad	banglore	10000
SBI-shivajiroad	bombay	20000
NULL	NULL	NULL

insert into bankcustomer

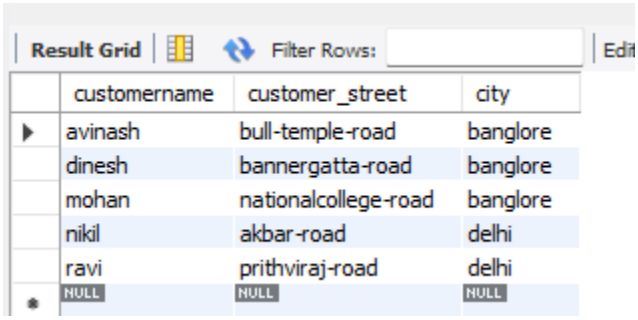
values('avinash','bull-temple-road','banglore'),

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

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

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

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



Result Grid | Filter Rows: | Edit

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 bankaccount

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

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

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

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

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

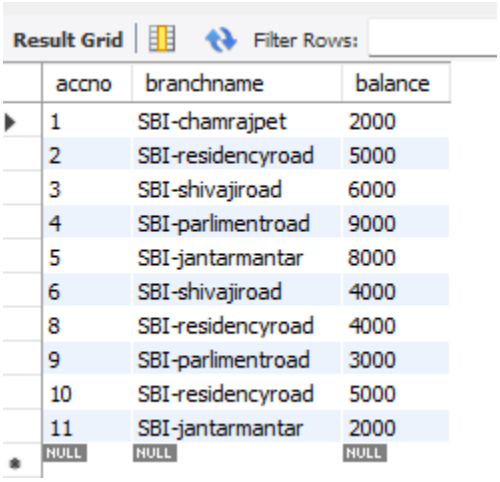
(6,'SBI-shivajiroad',4000),

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

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

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

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



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
9	SBI-parlimentroad	3000
10	SBI-residencyroad	5000
11	SBI-jantarmantar	2000
NULL	NULL	NULL

insert into depositer

values('avinash',1),

('dinesh',2),

('nikil',4),

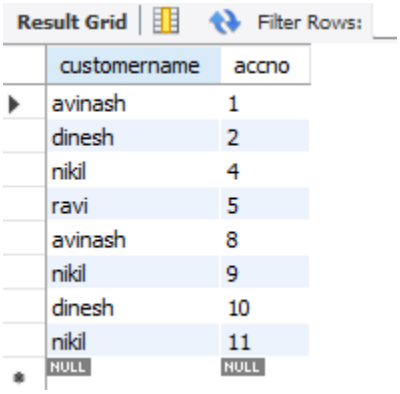
('ravi',5),

('avinash',8),

('nikil',9),

('dinesh',10),

('nikil',11);



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

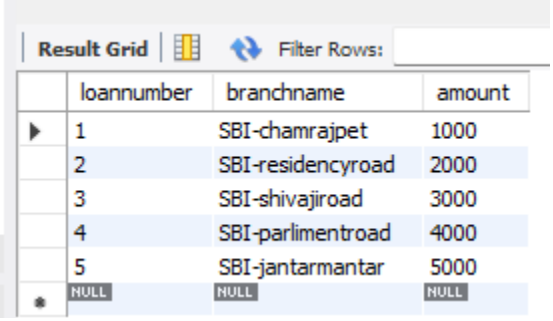
values(1,'SBI-chamrajpet',1000),

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

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

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

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



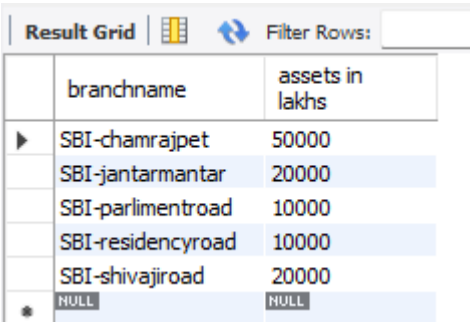
Result Grid | Filter Rows:

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

QUERIES

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

```
select branchname,assests as 'assets in lakhs'
from  branch;
```



Result Grid | Filter Rows:

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

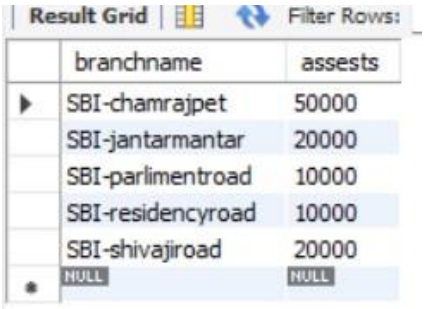
2. Find all the customers who have at least two accounts at the same branch (ex.SBI_ResidencyRoad).

```
select d.customername
from bankaccount b, depositer d
where b.accno=d.accno and branchname='SBI-residencyroad'
group by customername
having count(*)>=2;
```

WEEK -04

4. Retrieve all branches and their respective total assets

```
select branchname, assests
from branch;
```



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

5. List all customers who live in a particular city

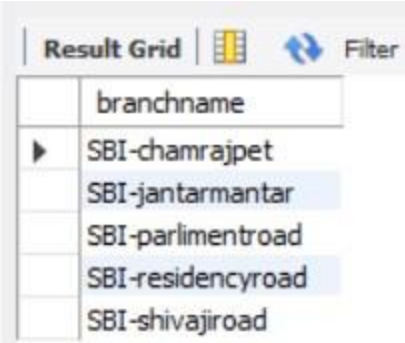
```
select customername
from bankcustomer
where city=”Delhi”;
```

customer_name
Nikil

6. List all customers with their account numbers

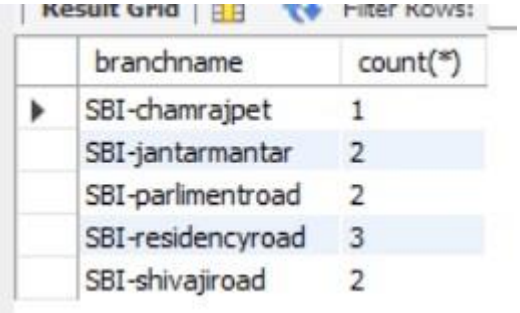
```
select customername ,accno
from depositer ;
```

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



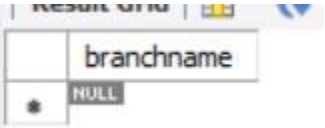
10. Get the number of accounts held at each branch

```
select branchname , count(*)
from bankaccount
group by branchname;
```



11. Find all branches that have no loans issued

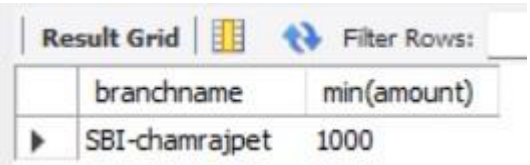
```
select b.branchname
from branch b
where b.branchname not in(select branchname
                          from loan);
```



12. Retrieve the branch with the smallest total loan amount

```
select branchname ,min(amount)
```

```
from loan
group by branchname
order by min(amount)
limit 1;
```



Result Grid | Filter Rows:

	branchname	min(amount)
▶	SBI-chamrajpet	1000

EMPLOYEE DATABASE

(WEEK -05)

QUESTION

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

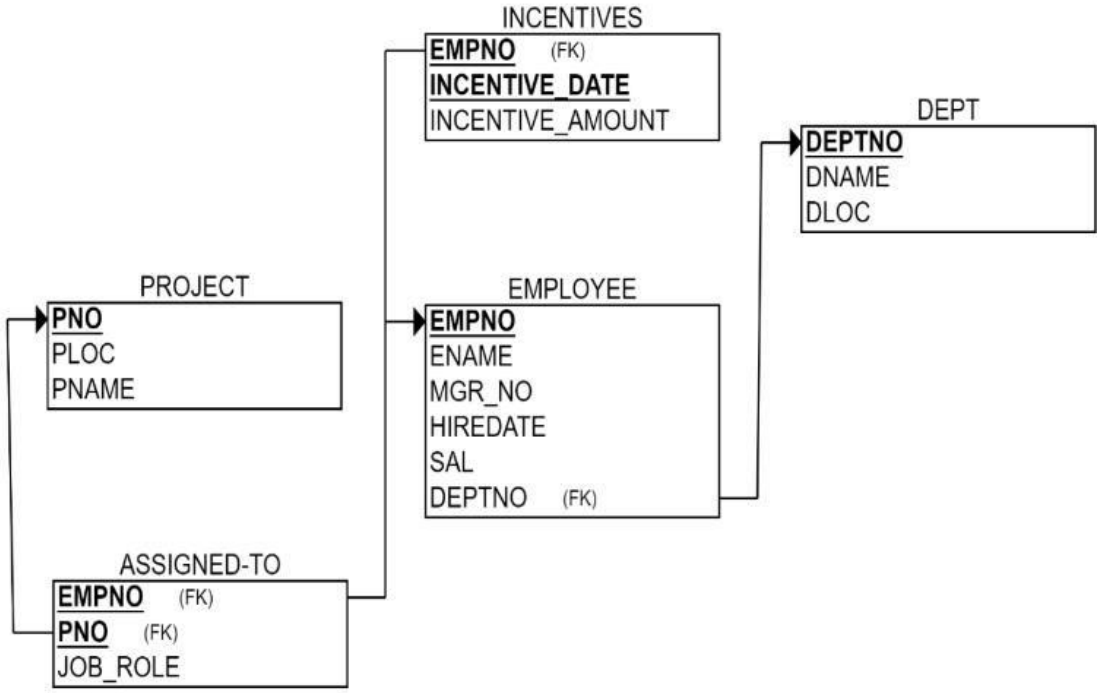
2. Enter greater than five tuples for each table.

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

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

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

Schema Diagram



CREATE DATABASE

Create database Hiteshcs114; Use database Hiteshcs114;

CREATE TABLES

```
create table project(
pno int,
ploc varchar(50),
pname varchar(50),
primary key (pno));

create table dept(
deptno int primary key,
dname varchar(50),
dloc varchar(50));

create table employee(
```

Result Grid | Filter Rows: | Export:

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

Desc employee;

Result Grid | Filter Rows: | Export:

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 assigned_to;

Result Grid | Filter Rows: | Export:

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

INSERT TO TABLE

insert into project
values(01,'bengaluru','pascalai'),
(02,'hyderbaad','p1m'),
(03,'mysuru','p2m'),
(04,'chennai','p3m'),
(05,'delhi','p4m');

pno	ploc	pname
1	bengaluru	pascalai
2	hyderbaad	p1m
3	mysuru	p2m
4	chennai	p3m
5	delhi	p4m

insert into dept

values(01,'sales','bengaluru'),

(02,'it','hyderabad'),

(03,'hr','mysuru'),

(04,'finance','delhi'),

(05,'production','chennai');

deptno	dname	dloc
1	sales	bengaluru
2	it	hyderabad
3	hr	mysuru
4	finance	delhi
5	production	chennai
NULL	NULL	NULL

insert into employee

values (101,"Dinesh",111,"2021-11-01",50000,1),

(102,"Dhanush",112,"2024-01-01",70000,2),

(103,"Daivya",113,"2024-01-01",80000,3),

(104,"Aditya",114,"2023-08-11",65000,1),

(105,"Arun",111,"2022-06-07",35000,2);

empno	empname	mgr_no	hiredate	sal	deptno
101	Dinesh	111	2021-11-01	50000	1
102	Dhanush	112	2024-01-01	70000	2
103	Daivya	113	2024-01-01	80000	3
104	Aditya	114	2023-08-11	65000	1
105	Arun	111	2022-06-07	35000	2
NULL	NULL	NULL	NULL	NULL	NULL

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);

empno
103
NULL

3. Write a SQL query to find the employees name, number, dept, job_role, department location and project location w
   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;

empno	empname	deptno	job_role	dloc	ploc
101	Dinesh	1	manager	bengaluru	bengaluru
103	Daivya	3	manager	mysuru	mysuru

(WEEK-06)

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

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

empname	pno	ploc	pname
Dinesh	1	bengaluru	pascalai
Dinesh	2	hyderbaad	p1m
Dhanush	2	hyderbaad	p1m
Daivya	3	mysuru	p2m
Aditya	5	delhi	p4m

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;
```

empno	empname	total_incentive
101	Dinesh	5000
102	Dhanush	2000
104	Aditya	5000
105	Arun	1000

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');
```

pname	ploc
pascalai	bengaluru
p2m	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;
```

```
from project p, assigned_to e
where e.pno=p.pno
group by p.pname;
```

pname	count(e.empno)
pascalai	1
p1m	2
p2m	1
p4m	1

9. 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;
```

empno	number_of_times	total_amt
101	2	5000
102	1	2000
104	1	5000
105	1	1000

10. 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='developer');
```

empno	empname
104	Aditya
NULL	NULL

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

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

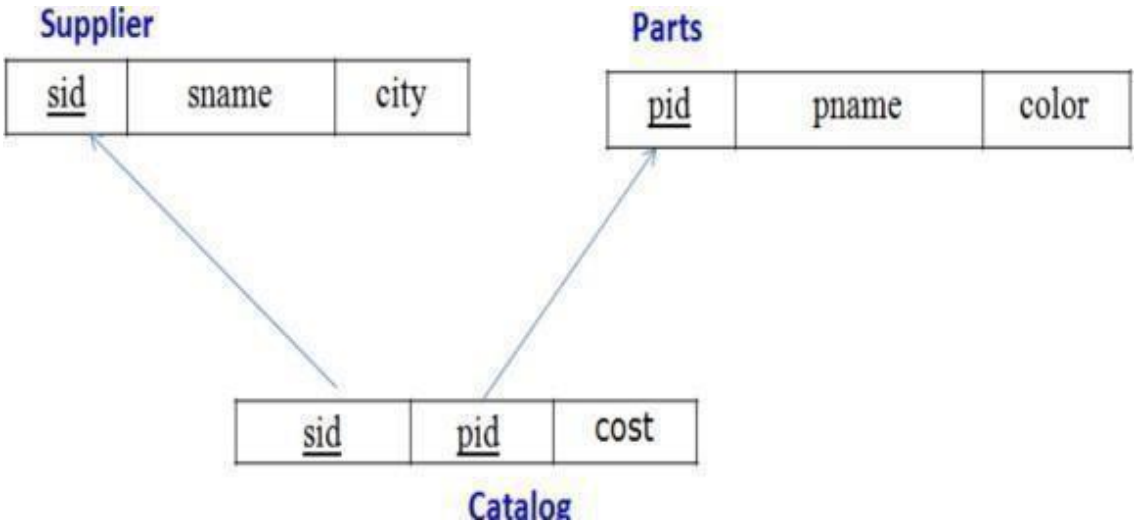
deptno	average
1	57500.0000
2	52500.0000
3	80000.0000

SUPPLIERS DATABASE

(WEEK -07)
QUESTION

1. Using Scheme diagram, Create tables by properly specifying the primary keys and the foreign keys.
2. Insert appropriate records in each table.
3. Find the pnames of parts for which there is some supplier.
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 supp;
use supp;
```

Create Tables:

```
create table Supplier(
 s_id int primary key,
s_name     varchar(30),     city
```

varchar(30)
create table Parts(p_id int
primary key, p_name
varchar(30), color
varchar(30));

create table Catalog(
s_id int,
p_id int,
cost float,
foreign key(s_id) references Supplier(s_id),
foreign key(p_id) references Parts(p_id));

Structure of the Table:

desc Supplier;

Field	Type	Null	Key	Default	Extra
s_id	int	NO	PRI	NULL	
s_name	varchar(30)	YES		NULL	
city	varchar(20)	YES		NULL	

desc Parts;

Field	Type	Null	Key	Default	Extra
p_id	int	NO	PRI	NULL	
p_name	varchar(30)	YES		NULL	
color	varchar(30)	YES		NULL	

desc Catalog;

Field	Type	Null	Key	Default	Extra
s_id	int	YES	MUL	NULL	
p_id	int	YES	MUL	NULL	
cost	float	YES		NULL	

Inserting Values to the tables:

insert into Supplier values
(10001, 'Acme_Widget', 'Bangalore'),
(10002, 'Johns', 'Kolkata'),
(10003, 'Vimal', 'Mumbai'),

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

s_id	s_name	city
10001	Acme_Widget	Bangalore
10002	Johns	Kolkata
10003	Vimal	Mumbai
10004	Reliance	Delhi
NULL	NULL	NULL

insert into Parts
values (20001, 'Book',
'Red'),
(20002, 'Pen', 'Red'),
(20003, 'Pencil', 'Green'),
(20004, 'Mobile', 'Green'),
(20005, 'Charger', 'Black');

p_id	p_name	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),
(10001, 20002, 10),
(10001, 20003, 30),
(10001, 20004, 10),
(10001, 20005, 10),
(10002, 20001, 10),
(10002, 20002, 20),
(10003, 20003, 30),
(10004, 20003, 40);

	s_id	p_id	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

Queries:

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

```
select distinct p.p_name
from Supplier s, Catalog c, Parts p where
s.s_id = c.s_id and
p.p_id = c.p_id and
c.s_id is not null;
```

	p_name
▶	Book
	Pen
	Pencil
	Mobile
	Charger

Find the snames of suppliers who supply every part.

```
select distinct s_name
from Supplier s, Catalog c, Parts p
where s.s_id = c.s_id
group by s.s_id, s.s_name
having count(distinct c.p_id)=(select count(*) from Parts p);
```

	s_name
▶	Acme_Widget

Find the snames of suppliers who supply every red part.

```
select distinct s_name
from Supplier s, Catalog c, Parts p
where s.s_id = c.s_id and
c.p_id in (select p_id from Parts p where p.color = 'Red')
```

	s_name
▶	Johns
	Acme_Widget

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

```
select distinct p_name from Supplier s, Parts p, Catalog c where
p.p_id in (select c.p_id from Catalog c, Supplier s where
s.s_id = c.s_id and s.s_name = 'Acme_Widget') and
p.p_id not in (select c.p_id from Catalog c, Supplier s where
s.s_id = c.s_id and s.s_name != 'Acme_Widget');
```

	p_name
▶	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)

```
create view Average(p_id, Average_Product_Cost) as
select c.p_id, avg(cost)
from Catalog c
group by
c.p_id;
 select c.s_id from Catalog c, Average a where
 c.p_id = a.p_id and
 c.cost>(a.Average_Product_Cost)
 group by c.p_id, c.s_id;
```

	s_id
▶	10002
	10004

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

```
select distinct s.s_name, c.cost, c.p_id from Catalog c, Supplier s where
s.s_id = c.s_id and
c.cost in (select max(cost) from Catalog c group by c.p_id);
```

sname
Acme Widget
Johns
Reliance

NO SQL STUDENT DATABASE

(WEEK -08)
QUESTION

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

Create Database:

db.createCollection("Student");

```
Atlas atlas-cci5oy-shard-0 [primary] test> db.createCollection("Student");
{ ok: 1 }
Atlas atlas-cci5oy-shard-0 [primary] test>
```

Inserting Values to the tables:

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

```
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("675fe28cf2355f925cc449c9") }
}
```

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

```
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("675fe295f2355f925cc449ca") }
}
```

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

```
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("675fe29df2355f925cc449cb") }
}
```

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

```
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("675fe2a5f2355f925cc449cc") }
}
```

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

```
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("675fe2abf2355f925cc449cd") }
}
```

Queries:

db.Student.find()

```
Atlas atlas-cci5oy-shard-0 [primary] test> db.Student.find()
[
  {
    _id: ObjectId("6746b3bd35240699686244 99"),
    RollNo: 1,
    Age: 21,
    Cont: 9876,
    email: 'antara.de9@gmail.com'
  },
  {
    _id: ObjectId("6746b3c7352406996862449a"),
    RollNo: 2,
    Age: 22,
    Cont: 9976,
    email: 'anushka.de9@gmail.com'
  },
  {
    _id: ObjectId("6746b3d035240699686244 9b"),
    RollNo: 3,
    Age: 21,
    Cont: 5576,
    email: 'anubhav.de9@gmail.com'
  },
  {
    _id: ObjectId("6746b3d835240699686244 9c"),
    RollNo: 4,
    Age: 20,
    Cont: 4476,
    email: 'pani.de9@gmail.com'
  },
  {
    _id: ObjectId("6746b3e135240699686244 9d"),
    RollNo: 10,
    Age: 23,
    Cont: 2276,
    email: 'Abhinav@gmail.com'
  },
```

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

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: 0,
  upsertedCount: 0
}
```

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. c om"});

```
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("675fe2cbf2355f925cc449ce") }
}
```

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

```
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
{
  _id: ObjectId("6746b41935240699686 2449e"),
  RollNo: 11,
  Age: 22,
  Name: 'FEM',
  Cont: 2276,
  email: 'rea.de9@gmail.com'
},
```

Import a given csv dataset from local file system into mongodb collectio

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

NO SQL CUSTOMERS DATABASE

(WEEK -09)

QUESTION

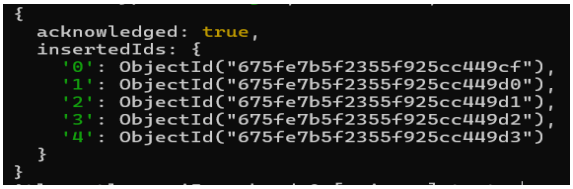
- Create a collection by name Customers with the following attributes. Cust_id, Acc_Bal, Acc_Type
- Insert at least 5 values into the table
- Write a query to display those records whose total account balance is greater than 1200 of account type ‘Z’ for each customer_id.
- Determine Minimum and Maximum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

Create Database:

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

Inserting Values to the tables:

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



Queries:

Write a query to display those records whose total account balance is greater than 1200 of account type 'Z' for each customer_id.

db.Customer.find({acc_bal: {$gt: 12000}, acc_type:"Checking"});

```
[
  {
    _id: ObjectId("675fe7b5f2355f925cc449d0"),
    custid: 1,
    acc_bal: 20000,
    acc_type: 'Checking'
  },
  {
    _id: ObjectId("675fe7b5f2355f925cc449d1"),
    custid: 3,
    acc_bal: 50000,
    acc_type: 'Checking'
  }
]
```

Determine Minimum and Maximum account balance for each customer_id.

db.Customer.aggregate([{$group:{_id:"$custi

```
[
  { _id: 3, minBal: 50000, maxBal: 50000 },
  { _id: 5, minBal: 2000, maxBal: 2000 },
  { _id: 1, minBal: 10000, maxBal: 20000 },
  { _id: 4, minBal: 10000, maxBal: 10000 }
]
```

d", minBal:{$min:"$acc_bal"}, maxBal:{$max:"$acc_bal"}}}]);

db.Customers.drop()

```
true
```

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

_id	custid	acc_bal	acc_type
674ff20946b4cd1ffe	1	10000	Saving
674ff20946b4cd1ffe	1	20000	Checking
674ff20946b4cd1ffe	3	50000	Checking
674ff20946b4cd1ffe	4	10000	Saving
674ff20946b4cd1ffe	5	2000	Checking

NO SQL RESTAURANTS DATABASE

(WEEK-10)
QUESTION

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

Create Database:

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

```
{ ok: 1 }
```

Inserting Values to the tables:

```
db.restaurants.insertMany([{ name: "Meghna Foods", town: "Jayanagar", cuisine: "Indian", score: 8, address: { zipcode: "10001", street: "Jayanagar"}},{ name: "Empire", town: "MG Road", cuisine: "Indian", score: 7, address: { zipcode: "10100", street: "MG Road"}},{ name: "Chinese WOK", town: "Indiranagar", cuisine: "Chinese", score: 12, address: { zipcode: "20000", street: "Indiranagar" }},{ name: "Kyotos", town: "Majestic", cuisine: "Japanese", score: 9, address: { zipcode: "10300", street: "Majestic" }},{ name: "WOW Momos", town: "Malleshwaram", cuisine: "Indian", score: 5, address: { zipcode: "10400", street: "Malleshwaram"}}])
```

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("67600441f2355f925cc449d4"),
    '1': ObjectId("67600441f2355f925cc449d5"),
    '2': ObjectId("67600441f2355f925cc449d6"),
    '3': ObjectId("67600441f2355f925cc449d7"),
    '4': ObjectId("67600441f2355f925cc449d8")
  }
}
```

Queries:

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

db.restaurants.find({})

```
[
  {
    _id: ObjectId("67600441f2355f925cc449d4"),
    name: 'Meghna Foods',
    town: 'Jayanagar',
    cuisine: 'Indian',
    score: 8,
    address: { zipcode: '10001', street: 'Jayanagar' }
  },
  {
    _id: ObjectId("67600441f2355f925cc449d5"),
    name: 'Empire',
    town: 'MG Road',
    cuisine: 'Indian',
    score: 7,
    address: { zipcode: '10100', street: 'MG Road' }
  },
  {
    _id: ObjectId("67600441f2355f925cc449d6"),
    name: 'Chinese WOK',
    town: 'Indiranagar',
    cuisine: 'Chinese',
    score: 12,
    address: { zipcode: '20000', street: 'Indiranagar' }
  },
  {
    _id: ObjectId("67600441f2355f925cc449d7"),
    name: 'Kyotos',
    town: 'Majestic',
    cuisine: 'Japanese',
    score: 9,
    address: { zipcode: '10300', street: 'Majestic' }
  },
  {
    _id: ObjectId("67600441f2355f925cc449d8"),
    name: 'WOW Momos',
    town: 'Malleshwaram',
    cuisine: 'Indian',
    score: 5,
    address: { zipcode: '10400', street: 'Malleshwaram' }
  }
]
```

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

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

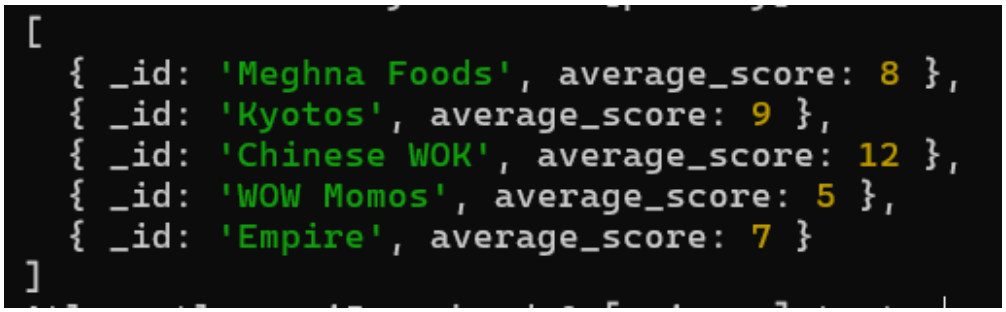
```
[
  {
    _id: ObjectId("67600441f2355f925cc449d8"),
    name: 'WOW Momos',
    town: 'Malleshwaram',
    cuisine: 'Indian',
    score: 5,
    address: { zipcode: '10400', street: 'Malleshwaram' }
  },
  {
    _id: ObjectId("67600441f2355f925cc449d4"),
    name: 'Meghna Foods',
    town: 'Jayanagar',
    cuisine: 'Indian',
    score: 8,
    address: { zipcode: '10001', street: 'Jayanagar' }
  },
  {
    _id: ObjectId("67600441f2355f925cc449d7"),
    name: 'Kyotos',
    town: 'Majestic',
    cuisine: 'Japanese',
    score: 9,
    address: { zipcode: '10300', street: 'Majestic' }
  },
  {
    _id: ObjectId("67600441f2355f925cc449d5"),
    name: 'Empire',
    town: 'MG Road',
    cuisine: 'Indian',
    score: 7,
    address: { zipcode: '10100', street: 'MG Road' }
  },
  {
    _id: ObjectId("67600441f2355f925cc449d6"),
    name: 'Chinese WOK',
    town: 'Indiranagar',
    cuisine: 'Chinese',
    score: 12,
    address: { zipcode: '20000', street: 'Indiranagar' }
  }
]
```

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

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

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

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



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

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

```
[
  { name: 'Meghna Foods', address: { street: 'Jayanagar' } },
  { name: 'Empire', address: { street: 'MG Road' } },
  { name: 'Kyotos', address: { street: 'Majestic' } },
  { name: 'WOW Momos', address: { street: 'Malleshwaram' } }
]
```