VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

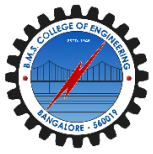
On

Database Management Systems (23CS3PCDBM)

Submitted by

Bhoomika B G (1BM23CS067)

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 **Bhoomika B G (1BM23CS067**), 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.

Dr.Kayarvizhy N Assistant Professor Department of CSE, BMSCE	Dr. Kavitha Sooda Professor, HOD Department of CSE, BMSCE

Index

Sl. No.	Date	Experimental Title	Page No.
1	04/10/2024	Insurance Database	4-12
2	11/10/2024	More Queries on Insurance Database	13-15
3	18/10/2024	Bank Database	16-22
4	25/10/2024	More Queries on Bank Database	23-26
5	08/11/2024	Employee Database	27-32
6	15/11/2024	More Queries on Employee Database	33-35
7	22/11/2024	Supplier Database	36-42
8	29/11/2024	NO SQL – Student Database	43-47
9	06/12/2024	NO SQL – Customer Database	48-50
10	27/12/2024	NO SQL – Restaurant Database	51-71

Insurance Database

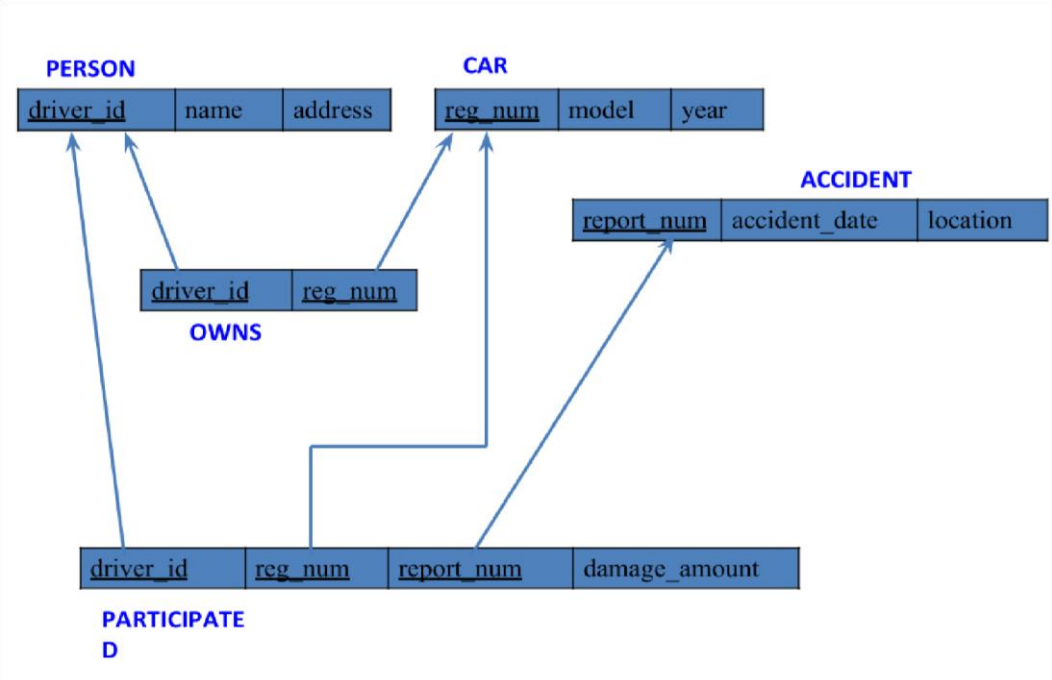
(Week 1)

Question

- 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 insurance; use insurance;

Create table

```
create table person(
driver_id varchar(20),
name varchar(30),
address varchar(50),

PRIMARY KEY(driver_id)
);
create table car(
reg_num varchar(15),
model varchar(10),
year int,

PRIMARY KEY(reg_num)
```

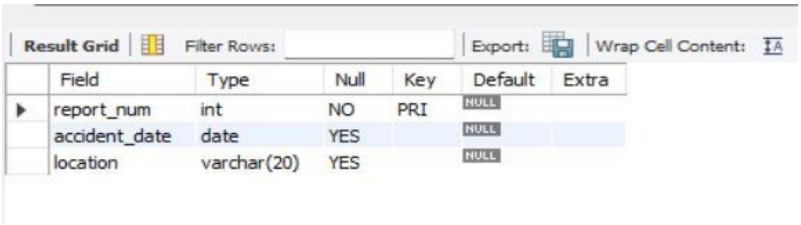
```
);
create table owns(
driver_id varchar(20),
reg_num varchar(10),

PRIMARY KEY(driver_id, reg_num),
FOREIGN KEY(driver_id) REFERENCES person(driver_id),
FOREIGN KEY(reg_num) REFERENCES car(reg_num)
);
create table accident(
report_num int,
accident_date date,
location varchar(50),

PRIMARY KEY(report_num)
);
create table participated(
driver_id varchar(20),
reg_num varchar(10),
report_num int,
damage_amount int,

PRIMARY KEY(driver_id,reg_num,report_num),
FOREIGN KEY(driver_id) REFERENCES person(driver_id),
FOREIGN KEY(reg_num) REFERENCES car(reg_num),

FOREIGN KEY(report_num) REFERENCES accident(report_num)
);
```

Structure of the Table

desc person;

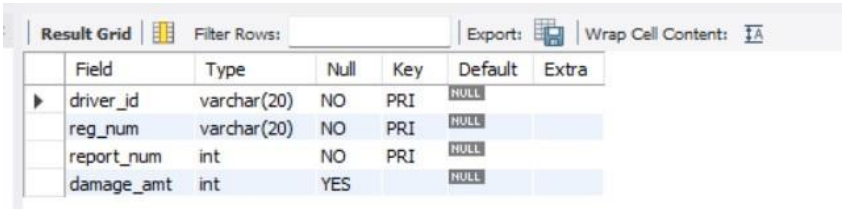
Field	Type	Null	Key	Default	Extra
driver_id	varchar(20)	NO	PRI	NULL	
reg_num	varchar(20)	NO	PRI	NULL	
report_num	int	NO	PRI	NULL	
damage_amt	int	YES		NULL	

desc accident;



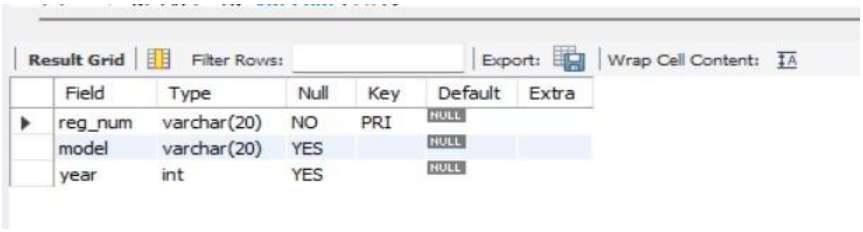
Field	Type	Null	Key	Default	Extra
report_num	int	NO	PRI	NULL	
accident_date	date	YES		NULL	
location	varchar(20)	YES		NULL	

desc participated;



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;



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”);

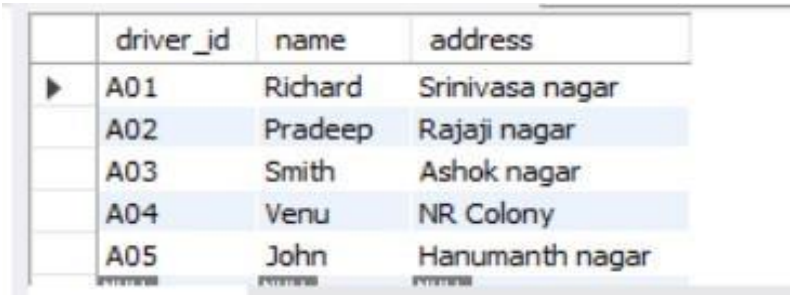
insert into person values(“A02”, “Pradeep”, “Rajaji nagar”);

insert into person values(“A03”, “Smith”, “Ashok nagar”);

insert into person values(“A04”, “Venu”, “N R Colony”);

insert into person values(“A05”, “John”, “Hanumanth nagar”);

select * from person;



driver_id	name	address
A01	Richard	Srinivasa nagar
A02	Pradeep	Rajaji nagar
A03	Smith	Ashok nagar
A04	Venu	NR Colony
A05	John	Hanumanth nagar

insert into car values(“KA052250”, “Indica”, “1990”);

insert into car values(“KA031181”, “Lancer”, “1957”);

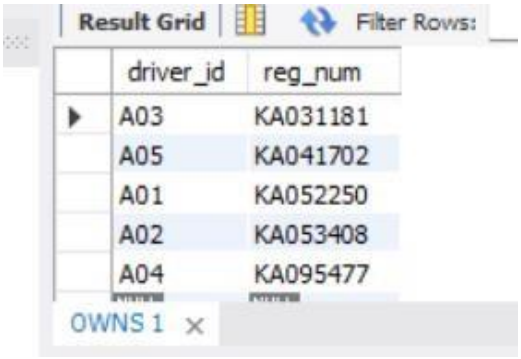
insert into car values(“KA095477”, “Toyota”, “1998”);

insert into car values(“KA053408”, “Honda”, “2008”);

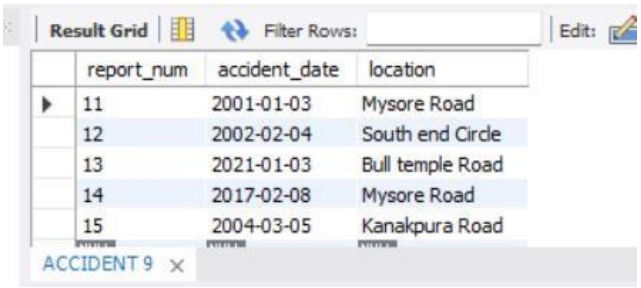
insert into car values(“KA041702”, “Audi”, “2005”);

select * from car;

insert into owns values(“A01”, “KA052250”); insert into owns values(“A02”, “KA031181”); insert into owns values(“A03”, “KA095477”); insert into owns values(“A04”, “KA053408”); insert into owns values(“A05”, “KA041702”); select * from owns;



insert into accident values(11, ‘2003-01-01’, “Mysore Road”); insert into accident values(12, ‘2004-02-02’, “South end Circle”); insert into accident values(13, ‘2003-01-21’, “Bull temple Road”); insert into accident values(14, ‘2008-02-17’, “Mysore Road”); insert into accident values(15, ‘2004-03-05’, “Kanakpura Road”); select * from accident;



Result Grid | Filter Rows: | Edit:

report_num	accident_date	location
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

ACCIDENT 9

insert into participated values(“A01”, “KA052250”,11,10000);

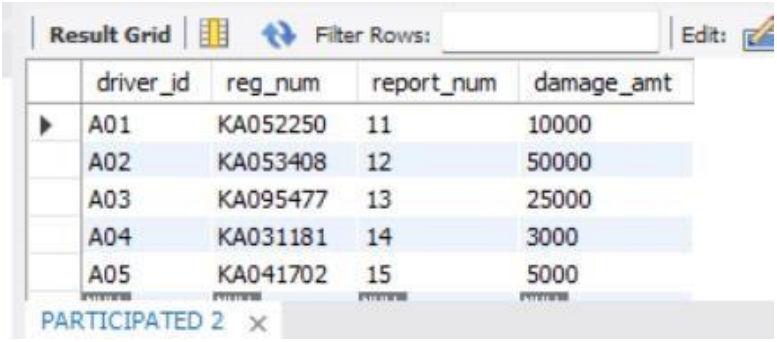
insert into participated values(“A02”, “KA053408”,12,50000);

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;



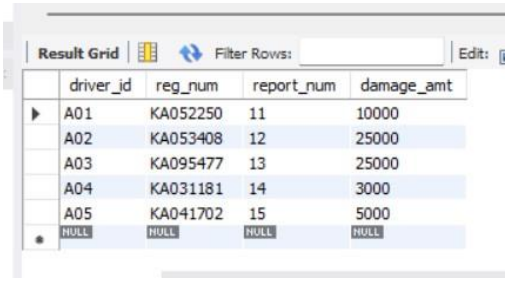
Result Grid | Filter Rows: | Edit:

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

PARTICIPATED 2

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_amount=25000 where reg_num='KA053408' and report_num=12; select * from participated;



Result Grid | Filter Rows: | Edit:

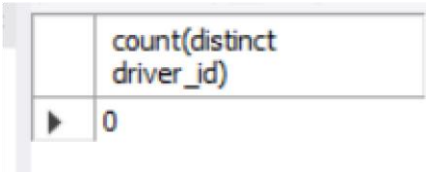
driver_id	reg_num	report_num	damage_amt
A01	KA052250	11	10000
A02	KA053408	12	25000
A03	KA095477	13	25000
A04	KA031181	14	3000
A05	KA041702	15	5000
NULL	NULL	NULL	NULL

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 "__08%";



count(distinct driver_id)
0

3) Add a new accident to the database.

Insert into accident values(16, '2008-03-08', "Domlur"); select

* from accident;

report_num	accident_date	location
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 2

4) Display accident date and location. Select accident_date,location from accident;

Result Grid | Filter Rows:

date	location
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

ACCIDENT 5

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;

```
81 • select driver_id
82   from PARTICIPATED
83   where damage_amt>=25000;
```

Result Grid | Filter Rows: | Export:

driver_id
A02
A03

More Queries on Insurance Database (week 2)

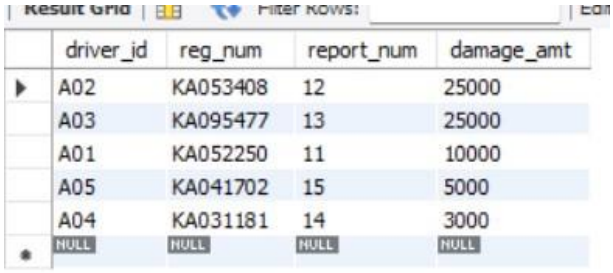
Question

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.

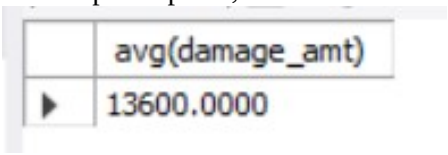
```
SELECT *
 FROM participated
 ORDER BY damage_amt DESC;
```



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

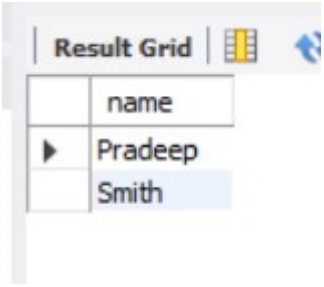
3) Delete the tuple from participated whose damage amount is below average damage amount.

```
    Delete FROM participated

WHERE damage_amt<
                    (SELECT avg (damage_amt)
                     FROM participated);
Select * from participated;
```

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

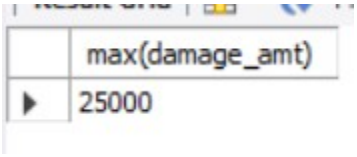
```
SELECT name
FROM person a, participated b
WHERE a.driver_id=b.driver_id AND damage_amt>
                                        (SELECT avg(damage_amt)
```

FROM PARTICIPATED);



5) Find maximum damage amount.

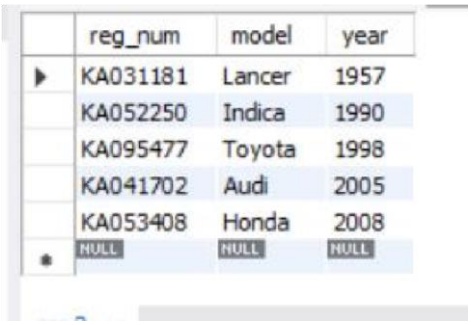
SELECT max(damage_amt) FROM participated;



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

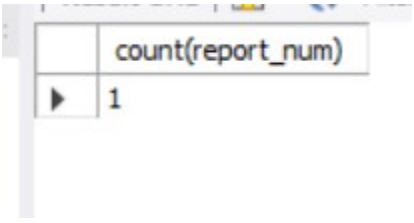
Select * from car

order by year asc;



7) Find the number of accidents in which cars belonging to a specific model (example 'Lancer') were involved. Select count(report_num) from car c, participated p

where c.reg_num=p.reg_num and c.model='Lancer';



Bank Database

(week 3)

Question

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

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

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

customer-city: String)

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

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

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

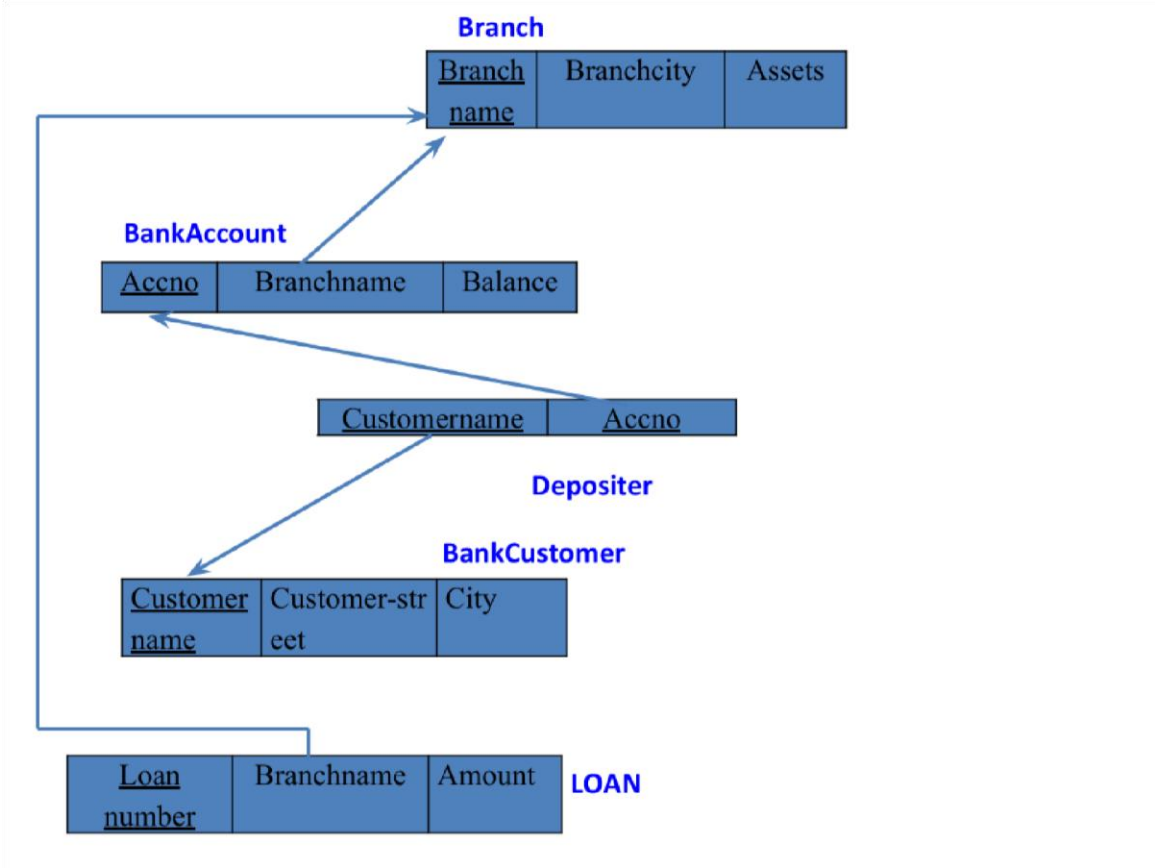
2.Enter at least five tuples for each relation.

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

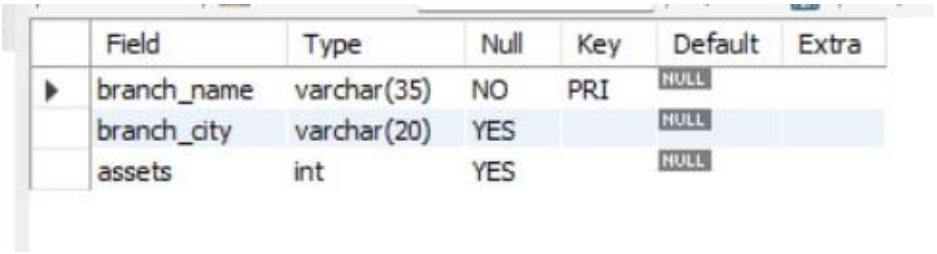
create database bank; use
bank;

Create Table

```
create table branch(
branch_name varchar(35),
branch_city varchar(20), assets
int,
primary key(branch_name)
);

create table BankAccount(
accno int, branch_name
varchar(35), balance int,
primary key(accno),
foreign key(branch_name) references branch(branch_name)
);

create table bankcustomer(
customer_name varchar(25),
customer_street varchar(50),
city varchar(100), primary
key(customer_name)
);

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

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

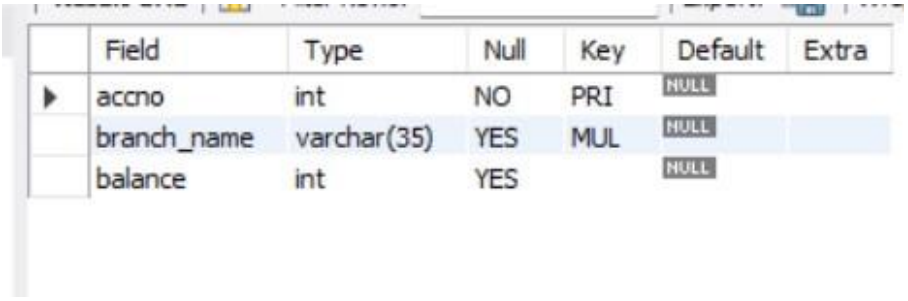
Structure of the Table

desc branch;



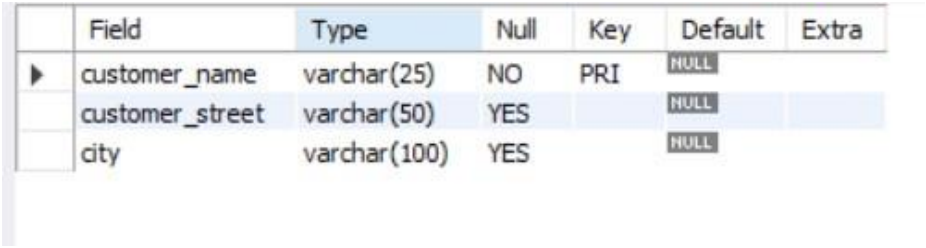
Field	Type	Null	Key	Default	Extra
branch_name	varchar(35)	NO	PRI	NULL	
branch_city	varchar(20)	YES		NULL	
assets	int	YES		NULL	

desc BankAccount;



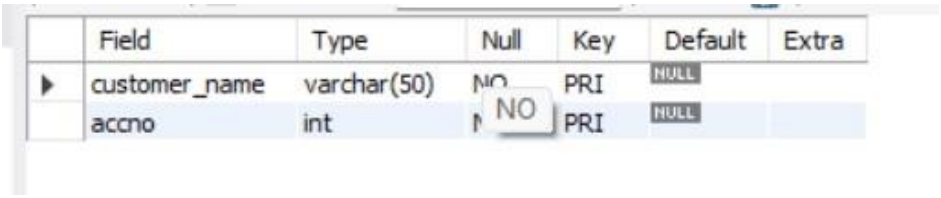
Field	Type	Null	Key	Default	Extra
accno	int	NO	PRI	NULL	
branch_name	varchar(35)	YES	MUL	NULL	
balance	int	YES		NULL	

desc bankcustomer;



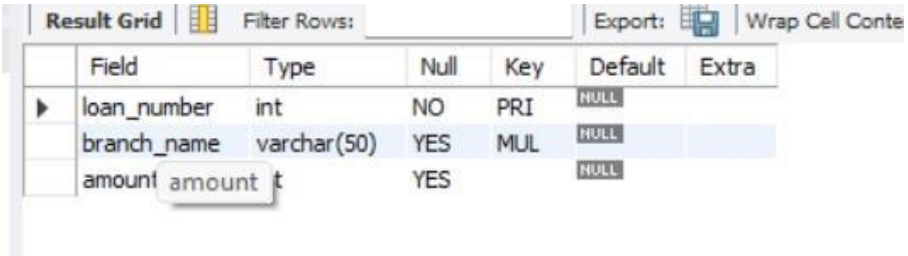
Field	Type	Null	Key	Default	Extra
customer_name	varchar(25)	NO	PRI	NULL	
customer_street	varchar(50)	YES		NULL	
city	varchar(100)	YES		NULL	

desc depositer;



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

desc loan;

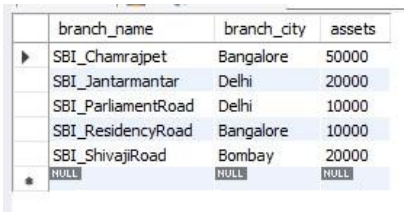


Result Grid | Filter Rows: | Export: | Wrap Cell Conte

Field	Type	Null	Key	Default	Extra
loan_number	int	NO	PRI	NULL	
branch_name	varchar(50)	YES	MUL	NULL	
amount	t	YES		NULL	

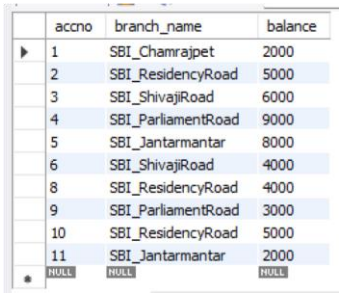
Inserting Values to the table

insert into branch values('SBI_Chamrajpet', 'Bangalore', 50000);
insert into branch values('SBI_ResidencyRoad', 'Bangalore', 10000);
insert into branch values('SBI_ShivajiRoad', 'Bombay', 20000);
insert into branch values('SBI_ParliamentRoad', 'Delhi', 10000);
insert into branch values('SBI_Jantarmantar', 'Delhi', 20000); select
* from branch;



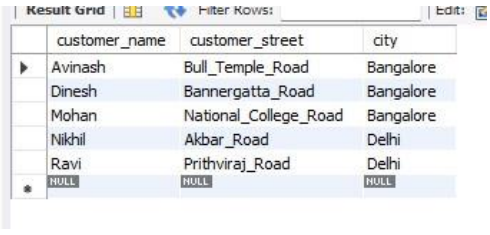
branch_name	branch_city	assets
SBI_Chamrajpet	Bangalore	50000
SBI_Jantarmantar	Delhi	20000
SBI_ParliamentRoad	Delhi	10000
SBI_ResidencyRoad	Bangalore	10000
SBI_ShivajiRoad	Bombay	20000
NULL	NULL	NULL

insert into BankAccount values(1, 'SBI_Chamrajpet', 2000); insert
into BankAccount values(2, 'SBI_ResidencyRoad', 5000); insert
into BankAccount values(3, 'SBI_ShivajiRoad', 6000); insert into
BankAccount values(4, 'SBI_ParliamentRoad', 9000); insert into
bankAccount values(5, 'SBI_Jantarmantar', 8000); insert into
BankAccount values(6, 'SBI_ShivajiRoad', 4000); insert into
BankAccount values(8, 'SBI_ResidencyRoad', 4000); insert into
BankAccount values(9, 'SBI_ParliamentRoad', 3000); insert into
BankAccount values(10, 'SBI_ResidencyRoad', 5000); insert into
BankAccount values(11, 'SBI_Jantarmantar', 2000); select * from
BankAccount;



accno	branch_name	balance
1	SBI_Chamrajpet	2000
2	SBI_ResidencyRoad	5000
3	SBI_ShivajiRoad	6000
4	SBI_ParliamentRoad	9000
5	SBI_Jantarmantar	8000
6	SBI_ShivajiRoad	4000
8	SBI_ResidencyRoad	4000
9	SBI_ParliamentRoad	3000
10	SBI_ResidencyRoad	5000
11	SBI_Jantarmantar	2000
NULL	NULL	NULL

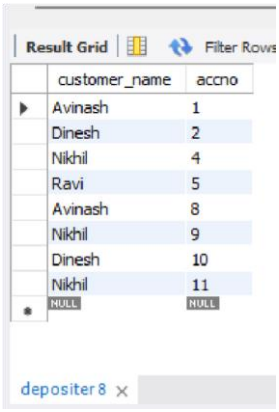
insert into bankcustomer values ('Avinash', 'Bull_Temple_Road', 'Bangalore');
insert into bankcustomer values ('Dinesh', 'Bannergatta_Road', 'Bangalore');
insert into bankcustomer values ('Mohan', 'National_College_Road', 'Bangalore');
insert into bankcustomer values ('Nikhil', 'Akbar_Road', 'Delhi'); insert into
bankcustomer values ('Ravi', 'Prithviraj_Road', 'Delhi'); select * from
bankcustomer;



Result Grid | Filter Rows: | Edit:

customer_name	customer_street	city
Avinash	Bull_Temple_Road	Bangalore
Dinesh	Bannergatta_Road	Bangalore
Mohan	National_College_Road	Bangalore
Nikhil	Akbar_Road	Delhi
Ravi	Prithviraj_Road	Delhi
NULL	NULL	NULL

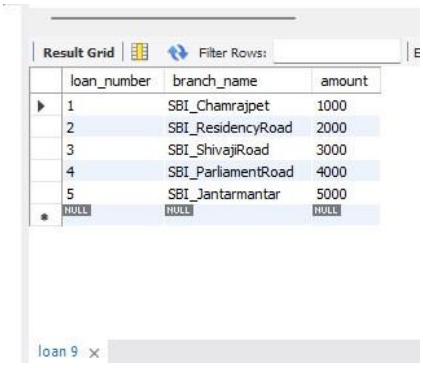
insert into depositer values(‘Avinash’, 1);
insert into depositer values(‘Dinesh’, 2);
insert into depositer values(‘Nikhil’, 4);
insert into depositer values(‘Ravi’, 5);
insert into depositer values(‘Avinash’, 8);
insert into depositer values(‘Nikhil’, 9);
insert into depositer values(‘Dinesh’, 10);
insert into depositer values(‘Nikhil’, 11);
select * from depositer;


Result Grid | Filter Rows

customer_name	accno
Avinash	1
Dinesh	2
Nikhil	4
Ravi	5
Avinash	8
Nikhil	9
Dinesh	10
Nikhil	11
NULL	NULL

depositer 8

insert into loan values (1,
‘SBI_Chamrajpet’, 1000),
(2, ‘SBI_ResidencyRoad’, 2000),
(3, ‘SBI_ShivajiRoad’, 3000),
(4, ‘SBI_ParliamentRoad’, 4000), (5,
‘SBI_Jantarmantar’, 5000);
select * from loan;


Result Grid | Filter Rows:

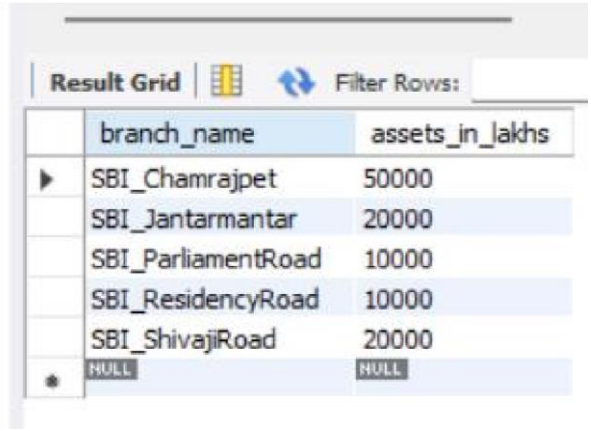
loan_number	branch_name	amount
1	SBI_Chamrajpet	1000
2	SBI_ResidencyRoad	2000
3	SBI_ShivajiRoad	3000
4	SBI_ParliamentRoad	4000
5	SBI_Jantarmantar	5000
NULL	NULL	NULL

loan 9

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 branch_name,assets as assets_in_lakhs from
branch;

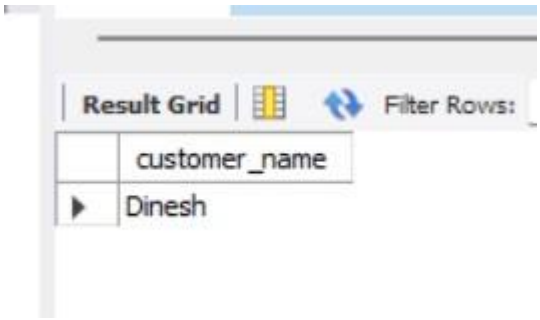


Result Grid | Filter Rows:

branch_name	assets_in_lakhs
SBI_Chamrajpet	50000
SBI_Jantarmantar	20000
SBI_ParliamentRoad	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.customer_name from
depositer d, BankAccount b
where b.accno=d.accno and branch_name='SBI_ResidencyRoad'
group by customer_name having count(customer_name)>1;



Result Grid | Filter Rows:

customer_name
Dinesh

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

```
CREATE VIEW BranchLoanSummary AS
SELECT
   branch_name,
   SUM(amount) AS total_loan_amount
FROM
   loans
GROUP BY
   branch_name;
```

More Queries on Bank Database

(week 4)

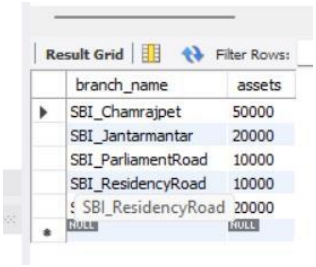
Question

1. Retrieve all branches and their respective total assets.
2. List all customers who live in a particular city.
3. List all customers with their account numbers.
4. Find all the customers who have an account at all the branches located in a specific city (Ex. Delhi).
5. Find all customers who have accounts with a balance greater than a specified amount (100000).
6. List all customers who have both a loan and an account at the same branch.
7. Get the number of accounts held at each branch.
8. Find all branches that have no loans issued.
9. Retrieve the branch with the smallest total loan amount.

Queries

1. Retrieve all branches and their respective total assets.

Select branch_name,assets from
branch;

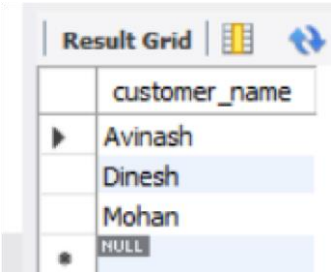


Result Grid | Filter Rows:

branch_name	assets
SBI_Chamrajpet	50000
SBI_Jantarmantar	20000
SBI_ParliamentRoad	10000
SBI_ResidencyRoad	10000
SBI_ResidencyRoad	20000
NULL	NULL

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

Select customer_name from
bankcustomer
where city=’Bangalore’;



Result Grid

customer_name
Avinash
Dinesh
Mohan
NULL

3. List all customers with their account numbers.

Select a.accno,d.customer_name from
bankaccount a,depositer d
where d.accno=a.accno;

accno	customer_name
1	Avinash
5	Ravi
11	Nikhil
4	Nikhil
9	Nikhil
2	Dines
8	Avinash
10	Dinesh

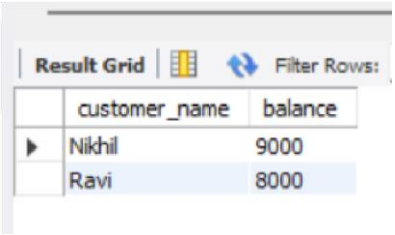
Dinesh

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

```
Select c.customer_name from
bankcustomer c
join depositer d ON c.customer_name = d.customer_name
join BankAccount a ON d.accno = a.accno join Branch b
ON a.branch_name = b.branch_name
where b.branch_city = 'Delhi' group
by c.customer_name
having count (distinct b.branch_name) = (
   select count(*)
from branch
   where branch_city = 'Delhi');
```

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

```
select c.customer_name,a.balance from bankcustomer c,BankAccount a,depositer d where
a.accno=d.accno and d.customer_name=c.customer_name having a.balance>5000;
```

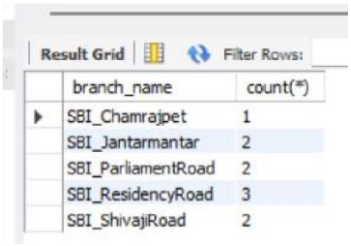


6. List all customers who have both a loan and an account at the same branch.

```
select a.accno,d.customer_name from
bankaccount a,loan l,depositer d
where a.branch_name=l.branch_name and a.accno=l.loan_number and d.accno=a.accno;
```

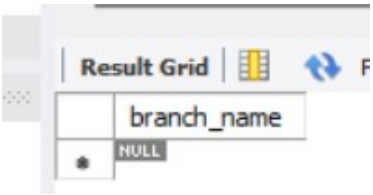
7. Get the number of accounts held at each branch.

```
select  branch_name,count(*)
from BankAccount group by
branch_name;
```



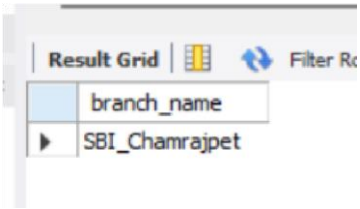
8. Find all branches that have no loans issued.

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



9. Retrieve the branch with the smallest total loan amount.

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



Employee Database

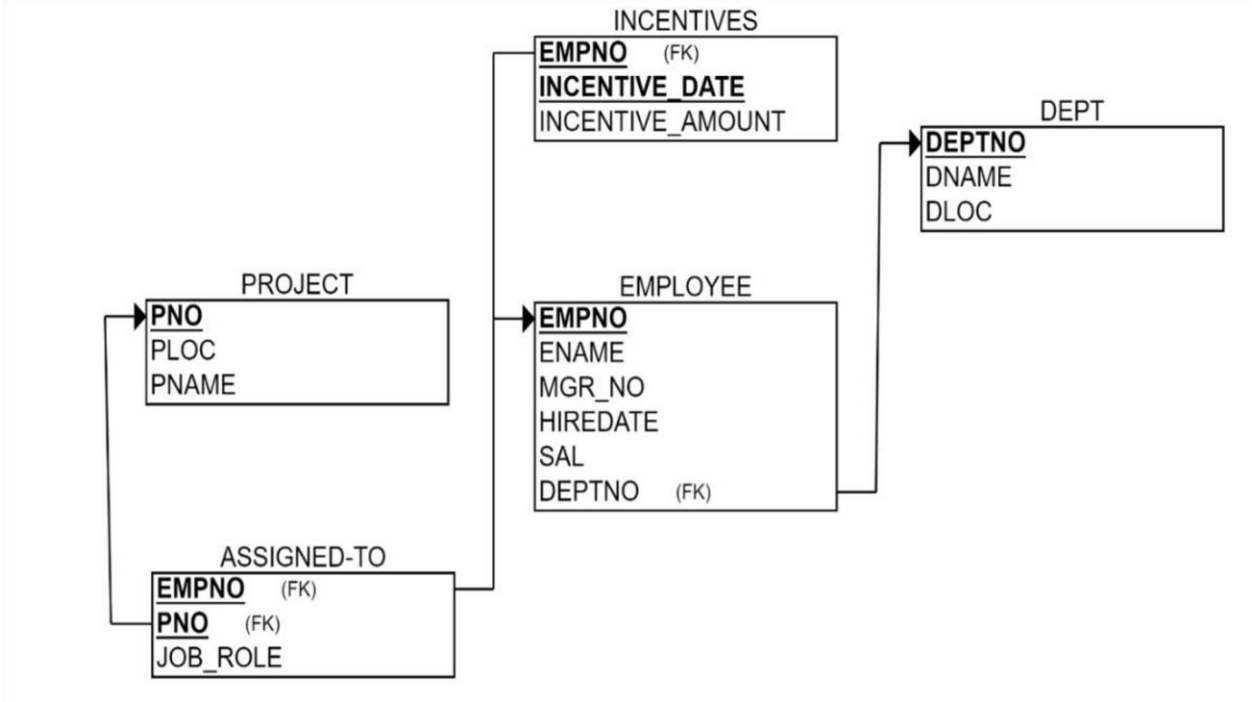
(week 5)

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.

Schema Diagram



Create Database

create database employee; use
employee;

Create tables

```
create table dept2(
deptno int, dname
varchar(10), dloc
varchar(200),
primary key(deptno));

create table employee2(
empno int primary key,
ename varchar(50),
mgr_no int, hiredate
date, sal float, deptno
int,
foreign key(deptno) references dept2(deptno));

create table incentives2(
empno int,
incentive_date date,
incentive_amt float,
primary key(empno,incentive_date),
foreign key(empno) references employee2(empno));
```

```
create table project2(
pno int primary key, ploc
varchar(100),
pname varchar(50));

create table assigned_to2( empno int, pno int,
job_role varchar(50), primary key(empno,pno),
foreign key(empno) references employee2(empno),
foreign key(pno) references project2(pno));
```

Structure of the table

desc dept1;

Field	Type	Null	Key	Default	Extra
deptno	int	NO	PRI	NULL	
dname	varchar(10)	YES		NULL	
dloc	varchar(200)	YES		NULL	

desc employee;

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

desc incentives;

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

desc project1;

Field	Type	Null	Key	Default	Extra
pno	int	NO	PRI	NULL	
ploc	varchar(100)	YES		NULL	
pname	varchar(50)	YES		NULL	

desc assigned_to;

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

Inserting values to the table

insert into dept2 values(1,”CSE”,”Bengaluru”),
(2,”ISE”,”Hyderabad”),
(3,”CIVIL”,”Mysuru”),
(4,”AIML”,”Bengaluru”),
(5,”ECE”,”Chennai”),
(6,”MECH”,”Kochi”);
select * from dept2;

deptno	dname	dloc
1	CSE	Bengaluru
2	ISE	Hyderabad
3	CIVIL	Mysuru
4	AIML	Bengaluru
5	ECE	Chennai
6	MECH	Kochi
NULL	NULL	NULL

insert into employee2
values(101,”AA”,111,’2022-07-19’,25000,4);
insert into employee2 values(102,”BB”,104,”2020-06-04”,50000,1),
(103,”CC”,104,”2024-01-20”,12000,1),
(104,”DD”,104,”2015-08-10”,70000,1),
(105,”EE”,106,”2020-10-12”,20000,3),
(106,”FF”,106,”2010-01-10”,80000,3);
select * from employee1;

empno	ename	mgr_no	hiredate	sal	deptno
101	AA	111	2022-07-19	25000	4
102	BB	104	2020-06-04	50000	1
103	CC	104	2024-01-20	12000	1
104	DD	NULL	2015-08-10	70000	1
105	EE	106	2020-10-12	20000	3
106	FF	NULL	2010-01-10	80000	3
107	GG	NULL	2018-02-10	50000	6
NULL	NULL	NULL	NULL	NULL	NULL

insert into incentives2
values(101,”2023-08-20”,5000), (102,”2022-10-08”,10000),
(106,”2014-04-01”,15000),
(104,”2020-07-19”,30000),
(105,”2021-08-23”,4000),
(106,”2018-03-31”,50000);
select * from incentives1;

empno	incentive_date	incentive_amt
101	2023-08-20	5000
102	2022-10-08	10000
104	2020-07-19	30000
105	2021-08-23	4000
106	2014-04-01	15000
106	2018-03-31	50000
NULL	NULL	NULL

insert into project2 values(1234,”Bengaluru”,”P1”),
(3456,”Hyderabad”,”P2”),
(5678,”Chennai”,”P3”),
(6789,”Mysuru”,”P4”),
(9101,”Bengaluru”,”P5”),
(1213,”Kochi”,”P6”);
select * from project1;

pno	ploc	pname
1213	Kochi	P6
1234	Bengaluru	P1
3456	Hyderabad	P2
5678	Chennai	P3
6789	Mysuru	P4
9101	Bengaluru	P5
NULL	NULL	NULL

insert into assigned_to2
values(101,9101,”Data_analyst”),
(103, 3456,”App_Developer”),
(104, 6789,”Civil_engineer”),
(106,1234,”tester”),
(104,1234,”AI_Manager”); select
* from assigned_to1;

empno	pno	job_role
101	9101	Data_analyst
103	3456	App_Developer
104	1234	AI_Manager
104	6789	Civil_engineer
107	1213	Tester
NULL	NULL	NULL

Queries

1. Retrieve the employee numbers of all employees who work on project located in Bengaluru, Hyderabad, or Mysuru. Select * from employee1; select distinct a.empno from assigned_to1 a where a.pno in (select p.pno
from project1 p
where p.ploc =”Bengaluru” or p.ploc=”Mysuru” or p.ploc=”Hyderabad”);

empno	ename	mgr_no	hiredate	sal	deptno
101	AA	111	2022-07-19	25000	4
102	BB	104	2020-06-04	50000	1
103	CC	104	2024-01-20	12000	1
104	DD	NULL	2015-08-10	70000	1
105	EE	106	2020-10-12	20000	3
106	FF	NULL	2010-01-10	80000	3
107	GG	NULL	2018-02-10	50000	6
NULL	NULL	NULL	NULL	NULL	NULL

2.Get Employee ID’s of those employees who didn’t receive incentives.
Select e.empno from employee1 e
where empno not in(select empno from incentives1 i where e.empno=i.empno);

empno
103
107
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.ename,e.deptno,a.job_role,d.dloc,p.ploc from employee1 e,assigned_to1 a,dept1 d,project1 p where e.deptno=d.deptno and e.empno=a.empno and
a.pno=p.pno and
d.dloc=p.ploc;

empno	ename	deptno	job_role	dloc	ploc
101	AA	4	Data_analyst	Bengaluru	Bengaluru
104	DD	1	AI_Manager	Bengaluru	Bengaluru
107	GG	6	Tester	Kochi	Kochi

More Queries on Employee Database

(Week 6)

Queries

1.List all employees along with their project details (if assigned).
Select e.ename,a.empno,a.pno,p.pname,p.ploc,a.job_role from employee2 e, assigned_to2 a,project2 p where e.empno=a.empno and a.pno=p.pno;

ename	empno	pno	pname	ploc	job_role
AA	101	9101	P5	Bengaluru	Data_analyst
CC	103	3456	P2	Hyderabad	App_Developer
DD	104	1234	P1	Bengaluru	AI_Manager
DD	104	6789	P4	Mysuru	Civil_engineer
GG	107	1213	P6	Kochi	Tester

2.Find all employees who received incentives, along with the total incentive amount.
Select e.ename,i.empno,sum(i.incentive_amt) from incentives2 i, employee2 e where e.empno=i.empno group by i.empno;

Result Grid | Filter Rows:

ename	empno	sum(i.incentive_amt)
AA	101	5000
BB	102	10000
DD	104	30000
EE	105	4000
FF	106	65000

Result 11

3.Retrieve the project names and locations of projects with employees assigned as 'Manager'.
Select p.pname,p.ploc,a.empno,a.job_role from project2 p, assigned_to2 a where p.pno=a.pno and job_role="AI_Manager

Result Grid | Filter Rows: | Export:

pname	ploc	empno	job_role
P1	Bengaluru	104	AI_Manager

4.List departments along with the number of employees in each department.
Select d.dname,d.deptno,count(e.empno) as no_of_employees from dept2 d,employee2 e where d.deptno=e.deptno group by d.deptno;

Result Grid | Filter Rows:

dname	deptno	no_of_employees
CSE	1	3
CIVIL	3	2
AIML	4	1

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

```
Select e.empno,e.ename
from employee2 e
where e.empno not in(select a.empno from assigned_to2 a);
```

empno	ename
102	BB
105	EE
106	FF
NULL	NULL

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

```
Select e.empno,e.ename,d.deptno,d.dloc from employee2 e,dept2 d where
d.deptno=e.deptno;
```

empno	ename	deptno	dloc
101	AA	4	Bengaluru
102	BB	1	Bengaluru
103	CC	1	Bengaluru
104	DD	1	Bengaluru
105	EE	3	Mysuru
106	FF	3	Mysuru
107	GG	6	Kochi

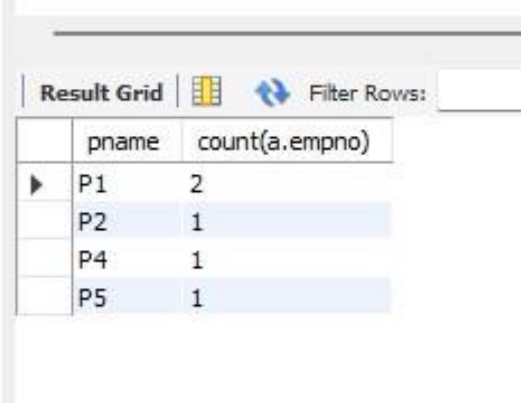
7.Retrieve the details of employees who work under a specific manager (e.g., manager with empno = 104).

```
Select e.* from
employee2 e where
mgr_no=104;
```

empno	ename	mgr_no	hiredate	sal	deptno
102	BB	104	2020-06-04	50000	1
103	CC	104	2024-01-20	12000	1
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(a.empno) from project2 p,assigned_to2 a
where p.pno=a.pno group
by p.pname;
```



pname	count(a.empno)
P1	2
P2	1
P4	1
P5	1

10.List the total number of incentives given to each employee and the sum of incentives for each.
Select i.empno,count(i.empno) as no_of_incentives,sum(i.incentive_amt) as total_incentive from incentives2 i group by i.empno;

empno	no_of_incentives	total_incentive
101	1	5000
102	1	10000
104	1	30000
105	1	4000
106	2	65000

11.Retrieve all employees who have the role of 'Developer' on any project.
Select e.ename,a.empno,a.job_role from employee2 e, assigned_to2 a
where e.empno=a.empno and a.job_role="App_Developer";

ename	empno	job_role
CC	103	App_Developer

12.Display the department-wise average salary of employee.
Select d.deptno,d.dname,avg(e.sal) as avg_salary from dept2 d,
employee2 e where e.deptno=d.deptno
group by d.deptno;

deptno	dname	avg_salary
4	AIML	25000
1	CSE	44000
3	CIVIL	50000
6	MECH	50000

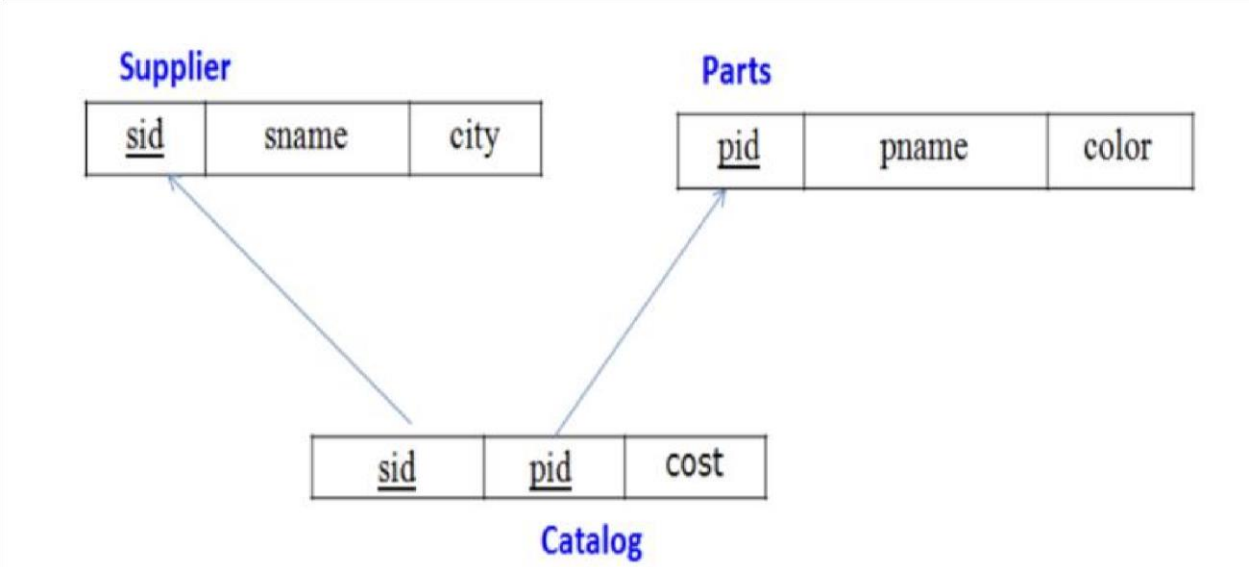
Supplier Database

(Week 7)

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 supply_204;
use supply_204;
```

Create table

```
create table supplier_204(
        sid int primary key,
   sname varchar(20),
   city varchar(30)
);
```

```
create table parts_204(    pid int primary key,    pname varchar(20),
  color varchar(20)
);

create table catalog_204(
 sid int, pid int,
 cost int,
  foreign key(sid) references supplier_204(sid),
foreign key(pid) references parts_204(pid)
);
```

Structure of the Table

desc Supplierr;

Field	Type	Null	Key	Default	E
sid	int	NO	PRI	NULL	
sname	varchar(20)	YES		NULL	
city	varchar(30)	YES		NULL	

desc Parts;

Field	Type	Null	Key	Default
pid	int	NO	PRI	NULL
pname	varchar(20)	YES		NULL
color	varchar(20)	YES		NULL

desc Catalog;

Field	Type	Null	Key	Default
sid	int	YES	MUL	NULL
pid	int	YES	MUL	NULL
cost	int	YES		NULL

Inserting Values to the table

```
insert into supplier_204 values
        (10001, "acne", "Bangalore"),
   (10002, "johns", "Kolkata"),
   (10003, "vimal", "Mumbai"),
   (10004, "reliance", "Delhi");
select * from supplier_204;
```

sid	sname	city
10001	acne	Bangalore
10002	johns	Kolkata
10003	vimal	Mumbai
10004	reliance	Delhi

```
insert into parts_204 values
        (20001, "Book", "Red"),
   (20002, "Pen", "Red"),
   (20003, "Pencil", "Green"),
   (20004, "Mobile", "Green"),
   (20005, "Charger", "Black");
select * from parts_204;
```

pid	pname	color
20001	Book	Red
20002	Pen	Red
20003	Pencil	Green
20004	Mobile	Green
20005	Charger	Black

```
insert into catalog_204 values
        (10001, 20001, 10),
   (10001, 20002, 10),
   (10001, 20003, 30),
   (10001, 20004, 10),
   (10001, 20005, 10),
   (10002, 20001, 10),
   (10002, 20002, 20),
   (10003, 20003, 30),
   (10004, 20003, 40);
select * from catalog_204;
```

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

Queries

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

```
select pname  from parts_204
where pid in (select pid from catalog_204);
```

pname
Book
Pen
Pencil
Mobile
Charger

Find the snames of suppliers who supply every part.

```
select sname from supplier_204 where sid in
(select sid from catalog_204 group by sid having count(distinct pid) = (select count(distinct pid)
from parts_204));
```

sname
acne

Find the snames of suppliers who supply every red part.

```
select distinct sname from supplier_204, parts_204, catalog_204
where supplier_204.sid = catalog_204.sid and parts_204.pid = catalog_204.pid and
parts_204.color="Red";
```

sname
acne
johns

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

```
select pname from parts_204 where pid not in
(select pid from catalog_204 where sid in (select sid from supplier_204 where sname !=
"acne"));
```

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 sid from catalog_204 a where a.cost > (select avg(b.cost) from catalog_204 b where a.pid = b.pid group by b.pid);

sid
10002
10004

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

select pid, sname from catalog_204 a, supplier_204
where a.cost = (select max(b.cost) from catalog_204 b where a.pid = b.pid group by b.pid) and supplier_204.sid = a.sid;

pid	sname
20001	acne
20004	acne
20005	acne
20001	johns
20002	johns
20003	reliance

NoSQL Lab 1

Question

(Week 8)

Perform the following DB operations using MongoDB.

1. Create a database “Student” with the following attributes Rollno, Age, ContactNo, Email-Id.
2. Insert appropriate values
3. Write query to update Email-Id of a student with rollno 10.
4. Replace the student name from “ABC” to “FEM” of rollno 11.
5. Export the created table into local file system
6. Drop the table
7. Import a given csv dataset from local file system into mongodb collection.

Create database

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

Create table & Inserting Values to the table

db.Student.insertMany([{**rollno:**1,**age:**21,**cont:**9876,**email:"prannay@gmail.com"**},{**rollno:**2,**age:**22,**cont:**9976,**email:"sohan@gmail.com"**} ,

{**rollno:**3,**age:**21,**cont:**5576,**email:"farhan@gmail.com"**} ,

{**rollno:**4,**age:**20,**cont:**4476,**email:"sakshi@gmail.com"**},{**rollno:**5,**age:**23,**cont:**2276,**email:"sanika@gmail.com"**}]) ;

Structure of Table

db.Student.find();

```
test> db.Student.find();
[
  {
    _id: ObjectId('65e36fda5b3b1935aac1fe45'),
    rollno: 1,
    age: 21,
    cont: 9876,
    email: 'prannay@gmail.com'
  },
  {
    _id: ObjectId('65e36fda5b3b1935aac1fe46'),
    rollno: 2,
    age: 22,
    cont: 9976,
    email: 'sohan@gmail.com'
  },
  {
    _id: ObjectId('65e36fda5b3b1935aac1fe47'),
    rollno: 3,
    age: 21,
    cont: 5576,
    email: 'farhan@gmail.com'
  },
  {
    _id: ObjectId('65e36fda5b3b1935aac1fe48'),
    rollno: 4,
    age: 20,
    cont: 4476,
    email: 'sakshi@gmail.com'
  },
  {
    _id: ObjectId('65e36fda5b3b1935aac1fe49'),
    rollno: 5,
    age: 23,
    cont: 2276,
    email: 'sanika@gmail.com'
  }
]
```

Queries

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

db.Student.update({**rollno**:5},{$**set**:{**email**:"**abhinav@gmail.com**"}}) ;

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

```
test> db.Student.insert({rollno:11,age:22,name:"ABC",cont:2276,email:"madhura@gmail.com"}); db.Student.update({rollno:11,name:"ABC"},{$set:{name:"FEM"}})
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

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

mongoexport
*U*sers \ *bhoomika* *D*ocuments\ *t*est.Students.json
mongodb+srv://204:<password>@cluster0.xbmgopf.mongodb.net/test --collection=Student -- out C:

- **Drop the table**

db.Student.drop();

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

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

mongoimport
*U*sers *bhoomika**D*ocuments*t*est.Students.json
mongodb+srv://204:<password>@cluster0.xbmgopf.mongodb.net/test
--collection=Student -- type json -file C: db.Student.find();

```
test> db.Student.find();
[
  {
    _id: ObjectId('65e36fda5b3b1935aac1fe45'),
    rollno: 1,
    age: 21,
    cont: 9876,
    email: 'prannay@gmail.com'
  },
  {
    _id: ObjectId('65e36fda5b3b1935aac1fe46'),
    rollno: 2,
    age: 22,
    cont: 9976,
    email: 'sohan@gmail.com'
  },
  {
    _id: ObjectId('65e36fda5b3b1935aac1fe47'),
    rollno: 3,
    age: 21,
    cont: 5576,
    email: 'farhan@gmail.com'
  },
  {
    _id: ObjectId('65e36fda5b3b1935aac1fe48'),
    rollno: 4,
    age: 20,
    cont: 4476,
    email: 'sakshi@gmail.com'
  },
  {
    _id: ObjectId('65e36fda5b3b1935aac1fe49'),
    rollno: 5,
    age: 23,
    cont: 2276,
    email: 'abhinav@gmail.com'
  },
  {
    _id: ObjectId('65e3e2175b3b1935aac1fe4a'),
    rollno: 11,
    age: 22,
    name: 'FEM',
    cont: 2276,
    email: 'madhura@gmail.com'
  }
]
```

NoSQL Lab 2

(Week 9)

Question

Perform the following DB operations using MongoDB.

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

Cust_id, Acc_Bal, Acc_Type

2. Insert at least 5 values into the table

3. Write a query to display those records whose total account balance is greater than 1200 of account type ‘Checking’ for each customer_id. 4. Determine Minimum and Maximum account balance for each customer_id.

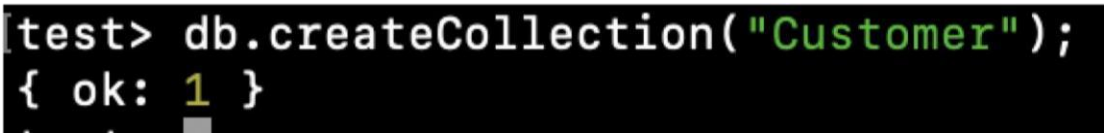
5. Export the created collection into local file system

6. Drop the table

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

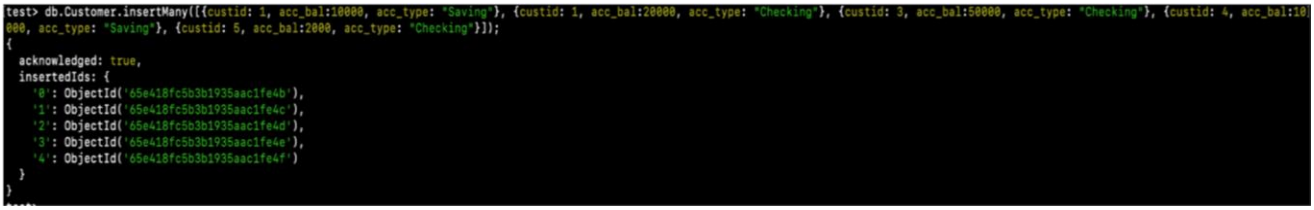
Create Table:

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



Inserting Values:

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



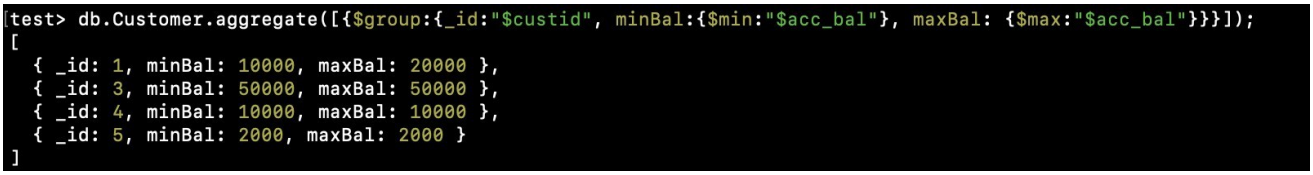
Queries:

- **Finding all checking accounts with balance greater than 12000**

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

- **Finding the maximum and minimum balance of each customer**

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



- **Exporting the collection to a json file** mongoexport

\Users\bhoomika \Documents\ test.Customer.json
mongodb+srv://204:<password>@cluster0.xbmgopf.mongodb.net/test --collection=Customer
-- out C:

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

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

- **Exporting from a json file to the collection** mongoimport

\Users \bhoomika\Documents \test.Customer.json mongodb+srv://204:<password>@cluster0.xbmgopf.mongodb.net/test --collection=Customer -- type json -file C:

db.Customer.find();

```
test> db.Customer.find();
[
  {
    _id: ObjectId('65e418fc5b3b1935aac1fe4b'),
    custid: 1,
    acc_bal: 10000,
    acc_type: 'Saving'
  },
  {
    _id: ObjectId('65e418fc5b3b1935aac1fe4c'),
    custid: 1,
    acc_bal: 20000,
    acc_type: 'Checking'
  },
  {
    _id: ObjectId('65e418fc5b3b1935aac1fe4d'),
    custid: 3,
    acc_bal: 50000,
    acc_type: 'Checking'
  },
  {
    _id: ObjectId('65e418fc5b3b1935aac1fe4e'),
    custid: 4,
    acc_bal: 10000,
    acc_type: 'Saving'
  },
  {
    _id: ObjectId('65e418fc5b3b1935aac1fe4f'),
    custid: 5,
    acc_bal: 2000,
    acc_type: 'Checking'
  }
]
```

NoSQL Lab 3

Question

(Week 10)

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

2. Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns.
3. Write a MongoDB query to find the restaurant Id, name, town and cuisine for those restaurants which achieved a score which is not more than 10.

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

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

Creating Table:

db.createCollection("**Restaurant**") ;

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

Inserting Values:

```
db.Restraunt.insertMany([
  {

    "address": {

     "building": "1007",

     "coord": [-73.856077, 48.848447],
```

```
    "street": "Morris Park Ave",

    "zipcode": "18462", "borough":
    "Bronx"

   },
   "cuisine": "Bakery",
   "grades": [
    {"date": new Date("2014-03-03"), "grade": "A", "score": 2},
    {"date": new Date("2013-09-11"), "grade": "A", "score": 6},
    {"date": new Date("2013-01-24"), "grade": "A", "score": 10},
    {"date": new Date("2011-11-23"), "grade": "A", "score": 9},
    {"date": new Date("2011-03-10"), "grade": "B", "score": 14}
   ],
   "name": "Morris Park Bake Shop",
   "restaurant_id": "30075445"
  },
{
   "address": {
    "building": "2001",
    "coord": [-74.005941, 40.712776],
    "street": "Broadway",
    "zipcode": "10001",
    "borough": "Manhattan"
   },
   "cuisine": "Italian",
   "grades": [
    {"date": new Date("2015-08-20"), "grade": "A", "score": 8},
    {"date": new Date("2014-06-10"), "grade": "B", "score": 4},
```

```
    {"date": new Date("2013-12-15"), "grade": "A", "score": 11},  {"date": new Date("2012-09-30"), "grade":
    "A", "score": 9},

    {"date": new Date("2011-05-12"), "grade": "A", "score": 12}

   ],

   "name": "Pasta Paradise",

   "restaurant_id": "40092138"
  },

{

   "address": {

    "building": "3003",

    "coord": [-118.243685, 34.052235],

    "street": "Hollywood Blvd",

    "zipcode": "90028",

    "borough": "Los Angeles"

   },

   "cuisine": "Mexican",

   "grades": [

    {"date": new Date("2016-04-15"), "grade": "A", "score": 9},

    {"date": new Date("2015-12-05"), "grade": "B", "score": 6},

    {"date": new Date("2014-09-20"), "grade": "A", "score": 11},

    {"date": new Date("2013-06-18"), "grade": "A", "score": 8},

    {"date": new Date("2012-02-10"), "grade": "A", "score": 10}

   ],

   "name": "Sizzling Tacos",

   "restaurant_id": "50065432"

  },

{
   "address": {
```

```
    "building": "4004",

    "coord": [77.209021, 28.613939],

    "street": "Connaught Place",

    "zipcode": "110001",
    "borough": "New Delhi"

   },

   "cuisine": "Indian",

   "grades": [

    {"date": new Date("2019-10-25"), "grade": "A", "score": 8},

    {"date": new Date("2018-07-15"), "grade": "B", "score": 5},

    {"date": new Date("2017-04-30"), "grade": "A", "score": 10},

    {"date": new Date("2016-01-12"), "grade": "A", "score": 9},

    {"date": new Date("2015-05-20"), "grade": "A", "score": 12}

   ],

   "name": "Spice Delight",

   "restaurant_id": "60098765"

  },

{

   "address": {

    "building": "5005",

    "coord": [76.780253, 30.728592],

    "street": "Balle Balle Lane",

    "zipcode": "160022",

    "borough": "Chandigarh"

   },

   "cuisine": "Punjabi", "grades":

   [
```

{"date": new Date("2020-12-10"), "grade": "A", "score": 9}, {"date": new Date("2019-08-25"), "grade": "B", "score": 7},

{"date": new Date("2018-04-15"), "grade": "A", "score": 11}, {"date": new Date("2017-01-22"), "grade": "A", "score": 8},

{"date": new Date("2016-06-30"), "grade": "A", "score": 10}

],

"name": "Pind Flavors",

"restaurant_id": "70087654"

},

{

"address": {

"building": "6006",

"coord": [77.594562, 12.971598],

"street": "Vidyarthi Bhavan Road",

"zipcode": "560004",

"borough": "Bangalore"

},

"cuisine": "Kannadiga",

"grades": [

{"date": new Date("2021-09-18"), "grade": "A", "score": 8},

{"date": new Date("2020-05-12"), "grade": "B", "score": 6},

{"date": new Date("2019-02-28"), "grade": "A", "score": 10},

{"date": new Date("2018-11-15"), "grade": "A", "score": 9},

{"date": new Date("2017-07-05"), "grade": "A", "score": 12}

],

"name": "Namma Oota",

"restaurant_id": "80076543"

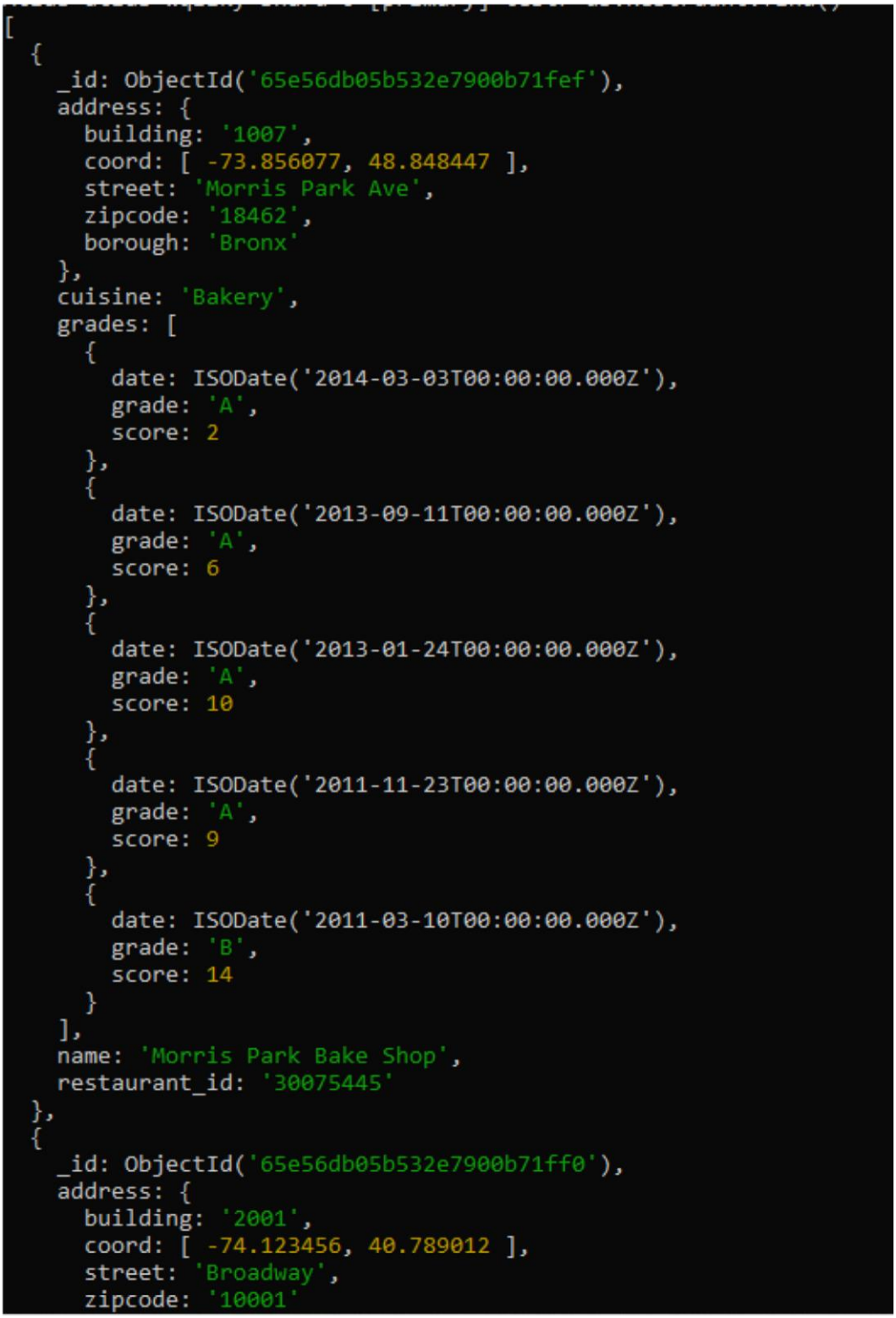
```
 },


{
  "address": {
   "building": "7007",
   "coord": [73.856743, 18.520430],
   "street": "Pune-Nashik Highway",
   "zipcode": "411001",
   "borough": "Pune"
  },
  "cuisine": "Maharashtrian",
  "grades": [
   {"date": new Date("2022-05-20"), "grade": "A", "score": 9},
   {"date": new Date("2021-01-15"), "grade": "B", "score": 7},
   {"date": new Date("2020-08-10"), "grade": "A", "score": 11},
   {"date": new Date("2019-04-25"), "grade": "A", "score": 8},
   {"date": new Date("2018-10-12"), "grade": "A", "score": 10}
  ],
  "name": "Misal Junction",
  "restaurant_id": "90065432"
 },
{
  "address": {
   "building": "7007",
   "coord": [73.856743, 18.520430],
   "street": "Shivaji Road",
   "zipcode": "411001",
   "borough": "Pune"
```

```
  },
  "cuisine": "Maharashtrian",

  "grades": [
   {"date": new Date("2022-04-30"), "grade": "A", "score": 9},
   {"date": new Date("2021-10-15"), "grade": "B", "score": 7}, {"date": new Date("2020-06-28"), "grade":
   "A", "score": 12},
   {"date": new Date("2019-03-12"), "grade": "A", "score": 8},
   {"date": new Date("2018-08-20"), "grade": "A", "score": 10}
  ],
  "name": "Vyanjan Vihar",
  "restaurant_id": "90065432"
 },
{
  "address": {
   "building": "8008",
   "coord": [79.312929, 9.288536],
   "street": "Temple Road",
   "zipcode": "623526",
   "borough": "Rameshwaram"
  },
  "cuisine": "Cafe",
  "grades": [
   {"date": new Date("2021-07-22"), "grade": "A", "score": 8},
   {"date": new Date("2020-02-10"), "grade": "B", "score": 5},
   {"date": new Date("2019-09-05"), "grade": "A", "score": 10},
   {"date": new Date("2018-04-18"), "grade": "A", "score": 9},
   {"date": new Date("2017-11-30"), "grade": "A", "score": 12}
  ],
```

```
  "name": "Rameshwaram Retreat",

  "restaurant_id": "10076543"
 },

 {

  "address": {

   "building": "9009",
   "coord": [80.270718, 13.082680],

   "street": "Anna Salai",

   "zipcode": "600002",

   "borough": "Chennai"

  },

  "cuisine": "Tamil",

  "grades": [

   {"date": new Date("2022-01-15"), "grade": "A", "score": 8},

   {"date": new Date("2021-06-05"), "grade": "B", "score": 6},

   {"date": new Date("2020-11-20"), "grade": "A", "score": 11},

   {"date": new Date("2019-08-12"), "grade": "A", "score": 9},

   {"date": new Date("2018-03-25"), "grade": "A", "score": 10}

  ],

  "name": "Tamil Delicacies",

  "restaurant_id": "11076543"

}]);
```

Queries

1) db.Restraunt.find()



```
},
{
  _id: ObjectId('65e56db05b532e7900b71ff1'),
  address: {
    building: '3003',
    coord: [ -118.243685, 34.052235 ],
    street: 'Hollywood Blvd',
    zipcode: '90028',
    borough: 'Los Angeles'
  },
  cuisine: 'Mexican',
  grades: [
    {
      date: ISODate('2016-04-15T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2015-12-05T00:00:00.000Z'),
      grade: 'B',
      score: 6
    },
    {
      date: ISODate('2014-09-20T00:00:00.000Z'),
      grade: 'A',
      score: 11
    },
    {
      date: ISODate('2013-06-18T00:00:00.000Z'),
      grade: 'A',
      score: 8
    },
    {
      date: ISODate('2012-02-10T00:00:00.000Z'),
      grade: 'A',
      score: 10
    }
  ],
  name: 'Sizzling Tacos',
  restaurant_id: '50065432'
},
{
  _id: ObjectId('65e56ec65b532e7900b71ff2'),
  address: {
    building: '4004',
    coord: [ 77.209021, 28.613939 ],
    street: 'Connaught Place',
    zipcode: '110001',
    borough: 'New Delhi'
  },
  cuisine: 'Indian',
  grades: [
    {
      date: ISODate('2019-10-25T00:00:00.000Z'),
      grade: 'A',
      score: 8
    },
    {
      date: ISODate('2018-07-15T00:00:00.000Z'),
      grade: 'B',
      score: 5
    },
    {
```

```
{
  _id: ObjectId('65e56ec65b532e7900b71ff3'),
  address: {
    building: '5005',
    coord: [ 76.780253, 30.728592 ],
    street: 'Balle Balle Lane',
    zipcode: '160022',
    borough: 'Chandigarh'
  },
  cuisine: 'Punjabi',
  grades: [
    {
      date: ISODate('2020-12-10T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2019-08-25T00:00:00.000Z'),
      grade: 'B',
      score: 7
    },
    {
      date: ISODate('2018-04-15T00:00:00.000Z'),
      grade: 'A',
      score: 11
    },
    {
      date: ISODate('2017-01-22T00:00:00.000Z'),
      grade: 'A',
      score: 8
    },
    {
      date: ISODate('2016-06-30T00:00:00.000Z'),
      grade: 'A',
      score: 10
    }
  ],
  name: 'Pind Flavors',
  restaurant_id: '70087654'
},
{
  _id: ObjectId('65e56ec65b532e7900b71ff4'),
  address: {
    building: '6006',
    coord: [ 77.594562, 12.971598 ],
    street: 'Vidyarthi Bhavan Road',
    zipcode: '560004',
    borough: 'Bangalore'
  },
  cuisine: 'Kannadiga',
  grades: [
    {
      date: ISODate('2021-09-18T00:00:00.000Z'),
      grade: 'A',
      score: 8
    },
    {
      date: ISODate('2020-05-12T00:00:00.000Z'),
      grade: 'B',
      score: 6
    },
    {
      date: ISODate('2019-02-28T00:00:00.000Z'),
```

```
      date: ISODate('2017-07-05T00:00:00.000Z'),
      grade: 'A',
      score: 12
    }
  ],
  name: 'Namma Oota',
  restaurant_id: '80076543'
},
{
  _id: ObjectId('65e56ec65b532e7900b71ff5'),
  address: {
    building: '7007',
    coord: [ 73.856743, 18.52043 ],
    street: 'Pune-Nashik Highway',
    zipcode: '411001',
    borough: 'Pune'
  },
  cuisine: 'Maharashtrian',
  grades: [
    {
      date: ISODate('2022-05-20T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2021-01-15T00:00:00.000Z'),
      grade: 'B',
      score: 7
    },
    {
      date: ISODate('2020-08-10T00:00:00.000Z'),
      grade: 'A',
      score: 11
    },
    {
      date: ISODate('2019-04-25T00:00:00.000Z'),
      grade: 'A',
      score: 8
    },
    {
      date: ISODate('2018-10-12T00:00:00.000Z'),
      grade: 'A',
      score: 10
    }
  ],
  name: 'Misal Junction',
  restaurant_id: '90065432'
},
{
  _id: ObjectId('65e56ec65b532e7900b71ff6'),
  address: {
    building: '7007',
    coord: [ 73.856743, 18.52043 ],
    street: 'Shivaji Road',
    zipcode: '411001',
    borough: 'Pune'
  },
  cuisine: 'Maharashtrian',
  grades: [
    {
      date: ISODate('2022-04-30T00:00:00.000Z'),
      grade: 'A',
      score: 9
```

2) db.Restraunt.find().sort({ "name": -1 });

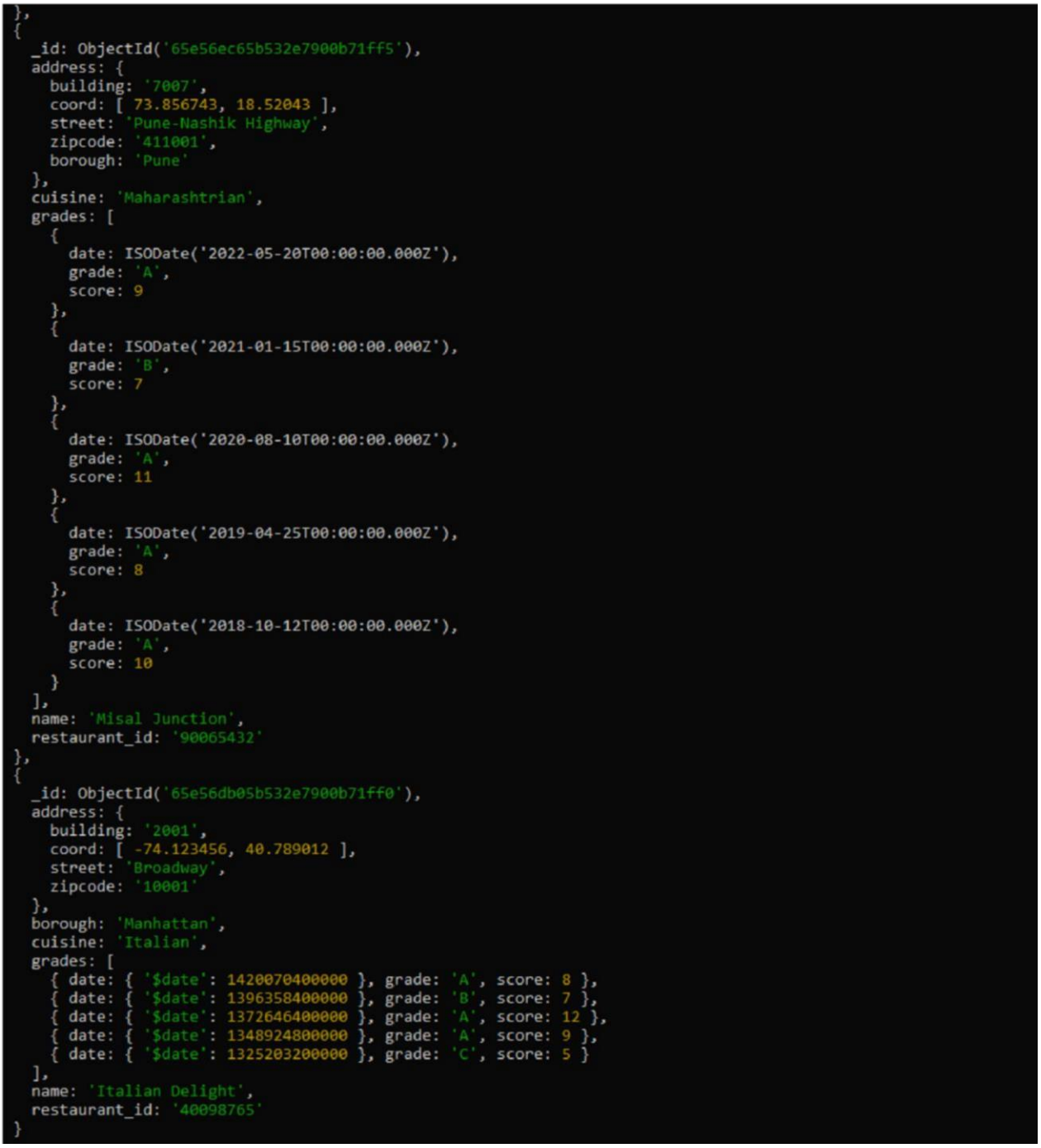
```
[
  {
    _id: ObjectId('65e56ec65b532e7900b71ff6'),
    address: {
      building: '7007',
      coord: [ 73.856743, 18.52043 ],
      street: 'Shivaji Road',
      zipcode: '411001',
      borough: 'Pune'
    },
    cuisine: 'Maharashtrian',
    grades: [
      {
        date: ISODate('2022-04-30T00:00:00.000Z'),
        grade: 'A',
        score: 9
      },
      {
        date: ISODate('2021-10-15T00:00:00.000Z'),
        grade: 'B',
        score: 7
      },
      {
        date: ISODate('2020-06-28T00:00:00.000Z'),
        grade: 'A',
        score: 12
      },
      {
        date: ISODate('2019-03-12T00:00:00.000Z'),
        grade: 'A',
        score: 8
      },
      {
        date: ISODate('2018-08-20T00:00:00.000Z'),
        grade: 'A',
        score: 10
      }
    ],
    name: 'Vyanjan Vihar',
    restaurant_id: '90065432'
  },
  {
    _id: ObjectId('65e56ec65b532e7900b71ff7'),
    address: {
      building: '9009',
      coord: [ 80.270718, 13.08268 ],
      street: 'Anna Salai',
      zipcode: '600002',
      borough: 'Chennai'
    },
    cuisine: 'Tamil',
    grades: [
      {
        date: ISODate('2022-01-15T00:00:00.000Z'),
        grade: 'A',
```

```
  },
  cuisine: 'Tamil',
  grades: [
    {
      date: ISODate('2022-01-15T00:00:00.000Z'),
      grade: 'A',
      score: 8
    },
    {
      date: ISODate('2021-06-05T00:00:00.000Z'),
      grade: 'B',
      score: 6
    },
    {
      date: ISODate('2020-11-20T00:00:00.000Z'),
      grade: 'A',
      score: 11
    },
    {
      date: ISODate('2019-08-12T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2018-03-25T00:00:00.000Z'),
      grade: 'A',
      score: 10
    }
  ],
  name: 'Tamil Delicacies',
  restaurant_id: '11076543'
},
{
  _id: ObjectId('65e56ec65b532e7900b71ff2'),
  address: {
    building: '4004',
    coord: [ 77.209021, 28.613939 ],
    street: 'Connaught Place',
    zipcode: '110001',
    borough: 'New Delhi'
  },
  cuisine: 'Indian',
  grades: [
    {
      date: ISODate('2019-10-25T00:00:00.000Z'),
      grade: 'A',
      score: 8
    },
    {
      date: ISODate('2018-07-15T00:00:00.000Z'),
      grade: 'B',
      score: 5
    },
    {
      date: ISODate('2017-04-30T00:00:00.000Z'),
      grade: 'A',
      score: 10
    },
    {
      date: ISODate('2016-01-12T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
```

```
      score: 12
    }
  ],
  name: 'Spice Delight',
  restaurant_id: '60098765'
},
{
  _id: ObjectId('65e56db05b532e7900b71ff1'),
  address: {
    building: '3003',
    coord: [ -118.243685, 34.052235 ],
    street: 'Hollywood Blvd',
    zipcode: '90028',
    borough: 'Los Angeles'
  },
  cuisine: 'Mexican',
  grades: [
    {
      date: ISODate('2016-04-15T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2015-12-05T00:00:00.000Z'),
      grade: 'B',
      score: 6
    },
    {
      date: ISODate('2014-09-20T00:00:00.000Z'),
      grade: 'A',
      score: 11
    },
    {
      date: ISODate('2013-06-18T00:00:00.000Z'),
      grade: 'A',
      score: 8
    },
    {
      date: ISODate('2012-02-10T00:00:00.000Z'),
      grade: 'A',
      score: 10
    }
  ],
  name: 'Sizzling Tacos',
  restaurant_id: '50065432'
},
{
  _id: ObjectId('65e56ec65b532e7900b71ff3'),
  address: {
    building: '5005',
    coord: [ 76.780253, 30.728592 ],
    street: 'Balle Balle Lane',
    zipcode: '160022',
    borough: 'Chandigarh'
  },
  cuisine: 'Punjabi',
  grades: [
    {
      date: ISODate('2020-12-10T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
```

```
      grade: 'A',
      score: 10
    }
  ],
  name: 'Pind Flavors',
  restaurant_id: '70087654'
},
{
  _id: ObjectId('65e56ec65b532e7900b71ff4'),
  address: {
    building: '6006',
    coord: [ 77.594562, 12.971598 ],
    street: 'Vidyarthi Bhavan Road',
    zipcode: '560004',
    borough: 'Bangalore'
  },
  cuisine: 'Kannadiga',
  grades: [
    {
      date: ISODate('2021-09-18T00:00:00.000Z'),
      grade: 'A',
      score: 8
    },
    {
      date: ISODate('2020-05-12T00:00:00.000Z'),
      grade: 'B',
      score: 6
    },
    {
      date: ISODate('2019-02-28T00:00:00.000Z'),
      grade: 'A',
      score: 10
    },
    {
      date: ISODate('2018-11-15T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2017-07-05T00:00:00.000Z'),
      grade: 'A',
      score: 12
    }
  ],
  name: 'Namma Oota',
  restaurant_id: '80076543'
},
{
  _id: ObjectId('65e56db05b532e7900b71fef'),
  address: {
    building: '1007',
    coord: [ -73.856077, 48.848447 ],
    street: 'Morris Park Ave',
```

```
  name: 'Namma Oota',
  restaurant_id: '80076543'
},
{
  _id: ObjectId('65e56db05b532e7900b71fef'),
  address: {
    building: '1007',
    coord: [ -73.856077, 48.848447 ],
    street: 'Morris Park Ave',
    zipcode: '18462',
    borough: 'Bronx'
  },
  cuisine: 'Bakery',
  grades: [
    {
      date: ISODate('2014-03-03T00:00:00.000Z'),
      grade: 'A',
      score: 2
    },
    {
      date: ISODate('2013-09-11T00:00:00.000Z'),
      grade: 'A',
      score: 6
    },
    {
      date: ISODate('2013-01-24T00:00:00.000Z'),
      grade: 'A',
      score: 10
    },
    {
      date: ISODate('2011-11-23T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2011-03-10T00:00:00.000Z'),
      grade: 'B',
      score: 14
    }
  ],
  name: 'Morris Park Bake Shop',
  restaurant_id: '30075445'
},
{
  _id: ObjectId('65e56ec65b532e7900b71ff5'),
  address: {
    building: '7007',
    coord: [ 73.856743, 18.52043 ],
    street: 'Pune-Nashik Highway',
    zipcode: '411001',
    borough: 'Pune'
  },
  cuisine: 'Maharashtrian',
  grades: [
    {
      date: ISODate('2022-05-20T00:00:00.000Z'),
      grade: 'A',
      score: 9
    },
    {
      date: ISODate('2021-01-15T00:00:00.000Z'),
      grade: 'B',
      score: 7
```



3) db.Restraunt.find(

{ "grades.score": { $lte: 10 } },

{ _id: 1, name: 1, town: 1, cuisine: 1, restaurant_id: 1 }

);

```
Atlas atlas-wqilky-shard-0 [primary] test> db.Restraunt.find(
...   { "grades.score": { $lte: 10 } },
...   { _id: 1, name: 1, town: 1, cuisine: 1, restaurant_id: 1 }
... );
[
  {
    _id: ObjectId('65e56db05b532e7900b71fef'),
    cuisine: 'Bakery',
    name: 'Morris Park Bake Shop',
    restaurant_id: '30075445'
  },
  {
    _id: ObjectId('65e56db05b532e7900b71ff0'),
    cuisine: 'Italian',
    name: 'Italian Delight',
    restaurant_id: '40098765'
  },
  {
    _id: ObjectId('65e56db05b532e7900b71ff1'),
    cuisine: 'Mexican',
    name: 'Sizzling Tacos',
    restaurant_id: '50065432'
  },
  {
    _id: ObjectId('65e56ec65b532e7900b71ff2'),
    cuisine: 'Indian',
    name: 'Spice Delight',
    restaurant_id: '60098765'
  },
  {
    _id: ObjectId('65e56ec65b532e7900b71ff3'),
    cuisine: 'Punjabi',
    name: 'Pind Flavors',
    restaurant_id: '70087654'
  },
  {
    _id: ObjectId('65e56ec65b532e7900b71ff4'),
    cuisine: 'Kannadiga',
    name: 'Namma Oota',
    restaurant_id: '80076543'
  },
  {
    _id: ObjectId('65e56ec65b532e7900b71ff5'),
    cuisine: 'Maharashtrian',
    name: 'Misal Junction',
    restaurant_id: '90065432'
  },
  {
    _id: ObjectId('65e56ec65b532e7900b71ff6'),
    cuisine: 'Maharashtrian',
    name: 'Vyanjan Vihar',
    restaurant_id: '90065432'
  },
  {
    _id: ObjectId('65e56ec65b532e7900b71ff7'),
    cuisine: 'Tamil',
    name: 'Tamil Delicacies',
    restaurant_id: '11076543'
  }
]
```

4) db.Restraunt.aggregate([

{

$unwind: "$grades" },

{

$group: {

```
        _id: "$restaurant_id", name:
        { $first: "$name" }, averageScore: {
        $avg: "$grades.score" } }
    },
    {
        $project: { _id:
        1, name:
        1,
        averageScore: 1
    }
}
]);
```

```
Atlas atlas-wqilky-shard-0 [primary] test> db.Restraunt.aggregate([
...   {
...     $unwind: "$grades"
...   },
...   {
...     $group: {
...       _id: "$restaurant_id",
...       name: { $first: "$name" },
...       averageScore: { $avg: "$grades.score" }
...     }
...   },
...   {
...     $project: {
...       _id: 1,
...       name: 1,
...       averageScore: 1
...     }
...   }
... ]);
[
  { _id: '30075445', name: 'Morris Park Bake Shop', averageScore: 8.2 },
  { _id: '50065432', name: 'Sizzling Tacos', averageScore: 8.8 },
  { _id: '70087654', name: 'Pind Flavors', averageScore: 9 },
  { _id: '80076543', name: 'Namma Oota', averageScore: 9 },
  { _id: '60098765', name: 'Spice Delight', averageScore: 8.8 },
  { _id: '40098765', name: 'Italian Delight', averageScore: 8.2 },
  { _id: '90065432', name: 'Misal Junction', averageScore: 9.1 },
  { _id: '11076543', name: 'Tamil Delicacies', averageScore: 8.8 }
]
```

5) db.Restraunt.find(

```
        { "address.zipcode": { $regex: /^10/ } },
        { _id: 0, name: 1, "address.street": 1, "address.zipcode": 1 }
```

```
Atlas atlas-wqilky-shard-0 [primary] test> db.Restraunt.find(
...   { "address.zipcode": { $regex: /^10/ } },
...   { _id: 0, name: 1, "address.street": 1, "address.zipcode": 1 }
... );
[
  {
    address: { street: 'Broadway', zipcode: '10001' },
    name: 'Italian Delight'
  }
]
```

);