

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.



LAB REPORT

on

Database Management Systems (22CS3PCDBM)

Submitted by

RIA JAIN(1BM21CS163)

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

October-2022 to Feb-2023

BMS College Of Engineering

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering



CERTIFICATE

This is to certify that the Lab work entitled “Database Management Systems (22CS3PCDBM)” carried out by **RIA JAIN (1BM21CS163)**, 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 (22CS3PCDBM) work prescribed for the said degree.

Vikrath BM
Assistant Professor
Department of CSE
BMSCE, Bengaluru

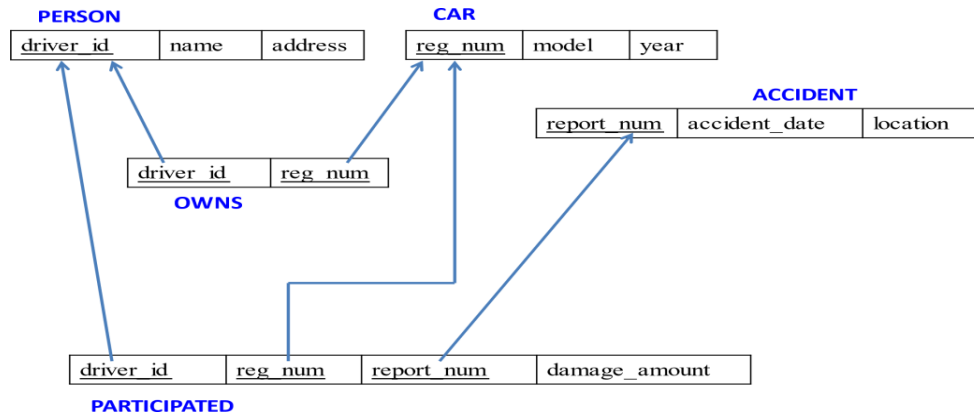
Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE, Bengaluru

Index

Sl. No.	Date	Experiment Title	Page No.
1		Insurance Database	4-11
2		More Queries on Insurance Database	12-21
3		Bank Database	22-29
4		More Queries on Bank Database	30-42
5		Employee Database	43-54
6		More Queries on Employee Database	55-64
7		Supplier Database	65-72
8		Flight Database	73-81
9		NoSQL	82-85

WEEK 1:

Schema Diagram



Insurance Database

PERSON (driver_id: String, name: String, address: String)

CAR (reg_num: String, model: String, year: int)

ACCIDENT (report_num: int, accident_date: date, location: String)

OWNS (driver_id: String, reg_num: String)

PARTICIPATED (driver_id: String, reg_num: String, report_num: int, damage_amount: int)

- Create the above tables by properly specifying the primary keys and the foreign keys.
- Enter at least five tuples for each relation
- Display Accident date and location
- Update the damage amount to 25000 for the car with a specific reg_num (example 'K 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

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

```
create database insurance;
```

```
create table person (  
  driver_id varchar(10),  
  name varchar(30),  
  address varchar(30),  
  primary  
  key(driver_id)  
);  
desc person;
```

```
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)
);

Table description:

```
5 • desc person;  
6  
7 • create table car( reg_num varchar(10), model v  
8   primary key(reg_num)  
9 );
```

< Result Grid Filter Rows: Export: Wrap Cell

	Field	Type	Null	Key	Default	Extra
▶	driver_id	varchar(10)	NO	PRI	NULL	
	name	varchar(30)	YES		NULL	
	address	varchar(30)	YES		NULL	

```
10 • desc car;  
11  
12 • create table accident( report_num int, accio
```

< Result Grid Filter Rows: Export: Wrap C

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

```
14 • desc accident;  
15 • create table owns( driver_id varchar(10), reg_num vai  
16   primary key(driver_id,reg_num),
```

< 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	

```
19 • desc owns;
```

```
20 • create table participated( driver_id var
```

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

```
26 • desc participated;
```

```
27
```

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	

Enter at least five tuples

```
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' );
```

```
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','Jhon','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);
```

```
insert into owns values('A01','KA052250');
insert into owns values('A02','KA053408');
insert into owns values('A03','KA095477');
insert into owns values('A04','KA031181');
insert into owns values('A05','KA041702');
```

```
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 person;
```

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

```
select *from car;
```

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

select *from owns;

Result Grid		Filter Rows:	Edit:
driver_id	reg_num		
A04	KA031181		
A05	KA041702		
A01	KA052250		
A02	KA053408		
A03	KA095477		
NULL	NULL		

person 1 car 2 owns 3 × accident 4 participated 5

Output

select *from accident;

Result Grid		Filter Rows:	Edit:
report_num	accident_date	location	
11	2003-01-01	Mysore road	
12	2004-02-02	South end circle	
13	2003-01-21	Bull temple road	
14	2008-02-17	Mysore road	
15	2004-03-05	Kanakpura road	
NULL	NULL	NULL	

person 1 car 2 owns 3 accident 4 × participated 5

select *from participated;

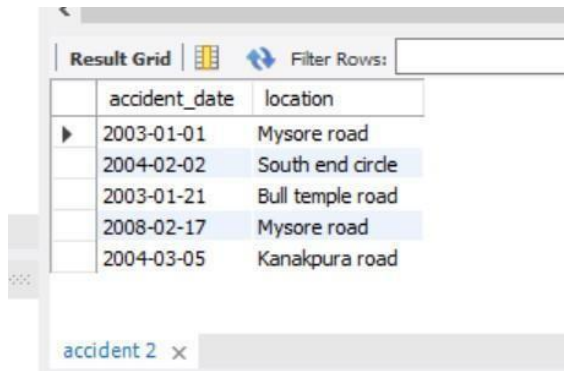
Result Grid		Filter Rows:	Edit:
driver_id	reg_num	report_num	damage_amount
A01	KA052250	11	10000
A02	KA053408	12	50000
A03	KA095477	13	25000
A04	KA031181	14	3000
A05	KA041702	15	5000
NULL	NULL	NULL	NULL

person 1 car 2 owns 3 accident 4 participated 5 ×

Output

Display Accident date and location

SQL> select accident_date,location from accident;



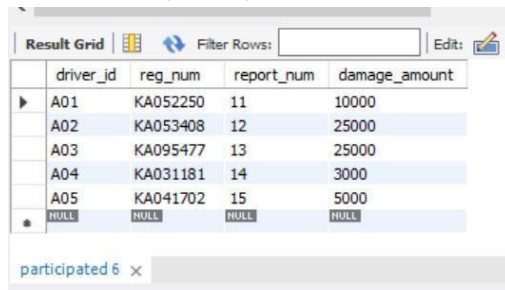
The screenshot shows a database application window with a 'Result Grid' tab. It displays a table with two columns: 'accident_date' and 'location'. There are five rows of data. Below the table, there is a tab labeled 'accident 2' with a close button.

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

1. 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.

SQL> update participated set damage_amount=25000
where reg_num='KA053408' and report_num=12;

select *from participated;



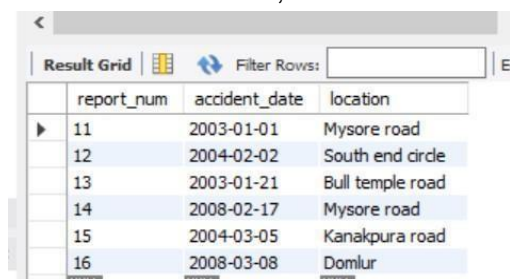
The screenshot shows a database application window with a 'Result Grid' tab. It displays a table with four columns: 'driver_id', 'reg_num', 'report_num', and 'damage_amount'. There are six rows of data, including a row with NULL values. Below the table, there is a tab labeled 'participated 6' with a close button.

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

Add a new accident to the database.

SQL> insert into accident values(16,'2008-03-08','Domlur');

select *from accident;

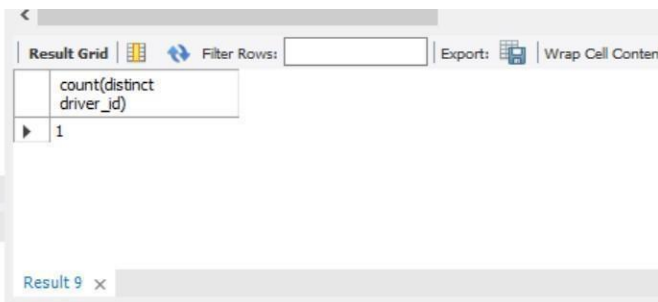


The screenshot shows a database application window with a 'Result Grid' tab. It displays a table with three columns: 'report_num', 'accident_date', and 'location'. There are six rows of data, including the newly added row with report number 16. Below the table, there is a tab labeled 'accident 6' with a close button.

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

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

```
SQL>select count(distinct driver_id)from participated a, accident b
where a.report_num=b.report_num and b.accident_date like '%08%';
```



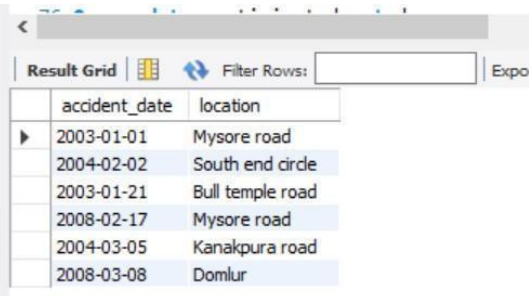
The screenshot shows a database query result grid. The query is: `select count(distinct driver_id)from participated a, accident b where a.report_num=b.report_num and b.accident_date like '%08%';`. The result grid displays a single row with the value 1, indicating that there is one distinct driver ID involved in accidents in 2008.

count(distinct driver_id)
1

TO DO

Display Accident date and location.

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



The screenshot shows a database query result grid for the query: `select accident_date,location from accident;`. The result grid displays a table with two columns: accident_date and location. The data rows are as follows:

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

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

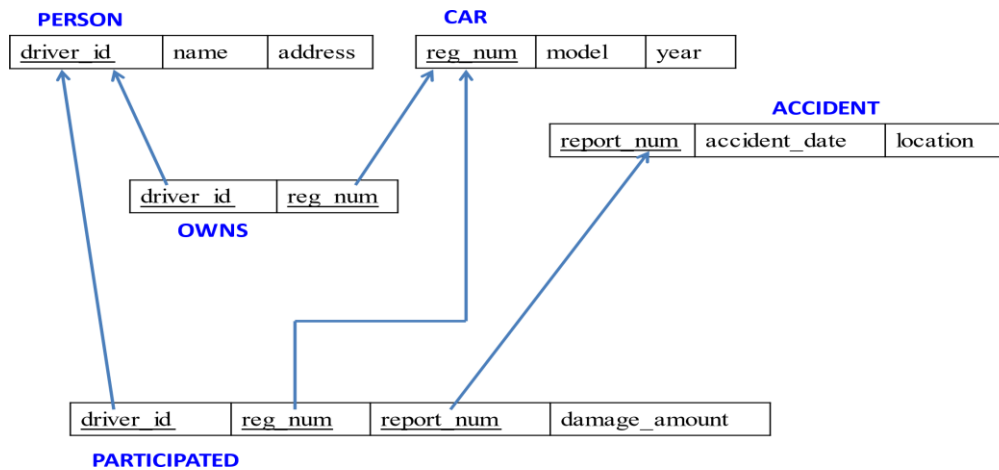
```
SQL>select driver_id from participated
where damage_amount>=25000;
```



The screenshot shows a database query result grid for the query: `select driver_id from participated where damage_amount>=25000;`. The result grid displays a table with one column: driver_id. The data rows are as follows:

driver_id
A02
A03

WEEK 2:



More Queries on INSURANCE DATABASE

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 as done in previous week's lab and Enter at least five tuples for each relation
- Enter at least five tuples for each relation
- Display the entire CAR relation in the ascending order of manufacturing year.
- Find the number of accidents in which cars belonging to a specific model (example 'Lancer') were involved.
- Find the total number of people who owned cars that involved in accidents in 2008.

To Do

- LIST THE ENTIRE PARTICIPATED RELATION IN THE DESCENDING ORDER OF DAMAGE AMOUNT.
- FIND THE AVERAGE DAMAGE AMOUNT
- DELETE THE TUPLE WHOSE DAMAGE AMOUNT IS BELOW THE AVERAGE DAMAGE AMOUNT
- LIST THE NAME OF DRIVERS WHOSE DAMAGE IS GREATER THAN THE AVERAGE DAMAGE AMOUNT.
- FIND MAXIMUM DAMAGE AMOUNT.

```
create database insurance;
```

```
create table person (  
  driver_id varchar(10),  
  name varchar(30),  
  address varchar(30),  
  primary  
  key(driver_id)  
);  
desc person;
```

```
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)
);

Table description:

```

5 • desc person;
6
7 • create table car( reg_num varchar(10), model v
8   primary key(reg_num)
9   );

```

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

Field	Type	Null	Key	Default	Extra
driver_id	varchar(10)	NO	PRI	NULL	
name	varchar(30)	YES		NULL	
address	varchar(30)	YES		NULL	

```

10 • desc car;
11
12 • create table accident( report_num int, accio

```

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

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

```

14 • desc accident;
15 • create table owns( driver_id varchar(10), reg_num vai
16   primary key(driver_id,reg_num),

```

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	

```
19 • desc owns;
```

```
20 • create table participated( driver_id var
```

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

```
26 • desc participated;
```

```
27
```

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

Enter at least five tuples

```
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' );
```

```
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','Jhon','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);
```

```
insert into owns values('A01','KA052250');
insert into owns values('A02','KA053408');
insert into owns values('A03','KA095477');
insert into owns values('A04','KA031181');
```

insert into owns values('A05','KA041702');

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

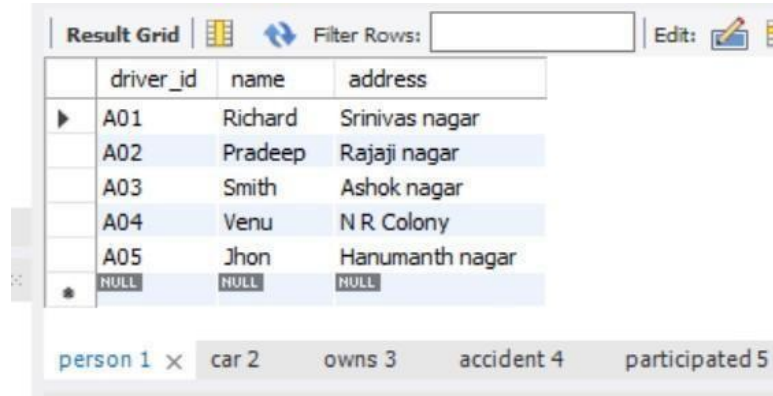
insert into participated values('A02','KA053408',12,50000);

insert into participated values('A03','KA095477',13,25000);

insert into participated values('A04','KA031181',14,3000);

insert into participated values('A05','KA041702',15,5000);

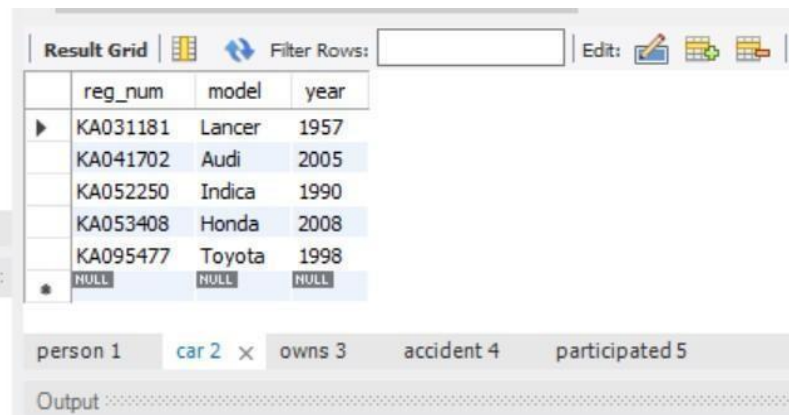
select *from person;



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

person 1 × car 2 owns 3 accident 4 participated 5

select *from car;



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

person 1 car 2 × owns 3 accident 4 participated 5

Output

select *from owns;

The screenshot shows a database query result grid for the 'owns' table. The grid has two columns: 'driver_id' and 'reg_num'. The data is as follows:

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

Below the grid, there is a breadcrumb trail: 'person 1 > car 2 > owns 3 > accident 4 > participated 5'. The 'owns 3' tab is currently selected. An 'Output' section is visible at the bottom.

select *from accident;

The screenshot shows a database query result grid for the 'accident' table. The grid has three columns: 'report_num', 'accident_date', and 'location'. The data is as follows:

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

Below the grid, there is a breadcrumb trail: 'person 1 > car 2 > owns 3 > accident 4 > participated 5'. The 'accident 4' tab is currently selected. An 'Output' section is visible at the bottom.

select *from participated

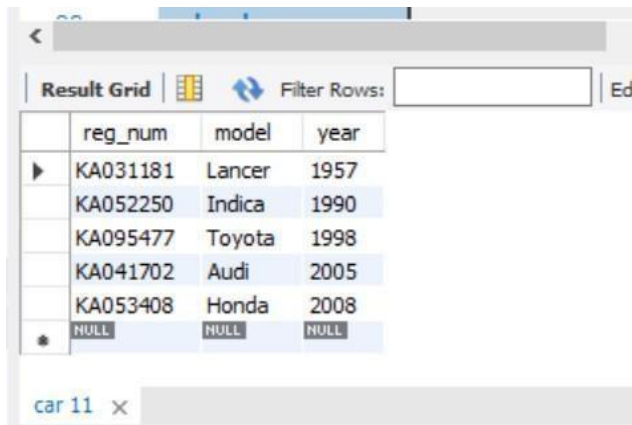
The screenshot shows a database query result grid for the 'participated' table. The grid has four columns: 'driver_id', 'reg_num', 'report_num', and 'damage_amount'. The data is as follows:

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

Below the grid, there is a breadcrumb trail: 'person 1 > car 2 > owns 3 > accident 4 > participated 5 >'. The 'participated 5' tab is currently selected. An 'Output' section is visible at the bottom.

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

```
SQL> select *from car order by year asc
```



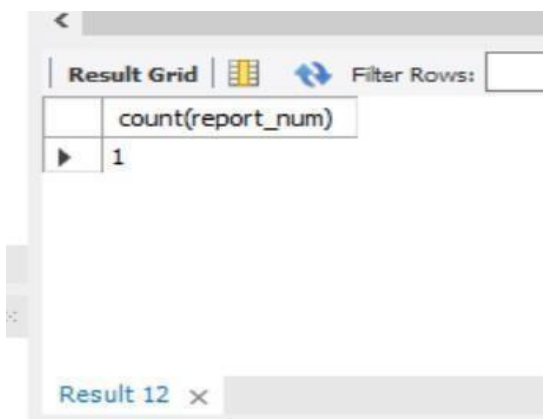
The screenshot shows a database query result grid with the following data:

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

The interface includes a 'Result Grid' tab, a 'Filter Rows' input field, and a tab labeled 'car 11'.

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

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



The screenshot shows a database query result grid with the following data:

	count(report_num)
▶	1

The interface includes a 'Result Grid' tab, a 'Filter Rows' input field, and a tab labeled 'Result 12'.

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

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

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

count(distinct driver_id)
1

Result 9 x

TO DO

List the entire participated relation in descending order of damage amount.

```
SQL> select *from participated
order by damage_amount desc;
```

Result Grid | Filter Rows: | Edit: |

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

participated 13 x

Find the average damage amount.

```
SQL> select
avg(damage_amount) from
```

Result Grid | Filter Rows: | Export: |

avg(damage_amount)
18600.0000

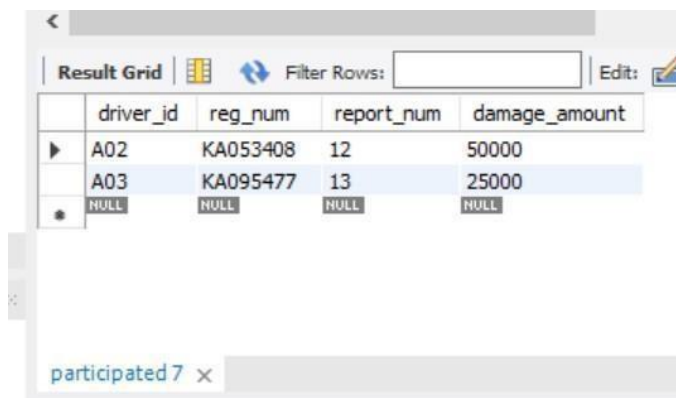
participated;

Delete the tuple whose damage amount is below the average damage amount.

```
SQL> delete from participated
```

```
where damage_amount < (select t.avg1 from (select avg(damage_amount) as avg1 from participated) t);
```

```
select *from participated;
```



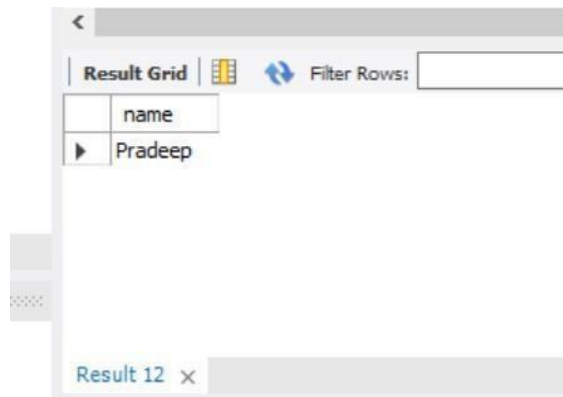
The screenshot shows a database query result grid with the following data:

	driver_id	reg_num	report_num	damage_amount
▶	A02	KA053408	12	50000
	A03	KA095477	13	25000
*	NULL	NULL	NULL	NULL

Below the table, there is a tab labeled "participated 7" with a close button (x).

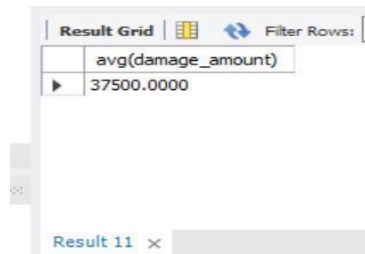
List the name of drivers whose damage is greater than the average damage amount.

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



The screenshot shows a database interface with a 'Result Grid' tab. The grid contains one column labeled 'name' and one row with the value 'Pradeep'. Above the grid is a 'Filter Rows' input field. Below the grid is a tab labeled 'Result 12' with a close button 'x'.

name
Pradeep

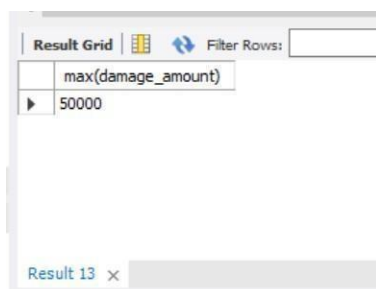


The screenshot shows a database interface with a 'Result Grid' tab. The grid contains one column labeled 'avg(damage_amount)' and one row with the value '37500.0000'. Above the grid is a 'Filter Rows' input field. Below the grid is a tab labeled 'Result 11' with a close button 'x'.

avg(damage_amount)
37500.0000

Find the maximum damage amount

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

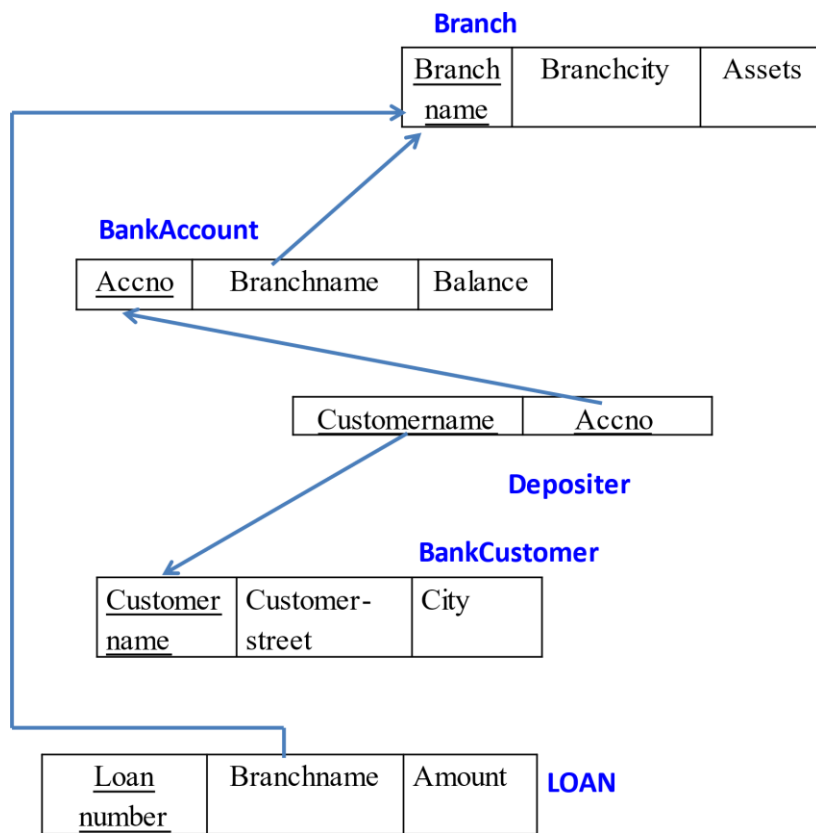


The screenshot shows a database interface with a 'Result Grid' tab. The grid contains one column labeled 'max(damage_amount)' and one row with the value '50000'. Above the grid is a 'Filter Rows' input field. Below the grid is a tab labeled 'Result 13' with a close button 'x'.

max(damage_amount)
50000

Week-3

Bank Database



· To do list

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.

```
create database bank;
```

```
create table branch (  
branchname varchar(20),  
branchcity varchar(20),  
assets int,  
primary key(branchname)  
);
```

```
create table bankaccount (  
accno int,  
branchname varchar(20),  
balance int,  
primary key(accno),  
foreign key (branchname) references branch(branchname)  
);
```

```
create table bankcustomer (  
customername varchar(30),  
customerstreet varchar(20),  
customercity varchar(20),  
primary key(customername)  
);
```

```

create table depositer (
customername varchar(30) ,
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(20),
amount int ,
primary key(loannumber),
foreign key(branchname) references branch(branchname)
);

```

7 • desc branch;

8

9 • create table bankaccount (

<

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

Field	Type	Null	Key	Default	Extra
branchname	varchar(20)	NO	PRI	NULL	
branchcity	varchar(20)	YES		NULL	
assets	int	YES		NULL	

15 • desc bankaccount;

16

<

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

Field	Type	Null	Key	Default	Extra
accno	int	NO	PRI	NULL	
branchname	varchar(20)	YES	MUL	NULL	
balance	int	YES		NULL	


```
20 • desc bankcustomer;
```

```
21
```

```
22 • create table depositer (
```

Field	Type	Null	Key	Default	Extra
customername	varchar(30)	NO	PRI	NULL	
customerstreet	varchar(20)	YES		NULL	
customercity	varchar(20)	YES		NULL	

```
28 • desc depositer;
```

```
29
```

Field	Type	Null	Key	Default	Extra
customername	varchar(30)	NO	PRI	NULL	
accno	int	NO	PRI	NULL	

```
35 • desc loan;
```

```
36
```

Field	Type	Null	Key	Default	Extra
loannumber	int	NO	PRI	NULL	
branchname	varchar(20)	YES	MUL	NULL	
amount	int	YES		NULL	

1. Enter at least five tuples for each relation.

insert into branch values ('SBI_Chamrajpet','banglore',50000);

insert into branch values ('SBI_Residencyroad','banglore',10000);

insert into branch values ('SBI_Shivajinagar','bombay',20000);

insert into branch values ('SBI_Parlimentroad','delhi',10000); insert

into branch values ('SBI_Jantarmantar','delhi',20000);

insert into bankcustomer values ('Avinash','BullTempleRoad','banglore');

insert into bankcustomer values ('Dinesh','BannerghattaRoad','banglore');

```

insert into bankcustomer values ('Mohan','NationalCollegeRoad','banglore');
insert into bankcustomer values ('Nikhil','AkbarRoad','delhi');
insert into bankcustomer values ('Ravi','PrithvirajRoad','delhi');

```

```

insert into bankaccount values (1,'SBI_Chamrajpet',2000);
insert into bankaccount values (2,'SBI_Residencyroad',5000);
insert into bankaccount values (3,'SBI_Shivajinagar',6000);
insert into bankaccount values (4,'SBI_Parlimentroad',9000);
insert into bankaccount values (5,'SBI_Jantarmantar',8000);
insert into bankaccount values (6,'SBI_Shivajinagar',4000);
insert into bankaccount values (8,'SBI_Residencyroad',4000);
insert into bankaccount values (9,'SBI_Parlimentroad',3000);
insert into bankaccount values (10,'SBI_Residencyroad',5000);
insert into bankaccount values (11,'SBI_Jantarmantar',2000);

```

```

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);




```

```

insert into loan values(1,'SBI_Chamrajpet',1000);
insert into loan values(2,'SBI_Residencyroad',2000);
insert into loan values(3,'SBI_Shivajinagar',3000);
insert into loan values(4,'SBI_Parlimentroad',4000);
insert into loan values(5,'SBI_Jantarmantar',5000);

```

select * from branch;

Result Grid			
Filter Rows: <input type="text"/>			
Edit:    Export/			
	branchname	branchcity	assets
▶	SBI_Chamrajpet	banglore	50000
	SBI_Jantarmantar	delhi	20000
	SBI_Parlimentroad	delhi	10000
	SBI_Residencyroad	banglore	10000
	SBI_Shivajinagar	bombay	20000
*	NULL	NULL	NULL
branch 12 x bankaccount 13 bankcustomer 14 depositer 15 loan 16			

select * from bankaccount;

Result Grid			
Filter Rows: <input type="text"/>			
Edit:    Export/			
	accno	branchname	balance
▶	1	SBI_Chamrajpet	2000
	2	SBI_Residencyroad	5000

select * from bankcustomer;

	customername	customerstreet	customercity
▶	Avinash	BullTempleRoad	banglore
	Dinesh	BannerghattaRoad	banglore
	Mohan	NationalCollegeRoad	banglore
	Nikhil	AkbarRoad	delhi
	Ravi	PrithvirajRoad	delhi
*	NULL	NULL	NULL

branch 12 bankaccount 13 bankcustomer 14 × depositer 15 loan 16

select * from depositer;

	customername	accno
▶	Avinash	1
	Dinesh	2
	Nikhil	4
	Ravi	5
	Avinash	8
	Nikhil	9
	Dinesh	10
	Nikhil	11
*	NULL	NULL

branch 12 bankaccount 13 bankcustomer 14 depositer 15 × loan 16

select * from loan;

	loannumber	branchname	amount
▶	1	SBI_Chamrajpet	1000
	2	SBI_Residencyroad	2000
	3	SBI_Shivajinagar	3000
	4	SBI_Parliamentroad	4000
	5	SBI_Jantarmantar	5000
*	NULL	NULL	NULL

branch 12 bankaccount 13 bankcustomer 14 depositer 15 loan 16 ×

WEEK-3 To do list

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

alter table branch

```

rename column assets to assets_in_lakhs ;

select branchname,(assets_in_lakhs/100000)
from branch;

```

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	branchname	(assets_in_lakhs/100000)			
▶	SBI_Chamrajpet	0.5000			
	SBI_Jantarmantar	0.2000			
	SBI_Parliamentroad	0.1000			
	SBI_Residencyroad	0.1000			
	SBI_Shivajinagar	0.2000			

Result 2 x

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

```

select b.branchname,d.customername
from bankaccount b, depositer d where
d.accno=b.accno

group by b.branchname,d.customername
having count(d.customername)>1;

```

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	branchname	customername			
▶	SBI_Residencyroad	Dinesh			
	SBI_Parliamentroad	Nikhil			

Result 3 x

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

```

create view sumloan

as select branchname, sum(amount)
from loan

group by branchname;

select * from sumloan;

```

Result Grid		
	branchname	sum(amount)
▶	SBI_Chamrajpet	1000
	SBI_Jantarmentar	5000
	SBI_Parliamentroad	4000
	SBI_Residencyroad	2000
	SBI_Shivajinagar	3000

sumloan 4 x

On spot Query: Update or add rupees 1000 to acc balance for the customers who are residing in bangalore

```
update bankaccount set balance=(balance+1000)
where accno=any (
    select accno
    from depositer
    where customername=any (
        select customername
        from bankcustomer
        where customercity='bangalore'));
select * from bankaccount;
```

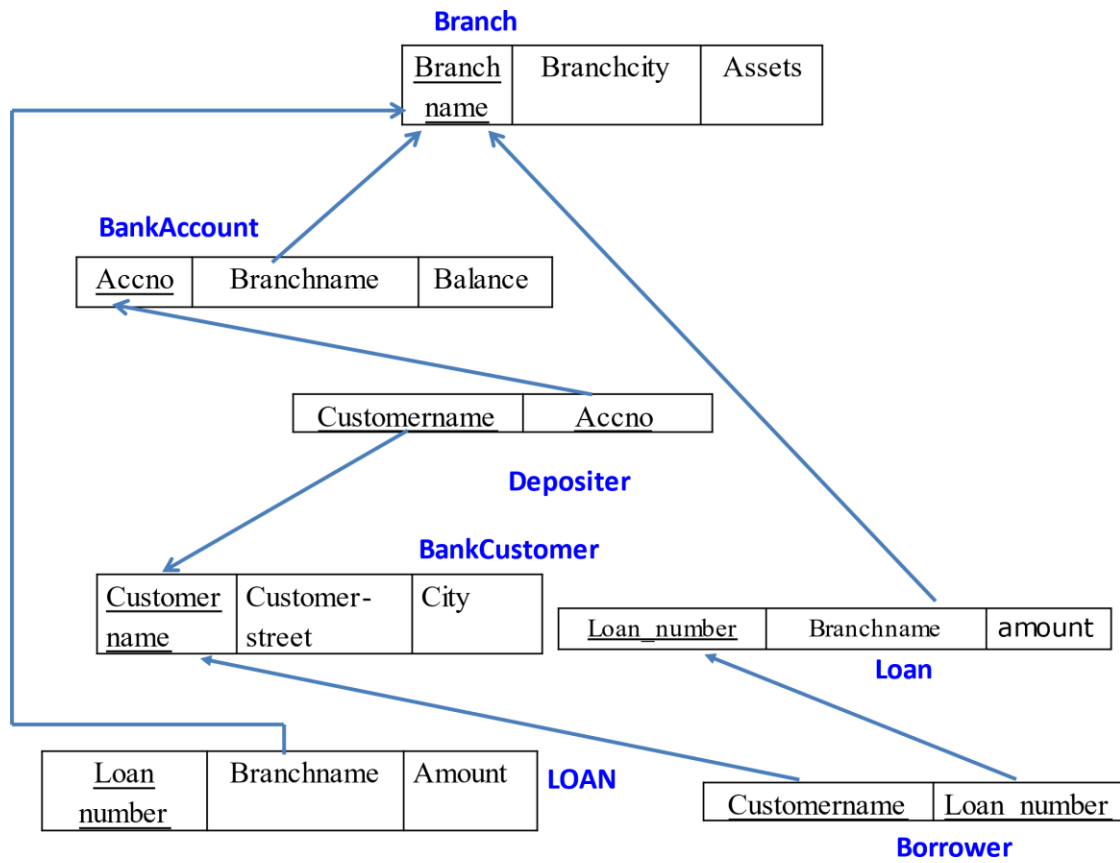
Result Grid			
	accno	branchname	balance
▶	1	SBI_Chamrajpet	3000
	2	SBI_Residencyroad	6000
	3	SBI_Shivajinagar	6000
	4	SBI_Parliamentroad	9000
	5	SBI_Jantarmentar	8000
	6	SBI_Shivajinagar	4000
	8	SBI_Residencyroad	5000
	9	SBI_Parliamentroad	3000
	10	SBI_Residencyroad	6000
	11	SBI_Jantarmentar	2000

bankaccount 5 x

Week-4

More Queries on
Bank Database

Schema Diagram:



Week-4 To do list

1. Find all the customers who have an account at all the branches located in a specific city (Ex. Delhi) .
2. Find all customers who have a loan at the bank but do not have an account.
3. Find all customers who have both an account and a loan at the Bangalore branch
4. Find the names of all branches that have greater assets than all branches located in Bangalore .
5. Demonstrate how you delete all account tuples at every branch located in a specific city (Ex. Bombay).
6. Update the Balance of all accounts by 5%

WEEK-4

BANK DATABASE

```
create database bank2;
```

```
create table branch(  
  branchname varchar(20),  
  branchcity varchar(20),  
  assets int,  
  primary key(branchname)  
);
```

```
create table bankcustomer(  
  customername varchar(20),  
  customerstreet varchar(20),  
  customercity varchar(20),  
  primary key(customername)  
);
```

```
create table bankacc(  
  accno int,  
  branchname varchar(20),  
  balance int,  
  primary key(accno),  
  foreign key(branchname) references branch(branchname)  
  on delete cascade  
  on update cascade  
);
```

```
create table depositer(  
  customername varchar(20),  
  accno int,  
  primary key(customername, accno),  
  foreign key(customername) references bankcustomer(customername),  
  foreign key(accno) references bankacc(accno)  
  on delete cascade  
  on update cascade  
);
```

```
create table loan(  
  loannumber int,  
  branchname varchar(20),
```

```

amount int,
primary key(loannumber),
foreign key(branchname) references branch(branchname)
on delete cascade
on update cascade
);

```

```

create table borrower(
customername varchar(20),
loannumber int,
primary key(loannumber, customername),
foreign key (customername) references bankcustomer(customername),
foreign key (loannumber) references loan(loannumber)
on delete cascade
on update cascade
);

```

6 • desc branch;

7

8 • create table bankcustomer(customername varchar(20)

9 •);

<

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

	Field	Type	Null	Key	Default	Extra
▶	branchname	varchar(20)	NO	PRI	NULL	
	branchcity	varchar(20)	YES		NULL	
	assets	int	YES		NULL	

10 • desc bankcustomer;

11

12 • create table bankacc(accno int,

<

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

	Field	Type	Null	Key	Default	Extra
▶	customername	varchar(20)	NO	PRI	NULL	
	customerstreet	varchar(20)	YES		NULL	
	customercity	varchar(20)	YES		NULL	

```

18 • desc bankacc;
19
20 • create table depositer( customername varchar(20),
21   primary key(customername, accno),

```

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

Field	Type	Null	Key	Default	Extra
accno	int	NO	PRI	NULL	
branchname	varchar(20)	YES	MUL	NULL	
balance	int	YES		NULL	

```

25 • desc depositer;
26
27 • create table loan( loannumber int, branchname

```

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

Field	Type	Null	Key	Default	Extra
customername	varchar(20)	NO	PRI	NULL	
accno	int	NO	PRI	NULL	

```

34 • desc loan;

```

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

Field	Type	Null	Key	Default	Extra
loannumber	int	NO	PRI	NULL	
branchname	varchar(20)	YES	MUL	NULL	
amount	int	YES		NULL	

41 • desc borrower;

<

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

	Field	Type	Null	Key	Default	Extra
▶	customername	varchar(20)	NO	PRI	NULL	
	loannumber	int	NO	PRI	NULL	

```
insert into branch values('SBI_Chamarajpet','bangalore',50000);
insert into branch values('SBI_Residencyroad','bangalore',10000);
insert into branch values('SBI_Shivajinagar','bombay',20000);
insert into branch values('SBI_Parlimentroad','delhi',10000);
insert into branch values('SBI_Jantarmantar','delhi',20000);
insert into branch values('SBI_Mantrimarg','delhi',200000);
```

```
insert into bankcustomer values('Avinash','BullTempleRoad','bangalore');
insert into bankcustomer values('Dinesh','BannerghattaRoad','bangalore');
insert into bankcustomer values('Mohan','NationalCollegeRoad','bangalore');
insert into bankcustomer values('Nikhil','AkbarRoad','delhi');
insert into bankcustomer values('Ravi','PrithvirajRoad','delhi');
```

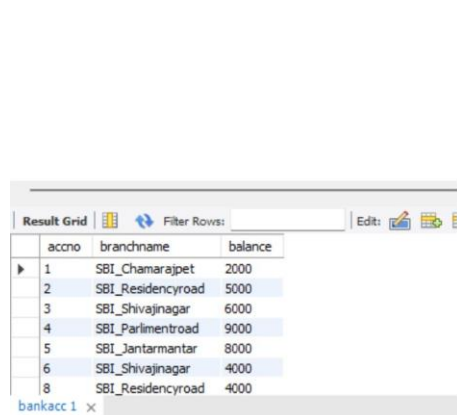
```
insert into bankacc values(1,'SBI_Chamarajpet',2000);
insert into bankacc values(2,'SBI_Residencyroad',5000);
insert into bankacc values(3,'SBI_Shivajinagar',6000);
insert into bankacc values(4,'SBI_Parlimentroad',9000);
insert into bankacc values(5,'SBI_Jantarmantar',8000);
insert into bankacc values(6,'SBI_Shivajinagar',4000);
insert into bankacc values(8,'SBI_Residencyroad',4000);
insert into bankacc values(9,'SBI_Parlimentroad',3000);
insert into bankacc values(10,'SBI_Residencyroad',5000);
insert into bankacc values(11,'SBI_Jantarmantar',2000);
insert into bankacc values(12,'SBI_Mantrimarg',2000);
```

```
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);
insert into depositer values('Nikhil',12);
```

```
insert into loan values(1,'SBI_Chamarajpet',1000);
insert into loan values(2,'SBI_Residencyroad',2000);
insert into loan values(3,'SBI_Shivajinagar',3000);
insert into loan values(4,'SBI_Parlimentroad',4000);
insert into loan values(5,'SBI_Jantarmantar',5000);
```

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



	accno	branchname	balance
1	1	SBI_Chamarajpet	2000
2	2	SBI_Residencyroad	5000
3	3	SBI_Shivajinagar	6000
4	4	SBI_Parlimentroad	9000
5	5	SBI_Jantarmantar	8000
6	6	SBI_Shivajinagar	4000
7	7	SBI_Shivajinagar	4000
8	8	SBI_Residencyroad	4000

Result Grid		
	Filter Rows:	Edit:
customername	customerstreet	customercity
▶ Avinash	BullTempleRoad	bangalore
Dinesh	BannerghattaRoad	bangalore
Mohan	NationalCollegeRoad	bangalore
Nikhil	AkbarRoad	delhi
Ravi	PrithvirajRoad	delhi
* NULL	NULL	NULL

Result Grid		
	Filter Rows:	Edit:
customername	loannumber	
▶ Avinash	1	
Dinesh	2	
Mohan	3	
Nikhil	4	
Ravi	5	
* NULL	NULL	

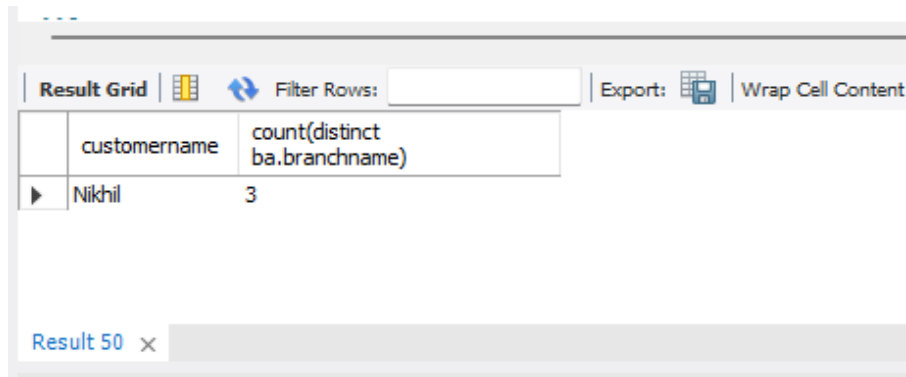
Result Grid		
	Filter Rows:	Edit:
customername	accno	
▶ Avinash	1	
Dinesh	2	
Nikhil	4	
Ravi	5	
Avinash	8	
Nikhil	9	
Dinesh	10	

depositer 1 ×

TODO

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

```
select distinct customername, count(distinct ba.branchname)
from depositer d, bankacc ba, branch b
where d.accno=ba.accno and ba.branchname=b.branchname and b.branchcity='delhi'
group by customername
having count(distinct b.branchname) = (select count(distinct x.branchname)
from branch x inner join bankacc y
on x.branchname=y.branchname
where branchcity='delhi');
```

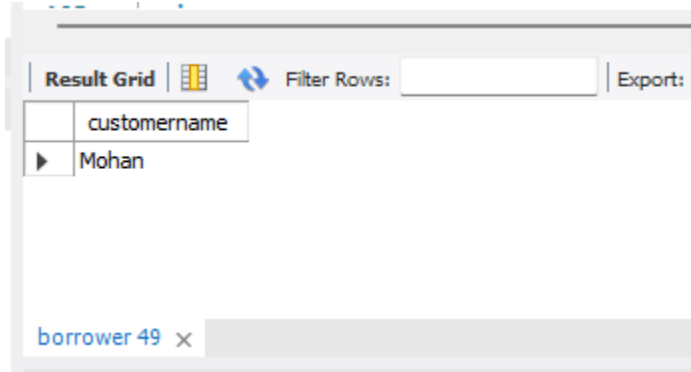


The screenshot shows a database query result grid. The grid has two columns: 'customername' and 'count(distinct ba.branchname)'. There is one row with the value 'Nikhil' in the first column and '3' in the second column. The grid is titled 'Result 50' and has a 'Filter Rows' button and an 'Export' button.

customername	count(distinct ba.branchname)
Nikhil	3

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

```
select customername
from borrower
where customername not in(
select customername
from depositer
);
```

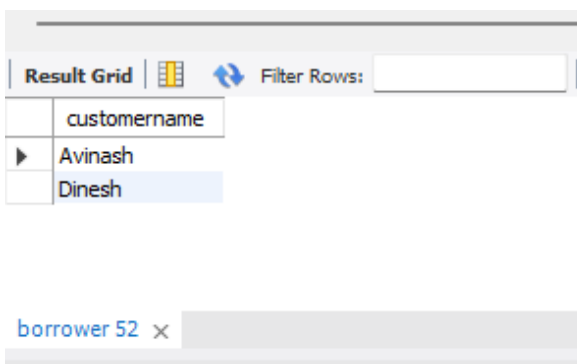


The screenshot shows a database query result grid. At the top, there is a toolbar with 'Result Grid', a grid icon, a 'Filter Rows' button with a double-headed arrow, and an 'Export' button. Below the toolbar is a table with one column labeled 'customername'. The table contains one row with the value 'Mohan'. At the bottom of the grid, there is a tab labeled 'borrower 49' with a close button (x).

customername
Mohan

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

```
select customername
from borrower
where customername in(
select customername from
depositer
where accno= any (
select accno
from bankacc ba inner join branch b on
ba.branchname=b.branchname where
b.branchcity='bangalore' )
);
```

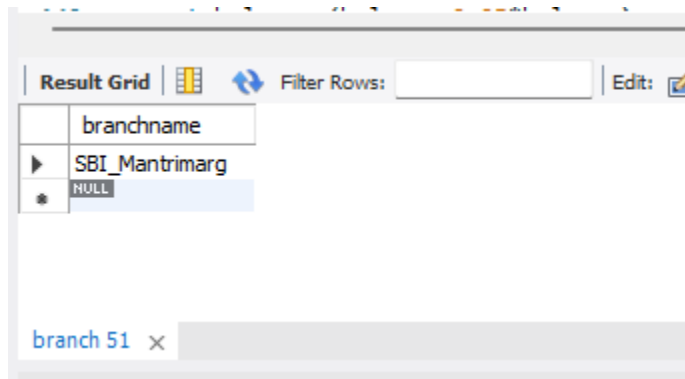


The screenshot shows a database query result grid. At the top, there is a toolbar with 'Result Grid', a grid icon, a 'Filter Rows' button with a double-headed arrow, and an 'Export' button. Below the toolbar is a table with one column labeled 'customername'. The table contains two rows: 'Avinash' and 'Dinesh'. At the bottom of the grid, there is a tab labeled 'borrower 52' with a close button (x).

customername
Avinash
Dinesh

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

```
select branchname
from branch
where assets>
(select sum(assets)
from branch
where branchcity='bangalore');
```



The screenshot shows a database query result grid. The grid has a single column labeled 'branchname'. The first row contains the value 'SBI_Mantrimarg'. Below this row, there is a row with a null value, indicated by a small asterisk icon. The grid is titled 'Result Grid' and has a 'Filter Rows' button. At the bottom, there is a tab labeled 'branch 51'.

branchname
SBI_Mantrimarg
NULL

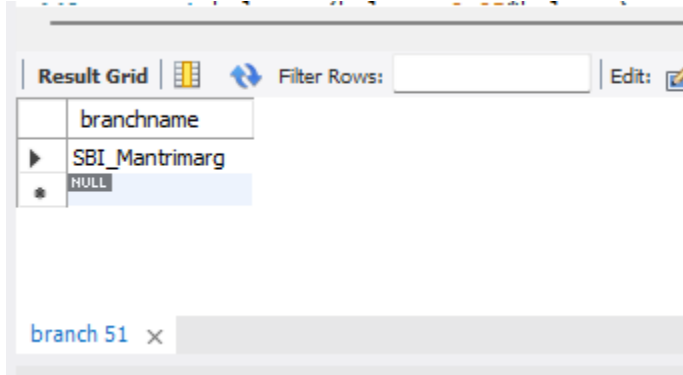
5. Demonstrate how you delete all account tuples at every branch located in a specific city (Ex. Bombay).

```
delete from bankacc
where branchname=any
(select branchname from
branch
where branchcity='bombay');
select * from bankacc;
```

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

```
select branchname  
from branch  
where assets> (select  
sum(assets) from  
branch
```

where branchcity='bangalore');



The screenshot shows a database application window with a 'Result Grid' tab. The grid has two columns: 'branchname' and an unnamed column. The first row contains 'SBI_Mantrimarg' and 'NULL'. Below the grid, there is a tab labeled 'branch 51' with a close button 'x'.

	branchname	
▶	SBI_Mantrimarg	
*	NULL	

branch 51 x

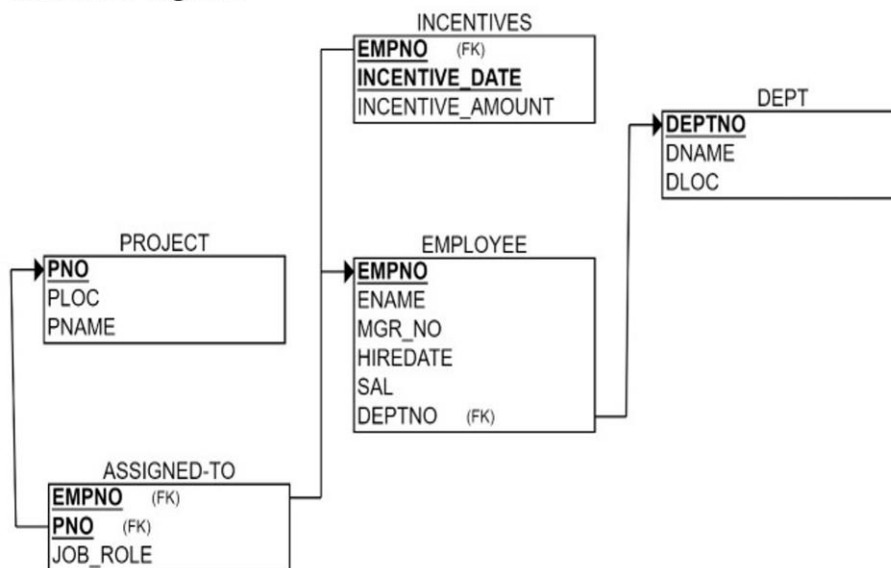
7.Demonstrate how you delete all account tuples at every branch located in a specific city(Ex. Bombay).

```
delete from bankacc where  
branchname=any (select  
branchname from branch  
where branchcity='bombay');  
select * from bankacc;
```

Week-5

Employee Database

Schema Diagram



Week-5: To Do List

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.

WEEK-5

EMPLOYEE

DATABASE

TO DO

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

```
create database employee;
```

```
create table dept(  
deptno int,  
dname varchar(20),  
dloc varchar(20),  
primary key(deptno)  
);
```

```
create table employee(  
empno int,  
ename varchar(20),  
mgr_no int, hiredate  
date,  
sal double,  
deptno int,  
primary key(empno),  
foreign key (deptno) references dept(deptno)  
on delete cascade  
on update cascade  
);
```

```
create table incentives(  
empno int,  
incentive_date date,  
incentive_amount float,  
primary key(empno,incentive_date),  
foreign key (empno) references employee(empno)  
on delete cascade  
on update cascade  
);  
create table project(
```

```

pno int,
ploc varchar(20),
pname varchar(20),
primary key(pno)
);

```

```

create table assigned_to(
empno int,
pno int,
job_role varchar(20),
primary key(empno,pno),
foreign key (empno) references employee(empno),
foreign key (pno) references project(pno)
on delete cascade
on update cascade
);

```

```

6 • desc dept;
7 • create table employee( empno int,
8   ename varchar(20), mgr_no int, hiredate date,
9   sal double, deptno int,

```

Result Grid

Field	Type	Null	Key	Default	Extra
deptno	int	NO	PRI	NULL	
dname	varchar(20)	YES		NULL	
dloc	varchar(20)	YES		NULL	

```

14 • desc employee;
15 • create table incentives( empno int, incentive

```

Result Grid

Field	Type	Null	Key	Default	Extra
empno	int	NO	PRI	NULL	
ename	varchar(20)	YES		NULL	
mgr_no	int	YES		NULL	
hiredate	date	YES		NULL	
sal	double	YES		NULL	
deptno	int	YES	MUL	NULL	

20 • desc incentives;

21 • create table project(pno int,
22 | ploc varchar(20), pname varchar(20), pr

25 • desc project;

32 • desc assigned_to;

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

Field	Type	Null	Key	Default	Extra
pno	int	NO	PRI	NULL	
ploc	varchar(20)	YES		NULL	
pname	varchar(20)	YES		NULL	

Field	Type	Null	Key	Default	Extra
empno	int	NO	PRI	NULL	
pno	int	NO	PRI	NULL	
job_role	varchar(20)	YES		NULL	

2. Enter greater than five tuples for each table.

insert into dept values(10,'cse','bangalore');
 insert into dept values(20,'ise','bangalore');
 insert into dept values(30,'aiml','hyderabad');
 insert into dept values(40,'ece','mysore');
 insert into dept values(50,'eee','delhi');
 insert into dept values(60,'iem','chennai');

insert into employee values(11,'Rajesh',21,'2000-04-03',80000,10);
 insert into employee values(12,'Ajay',11,'2003-04-06',70000,20); insert
 into employee values(13,'Divya',11,'2006-03-07',60000,30); insert into
 employee values(14,'Chandan',12,'2007-09-03',50000,40); insert into
 employee values(15,'Bhavesh',13,'2009-11-13',40000,50); insert into
 employee values(16,'Tarun',14,'2012-02-10',30000,60); insert into
 employee values(17,'Brinda',14,'2009-05-12',50000,10); insert into
 employee values(18,'Anil',15,'2015-01-01',30000,20); insert into
 employee values(19,'Puja',15,'2020-10-21',60000,30); insert into
 employee values(20,'Ram',16,'2021-09-17',45000,40);

```
insert into incentives values(11,'2002-09-08',40000);
insert into incentives values(12,'2005-07-10',33000);
insert into incentives values(13,'2008-01-21',7000);
insert into incentives values(14,'2014-08-05',8000);
insert into incentives values(15,'2017-09-13',5000);
insert into incentives values(17,'2021-03-17',6000);
insert into incentives values(18,'2021-04-16',8000);
insert into incentives values(19,'2021-08-11',9000);
```

```

insert into project values(121,'bangalore','proj1');
insert into project values(122,'bangalore','proj2');
insert into project values(123,'mysore','proj3');
insert into project values(124,'hyderabad','proj4');
insert into project values(125,'delhi','proj5'); insert
into project values(126,'mumbai','proj6'); insert
into project values(127,'calicut','proj7'); insert into
project values(128,'calicut','proj8');

```

```

insert into assigned_to values(11,121,'manager');
insert into assigned_to values(12,122,'team_lead');
insert into assigned_to values(13,123,'analyst'); insert
into assigned_to values(14,124,'team_lead'); insert
into assigned_to values(15,125,'manager'); insert into
assigned_to values(16,126,'programmer'); insert into
assigned_to values(17,127,'team_lead'); insert into
assigned_to values(19,128,'team_lead');

```

```

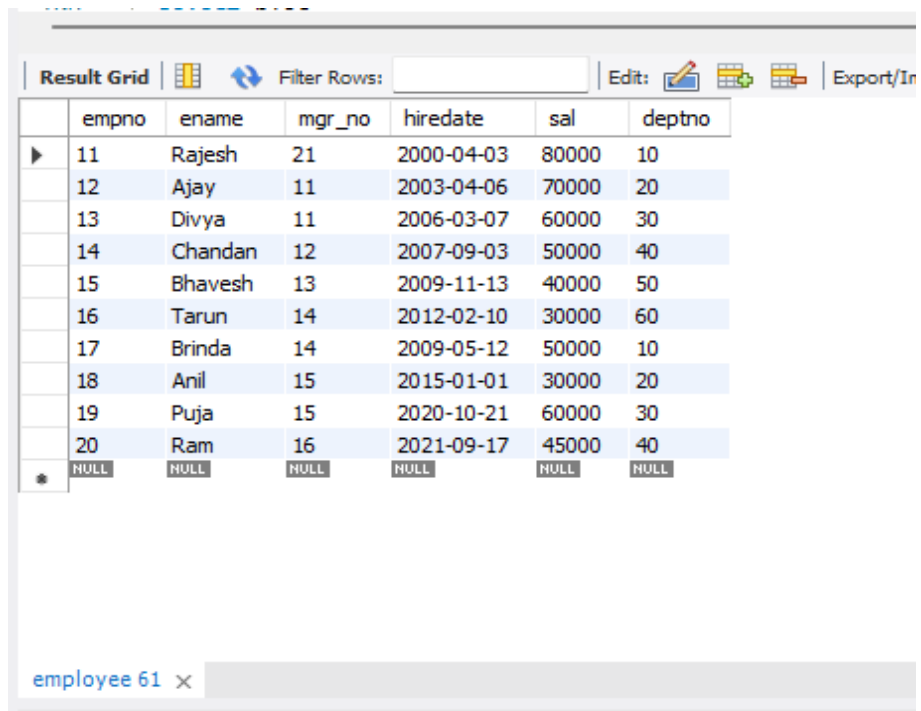
select *from dept;

```

Result Grid			
Filter Rows:			
	deptno	dname	dloc
▶	10	cse	bangalore
	20	ise	bangalore
	30	aiml	hyderabad
	40	ece	mysore
	50	eee	delhi
	60	iem	chennai
✱	NULL	NULL	NULL

dept 60 x

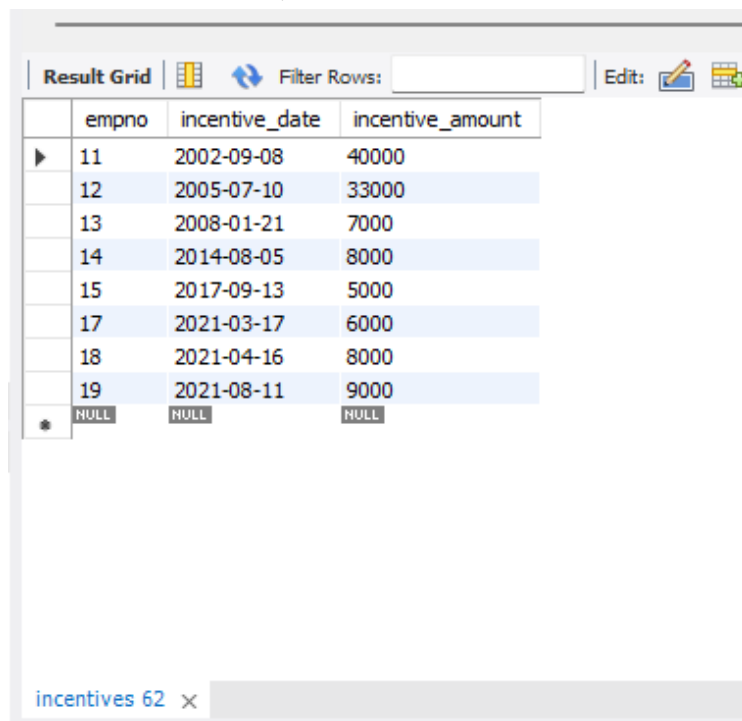
select *from employee;



	empno	ename	mgr_no	hiredate	sal	deptno
▶	11	Rajesh	21	2000-04-03	80000	10
	12	Ajay	11	2003-04-06	70000	20
	13	Divya	11	2006-03-07	60000	30
	14	Chandan	12	2007-09-03	50000	40
	15	Bhavesh	13	2009-11-13	40000	50
	16	Tarun	14	2012-02-10	30000	60
	17	Brinda	14	2009-05-12	50000	10
	18	Anil	15	2015-01-01	30000	20
	19	Puja	15	2020-10-21	60000	30
	20	Ram	16	2021-09-17	45000	40
*	NULL	NULL	NULL	NULL	NULL	NULL

employee 61 ×

select *from incentives;



	empno	incentive_date	incentive_amount
▶	11	2002-09-08	40000
	12	2005-07-10	33000
	13	2008-01-21	7000
	14	2014-08-05	8000
	15	2017-09-13	5000
	17	2021-03-17	6000
	18	2021-04-16	8000
	19	2021-08-11	9000
*	NULL	NULL	NULL

incentives 62 ×

select *from project;

	pno	ploc	pname
▶	121	bangalore	proj1
	122	bangalore	proj2
	123	mysore	proj3
	124	hyderabad	proj4
	125	delhi	proj5
	126	mumbai	proj6
	127	calicut	proj7
	128	calicut	proj8
*	NULL	NULL	NULL

project 63 ×

Output

select *from assigned_to;

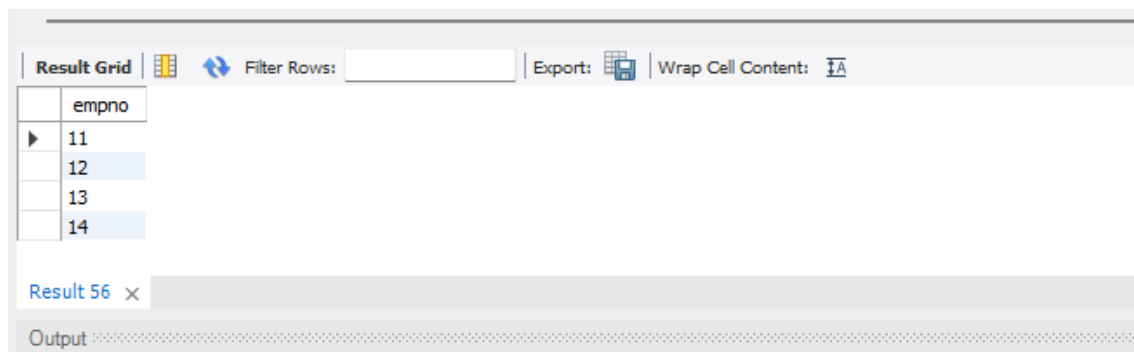
	empno	pno	job_role
▶	11	121	manager
	12	122	team_lead
	13	123	analyst
	14	124	team_lead
	15	125	manager
	16	126	programmer
	17	127	team_lead
	19	128	team_lead
*	NULL	NULL	NULL

assigned_to 64 ×

Output

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

```
select a.empno
from assigned_to a, project p
where p.pno=a.pno and p.ploc in(
select ploc
from project
where ploc='bangalore' or ploc='hyderabad' or ploc='mysore'
);
```

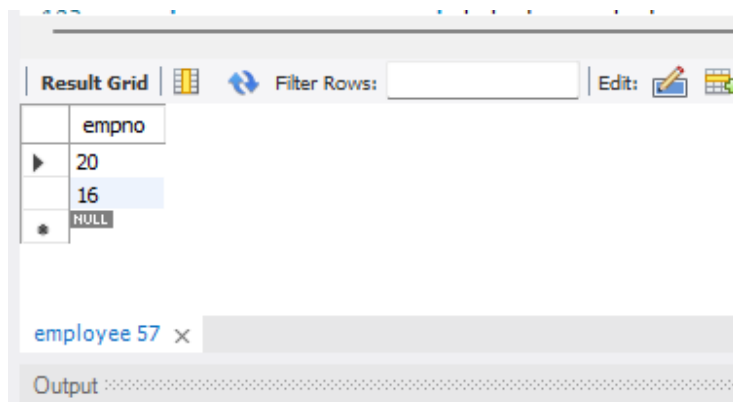


The screenshot shows a database query result grid. The grid has a header row with the column name 'empno'. Below the header, there are four rows with the values 11, 12, 13, and 14. The grid is titled 'Result 56'. There are icons for 'Filter Rows', 'Export', and 'Wrap Cell Content' at the top of the grid.

empno
11
12
13
14

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

```
select e.empno
from employee e
where e.empno not in(
select empno
from incentives
);
```

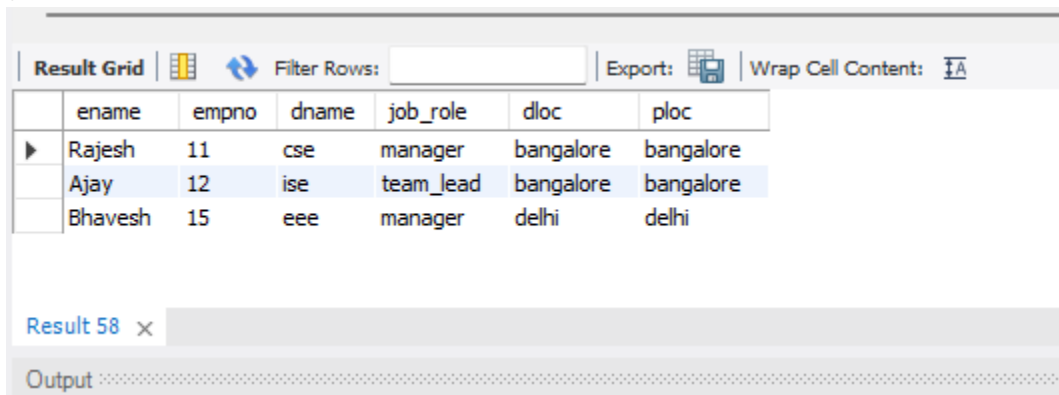


The screenshot shows a database query result grid. The grid has a header row with the column name 'empno'. Below the header, there are three rows with the values 20, 16, and NULL. The grid is titled 'employee 57'. There are icons for 'Filter Rows', 'Edit', and 'Wrap Cell Content' at the top of the grid.

empno
20
16
NULL

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.

```
select e.ename, d.dname, a.job_role
from employee e, dept d, assigned_to a
where e.deptno=d.deptno and a.empno=e.empno and e.empno in (
select empno
from incentives
where incentive_amount = (select max(incentive_AMOUNT) from incentives where
incentive_date between '2021-01-01'and '2021-12-31')
);
```



The screenshot shows a database interface with a 'Result Grid' tab. It contains a table with 7 columns: ename, empno, dname, job_role, dloc, and ploc. There are three rows of data. The first row is for Rajesh (empno 11, dept cse, manager, bangalore). The second row is for Ajay (empno 12, dept ise, team_lead, bangalore). The third row is for Bhavesh (empno 15, dept eee, manager, delhi). Below the table, there is a tab labeled 'Result 58' and an 'Output' section.

	ename	empno	dname	job_role	dloc	ploc
▶	Rajesh	11	cse	manager	bangalore	bangalore
	Ajay	12	ise	team_lead	bangalore	bangalore
	Bhavesh	15	eee	manager	delhi	delhi

Spot query

Find the employee name, dept name, job role of an employee who received maximum incentive in the year 2021.

```
select e.ename, d.dname, a.job_role
from employee e, dept d, assigned_to a
where e.deptno=d.deptno and a.empno=e.empno and e.empno in (
select empno
from incentives
where incentive_amount = (select max(incentive_AMOUNT) from incentives where
incentive_date between '2021-01-01'and '2021-12-31')
);
```

Result Grid				Filter Rows:		Export:		Wrap Cell Content:	
	ename	dname	job_role						
▶	Puja	aiml	team_lead						

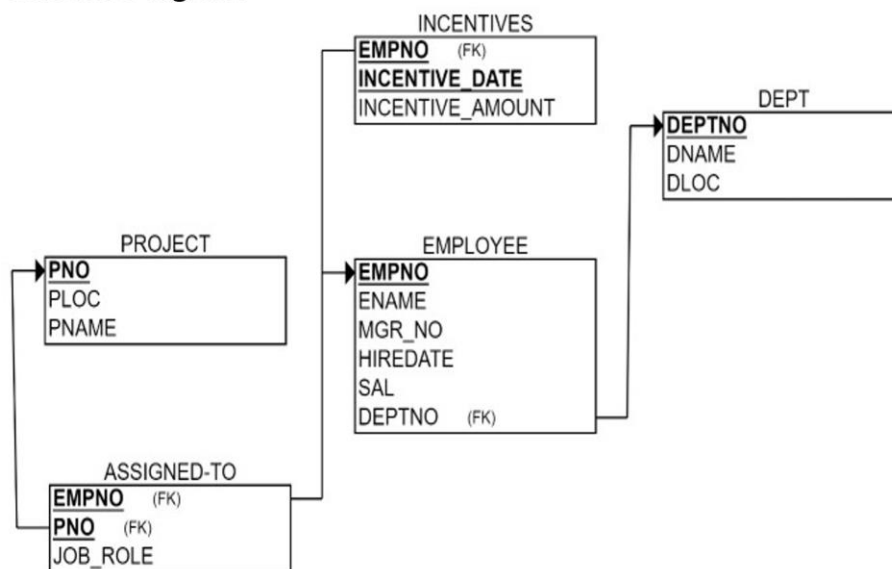
Result 59 ×

Output

Week-6

More Queries on Employee Database

Schema Diagram



Week-6: To Do List

1. Using Scheme diagram, Create tables by properly specifying the primary keys and the foreign keys.
2. Enter greater than five tuples for each table.
3. List the name of the managers with the maximum employees
4. Display those managers name whose salary is more than average salary of his employee.
5. Find the name of the second top level managers of each department.
6. Find the employee details who got second maximum incentive in January 2019.
7. Display those employees who are working in the same department where his manager is working.

```
create database employee2;
```

```
create table dept(  
deptno int,  
dname varchar(20),  
dloc varchar(20),  
primary key(deptno)  
);  
create table employee(  
empno int,  
ename varchar(20),  
mgr_no int, hiredate  
date,  
sal double,  
deptno int,  
primary key(empno),  
foreign key (deptno) references dept(deptno)  
on delete cascade  
on update cascade  
);  
create table incentives(  
empno int,  
incentive_date date,  
incentive_amount float,  
primary key(empno,incentive_date),  
foreign key (empno) references employee(empno)  
on delete cascade  
on update cascade  
);  
create table project(  
pno int,  
ploc varchar(20),  
pname varchar(20),
```

primary key(pno)

```
);
create table assigned_to(
empno int,
pno int,
job_role varchar(20),
primary key(empno,pno),
foreign key (empno) references employee(empno),
foreign key (pno) references project(pno)
on delete cascade
on update cascade
);
```

```
6 • desc dept;
```

```
7 • create table employee( empno int,
```

```
8   ename varchar(20), mgr_no int, hiredate date,
```

```
9   sal double, deptno int,
```

<

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

	Field	Type	Null	Key	Default	Extra
▶	deptno	int	NO	PRI	NULL	
	dname	varchar(20)	YES		NULL	
	dloc	varchar(20)	YES		NULL	

```
14 • desc employee;
```

```
15 • create table incentives( empno int, incentive,
```

<

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

	Field	Type	Null	Key	Default	Extra
▶	empno	int	NO	PRI	NULL	
	ename	varchar(20)	YES		NULL	
	mgr_no	int	YES		NULL	
	hiredate	date	YES		NULL	
	sal	double	YES		NULL	
	deptno	int	YES	MUL	NULL	

```
20 • desc incentives;
```

```
21 • create table project( pno int,
```

```
22   ploc varchar(20), pname varchar(20), pr
```

<

Result Grid | Filter Rows: | Export: |

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

25 • desc project;

Field	Type	Null	Key	Default	Extra
pno	int	NO	PRI	NULL	
ploc	varchar(20)	YES		NULL	
pname	varchar(20)	YES		NULL	

32 • desc assigned_to;

Field	Type	Null	Key	Default	Extra
empno	int	NO	PRI	NULL	
pno	int	NO	PRI	NULL	
job_role	varchar(20)	YES		NULL	

1. Enter greater than five tuples for each table.

```
insert into dept values(10,'cse','bangalore');
insert into dept values(20,'ise','bangalore');
insert into dept values(30,'aiml','hyderabad');
insert into dept values(40,'ece','mysore');
insert into dept values(50,'eee','delhi');
insert into dept values(60,'iem','chennai');
```

```
insert into employee values(11,'Rajesh',21,'2000-04-03',80000,10);
insert into employee values(12,'Ajay',11,'2003-04-06',70000,20); insert
into employee values(13,'Divya',11,'2006-03-07',60000,30); insert into
employee values(14,'Chandan',12,'2007-09-03',50000,40); insert into
employee values(15,'Bhavesh',13,'2009-11-13',40000,50); insert into
employee values(16,'Tarun',14,'2012-02-10',30000,60); insert into
employee values(17,'Brinda',11,'2009-05-12',50000,10); insert into
employee values(18,'Anil',15,'2015-01-01',30000,20); insert into
employee values(19,'Puja',15,'2020-10-21',60000,30); insert into
employee values(20,'Ram',16,'2021-09-17',45000,40); insert into
employee values(21,'Priya',22,'2002-03-13',85000,10);
```

```
insert into incentives values(11,'2012-09-08',40000);
insert into incentives values(12,'2015-07-10',33000);
insert into incentives values(13,'2019-01-21',7000);
insert into incentives values(14,'2019-01-05',8000);
insert into incentives values(15,'2019-01-13',5000);
insert into incentives values(17,'2021-03-17',6000);
insert into incentives values(18,'2021-04-16',8000);
insert into incentives values(19,'2021-08-11',9000);
```

```

insert into project values(121,'bangalore','proj1');
insert into project values(122,'bangalore','proj2');

insert into project values(123,'mysore','proj3');
insert into project values(124,'hyderabad','proj4');
insert into project values(125,'delhi','proj5'); insert
into project values(126,'mumbai','proj6'); insert
into project values(127,'calicut','proj7'); insert into
project values(128,'calicut','proj8');

```

```




insert into assigned_to values(11,121,'manager');
insert into assigned_to values(12,122,'team_lead');
insert into assigned_to values(13,123,'analyst'); insert
into assigned_to values(14,124,'team_lead'); insert
into assigned_to values(15,125,'manager'); insert into
assigned_to values(16,126,'programmer'); insert into
assigned_to values(17,127,'team_lead'); insert into
assigned_to values(19,128,'team_lead');

```




empno	pno	job_role
11	121	manager
12	122	team_lead
13	123	analyst
14	124	team_lead
15	125	manager
16	126	programmer
17	127	team_lead

deptno	dname	loc
10	cse	bangalore
20	ise	bangalore
30	aiml	hyderabad
40	ece	mysore
50	eee	delhi
60	iem	chennai
NULL	NULL	NULL

dept 1 x

Result Grid			
		Filter Rows:	
Edit:    Export/In			
empno	incentive_date	incentive_amount	
11	2012-09-08	40000	
12	2015-07-10	33000	
13	2019-01-21	7000	
14	2019-01-05	8000	
15	2019-01-13	5000	
17	2021-03-17	6000	
18	2021-04-16	8000	

incentives 1 x

Result Grid			
		Filter Rows:	
Edit:    Expo			
pno	ploc	pname	
121	bangalore	proj1	
122	bangalore	proj2	
123	mysore	proj3	
124	hyderabad	proj4	
125	delhi	proj5	
126	mumbai	proj6	
127	calicut	proj7	

project 1 x

Output

select * from employee

empno	ename	mgr_no	hiredate	sal	deptno
11	Rajesh	21	2000-04-03	80000	10
12	Ajay	11	2003-04-06	70000	20
13	Divya	11	2006-03-07	60000	30
14	Chandan	12	2007-09-03	50000	40
15	Bhavesh	13	2009-11-13	40000	50
16	Tarun	14	2012-02-10	30000	60
17	Brinda	11	2009-05-12	50000	10
18	Anil	15	2015-01-01	30000	20
19	Puja	15	2020-10-21	60000	30
20	Ram	16	2021-09-17	45000	40
21	Priya	22	2002-03-13	85000	10
23	Manas	11	2019-03-13	3000	10
24	Prem	11	2020-03-13	8000	10
*	NULL	NULL	NULL	NULL	NULL

3. List the name of the managers with the maximum employees

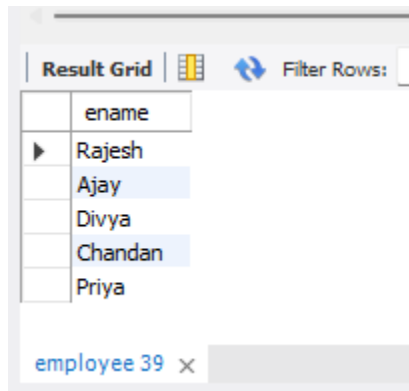
```
select emp.ename  
from employee emp  
where emp.empno=(
```

```
select mgr_no  
from employee e  
group by mgr_no  
having count(empno) >= all(  
select (count(empno))  
from employee  
group by mgr_no ));
```

ename
Rajesh

4. Display those managers name whose salary is more than average salary of his employee.

```
select emp.ename from
employee emp where
emp.sal > any ( select
avg(e.sal)
from employee e
where emp.empno=e.mgr_no
);
```



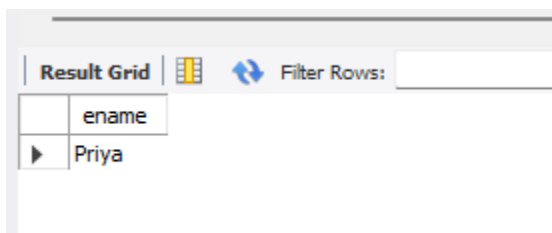
The screenshot shows a 'Result Grid' with a 'Filter Rows' button. The grid contains a single column labeled 'ename' with the following rows: Rajesh, Ajay, Divya, Chandan, and Priya. At the bottom, there is a tab labeled 'employee 39' with a close button 'x'.

ename
Rajesh
Ajay
Divya
Chandan
Priya

5. Find the name of the second top level managers of each department.

```
select emp.ename
from employee emp
```

```
where emp.ename = any(
select e2.ename
from employee e, employee e2
where e2.empno=e.mgr_no and e2.deptno = e.deptno and e.ename = any( select
e1.ename
from employee e1, employee e0
where e1.empno=e0.mgr_no and e1.deptno = e0.deptno
group by e1.mgr_no
having count(e1.empno)>1)
);
```

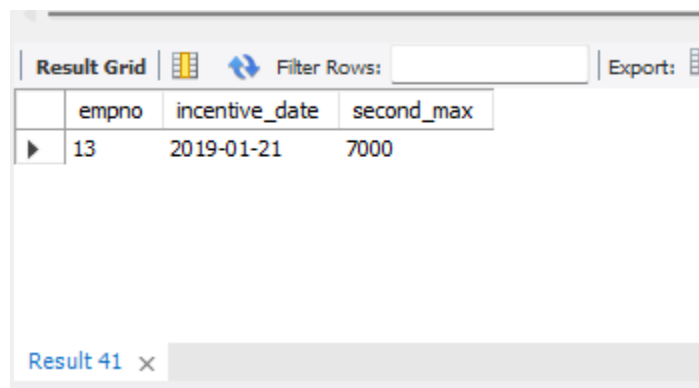


The screenshot shows a 'Result Grid' with a 'Filter Rows' button. The grid contains a single column labeled 'ename' with the following row: Priya.

ename
Priya

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

```
select i.empno, i.incentive_date, max(i.incentive_amount)second_max
from incentives i
where i.incentive_date between '2019-01-01' and '2019-01-31' and i.incentive_amount not in(
select max(incentive_amount)
from incentives
where incentive_date between '2019-01-01' and '2019-01-31');
```

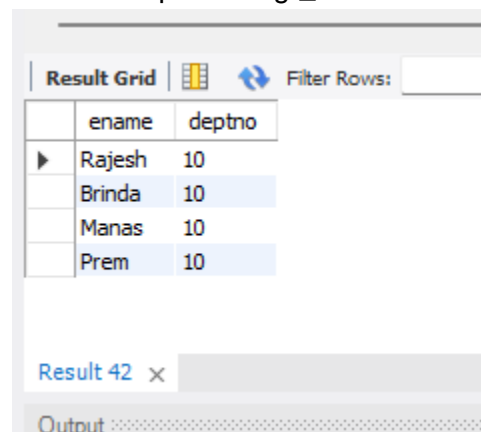


The screenshot shows a 'Result Grid' with a toolbar containing 'Filter Rows' and 'Export' buttons. The grid displays a single row of data. Below the grid, there is a tab labeled 'Result 41' with a close button 'x'.

	empno	incentive_date	second_max
▶	13	2019-01-21	7000

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

```
select e.ename, e.deptno
from employee e, employee e2
where e2.empno=e.mgr_no and e2.deptno = e.deptno;
```

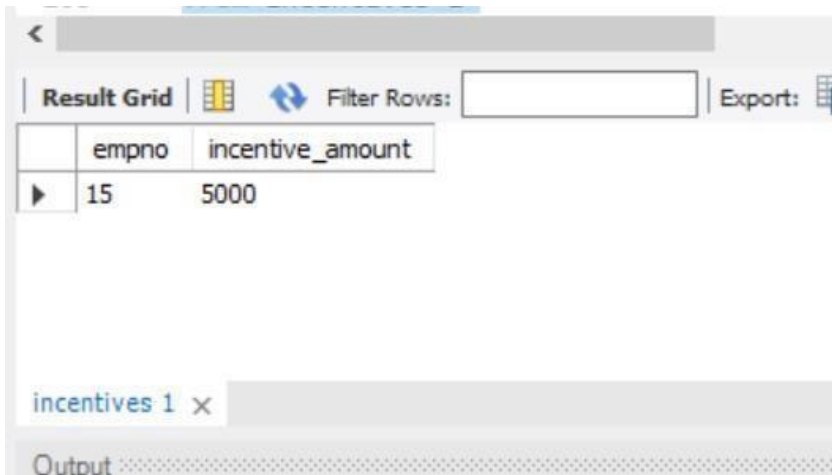


The screenshot shows a 'Result Grid' with a toolbar containing 'Filter Rows' and 'Export' buttons. The grid displays four rows of data. Below the grid, there is a tab labeled 'Result 42' with a close button 'x', and an 'Output' section with a dotted pattern.

	ename	deptno
▶	Rajesh	10
	Brinda	10
	Manas	10
	Prem	10

Spot query-Find the employee details who got third maximum incentive in January 2019.

```
select i.empno, i.incentive_amount
from incentives i
where 3 = (
    select count(*)
    from incentives j
    where incentive_date between '2019-01-01' and '2019-01-31' and i.incentive_amount
    <= j.incentive_amount)
and incentive_date between '2019-01-01' and '2019-01-31';
```



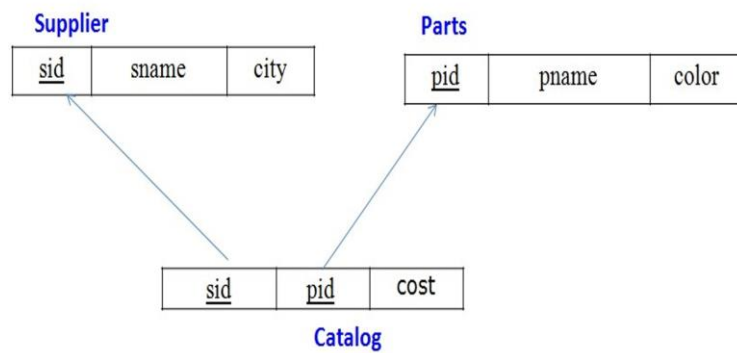
The screenshot shows a database query result grid. The grid has two columns: 'empno' and 'incentive_amount'. The first row contains the values '15' and '5000'. The grid is titled 'incentives 1' and has an 'Output' section below it.

empno	incentive_amount
15	5000

Week-7

Supplier Database

Schema Diagram



Week-7: To Do List

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.

```
create database Supplier;
create table supplier (
sid int,
sname varchar(20),
city varchar(20),
primary key(sid)
);
```

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

```
create table catalog
(
    sid int,
    pid int,
    cost int,
    primary key(pid,sid),
    foreign key (sid) references supplier(sid) on delete cascade on update cascade,
    foreign key (pid) references parts(pid) on delete cascade on update cascade
);
```

```
6 • desc supplier;
```

```
7
```

```
8 • create table parts (
```

```
9   pid int,
```

< Result Grid Filter Rows: Export: Wra

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

```
12 • desc parts;
```

```
13
```

< Result Grid Filter Rows: Export: Wra

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

```
19 • desc catalog;
```

< Result Grid Filter Rows: Export:

	Field	Type	Null	Key	Default	Extra
▶	sid	int	NO	PRI	NULL	
	pid	int	NO	PRI	NULL	
	cost	int	YES		NULL	

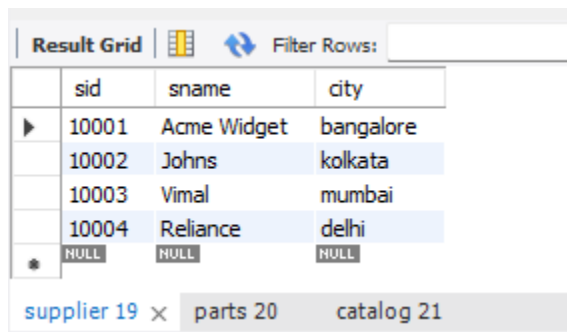
2.Insert appropriate records in each table.

```
insert into supplier values(10001,'Acme Widget','bangalore');
insert into supplier values(10002,'Johns','kolkata');
insert into supplier values(10003,'Vimal','mumbai');
insert into supplier values(10004,'Reliance','delhi');
```

```
insert into parts values(20001,'book','red');
insert into parts values(20002,'pen','red'); insert
into parts values(20003,'pencil','green'); insert
into parts values(20004,'mobile','green'); insert
into parts values(20005,'charger','black');
```

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

```
select * from supplier;
```



	sid	sname	city
▶	10001	Acme Widget	bangalore
	10002	Johns	kolkata
	10003	Vimal	mumbai
	10004	Reliance	delhi
*	NULL	NULL	NULL

supplier 19 x parts 20 catalog 21

```
select * from parts;
```

Result Grid			
	pid	pname	color
▶	20001	book	red
	20002	pen	red
	20003	pencil	green
	20004	mobile	green
	20005	charger	black
*	NULL	NULL	NULL

supplier 19 parts 20 × catalog 21

select * from catalog;

Result Grid			
	sid	pid	cost
▶	10001	20001	10
	10002	20001	10
	10001	20002	10
	10002	20002	20
	10001	20003	30
	10003	20003	30
	10004	20003	40
	10001	20004	10
	10001	20005	10
	NULL	NULL	NULL

supplier 19 parts 20 catalog 21 ×

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

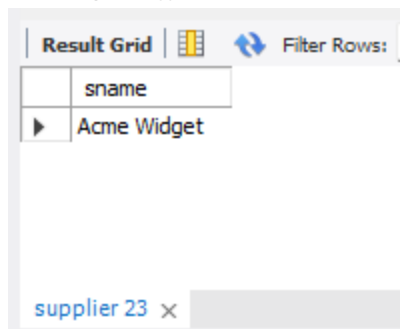
```
select pname
from parts p
where exists(
select *
from catalog c
where c.pid=p.pid
);
```

Result Grid	
	pname
▶	book
	pen
	pencil
	mobile
	charger

parts 22 ×

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

```
select s.sname
from supplier s
where s.sid in(
select c.sid from
catalog c group
by c.sid
having count(c.pid)=(select count(pid)
from parts));
```

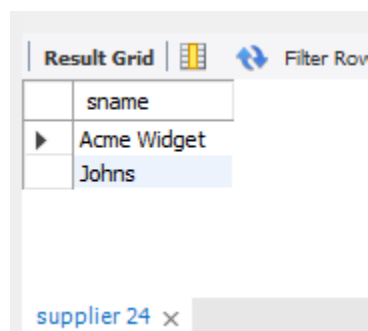


The screenshot shows a database interface with a 'Result Grid' tab. The grid has a single column labeled 'sname'. Below the header, there is one row with the value 'Acme Widget'. At the bottom of the window, there is a tab labeled 'supplier 23' with a close button (x).

sname
Acme Widget

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

```
select s.sname
from supplier s
where s.sid in(
select c.sid
from catalog c inner join parts p
on c.pid=p.pid
where p.color='red'
group by c.sid
having count(c.pid)=(select count(pid)
from parts
where color='red'));
```

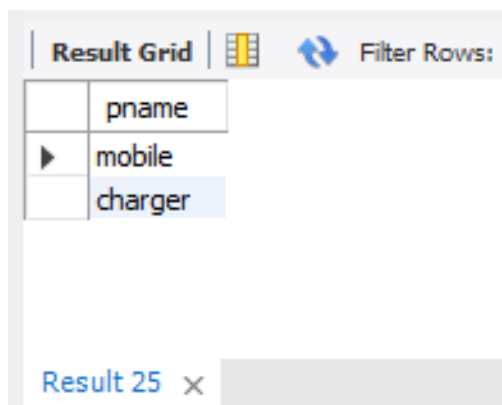


The screenshot shows a database interface with a 'Result Grid' tab. The grid has a single column labeled 'sname'. Below the header, there are two rows: 'Acme Widget' and 'Johns'. The 'Johns' row is highlighted with a blue background. At the bottom of the window, there is a tab labeled 'supplier 24' with a close button (x).

sname
Acme Widget
Johns

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

```
select p.pname
from catalog c1, parts p
where c1.sid=(select sid from supplier where sname='Acme Widget') and p.pid=c1.pid and
c1.pid
not in (select c.pid
        from catalog c
        where c.sid!=(select sid from supplier where sname='Acme Widget'));
```



	pname
▶	mobile
	charger

Result 25 x

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).

```
select c1.sid from
catalog c1 where
c1.cost> (select
avg(cost) from
```

```
catalog c2  
where c1.pid=c2.pid  
group by pid);
```

Result Grid		Filter Rows:
	sid	
▶	10002	
	10004	

catalog 26 ×

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

```
select c1.pid,s.sname from
supplier s, catalog c1
where s.sid=c1.sid and c1.cost in(select max(cost) from catalog c2 where c2.pid=c1.pid group
by pid
);
```

Result Grid		Filter Rows
	pid	sname
▶	20001	Acme Widget
	20004	Acme Widget
	20005	Acme Widget
	20001	Johns
	20002	Johns
	20003	Reliance

Result 27 ×

Week-8

Airline Flight Database

FLIGHTS

<u>flno</u>	from	to	distance	departs	arrives	price
-------------	------	----	----------	---------	---------	-------

AIRCRAFT

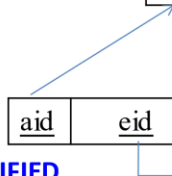
<u>aid</u>	aname	cruisingrange
------------	-------	---------------

EMPLOYEE

<u>eid</u>	ename	salay
------------	-------	-------

CERTIFIED

<u>aid</u>	<u>eid</u>
------------	------------



To Do

- i. Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80,000.
- ii. For each pilot who is certified for more than three aircrafts, find the aid and the maximum cruising range of the aircraft for which she or he is certified.
- iii. Find the names of pilots whose salary is less than the price of the cheapest route from Bengaluru to Frankfurt.
- iv. For all aircraft with cruising range over 1000 Kms, find the name of the aircraft and the Average salary of all pilots certified for this aircraft.
- v. Find the names of pilots certified for some Boeing aircraft.
- vi. Find the aids of all aircraft that can be used on routes from Bengaluru to New Delhi.

```
create database airline_flight;
```

```
create table
flights( flno int,
from_place
varchar(20), to_place
varchar(20), distance
int,
departs
time,
arrives
time, price
int,
primary key(flno)
);
```

```
create table
aircraft( aid int,
aname
varchar(20),
```

```
cruising_range int,  
primary key(aid)  
);
```

```
create table  
employee( eid int,  
ename  
varchar(20),  
salary int,  
primary key(eid)  
);
```



```
create table  
certified( eid int,  
aid int,  
foreign key (eid) references  
employee(eid), foreign key (aid)  
references aircraft(aid)  
on delete cascade  
on update cascade );
```

```
8 • desc flights;  
9
```

Result Grid							Filter Rows:	Export:	Wrap Cell Content:
Field	Type	Null	Key	Default	Extra				
flno	int	NO	PRI	NULL					
from_place	varchar(20)	YES		NULL					
to_place	varchar(20)	YES		NULL					
distance	int	YES		NULL					
departs	time	YES		NULL					
arrives	time	YES		NULL					
price	int	YES		NULL					

```
13 • desc aircraft;  
14  
15 • create table employee( eid int,
```

<

Result Grid  Filter Rows: Export:  Wrap C

	Field	Type	Null	Key	Default	Extra
▶	aid	int	NO	PRI	NULL	
	aname	varchar(20)	YES		NULL	
	cruising_range	int	YES		NULL	

19 • desc employee;

Field	Type	Null	Key	Default	Extra
eid	int	NO	PRI	NULL	
ename	varchar(20)	YES		NULL	
salary	int	YES		NULL	

26 • desc certified;

Field	Type	Null	Key	Default	Extra
eid	int	YES	MUL	NULL	
aid	int	YES	MUL	NULL	

```
insert into employee values (101,'Avinash',50000);
insert into employee values (102,'Lokesh',60000);
insert into employee values (103,'Rakesh',70000);
insert into employee values (104,'Santhosh',82000);
insert into employee values (105,'Tilak',5000);
```

```
insert into aircraft values (1,'Airbus',2000);
insert into aircraft values (2,'Boeing',700);
insert into aircraft values (3,'Jetairways',550);
insert into aircraft values (4,'Indigo',5000);
insert into aircraft values (5,'Boeing',4500);
insert into aircraft values (6,'Airbus',2200);
```

```
insert into certified values(101,2);
insert into certified values(101,4);
insert into certified values(101,5);
insert into certified values(101,6);
insert into certified values(102,1);
insert into certified values(102,3);
```


insert into certified values(102,5);

insert into certified values(103,2);

insert into certified values(103,3);

insert into certified values(103,5);

insert into certified values(103,6);

insert into certified values(104,6);

insert into certified values(104,1);

insert into certified values(104,3);

insert into certified values(105,3);

insert into flights values(1,'Bengaluru','New Delhi',500,'6:00','9:00',5000);

insert into flights values(2,'Bengaluru','Chennai',300,'7:00','8:30',3000);

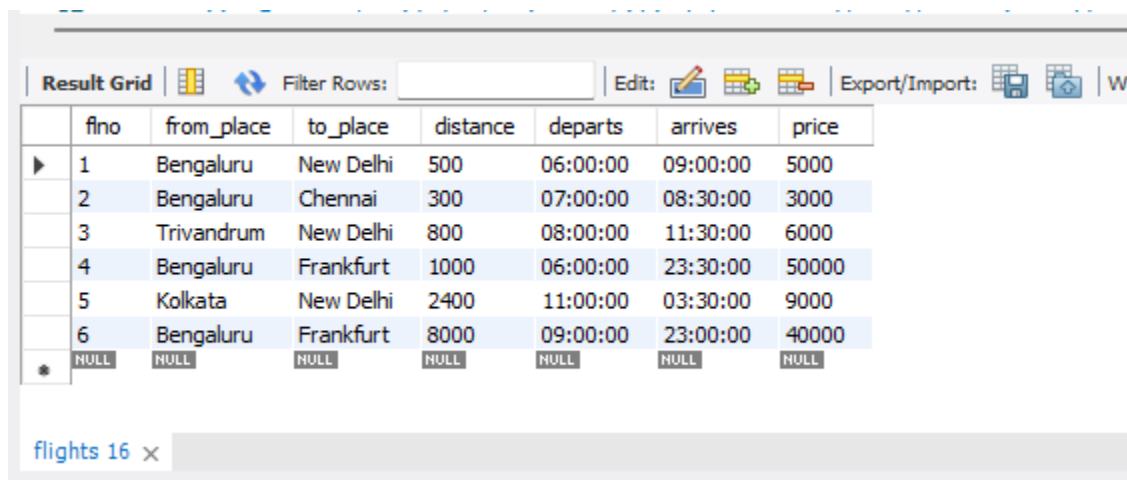
insert into flights values(3,'Trivandrum','New Delhi',800,'8:00','11:30',6000);

insert into flights values(4,'Bengaluru','Frankfurt',1000,'6:00','23:30',50000);

insert into flights values(5,'Kolkata','New Delhi',2400,'11:00','3:30',9000);

insert into flights values(6,'Bengaluru','Frankfurt',8000,'9:00','23:00',40000);

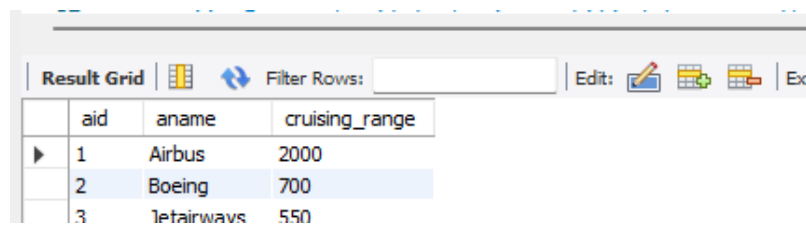
select * from flights;



	fno	from_place	to_place	distance	departs	arrives	price
▶	1	Bengaluru	New Delhi	500	06:00:00	09:00:00	5000
	2	Bengaluru	Chennai	300	07:00:00	08:30:00	3000
	3	Trivandrum	New Delhi	800	08:00:00	11:30:00	6000
	4	Bengaluru	Frankfurt	1000	06:00:00	23:30:00	50000
	5	Kolkata	New Delhi	2400	11:00:00	03:30:00	9000
	6	Bengaluru	Frankfurt	8000	09:00:00	23:00:00	40000
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

flights 16 x

select * from aircraft;



	aid	aname	cruising_range
▶	1	Airbus	2000
	2	Boeing	700
	3	Jetairways	550

select * from employee

Result Grid			
Filter Rows: <input type="text"/>			
Edit:			
	eid	ename	salary
▶	101	Avinash	50000
	102	Lokesh	60000
	103	Rakesh	70000
	104	Santhosh	82000
	105	Tilak	5000
*	NULL	NULL	NULL

employee 18 ×

select * from certified;

Result Grid		
Filter Rows: <input type="text"/>		
	eid	aid
▶	101	2
	101	4
	101	5
	101	6
	102	1
	102	3
	102	5
	103	2
	103	3
	103	5
	103	6
	104	6
	104	1
	104	3
	105	3

certified 19 ×

TO DO

1. Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80,000.

select a.aname

from aircraft a, certified c, employee e

where c.aid = a.aid and e.eid = c.eid and e.salary>80000;

Result Grid	
Filter Rows: <input type="text"/>	
	aname
▶	Airbus

2. For each pilot who is certified for more than three aircrafts, find the eid and the maximum cruising range of the aircraft for which she or he is certified.

```
select
e.eid,max(a.cruising_range) from
employee e, certified c, aircraft a
where c.aid=a.aid and e.eid
=c.eid group by c.eid
having count(distinct(c.aid))>=3;
```

Result Grid		Filter Rows:
	eid	max(a.cruising_range)
▶	101	5000
	102	4500
	103	4500
	104	2200

Result 21 ×

Output

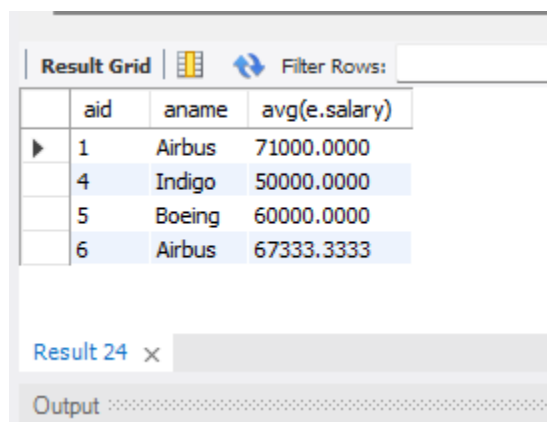
3. Find the names of pilots whose salary is less than the price of the cheapest route from Bengaluru to Frankfurt.

```
select ename
from employee
where salary <
( select
min(price) from
flights
where from_place='Bengaluru' and to_place='Frankfurt');
```

Result Grid		Filter Rows:
	ename	
▶	Tilak	

4. For all aircraft with cruising range over 1000 Kms, find the name of the aircraft and the Average salary of all pilots certified for this aircraft.

```
select a.aid, a.aname, avg(e.salary)
from aircraft a, employee e, certified
c
where c.aid = a.aid and e.eid = c.eid and a.cruising_range
>1000 group by c.aid;
```



The screenshot shows a 'Result Grid' with a table containing 4 columns: 'aid', 'aname', and 'avg(e.salary)'. There are 4 rows of data. Below the table, it says 'Result 24' and 'Output'.

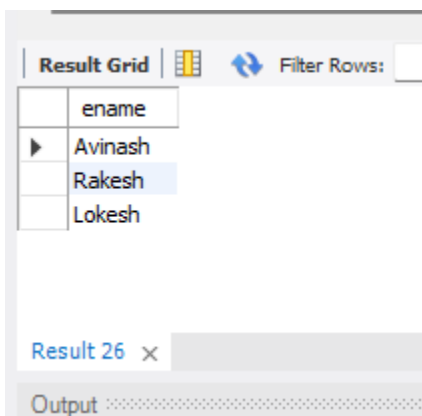
	aid	aname	avg(e.salary)
▶	1	Airbus	71000.0000
	4	Indigo	50000.0000
	5	Boeing	60000.0000
	6	Airbus	67333.3333

Result 24 ×

Output

5. Find the names of pilots certified for some Boeing aircraft.

```
select distinct(e.ename)
from aircraft a, employee e, certified c
where c.aid = a.aid and e.eid = c.eid and aname = some
( select aname from aircraft where aname = 'Boeing');
```



The screenshot shows a 'Result Grid' with a table containing 1 column: 'ename'. There are 3 rows of data. Below the table, it says 'Result 26' and 'Output'.

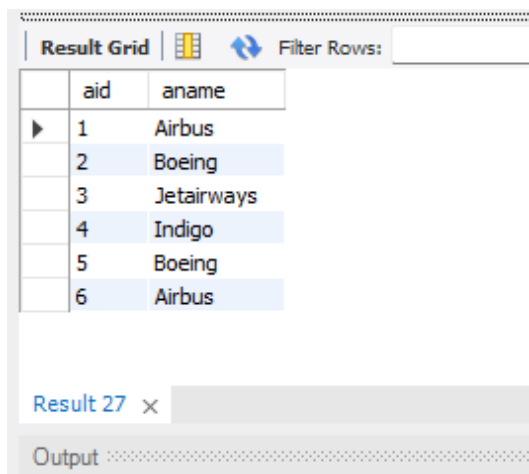
	ename
▶	Avinash
	Rakesh
	Lokesh

Result 26 ×

Output

6. Find the aids of all aircraft that can be used on routes from Bengaluru to New Delhi.

```
select aid, aname
from flights,
aircraft
where from_place='Bengaluru' and to_place='New Delhi' and cruising_range > (
select f.distance
from flights f
where f.from_place='Bengaluru' and f.to_place='New Delhi');
```



The screenshot shows a database interface with a 'Result Grid' tab. The grid contains 6 rows of data. The first row is highlighted with a mouse cursor. Below the grid, there is a 'Result 27' label and an 'Output' section.

	aid	aname
▶	1	Airbus
	2	Boeing
	3	Jetairways
	4	Indigo
	5	Boeing
	6	Airbus

Result 27 ×

Output

NoSQL Lab 1

Perform the following DB operations using MongoDB.

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

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

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

```
Atlas atlas-mdgaz1-shard-0 [primary] myDB> db.createCollection("Student");  
{ ok: 1 }
```

2. Insert appropriate values(at least 5)

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

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

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

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

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

```
Atlas atlas-mdgaz1-shard-0 [primary] myDB> db.Student.find()
[
  {
    _id: ObjectId("63bfcf9a56eba0e23c3a5c72"),
    RollNo: 1,
    Age: 21,
    Cont: 9876,
    email: 'antara.de9@gmail.com'
  },
  {
    _id: ObjectId("63bfcfb456eba0e23c3a5c73"),
    RollNo: 2,
    Age: 22,
    Cont: 9976,
    email: 'anushka.de9@gmail.com'
  },
  {
    _id: ObjectId("63bfcfd156eba0e23c3a5c74"),
    RollNo: 3,
    Age: 21,
    Cont: 5576,
    email: 'anubhav.de9@gmail.com'
  },
  {
    _id: ObjectId("63bfcfe456eba0e23c3a5c75"),
    RollNo: 4,
    Age: 20,
    Cont: 4476,
    email: 'pani.de9@gmail.com'
  },
  {
    _id: ObjectId("63bfcff656eba0e23c3a5c76"),
    RollNo: 5,
    Age: 23,
    Cont: 2276,
    email: 'rekha.de9@gmail.com'
  }
]
```

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

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

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

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

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

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

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

5. Export the created table into local file system

```
Mongoexport mongodb+srv://antararc:****@cluster0.mfnfeys.mongodb.net/myDB
```

```
--collection=Student-out C:\Users\BMSCECSE\Downloads\output.json
```

```
C:\Users\BMSCECSE>mongoexport mongodb+srv://antararc:Test1234@cluster0.mfnfeys.mongodb.net/myDB --collection=Student --out C:\Users\BMSCECSE\Downloads\output.json
2023-01-12T15:15:56.383+0530 connected to: mongodb+srv://[**REDACTED**]@cluster0.mfnfeys.mongodb.net/myDB
2023-01-12T15:15:56.497+0530 exported 7 records
```

6. Drop the table

```
db.Student.drop();
```

```
Atlas atlas-mdgaz1-shard-0 [primary] myDB> db.Student.drop();
true
Atlas atlas-mdgaz1-shard-0 [primary] myDB> db.Student.find()
```


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

```
Mongoimport mongodb+srv://antararc:Test1234@cluster0.mfnfeys.mongodb.net/myDB  
--collection=New_Student --type json --file C:\Users\BMSCECSE\Downloads\output.json
```

```
C:\Users\BMSCECSE>mongoimport mongodb+srv://antararc:Test1234@cluster0.mfnfeys.mongodb.net/myDB --collection=New_Student --type json --file C:\Users\BMSCECSE\Downloads\output.json  
2023-01-12T15:17:35.523+0530 connected to: mongodb+srv://[**REDACTED**]@cluster0.mfnfeys.mongodb.net/myDB  
2023-01-12T15:17:35.640+0530 7 document(s) imported successfully. 0 document(s) failed to import.
```