VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.

LAB REPORT
on

Database Management Systems (22CS3PCDBM)

Submitted by

G Sanjana Hebbar (1BM21CS062)

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

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "Database Management Systems (22CS3PCDBM)" carried out by **G Sanjana Hebbar (1BM21CS062),** 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.

Dr.Nandhini Vineeth	**Dr. Jyothi S Nayak**
Assistant Professor	Professor and Head
Department of CSE	Department of CSE
BMSCE, Bengaluru	BMSCE, Bengaluru

`

Index

Insurance Database

Question

Consider the Insurance database given below. The primary keys are underlined and the data types are specified.

PERSON (driver-id #: String, name: String, address: String)

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

ACCIDENT (report-number: int, date: date, location: String)

OWNS (driver-id #: String, Regno: String)

PARTICIPATED (driver-id: String, Regno: String, report-number: int, damage-

amount: int)

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

ii. Enter at least five tuples for each relation.

iii. Display Accident date and location.

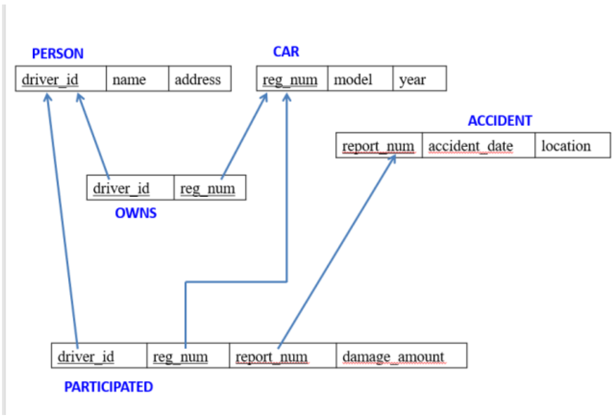
iv. Display driver id who did accident with damage amount greater than or equal to Rs.25000.

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

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

vii. Add a new accident to the database.

Schema Diagram



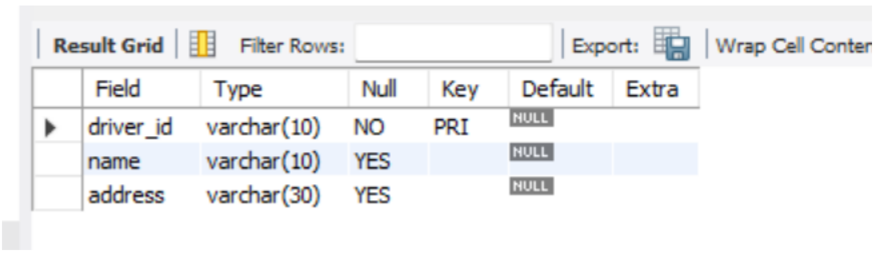
Create database

```
create database 1bm21cs062_insurance;
use 1bm21cs062_insurance;
```

Creating table

person:

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



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

car:

create table car (

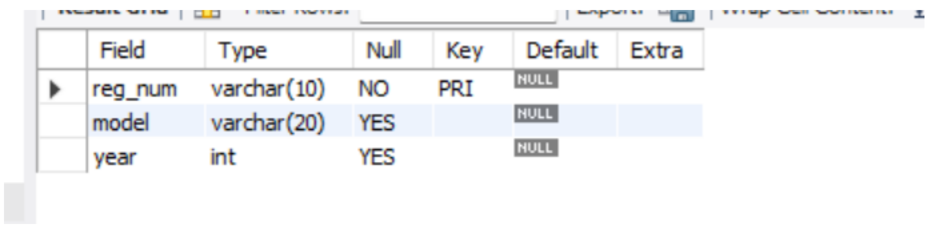
reg_num varchar (10),

model varchar (20),

year int,

primary key(reg_num)

);



	Field	Type	Null	Key	Default	Extra
▶	reg_num	varchar(10)	NO	PRI	NULL	
	model	varchar(20)	YES		NULL	
	year	int	YES		NULL	

accident:

create table accident (

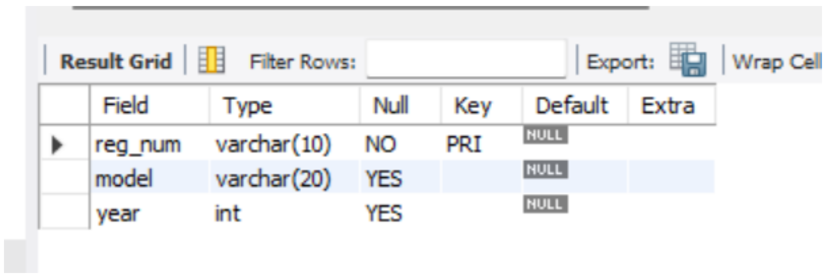
report_num int,

accident_date date,

location varchar (20),

primary key(report_num)

);



	Field	Type	Null	Key	Default	Extra
▶	reg_num	varchar(10)	NO	PRI	NULL	
	model	varchar(20)	YES		NULL	
	year	int	YES		NULL	

owns:

create table owns (

driver_id varchar (10),

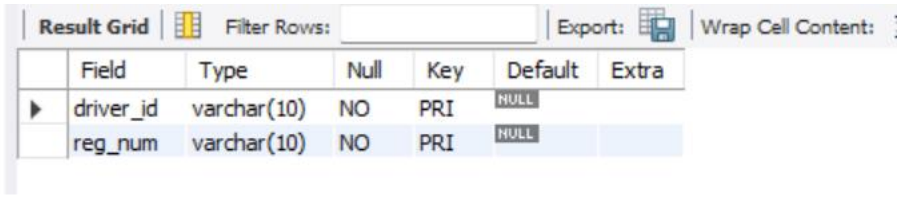
reg_num varchar (10),

primary key(driver_id,reg_num),

foreign key(driver_id) references person(driver_id),

foreign key(reg_num) references car(reg_num)

);



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

participated:

create table participated (

driver_id varchar (10),

reg_num varchar (10),

report_num int,

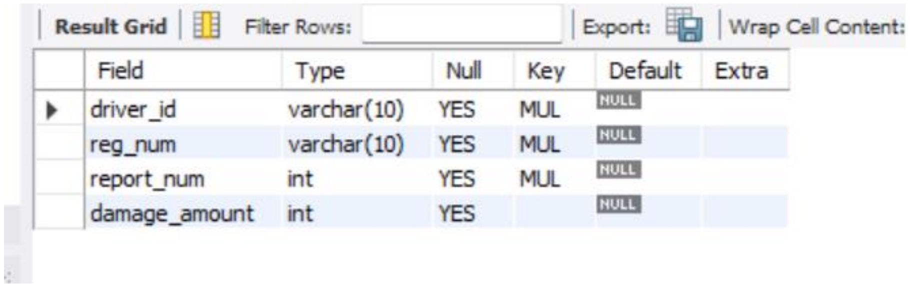
damage_amount int,

foreign key(driver_id) references person(driver_id),

foreign key(reg_num) references car(reg_num),

foreign key(report_num) references accident(report_num)

);



Field	Type	Null	Key	Default	Extra
driver_id	varchar(10)	YES	MUL	NULL	
reg_num	varchar(10)	YES	MUL	NULL	
report_num	int	YES	MUL	NULL	
damage_amount	int	YES		NULL	

Inserting Values to the table

person:

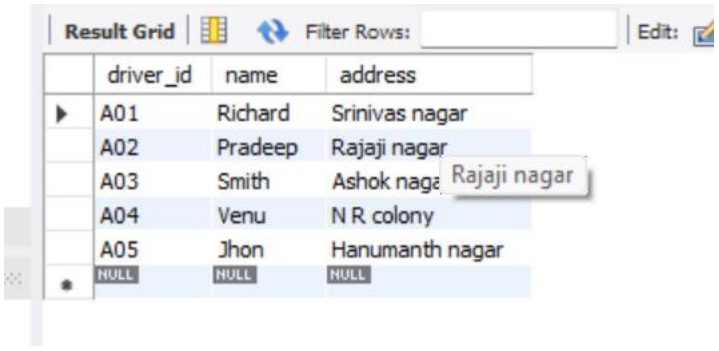
insert into person values ('A01','Richard','Srinivas nagar');

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');



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

car:

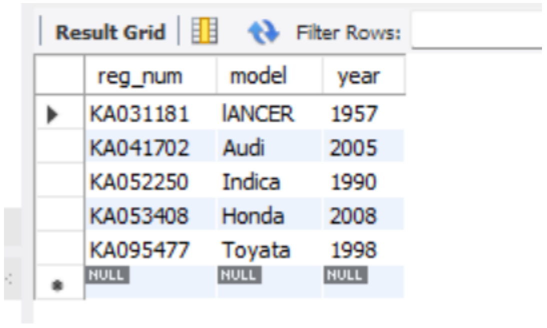
insert into car values('KA052250','Indica',1990);

insert into car values('KA031181','lANCER',1957);

insert into car values('KA095477','Toyata',1998);

insert into car values('KA053408','Honda',2008);

insert into car values('KA041702','Audi',2005);



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

accident:

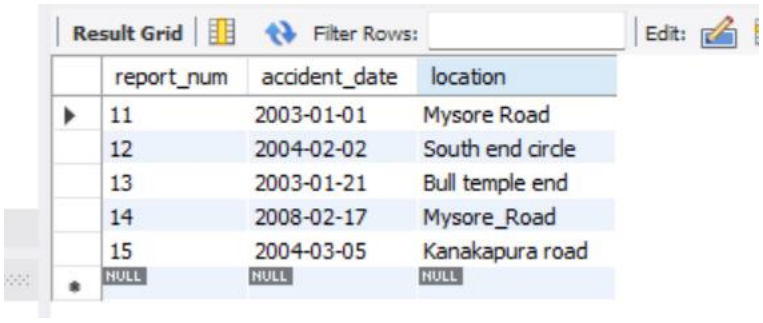
insert into accident values (11,'2003-01-01','Mysore Road');

insert into accident values (12,'2004-02-02','South end circle');

insert into accident values (13,'2003-01-21','Bull temple end');

insert into accident values (14,'2008-02-17','Mysore_Road');

insert into accident values (15,'2004-03-05','Kanakapura road');



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

owns:

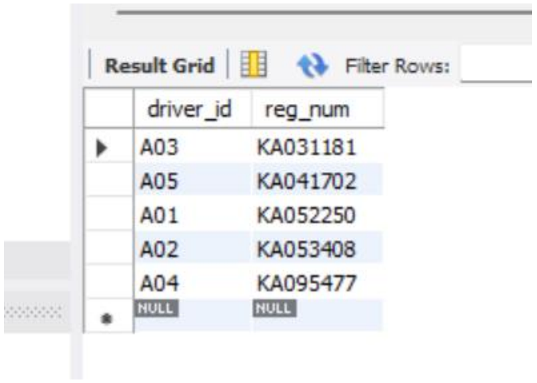
insert into owns values('A01','KA052250');

insert into owns values('A02','KA053408');

insert into owns values('A03','KA031181');

insert into owns values('A04','KA095477');

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



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

participated:

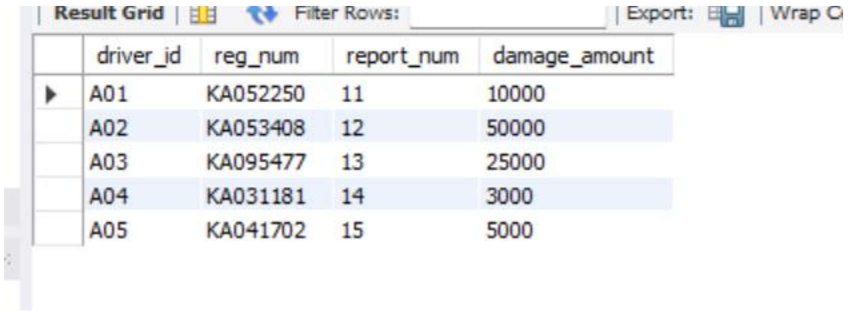
insert into participated values('A01','KA052250',11,10000);

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);

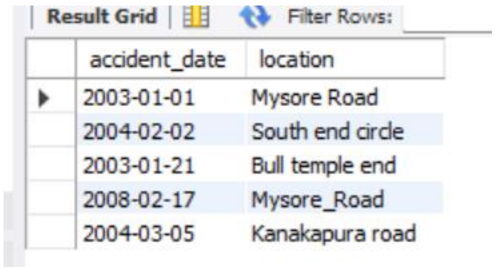


driver_id	reg_num	report_num	damage_amount
A01	KA052250	11	10000
A02	KA053408	12	50000
A03	KA095477	13	25000
A04	KA031181	14	3000
A05	KA041702	15	5000

Queries

1.Display Accident date and location.

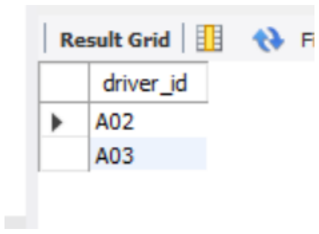
SQL> select accident_date, location from accident;



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

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

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



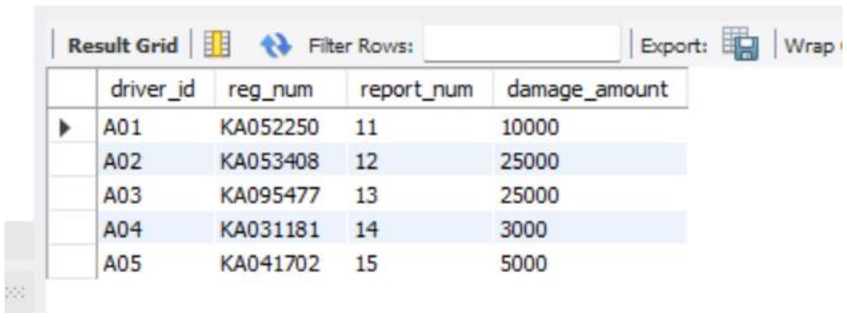
	driver_id
▶	A02
	A03

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

SQL>update participated set damage_amount=25000

where reg_num='KA053408' and report_num=12;

SQL>select * from participated;



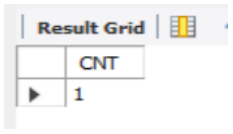
	driver_id	reg_num	report_num	damage_amount
▶	A01	KA052250	11	10000
	A02	KA053408	12	25000
	A03	KA095477	13	25000
	A04	KA031181	14	3000
	A05	KA041702	15	5000

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

SQL> select count (distinct driver_id)

CNT from participated a, accident b

where a.report_num=b.report_num and b.accident_date like '%08%';

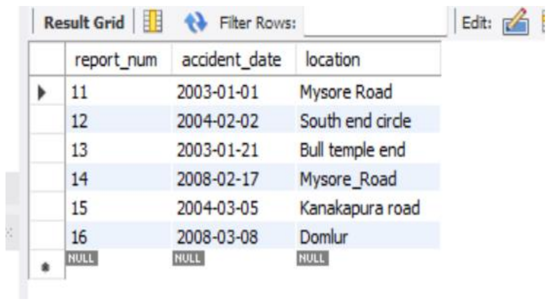


	CNT
▶	1

5.Add a new accident to the database.

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

SQL>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 end
14	2008-02-17	Mysore_Road
15	2004-03-05	Kanakapura road
16	2008-03-08	Domlur
NULL	NULL	NULL

More Queries on Insurance Database

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)

i. 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 relationEnter at least five tuples for each relation

ii. Enter at least five tuples for each relation

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

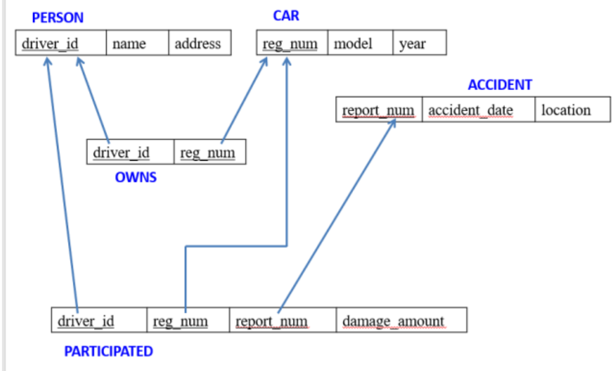
iv. Find the average damage amount

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

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

vii. Find maximum damage amount.

Schema Diagram



Create database

create database 1bm21cs062_insurance;

use 1bm21cs062_insurance;

Creating table

person:

create table person (

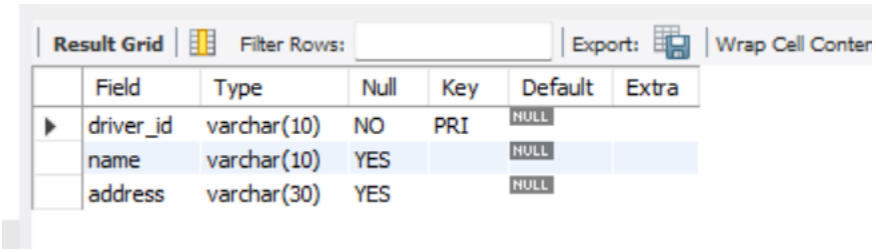
driver_id varchar (10),

name varchar (10),

address varchar (30),

primary key(driver_id)

);



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

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

car:

create table car (

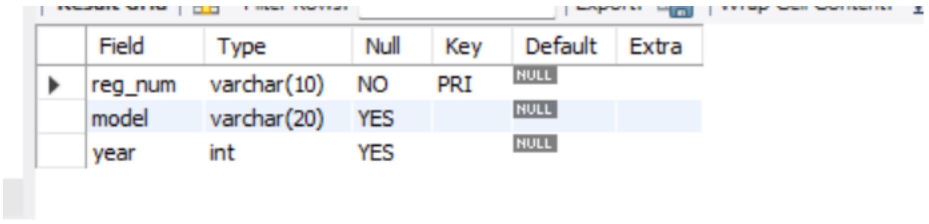
reg_num varchar (10),

model varchar (20),

year int,

primary key(reg_num)

);



	Field	Type	Null	Key	Default	Extra
▶	reg_num	varchar(10)	NO	PRI	NULL	
	model	varchar(20)	YES		NULL	
	year	int	YES		NULL	

accident:

create table accident (

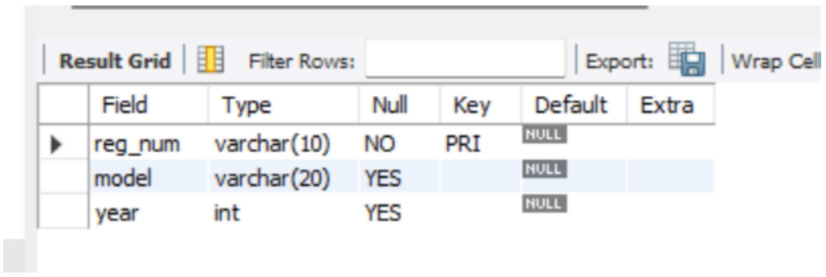
report_num int,

accident_date date,

location varchar (20),

primary key(report_num)

);



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

	Field	Type	Null	Key	Default	Extra
▶	reg_num	varchar(10)	NO	PRI	NULL	
	model	varchar(20)	YES		NULL	
	year	int	YES		NULL	

owns:

create table owns (

driver_id varchar (10),

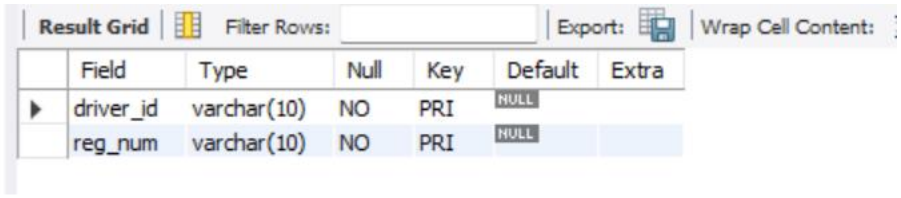
reg_num varchar (10),

primary key(driver_id,reg_num),

foreign key(driver_id) references person(driver_id),

foreign key(reg_num) references car(reg_num)

);



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

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

participated:

create table participated (

driver_id varchar (10),

reg_num varchar (10),

report_num int,

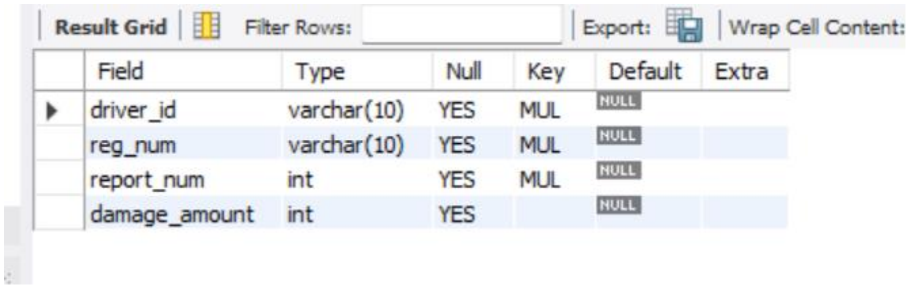
damage_amount int,

foreign key(driver_id) references person(driver_id),

foreign key(reg_num) references car(reg_num),

foreign key(report_num) references accident(report_num)

);



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

	Field	Type	Null	Key	Default	Extra
▶	driver_id	varchar(10)	YES	MUL	NULL	
	reg_num	varchar(10)	YES	MUL	NULL	
	report_num	int	YES	MUL	NULL	
	damage_amount	int	YES		NULL	

Inserting Values to the table

person:

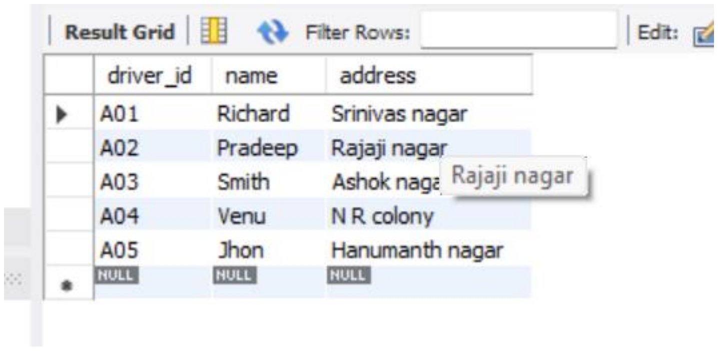
insert into person values ('A01','Richard','Srinivas nagar');

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');



Result Grid | Filter Rows: | Edit:

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

car:

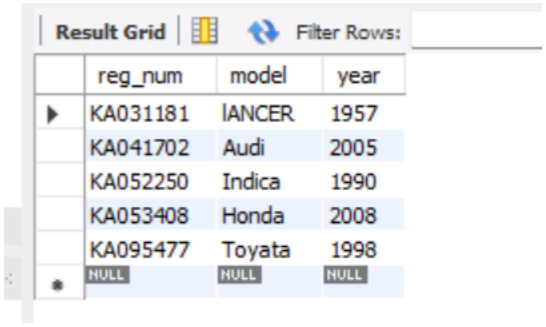
insert into car values('KA052250','Indica',1990);

insert into car values('KA031181','lANCER',1957);

insert into car values('KA095477','Toyata',1998);

insert into car values('KA053408','Honda',2008);

insert into car values('KA041702','Audi',2005);



Result Grid | Filter Rows:

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

accident:

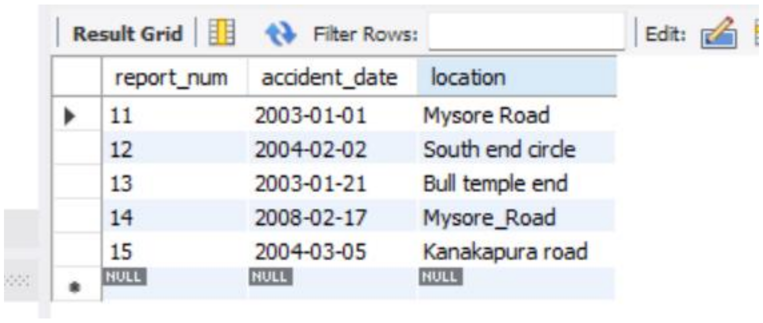
insert into accident values (11,'2003-01-01','Mysore Road');

insert into accident values (12,'2004-02-02','South end circle');

insert into accident values (13,'2003-01-21','Bull temple end');

insert into accident values (14,'2008-02-17','Mysore_Road');

insert into accident values (15,'2004-03-05','Kanakapura road');



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 end
14	2008-02-17	Mysore_Road
15	2004-03-05	Kanakapura road
NULL	NULL	NULL

owns:

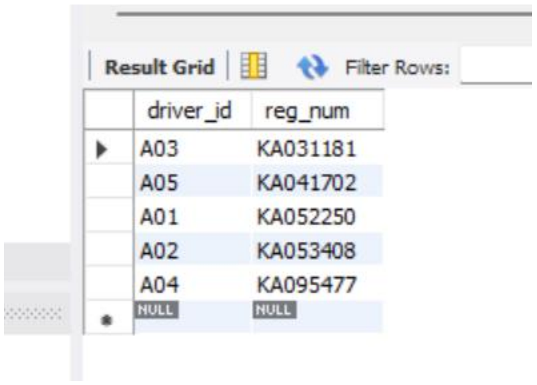
insert into owns values('A01','KA052250');

insert into owns values('A02','KA053408');

insert into owns values('A03','KA031181');

insert into owns values('A04','KA095477');

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



Result Grid | Filter Rows:

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

participated:

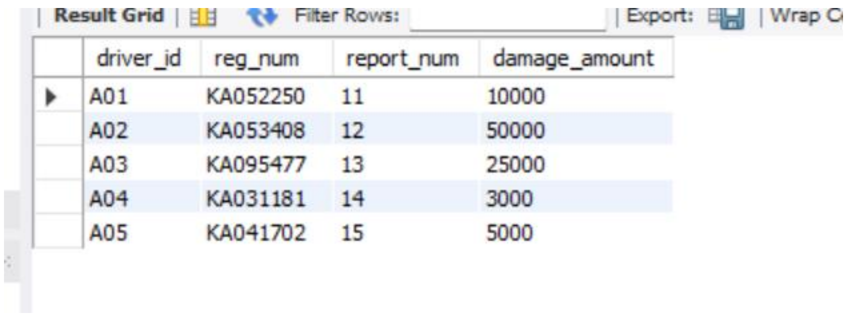
insert into participated values('A01','KA052250',11,10000);

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);



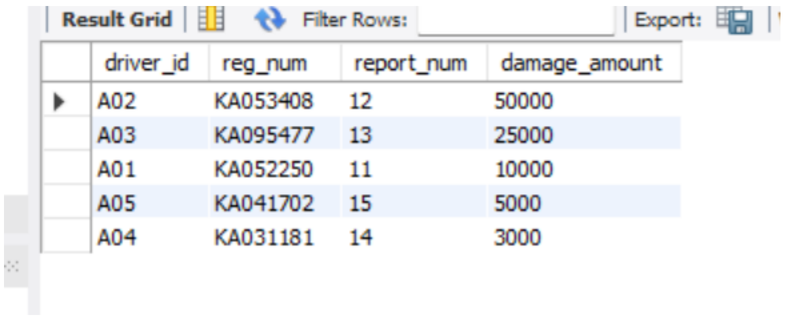
Result Grid | Filter Rows: | Export: | Wrap C

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

Queries:

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

SQL>select * from participated order by(damage_amout) desc;

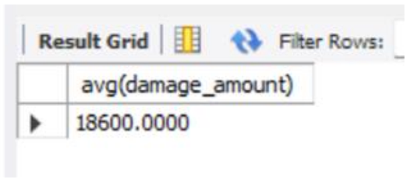


Result Grid | Filter Rows: | Export:

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

2.Find the average damage amount

SQL>select avg(damage_amount) from participated;



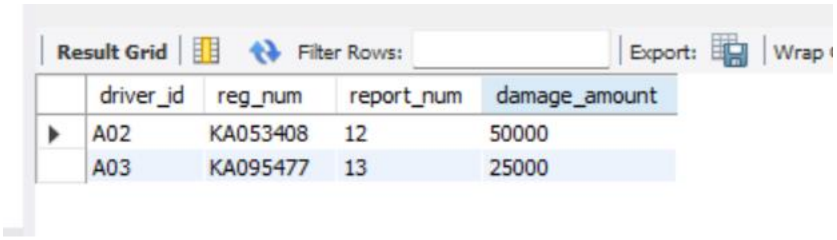
Result Grid | Filter Rows:

avg(damage_amount)
18600.0000

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

SQL>delete from participated where damage_amount<(select t.amt from(select avg(damage_amount)as amt from participated) t);

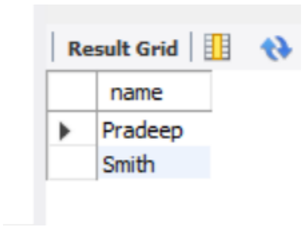
SQL>select * from participated;



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

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

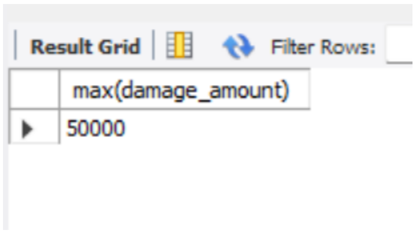
SQL>select name from person,participated where person.driver_id=participated.driver_id and damage_amount>(select avg(damage_amount) from participated);



name
Pradeep
Smith

5.Find maximum damage amount.

SQL>select max(damage_amount) from participated;



max(damage_amount)
50000

Bank Database

Question

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

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

BankCustomer (customer-name: String, customer-street: String,customer-city: String)

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

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

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

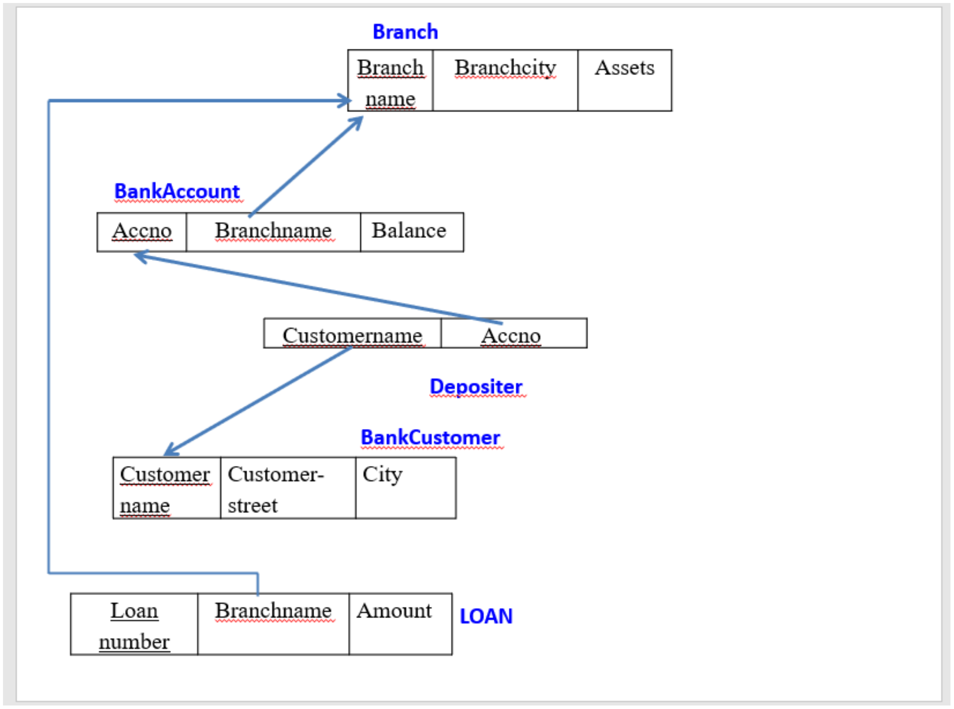
ii. Enter at least five tuples for each relation.

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

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

v.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 1bm21cs062_bankDb;

use 1bm21cs062_bankDb;

Creating Table

branch:

create table branch(

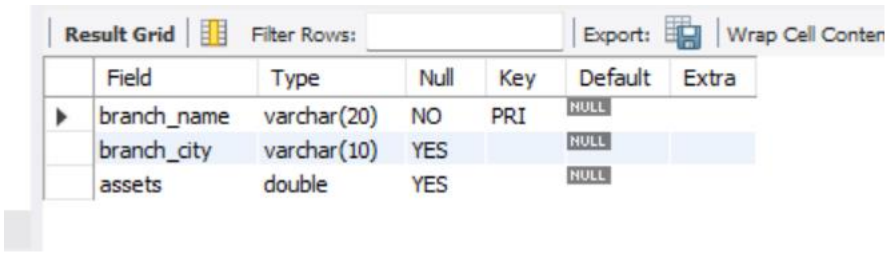
branch_name varchar(20),

branch_city varchar(10),

assets real,

PRIMARY KEY(branch_name)

);



Field	Type	Null	Key	Default	Extra
branch_name	varchar(20)	NO	PRI	NULL	
branch_city	varchar(10)	YES		NULL	
assets	double	YES		NULL	

customer:

create table bankCustomer(

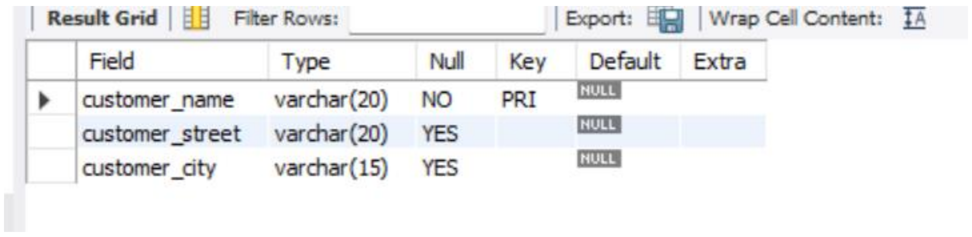
customer_name varchar(20),

customer_street varchar(20),

customer_city varchar(15),

PRIMARY KEY(customer_name)

);



Field	Type	Null	Key	Default	Extra
customer_name	varchar(20)	NO	PRI	NULL	
customer_street	varchar(20)	YES		NULL	
customer_city	varchar(15)	YES		NULL	

loan:

create table loan(

loan_no int,

branch_name varchar(20),

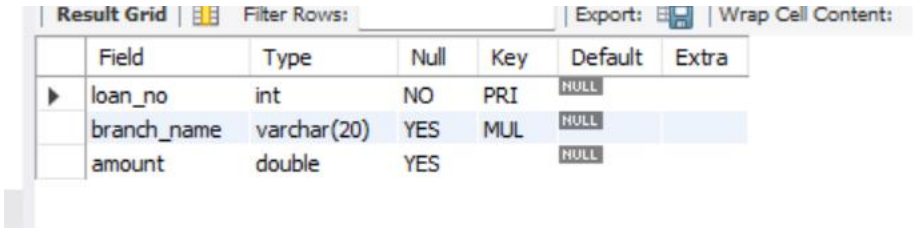
amount real,

PRIMARY KEY(loan_no),

FOREIGN KEY(branch_name) REFERENCES branch(branch_name)

ON UPDATE CASCADE ON DELETE CASCADE

);



Field	Type	Null	Key	Default	Extra
loan_no	int	NO	PRI	NULL	
branch_name	varchar(20)	YES	MUL	NULL	
amount	double	YES		NULL	

bankAccount:

create table bankAccount(

accno int,

branch_name varchar(20),

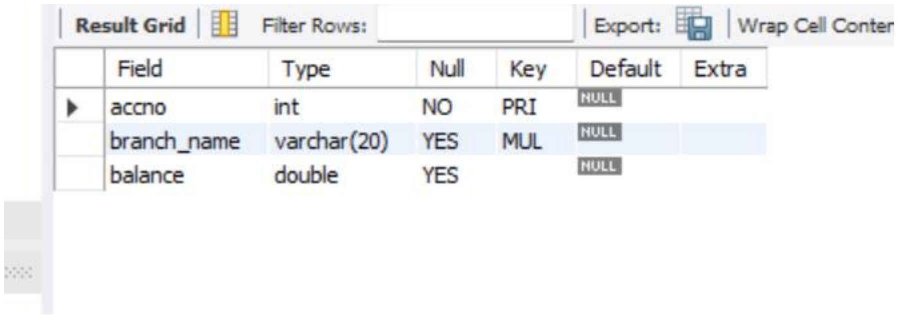
balance real,

PRIMARY KEY(accno),

FOREIGN KEY(branch_name) REFERENCES branch(branch_name)

ON UPDATE CASCADE ON DELETE CASCADE

);



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

depositor:

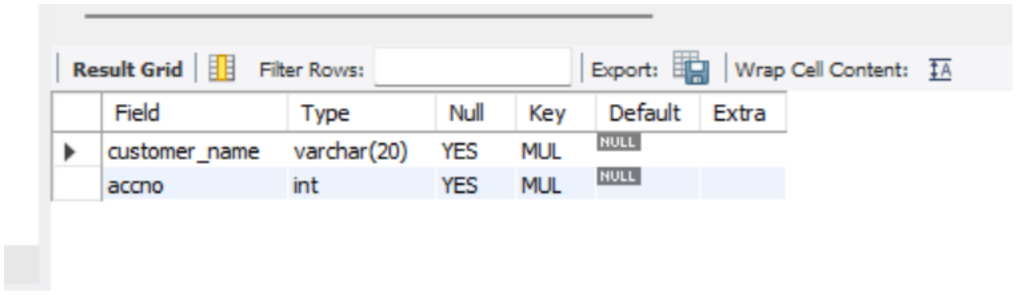
create table depositor(

customer_name varchar(20),

accno int,

FOREIGN KEY(customer_name) REFERENCES bankCustomer(customer_name)

```
ON UPDATE CASCADE ON DELETE CASCADE,
FOREIGN KEY(accno) REFERENCES bankAccount(accno)
ON UPDATE CASCADE ON DELETE CASCADE
);
```

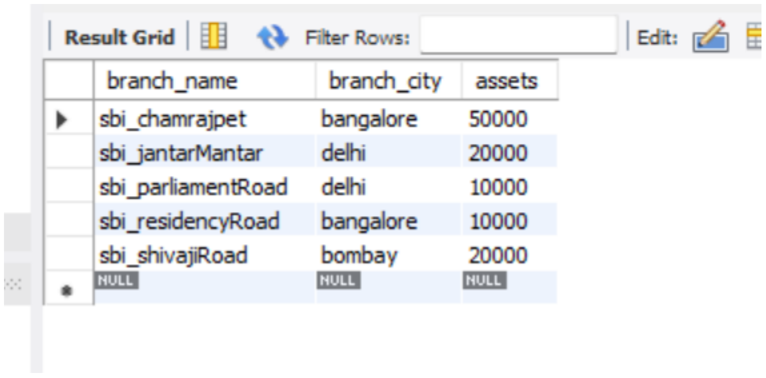


Field	Type	Null	Key	Default	Extra
customer_name	varchar(20)	YES	MUL	NULL	
accno	int	YES	MUL	NULL	

Inserting Values to the table

branch:

```
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);
```



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

bankAccount:

```
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);

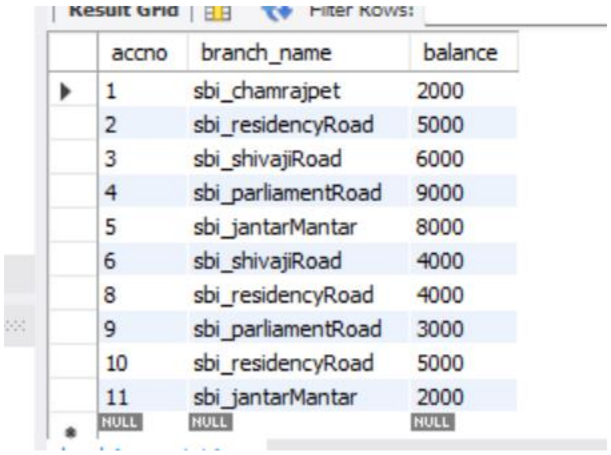
insert into bankAccount values(6,'sbi_shivajiRoad',4000);

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);



```
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
```

bankCustomer:

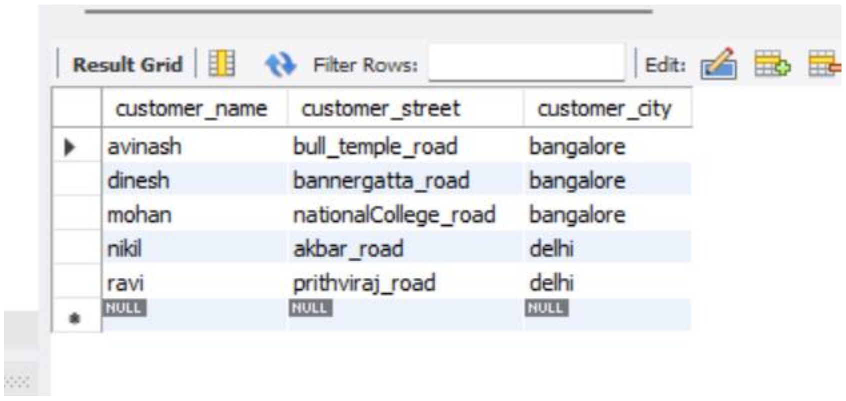
insert into bankCustomer values('avinash','bull_temple_road','bangalore');

insert into bankCustomer values('dinesh','bannergatta_road','bangalore');

insert into bankCustomer values('mohan','nationalCollege_road','bangalore');

insert into bankCustomer values('nikil','akbar_road','delhi');

insert into bankCustomer values('ravi','prithviraj_road','delhi');



Result Grid | Filter Rows: | Edit:

	customer_name	customer_street	customer_city
▶	avinash	bull_temple_road	bangalore
	dinesh	bannergatta_road	bangalore
	mohan	nationalCollege_road	bangalore
	nikil	akbar_road	delhi
	ravi	prithviraj_road	delhi
*	NULL	NULL	NULL

depositor:

insert into depositor values('avinash',1);

insert into depositor values('dinesh',2);

insert into depositor values('nikil',4);

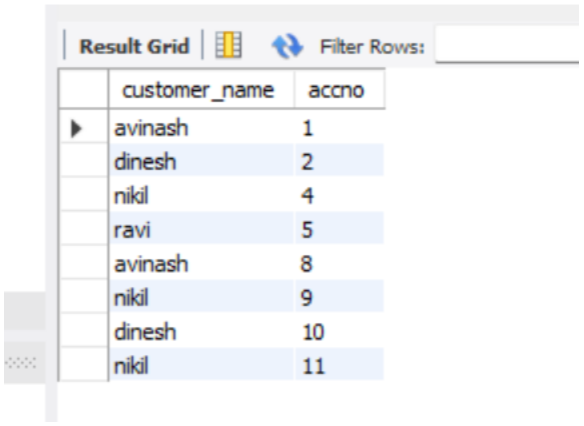
insert into depositor values('ravi',5);

insert into depositor values('avinash',8);

insert into depositor values('nikil',9);

insert into depositor values('dinesh',10);

insert into depositor values('nikil',11);



Result Grid | Filter Rows:

	customer_name	accno
▶	avinash	1
	dinesh	2
	nikil	4
	ravi	5
	avinash	8
	nikil	9
	dinesh	10
	nikil	11

loan:

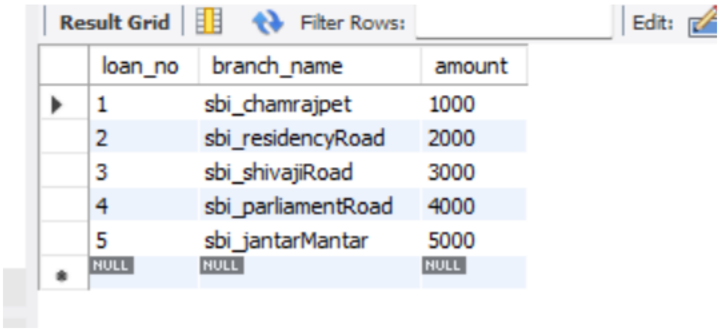
insert into loan values(1,'sbi_chamrajpet',1000);

insert into loan values(2,'sbi_residencyRoad',2000);

insert into loan values(3,'sbi_shivajiRoad',3000);

insert into loan values(4,'sbi_parliamentRoad',4000);

insert into loan values(5,'sbi_jantarMantar',5000);



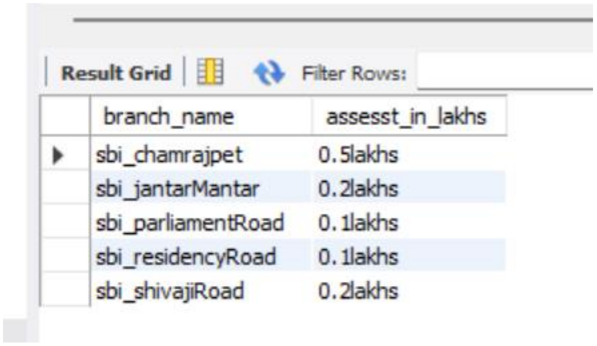
Result Grid | Filter Rows: | Edit:

loan_no	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

Queries

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

SQL> select branch_name, concat(assets/100000,'lakhs') as assesst_in_lakhs from branch;



Result Grid | Filter Rows:

branch_name	assesst_in_lakhs
sbi_chamrajpet	0.5lakhs
sbi_jantarMantar	0.2lakhs
sbi_parliamentRoad	0.1lakhs
sbi_residencyRoad	0.1lakhs
sbi_shivajiRoad	0.2lakhs

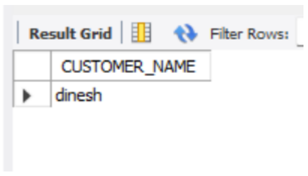
2. Find all the customers who have at least two accounts at the same branch

(ex. SBI_ResidencyRoad).

SQL>select d.customer_name as CUSTOMER_NAME from bankAccount b,depositor d

where b.branch_name='sbi_residencyRoad' and b.accno=d.accno

group by d.customer_name having count(d.accno)>=2;


Result Grid | Filter Rows:

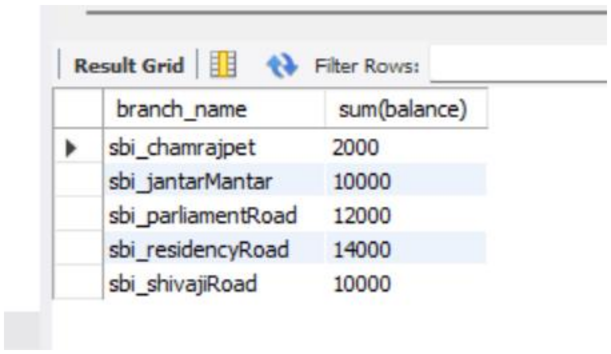
CUSTOMER_NAME
dinesh

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

SQL>create view sum_of_loan as select branch_name,sum(balance)

from bankAccount group by branch_name;

SQL>select * from sum_of_loan;


Result Grid | Filter Rows:

branch_name	sum(balance)
sbi_chamrajpet	2000
sbi_jantarMantar	10000
sbi_parliamentRoad	12000
sbi_residencyRoad	14000
sbi_shivajiRoad	10000

More Queries on Bank Database

Question

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

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

BankCustomer (customer-name: String, customer-street: String,customer-city: String)

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

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

Create the above tables by properly specifying the primary keys and the foreign keys &Enter at least five tuples for each relation.

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

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

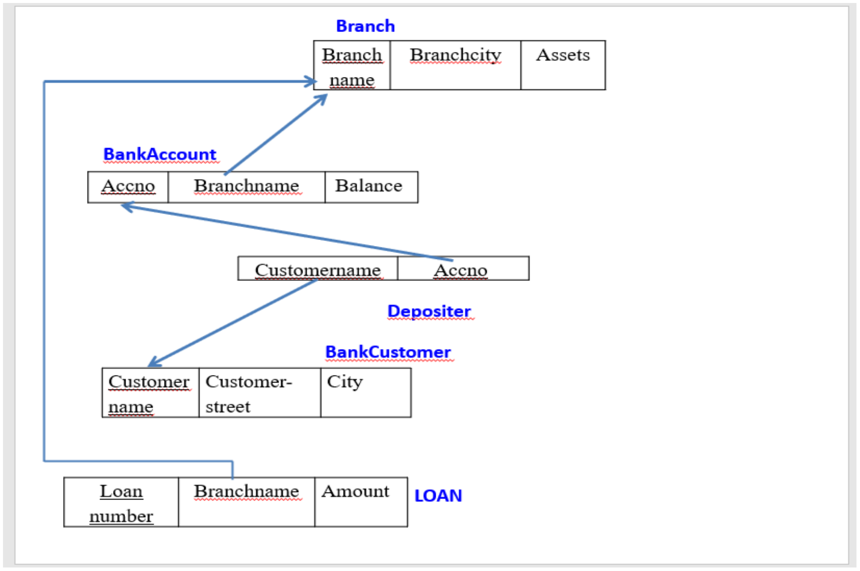
iii. Find all customers who have both an account and a loan at the Bangalore branch

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

v. Demonstrate how you delete all account tuples at every branchlocated in a specific city (Ex. Bombay).

vi. Update the Balance of all accounts by 5%

Schema Diagram



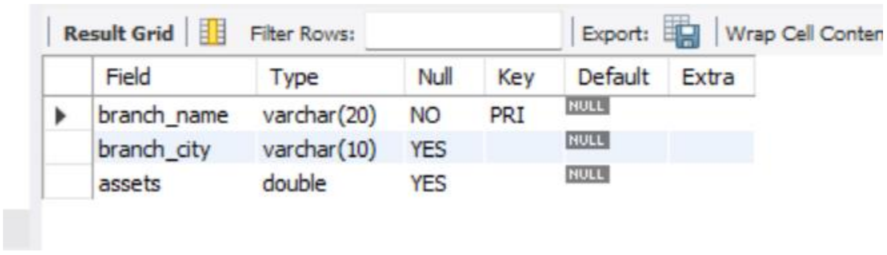
Create Database

```
create database 1bm21cs062_bankDb;
use 1bm21cs062_bankDb;
```

Creating Table

branch:

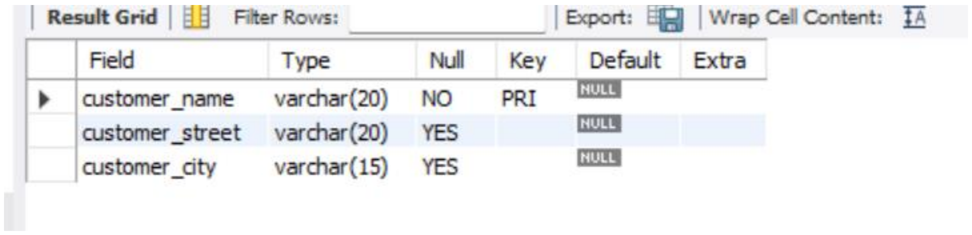
```
create table branch(
branch_name varchar(20),
branch_city varchar(10),
assets real,
PRIMARY KEY(branch_name)
);
```



Field	Type	Null	Key	Default	Extra
branch_name	varchar(20)	NO	PRI	NULL	
branch_city	varchar(10)	YES		NULL	
assets	double	YES		NULL	

customer:

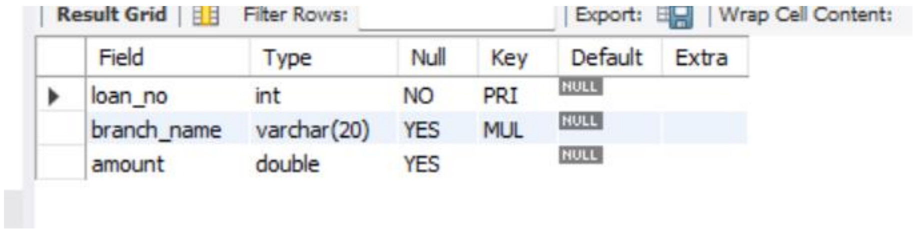
```
create table bankCustomer(
customer_name varchar(20),
customer_street varchar(20),
customer_city varchar(15),
PRIMARY KEY(customer_name)
);
```



Field	Type	Null	Key	Default	Extra
customer_name	varchar(20)	NO	PRI	NULL	
customer_street	varchar(20)	YES		NULL	
customer_city	varchar(15)	YES		NULL	

loan:

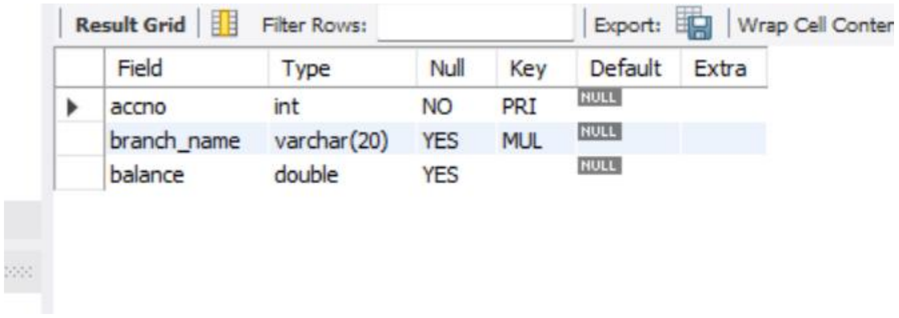
```
create table loan(
loan_no int,
branch_name varchar(20),
amount real,
PRIMARY KEY(loan_no),
FOREIGN KEY(branch_name) REFERENCES branch(branch_name)
ON UPDATE CASCADE ON DELETE CASCADE
);
```


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

Field	Type	Null	Key	Default	Extra
loan_no	int	NO	PRI	NULL	
branch_name	varchar(20)	YES	MUL	NULL	
amount	double	YES		NULL	

bankAccount:

```
create table bankAccount(
accno int,
branch_name varchar(20),
balance real,
PRIMARY KEY(accno),
FOREIGN KEY(branch_name) REFERENCES branch(branch_name)
ON UPDATE CASCADE ON DELETE CASCADE
);
```


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

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

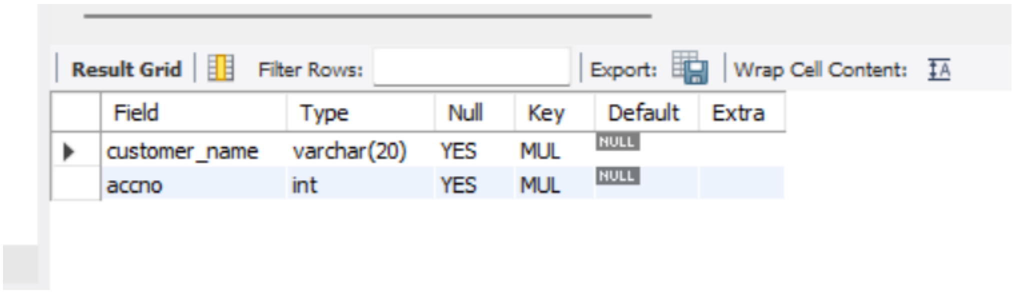
depositor:

```
create table depositor(
customer_name varchar(20),
accno int,
FOREIGN KEY(customer_name) REFERENCES bankCustomer(customer_name)
ON UPDATE CASCADE ON DELETE CASCADE,
```

FOREIGN KEY(accno) REFERENCES bankAccount(accno)

ON UPDATE CASCADE ON DELETE CASCADE

);



Field	Type	Null	Key	Default	Extra
customer_name	varchar(20)	YES	MUL	NULL	
accno	int	YES	MUL	NULL	

borrower:

create table borrower(

customer_name varchar(20),

accno int,

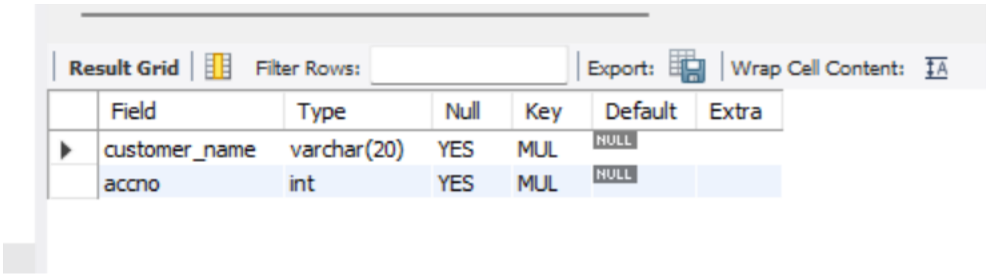
FOREIGN KEY(customer_name) REFERENCES bankCustomer(customer_name)

ON UPDATE CASCADE ON DELETE CASCADE,

FOREIGN KEY(accno) REFERENCES bankAccount(accno)

ON UPDATE CASCADE ON DELETE CASCADE

);



Field	Type	Null	Key	Default	Extra
customer_name	varchar(20)	YES	MUL	NULL	
accno	int	YES	MUL	NULL	

Inserting Values to the table

branch:

insert into branch values('sbi_chamrajpet','bangalore',50000);

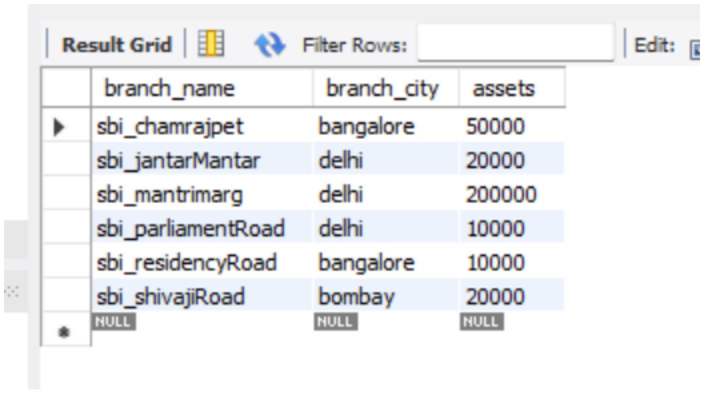
insert into branch values('sbi_residencyRoad','bangalore',10000);

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

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

insert into branch values('sbi_jantarMantar','delhi',20000);

insert into branch values('sbi_mantrimarg','delhi',200000);



branch_name	branch_city	assets
sbi_chamrajpet	bangalore	50000
sbi_jantarMantar	delhi	20000
sbi_mantrimarg	delhi	200000
sbi_parliamentRoad	delhi	10000
sbi_residencyRoad	bangalore	10000
sbi_shivajiRoad	bombay	20000
NULL	NULL	NULL

bankAccount:

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);

insert into bankAccount values(12,'sbi_mantrimarg',2000);

bankCustomer:

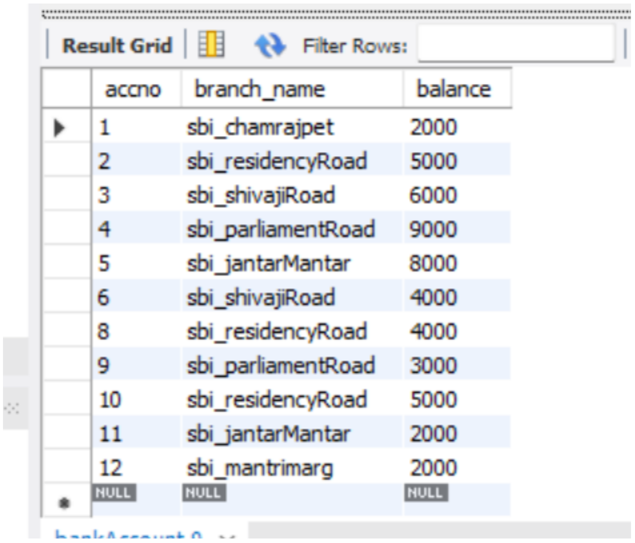
insert into bankCustomer values('avinash','bull_temple_road','bangalore');

insert into bankCustomer values('dinesh','bannergatta_road','bangalore');

insert into bankCustomer values('mohan','nationalCollege_road','bangalore');

insert into bankCustomer values('nikil','akbar_road','delhi');

insert into bankCustomer values('ravi','prithviraj_road','delhi');



```
Result Grid | Filter Rows:
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
12     sbi_mantrimarg       2000
NULL   NULL                 NULL
```

depositor:

insert into depositor values('avinash',1);

insert into depositor values('dinesh',2);

insert into depositor values('nikil',4);

insert into depositor values('ravi',5);

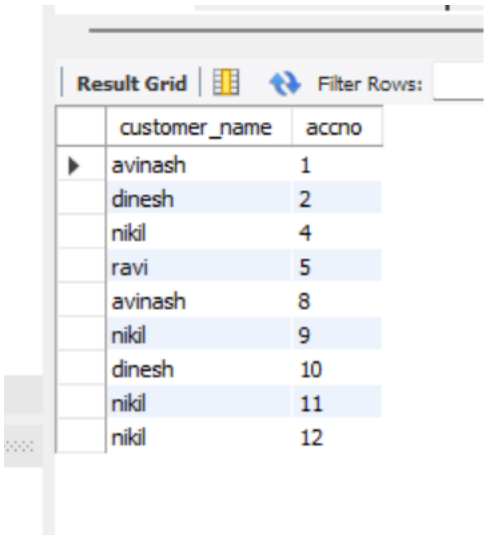
insert into depositor values('avinash',8);

insert into depositor values('nikil',9);

insert into depositor values('dinesh',10);

insert into depositor values('nikil',11);

insert into depositor values('nikil',12);



Result Grid | Filter Rows:

customer_name	accno
avinash	1
dinesh	2
nikil	4
ravi	5
avinash	8
nikil	9
dinesh	10
nikil	11
nikil	12

loan:

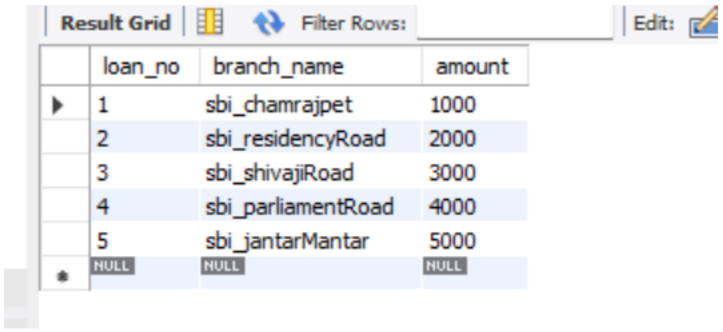
insert into loan values(1,'sbi_chamrajpet',1000);

insert into loan values(2,'sbi_residencyRoad',2000);

insert into loan values(3,'sbi_shivajiRoad',3000);

insert into loan values(4,'sbi_parliamentRoad',4000);

insert into loan values(5,'sbi_jantarMantar',5000);



Result Grid | Filter Rows: | Edit:

loan_no	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

borrower:

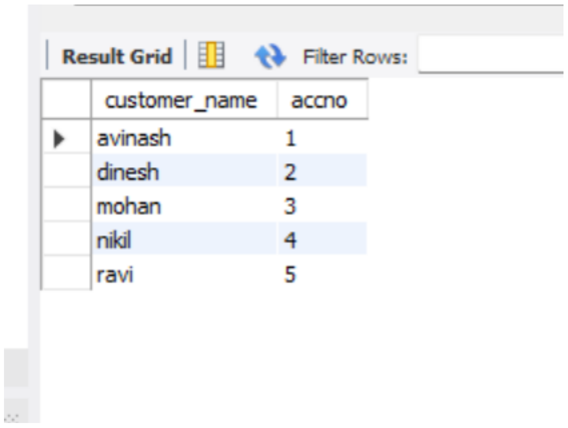
insert into borrower values('avinash',1);

insert into borrower values('dinesh',2);

insert into borrower values('mohan',3);

insert into borrower values('nikil',4);

insert into borrower values('ravi',5);



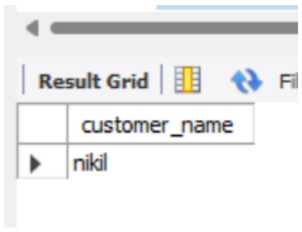
customer_name	accno
avinash	1
dinesh	2
mohan	3
nikil	4
ravi	5

Queries

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

SQL>

```
SELECT d.customer_name
FROM depositor d
INNER JOIN bankAccount a ON d.accno = a.accno
INNER JOIN branch b ON a.branch_name = b.branch_name
WHERE b.branch_city = 'Delhi'
GROUP BY d.customer_name
HAVING COUNT(DISTINCT B.branch_name) = (
SELECT COUNT(branch_name)
FROM branch
WHERE branch_city = 'Delhi');
```



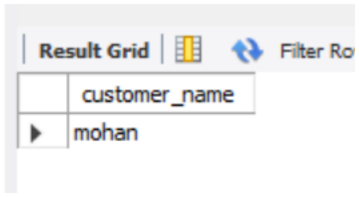
customer_name
nikil

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

SQL>

select customer_name from borrower

where customer_name not in(select customer_name from depositor);



customer_name
mohan

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

SQL>

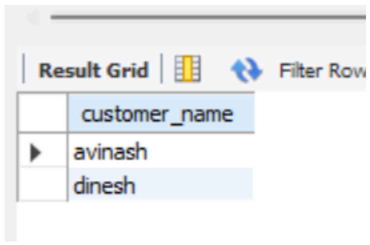
select distinct d.customer_name from depositor d

where d.customer_name IN(select d.customer_name from branch b,depositor

d,bankAccount ba where b.branch_city='bangalore'

and b.branch_name=ba.branch_name and ba.accno=d.accno and customer_name

IN(select customer_name from borrower));



customer_name
avinash
dinesh

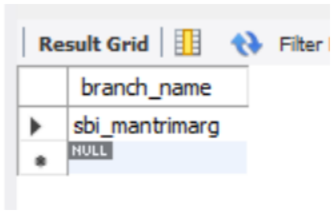
4. Find the names of all branches that have greater assets than all branches located in Bangalore.

SQL>

select b.branch_name from branch b

where b.assets>ALL(select sum(b.assets) from branch b where

b.branch_city='bangalore');



```
branch_name
sbi_mantrimarg
NULL
```

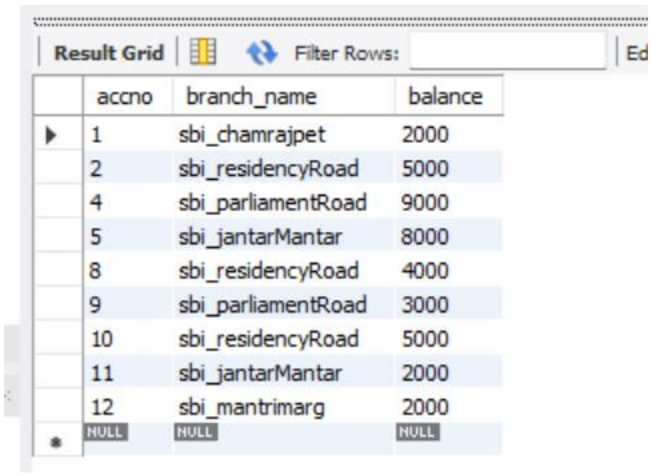
5. Demonstrate how you delete all account tuples at every branch located in a specific city (Ex. Bombay).

SQL>

delete ba.*from bankAccount ba,branch b

where branch_city='bombay'and ba.branch_name=b.branch_name;

SQL>select *from bankAccount;

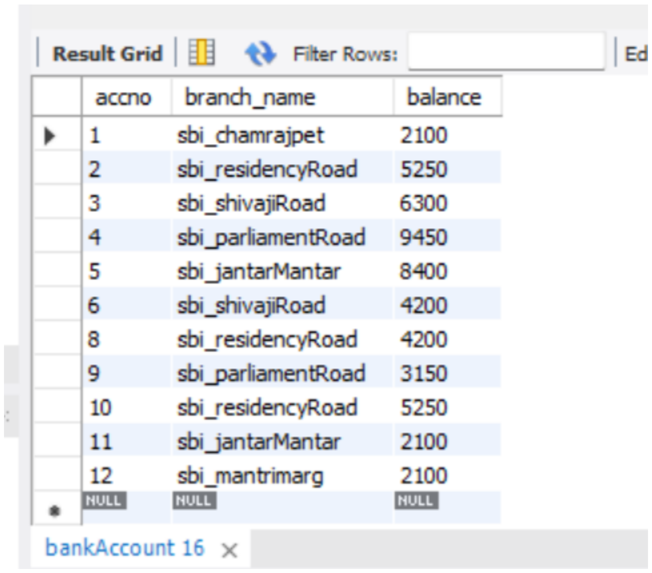


```
accno  branch_name         balance
1      sbi_chamrajpet      2000
2      sbi_residencyRoad   5000
4      sbi_parliamentRoad  9000
5      sbi_jantarMantar    8000
8      sbi_residencyRoad   4000
9      sbi_parliamentRoad  3000
10     sbi_residencyRoad   5000
11     sbi_jantarMantar    2000
12     sbi_mantrimarg      2000
NULL   NULL                NULL
```

6. Update the Balance of all accounts by 5%

SQL> UPDATE bankAccount set balance=(0.05*balance)+balance;

select * from bankAccount;



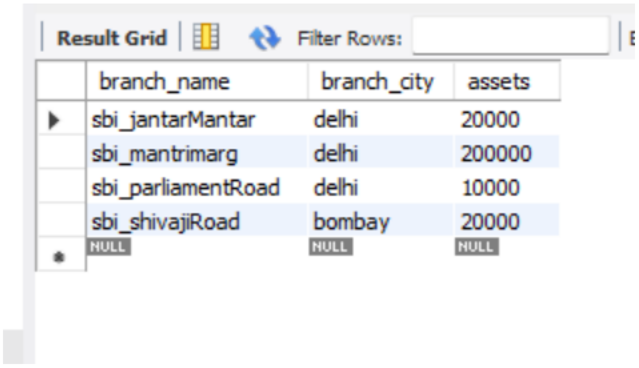
accno	branch_name	balance
1	sbi_chamrajpet	2100
2	sbi_residencyRoad	5250
3	sbi_shivajiRoad	6300
4	sbi_parliamentRoad	9450
5	sbi_jantarMantar	8400
6	sbi_shivajiRoad	4200
8	sbi_residencyRoad	4200
9	sbi_parliamentRoad	3150
10	sbi_residencyRoad	5250
11	sbi_jantarMantar	2100
12	sbi_mantrimarg	2100
NULL	NULL	NULL

bankAccount 16

7.(ON SPOT)How can you delete all branches in specific city located in bangalore?

SQL>delete from branch where branch_city='bangalore';

SQL>select * from branch;



branch_name	branch_city	assets
sbi_jantarMantar	delhi	20000
sbi_mantrimarg	delhi	200000
sbi_parliamentRoad	delhi	10000
sbi_shivajiRoad	bombay	20000
NULL	NULL	NULL

Employee Database

Question

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

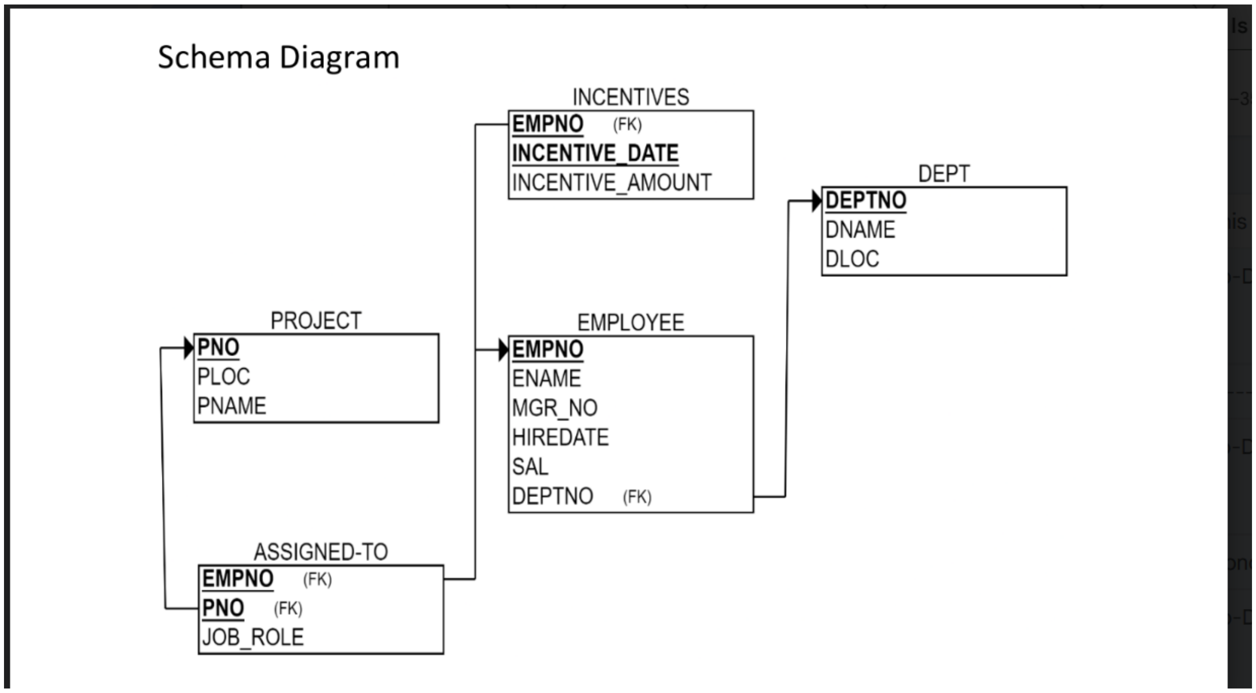
2. Enter greater than five tuples for each table.

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

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

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

Schema Diagram



Create Database

create database 1bm21cs062_emp_id;

use 1bm21cs062_emp_id;

Creating Table

dept:

create table dept(

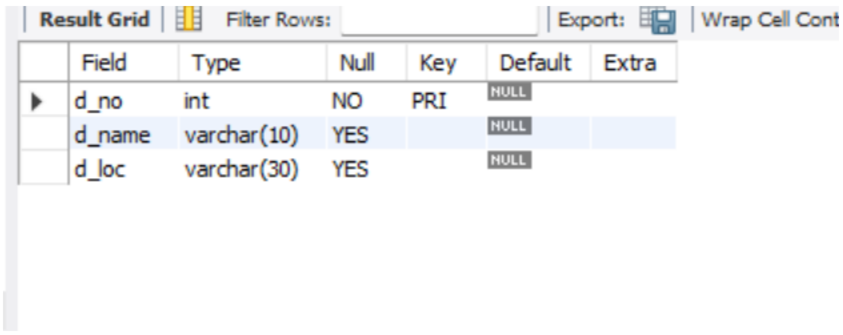
d_no int,

d_name varchar (10),

d_loc varchar (30),

primary key(d_no)

);



Field	Type	Null	Key	Default	Extra
d_no	int	NO	PRI	NULL	
d_name	varchar(10)	YES		NULL	
d_loc	varchar(30)	YES		NULL	

project:

create table project(

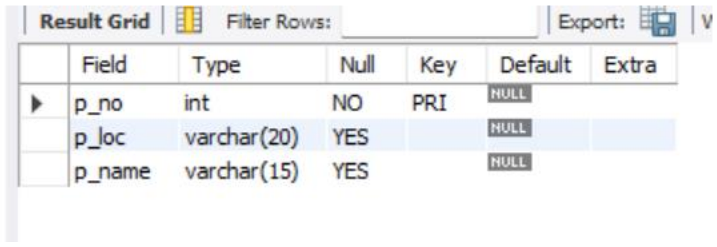
p_no int,

p_loc varchar(20),

p_name varchar(15),

PRIMARY KEY(p_no)

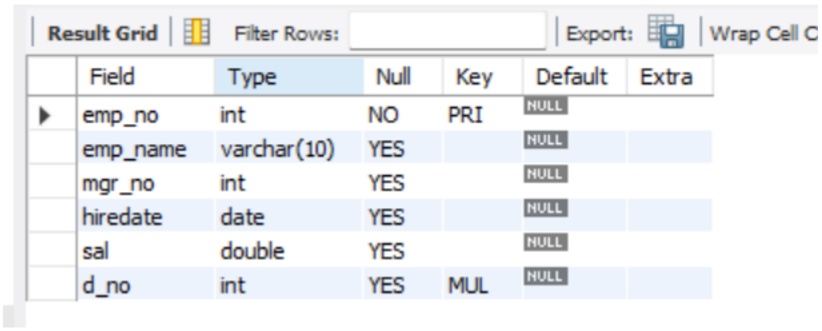
);



Field	Type	Null	Key	Default	Extra
p_no	int	NO	PRI	NULL	
p_loc	varchar(20)	YES		NULL	
p_name	varchar(15)	YES		NULL	

employee:

```
create table employee(
emp_no int,
emp_name varchar(10),
mgr_no int,
hiredate date,
sal real,
d_no int,
primary key(emp_no),
foreign key(d_no) references dept(d_no)
on update cascade on delete cascade
);
```



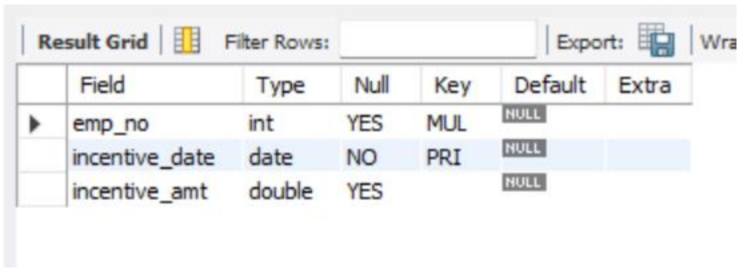
Field	Type	Null	Key	Default	Extra
emp_no	int	NO	PRI	NULL	
emp_name	varchar(10)	YES		NULL	
mgr_no	int	YES		NULL	
hiredate	date	YES		NULL	
sal	double	YES		NULL	
d_no	int	YES	MUL	NULL	

incentives:

```
create table incentives(
emp_no int,
incentive_date date,
incentive_amt real,
primary key(incentive_date),
foreign key(emp_no) references employee(emp_no)
```

on update cascade on delete cascade

);



Result Grid | Filter Rows: | Export: | Wra

	Field	Type	Null	Key	Default	Extra
▶	emp_no	int	YES	MUL	NULL	
	incentive_date	date	NO	PRI	NULL	
	incentive_amt	double	YES		NULL	

assigned:

create table assigned(

emp_no int,

p_no int,

job_role varchar(10),

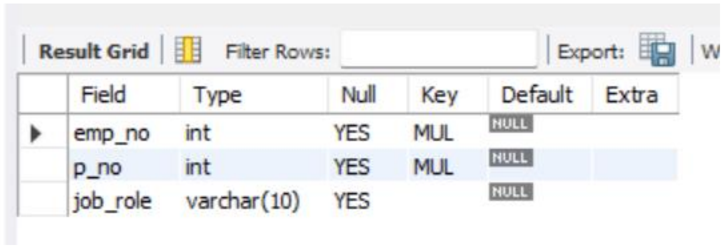
foreign key(emp_no) references employee(emp_no)

on update cascade on delete cascade,

foreign key(p_no) references project(p_no)

on update cascade on delete cascade

);



Result Grid | Filter Rows: | Export: | W

	Field	Type	Null	Key	Default	Extra
▶	emp_no	int	YES	MUL	NULL	
	p_no	int	YES	MUL	NULL	
	job_role	varchar(10)	YES		NULL	

Inserting Values to the table

dept:

insert into dept values(10,'IT','mysore');

insert into dept values(20,'Marketing','patna');

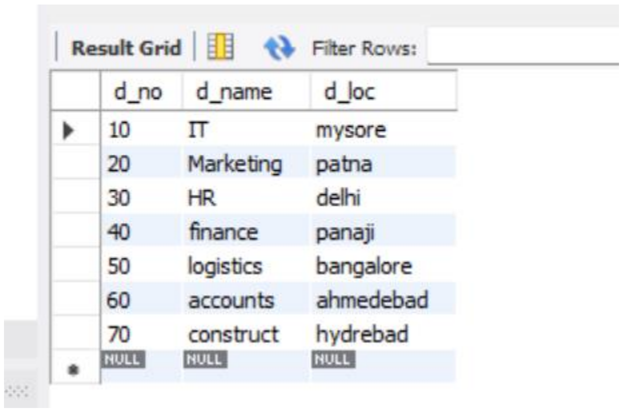
insert into dept values(30,'HR','delhi');

insert into dept values(40,'finance','panaji');

insert into dept values(50,'logistics','bangalore');

insert into dept values(60,'accounts','ahmedebad');

insert into dept values(70,'construct','hydrebad');



d_no	d_name	d_loc
10	IT	mysore
20	Marketing	patna
30	HR	delhi
40	finance	panaji
50	logistics	bangalore
60	accounts	ahmedebad
70	construct	hydrebad
NULL	NULL	NULL

project:

insert into project values(1,'mysore','0A1B1');

insert into project values(2,'patna','0A2B2');

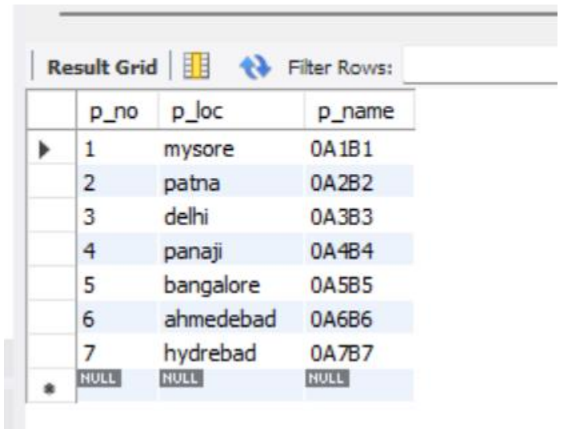
insert into project values(3,'delhi','0A3B3');

insert into project values(4,'panaji','0A4B4');

insert into project values(5,'bangalore','0A5B5');

insert into project values(6,'ahmedebad','0A6B6');

insert into project values(7,'hydrebad','0A7B7');

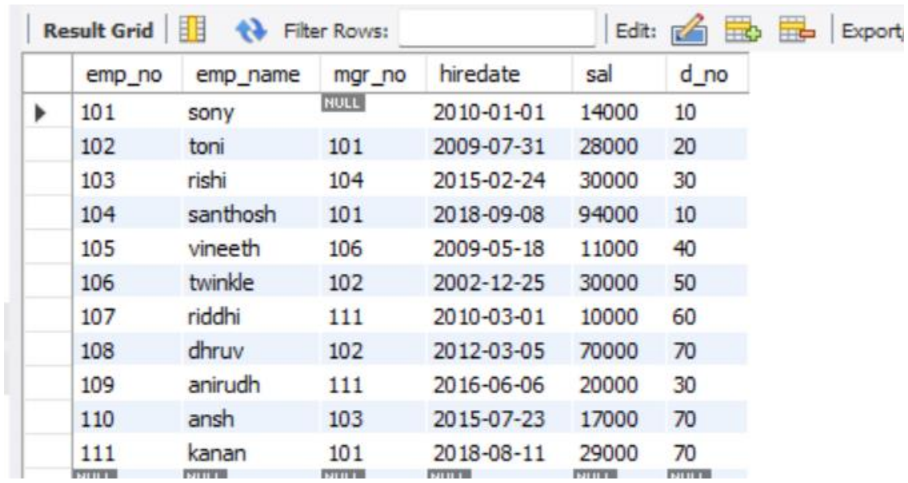


Result Grid | Filter Rows:

p_no	p_loc	p_name
1	mysore	0A1B1
2	patna	0A2B2
3	delhi	0A3B3
4	panaji	0A4B4
5	bangalore	0A5B5
6	ahmedebad	0A6B6
7	hydrebad	0A7B7
NULL	NULL	NULL

employee:

```
insert into employee values(101,'sony',null,'2010-01-01',14000,10);
insert into employee values(102,'toni',101,'2009-07-31',28000,20);
insert into employee values(103,'rishi',104,'2015-02-24',30000,30);
insert into employee values(104,'santhosh',101,'2018-09-08',94000,10);
insert into employee values(105,'vineeth',106,'2009-05-18',11000,40);
insert into employee values(106,'twinkle',102,'2002-12-25',30000,50);
insert into employee values(107,'riddhi',111,'2010-03-01',10000,60);
insert into employee values(108,'dhruv',102,'2012-03-05',70000,70);
insert into employee values(109,'anirudh',111,'2016-06-06',20000,30);
insert into employee values(110,'ansh',103,'2015-07-23',17000,70);
insert into employee values(111,'kanan',101,'2018-08-11',29000,70);
```



emp_no	emp_name	mgr_no	hiredate	sal	d_no
101	sony	NULL	2010-01-01	14000	10
102	toni	101	2009-07-31	28000	20
103	rishi	104	2015-02-24	30000	30
104	santhosh	101	2018-09-08	94000	10
105	vineeth	106	2009-05-18	11000	40
106	twinkle	102	2002-12-25	30000	50
107	riddhi	111	2010-03-01	10000	60
108	dhruv	102	2012-03-05	70000	70
109	anirudh	111	2016-06-06	20000	30
110	ansh	103	2015-07-23	17000	70
111	kanan	101	2018-08-11	29000	70

incentives:

insert into incentives values(103,'2022-07-12',2500);

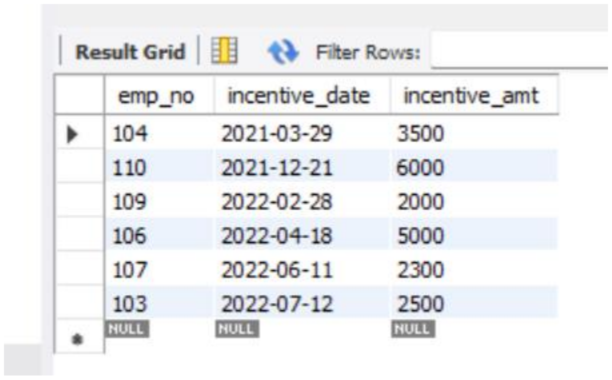
insert into incentives values(104,'2021-03-29',3500);

insert into incentives values(109,'2022-02-28',2000);

insert into incentives values(110,'2021-12-21',6000);

insert into incentives values(106,'2022-04-18',5000);

insert into incentives values(107,'2022-06-11',2300);



emp_no	incentive_date	incentive_amt
104	2021-03-29	3500
110	2021-12-21	6000
109	2022-02-28	2000
106	2022-04-18	5000
107	2022-06-11	2300
103	2022-07-12	2500
NULL	NULL	NULL

assigned:

insert into assigned values(101,1,'manager');

insert into assigned values(102,2,'assistant');

insert into assigned values(103,3,'supervisor');

insert into assigned values(104,1,'accountant');

insert into assigned values(105,4,'clerk');

insert into assigned values(106,7,'peon');

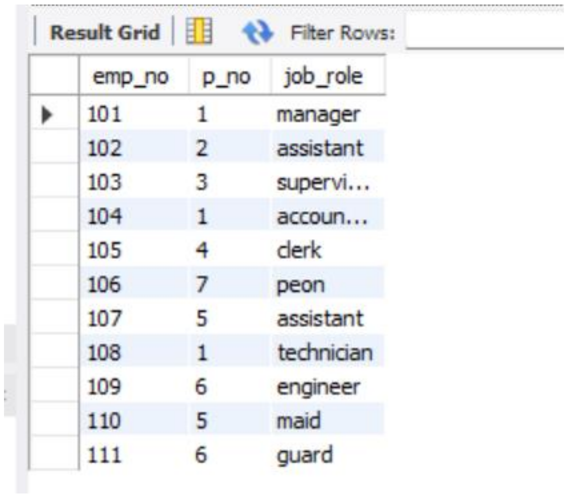
insert into assigned values(107,5,'assistant');

insert into assigned values(108,1,'technician');

insert into assigned values(109,6,'engineer');

insert into assigned values(110,5,'maid');

insert into assigned values(111,6,'guard');



emp_no	p_no	job_role
101	1	manager
102	2	assistant
103	3	supervi...
104	1	accoun...
105	4	clerk
106	7	peon
107	5	assistant
108	1	technician
109	6	engineer
110	5	maid
111	6	guard

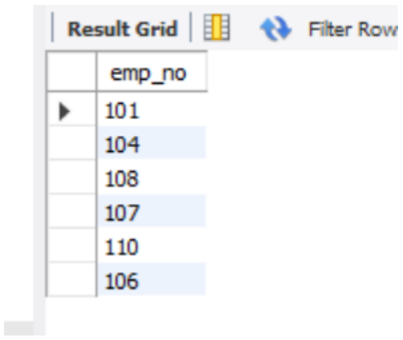
Queries

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

SQL>

select emp_no from assigned, project where assigned.p_no=project.p_no and project.p_loc

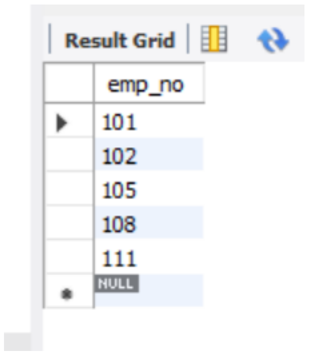
in('mysore','bangalore','hydrebad');


```
emp_no
101
104
108
107
110
106
```

2. Get Employee IDs of those employees who didn’t receive incentives

SQL>

select emp_no from employee where emp_no not in(select emp_no from incentives);


```
emp_no
101
102
105
108
111
NULL
```

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

SQL>

select e.emp_name,e.emp_no,d.d_name,a.job_role,d.d_loc,p.p_loc

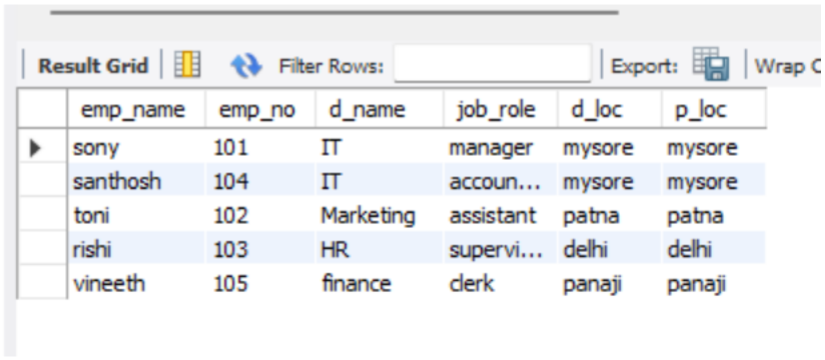
from employee e, dept d, assigned a, project p

where d.d_no=e.d_no

and e.emp_no=a.emp_no

and a.p_no=p.p_no

and p.p_loc=d.d_loc;



	emp_name	emp_no	d_name	job_role	d_loc	p_loc
▶	sony	101	IT	manager	mysore	mysore
	santhosh	104	IT	accoun...	mysore	mysore
	toni	102	Marketing	assistant	patna	patna
	rishi	103	HR	supervi...	delhi	delhi
	vineeth	105	finance	clerk	panaji	panaji

4. (ON SPOT)Find the employee's name, department name and job role of an employee who received the maximum incentive in year 2022.

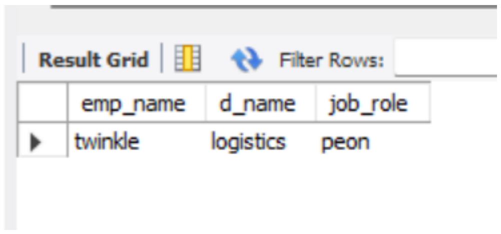
SQL> select e.emp_name,d.d_name,a.job_role

from employee e,dept d,assigned a

where e.emp_no in(select emp_no from incentives where incentive_amt=

(select max(incentive_amt)from incentives where incentive_date between '2022-01-01' and '2022-12-31'))

and d.d_no=e.d_no and e.emp_no=a.emp_no



	emp_name	d_name	job_role
▶	twinkle	logistics	peon

More Queries On Employee Database

Question

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

2. Enter greater than five tuples for each table.

3. 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.

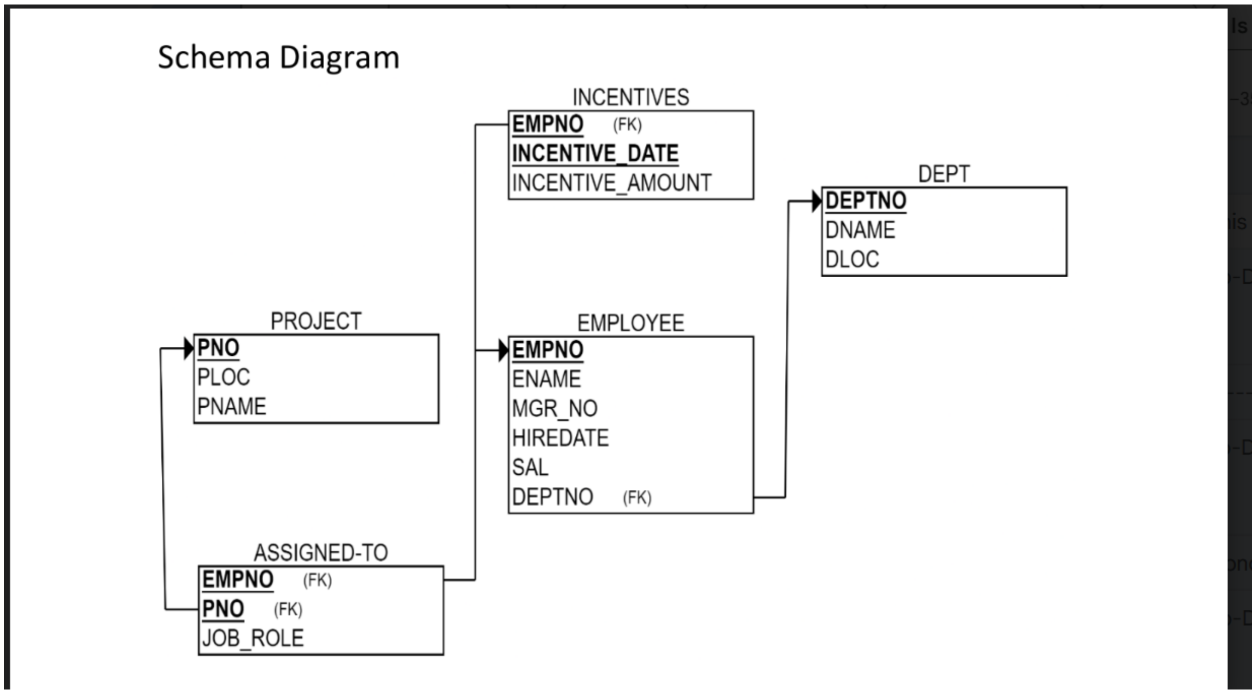
5. Find the name of the second top level managers of each department.

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.

Schema Diagram



Create Database

create database 1bm21cs062_emp_id;

use 1bm21cs062_emp_id;

Creating Table

dept:

create table dept(

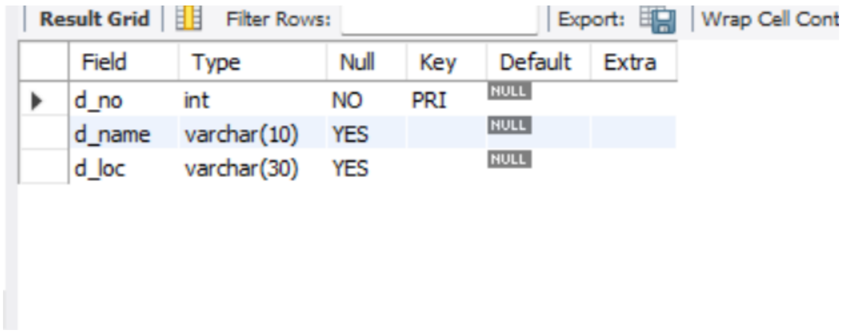
d_no int,

d_name varchar (10),

d_loc varchar (30),

primary key(d_no)

);



```
Field   Type         Null  Key  Default  Extra
d_no    int          NO    PRI  NULL
d_name  varchar(10)  YES        NULL
d_loc   varchar(30)  YES        NULL
```

project:

create table project(

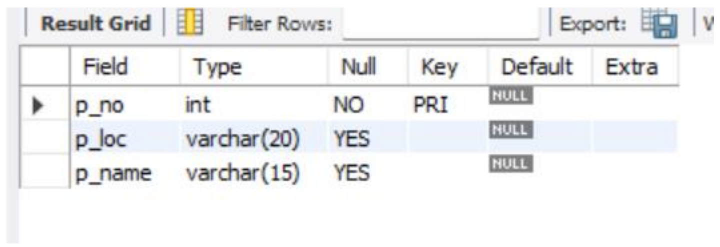
p_no int,

p_loc varchar(20),

p_name varchar(15),

PRIMARY KEY(p_no)

);



Field	Type	Null	Key	Default	Extra
p_no	int	NO	PRI	NULL	
p_loc	varchar(20)	YES		NULL	
p_name	varchar(15)	YES		NULL	

employee:

create table employee(

emp_no int,

emp_name varchar(10),

mgr_no int,

hiredate date,

sal real,

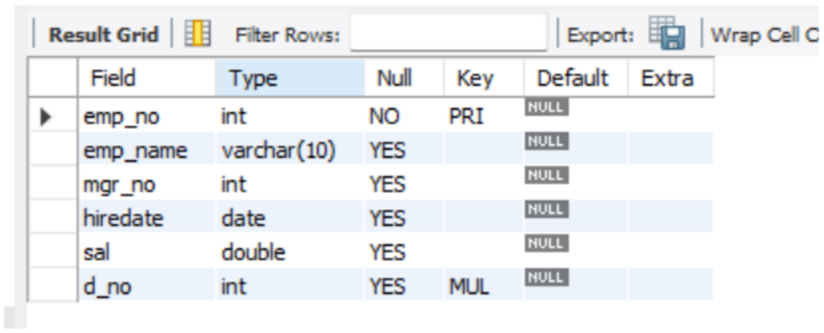
d_no int,

primary key(emp_no),

foreign key(d_no) references dept(d_no)

on update cascade on delete cascade

);

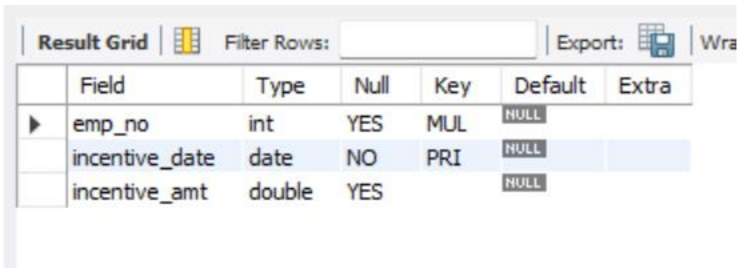


Field	Type	Null	Key	Default	Extra
emp_no	int	NO	PRI	NULL	
emp_name	varchar(10)	YES		NULL	
mgr_no	int	YES		NULL	
hiredate	date	YES		NULL	
sal	double	YES		NULL	
d_no	int	YES	MUL	NULL	

incentives:

create table incentives(

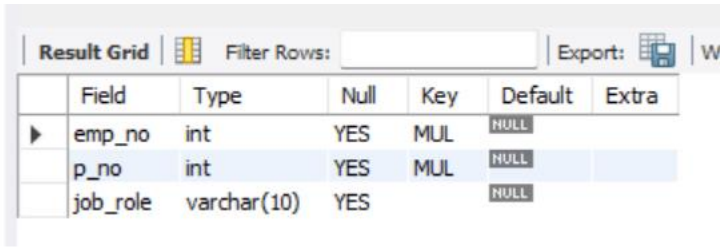
```
emp_no int,
incentive_date date,
incentive_amt real,
primary key(incentive_date),
foreign key(emp_no) references employee(emp_no)
on update cascade on delete cascade
);
```



Field	Type	Null	Key	Default	Extra
emp_no	int	YES	MUL	NULL	
incentive_date	date	NO	PRI	NULL	
incentive_amt	double	YES		NULL	

assigned:

```
create table assigned(
emp_no int,
p_no int,
job_role varchar(10),
foreign key(emp_no) references employee(emp_no)
on update cascade on delete cascade,
foreign key(p_no) references project(p_no)
on update cascade on delete cascade
);
```



Field	Type	Null	Key	Default	Extra
emp_no	int	YES	MUL	NULL	
p_no	int	YES	MUL	NULL	
job_role	varchar(10)	YES		NULL	

Inserting Values to the table

dept:

insert into dept values(10,'IT','mysore');

insert into dept values(20,'Marketing','patna');

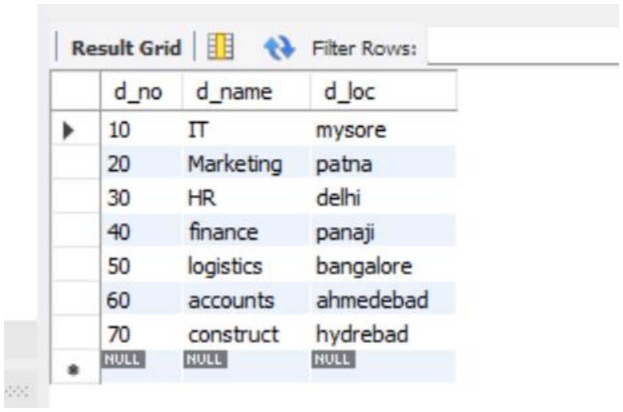
insert into dept values(30,'HR','delhi');

insert into dept values(40,'finance','panaji');

insert into dept values(50,'logistics','bangalore');

insert into dept values(60,'accounts','ahmedebad');

insert into dept values(70,'construct','hydrebad');



d_no	d_name	d_loc
10	IT	mysore
20	Marketing	patna
30	HR	delhi
40	finance	panaji
50	logistics	bangalore
60	accounts	ahmedebad
70	construct	hydrebad
NULL	NULL	NULL

project:

insert into project values(1,'mysore','0A1B1');

insert into project values(2,'patna','0A2B2');

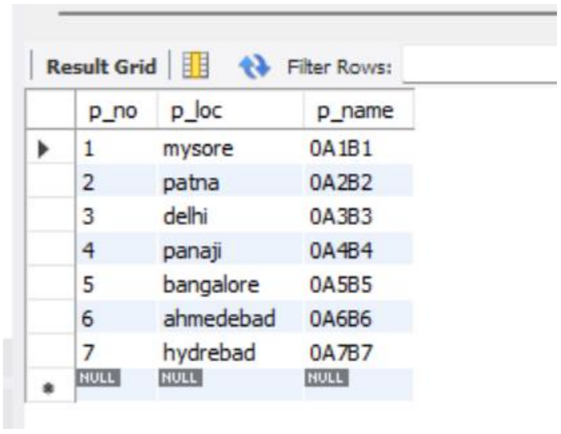
insert into project values(3,'delhi','0A3B3');

insert into project values(4,'panaji','0A4B4');

insert into project values(5,'bangalore','0A5B5');

insert into project values(6,'ahmedebad','0A6B6');

insert into project values(7,'hydrebad','0A7B7');



Result Grid | Filter Rows:

p_no	p_loc	p_name
1	mysore	0A1B1
2	patna	0A2B2
3	delhi	0A3B3
4	panaji	0A4B4
5	bangalore	0A5B5
6	ahmedebad	0A6B6
7	hydrebad	0A7B7
NULL	NULL	NULL

employee:

insert into employee values(101,'sony',null,'2010-01-01',140000,10);

insert into employee values(102,'toni',104,'2009-07-31',28000,20);

insert into employee values(103,'rishi',104,'2015-02-24',30000,30);

insert into employee values(104,'santhosh',101,'2018-09-08',94000,10);

insert into employee values(105,'vineeth',108,'2009-05-18',11000,40);

insert into employee values(106,'twinkle',104,'2002-12-25',30000,50);

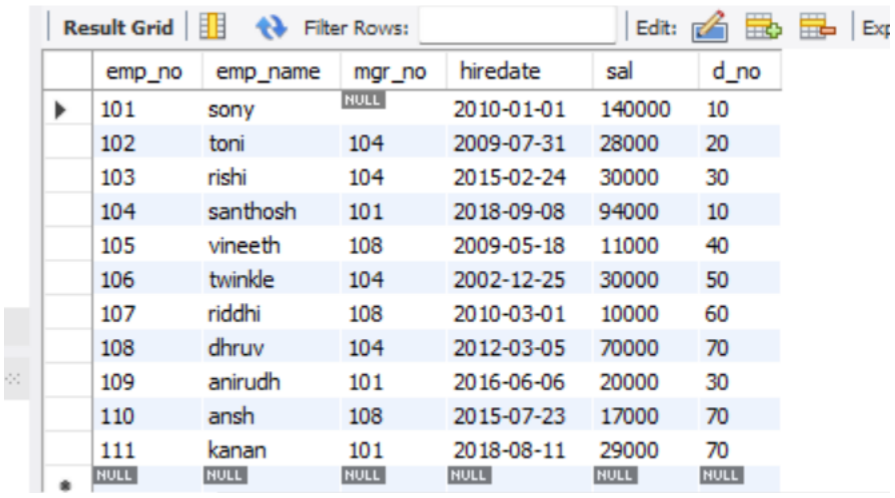
insert into employee values(107,'riddhi',108,'2010-03-01',10000,60);

insert into employee values(108,'dhruv',104,'2012-03-05',70000,70);

insert into employee values(109,'anirudh',101,'2016-06-06',20000,30);

insert into employee values(110,'ansh',108,'2015-07-23',17000,70);

insert into employee values(111,'kanan',101,'2018-08-11',29000,70);



emp_no	emp_name	mgr_no	hiredate	sal	d_no
101	sony	NULL	2010-01-01	140000	10
102	toni	104	2009-07-31	28000	20
103	rishi	104	2015-02-24	30000	30
104	santhosh	101	2018-09-08	94000	10
105	vineeth	108	2009-05-18	11000	40
106	twinkle	104	2002-12-25	30000	50
107	riddhi	108	2010-03-01	10000	60
108	dhruv	104	2012-03-05	70000	70
109	anirudh	101	2016-06-06	20000	30
110	ansh	108	2015-07-23	17000	70
111	kanan	101	2018-08-11	29000	70
NULL	NULL	NULL	NULL	NULL	NULL

incentives:

insert into incentives values(103,'2022-07-12',2500);

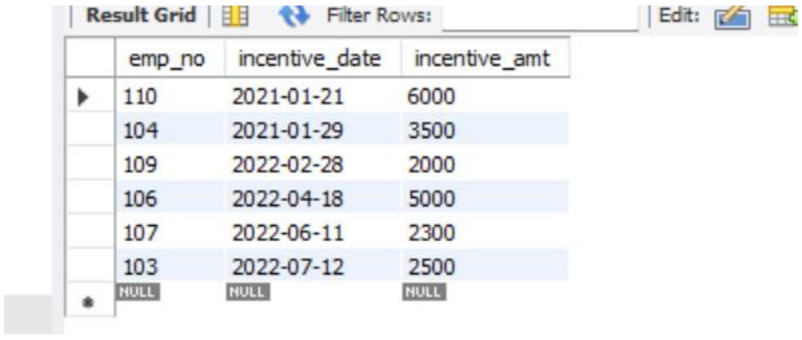
insert into incentives values(104,'2021-01-29',3500);

insert into incentives values(109,'2022-02-28',2000);

insert into incentives values(110,'2021-01-21',6000);

insert into incentives values(106,'2022-04-18',5000);

insert into incentives values(107,'2022-06-11',2300);



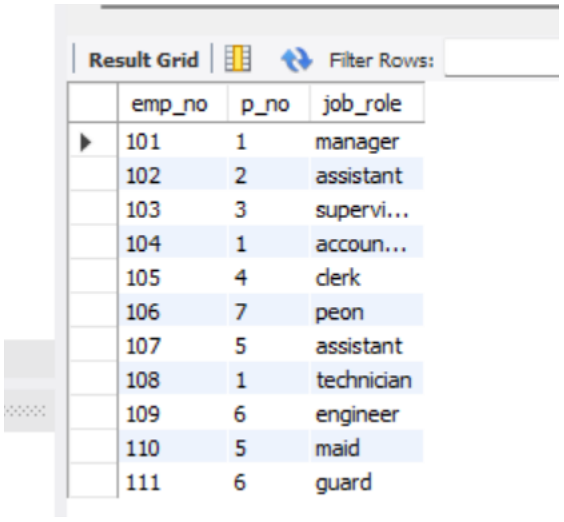
emp_no	incentive_date	incentive_amt
110	2021-01-21	6000
104	2021-01-29	3500
109	2022-02-28	2000
106	2022-04-18	5000
107	2022-06-11	2300
103	2022-07-12	2500
NULL	NULL	NULL

assigned:

insert into assigned values(101,1,'manager');

insert into assigned values(102,2,'assistant');

```
insert into assigned values(103,3,'supervisor');
insert into assigned values(104,1,'accountant');
insert into assigned values(105,4,'clerk');
insert into assigned values(106,7,'peon');
insert into assigned values(107,5,'assistant');
insert into assigned values(108,1,'technician');
insert into assigned values(109,6,'engineer');
insert into assigned values(110,5,'maid');
insert into assigned values(111,6,'guard');
```

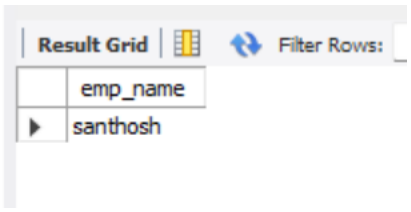


emp_no	p_no	job_role
101	1	manager
102	2	assistant
103	3	supervi...
104	1	accoun...
105	4	clerk
106	7	peon
107	5	assistant
108	1	technician
109	6	engineer
110	5	maid
111	6	guard

Queries

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

SQL> select emp_name from employee where emp_no =(select mgr_no from employee group by mgr_no having count(emp_no)=(select count(emp_no) from employee group by mgr_no order by count(emp_no) desc limit 1));



```
emp_name
santhosh
```

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

SQL> SELECT m.emp_name

FROM employee m

WHERE m.emp_no IN

 (SELECT mgr_no

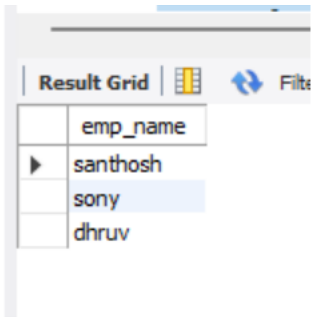
 FROM employee)

 AND m.sal >

 (SELECT avg(e.sal)

 FROM employee e

 WHERE e.mgr_no = m.emp_no);



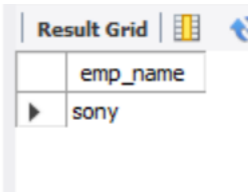
```
emp_name
santhosh
sony
dhruv
```

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

SQL> select emp_name from employee where emp_no in(select distinct mgr_no

from employee where emp_no in (select distinct mgr_no

from employee where emp_no in(select distinct mgr_no from employee)));

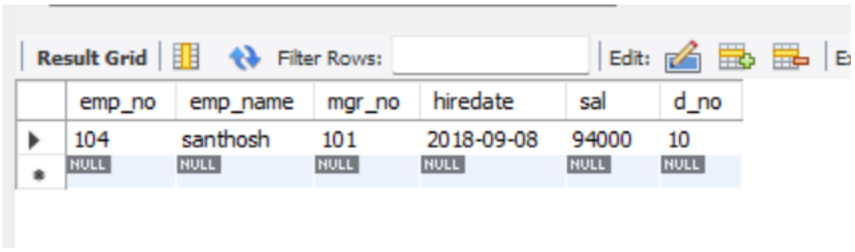


Result Grid

	emp_name
▸	sony

4. Find the employee details who got second maximum incentive in January 2021.

SQL> select * from employee where emp_no =(select emp_no from incentives where incentive_date between '20121-01-01' and '2021-01-31' and incentive_amt!=(select max(incentive_amt) from incentives where incentive_date between '2021-01-01' and '2021-01-31'));

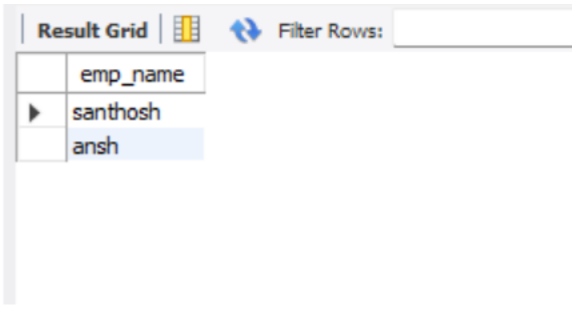


Result Grid | Filter Rows: | Edit:

	emp_no	emp_name	mgr_no	hiredate	sal	d_no
▸	104	santhosh	101	2018-09-08	94000	10
*	NULL	NULL	NULL	NULL	NULL	NULL

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

SQL> select e.emp_name from employee e where e.d_no=(select d_no from employee where e.mgr_no=emp_no);



Result Grid | Filter Rows:

	emp_name
▸	santhosh
	ansh

Supplier Database

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.

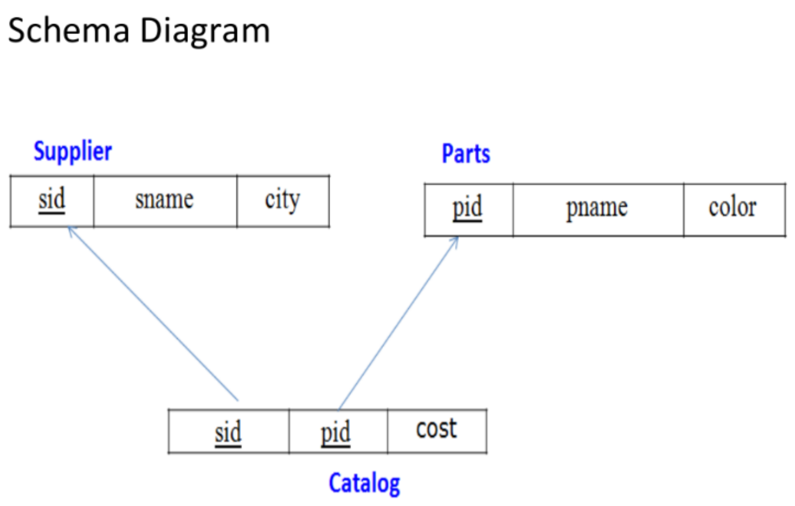
5. Find the snames of suppliers who supply every red part.

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.

Scheme Diagram



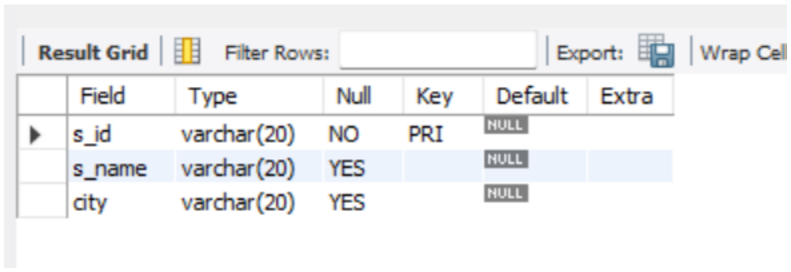
Create Database

```
create database supplier_062;
 use supplier_062;
```

Create Table

supplier:

```
create table supplier(
s_id varchar(20),
s_name varchar(20),
city varchar(20),
primary key(s_id)
);
```

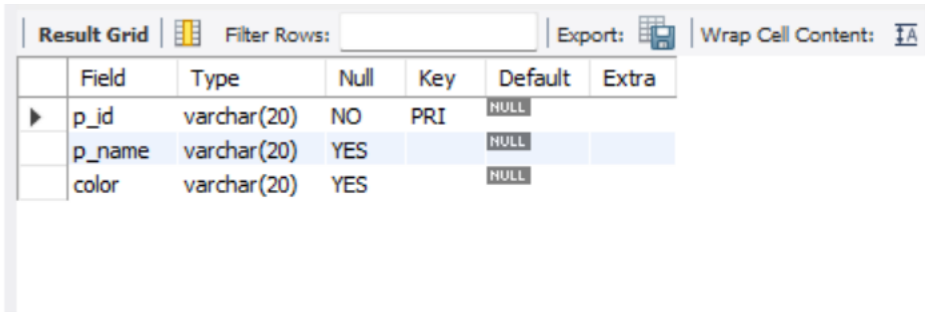


Result Grid | Filter Rows: | Export: | Wrap Cell

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

parts:

```
create table parts(
p_id varchar(20),
p_name varchar(20),
color varchar(20),
primary key(p_id)
);
```



Field	Type	Null	Key	Default	Extra
p_id	varchar(20)	NO	PRI	NULL	
p_name	varchar(20)	YES		NULL	
color	varchar(20)	YES		NULL	

clog:

create table clog(

s_id varchar(20),

p_id varchar(20),

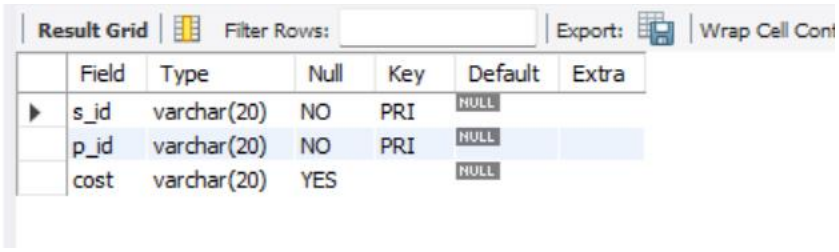
cost varchar(20),

primary key(s_id,p_id),

foreign key(p_id)references parts(p_id),

foreign key(s_id)references supplier(s_id)

);



Field	Type	Null	Key	Default	Extra
s_id	varchar(20)	NO	PRI	NULL	
p_id	varchar(20)	NO	PRI	NULL	
cost	varchar(20)	YES		NULL	

Inserting Values to the table

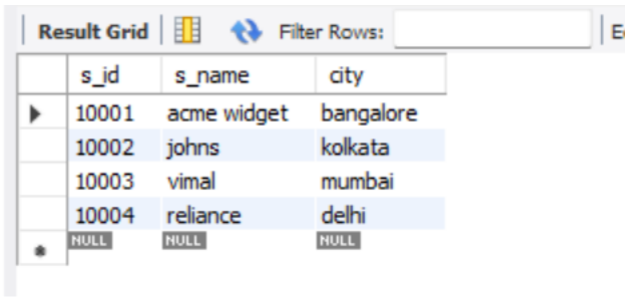
supplier:

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');



s_id	s_name	city
10001	acme widget	bangalore
10002	johns	kolkata
10003	vimal	mumbai
10004	reliance	delhi
NULL	NULL	NULL

parts:

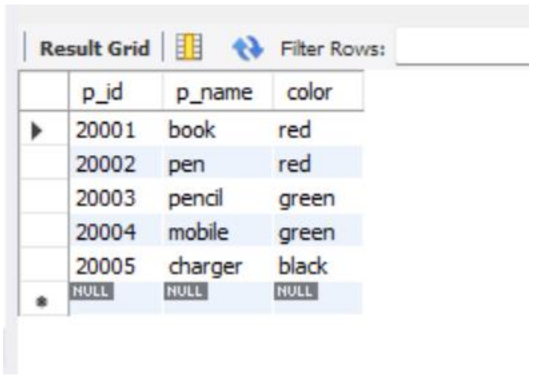
insert into parts values(20001,'book','red');

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');



p_id	p_name	color
20001	book	red
20002	pen	red
20003	pencil	green
20004	mobile	green
20005	charger	black
NULL	NULL	NULL

clog:

insert into clog values(10001,20001,10);

insert into clog values(10001,20002,10);

insert into clog values(10001,20003,30);

insert into clog values(10001,20004,10);

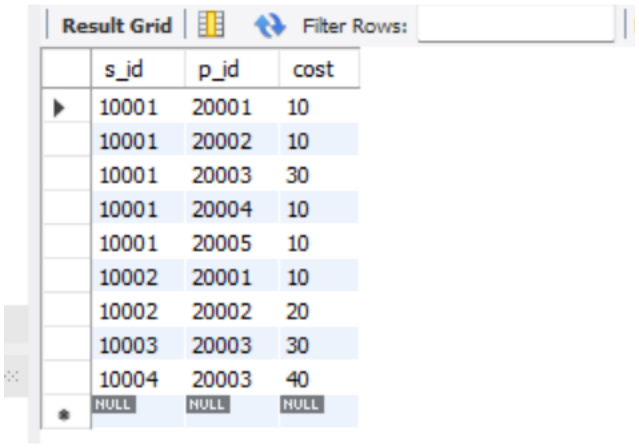
insert into clog values(10001,20005,10);

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

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

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

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



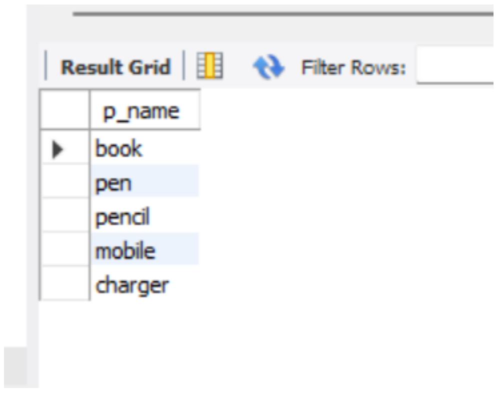
s_id	p_id	cost
10001	20001	10
10001	20002	10
10001	20003	30
10001	20004	10
10001	20005	10
10002	20001	10
10002	20002	20
10003	20003	30
10004	20003	40
NULL	NULL	NULL

Queries:

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

SQL>

select p_name from parts where p_id IN (select p_id from clog);

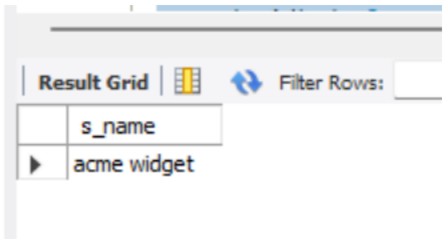


p_name
book
pen
pencil
mobile
charger

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

SQL>

select s_name from supplier where s_id in(select s_id from clog group by s_id having count(p_id)=(select count(p_id) from parts));

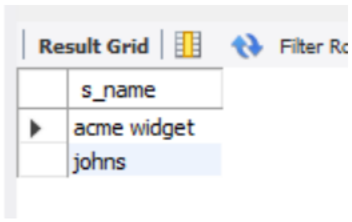


```
s_name
acme widget
```

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

SQL>

select s_name from supplier where s_id in (select s_id from clog where p_id in(select p_id from parts where color='red'));

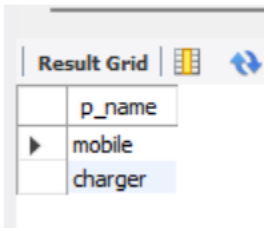


```
s_name
acme widget
johns
```

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

SQL>

select p_name from parts where p_id not in (select a.p_id from clog a,clog b where a.p_id=b.p_id and a.s_id!=b.s_id);

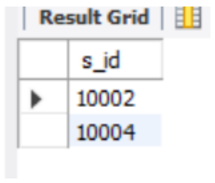


```
p_name
mobile
charger
```

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

SQL>

select a.s_id from clog a where cost>(select avg(b.cost) from clog b where b.p_id=a.p_id group by b.p_id);



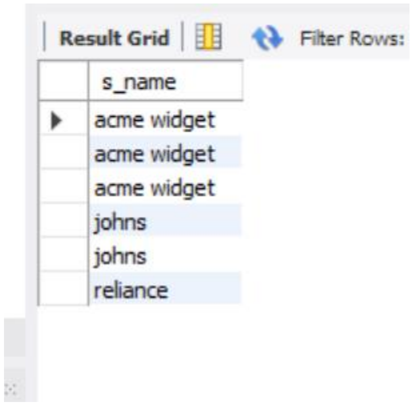
Result Grid

```
s_id
10002
10004
```

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

SQL>

select s.s_name from supplier s,clog a where s.s_id=a.s_id and a.cost=
(select max(cost) from clog where a.p_id=p_id group by p_id);



Result Grid | Filter Rows:

```
s_name
acme widget
acme widget
acme widget
johns
johns
reliance
```

Flight Database

Question

FLIGHTS(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time,

price: integer)

AIRCRAFT(aid: integer, aname: string, cruising_range: integer)

CERTIFIED(eid: integer, aid: integer)

EMPLOYEES(eid: integer, ename: string, salary: integer)

Note that the Employees relation describes pilots and other kinds of employees as well; Every pilot is certified for some aircraft, and only pilots are certified to fly.

Create database table and insert appropriate data

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 eid and the maximum cruisingrange of the aircraft for which she or he is certified.

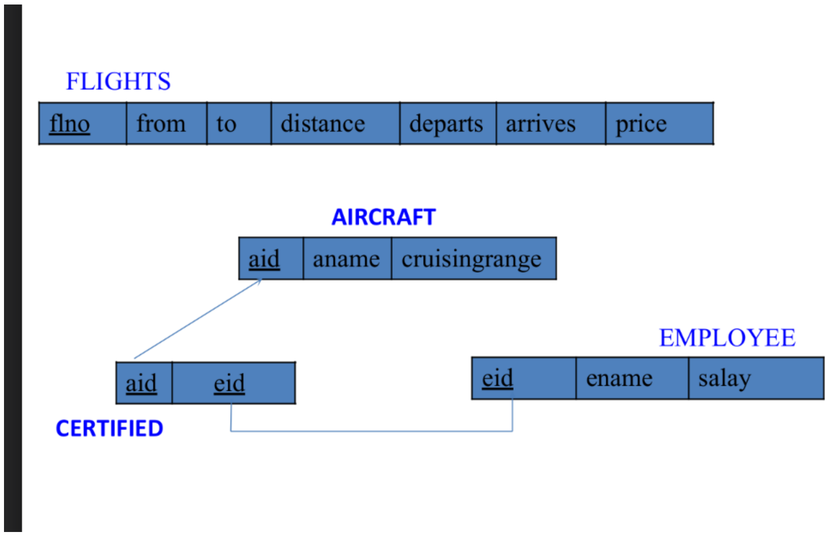
iii. Find the names of pilots whose salary is less than the price of the cheapest route fromBengaluru to Frankfurt.

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.

Schema Diagram



Create Database

create database flight_1bm21cs062;

use flight_1bm21cs062;

Creating Table

flights:

create table flights(

flno int,

from_place varchar(20),

to_place varchar(20),

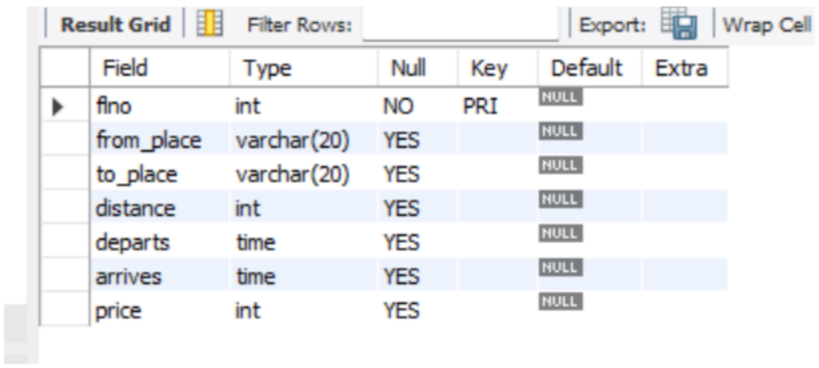
distance int,

departs time,

arrives time,

price int,

PRIMARY KEY(flno));


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

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	

aircraft:

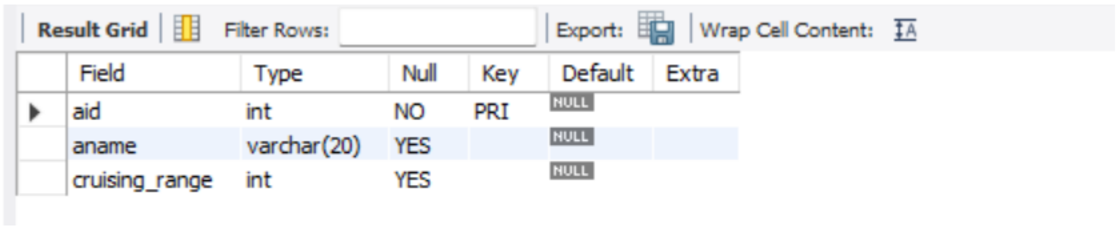
create table aircraft(

aid int,

aname varchar(20),

cruising_range int,

PRIMARY KEY(aid));


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

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

employee:

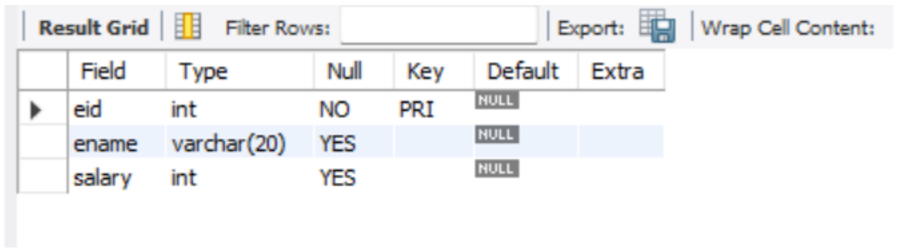
create table employee(

eid int,

ename varchar(20),

salary int,

PRIMARY KEY(eid));


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

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

certified:

create table certified(

eid int,

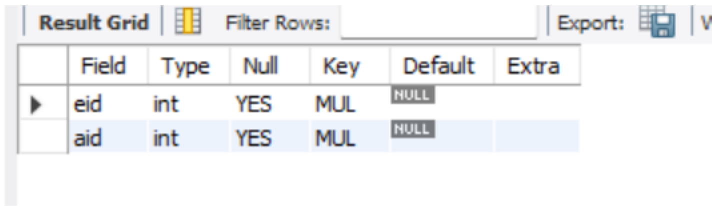
aid int,

FOREIGN KEY(eid) REFERENCES employee(eid)

on update cascade on delete cascade,

FOREIGN KEY(aid) REFERENCES aircraft(aid)

on update cascade on delete cascade);


Result Grid | Filter Rows: | Export:

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

Inserting Values to the table

employee:

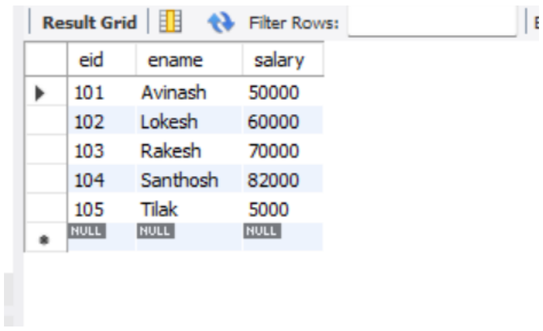
insert into employee values(101,'Avinash',50000);

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);



eid	ename	salary
101	Avinash	50000
102	Lokesh	60000
103	Rakesh	70000
104	Santhosh	82000
105	Tilak	5000
NULL	NULL	NULL

aircraft:

insert into aircraft values(1,'Airbus',2000);

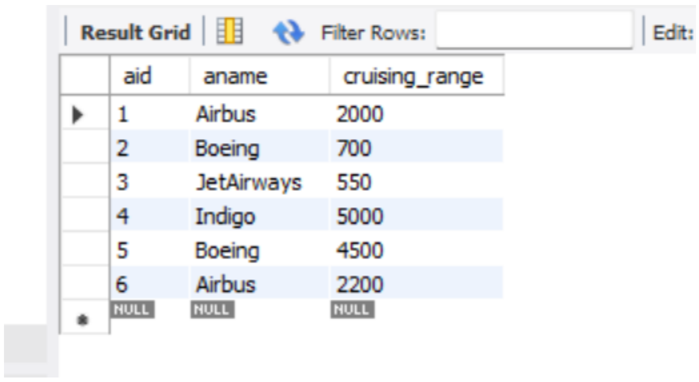
insert into aircraft values(2,'Boeing',700);

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);



aid	aname	cruising_range
1	Airbus	2000
2	Boeing	700
3	JetAirways	550
4	Indigo	5000
5	Boeing	4500
6	Airbus	2200
NULL	NULL	NULL

certified:

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);

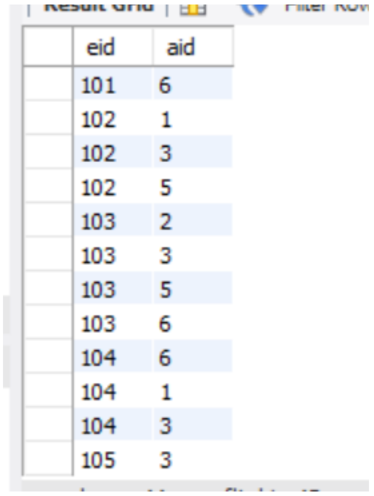
insert into certified values(103,6);

insert into certified values(104,6);

insert into certified values(104,1);

insert into certified values(104,3);

insert into certified values(105,3);



eid	aid
101	6
102	1
102	3
102	5
103	2
103	3
103	5
103	6
104	6
104	1
104	3
105	3

flights:

insert into flights values(1,'Bangalore','New Delhi',500,'06:00','09:00',5000);

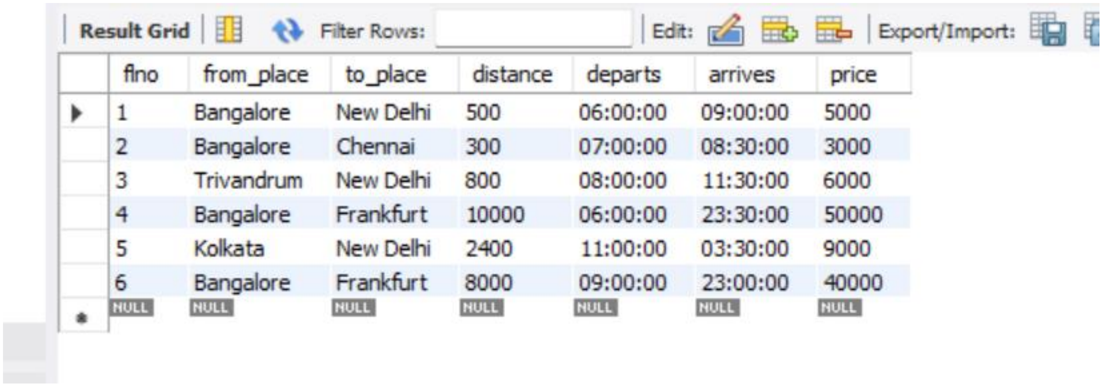
insert into flights values(2,'Bangalore','Chennai',300,'07:00','08:30',3000);

insert into flights values(3,'Trivandrum','New Delhi',800,'08:00','11:30',6000);

insert into flights values(4,'Bangalore','Frankfurt',10000,'06:00','23:30',50000);

insert into flights values(5,'Kolkata','New Delhi',2400,'11:00','03:30',9000);

insert into flights values(6,'Bangalore','Frankfurt',8000,'09:00','23:00',40000);



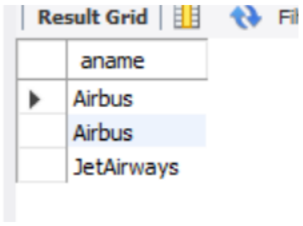
fno	from_place	to_place	distance	departs	arrives	price
1	Bangalore	New Delhi	500	06:00:00	09:00:00	5000
2	Bangalore	Chennai	300	07:00:00	08:30:00	3000
3	Trivandrum	New Delhi	800	08:00:00	11:30:00	6000
4	Bangalore	Frankfurt	10000	06:00:00	23:30:00	50000
5	Kolkata	New Delhi	2400	11:00:00	03:30:00	9000
6	Bangalore	Frankfurt	8000	09:00:00	23:00:00	40000
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Queries

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

SQL>

select a.aname from employee e,aircraft a,certified c where a.aid=c.aid and c.eid=e.eid and e.salary>80000;

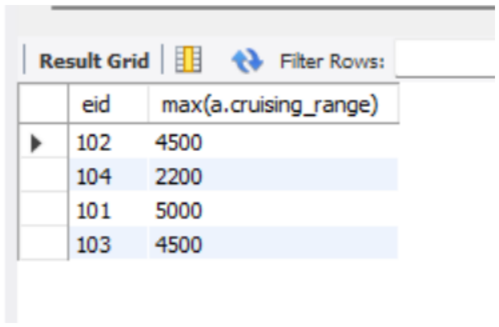


aname
Airbus
Airbus
JetAirways

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

SQL>

select c.eid, max(a.cruising_range) from aircraft a, certified c where c.aid=a.aid group by c.eid having count(*)>=3;

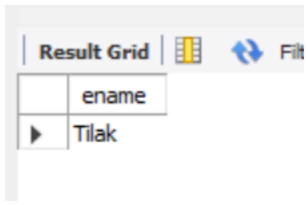

Result Grid | Filter Rows:

eid	max(a.cruising_range)
102	4500
104	2200
101	5000
103	4500

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

SQL>

select e.ename from employee e where e.salary<(select min(price) from flights where from_place='Bangalore' and to_place='Frankfurt');

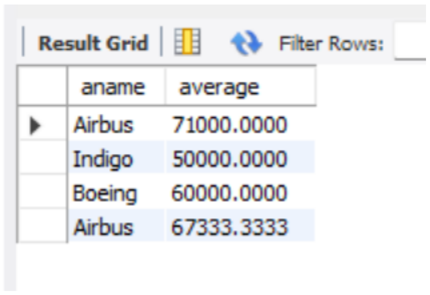

Result Grid | Filt

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.

SQL>

select a.aname, avg(e.salary) as average from certified c inner join aircraft a on c.aid=a.aid and a.cruising_range>1000 inner join employee e on e.eid=c.eid group by c.aid;

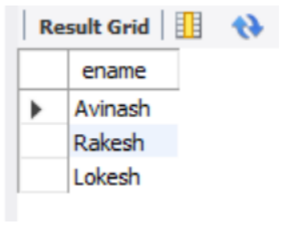

Result Grid | Filter Rows:

aname	average
Airbus	71000.0000
Indigo	50000.0000
Boeing	60000.0000
Airbus	67333.3333

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

SQL>

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

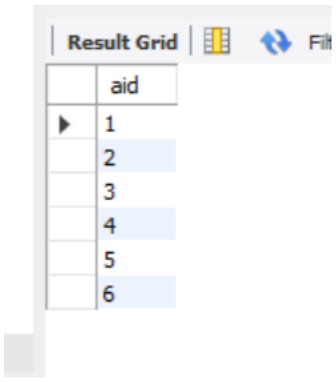


```
ename
Avinash
Rakesh
Lokesh
```

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

SQL>

select a.aid from aircraft a ,flights f where from_place='Bangalore' and to_place='New Delhi' and a.cruising_range>f.distance



```
aid
1
2
3
4
5
6
```

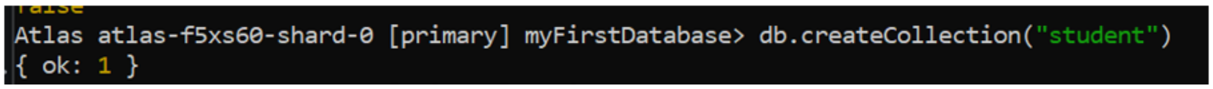
NoSQL

PART 1

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 rollno10.
4. Replace the student name from "ABC" to "FEM" of rollno11.
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");



```
Atlas atlas-f5xs60-shard-0 [primary] myFirstDatabase> db.createCollection("student")
{ ok: 1 }
```

Insert Values

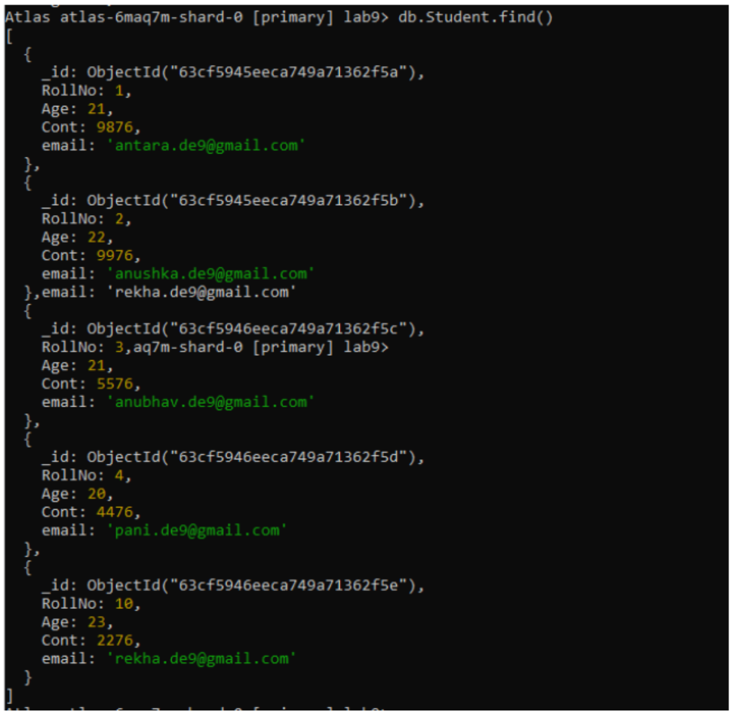
db.Student.insert({RollNo:1,Age:21,Cont:9876,email:"antara.de9@gmail.com"});

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

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

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

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



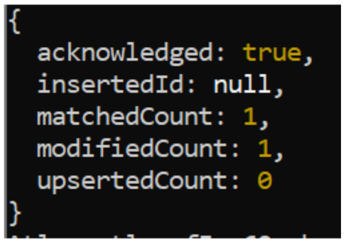
```
Atlas atlas-6maq7m-shard-0 [primary] lab9> db.Student.find()
[
  {
    _id: ObjectId("63cf5945eeca749a71362f5a"),
    RollNo: 1,
    Age: 21,
    Cont: 9876,
    email: 'antara.de9@gmail.com'
  },
  {
    _id: ObjectId("63cf5945eeca749a71362f5b"),
    RollNo: 2,
    Age: 22,
    Cont: 9976,
    email: 'anushka.de9@gmail.com'
  },email: 'rekha.de9@gmail.com'
  {
    _id: ObjectId("63cf5946eeca749a71362f5c"),
    RollNo: 3,aq7m-shard-0 [primary] lab9>
    Age: 21,
    Cont: 5576,
    email: 'anubhav.de9@gmail.com'
  },
  {
    _id: ObjectId("63cf5946eeca749a71362f5d"),
    RollNo: 4,
    Age: 20,
    Cont: 4476,
    email: 'pani.de9@gmail.com'
  },
  {
    _id: ObjectId("63cf5946eeca749a71362f5e"),
    RollNo: 10,
    Age: 23,
    Cont: 2276,
    email: 'rekha.de9@gmail.com'
  }
]
```

Queries

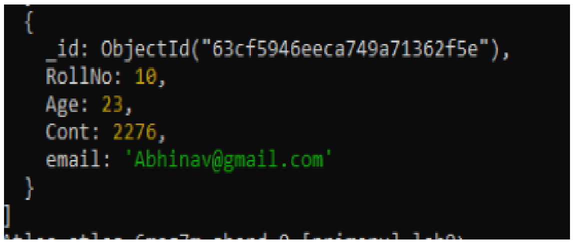
1. Write a query to update the Email-Id of a student with rollno10.

db.Student.update({RollNo:10},{$set:{

email:"Abhinav@gmail.com"}})



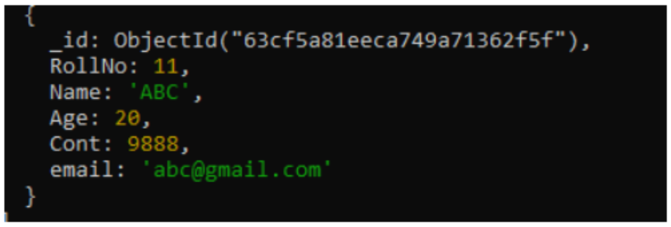
```
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```



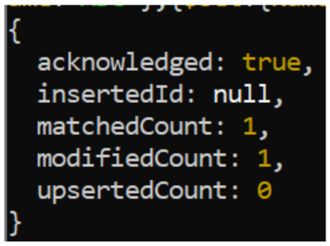
```
  {
    _id: ObjectId("63cf5946eeca749a71362f5e"),
    RollNo: 10,
    Age: 23,
    Cont: 2276,
    email: 'Abhinav@gmail.com'
  }
]
```

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

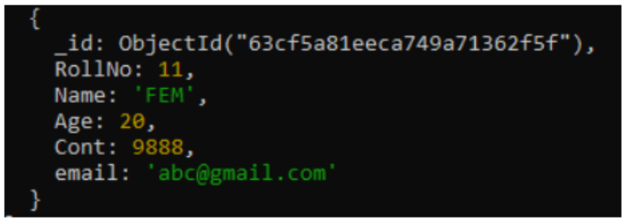
db.Student.update({RollNo:11,Name:"ABC"},{$set:{Name:"FEM"}})



```
{
  _id: ObjectId("63cf5a81eeca749a71362f5f"),
  RollNo: 11,
  Name: 'ABC',
  Age: 20,
  Cont: 9888,
  email: 'abc@gmail.com'
}
```



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



```
{
  _id: ObjectId("63cf5a81eeca749a71362f5f"),
  RollNo: 11,
  Name: 'FEM',
  Age: 20,
  Cont: 9888,
  email: 'abc@gmail.com'
}
```

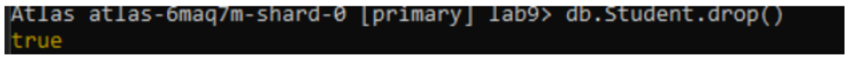
3.. Export the created table into local file system

mongodb+srv://sanj:sanj257 @cluster0.mfnfeys.mongodb.net/lab9 --

collection=customer --out C:\Users\s\Desktop\Downloads\output.json

4. Drop the table

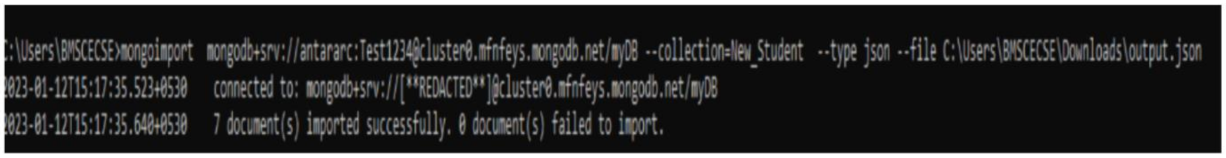
db.Student.drop();



```
Atlas atlas-6maq7m-shard-0 [primary] lab9> db.Student.drop()
true
```

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

mongoimport



```
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.
```

NoSQL

Part 2

1. Create a collection by name Customers with the following

attributes.Cust_id, Acc_Bal, Acc_Type

2. Insert at least 5 values into the table

3. Write a query to display those records whose total account

balance

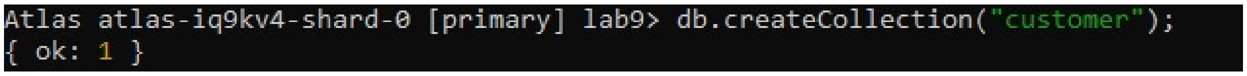
is greater than 1200 of account type ‘Z’ for each customer_id.

4. Determine Minimum and Maximum account balance for each

customer_id.

Create Database

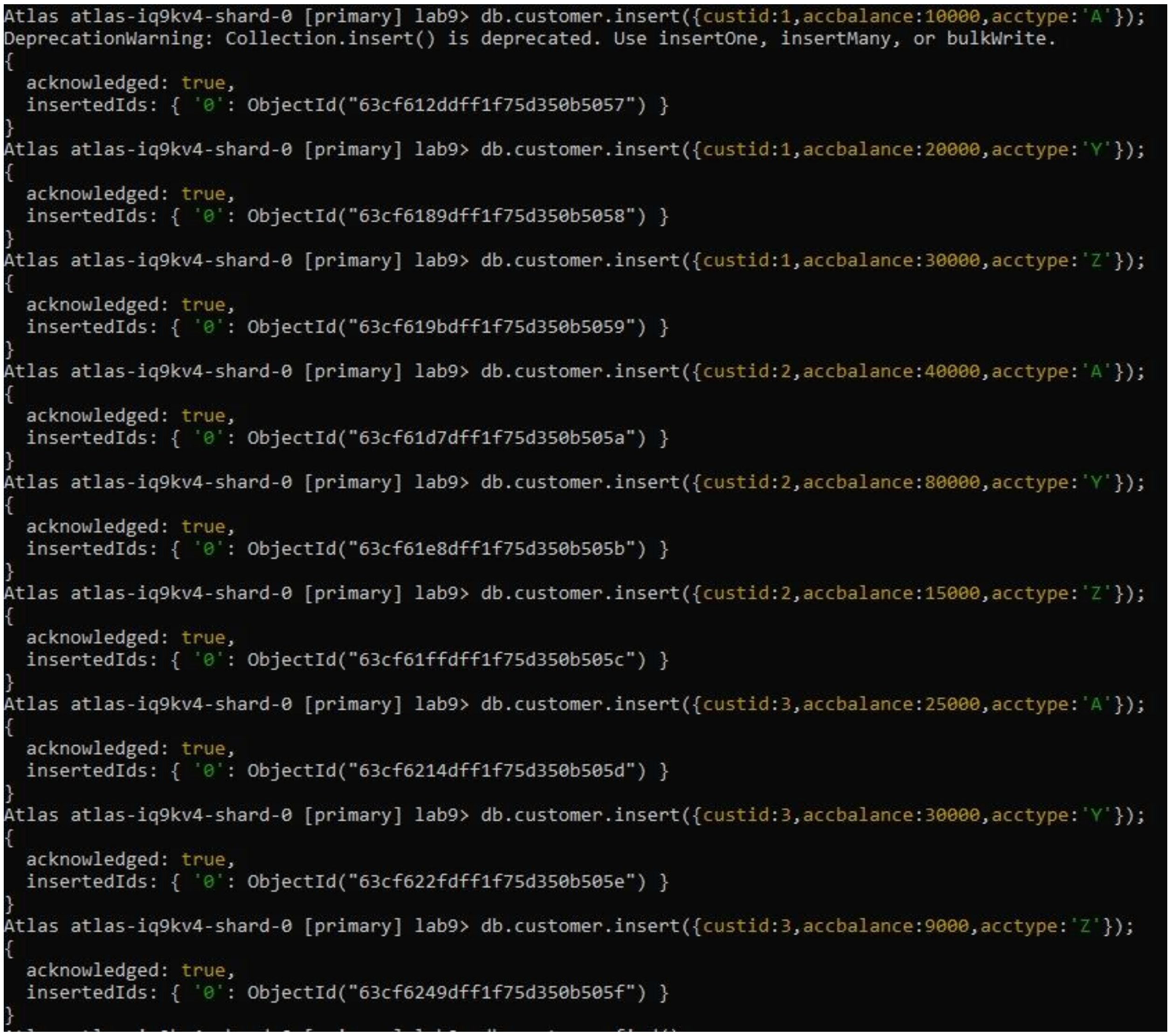
db.createCollection("Customers");



```
Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.createCollection("customer");
{ ok: 1 }
```

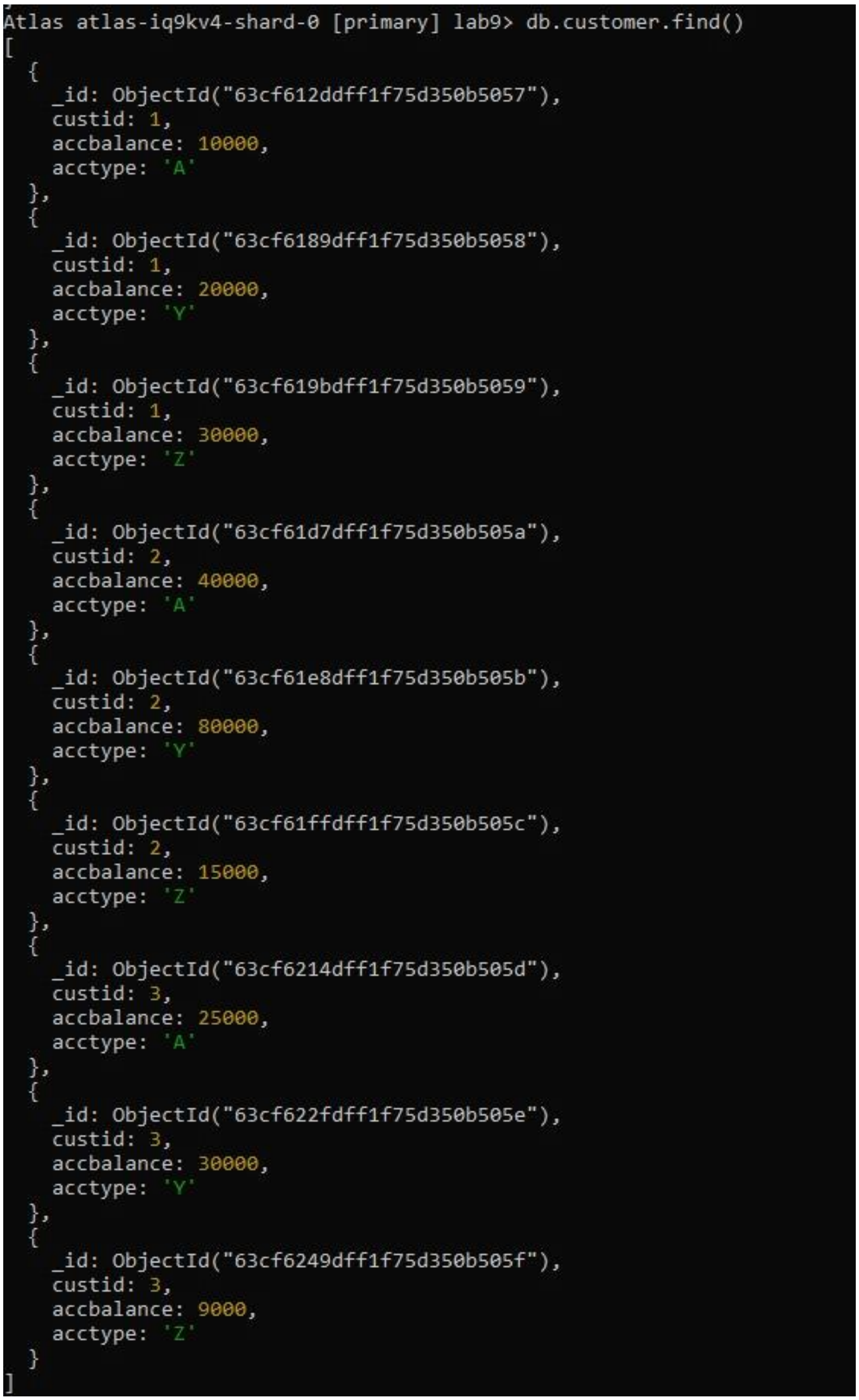
Insert Values

```
db.customer.insert({custid:1,accbalance:10000,acctype:’A’});
db.customer.insert({custid:1,accbalance:20000,acctype:’Y’});
db.customer.insert({custid:1,accbalance:30000,acctype:’Z’});
db.customer.insert({custid:2,accbalance:40000,acctype:’A’});
db.customer.insert({custid:2,accbalance:80000,acctype:’A’});
db.customer.insert({custid:2,accbalance:15000,acctype:’A’});
db.customer.insert({custid:3,accbalance:25000,acctype:’A’});
db.customer.insert({custid:3,accbalance:30000,acctype:’A’});
db.customer.insert({custid:3,accbalance:9000,acctype:’A’});
```



```
Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.customer.insert({custid:1,accbalance:10000,acctype:'A'});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cf612ddff1f75d350b5057") }
}
Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.customer.insert({custid:1,accbalance:20000,acctype:'Y'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cf6189dff1f75d350b5058") }
}
Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.customer.insert({custid:1,accbalance:30000,acctype:'Z'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cf619bdff1f75d350b5059") }
}
Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.customer.insert({custid:2,accbalance:40000,acctype:'A'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cf61d7dff1f75d350b505a") }
}
Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.customer.insert({custid:2,accbalance:80000,acctype:'Y'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cf61e8dff1f75d350b505b") }
}
Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.customer.insert({custid:2,accbalance:15000,acctype:'Z'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cf61ffdff1f75d350b505c") }
}
Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.customer.insert({custid:3,accbalance:25000,acctype:'A'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cf6214dff1f75d350b505d") }
}
Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.customer.insert({custid:3,accbalance:30000,acctype:'Y'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cf622fdff1f75d350b505e") }
}
Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.customer.insert({custid:3,accbalance:9000,acctype:'Z'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cf6249dff1f75d350b505f") }
}
```

]

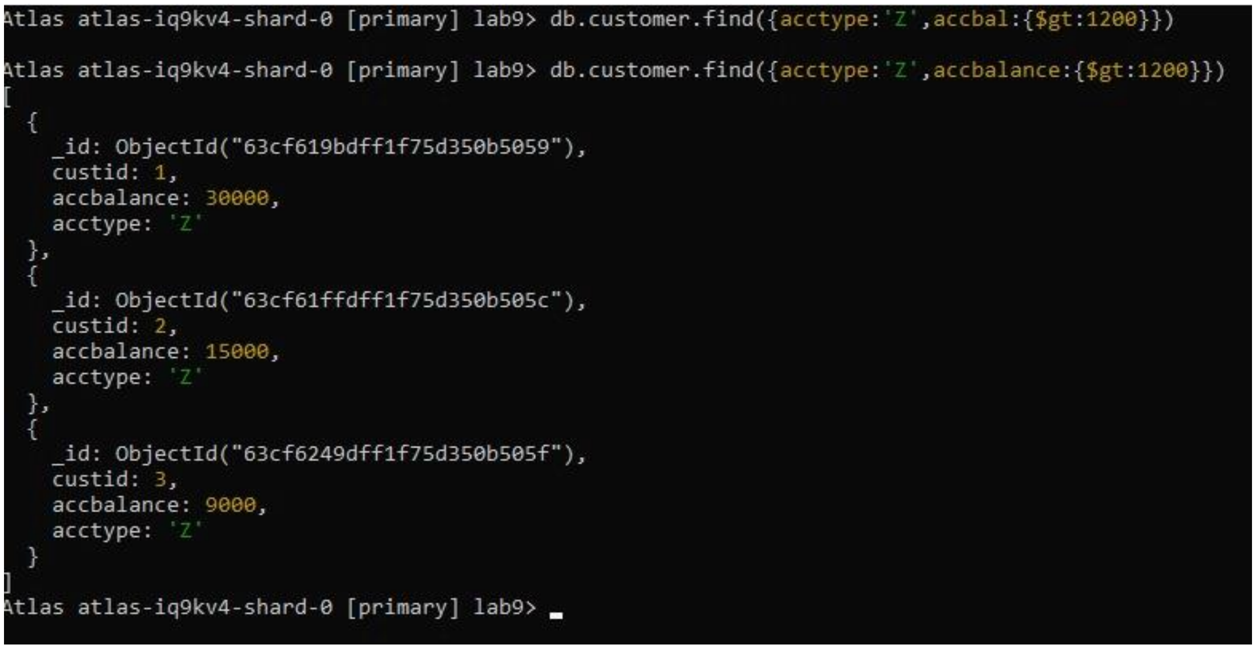


```
Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.customer.find()
[
  {
    _id: ObjectId("63cf612ddff1f75d350b5057"),
    custid: 1,
    accbalance: 10000,
    acctype: 'A'
  },
  {
    _id: ObjectId("63cf6189dff1f75d350b5058"),
    custid: 1,
    accbalance: 20000,
    acctype: 'Y'
  },
  {
    _id: ObjectId("63cf619bdff1f75d350b5059"),
    custid: 1,
    accbalance: 30000,
    acctype: 'Z'
  },
  {
    _id: ObjectId("63cf61d7dff1f75d350b505a"),
    custid: 2,
    accbalance: 40000,
    acctype: 'A'
  },
  {
    _id: ObjectId("63cf61e8dff1f75d350b505b"),
    custid: 2,
    accbalance: 80000,
    acctype: 'Y'
  },
  {
    _id: ObjectId("63cf61ffdff1f75d350b505c"),
    custid: 2,
    accbalance: 15000,
    acctype: 'Z'
  },
  {
    _id: ObjectId("63cf6214dff1f75d350b505d"),
    custid: 3,
    accbalance: 25000,
    acctype: 'A'
  },
  {
    _id: ObjectId("63cf622fdff1f75d350b505e"),
    custid: 3,
    accbalance: 30000,
    acctype: 'Y'
  },
  {
    _id: ObjectId("63cf6249dff1f75d350b505f"),
    custid: 3,
    accbalance: 9000,
    acctype: 'Z'
  }
]
```

Queries

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

db.customer.find({acctype:’Z’,accbalance:{$gt:1200}})



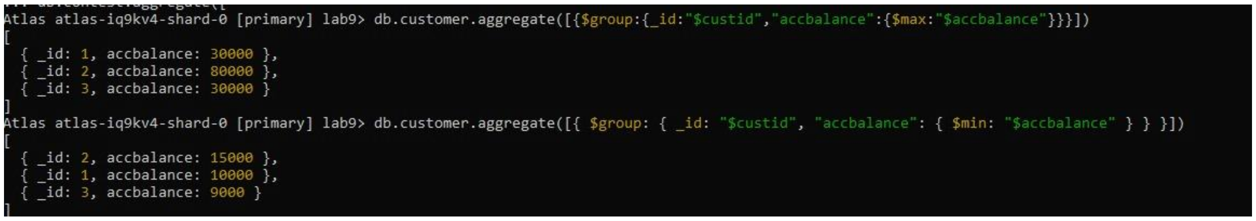
```
Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.customer.find({acctype:'Z',accbal:{$gt:1200}})

Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.customer.find({acctype:'Z',accbalance:{$gt:1200}})
[
  {
    _id: ObjectId("63cf619bdff1f75d350b5059"),
    custid: 1,
    accbalance: 30000,
    acctype: 'Z'
  },
  {
    _id: ObjectId("63cf61ffdff1f75d350b505c"),
    custid: 2,
    accbalance: 15000,
    acctype: 'Z'
  },
  {
    _id: ObjectId("63cf6249dff1f75d350b505f"),
    custid: 3,
    accbalance: 9000,
    acctype: 'Z'
  }
]
Atlas atlas-iq9kv4-shard-0 [primary] lab9> _
```

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

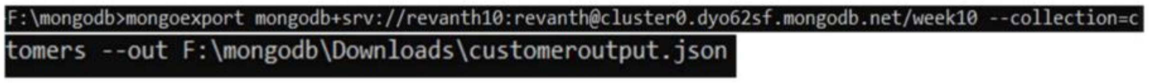
db.customer.aggregate([{$group:{_id:”$custid”,”accbalance”:{$max:”accbalance”}}}])

db.customer.aggregate([{$group:{_id:”$custid”,”accbalance”:{$min:”accbalance”}}}])

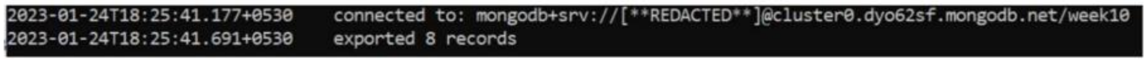


```
Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.customer.aggregate([{$group:{_id:"$custid","accbalance":{$max:"$accbalance"}}}])
[
  { _id: 1, accbalance: 30000 },
  { _id: 2, accbalance: 80000 },
  { _id: 3, accbalance: 30000 }
]
Atlas atlas-iq9kv4-shard-0 [primary] lab9> db.customer.aggregate([{ $group: { _id: "$custid", "accbalance": { $min: "$accbalance" } } }])
[
  { _id: 2, accbalance: 15000 },
  { _id: 1, accbalance: 10000 },
  { _id: 3, accbalance: 9000 }
]
```

3. Export the created collection into local file system



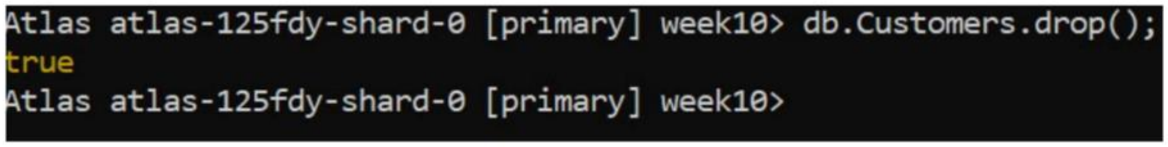
```
F:\mongodb>mongoexport mongodb+srv://revanth10:revanth@cluster0.dyo62sf.mongodb.net/week10 --collection=c
tomers --out F:\mongodb\Downloads\customeroutput.json
```



```
2023-01-24T18:25:41.177+0530    connected to: mongodb+srv://[**REDACTED**]@cluster0.dyo62sf.mongodb.net/week10
2023-01-24T18:25:41.691+0530    exported 8 records
```

4. Drop the table

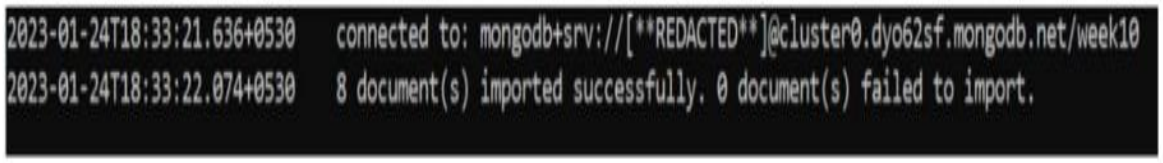
db.customer.drop();



```
Atlas atlas-125fdy-shard-0 [primary] week10> db.Customers.drop();
true
Atlas atlas-125fdy-shard-0 [primary] week10>
```

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

collection.



```
2023-01-24T18:33:21.636+0530    connected to: mongodb+srv://[**REDACTED**]@cluster0.dyo62sf.mongodb.net/week10
2023-01-24T18:33:22.074+0530    8 document(s) imported successfully. 0 document(s) failed to import.
```