VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Database Management Systems (22CS3PCDBM)

Submitted by

DHIKSHA RATHIS (1BM21CS055)

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 **DHIKSHA RATHIS(1BM21CS055),** 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.Pallavi G.B	Dr. Jyothi S Nayak
Assistant Professor	Professor and Head
Department of CSE	Department of CSE
BMSCE, Bengaluru	BMSCE, Bengaluru

`

Index

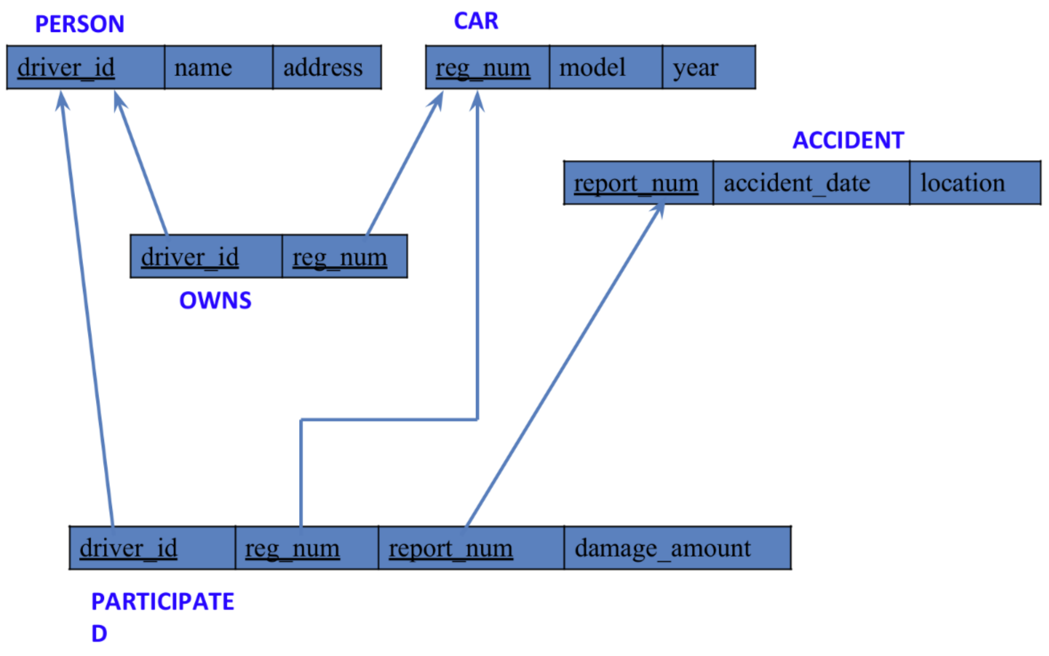
Insurance Database

Question

(Week 1)

- PERSON (driver_id: String, name: String, address: String)
- CAR (reg_num: String, model: String, year: int)
- ACCIDENT (report_num: int, accident_date: date, location: String)
- OWNS (driver_id: String, reg_num: String)
- PARTICIPATED (driver_id: String,reg_num: String, report_num: int, damage_amount: int)
- Create the above tables by properly specifying the primary keys and the foreign keys.
- Enter at least five tuples for each relation
- Display Accident date and location
- Update the damage amount to 25000 for the car with a specific reg_num (example 'K A053408') for which the accident report number was 12.
- Add a new accident to the database.
- To Do
- Display Accident date and location
- Display driver id who did accident with damage amount greater than or equal to Rs.25000

Schema Diagram



Create database

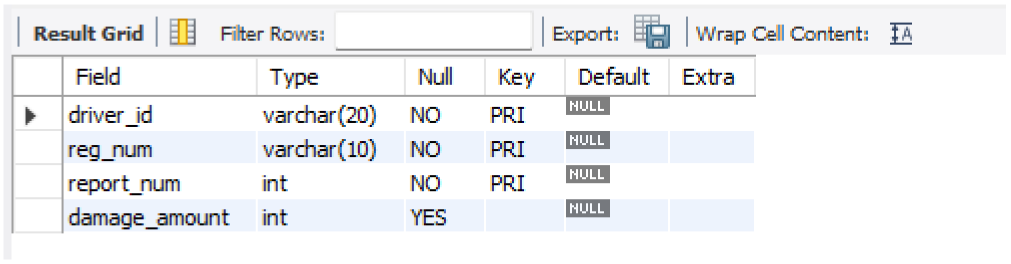
```
create database insurance_dhiksha;
use insurance_dhiksha;
```

Create table

```
create table insurance_dhiksha.person(
driver_id varchar(20),
name varchar(30),
address  varchar(50),
PRIMARY KEY(driver_id)
);
create table insurance_dhiksha.car(
reg_num varchar(15),
model varchar(10),
year int,
PRIMARY KEY(reg_num)
);
create table insurance_dhiksha.owns(
driver_id varchar(20),
reg_num varchar(10),
PRIMARY KEY(driver_id, reg_num),
FOREIGN KEY(driver_id) REFERENCES person(driver_id),
FOREIGN KEY(reg_num) REFERENCES car(reg_num)
);
create table insurance_dhiksha.accident(
report_num int,
accident_date date,
location varchar(50),
PRIMARY KEY(report_num)
);
create table insurance_dhiksha.participated(
driver_id varchar(20),
reg_num varchar(10),
report_num int,
damage_amount int,
PRIMARY KEY(driver_id,reg_num,report_num),
FOREIGN KEY(driver_id) REFERENCES person(driver_id),
FOREIGN KEY(reg_num) REFERENCES car(reg_num),
FOREIGN KEY(report_num) REFERENCES accident(report_num)
);
```

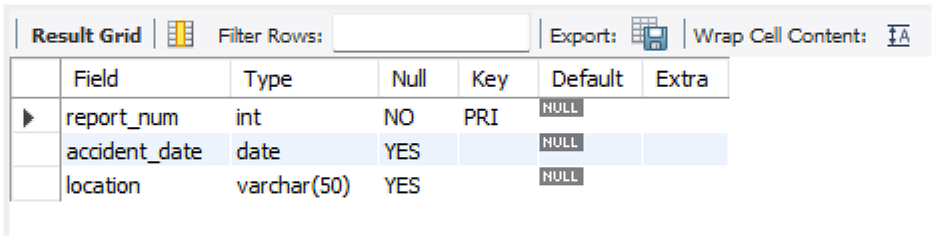
Structure of the table

desc person;



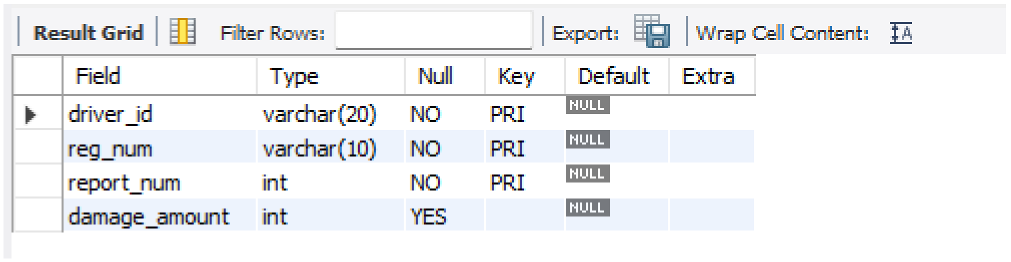
Field	Type	Null	Key	Default	Extra
driver_id	varchar(20)	NO	PRI	NULL	
reg_num	varchar(10)	NO	PRI	NULL	
report_num	int	NO	PRI	NULL	
damage_amount	int	YES		NULL	

desc accident;



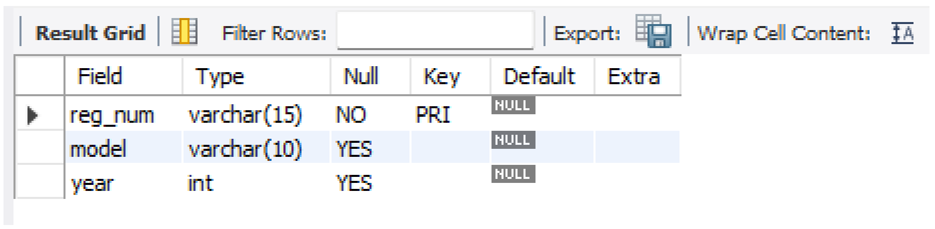
Field	Type	Null	Key	Default	Extra
report_num	int	NO	PRI	NULL	
accident_date	date	YES		NULL	
location	varchar(50)	YES		NULL	

desc participated;



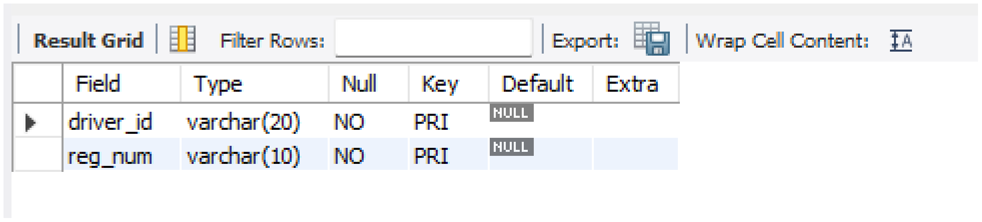
Field	Type	Null	Key	Default	Extra
driver_id	varchar(20)	NO	PRI	NULL	
reg_num	varchar(10)	NO	PRI	NULL	
report_num	int	NO	PRI	NULL	
damage_amount	int	YES		NULL	

desc car;



Field	Type	Null	Key	Default	Extra
reg_num	varchar(15)	NO	PRI	NULL	
model	varchar(10)	YES		NULL	
year	int	YES		NULL	

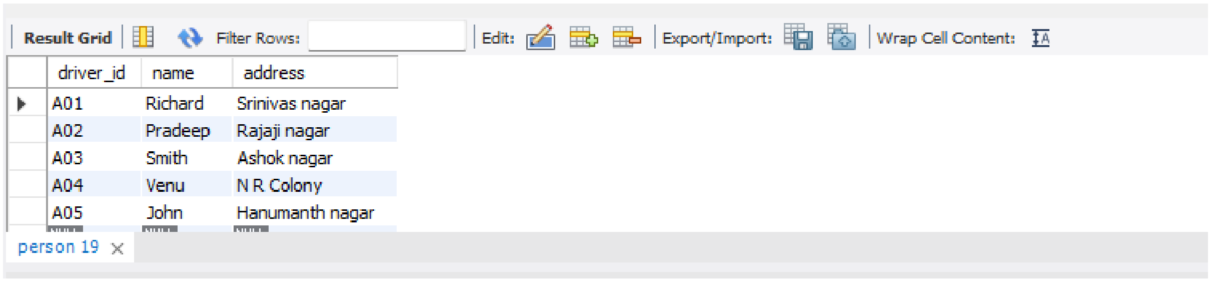
desc owns;



Field	Type	Null	Key	Default	Extra
driver_id	varchar(20)	NO	PRI	NULL	
reg_num	varchar(10)	NO	PRI	NULL	

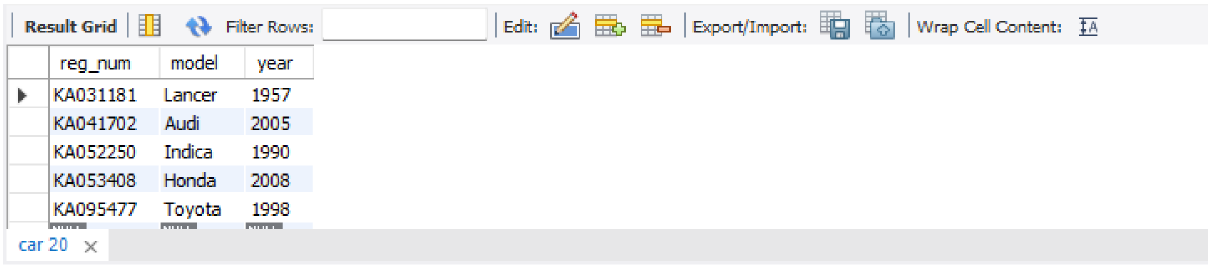
Inserting Values to the table

```
insert into person values("A01","Richard", "Srinivas nagar");
insert into person values("A02","Pradeep", "Rajaji nagar");
insert into person values("A03","Smith", "Ashok nagar");
insert into person values("A04","Venu", "N R Colony");
insert into person values("A05","John", "Hanumanth nagar");
select * from person;
```



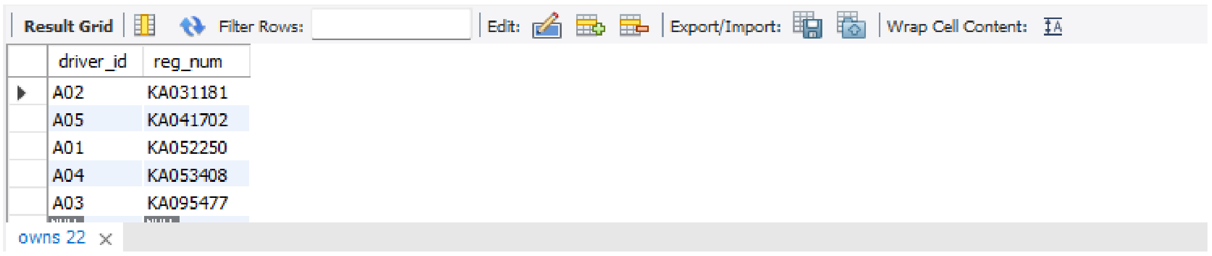
driver_id	name	address
A01	Richard	Srinivas nagar
A02	Pradeep	Rajaji nagar
A03	Smith	Ashok nagar
A04	Venu	N R Colony
A05	John	Hanumanth nagar

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



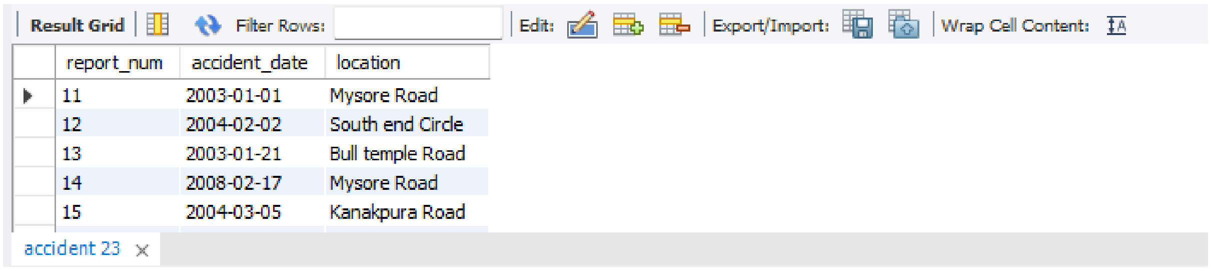
reg_num	model	year
KA031181	Lancer	1957
KA041702	Audi	2005
KA052250	Indica	1990
KA053408	Honda	2008
KA095477	Toyota	1998

```
insert into owns values("A01","KA052250");

insert into owns values("A02","KA031181");

insert into owns values("A03","KA095477");

insert into owns values("A04","KA053408");

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

select * from owns;
```



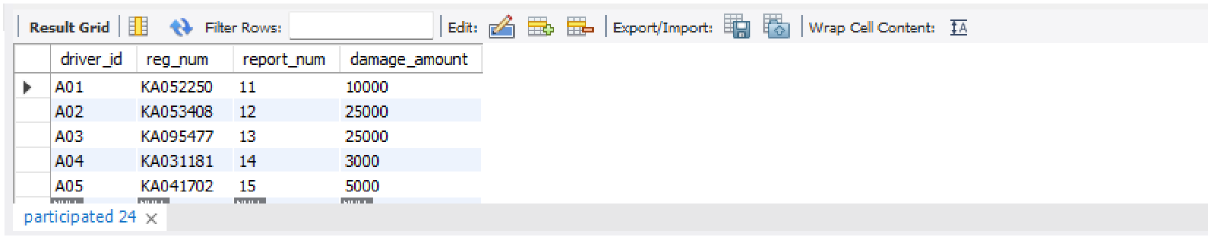
driver_id	reg_num
A02	KA031181
A05	KA041702
A01	KA052250
A04	KA053408
A03	KA095477

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

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

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

insert into accident values(14,'2008-02-17',"Mysore Road");

insert into accident values(15,'2004-03-05',"Kanakpura Road");

select * from accident;
```



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

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

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

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

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

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

select * from participated;
```

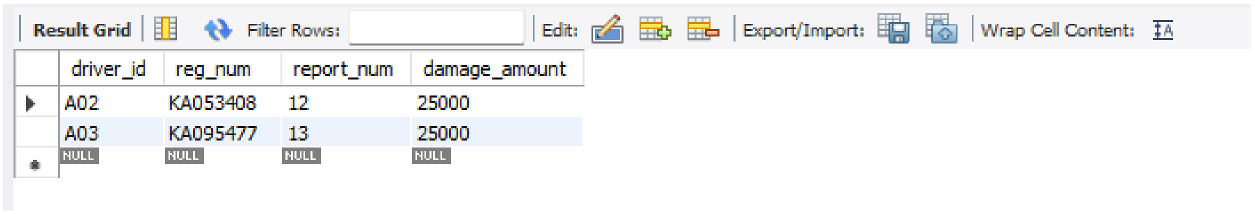


driver_id	reg_num	report_num	damage_amount
A01	KA052250	11	10000
A02	KA053408	12	25000
A03	KA095477	13	25000
A04	KA031181	14	3000
A05	KA041702	15	5000

Queries

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

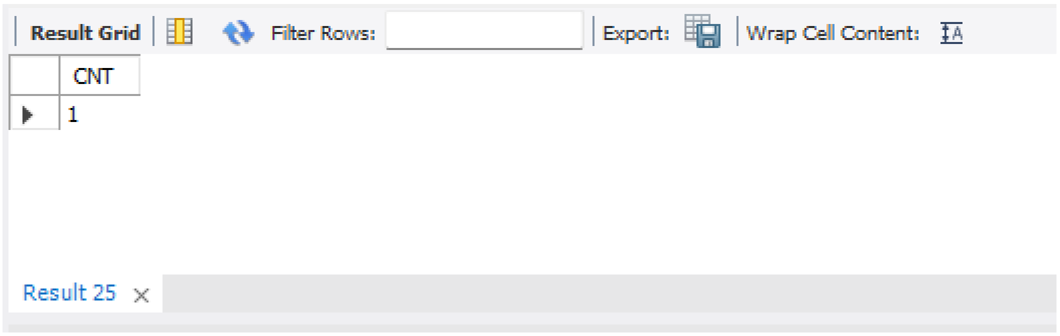
```
update participated

set damage_amount=25000

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



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

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

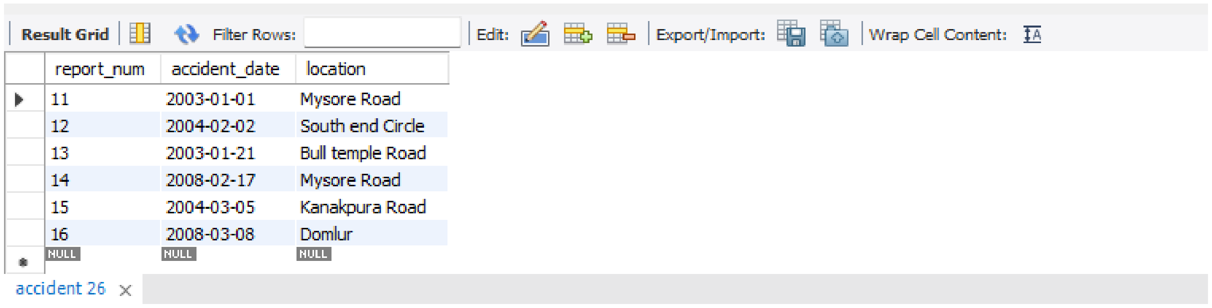
```
select count(distinct driver_id) CNT
from participated a, accident b
where a.report_num=b.report_num and b.accident_date like '2008%';
```



CNT
1

- **Add a new accident to the database.**

```
insert into accident values(16,'2008-03-08',"Domlur");
select * from accident;
```



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

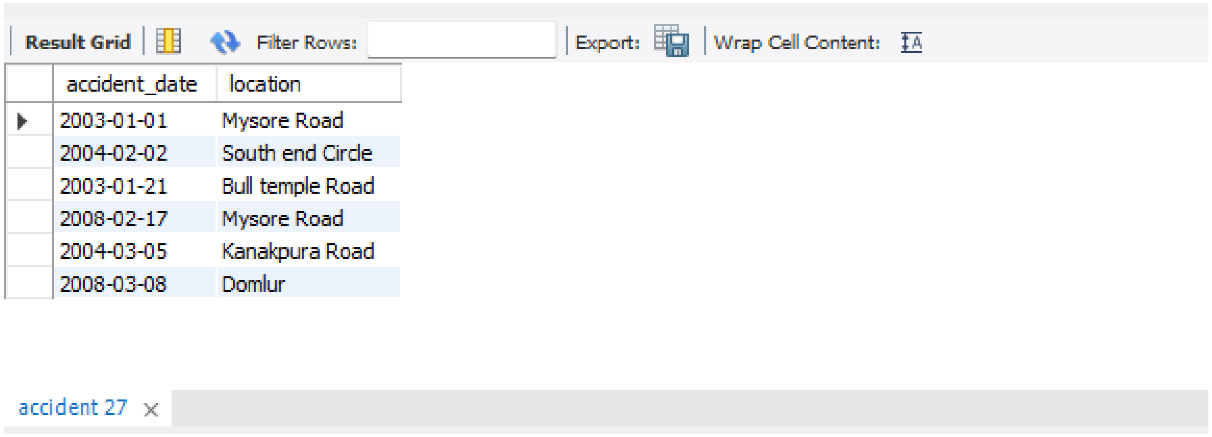
report_num	accident_date	location
11	2003-01-01	Mysore Road
12	2004-02-02	South end Circle
13	2003-01-21	Bull temple Road
14	2008-02-17	Mysore Road
15	2004-03-05	Kanakpura Road
16	2008-03-08	Domlur
NULL	NULL	NULL

accident 26

TO DO:

- **DISPLAY ACCIDENT DATE AND LOCATION**

  select accident_date, location from accident;



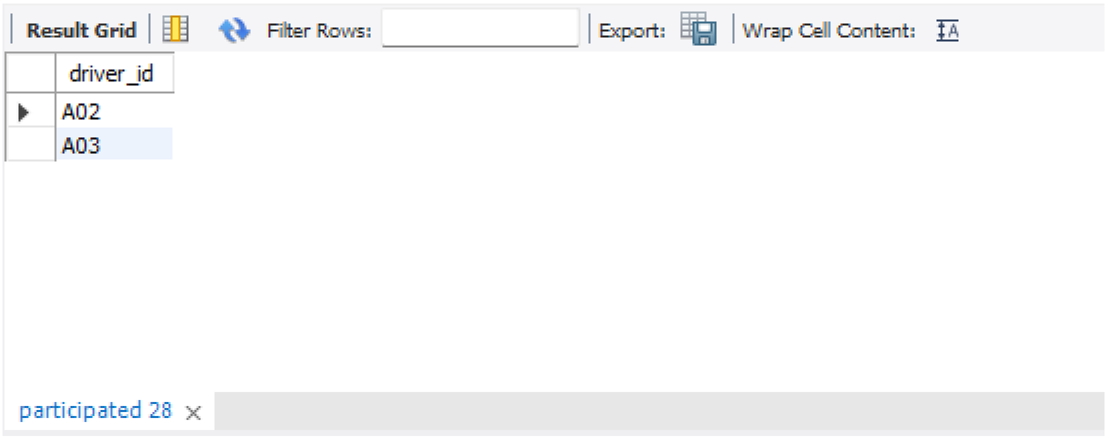
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

accident 27

- **DISPLAY DRIVER ID WHO DID ACCIDENT WITH DAMAGE AMOUNT GREATER THAN OR EQUAL TO RS.25000**

  select driver_id from participated where damage_amount>=25000;



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

driver_id
A02
A03

participated 28

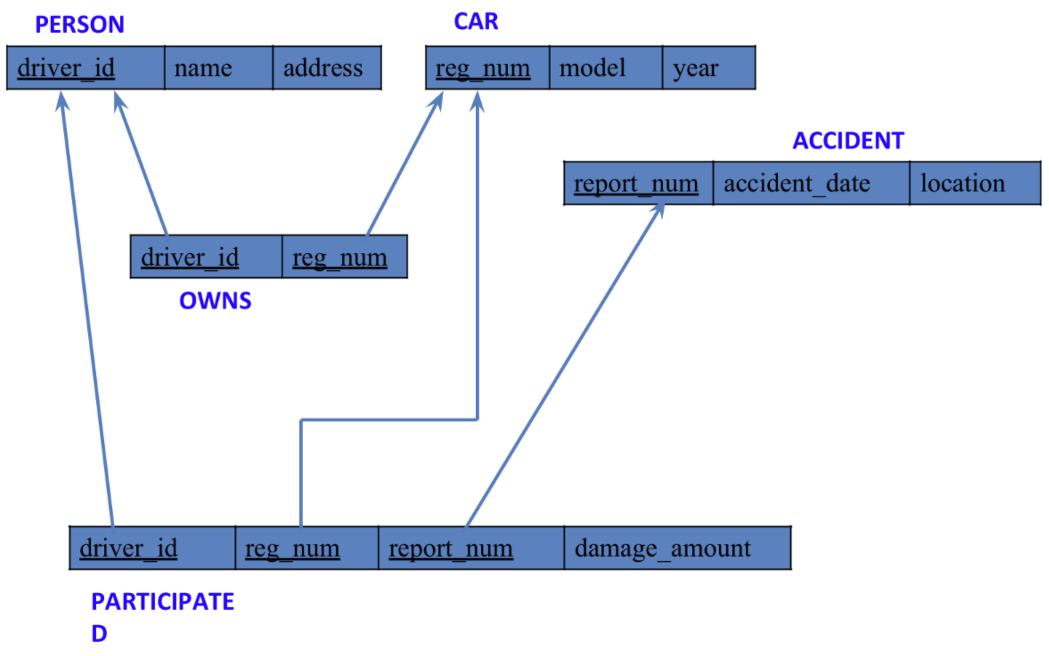
More Queries on Insurance Database

Question

(Week 2)

- 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)
- Display the entire CAR relation in the ascending order of manufacturing year.
- Find the number of accidents in which cars belonging to a specific model (example 'Lancer') were involved.
- Find the total number of people who owned cars that were involved in accidents in 2008.

Schema Diagram



Create database

```
create database insurance_dhiksha;

use insurance_dhiksha;
```

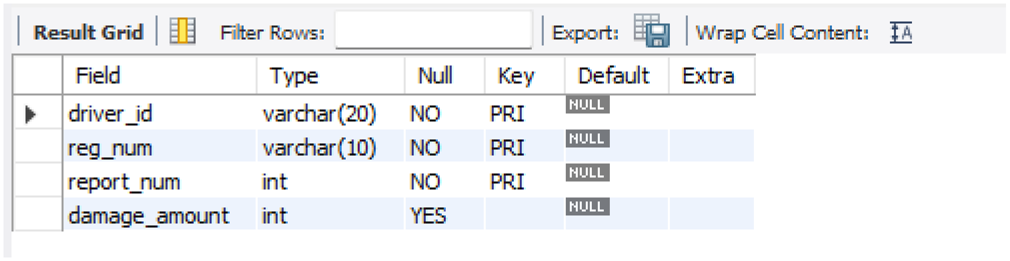
Create table

```
create table insurance_dhiksha.person(

driver_id varchar(20),

name varchar(30),

address  varchar(50),

PRIMARY KEY(driver_id)

);

create table insurance_dhiksha.car(

reg_num varchar(15),

model varchar(10),

year int,

PRIMARY KEY(reg_num)

);
```

```
create table insurance_dhiksha.owns(
driver_id varchar(20),
reg_num varchar(10),
PRIMARY KEY(driver_id, reg_num),
FOREIGN KEY(driver_id) REFERENCES person(driver_id),
FOREIGN KEY(reg_num) REFERENCES car(reg_num)
);
create table insurance_dhiksha.accident(
report_num int,
accident_date date,
location varchar(50),
PRIMARY KEY(report_num)
);
create table insurance_dhiksha.participated(
driver_id varchar(20),
reg_num varchar(10),
report_num int,
damage_amount int,
PRIMARY KEY(driver_id,reg_num,report_num),
FOREIGN KEY(driver_id) REFERENCES person(driver_id),
FOREIGN KEY(reg_num) REFERENCES car(reg_num),
FOREIGN KEY(report_num) REFERENCES accident(report_num)
);
```

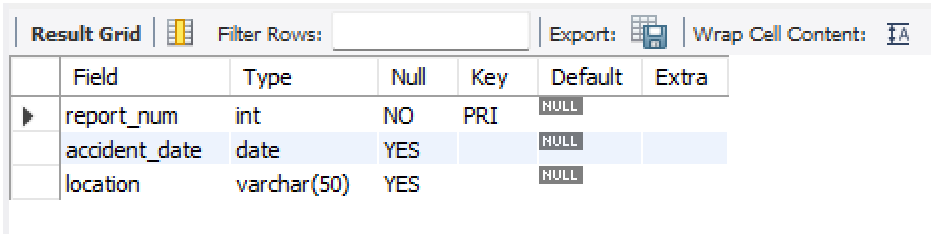
Structure of the table

```
desc person;
```



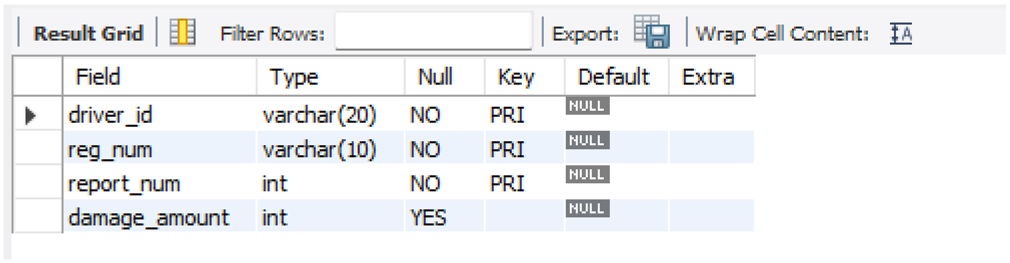
Field	Type	Null	Key	Default	Extra
driver_id	varchar(20)	NO	PRI	NULL	
reg_num	varchar(10)	NO	PRI	NULL	
report_num	int	NO	PRI	NULL	
damage_amount	int	YES		NULL	

desc accident;



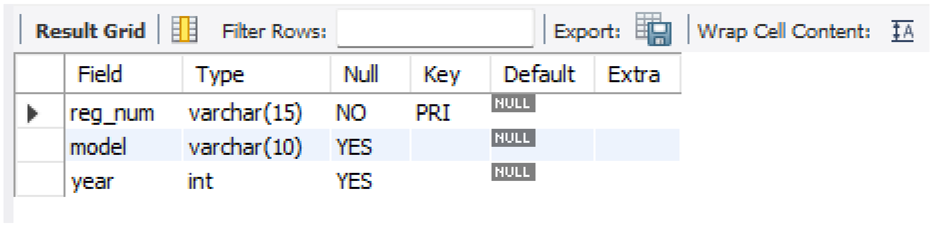
Field	Type	Null	Key	Default	Extra
report_num	int	NO	PRI	NULL	
accident_date	date	YES		NULL	
location	varchar(50)	YES		NULL	

desc participated;



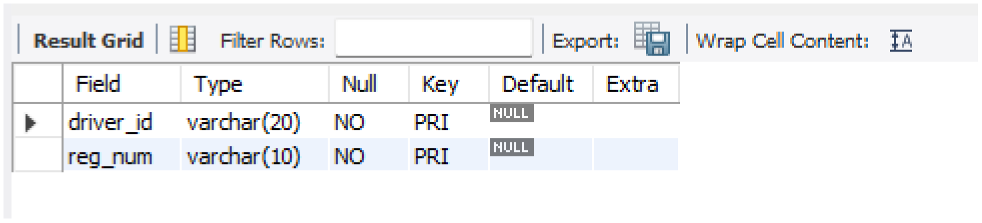
Field	Type	Null	Key	Default	Extra
driver_id	varchar(20)	NO	PRI	NULL	
reg_num	varchar(10)	NO	PRI	NULL	
report_num	int	NO	PRI	NULL	
damage_amount	int	YES		NULL	

desc car;



Field	Type	Null	Key	Default	Extra
reg_num	varchar(15)	NO	PRI	NULL	
model	varchar(10)	YES		NULL	
year	int	YES		NULL	

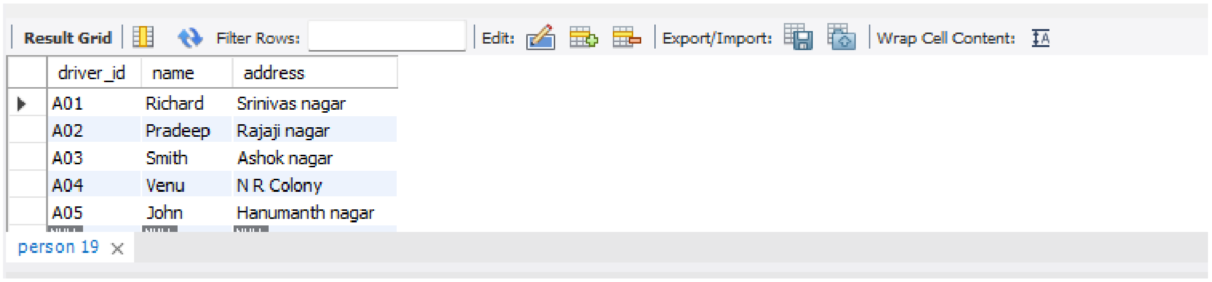
desc owns;



Field	Type	Null	Key	Default	Extra
driver_id	varchar(20)	NO	PRI	NULL	
reg_num	varchar(10)	NO	PRI	NULL	

Inserting Values to the table

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

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

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

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

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

select * from person;
```

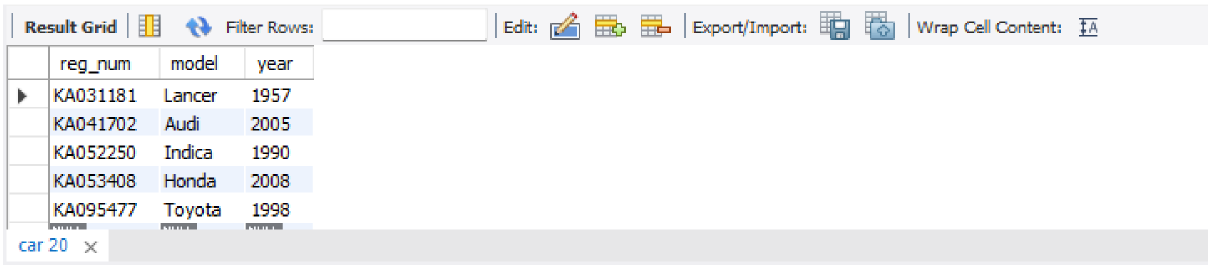


Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

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

person 19

```
insert into car values("KA052250","Indica", "1990");

insert into car values("KA031181","Lancer", "1957");

insert into car values("KA095477","Toyota", "1998");

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

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

select * from car;
```

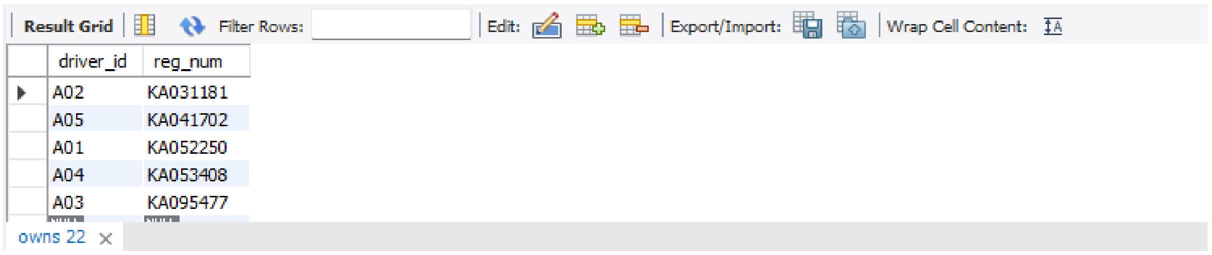


Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

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

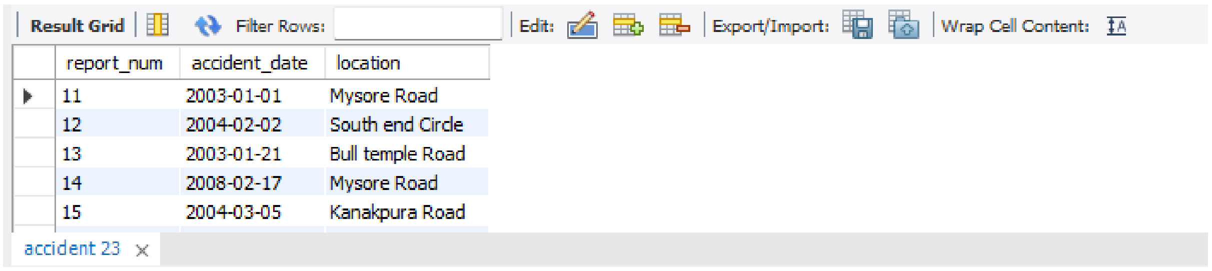
car 20

```
insert into owns values("A01","KA052250");

insert into owns values("A02","KA031181");

insert into owns values("A03","KA095477");

insert into owns values("A04","KA053408");

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

select * from owns;
```



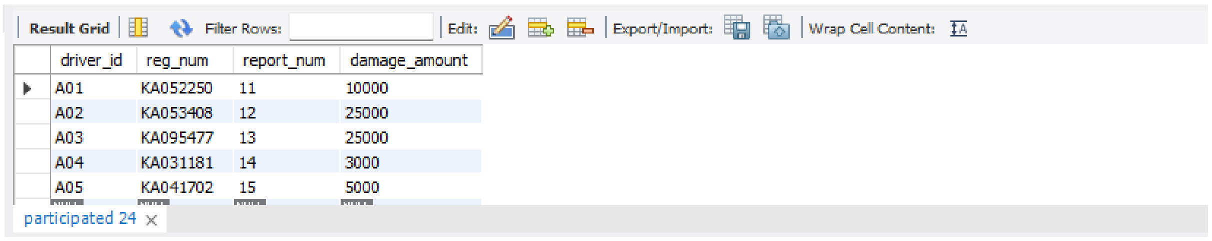
driver_id	reg_num
A02	KA031181
A05	KA041702
A01	KA052250
A04	KA053408
A03	KA095477

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

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

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

insert into accident values(14,'2008-02-17',"Mysore Road");

insert into accident values(15,'2004-03-05',"Kanakpura Road");

select * from accident;
```



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

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

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

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

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

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

select * from participated;
```

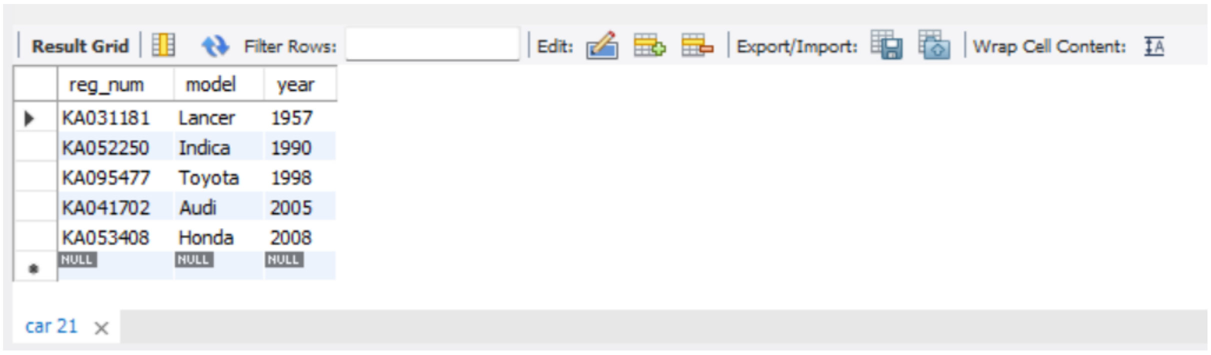


driver_id	reg_num	report_num	damage_amount
A01	KA052250	11	10000
A02	KA053408	12	25000
A03	KA095477	13	25000
A04	KA031181	14	3000
A05	KA041702	15	5000

Queries

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

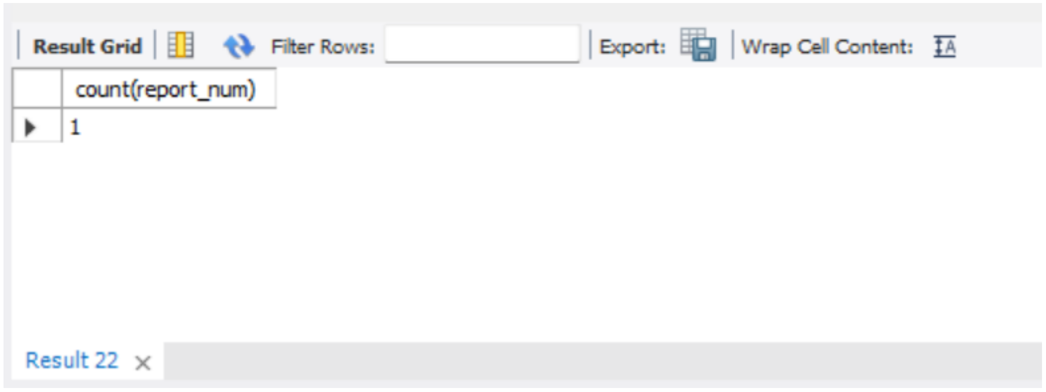
  select * from car order by year asc;



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

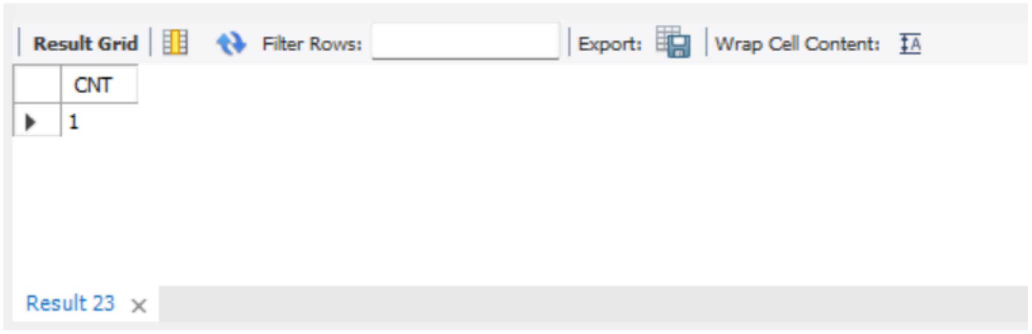
- **Find the number of accidents in which cars belonging to a specific model (example 'Lancer') were involved.**

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



count(report_num)
1

- **Find the total number of people who owned cars that were involved in accidents in 2008.**
  select count(distinct driver_id) CNT
  from participated a, accident b
  where a.report_num=b.report_num and b.accident_date like '__08%';



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

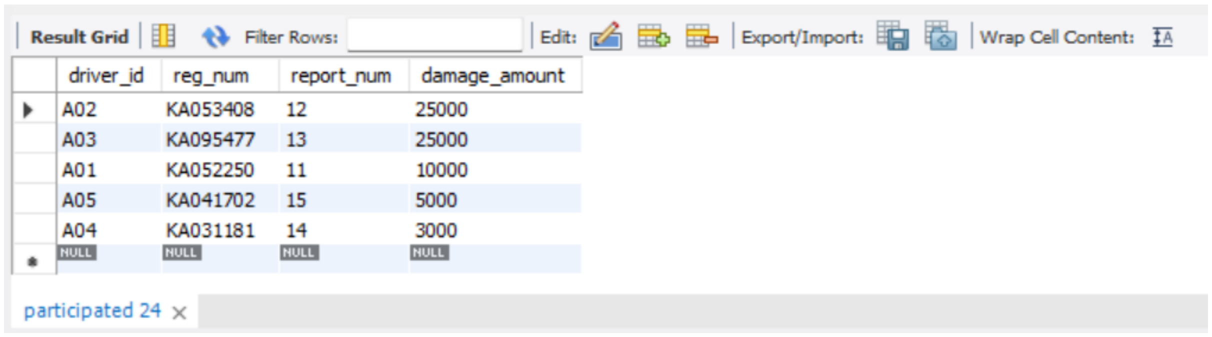
	CNT
▸	1

Result 23

TO DO:

- **LIST THE ENTIRE PARTICIPATED RELATION IN THE DESCENDING ORDER OF DAMAGE AMOUNT.**

  select * from participated order by damage_amount desc;



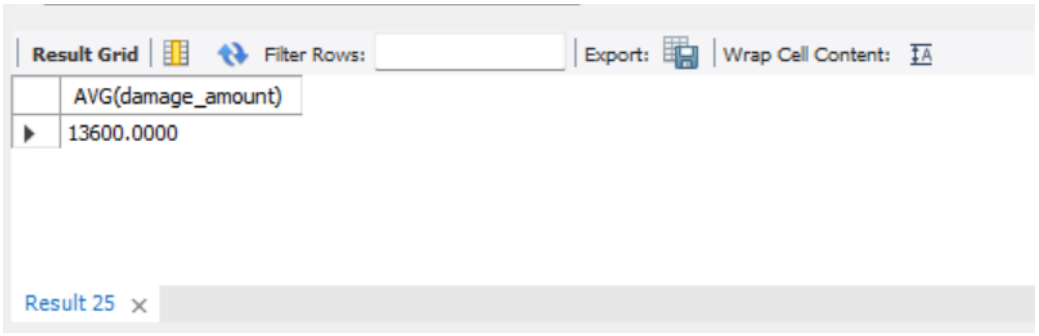
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

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

participated 24

- **FIND THE AVERAGE DAMAGE AMOUNT**

  SELECT AVG(damage_amount) from participated;



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

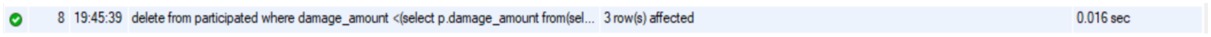
	AVG(damage_amount)
▸	13600.0000

Result 25

- **DELETE THE TUPLE WHOSE DAMAGE AMOUNT IS BELOW THE AVERAGE DAMAGE AMOUNT**

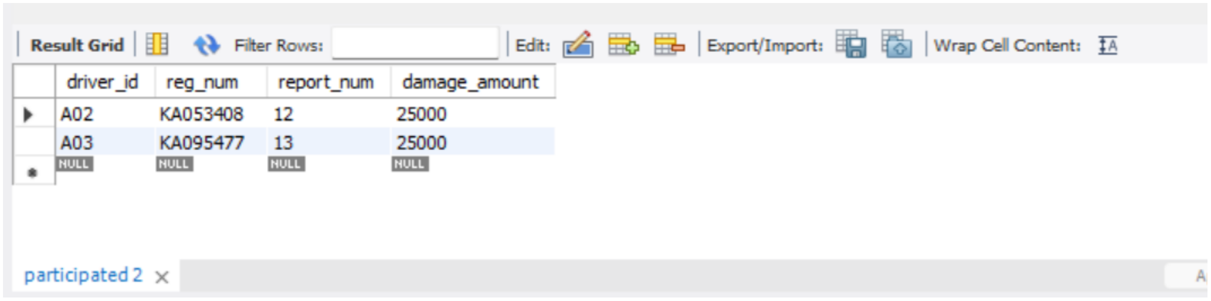
  delete from participated

where damage_amount < (select p.damage_amount from(select AVG(damage_amount) as damage_amount FROM participated)p);



```
8 19:45:39 delete from participated where damage_amount <(select p.damage_amount from(sel... 3 row(s) affected 0.016 sec
```

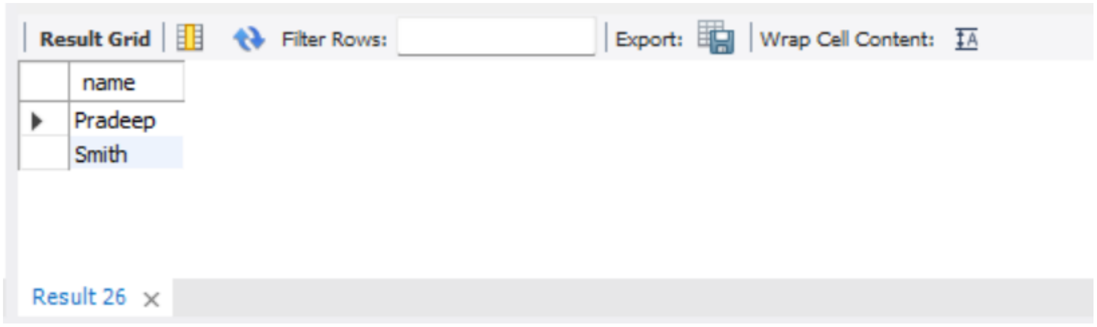
select * from participated;



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

- **LIST THE NAME OF DRIVERS WHOSE DAMAGE IS GREATER THAN THE AVERAGE DAMAGE AMOUNT.**

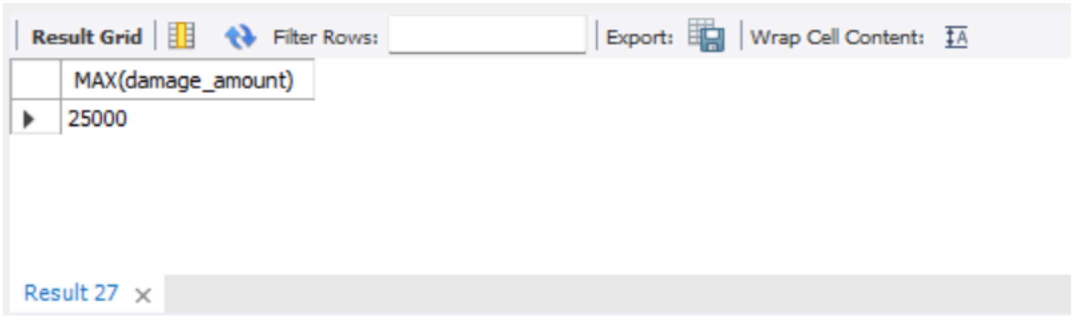
  select name from person p, participated part where p.driver_id=part.driver_id and damage_amount>(select AVG(damage_amount) FROM participated);



name
Pradeep
Smith

- **FIND MAXIMUM DAMAGE AMOUNT.**

  select MAX(damage_amount) from participated;



MAX(damage_amount)
25000

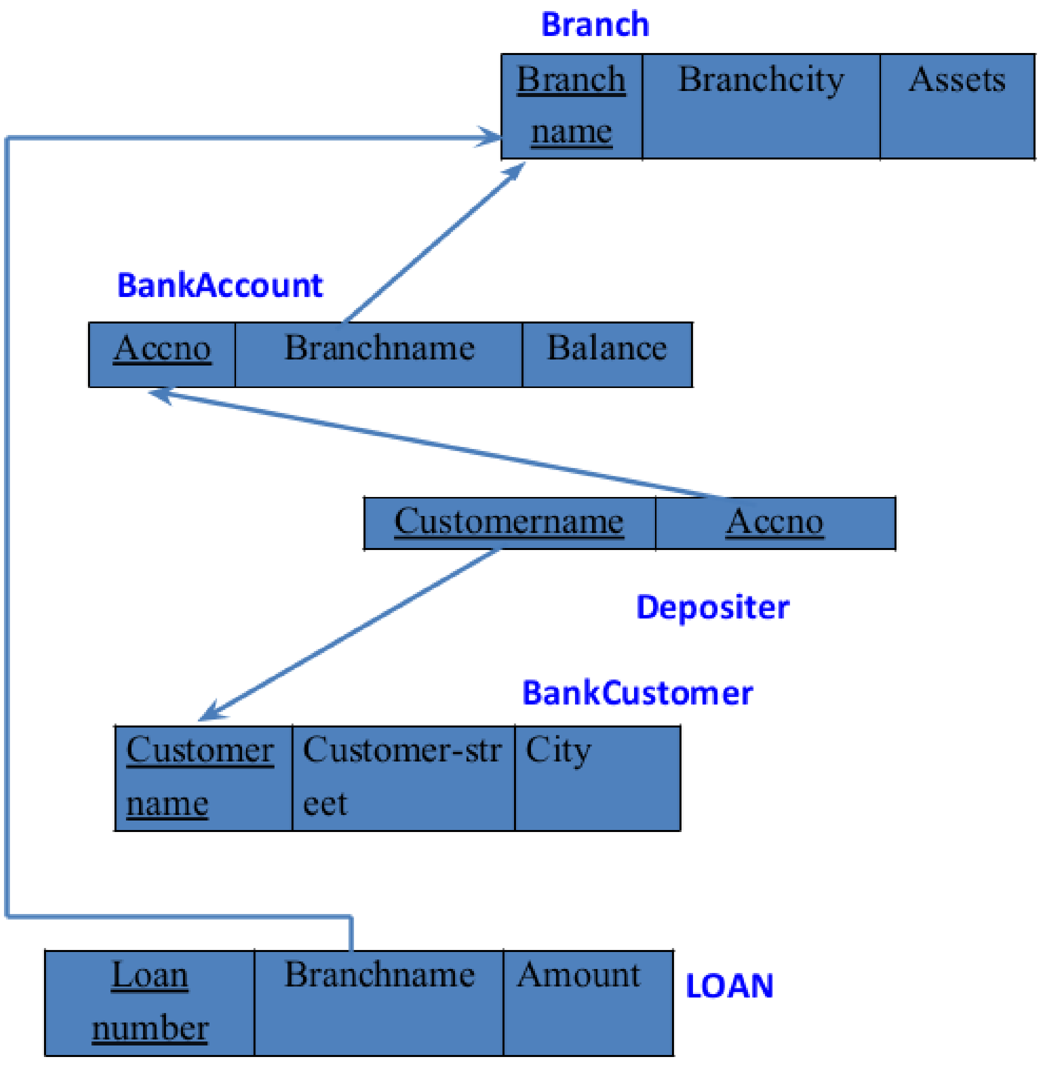
Bank Database

Question

(Week 3)

- 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.
- Display the branch name and assets from all branches in lakhs of rupees and rename the assets column to 'assets in lakhs'.
- Find all the customers who have at least two accounts at the same branch (ex. SBI_ResidencyRoad).
- Create a view which gives each branch the sum of the amount of all the loans at the branch.

Schema Diagram



Create database

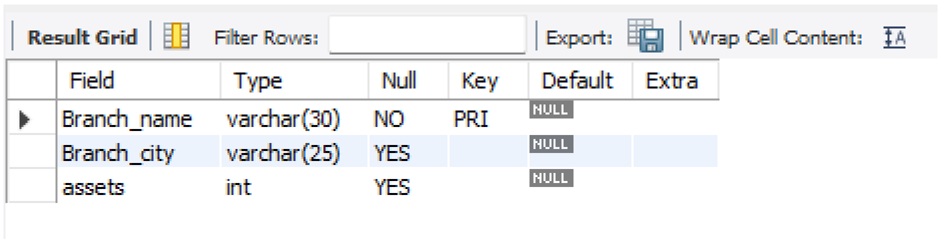
```
create database dhiksha_bank;

use dhiksha_bank;
```

Create table

```
create table dhiksha_bank.branch(

Branch_name varchar(30),

Branch_city varchar(25),

assets int,
PRIMARY KEY (Branch_name)

);

create table dhiksha_bank.BankAccount(

Accno int,

Branch_name varchar(30),

Balance int,

PRIMARY KEY(Accno),

foreign key (Branch_name) references branch(Branch_name)

);

create table dhiksha_bank.BankCustomer(

Customername varchar(20),

Customer_street varchar(30),

CustomerCity varchar (35),

PRIMARY KEY(Customername)

);

create table dhiksha_bank.Depositer(

Customername varchar(20),

Accno int,

PRIMARY KEY(Customername,Accno),

foreign key (Accno) references BankAccount(Accno),

foreign key (Customername) references BankCustomer(Customername)

);

create table dhiksha_bank.Loan(

Loan_number int,

Branch_name varchar(30),

Amount int,

PRIMARY KEY(Loan_number),

foreign key (Branch_name) references branch(Branch_name)

);
```

Structure of the table

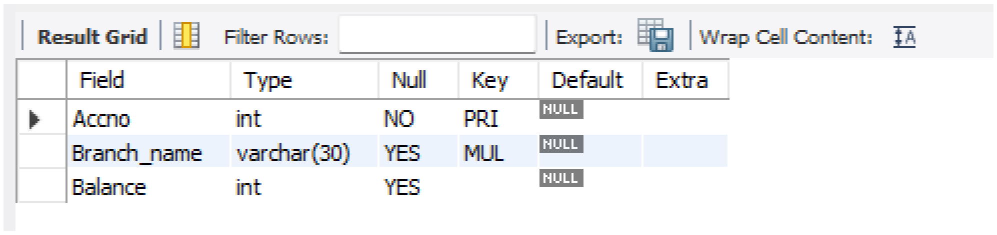
desc branch;



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

Field	Type	Null	Key	Default	Extra
Branch_name	varchar(30)	NO	PRI	NULL	
Branch_city	varchar(25)	YES		NULL	
assets	int	YES		NULL	

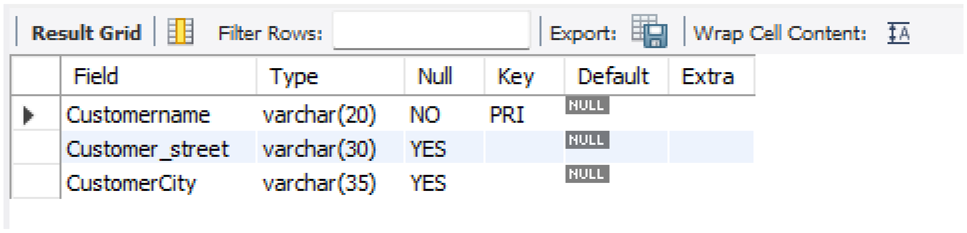
desc BankAccount;



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

Field	Type	Null	Key	Default	Extra
Accno	int	NO	PRI	NULL	
Branch_name	varchar(30)	YES	MUL	NULL	
Balance	int	YES		NULL	

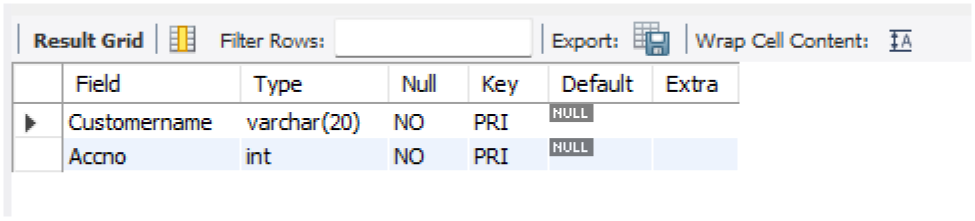
desc BankCustomer;



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

Field	Type	Null	Key	Default	Extra
Customername	varchar(20)	NO	PRI	NULL	
Customer_street	varchar(30)	YES		NULL	
CustomerCity	varchar(35)	YES		NULL	

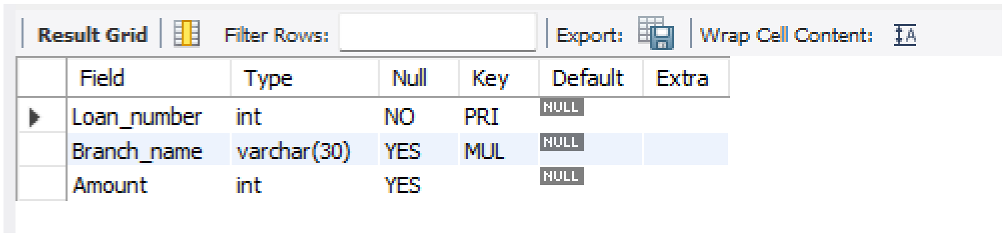
desc Depositer;



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

Field	Type	Null	Key	Default	Extra
Customername	varchar(20)	NO	PRI	NULL	
Accno	int	NO	PRI	NULL	

desc Loan;

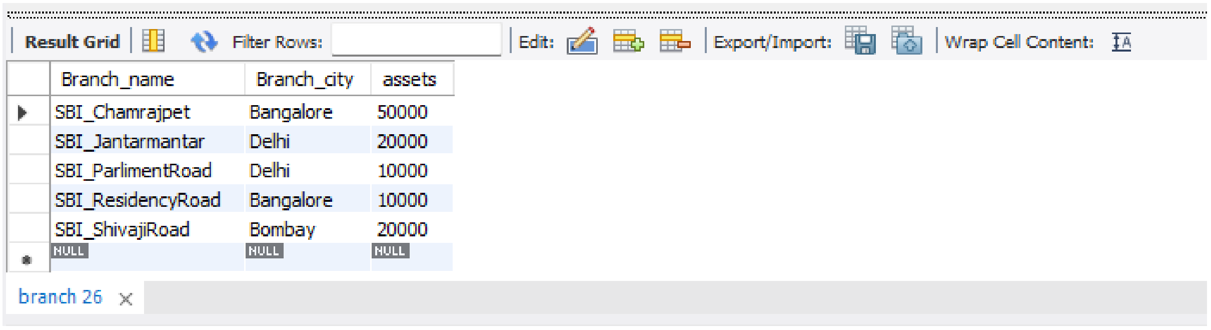


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
Loan_number	int	NO	PRI	NULL	
Branch_name	varchar(30)	YES	MUL	NULL	
Amount	int	YES		NULL	

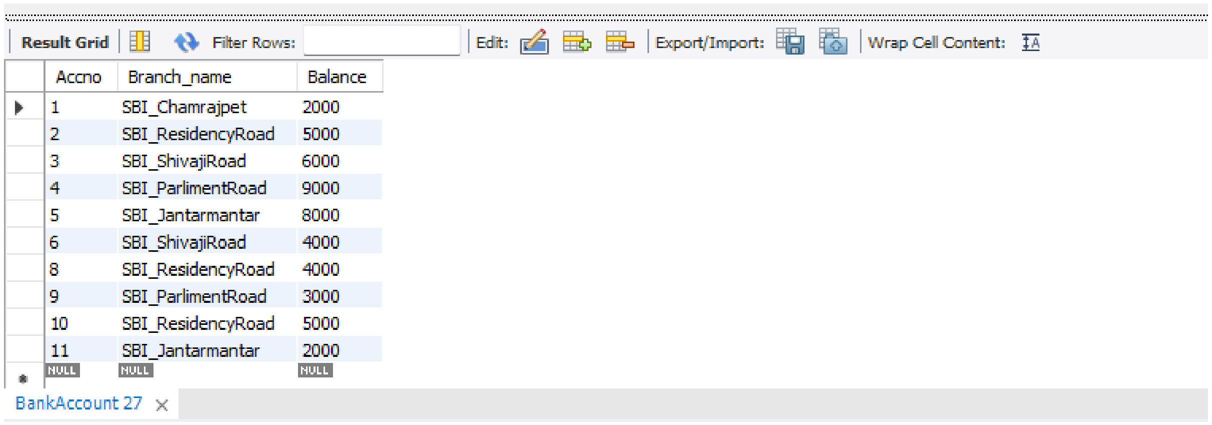
Inserting Values to the table

```
insert into branch values("SBI_Chamrajpet","Bangalore",50000);

insert into branch values("SBI_ResidencyRoad","Bangalore",10000);

insert into branch values("SBI_ShivajiRoad","Bombay",20000);

insert into branch values("SBI_ParlimentRoad","Delhi",10000);

insert into branch values("SBI_Jantarmantar","Delhi",20000);

select * from branch;
```



Branch_name	Branch_city	assets
SBI_Chamrajpet	Bangalore	50000
SBI_Jantarmantar	Delhi	20000
SBI_ParlimentRoad	Delhi	10000
SBI_ResidencyRoad	Bangalore	10000
SBI_ShivajiRoad	Bombay	20000
NULL	NULL	NULL

```
insert into BankAccount values(1,"SBI_Chamrajpet",2000);

insert into BankAccount values(2,"SBI_ResidencyRoad",5000);

insert into BankAccount values(3,"SBI_ShivajiRoad",6000);

insert into BankAccount values(4,"SBI_ParlimentRoad",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_ParlimentRoad",3000);

insert into BankAccount values(10,"SBI_ResidencyRoad",5000);

insert into BankAccount values(11,"SBI_Jantarmantar",2000);

select * from BankAccount;
```



Accno	Branch_name	Balance
1	SBI_Chamrajpet	2000
2	SBI_ResidencyRoad	5000
3	SBI_ShivajiRoad	6000
4	SBI_ParlimentRoad	9000
5	SBI_Jantarmantar	8000
6	SBI_ShivajiRoad	4000
8	SBI_ResidencyRoad	4000
9	SBI_ParlimentRoad	3000
10	SBI_ResidencyRoad	5000
11	SBI_Jantarmantar	2000
NULL	NULL	NULL

insert into BankCustomer values("Avinash","Bull_Temple_Road","Bangalore");

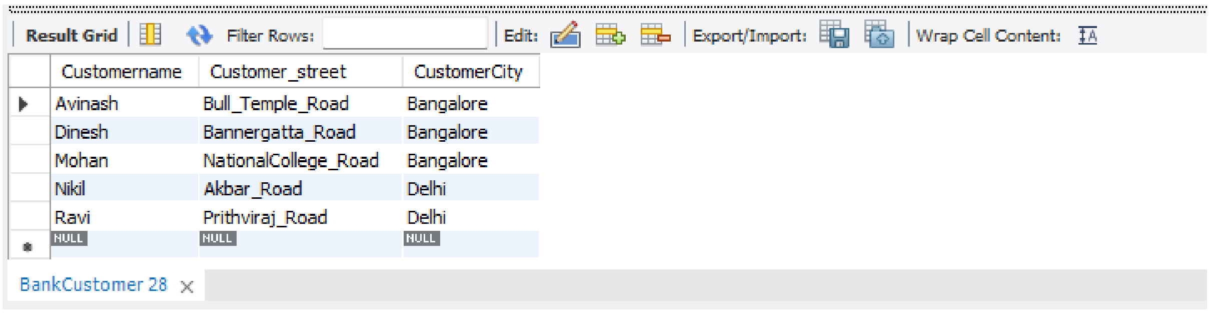
insert into BankCustomer values("Dinesh","Bannergatta_Road","Bangalore");

insert into BankCustomer values("Mohan","NationalCollege_Road","Bangalore");

insert into BankCustomer values("Nikil","Akbar_Road","Delhi");

insert into BankCustomer values("Ravi","Prithviraj_Road","Delhi");

select * from BankCustomer;



Customername	Customer_street	CustomerCity
Avinash	Bull_Temple_Road	Bangalore
Dinesh	Bannergatta_Road	Bangalore
Mohan	NationalCollege_Road	Bangalore
Nikil	Akbar_Road	Delhi
Ravi	Prithviraj_Road	Delhi
NULL	NULL	NULL

insert into Depositer values("Avinash",1);

insert into Depositer values("Dinesh",2);

insert into Depositer values("Nikil",4);

insert into Depositer values("Ravi",5);

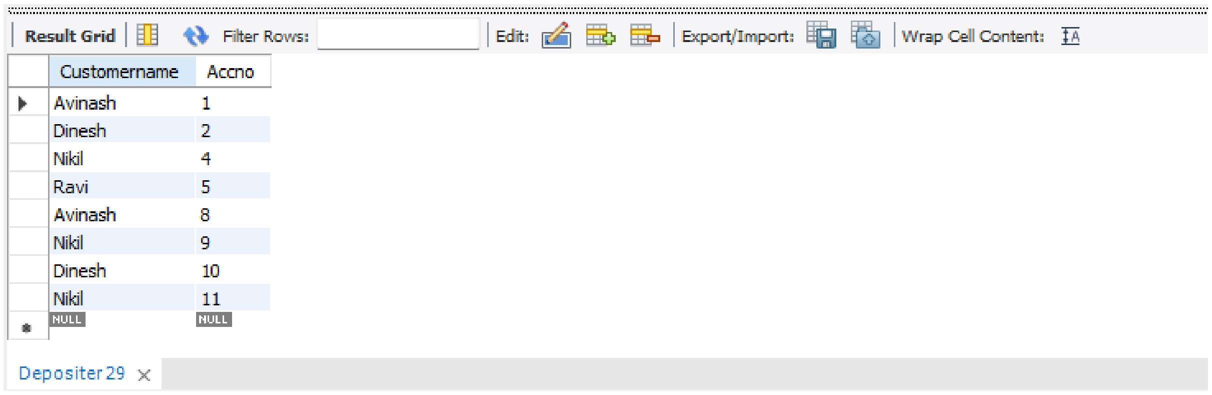
insert into Depositer values("Avinash",8);

insert into Depositer values("Nikil",9);

insert into Depositer values("Dinesh",10);

insert into Depositer values("Nikil",11);

select * from Depositer;



Customername	Accno
Avinash	1
Dinesh	2
Nikil	4
Ravi	5
Avinash	8
Nikil	9
Dinesh	10
Nikil	11
NULL	NULL

Depositer 29

insert into Loan values(1,"SBI_Chamrajpet",1000);

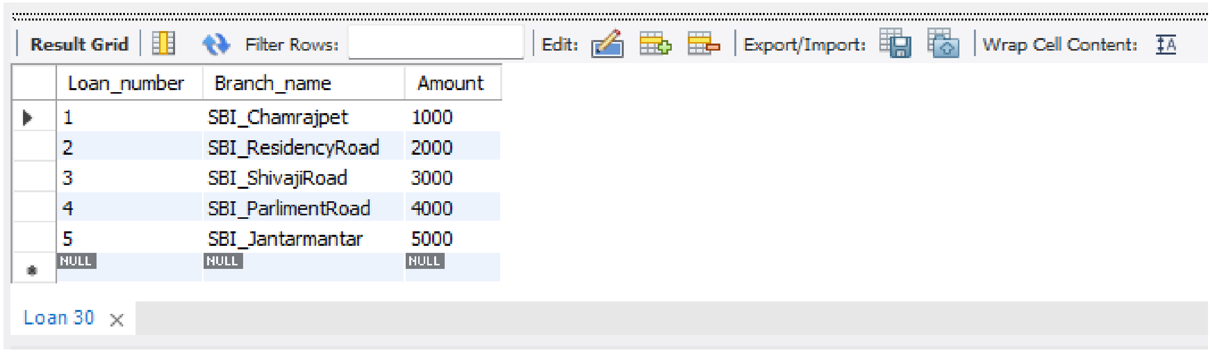
insert into Loan values(2,"SBI_ResidencyRoad",2000);

insert into Loan values(3,"SBI_ShivajiRoad",3000);

insert into Loan values(4,"SBI_ParlimentRoad",4000);

insert into Loan values(5,"SBI_Jantarmantar",5000);

select * from Loan;



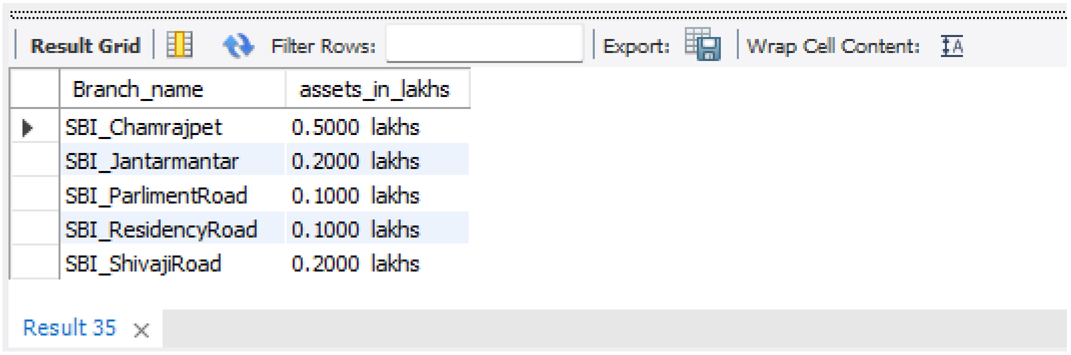
Loan_number	Branch_name	Amount
1	SBI_Chamrajpet	1000
2	SBI_ResidencyRoad	2000
3	SBI_ShivajiRoad	3000
4	SBI_ParlimentRoad	4000
5	SBI_Jantarmantar	5000
NULL	NULL	NULL

Loan 30

Queries

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

  select Branch_name, CONCAT(assets/100000,' lakhs')assets_in_lakhs from branch;

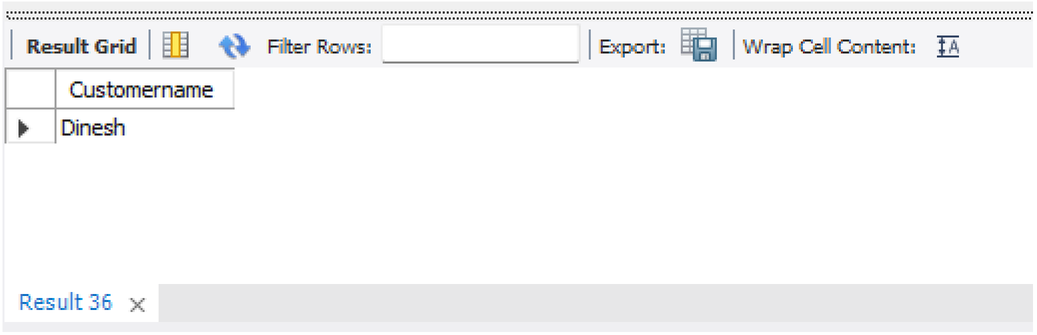


Branch_name	assets_in_lakhs
SBI_Chamrajpet	0.5000 lakhs
SBI_Jantarmantar	0.2000 lakhs
SBI_ParlimentRoad	0.1000 lakhs
SBI_ResidencyRoad	0.1000 lakhs
SBI_ShivajiRoad	0.2000 lakhs

Result 35

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

  select d.Customername from Depositer d, BankAccount b where b.Branch_name='SBI_ResidencyRoad' and d.Accno=b.Accno group by d.Customername having count(d.Accno)>=2;

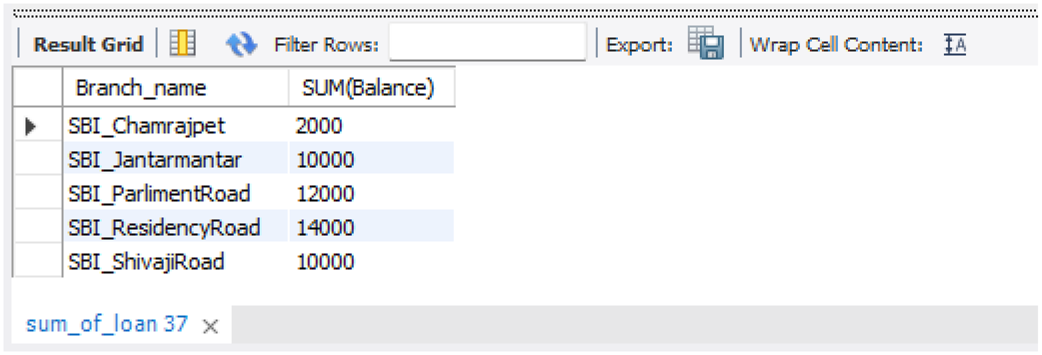


Customername
Dinesh

Result 36

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

  create view sum_of_loan
  as select Branch_name, SUM(Balance)
  from BankAccount
  group by Branch_name;
  select * from sum_of_loan;

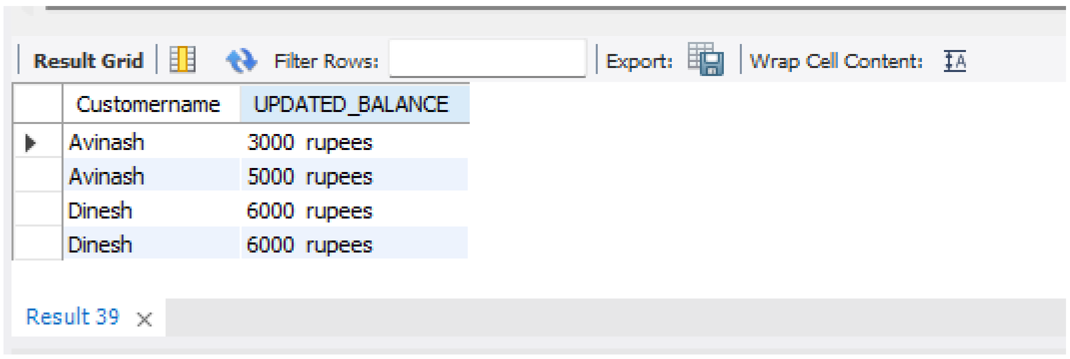

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Branch_name	SUM(Balance)
SBI_Chamrajpet	2000
SBI_Jantarmantar	10000
SBI_ParlimentRoad	12000
SBI_ResidencyRoad	14000
SBI_ShivajiRoad	10000

sum_of_loan 37

SPOT QUERY:
UPDATE OR ADD RUPEES 1000 TO ACCOUNT BALANCE FOR THE CUSTOMERS WHO ARE RESIDING IN BANGALORE.

```
select bc.Customername, CONCAT(Balance+1000,'  rupees') UPDATED_BALANCE from
BankAccount b, BankCustomer bc, Depositer d where
bc.Customername=d.Customername and b.Accno=d.Accno and
bc.Customercity='Bangalore';
```


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

Customername	UPDATED_BALANCE
Avinash	3000 rupees
Avinash	5000 rupees
Dinesh	6000 rupees
Dinesh	6000 rupees

Result 39

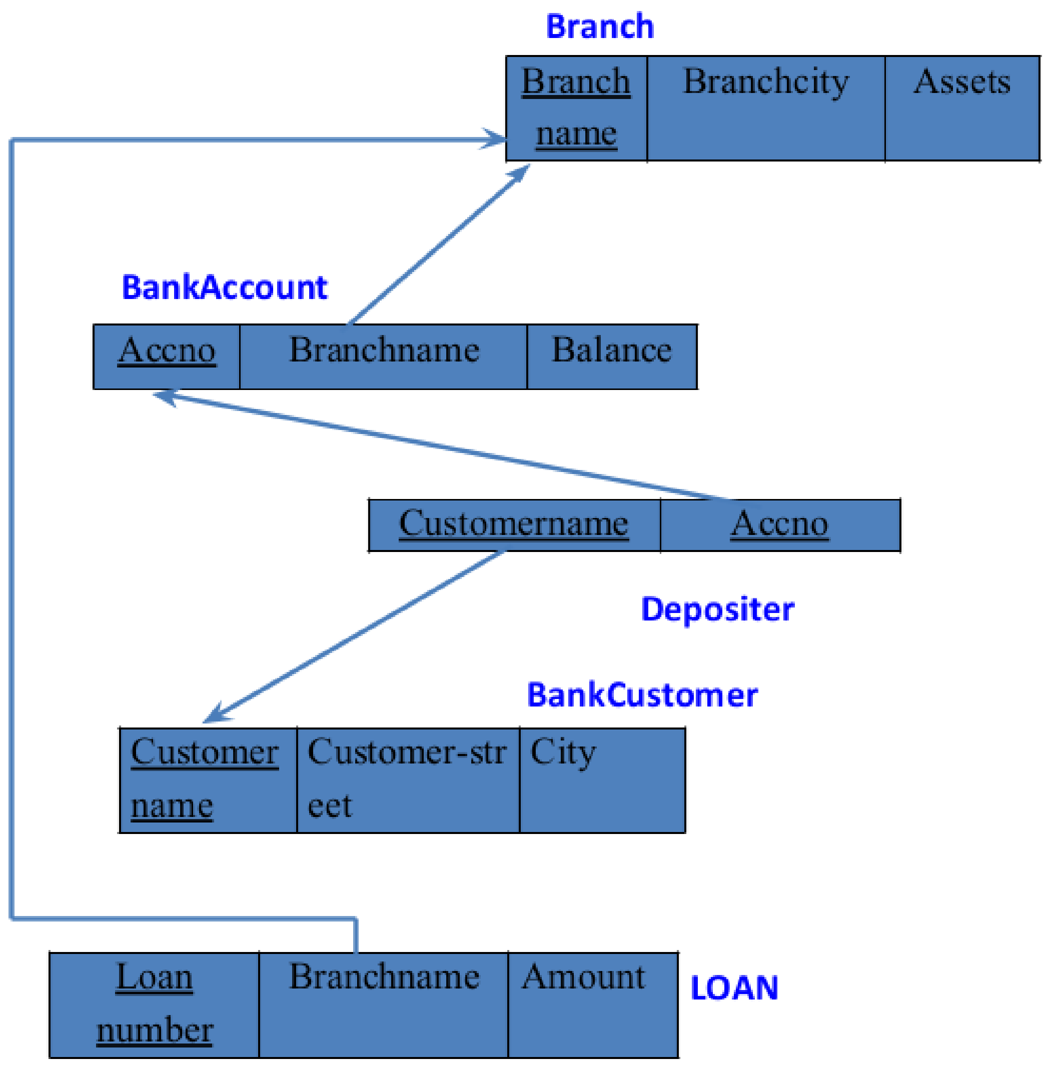
More Queries on Bank Database

Question

(Week 4)

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

Schema Diagram



Create database

```
create database dhiksha_bank;
use dhiksha_bank;
```

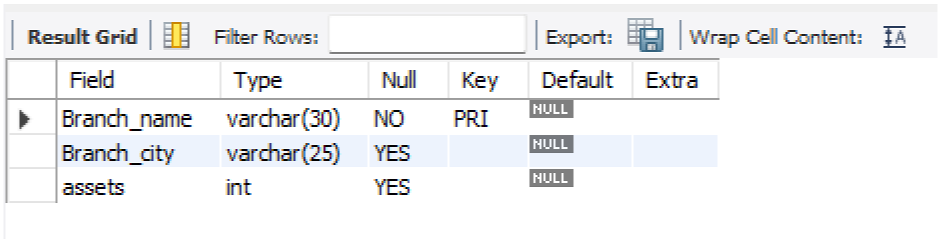
Create table

```
create table dhiksha_bank.branch(
Branch_name varchar(30),
Branch_city varchar(25),
assets int,
PRIMARY KEY (Branch_name)

);

create table dhiksha_bank.BankAccount(

Accno int,

Branch_name varchar(30),

Balance int,

PRIMARY KEY(Accno),

foreign key (Branch_name) references branch(Branch_name)

);

create table dhiksha_bank.BankCustomer(

Customername varchar(20),

Customer_street varchar(30),

CustomerCity varchar (35),

PRIMARY KEY(Customername)

);

create table dhiksha_bank.Depositer(

Customername varchar(20),

Accno int,

PRIMARY KEY(Customername,Accno),

foreign key (Accno) references BankAccount(Accno),

foreign key (Customername) references BankCustomer(Customername)

);

create table dhiksha_bank.Loan(

Loan_number int,

Branch_name varchar(30),

Amount int,

PRIMARY KEY(Loan_number),

foreign key (Branch_name) references branch(Branch_name)

);
```

```
create table Borrower(

Customername varchar(20),

Loan_number int,

foreign key(Customername) references BankCustomer(Customername),

foreign key(Loan_number) references Loan(Loan_number)

);
```

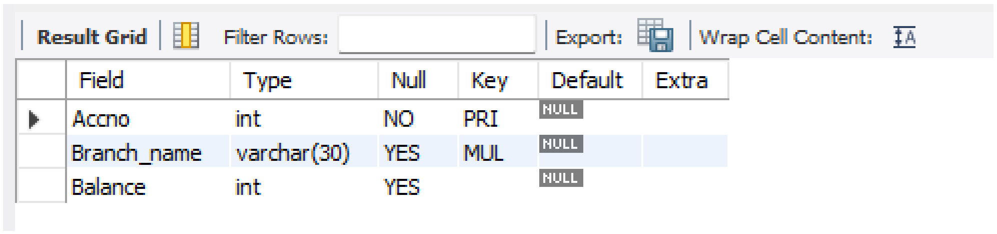
Structure of the table

desc branch;



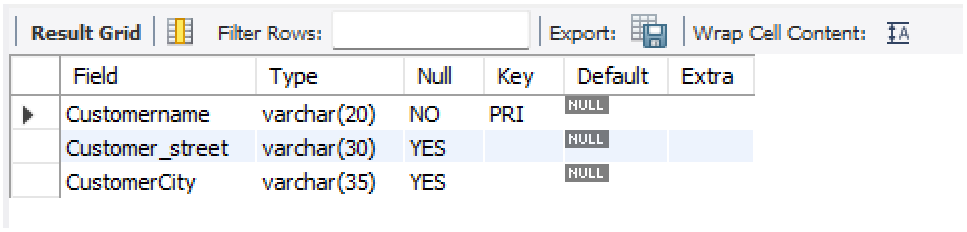
Field	Type	Null	Key	Default	Extra
Branch_name	varchar(30)	NO	PRI	NULL	
Branch_city	varchar(25)	YES		NULL	
assets	int	YES		NULL	

desc BankAccount;



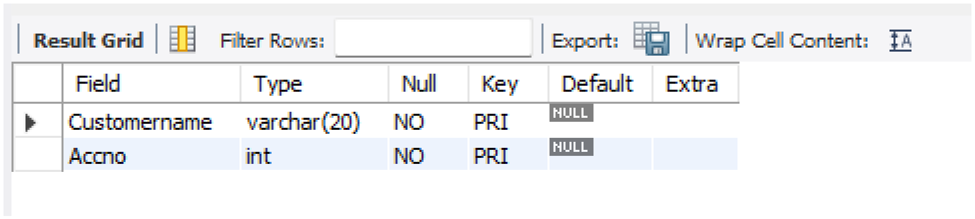
Field	Type	Null	Key	Default	Extra
Accno	int	NO	PRI	NULL	
Branch_name	varchar(30)	YES	MUL	NULL	
Balance	int	YES		NULL	

desc BankCustomer;



Field	Type	Null	Key	Default	Extra
Customername	varchar(20)	NO	PRI	NULL	
Customer_street	varchar(30)	YES		NULL	
CustomerCity	varchar(35)	YES		NULL	

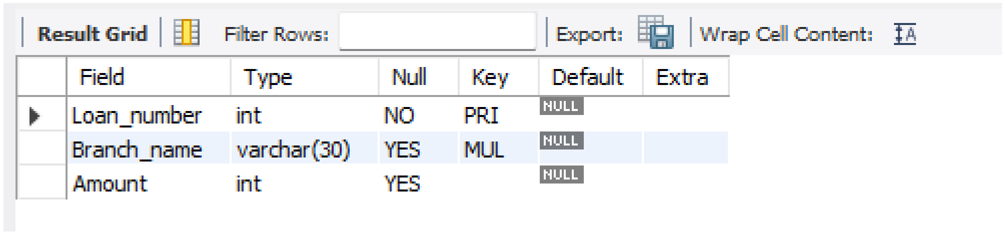
desc Depositer;



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

Field	Type	Null	Key	Default	Extra
Customername	varchar(20)	NO	PRI	NULL	
Accno	int	NO	PRI	NULL	

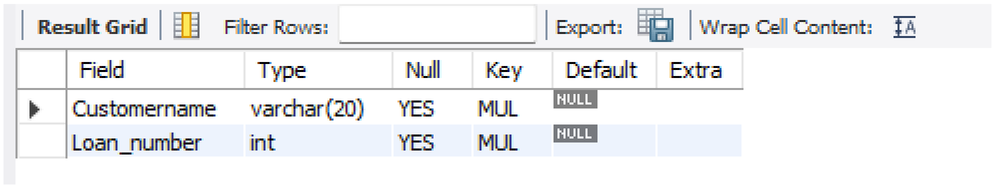
desc Loan;



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

Field	Type	Null	Key	Default	Extra
Loan_number	int	NO	PRI	NULL	
Branch_name	varchar(30)	YES	MUL	NULL	
Amount	int	YES		NULL	

desc Borrower;

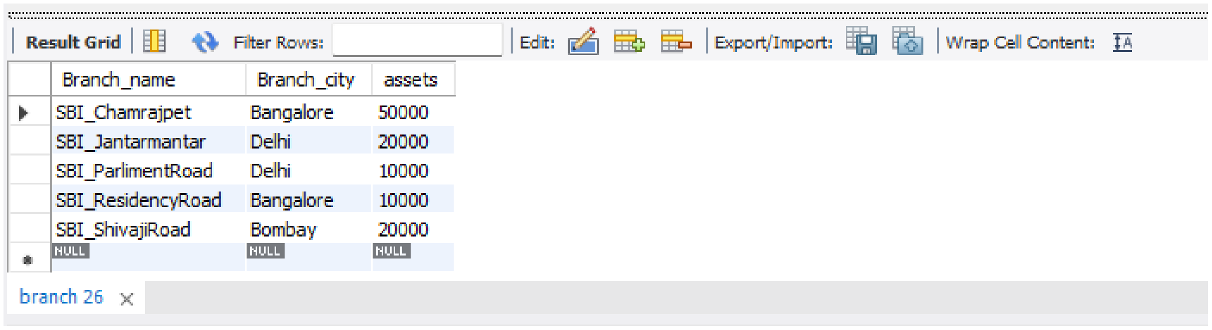


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
Customername	varchar(20)	YES	MUL	NULL	
Loan_number	int	YES	MUL	NULL	

Inserting Values to the table

```
insert into branch values("SBI_Chamrajpet","Bangalore",50000);

insert into branch values("SBI_ResidencyRoad","Bangalore",10000);

insert into branch values("SBI_ShivajiRoad","Bombay",20000);

insert into branch values("SBI_ParlimentRoad","Delhi",10000);

insert into branch values("SBI_Jantarmantar","Delhi",20000);

select * from branch;
```



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Branch_name	Branch_city	assets
SBI_Chamrajpet	Bangalore	50000
SBI_Jantarmantar	Delhi	20000
SBI_ParlimentRoad	Delhi	10000
SBI_ResidencyRoad	Bangalore	10000
SBI_ShivajiRoad	Bombay	20000
NULL	NULL	NULL

branch 26

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_ParlimentRoad",9000);

insert into BankAccount values(5,"SBI_Jantarmantar",8000);

insert into BankAccount values(6,"SBI_ShivajiRoad",4000);

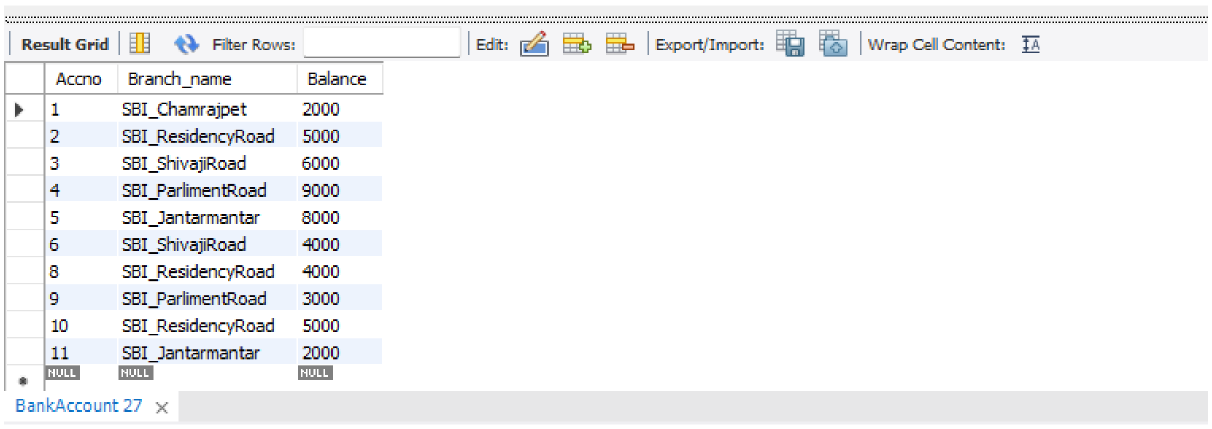
insert into BankAccount values(8,"SBI_ResidencyRoad",4000);

insert into BankAccount values(9,"SBI_ParlimentRoad",3000);

insert into BankAccount values(10,"SBI_ResidencyRoad",5000);

insert into BankAccount values(11,"SBI_Jantarmantar",2000);

select * from BankAccount;



Accno	Branch_name	Balance
1	SBI_Chamrajpet	2000
2	SBI_ResidencyRoad	5000
3	SBI_ShivajiRoad	6000
4	SBI_ParlimentRoad	9000
5	SBI_Jantarmantar	8000
6	SBI_ShivajiRoad	4000
8	SBI_ResidencyRoad	4000
9	SBI_ParlimentRoad	3000
10	SBI_ResidencyRoad	5000
11	SBI_Jantarmantar	2000
NULL	NULL	NULL

insert into BankCustomer values("Avinash","Bull_Temple_Road","Bangalore");

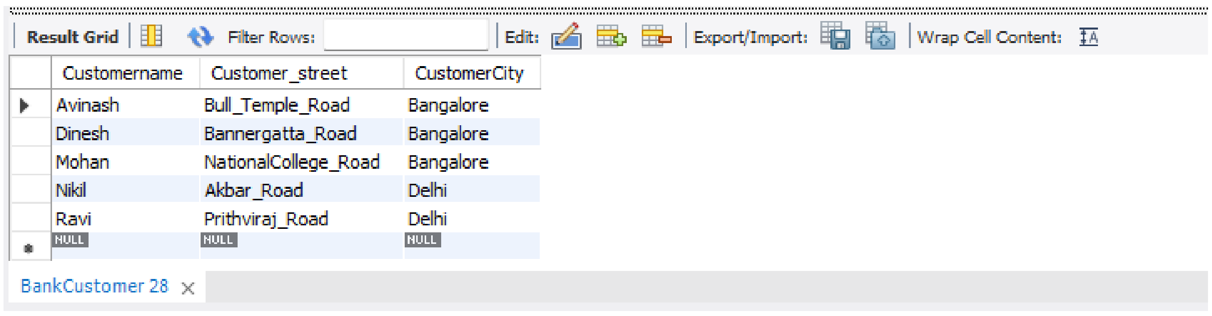
insert into BankCustomer values("Dinesh","Bannergatta_Road","Bangalore");

insert into BankCustomer values("Mohan","NationalCollege_Road","Bangalore");

insert into BankCustomer values("Nikil","Akbar_Road","Delhi");

insert into BankCustomer values("Ravi","Prithviraj_Road","Delhi");

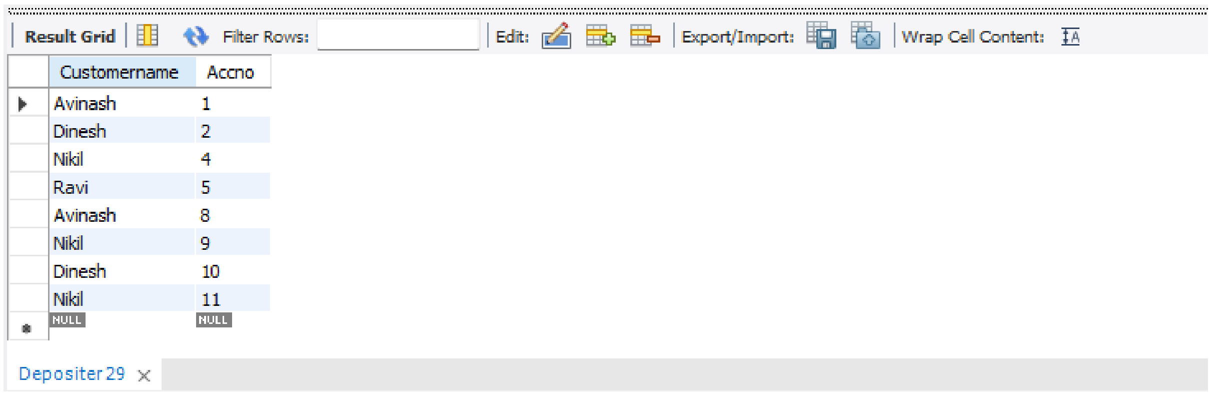
select * from BankCustomer;



Customername	Customer_street	CustomerCity
Avinash	Bull_Temple_Road	Bangalore
Dinesh	Bannergatta_Road	Bangalore
Mohan	NationalCollege_Road	Bangalore
Nikil	Akbar_Road	Delhi
Ravi	Prithviraj_Road	Delhi
NULL	NULL	NULL

BankCustomer 28

```
insert into Depositer values("Avinash",1);

insert into Depositer values("Dinesh",2);

insert into Depositer values("Nikil",4);

insert into Depositer values("Ravi",5);

insert into Depositer values("Avinash",8);

insert into Depositer values("Nikil",9);

insert into Depositer values("Dinesh",10);

insert into Depositer values("Nikil",11);

select * from Depositer;
```

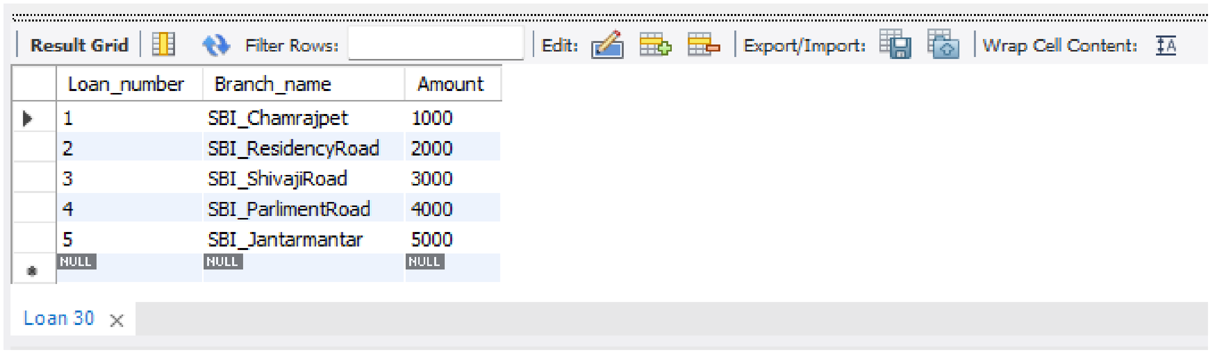


Customername	Accno
Avinash	1
Dinesh	2
Nikil	4
Ravi	5
Avinash	8
Nikil	9
Dinesh	10
Nikil	11
NULL	NULL

Depositer 29

```
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_ParlimentRoad",4000);

insert into Loan values(5,"SBI_Jantarmantar",5000);
```

select * from Loan;



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Loan_number	Branch_name	Amount
1	SBI_Chamrajpet	1000
2	SBI_ResidencyRoad	2000
3	SBI_ShivajiRoad	3000
4	SBI_ParlimentRoad	4000
5	SBI_Jantarmantar	5000
NULL	NULL	NULL

Loan 30

insert into Borrower values("Avinash",1);

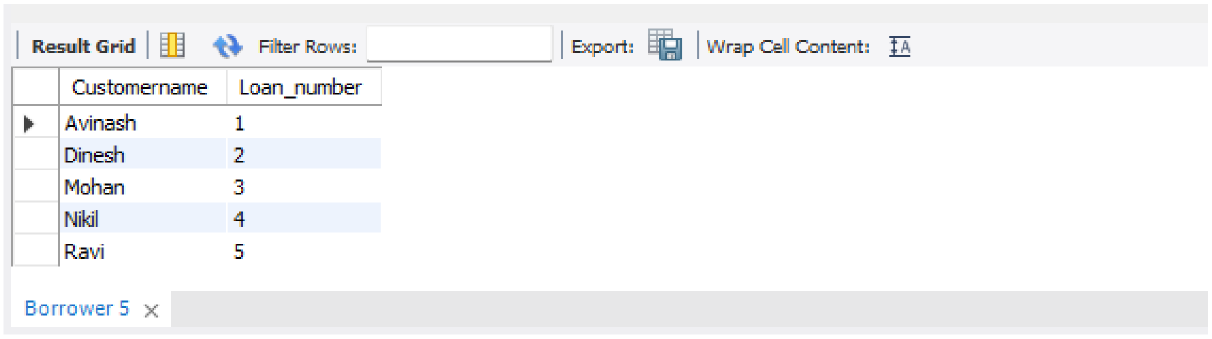
insert into Borrower values("Dinesh",2);

insert into Borrower values("Mohan",3);

insert into Borrower values("Nikil",4);

insert into Borrower values("Ravi",5);

select * from Borrower ;



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

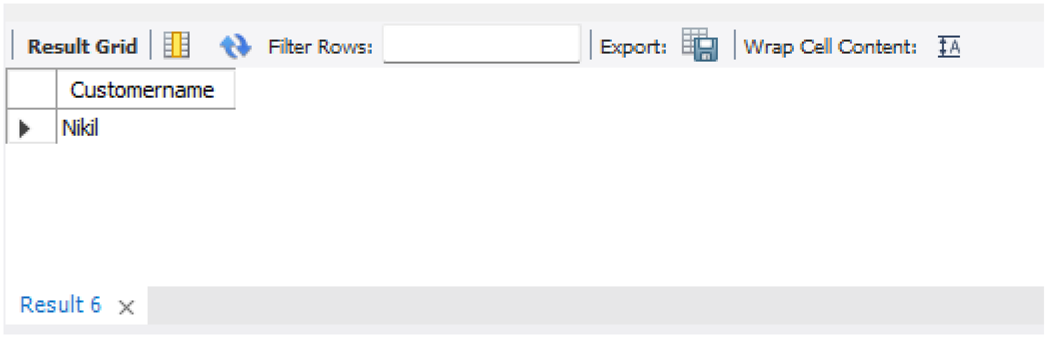
Customername	Loan_number
Avinash	1
Dinesh	2
Mohan	3
Nikil	4
Ravi	5

Borrower 5

Queries

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

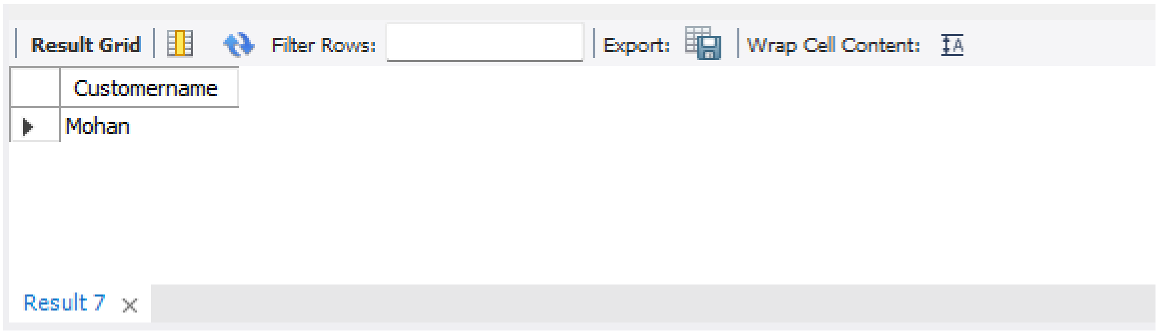
  select d.Customername from branch b, Depositer d, BankAccount ba where b.Branch_city='Delhi' and d.Accno=ba.Accno and b.Branch_name=ba.Branch_name group by d.Customername having count(distinct b.Branch_name)= (select count(distinct b.Branch_name) from branch b where b.Branch_city='Delhi';



```
Customername
Nikil
```

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

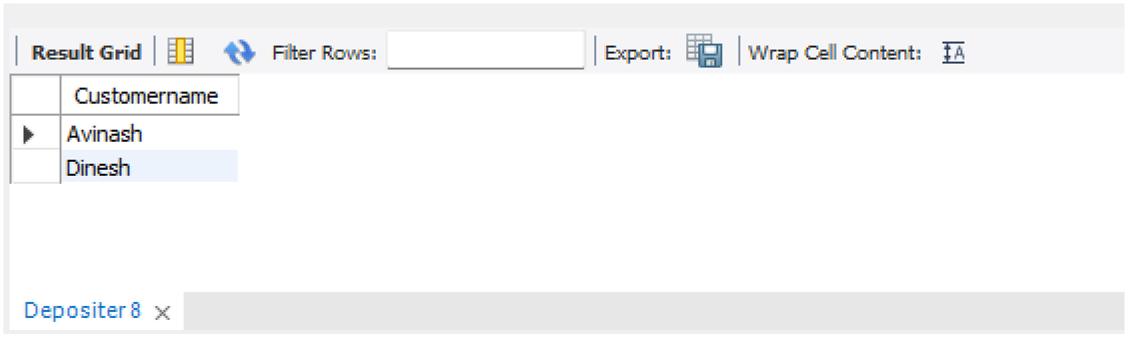
```
select distinct b.Customername from  Borrower b, Depositer d
where b.Customername NOT IN(
        select d.Customername from Loan l,Depositer d, Borrower b
                    where l.Loan_number=b.Loan_number and
                    d.Customername=b.Customername
        );
```



```
Customername
Mohan
```

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

```
select distinct d.Customername from Depositer d
where d.Customername IN(
            select d.Customername from branch br,Depositer d,   BankAccount ba
            where br.Branch_city='Bangalore' and br.Branch_name=ba.Branch_name
          and ba.accno=d.accno and Customername IN(
                    select Customername from Borrower)
);
```



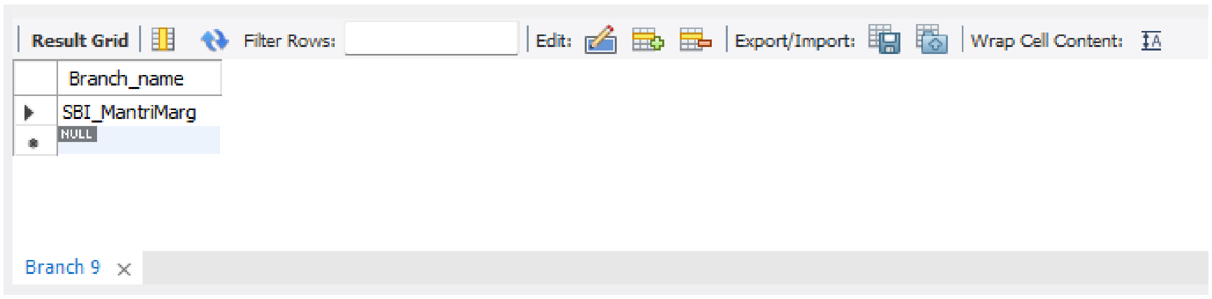
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	Customername
▶	Avinash
	Dinesh

Depositer 8 ×

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

```
select b.Branch_name from Branch b
where b.assets> ALL (
select SUM(b.assets) from Branch b
            where b.Branch_City='Bangalore' );
```



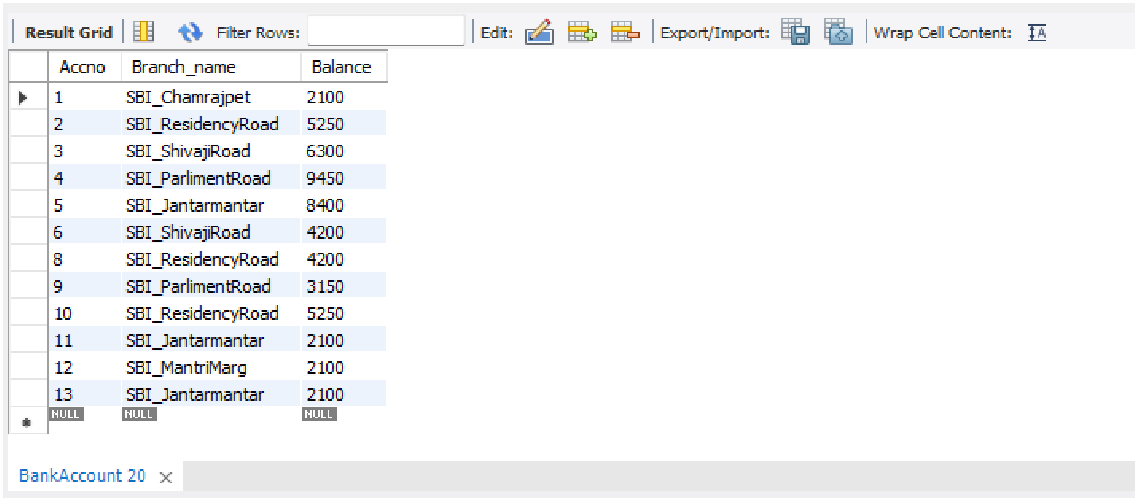
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

	Branch_name
▶	SBI_MantriMarg
*	NULL

Branch 9 ×

- **Update the Balance of all accounts by 5%**

```
UPDATE BankAccount set Balance=(Balance + (Balance*0.05));
```



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

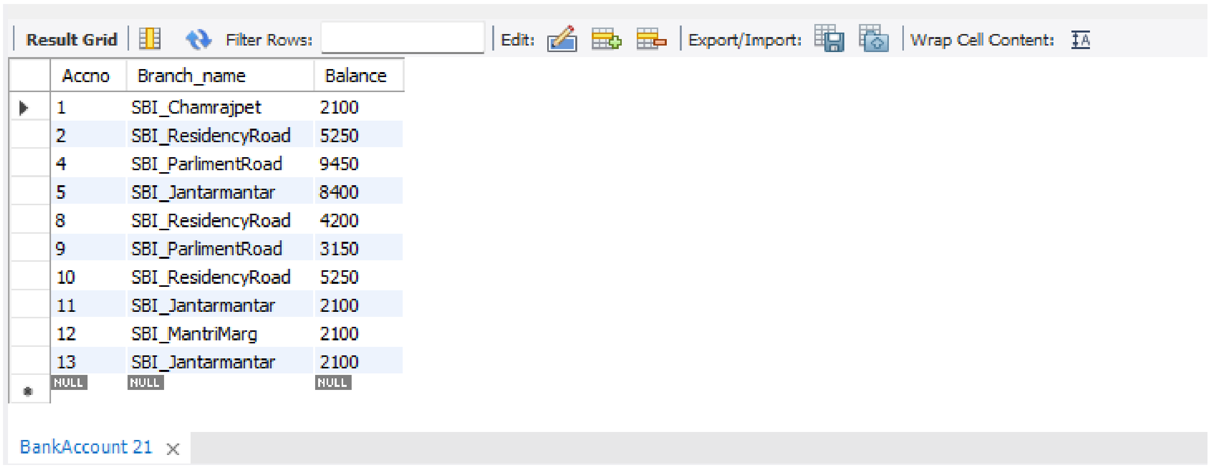
	Accno	Branch_name	Balance
▶	1	SBI_Chamrajpet	2100
	2	SBI_ResidencyRoad	5250
	3	SBI_ShivajiRoad	6300
	4	SBI_ParlimentRoad	9450
	5	SBI_Jantarmantar	8400
	6	SBI_ShivajiRoad	4200
	8	SBI_ResidencyRoad	4200
	9	SBI_ParlimentRoad	3150
	10	SBI_ResidencyRoad	5250
	11	SBI_Jantarmantar	2100
	12	SBI_MantriMarg	2100
	13	SBI_Jantarmantar	2100
*	NULL	NULL	NULL

BankAccount 20 ×

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

  delete ba.* from BankAccount ba, branch b where branch_city='Bombay' and ba.Branch_name=b.Branch_name;

  select * from BankAccount;



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

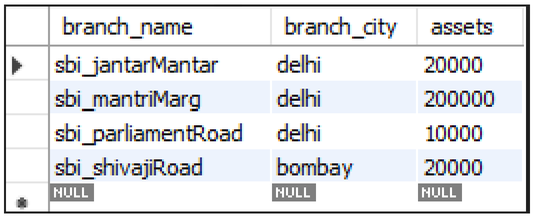
Accno	Branch_name	Balance
1	SBI_Chamrajpet	2100
2	SBI_ResidencyRoad	5250
4	SBI_ParlimentRoad	9450
5	SBI_Jantarmantar	8400
8	SBI_ResidencyRoad	4200
9	SBI_ParlimentRoad	3150
10	SBI_ResidencyRoad	5250
11	SBI_Jantarmantar	2100
12	SBI_MantriMarg	2100
13	SBI_Jantarmantar	2100
NULL	NULL	NULL

BankAccount 21

SPOT QUERY: Demonstrate how to delete all the branches located in Bangalore

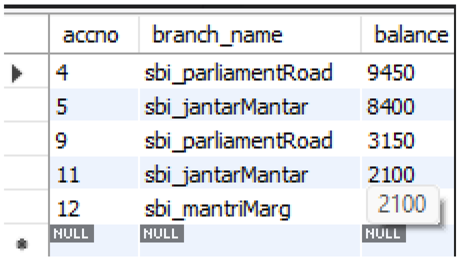
delete b.* from branch b where Branch_city='Bangalore';

select * from branch;



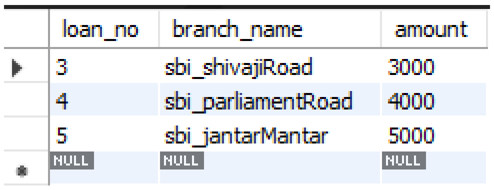
branch_name	branch_city	assets
sbi_jantarMantar	delhi	20000
sbi_mantriMarg	delhi	200000
sbi_parliamentRoad	delhi	10000
sbi_shivajiRoad	bombay	20000
NULL	NULL	NULL

select * from BankAccount;



accno	branch_name	balance
4	sbi_parliamentRoad	9450
5	sbi_jantarMantar	8400
9	sbi_parliamentRoad	3150
11	sbi_jantarMantar	2100
12	sbi_mantriMarg	2100
NULL	NULL	NULL

select * from Loan;



loan_no	branch_name	amount
3	sbi_shivajiRoad	3000
4	sbi_parliamentRoad	4000
5	sbi_jantarMantar	5000
NULL	NULL	NULL

Employee Database

Question

(Week 5)

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

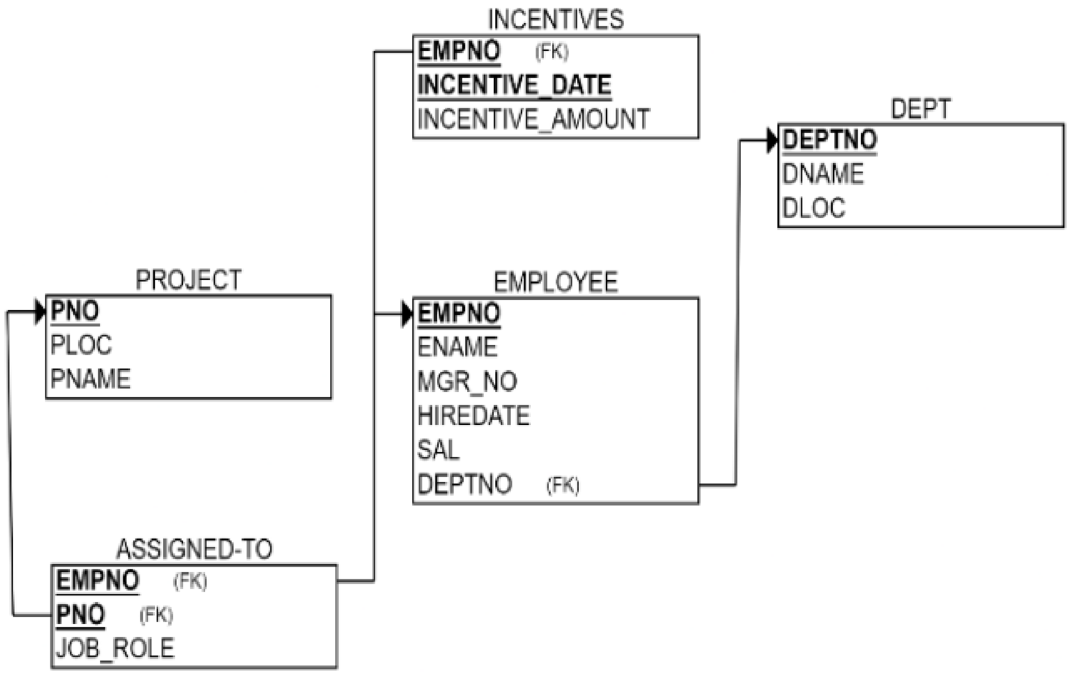
2. Enter greater than five tuples for each table.

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

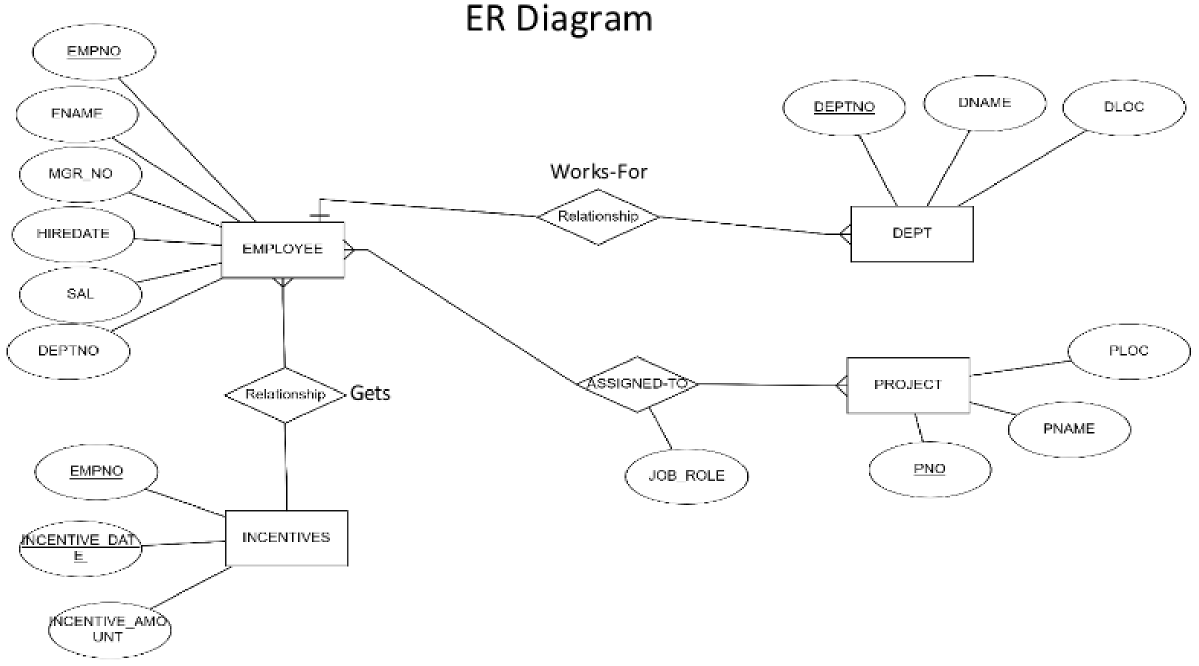
4. Get Employee ID's of those employees who didn't receive incentives

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



ER Diagram



Create database

create database dhiksha_employee;

use dhiksha_employee;

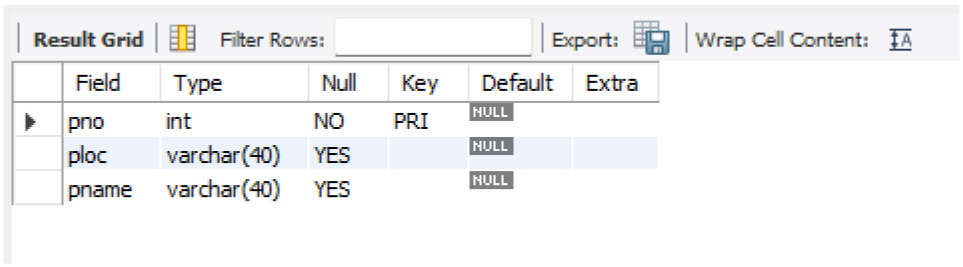
Create table

```
create table dhiksha_employee.project(
pno int,
ploc varchar(40),
pname varchar(40),
PRIMARY KEY(pno)
);
create table dhiksha_employee.dept(
deptno int,
dname varchar(40),
dloc varchar(40),
PRIMARY KEY(deptno)
);
```

```
create table dhiksha_employee.employee(
empno int,
ename varchar(40),
mgr_no int,
hiredate date,
sal int,
deptno int,
primary key (empno),
foreign key (deptno) references dept(deptno)
);
create table dhiksha_employee.incentives(
empno int,
incentive_date date,
incentive_amount int,
primary key(incentive_date),
foreign key (empno) references employee(empno)
);
create table dhiksha_employee.assigned_to(
empno int,
pno int,
job_role varchar(50),
foreign key (pno) references project(pno),
foreign key (empno) references employee(empno)
);
```

Structure of the table

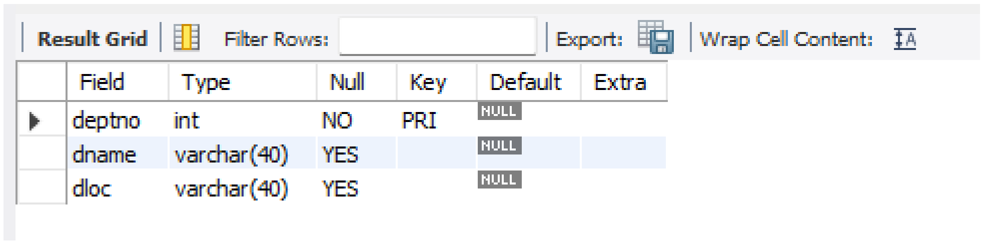
desc project;



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

Field	Type	Null	Key	Default	Extra
pno	int	NO	PRI	NULL	
ploc	varchar(40)	YES		NULL	
pname	varchar(40)	YES		NULL	

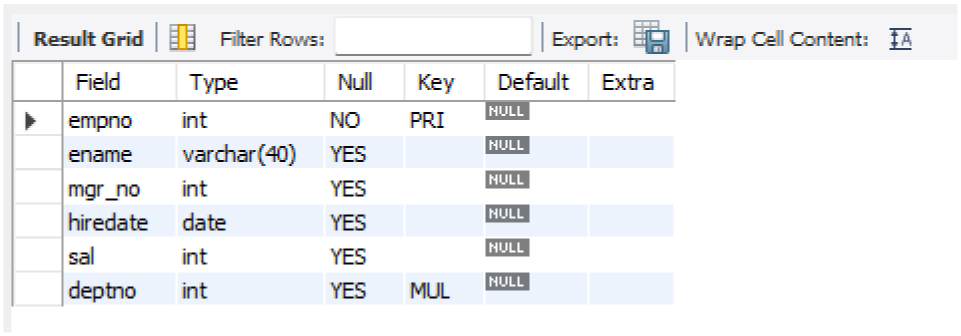
desc dept;



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

Field	Type	Null	Key	Default	Extra
deptno	int	NO	PRI	NULL	
dname	varchar(40)	YES		NULL	
dloc	varchar(40)	YES		NULL	

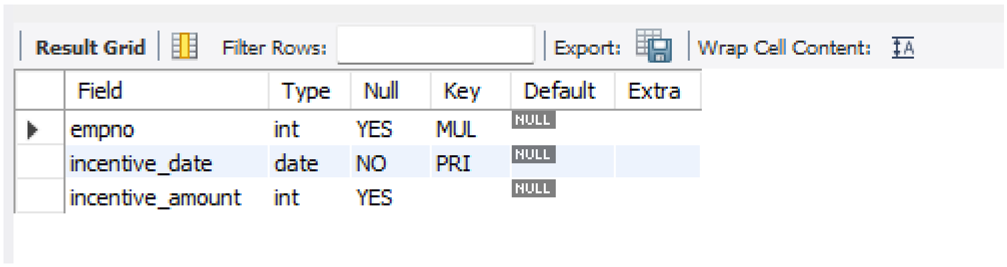
desc employee;



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

Field	Type	Null	Key	Default	Extra
empno	int	NO	PRI	NULL	
ename	varchar(40)	YES		NULL	
mgr_no	int	YES		NULL	
hiredate	date	YES		NULL	
sal	int	YES		NULL	
deptno	int	YES	MUL	NULL	

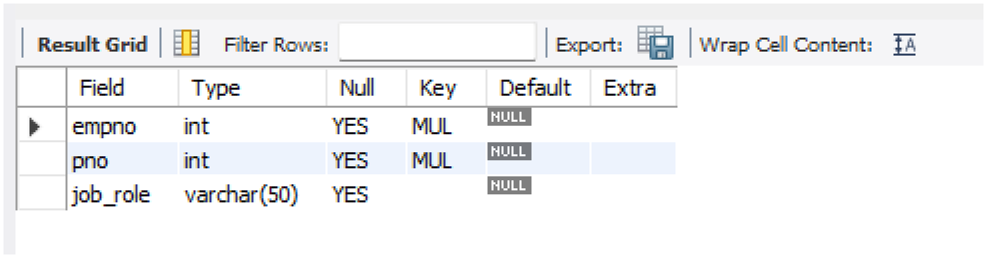
desc incentives;



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

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

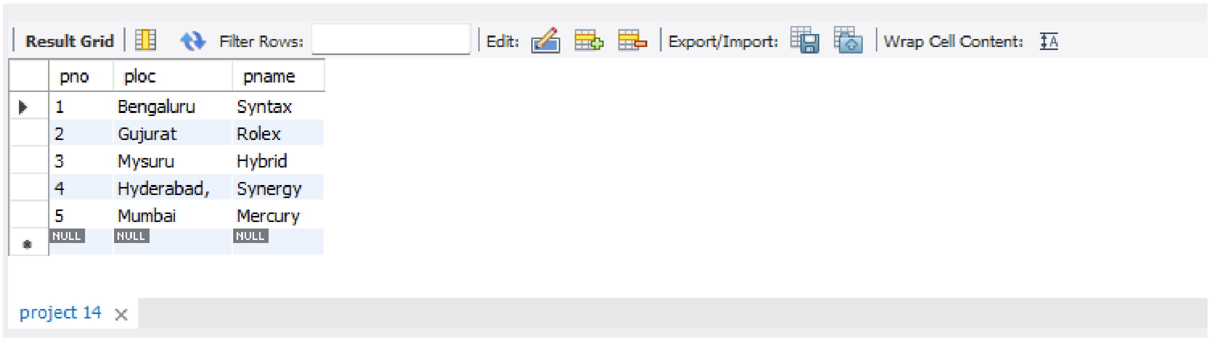
desc assigned_to;



Field	Type	Null	Key	Default	Extra
empno	int	YES	MUL	NULL	
pno	int	YES	MUL	NULL	
job_role	varchar(50)	YES		NULL	

Inserting Values to the table

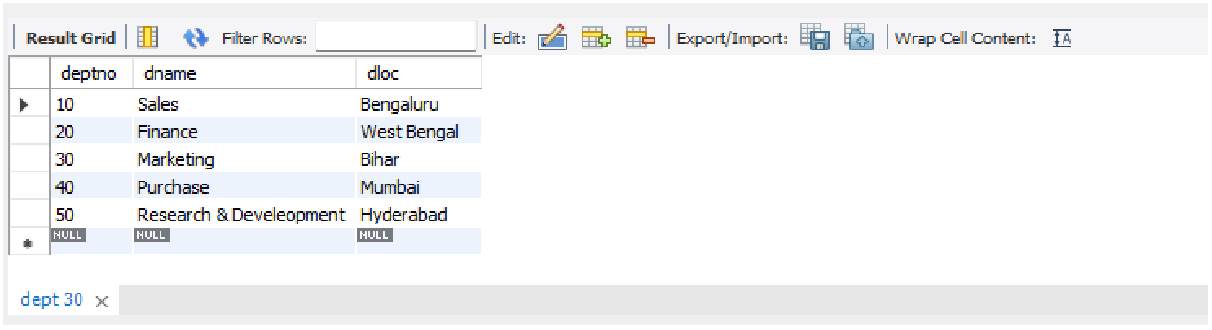
```
insert into project values(1,"Bengaluru","Syntax");
insert into project values(2,"Gujurat","Rolex");
insert into project values(3,"Mysuru","Hybrid");
insert into project values(4,"Hyderabad,","Synergy");
insert into project values(5,"Mumbai","Mercury");
select * from project;
```



pno	ploc	pname
1	Bengaluru	Syntax
2	Gujurat	Rolex
3	Mysuru	Hybrid
4	Hyderabad,	Synergy
5	Mumbai	Mercury
NULL	NULL	NULL

project 14

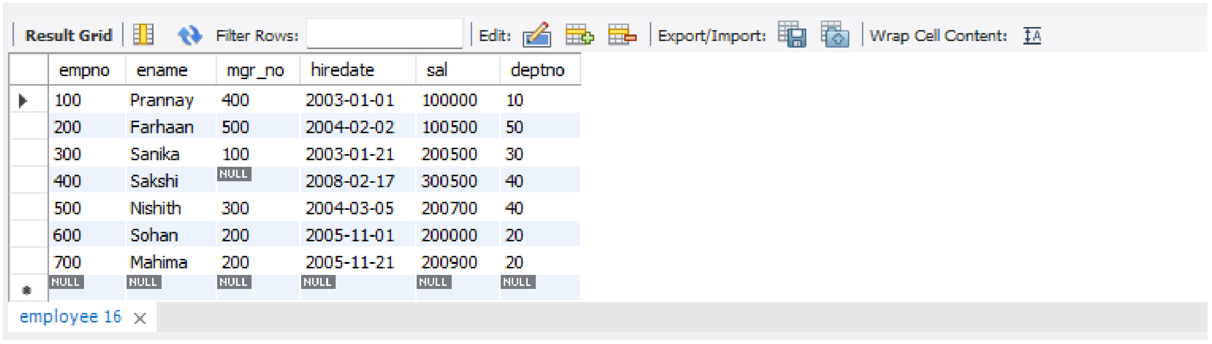
```
insert into dept values(10,"Sales","Bengaluru");
insert into dept values(20,"Finance","West Bengal");
insert into dept values(30,"Marketing","Bihar");
insert into dept values(40,"Purchase","Mumbai");
insert into dept values(50,"Research & Develeopment","Hyderabad");
select * from dept;
```



	deptno	dname	dloc
▶	10	Sales	Bengaluru
	20	Finance	West Bengal
	30	Marketing	Bihar
	40	Purchase	Mumbai
	50	Research & Develeopment	Hyderabad
*	NULL	NULL	NULL

dept 30 ×

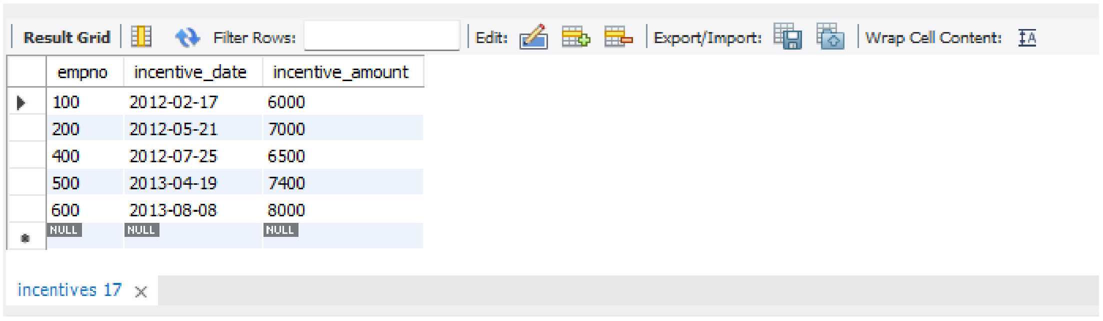
```
insert into employee values(100,"Prannay",400,'2003-01-01',100000,10);
insert into employee values(200,"Farhaan",500,'2004-02-02',100500,50);
insert into employee values(300,"Sanika",100,'2003-01-21',200500,30);
insert into employee values(400,"Sakshi", NULL ,'2008-02-17',300500,40);
insert into employee values(500,"Nishith",300,'2004-03-05',200700,40);
insert into employee values(600,"Sohan",200,'2005-11-01',200000,20);
insert into employee values(700,"Mahima",200,'2005-11-21',200900,20);
select * from employee;
```



	empno	ename	mgr_no	hiredate	sal	deptno
▶	100	Prannay	400	2003-01-01	100000	10
	200	Farhaan	500	2004-02-02	100500	50
	300	Sanika	100	2003-01-21	200500	30
	400	Sakshi	NULL	2008-02-17	300500	40
	500	Nishith	300	2004-03-05	200700	40
	600	Sohan	200	2005-11-01	200000	20
	700	Mahima	200	2005-11-21	200900	20
*	NULL	NULL	NULL	NULL	NULL	NULL

employee 16 ×

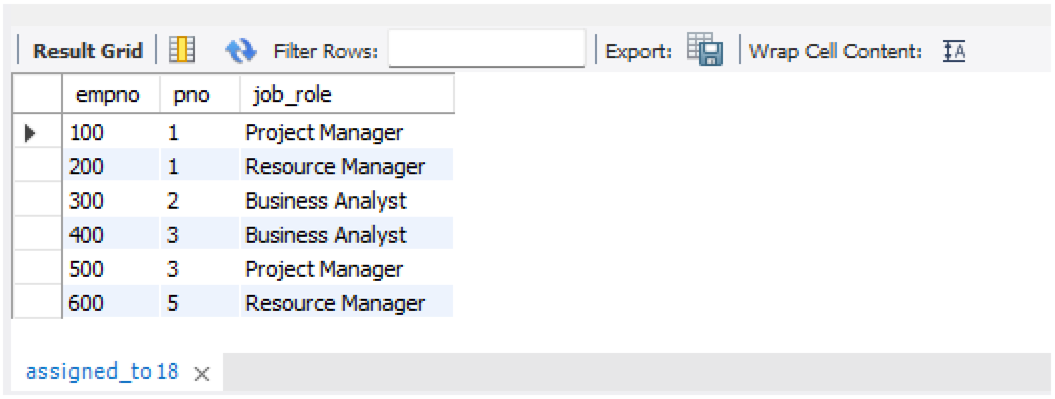
```
insert into incentives values(100,'2012-02-17',6000);
insert into incentives values(200,'2012-05-21',7000);
insert into incentives values(400,'2012-07-25',6500);
insert into incentives values(500,'2013-04-19',7400);
insert into incentives values(600,'2013-08-08',8000);
select * from incentives;
```



	empno	incentive_date	incentive_amount
▶	100	2012-02-17	6000
	200	2012-05-21	7000
	400	2012-07-25	6500
	500	2013-04-19	7400
	600	2013-08-08	8000
*	NULL	NULL	NULL

incentives 17 ×

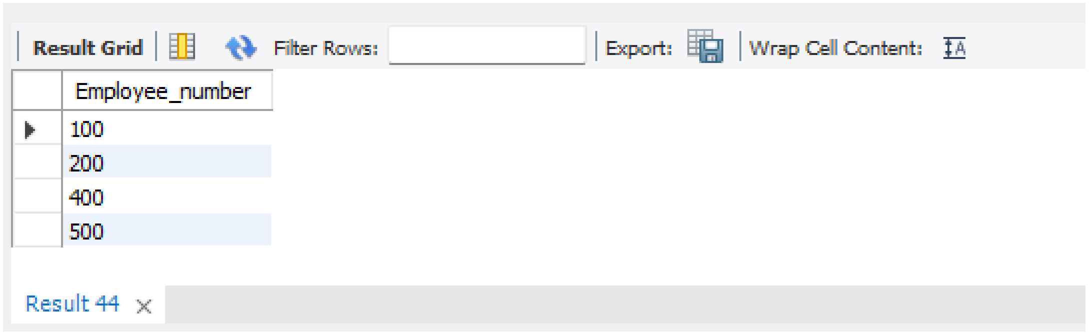
```
insert into assigned_to values(100,1, "Project Manager");
insert into assigned_to values(200,1, "Resource Manager");
insert into assigned_to values(300,2, "Business Analyst");
insert into assigned_to values(400,3, "Business Analyst");
insert into assigned_to values(500,3, "Project Manager");
insert into assigned_to values(600,5, "Resource Manager");
select * from assigned_to;
```



empno	pno	job_role
100	1	Project Manager
200	1	Resource Manager
300	2	Business Analyst
400	3	Business Analyst
500	3	Project Manager
600	5	Resource Manager

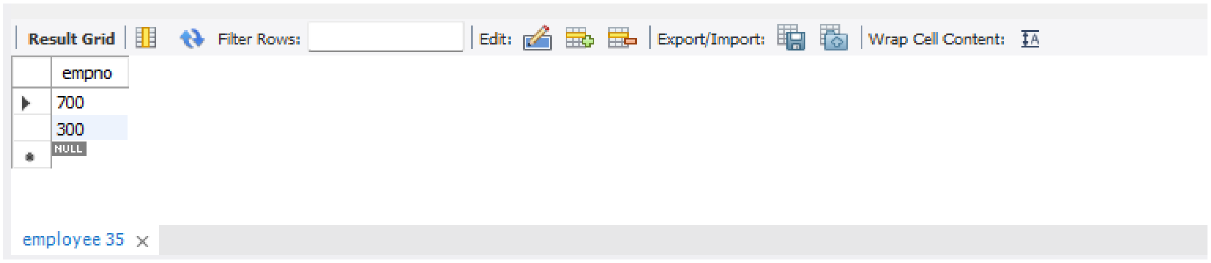
Queries

- **Retrieve the employee numbers of all employees who work on project located in Bengaluru, Hyderabad, or Mysuru.**
  ```
  select a.empno Employee_number from project p, assigned_to a
  where p.pno=a.pno and p.ploc in("Hyderabad","Bengaluru","Mysuru");
  ```



Employee_number
100
200
400
500

- **Get Employee ID's of those employees who didn't receive incentives**
  ```
  select e.empno from  employee e
  where e.empno NOT IN
              (select i.empno from incentives i);
  ```

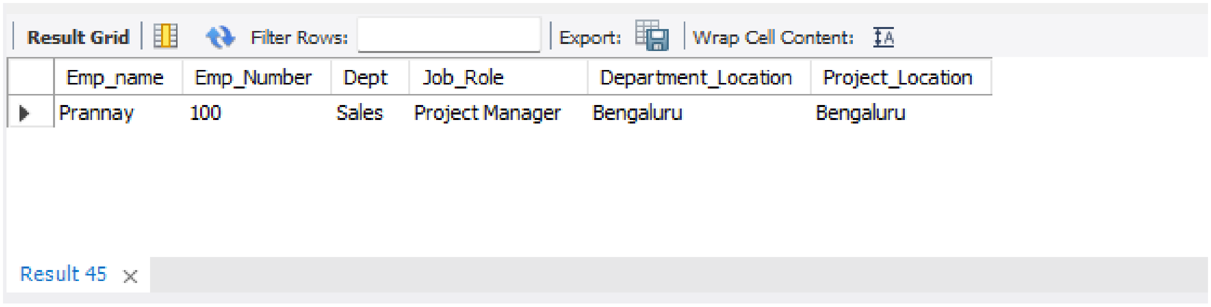


Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

	empno
▶	700
	300
*	NULL

employee 35 ×

- **Write a SQL query to find the employees name, number, dept, job_role, department location and project location who are working for a project location same as his/her department location.**
  select e.ename Emp_name, e.empno Emp_Number, d.dname Dept, a.job_role Job_Role, d.dloc Department_Location, p.ploc Project_Location
  from project p, dept d, employee e, assigned_to a
  where e.empno=a.empno and p.pno=a.pno and e.deptno=d.deptno and p.ploc=d.dloc;



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

	Emp_name	Emp_Number	Dept	Job_Role	Department_Location	Project_Location
▶	Prannay	100	Sales	Project Manager	Bengaluru	Bengaluru

Result 45 ×

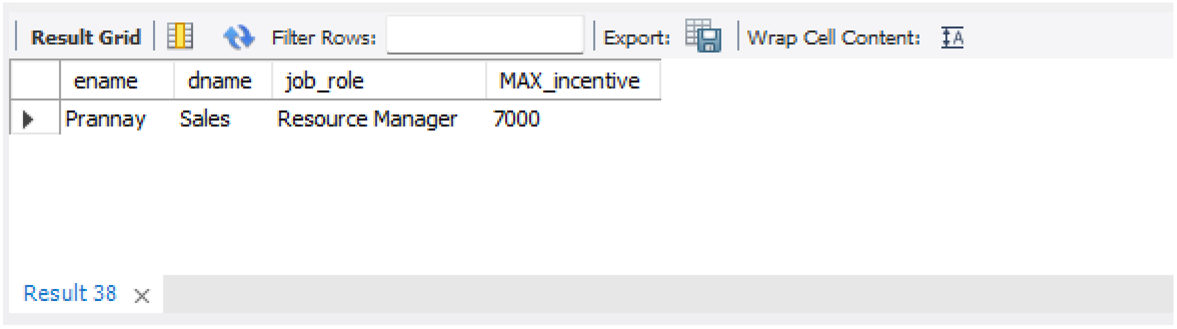
SPOT QUERY:

Find the employee name, dept name and job role of an employee who received maximum incentive in year 2012

select e.ename, d.dname, a.job_role, MAX(i.incentive_amount)   MAX_incentive

from employee e, dept d, incentives i, assigned_to a

where incentive_date between '2012-01-01' and '2012-12-31';



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

	ename	dname	job_role	MAX_incentive
▶	Prannay	Sales	Resource Manager	7000

Result 38 ×

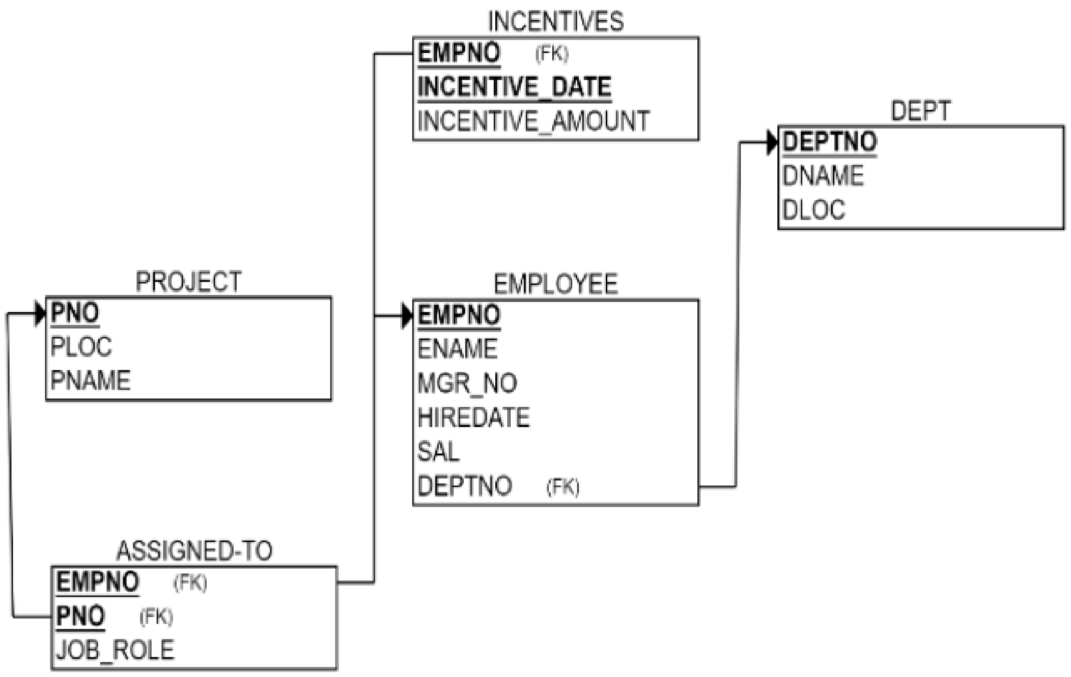
More Queries on Employee Database

Question

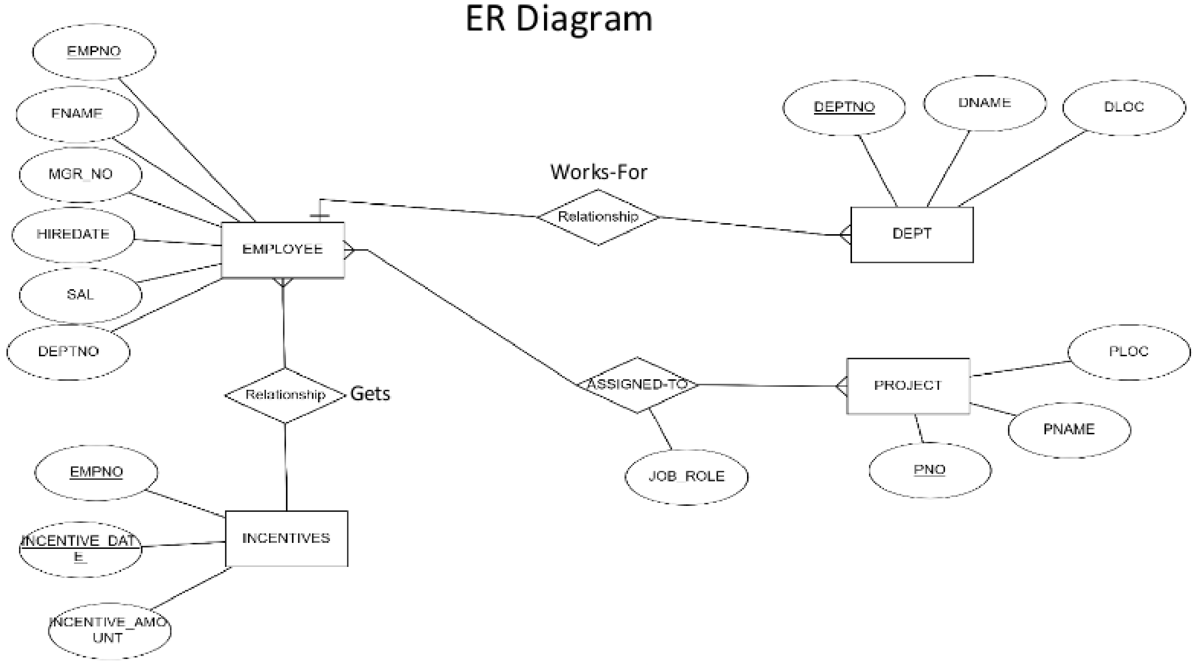
(Week 6)

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 the second maximum incentive in January 2019.
7. Display those employees who are working in the same department where his the manager is working.

Schema Diagram



ER Diagram



Create database

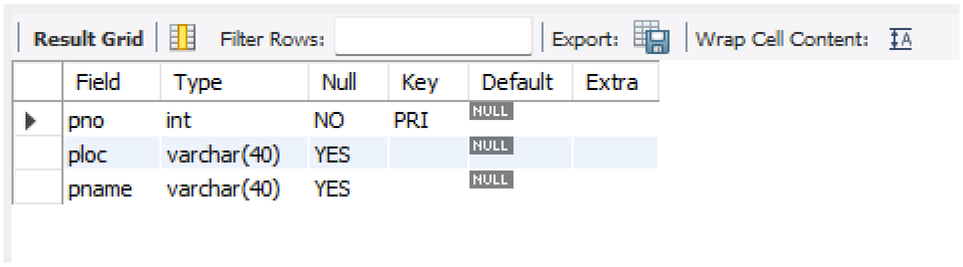
```
create database dhiksha_employee;
use dhiksha_employee;
```

Create table

```
create table dhiksha_employee.project(
pno int,
ploc varchar(40),
pname varchar(40),
PRIMARY KEY(pno)
);
create table dhiksha_employee.dept(
deptno int,
dname varchar(40),
dloc varchar(40),
PRIMARY KEY(deptno)
);
create table dhiksha_employee.employee(
empno int,
ename varchar(40),
mgr_no int,
hiredate date,
sal int,
deptno int,
primary key (empno),
foreign key (deptno) references dept(deptno)
);
create table dhiksha_employee.incentives(
empno int,
incentive_date date,
incentive_amount int,
primary key(incentive_date),
foreign key (empno) references employee(empno)
);
create table dhiksha_employee.assigned_to(
empno int,
pno int,
job_role varchar(50),
foreign key (pno) references project(pno),
foreign key (empno) references employee(empno)
);
```

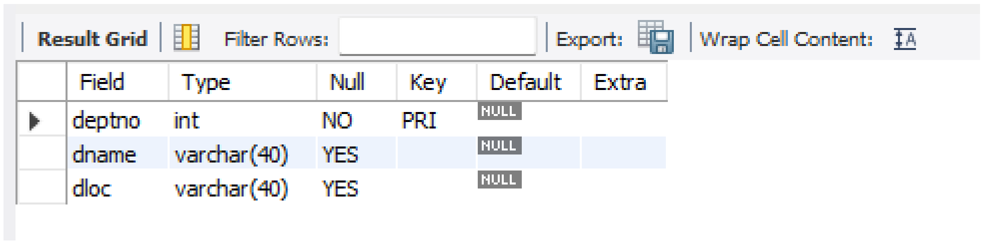
Structure of the table

desc project;



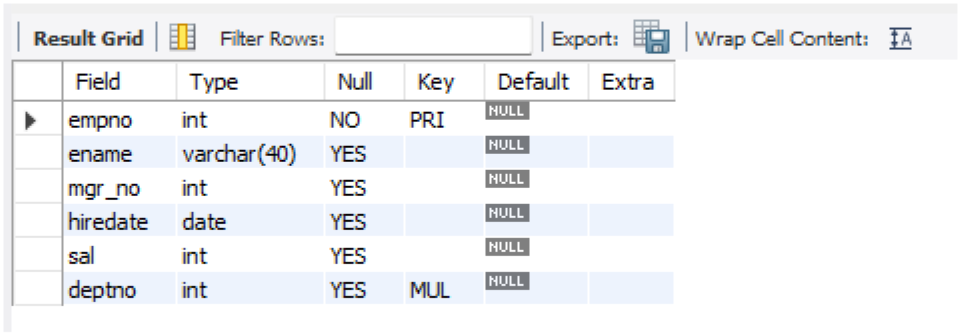
Field	Type	Null	Key	Default	Extra
pno	int	NO	PRI	NULL	
ploc	varchar(40)	YES		NULL	
pname	varchar(40)	YES		NULL	

desc dept;



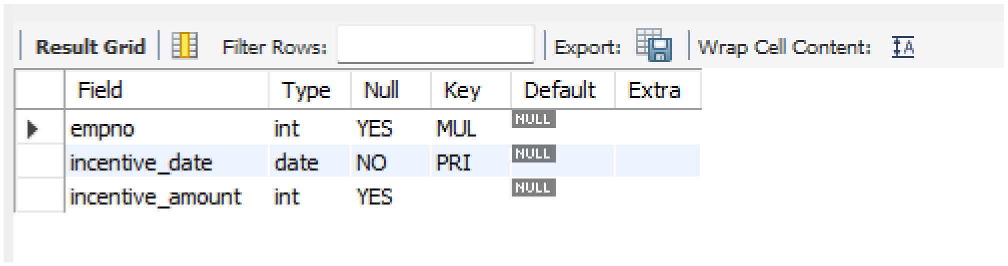
Field	Type	Null	Key	Default	Extra
deptno	int	NO	PRI	NULL	
dname	varchar(40)	YES		NULL	
dloc	varchar(40)	YES		NULL	

desc employee;



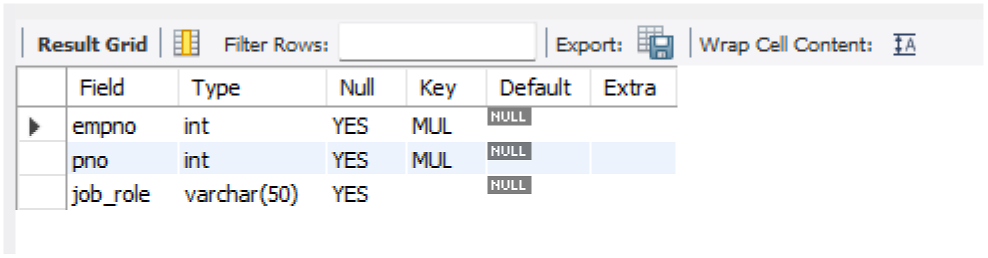
	Field	Type	Null	Key	Default	Extra
▶	empno	int	NO	PRI	NULL	
	ename	varchar(40)	YES		NULL	
	mgr_no	int	YES		NULL	
	hiredate	date	YES		NULL	
	sal	int	YES		NULL	
	deptno	int	YES	MUL	NULL	

desc incentives;



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

desc assigned_to;



	Field	Type	Null	Key	Default	Extra
▶	empno	int	YES	MUL	NULL	
	pno	int	YES	MUL	NULL	
	job_role	varchar(50)	YES		NULL	

Inserting Values to the table

insert into project values(1,"Bengaluru","Syntax");

insert into project values(2,"Gujurat","Rolex");

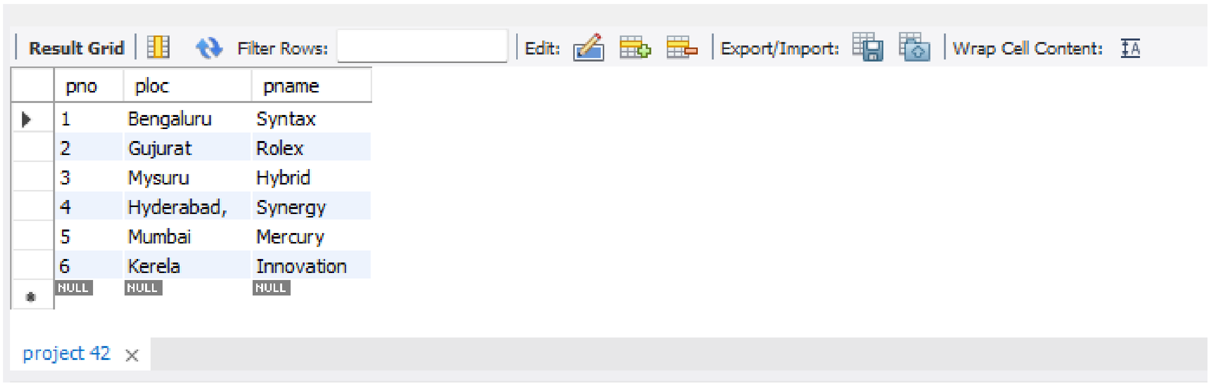
insert into project values(3,"Mysuru","Hybrid");

insert into project values(4,"Hyderabad,","Synergy");

insert into project values(5,"Mumbai","Mercury");

insert into project values(6,"Kerela","Innovation");

select * from project;


Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

pno	ploc	pname
1	Bengaluru	Syntax
2	Gujurat	Rolex
3	Mysuru	Hybrid
4	Hyderabad,	Synergy
5	Mumbai	Mercury
6	Kerela	Innovation
NULL	NULL	NULL

project 42

insert into dept values(10,"Sales","Bengaluru");

insert into dept values(20,"Finance","West Bengal");

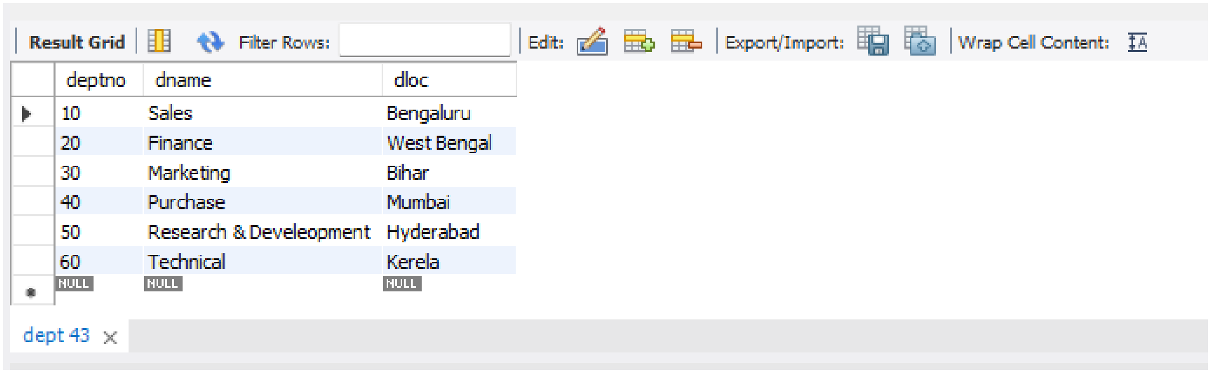
insert into dept values(30,"Marketing","Bihar");

insert into dept values(40,"Purchase","Mumbai");

insert into dept values(50,"Research & Development" ,"Hyderabad");

insert into dept values(60,"Technical","Kerela");

select * from dept;


Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

deptno	dname	dloc
10	Sales	Bengaluru
20	Finance	West Bengal
30	Marketing	Bihar
40	Purchase	Mumbai
50	Research & Develeopment	Hyderabad
60	Technical	Kerela
NULL	NULL	NULL

dept 43

insert into employee values(100,"Prannay",700,'2003-01-01',24000,10);

insert into employee values(200,"Farhaan",100,'2004-02-02',17000,50);

insert into employee values(300,"Sanika",100,'2003-01-21',9000,30);

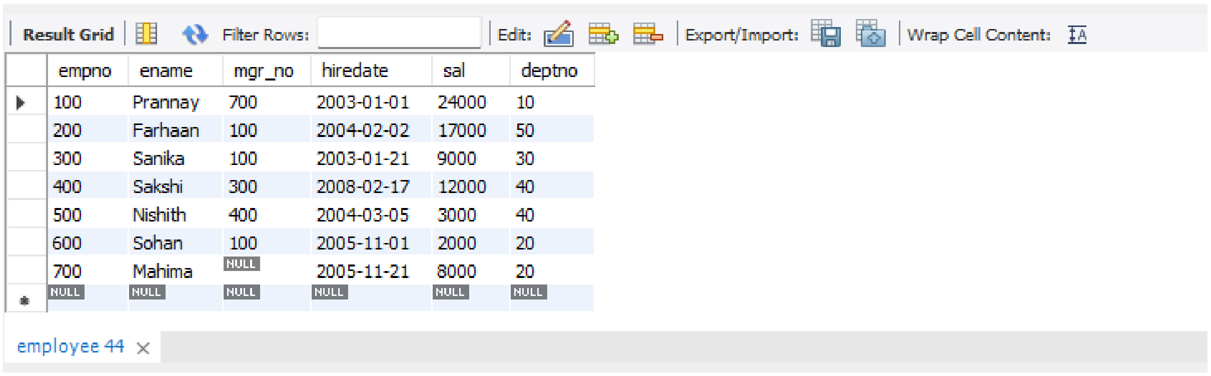
insert into employee values(400,"Sakshi", 300 ,'2008-02-17',12000,40);

insert into employee values(500,"Nishith",400,'2004-03-05',3000,40);

insert into employee values(600,"Sohan",100,'2005-11-01',2000,20);

insert into employee values(700,"Mahima",NULL,'2005-11-21',8000,20);

select * from employee;



empno	ename	mgr_no	hiredate	sal	deptno
100	Prannay	700	2003-01-01	24000	10
200	Farhaan	100	2004-02-02	17000	50
300	Sanika	100	2003-01-21	9000	30
400	Sakshi	300	2008-02-17	12000	40
500	Nishith	400	2004-03-05	3000	40
600	Sohan	100	2005-11-01	2000	20
700	Mahima	NULL	2005-11-21	8000	20
NULL	NULL	NULL	NULL	NULL	NULL

insert into incentives values(100,'2019-02-17',6000);

insert into incentives values(200,'2019-05-21',7000);

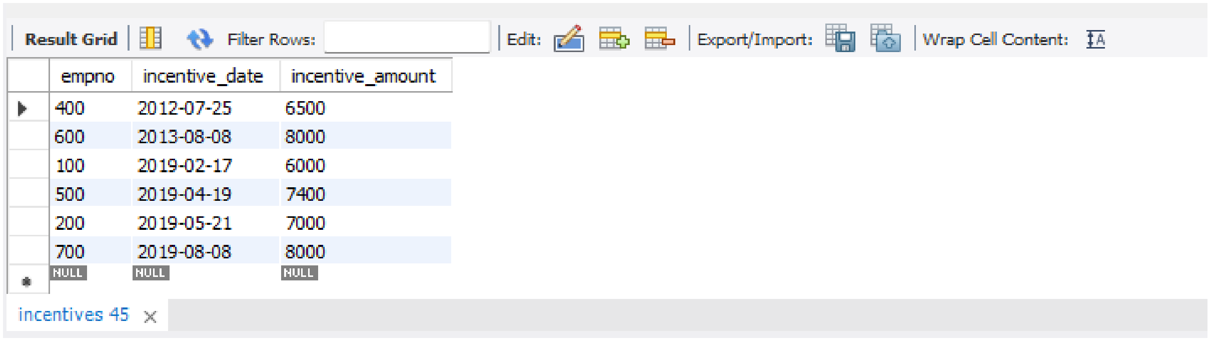
insert into incentives values(400,'2012-07-25',6500);

insert into incentives values(500,'2019-04-19',7400);

insert into incentives values(600,'2013-08-08',8000);

insert into incentives values(700,'2019-08-08',8000);

select * from incentives;



empno	incentive_date	incentive_amount
400	2012-07-25	6500
600	2013-08-08	8000
100	2019-02-17	6000
500	2019-04-19	7400
200	2019-05-21	7000
700	2019-08-08	8000
NULL	NULL	NULL

insert into assigned_to values(100,1, "Project Manager");

insert into assigned_to values(200,1, "Resource Manager");

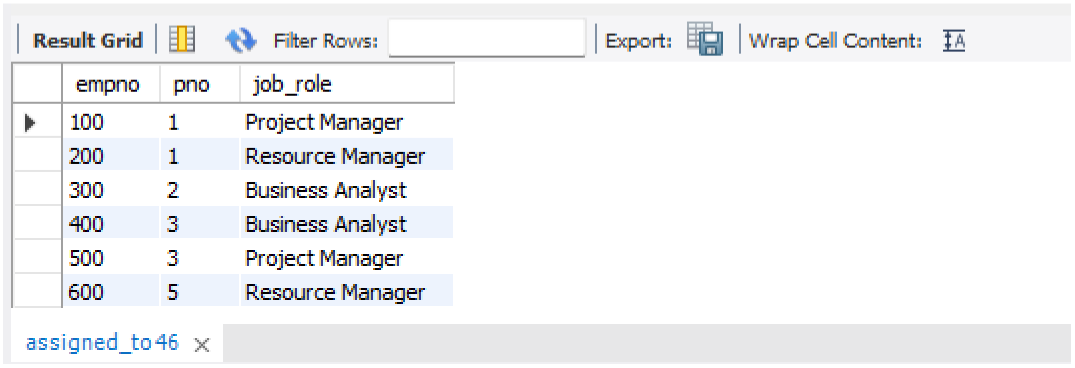
insert into assigned_to values(300,2, "Business Analyst");

insert into assigned_to values(400,3, "Business Analyst");

insert into assigned_to values(500,3, "Project Manager");

insert into assigned_to values(600,5, "Resource Manager");

select * from assigned_to;



empno	pno	job_role
100	1	Project Manager
200	1	Resource Manager
300	2	Business Analyst
400	3	Business Analyst
500	3	Project Manager
600	5	Resource Manager

assigned_to 46

Queries

- **List the name of the managers with the maximum employees**

select e1.ename

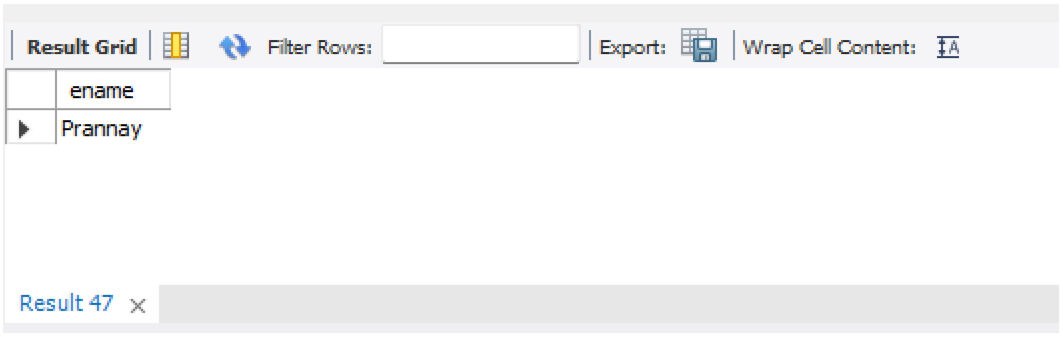
from employee e1, employee e2

where e1.empno=e2.mgr_no group by e1.ename

having count(e1.mgr_no)=(select count(e1.ename)

from employee e1, employee e2 where e1.empno=e2.mgr_no

group by e1.ename order by count(e1.ename) desc limit 1);

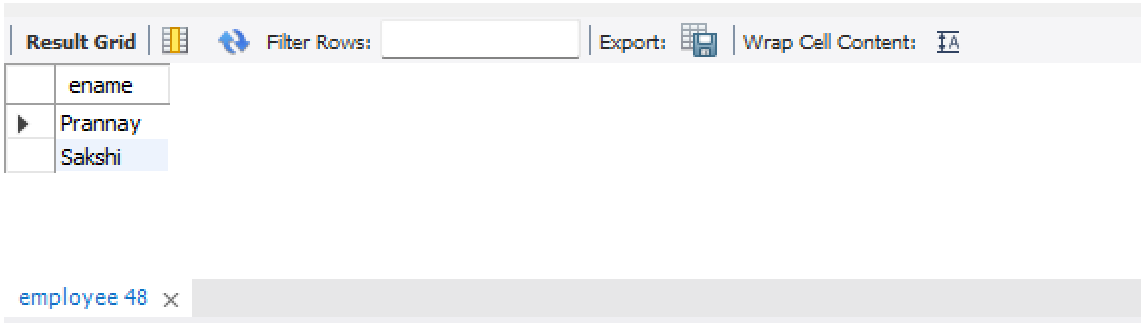


ename
Prannay

Result 47

- **Display those managers name whose salary is more than average salary of his employee**

```
select m.ename from employee m
where m.empno in
   (select mgr_no  from employee)
    and m.sal>(select avg(n.sal)  from employee n
    where n.mgr_no=m.empno);
```



```
ename
Prannay
Sakshi
```

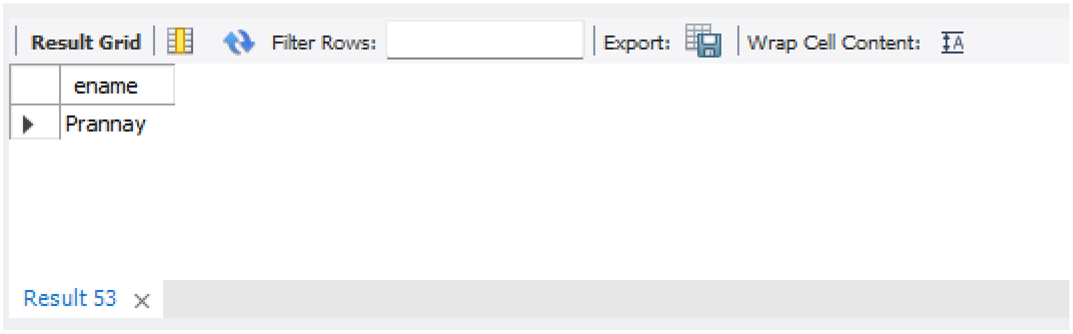
- **Find the name of the second top level managers of each department.**

select ename from employee where empno in(select distinct mgr_no

from employee where empno in

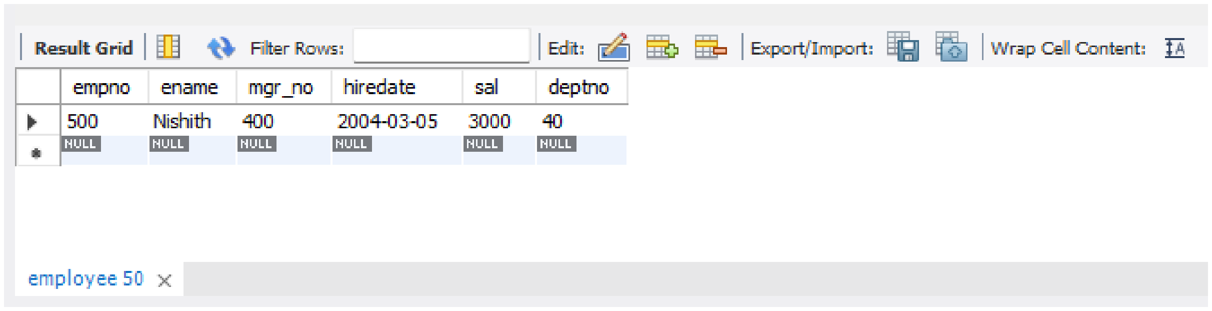
            (select distinct mgr_no from employee where empno in

                        (select distinct mgr_no from employee)));



```
ename
Prannay
```

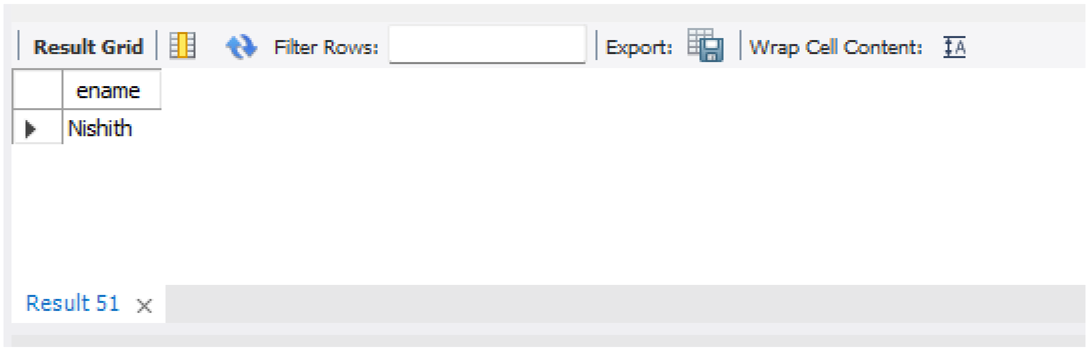
- **Find the employee details who got second maximum incentive in January 2019.**
  select * from employee where empno=
  (select i.empno from incentives i
  where i.incentive_amount= (select max(n.incentive_amount) from incentives n
  where n.incentive_amount<(select max(inc.incentive_amount) from incentives inc
  where inc.incentive_date between '2019-01-01' and '2019-12-31') and incentive_date
  between '2019-01-01' and '2019-12-31'));


Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

empno	ename	mgr_no	hiredate	sal	deptno
500	Nishith	400	2004-03-05	3000	40
NULL	NULL	NULL	NULL	NULL	NULL

employee 50

- **Display those employees who are working in the same department where his manager is working.**
  select e2.ename
  from employee e1, employee e2
  where e1.empno=e2.mgr_no and e1.deptno=e2.deptno;


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

ename
Nishith

Result 51

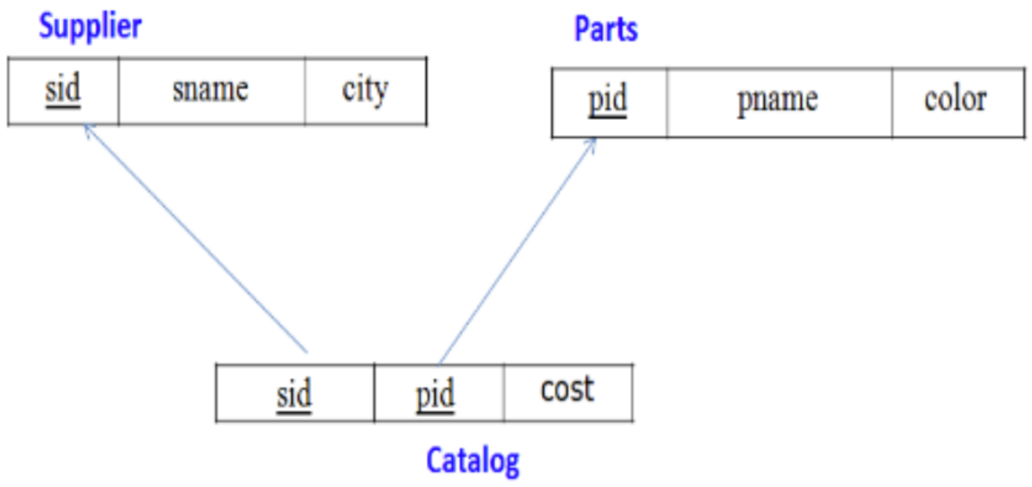
Supplier Database

Question

(Week 7)

1. Using Scheme diagram, Create tables by properly specifying the primary keys and the foreign keys.
2. Insert appropriate records in each table.
3. Find the pnames of parts for which there is some supplier.
4. Find the snames of suppliers who supply every part.
5. Find the snames of suppliers who supply every red part.
6. Find the pnames of parts supplied by Acme Widget Suppliers and by no one else.
7. Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over all the suppliers who supply that part).
8. For each part, find the sname of the supplier who charges the most for that part.

Schema Diagram



Create database

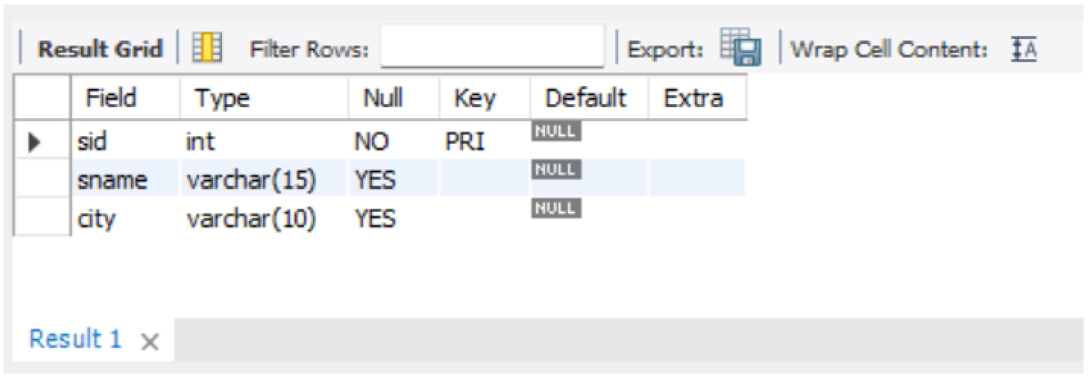
```
create database dhiksha_supplier;
use dhiksha_supplier;
```

Create table

```
create table dhiksha_supplier.Supplier(
sid int,
sname varchar(15),
city varchar(10),
PRIMARY KEY(sid)
);
create table dhiksha_supplier.Parts(
pid int,
pname varchar(10),
color varchar(5),
PRIMARY KEY(pid)
);
create table dhiksha_supplier.Catalog(
sid int,
pid int,
cost int,
PRIMARY KEY(sid, pid),
FOREIGN KEY(sid) REFERENCES Supplier(sid),
FOREIGN KEY(pid) REFERENCES Parts(pid)
);
```

Structure of the table

desc Supplier;

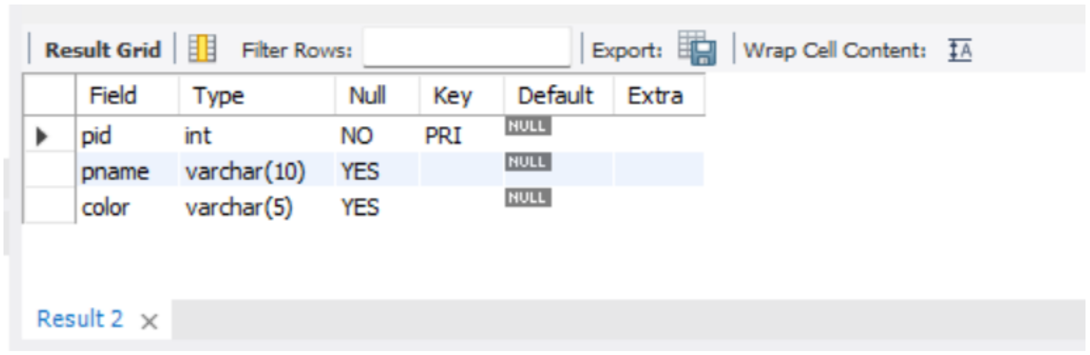


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
sid	int	NO	PRI	NULL	
sname	varchar(15)	YES		NULL	
city	varchar(10)	YES		NULL	

Result 1

desc Parts;

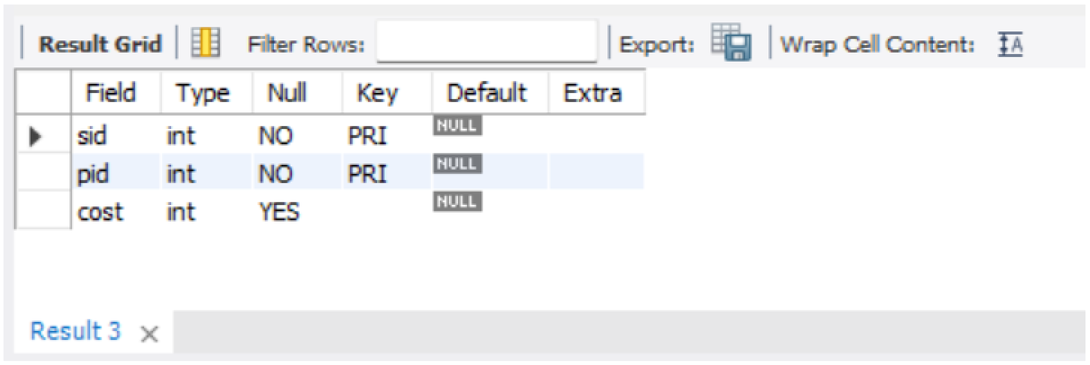


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
pid	int	NO	PRI	NULL	
pname	varchar(10)	YES		NULL	
color	varchar(5)	YES		NULL	

Result 2

desc Catalog;



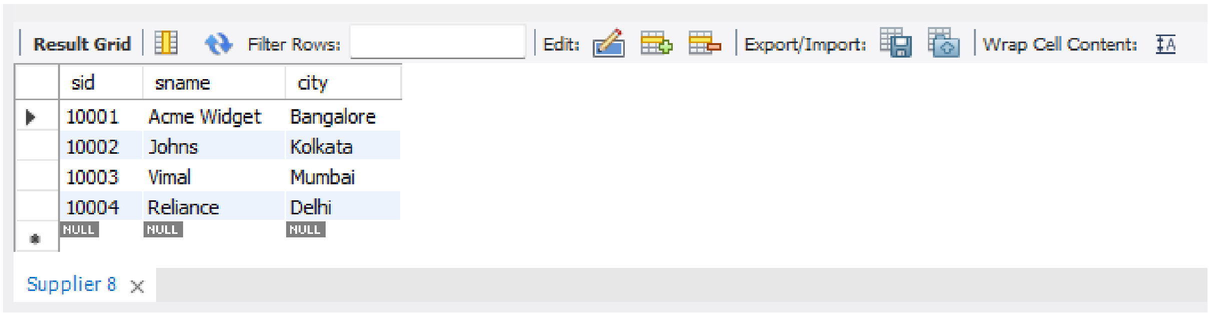
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

Result 3

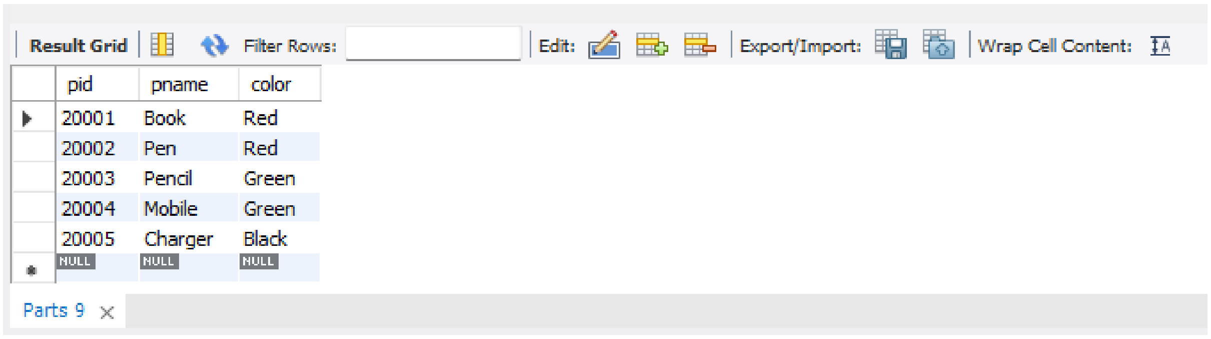
Inserting Values to the table

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

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

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

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

select * from Supplier;
```



sid	sname	city
10001	Acme Widget	Bangalore
10002	Johns	Kolkata
10003	Vimal	Mumbai
10004	Reliance	Delhi
NULL	NULL	NULL

```
insert into Parts values(20001,"Book", "Red");

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

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

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

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

select * from Parts;
```



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

```
insert into Catalog values(10001,20001, 10);

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

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

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

insert into Catalog values(10001,20005, 10);
```

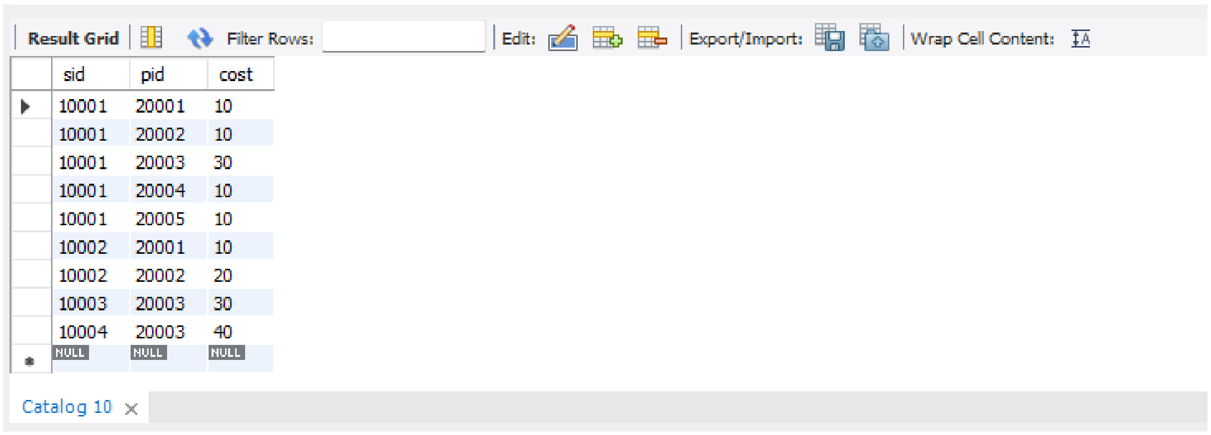
insert into Catalog values(10002,20001, 10);

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

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

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

select * from Catalog;

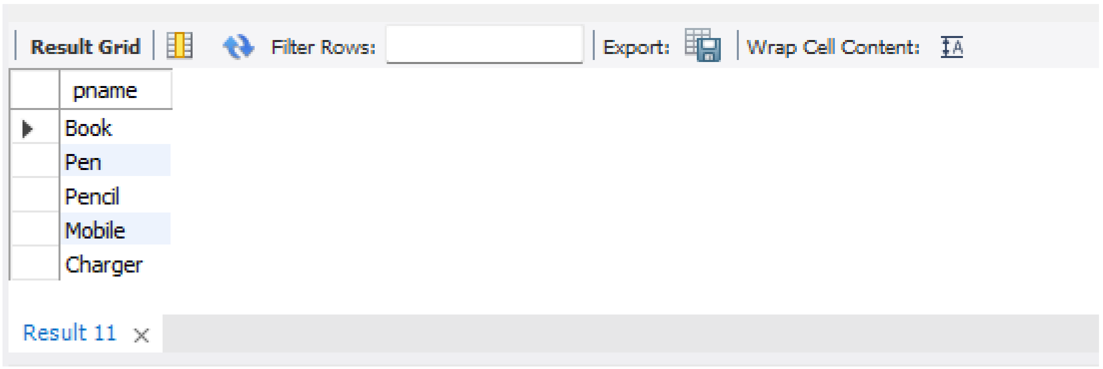

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

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

Catalog 10

Queries

- **Find the pnames of parts for which there is some supplier.**
  select distinct p.pname
  from Parts p, Catalog c
  where p.pid = c.pid;


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

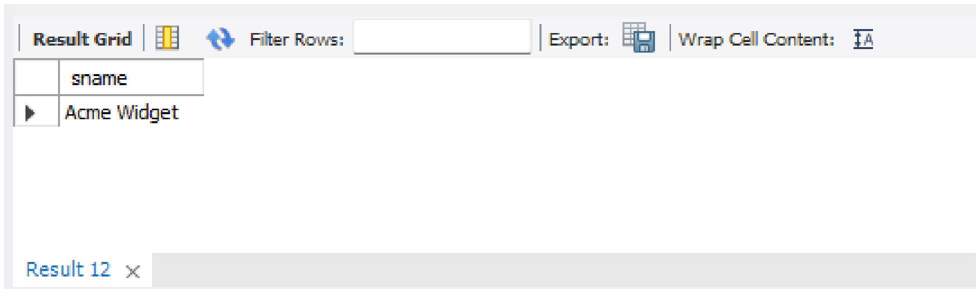
pname
Book
Pen
Pencil
Mobile
Charger

Result 11

- **Find the snames of suppliers who supply every part.**

  select distinct s.sname
  from Catalog C, Supplier s WHERE C.sid=s.sid and NOT EXISTS (select P.pid FROM Parts P

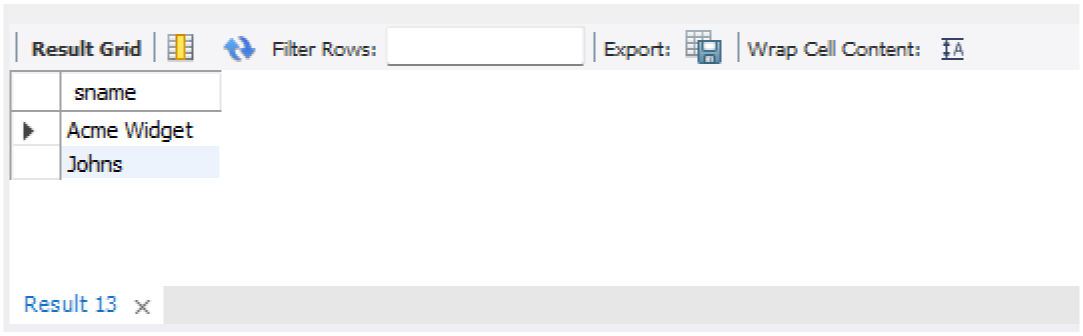
where NOT EXISTS (select C1.sid from Catalog C1
where C1.sid = C.sid and C1.pid = P.pid));



sname
Acme Widget

- **Find the snames of suppliers who supply every red part.**

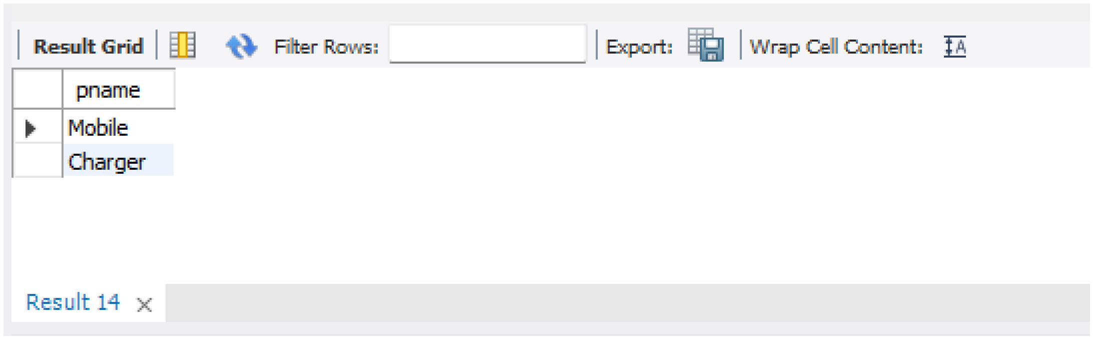
select distinct s.sname
from Catalog C, Supplier s where C.sid=s.sid and NOT EXISTS (select P.pid from Parts P
where P.color="Red" and NOT EXISTS (select C1.sid from Catalog C1
where C1.sid = C.sid and C1.pid = P.pid and P.color="Red"));



sname
Acme Widget
Johns

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

select P.pname
from Parts P, Catalog C, Supplier S
where P.pid = C.pid and C.sid = S.sid and S.sname = "Acme Widget"
and NOT EXISTS (select * from Catalog C1, Supplier S1
where P.pid = C1.pid and C1.sid = S1.sid and
S1.sname != "Acme Widget");

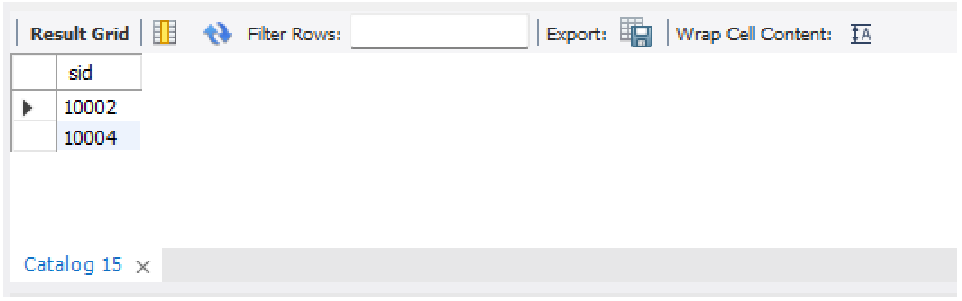

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

pname
Mobile
Charger

Result 14

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

  select distinct C.sid from Catalog C
  where C.cost > (select AVG(C1.cost)
  from Catalog C1 where C1.pid = C.pid);

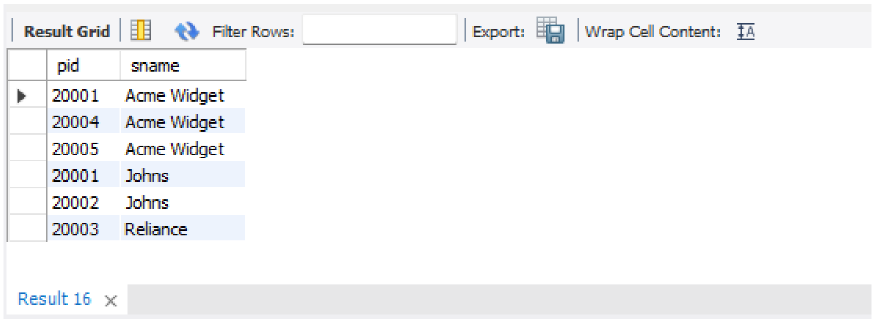

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

sid
10002
10004

Catalog 15

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

  select P.pid, S.sname
  from Parts P, Supplier S, Catalog C
  where C.pid = P.pid and
  C.sid = S.sid and
  C.cost = (select max(C1.cost)
  from Catalog C1
  where C1.pid = P.pid);


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

pid	sname
20001	Acme Widget
20004	Acme Widget
20005	Acme Widget
20001	Johns
20002	Johns
20003	Reliance

Result 16

Airline Flight Database

Question

(Week 8)

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.

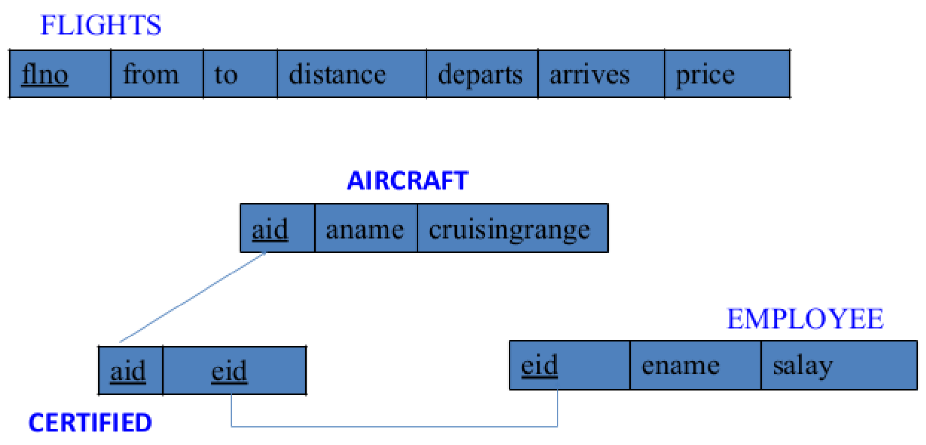
iii. Find the names of pilots whose salary is less than the price of the cheapest route from Bengaluru to Frankfurt.

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

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

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

Schema Diagram



Create database

```
create database flight;
use flight;
```

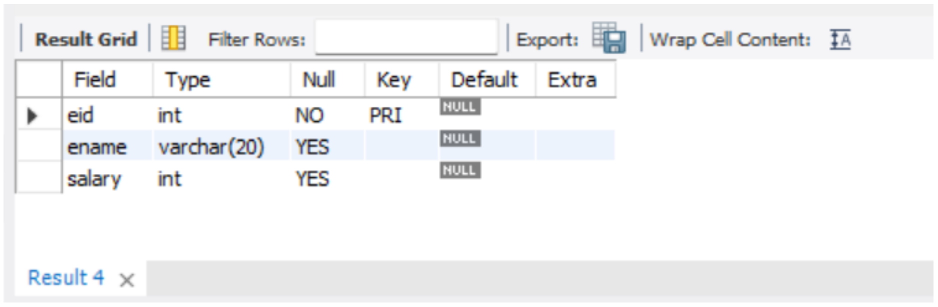
Create table

```
create table flights(
flno int,
from_ varchar(20),
to_ varchar(20),
distance int,
departs time,
arrives time,
price int,
PRIMARY KEY(flno)
);
create table aircraft(
aid int,
aname varchar(20),
cruisingRange int,
PRIMARY KEY(aid)
);
```

```
create table employee(
eid int,
ename varchar(20),
salary int,
PRIMARY KEY(eid)
);
create table certified(
eid int,
aid int,
FOREIGN KEY(eid) REFERENCES employee(eid) on update cascade on delete cascade,
FOREIGN KEY(aid) REFERENCES aircraft(aid) on update cascade on delete cascade
);
```

Structure of the table

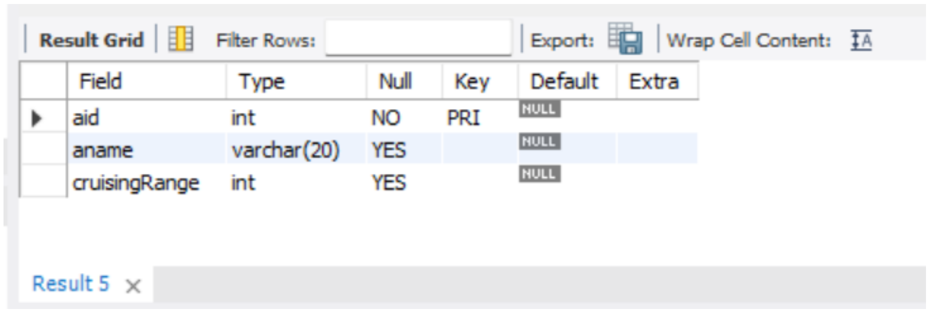
desc employee;



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

Result 4

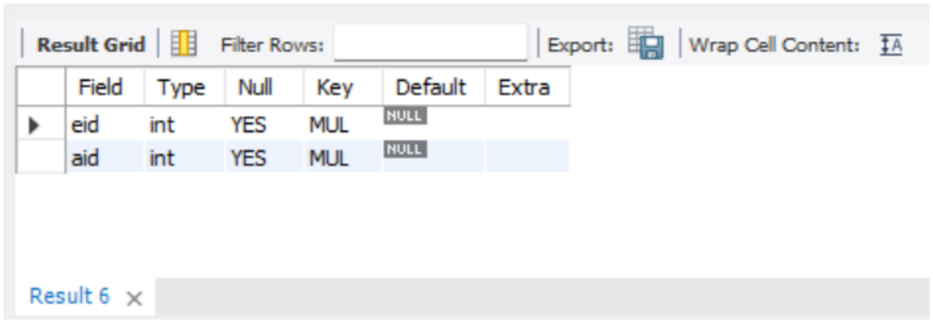
desc aircraft;



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

Result 5

desc certified;

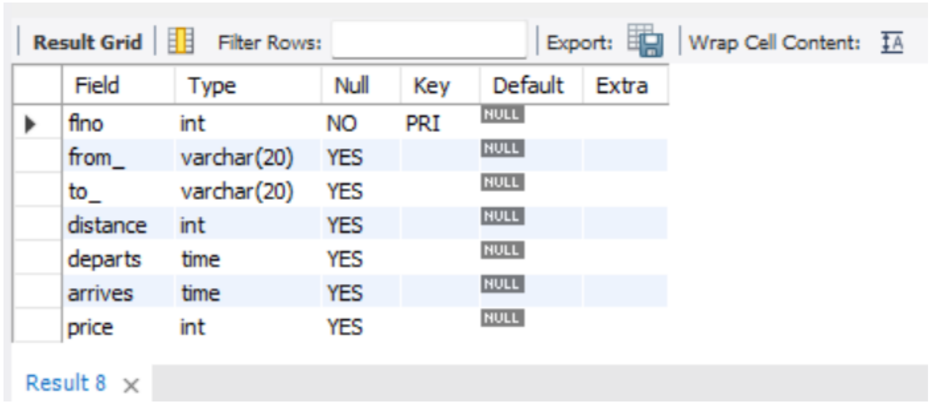


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

Result 6

desc flights;



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

Field	Type	Null	Key	Default	Extra
fino	int	NO	PRI	NULL	
from_	varchar(20)	YES		NULL	
to_	varchar(20)	YES		NULL	
distance	int	YES		NULL	
departs	time	YES		NULL	
arrives	time	YES		NULL	
price	int	YES		NULL	

Result 8

Inserting Values to the table

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

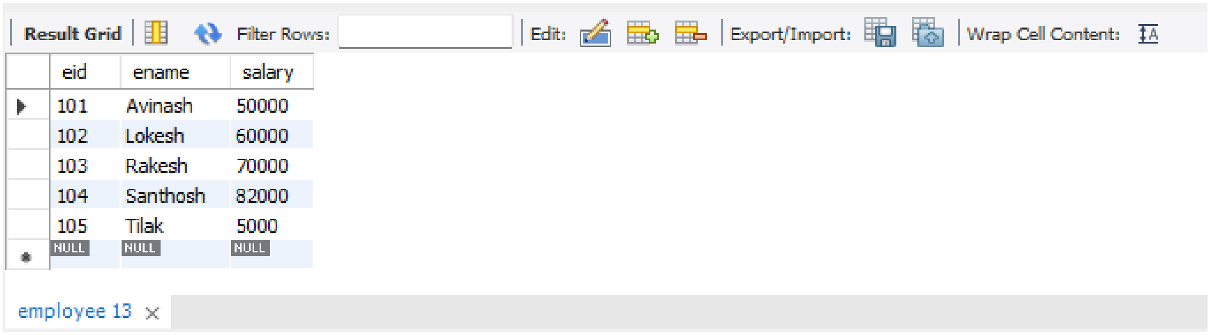
insert into employee values(102,'Lokesh',60000);

insert into employee values(103,'Rakesh',70000);

insert into employee values(104,'Santhosh',82000);

insert into employee values(105,'Tilak',5000);

select * from employee;



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

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

employee 13

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

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

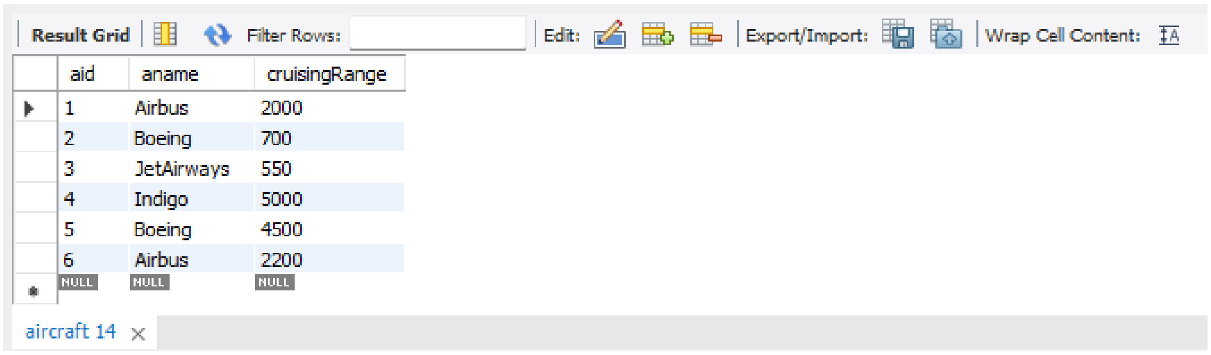
insert into aircraft values(3,'JetAirways',550);

insert into aircraft values(4,'Indigo',5000);

insert into aircraft values(5,'Boeing',4500);

insert into aircraft values(6,'Airbus',2200);

select * from aircraft;



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

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);

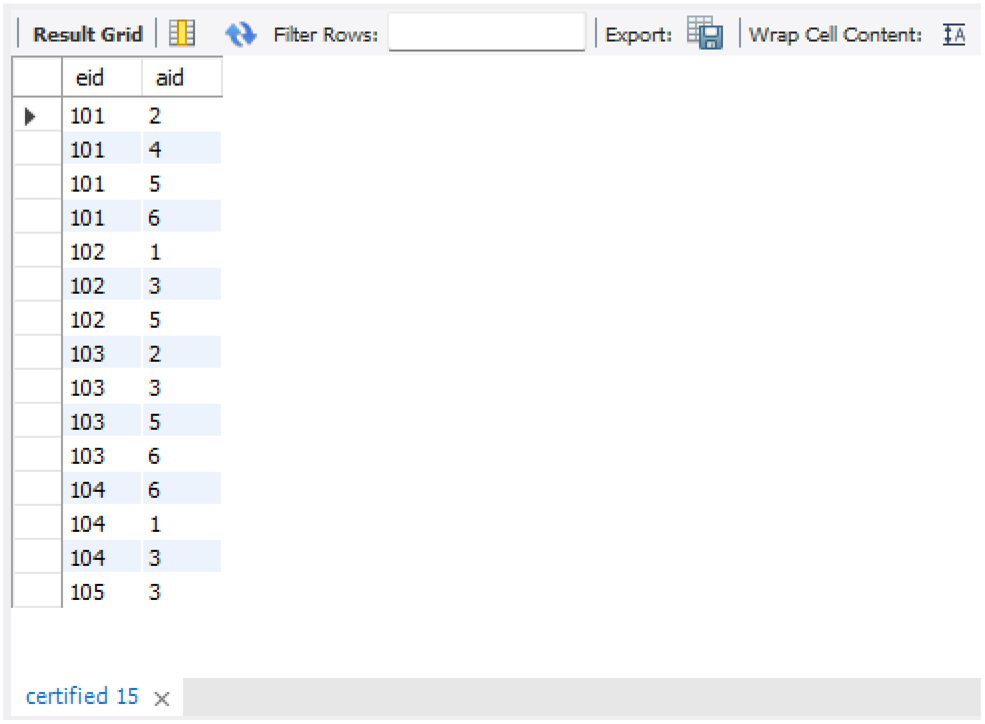
insert into certified values(104,6);

insert into certified values(104,1);

insert into certified values(104,3);

insert into certified values(105,3);

select * from certified;

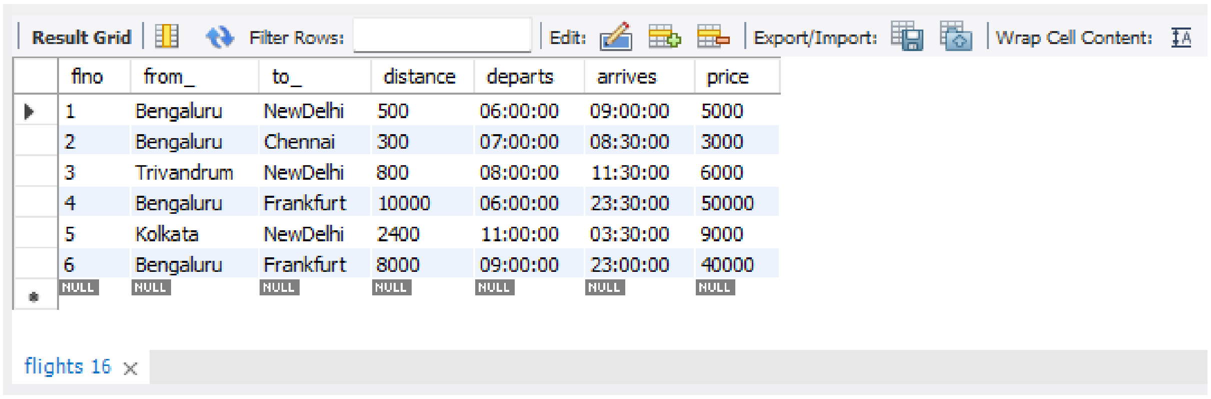


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

eid	aid
101	2
101	4
101	5
101	6
102	1
102	3
102	5
103	2
103	3
103	5
103	6
104	6
104	1
104	3
105	3

certified 15

```
insert into flights values(1,'Bengaluru','NewDelhi',500,'06:00','09:00',5000);
insert into flights values(2,'Bengaluru','Chennai',300,'07:00','08:30',3000);
insert into flights values(3,'Trivandrum','NewDelhi',800,'08:00','11:30',6000);
insert into flights values(4,'Bengaluru','Frankfurt',10000,'06:00','23:30',50000);
insert into flights values(5,'Kolkata','NewDelhi',2400,'11:00','03:30',9000);
insert into flights values(6,'Bengaluru','Frankfurt',8000,'09:00','23:00',40000);
select * from flights;
```



Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

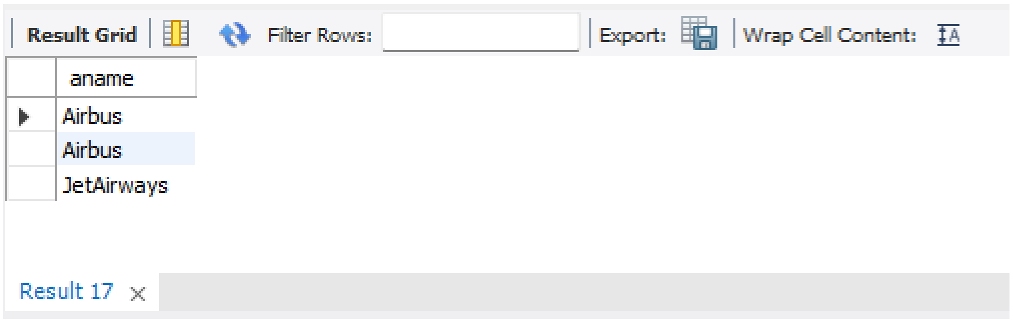
flno	from_	to_	distance	departs	arrives	price
1	Bengaluru	NewDelhi	500	06:00:00	09:00:00	5000
2	Bengaluru	Chennai	300	07:00:00	08:30:00	3000
3	Trivandrum	NewDelhi	800	08:00:00	11:30:00	6000
4	Bengaluru	Frankfurt	10000	06:00:00	23:30:00	50000
5	Kolkata	NewDelhi	2400	11:00:00	03:30:00	9000
6	Bengaluru	Frankfurt	8000	09:00:00	23:00:00	40000
NULL	NULL	NULL	NULL	NULL	NULL	NULL

flights 16

Queries

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

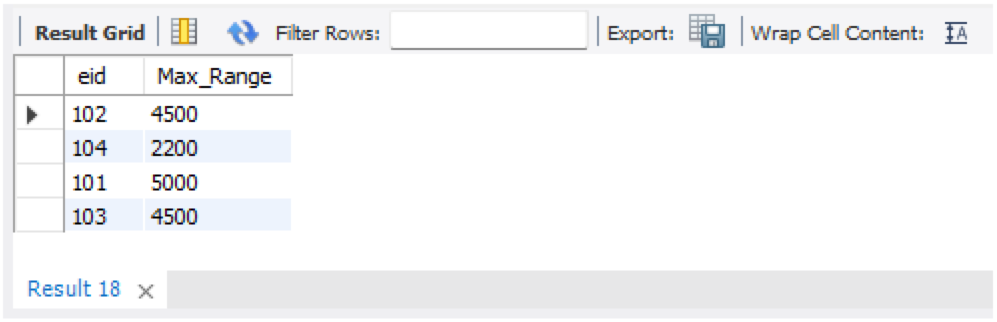
  select (a.aname) from employee e inner join certified c
  on e.eid=c.eid and e.salary>80000 inner join aircraft a on a.aid=c.aid;



aname
Airbus
Airbus
JetAirways

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

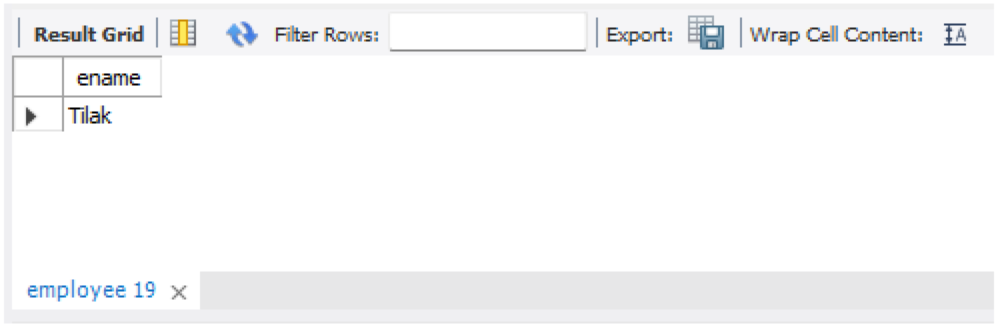
  select c.eid, max(a.cruisingRange) as Max_Range
  from aircraft a, certified c
  where c.aid=a.aid group by c.eid having count(*)>=3;



eid	Max_Range
102	4500
104	2200
101	5000
103	4500

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

  select ename from employee where salary<some(select price from flights where from_='Bengaluru' and to_='Frankfurt');

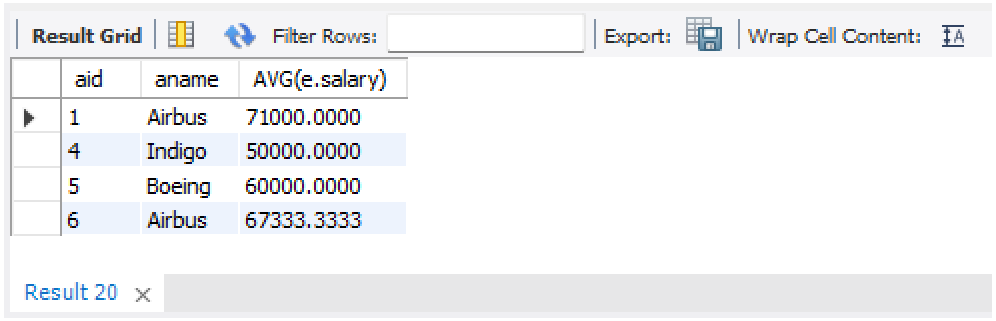

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ename
Tilak

employee 19

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

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

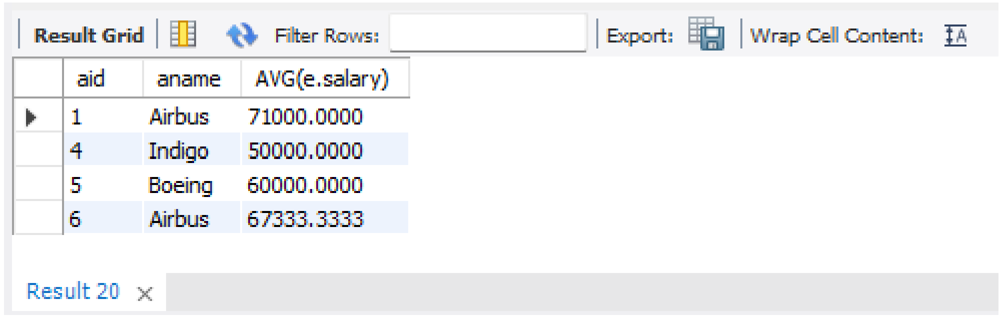

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

Result 20

- **Find the names of pilots certified for some Boeing aircraft.**

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

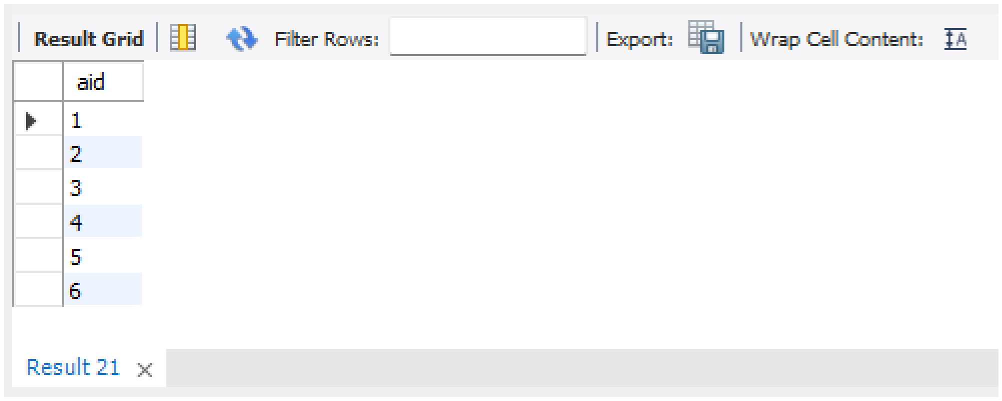

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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

Result 20

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

  select a.aid from flights f, aircraft a where (f.from_='Bengaluru' and f.to_='NewDelhi') and f.distance<=a.cruisingRange ;



aid
1
2
3
4
5
6

NoSQL Lab 1

Question

(Week 9)

Perform the following DB operations using MongoDB.

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

2. Insert appropriate values

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

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

5. Export the created table into local file system

6. Drop the table

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

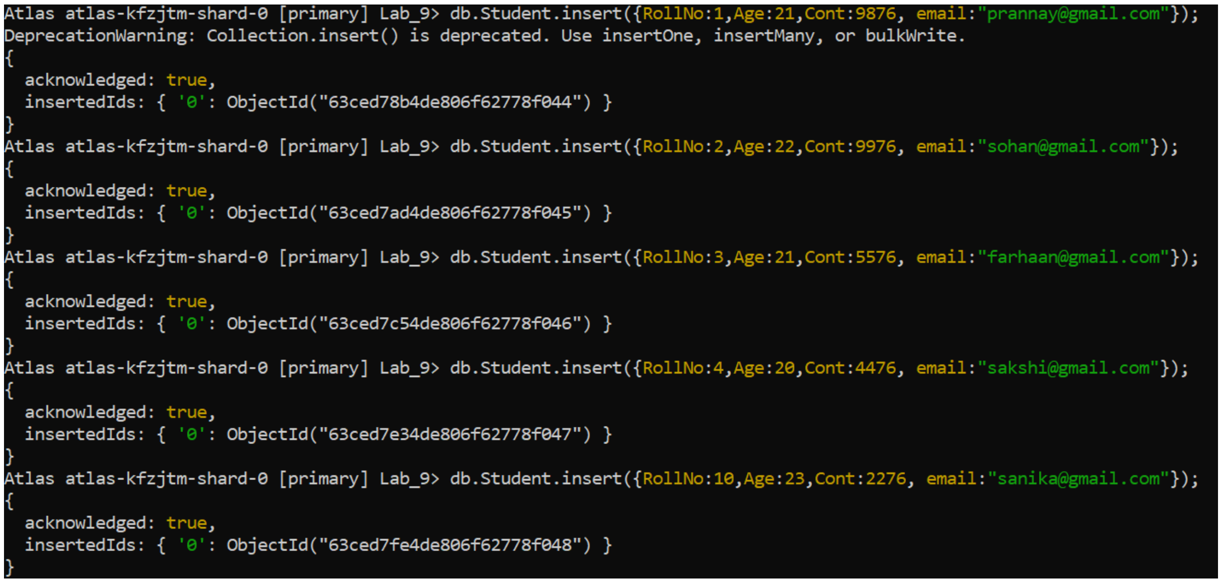
mongodb collection.

Create database

```
db.createCollect(“Student”);
```

Create table & Inserting Values to the table

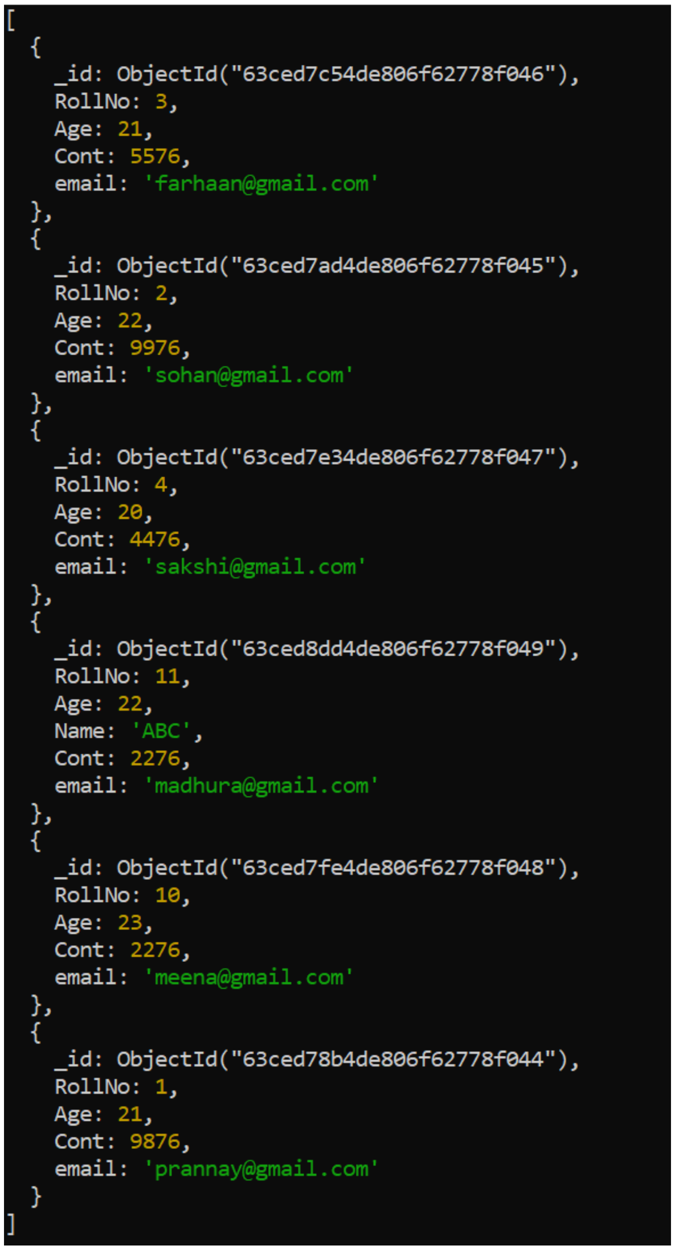
```
db.Student.insert({rollno:1,age:21,cont:9876,email:"prannay@gmail.com"});
db.Student.insert({rollno:2,age:22,cont:9976,email:"sohan@gmail.com"});
db.Student.insert({rollno:3,age:21,cont:5576,email:"farhaan@gmail.com"});
db.Student.insert({rollno:4,age:20,cont:4476,email:"sakshi@gmail.com"});
db.Student.insert({rollno:5,age:23,cont:2276,email:"sanika@gmail.com"});
```



```
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Student.insert({RollNo:1,Age:21,Cont:9876, email:"prannay@gmail.com"});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63ced78b4de806f62778f044") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Student.insert({RollNo:2,Age:22,Cont:9976, email:"sohan@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63ced7ad4de806f62778f045") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Student.insert({RollNo:3,Age:21,Cont:5576, email:"farhaan@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63ced7c54de806f62778f046") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Student.insert({RollNo:4,Age:20,Cont:4476, email:"sakshi@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63ced7e34de806f62778f047") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Student.insert({RollNo:10,Age:23,Cont:2276, email:"sanika@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63ced7fe4de806f62778f048") }
}
```

Structure of the table

db.Student.find();

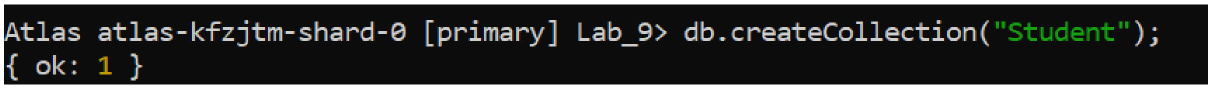


```
[
  {
    _id: ObjectId("63ced7c54de806f62778f046"),
    RollNo: 3,
    Age: 21,
    Cont: 5576,
    email: 'farhaan@gmail.com'
  },
  {
    _id: ObjectId("63ced7ad4de806f62778f045"),
    RollNo: 2,
    Age: 22,
    Cont: 9976,
    email: 'sohan@gmail.com'
  },
  {
    _id: ObjectId("63ced7e34de806f62778f047"),
    RollNo: 4,
    Age: 20,
    Cont: 4476,
    email: 'sakshi@gmail.com'
  },
  {
    _id: ObjectId("63ced8dd4de806f62778f049"),
    RollNo: 11,
    Age: 22,
    Name: 'ABC',
    Cont: 2276,
    email: 'madhura@gmail.com'
  },
  {
    _id: ObjectId("63ced7fe4de806f62778f048"),
    RollNo: 10,
    Age: 23,
    Cont: 2276,
    email: 'meena@gmail.com'
  },
  {
    _id: ObjectId("63ced78b4de806f62778f044"),
    RollNo: 1,
    Age: 21,
    Cont: 9876,
    email: 'prannay@gmail.com'
  }
]
```

Queries

- **Create a database “Student” with the following attributes Rollno, age, contactNo, Email-Id.**

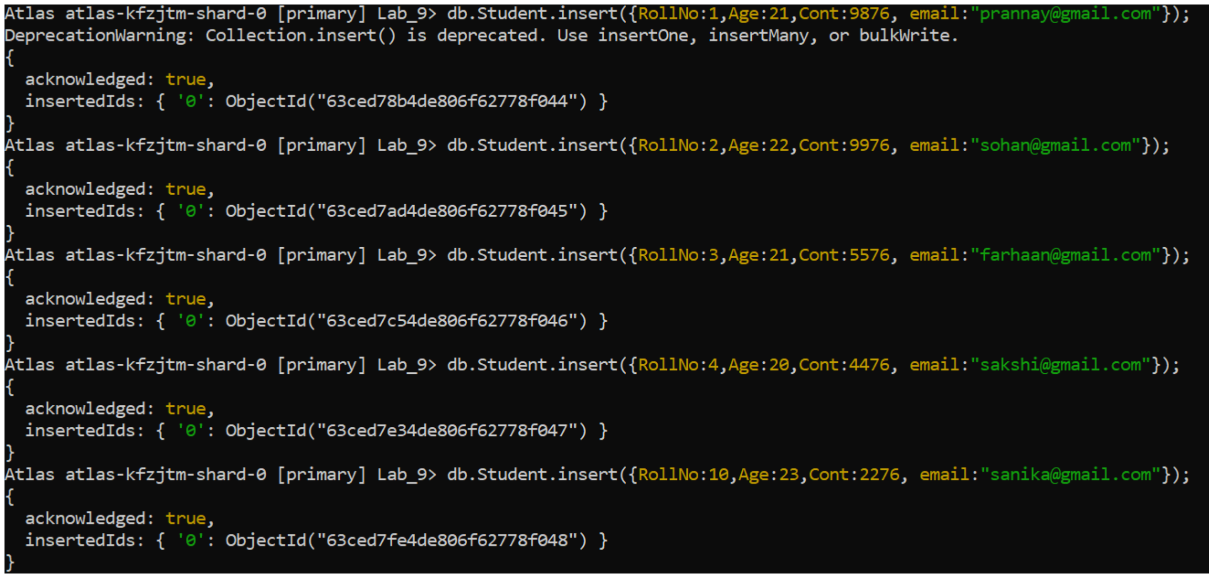
db.createCollection("Student");



```
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.createCollection("Student");
{ ok: 1 }
```

- **Insert appropriate values**

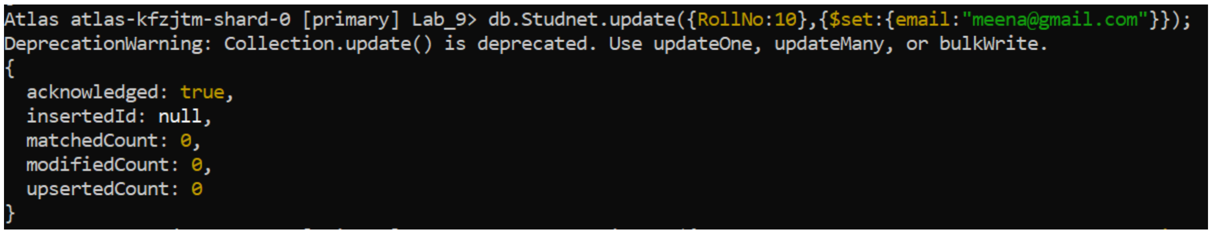
db.Student.insert({rollno:1,age:21,cont:9876,email:"prannay@gmail.com"});
db.Student.insert({rollno:2,age:22,cont:9976,email:"sohan@gmail.com"});
db.Student.insert({rollno:3,age:21,cont:5576,email:"farhaan@gmail.com"});
db.Student.insert({rollno:4,age:20,cont:4476,email:"sakshi@gmail.com"});
db.Student.insert({rollno:5,age:23,cont:2276,email:"sanika@gmail.com"});



```
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Student.insert({RollNo:1,Age:21,Cont:9876, email:"prannay@gmail.com"});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63ced78b4de806f62778f044") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Student.insert({RollNo:2,Age:22,Cont:9976, email:"sohan@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63ced7ad4de806f62778f045") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Student.insert({RollNo:3,Age:21,Cont:5576, email:"farhaan@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63ced7c54de806f62778f046") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Student.insert({RollNo:4,Age:20,Cont:4476, email:"sakshi@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63ced7e34de806f62778f047") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Student.insert({RollNo:10,Age:23,Cont:2276, email:"sanika@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63ced7fe4de806f62778f048") }
}
```

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

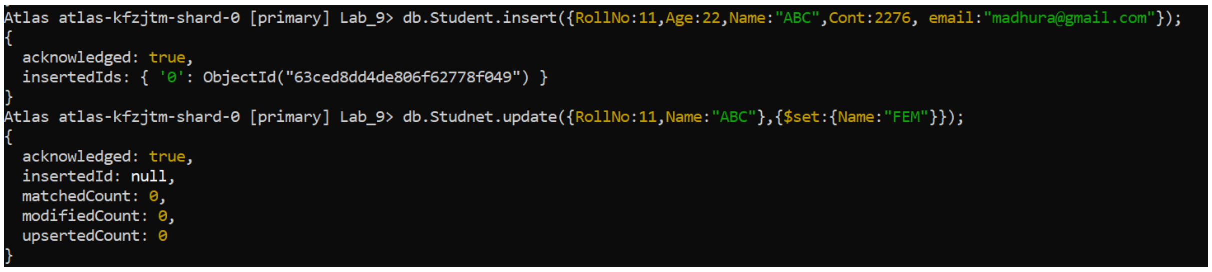
db.Student.update({rollno:5},{$set:{email:"abhinav@gmail.com"}})



```
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Studnet.update({RollNo:10},{$set:{email:"meena@gmail.com"}});
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
```

- **<u>Replace the student name from “ABC” to “FEM” of rollno 11.</u>**

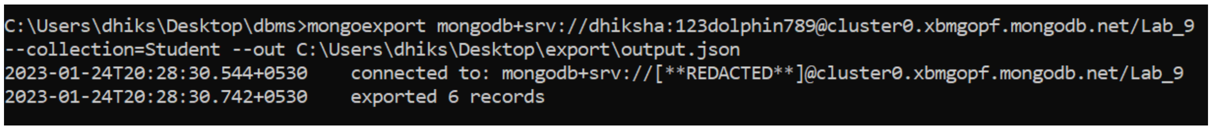
db.Student.insert({rollno:11,age:22,name:"ABC",cont:2276,email:"madhura@gmail.com"});
db.Student.update({rollno:11,name:"ABC"},{$set:{name:"FEM"}})



```
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Student.insert({RollNo:11,Age:22,Name:"ABC",Cont:2276, email:"madhura@gmail.com"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63ced8dd4de806f62778f049") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Studnet.update({RollNo:11,Name:"ABC"},{$set:{Name:"FEM"}});
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
```

- **<u>Export the created table into local file system</u>**

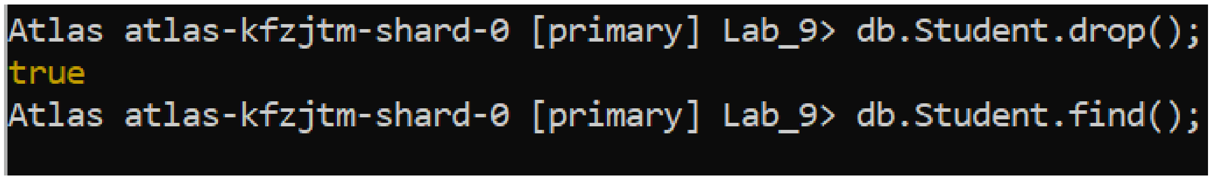
mongoexport mongodb+srv://dhiksha:<password>@cluster0.xbmgopf.mongodb.net/Lab_9
--collection=Student -- out C:\Users\dhiks\Desktop\export\output.json



```
C:\Users\dhiks\Desktop\dbms>mongoexport mongodb+srv://dhiksha:123dolphin789@cluster0.xbmgopf.mongodb.net/Lab_9
--collection=Student --out C:\Users\dhiks\Desktop\export\output.json
2023-01-24T20:28:30.544+0530    connected to: mongodb+srv://[**REDACTED**]@cluster0.xbmgopf.mongodb.net/Lab_9
2023-01-24T20:28:30.742+0530    exported 6 records
```

- **<u>Drop the table</u>**

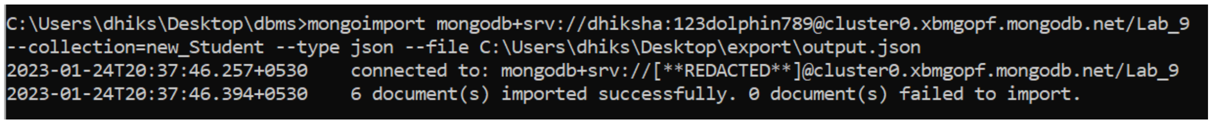
db.Student.drop();



```
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Student.drop();
true
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Student.find();
```

- **<u>Import a given csv dataset from local file system into mongodb collection.</u>**

mongoimport mongodb+srv://dhiksha:<password>@cluster0.xbmgopf.mongodb.net/Lab_9
--collection=new_Student – type json –file C:\Users\dhiks\Desktop\export\output.json



```
C:\Users\dhiks\Desktop\dbms>mongoimport mongodb+srv://dhiksha:123dolphin789@cluster0.xbmgopf.mongodb.net/Lab_9
--collection=new_Student --type json --file C:\Users\dhiks\Desktop\export\output.json
2023-01-24T20:37:46.257+0530    connected to: mongodb+srv://[**REDACTED**]@cluster0.xbmgopf.mongodb.net/Lab_9
2023-01-24T20:37:46.394+0530    6 document(s) imported successfully. 0 document(s) failed to import.
```

NoSQL Lab 2

Question

(Week 10)

Perform the following DB operations using MongoDB.

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

2. Insert at least 5 values into the table

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

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

5. Export the created collection into local file system

6. Drop the table

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

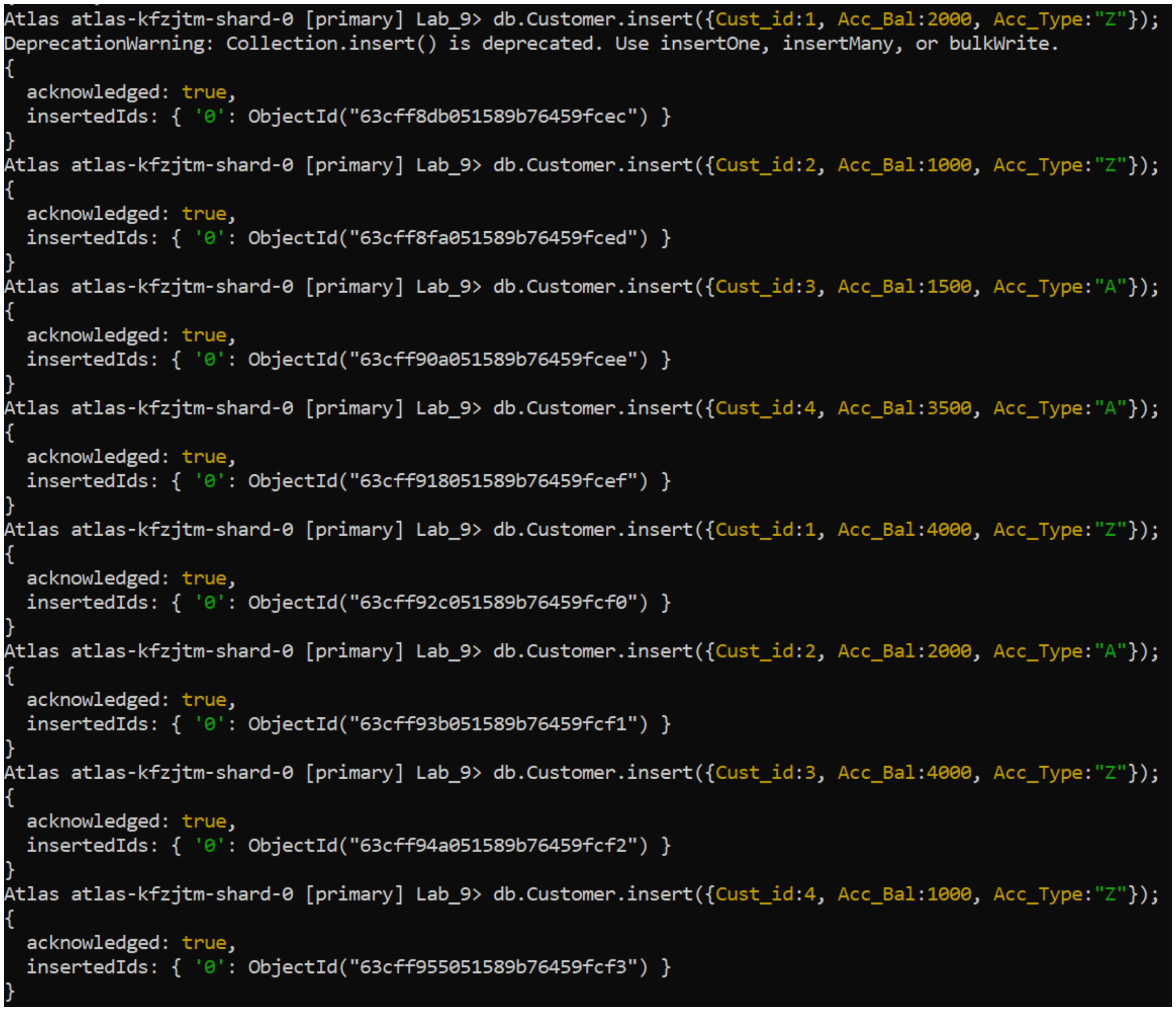
collection.

Create database

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

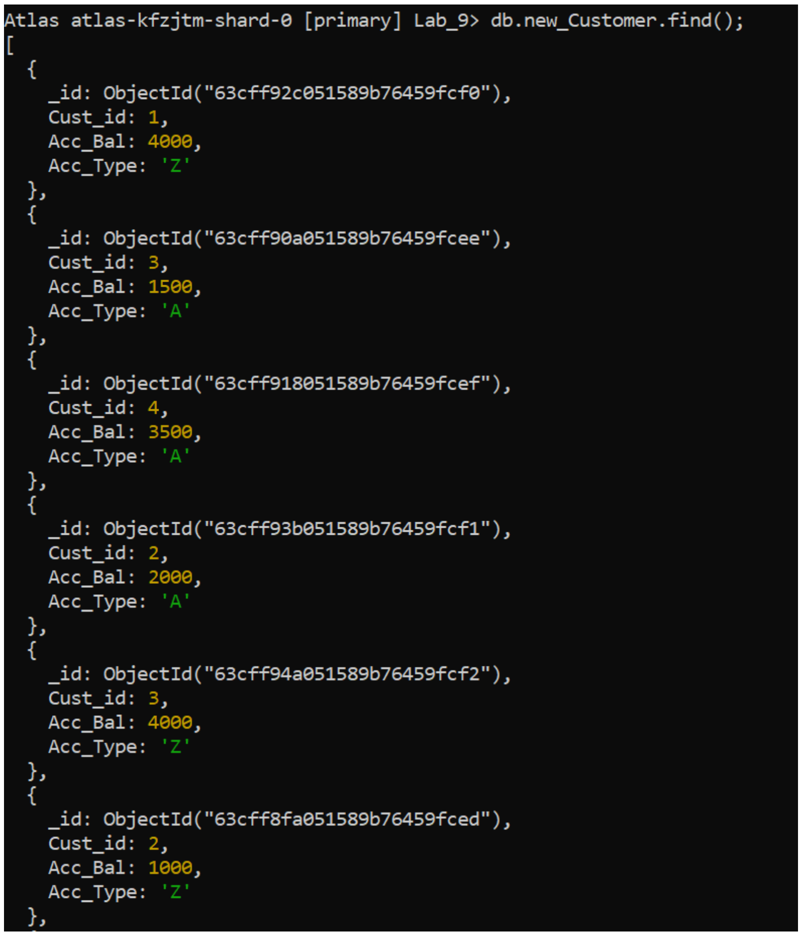
Create table & Inserting Values to the table

```
db.Customer.insert({Cust_id:1,Acc_bal:2000,Acc_type:"Z"});
db.Customer.insert({Cust_id:2,Acc_bal:1000,Acc_type:"Z"});
db.Customer.insert({Cust_id:3,Acc_bal:1500,Acc_type:"A"});
db.Customer.insert({Cust_id:4,Acc_bal:3500,Acc_type:"A"});
db.Customer.insert({Cust_id:1,Acc_bal:4000,Acc_type:"Z"});
db.Customer.insert({Cust_id:2,Acc_bal:2000,Acc_type:"A"});
db.Customer.insert({Cust_id:3,Acc_bal:4000,Acc_type:"Z"});
db.Customer.insert({Cust_id:4,Acc_bal:1000,Acc_type:"Z"});
```

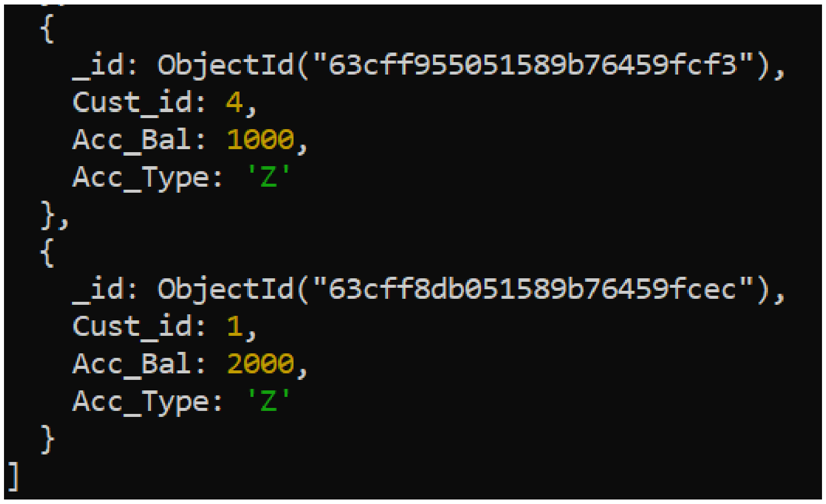

```
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:1, Acc_Bal:2000, Acc_Type:"Z"});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff8db051589b76459fcec") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:2, Acc_Bal:1000, Acc_Type:"Z"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff8fa051589b76459fced") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:3, Acc_Bal:1500, Acc_Type:"A"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff90a051589b76459fcee") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:4, Acc_Bal:3500, Acc_Type:"A"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff918051589b76459fcef") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:1, Acc_Bal:4000, Acc_Type:"Z"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff92c051589b76459fcf0") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:2, Acc_Bal:2000, Acc_Type:"A"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff93b051589b76459fcf1") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:3, Acc_Bal:4000, Acc_Type:"Z"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff94a051589b76459fcf2") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:4, Acc_Bal:1000, Acc_Type:"Z"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff955051589b76459fcf3") }
}
```

Structure of the table

db.Customer.find();



```
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.new_Customer.find();
[
  {
    _id: ObjectId("63cff92c051589b76459fcf0"),
    Cust_id: 1,
    Acc_Bal: 4000,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("63cff90a051589b76459fcee"),
    Cust_id: 3,
    Acc_Bal: 1500,
    Acc_Type: 'A'
  },
  {
    _id: ObjectId("63cff918051589b76459fcef"),
    Cust_id: 4,
    Acc_Bal: 3500,
    Acc_Type: 'A'
  },
  {
    _id: ObjectId("63cff93b051589b76459fcf1"),
    Cust_id: 2,
    Acc_Bal: 2000,
    Acc_Type: 'A'
  },
  {
    _id: ObjectId("63cff94a051589b76459fcf2"),
    Cust_id: 3,
    Acc_Bal: 4000,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("63cff8fa051589b76459fced"),
    Cust_id: 2,
    Acc_Bal: 1000,
    Acc_Type: 'Z'
  },
```

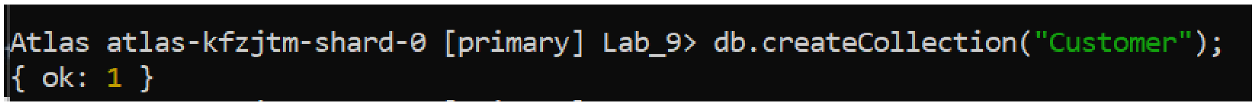


```
  {
    _id: ObjectId("63cff955051589b76459fcf3"),
    Cust_id: 4,
    Acc_Bal: 1000,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("63cff8db051589b76459fcec"),
    Cust_id: 1,
    Acc_Bal: 2000,
    Acc_Type: 'Z'
  }
]
```

Queries

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

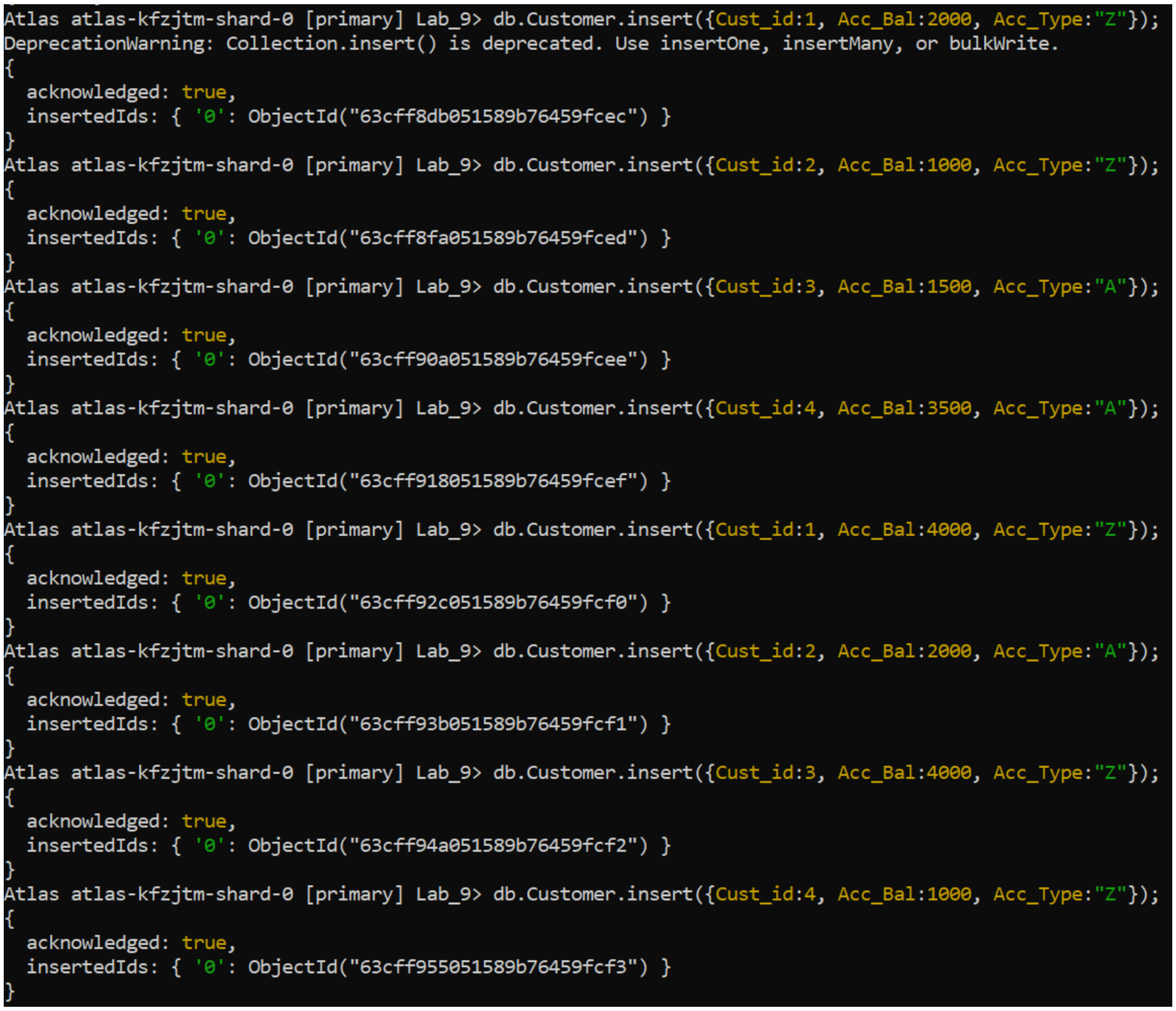
db.createCollection("Customer");



```
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.createCollection("Customer");
{ ok: 1 }
```

- **Insert at least 5 values into the table**

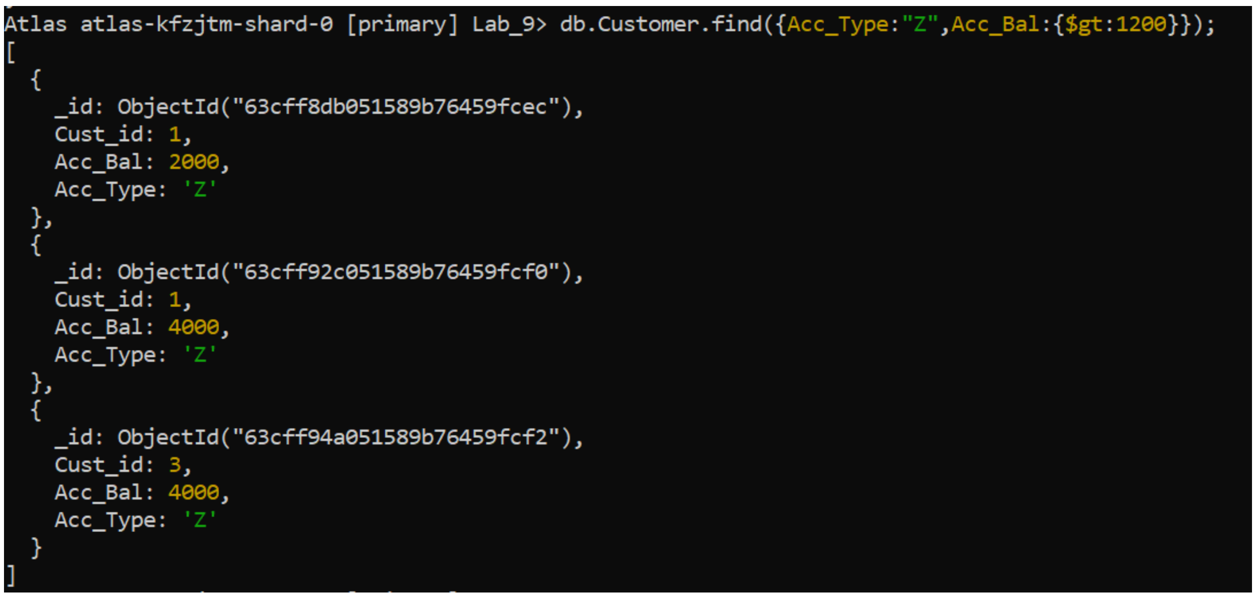
```
db.Customer.insert({Cust_id:1,Acc_bal:2000,Acc_type:"Z"});
db.Customer.insert({Cust_id:2,Acc_bal:1000,Acc_type:"Z"});
db.Customer.insert({Cust_id:3,Acc_bal:1500,Acc_type:"A"});
db.Customer.insert({Cust_id:4,Acc_bal:3500,Acc_type:"A"});
db.Customer.insert({Cust_id:1,Acc_bal:4000,Acc_type:"Z"});
db.Customer.insert({Cust_id:2,Acc_bal:2000,Acc_type:"A"});
db.Customer.insert({Cust_id:3,Acc_bal:4000,Acc_type:"Z"});
db.Customer.insert({Cust_id:4,Acc_bal:1000,Acc_type:"Z"});
```



```
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:1, Acc_Bal:2000, Acc_Type:"Z"});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff8db051589b76459fcec") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:2, Acc_Bal:1000, Acc_Type:"Z"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff8fa051589b76459fced") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:3, Acc_Bal:1500, Acc_Type:"A"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff90a051589b76459fcee") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:4, Acc_Bal:3500, Acc_Type:"A"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff918051589b76459fcef") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:1, Acc_Bal:4000, Acc_Type:"Z"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff92c051589b76459fcf0") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:2, Acc_Bal:2000, Acc_Type:"A"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff93b051589b76459fcf1") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:3, Acc_Bal:4000, Acc_Type:"Z"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff94a051589b76459fcf2") }
}
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.insert({Cust_id:4, Acc_Bal:1000, Acc_Type:"Z"});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("63cff955051589b76459fcf3") }
}
```

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

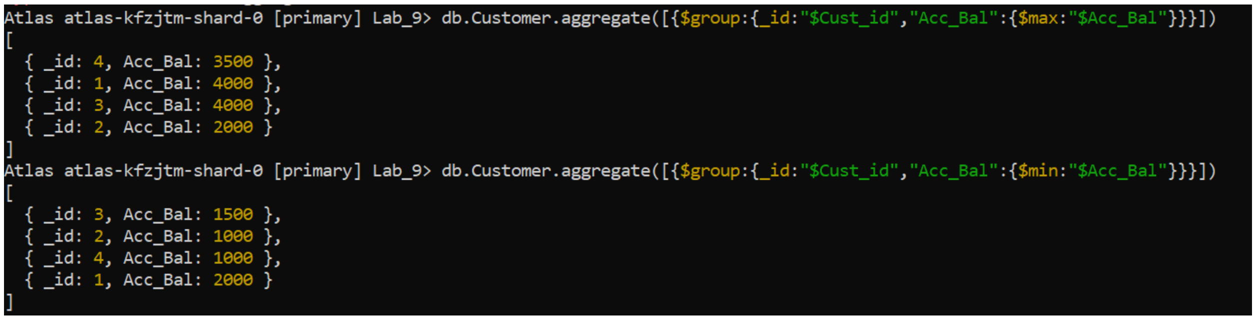
db.Customer.find({Acc_Type:”Z”, Acc_Bal:{$gt:1200}});



```
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.find({Acc_Type:"Z",Acc_Bal:{$gt:1200}});
[
  {
    _id: ObjectId("63cff8db051589b76459fcec"),
    Cust_id: 1,
    Acc_Bal: 2000,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("63cff92c051589b76459fcf0"),
    Cust_id: 1,
    Acc_Bal: 4000,
    Acc_Type: 'Z'
  },
  {
    _id: ObjectId("63cff94a051589b76459fcf2"),
    Cust_id: 3,
    Acc_Bal: 4000,
    Acc_Type: 'Z'
  }
]
```

- **<u>Determine Minimum and Maximum account balance for each customer_id.</u>**

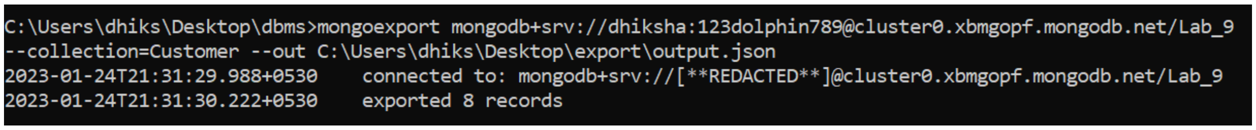
db.Customer.aggregrate([{$group:{_id:"$Cust_id","Acc_Bal":{$max:"$Acc_Bal"}}}])
db.Customer.aggregrate([{$group:{_id:"$Cust_id","Acc_Bal":{$min:"$Acc_Bal"}}}])



```
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.aggregate([{$group:{_id:"$Cust_id","Acc_Bal":{$max:"$Acc_Bal"}}}])
[
  { _id: 4, Acc_Bal: 3500 },
  { _id: 1, Acc_Bal: 4000 },
  { _id: 3, Acc_Bal: 4000 },
  { _id: 2, Acc_Bal: 2000 }
]
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.aggregate([{$group:{_id:"$Cust_id","Acc_Bal":{$min:"$Acc_Bal"}}}])
[
  { _id: 3, Acc_Bal: 1500 },
  { _id: 2, Acc_Bal: 1000 },
  { _id: 4, Acc_Bal: 1000 },
  { _id: 1, Acc_Bal: 2000 }
]
```

- **<u>Export the created collection into local file system</u>**

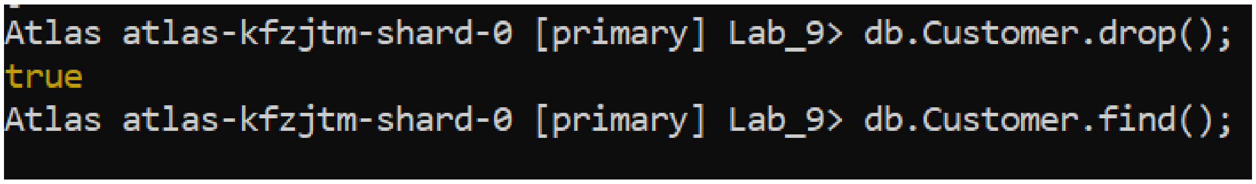
mongoexport mongodb+srv://dhiksha:<password>@cluster0.xbmgopf.mongodb.net/Lab_9
--collection=Customer -- out C:\Users\dhiks\Desktop\export\output.json



```
C:\Users\dhiks\Desktop\dbms>mongoexport mongodb+srv://dhiksha:123dolphin789@cluster0.xbmgopf.mongodb.net/Lab_9
--collection=Customer --out C:\Users\dhiks\Desktop\export\output.json
2023-01-24T21:31:29.988+0530    connected to: mongodb+srv://[**REDACTED**]@cluster0.xbmgopf.mongodb.net/Lab_9
2023-01-24T21:31:30.222+0530    exported 8 records
```

- **Drop the table**

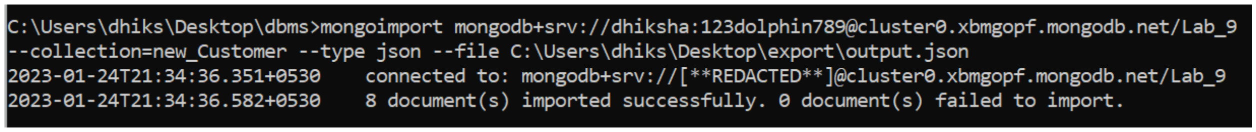
db.Customer.drop();



```
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.drop();
true
Atlas atlas-kfzjtm-shard-0 [primary] Lab_9> db.Customer.find();
```

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

mongoimport mongodb+srv://dhiksha:<password>@cluster0.xbmgopf.mongodb.net/Lab_9 --collection=new_Customer – type json –file C:\Users\dhiks\Desktop\export\output.json



```
C:\Users\dhiks\Desktop\dbms>mongoimport mongodb+srv://dhiksha:123dolphin789@cluster0.xbmgopf.mongodb.net/Lab_9
--collection=new_Customer --type json --file C:\Users\dhiks\Desktop\export\output.json
2023-01-24T21:34:36.351+0530    connected to: mongodb+srv://[**REDACTED**]@cluster0.xbmgopf.mongodb.net/Lab_9
2023-01-24T21:34:36.582+0530    8 document(s) imported successfully. 0 document(s) failed to import.
```