VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Database Management Systems (23CS3PCDBM)

Submitted by

Bhavya Goyal(1BM23CS063)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
Sep-2024 to Jan-2025

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "Database Management Systems (23CS3PCDBM)" carried out by **Bhavya Goyal(1BM23CS063),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a Database Management Systems (23CS3PCDBM) work prescribed for the said degree.

Lab faculty Incharge Name Assistant Professor Department of CSE, BMSCE	kayarvizhy N Professor Department of CSE, BMSCE

Index

Insurance Database

Question

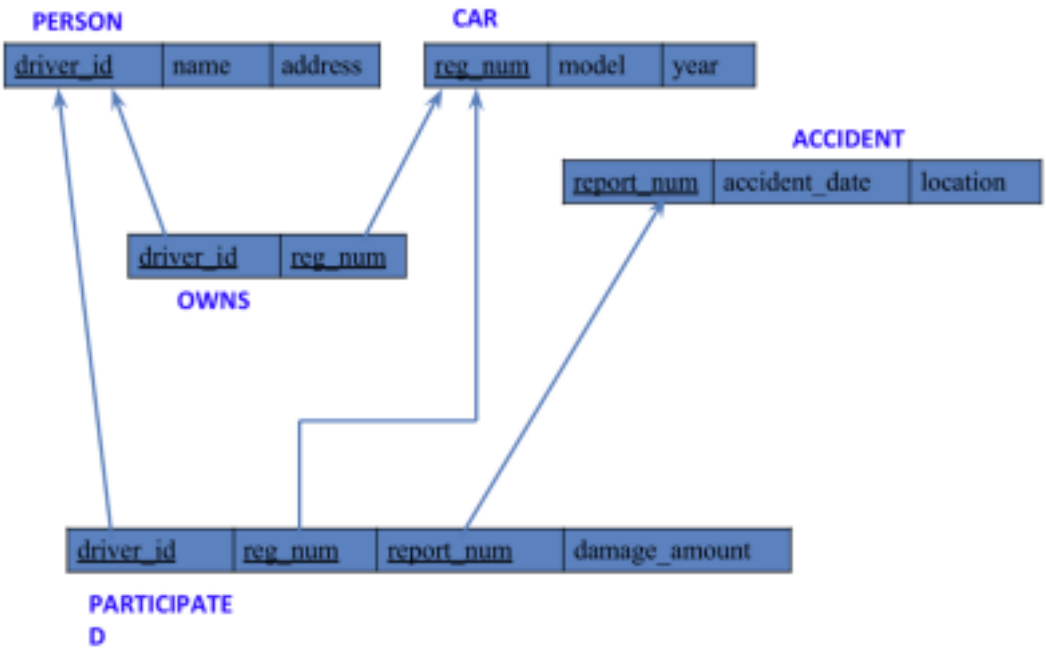
(Week 1)

- PERSON (driver_id: String, name: String, address: String)
- CAR (reg_num: String, model: String, year: int)
- ACCIDENT (report_num: int, accident_date: date, location: String)
- OWNS (driver_id: String, reg_num: String)
- PARTICIPATED (driver_id: String,reg_num: String, report_num: int, damage_amount: int)
- Create the above tables by properly specifying the primary keys and the foreign keys. -

Enter at least five tuples for each relation

- 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 BM;

use BM;

Create table

```
create table person (
driver_id varchar(10),
name varchar(20),
address varchar(30),
primary key(driver_id));

create table car(
 reg_num varchar(10),
 model varchar(20),
 laun_date int,
 primary key(reg_num));
```

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

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

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

Structure of the table

desc person;

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

Field	Type	Null	Key	Default	Extra
driver_id	varchar(10)	NO	PRI	NULL	
name	varchar(20)	YES		NULL	
address	varchar(30)	YES		NULL	

desc accident;

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

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

desc participated;

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

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

desc car;

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

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

desc owns;

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

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

Inserting Values to the table

insert into person

values('123@ram','ram singh','gandhi nagar jaipur');

insert into person

values ('1263@baao','babu rao singh','bandhi nagar jaipur'),

('1265@ram','ramsd singh','gandhdhhi nagar jaipur'),

('123@bhav','bhav singh','vijaya nagar jaipur');

insert into person

values ('1263@gud','kanis singh','bandhi nagar jaipur');

select * from person;

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

driver_id	name	address
123@bhav	bhav singh	vijaya nagar jaipur
123@ram	ram singh	gandhi nagar jaipur
1263@baao	babu rao singh	bandhi nagar jaipur
1263@gud	kanis singh	bandhi nagar jaipur
1265@ram	ramsd singh	gandhdhhi nagar jaipur
NULL	NULL	NULL

insert into car

values('rj23cs0533','BMW X1', 2015),

('rj24cs0533','BMW Z1', 2016),

('rj25cs0533','BMW Y1', 2018),

('rj14cs0533','BMW R1', 2019),

('rj15cs0533','BMW XYZ1', 2022);

select *

from car;

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

reg_num	model	laun_date
rj14cs0533	BMW R1	2019
rj15cs0533	BMW XYZ1	2022
rj23cs0533	BMW X1	2015
rj24cs0533	BMW Z1	2016
rj25cs0533	BMW Y1	2018
NULL	NULL	NULL

```
insert into owns
values('123@bhav','rj14cs0533');
insert into owns
values ('123@ram','rj15cs0533');
insert into owns
values('123@ram','rj25cs0533');
insert into owns
values('123@gud','rj24cs0533');
insert into owns
values('123@baao','rj23cs0533');
select * from owns;
```

driver_id	reg_num
123@bhav	rj14cs0533
123@ram	rj15cs0533
123@ram	rj25cs0533
NULL	NULL

```
insert into accident
values(11,'2003-01-01','mysore road'),
(12,'2004-02-02','south city'),
(13,'2003-01-21','Bull temple raod'),
(14,'2008-01-01','mysore road'),
(24,'2010-06-01','mysore road');
```

select *

from accident;

report_num	accident_date	location
11	2003-01-01	mysore road
12	2004-02-02	south city
13	2003-01-21	Bull temple raod
14	2008-01-01	mysore road
24	2010-06-01	mysore road
NULL	NULL	NULL

insert into participated

values('123@bhav','rj14cs0533',11,10000);

insert into participated

values ('123@ram','rj15cs0533',12,200000);

insert into participated

values ('123@ram','rj25cs0533',13,230000);

insert into owns

values('123@gud','rj24cs0533',14,24000);

select *

from participated;

driver_id	reg_num	report_num	damage_amount
123@bhav	rj14cs0533	11	25000
123@ram	rj15cs0533	12	200000
123@ram	rj25cs0533	13	230000
NULL	NULL	NULL	NULL

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;

Result Grid | Filter Rows: | Edit:

driver_id	reg_num	report_num	damage_amount
123@bhav	rj14cs0533	11	25000
123@ram	rj15cs0533	12	200000
123@ram	rj25cs0533	13	230000
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%';

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

CNT
0

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

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

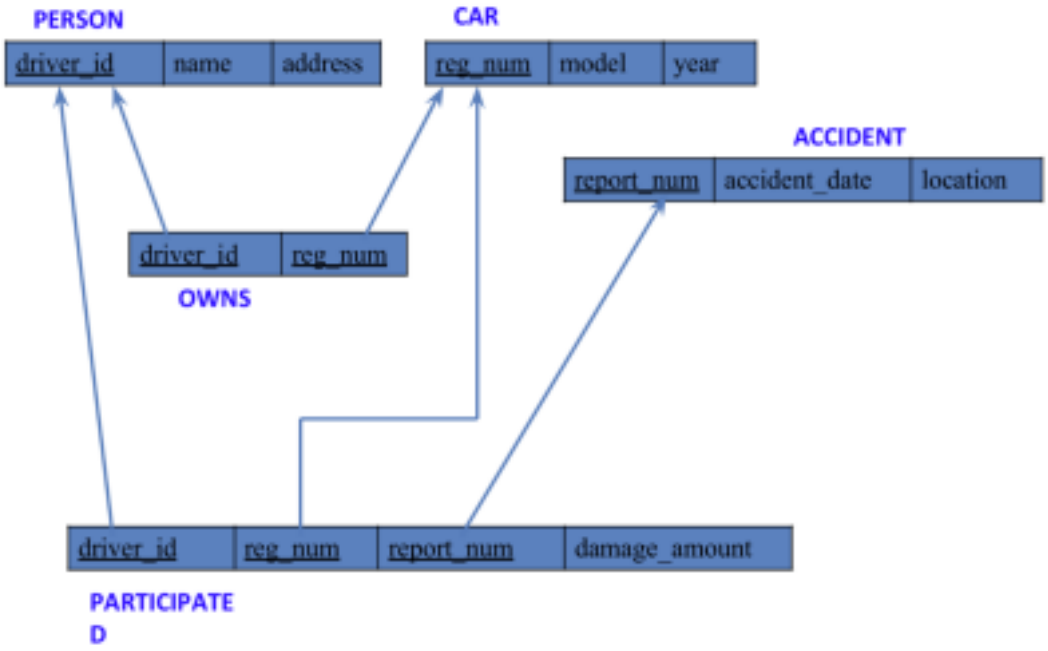
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

report_num	accident_date	location
11	2003-01-01	mysore road
12	2004-02-02	south city
13	2003-01-21	Bull temple raod
14	2008-01-01	mysore road
24	2010-06-01	mysore road
NULL	NULL	NULL

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:



Queries

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

select * from car_204 order by year asc;

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 city
13	2003-01-21	Bull temple raod
14	2008-01-01	mysore road
24	2010-06-01	mysore road
NULL	NULL	NULL

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

select model, count(model) from participated_204, car_204 where participated_204.reg_no = car_204.reg_no group by model;

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

accident_count
0

● FIND THE AVERAGE DAMAGE AMOUNT

select avg(damage_amount) as average from participated;

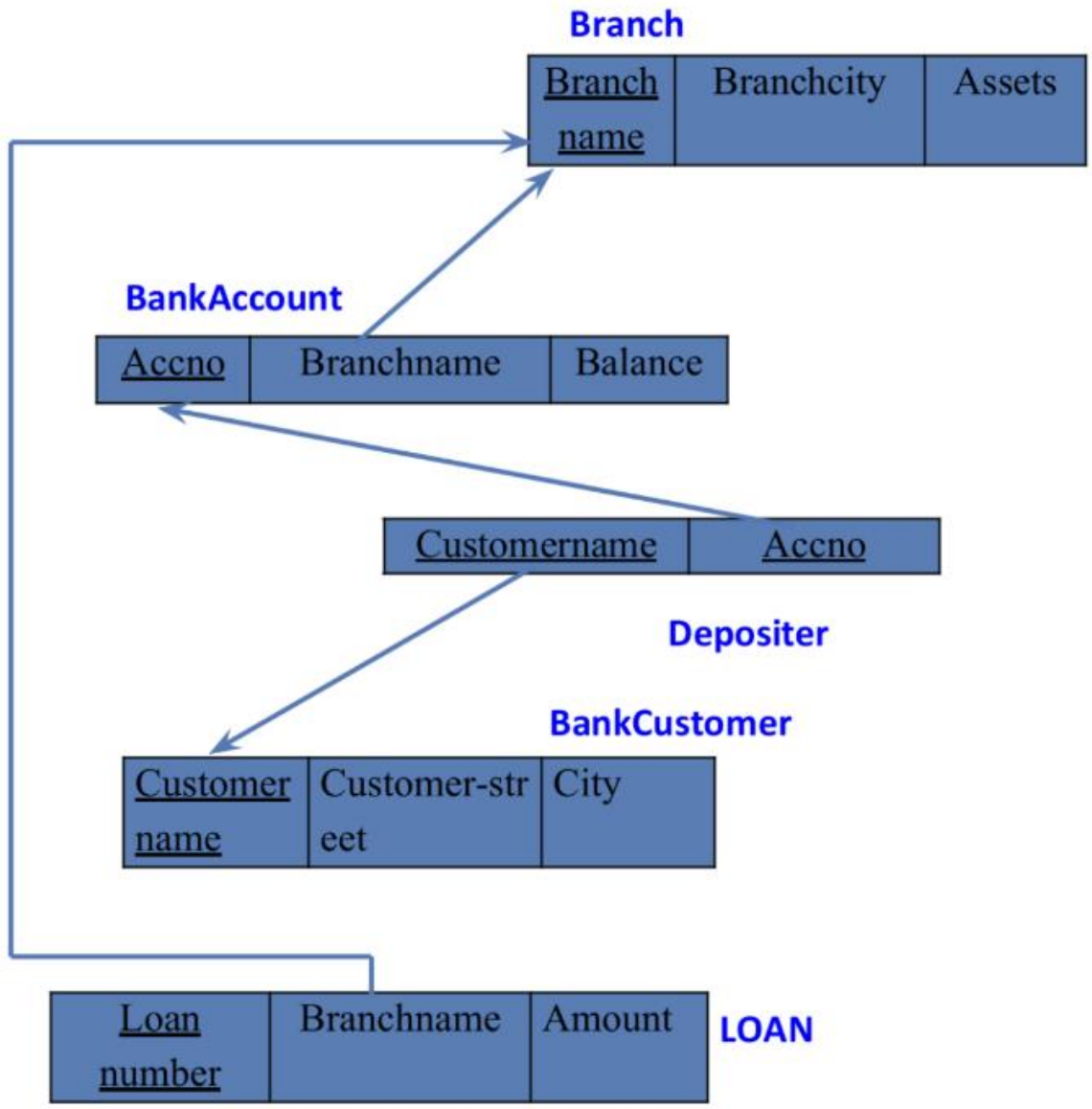
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

average
151666.6667

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 bank1;
use bank1;
```

Create table

```
create table Branch(
Branch_Name varchar(100),
Branch_city varchar(100),
Assets int,
primary key(Branch_Name)
);

select *
from Branch;

create table Bank_Account(
Accno int,
Branch_Name varchar(100),
Balance int,
primary key(Accno),
foreign key(Branch_Name) references Branch(Branch_Name)
);

select *
from Bank_Account;

create table Bank_Customer(
Customer_Name varchar(100),
Customer_Street Varchar(100),
City varchar(100),
primary key(Customer_Name)
);

select *
from  Bank_Customer;

create table Depositer(
Customer_Name varchar(100),
Accno int,
primary key(Customer_Name,Accno),
foreign key(Customer_Name) references Bank_Customer(Customer_Name),
foreign key(Accno) references Bank_Account(Accno)
);
```

```
select *
from  Depositer;

create table Loan(
Loan_Number int,
Branch_Name Varchar(100),
Amount int,
primary key(Loan_Number),
foreign key(Branch_Name) references Branch(Branch_Name)
);
select *
from  Loan;
```

Structure of the table

desc Branch;

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

Field	Type	Null	Key	Default	Extra
Branch_Name	varchar(100)	NO	PRI	NULL	
Branch_city	varchar(100)	YES		NULL	
Assets	int	YES		NULL	

desc Bank_Account;

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

Field	Type	Null	Key	Default	Extra
Accno	int	NO	PRI	NULL	
Branch_Name	varchar(100)	YES	MUL	NULL	
Balance	int	YES		NULL	

desc Bank_Customer;

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

Field	Type	Null	Key	Default	Extra
Customer_Name	varchar(100)	NO	PRI	NULL	
Customer_Street	varchar(100)	YES		NULL	
City	varchar(100)	YES		NULL	

desc Depositer;

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

Field	Type	Null	Key	Default	Extra
Customer_Name	varchar(100)	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(100)	YES	MUL	NULL	
Amount	int	YES		NULL	

Insert Value To the Table

insert into Branch
value ('SBI_Chamrajpet', 'Bangalore',50000),
('SBI_ResidencyRoad' ,'Bangalore',10000),
('SBI_ShivajiRoad', 'Bombay',20000),
('SBI_ParlimentRoad', 'Delhi',10000),
('SBI_Jantarmantar', 'Delhi',20000);

insert into Bank_Account
value (1,'SBI_Chamrajpet', 2000),
(2,'SBI_ResidencyRoad', 5000),
(3, 'SBI_ShivajiRoad', 6000),
(4, 'SBI_ParlimentRoad', 9000),
(5, 'SBI_Jantarmantar', 8000),
(6, 'SBI_ShivajiRoad', 4000),
(7, 'SBI_ResidencyRoad', 4000),
(8, 'SBI_ParlimentRoad', 3000),
(9, 'SBI_ResidencyRoad', 5000),
(10,'SBI_Jantarmantar', 2000);

insert into Bank_Customer
value ('Avinash','Bull Temple Road','Bangalore'),
('Dinesh','Bannergatta Road','Bangalore'),
('Mohan', 'NationalCollege Road', 'Bangalore'),
('Nikil', 'Akbar Road','Delhi'),

('Ravi','Prithviraj Road','Delhi');

insert into Depositer
value('Avinash',1),
('Dinesh',2),
('Nikil',4),
('Ravi',5),
('Avinash',6),
('Nikil',8),
('Dinesh',9),
('Nikil',10);

insert into Loan
value(1, 'SBI_Chamrajpet',1000),
(2, 'SBI_ResidencyRoad',2000),
(3, 'SBI_ShivajiRoad',3000),
(4, 'SBI_ParlimentRoad',4000),
(5, 'SBI_Jantarmantar',5000);

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

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

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
7	SBI_ResidencyRoad	4000
8	SBI_ParlimentRoad	3000
9	SBI_ResidencyRoad	5000
10	SBI_Jantarmantar	2000
NULL	NULL	NULL

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

Customer_Name	Customer_Street	City
Avinash	Bull Temple Road	Bangalore
Dinesh	Bannergatta Road	Bangalore
Mohan	NationalCollege Road	Bangalore
Nikil	Akbar Road	Delhi
Ravi	Prithviraj Road	Delhi
NULL	NULL	NULL

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

Customer_Name	Accno
Avinash	1
Dinesh	2
Nikil	4
Ravi	5
Avinash	6
Nikil	8
Dinesh	9
Nikil	10
NULL	NULL

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

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,assets as assets_in_lakhs
from Branch;

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

Branch_Name	assets_in_lakhs
SBI_Chamrajpet	50000
SBI_Jantarmantar	20000
SBI_ParlimentRoad	10000
SBI_ResidencyRoad	10000
SBI_ShivajiRoad	20000
NULL	NULL

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

```
select d.Customer_name
from Depositer d, Bank_Account b
where b.Accno=d.Accno and Branch_Name='SBI_ResidencyRoad'
group by Customer_Name
having count(Customer_Name)>1;
```

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

Customer_name
Dinesh

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

```
CREATE VIEW Branch_Loan_Sum AS
SELECT Branch_Name, SUM(Amount) AS Total_Loan_Amount
FROM Loan
GROUP BY Branch_Name;
SELECT * FROM Branch_Loan_Sum;
```

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

Branch_Name	Total_Loan_Amount
SBI_Chamrajpet	1000
SBI_Jantarmantar	5000
SBI_ParlimentRoad	4000
SBI_ResidencyRoad	2000
SBI_ShivajiRoad	3000

More Queries on Bank Database

Question
(Week 4)
1. Retrieve all branches and their respective total assets

2. List all customers who live in a particular city

3. List all customers with their account numbers

3. List all customers with their loan amounts

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

5. Find all customers who have accounts with a balance greater than a specified amount (100000)

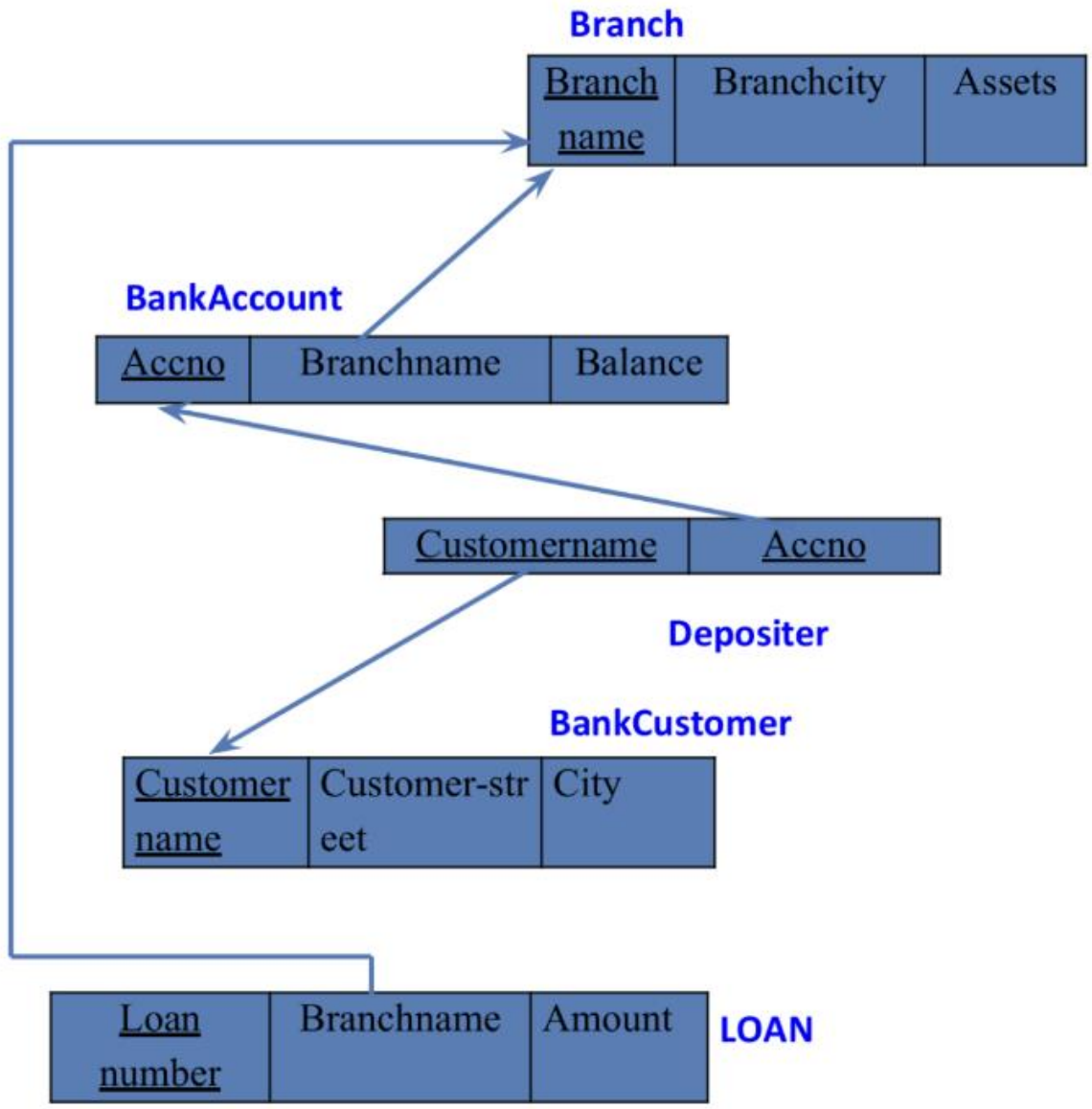
6. List all customers who have both a loan and an account at the same branch

7. Get the number of accounts held at each branch

8. Find all branches that have no loans issued

9. Retrieve the branch with the smallest total loan amount

Schema Diagram



Queries

Retrieve all branches and their respective total assets

```
select Branch_Name,assets as total_assets
from Branch;
```

Branch_Name	total_assets
SBI_Chamrajpet	50000
SBI_Jantarmantar	20000
SBI_ParlimentRoad	10000
SBI_ResidencyRoad	10000
SBI_ShivajiRoad	20000
NULL	NULL

List all customers who live in a particular city

```
select Customer_Name
from Bank_customer
where City='delhi';
```

Customer_Name
Nikil
Ravi
NULL

List all customers with their account numbers

```
select Customer_Name, Accno
from Depositer;
```

Customer_Name	Accno
Avinash	1
Dinesh	2
Nikil	4
Ravi	5
Avinash	6
Nikil	8
Dinesh	9
Nikil	10
NULL	NULL

Find all customers who have accounts with a balance greater than a specified amount (100000)

```
Select d.Customer_Name, a.Balance
from Depositer d
Left join Bank_Account a on d.Accno= a.Accno
where a.Balance >'1000';
```

Customer_Name	Balance
Avinash	2000
Dinesh	5000
Nikil	9000
Ravi	8000
Avinash	4000
Nikil	3000
Dinesh	5000
Nikil	2000

Get the number of accounts held at each branch

```
SELECT count(*) as "Number Of Accounts",Branch_Name
From Bank_Account
group by (Branch_Name);
```

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

Number Of Accounts	Branch_Name
1	SBI_Chamrajpet
2	SBI_Jantarmantar
2	SBI_ParlimentRoad
3	SBI_ResidencyRoad
2	SBI_ShivajiRoad

Retrieve the branch with the smallest total loan amount

```
Select Branch_Name, Sum(Amount) as Total_Loans
From Loan
group by Branch_Name
order by Total_Loans ASC
LIMIT 1;
```

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

Branch_Name	Total_Loans
SBI_Chamrajpet	1000

Find all branches that have no loans issued

```
select Branch_Name
From Branch
where Branch_Name not in (Select Branch_Name From Loan);
```

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

Branch_Name
NULL

Project DataBase

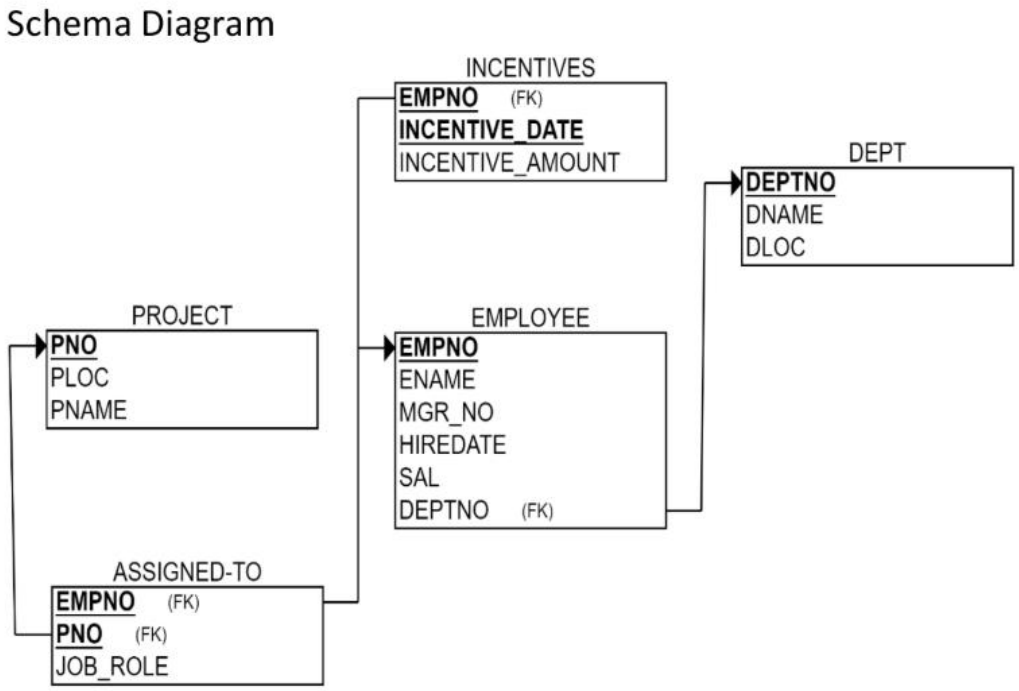
Question
(Week 5)
1. Using Scheme diagram, Create tables by properly specifying the primary keys and the foreign keys.

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.



Create database

```
create database nov8;
use nov8;
```

Create table

```
create table Dept(
Deptno int,
DName varchar(50),
DLoc varchar(50),
primary key(Deptno)
);
create table Project(
PNo int,
PLoc varchar(50),
PName varchar(50),
primary key (PNo)
);
create table Employee(
EmpNo int,
EName varchar(50),
Mgr_no int,
Hiredate varchar(50),
Salary int,
Deptno int,
primary key(EmpNo),
foreign key(Deptno) references Dept (Deptno)
);

create Table Incentives(
Incentive_Date varchar(50),
Incentive_Amount int,
EmpNo int,
primary key(Incentive_Date),
foreign key(EmpNo) references Employee(EmpNo)
);

create table Assigned_To(
Job_Role varchar(50),
EmpNo int,
PNo int,
```

primary key (Job_Role),
foreign key(EmpNo) references Employee(EmpNo),
foreign key(PNo) references Project(PNo)
);

Structure of the table

desc Dept;

Field	Type	Null	Key	Default	Extra
Deptno	int	NO	PRI	NULL	
DName	varchar(50)	YES		NULL	
Dloc	varchar(50)	YES		NULL	

desc Project;

Field	Type	Null	Key	Default	Extra
PNo	int	NO	PRI	NULL	
PLoc	varchar(50)	YES		NULL	
PName	varchar(50)	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(50)	YES		NULL	
Mgr_no	int	YES		NULL	
Hiredate	varchar(50)	YES		NULL	
Salary	int	YES		NULL	
Deptno	int	YES	MUL	NULL	

desc Incentives;

Field	Type	Null	Key	Default	Extra
Incentive_Date	varchar(50)	NO	PRI	NULL	
Incentive_Amount	int	YES		NULL	
EmpNo	int	YES	MUL	NULL	

desc Assigned_To;

	Field	Type	Null	Key	Default	Extra
▶	Job_Role	varchar(50)	NO	PRI	NULL	
	EmpNo	int	YES	MUL	NULL	
	PNo	int	YES	MUL	NULL	

Inserting Value in the table

INSERT INTO Dept VALUES
(1, 'Human Resources', 'Bengaluru'),
(2, 'Finance', 'Jaipur'),
(3, 'Research', 'Bengaluru'),
(4, 'IT', 'Hyderabad'),
(5, 'Marketing', 'Mysuru'),
(6, 'Sales', 'Ajmer');
select * from Dept;

INSERT INTO Project VALUES
(101, 'Bengaluru', 'Project Alpha'),
(102, 'Jaipur', 'Project Beta'),
(103, 'Bengaluru', 'Project Gamma'),
(104, 'Hyderabad', 'Project Delta'),
(105, 'Mysuru', 'Project Epsilon'),
(106, 'Ajmer', 'Project Zeta');
select * from Project;

INSERT INTO Employee VALUES
(1001, 'Alice', 1005, '2023-01-15', 70000, 1),
(1002, 'Bob', 1005, '2022-04-10', 85000, 2),
(1003, 'Charlie', 1001, '2021-08-23', 78000, 3),
(1004, 'Daisy', NULL, '2019-06-19', 95000, 4),
(1005, 'Edward', NULL, '2018-11-30', 120000, 1),
(1006, 'Fiona', 1003, '2020-03-15', 60000, 5);
select * from Employee;

INSERT INTO Incentives VALUES
('2024-01-15', 2000, 1001),
('2024-03-10', 2500, 1002),

('2024-05-05', 1500, 1003),
('2024-06-20', 3000, 1004),
('2024-09-25', 1800, 1005),
select * from Incentives;

INSERT INTO Assigned_To VALUES
('Team Lead', 1001, 101),
('Project Manager', 1002, 102),
('Developer', 1003, 103),
('System Analyst', 1004, 104),
('Consultant', 1005, 105),
('Business Analyst', 1006, 106);
select * from Assigned_To;

Deptno	DName	Dloc
1	Human Resources	Bengaluru
2	Finance	Jaipur
3	Research	Bengaluru
4	IT	Hyderabad
5	Marketing	Mysuru
6	Sales	Ajmer
NULL	NULL	NULL

PNo	PLoc	PName
101	Bengaluru	Project Alpha
102	Jaipur	Project Beta
103	Bengaluru	Project Gamma
104	Hyderabad	Project Delta
105	Mysuru	Project Epsilon
106	Ajmer	Project Zeta
NULL	NULL	NULL

EmpNo	EName	Mgr_no	Hiredate	Salary	Deptno
1001	Alice	1005	2023-01-15	70000	1
1002	Bob	1005	2022-04-10	85000	2
1003	Charlie	1001	2021-08-23	78000	3
1004	Daisy	NULL	2019-06-19	95000	4
1005	Edward	NULL	2018-11-30	120000	1
1006	Fiona	1003	2020-03-15	60000	5
NULL	NULL	NULL	NULL	NULL	NULL

Incentive_Date	Incentive_Amount	EmpNo
2024-01-15	2000	1001
2024-03-10	2500	1002
2024-05-05	1500	1003
2024-06-20	3000	1004
2024-09-25	1800	1005
NULL	NULL	NULL

Job_Role	EmpNo	PNo
Business Analyst	1006	106
Consultant	1005	105
Developer	1003	103
Project Manager	1002	102
System Analyst	1004	104
Team Lead	1001	101
NULL	NULL	NULL

Queries

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

```
select E.EmpNo
from Employee E
join Assigned_To A on E.EmpNo = A.EmpNo
join Project P on A.PNo = P.PNo
where P.PLoc in ('Bengaluru', 'Hyderabad', 'Mysuru');
```

EmpNo
1001
1003
1004
1005

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

```
select EmpNo
from Employee
where EmpNo not in (select EmpNo from Incentives);
```

EmpNo
1006
NULL

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.

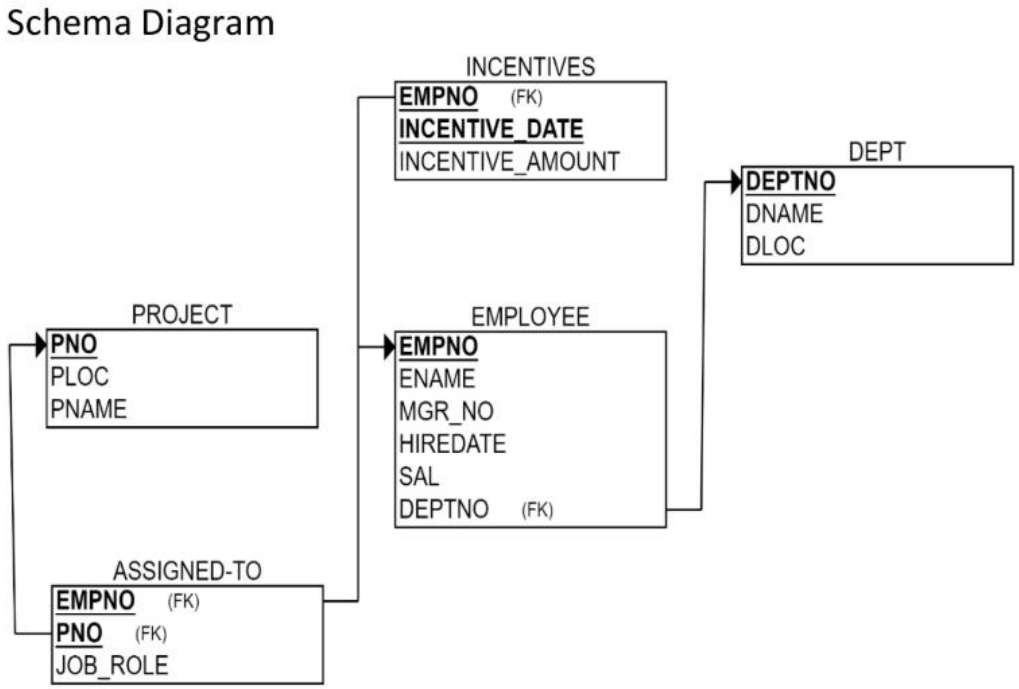
```
select E.EName, A.EmpNo, D.Deptno, A.Job_role, D.DLoc, P.PLoc
from Employee E
join Assigned_To A ON E.EmpNo = A.EmpNo
join Dept D ON E.Deptno = D.Deptno
join Project P ON A.PNo = P.PNo;
```

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

EName	EmpNo	Deptno	Job_role	DLoc	PLoc
Alice	1001	1	Team Lead	Bengaluru	Bengaluru
Bob	1002	2	Project Manager	Jaipur	Jaipur
Charlie	1003	3	Developer	Bengaluru	Bengaluru
Daisy	1004	4	System Analyst	Hyderabad	Hyderabad
Edward	1005	1	Consultant	Bengaluru	Mysuru
Fiona	1006	5	Business Analyst	Mysuru	Ajmer

More Query From Project

Question
(Week 6)
List all employees along with their project details (if assigned)
Find all employees who received incentives, along with the total incentive amount
Retrieve the project names and locations of projects with employees assigned as 'Manager'
List departments along with the number of employees in each department
Find employees who have not been assigned to any project
List all employees along with their department names and location
Retrieve the details of employees who work under a specific manager (e.g., manager with empno = 101)
List all projects that have employees assigned and the number of employees on each project



Queries

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

```
select E.EmpNo, E.EName, E.Deptno, A.Job_Role, P.PNo, P.PName, P.PLoc
from Employee E
join Assigned_To A on E.EmpNo = A.EmpNo
```

join Project P on A.PNo = P.PNo;

EmpNo	EName	Deptno	Job_Role	PNo	PName	PLoc
1001	Alice	1	Team Lead	101	Project Alpha	Bengaluru
1002	Bob	2	Project Manager	102	Project Beta	Jaipur
1003	Charlie	3	Developer	103	Project Gamma	Bengaluru
1004	Daisy	4	System Analyst	104	Project Delta	Hyderabad
1005	Edward	1	Consultant	105	Project Epsilon	Mysuru
1006	Fiona	5	Business Analyst	106	Project Zeta	Ajmer

-- question 2 Find all employees who received incentives, along with the total incentive amount

```
select E.EmpNo, E.EName, E.Deptno, sum(I.Incentive_Amount) as Total_Incentive
from Employee E,Incentives I
where E.EmpNo = I.EmpNo
group by E.EmpNo, E.EName, E.Deptno;

select E.EmpNo, E.EName, E.Deptno, sum(I.Incentive_Amount) as Total_Incentive
from Employee E
join Incentives I on E.EmpNo = I.EmpNo
group by E.EmpNo, E.EName, E.Deptno;
```

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

EmpNo	EName	Deptno	Total_Incentive
1001	Alice	1	2000
1002	Bob	2	2500
1003	Charlie	3	1500
1004	Daisy	4	3000
1005	Edward	1	1800

-- question 3 Retrieve the project names and locations of projects with employees assigned as 'Manager'

```
select P.PName, P.PLoc
From Project P, Assigned_To A
where A.Pno= P.Pno and A.Job_Role='Project Manager';

SELECT P.PName, P.PLoc
FROM Project P
JOIN Assigned_To A ON P.PNo = A.PNo
WHERE A.Job_Role = 'Project Manager';
```

Result Grid | Filter Rows:

PName	PLoc
Project Beta	Jaipur

-- question 5 Find employees who have not been assigned to any project

```
select EmpNo
from Assigned_To
where Job_Role in (select EmpNo from Assigned_To);
```

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

EmpNo

-- question 4 List departments along with the number of employees in each department

```
select D.DName, D.Deptno,count(E.EmpNo) as No_of_Employees
from Dept D, Employee E
where D.Deptno=E.Deptno
group by D.Deptno,D.DName;
```

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

DName	Deptno	No_of_Employees
Human Resources	1	2
Finance	2	1
Research	3	1
IT	4	1
Marketing	5	1

-- question 6 List all employees along with their department names and location

```
select E.EName, D.Deptno, D.DLoc
from Employee E,Dept D
where D.Deptno=e.Deptno;
```

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

EName	Deptno	DLoc
Alice	1	Bengaluru
Bob	2	Jaipur
Charlie	Bob	Bengaluru
Daisy	4	Hyderabad
Edward	1	Bengaluru
Fiona	5	Mysuru

-- question 7 Retrieve the details of employees who work under a specific manager (e.g., manager with empno = 1005)

```
select E.EName, E.EmpNo ,E.Salary,E.Hiredate,E.Deptno
from Employee E
where E.Mgr_no = '1005';
```

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

EName	EmpNo	Salary	Hiredate	Deptno
Alice	1001	70000	2023-01-15	1
Bob	1002	85000	2022-04-10	2
NULL	NULL	NULL	NULL	NULL

-- queston 8 List all projects that have employees assigned and the number of employees on each project:

```
select P.PNo, p.PName, count(E.EmpNo) as No_of_Employees
from Project P
join Assigned_To A on P.PNo=A.PNo
join Employee E on E.EmpNo=A.EmpNo
group by P.PNo, p.PName;
```

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

PNo	PName	No_of_Employees
101	Project Alpha	1
102	Project Beta	1
103	Project Gamma	1
104	Project Delta	1
105	Project Epsilon	1
106	Project Zeta	1

List the total number of incentives given to each employee and the sum of incentives for each:

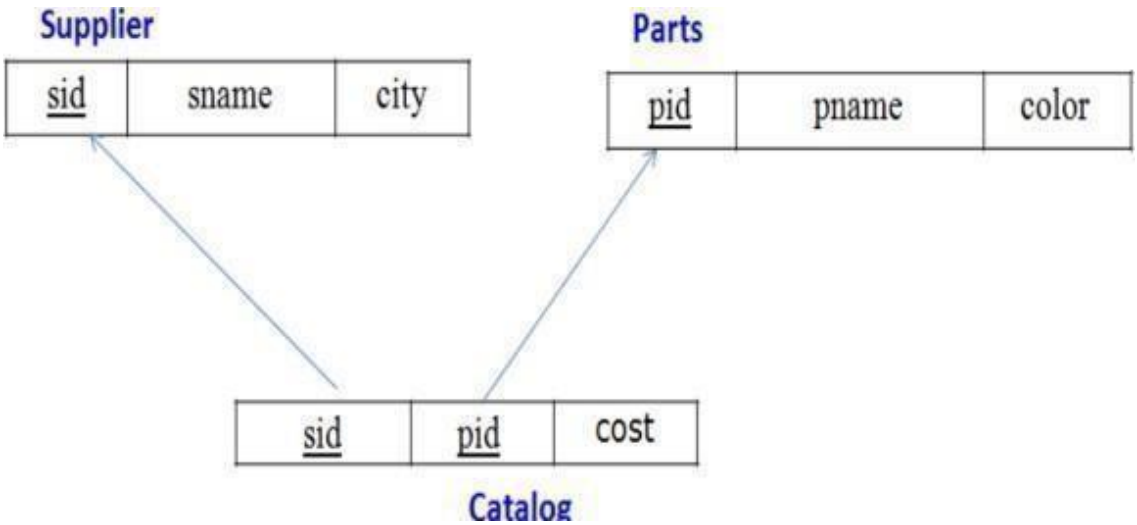
select empno, count(incentive_date) as number_of_times, sum(incentive_amt) as total_amt from incentives group by empno;

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

SUPPLIERS DATABASE (WEEK -07) QUESTION

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

Schema Diagram:



Create Database:

Create database supp;
Use supp;

Create Table:

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

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

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

Structure of the Table:

desc Supplier;

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

desc Parts;

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

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

desc catalog;

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

Inserting Values to the tables:

insert into Supplier values (10001, 'Acme_Widget', 'Bangalore'),
(10002, 'Johns', 'Kolkata'),
(10003, 'Vimal', 'Mumbai'),
(10004, 'Reliance', 'Delhi');
select * from Supplier;

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

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

p_id	p_name	color
20001	Book	Red
20002	Pen	Red
20003	Pencil	Green
20004	Mobile	Green
20005	Charger	Black
NULL	NULL	NULL

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

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

Queries:

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

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

p_name
Book
Pen
Pencil
Mobile
Charger

Find the snames of suppliers who supply every part.

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

s_name
Acme_Widget

Find the snames of suppliers who supply every red part.

```
SELECT DISTINCT s.s_name
FROM Supplier s
JOIN Catalog c ON s.s_id = c.s_id
```

```
WHERE c.p_id IN (
    SELECT p.p_id
    FROM Parts p
    WHERE p.color = 'Red'
);
```

	s_name
▶	Acme_Widget
	Johns

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

```
SELECT DISTINCT p.p_name
FROM Parts p
JOIN Catalog c ON p.p_id = c.p_id
JOIN Supplier s ON c.s_id = s.s_id
WHERE s.s_name = 'Acme_Widget'
  AND p.p_id NOT IN (
      SELECT c.p_id
      FROM Catalog c
      JOIN Supplier s ON c.s_id = s.s_id
      WHERE s.s_name != 'Acme_Widget'
  );
```

	p_name
▶	Mobile
	Charger

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

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

s_name	cost	p_id
Acme_Widget	10	20001
Acme_Widget	10	20002
Acme_Widget	10	20004
Acme_Widget	10	20005
Johns	10	20001
Johns	20	20002
Reliance	40	20003

NO SQL STUDENT DATABASE (WEEK -08) QUESTION

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

Create Database:

db.createCollection("Student");

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

Inserting Values to the tables:

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

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

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

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

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

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

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

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

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

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

Queries:

db.Student.find()

```
Atlas atlas-cci5oy-shard-0 [primary] test> db.Student.find()
[
  {
    _id: ObjectId("6746b3bd35240699686244 99"),
    RollNo: 1,
    Age: 21,
    Cont: 9876,
    email: 'antara.de9@gmail.com'
  },
  {
    _id: ObjectId("6746b3c73524069968624 49a"),
    RollNo: 2,
    Age: 22,
    Cont: 9976,
    email: 'anushka.de9@gmail.com'
  },
  {
    _id: ObjectId("6746b3d03524069968624 49b"),
    RollNo: 3,
    Age: 21,
    Cont: 5576,
    email: 'anubhav.de9@gmail.com'
  },
  {
    _id: ObjectId("6746b3d83524069968624 49c"),
    RollNo: 4,
    Age: 20,
    Cont: 4476,
    email: 'pani.de9@gmail.com'
  },
  {
    _id: ObjectId("6746b3e13524069968624 49d"),
    RollNo: 10,
    Age: 23,
    Cont: 2276,
    email: 'Abhinav@gmail.com'
  },
```

Write query to update Email-Id of a student with rollno 10.
db.Student.update({RollNo:10},{$set:{email:"Abhinav@gmail.com"}})

```
DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 0,
  upsertedCount: 0
}
```

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

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

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

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

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

```
{
  _id: ObjectId("6746b4193524069968624 49e"),
  RollNo: 11,
  Age: 22,
  Name: 'FEM',
  Cont: 2276,
  email: 'rea.de9@gmail.com'
},
```

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

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

NO SQL CUSTOMERS DATABASE

(WEEK09)

QUESTION

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

Insert at least 5 values into the table

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

Determine Minimum and Maximum account balance for each customer_id.

Export the created collection into local file system

Drop the table

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

Create Database:

db.createCollection("Customer");

```
{ ok: 1 }
```

Inserting Values to the tables:

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

```
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("675fe7b5f2355f925cc449cf"),
    '1': ObjectId("675fe7b5f2355f925cc449d0"),
    '2': ObjectId("675fe7b5f2355f925cc449d1"),
    '3': ObjectId("675fe7b5f2355f925cc449d2"),
    '4': ObjectId("675fe7b5f2355f925cc449d3")
  }
}
```

Queries:

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

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

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

Determine Minimum and Maximum account balance for each customer_id.

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

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

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

db.Customers.drop()

```
true
```

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

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

NO SQL RESTAURANTS DATABASE

(WEEK-10)

QUESTION

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

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

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

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

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

Create Database:

db.createCollection("restaurants");

```
{ ok: 1 }
```

Inserting Values to the tables:

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

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

Queries:

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

db.restaurants.find({})

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

Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns db.restaurants.find({}).sort({ name: -1 })

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

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

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

```
[
  {
    _id: ObjectId("67600441f2355f925cc449d4"),
    name: 'Meghna Foods',
    town: 'Jayanagar',
    cuisine: 'Indian'
  },
  {
    _id: ObjectId("67600441f2355f925cc449d5"),
    name: 'Empire',
    town: 'MG Road',
    cuisine: 'Indian'
  },
  {
    _id: ObjectId("67600441f2355f925cc449d7"),
    name: 'Kyotos',
    town: 'Majestic',
    cuisine: 'Japanese'
  },
  {
    _id: ObjectId("67600441f2355f925cc449d8"),
    name: 'WOW Momos',
    town: 'Malleshwaram',
    cuisine: 'Indian'
  }
]
```

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

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

```
[
  { _id: 'Meghna Foods', average_score: 8 },
  { _id: 'Kyotos', average_score: 9 },
  { _id: 'Chinese WOK', average_score: 12 },
  { _id: 'WOW Momos', average_score: 5 },
  { _id: 'Empire', average_score: 7 }
]
```

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

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

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