Sprint Evaluation - 3 SB101
Jai Phookan

Question 3)

```
create table Employee(primary key (id) int, name varchar(25), salary int, managerID int);
create table Employee(id int, firstname varchar(20), lastname varchar(20), age int, primary key (id));

create table Employee(id int, name varchar(20), salary int, managerID int, primary key (id));
insert into Employee values (1, "Joe", 70000, 3);
insert into Employee values (2, "Henry", 80000, 4);
insert into Employee values (3, "Sam", 60000, null);
insert into Employee values (4, "Max", 90000, null);
```

ANSWER:

```
select employee1.name as Employee
from Employee employee1, Employee employee2
where employee1.managerId = employee2.id and employee1.salary > employee2.salary;
```

Question 4

```
create table Employee(empID int, empName varchar(20), department varchar(20), contactNo int, emailID varchar(30), empHeadID int);
insert into Employee values (101, "Isha", "E-101", 1234567890, "isha@gmail.com", 105);
insert into Employee values (102, "Priya", "E-104", 1234567890, "priya@yahoo.com", 103);
insert into Employee values (103, "Neha", "E-101", 1234567890, "neha@gmail.com", 101);
insert into Employee values (104, "Rahul", "E-102", 1234567890, "rahul@yahoo.com", 105);
insert into Employee values (105, "Abhishek", "E-101", 1234567890, "abhishek@gmail.com", 102);

create table EmpDept(deptID varchar(20), deptName varchar(20), dept_off varchar(20), deptHead int);
insert into EmpDept values ("E-101", "HR", "Monday", 105);
insert into EmpDept values ("E-102", "Development", "Tuesday", 101);
insert into EmpDept values ("E-103", "House Keeping", "Saturday", 103);
insert into EmpDept values ("E-104", "Sales", "Sunday", 104);
insert into EmpDept values ("E-105", "Purchage", "Tuesday", 104);
```

ANSWER:

```
select empName from Employee where empID IN (select deptHead from EmpDept where deptName = "HR");
```

Question 5 Explain different types of Normalisation forms in a DBMS with an Example Table:

1st Normal Form (1NF) :

A relation is in 1NF if all repeating groups are eliminated and the table contains only atomic values. Every attribute in a relation must have atomic values in order for the relation to be in 1NF. This is violated if the relation/table contains multi-valued attributes.

Example:

Serial_No	Name	Subjects
1	Jai	English, Math
2	Shaswati	Psychology

To 1NF

Serial_No	Name	Subjects
1	**Jai**	**English**
1	**Jai**	**Math**
2	**Shaswati**	**Psychology**

2nd Normal Form (2NF)

For a relation to be in 2NF, a relation or table must be in 1NF and any non-prime attribute cannot be functionally dependent on any subset of the Candidate Key.

Example:

StoreLocation table has PK of CustomerID & StoreID. Non-Prime Attribute is City. Here, StoreLocation only depends on StoreID which is already part of candidate key. Thus it violates 2NF. So we split into 2 tables

StoreLocation:

CustomerID	StoreID	City	
1	A1	Delhi	
2	A2	Calcutta	

CustomerStoreID

CustomerID	StoreID
1	A1
2	A2

StoreCity

StoreID	City
A1	Delhi
A2	Calcutta

Third Normal Form (3NF)

For a relation to be in 3NF, it must already be in 2NF and there must not be any transitive dependency for non-prime attributes ie attributes that are not part of the Candidate Key must not be dependent on other non-prime attributes.

ID	Name	SubjectID	Subject	City
1	Jai	10	CS	Guwahati
2	Shaswati	11	Psychology	Delhi
3	Shreya	12	Political Science	London

In this table, ID determines SubjectID, and SubjectID depends on Subject. Thus ID depends on Subject as a result of SubjectID. As a result, there is transitive functional dependency and fails 3NF rules.

To get to 3NF we now divide the table

Table 1:

ID	Name	SubjectID	City
1	Jai	10	Guwahati
2	Shaswati	11	Delhi
3	Shreya	12	London

Table 2:

SubjectID	Subject
10	CS
11	Psychology
12	Political Science

Now all the non-prime attributes are dependent only on the Candidate Key and contain no transitive functional dependencies.