CSLR 51 : DBMS LAB-1

106119100, Rajneesh Pandey, CSE-B

Problem 1. Consider the Following Database:

A software company wants to track project details

Employee(Empid , Empname, Address, Doj, Salary) : Empid as Primary key

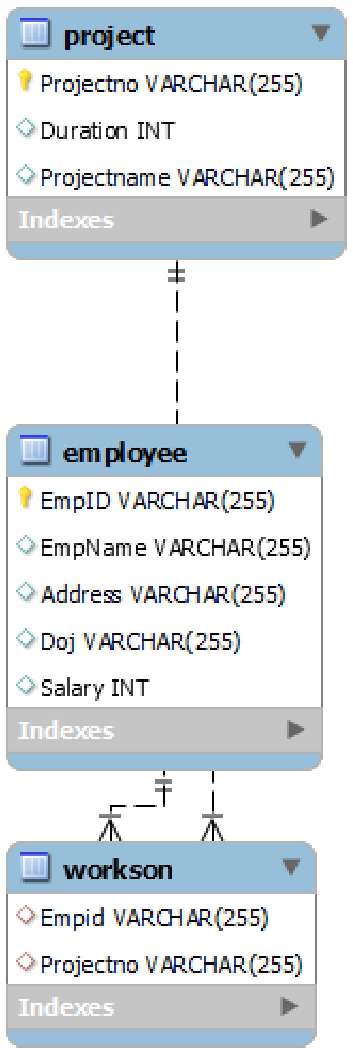
Project (Projectno, Duration, Projectname) : Project no as Primary Key

Workson(Empid,Projno) :

Empid as Foreign key references Employee

Projectno as Foreign key references Project

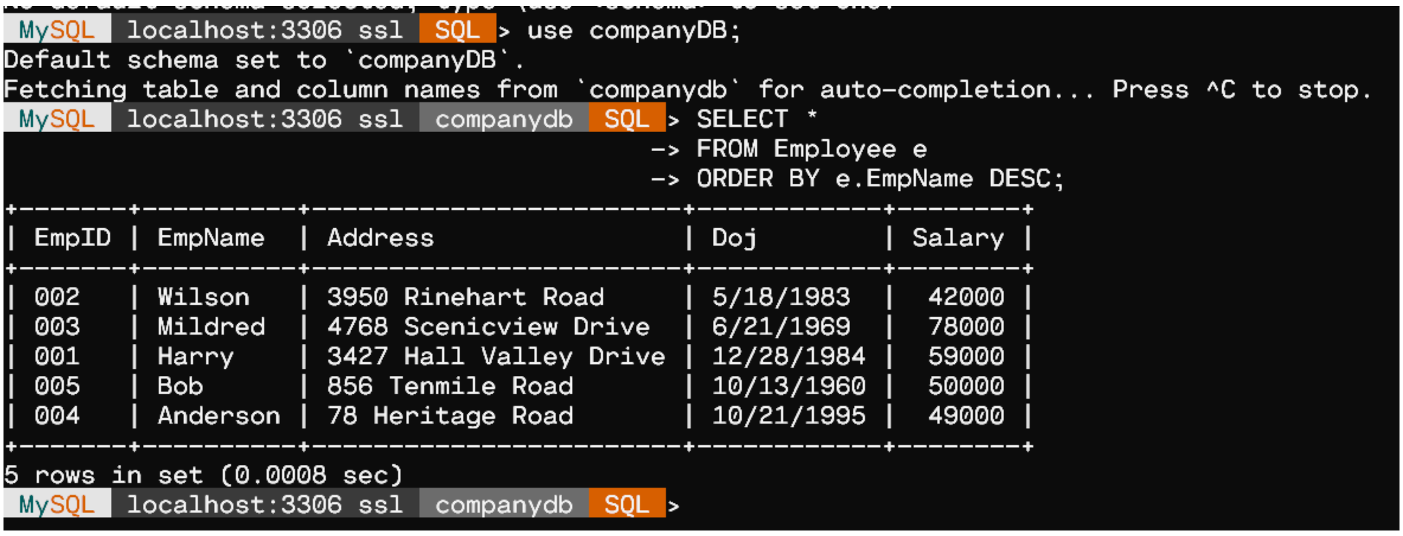
ER Diagram



```
CREATE DATABASE companyDB;
```

1. Display the Employee details in the descending order based on name.

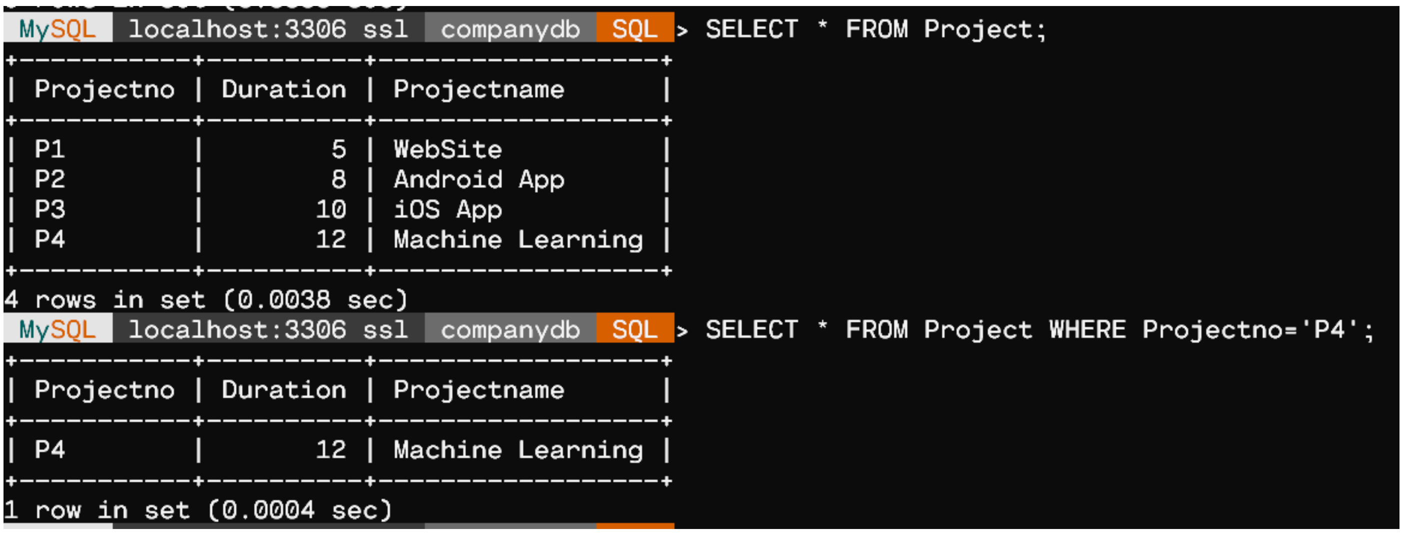
```sql
CREATE TABLE Employee(
  EmpID varchar(255) NOT NULL,
  EmpName varchar(255),
  Address varchar(255),
  Doj varchar(255),
  Salary int,
  UNIQUE (EmpID),
  PRIMARY KEY (EmpID)
  );

INSERT INTO Employee(EmpID, EmpName, Address, Doj, Salary)
VALUES
 ('001','Harry','3427 Hall Valley Drive','12/28/1984',59000),
 ('002','Wilson','3950 Rinehart Road','5/18/1983',42000),
 ('003','Mildred','4768 Scenicview Drive','6/21/1969',78000),
 ('004','Anderson','78 Heritage Road','10/21/1995',49000),
 ('005','Bob','856 Tenmile Road','10/13/1960',50000);

SELECT *
FROM Employee e
ORDER BY e.EmpName DESC;
```



```
 MySQL  localhost:3306 ssl  SQL > use companyDB;
Default schema set to `companyDB`.
Fetching table and column names from `companydb` for auto-completion... Press ^C to stop.
 MySQL  localhost:3306 ssl  companydb  SQL > SELECT *
                                          -> FROM Employee e
                                          -> ORDER BY e.EmpName DESC;
+-------+----------+------------------------+------------+--------+
| EmpID | EmpName  | Address                | Doj        | Salary |
+-------+----------+------------------------+------------+--------+
| 002   | Wilson   | 3950 Rinehart Road     | 5/18/1983  |  42000 |
| 003   | Mildred  | 4768 Scenicview Drive  | 6/21/1969  |  78000 |
| 001   | Harry    | 3427 Hall Valley Drive | 12/28/1984 |  59000 |
| 005   | Bob      | 856 Tenmile Road       | 10/13/1960 |  50000 |
| 004   | Anderson | 78 Heritage Road       | 10/21/1995 |  49000 |
+-------+----------+------------------------+------------+--------+
5 rows in set (0.0008 sec)
 MySQL  localhost:3306 ssl  companydb  SQL >
```

2. Display the project details if project id is given.

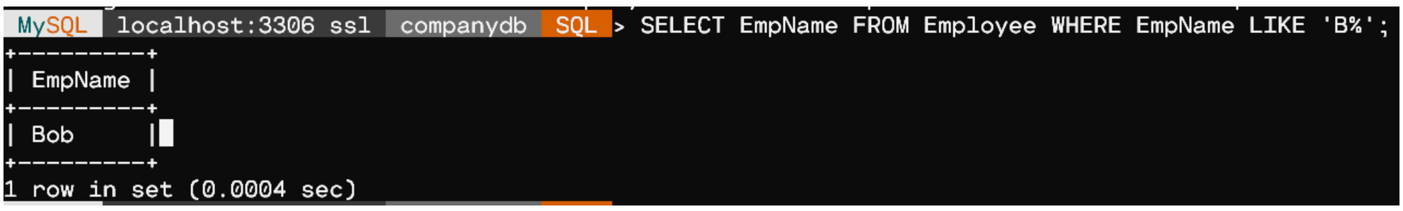
```
CREATE TABLE Project(
  Projectno varchar(255) NOT NULL,
  Duration int,
  Projectname varchar(255),
  UNIQUE (Projectno),
  PRIMARY KEY (Projectno)
  );

INSERT INTO Project(Projectno,Duration, Projectname)
VALUES
 ('P1',5,'WebSite'),
 ('P2',8,'Android App'),
 ('P3',10,'iOS App'),
 ('P4',12,'Machine Learning');

SELECT * FROM Project;
SELECT * FROM Project WHERE Projectno='P4';
```



```
 MySQL  localhost:3306 ssl  companydb  SQL > SELECT * FROM Project;
+-----------+----------+------------------+
| Projectno | Duration | Projectname      |
+-----------+----------+------------------+
| P1        |        5 | WebSite          |
| P2        |        8 | Android App      |
| P3        |       10 | iOS App          |
| P4        |       12 | Machine Learning |
+-----------+----------+------------------+
4 rows in set (0.0038 sec)
 MySQL  localhost:3306 ssl  companydb  SQL > SELECT * FROM Project WHERE Projectno='P4';
+-----------+----------+------------------+
| Projectno | Duration | Projectname      |
+-----------+----------+------------------+
| P4        |       12 | Machine Learning |
+-----------+----------+------------------+
1 row in set (0.0004 sec)
```

3. Display the employee names starting with 'B'

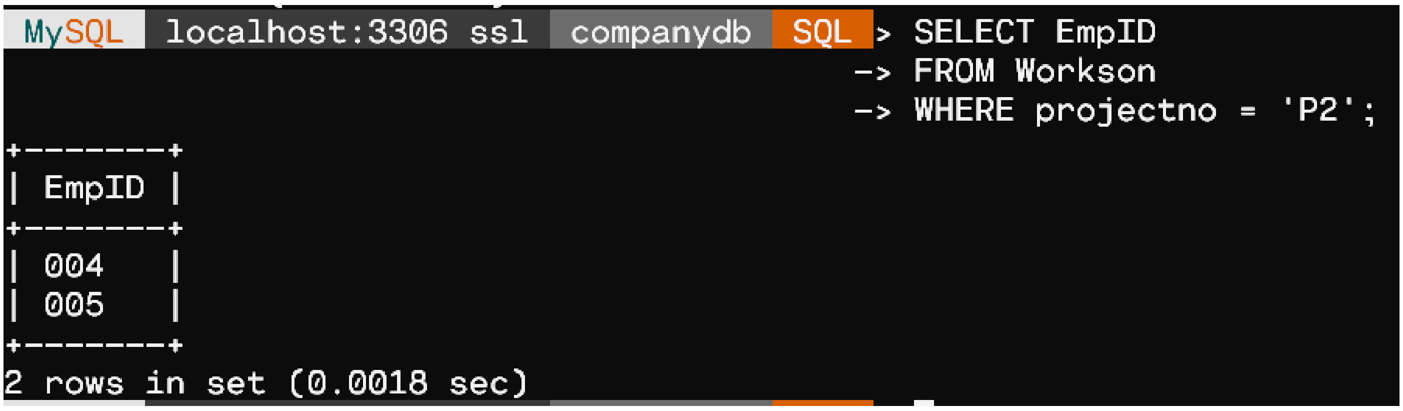
```
SELECT EmpName
from Employee
where EmpName LIKE 'B%';
```



```
 MySQL  localhost:3306 ssl  companydb  SQL > SELECT EmpName FROM Employee WHERE EmpName LIKE 'B%';
+---------+
| EmpName |
+---------+
| Bob     |
+---------+
1 row in set (0.0004 sec)
```

4. Display the employee ID's working in a particular project if project no is given.

```sql
CREATE TABLE Workson(
  Empid varchar(255),
  Projectno  varchar(255),
  FOREIGN KEY (Empid) REFERENCES Employee(Empid),
  FOREIGN KEY (Projectno) REFERENCES Project(Projectno)
  );

INSERT INTO Workson(Projectno,Empid)
VALUES
  ('P3','002'),
  ('P2','004'),
  ('P1','003'),
  ('P4','001'),
  ('P2','005');

SELECT EmpID
FROM Workson
WHERE projectno = 'P2';
```



```
MySQL  localhost:3306 ssl  companydb  SQL > SELECT EmpID
                                         -> FROM Workson
                                         -> WHERE projectno = 'P2';
+-------+
| EmpID |
+-------+
| 004   |
| 005   |
+-------+
2 rows in set (0.0018 sec)
```

Problem2. Consider the Following Database:

Student(Rollno, Name, Marks(of 6 subjects),total) : Rollno as Primary key

Department(Deptid, Deptname, HOD name) and Deptid as Primary key

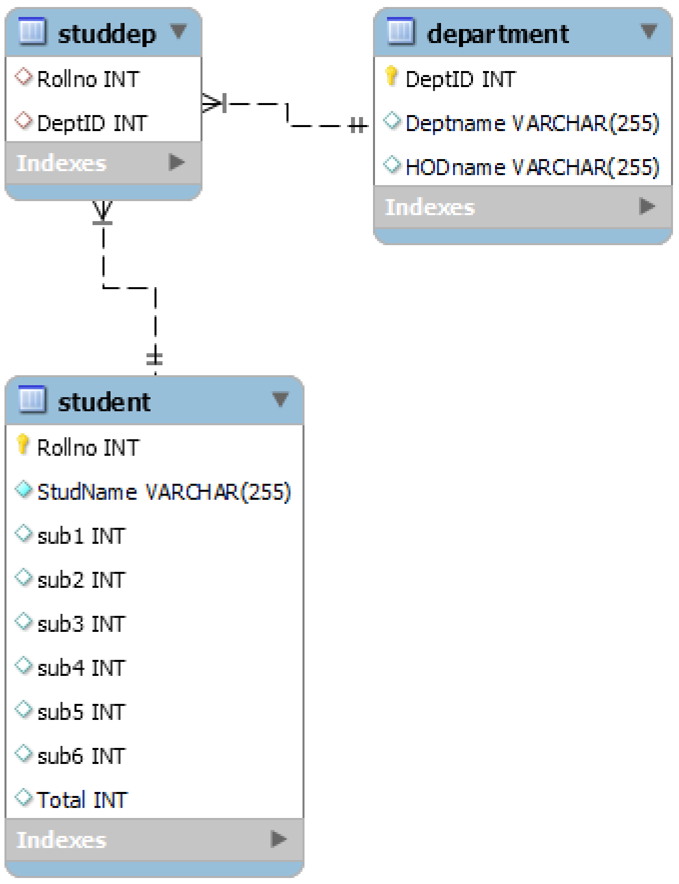
StudDep(Rollno, Deptid).

Rollno as foreign key references Student

Deptid as foreign key references Department

The total field is updated automatically

ER Diagram:



```sql
CREATE DATABASE collegeDB;
CREATE TABLE Student(
               Rollno int NOT NULL,
               StudName varchar(255) NOT NULL,
               sub1 int,
               sub2 int,
               sub3 int,
               sub4 int,
               sub5 int,
               sub6 int,
               Total int,
               UNIQUE (Rollno),
               PRIMARY KEY(Rollno)
           );

           CREATE TABLE Department(
               DeptID int NOT NULL,
               Deptname varchar(255),
               HODname varchar(255),
               UNIQUE (DeptID),
               PRIMARY KEY (DeptID)
           );
           CREATE TABLE StudDep(
               Rollno int,
               DeptID int,
               FOREIGN KEY (Rollno) REFERENCES Student(Rollno),
               FOREIGN KEY (DeptID) REFERENCES Department(DeptID)
           );
```

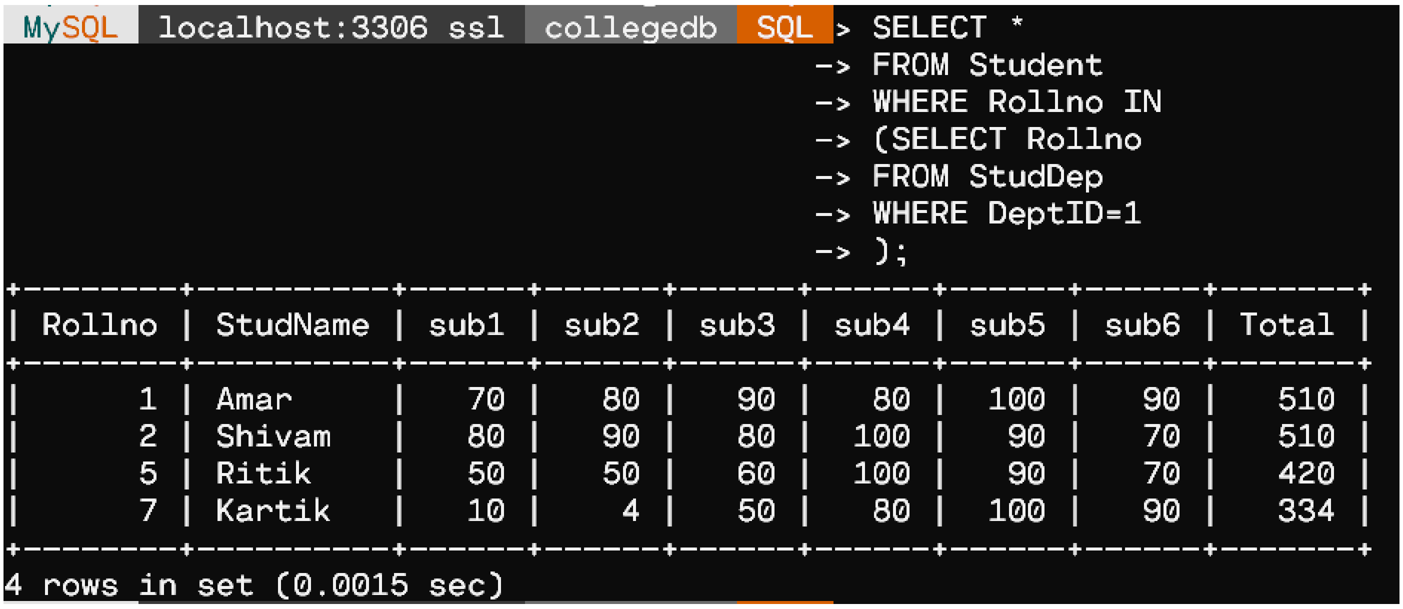
1.Insert 10 student details and 3 department details. Insert details in the studdep table.

```
INSERT INTO Student(Rollno,StudName, sub1, sub2, sub3, sub4, sub5, sub6)
VALUES
 (1,'Amar',70,80,90,80,100,90),
 (2,'Shivam',80,90,80,100,90,70),
 (3,'Radha',80,100,90,70,70,40),
 (4,'Nitin',40,60,50,80,60,70),
 (5,'Ritik',50,50,60,100,90,70),
 (6,'Vaibhav',100,100,100,100,100,90),
 (7,'Kartik',10,4,50,80,100,90),
 (8,'Ram',80,90,80,100,90,70),
 (9,'Tom',90,80,100,90,50,90),
 (10,'Katy',90,80,100,90,100,100);

SET SQL_SAFE_UPDATES = 0;

UPDATE Student SET Total = sub1 + sub2 + sub3 + sub4 + sub5 + sub6;
```

```sql
INSERT INTO Department(DeptID,Deptname,HODname)
VALUES
 (1,'CSE','Dinesh'),
 (2,'Mech','Lakshmi'),
 (3,'ECE','Surya');

INSERT INTO StudDep(Rollno,DeptID)
VALUES
 (1,1),
 (2,1),
 (3,3),
 (4,2),
 (5,1),
 (6,2),
 (7,1),
 (8,3),
 (9,3),
 (10,2);
```

2.Display the Student details if deptid is given.

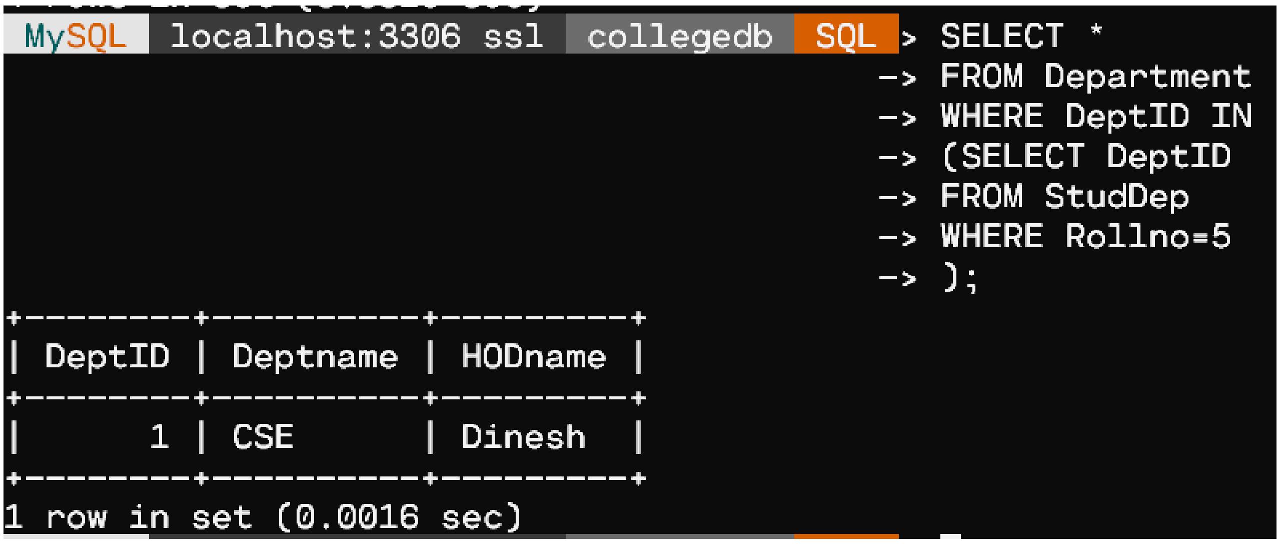
```sql
SELECT *
FROM Student
WHERE Rollno IN
(SELECT Rollno
FROM StudDep
WHERE DeptID=1
);
```



```
MySQL  localhost:3306 ssl  collegedb  SQL > SELECT *
                                        -> FROM Student
                                        -> WHERE Rollno IN
                                        -> (SELECT Rollno
                                        -> FROM StudDep
                                        -> WHERE DeptID=1
                                        -> );
+--------+----------+------+------+------+------+------+------+-------+
| Rollno | StudName | sub1 | sub2 | sub3 | sub4 | sub5 | sub6 | Total |
+--------+----------+------+------+------+------+------+------+-------+
|      1 | Amar     |   70 |   80 |   90 |   80 |  100 |   90 |   510 |
|      2 | Shivam   |   80 |   90 |   80 |  100 |   90 |   70 |   510 |
|      5 | Ritik    |   50 |   50 |   60 |  100 |   90 |   70 |   420 |
|      7 | Kartik   |   10 |    4 |   50 |   80 |  100 |   90 |   334 |
+--------+----------+------+------+------+------+------+------+-------+
4 rows in set (0.0015 sec)
```

3.Display the department details if rollno is given

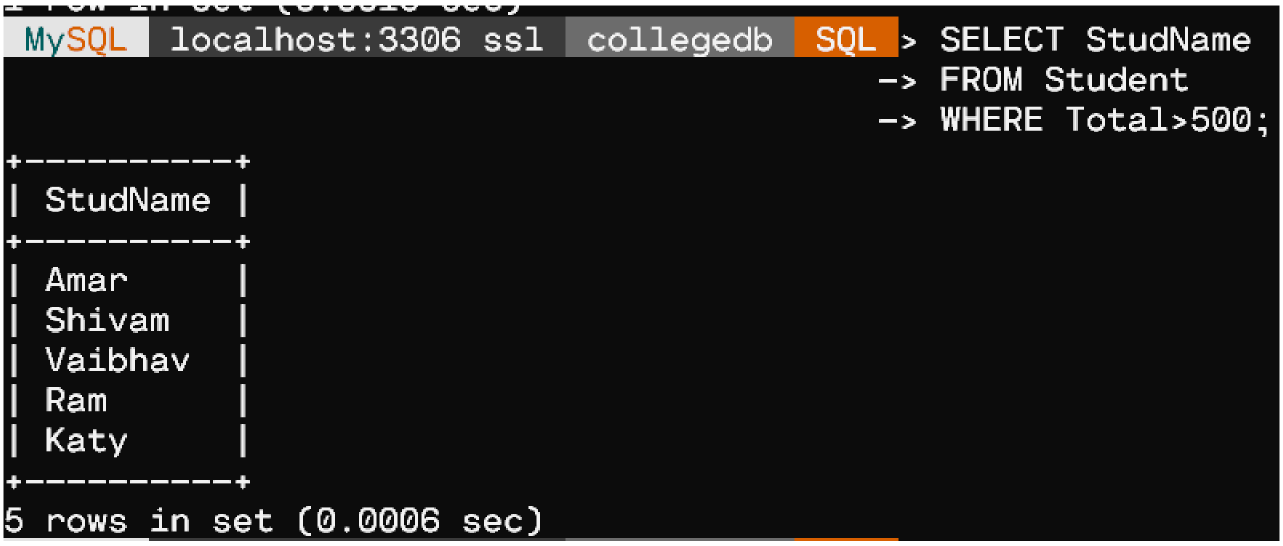
```
SELECT *
FROM Department
WHERE DeptID IN
(SELECT DeptID
FROM StudDep
WHERE Rollno=5
);
```



```
MySQL  localhost:3306 ssl  collegedb  SQL > SELECT *
                                        -> FROM Department
                                        -> WHERE DeptID IN
                                        -> (SELECT DeptID
                                        -> FROM StudDep
                                        -> WHERE Rollno=5
                                        -> );
+--------+----------+---------+
| DeptID | Deptname | HODname |
+--------+----------+---------+
|      1 | CSE      | Dinesh  |
+--------+----------+---------+
1 row in set (0.0016 sec)
```

4.Display the student names who got total greater than 500

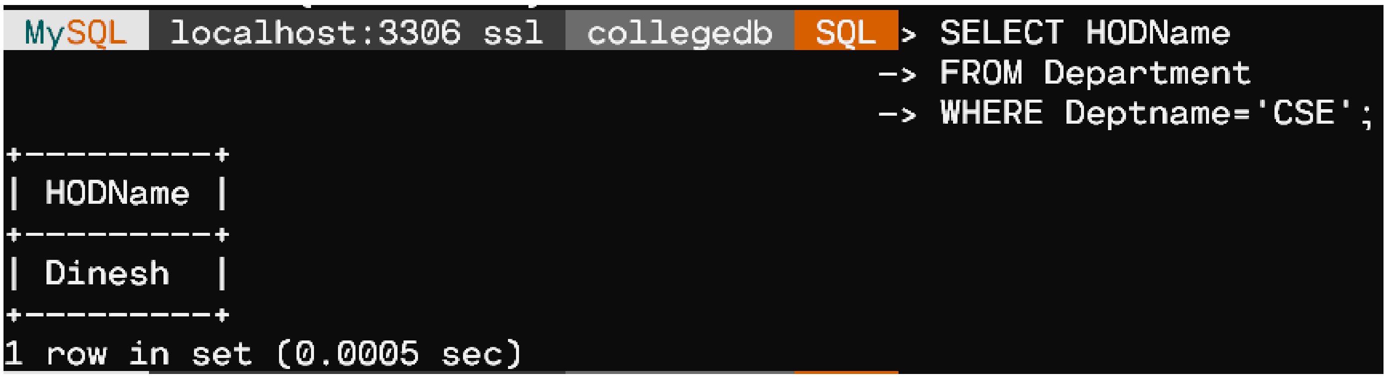
```
SELECT StudName
FROM Student
WHERE Total>500;
```



```
MySQL  localhost:3306 ssl  collegedb  SQL > SELECT StudName
                                        -> FROM Student
                                        -> WHERE Total>500;
+----------+
| StudName |
+----------+
| Amar     |
| Shivam   |
| Vaibhav  |
| Ram      |
| Katy     |
+----------+
5 rows in set (0.0006 sec)
```

5.Display the HOD name of the CSE department

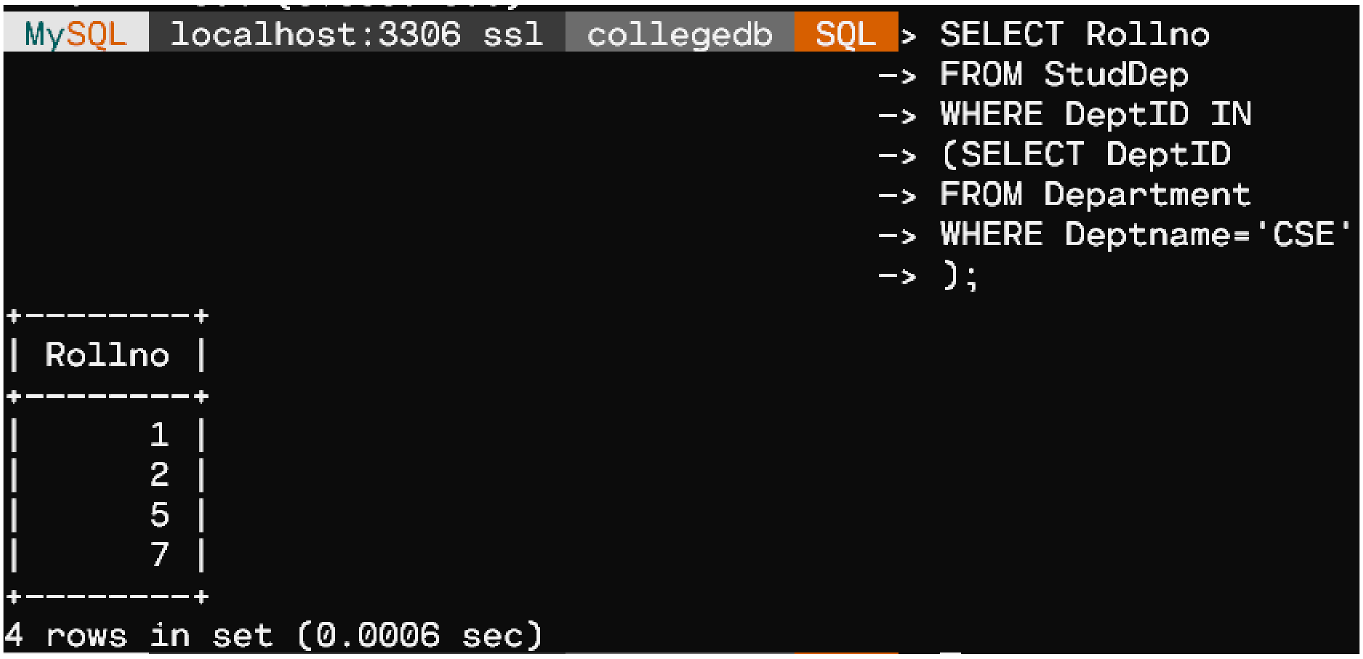
```
SELECT HODName
FROM Department
WHERE Deptname='CSE';
```



```
MySQL  localhost:3306 ssl  collegedb  SQL > SELECT HODName
                                        -> FROM Department
                                        -> WHERE Deptname='CSE';
+----------+
| HODName  |
+----------+
| Dinesh   |
+----------+
1 row in set (0.0005 sec)
```

6.Display the student rollnos of the CSE department

```
SELECT Rollno
FROM StudDep
WHERE DeptID IN
(SELECT DeptID
FROM Department
WHERE Deptname='CSE'
);
```

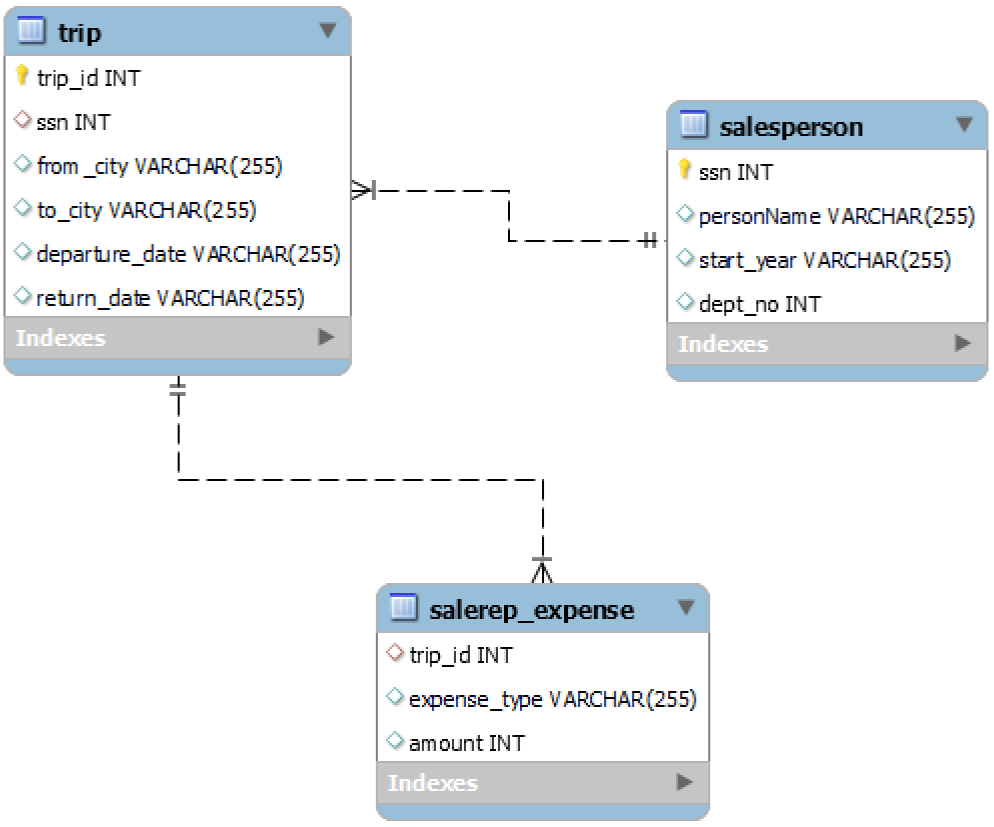


```
MySQL  localhost:3306 ssl  collegedb  SQL > SELECT Rollno
                                        -> FROM StudDep
                                        -> WHERE DeptID IN
                                        -> (SELECT DeptID
                                        -> FROM Department
                                        -> WHERE Deptname='CSE'
                                        -> );
+--------+
| Rollno |
+--------+
|      1 |
|      2 |
|      5 |
|      7 |
+--------+
4 rows in set (0.0006 sec)
```

Problem 3.Consider the Following Database:

salesperson(ssn, name, start_year, dept_no)
ssn – Primary Key
trip(ssn, from_city, to_city, departure_date, return_date, trip_id))
ssn – Foreign key
trip_id – Primary key
salerep_expense(trip_id, expense_type,amount)
trip_id – Foreign key
The expense types are 'TRAVEL', 'STAY' and 'FOOD'

ER DIAGRAM

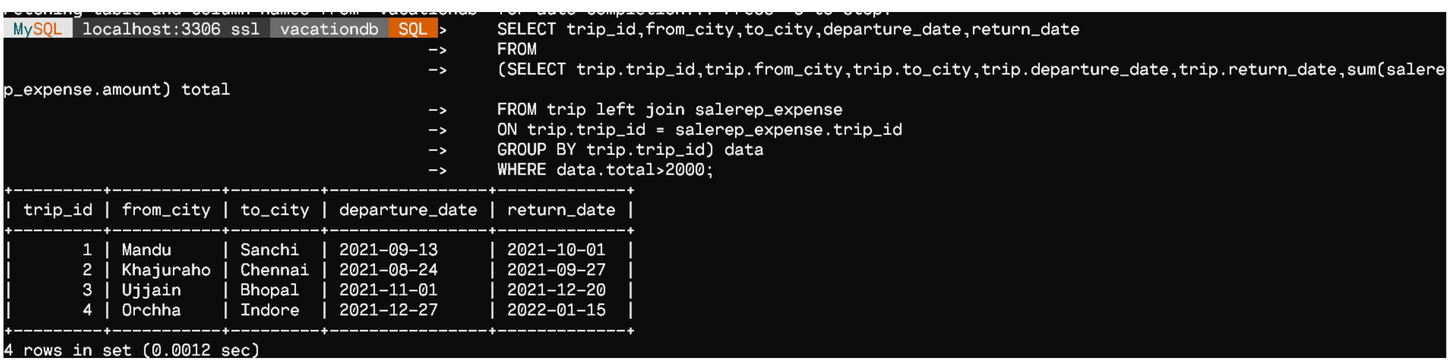


```
CREATE DATABASE vacationDB;
```

```sql
CREATE TABLE salesperson(
  ssn int NOT NULL,
  personName varchar(255),
  start_year varchar(255),
  dept_no int,
  UNIQUE (ssn),
  PRIMARY KEY (ssn)
  );

CREATE TABLE trip(
  trip_id int NOT NULL,
  ssn int,
  from_city varchar(255),
  to_city varchar(255),
  departure_date varchar(255),
  return_date varchar(255),
  UNIQUE (trip_id),
  PRIMARY KEY (trip_id),
  FOREIGN KEY (ssn) REFERENCES salesperson(ssn)
  );

CREATE TABLE salerep_expense(
  trip_id int,
  expense_type  varchar(255),
/* The expense types are ‘TRAVEL’, ‘STAY’ and ‘FOOD’ */
  amount int,
  FOREIGN KEY (trip_id) REFERENCES trip(trip_id)
  );


INSERT INTO salesperson(ssn,personName,start_year,dept_no)
VALUES
 (1000,'Harry','12/28/1984',100),
 (2000,'Wilson','5/18/1983',200),
 (3000,'Mildred','6/21/1969',300),
 (4000,'Anderson','10/21/1995',400),
 (5000,'Bob','10/13/1960',500);

INSERT INTO trip(trip_id,from_city,to_city,departure_date,return_date,ssn
)
VALUES
 (1,'Mandu','Sanchi','2021-09-13','2021-10-01',2000),
 (2,'Khajuraho','Chennai','2021-08-24','2021-09-27',1000),
 (3,'Ujjain','Bhopal','2021-11-01','2021-12-20',5000),
 (4,'Orchha','Indore','2021-12-27','2022-01-15',3000),
 (5,'Pachmarhi','Chennai','2021-11-23','2021-12-20',4000),
 (6,'Jammu','Chennai','2021-11-20','2021-12-20',4000);

INSERT INTO salerep_expense(trip_id,expense_type,amount)
VALUES
 (1,'TRAVEL',2000),
 (2,'TRAVEL',1500),
 (3,'TRAVEL',2100),
 (4,'TRAVEL',1300),
 (5,'TRAVEL',1800),
 (6,'TRAVEL',1000),
 (1,'STAY',500),
 (2,'STAY',600),
 (4,'STAY',620),
 (6,'STAY',600),
 (1,'FOOD',800),
 (2,'FOOD',100),
 (3,'FOOD',700),
 (4,'FOOD',600);
```

1.Give the details(all attributes of trip relation)for trips that exceed Rs2000

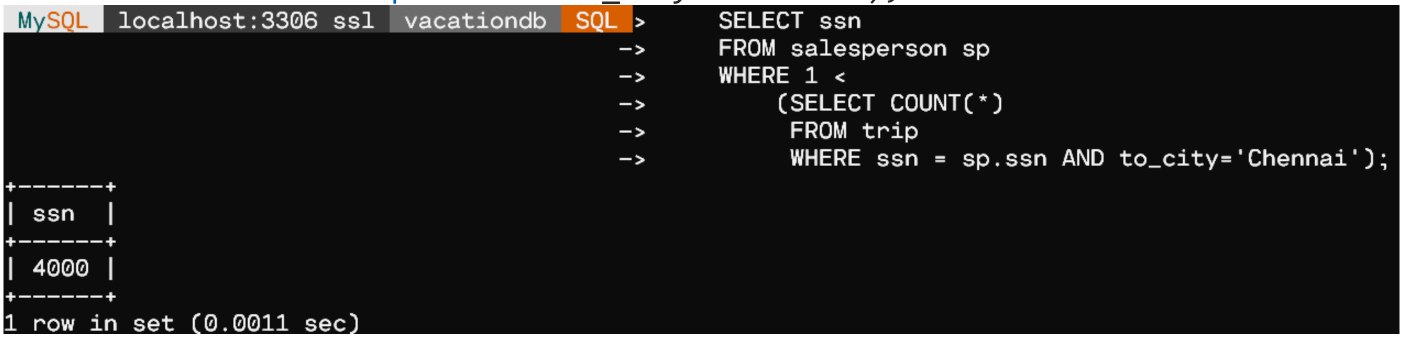
```sql
    SELECT trip_id,from_city,to_city,departure_date,return_date
    FROM
    (SELECT trip.trip_id,trip.from_city,trip.to_city,trip.departure_date,trip.r
eturn_date,sum(salerep_expense.amount) total
    FROM trip left join salerep_expense
    ON trip.trip_id = salerep_expense.trip_id
    GROUP BY trip.trip_id) data
    WHERE data.total>2000;
```



```
 MySQL  localhost:3306 ssl  vacationdb  SQL >     SELECT trip_id,from_city,to_city,departure_date,return_date
                                          ->     FROM
                                          ->     (SELECT trip.trip_id,trip.from_city,trip.to_city,trip.departure_date,trip.return_date,sum(salere
p_expense.amount) total
                                          ->     FROM trip left join salerep_expense
                                          ->     ON trip.trip_id = salerep_expense.trip_id
                                          ->     GROUP BY trip.trip_id) data
                                          ->     WHERE data.total>2000;
+---------+-----------+---------+----------------+-------------+
| trip_id | from_city | to_city | departure_date | return_date |
+---------+-----------+---------+----------------+-------------+
|       1 | Mandu     | Sanchi  | 2021-09-13     | 2021-10-01  |
|       2 | Khajuraho | Chennai | 2021-08-24     | 2021-09-27  |
|       3 | Ujjain    | Bhopal  | 2021-11-01     | 2021-12-20  |
|       4 | Orchha    | Indore  | 2021-12-27     | 2022-01-15  |
+---------+-----------+---------+----------------+-------------+
4 rows in set (0.0012 sec)
```

2.Print the ssn of salesperson who took trips to chennai more than once

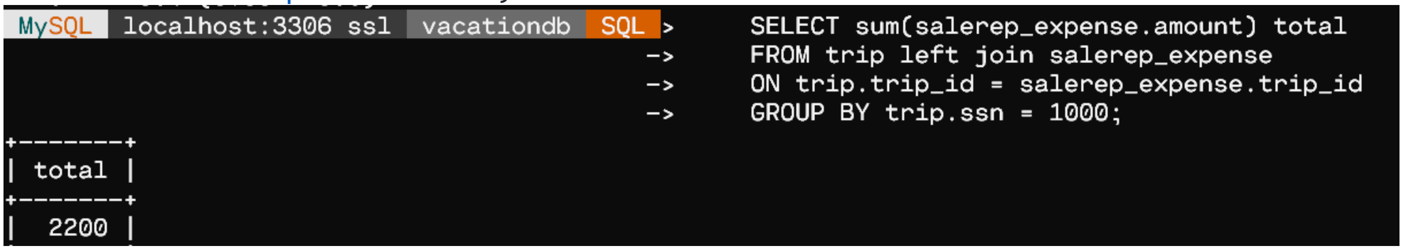
```
    SELECT ssn
    FROM salesperson sp
    WHERE 1 <
        (SELECT COUNT(*)
         FROM trip
         WHERE ssn = sp.ssn AND to_city='Chennai');
```



```
MySQL  localhost:3306 ssl  vacationdb  SQL >      SELECT ssn
                                           ->      FROM salesperson sp
                                           ->      WHERE 1 <
                                           ->          (SELECT COUNT(*)
                                           ->           FROM trip
                                           ->           WHERE ssn = sp.ssn AND to_city='Chennai');
+------+
| ssn  |
+------+
| 4000 |
+------+
1 row in set (0.0011 sec)
```

3.Print the total trip expenses incurred by the salesperson with ssn = 1000

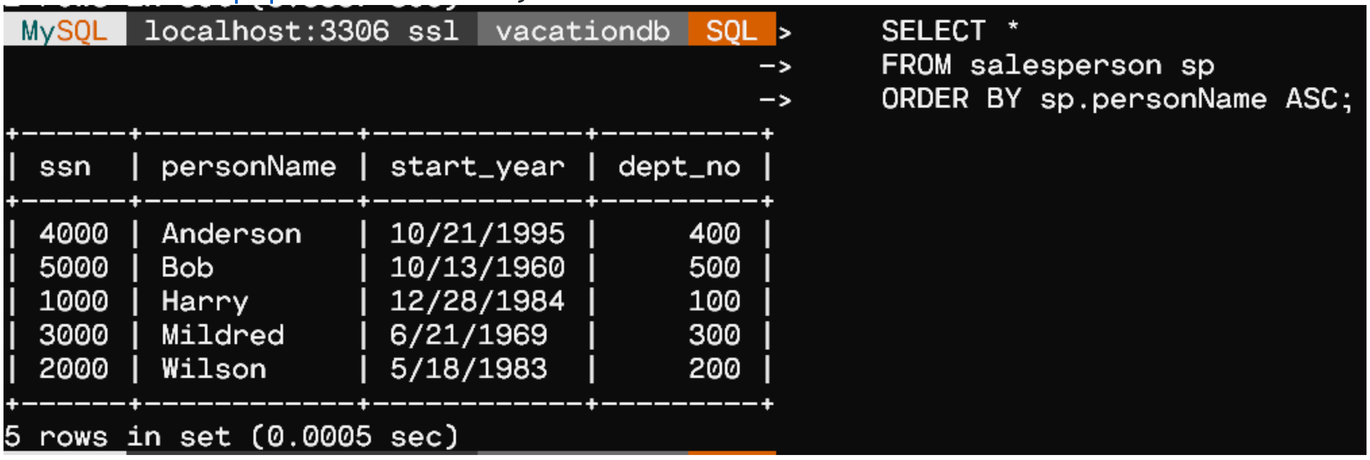
```
    SELECT sum(salerep_expense.amount) total
    FROM trip left join salerep_expense
    ON trip.trip_id = salerep_expense.trip_id
    GROUP BY trip.ssn = 1000;
```



```
MySQL  localhost:3306 ssl  vacationdb  SQL >      SELECT sum(salerep_expense.amount) total
                                           ->      FROM trip left join salerep_expense
                                           ->      ON trip.trip_id = salerep_expense.trip_id
                                           ->      GROUP BY trip.ssn = 1000;
+--------+
| total  |
+--------+
|   2200 |
```

4.Display the salesperson details in the sorted order based on name

```
    SELECT *
    FROM salesperson sp
    ORDER BY sp.personName ASC;
```

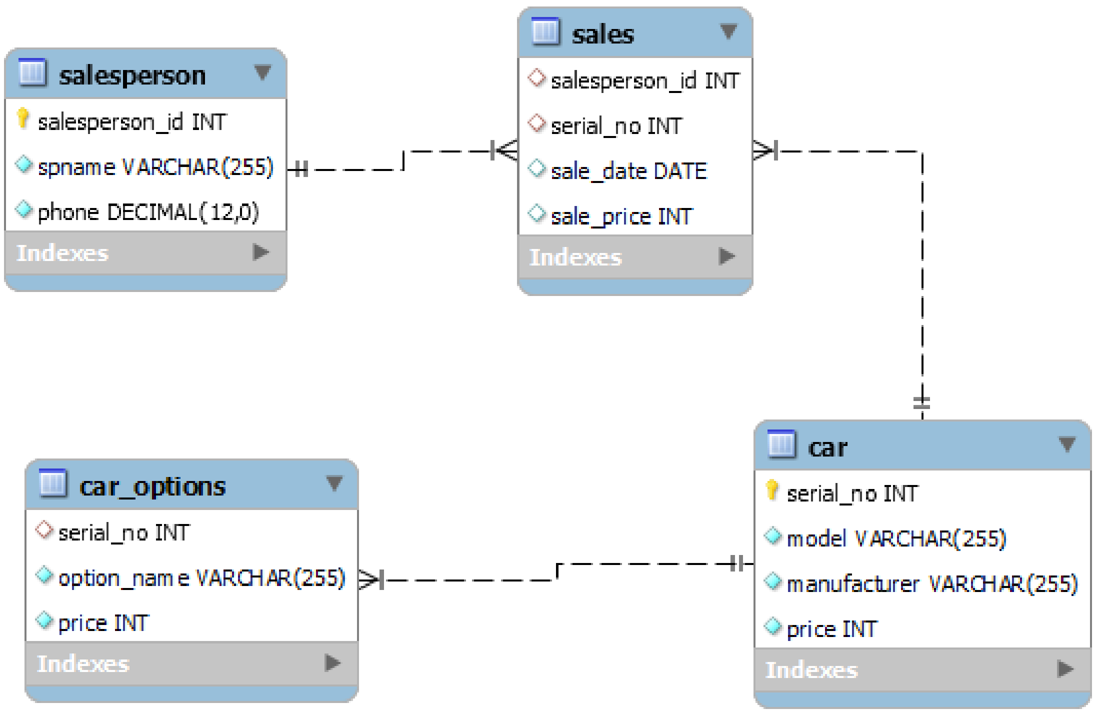


```
MySQL  localhost:3306 ssl  vacationdb  SQL >      SELECT *
                                           ->      FROM salesperson sp
                                           ->      ORDER BY sp.personName ASC;
+------+------------+------------+---------+
| ssn  | personName | start_year | dept_no |
+------+------------+------------+---------+
| 4000 | Anderson   | 10/21/1995 |     400 |
| 5000 | Bob        | 10/13/1960 |     500 |
| 1000 | Harry      | 12/28/1984 |     100 |
| 3000 | Mildred    | 6/21/1969  |     300 |
| 2000 | Wilson     | 5/18/1983  |     200 |
+------+------------+------------+---------+
5 rows in set (0.0005 sec)
```

Problem 4. Consider the Following Database:

car(serial_no, model, manufacturer, price)
serial_no – Primary key
options(serial_no, option_name, price)
serial_no – Foreign key
sales(salesperson_id, serial_no, date, sale_price)
serial_no – Foreign key
salesperson_id – Foreign key
salesperson(salesperson_id, name, phone)
salesperson_id – Primary key

ER DIAGRAM

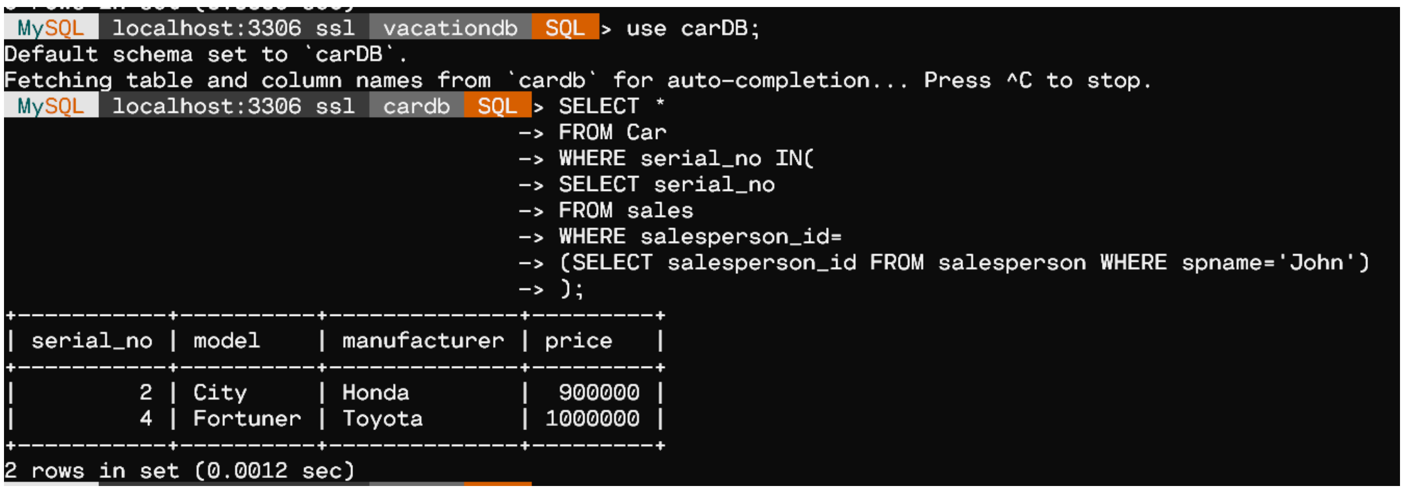


```
CREATE DATABASE carDB;


    CREATE TABLE Car(
        serial_no int NOT NULL,
        model varchar(255) NOT NULL,
        manufacturer varchar(255) NOT NULL,
        price int NOT NULL,
        UNIQUE (serial_no),
        PRIMARY KEY(serial_no)
    );
```

```sql
    CREATE TABLE salesperson(
        salesperson_id int NOT NULL,
        spname varchar(255) NOT NULL,
        phone DECIMAL(12) NOT NULL,
        UNIQUE (salesperson_id),
        PRIMARY KEY(salesperson_id)
    );

    CREATE TABLE car_options(
        serial_no int,
        option_name varchar(255) NOT NULL,
        price int NOT NULL,
        FOREIGN KEY (serial_no) REFERENCES Car(serial_no)
    );

    CREATE TABLE sales(
        salesperson_id int,
        serial_no int,
        sale_date DATE,
        sale_price int,
        FOREIGN KEY (serial_no) REFERENCES Car(serial_no),
        FOREIGN KEY (salesperson_id) REFERENCES salesperson(salesperson_id)
    );

INSERT INTO Car(serial_no,model,manufacturer,price)
VALUES
 (1,'Swift','Suzuki',700000),
 (2,'City','Honda',900000),
 (3,'Nano','Tata',500000),
 (4,'Fortuner','Toyota',1000000);

INSERT INTO salesperson(salesperson_id,spname,phone)
VALUES
 (1,'John',8168915356),
 (2,'Tom',9368570708),
 (3,'Martin',7895247308);

INSERT INTO car_options(serial_no,option_name,price)
VALUES
 (1,'Black',850000),
 (1,'Blue',760000),
 (1,'White',900000),
 (2,'Black',850000),
 (2,'Blue',760000),
 (4,'Blue',760000),
 (4,'White',900000);


INSERT INTO sales(salesperson_id,serial_no,sale_date,sale_price)
VALUES
 (2,1,'2021-01-15',850000),
 (1,2,'2021-04-07',750000),
 (3,3,'2021-03-23',600000),
 (1,4,'2021-03-12',900000);
```

1. For the sales person named 'John' list the following information for all the cars sold : serial no, manufacturer, sale_price

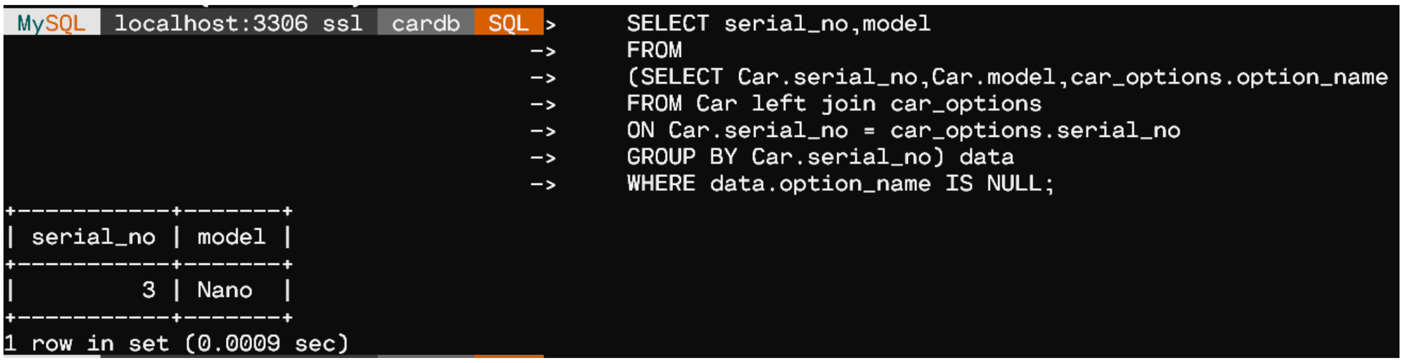
```
SELECT *
FROM Car
WHERE serial_no IN(
    SELECT serial_no
    FROM sales
    WHERE salesperson_id=
        (SELECT salesperson_id FROM salesperson WHERE spname='John')
);
```



```
 MySQL  localhost:3306 ssl  vacationdb  SQL > use carDB;
Default schema set to `carDB`.
Fetching table and column names from `cardb` for auto-completion... Press ^C to stop.
 MySQL  localhost:3306 ssl  cardb  SQL > SELECT *
                                     -> FROM Car
                                     -> WHERE serial_no IN(
                                     -> SELECT serial_no
                                     -> FROM sales
                                     -> WHERE salesperson_id=
                                     -> (SELECT salesperson_id FROM salesperson WHERE spname='John')
                                     -> );
+-----------+----------+--------------+---------+
| serial_no | model    | manufacturer | price   |
+-----------+----------+--------------+---------+
|         2 | City     | Honda        |  900000 |
|         4 | Fortuner | Toyota       | 1000000 |
+-----------+----------+--------------+---------+
2 rows in set (0.0012 sec)
```

2.List the serial_no and model of cars that have no options

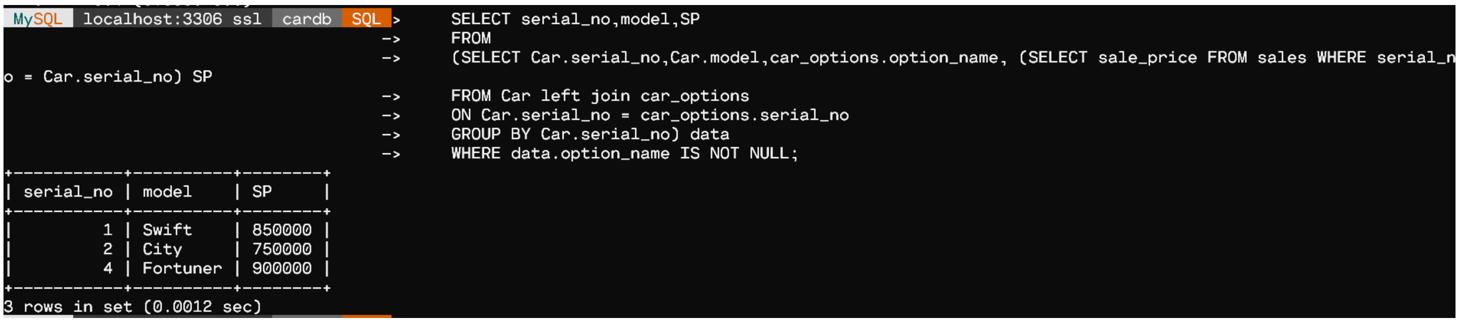
```sql
SELECT serial_no,model
FROM
(SELECT Car.serial_no,Car.model,car_options.option_name
FROM Car left join car_options
ON Car.serial_no = car_options.serial_no
GROUP BY Car.serial_no) data
WHERE data.option_name IS NULL;
```



```
 MySQL  localhost:3306 ssl  cardb  SQL >     SELECT serial_no,model
                                       ->     FROM
                                       ->     (SELECT Car.serial_no,Car.model,car_options.option_name
                                       ->     FROM Car left join car_options
                                       ->     ON Car.serial_no = car_options.serial_no
                                       ->     GROUP BY Car.serial_no) data
                                       ->     WHERE data.option_name IS NULL;
+-----------+-------+
| serial_no | model |
+-----------+-------+
|         3 | Nano  |
+-----------+-------+
1 row in set (0.0009 sec)
```

3.List the serial_no, model, sale_price for the cars that have optional parts.

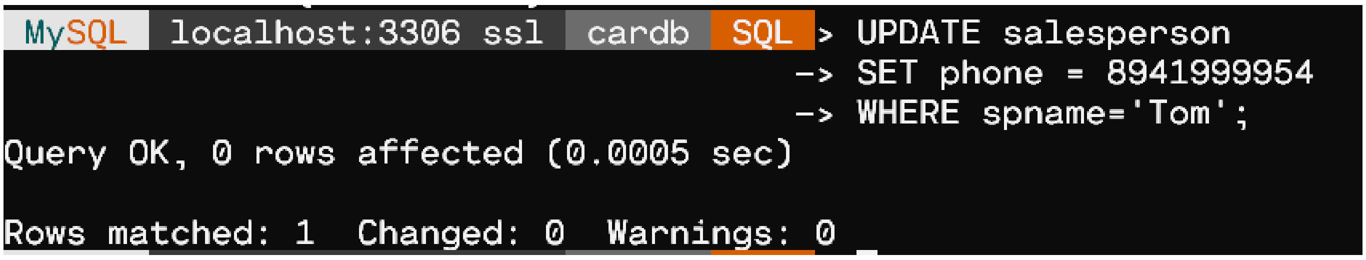
```sql
    SELECT serial_no,model,SP
    FROM
    (SELECT Car.serial_no,Car.model,car_options.option_name, (SELECT sale_price
 FROM sales WHERE serial_no = Car.serial_no) SP
    FROM Car left join car_options
    ON Car.serial_no = car_options.serial_no
    GROUP BY Car.serial_no) data
    WHERE data.option_name IS NOT NULL;
```



```
 MySQL  localhost:3306 ssl  cardb  SQL >     SELECT serial_no,model,SP
                                       ->     FROM
                                       ->     (SELECT Car.serial_no,Car.model,car_options.option_name, (SELECT sale_price FROM sales WHERE serial_n
o = Car.serial_no) SP
                                       ->     FROM Car left join car_options
                                       ->     ON Car.serial_no = car_options.serial_no
                                       ->     GROUP BY Car.serial_no) data
                                       ->     WHERE data.option_name IS NOT NULL;
+-----------+----------+--------+
| serial_no | model    | SP     |
+-----------+----------+--------+
|         1 | Swift    | 850000 |
|         2 | City     | 750000 |
|         4 | Fortuner | 900000 |
+-----------+----------+--------+
3 rows in set (0.0012 sec)
```

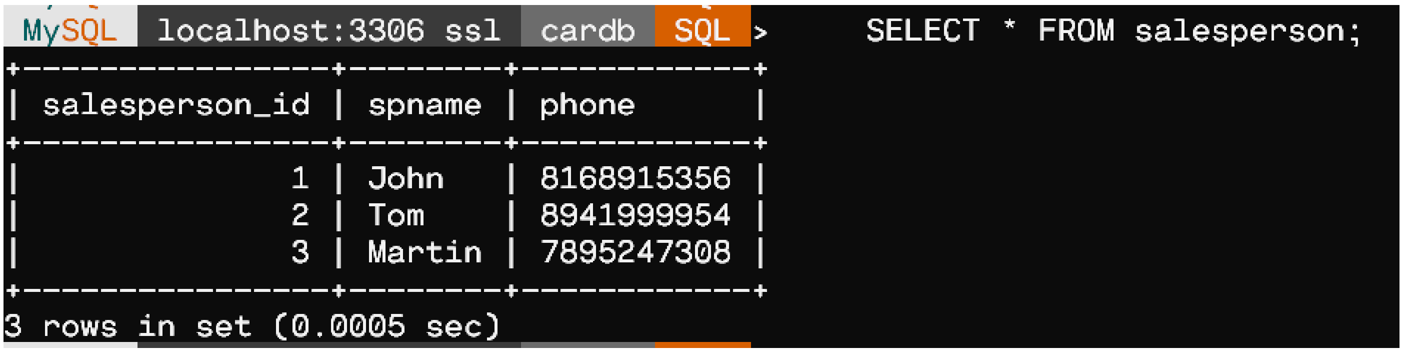
4.Modify the phone no of a particular sales person

```
UPDATE salesperson
SET phone = 8941999954
WHERE spname='Tom';
```



```
MySQL  localhost:3306 ssl  cardb  SQL > UPDATE salesperson
                                     -> SET phone = 8941999954
                                     -> WHERE spname='Tom';
Query OK, 0 rows affected (0.0005 sec)

Rows matched: 1  Changed: 0  Warnings: 0
```

```
SELECT * FROM salesperson;
```



```
MySQL  localhost:3306 ssl  cardb  SQL >     SELECT * FROM salesperson;
+----------------+--------+------------+
| salesperson_id | spname | phone      |
+----------------+--------+------------+
|              1 | John   | 8168915356 |
|              2 | Tom    | 8941999954 |
|              3 | Martin | 7895247308 |
+----------------+--------+------------+
3 rows in set (0.0005 sec)
```