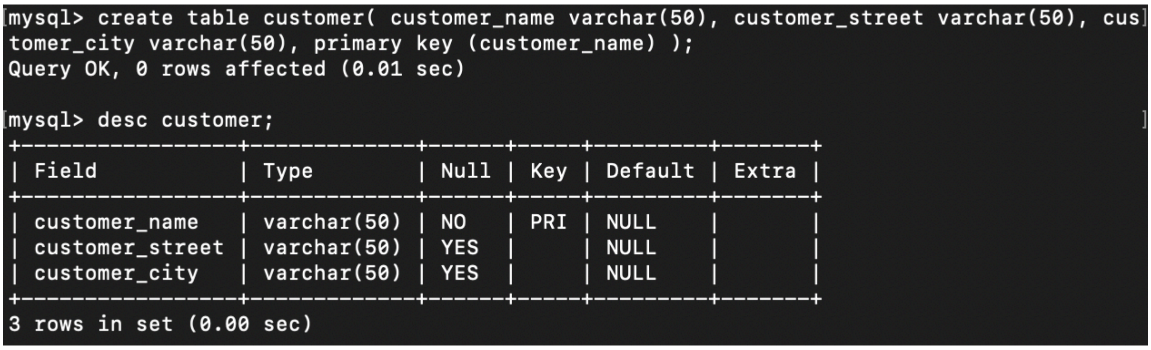
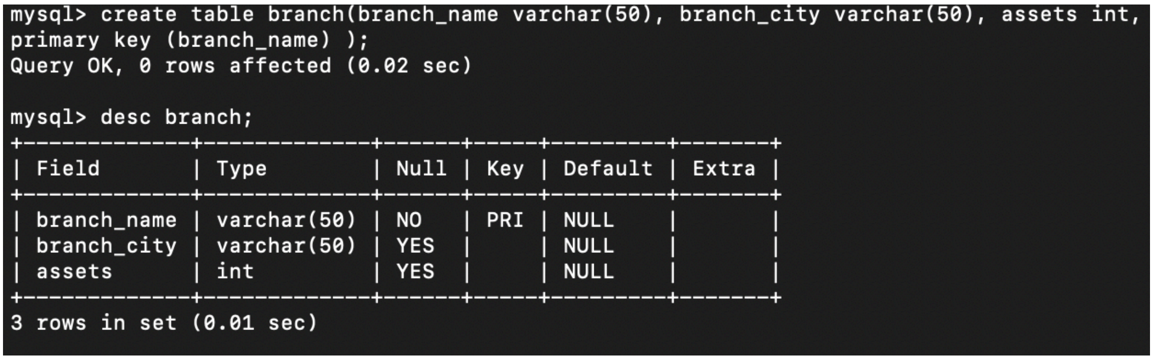
NAME – Sayan Paul
ROLL - 21CS8056
REG - 21U10209
ASSIGNMENT - 5

1. Consider the bank database. Construct the following SQL queries for this relational database.
Bank Database Schema:
branch(branch name, branch city, assets)



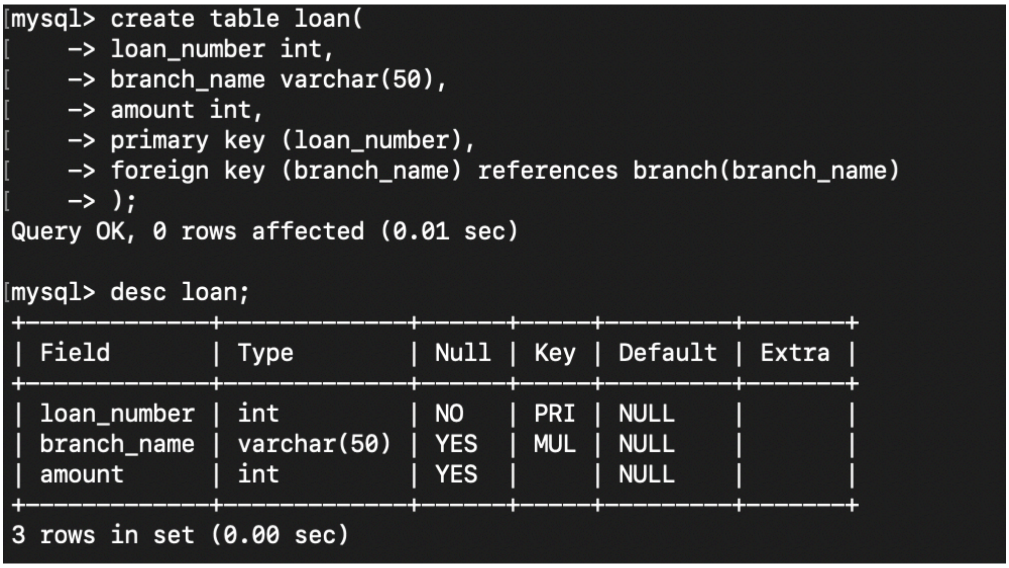
```
[mysql> create table customer( customer_name varchar(50), customer_street varchar(50), cus
tomer_city varchar(50), primary key (customer_name) );
Query OK, 0 rows affected (0.01 sec)

[mysql> desc customer;
+-----------------+-------------+------+-----+---------+-------+
| Field           | Type        | Null | Key | Default | Extra |
+-----------------+-------------+------+-----+---------+-------+
| customer_name   | varchar(50) | NO   | PRI | NULL    |       |
| customer_street | varchar(50) | YES  |     | NULL    |       |
| customer_city   | varchar(50) | YES  |     | NULL    |       |
+-----------------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

customer(customer name, customer street, customer city)



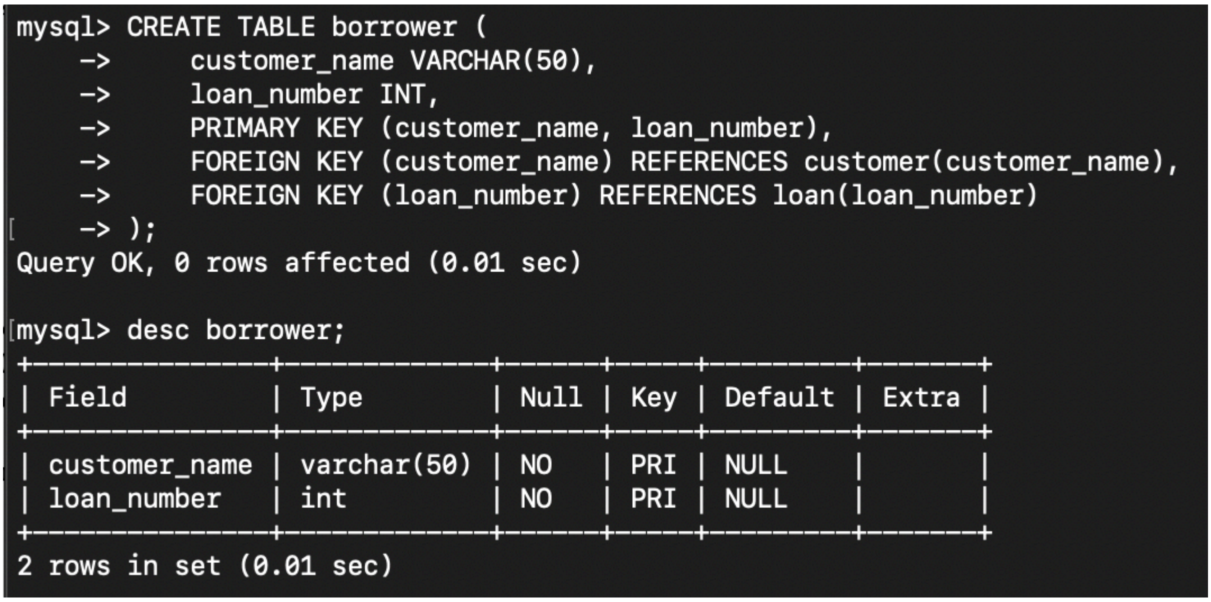
```
mysql> create table branch(branch_name varchar(50), branch_city varchar(50), assets int,
primary key (branch_name) );
Query OK, 0 rows affected (0.02 sec)

mysql> desc branch;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| branch_name | varchar(50) | NO   | PRI | NULL    |       |
| branch_city | varchar(50) | YES  |     | NULL    |       |
| assets      | int         | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

loan(loan number, branch name, amount)



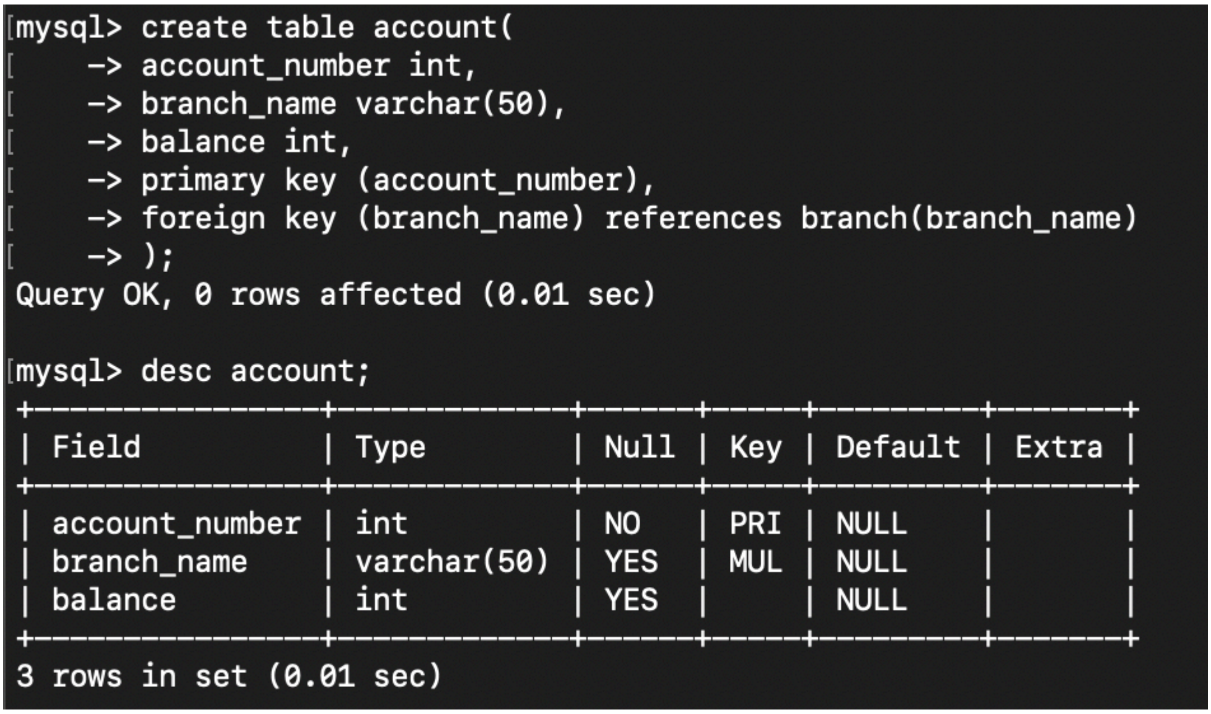
```
[mysql> create table loan(
[    -> loan_number int,
[    -> branch_name varchar(50),
[    -> amount int,
[    -> primary key (loan_number),
[    -> foreign key (branch_name) references branch(branch_name)
[    -> );
Query OK, 0 rows affected (0.01 sec)

[mysql> desc loan;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| loan_number | int         | NO   | PRI | NULL    |       |
| branch_name | varchar(50) | YES  | MUL | NULL    |       |
| amount      | int         | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

borrower(customer name, loan number)



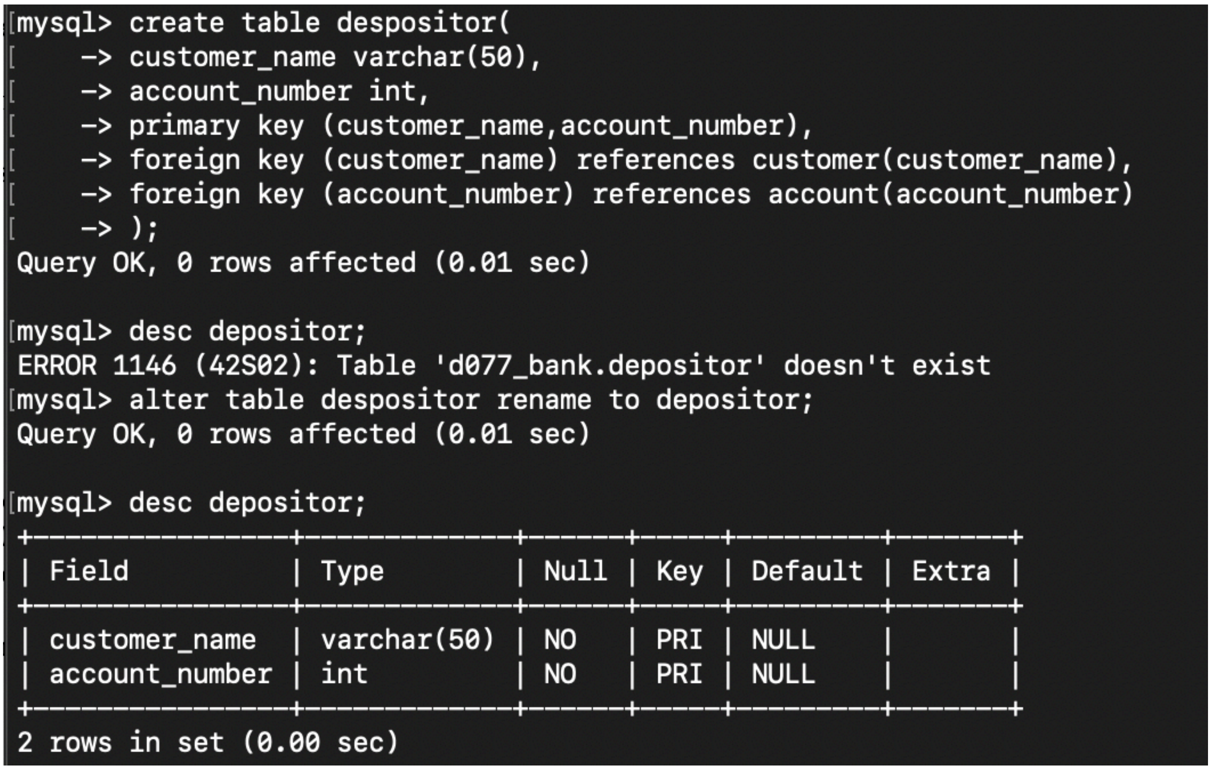
```
mysql> CREATE TABLE borrower (
    ->     customer_name VARCHAR(50),
    ->     loan_number INT,
    ->     PRIMARY KEY (customer_name, loan_number),
    ->     FOREIGN KEY (customer_name) REFERENCES customer(customer_name),
    ->     FOREIGN KEY (loan_number) REFERENCES loan(loan_number)
[   -> );
Query OK, 0 rows affected (0.01 sec)

[mysql> desc borrower;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| customer_name | varchar(50) | NO   | PRI | NULL    |       |
| loan_number   | int         | NO   | PRI | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
2 rows in set (0.01 sec)
```

account(account number, branch name, balance)



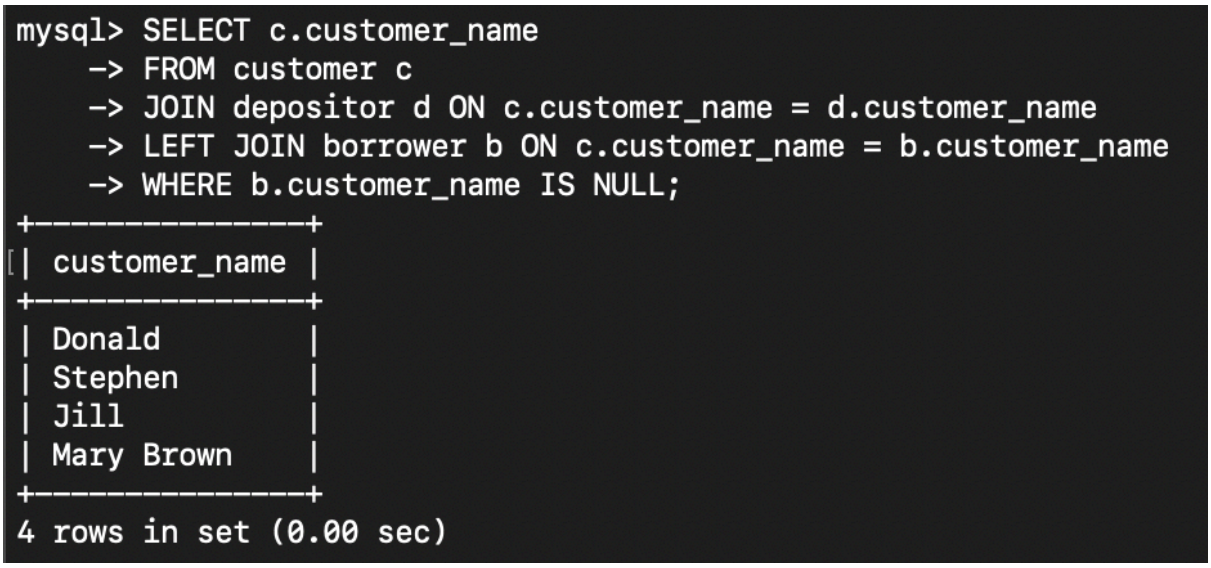
```
[mysql> create table account(
[    -> account_number int,
[    -> branch_name varchar(50),
[    -> balance int,
[    -> primary key (account_number),
[    -> foreign key (branch_name) references branch(branch_name)
[    -> );
Query OK, 0 rows affected (0.01 sec)

[mysql> desc account;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| account_number | int         | NO   | PRI | NULL    |       |
| branch_name    | varchar(50) | YES  | MUL | NULL    |       |
| balance        | int         | YES  |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

depositor(customer name, account number)



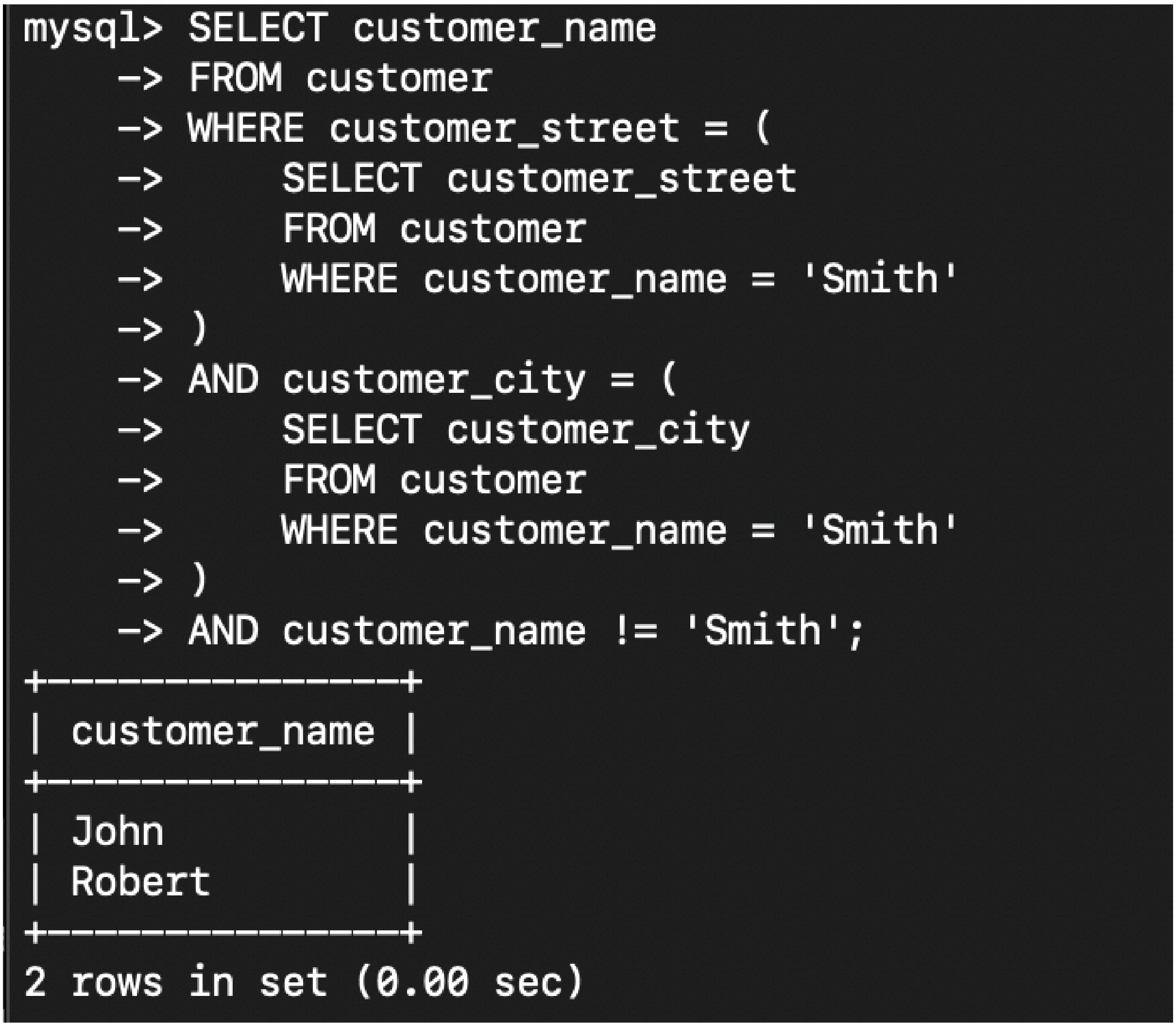
```
[mysql> create table despositor(
[    -> customer_name varchar(50),
[    -> account_number int,
[    -> primary key (customer_name,account_number),
[    -> foreign key (customer_name) references customer(customer_name),
[    -> foreign key (account_number) references account(account_number)
[    -> );
Query OK, 0 rows affected (0.01 sec)

[mysql> desc depositor;
ERROR 1146 (42S02): Table 'd077_bank.depositor' doesn't exist
[mysql> alter table despositor rename to depositor;
Query OK, 0 rows affected (0.01 sec)

[mysql> desc depositor;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| customer_name  | varchar(50) | NO   | PRI | NULL    |       |
| account_number | int         | NO   | PRI | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

a. Find all customers of the bank who have an account but not a loan.



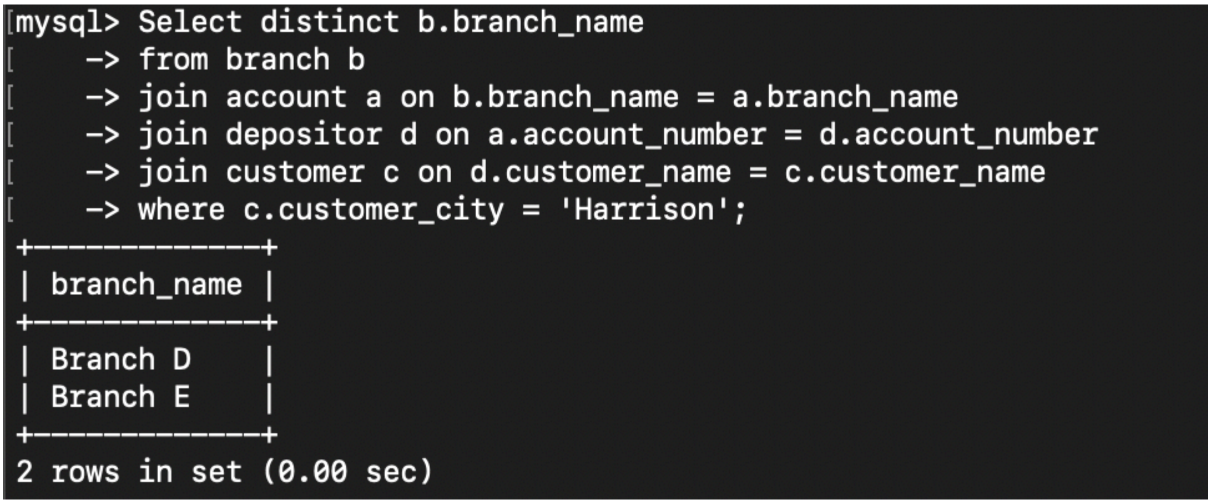
```
mysql> SELECT c.customer_name
    -> FROM customer c
    -> JOIN depositor d ON c.customer_name = d.customer_name
    -> LEFT JOIN borrower b ON c.customer_name = b.customer_name
    -> WHERE b.customer_name IS NULL;
+---------------+
[| customer_name |
+---------------+
| Donald        |
| Stephen       |
| Jill          |
| Mary Brown    |
+---------------+
4 rows in set (0.00 sec)
```

b. Find the names of all customers who live on the same street and in the same city as “Smith”.



```
mysql> SELECT customer_name
    -> FROM customer
    -> WHERE customer_street = (
    ->     SELECT customer_street
    ->     FROM customer
    ->     WHERE customer_name = 'Smith'
    -> )
    -> AND customer_city = (
    ->     SELECT customer_city
    ->     FROM customer
    ->     WHERE customer_name = 'Smith'
    -> )
    -> AND customer_name != 'Smith';
+---------------+
| customer_name |
+---------------+
| John          |
| Robert        |
+---------------+
2 rows in set (0.00 sec)
```

c. Find the names of all branches with customers who have an account in the bank and who live in “Harrison”.

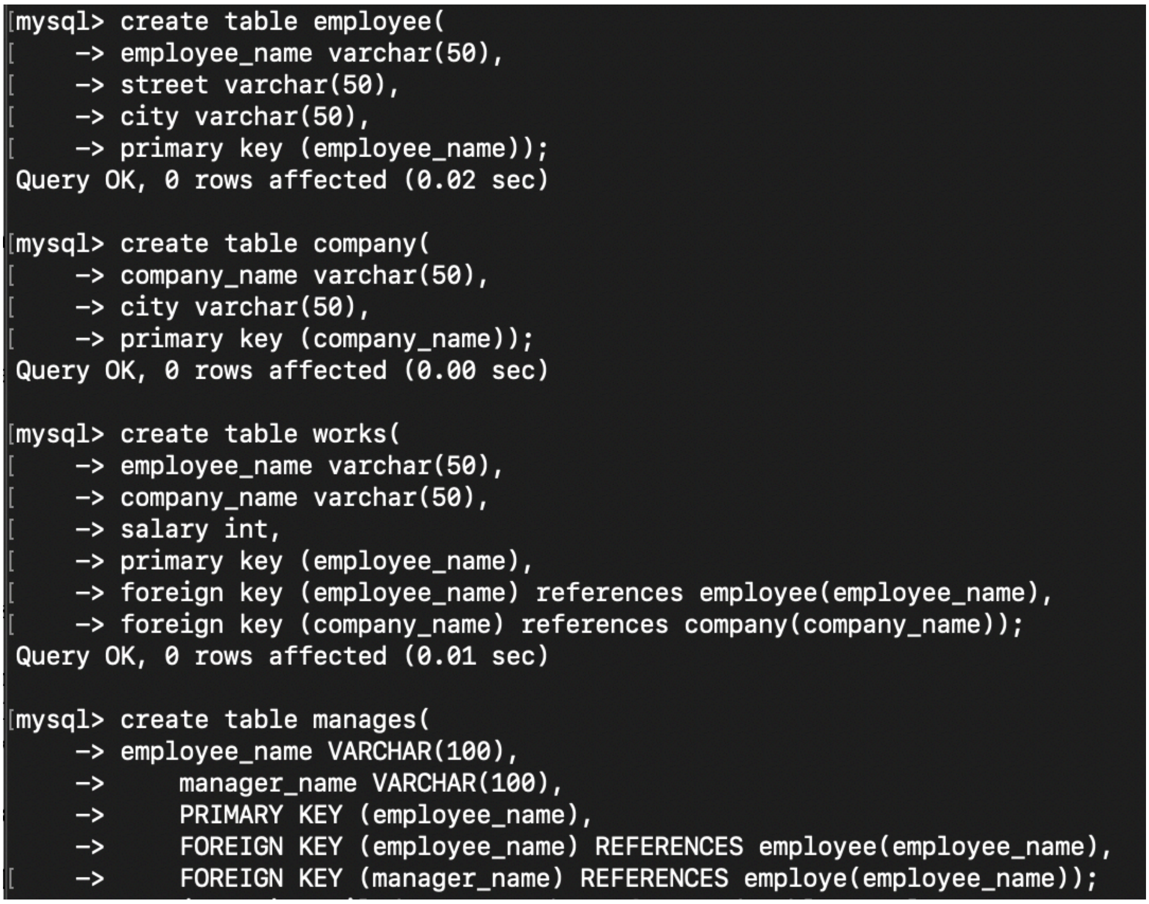


```
[mysql> Select distinct b.branch_name
[    -> from branch b
[    -> join account a on b.branch_name = a.branch_name
[    -> join depositor d on a.account_number = d.account_number
[    -> join customer c on d.customer_name = c.customer_name
[    -> where c.customer_city = 'Harrison';
+-------------+
| branch_name |
+-------------+
| Branch D    |
| Branch E    |
+-------------+
2 rows in set (0.00 sec)
```

2. Consider the employee database. Give an expression in SQL for each of the following queries.

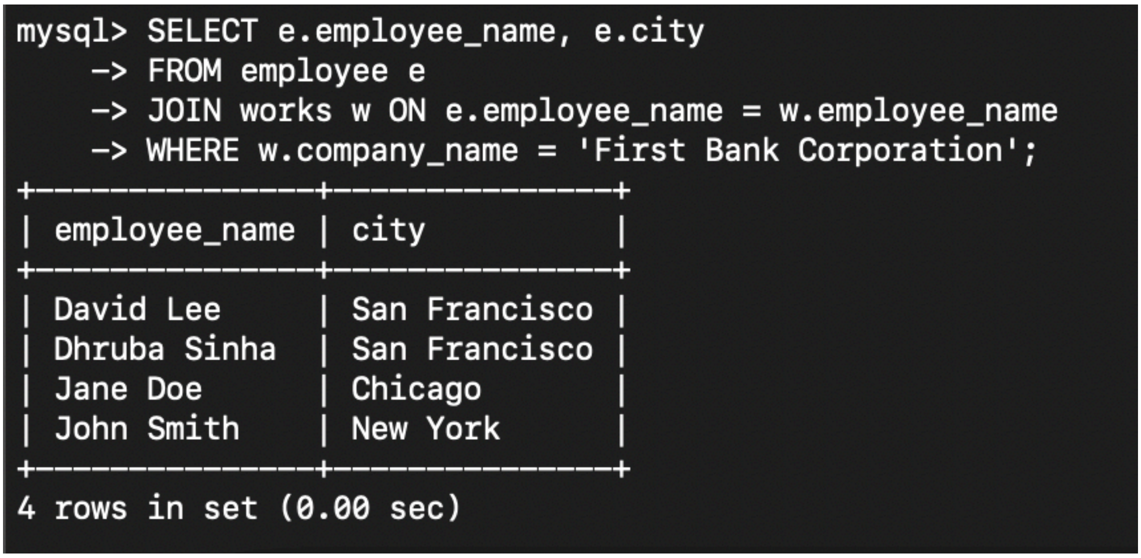
Employee Database Schema:

employee(employee name, street, city)
works(employee name, company name, salary)
company(company name, city)
manages(employee name, manager name)



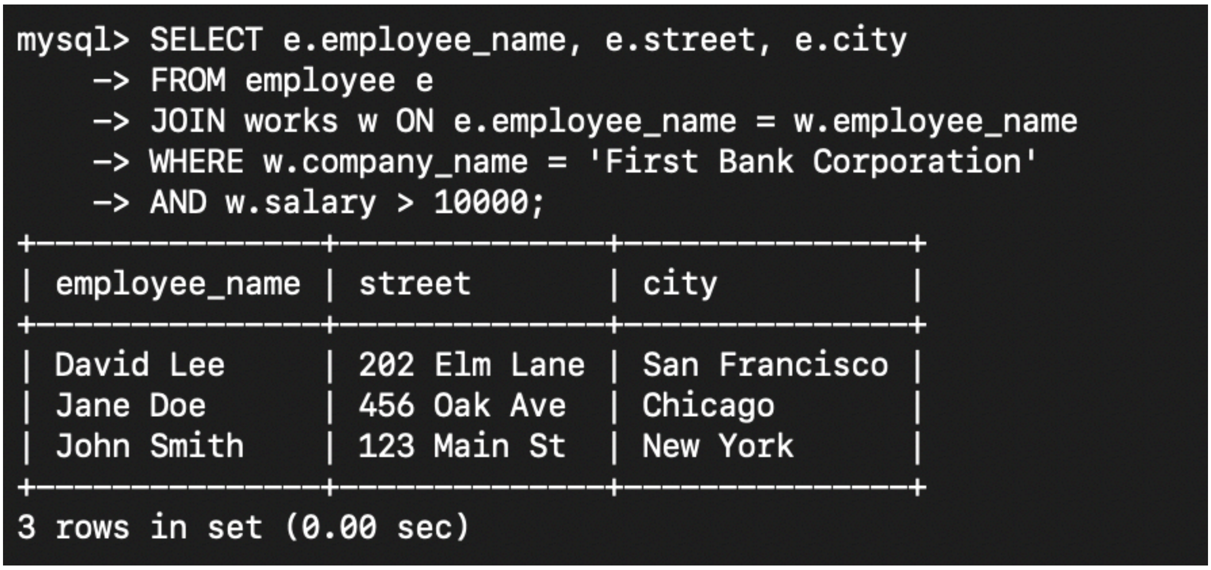
```
[mysql> create table employee(
[    -> employee_name varchar(50),
[    -> street varchar(50),
[    -> city varchar(50),
[    -> primary key (employee_name));
 Query OK, 0 rows affected (0.02 sec)

[mysql> create table company(
[    -> company_name varchar(50),
[    -> city varchar(50),
[    -> primary key (company_name));
 Query OK, 0 rows affected (0.00 sec)

[mysql> create table works(
[    -> employee_name varchar(50),
[    -> company_name varchar(50),
[    -> salary int,
[    -> primary key (employee_name),
[    -> foreign key (employee_name) references employee(employee_name),
[    -> foreign key (company_name) references company(company_name));
 Query OK, 0 rows affected (0.01 sec)

[mysql> create table manages(
     -> employee_name VARCHAR(100),
     ->     manager_name VARCHAR(100),
     ->     PRIMARY KEY (employee_name),
     ->     FOREIGN KEY (employee_name) REFERENCES employee(employee_name),
[    ->     FOREIGN KEY (manager_name) REFERENCES employe(employee_name));
```

a. Find the names and cities of residence of all employees who work for First Bank Corporation.



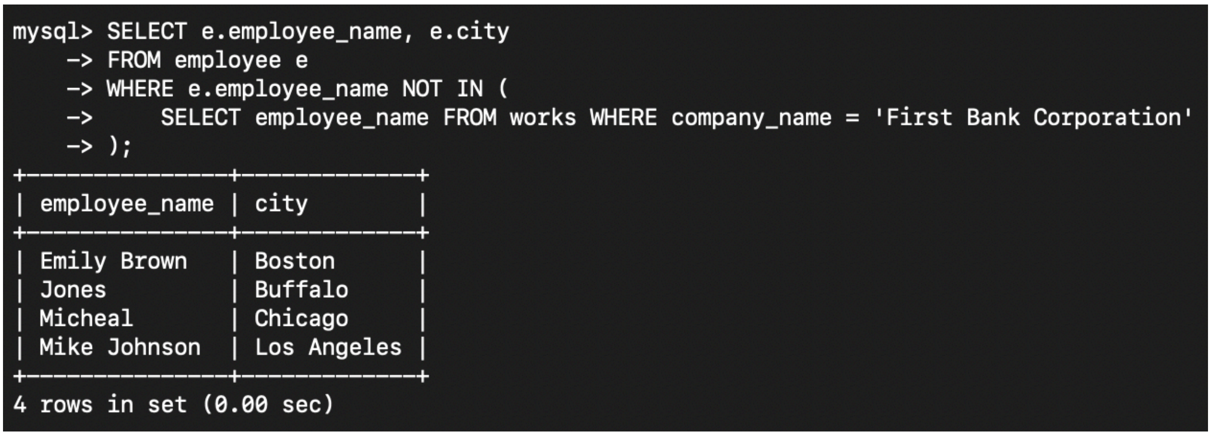
```
mysql> SELECT e.employee_name, e.city
    -> FROM employee e
    -> JOIN works w ON e.employee_name = w.employee_name
    -> WHERE w.company_name = 'First Bank Corporation';
+---------------+---------------+
| employee_name | city          |
+---------------+---------------+
| David Lee     | San Francisco |
| Dhruba Sinha  | San Francisco |
| Jane Doe      | Chicago       |
| John Smith    | New York      |
+---------------+---------------+
4 rows in set (0.00 sec)
```

b. Find the names, street addresses, and cities of residence of all employees who work for First Bank Corporation and earn more than 10, 000.



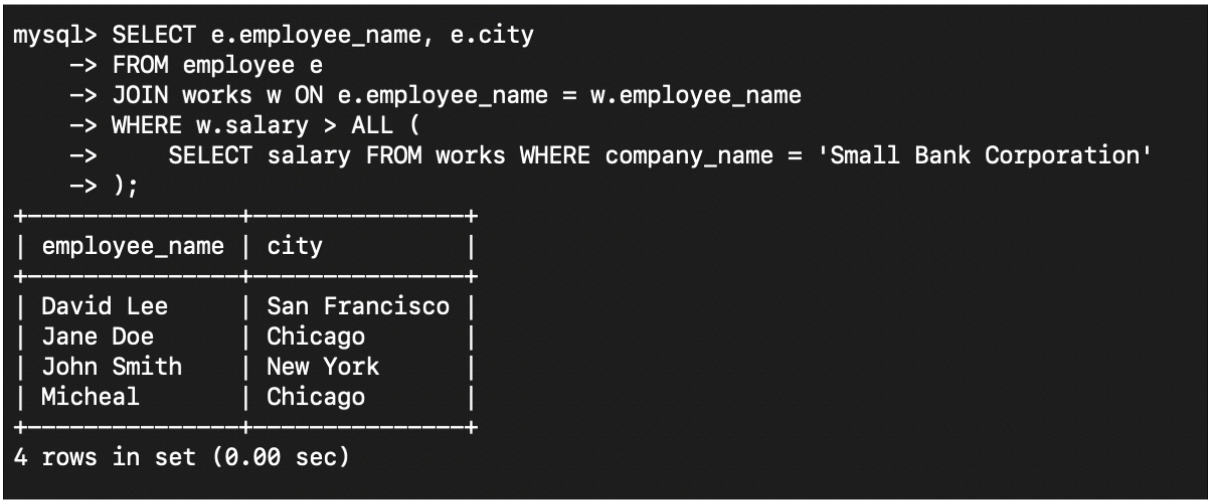
```
mysql> SELECT e.employee_name, e.street, e.city
    -> FROM employee e
    -> JOIN works w ON e.employee_name = w.employee_name
    -> WHERE w.company_name = 'First Bank Corporation'
    -> AND w.salary > 10000;
+---------------+--------------+---------------+
| employee_name | street       | city          |
+---------------+--------------+---------------+
| David Lee     | 202 Elm Lane | San Francisco |
| Jane Doe      | 456 Oak Ave  | Chicago       |
| John Smith    | 123 Main St  | New York      |
+---------------+--------------+---------------+
3 rows in set (0.00 sec)
```

c. Find all employees in the database who do not work for First Bank Corporation.



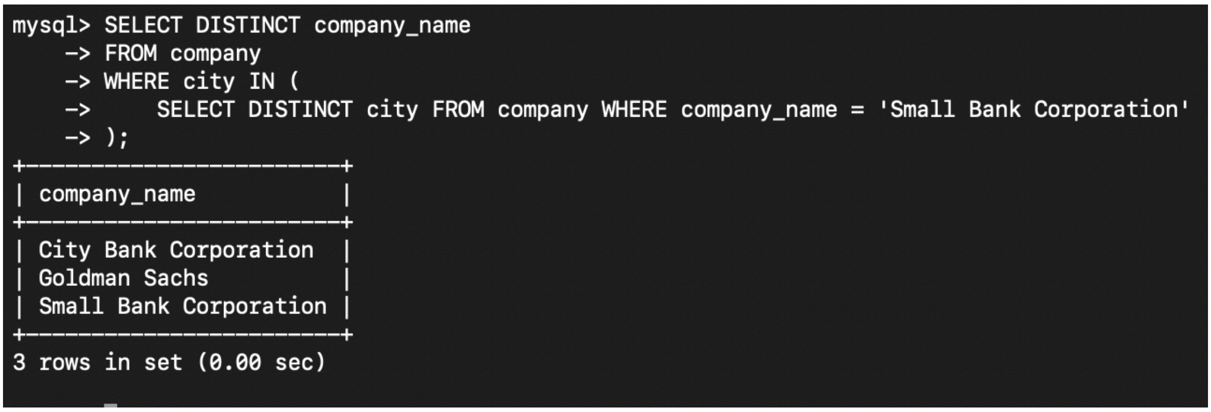
```
mysql> SELECT e.employee_name, e.city
    -> FROM employee e
    -> WHERE e.employee_name NOT IN (
    ->     SELECT employee_name FROM works WHERE company_name = 'First Bank Corporation'
    -> );
+---------------+-------------+
| employee_name | city        |
+---------------+-------------+
| Emily Brown   | Boston      |
| Jones         | Buffalo     |
| Micheal       | Chicago     |
| Mike Johnson  | Los Angeles |
+---------------+-------------+
4 rows in set (0.00 sec)
```

d. Find all employees in the database who earn more than each employee of Small Bank Corporation.



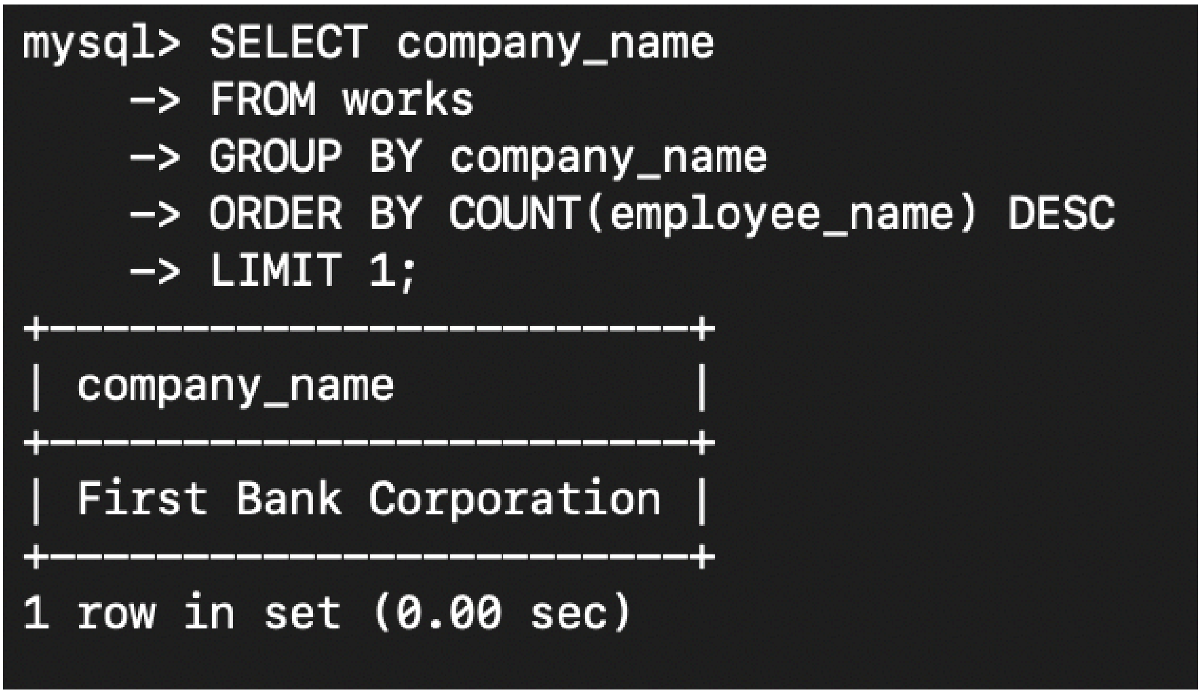
```
mysql> SELECT e.employee_name, e.city
    -> FROM employee e
    -> JOIN works w ON e.employee_name = w.employee_name
    -> WHERE w.salary > ALL (
    ->     SELECT salary FROM works WHERE company_name = 'Small Bank Corporation'
    -> );
+---------------+---------------+
| employee_name | city          |
+---------------+---------------+
| David Lee     | San Francisco |
| Jane Doe      | Chicago       |
| John Smith    | New York      |
| Micheal       | Chicago       |
+---------------+---------------+
4 rows in set (0.00 sec)
```

e. Find all companies located in every city in which Small Bank Corporation is located.



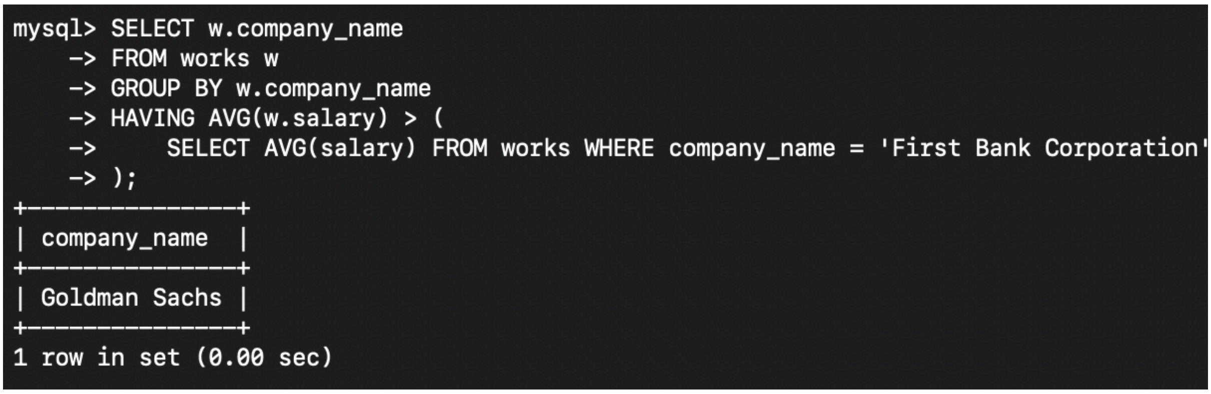
```
mysql> SELECT DISTINCT company_name
    -> FROM company
    -> WHERE city IN (
    ->     SELECT DISTINCT city FROM company WHERE company_name = 'Small Bank Corporation'
    -> );
+------------------------+
| company_name           |
+------------------------+
| City Bank Corporation  |
| Goldman Sachs          |
| Small Bank Corporation |
+------------------------+
3 rows in set (0.00 sec)
```

f. Find the company that has the most employees.



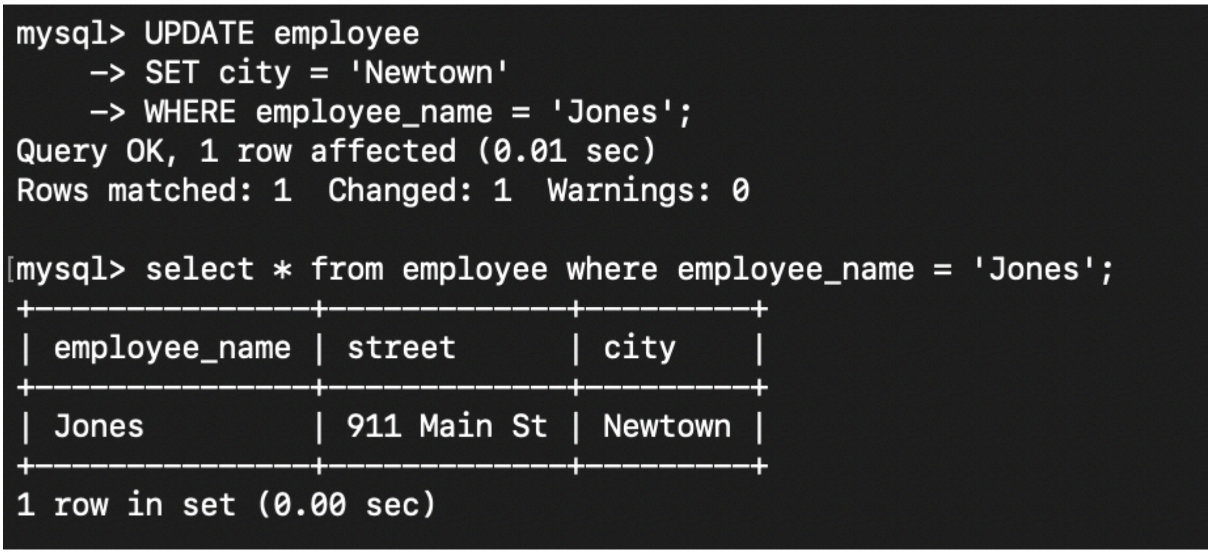
```
mysql> SELECT company_name
    -> FROM works
    -> GROUP BY company_name
    -> ORDER BY COUNT(employee_name) DESC
    -> LIMIT 1;
+------------------------+
| company_name           |
+------------------------+
| First Bank Corporation |
+------------------------+
1 row in set (0.00 sec)
```

g. Find those companies whose employees earn a higher salary, on average, than the average salary at First Bank Corporation.



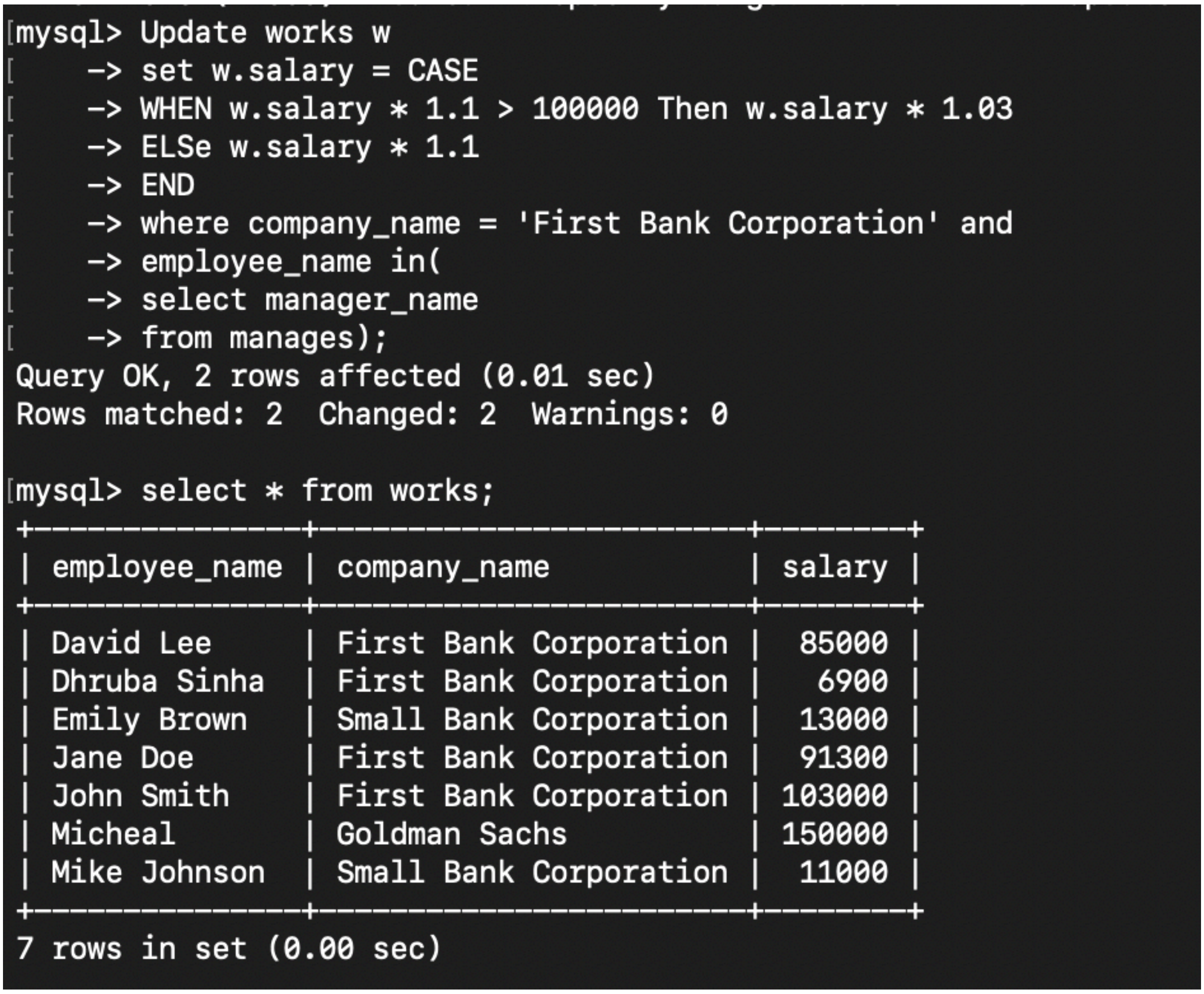
```
mysql> SELECT w.company_name
    -> FROM works w
    -> GROUP BY w.company_name
    -> HAVING AVG(w.salary) > (
    ->     SELECT AVG(salary) FROM works WHERE company_name = 'First Bank Corporation'
    -> );
+---------------+
| company_name  |
+---------------+
| Goldman Sachs |
+---------------+
1 row in set (0.00 sec)
```

h. Modify the database so that Jones now lives in Newtown.



```
mysql> UPDATE employee
    -> SET city = 'Newtown'
    -> WHERE employee_name = 'Jones';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

[mysql> select * from employee where employee_name = 'Jones';
+---------------+-------------+---------+
| employee_name | street      | city    |
+---------------+-------------+---------+
| Jones         | 911 Main St | Newtown |
+---------------+-------------+---------+
1 row in set (0.00 sec)
```

i. Give all managers of First Bank Corporation a 10 percent raise unless the salary becomes greater than 100,000; in such cases, give only a 3 percent raise.



```
[mysql> Update works w
[    -> set w.salary = CASE
[    -> WHEN w.salary * 1.1 > 100000 Then w.salary * 1.03
[    -> ELSe w.salary * 1.1
[    -> END
[    -> where company_name = 'First Bank Corporation' and
[    -> employee_name in(
[    -> select manager_name
[    -> from manages);
Query OK, 2 rows affected (0.01 sec)
Rows matched: 2  Changed: 2  Warnings: 0

[mysql> select * from works;
+---------------+------------------------+--------+
| employee_name | company_name           | salary |
+---------------+------------------------+--------+
| David Lee     | First Bank Corporation |  85000 |
| Dhruba Sinha  | First Bank Corporation |   6900 |
| Emily Brown   | Small Bank Corporation |  13000 |
| Jane Doe      | First Bank Corporation |  91300 |
| John Smith    | First Bank Corporation | 103000 |
| Micheal       | Goldman Sachs          | 150000 |
| Mike Johnson  | Small Bank Corporation |  11000 |
+---------------+------------------------+--------+
7 rows in set (0.00 sec)
```