Name: Gurdev singh

Id:100376466

LAB:7

Q-1: CREATE ALL three tables:

Ans:

CITY table:

```
CREATE TABLE city
(
        id INT PRIMARY KEY,
        name VARCHAR (255) NOT NULL
);
```

Data output | Messages | Notifications

	id [PK] integer	name character varying (255)
1	1	Halifax
2	2	Calgary
3	3	Boston
4	4	New York
5	5	Toronto

Occupation table:

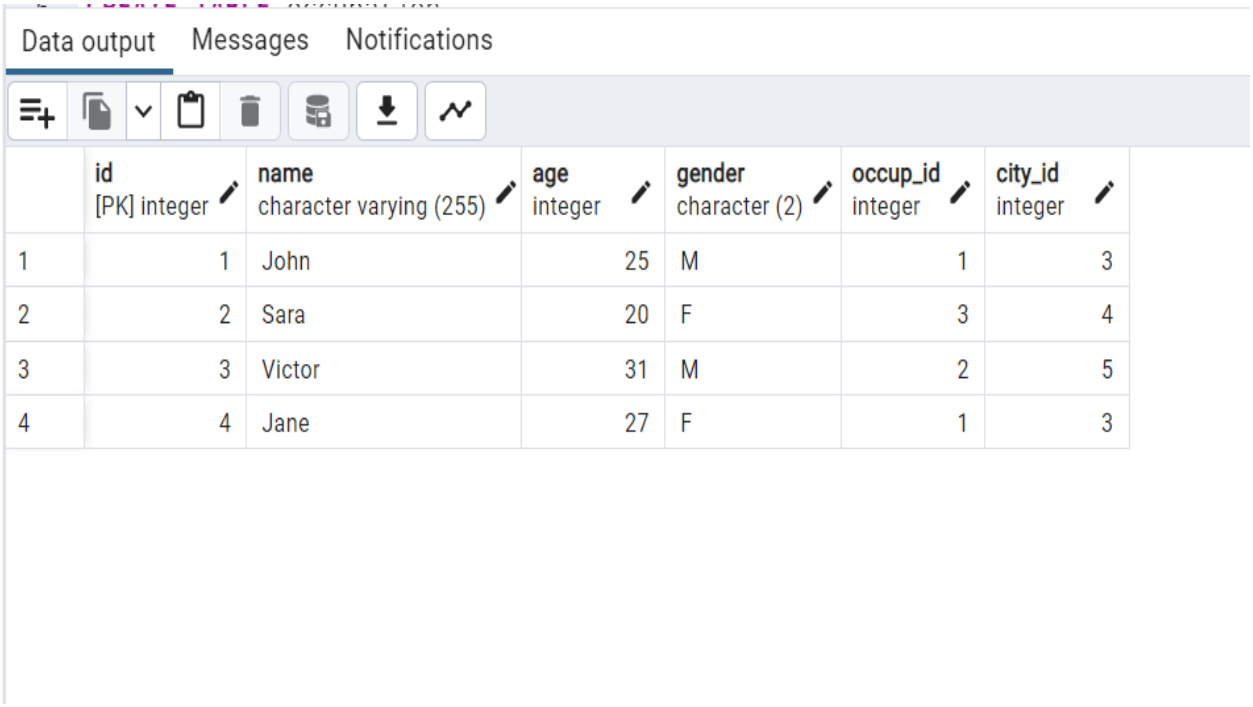
```
CREATE TABLE occupation
(
        id INT PRIMARY KEY,
        name VARCHAR(255) NOT NULL
);
```

Data output | Messages | Notifications

	id [PK] integer	name character varying (255)
1	1	Software Engineer
2	2	Accountant
3	3	Pharmacist
4	4	Library Assistant

Users table:

```
CREATE TABLE users
(
        id INT PRIMARY KEY,
        name VARCHAR(255) NOT NULL,
         age INT NOT NULL,
        gender char(2) NOT NULL CHECK(gender='M' OR gender='F'),
        occup_id   NOT NULL  int  REFERENCES occupation(id),
        city_id int   NOT NULL REFERENCES city(id)
);
```



Data output | Messages | Notifications

	id [PK] integer	name character varying (255)	age integer	gender character (2)	occup_id integer	city_id integer
1	1	John	25	M	1	3
2	2	Sara	20	F	3	4
3	3	Victor	31	M	2	5
4	4	Jane	27	F	1	3

b) Write 2 different types of SQL queries to find the users in city 'Boston'. Write 2 types of queries, one using joins to find

the answer and another using subqueries to find the same answer.

Ans:

Full-join:

```
SELECT*
FROM city FULL JOIN users
ON city.id = users.city_id
WHERE city.name='Boston';
```

Data output | Messages | Notifications

	id integer	name character varying (255)	id integer	name character varying (255)	age integer	gender character (2)	occup_id integer	city_id integer
1	3	Boston	1	John	25	M	1	3
2	3	Boston	4	Jane	27	F	1	3

Subquery:

```
SELECT*
FROM users
WHERE users.city_id=
(
        SELECT id
        from city
        WHERE city.name='Boston'
)
```

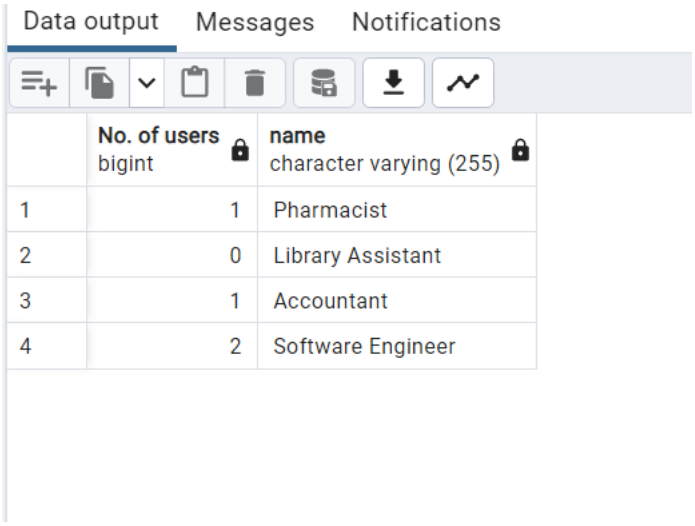
Data output | Messages | Notifications

	id [PK] integer	name character varying (255)	age integer	gender character (2)	occup_id integer	city_id integer
1	1	John	25	M	1	3
2	4	Jane	27	F	1	3

Q- Write SQL query to find how many users are there per occupation.

Ans:

```
SELECT count(users.id) AS "No. of users",occupation.name
FROM users FULL JOIN occupation
ON users.occup_id = occupation.id
GROUP BY(occupation.id,occupation.name);
```



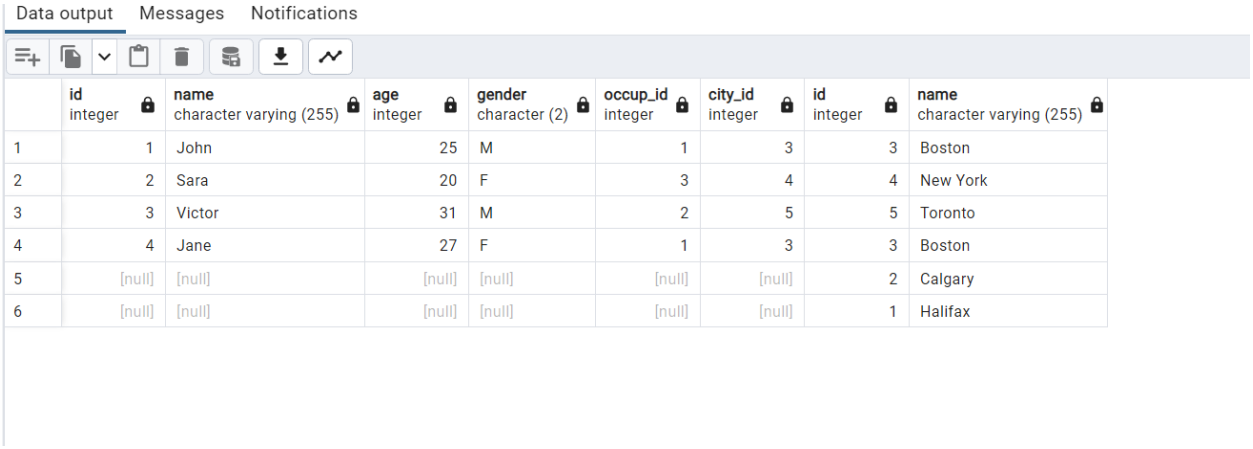
Data output | Messages | Notifications

	No. of users bigint	name character varying (255)
1	1	Pharmacist
2	0	Library Assistant
3	1	Accountant
4	2	Software Engineer

Q-Perform full outer join between users and city.

Ans:

```
SELECT *
FROM users FULL JOIN city
ON city.id = users.city_id
```



Data output | Messages | Notifications

	id integer	name character varying (255)	age integer	gender character (2)	occup_id integer	city_id integer	id integer	name character varying (255)
1	1	John	25	M	1	3	3	Boston
2	2	Sara	20	F	3	4	4	New York
3	3	Victor	31	M	2	5	5	Toronto
4	4	Jane	27	F	1	3	3	Boston
5	[null]	[null]	[null]	[null]	[null]	[null]	2	Calgary
6	[null]	[null]	[null]	[null]	[null]	[null]	1	Halifax

Q-Write query to make a copy of ‘users’ table known as ‘users_new’ without Data.

Ans:

```
CREATE TABLE users_new

AS

TABLE users

WITH NO DATA;
```

Q- Write query to insert all columns of ‘users’ to the ‘users_new’.

Ans:

```
INSERT INTO users_new (id, name, age, gender, occup_id, city_id)

SELECT *

FROM users;
```

```
SELECT*
FROM users_new

INSERT INTO users_new (id, name, age, gender,occup_id,city_id)
SELECT *
FROM users
```

Data output | Messages | Notifications

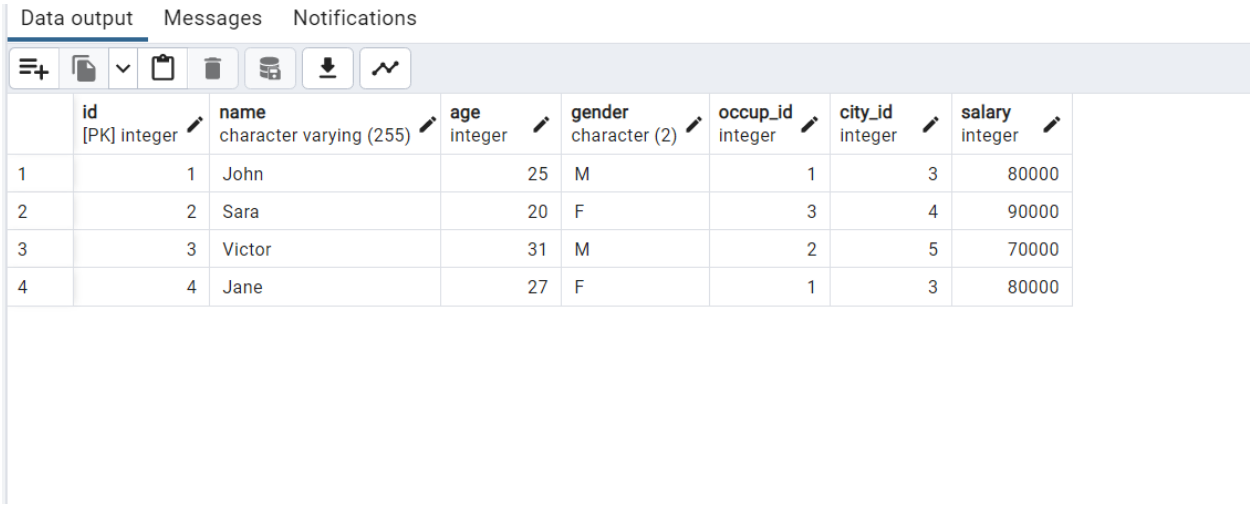
	id integer	name character varying (255)	age integer	gender character (2)	occup_id integer	city_id integer
1	1	John	25	M	1	3
2	2	Sara	20	F	3	4
3	3	Victor	31	M	2	5
4	4	Jane	27	F	1	3

Q-Write CASE query to add one more column with salary values to the ‘users’ table. Salary for Software engineer is 80,000, Accountant is 70,000 and Pharmacist is 90,000.

Ans:

```
ALTER TABLE users

ADD COLUMN salary INT ;


UPDATE users

SET salary=

CASE

   WHEN users.id IN

      (

            SELECT users.id

            FROM occupation JOIN users
```

```
        ON users.occup_id = occupation.id
        WHERE occupation.name='Software Engineer'
) THEN 80000
 WHEN users.id=
(
        SELECT users.id
        FROM occupation JOIN users
        ON users.occup_id = occupation.id
        WHERE occupation.name='Accountant'
) THEN 70000
WHEN users.id=
(
        SELECT users.id
        FROM occupation JOIN users
        ON users.occup_id = occupation.id
        WHERE occupation.name='Pharmacist'
) THEN 90000
END;
```



Data output | Messages | Notifications

	id [PK] integer	name character varying (255)	age integer	gender character (2)	occup_id integer	city_id integer	salary integer
1	1	John	25	M	1	3	80000
2	2	Sara	20	F	3	4	90000
3	3	Victor	31	M	2	5	70000
4	4	Jane	27	F	1	3	80000

d-Write query to add foreign keys constraints to ‘users’ table. Assuming you forgot to add it earlier.

Ans:

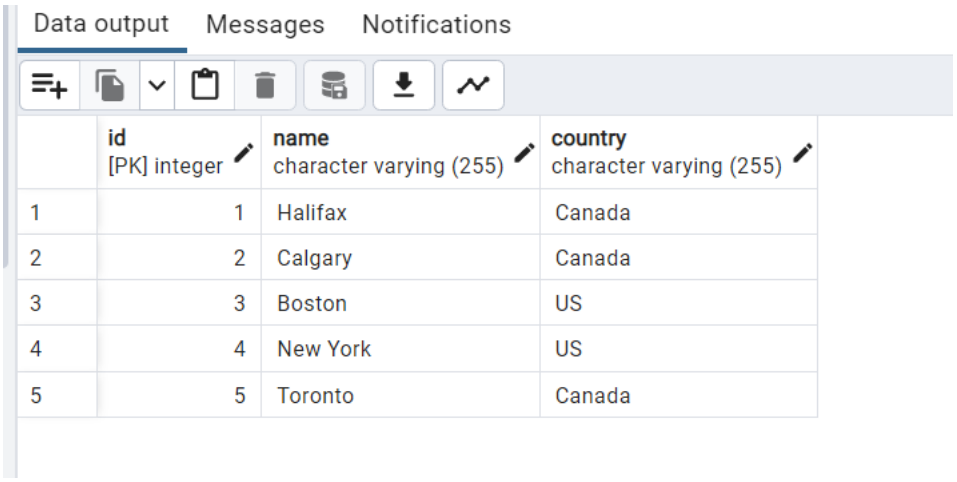
```
ALTER TABLE users

ADD CONSTRAINT occup_id

FOREIGN KEY (occup_id) REFERENCES occupation(id);

--Add second

ALTER TABLE users

ADD CONSTRAINT occup_id

FOREIGN KEY (occup_id) REFERENCES occupation(id);
```

Q-Add country column to ‘city’ table. DEFAULT constraint must be used to add Canada as a default country for cities. [Use DEFAULT Constraint to default your country to Canada, that way you only have to write the countries for cities not in Canada, Hint : remember ‘boston’ and ‘new York’ are cities in

US, rest all are in Canada, Use ALTER TABLE to add column and default constraint].

Ans:

```
ALTER TABLE city

ADD column country VARCHAR (255) DEFAULT 'Canada'


UPDATE city

SET country=

CASE

   WHEN name='Boston' THEN 'US'

      WHEN name='New York' THEN 'US'

      else 'Canada'

      END;
```



Data output | Messages | Notifications

	id [PK] integer	name character varying (255)	country character varying (255)
1	1	Halifax	Canada
2	2	Calgary	Canada
3	3	Boston	US
4	4	New York	US
5	5	Toronto	Canada