lab1

- Create your tables with their columns in PostgreSQL.

```
postgres=#
postgres=# CREATE TABLE track (
postgres(#     id SERIAL PRIMARY KEY,
postgres(#     track_name VARCHAR(100)
postgres(# );
CREATE TABLE
postgres=# CREATE TABLE subject (
postgres(#     id SERIAL PRIMARY KEY,
postgres(#     sub_name VARCHAR(100),
postgres(#     max_score INT
postgres(# );
CREATE TABLE
postgres=# CREATE TABLE student (
postgres(#     id SERIAL PRIMARY KEY,
postgres(#     e_name VARCHAR(100),
postgres(#     email VARCHAR(100),
postgres(#     address TEXT,
postgres(#     birth_date DATE,
postgres(#     gender VARCHAR(6) CHECK (gender IN ('Male', 'Female')),
postgres(#     track_id INT REFERENCES track(id)
postgres(# );CREATE TABLE exam (
CREATE TABLE
postgres(#     id SERIAL PRIMARY KEY,
postgres(#     date DATE
postgres(# );
CREATE TABLE
postgres=#
postgres=# CREATE TABLE grades (
postgres(#     stu_id INT REFERENCES student(id),
postgres(#     sub_id INT REFERENCES subject(id),
postgres(#     exam_id INT REFERENCES exam(id),
postgres(#     grade INT,
postgres(#     PRIMARY KEY (stu_id, sub_id)
postgres(# );
CREATE TABLE
postgres=#
postgres=# CREATE TABLE stu_sub (
postgres(#     stu_id INT REFERENCES student(id),
postgres(#     sub_id INT REFERENCES subject(id),
postgres(#     PRIMARY KEY (stu_id, sub_id)
postgres(# );
CREATE TABLE
postgres=#
postgres=# CREATE TABLE track_sub (
postgres(#     track_id INT REFERENCES track(id),
postgres(#     sub_id INT REFERENCES subject(id),
postgres(#     PRIMARY KEY (track_id, sub_id)
postgres(# );
```

==========

- Insert at minimum 3 Rows at each table.

```
postgres@mohamed-Inspiron-5570:~$ psql
psql (12.22 (Ubuntu 12.22-0ubuntu0.20.04.2))
Type "help" for help.

postgres=# INSERT INTO track (track_name) VALUES ('Computer Science'), ('MERN'), ('Full stack');
INSERT 0 3
postgres=#
postgres=# INSERT INTO subject (sub_name, max_score) VALUES
postgres-# ('Database', 100), ('Python', 90), ('Java Script', 95);
INSERT 0 3
postgres=#
postgres=# INSERT INTO student (e_name, email, address, birth_date, gender, track_id) VALUES
postgres-# ('Mohamed', 'mohamed@gmail.com', 'Assiut', '2001-04-11', 'Male', 1),
postgres-# ('Ahmed', 'ahmed@gmail.com', 'Assiut', '1998-01-27', 'Male', 2),
postgres-# ('Omar', 'omar@gmail.com', 'Minia', '2007-07-7', 'Male', 3);
INSERT 0 3
postgres=#
postgres=# INSERT INTO exam (date) VALUES ('2025-04-29'), ('2025-06-15');
INSERT 0 2
postgres=#
postgres=# INSERT INTO grades (stu_id, sub_id, exam_id, grade) VALUES
postgres-# (1, 1, 1, 85), (2, 2, 1, 88), (3, 3, 2, 92);
INSERT 0 3
postgres=#
postgres=# INSERT INTO stu_sub (stu_id, sub_id) VALUES
postgres-# (1, 1), (2, 2), (3, 3);
INSERT 0 3
postgres=#
postgres=# INSERT INTO track_sub (track_id, sub_id) VALUES
postgres-# (1, 1), (2, 2), (3, 3);
```

==========

- Add birth date column for the student table.
- Add gender column which hold only 2 values (Male or Female).

Already Exists

- Display male students who are born before 1991-10-01.

```
postgres=# SELECT * FROM student
postgres-# WHERE gender = 'Male' AND birth_date < '1991-10-01';
 id | e_name | email | address | birth_date | gender | track_id
----+--------+-------+---------+------------+--------+----------
(0 rows)

postgres=# SELECT * FROM student
WHERE gender = 'Male' AND birth_date < '2000-10-01';
 id | e_name |      email      | address | birth_date | gender | track_id
----+--------+-----------------+---------+------------+--------+----------
  2 | Ahmed  | ahmed@gmail.com | Assiut  | 1998-01-27 | Male   |        2
(1 row)

postgres=#
```

==========

- Display students' names that begin with A.

```
postgres=#
postgres=#
postgres=# SELECT * FROM student
postgres-# WHERE e_name LIKE 'A%';
 id | e_name |      email      | address | birth_date | gender | track_id
----+--------+-----------------+---------+------------+--------+----------
  2 | Ahmed  | ahmed@gmail.com | Assiut  | 1998-01-27 | Male   |        2
(1 row)

postgres=#
```

- Display subjects and their max score sorted by max score.

```
postgres=# SELECT sub_name, max_score FROM subject ORDER BY max_score DESC;
  sub_name   | max_score
-------------+-----------
 java script |       100
 Database    |       100
 python      |       100
 Java Script |        95
 database    |        95
 database    |        95
 Python      |        90
 Python      |        90
 java script |        80
(9 rows)

postgres=#
```

- Display the subject with highest max score

```
postgres=# SELECT sub_name, max_score FROM subject ORDER BY max_score DESC LIMIT 1;
 sub_name | max_score
----------+-----------
 python   |       100
(1 row)

postgres=#
```

end of lab1

Lab2

1. Display the number of students their name is “Mohammed”

```
postgres=# SELECT COUNT(*) FROM student WHERE e_name = 'Mohammed';
 count
-------
     0
(1 row)

postgres=# SELECT COUNT(*) FROM student WHERE e_name = 'Mohamed';
 count
-------
     1
(1 row)

postgres=#
```

2. Display the number of males and females.

```
postgres=# SELECT gender, COUNT(*) FROM student GROUP BY gender;
 gender | count
--------+-------
 Male   |     3
(1 row)

postgres=#
```

==========

3. Display the repeated first names and their counts if higher
than 2.

```
postgres=# SELECT e_name AS first_name, COUNT(*) AS count
postgres-# FROM student
postgres-# GROUP BY e_name
postgres-# HAVING COUNT(*) > 2;
 first_name | count
------------+-------
(0 rows)

postgres=# INSERT INTO student (e_name, email, address, birth_date, gender, track_id) VALUES
('Mohamed', 'mohamed@gmail.com', 'Assiut', '2001-04-11', 'Male', 1),
('Ahmed', 'ahmed@gmail.com', 'Assiut', '1998-01-27', 'Male', 2),
('Omar', 'omar@gmail.com', 'Minia', '2007-07-7', 'Male', 3);
INSERT 0 3
postgres=# SELECT e_name AS first_name, COUNT(*) AS count
FROM student
GROUP BY e_name
HAVING COUNT(*) > 2;
 first_name | count
------------+-------
(0 rows)

postgres=# INSERT INTO student (e_name, email, address, birth_date, gender, track_id) VALUES
('Mohamed', 'mohamed@gmail.com', 'Assiut', '2001-04-11', 'Male', 1),
('Ahmed', 'ahmed@gmail.com', 'Assiut', '1998-01-27', 'Male', 2),
('Omar', 'omar@gmail.com', 'Minia', '2007-07-7', 'Male', 3);
INSERT 0 3
postgres=# SELECT e_name AS first_name, COUNT(*) AS count
FROM student
GROUP BY e_name
HAVING COUNT(*) > 2;
 first_name | count
------------+-------
 Ahmed      |     3
 Omar       |     3
 Mohamed    |     3
(3 rows)
```

4. Display all students and track name they belong to.

```
postgres=# SELECT student.e_name, track.track_name
FROM student
JOIN track ON student.track_id = track.id;
 e_name  | track_name
---------+-------------
 Mohamed | fullstack
 Mohamed | fullstack
 Mohamed | fullstack
 Ahmed   | mern
 Ahmed   | mern
 Ahmed   | mern
 Omar    | open source
 Omar    | open source
 Omar    | open source
(9 rows)
```

==========

5. Display all students except those who are in OS track.

```
postgres=# SELECT s.*
FROM student s
JOIN track t ON s.track_id = t.id
WHERE t.track_name <> 'open source';
 id | e_name  |       email       | address | birth_date | gender | track_id
----+---------+-------------------+---------+------------+--------+----------
  7 | Mohamed | mohamed@gmail.com | Assiut  | 2001-04-11 | Male   |        1
  4 | Mohamed | mohamed@gmail.com | Assiut  | 2001-04-11 | Male   |        1
  1 | Mohamed | mohamed@gmail.com | Assiut  | 2001-04-11 | Male   |        1
  8 | Ahmed   | ahmed@gmail.com   | Assiut  | 1998-01-27 | Male   |        2
  5 | Ahmed   | ahmed@gmail.com   | Assiut  | 1998-01-27 | Male   |        2
  2 | Ahmed   | ahmed@gmail.com   | Assiut  | 1998-01-27 | Male   |        2
(6 rows)

postgres=#
```