

## Lab 1

### 1. Create your tables with their columns in PostgreSQL.

```
hajar@hajar-G3-3500: ~  
iti=# creat table student (  
id serial primary key ,e_name text ,email varchar(50),address text,track_id int ,CONSTRAINT  
fk_track FOREIGN KEY(track_id) REFERENCES track(id) );  
ERROR: syntax error at or near "creat"  
LINE 1: creat table student (  
      ^  
iti=# create table student (  
id serial primary key ,e_name text ,email varchar(50),address text,track_id int ,CONSTRAINT  
fk_track FOREIGN KEY(track_id) REFERENCES track(id) );  
CREATE TABLE  
iti=# create table grades (student_id int,sub_id int,exam_id int,grade numeric,CONSTRAINT f  
k_student FOREIGN KEY(student_id) REFERENCES student(id),CONSTRAINT fk_subject FOREIGN KEY(  
sub_id) REFERENCES subject(id),CONSTRAINT fk_exam FOREIGN KEY(exam_id) REFERENCES exam(id)  
);  
ERROR: relation "subject" does not exist  
iti=# create table subject (id serial primary key ,sub_name text,max_score numeric );  
CREATE TABLE  
iti=# create table grades (student_id int,sub_id int,exam_id int,grade numeric,CONSTRAINT f  
k_student FOREIGN KEY(student_id) REFERENCES student(id),CONSTRAINT fk_subject FOREIGN KEY(  
sub_id) REFERENCES subject(id),CONSTRAINT fk_exam FOREIGN KEY(exam_id) REFERENCES exam(id)  
);  
CREATE TABLE  
iti=# create table stu_sub (stu_id int,sub_id int,CONSTRAINT fk_student FOREIGN KEY(stu_id)  
REFERENCES student(id),CONSTRAINT fk_subject FOREIGN KEY(sub_id) REFERENCES subject(id) )  
;  
CREATE TABLE  
iti=# create table grades (sub_id int,exam_id int,CONSTRAINT fk_subject FOREIGN KEY(sub_id)  
REFERENCES subject(id),CONSTRAINT fk_exam FOREIGN KEY(exam_id) REFERENCES exam(id) );  
ERROR: relation "grades" already exists  
iti=# create table track_sub (sub_id int,exam_id int,CONSTRAINT fk_subject FOREIGN KEY(sub_  
id) REFERENCES subject(id),CONSTRAINT fk_exam FOREIGN KEY(exam_id) REFERENCES exam(id) );  
CREATE TABLE  
iti=# \d  
          List of relations  
Schema |      Name      | Type  | Owner  
-----+-----+-----+-----  
public | exam            | table | postgres  
public | exam_id_seq     | sequence | postgres  
public | grades          | table | postgres  
public | stu_sub         | table | postgres  
public | student         | table | postgres  
public | student_id_seq  | sequence | postgres  
public | subject         | table | postgres  
public | subject_id_seq  | sequence | postgres  
public | track          | table | postgres  
public | track_id_seq    | sequence | postgres  
public | track_sub       | table | postgres  
(11 rows)
```

## 2. Insert at minimum 3 Rows at each table.

```
hajar@hajar-G3-3500: ~  
iti=# insert into track values(1,'python');  
INSERT 0 1  
iti=# insert into track values(2,'java');  
INSERT 0 1  
iti=# insert into track values(3,'c++');  
INSERT 0 1  
iti=# insert into student values(1,'Hajar','hajar@gmail.com','Assuit',3);  
INSERT 0 1  
iti=# insert into student values(2,'Amira','amira@gmail.com','Assuit',1);  
INSERT 0 1  
iti=# insert into student values(2,'Sara','sara@gmail.com','Alex',1);  
ERROR: duplicate key value violates unique constraint "student_pkey"  
DETAIL: Key (id)=(2) already exists.  
iti=# insert into student values(3,'Sara','sara@gmail.com','Alex',1);  
INSERT 0 1  
iti=# insert into subject values(1,'oop',100);  
INSERT 0 1  
iti=# insert into subject values(2,'data structure',100);  
INSERT 0 1  
iti=# insert into subject values(3,'os',100);  
INSERT 0 1  
iti=# insert into exam values(1,'2025-01-15');  
INSERT 0 1  
iti=# insert into exam values(2,'2025-01-16');  
INSERT 0 1  
iti=# insert into exam values(3,'2025-01-17');  
INSERT 0 1  
iti=# insert into grades values(1,1,1,85);  
INSERT 0 1  
iti=# insert into grades values(1,2,2,95);  
INSERT 0 1  
iti=# insert into grades values(1,3,3,93);  
INSERT 0 1  
iti=# insert into stu-sub values(1,1);  
ERROR: syntax error at or near "-"  
LINE 1: insert into stu-sub values(1,1);  
                        ^  
iti=# insert into stu_sub values(1,1);  
INSERT 0 1  
iti=# insert into stu_sub values(1,2);  
INSERT 0 1  
iti=# insert into stu_sub values(1,3);  
INSERT 0 1  
iti=# insert into track_sub values(2,1);  
INSERT 0 1  
iti=# insert into track_sub values(1,2);  
INSERT 0 1  
iti=# insert into track_sub values(3,3);  
INSERT 0 1  
iti=# alter table student add birth_date date;  
ALTER TABLE
```

## 3. Add birth date column for the student table.

```
iti=# alter table student add birth_date date;  
ALTER TABLE  
iti=# select * from student  
iti=# ;  
 id | e_name | email | address | track_id | birth_date  
-----+-----+-----+-----+-----+-----  
  1 | Hajar  | hajar@gmail.com | Assuit |      3 |  
  2 | Amira  | amira@gmail.com | Assuit |      1 |  
  3 | Sara   | sara@gmail.com  | Alex   |      1 |  
(3 rows)  
iti=#
```

4. Add gender column which hold only 2 values (Male or Female).

```
iti=# create type gender_type as enum('male','female');
CREATE TYPE
iti=# alter table student add gender gender_type;
ALTER TABLE
iti=# select * from student
;
 id | e_name | email | address | track_id | birth_date | gender
-----+-----+-----+-----+-----+-----+-----
  1 | Hajar | hajar@gmail.com | Assuit | 3 |  | 
  2 | Amira | amira@gmail.com | Assuit | 1 |  | 
  3 | Sara | sara@gmail.com | Alex | 1 |  | 
(3 rows)

iti=#
```

5. Add/Alter foreign key constrains in your tables.

```
iti=# create table grades (student_id int,sub_id int,exam_id int,grade numeric,CONSTRAINT fk
_student FOREIGN KEY(student_id) REFERENCES student(id),CONSTRAINT fk_subject FOREIGN KEY(su
b_id) REFERENCES subject(id),CONSTRAINT fk_exam FOREIGN KEY(exam_id) REFERENCES exam(id) );
```

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

```
iti=# select *
from student
where gender = 'male' and birth_date < '1991-10-01';
 id | e_name | email | address | track_id | birth_date | gender
-----+-----+-----+-----+-----+-----+-----
(0 rows)

iti=#
```

7. Display students' names that begin with A.

```
iti=# select e_name
iti=# from student
iti=# where e_name like 'A%';
 e_name
-----
 Amira
(1 row)

iti=#
```

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

```
iti=# select sub_name,max_score
iti=# from subject
iti=# order by max_score;
 sub_name | max_score
-----+-----
 oop | 100
 data structure | 100
 os | 100
(3 rows)

iti=#
```

## 9. Display the subject with highest max score

```
iti=# select sub_name,max_score
iti=# from subject
iti=# order by max_score desc limit 1;
  sub_name | max_score
-----+-----
  oop      |        100
(1 row)

iti=#
```

## Lab 2

### 1. Display the number of students their name is "Mohammed"

```
iti=# select count(*)
iti=# from student
iti=# where e_name = 'mohammed';
  count
-----
       0
(1 row)

iti=#
```

### 2. Display the number of males and females.

```
iti=# select gender,count(*)
from student
group by gender;
  gender | count
-----+-----
       M |        3
(1 row)

iti=#
```

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

```
iti=# SELECT SPLIT_PART(e_name, ' ', 1) AS first_name, COUNT(*)
from student
group by first_name
having count(*) > 2;
 first_name | count
-----+-----
       Sara |        2
(1 row)
```

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

```
iti=# select student.e_name, track.track_name
from student join track on student.track_id = track.id;
  e_name | track_name
-----+-----
  Hajar  | c++
  Amira  | python
  Sara   | python
(3 rows)

iti=#
```

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

```
iti=# select student.*
iti-# from student join track on student.track_id = track.id
iti-# where track.track_name != 'os';
 id | e_name | email | address | track_id | birth_date | gender
-----+-----+-----+-----+-----+-----+-----
  1 | Hajar | hajar@gmail.com | Assuit | 3 |  | 
  2 | Amira | amira@gmail.com | Assuit | 1 |  | 
  3 | Sara | sara@gmail.com | Alex | 1 |  | 
(3 rows)

iti=#
```