PostgreSQL Lab3

Mazen Abdeltawab Saad

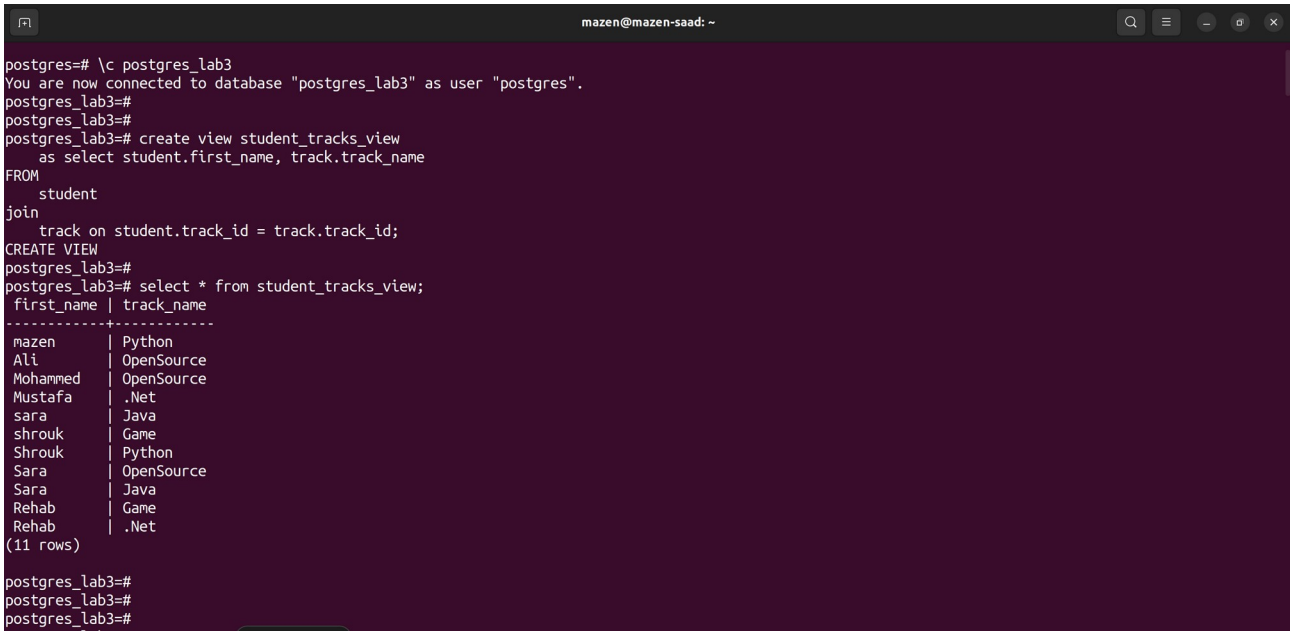
Track Python Fayoum

Lab3

su - postgres
mazen@@1
psql
create database postgres_lab3 TEMPLATE postgres_lab2;
\l
\c postgres_lab3

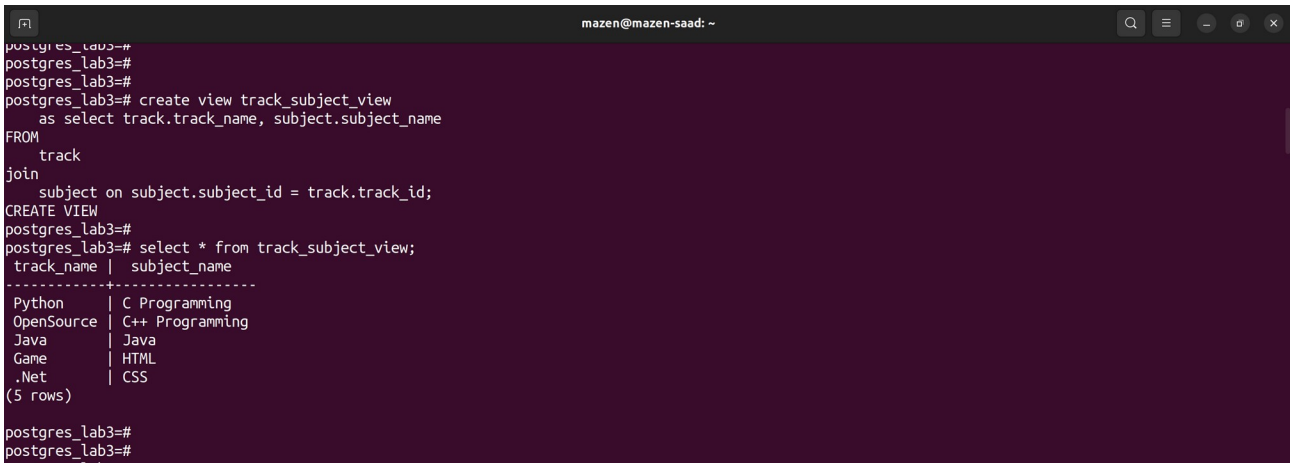
3. Create a view for student names with their Tracks names which is belong to it.

```
create view student_tracks_view
    as select student.first_name, track.track_name
FROM
    student
join
    track on student.track_id = track.track_id;

select * from student_tracks_view;
```



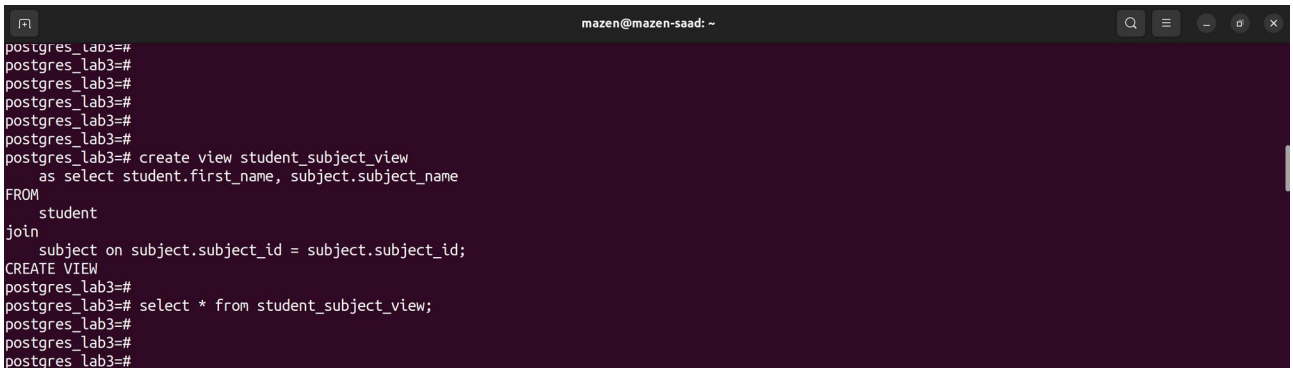
4. Create a view for Tracks names and the subjects which is belong/study to it.

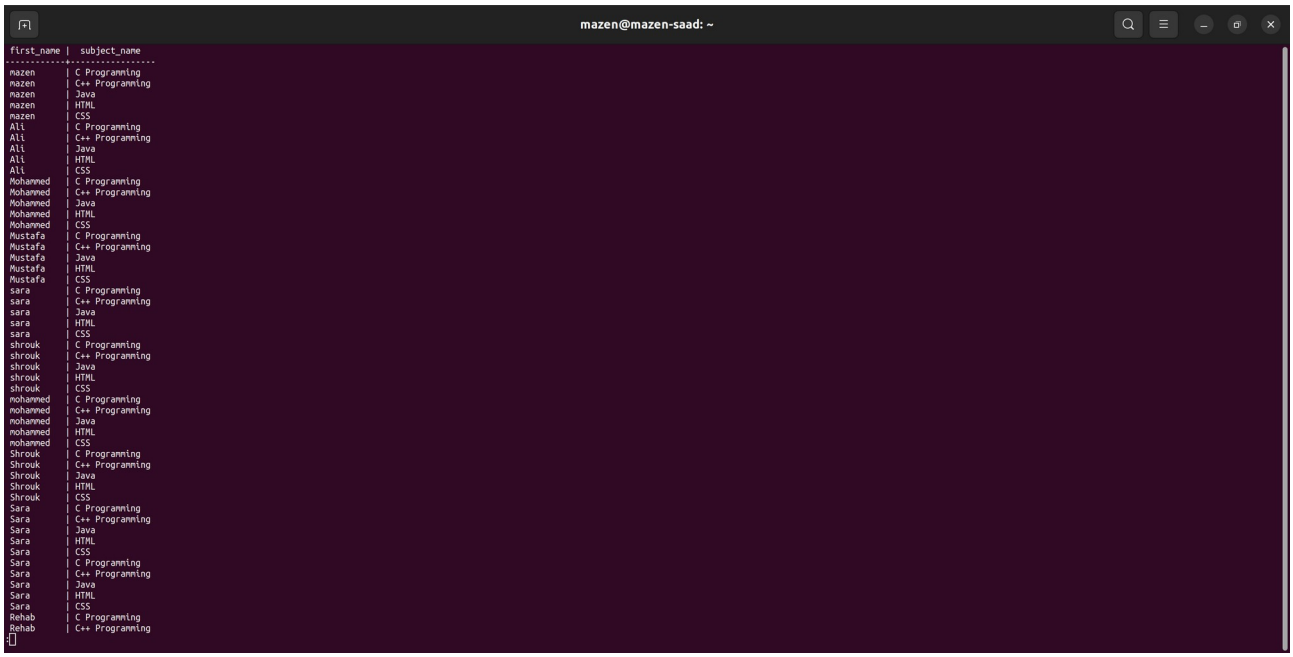
```
create view track_subject_view
    as select track.track_name, subject.subject_name
FROM
    track
join
    subject on subject.subject_id = track.track_id;

select * from track_subject_view;
```



5. Create a view for student names with their subject's names which will study.

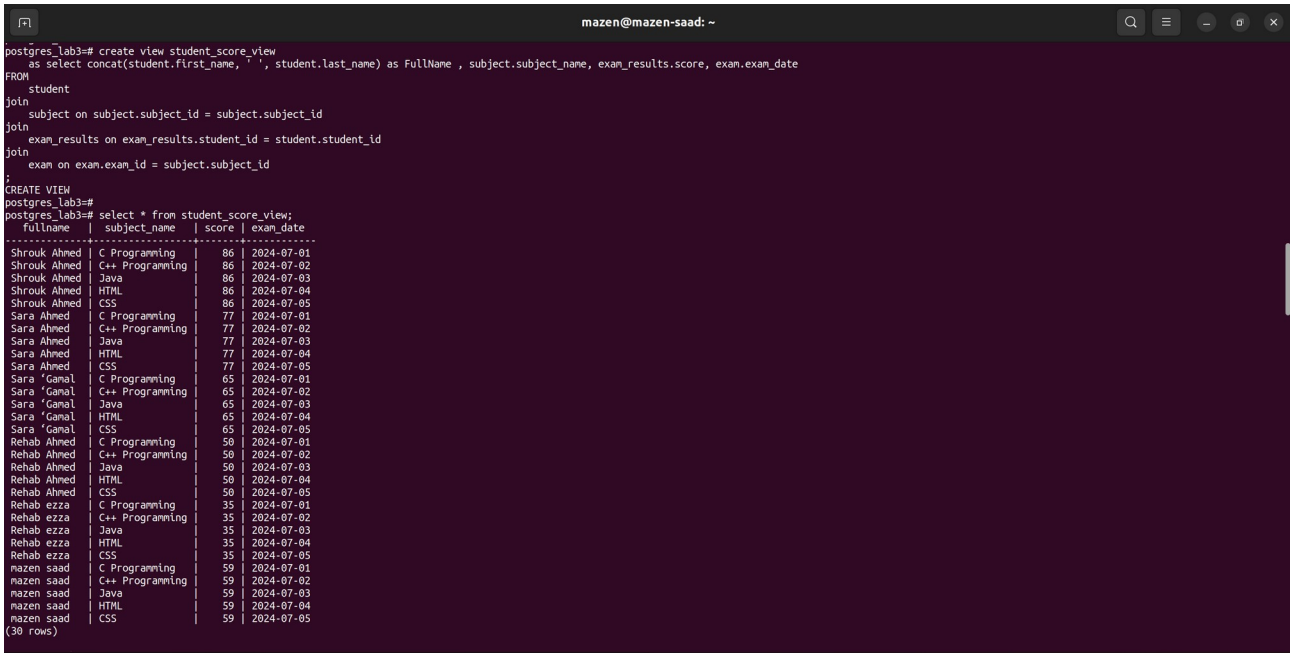
```
create view student_subject_view
    as select student.first_name, subject.subject_name
FROM
    student
join
    subject on subject.subject_id = subject.subject_id;

select * from student_subject_view;
```





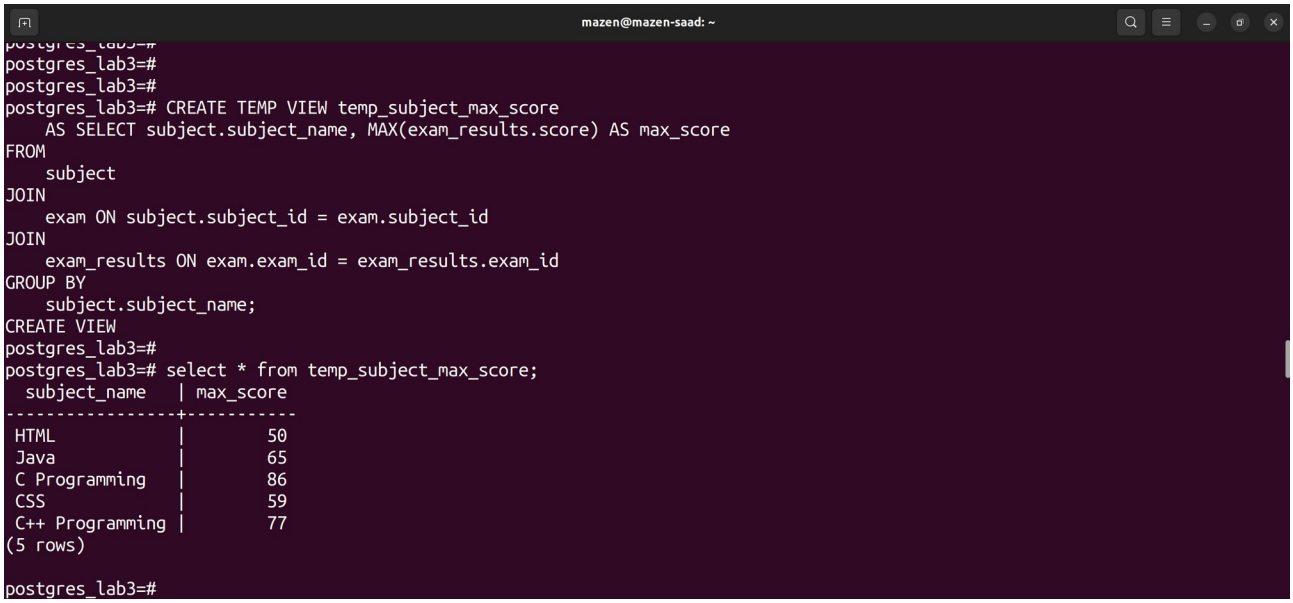
6. Create a view for all students name (Full Name) with their score in each subject and its date.

```
create view student_score_view
    as select concat(student.first_name, ' ', student.last_name) as
FullName , subject.subject_name, exam_results.score,
exam.exam_date
FROM
    student
join
    subject on subject.subject_id = subject.subject_id
join
    exam_results on exam_results.student_id = student.student_id
join
    exam on exam.exam_id = subject.subject_id
;

select * from student_score_view;
```



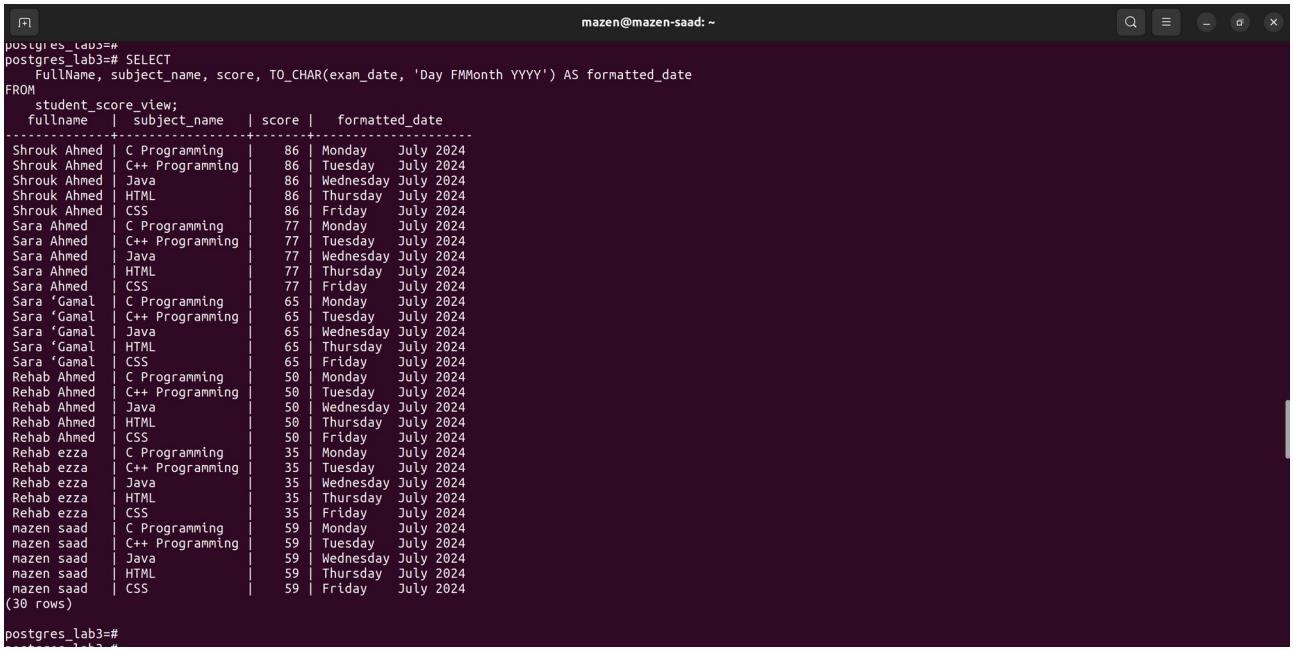
7. Create a temporary view for all subjects with their max_score.

```
CREATE TEMP VIEW temp_subject_max_score
    AS SELECT subject.subject_name, MAX(exam_results.score) AS
max_score
FROM
    subject
JOIN
    exam ON subject.subject_id = exam.subject_id
JOIN
    exam_results ON exam.exam_id = exam_results.exam_id
GROUP BY
    subject.subject_name;

select * from temp_subject_max_score;
```



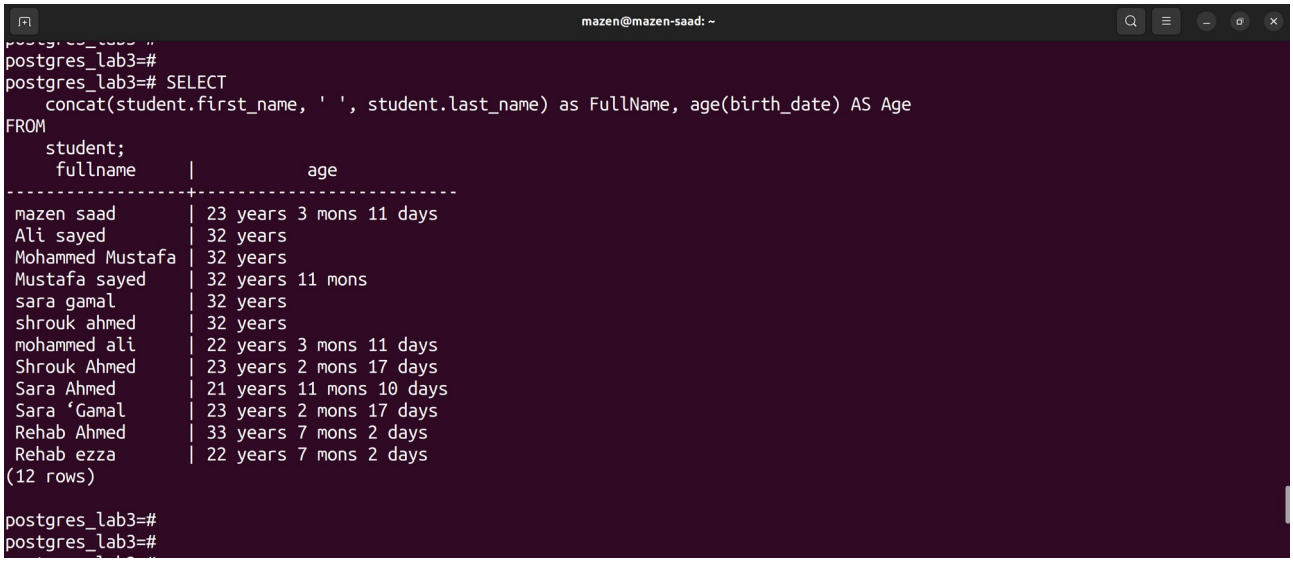
10. (from Q.6) Display the date of exam as the following: day 'month name' year.

```
SELECT
    FullName, subject_name, score, TO_CHAR(exam_date, 'Day FMMonth YYYY') AS formatted_date
FROM
    student_score_view;
```



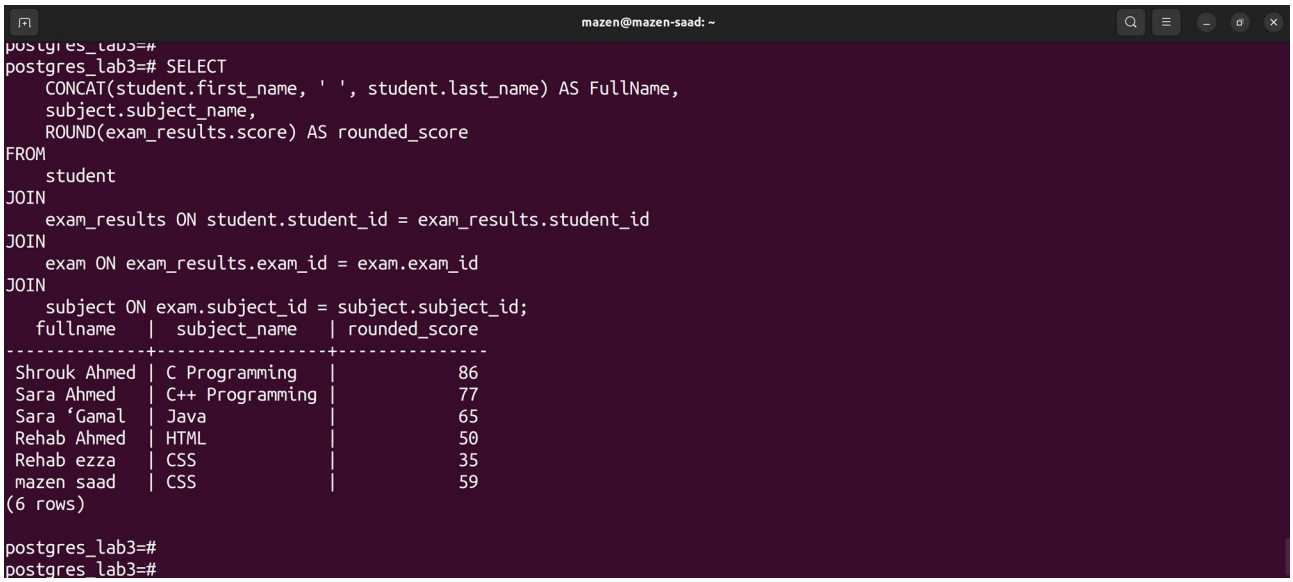
11. Display name and age of each students

```
SELECT
    concat(student.first_name, ' ', student.last_name) as FullName,
age(birth_date) AS Age
FROM
    student;
```



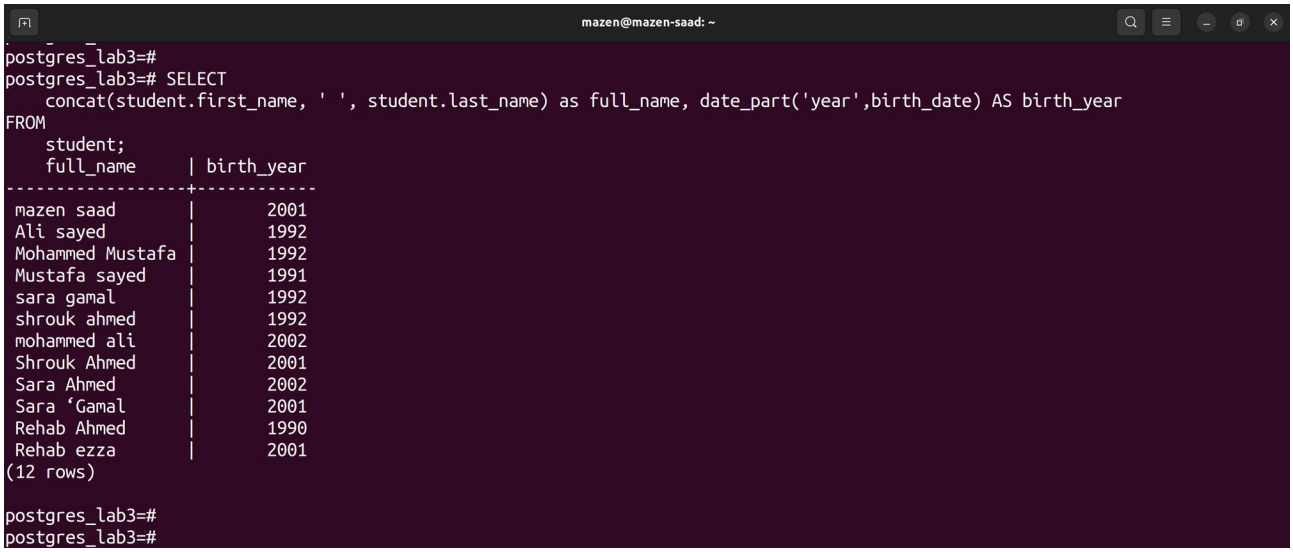
12. Display the name of students with their Rounded score in each subject

```
SELECT
    CONCAT(student.first_name, ' ', student.last_name) AS
FullName,
    subject.subject_name,
    ROUND(exam_results.score) AS rounded_score
FROM
    student
JOIN
    exam_results ON student.student_id = exam_results.student_id
JOIN
    exam ON exam_results.exam_id = exam.exam_id
JOIN
    subject ON exam.subject_id = subject.subject_id;
```



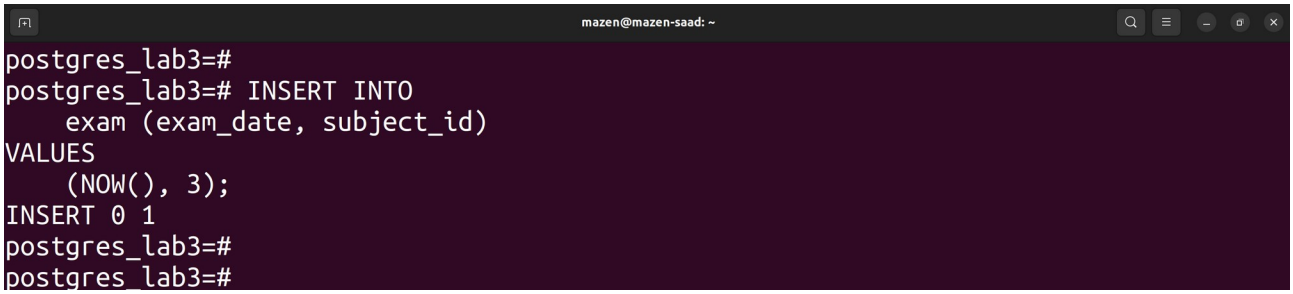
13. Display the name of students with the year of Birthdate

```
SELECT
    concat(student.first_name, ' ', student.last_name) as full_name,
date_part('year',birth_date) AS birth_year
FROM
    student;
```



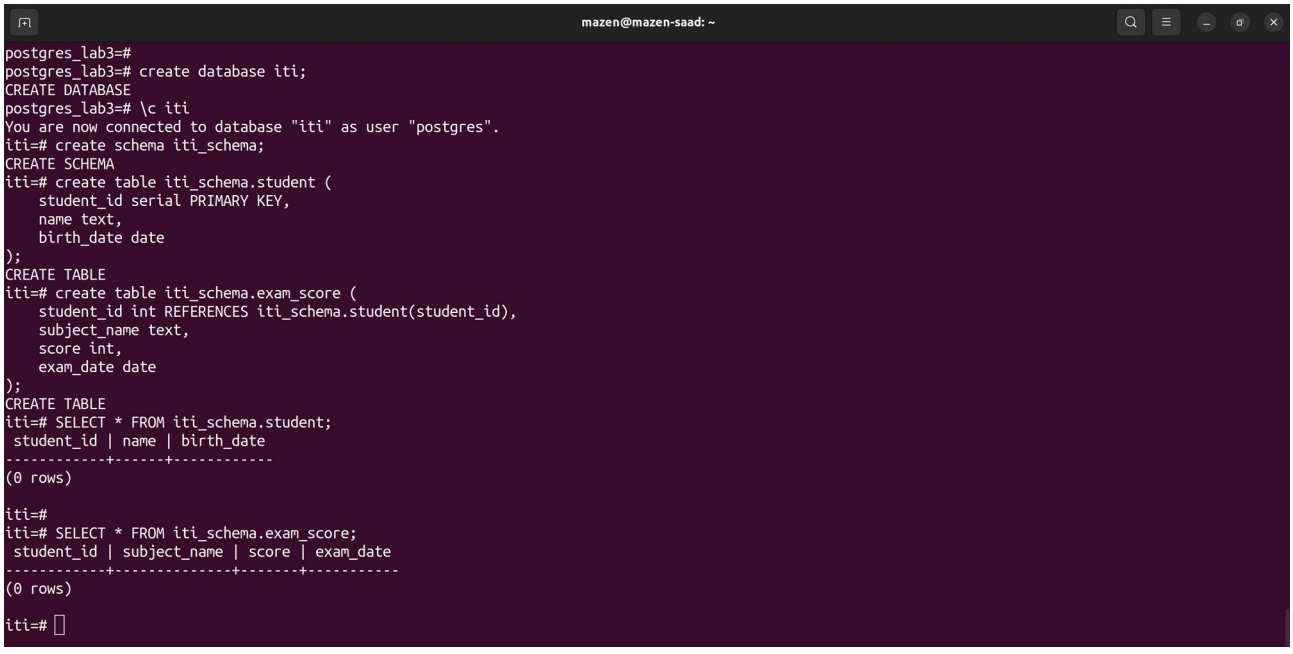
14. Add new exam result, in date column use NOW() function;

```
INSERT INTO
    exam (exam_date, subject_id)
VALUES
    (NOW(), 3);
```



15. Create database called ITI, and create different schema and Tables inside this schema

```
create database iti;
\c iti

create schema iti_schema;

create table iti_schema.student (
    student_id serial PRIMARY KEY,
    name text,
    birth_date date
);
create table iti_schema.exam_score (
    student_id int REFERENCES iti_schema.student(student_id),
    subject_name text,
    score int,
    exam_date date
);

SELECT * FROM iti_schema.student;
SELECT * FROM iti_schema.exam_score;
```

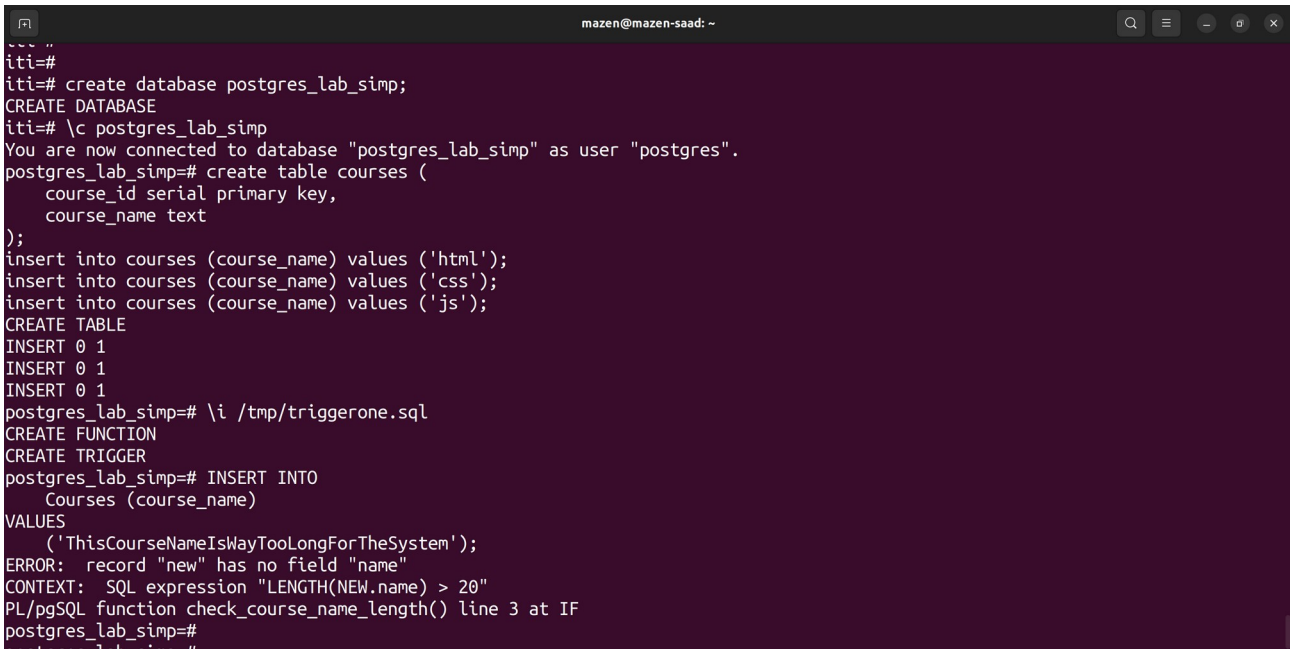


lab simp

1. Create trigger to prevent insert new Course with name length greater than 20 chars;

```
su - postgres
mazen@@1
psql
create database postgres_lab_simp;
\l
\c postgres_lab_simp

create table courses (
    course_id serial primary key,
    course_name text
);
insert into courses (course_name) values ('html');
insert into courses (course_name) values ('css');
insert into courses (course_name) values ('js');


\i /tmp/triggerone.sql


CREATE OR REPLACE FUNCTION check_course_name_length()
RETURNS TRIGGER AS $$
BEGIN
    IF LENGTH(NEW.name) > 20 THEN
        RAISE EXCEPTION 'Course name cannot exceed 20
characters';
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER course_name_length_trigger
```

```
BEFORE INSERT ON courses
FOR EACH ROW
EXECUTE FUNCTION check_course_name_length();


INSERT INTO
    Courses (course_name)
VALUES
    ('ThisCourseNameIsWayTooLongForTheSystem');
```

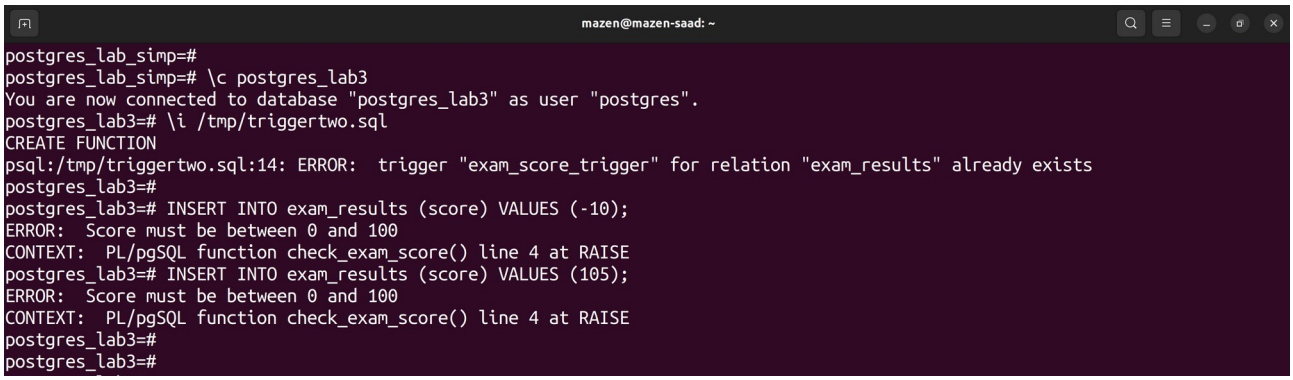


2. Create trigger to prevent user to insert or update Exam with Score greater than 100 or less than zero

\c postgres_lab3

\i /tmp/triggertwo.sql

```
CREATE OR REPLACE FUNCTION check_exam_score()
RETURNS TRIGGER AS $$
BEGIN
    IF NEW.score < 0 OR NEW.score > 100 THEN
        RAISE EXCEPTION 'Score must be between 0 and 100';
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER exam_score_trigger
BEFORE INSERT OR UPDATE ON exam_results
FOR EACH ROW
EXECUTE FUNCTION check_exam_score();


INSERT INTO exam_results (score) VALUES (-10);
INSERT INTO exam_results (score) VALUES (105);
INSERT INTO exam_results (score) VALUES (85);
```



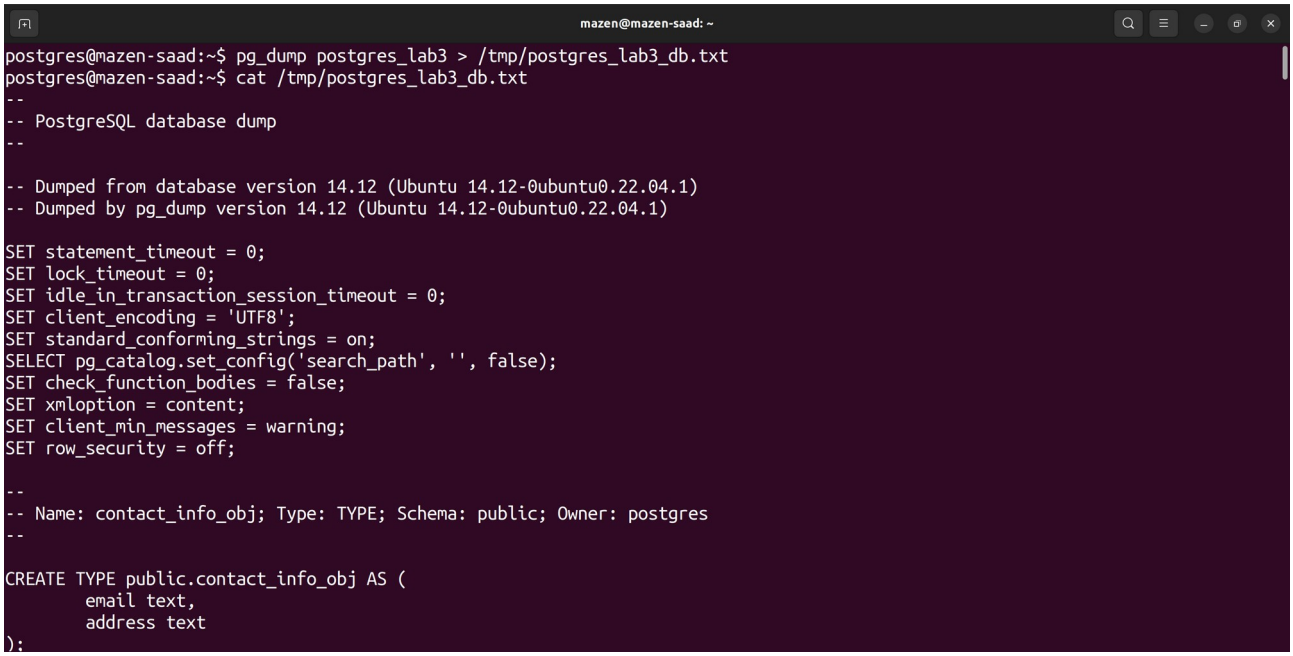
3. (bonus) Create trigger to prevent any user to update/insert/delete to all tables (Students, Exams, Tracks,..) after 7:00 PM

\i /tmp/triggerthree.sql

```
CREATE OR REPLACE FUNCTION restrict_after_7pm()
RETURNS TRIGGER AS $$
BEGIN
    IF EXTRACT(HOUR FROM CURRENT_TIME) >= 19 THEN
        RAISE EXCEPTION 'Updates, inserts, or deletes are not
allowed after 7:00 PM';
    END IF;
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER restrict_time_trigger
BEFORE INSERT OR UPDATE OR DELETE ON student
FOR EACH ROW
EXECUTE FUNCTION restrict_after_7pm();

INSERT INTO
    student (first_name)
VALUES
    ('Mazen');
```

4. Backup your Database to external file

pg_dump postgres_lab3 > /tmp/postgres_lab3_db.txt



5. Backup your Student table to external file

su - postgres
mazen@@1
psql
\c postgres_lab3

copy student to '/tmp/student_table.txt';