**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

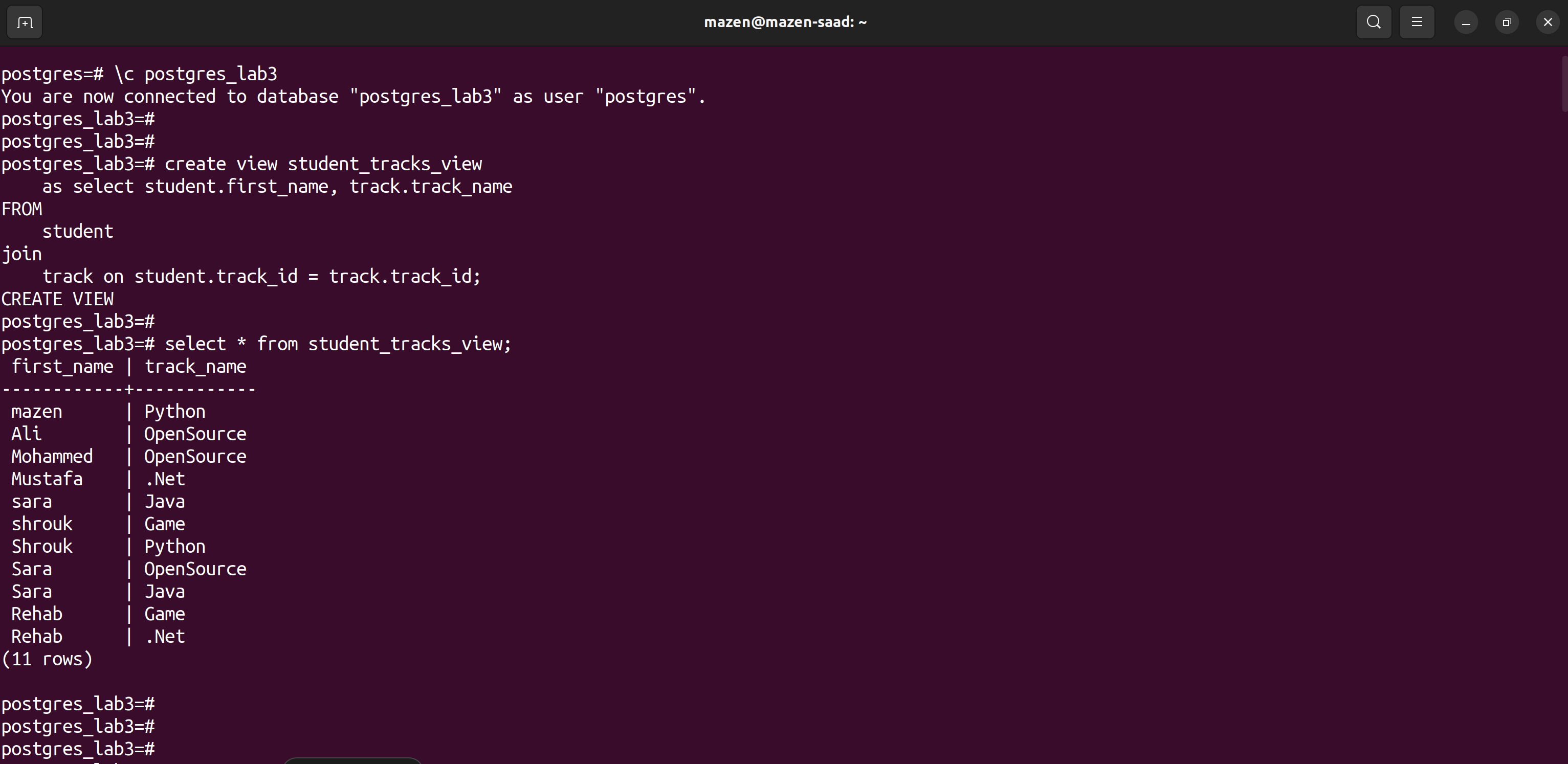
FROM

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

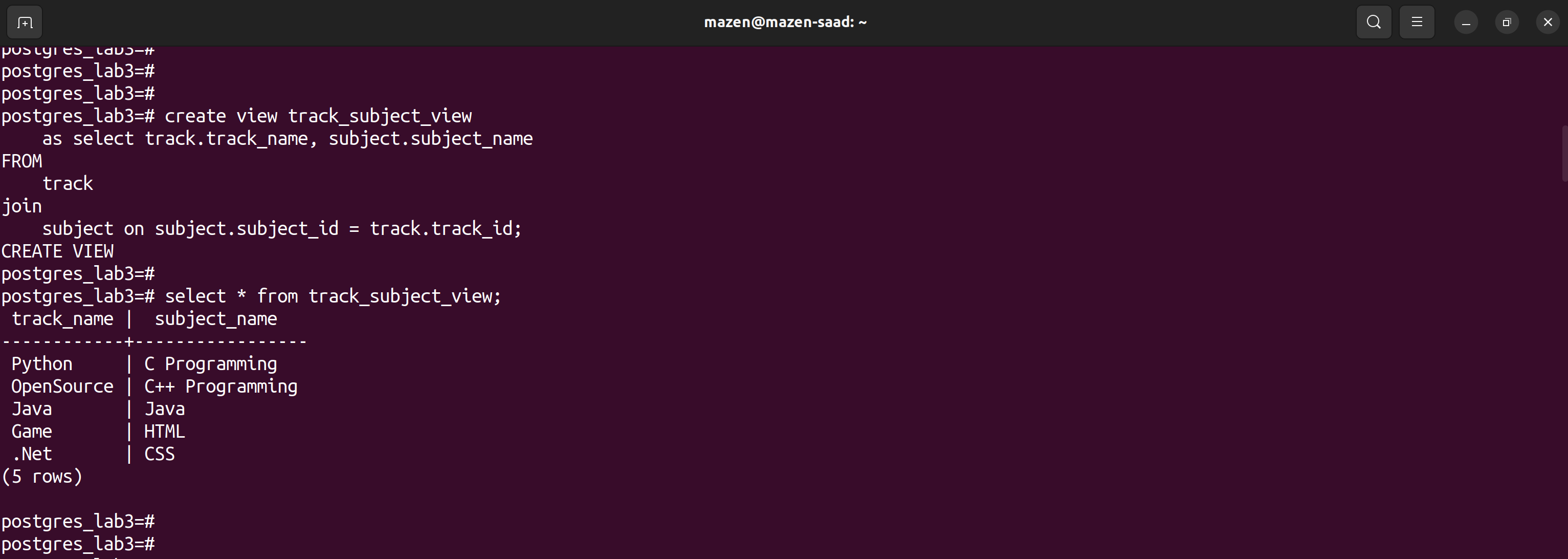
FROM

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

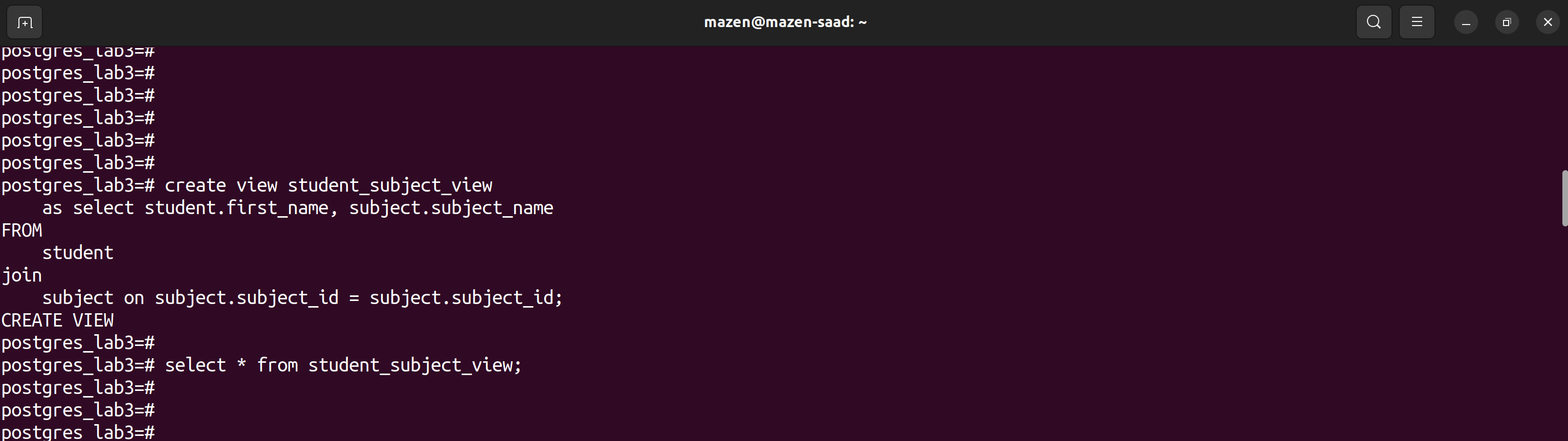
FROM

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

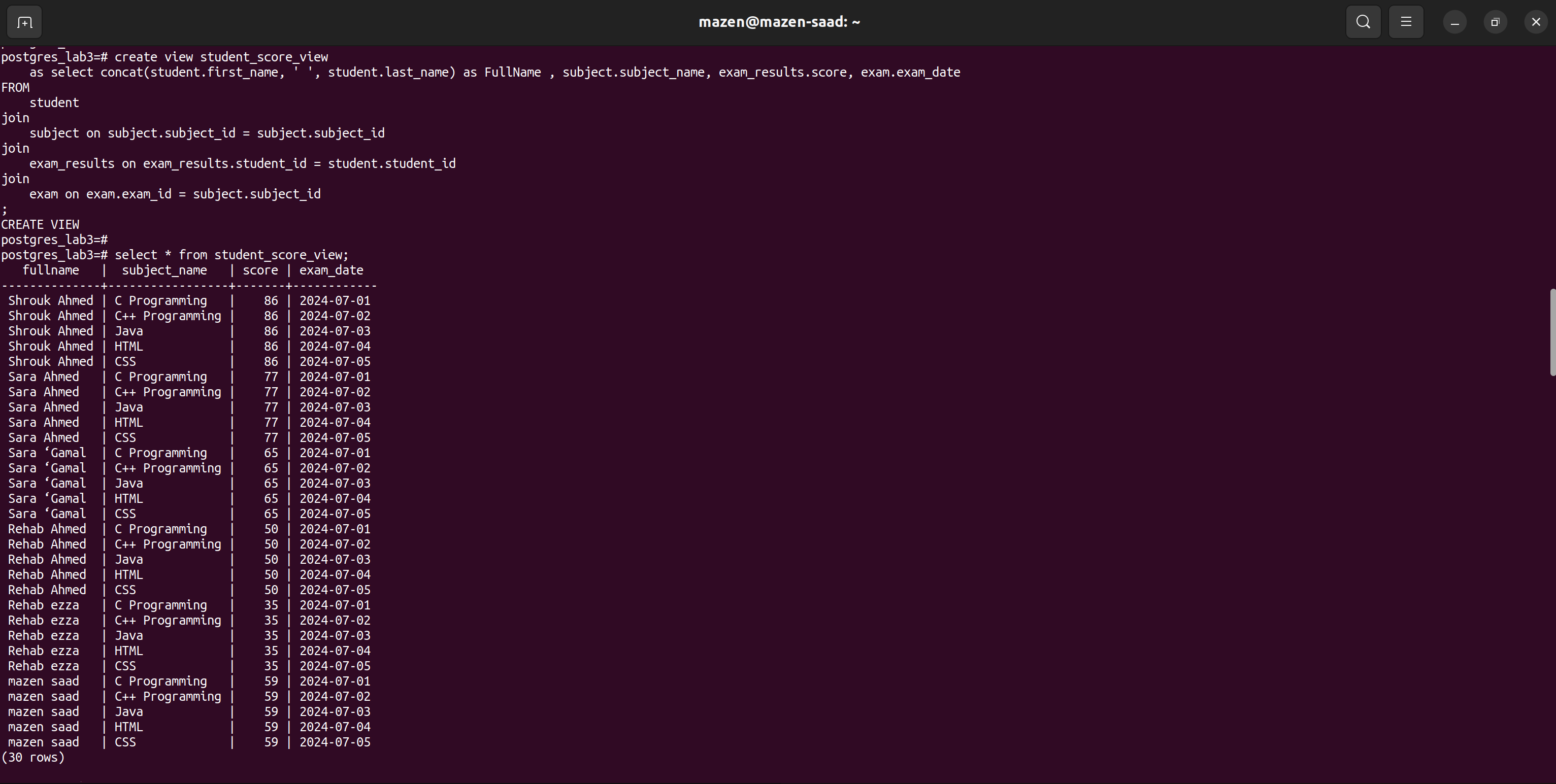
exam\_results on exam\_results.student\_id = student.student\_id

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

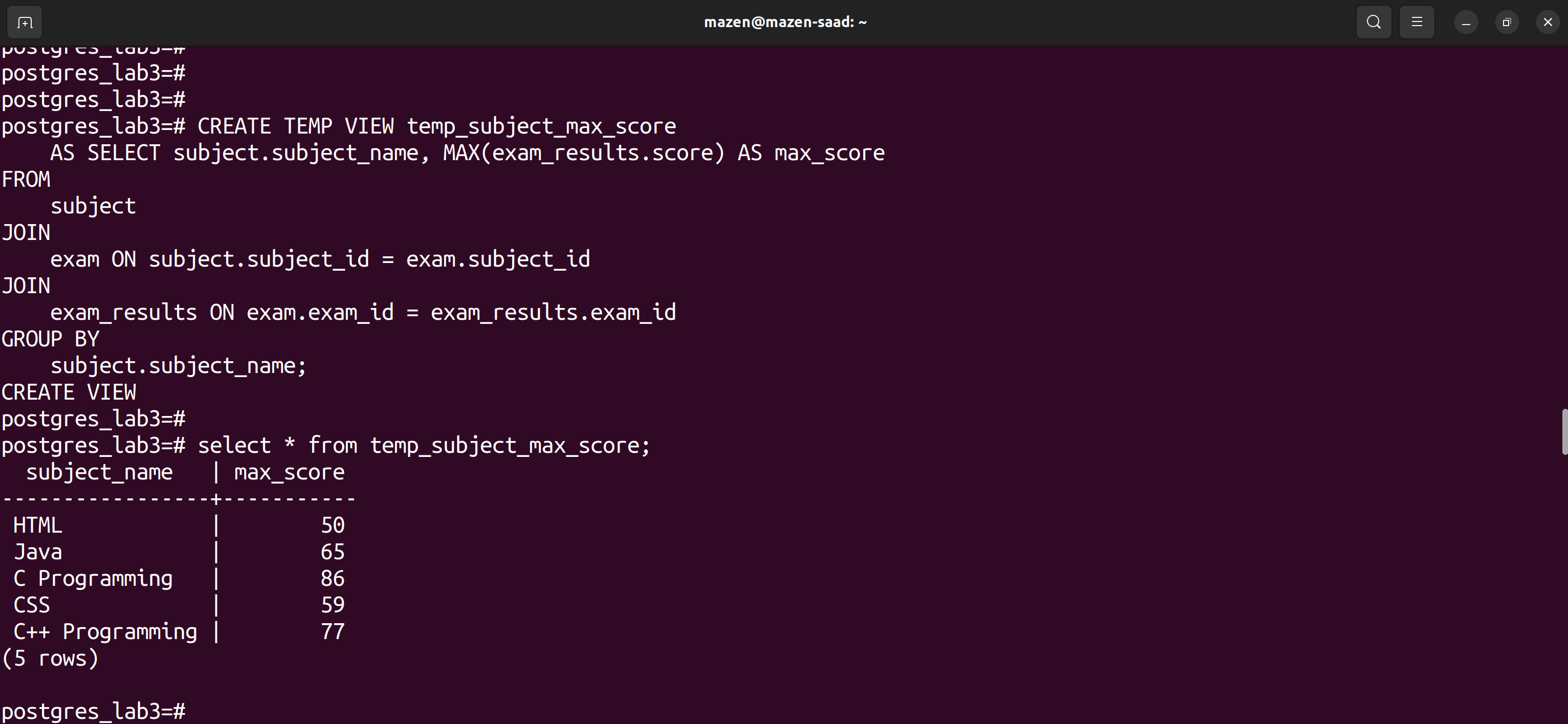
JOIN

exam\_results ON exam.exam\_id = exam\_results.exam\_id

GROUP BY

subject.subject\_name;

select \* from temp\_subject\_max\_score;



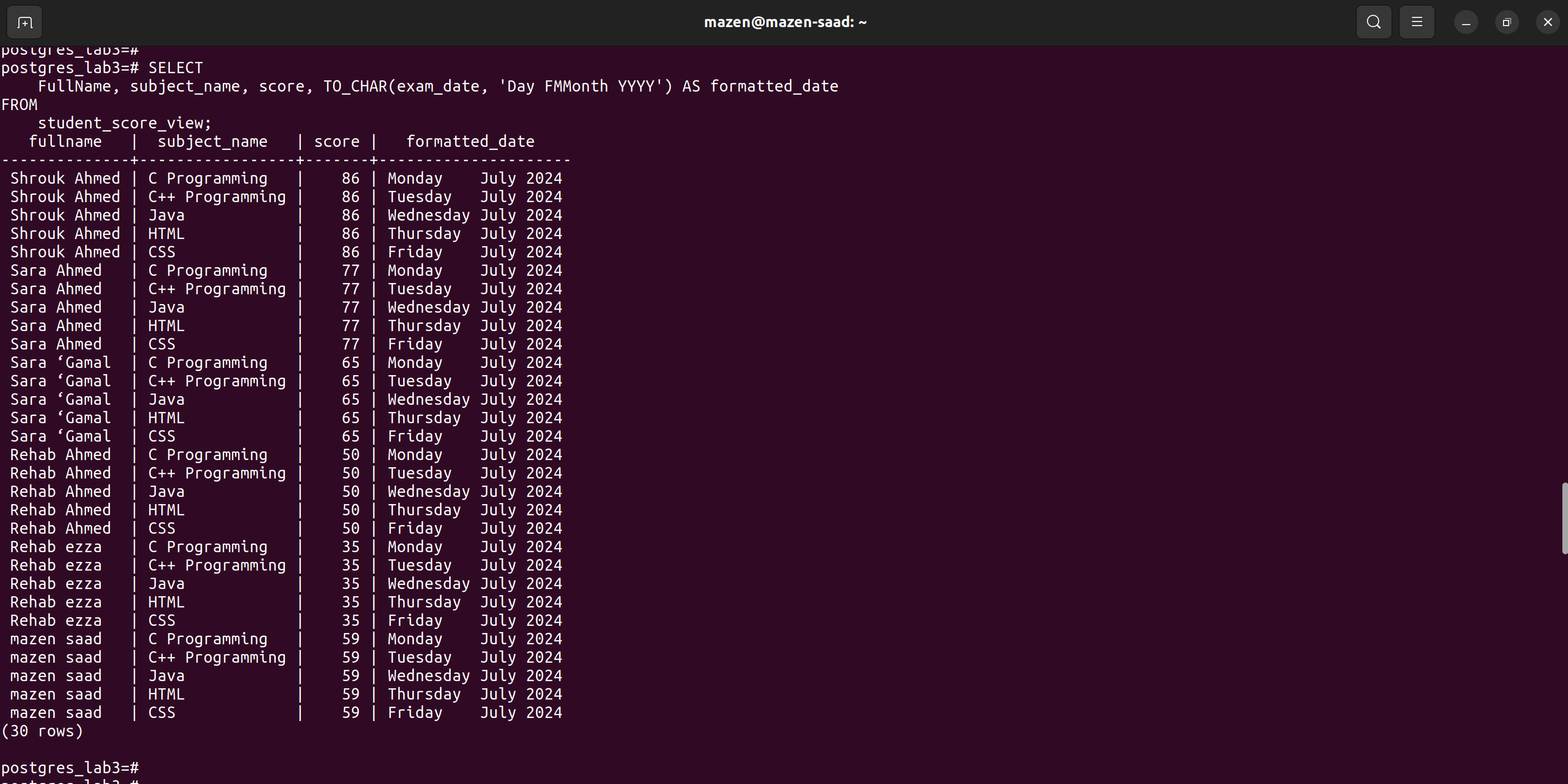
10. (from Q.6) Display the date of exam as the following: day 'month name' year.

SELECT

FullName, subject\_name, score, TO\_CHAR(exam\_date, 'Day FMMonth YYYY') AS formatted\_date

FROM

student\_score\_view;



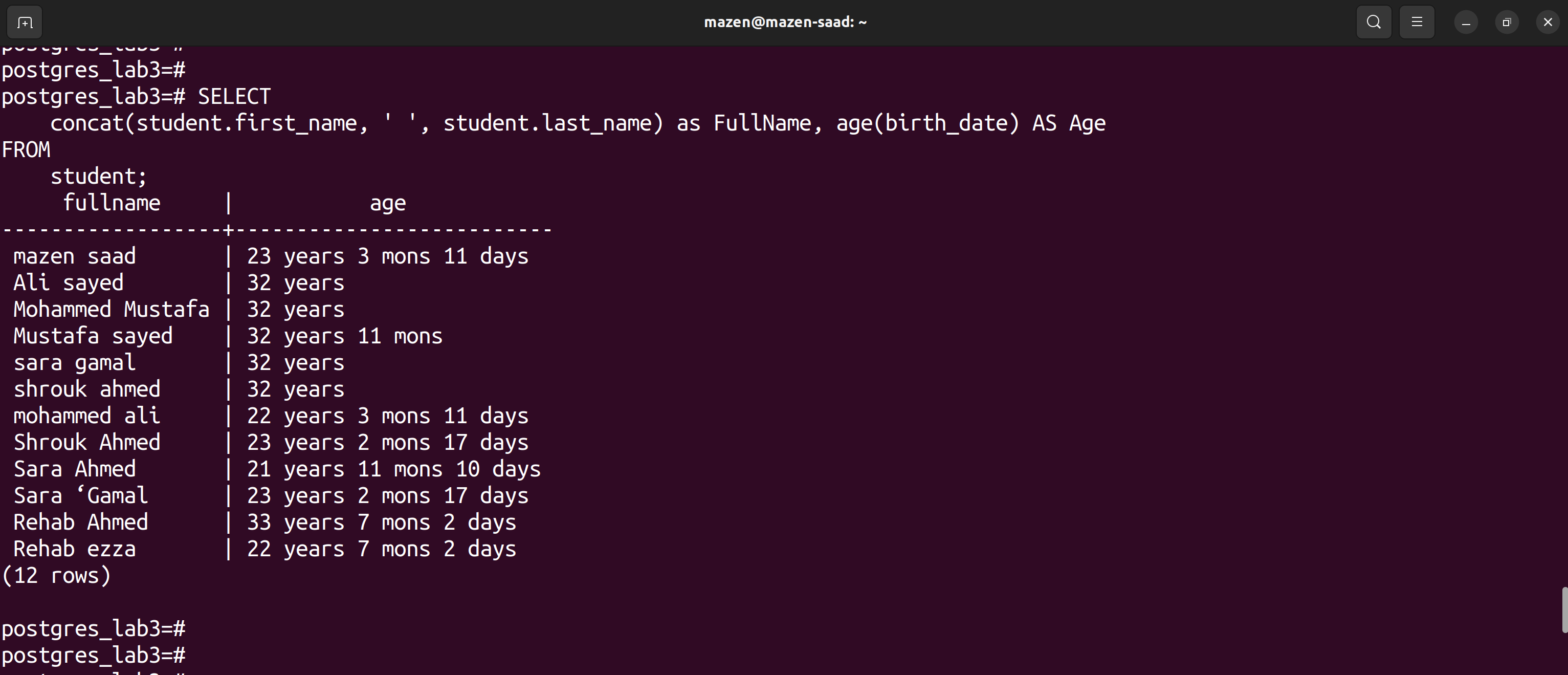
11. Display name and age of each students

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

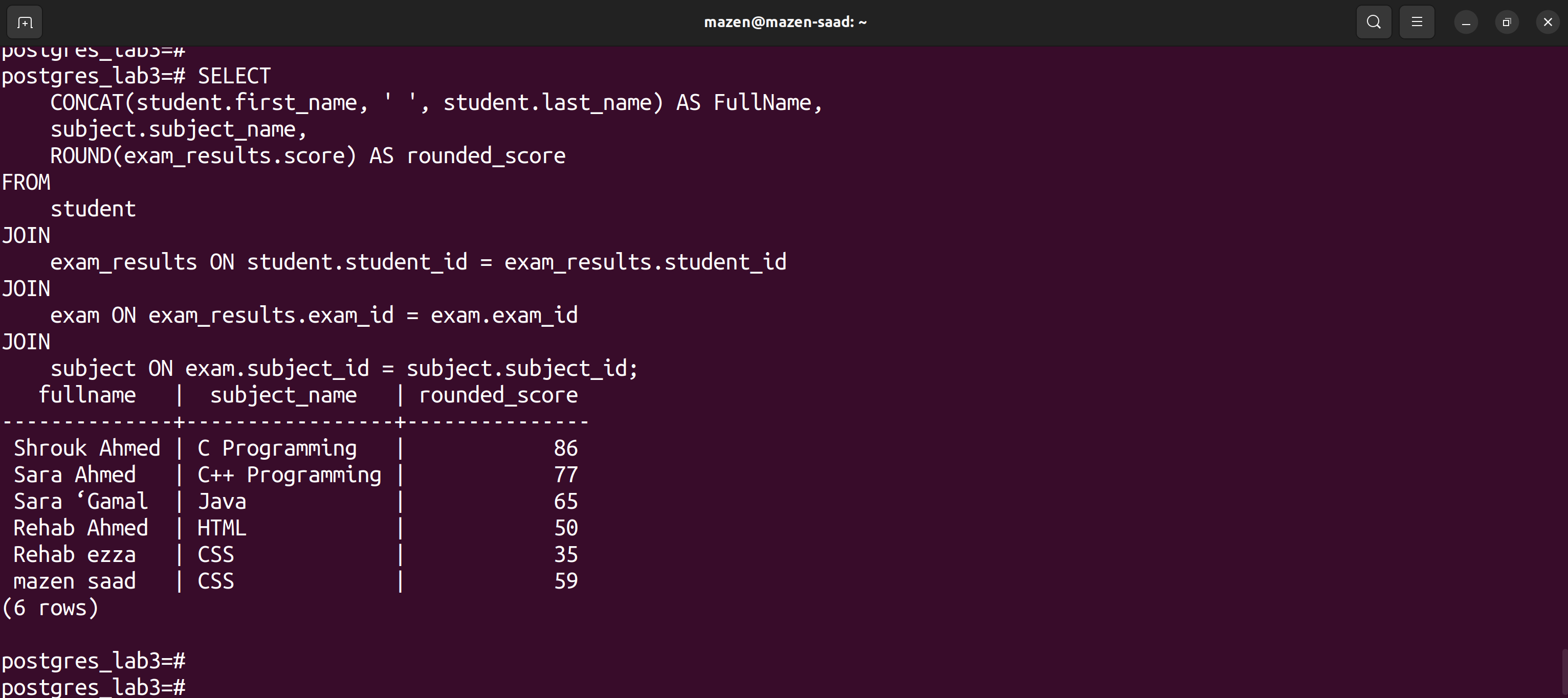
exam\_results ON student.student\_id = exam\_results.student\_id

JOIN

exam ON exam\_results.exam\_id = exam.exam\_id

JOIN

subject ON exam.subject\_id = subject.subject\_id;



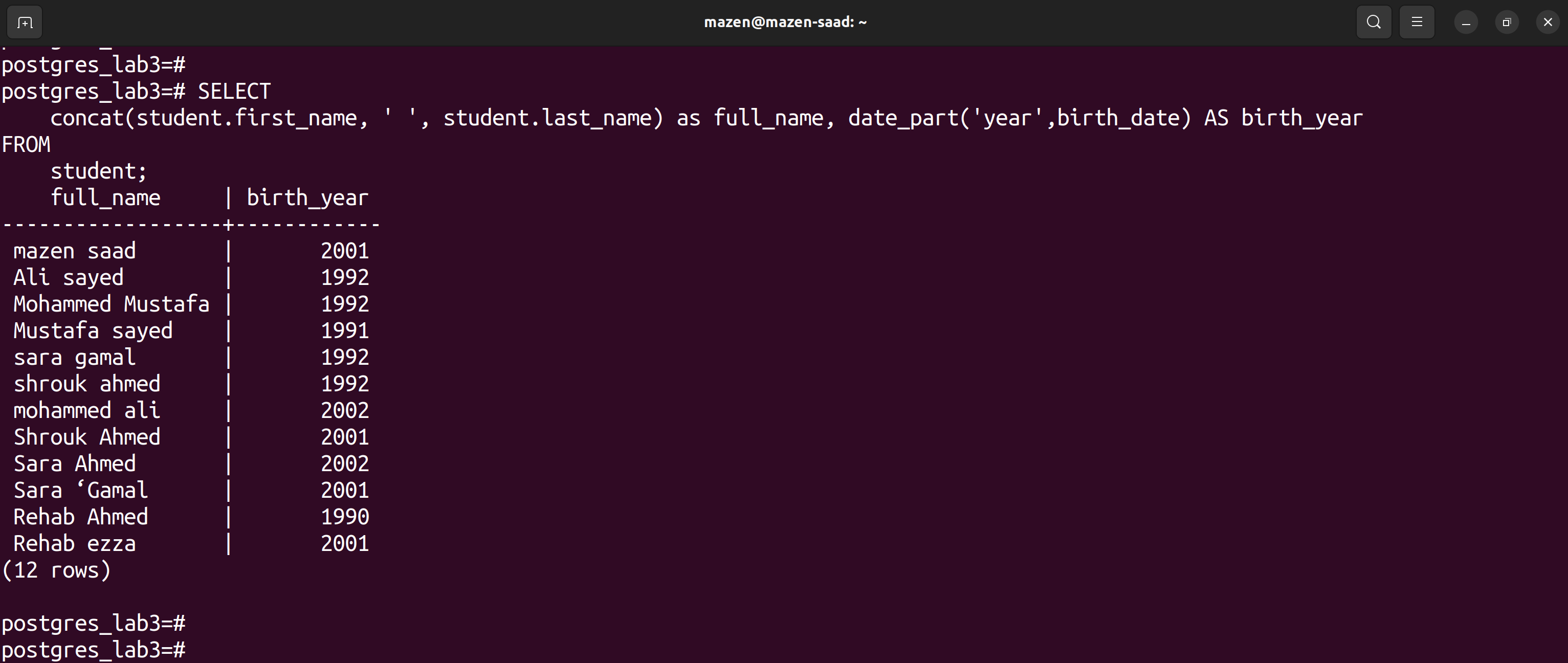
13. Display the name of students with the year of Birthdate

SELECT

concat(student.first\_name, ' ', student.last\_name) as full\_name, date\_part('year',birth\_date) AS birth\_year

FROM

student;



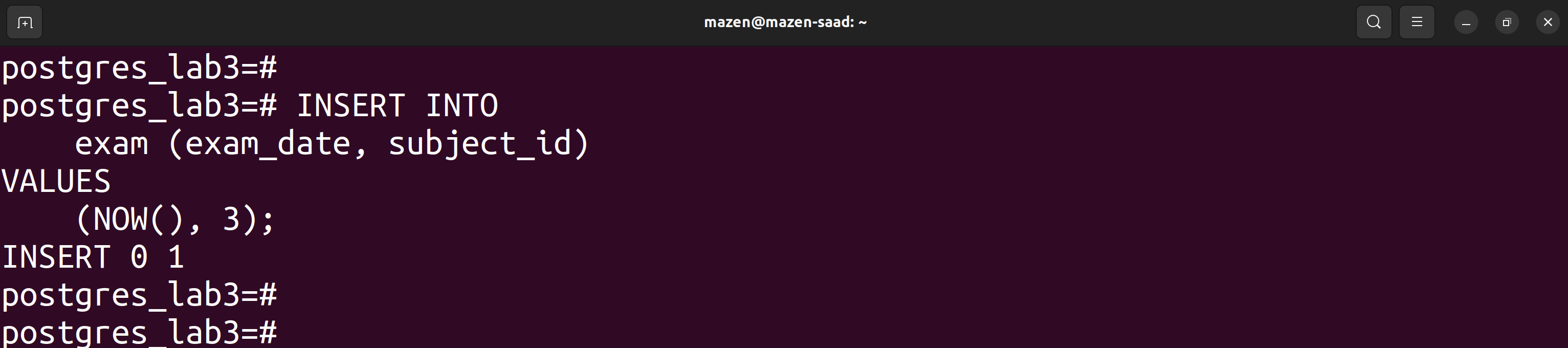
14. Add new exam result, in date column use NOW() function;

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,

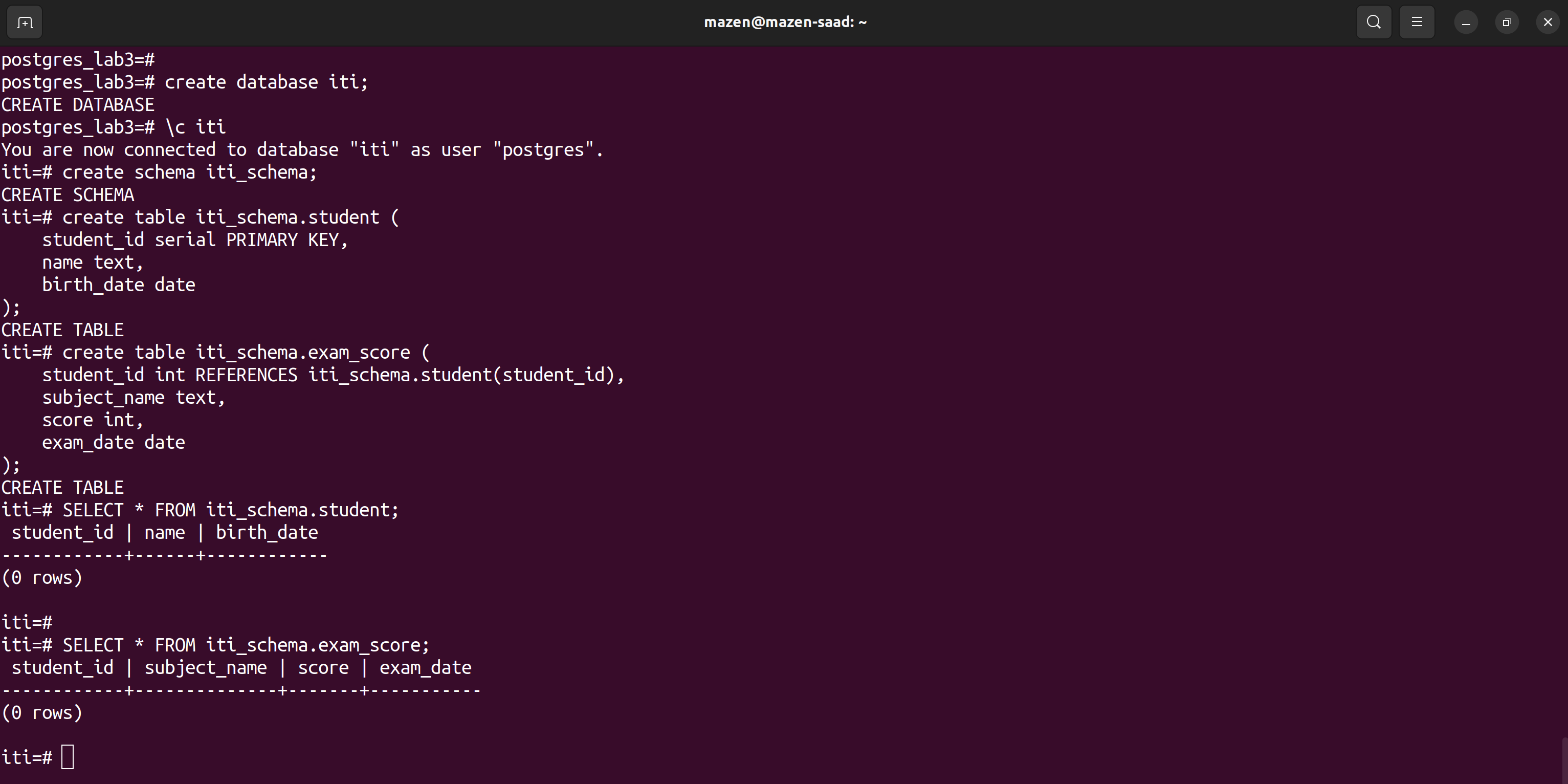
score int,

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

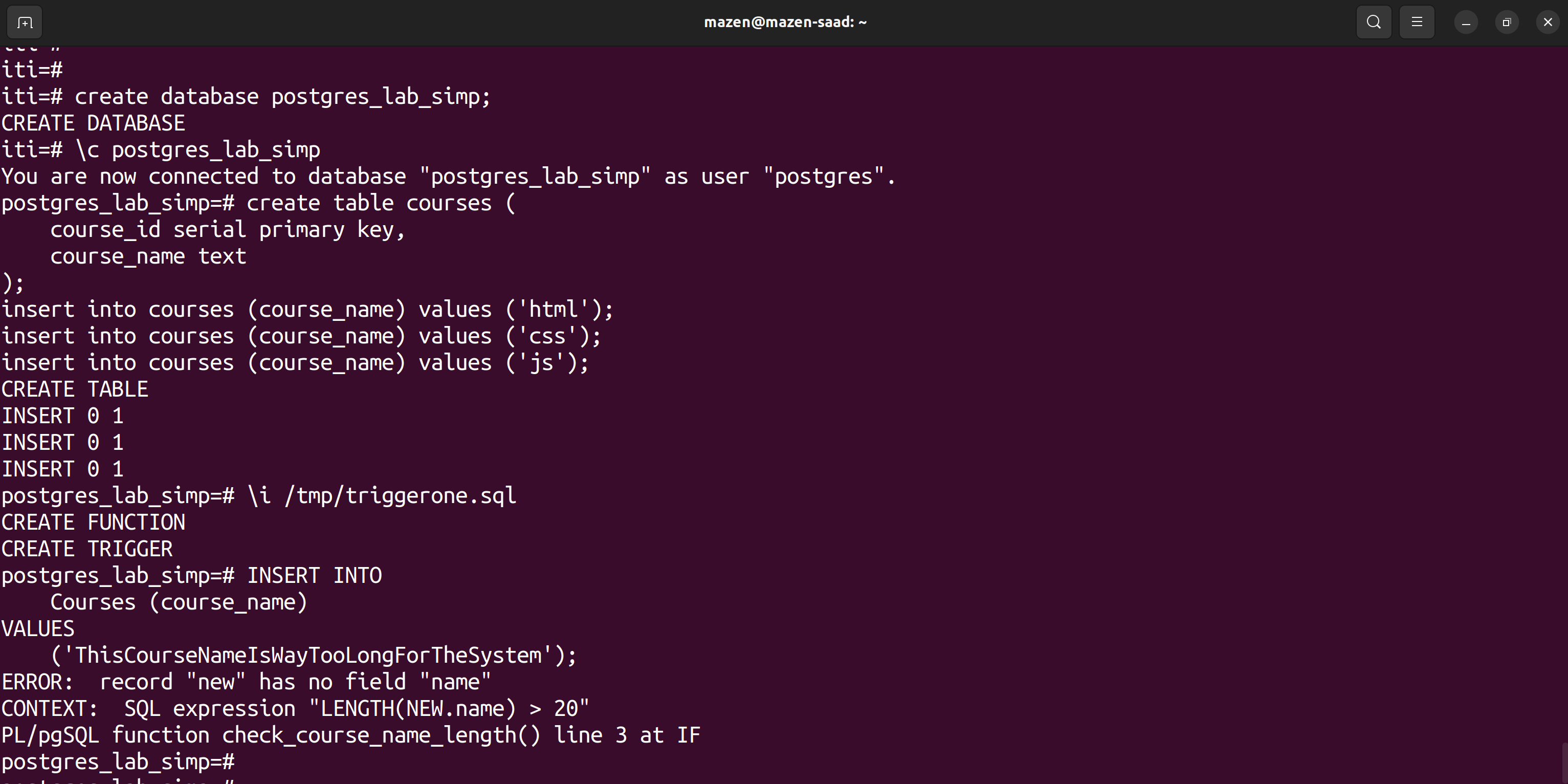
EXECUTE FUNCTION check\_course\_name\_length();

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

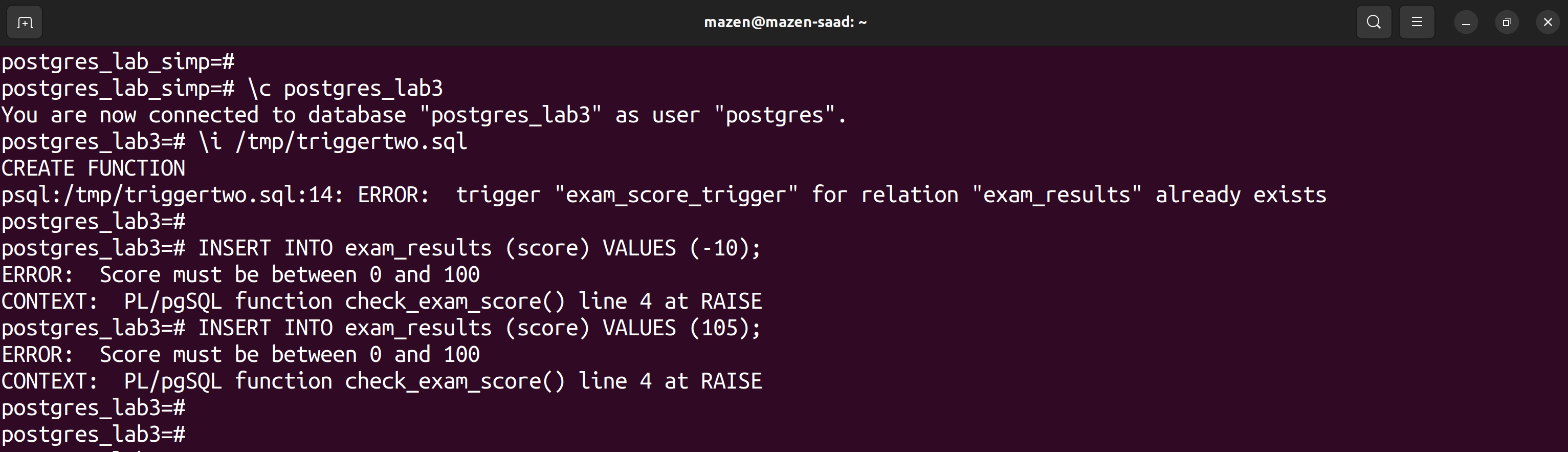
FOR EACH ROW

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

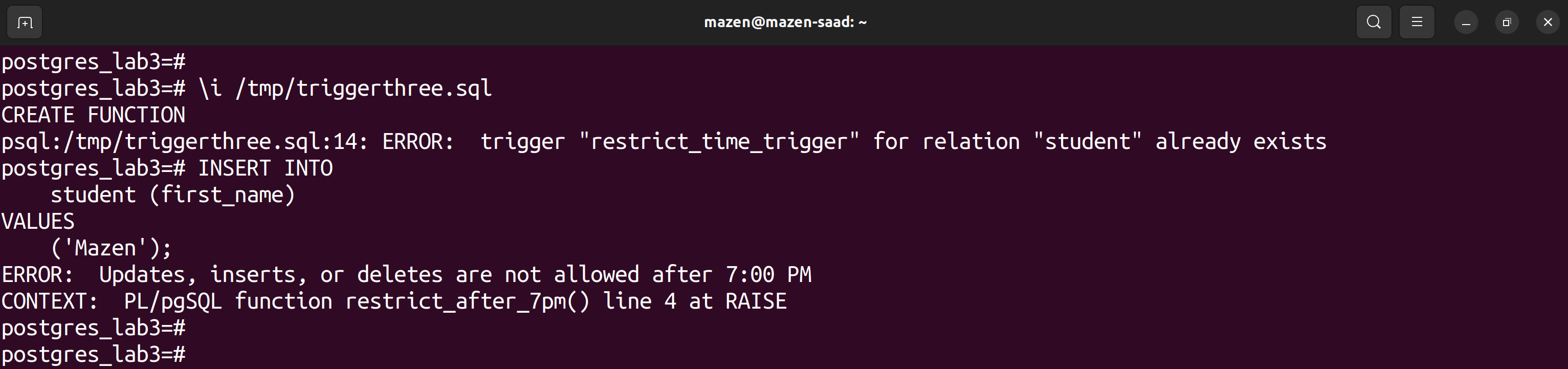
EXECUTE FUNCTION restrict\_after\_7pm();

INSERT INTO

student (first\_name)

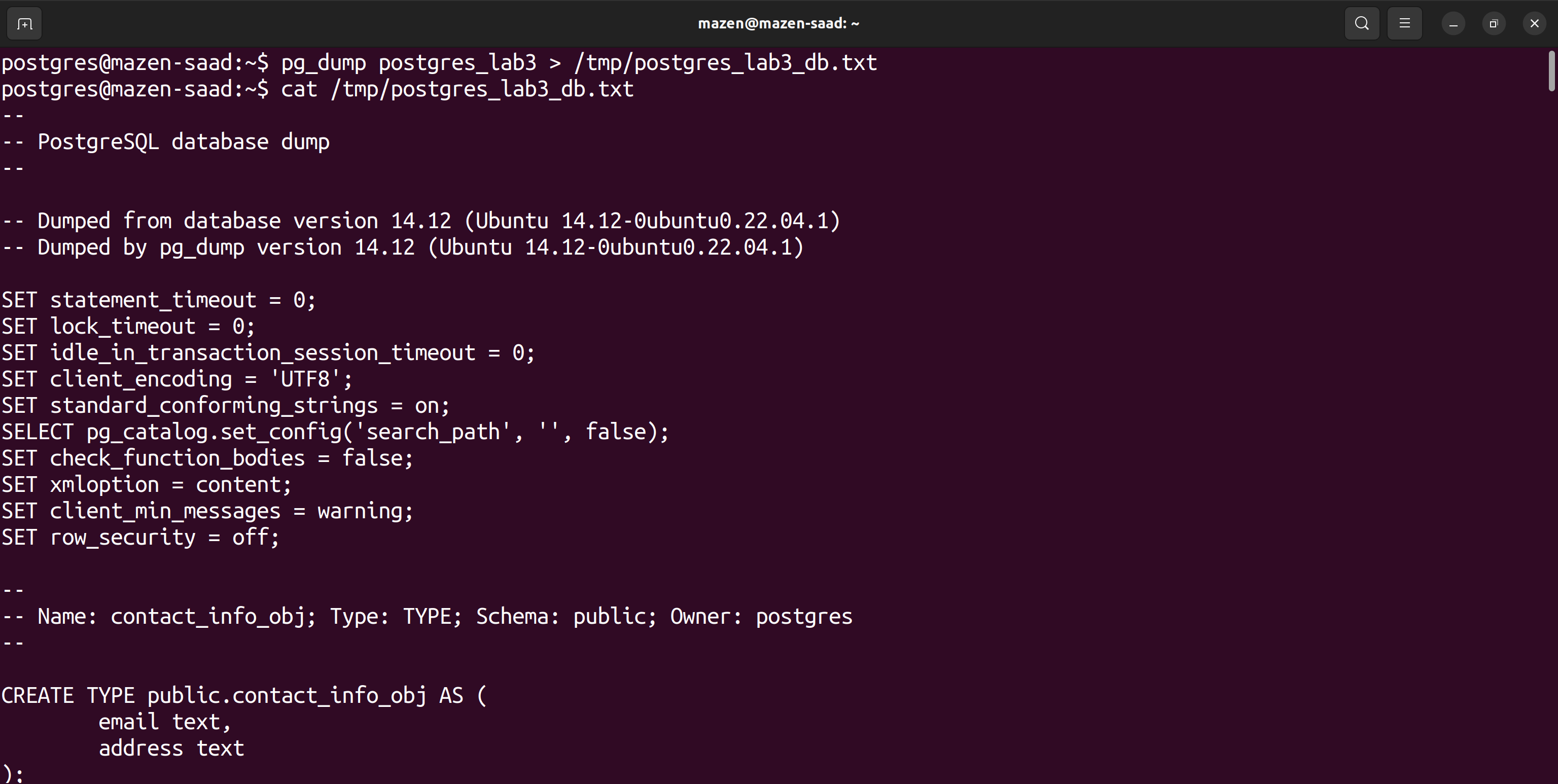
VALUES

('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';

