1. select id, name, student\_id, email,password, facial\_path, joined\_date,prefix,programme,current\_year,current\_semester from students;

trigger for sections students:  
-- Create function to update current\_students on insert

CREATE OR REPLACE FUNCTION update\_current\_students\_on\_insert()

RETURNS TRIGGER AS $$

BEGIN

UPDATE sections

SET current\_students = (

SELECT COUNT(\*)

FROM student\_subjects

WHERE subject\_section = NEW.subject\_section

AND subject\_id = NEW.subject\_id

)

WHERE section\_number = NEW.subject\_section

AND subject\_id = NEW.subject\_id;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

-- Create trigger for insert operations

CREATE TRIGGER after\_insert\_student\_subject

AFTER INSERT ON student\_subjects

FOR EACH ROW

EXECUTE FUNCTION update\_current\_students\_on\_insert();

-- Create function to update current\_students on delete

CREATE OR REPLACE FUNCTION update\_current\_students\_on\_delete()

RETURNS TRIGGER AS $$

BEGIN

UPDATE sections

SET current\_students = (

SELECT COUNT(\*)

FROM student\_subjects

WHERE subject\_section = OLD.subject\_section

AND subject\_id = OLD.subject\_id

)

WHERE section\_number = OLD.subject\_section

AND subject\_id = OLD.subject\_id;

RETURN OLD;

END;

$$ LANGUAGE plpgsql;

-- Create trigger for delete operations

CREATE TRIGGER after\_delete\_student\_subject

AFTER DELETE ON student\_subjects

FOR EACH ROW

EXECUTE FUNCTION update\_current\_students\_on\_delete();

-- Create function to update current\_students on update

CREATE OR REPLACE FUNCTION update\_current\_students\_on\_update()

RETURNS TRIGGER AS $$

BEGIN

-- Handle the old section

UPDATE sections

SET current\_students = (

SELECT COUNT(\*)

FROM student\_subjects

WHERE subject\_section = OLD.subject\_section

AND subject\_id = OLD.subject\_id

)

WHERE section\_number = OLD.subject\_section

AND subject\_id = OLD.subject\_id;

-- Handle the new section if it changed

IF NEW.subject\_section IS DISTINCT FROM OLD.subject\_section OR

NEW.subject\_id IS DISTINCT FROM OLD.subject\_id THEN

UPDATE sections

SET current\_students = (

SELECT COUNT(\*)

FROM student\_subjects

WHERE subject\_section = NEW.subject\_section

AND subject\_id = NEW.subject\_id

)

WHERE section\_number = NEW.subject\_section

AND subject\_id = NEW.subject\_id;

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

-- Create trigger for update operations

CREATE TRIGGER after\_update\_student\_subject

AFTER UPDATE ON student\_subjects

FOR EACH ROW

EXECUTE FUNCTION update\_current\_students\_on\_update();”

Delete all trigger and function in sections:   
-- Drop existing triggers

DROP TRIGGER IF EXISTS after\_insert\_student\_subject ON student\_subjects;

DROP TRIGGER IF EXISTS after\_delete\_student\_subject ON student\_subjects;

DROP TRIGGER IF EXISTS after\_update\_student\_subject ON student\_subjects;

-- Drop existing functions

DROP FUNCTION IF EXISTS update\_current\_students\_on\_insert();

DROP FUNCTION IF EXISTS update\_current\_students\_on\_delete();

DROP FUNCTION IF EXISTS update\_current\_students\_on\_update();

-- Drop existing functions

DROP FUNCTION IF EXISTS update\_current\_students\_on\_insert();

DROP FUNCTION IF EXISTS update\_current\_students\_on\_delete();

DROP FUNCTION IF EXISTS update\_current\_students\_on\_update();

DROP FUNCTION IF EXISTS update\_current\_lecturers\_on\_insert();

DROP FUNCTION IF EXISTS update\_current\_lecturers\_on\_delete();

DROP FUNCTION IF EXISTS update\_current\_lecturers\_on\_update();

Trigger for section lecturers:

-- Count the number of lecturers for each section

WITH lecturer\_counts AS (

SELECT

subject\_section,

subject\_id,

COUNT(DISTINCT lecturer\_id) AS lecturer\_count

FROM lecturer\_subjects

GROUP BY subject\_section, subject\_id

)

-- Update the sections table with the counted number of lecturers

UPDATE sections s

SET current\_lecturers = lc.lecturer\_count

FROM lecturer\_counts lc

WHERE s.section\_number = lc.subject\_section

AND s.subject\_id = lc.subject\_id;

-- Function to update current\_lecturers on insert

CREATE OR REPLACE FUNCTION update\_current\_lecturers\_on\_insert()

RETURNS TRIGGER AS $$

BEGIN

UPDATE sections

SET current\_lecturers = (

SELECT COUNT(DISTINCT lecturer\_id)

FROM lecturer\_subjects

WHERE subject\_section = NEW.subject\_section

AND subject\_id = NEW.subject\_id

)

WHERE section\_number = NEW.subject\_section

AND subject\_id = NEW.subject\_id;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

-- Trigger for insert operations

CREATE TRIGGER after\_insert\_lecturer\_subject

AFTER INSERT ON lecturer\_subjects

FOR EACH ROW

EXECUTE FUNCTION update\_current\_lecturers\_on\_insert();

-- Function to update current\_lecturers on delete

CREATE OR REPLACE FUNCTION update\_current\_lecturers\_on\_delete()

RETURNS TRIGGER AS $$

BEGIN

UPDATE sections

SET current\_lecturers = (

SELECT COUNT(DISTINCT lecturer\_id)

FROM lecturer\_subjects

WHERE subject\_section = OLD.subject\_section

AND subject\_id = OLD.subject\_id

)

WHERE section\_number = OLD.subject\_section

AND subject\_id = OLD.subject\_id;

RETURN OLD;

END;

$$ LANGUAGE plpgsql;

-- Trigger for delete operations

CREATE TRIGGER after\_delete\_lecturer\_subject

AFTER DELETE ON lecturer\_subjects

FOR EACH ROW

EXECUTE FUNCTION update\_current\_lecturers\_on\_delete();

-- Function to update current\_lecturers on update

CREATE OR REPLACE FUNCTION update\_current\_lecturers\_on\_update()

RETURNS TRIGGER AS $$

BEGIN

-- Update the old section

UPDATE sections

SET current\_lecturers = (

SELECT COUNT(DISTINCT lecturer\_id)

FROM lecturer\_subjects

WHERE subject\_section = OLD.subject\_section

AND subject\_id = OLD.subject\_id

)

WHERE section\_number = OLD.subject\_section

AND subject\_id = OLD.subject\_id;

-- Update the new section if it changed

IF NEW.subject\_section IS DISTINCT FROM OLD.subject\_section OR

NEW.subject\_id IS DISTINCT FROM OLD.subject\_id THEN

UPDATE sections

SET current\_lecturers = (

SELECT COUNT(DISTINCT lecturer\_id)

FROM lecturer\_subjects

WHERE subject\_section = NEW.subject\_section

AND subject\_id = NEW.subject\_id

)

WHERE section\_number = NEW.subject\_section

AND subject\_id = NEW.subject\_id;

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

-- Trigger for update operations

CREATE TRIGGER after\_update\_lecturer\_subject

AFTER UPDATE ON lecturer\_subjects

FOR EACH ROW

EXECUTE FUNCTION update\_current\_lecturers\_on\_update();