

**ĐẠI HỌC BÁCH KHOA HÀ NỘI**  
**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

-----\*\*\*-----



**BÁO CÁO PROJECT CUỐI KỲ**

**Môn học:** Thực hành cơ sở dữ liệu (IT3290)

**Chủ đề:** Thiết kế cơ sở dữ liệu quản lý trung tâm tiếng Anh

**Giáo viên hướng dẫn:** Trần Văn Đăng

Nguyễn Thị Oanh

**Nhóm 4**

Họ tên	Mã số sinh viên
Hoàng Yến Nhi	20225898
Phan Hoàng Hải	20225715
Nguyễn Hoàng Phúc	20225905

Hà Nội, năm 2024

<b>I. Mô tả dự án.....</b>	<b>3</b>
A. Mô tả nghiệp vụ.....	3
B. Các chức năng tiêu biểu.....	5
1. Đối với người quản lý.....	5
2. Đối với học viên.....	5
3. Đối với giáo viên.....	6
<b>II. Thiết kế cơ sở dữ liệu.....</b>	<b>6</b>
A. Sơ đồ thực thể - liên kết và danh sách các bảng.....	6
Sơ đồ thực thể - liên kết.....	6
Danh sách các bảng.....	7
B. Mô tả các bảng và các quan hệ của chúng.....	7
<b>III. Câu lệnh SQL.....</b>	<b>8</b>
A. Các hàm truy vấn và tính toán:.....	8
B. Các trigger:.....	40
C. Hiệu năng câu truy vấn:.....	43
<b>IV. Các khung nhìn.....</b>	<b>48</b>
<b>Khó khăn và giải pháp.....</b>	<b>54</b>
<b>Ưu điểm và nhược điểm.....</b>	<b>55</b>
A. Ưu điểm.....	55
B. Nhược điểm.....	55
<b>Nhiệm vụ của mỗi thành viên.....</b>	<b>55</b>

## I. Mô tả dự án

Ý tưởng khởi đầu của project là phát triển hệ thống quản lý trung tâm đào tạo tiếng Anh, nhằm giải quyết các vấn đề trong quản lý và vận hành của các trung tâm ngoại ngữ. Hệ thống được đề xuất sẽ tích hợp nhiều chức năng cần thiết để quản lý hiệu quả các hoạt động hằng ngày của trung tâm từ quản lý học viên, giáo viên, các khóa học, cho đến quản lý tài chính và báo cáo kết quả.

### A. Mô tả nghiệp vụ

Khi học viên mới đến trung tâm đào tạo tiếng Anh, người quản lý sẽ hướng dẫn học viên tạo tài khoản trên hệ thống của trung tâm. Lúc đó, học viên cần cung cấp các thông tin cá nhân bao gồm họ tên, số điện thoại, và địa chỉ. Đồng thời, học viên sẽ thực hiện đăng ký tham gia bài test đầu vào miễn phí. Bài test này gồm hai phần là bài test General và kỹ năng Nói.

- + Bài kỹ năng Nói: Học viên sẽ làm bài thi nói trực tiếp với giáo viên qua một link Meet được gắn cố định của từng giáo viên. Giáo viên sẽ chấm điểm bài nói trên thang điểm IELTS Speaking (0-9.0) và nhập điểm bài kỹ năng nói lên hệ thống.
- + Bài test General: Bài làm sẽ bao gồm 3 phần Listening, Reading và một bài Writing task 2. Giáo viên sẽ chấm bài này trên thang 9 điểm, rồi nhập điểm lên hệ thống.
- + Sau khi nhập điểm hoàn tất, hệ thống sẽ tự động tính điểm trung bình và gợi ý khóa học:  $(\text{điểm bài test General} + \text{điểm bài test kỹ năng Nói})/2$

Học viên có thể xem kết quả của mình qua tài khoản cá nhân. Dựa vào kết quả đánh giá, học viên sẽ được phân loại và tham gia vào lớp học phù hợp với trình độ. Các khóa học của trung tâm sẽ kéo dài 3 tháng (20 buổi học) và được chia thành như sau:

- + Band điểm 3.5: khóa nhận học viên có đầu vào  $< 3.5$ .
- + Band điểm 4.5: khóa nhận học viên có đầu vào  $> 3.5$  và  $< 4.5$ .
- + Band điểm 5.5: khóa nhận học viên có đầu vào  $> 4.5$  và  $< 5.5$ .
- + Band điểm 6.5: khóa nhận học viên có đầu vào  $> 5.5$  và  $< 6.5$ .
- + Học viên có trình độ trên 6.5: Không có khóa học phù hợp.

Khi tham gia lớp học tại trung tâm, học viên sẽ được điểm danh mỗi buổi học bởi giáo viên. Giáo viên sẽ cập nhật thông tin điểm danh của từng học viên

trong lên hệ thống. Nếu học viên có tỉ lệ đi học  $< 70\%$ , thì hệ thống sẽ đánh dấu học sinh trong diện bị cảnh báo

Trong quá trình học tại trung tâm, học viên sẽ cần làm 2 bài test Midterm và Final-term (Đánh giá trên thang 10). Giáo viên sẽ chấm điểm và nhập điểm 2 bài test này lên hệ thống. Điểm tổng kết của học viên sẽ được hệ thống tự động tính ( $30\%$  điểm Midterm +  $70\%$  điểm Final-Term), nếu điểm tổng kết của học viên  $\geq 7$  thì học viên sẽ được đánh giá là PASS khóa học, ngược lại học viên sẽ bị hủy cam kết đầu ra.

Sau mỗi khóa học, học viên sẽ được chuyển lên khóa học cao hơn theo lộ trình đã đăng ký ban đầu với trung tâm, cho đến khi hoàn thành lộ trình và đăng ký thi IELTS (nếu có nhu cầu). Điểm thi IELTS của học viên sau khi có kết quả sẽ được nhập vào hệ thống.

Nếu học viên hoàn thành khóa học hoặc bỏ ngang khóa học, trạng thái của học viên sẽ được xếp vào dạng “Inactive”, cùng với yêu cầu thanh toán toàn bộ số tiền còn đang nợ, giữ lại thông tin về trình độ để xếp học sinh vào lại lớp học nếu học sinh này có nhu cầu tiếp tục.

Học phí của học viên sẽ được hệ thống tính tự động sau khi đăng ký khóa học thành công và được hiển thị cho học viên về số tiền cũng như tình trạng học phí.

Trước khi tham gia vào trung tâm, các bạn học viên có thể được các bạn tư vấn viên tại trung tâm tư vấn về khóa học. Với mỗi học viên đăng ký học thành công, tư vấn viên có thể nhận thêm hoa hồng ( $10\%$  số tiền mỗi khóa học mà học viên đăng ký thành công). Từ đó hệ thống cũng sẽ tính tiền lương cho tư vấn viên (Tiền lương = tiền hoa hồng).

Cuối lộ trình, học viên sẽ làm một bài khảo sát để đánh giá giáo viên và trung tâm trên thang 3 điểm cùng với các nhận xét. Với các điểm số đánh giá trung bình, sẽ được đánh giá như sau:

- + Điểm đánh giá trung bình bằng 3 điểm thì giáo viên và trung tâm sẽ được đánh giá “Very Good” với bạn học viên đó
- + Điểm đánh giá trung bình hơn 2 điểm thì giáo viên và trung tâm sẽ được đánh giá “Good” với bạn học viên đó
- + Điểm đánh giá trung bình nhỏ hơn 2 điểm thì giáo viên và trung tâm sẽ được đánh giá “Bad” với bạn học viên đó

Từ đó làm cơ sở để trung tâm có thể cải thiện chất lượng dạy và học của mình.

Cách tổ chức trung tâm: vào tháng 1, 4, 7, 10 hàng năm sẽ mở các lớp mới cho học viên. Các lớp sẽ học 2 buổi / tuần, mỗi lớp giới hạn 20 học viên. Mỗi khóa học nhỏ sẽ kéo dài 3 tháng (20 buổi). Các lớp đều sẽ có lộ trình học lên đến band 6.5 IELTS, tuy nhiên học viên có thể hoàn thành lộ trình học hoặc dừng học bất cứ lúc nào theo lộ trình cá nhân đã đăng ký.

Giáo viên khi được tuyển vào trung tâm sẽ cung cấp các thông tin cá nhân (tên, tuổi, số điện thoại, bằng cấp). Giáo viên khi làm việc tại trung tâm sẽ có trách nhiệm giảng dạy các lớp được phân công, chấm điểm bài test đầu vào, chấm điểm hai bài test Midterm và bài test Final-term. Lương của giáo viên sẽ được tính như sau: Tiền lương tháng = 30% số tiền thu được từ các lớp giáo viên dạy trong tháng + 5\$ \* số bài test đầu vào giáo viên chấm trong tháng.

Cuối mỗi quý, năm sẽ có tổng kê doanh thu của cả trung tâm, nhằm mục đích đưa ra các chiến lược marketing tốt hơn trong tương lai. Ngoài ra hệ thống cũng sẽ tổng kê lại các đánh giá của học viên về các khóa học nhằm có thể cải thiện chất lượng dạy và học.

## **B. Các chức năng tiêu biểu**

### **1. Đối với người quản lý**

- Thêm giáo viên, học viên mới.
- Xem danh sách học viên, giáo viên.
- Tính điểm trung bình của học viên trong một lớp.
- Tạo danh sách khen thưởng cho học sinh có điểm đầu ra cao hơn điểm khóa học.
- Xem các lớp học trùng giờ để linh động xếp lớp học.
- Xác định học viên bị hủy cam kết đầu ra.

### **2. Đối với học viên**

- Học viên sẽ được gợi ý khóa học dựa trên điểm test đầu vào
- Kiểm tra học phí.
- Xem nội dung học trên lớp trong tuần và bài tập về nhà trong tuần đó của học viên.
- Theo dõi kết quả học tập, tình trạng đi học/ngỉ học của bản thân.

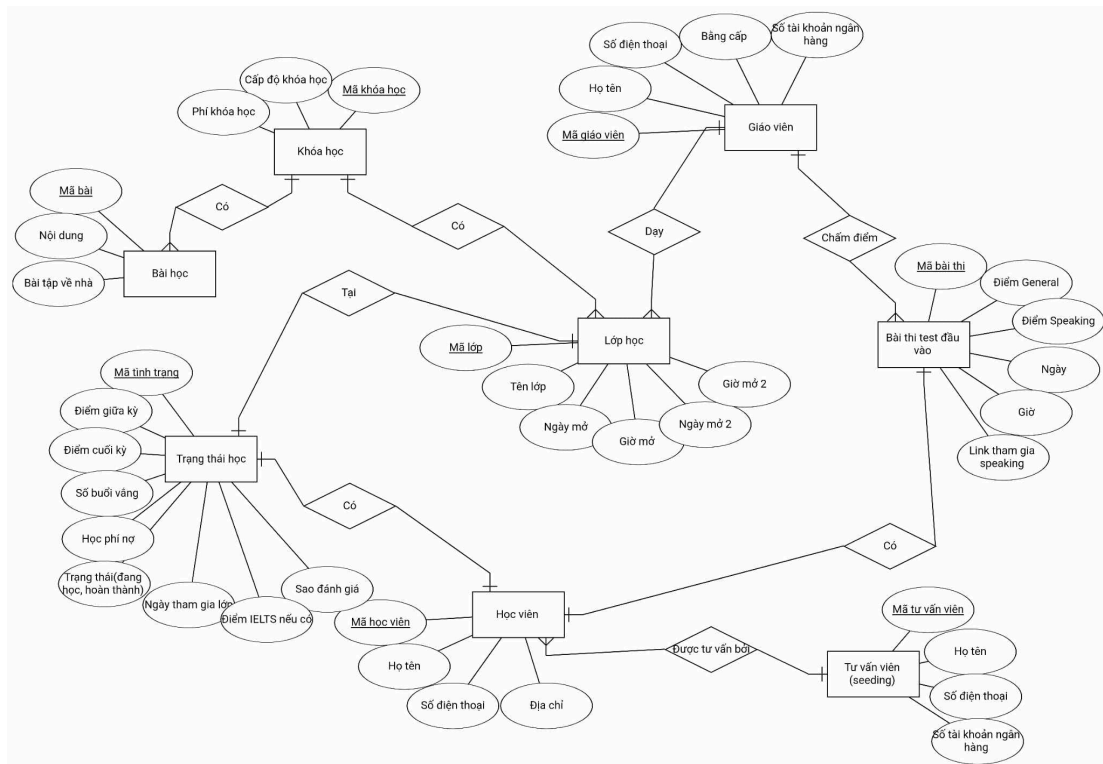
### 3. Đối với giáo viên

- Nhập điểm cho các bài test đầu vào, Midterm test và Final-term test.
- Cập nhật điểm danh cho lớp học.
- Xem lịch làm việc trong ngày.
- Tự động xếp bài test đầu vào cho học viên

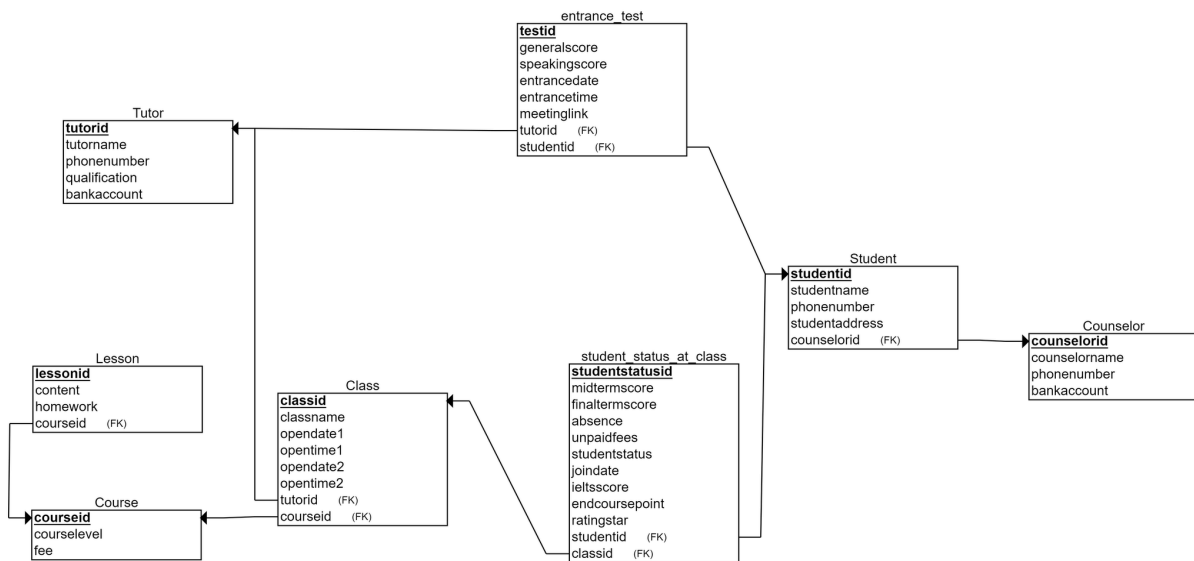
## II. Thiết kế cơ sở dữ liệu

### A. Sơ đồ thực thể - liên kết và danh sách các bảng

Sơ đồ thực thể - liên kết



## Danh sách các bảng



### B. Mô tả các bảng và các quan hệ của chúng

Cơ sở dữ liệu gồm 8 bảng: Counselor, Student, Tutor, Entrance\_Test, Class, Student\_status\_at\_class, lesson, course.

- Bảng Tutor: tutorid
- Bảng Lesson: lessonid, *courseid*
- Bảng Course: courseid
- Bảng Class: classid, *tutorid*, *courseid*  
(chú thích: *courseid* chỉ để mục đích lưu 20 buổi đầu của lớp học đó chứ KHÔNG lưu toàn bộ lesson mà lớp đó sẽ học)
- Bảng Entrance\_test: testid, *tutorid*, *studentid*
- Bảng Student\_status\_at\_class: studentstatusid, *studentid*, *classid*
- Bảng student: studentid, *counselorid*
- Bảng counselor: counselorid

Các liên kết giữa các bảng:

- Liên kết giữa course và lesson: 1-n
- Liên kết giữa student và student\_status\_at\_class: 1-1
- Liên kết giữa student\_status\_at\_class và class: 1-1
- Liên kết giữa course và class: 1-1
- Liên kết giữa tutor và class: 1-1
- Liên kết giữa tutor và entrance\_test: 1-n

- Liên kết giữa student và entrance\_test: 1-1
- Liên kết giữa counselor và student: 1-n

### III. Câu lệnh SQL

#### A. Các hàm truy vấn và tính toán:

1. Hàm tính doanh thu của một lớp trong trung tâm:

```
CREATE FUNCTION public.calculate_class_revenue(class_id text) RETURNS money
LANGUAGE plpgsql
AS $$
DECLARE
    revenue MONEY := 0;
    starting_point TEXT;
    student_record RECORD;
BEGIN
    starting_point := CASE
        WHEN class_id LIKE 'IE35%' THEN 'IE35'
        WHEN class_id LIKE 'IE45%' THEN 'IE45'
        WHEN class_id LIKE 'IE55%' THEN 'IE55'
        WHEN class_id LIKE 'IE65%' THEN 'IE65'
        ELSE NULL
    END;

    IF starting_point IS NULL THEN
        RETURN 0;
    END IF;

    FOR student_record IN (
        SELECT ssac.EndCoursePoint, c.fee
        FROM public.Student_Status_At_Class ssac
        JOIN public.Clazz cl ON ssac.ClassID = cl.ClassID
        JOIN public.Course c ON cl.CourseID = c.CourseID
        WHERE ssac.ClassID = calculate_class_revenue.class_id
    ) LOOP
        IF student_record.EndCoursePoint >= 3.5 AND starting_point = 'IE35' THEN
            revenue := revenue + student_record.fee;
        END IF;
        IF student_record.EndCoursePoint >= 4.5 AND starting_point IN ('IE35', 'IE45') THEN
            revenue := revenue + student_record.fee;
        END IF;
        IF student_record.EndCoursePoint >= 5.5 AND starting_point IN ('IE35', 'IE45', 'IE55') THEN
            revenue := revenue + student_record.fee;
        END IF;
        IF student_record.EndCoursePoint >= 6.5 AND starting_point IN ('IE35', 'IE45', 'IE55', 'IE65') THEN
            revenue := revenue + student_record.fee;
        END IF;
    END LOOP;

    RETURN revenue;
END;
$$;

select * from public.calculate_class_revenue('IE3500001')
```



- Kết quả câu truy vấn:

Query

Query History

1

`select * from public.calculate_class_revenue('IE3500001')`

Data Output

Messages

Notifications

<

- Cơ chế hoạt động:
  - + Đầu vào: ID lớp học. Nếu đầu vào không hợp lệ, hàm trả về 0.
  - + Lặp qua danh sách học sinh trong lớp trong bảng Student\_Status\_At\_Class, tính toán học phí cho từng học viên dựa vào điểm đầu vào và kết thúc khóa học và cộng tổng.

2. Hàm tính lương tháng giáo viên trong 1 tháng:

```

CREATE OR REPLACE FUNCTION public.calculate_tutor_salary2(tutor_id text, salary_month text)
RETURNS TABLE (
    entrance_test_salary MONEY,
    class_salary MONEY
)
LANGUAGE plpgsql
AS $$
DECLARE
    entrance_test_salary MONEY := 0;
    class_salary MONEY := 0;
    class_record RECORD;
    month_diff INT;
    payment_months INT;
BEGIN
    SELECT COUNT(*) * 5::money
    INTO entrance_test_salary
    FROM public.Entrance_Test
    WHERE TutorID = calculate_tutor_salary2.tutor_id
    AND to_char(EntranceDate, 'YYYY-MM') = calculate_tutor_salary2.salary_month;

    FOR class_record IN (
        SELECT c1.ClassID, c1.OpenDate1, c1.OpenDate2, c.fee
        FROM public.Clazz c1
        JOIN public.Course c ON c1.CourseID = c.CourseID
        WHERE c1.TutorID = calculate_tutor_salary2.tutor_id
    ) LOOP
        CASE
            WHEN class_record.ClassID LIKE 'IE35%' THEN
                payment_months := 9;
            WHEN class_record.ClassID LIKE 'IE45%' THEN
                payment_months := 7;
            WHEN class_record.ClassID LIKE 'IE55%' THEN
                payment_months := 5;
            WHEN class_record.ClassID LIKE 'IE65%' THEN
                payment_months := 3;
            ELSE
                payment_months := 0;
        END CASE;

        IF payment_months > 0 THEN
            month_diff := (EXTRACT(YEAR FROM to_date(salary_month || '-01', 'YYYY-MM-DD')) - EXTRACT(YEAR FROM class_record.OpenDate1)) * 12 +
                EXTRACT(MONTH FROM to_date(salary_month || '-01', 'YYYY-MM-DD')) - EXTRACT(MONTH FROM class_record.OpenDate1);

            IF month_diff >= 0 AND month_diff < payment_months THEN
                class_salary := class_salary +
                    (public.calculate_class_revenue(class_record.ClassID) * 0.3) / payment_months;
            END IF;
        END IF;
    END LOOP;

    RETURN QUERY SELECT entrance_test_salary, class_salary;
END;
$$;

select * from public.calculate_tutor_salary2('TU0001', '2023-01')

```

Kết quả câu truy vấn:

54

55

```
select * from public.calculate_tutor_salary2('TU0001', '2023-01')
```

Data Output

Messages

Notifications

	entrance_test_salary money	class_salary money
1	\$5.00	\$2,366.66

Cơ chế hoạt động:

- + Đầu vào: TutorID, tháng muốn tính lương giáo viên.
- + Tính lương bài test đầu vào: tính số lượng bài test đầu vào giáo viên chấm trong tháng đó và nhân với \$5.
- + Tính lương theo lớp: tính doanh thu lớp học, chia cho thời gian để lấy doanh thu theo tháng, sau đó lấy 30% doanh thu chia cho giáo viên hàng tháng.

### 3. Hàm tính lương cho tư vấn viên:

```
CREATE FUNCTION public.calculate_counselor_wage(counselor_id text) RETURNS money
LANGUAGE plpgsql
AS $$
DECLARE
    total_tuition money := 0;
    starting_point text;
    student_record RECORD;
BEGIN
    FOR student_record IN (
        SELECT ssac.EndCoursePoint, c.fee, cl.CourseID
        FROM public.Student_Status_At_Class ssac
        JOIN public.Student s ON ssac.StudentID = s.StudentID
        JOIN public.Clazz cl ON ssac.ClassID = cl.ClassID
        JOIN public.Course c ON cl.CourseID = c.CourseID
        WHERE s.CounselorID = calculate_counselor_wage.counselor_id
    ) LOOP
        starting_point := CASE
            WHEN student_record.CourseID = 'IE35' THEN 'IE35'
            WHEN student_record.CourseID = 'IE45' THEN 'IE45'
            WHEN student_record.CourseID = 'IE55' THEN 'IE55'
            WHEN student_record.CourseID = 'IE65' THEN 'IE65'
            ELSE NULL
        END;

        IF student_record.EndCoursePoint >= 3.5 AND starting_point = 'IE35' THEN
            total_tuition := total_tuition + student_record.fee;
        END IF;
        IF student_record.EndCoursePoint >= 4.5 AND starting_point IN ('IE35', 'IE45') THEN
            total_tuition := total_tuition + student_record.fee;
        END IF;
        IF student_record.EndCoursePoint >= 5.5 AND starting_point IN ('IE35', 'IE45', 'IE55') THEN
            total_tuition := total_tuition + student_record.fee;
        END IF;
        IF student_record.EndCoursePoint >= 6.5 AND starting_point IN ('IE35', 'IE45', 'IE55', 'IE65') THEN
            total_tuition := total_tuition + student_record.fee;
        END IF;
    END LOOP;

    RETURN total_tuition * 0.1;
END;
$$;
```

Cơ chế hoạt động:

- Đầu vào: CounselorID.
- Đầu tiên hàm thực hiện phép kết nối để lấy ra học phí của từng học viên được tư vấn viên tư vấn sau đó thực hiện tính toán dựa trên điểm đầu ra học viên.
- Tư vấn viên sẽ được ăn hoa hồng 10% dựa trên tổng doanh thu từ các học viên mà người này tư vấn.

Kết quả câu truy vấn:

43	
44	<b>SELECT</b> public.calculate_counselor_wage('CS0009');
Data Output Messages Notifications	
	calculate_counselor_wage money
1	\$6,400.00

4. Hàm tính tỉ lệ đi học của học viên trong toàn bộ khóa học:

```
CREATE OR REPLACE FUNCTION public.calculate_student_absence_rate(student_id text) RETURNS text
    LANGUAGE plpgsql
    AS $$
DECLARE
    absence_count integer;
    total_lessons integer;
    class_id text;
    absence_rate real;
BEGIN
    SELECT ssac.Absence, ssac.ClassID
    INTO absence_count, class_id
    FROM public.Student_Status_At_Class ssac
    WHERE ssac.StudentID = calculate_student_absence_rate.student_id;

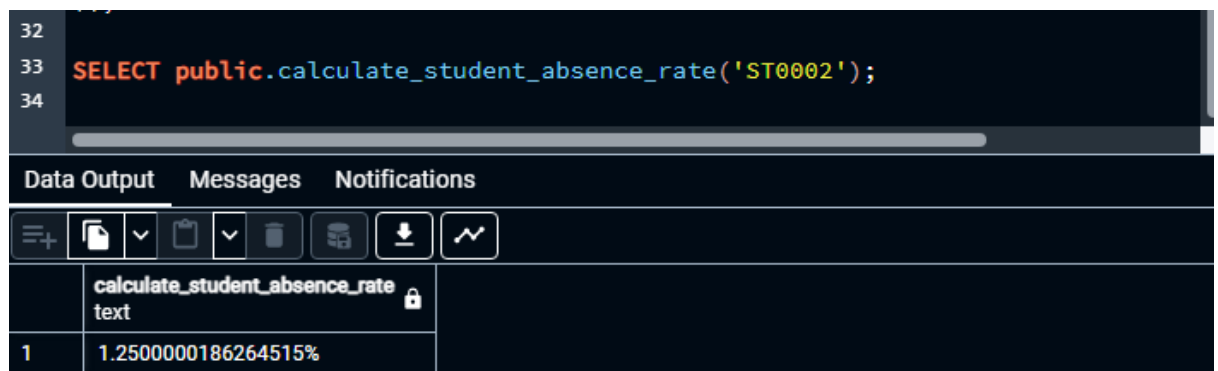
    total_lessons := CASE
        WHEN class_id LIKE 'IE35%' THEN 80
        WHEN class_id LIKE 'IE45%' THEN 60
        WHEN class_id LIKE 'IE55%' THEN 40
        WHEN class_id LIKE 'IE65%' THEN 20
        ELSE NULL
    END;

    IF total_lessons > 0 THEN
        absence_rate := absence_count::real / total_lessons;
    ELSE
        absence_rate := 0;
    END IF;

    RETURN (absence_rate * 100)::text || '%';
END;
$$;

SELECT public.calculate_student_absence_rate('ST0002');
```

Kết quả:



The screenshot shows a database IDE interface. At the top, a SQL query is entered in a text editor: `SELECT public.calculate_student_absence_rate('ST0002');`. Below the editor, there are tabs for "Data Output", "Messages", and "Notifications". The "Data Output" tab is active, displaying a table with the results of the query. The table has two columns: the function name and its return type, and a single row with the calculated absence rate for student 'ST0002'.

	calculate_student_absence_rate text
1	1.2500000186264515%

Cơ chế hoạt động:

- Đầu vào: Student\_ID.

- Tính toán tỉ lệ nghỉ học bằng cách lấy số buổi nghỉ chia cho tổng số buổi học mà học viên đăng ký.

## 5. Hàm tính rating star trung bình của một lớp.

```

1 CREATE OR REPLACE FUNCTION calculate_average_rating(class_id VARCHAR)
2 RETURNS NUMERIC AS $$
3 DECLARE
4     average_rating NUMERIC;
5 BEGIN
6     -- Tính rating star trung bình của một lớp
7     SELECT AVG(ratingstar) INTO average_rating
8     FROM student_status_at_class
9     WHERE classid = class_id;
10
11     RETURN COALESCE(average_rating, 0);
12 END;
13 $$ LANGUAGE plpgsql;

```

- Giải thích:
  - + Hàm calculate\_average\_rating sẽ nhận đầu vào là classid của một lớp và tính trung bình ratingstar của lớp đó bằng hàm AVG.
- Kết quả:

1	SELECT calculate_average_rating('IE3500030')
Data Output Messages Notifications	
<div> <div>calculate_average_rating</div> <div>numeric</div> </div>	
1	2.2500000000000000

## 6. Hàm hiển thị danh sách các bài test đầu vào mà giáo viên chấm trong ngày.

```

1  -- Hàm hiển thị danh sách entrancetest trong ngày của giáo viên
2  CREATE OR REPLACE FUNCTION list_entrance_tests(tutor_id VARCHAR, year INT, month INT, day INT)
3  RETURNS TABLE(test_id VARCHAR, general_score REAL, speaking_score REAL, entrance_time DATE) AS $$
4  BEGIN
5      RETURN QUERY|
6      SELECT testid, generalscore, speakingscore, entrancedate
7      FROM entrance_test
8      WHERE tutorid = tutor_id
9          AND EXTRACT(YEAR FROM entrancedate) = year
10         AND EXTRACT(MONTH FROM entrancedate) = month
11         AND EXTRACT(DAY FROM entrancedate) = day;
12 END;
13 $$ LANGUAGE plpgsql;

```

- Giải thích:
  - + Hàm list\_entrance\_tests nhận đầu vào là tutorid, year, month và day và trả về một bảng gồm các bài test đầu vào mà giáo viên cần chấm trong ngày hôm đó.
  - + Ta sử dụng hàm EXTRACT để có thể lọc ra ngày/tháng/năm của trường entrancedate làm điều kiện so sánh.
- Kết quả:

1 **SELECT** \* **FROM** list\_entrance\_tests('TU0272', 2022, 6, 3)

Data Output Messages Notifications

	test_id character varying	general_score real	speaking_score real	entrance_time date
1	TE0001	1.5	1.5	2022-06-03

7. Hàm hiển thị danh sách học viên của một lớp cùng với điểm giữa kỳ, cuối kỳ và điểm tổng kết của một lớp:

```

CREATE OR REPLACE FUNCTION public.get_student_scores(score_type VARCHAR(255), class_id VARCHAR(255))
RETURNS TABLE (
    StudentID VARCHAR(255),
    MidtermScore REAL,
    FinaltermScore REAL,
    TotalScore REAL
)
AS $$
BEGIN
    IF score_type = 'List' THEN
        RETURN QUERY
        SELECT
            s.StudentID,
            s.MidTermScore::real,
            s.FinalTermScore::real,
            ((s.MidTermScore * 0.3 + s.FinalTermScore * 0.7))::real AS TotalScore
        FROM public.Student_Status_At_Class s
        WHERE s.ClassID = get_student_scores.class_id;

    ELSIF score_type = 'Average' THEN
        RETURN QUERY
        SELECT
            'Class Average' AS StudentID,
            (AVG(s.MidTermScore))::real AS MidtermScore,
            (AVG(s.FinalTermScore))::real AS FinaltermScore,
            (AVG(s.MidTermScore * 0.3 + s.FinalTermScore * 0.7))::real AS TotalScore
        FROM public.Student_Status_At_Class s
        WHERE s.ClassID = get_student_scores.class_id;

    ELSE
        RAISE EXCEPTION 'Invalid score_type parameter. Use "List" or "Average".';
    END IF;
END;
$$ LANGUAGE plpgsql;

-- Example usage:
SELECT * FROM public.get_student_scores('List', 'IE3500001');
-- SELECT * FROM public.get_student_scores('Average', 'IE3500001');

```

Kết quả:

```

41
42 SELECT * FROM public.get_student_scores('List', 'IE3500001');
43 -- SELECT * FROM public.get_student_scores('Average', 'IE3500001');

```

	studentid character varying	midtermscore real	finaltermscore real	totalscore real
1	ST0001	10	6	7.2
2	ST0002	9.5	7	7.75
3	ST0003	8	7.5	7.65
4	ST0004	8.5	8	8.15
5	ST0005	9	8	8.3
6	ST0006	9	5	6.2
7	ST0007	9	8	8.3
8	ST0008	5.5	9.5	8.3
9	ST0009	7	9	8.4

Đầu vào:

1. score\_type: 'List' hoặc 'Average'
2. class\_id: ID lớp học.

Đầu ra:

- + Một bảng chứa ID học sinh, điểm giữa kỳ, điểm cuối kỳ và điểm tổng kết.
- \* Nếu `score\_type` là 'List', bảng sẽ chứa điểm của từng học sinh trong lớp.
- \* Nếu `score\_type` là 'Average', bảng sẽ chứa điểm trung bình giữa kỳ, cuối kỳ và tổng kết cho toàn bộ lớp học.

Giải thích:

- + Hàm sử dụng câu lệnh where để ghép nối tham số class\_id đầu vào với Student\_Status\_At\_Class, và dùng class\_id ghép nối với bảng học viên.



8. Hàm trả về 100 học sinh tiêu biểu để tuyên dương ở trung tâm:

```
CREATE OR REPLACE FUNCTION public.good_students_for_reward()
RETURNS TABLE (
    StudentID VARCHAR(255),
    StudentName VARCHAR(255),
    PhoneNumber VARCHAR(255),
    StudentAddress VARCHAR(255),
    CounselorID VARCHAR(255),
    ClassID VARCHAR(255),
    ClassName VARCHAR(255),
    IELTSScore REAL,
    EndCoursePoint REAL,
    Difference REAL
) AS $$
BEGIN
    RETURN QUERY
    SELECT
        ssac.StudentID,
        s.StudentName,
        s.PhoneNumber,
        s.StudentAddress,
        s.CounselorID,
        ssac.ClassID,
        c.ClassName,
        ssac.IELTSScore,
        ssac.EndCoursePoint,
        ssac.IELTSScore - ssac.EndCoursePoint AS Difference
    FROM public.Student_Status_At_Class ssac
    JOIN public.Student s ON ssac.StudentID = s.StudentID
    JOIN public.clazz c ON ssac.ClassID = c.ClassID
    WHERE ssac.IELTSScore IS NOT NULL
    ORDER BY ssac.IELTSScore DESC, ssac.IELTSScore - ssac.EndCoursePoint DESC
    LIMIT 100;
END;
$$ LANGUAGE plpgsql;

SELECT * FROM public.good_students_for_reward();
```

Query

Query History

```

25      ssac.EndCoursePoint,
26      ssac.IELTSScore - ssac.EndCoursePoint AS Difference
27  FROM public.Student_Status_At_Class ssac
28  JOIN public.Student s ON ssac.StudentID = s.StudentID
29  JOIN public.clazz c ON ssac.ClassID = c.ClassID
30  WHERE ssac.IELTSScore IS NOT NULL
31  ORDER BY ssac.IELTSScore DESC, ssac.IELTSScore - ssac.EndCoursePoint DESC
32  LIMIT 100;
33  END;
34  $$ LANGUAGE plpgsql;
35
36  SELECT * FROM public.good_students_for_reward();

```

Data Output

Messages

Notifications

+

📄

▼

📁

▼

🗑️

🔄

⬇️

📈

	ounselorid haracter varying 🔒	classid character varying 🔒	classname character varying 🔒	ieltsscore real 🔒	endcoursepoint real 🔒	difference real 🔒
1	0S0160	IE4500031	IELTS 4.5 Class 31	8	6.5	1.5
2	0S0196	IE3500076	IELTS 3.5 Class 76	8	6.5	1.5
3	0S0110	IE6500021	IELTS 6.5 Class 21	8	6.5	1.5

Total rows: 100 of 100

Query complete 00:00:00.154

Ln 36, Col 49

- Đầu vào: không nhận đầu vào.
- Đầu ra: trả về các cột thông tin của học viên tiêu biểu, quan trọng nhất là điểm IELTS cao và sự chênh lệch giữa điểm IELTS và điểm đầu ra khóa học là lớn nhất (học viên học vượt năng lực so với khóa học đào tạo - thể hiện chất lượng khóa học).
- Cơ chế:
  1. Hàm ghép nối 3 bảng Student\_Status\_At\_Class (ssac), Student (s) và clazz (c) để tìm kiếm thông tin về sinh viên.
  2. Lọc các sinh viên có điểm IELTSScore không NULL, sắp xếp theo điểm IELTS giảm dần, theo độ chênh lệch giữa IELTSScore và điểm đầu ra khóa học giảm dần.
  3. Hàm trả về tối đa 100 sinh viên tiêu biểu - điểm IELTS cao + học vượt năng lực

9. Hiển thị nội dung bài học, bài tập và ngày học - giờ học của một học viên:

```

CREATE OR REPLACE FUNCTION public.get_student_lesson_schedule10(student_id character varying)
RETURNS TABLE(
    student_name character varying(255),
    class_id character varying(255),
    lesson_unit character varying(255),
    lesson_homework character varying(255),
    lesson_date date,
    day_of_week text,
    date_time text
)
LANGUAGE plpgsql
AS $$
DECLARE
    opendate1 DATE;
    opendate2 DATE;
    opentime1 TIME;
    opentime2 TIME;
    endcoursepoint REAL;
    class_id VARCHAR(255);
    student_name VARCHAR(255);
    start_course VARCHAR(255);
    end_course VARCHAR(255);

```

```

BEGIN
    SELECT
        c.opendate1::DATE, c.opendate2::DATE, c.opentime1, c.opentime2, ss.endcoursepoint, ss.classid, s.studentname,
        CASE WHEN ss.classid LIKE 'IE35%' THEN 'IE35'
             WHEN ss.classid LIKE 'IE45%' THEN 'IE45'
             WHEN ss.classid LIKE 'IE55%' THEN 'IE55'
             WHEN ss.classid LIKE 'IE65%' THEN 'IE65'
             ELSE NULL END AS start_course,
        CASE WHEN ss.endcoursepoint = 3.5 THEN 'IE35'
             WHEN ss.endcoursepoint = 4.5 THEN 'IE45'
             WHEN ss.endcoursepoint = 5.5 THEN 'IE55'
             WHEN ss.endcoursepoint = 6.5 THEN 'IE65'
             ELSE NULL END AS end_course
    INTO opendate1, opendate2, opentime1, opentime2, endcoursepoint, class_id, student_name, start_course, end_course
    FROM public.Student_Status_At_Class ss
    JOIN public.Clazz c ON ss.ClassID = c.ClassID
    JOIN public.Student s ON ss.StudentID = s.StudentID
    WHERE ss.StudentID = student_id;

```

```

RETURN QUERY
SELECT
  student_name,
  class_id,
  l.content AS lesson_unit,
  l.homework AS lesson_homework,
  CASE
    WHEN MOD(CAST(SUBSTRING(l.lessonid, 5, 2) AS INT), 2) = 0 THEN (opendate2 + (CAST(SUBSTRING(l.lessonid, 5, 2) AS INT) - 1) / 2 * INTERVAL '7 days')::DATE
    ELSE (opendate1 + (CAST(SUBSTRING(l.lessonid, 5, 2) AS INT) - 1) / 2 * INTERVAL '7 days')::DATE
  END AS lesson_date,
  to_char(CASE
    WHEN MOD(CAST(SUBSTRING(l.lessonid, 5, 2) AS INT), 2) = 0 THEN (opendate2 + (CAST(SUBSTRING(l.lessonid, 5, 2) AS INT) - 1) / 2 * INTERVAL '7 days')::DATE
    ELSE (opendate1 + (CAST(SUBSTRING(l.lessonid, 5, 2) AS INT) - 1) / 2 * INTERVAL '7 days')::DATE
  END, 'Day') AS day_of_week,
  -- Combine date and time with commas
  to_char(CASE
    WHEN MOD(CAST(SUBSTRING(l.lessonid, 5, 2) AS INT), 2) = 0 THEN (opendate2 + (CAST(SUBSTRING(l.lessonid, 5, 2) AS INT) - 1) / 2 * INTERVAL '7 days')::DATE
    ELSE (opendate1 + (CAST(SUBSTRING(l.lessonid, 5, 2) AS INT) - 1) / 2 * INTERVAL '7 days')::DATE
  END, 'Day') || ', ' ||
  to_char(CASE
    WHEN MOD(CAST(SUBSTRING(l.lessonid, 5, 2) AS INT), 2) = 0 THEN opendate2 + (CAST(SUBSTRING(l.lessonid, 5, 2) AS INT) - 1) / 2 * INTERVAL '7 days'
    ELSE opendate1 + (CAST(SUBSTRING(l.lessonid, 5, 2) AS INT) - 1) / 2 * INTERVAL '7 days'
  END, 'YYYY-MM-DD') || ', ' ||
  to_char(CASE
    -- Corrected the logic here to alternate between OpenTime1 and OpenTime2
    WHEN CAST(SUBSTRING(l.lessonid, 5, 2) AS INT) % 2 = 1 THEN opentime1
    ELSE opentime2
  END, 'HH24:MI:SS') AS date_time
FROM public.lesson l
WHERE l.courseid IN (
  SELECT unnested_course_id
  FROM UNNEST(ARRAY[start_course, 'IE45', 'IE55', 'IE65']) AS unnested_course_id
  WHERE unnested_course_id IS NOT NULL
  AND unnested_course_id <= end_course
)
ORDER BY l.lessonid;
END;
$$;

```

Kết quả:

Query Query History

```
65         WHEN class_record.endcoursepoint = 4.5 THEN 'IE45'
66         WHEN class_record.endcoursepoint = 5.5 THEN 'IE55'
67         WHEN class_record.endcoursepoint = 6.5 THEN 'IE65'
68         ELSE NULL
69     END
70     ORDER BY l.lessonid;
71 END;
72 $$;
73
74 -- explain analyse
75 select * from public.get_student_lesson_schedule10('ST0001')
```

Data Output Messages Notifications

<

Cơ chế:

- a. Lấy thông tin lớp học và học viên từ các bảng Student\_Status\_At\_Class, Clazz và Student dựa trên student\_id đầu vào.
- b. Tính toán lịch học và thông tin bài học dựa trên ghép nối khóa học của từng học viên với studentid đầu vào, sau đó tính toán lịch học của các buổi học sau dựa theo ngày bắt đầu 1 và ngày bắt đầu 2 (ví dụ buổi số 3 sau 7 ngày tính từ ngày bắt đầu 1, buổi số 4 bắt đầu từ 7 ngày sau ngày bắt đầu 2).
- c. Lấy thông tin bài học từ bảng Lesson dựa trên courseid tương ứng với lớp học và mức độ của học viên (endcoursepoint).
- d. Trả về bảng kết quả:
  - + student\_name: Tên học viên
  - + class\_id: Mã lớp học
  - + lesson\_unit: Nội dung bài học
  - + lesson\_homework: Bài tập về nhà
  - + lesson\_date: Ngày học bài này
  - + day\_of\_week: Ngày trong tuần của ngày học bài
  - + date\_time: Ngày, giờ và thời gian của buổi học (kết hợp ngày, giờ và thời gian)

10. Hàm để thêm thông tin học viên khi học viên đến trung tâm và tạo bài test đầu vào cho học viên đó:

```
CREATE OR REPLACE FUNCTION public.add_student_and_entrance_test(  
    student_name character varying,  
    phone_number character varying,  
    student_address character varying,  
    counselor_id character varying,  
    entrance_date date,  
    entrance_time time without time zone  
) RETURNS void  
LANGUAGE plpgsql  
AS $$  
DECLARE  
    new_student_id character varying;  
    new_test_id character varying;  
    meeting_link character varying;  
    is_duplicate boolean;  
BEGIN  
    SELECT 'ST' || LPAD((COALESCE(MAX(SUBSTRING(StudentID, 3)::INT), 0) + 1)::TEXT, 4, '0')  
    INTO new_student_id  
    FROM public.Student;  
  
    INSERT INTO public.Student (StudentID, StudentName, PhoneNumber, StudentAddress, CounselorID)  
    VALUES (new_student_id, student_name, phone_number, student_address, counselor_id);  
  
    new_test_id := 'TE' || SUBSTRING(new_student_id, 3);
```

```

LOOP
meeting_link := 'meet.google.com/' || (
CASE
WHEN random() < 0.5 THEN 'abc'
ELSE 'def'
END
) || (random() * 1000000)::INT::VARCHAR || '-' || (random() * 1000000)::INT::VARCHAR;

SELECT EXISTS(SELECT 1 FROM public.Entrance_Test WHERE MeetingLink = meeting_link) INTO is_duplicate;
IF NOT is_duplicate THEN
EXIT;
END IF;
END LOOP;

INSERT INTO public.Entrance_Test (TestID, GeneralScore, SpeakingScore, EntranceDate, EntranceTime, MeetingLink, TutorID, StudentID)
VALUES (new_test_id, NULL, NULL, entrance_date, entrance_time, meeting_link, NULL, new_student_id);
END;
$$;

```

Cơ chế:

- Đầu vào: Các thông tin của học sinh và bài test đầu vào của học sinh đó.
- Hàm sẽ tự sinh ra một link meeting mới và check xem link đó có trùng hay không, sau đó thêm các thông tin vào bảng học viên và bảng bài test đầu vào.

11. Hàm để cập nhật điểm bài test đầu vào sau khi giáo viên đã chấm xong:

```

CREATE OR REPLACE FUNCTION public.update_entrance_test_score(
tutor_id VARCHAR(255),
student_id VARCHAR(255),
general_score REAL,
speaking_score REAL
) RETURNS VOID AS $$
BEGIN
-- Update the student's score in Entrance_Test table
UPDATE public.Entrance_Test
SET GeneralScore = general_score,
    SpeakingScore = speaking_score,
    TutorID = tutor_id -- Assuming the tutor is also being updated
WHERE StudentID = student_id;
END;
$$ LANGUAGE plpgsql;

SELECT public.update_entrance_test_score('TU0023', 'ST0005', 6.0, 5.5);

```

Query
Query History

```

1 select speakingscore, generalscore from public.entrance_test
2 where studentid = 'ST0005'
3
4 -- SELECT public.update_entrance_test_score('TU0023', 'ST0005', 6.0, 5.5);

```

Data Output
Messages
Notifications

+

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

	speakingscore real	generalscore real
1	5.5	6

- Cơ chế: update trên bảng Entrance\_Test

## 12. Hàm để tìm các lớp học trùng giờ:

```

CREATE OR REPLACE FUNCTION public.find_overlapping_classes4(
    input_date DATE,
    class_time TIME WITHOUT TIME ZONE
)
RETURNS TABLE (
    ClassID VARCHAR(255)
)
AS $$
BEGIN
    RETURN QUERY
    SELECT DISTINCT c1.ClassID
    FROM public.Clazz AS c1
    JOIN public.Clazz AS c2 ON c1.ClassID <> c2.ClassID
        AND c1.OpenDate1 <= c2.OpenDate2 AND c1.OpenDate2 >= c2.OpenDate1
        AND c1.OpenTime1 <= c2.OpenTime2 AND c1.OpenTime2 >= c2.OpenTime1
    WHERE
        input_date BETWEEN c1.OpenDate1 AND c1.OpenDate2
        AND class_time BETWEEN c1.OpenTime1 AND c1.OpenTime2
        AND class_time BETWEEN c2.OpenTime1 AND c2.OpenTime2;
END;
$$ LANGUAGE plpgsql;

select * from public.find_overlapping_classes4('2022-07-04', '14:00:00');

```

```

Query Query History
12 FROM public.Clazz AS c1
13 JOIN public.Clazz AS c2 ON c1.ClassID <> c2.ClassID
14 AND c1.OpenDate1 <= c2.OpenDate2 AND c1.OpenDate2 >= c2.OpenDate1
15 AND c1.OpenTime1 <= c2.OpenTime2 AND c1.OpenTime2 >= c2.OpenTime1
16 WHERE
17 input_date BETWEEN c1.OpenDate1 AND c1.OpenDate2
18 AND class_time BETWEEN c1.OpenTime1 AND c1.OpenTime2
19 AND class_time BETWEEN c2.OpenTime1 AND c2.OpenTime2;
20 END;
21 $$ LANGUAGE plpgsql;
22
23 select * from public.find_overlapping_classes4('2022-07-04', '14:00:00');

Data Output Messages Notifications
classid
character varying
3 IE3500009
4 IE5500006
5 IE5500009
6 IE6500009
Total rows: 6 of 6 Query complete 00:00:00.112 Ln 22, Col 1

```

Cơ chế hoạt động:

- Đầu vào: ngày và giờ muốn tìm lớp học trùng.
- Function sẽ so sánh ngày - giờ mở lớp và đưa ra danh sách các lớp trùng giờ mở  $\Leftrightarrow$  trùng thời gian.

13. Function tìm ra danh sách học viên nghỉ học quá nhiều - có điểm tổng kết dưới 7 - hủy cam kết đầu ra:

```

CREATE OR REPLACE FUNCTION public.identify_at_risk_students2()
RETURNS TABLE (
    Student_ID VARCHAR(255),
    Problem TEXT
)
AS $$
BEGIN
    UPDATE public.Student_Status_At_Class
    SET StudentStatus = CASE
        WHEN (0.3 * MidTermScore + 0.7 * FinalTermScore) < 7 AND Absence >= 4 THEN 'Active* - Low score and high absence'
        WHEN (0.3 * MidTermScore + 0.7 * FinalTermScore) < 7 THEN 'Active* - Low score'
        WHEN Absence >= 4 THEN 'Active* - High absence'
        ELSE StudentStatus
    END
    WHERE StudentStatus = 'Active'
    AND ((0.3 * MidTermScore + 0.7 * FinalTermScore) < 7 OR Absence >= 4);

```



```

RETURN QUERY
SELECT StudentID,
CASE
WHEN (0.3 * MidTermScore + 0.7 * FinalTermScore) < 7 AND Absence >= 4 THEN 'Low average score and high absence count'
WHEN (0.3 * MidTermScore + 0.7 * FinalTermScore) < 7 THEN 'Low average score'
WHEN Absence >= 4 THEN 'High absence count'
ELSE NULL
END AS Problem
FROM public.Student_Status_At_Class
WHERE StudentStatus LIKE 'Active*';
END;
$$ LANGUAGE plpgsql;

```

Query Query History

```

1 CREATE OR REPLACE FUNCTION public.identify_at_risk_students2()
2 RETURNS TABLE (
3     Student_ID VARCHAR(255),
4     Problem TEXT
5 )
6 AS $$
7 BEGIN
8     UPDATE public.Student_Status_At_Class
9     SET StudentStatus = CASE |
10         WHEN (0.3 * MidTermScore + 0.7 * FinalTermScore) < 7 AND
11         WHEN (0.3 * MidTermScore + 0.7 * FinalTermScore) < 7 TH
12         WHEN Absence >= 4 THEN 'Active* - High absence'

```

Data Output Messages Notifications

	student_id character varying	problem text
1	ST4227	High absence count
2	ST4239	High absence count
3	ST4312	High absence count
4	ST4317	High absence count

Total rows: 115 of 115 Query complete 00:00:00.077 Ln 9, Col 28

Cơ chế:

- Đầu vào: không có đầu vào.
- Cơ chế:
  - + Quét qua thông số buổi nghỉ học và điểm tổng của học viên để tìm ra những học viên bị hủy cam kết đầu ra.
  - + Cập nhật trạng thái của các học viên còn đang học thành Active\* nếu:
    - Active\* - Low score and high absence: nghỉ học nhiều - điểm thấp.
    - Active\* - Low score: điểm trung bình thấp
    - Active\* - High absence: nghỉ học nhiều.
  - + Những học viên thuộc diện này sẽ bị hủy cam kết đầu ra.

14. Đưa ra khuyến nghị khóa học cho học viên dựa theo kết quả test đầu vào:

```
CREATE OR REPLACE FUNCTION public.recommend_course(student_id VARCHAR(255))
RETURNS TEXT AS $$
DECLARE
    avg_score REAL;
    course_level TEXT;
BEGIN
    SELECT (GeneralScore + SpeakingScore) / 2
    INTO avg_score
    FROM PUBLIC.Entrance_Test
    WHERE StudentID = student_id;

    CASE
        WHEN avg_score < 3.5 THEN course_level := 'IE35';
        WHEN avg_score >= 3.5 AND avg_score < 4.5 THEN course_level := 'IE45';
        WHEN avg_score >= 4.5 AND avg_score < 5.5 THEN course_level := 'IE55';
        WHEN avg_score >= 5.5 THEN course_level := 'IE65';
        ELSE course_level := 'Not enough data to recommend';
    END CASE;

    RETURN course_level;
END;
$$ LANGUAGE plpgsql;

select public.recommend_course('ST0100')
```

Query Query History

```
15 WHEN avg_score < 3.5 THEN course_level := 'IE35';
16 WHEN avg_score >= 3.5 AND avg_score < 4.5 THEN course_level := 'IE45';
17 WHEN avg_score >= 4.5 AND avg_score < 5.5 THEN course_level := 'IE55';
18 WHEN avg_score >= 5.5 THEN course_level := 'IE65';
19 ELSE course_level := 'Not enough data to recommend';
20 END CASE;
21
22 RETURN course_level;
23 END;
24 $$ LANGUAGE plpgsql;
25
26 select public.recommend_course('ST0100')
```

Data Output Messages Notifications

≡+

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

	recommend_course	🔒
1	IE65	

- Đầu vào: mã học viên.
- Cơ chế: tính trung bình điểm Speaking và General đầu vào, sau đó đưa ra khuyến nghị khóa học gần nhất với điểm trung bình này của học viên.

## 15. Hàm hiển thị bài test đầu vào khi biết mã học sinh

```
CREATE OR REPLACE FUNCTION get_student_entrance_test_result(student_id VARCHAR(255))
RETURNS TABLE (
    "Test_ID" VARCHAR(255),
    "General Score" REAL,
    "Speaking Score" REAL,
    "Entrance Date" DATE,
    "Entrance Time" TIME
) AS $$
BEGIN
    RETURN QUERY
        SELECT Entrance_Test.TestID, Entrance_Test.GeneralScore, Entrance_Test.SpeakingScore,
            Entrance_Test.EntranceDate, Entrance_Test.EntranceTime
        FROM Entrance_Test
        WHERE Entrance_Test.StudentID = student_id;
END; $$

LANGUAGE 'plpgsql';
```

### Cơ chế:

- Tìm thông tin về bài test đầu vào trong bảng Entrance\_Test dựa vào mã học sinh trong bảng.

Query

Query History

1

`select` get\_student\_entrance\_test\_result('ST0005')


2

Data Output


Messages

Notifications


≡+





▼




▼










get\_student\_entrance\_test\_result

record



1

(TE0005,1.5,1.5,2022-06-12,08:00:00)

16. Liệt kê các học sinh có điểm test đầu vào > 6.5(không có lớp học phù hợp tại trung tâm):

```
SELECT StudentID, (GeneralScore + SpeakingScore) / 2 AS AverageScore
FROM Entrance_Test
WHERE (GeneralScore + SpeakingScore) / 2 > 6.5;
```

Query		Query History
1	<b>SELECT</b> StudentID, (GeneralScore + SpeakingScore) / 2 <b>AS</b> AverageScore	
2	<b>FROM</b> Entrance_Test	
3	<b>WHERE</b> (GeneralScore + SpeakingScore) / 2 > 6.5;	
4		
Data Output		Messages    Notifications
	<b>studentid</b> character varying (255) 🔒	<b>averagescore</b> double precision 🔒
1	ST0101	7.25
2	ST0102	7
3	ST0103	7.5
4	ST0104	7
5	ST0105	6.75
6	ST0206	7
7	ST0207	7.25

17. Hàm tính doanh thu theo quý của toàn bộ trung tâm:

```

CREATE OR REPLACE FUNCTION public.calculate_quarterly_revenue(input_year INT, quarter_number INT) RETURNS money
LANGUAGE plpgsql
AS $$
DECLARE
    quarterly_revenue MONEY := 0;
    quarter_start_month INT;
    class_record RECORD;
BEGIN
    quarter_start_month := (quarter_number - 1) * 3 + 1;

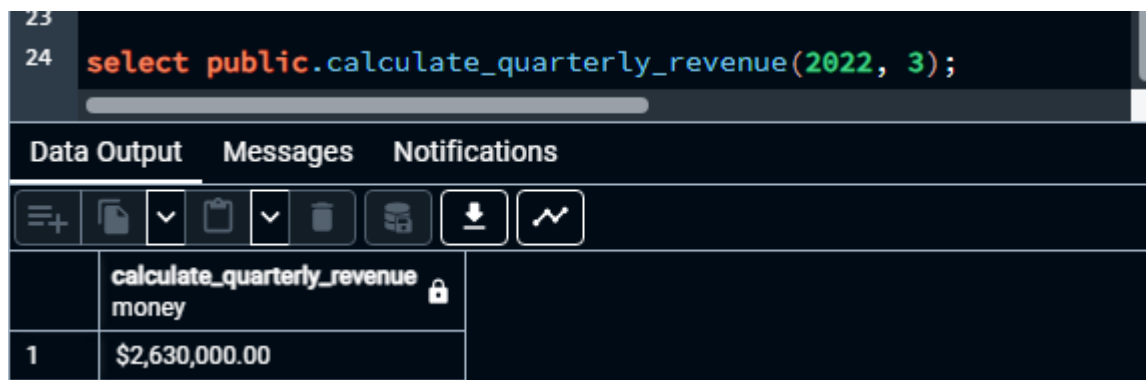
    FOR class_record IN (
        SELECT ClassID
        FROM public.Clazz
        WHERE EXTRACT(MONTH FROM OpenDate1) = quarter_start_month
        AND EXTRACT(YEAR FROM OpenDate1) = input_year
    ) LOOP
        quarterly_revenue := quarterly_revenue + public.calculate_class_revenue(class_record.ClassID);
    END LOOP;

    RETURN quarterly_revenue;
END;
$$;

select public.calculate_quarterly_revenue(2022, 1);

```

Kết quả:



The screenshot shows a SQL IDE interface. At the top, a query is entered in a text area: `select public.calculate_quarterly_revenue(2022, 3);`. Below the text area, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, displaying a table with the results of the query. The table has two columns: the function name 'calculate\_quarterly\_revenue' and the data type 'money'. The first row shows the result '1' and the value '\$2,630,000.00'.

	calculate_quarterly_revenue	money
1	\$2,630,000.00	

Cơ chế:

- Đầu vào: năm, quý.
- Tính doanh thu của tất cả các lớp mở trong 1 quý (có ngày OpenDate1 trùng với tháng bắt đầu quý).

18. Hàm hiển thị điểm giữa kỳ, cuối kỳ, số buổi nghỉ và học phí nợ của học viên:

```

CREATE OR REPLACE FUNCTION public.get_student_grades(student_id
VARCHAR(255))
RETURNS TABLE (
    midterm_score REAL,

```

```
    finalterm_score REAL,

    absence INT,

    unpaid_fees MONEY
)

AS $$

BEGIN

    RETURN QUERY

    SELECT

        ssc.MidTermScore,

        ssc.FinalTermScore,

        ssc.Absence,

        ssc.UnpaidFees

    FROM Student_Status_At_Class ssc

    WHERE ssc.StudentID = get_student_grades.student_id;

END;

$$ LANGUAGE plpgsql;
```

Query Query History

1 select \* from public.get\_student\_grades('ST5555')

Data Output Messages Notifications

	midterm_score real	finalterm_score real	absence integer	unpaid_fees money
1	8	9	4	\$0.00

19. Hàm đưa ra danh sách các lớp học phù hợp cho học viên dựa theo thời gian học và kết quả test đầu vào:



```

CREATE OR REPLACE FUNCTION public.recommend_classes(student_id text, input_year int, quarter_number int)
RETURNS TABLE (classid varchar(255))
LANGUAGE plpgsql
AS $$
DECLARE
    entrance_average_score REAL;
    recommended_course_id TEXT;
    quarter_start_month INT;
BEGIN
    SELECT (GeneralScore + SpeakingScore) / 2
    INTO entrance_average_score
    FROM public.Entrance_Test
    WHERE StudentID = recommend_classes.student_id;

    recommended_course_id := CASE
        WHEN entrance_average_score >= 0 AND
            entrance_average_score < 3.5 THEN 'IE35'
        WHEN entrance_average_score >= 3.5 AND
            entrance_average_score < 4.5 THEN 'IE45'
        WHEN entrance_average_score >= 4.5 AND
            entrance_average_score < 5.5 THEN 'IE55'
        WHEN entrance_average_score >= 5.5 THEN 'IE65'
        ELSE NULL
    END;

    quarter_start_month := (quarter_number - 1) * 3 + 1;

    RETURN QUERY
    SELECT cl.ClassID
    FROM public.Clazz cl
    WHERE cl.CourseID = recommended_course_id
        AND EXTRACT(MONTH FROM cl.OpenDate1) = quarter_start_month
        AND EXTRACT(YEAR FROM cl.OpenDate1) = input_year;
END;
$$;

select * from public.recommend_classes('ST0001', 2022, 3);

```

Kết quả:

37	
38	<code>select * from public.recommend_classes('ST0001', 2022, 3);</code>
Data Output Messages Notifications	
<div> <div>+</div> <div>📄</div> <div>▼</div> <div>📋</div> <div>▼</div> <div>🗑️</div> <div>🗄️</div> <div>⬇️</div> <div>📈</div> </div>	
	<div> <div>classid</div> <div>character varying 🔒</div> </div>
1	IE3500001
2	IE3500002
3	IE3500003
4	IE3500004
5	IE3500005
6	IE3500006
7	IE3500007
8	IE3500008
9	IE3500009
<div> <div>Total rows: 20 of 20</div> <div>Query complete 00:00:00.045</div> <div>Ln 8, Col 32</div> </div>	

Cơ chế:

- Đầu vào: StudentID, năm học, quý muốn học.
- Hàm sẽ tính trung bình kết quả thi test đầu vào của học viên và đưa ra danh sách các lớp học phù hợp theo thời điểm học viên muốn theo học.

20. Hàm lập danh sách học sinh giỏi trong 1 lớp học:

```

CREATE OR REPLACE FUNCTION public.get_class_details(class_id text, mode text)
RETURNS TABLE (
    student_id character varying(255),
    student_name character varying(255),
    phone_number character varying(255),
    student_address character varying(255),
    midterm_score real,
    finalterm_score real,
    ielts_score real
)
LANGUAGE plpgsql
AS $$
BEGIN
    IF mode = 'Class' THEN
        RETURN QUERY
        SELECT
            st.StudentID,
            st.StudentName,
            st.PhoneNumber,
            st.StudentAddress,
            ssac.MidTermScore,
            ssac.FinalTermScore,
            ieltsscore AS ieltsscore
        FROM public.Student st
        JOIN public.Student_Status_At_Class ssac ON st.StudentID = ssac.StudentID
        WHERE ssac.ClassID = class_id
            AND (ssac.MidTermScore * 0.3 + ssac.FinalTermScore * 0.7) >= 8;

    ELSIF mode = 'IELTS' THEN
        RETURN QUERY
        SELECT
            st.StudentID,
            st.StudentName,
            st.PhoneNumber,
            st.StudentAddress,
            midtermscore AS midtermscore,
            finaltermscore AS finaltermscore,
            ssac.IELTSScore
        FROM public.Student st
        JOIN public.Student_Status_At_Class ssac ON st.StudentID = ssac.StudentID
        WHERE ssac.ClassID = class_id
            AND ssac.IELTSScore >= ssac.EndCoursePoint;

    ELSE
        RAISE EXCEPTION 'Invalid mode. Please use "Class" or "IELTS".';
    END IF;
END;

```

✿;

```

select * from public.get_class_details('IE3500001', 'Class');

```

Kết quả:

49

50

</

Cơ chế:

- Đầu vào: mã lớp học, mode: Class hoặc IELTS.
- Nếu mode là Class: hàm sẽ tính những học sinh có điểm tổng kết khóa học lớn hơn hoặc bằng 8 trong lớp học.
- Nếu mode là IELTS: hàm sẽ liệt kê ra những học viên có điểm IELTS lớn hơn hoặc bằng điểm đầu ra khóa học.

21. Liệt kê các lớp học vẫn đang học tới thời điểm hiện tại:

```

CREATE OR REPLACE FUNCTION public.find_active_classes()
RETURNS TABLE (
    class_id character varying(255),
    class_name character varying(255),
    course_id character varying(255),
    tutor_id character varying(255),
    open_date1 date,
    open_time1 time without time zone
)
LANGUAGE plpgsql
AS $$
BEGIN
    RETURN QUERY
    SELECT
        c.ClassID,
        c.ClassName,
        c.CourseID,
        c.TutorID,
        c.OpenDate1,
        c.OpenTime1
    FROM public.Clazz c
    WHERE EXISTS (
        SELECT 1
        FROM public.Student_Status_At_Class ssac
        WHERE ssac.ClassID = c.ClassID
        AND (ssac.StudentStatus = 'Active' OR ssac.StudentStatus = 'Active*')
    );
END;
$$;

select * from public.find_active_classes();

```

15

16

17

18

select \* from public.find\_active\_classes();

Data Output

Messages

Notifications

	class_id character varying	class_name character varying	course_id character varying	tutor_id character varying	open_date1 date	open_time1 time without time zone
1	IE3500081	IELTS 3.5 Class 81	IE35	TU0201	2023-07-03	14:00:00
2	IE3500082	IELTS 3.5 Class 82	IE35	TU0202	2023-07-03	14:00:00
3	IE3500083	IELTS 3.5 Class 83	IE35	TU0203	2023-07-03	19:00:00
4	IE3500084	IELTS 3.5 Class 84	IE35	TU0204	2023-07-03	16:00:00
5	IE3500085	IELTS 3.5 Class 85	IE35	TU0205	2023-07-03	16:00:00
6	IE3500086	IELTS 3.5 Class 86	IE35	TU0206	2023-07-03	14:00:00
7	IE3500087	IELTS 3.5 Class 87	IE35	TU0207	2023-07-03	19:00:00
8	IE3500088	IELTS 3.5 Class 88	IE35	TU0208	2023-07-03	19:00:00
9	IE3500089	IELTS 3.5 Class 89	IE35	TU0209	2023-07-03	19:00:00
10	IE3500090	IELTS 3.5 Class 90	IE35	TU0210	2023-07-03	19:00:00
11	IE3500091	IELTS 3.5 Class 91	IE35	TU0211	2023-07-03	16:00:00
12	IE3500092	IELTS 3.5 Class 92	IE35	TU0212	2023-07-03	16:00:00
13	IE3500093	IELTS 3.5 Class 93	IE35	TU0213	2023-07-03	08:00:00
14	IE3500094	IELTS 3.5 Class 94	IE35	TU0214	2023-07-03	10:00:00

Total rows: 80 of 80

Query complete 00:00:00.068

Cơ chế:

- Tìm các lớp học trong đó có học sinh đang còn trạng thái Active và liệt kê thông tin.

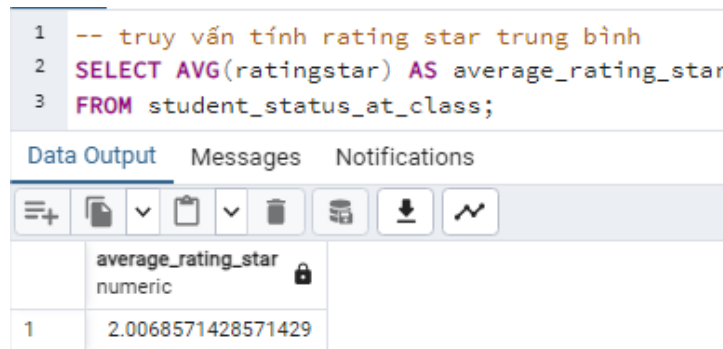
22. Truy vấn hiển thị danh sách học viên còn nợ học phí

1	<code>SELECT s.studentid, s.phonenumber, s.studentaddress</code>
2	<code>FROM student s JOIN student_status_at_class ssc ON s.studentid = ssc.studentid</code>
3	<code>WHERE ssc.unpaidfees :: NUMERIC &gt; 0;</code>
4	

Data Output	Messages	Notifications
<div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> <div></div> </div>		
studentid	phonenumber	studentaddress
[PK] character varying (255)	character varying (255)	character varying (255)
133	ST4638	0336305644 860 Highland Avenue, Indianapolis
134	ST4639	0326978917 582 Highland Avenue, Fort Worth
135	ST4640	0336986209 711 Stone Road, San Francisco
136	ST4642	0321951019 791 Sunset Boulevard, Philadelphia
137	ST4643	0347233377 589 Ocean Avenue, Philadelphia
138	ST4645	033181514 164 Birch Street, El Paso
139	ST4646	0311811432 3 Ridge Road, Los Angeles
140	ST4647	0311031689 198 Pine Street, Denver
141	ST4649	0329518754 842 Sixth Street, San Diego
142	ST4650	0329920191 342 Highland Avenue, Indianapolis
143	ST4651	0338985572 979 Lake Drive, Denver
144	ST4653	0334551179 897 Ninth Street, Chicago
145	ST4654	0346400133 570 Meadow Lane, San Jose

- Giải thích: truy vấn sẽ hiển thị ra số điện thoại và địa chỉ của học viên chưa đóng tiền học phí thông qua hai bảng student và student\_status\_at\_class.

23. Truy vấn tính rating star trung bình của tất cả các lớp.



```
1 -- truy vấn tính rating star trung bình
2 SELECT AVG(ratingstar) AS average_rating_star
3 FROM student_status_at_class;
```

	average_rating_star
1	2.0068571428571429

- Giải thích: truy vấn sẽ hiển thị ra rating star trung bình của tất cả học sinh từ đó làm cơ sở đánh giá hoạt động dạy và học của trung tâm.

24. Hiển thị số lượng từng band điểm mà học sinh đạt được trong bài test đầu vào:

```
SELECT COUNT(*), (GeneralScore + SpeakingScore) / 2 AS ScoreGroup
FROM public.Entrance_Test
GROUP BY ScoreGroup
ORDER BY ScoreGroup;
```

25. Hiển thị tỉ lệ phần trăm học viên theo phân loại điểm đầu ra:

```

WITH StudentCounts AS (
    SELECT
        EndCoursePoint,
        COUNT(*) AS total_students
    FROM
        public.Student_Status_At_Class
    GROUP BY
        EndCoursePoint
),
TotalStudents AS (
    SELECT COUNT(*) AS all_students FROM public.Student_Status_At_Class
)
SELECT
    sc.EndCoursePoint,
    (sc.total_students::decimal / ts.all_students) * 100 AS percentage
FROM
    StudentCounts sc
JOIN
    TotalStudents ts ON 1=1;

```

Kết quả:

```

7      GROUP BY
8          EndCoursePoint
9  ),
10 TotalStudents AS (
11     SELECT COUNT(*) AS all_students FROM public.Student_Status_
12 )
13 SELECT
14     sc.EndCoursePoint,
15     (sc.total_students::decimal / ts.all_students) * 100 AS per
16 FROM
17     StudentCounts sc
18 JOIN
19     TotalStudents ts ON 1=1;

```

Data Output   Messages   Notifications



	endcoursepoint real	percentage numeric
1	5.5	5.00000000000000000000
2	3.5	2.00000000000000000000
3	6.5	90.00000000000000000000
4	4.5	3.00000000000000000000

Cơ chế:

- Tạo một câu truy vấn StudentCounts: tính số học sinh đi học theo từng band điểm.
- Dùng câu lệnh truy vấn để tính tỉ lệ phần trăm các điểm đầu ra trên tổng số.

## 26. Hàm tìm số học viên mà một tư vấn viên đã tư vấn

```

1 CREATE OR REPLACE FUNCTION count_students_advised(counselor_id VARCHAR(255))
2 RETURNS INTEGER AS $$
3 DECLARE
4     student_count INTEGER;
5 BEGIN
6     SELECT COUNT(*)
7     INTO student_count
8     FROM student
9     WHERE counselorid = counselor_id;
10
11     RETURN student_count;
12 END;
13 $$ LANGUAGE plpgsql;
14

```

Kết quả:

1	SELECT count_students_advised('CS0001')
2	

Data Output	Messages	Explain X	Notifications
<div> <div>count_students_advised</div> <div>integer</div> </div>			
1		25	

## B. Các trigger:

### 1. Trigger kiểm tra định dạng meeting link

```

1 -- Tạo lại trigger
2 CREATE OR REPLACE FUNCTION validate_meeting_link()
3 RETURNS TRIGGER AS $$
4 BEGIN
5     IF NEW.meetinglink IS NOT NULL AND NOT NEW.meetinglink LIKE 'meet.google.com%' THEN
6         RAISE EXCEPTION 'Meeting link must start with "meet.google.com": %', NEW.meetinglink;
7     END IF;
8
9     RETURN NEW;
10 END;
11 $$ LANGUAGE plpgsql;
12
13 CREATE TRIGGER check_meeting_link_format
14 BEFORE INSERT OR UPDATE ON entrance_test
15 FOR EACH ROW
16 EXECUTE FUNCTION validate_meeting_link();

```



- Giải thích:
  - + Trigger sẽ kiểm tra định dạng khi insert hoặc update vào bảng entrance\_test xem có đúng định dạng: meet.google.com... hay không.
- Kết quả:

```

1 UPDATE entrance_test
2 SET meetinglink = 'meet.google/updated-link'
3 WHERE testid = 'TE0001';
4

```

Data Output Messages Notifications

ERROR: Meeting link must start with "meet.google.com": meet.google/updated-link  
 CONTEXT: PL/pgSQL function validate\_meeting\_link() line 4 at RAISE  
 SQL state: P0001

## 2. Trigger: không thể nhập hoặc update điểm speaking và điểm general ngoài khoảng 0-9

```

1 CREATE OR REPLACE FUNCTION validate_entrance_test_scores()
2 RETURNS TRIGGER AS $$
3 BEGIN
4     IF NEW.globalscore IS NOT NULL AND (NEW.globalscore < 0 OR NEW.globalscore > 9.0) THEN
5         RAISE EXCEPTION 'Invalid globalscore value: %. Valid values are in range 0-9.0', NEW.globalscore;
6     END IF;
7     IF NEW.speakingscore IS NOT NULL AND (NEW.speakingscore < 0 OR NEW.speakingscore > 9.0) THEN
8         RAISE EXCEPTION 'Invalid speakingscore value: %. Valid values are in range 0-9.0', NEW.speakingscore;
9     END IF;
10    RETURN NEW;
11 END;
12 $$ LANGUAGE plpgsql;
13
14 CREATE TRIGGER validate_entrance_test_scores_trigger
15 BEFORE INSERT OR UPDATE ON entrance_test
16 FOR EACH ROW
17 EXECUTE FUNCTION validate_entrance_test_scores();

```

- Giải thích:
  - + Trigger sẽ kiểm tra điểm globalscore và điểm speakingscore nhập vào hoặc update có nằm trong khoảng 0-9 không.
  - + Nếu nằm trong khoảng sẽ thực hiện hành động update hoặc insert.
  - + Nếu không nằm trong khoảng sẽ hiển thị thông báo lỗi.
- Kết quả:

```

1 UPDATE entrance_test
2 SET globalscore = 9.2
3 WHERE testid = 'TE0001';

```

Data Output Messages Notifications

ERROR: Invalid globalscore value: 9.2. Valid values are in range 0-9.0  
 CONTEXT: PL/pgSQL function validate\_entrance\_test\_scores() line 4 at RAISE  
 SQL state: P0001

3 - 4. Trigger để báo lỗi và ngăn chặn thay đổi khi số điện thoại và số tài khoản ngân hàng sai format:

```
CREATE OR REPLACE FUNCTION public.validate_phone_number()
RETURNS TRIGGER AS $$
BEGIN
    IF NEW.PhoneNumber !~ '^0\d{9}$' THEN
        RAISE EXCEPTION 'So dien thoai phai bat dau bang so 0 va co 10 chu so';
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE OR REPLACE FUNCTION public.validate_bank_account()
RETURNS TRIGGER AS $$
BEGIN
    IF NEW.BankAccount !~ '^\\d{9,12}$' THEN
        RAISE EXCEPTION 'So tai khoan ngan hang phai co tu 9 den 12 chu so';
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER tutor_phone_validation
BEFORE INSERT OR UPDATE ON public.Tutor
FOR EACH ROW
EXECUTE PROCEDURE public.validate_phone_number();

CREATE TRIGGER tutor_bank_account_validation
BEFORE INSERT OR UPDATE ON public.Tutor
FOR EACH ROW
EXECUTE PROCEDURE public.validate_bank_account();

CREATE TRIGGER student_phone_validation
BEFORE INSERT OR UPDATE ON public.Student
FOR EACH ROW
EXECUTE PROCEDURE public.validate_phone_number();

CREATE TRIGGER counselor_phone_validation
BEFORE INSERT OR UPDATE ON public.Counselor
FOR EACH ROW
EXECUTE PROCEDURE public.validate_phone_number();

CREATE TRIGGER counselor_bank_account_validation
BEFORE INSERT OR UPDATE ON public.Counselor
FOR EACH ROW
EXECUTE PROCEDURE public.validate_bank_account();
```

a. Hàm validate\_phone\_number():

- Đầu vào: Hàm nhận một bản ghi mới (NEW) là đối tượng đang được thêm vào hoặc cập nhật.
  - Cơ chế hoạt động: Hàm kiểm tra xem số điện thoại trong cột PhoneNumber của bản ghi mới (NEW) có bắt đầu bằng số 0 và có 10 chữ số hay không. Nếu không, hàm sẽ báo lỗi **“So dien thoai phai bat dau bang so 0 va co 10 chu so”**.
  - Đầu ra: Hàm trả về bản ghi mới (NEW) nếu số điện thoại hợp lệ, hoặc báo lỗi nếu không hợp lệ.
- b. Hàm validate\_bank\_account():
- Đầu vào: Hàm nhận một bản ghi mới (NEW) là đối tượng đang được thêm vào hoặc cập nhật.
  - Cơ chế hoạt động: Hàm kiểm tra xem số tài khoản ngân hàng trong cột BankAccount của bản ghi mới (NEW) có từ 9 đến 12 chữ số hay không. Nếu không, hàm sẽ báo lỗi **“So tai khoan ngan hang phai co tu 9 den 12 chu so.”**
  - Đầu ra: Hàm trả về bản ghi mới (NEW) nếu số tài khoản hợp lệ, hoặc báo lỗi nếu không hợp lệ.

Query Query History

```
1 insert into public.student values('ST7500', 'ABC', '1234567891', 'Oak St.', 'CS6
```

Data Output Messages Notifications

ERROR: So dien thoai phai bat dau bang so 0 va co 10 chu so  
CONTEXT: PL/pgSQL function validate\_phone\_number() line 4 at RAISE

## C. Hiệu năng câu truy vấn:

- Hàm 1,2,3 phụ thuộc vào hiệu năng của function calculate\_class\_revenue, do vậy chúng ta sẽ phải tối ưu function này:
  - + **Hiệu năng calculate\_class\_revenue trước khi index:**

Data Output	Messages	Notifications
<div> <div>+</div> <div>📄</div> <div>▼</div> <div>📋</div> <div>▼</div> <div>🗑️</div> <div>🗄️</div> <div>⬇️</div> <div>📈</div> </div>		
	<b>QUERY PLAN</b> text	
1	Function Scan on calculate_class_revenue (cost=0.25..0.26 rows=1 width=8) (actual time=1.309	
2	Planning Time: 0.025 ms	
3	Execution Time: 1.317 ms	

- Index để tối ưu:

```

-- Index on Student_Status_At_Class
CREATE INDEX student_status_classid_idx ON Student_Status_At_Class (ClassID);

-- Index on Clazz
CREATE INDEX clazz_classid_courseid_idx ON Clazz (ClassID, CourseID);

-- Index on Course
CREATE INDEX course_courseid_fee_idx ON Course (CourseID, fee);

```

- Sau khi sử dụng index:

45

46

47

explain analyse

select \* from public.calculate\_class\_revenue('IE5500001')

Data Output

Messages

Notifications

QUERY PLAN

text

1

Function Scan on calculate\_class\_revenue (cost=0.25..0.26 rows=1 width=8) (actual time=0.524

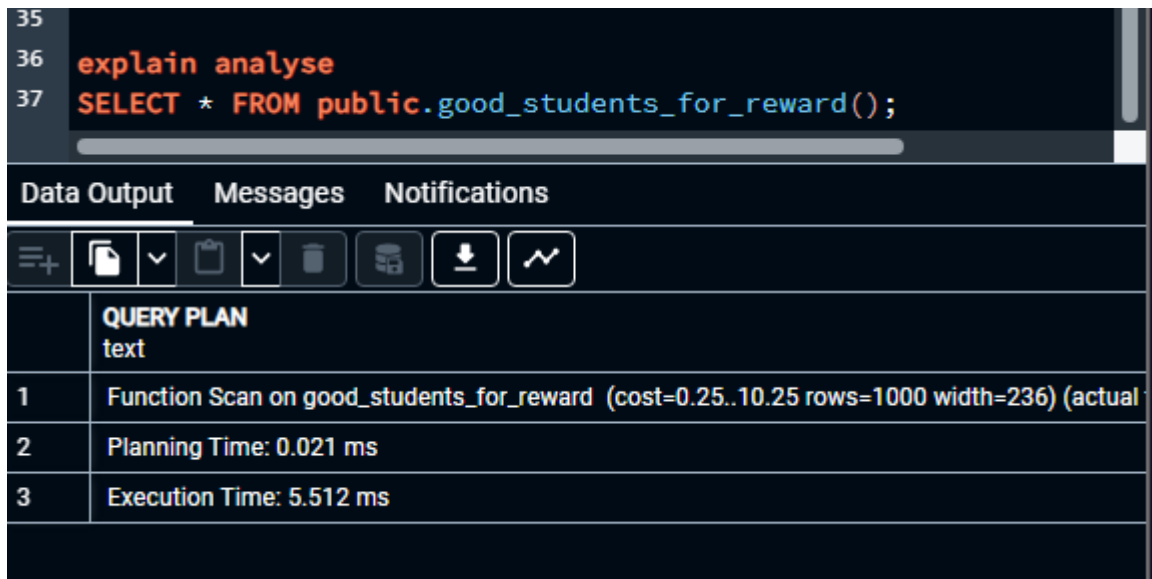
2

Planning Time: 0.034 ms

3

Execution Time: 0.537 ms

- Liệt kê danh sách top 100 học sinh có điểm tốt nhất - chưa được tối ưu bằng index:



The screenshot shows a database query execution interface. The query entered is `explain analyse SELECT * FROM public.good_students_for_reward();`. Below the query editor, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, displaying a table with the query plan. The table has two columns: a step number and a description of the operation. The steps are: 1. Function Scan on good\_students\_for\_reward (cost=0.25..10.25 rows=1000 width=236) (actual: 1000); 2. Planning Time: 0.021 ms; 3. Execution Time: 5.512 ms.

	QUERY PLAN text
1	Function Scan on good_students_for_reward (cost=0.25..10.25 rows=1000 width=236) (actual: 1000)
2	Planning Time: 0.021 ms
3	Execution Time: 5.512 ms

- Index sử dụng để tối ưu:  
**CREATE INDEX student\_performance\_idx  
ON Student\_Status\_At\_Class (IELTSScore DESC,  
(IELTSScore - EndCoursePoint) DESC);  
CREATE INDEX student\_studentid\_idx ON Student  
(StudentID);**

- Kết quả sau khi tối ưu:

QueryQuery History

```

1 CREATE OR REPLACE FUNCTION public.good_students_for_reward()
2 RETURNS TABLE (
3     StudentID VARCHAR(255),
4     StudentName VARCHAR(255),
5     PhoneNumber VARCHAR(255),
6     StudentAddress VARCHAR(255),
7     CounselorID VARCHAR(255),
8     ClassID VARCHAR(255),
9     ClassName VARCHAR(255),
10    IELTSscore REAL,
11    EndCoursePoint REAL,
12    Difference REAL
13 ) AS $$
14 BEGIN

```

Data OutputMessagesNotifications

≡+

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

	QUERY PLAN text
1	Function Scan on good_students_for_reward (cost=0.25..10.25 rows=1000 width=236) (actual
2	Planning Time: 0.042 ms
3	Execution Time: 0.634 ms

- Hàm `find_overlapping_classes()`:
  - + Trước khi index:

```

9▼ BEGIN
10     RETURN QUERY
11     SELECT DISTINCT c1.ClassID
12     FROM public.Clazz AS c1
13     JOIN public.Clazz AS c2 ON c1.ClassID <> c2.ClassID
14     AND c1.OpenDate1 <= c2.OpenDate2 AND c1.OpenDate2 >=

```

Data Output Messages Notifications

QUERY PLAN  
text

1	Function Scan on find_overlapping_classes4 (cost=0.25..10.25 rows=1000 width=32) (actual ti
2	Planning Time: 0.032 ms
3	Execution Time: 0.456 ms

## + Sử dụng chung index

```

-- Index on Student_Status_At_Class
CREATE INDEX student_status_classid_idx ON Student_Status_At_Class (ClassID);

-- Index on Clazz
CREATE INDEX clazz_classid_courseid_idx ON Clazz (ClassID, CourseID);

-- Index on Course
CREATE INDEX course_courseid_fee_idx ON Course (CourseID, fee);

```

## + Hiệu năng sau index:

```

22
23 explain analyse
24 select * from public.find_overlapping_classes4('2022-07-04', '

```

Data Output Messages Notifications

QUERY PLAN  
text

1	Function Scan on find_overlapping_classes4 (cost=0.25..10.25 rows=1000 width=32) (actual ti
2	Planning Time: 0.024 ms
3	Execution Time: 0.275 ms

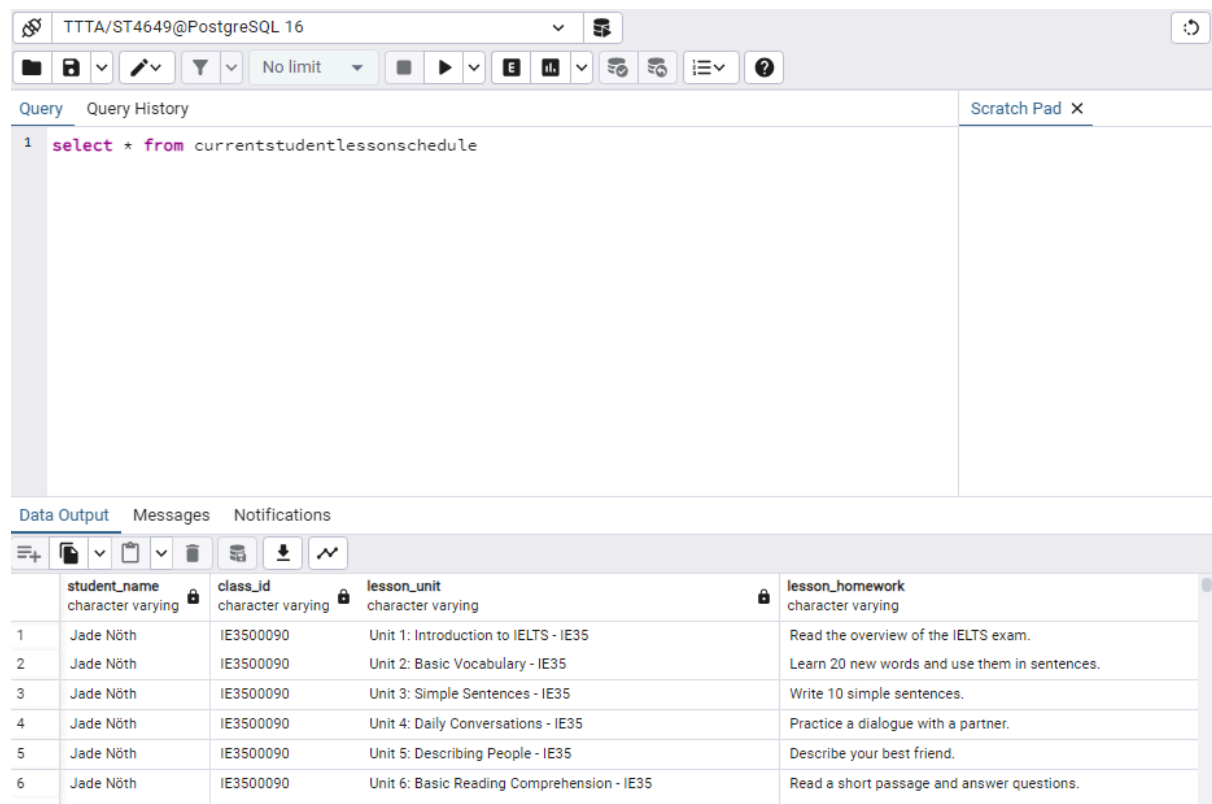
## IV. Các khung nhìn

Minh họa khung nhìn cho một số chức năng tiêu biểu đối với mỗi người dùng.  
Có 3 người dùng đại diện cho 3 group role:

- Với tutor: tài khoản ‘TU0305’
- Với student: tài khoản ‘ST4649’
- Với counselor: tài khoản ‘CS0009’

Các tài khoản này đều được cấp những quyền phù hợp vào các khung nhìn tương ứng.

### 1. Khung nhìn xem lịch học của học sinh



The screenshot shows a PostgreSQL query editor interface. The top bar indicates the connection is 'TTTA/ST4649@PostgreSQL 16'. The query editor contains the following SQL query:

```
1 select * from currentstudentlessonschedule
```

The 'Data Output' tab is active, displaying the results of the query in a table format. The table has four columns: student\_name, class\_id, lesson\_unit, and lesson\_homework. The data shows six rows of lesson schedules for a student named Jade Nöth.

	student_name character varying	class_id character varying	lesson_unit character varying	lesson_homework character varying
1	Jade Nöth	IE3500090	Unit 1: Introduction to IELTS - IE35	Read the overview of the IELTS exam.
2	Jade Nöth	IE3500090	Unit 2: Basic Vocabulary - IE35	Learn 20 new words and use them in sentences.
3	Jade Nöth	IE3500090	Unit 3: Simple Sentences - IE35	Write 10 simple sentences.
4	Jade Nöth	IE3500090	Unit 4: Daily Conversations - IE35	Practice a dialogue with a partner.
5	Jade Nöth	IE3500090	Unit 5: Describing People - IE35	Describe your best friend.
6	Jade Nöth	IE3500090	Unit 6: Basic Reading Comprehension - IE35	Read a short passage and answer questions.

### 2. Khung nhìn xem kết quả kiểm tra đầu vào của học sinh



TTTA/ST4649@PostgreSQL 16

Query Query History Scratch Pad X

```
1 select * from studententrancetest
```

Data Output Messages Notifications

	testid character varying (255)	generalscore real	speakingsscore real	meetinglink character varying (255)
1	TE4649	1.5	1.5	meet.google.com/abc956713-996325

### 3. Khung nhìn các khóa học được đề xuất của học sinh

Query Query History Scratch Pad X

```
1 select * from studentrecommendedcourse
```

Data Output Messages Notifications

	recommendedcourselevel text
1	IE35

### 4. Khung nhìn các lớp dạy của giáo viên



Query

Query History

Scratch Pad X

```

1 update teacherentrancetest
2 set generalscore = 2.5
3 where testid = 'TE0664'

```

Data Output

Messages

Notifications

UPDATE 1

Query returned successfully in 49 msec.

✓ Query returned successfully in 49 msec. X

Total rows: 19 of 19

Query complete 00:00:00.049

Ln 3, Col 23

Query

Query History

Scratch Pad X

```

1 select * from teacherentrancetest
2 where testid = 'TE0664'

```

Data Output

Messages

Notifications

	testid character varying (255)	generalscore real	speakingsscore real	entrancedate date	entrancetime time without time zone	meetinglink character varying (255)	tutorid character varying (255)
1	TE0664	2.5	2.5	2022-06-09	09:00:00	meet.google.com/abc207822-238258	TU0305

6. Khung nhìn cho phép giáo viên sửa, nhập điểm midterm, finalterm của học sinh do các lớp mình quản lý

TTTA/TU0305@PostgreSQL 16

Query Query History Scratch Pad X

```
1 select * from tutorstudentperformance
```

Data Output Messages Notifications

	studentid character varying (255)	classid character varying (255)	midtermscore real	finaltermscore real	absence integer	tutorid character varying (255)
1	ST6512	IE3500125	9	4.5	4	TU0305
2	ST6513	IE3500125	8.5	7	4	TU0305
3	ST6514	IE3500125	8	9.5	0	TU0305
4	ST6515	IE3500125	9	5.5	2	TU0305
5	ST6516	IE3500125	4.5	9.5	4	TU0305
6	ST6517	IE3500125	8	8	2	TU0305
7	ST6518	IE3500125	7	8	2	TU0305

## - Nhập/update điểm:

TTTA/TU0305@PostgreSQL 16

Query Query History Scratch Pad X

```
1 update tutorstudentperformance
2 set midtermscore = 8.5
3 where studentid = 'ST6512'
```

Data Output Messages Notifications

UPDATE 1

Query returned successfully in 47 msec.

✓ Query returned successfully in 47 msec. X

Total rows: 20 of 20 Query complete 00:00:00.047

Ln 3, Col 26

Query Query History Scratch Pad X

```

1 select * from tutorstudentperformance
2 where studentid = 'ST6512'

```

Data Output Messages Notifications

	studentid character varying (255)	classid character varying (255)	midtermscore real	finaltermscore real	absence integer	tutorid character varying (255)
1	ST6512	IE3500125	8.5	4.5	4	TU0305

Total rows: 1 of 1    Query complete 00:00:00.095    ✓ Successfully run. Total query runtime: 95 msec. 1 rows affected. ✕    Ln 2, Col 26

## 7. Khung nhìn cho phép học sinh xem điểm midterm, finalterm và học phí nợ

TTTA/ST4649@PostgreSQL 16

Query Query History Scratch Pad X

```

1 select * from studentgrades

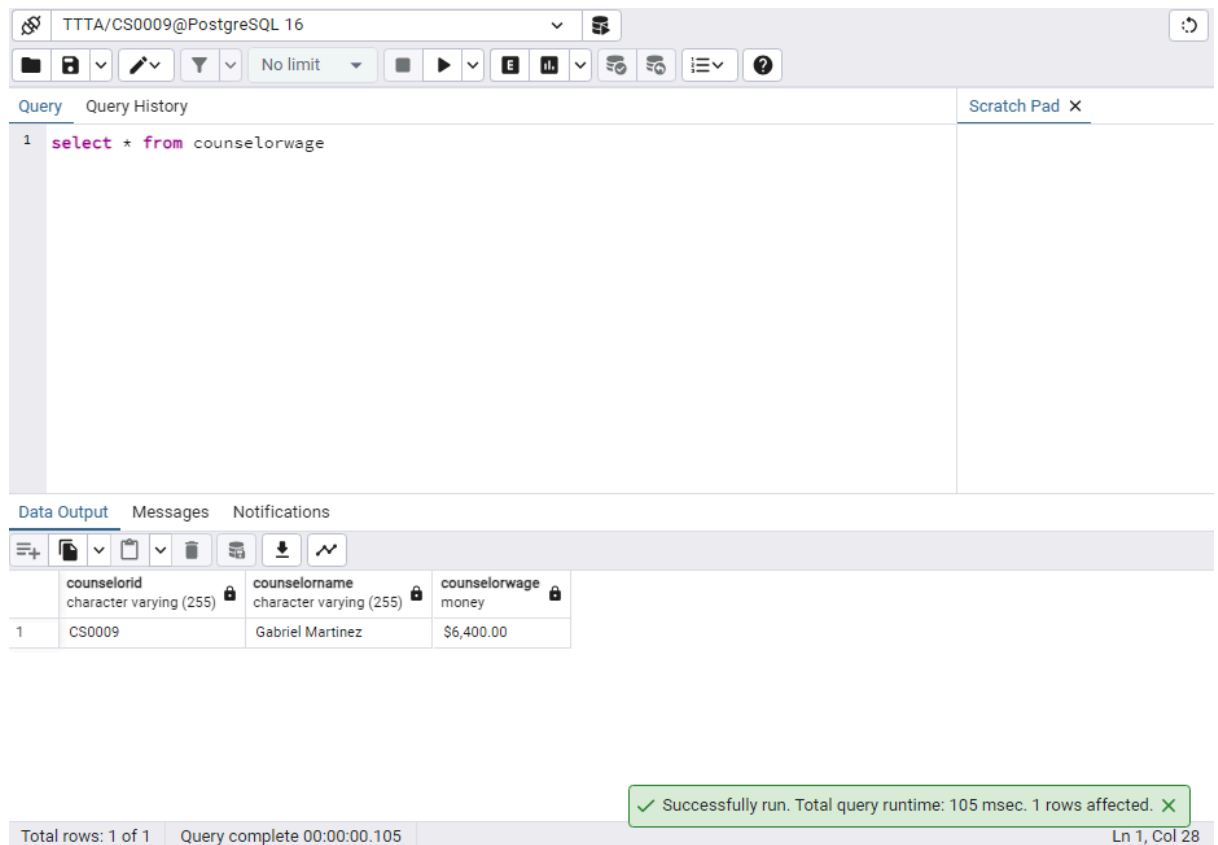
```

Data Output Messages Notifications

	midtermscore real	finaltermscore real	absence integer	unpaidfees money
1	9	6.5	0	\$1,000.00

Total rows: 1 of 1    Query complete 00:00:00.079    ✓ Successfully run. Total query runtime: 79 msec. 1 rows affected. ✕    Ln 1, Col 15

## 8. Khung nhìn cho phép tư vấn viên xem lương của mình



TTTA/CS0009@PostgreSQL 16

Query Query History Scratch Pad X

```
1 select * from counselorwage
```

Data Output Messages Notifications

	counselorid character varying (255)	counselorname character varying (255)	counselorwage money
1	CS0009	Gabriel Martinez	\$6,400.00

✓ Successfully run. Total query runtime: 105 msec. 1 rows affected. ✕

Total rows: 1 of 1 Query complete 00:00:00.105 Ln 1, Col 28

## Khó khăn và giải pháp

1. Khó khăn trong thiết kế ERD và generate data ban đầu
  - Nguyên nhân: Chưa hiểu rõ mô tả bài toán từ ban đầu dẫn tới khi bắt đầu generate data xảy ra hiện tượng dư thừa dữ liệu.
  - Giải pháp: Đọc kỹ lại mô tả bài toán và tách các bảng xảy ra hiện tượng dư thừa dữ liệu (VD: bảng student ban đầu đã được tách thành bảng student và bảng student\_status\_at\_class sau này)
2. Khó khăn trong việc viết các function
  - Nguyên nhân: Do chưa hoàn toàn thống nhất về database, lúc viết function còn làm việc một cách cá nhân.
  - Giải pháp: Ngồi lại họp và review từng function, chỉ ra những điểm còn sai sót trong các function.
3. Khó khăn khi viết index
  - Nguyên nhân: Mỗi câu function khi tối ưu cần một số index riêng.

- Giải pháp: Tìm ra index hiệu quả cho tất cả các function bằng cách liệt kê các index cần thiết của từng function. Sau đó index nào thấy lặp lại nhiều nhất thì sử dụng index đó.

## Ưu điểm và nhược điểm

### A. Ưu điểm

- Số lượng bản ghi lớn.
- Quản lý bao quát được các công việc của trung tâm (từ góc nhìn của quản lý, của giáo viên và cả của học viên)
- Có thể thực hiện được một số yêu cầu của trung tâm.
- Phân phối điểm học sinh sát thực tế
- Phân phối tỉ lệ học sinh bỏ học giữa chừng sát thực tế

### B. Nhược điểm

- Học viên sẽ không xem được điểm thi midterm và final term của khóa nhỏ trước đó mà sẽ chỉ xem được điểm của khóa gần nhất.
- Generate data còn cục bộ (VD: chỉ generate data cho các tháng 4-7 hoặc 1-4, 7-10)
- Nếu số học sinh không học hết 6.5 nhiều (bỏ dở) thì một lớp có thể sẽ vắng.

## Nhiệm vụ của mỗi thành viên

**Hoàng Yến Nhi:** Viết báo cáo, làm slide, lên ý tưởng khởi đầu, viết 10 câu truy vấn gồm các câu tính doanh thu, tỉ lệ đi học, các câu liên quan tới ratingstar, danh sách entrancetest trong ngày, hiển thị học sinh chưa đóng đủ học phí và các trigger trong bảng entrance\_test, đóng góp viết index, góp ý sửa vẽ sơ đồ thực thể liên kết.

**Phan Hoàng Hải:** Đóng góp và chỉnh sửa phần viết mô tả, đóng góp vào phần vẽ sơ đồ thực thể liên kết và lập sơ đồ quan hệ, sinh dữ liệu và xử lý dữ liệu trong database, viết 10 câu truy vấn hàm/ trigger về lịch học - giờ học của học viên, về điểm số và thống kê điểm đầu ra của lớp học, đánh dấu học sinh nghỉ học, viết index.

**Nguyễn Hoàng Phúc:** Đóng góp vào phần thiết kế mô hình thực thể liên kết, chuyển đổi sang sơ đồ quan hệ, chỉnh sửa mô tả, xây dựng các câu sql tạo bảng, viết 10 câu truy vấn/hàm/trigger, demo các chức năng tiêu biểu qua khung nhìn và tài khoản, đóng góp sinh ra dữ liệu cho các bảng, đóng góp viết index.