HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

---□□---

Database Lab

Student Management System (SMS)

Group 5

Lecturer: Vu Tuyet Trinh

Project Report

Class: 135409

Group: 5

Lecturer: Vu Tuyet Trinh

Members

Name	Student ID	Tasks
Le Minh Duc	20200164	Design, Generate data, Optimize queries, Grant users, Code demo
Tong Tran Minh Duc	20205147	Write functions (2.1.1 → 2.2.3), Create triggers
Hoang Van Phuong	20200478	Write functions (2.2.4 → 2.3.5), Create views

Table of Contents

Title

Student Management System (SMS)

1. Context

A Student Management System (SMS) is a database-driven application that assists educational institutions in digitizing student data and managing it more efficiently. The system creates a simple interface for students, lecturers, and administrative staff, offering great assistance within and outside classrooms. In this project, we want to develop a simple and effective database, which will then be connected to an application to simulate real-life situations.

2. Requirements

The system should provide different users with different capabilities:

2.1. Staff (Admin)

2.1.1. Adding students, lecturers, subjects, and classes with the following data.
   a. Students: Name, ID, School/Faculty, Program, Date of Birth, Address, Contact, etc.
   b. Lecturers: Name, ID, School/Faculty, Teaching Subjects, Date of Birth, Address, Contact, etc.
   c. Subjects: Prerequisites, Number of credits, etc.
   d. Classes: Time, Location, Slot Availability, Semester, Number of credits, Schedule of classes, etc.

2.1.2. Managing information of students, lecturers, subjects, and classes.

2.1.3. Creating a tentative timetable for the upcoming semester, i.e creating classes for each subject. A subject can have 0, 1, or many classes open in a semester.

2.1.4. Making changes to the timetable.

2.1.5. Assigning lecturers to classes based on the timetable.

2.1.6. Getting reports on
   a. Classes having a number of enrolled students fewer than [X], used for canceling classes with low enrollment.
   b. Students with credit debt from unfinished (failed) subjects in the range [A, B], used for sending warnings.
   c. Students qualified for semester scholarships with GPA ≥ [G].

2.2. Students

2.2.1. Viewing data of subjects, classes, and results of themselves.

2.2.2. View the tentative timetable to find suitable opening classes.

2.2.3. Enrolling in classes.
   a. Showing enrolling information such as time, location, credit, slot availability, and prerequisites.
   b. Check for slot availability, and class prerequisites.
   c. Check for time conflicts.
   d. Calculate total studying credits.
   e. Schedules are automatically identified after students enroll in classes.

2.2.4. Looking up data related to lecturers, subjects, classes, and other students. Retrieve only essential information, excluding personal ones like address, scores, and student timetable.
   a. Students: Name, ID, School/Faculty, Program, Contact, etc.
   b. Lecturers: Name, ID, School/Faculty, Teaching Subjects, Contact, etc.
   c. Subjects, Classes: Number of credits, Schedule of classes, etc.

2.2.5. Getting estimated fees for the current semester.
   a. Total Fees = Tuition Credits * Credit Price + Other Fees.
   b. The Credit Price can vary between programs.
   c. Other Fees can be Insurance Fees, Previous Debt, etc.
   d. Tuition Credits ≠ Study Credits.

2.2.6. Getting reports on Study Credits earned, GPA/CPA.

2.3. Lecturers

2.3.1. Viewing data of all teaching subjects, classes, and results of themselves.

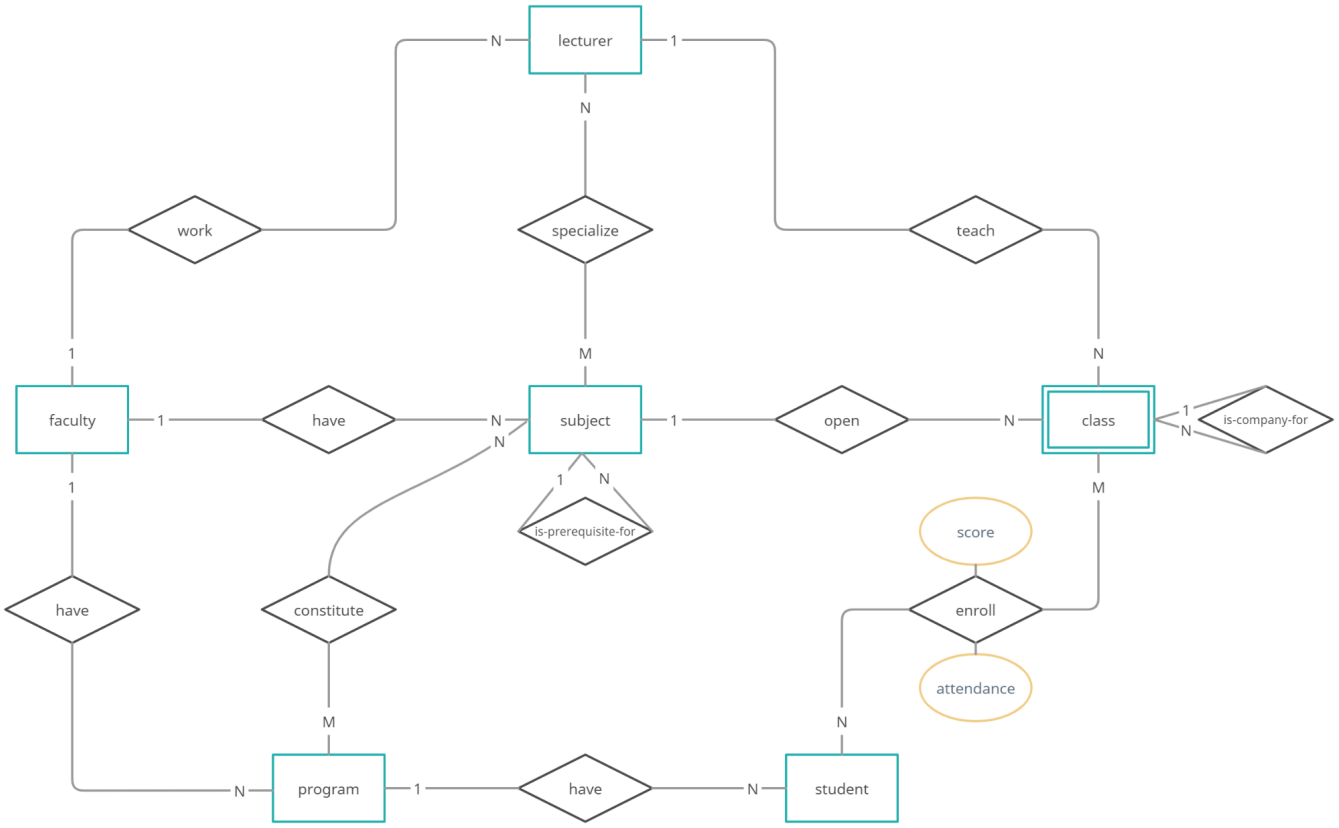
2.3.2. Looking up data related to students, subjects, classes, and other lecturers. Retrieve only essential information, excluding personal ones like address, scores, and student timetable.

a. Students: Name, ID, School/Faculty, Program, Contact, etc.
b. Lecturers: Name, ID, School/Faculty, Teaching Subjects, Contact, etc.
c. Subjects, Classes: Number of credits, Schedule of classes, etc.

2.3.3. Recording student academic performance.
2.3.4. Tracking student attendance.
2.3.5. Getting reports on
a. Exam grade distribution (statistics)
b. Student attendance up to today

3. Database Structure

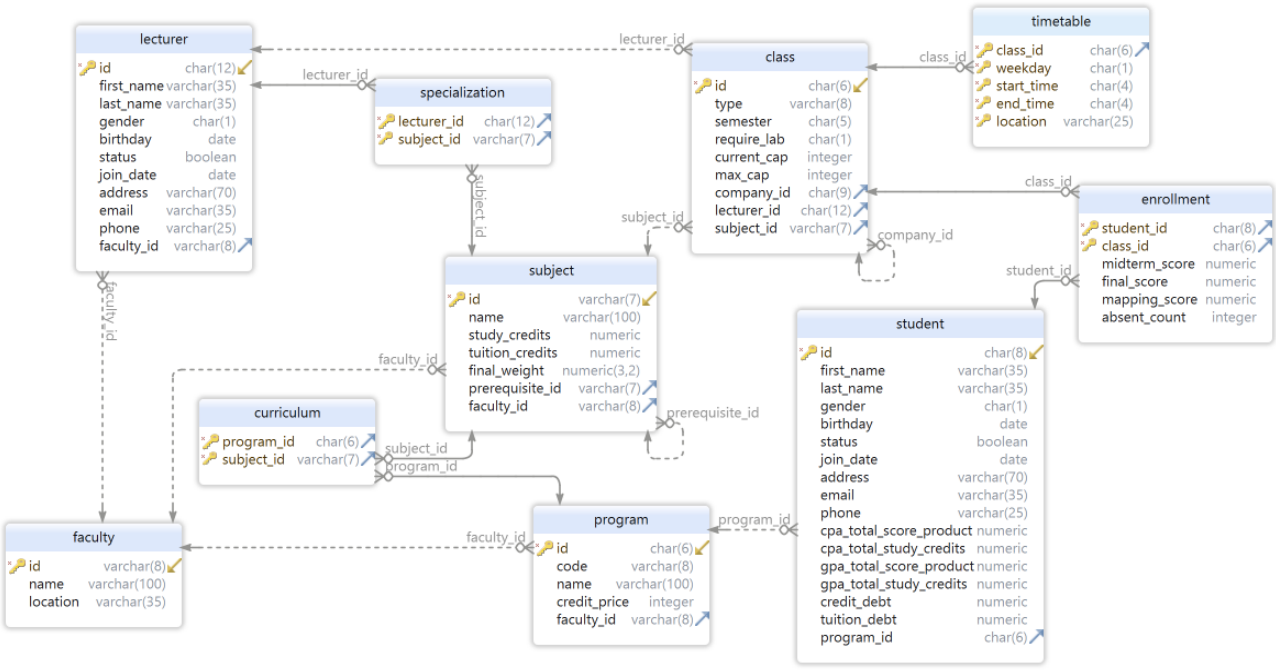
3.1. ERD



- Summary of changes:
  - ERD_v1: Initial design. The multivalued attributes aren't highlighted.
  - ERD_v2: Added relation between "subject" and "program". **This diagram doesn't include attributes**, leading to mapping mistakes of multivalued attributes in the "class" table as can be seen in SCHEMA_v1. Fixed in later schema's versions.
- Check GitHub for more details:

[https://github.com/duclm278/database-project/tree/main/structure]

3.2. Schema



- Summary of changes:
  - SCHEMA_v1: Initial mapping from ERD_v2.
  - SCHEMA_v2: Fixed mapping mistakes of multivalued attributes in the "class" table by creating the "timetable" table. Add the "curriculum" table. Add cpa, gpa attributes to the "student" table for storing results of the update_score trigger.
  - SCHEMA_v3: Separate cpa, gpa attributes into their corresponding total_score_product and total_study_credits attributes. This helps the trigger update cpa and gpa easier.
  - SCHEMA_v4: Add final_score attribute to the "enrollment" table.
- Check GitHub for more details:

[https://github.com/duclm278/database-project/tree/main/structure]

3.3. Tables

- Tables are created in the schema "public".
- Check GitHub for more details:

[https://github.com/duclm278/database-project/blob/main/sql/sms-create.sql]

4. Generate Data

- Use Python as the main programming language.
- Use Psycopg 2 to connect to and control the database.
- Use Faker to generate data.
- Some details about generated data:
  - 18 faculties, 64 programs, 2361 subjects

    - 12786 classes (5254 classes in 20212 & 7532 classes in 20221)
    - 14894 timetable slots (6096 slots in 20212 & 8798 slots in 20221)
    - Each lecturer is assigned to a maximum of 5 subjects and 25 classes.
    - Each curriculum has a maximum of 50 subjects.
    - Generate 5000 students per year in the total of three years (2020, 2021, 2022).
    - Enroll these students to classes of the semester 20212 (maximum 10 subjects).
- Check GitHub for more details:

[https://github.com/duclm278/database-project/blob/main/gen.py]

5. Views

There are 4 views for full self info and 3 views for restricted searchings:

- student.self_view_info
- student.self_view_curriculum
- lecturer.self_view_info
- lecturer.self_view_specializations
- search.view_search_student
- search.view_search_lecturer
- search.view_search_lecturer_specialization

Other restricted searchings are implemented by functions that support custom inputs.

Check GitHub for more details:

[https://github.com/duclm278/database-project/blob/main/sql/sms-view.sql]

6. Constraints & Triggers

6.1. Constraints

Suitable constraints beside primary and foreign keys are added while creating tables:

- constraint ck_lecturer_gender check (gender in ('F', 'M'))
- constraint ck_program_credit_price check ((credit_price >= 0))
- constraint ck_student_cpa_total_score_product check (cpa_total_score_product >= 0),
- constraint ck_student_cpa_total_study_credits check (cpa_total_study_credits >= 0),
- constraint ck_student_gpa_total_score_product check (gpa_total_score_product >= 0),
- constraint ck_student_gpa_total_study_credits check (gpa_total_study_credits >= 0),
- constraint ck_student_credit_debt check (credit_debt >= 0)
- constraint ck_student_study_credits check (study_credits >= 0),
- constraint ck_student_tuition_credits check (tuition_credits >= 0),
- constraint ck_subject_final_weight check (final_weight >= 0 and final_weight <= 1)
- constraint ck_require_lab_same_term check (require_lab in ('Y', 'N')),
- constraint ck_class_current_cap check (current_cap >= 0 and current_cap <= max_cap)
- constraint ck_timetable_weekday check (weekday in ('2', '3', '4', '5', '6', '7', '8')),
- constraint ck_timetable_start_time check (start_time < end_time)

- constraint ck_enrollment_midterm_score check (midterm_score >= 0 and midterm_score <= 10),
- constraint ck_enrollment_final_score check (final_score >= 0 and final_score <= 10),
- constraint ck_enrollment_absent_count check (absent_count >= 0)

Check GitHub for more details:

[https://github.com/duclm278/database-project/blob/main/sql/sms-create.sql]

6.2. Triggers

There are 10 triggers used to maintain the integrity of the data.

- timetable_conflict_trigger
- teaching_conflict_trigger
- convert_score_trigger
- update_score_trigger
- check_cap_trigger
- check_prerequisite_trigger
- check_lab_trigger
- check_time_enrolled_trigger
- credits_trigger
- company_id_trigger

Check GitHub for more details:

[https://github.com/duclm278/database-project/blob/main/sql/sms-staff.sql]

[https://github.com/duclm278/database-project/blob/main/sql/sms-student.sql]

7. Users

Beside the default admin user, the demo also creates 2 more users with suitable rights:

- Grant read-only privilege on all tables to student “20200164”
- Grant all privileges on table “enrollment” to student “20200164”
- Revoke privilege on table “student”, “lecturer”
- Instead grant all privileges on schema “student” (having self_view views, custom functions)
- Grant all privileges on schema “search” (having restricted searching views)

Similarly, grant suitable rights to the lecturer having id “aaaaaaaaaaa”.

Check GitHub for more details:

[https://github.com/duclm278/database-project/blob/main/sql/sms-demo.sql#L3-L37]

8. Results

Check GitHub for more details about the demo below:

[https://github.com/duclm278/database-project/blob/main/sql/sms-demo.sql]

2.1. Staff (Admin)

Login as "**postgres"**.

Check GitHub for more details about functions and triggers used for staff:

[https://github.com/duclm278/database-project/blob/main/sql/sms-staff.sql]

2.1.1. Adding students, lecturers, subjects, and classes with the following data.

```
CALL add_student('20205147', 'Tong Tran Minh', 'Duc', 'M', '2002-09-03',
true, '2022-02-04', '67 Le Thanh Nghi', 'duc.ttm205147@sis.hust.edu.vn',
'0902112042', '509463');
SELECT * FROM student WHERE id LIKE '20205147';
```

	id [PK] character (8)	first_name character varying (35)	last_name character varying (35)	gender character (1)	birthday date	status boolean	join_date date	address character varying (70)	email character varying
1	20205147	Tong Tran Minh	Duc	M	2002-09-03	true	2022-02-04	67 Le Thanh Nghi	duc.ttm205147@

Total rows: 1 of 1 | Query complete 00:00:00.088 | Ln 43, Col 1

```
CALL add_lecturer('aaaaaaaaaaaa', 'Tong Tran Minh', 'Duc', 'M',
'2002-09-03', true, '2022-02-04', '67 Le Thanh Nghi',
'duc.ttm205147@sis.hust.edu.vn', '0902112042', 'TCNTT');
SELECT * FROM lecturer WHERE id LIKE 'aaaaaaaaaaaa';
```

	id [PK] character (12)	first_name character varying (35)	last_name character varying (35)	gender character (1)	birthday date	status boolean	join_date date	address character varying (70)	email character varying
1	aaaaaaaaaaaa	Tong Tran Minh	Duc	M	2002-09-03	true	2022-02-04	67 Le Thanh Nghi	duc.ttm205147@

Total rows: 1 of 1 | Query complete 00:00:00.097 | Ln 48, Col 1

```
CALL add_subject('IT3333', 'Data Structure and Algorithm', 3, 4, 0.7,
'IT2000', 'TCNTT');
SELECT * FROM subject WHERE id LIKE 'IT3333';
```

	id [PK] character varying (7)	name character varying (100)	study_credits numeric	tuition_credits numeric	final_weight numeric (3,2)	prerequisite_id character varying (7)	faculty_id character varying (8)
1	IT3333	Data Structure and Algorithm	3	4	0.70	IT2000	TCNTT

Total rows: 1 of 1 | Query complete 00:00:00.095

✓ Successfully run. Total query runtime: 95 msec. 1 rows affected.

```
CALL add_class('555555', 'LT', '20201', 'N', 0, 80, NULL, NULL, 'IT3333');
SELECT * FROM class WHERE id LIKE '555555';
```

	id [PK] character (6)	type character varying (8)	semester character (5)	require_lab character (1)	current_cap integer	max_cap integer	company_id character (9)	lecturer_id character (12)	subject_id character varying (7)
1	555555	LT	20201	N	0	80	[null]	[null]	IT3333

Total rows: 1 of 1 | Query complete 00:00:00.087

✓ Successfully run. Total query runtime: 87 msec. 1 rows affected.

2.1.2. Managing information of students, lecturers, subjects, and classes.

```
UPDATE student SET first_name = 'Tong' WHERE id = '20205147';
SELECT * FROM student WHERE id LIKE '20205147';
```

	id [PK] character (8)	first_name character varying (35)	last_name character varying (35)	gender character (1)	birthday date	status boolean	join_date date	address character varying (70)	email character varying
1	20205147	Tong	Duc	M	2002-09-03	true	2022-02-04	67 Le Thanh Nghi	duc.ttm205147@

Total rows: 1 of 1 | Query complete 00:00:00.072

✓ Successfully run. Total query runtime: 72 msec. 1 rows affected.

2.1.3. Creating a tentative timetable for the upcoming semester, i.e creating classes for each subject. A subject can have 0, 1, or many classes open in a semester.

```
CALL add_timetable('555555', '3', '1800', '1900', 'D9-505');
SELECT * FROM timetable WHERE class_id LIKE '555555';
```

	class_id [PK] character (6)	weekday [PK] character (1)	start_time [PK] character (4)	end_time [PK] character (4)	location [PK] character varying (25)
1	555555	3	1800	1900	D9-505

Total rows: 1 of 1 | Query complete 00:00:00.095 | Successfully run. Total query runtime: 95 msec. 1 rows affected.

2.1.4. Making changes to the timetable.

```
UPDATE timetable
SET location = 'D9-510'
WHERE class_id = '555555';
SELECT * FROM timetable WHERE class_id LIKE '555555';
```

	class_id [PK] character (6)	weekday [PK] character (1)	start_time [PK] character (4)	end_time [PK] character (4)	location [PK] character varying (25)
1	555555	3	1800	1900	D9-510

Total rows: 1 of 1 | Query complete 00:00:00.067 | Successfully run. Total query runtime: 67 msec. 1 rows affected.

2.1.5. Assigning lecturers to classes based on the timetable.

```
CALL assign_lecturer('aaaaaaaaaaaa', '555555');
SELECT * FROM class WHERE lecturer_id LIKE 'aaaaaaaaaaaa';
```

	id [PK] character (6)	type character varying (8)	semester character (5)	require_lab character (1)	current_cap integer	max_cap integer	company_id character (9)	lecturer_id character (12)	subject_id character varying (7)
1	555555	LT	20201	N	0	80	[null]	aaaaaaaaaaaa	IT3333

Total rows: 1 of 1 | Query complete 00:00:00.090 | Successfully run. Total query runtime: 90 msec. 1 rows affected.

Test teaching conflict TRIGGER:

```
CALL add_class('666666', 'LT', '20201', 'N', 0, 80, NULL, NULL, 'IT3333');
CALL add_timetable('666666', '3', '1830', '1930', 'D9-505');
CALL assign_lecturer('aaaaaaaaaaaa', '666666');
```

```
ERROR:  Cannot execute. Teaching conflict found.
CONTEXT:  PL/pgSQL function check_teaching_conflict() line 35 at RAISE
SQL statement "UPDATE class
    SET lecturer_id = i_lecturer_id
    WHERE id = i_class_id"
PL/pgSQL function assign_lecturer(character,character) line 3 at SQL statement
SQL state: P0001
```

Total rows: 1 of 1 | Query complete 00:00:00.117 | Ln 88, Col 1

2.1.6. Getting reports on

a. Classes having a number of enrolled students fewer than [X], used for canceling classes with low enrollment.

Update the current_cap of the class "124090" to have the status "Enough students"

```
SELECT * FROM report_enrolled('20212') WHERE id = '134090';
UPDATE class SET current_cap = max_cap / 2 WHERE id = '134090';
SELECT * FROM report_enrolled('20212') WHERE id = '134090';
```

BEFORE:

	id character	type character varying	semester character	require_lab character	current_cap integer	max_cap integer	company_id character	lecturer_id character	subject_id character varying	class_status text
1	134090	LT+BT	20212	N	0	40	[null]	ClBXeIgjyZFy	AC2060	Not enough students

Total rows: 1 of 1 | Query complete 00:00:00.069

✓ Successfully run. Total query runtime: 69 msec. 1 rows affected. ×

AFTER:

	id character	type character varying	semester character	require_lab character	current_cap integer	max_cap integer	company_id character	lecturer_id character	subject_id character varying	class_status text
1	134090	LT+BT	20212	N	20	40	[null]	ClBXeIgjyZFy	AC2060	Enough students

Total rows: 1 of 1 | Query complete 00:00:00.075

✓ Successfully run. Total query runtime: 75 msec. 1 rows affected. ×

b. Students with credit debt from unfinished (failed) subjects in the range [A, B], used for sending warnings.

```
SELECT * FROM report_credit_debt();
```

	id character	first_name character varying	last_name character varying	status boolean	credit_debt numeric	warning_level text
1	20200001	Angela	Cohen	false	13	Expelled
2	20200003	Joshua	Jordan	false	13	Expelled
3	20200002	Wendy	Oliver	true	11	Warning level 3
4	20200005	Laurie	White	false	7	Warning level 2
5	20200006	Joseph	Jordan	false	13	Expelled
6	20200007	Richard	Pena	true	8	Warning level 2
7	20200008	Yvette	Daniels	false	12	Warning level 3
8	20200009	Ruth	Hernandez	false	3	Warning level 1

Total rows: 1000 of 15001 | Query complete 00:00:00.085

Successfully run. Total query runtime: 85 msec. 15001 rows affected.

c. Students qualified for semester scholarships with GPA ≥ [G].

Make the student 20200164 enroll in some classes and get all A+.

```
CALL student.enroll_class('20200164', '135387'); -- enroll_class() in
2.2.3f.
CALL student.enroll_class('20200164', '135404');
CALL student.enroll_class('20200164', '135406');
CALL student.enroll_class('20200164', '135409');
CALL student.enroll_class('20200164', '135410');
CALL student.enroll_class('20200164', '135411');
CALL student.enroll_class('20200164', '135469');
CALL student.enroll_class('20200164', '722873');
UPDATE enrollment SET midterm_score = 10, final_score = 10 WHERE student_id
= '20200164'
    AND class_id IN ('135387', '135404', '135406', '135409', '135410',
'135411', '135469', '722873');

SELECT * FROM report_scholarship() WHERE o_id = '20200164';
```

	o_id character	o_first_name character varying	o_last_name character varying	o_gpa numeric	o_scholarship_status text
1	20200164	Diana	Brown	4.0	Scholarship A

Total rows: 1 of 1 | Query complete 00:00:00.106

Successfully run. Total query runtime: 106 msec. 1 rows affected.

2.2. Students

Login as "**20200164"**. Pass is "**demo**".

Check GitHub for more details about functions and triggers used for students:

[https://github.com/duclm278/database-project/blob/main/sql/sms-student.sql]

2.2.1. Viewing data of subjects, classes, and results of themselves.

```sql
-- View all subjects of their curriculum
SELECT * FROM student.self_view_curriculum; -- As 20200164
```

	id character varying (7)	name character varying (100)	study_credits numeric	tuition_credits numeric	final_weight numeric (3,2)	prerequisite_id character varying (7)	faculty_id character varying (8
1	EM1014	Quản trị học	2	4.5	0.70	[null]	KKTVQL
2	EM1100	Kinh tế vi mô đại cương	3	4.5	0.70	[null]	KKTVQL
3	EM1180	Văn hóa kinh doanh và tinh thần khởi nghiệp	2	3	0.60	[null]	KKTVQL
4	EM1322	Academic Writing and Presentation	3	4.5	0.60	[null]	KKTVQL
5	EM1422	Academic Writing and Presentation	3	4	0.70	EM1014	KKTVQL
6	EM1722	Academic Writing and Presentation	3	5	0.50	[null]	KKTVQL
7	EM2104	Quản trị doanh nghiệp	2	4	0.70	[null]	KKTVQL
8	EM2131	Toán ứng dụng trong kinh doanh và thương mại					

✓ Successfully run. Total query runtime: 108 msec. 52 rows affected. ×

Total rows: 52 of 52 | Query complete 00:00:00.108 | Ln 132, Col 1

```sql
-- View their classes in of any semesters
SELECT * FROM student.self_view_class_enrolled('20221'); -- As 20200164
```

	id character	type character varying	semester character	require_lab character	current_cap integer	max_cap integer	company_id character	lecturer_id character	subject_id character varying
1	135387	LT+BT	20221	N	1	123	[null]	VYxFdcFNsgcZ	IT4542E
2	135404	LT+BT	20221	N	1	80	[null]	VYxFdcFNsgcZ	IT4593E
3	135406	LT+BT	20221	N	1	80	[null]	FydMcQmfXM...	IT4172E
4	135409	BT	20221	N	1	42	[null]	fmXzwBlFBfGw	IT3290E
5	135410	LT+BT	20221	N	1	80	[null]	jVpHCeHteTaB	IT3283E
6	135411	BT	20221	N	1	40	[null]	HJqwnzBHtPFB	IT3280E
7	135469	?A	20221	N	1	90	[null]	[null]	IT5023E
8	722873	LT+BT	20221	N	1	5			

✓ Successfully run. Total query runtime: 76 msec. 8 rows affected. ×

Total rows: 8 of 8 | Query complete 00:00:00.076 | Ln 137, Col 1

```sql
-- View their results in of any semesters
SELECT * FROM student.self_view_results('20221'); -- As 20200164
```

	class_id character	subject_id character varying	subject_name character varying	midterm_score numeric	final_score numeric	mapping_score numeric
1	135387	IT4542E	Management of Software Development	10	10	4
2	135404	IT4593E	Introduction to Communication Engineering	10	10	4
3	135406	IT4172E	Signal processing	10	10	4
4	135409	IT3290E	Database Lab	10	10	4
5	135410	IT3283E	Computer Architecture	10	10	4
6	135411	IT3280E	Assembly Language and Computer Architecture Lab	10	10	4
7	135469	IT5023E	Graduation Research 1	10	10	4
8	722873	PE2401	Bóng bàn 1			

✓ Successfully run. Total query runtime: 81 msec. 8 rows affected. ×

Total rows: 8 of 8 | Query complete 00:00:00.081 | Ln 142, Col 1

```sql
-- View their timetable in of a semester
SELECT * FROM student.self_view_timetable('20221');
```

	class_id character	subject_id character varying	subject_name character varying	type character varying	weekday character	start_time character	end_time character	location character varying
1	135387	IT4542E	Management of Software Development	LT+BT	6	0645	0815	D9-401
2	135404	IT4593E	Introduction to Communication Engineering	LT+BT	3	0645	0910	D9-407
3	135406	IT4172E	Signal processing	LT+BT	6	0920	1145	D9-407
4	135409	IT3290E	Database Lab	BT	5	1410	1730	B1-302
5	135410	IT3283E	Computer Architecture	LT+BT	2	1230	1455	D9-505
6	135411	IT3280E	Assembly Language and Computer Architecture Lab	BT	5	1015	1145	B1-303
7	135411	IT3280E	Assembly Language and Computer Architecture Lab	BT	5	1230	1400	B1-303
8	722873	PE2401	Bóng bàn 1	LT+BT	2	1530	1630	NTD

Total rows: 8 of 8 | Query complete 00:00:00.084 | Ln 146, Col 1

2.2.2. View the tentative timetable to find suitable opening classes.

```
SELECT * FROM student.show_class_info(NULL, '20221'); -- All classes in
20221
```

	class_id character	subject_id character varying	subject_name character varying	prerequisite_id character varying	type character varying	require_lab character	study_c numeric
1	137978	AC2010	Kỹ thuật lập trình	[null]	LT+BT	N	
2	137982	AC2020	Đồ họa hình động 2D, 3D	[null]	LT+BT	N	
3	137983	AC2040	Cơ sở dữ liệu	[null]	LT+BT	N	
4	137984	AC2050	Cấu trúc dữ liệu và giải thuật	AC2020	LT+BT	N	
5	137987	AC3010	Phân tích và thiết kế hệ thống	[null]	LT+BT	N	
6	137986	AC3020	Trò chơi số và tương tác II	[null]	LT+BT	N	
7	137996	AC4010	Thực tại ảo	[null]	LT+BT	N	
8	137997	AC4020	Thực tại tăng cường				

✓ Successfully run. Total query runtime: 101 msec. 8798 rows affected. ×

Total rows: 1000 of 8798 | Query complete 00:00:00.101 | Ln 150, Col 1

2.2.3. Enrolling in classes.

-- Done in 2.1.6c.

a. Showing enrolling information such as time, location, credit, slot availability, and prerequisites.

-- Done in 2.2.2.

b. Check for slot availability, and class prerequisites.

-- Use TRIGGER check_cap_trigger BEFORE INSERT ON enrollment
-- Use TRIGGER check_prerequisite_trigger BEFORE INSERT ON enrollment
-- Use TRIGGER check_lab_trigger BEFORE INSERT ON enrollment

c. Check for time conflicts.

-- Use TRIGGER check_cap_trigger BEFORE INSERT ON enrollment
-- Use TRIGGER check_prerequisite_trigger BEFORE INSERT ON enrollment
-- Use TRIGGER check_lab_trigger BEFORE INSERT ON enrollment

d. Calculate total studying credits.

```
SELECT * FROM student.show_credits_enrolled(); -- As 20200164
```

	show_credits_enrolled numeric
1	14

Successfully run. Total query runtime: 72 msec. 1 rows affected.

Total rows: 1 of 1 | Query complete 00:00:00.072 | Ln 167, Col 1

e. Schedules are automatically identified after students enroll in classes.

```
SELECT * FROM student.self_view_timetable('20221'); -- As 20200164
```

	class_id character	subject_id character varying	subject_name character varying	type character varying	weekday character	start_time character	end_time character	location character varying
1	135387	IT4542E	Management of Software Development	LT+BT	6	0645	0815	D9-401
2	135404	IT4593E	Introduction to Communication Enginieering	LT+BT	3	0645	0910	D9-407
3	135406	IT4172E	Signal processing	LT+BT	6	0920	1145	D9-407
4	135409	IT3290E	Database Lab	BT	5	1410	1730	B1-302
5	135410	IT3283E	Computer Architecture	LT+BT	2	1230	1455	D9-505
6	135411	IT3280E	Assembly Language and Computer Architecture Lab	BT	5	1015	1145	B1-303
7	135411	IT3280E	Assembly Language and Computer Architecture Lab	BT	5	1230	1400	B1-303
8	722873	PE2401	Bóng bàn 1	LT+BT	2	1530	1630	NTD

Total rows: 8 of 8 | Query complete 00:00:00.084 | Ln 146, Col 1

2.2.4. Looking up data related to lecturers, subjects, classes, and other students. Retrieve only essential information, excluding personal ones like address, scores, and student timetable.

a. Students: Name, ID, School/Faculty, Program, Contact, etc.

```
SELECT * FROM search.search_student_by_id('20200164');
```

	id character	first_name character varying	last_name character varying	gender character	status boolean	email character varying	program_code character varying	program_name character varying	faculty_name character varyi
1	20200164	Diana	Brown	M	true	andersonsteven@example.com	EM3	Quản trị kinh doanh	Viện Kinh tế &

Successfully run. Total query runtime: 74 msec. 1 rows affected.

Total rows: 1 of 1 | Query complete 00:00:00.074 | Ln 182, Col 1

```
SELECT * FROM search.search_student_by_name('Robinson');
```

	id character	first_name character varying	last_name character varying	gender character	status boolean	email character varying	program_code character varying	program_name character varying
1	20210559	Mary	Robinson	F	false	guerrafrank@example.net	TX1	Kỹ thuật Dệt - May
2	20201000	Kendra	Robinson	M	false	elizabethramirez@example.net	CH1	Kỹ thuật Hóa học
3	20201122	Ashley	Robinson	F	false	justin70@example.net	CH1	Kỹ thuật Hóa học
4	20204271	Amanda	Robinson	M	false	kreed@example.net	CH1	Kỹ thuật Hóa học
5	20212544	Mary	Robinson	M	false	kaitlinhayden@example.net	CH1	Kỹ thuật Hóa học
6	20202135	Kathleen	Robinson	M	false	kristinamitchell@example.org	CH2	Hóa học
7	20204414	Andrew	Robinson	F	true	yhoward@example.org	CH2	Hóa học
8	20210394	Makayla	Robinson	F	false	kyle0		

✓ Successfully run. Total query runtime: 57 msec. 83 rows affected. ×

Total rows: 83 of 83 | Query complete 00:00:00.057 | Ln 183, Col 1

b. Lecturers: Name, ID, School/Faculty, Teaching Subjects, Contact, etc.

```
SELECT * FROM search.search_lecturer_by_id('aaaaaaaaaaaa');
```

	id character	first_name character varying	last_name character varying	gender character	status boolean	email character varying	faculty_name character varying
1	aaaaaaaaaaaa	Tong Tran Minh	Duc	M	true	duc.ttm205147@sis.hust.edu.vn	Trường Công nghệ Thông tin và Truyền thông

✓ Successfully run. Total query runtime: 60 msec. 1 rows affected. ×

Total rows: 1 of 1 | Query complete 00:00:00.060 | Ln 189, Col 1

```
SELECT * FROM search.search_lecturer_by_name('Robinson');
```

	id character	first_name character varying	last_name character varying	gender character	status boolean	email character varying	faculty_name character varying
1	ljIABLFFxXPO	Cynthia	Robinson	F	true	jessica77@example.org	Viện Công nghệ Sinh học và công nghệ Thực phẩm
2	OFyschYnXK...	Jessica	Robinson	F	true	annwest@example.net	Viện Kỹ thuật Hoá học
3	nGVJiyvhJM...	Jessica	Robinson	M	true	kjohnson@example.com	Trường Cơ Khí
4	FENhSULEdg...	Mark	Robinson	M	false	townsendwilliam@example.com	Trường Cơ Khí
5	mbNQtVaCA...	Nicole	Robinson	M	true	emccarthy@example.net	Viện Khoa học và Kỹ thuật Vật liệu
6	dustwHpezhjV	Connor	Robinson	F	true	martinduncan@example.com	Viện Vật lý kỹ thuật
7	gzzxYAkuRqrl	Elizabeth	Robinson	M	false	danny67@example.com	Viện Vật lý kỹ thuật

✓ Successfully run. Total query runtime: 87 msec. 7 rows affected. ×

Total rows: 7 of 7 | Query complete 00:00:00.087 | Ln 190, Col 1

```
SELECT * FROM search.search_lecturer_specialization_by_id('ljIABLFFxXPO');
```

	id character	first_name character varying	last_name character varying	subject_id character varying	subject_name character varying
1	ljIABLFFxXPO	Cynthia	Robinson	BF3508	Thí nghiệm hóa sinh
2	ljIABLFFxXPO	Cynthia	Robinson	BF3509	Vi sinh vật thực phẩm
3	ljIABLFFxXPO	Cynthia	Robinson	BF3513	Công nghệ thực phẩm đại cương
4	ljIABLFFxXPO	Cynthia	Robinson	BF3515	An toàn thực phẩm
5	ljIABLFFxXPO	Cynthia	Robinson	BF3522	Vật lý học Thực phẩm

✓ Successfully run. Total query runtime: 105 msec. 5 rows affected. ×

Total rows: 5 of 5 | Query complete 00:00:00.105 | Ln 195, Col 1

```
SELECT * FROM search.search_lecturer_specialization_by_name('Robinson');
```

	id character	first_name character varying	last_name character varying	subject_id character varying	subject_name character varying
1	IjIABLFFxXPO	Cynthia	Robinson	BF3508	Thí nghiệm hóa sinh
2	IjIABLFFxXPO	Cynthia	Robinson	BF3509	Vi sinh vật thực phẩm
3	IjIABLFFxXPO	Cynthia	Robinson	BF3513	Công nghệ thực phẩm đại cương
4	IjIABLFFxXPO	Cynthia	Robinson	BF3515	An toàn thực phẩm
5	IjIABLFFxXPO	Cynthia	Robinson	BF3522	Vật lý học Thực phẩm
6	OFyschYnXK...	Jessica	Robinson	CH3130	TN Hóa vô cơ
7	OFyschYnXK...	Jessica	Robinson	CH3131	TN Hóa vô cơ
8	OFyschYnXK...	Jessica	Robinson	CH3202	Hóa Hữu c
9	OFyschYnXK...	Jessica	Robinson	CH3220	Hóa hữu c

Successfully run. Total query runtime: 69 msec. 35 rows affected.

Total rows: 35 of 35 | Query complete 00:00:00.069 | Ln 196, Col 1

c. Subjects, Classes: Number of credits, Schedule of classes, etc.

-- Done in 2.2.1.

2.2.5. Getting estimated fees for the current semester.

```
SELECT * FROM student.show_estimated_fees(); -- As 20200164
```

	o_total_fees numeric
1	6720000

Successfully run. Total query runtime: 76 msec. 1 rows affected.

Total rows: 1 of 1 | Query complete 00:00:00.076 | Ln 207, Col 1

2.2.6. Getting reports on Study Credits earned, GPA/CPA.

```
SELECT * FROM student.report_student(); -- As 20200164
```

	o_cpa_total_study_credits numeric	o_cpa numeric	o_gpa_total_study_credits numeric	o_gpa numeric
1	34.0	2.3	14.0	4.0

Successfully run. Total query runtime: 81 msec. 1 rows affected.

Total rows: 1 of 1 | Query complete 00:00:00.081 | Ln 212, Col 1

2.3. Lecturers

Login as "**aaaaaaaaaaa**". Pass is "**demo**".

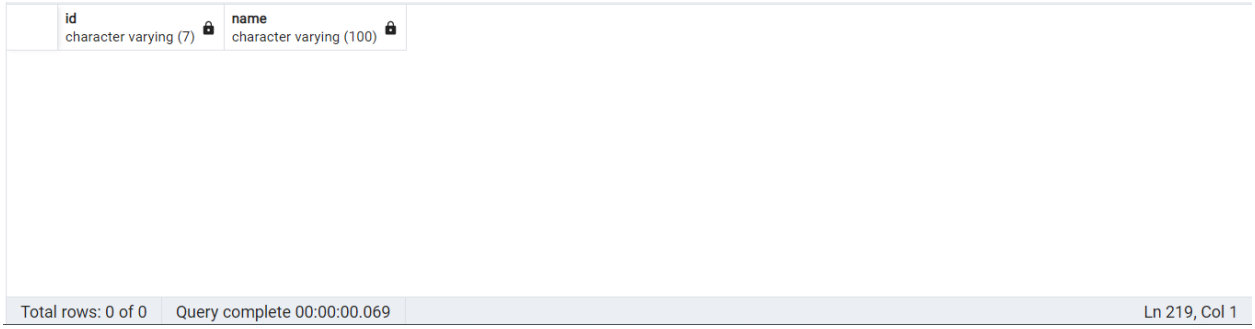
Check GitHub for more details about functions and triggers used for lecturers:

[https://github.com/duclm278/database-project/blob/main/sql/sms-lecturer.sql]

2.3.1. Viewing data of all teaching subjects, classes, and results of themselves.

```
-- View all subjects of their specializations
SELECT * FROM lecturer.self_view_specializations;
```

-- Lecturer "aaaaaaaaaaaa" has been just created above and hasn't had any specializations yet.

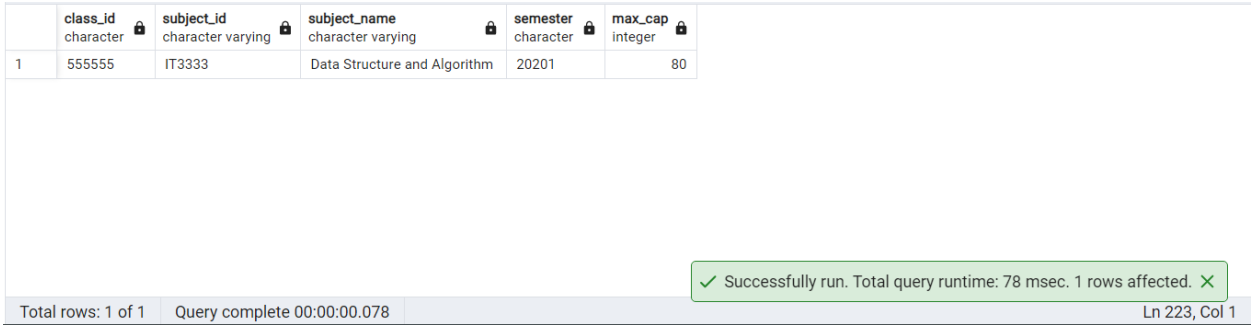


	id character varying (7)	name character varying (100)

Total rows: 0 of 0 | Query complete 00:00:00.069 | Ln 219, Col 1

-- Lecturer "aaaaaaaaaaaa" just has been assigned class "555555" in 2.1.

```
-- View all classes of their teachings of any semesters
SELECT * FROM lecturer.self_view_class_assigned('20201');
```



	class_id character	subject_id character varying	subject_name character varying	semester character	max_cap integer
1	555555	IT3333	Data Structure and Algorithm	20201	80

Successfully run. Total query runtime: 78 msec. 1 rows affected.

Total rows: 1 of 1 | Query complete 00:00:00.078 | Ln 223, Col 1

2.3.2. Looking up data related to students, subjects, classes, and other lecturers. Retrieve only essential information, excluding personal ones like address, scores, and student timetable.

a. Students: Name, ID, School/Faculty, Program, Contact, etc.

-- Done in 2.2.4a.

b. Lecturers: Name, ID, School/Faculty, Teaching Subjects, Contact, etc.

-- Done in 2.2.4b.

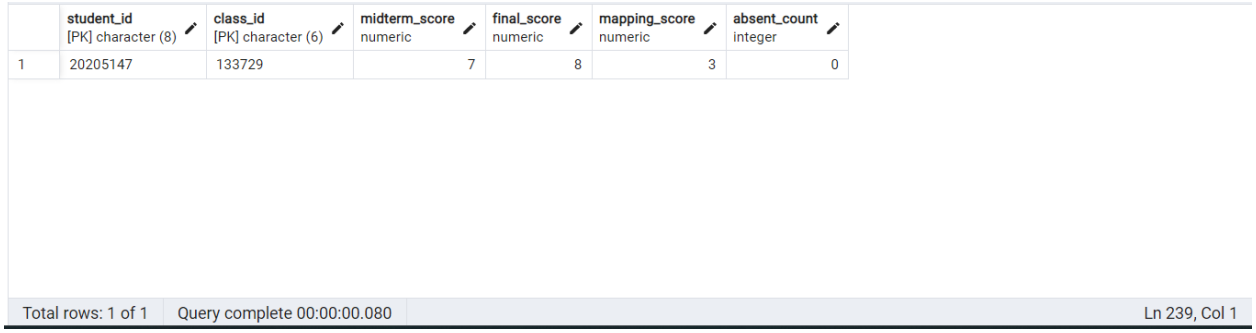
c. Subjects, Classes: Number of credits, Schedule of classes, etc.

-- Done in 2.2.1.

2.3.3. Recording student academic performance.

```
CALL student.enroll_class('20205147', '133729'); -- enroll_class() in
2.2.3f.
CALL lecturer.update_grade('20205147', '133729', 7, 8);
SELECT * FROM enrollment WHERE student_id = '20205147' AND class_id =
```

```
'133729';
```



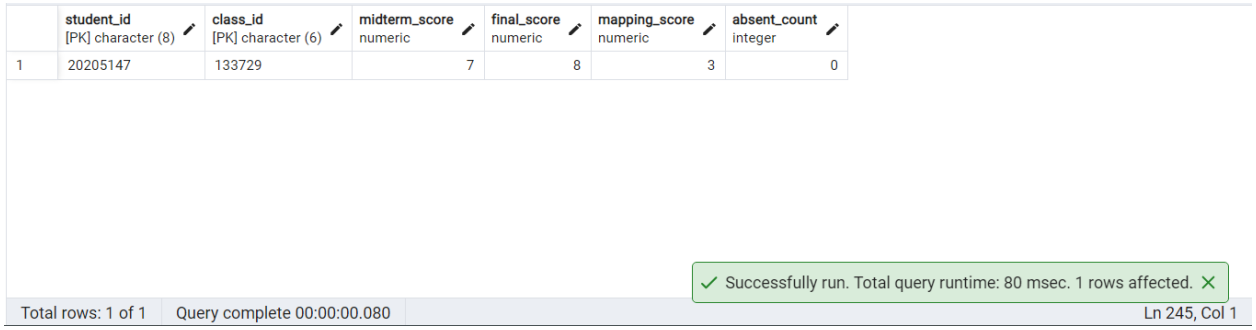
	student_id [PK] character (8)	class_id [PK] character (6)	midterm_score numeric	final_score numeric	mapping_score numeric	absent_count integer
1	20205147	133729	7	8	3	0

Total rows: 1 of 1 | Query complete 00:00:00.080 | Ln 239, Col 1

2.3.4. Tracking student attendance.

-- Before using PROCEDURE mark_absence()

```
SELECT * FROM enrollment WHERE student_id = '20205147' AND class_id =
'133729';
```



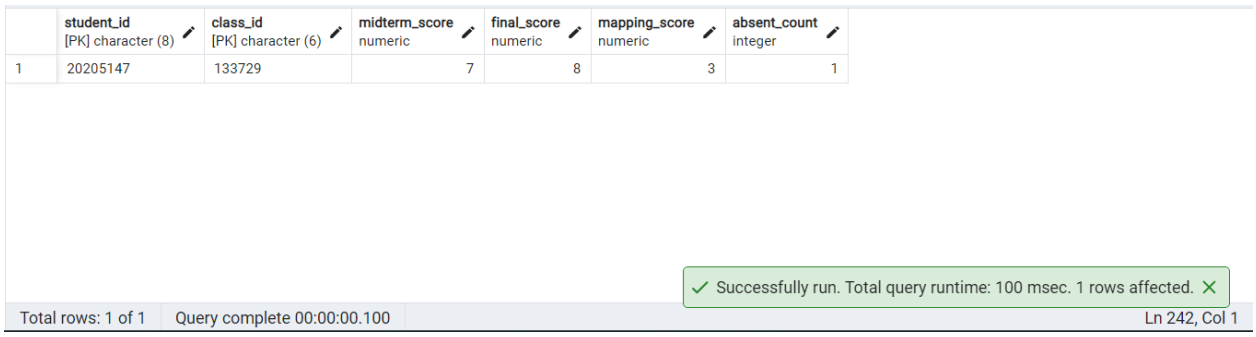
	student_id [PK] character (8)	class_id [PK] character (6)	midterm_score numeric	final_score numeric	mapping_score numeric	absent_count integer
1	20205147	133729	7	8	3	0

Successfully run. Total query runtime: 80 msec. 1 rows affected.

Total rows: 1 of 1 | Query complete 00:00:00.080 | Ln 245, Col 1

-- Use PROCEDURE mark_absence() to mark absence of student

```
CALL lecturer.mark_absence('20205147', '133729');
SELECT * FROM enrollment WHERE student_id = '20205147' AND class_id =
'133729';
```



	student_id [PK] character (8)	class_id [PK] character (6)	midterm_score numeric	final_score numeric	mapping_score numeric	absent_count integer
1	20205147	133729	7	8	3	1

Successfully run. Total query runtime: 100 msec. 1 rows affected.

Total rows: 1 of 1 | Query complete 00:00:00.100 | Ln 242, Col 1

-- Use PROCEDURE undo_absence() to undo absence of student

```
CALL lecturer.undo_absence('20205147', '133729');
SELECT * FROM enrollment WHERE student_id = '20205147' AND class_id =
'133729';
```

	student_id [PK] character (8)	class_id [PK] character (6)	midterm_score numeric	final_score numeric	mapping_score numeric	absent_count integer
1	20205147	133729	7	8	3	0

Successfully run. Total query runtime: 80 msec. 1 rows affected.

Total rows: 1 of 1 | Query complete 00:00:00.080 | Ln 245, Col 1

2.3.5. Getting reports on

a. Exam grade distribution (statistics)

```
SELECT * FROM lecturer.report_grade_distribution('133729');
```

	mapping_score numeric	count bigint
1	3.0	3
2	2.5	1
3	2.0	3
4	1.0	8
5	0.0	9

Successfully run. Total query runtime: 70 msec. 5 rows affected.

Total rows: 5 of 5 | Query complete 00:00:00.070 | Ln 251, Col 1

b. Student attendance up to today

```
SELECT * FROM lecturer.report_attendance('133729');
```

	student_id character	absent_count integer
1	20200110	2
2	20200448	7
3	20201039	3
4	20201308	4
5	20201966	5
6	20202315	6
7	20203477	1
8	20203809	8
9	20204232	7

Successfully run. Total query runtime: 88 msec. 24 rows affected.

Total rows: 24 of 24 | Query complete 00:00:00.088 | Ln 255, Col 1

9. Optimizations

Check GitHub for more details about experiments:

[https://github.com/duclm278/database-project/blob/main/sql/sms-index.sql]

1. Create necessary indexes for searching

```
EXPLAIN
SELECT c.* FROM class c
WHERE c.semester = '20212';
```

BEFORE: Seq scan on class, **149ms**

	QUERY PLAN text
1	Seq Scan on class c (cost=0.00..297.83 rows=5254 width=57)
2	Filter: (semester = '20212'::bpchar)

Total rows: 2 of 2 | Query complete 00:00:00.050 | Ln 63, Col 1

	id [PK] character (6)	type character varying (8)	semester character (5)	require_lab character (1)	current_cap integer	max_cap integer	company_id character (9)	lecturer_id character (12)	subject_id character varying (7)
1	134090	LT+BT	20212	N	0	40	[null]	CIBXeIgjyZFy	AC2060
2	134092	LT+BT	20212	N	0	40	[null]	CIBXeIgjyZFy	AC2080
3	134099	LT+BT	20212	N	0	25	[null]	CIBXeIgjyZFy	AC3030
4	134100	LT+BT	20212	N	0	25	[null]	ctTWcAazbCTD	AC3040
5	131622	LT+BT	20212	Y	0	55	[null]	whnENplJHrge	BF1016
6	717470	TN	20212	N	0	18	[null]	whnENplJHrge	BF1016
7	717471	TN	20212	N	0	18	[null]	whnENplJHrge	BF1016
8	133705	LT+BT	20212	N	0	40	[null]	whnENplJHrge	BF2020

Total rows: 1000 of 5254 | Query complete 00:00:00.149

Successfully run. Total query runtime: 149 msec. 5254 rows affected.

AFTER: Used class_semster, bitmap index scan, **82ms**

	QUERY PLAN text
1	Bitmap Heap Scan on class c (cost=65.00..268.68 rows=5254 width=57)
2	Recheck Cond: (semester = '20212'::bpchar)
3	-> Bitmap Index Scan on class_semester (cost=0.00..63.69 rows=5254 width=0)
4	Index Cond: (semester = '20212'::bpchar)

Total rows: 4 of 4 | Query complete 00:00:00.055

Successfully run. Total query runtime: 55 msec. 4 rows affected.

	id [PK] character (6)	type character varying (8)	semester character (5)	require_lab character (1)	current_cap integer	max_cap integer	company_id character (9)	lecturer_id character (12)	subject_id character varying (7)
1	134090	LT+BT	20212	N	0	40	[null]	CIBXeIgjyZFy	AC2060
2	134092	LT+BT	20212	N	0	40	[null]	CIBXeIgjyZFy	AC2080
3	134099	LT+BT	20212	N	0	25	[null]	CIBXeIgjyZFy	AC3030
4	134100	LT+BT	20212	N	0	25	[null]	ctTWcAazbCTD	AC3040
5	131622	LT+BT	20212	Y	0	55	[null]	whnENplJHrge	BF1016
6	717470	TN	20212	N	0	18	[null]	whnENplJHrge	BF1016
7	717471	TN	20212	N	0	18	[null]	whnENplJHrge	BF1016
8	133705	LT+BT	20212	N	0	40	[null]	whnENplJHrge	BF2020

Total rows: 1000 of 5254 | Query complete 00:00:00.082

Successfully run. Total query runtime: 82 msec. 5254 rows affected.

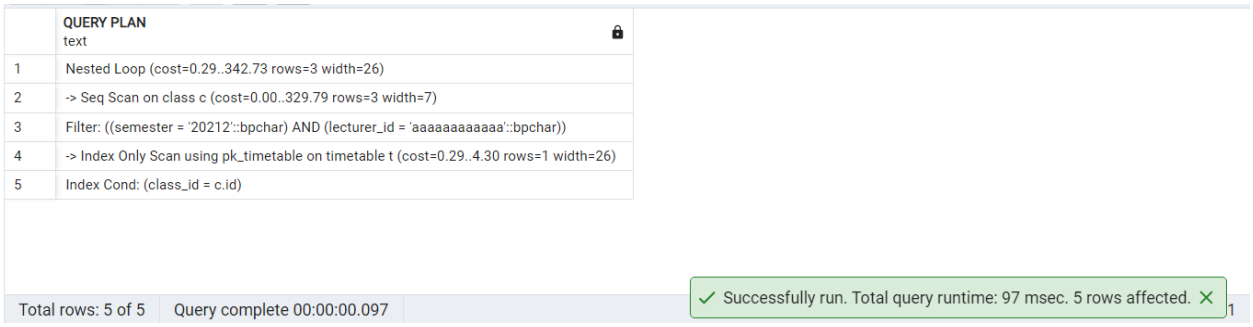
2. Create necessary indexes on non-prime attributes while joining

```
EXPLAIN
SELECT t.*
FROM timetable t
JOIN class c ON t.class_id = c.id
```

```
WHERE c.lecturer_id = 'aaaaaaaaaaaa'
AND c.semester = '20212';

DROP INDEX IF EXISTS class_lecturer_id;
CREATE INDEX class_lecturer_id ON class (lecturer_id);
```

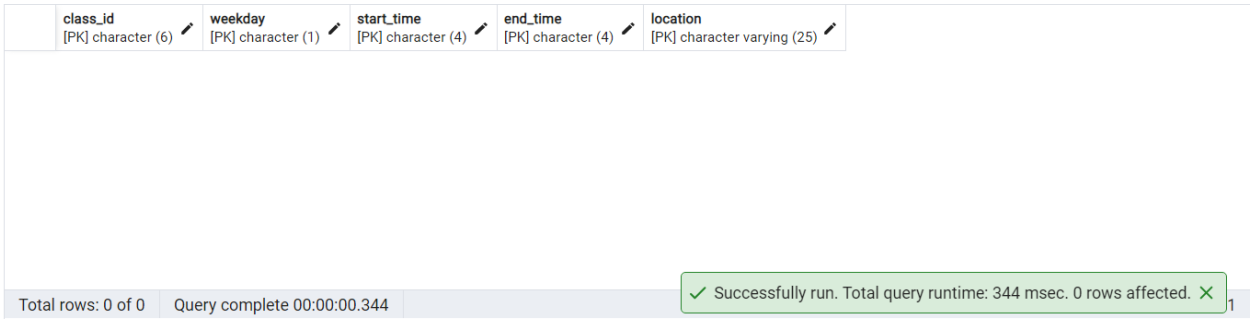
BEFORE: Seq scan on class, **334ms**



	QUERY PLAN text
1	Nested Loop (cost=0.29..342.73 rows=3 width=26)
2	-> Seq Scan on class c (cost=0.00..329.79 rows=3 width=7)
3	Filter: ((semester = '20212'::bpchar) AND (lecturer_id = 'aaaaaaaaaaaa'::bpchar))
4	-> Index Only Scan using pk_timetable on timetable t (cost=0.29..4.30 rows=1 width=26)
5	Index Cond: (class_id = c.id)

Total rows: 5 of 5 | Query complete 00:00:00.097

✓ Successfully run. Total query runtime: 97 msec. 5 rows affected. ×



	class_id [PK] character (6)	weekday [PK] character (1)	start_time [PK] character (4)	end_time [PK] character (4)	location [PK] character varying (25)

Total rows: 0 of 0 | Query complete 00:00:00.344

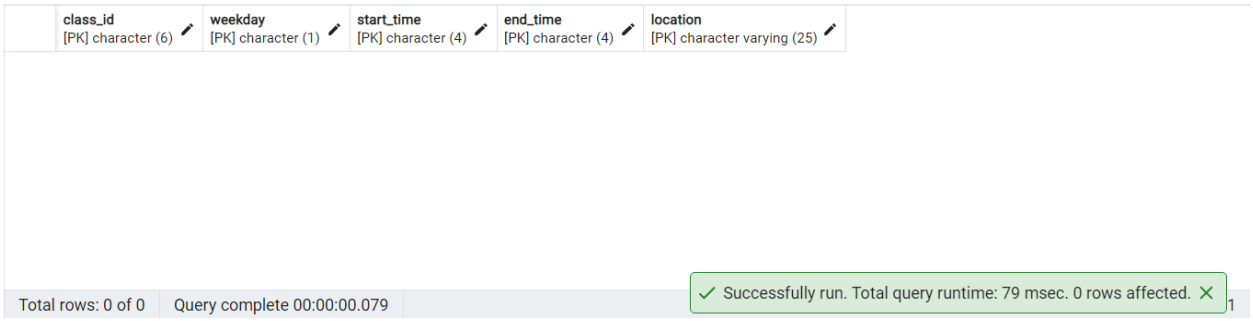
✓ Successfully run. Total query runtime: 344 msec. 0 rows affected. ×

AFTER: Used class_lecturer_id, bitmap index scan, **79ms**

	QUERY PLAN text
1	Nested Loop (cost=4.63..43.62 rows=3 width=26)
2	-> Bitmap Heap Scan on class c (cost=4.35..30.69 rows=3 width=7)
3	Recheck Cond: (lecturer_id = 'aaaaaaaaaaaa'::bpchar)
4	Filter: (semester = '20212'::bpchar)
5	-> Bitmap Index Scan on class_lecturer_id (cost=0.00..4.34 rows=8 width=0)
6	Index Cond: (lecturer_id = 'aaaaaaaaaaaa'::bpchar)
7	-> Index Only Scan using pk_timetable on timetable t (cost=0.29..4.30 rows=1 width=26)
8	Index Cond: (class_id = c.id)

Total rows: 8 of 8 | Query complete 00:00:00.059

✓ Successfully run. Total query runtime: 59 msec. 8 rows affected. ×



	class_id [PK] character (6)	weekday [PK] character (1)	start_time [PK] character (4)	end_time [PK] character (4)	location [PK] character varying (25)

Total rows: 0 of 0 | Query complete 00:00:00.079

✓ Successfully run. Total query runtime: 79 msec. 0 rows affected. ×

3. Do not create indexes for large varchar, but create indexes for char

Searching by id is fast as it uses primary keys of type char.

Searching by name is slow as varchar is too big to be indexed and "%" is used to match partially.

```
SELECT * FROM search.view_search_student v
WHERE v.first_name || ' ' || v.last_name LIKE '%' || i_student_name || '%';
```

Sources:

[https://stackoverflow.com/questions/8001905/sql-server-worth-indexing-large-string-keys]

[https://stackoverflow.com/questions/1388059/sql-server-index-columns-used-in-like]

4. Do not create indexes for fully joining

```
EXPLAIN
SELECT c.semester, t.* FROM timetable t
JOIN class c ON c.id = t.class_id;
```

BEFORE: Seq scan, hash join

	QUERY PLAN text
1	Hash Join (cost=425.69..723.73 rows=14894 width=32)
2	Hash Cond: (t.class_id = c.id)
3	-> Seq Scan on timetable t (cost=0.00..258.94 rows=14894 width=26)
4	-> Hash (cost=265.86..265.86 rows=12786 width=13)
5	-> Seq Scan on class c (cost=0.00..265.86 rows=12786 width=13)

Total rows: 5 of 5 | Query complete 00:00:00.102

✓ Successfully run. Total query runtime: 102 msec. 5 rows affected. ×

AFTER: Not used due to full table scan.

	QUERY PLAN text
1	Hash Join (cost=425.69..723.73 rows=14894 width=32)
2	Hash Cond: (t.class_id = c.id)
3	-> Seq Scan on timetable t (cost=0.00..258.94 rows=14894 width=26)
4	-> Hash (cost=265.86..265.86 rows=12786 width=13)
5	-> Seq Scan on class c (cost=0.00..265.86 rows=12786 width=13)

Total rows: 5 of 5 | Query complete 00:00:00.096

✓ Successfully run. Total query runtime: 96 msec. 5 rows affected. ×

5. Do not create indexes for table having few rows

-- E.g. Querying from or joining with self_view_info doesn't need indexes as it only stores one record per user!

10. Conclusion

Conclusion about this project:

- Searching and joining are fast as the system works mainly with primary keys.
- Triggers and custom views are very powerful as the system needs complex checking and restrictions.

Insights gained from this project:

- See how triggers and views are applied in practice.
- Make use of variables and triggers for fast querying and maintaining data.
- The ERD process helps a lot with the design of the database.
- Experience when working with many entities and relations involed.