

**Національний технічний університет України  
“Київський політехнічний інститут ім. Ігоря Сікорського”**

**Факультет прикладної математики  
Кафедра системного програмування і спеціалізованих  
комп’ютерних систем**

**РОЗРАХУНКОВО-ГРАФІЧНА РОБОТА**

з дисципліни  
“Бази даних та засоби управління”

ТЕМА: “Створення додатку бази даних, орієнтованого на взаємодію з  
СУБД PostgreSQL”

Виконав: студент 3 курсу ФПМ групи КВ-21  
Гуманіцький Андрій (5 варіант)

Перевірив(-ла):

Оцінка:

**Київ – 2024**

**Мета:** здобуття вмінь програмування прикладних додатків баз даних PostgreSQL.

**Завдання:**

1. Реалізувати функції перегляду, внесення, редагування та видалення даних у таблицях бази даних, створених у лабораторній роботі №1, засобами консольного інтерфейсу.
2. Передбачити автоматичне пакетне генерування «рандомізованих» даних у базі.
3. Забезпечити реалізацію пошуку за декількома атрибутами з двох та більше сутностей одночасно: для числових атрибутів – у рамках діапазону, для рядкових – як шаблон функції LIKE оператора SELECT SQL, для логічного типу – значення True/False, для дат – у рамках діапазону дат.
4. Програмний код виконати згідно шаблону MVC (модель-подання-контролер).

**Посилання в телеграмі:** <https://t.me/axepent>

**Посилання на репозиторій:** <https://github.com/Axepent/BD>

**Опис предметної області**

Обрана предметна область - система обліку екзаменаційних балів студентів. Система має дозволити викладачам та студентам слідкувати за загальною успішністю студентів. Це допомагає формувати звіти про результати навчання та забезпечувати прозорість процесу оцінювання.

**Опис сутностей:**

1. Студент (Student)

- `student\_id` (первинний ключ)
- `first\_name` (ім'я)
- `last\_name` (прізвище)
- `com\_method` (спосіб комунікації)

Призначення: збереження даних про студентів

2. Викладач (Teacher)

- `teacher\_id` (первинний ключ)
- `first\_name` (ім'я)

- `last\_name` (прізвище)
- `com\_method` (спосіб комунікації)

Призначення: збереження даних про викладачів

### 3. Іспит (Exam)

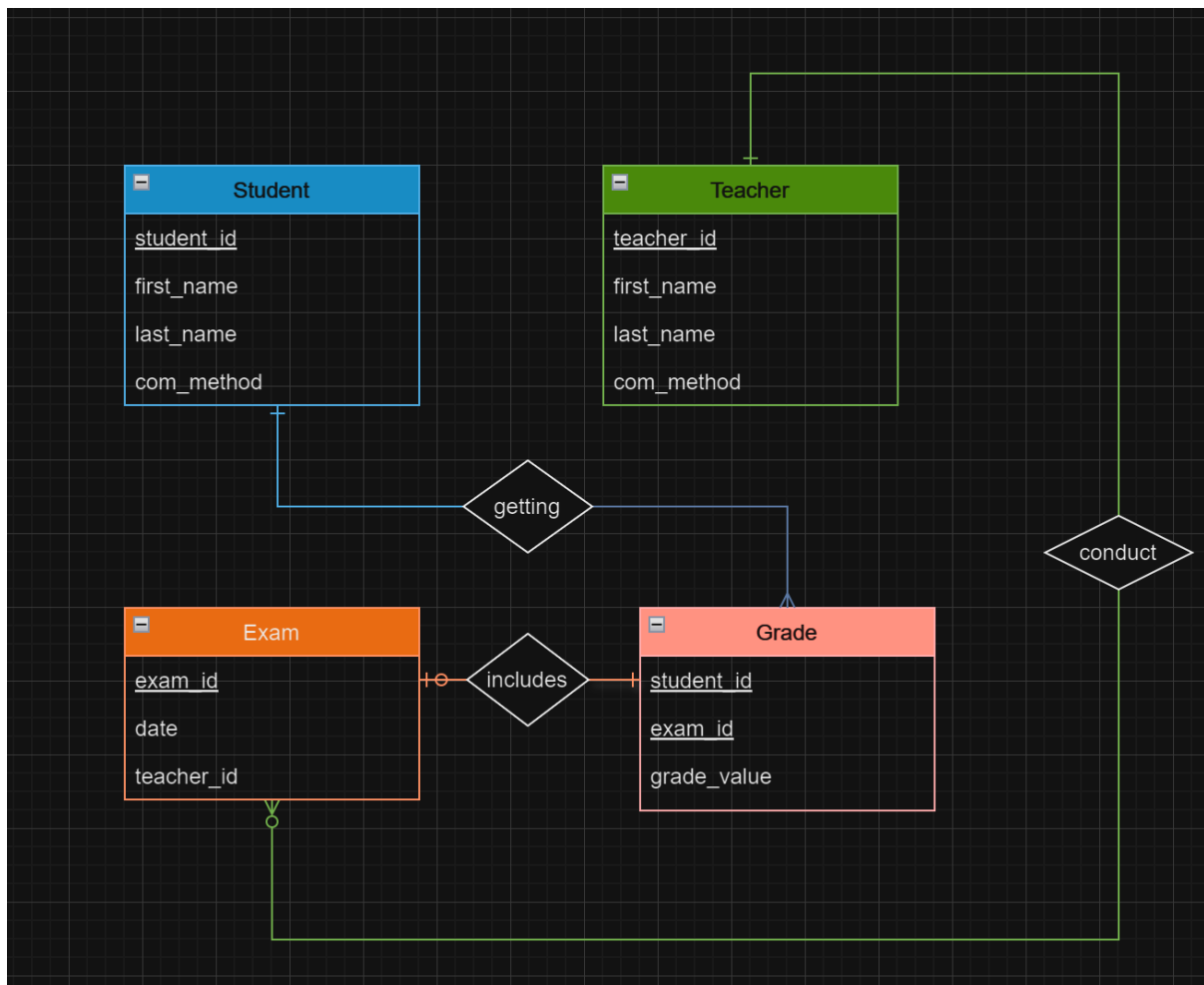
- `exam\_id` (первинний ключ)
- `teacher\_id` (посилання на вчителя)
- `date` (дата проведення іспиту)

Призначення: збереження даних про іспити

### 4. Оцінка (Grade)

- `student\_id` (посилання на студента)
- `exam\_id` (посилання на іспит)
- `grade\_value` (оцінка студента, числове значення)

Призначення: збереження оцінок студентів за конкретні іспити



## Хід роботи

### 1. Програмна взаємодія з базою даних

Menu:

1. Add Student
2. View Students
3. Update Student
4. Delete Student
5. Add Teacher
6. View Teachers
7. Update Teacher
8. Delete Teacher
9. Add Exam
10. View Exams
11. Update Exam
12. Delete Exam
13. Add Grade
14. View Grades
15. Update Grade
16. Delete Grade
17. Quit

Enter your choice:

Оновлене консольне меню

1. “Add Student” – додати нового студента, після вибору цієї опції проходить повне заповнення усіх атрибутів необхідний в таблиці Student

```
Menu:
1. Add Student
2. View Students
3. Update Student
4. Delete Student
5. Quit
Enter your choice: 1
Enter student ID: 100001
Enter first name: Name
Enter last name: Surname
Enter communication method (e.g., email, phone): sur.nam@ghgh.com
Student added successfully!
```

Query

Query History

1

▼

SELECT \* FROM public."Student"

2

ORDER BY student\_id DESC LIMIT 100

Data Output

Messages

Notifications

≡+

▼

▼

	student_id [PK] integer	first_name character varying	last_name character varying	com_method character varying
1	100001	Name	Surname	sur.nam@ghgh.com
2	100000	YEF	ODL	odl@example.com
3	99999	XTL	JWQ	jwq@example.com
4	99998	NWU	BYN	byn@example.com

Додавання нового студента до бази даних та перевірка дії в pgAdmin4

## 2. “View Students” – перегляд усіх студентів, що уже є в базі даних

```
ID: 99989, First Name: IYD, Last Name: HFY, Communication Method: hfy@example.com
ID: 99990, First Name: GFU, Last Name: MCI, Communication Method: mci@example.com
ID: 99991, First Name: YLW, Last Name: KMT, Communication Method: kmt@example.com
ID: 99992, First Name: WFY, Last Name: WRU, Communication Method: wru@example.com
ID: 99993, First Name: QFW, Last Name: LMS, Communication Method: lms@example.com
ID: 99994, First Name: HSP, Last Name: KIG, Communication Method: kig@example.com
ID: 99995, First Name: PHA, Last Name: UVC, Communication Method: uvc@example.com
ID: 99996, First Name: FVF, Last Name: LNA, Communication Method: lna@example.com
ID: 99997, First Name: CTL, Last Name: UHN, Communication Method: uhn@example.com
ID: 99998, First Name: NWU, Last Name: BYN, Communication Method: byn@example.com
ID: 99999, First Name: XTL, Last Name: JWQ, Communication Method: jwq@example.com
ID: 100000, First Name: YEF, Last Name: ODL, Communication Method: odl@example.com
ID: 100001, First Name: Name, Last Name: Surname, Communication Method: namsur@gmail.com

Menu:
1. Add Student
2. View Students
3. Update Student
4. Delete Student
5. Quit
Enter your choice: █
```

Через заздалегідь згенеровані дані неможливо показати усе через консоль, проте на скріншоті видні коректні дані, що збігаються з тим що можна побачити в pdAdmin4

3. “Update Student” – зміна уже існуючих даних студента, після вибору цієї опції з’являється запит який саме атрибут користувач хоче змінити

```
Menu:
1. Add Student
2. View Students
3. Update Student
4. Delete Student
5. Quit
Enter your choice: 3
Enter student ID: 100001
Enter column to update (first_name, last_name, com_method): com_method
Enter new value: namsur@gmail.com
Student updated successfully!
```

Query

Query History

1

▼

SELECT \* FROM public."Student"

2

ORDER BY student\_id DESC LIMIT 100

Data Output

Messages

Notifications

≡+

▼

▼

	<div>student_id</div> <div>[PK] integer</div> <div></div>	<div>first_name</div> <div>character varying</div> <div></div>	<div>last_name</div> <div>character varying</div> <div></div>	<div>com_method</div> <div>character varying</div> <div></div>
1	100001	Name	Surname	namsur@gmail.com
2	100000	YEF	ODL	odl@example.com

Зміна інформації про студента та перевірка в pgAdmin4



#### 4. “Delete Student” – видалення студента з бази даних

Menu:

1. Add Student
2. View Students
3. Update Student
4. Delete Student
5. Quit

Enter your choice: 4

Enter student ID: 100001

Student deleted successfully!

Query

Query History

1

2

3

SELECT \* FROM public."Student"

ORDER BY student\_id DESC LIMIT 100

Data Output

Messages

Notifications

	student_id [PK] integer	first_name character varying	last_name character varying	com_method character varying
1	100000	YEF	ODL	odl@example.com
2	99999	XTL	JWQ	jwq@example.com
3	99998	NWU	BYN	byn@example.com
4	99997	CTL	UHN	uhn@example.com
5	99996	FVF	LNA	lna@example.com
6	99995	PHA	UVC	uvc@example.com

Видалення студента з бази даних та перевірка в pgAdmin4

Це були приклади до ще не повного коду і відповідно не було доступу до повної бази даних



Подальші дії виконуються ті самі функції, але вже для відповідних таблиць

В pgAdmin4 були перевизначені дії при видаленні та оновленні даних для коректної обробки в деяких ситуаціях

### 5. Перевірка дій з таблицею “Teacher”

```
Menu:
1. Add Student
2. View Students
3. Update Student
4. Delete Student
5. Add Teacher
6. View Teachers
7. Update Teacher
8. Delete Teacher
9. Add Exam
10. View Exams
11. Update Exam
12. Delete Exam
13. Add Grade
14. View Grades
15. Update Grade
16. Delete Grade
17. Quit
Enter your choice: 7
Enter teacher ID: 1
Enter column to update (first_name, last_name, com_method): com_method
Enter new value: 4me2me@glue.com
Teacher updated successfully!
```

```
1 SELECT * FROM public."Teacher"
2 ORDER BY teacher_id ASC LIMIT 100
3
```

Data Output				
	teacher_id [PK] integer	first_name character varying	last_name character varying	com_method character varying
1	1	Robert	Lewis	4me2me@glue.com

Оновлення даних вчителя та перевірка в pgAdmin4

## 6. Перевірка дій з таблицею "Exam"

Menu:

1. Add Student
2. View Students
3. Update Student
4. Delete Student
5. Add Teacher
6. View Teachers
7. Update Teacher
8. Delete Teacher
9. Add Exam
10. View Exams
11. Update Exam
12. Delete Exam
13. Add Grade
14. View Grades
15. Update Grade
16. Delete Grade
17. Quit

Enter your choice: 9

Enter exam ID: 3333

Enter exam date (YYYY-MM-DD): 2099-12-12

Enter teacher ID for the exam: 676

Exam added successfully!

```
1  SELECT * FROM public."Exam"  
2  ORDER BY exam_id DESC LIMIT 100  
3
```

Data Output Messages Notifications

	exam_id [PK] integer	date date	teacher_id integer
1	3333	2099-12-12	676
2	3000	2046-07-04	357

Додавання нового екзамену та перевірка дії в pdAdmin4

Menu:

1. Add Student
2. View Students
3. Update Student
4. Delete Student
5. Add Teacher
6. View Teachers
7. Update Teacher
8. Delete Teacher
9. Add Exam
10. View Exams
11. Update Exam
12. Delete Exam
13. Add Grade
14. View Grades
15. Update Grade
16. Delete Grade
17. Quit

Enter your choice: 11

Enter exam ID: 1

Enter column to update (date, teacher\_id): date

Enter new value: 2026-11-11

Exam updated successfully!

1

▼

SELECT \* FROM public."Exam"

2

ORDER BY exam\_id ASC LIMIT 100


3

Data Output


Messages

Notifications


≡+





▼




▼












	exam_id [PK] integer	date date	teacher_id integer
1	1	2026-11-11	1

Оновлення даних про екзамен та перевірка в pdAdmin4




## 7. Комплексна перевірка дій з таблицею “Grade”

	<b>student_id</b> [PK] integer 	<b>exam_id</b> [PK] integer 	<b>grade_value</b> integer 
1	1	477	67
2	1	1404	44
3	1	2070	74

Перевірка даних до видалення одного з екзаменів




```
Menu:
1. Add Student
2. View Students
3. Update Student
4. Delete Student
5. Add Teacher
6. View Teachers
7. Update Teacher
8. Delete Teacher
9. Add Exam
10. View Exams
11. Update Exam
12. Delete Exam
13. Add Grade
14. View Grades
15. Update Grade
16. Delete Grade
17. Quit
Enter your choice: 16
Enter student ID: 1
Enter exam ID: 1404
Grade deleted successfully!
```

Видалення одного з результатів екзамену, з таблички з двома первинними ключами

	student_id [PK] integer 	exam_id [PK] integer 	grade_value integer 
1	1	477	67
2	1	2070	74

Перевірка видалення екзамену в pgAdmin4

```
Menu:
1. Add Student
2. View Students
3. Update Student
4. Delete Student
5. Add Teacher
6. View Teachers
7. Update Teacher
8. Delete Teacher
9. Add Exam
10. View Exams
11. Update Exam
12. Delete Exam
13. Add Grade
14. View Grades
15. Update Grade
16. Delete Grade
17. Quit
Enter your choice: 13
Enter student ID: 1
Enter exam ID: 1000
Enter grade value: 99
Grade added successfully!
```

	student_id [PK] integer 	exam_id [PK] integer 	grade_value integer 
1	1	477	67
2	1	1000	99
3	1	2070	74

Додавання нового екзамену до бази даних та відповідна перевірка



## 2. Генерація даних до відповідних таблиць у базі даних

```
Query Query History
1  INSERT INTO public."Student" (student_id, first_name, last_name, com_method)
2  SELECT
3      student_id,
4      first_name,
5      last_name,
6      lower(last_name) || '@example.com' AS com_method
7  FROM (
8      SELECT
9          generate_series(10001, 100000) AS student_id,
10         chr(trunc(65 + random() * 25)::int) ||
11         chr(trunc(65 + random() * 25)::int) ||
12         chr(trunc(65 + random() * 25)::int) AS first_name,
13         chr(trunc(65 + random() * 25)::int) ||
14         chr(trunc(65 + random() * 25)::int) ||
15         chr(trunc(65 + random() * 25)::int) AS last_name
16     ) AS generated_data;
17
```

Query Query History

1

SELECT \* FROM public."Student"

2

ORDER BY student\_id DESC LIMIT 100

3

Data Output

Messages

Notifications

	student_id [PK] integer	first_name character varying	last_name character varying	com_method character varying
1	100000	YEF	ODL	odl@example.com
2	99999	XTL	JWQ	jwq@example.com
3	99998	NWU	BYN	byn@example.com
4	99997	CTL	UHN	uhn@example.com
5	99996	FVF	LNA	lna@example.com
6	99995	PHA	UVC	uvc@example.com
7	99994	HSP	KIG	kig@example.com
8	99993	QFW	LMS	lms@example.com
9	99992	WFY	WRU	wru@example.com
10	99991	YLW	KMT	kmt@example.com
11	99990	GFU	MCI	mci@example.com
12	99989	IYD	HFY	hfy@example.com
13	99988	FVT	TUX	tux@example.com
14	99987	JHA	ROK	rok@example.com
15	99986	HQY	TGI	tgi@example.com
16	99985	NFJ	DMF	dmf@example.com
17	99984	WJF	YYN	yy@example.com
18	99983	MWO	GUF	guf@example.com
19	99982	UWF	QRY	qry@example.com
20	99981	QDM	EGK	egk@example.com
21	99980	EGE	IAR	iar@example.com
22	99979	UOU	EBA	eba@example.com
23	99978	XLA	SQJ	sqj@example.com
24	99977	VJU	VHN	vhn@example.com
25	99976	MVV	GFU	gfu@example.com

Останні елементи таблиці “Student”, що були згенеровано

‘student\_id’ – послідовність чисел

‘first\_name’ та ‘last\_name’ – 3 випадкові букви сконкантиновані до одного рядку

‘com\_method’ – згенеровано за правилом “last\_name” + “@example.com”

```

Query Query History
1  INSERT INTO public."Teacher" (teacher_id, first_name, last_name, com_method)
2  SELECT
3      teacher_id,
4      first_name,
5      last_name,
6      lower(last_name || '.' || first_name) || '@example.com' AS com_method
7  FROM (
8      SELECT
9          generate_series(2, 1000) AS teacher_id,
10         chr(trunc(65 + random() * 25)::int) ||
11         chr(trunc(65 + random() * 25)::int) ||
12         chr(trunc(65 + random() * 25)::int) AS first_name,
13         chr(trunc(65 + random() * 25)::int) ||
14         chr(trunc(65 + random() * 25)::int) ||
15         chr(trunc(65 + random() * 25)::int) AS last_name
16     ) AS generated_data;
17

```

Query Query History

1

2

3

SELECT \* FROM public."Teacher"

ORDER BY teacher\_id DESC LIMIT 100

Data Output Messages Notifications

	teacher_id [PK] integer	first_name character varying	last_name character varying	com_method character varying
1	1000	YMP	JHR	jhr.ym@example.com
2	999	BYW	NSJ	nsj.by@example.com
3	998	NSR	KNJ	kni.nsr@example.com
4	997	QYA	NXX	nxx.qya@example.com
5	996	UUA	WRD	wrd.uua@example.com
6	995	QFC	QPU	qpu.qfc@example.com
7	994	CBQ	WMI	wmi.cbq@example.com
8	993	YVU	XSJ	xsy.yvu@example.com
9	992	LVR	PFD	pdf.lvr@example.com
10	991	XJW	CQN	cqn.xjw@example.com
11	990	OAC	YEX	yex.oac@example.com
12	989	SGW	NPV	npv.sgw@example.com
13	988	KWN	XVI	xvi.kwn@example.com
14	987	RSY	NJI	nji.rsy@example.com
15	986	HAE	GIV	giv.hae@example.com
16	985	JND	VNT	vnt.jnd@example.com
17	984	BHP	WIW	wiw.bhp@example.com
18	983	KEX	VMD	vmd.kex@example.com
19	982	PHK	AKS	aks.phk@example.com
20	981	XPA	MFD	mfd.xpa@example.com
21	980	OLY	XAQ	xaq.oly@example.com
22	979	CDO	BOD	bod.cdo@example.com
23	978	KUT	KUK	kuk.kut@example.com
24	977	XIB	RWY	rwy.xib@example.com
25	976	NBS	AVF	avf.nbs@example.com

Останні елементи таблиці “Teacher”, що були згенеровано відмінність від таблиці “Student” в формі запису ‘com\_method’ при генерації (“last\_name” + “.” + “first\_name” + “@example.com”)



```
Query Query History
1 1 v INSERT INTO public."Exam" (exam_id, date, teacher_id)
2 SELECT
3     generate_series(2, 3000) AS exam_id,
4     '2040-01-01'::date + (random() * (interval '10 years')) AS date,
5     trunc(random() * 999 + 1)::int AS teacher_id;
6
```

```
1 1 v SELECT * FROM public."Exam"
2 ORDER BY exam_id DESC LIMIT 100
3
```

	exam_id [PK] integer	date date	teacher_id integer
1	3000	2046-07-04	357
2	2999	2044-09-15	660
3	2998	2049-04-13	914
4	2997	2042-06-17	827
5	2996	2040-03-14	792
6	2995	2046-11-24	727
7	2994	2046-07-26	411
8	2993	2047-01-16	184
9	2992	2047-02-02	890
10	2991	2049-11-26	449
11	2990	2042-02-18	276
12	2989	2048-08-27	448
13	2988	2049-09-28	878
14	2987	2041-12-04	394
15	2986	2047-02-25	856
16	2985	2041-07-26	945
17	2984	2049-08-16	556
18	2983	2049-10-13	212
19	2982	2041-02-13	388
20	2981	2041-10-06	970
21	2980	2041-04-16	199
22	2979	2049-01-07	156
23	2978	2046-04-30	704
24	2977	2045-10-02	868
25	2976	2047-12-16	698

Останні елементи таблиці “Exam”, що були згенеровано

‘exam\_id’ – послідовність чисел

‘date’ – дата в діапазоні 10 років починаючи з 2040-01-01

‘teacher\_id’ – випадкова вибірка з існуючих 1000 вчителів

```

Query Query History
1  INSERT INTO public."Grade" (student_id, exam_id, grade_value)
2  SELECT
3      student_id,
4      exam_id,
5      (trunc(random() * 61 + 40))::int AS grade_value
6  FROM (
7      SELECT
8          student_id,
9          DISTINCT ON (exam_id) (trunc(random() * 3000 + 1))::int AS exam_id
10     FROM generate_series(1, 100000) AS student_id,
11          generate_series(1, 3) AS exam_count
12     ) AS unique_exams
13     ORDER BY student_id, exam_id;
14

```

Query

Query History

1

2

3

SELECT \* FROM public."Grade"

ORDER BY student\_id DESC, exam\_id DESC LIMIT 100

Data Output

Messages

Notifications

	student_id [PK] integer	exam_id [PK] integer	grade_value integer
1	100000	2988	44
2	100000	1573	50
3	100000	761	72
4	99999	2998	87
5	99999	1944	83
6	99999	1713	57
7	99998	2839	61
8	99998	891	86
9	99998	5	50
10	99997	1938	91
11	99997	1550	97
12	99997	1462	80
13	99996	2242	84
14	99996	1616	67
15	99996	574	93
16	99995	1989	89
17	99995	1849	99
18	99995	8	76
19	99994	2615	97
20	99994	2277	40
21	99994	1848	52
22	99993	2933	44
23	99993	1806	83
24	99993	1094	75
25	99992	2259	80

Останні елементи таблиці “Grade”, що були згенеровано

‘student\_id’ – випадкова вибірка з існуючих 100000 учнів

‘exam\_id’ – випадкова вибірка з існуючих 3000 екзаменів

‘grade\_value’ – випадкова оцінка від 40 до 100

Для кожного студента генерується 3 екзамени

### 3. SQL запити та результати, які вони виводять

Query

Query History

1

2

3

4

5

6

7

8

9

10

11

12

13

14

SELECT

t.teacher\_id,

t.first\_name || ' ' || t.last\_name AS teacher\_name,

e.exam\_id,

e.date

FROM

public."Teacher" t

JOIN

public."Exam" e ON t.teacher\_id = e.teacher\_id

WHERE

t.teacher\_id = 99

ORDER BY

e.date;

Data Output

Messages

Notifications

≡+

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

SQL

	teacher_id integer	teacher_name text	exam_id integer	date date
1	99	OJQ HNY	407	2047-05-20
2	99	OJQ HNY	1398	2047-12-22

Інформація, що вводиться з клавіатури - 'teacher\_id'

Вивід 'first\_name' та 'last\_name' обраного вчителя в одну клітинку

Вивід 'exam\_id', що проводить цей вчитель, та дата їх проведення

Query

Query History

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

SELECT

g.student\_id,

s.first\_name || ' ' || s.last\_name AS student\_name,

g.exam\_id,

e.date,

g.grade\_value

FROM

public."Grade" g

JOIN

public."Student" s ON g.student\_id = s.student\_id

JOIN

public."Exam" e ON g.exam\_id = e.exam\_id

WHERE

g.grade\_value BETWEEN 95 AND 100

ORDER BY

g.student\_id random()

Data Output

Messages

Notifications

SQL

	student_id integer	student_name text	exam_id integer	date date	grade_value integer
1	2	Amane Kuro	573	2044-09-15	100
2	3	VCC OTD	1515	2041-05-11	96
3	4	JJF IBW	1593	2042-10-26	95
4	5	PLM BYG	57	2046-07-16	100
5	6	SYS DSJ	1114	2048-11-09	97
6	6	SYS DSJ	1267	2042-09-03	97
7	7	YEH FEY	2368	2045-11-06	97
8	10	HDN IYH	649	2048-01-17	96
9	15	TUD OYJ	1676	2046-03-12	98
10	15	TUD OYJ	272	2049-01-01	98
11	17	EEO CVH	2120	2049-03-20	99
12	18	CYR HNT	403	2047-02-26	95
13	18	CYR HNT	309	2043-12-26	100
14	26	VVD KAY	1820	2045-12-23	95
15	32	BXU QLH	2438	2047-10-21	99

Інформація, що вводиться з клавіатури - діапазон оцінок про який потрібно дізнатися

Вивід 'first\_name' та 'last\_name' студента, що отримав оцінку з вибраного діапазону, в одну клітинку

Вивід 'exam\_id' та 'date' для конкретного розуміння коли і за що конкретно була отримана оцінка

Вивід 'grade\_value' для перевірки конкретної оцінки з діапазону

Query	Query History
-------	---------------

```

1 SELECT
2     t.teacher_id,
3     t.first_name || ' ' || t.last_name AS teacher_name,
4     s.student_id,
5     s.first_name || ' ' || s.last_name AS student_name,
6     ROUND(AVG(g.grade_value))::int AS average_grade
7 FROM
8     public."Grade" g
9 JOIN
10    public."Exam" e ON g.exam_id = e.exam_id
11 JOIN
12    public."Teacher" t ON e.teacher_id = t.teacher_id
13 JOIN
14    public."Student" s ON g.student_id = s.student_id
15 WHERE
16     t.teacher_id BETWEEN 777 AND 797
17 GROUP BY
18     t.teacher_id, t.first_name, t.last_name, s.student_id, s.first_name, s.last_name
19 HAVING
20     AVG(g.grade_value) > 60
21 ORDER BY
22     t.teacher_id, average_grade DESC;
23

```

Data Output	Messages	Notifications
-------------	----------	---------------

SQL
-----

	teacher_id integer	teacher_name text	student_id integer	student_name text	average_grade integer
249	779	PFH YRG	760	PGU YLP	87
250	779	PFH YRG	54787	DWK ECI	85
251	779	PFH YRG	3840	JLR YSK	85
252	779	PFH YRG	50635	PDF GVC	84
253	779	PFH YRG	31667	DTN VTP	83
254	779	PFH YRG	62751	RBH UEX	83
255	779	PFH YRG	64811	KWP CQE	82

Інформація, що вводиться з клавіатури - діапазон вчителів та оцінка яка буде еталоном для порівняння

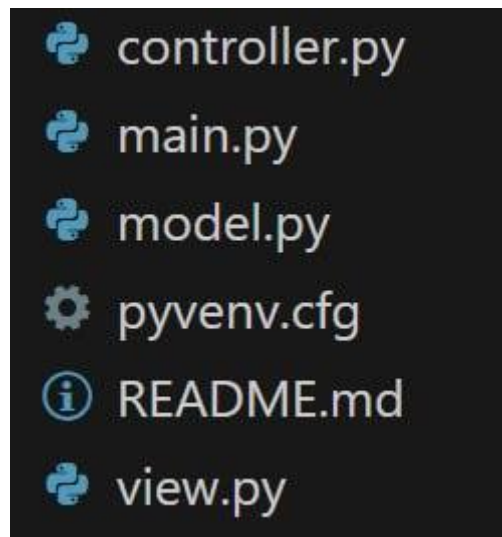
Вивід 'first\_name' та 'last\_name' вчителя, якщо в нього він взагалі проводив екзамени, в одну клітинку (а також відповідний 'teacher\_id')

Вивід 'first\_name' та 'last\_name' студента, який набрав бал вищий за еталон на екзамені у відповідного вчителя, в одну клітинку (а також відповідний 'student\_id')

Вивід оцінки 'average\_grade', що перевищує еталон та була отримана за екзамен

#### 4. Опис модулів програми

1. Модель (Model) – це клас, що відображає логіку роботи з даними, обробляє всі операції з даними, такі як додавання, оновлення, вилучення
2. Представлення (View) – це клас, через який користувач взаємодіє з програмою. У даному випадку, консольний інтерфейс, який відображає дані для користувача та зчитує їх з екрану
3. Контролер (Controller) – це клас, який відповідає за зв'язок між користувачем і системою. Він приймає введені користувачем дані та обробляє їх. В залежності від результатів, викликає відповідні дії з Model або View



main.py - виклик контролера та передача йому управління

model.py - відповідає за управління підключенням до бази даних і виконанням низькорівневих запитів до неї

controller.py - інтерфейс взаємодії з користувачем, включаючи обробку запитів користувача, виконання пошуку, а також інші дії, необхідні для взаємодії з моделлю та представленням

view.py - відображає результати виконання різних дій користувача на екрані консолі