



МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики
Кафедра програмного забезпечення комп’ютерних систем

Лабораторна робота № 3
з дисципліни “Бази даних”
тема “Засоби оптимізації роботи СУБД PostgreSQL”
Варіант 15

Виконала
студентка II курсу
групи КП-03

Серьодкіна Анна Євгенівна
(прізвище, ім'я, по батькові)

Київ 2021

Мета роботи

Метою роботи є здобуття практичних навичок використання засобів оптимізації СУБД PostgreSQL.

Завдання

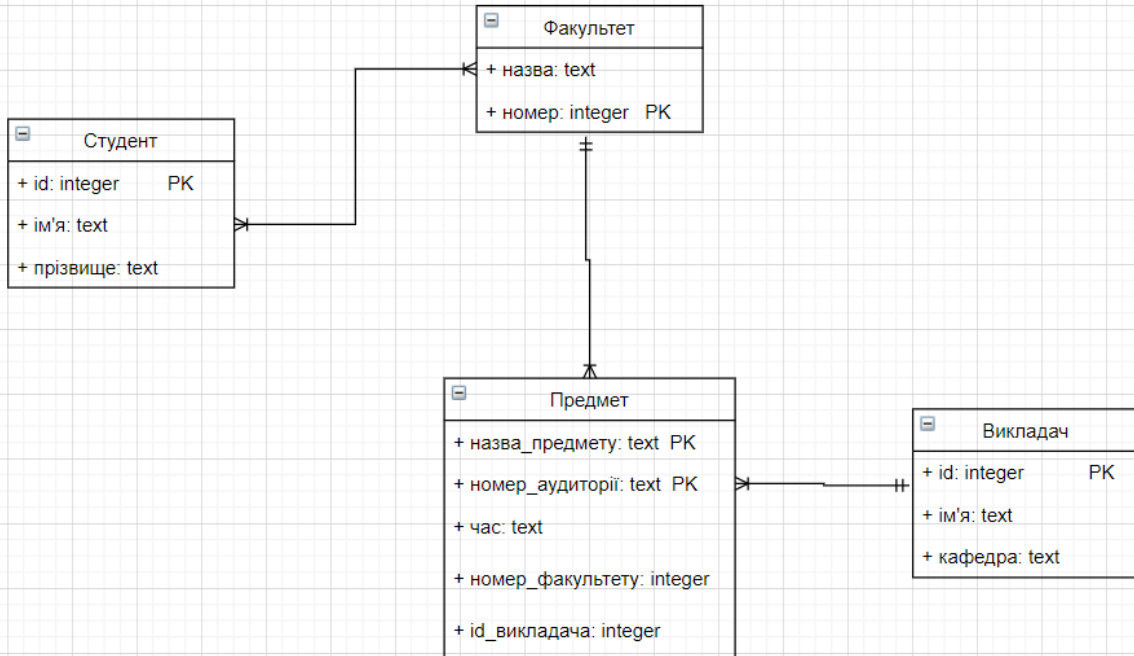
Завдання роботи полягає у наступному:

1. Перетворити модуль “Модель” з шаблону MVC лабораторної роботи No2 у вигляд об’єктно-реляційної проекції (ORM).
2. Створити та проаналізувати різні типи індексів у PostgreSQL.
3. Розробити тригер бази даних PostgreSQL.

15	Hash, BRIN	before delete, update
----	------------	-----------------------

Результати роботи

Завдання 1



Класи:

```
class Student(Base):
    __tablename__ = "students"
    first_name = Column(String)
    id = Column(Integer, primary_key=True)
    last_name = Column(String)
    faculties = relationship("Faculty",
secondary=students_faculties_association)

class Faculty(Base):
    __tablename__ = "faculties"
    name = Column(String)
    number = Column(Integer, primary_key=True)

class Teacher(Base):
    __tablename__ = "teachers"
    id = Column(Integer, primary_key=True)
    name = Column(String())
    department = Column(String)
```

```



class Subject(Base):
    __tablename__ = "subjects"
    name = Column(String(), primary_key=True)
    lecture_hall_number = Column(String(), primary_key=True)
    time = Column(String())
    faculty_number = Column(Integer, ForeignKey(Faculty.number))
    teacher_id = Column(Integer, ForeignKey(Teacher.id))

students_faculties_association = Table('students_faculties', Base.metadata,
    Column('student_id', Integer, ForeignKey('students.id')),
    Column('faculty_number', Integer, ForeignKey('faculties.number'))
)



```

Завдання 2

explain analyze select * from faculties where name = 'CPU'

	QUERY PLAN	
	text	
1	Seq Scan on faculties (cost=0.00..1.70 rows=1 width=8) (actual time=0.062..0.065 rows=1 loops=1)	
2	[...] Filter: ((name)::text = 'CPU'::text)	
3	[...] Rows Removed by Filter: 55	
4	Planning Time: 6.885 ms	
5	Execution Time: 0.117 ms	

```
create index hash_index on faculties using hash(name);
```

	QUERY PLAN	
	text	
1	Seq Scan on faculties (cost=0.00..1.70 rows=1 width=8) (actual time=0.102..0.111 rows=1 loops=1)	
2	[...] Filter: ((name)::text = 'CPU'::text)	
3	[...] Rows Removed by Filter: 55	
4	Planning Time: 4.761 ms	
5	Execution Time: 0.153 ms	

explain analyze select (name) from teachers where department = '121'

	QUERY PLAN
	text
1	Seq Scan on teachers (cost=0.00..1.01 rows=1 width=32) (actual time=0.021..0.025 rows=1 loops=1)
2	[...] Filter: ((department)::text = '121'::text)
3	[...] Rows Removed by Filter: 36
4	Planning Time: 0.098 ms
5	Execution Time: 0.041 ms

```
CREATE INDEX my_brin_index_x ON teachers USING BRIN (department) WITH  
(pages_per_range = 128);
```

	QUERY PLAN
	text
1	Seq Scan on teachers (cost=0.00..1.46 rows=1 width=32) (actual time=0.067..0.093 rows=1 loops=1)
2	[...] Filter: ((department)::text = '121'::text)
3	[...] Rows Removed by Filter: 36
4	Planning Time: 5.404 ms
5	Execution Time: 0.128 ms

Отже, у випадку з hash індексом вдалося підвищити швидкодію виконання запитів. А після створення brin індексу швидкодія помітно погіршується.

Завдання 3

Trigger 1:

create or replace function updateFunc()

returns trigger AS

\$\$

begin

IF NEW.id < OLD.id THEN

NEW.id := NEW.id || '1000';

END IF;

NEW.id := OLD.id / NEW.id;

EXCEPTION

WHEN division_by_zero THEN

RAISE NOTICE 'error is handled: division_by_zero';

return new;

end;

```
$$  
language 'plpgsql';
```

```
create trigger myUpdateTrigger  
before update  
ON students  
FOR EACH ROW  
execute procedure updateFunc();
```

Trigger 2:

```
create or replace function deleteFunc()  
returns trigger AS  
$$  
begin
```

```
SELECT count(*)  
FROM faculties;
```

```
UPDATE subjects  
SET time = '12:25';
```

```
return new;  
end;  
$$  
language 'plpgsql';
```

```
create trigger myDeleteTrigger  
before delete  
ON teachers  
FOR EACH ROW  
execute procedure deleteFunc();
```