



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 “Проектування бази даних та ознайомлення з базовими операціями СУБД  
PostgreSQL”

Виконала

студентка II курсу

групи КП-02

Гудзіцька Дарина Сергіївна

варіант №3

Київ 2021

## Мета роботи

Здобути практичні навички використання засобів оптимізації СУБД PostgreSQL.

## Постановка завдання

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.

## Результати

### Завдання №1:

Класи ORM:

```
from sqlalchemy import Column, String, Integer,

from base import Base

class Subject(Base):
    __tablename__ = 'subjects'

    id = Column(Integer, primary_key=True)
    name = Column(String)
    credits = Column(Integer)

    def __init__(self, name, credits):
        self.name = name
        self.credits = credits
```

```
from sqlalchemy import Column, String, Integer, Boolean,
from sqlalchemy.orm import relationship, backref

from base import Base

class Student(Base):
    __tablename__ = 'students'
    id = Column(Integer, primary_key=True)
    firstname = Column(String)
    lastname = Column(String)
    group_id = Column(Integer, ForeignKey('groups.id'))
    group = relationship("Group", backref="students")

    def __init__(self, firstname, lastname, group):
        self.firstname = firstname
        self.lastname = lastname
        self.group = group
```

```
from sqlalchemy import Column, String, Integer, Date,

from base import Base

class Group(Base):
    __tablename__ = 'groups'

    id = Column(Integer, primary_key=True)
    name = Column(String)
    faculty = Column(String)
    student_number = Column(Integer)

    def __init__(self, name, faculty, student_number):
        self.name = name
        self.faculty = faculty
        self.student_number = student_number
```

## Приклади запитів:

```
Введите действие:  
update student  
  
Введите данные:  
1 firstname qqq  
[INFO] Update students successful
```

```
Введите действие:  
add group  
  
Введите данные:  
qqq zzz 23  
[INFO] Add to 'groups' successful
```

## Завдання №2:

Текст коду індексування:

```
def create_index1_group(connection):  
    with connection.cursor() as cursor:  
        cursor.execute(  
            """CREATE INDEX idx1_group_st_num ON groups USING hash (student_number);  
            SELECT * FROM groups WHERE student_number < 100"""  
        )  
        c = cursor.fetchall()  
        for i in c:  
            print("|", i[0], "|", i[1], "|", i[2], "|\\n")  
  
def create_index2_group(connection):  
    with connection.cursor() as cursor:  
        cursor.execute(  
            """CREATE INDEX idx2_group_st_num ON groups USING hash (name);  
            SELECT * FROM groups WHERE student_number > 200 AND name LIKE 'C__'  
            ORDER BY name ASC"""  
        )  
        c = cursor.fetchall()  
        for i in c:  
            print("|", i[0], "|", i[1], "|", i[2], "|\\n")
```

## Приклади результатів:

33 Введіть номер прикладу індексації

1

| 25 | LQ4 | QYF |

| 29 | FX17 | JES |

| 35 | XF20 | JTD |

| 38 | WC20 | AQB |

| 40 | CW29 | FYQ |

| 44 | BE27 | OGY |

33 Введіть номер прикладу індексації

2

| 80171 | CA0 | GHD |

| 5770 | CA0 | XTL |

| 14993 | CA0 | TPL |

| 53562 | CA0 | NQS |

| 38113 | CA0 | CSW |

| 5660 | CA0 | KGV |

33 Введіть номер прикладу індексації

3

| 35 | XYZ0123456789 | 89 | 1 |

| 40 | ABCDEFGHIJKLMNOP | 89 | 8 |

| 68 | TUVWXYZ01234567 | 89 | 4 |

| 131 | GHIJKLMNOPQRSTU | 89 | 10 |

| 151 | FGHJKLMNOPQRST | 89 | 6 |

| 224 | BCDEFGHIJKLMNOP | 89 | 1 |

| 308 | QRSTUVWXYZ01234 | 89 | 5 |

33 Введіть номер прикладу індексації

4

| 14481 | 0123456789 | 6789 | 8 |

| 51942 | 0123456789 | 6789 | 8 |

| 97612 | 0123456789 | 6789 | 9 |

| 33246 | 0123456789 | 6789 | 7 |

| 96578 | 0123456789 | 6789 | 8 |

### Завдання №3:

Текст коду:

```
def create_trigger1(connection):
    with connection.cursor() as cursor:
        cursor.execute(
            """CREATE TRIGGER subject_update_trigger
            BEFORE UPDATE
            ON subjects
            FOR EACH ROW
            EXECUTE PROCEDURE log_name_changes()
            """
        )





def create_trigger1_fun(connection):
    with connection.cursor() as cursor:
        cursor.execute(
            """CREATE OR REPLACE FUNCTION log_name_changes()
            RETURNS TRIGGER
            LANGUAGE PLPGSQL
            AS
            $$
            BEGIN
                IF NEW.name <> OLD.name THEN
                    INSERT INTO subjects_audits(subject_id,name,changed_on)
                    VALUES(OLD.id,OLD.name,now());
                END IF;

                RETURN NEW;
            END;
            """
        )
```

Ініціювання виконання тригера:

```
def test_trigger1(connection):
    with connection.cursor() as cursor:
        cursor.execute(
            """UPDATE subjects SET name = 'qqq', credits = 13 WHERE id = 374741"""
        )
    print("[INFO] Update (trigger) 'subjects' successful")
```

Результуюча таблиця:

	id 	subject_id 	name 	changed_on 
1	1	374741	IJKLMNOPQRSTUVWXYZ	2021-12-06 18:36:51.233585