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”

Виконав студент
ІІ курсу групи КП-01
Тітов Єгор Павлович

Перевірив
“____” “_____” 20___р
асистент
Радченко Костянтин
Олександрович

Київ 2021

Мета: здобуття практичних навичок використання засобів оптимізації СУБД PostgreSQL.

Завдання

Завдання роботи полягає у наступному:

1. Перетворити модуль “Модель” з шаблону MVC лабораторної роботи 2 у вигляд об’єктно-реляційної проекції (ORM).

2. Створити та проаналізувати різні типи індексів у PostgreSQL.

3. Розробити тригер бази даних PostgreSQL.

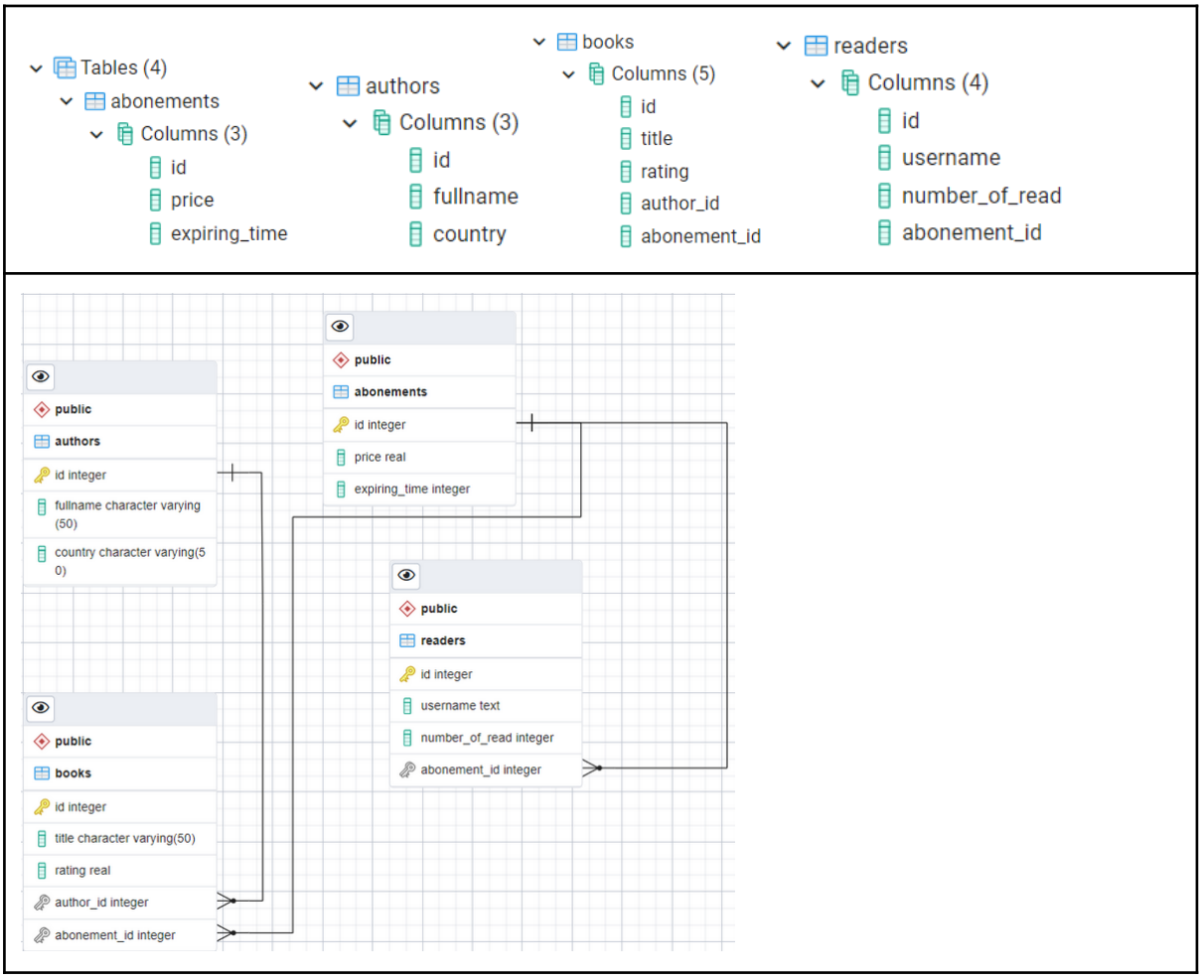
Варіант №17.

Індекси: BTree, GIN

Умови для тригера: before update, delete.

Копії екрану, що підтверджують виконання вимог

1. Таблиці бази даних:



Класи ORM:

```python
class Author(Base):
    __tablename__ = 'authors'
    id = Column(Integer, primary_key=True, unique=True, nullable=False)
    fullname = Column(String(50))
    country = Column(String(50))
    books = relationship('Book')
    __table_args__ = {'extend_existing': True}
```

```python
class Book(Base):
    __tablename__ = 'books'
    id = Column(Integer, primary_key=True, unique=True, nullable=False)
    title = Column(String(50))
    rating = Column(Float)
    author_id = Column(Integer, ForeignKey('authors.id'))
    abonement_id = Column(Integer, ForeignKey('abonements.id'))
    __table_args__ = {'extend_existing': True}
```

```python
class Reader(Base):
    __tablename__ = 'readers'
    id = Column(Integer, primary_key=True, unique=True, nullable=False)
    username = Column(Text)
    number_of_read = Column(Integer)
    abonement_id = Column(Integer, ForeignKey('abonements.id'))
    __table_args__ = {'extend_existing': True}
```

```python
class Abonement(Base):
    __tablename__ = 'abonements'
    id = Column(Integer, primary_key=True, unique=True, nullable=False)
    price = Column(Float)
    expiring_time = Column(Integer)
    __table_args__ = {'extend_existing': True}
```

Приклад команди

```
def create(self, id, title, rating, author_id, abonement_id):

    if (id < 1):
        print('Invalid id entered')
        return
    try:
        session = Session()
        session.add(Book(
            id = id,
            title = title,
            rating = rating,
            author_id = author_id,
            abonement_id = abonement_id
        ))
        session.commit()
        print("Entity successfully inserted")
```

Приклад запиту до користувача

```
Enter command (create, update, delete): create
Enter table name: authors
Enter integer value: 99
Enter value: someName
Enter value: SomeCountry
Entity successfully inserted
End of operation
```

id [PK] integer	fullname character varying (50)	country character varying (50)
1	2	2
99	someName	SomeCountry

2. Приклади індексування:

Команди створення індексів:

BTree:

```
selecr_query = """CREATE INDEX ON authors USING BTREE(id);
```

GIN:

```
CREATE INDEX book_name ON books USING gin (to_tsvector('english', code));
```

Результати виконання команд:

```
selecr_query = """SELECT * FROM authors WHERE id = 101000"""

Result [(101000, '22567b508261e608bab4f56dfb6a716b', 'fa980b4379551f995b1e0b1f4bb36678')]
Time for operation 0.0008388003334403038
```

```
selecr_query = """SELECT authors.fullname, books.title FROM books, authors WHERE books.author_id = 122495 AND authors.id = 122495"""

Result [('some author', '8cbc9b66e7c48a8245241aae5b2d9d89')]
Time for operation 0.0007518003694713116
```

3. Тригери:

Команда створення тригеру

```
query = """DROP TABLE IF EXISTS book_logs;
        CREATE TABLE book_logs(id integer NOT NULL, old_title text, new_title text, author_id integer);
        CREATE OR REPLACE FUNCTION log_book() RETURNS trigger AS $BODY$
        BEGIN
            IF NEW.title IS NULL THEN
                RAISE EXCEPTION 'Name cannot be null';
            END IF;
            IF NEW.author_id IS NULL THEN
                RAISE EXCEPTION 'Book cannot have null author_id';
            END IF;
            INSERT INTO book_logs VALUES(OLD.id, OLD.title, NEW.title, NEW.author_id);
            RETURN NEW;
        END;
    $BODY$ LANGUAGE plpgsql;
    DROP TRIGGER IF EXISTS book_subj ON subjects;
    CREATE TRIGGER book_subj BEFORE UPDATE OR DELETE ON books
        FOR EACH ROW EXECUTE PROCEDURE book_subj();"""
```

Приклади роботи:

Редагування

```
Enter command (create, update, delete): update
Enter table name: books
Enter integer value: 11
Enter value: abcd
Enter integer value: 7
Enter integer value: 122495
Enter integer value: 19626
1 Entity updated
```

	id [PK] integer	title character varying (50)	rating real	author_id integer	abonement_id integer
1	11	abcd	7	122495	19626

```
def update(self, id, title, rating, author_id, abonement_id):

    if (id < 1):
        print('Invalid id entered')
        return
    try:
        t = session.query(Book).get(id)
        t.title = title
        t.rating = rating
        t.author_id = author_id
        t.abonement_id = abonement_id
        session.add(t)
        session.commit()
        print("Entity successfully updated")

    except (Exception, Error) as error:
        print("Error occured in PostgreSQL: ", error)
    finally:
        print("End of operation")
```

Видалення

```
Enter command (create, update, delete): delete
Enter table name: books
Enter integer value: 11
1 Entity deleted
```

```
def delete(self, id):

    if (id < 1):
        print('Invalid id entered')
        return
    try:
        t = session.query(Book).get(id)
        session.delete(t)
        session.commit()
        print("Entity successfully deleted")

    except (Exception, Error) as error:
        print("Error occured in PostgreSQL: ", error)
    finally:
        print("End of operation")
```

Контрольні запитання

1. Сформулювати призначення та задачі об’єктно-реляційної проекції (ORM).

Призначенням є пов'язати бази даних з концепціями об'єктно-орієнтованих мов програмування, створюючи «віртуальну об'єктну базу даних». Необхідно забезпечити роботу з даними в термінах класів, а не таблиць даних, і, навпаки, перетворити терміни та дані класів на дані, придатні для зберігання в СУБД. Необхідно також забезпечити інтерфейс для CRUD- операцій над даними. Загалом, необхідно позбутися необхідності писати SQL-код для взаємодії в СУБД.

2. Проаналізувати основні види індексів у PostgreSQL (BTree, BRIN, GIN, Hash): призначення, сфера застосування, переваги та недоліки.

Незважаючи на всі відмінності між типами індексів (названими також методами доступу), зрештою будь-який з них встановлює відповідність між ключем (наприклад, значенням проіндексованого стовпця) та рядками таблиці, в яких цей ключ зустрічається. У PostgreSQL використовуються такі основні види індексів: BTree, BRIN, GIN, Hash. Рядки ідентифікуються за допомогою TID (tuple id), який складається з номера блоку файлу та позиції рядка всередині блоку. Тоді, знаючи ключ або деяку інформацію про нього, можна швидко прочитати ті рядки, в яких може знаходитися інформація, що цікавить нас, не переглядаючи всю таблицю повністю. Важливо розуміти, що індекс, прискорюючи доступ до даних, натомість потребує певних витрат на свою підтримку.

3. Пояснити призначення тригерів та функцій у базах даних.

Тригер запускається сервером автоматично при спробі зміни даних в таблиці, з якою він зв'язаний. Функції такого типу дозволяють реалізувати складнішу логіку.

Висновки

В результаті виконання лабораторної роботи я здобув практичні навички з використання засобів оптимізації СУБД PostgreSQL.