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
Грищенко Олександр Сергійович

Перевірив
“__” “_____” 20__р
викладач
Радченко Костянтин Олександрович

Київ 2021

Мета

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

Постановка задачі

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.

Варіант №18

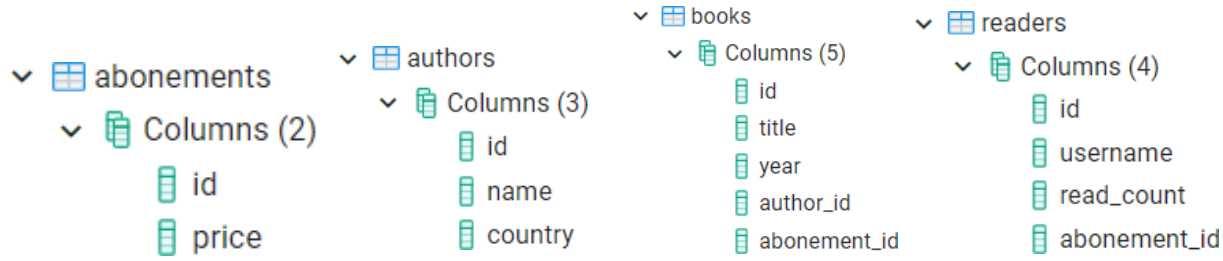
Індекси: BTree, GIN

Тригери: after update, insert

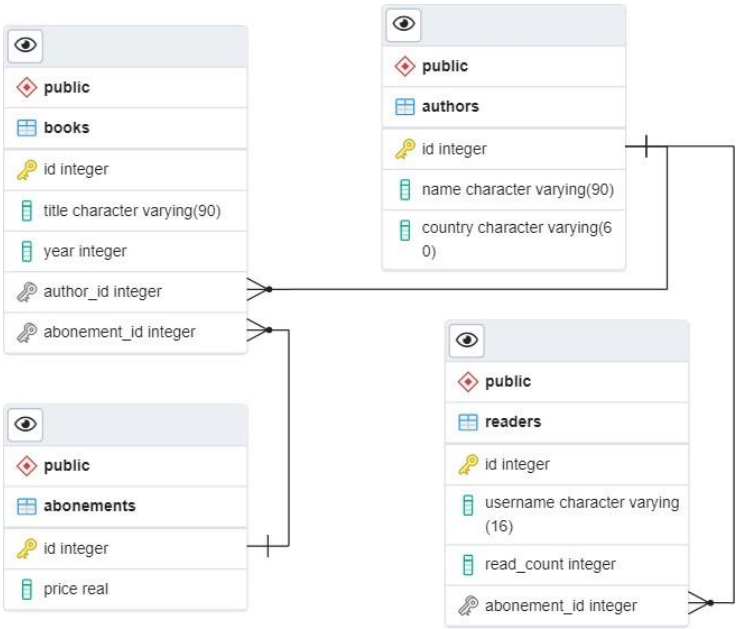
Результат роботи

Завдання №1

Таблиці бази даних:



Зв’язки між таблицями



Класи ORM для таблиць:

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

```python
class Author(Base):
    __tablename__ = 'authors'
    id = Column(Integer, primary_key=True, unique=True, nullable=False)
    name = Column(String(90))
    country = Column(String(60))
    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(90))
    year = Column(Integer)
    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(String(60))
    read_count = Column(Integer)
    abonement_id = Column(Integer, ForeignKey('abonements.id'))
    __table_args__ = {'extend_existing': True}
```

Приклад запиту у ORM:

```
Enter command (create, update, delete): create
Enter table name (authors, books, abonements, readers): authors
Enter name, country
Enter string: Lesya Ukrainka
Enter string: Ukraine
Author was inserted
```

100003	100003	cb1902b2e7c7e7b693e5dcb5016d92aa	7019ffc10f0bfa40e329d5fb21f607de
100004	100004	Lesya Ukrainka	Ukraine

Завдання №2

Команди створення індексів

BTree:

```python
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 = 46142"""
```

```
Result [(46142, '68b7bd86475eef507d1f0594566955a6', '1abf68fd91905a2f1d27ebe64fffbcd0')]
Time for operation 0.0015976000000001989
```

```
selecr_query = """SELECT authors.name, books.title FROM books, authors WHERE books.author_id = 12933 AND authors.id = 12933"""
```

```
Result [('6e868255e8da8cfa7f050bf440a041ad', '63ce0cce68de7800dd3bb577dcfb71be'), ('6e868
Time for operation 0.00023319999999937835
```

Завдання №3

Команди що викликають тригер:

```
def create(self, title, year, author_id, abonement_id):
    try:
        session = Session()
        session.add(Book(title=title, year=year, author_id=author_id, abonement_id=abonement_id))
        session.commit()
        print("Book was inserted")

    except (Exception, Error) as error:
        print("Error occured in PostgreSQL: ", error)
```

```
def update(self, id, title, year, author_id, abonement_id):
    if (id < 1):
        print('Invalid id')
        return
    try:
        t = session.query(Book).get(id)
        t.title = title
        t.year = year
        t.author_id = author_id
        t.abonement_id = abonement_id
        session.add(t)
        session.commit()
        print("Book was updated")
```

Текст тригера:

```
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 AFTER UPDATE OR INSERT ON books
           FOR EACH ROW EXECUTE PROCEDURE book_subj();"""
```

Зміни в таблиці після виконання команди:

```
Enter command (create, update, delete): create
Enter table name (authors, books, abonements, readers): books
Enter title, year, author_id, abonement_id
Enter string: poems
Enter integer: 1900
Enter integer: 71998
Enter integer: 15401
Book was inserted
```

	id [PK] integer	title text	author_id integer	abonement_id integer
1	100014	poems	71988	15401

Висновки

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