Міністерство освіти і науки України

Національний технічний університет

«Дніпровська політехніка»

ЗВІТ

Практична робота №5

з дисципліни

«Поглиблене програмування в середовищі Java»

Виконала:

ст. гр. 122-21-3

Бєляєва В. В.

Прийняв:

Доцент каф. САіУ

Мінєєв О. С.

м. Дніпро

2024 рік

Практична робота №5

Jdbc

Мета роботи: навчитися працювати з Jdbc

Завдання до виконання: Лабораторна робота номер 5. Jdbc

Створити базу даних в будь якому сервері баз даних. Створити таблицю з переліком студентів вказати їх прізвище, ім'я, по батькові, день народження номер залікової книжки та ID.

Створити програму що буде дозволяти виводити на екран інформацію про студентів які народилися в тому чи іншому місяці року. Програма повинна завдяки системі jdbc під'єднатися до вашої бази даних та робити до неї запроси. Вимог до розробки бази даних немає. Програма ж має бути написана за усіма стандартами ООП. Та може бути спроектована за двох принципів:

- при будь-якій ситуації буде забиратися весь перелік студентів, а вже на стороні java буде зроблено пошук необхідного

- SQL запрос буде сформований згідно запросу який зробив користувач і вже сервер управління баз даних буде вирішувати, які самі студенти народилися в тому чи іншому місяці.

У висновку обов'язково пояснити чому вибрали той чи інший принцип, які в нього переваги та недоліки. Оцінка не залежить від того який сервер управління баз даних вибрали. Перелік студентів зробити не менше 20 людей. Місяць червень зробити місяцем, коли в жодного зі студентів немає дня народження.

SQL код створення бази даних розмістити проекті 6 лабораторної роботи в файлі database в пекеджі resources. Для використання цієї лабораторної роботи рекомендується активно використовувати знання отримані на дисципліні що стосуються розробки баз даних.

До паперового звіту обов'язково додати принтскрин з програми в якій ви дивитесь інформацію вашого сервера управління баз даних, де показати створену таблицю, її ім'я та загальні відомості бази даних, наприклад назва, ім'я, назва користувача адміністратора, пароль тощо. Для роботи з сервером управління баз даних рекомендуємо використовувати програмне забезпечення компанії jetbrains datagrip. Або вбудовану панель користування базами даних, що міститься у середовищі intellij Idea, яка на сьогоднішній день підтримує майже всі сервери управління баз даних.

Хід роботи

CONNECTION NAME: java-lab5-studentDB

USERNAME: students_db

PASSWORD: - (жодного паролю не встановлено)

Було створено нового користувача, базу даних та перевірено підключення.

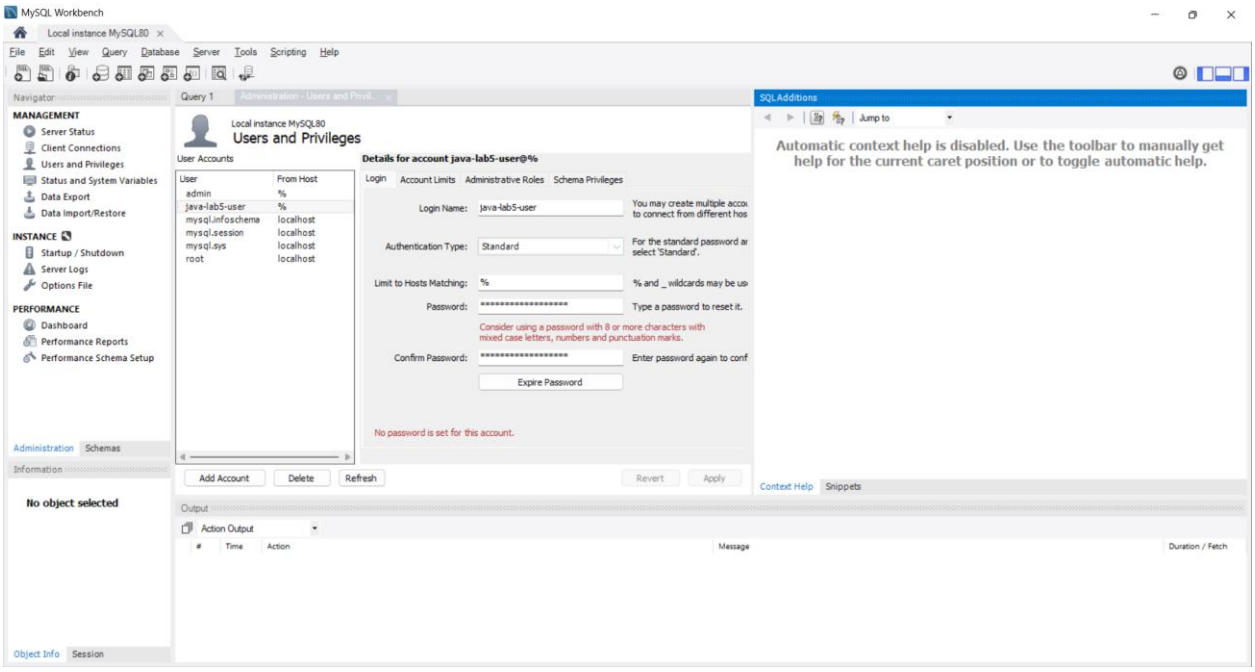

Рисунок 1 – Створення користувача

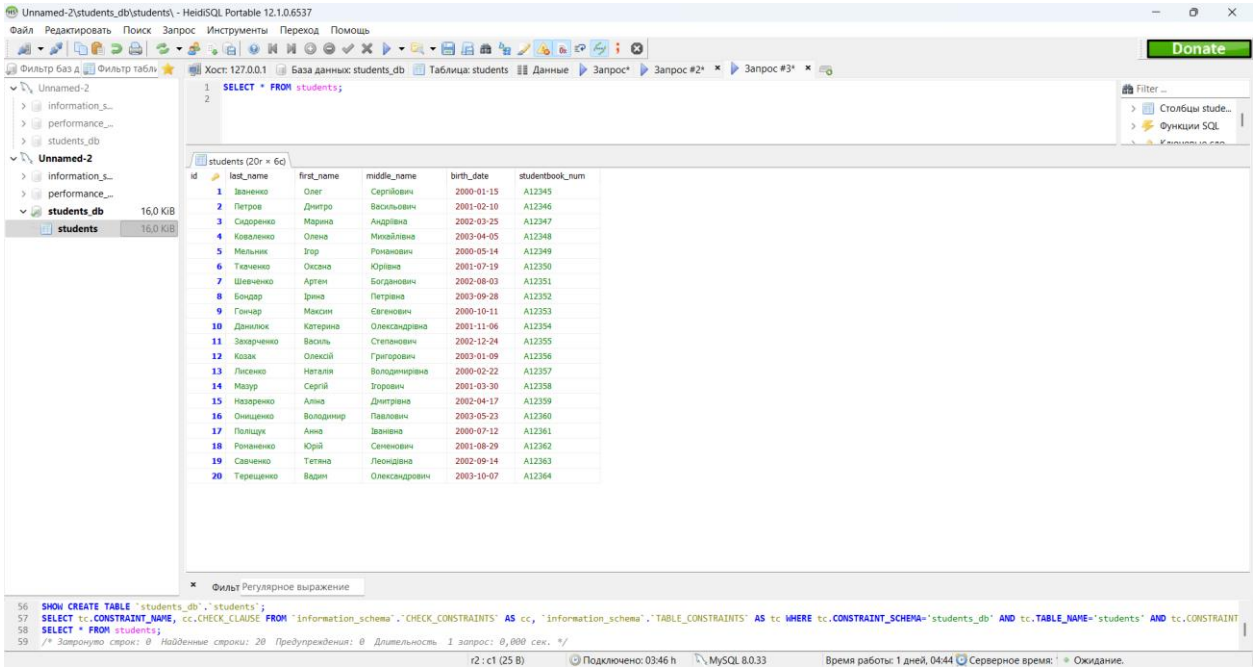

Рисунок 2 – Заповнена таблиця бази даних

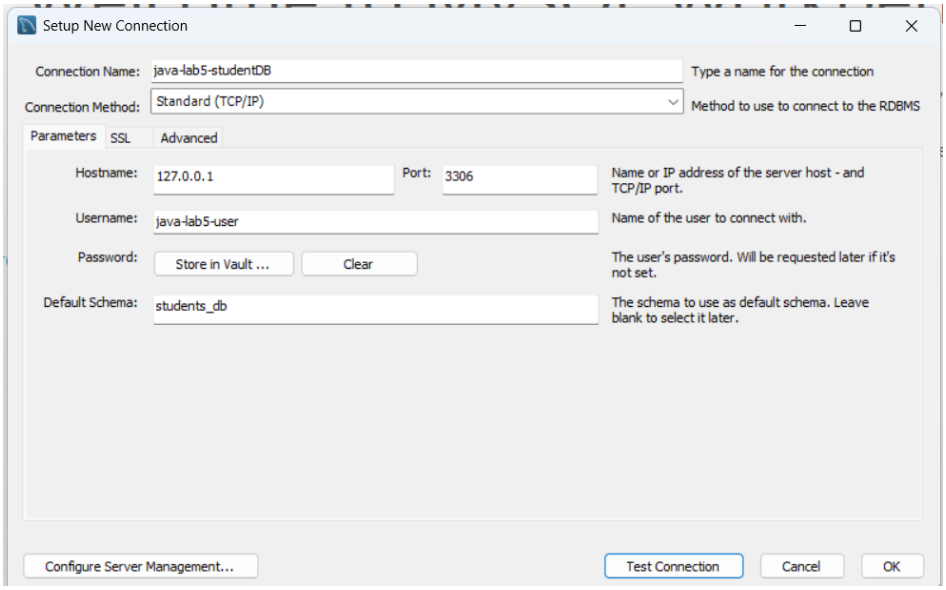

Рисунок 3 – Підключення до БД

Далі було створено програму:

Main.java (головний клас)

```
package org.example;

import java.util.List;
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        StudentDAO studentDAO = new StudentDAO();
        StudentService studentService = new StudentService();
        Scanner scanner = new Scanner(System.in);

        System.out.print("Введіть номер місяця (1-12): ");
```

```
        int month = scanner.nextInt();

        System.out.println("\n1. Завантажити всіх студентів і відфільтрувати
в Java");
        System.out.println("2. Виконати SQL-запит для отримання студентів");
        System.out.print("Виберіть варіант (1 або 2): ");
        int choice = scanner.nextInt();

        List<Student> students;

        if (choice == 1) {
            students =
studentService.filterStudentsByMonth(studentDAO.getAllStudents(), month);
        } else {
            students = studentDAO.getStudentsByMonth(month);
        }

        if (students.isEmpty()) {
            System.out.println("Немає студентів, народжених у цьому
місяці.");
        } else {
            System.out.println("Список студентів:");
            students.forEach(System.out::println);
        }
    }
}
```

DatabaseManager.java (клас для підключення до БД)

```
package org.example;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DatabaseManager {
    private static final String URL =
"jdbc:mysql://localhost:3306/students_db";
    private static final String USER = "java-lab5-user";  // Ваш користувач
    private static final String PASSWORD = "";  // Ваш пароль

    static {
        try {
            Class.forName("com.mysql.cj.jdbc.Driver"); // Завантаження
драйвера
        } catch (ClassNotFoundException e) {
            throw new RuntimeException("Помилка завантаження драйвера MySQL",
e);
        }
    }

    public static Connection getConnection() throws SQLException {
        return DriverManager.getConnection(URL, USER, PASSWORD);
    }
}
```

Student.java (модель студента)

```
package org.example;

public class Student {
    private int id;
```

```
    private String lastName;
    private String firstName;
    private String middleName;
    private String birthDate;
    private String studentId;

    public Student(int id, String lastName, String firstName, String
middleName, String birthDate, String studentId) {
        this.id = id;
        this.lastName = lastName;
        this.firstName = firstName;
        this.middleName = middleName;
        this.birthDate = birthDate;
        this.studentId = studentId;
    }

    public String getBirthDate() {
        return birthDate;
    }

    @Override
    public String toString() {
        return String.format("%d: %s %s %s, Дата народження: %s, Номер
залікової: %s",
                id, lastName, firstName, middleName, birthDate, studentId);
    }
}
```

StudentDAO.java (робота з БД)

```
package org.example;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class StudentDAO {

    public List<Student> getAllStudents() {
        List<Student> students = new ArrayList<>();
        String query = "SELECT * FROM students";

        try (Connection conn = DatabaseManager.getConnection();
             Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(query)) {

            while (rs.next()) {
                students.add(new Student(
                        rs.getInt("id"),
                        rs.getString("last_name"),
                        rs.getString("first_name"),
                        rs.getString("middle_name"),
                        rs.getString("birth_date"),
                        rs.getString("studentbook_num")
                ));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return students;
    }
```

```
    public List<Student> getStudentsByMonth(int month) {
        List<Student> students = new ArrayList<>();
        String query = "SELECT * FROM students WHERE MONTH(birth_date) = ?";

        try (Connection conn = DatabaseManager.getConnection();
             PreparedStatement stmt = conn.prepareStatement(query)) {

            stmt.setInt(1, month);
            ResultSet rs = stmt.executeQuery();

            while (rs.next()) {
                students.add(new Student(
                        rs.getInt("id"),
                        rs.getString("last_name"),
                        rs.getString("first_name"),
                        rs.getString("middle_name"),
                        rs.getString("birth_date"),
                        rs.getString("studentbook_num")
                ));
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return students;
    }
}
```

StudentService.java (логіка пошуку)

```
package org.example;

import java.util.List;
import java.util.stream.Collectors;

public class StudentService {
    private final StudentDAO studentDAO = new StudentDAO();

    public List<Student> filterStudentsByMonth(List<Student> students, int
month) {
        return students.stream()
                .filter(s -> Integer.parseInt(s.getBirthDate().split("-")[1])
== month)
                .collect(Collectors.toList());
    }
}
```

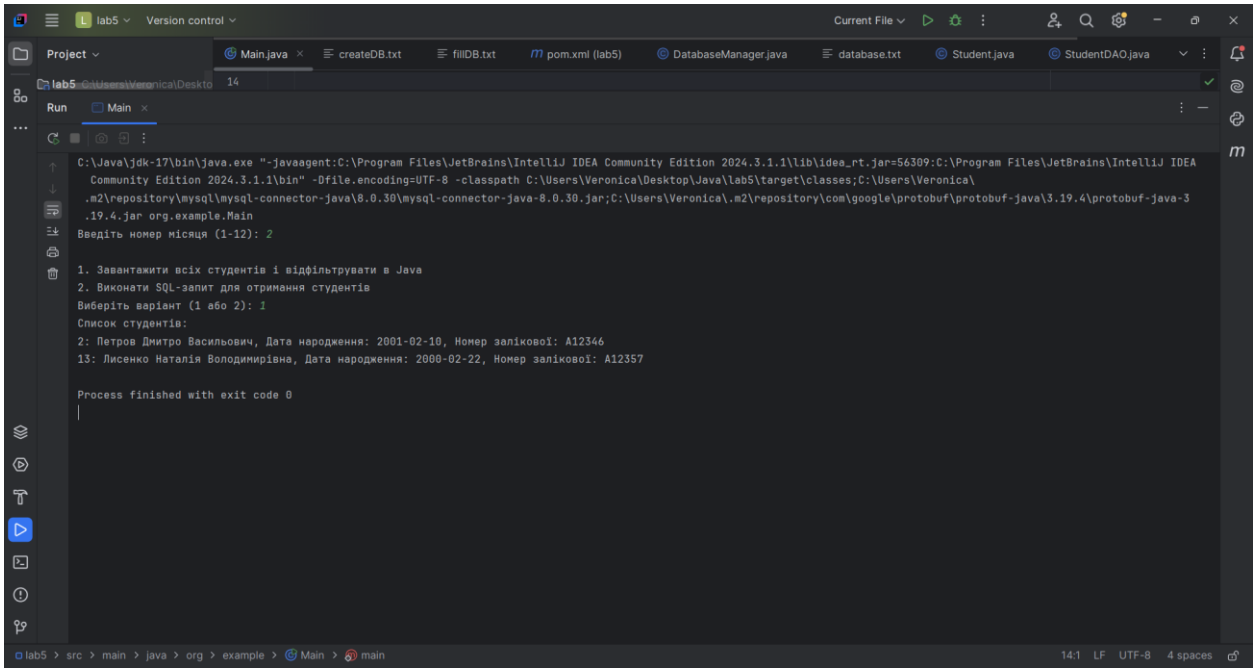

Рисунок 4 – Результат роботи програми

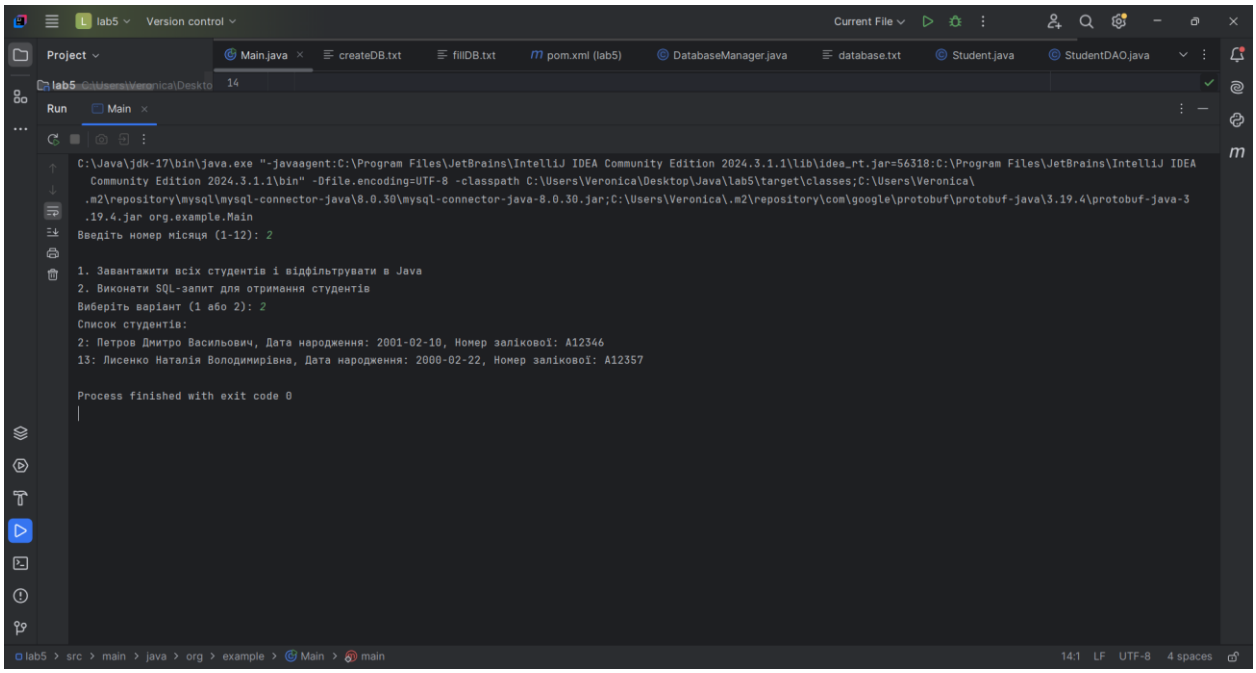

Рисунок 5 – Результат роботи програми

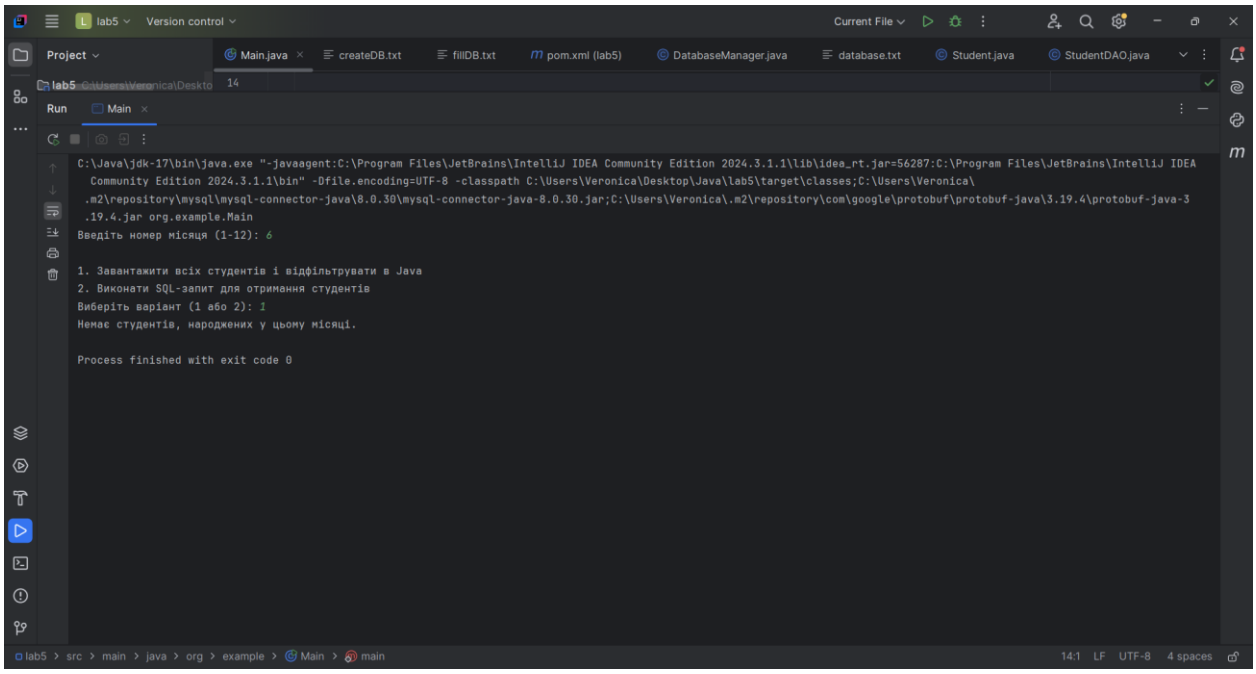

Рисунок 6 – Результат роботи програми

Висновок: у створеній програмі логіка пошуку студентів було реалізовано за обома принципами. Далі наведено їх порівняння:

1. Завантаження всіх студентів і фільтрація в Java

Спочатку виконується запит, що повертає всіх студентів з таблиці. Далі у Java-програмі список студентів фільтрується за потрібним місяцем.

Переваги:

- Менше навантаження на сервер баз даних – запит завжди один і не змінюється.
- Не потрібно динамічно змінювати SQL-запит.
- Можна додати складніші умови фільтрації (наприклад, відбір за декількома критеріями).

Недоліки:

- Підхід не оптимальний при великій кількості даних. Якщо в базі тисячі студентів, передавати їх усіх у Java – це зайве використання ресурсів.

2. Фільтрація безпосередньо в SQL

Програма формує SQL-запит динамічно, враховуючи введені користувачем параметри. Далі сервер бази даних виконує пошук студентів за вказаним місяцем.

Переваги:

- Підхід не використовує зайві ресурси– передаються тільки ті студенти, які відповідають умовам.
- Менше навантаження на Java-додаток, оскільки основну роботу виконує СУБД.

Недоліки:

- Виникає певна залежність від бази даних. Запит може бути специфічним для конкретної СУБД (наприклад, MySQL або PostgreSQL мають свої особливості).
- Програма з такою логікою є потенційно вразливою. Якщо неправильно обробляти введені користувачем дані, може з’явитися SQL-ін'єкція.