МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Вятский государственный университет»**

**(ФГБОУ ВО «ВятГУ»)**

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

**«Базы данных»**

Связывание приложения на Java с базой данных под управлением PostgreSQL

Выполнил студент группы ИВТб-31 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Кудяшев Я.Ю./

(подпись)

Проверил преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Клюкин В.Л/

(подпись)

Киров 2022

# Задание и цель лабораторной работы

Цель работы:

* Познакомиться со стандартным API в Java для связывания приложения с БД
* Изучить некоторые шаблоны проектирования, связанные с работой с БД
* Освоить на практике основы взаимодействия с БД под управлением PostgreSQL в приложении на Java

Задание на лабораторную работу:

Создать приложение с графическим приложением на языке Java. Приложение должно использовать БД, разработанную в предыдущих лабораторных работах.

Для любой одной таблицы, которая содержит внешний ключ на другую таблицу, приложение должно выполнять следующее:

* Выводить строки таблицы
* Предоставлять любой фильтр по значениям строк. (Например, «Дата с … по …» или «Имя содержит …»)
* Предоставлять возможность добавления новых строк
* Предоставлять возможность удаления строки

Требования к реализации:

* Заголовки должны быть осмысленными. Например, вместо «name» в таблице должен быть заголовок «Имя»
* При добавлении новой строки внешний ключ выбирается из списка
* Сохранение или удаление строки должно быть реализовано с помощью функции PL/pgSQL
* Фильтрация значений при поиске должна производиться через запрос, а не в полученной коллекции
* Разрешается использование любого фреймворка
* При разработке нужно использовать шаблоны проектирования, связанные с работой с БД

# Экранные формы

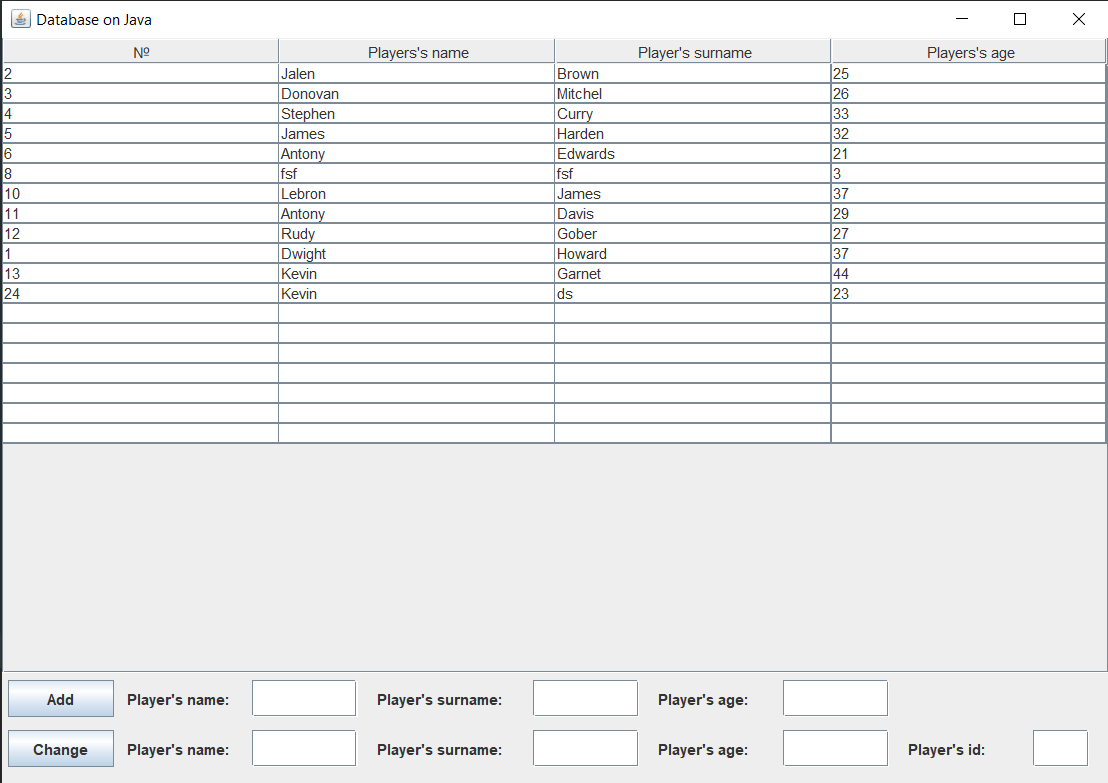


Рисунок 1 – Пример отображения приложения

# Код программы

**Main.java**

package com.ben;

import Helpers.DBHelper;

public class Main {

public static void main(String[] args) {

DBHelper db = new DBHelper();

db.simpleSelect();

Interface app = new Interface();

app.setVisible(true);

db.closeConnection();

}

}

**DBconnection.java**

package Helpers;

import com.ben.Interface;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.sql.\*;

import static com.ben.Interface.text1;

public class DBHelper {

public static Object[][] array = new String[][] {{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

{ "" , "", "","" },

};

// Заголовки столбцов

public static Object[] columnsHeader = new String[] {"№","Players's name", "Player's surname",

"Players's age"};

//Объект соединения с БД

private Connection connection;

//Строка подключения к БД

private static String DB\_URL = "jdbc:postgresql://localhost:5432/lab";

//Имя пользователя и пароль для подключения

private static String DB\_USER = "postgres";

private static String DB\_PASSWORD = "123123";

public DBHelper() {

try {

//Регистрируем драйвер

Class.forName("org.postgresql.Driver");

//Соединяемся с базой

connection = DriverManager.getConnection(DB\_URL, DB\_USER, DB\_PASSWORD);

System.out.println("Successful launch");

} catch (ClassNotFoundException e) {

//Сюда мы попадем, если драйвер не будет установлен

System.out.println("Failed to load driver");

} catch (SQLException e) {

//Сюда мы попадем, если возникнет проблема при соединении

System.out.println("Failed to connect to database");

}

}

//Закрытие соединения

public void closeConnection()

{

try {

if (connection != null && !connection.isClosed()) {

connection.close();

}

}

catch (Exception e) {

e.printStackTrace();

}

}

public void simpleSelect() {

//Объявляем объекты запроса и результата

Statement statement;

ResultSet result;

try {

//Создаем экземпляр запроса

statement = connection.createStatement();

//Выполняем запрос к БД

result = statement.executeQuery("SELECT \* FROM public.players");

//Выполняем проход по строкам таблицы и выводим в консоль

int i=0;

while(result.next())

{

// System.out.printf("Player: %s %s\nHis age: %d\n\n",

array[i][0] = String.valueOf(result.getInt(1));

array[i][1] = result.getString(2);

array[i][2] = result.getString(3);

array[i][3] = String.valueOf(result.getInt(5));

i+=1;

}

//Закрываем соединение

statement.close();

} catch (Exception e) {

e.printStackTrace();

}

}

public void simpleInsert(String first\_name, String second\_name, int age) {

//Объявляем объекты запроса и результата

PreparedStatement preparedStatement;

try {

//Создаем экземпляр запроса

preparedStatement = connection.prepareStatement("INSERT INTO players(first\_name, second\_name,id\_teams,age) VALUES(?, ?, 1,?)");

//Добавление параметров

preparedStatement.setString(1, first\_name);

preparedStatement.setString(2, second\_name);

preparedStatement.setInt(3, age);

//Выполняем запрос к БД

preparedStatement.execute();

//Закрываем соединение

preparedStatement.close();

} catch (Exception e) {

e.printStackTrace();

}

}

public void simpleUpdate(String first\_name, String second\_name, int age,int id) {

//Объявляем объекты запроса и результата

PreparedStatement preparedStatement;

try {

//Создаем экземпляр запроса

preparedStatement = connection.prepareStatement("UPDATE players SET first\_name = ?, second\_name = ?, age = ? WHERE id = " + String.valueOf(id));

//Добавление параметров

preparedStatement.setString(1, first\_name);

preparedStatement.setString(2, second\_name);

preparedStatement.setInt(3, age);

//Выполняем запрос к БД

preparedStatement.execute();

//Закрываем соединение

preparedStatement.close();

} catch (Exception e) {

e.printStackTrace();

}

}

}

**UI.java**

package com.ben;

import Helpers.DBHelper;

import javax.swing.\*;

import java.util.\*;

import java.awt.\*;

import java.awt.event.\*;

import static Helpers.DBHelper.\*;

public class Interface extends JFrame {

ActionListener actionListener = new TestActionListener();

public static JButton button1 = new JButton("Add");

public static JTextField text1 = new JTextField("");

public static JTextField text2 = new JTextField("");

public static JTextField text3 = new JTextField("");

public static JButton button2 = new JButton("Change");

public static JButton button3 = new JButton("Delete");

public static JTextField text4 = new JTextField("");

public static JTextField text5 = new JTextField("");

public static JTextField text6 = new JTextField("");

public static JTextField text7 = new JTextField("");

public static JTextField delete\_text = new JTextField("");

/\*\*

\* Constructor for main window

\*/

public Interface() {

super("Database on Java");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(900, 650);

setLocationRelativeTo(null);

JTable table1 = new JTable(array, columnsHeader);

Box contents = new Box(BoxLayout.Y\_AXIS);

contents.add(new JScrollPane(table1));

setContentPane(contents);

JPanel panel = new JPanel();

panel.setLayout(null);

button1.setBounds(5, 5, 85, 30);

button1.addActionListener(actionListener);

panel.add(button1);

JLabel label1 = new JLabel("Player's name: ");

label1.setBounds(100, 5, 85, 30);

text1.setBounds(200, 5, 85, 30);

panel.add(label1);

panel.add(text1);

JLabel label2 = new JLabel("Player's surname: ");

label2.setBounds(300, 5, 110, 30);

text2.setBounds(425, 5, 85, 30);

panel.add(label2);

panel.add(text2);

JLabel label3 = new JLabel("Player's age: ");

label3.setBounds(525, 5, 85, 30);

text3.setBounds(625, 5, 85, 30);

panel.add(label3);

panel.add(text3);

button2.setBounds(5, 45, 85, 30);

button2.addActionListener(actionListener);

panel.add(button2);

JLabel label4 = new JLabel("Player's name: ");

label4.setBounds(100, 45, 85, 30);

text4.setBounds(200, 45, 85, 30);

panel.add(label4);

panel.add(text4);

JLabel label5 = new JLabel("Player's surname: ");

label5.setBounds(300, 45, 110, 30);

text5.setBounds(425, 45, 85, 30);

panel.add(label5);

panel.add(text5);

JLabel label6 = new JLabel("Player's age: ");

label6.setBounds(525, 45, 85, 30);

text6.setBounds(625, 45, 85, 30);

panel.add(label6);

panel.add(text6);

JLabel label7 = new JLabel("Player's id: ");

label7.setBounds(725, 45, 85, 30);

text7.setBounds(825, 45, 45, 30);

panel.add(label7);

panel.add(text7);

getContentPane().add(panel);

}

public class TestActionListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

if (e.getSource() == button1){

new DBHelper().simpleInsert(text1.getText(),Interface.text2.getText(),Integer.valueOf(Interface.text3.getText()));

new DBHelper().simpleSelect();

}

if (e.getSource() == button2){

new DBHelper().simpleUpdate(text4.getText(),Interface.text5.getText(),Integer.valueOf(Interface.text6.getText()),Integer.valueOf(Interface.text7.getText()));

new DBHelper().simpleSelect();

}

}

}

}

1. **Вывод**

В ходе выполнения лабораторной работы были изучены стандартные API в Java для связывания приложения с БД**,** изучены некоторые шаблоны проектирования, связанные с работой с БД и освоены на практике основы взаимодействия с БД под управлением PostgreSQL в приложении на Java.