МИНОБРНАУКИ РФ

ФГБОУ ВО «ИжГТУ имени М.Т. Калашникова»

Кафедра «Программное обеспечение»

Отчет

по лабораторной работе №4 по дисциплине

"Архитектура электронных вычислительных машин"

Вариант 5

Выполнил студент группы Б18-191-2: Р. А. Гумметов

Принял: К. С. Чернышев

Ижевск 2020

1. Постановка задачи

Визуализировать таблицы БД lecturer, exam\_marks, subject, university, subj\_lect. Часть сделать встроенными в главное окно, часть – в отдельных окнах. Во всех показываемых колонках должны быть реальные данные, а не id (внешние ключи).

2. Текст программы

import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.sql.\*;  
import java.util.logging.Level;  
import java.util.logging.Logger;  
  
public class MainForm {  
 static Connection conn=null; // объект для связи с БД  
 static Statement stmt=null; //объект для создания SQL-запросов  
 static JFrame mainFrame; //окно программы  
 static JFrame secondFrame;  
 static JScrollPane scrollPane; //панель с полосой прокрутки  
 public static void main(String[] args) {  
 mainFrame=new JFrame("SuperDB\_Viewer");  
 mainFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  
 scrollPane = new JScrollPane();  
 mainFrame.add(setMenu(), BorderLayout.NORTH);  
 mainFrame.setSize(750, 400);  
 mainFrame.setMinimumSize(mainFrame.getSize());  
 mainFrame.setVisible(true);  
 }  
 private static Component setMenu() { //функция, возвращающая панель с кнопками  
//создаем объект класса Box для хранения кнопки (и в дальнейшем других кнопок) для таблицы  
 Box mainMenu = new Box(BoxLayout.X\_AXIS);  
 JButton studentsButton=new JButton("Show students");  
 studentsButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent arg0) {  
 mainFrame.remove(scrollPane); //убираем предыдущую панель с таблицей  
//даже если не было еще панели с таблицей  
 ResultSet rs=null; //объект для хранения результатов SQL-запроса  
 System.out.println("Students Table!!"); //вывод в консоль для контроля  
//записываем в rs результат работы функции fromDB с переданным ей запросом  
 rs=fromDB("select \* from student");  
//создаем таблицу из результатов запроса rs при помощи функции toTableView  
 JTable studentsTable=toTableView(rs, "students");  
//создаем новую панель с прокруткой  
 scrollPane = new JScrollPane(studentsTable);  
//нужно для корректного отображения больших таблиц  
 studentsTable.setFillsViewportHeight(true);  
//вставляем панель с таблицей в центр окна  
 mainFrame.add(scrollPane,BorderLayout.CENTER);  
 mainFrame.pack(); //подстраиваем размеры окна под размеры таблицы  
 }  
 });  
 mainMenu.add(studentsButton); //добавляем в Box кнопку  
//подобным способом можно добавить другие кнопки для показа всех таблиц БД  
  
 //  
 JButton lecturerButton=new JButton("Show lecturer");  
 lecturerButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent arg0) {  
 mainFrame.remove(scrollPane); //убираем предыдущую панель с таблицей  
//даже если не было еще панели с таблицей  
 ResultSet rs=null; //объект для хранения результатов SQL-запроса  
 System.out.println("Lecturer Table!!"); //вывод в консоль для контроля  
//записываем в rs результат работы функции fromDB с переданным ей запросом  
 rs=fromDB("select \* from lecturer");  
//создаем таблицу из результатов запроса rs при помощи функции toTableView  
 JTable lecturerTable=toTableView(rs, "lecturer");  
//создаем новую панель с прокруткой  
 scrollPane = new JScrollPane(lecturerTable);  
//нужно для корректного отображения больших таблиц  
 lecturerTable.setFillsViewportHeight(true);  
//вставляем панель с таблицей в центр окна  
 mainFrame.add(scrollPane,BorderLayout.CENTER);  
 mainFrame.pack(); //подстраиваем размеры окна под размеры таблицы  
 }  
 });  
 mainMenu.add(lecturerButton);  
  
  
 JButton examMarksButton=new JButton("Show exam\_marks");  
 examMarksButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent arg0) {  
 mainFrame.remove(scrollPane); //убираем предыдущую панель с таблицей  
//даже если не было еще панели с таблицей  
 ResultSet rs=null; //объект для хранения результатов SQL-запроса  
 System.out.println("Exam\_marks Table!!"); //вывод в консоль для контроля  
//записываем в rs результат работы функции fromDB с переданным ей запросом  
 rs=fromDB("select \* from exam\_marks");  
//создаем таблицу из результатов запроса rs при помощи функции toTableView  
 JTable exam\_marksTable=toTableView(rs, "exam\_marks");  
//создаем новую панель с прокруткой  
 scrollPane = new JScrollPane(exam\_marksTable);  
//нужно для корректного отображения больших таблиц  
 exam\_marksTable.setFillsViewportHeight(true);  
//вставляем панель с таблицей в центр окна  
 mainFrame.add(scrollPane,BorderLayout.CENTER);  
 mainFrame.pack(); //подстраиваем размеры окна под размеры таблицы  
 }  
 });  
 mainMenu.add(examMarksButton);  
  
 JButton subjectButton=new JButton("Show subject");  
 subjectButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent arg0) {  
 //mainFrame.remove(scrollPane);  
 secondFrame=new JFrame("Subject\_show");  
 secondFrame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);  
 scrollPane = new JScrollPane();  
 //secondFrame.add(setMenu(), BorderLayout.NORTH);  
 secondFrame.setSize(600, 400);  
 secondFrame.setMinimumSize(secondFrame.getSize());  
 secondFrame.setVisible(true);  
 mainFrame.setVisible(false);  
 secondFrame.addWindowListener(new java.awt.event.WindowAdapter() {  
 @Override  
 public void windowClosing(java.awt.event.WindowEvent windowEvent) {  
 mainFrame.setVisible(true);  
 }  
 });  
 secondFrame.remove(scrollPane);  
 ResultSet rs=null;  
 System.out.println("Subject Table!!");  
 rs=fromDB("select \* from subject");  
 JTable subjectTable=toTableView(rs, "subject");  
 scrollPane = new JScrollPane(subjectTable);  
 subjectTable.setFillsViewportHeight(true);  
 //mainFrame.add(scrollPane,BorderLayout.CENTER);  
 secondFrame.add(scrollPane,BorderLayout.CENTER);  
 //mainFrame.pack();  
 secondFrame.pack();  
 }  
 });  
 mainMenu.add(subjectButton);  
  
 JButton universityButton=new JButton("Show university");  
 universityButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent arg0) {  
 //mainFrame.remove(scrollPane);  
 secondFrame=new JFrame("Subject\_show");  
 secondFrame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);  
 scrollPane = new JScrollPane();  
 secondFrame.setSize(600, 400);  
 secondFrame.setMinimumSize(secondFrame.getSize());  
 secondFrame.setVisible(true);  
 mainFrame.setVisible(false);  
 secondFrame.addWindowListener(new java.awt.event.WindowAdapter() {  
 @Override  
 public void windowClosing(java.awt.event.WindowEvent windowEvent) {  
 mainFrame.setVisible(true);  
 }  
 });  
 secondFrame.remove(scrollPane);  
 ResultSet rs=null;  
 System.out.println("University Table!!");  
 rs=fromDB("select \* from university");  
 JTable universityTable=toTableView(rs, "university");  
 scrollPane = new JScrollPane(universityTable);  
 universityTable.setFillsViewportHeight(true);  
 //mainFrame.add(scrollPane,BorderLayout.CENTER);  
 secondFrame.add(scrollPane,BorderLayout.CENTER);  
 //mainFrame.pack();  
 secondFrame.pack();  
 }  
 });  
 mainMenu.add(universityButton);  
  
 JButton subj\_lectButton=new JButton("Show subj\_lect");  
 subj\_lectButton.addActionListener(new ActionListener() {  
 @Override  
 public void actionPerformed(ActionEvent arg0) {  
 mainFrame.remove(scrollPane); //убираем предыдущую панель с таблицей  
//даже если не было еще панели с таблицей  
 ResultSet rs=null; //объект для хранения результатов SQL-запроса  
 System.out.println("Subj\_lect Table!!"); //вывод в консоль для контроля  
//записываем в rs результат работы функции fromDB с переданным ей запросом  
 rs=fromDB("select \* from subj\_lect");  
//создаем таблицу из результатов запроса rs при помощи функции toTableView  
 JTable subj\_lectTable=toTableView(rs, "subj\_lect");  
//создаем новую панель с прокруткой  
 scrollPane = new JScrollPane(subj\_lectTable);  
//нужно для корректного отображения больших таблиц  
 subj\_lectTable.setFillsViewportHeight(true);  
//вставляем панель с таблицей в центр окна  
 mainFrame.add(scrollPane,BorderLayout.CENTER);  
 mainFrame.pack(); //подстраиваем размеры окна под размеры таблицы  
 }  
 });  
 mainMenu.add(subj\_lectButton);  
  
 //  
 return mainMenu; //возвращаем Box  
 }  
 //функция для формирования таблицы из результатов SQL-запроса  
 protected static JTable toTableView(ResultSet rs2, String str) {  
 int rowCount=0; //переменная для кол-ва записей  
 String[][] data = null; //массив строк для содержимого таблицы  
 JTable tempTable= null;  
  
 switch (str) {  
 case "students": {  
 try { //нужен блок try … catch для работы с SQL-запросами  
 rs2.last(); //переходим на последнюю запись результата SQL-запроса  
 rowCount= rs2.getRow(); //считаем кол-во записей в результате SQL-запроса  
 data=new String[rowCount][7]; //создаем массив строк для содержимого таблицы  
 System.out.println("RowsCount="+rowCount); //вывод в консоль кол-ва строк  
 rs2.beforeFirst(); //переходим на место в результате запроса перед первой записью  
 while (rs2.next()) { //цикл пока есть записи в результате запроса  
 //цикл от 2-й до 7-й колонки в таблице, потому что первая колонка – id,  
 for (int i = 2; i <8; i++) { //а мы не будем его выводить  
 //в массив data в соответствующую позицию записываем данные из результатов запроса  
 data[rs2.getRow()-1][i-2]=rs2.getString(i);  
 //контрольный вывод в консоль  
 System.out.print(data[rs2.getRow()-1][i-2]+" ");  
 }  
 //дополнительный ResultSet для получения названия университета из его id  
 ResultSet tempRS=fromDB("select \* from university where \"univ\_id\"=" +  
 rs2.getString("univ\_id"));  
 tempRS.next(); // переходим на первую запись в объекте tempRS  
 //в строку university записываем значение колонки univ\_name из таблицы university  
 String university=tempRS.getString("univ\_name");  
 // в последний элемент текущей строки массива записываем название университета  
 data[rs2.getRow()-1][6]=university;  
 System.out.print(data[rs2.getRow()-1][6]+" "); //вывод в консоль  
 System.out.println();  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 //массив строк с названиями столбцов  
 String[] columnNames = {"Фамилия","Имя","Стипендия","Курс","Город","Дата рождения",  
 "Университет"};  
 tempTable=new JTable(data,columnNames);  
 return tempTable;  
 }  
  
 case "lecturer": {  
 try {  
 rs2.last();  
 rowCount= rs2.getRow();  
 data=new String[rowCount][4];  
 System.out.println("RowsCount="+rowCount);  
 rs2.beforeFirst();  
 while (rs2.next()) {  
 for (int i = 2; i <5; i++) {  
 data[rs2.getRow()-1][i-2]=rs2.getString(i);  
 System.out.print(data[rs2.getRow()-1][i-2]+" ");  
 }  
 ResultSet tempRS=fromDB("select \* from university where \"univ\_id\"=" +  
 rs2.getString("univ\_id"));  
 tempRS.next();  
 String university=tempRS.getString("univ\_name");  
 data[rs2.getRow()-1][3]=university;  
 System.out.print(data[rs2.getRow()-1][3]+" ");  
 System.out.println();  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 String[] columnNames = {"Фамилия","Имя", "Город", "Университет"};  
 tempTable=new JTable(data,columnNames);  
 return tempTable;  
 }  
  
 case "exam\_marks": {  
 try {  
 rs2.last();  
 rowCount= rs2.getRow();  
 data=new String[rowCount][4];  
 System.out.println("RowsCount="+rowCount);  
 rs2.beforeFirst();  
 while (rs2.next()) {  
 for (int i = 2; i <6; i++) {  
 data[rs2.getRow()-1][i-2]=rs2.getString(i);  
 System.out.print(data[rs2.getRow()-1][i-2]+" ");  
 }  
  
 ResultSet tempRS1=fromDB("select \* from student where \"student\_id\"=" +  
 rs2.getString("student\_id"));  
 tempRS1.next(); // переходим на первую запись в объекте tempRS  
 //в строку university записываем значение колонки univ\_name из таблицы university  
  
 String surname=tempRS1.getString("surname");  
 // в последний элемент текущей строки массива записываем название университета  
 data[rs2.getRow()-1][0]=surname;  
  
  
 ResultSet tempRS2=fromDB("select \* from subject where \"subj\_id\"=" +  
 rs2.getString("subj\_id"));  
 tempRS2.next(); // переходим на первую запись в объекте tempRS  
 //в строку university записываем значение колонки univ\_name из таблицы university  
 String subject=tempRS2.getString("subj\_name");  
 // в последний элемент текущей строки массива записываем название университета  
 data[rs2.getRow()-1][1]=subject;  
  
 System.out.print(data[rs2.getRow()-1][3]+" ");  
 System.out.println();  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 String[] columnNames = {"Фамилия","Предмет", "Оценка", "Дата экзамена"};  
 tempTable=new JTable(data,columnNames);  
 return tempTable;  
 }  
  
 case "subject": {  
 try { //нужен блок try … catch для работы с SQL-запросами  
 rs2.last(); //переходим на последнюю запись результата SQL-запроса  
 rowCount= rs2.getRow(); //считаем кол-во записей в результате SQL-запроса  
 data=new String[rowCount][3]; //создаем массив строк для содержимого таблицы  
 System.out.println("RowsCount="+rowCount); //вывод в консоль кол-ва строк  
 rs2.beforeFirst(); //переходим на место в результате запроса перед первой записью  
 while (rs2.next()) { //цикл пока есть записи в результате запроса  
 //цикл от 2-й до 7-й колонки в таблице, потому что первая колонка – id,  
 for (int i = 2; i <5; i++) { //а мы не будем его выводить  
 //в массив data в соответствующую позицию записываем данные из результатов запроса  
 data[rs2.getRow()-1][i-2]=rs2.getString(i);  
 //контрольный вывод в консоль  
 System.out.print(data[rs2.getRow()-1][i-2]+" ");  
 }  
 //дополнительный ResultSet для получения названия университета из его id  
  
 System.out.print(data[rs2.getRow()-1][2]+" "); //вывод в консоль  
 System.out.println();  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 //массив строк с названиями столбцов  
 String[] columnNames = {"Предмет","Кол-во часов","Кол-во семестров"};  
 tempTable=new JTable(data,columnNames);  
 return tempTable;  
 }  
  
 case "university": {  
 try { //нужен блок try … catch для работы с SQL-запросами  
 rs2.last(); //переходим на последнюю запись результата SQL-запроса  
 rowCount= rs2.getRow(); //считаем кол-во записей в результате SQL-запроса  
 data=new String[rowCount][3]; //создаем массив строк для содержимого таблицы  
 System.out.println("RowsCount="+rowCount); //вывод в консоль кол-ва строк  
 rs2.beforeFirst(); //переходим на место в результате запроса перед первой записью  
 while (rs2.next()) { //цикл пока есть записи в результате запроса  
 //цикл от 2-й до 7-й колонки в таблице, потому что первая колонка – id,  
 for (int i = 2; i <5; i++) { //а мы не будем его выводить  
 //в массив data в соответствующую позицию записываем данные из результатов запроса  
 data[rs2.getRow()-1][i-2]=rs2.getString(i);  
 //контрольный вывод в консоль  
 System.out.print(data[rs2.getRow()-1][i-2]+" ");  
 }  
 //дополнительный ResultSet для получения названия университета из его id  
  
 System.out.print(data[rs2.getRow()-1][2]+" "); //вывод в консоль  
 System.out.println();  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 //массив строк с названиями столбцов  
 String[] columnNames = {"Название","Рейтинг","Город"};  
 tempTable=new JTable(data,columnNames);  
 return tempTable;  
 }  
  
 case "subj\_lect": {  
 try { //нужен блок try … catch для работы с SQL-запросами  
 rs2.last(); //переходим на последнюю запись результата SQL-запроса  
 rowCount= rs2.getRow(); //считаем кол-во записей в результате SQL-запроса  
 data=new String[rowCount][2]; //создаем массив строк для содержимого таблицы  
 System.out.println("RowsCount="+rowCount); //вывод в консоль кол-ва строк  
 rs2.beforeFirst(); //переходим на место в результате запроса перед первой записью  
 while (rs2.next()) { //цикл пока есть записи в результате запроса  
 //цикл от 2-й до 7-й колонки в таблице, потому что первая колонка – id,  
  
 //дополнительный ResultSet для получения названия университета из его id  
 ResultSet tempR1=fromDB("select \* from lecturer where \"lecturer\_id\"=" +  
 rs2.getString("lecturer\_id"));  
 tempR1.next(); // переходим на первую запись в объекте tempRS  
 //в строку university записываем значение колонки univ\_name из таблицы university  
 String surname=tempR1.getString("surname");  
 // в последний элемент текущей строки массива записываем название университета  
 data[rs2.getRow()-1][0]=surname;  
  
  
 ResultSet tempR2=fromDB("select \* from subject where \"subj\_id\"=" +  
 rs2.getString("subj\_id"));  
 tempR2.next(); // переходим на первую запись в объекте tempRS  
 //в строку university записываем значение колонки univ\_name из таблицы university  
 String subject=tempR2.getString("subj\_name");  
 // в последний элемент текущей строки массива записываем название университета  
 data[rs2.getRow()-1][1]=subject;  
 System.out.print(data[rs2.getRow()-1][1]+" "); //вывод в консоль  
 System.out.println();  
 }  
 } catch (SQLException e) {  
 e.printStackTrace();  
 }  
 //массив строк с названиями столбцов  
 String[] columnNames = {"Лектор","Предмет"};  
 tempTable=new JTable(data,columnNames);  
 return tempTable;  
 }  
  
 default: {  
 break;  
 }  
 }  
  
  
 //создаем таблицу из массива с данными и массива с названиями столбцов  
 return tempTable; //возвращаем созданную таблицу  
 }  
  
 private static ResultSet fromDB(String sqlString){//функция для работы с БД  
 ResultSet localRS=null; //локальный объект для результатов запроса  
 conn=null; // объект для связи с БД  
 stmt=null; //объект для создания SQL-запросов  
 try { //нужен блок try … catch для работы с БД  
 Class.forName("org.h2.Driver"); //подгружаем драйвер для H2  
 try { //еще один блок try … catch  
// получаем доступ к БД jdbc:h2:file://d:\\test\\my\_univer  
 conn= DriverManager.getConnection("jdbc:h2:file://c:\\test\\my\_univer","sa","");  
 //получаем объект для выполнения команд SQL  
 stmt=conn.createStatement(ResultSet.TYPE\_SCROLL\_INSENSITIVE,  
 ResultSet.CONCUR\_UPDATABLE);  
//объекту localRS присваиваем результат выполнения команды SQL  
 localRS=stmt.executeQuery(sqlString);  
  
 } catch (SQLException ex) { //обрабатываем возможные ошибки  
 Logger.getLogger(MainForm.class.getName()).log(Level.SEVERE, null, ex);  
 System.out.println("Trouble with query!!");  
 }  
 } catch (ClassNotFoundException ex) { //обрабатываем возможные ошибки  
 Logger.getLogger(MainForm.class.getName()).log(Level.SEVERE, null, ex);  
 System.out.println("Trouble with connection!!");  
 }  
 return localRS; //возвращаем результат  
 }  
}

3. Пример работы программы

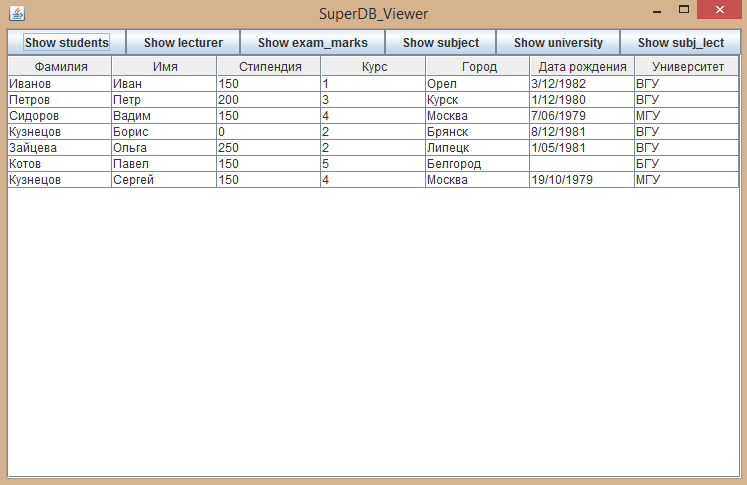


Рис. 1

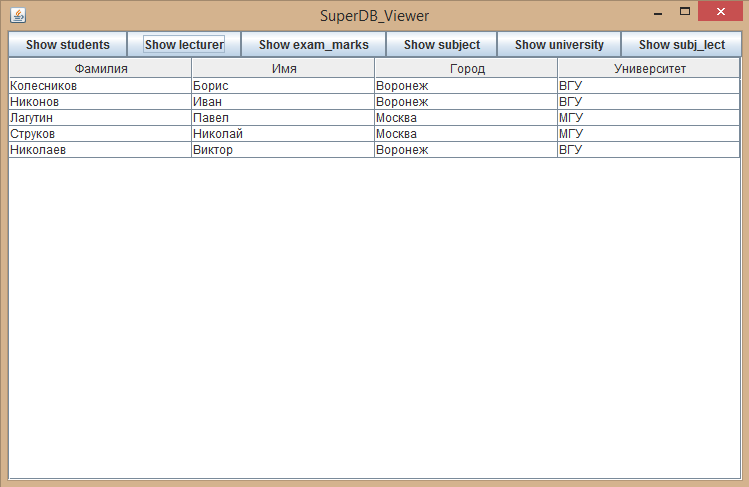


Рис. 2

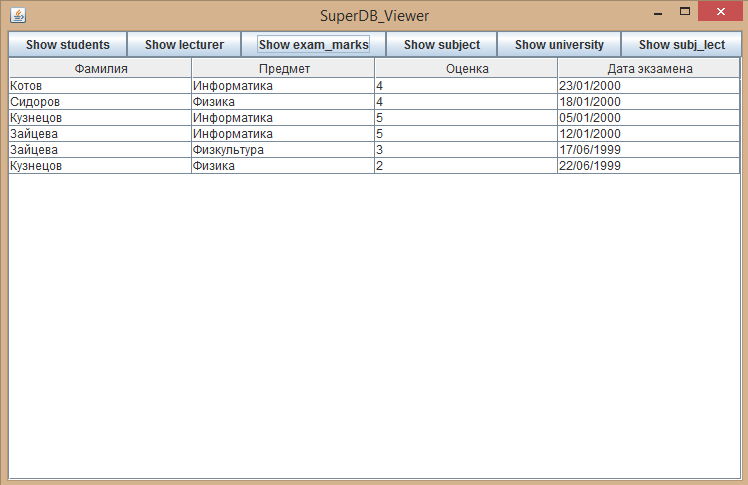


Рис. 3

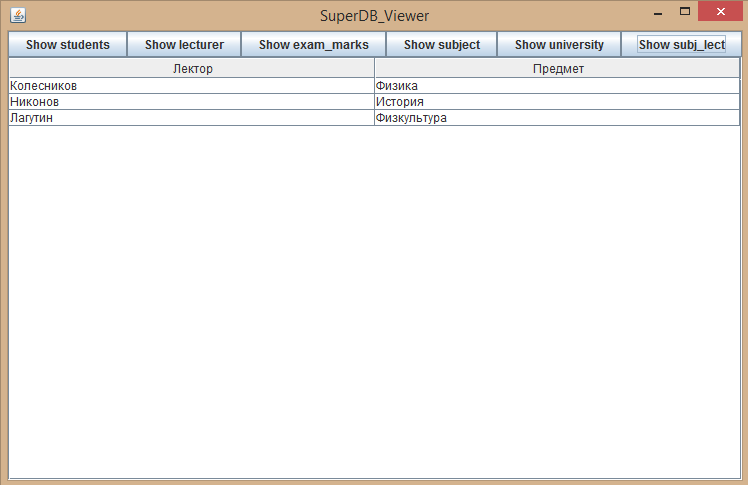


Рис. 4

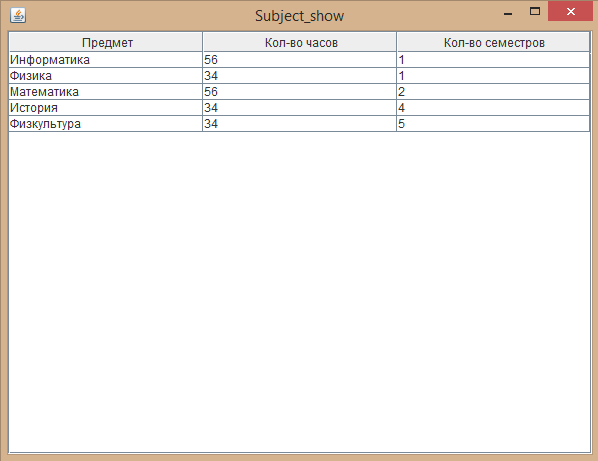


Рис. 5

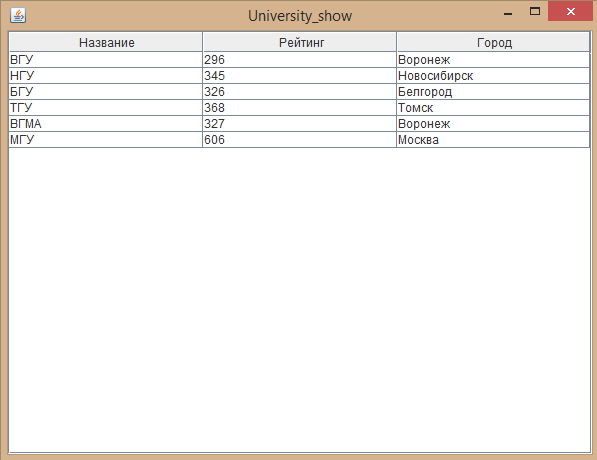


Рис. 6