Министерство образования и науки Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение науки и высшего образования**

«Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

**Институт информационных технологий и радиотехники**

**(ИИТР)**

Кафедра информационных систем и программной инженерии

**Лабораторная работа № 01**

**по дисциплине**

**«Распределённые программные системы»**

**Тема «Работа с реляционной СУБД**

**средствами JDBC API.»**

Выполнил:

ст. гр. ПРИ-117

Хлызова В.Г.

Принял:

Трифонов Д.А.

Владимир, 2019

ЦЕЛЬ РАБОТЫ

Изучить средства JDBC API для работы с реляционными СУБД и получить навыки их практического использования.

ВЫПОЛНЕНИЕ РАБОТЫ

1. Скрипт создания бд:

create database lab02 default charset utf8;

use lab02;

create table Grooup (

id int(11) not null auto\_increment primary key,

groupName varchar(100) not null,

creationDate date not null,

curatorName varchar(100) not null

);

create table Student (

id int(11) not null auto\_increment primary key,

grooupID int(11) not null,

lastName varchar(100) not null,

firstName varchar(100) not null,

birthDate date not null,

age int(2),

foreign key (grooupID) references Grooup(id)

);

1. Скрипт начального заполнения:

insert into Grooup

(groupName, creationDate, curatorName )

values

('ПРИ-117', '2017-08-27', 'Иванов В.Г.'),

('ИСТ-117', '2017-08-27', 'Иванов В.Г.'),

('ИБ-119', '2019-08-29', 'Петров В.Г.')

;

insert into Student

( grooupID, lastName, firstName, birthDate,age)

values

(1, 'Михайлов', 'Михаил', '1999-02-13', 20),

(2, 'Павлов', 'Павел', '1999-04-03', 20),

(1, 'Марьева', 'Мария', '1999-01-19', 20),

(3, 'Николаев', 'Николай', '2000-03-09', 19),

(3, 'Дмитриев', 'Дмитрий', '2000-12-09', 19),

(2, 'Романов', 'Роман', '1999-03-09', 20)

;

1. Скрипт очистки бд:

drop table if exists Grooup;

drop table if exists Student;

drop schema if exists lab02;

1. Код Java-приложения:

**package** ru.arranay.lab02.core;  
  
**import** java.sql.\*;  
**import** java.util.Scanner;  
**import** java.text.SimpleDateFormat;  
**import** java.util.Date;  
**import** java.text.ParseException;  
**import** java.util.Properties;  
**import** java.io.InputStream;  
**import** java.io.PrintStream;  
**import** java.io.IOException;  
**import** java.io.UnsupportedEncodingException;  
  
**public class** Application {  
  
 **public static final** String ***DEFAULT\_CONSOLE\_ENCODING*** = **"UTF-8"**;  
 **public static final** String ***CONSOLE\_ENCODING\_PROPERTY*** = **"consoleEncoding"**;  
  
 **static** Properties *props* = **new** Properties();  
 **private static final** String ***filename*** = **"application.properties"**;  
 **private static** String *url*= **"db.url"**;  
 **private static** String *username*=**"db.username"**;  
 **private static** String *password*=**"db.password"**;  
 **private static** String *query*;  
  
 **public static void** main(String[] args){  
  
 Application app = **new** Application();  
 app.loadProperties(); app.setConsoleEncoding();  
 *url* = *props*.getProperty(*url*);  
 *username* = *props*.getProperty(*username*);  
 *password* = *props*.getProperty(*password*);  
  
 Scanner in = **new** Scanner(System.***in***);  
 **int** a = 0;  
  
 **while** (a != 7){  
  
 System.***out***.println(**"choose a number:"**);  
 System.***out***.println(**" 1. View group list"**);  
 System.***out***.println(**" 2. View student list"**);  
 System.***out***.println(**" 3. Add student"**);  
 System.***out***.println(**" 4. Add group"**);  
 System.***out***.println(**" 5. Delete student"**);  
 System.***out***.println(**" 6. Student transfer"**);  
 System.***out***.println(**" 7. exit"**);  
  
 a = in.nextInt();  
  
 **switch** (a){  
 **case** 1:  
 *ShowGroupList*();  
 **break**;  
  
 **case** 2:  
 *ShowStudentList*();  
 **break**;  
  
 **case** 3:  
 *AddStudent*();  
 **break**;  
  
 **case** 4:  
 *AddGroup*();  
 **break**;  
  
 **case** 5:  
 *DeleteStudent*();  
 **break**;  
  
 **case** 6:  
 *studentTransfer*();  
 **break**;  
  
 **case** 7:  
 System.***out***.println(**"goodbye"**);  
 **break**;  
  
 **default**:  
 System.***out***.println(**"please try again"**);  
 }  
 }  
 }  
  
 **private static void** ShowGroupList(){  
  
 System.***out***.println(**"GROUP LIST"**);  
 *query* = **"select id, groupName, creationDate, curatorName from grooup"**;  
  
 **try**(Connection con = DriverManager.*getConnection*(*url*, *username*, *password*);  
 Statement stmt = con.createStatement(); ResultSet rs = stmt.executeQuery(*query*);){  
  
 **while** (rs.next()) {  
 System.***out***.print(rs.getInt(**"id"**) + **" "**);  
 System.***out***.print(rs.getString(**"groupName"**) + **" "**);  
 System.***out***.print(rs.getDate(**"creationDate"**) + **" "**);  
 System.***out***.print(rs.getString(**"curatorName"**)+ **" "**);  
 System.***out***.println();  
 }  
 } **catch** (SQLException sqlEx) {  
 sqlEx.printStackTrace();  
 }  
 }  
  
 **private static void** ShowStudentList(){  
 System.***out***.println(**"STUDENT LIST"**);  
 *query* = **"select Student.id, groupName, lastName, firstName, birthDate, age "** +  
 **"from Student inner join grooup "** +  
 **"on Student.grooupID = grooup.Id;"**;  
  
 **try**(Connection con = DriverManager.*getConnection*(*url*, *username*, *password*);  
 Statement stmt = con.createStatement(); ResultSet rs = stmt.executeQuery(*query*);){  
  
 **while** (rs.next()){  
 System.***out***.print(rs.getInt(**"Student.id"**) + **" "**);  
 System.***out***.print(rs.getString(**"groupName"**) + **" "**);  
 System.***out***.print(rs.getString(**"lastName"**) + **" "**);  
 System.***out***.print(rs.getString(**"firstName"**) + **" "**);  
 System.***out***.print(rs.getDate(**"birthDate"**) + **" "**);  
 System.***out***.print(rs.getInt(**"age"**) + **" "**);  
 System.***out***.println();  
 }  
  
 } **catch** (SQLException sqlEx){  
 sqlEx.printStackTrace();  
 }  
 }  
  
 **private static void** AddStudent(){  
  
 Scanner in = **new** Scanner(System.***in***);  
  
 System.***out***.print(**"Enter group id: "**);  
 **int** groupId = in.nextInt();  
  
 System.***out***.print(**"Enter lastname: "**);  
 String lastname = in.next();  
  
 System.***out***.print(**"Enter firstname: "**);  
 String firstname = in.next();  
  
  
 System.***out***.print(**"Enter birthDate: "**);  
 SimpleDateFormat sdf = **new** SimpleDateFormat(**"yyyy-MM-dd"**);  
 Date birthDate = **null**;  
 **try** {  
 String bDate = in.next();  
 birthDate = sdf.parse(bDate);  
 } **catch** (ParseException pe) {  
 System.***out***.println(**"incorrect date "**);  
 }  
  
 System.***out***.print(**"Enter age: "**);  
 **int** age = in.nextInt();  
  
 *query* = **"insert into Student (grooupId, lastName, firstName, birthDate, age) "** +  
 **"values (?, ?, ?, ?, ?)"**;  
  
 **try** (Connection con = DriverManager.*getConnection*(*url*, *username*, *password*)){  
  
 PreparedStatement pStmt = con.prepareStatement(*query*);  
 pStmt.setInt(1, groupId);  
 pStmt.setString(2,lastname);  
 pStmt.setString(3,firstname);  
 pStmt.setDate(4,**new** java.sql.Date(birthDate.getTime()));  
 pStmt.setInt(5,age);  
 **if** (pStmt.executeUpdate()>0) System.***out***.println(**"student added!"**);  
 }**catch** (SQLException sqlEx){  
 sqlEx.printStackTrace();  
 }  
 }  
  
 **private static void** AddGroup(){  
  
 Scanner in = **new** Scanner(System.***in***);  
  
 System.***out***.print(**"Enter groupName: "**);  
 String groupName = in.next();  
  
 System.***out***.print(**"Enter creationDate: "**);  
 SimpleDateFormat sdf = **new** SimpleDateFormat(**"yyyy-MM-dd"**);  
 Date creationDate = **null**;  
 **try** {  
 String bDate = in.next();  
 creationDate = sdf.parse(bDate);  
 } **catch** (ParseException pe) {  
 System.***out***.println(**"incorrect date "**);  
 }  
  
 System.***out***.print(**"Enter curatorName: "**);  
 String curatorName = in.next();  
  
 *query* = **"insert into Grooup (groupName, creationDate, curatorName) "** +  
 **"values (?, ?, ?)"**;  
  
 **try**(Connection con = DriverManager.*getConnection*(*url*, *username*, *password*)){  
  
 PreparedStatement pStmt = con.prepareStatement(*query*);  
 pStmt.setString(1, groupName);  
 pStmt.setDate(2,**new** java.sql.Date(creationDate.getTime()));  
 pStmt.setString(3,curatorName);  
 **if** (pStmt.executeUpdate()>0) System.***out***.println(**"group added!"**);  
 }**catch** (SQLException sqlEx){  
 sqlEx.printStackTrace();  
 }  
 }  
  
 **private static void** DeleteStudent(){  
 *ShowStudentList*();  
 System.***out***.print(**"enter student id: "**);  
 Scanner in = **new** Scanner(System.***in***);  
 **int** id = in.nextInt();  
 *query* = **"delete from student where id= ?"**;  
 **try** (Connection con = DriverManager.*getConnection*(*url*, *username*, *password*)) {  
  
 PreparedStatement pStmt = con.prepareStatement(*query*);  
 pStmt.setInt(1, id);  
 **if** (pStmt.executeUpdate()>0) System.***out***.println(**"student deleted!"**);  
 }**catch** (SQLException sqlEx){  
 sqlEx.printStackTrace();  
 }  
 }  
  
 **private static void** studentTransfer(){  
 *ShowStudentList*();  
 *ShowGroupList*();  
  
 System.***out***.print(**"enter student id: "**);  
 Scanner in = **new** Scanner(System.***in***);  
 **int** idS = in.nextInt();  
  
 System.***out***.print(**"enter group id: "**);  
 **int** idG = in.nextInt();  
  
 *query* = **"update student set grooupID=? where id=?;"**;  
 **try** (Connection con = DriverManager.*getConnection*(*url*, *username*, *password*)) {  
  
 PreparedStatement pStmt = con.prepareStatement(*query*);  
 pStmt.setInt(1, idG);  
 pStmt.setInt(2, idS);  
 **if** (pStmt.executeUpdate()>0) System.***out***.println(**"update!"**);  
 }**catch** (SQLException sqlEx){  
 sqlEx.printStackTrace();  
 }  
 }  
  
 **private void** loadProperties() {  
 InputStream inputStream = **null**;  
 **try**{  
 inputStream = getClass().getClassLoader().getResourceAsStream(***filename***);  
 **this**.*props*.load(inputStream);  
 } **catch** (IOException ioe) {  
 System.***out***.println(**"error connecting to database"**);  
 } **finally** {  
 **try** {  
 **if** (inputStream != **null**) {  
 inputStream.close();  
 }  
 } **catch** (IOException ioe) {  
 }  
 }  
 }  
  
 **private void** setConsoleEncoding() {String consoleEncoding = System.*getProperty*(***CONSOLE\_ENCODING\_PROPERTY***, ***DEFAULT\_CONSOLE\_ENCODING***);  
  
 **try** {System.*setOut*(**new** PrintStream(System.***out***, **true**, consoleEncoding));  
 } **catch** (UnsupportedEncodingException ex) {  
 System.***err***.println(**"Unsupported encoding set for console: "** + consoleEncoding);  
 }  
 }  
}

Параметры подключения к БД находятся в файле настроек в каталоге main/resources/application.properties:

**db.url**=**jdbc:mysql://localhost:3306/lab02?useSSL=false&useUnicode=true&useJDBCCompliantTimezoneShift=true&useLegacyDatetimeCode=false&serverTimezone=UTC  
db.username**=**root  
db.password**=**root**

Pom.xml:

*<?***xml version="1.0" encoding="UTF-8"***?>*<**project xmlns="http://maven.apache.org/POM/4.0.0"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"**>  
 <**modelVersion**>4.0.0</**modelVersion**>  
 <**parent**>  
 <**groupId**>ru.arranay.lab02</**groupId**>  
 <**artifactId**>lab02</**artifactId**>  
 <**version**>1.0-SNAPSHOT</**version**>  
 </**parent**>  
 <**groupId**>ru.arranay.lab02</**groupId**>  
 <**artifactId**>core</**artifactId**>  
 <**version**>1.0-SNAPSHOT</**version**>  
 <**name**>core</**name**>  
  
 <**dependencies**>  
 <**dependency**>  
 <**groupId**>mysql</**groupId**>  
 <**artifactId**>mysql-connector-java</**artifactId**>  
 <**version**>8.0.17</**version**>  
 <**scope**>compile</**scope**>  
 </**dependency**>  
 </**dependencies**>  
  
 <**build**>  
 <**plugins**>  
 <**plugin**>  
 <**groupId**>org.codehaus.mojo</**groupId**>  
 <**artifactId**>exec-maven-plugin</**artifactId**>  
 <**version**>1.6.0</**version**>  
 <**configuration**>  
 <**mainClass**>ru.arranay.lab02.core.Application</**mainClass**>  
 </**configuration**>  
 </**plugin**>  
 </**plugins**>  
 </**build**>  
</**project**>

ТЕСТИРОВАНИЕ

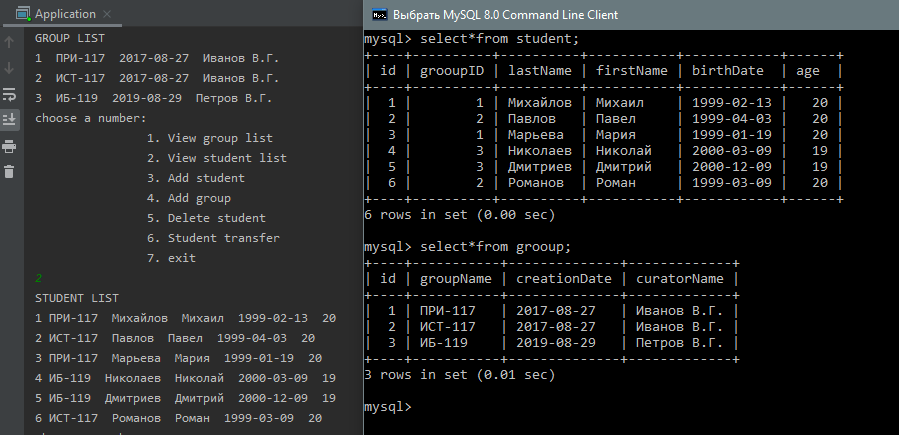


Рисунок 1. Вывод таблиц списка групп и студентов.

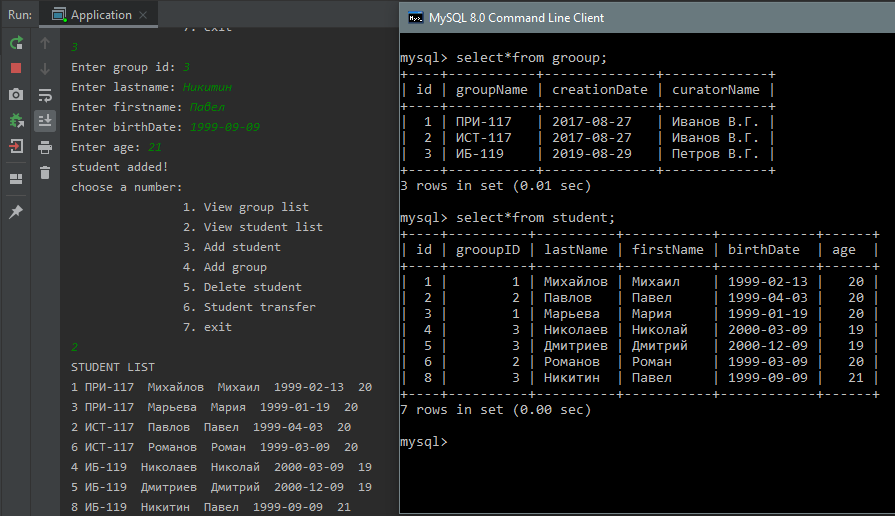


Рисунок 2. Добавление нового студента.

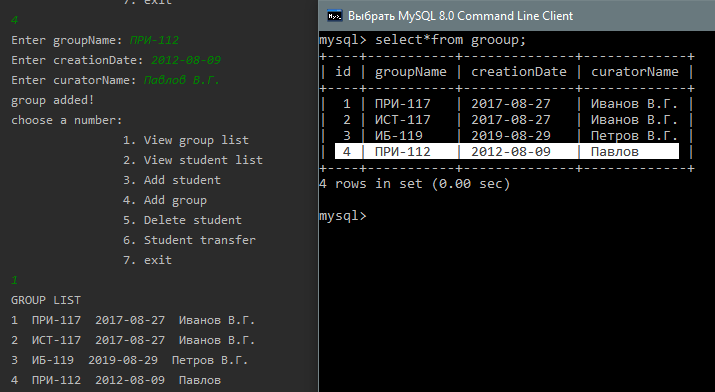


Рисунок 3. Добавление новой группы.

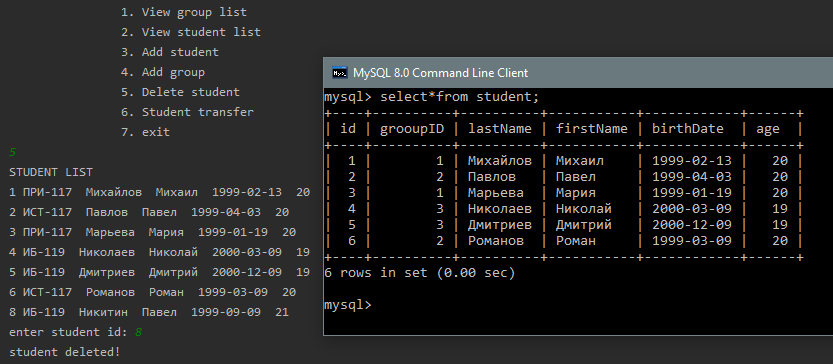


Рисунок 4. Удаление студента.

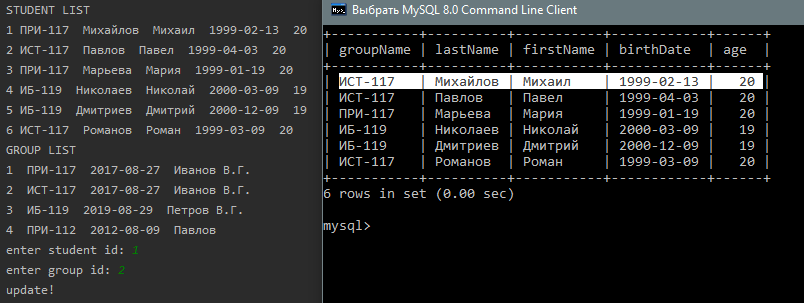


Рисунок 5. Перевод студента в другую группу.

ВЫВОД

В ходе лабораторной работы были изучены средства JDBC API для работы с реляционными СУБД и получены навыки их практического использования.