Частное учреждение образования

«Колледж бизнеса и права»

Тема: «Построение приложений баз данных с использованием JDBC API»

Учащийся М.Н. Гончаров.

группа: Т-992

2022

Задания:

Реализация БД Вариант 6 (Делал через MySql)

Книги – шифр книги, автор, название, год издания, количество экземпляров.

Читатели – читательский билет, фамилия, имя, отчество, адрес.

Выданные книги – шифр книги, читательский билет, дата выдачи, дата

возвращения, дата фактического возвращения.

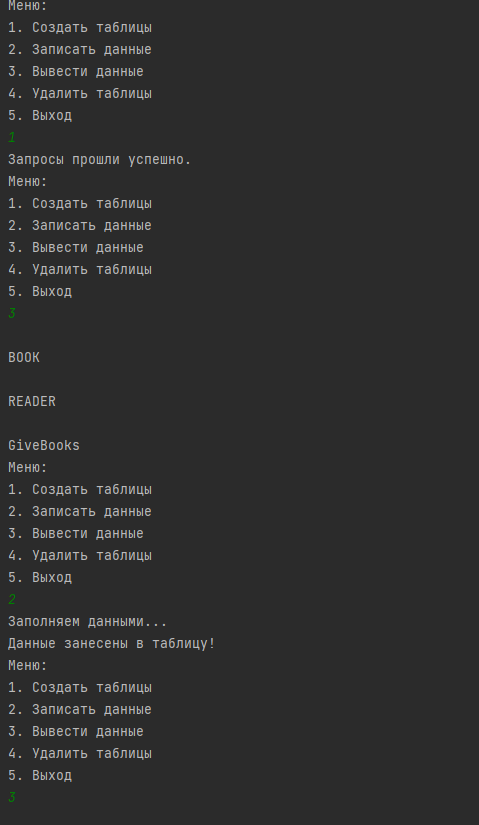
// Main

import java.sql.\*;  
import java.util.Scanner;  
  
public class Main  
{  
 public static void main(String[] args) throws Exception {  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 Scanner in = new Scanner(System.*in*);  
 int p = 0;  
 do {  
 System.*out*.println("Меню:" +  
 "\n1. Создать таблицы" +  
 "\n2. Записать данные" +  
 "\n3. Вывести данные" +  
 "\n4. Удалить таблицы" +  
 "\n5. Выход");  
 p = in.nextInt();  
 switch (p) {  
 case (1):  
 *CreateTable*();  
 break;  
 case (2):  
 *InsertDate*();  
 break;  
 case (3):  
 *ReadDate*();  
 break;  
 case(4):  
 *DeleteTable*();  
 break;  
 }  
 }while (p!=5);  
 }  
 public static void CreateTable()  
 {  
 SqlConnector sql = new SqlConnector();  
 try  
 {  
 sql.*statement*.executeUpdate("CREATE TABLE IF NOT EXISTS Book (" +  
 "indexBook int auto\_increment primary key," +  
 "avtorBook varchar(30) not null," +  
 "nameBook varchar(20) not null," +  
 "yearBook int not null," +  
 "colvoBook int not null)");  
 sql.*statement*.executeUpdate("CREATE TABLE IF NOT EXISTS Reader (" +  
 "indexID int auto\_increment primary key," +  
 "FIO varchar(40) not null," +  
 "address varchar(30) not null)");  
 sql.*statement*.executeUpdate("CREATE TABLE IF NOT EXISTS giveBooks (" +  
 "indexBook\_g int primary key," +  
 "indexBook int not null," +  
 "indexTicket int not null," +  
 "date int)");  
 sql.*statement*.execute("ALTER TABLE giveBooks ADD FOREIGN KEY (indexBook) REFERENCES Book(indexBook)");  
 sql.*statement*.execute("ALTER TABLE giveBooks ADD FOREIGN KEY (indexTicket) REFERENCES Reader(indexID)");  
 System.*out*.println("Запросы прошли успешно.");  
 }  
 catch (SQLException ex)  
 {  
 System.*out*.println("ERROR DB!!");  
 ex.printStackTrace();  
 }  
 }  
 public static void InsertDate ()  
 {  
 SqlConnector sql = new SqlConnector();  
 System.*out*.println("Заполняем данными...");  
 try  
 {  
 sql.*statement*.execute("INSERT INTO Book VALUES(0,'Пушкин','NameBook\_0',1987, 100)");  
 sql.*statement*.execute("INSERT INTO Reader VALUES(0, 'ГМН','Adress\_0')");  
 sql.*statement*.execute("INSERT INTO giveBooks VALUES(1,1,1,1202)");  
 System.*out*.println("Данные занесены в таблицу!");  
 }  
 catch (SQLException ex)  
 {  
 System.*out*.println("ERROR WRITE DB!!!");  
 ex.printStackTrace();  
 }  
 }  
 public static void ReadDate () throws SQLException  
 {  
 try {  
 SqlConnector sql = new SqlConnector();  
 System.*out*.println("\nBOOK");  
 ResultSet resultSet1 = sql.*statement*.executeQuery("SELECT \* FROM Book");  
 while (resultSet1.next()) {  
 System.*out*.printf("%d. %s %s %s \n", resultSet1.getInt(1), resultSet1.getString(2), resultSet1.getString(3), resultSet1.getInt(4), resultSet1.getInt(5));  
 }  
 ResultSet resultSet2 = sql.*statement*.executeQuery("SELECT \* FROM Reader");  
 System.*out*.println("\nREADER");  
 while (resultSet2.next()) {  
 System.*out*.printf("%d. %s %s \n", resultSet2.getInt(1), resultSet2.getString(2), resultSet2.getString(3));  
 }  
 System.*out*.println("\nGiveBooks");  
 ResultSet resultSet3 = sql.*statement*.executeQuery("SELECT \* FROM giveBooks");  
 while (resultSet3.next()) {  
 System.*out*.printf("%d. %d, %d, %d \n", resultSet3.getInt(1), resultSet3.getInt(2), resultSet3.getInt(3), resultSet3.getInt(4));  
 }  
 }  
 catch (SQLException ex)  
 {  
 System.*out*.println("ERROR READ DB!!!");  
 ex.printStackTrace();  
 }  
 }  
 public static void DeleteTable() throws SQLException  
 {  
 try  
 {  
 SqlConnector sql = new SqlConnector();  
 System.*out*.println("Удаляем таблицы...");  
 sql.*statement*.execute("DROP TABLE giveBooks");  
 sql.*statement*.execute("DROP TABLE Reader");  
 sql.*statement*.execute("DROP TABLE Book");  
 }  
 catch(SQLException ex)  
 {  
 System.*out*.println("ERROR DELETE DB!!!");  
 ex.printStackTrace();  
 }  
 }  
}

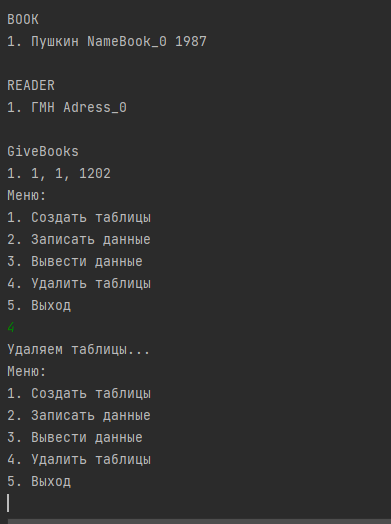
// SqlConnector

import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.SQLException;  
import java.sql.Statement;  
  
public class SqlConnector  
{  
 public static final String *user* = "root";  
 public static final String *pass* = "mypassword";  
 public static final String *url* = "jdbc:mysql://localhost:3306/mysql";  
 public static Statement *statement*;  
 public static Connection *connection*;  
 static  
 {  
 try  
 {  
 *connection*= DriverManager.*getConnection*(*url*, *user*, *pass*);  
 }  
 catch (SQLException throwables)  
 {  
 throwables.printStackTrace();  
 throw new RuntimeException();  
 }  
 }  
 static  
 {  
 try  
 {  
 *statement* = *connection*.createStatement();  
 }  
 catch (SQLException throwables)  
 {  
 throwables.printStackTrace();  
 throw new RuntimeException();  
 }  
 }  
}

// output



Рисунок



Рисунок