Министерство образования Республики Беларусь
Учреждение образования
“Брестский государственный технический университет”
Кафедра ИИТ

Отчёт
По лабораторной работе №9
По дисциплине СПП

Выполнил
Студент группы ПО-3
3-го курса
Кулинкович И. Т.

Проверил
Крощенко А. А.

Брест 2021

Лабораторная работа №9

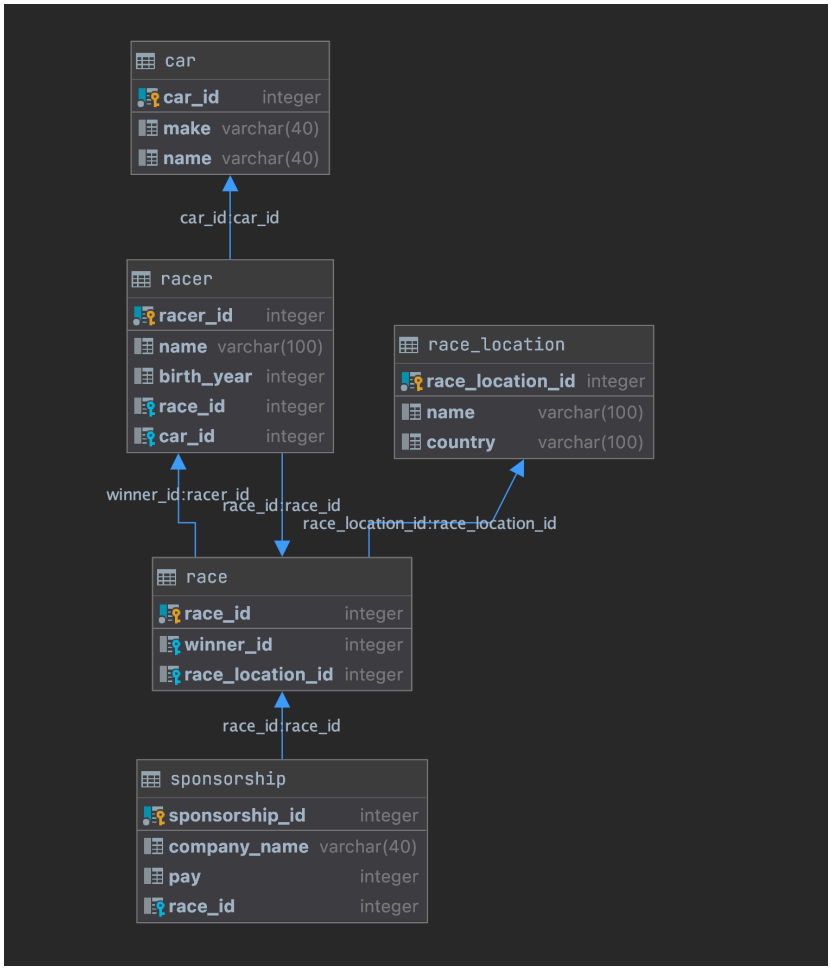
ВАРИАНТ 13

Реализовать базу данных из не менее 5 таблиц на заданную тематику. При реализации продумать типизацию полей и внешние ключи в таблицах. Визуализировать разработанную БД с помощью схемы, на которой отображены все таблицы и связи между ними (пример, схема на рис. 1). На языке Java с использованием JDBC реализовать подключение к БД и выполнить основные типы запросов, продемонстрировать результаты преподавателю и включить тексты составленных запросов в отчет. Основные типы запросов – 1. На выборку/на выборку с упорядочиванием (`SELECT`); 2. На добавление (`INSERT INTO`); 3. На удаление (`DELETE FROM`); 4. На модификацию (`UPDATE`). Базу данные можно реализовать в любой СУБД (MySQL, PostgreSQL, SQLite и др.)

13) База данных *«Формула-1»*

Для разработки выбираем базу данных SQLite и библиотеку sqlite-jdbc.

Структура базы данных



Код программы

live.ilyusha.DBProcess

```
package live.ilyusha;

import java.sql.*;

public class DBProcess {
    private static final String url = "jdbc:sqlite:sample.db";
```

```
    public Connection con;
    private static Statement stmt;
    private static ResultSet rs;
    public Savepoint point;

    public DBProcess() {
        try {
            con = DriverManager.getConnection(url);
            stmt = con.createStatement();
        } catch (SQLException sqlEx) {
            sqlEx.printStackTrace();
        }
    }

    public void insert(String query) throws SQLException {
        stmt.executeUpdate(query);
    }

    public ResultSet select(String query) throws SQLException {
        rs = stmt.executeQuery(query);
        return rs;
    }

    public void delete(String query) throws SQLException {
        stmt.executeUpdate(query);
    }

    public void update(String query) throws SQLException {
        stmt.executeUpdate(query);
    }
}
```

live.ilyusha.Main

```
package live.ilyusha;

import java.sql.ResultSet; import java.sql.SQLException;

public class Main {
    public static void main(String args[]) throws SQLException {
        DBProcess db = new DBProcess();
        ResultSet rs;

        // Database setup
        db.update("create table car ( car_id integer not null constraint
car_pk primary key autoincrement, make varchar(40), name varchar(40) )");
        db.update("create unique index car_car_id_uindex on car (car_id)");
        db.update("create table race_location ( race_location_id integer not
null constraint race_location_pk primary key autoincrement, name
varchar(100), country varchar(100) )");
        db.update("create table race ( race_id integer not null constraint
races_pk primary key autoincrement, winner_id integer references racer
(racer_id), race_location_id integer references race_location )");
        db.update("create unique index races_race_id_uindex on race
(race_id)");
        db.update("create unique index race_location_race_location_id_uindex
on race_location (race_location_id)");
        db.update("create table racer ( racer_id integer not null constraint
racer_pk primary key autoincrement, name varchar(100), birth_year integer,
race_id integer references race, car_id integer references car )");
```

```
        db.update("create unique index racer_racer_id_uindex on racer
(racer_id)");
        db.update("create table sponsorship ( sponsorship_id integer not null
constraint sponsorship_pk primary key autoincrement, company_name
varchar(40), pay integer, race_id integer references race )");
        db.update("create unique index sponsorship_sponsorship_id_uindex on
sponsorship (sponsorship_id)");

        // Fill with test data
        db.insert("insert or ignore into car (car_id, make, name) values (0,
\"Toyota\", \"Supra\")");
        db.insert("insert or ignore into race_location (race_location_id,
name, country) values (0, \"Nürburgring\", \"Germany\")");
        db.insert("insert or ignore into racer (racer_id, name, birth_year,
race_id, car_id) values (0, \"Valtteri Bottas\", 1989, 0, 0)");
        db.insert("insert or ignore into race (race_id, winner_id,
race_location_id) values (0, 0, 0)");
        db.insert("insert or ignore into sponsorship (sponsorship_id,
company_name, pay, race_id) values (0, \"Nissan\", 2900000, 0)");
        db.insert("insert or ignore into car (car_id, make, name) values (1,
\"Bmw\", \"M4\")");

        // Delete
        db.delete("delete from car where car_id = 1;");

        // Update
        db.update("update car set name = \"TF104\" where car_id = 0;");

        // Select
        print(db.select("select racer.name as racerName, car.make as carMake,
car.name as carName from race inner join racer on race.winner_id =
racer.racer_id inner join car on racer.car_id = car.car_id;"));
    }

    static void print(ResultSet rs) throws SQLException {
        System.out.println();

        while (rs.next()) {
            String racerName = rs.getString("racerName");
            String carMake = rs.getString("carMake");
            String carName = rs.getString("carName");

            System.out.println("Racer " + racerName + " won using a " +
carMake + " " + carName + "!");
        }
    }
}
```

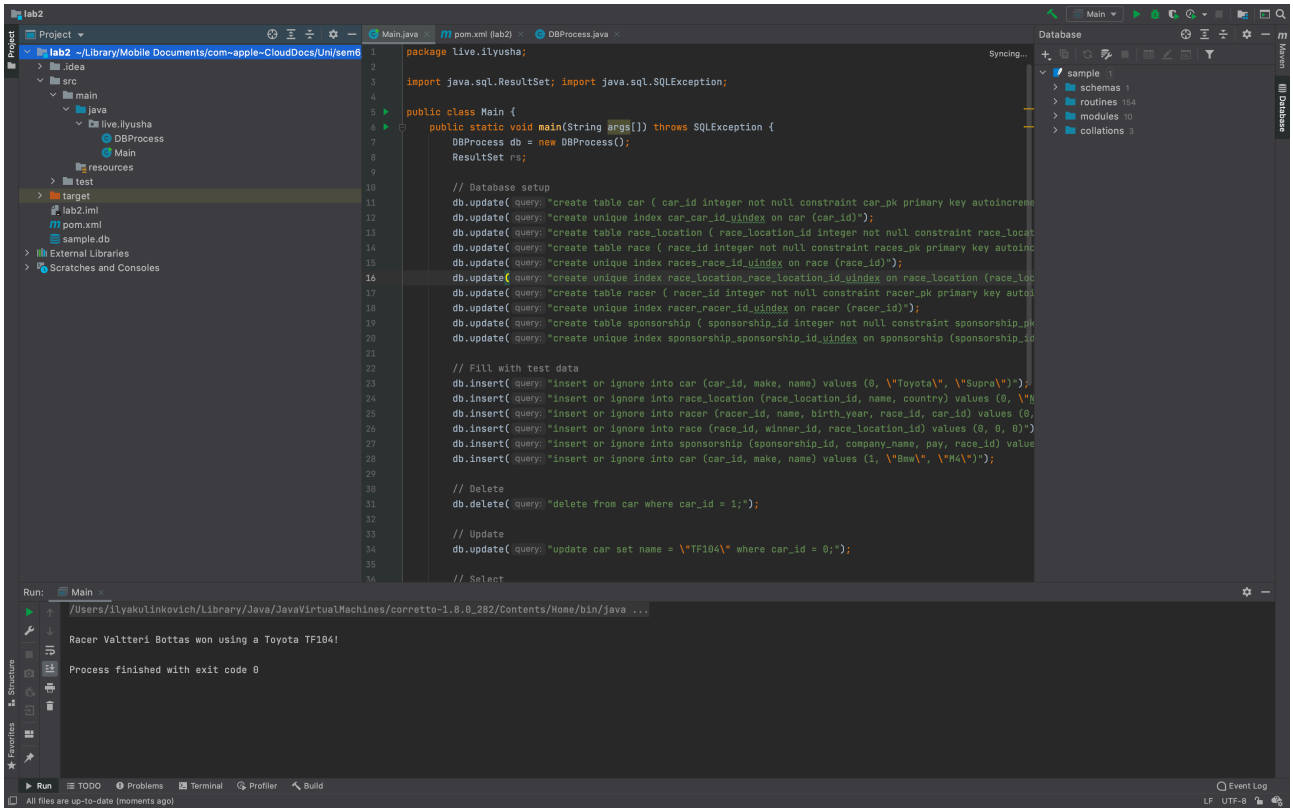
Спецификация ввода

Данные не вводятся

Спецификация вывода

Выводятся полученные из БД данные в формате:
`Racer <name> won using a <car make> <car model name>.`

Рисунки с результатами работы программы



Вывод

В данной лабораторной работе я приобрел практические навыки разработки баз данных и начальной интеграции БД с кодом Java с помощью JDBC.