Федеральное Государственное Бюджетное Образовательное Учреждение Высшего Образования

«Сибирский Государственный Университет   
Телекоммуникаций и Информатики»

Кафедра Прикладной Математики и Кибернетики

(ПМиК)

**РГР по дисциплине**

**«Визуальное программирование и человеко-машинное взаимодействие»**

**Вариант №4**

Выполнил: студент 2 курса  
группы ИП-016  
Кобелев В. С.

Проверил: Милешко А.В.

Новосибирск, 2022

**Содержание:**

1. Задание на РГР 3
2. Ход работы 4
3. Исследование предметной области и создание ER диаграммы 5
4. Перевод ER диаграммы в реляционную модель, создание и заполнение БД 8
5. Проработка визуального интерфейса приложения 10
6. Создание диаграммы классов приложения 12
7. Реализация основного окна приложения 14
8. **Задание на РГР**

Создать ПО для отображения и обработки статистических данных для определённого вида спорта. ПО должно включать 2 основных окна: окно, отображающее таблицы БД со статистической информацией и результаты запросов к БД, переключение таблиц и результатов должно быть реализовано через вкладки; и окно для менеджера запросов к БД.

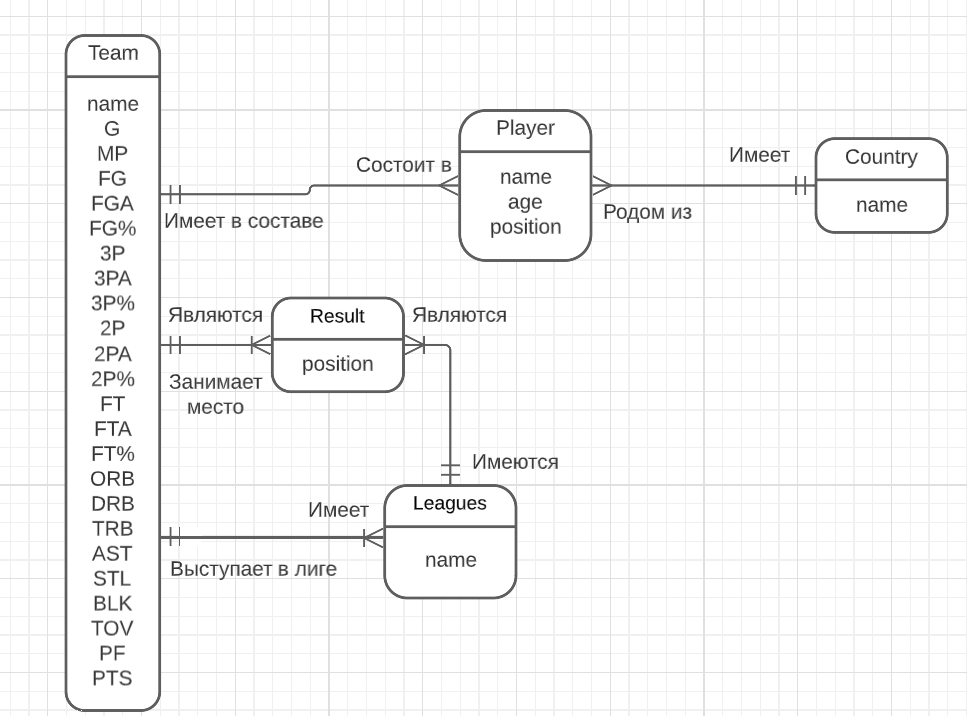
Первое окно должно давать возможность просматривать и изменять все таблицы БД, а также просматривать результаты запросов к БД. Должна иметься возможность удалить вкладки с результатами запросов, но не вкладки с таблицами. Также должна иметься возможность перейти к окну менеджера запросов.

Окно менеджера запросов должно предоставлять интерфейс для создания, сохранения, удаления, редактирования запросов. Созданные запросы должны отображаться в виде списка с названиями запросов, в который можно добавлять новые запросы, удалять, просматривать существующие. Для создания и редактирования запросов должен предоставляться визуальный интерфейс, а не язык запросов. Редактор запросов должен поддерживать операции выборки, соединения, группирования, подзапросы (в качестве подзапроса используются ранее сохранённые запросы).

В качестве базы данных должна использоваться SQLite. Для работы с БД можно использовать SQLite Studio. Все таблицы должны находиться в третьей нормальной форме.

Приложение и отчёт по работе должны быть размещены на GitHub, ссылка на репозиторий отправляется в качестве ответа на задание.

1. **Ход работы**
2. Исследование предметной области и создание ER диаграммы;
3. Перевод ER диаграммы в реляционную модель, создание и заполнение БД;
4. Проработка визуального интерфейса приложения;
5. Создание диаграммы классов приложения;
6. Реализация основного окна приложения;
7. Реализация менеджера запросов;
8. Тестирование и отладка.
9. **Исследование предметной области и создание ER диаграммы**

**

*Рисунок 1. ER диаграмма*

Имеем 5 сущностей:

1. **Player**, состоящая из полей:

|  |  |
| --- | --- |
| name | ФИО игрока |
| age | Возраст игрока |
| position | Позиция игрока на площадке |

*Таблица 1. Поля сущности Player*

1. **Country**, состоящая из полей:

|  |  |
| --- | --- |
| name | Название страны |

*Таблица 2. Поля сущности Country*

1. **Team**, состоящая из полей:

|  |  |
| --- | --- |
| name | Название команды |
| G | Количество игр |
| MP | Количество игрового времени |
| FG | Количество бросков, попавших в корзину |
| FGA | Количество всех выполненных бросков |
| FG% | Процентное соотношение между результативными бросками и общим количеством выполненных бросков |
| 3P | Количество результативных бросков из трёхочковой зоны |
| 3PA | Количество всех выполненных бросков из трёхочковой зоны |
| 3P% | Процентное соотношение между результативными трёхочковыми бросками и общим количеством выполненных трёхочковых бросков |
| 2P | Количество результативных двухочковых бросков |
| 2PA | Количество выполненных двухочковых бросков |
| 2P% | Процентное соотношение между результативными двухочковыми бросками и общим количеством выполненных двухочковых бросков |
| FT | Количество результативных штрафных бросков |
| FTA | Общее количество штрафных бросков |
| FT% | Процентное соотношение между результативными штрафными бросками и общим количеством штрафных бросков |
| ORB | Количество выигранных подборов под корзиной соперника |
| DRB | Количество выигранных подборов под своей корзиной |
| TRB | Полное количество выигранных подборов |
| AST | Количество передач, после которых был выполнен результативный бросок |
| STL | Количество удачных перехватов мяча |
| BLK | Количество удачно выполненных блоков |
| TOV | Количество потерянных мячей |
| PF | Количество персональных замечаний |
| PTS | Количество набранных очков |

*Таблица 3. Поля сущности Team*

1. **Leagues**, состоящая из полей:

|  |  |
| --- | --- |
| name | Название лиги |

*Таблица 4. Поля сущности Leagues*

1. **Result**, состоящая из полей:

|  |  |
| --- | --- |
| position | Позиция в лиге |

*Таблица 5. Поля сущности Result*

Взаимодействие сущностей между собой:

* Игрок родом только из одной страны. Страна является родиной многих игроков;
* Команда может иметь в своём составе множество игроков. Игрок может состоять только в одной команде;
* В лиге может быть множество результатов. Результаты для каждой лиги только одни;
* Команда занимает определённое место. Результат для команды только один.
* Команда выступает множестве лиг. В лиге только одна такая команда.

1. **Перевод ER диаграммы в реляционную модель, создание и заполнение БД**

Для перевода из ER диаграммы в реляционную модель достаточно преобразовать все сущности в таблицы.

Каждый простой тип сущности превращается в таблицу. Имя сущности становится именем таблицы. Каждый атрибут становится столбцом таблицы с тем же именем. Выставляются первичный и внешние ключи.

|  |  |
| --- | --- |
| Player | |
| id\_player | INTEGER | первичный ключ |
| name | STRING |
| age | INTEGER |
| position | STRING |
| id\_county | INTEGER | внешний ключ |
| id\_team | INTEGER | внешний ключ |

*Таблица 6. Таблица БД «Player»*

|  |  |
| --- | --- |
| County | |
| id\_county | INTEGER | первичный ключ |
| name | STRING |

*Таблица 7. Таблица БД «County»*

|  |  |
| --- | --- |
| Team | |
| id\_team | INTEGER | первичный ключ |
| name | STRING |
| G | INTEGER |
| MP | INTEGER |
| FG | DOUBLE |
| FGA | DOUBLE |
| FG% | DOUBLE |
| 3P | DOUBLE |
| 3PA | DOUBLE |
| 3P% | DOUBLE |
| 2P | DOUBLE |
| 2PA | DOUBLE |
| 2P% | DOUBLE |
| FT | DOUBLE |
| FTA | DOUBLE |
| FT% | DOUBLE |
| ORB | DOUBLE |
| DRB | DOUBLE |
| TRB | DOUBLE |
| AST | DOUBLE |
| STL | DOUBLE |
| BLK | DOUBLE |
| TOV | DOUBLE |
| PF | DOUBLE |
| PTS | DOUBLE |

*Таблица 8. Таблица БД «Team»*

|  |  |
| --- | --- |
| League | |
| id\_league | INTEGER | первичный ключ |
| name | STRING |

*Таблица 9. Таблица БД «League»*

|  |  |
| --- | --- |
| Result | |
| id\_league | INTEGER | внешний ключ |
| id\_team | INTEGER | внешний ключ |
| place | INTEGER |

*Таблица 10. Таблица БД «Result»*

Получившаяся БД находится в 3-ей нормальной форме.

1. **Проработка визуального интерфейса приложения**

Главное окно приложения содержит в себе основное меню:

File:

* Save (сохранение БД);
* Load (загрузка БД);
* Exit (выход из приложения).

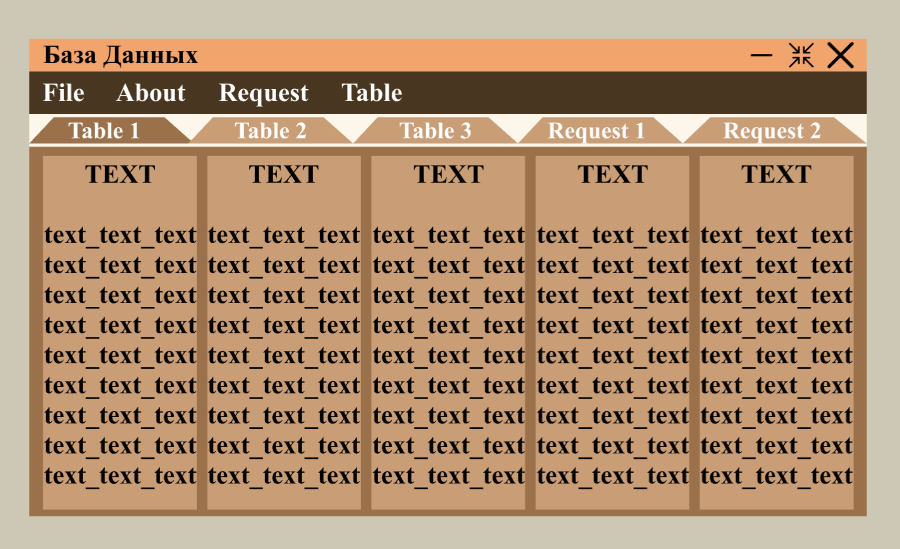
About: кто сделал и как пользоваться.

Request: открывает окно с менеджером запросов.

Table: редактирование БД, удаление и добавление новых строк.

Ниже под меню находятся вкладки с таблицами и результатами запросов.

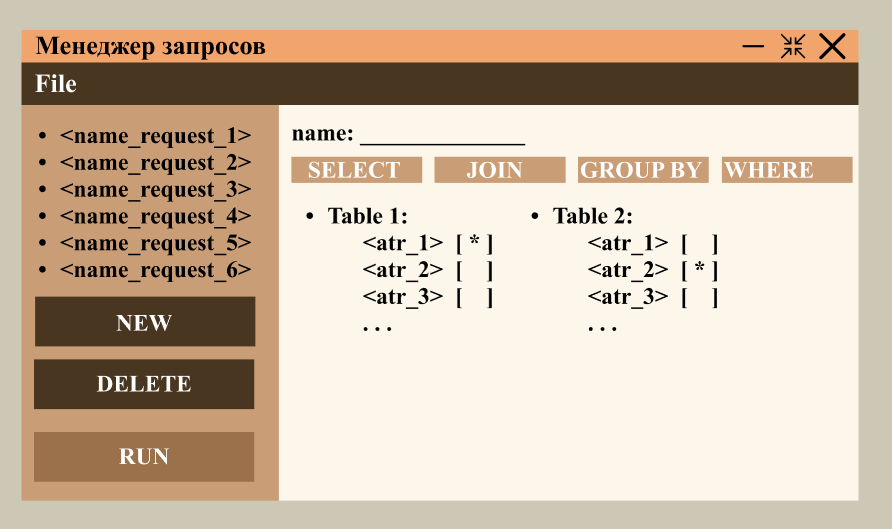
При смене вкладок меняется содержимое, отображаемое в таблице ниже.



*Рисунок 2. Главное окно*

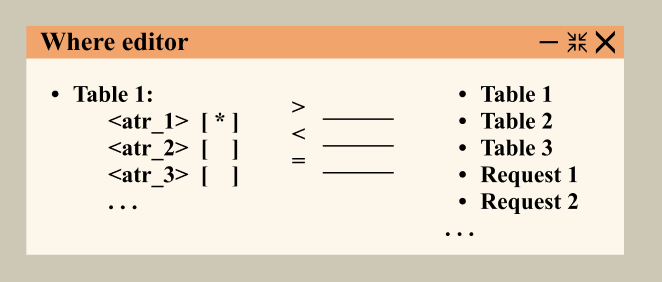
В менеджере запросов слева хранится список запросов. Можно создать новый или удалить. Также можно запустить выполнение запроса.

Справа находится конструктор запроса. Можно ввести имя запроса. Выбрать нужные столбцы таблиц а также выполнить запросы SELECT, JOIN, GROUP BY, WHERE, нажав на соответствующие кнопки.

**

*Рисунок 3. Менеджер запросов*

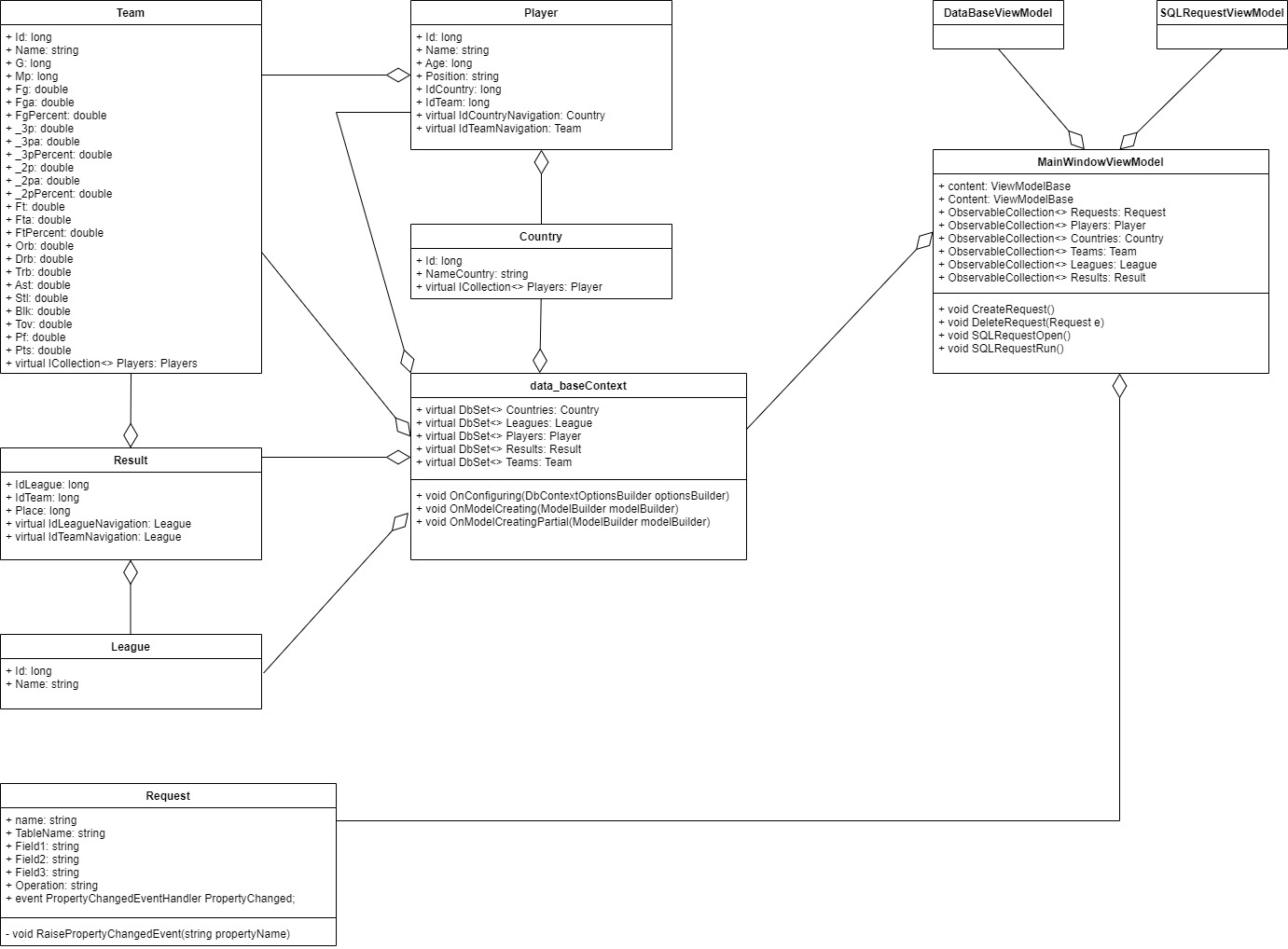
В окошке WHERE editor можно написать условие для работы остальных запросов.



*Рисунок 4. Where editor*

1. **Создание диаграммы классов приложения**

В своей основе классы из слоя Models повторяют ER диаграмму и составленную базу данных. Также добавлен класс Request, описывающий запрос к БД, созданный в конструкторе.



*Рисунок 5. Диаграмма классов*

1. **Реализация основного окна приложения**

