Министерство цифрового развития

Сибирский Государственный Университет Телекоммуникация и Информатики

СибГУТИ

Кафедра прикладной математики и кибернетики

Расчетно-графическая работа.

Лошадиные скачки RSA

Вариант 10

Выполнил: студент 2 курса группы ИП-016

Мосолов Павел Александрович

Преподаватель: Милешко Антон Владимирович

Новосибирск, 2022

Исследование предметной области и создание ER диаграммы

Теоретической составляющая конного спорта Терминология:

1. Owner - юридическое или физическое лицо, обладающее имущественным правом владения **лошадью** или фактически владеющее ею, ответственное за здоровье и использование **лошади**.
2. Trainer - **это** человек, который ухаживает за **лошадьми** и обучает их различным дисциплинам. Некоторые из обязанностей **тренеров** включают заботу о физических потребностях животных, а также обучение их покорному поведению и/или подготовку их к мероприятиям, которые могут включать соревнования и другие цели верховой езды.
3. Jockey - **это** тот, кто ездит на **лошадях** на скачках или скачках с препятствиями, в первую очередь как профессия.
4. Horse - **Лошадь** зоол. крупное непарнокопытное млекопитающее (лат. «Equus caballus»), одомашненное и широко использующееся человеком для передвижения верхом, перевозки тяжестей и т. п.

Как проводятся конные скачки:

Проходят скачки на ипподромах. Как правило, площадки имеют форму круга или овала, но в Англии встречаются и другие варианты — незамкнутые линии в виде букв U или L. Финишная прямая всегда укладывается ровно напротив трибун, а старт, в зависимости от длины дистанции, может быть, как там же, так и на противоположной стороне дорожки.

Для начала гонки используется как условная линия, на которых участники выстраиваются в ряд, так и стартовые ворота — конструкция, шириной во всю дорожку с навесными боксами под каждую лошадь. Таким образом, перед стартом все оказываются в условных клетках, что исключает фальстарты и драки между животными (а такое случается, лошади тоже волнуются, горячатся и могут кого-то «поставить на место»). По сигналу все дверцы ворот открываются и скачка начинается.

Финиш редко представляет собой ленточку, чаще — всю ту же условную линию от финишного столба за краем дорожки. Победителем считается тот скакун, чья голова первой пересекла эту линию. А так как не всегда это возможно определить положение участников на глаз, используется фотофиниш.

ER – диаграмма

1. Jockey – сущность хранящая в себе ID Жокея, ID лошади с которой он взаимодействует, Имя и Пол.
2. Trainer – сущность хранящая в себе ID Тренера, ID лошади с которой он взаимодействует, Имя и Пол.
3. Owner – сущность хранящая в себе ID Владельца, ID лошади с которой он взаимодействует, Имя и Пол.
4. Horse – сущность хранящая в себе ID Жокея, ID Тренера, ID Владельца, ID результата в скачках, Имя и возраст лошади (каждая лошадь соревнуется в своем возрастном диапазоне).
5. Result – самая большая сущность, хранящая в себе все результаты скачек и свой ID.

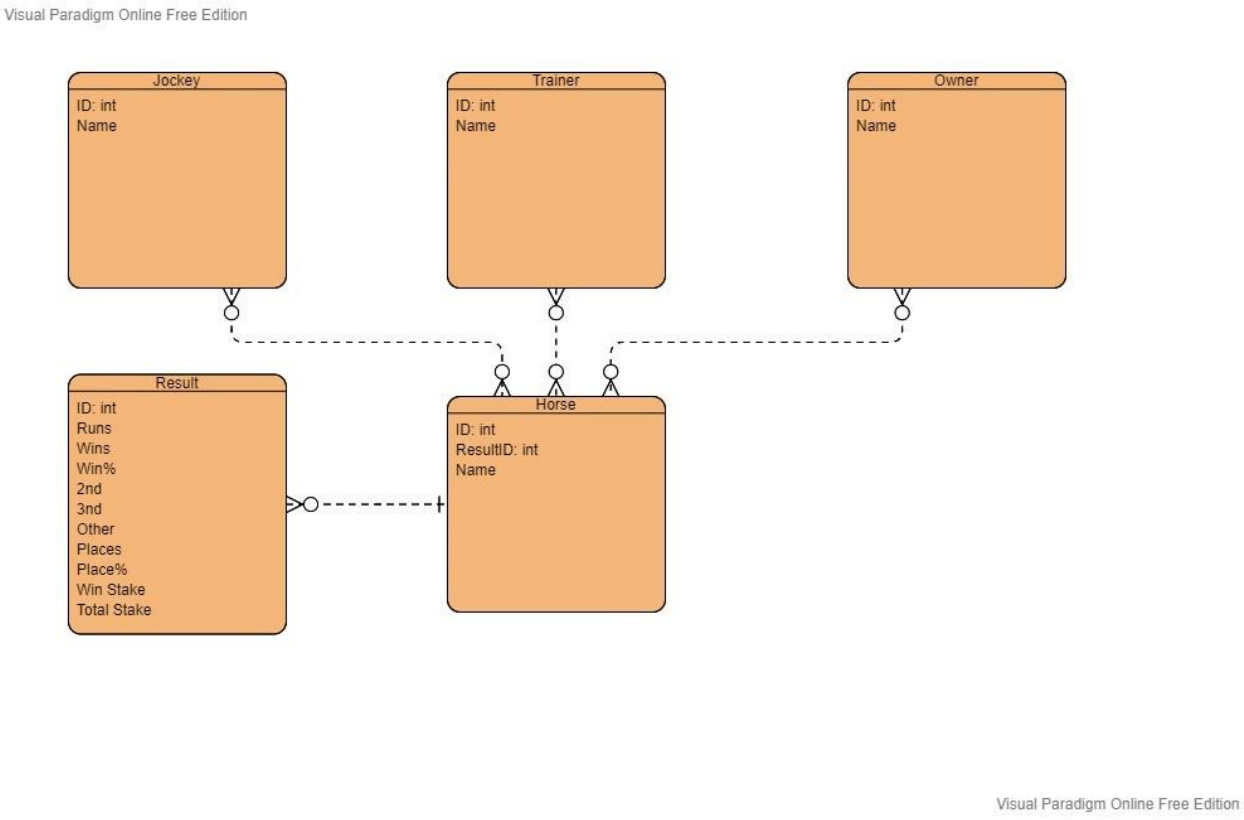

Рис. 1. ER – диаграмма.

Перевод ER диаграммы в реляционную модель, создание и заполнение БД

Для перевода данных из ER диаграммы мы используем программу SQLite Studio. Так же нужно учесть, что все таблицы должны находиться в третьей нормальной форме.

Теория: **Третья нормальная форма (3**NF) - это подход к разработке схемы базы данных для реляционных баз данных, который использует принципы нормализации для уменьшения дублирования данных, предотвращения аномалий данных, обеспечения ссылочной целостности и упрощения управления данными.

Практика: Horse:

Имя	Тип данных	Первичный ключ	Внешний ключ
ID	INTEGER	✓	
Name	STRING		
RESULT_ID	INTEGER		✓

Jockey:

Имя	Тип данных	Первичный ключ	Внешний ключ
ID	INTEGER	✓	
Name	STRING		

Owner:

Имя	Тип данных	Первичный ключ	Внешний ключ
ID	INTEGER	✓	
Name	STRING		

Result:

Имя	Тип данных	Первичный ключ	Внешний ключ
ID	INTEGER	✓	
Runs	INTEGER		
Wins	INTEGER		
Win%	DOUBLE		
2nd	INTEGER		
3nd	INTEGER		
Otner	INTEGER		
Places	INTEGER		
Place%	DOUBLE		
Win Stake	INTEGER		
Total Stake	INTEGER		
Year	INTEGER		

Trainer:

Имя	Тип данных	Первичный ключ	Внешний ключ
ID	INTEGER	✓	
Name	STRING		

Owner_to_Horse:

Имя	Тип данных	Первичный ключ	Внешний ключ
HORSE_ID	INTEGER		✓
OWNER_ID	INTEGER		✓

Jockey_to_Horse:

Имя	Тип данных	Первичный ключ	Внешний ключ
HORSE_ID	INTEGER		✓
OWNER_ID	INTEGER		✓

Trainer_to_Horse:

Имя	Тип данных	Первичный ключ	Внешний ключ
HORSE_ID	INTEGER		✓
TRAINER_ID	INTEGER		✓

Взаимодействие таблиц между собой:

1. Самая главная таблица это “Horse”, она взаимодействует со всеми.
2. Лошадь имеет историю скачек, поэтому принимает таблицу “Result”.
3. Лощадь имеет владельца (человек который купил лощадь), жокея (человек который управляет ей на соревнованиях) и тренер (человек, которые тренирует лошадь для разных видов скачек). Поэтому она взаимодействует со всеми этими таблицами, с помощью дополнительных таблиц.
4. Чтобы добиться 3ей нормальной формы, были созданы дополнительные таблицы: “ Owner_to_Horse”, “ Jockey_to_Horse”, “Trainer_to_Horse ”. Они нужны для взаимодействия в обе стороны.

Проработка визуального интерфейса приложения

Главное окно приложения содержит в себе основное меню:

File:

1. **Save** (сохранение БД);
2. **Load** (загрузка БД);
3. **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.

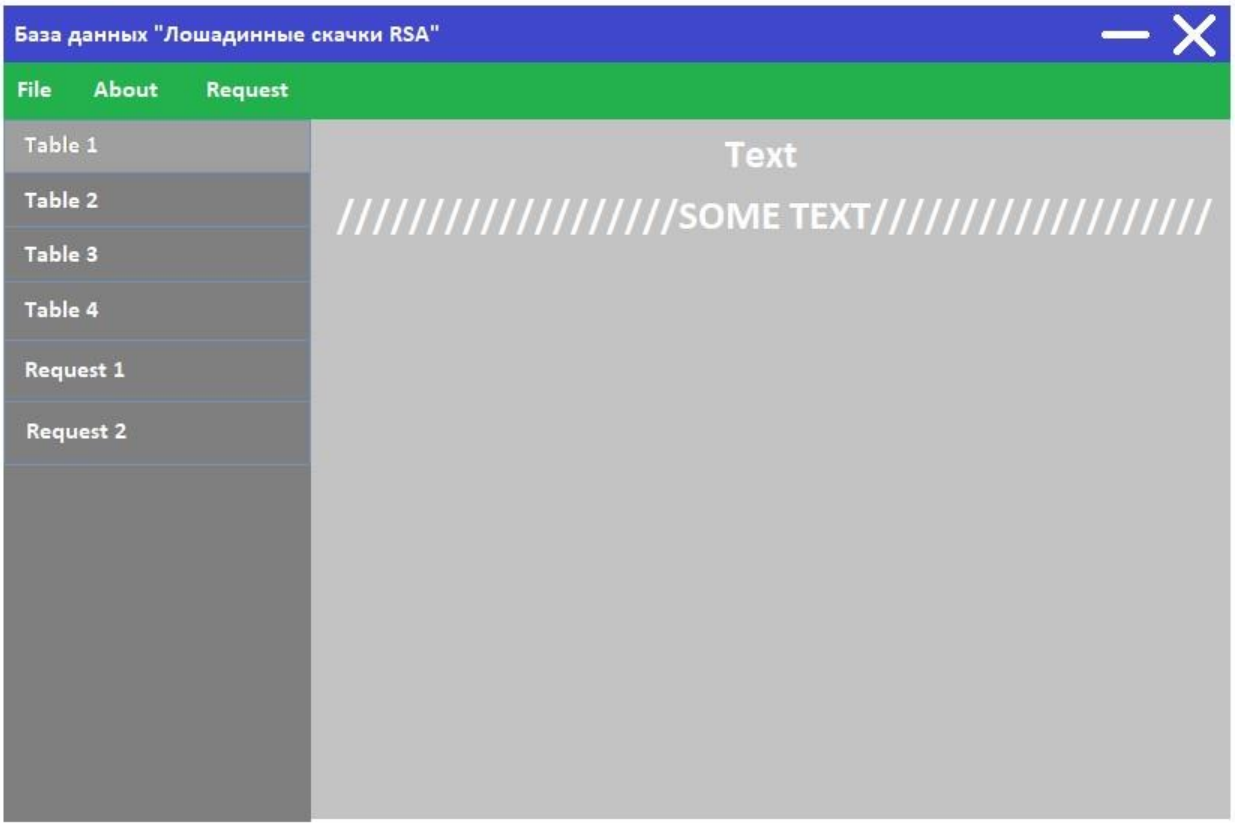

Рис. 2. Главное меню.

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.

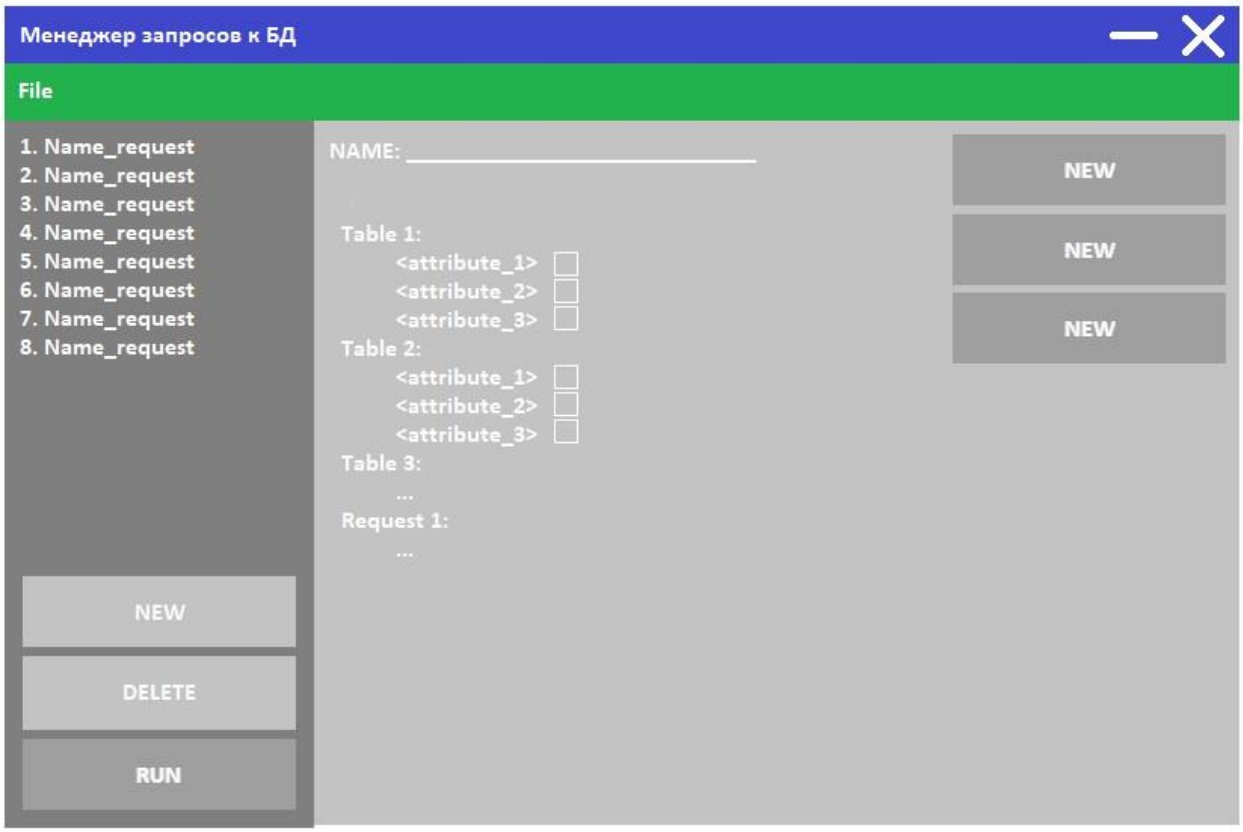

Рис. 3. Менеджер запросов.

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

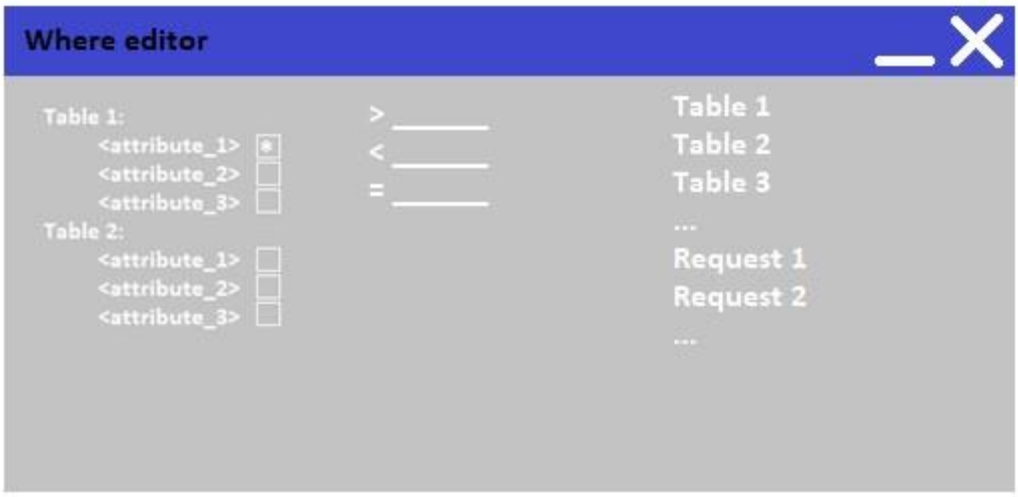

Рис. 4. Окно Where.

Создание диаграммы классов приложения

Диаграмма классов в точности повторяет ER

Диаграмму и составленную базу данных. Для связей между классами добавлены переменные с типом данных другого класса.

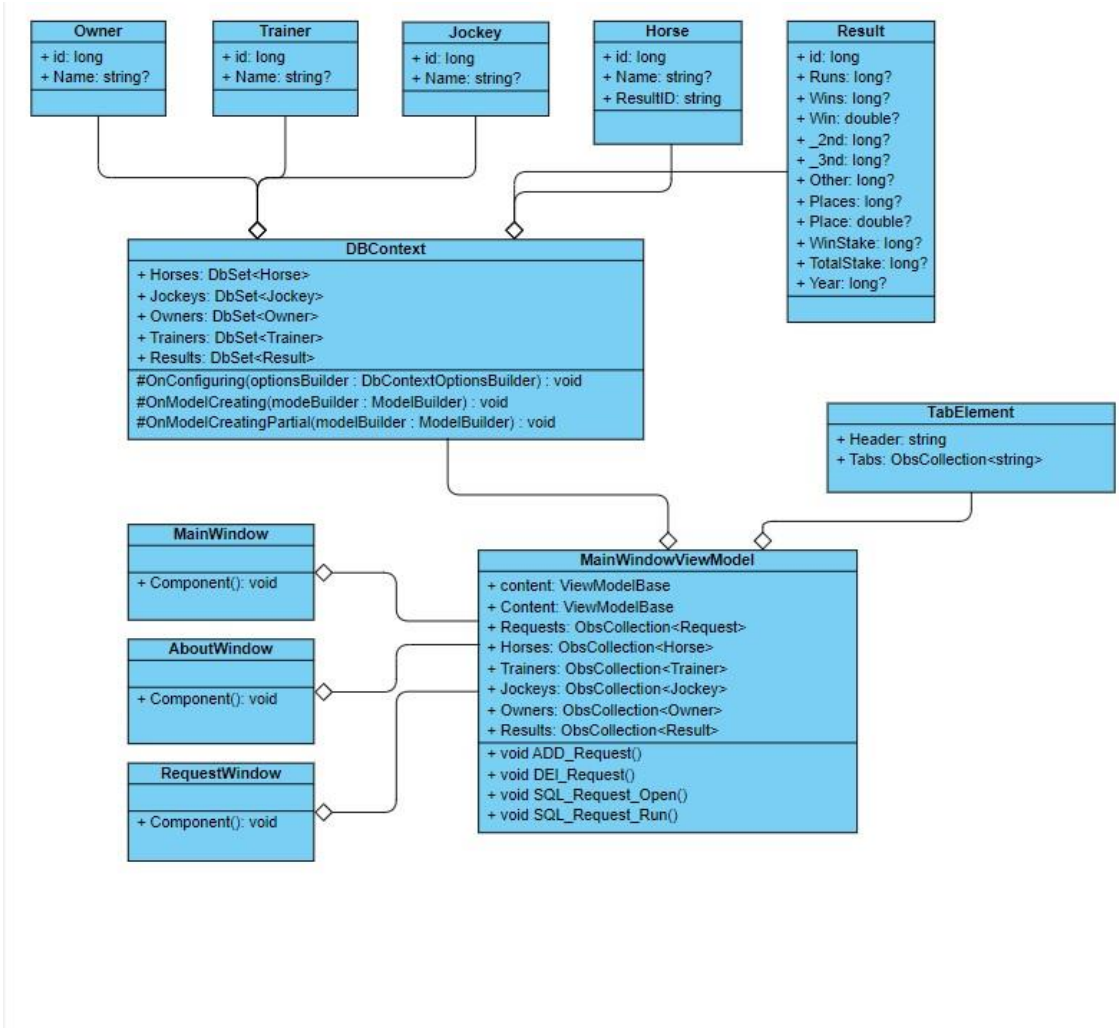

Рис. 5. Диаграмма классов.

Реализация основного окна приложения

В главном меню выводятся таблицы базы данных. Таблицы можно просматривать, изменять и сохранять. Так же тут расположена кнопка “Request Manager” для перехода в меню запросов. Сверху написано меню, в котором реализовано две кнопки. Первая - просмотреть кто сделал работу и вторая – работа с файлом.

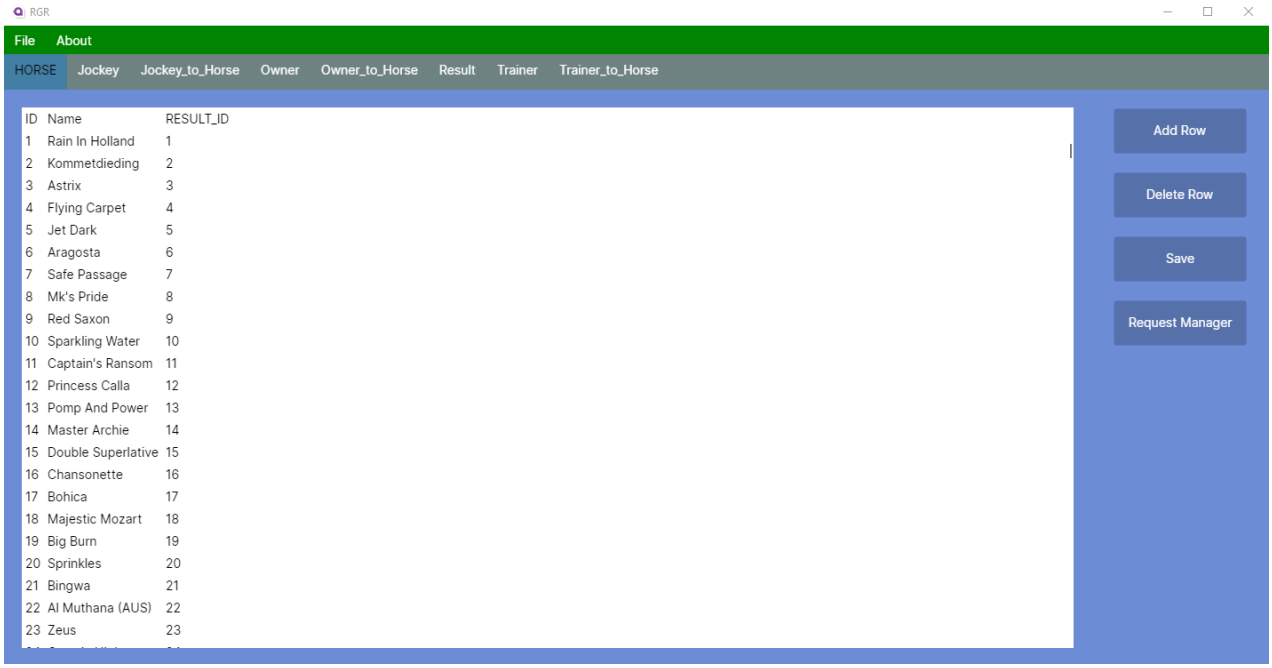

Рис. 6. Главное меню программы.

Рис. 7. “About”.

Реализация менеджера запросов

В менеджере запросов реализованы методы “Delete”, “Join”, “Group”. Есть возможность изменять название запроса, так же запрос будет отображаться в главном меню. Плюс ко всему реализован метод “Where”.

Рис. 8. Менеджер запросов.

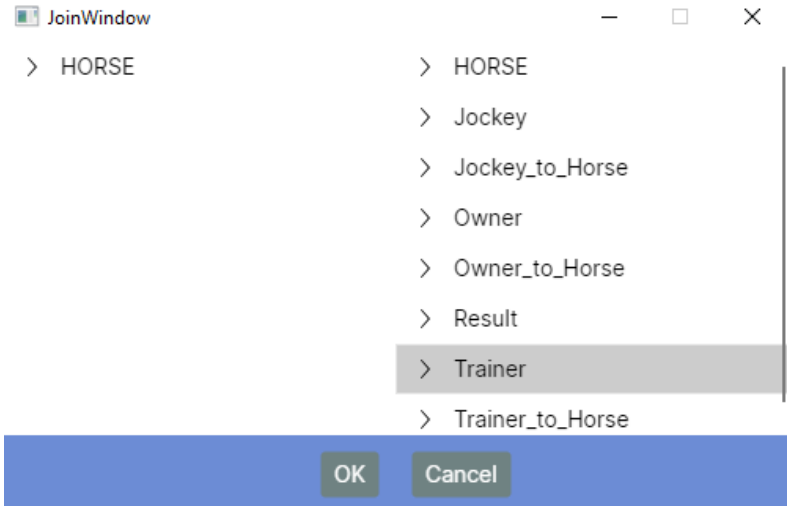

Рис. 9. “Join”.

Вывод

В процессе написания РГР были получены навыки работы с Фреймворком “Avalonia”, а также навыки написания кода на “C#”. Вся программа была проработана досконально, с минимальными недочетами. Было много сложностей в понимании Фреймворка, но я их преодолел и написал РГР.