|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ | | | | | | | | | | |
|  | | | |  | | |  | | | |
| ФГБОУ ВПО «Пермский государственный национальный исследовательский университет» | | | | | | | | | | |
|  | |  | | | | | | |  | |
|  | | ОТЧЕТ  по лабораторной работе «Нестандартный запрос» по дисциплине «Учебная практика по базам данных» | | | | | | |  | |
|  | | |  | | | | |  | | |
|  | Работу выполнил  студент гр. ПМИ-1,2  Васильевых М.Г. \_\_\_\_\_\_\_\_\_\_  (подпись)  «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2021 | | | |  | Проверил  ассистент кафедры МОВС  Турова И.А. \_\_\_\_\_\_\_\_\_\_  (подпись)  «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2021 | | | |  |
|  | |  | | | | | | |  | |
|  | | Пермь 2021 | | | | | | |  | |

Содержание

[1 Постановка задачи 3](#_Toc92981226)

[3 Описание функционала разработанной программы 6](#_Toc92981227)

[3.1. Внешний вид и функционал приложения 6](#_Toc92981228)

[3.2 Общий алгоритм решения задачи 8](#_Toc92981229)

[4 Реализация программы 9](#_Toc92981230)

[5 Тестирование программы 14](#_Toc92981231)

# 1 Постановка задачи

Как должен работать нестандартный запрос

Нестандартный запрос предназначен для генерации «на лету» заранее не описанных запросов к БД на основе прямого использования метаданных о схеме БД либо из сформированных вручную вспомогательных таблиц, описывающих всю или часть схемы БД. Интерфейс нестандартного запроса должен позволять конструировать новые запросы по принципу «не набирать, а выбирать».

Примерный вид графического пользовательского интерфейса:

На вкладке «Поля» – все поля из всех таблиц из последнего задания предыдущей контрольной точки (Задание №8. «Автоматизированное рабочее место»). Названия полей можно сгруппировать по смыслу, отделив друг от друга специальными символами, как показано на рисунке. На данной вкладке пользователь выбирает поля для вывода результатов запроса (заголовки полей в результирующей таблице, т. е. в тексте сгенерированного SQL запроса данный список полей должен непосредственно следовать за ключевым словом SELECT). Например, для выбранных полей на рисунке выше:

SELECT people\_info.fio, people\_info.pol, sogl\_info.doc\_type, goods.name, pep\_good.sale\_date

Далее, пользователь должен иметь возможность наложить определённые условия на результат запроса:

В поле «Имя поля» должны попадать все поля из всех таблиц, а не только те, которые были выбраны на предыдущей вкладке.

В поле «Критерий» - операции сравнения, доступные для типа данных выбранного поля.

В поле «Выражение» должны подчитываться значения выбранного поля «Имя поля» из соответствующей таблицы БД. Условия могут быть связаны через И, ИЛИ. Созданные условия добавляются в раздел запроса WHERE. Например, здесь это будет так:

SELECT people\_info.fio, people\_info.pol, sogl\_info.doc\_type, goods.name, pep\_good.sale\_date

FROM people\_info,solg\_info, goods, pep\_good

WHERE peoples\_info.birth\_date = {01.02.1971} AND peoples\_info.pol = ‘Ж’

Фигурные скобки {} или апострофы ‘’ ставятся в зависимости от типа поля (строковое, дата и т. п.)

Далее, на вкладке «порядок» - возможность задать сортировку по выбранным полям. Данную вкладку реализовывать необязательно.

Результат выбора попадает в раздел запроса ORDER BY. Например,

SELECT people\_info.fio, people\_info.pol, sogl\_info.doc\_type, goods.name, pep\_good.sale\_date

FROM people\_info,solg\_info, goods, pep\_good

WHERE peoples\_info.birth\_date = {01.02.1971} AND peoples\_info.pol = ‘Ж’

ORDER BY peoples\_info.fio ASC, pep\_good.sale\_date ASC

На вкладке «Результат» - результат выполнения запроса

Кроме того, должна быть возможность посмотреть текст созданного SQL-запроса по нажатию на кнопку «Просмотр SQL»

Как видно из текста запроса, в разделе WHERE кроме условий, заданных пользователем, содержатся ещё и условия соединения таблиц по ключевым полям. Это необходимо для

корректного выполнения запроса. Рассмотрим, что необходимо для того, чтобы правильно построить эти условия соединения.

Вспомогательная таблица с метаданными о таблицах БД

Используется на первой вкладке для вывода всех полей всех таблиц БД, а также для учёта типа поля при формировании условия на вкладке «Условия». Таблица содержит два поля с названиями перечисленных полей. Одно используется при формировании части SELECT, а другое, переведённое на русский, для вывода на форму для наглядного описания имён полей.

Вспомогательная таблица с метаданными о связях между таблицами БД

Данная таблица используется для формирования строки соединения между любыми двумя таблицами БД. Как видно на рисунке, сначала здесь прописываются соединения между соседними таблицами (по каким полям соединены соседние таблицы).

А затем, прописываем соединения каждой таблицы с каждой, которые непосредственно не соединены, но между которыми существует путь. Для этого в поле “Via” записываем имя таблицы, через которую они соединены. Например, пусть у нас есть таблицы 1, 2, 3, 4.

Соединены они так: 1<-2<-3<-4

Тогда сначала записываем непосредственные связи таблиц:

1 2 1.code = 2.code\_1

2 3 2.code = 3.code\_2

3 4 3.code = 4.code\_3

Затем прописываем остальные связи:

1 3 соединены через 2

1 4 соединены через 2

2 4 соединены через 3

Допустим, пользователь выбрал в запрос поля из таблиц 1 и 4. Нам нужно соединить эти две таблицы. Для этого сначала ищем в нашей таблице строчку 1 4. Они соединены через 2. Поэтому ищем, как соединены 1 и 2, 2 и 4. 1 и 2 соединены через 1.code = 2.code\_1. Получили первое условие соединения.

Далее, ищем 2 и 4. Они соединены через 3. Ищем 2 и 3, 3 и 4. Получаем ещё условия

2.code = 3.code\_2 и 3.code = 4.code\_3

Соединяем все условия через AND:

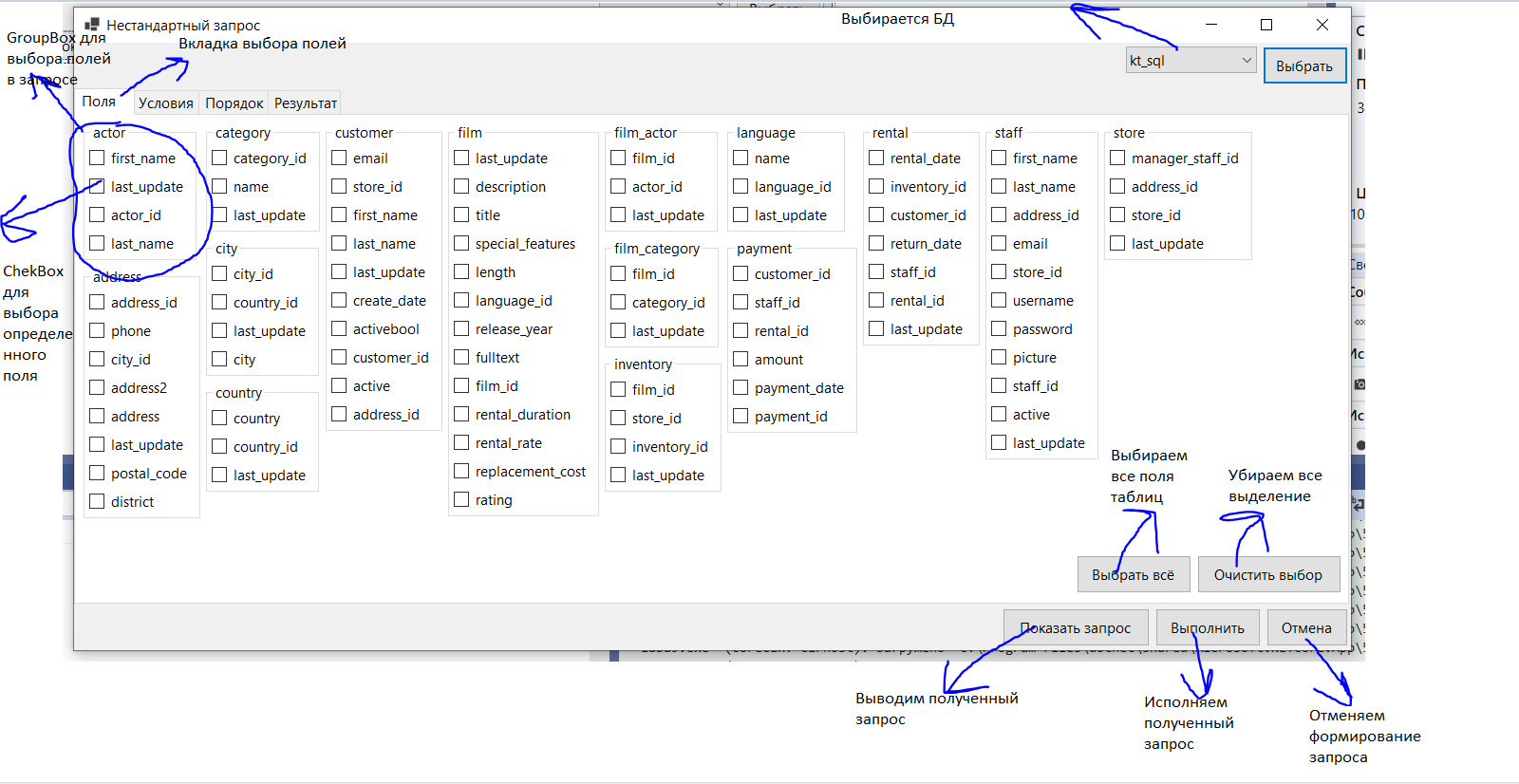
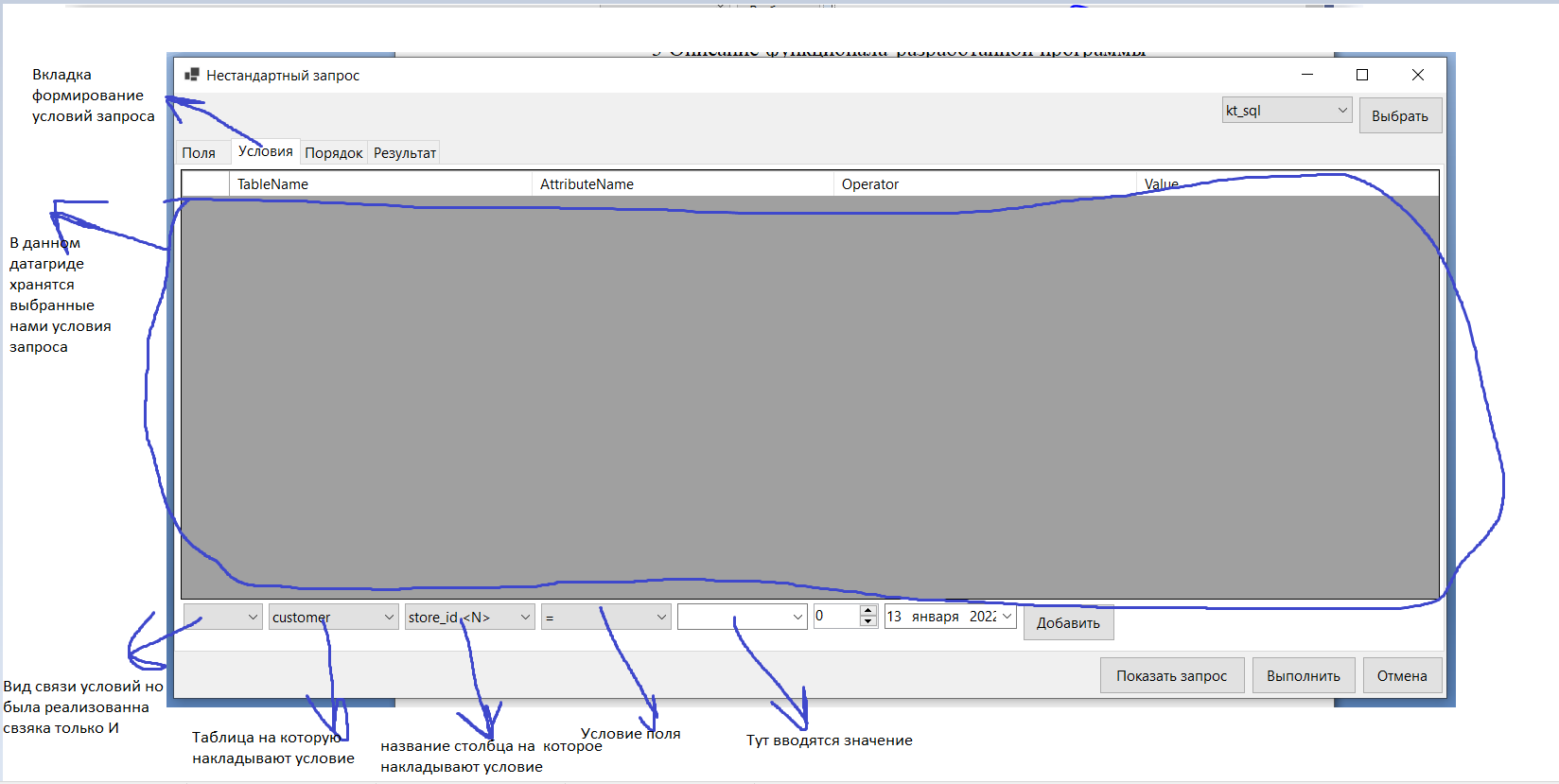
WHERE 1.code = 2.code\_1 AND

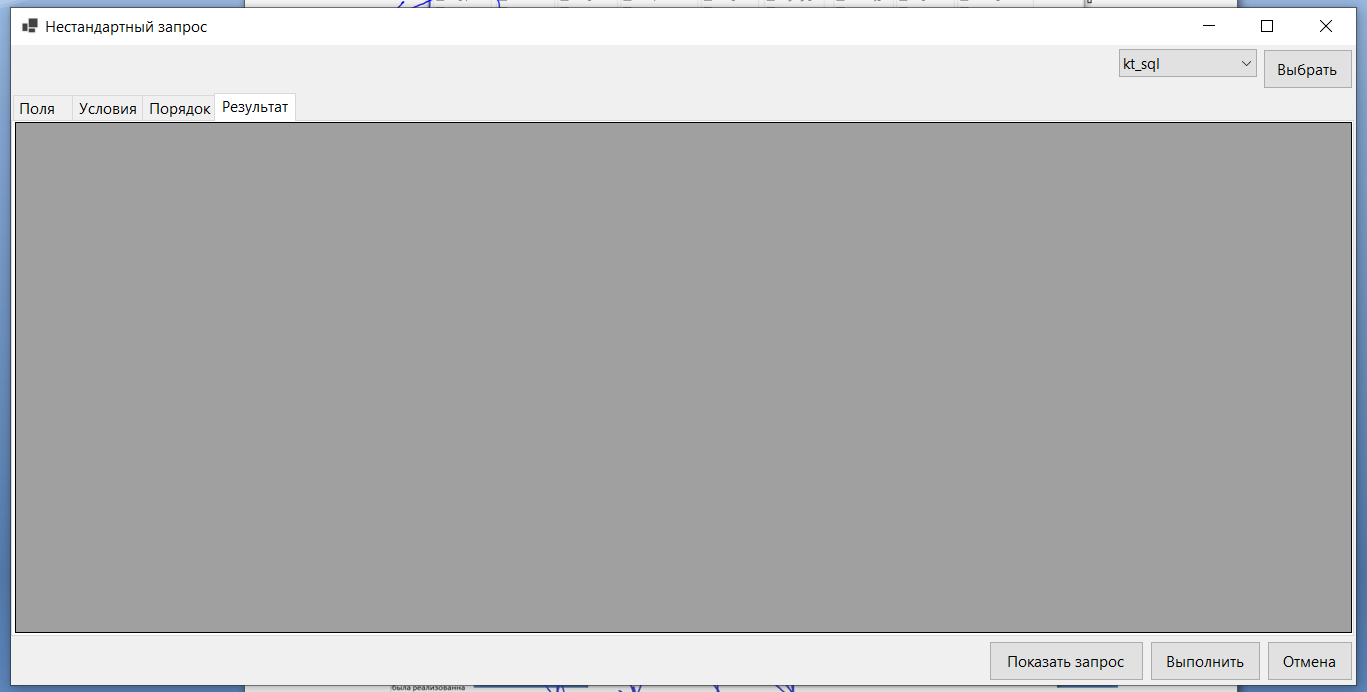
2.code = 3.code\_2 AND

3.code = 4.code\_3

# 3 Описание функционала разработанной программы

3.1. Внешний вид и функционал приложения

** **

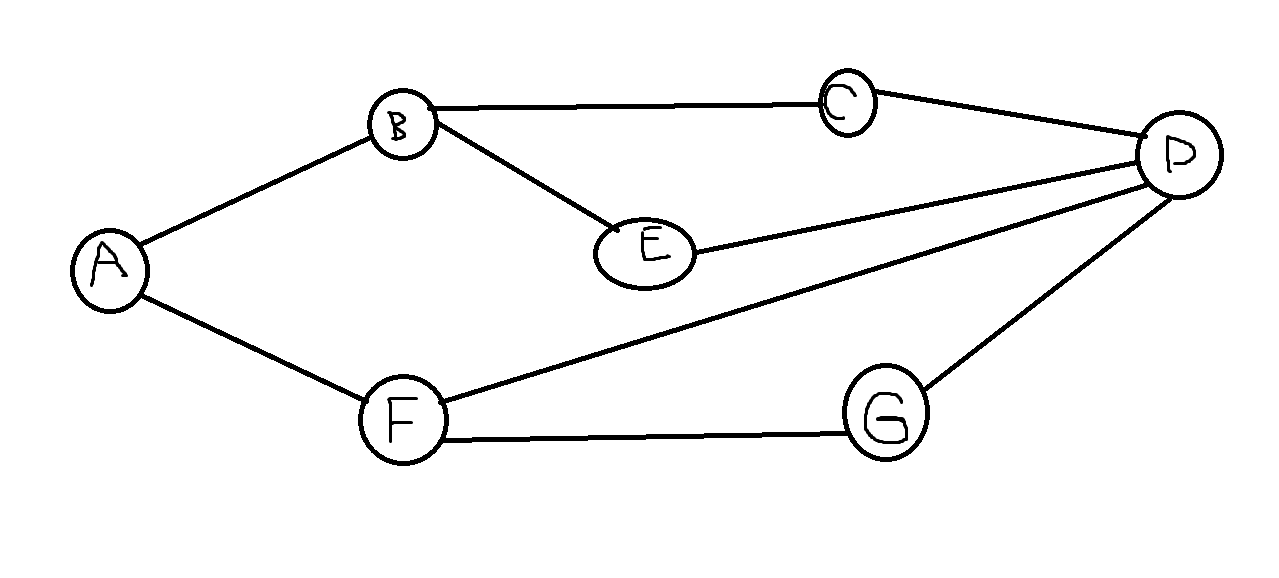
****

В данной вкладке выводится только результат запроса

## 3.2 Общий алгоритм решения задачи

Уделим внимание реализации связей между таблицами, так как это единственно не тривиальная задача в данном приложении.

Представим нашу БД как граф, где вершины это таблицы, а связи идут через ключи

Пусть нам необходимо найти путь от вершины A до Б.

Для этого применим алгоритм DFS - один из методов обхода графа. Стратегия поиска в глубину, как и следует из названия, состоит в том, чтобы идти «вглубь» графа, насколько это возможно.

Алгоритм:

1. Пометить текущую вершину использованную
2. Проверить есть ли внешний ключ между текущими двумя вершинами
3. Для всех соседних таблиц

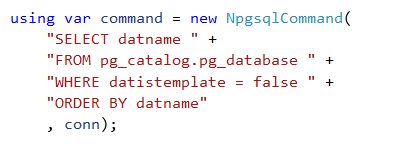
3.1. Если соседняя таблица не использована:

3.2. Получить путь от соседней вершины до конечной вершины

3.3. Если путь не пустой то вернуть его, добавив текущий внешний ключ между текущей таблицей и соседом

# 4 Реализация программы

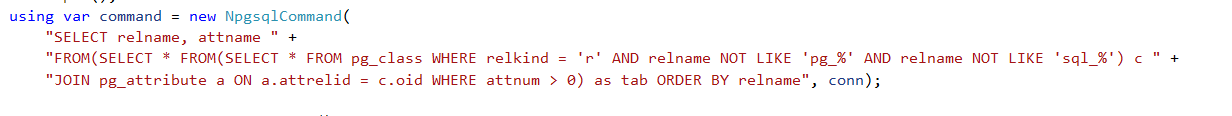
Для начала нам необходимо получить все БД, для этого отправим запрос в СУБД, используя метаданные СУБД.



Затем из них выбираем необходимую для нас БД

Возникает проблема формирования атрибутов выводимых в нашей БД

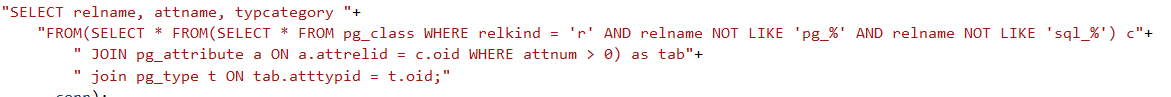
Для этого используя метаданные вывожу все таблицы и их атрибуты.

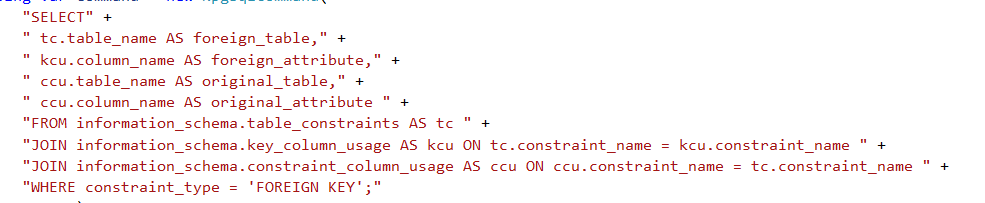


И используя словари, постепенно заполняю нашу форму.

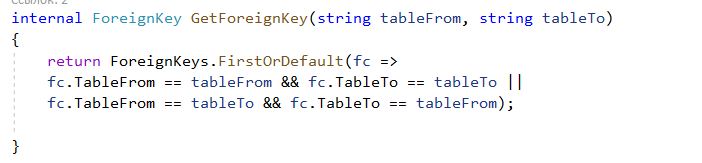
Затем нам необходимо запомнить в нашем приложении все атрибуты с их типами для каждой таблицы, для этого был использован класс attribute для хранения отдельного атрибуты и коллекция attributes в объекте dbModel.

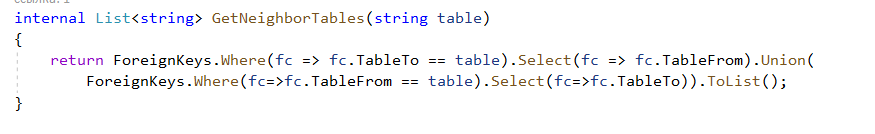
Что бы заполнить нашу коллекцию необходимо выполнить запрос в БД(используя метаданные)

 Для формирования запроса из нескольких таблиц нам необходимо знать связи между ними через ключи для этого так же обратимся к метаданным БД.



Определим часть ключевых методов для работы с ключами





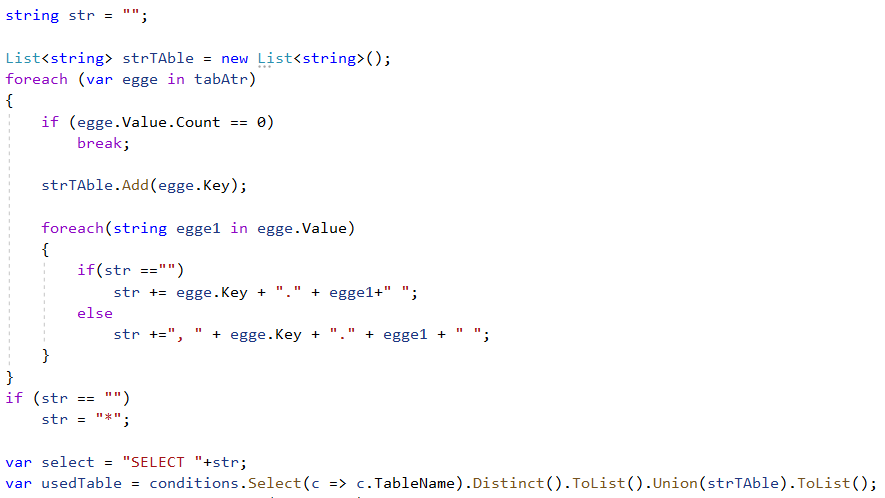
Формирование запроса:

Формирование строки SELECT

Так как некоторые атрибуты могут повторять имена у разных таблиц поэтому был введено условие имя таблицы . имя столбца

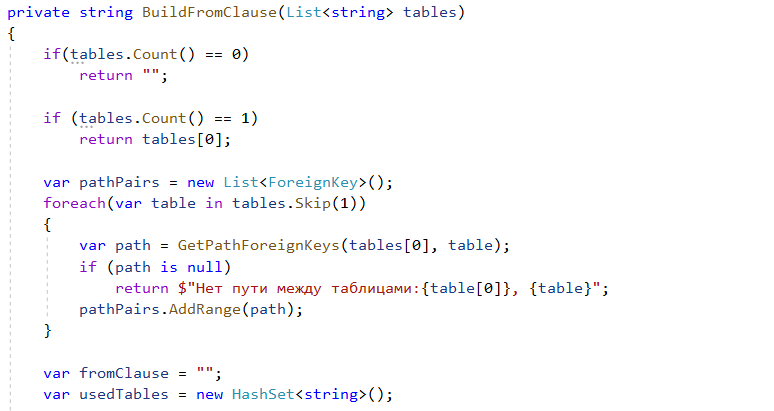
Так же нам необходимо сформировать список используемых таблиц:

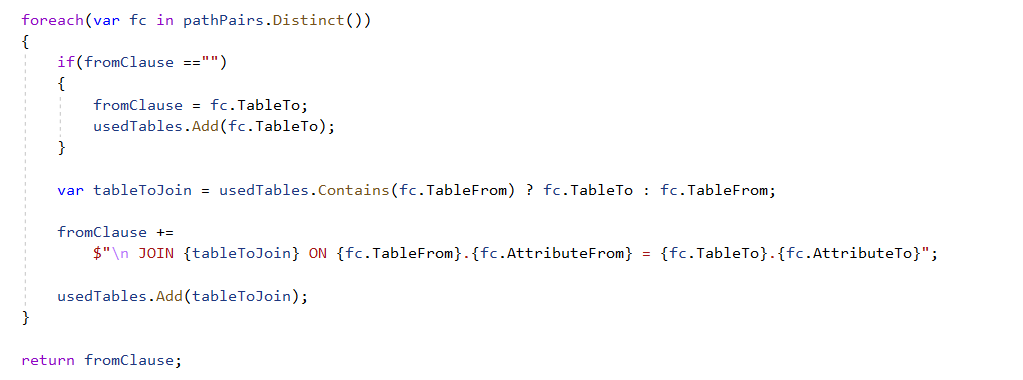
Таблицы фигурирующие в условии, таблицы фигурирующие в параметрах запроса.



Затем нам необходимо сформировать блок FROM







Если пользователь не выбрал ни одной таблицы, то вставляем пустую строку во FROM.

Если таблица одна, то вставляем её имя.

Иначе нам нужно найти связи между таблицами и сформировать запрос через JOIN.

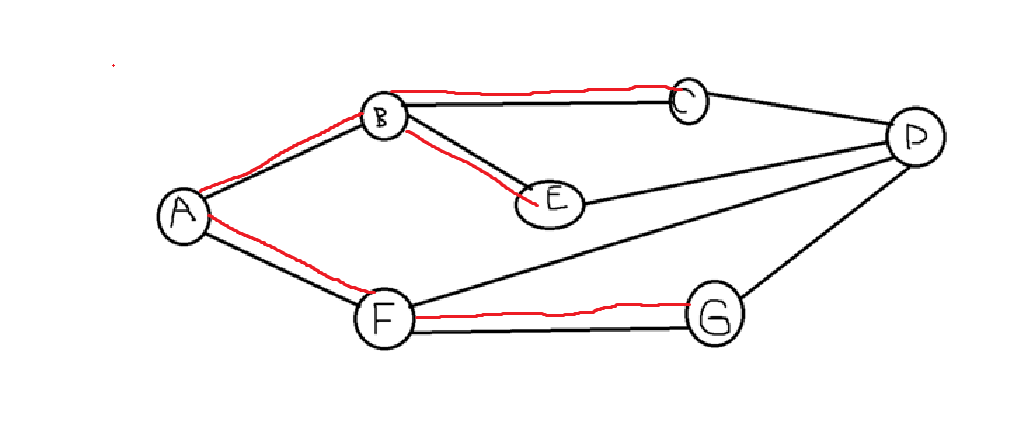
Для этого нам нужно сформировать список для каждой вершины от искомой таблицы (которая должна лежать во FROM до каждой следующей)

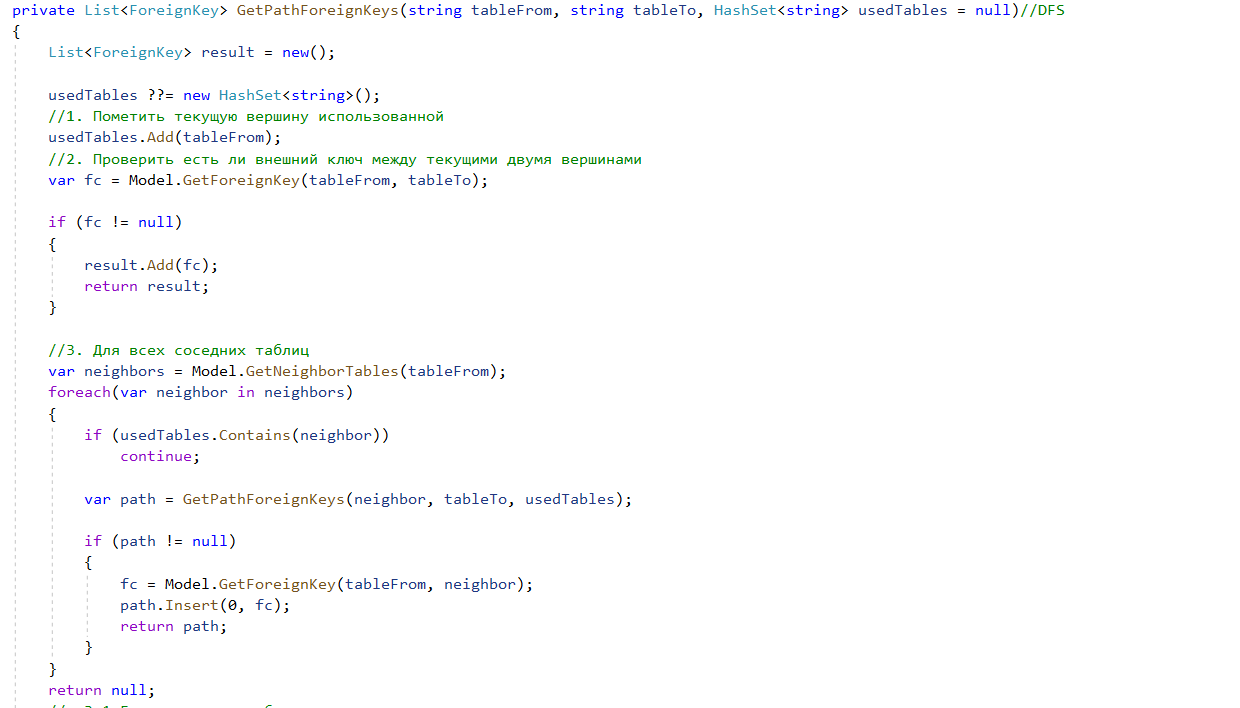
То есть пусть нам нужно получить пути от вершины до вершин C, E, G

Будем хранить пути в списке {(A->B, B->C); (A->F, F->G); (A->B, B->E)}

Что бы исключить повторного включения одних и тех же путей используем параметр Distinct для коллекции путей.

И формируем связи следя за порядком и именно какая таблица, с какой связывается что бы избежать случаев где имеем A->B C->B.

Для этого используем метод поиска в глубину



Используем список помеченных вершин.

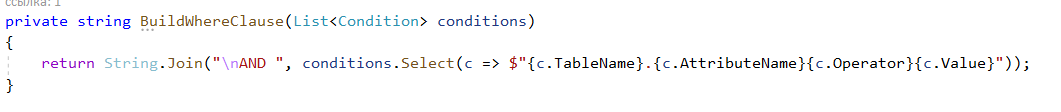
Текущую вершину, с которой вошли, пометим как использованную

Проверим, есть ли ключ текущей таблицей и ту, которую ищем, если есть, то значит, мы его нашли и возвращаем путь рекурсивно иначе

Проверяем всех соседей (рекурсивно вызывая их как вершину из которой ищем путь), если путь нашелся то добавляем их ключ связь в результат, в нулевую позицию списка, так как результат который возвращает DFS отображен зеркально.



Затем выполняем формирование строки условия



Используя оператор JOIN для соединения элементов коллекции через связку AND.

А дальше уже исполняем полученный запрос и отправляем его результат пользователю.

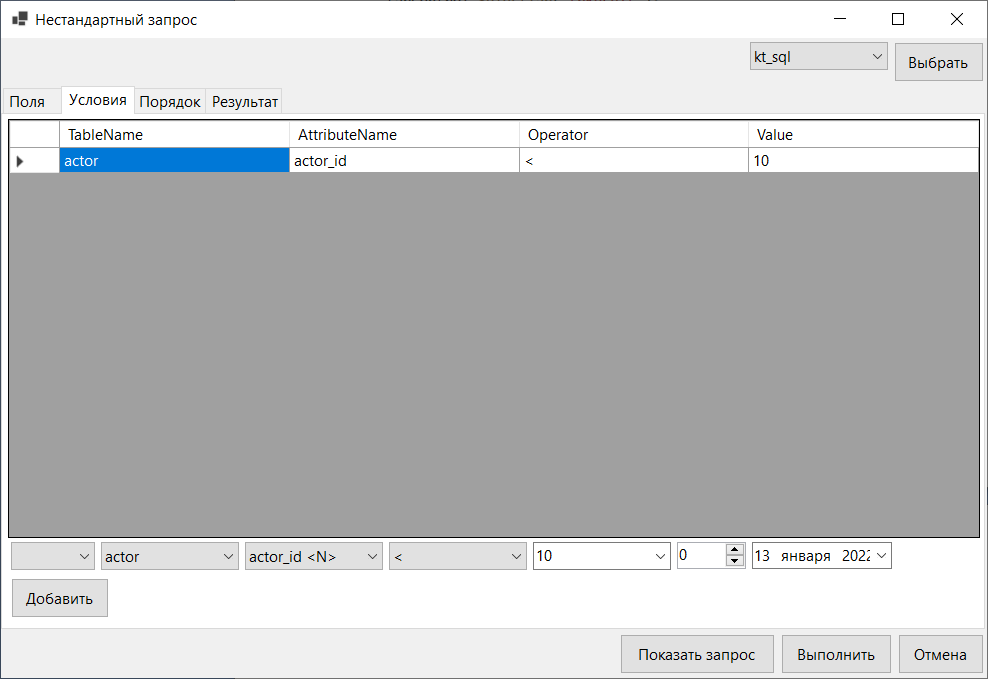


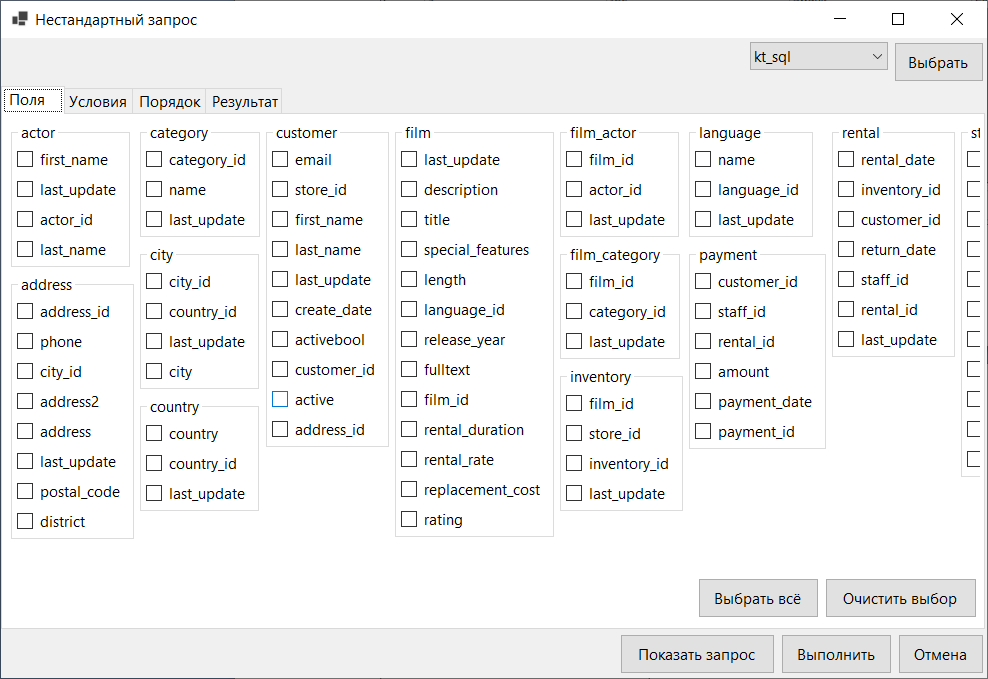
# 5 Тестирование программы

Тестирование программы можно разделить на 3 случая:

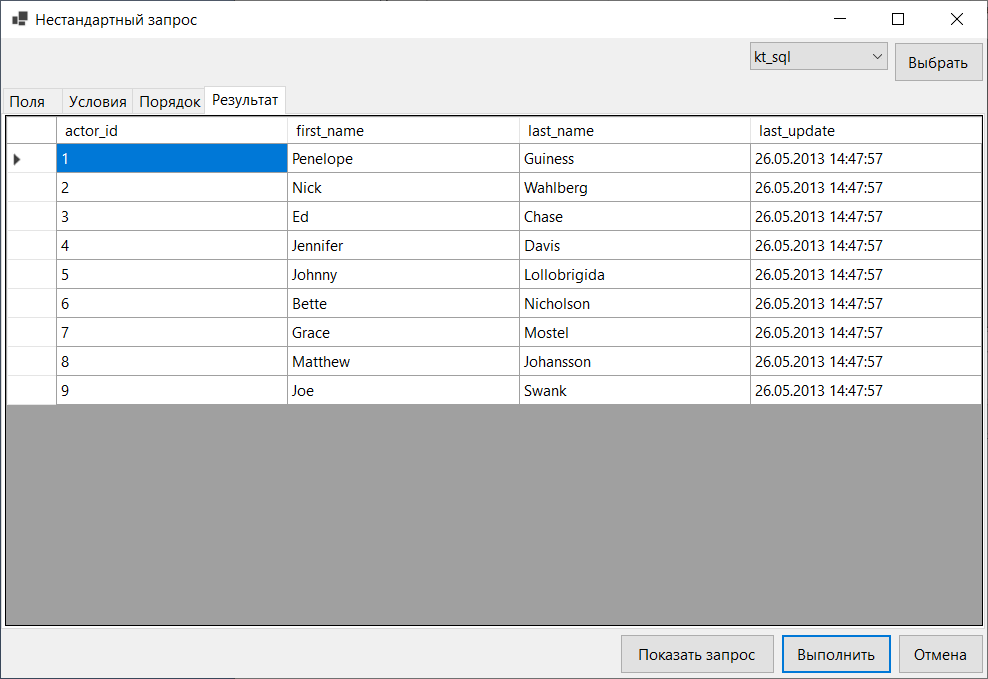
1. Работа с одной таблицей с условиями
2. Работа с несколькими таблицами связь явная
3. Работа с несколькими таблицами связь через атрибуты вывода
4. Работа с одной таблицей без условий

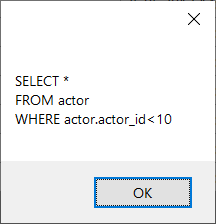
Входные данные





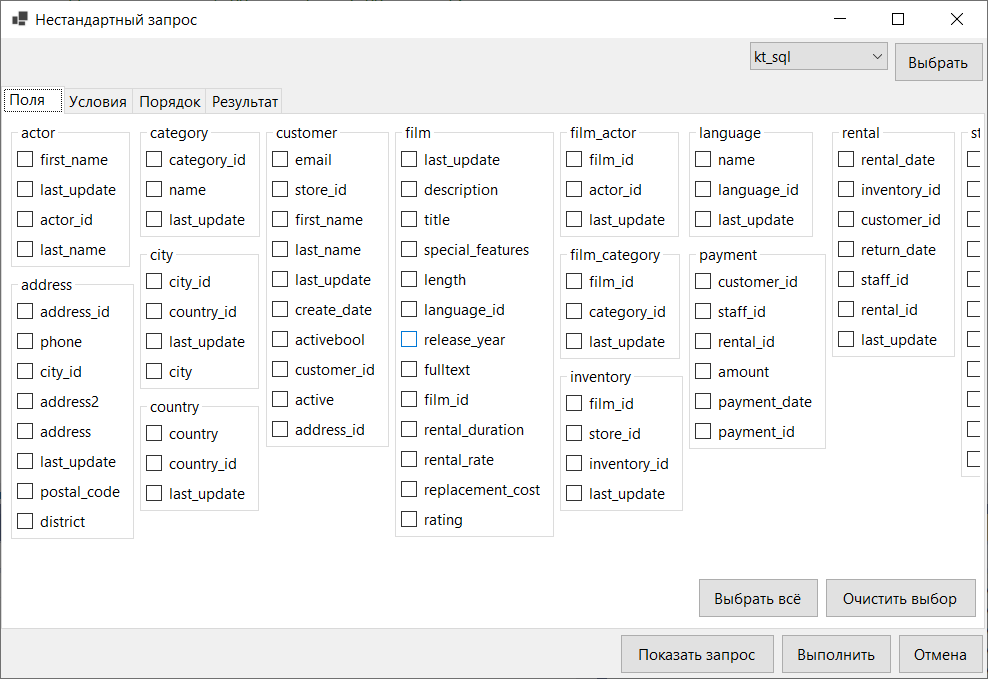
Выходные данные

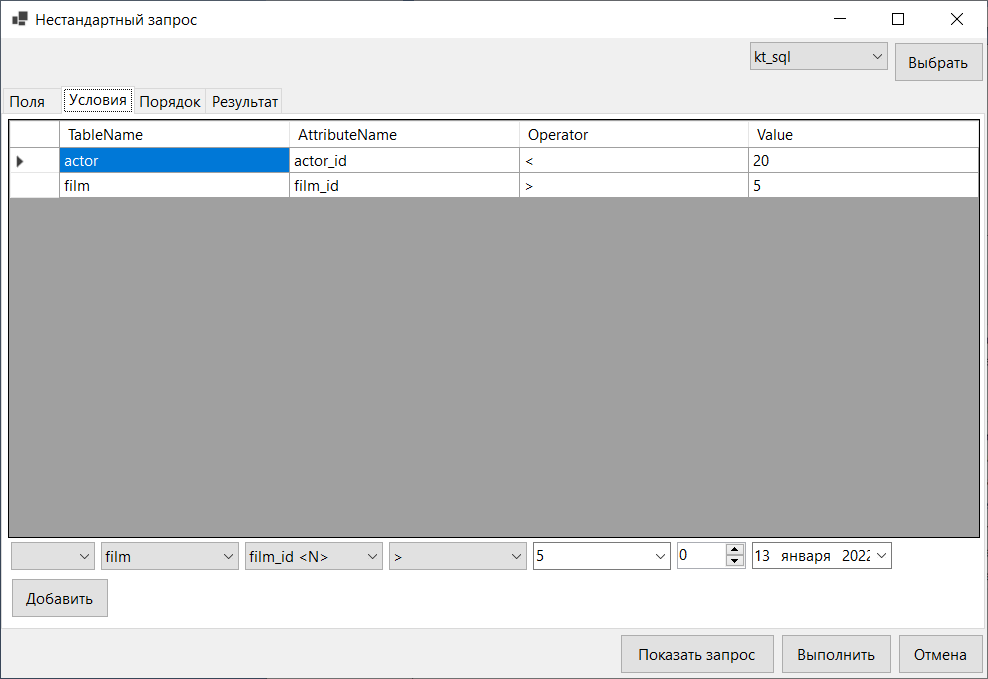




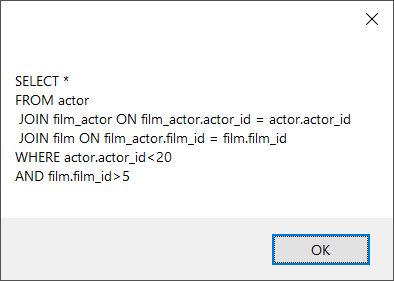
1. Работа с несколькими таблицами связь явная

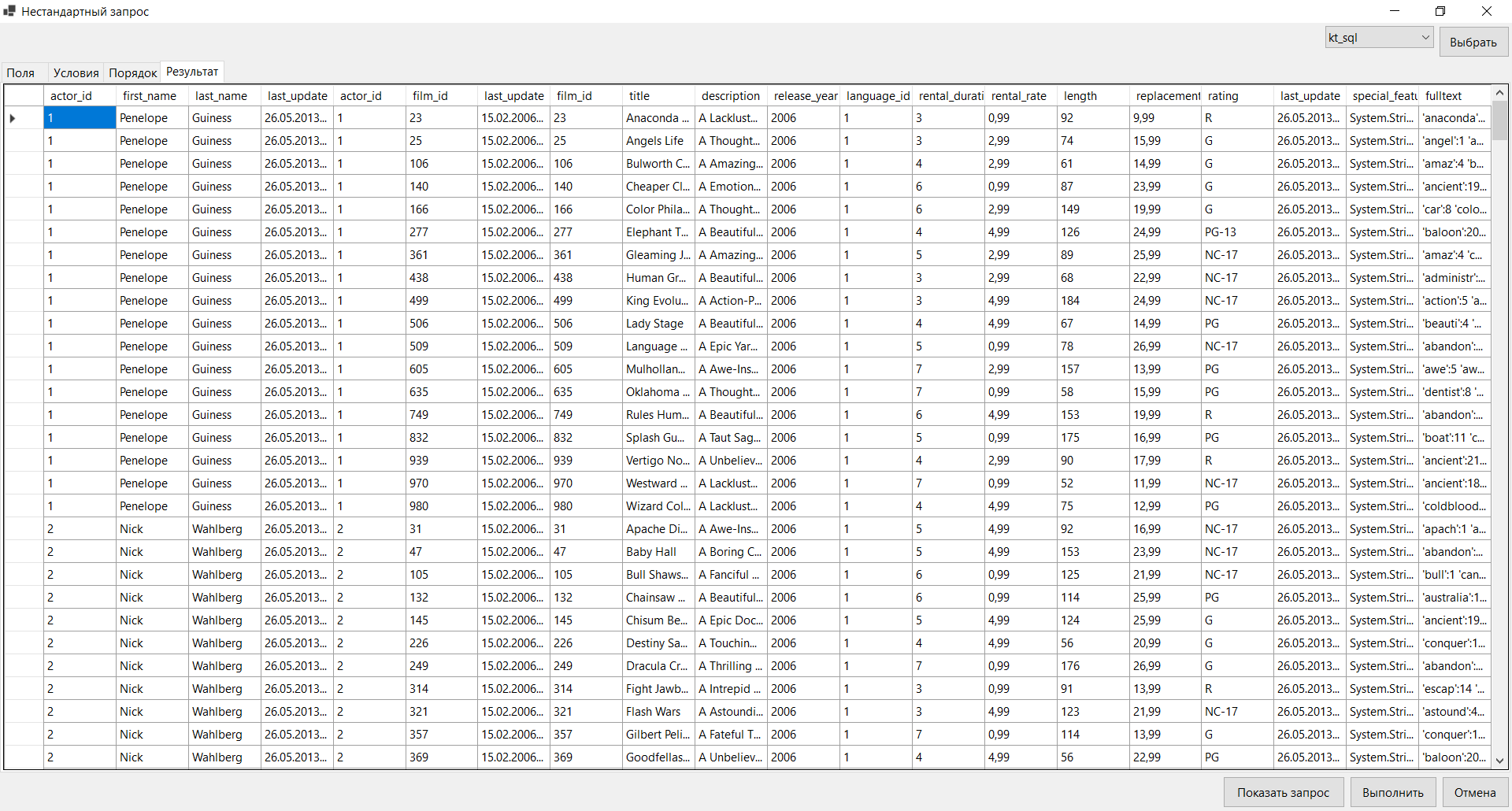
Входные данные



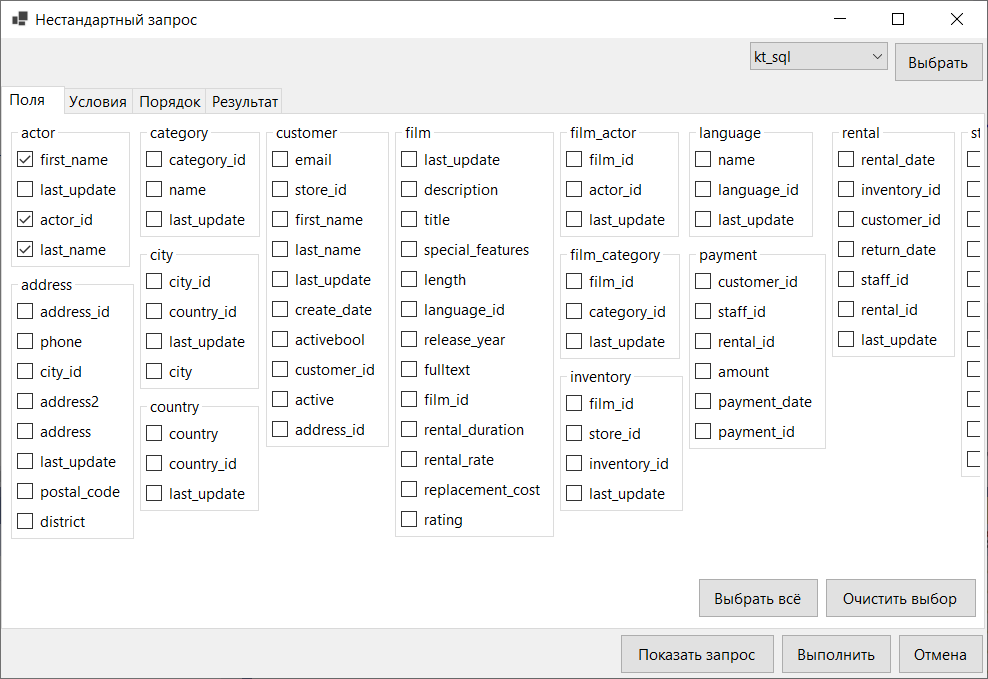


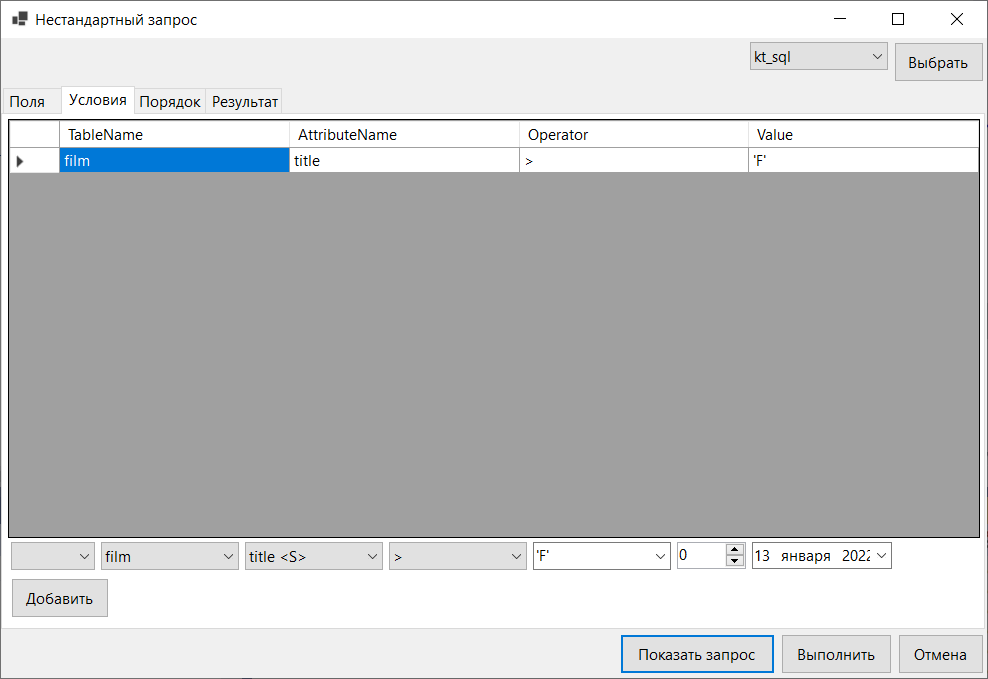
Выходные данные



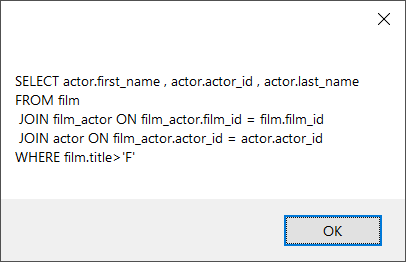


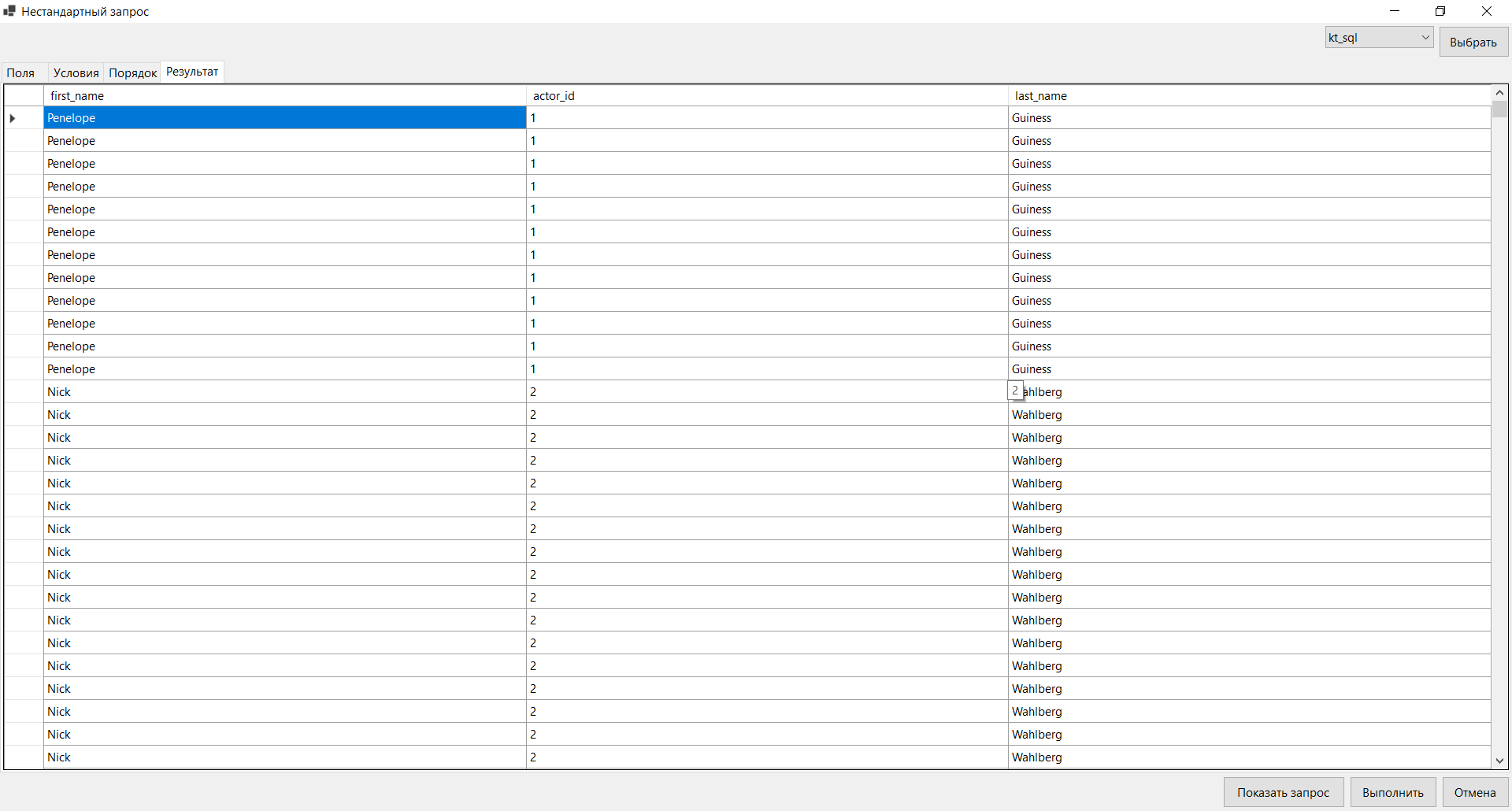
1. Работа с несколькими таблицами связь через атрибуты вывода





Выходные данные





Вывод: Программа прошла все тесты, работает корректно.