Министерство образования Республики Беларусь

Учреждение образования

Белорусский государственный университет информатики и радиоэлектроники

Факультет компьютерного проектирования

Кафедра инженерной психологии и эргономики

Дисциплина: Базы данных

Лабораторная работа № 3

«Оператор объединения. Соединение таблиц. Вложенные запросы»

Выполнил:

ст.гр. 113802

Разумов Д.А.

Проверила:

Василькова А.Н.

Минск 2022

Когда мы хотели узнать, какой маршрут соответствует машине с ID=4, то делали соответствующий запрос:

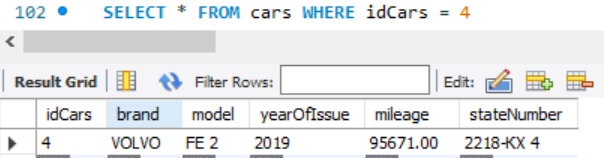


Рисунок 1 – Запрос всех маршрутов с соответствующим ID машины

В SQL предусмотрена возможность соединять такие запросы в один путем превращения одного из них в подзапрос (вложенный запрос). Например, чтобы узнать, какой марки машина проехала за свой маршрут более 100000 километров, мы сделаем следующий запрос:

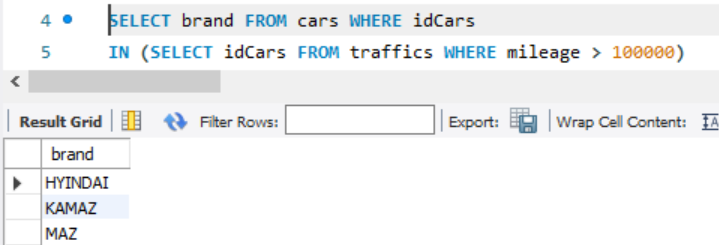


Рисунок 2 – Запрос к таблицам машины и маршруты

Предположим, надо узнать, сколько проехал соответствующий водитель на соответствующей машине. Для этого проще всего обратиться к таблице маршруты (routs):

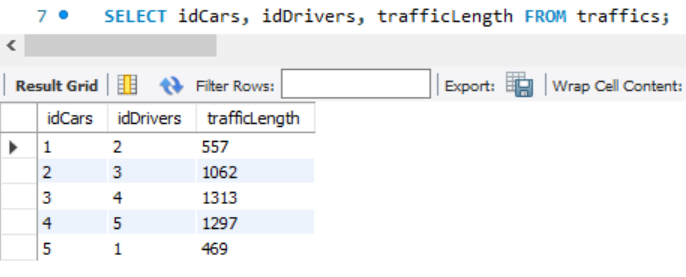


Рисунок 3 – Запрос к таблице маршруты

Но, если нам необходимо, чтобы в ответе на запрос были не идентификаторы категории, а их названия? Вложенные запросы нам не помогут, т.к. в конечном итоге они выдают данные из одной таблицы. А нам надо получить данные из двух таблиц (диски и категории) и объединить их в одну. Запросы, которые позволяют это сделать, в SQL называются *Объединениями*.

Синтаксис самого простого объединения, следующий:

SELECT имена\_столбцов\_таблицы\_1, имена\_столбцов\_таблицы\_2 FROM имя\_таблицы\_1, имя\_таблицы\_2;

Создадим простое соединение:

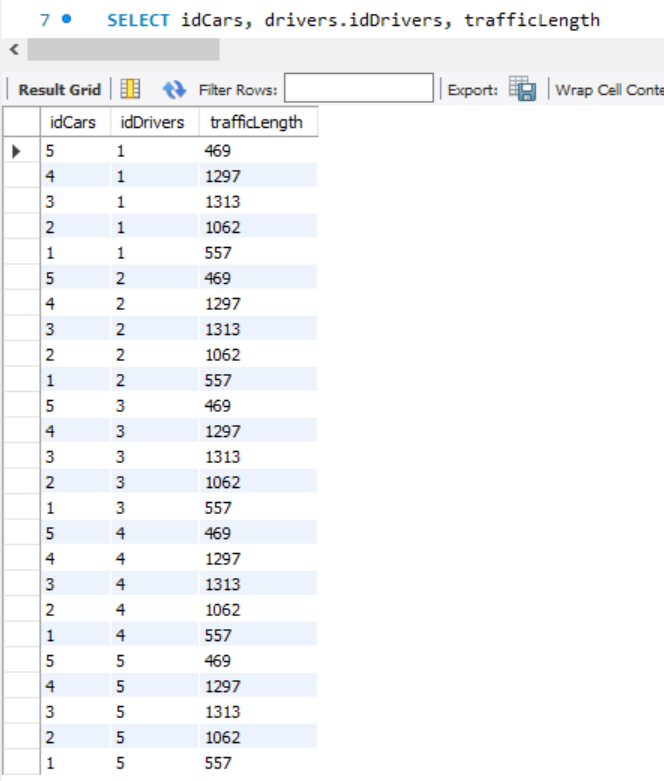


Рисунок 4 – Запрос к таблицам маршруты, водители

Получилось не совсем то, что ожидали. Такое соединение называется декартовым произведением, когда каждой строке первой таблицы ставится в соответствие каждая строка второй таблицы. Возможно, бывают случаи, когда такое объединение полезно, но это явно не наш случай.

Чтобы результирующая таблица выглядела так, как мы хотели, необходимо указать условие объединения. Мы связываем наши таблицы по идентификатору категории, это и будет нашим условием. Т.е. мы укажем в запросе, что необходимо выводить только те строки, в которых значения поля idDrivers таблицы traffics совпадает со значениями поля idDrivers таблицы drivers:

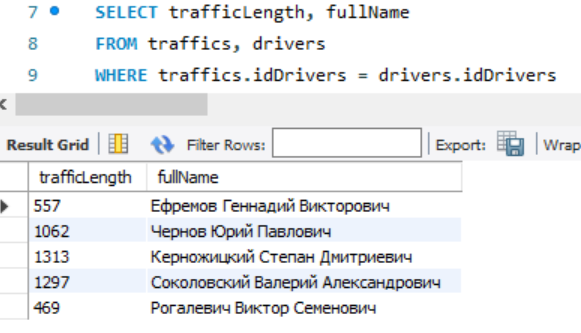


Рисунок 5 – Запрос к таблицам маршруты, машины и водители

Т.е. мы в запросе сделали следующее условие: если в обеих таблицах есть одинаковые идентификаторы, то строки с этим идентификатором необходимо объединить в одну результирующую строку.

**Важно знать:**

1. Если в одной из соединяемых таблиц есть строка с идентификатором, которого нет в другой таблице, то в результирующей таблице строки с таким идентификатором не будет. В нашем примере есть пользователь Oleg (id=5), но он не создавал тем, поэтому в результате запроса его нет.
2. При указании условия название столбца пишется после названия таблицы, в которой этот столбец находится (через точку). Это сделано во избежание путаницы, ведь столбцы в разных таблицах могут иметь одинаковые названия, и MySQL может не понять, о каких конкретно столбцах идет речь.

Вообще, корректный синтаксис объединения с условием выглядит так:

SELECT имя\_таблицы\_1.имя\_столбца1\_таблицы\_1, имя\_таблицы\_1.имя\_столбца2\_таблицы\_1, имя\_таблицы\_2.имя\_столбца1\_таблицы\_2, имя\_таблицы\_2.имя\_столбца2\_таблицы\_2 FROM имя\_таблицы\_1, имя\_таблицы\_2

WHERE имя\_таблицы\_1.имя\_столбца\_по\_которому\_объединяем= имя\_таблицы\_2.имя\_столбца\_по\_которому\_объединяем;

Если имя столбца уникально, то название таблицы можно опустить (как мы делали в примере), но делать это не рекомендуется.

Соединения таблиц дают возможность выбирать информацию из любого количества таблиц.

Но чтобы вместо идентификаторов отображались имена авторов и названия, нам придется сделать объединение трех таблиц:

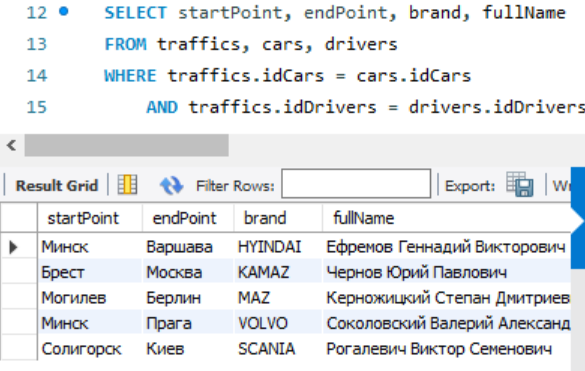


Рисунок 6 – Запрос к таблицам маршруты, машины, водители

Соединения, которые мы рассматривали, называются *Внутренними соединениями*.

Но бывают ситуации, когда необходимо, чтобы в результат были включены строки, не имеющие связанных.

Поэтому, если нам потребуется составить несколько иной запрос.

Например, надо вывести всех машины и длины маршрутов, которые они проехали, если таковые имеются, то нам придется воспользоваться *Внешним объединением*, позволяющим выводить все строки одной таблицы и имеющиеся связанные с ними строки из другой таблицы.

Если мы воспользуемся внутренним объединением, то получим в итоге следующее:

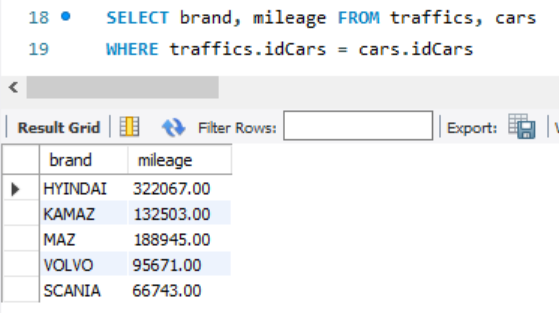


Рисунок 7 – Запрос к таблицам маршруты. машины

То есть в результирующей таблице есть только те пользователи, которые создавали темы. А нам надо, чтобы выводились все имена. Для этого мы немного изменим запрос.

И получим желаемый результат – все машины и длины маршрутов.

Если пользователь не создавал тему, но в соответствующем столбце стоит значение NULL.

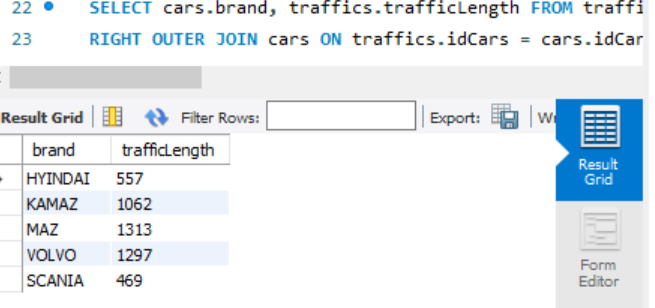


Рисунок 8 – Запрос к таблицам машины, маршруты

Итак, мы добавили в наш запрос ключевое слово - *RIGHT OUTER JOIN*, указав тем самым, что из таблицы справа надо взять все строки, и поменяли ключевое слово *WHERE* на *ON*. Кроме ключевого слова *RIGHT OUTER JOIN* может быть использовано ключевое слово *LEFT OUTER JOIN*. Тогда будут выбираться все строки из правой таблицы и имеющиеся связанные с ними из левой таблицы.

И наконец, возможно полное внешнее объединение, которое извлечет все строки из обеих таблиц и свяжет между собой те, которые могут быть связаны. Ключевое слово для полного внешнего объединения - *FULL OUTER JOIN*.

Например, поменяем в нашем запросе левостороннее объединение на правостороннее:

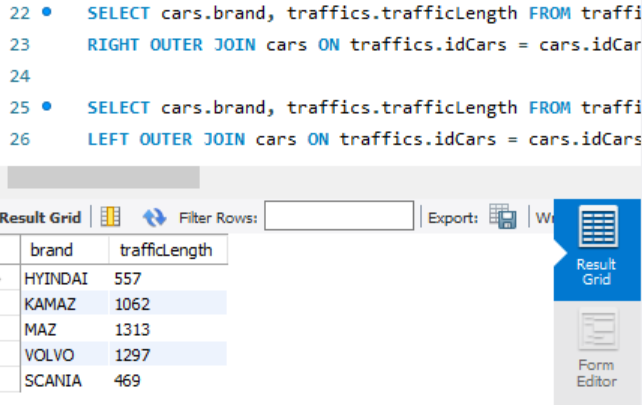


Рисунок 9 – Запрос к таблицам маршруты, машины

Как видите, теперь у нас есть все длины маршрутов (все строки из правой таблицы), а вот машины только те, которые проезжали маршрут (т.е. из левой таблицы выбираются только те строки, которые связаны с правой таблицей).

**SQL EXISTS**

Оператор EXISTS используется для проверки наличия любой записи в подзапросе.

Оператор EXISTS возвращает true, если подзапрос возвращает одну или несколько записей.

Примеры SQL EXISTS:

Следующая инструкция SQL возвращает TRUE и перечисляет водителей, которые проехали на последний рейс больше 1000км:

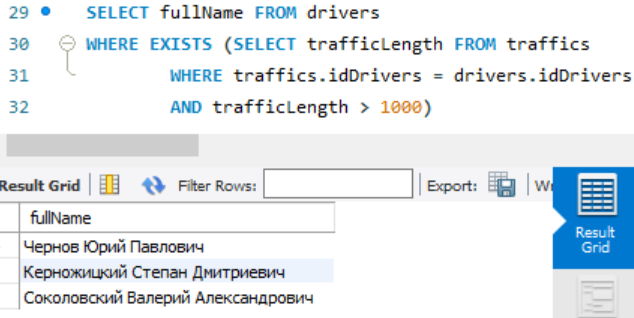


Рисунок 10 – Запрос к таблице водители

**SQL Операторы ANY и ALL**

Операторы ANY и ALL используются с предложением WHERE или HAVING.

Оператор ANY возвращает true, если какое-либо из значений подзапроса удовлетворяет условию.

Оператор ALL возвращает true, если все значения подзапроса удовлетворяют условию.

Примеры SQL ANY

Следующий оператор SQL возвращает TRUE и перечисляет километраж крайней поездки, если он находит какие-либо записи в таблице traffics, что mileage >100000:

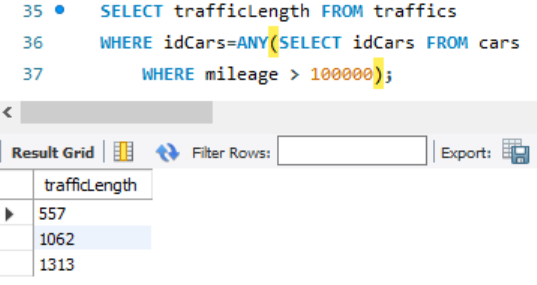


Рисунок 11 – Запрос к таблице маршруты

Пример SQL ALL

Следующая инструкция SQL возвращает TRUE и перечисляет названия продуктов, если все записи в таблице cars имеют значение mileage>10.

Таким образом, этот пример возвращает FALSE, поскольку не все записи в таблице cars имеют значение mileage>100000:

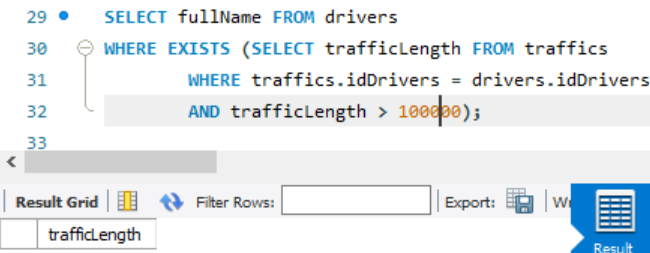


Рисунок 12 – Запрос к таблице маршруты