

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»

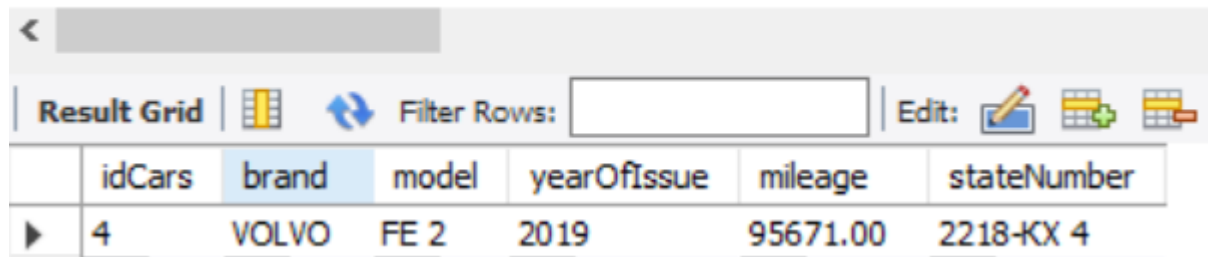
Выполнил:  
ст.гр. 113802  
Разумов Д.А.

Проверила:  
Василькова А.Н.

Минск 2022

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

```
102 • SELECT * FROM cars WHERE idCars = 4
```




The screenshot shows a database query interface. At the top, the query is: `SELECT * FROM cars WHERE idCars = 4`. Below the query, there is a toolbar with options like 'Result Grid', 'Filter Rows', and 'Edit'. The main area displays a table with the following data:

	idCars	brand	model	yearOfIssue	mileage	stateNumber
▶	4	VOLVO	FE 2	2019	95671.00	2218-KX 4

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:

```
4 • SELECT brand FROM cars WHERE idCars  
5 IN (SELECT idCars FROM traffics WHERE mileage > 100000)
```



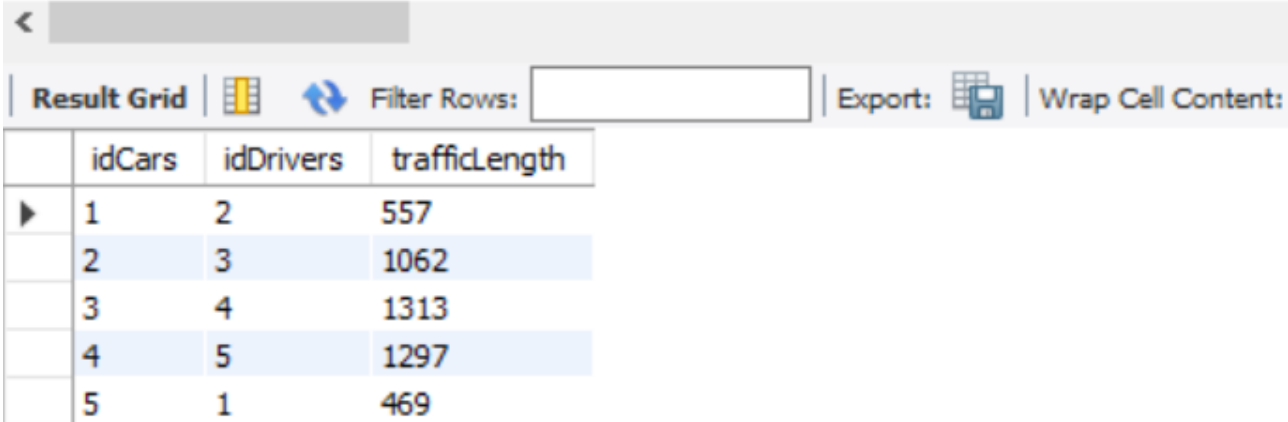
The screenshot shows a database query interface. At the top, the query is: `SELECT brand FROM cars WHERE idCars IN (SELECT idCars FROM traffics WHERE mileage > 100000)`. Below the query, there is a toolbar with options like 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content'. The main area displays a table with the following data:

	brand
▶	HYUNDAI
	KAMAZ
	MAZ

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 (routes):

7 • `SELECT idCars, idDrivers, trafficLength FROM traffics;`



The screenshot shows a database query result grid. At the top, the query `SELECT idCars, idDrivers, trafficLength FROM traffics;` is entered. Below the query bar, there is a toolbar with options: 'Result Grid' (selected), a grid icon, 'Filter Rows:' with an input field, 'Export:' with a download icon, and 'Wrap Cell Content:'. The result grid itself has three columns: 'idCars', 'idDrivers', and 'trafficLength'. It contains five rows of data.

	idCars	idDrivers	trafficLength
▶	1	2	557
	2	3	1062
	3	4	1313
	4	5	1297
	5	1	469

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:

7 • `SELECT idCars, drivers.idDrivers, trafficLength`

<

Result Grid   Filter Rows:  Export:  Wrap Cell Conte

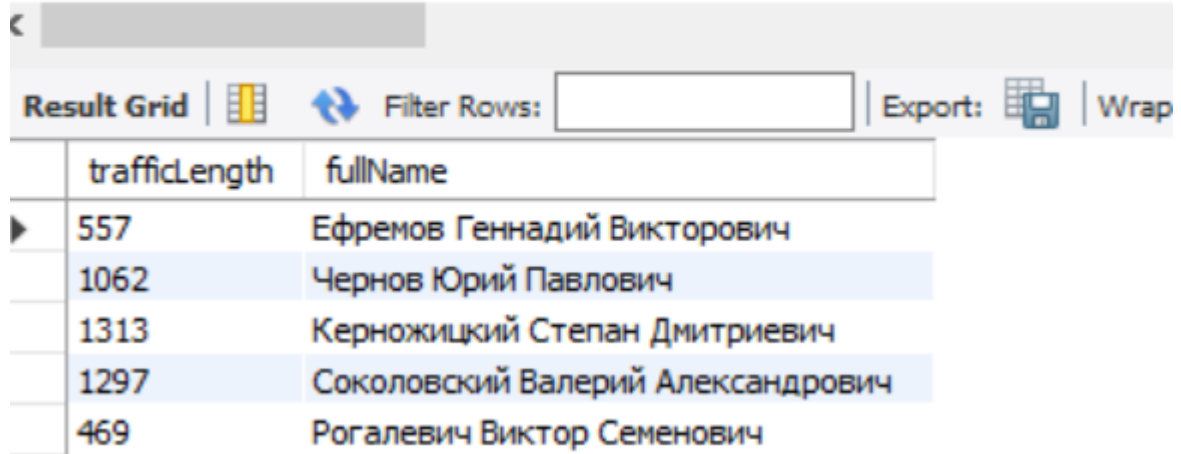
	idCars	idDrivers	trafficLength
▶	5	1	469
	4	1	1297
	3	1	1313
	2	1	1062
	1	1	557
	5	2	469
	4	2	1297
	3	2	1313
	2	2	1062
	1	2	557
	5	3	469
	4	3	1297
	3	3	1313
	2	3	1062
	1	3	557
	5	4	469
	4	4	1297
	3	4	1313
	2	4	1062
	1	4	557
	5	5	469
	4	5	1297
	3	5	1313
	2	5	1062
	1	5	557

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`:

```
7 • SELECT trafficLength, fullName
8     FROM traffics, drivers
9     WHERE traffics.idDrivers = drivers.idDrivers
```



	trafficLength	fullName
▶	557	Ефремов Геннадий Викторович
	1062	Чернов Юрий Павлович
	1313	Керножицкий Степан Дмитриевич
	1297	Соколовский Валерий Александрович
	469	Рогалевич Виктор Семенович

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.

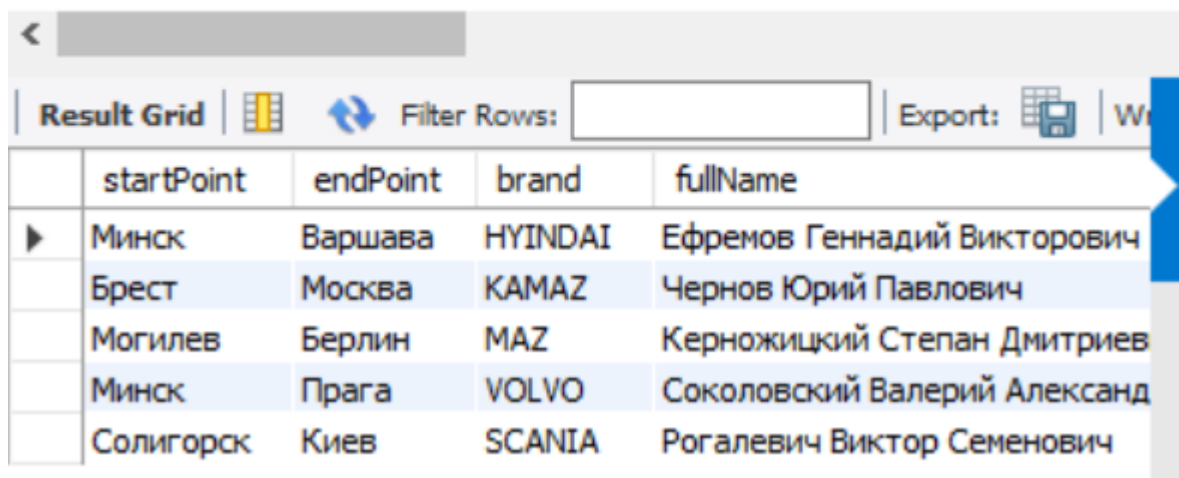
Соединения таблиц дают возможность выбирать информацию из любого количества таблиц.

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

```

12 • SELECT startPoint, endPoint, brand, fullName
13 FROM traffics, cars, drivers
14 WHERE traffics.idCars = cars.idCars
15 AND traffics.idDrivers = drivers.idDrivers

```



	startPoint	endPoint	brand	fullName
▶	Минск	Варшава	HYUNDAI	Ефремов Геннадий Викторович
	Брест	Москва	KAMAZ	Чернов Юрий Павлович
	Могилев	Берлин	MAZ	Керножицкий Степан Дмитриевич
	Минск	Прага	VOLVO	Соколовский Валерий Александрович
	Солигорск	Киев	SCANIA	Рогалевич Виктор Семенович

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.

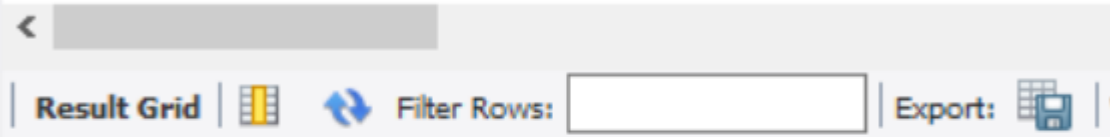
Поэтому, если нам потребуется составить несколько иной запрос.

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:

```
18 • SELECT brand, mileage FROM traffics, cars
19 WHERE traffics.idCars = cars.idCars
```



	brand	mileage
▶	HYINDAI	322067.00
	KAMAZ	132503.00
	MAZ	188945.00
	VOLVO	95671.00
	SCANIA	66743.00


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.

```
22 • SELECT cars.brand, traffics.trafficLength FROM traffi
23 RIGHT OUTER JOIN cars ON traffics.idCars = cars.idCar
```



	brand	trafficLength
▶	HYINDAI	557
	KAMAZ	1062
	MAZ	1313
	VOLVO	1297
	SCANIA	469

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:

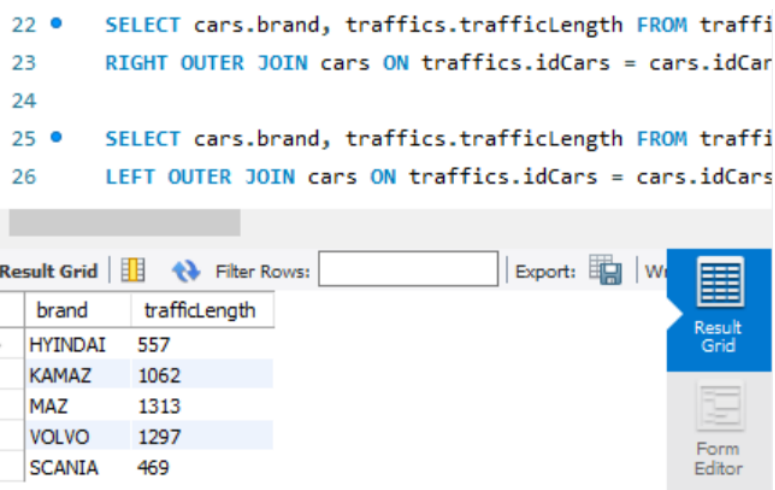


Рисунок 9 – Запрос к таблицам маршруты, машины

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.

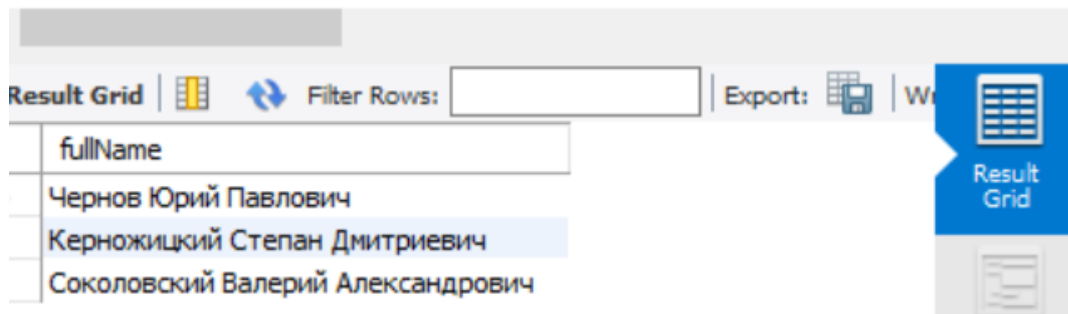
Оператор EXISTS возвращает true, если подзапрос возвращает одну или несколько записей.



Примеры SQL EXISTS:

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

```
29 • SELECT fullName FROM drivers
30 WHERE EXISTS (SELECT trafficLength FROM traffics
31               WHERE traffics.idDrivers = drivers.idDrivers
32               AND trafficLength > 1000)
```



fullName
Чернов Юрий Павлович
Керножицкий Степан Дмитриевич
Соколовский Валерий Александрович

Рисунок 10 – Запрос к таблице водители

## SQL Операторы ANY и ALL

Операторы ANY и ALL используются с предложением WHERE или HAVING.

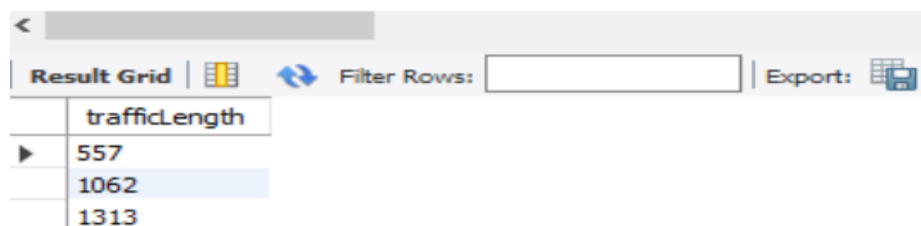
Оператор ANY возвращает true, если какое-либо из значений подзапроса удовлетворяет условию.

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:

```
35 • SELECT trafficLength FROM traffics
36 WHERE idCars=ANY(SELECT idCars FROM cars
37                  WHERE mileage > 100000);
```



trafficLength
557
1062
1313

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:

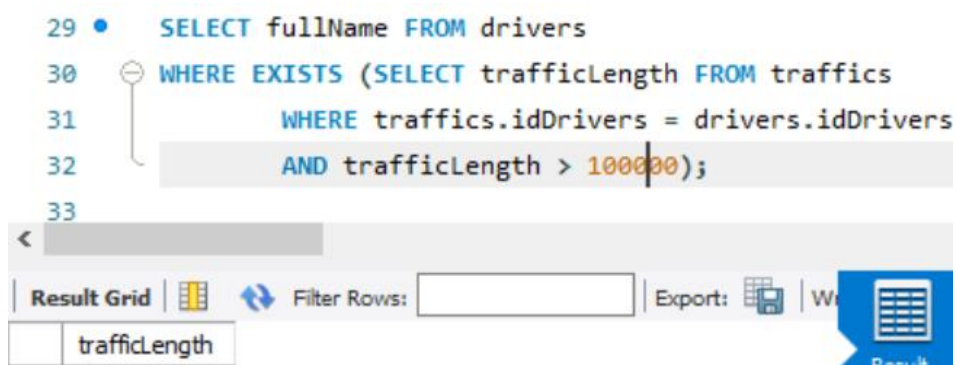


Рисунок 12 – Запрос к таблице маршруты