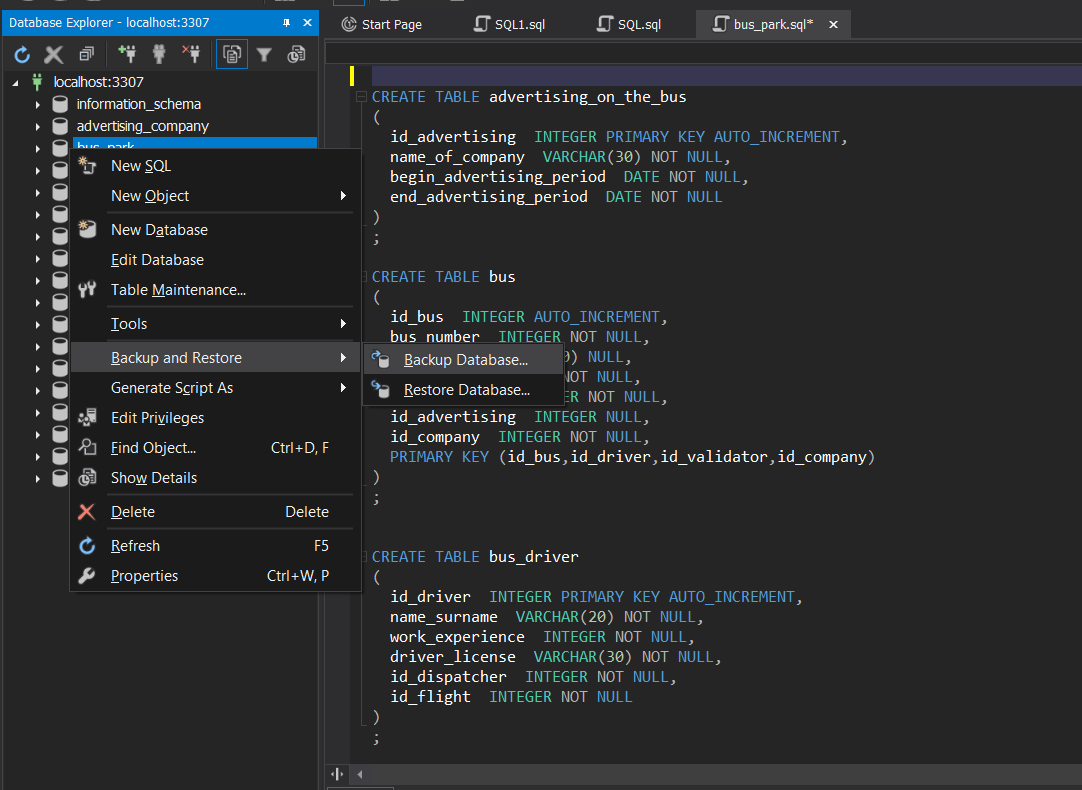
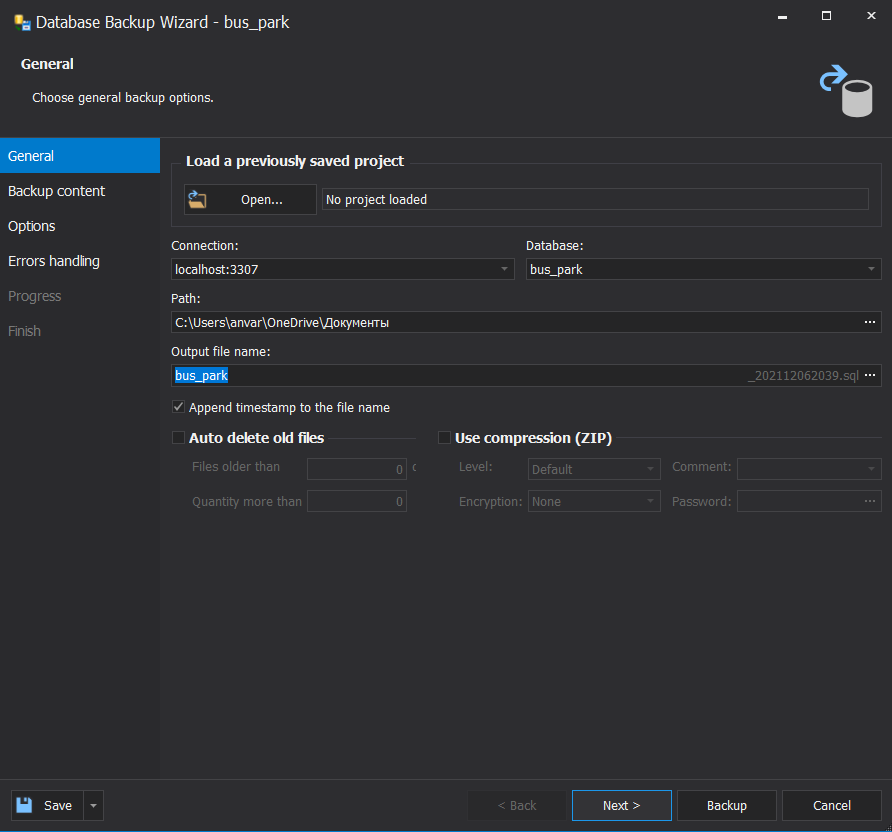
**Практическая №3**

**1.2. Перенос базы данных на другой сервер**

Выберем в dbForce резервную копию базы данных





Покажем скрипт, который получился:

*--*

*-- Script was generated by Devart dbForge Studio 2020 for MySQL, Version 9.0.782.0*

*-- Product home page: http://www.devart.com/dbforge/mysql/studio*

*-- Script date 06.12.2021 20:40:15*

*-- Server version: 8.0.26*

*-- Client version: 4.1*

*--*

*--*

*-- Disable foreign keys*

*--*

*/\*!40014 SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0 \*/*;

*--*

*-- Set SQL mode*

*--*

*/\*!40101 SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='NO\_AUTO\_VALUE\_ON\_ZERO' \*/*;

*--*

*-- Set character set the client will use to send SQL statements to the server*

*--*

SET NAMES 'utf8';

DROP DATABASE IF EXISTS bus\_park;

CREATE DATABASE bus\_park

CHARACTER SET utf8mb4

COLLATE utf8mb4\_0900\_ai\_ci;

*--*

*-- Set default database*

*--*

USE bus\_park;

*--*

*-- Create table `dispatcher`*

*--*

CREATE TABLE dispatcher (

  id\_dispatcher int NOT NULL AUTO\_INCREMENT,

  name varchar(20) NOT NULL,

  surname varchar(20) NOT NULL,

  PRIMARY KEY (id\_dispatcher)

)

ENGINE = INNODB,

AUTO\_INCREMENT = 6,

AVG\_ROW\_LENGTH = 3276,

CHARACTER SET utf8mb4,

COLLATE utf8mb4\_0900\_ai\_ci;

*--*

*-- Create table `bus\_flight`*

*--*

CREATE TABLE bus\_flight (

  id\_flight int NOT NULL AUTO\_INCREMENT,

  name\_start\_flight varchar(30) NOT NULL,

  name\_end\_flight varchar(30) NOT NULL,

  number\_flight int NOT NULL,

  PRIMARY KEY (id\_flight)

)

ENGINE = INNODB,

AUTO\_INCREMENT = 7,

AVG\_ROW\_LENGTH = 2730,

CHARACTER SET utf8mb4,

COLLATE utf8mb4\_0900\_ai\_ci;

*--*

*-- Create table `bus\_driver`*

*--*

CREATE TABLE bus\_driver (

  id\_driver int NOT NULL AUTO\_INCREMENT,

  name\_surname varchar(20) NOT NULL,

  work\_experience int NOT NULL,

  kol\_flight int DEFAULT NULL,

  driver\_license varchar(30) NOT NULL,

  id\_dispatcher int NOT NULL,

  id\_flight int NOT NULL,

  PRIMARY KEY (id\_driver)

)

ENGINE = INNODB,

AUTO\_INCREMENT = 19,

AVG\_ROW\_LENGTH = 910,

CHARACTER SET utf8mb4,

COLLATE utf8mb4\_0900\_ai\_ci;

*--*

*-- Create foreign key*

*--*

ALTER TABLE bus\_driver

ADD CONSTRAINT bus\_driver\_ibfk\_1 FOREIGN KEY (id\_dispatcher)

REFERENCES dispatcher (id\_dispatcher);

*--*

*-- Create foreign key*

*--*

ALTER TABLE bus\_driver

ADD CONSTRAINT bus\_driver\_ibfk\_2 FOREIGN KEY (id\_flight)

REFERENCES bus\_flight (id\_flight);

*--*

*-- Create table `validator`*

*--*

CREATE TABLE validator (

  id\_validator int NOT NULL AUTO\_INCREMENT,

  isActive enum ('да', 'нет') NOT NULL DEFAULT (\_cp866 '¤ '),

  PRIMARY KEY (id\_validator)

)

ENGINE = INNODB,

AUTO\_INCREMENT = 28,

AVG\_ROW\_LENGTH = 606,

CHARACTER SET utf8mb4,

COLLATE utf8mb4\_0900\_ai\_ci;

*--*

*-- Create table `passenger\_insurance\_company`*

*--*

CREATE TABLE passenger\_insurance\_company (

  id\_company int NOT NULL AUTO\_INCREMENT,

  name\_insurance\_company varchar(20) NOT NULL,

  contract\_number int NOT NULL,

  date\_of\_conclusion date NOT NULL,

  PRIMARY KEY (id\_company)

)

ENGINE = INNODB,

AUTO\_INCREMENT = 6,

AVG\_ROW\_LENGTH = 3276,

CHARACTER SET utf8mb4,

COLLATE utf8mb4\_0900\_ai\_ci;

*--*

*-- Create table `advertising\_on\_the\_bus`*

*--*

CREATE TABLE advertising\_on\_the\_bus (

  id\_advertising int NOT NULL AUTO\_INCREMENT,

  name\_of\_company varchar(30) NOT NULL,

  begin\_advertising\_period date NOT NULL,

  end\_advertising\_period date NOT NULL,

  PRIMARY KEY (id\_advertising)

)

ENGINE = INNODB,

AUTO\_INCREMENT = 7,

AVG\_ROW\_LENGTH = 2730,

CHARACTER SET utf8mb4,

COLLATE utf8mb4\_0900\_ai\_ci;

*--*

*-- Create table `bus`*

*--*

CREATE TABLE bus (

  id\_bus int NOT NULL AUTO\_INCREMENT,

  bus\_number int NOT NULL,

  bus\_name varchar(30) DEFAULT NULL,

  id\_driver int NOT NULL,

  id\_validator int NOT NULL,

  id\_advertising int DEFAULT NULL,

  id\_company int NOT NULL,

  PRIMARY KEY (id\_bus, id\_driver, id\_validator, id\_company)

)

ENGINE = INNODB,

AUTO\_INCREMENT = 28,

AVG\_ROW\_LENGTH = 910,

CHARACTER SET utf8mb4,

COLLATE utf8mb4\_0900\_ai\_ci;

*--*

*-- Create foreign key*

*--*

ALTER TABLE bus

ADD CONSTRAINT bus\_ibfk\_1 FOREIGN KEY (id\_driver)

REFERENCES bus\_driver (id\_driver);

*--*

*-- Create foreign key*

*--*

ALTER TABLE bus

ADD CONSTRAINT bus\_ibfk\_2 FOREIGN KEY (id\_validator)

REFERENCES validator (id\_validator);

*--*

*-- Create foreign key*

*--*

ALTER TABLE bus

ADD CONSTRAINT bus\_ibfk\_3 FOREIGN KEY (id\_advertising)

REFERENCES advertising\_on\_the\_bus (id\_advertising);

*--*

*-- Create foreign key*

*--*

ALTER TABLE bus

ADD CONSTRAINT bus\_ibfk\_4 FOREIGN KEY (id\_company)

REFERENCES passenger\_insurance\_company (id\_company);

*--*

*-- Dumping data for table dispatcher*

*--*

INSERT INTO dispatcher VALUES

(1, 'Татьяна', 'Петровна'),

(2, 'Алекса', 'Маршова'),

(3, 'Акакий', 'Персов'),

(4, 'Владислав', 'Чикатило'),

(5, 'Афанасья', 'Мармеладова');

*--*

*-- Dumping data for table bus\_flight*

*--*

INSERT INTO bus\_flight VALUES

(1, 'Медведково', 'Китай-город', 774),

(2, 'метро Некрасовка', 'Орехово', 321),

(3, 'Шляпково', 'Салатовка', 123),

(4, 'Арханельская улица', 'ДЕПО Кирзанов', 21),

(5, 'Широкая улица', 'улица Вернадского', 80),

(6, 'метро Юго-западная', 'метро Раменки', 664);

*--*

*-- Dumping data for table validator*

*--*

INSERT INTO validator VALUES

(1, 'да'),

(2, 'да'),

(3, 'нет'),

(4, 'да'),

(5, 'да'),

(6, 'да'),

(7, 'да'),

(8, 'да'),

(9, 'да'),

(10, 'да'),

(11, 'да'),

(12, 'да'),

(13, 'да'),

(14, 'да'),

(15, 'да'),

(16, 'нет'),

(17, 'да'),

(18, 'да'),

(19, 'нет'),

(20, 'да'),

(21, 'нет'),

(22, 'да'),

(23, 'да'),

(24, 'нет'),

(25, 'да'),

(26, 'да'),

(27, 'да');

*--*

*-- Dumping data for table passenger\_insurance\_company*

*--*

INSERT INTO passenger\_insurance\_company VALUES

(1, 'АО С2ОГАЗ', 221479, '2006-09-20'),

(2, 'АО СОГАЗ', 223231, '2006-09-20'),

(3, 'АО ВТБ', 313279, '2016-03-20'),

(4, 'БВБК Страхование', 9696432, '2001-02-20'),

(5, 'CберСтрахование', 228322, '2020-01-20');

*--*

*-- Dumping data for table bus\_driver*

*--*

INSERT INTO bus\_driver VALUES

(1, 'Платон Крышесносов', 15, 234, 'AA10230D', 1, 2),

(2, 'Алишер Белозубов', 2, 234, 'AB2340A', 3, 1),

(3, 'Мария Блаженая', 10, NULL, 'AA15292H', 5, 3),

(4, 'Златон Ибрагимов', 9, 234, 'B249955D', 3, 1),

(5, 'Марк Золорылов', 21, 234, 'A1439141D', 4, 4),

(6, 'Игорь Индюков', 4, 234, 'AA103203D', 4, 5),

(7, 'Михаил Мафиозов', 7, 234, 'AA4525G', 5, 6),

(8, 'Эсми Мирзалиева', 25, 234, 'AA10423DD', 1, 6),

(9, 'Зокир Джамшутов', 4, 234, 'AA10425W', 2, 2),

(10, 'Мишутка Алфертов', 30, 234, 'A4141D', 1, 2),

(11, 'Али Белокуров', 6, 234, 'B7474', 3, 1),

(12, 'Мар Блаженая', 10, NULL, 'AA15192H', 5, 3),

(13, 'Злат ЗАбрагимов', 10, NULL, 'B219955D', 3, 1),

(14, 'Марк рылов', 21, 234, 'A1439741D', 4, 4),

(15, 'Игорь Индюк', 4, 234, 'AA109203D', 4, 5),

(16, 'Евгений Павлов', 7, 234, 'AA43525G', 5, 6),

(17, 'Эсвоьд Мирзалиев', 25, 234, 'A150423DD', 1, 6),

(18, 'Оскар Джамшутов', 5, 234, 'AA10485W', 2, 2);

*--*

*-- Dumping data for table advertising\_on\_the\_bus*

*--*

INSERT INTO advertising\_on\_the\_bus VALUES

(1, 'МТС', '2003-02-20', '2003-02-20'),

(2, 'Cбербанк', '2012-01-20', '2012-02-20'),

(3, 'ВафляShop', '2003-02-20', '2003-03-20'),

(4, 'ShocoShop', '2003-10-20', '2006-12-20'),

(5, 'ShipBorard', '2008-09-20', '2003-02-20'),

(6, 'MagomedShop', '2003-02-20', '2010-06-20');

*--*

*-- Dumping data for table bus*

*--*

INSERT INTO bus VALUES

(1, 772, 'Электробус', 1, 1, 4, 2),

(3, 772, 'Обычный автобус', 2, 2, 3, 1),

(4, 772, 'Электробус', 3, 3, 6, 2),

(5, 772, 'Электробус', 4, 4, 6, 3),

(6, 72, 'Электробус', 5, 5, 5, 5),

(7, 1, 'Обычный автобус', 6, 6, 5, 1),

(8, 3, 'Электробус', 7, 7, 2, 3),

(9, 10, 'Обычный автоьус', 8, 8, 1, 2),

(10, 553, 'Электробус', 9, 9, 4, 5),

(11, 1, 'Обычный автобус', 2, 2, 3, 1),

(20, 456, 'Обычный автобус', 6, 10, 3, 1),

(21, 966, 'Электробус', 9, 11, 5, 5),

(22, 453, 'Обычный автобус', 9, 6, 5, 1),

(23, 353, 'Электробус', 1, 13, 2, 3),

(24, 105, 'Обычный автоьус', 14, 8, 1, 2),

(25, 553, 'Электробус', 3, 16, 4, 5),

(26, 666, 'Электробус', 6, 18, 6, 2),

(27, 78, 'Электробус', 7, 4, 6, 3);

*--*

*-- Restore previous SQL mode*

*--*

*/\*!40101 SET SQL\_MODE=@OLD\_SQL\_MODE \*/*;

*--*

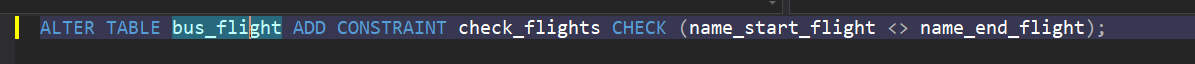
*-- Enable foreign keys*

*--*

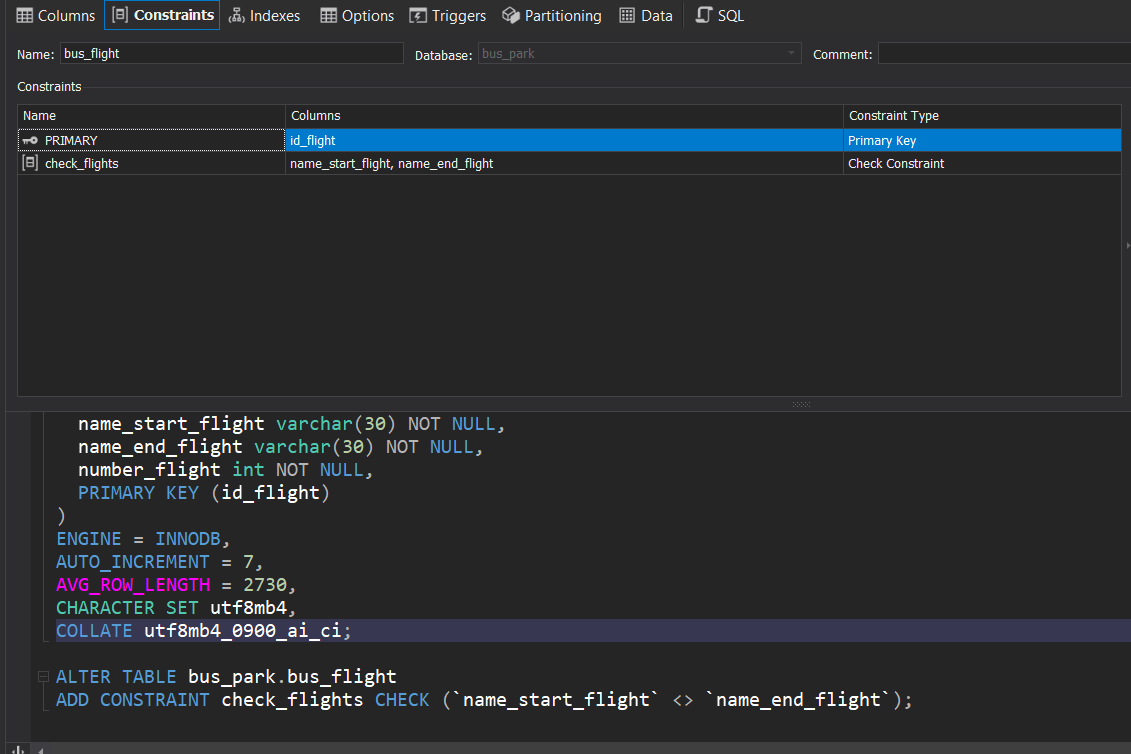
*/\*!40014 SET FOREIGN\_KEY\_CHECKS = @OLD\_FOREIGN\_KEY\_CHECKS \*/*;

Пример 1. Добавим ограничение уникальности на номер водительских прав водителя:

Пример 2. Добавим ограничение проверки условия для таблицы автобусных рейсов , означающее, что название начала рейса не должно быть равно названию конца рейса



Покажем успешность добавления:

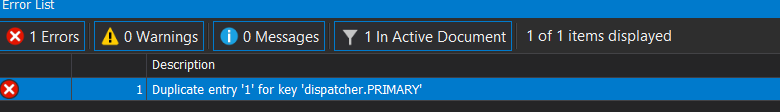


**1.3. Команды модификации данных (DML)**

Попытаемся вставить в таблицу диспетчеров с id, который уже существует в базе:



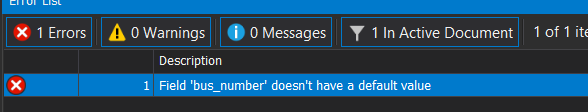
Ошибка, которая показывает, в чем мы ошиблись:



Попробуем сделать запись с нарушением ограничения внешнего ключа. Для примера возьмем таблицу bus и попытаемся вставить id водителя, которого не существует:



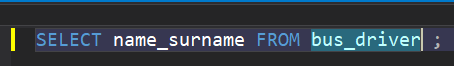
Ошибка:

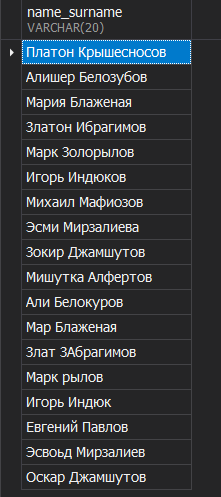


**1.4. Выборка данных. Оператор SELECT (DQL)**

**Запрос 1. Операции проекции.** Осуществляется выбор только части полей таблицы, т.е. производится вертикальная выборка данных.

Распечатаем имена всех водителей:



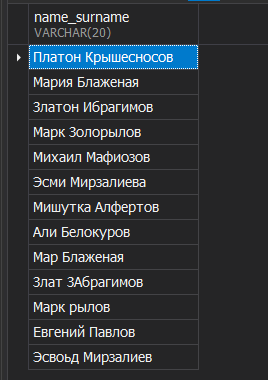


**Запрос 2. Операции селекции.** Осуществляется горизонтальная выборка

– в результат попадают только записи, удовлетворяющие условию.

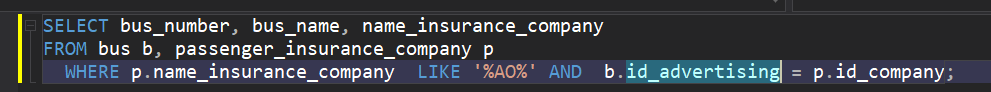
Для примера распечатаем имя и фамилию водителя у которого стаж работы

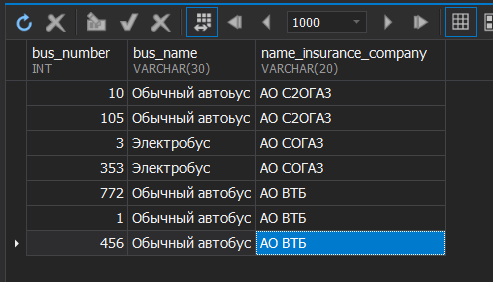




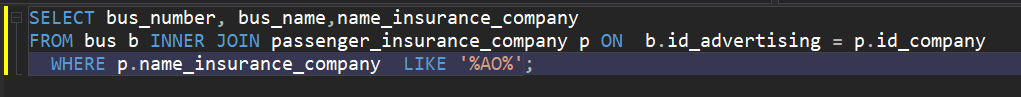
**Запрос 3. Операции соединения.** Здесь следует выделить декартово произведение и на его основе соединение по условию, а также естественное соединение (по одноименным полям или равенству полей с одинаковым смыслом).

Для примера выведем номер автобуса, его название и название страховой компании в этом автобусе, которая имеет аббревиатуру на «АО»





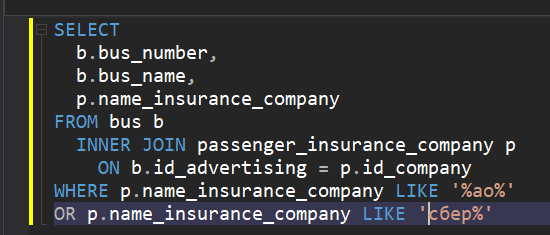
Запишем этот же запрос с помощью операции внутреннего соединения

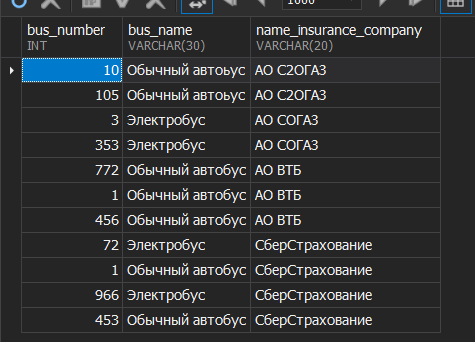


Результат будет таким же.

**Запрос 4. Операция объединения.** Теоретико-множественные операции

часто можно записать с помощью логических операций, примененных в конструкции WHERE запроса



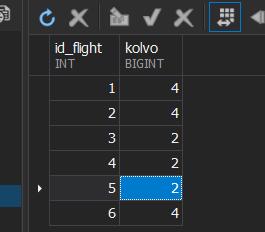


**Запрос 5. Операции группировки.** Эта операция связана со своеобразной

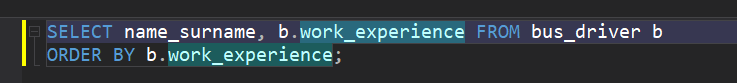
сверткой таблицы по полям группировки.

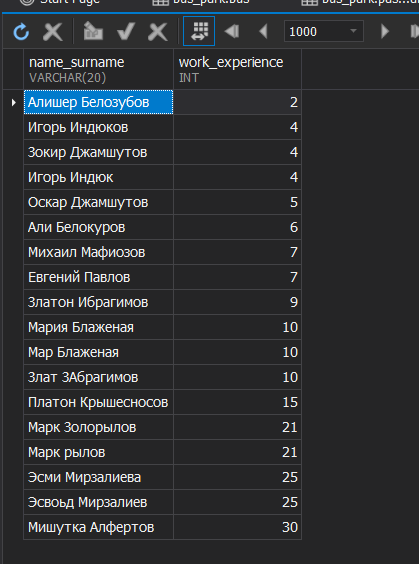
Выведем рейс и количество водителей, которые ездят на нем:





**Запрос 6. Операция сортировки.** Вывести имя и фамилию и стаж работы, отсортированной в возрастающем порядке по опыту работы.

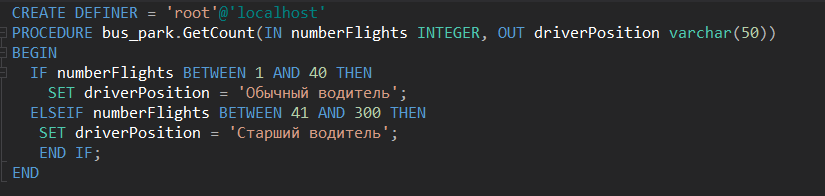




**1.5. ХРАНИМЫЕ ПРОЦЕДУРЫ, ФУНКЦИИ И ТРИГГЕРЫ**

**Пример 1.**  Напишем хранимую процедуру, которая получает в качестве входного параметра количество рейсов, которые проделал водитель автобуса и на основании шкалы проделанных рейсах вычисляет должность этого водителя.

Покажем скрипт для хранимой процедуры:

**

*Рис 10– скрипт для работы с хранимой процедурой*

Вывод:

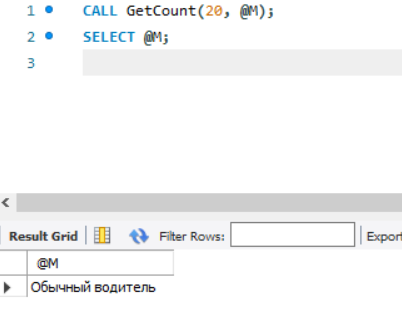


Рис – вызов процедуры и печать результата (1)

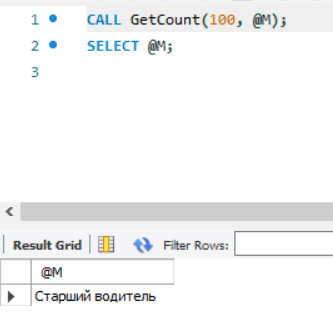


Рис – вызов процедуры и печать результата (2)

**Пример 2. (Функция)**

Cоздадим функцию для получения должности водители по количеству проделанных рейсов.

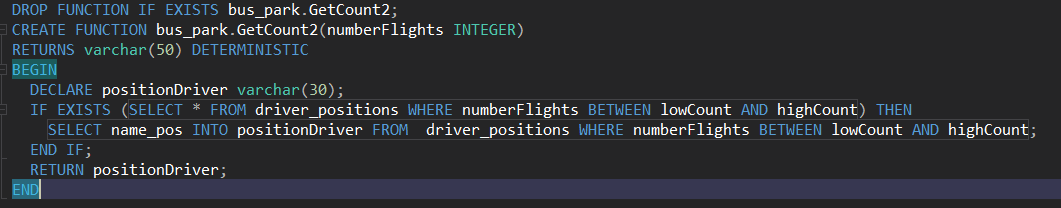


Рисунок – скрипт функции

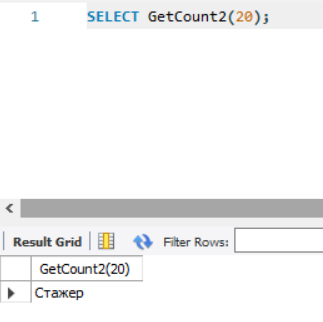


Рисунок – Вывод функции при количестве рейсов - 20

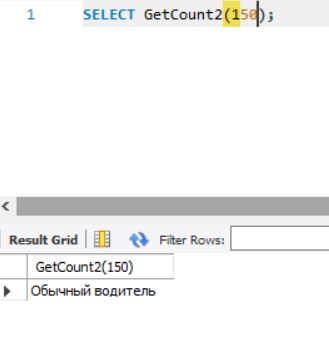


Рисунок – Вывод функции при количестве рейсов – 150

**Пример 3. (Триггер)**

В этом примере представлен скрипт триггера на вставку новой записи в таблицу количества рейсов водителя. Этот триггер должен срабатывать после вставки и быть связан с подсчетом актуального количества рейсов

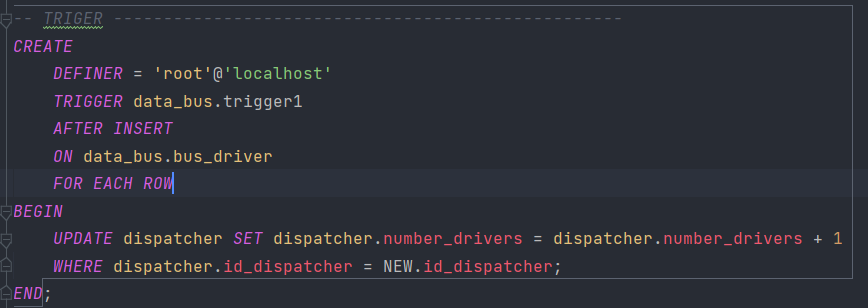
****

Рисунок – скрипт триггера