Министерство образования Республики Беларусь

Учреждение образования

«Витебский государственный технологический университет»

Кафедра «Информационные системы и автоматизация производства»

Расчётно-графическая работа

по дисциплине

«Базы данных»

По теме: «Автопарк»

Выполнила: студентка гр. 2Ит-7

Тарбеева Н.А.

Проверил: ст. пр.

Черненко Д. В.

Витебск, 2020

Для того, чтобы создать базу данных, посредством запросов, необходимо описание предметной области: «Автопарк» для понимания структуры будущей базы данных.

Описание предметной области:

Заказчик обращается, с целью перевозки груза, за ним закрепляется водитель, которого отправляют в рейс и назначают ему подходящий автомобиль.

КАЖДЫЙ ВОДИТЕЛЬ может отправиться только в один рейс.

КАЖДЫЙ АВТОМОБИЛЬ может быть привязан только к одному водителю.

Из данного описания мы можем вынести 4 “предмета”:

* Заказчик;
* Водитель;
* Рейс;
* Автомобиль.

Теперь, используя MySQL Workbench, мы можем создать базу данных, состоящую из 4 таблиц в которых будут следующие поля:

Заказчик(Customer):

* ID(поле будет являться первичным ключом, необходимым для связи таблиц);
* Name(ФИО заказчика);
* Tel(телефонный номер заказчика);
* Address(домашний адрес заказчика);

Водитель(Driver):

* ID(поле будет являться первичным ключом, необходимым для связи таблиц);
* Name(ФИО водителя);
* Phone(телефонный номер водителя);
* License(категории прав);
* Driving\_exeperience(стаж вождения);
* ID\_Auto(поле будет являться вторичным ключом, необходимым для связи таблиц);
* ID\_Trip(поле будет являться вторичным ключом, необходимым для связи таблиц);

Рейс(Trip):

* ID(поле будет являться первичным ключом, необходимым для связи таблиц);
* Data(дата отправления в рейс);
* Сargo(тип, перевозимого груза);
* Point\_of\_departure(точка отправления);
* Destinetion(город прибытия);
* ID\_Driver

Автомобиль (Auto):

* ID(поле будет являться первичным ключом, необходимым для уникальности строк таблицы);
* Registration\_number(регистрационный номер автомобиля);
* Model(марка автомобиля);
* Carrying\_capacity(грузоподъемность);
* Repair\_date(дата ремонта);
* ID\_Customer(поле будет являться вторичным ключом, необходимым для связи таблиц);
* ID Driver(поле будет являться вторичным ключом, необходимым для связи таблиц)

Построение базы данных произведём посредством запросов на языке SQL в MySQL Workbench:

* CREATE TABLE Customer (ID int, Name char(50), Tel char(50), Address char(50), PRIMARY KEY(ID)); -Создание таблицы Customer;
* CREATE TABLE Driver (ID int, Name char(50), License char(50), Phone char(50), Driving\_experience Int, ID\_Auto Int, ID\_Trip Int,

PRIMARY KEY(ID),

FOREIGN KEY (ID\_Auto) REFERENCES Auto (ID), FOREIGN KEY (ID\_Trip) REFERENCES Trip (ID));-Создание таблицы Driver;

* CREATE TABLE Trip (ID INT, Data DATETIME, Cargo CHAR(50),Point\_of\_departure CHAR(50),Desctination CHAR(50),ID\_Driver INT,PRIMARY KEY (ID)); -Создание таблицы Trip;
* CREATE TABLE Auto (ID int, Registration\_number char(50), Model char(50),Carrying\_capacity Int, Repair\_date DATETIME,ID\_Driver Int, ID\_Customer Int, PRIMARY KEY(ID),

foreign key(ID\_Customer) references Customer(ID));-Создание таблицы Auto;

После выполнения данных запросов мы получаем базу данных, схема которой приведена на Рисунке 1:

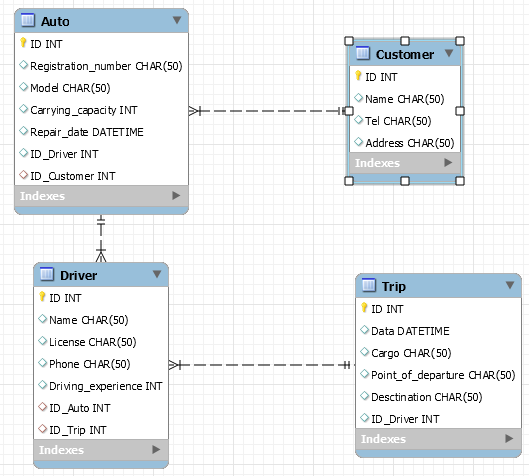


Рисунок 1.Схема базы данных

Теперь произведём заполнение таблиц посредством запросов:

Заполнение таблицы Customer:

INSERT INTO Customer VALUES (1, "Петров Василий Петрович", "8(029)586-21-23", "г.Витебск пр-т Московский 76-32");

INSERT INTO Customer VALUES (2, "Рашкевич Анастасия Владимировна", " 8(033)123-45-68", "г.Витебск ул.Сметанина 12-106");

INSERT INTO Customer VALUES (3, "Тарбеева Ника Андреевна", "8(029)845-23-61", "г.Витебск пр-т Черняховского 32-17");

INSERT INTO Customer VALUES (4, "Федорцов Павел Алексеевич", "8(029)231-25-12", "г.Минск пр-т Независимости 235-2-120");

INSERT INTO Customer VALUES (5, "Сякин Ян Александрович", "8(025)152-10-78", "г.Витебск пр-т Победы 3-52");

Результат выполнения запросов:

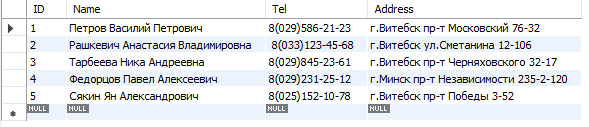


Рисунок 2. Результат заполнения таблицы Customer

Заполнение таблицы Driver:

INSERT INTO Driver VALUES (1, "Панкевич Дмитрий", "B + C + D", "8(033)452-13-58", 5, 2, 1);

INSERT INTO Driver VALUES (2, "Васильев Василий", "А + B + C + D", "8(033)862-15-33", 25, 1, 2);

INSERT INTO Driver VALUES (3, "Константинов Илья", "А + B + C + D", "8(025)658-71-20 ", 18, 3, 3);

INSERT INTO Driver VALUES (4, "Гришаев Даниил", "А + B + C + D", "8(029)582-12-39", 9, 5, 5);

INSERT INTO Driver VALUES (5, "Лебедев Леонид", "B + C + D", "8(029)445-52-65", 27, 4, 4);

Результат выполнения запросов:

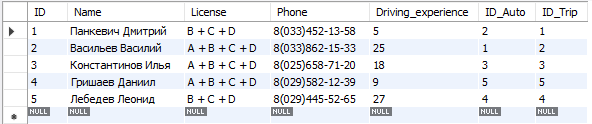


Рисунок 3. Результат заполнения таблицы Driver

Аналогичным образом заполняем таблицу Trip(рис.4)

INSERT INTO Auto VALUES (1, "2320 BM-2", "Opel", 30, "26.12.2019", 1, 3);

INSERT INTO Auto VALUES (2, "1123 BN-7", "Mercedes-Benz", 45, "02.02.2020", 3, 2);

INSERT INTO Auto VALUES (3, "3218 AM-5", "Honda", 8, "05.09.2019", 5, 1);

INSERT INTO Auto VALUES (4, "0504 EB-2", "Volkswagen", 38, "26.02.2019", 4, 5);

INSERT INTO Auto VALUES (5, "1125 EB-7", "Volkswagen", 60, "22.02.2020", 2, 4);

И Auto(рис.5):

insert into Trip values(1, "29.02.2020", "Продукты питания", "Минск", "Вильнюс", 2);

insert into Trip values(2, "09.04.2020", "Нефть", "Санкт-Петербург", "Минск", 1);

insert into Trip values(3, "08.03.2020", "Медикаменты", "Брест", "Варшава",4);

insert into Trip values(4, "19.04.2020", "Парфюмерия", "Витебск", "Люксембург", 5);

insert into Trip values(5, "03.04.2020", "Верхняя одежда", "Могилев", "Москва", 3);

В результате получаем:

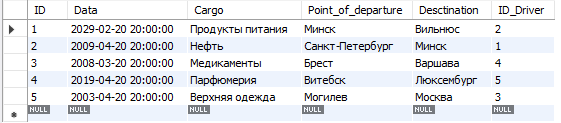


Рисунок 4. Результат заполнения таблицы Trip

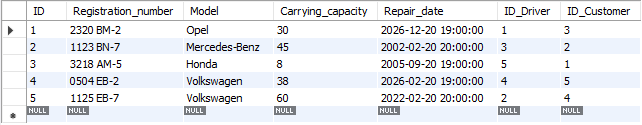


Рисунок 5. Результат заполнения таблицы Auto

Теперь, когда таблицы созданы, связаны и заполнены можно приступать к выборке при помощи SELECT:

**Фильтрация**

* Вывести имена всех водителей у которых стаж вождения больше 9:

SELECT Name FROM Driver WHERE (Driving\_experience > 9);

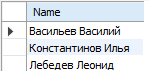


Рисунок 6.

**Вычисляемое поле результирующей таблицы**

* Вывести регистрационный номер автомобиля и его вместимость в кг

SELECT Registration\_number,

round(Carrying\_capacity\*1000) as Vmestimost\_v\_kg

FROM Auto;

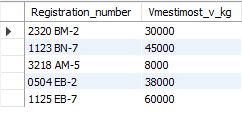


Рисунок 7.

**Объединение нескольких таблиц в источнике выборки**

* Вывести Регистрационный номер автомобиля, ФИО водителя и ФИО заказчика

select auto.Registration\_number, driver.Name, customer.Name

from customer join

(

auto inner join

driver on auto.ID = driver.ID\_Auto

)

on customer.ID = driver.ID\_Auto

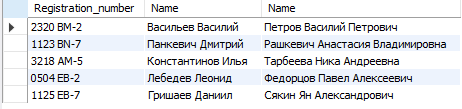


Рисунок 8.

**Группировка**

* Вывести автомобиль, закрепленный за водителем

select auto.Model as Марка\_автомобиля,

driver.Name as Фамилия\_Имя\_водителя

from auto inner join driver

on auto.ID = driver.ID\_auto

group by Auto.model;

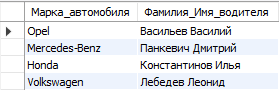


Рисунок 9.

**Фильтрация сгруппированной таблицы**

* Вывести марки автомобиля и водителей(у которых в ФИО есть буква «о»), закрепленных за этими автомобилями.

select auto.Model as Марка\_автомобиля,

driver.Name as Фамилия\_Имя\_водителя

from auto inner join driver

on auto.ID = driver.ID\_auto

group by Auto.model

having driver.Name like '%о%';



Рисунок 10.

**Простые подзапросы**

Подзапрос — это запрос на выборку данных, вложенный в другой запрос. Простые подзапросы характеризуются тем, что они формально никак не связаны с содержащими их внешними запросами. Это позволяет сначала выполнить подзапрос, результат которого затем используется для выполнения внешнего запроса.

**Подзапросы, возвращающие единственное значение**

* Выводит водителя, который отправляется в рейс из Минска

SELECT Name

FROM driver

WHERE ID\_Trip=(SELECT ID FROM trip where Point\_of\_departure='Минск');



Рисунок 11.

**Подзапросы, возвращающие список значений из одного столбца таблицы**

* Выводит водителей, которые отправляются в рейс на автомобиле Volkswagen

SELECT Name FROM driver

WHERE ID\_Auto=

SOME(SELECT ID FROM auto WHERE

Model="Volkswagen");



Рисунок 12.

**Использование объединения UNION**

Результирующая таблица, полученная в результате выполнения запроса с использованием объединения UNION, будет представлять собой вертикальное объединение двух и более запросов (т.е. объединение записей).

* Выводит сведения о книгах, расположив их в три ценовые категории: 'низкая грузоподъемность', 'средняя грузоподъемность', ' высокая грузоподъемность' с грузоподъемностью до 8 и свыше 60 соответственно.

SELECT Registration\_number, Carrying\_capacity, 'низкая грузоподъемность' AS Уровень\_грзоподъемности

FROM auto WHERE Carrying\_capacity<30

UNION

SELECT Registration\_number, Carrying\_capacity, 'средняя грузоподъемность' AS Уровень\_грзоподъемности

FROM auto WHERE Carrying\_capacity>=30 and Carrying\_capacity<=40

UNION

SELECT Registration\_number, Carrying\_capacity, 'высокая грузоподъемность' AS Уровень\_грзоподъемности

FROM auto WHERE Carrying\_capacity>40

ORDER BY Carrying\_capacity

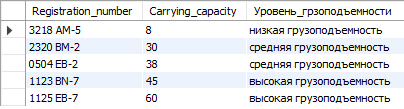


Рисунок 13.