Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

Рязанский государственный радиотехнический университет  
имени В.Ф. Уткина

Кафедра ЭВМ

К защите

Руководитель работы:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

дата, подпись

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОЙ РАБОТЕ**

по дисциплине

**«Базы данных»**

Тема:

«Разработка базы данных для предметной области:

Служба такси»

Выполнил студент группы 143

Вербицкая И.С.

дата сдачи на проверку, подпись

Руководитель работы

ассистент каф. ЭВМ

Хизриева Н.И.

оценка дата защиты, подпись

Оглавление

[Введение 3](#_Toc122410740)

[1 Семантическое описание предметной области 4](#_Toc122410741)

[2 Разработка базы данных 5](#_Toc122410742)

[2.1 Разработка логической модели данных 5](#_Toc122410743)

[2.2 Определение типов хранимых данных 7](#_Toc122410744)

[2.3 Задание ограничений целостности данных 8](#_Toc122410745)

[2.4 Построение схемы БД 9](#_Toc122410746)

[2.5 Разработка сценария для создания БД и основных объектов структуры БД 10](#_Toc122410747)

[Создание файлов БД 10](#_Toc122410748)

[Создание и заполнение таблиц 10](#_Toc122410749)

[3 Разработка основных операций реляционной алгебры 15](#_Toc122410750)

[4 Разработка запросов к БД для реализации информационных потребностей пользователя 18](#_Toc122410751)

[Заключение 34](#_Toc122410752)

[Список литературы 35](#_Toc122410753)

# Введение

Не секрет, что практически любая крупная разработка не обходится без использования баз данных. Будь то работа с данными клиентов интернет-магазина или исходными данными для текстовой интерактивной игры-новеллы. Именно поэтому умение работать с базами данных – один из основных, базовых навыков для разработчика и крайне важно уметь этим навыком грамотно пользоваться.

Для работы с базами данных существуют СУБД – системы управления базами данных. СУБД бывают разных видов и могут работать с различными моделями данных. Одной из самых удобных и популярных моделей данных является РМД – реляционная модель данных. Она позволяет представлять данные в удобном и интуитивно понятном виде: в виде таблиц. С РМД работает и СУБД Microsoft SQL Server – именно в этой среде будет выполняться данная курсовая работа.

Согласно заданию, необходимо разработать работоспособную базу данных в предметной области «Служба такси». Выполнение этого задания позволит изучить основные принципы составления баз данных и работы с ними. Для выполнения данной работы потребуется научиться грамотно формулировать и составлять запросы на языке реляционной алгебры и языке программирования SQL.

Работа в выбранной предметной области имеет важное практическое значение. Она позволит научиться «реляционному мышлению», а также изучить основы работы с базами данных, что с большой вероятностью пригодится в дальнейшем. Эта работа даст бесценный опыт в разработке структуры БД и работы с ней.

# 1 Семантическое описание предметной области

База данных «Служба такси» содержит информацию о типах автомобилей, типах тарифов, клиентах и поездках.

Любая поездка на такси совершается на каком-либо виде автомобиля – будь то дорогой или экономный вариант. Информация о типах автомобилей включает в себя название типа автомобиля (названия представлены следующим множеством: {эконом, комфорт, бизнес, премиум, люкс}), описание типа и стоимость поездки за километр. Могут быть записи об автомобилях, для которых не указано никакого описания.

Если поездка на такси необычная, например, если среди пассажиров есть дети или животные, то для такой поездки требуется указывать тип тарифа. О типе тарифа необходимо хранить следующую информацию: название (названия представляют собой следующее множество: {обычный, детское кресло, бустер, перевозка животных, малогабаритный багаж, крупногабаритный багаж}), описание, доплата. Могут быть записи о тарифах, для которых не указано описание.

При заказе такси клиент сообщает свои данные службе такси. Информация о клиентах включает в себя ФИО клиента, номер его телефона, а также номер банковской карты для оплаты; кроме того, каждому клиенту должен соответствовать собственный табельный номер, чтобы можно было различать записи об однофамильцах. Могут быть записи о клиентах, которые не указали номер банковской карты.

Наконец, служба такси хранит информацию о самих совершенных поездках. Эта информация включает в себя номер клиента, заказавшего такси, тип выбранных автомобиля и тарифа, адрес назначения, дату начала поездки и время начала поездки, а также расстояние в километрах.

# 2 Разработка базы данных

## 2.1 Разработка логической модели данных

***Описание основных сущностей ПО.***

С учетом информации, которую необходимо хранить, в заданной предметной области «Служба такси» можно выделить четыре сущности: Автомобиль, Тариф, Клиент, Поездка. Определим назначение для каждой сущности (таблица 1).

Таблица 1. Список сущностей.

|  |  |  |
| --- | --- | --- |
| № | Название | Назначение |
| 1 | Автомобиль | Перечень доступных в службе такси типов автомобилей в зависимости от уровня обслуживания и стоимости автомобиля. |
| 2 | Тариф | Перечень доступных при вызове такси тарифов в зависимости от специфики пассажирского состава или перевозимого груза. |
| 3 | Клиент | Перечень клиентов, вызывавших такси. |
| 4 | Поездка | Перечень совершенных клиентами поездок. |

Выделим атрибуты выбранных сущностей и выделим среди них ключевые и неключевые (таблицы 2-5).

Таблица 2. Список атрибутов таблицы «Автомобиль»

|  |  |  |
| --- | --- | --- |
| Ключевое поле | Название | Назначение |
| ПК (первичный ключ) | Название | Первичный ключ. Уникальное название, соответствующее каждому типу автомобиля. |
|  | Описание |  |
|  | Стоимость |  |

Таблица 3. Список атрибутов таблицы «Тариф»

|  |  |  |
| --- | --- | --- |
| Ключевое поле | Название | Назначение |
| ПК (первичный ключ) | Название | Первичный ключ. Уникальное название, соответствующее каждому типу тарифа. |
|  | Описание |  |
|  | Доплата |  |

Таблица 4. Список атрибутов таблицы «Клиент»

|  |  |  |
| --- | --- | --- |
| Ключевое поле | Название | Назначение |
| ПК (первичный ключ) | Номер | Первичный ключ. Уникальный номер, соответствующий каждому клиенту. Значение автоматически генерируется СУБД при вставке новой записи в таблицу. |
|  | ФИО |  |
|  | Телефон |  |
|  | Карта |  |

Таблица 5. Список атрибутов таблицы «Поездка»

|  |  |  |
| --- | --- | --- |
| Ключевое поле | Название | Назначение |
| ПК (первичный ключ)  ВК (внешний ключ) | Клиент | Является частью составного первичного ключа.  Внешний ключ. Позволяет определить какой клиент заказывал такси. |
| ВК (внешний ключ) | Автомобиль | Внешний ключ. Позволяет определить какой тип автомобиля был выбран для поездки. |
| ВК (внешний ключ) | Тариф | Внешний ключ. Позволяет определить какой тариф был выбран для поездки. |
|  | Адрес |  |
| ПК (первичный ключ) | Дата | Является частью составного первичного ключа. |
| ПК (первичный ключ) | Время | Является частью составного первичного ключа. |
|  | Расстояние |  |

***Построение логической модели.***

Выделим связи между сущностями и определим типы этих связей (таблица 6). Представим получившуюся логическую модель в графическом виде (рисунок 1).

Таблица 6. Список связей

|  |  |  |  |
| --- | --- | --- | --- |
| № | Сущности, участвующие в связи | Тип связи | Обоснование |
| 1 | Клиент - Поездка | 1:N | Один клиент может заказать несколько поездок, но каждая поездка может быть заказана только одним клиентом. |
| 2 | Автомобиль - Поездка | 1:N | Один тип автомобиля может использоваться в огромном количестве поездок, но для каждой конкретной поездки может быть выбран только один тип автомобиля. |
| 3 | Тариф - Поездка | 1:N | Каждому типу тарифа может соответствовать несколько поездок, но для конкретной поездки может быть выбран только один тариф. |



Рисунок 1 – Логическая модель предметной области «Служба такси»

## 2.2 Определение типов хранимых данных

Для каждой таблицы определим типы ее атрибутов в среде СУБД Microsoft SQL (таблицы 7-10).

Таблица 7. Список атрибутов таблицы «Автомобиль»

|  |  |
| --- | --- |
| Название | Тип данных |
| Название | varchar(20) |
| Описание | varchar(1000) |
| Стоимость | money |

Таблица 8. Список атрибутов таблицы «Тариф»

|  |  |
| --- | --- |
| Название | Тип данных |
| Название | varchar(30) |
| Описание | varchar(1000) |
| Доплата | money |

Таблица 9. Список атрибутов таблицы «Клиент»

|  |  |
| --- | --- |
| Название | Тип данных |
| Номер | int, нач. значение = 1, приращение = 1 |
| ФИО | varchar(30) |
| Телефон | varchar(17) |
| Карта | varchar(19) |

Таблица 10. Список атрибутов таблицы «Поездка»

|  |  |
| --- | --- |
| Название | Тип данных |
| Клиент | int |
| Автомобиль | varchar(20) |
| Тариф | varchar(20) |
| Адрес | varchar(50) |
| Дата | date |
| Время | time(7) |
| Расстояние | float |

## 2.3 Задание ограничений целостности данных

Определим ограничения атрибутов для каждой таблицы. Результат представлен в таблицах 11-14. Заметим также, что для всех приведенных ниже атрибутов действует запрет на использование NULL-значений, если не указано обратное.

Таблица 11. Ограничения таблицы «Автомобиль»

|  |  |
| --- | --- |
| Название | Ограничение |
| Название | Эконом, комфорт, бизнес, премиум, люкс. Первичный ключ (уникальное значение для каждой отдельной записи). |
| Описание | Разрешение на использование NULL-значения. |
| Стоимость |  |

Таблица 12. Ограничения таблицы «Тариф»

|  |  |
| --- | --- |
| Название | Ограничение |
| Название | Детское кресло, бустер, перевозка животных, малогабаритный багаж, крупногабаритный багаж.  Первичный ключ (уникальное значение для каждой отдельной записи). |
| Описание | Разрешение на использование NULL-значения. |
| Доплата |  |

Таблица 13. Ограничения таблицы «Клиент»

|  |  |
| --- | --- |
| Название | Ограничение |
| Номер | Первичный ключ (уникальное значение для каждой отдельной записи). Значения автоматически генерируются СУБД, начиная от 1 с шагом 1. |
| ФИО |  |
| Телефон | +7 (NNN) NNN-NN-NN, где N – цифра от 0 до 9. Также является уникальным. |
| Карта | NNNN NNNN NNNN NNNN, где N – цифра от 0 до 9. Разрешение на использование NULL-значения. |

Таблица 14. Ограничения таблицы «Поездка»

|  |  |
| --- | --- |
| Название | Ограничение |
| Клиент | Часть составного первичного ключа. Внешний ключ. |
| Автомобиль | Внешний ключ. |
| Тариф | Внешний ключ. |
| Адрес |  |
| Дата | Часть составного первичного ключа.  ДД-ММ-ГГГГ |
| Время | Часть составного первичного ключа.  ЧЧ-ММ-СС |
| Расстояние | Значение должно быть больше нуля. |

## 2.4 Построение схемы БД

Для наглядного представления схемы базы данных воспользуемся средой Microsoft SQL, в которой отдельно зададим таблицы с атрибутами, а затем объединим их в диаграмму. На рисунке 2 представлена получившаяся диаграмма БД «Служба такси», иллюстрирующая состав атрибутов каждой таблицы, ограничения и типы этих атрибутов, а также связи между таблицами.



Рисунок 2 – Схема базы данных «Служба такси»

## 2.5 Разработка сценария для создания БД и основных объектов структуры БД

### Создание файлов БД

USE master

CREATE DATABASE СлужбаТакси

ON

(NAME='143\_Вербицкая\_БД',

FILENAME='C:\Program Files\Microsoft SQL Server\MSSQL15.SQLEXPRESS\MSSQL\DATA\143\_Вербицкая\_БД.mdf',

SIZE=1,

MAXSIZE=10,

FILEGROWTH=2)

LOG ON

(NAME='143\_Вербицкая\_БД\_log',

FILENAME='C:\Program Files\Microsoft SQL Server\MSSQL15.SQLEXPRESS\MSSQL\DATA\143\_Вербицкая\_БД\_log.ldf',

SIZE=1,

MAXSIZE=20,

FILEGROWTH=2)

GO

### Создание и заполнение таблиц

USE СлужбаТакси

CREATE TABLE Тариф

(Название varchar(30) not null default 'Обычный'

check (Название in ('Бустер','Детское кресло','Крупногабаритный багаж',

'Малогабаритный багаж','Обычный','Перевозка животных')),

Описание varchar(1000),

Стоимость money not null default 0)

GO

ALTER TABLE Тариф

ADD CONSTRAINT PK\_Тариф primary key (Название)

GO

CREATE TABLE Клиент

(Номер int identity(1,1) not null,

ФИО varchar(30) not null,

Телефон varchar(17) not null

check (Телефон like '+7([0-9][0-9][0-9]) [0-9][0-9][0-9]-[0-9][0-9]-[0-9][0-9]'),

Карта varchar(19) check (Карта like

'[0-9][0-9][0-9][0-9] [0-9][0-9][0-9][0-9] [0-9][0-9][0-9][0-9] [0-9][0-9][0-9][0-9]'))

GO

ALTER TABLE Клиент

ADD CONSTRAINT PK\_Клиент primary key (Номер)

GO

CREATE TABLE Автомобиль

(Название varchar(20) not null default 'Эконом'

check (Название in ('Бизнес','Комфорт','Люкс','Премиум','Эконом')),

Описание varchar(1000),

Стоимость money not null default 50)

GO

ALTER TABLE Автомобиль

ADD CONSTRAINT PK\_Автомобиль primary key (Название)

GO

CREATE TABLE Поездка

(Клиент int not null,

Автомобиль varchar(20) not null default 'Эконом'

check (Автомобиль in ('Бизнес','Комфорт','Люкс','Премиум','Эконом')),

Тариф varchar(30) not null default 'Обычный'

check (Тариф in ('Бустер','Детское кресло','Крупногабаритный багаж',

'Малогабаритный багаж','Обычный','Перевозка животных')),

Адрес varchar(50) not null,

Дата date not null,

Время time not null,

Расстояние float not null check (Расстояние>0))

GO

ALTER TABLE Поездка

ADD CONSTRAINT PK\_Поездка primary key (Клиент,Дата,Время)

GO

ALTER TABLE Поездка

ADD CONSTRAINT FK\_Поездка\_Автомобиль foreign key (Автомобиль) references Автомобиль(Название)

ALTER TABLE Поездка

ADD CONSTRAINT FK\_Поездка\_Клиент foreign key (Клиент) references Клиент(Номер)

ALTER TABLE Поездка

ADD CONSTRAINT FK\_Поездка\_Тариф foreign key (Тариф) references Тариф(Название)

INSERT INTO Автомобиль VALUES

('Бизнес',

'Поездка в удобной машине. Дополнительно прилагается

приятная компания в лице водителя-бизнесмена,

таксующего только для души.',

125),

('Комфорт',

'Для тех, кому от автомобиля нужно чуть больше чем

сидение. Более того, сидение будет даже чистым,

а в машине не будет неприятно пахнуть.', 100),

('Люкс',

'Шикарная машина с массажными креслами и водителем -

народным артистом РФ ждет вас всего за 300 рублей

за километр. Специально для интровертов доступен

вариант с немым водителем.', 200),

('Премиум',

'Хотите - пейте в машине кофе, хотите - вылейте его

на водителя. Любой каприз за ваши деньги. Кофе

предлагается пассажиру бесплатно.', 150),

('Эконом',

'Поездка на недорогом автомобиле, и ничего лишнего.

Из лишнего в машине может оказаться только конфликтный

водитель и громкая музыка жанра "шансон".', 50)

GO

DBCC CHECKIDENT('Клиент',RESEED,0) --сброс счётчика

GO

INSERT INTO Клиент VALUES

('Иванов И.И.', '+7(919) 456-37-93', '4567 4567 4567 4567'),

('Петров П.П.', '+7(567) 345-29-47', '8765 5674 3212 0987'),

('Сидоров С.С.', '+7(904) 567-34-92', '4567 3456 7894 3090'),

('Попов П.П.', '+7(845) 637-76-46', '0987 6543 2112 3245'),

('Попов П.П.', '+7(465) 376-54-78', '1234 5678 9012 3456'),

('Кузнецов К.К.', '+7(915) 674-87-32', '4567 3456 7543 3246'),

('Смирнов С.С.', '+7(864) 567-09-09', NULL),

('Егоров Е.Е.', '+7(905) 678-34-23', '9438 6729 8769 5879'),

('Федоров Ф.Ф.', '+7(465) 345-88-77', '8945 7489 5678 5685'),

('Семенов С.С.', '+7(804) 536-89-37', '0978 6560 9856 8754'),

('Козлов К.К.', '+7(809) 537-89-80', '4376 8437 8634 8794'),

('Смирнов Г.Г.', '+7(809) 666-66-66', NULL)

GO

INSERT INTO Тариф VALUES

('Бустер',

'Тариф для поездки с ребенком от 7 до 12 лет.

Внимание, ребенок в комплект не входит!',

100),

('Детское кресло',

'Тариф для поездки с ребенком младше семи лет.

В машине присутствуют детское кресло и беруши,

стоимость которых входит в стоимость тарифа.',

200),

('Крупногабаритный багаж',

'Если вам вдруг захотелось съездить на такси

в театр, прихватив с собой из дома газовую плиту

или раковину - этот тариф для вас.',

300),

('Малогабаритный багаж',

'Если вы хотите проехаться на такси с чемоданом,

то вам придется доплатить водителю за открывание

и закрывание багажника, выбрав это тариф.',

50),

('Обычный',

NULL,

0),

('Перевозка животных',

'Тариф для перевозки животных в такси. Перевозка

носорогов и пассажиров, находящихся в состоянии

алкогольного опьянения, в тариф не входит.',

150)

GO

INSERT INTO Поездка VALUES

(2,'Бизнес', 'Обычный', 'ул. Голубая 56', '2022-09-09', '07:45', 7),

(2,'Комфорт', 'Обычный', 'ул. Голубая 56', '2022-09-10', '07:30', 8),

(2,'Премиум', 'Обычный', 'ул. Голубая 56', '2022-09-11', '07:39', 7),

(2,'Эконом', 'Обычный', 'ул. Голубая 56', '2022-09-12', '07:40', 7),

(8,'Бизнес', 'Обычный', 'ул. Голубая 56', '2022-09-13', '07:40', 7),

(2,'Бизнес', 'Обычный', 'ул. Голубая 56', '2022-09-14', '07:40', 7),

(2,'Бизнес', 'Обычный', 'ул. Голубая 56', '2022-09-15', '07:40', 7),

(2,'Люкс', 'Крупногабаритный багаж', 'ул. Синяя 46', '2022-10-31', '14:30', 47),

(3,'Премиум', 'Обычный', 'ул. Красная 17', '2022-10-08', '14:25', 5),

(4,'Бизнес', 'Перевозка животных', 'ул. Оранжевая 30', '2022-10-10', '15:30', 3),

(5,'Эконом', 'Бустер', 'ул. Белая 10', '2022-10-20', '13:12', 1),

(6,'Эконом', 'Обычный', 'ул. Розовая 5', '2022-10-31', '17:48', 41),

(7,'Бизнес', 'Детское кресло', 'ул. Синяя 25', '2022-10-16', '23:17', 3),

(8,'Эконом', 'Крупногабаритный багаж', 'ул. Фиолетовая 22','2022-10-31', '08:37', 21),

(9,'Премиум', 'Бустер', 'ул. Белая 17', '2022-10-20', '18:12', 3),

(10,'Люкс', 'Перевозка животных', 'ул. Черная 35', '2022-10-23', '16:10', 0.9),

(11,'Премиум', 'Обычный', 'ул. Красная 33', '2022-10-05', '11:57', 2),

(1,'Люкс', 'Перевозка животных', 'ул. Зелёная 33', '2022-09-29', '20:46', 4),

(1,'Эконом', 'Бустер', 'ул. Зелёная 24', '2022-10-11', '20:00', 2),

(1,'Эконом', 'Детское кресло', 'ул. Фиолетовая 32', '2022-10-31', '21:45', 0.9),

(9,'Комфорт', 'Крупногабаритный багаж', 'ул. Жёлтая 77', '2022-09-09', '16:39', 40)

GO

В результате выполнения кода в БД создаются 4 таблицы: Клиент (рисунок 3), Автомобиль (рисунок 4), Тариф (рисунок 5) и Поездка (рисунок 6).

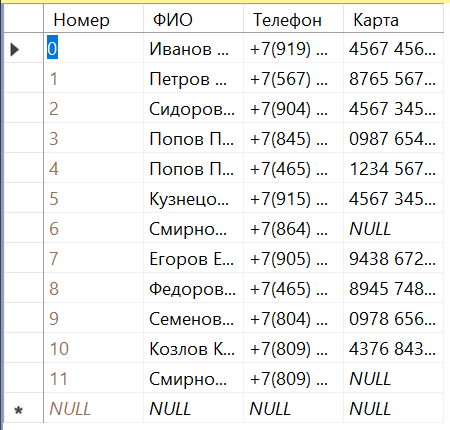


Рисунок 3 – Содержимое таблицы «Клиент»

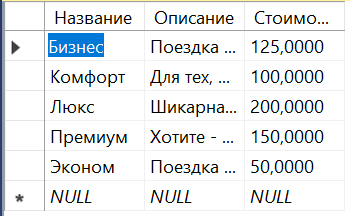


Рисунок 4 – Содержимое таблицы «Автомобиль»

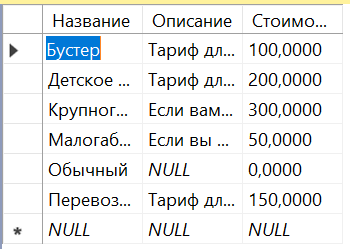


Рисунок 5 – Содержимое таблицы «Тариф»

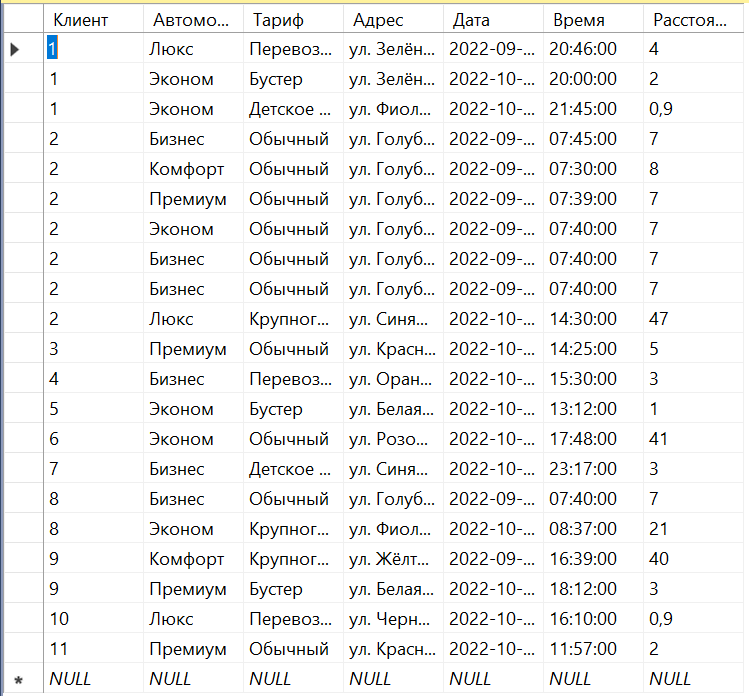


Рисунок 6 – Содержимое таблицы «Поездка»

# 3 Разработка основных операций реляционной алгебры

Составим запросы для базы данных «служба такси» и реализуем их на языке реляционной алгебры.

1. Вывести номера клиентов, стоимость поездок которых превысила 2000 рублей.

A = Автомобиль RENAME Название AS Автомобиль

B = Тариф RENAME Название AS Тариф

C = (Поездка JOIN A) [Клиент, Стоимость, Тариф, Расстояние]

D = (С JOIN В) [Клиент, Стоимость, Доплата, Расстояние]

R = (D WHERE Стоимость \* Расстояние + Доплата > 2000) [Клиент]

1. Вывести названия тарифов, которые ещё никто не использовал в поездках.

A = (Тариф [Название]) RENAME Название AS Тариф

В = Поездка [Тариф]

R = A EXCEPT B

1. Вывести информацию о поездках, исключив дату, время и расстояние, а также добавив к этой информации имена клиентов и номера их телефонов.

A = Клиент RENAME Номер AS НомерКлиента

B = Поездка TIMES A WHERE Клиент=НомерКлиента

R = B [Клиент, ФИО, Телефон, Автомобиль, Тариф, Адрес]

1. Вывести информацию о поездках, расстояние которых меньше 1 км или больше 20 км.

A = Поездка WHERE Расстояние < 1

B = Поездка WHERE Расстояние > 20

R = A UNION B

1. Вывести номера клиентов, бравших автомобили со стоимостью меньше 150, но больше 50 рублей за километр.

A = Автомобиль RENAME Название AS Автомобиль

B = Поездка JOIN А

С = (В WHERE Стоимость > 50) [Клиент]

D = (В WHERE Стоимость < 150) [Клиент]

R = С INTERSECT В

1. Вывести информацию о клиентах, попробовавших все виды автомобилей в поездках.

A = Поездка [Клиент, Автомобиль]

В = (Автомобиль RENAME Название AS Автомобиль) [Автомобиль]

C = A DIVIDE BY B

D = Клиент RENAME Номер AS Клиент

R = С JOIN D

1. Вывести ФИО и номера телефонов клиентов, попробовавших тип автомобиля «премиум», но не попробовавших тип автомобиля «люкс»

A = (Поездка WHERE Автомобиль = «Премиум») [Клиент]

B = (Поездка WHERE Автомобиль = «Люкс») [Клиент]

C = A EXCEPT B

D = Клиент RENAME Номер AS Клиент

R = C JOIN D

1. Вывести адрес, дату, время и расстояние поездок с типом автомобиля «эконом»

R = (Поездка WHERE Автомобиль = «Эконом») [Адрес, Дата, Время, Расстояние]

1. Вывести номера клиентов, бравших только поездки с типом тарифа «детское кресло»

A = (Поездка WHERE Тариф = «Детское кресло») [Клиент]

B = (Поездка WHERE Тариф ≠ «Детское кресло») [Клиент]

R = A EXCEPT B

1. Вывести типы и описания автомобилей, которые использовались в поездках на расстояние больше 20 км.

A = (Поездка WHERE Расстояние > 20) [Автомобиль]

R = (A TIMES Автомобиль WHERE Автомобиль = Название) [Название, Описание]

1. Вывести информацию о клиентах, выбравших в поездке тип автомобиля «люкс» вместе с типом тарифа «перевозка животных» и при этом проехавших на такси расстояние меньше километра.

A = ((Автомобиль WHERE Название = «Люкс») [Название]) RENAME Название AS Автомобиль

B = ((Тариф WHERE Название = «Перевозка животных») [Название]) RENAME Название AS Тариф

C = A TIMES B

D = Поездка [Клиент, Автомобиль, Тариф, Расстояние]

E = (D DIVIDE BY C) WHERE Расстояние < 1

F = Клиент RENAME Номер AS Клиент

R = E JOIN F

1. Добавить к информации о поездках описание типов автомобилей и тарифов.

A = (Автомобиль RENAME Название, Описание AS Автомобиль, ОписаниеАвтомобиля) [Автомобиль, ОписаниеАвтомобиля]

В = (Тариф RENAME Название, Описание AS Тариф, ОписаниеТарифа) [Тариф, ОписаниеТарифа]

R = (Поездка JOIN A) JOIN B

1. Вывести полную информацию о клиентах, которые не пробовали поездку ни «экономом», ни «комфортом».

A = (Поездка WHERE Автомобиль = «Эконом») [Клиент]

В = (Поездка WHERE Автомобиль = «Комфорт») [Клиент]

С = A INTERSECT B

D = Клиент RENAME Номер AS Клиент

E = (D [Клиент]) EXCEPT C

R = E JOIN D

1. Вывести время поездок и типы автомобилей, которые брали 31 октября 2022 года.

A = Поездка WHERE Дата = 31.10.2022

R = A [Время, Автомобиль]

1. Вывести информацию о клиентах, бравших тип автомобиля «бизнес».

A = Поездка WHERE Автомобиль = «Бизнес»

B = (A [Клиент]) RENAME Клиент AS Номер

R = B JOIN Клиент

1. Вывести названия тарифов, поездкам по которым соответствовала стоимость за километр меньше 100 рублей и расстояние больше 500 километров.

A = Автомобиль RENAME Название AS Автомобиль

B = Поездка JOIN А

C = (A WHERE Расстояние > 500) [Тариф]

D = (B WHERE Стоимость < 100) [Тариф]

R = C INTERSECT D

# 4 Разработка запросов к БД для реализации информационных потребностей пользователя

1. Запросы с выбором строк по условию:

- запрос с использованием операторов сравнения (рисунок 7);

-- вывести названия автомобилей и тарифов поездок на расстояния больше 10-ти км

SELECT Автомобиль, Тариф

FROM Поездка

WHERE Расстояние > 10

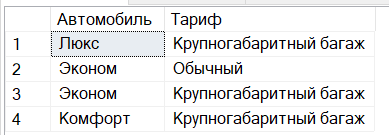


Рисунок 7 – Результат выполнения запроса

- запрос с использованием логических операторов AND, OR и NOT (рисунок 8);

-- вывести информацию о поездках, которые заказывались в сентябре 2022 года

SELECT \*

FROM Поездка

WHERE MONTH(Дата) = 9 AND YEAR(Дата) = 2022

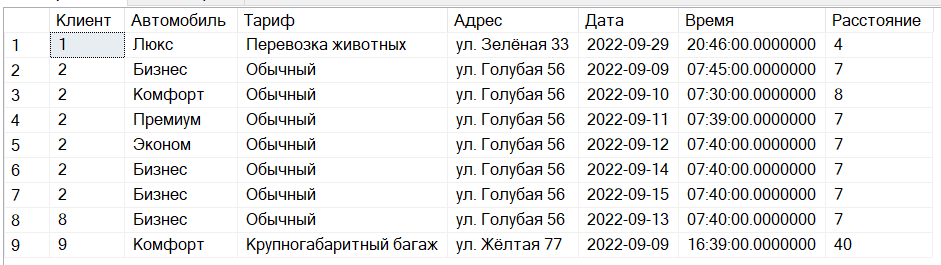


Рисунок 8 – Результат выполнения запроса

- запрос на использование комбинации логических операторов (рисунок 9);

-- вывести тарифы поездок, заказанных 31 или 1 октября любого года, расстояние которых не превышало 5 км

SELECT Тариф

FROM Поездка

WHERE MONTH(Дата) = 10 AND (DAY(Дата) = 31 OR DAY(Дата) = 1) AND NOT (Расстояние > 5)

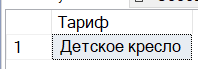


Рисунок 9 – Результат выполнения запроса

- запрос на использование выражений над столбцами (рисунок 10);

-- рассчитать стоимость всех поездок

SELECT Поездка.\*, Расстояние \* Автомобиль.Стоимость + Тариф.Стоимость AS [Стоимость поездки]

FROM Поездка JOIN Автомобиль ON Поездка.Автомобиль = Автомобиль.Название

JOIN Тариф ON Поездка.Тариф = Тариф.Название

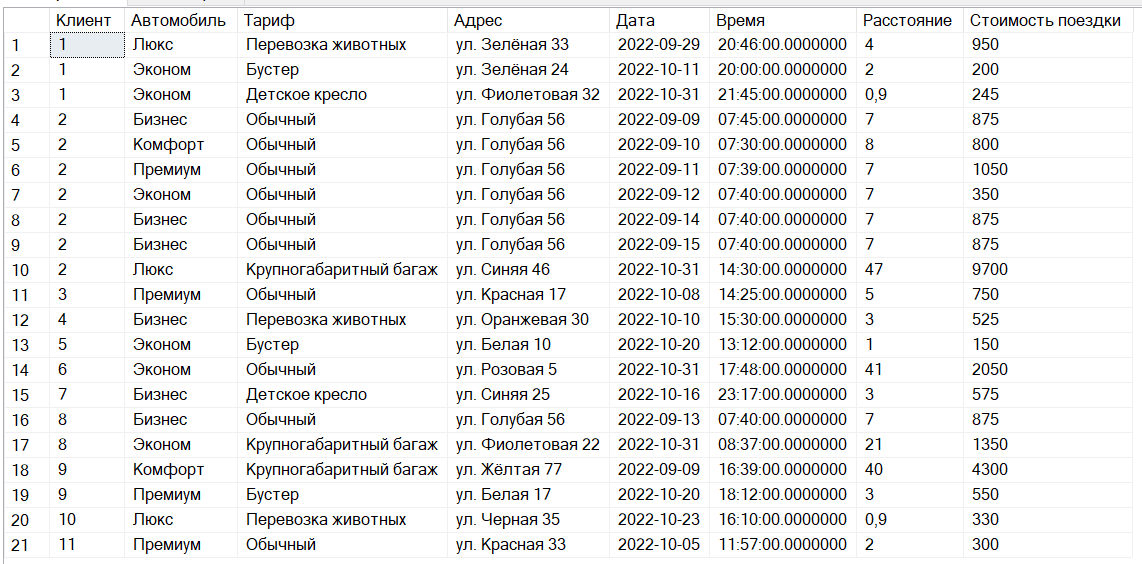


Рисунок 10 – Результат выполнения запроса

- запрос с проверкой на принадлежность множеству (рисунок 11);

-- вывести информацию о тарифах "Бустер" и "Обычный"

SELECT \*

FROM Тариф

WHERE Название IN ('Бустер','Обычный')

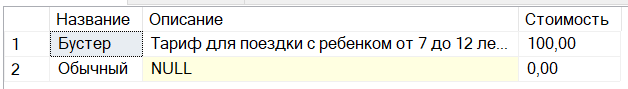


Рисунок 11 – Результат выполнения запроса

- запрос с проверкой на принадлежность диапазону значений (рисунок 12);

-- вывести названия автомобилей стоимостью не меньше 150 и не больше 200 рублей

SELECT Название

FROM Автомобиль

WHERE Стоимость BETWEEN 150 AND 200

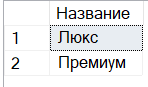


Рисунок 12 – Результат выполнения запроса

- запрос с проверкой на соответствие шаблону (рисунок 13);

-- вывести информацию о клиентах, фамилии которых начинаются с букв "А"-"И"

SELECT \*

FROM Клиент

WHERE ФИО LIKE '[А-И]%'

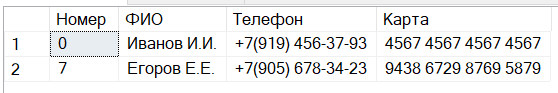


Рисунок 13 – Результат выполнения запроса

- запрос с проверкой на неопределенное значение (рисунок 14).

-- вывести информацию об клиентах, которые не указали номер своей карты

SELECT \*

FROM Клиент

WHERE Карта IS NULL

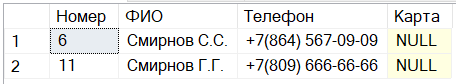


Рисунок 14 – Результат выполнения запроса

2. Запросы на теоретико-множественные реляционные операции:

- запрос на объединение таблиц (рисунок 15);

-- вывести названия автомобилей стоимостью меньше 150 или больше 200 рублей

SELECT Название

FROM Автомобиль

WHERE Стоимость > 200

UNION

SELECT Название

FROM Автомобиль

WHERE Стоимость < 150

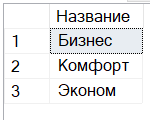


Рисунок 15 – Результат выполнения запроса

- запрос на пересечение таблиц (рисунок 16);

-- вывести названия автомобилей стоимостью не меньше 150 и не больше 200 рублей

SELECT Название

FROM Автомобиль

WHERE Стоимость <= 200

INTERSECT

SELECT Название

FROM Автомобиль

WHERE Стоимость >= 150

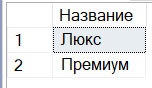


Рисунок 16 – Результат выполнения запроса

- запрос на вычитание таблиц (рисунок 17);

-- вывести названия тарифов, которые еще никто не пробовал в поездках

SELECT Название

FROM Тариф

EXCEPT

SELECT DISTINCT Тариф

FROM Поездка

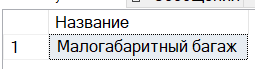


Рисунок 17 – Результат выполнения запроса

- запрос с использованием декартового произведения таблиц (рисунок 18);

-- вывести информацию о поездках вместе с фамилиями клиентов

SELECT Поездка.\*, Клиент.ФИО

FROM Поездка, Клиент

WHERE Поездка.Клиент = Клиент.Номер

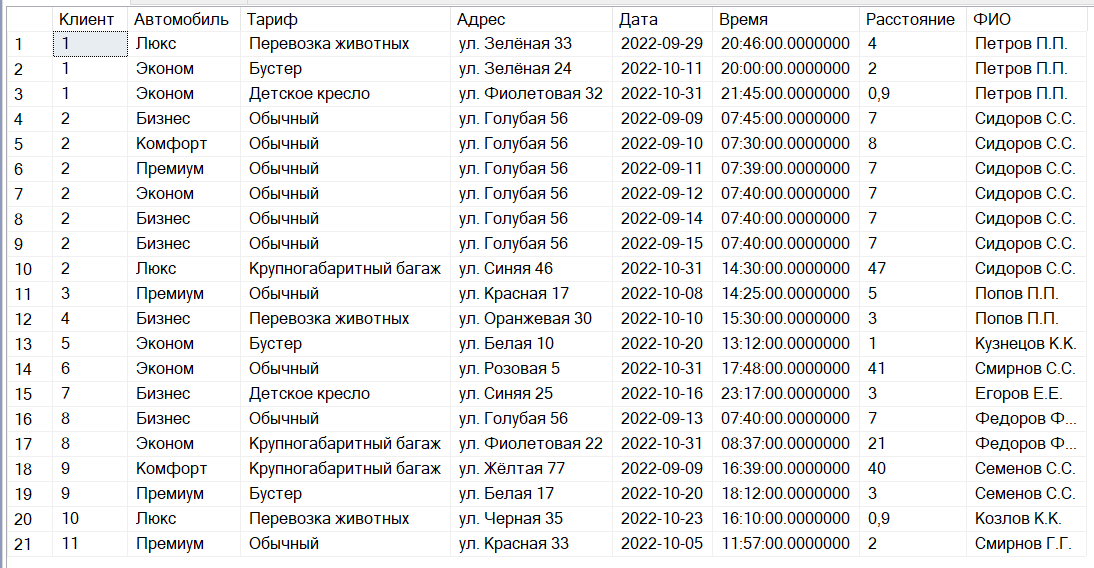


Рисунок 18 – Результат выполнения запроса

3. Многотабличные запросы:

- запрос с использованием соединения двух таблиц по равенству и условием отбора (рисунок 19);

-- вывести информацию о поездках по тарифу "обычный" вместе с фамилиями клиентов

SELECT Поездка.\*, Клиент.ФИО

FROM Поездка JOIN Клиент ON Поездка.Клиент = Клиент.Номер

WHERE Тариф = 'Обычный'

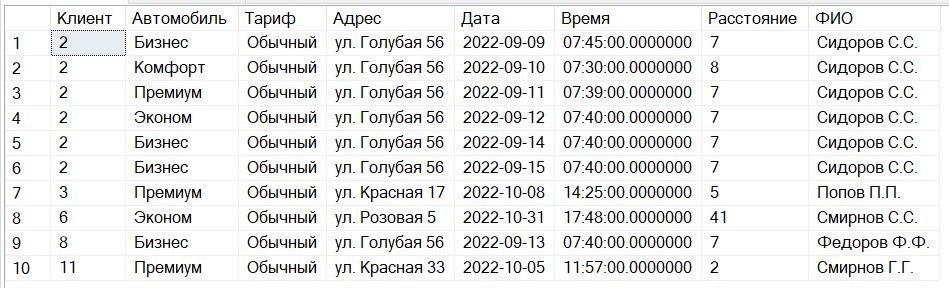


Рисунок 19 – Результат выполнения запроса

- запрос с использованием соединения трех таблиц по равенству и условием отбора (рисунок 20);

-- вывести информацию о поездках стоимостью больше 1000 рублей вместе с их стоимостью

SELECT Поездка.\*, Расстояние \* Автомобиль.Стоимость + Тариф.Стоимость AS [Стоимость поездки]

FROM Поездка JOIN Автомобиль ON Поездка.Автомобиль = Автомобиль.Название

JOIN Тариф ON Поездка.Тариф = Тариф.Название

WHERE Расстояние \* Автомобиль.Стоимость + Тариф.Стоимость > 1000

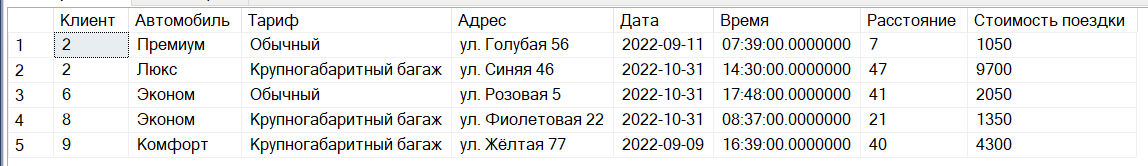


Рисунок 20 – Результат выполнения запроса

- запрос с использованием соединения по четырем таблицам (рисунок 21);

-- добавить к информации о поездках ФИО клиентов, стоимость тарифа и стоимость автомобиля

SELECT Поездка.\*, ФИО [ФИО клиента], Тариф.Стоимость [Стоимость тарифа], Автомобиль.Стоимость [Стоимость автомобиля]

FROM Поездка

JOIN Автомобиль ON Поездка.Автомобиль = Автомобиль.Название

JOIN Тариф ON Поездка.Тариф = Тариф.Название

JOIN Клиент ON Поездка.Клиент = Клиент.Номер



Рисунок 21 – Результат выполнения запроса

- запрос с использованием левого внешнего соединения (рисунок 22);

-- вывести количество поездок для всех видов тарифов

SELECT Название, [Кол-во поездок]

FROM Тариф LEFT JOIN (SELECT Тариф, COUNT(\*) [Кол-во поездок]

FROM Поездка

GROUP BY Тариф) [dop]

ON Тариф.Название = dop.Тариф

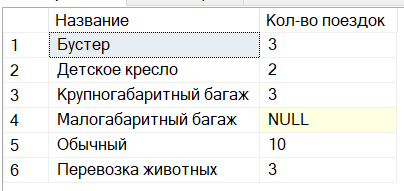


Рисунок 22 – Результат выполнения запроса

- запрос на использование правого внешнего соединения (рисунок 23);

-- вывести суммарное расстояние по поездкам для всех видов тарифов

SELECT Название, [Общее расстояние]

FROM (SELECT Тариф, SUM(Расстояние) [Общее расстояние]

FROM Поездка

GROUP BY Тариф) [dop] RIGHT JOIN Тариф ON dop.Тариф = Тариф.Название

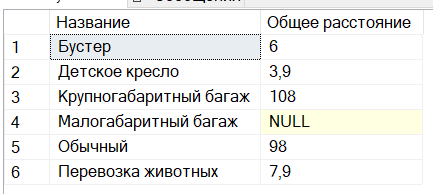


Рисунок 23 – Результат выполнения запроса

- запрос с использованием функции COUNT (рисунок 24);

-- посчитать для каждого клиента количество заказанных поездок

SELECT Клиент, COUNT(\*) [Кол-во заказов]

FROM Поездка

GROUP BY Клиент

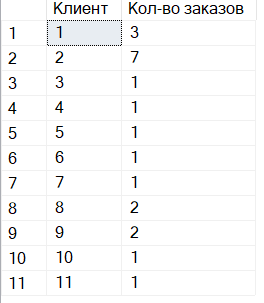


Рисунок 24 – Результат выполнения запроса

- запрос с использованием функции SUM (рисунок 25);

-- посчитать для каждого клиент суммарную стоимость всех сделанных заказов

SELECT Клиент, SUM(Расстояние \* Автомобиль.Стоимость + Тариф.Стоимость) [Общая стоимость поездок]

FROM Поездка JOIN Автомобиль ON Поездка.Автомобиль = Автомобиль.Название

JOIN Тариф ON Поездка.Тариф = Тариф.Название

GROUP BY Клиент

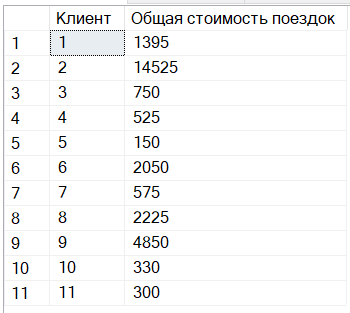


Рисунок 25 – Результат выполнения запроса

- 3 запроса с использованием временных функций (рисунки 26-28);

-- вывести информацию о поездках за первые десять дней октября 2022 года

SELECT \*

FROM Поездка

WHERE YEAR(Дата) = 2022 AND MONTH(Дата) = 10 AND DAY(Дата) BETWEEN 1 AND 10

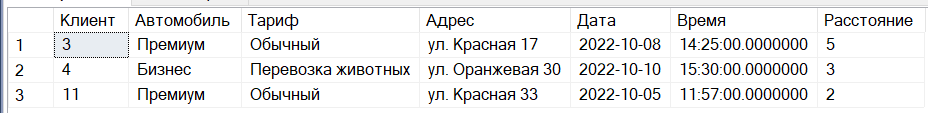


Рисунок 26 – Результат выполнения запроса

-- вывести информацию о заказанных не раньше восьми вечера поездках за текущий год

SELECT \*

FROM Поездка

WHERE YEAR(Дата) = YEAR(GETDATE()) AND Время >= '20:00:00'

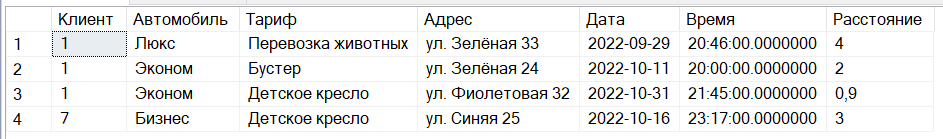


Рисунок 27 – Результат выполнения запроса

-- вывести информацию о поездках, совершенных не раньше 20 октября текущего года

SELECT \*

FROM Поездка

WHERE YEAR(Дата) = YEAR(GETDATE()) AND MONTH(Дата) >= 10 AND DAY(Дата) >= 20

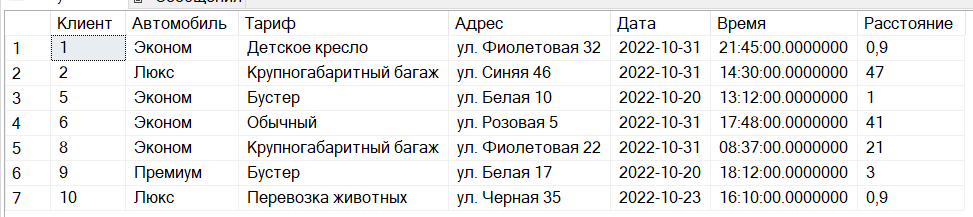


Рисунок 28 – Результат выполнения запроса

- запрос с использованием группировки по одному столбцу (рисунок 29);

-- определить наибольшую длину поездки для каждого клиента

SELECT Клиент, MAX(Расстояние) [Наибольшее расстояние]

FROM Поездка

GROUP BY Клиент

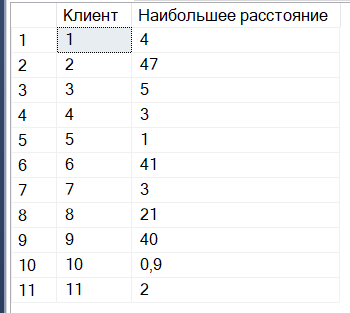


Рисунок 29 – Результат выполнения запроса

- запрос на использование группировки по нескольким столбцам (рисунок 30);

-- определить наибольшую длину поездки для каждого клиента и вывести ее вместе с информацией о клиенте

-- (информацию о клиентах, не заказывавших такси, не выводить)

SELECT Клиент.\*, MAX(Расстояние) [Наибольшее расстояние]

FROM Поездка LEFT JOIN Клиент ON Поездка.Клиент = Клиент.Номер

GROUP BY Клиент.Номер, Клиент.ФИО, Клиент.Телефон, Клиент.Карта

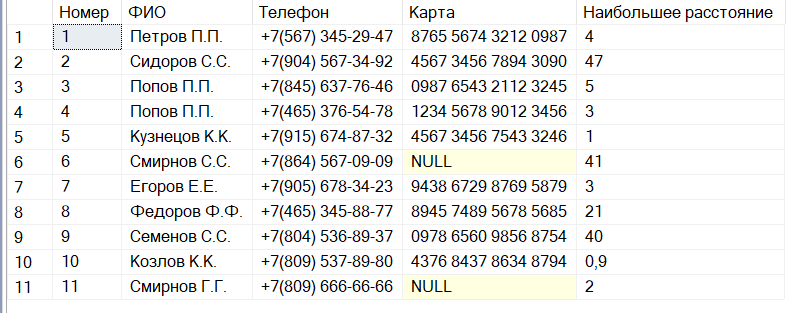


Рисунок 30 – Результат выполнения запроса

- запрос с использованием условия отбора групп HAVING (рисунок 31);

-- вывести информацию о клиентах, заказывавших такси менее 2 раз

SELECT \*

FROM Клиент JOIN (SELECT Клиент

FROM Поездка

GROUP BY Клиент

HAVING COUNT(\*) < 2) [dop]

ON Клиент.Номер = dop.Клиент

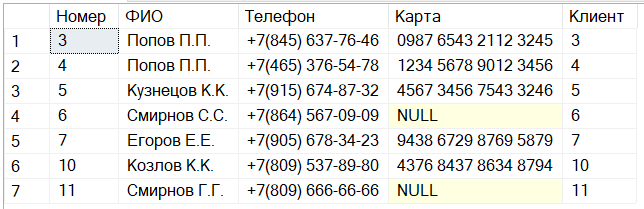


Рисунок 31 – Результат выполнения запроса

- запрос с использованием сортировки по столбцу (рисунок 32);

-- вывести информацию о клиентах, где их ФИО будут расположены по алфавиту

SELECT \*

FROM Клиент

ORDER BY ФИО ASC



Рисунок 32 – Результат выполнения запроса

4. Запросы на модификацию таблиц:

- запрос на добавление новых данных в таблицу (рисунок 33);

-- добавить в таблицу поездок новую запись о поездке 11-го клиента на автомобиле "Премиум" по тарифу "Обычный"

-- по адресу ул.Синяя 107 6 октября 2022 года в 12:14 на 2 километра,

-- и точно таккую же запись, но для 10-го клиента

INSERT INTO Поездка

VALUES (11, 'Премиум', 'Обычный', 'ул.Синяя 107', '2022-10-06', '12:14', 2),

(10, 'Премиум', 'Обычный', 'ул.Синяя 107', '2022-10-06', '12:14', 2)

GO

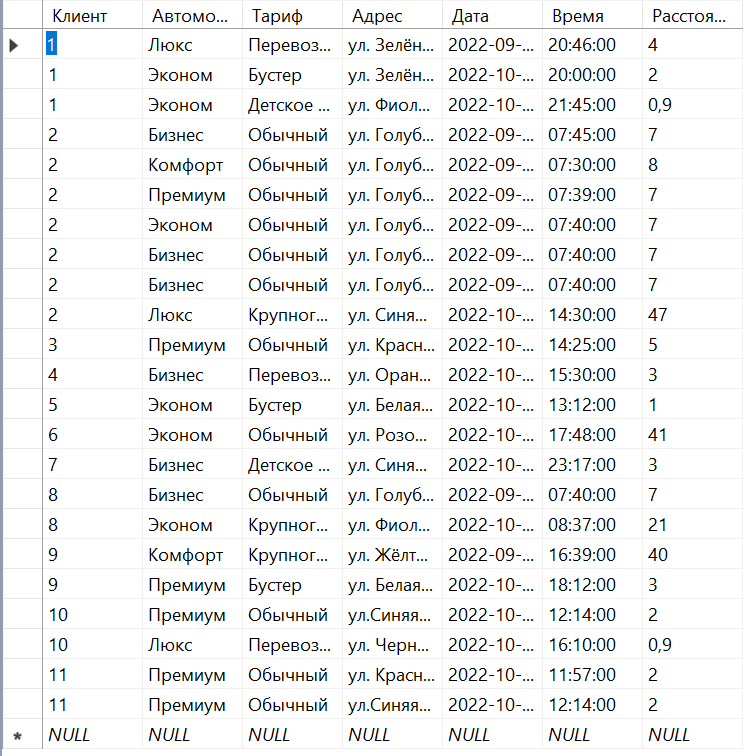


Рисунок 33 – Результат добавления

- запрос на добавление новых данных по результатам выполненного запроса (рисунок 34);

-- добавить в таблицу поездок запись для 11-го клиента по тарифу "Крупногабаритный багаж"

-- и с таким же автомобилем, адресом, датой, временем и расстоянием, как у самой длинной поездки / самых длинных поездок

INSERT INTO Поездка (Клиент, Тариф, Автомобиль, Адрес, Дата, Время, Расстояние)

SELECT \*

FROM (SELECT Номер

FROM Клиент

WHERE Номер = 11) [d1],

(SELECT Название

FROM Тариф

WHERE Название = 'Крупногабаритный багаж') [d2],

(SELECT Автомобиль, Адрес, Дата, Время, Расстояние

FROM Поездка

WHERE Расстояние = (SELECT MAX(Расстояние)

FROM Поездка)) [d3]

GO

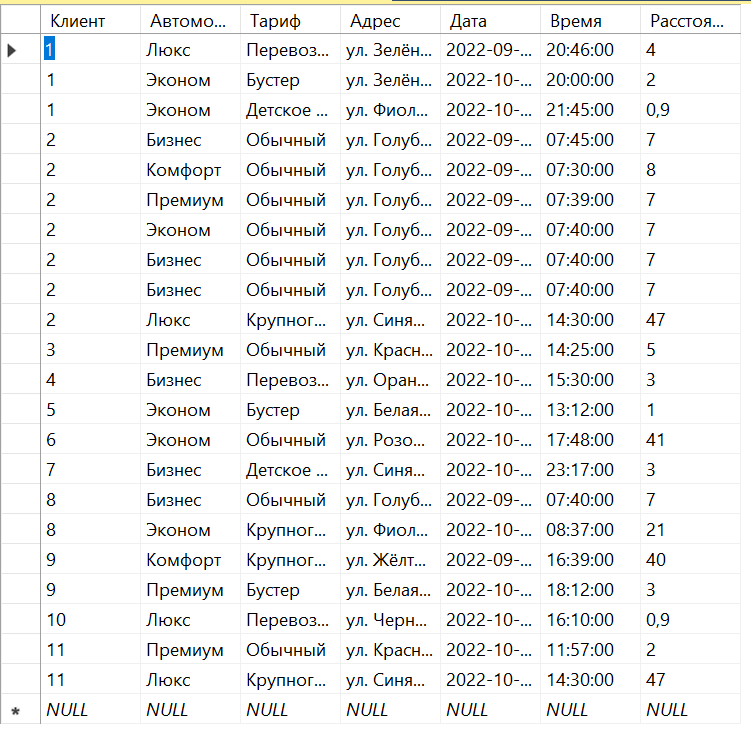


Рисунок 34 – Результат добавления

- запрос на обновление существующих данных в таблице (рисунок 35);

-- увеличить стоимость всех типов автомобилей на 25 рублей

UPDATE Автомобиль

SET Стоимость = Стоимость + 25

GO

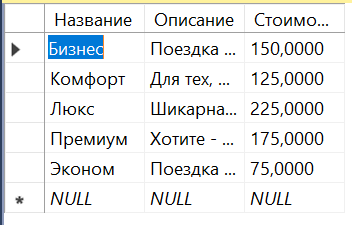


Рисунок 35 – Результат обновления

- запрос на обновление существующих данных по результатам подзапроса во фразе WHERE (рисунок 36);

-- увеличить стоимость самого дорогого автомобиля на 25 рублей

UPDATE Автомобиль

SET Стоимость = Стоимость + 25

FROM Автомобиль

WHERE Стоимость = (SELECT MAX(Стоимость)

FROM Автомобиль)

GO

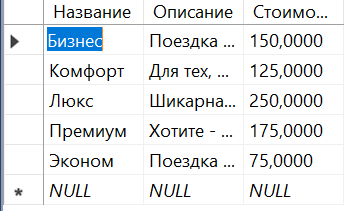


Рисунок 36 – Результат обновления

- запрос на удаление существующих данных (рисунок 37);

-- удалить все записи о поездках

DELETE

FROM Поездка

GO

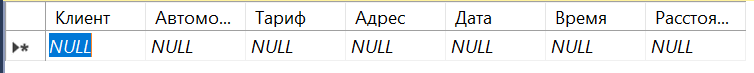


Рисунок 37 – Результат удаления

- запрос на удаление существующих данных по результатам подзапроса во фразе WHERE (рисунок 38);

-- удалить информацию о поездках на расстояние больше среднего расстояния

DELETE

FROM Поездка

WHERE Расстояние > (SELECT AVG(Расстояние)

FROM Поездка)

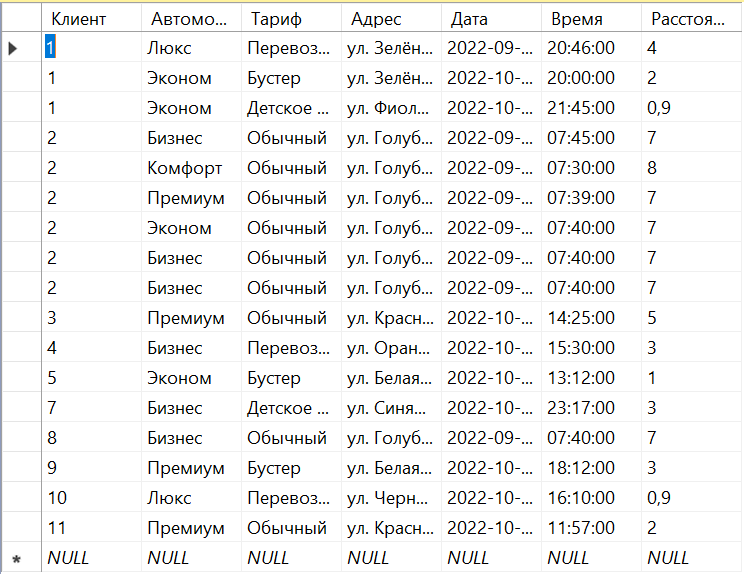
GO

Рисунок 38 – Результат удаления

- запрос на удаление существующих данных из одной таблицы на основе связанных с ней таблиц (рисунок 39).

-- удалить информацию о поездках с самым дорогим автомобилем и тарифом, у которого нет описания и расстоянием меньше километра

DELETE Поездка

FROM Поездка JOIN Автомобиль ON Поездка.Автомобиль = Автомобиль.Название

WHERE Тариф = 'Перевозка животных' AND Расстояние < 1

AND Автомобиль.Стоимость = (SELECT MAX(Стоимость) FROM Автомобиль)

GO

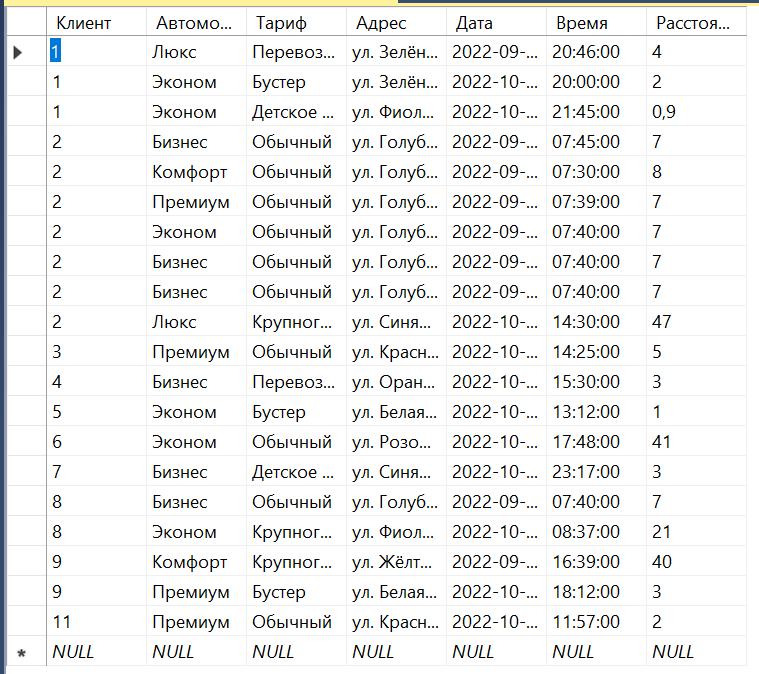


Рисунок 39 – Результат удаления

5. Многотабличные запросы с использованием подзапросов:

- 3 запроса с использованием операций сравнения (рисунки 40-42);

-- вывести информацию о поездках, расстояние которых больше среднего

SELECT \*

FROM Поездка

WHERE Расстояние > (SELECT AVG(Расстояние)

FROM Поездка)

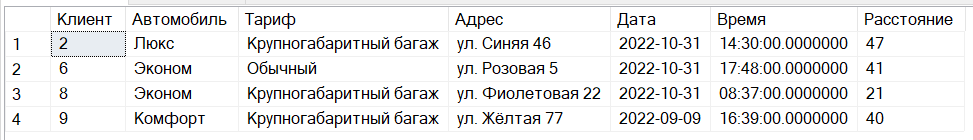


Рисунок 40 – Результат выполнения запроса

-- вывести описание самого дорогого тарифа

SELECT Описание

FROM Тариф

WHERE Стоимость = (SELECT MAX(Стоимость)

FROM Тариф)



Рисунок 41 – Результат выполнения запроса

-- определить номера клиентов, которые вызывали такси больше раз, чем 3-ий клиент

SELECT Клиент

FROM Поездка

GROUP BY Клиент

HAVING COUNT(\*) > (SELECT COUNT(\*)

FROM Поездка

WHERE Клиент = 3)

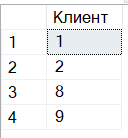


Рисунок 42 – Результат выполнения запроса

- запрос с использованием операции ANY (рисунок 43);

-- вывести информацию о клиентах, которые хотя бы раз вызывали такси

SELECT \*

FROM Клиент

WHERE Номер = ANY (SELECT Клиент

FROM Поездка)



Рисунок 43 – Результат выполнения запроса

- запрос с использованием операции ALL (рисунок 44);

-- вывести названия тарифов, которые ещё никто не пробовал в поездках

SELECT Название

FROM Тариф

WHERE Название <> ALL (SELECT DISTINCT Тариф

FROM Поездка)

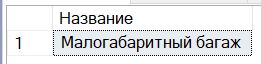


Рисунок 44 – Результат выполнения запроса

- запрос с использованием операции IN (рисунок 45);

-- вывести номера клиентов, которые не пробовали тариф "Обычный"

SELECT Номер

FROM Клиент

WHERE Номер NOT IN (SELECT Клиент

FROM Поездка

WHERE Тариф = 'Обычный')

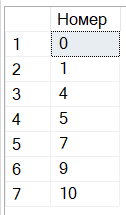


Рисунок 45 – Результат выполнения запроса

- запрос с использованием операции EXISTS (рисунок 46);

-- вывести названия тарифов, которые ещё никто не пробовал в поездках

SELECT Название

FROM Тариф

WHERE NOT EXISTS (SELECT \*

FROM Поездка

WHERE Поездка.Тариф = Тариф.Название)



Рисунок 46 – Результат выполнения запроса

- запрос с использованием двух вложенных друг в друга подзапросов (рисунок 47);

-- вывести информацию о клиенте(-ах), заказавшем(-их) такси наибольшее количество раз

SELECT \*

FROM Клиент

WHERE Номер IN (SELECT Клиент

FROM Поездка

GROUP BY Клиент

HAVING COUNT(\*) >= ALL (SELECT COUNT(\*)

FROM Поездка

GROUP BY Клиент))

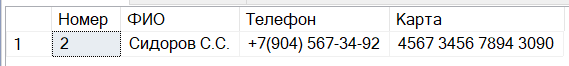


Рисунок 47 – Результат выполнения запроса

- запрос с использованием трех вложенных друг в друга подзапросов (рисунок 48);

-- вывести информацию о поездках с типом тарифа, который встречался среди заказов наибольшее количество раз

SELECT Тариф

FROM (SELECT Тариф, COUNT(\*) [Кол-во заказов]

FROM Поездка

GROUP BY Тариф) [d1]

WHERE [Кол-во заказов] = (SELECT MAX([Кол-во заказов])

FROM (SELECT COUNT(\*) [Кол-во заказов]

FROM Поездка

GROUP BY Тариф)[d2])

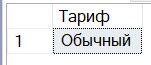


Рисунок 48 – Результат выполнения запроса

6. Запросы на изменение структуры существующей таблицы:

- запрос на добавление нового столбца к таблице (рисунок 49);

-- добавить к таблице поездок столбец с номером поездки

ALTER TABLE Поездка

ADD Номер int identity (1,1)

GO

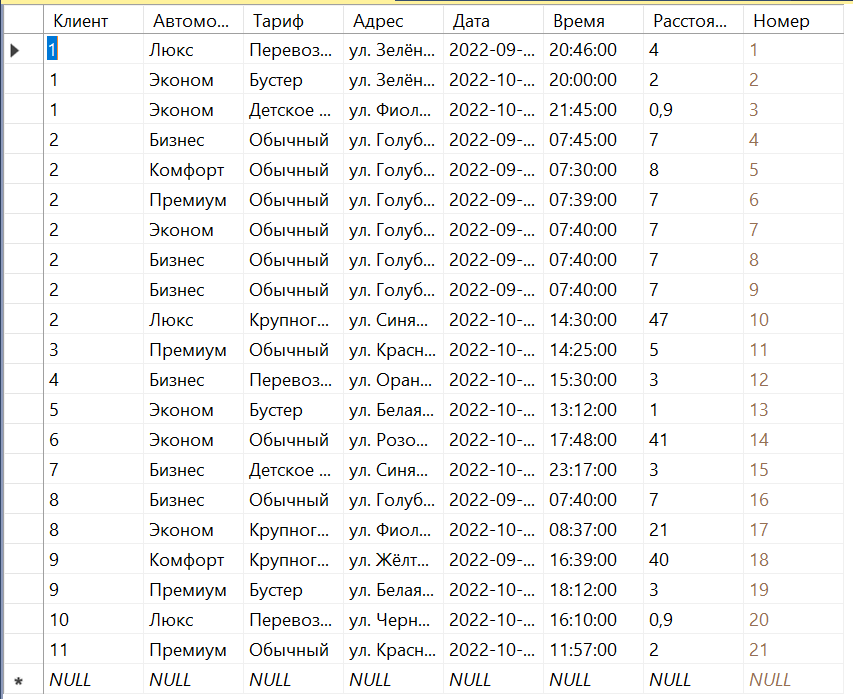


Рисунок 49 – Результат добавления столбца

- запрос на добавление нового ограничения к таблице;

-- сделать Номер в таблице Поездка уникальным

ALTER TABLE Поездка

ADD CONSTRAINT Поездка\_УникНомер unique (Номер)

GO

- запрос на удаление нового ограничения к таблице;

-- удалить ограничение уникальности для Номер в таблице Поездка

ALTER TABLE Поездка

DROP CONSTRAINT Поездка\_УникНомер

GO

- запрос на удаление нового столбца к таблице (рисунок 50);

-- удалить столбец Номер из таблицы Поездка

ALTER TABLE Поездка

DROP COLUMN Номер

GO

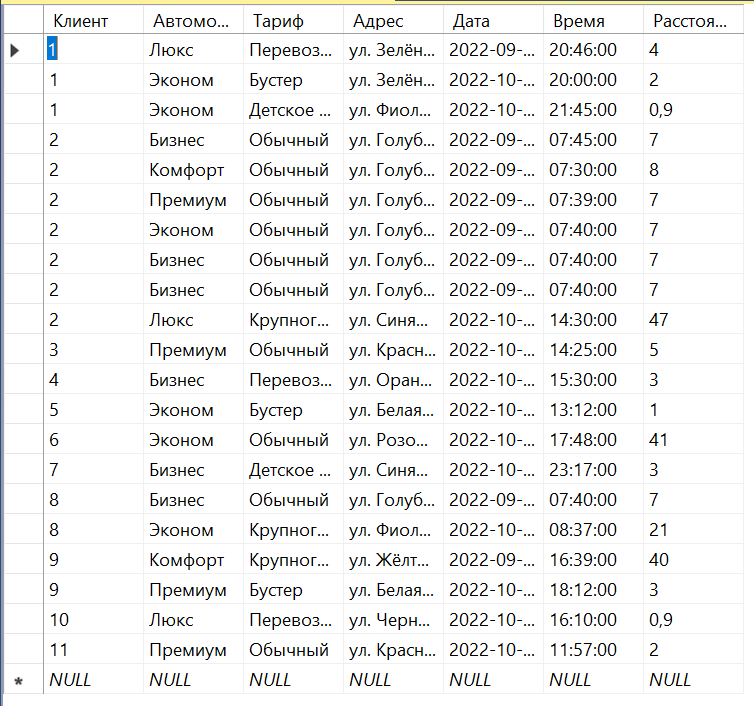


Рисунок 50 – Результат удаления столбца

- запрос на изменение типа данных в таблице (рисунок 51);

-- увеличить размер памяти, отведенный под столбец Описание таблицы Тариф

ALTER TABLE Тариф

ALTER COLUMN Описание varchar(2000)

GO

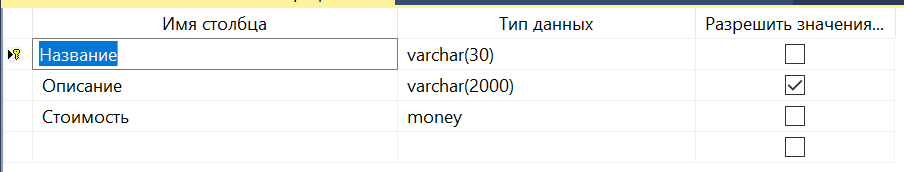


Рисунок 51 – Результат изменения типа данных

- запрос на добавление нового первичного ключа к таблице с сохранением целостности таблицы (рисунок 52, 53);

-- добавить в таблицу Поездка новый столбец - Номер, проставляемый БД автоматически

-- и сделать Номер первичным ключом

ALTER TABLE Поездка

ADD CONSTRAINT Unique\_Klient\_Date\_Time Unique (Клиент, Дата, Время)

GO

ALTER TABLE Поездка

DROP CONSTRAINT PK\_Поездка

GO

ALTER TABLE Поездка

ADD Номер int identity (1,1)

GO

ALTER TABLE Поездка

ADD CONSTRAINT PK\_Поездка primary key (Номер)

GO



Рисунок 52 – Результат добавления столбца

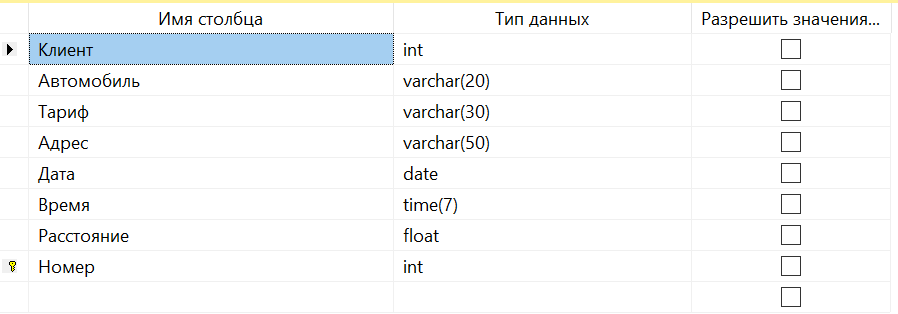


Рисунок 53 – Результат изменения первичного ключа

# Заключение

В ходе данной курсовой работы были приобретен крайне важный для разработчика навык – навык работы с базами данных. Он заключается не только в изучении языка реляционной алгебры и языка программирования SQL, но и в формировании рациональных, простых и правильных способов мышления для работы с большими объемами данных. Значение этого навыка трудно переоценить.

В результате работы была разработана БД «Служба такси». Кроме того, для неё были составлены запросы для реализации информационных потребностей пользователя – а это одна из самых главных вещей при создании БД, ведь она должна давать пользователю возможность работать с данными.

Если говорить о работе в целом, то опыт, полученный в ходе её выполнения, безусловно, пригодится в будущем. Это важная и нужная работа, знакомящая нас с основами работы с базами данных.

# Список литературы

1. Кренке Д. Теория и практика построения баз данных, - СПб.: Питер, 2005.
2. Черноусова А.М. Создание и использование баз данных, - ГОУ ОГУ, 2009. – 244 с.
3. Токмаков Г.П. Базы данных. Концепция баз данных, реляционная модель данных, языки SQL и XML. – УлГТУ, 2010. - 193 с.
4. Дунаев В.В. Базы данных. Язык SQL. – СПб.: БХВ-Петербург, 2006. – 288 с.: ил.
5. Р. Виейра. Программирование баз данных Microsoft SQL Server 2005. Базовый курс — М.: «Диалектика», 2007. —832 c.
6. Андон Ф., Резниченко В. Язык запросов SQL. Учебный курс. – СПб.: Питер; Киев: Издательская группа BHV, 2006. – 416 c.: ил.
7. Гринченко Н.Н., Громов А.Ю. Инструментальные средства поддержки проектирования баз данных: учеб. пособие / Рязан. гос. радиотехн. ун-т. - Рязань, 2015. 48 с.
8. Маркин А.В. Программирование на SQL : учеб. и практикум для бакалавриата и магистратуры. Ч.1 / Маркин Александр Васильевич. - М. : Юрайт, 2017. - 363с.
9. Маркин А.В. Программирование на SQL : учеб. и практикум для бакалавриата и магистратуры. Ч.2 / Маркин Александр Васильевич. - М. : Юрайт, 2017. - 293с.