**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра САПР**

ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

**по дисциплине «Базы данных»**

Тема: разработка баз данных

|  |  |  |
| --- | --- | --- |
| Студент гр. 7302 |  | Карманов Д.А. |
| Преподаватель |  | Новакова Н.Е. |

Санкт-Петербург

2019

**ЗАДАНИЕ**

**на курсовую работу**

|  |  |  |
| --- | --- | --- |
| Студент Карманов Д.А. | | |
| Группа 7302 | | |
| Тема работы: разработка баз данных | | |
| Исходные данные: представления о модели базы данных автобусного парка  Язык программирования SQL | | |
| Содержание пояснительной записки:  «Содержание», «Введение», «Формулировка задания», «Формализация задачи», «Спецификация базы данных», «Создание вторичного слоя», «Пример работы», «Разработка стратегии резервного копирования», «Итоговая диаграмма», «Вывод», «Список литературы» | | |
| Предполагаемый объем пояснительной записки:  Не менее 12 страниц(ы). | | |
| Дата выдачи задания: | | |
| Дата сдачи: | | |
| Дата защиты: | | |
| Студент |  | Карманов Д.А. |
| Преподаватель |  | Новакова Н.Е. |

**АННОТАЦИЯ**

Индивидуальное домашнее задание содержит в себе проектирование и реализацию базы данных заданной темы. В эту работу также входят такие темы как: создание триггеров, создание и работа с хранимыми процедурами и UDF функциями. Также не обошлось и без создания резервных копий и восстановления базы данных из них.

**SUMMARY**

Individual homework contains a database of a given topic. This work also identifies topics such as: creating triggers, creating and using procedures and UDF. It also could not do without creating backups and restoring the database from them.

Оглавление

[**ВВЕДЕНИЕ** 5](#_Toc28107343)

[**Формулировка задания** 6](#_Toc28107344)

[**Формализация задачи** 6](#_Toc28107345)

[**Создание базы данных** 9](#_Toc28107346)

[**Создание таблиц** 10](#_Toc28107347)

[**Спецификация базы данных** 11](#_Toc28107348)

[**Создание вторичного слоя и триггеров** 19](#_Toc28107349)

[**Создание процедур** 21](#_Toc28107350)

[**Создание функций** 22](#_Toc28107351)

[**Создание триггеров** 22](#_Toc28107352)

[**Пример работы** 23](#_Toc28107353)

[**Разработка стратегии резервного копирования** 25](#_Toc28107354)

[**Итоговая диаграмма** 27](#_Toc28107355)

[**Вывод** 27](#_Toc28107356)

[**СПИСОК ЛИТЕРАТУРЫ** 29](#_Toc28107357)

# **ВВЕДЕНИЕ**

Данное индивидуальное домашнее задание является объектом для закрепления теоретических знаний и практических навыков проектирования и реализации баз данных.

## **Формулировка задания**

Разработать и реализовать базу данных автобусного парка. Необходимо:

1) Продумать структуру базы данных и изобразить ее на UML-диаграмме;

2) Спроектировать базу данных исходя из диаграммы, полученной в пункте 1;

3) Написать объекты промежуточного слоя (хранимые процедуры, функции, триггеры и т.д.);

4) Заполнить данными полученную базу данных;

5) Разработать стратегию резервного копирования.

## **Формализация задачи**

Необходимо спроектировать базу данных, содержащую таблицы с информацией о работе автобусного парка.

Используемая предметная область - автобусный парк. Данные распределены в 8 таблиц:

1) Guides - содержит информацию об именах гидов. Первичный ключ выставлен по полю Id;

2) TourGroups - содержит информацию о туристических группах: об Id гида, который их ведет и количестве человек в группе. Первичный ключ выставлен по полю Id, внешний ключ выставлен по полю GuideId, создавая связь с таблицей Guides;

3) Orders - содержит информацию о действующих заказах: номере автобуса, Id гида и цене. Первичный ключ выставлен по полю Id, внешние ключи выставлены по полю GuideId таблицы TourGroups и по полю Num таблицы Buses;

4) Routes - содержит информацию о маршрутах: их номерах, точках отправления и прибытия, интервале (время, за которое можно проехать маршрут) и длинах маршрута. Первичный ключ выставлен по полю Num, внешние ключи по полям StartPoint и EndPoint, создавая связь с таблицей Points;

5) Points - содержит информацию об остановках: номере, названии, следующей остановке, времени пути до следующей остановки и длине пути до следующей остановки. Первичный ключ выставлен по полю Num.

6) Buses - содержит информацию об автобусах: их регистрационных номерах, типе, количестве сидячих мест, мест всего, номере маршрута, по которому эти автобусы ездят и доступности этого автобуса. Первичный ключ выставлен по полю RegNum. Внешний ключ выставлен по полю RouteNum, связываясь с таблицей Routes по полю Num.

7) Drivers - содержит информацию о водителях: их имена, персональные номера, опыт вождения, зарплата, дата рождения, номере автобуса, которые они водят. Первичный ключ выставлен по полю PerNum. Внешний ключ по полю BusNum и связан с таблицей Buses по полю RegNum.

8) History - содержит информацию о всех заказах: кто, когда, по какому маршруту и за сколько проехал. Первичный ключ выставлен по полю Id.

Ограничение первичными ключами используется для того, чтобы обеспечить уникальность данных в таблице и избежать неоднозначности в случае наличия в таблицах одинаковых записей.

Ограничение внешними ключами применяются для установки связи между таблицами и для обеспечения целостности базы данных.

Проектируемая база данных находится в третьей нормальной форме, так как в ней нет значений, которые не элементарны, не содержит неполных функциональных зависимостей, и в ней отсутствуют транзитивные функциональные зависимости. На Рис. 1 представлена связь между таблицами.



Рисунок 1. Диаграмма таблиц

## **Создание базы данных**

USE master

GO

CREATE DATABASE IDZ\_DB\_Karmanov

ON PRIMARY

( NAME = 'DB\_Karman',

FILENAME = 'D:\Ucheba\5sem\БД\IDZ\Я\primary filegroup\DB\_Karman.mdf',

SIZE = 4MB,

MAXSIZE = 10MB,

FILEGROWTH = 1MB),

FILEGROUP DB\_FG1\_Karman

(

NAME = 'DB\_FG1\_Karman1',

FILENAME = 'D:\Ucheba\5sem\БД\IDZ\Я\secondary filegroup\DB\_Karman1\_FG1.ndf',

SIZE = 1MB,

MAXSIZE = 10MB,

FILEGROWTH = 1MB

),

(

NAME = 'DB\_FG1\_Karman2',

FILENAME = 'D:\Ucheba\5sem\БД\IDZ\Я\secondary filegroup\DB\_Karman2\_FG1.ndf',

SIZE = 1MB,

MAXSIZE = 10MB,

FILEGROWTH = 1MB

)

LOG ON

( NAME = 'IDZ\_Karman\_log',

FILENAME = 'D:\Ucheba\5sem\БД\IDZ\Я\log filegroup\DB\_Karman.ldf',

SIZE = 1MB,

MAXSIZE = 10MB,

FILEGROWTH = 1MB);

GO

## **Создание таблиц**

USE IDZ\_DB\_Karmanov

GO

CREATE TABLE Points

(

Num INT PRIMARY KEY NOT NULL,

Name NVARCHAR(20) NULL,

NextPoint INT NULL,

DrivingTime INT NOT NULL CHECK (DrivingTime > 0),

LengthToNextPoint INT NULL CHECK (LengthToNextPoint IS NULL OR LengthToNextPoint >= 0)

)

ON DB\_FG1\_Karman

GO

CREATE TABLE Routes

(

Num INT NOT NULL PRIMARY KEY,

StartPoint INT NULL,

EndPoint INT NULL,

Interval INT NOT NULL,

Length INT CHECK (Length > 0)

CONSTRAINT FK\_Route\_to\_Point\_SP FOREIGN KEY (StartPoint) REFERENCES Points(Num) ON DELETE CASCADE,

CONSTRAINT FK\_Route\_to\_Point\_EP FOREIGN KEY (EndPoint) REFERENCES Points(Num) ON DELETE NO ACTION

)

ON DB\_FG1\_Karman

GO

CREATE TABLE Buses

(

RegNum INT NOT NULL PRIMARY KEY,

Type NVARCHAR(20) NOT NULL,

Seats INT NOT NULL CHECK(Seats >= 0 ),

Capacity INT NOT NULL CHECK(Capacity >= 10),

RouteNum INT NULL,

Availability BIT DEFAULT (0) NOT NULL,

CONSTRAINT FK\_Bus\_To\_Route FOREIGN KEY (RouteNum) REFERENCES Routes(Num) ON DELETE SET NULL

)

ON DB\_FG1\_Karman

GO

CREATE TABLE Drivers

(

PersNum INT PRIMARY KEY NOT NULL,

FullName NVARCHAR(20) NULL,

Experience INT NULL CHECK(Experience IS NULL OR Experience >= 0),

Salary MONEY NOT NULL CHECK (Salary > 0),

DateOfBirth DATE NOT NULL,

BusNum INT NULL,

CONSTRAINT FK\_Drivers\_to\_Buses FOREIGN KEY (BusNum) REFERENCES Buses(RegNum) ON DELETE SET NULL

)

ON DB\_FG1\_Karman

GO

## **Спецификация базы данных**

Таблица 1. Описание структуры таблицы Drivers

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Описание структуры таблицы БД** | | | Наименование таблицы БД:  **Водители автобусов** | | | | Имя таблицы:  Drivers | | | |
| Дата разработки:  15.12.2019 | | |
| Порядковый номер таблицы:  **1** | | |  | | | |
|  |  |  | |  | |  |  | |  |  |
|  | № п/п | Наименование поля | | Спецификация данных | | | | | |  |
|  | Имя поля | Тип данных | | Ключ | Ограничения целостности | |  |
|  | 1 | Персональный номер | | PersNum | INT | | P | NOT NULL | |  |
|  | 2 | Полное имя | | FullName | NVARCHAR(20) | |  | NULL | |  |
|  | 3 | Стаж | | Experience | INT | |  | CHECK(Experience IS NULL OR Experience > 0), NULL | |  |
|  | 4 | Зарплата | | Salary | MONEY | |  | CHECK(Salary > 0), NOT NULL | |  |
|  | 5 | Дата рождения | | DateOfBirth | DATETIME | |  | NOT NULL | |  |
|  | 6 | Номер автобуса | | BusNum | INT | | F | NULL | |  |
|  |  |  | |  |  | |  |  | |  |
|  |  |  | |  |  | |  |  | |  |

Таблица 2. Описание структуры таблицы Buses

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Описание структуры таблицы БД** | | | Наименование таблицы БД:  **Автобусы** | | | | Имя таблицы:  Buses | | | |
| Дата разработки:  15.12.2019 | | |
| Порядковый номер таблицы:  **2** | | |  | | | |
|  |  |  | |  | |  |  | |  |  |
|  | № п/п | Наименование поля | | Спецификация данных | | | | | |  |
|  | Имя поля | Тип данных | | Ключ | Ограничения целостности | |  |
|  | 1 | Регистрационный номер автобуса | | RegNum | INT | | P | NOT NULL | |  |
|  | 2 | Тип автобуса | | Type | NVARCHAR(20) | |  | NOT NULL | |  |
|  | 3 | Сидячие места | | Seats | INT | |  | NOT NULL, CHECK(Seats >= 0) | |  |
|  | 4 | Мест всего | | Capacity | INT | |  | NOT NULL, CHECK(Capacity >= 10) | |  |
|  | 5 | Маршрут следования | | RouteNum | INT | | F | NULL | |  |
|  | 6 | Доступность автобуса | | Availability | BIT | |  | NOT NULL | |  |
|  |  |  | |  |  | |  |  | |  |
|  |  |  | |  |  | |  |  | |  |

Таблица 3. Описание структуры таблицы Routes

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Описание структуры таблицы БД** | | | Наименование таблицы БД:  **Маршруты** | | | | Имя таблицы:  Routes | | | |
| Дата разработки:  15.12.2019 | | |
| Порядковый номер таблицы:  **3** | | |  | | | |
|  |  |  | |  | |  |  | |  |  |
|  | № п/п | Наименование поля | | Спецификация данных | | | | | |  |
|  | Имя поля | Тип данных | | Ключ | Ограничения целостности | |  |
|  | 1 | Номер маршрута | | Num | INT | | P | NOT NULL | |  |
|  | 2 | Пункт отправления | | StartPoint | INT | | F | NULL | |  |
|  | 3 | Пункт прибытия | | EndPoint | INT | | F | NULL | |  |
|  | 4 | Время пути | | Interval | INT | |  | NOT NULL | |  |
|  | 5 | Длина маршрута | | Length | INT | |  | CHECK(Length > 0) | |  |
|  |  |  | |  |  | |  |  | |  |
|  |  |  | |  |  | |  |  | |  |

Таблица 4. Описание структуры таблицы Points

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Описание структуры таблицы БД** | | | Наименование таблицы БД:  **Остановки** | | | | Имя таблицы:  Points | | | |
| Дата разработки:  15.12.2019 | | |
| Порядковый номер таблицы:  **4** | | |  | | | |
|  |  |  | |  | |  |  | |  |  |
|  | № п/п | Наименование поля | | Спецификация данных | | | | | |  |
|  | Имя поля | Тип данных | | Ключ | Ограничения целостности | |  |
|  | 1 | Номер остановки | | Num | INT | | P | NOT NULL | |  |
|  | 2 | Название остановки | | Name | NVARCHAR(20) | |  | NULL | |  |
|  | 3 | Следующая остановка | | NextPoint | INT | |  | NULL | |  |
|  | 4 | Время пути до следующей остановки | | DrivingTime | INT | |  | NOT NULL, CHECK (DrivingTime>0) | |  |
|  | 5 | Длина пути до следующей остановки | | LengthToNextPoint | INT | |  | NULL, CHECK(LengthToNextPoint IS NULL OR LengthToNextPoint >= 0) | |  |
|  |  |  | |  |  | |  |  | |  |
|  |  |  | |  |  | |  |  | |  |

Таблица 5. Описание структуры таблицы Guides

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Описание структуры таблицы БД** | | | Наименование таблицы БД:  **Гиды** | | | | Имя таблицы:  Guides | | | |
| Дата разработки:  15.12.2019 | | |
| Порядковый номер таблицы:  **5** | | |  | | | |
|  |  |  | |  | |  |  | |  |  |
|  | № п/п | Наименование поля | | Спецификация данных | | | | | |  |
|  | Имя поля | Тип данных | | Ключ | Ограничения целостности | |  |
|  | 1 | Идентификатор | | Id | INT | | P | NOT NULL | |  |
|  | 2 | Имя гида | | Name | NVARCHAR(20) | |  | NOT NULL | |  |
|  |  |  | |  |  | |  |  | |  |
|  |  |  | |  |  | |  |  | |  |

Таблица 6. Описание структуры таблицы TourGroups

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Описание структуры таблицы БД** | | | Наименование таблицы БД:  **Туристические группы** | | | | Имя таблицы:  TourGroups | | | |
| Дата разработки:  15.12.2019 | | |
| Порядковый номер таблицы:  **6** | | |  | | | |
|  |  |  | |  | |  |  | |  |  |
|  | № п/п | Наименование поля | | Спецификация данных | | | | | |  |
|  | Имя поля | Тип данных | | Ключ | Ограничения целостности | |  |
|  | 1 | Номер гида | | GuideId | INT | | P F | NOT NULL | |  |
|  | 2 | Количество людей в группе | | PeopleNum | INT | |  | NOT NULL, CHECK(PeopleNum > 0) | |  |
|  |  |  | |  |  | |  |  | |  |
|  |  |  | |  |  | |  |  | |  |

Таблица 7. Описание структуры таблицы Orders

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Описание структуры таблицы БД** | | | Наименование таблицы БД:  **Заказы** | | | | Имя таблицы:  Orders | | | |
| Дата разработки:  15.12.2019 | | |
| Порядковый номер таблицы:  **7** | | |  | | | |
|  |  |  | |  | |  |  | |  |  |
|  | № п/п | Наименование поля | | Спецификация данных | | | | | |  |
|  | Имя поля | Тип данных | | Ключ | Ограничения целостности | |  |
|  | 1 | Идентификатор | | Id | INT | | P | NOT NULL | |  |
|  | 2 | Номер автобуса | | BusNum | NVARCHAR(20) | | F | NOT NULL | |  |
|  | 3 | Идентификатор гида | | GuideId | INT | | F | NOT NULL | |  |
|  | 4 | Цена за заказ | | Price | INT | |  | NOT NULL | |  |
|  |  |  | |  |  | |  |  | |  |
|  |  |  | |  |  | |  |  | |  |

Таблица 8. Описание структуры таблицы History

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Описание структуры таблицы БД** | | | Наименование таблицы БД:  **История** | | | | Имя таблицы:  History | | | |
| Дата разработки:  15.12.2019 | | |
| Порядковый номер таблицы:  **8** | | |  | | | |
|  |  |  | |  | |  |  | |  |  |
|  | № п/п | Наименование поля | | Спецификация данных | | | | | |  |
|  | Имя поля | Тип данных | | Ключ | Ограничения целостности | |  |
|  | 1 | Идентификатор | | Id | INT | | P | NOT NULL | |  |
|  | 2 | Имя гида | | GuidName | NVARCHAR(20) | |  | NOT NULL | |  |
|  | 3 | Количество людей | | PeopleNum | INT | |  | NOT NULL | |  |
|  | 4 | Номер автобуса | | BusNum | INT | |  | NOT NULL | |  |
|  | 5 | Цена за заказ | | Price | MONEY | |  | NOT NULL | |  |
|  | 6 | Дата отправления | | Date | DATETIME | |  | NOT NULL | |  |
|  |  |  | |  |  | |  |  | |  |
|  |  |  | |  |  | |  |  | |  |

## **Создание вторичного слоя и триггеров**

Для некоторого функционала в базе данных присутствуют четыре процедуры, две функции и два триггера.

*Процедуры:*

1. AddRoute. Принимает на вход номер маршрута, пункт отправления и пункт прибытия. По входным параметрам либо создает новую запись маршрута, рассчитывая длину пути и время пути, либо выдает сообщение о том, что маршрут проложить невозможно, если нет такого пути, либо запись с этим маршрутом уже есть.

2. AddBus. Принимает на вход регистрационный номер автобуса, тип, количество сидячих мест, мест в общем и номер маршрута, по которому этот автобус следует. По входным данным либо создает новую запись автобусы, либо сообщает о невозможности добавить данный автобус, если запись с таким номером уже есть, либо записи с таким маршрутом не существует.

3. AddDriver. Принимает на вход персональный номер водителя, полное имя, стаж, дату рождения и номер автобуса для этого водителя. По входным данным либо создает новую запись водителя, рассчитывая ЗП в зависимости от стажа, либо выдает сообщение о невозможности добавить такую запись, если либо водитель с таким номером уже существует, либо такого автобуса не существует, либо водитель слишком молод (допускаются только те, кто старше 17ти лет).

4. AddOrder. Принимает на вход Id гида и номер маршрута. По этим параметрам находит доступный автобус с достаточным количеством мест и нужным маршрутом. Если таков нашелся, добавляет заказ, иначе нет.

*Функции:*

1. GetBusAmountByType(@Type NVARCHAR(20)). Принимает на вход тип автобуса и возвращает таблицу с количеством автобусов этого типа.

2. GetPointsAmount(@RouteNum INT). Возвращает количество остановок на маршруте (не считая отправной).

*Триггеры:*

1. Orders\_INSERTED. Срабатывает при добавлении в таблицу заказов строки и добавляет строку в таблицу History.

2. Orders\_DELETED. Срабатывает при удалении из таблицы заказов и снимает метку доступности с автобуса, занятого этим заказом.

## **Создание процедур**

USE IDZ\_DB\_Karmanov

GO

CREATE PROCEDURE AddBus --Добавляет автобус в таблицу автобусов

@Num INT, @Type NVARCHAR(20), @Seats INT, @Capacity INT, @RouteNum INT

AS

BEGIN

IF(EXISTS(SELECT Num FROM Routes WHERE Num = @RouteNum))

BEGIN

IF(NOT EXISTS(SELECT RegNum FROM Buses WHERE RegNum = @Num))

BEGIN

INSERT INTO Buses VALUES (@Num, @Type, @Seats, @Capacity, @RouteNum, DEFAULT);

END

ELSE PRINT 'Автобус с таким номером уже существует'

END

ELSE PRINT 'Невозможно добавить автобус на несуществующий маршрут'

END

GO

CREATE PROCEDURE AddDriver --Добавляет водителя в таблицу водителей

@PersNum INT, @FullName NVARCHAR(20), @Experience INT, @DateOfBirth DATE, @BusNum INT

AS

BEGIN

DECLARE @Salary MONEY;

IF(@Experience > 0 AND @Experience <= 1)

SET @Salary = 15000;

IF(@Experience > 1 AND @Experience <= 3)

SET @Salary = 25000;

IF(@Experience > 3)

SET @Salary = 32000;

IF(EXISTS(SELECT PersNum FROM Drivers WHERE PersNum = @PersNum))

PRINT 'Водитель с таким персональным номером уже есть в базе'

ELSE

BEGIN

IF(NOT EXISTS(SELECT RegNum FROM Buses WHERE RegNum = @BusNum))

PRINT 'Невозможно привязать водителя к несуществующему автобусу'

ELSE

BEGIN

IF(ABS(DATEDIFF(YEAR, @DateOfBirth, GETDATE())) < 18)

PRINT 'Водитель должен быть старше 17 лет'

ELSE INSERT INTO Drivers VALUES (@PersNum, @FullName, @Experience, @Salary, @DateOfBirth, @BusNum);

END

END

END

GO

## **Создание функций**

CREATE FUNCTION GetBusAmountByType(@Type NVARCHAR(20)) --Возвращает в виде таблицы число автобусов заданного входным параметром типа

RETURNS TABLE

AS

RETURN (SELECT Type, COUNT(\*) AS Amount FROM Buses WHERE Type = @Type GROUP BY Type);

GO

CREATE FUNCTION GetPointsAmount(@RouteNum INT) --Возвращает количество остановок в маршруте

RETURNS INT

AS

BEGIN

DECLARE @Count INT; SET @Count = 0;

DECLARE @CurrentPoint INT; SET @CurrentPoint = (SELECT StartPoint FROM Routes WHERE Num = @RouteNum);

DECLARE @Timer INT;

WHILE(@CurrentPoint <> (SELECT EndPoint FROM Routes WHERE Num = @RouteNum))

BEGIN

SET @CurrentPoint = (SELECT NextPoint FROM Points WHERE Num = @CurrentPoint);

SET @Count = @Count + 1;

END

RETURN @Count;

END

GO

## **Создание триггеров**

USE IDZ\_DB\_Karmanov

GO

CREATE TRIGGER Orders\_INSERT

ON Orders

AFTER INSERT

AS

INSERT INTO History (GuidName, PeopleNum, BusNum, Price, Date)

SELECT Name, PeopleNum, inserted.BusNum, inserted.Price, GETDATE() FROM inserted

JOIN Guides ON GuideId = Guides.Id

JOIN TourGroups ON Guides.Id = TourGroups.GuideId

GO

CREATE TRIGGER Orders\_DELETE

ON Orders

AFTER DELETE

AS

UPDATE Buses SET Availability = 0 WHERE RegNum = (SELECT deleted.BusNum FROM deleted)

## **Пример работы**

Проверим работу процедуры AddRoute:

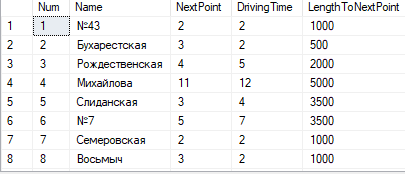


Рисунок 2. Таблица Points



Рисунок 3. Таблица Routes до создания маршрута

Добавим маршрут от точки 1 до точки 4



Рисунок 4. Таблица Routes после создания маршрута

Как видно на скриншоте выше, маршрут проложен, а путь и время рассчитаны.

Попробуем проложить несуществующий маршрут из точки 3 до точки 7, например

Результат:



Рисунок 5. Результат неудачной попытки прокладки маршрута

Проверим работу функции GetPointAmount. Запросим количество остановок на маршруте 1, их должно быть 3.

Результат:



Рисунок 6. Результат работы функции GetPointAmount

Как видно из рисунка выше, итоговый результат совпал с нашими предположениями.

Узнаем, сколько пассажирских автобусов у нас есть



Рисунок 7. Результат работы функции

На рисунке выше можно увидеть, сколько пассажирских автобусов записано в таблице автобусов.

Напоследок протестируем работу триггеров. Добавим три заказа, один из них удалим и удостоверимся, что автобус удаленного заказа стал доступен (метка 0), а в историю сохранились все заказы:

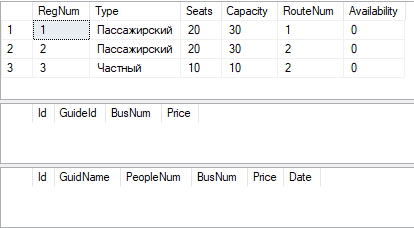


Рисунок 8. Состояние Buses, Orders и History до

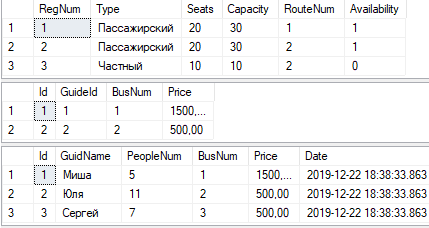


Рисунок 9. Состояние Buses, Orders и History после

Как видим на рисунке выше, ожидания оправдались.

## **Разработка стратегии резервного копирования**

Для проектируемой БД была выбрана стратегия резервного копирования базы данных и журнала транзакций. На это есть несколько причин: база данных магазина будет часто изменяться и пополняться новыми данными о заказах, следовательно, ее размеры будут быстро расти, что не позволит нам использовать стратегию полного резервного копирования базы данных, файлы которой имеют большие размеры.

Проверим, работает ли восстановление базы данных из резервной копии. Сымитируем потерю базы данных путем ее отключения:

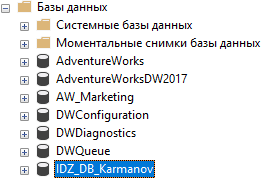


Рисунок 10. База данных на месте

И в таблице Points присутствуют следующие данные:

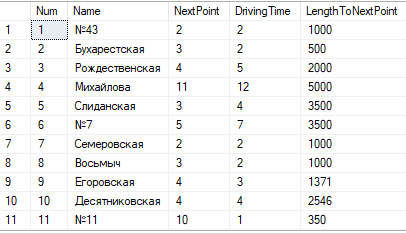


Рисунок 11. Данные таблицы Point до потери базы данных

Выполним создание резервной копии

USE master

GO

BACKUP DATABASE IDZ\_DB\_Karmanov

TO DISK = 'D:\Ucheba\5sem\БД\IDZ\Я\BackupFull.bak'

Затем отключим базу данных и попытаемся восстановить:



Рисунок 11. На наше действие что-то достать из этой базы данных, произошла ошибка, т.к. такой базы данных уже не существует



Рисунок 12. Восстановили базу данных

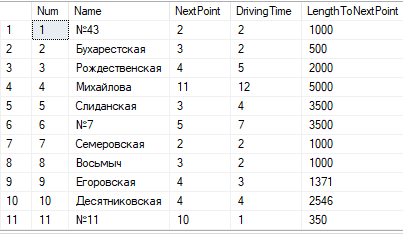


Рисунок 13. И снова смотрим на данные таблицы Points

Как видно из рисунков выше, база данных с успехом была восстановлена и данных были сохранены.

## **Итоговая диаграмма**

В результате разработки базы данных была получена диаграмма связей, представленная на рисунке 14.

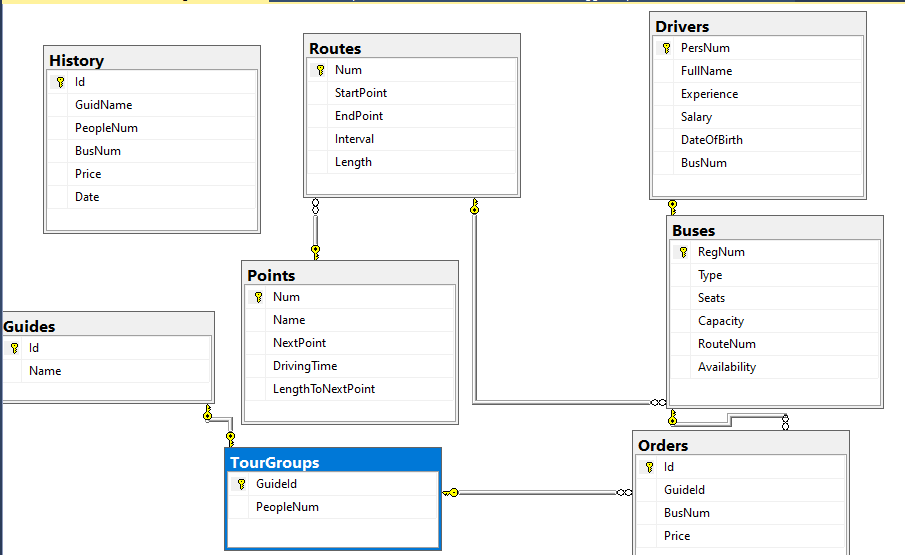


Рисунок 14. Диаграмма базы данных

Итоговая диаграмма полностью соответствует диаграмме, которую мы создали на этапе проектирования.

## **Вывод**

В ходе данного индивидуального задания были изучены методы работы с базами данных, триггерами, хранимыми процедурами, UDF и разработана собственная база, регулирующая работу автобусного парка.

На практике был разработан план будущего проекта, следуя которому, была спроектирована база данных. Итоговая диаграмма, полностью соответствует той, что мы составили на этапе планирования. Значит, цель работы достигнута.

# **СПИСОК ЛИТЕРАТУРЫ**

1. Сайт технической документации Microsoft. // Microsoft Docs. URL: https://docs.microsoft.com/

2. Руководство по MS SQL Server 2017. // Metanit. URL: <https://metanit.com/sql/sqlserver/>

3. Базы данных: основы, проектирование, использование: учеб. Пособие для вузов по направлению подгот. «Информатика и вычисл. Техника» / М.П. Малыхина. - 2-е изд. - СПб. : БХВ-Петербург, 2007. - X, 517 с.

4. Стратегия резервного копирования базы данных для MS SQL Server // Infostart. URL: <https://infostart.ru/public/101210/>

5. Автоматизация резервного копирования в MS SQL SERVER EXPRESS // Infostart. URL: <https://infostart.ru/public/1063767/>