**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

дисциплина «Разработка приложений баз данных для информационных систем»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

«Разработка серверной части информационной системы

в СУБД *MS* *SQL Server*»

ВАРИАНТ № 14

Выполнил:

студент гр. ИТП-31,

Ковтун А. В.

Принял:

доцент Асенчик О. Д.

Гомель 2023

**Цель работы:** разработать серверную часть клиент-серверной информационной системы, основанной на базе данных в заданной предметной области средствами СУБД *MS SQL Server*.

**Задание:** 1. Разработать логическую модель реляционной базы данных, моделирующую предметную область согласно своему варианту задания. Структура БД должна быть нормализована – таблицы должны удовлетворять требованиям третьей нормальной формы.

2. Создать базу данных и таблицы в СУБД *MS SQL Server* и заполнить их тестовым набором данных, для этого написать *Transact SQL* скрипт:

2.1. Создания базы данных и ее таблиц.

При создании таблиц должны быть назначены первичные и внешние ключи и установлены необходимые ограничения целостности данных. Наименования таблиц и полей давать в соответствии с соглашением об именовании этих объектов.

2.2. Заполнения не менее чем трех таблиц БД данными (см. пример внутри этого электронного курса).

При выборе таблиц для заполнения тестовыми наборами руководствоваться следующим:

выбранные таблицы должны содержать основную информацию, касающиеся предметной области приложения;

таблицы должны быть связаны непосредственно;

следует воздерживаться от выбора таблиц, характеризующих кадровую подсистему приложения (данные о сотрудниках и их должностях);

не менее, чем одна таблица должна находиться в схеме базы данных на стороне отношения «многие».

При этом заполнение осуществлять в следующем порядке: сначала генерируются данные для таблиц на стороне отношения «один» (таблицы- «справочники»), потом – в таблицы на стороне отношения «многие» («оперативные таблицы»).

БД заполнить записями в количестве, необходимом для отладки и демонстрации возможностей приложения. Таблицы на стороне отношения один должны содержать не менее 500 записей, таблице на стороне отношения многие должны содержать не менее 20000 записей.

3. Создать с использованием средств *Transact SQL* представления, позволяющие отображать данные в удобном для пользователя виде, и заменяющие часто используемые запросы на выборку из двух и более связанных таблиц.

4. Написать не менее трех хранимых процедур с параметрами для вставки и (или) обновление данных в таблицы базы данных.

**Ход работы:** выполнение лабораторной работы заключается в создании *SQL-*скрипта, который выполнит по порядку следующие действия:

1. Создание базы данных;
2. Создание таблиц в базе данных;
3. Создание связей между таблицами;
4. Генерация случайных данных и заполнение ими таблиц;
5. Создание представлений и процедур.

Предметная область и задание варианта показано на рисунке 1.

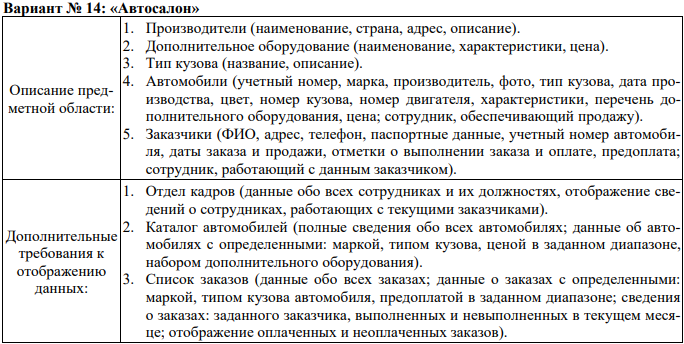


Рисунок 1 – Задание варианта

Для создания базы данных и таблиц использовались стандартные команды *CREATE DATABASE* и *CREATE TABLE*. В разработанном скрипте создается 9 таблиц: *CarModels, Cars, Positions, Employees, CarEquipments, ExtraEquipments, Manufacturers, CarcaseTypes, Clients.* После генерации таблиц были настроены связи между ними. На рисунке 2 показана диаграмма получившейся базы данных, позволяющая наглядно увидеть взаимодействия между таблицами.

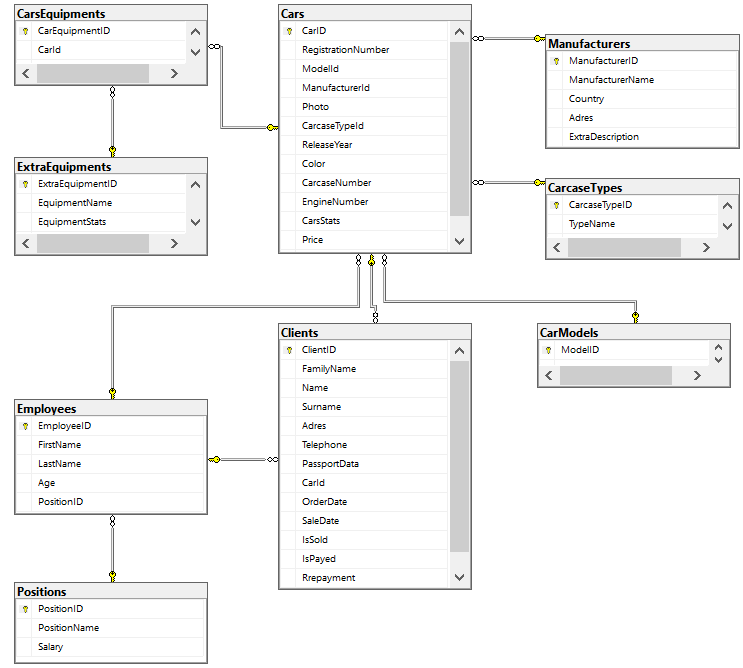
****

Рисунок 2 – Диаграмма базы данных

Заполнение таблиц выполнялось при помощи циклов и случайной генерации чисел и комбинаций символов. На рисунке 3 приведен пример одной из заполненных таблиц.

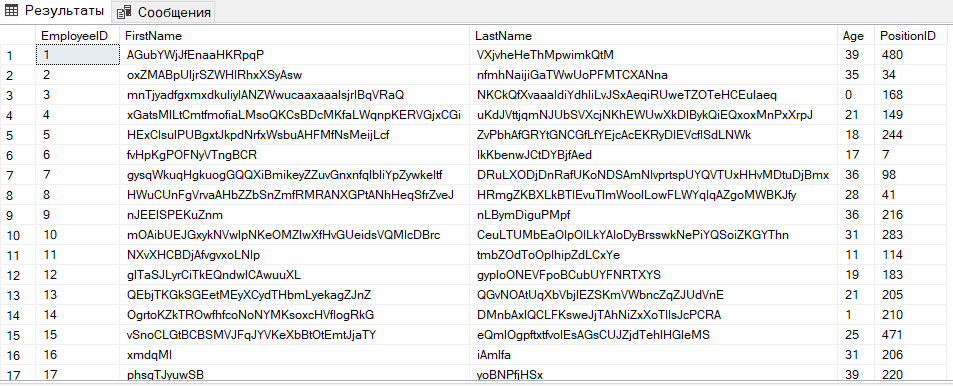


Рисунок 3 – Заполненная таблица *Employees*

Заключительной частью работы стало написание процедуры и представлений. Разработанное представление позволяет получить данный о должностях каждого сотрудника. Процедуры же предоставляют функции по вставке и обновлению данных в таблицах.

Листинг разработанного скрипта приведен в приложении А.

**Вывод:** результатом выполнения лабораторной работы является *SQL-*скрипт. Данный скрипт выполняет задачу по созданию базы данный и таблиц в ней, а также заполняет таблицы случайными значениями. Помимо таблицы были созданы представления и процедуры.

***GitHub:*** *https://github.com/postgoatleg/db\_lab1*

**ПРИЛОЖЕНИЕ А**

(обязательное)

***SQL*-скрипт**

USE master

CREATE DATABASE CarDealership

GO

ALTER DATABASE CarDealership SET RECOVERY SIMPLE

GO

USE CarDealership

CREATE TABLE dbo.Manufacturers (ManufacturerID int IDENTITY(1,1) NOT NULL PRIMARY KEY,

ManufacturerName nvarchar(50),

Country nvarchar(50),

Adres nvarchar(50),

ExtraDescription nvarchar(50))

CREATE TABLE dbo.ExtraEquipments (ExtraEquipmentID int IDENTITY(1,1) NOT NULL PRIMARY KEY,

EquipmentName nvarchar(50),

EquipmentStats nvarchar(50),

Price decimal(18,2))

CREATE TABLE dbo.CarcaseTypes (CarcaseTypeID int IDENTITY(1,1) NOT NULL PRIMARY KEY,

TypeName nvarchar(50),

TypeDescription nvarchar(50))

CREATE TABLE dbo.CarModels (ModelID int IDENTITY(1,1) NOT NULL PRIMARY KEY,

ModelName nvarchar(50))

CREATE TABLE dbo.Cars (CarID int IDENTITY(1,1) NOT NULL PRIMARY KEY,

RegistrationNumber nvarchar(20),

ModelId int,

ManufacturerId int,

Photo image,

CarcaseTypeId int,

ReleaseYear date,

Color nvarchar(50),

CarcaseNumber nvarchar(50),

EngineNumber nvarchar(50),

CarsStats nvarchar(50),

Price decimal(18,2),

SellerEmployeeId int) -- автомобили

CREATE TABLE dbo.CarsEquipments (CarEquipmentID int IDENTITY(1,1) NOT NULL PRIMARY KEY,

CarId int,

EquipmentId int) -- должности

CREATE TABLE dbo.Positions (PositionID int IDENTITY(1,1) NOT NULL PRIMARY KEY,

PositionName nvarchar(50),

Salary decimal(18,2)) -- должности

CREATE TABLE dbo.Employees (EmployeeID int IDENTITY(1,1) NOT NULL PRIMARY KEY,

FirstName nvarchar(50),

LastName nvarchar(50),

Age int,

PositionID int) -- сотрудники

CREATE TABLE dbo.Clients (ClientID int IDENTITY(1,1) NOT NULL PRIMARY KEY,

FamilyName nvarchar(20),

[Name] nvarchar(20),

Surname nvarchar(20),

Adres nvarchar(20),

Telephone varchar(12),

PassportData nvarchar(50),

CarId int,

OrderDate date,

SaleDate date,

IsSold bit,

IsPayed bit,

Rrepayment decimal(18,2),

ClientEmployeeId int) -- автомобили

-- Добавление связей между таблицами

ALTER TABLE dbo.Cars WITH CHECK ADD CONSTRAINT FK\_Cars\_CarModels FOREIGN KEY(ModelId)

REFERENCES dbo.CarModels (ModelID) ON DELETE CASCADE

GO

ALTER TABLE dbo.Cars WITH CHECK ADD CONSTRAINT FK\_Cars\_Manufacturers FOREIGN KEY(ManufacturerId)

REFERENCES dbo.Manufacturers (ManufacturerID) ON DELETE NO ACTION

GO

ALTER TABLE dbo.Cars WITH CHECK ADD CONSTRAINT FK\_Cars\_CarcaseTypes FOREIGN KEY(CarcaseTypeId)

REFERENCES dbo.CarcaseTypes (CarcaseTypeID) ON DELETE NO ACTION

GO

ALTER TABLE dbo.Cars WITH CHECK ADD CONSTRAINT FK\_Cars\_Employees FOREIGN KEY(SellerEmployeeId)

REFERENCES dbo.Employees (EmployeeID) ON DELETE CASCADE

GO

ALTER TABLE dbo.Employees WITH CHECK ADD CONSTRAINT FK\_Employees\_Positions FOREIGN KEY(PositionID)

REFERENCES dbo.Positions (PositionID) ON DELETE CASCADE

GO

ALTER TABLE dbo.Clients WITH CHECK ADD CONSTRAINT FK\_Clients\_Cars FOREIGN KEY(CarId)

REFERENCES dbo.Cars (CarID) ON DELETE CASCADE

GO

ALTER TABLE dbo.Clients WITH CHECK ADD CONSTRAINT FK\_Clients\_Employees FOREIGN KEY(ClientEmployeeId)

REFERENCES dbo.Employees (EmployeeID) ON DELETE NO ACTION

GO

ALTER TABLE dbo.CarsEquipments WITH CHECK ADD CONSTRAINT FK\_CarsEquipments\_Cars FOREIGN KEY(CarId)

REFERENCES dbo.Cars (CarID) ON DELETE CASCADE

GO

ALTER TABLE dbo.CarsEquipments WITH CHECK ADD CONSTRAINT FK\_CarsEquipments\_ExtraEquipments FOREIGN KEY(EquipmentId)

REFERENCES dbo.ExtraEquipments (ExtraEquipmentID) ON DELETE CASCADE

GO

SET NOCOUNT ON

DECLARE @Symbol CHAR(52)= 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz',

@Position int,

@ManufacturerName nvarchar(50),

@Country nvarchar(50),

@Adres nvarchar(50),

@ExtraDescription nvarchar(50),

@EquipmentName nvarchar(50),

@EquipmentStats nvarchar(50),

@Price decimal(18,2),

@PositionName nvarchar(50),

@Salary decimal(18,2),

@FirstName nvarchar(50),

@LastName nvarchar(50),

@Age int,

@PositionID int,

@ModelName nvarchar(50),

@Specifications nvarchar(50),

@RegistrationNumber nvarchar(20),

@ModelId int,

@CarcaseNumber nvarchar(17),

@EngineNumber nvarchar(17),

@ReleaseYear date,

@Mileage int,

@DriverId int,

@LastTI date,

@MechanicId int,

@SpecialMarks nvarchar(50),

@i int,

@NameLimit int,

@odate date,

@Inc\_Exp real,

@RowCount INT,

@NumberExtraEquipments int,

@NumberManufacturers int,

@NumberPositions int,

@NumberEmployees int,

@NumberCarModels int,

@NumberCars int,

@MinNumberSymbols int,

@MaxNumberSymbols int

SET @NumberManufacturers = 500

SET @NumberExtraEquipments = 500

SET @NumberPositions = 500

SET @NumberCarModels = 500

SET @NumberCars = 20000

SET @NumberEmployees = 20000

BEGIN TRAN

SELECT @i=0 FROM dbo.ExtraEquipments WITH (TABLOCKX) WHERE 1=0

SET @RowCount=1

SET @MinNumberSymbols=5

SET @MaxNumberSymbols=50

WHILE @RowCount<=@NumberExtraEquipments

BEGIN

SET @NameLimit=@MinNumberSymbols+RAND()\*(@MaxNumberSymbols-@MinNumberSymbols) -- имя от 5 до 50 символов

SET @i=1

SET @EquipmentName=''

SET @EquipmentStats=''

WHILE @i<=@NameLimit

BEGIN

SET @Position=RAND()\*52

SET @EquipmentName = @PositionName + SUBSTRING(@Symbol, @Position, 1)

SET @Position=RAND()\*52

SET @EquipmentStats = @PositionName + SUBSTRING(@Symbol, @Position, 1)

SET @Price = ROUND(RAND(CHECKSUM(NEWID())) \* (100), 2)

SET @i=@i+1

END

INSERT INTO dbo.ExtraEquipments (EquipmentName, EquipmentStats, Price) SELECT @EquipmentName, @EquipmentStats, @Price

SET @RowCount +=1

END

SELECT @i=0 FROM dbo.Manufacturers WITH (TABLOCKX) WHERE 1=0

SET @RowCount=1

SET @MinNumberSymbols=5

SET @MaxNumberSymbols=50

WHILE @RowCount<=@NumberManufacturers

BEGIN

SET @NameLimit=@MinNumberSymbols+RAND()\*(@MaxNumberSymbols-@MinNumberSymbols) -- имя от 5 до 50 символов

SET @i=1

SET @ManufacturerName=''

SET @Country=''

SET @Adres=''

SET @ExtraDescription=''

WHILE @i<=@NameLimit

BEGIN

SET @Position=RAND()\*52

SET @ManufacturerName = @PositionName + SUBSTRING(@Symbol, @Position, 1)

SET @Position=RAND()\*52

SET @Country = @PositionName + SUBSTRING(@Symbol, @Position, 1)

SET @Position=RAND()\*52

SET @Adres = @PositionName + SUBSTRING(@Symbol, @Position, 1)

SET @Position=RAND()\*52

SET @ExtraDescription = @PositionName + SUBSTRING(@Symbol, @Position, 1)

SET @i=@i+1

END

INSERT INTO dbo.Manufacturers (ManufacturerName, Country, Adres, ExtraDescription) SELECT @ManufacturerName, @Country, @Adres, @ExtraDescription

SET @RowCount +=1

END

SELECT @i=0 FROM dbo.Positions WITH (TABLOCKX) WHERE 1=0

SET @RowCount=1

SET @MinNumberSymbols=5

SET @MaxNumberSymbols=50

WHILE @RowCount<=@NumberPositions

BEGIN

SET @NameLimit=@MinNumberSymbols+RAND()\*(@MaxNumberSymbols-@MinNumberSymbols) -- имя от 5 до 50 символов

SET @i=1

SET @PositionName=''

WHILE @i<=@NameLimit

BEGIN

SET @Position=RAND()\*52

SET @PositionName = @PositionName + SUBSTRING(@Symbol, @Position, 1)

SET @Salary = ROUND(RAND(CHECKSUM(NEWID())) \* (100), 2)

SET @i=@i+1

END

INSERT INTO dbo.Positions (PositionName, Salary) SELECT @PositionName, @Salary

SET @RowCount +=1

END

SELECT @i=0 FROM dbo.Employees WITH (TABLOCKX) WHERE 1=0

SET @RowCount=1

SET @MinNumberSymbols=5

SET @MaxNumberSymbols=50

WHILE @RowCount<=@NumberEmployees

BEGIN

SET @NameLimit=@MinNumberSymbols+RAND()\*(@MaxNumberSymbols-@MinNumberSymbols) -- имя от 5 до 50 символов

SET @i=1

SET @FirstName = ''

SET @LastName = ''

WHILE @i<=@NameLimit

BEGIN

SET @Position=RAND()\*52

SET @FirstName = @FirstName + SUBSTRING(@Symbol, @Position, 1)

SET @Position=RAND()\*52

SET @LastName = @LastName + SUBSTRING(@Symbol, @Position, 1)

SET @Age = CAST(RAND()\*52 as int)

SET @PositionID = CAST( (1+RAND()\*(@NumberPositions-1)) as int)

SET @i=@i+1

END

INSERT INTO dbo.Employees (FirstName, LastName, Age, PositionID) SELECT @FirstName, @LastName, @Age, @PositionID

SET @RowCount +=1

END

SELECT @i=0 FROM dbo.CarModels WITH (TABLOCKX) WHERE 1=0

SET @RowCount=1

SET @MinNumberSymbols=5

SET @MaxNumberSymbols=50

WHILE @RowCount<=@NumberCarModels

BEGIN

SET @NameLimit=@MinNumberSymbols+RAND()\*(@MaxNumberSymbols-@MinNumberSymbols) -- имя от 5 до 50 символов

SET @i=1

SET @ModelName = ''

WHILE @i<=@NameLimit

BEGIN

SET @Position=RAND()\*52

SET @ModelName = @ModelName + SUBSTRING(@Symbol, @Position, 1)

SET @i=@i+1

END

INSERT INTO dbo.CarModels (ModelName) SELECT @ModelName

SET @RowCount +=1

END

COMMIT TRAN

GO

CREATE VIEW [dbo].[View\_EmployeeAndPositions]

AS

SELECT dbo.Employees.EmployeeID, dbo.Employees.FirstName, dbo.Employees.LastName, dbo.Employees.Age, dbo.Employees.PositionID,

dbo.Positions.PositionName, dbo.Positions.Salary

FROM dbo.Positions INNER JOIN

dbo.Employees ON dbo.Positions.PositionID = dbo.Employees.PositionID

GO

CREATE PROCEDURE AddExtraEqupment (@EquipmentName nvarchar(20), @EquipmentStats int, @Price decimal(18,2))

AS INSERT INTO dbo.ExtraEquipments(

EquipmentName,

EquipmentStats,

Price

)

VALUES

(@EquipmentName,

@EquipmentStats,

@Price)

GO

CREATE PROCEDURE AddEmployee (@FirstName nvarchar(50), @LastName nvarchar(50), @Age int, @PositionID int)

AS INSERT INTO dbo.Employees(

FirstName,

LastName,

Age,

PositionID)

VALUES

(@FirstName,

@LastName ,

@Age,

@PositionID)

GO

CREATE PROCEDURE UpdateManufacturerAdres (@ManufacturerId int, @NewAdres nvarchar(50))

AS UPDATE dbo.Manufacturers

SET Adres = @NewAdres

WHERE(

dbo.Manufacturers.ManufacturerID = @NewAdres

);

GO