**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

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

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

по дисциплине «Разработка приложений баз данных для информационных систем»

на тему: «Разработка серверной части информационной системы

в СУБД *MS* *SQL Server*»

Выполнил: студент гр. ИТП-31

Козлов О. 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:** для выполнения данной лабораторной работы был разработан *SQL* скрипт. Первый этап заключается в создании базы данных и последующей генерации таблиц в ней согласно варианту. На рисунке 1 показано задание для 15 варианта. Стоит отметить, что помимо создания все указанных в задании таблиц были созданы вспомогательные таблицы для лучшей настройки связей.

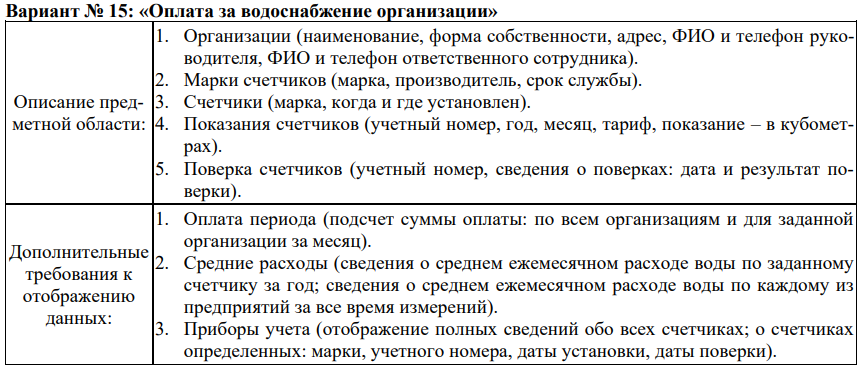


Рисунок 1 – Условие задания варианта

Листинг скрипта по созданию базы данных и последующего занесения в созданную БД таблиц приведен в приложении А. На рисунке 2 показано структура таблиц, полученная в результате выполнения скрипта.

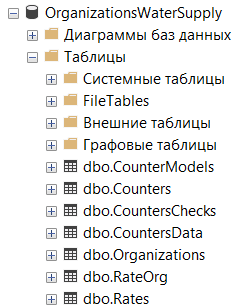


Рисунок 2 – Таблицы базы данных

После создания всех необходимых таблиц необходимо настроить все связи между ними. За это отвечает следующая часть скрипта, листинг которой приведен в приложении Б. Для наглядной демонстрации созданных связей между таблицами была создана диаграмма базы данных. Она показана на рисунке 3. Как видно из диаграммы в БД помимо основных таблиц также присутствуют вспомогательная таблица *Rates* и связующая *RateOrg.*

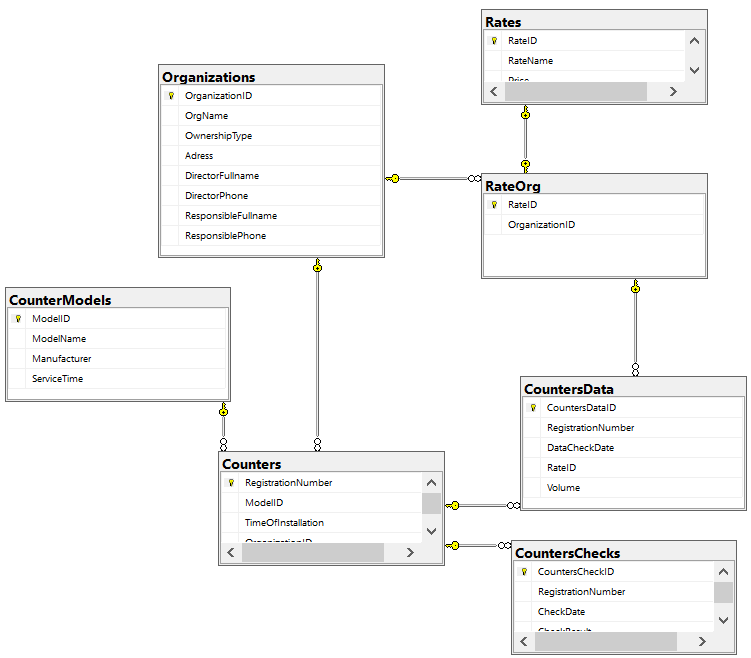


Рисунок 3 – Связи между таблицами

Как только все связи настроены можно приступить к генерации записей в данных таблицах. Для заполнения были выбраны таблицы: *Organizations, CounterModels, Counters*. В первую очередь были заполнены таблицы на стороне отношения «один». Так как их данные будут использоваться в дальнейших таблицах. Листинг части скрипта, отвечающего за заполнение трех таблиц, приведен в приложении В. На рисунке 4 приведен пример заполненной таблицы.

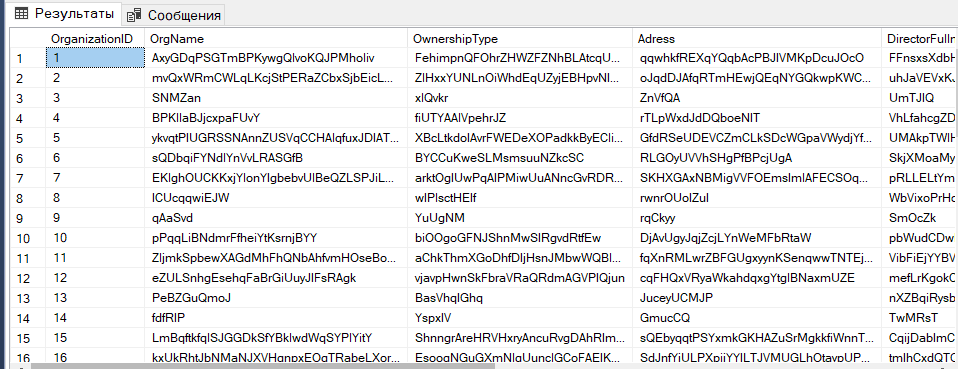


Рисунок 4 – Заполненная таблица

Заключительной частью работы является создание представления и нескольких хранимых процедур для вставки и обновления данных таблиц. Листинг создания представления и процедур представлен в приложении Г.

**Вывод:** в результате выполнения лабораторной работы был разработан *SQL* скрипт, позволяющий создать базу данных и таблицы, настроить связи между полученными таблицами, провести автоматическое заполнение данными. Так же было разработано представление и несколько хранимых процедур, предоставляющих функционал по созданию и обновлению записей в таблицах.

https://github.com/OLagsV

**ПРИЛОЖЕНИЕ А**

(обязательное)

**Создание БД и таблиц**

USE master

CREATE DATABASE OrganizationsWaterSupply

GO

ALTER DATABASE OrganizationsWaterSupply SET RECOVERY SIMPLE

GO

USE OrganizationsWaterSupply

--DROP TABLE Fuels, Tanks, Operations

--DROP VIEW View\_AllOperations

-- Создание таблиц

CREATE TABLE dbo.Organizations(OrganizationID int IDENTITY(1,1) NOT NULL PRIMARY KEY,

OrgName nvarchar(50),

OwnershipType nvarchar(50),

Adress nvarchar(50),

DirectorFullname nvarchar(50),

DirectorPhone varchar(11),

ResponsibleFullname nvarchar(50),

ResponsiblePhone varchar(11))

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

ModelName nvarchar(50),

Manufacturer nvarchar(50),

ServiceTime int)

CREATE TABLE dbo.Counters (RegistrationNumber int IDENTITY(1,1) NOT NULL PRIMARY KEY,

ModelID int,

TimeOfInstallation date,

OrganizationID int)

CREATE TABLE dbo.CountersData (CountersDataID int IDENTITY(1,1) NOT NULL PRIMARY KEY,

RegistrationNumber int,

DataCheckDate date,

RateID int,

Volume int,)

CREATE TABLE dbo.CountersChecks (CountersCheckID int IDENTITY(1,1) NOT NULL PRIMARY KEY,

RegistrationNumber int,

CheckDate date,

CheckResult nvarchar(50))

CREATE TABLE dbo.Rates (RateID int IDENTITY(1,1) NOT NULL PRIMARY KEY,

RateName nvarchar(50),

Price decimal(18,2))

CREATE TABLE dbo.RateOrg (RateID int NOT NULL PRIMARY KEY,

OrganizationID int)

**ПРИЛОЖЕНИЕ Б**

(обязательное)

**Настройка связей**

ALTER TABLE dbo.Counters WITH CHECK ADD CONSTRAINT FK\_Counters\_CounterModels FOREIGN KEY(ModelID)

REFERENCES dbo.CounterModels (ModelID) ON DELETE CASCADE

GO

ALTER TABLE dbo.Counters WITH CHECK ADD CONSTRAINT FK\_Counters\_Organizations FOREIGN KEY(OrganizationID)

REFERENCES dbo.Organizations (OrganizationID) ON DELETE CASCADE

GO

ALTER TABLE dbo.CountersChecks WITH CHECK ADD CONSTRAINT FK\_CountersChecks\_Counters FOREIGN KEY(RegistrationNumber)

REFERENCES dbo.Counters (RegistrationNumber) ON DELETE CASCADE

GO

ALTER TABLE dbo.RateOrg WITH CHECK ADD CONSTRAINT FK\_RateOrg\_Rates FOREIGN KEY(RateID)

REFERENCES dbo.Rates (RateID) ON DELETE CASCADE

GO

ALTER TABLE dbo.RateOrg WITH CHECK ADD CONSTRAINT FK\_RateOrg\_Organizations FOREIGN KEY(OrganizationID)

REFERENCES dbo.Organizations (OrganizationID) ON DELETE CASCADE

GO

ALTER TABLE dbo.CountersData WITH CHECK ADD CONSTRAINT FK\_CountersData\_Counters FOREIGN KEY(RegistrationNumber)

REFERENCES dbo.Counters (RegistrationNumber) ON DELETE CASCADE

GO

ALTER TABLE dbo.CountersData WITH CHECK ADD CONSTRAINT FK\_CountersData\_RateOrg FOREIGN KEY(RateID)

REFERENCES dbo.RateOrg (RateID) ON DELETE NO ACTION

GO

**ПРИЛОЖЕНИЕ В**

(обязательное)

**Заполнение таблиц**

SET NOCOUNT ON

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

@Position int,

@OrgName nvarchar(50),

@OwnershipType nvarchar(50),

@Adress nvarchar(50),

@DirectorFullname nvarchar(50),

@DirectorPhone varchar(11),

@ResponsibleFullname nvarchar(50),

@ResponsiblePhone varchar(11),

@ModelName nvarchar(50),

@Manufacturer nvarchar(50),

@ServiceTime int,

@ModelID int,

@TimeOfInstallation date,

@OrganizationID int,

@i int,

@NameLimit int,

@odate date,

@Inc\_Exp real,

@RowCount INT,

@NumberOrganizations int,

@NumberCounterModels int,

@NumberCounters int,

@MinNumberSymbols int,

@MaxNumberSymbols int

SET @NumberOrganizations = 500

SET @NumberCounterModels = 500

SET @NumberCounters = 20000

BEGIN TRAN

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

-- Заполнение видов топлива

SET @RowCount=1

SET @MinNumberSymbols=5

SET @MaxNumberSymbols=50

WHILE @RowCount<=@NumberOrganizations

BEGIN

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

SET @i=1

SET @OrgName=''

SET @OwnershipType=''

SET @Adress=''

SET @DirectorFullname=''

SET @DirectorPhone=''

SET @ResponsibleFullname=''

SET @ResponsiblePhone=''

WHILE @i<=@NameLimit

BEGIN

SET @Position=RAND()\*52

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

SET @Position=RAND()\*52

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

SET @Position=RAND()\*52

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

SET @Position=RAND()\*52

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

SET @Position=RAND()\*52

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

SET @Position=RAND()\*52

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

SET @Position=RAND()\*52

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

SET @i=@i+1

END

INSERT INTO dbo.Organizations(OrgName, OwnershipType,Adress, DirectorFullname, DirectorPhone, ResponsibleFullname,ResponsiblePhone) SELECT @OrgName, @OwnershipType,@Adress, @DirectorFullname, @DirectorPhone, @ResponsibleFullname,@ResponsiblePhone

SET @RowCount +=1

END

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

SET @RowCount=1

SET @MinNumberSymbols=5

SET @MaxNumberSymbols=50

WHILE @RowCount<=@NumberCounterModels

BEGIN

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

SET @i=1

SET @ModelName=''

SET @Manufacturer=''

WHILE @i<=@NameLimit

BEGIN

SET @Position=RAND()\*52

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

SET @Position=RAND()\*52

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

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

SET @i=@i+1

END

INSERT INTO dbo.CounterModels(ModelName, Manufacturer, ServiceTime) SELECT @ModelName, @Manufacturer, @ServiceTime

SET @RowCount +=1

END

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

SET @RowCount=1

SET @MinNumberSymbols=5

SET @MaxNumberSymbols=50

WHILE @RowCount<=@NumberCounters

BEGIN

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

SET @i=1

WHILE @i<=@NameLimit

BEGIN

SET @ModelID = CAST( (1+RAND()\*(@NumberCounterModels-1)) as int)

SET @TimeOfInstallation=dateadd(day,-RAND()\*15000,GETDATE())

SET @OrganizationID = CAST( (1+RAND()\*(@NumberOrganizations-1)) as int)

SET @i=@i+1

END

INSERT INTO dbo.Counters(ModelID, TimeOfInstallation, OrganizationID) SELECT @ModelID, @TimeOfInstallation, @OrganizationID

SET @RowCount +=1

END

COMMIT TRAN

GO

**ПРИЛОЖЕНИЕ Г**

(обязательное)

**Представление и хранимые процедуры**

CREATE VIEW [dbo].[View\_OrganizationsCounters]

AS

SELECT dbo.Organizations.OrganizationID, dbo.Organizations.OrgName, dbo.Organizations.OwnershipType, dbo.Organizations.Adress, dbo.Organizations.DirectorFullname, dbo.Organizations.DirectorPhone, dbo.Organizations.ResponsibleFullname, dbo.Organizations.ResponsiblePhone,

dbo.CounterModels.ModelID, dbo.CounterModels.ModelName, dbo.CounterModels.Manufacturer, dbo.CounterModels.ServiceTime,

dbo.Counters.RegistrationNumber, dbo.Counters.TimeOfInstallation

FROM dbo.CounterModels INNER JOIN

dbo.Counters ON dbo.CounterModels.ModelID = dbo.Counters.ModelID INNER JOIN

dbo.Organizations ON dbo.Counters.OrganizationID = dbo.Organizations.OrganizationID

GO

CREATE PROCEDURE ChangeOrganizationAdress (@OrganizationID int, @NewAdress nvarchar(50))

AS UPDATE dbo.Organizations

SET dbo.Organizations.Adress = @NewAdress

WHERE(

dbo.Organizations.OrganizationID = @OrganizationID

);

GO

CREATE PROCEDURE AddOrganization (@OrgName nvarchar(50), @OwnershipType nvarchar(50), @Adress nvarchar(50), @DirectorFullname nvarchar(50), @DirectorPhone nvarchar(11), @ResponsibleFullname nvarchar(50), @ResponsiblePhone nvarchar(11))

AS INSERT INTO dbo.Organizations(

OrgName,

OwnershipType,

Adress,

DirectorFullname,

DirectorPhone,

ResponsibleFullname,

ResponsiblePhone)

VALUES

(@OrgName,

@OwnershipType ,

@Adress,

@DirectorFullname,

@DirectorPhone,

@ResponsibleFullname,

@ResponsiblePhone)

GO

CREATE PROCEDURE AddCounter (@ModelID int, @TimeOfInstallation date, @OrganizationID int)

AS INSERT INTO dbo.Counters(

ModelID,

TimeOfInstallation,

OrganizationID)

VALUES(

@ModelID,

@TimeOfInstallation,

@OrganizationID)

GO