**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

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

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

**ОТЧЁТ 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»

Вариант 2

Выполнила: студентка гр. ИТП-31

Гайкевич Т. Г.

Принял: доцент

Асенчик О. Д.

Гомель 2020

**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.

**Вариант задания:**

Предметная область задания изображена на рисунке 1.

**Изображение выглядит как снимок экрана

Автоматически созданное описание**

Рисунок 1 – Предметная область

**Ход работы:**

**Задание 1**

Первым делом была сформирована диаграмма базы данных, соответствующая заданному варианту и удовлетворяющая условиям задания.

Сформированная диаграмма изображена на рисунке 2.

Изображение выглядит как снимок экрана

Автоматически созданное описание

Рисунок 2 – Сформированная диаграмма базы данных

**Задание 2**

Далее была создана база данных и были написаны скрипты для создания таблиц базы данных.

Скриншоты скрипта создания таблиц базы данных изображён на рисунке 3.

Изображение выглядит как снимок экрана

Автоматически созданное описаниеИзображение выглядит как снимок экрана

Автоматически созданное описание

Изображение выглядит как снимок экрана

Автоматически созданное описание

Рисунок 3 – Скрипт создания таблиц базы данных

Скриншот скрипта, выполняющего заполнение таблиц тестовыми данными изображён на рисунке 4.

Изображение выглядит как снимок экрана

Автоматически созданное описание

Рисунок 4 – Скрипт заполнения таблиц тестовыми данными

**Задание 3**

Создать скрипт, создающий представление (рисунок 5).

Изображение выглядит как птица

Автоматически созданное описание

Рисунок 5 – Скрипт, создающий представление данных

Вывод представления выглядит следующим образом (рисунок 6).

Изображение выглядит как снимок экрана

Автоматически созданное описание

Рисунок 6 – Вывод представления на экран

**Задание 4**

Создать хранимые процедуры на вставку данных всех таблиц базы данных (рисунок 7).

Изображение выглядит как снимок экрана

Автоматически созданное описание

Рисунок 7 – Скрипт для создания хранимых процедур для вставки данных в таблицы

**Вывод:**

В ходе работы была разработана база данных, соответствующая заданной предметной области. Написаны скрипты для взаимодействия с данными таблиц этой базы данных.

**ПРИЛОЖЕНИЕ А**

Листинг созданных скриптов

***CreateTables.sql***

begin tran

CREATE TABLE [Employees] (

[Id] int IDENTITY(1,1) PRIMARY KEY,

[Surname] varchar(50) not null,

[Name] varchar(50) not null,

[Patronymic] varchar(50) not null,

[Birthday] date not null,

[PositionId] int not null

)

GO

CREATE TABLE [Positions] (

[Id] int IDENTITY(1,1) PRIMARY KEY,

[Name] varchar(50) not null

)

GO

CREATE TABLE [Hotels] (

[Id] int IDENTITY(1,1) NOT NULL PRIMARY KEY,

[Name] varchar(50) not null,

[Country] varchar(50) not null,

[Town] varchar(50) not null,

[Address] varchar(50) not null,

[Phone] int not null,

[NumberOfStars] int not null,

[ContactPerson] varchar(50) not null,

[HotelPhoto] varbinary(max) not null,

[RoomFoto] varbinary(max) not null

)

GO

CREATE TABLE [ServicesVouchers] (

[Id] int IDENTITY(1,1) NOT NULL PRIMARY KEY,

[ReservationMark] varchar(50) not null,

[PaymentMark] varchar(50) not null,

[VoucherId] int not null,

[ServiceId] int not null

)

GO

CREATE TABLE [Clients] (

[Id] int IDENTITY(1,1) NOT NULL PRIMARY KEY,

[Surname] varchar(50) not null,

[Name] varchar(50) not null,

[Patronymic] varchar(50) not null,

[Birthday] date not null,

[Gender] bit not null,

[Address] varchar(50) not null,

[Phone] int not null,

[PassportData] varchar(50) not null

)

GO

CREATE TABLE [TypesOfRest] (

[Id] int IDENTITY(1,1) NOT NULL PRIMARY KEY,

[Name] varchar(50) not null,

[Description] varchar(150) not null,

[Limitation] varchar(100) not null

)

GO

CREATE TABLE [Services] (

[Id] int IDENTITY(1,1) NOT NULL PRIMARY KEY,

[Name] varchar(50) not null,

[Description] varchar(150) not null,

[Cost] money not null

)

GO

CREATE TABLE [Vouchers] (

[Id] int IDENTITY(1,1) NOT NULL PRIMARY KEY,

[Name] varchar(50) not null,

[StartDate] date not null,

[EndDate] date not null,

[HotelId] int not null,

[TypeOfRestId] int not null,

[EmployeeId] int not null,

[ClientId] int not null

)

GO

ALTER TABLE [ServicesVouchers] ADD FOREIGN KEY ([VoucherId]) REFERENCES [Vouchers] ([Id]) on delete cascade

GO

ALTER TABLE [ServicesVouchers] ADD FOREIGN KEY ([ServiceId]) REFERENCES [Services] ([Id]) on delete cascade

GO

ALTER TABLE [Vouchers] ADD FOREIGN KEY ([HotelId]) REFERENCES [Hotels] ([Id]) on delete cascade

GO

ALTER TABLE [Vouchers] ADD FOREIGN KEY ([TypeOfRestId]) REFERENCES [TypesOfRest] ([Id]) on delete cascade

GO

ALTER TABLE [Vouchers] ADD FOREIGN KEY ([EmployeeId]) REFERENCES [Employees] ([Id]) on delete cascade

GO

ALTER TABLE [Vouchers] ADD FOREIGN KEY ([ClientId]) REFERENCES [Clients] ([Id]) on delete cascade

GO

commit tran

***CreateInsertProcedures.sql***

begin tran

go

create procedure InsertPosition

@Name varchar(50)

as

begin

insert into Positions(Name)

values (@Name)

end

go

create procedure InsertEmployee

@Surname varchar(50),

@Name varchar(50),

@Patronymic varchar(50),

@Birthday date,

@PositionId int

as

begin

if exists(select \* from Positions where Id = @PositionId)

insert into Employees(Surname,

Name,

Patronymic,

Birthday,

PositionId)

values (@Surname,

@Name,

@Patronymic,

@Birthday,

@PositionId)

end

go

create procedure InsertClient

@Surname varchar(50),

@Name varchar(50),

@Patronymic varchar(50),

@Birthday date,

@Gender bit,

@Address varchar(50),

@Phone int,

@PassportData varchar(50)

as

begin

insert into Clients(Surname,

Name,

Patronymic,

Birthday,

Gender,

Address,

Phone,

PassportData)

values (@Surname,

@Name,

@Patronymic,

@Birthday,

@Gender,

@Address,

@Phone,

@PassportData)

end

go

commit tran

***InitializeTables.sql***

use TouristAgency

go

declare @ReferenceRecordCount int = 500,

@OperationRecordCount int = 20000

declare @Symbol char(52)= 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz',

@Digit char(10) = '0123456789',

@Position int,

@i int,

@NameLimit int,

@RowCount int,

@MinNumberSymbols int = 5,

@MaxNumberSymbols int = 50

declare @PositionsCount int = @ReferenceRecordCount

declare @EmployeesCount int = @OperationRecordCount

declare @ClientsCount int = @ReferenceRecordCount

declare @TypesOfRestCount int = @ReferenceRecordCount

declare @HotelsCount int = @ReferenceRecordCount

begin tran

-- Заполнение таблицы Должности

declare @Name varchar(50)

set @RowCount = 1

while @RowCount <= @PositionsCount

begin

set @NameLimit=@MinNumberSymbols + rand() \* (@MaxNumberSymbols-@MinNumberSymbols) -- длина строки от 5 до 50

set @Name = ''

set @i = 1

while @i < @NameLimit

begin

set @Position = rand() \* 52

set @Name = @Name + substring(@Symbol, @Position, 1)

set @i = @i + 1

end

exec InsertPosition @Name

set @RowCount += 1

end

--Заполнение таблицы Сотрудники

declare @Surname varchar(50),

@Patronymic varchar(50),

@Birthday date,

@PositionId int

set @RowCount = 1

while @RowCount <= @EmployeesCount

begin

set @NameLimit=@MinNumberSymbols + rand() \* (@MaxNumberSymbols-@MinNumberSymbols) -- длина строки от 5 до 50

set @Surname = ''

set @Name = ''

set @Patronymic = ''

set @PositionId = rand() \* @PositionsCount

set @Birthday = dateadd(day,-8000 - RAND()\*10000,GETDATE())

set @i = 1

while @i < @NameLimit

begin

set @Position = rand() \* 52

set @Surname = @Surname + substring(@Symbol, @Position, 1)

set @Position = rand() \* 52

set @Name = @Name + substring(@Symbol, @Position, 1)

set @Position = rand() \* 52

set @Patronymic = @Patronymic + substring(@Symbol, @Position, 1)

set @i = @i + 1

end

exec InsertEmployee @Surname,

@Name,

@Patronymic,

@Birthday,

@PositionId

set @RowCount += 1

end

--Заполнение таблицы Клиенты

declare @Gender bit,

@Address varchar(50),

@Phone int,

@PassportData varchar(50)

set @RowCount = 1

while @RowCount <= @ClientsCount

begin

set @NameLimit=@MinNumberSymbols + rand() \* (@MaxNumberSymbols-@MinNumberSymbols) -- длина строки от 5 до 50

set @Surname = ''

set @Name = ''

set @Patronymic = ''

set @Birthday = dateadd(day,-5000-RAND()\*10000,GETDATE())

set @Gender = cast(round(rand(), 0) as bit)

set @Address = ''

set @Phone = rand() \* 7

set @PassportData = ''

set @i = 1

while @i < @NameLimit

begin

set @Position = rand() \* 52

set @Surname = @Surname + substring(@Symbol, @Position, 1)

set @Position = rand() \* 52

set @Name = @Name + substring(@Symbol, @Position, 1)

set @Position = rand() \* 52

set @Patronymic = @Patronymic + substring(@Symbol, @Position, 1)

set @Position = rand() \* 52

set @Address = @Address + substring(@Symbol, @Position, 1)

set @Position = rand() \* 52

set @PassportData = @PassportData + substring(@Symbol, @Position, 1)

set @i = @i + 1

end

exec InsertClient @Surname,

@Name,

@Patronymic,

@Birthday,

@Gender,

@Address,

@Phone,

@PassportData

set @RowCount += 1

end

commit tran