Лабораторная работа № 1

Создание базы данных и наполнение ее тестовыми данными

Задание

1. Самостоятельно выбрать предметную область и разработать логический макет реляционной базы данных, состоящей из 4-х таблиц – трех «родительских» и одной «дочерней». Родительские таблицы соответствуют сущностям ER-модели, дочерняя таблица соответствует связи ER-модели и реализует тернарное отношение типа «многие-ко-многим». Прототипом разрабатываемой базы данных может служить учебная база данных «Поставщики, детали и проекты», условно идентифицируемая именем dbSPJ. Для каждой из родительских таблиц определить по одному суррогатному ключу IDENTITY(1,1). Дочерняя таблица должна содержать дополнительные атрибуты, чтобы не превратиться в три внешних ключа.
2. Написать программу (консольное приложение), которая автоматически генерирует относительно правдоподобные тестовые данные для всех четырех таблиц базы данных. Количество записей – не менее 1000 на каждую таблицу.
3. Создать три сценария на Transact-SQL:
    a. Сценарий создания базы данных и базовых таблиц,
    b. Сценарий определения ограничений,
    c. Сценарий массированного копирования данных в базу данных.
4. Последовательно выполнить сценарии (запустить сценарии на выполнение) двумя способами:
    a. С помощью SQL Server Management Studio,
    b. С командной строки (SQLCMD.exe).
5. Построить диаграмму базы данных в среде SQL Server Management Studio.

Примечания.

1. При создании сценария (сценариев) создания базы данных, объектов базы данных (таблиц, ограничений, правил и умолчаний) следует руководствоваться следующими инструкциями Transact-SQL.

Создание новой базы данных и файлов, используемых для ее хранения
Инструкция CREATE DATABASE (Transact-SQL)
http://msdn.microsoft.com/ru-ru/library/ms176061(v=sql.105).aspx

Создание новой таблицы в базе данных
Инструкция CREATE TABLE (Transact-SQL)
http://msdn.microsoft.com/ru-ru/library/ms174979(v=sql.105).aspx

Изменение определения таблицы путем изменения, добавления или удаления столбцов и ограничений
Инструкция ALTER TABLE (Transact-SQL)
http://msdn.microsoft.com/ru-ru/library/ms190273(v=sql.105).aspx

Создание объекта, называемого правилом
Инструкция CREATE RULE (Transact-SQL)
http://msdn.microsoft.com/ru-ru/library/ms188064(v=sql.105).aspx

Привязывание правила к столбцу таблицы
Системная хранимая процедура sp_bindrule (Transact-SQL)
http://msdn.microsoft.com/ru-ru/library/ms176063(v=sql.105).aspx

Создание объекта «Значение по умолчанию»
Инструкция CREATE DEFAULT (Transact-SQL)
http://msdn.microsoft.com/ru-ru/library/ms173565(v=sql.105).aspx

Привязывание значения по умолчанию к столбцу таблицы
Системная хранимая процедура sp_bindefault (Transact-SQL)
http://msdn.microsoft.com/ru-ru/library/ms177503(v=sql.105).aspx

Импорт файла данных в таблицу базы данных в формате, указанном пользователем
Инструкция BULK INSERT (Transact-SQL)
http://msdn.microsoft.com/ru-ru/library/ms188365(v=sql.105).aspx

2. При создании таблиц базы данных следует руководствоваться следующими типами данных (см. https://msdn.microsoft.com/ru-ru/library/ms187752(v=sql.110).aspx):
    a. Точные числа: bigint int smallint tinyint bit decimal numeric money smallmoney;
    b. Приблизительные числа: float real Дата и время: date, datetime2, datetime, datetimeoffset, smalldatetime, time;
    c. Символьные строки: char varchar text;
    d. Символьные строки в Юникоде: nchar nvarchar ntext;
    e. Двоичные данные: binary varbinary image;
    f. Прочие типы данных: cursor, hierarchyid, sql_variant, table, timestamp, uniqueidentifier, xml.

3. В зависимости от параметров хранения, некоторые типы данных в SQL Server относятся к следующим группам:
    a. типы данных больших значений: varchar(max), nvarchar(max) и varbinary(max);
    b. типы данных больших объектов: text, ntext, image, varchar(max), nvarchar(max), varbinary(max) и xml.

Пример выполнения лабораторной работы

Описание демонстрационной базы данных «Поставщики, детали и проекты» (dbSPJ)

В состав демонстрационной базы данных dbSPJ входят четыре таблицы:

1. Таблица S представляет поставщиков. Каждый поставщик имеет уникальный номер (Sno), уникальное имя (Sname), значение рейтинга или статуса (Status), место расположения (City). Предполагается, что каждый поставщик находится только в одном городе.
2. Таблица P представляет детали. У каждой детали есть уникальный номер (Pno), название детали (Pname), цвет (Color), вес (Weight), город, где хранится этот вид деталей (City). Предполагается (где это имеет значение), что вес детали приведен в граммах. Также предполагается, что каждая деталь имеет только один цвет и хранится на складе только в одном городе.
3. Таблица J представляет проекты. Каждый проект имеет уникальный номер (Jno), уникальное имя (Jname), место расположения (City).
4. Таблица SPJ представляет поставки. Она в известном смысле служит для организации логической связи трех других таблиц. Каждая поставка характеризуется следующими атрибутами: номером поставщика (Sno), номером детали (Pno), номером проекта (Jno) и количеством (Qty). Предполагается, что в одно и то же время может быть не более одной поставки для одного поставщика, одной детали и одного проекта, поэтому для каждой поставки комбинация значений атрибутов Sno, Pno и Jno уникальна с точки зрения набора текущих поставок, представленных в таблице SPJ. Атрибуты Sno, Pno и Jno называются внешними ключами. База данных должна удовлетворять следующему правилу поддержки ссылочной целостности: не должно быть значений внешних ключей, не имеющих соответствия. Проще говоря, правило утверждает, что если B ссылается на A, то A должно существовать.

Пример заполнения исходными данными таблиц базы данных dbSPJ

Таблица поставщиков (S)

Sno	Sname	Status	City
1	Алмаз	20	Смоленск
2	Циклон	10	Владимир
3	Дельта	30	Владимир
4	Орион	20	Смоленск
5	Аргон	30	Ярославль

Таблица деталей (P)

Pno	Pname	Color	Weight	City
1	Гайка	Красный	12	Смоленск
2	Болт	Зеленый	17	Владимир
3	Винт	Синий	17	Рязань
4	Винт	Красный	14	Смоленск
5	Шайба	Синий	12	Владимир
6	Шпунт	Красный	19	Смоленск

Таблица проектов (J)

Jno	Jname	City
1	Байкал	Владимир
2	Ангара	Рязань
3	Енисей	Ярославль
4	Алтай	Ярославль
5	Урал	Смоленск
6	Амур	Тверь
7	Алдан	Смоленск

Таблица поставок (SPJ)

Sno	Pno	Jno	Qty
1	1	1	200
1	1	4	700
2	3	1	400
2	3	2	200
2	3	3	200
2	3	4	500
2	3	5	600
2	3	6	400
2	3	7	800
2	5	2	100
3	3	1	200
3	4	2	500
4	6	3	300
4	6	7	300
5	2	2	200
5	2	4	100
5	5	5	500
5	5	7	100
5	6	2	200
5	1	4	100
5	3	4	200
5	4	4	800
5	5	4	400
5	6	4	500

Основные (обязательные) ограничения целостности для базы данных dbSPJ

1. Допустимыми номерами поставщиков, деталей и проектов являются целые положительные числа.
2. В случае удаления объекта ссылки внешнего ключа операцию удаления ссылочных кортежей запретить.
3. Значения статуса поставщика должно целым числом в диапазоне от 0 до 100.
4. Допустимыми городами являются:
   Смоленск

Владимир
Рязань
Тверь
Тула
Калуга
Ярославль

5. Допустимыми цветами деталей являются:
   Красный
   Зеленый
   Синий
6. Вес детали должен быть положительным вещественным числом или нулем.
7. Количество деталей в поставке должно быть положительным целым числом или нулем.

Дополнительные (желательные, но не обязательные) ограничения целостности для базы данных dbSPJ

1. Все красные детали должны весить не менее 50 Г.
2. В любой момент в Ярославле может находиться не более одного поставщика.
3. Ни одна поставка по количеству деталей не может превышать удвоенное среднее значение количества для всех поставок.
4. Поставщик с наибольшим статусом не может находиться в одном городе с поставщиком с наименьшим статусом.
5. Должна существовать, по крайней мере, одна красная деталь.
6. Среднее значение статуса поставщика должно быть больше 18.
7. Каждый поставщик в Смоленске должен поставлять деталь под номером 2.
8. Хотя бы одна красная деталь должна весить меньше 50 Г.
9. Поставщики в Смоленске должны поставлять больше видов деталей, чем поставщики во Владимире.
10. Поставщики в Смоленске должны в сумме поставлять больше деталей, чем поставщики во Владимире.
11. Ни одна поставка не может быть сокращена (за одно обновление) более чем вдвое по сравнению с текущим значением.
12. Поставщики из Ярославля могут переехать только в Смоленск или Владимир, а поставщики из Смоленска – только во Владимир.

Сценарии создания базы данных dbSPJ

Сценарий № 1: Создание базы данных и базовых таблиц
(сохраняем в файле, например, в SQLQuery1.sql)

```
/****** Начинаем работать в контексте системной базы данных [master] ******/
USE [master]
GO

-- Если база данных [dbSPJ] уже существует, уничтожаем ее
IF EXISTS (SELECT name FROM sys.databases WHERE name = N'dbSPJ')
DROP DATABASE [dbSPJ]
GO

/****** Создаем базу данных [dbSPJ] ******/
CREATE DATABASE [dbSPJ]
GO

/****** Переходим в контекст созданной базы данных [dbSPJ] ******/
USE [dbSPJ]
GO

/****** Создаем таблицу поставщиков [S] ******/
CREATE TABLE [dbo].[S](
[Sno] [int] IDENTITY(1,1) NOT NULL,
[Sname] [varchar](20) NOT NULL,
[Status] [smallint] NULL,
[City] [varchar](15) NULL
)
GO
```

```
/****** Создаем таблицу деталей [P] ******/
CREATE TABLE [dbo].[P](
[Pno] [int] IDENTITY(1,1) NOT NULL,
[Pname] [varchar](20) NOT NULL,
[Color] [char](10) NULL,
[Weight] [real] NULL,
[City] [varchar](15) NULL
)
GO

CREATE TABLE [dbo].[J](
[Jno] [int] IDENTITY(1,1) NOT NULL,
[Jname] [varchar](20) NOT NULL,
[City] [varchar](15) NULL
)
GO

CREATE TABLE [dbo].[SPJ](
[Sno] [int] NOT NULL,
[Pno] [int] NOT NULL,
[Jno] [int] NOT NULL,
[Qty] [int] NULL
)
GO
```

Сценарий № 2: Изменение определений таблиц путем добавления ограничений
(сохраняем в файле, например, в SQLQuery2.sql)

```
/****** Начинаем работать в контексте созданной базы данных [dbSPJ] ******/
USE [dbSPJ]
GO

/****** Изменяем определение таблицы поставщиков [S] путем добавления ограничений первичного ключа и ключа уникальности ******/
ALTER TABLE [dbo].[S] ADD
CONSTRAINT [PK_S] PRIMARY KEY ([Sno]),
CONSTRAINT [UK_S] UNIQUE ([Sname])
GO

/****** Изменяем определение таблицы деталей [P] путем добавления ограничения первичного ключа ******/
ALTER TABLE [dbo].[P] ADD
CONSTRAINT [PK_P] PRIMARY KEY ([Pno])
GO

/****** Изменяем определение таблицы проектов [J] путем добавления ограничений ограничений первичного ключа и ключа уникальности ******/
ALTER TABLE [dbo].[J] ADD
CONSTRAINT [PK_J] PRIMARY KEY ([Jno]),
CONSTRAINT [UK_J] UNIQUE ([Jname])
GO

/****** Изменяем определение таблицы поставок [SPJ] путем добавления ограничений первичного ключа и внешних ключей ******/
ALTER TABLE [dbo].[SPJ] ADD
CONSTRAINT [PK_SP] PRIMARY KEY ( [Sno], [Pno], [Jno] ),
CONSTRAINT [FK_SP_J] FOREIGN KEY([Jno]) REFERENCES [dbo].[J] ([Jno]) ,
CONSTRAINT [FK_SP_P] FOREIGN KEY([Pno]) REFERENCES [dbo].[P] ([Pno]) ,
CONSTRAINT [FK_SP_S] FOREIGN KEY([Sno]) REFERENCES [dbo].[S] ([Sno])
GO

/****** Изменяем определения таблиц [S], [P] и [SPJ] путем добавления ограничений CHECK ******/
ALTER TABLE [dbo].[S] ADD
CONSTRAINT [Status_chk] CHECK ([Status] BETWEEN 0 AND 100)
GO
```

```
ALTER TABLE [dbo].[P] ADD
CONSTRAINT [Weight_chk] CHECK ([Weight] >= 0)
GO

ALTER TABLE [dbo].[P] ADD
CONSTRAINT    [Color_chk]    CHECK    (([Color]='Красный'    OR    [Color]='Зеленый'    OR
[Color]='Синий'))
GO

ALTER TABLE [dbo].[SPJ] ADD
CONSTRAINT [Qty_chk] CHECK ([Qty] >= 0)
GO

/****** Создаем правило для City и привязываем правило к полям City таблиц [S], [P] и
[J] ******/
CREATE RULE [dbo].[City_rule]
AS
@city IN ('Смоленск', 'Владимир', 'Рязань', 'Тверь', 'Тула', 'Калуга', 'Ярославль')
GO
EXEC sp_bindrule 'City_rule', 'dbo.S.City'
EXEC sp_bindrule 'City_rule', 'dbo.P.City'
EXEC sp_bindrule 'City_rule', 'dbo.J.City'
GO
```

Сценарий № 3: Массированное копирование данных в базу данных
(сохраняем в файле, например, в SQLQuery3.sql)

```
BULK INSERT [dbSPJ].[dbo].[S]
FROM 'F:\dbSPJ\S.txt'
WITH (CODEPAGE = 'ACP', DATAFILETYPE = 'char', FIELDTERMINATOR = '\t', ROWTERMINATOR =
'\n');
GO

BULK INSERT [dbSPJ].[dbo].[P]
FROM 'F:\dbSPJ\P.txt'
WITH (CODEPAGE = 'ACP', DATAFILETYPE = 'char', FIELDTERMINATOR = '\t', ROWTERMINATOR =
'\n');
GO

BULK INSERT [dbSPJ].[dbo].[J]
FROM 'F:\dbSPJ\J.txt'
WITH (CODEPAGE = 'ACP', DATAFILETYPE = 'char', FIELDTERMINATOR = '\t', ROWTERMINATOR =
'\n');
GO

BULK INSERT [dbSPJ].[dbo].[SPJ]
FROM 'F:\dbSPJ\SPJ.txt'
WITH (CODEPAGE = 'ACP', DATAFILETYPE = 'char', FIELDTERMINATOR = '\t', ROWTERMINATOR =
'\n');
GO
```

Примечания.

1. Файлы данных для импорта в таблицы базы данных с помощью инструкции BULK INSERT представлены в папке «Данные для учебной базы данных dbSPJ».
2. Вопросы генерации тестовых данных рассмотрены в файле «Автоматическое генерирование тестовых данных».
3. Общие сведения о сценариях и пакетах на языке SQL приведены в файле «Сценарии и пакеты».
4. Законченный пример выполнения лабораторной работы студентом содержится в папке «Пример выполнения лабораторной работы № 1».