

Лабораторная работа 2

СОЗДАНИЕ БАЗЫ ДАННЫХ В SQL SERVER MANAGEMENT STUDIO

Цель работы: научиться создавать базы данных в среде SQL Server Management Studio.

Теоретические сведения

Родоначальником серии SQL Server и его основой является язык запросов SQL. Данный язык был предложен сотрудником компании IBM Эдгаром Коддом в начале 1970-х г.г. Изначально он назывался SEQUEL (Structured English Query Language, структурированный английский язык для запросов), который впоследствии по юридическим соображениям был переименован в *SQL* (Structured Query Language, структурированный язык запросов). Официальным произношением стало [es kju:' el] — *эс-кью-эл*. Несмотря на это, даже англоязычные специалисты по-прежнему часто называют SQL *сиквел*, вместо *эс-кью-эл* (по-русски также часто говорят "эс-ку-эль"). Целью разработки было создание простого непроцедурного языка, которым мог воспользоваться любой пользователь, даже не имеющий навыков программирования. К началу 1980-х годов SQL завоевал популярность как язык реляционных систем управления базами данных (СУБД) и привлек внимание Американского национального института по стандартизации (American National Standards Institute, ANSI), который в 1986, 1989, 1992, 1999 и 2003 годах выпустил стандарты языка SQL. В 1989 году SQL был включен в стандарты международной организации по стандартизации ISO (SQL:1989), а затем были приняты и опубликованы стандарты SQL:1992, SQL:1999 и SQL:2003. В настоящее время все производители распространенных реляционных СУБД поддерживают с различной степенью соответствия стандарт SQL:2003. В основу языка SQL, используемого в SQL Server, легла разновидность языка T-SQL (Transact-SQL).

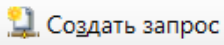
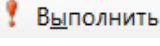
В начале 1980-х г.г. фирма IBM и в частности в то время ее подразделениями Microsoft и Sybase была создана первая версия сетевой СУБД, которая называлась SQL Server версия 1.0, для операционной системы IBM OS/2. После этого под эту операционную систему было выпущено еще 3 версии SQL Server. В середине 1980-х г.г. компании Microsoft и Sybase отделяются от фирмы IBM, и Microsoft начинает работу над своей операционной системой Windows, и вместе с компанией Sybase продолжает развитие SQL Server. В середине 1990-х г.г. Microsoft создала операционную систему Windows NT и вместе с компанией Sybase выпустила первую версию SQL Server для Windows версии 4.1.

В дальнейшем компания Sybase разорвала свои отношения с Microsoft, и Microsoft сама продолжила работу над Microsoft SQL Server 6.0. Данная

версия была предназначена для работы в операционной системе Windows NT, 95 и 98. В 1999 г. выходит версия Microsoft SQL Server 7.0, которая стала одной из самых популярных серверных СУБД в мире. В 2000 г. выходит 8-я версия Microsoft SQL Server 2000. В 2005 году появляется новая версия сервера, основанная на новой технологии .NET, а в 2008 году – её улучшенная версия – Microsoft SQL Server 2008.

В СУБД Microsoft SQL Server новую базу данных (БД) можно создать, используя стандартные команды языка T-SQL.

Для создания новой БД с помощью T-SQL необходимо вначале перейти в БД «Master». Это можно сделать либо выбором ее из выпадающего списка баз данных на панели инструментов, либо набором команды USE Master на вкладке нового запроса.

Все команды языка T-SQL набираются на вкладке нового запроса (SQLQuery). Для того чтобы создать новый запрос на панели инструментов необходимо нажать кнопку . Для выполнения команд языка T-SQL на панели инструментов необходимо нажать кнопку  или на вкладке нового запроса набрать команду GO.

В Microsoft SQL Server БД состоит из двух частей:

- файл данных – файл, имеющий расширение mdf и где находятся все таблицы и запросы;
- файл журнала транзакций - файл, имеющий расширение ldf, содержит журнал, где фиксируются все действия с БД. Данный файл предназначен для восстановления БД в случае её выхода из строя.

Для создания нового файла данных используется SQL-команда CREATE DATABASE, которая имеет следующий синтаксис:

```
CREATE DATABASE <Имя БД>
(Name=<Логическое имя>,
FileName=<Имя файла>
[Size=<Нач. размер>,)
[Maxsize=<Макс. размер>,)
[FileGrowth=<Шаг>])
[LOG ON
(Name=<Логическое имя>,
FileName=<Имя файла>
[Size=<Нач. размер>,)
[Maxsize=<Макс. размер >,)
[FileGrowth=<Шаг>])
```

Здесь Имя БД – имя создаваемой БД; Логическое имя – определяет логическое имя файла данных БД, по которому происходит обращение к файлу данных; Имя файла – определяет полный путь к файлу данных; Нач. размер – начальный размер файла данных в МБ; Макс. размер – максимальный размер файла данных в МБ; Шаг – шаг увеличения файла данных, либо в МБ либо в %.

Параметры в разделе LOG ON аналогичны параметрам в разделе CREATE DATABASE. Однако они определяют параметры журнала транзакций.

Пример: Создать БД «Artworks», расположенную в файле «D:\Artworks.mdf» и имеющую начальный размер файла данных – 3 МБ, максимальный размер файла данных – 100 МБ и шаг увеличения файла данных, равный 1 МБ. Файл журнала транзакций данной БД имеет имя «ArtworksLog» и расположен в файле «D:\Artworks.ldf». Данный файл имеет начальный размер, равный 1 МБ, максимальный размер, равный 20 МБ и шаг увеличения, равный 1 МБ.

```
CREATE DATABASE Artworks
ON
(Name = Artworks,
FileName = 'D:\ Artworks.mdf',
Size = 3MB,
Maxsize = 100MB,
FileGrowth= 1MB)
LOG ON
(Name = Artworks Log,
FileName = 'D:\ Artworks.ldf',
Size = 1MB,
Maxsize = 20MB,
FileGrowth = 1MB)
```

В языке запросов T-SQL с БД возможны следующие действия:

1. Отображение сведений о БД: EXEC SP_HELPDB <Имя БД>;
2. Изменение параметров БД: EXEC SP_DBOPTION <Имя БД>, <Параметр>, <Значение>;
3. Добавление новых файлов, удаление файлов и переименование файлов, входящих в БД:

```
ALTER DATABASE <Имя БД>
ADD FILE (<Параметры>) |
REMOVE FILE <Логическое имя файла> |
MODIFY FILE (<Параметры>)
```

где, раздел ADD FILE – добавляет файл, REMOVE FILE – удаляет, а раздел MODIFY FILE – изменяет параметры файла;

4. Сжатие всей БД: DBCC SHRINKDATABASE <Имя БД>;
5. Сжатие конкретного файла БД: DBCC SHRINKFILE <Логическое имя файла>;
6. Переименование БД: EXEC SP_RENAMEDB <Имя БД>, <Новое имя БД>;
7. Удаление БД: DROP DATABASE <Имя БД>.

Вышеперечисленные команды используют следующие параметры:

- <Имя БД> – имя БД с которой производится действие;
- <Параметр> – изменяемый параметр;
- <Значение> – новое значение изменяемого параметра;
- <Параметры> – параметры файла БД, аналогичные параметрам, используемым в команде CREATE DATABASE;
- <Логическое имя файла> – логическое имя файла, входящего в БД;
- <Новое имя БД> – новое имя БД.

Пример выполнения лабораторной работы

Для начала запустим среду разработки «SQL Server Management Studio». Для этого в меню «Пуск» выбираем пункт «Все программы\Microsoft SQL Server 2008\Среда SQL Server Management Studio» (рис. 1)

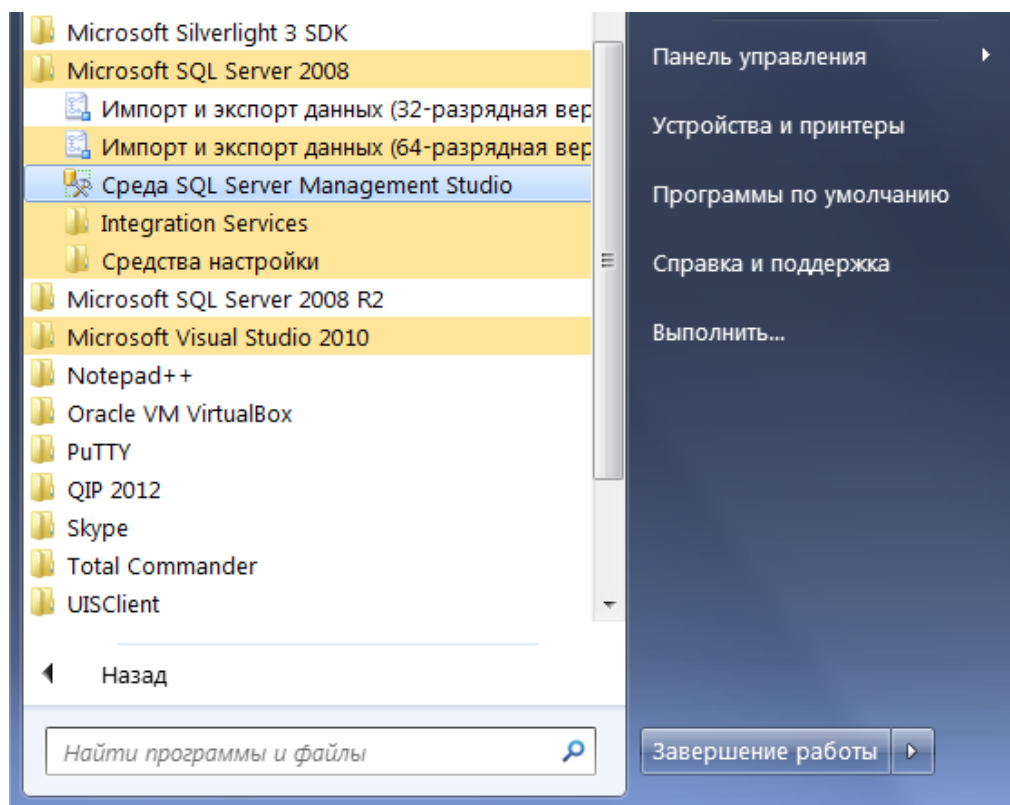


Рис. 1

После запуска среды разработки появится окно подключения к серверу «Соединение с сервером» (рис. 2).

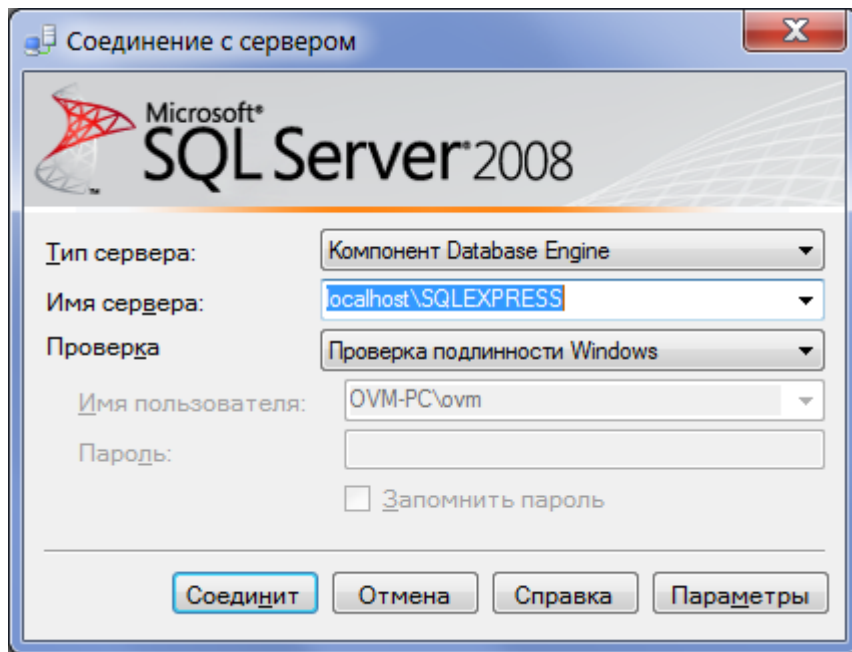


Рис. 2

В этом окне необходимо нажать кнопку «Соединить».

После нажатия кнопки «Соединить» появится окно среды разработки «SQL Server Management Studio» (рис. 3).

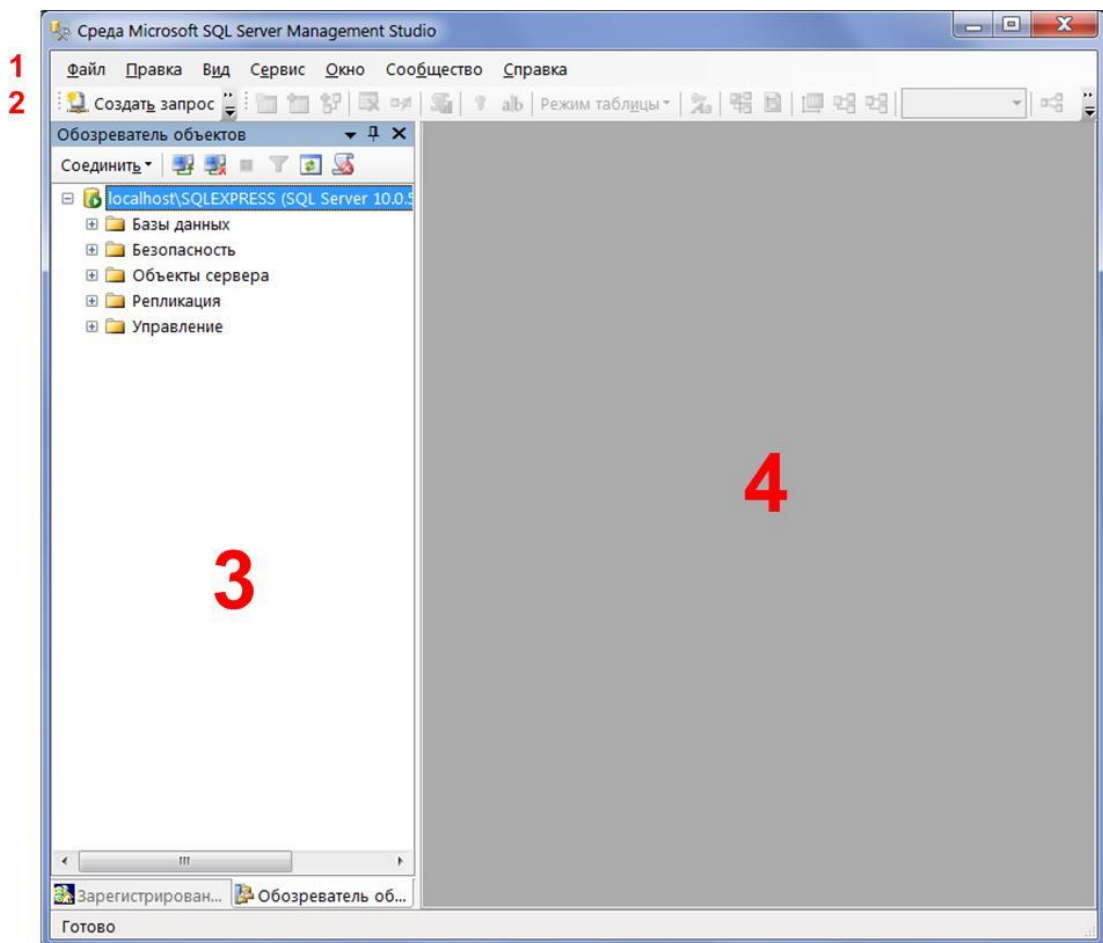


Рис. 3

Данное окно имеет следующую структуру:

1. Оконное меню – содержит полный набор команд для управления сервером и выполнения различных операций.
2. Панель инструментов – содержит кнопки для выполнения наиболее часто производимых операций. Внешний вид данной панели зависит от выполняемой операции.
3. Панель «Обозреватель объектов» – обозреватель объектов. Обозреватель объектов – это панель с древовидной структурой, отображающая все объекты сервера, а также позволяющая производить различные операции, как с самим сервером, так и с БД. Обозреватель объектов является основным инструментом для разработки БД.
4. Рабочая область. В рабочей области производятся все действия с БД, а также отображается её содержимое.

В обозревателе объектов сами объекты находятся в папках. Чтобы открыть папку необходимо щёлкнуть по знаку «+» слева от изображения папки.

Нажатие кнопки  «Создать запрос» приводит к открытию окна запросов (рис. 4).

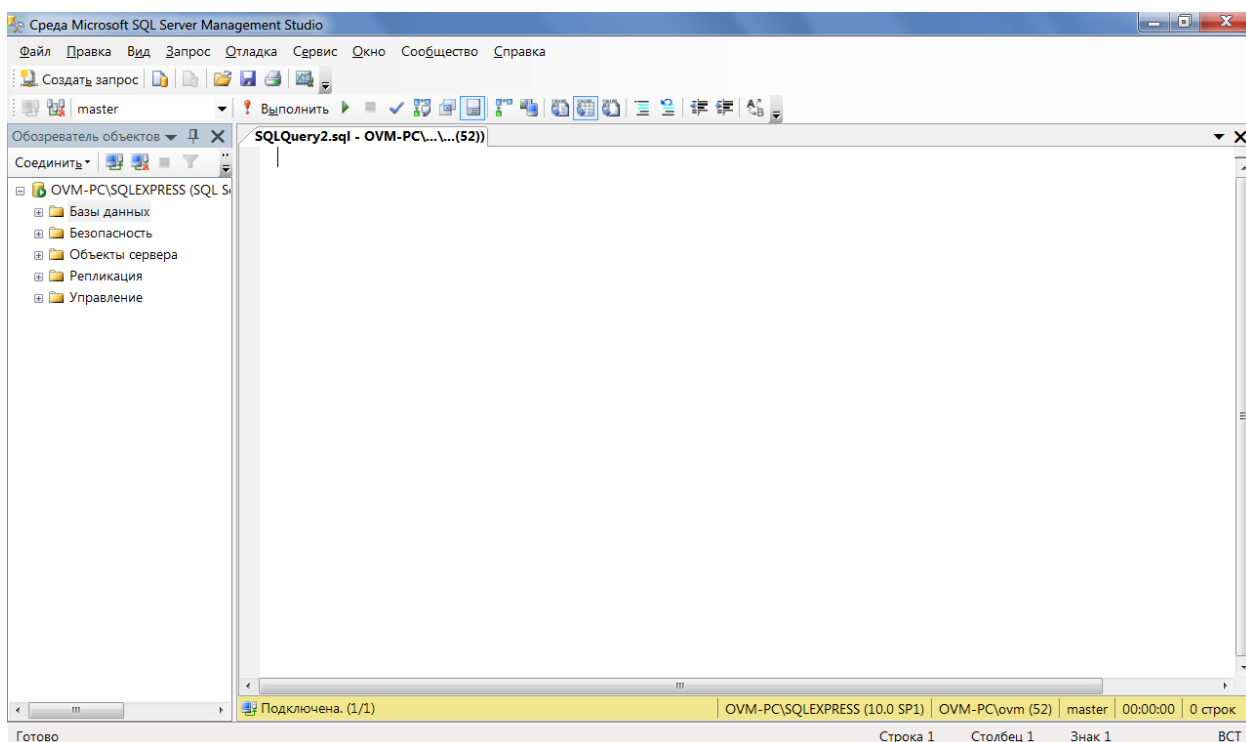



Рис. 4

Для выполнения запроса, введенного в этом окне, нужно нажать на кнопку .

Выполним создание базы данных «Artworks» в среде разработки SQL Server Management Studio. В новом окне запросов введем запрос на создание БД:

```
IF DB_ID('Artworks') IS NULL
    CREATE DATABASE Artworks;
```

Если базы данных с именем **Artworks** не существует, приведенный код создаст новую БД. Функция **DB_ID** принимает имя базы данных в качестве входного параметра и возвращает внутренний ID базы данных. Если БД с именем входного параметра не существует, функция возвращает значение NULL. Это простой способ проверить наличие заданной БД.

В инструкции **CREATE DATABASE** используются установочные параметры файла для задания его местоположения и начального размера.

После выполнения данного запроса в окне среды разработки SQL Server Management Studio на панели обозревателя объектов в папке «Базы данных» появится новая БД **Artworks** (рис. 5).

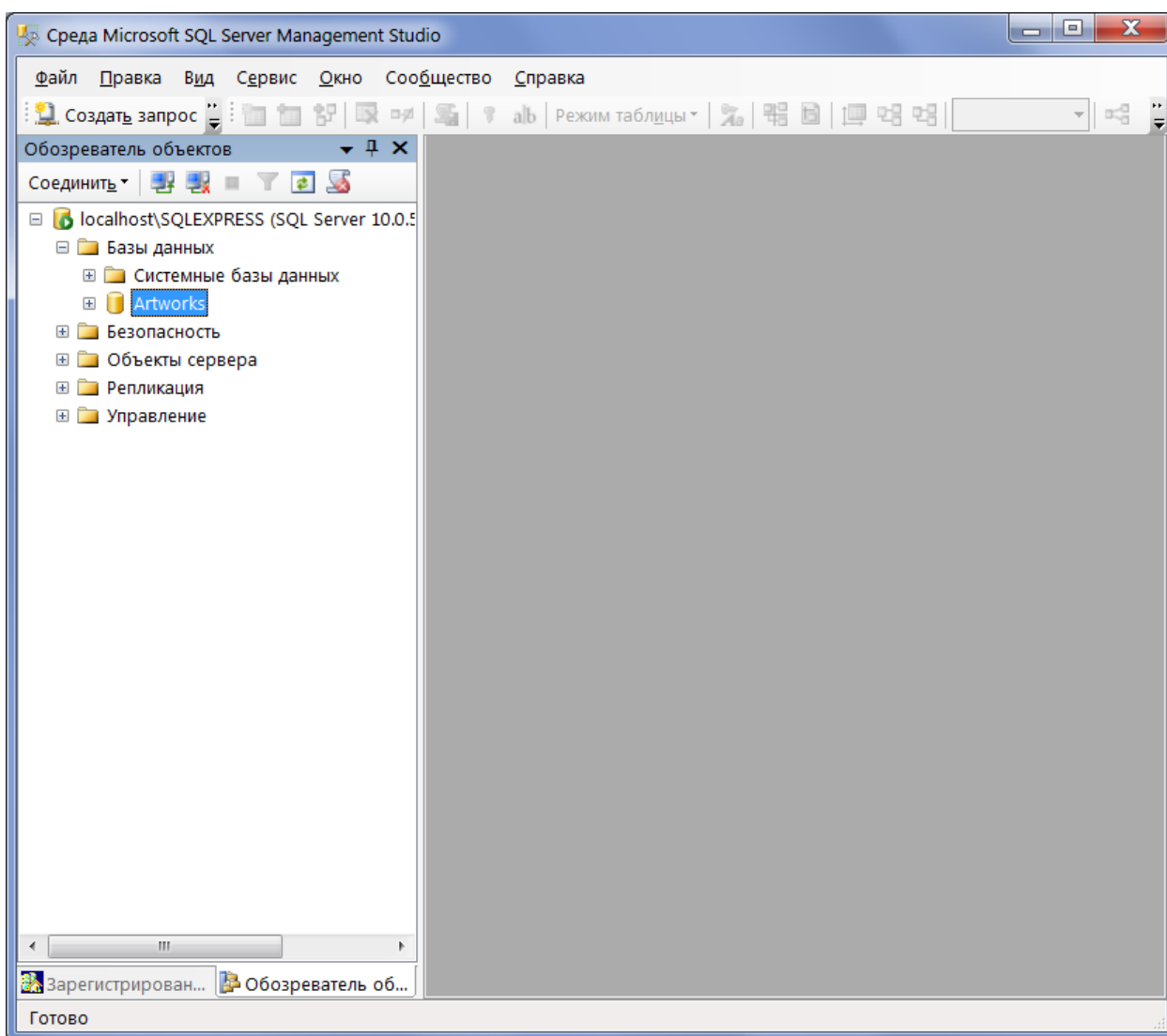


Рис. 5

Вся информация в базе данных хранится в таблицах, которые представляют собой средство хранения данных. Таблицы состоят из строк или записей и столбцов или полей. Каждое поле имеет три характеристики:

1. Имя поля – используется для обращения к полю;
2. Значение поля – определяет информацию, хранимую в поле;

3. Тип данных поля – определяет, какой вид информации можно хранить в поле.

В SQL сервер используются следующие типы данных:

- **Двоичные типы данных**, которые содержат последовательности нулей и единиц: 1) `binary(n)` – двоичный тип фиксированной длины размером в n байт, где n — значение от 1 до 8000; размер при хранении составляет n байт; 2) `varbinary(n)` – двоичный тип с переменной длиной, n может иметь значение от 1 до 8000, **max** указывает, что максимальный размер при хранении составляет $2^{31}-1$ байт;
- **Целочисленные типы данных** – типы данных для хранения целых чисел (в скобках указан диапазон значений типа данных): `tinyint` (0..255), `smallint` (-32768..+32767), `int` ($-2^{31}..+(2^{31}-1)$), `bigint` ($-2^{63}..+(2^{63}-1)$);
- **Типы данных для хранения чисел с плавающей запятой**: `real` занимает в памяти 4 байта; `float(n)`, где n — это количество битов, используемых для хранения мантиссы числа в формате **float** при экспоненциальном представлении, определяет точность данных и размер для хранения; значение параметра n должно лежать в пределах от 1 до 53; значением по умолчанию для параметра n является 53;
- **Типы данных для хранения чисел с фиксированной точностью и масштабом**: `decimal(p, s)` и `numeric(p, s)`, где p (точность) – максимальное количество десятичных разрядов числа (как слева, так и справа от десятичной запятой). Точность должна быть значением в диапазоне от 1 до 38. По умолчанию это значение равно 18. s (масштаб) – максимальное количество десятичных разрядов числа справа от десятичной запятой. Масштаб может принимать значение от 0 до p и может быть указан только совместно с точностью. По умолчанию масштаб принимает значение 0; поэтому $0 \leq s \leq p$;
- **Символьные типы данных**: `char(n)` – строковые данные фиксированной длины не в Юникоде, аргумент n определяет длину строки и должен иметь значение от 1 до 8000, размер при хранении составляет n байт; `varchar(n | max)` – строковые данные переменной длины не в Юникоде, аргумент n определяет длину строки и должен иметь значение от 1 до 8000, значение **max** указывает, что максимальный размер при хранении составляет $2^{31}-1$ байт (2 ГБ); `text` – данные переменной длины не в Юникоде в кодировке сервера и с максимальной длиной строки $2^{31}-1$;
- **Специальные типы данных**: `bit` – целочисленный тип данных, который может принимать значения 1, 0 или NULL; `image` – тип данных для хранения рисунка размером до 2ГБ;
- **Типы данных даты и времени**: `date` (от 01.01.0001 до 31.12.9999); `datetime` (диапазон даты – от 01.01.1753 до 31.12.1999, диапазон времени – от 00:00:00 до 23:59:59,997); `smalldatetime` (диапазон даты –

от 01.01.1900 до 6.06.2079, диапазон времени – от 00:00:00 до 23:59:59);
time (от 00:00:00.0000000 до 23:59:59.9999999);

- **Денежные типы данных для хранения финансовой информации:** money (8 байт) и smallmoney (4 байта) – типы данных, представляющие денежные (валютные) значения.

Для создания таблиц в SQL Server в первую очередь необходимо сделать активной ту БД, в которой создается таблица. Для этого в новом запросе можно набрать команду: USE <Имя БД>, либо на панели инструментов необходимо выбрать в выпадающем списке рабочую БД. После выбора БД можно создавать таблицы.

Таблицы создаются командой

```
CREATE TABLE table_name  
( { <column_definition> } )  
[ ; ]
```

```
<column_definition> ::= column_name <data_type> [ NULL  
| NOT NULL ] [DEFAULT constant_expression ] | [  
IDENTITY [ ( seed,increment ) ] ] [ <column_constraint>  
[ ...n ] ]
```

```
<column_constraint> ::= [ CONSTRAINT constraint_name ]  
{ PRIMARY KEY | UNIQUE }
```

Здесь *table_name* – имя таблицы; *column_name* – имя столбца в таблице; *data_type* – тип данных для столбца; *IDENTITY* указывает, что новый столбец является столбцом идентификаторов, при этом *seed* – значение, используемое для самой первой строки, загружаемой в таблицу, *increment* – значение приращения, добавляемое к значению идентификатора предыдущей загруженной строки; *CONSTRAINT* – необязательное ключевое слово, указывающее на начало определения ограничения, *constraint_name* – имя ограничения; *NULL | NOT NULL* определяет, допустимы ли для столбца значения *NULL*; *PRIMARY KEY* – ограничение, которое определяет столбец первичным ключом таблицы.

Если имя поля содержит пробел, то оно заключается в квадратные скобки.

Пример: Создать таблицу «Artworks», содержащую поля:

- а) код произведения (**ArtworkId**);
- б) название произведения (**Title**);
- в) жанр (**Genre**);
- г) средства создания (**Tools**);
- д) код автора (**AuthorId**);
- е) дата создания (**CreatDate**);
- ж) цена (**Price**);

3) отдел хранения (**DepId**).

SQL-запрос для создания этой таблицы имеет следующий вид:

```
USE Artworks;
IF OBJECT_ID('dbo.Artworks', 'U') IS NOT NULL
    DROP TABLE dbo.Artworks;

CREATE TABLE dbo.Artworks
(
    ArtworkId BIGINT IDENTITY(1,1) CONSTRAINT
                                                PK_Artworks PRIMARY KEY,
    Title VARCHAR(100) NULL,
    Genre VARCHAR (50) NULL,
    Tools VARCHAR (50) NULL,
    AuthorId BIGINT NULL,
    CreatDate DATE NULL,
    Price MONEY NULL,
    DepId INT NOT NULL
);
```

Инструкция **USE** изменяет текущую связь с БД на связь с **Artworks**. Включение этой инструкции в сценарии создания объектов очень важно, т.к. гарантирует создание объектов в требуемой БД.

Инструкция **IF** запускает функцию **OBJECT_ID**, которая в качестве входных параметров принимает имя объекта и его тип. Тип **'U'** представляет пользовательские таблицы. Данная функция возвращает внутренний ID объекта, если объект с заданным именем и типом уже существует, и значение **NULL** в противном случае.

При создании таблицы используется схема с именем **dbo**, которая создается автоматически в каждой базе данных и используется как схема по умолчанию. Если опустить имя схемы при создании таблицы, то SQL Server свяжет с таблицей схему, используемую по умолчанию для имени пользователя, выполняющего программный код.

Для каждого атрибута сущности **Artworks** задается его имя, тип и допустимость значений **NULL**.

Для столбца **ArtworkId** определено ограничение в виде первичного ключа (**PK_Artworks**), при этом значения **ArtworkId** будут начинаться с 1 и увеличиваться при каждом добавлении новых строк в таблицу тоже на 1 (**IDENTITY(1,1)**).

Аналогично создаются и другие таблицы БД **Artworks : Authors, Employees, Departments**. SQL-запросы для создания этих таблиц приведены ниже.

```

USE Artworks;
IF OBJECT_ID('dbo.Authors', 'U') IS NOT NULL
    DROP TABLE dbo.Authors;

CREATE TABLE dbo.Authors
(
    AuthorId BIGINT IDENTITY(1,1) CONSTRAINT
                                     PK_Authors PRIMARY KEY,
    Lastname VARCHAR(25) NOT NULL,
    Firstname VARCHAR (25) NOT NULL,
    Middlename VARCHAR (25) NULL,
    DateOfBirth DATE NULL,
    DateOfDeath DATE NULL,
    Country VARCHAR(25) NULL
);


IF OBJECT_ID('dbo.Employees', 'U') IS NOT NULL
    DROP TABLE dbo.Employees;

CREATE TABLE dbo.Employees
(
    EmpId BIGINT IDENTITY(1,1) CONSTRAINT
                                     PK_Employees PRIMARY KEY,
    Lastname VARCHAR(25) NOT NULL,
    Firstname VARCHAR (25) NOT NULL,
    Middlename VARCHAR (25) NOT NULL,
    Position VARCHAR (25) NULL,
    Salary MONEY NULL,
    BeginDate DATE NOT NULL,
    EndDate DATE NULL,
    DepId INT NULL
);

IF OBJECT_ID('dbo.Departments', 'U') IS NOT NULL
    DROP TABLE dbo.Departments;

CREATE TABLE dbo.Departments
(
    DepId INT IDENTITY(1,1) CONSTRAINT
                                     PK_Departments PRIMARY KEY,
    Name VARCHAR(25) NOT NULL
);

```

Для выполнения запросов, введенных в среде SQL Server Management Studio, нужно нажать на кнопку  Выполнить.

Для того чтобы обеспечить ссылочную целостность в БД **Artworks** нужно добавить в созданные таблицы ограничение по внешним ключам.

Таблицы **Artworks** и **Authors** нужно связать по столбцу **AuthorId**, а таблицы **Artworks** и **Departments** – по столбцу **DepId**. Аналогично должны быть связаны между собой таблицы **Employees** и **Departments**.

```
USE Artworks;
```

```
ALTER TABLE dbo.Artworks
  ADD CONSTRAINT FK_Artw_Auth
  FOREIGN KEY (AuthorId)
  REFERENCES Authors (AuthorId);
```

```
ALTER TABLE dbo.Artworks
  ADD CONSTRAINT FK_Artw_Dep
  FOREIGN KEY (DepId)
  REFERENCES Departments (DepId);
```

```
ALTER TABLE dbo.Employees
  ADD CONSTRAINT FK_Emp_Dep
  FOREIGN KEY (DepId)
  REFERENCES Departments (DepId);
```

Теперь, когда установлены связи между всеми таблицами, можно создать диаграмму, описывающую эти взаимосвязи.

Для создания диаграммы нужно щелкнуть правой кнопкой мыши на элементе БД «Диаграммы баз данных» и в открывшемся контекстном меню выбрать пункт «Создать диаграмму базы данных» (рис. 6).

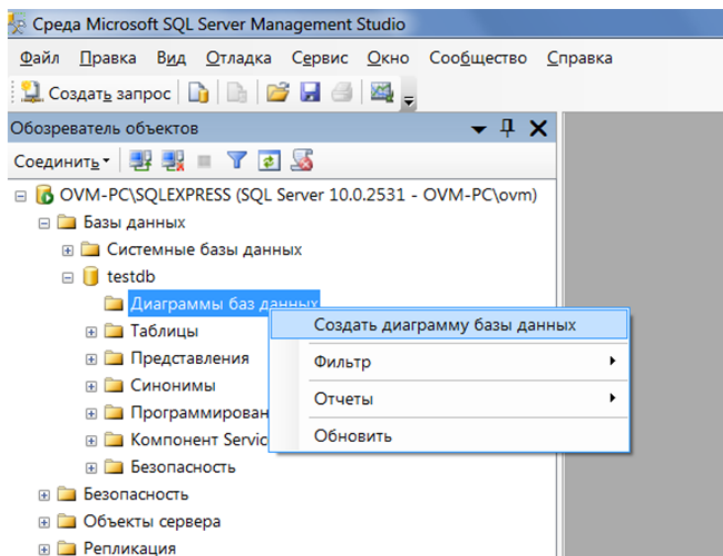


Рис. 6

В открывшемся диалоговом окне нужно выбрать таблицы, которые будут добавлены на диаграмму (рис. 7).

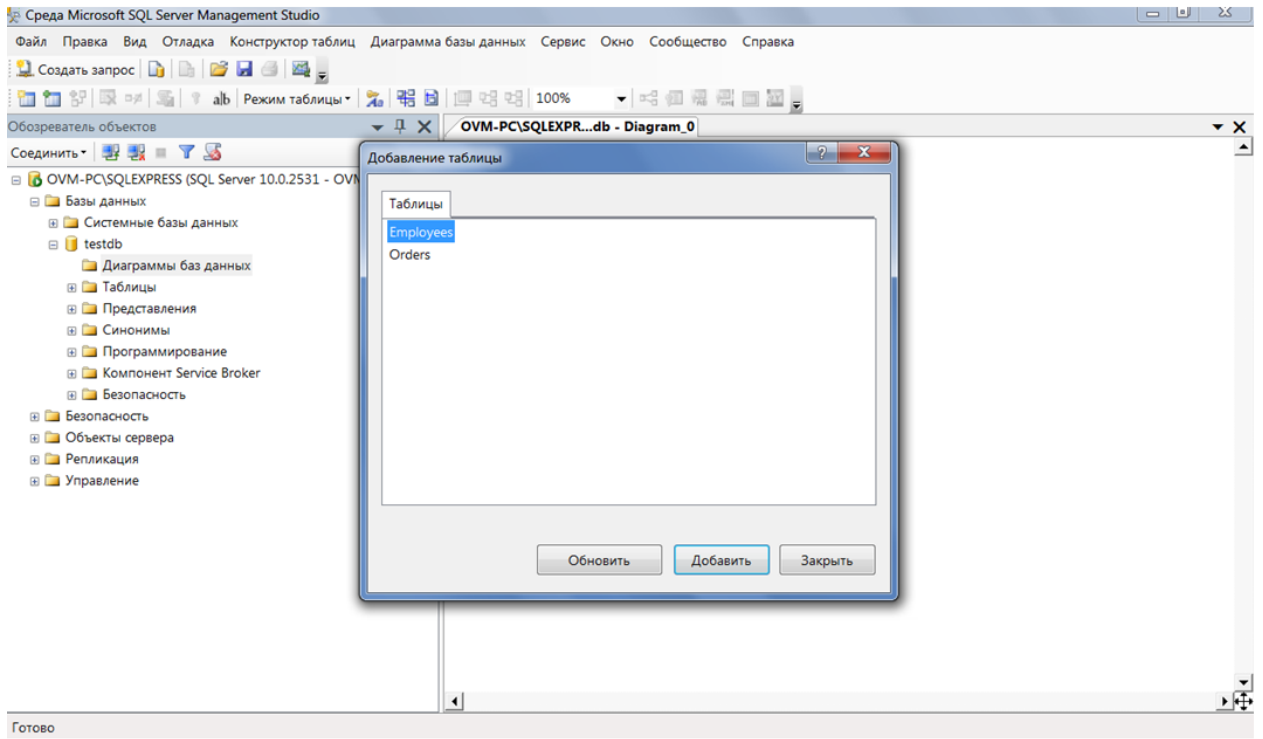


Рис. 7

В результате будет получена диаграмма следующего вида:

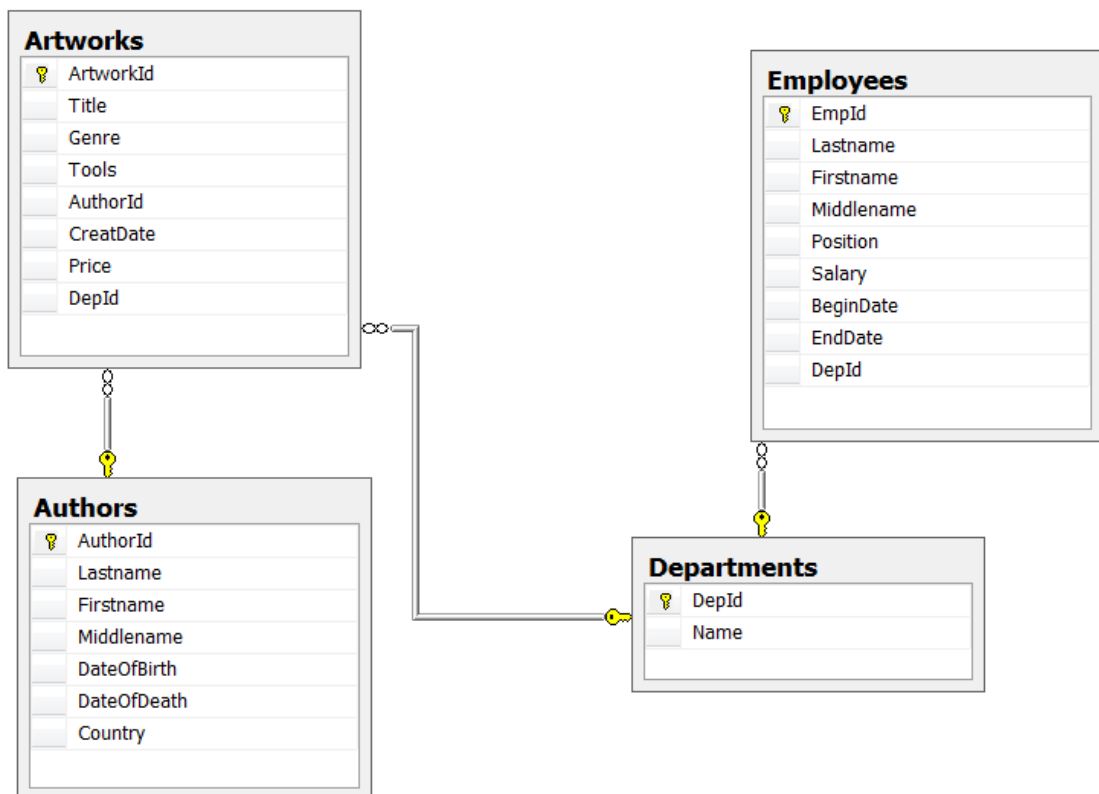


Рис. 8

Задание

- 1) Создать базу данных и ее таблицы в соответствии с вариантом. Выполнить задание с помощью стандартных команд языка T-SQL.
- 2) Создать диаграмму БД средствами среды SQL Server Management Studio.