Редактор запросов

Чтобы открыть панель редактора запросов Query Editor (Редактор запросов), на панели инструментов среды SQL Server Management Studio нажмите кнопку New Query (Создать запрос).

Редактор запросов можно использовать для выполнения следующих задач:

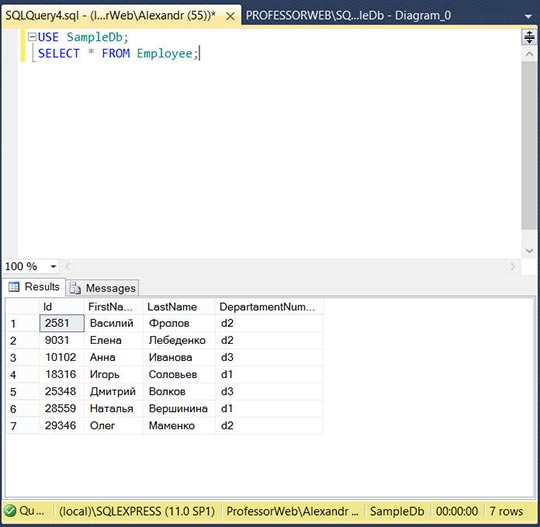
создания и выполнения инструкций языка Transact-SQL;

сохранения созданных инструкций языка Transact-SQL в файл;

создания и анализирования планов выполнения общих запросов;

графического иллюстрирования плана выполнения выбранного запроса.

Редактор запросов содержит встроенный текстовый редактор и панель инструментов с набором кнопок для разных действий. Главное окно редактора запросов разделено по горизонтали на панель запросов (вверху) и панель результатов (внизу). Инструкции Transact-SQL (т.е. запросы) для исполнения вводятся в верхнюю панель, а результаты обработки системой этих запросов отображаются в нижней панели. На рисунке ниже показан пример ввода запроса в редактор запросов и результатов выполнения этого запроса:



Создание.

Как и у многих инструкций языка Transact-SQL, у инструкции CREATE DATABASE имеется обязательный параметр: имя базы данных. Кроме этого, у инструкции CREATE DATABASE имеется ряд необязательных параметров, таких как расположение на диске, где требуется хранить файлы базы данных. При выполнении инструкции CREATE DATABASE без дополнительных параметров, для многих из них SQL Server использует значения по умолчанию.

**Задание.**

Создать базу данных с именем Группа\_Фамилия, пример КН101\_Иванов.

**Создание базы данных**

1. В окне редактора запросов введите, но не выполняйте, следующий код:

CREATE DATABASE TestData

GO

1. С помощью указателя выделите слова CREATE DATABASE и нажмите клавишу F1. Должен открыться раздел CREATE DATABASE электронной документации по SQL Server. Таким же способом можно найти полный синтаксис инструкции CREATE DATABASE и других инструкций, используемых в данном учебнике.
2. В редакторе запросов надо нажать клавишу F5, чтобы выполнить инструкцию и создать базу данных с именем TestData.

При создании базы данных сервер SQL Server создает копию базы данных **model** и присваивает ей указанное имя базы данных. Эта операция обычно занимается несколько секунд, если только с помощью дополнительного параметра не указан большой исходный размер базы данных.

|  |
| --- |
| **Примечание** |
| Когда в одном пакете представлено несколько инструкций, они разделяются с помощью ключевого слова GO. Ключевое слово GO является необязательным, если в пакете содержится только одна инструкция.    **Создание таблицы.**  Чтобы создать таблицу, нужно указать имя таблицы, имена и типы данных для каждого столбца таблицы. Также рекомендуется указывать, допускаются ли значения NULL для каждого из столбцов.  Большинство таблиц имеют первичный ключ, состоящий из одной или нескольких столбцов таблицы. Первичный ключ всегда уникален. Компонент Database Engine потребует выполнения условия неповторения значения первичного ключа в таблице..   |  | | --- | | **Примечание** | | Компонент Database Engine может быть установлен с учетом регистра и без учета регистра. Если компонент Database Engine установлен с учетом регистра, имена объектов должны иметь одно и тоже имя. Например, таблица с именем OrderData будет отличаться от таблицы ORDERDATA. Если компонент Database Engine установлен без учета регистра, эти два имени таблицы будут рассматриваться как одна таблица, то есть имя может быть использовано только один раз. |  1. **Переключение соединения редактора запросов на базу данных TestData**   В окне редактора запросов надо ввести и выполнить следующий код, чтобы изменить соединение на базу данных TestData.  USE TestData  GO  **Переключитесь на вашу базу данных.**   1. **Создание таблицы**   В окне редактора запросов **введите и выполните** следующий код, чтобы создать простую таблицу Products. Столбцы таблицы имеют имена ProductID, ProductName, Price и ProductDescription. Столбец ProductID является первичным ключом таблицы. Типы данных int, varchar(25), money и text. Только столбцы Price и ProductionDescription могут быть пустыми при вставке или изменении строки. Данная инструкция содержит необязательный элемент (dbo.), называемый схемой. Схема — это объект базы данных, к которому принадлежит таблица. Если вы являетесь администратором, схемой по умолчанию будет схема dbo. Схема dbo означает владельца базы данных.  CREATE TABLE dbo.Products     (ProductID int PRIMARY KEY NOT NULL,      ProductName varchar(25) NOT NULL,      Price money NULL,      ProductDescription text NULL)  GO   1. ПРОВЕРЯЕМ, СУЩЕСТВУЕТ ЛИ В БАЗЕ ДАННЫХ ТАБЛИЦА   */\*ПРОВЕРЯЕМ, СУЩЕСТВУЕТ ЛИ В БАЗЕ ДАННЫХ*  *[ИМЯ БАЗЫ] И СХЕМЕ С ИМЕНЕМ Фамилия ТАБЛИЦА ut\_students ЕСЛИ ДА,*  *ТО ПРЕДВАРИТЕЛЬНО УДАЛЯЕМ ЕЕ ИЗ БАЗЫ И СХЕМЫ.*  *ЕСЛИ ВЫ УАЛЯЕТЕ ВСЮ БАЗУ ЦЕЛИКОМ - ЭТА ЧАСТЬ СКРИПТА НЕ НУЖНА \*/*  *IF OBJECT\_ID('[ИМЯ БАЗЫ].Фамилия.ut\_students', 'U') IS NOT NULL*  *DROP TABLE [ИМЯ БАЗЫ].Фамилия.ut\_students*  *GO*  **По приведенному примеру.**  IF OBJECT\_ID('[ИМЯ БАЗЫ].dbo.Products', 'U') IS NOT NULL  DROP TABLE [ИМЯ БАЗЫ].dbo.Products  GO  CREATE TABLE dbo.Products     (ProductID int PRIMARY KEY NOT NULL,      ProductName varchar(25) NOT NULL,      Price money NULL,      ProductDescription text NULL)  GO  **Добавление нового вычисляемого столбца**  Вычисляемый столбец представляет собой виртуальный столбец, физически не хранящийся в таблице, если для него не установлен признак PERSISTED. В выражении вычисляемого столбца для вычисления значения могут использоваться данные из других столбцов.  **Добавление вычисляемого столбца при создании таблицы**  CREATE TABLE dbo.Prod  (  ProductID int IDENTITY (1,1) NOT NULL  , QtyAvailable smallint  , UnitPrice money  , InventoryValue AS QtyAvailable \* UnitPrice  )  **Добавление нового вычисляемого столбца в существующую таблицу**  ALTER TABLE dbo.Prod ADD RetailValue AS (QtyAvailable \* UnitPrice \* 1.5)   1. **Чтение данных из таблицы.**   Для чтения данных в таблице используется инструкция SELECT. Инструкция SELECT является одной из наиболее важных инструкций Transact-SQL, и для нее существует много разновидностей синтаксиса.  Чтобы выбрать все столбцы в таблице, можно использовать звездочку. Такой способ часто используется в нерегламентированных запросах. Необходимо предоставить список всех столбцов в постоянном коде, чтобы инструкция возвращала нужные столбцы, даже если какой-то столбец будет добавлен в таблицу позднее.  -- Returns all columns in the table  -- Does not use the optional schema, dbo  SELECT \* FROM Products  GO  **Вставка данных в таблицу и их обновление.**  После создания таблицы **Products** можно вставлять данные в таблицу с помощью инструкции INSERT. После вставки данных содержимое строки изменяется с помощью инструкции UPDATE. Предложение WHERE предназначено для ограничения числа строк, изменяемых в процессе выполнения инструкции UPDATE до одной строки. Чтобы ввести следующие данные, потребуется четыре инструкции.   |  |  |  |  | | --- | --- | --- | --- | | **ProductID** | **ProductName** | **Price** | **ProductDescription** | | 1 | Clamp | 12.48 | Workbench clamp | | 50 | Screwdriver | 3.17 | Flat head | | 75 | Tire Bar |  | Tool for changing tires. | | 3000 | 3mm Bracket | .52 |  |   Базовый синтаксис: INSERT, имя таблицы, список столбцов, VALUES, а затем список добавляемых значений. Два дефиса в начале строки означают, что строка является примечанием и текст не будет обрабатываться компилятором. В этом случае примечание описывает возможные варианты синтаксиса.   1. **Вставка данных в таблицу**   Выполните следующую инструкцию, чтобы добавить строку в таблицу Products, которая была создана в предыдущей задаче. Далее представлен базовый синтаксис.  -- Standard syntax  INSERT dbo.Products (ProductID, ProductName, Price, ProductDescription)      VALUES (1, 'Clamp', 12.48, 'Workbench clamp')  GO    В следующей инструкции показано, как можно изменить порядок, в котором приведены параметры, изменив расположение ProductID и ProductName одновременно как в списке полей (в круглых скобках), так и в списке значений.  -- Changing the order of the columns  INSERT dbo.Products (ProductName, ProductID, Price, ProductDescription)      VALUES ('Screwdriver', 50, 3.17, 'Flat head')  GO    Следующая инструкция показывает, что имена столбцов перечислять не обязательно, если значения перечислены в нужном порядке. Этот синтаксис является общепринятым, но его не рекомендуется использовать, так как он усложняет понимание кода. Для столбца Price указано значение NULL, так как цена для этого продукта еще не известна.  -- Skipping the column list, but keeping the values in order  INSERT dbo.Products      VALUES (75, 'Tire Bar', NULL, 'Tool for changing tires.')  GO    Имя схемы указывать не обязательно, пока доступ и изменение таблицы осуществляются с помощью схемы по умолчанию. Поскольку в столбце ProductDescription разрешены значения NULL и значение для столбца не приведено, имя и значение столбца ProductDescription в инструкции могут быть полностью опущены.  -- Dropping the optional dbo and dropping the ProductDescription column  INSERT Products (ProductID, ProductName, Price)      VALUES (3000, '3mm Bracket', .52)  GO  Помимо вставки одной записи, SQL позволяет выполнить запрос для добавления сразу группы строк в таблицу. Такой запрос будет выполняться гораздо быстрее, чем последовательное добавление по одной записи.  Этот запрос выглядит подобно запросу добавления одной записи. Отличие только в том, что указывается несколько групп данных для вставки в таблицу разделенных запятой:  INSERT  таблица(список полей)  VALUES  (список добавляемых значений 1),  (список добавляемых значений 2),  ...  (список добавляемых значений N)  Каждая из таких групп указывается в отдельных скобках.  Например, вставка товаров в таблицу:  INSERT Products (ProductID, ProductName, Price)      VALUES (3, '5mm Bracket', 1.52),  (4, '7mm Bracket', 1.55),  (5, '9mm Bracket', 1.62)  GO  Такой запрос добавит в таблицу три записи с соответствующими значениями.  Все остальные правила, такие как порядок указания или обязательность полей сохраняются так же, как и для вставки одной записи в таблицу БД.   1. **Обновление таблицы продуктов**   Введите и выполните следующую инструкцию UPDATE, чтобы изменить значение ProductName второго продукта со значения Screwdriver на значение Flat Head Screwdriver.  UPDATE dbo.Products      SET ProductName = 'Flat Head Screwdriver'      WHERE ProductID = 50  GO   1. **Чтение определенных данных из таблицы.**   Чтобы прочитать данные из таблицы Products, введите и выполните следующие инструкции.  -- The basic syntax for reading data from a single table  SELECT ProductID, ProductName, Price, ProductDescription      FROM dbo.Products  GO    Если нет необходимости возвращать определенные столбцы, их можно опустить. Столбцы возвращаются в том порядке, в котором они перечислены.  -- Returns only two of the columns from the table  SELECT ProductName, Price      FROM dbo.Products  GO    Чтобы ограничить количество строк, возвращаемых пользователю, используйте предложение WHERE.  -- Returns only two of the records in the table  SELECT ProductID, ProductName, Price, ProductDescription      FROM dbo.Products      WHERE ProductID < 60  GO    Можно работать со значениями столбцов, по мере того как столбцы возвращаются. В следующем примере выполняется математическая операция над столбцом Price. Столбцы, изменяемые подобным образом, не имеют имени, если только имя не указывается с использованием ключевого слова AS.  -- Returns ProductName and the Price including a 7% tax  -- Provides the name CustomerPays for the calculated column  SELECT ProductName, Price \* 1.07 AS CustomerPays      FROM dbo.Products  GO  Оператор **ORDER BY** позволяет отсортировать извлекаемые значения по определенному столбцу:  В данном случае строки сортируются по возрастанию значения столбца ProductName:  SELECT \*  FROM Products  ORDER BY ProductName  Сортировку также можно проводить по псевдониму столбца, который определяется с помощью оператора AS.  По умолчанию применяется сортировка по возрастанию. С помощью дополнительного оператора **DESC** можно задать сортировку по убыванию. |