### Лабораторная работа №1 ,2

#### Тема: Создание базы данных в MS SQL SERVER 2014. Типы данных.

**SQL Server Management Studio** — утилита из Microsoft SQL Server 2005 и более поздних версий для конфигурирования, управления и администрирования всех компонентов Microsoft SQL Server. Утилита включает скриптовый редактор и графическую программу, которая работает с объектами и настройками сервера.

Главным инструментом SQL Server Management Studio является Object Explorer, который позволяет пользователю просматривать, извлекать, и полностью управлять объектами сервера.

**Компонент Database Engine** представляет собой основную службу для хранения, обработки и обеспечения безопасности данных. Этот компонент обеспечивает управляемый доступ к ресурсам и быструю обработку транзакций, что позволяет использовать его даже в самых требовательных корпоративных приложениях обработки данных. Кроме того, компонент Database Engine обеспечивает разностороннюю поддержку высокого уровня доступности.

#### Открытие среды SQL Server Management Studio

1. В меню **Пуск** последовательно выберите пункты **Все программы**, **Microsoft SQL Server 2005** и выберите команду **Среда SQL Server Management Studio**.
2. В диалогом окне **Соединение с сервером** подтвердите заданные по умолчанию параметры и нажмите кнопку **Подключиться**. Для соединения необходимо, чтобы поле **Имя сервера** содержало имя компьютера, на котором установлен SQL Server. Если компонент Database Engine представляет собой именованный экземпляр, то поле «Имя сервера» должно также содержать имя экземпляра в формате <*имя\_компьютера*>\<*имя\_экземпляра*>.

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. В этом учебнике используются лишь некоторые дополнительные синтаксические параметры.

#### 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 2005. Таким же способом можно найти полный синтаксис инструкции CREATE DATABASE и других инструкций, используемых в данном учебнике.
2. В редакторе запросов нажмите 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.

Список типов данных и ссылки на их описание см. в разделе Типы данных (Transact-SQL).

|  |
| --- |
| **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** не будет аналогична таблице **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. |

Перед созданием таблицы выполните команду USE, чтобы изменить контекст базы данных на базу данных **TestData**. Иначе вы создадите таблицу в базе данных, к которой вы ранее подключились. Вероятно, это будет ваша база данных по умолчанию. Если не была выбрана другая база данных по умолчанию, ею будет база данных **master**. Не нужно создавать объекты в базе данных **master**.

#### Переключение соединения редактора запросов на базу данных TestData

1. В окне редактора запросов введите и выполните следующий код, чтобы изменить соединение на базу данных TestData.

USE TestData

GO

#### [Создание таблицы](javascript:void(0))

1. 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

# Вставка данных в таблицу и их обновление (учебник)

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.

# Вставка данных в таблицу

1. 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

1. 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

1. 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

1. 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

# Обновление таблицы продуктов

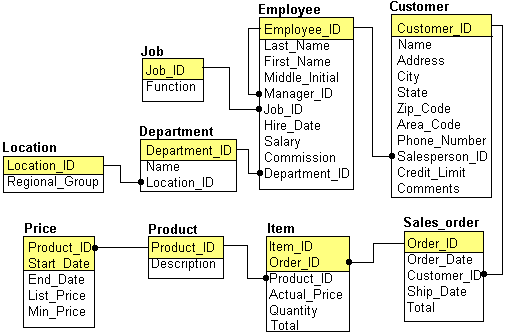
1. Введите и выполните следующую инструкцию UPDATE, чтобы изменить значение ProductName второго продукта со значения Screwdriver на значение Flat Head Screwdriver.

UPDATE dbo.Products

    SET ProductName = 'Flat Head Screwdriver'

    WHERE ProductID = 50

# Задания

1. Изучите концептуальную схему БД "King Corporation"
2. Соединитесь с сервером.

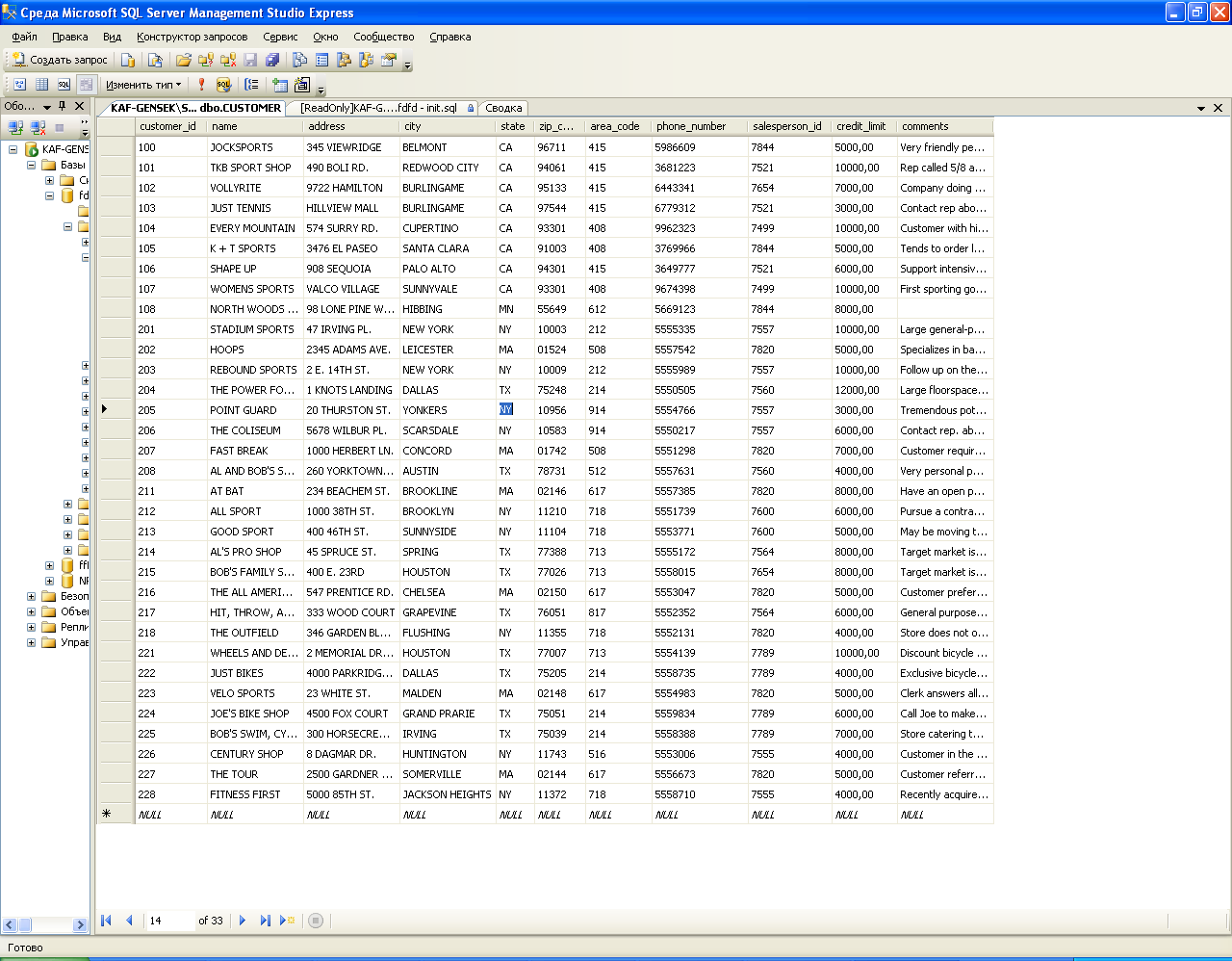
# Концептуальная схема базы данных "King Corporation"

1. Создайте БД, назовите его КС\_фамилия студента.
2. Создайте все 9 таблиц, пример создание таблицы и структуры таблиц описаны ниже:

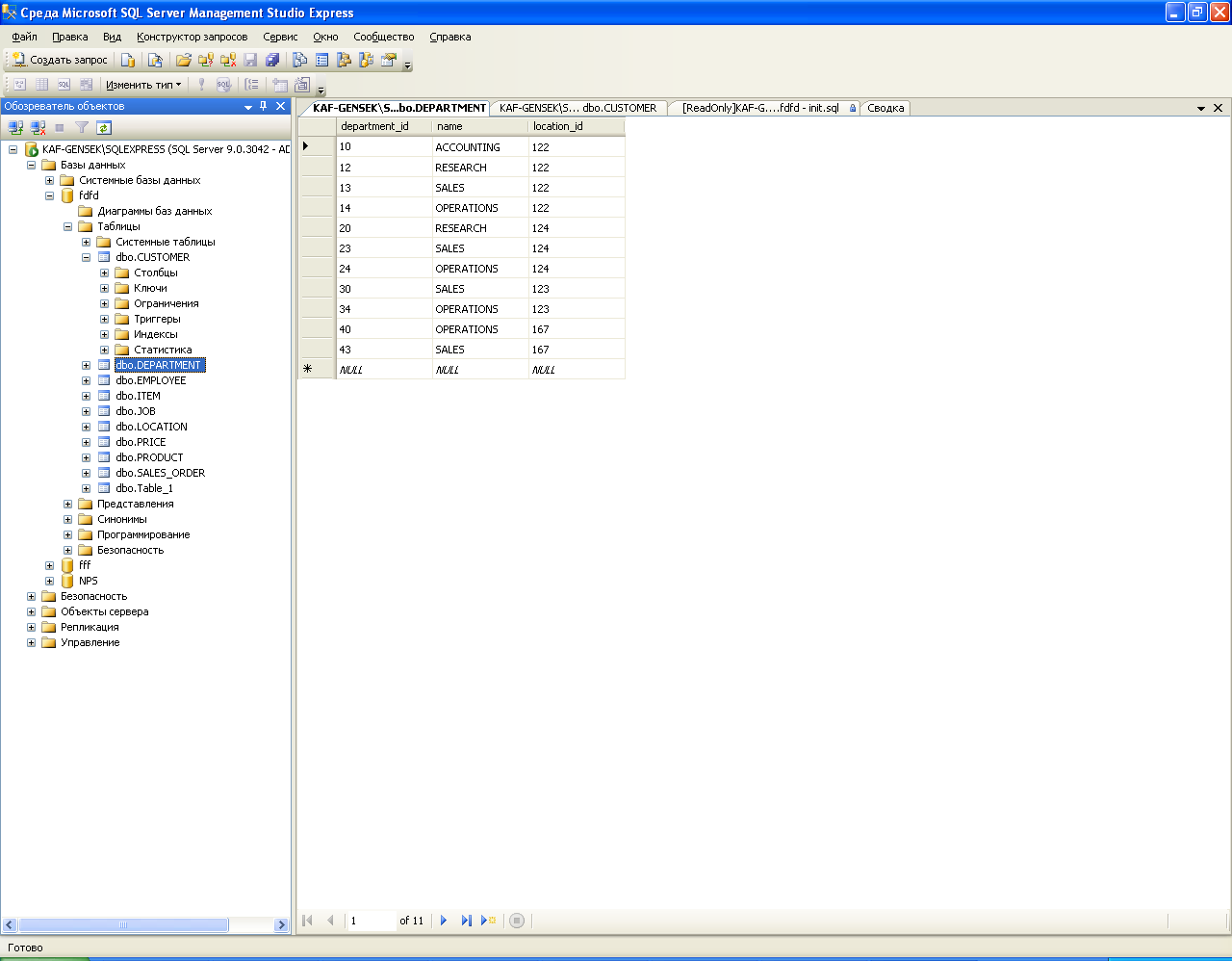
|  |  |  |  |
| --- | --- | --- | --- |
| N пп | Имя столбца | Тип данных DB2(MSSQL) | Комментарий |
| **Таблица EMPLOYEE - сотрудники фирмы** | | | |
| 1 | employee\_id | SMALLINT | Код сотрудника |
| 2 | last\_name | VARCHAR(15) | Фамилия |
| 3 | first\_name | VARCHAR(15) | Имя |
| 4 | middle\_initial | VARCHAR(1) | Средний инициал |
| 5 | manager\_id | SMALLINT | Код начальника |
| 6 | job\_id | SMALLINT | Код должности |
| 7 | hire\_date | DATE(DATETIME) | Дата поступления в фирму |
| 8 | salary | NUMERIC(7,2) | Зарплата |
| 9 | commission | NUMERIC(7,2) | Комиссионные |
| 10 | department\_id | SMALLINT | Код отдела |
| CREATE TABLE EMPLOYEE (  employee\_id SMALLINT NOT NULL PRIMARY KEY,  last\_name VARCHAR(15),  first\_name VARCHAR(15),  middle\_initial VARCHAR(1),  manager\_id SMALLINT,  job\_id SMALLINT,  hire\_date DATETIME,  salary NUMERIC(7,2),  commission NUMERIC(7,2),  department\_id SMALLINT ); | | | |
| **Таблица DEPARTMENT - отделы фирмы** | | | |
| 1 | department\_id | SMALLINT | Код отдела |
| 2 | name | VARCHAR(14) | Название отдела |
| 3 | location\_id | SMALLINT | Код места размещения |
| **Таблица LOCATION - места размещения отделов** | | | |
| 1 | location\_id | SMALLINT | Код места размещения |
| 2 | regional\_group | VARCHAR(20) | Город |
| **Таблица JOB - должности в фирме** | | | |
| 1 | job\_id | SMALLINT | Код должности |
| 2 | function | VARCHAR(30) | Название должности |
| **Таблица CUSTOMER - фирмы-покупатели** | | | |
| 1 | customer\_id | INTEGER | Код покупателя |
| 2 | name | VARCHAR(45) | Название покупателя |
| 3 | address | VARCHAR(40) | Адрес |
| 4 | city | VARCHAR(30) | Город |
| 5 | state | VARCHAR(2) | Штат |
| 6 | zip\_code | VARCHAR(9) | Почтовый код |
| 7 | area\_code | SMALLINT | Код региона |
| 8 | phone\_number | INTEGER | Телефон |
| 9 | salesperson\_id | SMALLINT | Код сотрудника-продавца, обслуживаю-щего данного покупателя |
| 10 | credit\_limit | NUMERIC(9,2) | Кредит для покупателя |
| 11 | comments | VARCHAR(500) | Примечания |
| **Таблица SALES\_ORDER - договоры о продаже** | | | |
| 1 | order\_id | SMALLINT | Код договора |
| 2 | order\_date | DATE(DATETIME) | Дата договора |
| 3 | customer\_id | INTEGER | Код покупателя |
| 4 | ship\_date | DATE(DATETIME) | Дата поставки |
| 5 | total | NUMERIC(8,2) | Общая сумма договора |
| **Таблица ITEM - акты продаж** | | | |
| 1 | order\_id | SMALLINT | Код договора, в состав которого входит акт |
| 2 | item\_id | SMALLINT | Код акта |
| 3 | product\_id | INTEGER | Код продукта |
| 4 | actual\_price | NUMERIC(8,2) | Цена продажи |
| 5 | quantity | INTEGER | Количество |
| 6 | total | NUMERIC(8,2) | Общая сумма |
| **Таблица PRODUCT - товары** | | | |
| 1 | product\_id | INTEGER | Код продукта |
| 2 | description | VARCHAR(30) | Название продукта |
| **Таблица PRICE - цены** | | | |
| 1 | product\_id | INTEGER | Код продукта |
| 2 | list\_price | NUMERIC(8,2) | Объявленная цена |
| 3 | min\_price | NUMERIC(8,2) | Минимально возможная цена |
| 4 | start\_date | DATE(DATETIME) | Дата установления цены |
| 5 | end\_date | DATE(DATETIME) | Дата отмены цены |

1. Вставьте данные в таблицы

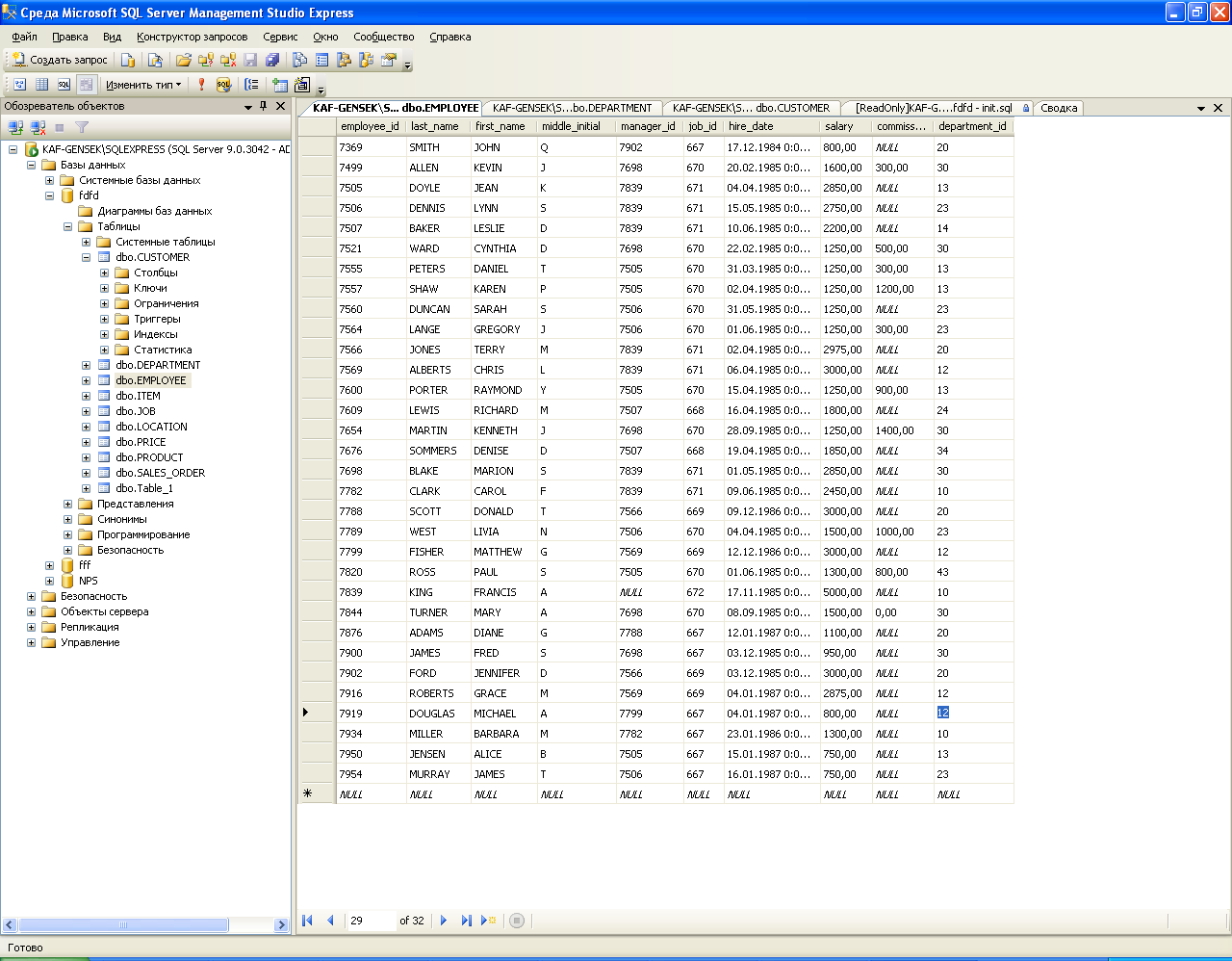
**Таблица CUSTOMER - фирмы-покупатели**



**Таблица DEPARTMENT - отделы фирмы**



**Таблица EMPLOYEE - сотрудники фирмы**



**Таблица ITEM - акты продаж**

