**Практическая работа № 5. Реализация таблиц и ограничений в базе данных.**

**Программное обеспечение: Windows 8 и выше, Microsoft Office Word 2016 и выше, Microsoft SQL Server 2017 и выше, Microsoft SQL Server Management Studio 2017 и выше.**

**Цель работы:**

**На основании даталогической модели данных и словаря данных, разработать структуру таблиц базы данных, на сервере СУБД.**

**Время на выполнение практической работы:** 2 часа.

**Получение навыков:**

* **В использовании запросов группы команд DDL;**
* **В разработке структуры таблиц базы данных;**
* **Определения типов данных в таблицах базы данных;**
* **В создании ограничений: уникальных полей и полей с проверками валидации;**
* **В применении оператора Default;**
* **В реализации межтабличных связей;**
* **В предварительной настройке файла базы данных;**
* **В работе с инструментальной средой разработки БД.**

**Теоретические сведения к выполнению практической работы:**

Система базы данных.

**СБД** - совокупность взаимосвязанных элементов действующих, как единое целое (Приложение БД - СУБД - БД).

СУБД.

**Система управления базами данных** - комплекс: языковых и программных средств, направленный на манипулирование базой данных, многими пользователями.

Функции системы управления базами данных.

1. Управление данными во временной памяти ПК;
2. Управление данными в постоянной памяти ПК;
3. Обработка запросов на манипулирование данных (SQL);
4. Обеспечивает механизмы разработки БД;
5. Обеспечивает защиту данных и целостность данных;
6. Обеспечивает передачу информации надлежащим пользователям.

База данных.

**БД** - совокупность взаимосвязанных данных, предусматривающие общие принципы описания, хранения и манипулирование данными, вне зависимости от прикладных программ.

Язык SQL.

**SQL (Structured Query Language)** – язык структурированных, запросов, который направлен на разработку и ведение структуры баз данных и на манипулирование и актуализацию данных в базе данных.

**Ведение базы данных** – это процесс сопровождения структуры базы данных;

**Сопровождение** – процесс обновления и актуализации элементов программного обеспечения в соответствии с требованиями заказчика и спецификой предметной области;

**Манипулирование данными** – обобщающий термин для добавления, изменения, удаления, сортировки и поиска данных;

**Актуализация данных** – процесс приведения, конечным пользователем, информации к достоверному виду в базе данных, за счёт средств манипулирования данными.

DDL.

**DDL (Data Definition Language)** – язык определения данных, задачей которого является разработка и ведение структуры объектов базы данных.

Состоит из команд:

**1. Create** – команда создания объектов базы данных;

**2. Alter** – команда изменения структуры объектов базы данных;

**3. Drop** – команда удаления объектов базы данных.

Типы данных Microsoft SQL Server.

|  |  |  |
| --- | --- | --- |
| **Название типа данных** | **Размер/Параметры** | **Способ описания в SQL** |
| **Char(n)** – строковая переменная фиксированного размера | N – от 1 до 8000 | ‘Some Text’ |
| **Varchar(n)** – строковая переменная переменного размера | N – от 1 до 8000  N = Max <= 2Gb | ‘Some Text’ |
| **Date** – переменная даты | 0001-01-01 – 9999-12-31 | ’2022-01-31’ |
| **Datetime** – переменная даты и времени | Диапазон даты: 01.01.1753 – 31.12.9999  Диапазон времени: 00:00:00 – 23:59:590,997 | ‘2022-01-31 12:30:40.123’ |
| **Time** – переменная времени | 00:00:00.0000000 – 23.59.59.9999999 | ’12:30:40.1234’ |
| **Int** – целочисленный, положительный и отрицательный тип данных | -2 147 483 648 - 2 147 483 647 | 1234 |
| **Decimal(n,m)** – вещественный тип данных | N – разрядность числа от 1 до 38  M – точность числа 0 <= m <=n | 1234.56 |
| **Bit** – целочисленная переменная | 1, 0, null | 1 | null |

Структура кода языка SQL.

* Язык SQL – язык скриптов, т.е. язык сценариев;
* Язык SQL – выполняется с верху в низ;
* Скрипт языка SQL возможно выполнять выборочно;
* Обработчик кода выполняет все возможные сценарии и не останавливает обработку, если какой-либо из сценариев написан не верно;
* Если скрипт написан верно – транзакция автоматически принимается, в противном случае производится откат до предыдущей стабильной версии БД.

Объект Database.

Команды и ключевые слова:

**Set** – команда установки какого либо значения оператору или строке;

**Ansi\_Nulls** – задаёт совместно со стандартом ISO поведение операторов = и <> с использованием значения Null;

**Ansi\_Padding** – управление значениями, которые короче, чем определённый размер столбца, не заполняя его пробелами;

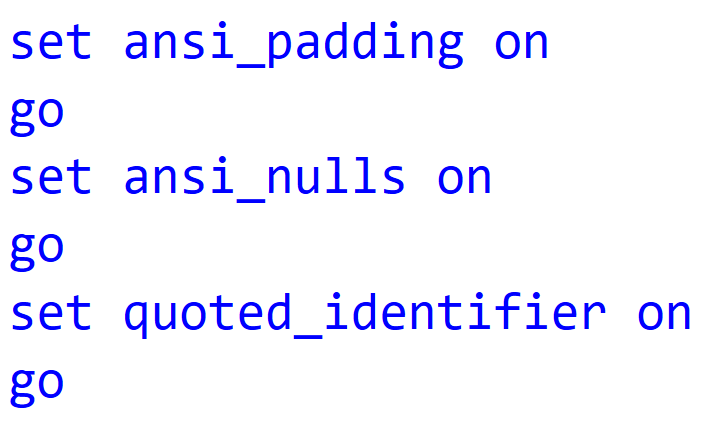
**Quoted\_Identifier** – автоматическое экранирование, специальных символов и зарезервированных слов, «”» в языке Transact-SQL;

**Database [<Name>]** – команда создание экземпляра файла базы данных;

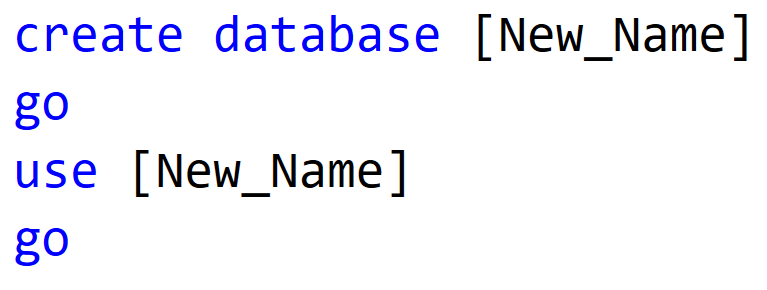
**Use [<Database\_Name>]** – команда определения работы скрипта, с конкретным экземпляром базы данных;

**Go** – ключевое слово конца логического блока транзакции.

Перед созданием любой базы данных в Microsoft SQL Server, необходимо задать следующие команды:



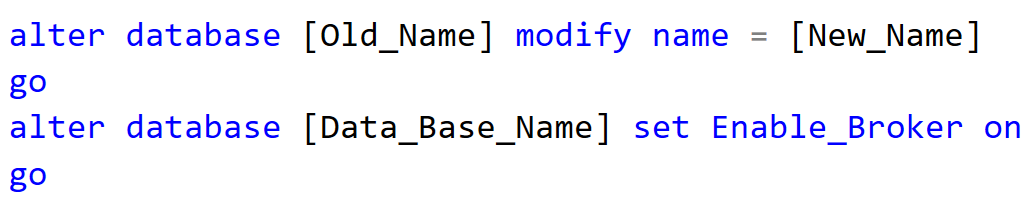
Создание и последующие использование базы данных:



Ключевые слова и команды для изменения базы данных:

**Modify name** – команда определения нового имени для существующей базы данных;

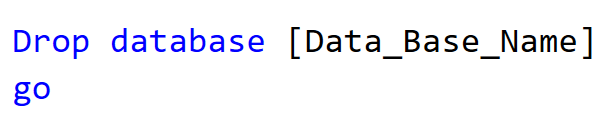
**Enable\_Broker** – установка для файла базы данных, возможности доступности экземпляра базы данных в многопользовательском режиме.



Удаление объекта базы данных.

Для успешного удаления объекта базы данных, необходимо выполнение следующих условий:

1. К базе данных закрыты все подключения;
2. База данных не используется в данный момент конкретным подключением инструментальной среды;
3. Снято действие команды use.



Объект Table.

Команды и ключевые слова:

**DBO** – **D**ata**b**ase **O**wner, базовая схема создания объектов в базе данных;

**Identity(n,m)** – команда, определяет поле как автоинкрементное, где n – стартовый шаг, а m – шаг инкрементирования;

**Автоикриментирование** – процесс автоматического увеличения значения столбца на шаг инкреминтирования.

**Constraint** – объект базы данных, который отвечает за ограничения на поля, различают 4 вида:

* **Primary key** – ограничение на столбец, который устанавливает поле как первичный ключ, для обеспечения межтабличной связи;
* **Foreign key** – ограничение на столбце, который устанавливает у поля тип ссылки на значение поля с ограничением Primary key;
* **Unique** – ограничение на столбец, который устанавливает на поле проверку ввода уникальных значений, при добавлении и изменении данных. Если ограничение возвращает True – манипулирование над столбцом будет принято, в противном случае – откат транзакции;
* **Check** – ограничение на столбец, который устанавливает проверку ввода данных, на выполнение вложенных условий. Если запись удовлетворяет условиям – манипулирование над столбцом будет принято, в противном случае – откат транзакции;

**Default(n)** – команда выполняется тогда и только тогда, когда в поле при добавлении данных попадает значение Null, формальный параметр n, не типизированное значение, которое наследует свой тип у столбца;

**Clustered** – создаёт уникальный индекс для хранения значений первичных ключей, с целью быстрой обработки данных: поиска, изменении, удаления и фильтрации;

**ASC** – порядок сортировки значений столбца или индекса, в прямом порядке;

**[PRIMARY]** – указывает место хранения индекса в структуре сервера БД;

**References** – команда определяет подход к организации ссылки между значением первичного ключа родительской таблицы и внешним ключом дочерней таблицы.

Общая конструкция – Check ([Filed\_Name] логический оператор ‘шаблон’)

Операторы для работы с Constraint Check:

**=, <, >, <=, >=, <>** - логические операторы сравнения значений поля;

**In (n)** – функция проверки соответствия вводимого значения в столбец, с одним или несколькими значениями перечисленными в параметре n;

**Between n and m** – команда определения диапазона для, числовых типов данных и для формата даты и время, где n – стартовое значение, m – конечное значение.

**Like (n)** – функция проверки шаблона n, с значением столбца или строки;

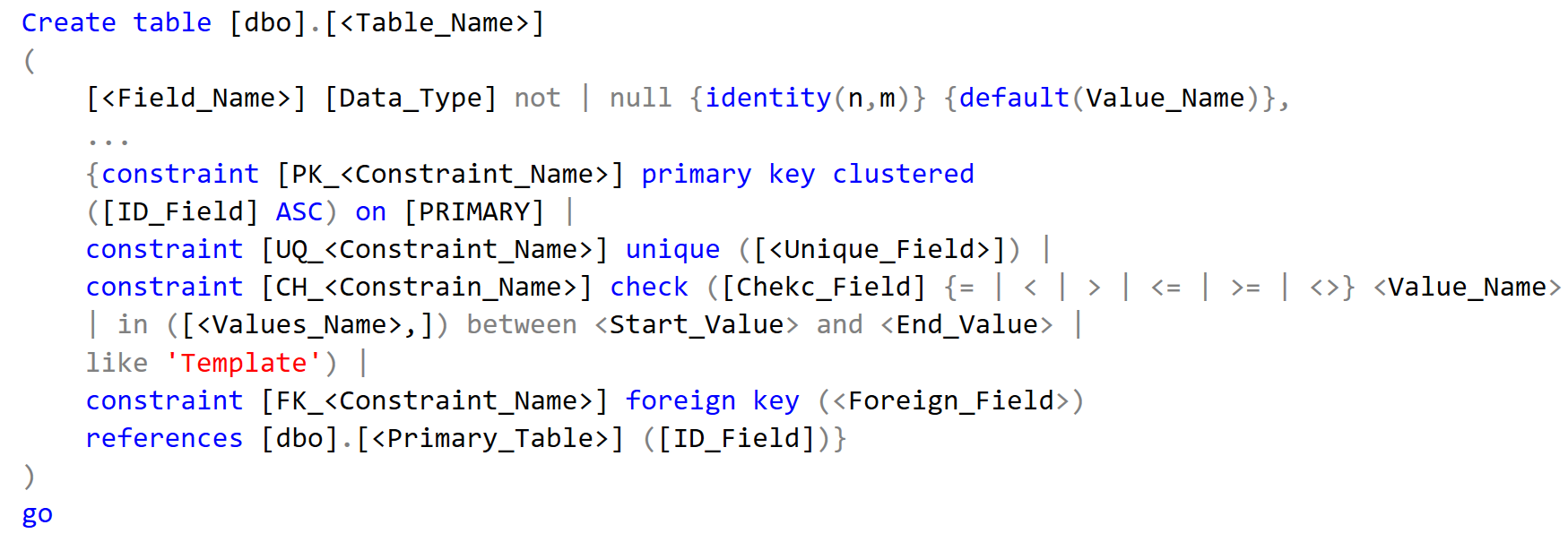
Команды для шаблона:

**[ ]** – определяет массив значений для одной позиции в искомой строке, пример: [0-9] или [a-z] или [!@#$%^&\*];

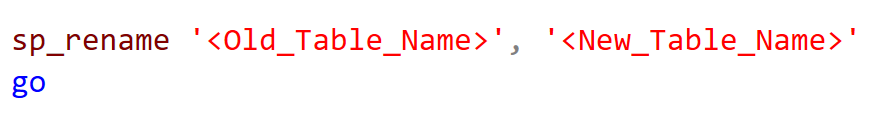
**%** – определяет любое количество позиций любого символа в искомой строке;

**\_** – определяет любой символ в одной позиции подстроки.

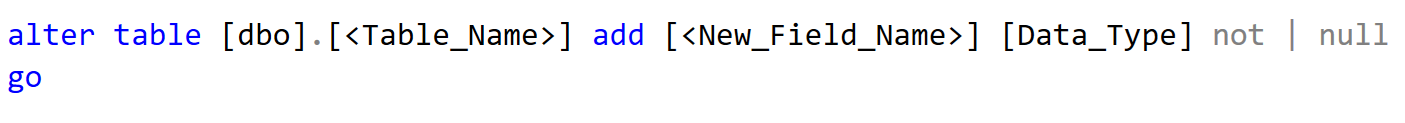
Конструкция Table DDL.



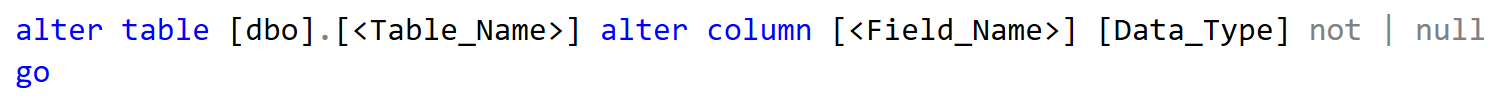
Переименование названия таблицы, с помощью метода sp\_rename, где указывается старое название таблицы и новое название:



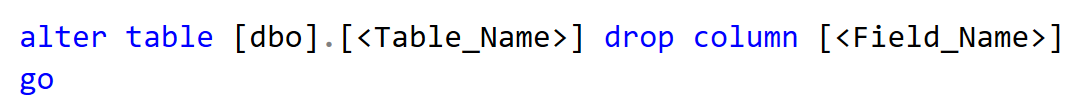
Добавление нового столбца, изменение текущей таблицы, добавив новый столбец с типом данных и настройкой обязательного заполнения:



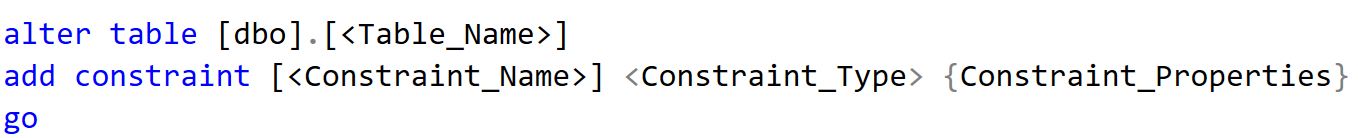
Изменение структуры таблицы, изменив тип данных у столбца с присвоением свойств обязательного заполнения данных:



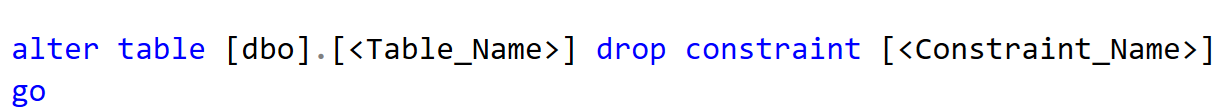
Изменение структуры таблицы, удалив существующий столбец:



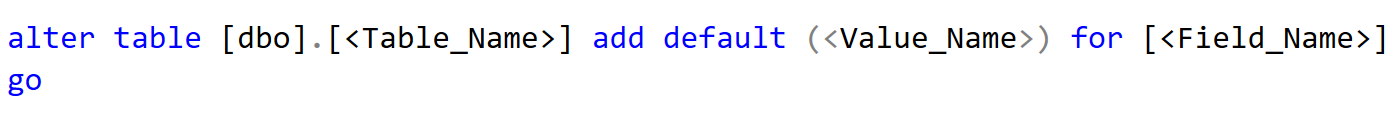
Изменение таблицы, с добавлением нового ограничения со свойствами одного из 4 ограничений к столбцу:



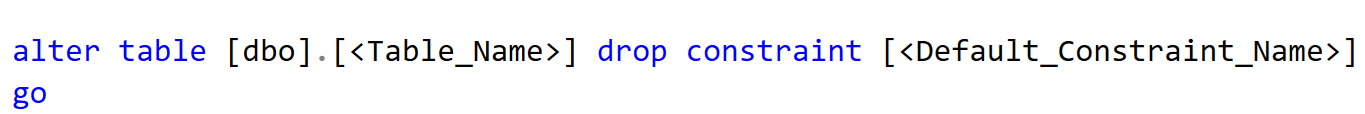
Изменение таблицы, удалив ограничение по названию:



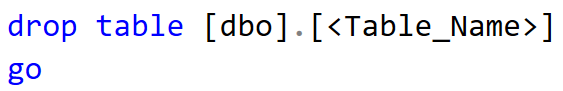
Изменение структуры таблицы, с присвоением существующему полю, значения по умолчанию:



Изменение структуры таблицы, с удалением ограничения по умолчанию:

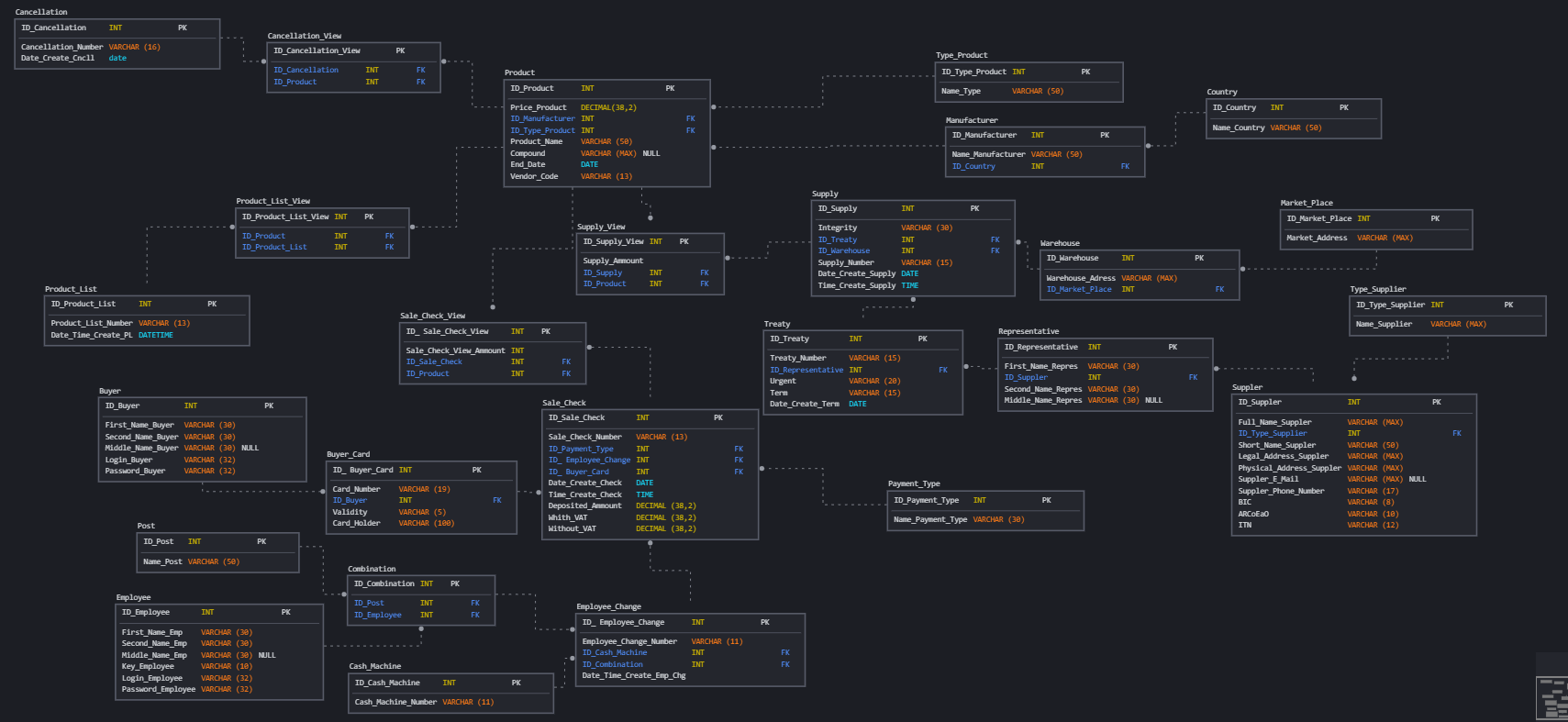


Удаление таблицы из базы данных:



**Этапы выполнения:**

1. Проект даталогической модели данных, на основании 4 практической работы;



1. Реализация скрипта настройки и создания файла базы данных;

|  |  |
| --- | --- |
| Название объекта | Скрипт объекта |
| Sale\_Product\_Database | set ansi\_nulls on  go  set ansi\_padding on  go  set quoted\_identifier on  go  create database [Sale\_Product\_Database]  go  use [Sale\_Product\_Database]  go |

1. На основании словаря данных, произвести реализацию родительских таблиц базы данных (Поле с типом данных, Varchar (max), не может быть с ограничением unique – пометить жёлтым цветом);

| Ключ | Наименование | Тип данных | Примечание | Скрипт объекта |
| --- | --- | --- | --- | --- |
| Type\_Product | | | | |
| PK | ID\_Type\_Product | INT | Первичный ключ таблицы «Вид Товара» | create table [dbo].[Type\_Product]  (  [ID\_Type\_Product] [int] not null identity(1,1),  [Name\_Type] [varchar] (50) not null  constraint [PK\_Type\_Product] primary key clustered  ([ID\_Type\_Product] ASC) on [PRIMARY],  constraint [UQ\_Name\_Type] unique ([Name\_Type])  )  go |
|  | Name\_Type | VARCHAR (50) | Уникальное поле |
| Country | | | | |
| PK | ID\_Country | INT | Первичный ключ таблицы «Страна» | create table [dbo].[Country]  (  [ID\_Country] [int] not null identity(1,1),  [Name\_Country] [varchar] (50) not null  constraint [PK\_Country] primary key clustered  ([ID\_Country] ASC) on [PRIMARY],  constraint [UQ\_Name\_Country] unique ([Name\_Country])  )  go |
|  | Name\_Country | VARCHAR (50) | Уникальное поле |
| Market\_Place | | | | |
| PK | ID\_Market\_Place | INT | Первичный ключ таблицы «Товарная точка» | create table [dbo].[Market\_Place]  (  [ID\_Market\_Place] [int] not null identity(1,1),  [Market\_Address] [varchar] (max) not null  constraint [PK\_Market\_Place] primary key clustered  ([ID\_Market\_Place] ASC) on [PRIMARY]  )  go |
|  | Market\_Address | VARCHAR (MAX) | Уникальное поле |
| Type\_Supplier | | | | |
| PK | ID\_Type\_Supplier | INT | Первичный ключ таблицы «Тип поставщика» | create table [dbo].[Type\_Supplier]  (  [ID\_Type\_Supplier] [int] not null identity(1,1),  [Name\_Supplier] [varchar] (30) not null  constraint [PK\_Type\_Supplier] primary key clustered  ([ID\_Type\_Supplier] ASC) on [PRIMARY],  constraint [UQ\_Name\_Supplier] unique ([Name\_Supplier])  )  go |
|  | Name\_Supplier | VARCHAR (30) | Уникальное поле |
| Payment\_Type | | | | |
| PK | ID\_Payment\_Type | INT | Первичный ключ таблицы «Вид расчёта» | create table [dbo].[Payment\_Type]  (  [ID\_Payment\_Type] [int] not null identity(1,1),  [Name\_Payment\_Type] [varchar] (30) not null  constraint [PK\_Payment\_Type] primary key clustered  ([ID\_Payment\_Type] ASC) on [PRIMARY],  constraint [UQ\_Name\_Payment\_Type] unique ([Name\_Payment\_Type])  )  go |
|  | Name\_Payment\_Type | VARCHAR (30) | Уникальное поле |
| Post | | | | |
| PK | ID\_Post | INT | Первичный ключ таблицы «Должность» | create table [dbo].[Post]  (  [ID\_Post] [int] not null identity(1,1),  [Name\_Post] varchar (50) not null  constraint [PK\_Post] primary key clustered  ([ID\_Post] ASC) on [PRIMARY],  constraint [UQ\_Name\_Post] unique ([Name\_Post])  )  go |
|  | Name\_Post | VARCHAR (50) | Уникальное поле |
| Employee | | | | |
| PK | ID\_Employee | INT | Первичный ключ таблицы «Сотрудник» | create table [dbo].[Employee]  (  [ID\_Employee] [int] not null identity(1,1),  [First\_Name\_Emp] [varchar] (30) not null,  [Second\_Name\_Emp] [varchar] (30) not null,  [Middle\_Name\_Emp] [varchar] (30) null default ('-'),  [Key\_Employee] [varchar] (10) not null,  [Login\_Employee] [varchar] (32) not null,  [Password\_Employee] [varchar] (32) not null  constraint [PK\_Employee] primary key clustered  ([ID\_Employee] ASC) on [PRIMARY],  constraint [UQ\_Login\_Employee] unique ([Login\_Employee]),  constraint [CH\_Key\_Employee] check (len([Key\_Employee]) = 10),  constraint [CH\_Key\_Employee\_Letter\_Upper] check ([Key\_Employee] like ('%[A-Z]%')),  constraint [CH\_Key\_Employee\_Letter\_Lower] check ([Key\_Employee] like ('%[a-z]%')),  constraint [CH\_Key\_Employee\_Symbols] check ([Key\_Employee] like ('%[!@#$%^&\*()]%')),  constraint [CH\_Login\_Employee] check (len([Login\_Employee])>=8),  constraint [CH\_Password\_Employee\_Upper] check ([Password\_Employee] like ('%[A-Z]%')),  constraint [CH\_Password\_Employee\_Letter\_Lower] check ([Password\_Employee] like ('%[a-z]%')),  constraint [CH\_Password\_Employee\_Symbols] check ([Password\_Employee] like ('%[!@#$%^&\*()]%'))  )  go |
|  | First\_Name\_Emp | VARCHAR (30) |  |
|  | Second\_Name\_Emp | VARCHAR (30) |  |
|  | Middle\_Name\_Emp | VARCHAR (30) |  |
|  | Key\_Employee | VARCHAR (10) | Спец символы, минимум одна заглавная одна прописная латинская буквы, ровно 10 символов |
|  | Login\_Employee | VARCHAR (32) | Не менее 8 символов, уникальное поле |
|  | Password\_Employee | VARCHAR (32) | Спец символы, минимум одна заглавная одна прописная латинская буквы |
| Cash\_Machine | | | | |
| PK | ID\_Cash\_Machine | INT | Первичный ключ таблицы «Кассовый аппарат» | create table [dbo].[Cash\_Machine]  (  [ID\_Cash\_Machine] [int] not null identity(1,1),  [Cash\_Machine\_Number] [varchar] (11) not null  constraint [PK\_Cash\_Machine] primary key clustered  ([ID\_Cash\_Machine] ASC) on [PRIMARY],  constraint [UQ\_Cash\_Machine\_Number] unique ([Cash\_Machine\_Number]),  constraint [CH\_Cash\_Machine\_Number] check ([Cash\_Machine\_Number] like('КА-[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'))  )  go |
|  | Cash\_Machine\_Number | VARCHAR (11) | Уникальное поле, КА-00000000 |
| Buyer | | | | |
| PK | ID\_Buyer | INT | Первичный ключ таблицы «Покупатель» | create table [dbo].[Buyer]  (  [ID\_Buyer] [int] not null identity(1,1),  [First\_Name\_Buyer] [varchar] (30) not null,  [Second\_Name\_Buyer] [varchar] (30) not null,  [Middle\_Name\_Buyer] [varchar] (30) null default ('-'),  [Login\_Buyer] [varchar] (32) not null,  [Password\_Buyer] [varchar] (32) not null  constraint [PK\_Buyer] primary key clustered  ([ID\_Buyer] ASC) on [PRIMARY],  constraint [UQ\_Login\_Buyer] unique ([Login\_Buyer]),  constraint [CH\_Login\_Buyer] check (len([Login\_Buyer])>=8),  constraint [CH\_Password\_Buyer\_Upper] check ([Password\_Buyer] like ('%[A-Z]%')),  constraint [CH\_Password\_Buyer\_Letter\_Lower] check ([Password\_Buyer] like ('%[a-z]%')),  constraint [CH\_Password\_Buyer\_Symbols] check ([Password\_Buyer] like ('%[!@#$%^&\*()]%'))  )  go |
|  | First\_Name\_Buyer | VARCHAR (30) |  |
|  | Second\_Name\_Buyer | VARCHAR (30) |  |
|  | Middle\_Name\_Buyer | VARCHAR (30) |  |
|  | Login\_Buyer | VARCHAR (32) | Не менее 8 символов, уникальное поле |
|  | Password\_Buyer | VARCHAR (32) | Спец символы, минимум одна заглавная одна прописная латинская буквы |
| Product\_List | | | | |
| PK | ID\_Product\_List | INT | Первичный ключ таблицы «Товарная смета» | create table [dbo].[Product\_List]  (  [ID\_Product\_List] [int] not null identity(1,1),  [Product\_List\_Number] [varchar] (13) not null,  [Date\_Time\_Create\_PL] [datetime] null default(getdate())  constraint [PK\_Product\_List] primary key clustered  ([ID\_Product\_List] ASC) on [PRIMARY],  constraint [CH\_Product\_List\_Number] check ([Product\_List\_Number] like '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]-[0-9][0-9]')  )  go |
|  | Product\_List\_Number | VARCHAR (13) | Уникальное поле, 0000000000-YY |
|  | Date\_Time\_Create\_PL | DATETIME | Текущая дата и время |
| Cancellation | | | | |
| PK | ID\_Cancellation | INT | Первичный ключ таблицы «Отмена приёма» | create table [dbo].[Cancellation]  (  [ID\_Cancellation] [int] not null identity(1,1),  [Cancellation\_Number] [varchar] (16) not null,  [Date\_Create\_Cncll] [date] null default (getdate())  constraint [PK\_Cancellation] primary key clustered  ([ID\_Cancellation] ASC) on [PRIMARY],  constraint [CH\_Cancellation\_Number] check ([Cancellation\_Number] like ('[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]-[0-9][0-9]-ОП'))  )  go |
|  | Cancellation\_Number | VARCHAR (16) | Уникальное поле, 0000000000-YY-ОП |
|  | Date\_Create\_Cncll | DATE | Текущая дата |

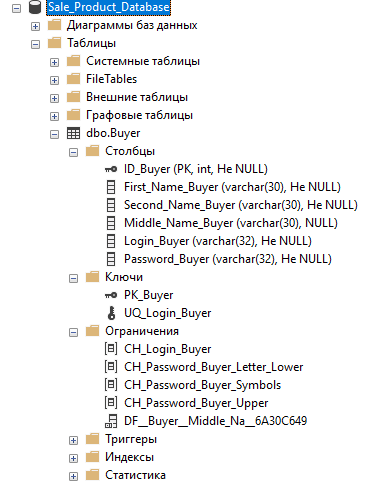
1. На основании словаря данных, произвести реализацию дочерних таблиц с одной связью 1:М;

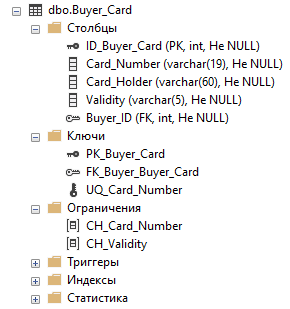
| Ключ | Наименование | Тип данных | Примечание | Скрипт объекта |
| --- | --- | --- | --- | --- |
| Manufacturer | | | | |
| PK | ID\_Manufacturer | INT | Первичный ключ таблицы «Производитель» | create table [dbo].[Manufacturer]  (  [ID\_Manufacturer] [int] not null identity(1,1),  [Name\_Manufacturer] [varchar] (50) not null,  [Country\_ID] [int] not null  constraint [PK\_Manufacturer] primary key clustered  ([ID\_Manufacturer] ASC) on [PRIMARY],  constraint [UQ\_Name\_Manufacturer] unique ([Name\_Manufacturer]),  constraint [FK\_Country\_Manufacturer] foreign key ([Country\_ID])  references [dbo].[Country] ([ID\_Country])  )  go |
|  | Name\_Manufacturer | VARCHAR (50) | Уникальное поле |
| FK | Country\_ID | INT |  |
| Warehouse | | | | |
| PK | ID\_Warehouse | INT | Первичный ключ таблицы «Склад» | create table [dbo].[Warehouse]  (  [ID\_Warehouse] [int] not null identity(1,1),  [Warehouse\_Adress] [varchar] (max) not null,  [Market\_Place\_ID] [int] not null  constraint [PK\_Warehouse] primary key clustered  ([ID\_Warehouse] ASC) on [PRIMARY],  constraint [FK\_Market\_Place\_Warehouse] foreign key ([Market\_Place\_ID])  references [dbo].[Market\_Place] ([ID\_Market\_Place])  )  go |
|  | Warehouse\_Adress | VARCHAR (MAX) | Уникальное поле |
| FK | Market\_Place\_ID | INT |  |
| Suppler | | | | |
| PK | ID\_Suppler | INT | Первичный ключ таблицы «Поставщик» | create table [dbo].[Suppler]  (  [ID\_Suppler] [int] not null identity(1,1),  [Type\_Suppler\_ID] [int] not null,  [Full\_Name\_Suppler] [varchar] (max) not null,  [Short\_Name\_Suppler] [varchar] (50) not null,  [Legal\_Address\_Suppler] [varchar] (max) not null,  [Physical\_Address\_Suppler] [varchar] (max) not null,  [Suppler\_E\_Mail] [varchar] (max) not null,  [Suppler\_Phone\_Number] [varchar] (17) not null,  [BIC] [varchar] (8) not null,  [ITN] [varchar] (12) not null,  [ARCoEaO] [varchar] (10) not null  constraint [PK\_Suppler] primary key clustered  ([ID\_Suppler] ASC) on [PRIMARY],  constraint [UQ\_Suppler\_Phone\_Number] unique ([Suppler\_Phone\_Number]),  constraint [UQ\_ITN] unique ([ITN]),  constraint [UQ\_ARCoEaO] unique ([ARCoEaO]),  constraint [CH\_Suppler\_E\_Mail] check ([Suppler\_E\_Mail] like ('%@%.%')),  constraint [CH\_Suppler\_Phone\_Number] check ([Suppler\_Phone\_Number] like '+7([0-9][0-9][0-9])[0-9][0-9][0-9]-[0-9][0-9]-[0-9][0-9]'),  constraint [CH\_BIC] check ([BIC] like ('[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]')),  constraint [CH\_ITN] check ([ITN] like ('[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]')),  constraint [CH\_ARCoEaO] check ([ARCoEaO] like ('[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]')),  constraint [FK\_Type\_Suppler\_Suppler] foreign key ([Type\_Suppler\_ID])  references [dbo].[Type\_Supplier] ([ID\_Type\_Supplier])  )  go |
| FK | Type\_Suppler\_ID | INT |  |
|  | Full\_Name\_Suppler | VARCHAR (MAX) |  |
|  | Short\_Name\_Suppler | VARCHAR (50) |  |
|  | Legal\_Address\_Suppler | VARCHAR (MAX) |  |
|  | Physical\_Address\_Suppler | VARCHAR (MAX) |  |
|  | Suppler\_E\_Mail | VARCHAR (MAX) | Наличие «@» и «.», уникальное поле |
|  | Suppler\_Phone\_Number | VARCHAR (17) | +7(000)000-00-00, уникальное поле |
|  | BIC | VARCHAR (8) | 00000000 |
|  | ITN | VARCHAR (12) | Уникальное поле, 000000000000 |
|  | ARCoEaO | VARCHAR (10) | Уникальное поле, 0000000000 |
| Representative | | | | |
| PK | ID\_Representative | INT | Первичный ключ таблицы «Представитель» | create table [dbo].[Representative]  (  [ID\_Representative] [int] not null identity(1,1),  [Suppler\_ID] [int] not null,  [First\_Name\_Repres] [varchar] (30) not null,  [Second\_Name\_Repres] [varchar] (30) not null,  [Middle\_Name\_Repres] [varchar] (30) null default('-')  constraint [PK\_Representative] primary key clustered  ([ID\_Representative] ASC) on [PRIMARY],  constraint [FK\_Suppler\_Representative] foreign key ([Suppler\_ID])  references [dbo].[Suppler] ([ID\_Suppler])  )  go |
| FK | Suppler\_ID | INT |  |
|  | First\_Name\_Repres | VARCHAR (30) |  |
|  | Second\_Name\_Repres | VARCHAR (30) |  |
|  | Middle\_Name\_Repres | VARCHAR (30) |  |
| Treaty | | | | |
| PK | ID\_Treaty | INT | Первичный ключ таблицы «Договор» | create table [dbo].[Treaty]  (  [ID\_Treaty] [int] not null identity(1,1),  [Treaty\_Number] [varchar] (15) not null,  [Urgent] [varchar] (3) not null,  [Term] [varchar] (15) not null,  [Date\_Create\_Term] [date] null default(getdate()),  [Representative\_ID] [int] not null  constraint [PK\_Treaty] primary key clustered  ([ID\_Treaty] ASC) on [PRIMARY],  constraint [UQ\_Treaty\_Number] unique ([Treaty\_Number]),  constraint [CH\_Treaty\_Number] check ([Treaty\_Number] like ('[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]-[0-9][0-9]-Д')),  constraint [CH\_Urgent] check ([Urgent] in ('Да','Нет')),  constraint [CH\_Term] check ([Term] like ('[0-9][0-9] лет') or [Term] like ('[0-9][0-9][0-9] месяцев')),  constraint [FK\_Representative\_Treaty] foreign key ([Representative\_ID])  references [dbo].[Representative] ([ID\_Representative])  )  go |
|  | Treaty\_Number | VARCHAR (15) | Уникальное поле, 0000000000-YY-Д |
|  | Urgent | VARCHAR (3) | «Да», «Нет» |
|  | Term | VARCHAR (15) | «0 лет», «0 меяцев» |
|  | Date\_Create\_Term | DATE | Текущая дата |
| FK | Representative\_ID | INT |  |
| Buyer\_Card | | | | |
| PK | ID\_ Buyer\_Card | INT | Первичный ключ таблицы «Карта клиента» | create table [dbo].[Buyer\_Card]  (  [ID\_Buyer\_Card] [int] not null identity(1,1),  [Card\_Number] [varchar] (19) not null,  [Card\_Holder] [varchar] (60) not null,  [Validity] [varchar] (5) not null,  [Buyer\_ID] [int] not null  constraint [PK\_Buyer\_Card] primary key clustered  ([ID\_Buyer\_Card] ASC) on [PRIMARY],  constraint [UQ\_Card\_Number] unique ([Card\_Number]),  constraint [CH\_Card\_Number] check ([Card\_Number] like '[0-9][0-9][0-9][0-9] [0-9][0-9][0-9][0-9] [0-9][0-9][0-9][0-9] [0-9][0-9][0-9][0-9]'),  constraint [CH\_Validity] check ([Validity] like ('[0-9][0-9]/[0-9][0-9]')),  constraint [FK\_Buyer\_Buyer\_Card] foreign key ([Buyer\_ID])  references [dbo].[Buyer] ([ID\_Buyer])  )  go |
|  | Card\_Number | VARCHAR (19) | Уникальное поле, 0000 0000 0000 0000 |
|  | Card\_Holder | VARCHAR (60) |  |
|  | Validity | VARCHAR (5) | MM/YY |
| FK | Buyer\_ID | INT |  |

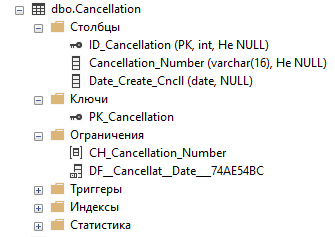
* 1. На основании словаря данных, произвести реализацию, дочерних таблиц со связями М:М;

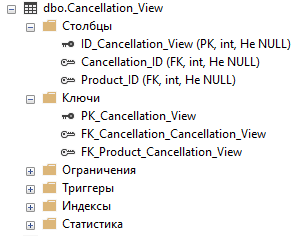
| Ключ | Наименование | Тип данных | Примечание | Скрипт объекта |
| --- | --- | --- | --- | --- |
| Combination | | | | |
| PK | ID\_Combination | INT | Первичный ключ таблицы «Совместительство» | create table [dbo].[Combination]  (  [ID\_Combination] [int] not null identity(1,1),  [Post\_ID] [int] not null,  [Employee\_ID] [int] not null  constraint [PK\_Combination] primary key clustered  ([ID\_Combination] ASC) on [PRIMARY],  constraint [FK\_Post\_Combination] foreign key ([Post\_ID])  references [dbo].[Post] ([ID\_Post]),  constraint [FK\_Employee\_Combination] foreign key ([Employee\_ID])  references [dbo].[Employee] ([ID\_Employee])  )  go |
| FK | Post\_ID | INT |  |
| FK | Employee\_ID | INT |  |
| Employee\_Change | | | | |
| PK | ID\_ Employee\_Change | INT | Первичный ключ таблицы «Смена» | create table [dbo].[Employee\_Change]  (  [ID\_Employee\_Change] [int] not null identity(1,1),  [Employee\_Change\_Number] [varchar] (11) not null,  [Combination\_ID] [int] not null,  [Cash\_Machine\_ID] [int] not null,  [Date\_Time\_Create\_Emp\_Chg] [datetime] null default(getdate())  constraint [PK\_Employee\_Change] primary key clustered  ([ID\_Employee\_Change] ASC) on [PRIMARY],  constraint [FK\_Combination\_Employee\_Change] foreign key ([Combination\_ID])  references [dbo].[Combination] ([ID\_Combination]),  constraint [FK\_Cash\_Machine\_Employee\_Change] foreign key ([Cash\_Machine\_ID])  references [dbo].[Cash\_Machine] ([ID\_Cash\_Machine])  )  go |
|  | Employee\_Change\_Number | VARCHAR (11) | Уникальное поле, 00000000000 |
| FK | Combination\_ID | INT |  |
| FK | Cash\_Machine\_ID | INT |  |
|  | Date\_Time\_Create\_Emp\_Chg | DATETIME | Текущая дата и время |
| Product | | | | |
| PK | ID\_Product | INT | Первичный ключ таблицы «Товар» | create table [dbo].[Product]  (  [ID\_Product] [int] not null identity(1,1),  [Price\_Product] [decimal] (38,2) null default (0.0),  [Product\_Name] [varchar] (50) not null,  [Compound] [varchar] (max) not null,  [End\_Date] [date] not null,  [Vendor\_Code] [varchar] (13) not null,  [Product\_Type\_ID] [int] not null,  [Manufacturer\_ID] [int] not null  constraint [PK\_Product] primary key clustered  ([ID\_Product] ASC) on [PRIMARY],  constraint [UQ\_Vendor\_Code] unique ([Vendor\_Code]),  constraint [CH\_Vendor\_Code] check ([Vendor\_Code] like '[A-Z][0-9][0-9][0-9][0-9][0-9][0-9]-[0-9][0-9][0-9][0-9][0-9]'),  constraint [CH\_Price\_Product] check ([Price\_Product] >= 0),  constraint [CH\_End\_Date] check ([End\_Date] >= Getdate()),  constraint [FK\_Product\_Type\_Product] foreign key ([Product\_Type\_ID])  references [dbo].[Type\_Product] ([ID\_Type\_Product]),  constraint [FK\_Manufacturer\_Product] foreign key ([Manufacturer\_ID])  references [dbo].[Manufacturer] ([ID\_Manufacturer])  )  go |
|  | Price\_Product | DECIMAL(38,2) | Не отрицательное |
|  | Product\_Name | VARCHAR (50) |  |
|  | Compound | VARCHAR (MAX) |  |
|  | End\_Date | DATE | Значение больше или равно текущей даты |
|  | Vendor\_Code | VARCHAR (13) | Уникальное поле, [A-Z]000000-00000 |
| FK | Type\_Product \_ID | INT |  |
| FK | Manufacturer\_ID | INT |  |
| Product\_List\_View | | | | |
| PK | ID\_Product\_List\_View | INT | Первичный ключ таблицы «Состав товарной сметы» | create table [dbo].[Product\_List\_View]  (  [ID\_Product\_List\_View] [int] not null identity(1,1),  [Product\_List\_ID] [int] not null,  [Product\_ID] [int] not null  constraint [PK\_Product\_List\_View] primary key clustered  ([ID\_Product\_List\_View] ASC) on [PRIMARY],  constraint [FK\_Product\_List\_Product\_List\_View] foreign key ([Product\_List\_ID])  references [dbo].[Product\_List] ([ID\_Product\_List]),  constraint [FK\_Product\_Product\_List\_View] foreign key ([Product\_ID])  references [dbo].[Product] ([ID\_Product])  )  go |
| FK | Product\_List\_ID | INT |  |
| FK | Product\_ID | INT |  |
| Cancellation\_View | | | | |
| PK | ID\_Cancellation\_View | INT | Первичный ключ таблицы «Состав отмены приёма» | create table [dbo].[Cancellation\_View]  (  [ID\_Cancellation\_View] [int] not null identity(1,1),  [Cancellation\_ID] [int] not null,  [Product\_ID] [int] not null  constraint [PK\_Cancellation\_View] primary key clustered  ([ID\_Cancellation\_View] ASC) on [PRIMARY],  constraint [FK\_Cancellation\_Cancellation\_View] foreign key ([Cancellation\_ID])  references [dbo].[Cancellation] ([ID\_Cancellation]),  constraint [FK\_Product\_Cancellation\_View] foreign key ([Product\_ID])  references [dbo].[Product] ([ID\_Product])  )  go |
| FK | Cancellation\_ID | INT |  |
| FK | Product\_ID | INT |  |
| Supply | | | | |
| PK | ID\_Supply | INT | Первичный ключ таблицы «Поставка» | create table [dbo].[Supply]  (  [ID\_Supply] [int] not null identity(1,1),  [Integrity] [varchar] (30) not null,  [Supply\_Number] [varchar] (15) not null,  [Date\_Create\_Supply] [date] null default (getdate()),  [Time\_Create\_Supply] [time] null default (getdate()),  [Warehouse\_ID] [int] not null,  [Treaty\_ID] [int] not null  constraint [PK\_Supply] primary key clustered  ([ID\_Supply] ASC) on [PRIMARY],  constraint [UQ\_Supply\_Number] unique ([Supply\_Number]),  constraint [CH\_Integrity] check ([Integrity] in ('Нормально','Нарушенное')),  constraint [CH\_Supply\_Number] check ([Supply\_Number] like ('[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]-[0-9][0-9]-П')),  constraint [FK\_Warehouse\_Supply] foreign key ([Warehouse\_ID])  references [dbo].[Warehouse] ([ID\_Warehouse]),  constraint [FK\_Treaty\_Supply] foreign key ([Treaty\_ID])  references [dbo].[Treaty] ([ID\_Treaty])  )  go |
|  | Integrity | VARCHAR (30) | «Нормально», «Нарушенное» |
|  | Supply\_Number | VARCHAR (15) | Уникальное поле, 0000000000-YY-П |
|  | Date\_Create\_Supply | DATE | Текущая дата |
|  | Time\_Create\_Supply | TIME | Текущее время |
| FK | Warehouse\_ID | INT |  |
| FK | Treaty\_ID | INT |  |
| Supply\_View | | | | |
| PK | ID\_Supply\_View | INT | Первичный ключ таблицы «Состав поставки» | create table [dbo].[Supply\_View]  (  [ID\_Supply\_View] [int] not null identity(1,1),  [Supply\_Ammount] [int] not null,  [Supply\_ID] [int] not null,  [Product\_ID] [int] not null  constraint [PK\_Supply\_View] primary key clustered  ([ID\_Supply\_View] ASC) on [PRIMARY],  constraint [CH\_Supply\_Ammount] check ([Supply\_Ammount] > 0),  constraint [FK\_Supply\_Supply\_View] foreign key ([Supply\_ID])  references [dbo].[Supply] ([ID\_Supply]),  constraint [FK\_Product\_Supply\_View] foreign key ([Product\_ID])  references [dbo].[Product] ([ID\_Product])  )  go |
|  | Supply\_Ammount | INT | Не отрицательное |
| FK | Supply\_ID | INT |  |
| FK | Product\_ID | INT |  |
| Sale\_Check | | | | |
| PK | ID\_Sale\_Check | INT | Первичный ключ таблицы «Чек» | create table [dbo].[Sale\_Check]  (  [ID\_Sale\_Check] [int] not null identity(1,1),  [Sale\_Check\_Number] [varchar] (13) not null,  [Date\_Create\_Check] [date] null default(getdate()),  [Time\_Create\_Check] [time] null default(getdate()),  [Deposited\_Ammount] [decimal] (38,2) not null,  [Whith\_VAT] [decimal] (38,2) not null,  [Without\_VAT] [decimal] (38,2) not null,  [Payment\_Type\_ID] [int] not null,  [Employee\_Change\_ID] [int] not null,  [Buyer\_Card\_ID] [int] not null  constraint [PK\_Sale\_Check] primary key clustered  ([ID\_Sale\_Check] ASC) on [PRIMARY],  constraint [UQ\_Sale\_Check\_Number] unique ([Sale\_Check\_Number]),  constraint [CH\_Sale\_Check\_Number] check ([Sale\_Check\_Number] like '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]-ФК'),  constraint [CH\_Deposited\_Ammount] check ([Deposited\_Ammount] > 0),  constraint [CH\_Whith\_VAT] check ([Whith\_VAT] > 0),  constraint [CH\_Without\_VAT] check ([Without\_VAT] > 0),  constraint [FK\_Payment\_Type\_Sale\_Check] foreign key ([Payment\_Type\_ID])  references [dbo].[Payment\_Type] ([ID\_Payment\_Type]),  constraint [FK\_Employee\_Change\_Sale\_Check] foreign key ([Employee\_Change\_ID])  references [dbo].[Employee\_Change] ([ID\_Employee\_Change]),  constraint [FK\_Buyer\_Card\_Sale\_Check] foreign key ([Buyer\_Card\_ID])  references [dbo].[Buyer\_Card] ([ID\_Buyer\_Card])  )  go |
|  | Sale\_Check\_Number | VARCHAR (13) | Уникальное поле, 0000000000-ФК |
|  | Date\_Create\_Check | DATE | Текущая дата |
|  | Time\_Create\_Check | TIME | Текущее время |
|  | Deposited\_Ammount | DECIMAL (38,2) | Не отрицательное |
|  | Whith\_VAT | DECIMAL (38,2) | Не отрицательное |
|  | Without\_VAT | DECIMAL (38,2) | Не отрицательное |
| FK | Payment\_Type\_ID | INT |  |
| FK | Employee\_Change\_ID | INT |  |
| FK | Buyer\_Card\_ID | INT |  |
| Sale\_Check\_View | | | | |
| PK | ID\_ Sale\_Check\_View | INT | Первичный ключ таблицы «Состав чека» | create table [dbo].[Sale\_Check\_View]  (  [ID\_Sale\_Check\_View] [int] not null identity(1,1),  [Sale\_Check\_View\_Ammount] [int] not null,  [Sale\_Check\_ID] [int] not null,  [Product\_ID] [int] not null  constraint [PK\_Sale\_Check\_View] primary key clustered  ([ID\_Sale\_Check\_View] ASC) on [PRIMARY],  constraint [CH\_Sale\_Check\_View\_Ammount] check ([Sale\_Check\_View\_Ammount] > 0),  constraint [FK\_Sale\_Check\_Sale\_Check\_View] foreign key ([Sale\_Check\_ID])  references [dbo].[Sale\_Check] ([ID\_Sale\_Check]),  constraint [FK\_Product\_Sale\_Check\_View] foreign key ([Product\_ID])  references [dbo].[Product] ([ID\_Product])  )  go |
|  | Sale\_Check\_View\_Ammount | INT | Не отрицательное |
| FK | Sale\_Check\_ID | INT |  |
| FK | Product\_ID | INT |  |

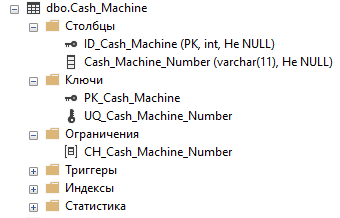
1. Продемонстрировать результат разработанной базы данных, приложив иллюстрации обозревателя объектов, среды разработки базы данных;

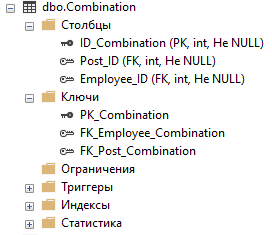


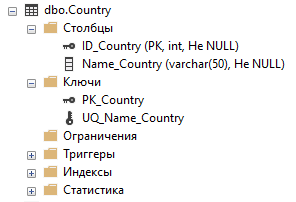


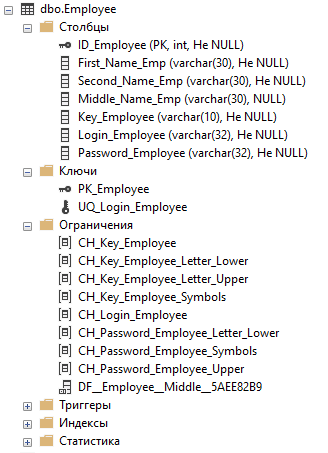


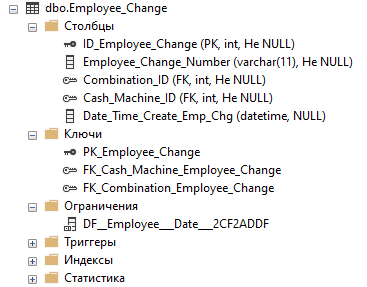


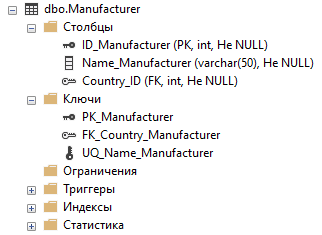


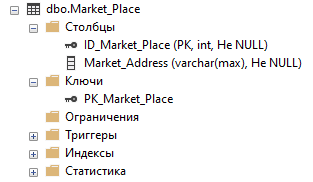


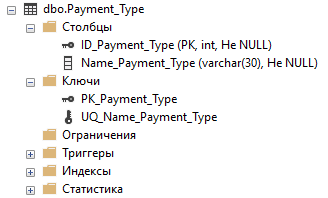


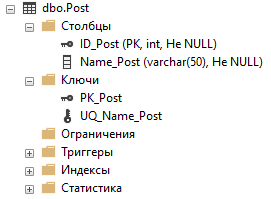


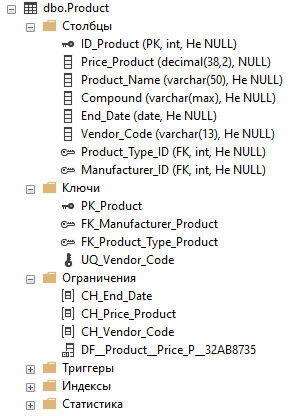


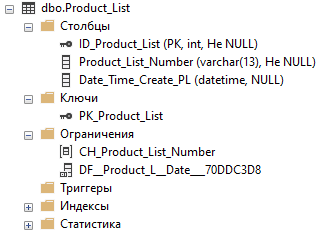


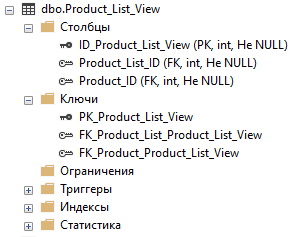


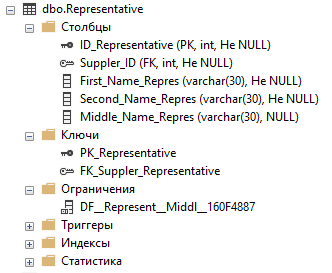


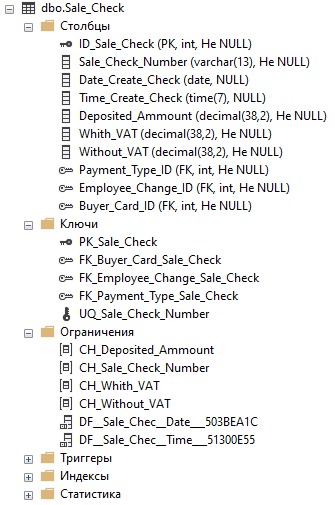


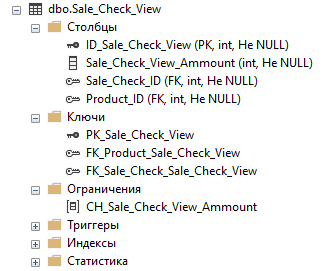


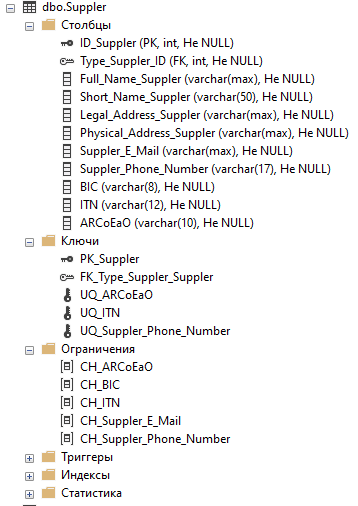


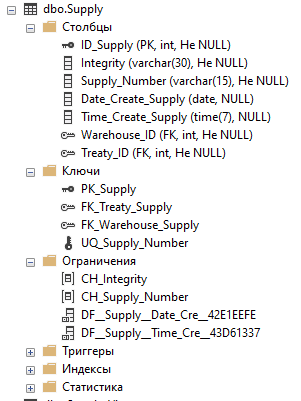


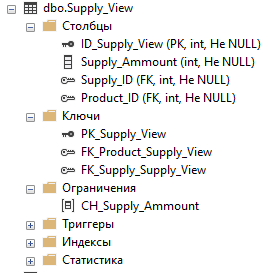


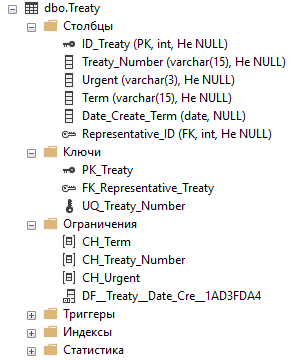


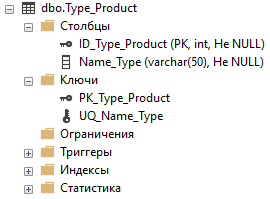


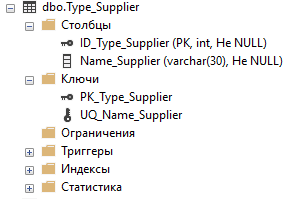


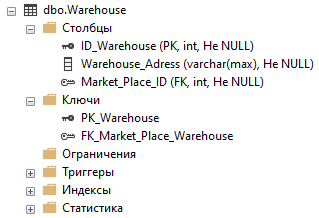












1. Продемонстрировать даталогическую модель данных, из инструментальной среды разработки баз данных.

