**Домашняя работа №8**

**Архитектура MS SQL**

**Задания**

1. Создайте вашу базу данных: таблицы, ограничения, ключи.

DROP TABLE IF EXISTS DimEquipment

CREATE TABLE DimEquipment (

[InventoryNumber] INT IDENTITY(1,1),

[Category] VARCHAR(50),

[Name] NVARCHAR (100) NOT NULL,

[RentalPrice] MONEY,

[SecurityDeposit] MONEY,

[VendorPrice] MONEY NOT NULL,

[VendorName] NVARCHAR (100) NOT NULL

CONSTRAINT PK\_DimEquipment\_InventoryNumber PRIMARY KEY ([InventoryNumber])

)

DROP TABLE IF EXISTS DimEmployees

CREATE TABLE DimEmployees (

[EmployeeNumber] INT IDENTITY(1,1),

[Position] NVARCHAR(50) NOT NULL,

[FullName] NVARCHAR(100) NOT NULL,

[FixSalary] MONEY NOT NULL,

[EmploymentDate] DATE NOT NULL,

[DismissalDate] DATE

CONSTRAINT PK\_DimEmployees\_EmployeeNumber PRIMARY KEY ([EmployeeNumber])

)

DROP TABLE IF EXISTS DimDiscount

CREATE TABLE DimDiscount (

[IDDiscount] INT IDENTITY(1,1),

[TypeOfDiscount] VARCHAR(50),

[DiscountPercentage] DEC (3,2),

[StartDate] DATE NOT NULL,

[EndDate] DATE,

CONSTRAINT PK\_DimDiscount\_IDDiscount PRIMARY KEY ([IDDiscount])

)

DROP TABLE IF EXISTS DimCustomers

CREATE TABLE DimCustomers (

[IDCustomer] INT IDENTITY(1,1),

[Category] VARCHAR (50),

[Name] NVARCHAR (50) NOT NULL,

[RegistrationNumber] CHAR (9) Unique,

[PassportNumber] CHAR (14) Unique,

[PhoneNumber] VARCHAR (20),

[Email] VARCHAR (50),

CONSTRAINT PK\_DimCustomers\_IDCustomer PRIMARY KEY ([IDCustomer])

)

DROP TABLE IF EXISTS FctMaintenance

CREATE TABLE FctMaintenance (

[IDMaintenance] INT IDENTITY(1,1),

[StartDate] DATE NOT NULL,

[EndDate] DATE,

[MaintenanceCost] MONEY,

[InventoryNumber] INT,

[EmployeeNumber] INT,

CONSTRAINT PK\_FctMaintenance\_IDMaintenance PRIMARY KEY ([IDMaintenance]),

CONSTRAINT FK\_FctMaintenance\_InventoryNumber FOREIGN KEY ([InventoryNumber])

REFERENCES DimEquipment ([InventoryNumber]),

CONSTRAINT FK\_FctMaintenance\_EmployeeNumber FOREIGN KEY ([EmployeeNumber])

REFERENCES DimEmployees ([EmployeeNumber])

)

DROP TABLE IF EXISTS FctOrders

CREATE TABLE FctOrders (

[IDOrder] INT IDENTITY(1,1),

[InventoryNumber] INT,

[EmployeeNumber] INT,

[BookingDate] DATE NOT NULL,

[RentDate] DATE,

[PlanReturnDate] DATE,

[FactReturnDate] DATE,

[IDPaymentType] INT,

[IDCustomer] INT,

[IDDiscount] INT,

CONSTRAINT PK\_FctOrders\_IDOrder PRIMARY KEY ([IDOrder]),

CONSTRAINT FK\_FctOrders\_InventoryNumber FOREIGN KEY ([InventoryNumber])

REFERENCES DimEquipment ([InventoryNumber]),

CONSTRAINT FK\_FctOrders\_EmployeeNumber FOREIGN KEY ([EmployeeNumber])

REFERENCES DimEmployees ([EmployeeNumber]),

CONSTRAINT FK\_FctOrders\_IDCustomer FOREIGN KEY ([IDCustomer])

REFERENCES DimCustomers ([IDCustomer]),

CONSTRAINT FK\_FctOrders\_IDDiscount FOREIGN KEY ([IDDiscount])

REFERENCES DimDiscount ([IDDiscount])

)

1. Напишите скрипт для получения 1 млн человек с различными именами и фамилиями. Полезная [ссылка](https://mockaroo.com/).

-- Загрузила файл с данными FirstName, LastName с <https://mockaroo.com/> в таблицу mock\_data\_customers\_individual

-- Создала таблицу dbo.Test

CREATE TABLE dbo.Test

(

ID INT PRIMARY KEY IDENTITY (1,1) NOT NULL,

FirstName NVARCHAR(100) collate Cyrillic\_General\_CI\_AS,

LastName NVARCHAR(100) collate Cyrillic\_General\_CI\_AS

);

-- Добавила данные из таблицы mock\_data\_customers\_individual с данными FirstName, LastName в таблицу dbo.Test

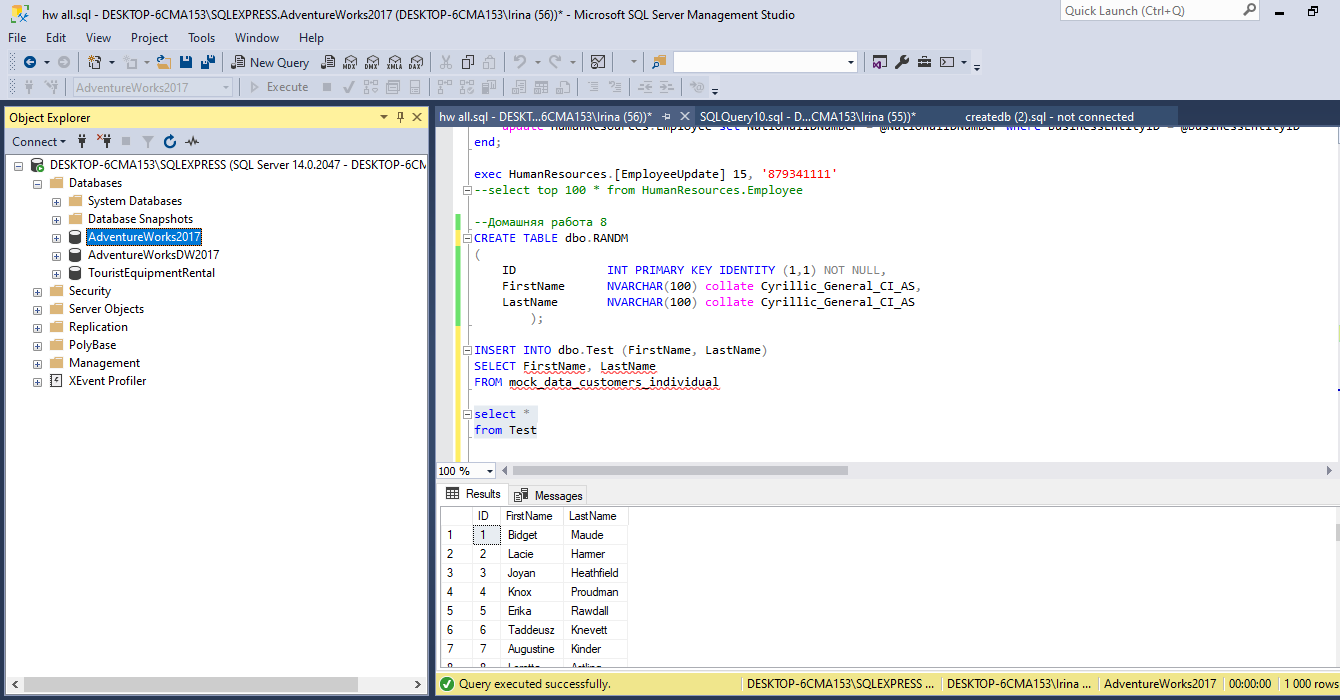
INSERT INTO dbo.Test (FirstName, LastName)

SELECT FirstName, LastName

FROM mock\_data\_customers\_individual

select \*

from Test



1. Познакомьтесь с [обобщенным табличным выражением WITH](https://docs.microsoft.com/ru-ru/sql/t-sql/queries/with-common-table-expression-transact-sql?view=sql-server-ver15). Когда можем использовать данную структуру? Чем отличается от подзапроса?

Обобщенное табличное выражение или CTE (Common Table Expressions) - это временный результирующий набор данных, к которому можно обращаться в последующих запросах. Для написания обобщенного табличного выражения используется оператор WITH. Обобщенные табличные выражения позволяют существенно уменьшить объем кода, если многократно приходится обращаться к одним и тем же производным таблицам.