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

CREATE DATABASE Swimmingcompetitions

CREATE TABLE Competition

(

CompetitionID int identity (1,1),

CompetitionName nvarchar(100) not null,

DateEnd date not null,

DateStart date not null

PRIMARY KEY (CompetitionID)

)

CREATE TABLE CompetitionVenue

(

CompetitionVenueID int identity (1,1),

CompetitionVenueName nvarchar(50) not null,

CountryName nvarchar(50) not null,

CityName nvarchar(50) not null

PRIMARY KEY (CompetitionVenueID)

)

CREATE TABLE Bridge\_Competition\_CompetitionVenue

(

CompetitionID int,

CompetitionVenueID int,

FOREIGN KEY (CompetitionID) REFERENCES Competition (CompetitionID) ON DELETE CASCADE,

FOREIGN KEY (CompetitionVenueID) REFERENCES CompetitionVenue (CompetitionVenueID) ON DELETE CASCADE

)

CREATE TABLE Results

(

ResultID int identity (1,1),

SwimmerID int,

TypeID int,

CompetitionID int,

ResultTime datetime

PRIMARY KEY (ResultID)

FOREIGN KEY (SwimmerID) REFERENCES Swimmer (SwimmerID),

FOREIGN KEY (TypeID) REFERENCES TypeSwimming (TypeID),

FOREIGN KEY (CompetitionID) REFERENCES Competition (CompetitionID)

)

CREATE TABLE TypeSwimming

(

TypeID int identity (1,1),

Name nvarchar(50)

PRIMARY KEY (TypeID)

)

CREATE TABLE Swimmer

(

SwimmerID int,

FirstName nvarchar(50) not null,

LastName nvarchar(50) not null,

PassportNumber nvarchar(20),

BirthDate date,

Gender nchar(1),

Rank nvarchar(10),

UNIQUE(PassportNumber),

PRIMARY KEY (SwimmerID)

)

CREATE TABLE AnthropometricTest

(

AnthropometricTestID int identity (1,1),

SwimmerID int not null,

Date datetime not null,

Weight decimal(3,3) not null,

Height decimal(3,2) not null,

Chest decimal(3,2) not null,

Waist decimal(3,2) not null,

PRIMARY KEY (AnthropometricTestID),

FOREIGN KEY (SwimmerID) REFERENCES Swimmer (SwimmerID)

)

CREATE TABLE Trainers

(

TrainerID int,

FirstName nvarchar(50) not null,

LastName nvarchar(50) not null,

PassportNumber nvarchar(20),

Rank nvarchar(10),

UNIQUE(TrainerID),

PRIMARY KEY (TrainerID)

)

CREATE TABLE Bridge\_Swimmer\_Trainer

(

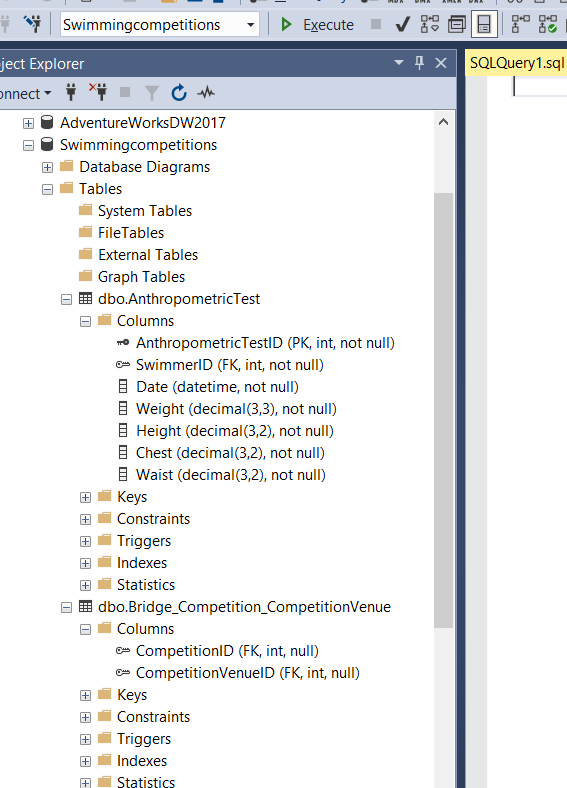
TrainerID int REFERENCES Trainers (TrainerID),

SwimmerID int REFERENCES Swimmer (SwimmerID),

FOREIGN KEY (TrainerID) REFERENCES Trainers (TrainerID),

FOREIGN KEY (SwimmerID) REFERENCES Swimmer (SwimmerID)

)



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

SELECT [FirstName], [LastName],IIF([Title] IN ('Mrs.', 'Sra.', 'Ms', 'Ms.'), 'F', 'M') AS Gender

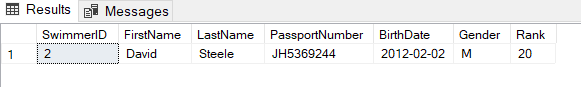
INTO #NamesGenders

FROM [AdventureWorks2017].[Person].[Person]

WHERE [Title] IS NOT NULL;

DECLARE @Name nvarchar(50) = (SELECT TOP 1 FirstName FROM #NamesGenders ORDER BY NEWID());

Получаю результат:



А вывести миллион не получается((

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?

Данную структуру используем в качестве альтернативы производным таблицам и представлениям

Отличия от подзапроса:

* Можно повысить скорость выполнения и написания sql-запроса с помощью предложения WITH
* Можно ссылаться на предложение WITH несколько раз
* WITH может быть использован для создания рекурсивных подзапросов
* WITH используется только с конструкциями SELECT, подзапрос в SELECT, WHERE, HAVING и FROM
* предложение WITH может содержать более одного запроса, в этом случае каждый запрос отделяется запятой