Задачи для программиста в ДИТ.

Версия: v08

# Задачи

Внимание! В начале каждого скрипта-решения укажите:

1. Вашу фамилию.
2. Номер и версию (см. в заголовке) задачи.
3. Решение присылайте без части скрипта создания (особенно без очистки) структуры таблиц.

Также обратите внимание на то, что вместе с данным документом-задачником должны присылаться SQL-скрипты, которые содержат скрипты создания структур тестовых БД, а иногда и генерации тестовых данных.

## Задача **1**

### Структура данных

SET NOCOUNT ON

SET ANSI\_NULLS ON

SET QUOTED\_IDENTIFIER ON

GO

--/\*

IF OBJECT\_ID('[TestDoc].[Accounts]') IS NOT NULL

DROP TABLE [TestDoc].[Accounts]

IF OBJECT\_ID('[TestDoc].[Contracts]') IS NOT NULL

DROP TABLE [TestDoc].[Contracts]

--\*/

--------------------------------------------------------------------------------------

-- Структура данных --

--------------------------------------------------------------------------------------

IF SCHEMA\_ID('TestDoc') IS NULL

EXEC('

CREATE SCHEMA [TestDoc]

')

GO

IF OBJECT\_ID('[TestDoc].[Contracts]') IS NULL

-- Договора

CREATE TABLE [TestDoc].[Contracts]

(

[Id] Int NOT NULL IDENTITY(1,1),

[DocNo] NVarChar(50) NOT NULL,

[DateFrom] Date NOT NULL, -- Дата, когда договор начал действовать

[DateTo] Date NULL, -- Дата, когда договор прекращает действовать (последний день действия договора); NULL = бесконечность

-- ... И еще какие-то поля

PRIMARY KEY CLUSTERED([Id])

)

GO

IF OBJECT\_ID('[TestDoc].[Accounts]') IS NULL

-- Счета

CREATE TABLE [TestDoc].[Accounts]

(

[Id] Int NOT NULL IDENTITY(1,1),

[Contract\_Id] Int NOT NULL, -- Договор, в рамках которого счет заключен

[Number] NVarChar(50) NOT NULL, -- Номер счета

[DateTimeFrom] DateTime NOT NULL, -- Момент времени (дата+время!), когда счет начал действовать

[DateTimeTo] DateTime NULL, -- Момент времени (дата+время!), когда счет прекратил действовать

-- ... И еще какие-то поля

PRIMARY KEY CLUSTERED([Id]),

FOREIGN KEY ([Contract\_Id]) REFERENCES [TestDoc].[Contracts] ([Id])

)

GO

Есть таблица контрактов [TestDoc].[Contracts]. У контракта есть период действия. Поле [DateFrom] – дата начала действия договора. Поле [DateTo] – последний день действия договора. NULL в поле [DateTo] – бесконечность.

Есть таблица счетов. Счета открываются в рамках договора. Поле [DateTimeFrom] – момент времени, когда счет открыли. [DateTimeTo] – момент времени, когда счета закрыли.

### Задание

Напишите скрипт, который вернет список «ошибок» в системе. Под «ошибкой» подразумевается ситуация, когда счет действует (действовал), а договор, в рамках которого заключен счет, не действует (не действовал).

## Задача **2**

### Структура

set nocount on

set ansi\_nulls on

set quoted\_identifier on

go

------------------------------------------------------------------

-- Создание структуры

------------------------------------------------------------------

if schema\_id('Test') is null

exec('create schema [Test]')

go

if object\_id('[Test].[Contracts]') is null

create table [Test].[Contracts]

(

[Id] int not null identity(1,1),

[Type\_Id] int not null,

[Client\_Id] int not null,

[DateFrom] date not null,

[DateTo] date null,

primary key clustered ([Id])

)

go

Есть таблица договоров ([Test].[Contracts]). У договора есть период действия. Также есть тип ([Type\_Id]) и указание клиента ([Client\_Id]).

Напишите скрипт, который вернет список всех неперсекающихся и несмежных периодов(т.е., между двумя периодами должен быть минимум один «пустой» день) между@DateBegin и @DateEnd включительно, когда у клиента был хотя бы один активный договор типа @Type\_Id. Если, можете

Пример параметров:

declare

@Type\_Id int = 1,

@DateBegin date = '20000601',

@DateEnd date = '20010131'

Структура результата:

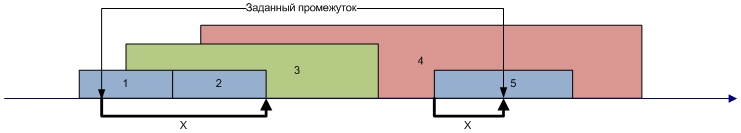
[Client\_Id] -- Клиент

, [First\_Date] -- Дата начала непрерывного периода действия договора(-ов) заданного типа

, [Last\_Date] -- Дата окончания непрерывного периода действия договора(-ов) заданного типа

Учесть, что у клиента может быть несколько действующих договоров одного типа одновременно.

## Пояснение:



## 

На данной картинке каждый прямоугольник – договор, расположенный на временной шкале. Цветом обозначен тип договора (т.е. например, синие – с типом 1, зеленые с типом 2, красные с типом 3).

В заданном временном промежутке нужно найти все диапазоны времени, когда действовал хотя бы один договор «синего типа». На рисунке то, что нужно вернуть обозначено как «X».

## Задача 3

### Структура

if schema\_id('TestCars') is null

exec('create schema [TestCars]')

go

if object\_id('[TestCars].[Points]') is null

-- Точки

create table [TestCars].[Points]

(

[Id] int not null identity(1,1),

[Type\_Id] char(1) not null, -- тип точки: D = Склад; S = Магазин

primary key clustered ([Id]),

check ([Type\_Id] in ('D', 'S'))

)

go

if object\_id('[TestCars].[Cars]') is null

-- Машины

create table [TestCars].[Cars]

(

[Id] int not null identity(1,1),

[Capacity] int not null, -- Грузоподьемность

primary key clustered ([Id])

)

go

if object\_id('[TestCars].[Routes]') is null

-- Машины

create table [TestCars].[Routes]

(

[Id] int not null identity(1,1),

[Point\_Id] int not null,

[Car\_Id] int not null,

[Load] int not null, -- Изменение загрузки авто при посещении данной точки. +N = в авто дозагрузили N кг; -N = из машины выгрузили N кг

-- При этом в магазине точка разгружается, а при посещении склада загружается/дозагружается

[ArrivalTime] datetime not null,

[DepartureTime] datetime not null,

primary key clustered ([Id]),

foreign key ([Point\_Id]) references [TestCars].[Points] ([Id]),

foreign key ([Car\_Id]) references [TestCars].[Cars] ([Id])

)

go

Есть таблица точек [TestCars].[Points]. Точками являются склады или магазины.

Есть таблица машин [TestCars].[Cars]. У каждой машины может быть своя грузоподъемность.

Есть точки маршрута машин [TestCars].[Routes], которые посещены машинами.

В течение дня машина совершает несколько поездок. Каждая поездка начинается с загрузки машины на складе. Далее машина посещает несколько магазинов, в каждом из которых частично разгружается. Затем машина может прибыть на один из складов, дозагрузиться и начать следующий маршрут. Каждый маршрут всегда проезжается в течение одного календарного дня (нет маршрутов с началом и концом в разных днях).

Машина движется всегда с одной и той же скоростью, скорости разных машин могут отличаться. По одному и тому же маршруту может проехать несколько различных машин.

### Задание

1. найти топ-3 неэффективных маршрутов и топ-3 неэффективных машин по каждому из критериев:

1.1. Недозагруженность машины, т.е. отношение «загрузка машины/грузоподъемность» после выезда со склада в начале маршрута

1.2. Размер остатка в машине после посещения всех магазинов на маршруте

2. Найти самую быструю машину или доказать, что по представленным данным это невозможно сделать.

## Задача 4

### Структура

IF SCHEMA\_ID('TestMoney') IS NULL

EXEC('

CREATE SCHEMA [TestMoney]

')

GO

IF OBJECT\_ID('[TestMoney].[Clients]') IS NULL

CREATE TABLE [TestMoney].[Clients]

(

[Id] Int NOT NULL IDENTITY(1,1),

[Name] NVarChar(128) NOT NULL,

-- ... И еще какие-то поля

PRIMARY KEY CLUSTERED([Id])

)

GO

IF OBJECT\_ID('[TestMoney].[Currencies]') IS NULL

CREATE TABLE [TestMoney].[Currencies]

(

[Id] Int NOT NULL,

[CodeLat3] Char(3) COLLATE Cyrillic\_General\_BIN NOT NULL,

[Name] NVarChar(128) NOT NULL,

PRIMARY KEY CLUSTERED([Id])

)

GO

IF OBJECT\_ID('[TestMoney].[Currencies Rates]') IS NULL

CREATE TABLE [TestMoney].[Currencies Rates]

(

[Currency\_Id] Int NOT NULL, -- Валюту, курс которой указан

[BaseCurrency\_Id] Int NOT NULL, -- Базовая валюта; для пары USD/RUB в поле ВaseCurrency\_Id будет ссылка на RUB; в поле Currency\_Id - ссылка на USD; в поле Rate = курс = 60; т.е. 60 RUB = 1 USD

[Date] Date NOT NULL, -- Дата курса

[Rate] Numeric(32,8) NOT NULL, -- Собственно курс

[Volume] Numeric(18,8) NOT NULL, -- За количество; Например, за 10 000 Белорусских рублей дают 39.4419 Рублей; Rate = 39.4419; Volume = 10 000;

PRIMARY KEY CLUSTERED([Currency\_Id], [BaseCurrency\_Id], [Date]),

FOREIGN KEY ([Currency\_Id]) REFERENCES [TestMoney].[Currencies] ([Id]),

FOREIGN KEY ([BaseCurrency\_Id]) REFERENCES [TestMoney].[Currencies] ([Id])

)

-- Чтобы получить сумму Sb в @BaseCurrency\_Id, если у Вас есть сумма Sc в валюте Currency\_Id,

-- то надо найти на интересующую дату такую ближайшую по дате запись (курс есть не на каждый день),

-- где [BaseCurrency\_Id] = @BaseCurrency\_Id и [Currency\_Id] = @Currency\_Id и [Date] <= @Date,

-- умножить на Rate и разделить на Volume

-- т.е. Sb = Sc \* Rate / Volume на требуемую дату

GO

IF OBJECT\_ID('[TestMoney].[Operations]') IS NULL

CREATE TABLE [TestMoney].[Operations]

(

[Id] Int NOT NULL IDENTITY(1,1),

[Date] Date NOT NULL, -- Дата операции

[Client\_Id] Int NOT NULL, -- Клиент, по которому меняется баланс

[Value] Numeric(32,8) NOT NULL, -- Сумма, на которую меняется баланс

[Currency\_Id] Int NOT NULL, -- Валюта операции

[DocNo] NVarChar(32) NULL,

-- ... И еще какие-то поля

PRIMARY KEY CLUSTERED([Id]),

FOREIGN KEY ([Currency\_Id]) REFERENCES [TestMoney].[Currencies] ([Id]),

FOREIGN KEY ([Client\_Id]) REFERENCES [TestMoney].[Clients] ([Id])

)

GO

### Задание

Напишите скрипт, который вернет обороты по клиентам в следующем виде:

* Client\_Id – клиент;
* Currency\_Id – валюта;
* BaseBalance – изменение баланса клиента в базовой валюте = сумма всех операций, у которых Date >= @DateFrom и Date < @DateTo.

Порядок вывода данных:

* Client\_Id ASC,
* Currency\_Id ASC

При этом входящие параметры:

* @BaseCurrency\_Id – идентификатор базовой валюты, к которой надо привести сумму BaseCurrency\_Id;
* @DateFrom, @DateTo – период за который надо расчитать баланс клиента.

При написании скрипта предположите, что в таблице [TestMoney].[Currencies Rates] есть курсы для [BaseCurrency\_Id] = @BaseCurrency\_Id

(т.е. не надо усложнять задачу до кросс-курсов)

Условие: в таблице Operations очень много данных. @DateFrom и @DateTo, как правило, небольшой период (не более 1 недели при общей истории данных в Operations не менее 3-х лет)

Предложите хороший(е) индекс(ы) для данного запроса(ов).

Прокомментируйте/обоснуйте Ваш выбор. Какие были варианты. Почему «да»/«нет»?