**SQL**

*Задача №1*

Написать запрос, приводящий таблицу 1 Динамики курса валют (CurrencyDynamics) к виду таблицы 2 (CurrencyDynamicsRes).

Таблица 1

|  |  |  |
| --- | --- | --- |
| CURR | DT | VAL |
| USD | 24.10.2019 | 63,7997 |
| USD | 25.10.2019 | 63,8600 |
| USD | 26.10.2019 | 63,9966 |
| USD | 29.10.2019 | 69,8700 |
| USD | 30.10.2019 | 63,8320 |
| USD | 31.10.2019 | 63,8734 |
| EUR | 24.10.2019 | 70,9644 |
| EUR | 25.10.2019 | 71,1400 |
| EUR | 26.10.2019 | 71,1194 |
| EUR | 29.10.2019 | 70,8382 |
| EUR | 30.10.2019 | 70,7769 |
| EUR | 31.10.2019 | 71,0081 |

Таблица 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CURR | DT\_FROM | VAL\_FROM | DT\_TO | VAL |
| EUR | 24.10.2019 | 70,9644 | 25.10.2019 | 71,1400 |
| EUR | 25.10.2019 | 71,1400 | 26.10.2019 | 71,1194 |
| EUR | 26.10.2019 | 71,1194 | 29.10.2019 | 70,8382 |
| EUR | 29.10.2019 | 70,8382 | 30.10.2019 | 70,7769 |
| EUR | 30.10.2019 | 70,7769 | 31.10.2019 | 71,0081 |
| EUR | 31.10.2019 | 71,0081 |  |  |
| USD | 24.10.2019 | 63,7997 | 25.10.2019 | 63,8600 |
| USD | 25.10.2019 | 63,8600 | 26.10.2019 | 63,9966 |
| USD | 26.10.2019 | 63,9966 | 29.10.2019 | 69,8700 |
| USD | 29.10.2019 | 69,8700 | 30.10.2019 | 63,8320 |
| USD | 30.10.2019 | 63,8320 | 31.10.2019 | 63,8734 |
| USD | 31.10.2019 | 63,8734 |  |  |

*Задача №2*

В БД определена таблица [dbo].[Problem\_2\_Test] - таблица со следующими полями:

1 ID - уникальный идентификатор (Primary Key)  
2 Name - ФИО  
  
Некоторые записи удалены. То есть, существуют пропущенные ID.  
  
Задача: Необходимо написать sql-запрос, который выводит на экран минимальный пропущенный ID.  
  
\* В рамках данной задачи считается, что всегда существуют запись с ID = 1 и между 1 и max(ID) существует хотя бы один пропуск.  
  
Пример: Пропущенные значения ID - 2, 4, 6, 9. Ответ - 2.

|  |  |
| --- | --- |
| **ID** | **ФИО** |
| 1 | Иванов А. А. |
| 3 | Иванов А. Б. |
| 5 | Иванов А. В. |
| 7 | Иванов А. Г. |
| 8 | Иванов А. Д. |
| 10 | Иванов А. Ж. |

Для решения данной задачи предлагаем воспользоваться ресурсом: <https://sqliteonline.com/>.  
Код создания таблицы ниже:

create table [dbo].[Problem\_2\_Test]  
(  
  ID int primary key,  
  FIO nvarchar(255) not null  
)  
insert into [dbo].[Problem\_2\_Test]  
values(1, 'Ivanov A. A.'),  
(3, 'Ivanov A. B.'),  
(5, 'Ivanov A. C.'),  
(7, 'Ivanov A. D.'),  
(8, 'Ivanov A. E.'),  
(10, 'Ivanov A. F.')

*Задача №3*

Таблица [dbo].[turnover]

В БД магазина имеется таблица [dbo].[turnover], в которой хранится информация об обороте товара на складах.  
Это сеть магазинов, поэтому складов несколько.  
В отдел аналитики поступил запрос на поиск случаев, когда какой-нибудь товар полностью пропадал со всех складов.  
Менеджеры хотят выявить такие периоды и разобраться почему так получилось.

Необходимо вывести:

* ID товара
* с какой даты пропал товар с полок
* по какую дату товар отсутствовал на полках

Работать предлагаем в <https://sqliteonline.com/>  
Просто вставьте туда код, расположенный ниже

create table [dbo].[turnover]

(

    ID int primary key identity(1, 1),

    branch\_id int not null, --ID филиала

    good\_id int not null, --ID товара

    oper\_date datetime not null, --дата операции

    oper\_type bit not null, --тип операции 1 - приход, 0 - расход

    good\_count int not null --количество товара в результате операции

)

insert into [dbo].[turnover] (branch\_id, good\_id, oper\_date, oper\_type, good\_count)

values(1, 1, '20200201 10:05:25', 1, 20),

(2, 1, '20200201 11:24:48', 1, 10),

(1, 2, '20200201 10:20:38', 1, 15),

(1, 1, '20200201 14:26:28', 0, 4),

(2, 1, '20200204 11:48:29', 0, 8),

(1, 1, '20200205 20:13:57', 0, 9),

(1, 1, '20200206 16:15:38', 1, 4),

(1, 1, '20200215 09:00:26', 0, 10),

(2, 1, '20200216 12:27:59', 0, 2),

(1, 1, '20200217 22:16:36', 0, 1),

(1, 2, '20200219 14:28:49', 0, 5),

(1, 1, '20200301 14:25:36', 1, 10),

(2, 1, '20200301 15:26:36', 1, 10),

(1, 2, '20200305 11:12:34', 0, 10),

(2, 1, '20200307 16:16:46', 0, 7)

*Задача №4*

Дан набор данных по маршрутам инкассации следующей структуры:

Таблица ROUTES

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Описание |
| ROUTE\_ID | number | Уникальный идентификатор маршрута |
| ROUTE\_START\_TIME | date | Дата и время выезда маршрута с базы |
| ROUTE\_END\_TIME | date | Дата и время возвращения маршрута на базу |

Требуется определить максимальное количество одновременно работающих маршрутов по всей выборке (даже если маршруты пересекаются на 1 секунду они считаются одновременно работающими).

Например: если маршрут 1 выехал в 10.00  и вернулся в 14.00, а маршрут 2 выехал в 13.00 и вернулся в 18.00 и больше никаких маршрутов не было, то в период с 13.00 до 14.00 одновременно работали 2 маршрута.

Во вложении тестовый набор данных dataset5.csv.

*Задача №5*

Дан набор данных по маршрутам инкассации следующей структуры:

Таблица OPERATIONS

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Описание |
| ROUTE\_ID | number | Уникальный идентификатор маршрута |
| OBJECT\_NUM | number | Порядковый номер объекта на маршруте |
| START\_TIME | date | Дата и время начала обслуживания клиента |
| END\_TIME | date | Дата и время окончания обслуживания клиента |

Требуется посчитать:

* 1. время операции в минутах
  2. как соотносится время текущей операции к среднему времени операции на маршруте (если больше -1, если меньше – 0)
  3. как соотносится время текущей операции к среднему времени трех предыдущих операций на маршруте (если больше -1, если меньше – 0)

Во вложении тестовый набор данных dataset6.csv и образец с решением в Excel (task 6 example.xlsx)

*Задача №6*

Есть таблица с транзакциями юр. лиц. Название table1.

Поля:  
company\_id – id компании  
acc\_numb – номер счета. У 1 компании может быть n-е количество счетов.  
tran\_dttm – дата и время транзакции  
tran\_sum – сумма транзакции  
tran\_nazn – назначение платежа, заполняется от руки

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| company\_id | acc\_numb | tran\_dttm | tran\_sum | tran\_nazn |
| 1 | 4\*\*\*\*\*\*01 | 10.05.2021 10:00 | 18234 | НДС не облагается |
| 1 | 4\*\*\*\*\*\*02 | 10.05.2021 10:40 | 100300 | Перевод на свою карту |
| 2 | 4\*\*\*\*\*\*40 | 18.05.2021 10:40 | 2000 | ЗП |
| 3 | 4\*\*\*\*\*\*39 | 18.04.2021 20:10 | 50031 | платеж по зарплате |
| 2 | 4\*\*\*\*\*\*40 | 19.04.2021 15:03 | 12335 | перевод зарплаты |
| 4 | 4\*\*\*\*\*\*88 | 20.03.2021 18:26 | 1327 | ЗАРПЛАТА ЗА ПЕРИОД \*\*\* |
| 5 | 4\*\*\*\*\*\*33 | 21.04.2021 22:00 | 23452 | оплата по договору |
| 1 | 4\*\*\*\*\*\*01 | 01.03.2021 23:48 | 42367 | налог |
| 1 | 4\*\*\*\*\*\*02 | 23.05.2021 21:00 | 23467 | перевод на карту |
| … | … | … | … | … |

Требуется:

1. Вывести ID компании, месяц (важно не потерять год), сумму всех транзакций компании.
2. Вывести тех, кто за последние 3 месяца потратил более 200тыс рублей на выплату заработной платы. Вывести id компании и сумму трат за 3 месяца.
3. Взять транзакции только мая, вывести всю информацию о последней транзакции по счету.
4. Необходимо посчитать сумму всех транзакций для каждого месяца (важно не потерять год) и вывести 3 месяца по убыванию с самыми большими суммами. (Если вдруг сумма совпадает, можно вывести больше чем 3 записи.)

Код создания таблицы ниже:

CREATE TABLE [dbo].[table1] (

company\_id integer NULL,

acc\_numb varchar(16) NULL,

tran\_dttm date NULL,

tran\_sum integer NULL,

tran\_nazn varchar(32) NULL

);

INSERT INTO [dbo].[table1] (company\_id, acc\_numb, tran\_dttm, tran\_sum, tran\_nazn)

VALUES

(1, '4\*\*\*\*\*\*01', '2021-05-10', 18234, 'НДС не облагается'),

(1, '4\*\*\*\*\*\*02', '2021-05-10', 1003000, 'Перевод на свою карту'),

(2, '4\*\*\*\*\*\*40', '2021-05-18', 2000, 'ЗП'),

(3, '4\*\*\*\*\*\*39', '2021-04-18', 50031, 'платеж по зарплате'),

(2, '4\*\*\*\*\*\*40', '2021-04-19', 12335, 'перевод зарплаты'),

(4, '4\*\*\*\*\*\*88', '2021-03-20', 1327, 'ЗАРПЛАТА ЗА ПЕРИОД \*\*\*'),

(5, '4\*\*\*\*\*\*33', '2021-04-21', 23452, 'оплата по договору'),

(1, '4\*\*\*\*\*\*01', '2021-03-01', 42367, 'налог'),

(1, '4\*\*\*\*\*\*02', '2021-05-23', 23467, 'перевод на карту');

*Задача №7*

Написать скрипты на языке SQL, чтобы получить список кредитов, которые на момент расчета имеют непогашенную задолженность, и рассчитать для каждого такого кредита:

1. Общую (накопленную) сумму просроченного долга непогашенную (не выплаченную) к моменту расчета.
2. Дату начала текущей просрочки. Под датой начала просрочки, в данной задаче понимается первая дата непрерывного периода, в котором общая сумма просроченного непогашенного долга> 0.
3. Кол-во дней текущей просрочки.

Исходные данные: таблица PDCL – содержит информацию о выносах на просрочку неоплаченных сумм по кредиту и о погашениях просроченного долга.

Структура:

|  |  |
| --- | --- |
| Date | Дата |
| Customer | Номер клиента |
| Deal | Номер кредита |
| Currency | Валюта кредита |
| Sum | сумма, вынесенная на просрочку ("+") или выплаченная ("-") |

Пример:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Customer | Deal | Currency | Sum |
| 12.12.2009 | 111110 | 111111 | RUR | 12000 |
| 25.12.2009 | 111110 | 111111 | RUR | 5000 |
| 12.12.2009 | 111110 | 122222 | RUR | 10000 |
| 12.01.2010 | 111110 | 111111 | RUR | -10100 |
| 20.11.2009 | 220000 | 222221 | RUR | 25000 |
| 20.12.2009 | 220000 | 222221 | RUR | -25000 |
| 21.12.2009 | 220000 | 222221 | RUR | 20000 |
| 29.12.2009 | 111110 | 122222 | RUR | -10000 |

Один клиент может иметь несколько кредитов. Если клиент, имеющий кредит, пропускает очередную выплату по графику, у него возникает просрочка. В таблице PDCL при этом появляется соответствующая запись, где Sum – невыплаченная сумма (с положительным знаком). Если, затем, клиент совершает выплату (полную сумму или ее часть) появляется новая запись, где Sum – выплаченная сумма (со знаком «-»). Следует учесть, что выплата клиента не обязательно полностью гасит накопленный долг, она может составлять лишь часть долга.

В приведенном примере по кредиту 111111 имеется невыплаченный просроченный долг на сумму 6900руб. (12000 + 5000 - 10100), дата начала текущей просрочки 12.12.2009.

Код создания таблицы:

CREATE TABLE [dbo].[pdcl] (

dt date NULL,

customer int4 NULL,

deal int4 NULL,

currency varchar(4) NULL,

sum\_op int4 NULL

);

INSERT INTO [dbo].[pdcl] (dt, customer, deal, currency, sum\_op)

VALUES

('2009-12-12', 111110, 111111, 'RUR', 12000),

('2009-12-25', 111110, 111111, 'RUR', 5000),

('2009-12-12', 111110, 122222, 'RUR', 10000),

('2010-01-12', 111110, 111111, 'RUR', -10100),

('2009-11-20', 220000, 222221, 'RUR', 25000),

('2009-12-20', 220000, 222221, 'RUR', -25000),

('2009-12-21', 220000, 222221, 'RUR', 20000),

('2009-12-29', 111110, 122222, 'RUR', -10000);

*Задача №8*

В таблице хранятся сведения о еженедельных продажах продукта.

Необходимо настроить автоматическое преобразование еженедельных значений в ежемесячные. Продажи в переходных неделях (часть недели в одном месяце, часть в другом) необходимо распределить по календарным дням.

В данном примере продажи за неделю, окончившуюся 05.03.2013, необходимо распределить следующим образом 2 дня на февраль, 5 дней на март.

Результатом решения задачи является SQL код, который позволит автоматически получать преобразованные данные в формате «Таблица результат». Таблица с данными условно называется test. Ограничения на использование различных инструментов отсутствуют.

Таблица с данными:

|  |  |
| --- | --- |
| Дата | Сумма |
| 26.02.2013 | 312,00 |
| 05.03.2013 | 833,00 |
| 12.03.2013 | 225,00 |
| 19.03.2013 | 453,00 |
| 26.03.2013 | 774,00 |
| 02.04.2013 | 719,00 |
| 09.04.2013 | 136,00 |
| 16.04.2013 | 133,00 |
| 23.04.2013 | 157,00 |
| 30.04.2013 | 850,00 |
| 07.05.2013 | 940,00 |
| 14.05.2013 | 933,00 |
| 21.05.2013 | 422,00 |
| 28.05.2013 | 952,00 |
| 04.06.2013 | 136,00 |
| 11.06.2013 | 701,00 |
|  | 8 676,00 |

Таблица результат:

|  |  |
| --- | --- |
| Дата мес | Сумма |
| 28.фев.13 | 550,00 |
| 31.мар.13 | 2 560,57 |
| 30.апр.13 | 1 481,43 |
| 31.май.13 | 3 305,29 |
| 30.июн.13 | 778,71 |
| 31.июл.13 | 0,00 |
|  | 8 676,00 |

Код создания таблицы:

CREATE TABLE [dbo].[test] (

REPORT\_DT date NULL,

VALUE int4 NULL

);

INSERT INTO [dbo].[test] (REPORT\_DT, VALUE)

VALUES

('2013-02-26', 312),

('2013-03-05', 833),

('2013-03-12', 225),

('2013-03-19', 453),

('2013-03-26', 774),

('2013-04-02', 719),

('2013-04-09', 136),

('2013-04-16', 133),

('2013-04-23', 157),

('2013-04-30', 850),

('2013-05-07', 940),

('2013-05-14', 933),

('2013-05-21', 422),

('2013-05-28', 952),

('2013-06-04', 136),

('2013-06-11', 701)