# Есть 5000 гуидов, надо оптимально вставить их в базу. Если такой id существует, то обновить дату.

Решение 1:

Создаем временную таблицу из 1000 id, делаем inner join с существующей таблице, и результат джоина обновляем, потом left join where table2.id = null , и на эту выборку вставка. И так 5 раз

Решение 2:

Балк инсерт (1000 id), а внутри MERGE. И так 5 раз.

Решение 3:  
Если My SQL, там нет Merge, поэтому

INSERT INTO t1 (id) VALUES ([1,2,3,4…])

ON DUPLICATE KEY UPDATE date = CURDATE();

# There are 2 tables

**CREATE TABLE [dbo].[Buyers](**

**[Id] [int] IDENTITY(1,1) NOT NULL,**

**[Name] [varchar](250) NOT NULL**

**) ON [PRIMARY]**

**CREATE TABLE [dbo].[Transactions](**

**[Id] [int] IDENTITY(1,1) NOT NULL,**

**[Cost] [decimal] (18, 0) NOT NULL,**

**[BuyerId] [int] NOT NULL,**

**) ON [PRIMARY]**

**The task is to select buyer with the maximum total sum of transactions. If there are multiple buyers with the same sum then show all of them.**

Решение0:

select b.Name, b. Id, Sum(t.Cost) as TotalCost from Buyers b

join Transactions t on b. Id = t. BuyerId

group by b.Id, b.Name

having sum(t.Cost) =

(

select top 1 Sum(t. Cost) as Total Cost from Transactions t

group by t.Buyer Id

order by TotalCost desc

)

Решение 1:

WITH BuyerTransactionSum AS (

SELECT

b.Id AS BuyerId,

b.Name AS BuyerName,

SUM(t.Cost) AS TotalTransactionSum

FROM

Buyers b

INNER JOIN

Transactions t ON b.Id = t.BuyerId

GROUP BY

b.Id, b.Name

)

SELECT

BuyerId,

BuyerName,

TotalTransactionSum

FROM

BuyerTransactionSum

WHERE

TotalTransactionSum = (

SELECT

MAX(TotalTransactionSum)

FROM

BuyerTransactionSum

);

Решение 2:

WITH BuyerTransactionSum AS (

SELECT

b.Id AS BuyerId,

b.Name AS BuyerName,

SUM(t.Cost) AS TotalTransactionSum,

**RANK() OVER (ORDER BY SUM(t.Cost) DESC) AS Rank**

FROM

Buyers b

INNER JOIN

Transactions t ON b.Id = t.BuyerId

GROUP BY

b.Id, b.Name

)

SELECT

BuyerId,

BuyerName,

TotalTransactionSum

FROM

BuyerTransactionSum

WHERE

Rank = 1;

# I have a table with columns A, B, C. There is an index that includes A and B. Will this index be used if I write a query with "where" conditions on columns A and C?

Whether the index on columns A and B will be used in a query with "where" conditions on columns A and C depends on the specifics of the query and the database engine's query optimizer. **In most cases, the index will not be directly used because column B is not directly involved in the query's filter conditions.**

However, depending on the selectivity of the columns and the cardinality of the index, the database engine might still choose to use the index on columns A and B. It might perform an index scan on the index to filter out rows based on column A and then perform additional filtering on column C.

The effectiveness of such an approach depends on various factors like the size of the table, distribution of data in columns A, B, and C, and the database engine's query optimizer capabilities.

# У меня много записи, но мало чтения. Как мне решить такую задачу?

Для обновления реплик баз данных на чтение существует несколько подходов:

1. **Транзакционная репликация**: В этом подходе изменения, внесенные в исходную (базу данных на запись), передаются транзакционным образом реплике (базе данных на чтение). Этот метод поддерживает более мгновенное обновление данных на реплике, так как изменения передаются по мере их возникновения.

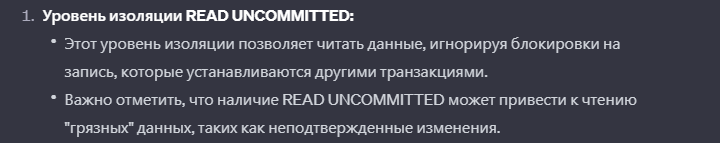
2. **Моментальные снимки (Snapshot Replication**): Этот метод включает создание моментального снимка исходной базы данных, который затем передается на реплику. Моментальные снимки могут быть запланированы на определенные интервалы времени с использованием задач (jobs) в SQL Server Agent.

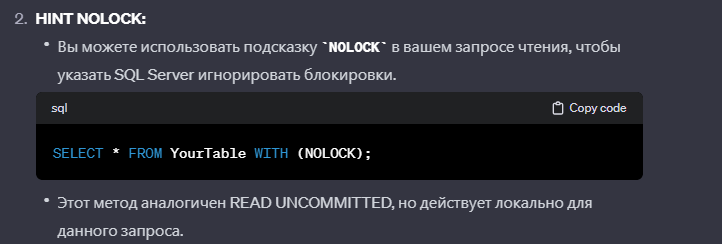
3. **Логическая репликация**: Этот метод основан на передаче логических изменений в данных (INSERT, UPDATE, DELETE) с исходной базы данных на реплику.Логическая репликация позволяет более гибко управлять тем, какие изменения передаются на реплику.

4. **Журнал изменений (Change Data Capture - CDC)**: CDC позволяет отслеживать изменения в таблицах и предоставлять их в виде событий изменений.

5 Availability Groups:Availability Groups в SQL Server предоставляют механизм для создания групп баз данных, которые могут быть автоматически реплицированы и обеспечивают высокую доступность.Этот метод может быть более сложным в настройке, но обеспечивает более высокую доступность и защиту от сбоев.

**sql как читать данные без блокировки на запись?**





Отличие with NoLock от read uncommited  - NoLock только для тек запроса! (никто не ответил!) и NoLock не лочит по дефолту **5000** записей