SQL Server Questions and Answers

FOLLOWERS

Followers (37) Next

Follow

LABELS

Backup and Restore CHECK Constraints Computed Column Date Puzzle DEFAULT Constraints File Group Function HAVING Interview Questions JOINs Key Constraints LOCK Log Shipping query hint Security SQL Challenge SQL Server Stored Procedure T-SQL Tally Table Transactions UDF WHERE XML Path

FEEDJIT LIVE TRAFFIC

Tips and Tricks for SQL Server Developers

MONDAY, OCTOBER 1, 2012

What is Tally Table in SQL Server?

In this post, I am going to explain about **Tally** table and it’s uses in T-SQL programming.

A **Tally** table is like any other table but having a single column of sequential numbers, values starting from 1 (or 0) to some N (int) number.

The largest number in the Tally table should be based on what suits your system, application, or database most. So don't use very high number. Also column of Tally should be indexed for better performance.

I use Tally table to generate 25 years of dates, so my Tally tables will have values from 1 to 10,000 (25 years * 365.25 days = 9131.25)

How to Create a Tally Table

There are several methods to create a Tally table. I will use one of the simplest and obvious option - WHILE loop because it's easier to explain and simpler than others.

```
SET NOCOUNT ON;
IF OBJECT_ID('dbo.Tally') IS NOT NULL  DROP TABLE dbo.Tally
GO

-- Define how many rows you want in Tally table.
-- I am inserting only 10000 rows
SET ROWCOUNT 10000

SELECT IDENTITY(INT, 1, 1) ID
INTO dbo.Tally
FROM master.sys.all_columns c
CROSS JOIN master.sys.all_columns c1
-- you may use one more cross join if tally table required hundreds of million rows

SET ROWCOUNT 0

-- ADD (unique) clustered index
CREATE UNIQUE CLUSTERED INDEX PKC_Tally ON dbo.Tally (ID)
GO
```

How to use Tally table in T-SQL and what are the advantages?

There are several advantages of a Tally table. Here are some of the examples:

- To generate Date Range for given Start Date and End Date
- To Manipulate strings, like:
  - Find the positions of a character in a string.
  - Find the total occurances of a character in a string
  - Split comma seperated values

I will explain these advantages with examples.

Generate Date Range using Tally Table

Generally you would require a WHILE loop to create Date Range values. However, it is very easy to generate date range using tally. Its much faster than WHILE loop:

```
-- Generate Date range
DECLARE @BeginDate DATE = '2001-01-01', @EndDate DATE = '2025-12-31'
SELECT DATEADD(DD, ID-1, @BeginDate) [Date]
,DAY(DATEADD(DD, ID-1, @BeginDate)) [Day]
,MONTH(DATEADD(DD, ID-1, @BeginDate)) [Month]
,YEAR(DATEADD(DD, ID-1, @BeginDate)) [Year]
```

SEARCH THIS BLOG

BLOG ARCHIVE

- ▼ 2012 (7)
  - ► November (2)
  - ▼ October (2)
    - SQL Fun #01
    - What is Tally Table in SQ
  - ► September (2)
  - ► January (1)
- ► 2011 (6)
- ► 2010 (38)

POPULAR POSTS

Explain SQL Server JOINs w

What is Tally Table in SQL S

Types of User Defined Func SQL Server 2008?

How to get SQL Server Rest using T-SQL?

How to CREATE Foreign Key table?

How to ADD CHECK CONSTF existing table?

How to get Month Number Name

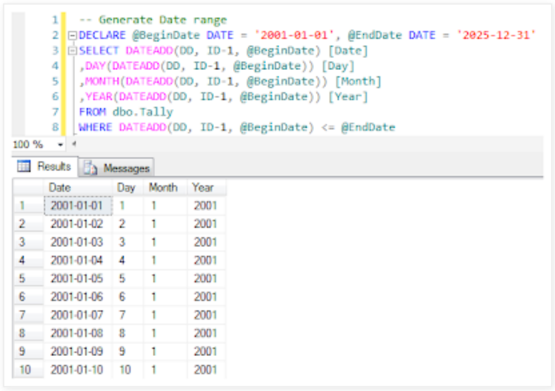
What is the use UPDATE_ST command?

How to Find Database Grov SQL

What is Log Shipping in SQL

```
FROM dbo.Tally
WHERE ID <= DATEDIFF(DD, @BeginDate, @EndDate) + 1
```

Here is the output:



```
-- Generate Date range
DECLARE @BeginDate DATE = '2001-01-01', @EndDate DATE = '2025-12-31'
SELECT DATEADD(DD, ID-1, @BeginDate) [Date]
,DAY(DATEADD(DD, ID-1, @BeginDate)) [Day]
,MONTH(DATEADD(DD, ID-1, @BeginDate)) [Month]
,YEAR(DATEADD(DD, ID-1, @BeginDate)) [Year]
FROM dbo.Tally
WHERE DATEADD(DD, ID-1, @BeginDate) <= @EndDate
```

	Date	Day	Month	Year
1	2001-01-01	1	1	2001
2	2001-01-02	2	1	2001
3	2001-01-03	3	1	2001
4	2001-01-04	4	1	2001
5	2001-01-05	5	1	2001
6	2001-01-06	6	1	2001
7	2001-01-07	7	1	2001
8	2001-01-08	8	1	2001
9	2001-01-09	9	1	2001
10	2001-01-10	10	1	2001

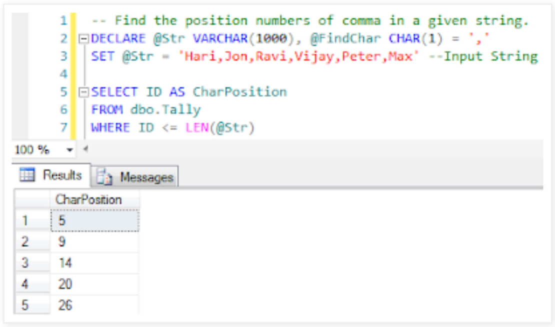
Find a Character Positions in a String using Tally Table

You can find a character position using string functions and WHILE loop. But Tally table makes it much simpler than any other method, yet faster. Here is an example:

```
-- Find the position numbers of comma in a given string.
DECLARE @Str VARCHAR(1000), @FindChar CHAR(1) = ','
SET @Str = 'Hari,Jon,Ravi,Vijay,Peter,Max' --Input String

SELECT ID AS CharPosition
FROM dbo.Tally
WHERE ID <= LEN(@Str)
AND SUBSTRING(@Str, ID, 1) = @FindChar
ORDER BY ID
```

Here is the output:



```
-- Find the position numbers of comma in a given string.
DECLARE @Str VARCHAR(1000), @FindChar CHAR(1) = ','
SET @Str = 'Hari,Jon,Ravi,Vijay,Peter,Max' --Input String

SELECT ID AS CharPosition
FROM dbo.Tally
WHERE ID <= LEN(@Str)
```

	CharPosition
1	5
2	9
3	14
4	20
5	26

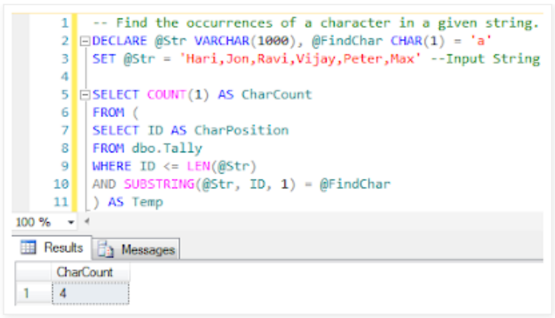
Find count of all the occurrences of a Character in a String using Tally Table

You can find the count of all the occurrences of a character in a string using Tally table by slightly modifying above query:

```
-- Find the occurrences of a character in a given string.
DECLARE @Str VARCHAR(1000), @FindChar CHAR(1) = 'a'
SET @Str = 'Hari,Jon,Ravi,Vijay,Peter,Max' --Input String

SELECT COUNT(1) AS CharCount
FROM (
SELECT ID AS CharPosition
FROM dbo.Tally
WHERE ID <= LEN(@Str)
AND SUBSTRING(@Str, ID, 1) = @FindChar
) AS Temp
```

Here is the output:


```
-- Find the occurrences of a character in a given string.
DECLARE @Str VARCHAR(1000), @FindChar CHAR(1) = 'a'
SET @Str = 'Hari,Jon,Ravi,Vijay,Peter,Max' --Input String

SELECT COUNT(1) AS CharCount
FROM (
SELECT ID AS CharPosition
FROM dbo.Tally
WHERE ID <= LEN(@Str)
AND SUBSTRING(@Str, ID, 1) = @FindChar
) AS Temp
```

	CharCount
1	4

Split Comma Seperated values using Tally table

I had posted a separate article about Function to Split Multi-valued String couple of years back.

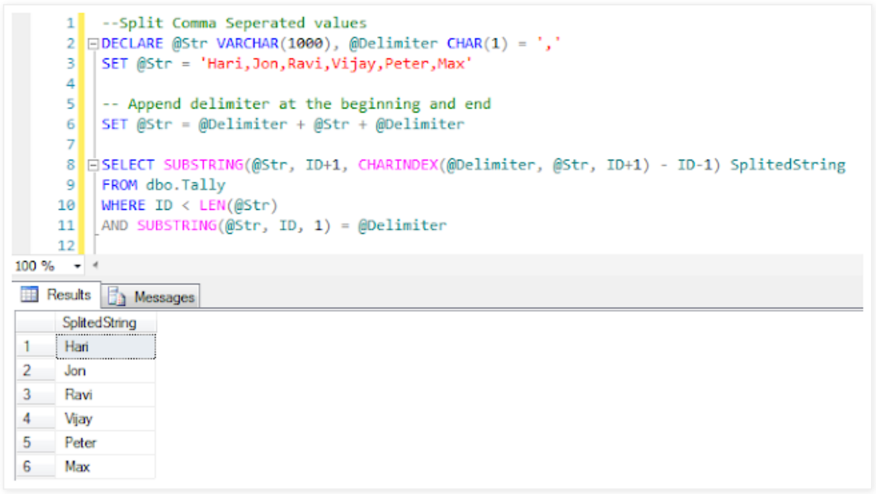
The logic implemented in that function could be much simpler by using Tally table. You can split the values without WHILE loop. It would be interesting to compare the performance of these two mechanism.

```
--Split Comma Seperated values
DECLARE @Str VARCHAR(1000), @Delimiter CHAR(1) = ','
SET @Str = 'Hari,Jon,Ravi,Vijay,Peter,Max'

-- Append delimiter at the beginning and end
SET @Str = @Delimiter + @Str + @Delimiter

SELECT SUBSTRING(@Str, ID+1, CHARINDEX(@Delimiter, @Str, ID+1) - ID-1) SplitedString
FROM dbo.Tally
WHERE ID < LEN(@Str)
AND SUBSTRING(@Str, ID, 1) = @Delimiter
```

Here is the output:


```
--Split Comma Seperated values
DECLARE @Str VARCHAR(1000), @Delimiter CHAR(1) = ','
SET @Str = 'Hari,Jon,Ravi,Vijay,Peter,Max'

-- Append delimiter at the beginning and end
SET @Str = @Delimiter + @Str + @Delimiter

SELECT SUBSTRING(@Str, ID+1, CHARINDEX(@Delimiter, @Str, ID+1) - ID-1) SplitedString
FROM dbo.Tally
WHERE ID < LEN(@Str)
AND SUBSTRING(@Str, ID, 1) = @Delimiter
```

	SplitedString
1	Hari
2	Jon
3	Ravi
4	Vijay
5	Peter
6	Max

Like Share 4 people like this. Sign Up to see what your friends like.

Labels: Interview Questions, SQL Challenge, SQL Server, T-SQL, Tally Table

13 comments:

Anonymous October 28, 2012 at 4:16 PM

When someone practices the piano and unless they're trying to be a comedian, one does not practice hitting the wrong notes.

I feel the same way about someone posting a WHILE loop solution to create a Tally Table. Yes, it's simple. Yes, people understand it. And, yes, it's hitting the wrong notes.

As you so very correctly pointed out, the Tally Table is used to avoid certain types of loops. Why would you taint what you're trying to teach by creating the Tally Table with a loop? Show people what you've got. Build the Tally Table without a loop and without the hidden RBAR of a counting recursive CTE.

Reply

Replies

Kevin January 12, 2016 at 7:24 AM

Jeff, I'm going to have to disagree with you. The purpose of this article was not to demonstrate the most efficient way to BUILD a tally table, but how and why you USE one. That's what I came here looking for, and that's exactly what I found. This article was extremely helpful to me. Brilliant, actually.

RBAR concerns don't apply to tasks that take a fraction of a second and are only going to be performed ONCE. Hari chose the simplest and easiest method to get the job done, which is exactly what beginners need to know. A good teacher knows how keep from overloading the students with all the complex details up front; otherwise, they'll never make it to the conclusion.

Now, would I like to read a post on how to build simple sequences in SQL Server? Sure. But that's not what this article was for. This article is perfect the way it is. (But, Hari, if you choose to write one on that, let me know. I'll be sure to read it!)

Anonymous February 13, 2016 at 11:16 AM

Kevin. The point is to teach the best ways to do everything. Using a WHILE Loop to generate a Tally Table does nothing but to perpetuate the wrong type of thinking even if the task is supposedly only being done once.

Anonymous February 13, 2016 at 11:26 AM

p.s. BTW, I'm currently fixing a database full of thinks that take a "fraction" of a second because they're way to slow for the volume of usage. For example, I'm currently working on several stored procedures and "managed code" snippets that only take 250ms each. Unfortunately, each of those are called 40,000 times every 8 hours. That means that each of those "fraction of a second" items are consuming 2.7 hours of CPU time every 8 hours. Every 10 of those (and there are more than that) consume 27 hours of CPU time every 8 hours. That means that more the 3 CPUs are working full time during those 8 hours just to handle those 10 snippets. They were all written by people with the same mentality that some things don't matter.

To coin a phrase, "Sweat all the small stuff and everything is small stuff". ;-)

Reply

Hari Sharma October 28, 2012 at 4:31 PM

Thanks for the great comment, Jeff.
I totally agree with you regarding the perf hit while generating Tally table using given method. I generally prefer using system tables (one of the solution could be cross join with sys.columns) rather than LOOP.

While writing the article, I kept in my mind that this is one-time load and not a destructive load on daily basis. And of course, I concur that we should always use best method instead of shortcuts :)

Reply

Replies

Anonymous December 31, 2012 at 8:56 PM

Despite your concurance, you still took the second worst "shortcut" there is. Building sequences is an important part of SQL Server. It's not just for 1 time things like building a Tally Table. Help the newbies out that might read your good article. Change the code in your article to show one of the right ways instead of using a loop. Set the example. ;-)

Reply

SRIDHAR REDDY October 31, 2012 at 3:00 AM

```
SELECT dateadd(dd,a.number*1025+b.number,@BeginDate) ,
datepart( dd , dateadd(dd,a.number*1025+b.number,@BeginDate) ),
datepart( mm , dateadd(dd,a.number*1025+b.number,@BeginDate) ),
datepart( yy , dateadd(dd,a.number*1025+b.number,@BeginDate) )
FROM master..spt_values a
, master..spt_values b
WHERE a.type='P'
AND b.type='P'
AND a.number<=1024
AND b.number<=1024
AND dateadd(dd,a.number*1025+b.number,@BeginDate )<= @EndDate
```

Reply

Hari Sharma November 2, 2012 at 6:32 PM

@Sridhar - Good solution. Albeit, I am wondering how come your solution is better than the one explained in the post once you have Tally table ready in your database?

IMHO, the query written in the article is much simpler and faster than what you have.

Reply

Replies

Anonymous January 26, 2013 at 10:29 AM

Agreed. Lot's easier to read and maintain, as well.

Reply

Anonymous January 26, 2013 at 10:13 AM

So, considering that you agreed with my statement and some reasonable period of time has passed, I have to ask... are ya gonna fix it? ;-)

Reply

Jonathan B August 19, 2016 at 7:00 AM

Thanks for the great article. I got your point that I think Jeff missed which was, this article isn't about how to efficiently build a tally table (which would only be run once anyhow). I'm sure if you added that in some, including myself, would have gotten bogged down in that instead of learning about how useful tally tables are.

Reply

Jonathan B August 19, 2016 at 7:00 AM

Thanks for the great article. I got your point that I think Jeff missed which was, this article isn't about how to efficiently build a tally table (which would only be run once anyhow). I'm sure if you added that in some, including myself, would have gotten bogged down in that instead of learning about how useful tally tables are.

Reply

Rigid Box March 15, 2019 at 1:28 AM

Information from this blog is very useful for me, am very happy to read this blog Kindly visit us @ Luxury Watch Box | Shoe Box Manufacturer | Candle Packaging Boxes

Reply

Enter your comment...

Comment as: Google Accou ▼

Publish **Preview**

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)

Powered by Blogger.