Complete Common SQL Server Database Administration Tasks In Parallel with PowerShell V3 WorkFlow

Written By: Jeffrey Yao -- 3/6/2015

Problem

In my daily work as a SQL Server DBA, there are many times I want to do things in parallel, for example:

- I have a weekly full backup job for multiple big databases on a server, each job step does a full backup of one database. However due to the size of each database, the backup can take many hours. What if I can do backup in parallel, i.e. one backup session for one database in parallel.
- I have a daily index maintenance job that will rebuild or defrag indexes, but it takes a long time to do maintenance on each index one by one. What if I can do index maintenance in parallel, i.e. one maintenance session on one index with multiple sessions in parallel.
- When I push out a code deployment, there are many T-SQL scripts that can be executed in parallel either on different databases and/or on different servers.

Is there an easy way for me to do the above mentioned SQL Server DBA tasks in parallel, so I can save as much time as possible?

Solution

Starting with PowerShell (PS here after) V3, there is a new feature called WorkFlow, which can execute PS cmdlets in parallel. So in this tip, I will first use a simple example to demonstrate how the PS workflow feature will help DBAs in reducing administration time, and later I will come up with two practical solutions about doing backups in parallel or running a script in parallel against multiple SQL Server instances.

Example 1 - Parallel Execution of Multiple T-SQL Scripts

This example will run this T-SQL statement five times, which is just a delay of 30 seconds:

```
waitfor delay '00:00:30';
```

In theory, if we run this T-SQL statement in sequence, it will take 30 X 5 = 150 seconds, but if we run the code in parallel, it should take only 30 seconds. So here is the code to verify this concept (i.e. run in parallel via PS):

```
#requires -version 3.0

#assume SQLPS module is installed (which comes with sql server 2012)
Import-Module sqlps -DisableNameChecking;
set-location c:
#create a workflow to run multiple sql in parallel
WorkFlow Run-PSQL #PSQL means Parallel SQL
{
    param(
        [Parameter(Mandatory=$true)]
        [string]$ServerInstance,

        [Parameter(Mandatory=$false)]
        [string]$Database,

        [Parameter(Mandatory=$true)]
        [string[]]$Query #a string array to hold t-sqls
    )

    foreach -parallel ($q in $query)
    { invoke-sqlcmd -ServerInstance $ServerInstance -Database $Database -Query $q -querytimeout 60000;
```

When I run this PS script in my PS IDE, the final result is about 34 seconds as shown below:

```
#now we can run the workflow and measure its execution duration

$dt_start=get-date; #start time
Run-PSQL -Server tp_w520 -database master -query $sqlcmds;
$dt_end = get-date; #end time
$dt_end -$dt_start; # how long it will take
```

```
Days              : 0
Hours             : 0
Minutes           : 0
Seconds           : 34
Milliseconds      : 147
Ticks             : 341479532
TotalDays         : 0.000395230939814815
TotalHours        : 0.00948554255555556
TotalMinutes      : 0.569132553333333
TotalSeconds      : 34.1479532
TotalMilliseconds : 34147.9532
```

Actually, when I was running the PS script, I opened an SSMS window, and ran the T-SQL code to check the time for the running code as shown below:

```
select r.session_id, r.start_time, st.text
from sys.dm_exec_requests r
cross apply sys.dm_exec_sql_text(r.sql_handle) as st
where session_id > 50
and st.text like 'waitfor delay%'
```

esults | Messages

session...	start_time	text
56	2015-02-21 21:22:11.350	waitfor delay '00:00:30'
57	2015-02-21 21:22:11.550	waitfor delay '00:00:30'
58	2015-02-21 21:22:11.813	waitfor delay '00:00:30'
60	2015-02-21 21:22:12.090	waitfor delay '00:00:30'
61	2015-02-21 21:22:12.333	waitfor delay '00:00:30'

You can see that the 5 sessions start almost at the same time (there is only a 1 second difference between the first and last session start time).

Example 2 - Parallel Execution of SQL Server Database Backups

When I have multiple (like 100+) databases on a SQL Server instance, I'd like to have multiple sessions complete the backups so I can shorten the backup time.

The solution design has three key points:

1. With SQL PSProvider, we list all user databases by using 'dir'.
2. Loop through the databases, and compose the corresponding backup T-SQL statement, and put all the T-SQL code into a string array
3. Use PS workflow Run-PSQL to run the T-SQL code in the array of Step 2

```
#requires -version 3.0

#assume SQLPS module is installed (which comes with sql server 2012)
Import-Module sqlps -DisableNameChecking;
set-location c:
```

```
#create a workflow to run multiple sqls against one sql instance
WorkFlow Run-PSQL #PSQL means Parallel SQL
{
    param(
        [Parameter(Mandatory=$true)]
        [string]$ServerInstance,

        [Parameter(Mandatory=$false)]
        [string]$Database,

        [Parameter(Mandatory=$true)]
        [string[]]$Query
    )

    foreach -parallel ($q in $query)
    { invoke-sqlcmd -ServerInstance $ServerInstance -Database $Database -Query $q -querytimeout 60000;
```

Example 3 - Execute the Same T-SQL Script Against Multiple SQL Server Instances in Parallel

Running same T-SQL code against multiple SQL Server instances is also a common task for DBAs. So the following code will demonstrate how to do this in parallel:

```
#requires -version 3.0

#assume SQLPS module is installed (which comes with sql server 2012)
Import-Module sqlps -DisableNameChecking;
set-location c:
#create a workflow to run one script against multiple sql instances
WorkFlow Run-PSQL2 #PSQL means Parallel SQL
{
    param(
        [Parameter(Mandatory=$true)]
        [string[]]$ServerInstance,  # string array to hold multiple sql instances

        [Parameter(Mandatory=$false)]
        [string]$Database,

        [Parameter(Mandatory=$true)]
        [string]$FilePath # filepath to the t-sql script to be run
    )

    foreach -parallel ($s in $ServerInstance)
    { invoke-sqlcmd -ServerInstance $s -Database $Database -InputFile $FilePath -querytimeout 60000;  }
```

Summary

Running T-SQL scripts in parallel is a great way to boost DBA work efficiency and productivity. With the use of PowerShell V3+, parallel execution becomes available and is easy to implement.

I hope this tip will broaden your choices when designing / architecting your DBA tasks / tools.

One note, in PS V3, the parallel execution has a fixed hard limit of 5 concurrent threads when a workflow runs. I have tested lots of scenarios running T-SQL code, either via invoke-sqlcmd or by SMO methods, it seems each thread (among the 5 concurrent threads) in a workflow does not really start at the same time and can have 0.1 to 9+ seconds difference in starting time. This means the parallel execution is more useful when T-SQL scripts run a long time. If each script runs 0.01 seconds (like creating tables, stored procedures, etc.), it really does not matter that much whether you run scripts in parallel or in sequence.

Next Steps

Use the code above and test in your own environment. I have tested the PS script in my various test environments where PS V3/V4, Windows 7, Windows 2012 R2 or Windows 2008 R2 installed. Also make sure to use SQL Server 2012+ with SQL PS module is installed on the host computer where you will run the PS script.

You may also check the following links to learn more about PS workflow parallel activity behavior, or try coming up with a new method to solve an old issue.

- Use PS Workflow to Ping Computers in Parallel
- How to throttle workflow activities in PS V4
- Reduce Time for SQL Server Index Rebuilds and Update Statistics
- Review your current home-made DBA scripts/tools and see whether you can modify them to take advantage of parallel execution.

Follow

Get Free SQL Tips

Twitter

LinkedIn

Google+

Facebook

Pinterest

RSS

Learning

DBAs

Developers

BI Professionals

Careers

Q and A

Today's Tip

Resources

Tutorials

Webcasts

Whitepapers

Tools

Search

Tip Categories

Search By TipID

Top Ten

Authors

Community

First Timer?

Pictures

Free T-shirt

Contribute

Events

User Groups

Author of the Year

More Info

Join

About

Copyright

Privacy

Disclaimer

Feedback

Advertise