Search   Go

Write For Us
Find out more

★★★★☆ Rate this | 💬   Thank this author by sharing: g+1 in f 𝕍 6
Join the discussion | 📂 Add to briefcase | 🖨 Print

Parallel Processing Using Jobs

By Peter Sloboda, 2013/04/29

Introduction:

We've all been there. Long queues of reports are piling up, your inbox is full of SQL requests, or end of the month accounting craziness breaks out. The following is a simple way of handling these, mostly uniform, tasks, which usually run in serial fashion.

The idea is simple. Set up a queue, a table which holds the code/stored procedures which need to be executed. Loop over the queue, pick the command value, create a job through dynamic sql and pass all the values (name, command...), which vary from task to task. You then start the job and monitor number of jobs already running before starting a new one (last inner loop involving 'waitfor'). When the job finishes it will automatically self-destruct (@delete_level=3).

Here are the steps we'll follow to handle this.

1. We'll create the 'job queue' and fill it with sample 'waitfor delay' commands with variable wait time using rand() function. The jobs will just simulate the work.

Related Articles

FORUM   **msdb.dbo.sp_send_dbmail**

msdb.dbo.sp_send_dbmail

FORUM   **How do i grant permission to msdb.dbo.sp_send_dbmail ?**

How do i grant permission to msdb.dbo.sp_send_dbmail ?

ARTICLE   **Working with Queues in SQL Server**

This article shows sow to create a queue from a SQL table to serve each operator exactly one item fr...

FORUM   **create table and indexes in the same command**

i am currently creating the table then in a seperate command create the indexes

FORUM   **how to retrieve value from msdb.dbo.sp_help_job**

I need to retreive the current status of a job from msdb.dbo.sp_help_job

Tags

jobs
parallel processing

Write For Us
Find out more

```
create table #job_queue (id int identity(1,1) primary key

declare @create_job nvarchar(max),@db_name sysname,@job_n
declare @val int,@command nvarchar(max),@delay varchar(16

set @db_name=db_name()

set @i=1
while @i<=100
    begin
        insert #job_queue values('waitfor delay ''''0:0:'
        set @i=@i+1
    end
```

2. Starting the main loop. Create the SQL for the job first:

```
while exists(select 1 from #job_queue)
 begin
  select top 1 @val=id,@command=exe_proc from #job_queu
  set @job_name='_insert_base_job_name_'+cast(@val as v
  set @create_job='

  USE [msdb]

  DECLARE @jobId BINARY(16)
  EXEC  msdb.dbo.sp_add_job
      @job_name='''+@job_name+''',
      @enabled=1,
      @delete_level=3,
      @category_name=N''[Uncategorized (Local)]'',
      @owner_login_name=N''sa''

  EXEC msdb.dbo.sp_add_jobserver
      @job_name='''+@job_name+''',
```

3.Create the job and start it. Then remove the command from the queue:

```
exec sp_executesql @create_job

exec msdb.dbo.sp_start_job @job_name

delete top (1) #job_queue

print 'starting '+@job_name
```

4. Monitor number of jobs already running and wait until the number drops under 30 before starting a new one:

```
while (select count(*) from msdb.dbo.sysjobs_view job
        inner join msdb.dbo.sysjobactivity activity
        on (job.job_id = activity.job_id)
        where run_Requested_date is not null
        and stop_execution_date is null
        and job.name like  '%_Card360_peter_jobs_%')>=30
  waitfor delay '0:0:2'
 end


drop table #job_queue
```

Total article view s: 5399 | View s in the last 30 days: 5399