**Report Notes:**
This report includes the long running queries that are still in the procedure cache. Restarting the server or flushing the procedure cache will clear this list of long running queries and start with a clean list. To clear the procedure cache run DBCC FREEPROCCACHE; but use caution on production servers as this will flush all parsed queries from the procedure cache. All times shown are in milliseconds.

<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>815046</td>
<td>1</td>
<td>815046</td>
<td>815046</td>
<td>11/26/2011 4:40:46 PM</td>
</tr>
</tbody>
</table>

(CASE WHEN sp.type = N'P' THEN 1 WHEN sp.type = N'PC' THEN 2 ELSE 1 END AS [ImplementationType],
CAST(CASE WHEN ISNULL(smsp.definition, ssmsp.definition) IS NULL THEN 1 ELSE 0 END AS bit) AS [IsEncrypted]
FROM sys.all_objects AS sp
LEFT OUTER JOIN sys.sql_modules AS smsp ON smsp.object_id = sp.object_id
LEFT OUTER JOIN sys.system_sql_modules AS ssmsp ON ssmsp.object_id = sp.object_id
WHERE (sp.type = @_msparam_0 OR sp.type = @_msparam_1 OR sp.type=@_msparam_2)
ORDER BY [Schema] ASC,[Name] ASC


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>583033</td>
<td>1</td>
<td>583033</td>
<td>583033</td>
<td>11/26/2011 4:40:45 PM</td>
</tr>
</tbody>
</table>
```sql
SELECT SCHEMA_NAME(udf.schema_id) AS [Schema],
udf.name AS [Name],
udf.object_id AS [ID],
(case when 'FN' = udf.type then 1 when 'FS' = udf.type then 1 when 'IF' = udf.type then 3 when 'TF' = udf.type then 2 when 'FT' = udf.type then 2 else 0 end) AS [FunctionType],
CASE WHEN udf.type IN ('FN','IF','TF') THEN 1 WHEN udf.type IN ('FS','FT') THEN 2 ELSE 1 END AS [ImplementationType],
CAST(
    case when udf.is_ms_shipped = 1 then
    when (
        select major_id
        from sys.extended_properties
        where
            major_id = udf.object_id and
            minor_id = 0 and
            class = 1 and
            name = N'microsoft_database_tools_support'
        is not null then 1
        else 0
    end
) AS bit) AS [IsSystemObject],
CAST(CASE WHEN ISNULL(smudf.definition, ssmudf.definition) IS NULL THEN 1 ELSE 0 END AS bit) AS [IsEncrypted],
CAST(OBJECTPROPERTYEX(udf.object_id, N'IsSchemaBound') AS bit) AS [IsSchemaBound],
usrt.name AS [DataType],
ISNULL(baset.name, N'') AS [SystemType],
CAST(CASE WHEN baset.name IN (N'nchar', N'nvarchar') AND ret_param.max_length <> -1 THEN ret_param.max_length/2 ELSE ret_param.max_length END AS int) AS [Length],
CAST(ret_param.precision AS int) AS [NumericPrecision],
CAST(ret_param.scale AS int) AS [NumericScale],
ISNULL(xscret_param.name, N'') AS [XmlSchemaNamespace],
ISNULL(s2ret_param.name, N'') AS [XmlSchemaNamespaceSchema],
ISNULL( (case ret_param.is_xml_document when 1 then 2 else 1 end), 0) AS [XmlDocumentConstraint],
sret_param.name AS [DataTypeSchema]
FROM sys.all_objects AS udf
LEFT OUTER JOIN sys.sql_modules AS smudf ON smudf.object_id = udf.object_id
LEFT OUTER JOIN sys.system_sql_modules AS ssmudf ON ssmudf.object_id = udf.object_id
LEFT OUTER JOIN sys.all_parameters AS ret_param ON ret_param.object_id = udf.object_id and ret_param.is_output = 0
LEFT OUTER JOIN sys.system_sql_modules AS smudf ON smudf.object_id = udf.object_id
WHERE udf.type in ('TF', 'FN', 'IF', 'FS', 'FT')
ORDER BY [Name] ASC, [Schema] ASC
```


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>285016</td>
<td>1</td>
<td>285016</td>
<td>285016</td>
<td>11/26/2011 4:39:50 PM</td>
</tr>
</tbody>
</table>

SELECT top 40
rank() OVER (order by s.max_elapsed_time desc, s.execution_count desc, t.TEXT, s.creation_time) as Row,
s.execution_count * ISNULL(s.total_elapsed_time / s.execution_count, 0) as TotalRunTime,
t.TEXT TheQuery,
s.execution_count AS TimesRun,
s.max_elapsed_time AS LongestRunTime,
ISNULL(s.total_elapsed_time / s.execution_count, 0) AS AvgRunTime,
s.creation_time AS LogCreatedOn
FROM sys.dm_exec_query_stats s CROSS APPLY sys.dm_exec_sql_text( s.sql_handle ) t
WHERE s.max_elapsed_time > 0
ORDER BY LongestRunTime desc;


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>231013</td>
<td>1</td>
<td>231013</td>
<td>231013</td>
<td>11/26/2011 4:40:30 PM</td>
</tr>
</tbody>
</table>

create table #tmp_sp_get_sqlagent_properties
(auto_start int null, msx_server_name sysname null, sqlagent_type int null, startup_account nvarchar(255) null, sqlserver_restart int null,
drop table #tmp_sp_get_sqlagent_properties

declare @DatabaseMailProfile nvarchar(255)
exec master.dbo.xp_instance_regread N'HKEY_LOCAL_MACHINE', N'SOFTWARE\Microsoft\MSSQLServer\SQLServerAgent', N'DatabaseMailProfile', @param = @DatabaseMailProfile OUT, @no_output = N'no_output'

declare @AgentMailType int
exec master.dbo.xp_instance_regread N'HKEY_LOCAL_MACHINE', N'SOFTWARE\Microsoft\MSSQLServer\SQLServerAgent', N'UseDatabaseMail', @param = @AgentMailType OUT, @no_output = N'no_output'

declare @ServiceStartMode int
EXEC master.sys.xp_instance_regread 'HKEY_LOCAL_MACHINE', 'SYSTEM\CurrentControlSet\Services\SQLSERVERAGENT', N'Start', @ServiceStartMode OUTPUT

declare @ServiceAccount nvarchar(512)
EXEC master.sys.xp_instance_regread 'HKEY_LOCAL_MACHINE', 'SYSTEM\CurrentControlSet\Services\SQLSERVERAGENT', N'ObjectName', @ServiceAccount OUTPUT

declare @AgtGroup nvarchar(512)
exec master.dbo.xp_instance_regread N'HKEY_LOCAL_MACHINE', N'SOFTWARE\Microsoft\MSSQLServer\Setup', N'AGTGroup', @AgtGroup OUTPUT

SELECT
CAST(serverproperty(N'Servername') AS sysname) AS [Name],
ISNULL(tsgsp.msx_server_name,N'') AS [MsxServerName],
tsgsp.sqlagent_type AS [JobServerType],
CAST(tsgsp.sqlserver_restart AS bit) AS [SqlServerRestart],
CAST(tsgsp.monitor_autostart AS bit) AS [SqlAgentRestart],
tsgsp.jobhistory_max_rows AS [MaximumHistoryRows],
tsgsp.jobhistory_max_rows_per_job AS [MaximumJobHistoryRows],
tsgsp.errorlog_file AS [ErrorLogFile],
tsgsp.errorlogging_level AS [AgentLogLevel],
ISNULL(tsgsp.error_recipient,N'') AS [NetSendRecipient],
tsgsp.job_shutdown_timeout AS [AgentShutdownWaitTime],
ISNULL(tsgsp.email_profile,N'') AS [SqlAgentMailProfile],
CAST(tsgsp.email_save_in_sent_folder AS bit) AS [SaveInSentFolder],
CAST(tsgsp.oem_errorlog AS bit) AS [WriteOemErrorLog],
CAST(tsgsp.cpu_poller_enabled AS bit) AS [IsCpuPollingEnabled],
tsgsp.idle_cpu_percent AS [IdleCpuPercentage],
tsgsp.idle_cpu_duration AS [IdleCpuDuration],
tsgsp.login_timeout AS [LoginTimeout],
ISNULL(tsgsp.host_login_name,N'') AS [HostLoginName],
ISNULL(tsgsp.local_host_server,N'') AS [LocalHostAlias],
CAST(tsgsp.auto_start AS bit) AS [SqlAgentAutoStart],
CAST(tsgsp.replace_alert_tokens_enabled AS bit) AS [ReplaceAlertTokensEnabled],
ISNULL(tsgsp.@AgentMailType, 0) AS [AgentMailType],
CAST(1 AS bit) AS [SysAdminOnly],
@ServiceStartMode AS [ServiceStartMode],
ISNULL(@ServiceAccount,N'') AS [ServiceAccount],
ISNULL(suser_sname(sid_binary(ISNULL(@AgtGroup,N''))),N'') AS [AgentDomainGroup]
FROM #tmp_sp_get_sqlagent_properties AS tsgsp

drop table #tmp_sp_get_sqlagent_properties
set nocount on;
-- now fill in the two sample tables with the exact same names

declare @FName varchar(200);
declare @MName varchar(200);
declare @LName varchar(200);
declare @Age int;
declare @id uniqueidentifier;

select @Age = ABS(CHECKSUM(NEWID()))%110;  -- maximum age for this example is 110
select @id = NEWID();

SELECT top 1 @fName = fn.FirstName, @MName = m.FirstName, @lName = ln.LastName
FROM [FirstNames] fn
FULL JOIN [LastNames] ln ON fn.FirstName <> ln.LastName
FULL JOIN [FirstNames] m ON fn.FirstName <> m.FirstName
ORDER BY NEWID();

INSERT INTO [Good1]
  ([firstname], [middlename], [lastname], [age])
VALUES (@FName, @MName, @LName, @Age);

INSERT INTO [Bad1]
  ([id], [firstname], [middlename], [lastname], [age])
VALUES (@id, @FName, @MName, @LName, @Age);

INSERT INTO [Bad2]
  ([id], [firstname], [middlename], [lastname], [age])
VALUES (@id, @FName, @MName, @LName, @Age);
```sql
SELECT top 40
  rank() OVER (order by s.max_elapsed_time desc, s.execution_count desc) as Row,
  s.execution_count * ISNULL(s.total_elapsed_time / s.execution_count, 0) as TotalRunTime,
  t.TEXT TheQuery,
  s.execution_count AS TimesRun,
  s.max_elapsed_time AS LongestRunTime,
  ISNULL(s.total_elapsed_time / s.execution_count, 0) AS AvgRunTime,
  s.creation_time AS LogCreatedOn
FROM sys.dm_exec_query_stats s
  CROSS APPLY sys.dm_exec_sql_text( s.sql_handle ) t
WHERE s.max_elapsed_time > 0
ORDER BY LongestRunTime desc;
```


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>79004</td>
<td>1</td>
<td>79004</td>
<td>79004</td>
<td>11/26/2011 4:56:15 PM</td>
</tr>
</tbody>
</table>

```sql
SELECT param.is_readonly AS [IsReadOnly],
  param.name AS [Name],
  param.parameter_id AS [ID],
  param.default_value AS [DefaultValue],
  usrt.name AS [DataType],
  sparam.name AS [DataTypeSchema],
  ISNULL(baset.name, N'') AS [SystemType],
  CAST(CASE WHEN baset.name IN (N'nchar', N'nvarchar') AND param.max_length <> -1 THEN param.max_length/2 ELSE param.max_length_END AS int) AS [Length],
  CAST(param.precision AS int) AS [NumericPrecision],
  CAST(param.scale AS int) AS [NumericScale],
  ISNULL(xscparam.name, N'') AS [XmlSchemaNamespace],
  ISNULL(s2param.name, N'') AS [XmlSchemaNamespaceSchema],
  ISNULL((case param.is_xml_document when 1 then 2 else 1 end), 0) AS [XmlDocumentConstraint],
  CASE WHEN usrt.is_table_type = 1 THEN N'structured' ELSE N' END AS [UserType],
  udf.object_id AS [IDText],
  db_name() AS [DatabaseName],
  param.name AS [ParamName],
  CAST(
    when udf.is_ms_shipped = 1 then 1
    when (select
      major_id
    from sys.extended_properties
      where
        major_id = udf.object_id and
        minor_id = 0 and
        class = 1 and
        name = N'microsoft_database_tools_support')
      is not null then 1
    else 0
  end AS bit) AS [ParentSysObj],
  -1 AS [Number]
FROM sys.all_objects AS udf
  INNER JOIN sys.all_parameters AS param ON (param.is_output = 0) AND (param.object_id=udf.object_id)
  LEFT OUTER JOIN sys.types AS usrt ON usrt.user_type_id = param.user_type_id
  LEFT OUTER JOIN sys.schemas AS sparam ON sparam.schema_id = usrt.schema_id
  LEFT OUTER JOIN sys.types AS baset ON (baset.user_type_id = param.system_type_id and baset.object_id = baset.object_id) OR ((baset.system_type_id = param.system_type_id) and (baset.object_id = param.object_id) and (baset.is_user_defined = 0) and (baset.is_assembly_type = 1))
  LEFT OUTER JOIN sys.xml_schema_collections AS xscparam ON xscparam.xml_collection_id = param.xml_collection_id
  LEFT OUTER JOIN sys.schemas AS s2param ON s2param.schema_id = xscparam.schema_id
WHERE
  (param.name=@_msparam_0 and((udf.type in ('TF', 'FN', 'IF', 'FS', 'FT') and (udf.name=@_msparam_1 and SCHEMA_NAME(udf.schema_id)=@_msparam_2))


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>223012</td>
<td>7</td>
<td>50002</td>
<td>31858</td>
<td>11/26/2011 4:40:02 PM</td>
</tr>
</tbody>
</table>
SELECT top 40
  rank() OVER (order by s.max_elapsed_time desc, s.execution_count desc, t.TEXT, s.creation_time) as Row,
  s.execution_count * ISNULL(s.total_elapsed_time / s.execution_count, 0) as TotalRunTime,
  t.TEXT TheQuery,
  s.execution_count AS TimesRun,
  s.max_elapsed_time AS LongestRunTime,
  ISNULL(s.total_elapsed_time / s.execution_count, 0) AS AvgRunTime,
  s.creation_time AS LogCreatedOn
FROM sys.dm_exec_query_stats s CROSS APPLY sys.dm_exec_sql_text( s.sql_handle ) t
WHERE s.max_elapsed_time > 0
ORDER BY LongestRunTime desc;


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>47002</td>
<td>2</td>
<td>47002</td>
<td>23501</td>
<td>11/26/2011 4:40:28 PM</td>
</tr>
</tbody>
</table>

(@_msparam_0 nvarchar(4000))SELECT
  'PolicyStore[@Name=' + quotename(CAST(serverproperty(N'Servername') AS sysname), '''') + '] AS [Urn],
  CAST(serverproperty(N'Servername') AS sysname) AS [Name],
  CAST(
    (SELECT current_value FROM msdb.dbo.syspolicy_configuration WHERE name = 'Enabled')
    AS bit) AS [Enabled],
  CAST(
    (SELECT current_value FROM msdb.dbo.syspolicy_configuration WHERE name = 'HistoryRetentionInDays')
    AS int) AS [HistoryRetentionInDays],
  CAST(
    (SELECT current_value FROM msdb.dbo.syspolicy_configuration WHERE name = 'LogOnSuccess')
    AS bit) AS [LogOnSuccess]
WHERE
  (CAST(serverproperty(N'Servername') AS sysname)=@_msparam_0)


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>42002</td>
<td>1</td>
<td>42002</td>
<td>42002</td>
<td>11/26/2011 4:40:45 PM</td>
</tr>
</tbody>
</table>

(@_msparam_0 nvarchar(4000))SELECT
  SCHEMA_NAME(v.schema_id) AS [Schema],
  v.name AS [Name]
FROM
  sys.all_views AS v
WHERE
  (v.type = @_msparam_0)
ORDER BY
  [Schema] ASC,[Name] ASC


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>40002</td>
<td>1</td>
<td>40002</td>
<td>40002</td>
<td>11/26/2011 4:37:12 PM</td>
</tr>
</tbody>
</table>

SELECT top 40
  rank() OVER (order by s.max_elapsed_time desc, s.execution_count desc) as Row,
  s.execution_count * ISNULL(s.total_elapsed_time / s.execution_count, 0) as TotalRunTime,
  t.TEXT TheQuery,
  s.execution_count AS TimesRun,
  s.max_elapsed_time AS LongestRunTime,
  ISNULL(s.total_elapsed_time / s.execution_count, 0) AS AvgRunTime,
  s.creation_time AS LogCreatedOn
FROM sys.dm_exec_query_stats s CROSS APPLY sys.dm_exec_sql_text( s.sql_handle ) t
WHERE s.max_elapsed_time > 0
ORDER BY LongestRunTime desc;


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>39002</td>
<td>1</td>
<td>39002</td>
<td>39002</td>
<td>11/26/2011 4:49:05 PM</td>
</tr>
</tbody>
</table>
select top 10 *
from ( 
SELECT top 40
    rank() OVER (order by s.max_elapsed_time desc, s.execution_count desc, t.TEXT, s.creation_time) as Row,
    s.total_elapsed_time as TotalRunTime,
    t.TEXT TheQuery,
    s.execution_count AS TimesRun,
    s.max_elapsed_time AS LongestRunTime,
    ISNULL(s.total_elapsed_time / s.execution_count, 0) AS AvgRunTime,
    s.creation_time AS LogCreatedOn
FROM sys.dm_exec_query_stats s CROSS APPLY sys.dm_exec_sql_text( s.sql_handle ) t
WHERE s.max_elapsed_time > 0
ORDER BY LongestRunTime desc) as t order by t.TotalRunTime desc;


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>37002</td>
<td>1</td>
<td>37002</td>
<td>37002</td>
<td>11/26/2011 4:47:34 PM</td>
</tr>
</tbody>
</table>

SELECT top 10
    rank() OVER (order by s.total_elapsed_time desc, t.TEXT, s.creation_time) as Row,
    s.execution_count * ISNULL(s.total_elapsed_time / s.execution_count, 0) as TotalRunTime,
    t.TEXT TheQuery,
    s.execution_count as TimesRun,
    s.max_elapsed_time as LongestRunTime,
    ISNULL(s.total_elapsed_time / s.execution_count, 0) AS AvgRunTime,
    s.creation_time AS LogCreatedOn
FROM sys.dm_exec_query_stats s CROSS APPLY sys.dm_exec_sql_text( s.sql_handle ) t
WHERE s.max_elapsed_time > 0
ORDER BY TotalRunTime desc;


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>37002</td>
<td>1</td>
<td>37002</td>
<td>37002</td>
<td>11/26/2011 4:24:23 PM</td>
</tr>
</tbody>
</table>

SELECT top 40
    s.execution_count * ISNULL(s.total_elapsed_time / s.execution_count, 0) as TotalRunTime,
    t.TEXT TheQuery,
    s.execution_count AS TimesRun,
    s.max_elapsed_time AS LongestRunTime,
    ISNULL(s.total_elapsed_time / s.execution_count, 0) AS AvgRunTime,
    s.creation_time AS LogCreatedOn
FROM sys.dm_exec_query_stats s CROSS APPLY sys.dm_exec_sql_text( s.sql_handle ) t
WHERE s.max_elapsed_time > 0
ORDER BY TotalRunTime desc;


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>58003</td>
<td>2</td>
<td>35002</td>
<td>29001</td>
<td>11/26/2011 4:49:30 PM</td>
</tr>
</tbody>
</table>

SELECT top 10 *
from ( 
SELECT top 40
    rank() OVER (order by s.max_elapsed_time desc, s.execution_count desc, t.TEXT, s.creation_time) as Row,
    s.total_elapsed_time as TotalRunTime,
    t.TEXT TheQuery,
    s.execution_count AS TimesRun,
    s.max_elapsed_time AS LongestRunTime,
    ISNULL(s.total_elapsed_time / s.execution_count, 0) AS AvgRunTime,
    s.creation_time AS LogCreatedOn
FROM sys.dm_exec_query_stats s CROSS APPLY sys.dm_exec_sql_text( s.sql_handle ) t
WHERE s.max_elapsed_time > 0
ORDER BY LongestRunTime desc) as t order by t.TotalRunTime desc;


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>31001</td>
<td>1</td>
<td>31001</td>
<td>31001</td>
<td>11/26/2011 4:04:04 PM</td>
</tr>
</tbody>
</table>
CREATE PROCEDURE sp_get_sqlagent_properties
AS
BEGIN
DECLARE @auto_start INT
DECLARE @startup_account NVARCHAR(100)

SELECT clmns.column_id AS [ID],
clmns.name AS [Name],
clmns.is_nullable AS [Nullable],
CAST(ISNULL(cik.index_column_id, 0) AS bit) AS [InPrimaryKey],
clmns.is_identity AS [Identity],
usrt.name AS [DataType],
ISNULL(baset.name, N'') AS [SystemType],
CAST(CASE WHEN baset.name IN (N'nchar', N'nvarchar') AND clmns.max_length <> -1 THEN clmns.max_length/2 ELSE clmns.max_length END AS int) AS [Length],
CAST(clmns.precision AS int) AS [NumericPrecision],
CAST(clmns.scale AS int) AS [NumericScale],
ISNULL(xscclmns.name, N'') AS [XmlSchemaNamespace],
ISNULL(s2clmns.name, N'') AS [XmlSchemaNamespaceSchema],
ISNULL( (case clmns.is_xml_document when 1 then 2 else 1 end), 0) AS [XmlDocumentConstraint],
sclmns.name AS [DataTypeSchema]
FROM sys.all_views AS v
INNER JOIN sys.all_columns AS clmns ON clmns.object_id=v.object_id
LEFT OUTER JOIN sys.indexes AS ik ON ik.object_id = clmns.object_id and 1=ik.is_primary_key
LEFT OUTER JOIN sys.index_columns AS cik ON cik.index_id = ik.index_id and cik.column_id = clmns.column_id and cik.object_id = clmns.object_id and 0 = cik.is_included_column
LEFT OUTER JOIN sys.types AS usrt ON usrt.user_type_id = clmns.user_type_id
LEFT OUTER JOIN sys.types AS baset ON (baset.user_type_id = clmns.system_type_id and baset.user_type_id = baset.system_type_id) or ((baset.system_type_id = clmns.system_type_id) and (baset.user_type_id = clmns.user_type_id) and (baset.is_user_defined = 0) and (baset.is_assembly_type = 1))
LEFT OUTER JOIN sys.xml_schema_collections AS xscclmns ON xscclmns.object_id = clmns.object_id
LEFT OUTER JOIN sys.schemas AS s2clmns ON s2clmns.schema_id = clmns.schema_id
LEFT OUTER JOIN sys.schemas AS sclmns ON sclmns.schema_id = usrt.schema_id
WHERE (v.type = @_msparam_0 and v.name=@_msparam_1 and SCHEMA_NAME(v.schema_id)=@_msparam_2)
ORDER BY [ID] ASC

CREATE PROCEDURE sp_get_sqlagent_properties
AS
BEGIN
DECLARE @auto_start INT
DECLARE @startup_account NVARCHAR(100)

SELECT clmns.column_id AS [ID],
clmns.name AS [Name],
clmns.is_nullable AS [Nullable],
CAST(case when dtb.name in ('master','model','msdb','tempdb') then 1 else dtb.is_distributor end AS bit) AS [IsSystemObject],
CAST(has_dbaccess(dtb.name) AS bit) AS [IsAccessible],
dtb.collation_name AS [Collation],
dtb.name AS [DatabaseName]
FROM master.sys.databases AS dtb
WHERE (dtb.name=@_msparam_0)

SELECT @auto_start,
@startup_account
FROM sys.sql_modules
WHERE name = 'sp_get_sqlagent_properties'

SELECT top 40 rank() OVER (order by s.max_elapsed_time desc, s.execution_count desc),
s.execution_count * ISNULL(s.total_elapsed_time / s.execution_count, 0) as TotalRunTime,
t.TEXT TheQuery,
s.execution_count AS TimesRun,
s.max_elapsed_time AS LongestRunTime,
ISNULL(s.total_elapsed_time / s.execution_count, 0) AS AvgRunTime,
s.creation_time AS LogCreatedOn
FROM sys.dm_exec_query_stats s CROSS APPLY sys.dm_exec_sql_text( s.sql_handle ) t
WHERE s.max_elapsed_time > 0
ORDER BY LongestRunTime desc;

CREATE PROCEDURE sp_get_sqlagent_properties
AS
BEGIN
DECLARE @auto_start INT
DECLARE @startup_account NVARCHAR(100)

SELECT clmns.column_id AS [ID],
clmns.name AS [Name],
clmns.is_nullable AS [Nullable],
CAST(ISNULL(cik.index_column_id, 0) AS bit) AS [InPrimaryKey],
clmns.is_identity AS [Identity],
usrt.name AS [DataType],
ISNULL(baset.name, N'') AS [SystemType],
CAST(CASE WHEN baset.name IN (N'nchar', N'nvarchar') AND clmns.max_length <> -1 THEN clmns.max_length/2 ELSE clmns.max_length END AS int) AS [Length],
CAST(clmns.precision AS int) AS [NumericPrecision],
CAST(clmns.scale AS int) AS [NumericScale],
ISNULL(xscclmns.name, N'') AS [XmlSchemaNamespace],
ISNULL(s2clmns.name, N'') AS [XmlSchemaNamespaceSchema],
ISNULL( (case clmns.is_xml_document when 1 then 2 else 1 end), 0) AS [XmlDocumentConstraint],
sclmns.name AS [DataTypeSchema]
FROM sys.all_views AS v
INNER JOIN sys.all_columns AS clmns ON clmns.object_id=v.object_id
LEFT OUTER JOIN sys.indexes AS ik ON ik.object_id = clmns.object_id and 1=ik.is_primary_key
LEFT OUTER JOIN sys.index_columns AS cik ON cik.index_id = ik.index_id and cik.column_id = clmns.column_id and cik.object_id = clmns.object_id and 0 = cik.is_included_column
LEFT OUTER JOIN sys.types AS usrt ON usrt.user_type_id = clmns.user_type_id
LEFT OUTER JOIN sys.types AS baset ON (baset.user_type_id = clmns.system_type_id and baset.user_type_id = baset.system_type_id) or ((baset.system_type_id = clmns.system_type_id) and (baset.user_type_id = clmns.user_type_id) and (baset.is_user_defined = 0) and (baset.is_assembly_type = 1))
LEFT OUTER JOIN sys.xml_schema_collections AS xscclmns ON xscclmns.object_id = clmns.object_id
LEFT OUTER JOIN sys.schemas AS s2clmns ON s2clmns.schema_id = clmns.schema_id
LEFT OUTER JOIN sys.schemas AS sclmns ON sclmns.schema_id = usrt.schema_id
WHERE (v.type = @_msparam_0 and v.name=@_msparam_1 and SCHEMA_NAME(v.schema_id)=@_msparam_2)
ORDER BY [ID] ASC
DECLARE @msx_server_name sysname

-- Non-SQLDMO exposed properties
DECLARE @sqlserver_restart INT
DECLARE @jobhistory_max_rows INT
DECLARE @jobhistory_max_rows_per_job INT
DECLARE @errorlog_file NVARCHAR(255)
DECLARE @errorlogging_level INT
DECLARE @error_recipient NVARCHAR(30)
DECLARE @monitor_autostart INT
DECLARE @local_host_server sysname
DECLARE @job_shutdown_timeout INT
DECLARE @cmdexec_account VARBINARY(64)
DECLARE @regular_connections INT
DECLARE @host_login_name sysname
DECLARE @host_login_password VARBINARY(512)
DECLARE @login_timeout INT
DECLARE @idle_cpu_percent INT
DECLARE @idle_cpu_duration INT
DECLARE @oem_errorlog INT
DECLARE @(email_profile NVARCHAR(64)
DECLARE @email_save_in_sent_folder INT
DECLARE @cpu_poller_enabled INT
DECLARE @alert_replace_runtime_tokens INT

SET NOCOUNT ON

-- NOTE: We return all SQLServerAgent properties at one go for performance reasons

-- Read the values from the registry
IF ((PLATFORM() & 0x1) = 0x1) -- NT
BEGIN
DECLARE @key NVARCHAR(200)
SELECT @key = N'SYSTEM\CurrentControlSet\Services\'
IF (SERVERPROPERTY('INSTANCENAME') IS NULL)
SELECT @key = @key + N'SQLAgent$' + CONVERT (sysname, SERVERPROPERTY('INSTANCENAME'))
ELSE
SELECT @key = @key + N'SQLServerAgent'

EXECUTE master.dbo.xp_regread N'HKEY_LOCAL_MACHINE',
@key,
N'Start', @auto_start OUTPUT,
N'no_output'
EXECUTE master.dbo.xp_regread N'HKEY_LOCAL_MACHINE',
@key,
N'ObjectName', @startup_account OUTPUT,
N'no_output'
END
ELSE
BEGIN
SELECT @auto_start = 3 -- Manual start
SELECT @startup_account = NULL
END
EXECUTE master.dbo.xp_instance_regread N'HKEY_LOCAL_MACHINE',
N'SOFTWARE\Microsoft\MSSQLServer\SQLServerAgent',
N'MSXServerName',
@msx_server_name OUTPUT,
N'no_output'

-- Non-SQLDMO exposed properties
EXECUTE master.dbo.xp_instance_regread N'HKEY_LOCAL_MACHINE',
N'SOFTWARE\Microsoft\MSSQLServer\SQLServerAgent',
N'RestartSQLServer',
@sqlserver_restart OUTPUT,
N'no_output'
EXECUTE master.dbo.xp_instance_regread N'HKEY_LOCAL_MACHINE',
N'SOFTWARE\Microsoft\MSSQLServer\SQLServerAgent',
N'JobHistoryMaxRows',
@jobhistory_max_rows OUTPUT,
N'no_output'
EXECUTE master.dbo.xp_instance_regread N'HKEY_LOCAL_MACHINE',
N'SOFTWARE\Microsoft\MSSQLServer\SQLServerAgent',
N'JobHistoryMaxRowsPerJob',
@jobhistory_max_rows_per_job OUTPUT,
N'no_output'
EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
  N'SOFTWARE\Microsoft\SQLServer\SQLServerAgent',
  N'ErrorLogFile',
  @errorlog_file OUTPUT,
  N'no_output'

EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
  N'SOFTWARE\Microsoft\SQLServer\SQLServerAgent',
  N'ErrorLoggingLevel',
  @errorlogging_level OUTPUT,
  N'no_output'

EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
  N'SOFTWARE\Microsoft\SQLServer\SQLServerAgent',
  N'ErrorMonitor',
  @error_recipient OUTPUT,
  N'no_output'

EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
  N'SOFTWARE\Microsoft\SQLServer\SQLServerAgent',
  N'MonitorAutoStart',
  @monitor_autostart OUTPUT,
  N'no_output'

EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
  N'SOFTWARE\Microsoft\SQLServer\SQLServerAgent',
  N'ServerHost',
  @local_host_server OUTPUT,
  N'no_output'

EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
  N'SOFTWARE\Microsoft\SQLServer\SQLServerAgent',
  N'JobShutdownTimeout',
  @job_shutdown_timeout OUTPUT,
  N'no_output'

EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
  N'SOFTWARE\Microsoft\SQLServer\SQLServerAgent',
  N'CmdExecAccount',
  @cmdexec_account OUTPUT,
  N'no_output'

SET @regular_connections = 0
SET @host_login_name = NULL
SET @host_login_password = NULL

EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
  N'SOFTWARE\Microsoft\SQLServer\SQLServerAgent',
  N'LoginTimeout',
  @login_timeout OUTPUT,
  N'no_output'

EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
  N'SOFTWARE\Microsoft\SQLServer\SQLServerAgent',
  N'IdleCPUPercent',
  @idle_cpu_percent OUTPUT,
  N'no_output'

EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
  N'SOFTWARE\Microsoft\SQLServer\SQLServerAgent',
  N'IdleCPUDuration',
  @idle_cpu_duration OUTPUT,
  N'no_output'

EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
  N'SOFTWARE\Microsoft\SQLServer\SQLServerAgent',
  N'OemErrorLog',
  @oem_errorlog OUTPUT,
  N'no_output'

EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
  N'SOFTWARE\Microsoft\SQLServer\SQLServerAgent',
  N'EmailProfile',
  @email_profile OUTPUT,
  N'no_output'

EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
  N'SOFTWARE\Microsoft\SQLServer\SQLServerAgent',
  N'EmailSaveSent',
  @email_save_in_sent_folder OUTPUT,
  N'no_output'

EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
  N'SOFTWARE\Microsoft\SQLServer\SQLServerAgent',
  N'AlertReplaceRuntimeTokens',
  @alert_replace_runtime_tokens OUTPUT,
  N'no_output'

EXECUTE master.dbo.xp_instance_regread N'\HKEY_LOCAL_MACHINE',
IF (@cpu_poller_enabled IS NOT NULL)
SELECT @cpu_poller_enabled = CASE WHEN (@cpu_poller_enabled & 32) = 32 THEN 0 ELSE 1 END

-- Return the values to the client
SELECT auto_start = CASE @auto_start
WHEN 2 THEN 1 -- 2 means auto-start
WHEN 3 THEN 0 -- 3 means don't auto-start
ELSE 0
END,
msx_server_name = @msx_server_name,
sqlagent_type = (SELECT CASE
WHEN (COUNT(*) = 0) AND (ISNULL(DATALENGTH(@msx_server_name), 0) = 0) THEN 1 -- Standalone
WHEN (COUNT(*) = 0) AND (ISNULL(DATALENGTH(@msx_server_name), 0) > 0) THEN 2 -- TSX
WHEN (COUNT(*) > 0) AND (ISNULL(DATALENGTH(@msx_server_name), 0) = 0) THEN 3 -- MSX
WHEN (COUNT(*) > 0) AND (ISNULL(DATALENGTH(@msx_server_name), 0) > 0) THEN 0 -- Multi-Level MSX (currently invalid)
ELSE 0
END
FROM msdb.dbo.systargetservers),
startup_account = @startup_account,
-- Non-SQLDMO exposed properties
sqlserver_restart = ISNULL(@sqlserver_restart, 1),
jobhistory_max_rows = @jobhistory_max_rows,
jobhistory_max_rows_per_job = @jobhistory_max_rows_per_job,
errorlog_file = @errorlog_file,
errorlogging_level = ISNULL(@errorlogging_level, 7),
error_recipient = @error_recipient,
monitor_autostart = ISNULL(@monitor_autostart, 0),
local_host_server = @local_host_server,
job_shutdown_timeout = ISNULL(@job_shutdown_timeout, 15),
cmdexec_account = @cmdexec_account,
regular_connections = ISNULL(@regular_connections, 0),
host_login_name = @host_login_name,
host_login_password = @host_login_password,
login_timeout = ISNULL(@login_timeout, 30),
idle_cpu_percent = ISNULL(@idle_cpu_percent, 10),
idle_cpu_duration = ISNULL(@idle_cpu_duration, 600),
oem_errorlog = ISNULL(@oem_errorlog, 0),
sysadmin_only = NULL,
email_profile = @email_profile,
email_save_in_sent_folder = ISNULL(@email_save_in_sent_folder, 0),
cpu_poller_enabled = ISNULL(@cpu_poller_enabled, 0),
alert_replace_runtime_tokens = ISNULL(@alert_replace_runtime_tokens, 0)
END


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
<td>29001</td>
<td>1</td>
<td>29001</td>
<td>29001</td>
<td>11/26/2011 4:50:05 PM</td>
</tr>
</tbody>
</table>

SELECT top 40
rank() OVER (order by s.max_elapsed_time desc, s.execution_count desc, t.TEXT, s.creation_time) as Row,
s.total_elapsed_time as TotalRunTime,
t.TEXT TheQuery,
s.execution_count AS TimesRun,
s.max_elapsed_time AS LongestRunTime,
ISNULL(s.total_elapsed_time / s.execution_count, 0) AS AvgRunTime,
s.creation_time AS LogCreatedOn
FROM sys.dm_exec_query_stats s CROSS APPLY sys.dm_exec_sql_text( s.sql_handle ) t
WHERE s.max_elapsed_time > 0
ORDER BY LongestRunTime desc;


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>26001</td>
<td>4</td>
<td>26001</td>
<td>6500</td>
<td>11/26/2011 4:40:30 PM</td>
</tr>
</tbody>
</table>
CREATE PROCEDURE [dbo].[GetUpgradeItems]
AS
SELECT [Item], [Status]
FROM [UpgradeInfo]

SELECT top 20
rank() OVER (order by s.max_elapsed_time desc, s.execution_count desc, t.TEXT, s.creation_time) as Row,
s.execution_count * ISNULL(s.total_elapsed_time / s.execution_count, 0) as TotalRunTime,
l.TEXT TheQuery,
s.execution_count AS TimesRun,
s.max_elapsed_time AS LongestRunTime,
ISNULL(s.total_elapsed_time / s.execution_count, 0) AS AvgRunTime,
s.creation_time AS LogCreatedOn
FROM sys.dm_exec_query_stats s CROSS APPLY sys.dm_exec_sql_text( s.sql_handle ) t
WHERE s.max_elapsed_time > 0
ORDER BY LongestRunTime desc;

CREATE PROCEDURE [dbo].[GetUpgradeItems]
AS
SELECT [Item], [Status]
FROM [UpgradeInfo]
-- Cleaning orphan policies
CREATE PROCEDURE [dbo].[CleanOrphanedPolicies]
AS
SET NOCOUNT OFF
DELETE
[Policies]
WHERE
[Policies].[PolicyFlag] = 0
AND
NOT EXISTS (SELECT ItemID FROM [Catalog] WHERE [Catalog].[PolicyID] = [Policies].[PolicyID])
DELETE
[Policies]
FROM
[Policies]
INNER JOIN [ModelItemPolicy] ON [ModelItemPolicy].[PolicyID] = [Policies].[PolicyID]
WHERE
NOT EXISTS (SELECT ItemID FROM [Catalog] WHERE [Catalog].[ItemID] = [ModelItemPolicy].[CatalogItemID])


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>27</td>
<td>12000</td>
<td>1</td>
<td>12000</td>
<td>12000</td>
<td>11/26/2011 4:29:56 PM</td>
</tr>
<tr>
<td>28</td>
<td>10000</td>
<td>1</td>
<td>10000</td>
<td>10000</td>
<td>11/26/2011 4:40:45 PM</td>
</tr>
<tr>
<td>29</td>
<td>8000</td>
<td>1</td>
<td>8000</td>
<td>8000</td>
<td>11/26/2011 4:40:29 PM</td>
</tr>
</tbody>
</table>

CREATE TABLE #tmp_sp_get_sqlagent_properties
(auto_start INT NULL, msx_server_name SYSNAME NULL, sqlagent_type INT NULL, startup_account NVARCHAR(255) NULL, sqlserver_restart INT NULL, jobhistory_max_rows INT NULL, jobhistory_max_rows_per_job INT NULL, errorlog_file NVARCHAR(255) NULL, errorlogging_level INT NULL, error_recipient NVARCHAR(30) NULL, monitor_autostart INT NULL, local_host_server SYSNAME NULL, job_shutdown_timeout INT NULL, cmdexec_account VARBINARY(64) NULL, regular_connections INT NULL, host_login_name SYSNAME NULL, host_login_password VARBINARY(512) NULL, login_timeout INT NULL, idle_cpu_percent INT NULL, idle_cpu_duration INT NULL, oem_errorlog INT NULL, sysadmin_only INT NULL, email_profile NVARCHAR(64) NULL, email_save_in_sent_folder INT NULL, cpu_poller_enabled INT NULL, replace_alert_tokens_enabled INT NULL)

INSERT INTO #tmp_sp_get_sqlagent_properties(auto_start, msx_server_name, sqlagent_type, startup_account, sqlserver_restart, jobhistory_max_rows,
SELECT
CAST(serverproperty(N'Servername') AS sysname) AS [Name],
ISNULL(tsgsp.sql_server_name,N') AS [MsxServerName],
tsgsp.sqlagent_type AS [JobServerType],
CAST(tsgsp.sqlserver_restart AS bit) AS [SqlServerRestart],
CAST(tsgsp.monitor_autostart AS bit) AS [SqlAgentRestart],
tsgsp.jobhistory_max_rows AS [MaximumHistoryRows],
tsgsp.jobhistory_max_rows_per_job AS [MaximumJobHistoryRows],
tsgsp.errorlog_file AS [ErrorLogFile],
tsgsp.errorlogging_level AS [AgentLogLevel],
ISNULL(tsgsp.error_recipient,N') AS [NetSendRecipient],
tsgsp.job_shutdown_timeout AS [AgentShutdownWaitTime],
ISNULL(tsgsp.email_profile,N') AS [SqlAgentMailProfile],
CAST(tsgsp.email_save_in_sent_folder AS bit) AS [SaveInSentFolder],
CAST(tsgsp.oem_errorlog AS bit) AS [WriteOemErrorLog],
CAST(tsgsp.cpu_poller_enabled AS bit) AS [IsCpuPollingEnabled],
tsgsp.idle_cpu_percent AS [IdleCpuPercentage],
tsgsp.idle_cpu_duration AS [IdleCpuDuration],
tsgsp.login_timeout AS [LoginTimeout],
ISNULL(tsgsp.host_login_name,N') AS [LoginUserName],
@serviceStartMode AS [ServiceStartMode],
CAST(@ServiceAccount AS [ServiceAccount]),
ISNULL(@AGENTGroup,N') AS [AgentDomainGroup]
FROM
#tmp_sp_get_sqlagent_properties AS tsgsp

drop table #tmp_sp_get_sqlagent_properties

declare @DatabaseMailProfile nvarchar(255)
exec master.dbo.xp_instance_regread N'HKEY_LOCAL_MACHINE', N'SOFTWARE\Microsoft\MSSQLServer\SQLServerAgent', N'DatabaseMailProfile', @param = @DatabaseMailProfile OUT, @no_output = N'no_output'

declare @AgentMailType int
exec master.dbo.xp_instance_regread N'HKEY_LOCAL_MACHINE', N'SOFTWARE\Microsoft\MSSQLServer\SQLServerAgent', N'UseDatabaseMail', @param = @AgentMailType OUT, @no_output = N'no_output'

declare @ServiceStartMode int
EXEC master.sys.xp_instance_regread 'HKEY_LOCAL_MACHINE', 'SYSTEM\CurrentControlSet\Services\SQLSERVERAGENT', N'Start', @ServiceStartMode OUTPUT

declare @ServiceAccount nvarchar(512)
EXEC master.sys.xp_instance_regread 'HKEY_LOCAL_MACHINE', 'SYSTEM\CurrentControlSet\Services\SQLSERVERAGENT', N'ObjectName', @ServiceAccount OUTPUT

declare @AgtGroup nvarchar(512)
exec master.dbo.xp_instance_regread N'HKEY_LOCAL_MACHINE', N'SOFTWARE\Microsoft\MSSQLServer\Setup', N'AGTGroup', @AgtGroup OUTPUT

SELECT
Row, Total Run Time, Times Run, Longest Run Time, Avg Run Time, Creation Date
FROM

<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>3000</td>
<td>1</td>
<td>3000</td>
<td>3000</td>
<td>11/26/2011 12:06:08 PM</td>
</tr>
</tbody>
</table>
CREATE PROCEDURE [dbo].[ListUsedDeliveryProviders]
AS
select distinct [DeliveryExtension] from Subscriptions where [DeliveryExtension] <> ''


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>3000</td>
<td>1</td>
<td>3000</td>
<td>3000</td>
<td>11/26/2011 4:40:46 PM</td>
</tr>
</tbody>
</table>

SELECT
SCHEMA_NAME(obj.schema_id) AS [Schema],
obj.name AS [Name],
obj.object_id AS [ID],
usrt.name AS [DataType],
ISNULL(baset.name, N'') AS [SystemType],
CAST(CASE WHEN baset.name IN (N'Nchar', N'nvarchar') AND ret_param.max_length <> -1 THEN ret_param.max_length/2 ELSE ret_param.max_length END AS int) AS [Length],
CAST(ret_param.precision AS int) AS [NumericPrecision],
CAST(ret_param.scale AS int) AS [NumericScale],
ISNULL(xscret_param.name, N'') AS [XmlSchemaNamespace],
ISNULL(s2ret_param.name, N'') AS [XmlSchemaNamespaceSchema],
ISNULL((case ret_param.is_xml_document when 1 then 2 else 1 end), 0) AS [XmlDocumentConstraint],
sret_param.name AS [DataTypeSchema]
FROM
sys.objects AS obj
LEFT OUTER JOIN sys.all_parameters AS ret_param ON ret_param.object_id = obj.object_id and ret_param.is_output = 1
LEFT OUTER JOIN sys.types AS usrt ON usrt.user_type_id = ret_param.user_type_id
LEFT OUTER JOIN sys.types AS baset ON (baset.user_type_id = ret_param.system_type_id and baset.user_type_id = baset.system_type_id) or ((baset.system_type_id = ret_param.system_type_id) and (baset.user_type_id = ret_param.user_type_id) and (baset.is_user_defined = 0) and (baset.is_assembly_type = 1))
LEFT OUTER JOIN sys.xml_schema_collections AS xscret_param ON xscret_param.xml_collection_id = ret_param.xml_collection_id
LEFT OUTER JOIN sys.schemas AS s2ret_param ON s2ret_param.schema_id = xscret_param.schema_id
LEFT OUTER JOIN sys.schemas AS sret_param ON sret_param.schema_id = usrt.schema_id
WHERE
(obj.type=N'AF')
ORDER BY
[Schema] ASC,[Name] ASC


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>33</td>
<td>315018</td>
<td>1762</td>
<td>1000</td>
<td>178</td>
<td>11/26/2011 12:06:08 PM</td>
</tr>
</tbody>
</table>

declare @BatchID uniqueidentifier
set @BatchID = NEWID()

UPDATE [Event] WITH (TABLOCKX)
SET [BatchID] = @BatchID,
[ProcessStart] = GETUTCDATE(),
[ProcessHeartbeat] = GETUTCDATE()
FROM (SELECT TOP 4 [EventID] FROM [Event] WITH (TABLOCKX) WHERE [ProcessStart] is NULL ORDER BY [TimeEntered]) AS t1
WHERE [Event].[EventID] = t1.[EventID]

select top 4
E.[EventID],
E.[EventType],
E.[EventData]
from
[Event] E WITH (TABLOCKX)
WHERE [BatchID] = @BatchID
ORDER BY [TimeEntered]


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>34</td>
<td>192010</td>
<td>1762</td>
<td>1000</td>
<td>108</td>
<td>11/26/2011 12:06:08 PM</td>
</tr>
</tbody>
</table>
declare @BatchID uniqueidentifier

set @BatchID = NEWID()

UPDATE [Event] WITH (TABLOCKX)
    SET [BatchID] = @BatchID,
        [ProcessStart] = GETUTCDATE(),
        [ProcessHeartbeat] = GETUTCDATE()
FROM (  
    ) AS t1
WHERE [Event].[EventID] = t1.[EventID]

select top 4 
    E.[EventID],
    E.[EventType],
    E.[EventData]
from 
    [Event] E WITH (TABLOCKX)
where 
    [BatchID] = @BatchID
ORDER BY [TimeEntered]

<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>35</td>
<td>7003</td>
<td>1762</td>
<td>1000</td>
<td>39</td>
<td>11/26/2011 12:06:08 PM</td>
</tr>
</tbody>
</table>

declare @BatchID uniqueidentifier

set @BatchID = newid()

UPDATE [Notifications] WITH (TABLOCKX)
SET [BatchID] = @BatchID,
    [ProcessStart] = GETUTCDATE(),
    [ProcessHeartbeat] = GETUTCDATE()
FROM (SELECT TOP 4 [NotificationID] FROM [Notifications] WITH (TABLOCKX) WHERE ProcessStart is NULL and (ProcessAfter is NULL or ProcessAfter < GETUTCDATE()) ORDER BY [NotificationEntered]) AS t1
WHERE [Notifications].[NotificationID] = t1.[NotificationID]

select top 4 -- Notification data
N.[NotificationID],
N.[SubscriptionID],
N.[ActivationID],
N.[ReportID],
N.[SnapShotDate],
N.[DeliveryExtension],
N.[ExtensionSettings],
N.[Locale],
N.[Parameters],
N.[SubscriptionLastRunTime],
N.[ProcessStart],
N.[NotificationEntered],
N.[Attempt],
N.[IsDataDriven],
SUSER_SNAME(Owner.[Sid]),
Owner.[UserName],
-- Report Data
O.[Path],
O.[Type],
SD.NtSecDescPrimary,
N.[Version],
Owner.[AuthType]
inner join [Users] Owner on N.SubscriptionOwnerID = Owner.UserID
left outer join [SecData] SD on O.[PolicyID] = SD.[PolicyID] AND SD.AuthType = Owner.AuthType
where N.[BatchID] = @BatchID
ORDER BY [NotificationEntered]


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>36</td>
<td>3000</td>
<td>1762</td>
<td>1000</td>
<td>1</td>
<td>11/26/2011 12:06:08 PM</td>
</tr>
</tbody>
</table>
declare @BatchID uniqueidentifier

set @BatchID = newid()

UPDATE [Notifications] WITH (TABLOCKX)
    SET [BatchID] = @BatchID,
            [ProcessStart] = GETUTCDATE(),
            [ProcessHeartbeat] = GETUTCDATE()
FROM (SELECT TOP 4 [NotificationID] FROM [Notifications] WITH (TABLOCKX) WHERE ProcessStart is NULL and (ProcessAfter is NULL or ProcessAfter < GETUTCDATE()) ORDER BY [NotificationEntered]) AS t1
WHERE [Notifications].[NotificationID] = t1.[NotificationID]

select top 4
    -- Notification data
    N.[NotificationID],
    N.[SubscriptionID],
    N.[ActivationID],
    N.[ReportID],
    N.[SnapShotDate],
    N.[DeliveryExtension],
    N.[ExtensionSettings],
    N.[Locale],
    N.[Parameters],
    N.[SubscriptionLastRunTime],
    N.[ProcessStart],
    N.[NotificationEntered],
    N.[Attempt],
    N.[IsDataDriven],
    SUSER_SNAME(Owner.[Sid]),
    Owner.[UserName],
    -- Report Data
    O.[Path],
    O.[Type],
    SD.NtSecDescPrimary,
    N.[Version],
    Owner.[AuthType]
inner join [Users] Owner on N.SubscriptionOwnerID = Owner.UserID
left outer join [SecData] SD on O.[PolicyID] = SD.[PolicyID] AND SD.AuthType = Owner.AuthType
where N.[BatchID] = @BatchID
ORDER BY [NotificationCreated]

CREATE PROCEDURE [dbo].[GetMyRunningJobs]
@ComputerName as nvarchar(32),
@JobType as smallint
AS
    SELECT JobID, StartDate, ComputerName, RequestName, RequestPath, SUSER_SNAME(Users.[Sid]), Users.[UserName], Description,
            Timeout, JobAction, JobType, JobStatus, Users.[AuthType]
    FROM RunningJobs INNER JOIN Users
        ON RunningJobs.UserID = Users.UserID
    WHERE ComputerName = @ComputerName
    AND JobType = @JobType
ORDER BY CreationDate


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>37</td>
<td>45002</td>
<td>293</td>
<td>1000</td>
<td>153</td>
<td>11/26/2011 12:07:08 PM</td>
</tr>
</tbody>
</table>

CREATE PROCEDURE [dbo].[GetMyRunningJobs]
@ComputerName as nvarchar(32),
@JobType as smallint
AS
    SELECT JobID, StartDate, ComputerName, RequestName, RequestPath, SUSER_SNAME(Users.[Sid]), Users.[UserName], Description,
            Timeout, JobAction, JobType, JobStatus, Users.[AuthType]
    FROM RunningJobs INNER JOIN Users
        ON RunningJobs.UserID = Users.UserID
    WHERE ComputerName = @ComputerName
    AND JobType = @JobType
ORDER BY CreationDate


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td>6000</td>
<td>83</td>
<td>1000</td>
<td>72</td>
<td>11/26/2011 4:59:36 PM</td>
</tr>
</tbody>
</table>
-- I use NEWID() a lot to generate some fairly random numbers for test data.

set nocount on;
-- now fill in the two sample tables with the exact same names

declare @FName varchar(200);
declare @MName varchar(200);
declare @LName varchar(200);
declare @Age int;
declare @id uniqueidentifier;

select @Age = ABS(CHECKSUM(NEWID()))%110;  -- maximum age for this example is 110
select @id = NEWID();

SELECT top 1 @fName = fn.FirstName, @MName = m.FirstName, @lName = ln.LastName
FROM [FirstNames] fn
FULL JOIN [LastNames] ln ON fn.FirstName <> ln.LastName
FULL JOIN [FirstNames] m ON fn.FirstName <> m.FirstName
ORDER BY NEWID();

INSERT INTO [Good1]
    ([firstname], [middlename], [lastname], [age])
VALUES (@FName, @MName, @LName, @Age);

INSERT INTO [Bad1]
    ([id], [firstname], [middlename], [lastname], [age])
VALUES (@id, @FName, @MName, @LName, @Age);

INSERT INTO [Bad2]
    ([id], [firstname], [middlename], [lastname], [age])
VALUES (@id, @FName, @MName, @LName, @Age);


<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td>7000</td>
<td>83</td>
<td>1000</td>
<td>84</td>
<td>11/26/2011 4:59:36 PM</td>
</tr>
</tbody>
</table>

-- I use NEWID() a lot to generate some fairly random numbers for test data.

set nocount on;
-- now fill in the two sample tables with the exact same names

declare @FName varchar(200);
declare @MName varchar(200);
declare @LName varchar(200);
declare @Age int;
declare @id uniqueidentifier;

select @Age = ABS(CHECKSUM(NEWID()))%110;  -- maximum age for this example is 110
select @id = NEWID();

SELECT top 1 @fName = fn.FirstName, @MName = m.FirstName, @lName = ln.LastName
FROM [FirstNames] fn
FULL JOIN [LastNames] ln ON fn.FirstName <> ln.LastName
FULL JOIN [FirstNames] m ON fn.FirstName <> m.FirstName
ORDER BY NEWID();

INSERT INTO [Good1]
    ([firstname], [middlename], [lastname], [age])
VALUES (@FName, @MName, @LName, @Age);

INSERT INTO [Bad1]
    ([id], [firstname], [middlename], [lastname], [age])
VALUES (@id, @FName, @MName, @LName, @Age);

INSERT INTO [Bad2]
    ([id], [firstname], [middlename], [lastname], [age])
VALUES (@id, @FName, @MName, @LName, @Age);
<table>
<thead>
<tr>
<th>Row</th>
<th>Total Run Time</th>
<th>Times Run</th>
<th>Longest Run Time</th>
<th>Avg Run Time</th>
<th>Creation Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>38</td>
<td>6000</td>
<td>83</td>
<td>1000</td>
<td>72</td>
<td>11/26/2011 4:59:36 PM</td>
</tr>
</tbody>
</table>

-- I use NEWID() a lot to generate some fairly random numbers for test data.

set nocount on;
-- now fill in the two sample tables with the exact same names

declare @FName varchar(200);
declare @MName varchar(200);
declare @LName varchar(200);
declare @Age int;
declare @id uniqueidentifier;

select @Age = ABS(CHECKSUM(NEWID()))%110; -- maximum age for this example is 110
select @id = NEWID();

SELECT top 1 @fName = fn.FirstName, @MName = m.FirstName, @LName = ln.LastName
FROM [FirstNames] fn
FULL JOIN [LastNames] ln ON fn.FirstName <> ln.LastName
FULL JOIN [FirstNames] m ON fn.FirstName <> m.FirstName
ORDER BY NEWID();

INSERT INTO [Good1]
    ([firstname], [middlename], [lastname], [age])
VALUES (@FName, @MName, @LName, @Age);

INSERT INTO [Bad1]
    ([id], [firstname], [middlename], [lastname], [age])
VALUES (@id, @FName, @MName, @LName, @Age);

INSERT INTO [Bad2]
    ([id], [firstname], [middlename], [lastname], [age])
VALUES (@id, @FName, @MName, @LName, @Age);