ADMINISTRATION GUIDE | PUBLIC

SAP Adaptive Server Enterprise 16.0 SP03

Document Version: 1.0 – 2020-03-04

Job Scheduler Users Guide

THE BEST RUN SAP

Content

1 Introduction to Job Scheduler

Use Job Scheduler to define and schedule database administration and maintenance tasks. Jobs that normally require interaction from a database administrator can be scheduled to run unattended at the appropriate times, freeing the database administrator to attend to other issues.

Job Scheduler allows you to create and schedule jobs, and to share jobs and schedules. One database administrator can create a job, and other database administrators can then schedule and run that job on another server.

You can create jobs from:

- Scratch, using the command line
- SAP Adaptive Server Enterprise Cockpit
- SQL batch file
- A template

Job Scheduler captures the results and output of jobs and records that information in log tables. You can view this data at any time. In addition, Job Scheduler keeps a history of scheduled jobs; however, to keep a limit on the size of the history table, Job Scheduler outdated and unnecessary history records are automatically removed based on criteria you specify.

> i Note
>
> To use Job Scheduler effectively, you should understand database management and administration. Job Scheduler executes operations that are usually performed by a database administrator.

1.1 Terminology and Concepts

Basic concepts and terminology associated with using Job Scheduler.

- Job – one or more actions performed on a database in a single operation, such as backing up, updating statistics, and dumping a database.
- Schedule – a timescale for how and when jobs are executed and reexecuted.
- Scheduled job – a job that has been bound to a schedule. Only scheduled jobs are executed.
- Job Scheduler Task (JS Task) – the functional component that manages the schedules and provides timely notification to the Job Scheduler Agent to execute a particular job.
- Job Scheduler Agent (JS Agent) – the functional component that executes a job when notified by the JS Task.
- Repeating schedule – a schedule that is active more than once. All repeating schedules must include both start and end times.
- Target server – server on which a job is scheduled to run.
- Template – a set of Transact-SQL statements with parameters that can be used to create a job within the Job Scheduler.

- sybmgmtdb – Job Scheduler database

1.2 Job Scheduler Architecture

The JS Task determines when scheduled jobs run, and creates a historical record of completed jobs. It starts the JS Agent process and feeds it the necessary information to retrieve job information and run the job on the specified server.

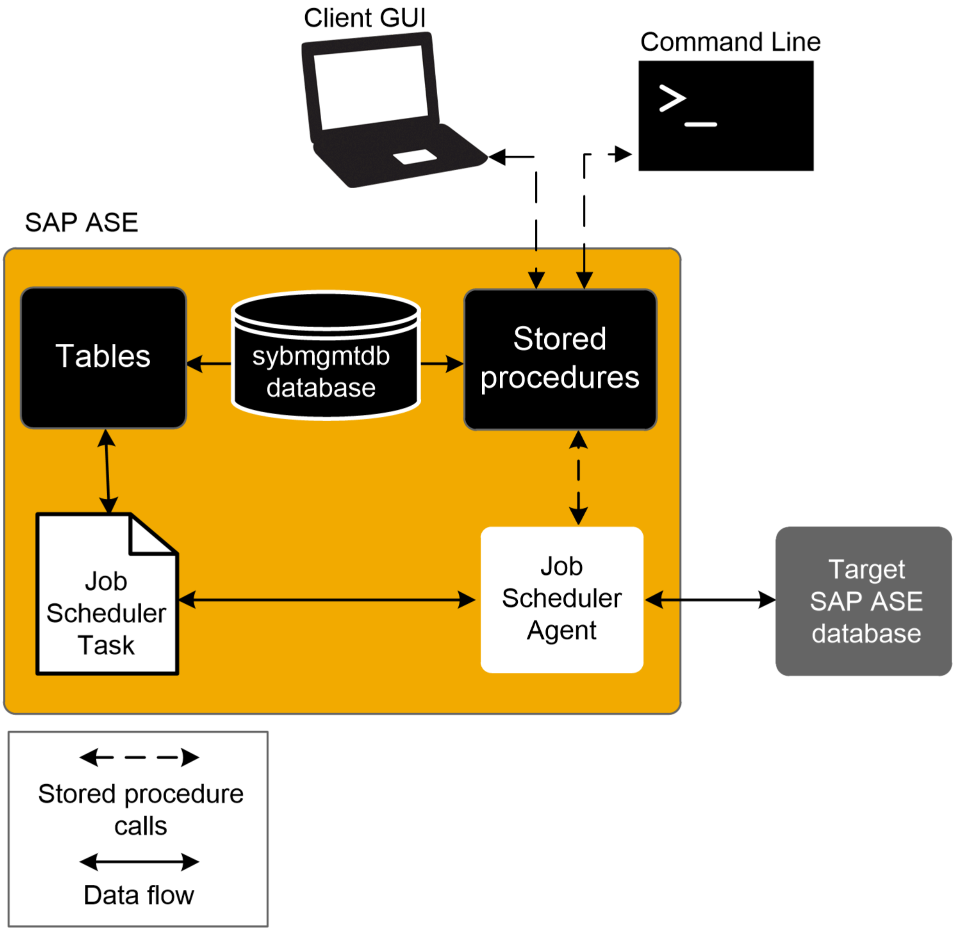
When the JS Agent retrieves the job information from the Job Scheduler database, sybmgmtdb, it logs in to the target server and issues the job commands. When the job completes, JS Agent logs any result or output to the log tables in the sybmgmtdb database.

All of the information about jobs, schedules, scheduled jobs, and data needed by the JS Task for internal processing is stored in the sybmgmtdb database. Stored procedures are used to access the data and make it available to the GUI (SAP ASE Cockpit), the JS Agent, and the command line interface. Only the JS Task accesses data directly from the sybmgmtdb database.

JS Task wakes up on demand, either by a job scheduled to be run or by a wake-up event that has occurred, such as the creation of a new job. At regular intervals (configurable by the database administrator), JS Task scans the sybmgmtdb database tables and collects applicable schedule information to maintain its dedicated server tables js_callouts and js_history.

Using the data it receives, the GUI helps you create and schedule jobs, view job status and job history, and control jobs. It also provides an administration feature that turns the Job Scheduler internal task on and off and, therefore, allows Job Scheduler to process and execute scheduled jobs.

Templates are an important tool you can use to define parameterized tasks for self-management of the database, such as database backups, reorganization rebuilds, modification of configuration parameters, and statistics updates and monitoring. They are implemented as batch T-SQL commands for which you provide parameter values. Database administrators can use templates to generate jobs and schedule them to run at specific times.



As shown in the Process Flow figure, the client GUI, and the command line interface interact with the Job Scheduler database tables using stored procedures. The stored procedures perform the actions requested by the user and maintain the definitions of the job commands, schedules, and other job information in the database tables.

The JS Task reads the job and schedule information from these tables and determines when a scheduled job needs to be executed. At the appropriate time, the JS Task informs the JS Agent of the job that is scheduled to run. JS Agent retrieves the job information and executes it on the target SAP ASE database. The target database can be either local (on the same server as Job Scheduler), or remote (anywhere outside the JS Agent, including the same server where Job Scheduler is installed). The JS Agent collects the output and any result sets and stores this output in the appropriate tables in the Job Scheduler `sybmgmtdb` database.

1.3 Security

Job Scheduler uses predefined roles to manage security: `js_user_role`, `js_admin_role`, and `js_sa_role`.

You must have sa_role to execute `sp_configure` for Job Scheduler configuration parameters. See Configuration Parameters [page 23].

- `js_user_role` – allows user to create, modify, delete, and run jobs, schedules, and scheduled jobs using SAP ASE Cockpit or the command line procedures. The owner of a job, schedule, or scheduled job, can limit which other users can see and use the object, but has no control over objects owned by another user.
- `js_admin_role` – grants all the permissions of the `js_user_role` to the user.
- `js_sa_role` – allows access to the `sa` account.

> i Note
>
> Use the `sp_addexternlogin` command whenever the `sa` password is changed.

1.3.1 Changing Passwords

Job Schedule may stop working if you change your password.

This is because Job Scheduler uses a loopback server entry and issues `sp_externalogin` to register the Job Scheduler user as a remote user, caching the original password.

If you change your password, you must also use `sp_adddexternlogin` to update the registered external password.

1.3.2 Shared Objects

By default, scheduled jobs can only be run by the owner.

However, you can set the `shared_run` property on a scheduled job to make it available for other Job Scheduler users to run. You can also set it to share jobs and schedules with other users. This allows them to create their own scheduled jobs using these shared jobs and schedules. Sharing jobs, schedules, and scheduled jobs does not give other Job Scheduler users permission to modify or delete them; the sharing is read-only.

2 Configuring and Running Job Scheduler

You can install and configure Job Scheduler as part of the SAP ASE installation, or you can install Job Scheduler manually, after you have installed SAP ASE.

When you upgrade to a new SAP ASE server, you must also upgrade Job Scheduler. See *Upgrading Job Scheduler* in the installation guide for your platform.

2.1 Installing Job Scheduler Manually

You can install Job Scheduler manually using the `installjsdb` script.

Procedure

1. Create a device called `sybmgmtdev` that has a size of at least 160 MB.
2. Run the `installjsdb` script:

   ```
   isql -Usa -Psa_password -Sservername -i $SYBASE/$SYBASE_ASE/scripts/
   installjsdb
   ```

   > **i Note**
   >
   > You must include the directory that contains location of the `isql` executable (`$SYBASE/$SYBASE_OCS/bin`) in your path.

   The `installjsdb` script looks for the `sybmgmtdb` database. If it exists, the script creates Job Scheduler tables and stored procedures. If it does not exist, the script looks for the `sybmgmtdev` device on which to create the `sybmgmtdb` database, the tables, and stored procedures.

   > **i Note**
   >
   > If the `installjsdb` script finds neither a `sybmgmtdev` device nor a `sybmgmtdb` database, it creates a `sybmgmtdb` database on the `master` device. If this happens, SAP strongly recommends that you remove the `sybmgmtdb` database from the `master` device to prevent cluttering it and to make recovery easier in the case of a disk failure.

3. Create a directory services entry for the JS Agent using `dscp`, `dsedit`, or a text editor as appropriate for your specific directory services setup and operating system. SAP suggests the name "`<server name>`_JSAGENT":

   ```
   servername_JSAGENT
       master tcp ether <server_machine> <port_number>
   ```

```
     query  tcp ether <server_machine> <port_number>
```

where:

- `<server_machine>` is the name of the machine on which the Job Scheduler server is installed.
- `<port_number>` is the port on which the JS Task communicates with the JS Agent. You must specify a port currently not in use.

See the *System Administration Guide: Volume 1* for more information about directory services.

4. Create an entry in the `sysservers` table:

   ```
   sp_addserver SYB_JSAGENT, null, <servername_JSAGENT>
   ```

5. Enable Job Scheduler:

   ```
   sp_configure “enable job scheduler”, 1
   ```

6. Set the number of user connections. SAP recommends that you increase the number of user connections by at least 3 times the amount you increase job scheduler tasks. For example, if you increase the value for job scheduler tasks from the default (4) to 8, you should increase the value for number of user connections by 12 because (8 - 4) x 3 = 12.

   > **Note**
   >
   > Changing the number of user connections may also require you to increase the value for `max memory`.

7. To start Job Scheduler, you can either restart the server or use the following command:

   ```
   use sybmgmtdb
   go
   sp_sjobcontrol @name=NULL, @option="start_js"
   go
   ```

2.2 Setting Up Access to Target Servers

The JS Agent runs outside of SAP ASE; because of this, target servers consider it a remote user. You must define each target server (even when Job Scheduler is installed on the target server) and login to the local server.

Procedure

1. Create a directory services entry for each target server using `dscp`, `dsedit`, or a text editor as appropriate to your directory services setup:

   ```
   <target_servername>
      master tcp ether <targetserver_machine> <port_number>
      query tcp  ether <targetserver_machine> <port_number>
   ```

   where:

- <target_servername> is the name of the server on which you want to schedule and run a job.
- <targetserver_machine> is the name of the machine on which you want to schedule and run a job.
- <port_number> is the port on which JS Agent connects to the target server.

See the *System Administration Guide: Volume 1* for more information about directory services.

2. On the SAP ASE server that is running Job Scheduler, create an entry in the `sysservers` table for each target server on which you want to run a job. A target server can be remote or it can be the Job Scheduler server itself. In either cases, add a remote entry for the target server in the `sysservers` table:

Use `sp_addserver` to add a remote target server. The server class must be `ASEnterprise`:

```
sp_addserver <target_servername, >ASEnterprise, <directoryservices_name>
go
```

where:

- <target_servername> is an alias for the local server. When you install Job Scheduler with SAP ASE, "loopback" is used as the alias for the server running Job Scheduler.
- <directoryservices_name> is the name used for the local server in the directory services file.

> **i Note**
>
> A remote target server can be the same server as your Job Scheduler installation.

The `sysservers` table already contains a local reference to the server where Job Scheduler is installed; however, to allow this server to be a target server, create a remote entry for the server in the `sysservers` table:

```
sp_addserver <target_servername>, ASEnterprise,    directoryservices_name
go
```

where:

- <target_servername> is the server on which you want to schedule and run a job, in this case, the local (Job Scheduler) server.
- <directoryservices_name> is the name used for this target server in the directory services file.

3. Add logins for Job Scheduler users:

```
sp_addexternlogin <target_servername>, <localname>, <remotename>, <remotepwd>
go
```

where:

- <target_servername> is the server on which you want to schedule and run a job.
- <localname> is the login name on the Job Scheduler server.
- <remotename> is the login name on the target server.
- <remotepwd> is the password on the target server.

> **i Note**
>
> Add external logins for all users for all target servers, even if the target server is the server where Job Scheduler is installed.

You can now create jobs, schedules, and scheduled jobs using stored procedures from the command line or using SAP ASE Cockpit.

2.3 Setting Up Users for Job Scheduler

Add users and grant roles for Job Scheduler tasks.

Prerequisites

Each user must have an SAP ASE login before you set up users in Job Scheduler.

Procedure

1. Add users to the `sybmgmtdb` database:

   ```
   use sybmgmtdb
   go
   sp_adduser <login_name>
   go
   ```

   where `<login_name>` is the user's login name on the Job Scheduler server.
2. Grant the appropriate roles to users.

   For Job Scheduler administrators:

   ```
   grant role js_admin_role to <login_name>
   ```

   For Job Scheduler users:

   ```
   grant role js_user_role to <login_name>
   ```

2.4 Enable or Disable JS Agent Core Dumps

You can use the `disable jsagent core dump` configuration parameter to enable or disable JS Agent core dumps.

When `disable jsagent core dump` is off, the core dump for JS Agent is enabled during signal handling, allowing you to diagnose JS Agent failures. Setting `disable jsagent core dump` to on (1) disables core dumps and is not recommended.

The JS Agent can generate core dumps files when it encounters a fatal signal (or an exception on Windows). This can be useful when diagnosing Job Scheduler issues. The default location for a core dump is determined by the environment variable `JSA_CORE_PATH`.

- If `JSA_CORE_PATH` is not set, or you do not have write permission to the set path, the `$SYBASE/$SYBASE_ASE/install` path is used.

- If `$SYBASE` or `$SYBASE_ASE` is not set, the directory from where the JS Agent process was started is used. $SYBASE path is used for Windows.

> **i Note**
>
> For the AIX platform, the core file is generated in the directory from where the JS Agent process was started.

2.5 Automatic Restart of Job Scheduler

You can restart Job Scheduler while JS Agent and JS Task are in various states.

Use these configuration parameters to manage automatic restart of Job Scheduler:

- `max js restart attempts` – restricts the number of restart attempts, and prevents the Job Scheduler restart feature from going into an infinite loop. The value 0 (zero) indicates that the Job Scheduler Auto restart feature is disabled.
- `enable js restart logging` – enables or disables diagnostics logging after the restart of Job Scheduler.
- `js heartbeat interval` – the interval (in minutes) between two JS Agent heartbeat checks.

Automatic restart occurs when:

- JS Agent is in an inconsistent state but still running – This may be as a result of the connection in a pool stopping, or if the target server has stopped while collecting results. In this situation, both the JS Task and JS Agent are shut down and restarted.
- JS Task is in an inconsistent state and is going to shut down Job Scheduler – This may be as a result of an inconsistency detected in Job Scheduler tables; alternatively, it may be due to incorrect or missing job callouts. In this situation, both the JS Task and JS Agent are shut down and restarted.
- JS Agent has stopped running – This may be as a result of JS Agent hitting a fatal signal, JS Agent shutting down due to inconsistency detected in its operation, or an external event, such as the killing of a JS Agent process. JS Task periodically monitors JS Agent health. If it finds that the JS Agent has stopped, it restarts the JS Agent.
- JS Task unexpectedly stops running – In this situation, you must manually restart the server. A new JS Task aborts the existing JS Agent, if any, and starts a new JS Agent.

2.6 Logging of Job Scheduler and Job Execution Errors

By default, Job Scheduler errors and job execution errors are logged to the JS Agent log. You can enable additional diagnostics information using the 3641 trace flag.

3 Using Templates to Create Jobs

You can use templates to manage your SAP ASE databases.

A template helps database administrators to create scripts that perform specific database tuning and maintenance tasks; the Transact-SQL commands created by the template becomes the text for the job itself. You can then customize a particular server and schedule them to run on any network server at any specified time. As a result, templates can save database administrators the work of creating their own Transact-SQL or `cron` scripts.

In SAP ASE Cockpit, a wizard leads you through creating script that invokes the procedures for managing your databases.

3.1 Installing Stored Procedures

Before you can run jobs on target servers, you must install the stored procedures used by the templates on the target servers where you want your jobs to run.

Procedure

1. Go to the directory that contains the installation utility, which is, by default, `$SYBASE/$SYBASE_ASE/jobscheduler/Templates/sprocs`.
2. Run `installTemplateProcs`:
   - On UNIX:

     ```
     installTemplateProcs <target_servername ><username> <password>
     ```

   - On Windows:

     ```
     installTemplateProcs.bat <target_servername ><username> <password>
     ```

   where:
   - `<target_servername >` is the name of the target server.
   - `<username>` is the login name on the target server for the user who is installing the templates.
   - `<password>` is the password on the target server for the user who is installing the templates.

   The stored procedures are installed in the `sybsystemprocs` database on the target server.

3.2 Installing Templates

Before you can use the templates, you must install the templates into the Job Scheduler you select to manage your jobs.

Procedure

1. Go to the directory containing the installation utility, which is, by default, `$SYBASE/$SYBASE_ASE/jobscheduler/Templates/xml`.
2. Set these environment variables:
   - `<SYBASE_ASE_SA_USER>` – a user with system administrator privileges.
   - `<SYBASE_ASE_SA_PWD>` – the password for the user with system administrator privileges.
3. Run `installTemplateXml` :
   - On UNIX:

     ```
     installTemplateXml< servername ><machinename> <serverport>
     ```
   - On Windows:

     ```
     installTemplateXml.bat< servername ><machinename> <serverport> <language>
     ```

   where:
   - `<servername >` is the name of the server where Job Scheduler is installed.
   - `<machinename>` is the name of the machine hosting the Job Scheduler server.
   - `<serverport>` is the port number for connecting to the Job Scheduler server.
   - `<language>` is the that indicates which language version of the templates is installed:
     - English — en
     - French — fr
     - Japanese — ja
     - Korean — ko
     - Simplified Chinese — zh

     If you do not specify a language code, English templates are installed.

The Job Scheduler templates are installed on the Job Scheduler server.

3.3 Using Job Scheduler Templates

Use Job Scheduler templates to create backup jobs, generate query plans, or reorganize or reconfigure your database.

3.4 Backup Templates

Backup templates, which enable you to easily create database backup jobs, help preserve your data.

- Backup database to disk template – uses the `dump database` command to back up one or more databases to disk. When creating a job from this template, you specify the database name and dump directory. You can also specify whether to compress or stripe the database, and whether to include the server name and date in the dump file name.
- Backup transaction log to disk template – uses the `dump transaction` command to back up the transaction log for one or more databases. When creating a job from this template, you specify the database name and dump directory. You can also specify whether to compress or stripe the database, and whether to include the server name and date in the dump file name. Additionally, you can specify a time threshold so if that amount of time has passed since the last dump, the log is dumped. Similarly, you can specify a row threshold to force a dump of the log if it contains at least that number of rows.

See *Backing Up and Restoring User Databases* in the *System Administration Guide: Volume 2*.

3.5 Statistics Management Templates

The statistics management templates help keep statistics current, enabling you to generate efficient query plans.

- Server update statistics template – uses the `update statistics` command to update statistics on all tables in all server databases. The template allows you to supply values for the various options of the `update statistics` command, while the table, column, and index values are retrieved from the system tables.
  Additionally, you can provide threshold values for data change, page count, and row count. If you supply any of these optional values, they are used to determine whether the `update statistics` command should run. If the current value for any one of these thresholds meets or exceeds the threshold for a given table, `update statistics` is executed on that table. After `update statistics` runs, the `sp_recompile` procedure is executed on the table.
- Update statistics template – uses the `update statistics` command to update statistics at the table, partition, column, and index levels. Additionally, you can provide threshold values for data change, page count, and row count. Data change is a metric that allows you to track the number of changes to a table, column, or partition. You can use it to determine whether the `update statistics` command might improve performance.
- Delete statistics template – uses the `delete statistics` command to delete statistics for all columns in the specified table, for a specific column in the table, or for a partition and local indexes on a partition. You

must supply the database and table name. You can also specify a list of columns for which to delete statistics. If you do not specify a list of columns, the entire table is deleted.

3.6 Reorganization Templates

The reorganization templates help keep your database organized, by preventing lost table space and noncontiguous table data.

- Rebuild indexes template – runs the `reorg rebuild` command on an index or indexes for a table. You must specify the database, table, and indexes required by the `reorg rebuild` command. You can also specify an index partition name and a database backup after the `reorg rebuild` command executes.
- Rebuild tables template – runs the `reorg rebuild` command on the entire table. You must specify the database and tables required by the `reorg rebuild` command. You can also back up the database after the `reorg rebuild` command executes.
- Reclaim index space template – reclaims unused space left on a page as a result of deletions and updates that shorten rows. It invokes the `reorg reclaim space` command to reorganize an index’s data pages. If you specify more than one index, space is reclaimed for each index. You can also specify the name of the index partition that contains the specified index.
- Reclaim table space template – reclaims unused space left on a page as a result of deletions and updates that shorten rows. It invokes the `reorg reclaim space` command to reorganize a table’s data pages. If you specify more than one table, space is reclaimed for each table. You can provide a partition name to limit the `reorg reclaim space` command to that partition of the table on the specified partition. You can also specify a partition name to limit reorganization to a partitioned portion of a table.

3.7 Reconfiguration Templates

The reconfiguration templates help keep user connections, metadata cache, and locks configured properly to meet your current demands.

- Reconfigure locks template – allows you to set up automatic reconfiguration for the number of locks allowed by the server at any one time, based on server data and user input. You can specify the number of locks to add.
- Reconfigure metadata cache template – create jobs to determine the proper number of each metadata object type for your server. The metadata cache is a reserved area of memory that is used for databases, indexes, and objects. While creating the job, supply values that are used at runtime to determine the proper number of objects needed for your current server conditions. The job determines the current number of active objects and adjusts the allowable number of objects of the specified type. Alternatively, you can specify the exact number for one or all metadata object types, bypassing the self-tuning feature of this template.
- Reconfigure user connections template – calls `sp_configure` to set the number of user connections. You can provide the new number of user connections, or the template can calculate it using information you supplied in the wizard.

4 Using Job Scheduler at the Command Line

Create and schedule jobs from the command line.

4.1 Creating a Hello World Job

Create a job that prints "Hello World" and a list of its history.

Prerequisites

You must have permission to log in to your target server using the user name and password that was supplied to the `sp_addexternlogin` stored procedure.

Procedure

1. Create the job:

```
use sybmgmtdb
go
declare @jobcmd varchar(255), @jobid int
select  @jobcmd='jcmd=print “Hello World.”,server=<YOUR_SERVER>'
+ ',starttime=' + convert(varchar(32),getdate())
exec @jobid=sp_sjobcreate 'sjname=hello', @jobcmd
go
```

   where `<YOUR_SERVER>` is the name of your target server.

2. View the scheduled job that you created:

```
> exec sp_sjobhelp 'sjname=hello'
> go
```

   The following summary of the scheduled job is returned:

```
sjob_id: 127  name: 'hello'
owner          : jsadmin1
created        : Jul 14 2005 4:42AM
state          : enabled
job name       : 114 - 'job_114'
schedule name  : 115 - 'sched_115'
server         : pgibson_js
```

```
-- job --------:
description    :
owner          : jsadmin1
created        : Jul 14 2005 4:42AM
-- schedule ---:
description    :
owner          : jsadmin1
created        : Jul 14 2005 4:42AM
starttime      : 04:42
startdate      : 14 Jul 2005
```

3. View the short list history of executing a scheduled job:

   ```
   sp_sjobhistory 'sjname=hello', @option='list_short'
   go
   ```

   The following short list is returned:

   ```
   sjob_id  sjob_jobname  sjob_schedname  sjob_server    sjob_state
   -------  ------------  --------------  -----------    ----------
   127      job_114       sched_115       pgibson_js     C2
   sjob_start             sjob_user_run   sjob_user_req  sjob_size
   ----------             -------------   -------------  ---------
   Jul 14 2005 4:42AM    jsadmin1        jsadmin1       51
   ```

4. View the output from executing a scheduled job:

   ```
   sp_sjobhistory 'sjname=hello', 'list_output'
   go
   ```

   The following output is returned:

   ```
    jsout_run_id   jsout_seqno   jsout_size    jsout_text
   ------------   -----------   -----------   ----------
   144            0             51            Changed database context to
   'master'.
   hello world
   ```

4.2 Job State Codes

Job state codes are returned in the history table.

State Name	State Code	Description
waiting	W	An initial state for a job that has been created or is waiting to schedule.
queued	Q	Job Scheduler may be waiting for a free thread to run the job.
busy	B	Job Scheduler cannot start the job, which is already running.
runable	R1	Job Scheduler has started the job.
running	R2	The job is running.

State Name	State Code	Description
completing	C1	The job has completed, and Job Scheduler is cleaning up logs, history, and so on.
completed	C2	The job has completed. Post processing SQL is complete.
terminating	T1	Job Scheduler is terminating the job, killing the thread.
terminated	T2	Job Scheduler has terminated the job. Post processing SQL is complete.
timing-out	X1	The job has timed out, and Job Scheduler is terminating the job.
timed-out	X2	The job has timed out and been terminated.
missed	M	The scheduled run of the job has been missed because Job Scheduler was inactive.

4.3 Create a Schedule

Create a one-time or repeating schedule in from the command line.

A repeating schedule must have a start time and an end time.

> **Note**
>
> Schedule names must begin with a letter.

This example creates a schedule that operates every five minutes, between 08:00 and 18:00, and is valid every day of the week:

```
sp_sjobcreate @name='sname=every5m_8to6',@option='repeats=5minutes,
starttime=08:00am, endtime=18:00'
```

This example creates a schedule that operates every hour on Saturdays and Sundays between 08:00 and 18:00:

```
sp_sjobcreate @name='sname=hourly_8to6_weekends',
@option='repeats=1hour,starttime=08:00am,endtime=18:00,days=Saturday:Sunday
```

This example creates a schedule that operates at 04:00 on the first and last day of every month, where `32` is the last day of the month:

```
sp_sjobcreate @name='sname=run_4am_1st_and_last',
@option='starttime=04:00,endtime=04:00,dates=1:32'
```

This example creates a schedule that operates at 09:00, beginning the first of January and ending the first of February:

```
sp_sjobcreate @name='sname=run_daily_9am_Jan',
@option='starttime=09:00,endtime=09:00,repeats=1day,startdate=1 January
2005,enddate=1 February 2005'
```

> i Note
>
> The schedule start time is inclusive, which means that a job begins at the start time. The schedule end time is exclusive, which means that a job runs up to, but does not include the end time. For example, if a schedule has a start time of 13:00 and an end time of 16:00 and repeats every hour, it runs at 13:00, 14:00, and 15:00, but does not run at 16:00.

4.4 Creating a Scheduled Job

Create a scheduled job that gathers statistics on a trial basis for one month.

Context

> i Note
>
> Schedule names must begin with a letter.

Procedure

1. Create the schedule:

   ```
   sp_sjobcreate @name='sname=Aug_stats_trial',
   @option='starttime=23:00,endtime=23:00,repeats=1day,startdate=1 August
   2005,enddate=31 August 2005'
   ```

2. Create the job:

   ```
   sp_sjobcreate @name='jname=new_stats_proc',
   @option='jcmd=''exec statsdb..
   new_stats_proc'',jdesc=New statistics
   proc.,jproperties=shared'
   ```

3. Create a scheduled job on a server called “devtest1”:

   ```
   sp_sjobcreate @name='sjname=new_stats_devtest1',
   @option='sname=Aug_stats_trial,jname=new_stats_proc,server=devtest1'
   ```

   > i Note
   >
   > The server name assigned to the scheduled job is the network name (the `<pname >` property in `sp_addserver`) in the interfaces file, not the logical name defined in the `sysservers` table.

4.5 Deleting a Scheduled Job

Delete a scheduled job from the command line.

Procedure

Delete a scheduled job called “new_stats_devtest1”:

```
sp_sjobdrop 'sjname=new_stats_devtest1'
go
```

4.6 Modify a Scheduled Job

Modify an existing scheduled job.

The examples below modify the “Hello World” job:

```
use sybmgmtdb
go
declare @jobcmd varchar(255), @jobid int
select  @jobcmd='jcmd=print “Hello World.”,server=<YOUR_SERVER>'
+ ',starttime=' + convert(varchar(32),getdate())
exec @jobid=sp_sjobcreate 'sjname=hello', @jobcmd
go
```

This code modifies the job to run every day at 09:00:

```
sp_sjobmodify 'sjname=hello', 'starttime=09:00,repeats=1day'
go
```

This code modifies the job `timeout` property to 120 minutes::

```
sp_sjobmodify @name='sjname=hello',
@option='default_timeout=120'
go
```

This code modifies the job to run at 09:00 on Mondays and Fridays:

```
sp_sjobmodify 'sjname=hello', 'starttime=09:00,endtime=09:00,days=Monday:Friday'
go
```

This code changes the target server for the scheduled job:

```
sp_sjobmodify 'sjname=hello', 'server=prodASE'
go
```

4.7 Invoking Stored Procedures on a Target Server

Scheduled jobs often invoke stored procedures on a target server to perform a required function.

This example shows a scheduled job that invokes `sp_who`:

```
use sybmgmtdb
go
declare @jobcmd varchar(255), @jobid int
select  @jobcmd='jcmd=exec sp_who,server=<YOUR_SERVER>'
+ ',starttime=' + convert(varchar(32),getdate())
exec @jobid=sp_sjobcreate 'sjname=hello', @jobcmd
go
```

where `<YOUR_SERVER>` is the name of your target server.

When Job Scheduler executes a job it prefixes the job text with SQL that creates the job `runid` and the scheduled job ID. Therefore, the call to `sp_who` needs to use `exec` to invoke the stored procedure.

4.8 Managing Jobs

You can use `sp_sjobcontrol` to perform ad hoc actions on Job Scheduler.

You can:

- Terminate a running job
- Run a scheduled job immediately
- Enable and disable a scheduled job

Related Information

sp_sjobcontrol [page 44]

5 Configuration Parameters

Job Scheduler includes a set of configuration parameters.

5.1 enable job scheduler

Determines whether Job Scheduler starts when the SAP ASE server starts.

Summary	Description
Default value	0 (off)
Valid values	0 (off), 1 (on)
Status	Dynamic
Display level	Comprehensive
Required role	System administrator
Configuration group	SQL Server Administration

5.2 enable js restart logging

Enables or disables diagnostics logging after the restart of Job Scheduler.

Summary	Description
Default value	0 (off)
Valid values	0 (off), 1 (on)
Status	Dynamic
Display level	10
Required role	System administrator
Configuration group	SQL Server Administration

5.3 job scheduler interval

Sets the interval when the Job Scheduler checks which scheduled jobs are due to be executed.

Summary	Description
Default value	1 (in seconds)
Valid values	1 – 600
Status	Dynamic
Display level	Comprehensive
Required role	System administrator
Configuration group	SQL Server Administration

5.4 job scheduler memory

Determines the size of the memory pool (which is of type `bucketpool`) assigned to the Job Scheduler.

Summary	Description
Default value	8 MB
Valid values	4 – 1024
Status	Dynamic
Display level	Comprehensive
Required role	System administrator
Configuration group	SQL Server Administration

5.5 job scheduler tasks

Sets the maximum number of jobs that can run simultaneously through Job Scheduler.

Summary	Description
Default value	4
Valid values	1 – 640
Status	Dynamic
Display level	Comprehensive
Required role	System administrator
Configuration group	SQL Server Administration

If you increase the value of `job scheduler tasks` to a higher value, increase the value for `number of user connections` by at least twice the value you incremented `job scheduler tasks` before starting the Job Scheduler.

However, if the SAP ASE running the scheduled jobs is the same SAP ASE that is hosting the Job Scheduler, you must increase the value for `number of user connections` by three times the value you incremented `job scheduler tasks` before starting the Job Scheduler.

Increasing the `number of user connections` may require that you increase the value for `max memory`.

For compatibility with RAP – The Trading Edition R4, you must set `job scheduler tasks` to 32.

If you set the value of `job scheduler tasks` to "default" before you upgrade SAP ASE, the server automatically sets the new default to 4.

5.6 js heartbeat interval

Specifies the intervals between two JS Agent heartbeat checks, in minutes.

Summary	Description
Default value	1
Valid values	1 – 1440
Status	Dynamic
Display level	10
Required role	System administrator

Summary	Description
Configuration group	SQL Server Administration

5.7 js job output width

Determines the line width the output uses for jobs stored in the `js_output` table.

Summary	Description
Default value	80
Valid values	1 – 32768
Status	Dynamic
Display level	Comprehensive
Required role	System administrator
Configuration group	SQL Server Administration

5.8 js restart delay

Sets the delay period between two Job Scheduler auto restart attempts after abnormal shutdown of Job Scheduler.

Summary	Description
Default value	60
Valid values	0 – 1440
Status	Dynamic
Display level	Comprehensive
Required role	System administrator
Configuration group	SQL Server Administration

5.9 max js restart attempts

Restricts the number of restart attempts and prevents the Job Scheduler restart feature from going into an infinite loop.

Summary	Description
Default value	3
Valid values	0 – 10. The value 0 indicates that the Job Scheduler Auto restart feature is disabled.
Status	Dynamic
Display level	10
Required role	System administrator
Configuration group	SQL Server Administration

5.10 percent output free

Specifies the percentage of reserved space kept free in `sybmgmtdb` that is reserved for Job Scheduler output.

Summary	Description
Default value	50
Valid values	0 – 100
Status	Dynamic
Display level	Intermediate
Required role	System administrator
Configuration group	SQL Server Administration

For example, if you use the default value, SAP ASE starts purging the oldest history records to make room for new records when 50 percent of `sybmgmtdb` is filled.

5.11 suppress js max task message

Prevents SAP ASE from printing the Job Scheduler `js maxtask` error messages to the error log.

Summary	Description
Default value	0 (off)
Valid values	0 (off), 1 (on)
Status	Dynamic
Display level	10
Required role	System administrator
Configuration group	Application Functionality

> i Note
>
> The default value for `suppress js max task message` depends on the value to which `enable functionality group` is set. If you set `enable functionality group` to:
>
> - 0 – the default value for `suppress js max task message` is 0.
> - 1 – the default value for `suppress js max task message` is 1.
>
> However, if you set `suppress js max task message` to 1, it uses a value of 1 regardless of what you set `enable functionality group` to.
>
> See `enable functionality group`.

6 Command Syntax

Job Scheduler includes a set of stored procedures for working with scheduled jobs.

The stored procedures for creating jobs, schedules, and scheduled jobs use the server user name, obtained via the `suser_name` system function, to record the owner. The stored procedures also work with the underlying job and schedule objects, providing an interface to control scheduled jobs that are currently running and managing the history and job output that they produce.

The stored procedures accept a name or ID argument, which specifies the object they should operate on. Stored procedures that operate on several different objects, such as a job, a schedule, or a scheduled job, accept a prefix to the name or ID argument, which specifies the sort of object the name or ID refers to: for schedules – `@sname;` for jobs – `@jname`.

For example:

```
@name= ‘sname=daily_schedule’
```

```
@name= ‘jname=run_update_stats’
```

The name or ID default is to reference a scheduled job.

General Usage

- Command names, command keywords, the word “null”, data items, and statistic types are case-insensitive. File names, view names, and other user-supplied names are case-sensitive.
- If a parameter value contains embedded spaces (such as those in data items, statistic types, and date-time specifications), surround the value with quotes. Matched pairs of single-quote or double-quotes are valid delimiters.
- If the parameter value contains an embedded quote that is the same as the character used to delimit the entire value, supply a pair of the quotes within the parameter value. Job Scheduler compresses the pair of quotes to a single character.
- The word “null” within quotes is not considered a keyword.
- You can enter Job Scheduler commands on multiple lines.

6.1 Command Line Stored Procedures

Use stored procedures to add, modify, delete, run, or obtain information about scheduled jobs.

- Adding, modifying, and deleting – there are two methods for creating a scheduled job: by specifying both the job and schedule information in one operation, or by defining the job and the schedule separately and then combining them into a scheduled job.

- Reporting and listing – there are stored procedures to obtain information about the jobs and schedules configured in the Job Scheduler database. The information can be provided in a formatted report or as a simple list.
- Running jobs – you can run jobs by scheduling them or using ad hoc commands.
- Jobs history and output – you can manage job history and job output separately. Deleting job history also deletes any corresponding job output, but deleting job output records does not delete job history. This flexibility allows for different rules to govern the amount of job history and job output you retain.
  A Job Scheduler user who does not have the `js_admin_role` can only view and manipulate the job history and job output from jobs that have his or her user name recorded in the history entry. However, a user who has `js_admin_role` can use the `all` option to override this limitation and extend the scope to the history and output to all Job Scheduler users.
- Jobs history and output – you can manage job history and job output separately. Deleting job history also deletes any corresponding job output, but deleting job output records does not delete job history. This flexibility allows for different rules to govern the amount of job history and job output you retain.
  Job Scheduler user who does not have the `js_admin_role` can only view and manipulate the job history and job output from jobs that have his or her user name recorded in the history entry. However, a user who has `js_admin_role` can use the `all` option to override this limitation and extend the scope to the history and output to all Job Scheduler users.

Procedure Name	Procedure Purpose
Create, modify, delete	
`sp_sjobcreate`	Create scheduled jobs, jobs, or schedules
`sp_sjobcmd`	Manage the SQL command text of a job
`sp_sjobmodify`	Modify scheduled jobs, jobs, and schedules
`sp_sjobdrop`	Remove scheduled jobs, jobs, and schedules
Reporting, listing	
`sp_sjobhelp`	Report and list of scheduled and running jobs
Running jobs	
`sp_sjobcontrol`	Job Scheduler administration and control of scheduled or running jobs
Jobs history, output	
`sp_sjobhistory`	View and administration of the job history and output
Agent Configuration	
`sp_jsconfigure`	Configures the Job Scheduler Agent.

6.2 sp_jsconfigure

Configures the Job Scheduler Agent.

Syntax

```
sp_jsconfigure [<option> [, <value>]]
```

Parameters

<option>

One of:

- <interfaces path> – path to the interface file
- <errorlog> – path to the errorlog
- <help> – displays the syntax for sp_jsconfigure

<value>

Specifies the value to which you are setting <option>.

Examples

Example 1

Displays the sp_jsconfigure syntax:

```
sp_jsconfigure "help"
Usage: sp_jsconfigure [option [, value]]
        where option : 'interfaces path', ' errorlog ', 'help'
              value  : value to set for the 'option'
```

Example 2

Sets the path to the interfaces file:

```
sp_jsconfigure 'interfaces path', "$SYBASE_ASE/data"
```

Example 3

Sets the path to the errorlog:

```
1> sp_jsconfigure "errorlog", "$SYBASE_ASE/data/js.log"
```

Example 4

Displays the values to which you have set sp_jsconfigure:

```
sp_jsconfigure
Parameter Name                   Config
Value

 --------------------------------
------------------------------------------------------------------------------
----------------------------
 interfaces path                  $SYBASE_ASE/
data
 errorlog                         $SYBASE_ASE/data/
js.log
```

Usage

- Use the installjsdb script to install sp_jsconfigure. By default, installjsdb is located in $SYBASE/$SYBASE_ASE/scripts.
- Restart JS Agent for the configuration changes to take effect.
- sp_jsconfigure uses default values if you do not supply values for <interfaces path> or <errorlog>.
- Job Scheduler does not start if you include paths for the <interfaces path> or <errorlog> options that do not exist.
- <interfaces path> must include an interfaces file with an entry for the Job Scheduler host and target server on which the scheduled job is executed (the target and host server can be the same machine).

Permissions

You must have the js_admin_role to execute sp_jsconfigure.

6.3 sp_sjobcreate

Creates new jobs or new schedules.

sp_sjobcreate serves several purposes:

- When only job information is supplied, it creates a new job.
- When only schedule information is supplied, it creates a new schedule.
- When job and schedule information is supplied, it either combines an existing job and schedule to create a new scheduled job, or it creates a new job and a new schedule, and combines them into a new scheduled job.

By default, the @name argument is the name of a scheduled job. To specify the name of a job or a schedule, prefix the @name argument with jname or sname.

Syntax

```
sp_sjobcreate @name='<jsname>', @options='server, jname, jdesc, jcmd, sname,
sdesc, repeats, properties, starttime, enddate, endtime, days, dates'
```

Parameters

<name (jname, sname, sjname)>

The name of the new job, schedule, or scheduled job.

When you create a name for a job, a schedule, or a scheduled job, the name must begin with a letter. If you create a name beginning with a digit, an error occurs.

<option>

A comma-separated list of the field names and values you use to create a job, a schedule, or a scheduled job. Values are:

- server – the name of the server where the job runs. The default is the local server.
- jname – the name of the job, which must be unique.
- jdesc – comments describing the job.
- jcmd – the SQL text, used for simple jobs when it is easy to provide the text directly.
- sname – the name of the schedule, which must be unique.
- default_timeout – the maximum amount of time permitted for the execution of a job. This value is used by scheduled jobs if the scheduled job's timeout property is not set.
- sdesc – comments describing the schedule.
- timeout – the maximum amount of time allowed for the execution of a scheduled job. This value supersedes the job's default_timeout value.
- repeats – the interval at which the schedule repeats. 0 (zero) or NULL means it does not repeat. The value is a number followed by:
  - day or d
  - days or dd
  - hour or h
  - hours or hh
  - minute or m
  - minutes or mm
- properties – the properties of the job, the schedule, or scheduled job:
  - jproperties (job properties)
    - multi_task

        - `run_as_owner`
        - `no_job_header`
        - `no_sql_batching`
        - `no_conn_redirection`
        - `shared`
    - `sjproperties` (scheduled job properties)
        - `shared_run`
        - `only_at_starttime`
        - `disable_on_faillure`
        - `delete_on_completion`
        - `no_output_log`
    - `sproperties` (schedule property)
        - `continuous_run` – allows you to execute jobs from a starting time, and run according to specified intervals, until an end time.
- `startdate` – the date on which a schedule becomes active.
- `starttime` – the time of day when a scheduled job begins.
- `enddate` – the date on which a schedule becomes inactive.
- `endtime` – the time at which a schedule becomes inactive for the remainder of the day.
- `days` – a list of days separated by colons. Full names or abbreviations in the locale of the server may be used.
- `dates` – a list of dates in a month, from 1 to 31, separated by colons. The number 32 represents the last day of any month.

    > i Note
    >
    > You cannot specify values for both days and dates.

Returns

One of:

- The job ID for a new job.
- The schedule ID for a new schedule.
- The scheduled job ID for a new scheduled job.
- An error code.

Examples

Example 1

Creates a new job called “find_old_logins” that consists of running a stored procedure:

```
sp_sjobcreate @name=’jname=find_old_logins’,
@option=’jcmd=exec sp_find_old_logins,jproperties=shared’
```

Example 2

Creates a new schedule that becomes valid at 1:00 am and repeats every day:

```
sp_sjobcreate @name=’sname=daily 01:00am’,
@option=’repeats=1day,starttime=01:00am, endtime=02:00am’
```

Example 3

Creates a new scheduled job to run on server “dev1” using the existing job “find_old_logins” and schedule called “daily 01:00am”:

```
sp_sjobcreate @name=’dev1_old_logins’,
@option=’server=dev1,jname=find_old_logins, sname=daily 01:00am’
```

Example 4

Creates a new schedule called “evening_sales_report,” using new job called “load_sales_data,” and a new schedule that runs every Monday, Wednesday, and Friday at 23:00. The schedule is given a default name based upon the ID value in the schedule table.

```
sp_sjobcreate @name=’evening_sales_report’,
@option=’server=reports, jname=load_sales_data,
jcmd=exec sp_new_sales_data,
starttime=23:00,endtime=23:00,days=Monday:Wednesday:Friday’
```

Example 5

Uses continuous_run to schedule a job to run every 10 minutes, starting at 2:00 p.m. on May 14, 2012 and ending on May 16 at 4:00 p.m.:

```
sp_sjobcreate 'sjname=sjob', @option='repeats=10minutes, startdate=14 May
2012,enddate=16
    May 2012,starttime=02:00pm, endtime=04;00pm,sproperties=continuous_run’
```

Example 6

Runs the scheduled job every 25 hours on repeating days, executing on May 14, 2012 at 2:00 p.m., on May 15, 2012 at 03:00 p.m., and on May 16, 2012 at 04:00 p.m.:

```
sp_sjobcreate 'sjname=sjob', @option='repeats=25hours,startdate=14 May
2012,enddate=16 May
    2012,starttime=02:00pm,endtime=04:30pm, sproperties=continuous_run'
```

Usage

- Although you cannot specify both days and dates for a single schedule, you can bind separate schedules (for example, one using days and another using dates) to the same job.

- A schedule using a `repeats` value of more than one day is not valid with `days` or `dates` values.
- A schedule using a `repeats` value equal to one day is not valid with `days` or `dates` values. By default all days are valid when `repeats` equals one day.
- A schedule using a `repeats` value of less than one day and a `days` value of NULL is, by default, valid on all days.
- If the `startdate` is equal to the `enddate` and the `endtime` is specified, the `endtime` value must be equal to or later than the `starttime` value.
- For schedules with a `repeats` value of one day or more, the `endtime` value has no meaning and is ignored.
- If `endtime` is not specified, the default is midnight.
- If `starttime` is not specified, the default is 00:00am.

6.4 sp_sjobcmd

Allows the SQL text of a job to be manipulated.

When `@option=’list’`, the SQL text is listed. By default, the `@name` argument is the name or ID of a scheduled job. To specify the name or ID of a job, prefix the `@name` argument with `jname`.

Syntax

```
sp_sjobcmd @name=’…’,  @option=’…’, @text=’...’
```

Parameters

`<name>`

The name or ID of a scheduled job or job. Prefixed `<name>` with `jname` to specify the name or ID of a job.

`<option>`

One of.

- `drop` – deletes all SQL text for the specified job.
- `list` – displays the job command text stored in the database.
- `add` – appends to any existing SQL text for the specified job, allowing large SQL batches to be stored in several chunks. When JS Agent runs the job, the SQL text for the job is concatenated.

`<text>`

SQL text to store.

Returns

Returns 0 (zero) or an error code.

Examples

Example 1

Lists the SQL text for the job that is referenced by the scheduled job called “svr1_check_stats”:

```
sp_sjobcmd ’sjname=svr1_check_stats’, ’list’
```

Example 2

Deletes the existing SQL text for the “load_sales_data” job and then stores new SQL text for the job:

```
sp_sjobcmd ’jname=load_sales_data’, ’drop’
sp_sjobcmd ’jname=load_sales_data’, ’add’,
@text=’truncate table  sales_report_data’
go
```

Usage

When you create a job with `sp_sjobcmd`, add a new line after you execute a call:

```
sp_sjobcmd ’jname=load_sales_data’, ’drop’
sp_sjobcmd ’jname=load_sales_data’, ’add’,
@text=’truncate table  sales_report_data’
go
```

6.5 sp_sjobmodify

Allows you to modify any of the fields in a scheduled job, a job, or a schedule.

Syntax

```
sp_sjobmodify @name=’...’, @option=’...’
```

Parameters

<name>

The name or ID of the scheduled job, job, or schedule that is to be modified.

<option>

A comma-separated list of the field names to modify and their new values:

- `sjname` – a new name for the scheduled job that must be unique. You cannot supply a new ID.
- `enable` – 1 enables the scheduled job, 0 disables it.
- `sjproperties` – the properties of the scheduled job.
- `sjowner` – the owner of the scheduled job. Caller must have `js_admin_role` to change the owner.
- `server` – the server on which the scheduled job should run.
- `timeout` – a timeout value, in minutes specified for the scheduled job and server.
- `locale` – the locale to be used by the client connection when the scheduled job runs.
- `jname` – a new name for the job, which must be unique. You cannot supply a new ID.
- `jdesc` – comments describing the job.
- `jcmd` – SQL text used for simple jobs when it is convenient to supply the text directly. Replaces all existing SQL text.
- `properties` – the properties of the job, the schedule, or scheduled job:
    - `jproperties` (job properties)
        - `multi_task`
        - `run_as_owner`
        - `no_job_header`
        - `no_sql_batching`
        - `no_conn_redirection`
        - `shared`
    - `sjproperties` (scheduled job properties)
        - `shared_run`
        - `only_at_starttime`
        - `disable_on_faillure`
        - `delete_on_completion`
        - `no_output_log`
    - `sproperties` (schedule property)
        - `continuous_run` – allows you to execute jobs from a starting time, and run according to specified intervals, until an end time.
- `jowner` – the owner of the job. Caller must have js_admin_role to change the owner.
- `default_timeout` – a timeout value, in minutes, for the job.
- `sname` – a new name for the schedule, which must be unique. You cannot supply a new ID..

- `sdesc –` comments about the schedule.
- `sowner –` must have `js_admin_role` to change the owner.
- `sproperties –` schedule properties.
- `reset –` a value of true resets the timing information in the schedule, clearing: `repeat`, `units`, `startdate`, `starttime`, `enddate`, `endtime`, `days`, and `dates`.
- `repeats` – the interval at which the schedule repeats. NULL or 0 means it does not repeat. The value is a number followed by:
  - `day` or `d`
  - `days` or `dd`
  - `hour` or `h`
  - `hours` or `hh`
  - `minute` or `m`
  - `minutes` or `mm`
- `startdate` – the date on which the schedule becomes active. The time part is ignored.
- `enddate` – the date on which the schedule becomes inactive. The time part is ignored.
- `starttime` – the time of day when an active schedule begins to operate. The date part is ignored. The default is midnight (00:00).
- `endtime` – the time of day when an active schedule ceases to operate. The date part is ignored. The default is midnight (00:00).
- `days` – a colon-separated list of days. Full or abbreviated names in the locale of the server may be used.
- `dates` – a colon-separated list of dates in a month.

Returns

Returns 0 on success or an error code.

Examples

Example 1

Modifies the scheduled job “svr1_clean_stats,” updating the schedule to repeat every two hours between 10:00 and 16:00. The days or dates on which the scheduled job operates are unchanged.

```
sp_sjobmodify @name=’svr1_clean_stats’,
@option=’repeats=2hours,starttime=10:00,endtime=16:00’
```

Example 2

Modifies the job “backup_db5,” updating the default timeout to 120 minutes:

```
sp_sjobmodify @name=’jname=backup_db5’, @option=’default_timeout=120’
```

Example 3

Modifies the job named “orders_picked_report,” removing the shared property and changing the owner to mary. The caller requires `js_admin_role` to change the job owner.

```
sp_sjobmodify @name=’jname=orders_picked_report’,
@option=’jproperties=shared:false,owner=mary’
```

Usage

By default, the `@name` argument is the name or ID of a scheduled job. To specify the name or ID of a job or a schedule, prefix the `@name` argument with `jname` or `sname`, for example:

```
@name= ‘jname=run_update_stats’
```

```
@name= ‘sname=daily_schedule’
```

6.6 sp_sjobdrop

Deletes a job, a schedule, or a scheduled job.

By default, the `@name` argument is the name or ID of a scheduled job. Prefix the `@name` argument with `jname` or `sname` to specify the `@name` or ID for a job or a schedule.

Syntax

```
sp_sjobdrop @name=’...’, @option=’...’
```

Parameters

`<name>`

The ID of the scheduled job, job, or schedule to delete.

`<option>`

A list of options for the command:

- `all` – used with a job or schedule to specify that all scheduled jobs owned by the caller that are using the specified job or schedule should also be deleted. When used with a scheduled job, `all` specifies that the job and schedule associated with the scheduled job should also be deleted, provided that no other scheduled jobs reference them.

- `all_users` – allows a user with the `js_admin_role` to use `sp_sjobdrop` on scheduled jobs, jobs, and schedules that are owned by someone else.
- `show` – displays the jobs, schedules, and scheduled jobs that will be deleted by this call to `sp_sjobdrop`, before any deleting occurs.
- `force` – allows you to remove jobs from the database, even if they are currently running.

Returns

Returns 0 (zero) on success or an error code.

Examples

Example 1

Deletes the scheduled job called “svr1_clean_stats”:

```
sp_sjobdrop ’svr1_clean_stats’
```

Example 2

Deletes the schedule called “daily 01:00am”:

```
sp_sjobdrop ’sname=daily 01:00am’
```

Example 3

Deletes the job called “load_sales_data”:

```
sp_sjobdrop ’jname=load_sales_data’
```

Example 4

Deletes the job called “load_sales_data” and any scheduled jobs the caller owns that were using it.

```
sp_sjobdrop @name=’jname=load_sales_data’, @option=’all’
```

Usage

- An error occurs if you attempt to delete a scheduled job that is currently running, unless you use the `force` option.
- By default, `sp_sjobdrop` operates only on the scheduled job, job, or schedule specified by the `@name` argument.

For the general administration of jobs and schedules, the `all` and `all_users` options are provided. The following describes these options:

- When a scheduled job is specified and the `all` or `all_users` option is not used, only the specified scheduled job is deleted.
- When the `all` option is used for a scheduled job, the scheduled job is deleted. If there are no more references to the job, it is deleted, and if there are no more references to the schedule, it is deleted.
- When a job or a schedule is specified and the `all` or `all_users` option is not used, an error occurs if the job or schedule is referenced by one or more scheduled jobs.
- When the `all_users` option is used for a specified job or a schedule and the scheduled jobs that reference the job or schedule are owned by the caller, then the job or schedule and the referencing scheduled jobs are deleted.
- When the `all_users` option is used for a specified job or a schedule and the caller has the `js_admin_role`, then the job or schedule and all referencing scheduled jobs are deleted.

The `show` option lets you see which scheduled jobs, jobs, and schedules would be deleted by a call to `sp_dropjob`—without actually deleting them.

6.7 sp_sjobhelp

Generates a list or report about all the scheduled jobs, jobs, or running jobs that are visible to the caller, or about a restricted set based on the name and the option values.

Syntax

```
sp_sjobhelp @name='...', @option='...'
```

Parameters

`<name>`

The name or ID of the scheduled job, a job, or a `runid`. The `@name` argument restricts the scope to a single scheduled job, all scheduled jobs using a particular job, or a single run of a scheduled job. By default, `@name` is the name ID of a scheduled job. To specify the name, the ID of a job, or the `runid`, prefix the `@name` argument with `jname` or `runid`.

`<option>`

Specifies a comma-separated list of options, the `option_name=option_value pairs` that define the action to perform, and filters to restrict the information returned.

- `report` – format as a report.

- `list` – format as a list.
- `running` – produce a report of running jobs.
- `scheduled` – produce a report of scheduled jobs.
- `unscheduled` – produce a report of unscheduled jobs.
- `all_users` – include information about all users.
- `owner` – restrict to information owned by this user.
- `jobs` – provide information about jobs.
- `schedules` – provide information about schedules.
- `user` – restrict information for scheduled jobs to run on behalf of this user. Applies only to running jobs.

Returns

Returns 0 on success or an error code.

Examples

Example 1

Produces a list of all the scheduled jobs that are running:

```
sp_sjobhelp  @option=‘list,all_users,running’
```

Example 2

Produces a report showing all the jobs scheduled by the caller:

```
sp_sjobhelp @option=’report,scheduled’
```

Example 3

Produces a list of all the jobs and schedules that are not used in any scheduled jobs:

```
sp_sjobhelp  @option=‘list,all_users,unscheduled’
```

Usage

A report is in several sections, showing the scheduled jobs, jobs that do not have schedules, and the schedules.

6.8 sp_sjobcontrol

Defines an interface for controlling running jobs.

Syntax

```
sp_sjobcontrol @name='...', @option='...'
```

Parameters

<name>

The name or ID of the scheduled job, a job, or a `runid` to be controlled.

<option>

Specifies a comma-separated list of options and `option_name=option_value` pairs that define the action to perform.

- `terminate` – instances of the job with the given name, ID, or `runid` are terminated.
- `enable` – instances of the scheduled job with the given name or ID are enabled.
- `disable` – instances of the scheduled job with the given name or ID are disabled. Any currently running jobs are unaffected.
- `run_now` – schedules the job with the given name or ID to run immediately. However, there must be a nonconflicting scheduled job created for the corresponding job or `sp_sjobcontrol` returns an error.
- all_users – allows users with the role `js_admin_role` to administer scheduled jobs owned by other users.
- `start_js` – starts Job Scheduler.
- `stop_js` – stops Job Scheduler and terminates any running jobs.
- `stop_js_wait` – waits for running jobs to complete, and then stops Job Scheduler.
- `stop_js_timeout=<minutes>` – waits a specified number of minutes, and then stops Job Scheduler, terminating any running jobs.

Returns

Returns 1 (when started), 0 (when stopped), or an error code.

Examples

Example 1

Requests the JS Task and JS Agent to terminate the running job with `runid` 1532:

```
sp_sjobcontrol @name='runid=1532', @option='terminate'
```

Example 2

Marks the scheduled job called "svr1_load_sales_data" so that it can be run now. The existing schedule is unchanged.

```
sp_sjobcontrol @name='svr1_load_sales_data', @option='run_now'
```

Example 3

Disables all of the caller's scheduled jobs that use the job called "check_user_logins".

```
sp_sjobcontrol @name='jname=check_user_logins', @option='disable'
```

Usage

By default, `@name` is the name, the ID, or a scheduled job. To specify the name or ID of a job or as a `runid`, prefix the `@name` argument with `jname` or `runid`.

6.9 sp_sjobhistory

Lists or deletes job history and job output logs. You can use this procedure to look at the results of scheduled jobs that have completed. You can also use it during administration of the job history and job output logs.

Syntax

```
sp_sjobhistory @name='...', @option='...'
```

Parameters

`<name>`

The name or ID of the scheduled job, a job, or a `runid`.

You can use the `@name` argument to restrict the scope to the history of a single scheduled job, all scheduled jobs using a particular job, or a single run of a scheduled job. By default, `@name` is the name or ID or a scheduled job. To specify the name or ID of a job or as a `runid`, prefix the `@name` argument with `jname` or `runid`.

`<option>`

Specifies a comma-separated list of options, and `option_name=option_value` pairs that define the action to perform on a filtered set of the job history and job output.

- `list` – lists the job history fields for the jobs matching the filter conditions.
- `list_short` – lists a subset of the job history fields for the jobs matching the filter conditions.
- `drop` – deletes the job history entries and job output entries for the jobs matching the filter conditions.
- `list_output` – lists the job output for the jobs matching the filter conditions.
- `drop_output` – deletes the job output entries for the jobs matching the filter conditions. The corresponding job history entries are not deleted.
- `all_users` – includes all user jobs in the scope. The caller must have the `js_admin_role`.
- `user` – restricts the scope to jobs run by or on behalf of this user. If the user specified is not the caller's server user name, the caller must have the `js_admin_role`.
- `owner` – restricts the scope to jobs run using this server user name. If the owner specified is not the caller's server user name, the caller must have the `js_admin_role`.
- `age` – restricts the scope to jobs recorded earlier than this number of days ago.
- `minsize` – restricts the scope to jobs with more than minsize bytes of output.
- `force –` allows the history and output to be deleted for running jobs.

Returns

Returns 0 on success or an error code.

Examples

Example 1

Lists the concise history of the job called "orders_processed_report" that was run for the caller:

```
sp_sjobhistory @name='jname=orders_processed_report', @option='list_short'
```

Example 2

Deletes the history and job output for the scheduled job that ran with `runid` 12389:

```
sp_sjobhistory @name='runid=12389', @option='drop'
```

Example 3

Lists all the caller's job history that has job output greater than 10,000 bytes:

```
sp_sjobhistory @option='list,minsize=10000'
```

Example 4

Drops all the caller job history and job output that has job output greater than 20,000 bytes:

```
sp_sjobhistory @option='drop,minsize=20000'
```

Example 5

Lists the history of the job called “load_sales_data” that was run by any user:

```
sp_sjobhistory @name='jname=load_sales_data', @option='list,all_users'
```

Example 6

Lists the concise history of all the jobs run by the user called “mary”:

```
sp_sjobhistory @option='list_short,user=mary'
```

Usage

- To protect against accidental removal of all the job history or output logs, an error occurs when you call `sp_sjobhistory` with the `drop` option and no filter arguments; you must supply at least one filter argument.
- `sp_sjobhistory` does not delete job output log entries for jobs that are currently running. These entries are silently ignored during the deletion process, unless the `runid` argument specifies an entry for a running job. In this case, `sp_sjobhistory` returns an error.

Table 1: Output returned from the list argument

Column	Type
job_runid	int
job_name	JS_DESC
job_state	char(2)
job_end	datetime
job_user_code	int
job_os_code	int
job_user_req	SUSER_NAME
job_long_message	JS_LMSG

Column	Type
sjob_id	JS_NAME_ID
sched_name	JS_NAME_ID
job_start	datetime
job_exit_code	int
job_atat_error	int
job_user_run	SUSER_NAME
job_short_message	JS_SMSG
job_size	int

Table 2: Output returned from the list_short argument

Column	Type
job_runid	int
job_name	JS_DESC
job_state	char(2)
job_end	datetime
job_user_code	int
job_size	int
sjob_id	JS_NAME_ID
sched_name	JS_NAME_ID
job_start	datetime
job_exit_code	int
job_atat_error	int

Table 3: Output returned from the list_output argument

Column	Type
job_runid	int
job_size	int
job_name	JS_DESC

Column	Type
job_output	JS_OUTPUT

7 Troubleshooting

Solve problems that occur in Job Scheduler.

7.1 Logging Error Messages

When you configure a new SAP ASE server, the installation program automatically sets the error log location. Then, SAP ASE writes start-up information to the local error log file each time it starts. When you install Job Scheduler, the JS Agent creates its own log file in the same directory.

See the configuration documentation for your platform to learn the default location and file name of the error log.

Many error messages from SAP ASE server only go to the user's terminal. However, fatal error messages (severity levels 19 and above), kernel error messages, and informational messages from SAP ASE are recorded in the error log file.

SAP ASE keeps the error log file open until you stop the server process. If you need to reduce the size of the error log by deleting old messages, stop the server process before you do so.

> i Note
>
> On some platforms (such as Windows), SAP ASE also records error messages in the operating system event log. See the SAP ASE installation and configuration guides for additional information about error logs.

7.2 A Job Fails to Run at the Scheduled Time

When you create a schedule, or schedule a job, make sure the time you are using is the time on the server where Job Scheduler is installed.

If you schedule the job using the local time on your machine, it is not necessarily the same time as the time on the Job Scheduler installation server.

7.3 A Scheduled Job Created from a Template Fails

If a scheduled job created from a template fails to run, check the job history for error messages. If a job failed to run, the reason will be reported in the job history.

If you have not installed the stored procedures used by the templates on the target server where your job will run, you see an error message similar to:

```
Procedure <'dump_databases'> not found.
Specify owner.objectname or use sp_help to check whether the object exists
(sp_help may produce lots of output).
```

7.4 A Job with Multiple Calls to sp_sjobcmd Fails

When you create a job with several calls to `sp_sjobcmd`, you must have a new line after you execute the call `go`.

7.5 A Stored Procedure Fails

All Job Scheduler jobs start in the `master` database. If your job is invokes a stored procedure that is on another database, you may have to prefix the stored procedure with the name of the database on which it resides.

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.
About the icons:

- Links with the icon ➦ : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
    - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
    - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon ➦: You are leaving the documentation for that particular SAP product or service and are entering a SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Gender-Related Language

We try not to use gender-specific word forms and formulations. As appropriate for context and readability, SAP may use masculine word forms to refer to all genders.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

www.sap.com/contactsap

THE BEST RUN SAP