SGE

This chapter provides information about the common administrative tasks. Please consult SGE documentation for in-depth info.

Docs: http://docs.oracle.com/cd/E19080-01/n1.grid.eng6

Note: Most administrative tasks can only be performed on the administrative hosts. You can get a list of the administrative hosts by running

```
# qconf  -sh
```

Queue management

Note: operations on a queue can be performed by either the administrative users or the queue owner(s)

To get the list of the administrative users(managers)

```
$ qconf -sm
```

To get the list of the queue owners

```
$ qconf -sq queueName | grep owner_list
```

Listing queues

Display a list of all currently configured queues.

```
$ qconf -sql
```

Displaying queue contents

All queues:

```
$ qstat -u '*'
```

Specific queue:

```
$ qstat -u '*' -q queueName
```

Displaying queue usage

To display the queue usage:

```
$ qstat -g c
```

To display parallel jobs information (status, slave and master tasks and running queue instance)

```
$ qstat -g
```

To display a list of complex values

```
$ qstat -sc
```

Suspending queue

```
$ qmod -sq queueName
```

Resuming queue

```
$ qmod -usq queueName
```

Modifying queue

All queue modifications are performed by running

```
$ qconf -mq queueName
```

which opens an editor (defined by the EDITOR environment variable) with the queue configuration text

Installing a new host

After installing a new host by means of OSCAR run the following script:

```
# $SGE_ROOT/scripts/add_newhost
```

The script may prompt you to configure the new host as an administration host if you haven't already done that.

Adding/removing a host

There are two ways of adding a host to the queue:

1. Add the host to a hostgroup. This is the preferred method.

   a. Add a host to the existing host group:

   ```
   $ qconf -mhgrp @hostGroupName
   ```

   OR

   b. Create and configure a new host group:

   ```
   $ qconf -ahgrp @hostGroupName
   ```

   And then add the new hosts group to the queue's *hostgroup* by running `qconf -mq queueName`

   Note: You can run `qconf -shgrpl` to get a list of the existing queues

2. Add the host directly to the queue.

   This can be performed by `qconf -mq queueName`, as in 1.b. above, with the only difference that the host name is not prepended by @, as opposed to the group name.

Adding/removing a user

There are two ways of allowing user access to the queue:

1. Add the users to the users group. This is the preferred method.

   ```
   $ qconf -au userName groupName
   ```

   And then allow the group access to the queue by adding the group name to the queue's *user_lists* (space separated list) by running `qconf -mq queueName`

Note: If the group doesn't exist – it will be automatically created.

2. Add the user directly to the queue's *user_lists.*

Parallel Environments

- List configured parallel environments

```
$ qconf -spl
```

- Modify parallel environments

```
$ qconf -mp
```

- Important configuration parameters:

`slots` a maximum number of allowed slots for the particular pe

`start_proc_args` start script and it's arguments.

`end_proc_args` stop script and it's arguments.

Log files

The log files for the queue master and shadows are located under `$SGE_ROOT/default/spool/qmaster` and are named as following:

- `messages` - sge_qmaster log file
- `messages_shadowd.`***`hostname`*** - sge_shadowd logs for host ***hostname***

The log files for execution hosts are located under `$SGE_ROOT/default/spool/`***`hostname`***`/messages,` where ***hostname*** is the execution host short name (non-FQDN)

Job Management

Establishing an interactive session

```
$ qlogin -l queueName
```

Submitting job to a queue

```
$ qsub -l queueName [options] /path/to/job/script [job_script_parameters]
```

Commonly used options:

`-r y` make the job "rerunable" if it aborts without leaving a consistent exit state (typically when the node running a job crashes or the job has been suspended on a proprietary host)

`-V` import environment variables into execution context

`-cwd` use $CWD as the excution environment's initial $CWD

`-b y` use if executing a binary instead of the job script (not recommended)

`-l queueName,`**`mem=32`**

Submit to a queue queueName, limit to machines with 32GB RAM

`-M email` Email(s) of the user(s) to be notified

`-R y` Reserve slots for the job

`-m [bes]` Notify on job start/end/suspend

Example:

```
[user1@rf1 ~]$ qsub -l all.q -V -cwd /usr/local/scripts/sge_dytran 1 18
```

Displaying a job status

For existing job

```
$ qstat -j jobID
```

For a job that is no longer running

```
$ qacct -j jobID
```

Suspending a job

```
$ qmod -sj jobID
```

Resuming a job

```
$ qmod -usj jobID
```

Canceling a job

To delete a specific job from the queue:

```
$ qdel jobID
```

If you've performed the command above and you still see the job in the queue after a while – try forcefully deleting the job:

```
$ qdel -f jobID
```

Tip1: If you wish to perform a bulk operation on the jobs by a specific parameter you may use the following one-liner:

```
qstat | grep string | awk '{print $1}' | xargs qdel -f
```

For example, the following command will forcefully delete all jobs owned by user *user1*

```
$ qstat -u user1 | awk '{print $1}' | xargs qdel -f
```

Or if you wish to forcefully delete all jobs named "my_job.sh", run by any user on the system, you may run the following:

```
$ qstat -u '*' | grep 'my_job.sh' | awk '{print $1} | xargs qdel -f
```

Tip2: If you wish to get a list of all jobs run by a specific group of users (AKA user_list) you may use the following one-liner:

```
qstat -u "`qconf -su user_list | tail +5 | sed -e 's/^entries//'`"
```

where ***`user_list`*** is an users group name, as per output of `qconf -sul`

Output Files

Any running job creates output and errors files in the jobs in the either the current directory. By default, the files are prefixed by the job name, but that can be changed by supplying -o and -e parameters to qsub.

jobname.**o***jobid*	STDOUT for a job *jobname* and job number *jobid*
jobname.**e***jobid*	STDERR for a job *jobname* and job number *jobid*
jobname.**po***jobid*	STDOUT for a parallel job *jobname* and job number *jobid*
jobname.**pe***jobid*	STDERR for a parallel job *jobname* and job number *jobid*