Maxwell useful commands

Overview

Command	Environment	man page	Explained	Example
my-resources			Show major resources and their availability for your account	
my-partitions			Show all maxwell partitons and the access for your account	
my-licenses			Show licenses for major commercial products. Lists all the licenses you are currently using	my-licenses -p matlab
my-quota			Show quota of maxwell home directory	
xwhich			find applications and print required setup	xwhich Avizo
(u)mount-desycloud (u)mount-winhome (u)mount-wingroup			access remote storage, like desycloud or windows shares	
Job controls				
sbatch		man sbatch	Submit a batch job to the cluster	sbatch -p all --constraint=P100 --time=1-12:00:00
salloc		man salloc	Submit a request for an interactive job to the cluster	salloc --partition=all --nodes=1
srun		man srun	Run interactive job	srun -p all --pty -t 0-06:00 matlab_R2018a
scancel		man scancel	signal jobs or job steps	scancel -j 12345
scontrol		man scontrol	view and modify Slurm configuration and state	
Job & Cluster information				
sview		man sview	GUI for SLURM to show current status of the cluster	
sinfo		man sinfo	view information about Slurm nodes and partitions	
squeue		man squeue	view information about jobs in scheduling queue	
sacct		man sacct	Accounting information for jobs invoked with Slurm	
sstat		man sstat	Status information for running jobs invoked with Slurm	
slurm	module load maxwell tools			
savail	module load maxwell tools	savail -h	Show the real availability of nodes	savail -p maxwell
max-limits			Show limits of partitions	max-limits -p jhub -a

sbatch

Create a batch script my-script.sh like the following and submit with sbatch my-script.sh:

```
#!/bin/bash
#SBATCH --time      0-00:01:00
#SBATCH --nodes     1
#SBATCH --partition maxwell
#SBATCH --job-name  slurm-01
export LD_PRELOAD=""                # useful on max-display nodes, harmless on others
source /etc/profile.d/modules.sh    # make the module command is available
env                                 # just list the environment
```

That's the core information would you probably should also keep. Note: never add a #SBATCH after a regular command. It will be ignored like any other comment.

A simple example for a mathematica:

```
#!/bin/bash
#SBATCH --time      0-00:01:00
#SBATCH --nodes     1
#SBATCH --partition all
#SBATCH --job-name  mathematica
export LD_PRELOAD=""                # useful on max-display nodes, harmless on others
source /etc/profile.d/modules.sh    # make the module command is available
module load mathematica
export nprocs=$((`/usr/bin/nproc` / 2))   # we have hyperthreading enabled. nprocs==number of physical cores
math -noprompt -run '<<math-trivial.m'

# sample math-trivial.m:
tmp = Environment["nprocs"]
nprocs = FromDigits[tmp]
LaunchKernels[nprocs]
Do[Pause[1];f[i],{i,nprocs}] // AbsoluteTiming        >> "math-trivial.out"
ParallelDo[Pause[1];f[i],{i,nprocs}] // AbsoluteTiming  >>> "math-trivial.out"
Quit[]
```

It might be important to define the type of hardware to use, in particular for multi-host or GPU jobs

```
#!/bin/bash
#SBATCH --time      0-12:00:00
#SBATCH --nodes     8
#SBATCH --partition all
#SBATCH --job-name  constrained
export LD_PRELOAD=""                # useful on max-display nodes, harmless on others
source /etc/profile.d/modules.sh    # make the module command is available
mpirun ...

# submit
sbatch --constraint=INTEL         # you don't want to mix AMD and INTEL nodes in mpi jobs
sbatch --constraint="INTEL&GPU"   # only use INTEL nodes with GPU. Currently specifying GPUs would be
sufficient, there are no AMD+GUI at this time
sbatch --constraint="Gold-6140"   # explicitly fix the CPU type
sbatch --constraint="768G|512G"   # only use nodes with 512 or 768G
```

salloc

salloc uses the same syntax as sbatch.

```
# request one node with a P100 GPU for 8hours in the all partition:
salloc --nodes=1 --partition=all --constraint=P100 --time=08:00:00

# start an interactive graphical matlab session on the allocated host.
ssh -t -Y $SLURM_JOB_NODELIST matlab_R2018a

# the allocation won't disappear when being idle. You have to terminate the session
exit
```

scancel

```
scancel 1234                     # cancel job 1234
scancel -u $USER                 # cancel all my jobs
scancel -u $USER -t PENDING      # cancel all my pending jobs
scancel --name myjob             # cancel a named job
scancel 1234_3                   # cancel an indexed job in a job array
```

sinfo

```
sinfo                                                                  # basic list of partitions
sinfo -N -p all                                                        # list all nodes and state in all
partition
sinfo -N -p petra4 -o "%10P %.6D %8c %8L %12l %8m %30f %N"             # list all nodes with limits and
features in petra4 partition
```

squeue

```
squeue                              # show all jobs
squeue -u $USER                     # show all jobs of user
squeue -u $USER -p upex -t PENDING  # all pending jobs of user in upex partition
```

sacct

```
sacct -j 1628456                                                       # accounting information for jobid
sacct -u $USER                                                         # todays jobs

# get detailed information about all my jobs since 2019-01-01 and grep for all that FAILED:
sacct -u $USER --format="partition,jobid,state,start,end,nodeList,CPUTime,MaxRSS" --starttime 2019-01-01 |
grep FAILED
```

slurm

```
module load maxwell tools
slurm

#Show or watch job queue:
 slurm [watch] queue     # show own jobs
 slurm [watch] q <user>  # show user's jobs
 slurm [watch] quick     # show quick overview of own jobs
 slurm [watch] shorter   # sort and compact entire queue by job size
 slurm [watch] short     # sort and compact entire queue by priority
 slurm [watch] full      # show everything
 slurm [w] [q|qq|ss|s|f] shorthands for above!

 slurm qos               # show job service classes
 slurm top [queue|all]   # show summary of active users

#Show detailed information about jobs:
 slurm prio [all|short]  # how priority components
 slurm j|job <jobid>     # how everything else
 slurm steps <jobid>     # show memory usage of running srun job steps

#Show usage and fair-share values from accounting database:
 slurm h|history <time>  # show jobs finished since, e.g. "1day" (default)
 slurm shares

#Show nodes and resources in the cluster:
 slurm p|partitions      # all partitions
 slurm n|nodes           # all cluster nodes
 slurm c|cpus            # total cpu cores in use
 slurm cpus <partition>  # cores available to partition, allocated and free
 slurm cpus jobs         # cores/memory reserved by running jobs
 slurm cpus queue        # cores/memory required by pending jobs
 slurm features          # List features and GRES
 slurm brief_features    # List features with node counts
 slurm matrix_features   # List possible combinations of features with node counts
```