

SLURM (basics)

Claudia Blaas-Schenner

VSC Research Center, TU Wien

VSC Training Course: Introduction to Working on the VSC Clusters, January 16, 2024

Quickstart – demo

VSC-4 script: examples/05_submitting_batch_jobs/job_vsc4.sh

```
#!/bin/bash # VSC-4

#SBATCH -J test
#SBATCH -N 1
#SBATCH --qos=skylake_0096 # use a qos
#SBATCH --partition=skylake_0096 # use partition (qos)
#SBATCH --tasks-per-node=48 # SLURM_NTASKS_PER_NODE

module purge # recommended
# module load <modules>

echo
echo 'Hello from node: '$HOSTNAME
echo 'Number of nodes: '$SLURM_JOB_NUM_NODES
echo 'Tasks per node: '$SLURM_TASKS_PER_NODE
echo 'Partition used: '$SLURM_JOB_PARTITION
echo 'QOS used: '$SLURM_JOB_QOS
echo 'Using the nodes: '$SLURM_JOB_NODELIST
echo
sleep 30 # <do_my_work>
```

```
# submission:
sbatch job.sh

# check what is going on:
queue -u $USER ! sq

# output:
slurm-<job_id>.out

# cancel jobs:
scancel <job_id>
scancel <job_name>
scancel -u $USER
```

Quickstart – demo

VSC-5 script: examples/05_submitting_batch_jobs/job_vsc5.sh

```
#!/bin/bash # VSC-5

#SBATCH -J test
#SBATCH -N 1
#SBATCH --qos=zen3_0512 # use a qos
#SBATCH --partition=zen3_0512 # use partition (qos)
#SBATCH --tasks-per-node=128 # SLURM_NTASKS_PER_NODE

module purge # recommended
# module load <modules>

echo
echo 'Hello from node: '$HOSTNAME
echo 'Number of nodes: '$SLURM_JOB_NUM_NODES
echo 'Tasks per node: '$SLURM_TASKS_PER_NODE
echo 'Partition used: '$SLURM_JOB_PARTITION
echo 'QOS used: '$SLURM_JOB_QOS
echo 'Using the nodes: '$SLURM_JOB_NODELIST
echo
sleep 30 # <do_my_work>
```

```
# submission:
sbatch job.sh

# check what is going on:
squeue -u $USER ! sq

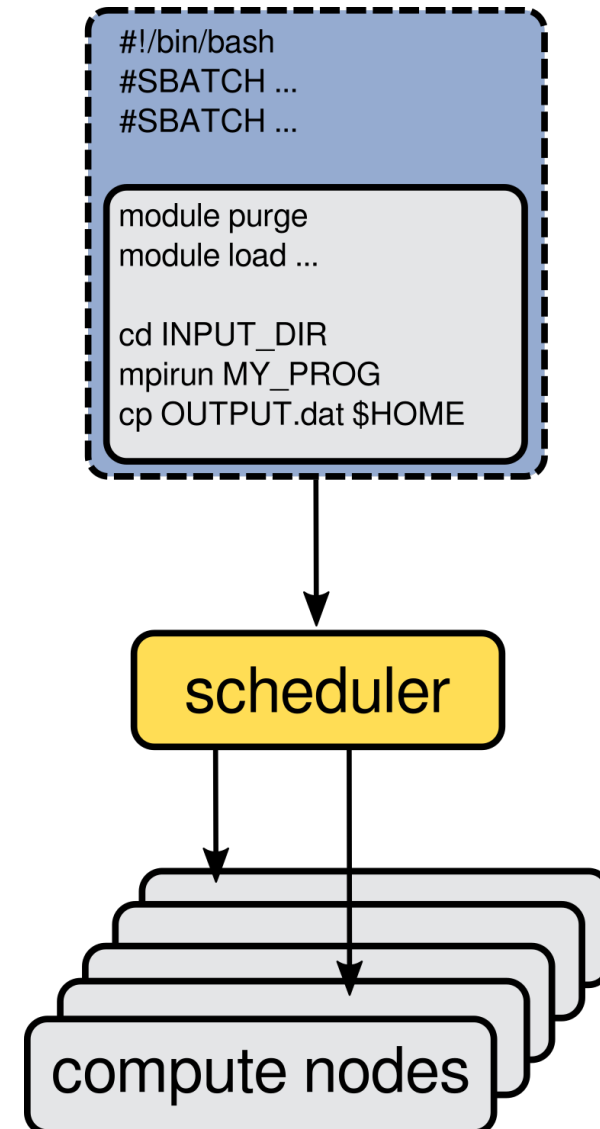
# output:
slurm-<job_id>.out

# cancel jobs:
scancel <job_id>
scancel <job_name>
scancel -u $USER
```

SLURM – basic concepts

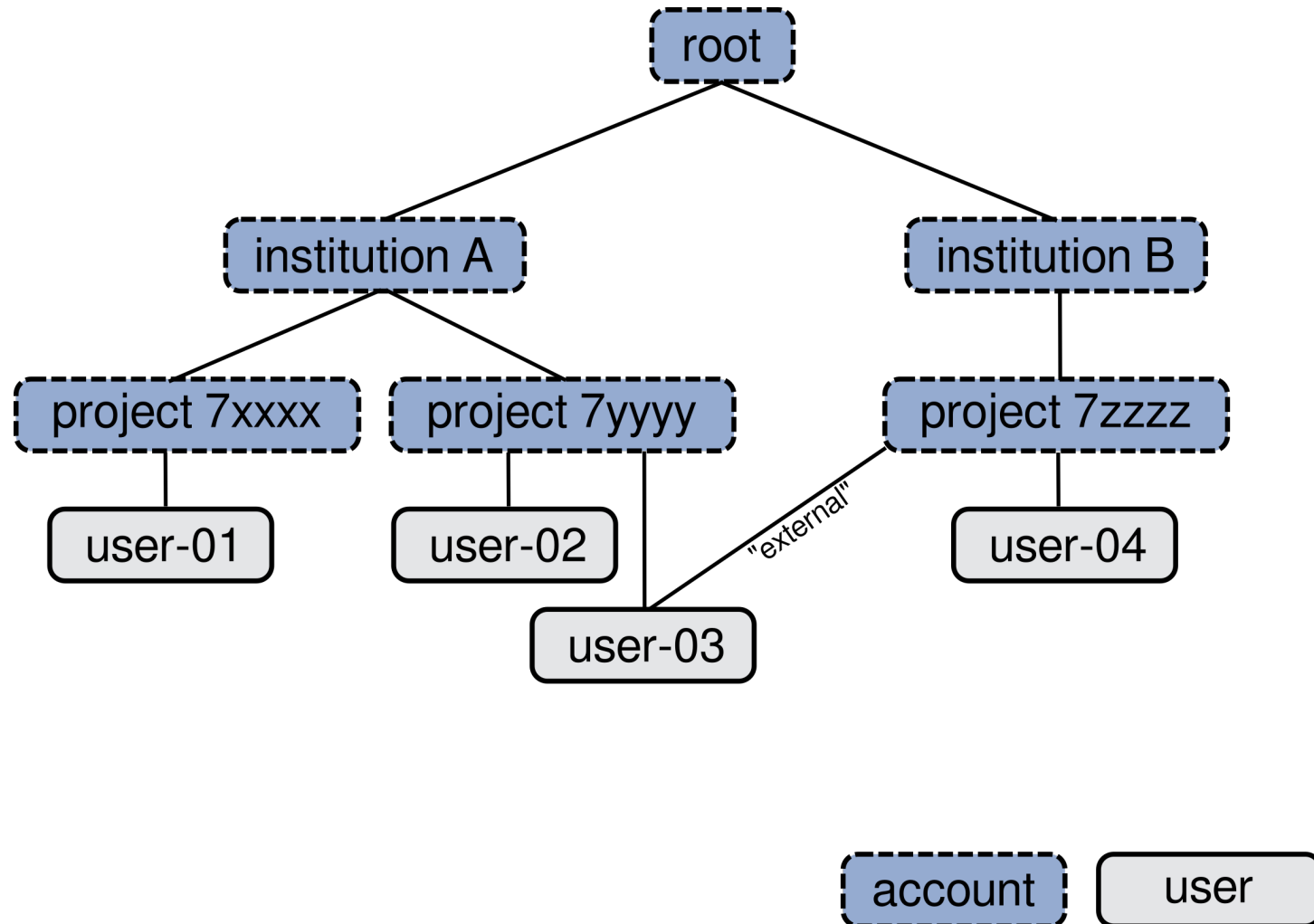
Queueing system

- job/batch script
 - **shell script – #!**
that does everything needed to run your calculation
 - **independent of queueing system**
 - **use simple scripts**
max 50 lines, i.e., put complicated logic elsewhere
 - **load modules from scratch**
purge, then load
- tell scheduler where/how to run jobs
 - *number of nodes (or cores)*
 - *nodetype (i.e., partition & qos)*
 - ...
- scheduler manages job allocation to compute nodes



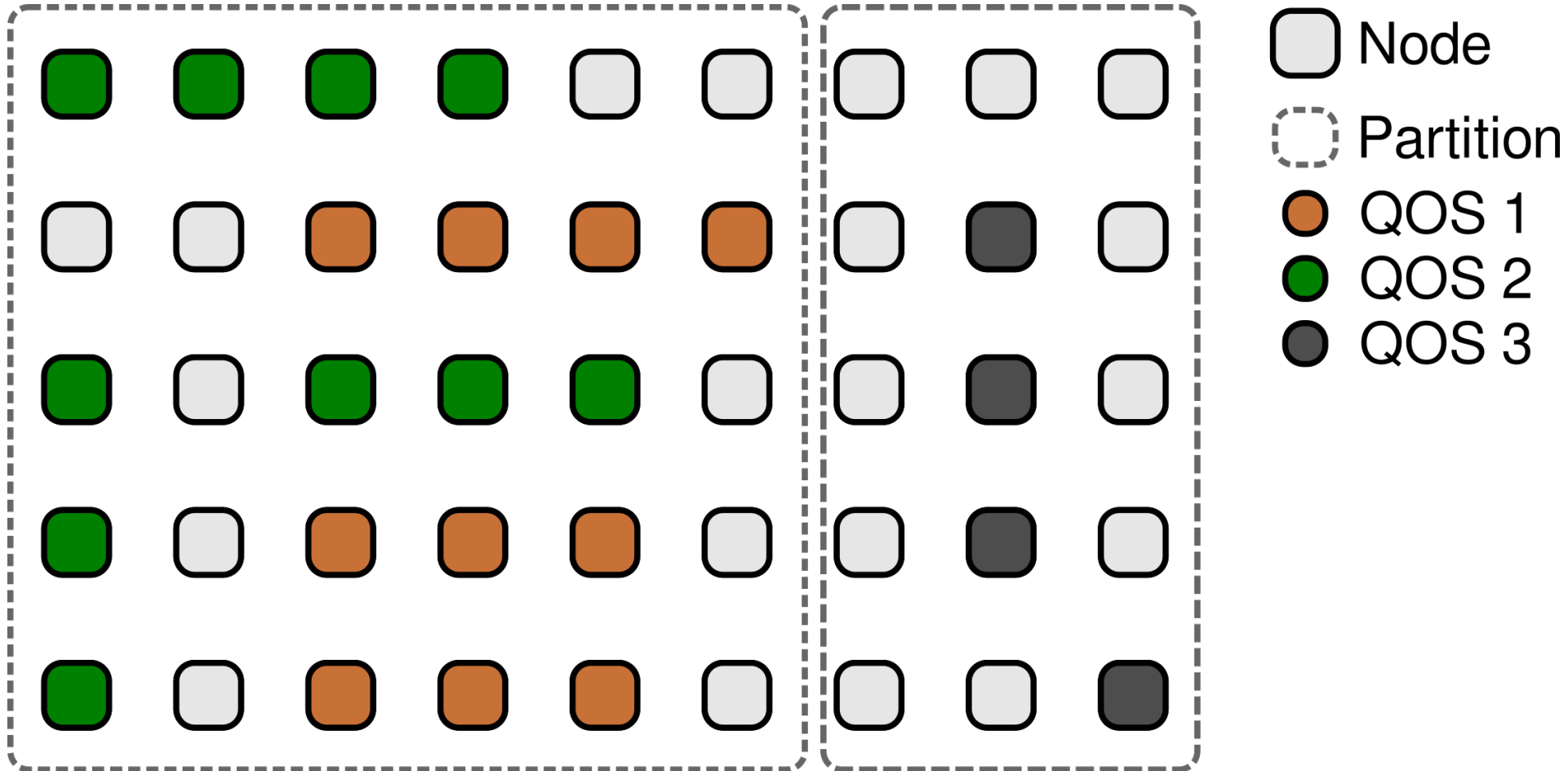
SLURM – account and user

#SBATCH --account=<account> ! use a specific account/project p7.... (other than default)



SLURM – partition and QOS (quality of service)

```
#SBATCH --qos=<qos>           ! specify quality of service   ! always provide:
#SBATCH --partition=<partition> ! specify type of hardware     ! qos & partition
```



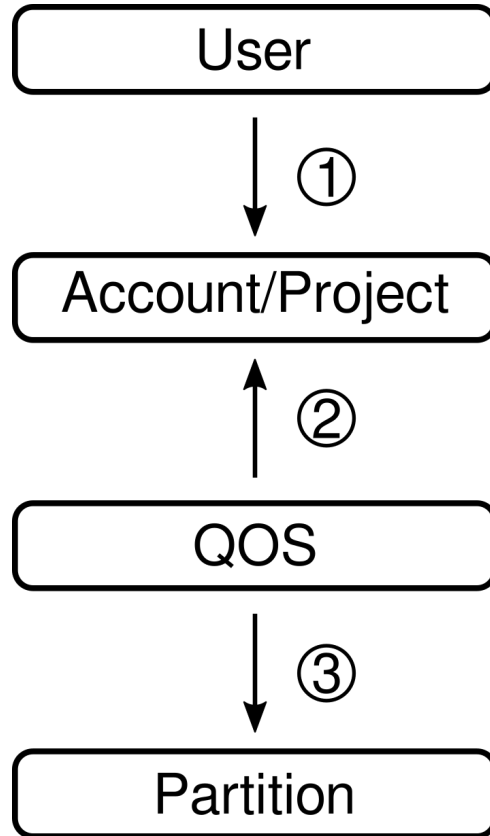
VSC hardware – display information

- Display information about partitions and their nodes:

```
VSC-4> sinfo  
VSC-4> sinfo -o %P  
VSC-4> scontrol show partition skylake_0096  
VSC-4> scontrol show node n4901-001
```

```
VSC-5> sinfo  
VSC-5> sinfo -o %P  
VSC-5> scontrol show partition zen3_0512  
VSC-5> scontrol show node n3501-001
```


QOS – account/project assignment



→
"is assigned to"

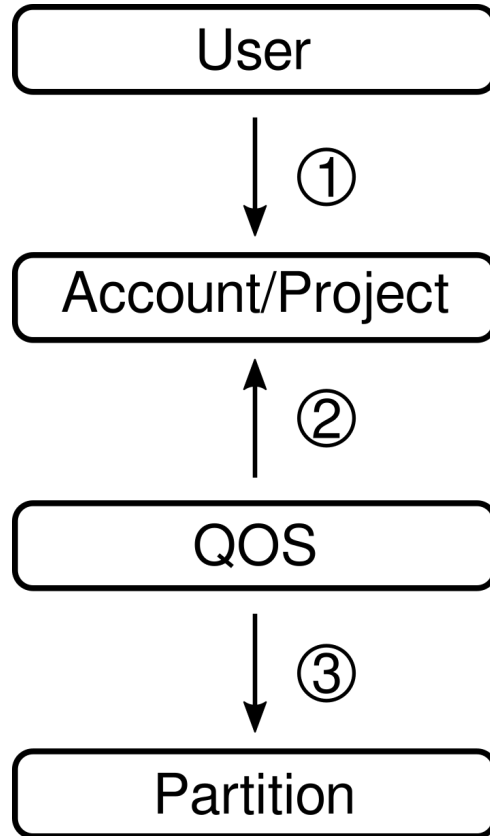
```

VSC-4 login: # VSC-4
=====
Your jobs can run with the following account(s) and ...(QOS):

default_account:          p70824
  account:                p70824

default_qos:              skylake_0096
  qos:                    skylake_0096
                           skylake_0096_devel
                           skylake_0096_jupyter
                           skylake_0384
                           skylake_0768
  
```

QOS – account/project assignment



→
"is assigned to"

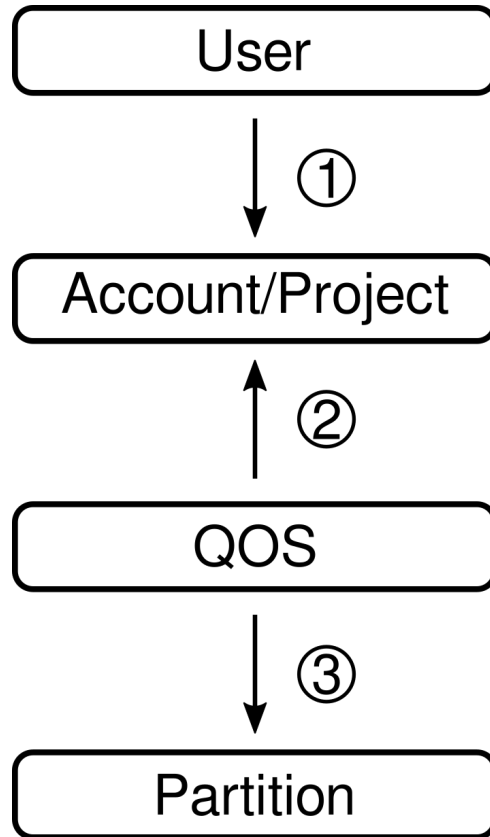
```

VSC-5 login: # VSC-5
=====
Your jobs can run with the following account(s) and ...(QOS):

default_account:          p70824
   account:              p70824

   default_qos:          zen3_0512
   qos:                  cascadelake_0384
                        zen2_0256_a40x2
                        zen2_0256_a40x2_jupyter
                        zen3_0512
                        zen3_0512_a100x2
                        zen3_0512_a100x2_jupyter
                        zen3_0512_devel
                        zen3_0512_jupyter
                        zen3_1024
                        zen3_2048
  
```

QOS – partition assignment



→
"is assigned to"

```

VSC-4> sqos # VSC-4

=====
      qos_name  type  ...  walltime  priority  ...
=====
      skylake_0096  cpu  ...  3-00:00:00  1000  ...
      skylake_0096_devel  cpu  ...  00:10:00  5000000  ...
      skylake_0096_jupyter  cpu  ...  3-00:00:00  1000  ...
      skylake_0384  cpu  ...  3-00:00:00  1000  ...
      skylake_0768  cpu  ...  3-00:00:00  1000  ...
  
```

```

VSC-4> cat /etc/motd # VSC-4

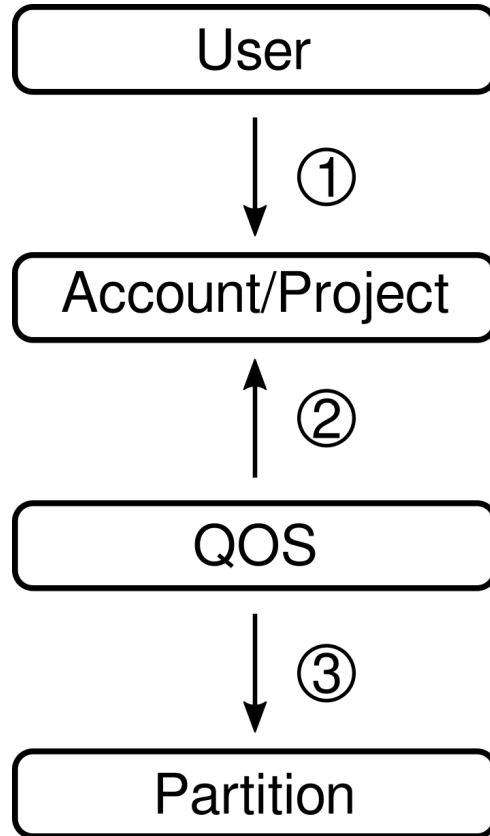
Available QoS:

partition          QoS
-----
skylake_0096       skylake_0096,skylake_0096_devel
skylake_0384       skylake_0384
skylake_0768       skylake_0768
  
```

```

# Specification in job script:
#SBATCH --account=<account>      ! specify account/project p7....
#SBATCH --qos=<qos>              ! specify quality of service      ! always provide:
#SBATCH --partition=<partition> ! specify type of hardware      ! qos & partition
  
```

QOS – partition assignment



```
VSC-5> sqos # VSC-5
      qos_name  type  ...  walltime  priority  ...
=====
. . . . .
```

```
VSC-5> cat /etc/motd # VSC-5
Available QoS:
partition          QoS
-----
cascadelake_0384   cascadelake_0384
zen2_0256_a40x2    zen2_0256_a40x2
zen3_0512_a100x2   zen3_0512_a100x2
zen3_0512           zen3_0512,zen3_0512_devel
zen3_1024           zen3_1024
zen3_2048           zen3_2048
```

→
"is assigned to"

```
# Specification in job script:
#SBATCH --account=<account>      ! specify account/project p7....
#SBATCH --qos=<qos>              ! specify quality of service    ! always provide:
#SBATCH --partition=<partition> ! specify type of hardware      ! qos & partition
```

Sample batch job

```
#!/bin/bash

#SBATCH -J <jobname>
#SBATCH -N <number_of_nodes>

#SBATCH --account=<account>      # optional, if omitted use default project
#SBATCH --qos=<qos>              # use a qos
#SBATCH --partition=<partition>  # use partition that fits to the qos
#SBATCH --tasks-per-node=<nn>    # SLURM_NTASKS_PER_NODE (mpi procs / node)

module purge                    # recommended to be done in all jobs !!!!!
# module load <modules>        # load only modules actually needed by job

echo 'Hello from node: '$HOSTNAME
echo 'Number of nodes: '$SLURM_JOB_NUM_NODES
echo 'Tasks per node: '$SLURM_TASKS_PER_NODE
echo 'Partition used: '$SLURM_JOB_PARTITION
echo 'QOS used: '$SLURM_JOB_QOS
echo 'Using the nodes: '$SLURM_JOB_NODELIST
# <do_my_work>
```

- must be a shell script – first line – #!
- '#SBATCH' for marking SLURM parameters (for omitted lines corresponding defaults are used)
- environment variables are set by SLURM for use within the script (e.g. SLURM_JOB_NUM_NODES)

Single (few) core(s) jobs – shared compute nodes

```
#!/bin/bash # VSC-4

#SBATCH -J single
#SBATCH -n 1 # specify number of cores
#SBATCH --mem=2G # memory limit in Gigabytes

###SBATCH --account=<account> # optional, if omitted use default project
#SBATCH --qos=skylake_0096 # use a qos
#SBATCH --partition=skylake_0096 # use partition that fits to the qos

module purge # recommended to be done in all jobs !!!!!
# module load <modules> # load only modules actually needed by job

echo 'Hello from node: '$HOSTNAME
echo 'Number of nodes: '$SLURM_JOB_NUM_NODES
echo 'Tasks per node: '$SLURM_TASKS_PER_NODE
echo 'Partition used: '$SLURM_JOB_PARTITION
echo 'Using the nodes: '$SLURM_JOB_NODELIST
# <do_my_work>
```

Single (few) core(s) jobs – shared compute nodes

```
#!/bin/bash # VSC-5

#SBATCH -J single
#SBATCH -n 1 # specify number of cores
#SBATCH --mem=4G # memory limit in Gigabytes

###SBATCH --account=<account> # optional, if omitted use default project
#SBATCH --qos=zen3_0512 # use a qos
#SBATCH --partition=zen3_0512 # use partition that fits to the qos

module purge # recommended to be done in all jobs !!!!!
# module load <modules> # load only modules actually needed by job

echo 'Hello from node: '$HOSTNAME
echo 'Number of nodes: '$SLURM_JOB_NUM_NODES
echo 'Tasks per node: '$SLURM_TASKS_PER_NODE
echo 'Partition used: '$SLURM_JOB_PARTITION
echo 'Using the nodes: '$SLURM_JOB_NODELIST
# <do_my_work>
```

Job submission

```
# submission:

sbatch job.sh

sbatch <SLURM_PARAMETERS> job.sh <JOB_PARAMETERS>
parameters are specified as in job script, command-line overrides job-script

# check what is going on:

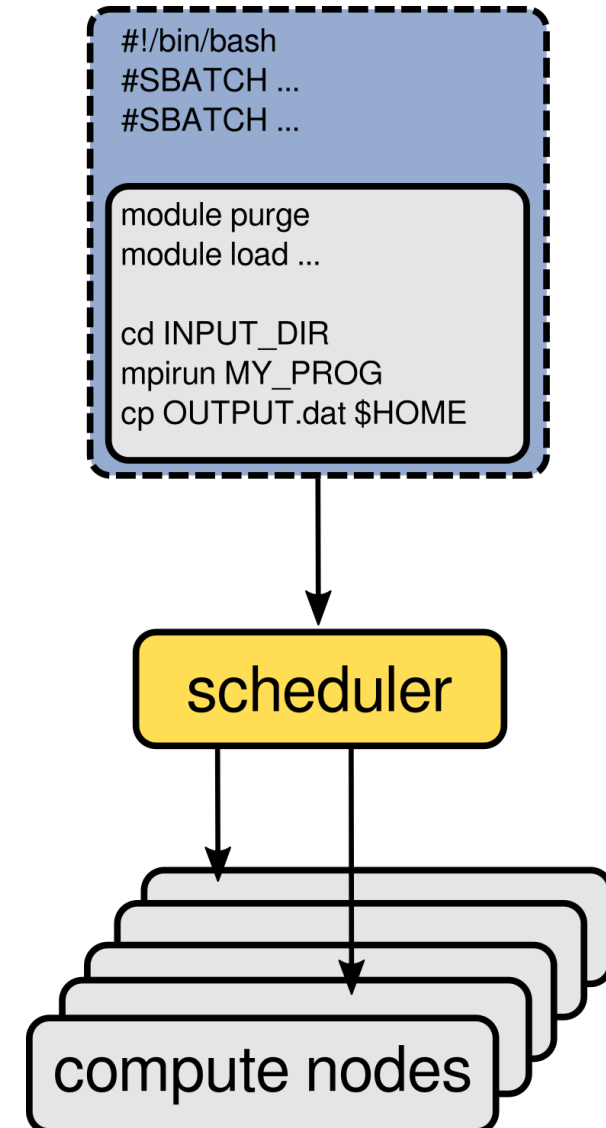
squeue -u $USER          ! alias sq='squeue -u $USER'

# output:

slurm-<job_id>.out

# cancel jobs:

scancel <job_id>
scancel <job_name>
scancel -u $USER
```



Exercises (1/2)

- switch on reservation to avoid queuing time:

```
VSC-4/5> source ~training/start_using_training_vsc4/5      ! during course only
```

- try all commands explained in SLURM – basics on both VSC-4 and VSC-5:

examples/05_submitting_batch_jobs/job_vsc4.sh # job_vsc5.sh

```
sbatch job_vsc4.sh          #          sbatch job_vsc5.sh
queue -u $USER ! sq
scancel <job_id>
! output in: slurm-<job_id>.out
```

```
sqos
```

```
sinfo
sinfo -o %P
scontrol show partition ...
scontrol show node ...
```

Exercises (2/2)

- switch off the reservation for the next task:

```
VSC-4/5> source ~training/switch_2_default      ! during course only
```

- write & submit to the development queue:

```
hostname      (solution = job_devel_vsc4.sh)  
free          (solution = job_devel_vsc5.sh)
```

- switch on the reservation again:

```
VSC-4> source ~training/start_using_training_vsc4/5      ! during course only
```

```
submit a single core job:  
job_single_core_vsc4.sh  
job_single_core_vsc5.sh
```

```
look at & submit:  
(DOES NOT WORK ON JH)  
job_mpi_vsc4.sh  
job_mpi_vsc5.sh
```

```
write & submit a matlab job:  
echo "2+2" | matlab  
(solution = job_matlab_vsc4.sh)  
(solution = job_matlab_vsc5.sh)
```

Enjoy 😊 ➡ Working on the VSC Clusters

Thank you for your attention!

Please provide an anonymous feedback (at the end of the course)

➡ <https://events.vsc.ac.at/event/112/surveys/107>