Taming the Cluster: A Researcher's Guide to Slurm

Research Computing
UNIVERSITY OF COLORADO BOULDER

Meet the User Support Team

Layla
Freeborn

Brandon
Reyes

Andy
Monaghan

Michael
Schneider

John
Reiland

Dylan
Gottlieb

Mohal
Khandelwal

Ragan
Lee

View The Slides

https://github.com/ResearchComputing/rmacc_2025

What is Slurm?

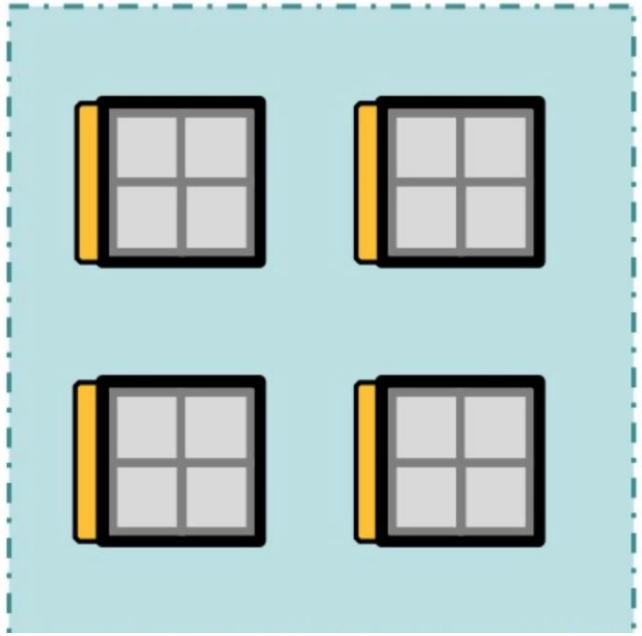
Slurm is an open source cluster management and job scheduling system for Linux clusters.

What's in the name? Simple **L**inux **U**tility for **R**esource **M**anagement

1. Keeps track of available resources on the cluster
2. Collects users' resources requests for jobs
3. Assign priorities to jobs
4. Run jobs on assigned compute nodes

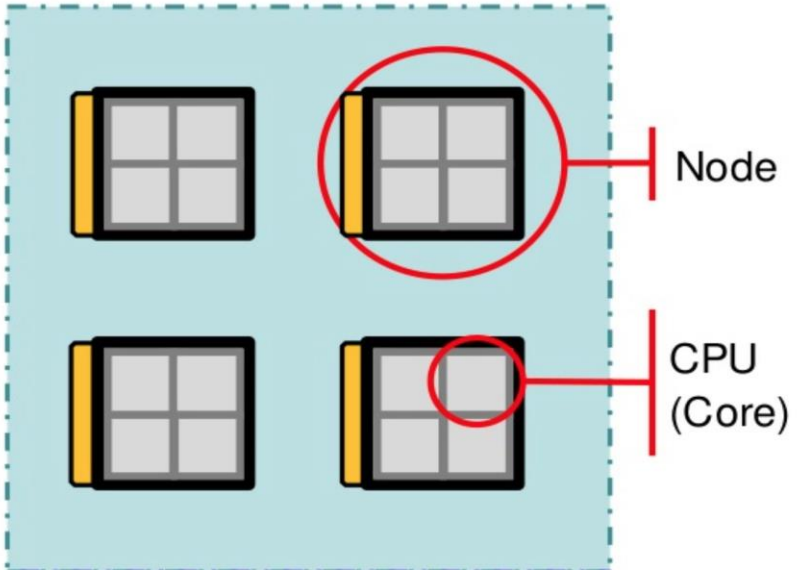
SLURM VOCABULARY:

Nodes | Partitions | Jobs



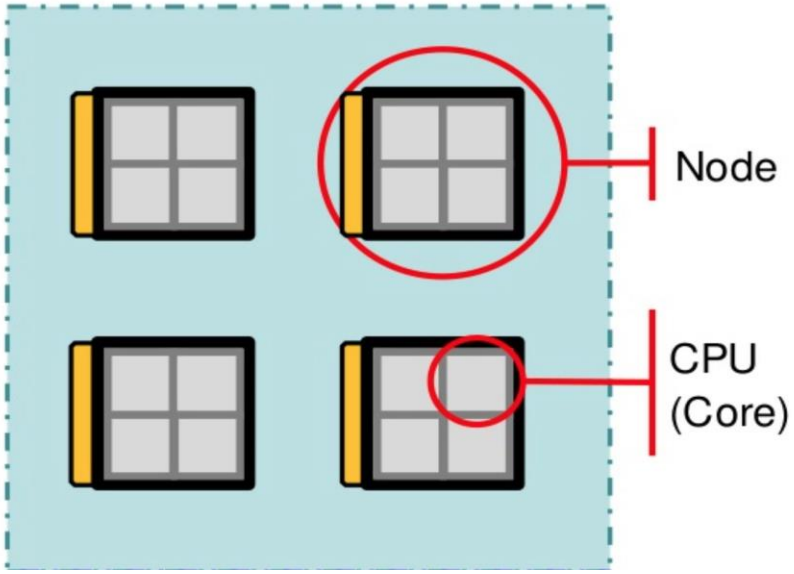
SLURM VOCABULARY:

Nodes | Partitions | Jobs

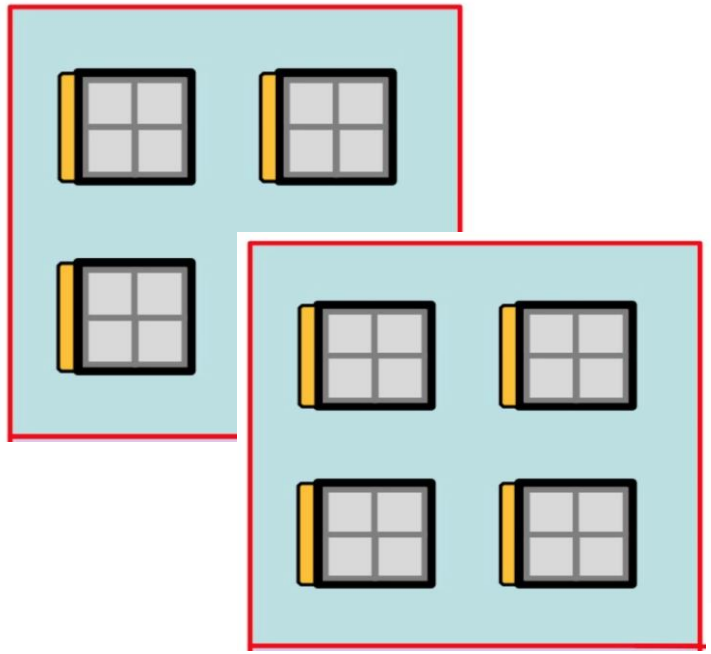


SLURM VOCABULARY:

Nodes | Partitions | Jobs

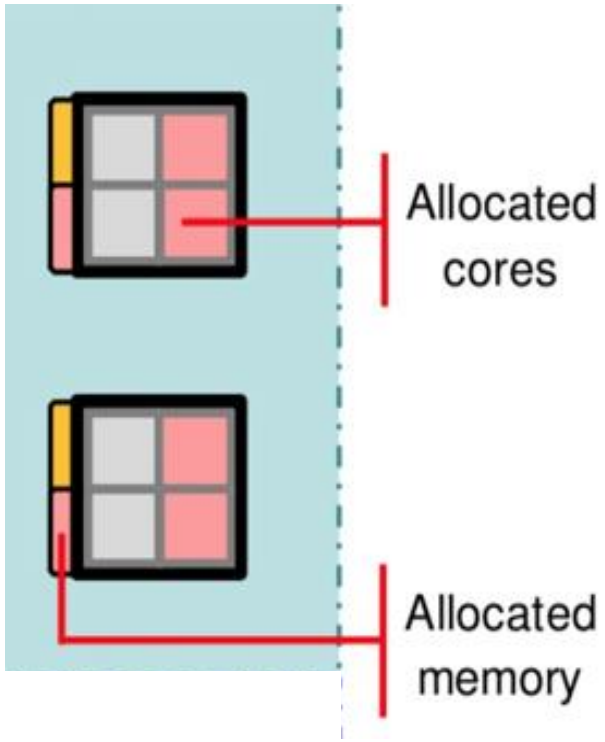


Nodes | **Partitions** | Jobs



SLURM VOCABULARY:

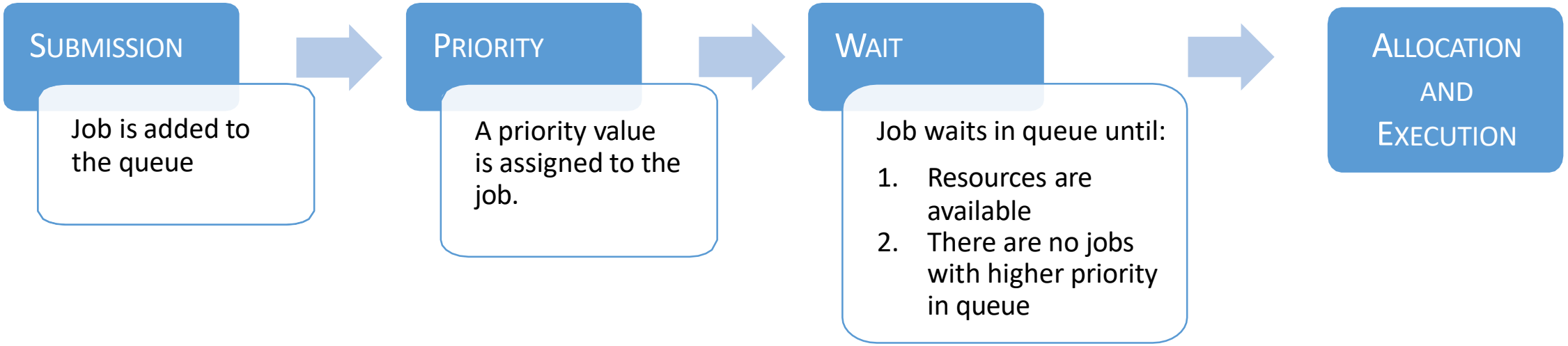
Nodes | Partitions | **Jobs**



GETTING AN ACCOUNT

- CU Boulder, CSU users and affiliates:
  - Request an account through the RC Account request portal: https://rcamp.rc.colorado.edu/accounts/account-request/create/organization
- AMC, RMACC users and affiliates:
  - Request an account through the ACCESS-CI User Registration Portal: https://identity.access-ci.org/new-user.html

Slurm Job lifecycle : the happy path



ANATOMY OF A JOB SCRIPT

A **batch job** consists of a sequence of commands listed in a file with the purpose of being executed by the OS as a single instruction.

myjob.sh

```
#!/bin/bash

#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --mem=1G
#SBATCH --partition=amilan
#SBATCH --time=00:10:00
#SBATCH --job-name=myjob
#SBATCH --output=myjob_%j.out

# script commands
echo "Job started on $(hostname) at $(date)"
sleep 60
echo "Job completed at $(date)"
```

ANATOMY OF A JOB SCRIPT

A **batch job** consists of a sequence of commands listed in a file with the purpose of being executed by the OS as a single instruction.

SHEBANG

- Specify the script interpreter (Bash)
- Must be the first line!

myjob.sh

```
#!/bin/bash

#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --mem=1G
#SBATCH --partition=amilan
#SBATCH --time=00:10:00
#SBATCH --job-name=myjob
#SBATCH --output=myjob_%j.out

# script commands
echo "Job started on $(hostname) at $(date)"
sleep 60
echo "Job completed at $(date)"
```

Anatomy of a Job Script

A **batch job** consists of a sequence of commands listed in a file with the purpose of being executed by the OS as a single instruction.

Shebang

- Specify the script interpreter (Bash)
- Must be the first line!

SLURM directives

- Start with "#SBATCH":
  Parsed by Slurm but ignored by Bash.
- Must be before actual commands!

myjob.sh

```
#!/bin/bash

#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --mem=1G
#SBATCH --partition=amilan
#SBATCH --time=00:10:00
#SBATCH --job-name=myjob
#SBATCH --output=myjob_%j.out

# script commands
echo "Job started on $(hostname) at $(date)"
sleep 60
echo "Job completed at $(date)"
```

Anatomy of a Job Script

A **batch job** consists of a sequence of commands listed in a file with the purpose of being executed by the OS as a single instruction.

Shebang

- Specify the script interpreter (Bash)
- Must be the first line!

SLURM directives

- Start with "#SBATCH":
  Parsed by Slurm but ignored by Bash.
- Must be before actual commands!

Script commands

- Commands you want to execute on the compute nodes.

myjob.sh

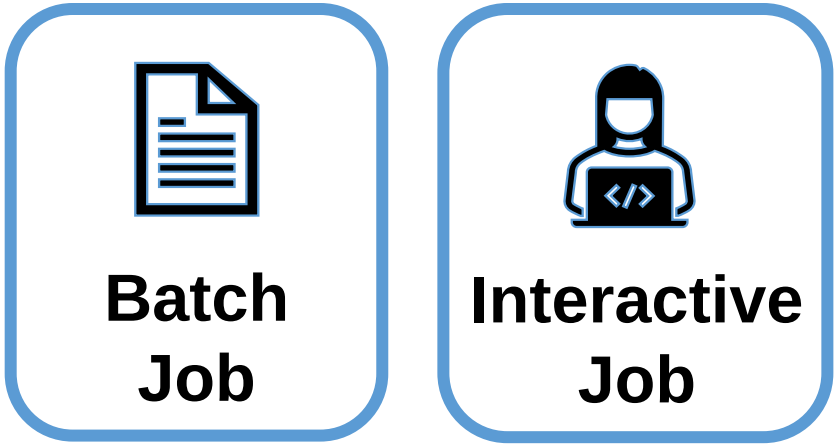
```
#!/bin/bash

#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --mem=1G
#SBATCH --partition=amilan
#SBATCH --time=00:10:00
#SBATCH --job-name=myjob
#SBATCH --output=myjob_%j.out

# script commands
echo "Job started on $(hostname) at $(date)"
sleep 60
echo "Job completed at $(date)"
```

Job Submission

Batch vs Interactive



JOB SUBMISSION

Batch vs Interactive

```
sbatch batch_file
```

- Runs the job in the background, without real-time user input.
- If successful, returns a job ID.
- Best for scheduled and long/big jobs

JOB SUBMISSION

Batch vs Interactive

`sbatch batch_file`

- Runs the job in the background, without real-time user input.
- If successful, returns a job ID.
- Best for scheduled and long/big jobs

`salloc options`

- Used to request an interactive job allocation.
- Accepts similar options as sbatch
- Best for testing and debugging

SUBMIT YOUR FIRST JOB

- Using sbatch to submit myjob.sh script to Slurm.

```
sbatch myjob.sh
```

```
$ sbatch myjob.sh
Submitted batch job 123456
```

SUBMIT YOUR FIRST JOB

- Using sbatch to submit myjob.sh script to Slurm.

```
sbatch myjob.sh
```

```
$ sbatch myjob.sh
Submitted batch job 123456
```

How do I remove a job from the queue?

SUBMIT YOUR FIRST JOB

- Using sbatch to submit myjob.sh script to Slurm.

`sbatch myjob.sh`

```
$ sbatch myjob.sh
Submitted batch job 123456
```

How do I remove a job from the queue?

`scancel job_id`

- Cancel the running or pending job from the queue using the corresponding *job_id*.

CHECK JOB QUEUE

```
squeue options
```

- This command is used to pull up information about the jobs currently in the Slurm queue.
- By default, the command will print out information for all jobs queued or running.
- To view jobs queued by a specific user use the -u flag

```
squeue -u user_name
```

```
[user@login ~]$ squeue
JOBID    PARTITION   NAME     USER     ST   TIME   NODES  NODELIST(REASON)
123456   amilan      myjob1   alice    R    10:34      2  c3cpu-c15-u1-3
123457   acompile    myjob2   bob      PD   00:00      1  (Priority)
123458   aa100       myjob3   charlie  PD   00:00      2  (AssocMaxJobsLimit)
```

CHECK JOB STATUS

squeue

- By default, the command will print out the job ID, partition, username, job status, number of nodes, and name of nodes for all jobs queued or running.

```
[user@login ~]$ squeue
JOBID    PARTITION   NAME     USER     ST   TIME   NODES  NODELIST(REASON)
123456   amilan      myjob1   alice    R    10:34      2  c3cpu-c15-u1-3
123457   acompile    myjob2   bob      PD   00:00      1  (Priority)
123458   aa100       myjob3   charlie  PD   00:00      2  (AssocMaxJobsLimit)
```

STATUS

R = Running

PD = Pending

CA = Cancelled

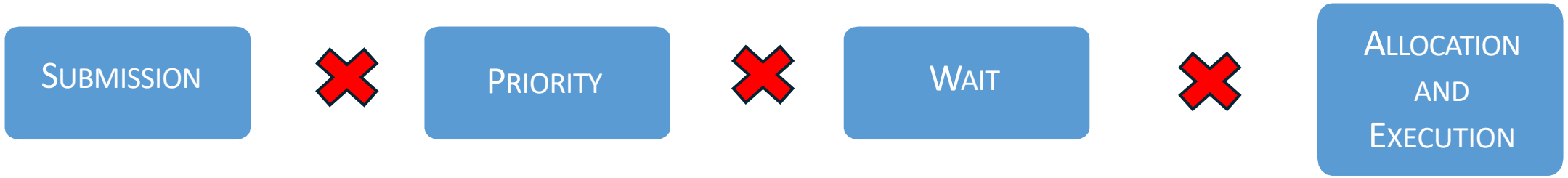
CD = Completed

Interpreting Reason Codes

```
[user@login ~]$ squeue
JOBID    PARTITION   NAME     USER     ST   TIME   NODES  NODELIST(REASON)
123456   amilan      myjob1   alice    R    10:34      2  c3cpu-c15-u1-3
123457   acompile    myjob2   bob      PD   00:00      1  (Priority)
123458   aa100       myjob3   charlie  PD   00:00      2  (AssocMaxJobsLimit)
```

Priority	Other jobs in queue have higher priority.
Resources	Insufficient resources available on the cluster.
Dependency	This job is waiting for a dependent job to complete and will run afterward.
AssociationMaxJobsLimit	Maximum number of jobs for your job's association have been met.
QOSGrpCpuLimit	All CPUs assigned to your job's specified QoS are in use.

When Things Go Wrong With Slurm Jobs



After a Job Fails, Ask Yourself...

- What do I know about the failure?
- Did I get any error messages?
- Was my job request realistic?

After a Job Fails, Ask Yourself...

What do I know about the failure?

- Did the job finish or crash midway?
- Did it run, or just queued and failed before starting?

Did I get any error messages?

Was my job request realistic?

View Historical Job Info

```
sacct options
```

- Check back on usage statistics of previous jobs.
- By default, only checks all jobs from the start of the current day

```
sacct --format=var_1,var_2, ... ,var_N
```

```
sacct --start=MM/DD/YYYY
```

```
sacct -j job_id
```

Variable	Description
account	Account the job ran under.
elapsed	Jobs elapsed time formatted as DD-HH:MM:SS.
exitcode	The exit code returned by the job script
jobid	The id of the Job.
jobname	The name of the Job.
ntasks	Number of tasks in a job.
reqmem	Required amount of memory for a job.

After a Job Fails, Ask Yourself...

What do I know about the failure?

Did I get any error messages?

- Were there any module errors?
- Was a file or command missing?
- Any permission errors?

Was my job request realistic?

CHECK THE LOGS

```
less slurm-<job_id>.out
less slurm-<job_id>.err
```

- Look for common issues like:

✕ Module errors: command not found, module load failed

✕ Missing files: No such file or directory

✕ Permissions: Permission denied

CHECK THE LOGS

```
less slurm-<job_id>.out
less slurm-<job_id>.err
```

- Look for common issues like:

✕ Module errors: command not found, module load failed

✕ Missing files: No such file or directory

✕ Permissions: Permission denied

```
[user@login ~]$ less slurm-345678.out
Starting job at Sat May 18 14:32:17 MDT 2025
Job completed at Sat May 18 14:32:37 MDT 2025
```

```
[user@login ~]$ less slurm-345678.err
cat: missing_file.txt: No such file or directory
```

CHECK THE LOGS

```
less slurm-<job_id>.out
less slurm-<job_id>.err
```

- Look for common issues like:
- ✕ Module errors: command not found, module load failed
- ✕ Missing files: No such file or directory
- ✕ Permissions: Permission denied

```
sacct -j job_id -B
```

- The *-B* option will print the batch script of job if the job used one. If the job didn't have a script 'NONE' is output.

CHECK THE LOGS

```
less slurm-<job_id>.out
less slurm-<job_id>.err
```

- Look for common issues like:
- ✕ Module errors: command not found, module load failed
- ✕ Missing files: No such file or directory
- ✕ Permissions: Permission denied

```
sacct -j job_id -B
```

- The *-B* option will print the batch script of job if the job used one. If the job didn't have a script 'NONE' is output.

```
[user@login ~]$ sacct -j 345678 -B
Batch Script for 345678
--------------------------------------------------------------------------------
#!/bin/bash

#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --mem=1G
#SBATCH --partition=amilan
#SBATCH --time=00:05:00
#SBATCH --job-name=error_demo
#SBATCH --output=error_demo_%j.out
#SBATCH --error=error_demo_%j.err

# script commands
echo "Starting job at $(date)"
cat missing_file.txt
sleep 20
echo "Job completed at $(date)"
```

After a Job Fails, Ask Yourself...

What do I know about the failure?

Did I get any error messages?

Was my job request realistic?

- Did I request too much memory, time, or CPUs?
- Was the partition selected available and ready for my job?
- Did my job use the resources I requested efficiently?

Cluster State and Partition Details

`sinfo` *options*

- A quick way to check the cluster

```
[user@login ~]$ sinfo
PARTITION   AVAIL  TIMELIMIT     NODES  STATE     NODELIST
amem        up     7-00:00:00        1  drain*    c3mem-a4-u42-3
amem        up     7-00:00:00        2  mix       c3mem-a9-u21-1,c3mem-a9-u22-1
amem        up     7-00:00:00       12  alloc     c3mem-a4-u34-[1-4],c3mem-a4-u36-[1-4]
amem        up     7-00:00:00        9  idle      c3mem-a4-u38-[2-3],c3mem-a4-u40-[1-4]
aa100       up     1-00:00:00       11  mix       c3gpu-a9-u29-1,c3gpu-a9-u31-1
al40        up     1-00:00:00        3  mix       c3gpu-a9-u15-1,c3gpu-a9-u17-1
ami100      up     1-00:00:00        1  resv      c3gpu-c2-u29
```

CLUSTER STATE AND PARTITION DETAILS

`sinfo options`

- A quick way to check the cluster

```
[user@login ~]$ sinfo
PARTITION   AVAIL  TIMELIMIT     NODES  STATE    NODELIST
amem        up     7-00:00:00        1  drain*   c3mem-a4-u42-3
amem        up     7-00:00:00        2  mix      c3mem-a9-u21-1,c3mem-a9-u22-1
amem        up     7-00:00:00       12  alloc    c3mem-a4-u34-[1-4],c3mem-a4-u36-[1-4]
amem        up     7-00:00:00        9  idle     c3mem-a4-u38-[2-3],c3mem-a4-u40-[1-4]
aa100       up     1-00:00:00       11  mix      c3gpu-a9-u29-1,c3gpu-a9-u31-1
al40        up     1-00:00:00        3  mix      c3gpu-a9-u15-1,c3gpu-a9-u17-1
ami100      up     1-00:00:00        1  resv     c3gpu-c2-u29
```

idle	Node is up and fully available for job scheduling
alloc	Node is fully allocated to one or more running jobs
mix	Node is partially allocated – some CPUs in use, others still available
drain*	Node is in a failed state and draining.
resv	Node is reserved (e.g., for specific users, groups, or jobs)
down	Node is offline or unreachable

JOB EFFICIENCY SUMMARY

```
seff job_id
```

- This command is used to display the information about the job's CPU and memory utilization.
- Gives users feedback on how well their job used the requested resources.
- Efficiency statistics are only available for jobs that have completed

```
[user@login]$ seff 123456
Job ID: 123456
Cluster: alpine
User/Group: alice/rmacc-group
State: COMPLETED (exit code 0)
Nodes: 1
Cores per node: 16
CPU Utilized: 00:44:43
CPU Efficiency: 45.44% of 01:38:24 core-walltime
Job Wall-clock time: 00:06:09
Memory Utilized: 1.84 GB
Memory Efficiency: 3.07% of 60.00 GB
```

```
[user@login]$ seff 789101
Job ID: 789101
Cluster: alpine
User/Group: bob/rmacc-group
State: OUT_OF_MEMORY (exit code 0)
Nodes: 1
Cores per node: 8
CPU Utilized: 00:18:08
CPU Efficiency: 21.38% of 01:24:48 core-walltime
Job Wall-clock time: 00:10:36
Memory Utilized: 23.22 GB
Memory Efficiency: 79.80% of 29.10 GB
```

HANDS-ON TUTORIAL

Logging Into Research Computing

Login to CURC via your terminal (CU Boulder):

$ ssh mokh8410@login.rc.colorado.edu

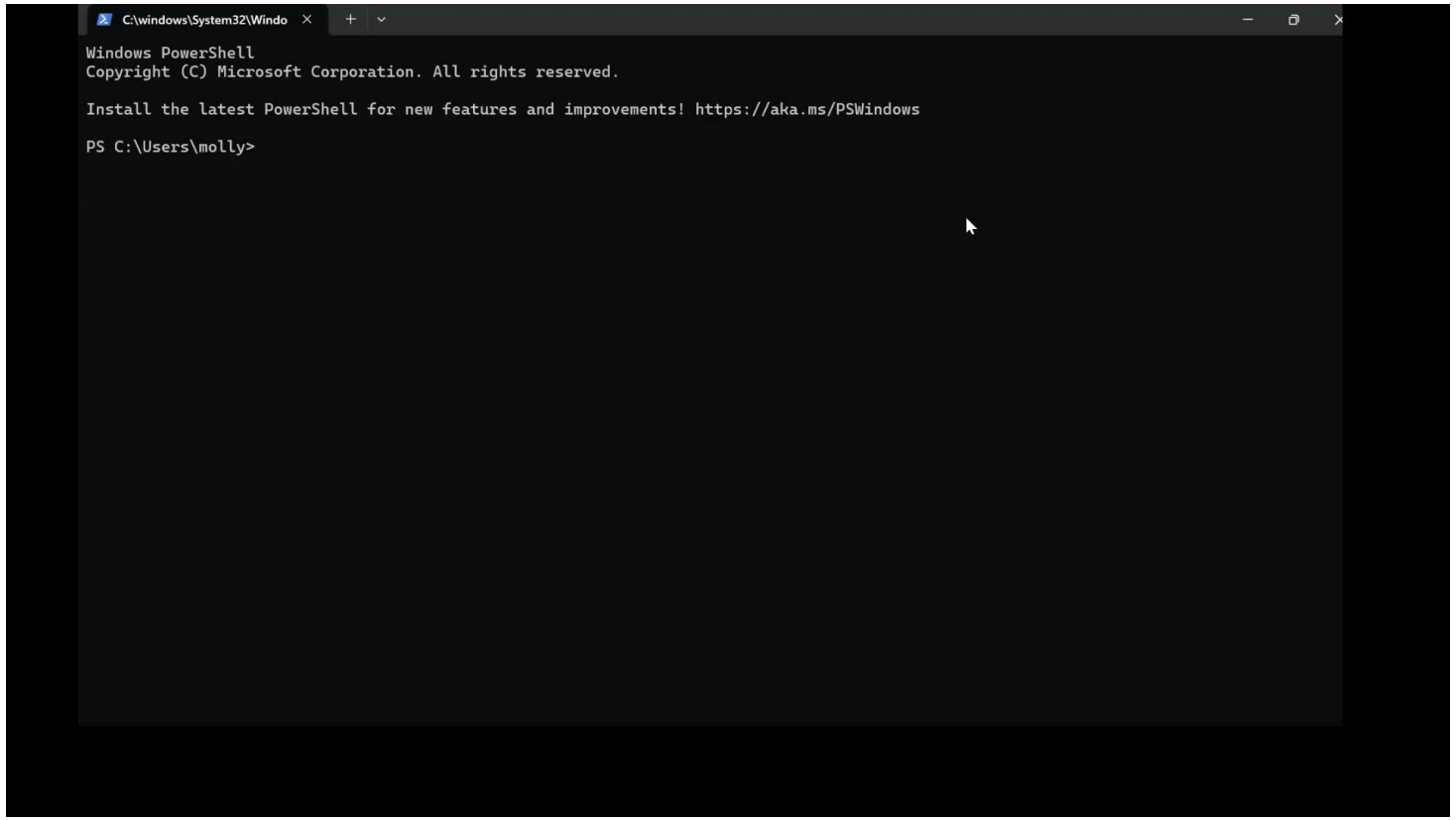
…or login to CURC via your browser (all users):

https://ondemand-rmacc.rc.colorado.edu

(once logged in, navigate to Clusters -> Alpine shell)

EXERCISE

- In this section, we'll walk through a live demo of how to write, submit, troubleshoot, and evaluate a Slurm job, step by step.



Want to try it later?

- Download this full hands-on example from our GitHub: github.com/ResearchComputing/rmacc_2025/

More Documentation!

Official Slurm Docs!

https://slurm.schedmd.com/documentation.html

CURC Slurm Guide

https://curc.readthedocs.io/en/latest/running-jobs/slurm-commands.html

Questions

Comments