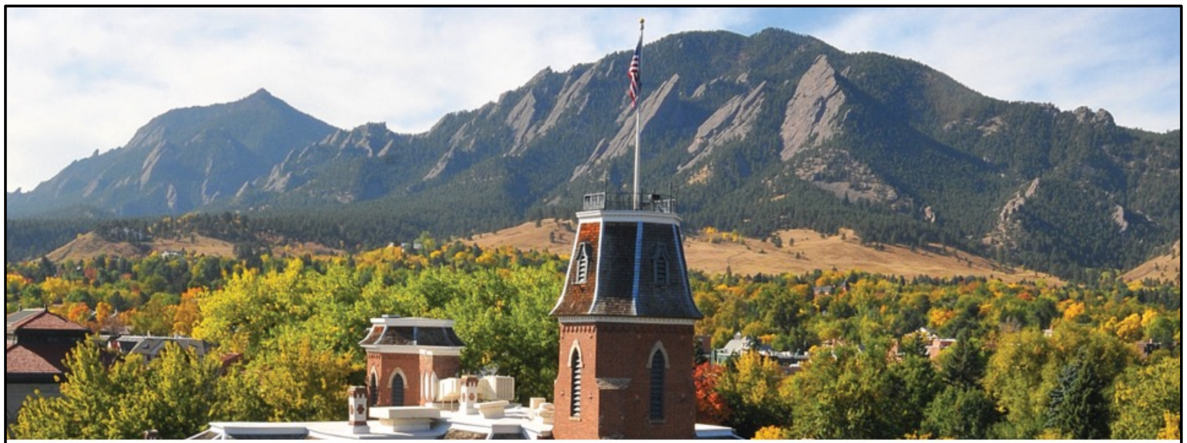


Scheduling Jobs on a Supercomputer

Research Computing
UNIVERSITY OF COLORADO BOULDER

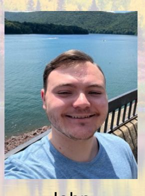
Meet the User Support Team

Layla Freeborn

Brandon Reyes

Andy Monaghan

Michael Schneider


John Reiland

Dylan Gottlieb

Mohal Khandelwal

Ragan Lee

Website: www.rc.colorado.edu

Documentation: https://curc.readthedocs.io

Helpdesk: rc-help@colorado.edu

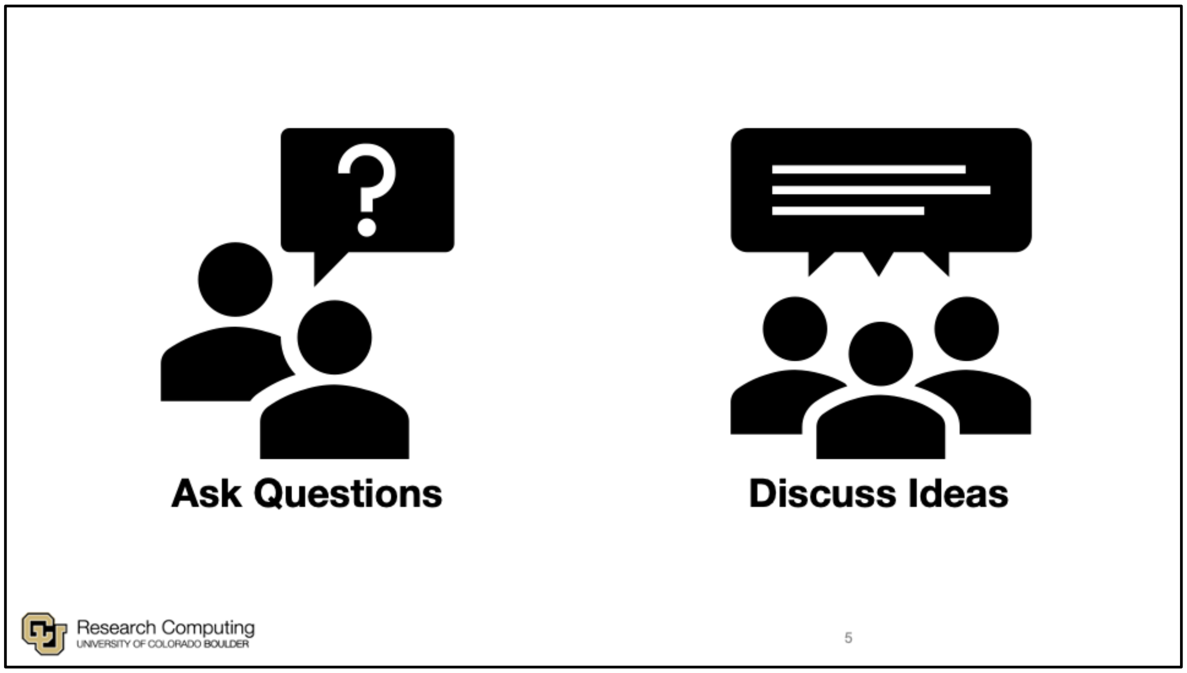
Research Computing
UNIVERSITY OF COLORADO **BOULDER**

https://github.com/ResearchComputing/scheduling_a_job_on_a_supercomputer_rc_shortcourse

Research Computing
UNIVERSITY OF COLORADO BOULDER

9/5/24

4

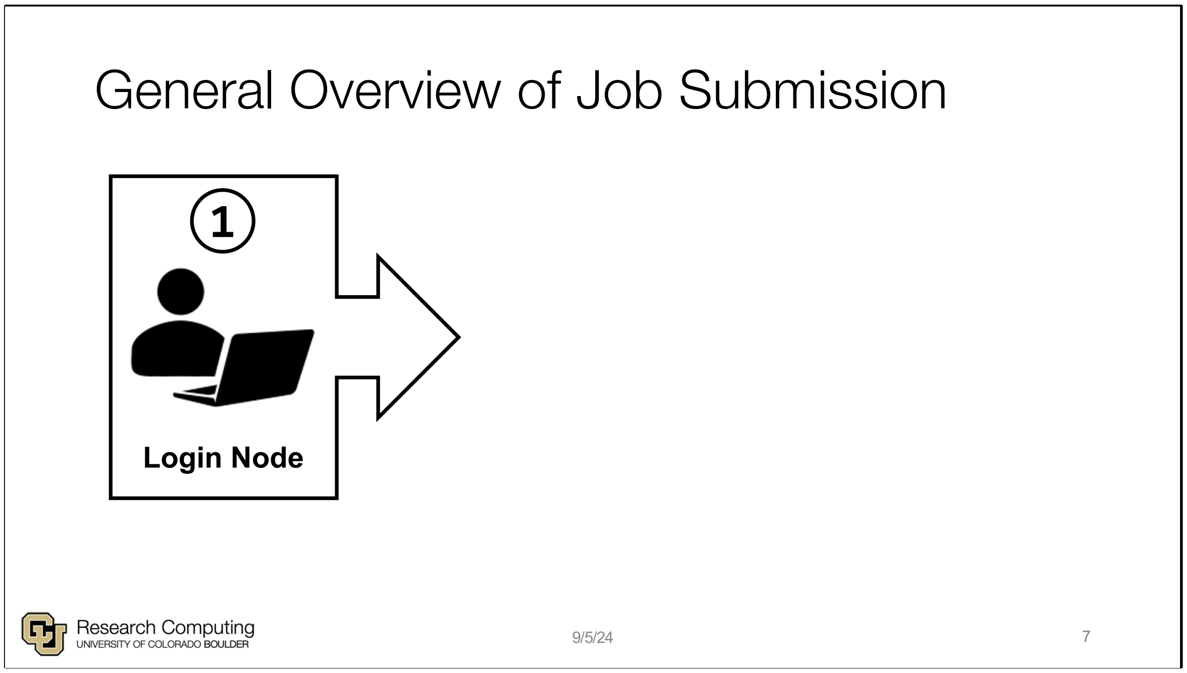


This workshop is meant to be empowering and information-rich – to help you get the most out of your HPC workflow.

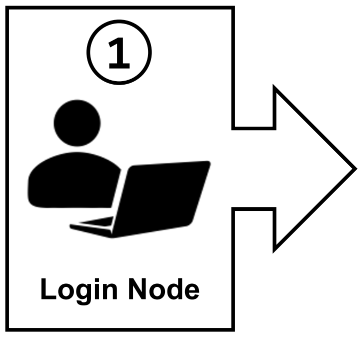
Feel free to ask questions during the presentations and to discuss ideas/thoughts during breaks and the hands-on portion of the workshop.

Session Overview

- Overview of Jobs Submissions
- Resource Requests
- Interactive & Batch Jobs
- Monitoring jobs



General Overview of Job Submission

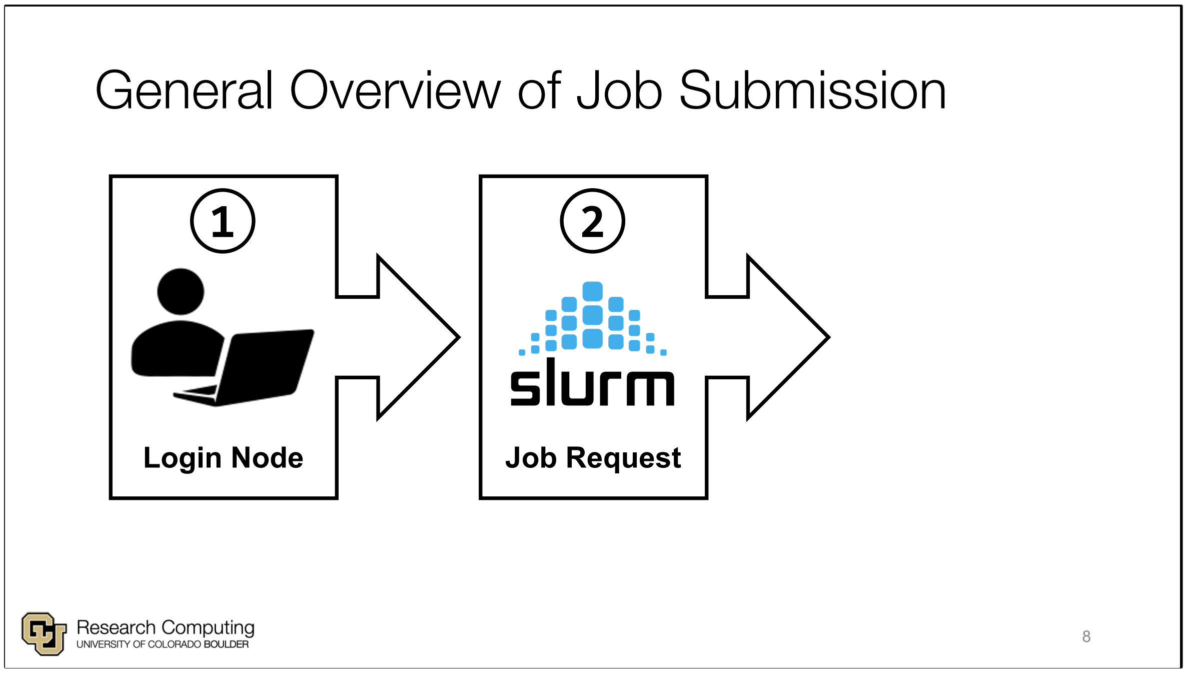


A job submission consists of three general steps:

1. Log onto a CURC Login Node (ssh, OnDemand)

Warning/Reminder

Software cannot be accessed/run from a login node. You must run your applications from a compute node. But, in order to run on a compute node, you must first request the system resources through a ”job”



General Overview of Job Submission

1
Login Node
2
slurm
Job Request

Research Computing
UNIVERSITY OF COLORADO BOULDER

8

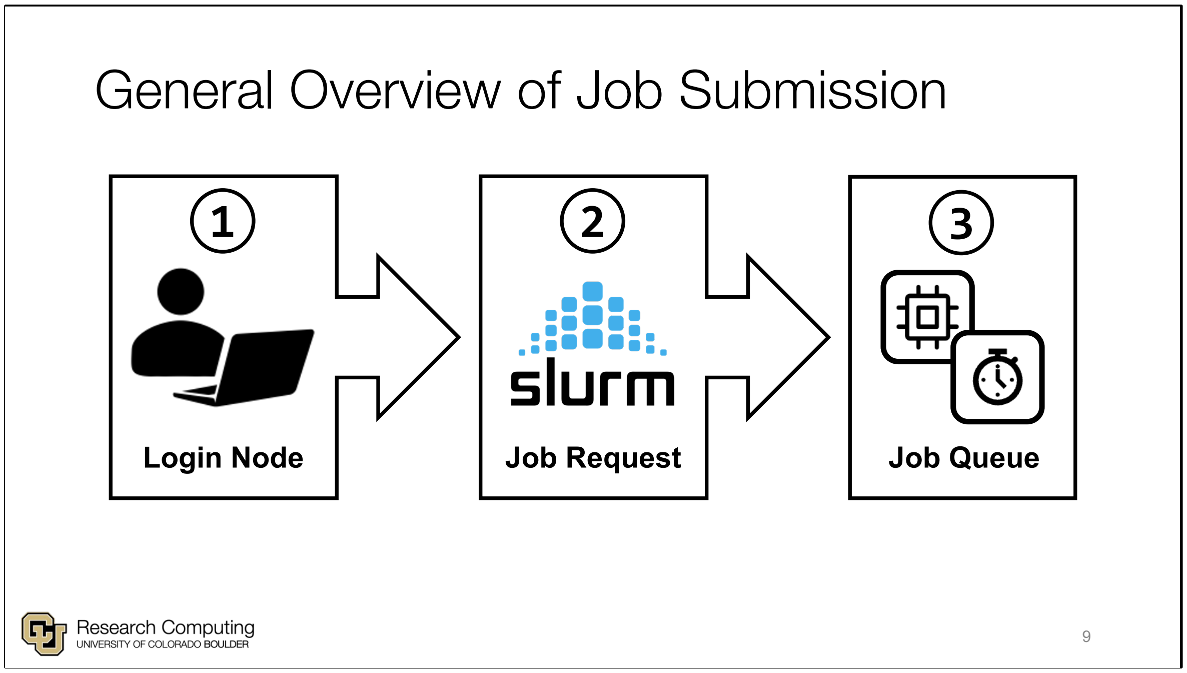
A job submission consists of three general steps:

1. Log onto a CURC Login Node (ssh, OnDemand)

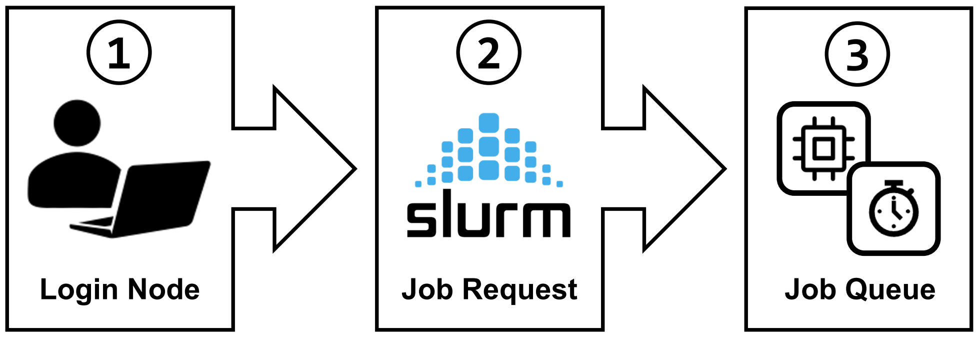
Warning/Reminder

Software cannot be accessed/run from a login node. You must run your applications from a compute node. But, in order to run on a compute node, you must first request the system resources through a ”job”

2. Job requests are submitted to SLURM, our system’s Job Scheduler which manages how and when system resources are allocated to user submitted jobs.



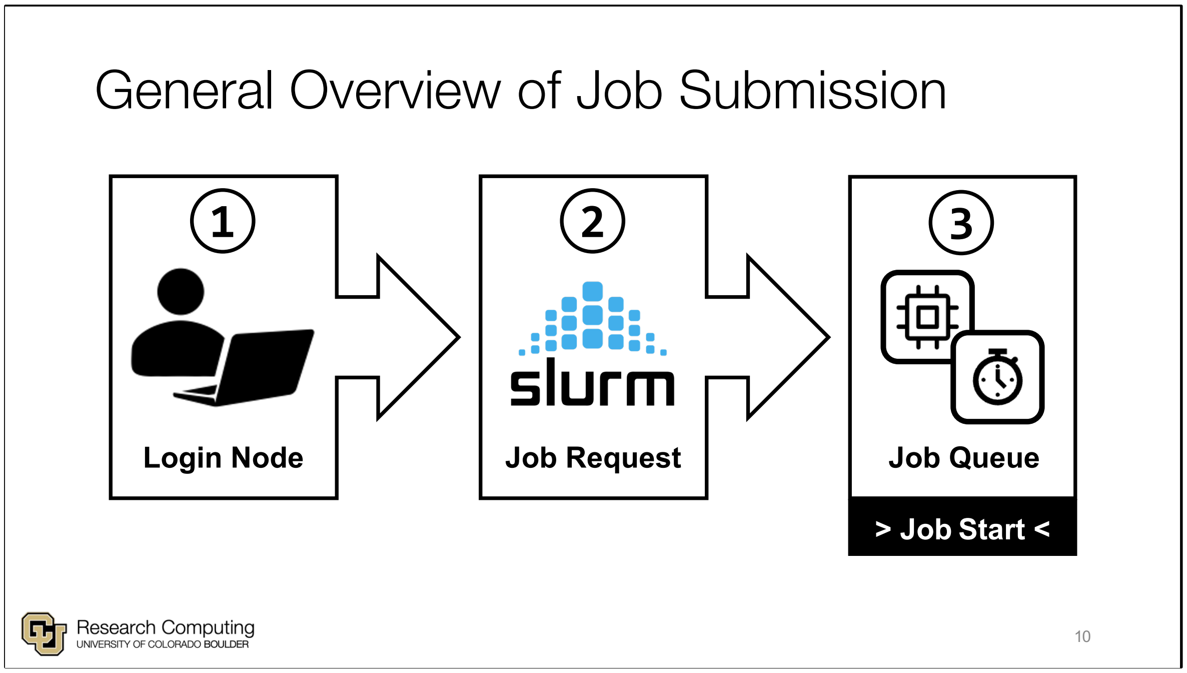
General Overview of Job Submission



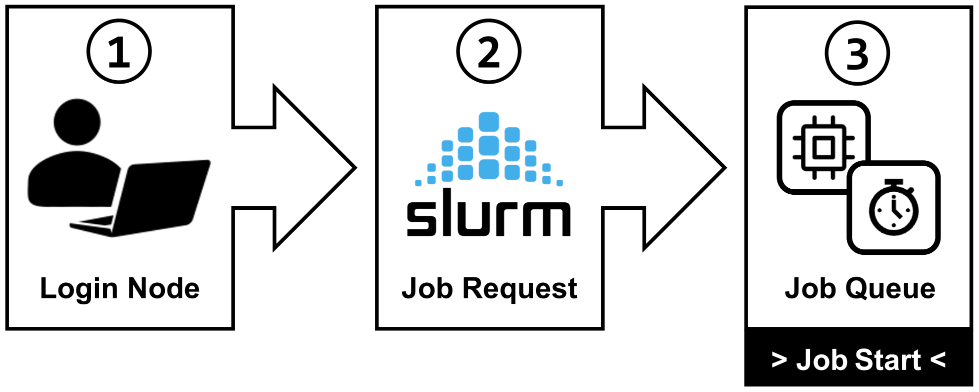
Research Computing
UNIVERSITY OF COLORADO BOULDER

9

3. Jobs do not run immediately but instead wait in a SLURM managed queue. A job's placement in the queue is dependent on multiple factors. We will discuss a few of the most common factors, but a general rule to follow is that the more resources a job requests the longer it will wait in the queue.



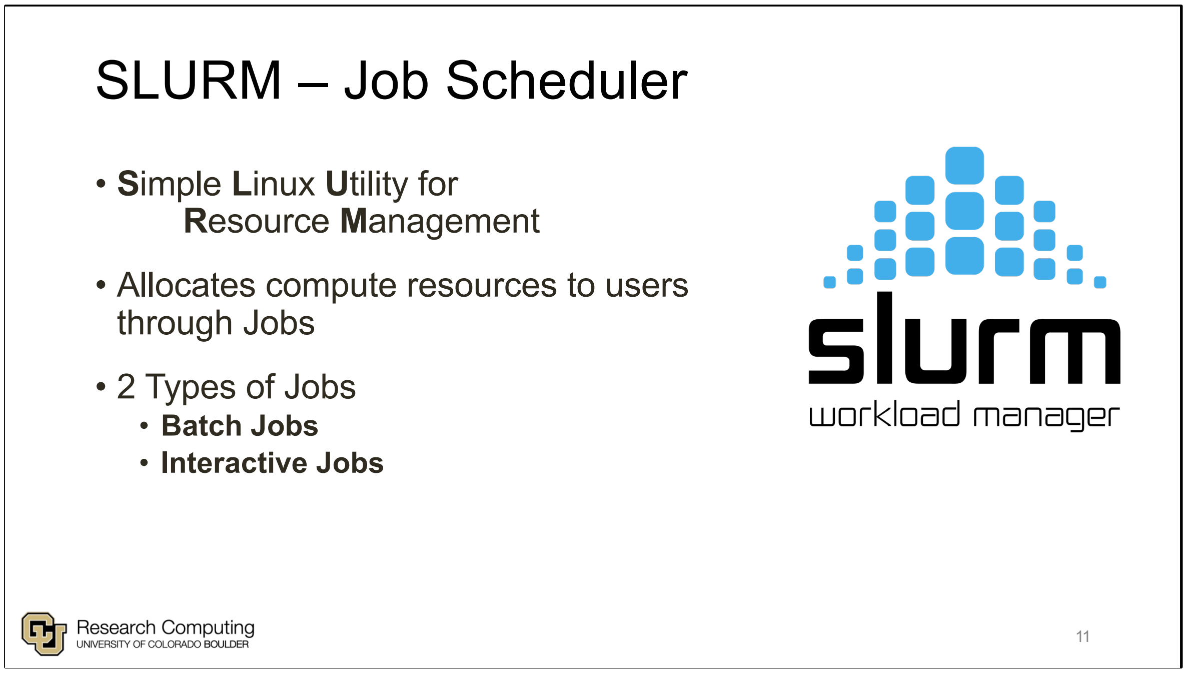
General Overview of Job Submission



Research Computing
UNIVERSITY OF COLORADO BOULDER

10

Once SLURM determines they system is ready, it will start the job. As we will discuss, quite a few factors influence how long a submitted job waits on the queue. But, in general, the more resources you request the longer your job will have to wait in the queue.



SLURM – Job Scheduler

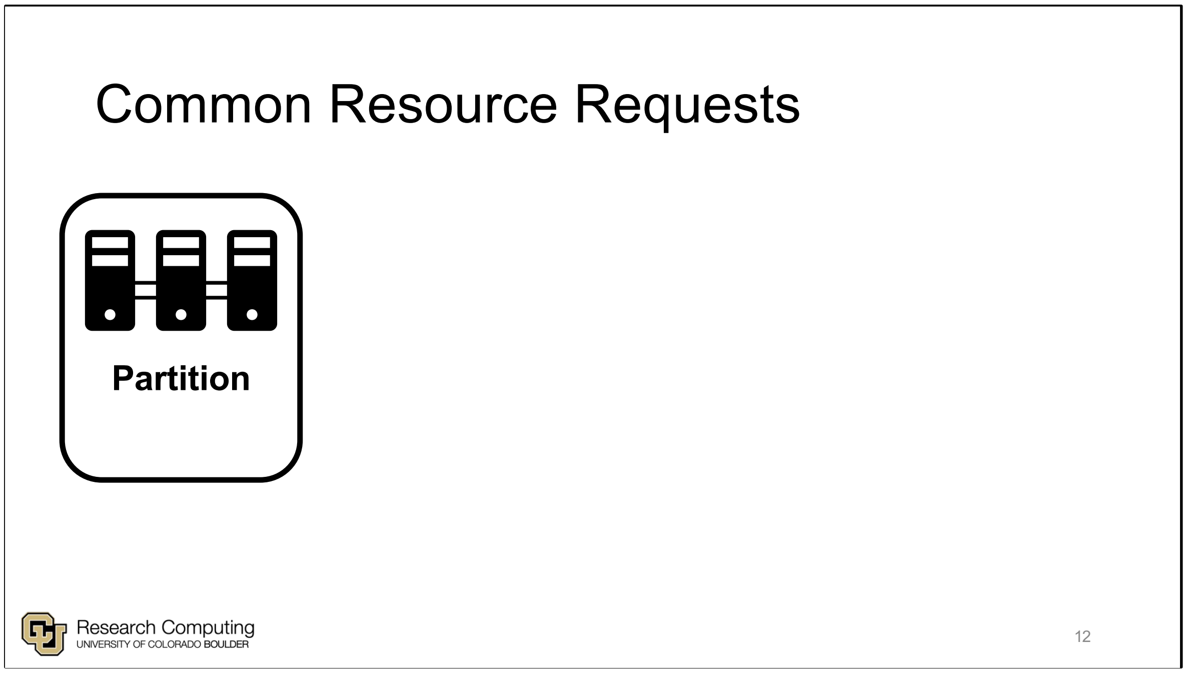
- **S**imple **L**inux **U**tility for **R**esource **M**anagement
- Allocates compute resources to users through Jobs
- 2 Types of Jobs
  - **Batch Jobs**
  - **Interactive Jobs**

Research Computing
UNIVERSITY OF COLORADO BOULDER

11

Slurm manages our system's shared resources and ensures they are equitably shared amongst CURC users.

In the next section we will discuss common resources a job can request and then cover the two types of jobs we support (Batch and Interactive).



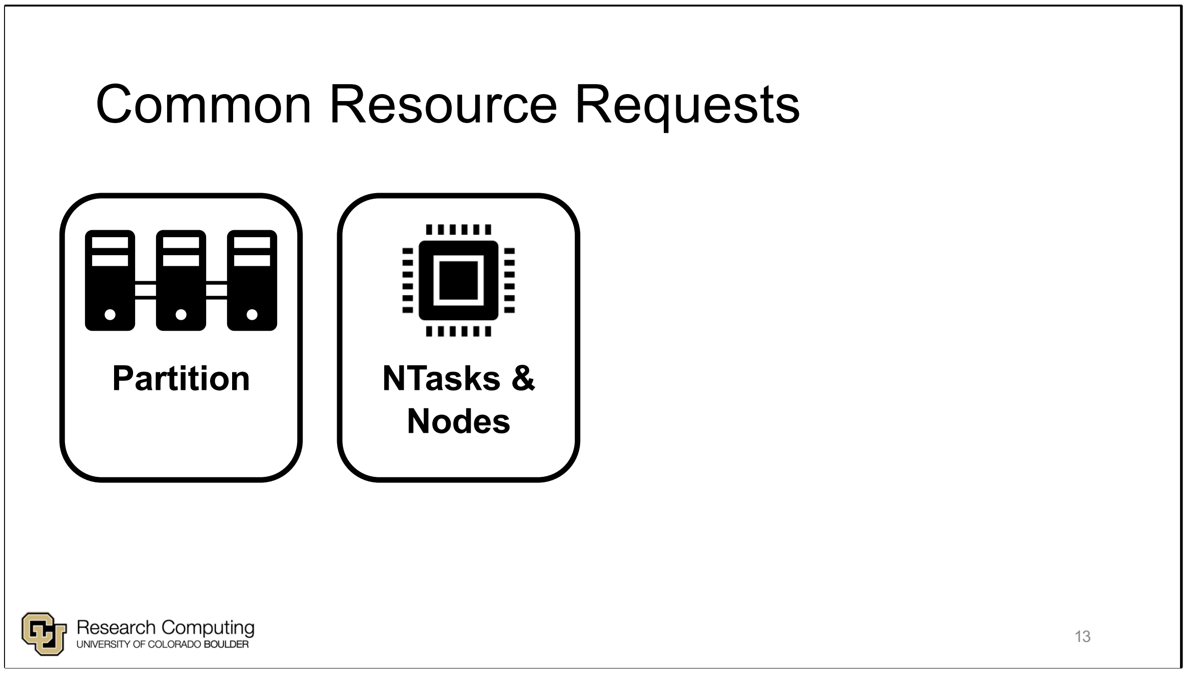
Common Resource Requests

Partition

Research Computing
UNIVERSITY OF COLORADO BOULDER

12

Partition – Defines what type of hardware your job has access to.



Common Resource Requests

Partition

NTasks & Nodes

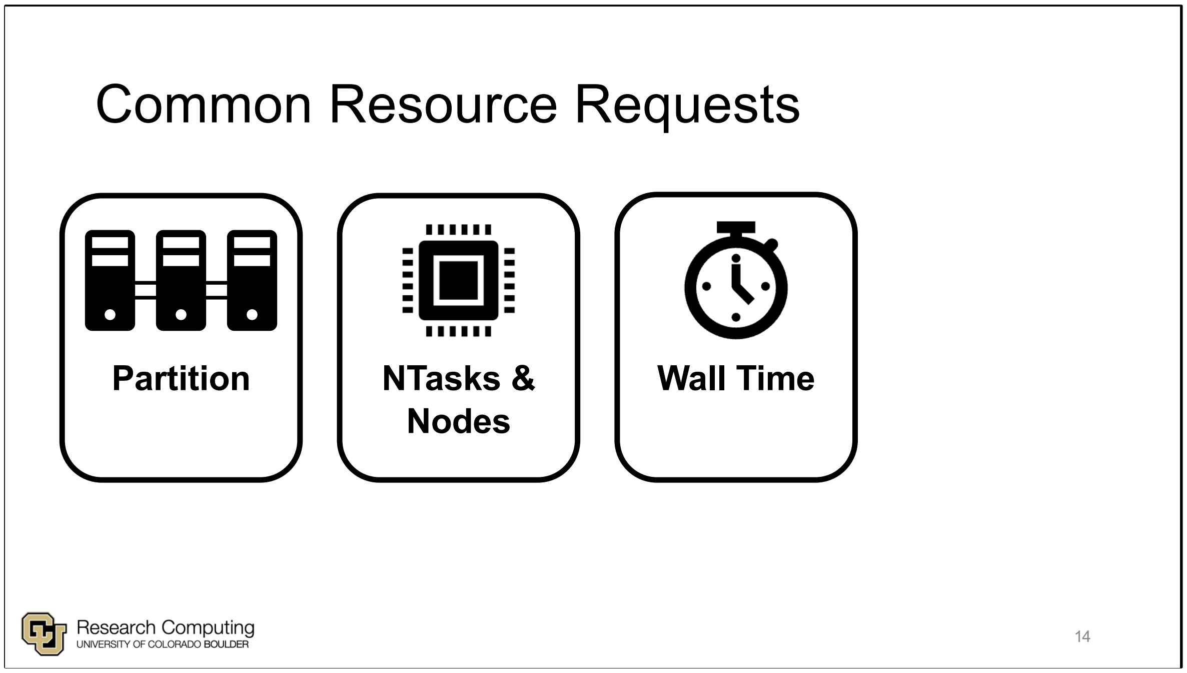
Research Computing
UNIVERSITY OF COLORADO BOULDER

13

Partition – Defines what type of hardware your job has access to.

Ntasks – How many CPU cores your job is requesting

Nodes – How many nodes your job will run across (typically 1, but can be more)



Common Resource Requests

Partition

NTasks & Nodes

Wall Time

Research Computing
UNIVERSITY OF COLORADO BOULDER

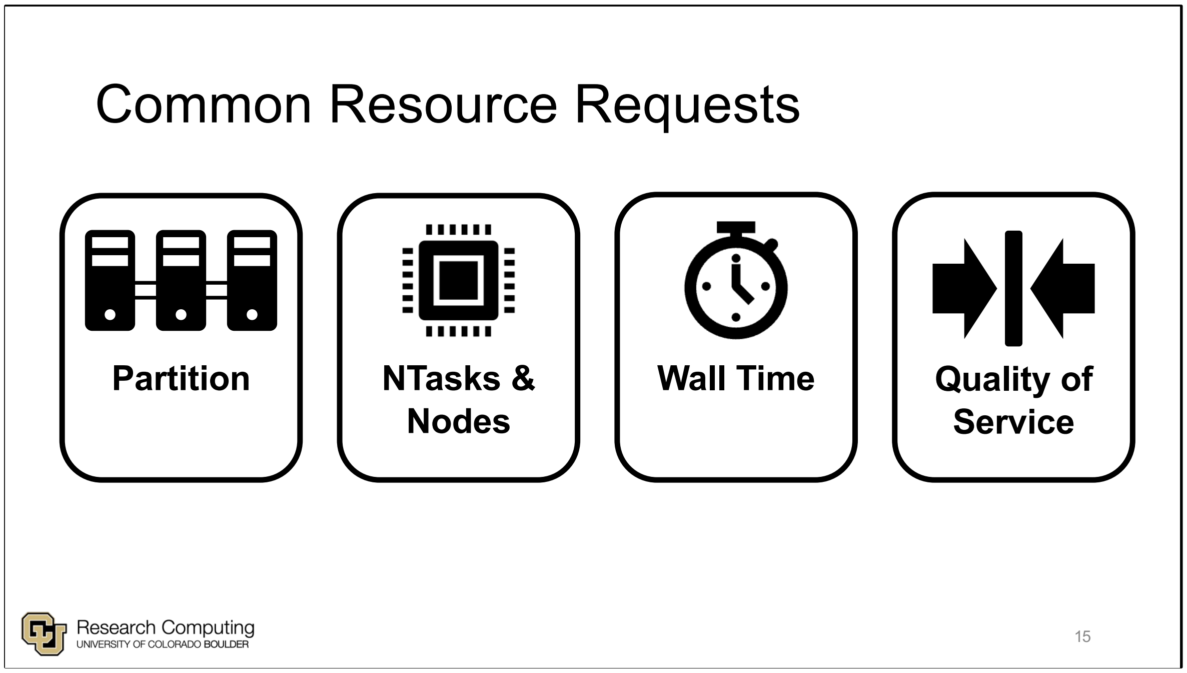
14

Partition – Defines what type of hardware your job has access to.

Ntasks – How many CPU cores your job is requesting

Nodes – How many nodes your job will run across (typically 1, but can be more)

Wall Time – How much time your job needs to run on the system (can be a few minutes, a few hours, to multiple days)



Common Resource Requests

Partition
NTasks & Nodes
Wall Time
Quality of Service
Research Computing
UNIVERSITY OF COLORADO BOULDER
15

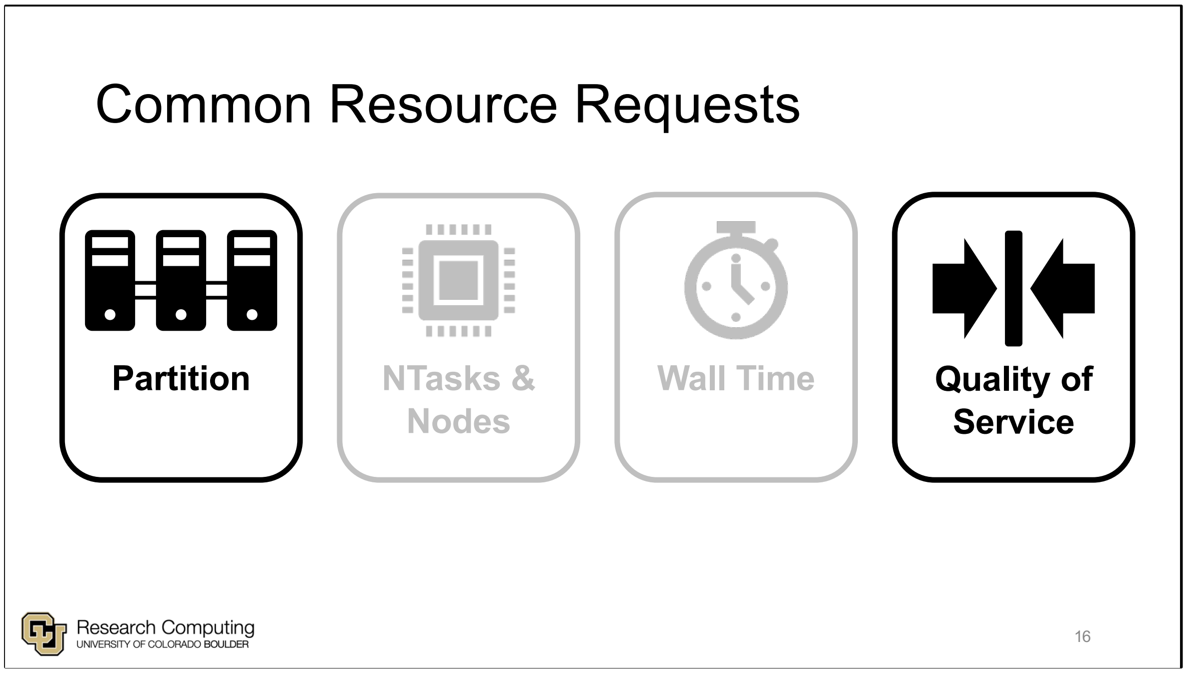
Partition – Defines what type of hardware your job has access to.

Ntasks – How many CPU cores your job is requesting

Nodes – How many nodes your job will run across (typically 1, but can be more)

Wall Time – How much time your job needs to run on the system (can be a few minutes, a few hours, to multiple days)

Quality of Service (QOS) – Defines additional job constraints – i.e. limits to how much your job can request.



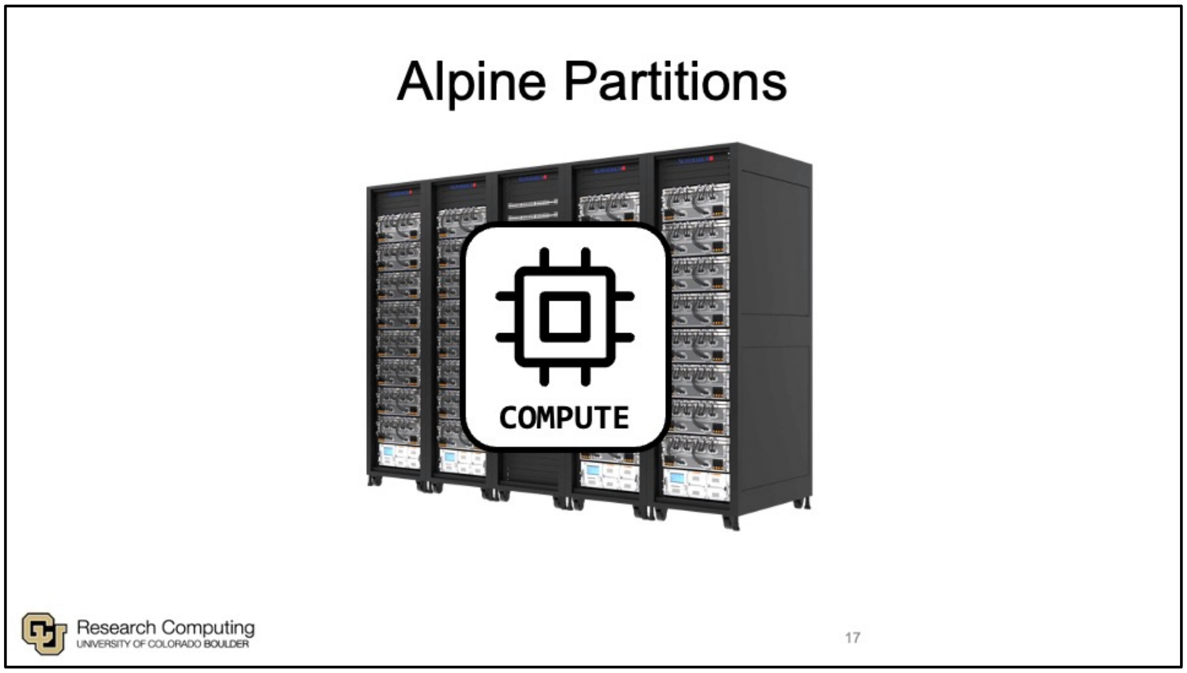
Common Resource Requests

Partition
NTasks & Nodes
Wall Time
Quality of Service

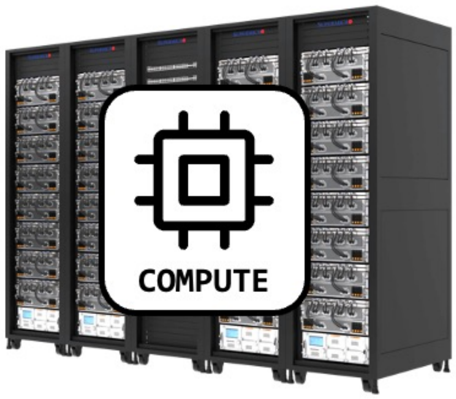
Research Computing
UNIVERSITY OF COLORADO BOULDER

16

What resources and how many you can request is heavily dependent on what partition and qos you select. We will now provide a general overview of our primary partitions and qos's.



Alpine Partitions



Research Computing
UNIVERSITY OF COLORADO BOULDER

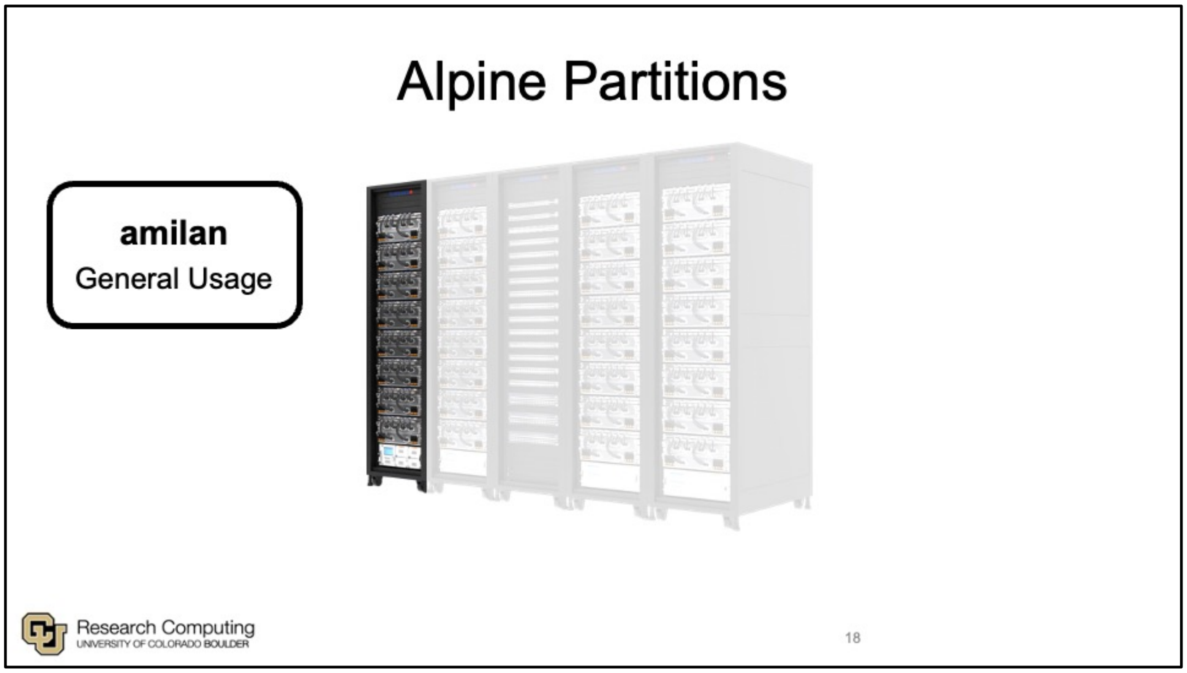
17

Compute nodes are organized into partitions – logical groups of similarly spec'd hardware.

Partitions make it easier to request specific resources and ensure a more consistent user experience.

The Alpine Cluster has four primary partitions.

Detailed Information on Alpine Hardware:
https://curc.readthedocs.io/en/latest/clusters/alpine/alpine-hardware.html



Alpine Partitions

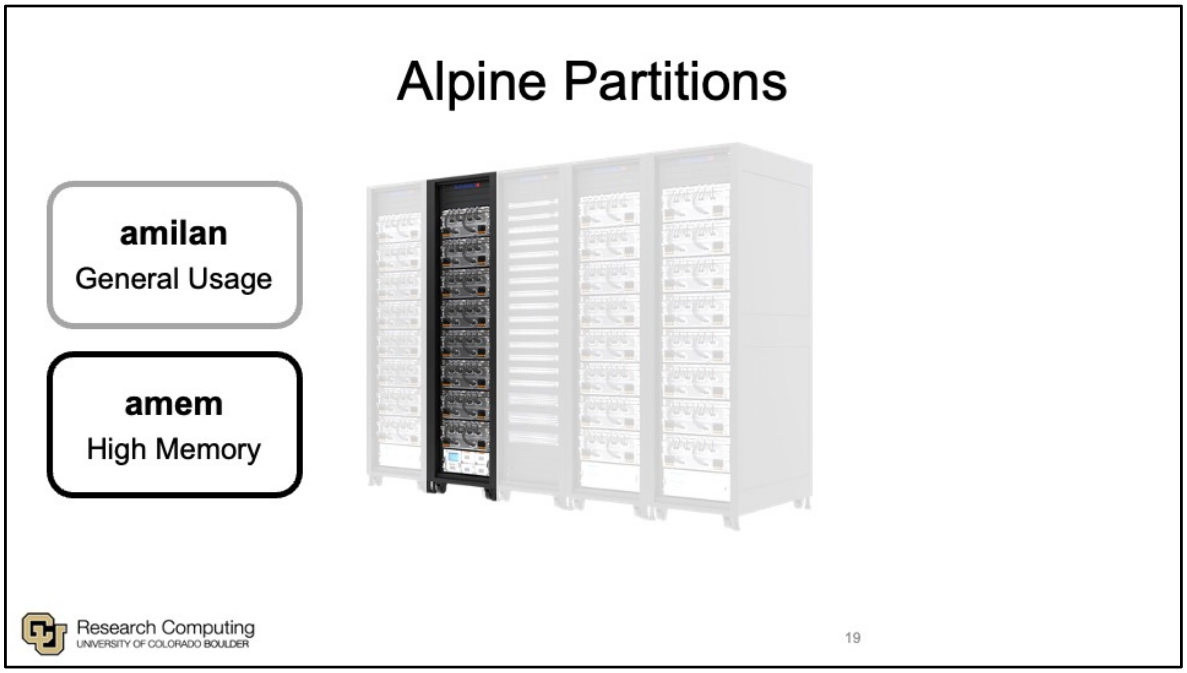
amilan
General Usage

Research Computing
UNIVERSITY OF COLORADO BOULDER

18

amilan:
285 Nodes
64 CPU Cores per Node
3.75 GB of RAM per CPU (240 GB total)

Detailed Information on Alpine Hardware:
https://curc.readthedocs.io/en/latest/clusters/alpine/alpine-hardware.html



Alpine Partitions

amilan
General Usage

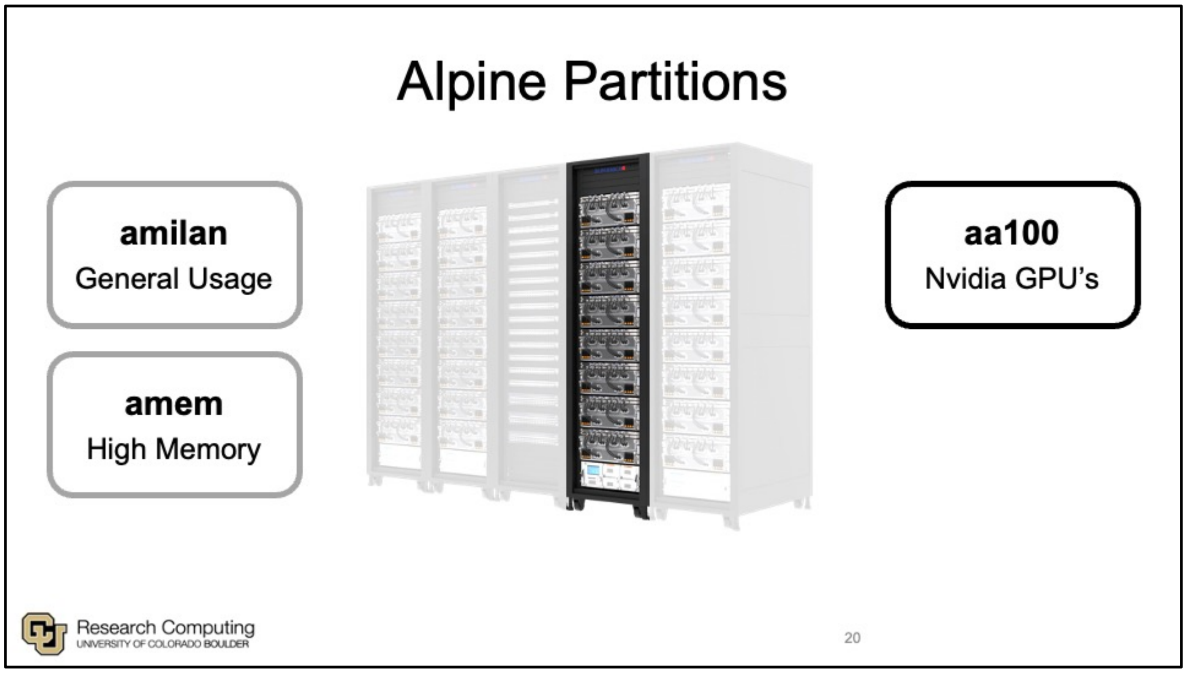
amem
High Memory

Research Computing
UNIVERSITY OF COLORADO BOULDER

19

amem:
22 Nodes
48 CPU Cores per node (12)
64 CPU Cores per node (10)
21.5 GB of RAM per CPU (1 TB total)

Detailed Information on Alpine Hardware:
https://curc.readthedocs.io/en/latest/clusters/alpine/alpine-hardware.html



Alpine Partitions

amilan
General Usage

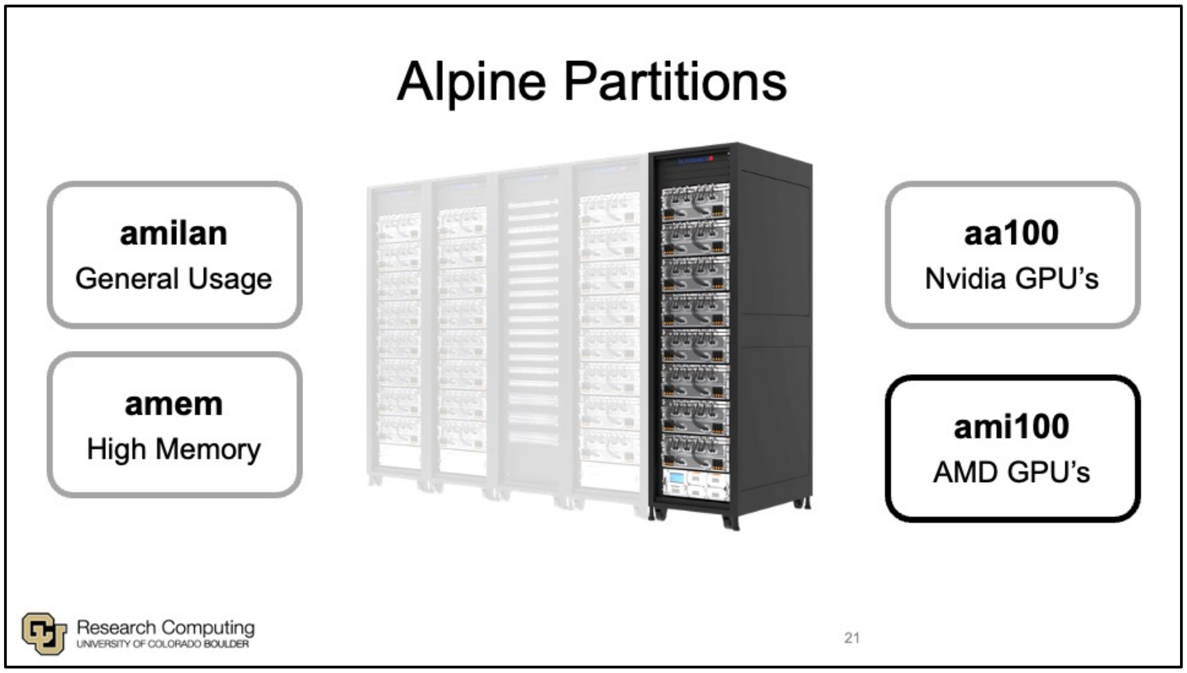
amem
High Memory

aa100
Nvidia GPU's

Research Computing
UNIVERSITY OF COLORADO BOULDER

20

aa100
12 Nodes
64 CPU Cores per Node
3.75 GB of RAM per CPU (240 GB total)
3 GPU's per Node (Nvidia A100)



Alpine Partitions

amilan
General Usage

amem
High Memory

aa100
Nvidia GPU's

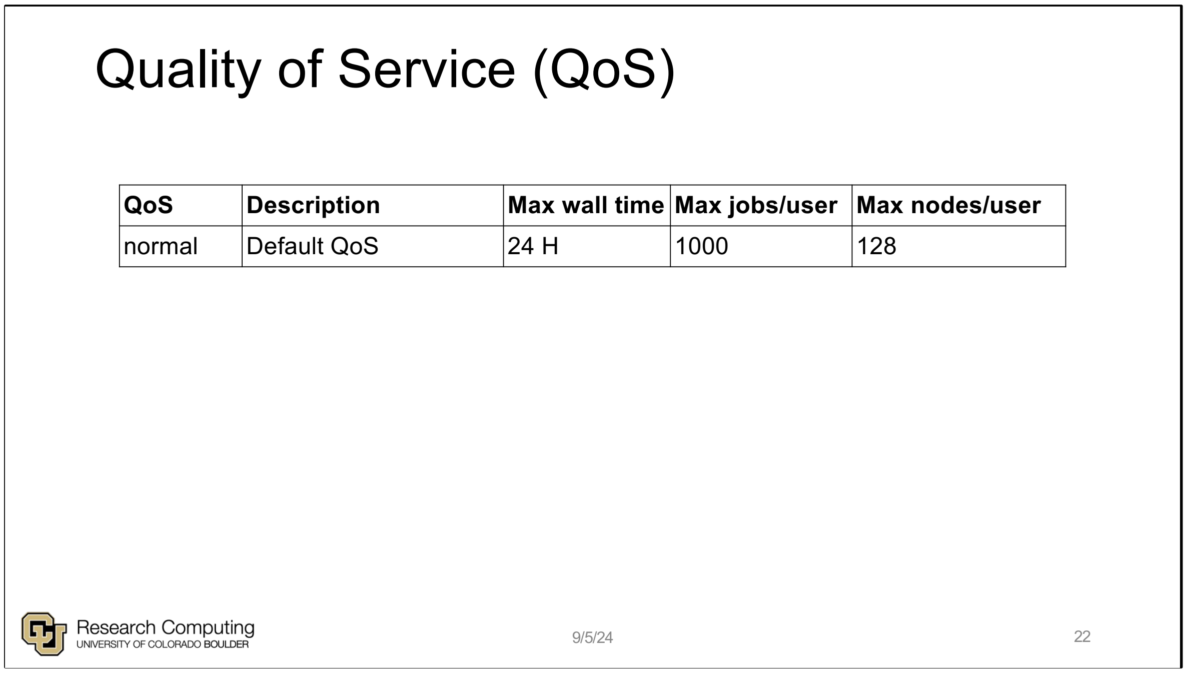
ami100
AMD GPU's

Research Computing
UNIVERSITY OF COLORADO BOULDER

21

ami100
8 Nodes
64 CPU Cores per Node
3.75 GB of RAM per CPU (240 GB total)
3 GPU's per Node (AMD MI100)

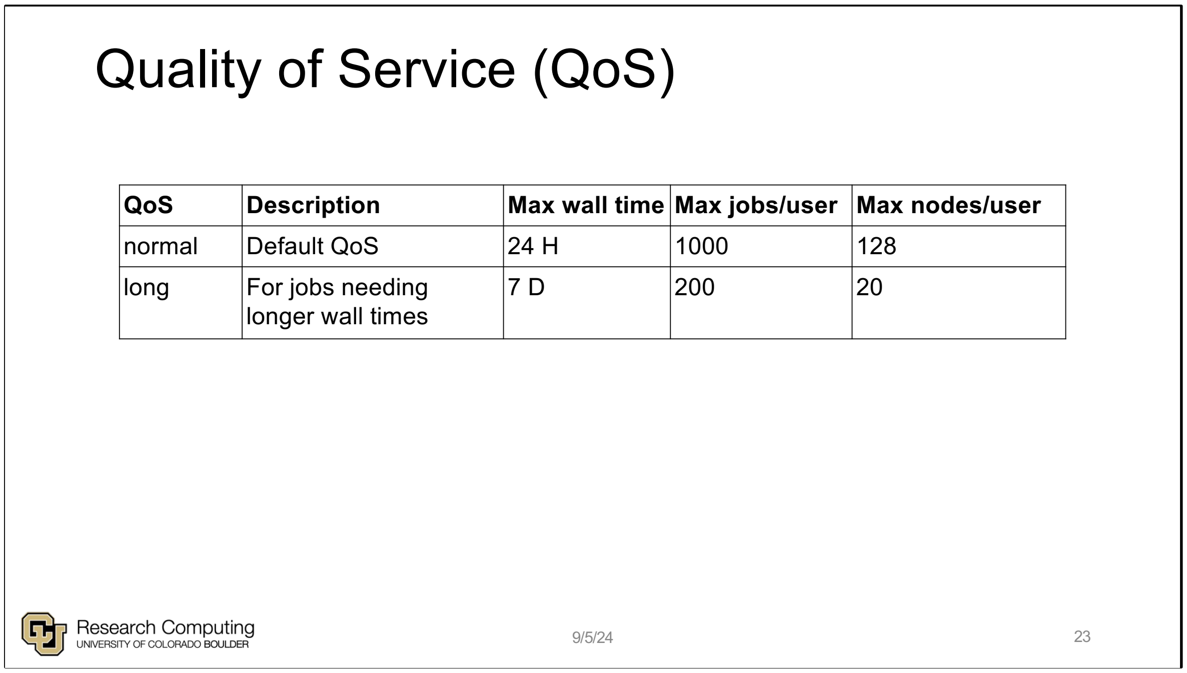
Detailed Information on Alpine Hardware:
https://curc.readthedocs.io/en/latest/clusters/alpine/alpine-hardware.html



Quality of Service (QoS)

QoS	Description	Max wall time	Max jobs/user	Max nodes/user
normal	Default QoS	24 H	1000	128

The normal qos is the default qos for all submitted jobs



Quality of Service (QoS)

QoS	Description	Max wall time	Max jobs/user	Max nodes/user
normal	Default QoS	24 H	1000	128
long	For jobs needing longer wall times	7 D	200	20

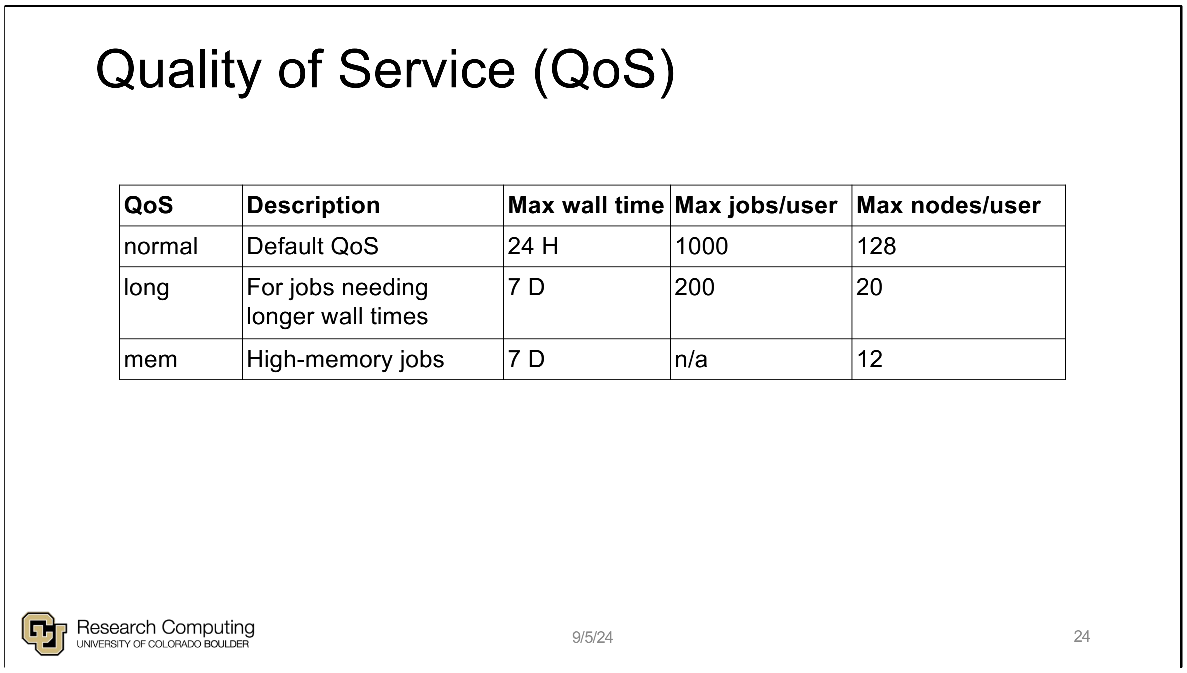
Research Computing
UNIVERSITY OF COLORADO BOULDER

9/5/24

23

The long qos provides a subset of amilan nodes to run longer jobs.

Please note! The long qos has fewer resources than normal and may have a longer wait time in the queue.



Quality of Service (QoS)

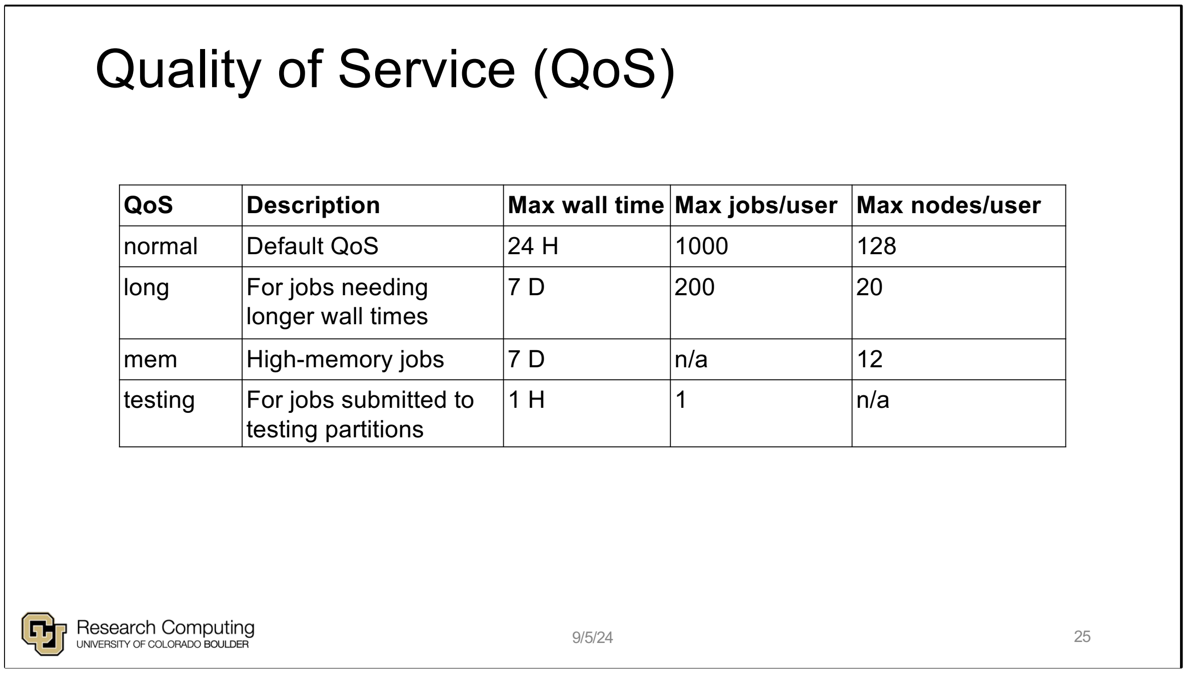
QoS	Description	Max wall time	Max jobs/user	Max nodes/user
normal	Default QoS	24 H	1000	128
long	For jobs needing longer wall times	7 D	200	20
mem	High-memory jobs	7 D	n/a	12

Research Computing
UNIVERSITY OF COLORADO BOULDER

9/5/24

24

The mem qos is specifically for jobs that run on the amem partition.



Quality of Service (QoS)

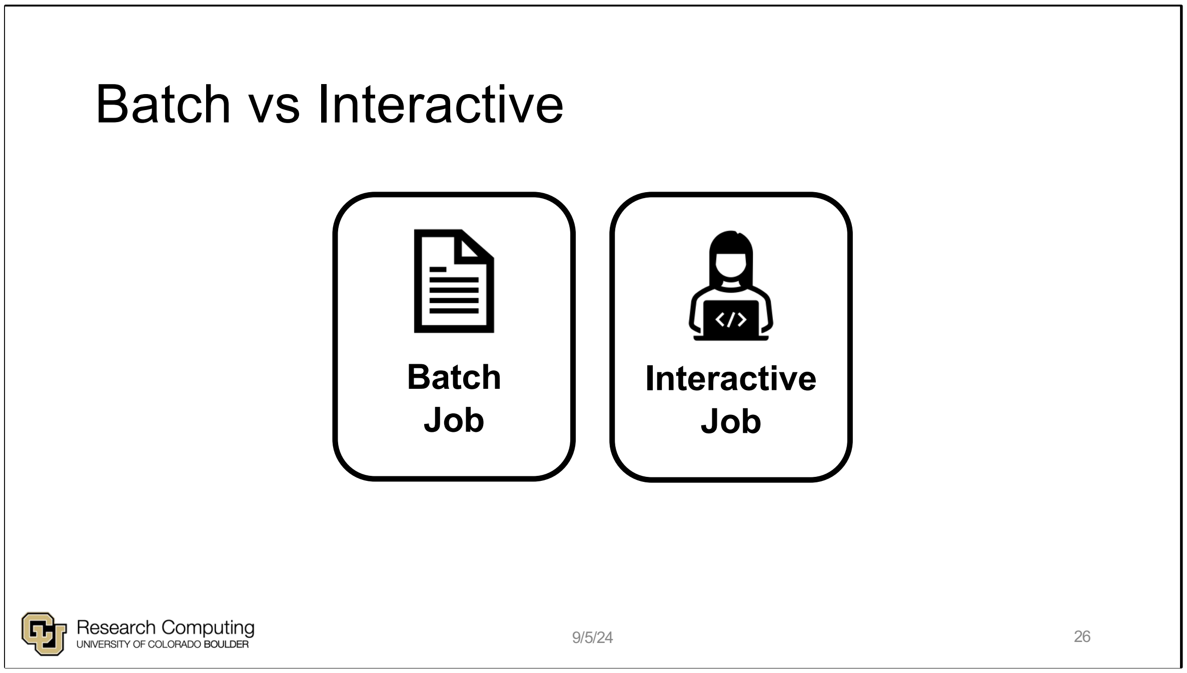
QoS	Description	Max wall time	Max jobs/user	Max nodes/user
normal	Default QoS	24 H	1000	128
long	For jobs needing longer wall times	7 D	200	20
mem	High-memory jobs	7 D	n/a	12
testing	For jobs submitted to testing partitions	1 H	1	n/a

Research Computing
UNIVERSITY OF COLORADO BOULDER

9/5/24

25

The testing qos supports the different testing partitions, ensuring that users can quickly access resources for testing their workflows. Keep in mind the stringent constraints – this partition is not intended for running regular jobs.



Batch vs Interactive

Batch Job

Interactive Job

Research Computing
UNIVERSITY OF COLORADO BOULDER

9/5/24

26

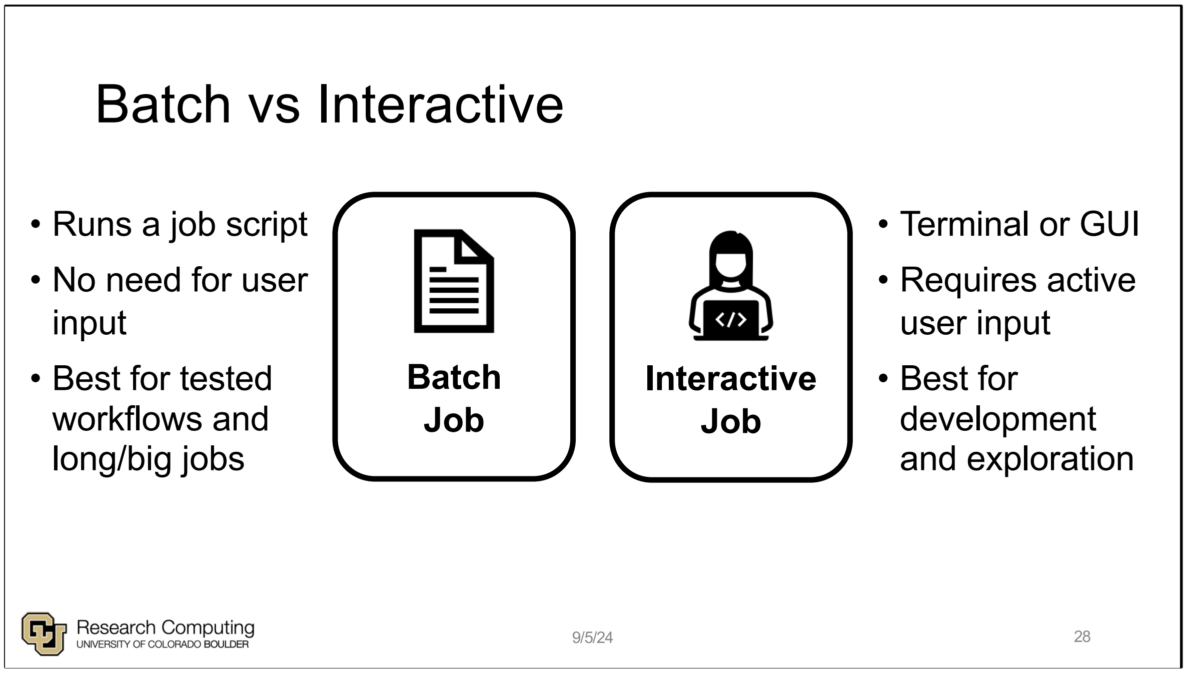
Now that you have a general idea of what resources you plan to request, you will need to consider the type of job you are wanting to schedule. A batch job or an interactive job.

Batch vs Interactive

- Runs a job script
- No need for user input
- Best for tested workflows and long/big jobs

Batch Job

Interactive Job



Batch vs Interactive

- Runs a job script
- No need for user input
- Best for tested workflows and long/big jobs

Batch Job

Interactive Job

- Terminal or GUI
- Requires active user input
- Best for development and exploration

Research Computing
UNIVERSITY OF COLORADO BOULDER

9/5/24

28

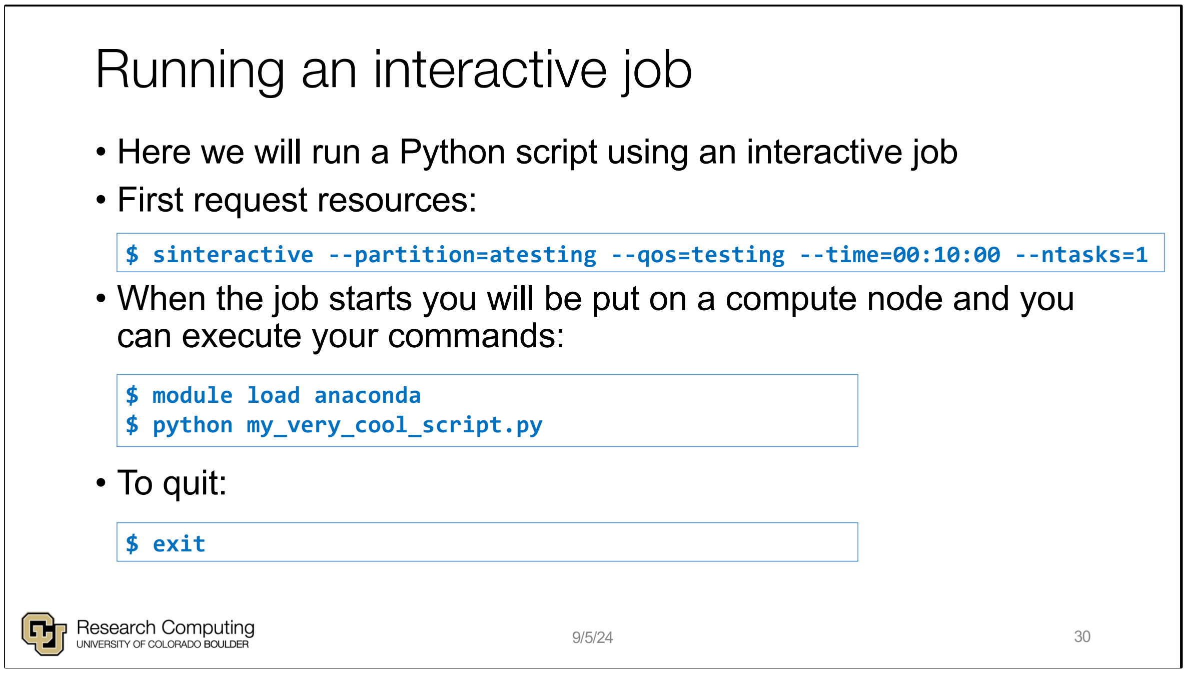
Interactive jobs

acompile

- Quick access to resources
- Limit of 4 CPUs for up to 12 hours
- Focused on compiling/building software and small workflows

`sinteractive`

- Request any partition, more resources, and longer walltime
- Subject to standard wait times (not immediate access)
- Focused on GUI-based software



Running an interactive job

- Here we will run a Python script using an interactive job
- First request resources:

```
$ sinteractive --partition=atesting --qos=testing --time=00:10:00 --ntasks=1
```

- When the job starts you will be put on a compute node and you can execute your commands:

```
$ module load anaconda
$ python my_very_cool_script.py
```

- To quit:

```
$ exit
```

Research Computing
UNIVERSITY OF COLORADO BOULDER

9/5/24

30

Batch Jobs

- **Batch Jobs** are jobs you submit to the scheduler that are run later without supervision
- A job script is simply a script that includes **SLURM directives** (resource specifics) ahead of any commands.

Anatomy of a job script

```
#!/bin/bash

## Directives
#SBATCH --<option>=<value>

## Software
module load <software>

## User scripting
<command>
```

Anatomy of a job script

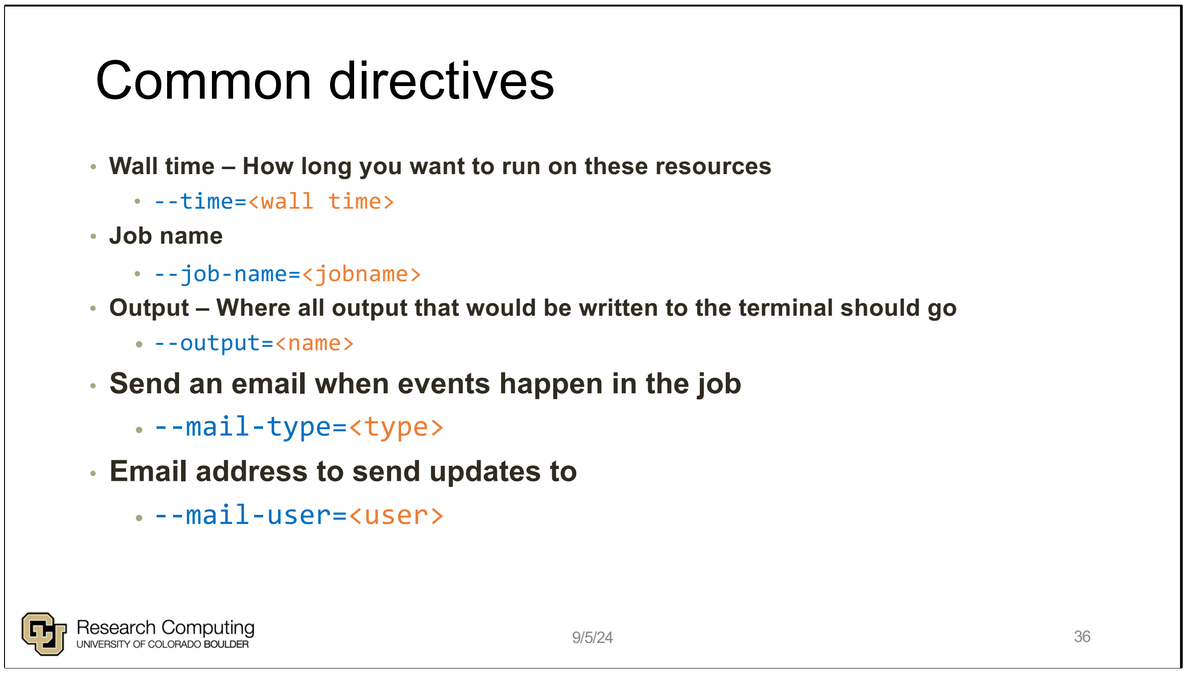
```
#!/bin/bash

## Directives
#SBATCH --<option>=<value>

## Software
module load <software>

## User scripting
<command>
```

Directives in a job script

```
#SBATCH --<option>=<value>
```

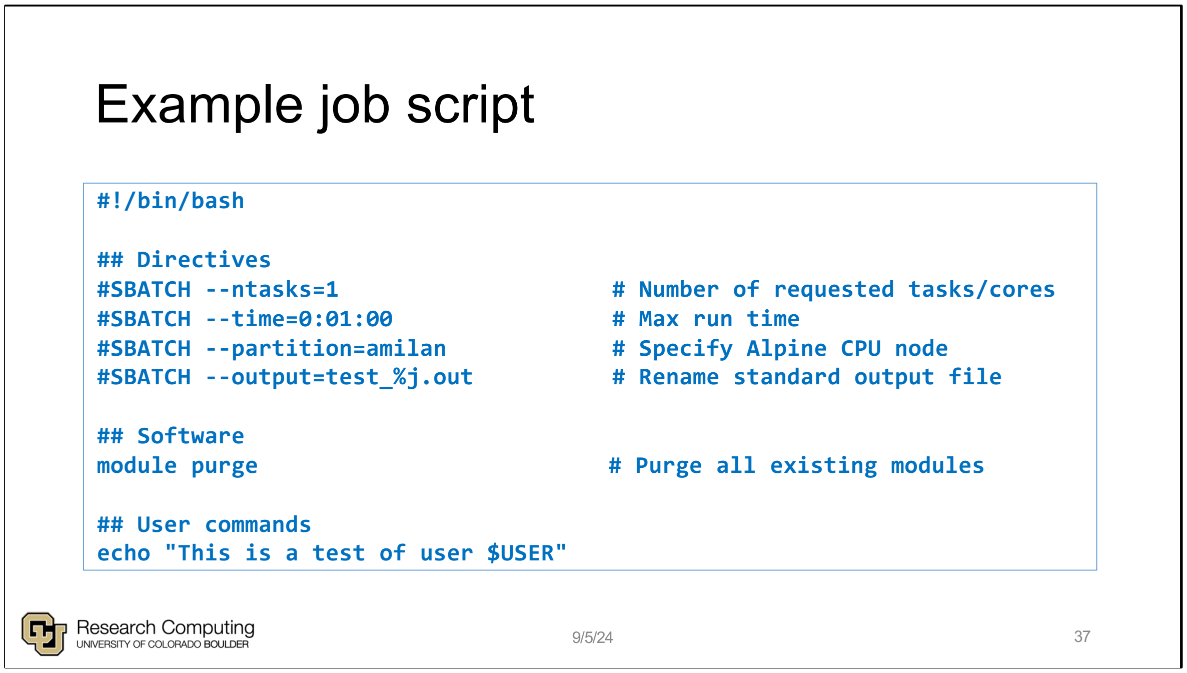
Common directives

- **Partition – A collection of compute nodes**
  - `--partition=<partition_name>`
- **Quality of service (QoS) – System defined constraints for a job (more on this later!)**
  - `--qos=<qos>`
- **Allocation – Account to “charge to”**
  - `--account=<account_name>`
- **Number of nodes to run on**
  - `--nodes=<nodes>`
- **Number of cores to run on**
  - `--ntasks=<number-of-tasks>`



Common directives

- Wall time – How long you want to run on these resources
  - `--time=<wall time>`
- Job name
  - `--job-name=<jobname>`
- Output – Where all output that would be written to the terminal should go
  - `--output=<name>`
- Send an email when events happen in the job
  - `--mail-type=<type>`
- Email address to send updates to
  - `--mail-user=<user>`



Example job script

```
#!/bin/bash

## Directives
#SBATCH --ntasks=1                    # Number of requested tasks/cores
#SBATCH --time=0:01:00                # Max run time
#SBATCH --partition=amilan            # Specify Alpine CPU node
#SBATCH --output=test_%j.out          # Rename standard output file

## Software
module purge                         # Purge all existing modules

## User commands
echo "This is a test of user $USER"
```

Research Computing
UNIVERSITY OF COLORADO BOULDER

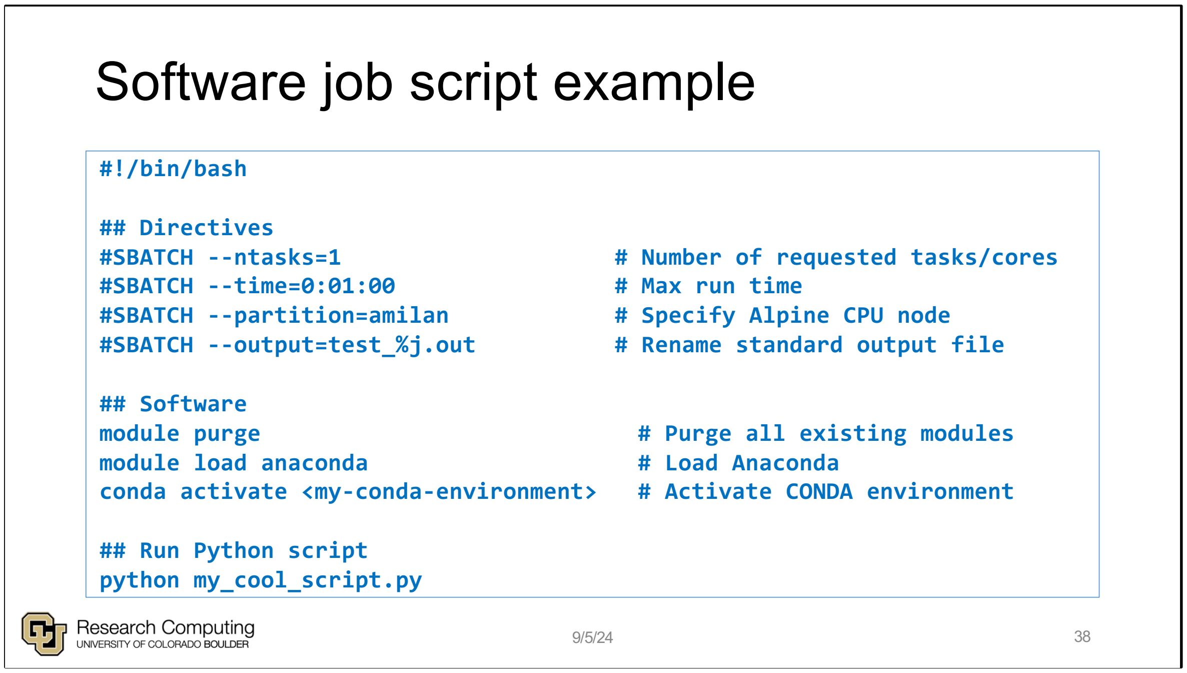
9/5/24

37

Output files are important for debugging and verifying a job has completed successfully. Always include a --output directive!

The output file is created in directory job was run unless specified in your --output directive.

If the directive --output is not provided, then a generic file name will be used (slurm_xxxxxx.out).



Software job script example

```
#!/bin/bash

## Directives
#SBATCH --ntasks=1                    # Number of requested tasks/cores
#SBATCH --time=0:01:00                # Max run time
#SBATCH --partition=amilan            # Specify Alpine CPU node
#SBATCH --output=test_%j.out          # Rename standard output file

## Software
module purge                            # Purge all existing modules
module load anaconda                    # Load Anaconda
conda activate <my-conda-environment>   # Activate CONDA environment

## Run Python script
python my_cool_script.py
```

Research Computing
UNIVERSITY OF COLORADO BOULDER

9/5/24

38

Software can be loaded using the software module system or by accessing user installed software in their /projects directory.

https://curc.readthedocs.io/en/latest/compute/modules.html

Be careful not to run GUI or other software that requires user interaction!

Submitting a Job script

```
sbatch <job_file> <other-directives>
```

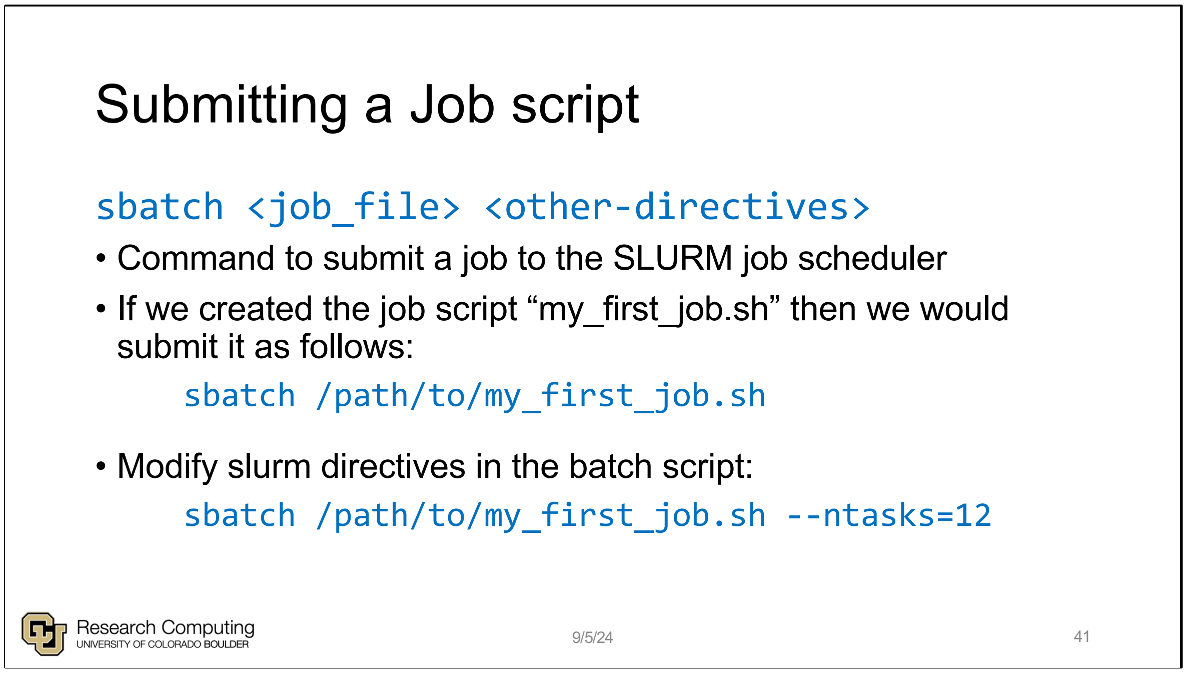
- Command to submit a job to the SLURM job scheduler

Submitting a Job script

```
sbatch <job_file> <other-directives>
```

- Command to submit a job to the SLURM job scheduler
- If we created the job script “my_first_job.sh” then we would submit it as follows:

```
    sbatch /path/to/my_first_job.sh
```



Submitting a Job script

```
sbatch <job_file> <other-directives>
```

- Command to submit a job to the SLURM job scheduler
- If we created the job script “my_first_job.sh” then we would submit it as follows:

```
sbatch /path/to/my_first_job.sh
```

- Modify slurm directives in the batch script:

```
sbatch /path/to/my_first_job.sh --ntasks=12
```

Checking your jobs

- squeue: Monitor your jobs status **in queue and while running**:
  - By default, shows all jobs in queue can specify using:

```
$ squeue -u <username>
$ squeue -p <partition>
```

- sacct: Check back on usage statistics of **previous Jobs**
  - By default, only checks all jobs from the start of the current day can specify using:

```
$ sacct -u <username>
$ sacct --start=MM/DD/YY -u <username>
$ sacct -j <job-id>
```

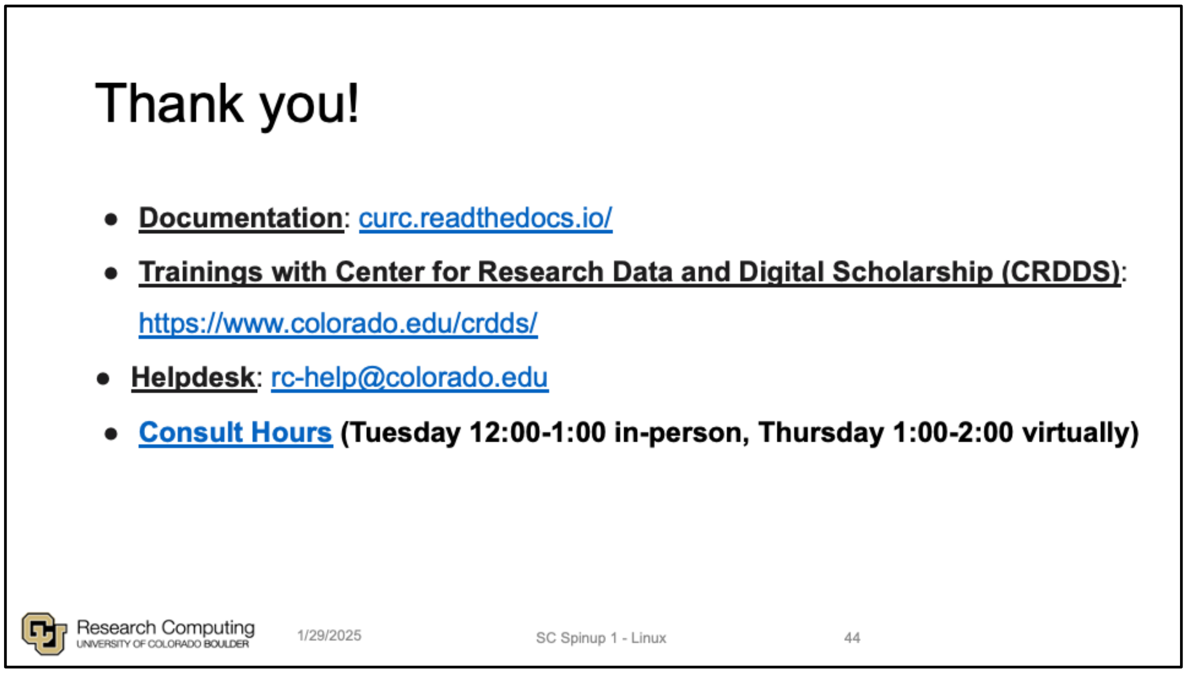
Checking your jobs

- To check the percentage of CPU and memory usage of a job **after it completes,** use seff

```
$ module load slurmtools
$ seff <job number>
```

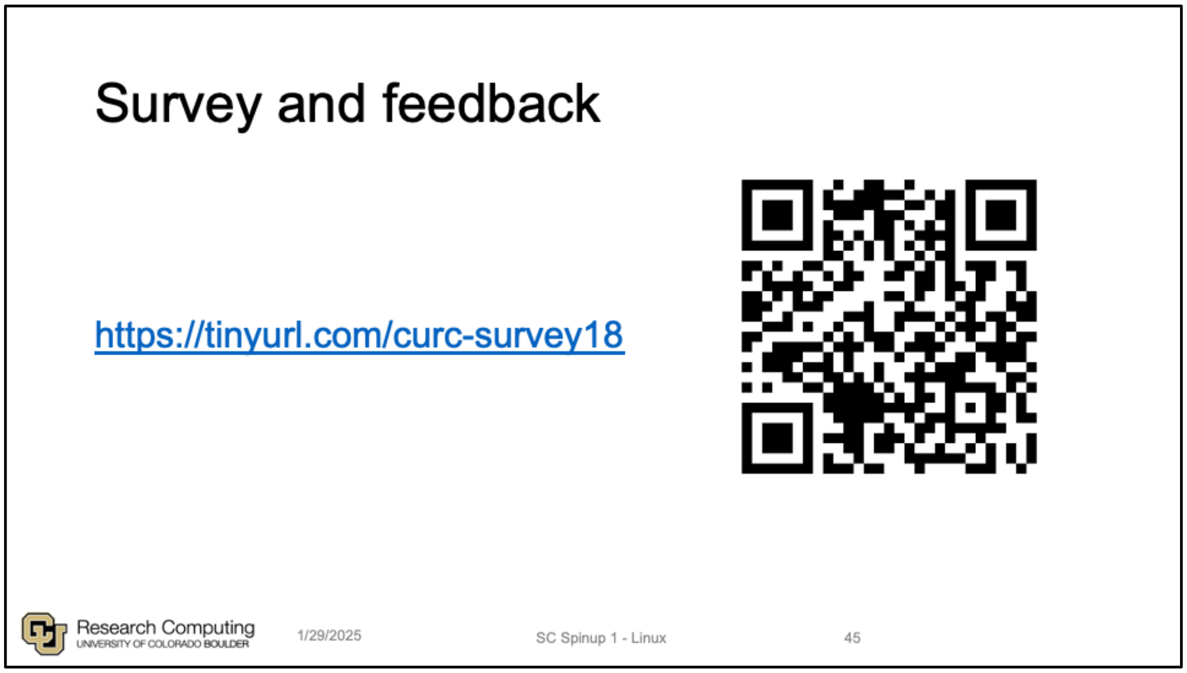
- To see general statistics for your submitted jobs, use jobstats

```
$ module load slurmtools
$ jobstats <username> <# of days>
```



Thank you!

- **Documentation**: curc.readthedocs.io/
- **Trainings with Center for Research Data and Digital Scholarship (CRDDS)**: https://www.colorado.edu/crdds/
- **Helpdesk**: rc-help@colorado.edu
- **Consult Hours (Tuesday 12:00-1:00 in-person, Thursday 1:00-2:00 virtually)**



Survey and feedback

https://tinyurl.com/curc-survey18

Research Computing
UNIVERSITY OF COLORADO BOULDER 1/29/2025 SC Spinup 1 - Linux 45