HPRC Primer

Using the Slurm Scheduler on the ACES Cluster

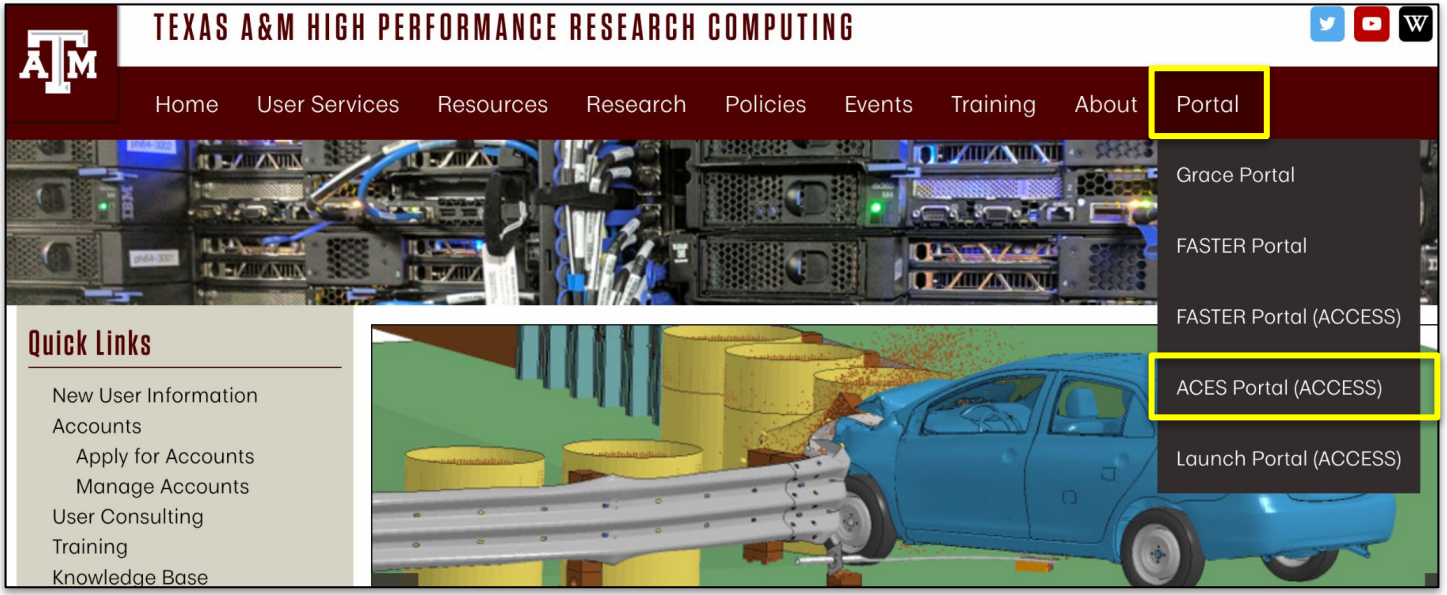
March 07, 2025

Gabriel Floreslovo

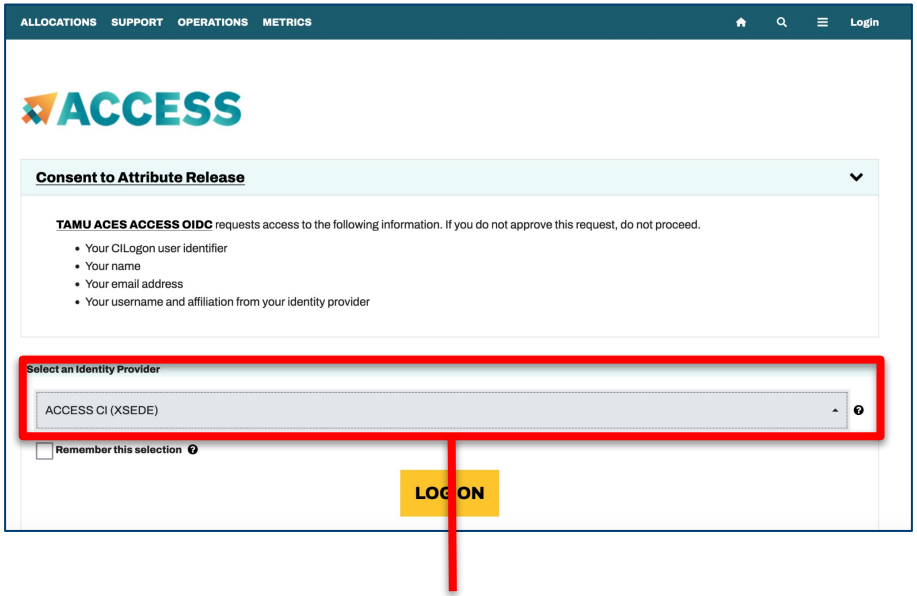
High Performance
Research Computing
DIVISION OF RESEARCH

Accessing the ACES Portal

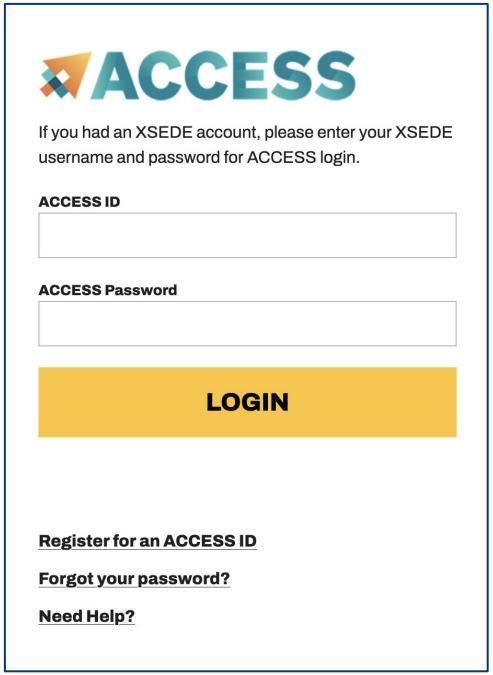
- HPRC webpage: hprc.tamu.edu
- Aces portal shortcut: portal-aces.hprc.tamu.edu
- Requires an ACCESS ID!



Accessing ACES via the Portal (ACCESS)



Select the Identity Provider appropriate for your account.

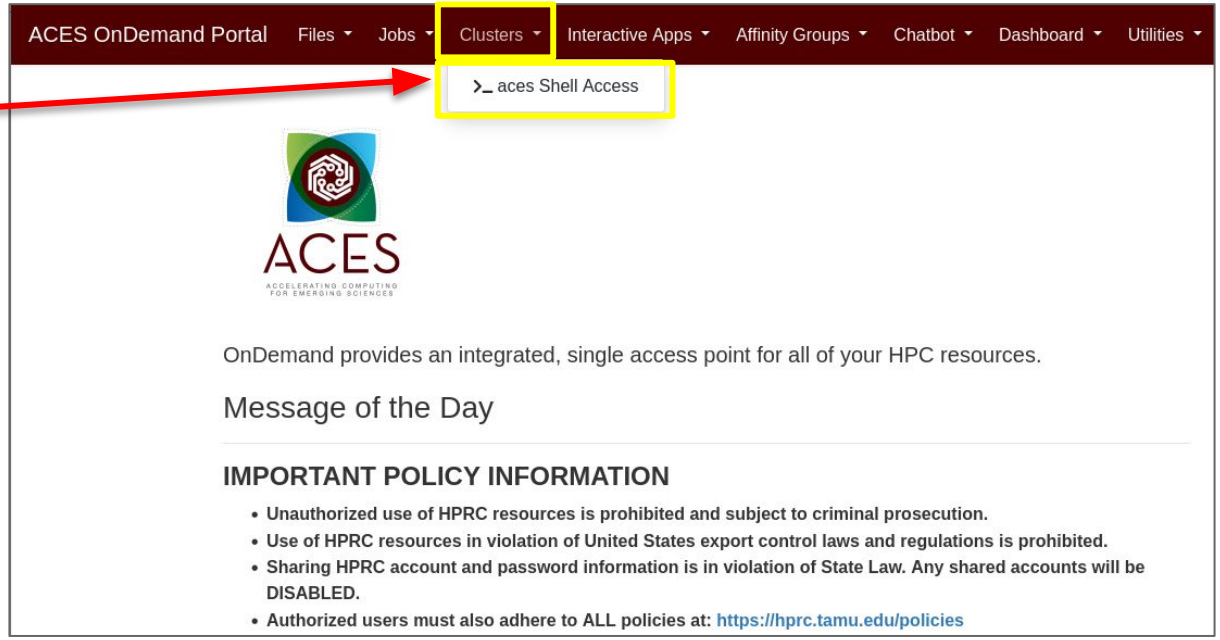


Log-in using your ACCESS or institutional credentials.

Shell Access via the HPRC Portal

Once in the Portal, select at the top: “Clusters” → “aces Shell Access”

- *shell* is also called *terminal* or *command line*

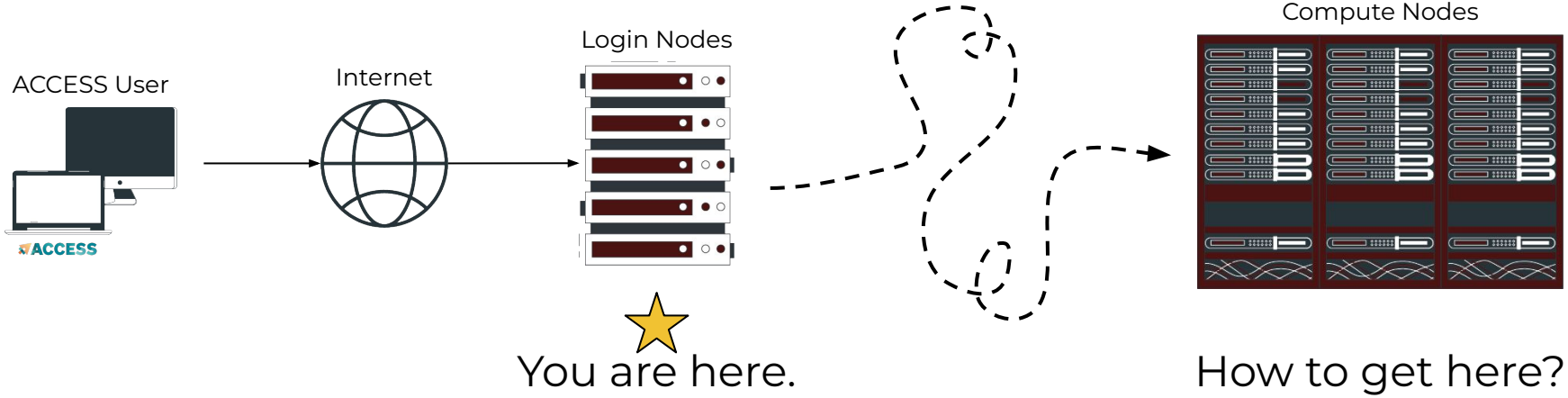


Hands-On Activity - 2 Minutes

- Connect to ACES now through the portal using portal-aces.hprc.tamu.edu
- Get a shell on the ACES cluster from the Clusters menu

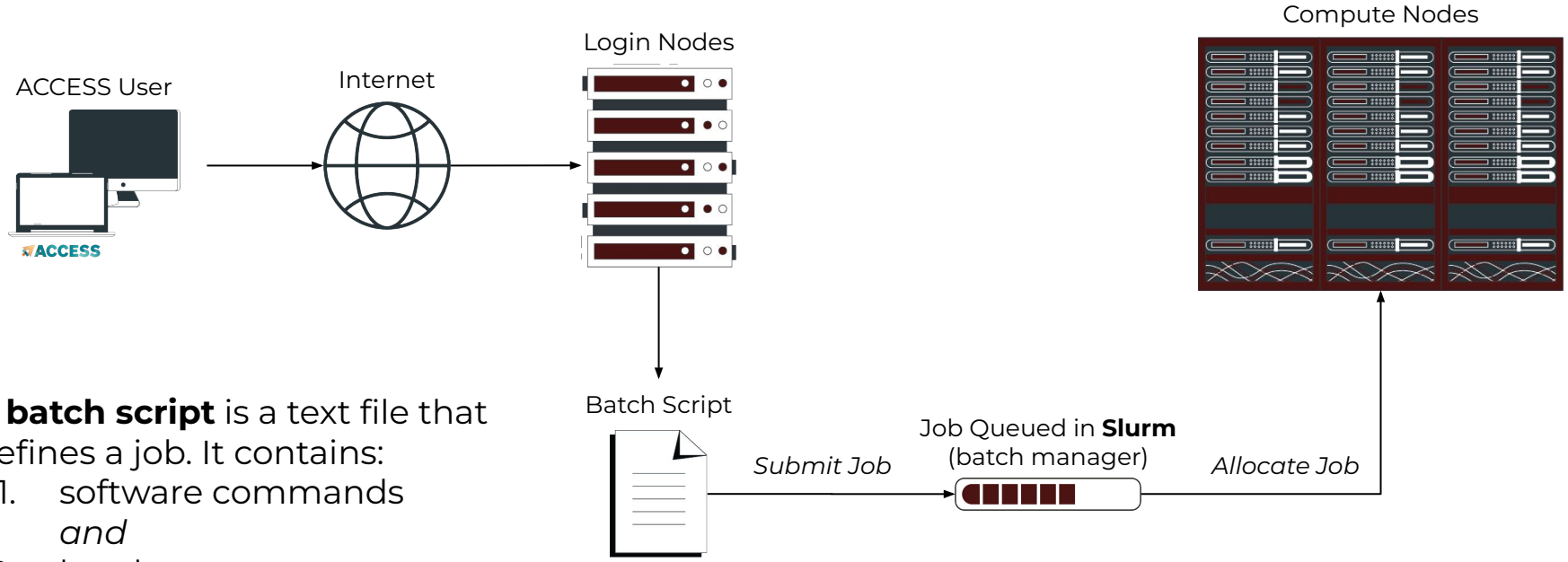
What is the hostname of the machine you connected to?

Batch Computing on the Clusters



- Types of nodes:
  - Login node - a shared machine for light editing
  - Compute node - an allocated machine for heavy computation

Batch Computing on ACES cluster



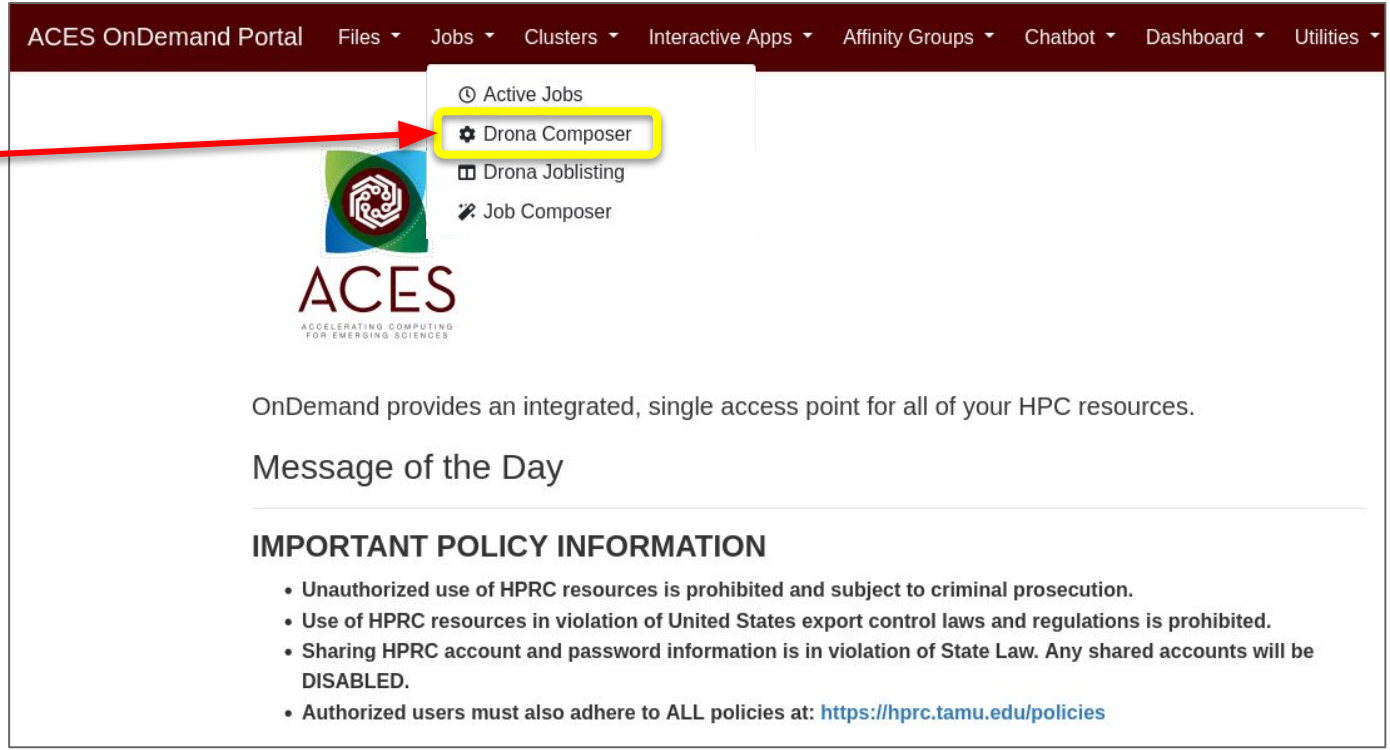
A **batch script** is a text file that defines a job. It contains:

1. software commands
   and
2. batch parameters

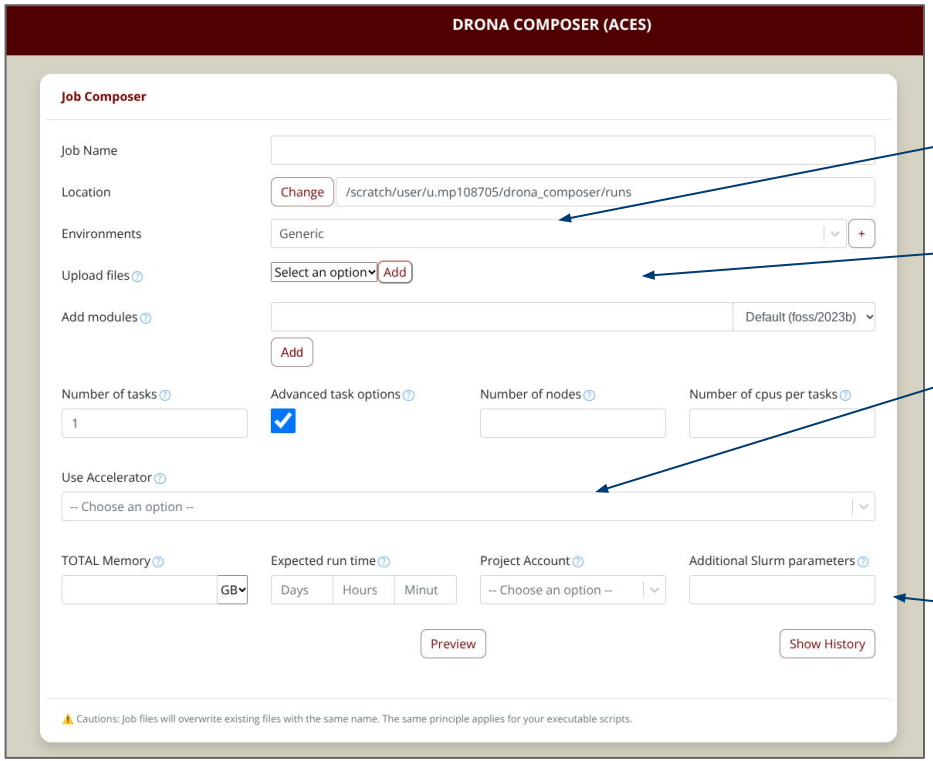
The Drona Composer

- A framework to assist you with generating and creating jobs

Once in the Portal, select at the top: “Jobs” → “Drona Composer”



DRONA COMPOSER - JOB SUBMISSION MADE EASY



Select environment.

Optionally add modules.

Select choice of accelerator.

Set optional job parameters (or let Drona pick defaults).

Sample Job Script Structure

```
#!/bin/bash

##NECESSARY JOB SPECIFICATIONS
#SBATCH --job-name=JobExample1
#SBATCH --time=01:30:00
#SBATCH --ntasks-per-node=48
#SBATCH --nodes=2
#SBATCH --mem=488G
#SBATCH --output=Example1Out.%j
#SBATCH --partition=gpu
#SBATCH --gres=gpu:h100:1

##OPTIONAL JOB SPECIFICATIONS
##SBATCH --account=123456
##SBATCH --mail-type=ALL
##SBATCH --mail-user=email_address

# load required module(s)
module purge
module load GCCcore/13.3.0
module load Python/3.12.3
module load CUDA/12.6.0

# Run your program
python my_gpu_script.py
```

This is a single-line comment and not run as part of the script.

These parameters describe your job to the job scheduler.
The lines starting with #SBATCH are NOT comments!
See the Knowledge Base for more info.

Whatever commands or scripts you want to run.
Here, we set up the modules we need for our environment and run a python program.

Hands on Activity - 5 minutes

1. Create a directory in **$SCRATCH** called **drona_composer**:
```
cd $SCRATCH
mkdir drona_composer
mkdir environments
```

2. Copy **MyGeneric** in **/scratch/training/slurm_scheduler**:
```
cd $SCRATCH/drona_composer/environments/
cp -r /scratch/training/slurm_scheduler/MyGeneric
```

3. Customize your environment by editing the **schema.json** file inside **MyGeneric**:
```
cd MyGeneric
vi schema.json
```

4. See the changes after reloading the Drona Composer page.

Important Batch Job Parameters

Slurm	Comment
`#SBATCH --time=HH:MM:SS`	Specifies the time limit for the job. Must specify seconds SS on ACES
`#SBATCH --ntasks=x`	Total number of tasks (cores) for the job.
`#SBATCH --ntasks-per-node=xx`	Specifies the maximum number of tasks (cores) to allocate per node

`#SBATCH --mem=xxxxM`
`or`
`#SBATCH --mem=xG`
	Sets the maximum amount of memory (MB) per *node*.
	G for GB is supported on ACES

`#SBATCH --nodes=x`	Specifies the number of nodes to use

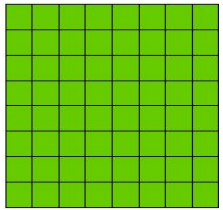
(These go in your job script file)

Mapping Jobs to Cores per Node on ACES

A.

allocated cores

unused cores

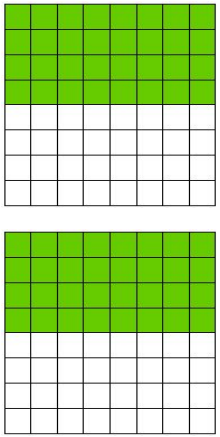


96 cores on
1 compute node

```
#SBATCH --ntasks=96
#SBATCH --ntasks-per-node=96
```

Preferred Mapping
(if applicable)

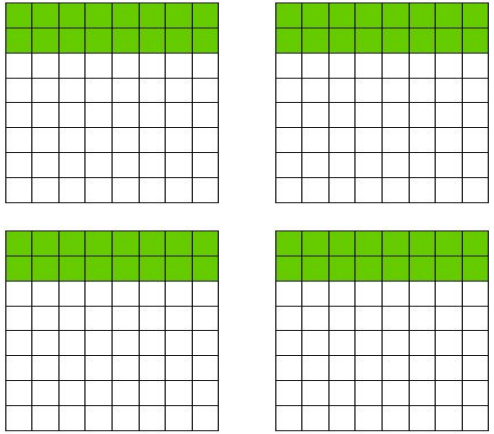
B.



96 cores on
2 compute nodes

```
#SBATCH --ntasks=96
#SBATCH --ntasks-per-node=48
```

C.



96 cores on
4 compute nodes

```
#SBATCH --ntasks=96
#SBATCH --ntasks-per-node=24
```

Pop Quiz

```
#SBATCH --job-name=JobExample2
#SBATCH --time=48:00:00
#SBATCH --ntasks=96
#SBATCH --ntasks-per-node=24
#SBATCH --nodes=4
#SBATCH --mem=48G
#SBATCH --output=stdout.%J
#SBATCH --error=stderr.%J
```

How many cores is this job requesting?

A. 1024
B. 96
C. 960
D. 4

Job Submission and Tracking

Slurm	Description
`sbatch jobfile`	Submit jobfile to batch system
`squeue [--me] [-j job_id]`	List jobs
`scancel job_id`	Kill a job
`sacct -X -j job_id`	Show information for a job (can be when job is running or recently finished)
`sacct -X -S YYYY-MM-DD`	Show information for all of your jobs since YYYY-MM-DD

(These are command-line commands, not part of your job script)

Job Environment Variables

Each job has access to several self-referential variables:

- **`$SLURM_JOBID`** = job id
- **`$SLURM_SUBMIT_DIR`** = directory from where job was submitted
- **`$TMPDIR`** = high speed 1.5TB disk on the job's compute node

You can also use non-Slurm variables such as:

- **`$HOME`** = /home/username
- **`$SCRATCH`** = /scratch/user/username

https://hprc.tamu.edu/kb/Helpful-Pages/Batch-Translation/#environment-variables

Submit a Job and Check Job Status

Submit job

```
sbatch example01.slurm
```

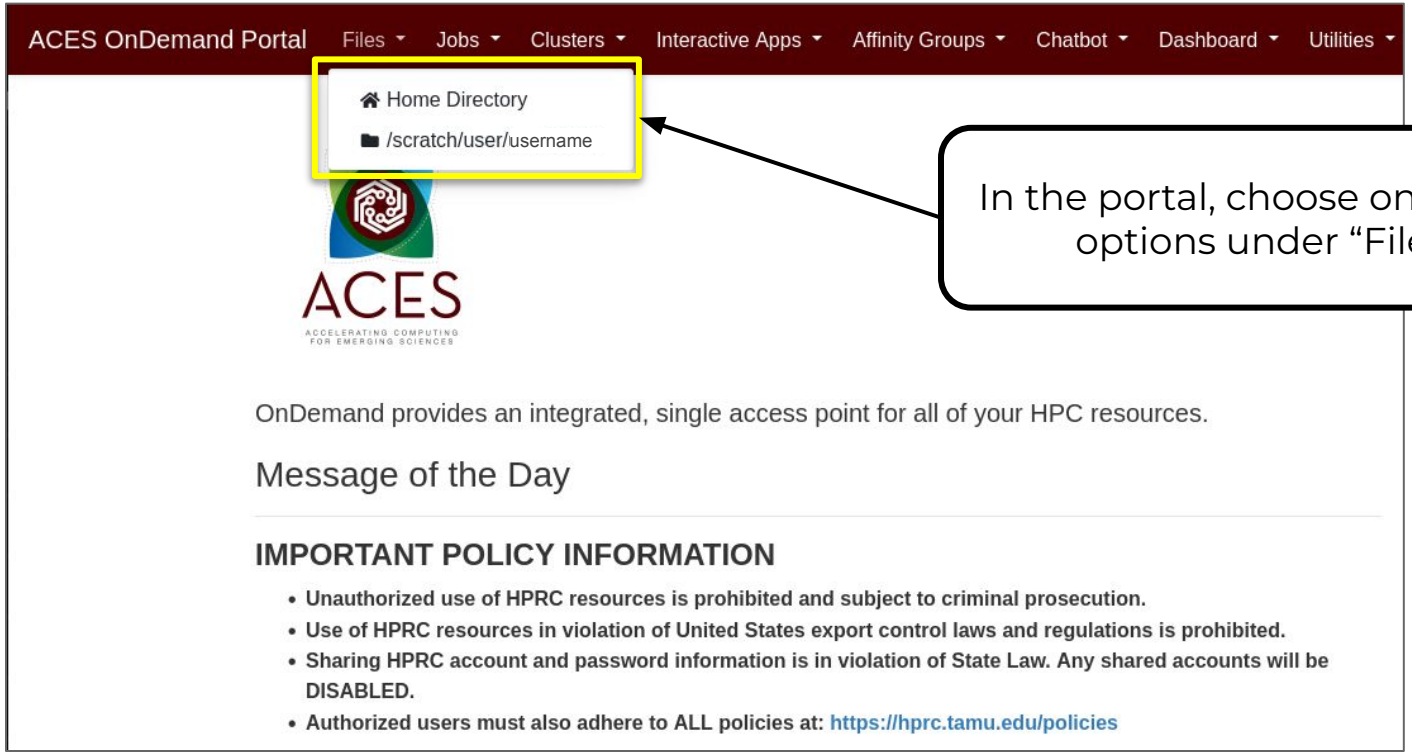
```
sbatch: Job submitted using account 1234567891011
sbatch: Current account balance = 9794.371667
Submitted batch job 982272
```

Check status

```
squeue --me
```

```
JOBID    NAME      USER      PARTITION  NODES  CPUS  STATE    TIME       TIME_LEFT  START_TIME          REASON     NODELIST
982272   jobname   someuser   cpu       1      96    RUNNING  0:15       23:59:45   2025-03-06T11:49    None       ac058
```

Open the File Navigator



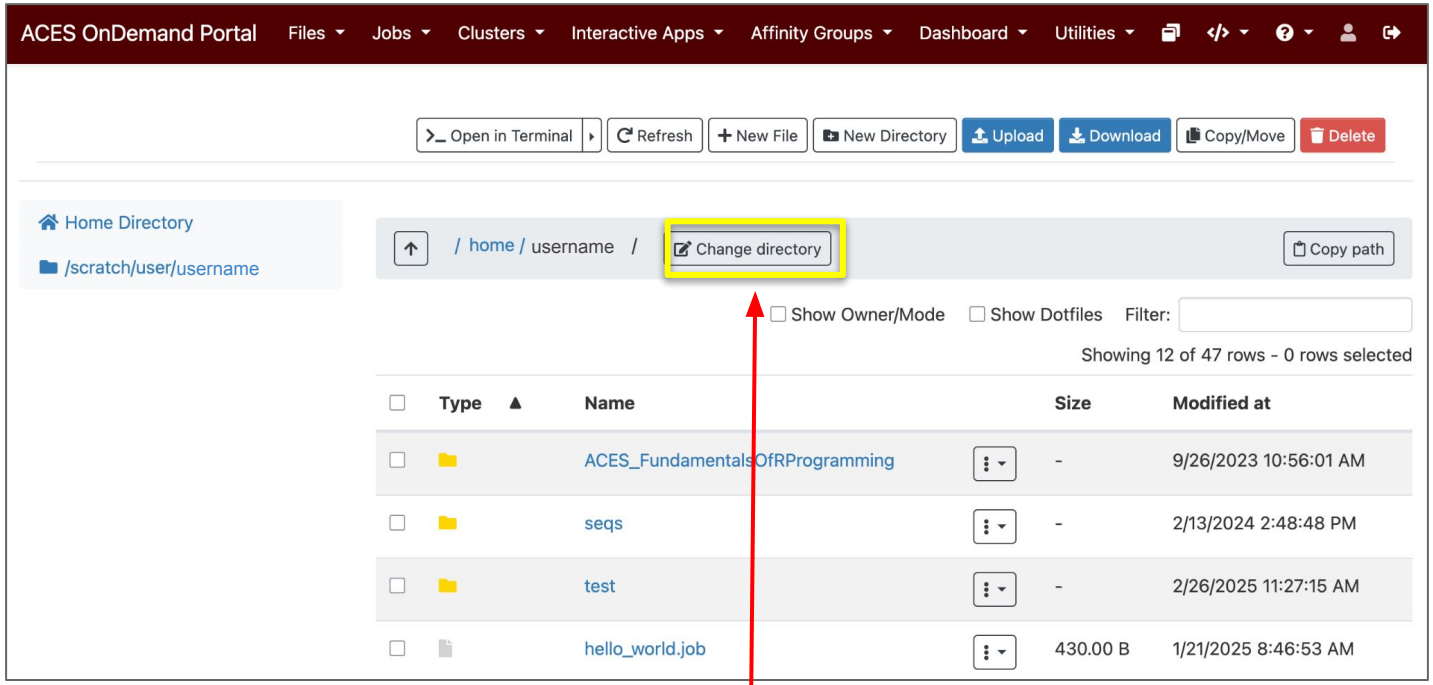
Hands-On Activity- 10 mins

There are example files located at `/scratch/training/slurm_scheduler`

1. Copy these files to your home directory
2. Edit a batch file.
3. Submit a batch file using `sbatch`.
4. Check that the job is running in a Slurm queue with `squeue`.
5. Check the contents of the output file.

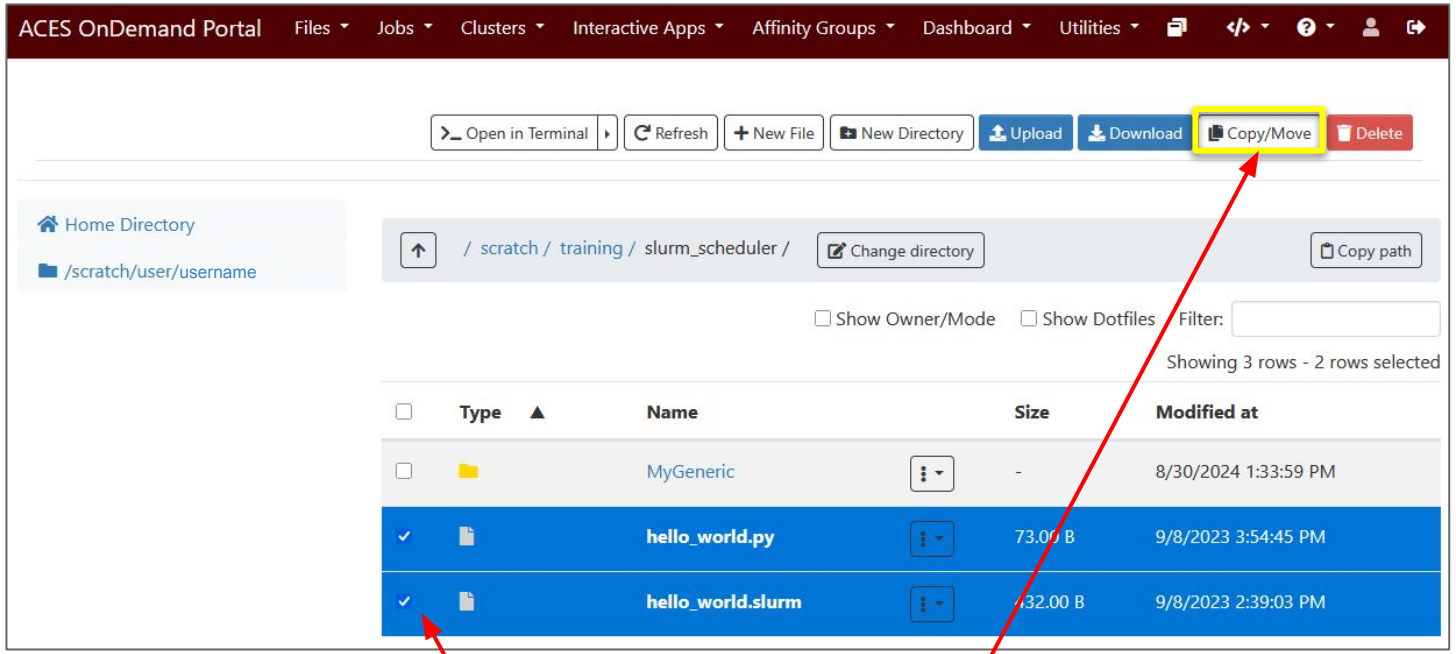
We are going to see above steps one-by-one in action in following slides.

Navigating to Training Directory



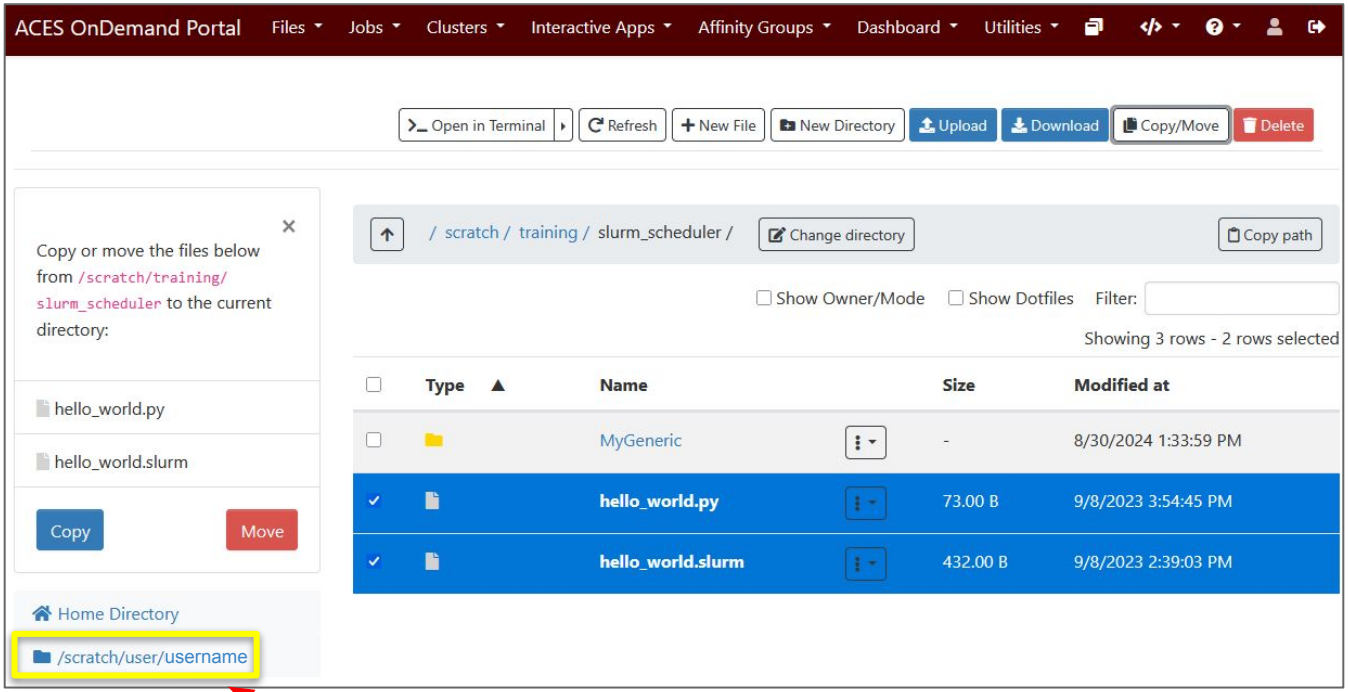
Click on “change directory” and type
/scratch/training/slurm_scheduler

Copy the Example Files



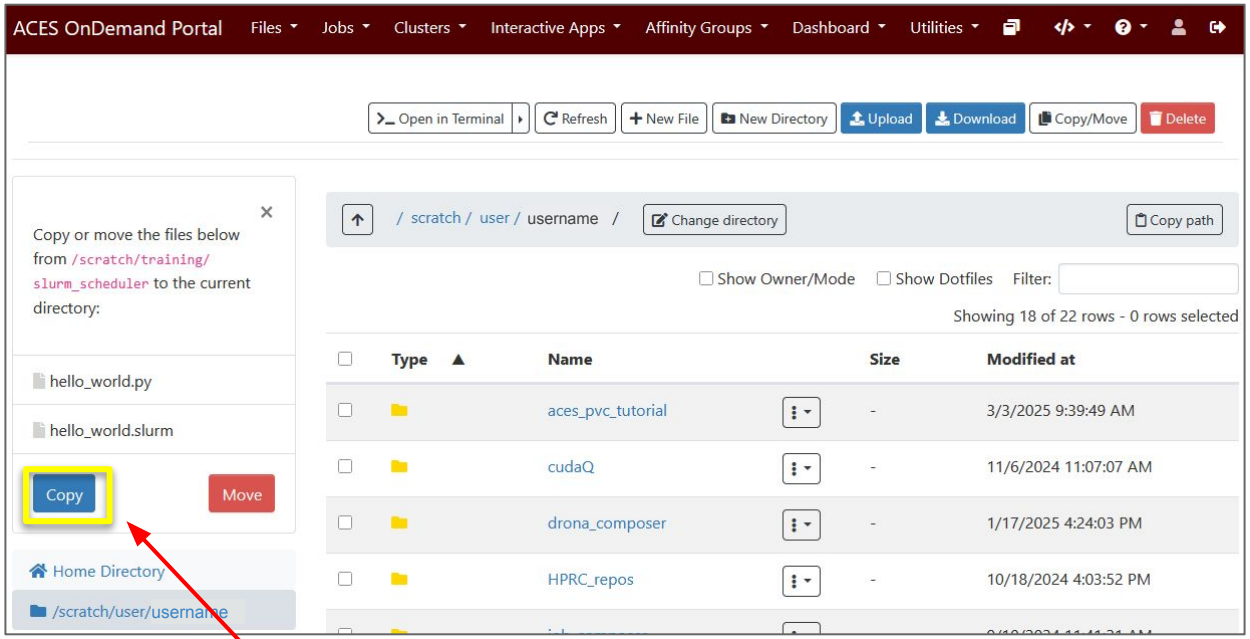
Check box the two files and click Copy/Move

Move to Scratch



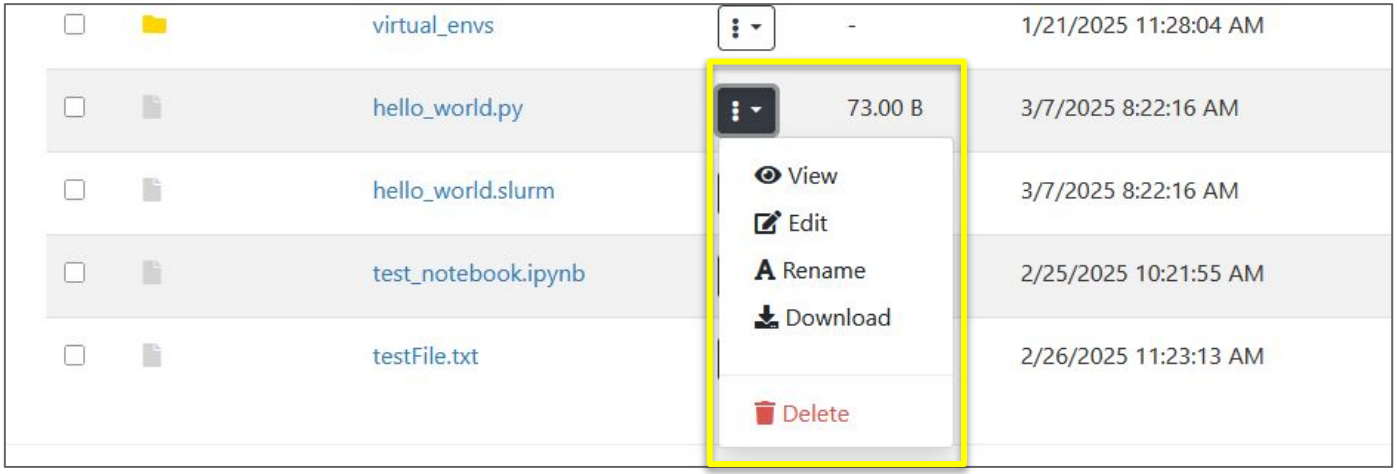
Select the directory to copy to

Paste the Example Files

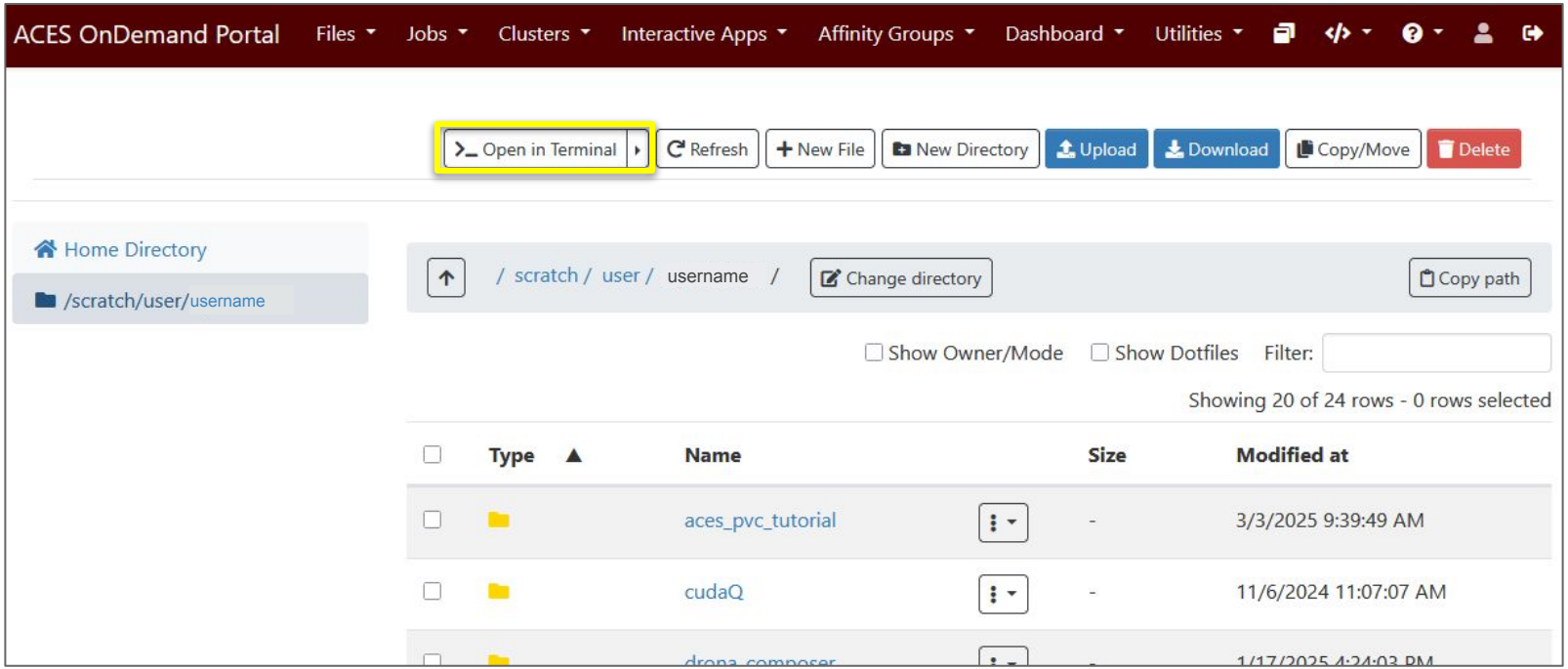


The files on the right will change to show the directory you chose. Hit “Copy” to actually copy the files to that directory.

View and Edit the Example Files



Open a Terminal (another option)



Hands on Activity- Cont.

1. Investigate the two example files
   - Make a small edit to personalize. (e.g., “Hello MyName”)
2. Submit the batch file using:
   ```
   sbatch hello_world.slurm
   ```
3. Check the job status using:
   ```
   squeue -u $USER
   ```
4. Once the job is completed, inspect the output file:
   hello_world_log.<job_id>

Consumable Computing Resources

- Resources which we can specify in a job/slurm file:
  - Processor cores
  - Memory
  - Walltime
  - GPU and other accelerators
- Other resources:
  - SUs
  - Software license/token
    - Use `license_status` to query
    - License Checker: hprc.tamu.edu/kb/Software/useful-tools/License_Checker/

Find available license for "FEMZIP":

```
license_status -s matlab
```

```
License status for Matlab:
--------------------------------------------------------------------
|License Name                | # Issued| # In Use | # Available  |
--------------------------------------------------------------------
|Matlab                      |    50   |     0    |     50       |
--------------------------------------------------------------------
```

More information on this command:

```
license_status -h
```

Key: *Your inputs in green*; *Terminal output in blue*

How Does Slurm Assign Jobs?

- Job submissions are auto-assigned to batch *queues* (also called *partitions*) based on the resources requested
  - number of cores/nodes and walltime limit
  - specific resources requested
- Some jobs can be directly submitted to a queue:
  - If gpu nodes are needed, use the gpu partition/queue: #SBATCH --partition=gpu

https://hprc.tamu.edu/kb/User-Guides/Common/BatchProcessing/#batch-queues

sinfo: Info for Node/Partition

To check the status of the nodes/partitions:

```
sinfo
```

```
PARTITION       AVAIL  TIMELIMIT    JOB_SIZE  NODES(A/I/O/T)    CPUS(A/I/O/T)
cpu*            up     3-00:00:00   1-64      53/1/0/54         3984/1200/0/5184
gpu             up     2-00:00:00   1-8       5/2/0/7           336/336/0/672
gpu_debug       up     2:00:00      1         0/3/0/3           0/288/0/288
pvc             up     2-00:00:00   1-30      0/27/5/32         0/2592/480/3072
bittware        up     2-00:00:00   1         0/0/2/2           0/0/192/192
memverge        up     2-00:00:00   1         0/6/4/10          0/576/384/960
nextsilicon     up     2-00:00:00   1         0/2/0/2           0/192/0/192
staff           up     2-00:00:00   1-110     58/39/11/108      4320/4992/1056/10368
```

For the NODES and CPUS columns:

- A = Active (in use by running jobs)
- I = Idle (available for jobs)
- O = Offline (unavailable for jobs)
- T = Total

pestat : Processor Status

- **pestat** allows you to check the status of the nodes on ACES
- -p allows you to show a specific partition
- Example:

```
pestat -p gpu -G
```

```
Print only nodes in partition gpu
GPU GRES (Generic Resource) is printed after each JobID
Hostname  Partition Node  Num CPU  CPUload  Memsize  Freemem  GRES/node              Joblist
                    State Use/Tot  (15min)  (MB)     (MB)                            JobID User GRES/job ...
ac041         gpu  alloc  96   96   1.95*   500000   491133   gpu:h100:8(S:0)   649110 u.rl117197 gpu:h100=6
ac045         gpu  alloc  96   96   1.92*   500000   489023   gpu:h100:8(S:0)   649110 u.rl117197 gpu:h100=6
ac049         gpu  mix    24   96  18.64*   500000   277410   gpu:h100:4(S:0)   750075 u.xw127610 gpu:h100=8 *
ac055         gpu  alloc  96   96   1.94*   500000   462834   gpu:h100:4(S:0)   649110 u.rl117197 gpu:h100=6
ac064         gpu  idle   0    96   0.00    500000   508735   gpu:a30:2(S:0)
ac065         gpu  idle   0    96   0.00    500000   510542   gpu:a30 1g.6gb:8(S:0)
ac096         gpu   mix   24   96   18.95*  500000   276939   gpu:h100:4(S:0)   750075 u.xw127610 gpu:h100=8 *
```

Job Memory Requests on ACES

- If you request more resources than is allowed, Slurm will reject the job
- To check the maximum requestable amount, use: **maxconfig**

```
  ACES partitions:   cpu  gpu  gpu_debug  pvc  bittware  memverge  nextsilicon
  ACES GPUs in gpu partition:  a30:2  h100:2  h100:4  h100:8  pvc:2  pvc:4  pvc:8

  Showing max parameters (cores, mem, time) for partition cpu

  CPU-billing * hours * nodes =   SUs
           96 *  72 *  1 = 6,912

#!/bin/bash
#SBATCH --job-name=my_job
#SBATCH --time=3-00:00:00
#SBATCH --nodes=1           # max 64 nodes for partition cpu
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=96
#SBATCH --mem=488G
#SBATCH --output=stdout.%x.%j
#SBATCH --error=stderr.%x.%j
```

Check your Service Unit (SU) Balance

- List the SU Balance of your Account(s).

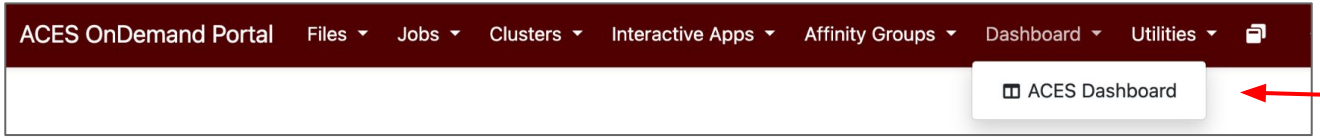
```
myproject -m
```

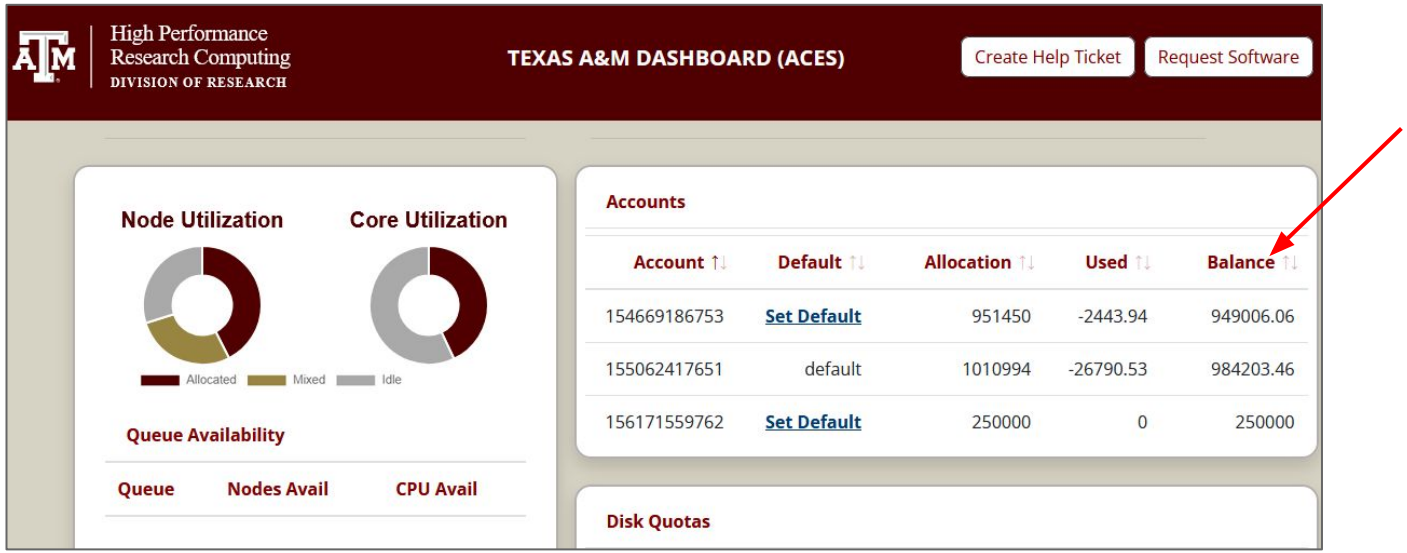
```
 ================================================================================================
               List of u.lp46691's Project Accounts
------------------------------------------------------------------------------------------------
|  Account    |  FY   | Default | Allocation |Used & Pending SUs|   Balance   |          PI         |
------------------------------------------------------------------------------------------------
|154669186753|  2025|        Y|   951450.00|          -2443.94|   949006.06|Perez, Lisa          |
------------------------------------------------------------------------------------------------
|155466915524|  2025|        N|   602891.00|             -3.51|   602887.49|Perez, Lisa          |
------------------------------------------------------------------------------------------------
============================
   Account       Project
----------------------------
 154669186753   TRA220029
 155466915524   TRA240008
----------------------------
```

- Run "myproject -d *Account#*" to change default project account.
- Run "myproject -h" to see more options.

https://hprc.tamu.edu/kb/User-Guides/AMS/#service-unit
https://hprc.tamu.edu/kb/User-Guides/AMS/#ams-user-interfaces

Check your Service Unit (SU) Balance





SU Charging Scheme

Resource	Service Units (per hour)	ACCESS Credits (per hour)
Intel SPR / Icelake	1	0.1
NVIDIA H100 and A30	128	16
Intel PVC GPUs	60	7.5
Bittware Agilex FPGA	100	12.5
Intel D5005 FPGA	50	6.25
NEC Vector Engine	150	18.75
NextSilicon coprocessor	100	12.5
Graphcore IPU Classic	90	11.25
Graphcore IPU Bow	120	15
Intel Optane Memory	60	7.5

8 SUs on ACES = 1 ACCESS credits

Continued Learning

HPRC YouTube

HPRC Homepage

ACES Quick Start Guide

ACES Portal (ACCESS users)

help@hprc.tamu.edu

HIGH PERFORMANCE RESEARCH COMPUTING
TEXAS A&M UNIVERSITY

https://hprc.tamu.edu

HPRC Helpdesk:

help@hprc.tamu.edu
Phone: 979-845-0219

Take our short course survey!

HPRC Survey

https://u.tamu.edu/hprc_shortcourse_survey

Help us help you. Please include details in your request for support, such as, Cluster (ACES, FASTER, Grace, Launch), NetID (UserID), Job information (JobID(s), Location of your jobfile, input/output files, Application, Module(s) loaded, Error messages, etc), and Steps you have taken, so we can reproduce the problem.