



Ilifu Online Training

Oarabile Hope Moloko

Astronomy Support Specialist , IDIA, Department of Astronomy,
University of Cape Town, September 2022




UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD



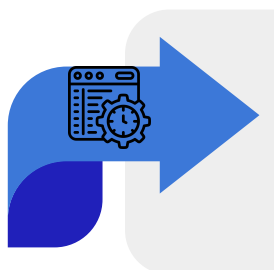

Inter-University Institute
for Data Intensive Astronomy





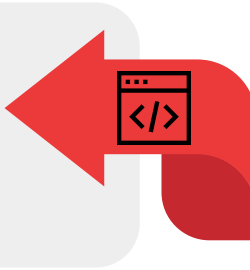
Login node
Run SLURM & bash commands
cd, mkdir, ls, etc

Jupyter/Dev. node
Development space
New code / workflows / routines
Debugging / testing software



Main partition
Stable, computationally
heavy processing

HighMem/GPU
For single-high memory
jobs or GPU resources

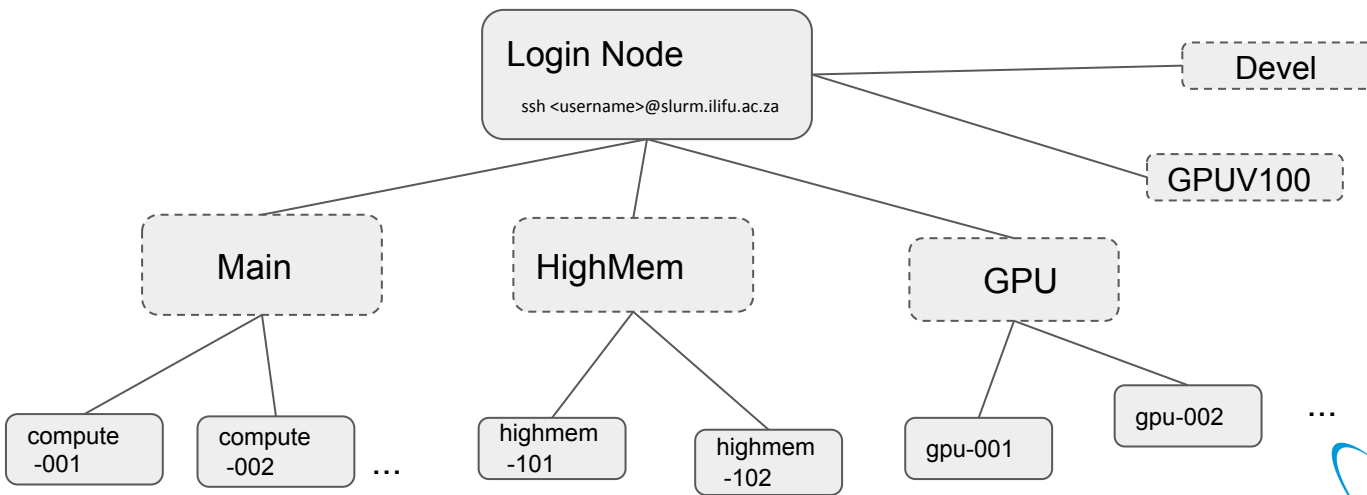
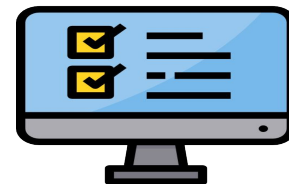


SLURM

http://docs.ilifu.ac.za/#/tech_docs/running_jobs?id=specifying-resources-when-running-jobs-on-slurm



Main	HighMem	GPU	GPUV100	Devel
~80 nodes	2 nodes	4 nodes	1 node	1 node
32 CPUs / 232 GB RAM	32 CPUs / 480 GB RAM	2 GPUs / 232 GB RAM	24 CPUs / 237 GB RAM	32 CPUs / 237 GB RAM



SLURM - running an interactive job



- X11 forwarding support

```
$ ssh -Y <username>@slurm.ilifu.ac.za
```

- Allocates a Slurm compute node:

```
$ sinteractive --x11
```

- Devel partition

```
$ srun --x11 --pty bash
```

- Main partition



- Specify lower wall-time (default 3 hours) and less memory (default ~7GB) increases chance of job launching immediately

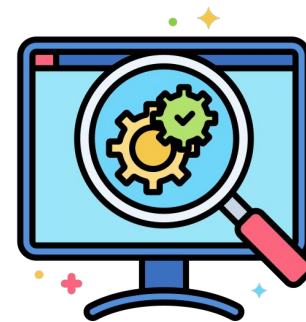
- In steps:

```
$ srun --pty --time=10 --mem=1GB bash
$ singularity shell /idia/software/containers/python-3.6.img
$ python3 job_script.py
```

- In single call:

```
$ srun --time=10 --mem=1GB singularity exec
/idia/software/containers/python-3.6.img python3 job_script.py
```

- Must manually process after this



- Email parameters

```
$ srun --mail-user=<address> --mail-type=<event_types>  
-   Events : BEGIN,END,FAIL,TIME_LIMIT_80
```

- Exclude nodes
 - e.g. problematic nodes (report to ilifu support)

```
$ srun --exclude=compute-[101,101-105]
```



SLURM - advanced user commands



- Before running jobs :

List partitions and their specs

```
$ sinfo -O "partition,available,cpus,nodes,memory,statecompact"
```

List your SLURM accounting groups

```
$ sacctmgr show user <username> cluster=ilifu-slurm2021 -s format=account%30
```



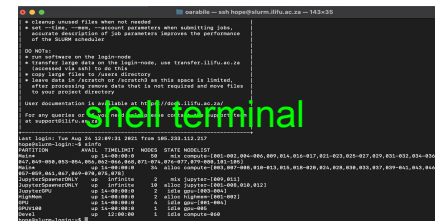
- Syntax for srun:

Submit to Main under specific account

```
$ srun --partition=Main --account=b34-admins-ag
```

Submit job to GPU partition

```
$ srun --partition=GPU job_script.sh
```



- After / during running jobs :
 - jobID is given from sbatch output / queue

Shows info about job running including working directory

```
$ scontrol show jobID <jobID>
```

Shows info for multi-CPU jobs

```
$ sacct -o JobID%-15,JobName%-15,Partition,Account,  
Elapsed,NNodes%6,NTASK%6,NCPUS%5,MaxDiskRead,MaxDiskWrite,  
NodeList%20,MaxRSS,CPUTime,State,ExitCode
```

Shows jobs started and completed between these dates

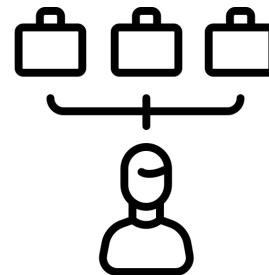
```
$ sacct -S 2021-09-01-09:00 -E 2021-09-14-10:00 -X -o  
Jobid,JobName,Start,End,State
```



DEMO TIME!

- Array jobs allow quick submission of many similar jobs, each with same resources, without any manual launch
 - Passes array task ID into script, which changes behavior of each job each time i.e different inputs
 - Can be used to run many related steps in a serial process
- Example job array Running, 20 jobs, with 5 run simultaneously
-

```
#!/bin/bash
#SBATCH --array=1-20%5
#SBATCH --job-name=myarrayjob
#SBATCH --output=logs/%x-%A_%a.out
#SBATCH --error=logs/%x-%A_%a.err
singularity exec python myscript.py --input $SLURM_ARRAY_TASK_ID
```



Parameter	Substitution / filename pattern	Environment Variables
jobID of running job	%j	SLURM_JOB_ID
Job name	%x	SLURM_JOB_NAME
Job array's master job allocation number	%A	SLURM_ARRAY_JOB_ID
Job array task ID (index) number	%a	SLURM_ARRAY_TASK_ID
CPUs per task		SLURM_CPUS_PER_TASK

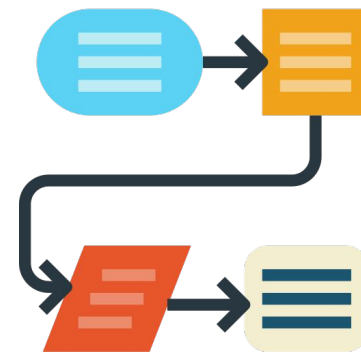
- Allow jobs to be scheduled for running, based on the status of a previous job
 - e.g only begin a particular job once previous one successfully completes

Submit `another_job.sh` to SLURM queue, to begin after jobID 1234 successfully completes , or cancel the job if jobID 1234 fails

```
$ sbatch -d afterok:1234 --kill-on-invalid-dep=yes another_job.sh
```

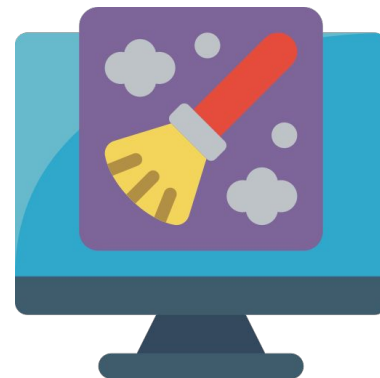
Submit `another_job.sh` to SLURM queue, to begin after jobID 1234 & 5678 completes

```
$ sbatch -d afterany:1234:5678 another_job.sh
```



Do's :

- Run jobs using sbatch rather than interactive jobs
- Identify job resources requirements:
 - No. of nodes and CPUs, amount of RAM and wall-time.
- Remove files that aren't needed
 - /scratch3 folder after data processing is complete
 - Old raw data, temporary products , etc.
- Use Singularity (cannot install software on nodes)



Don't:

- Don't run software/heavy processes on login node
- Don't place large files in your home directory (/users)
- Don't transfer using scp/rsync on the login node

Thank you!