

# Submitting batch jobs

## Slurm on ecgate

Xavi Abellan  
xavier.abellan@ecmwf.int  
User Support Section

# Outline

- Interactive mode versus Batch mode
- Overview of the Slurm batch system on ecgate
- Batch basic concepts
- Creating a batch job
- Basic job management
- Checking the batch system status
- Accessing the Slurm Accounting database
- Trouble-shooting
- Bonus: migration from LoadLeveler

# Interactive vs Batch

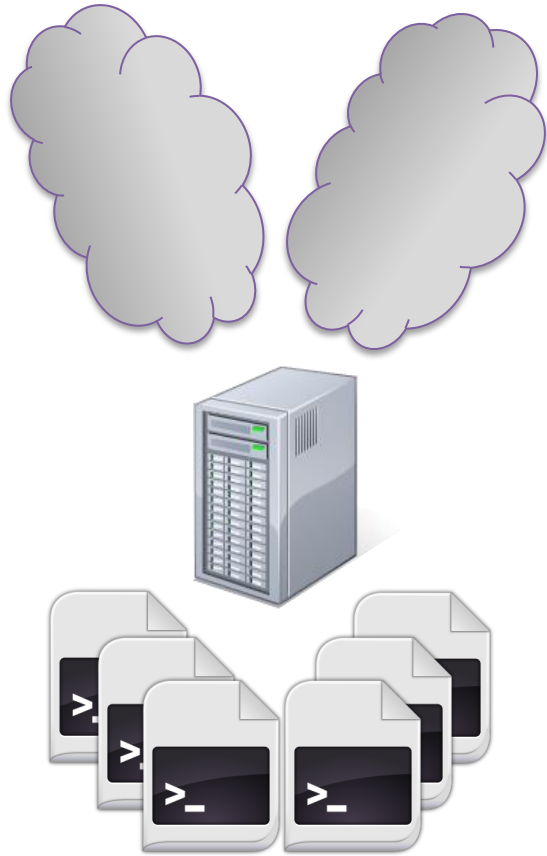
- When you login, the default shell on ecgate is either the Korn-shell (ksh), Bash or the C-shell (csh).
- To run a script or a program **interactively**, enter the executable name and any necessary arguments at the system prompt.
- You can also run your job in **background** so that other commands can be executed at the same time...

```
$> ./your-program arg1 arg2  
$> ./your-program arg1 arg2 &
```

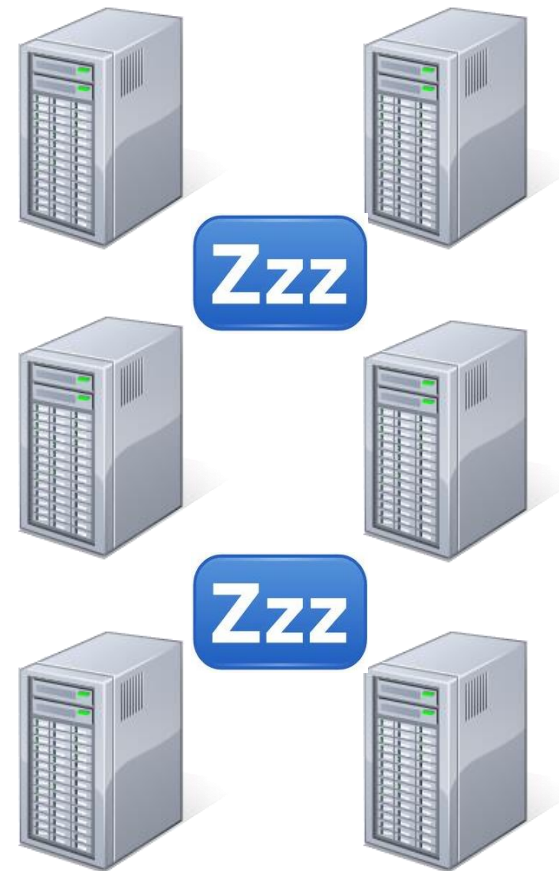
# Interactive vs Batch

- But... **Background is not batch**
  - The program is still running interactively on the login node
    - You share the node with the rest of the users
  - The limits for interactive sessions still apply:
    - CPU time limit of 30 min per process
- ```
$> ulimit -a
```
- Interactive sessions should be limited to development tasks, editing files, compilation or very small tests

# Interactive vs Batch

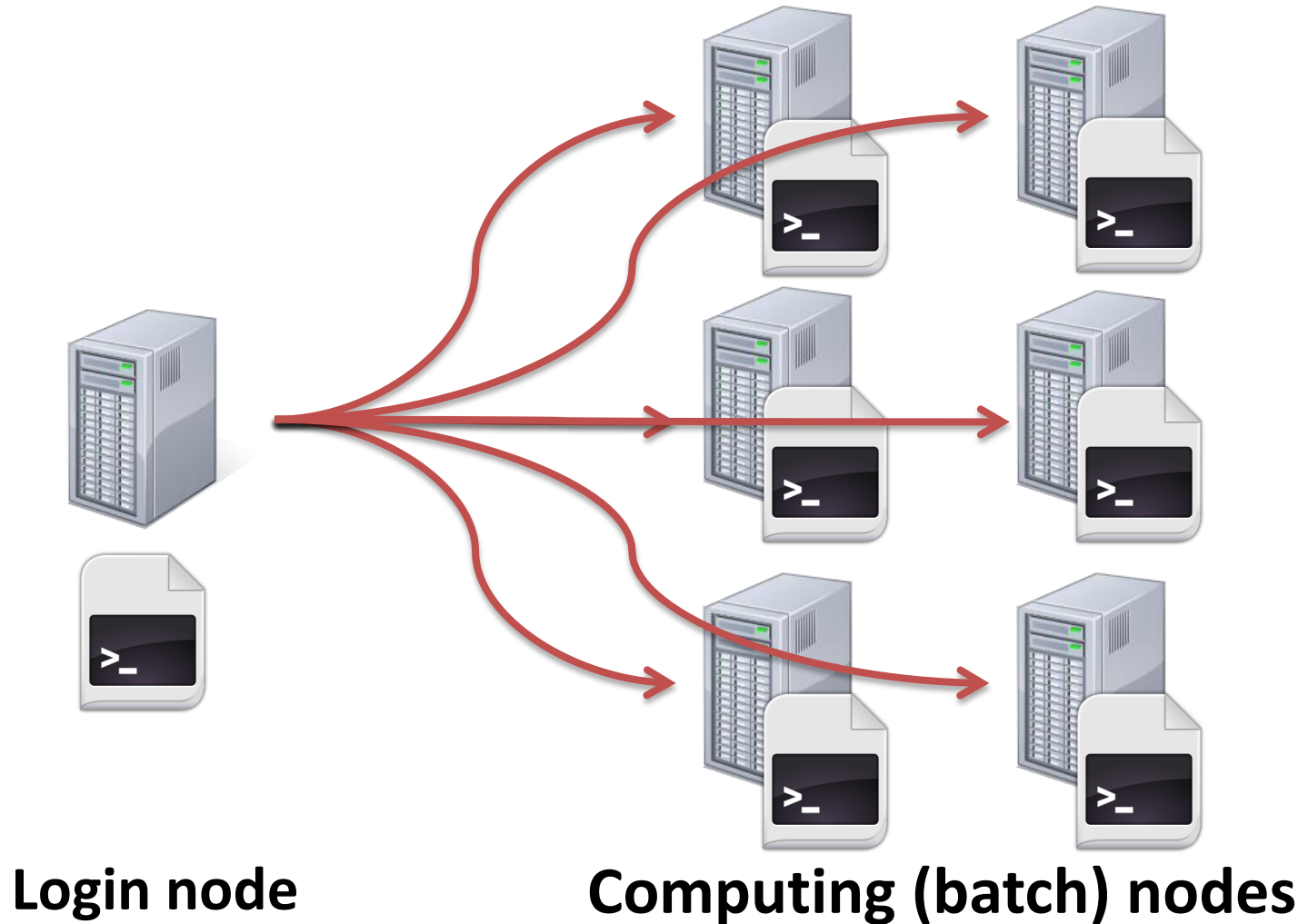


**Login node**



**Computing (batch) nodes**

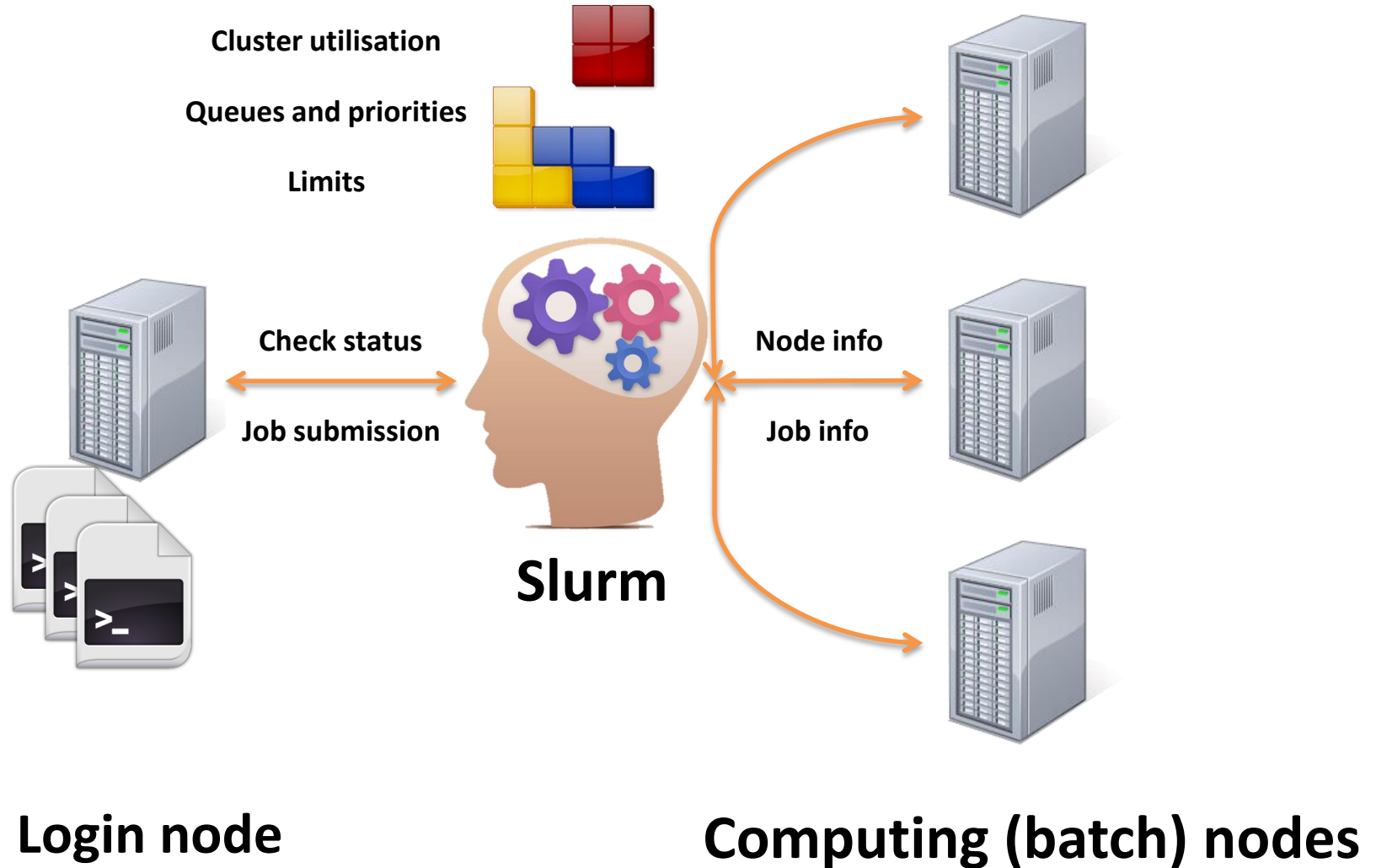
# Interactive vs Batch



# Batch on ecgate

- We used LoadLeveler in the previous ecgate
- The batch system used on the current is Slurm:
  - Cluster workload manager:
    - Framework to execute and monitor batch work
    - Resource allocation (where?)
    - Scheduling (when?)
- **Batch job:** shell script that will run unattended, with some special directives describing the job itself

# How does it work?





# Quality of service (queues)

- In Slurm, QoS (Quality of Service) = queue
- The queues have an associated priority and have certain limits
- Standard queues available to all users

| QoS     | Description                                        | Priority | Wall Time Limit | Total Jobs | User Jobs |
|---------|----------------------------------------------------|----------|-----------------|------------|-----------|
| express | Suitable for short jobs                            | 400      | 3 hours         | 128        | 16        |
| normal  | Suitable for most of the work. This is the default | 300      | 1 day           | 256        | 20        |
| long    | Suitable for long jobs                             | 200      | 7 days          | 32         | 4         |

- Special queues with the access restricted to meet certain conditions

| QoS       | Description                                                   | Priority | Wall Time Limit | Total Jobs | User Jobs |
|-----------|---------------------------------------------------------------|----------|-----------------|------------|-----------|
| timecrit1 | Automatically set by EcAccess for Time Critical Option 1 jobs | 500      | 8 hours         | 128        | 16        |
| timecrit2 | Only for jobs belonging to Time Critical Option 2 suites      | 600      | 3 hours         | 96         | 32        |

# Batch job script

```
#!/bin/bash
# The job name
#SBATCH --job-name=helloworld
# Set the error and output files
#SBATCH --output=hello-%J.out
#SBATCH --error=hello-%J.out
# Set the initial working directory
#SBATCH --workdir=/scratch/us/usxa
# Choose the queue
#SBATCH --qos=express
# Wall clock time limit
#SBATCH --time=00:05:00
# Send an email on failure
#SBATCH --mail-type=FAIL

# This is the job
echo "Hello World!"
sleep 30
```

- A job is a shell script
  - bash/ksh/csh
- Directives are shell comments:
  - starting with **#SBATCH**
  - Lowercase only
  - No spaces in between
  - No variable expansion
- All directives are optional
  - System defaults in place

# Job directives

| Directive                    | Description                                                                                                                | Default        |
|------------------------------|----------------------------------------------------------------------------------------------------------------------------|----------------|
| <code>--job-name=...</code>  | A descriptive name for the job                                                                                             | Script name    |
| <code>--output=...</code>    | Path to the file where standard output is redirected. Special placeholders for job id ( %j ) and the execution node ( %N ) | slurm-%j.out   |
| <code>--error=...</code>     | Path to the file where standard error is redirected. Special placeholders for job id ( %j ) and the execution node ( %N )  | output value   |
| <code>--workdir=...</code>   | Working directory of the job. The output and error files can be defined relative to this directory.                        | submitting dir |
| <code>--qos=...</code>       | Quality of service (queue) where the job is to be submitted                                                                | normal*        |
| <code>--time=...</code>      | Wall clock limit of the job (not cpu time limit!)<br>Format: m, m:s, h:m:s, d-h, d-h:m or d-h:m:s                          | qos default    |
| <code>--mail-type=...</code> | Notify user by email when certain event types occur. Valid type values are BEGIN, END, FAIL, REQUEUE, and ALL              | disabled       |
| <code>--mail-user=...</code> | Email address to send the email                                                                                            | submit user    |
| <code>--hold</code>          | Submit the job in held state. It won't run until released with scontrol release <jobid>                                    | not used       |

# Submitting a job: sbatch

- **sbatch**: Submits a job to the system. Job is configured:
  - including the directives in the job script
  - using the same directives as command line options
- The job to be submitted can be specified:
  - As an argument of sbatch
  - If no script is passed as an argument, sbatch will read the job from standard input

```
$> sbatch hello.sh
Submitted batch job
1250968
$> cat hello-1250968.out
Hello world!
$>
```

- The corresponding job id will be returned if successful, or an error if the job could not be submitted

# Submitting a job from cron

- Slurm jobs take the environment from the submission session
  - Submitting from cron will cause the jobs to run with a very limited environment and will most likely fail
  - Use a crontab line similar to:

```
$> 05 12 * * * $HOME/cronrun sbatch $HOME/cronjob
```

- Where the script cronrun is:

```
#!/bin/ksh
# cronrun script
. ~/.profile
. ~/.kshrc
$@
```

```
#!/bin/bash
# cronrun script
. ~/.bash_profile
$@
```

```
#!/bin/csh
# cronrun script
. ~/.login
$@
```

# Checking the queue: squeue

- **squeue**: displays some information about the jobs currently running or waiting
- By default it shows all jobs from all users, but some filtering options are possible:
  - -u <comma separated list of users>
  - -q <comma separated list of QoSs>
  - -n <comma separated list of job names>
  - -j <comma separated list of job ids>
  - -t <comma separated list of job states>

```
$> squeue -u $USER
JOBID      NAME      USER      QOS      STATE      TIME  TIMELIMIT  NODELIST (REASON)
1250968  helloworld  usxa      express  RUNNING    0:08      5:00      ecgb07
```

# Canceling a job: scancel

- **scancel:** Cancels the specified job(s)

```
$> sbatch hello.sh
Submitted batch job 1250968
$> scancel 1250968
$> scancel 1250968
scancel: error: Kill job error on job id 1250968: Invalid job id
specified
$> sbatch hello.sh
Submitted batch job 1250969
$> scancel -in hello.sh
Cancel job_id= 1250969 name=hello.sh partition=batch [y/n]? y
$> sbatch hello.sh
Submitted batch job 1250970
$> scancel -i -v 1250970
scancel: auth plugin for Munge (http://code.google.com/p/munge/) loaded
Cancel job_id=1250970 name=hello.sh partition=batch [y/n]? y
scancel: Terminating job 1250970
```

- A job can be cancelled either if it is running or still waiting on the queue

```
slurmd[ecgb07]: *** JOB 1250968 CANCELLED AT 2014-02-28T17:08:29 ***
```

# Canceling a job: scancel options

- The most common usage of scancel is:

```
$> scancel <jobid1> <jobid2> <jobid3>
```

| Option       | Description                                                           |
|--------------|-----------------------------------------------------------------------|
| -n <jobname> | Cancel all the jobs with the specified job name                       |
| -t <state>   | Cancel all the jobs that are in the specified state (PENDING/RUNNING) |
| -q <qos>     | Cancel only jobs on the specified QoS                                 |
| -u \$USER    | Cancel ALL the jobs of the current user. Use carefully!               |
| -i           | Interactive option: ask for confirmation before cancelling jobs       |
| -v           | Verbose option. It will show what is being done                       |

**Note: An ordinary user can only cancel their own jobs**



# Practical 1: Basic job submission

- Practicals must be run on **ecgate**, so make sure you log in there first!

```
$> ssh ecgate
$> cd $SCRATCH
$> tar xvzf ~trx/intro/batch_ecgate_practicals.tar.gz
$> cd batch_ecgate_practicals/basic
```

1. Have a look at the script “env.sh”
2. Submit the job and check whether it is running
  - What QoS is it using? What is the time limit of the job?
3. Where did the output of the job go? Have a look at the output
4. Submit the job again and then once it starts cancel it
5. Check the output



# Practical 1: Basic job submission

- Can you modify the previous job so it...
  1. ... runs in the express QoS, with a wall clock limit of 5 minutes?
  2. ... uses the subdirectory work/ as the working directory?
  3. ... sends the...
    - a) ... output to the file work/env\_out\_<jobid>.out ?
    - b) ... error to work/env\_out\_<jobid>.err?
  4. ... sends you an email when the job starts?
- Try your job after the modifications and check if they are correct
  - You can do the modifications one by one or all at once...



# Why doesn't my job start?

- Check the last column of the squeue output for a hint...

```
$> squeue -j 1261265
JOBID      NAME      USER      QOS      STATE      TIME  TIMELIMIT      NODELIST (REASON)
1261265    sbatch    usxa      long     PENDING    0:00  7-00:00:00     (QOSResourceLimit)
```

| Reason           | Description                                                            |
|------------------|------------------------------------------------------------------------|
| Priority         | There are other jobs with more priority                                |
| Resources        | No free resources are available                                        |
| JobUserHeld      | The job is held. Release with scontrol release <jobid>                 |
| QOSResourceLimit | You have reached a limit in the number of jobs you can submit to a QoS |

- My job is PENDING because of a QOSResourceLimit...
- How do I check my limits?

# Checking limits and general usage: sqos

- **sqos**: Utility to have an overview of the different QoSs where the user have access, including usage and limits
  - This utility is ECMWF specific (not part of a standard Slurm installation)

```
$> sqos
```

| QoS       | Prio | Max Wall   | Total Jobs | User Jobs | Max CPUS | Max Mem  |
|-----------|------|------------|------------|-----------|----------|----------|
| express   | 400  | 03:00:00   | 11 / 128   | 7 / 12    | 1        | 10000 MB |
| normal    | 300  | 1-00:00:00 | 23 / 128   | 4 / 12    | 1        | 10000 MB |
| long      | 200  | 7-00:00:00 | 7 / 32     | 4 / 4     | 1        | 10000 MB |
| large     | 200  | 08:00:00   | 0 / 8      | 0 / 4     | 1        | 10000 MB |
| timecrit1 | 500  | 08:00:00   | 0 / 96     | 0 / 16    | 1        | 10000 MB |

Total: 43 Jobs, 41 RUNNING, 2 PENDING

| Account  | Def QoS | Running Jobs | Submitted Jobs |
|----------|---------|--------------|----------------|
| *ectrain | normal  | 15 / 36      | 17 / 1000      |

User trx: 17 Jobs, 15 RUNNING, 2 PENDING

# More details on current jobs and nodes

- **scontrol:** view and modify Slurm jobs and node configuration

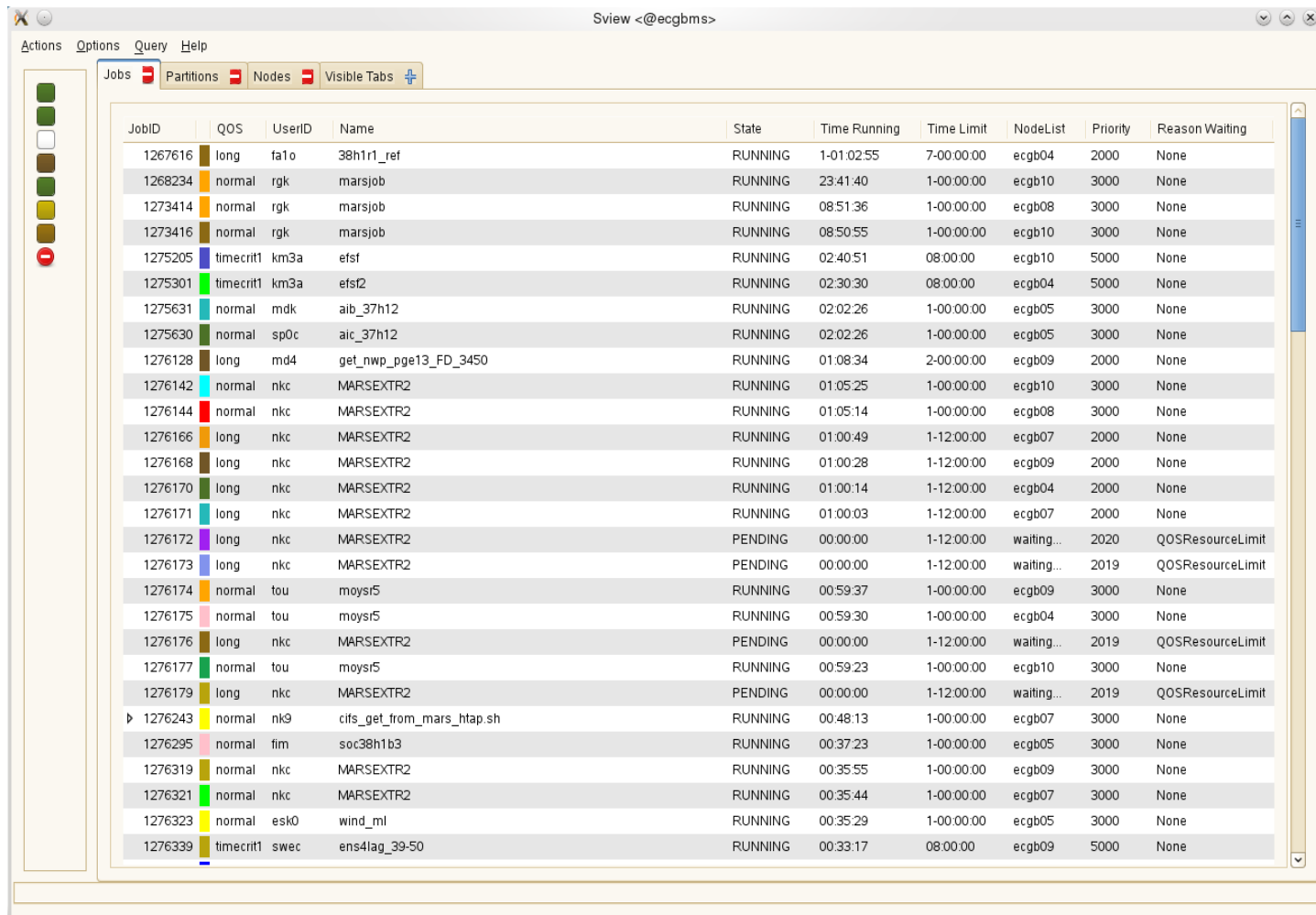
```
$> scontrol show job 24789
JobId=24789 Name=test_slurm_csh
  UserId=us2(1666) GroupId=gb(3070)
  Priority=3000 Account=ecus QOS=normal
  JobState=COMPLETED Reason=None Dependency=(null)
  Requeue=0 Restarts=0 BatchFlag=1 ExitCode=0:0
  RunTime=00:01:25 TimeLimit=00:10:00 TimeMin=N/A
  SubmitTime=2013-05-09T08:55:34 EligibleTime=2013-05-09T08:55:34
  StartTime=2013-05-09T08:55:34 EndTime=2013-05-09T08:56:59
  PreemptTime=None SuspendTime=None SecsPreSuspend=0
  Partition=batch AllocNode:Sid=ecgb05:36790
  ReqNodeList=(null) ExcNodeList=(null)
  NodeList=ecgb05
  BatchHost=ecgb05
  NumNodes=1 NumCPUs=1 CPUs/Task=1 ReqS:C:T=*:*:~*
  MinCPUsNode=1 MinMemoryCPU=1900M MinTmpDiskNode=0
  Features=(null) Gres=(null) Reservation=(null)
  Shared=1 Contiguous=0 Licenses=(null) Network=(null)
  Command=/home/ms/gb/us2/slurm_csh.job
  WorkDir=/scratch/ms/gb/us2/csh
  Comment=Output=/scratch/ms/gb/us2/csh/slurm_24789.out;Error=/scratch/ms/gb/us2/csh/slurm_24789.out;
```

- **sinfo:** View information about node status

```
$> sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
batch*    up        infinite   1  drain ecgb11
batch*    up        infinite   6  alloc ecgb[04-05,07-10]
test      up        infinite   1  idle  ecgb06
```

# More details on current jobs and nodes

- **sview**: GUI to see the queue and node status



The screenshot shows the sview GUI window titled "Sview <@ecgbms>". The window has a menu bar with "Actions", "Options", "Query", and "Help". Below the menu bar are tabs for "Jobs", "Partitions", "Nodes", and "Visible Tabs". The main area displays a table of job details with the following columns: JobID, QOS, UserID, Name, State, Time Running, Time Limit, NodeList, Priority, and Reason Waiting. The table contains 33 rows of job data.

| JobID   | QOS       | UserID | Name                       | State   | Time Running | Time Limit | NodeList   | Priority | Reason Waiting   |
|---------|-----------|--------|----------------------------|---------|--------------|------------|------------|----------|------------------|
| 1267616 | long      | fa1o   | 38h1r1_ref                 | RUNNING | 1-01:02:55   | 7-00:00:00 | ecgb04     | 2000     | None             |
| 1268234 | normal    | rgk    | marsjob                    | RUNNING | 23:41:40     | 1-00:00:00 | ecgb10     | 3000     | None             |
| 1273414 | normal    | rgk    | marsjob                    | RUNNING | 08:51:36     | 1-00:00:00 | ecgb08     | 3000     | None             |
| 1273416 | normal    | rgk    | marsjob                    | RUNNING | 08:50:55     | 1-00:00:00 | ecgb10     | 3000     | None             |
| 1275205 | timecrit1 | km3a   | etsf                       | RUNNING | 02:40:51     | 08:00:00   | ecgb10     | 5000     | None             |
| 1275301 | timecrit1 | km3a   | etsf2                      | RUNNING | 02:30:30     | 08:00:00   | ecgb04     | 5000     | None             |
| 1275631 | normal    | mdk    | aib_37h12                  | RUNNING | 02:02:26     | 1-00:00:00 | ecgb05     | 3000     | None             |
| 1275630 | normal    | sp0c   | aic_37h12                  | RUNNING | 02:02:26     | 1-00:00:00 | ecgb05     | 3000     | None             |
| 1276128 | long      | md4    | get_nwp_pge13_FD_3450      | RUNNING | 01:08:34     | 2-00:00:00 | ecgb09     | 2000     | None             |
| 1276142 | normal    | nkc    | MARSEXTR2                  | RUNNING | 01:05:25     | 1-00:00:00 | ecgb10     | 3000     | None             |
| 1276144 | normal    | nkc    | MARSEXTR2                  | RUNNING | 01:05:14     | 1-00:00:00 | ecgb08     | 3000     | None             |
| 1276166 | long      | nkc    | MARSEXTR2                  | RUNNING | 01:00:49     | 1-12:00:00 | ecgb07     | 2000     | None             |
| 1276168 | long      | nkc    | MARSEXTR2                  | RUNNING | 01:00:28     | 1-12:00:00 | ecgb09     | 2000     | None             |
| 1276170 | long      | nkc    | MARSEXTR2                  | RUNNING | 01:00:14     | 1-12:00:00 | ecgb04     | 2000     | None             |
| 1276171 | long      | nkc    | MARSEXTR2                  | RUNNING | 01:00:03     | 1-12:00:00 | ecgb07     | 2000     | None             |
| 1276172 | long      | nkc    | MARSEXTR2                  | PENDING | 00:00:00     | 1-12:00:00 | waiting... | 2020     | QOSResourceLimit |
| 1276173 | long      | nkc    | MARSEXTR2                  | PENDING | 00:00:00     | 1-12:00:00 | waiting... | 2019     | QOSResourceLimit |
| 1276174 | normal    | tou    | moysr5                     | RUNNING | 00:59:37     | 1-00:00:00 | ecgb09     | 3000     | None             |
| 1276175 | normal    | tou    | moysr5                     | RUNNING | 00:59:30     | 1-00:00:00 | ecgb04     | 3000     | None             |
| 1276176 | long      | nkc    | MARSEXTR2                  | PENDING | 00:00:00     | 1-12:00:00 | waiting... | 2019     | QOSResourceLimit |
| 1276177 | normal    | tou    | moysr5                     | RUNNING | 00:59:23     | 1-00:00:00 | ecgb10     | 3000     | None             |
| 1276179 | long      | nkc    | MARSEXTR2                  | PENDING | 00:00:00     | 1-12:00:00 | waiting... | 2019     | QOSResourceLimit |
| 1276243 | normal    | nk9    | cifs_get_from_mars_htap.sh | RUNNING | 00:48:13     | 1-00:00:00 | ecgb07     | 3000     | None             |
| 1276295 | normal    | fim    | soc38h1b3                  | RUNNING | 00:37:23     | 1-00:00:00 | ecgb05     | 3000     | None             |
| 1276319 | normal    | nkc    | MARSEXTR2                  | RUNNING | 00:35:55     | 1-00:00:00 | ecgb09     | 3000     | None             |
| 1276321 | normal    | nkc    | MARSEXTR2                  | RUNNING | 00:35:44     | 1-00:00:00 | ecgb07     | 3000     | None             |
| 1276323 | normal    | esk0   | wind_ml                    | RUNNING | 00:35:29     | 1-00:00:00 | ecgb05     | 3000     | None             |
| 1276339 | timecrit1 | swec   | ens4lag_39-50              | RUNNING | 00:33:17     | 08:00:00   | ecgb09     | 5000     | None             |

# Access to the Slurm accounting DB: sacct

- **sacct**: View present and past job information

```
$> sacct -X
```

| JobID | JobName    | QOS     | State      | ExitCode | Elapsed  | NodeList |
|-------|------------|---------|------------|----------|----------|----------|
| 24804 | test.sh    | normal  | COMPLETED  | 0:0      | 00:00:13 | ecgb04   |
| 24805 | test.sh    | normal  | COMPLETED  | 0:0      | 00:01:10 | ecgb04   |
| 24806 | test.sh    | normal  | COMPLETED  | 0:0      | 00:00:47 | ecgb04   |
| 24807 | test.sh    | normal  | COMPLETED  | 0:0      | 00:01:32 | ecgb04   |
| 24808 | test.sh    | normal  | COMPLETED  | 0:0      | 00:02:19 | ecgb04   |
| 24809 | test.sh    | normal  | COMPLETED  | 0:0      | 00:00:45 | ecgb04   |
| 24972 | test.sh    | normal  | RUNNING    | 0:0      | 00:02:35 | ecgb04   |
| 24973 | test.sh    | normal  | RUNNING    | 0:0      | 00:02:35 | ecgb04   |
| 24974 | test.sh    | normal  | CANCELLED+ | 0:0      | 00:01:24 | ecgb04   |
| 24975 | test.sh    | normal  | RUNNING    | 0:0      | 00:02:35 | ecgb04   |
| 24976 | test.sh    | normal  | COMPLETED  | 0:0      | 00:00:40 | ecgb04   |
| 24977 | test.sh    | normal  | RUNNING    | 0:0      | 00:02:35 | ecgb04   |
| 24978 | test.sh    | normal  | COMPLETED  | 0:0      | 00:00:40 | ecgb04   |
| 24979 | test.sh    | normal  | RUNNING    | 0:0      | 00:02:33 | ecgb04   |
| 24981 | helloworld | normal  | FAILED     | 1:0      | 00:00:01 | ecgb04   |
| 24983 | test.sh    | normal  | CANCELLED+ | 0:0      | 00:00:33 | ecgb04   |
| 24984 | test.sh    | normal  | RUNNING    | 0:0      | 00:01:39 | ecgb04   |
| 24985 | test.sh    | express | RUNNING    | 0:0      | 00:01:23 | ecgb04   |
| 24986 | test.sh    | express | RUNNING    | 0:0      | 00:01:23 | ecgb04   |
| 24987 | test.sh    | long    | RUNNING    | 0:0      | 00:01:19 | ecgb04   |

# Access to the Slurm accounting DB: sacct options

- By default, sacct will return information about your jobs that started today

| Option                            | Description                                                                                                                                                  |
|-----------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>-j &lt;jobid&gt;</code>     | Show the job with that jobid                                                                                                                                 |
| <code>-u &lt;user&gt;</code>      | Show jobs for the specified user. Use option <code>-a</code> for all users                                                                                   |
| <code>-E &lt;endtime&gt;</code>   | Show jobs eligible before that date and time                                                                                                                 |
| <code>-S &lt;starttime&gt;</code> | Show jobs eligible after that date and time                                                                                                                  |
| <code>-s &lt;statelist&gt;</code> | Show jobs on the states (comma-separated) given during the time period. Valid states are: CANCELLED, COMPLETED, FAILED, NODE_FAIL, RUNNING, PENDING, TIMEOUT |
| <code>-q &lt;qos&gt;</code>       | Show jobs only for the qos selected                                                                                                                          |
| <code>-o &lt;outformat&gt;</code> | Format option. Comma-separated names of fields to display                                                                                                    |
| <code>-e</code>                   | Show the different columns to be used for the <code>-o</code> option                                                                                         |
| <code>-X</code>                   | Hide the job step information, showing the allocation only                                                                                                   |



# What happened to my job: job\_forensics

- **job\_forensics**: Custom ECMWF utility to dump forensic information about a job

```
$> job_forensics 1261917
DB Information:
-----
Job:
  JobID:1261917
  JobName:sbatch
  User:trx
  UID:414
  Group:ectrain
  GID:1400
  Account:ectrain
  QOS:long
  Priority:2000
  Partition:batch
  NCPUS:32
  NNodes:1
  NodeList:ecgb09
  State:COMPLETED
  Timelimit:7-00:00:00
  Submit:2014-03-01T16:19:06
  Eligible:2014-03-01T16:19:06
  Start:2014-03-01T16:19:06
  End:2014-03-01T16:20:07
  Elapsed:00:01:01
  CPUtime:00:32:32
  UserCPU:00:00.005
  SystemCPU:00:00.004
  TotalCPU:00:00.010
  DerivedExitCode:0:0
  ExitCode:0:0
  Output:/home/ectrain/trx/slurm-1261917.out
  Error:/home/ectrain/trx/slurm-1261917.out
```

```
...
Main step:
  JobID:1261917.batch
  JobName:batch
  NCPUS:1
  CPUtime:00:01:01
  AveRSS:1796K
  MaxRSS:1796K
  MaxRSSNode:ecgb09
  MaxRSSTask:0

Controller Logs:
-----
[2014-03-01T16:19:06+00:00]
_slurm_rpc_submit_batch_job JobId=1261917
usec=4494
...

ecgb09 log (main):
-----
[2014-03-01T16:19:07+00:00] Launching batch job
1261917 for UID 414
[2014-03-01T16:20:07+00:00] [1261917] sending
REQUEST_COMPLETE_BATCH_SCRIPT, error:0
[2014-03-01T16:20:07+00:00] [1261917] done with
job
```

# Practical 2: reviewing past runs

- How would you...

- retrieve the list of jobs that you ran today?

```
$> sacct ...
```

- retrieve the list of all the jobs that were cancelled today by user trx?

```
$> sacct ...
```

- ask for the submit, start and end times for a job of your choice?

```
$> sacct ...
```

- find out the output and error paths for a job of your choice?

```
$> sacct ...
```



# Practical 3: Fixing broken jobs

```
$> cd $SCRATCH/batch_ecgate_practicals/broken
```

- What is wrong in job1? Can you fix it?
- What is wrong in job2? Can you fix it?
- What is wrong in job3? Can you fix it?



# Bonus: Migrating from LoadLeveler

- You can submit a LL job to Slurm, but the LL directives will be ignored!
  - Translation required: manually or using **ll2slurm**

```
$> ll2slurm -h
usage: ll2slurm [-h] [-i INSCRIPT] [-o OUTSCRIPT] [-q] [-f]

Job translator from LoadLeveler to Slurm

optional arguments:
  -h, --help                show this help message and exit
  -i INSCRIPT, --inscript INSCRIPT
                           Input script. By default reads stdin
  -o OUTSCRIPT, --outscript OUTSCRIPT
                           Output translated script. By default writes to stdout
  -q, --quiet               Do not produce warning or error messages on stderr
  -f, --force               Overwrite the output file if it exists
```

- Not all the LoadLeveler keywords can be translated.
  - Some manual additions might be required! You may play with the example:

```
$SCRATCH/batch_ecgate_practicals/loadleveler
```

# Migration cheatsheet (I)

| User Commands  | LoadLeveler       | SLURM            |
|----------------|-------------------|------------------|
| Job submission | llsubmit [script] | sbatch [script]  |
| Job cancel     | llcancel [job_id] | scancel [job_id] |
| Job status     | llq [-j job_id]   | squeue [job_id]  |
| Queue list     | llq               | squeue           |

| Environment Variables | LoadLeveler            | SLURM                |
|-----------------------|------------------------|----------------------|
| Job ID                | \$LOADL_STEP_ID        | \$SLURM_JOBID        |
| Working Dir           | \$LOADL_STEP_INITDIR   | pwd command          |
| Node List             | \$LOADL_PROCESSOR_LIST | \$SLURM_JOB_NODELIST |

# Migration cheatsheet (II)

| Job Configuration  | LoadLeveler                 | SLURM                                         |
|--------------------|-----------------------------|-----------------------------------------------|
| Script directive   | #@                          | #SBATCH                                       |
| Job Name           | job_name=[name]             | --job-name=[name]                             |
| Queue              | class=[queue]               | --qos=[queue]                                 |
| Wall Clock Limit   | wall_clock_limit=[hh:mm:ss] | --time=[min]<br>--time=[days-hh:mm:ss]        |
| Std Output File    | output=[file]               | --output=[file]                               |
| Std Error File     | error=[file]                | --error=[file]                                |
| Working Directory  | initialdir=[dir_name]       | --workdir=[dir_name]                          |
| Copy Environment   | environment=COPY_ALL        | --export=[ALL   NONE]<br>--export=[variables] |
| Email Notification | notification=...            | --mail-type=[events]                          |
| Email Address      | notify_user=[address]       | --mail-user=[address]                         |

# Additional Info

- Ecgate job examples:
  - <https://software.ecmwf.int/wiki/display/UDOC/ecgate+Slurm+batch+system>
- SLURM website and documentation:
  - <http://www.schedmd.com/>
  - <http://www.schedmd.com/slurmdocs/documentation.html>
  - <http://www.schedmd.com/slurmdocs/tutorials.html>

# Questions?