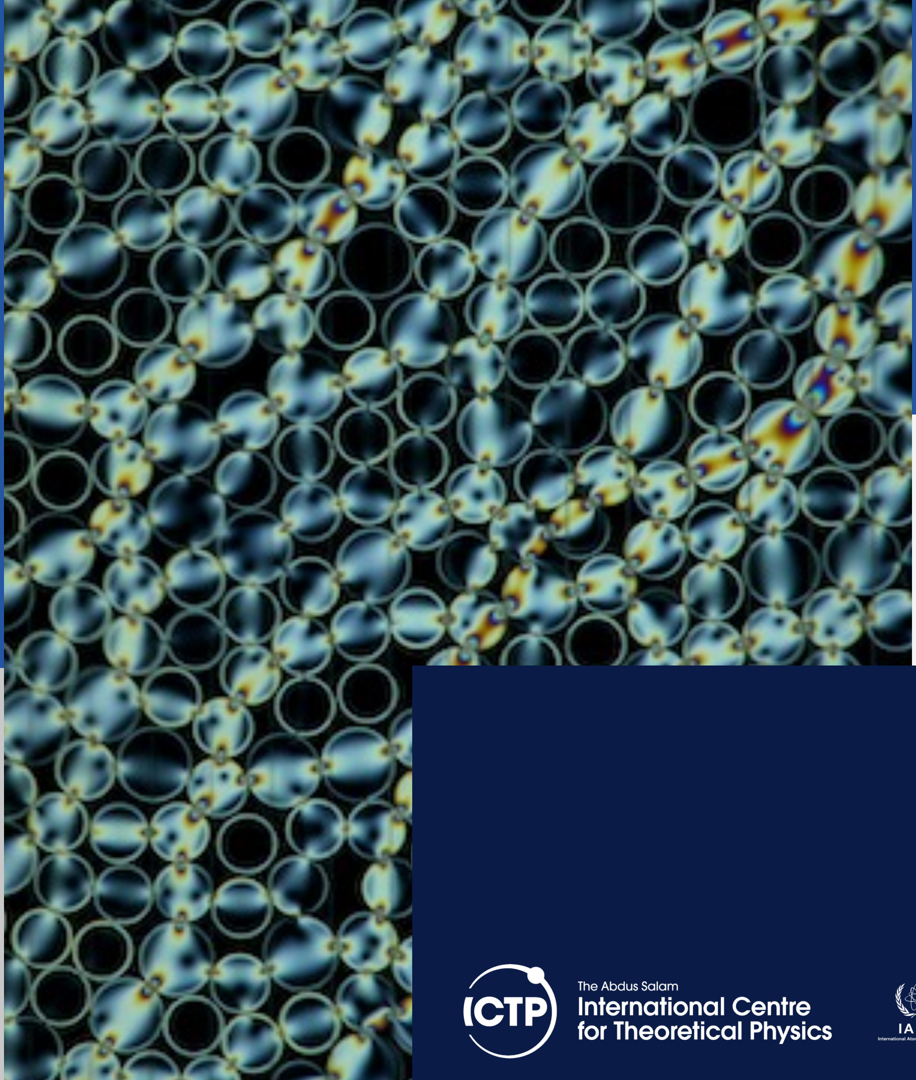
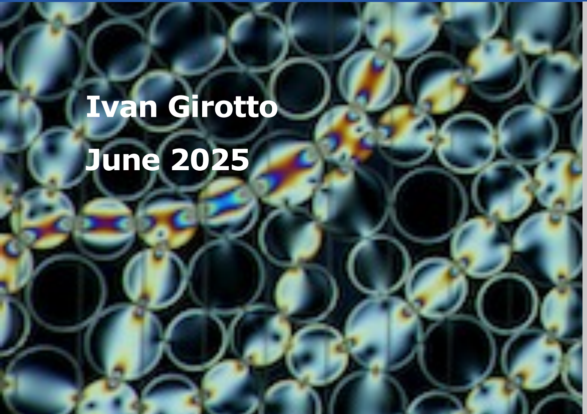
HPC 101

Use of Leonardo Supercomputer





Ivan Girotto

June 2025

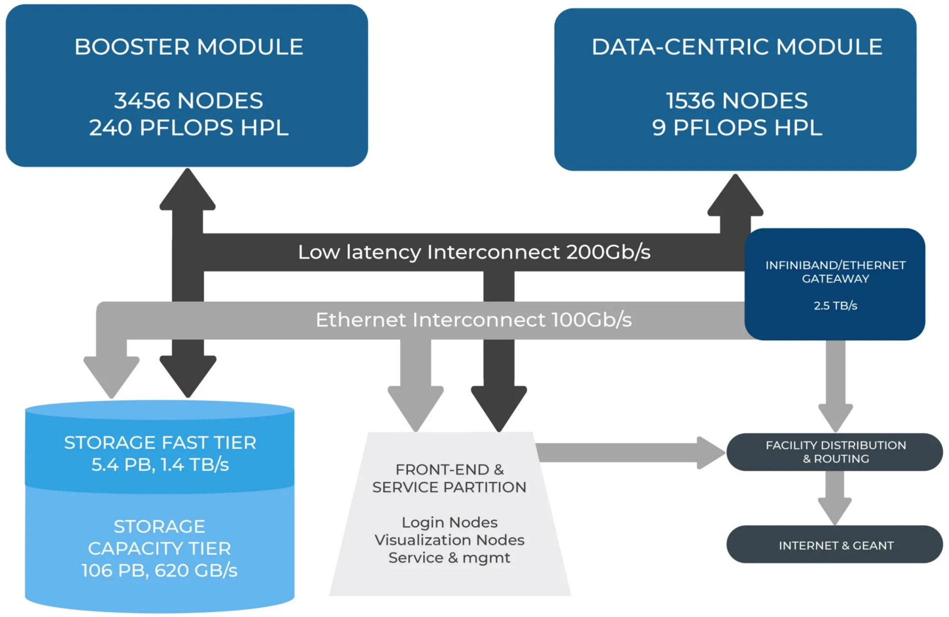
The Abdus Salam International Centre for Theoretical Physics

Outline

- Leonardo's architecture
- Accessing Leonardo
- Login nodes, accounting and disk Spaces
- Data transfer
- Software modules
- Scheduler

Leonardo Architecture

Leonardo infrastructure and login nodes



Atos BullSequana X430-E6

- Processors: **2 x CPU Intel Whitley ICP06, 32 cores Intel Ice Lake, 2.4 GHz**
- Hyper Threading is enabled!
- RAM: 512 (16x32) GB RAM DDR4 3200MHz
- 14TB disk in RAID1 configuration
- **NO GPUs**
- *Serial* partition on one login node (open to outside network)

Booster (GPU) module

In production
since August 2023

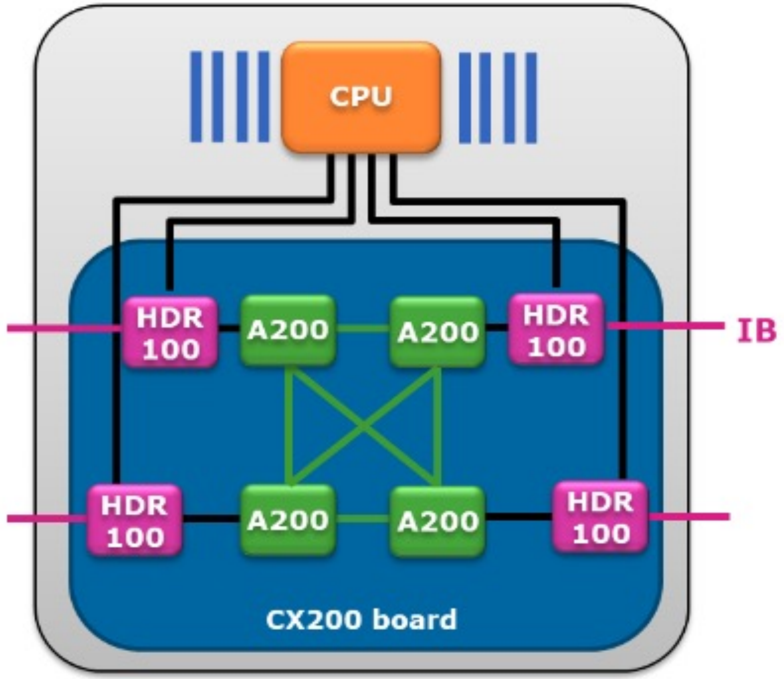
Atos BullSequana X2135 "Da Vinci" blade

- 3456 nodes
- Processors: **1 x CPU Intel Xeon 8358,**
  32 cores Intel Ice Lake, 2.6 GHz **(ONE SOCKET!)**
- RAM: 512 (8 x 64) GB DDR4 3200 MHz
- Accelerators: **4 x NVidia custom Ampere GPU A100 SXM4 64 GB,**
  NVLink 3.0
- Internal network: NVIDIA Mellanox HDR DragonFly+ 200Gb/s
- **DISKLESS!!!**
- Shared (via infiniband) storage space: 106 PB Capacity tier storage
  + 5.4 PB Fast tier storage

Peak performance per node: about 89,4 TFlops

Peak performance: about 309 Pflops (sustained: ~240 Pflops)

Booster (GPU) module



GPU performance

- ➢ 11.2 TFlops Peak FP64 per GPU or….
- ➢ 22.4 TFlops Peak FP32 per GPU or ….

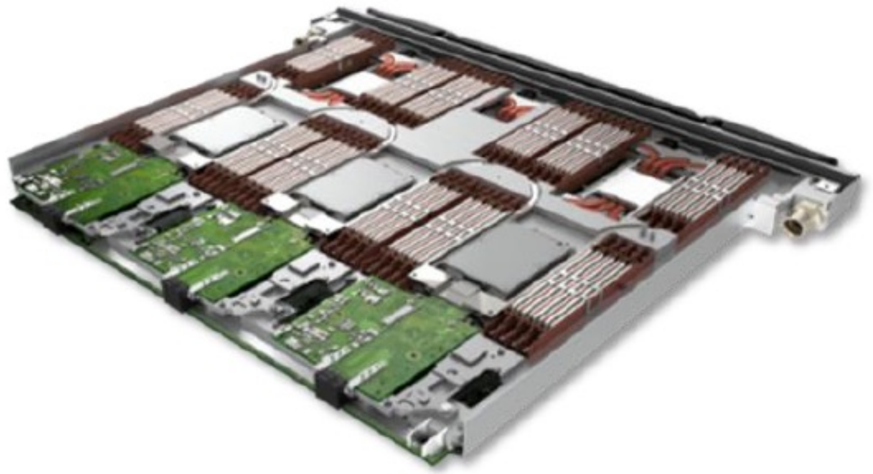
Intra-node connections

- ➢ NVLink, PCIe, GPU direct
- ➢ 200 GB/s between the GPU pairs
- ➢ Each GPU has direct 100Gb/s connection to the InfiniBand network
- ➢ PCIe Gen4 @ 31.5 GB/s

Memory

- ➢ 6.5 TB/s GPU memory bandwidth

Data Centric & General Purpose (CPU) module



Peak performance per node: about 8.46 TFlops
Peak performance: about 13 PFlops

BullSequana X2140 three-node CPU Blade

- 1536 nodes (512 blades)
- Processors: **2 x CPU Intel Xeon 8480+, 56 cores Intel Sapphire Rapids, 2.0 GHz**
- RAM: 256 (16x16) GB DDR5 4800MHz
  512 (16 x 32) GB DDR5 4800 MHz
- Infiniband: 1 x NVIDIA HDR cards 100 Gbps via PCIe Gen 5
- Disk: 1 x M.2 SSD 3,84 TB
- In production since February 2024

Storage

Fast Tier

5.4 PB @ 1.4 TB/s

NVMe storage (SSD disks)
(home + fast scratch)

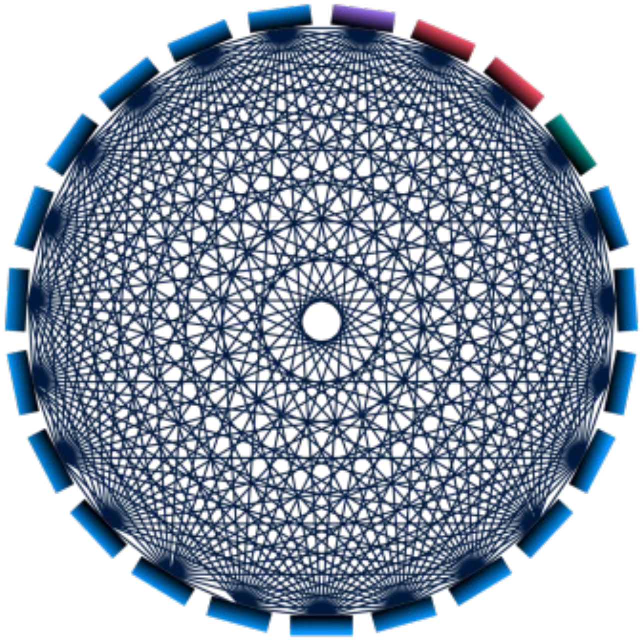
Capacity Tier

106 PB @ read 744 GB/s - write 620 GB/s

HDD disks
(work + large scratch + DRES)

DRES are at present “local” to Leonardo (no export to/from clusters @ Casalecchio)

Inter-node network topology



Booster Module nodes
I/O cell
Data-Centric cells
Hybrid cell (Booster + Data-Centric nodes)

Dragonfly+ topology (200 GBit/s bidirectional)

Based on NVidia Mellanox Infiniband HDR

- ➢ All nodes are divided into cells
- ➢ Non-blocking, two-layer Fat Tree within the cells
- ➢ All to all connection between cells

ADAPTIVE ROUTING ALGORITHM

alleviates traffic congestion (Slurm will take care of the “best”-possible node allocations on the dragonfly+ network with ARA enabled)

Accessing Leonardo

Preliminary step: login to Leonardo

The exercises of the school will be performed on **Leonardo-Booster** module.

You have received a Username and a Password via mail for the login. Such credentials can not be changed and will last for all the days of the course and some more.

Requirements to login:

- If you are a **Linux** or **Mac** user, a standard ssh terminal would be sufficient.

- If you are a **Windows** user, you can use the prompt offered by Powershell or WSL, or an SSH client such as Putty or MobaXTerm

Login to Leonardo

To login, the command is simple:

```
ssh <username>@login.leonardo.cineca.it
```

It will ask for the password. The characters will be input but not displayed: don’t worry!!

After various logins, you may get an error saying
WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!

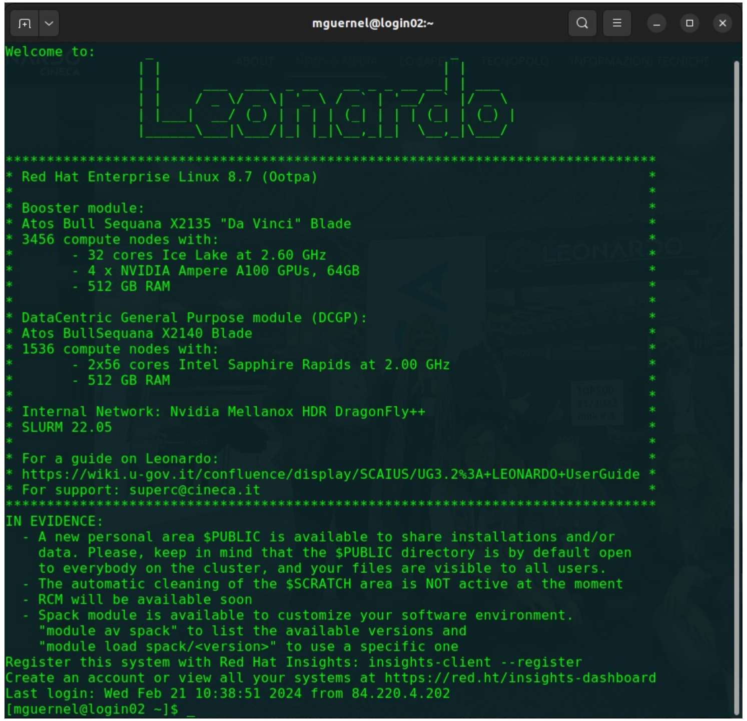
Please check our FAQs for the solution

WARNING: getting the username or password wrong for 3 times **blocks** the IP address from which the login was attempted for 30 minutes. **BE VERY CAREFUL**!

Login nodes, accounting and disk spaces

Login nodes

$ ssh username@login.leonardo.cineca.it

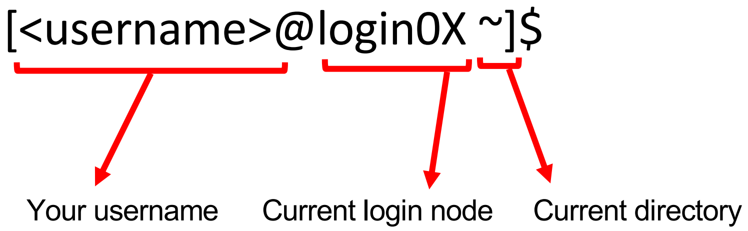


```
Welcome to:
                 _                                _
                | |                              | |
                | |     ___  ___  _ __   __ _ _ __ __| | ___
                | |    / _ \/ _ \| '_ \ / _` | '__/ _` |/ _ \
                | |___|  __/ (_) | | | | (_| | | | (_| | (_) |
                |______\___|\___/|_| |_|\__,_|_|  \__,_|\___/

*******************************************************************************
* Red Hat Enterprise Linux 8.7 (Ootpa)                                        *
*                                                                             *
* Booster module:                                                             *
* Atos Bull Sequana X2135 "Da Vinci" Blade                                    *
* 3456 compute nodes with:                                                    *
*       - 32 cores Ice Lake at 2.60 GHz                                       *
*       - 4 x NVIDIA Ampere A100 GPUs, 64GB                                   *
*       - 512 GB RAM                                                          *
*                                                                             *
* DataCentric General Purpose module (DCGP):                                  *
* Atos BullSequana X2140 Blade                                                *
* 1536 compute nodes with:                                                    *
*       - 2x56 cores Intel Sapphire Rapids at 2.00 GHz                        *
*       - 512 GB RAM                                                          *
*                                                                             *
* Internal Network: Nvidia Mellanox HDR DragonFly++                           *
* SLURM 22.05                                                                 *
*                                                                             *
* For a guide on Leonardo:                                                    *
* https://wiki.u-gov.it/confluence/display/SCAIUS/UG3.2%3A+LEONARDO+UserGuide *
* For support: superc@cineca.it                                               *
*******************************************************************************
IN EVIDENCE:
  - A new personal area $PUBLIC is available to share installations and/or
    data. Please, keep in mind that the $PUBLIC directory is by default open
    to everybody on the cluster, and your files are visible to all users.
  - The automatic cleaning of the $SCRATCH area is NOT active at the moment
  - RCM will be available soon
  - Spack module is available to customize your software environment.
    "module av spack" to list the available versions and
    "module load spack/<version>" to use a specific one
Register this system with Red Hat Insights: insights-client --register
Create an account or view all your systems at https://red.ht/insights-dashboard
Last login: Wed Feb 21 10:38:51 2024 from 84.220.4.202
[mguernel@login02 ~]$ _
```

Motto of the day

- Short system description
- System status
- “In evidence” messages
- “Important” messages

Bash shell



$$\underbrace{\text{[<username>}}_{\text{Your username}}\text{@}\underbrace{\text{login0X}}_{\text{Current login node}}\ \underbrace{\sim}_{\text{Current directory}}\text{]\$}$$

Login nodes

Leonardo (as the other CINECA HPC clusters) is shared among many users, so a

responsible use is crucial!

Login nodes

- The purpose of login nodes is mainly to perform small operations and submitting computing jobs;
- Interactive runs on login nodes are strongly discouraged and should be limited to short test runs:
- → ***10 minutes cpu-time limit***
- Avoid running large and parallel applications on login nodes;
- ***No GPUs on login nodes.***

Filesystems

$HOME

- 50 GB per user
- User specific
- Permanent (till user is active)
- Daily backup (**soon**)

$PUBLIC

- 50 GB per user
- User specific (permissions **755**)
- Permanent (till user is active)
- **No** backup

$FAST

- $WORK mirror
- Fast (I/O relevant appl.)

Data resources (DRES)

Shared area among different projects platforms.

$WORK

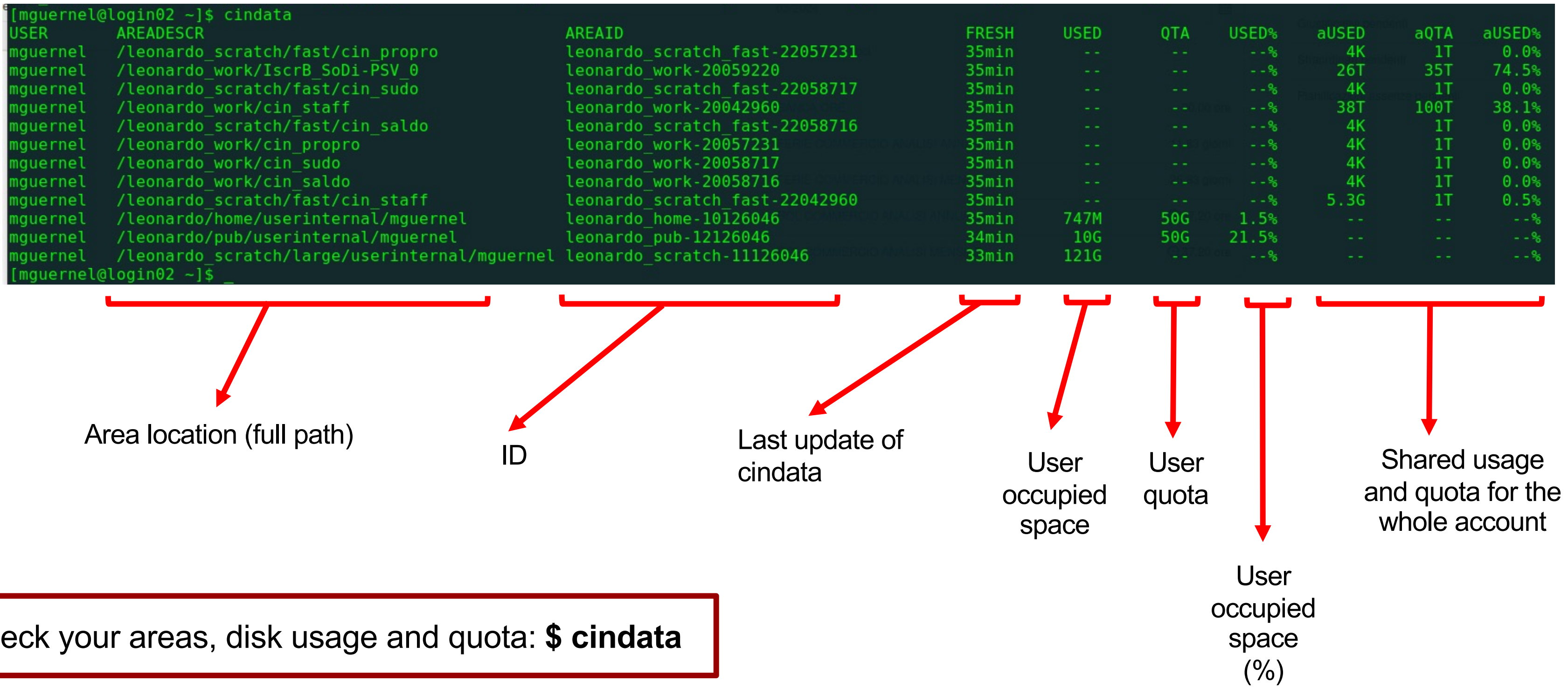
- Quota per account (default 1TB)
- Account specific
- Permanent (account + 6 month)
- **No** backup

$SCRATCH

- No quota
- User specific
- Temporary (data will be removed after 40 days, and no backup)

All the filesystems are based on **Lustre**

Filesystems



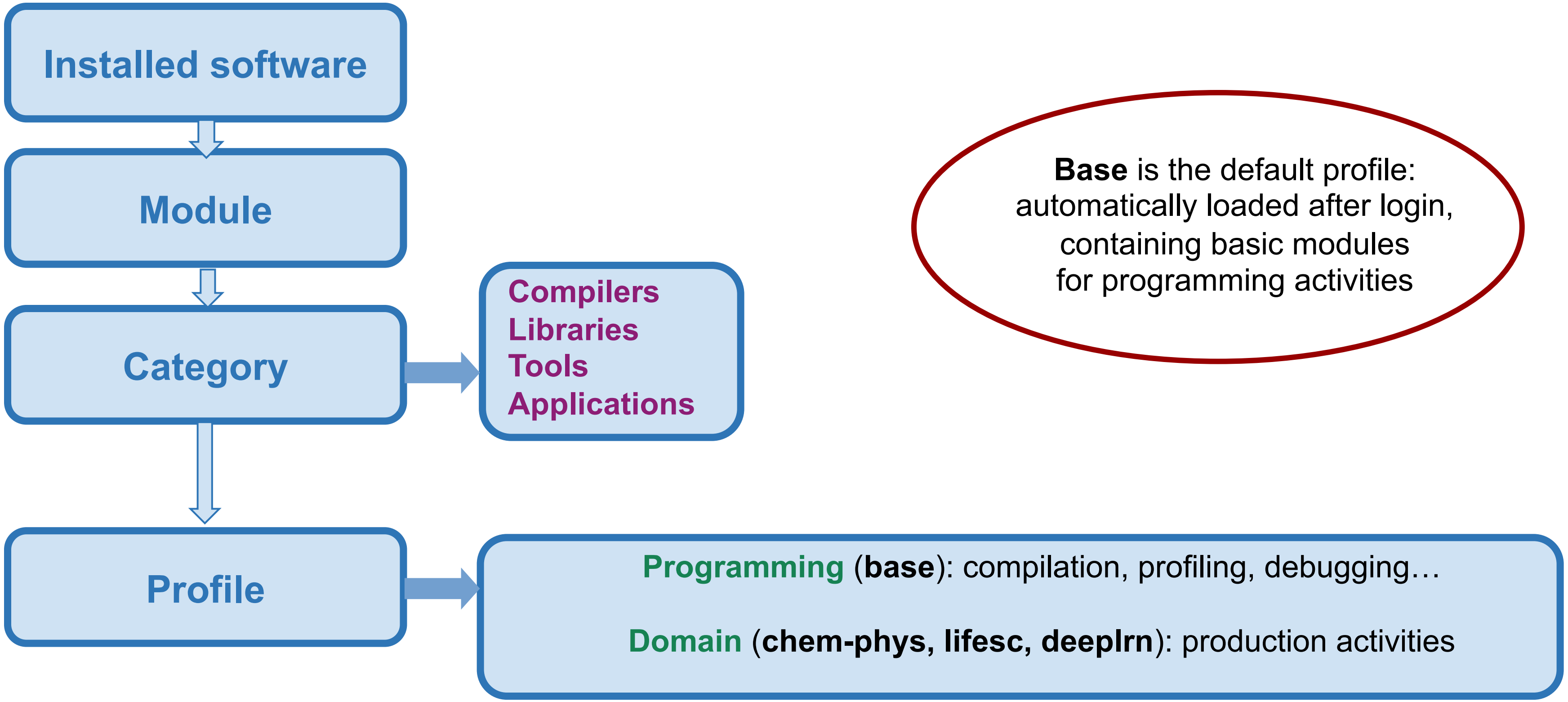
```
[mguernel@login02 ~]$ cindata
USER       AREADESCR                                     AREAID                                   FRESH     USED      QTA    USED%    aUSED     aQTA   aUSED%
mguernel   /leonardo_scratch/fast/cin_propro             leonardo_scratch_fast-22057231           35min       --       --      --%       4K       1T     0.0%
mguernel   /leonardo_work/IscrB_SoDi-PSV_0               leonardo_work-20059220                   35min       --       --      --%      26T      35T    74.5%
mguernel   /leonardo_scratch/fast/cin_sudo               leonardo_scratch_fast-22058717           35min       --       --      --%       4K       1T     0.0%
mguernel   /leonardo_work/cin_staff                      leonardo_work-20042960                   35min       --       --      --%      38T     100T    38.1%
mguernel   /leonardo_scratch/fast/cin_saldo              leonardo_scratch_fast-22058716           35min       --       --      --%       4K       1T     0.0%
mguernel   /leonardo_work/cin_propro                     leonardo_work-20057231                   35min       --       --      --%       4K       1T     0.0%
mguernel   /leonardo_work/cin_sudo                       leonardo_work-20058717                   35min       --       --      --%       4K       1T     0.0%
mguernel   /leonardo_work/cin_saldo                      leonardo_work-20058716                   35min       --       --      --%       4K       1T     0.0%
mguernel   /leonardo_scratch/fast/cin_staff              leonardo_scratch_fast-22042960           35min       --       --      --%     5.3G       1T     0.5%
mguernel   /leonardo/home/userinternal/mguernel          leonardo_home-10126046                   35min     747M      50G     1.5%       --       --      --%
mguernel   /leonardo/pub/userinternal/mguernel           leonardo_pub-12126046                    34min      10G      50G    21.5%       --       --      --%
mguernel   /leonardo_scratch/large/userinternal/mguernel leonardo_scratch-11126046                33min     121G       --      --%       --       --      --%
[mguernel@login02 ~]$ _
```

Check your areas, disk usage and quota: **$ cindata**

Software Modules

Module environment

Any available software is offered on Leonardo in a **module environment**.
The modules are organized in functional **categories** and collected in different **profiles**.



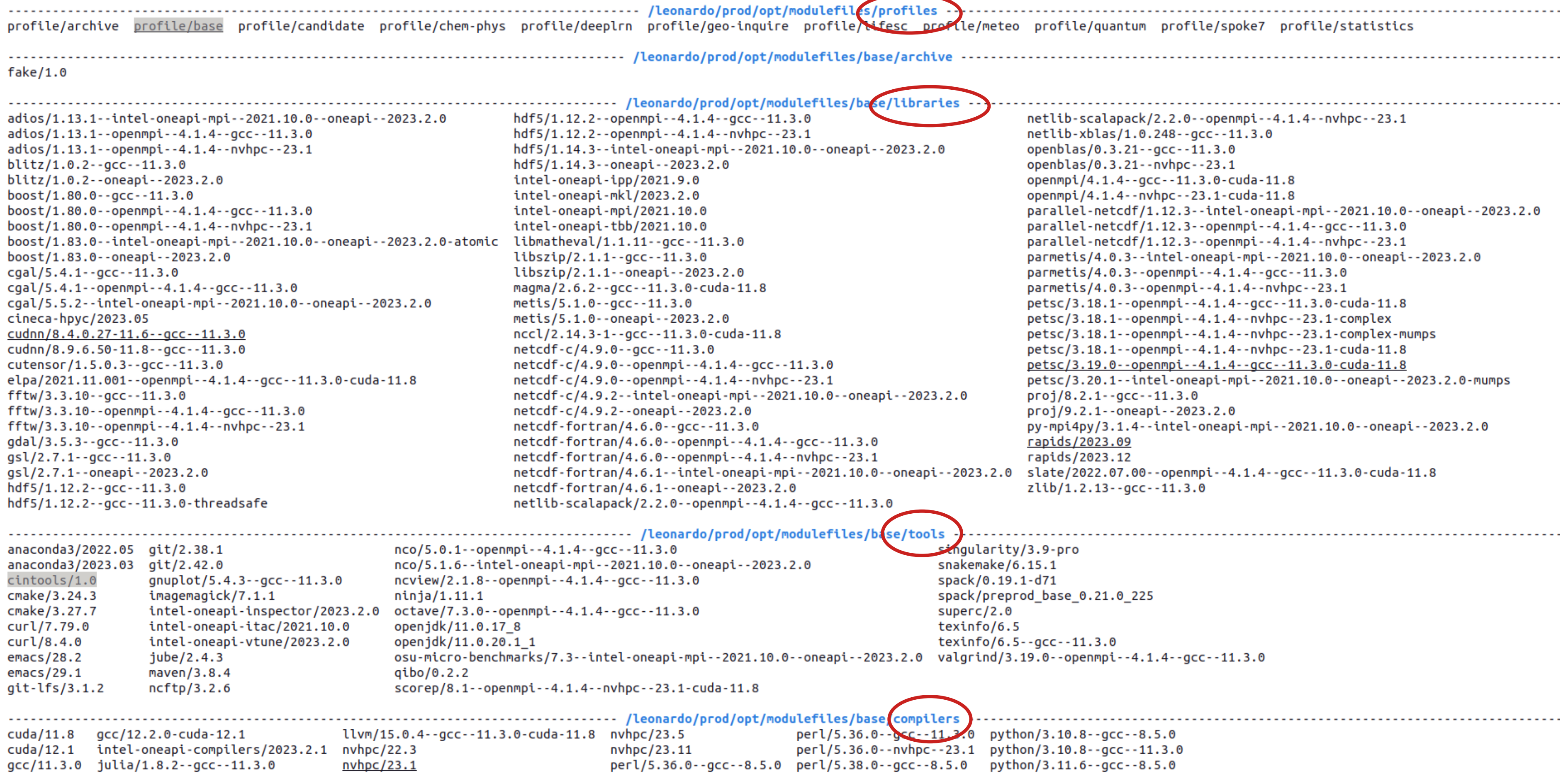
Programming (**base**): compilation, profiling, debugging…

Domain (**chem-phys, lifesc, deeplrn**): production activities

Base is the default profile:
automatically loaded after login,
containing basic modules
for programming activities

Available modules

$ module av

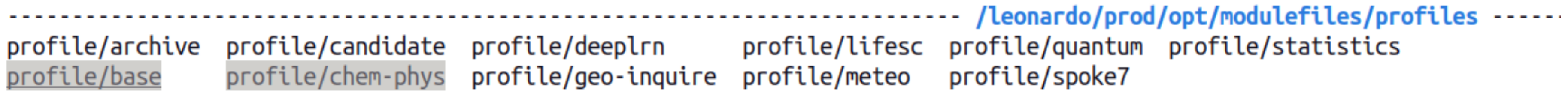


```
-------------------------------------------------------------------------- /leonardo/prod/opt/modulefiles/profiles ---------------------------------------------------------------------------
profile/archive   profile/base   profile/candidate   profile/chem-phys   profile/deeplrn   profile/geo-inquire   profile/lifesc   profile/meteo   profile/quantum   profile/spoke7   profile/statistics

------------------------------------------------------------------------ /leonardo/prod/opt/modulefiles/base/archive -------------------------------------------------------------------------
fake/1.0

----------------------------------------------------------------------- /leonardo/prod/opt/modulefiles/base/libraries ------------------------------------------------------------------------
adios/1.13.1--intel-oneapi-mpi--2021.10.0--oneapi--2023.2.0    hdf5/1.12.2--openmpi--4.1.4--gcc--11.3.0                             netlib-scalapack/2.2.0--openmpi--4.1.4--nvhpc--23.1
adios/1.13.1--openmpi--4.1.4--gcc--11.3.0                      hdf5/1.12.2--openmpi--4.1.4--nvhpc--23.1                             netlib-xblas/1.0.248--gcc--11.3.0
adios/1.13.1--openmpi--4.1.4--nvhpc--23.1                      hdf5/1.14.3--intel-oneapi-mpi--2021.10.0--oneapi--2023.2.0           openblas/0.3.21--gcc--11.3.0
blitz/1.0.2--gcc--11.3.0                                       hdf5/1.14.3--oneapi--2023.2.0                                        openblas/0.3.21--nvhpc--23.1
blitz/1.0.2--oneapi--2023.2.0                                  intel-oneapi-ipp/2021.9.0                                            openmpi/4.1.4--gcc--11.3.0-cuda-11.8
boost/1.80.0--gcc--11.3.0                                      intel-oneapi-mkl/2023.2.0                                            openmpi/4.1.4--nvhpc--23.1-cuda-11.8
boost/1.80.0--openmpi--4.1.4--gcc--11.3.0                      intel-oneapi-mpi/2021.10.0                                           parallel-netcdf/1.12.3--intel-oneapi-mpi--2021.10.0--oneapi--2023.2.0
boost/1.80.0--openmpi--4.1.4--nvhpc--23.1                      intel-oneapi-tbb/2021.10.0                                           parallel-netcdf/1.12.3--openmpi--4.1.4--gcc--11.3.0
boost/1.83.0--intel-oneapi-mpi--2021.10.0--oneapi--2023.2.0-atomic libmatheval/1.1.11--gcc--11.3.0                                  parallel-netcdf/1.12.3--openmpi--4.1.4--nvhpc--23.1
boost/1.83.0--oneapi--2023.2.0                                 libszip/2.1.1--gcc--11.3.0                                           parmetis/4.0.3--intel-oneapi-mpi--2021.10.0--oneapi--2023.2.0
cgal/5.4.1--gcc--11.3.0                                        libszip/2.1.1--oneapi--2023.2.0                                      parmetis/4.0.3--openmpi--4.1.4--gcc--11.3.0
cgal/5.4.1--openmpi--4.1.4--gcc--11.3.0                        magma/2.6.2--gcc--11.3.0-cuda-11.8                                   parmetis/4.0.3--openmpi--4.1.4--nvhpc--23.1
cgal/5.5.2--intel-oneapi-mpi--2021.10.0--oneapi--2023.2.0      metis/5.1.0--gcc--11.3.0                                             petsc/3.18.1--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
cineca-hpyc/2023.05                                            metis/5.1.0--oneapi--2023.2.0                                        petsc/3.18.1--openmpi--4.1.4--nvhpc--23.1-complex
cudnn/8.4.0.27-11.6--gcc--11.3.0                               nccl/2.14.3-1--gcc--11.3.0-cuda-11.8                                 petsc/3.18.1--openmpi--4.1.4--nvhpc--23.1-complex-mumps
cudnn/8.9.6.50-11.8--gcc--11.3.0                               netcdf-c/4.9.0--gcc--11.3.0                                          petsc/3.18.1--openmpi--4.1.4--nvhpc--23.1-cuda-11.8
cutensor/1.5.0.3--gcc--11.3.0                                  netcdf-c/4.9.0--openmpi--4.1.4--gcc--11.3.0                          petsc/3.19.0--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
elpa/2021.11.001--openmpi--4.1.4--gcc--11.3.0-cuda-11.8        netcdf-c/4.9.0--openmpi--4.1.4--nvhpc--23.1                          petsc/3.20.1--intel-oneapi-mpi--2021.10.0--oneapi--2023.2.0-mumps
fftw/3.3.10--gcc--11.3.0                                       netcdf-c/4.9.2--intel-oneapi-mpi--2021.10.0--oneapi--2023.2.0        proj/8.2.1--gcc--11.3.0
fftw/3.3.10--openmpi--4.1.4--gcc--11.3.0                       netcdf-c/4.9.2--oneapi--2023.2.0                                     proj/9.2.1--oneapi--2023.2.0
fftw/3.3.10--openmpi--4.1.4--nvhpc--23.1                       netcdf-fortran/4.6.0--gcc--11.3.0                                    py-mpi4py/3.1.4--intel-oneapi-mpi--2021.10.0--oneapi--2023.2.0
gdal/3.5.3--gcc--11.3.0                                        netcdf-fortran/4.6.0--openmpi--4.1.4--gcc--11.3.0                    rapids/2023.09
gsl/2.7.1--gcc--11.3.0                                         netcdf-fortran/4.6.0--openmpi--4.1.4--nvhpc--23.1                    rapids/2023.12
gsl/2.7.1--oneapi--2023.2.0                                    netcdf-fortran/4.6.1--intel-oneapi-mpi--2021.10.0--oneapi--2023.2.0  slate/2022.07.00--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
hdf5/1.12.2--gcc--11.3.0                                       netcdf-fortran/4.6.1--oneapi--2023.2.0                               zlib/1.2.13--gcc--11.3.0
hdf5/1.12.2--gcc--11.3.0-threadsafe                            netlib-scalapack/2.2.0--openmpi--4.1.4--gcc--11.3.0

------------------------------------------------------------------------- /leonardo/prod/opt/modulefiles/base/tools --------------------------------------------------------------------------
anaconda3/2022.05  git/2.38.1                      nco/5.0.1--openmpi--4.1.4--gcc--11.3.0                                    singularity/3.9-pro
anaconda3/2023.03  git/2.42.0                      nco/5.1.6--intel-oneapi-mpi--2021.10.0--oneapi--2023.2.0                  snakemake/6.15.1
cintools/1.0       gnuplot/5.4.3--gcc--11.3.0      ncview/2.1.8--openmpi--4.1.4--gcc--11.3.0                                 spack/0.19.1-d71
cmake/3.24.3       imagemagick/7.1.1               ninja/1.11.1                                                              spack/preprod_base_0.21.0_225
cmake/3.27.7       intel-oneapi-inspector/2023.2.0 octave/7.3.0--openmpi--4.1.4--gcc--11.3.0                                 superc/2.0
curl/7.79.0        intel-oneapi-itac/2021.10.0     openjdk/11.0.17_8                                                         texinfo/6.5
curl/8.4.0         intel-oneapi-vtune/2023.2.0     openjdk/11.0.20.1_1                                                       texinfo/6.5--gcc--11.3.0
emacs/28.2         jube/2.4.3                      osu-micro-benchmarks/7.3--intel-oneapi-mpi--2021.10.0--oneapi--2023.2.0   valgrind/3.19.0--openmpi--4.1.4--gcc--11.3.0
emacs/29.1         maven/3.8.4                     qibo/0.2.2
git-lfs/3.1.2      ncftp/3.2.6                     scorep/8.1--openmpi--4.1.4--nvhpc--23.1-cuda-11.8

----------------------------------------------------------------------- /leonardo/prod/opt/modulefiles/base/compilers ------------------------------------------------------------------------
cuda/11.8   gcc/12.2.0-cuda-12.1             llvm/15.0.4--gcc--11.3.0-cuda-11.8   nvhpc/23.5               perl/5.36.0--gcc--11.3.0   python/3.10.8--gcc--8.5.0
cuda/12.1   intel-oneapi-compilers/2023.2.1  nvhpc/22.3                           nvhpc/23.11              perl/5.36.0--nvhpc--23.1   python/3.10.8--gcc--11.3.0
gcc/11.3.0  julia/1.8.2--gcc--11.3.0         nvhpc/23.1                           perl/5.36.0--gcc--8.5.0  perl/5.38.0--gcc--8.5.0    python/3.11.6--gcc--8.5.0
```

How to load/unload modules

How to **load** additional profiles/modules?

```
$ module load profile/chem-phys
```



```
------------------------------------------------------------- /leonardo/prod/opt/modulefiles/profiles -----
profile/archive   profile/candidate  profile/deeplrn      profile/lifesc  profile/quantum  profile/statistics
profile/base      profile/chem-phys  profile/geo-inquire  profile/meteo   profile/spoke7
```

Which modules have **I already loaded**?

```
$ module list
Currently Loaded Modulefiles:
 1) profile/base   2) profile/chem-phys
```

How to **unload** a profile/module?

Specific profile/module

```
$ module unload profile/chem-phys
```

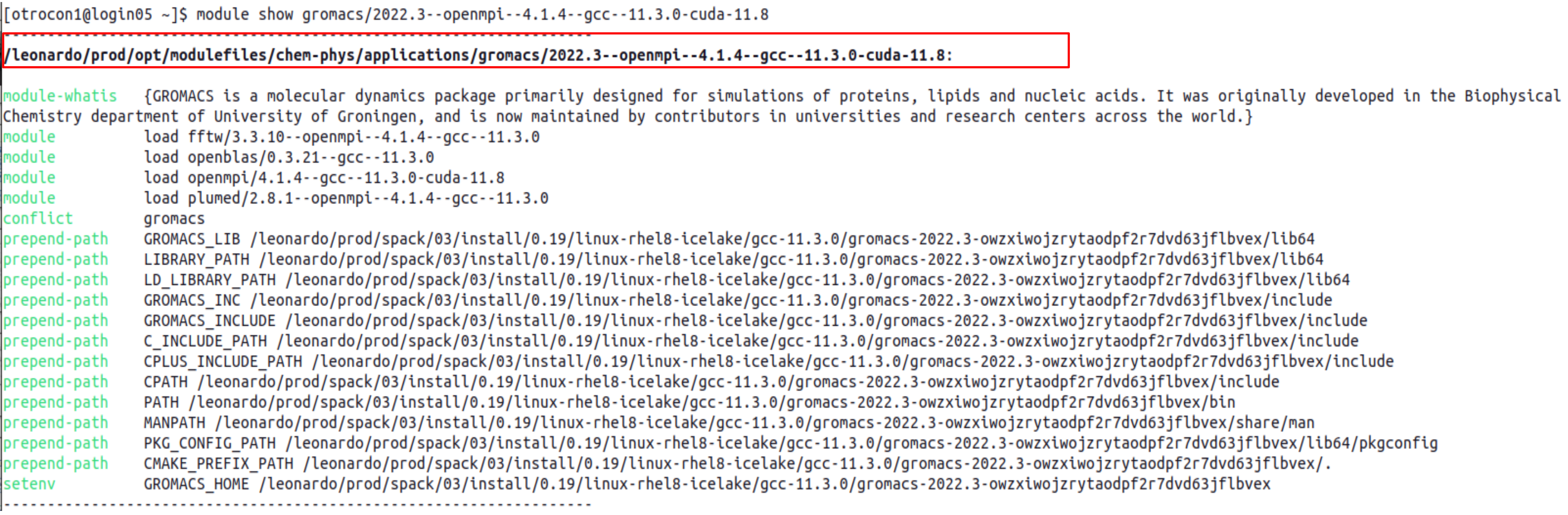
Unload all of them

```
$ module purge
```

Effects of loading a module

A module defines and/or modifies environment variables, allowing to use other executables or libraries

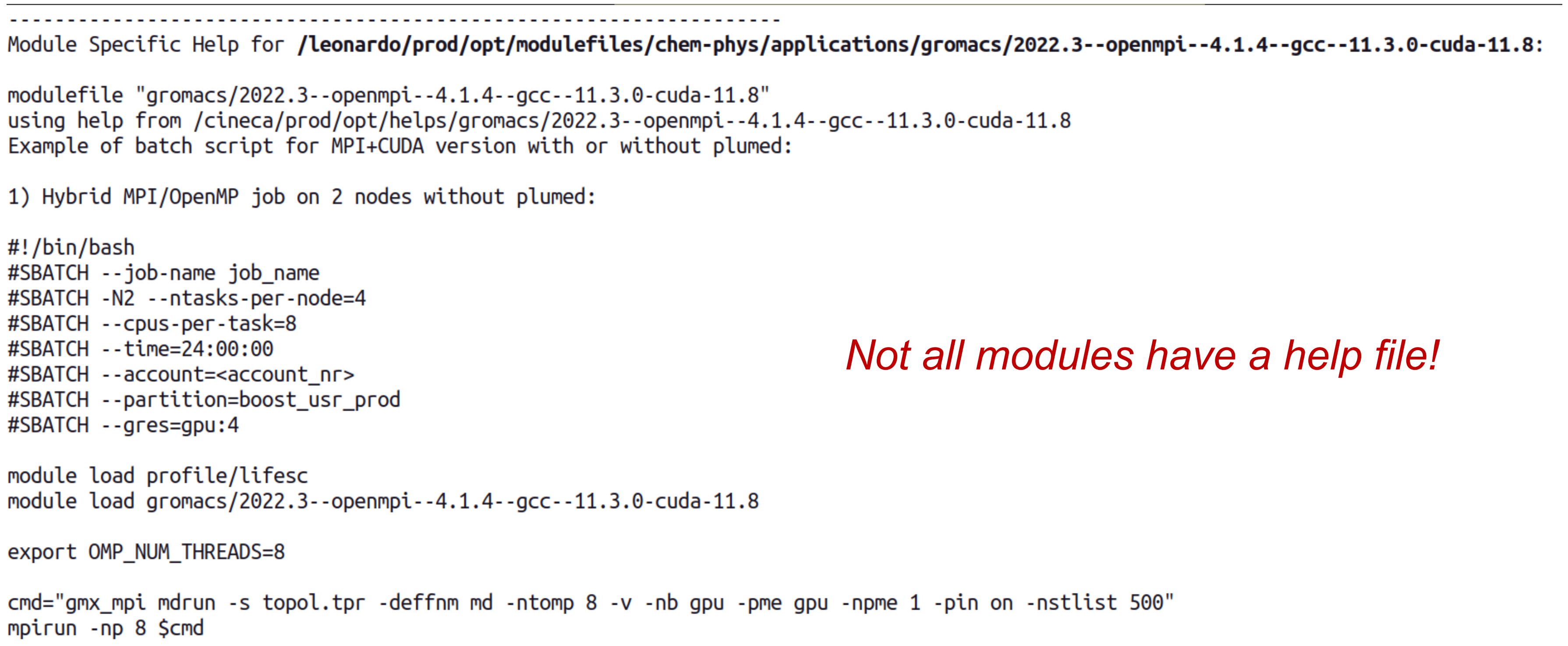
$ module show <module_name>/<version>



```
[otrocon1@login05 ~]$ module show gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
-------------------------------------------------------------------
/leonardo/prod/opt/modulefiles/chem-phys/applications/gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8:

module-whatis   {GROMACS is a molecular dynamics package primarily designed for simulations of proteins, lipids and nucleic acids. It was originally developed in the Biophysical
Chemistry department of University of Groningen, and is now maintained by contributors in universities and research centers across the world.}
module          load fftw/3.3.10--openmpi--4.1.4--gcc--11.3.0
module          load openblas/0.3.21--gcc--11.3.0
module          load openmpi/4.1.4--gcc--11.3.0-cuda-11.8
module          load plumed/2.8.1--openmpi--4.1.4--gcc--11.3.0
conflict        gromacs
prepend-path    GROMACS_LIB /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/lib64
prepend-path    LIBRARY_PATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/lib64
prepend-path    LD_LIBRARY_PATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/lib64
prepend-path    GROMACS_INC /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/include
prepend-path    GROMACS_INCLUDE /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/include
prepend-path    C_INCLUDE_PATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/include
prepend-path    CPLUS_INCLUDE_PATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/include
prepend-path    CPATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/include
prepend-path    PATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/bin
prepend-path    MANPATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/share/man
prepend-path    PKG_CONFIG_PATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/lib64/pkgconfig
prepend-path    CMAKE_PREFIX_PATH /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex/.
setenv          GROMACS_HOME /leonardo/prod/spack/03/install/0.19/linux-rhel8-icelake/gcc-11.3.0/gromacs-2022.3-owzxiwojzrytaodpf2r7dvd63jflbvex
-------------------------------------------------------------------
```

Module help

```
$ module load profile/lifesc
$ module help gromacs/2022.3—openmpi--4.1.4--gcc--11.3.0-cuda-11.8
```



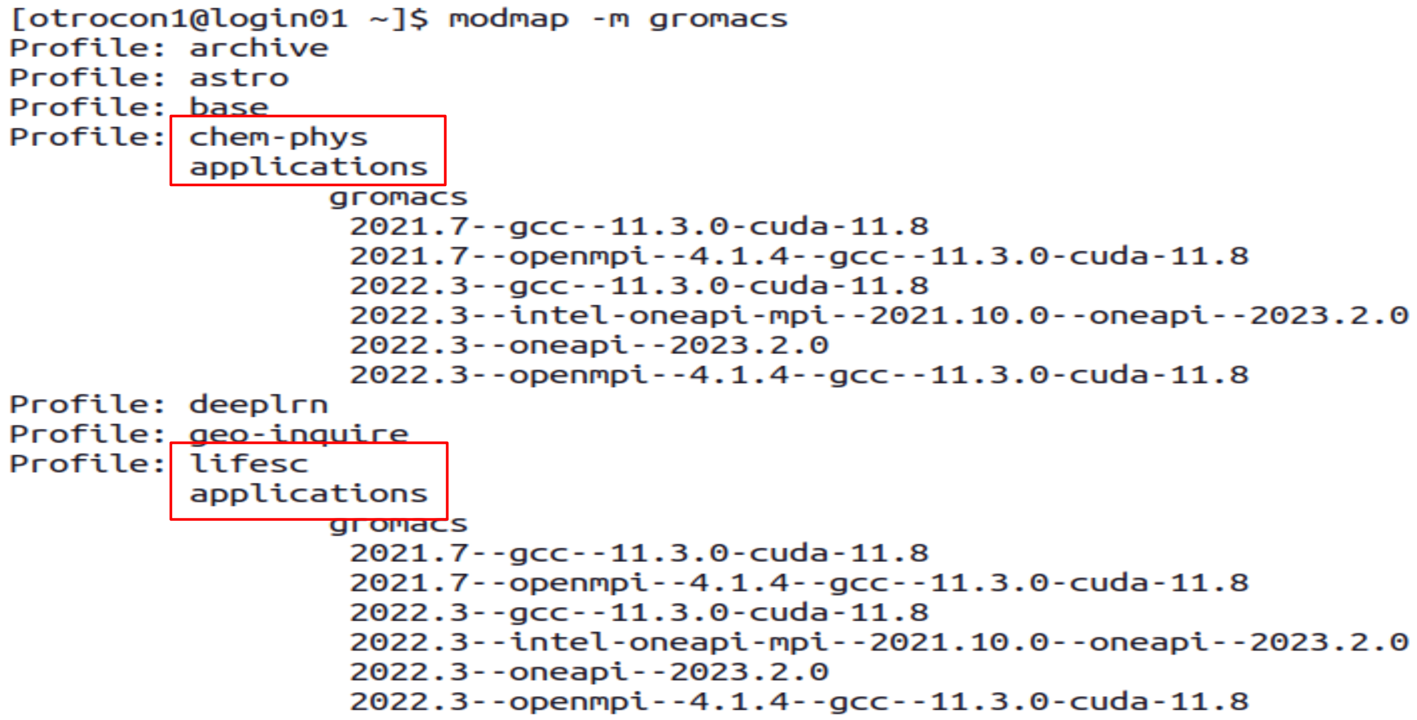
```
-------------------------------------------------------------------
Module Specific Help for /leonardo/prod/opt/modulefiles/chem-phys/applications/gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8:

modulefile "gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8"
using help from /cineca/prod/opt/helps/gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
Example of batch script for MPI+CUDA version with or without plumed:

1) Hybrid MPI/OpenMP job on 2 nodes without plumed:

#!/bin/bash
#SBATCH --job-name job_name
#SBATCH -N2 --ntasks-per-node=4
#SBATCH --cpus-per-task=8
#SBATCH --time=24:00:00
#SBATCH --account=<account_nr>
#SBATCH --partition=boost_usr_prod
#SBATCH --gres=gpu:4

module load profile/lifesc
module load gromacs/2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8

export OMP_NUM_THREADS=8

cmd="gmx_mpi mdrun -s topol.tpr -deffnm md -ntomp 8 -v -nb gpu -pme gpu -npme 1 -pin on -nstlist 500"
mpirun -np 8 $cmd
```

Not all modules have a help file!

How to search for a module

How to find a module that I do not know in which profile is it?

$ modmap -m <module_name>   a command that looks for a module in all profiles



```
[otrocon1@login01 ~]$ modmap -m gromacs
Profile: archive
Profile: astro
Profile: base
Profile: chem-phys
         applications
                gromacs
                 2021.7--gcc--11.3.0-cuda-11.8
                 2021.7--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
                 2022.3--gcc--11.3.0-cuda-11.8
                 2022.3--intel-oneapi-mpi--2021.10.0--oneapi--2023.2.0
                 2022.3--oneapi--2023.2.0
                 2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
Profile: deeplrn
Profile: geo-inquire
Profile: lifesc
         applications
                gromacs
                 2021.7--gcc--11.3.0-cuda-11.8
                 2021.7--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
                 2022.3--gcc--11.3.0-cuda-11.8
                 2022.3--intel-oneapi-mpi--2021.10.0--oneapi--2023.2.0
                 2022.3--oneapi--2023.2.0
                 2022.3--openmpi--4.1.4--gcc--11.3.0-cuda-11.8
```

Compiling on Leonardo

In LEONARDO you can choose between three different compiler families: **gcc, intel** and **nvhpc (for GPUs)**

You can take a look at the versions available with “*module av” and then* load the module you want.

```
module load gcc            # loads default gnu compilers suite
module load gcc/12.2.0     # loads specific compilers suite
```

COMPILERS (Fortran/C/C++):
GNU: gfortran/gcc/g++
INTEL: ifx/icx/icpx
NVHPC: nvfortran/nvcc/nvc+

Get a list of the compiler flags with the command **man**

Parallel compiling on Leonardo

MPI libraries available: **OpenMPI/IntelMPI**

The library and special wrappers to compile and link the personal programs are contained in several modules, one for each supported suite of compilers

Load a version of **OpenMPI**:

```
modmap -m openmpi
      4.1.6--gcc--12.2.0
module load openmpi/4.1.6--gcc--12.2.0
```

Load a version of **IntelMPI**:

```
modmap -m intel-oneapi-mpi
            2021.7.1
module load intel-oneapi-mpi/2021.7.1
```

COMPILERS (Fortran/C/C++):
OPENMPI: mpif90/mpicc/mpiCC
INTELMPI: mpiifx/mpiicx/mpiicpx

OpenMP is provided with the following compiler flags:
gnu: -fopenmp
intel : -qopenmp
hpc-sdk: -mp

GPU compiling on Leonardo

DISCLAIMER: In the slides for GPU management I will be as basic as possible. Better and more aimed teachings about the subject will come from the teachers of the following days (if they contradict me, listen to them!)

To compile codes suited for GPU application (*.cu), we have to load the specific compiler module, **CUDA** or its latest upgrade called **NVHPC**
CUDA is available on profile/base:

```
--- /leonardo/prod/opt/modulefiles/base/compilers ---
cuda/12.1  cuda/12.2  cuda/12.3
```

Load the module and use the CUDA compiler, nvcc:

```
nvcc mycudacode.cu -o mycudaexe.x
```

Scheduler

Login nodes

If you have a serial program, the most intuitive thing to do is to just launch `./myprogram` wherever you are.

When you log into LEONARDO, you find yourself in one of the four login nodes, selcted in round robin fashion to balance out the load of users.

Interactive runs on login nodes are strongly discouraged and should be limited to **short test runs**

- **There are per user limits** on cpu-time (10 minutes)
- **IMPORTANT: avoid running large parallel applications** on the front-ends!!

LEONARDO is a general purpose system used by hundreds of users.

Compute nodes: jobs & scheduler

What we actually want most of the time is to gain access to the compute nodes to exploit their power

Like in any HPC cluster, LEONARDO allows you to run your simulations by submitting **"jobs"** to the compute nodes

Your job is then taken in consideration by a **scheduler**, that adds it to a queuing line and allows its execution when the resources required are available

The operative scheduler on LEONARDO is **SLURM**

SLURM stands for "Simple Linux Utility for Resource Management"

- Allocating access to resources
- Job starting, executing and monitoring
- Queue of pending jobs management

Compute nodes: jobs & scheduler

The scheme for a SLURM job script is as follows:

#!/bin/bash

#SLURM directives

variables environment

execution line

Jobscript example

```
#!/bin/bash
#SBATCH --job-name=my_first_job
#SBATCH --output=job.out
#SBATCH --error=job.err
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=4
#SBATCH –gres:gpu=4
#SBATCH --cpus-per-task=8
#SBATCH --mem=0
#SBATCH --time=1-00:00:00
#SBATCH --partition=boost_usr_prod
#SBATCH --account=<my_account>

module load openmpi/4.1.4--gcc--11.3.0-cuda-11.8
export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK
mpirun -n $SLURM_NTASKS ./my_program
```

Slurm directives

#SBATCH --job-name=myname, -J myname

Defines the name of your job

#SBATCH --output=job.out, -o job.out

Specifies the file where the standard output is directed (default=slurm-%j.out)

#SBATCH --error=job.err, -e job.err

Specifies the file where the standard error is directed (default=slurm-%j.out)

#SBATCH --mail-type=ALL (optional)

Specifies e-mail notification. An e-mail will be sent to you when something happens to your job, according to the keywords you specified (NONE, BEGIN, END, FAIL, ALL)

#SBATCH --mail-user=user@email.com (optional)

Specifies the e-mail address for the keyword above

Slurm directives: resource requirements

```
#SBATCH --nodes=2, -N 2
#SBATCH --ntasks-per-node=4        # Be careful, this is not equal to -n=4
#SBATCH --cpus-per-task=8
#SBATCH --mem=100GB                # mem=0 to ask for all the mem of the node
#SBATCH –gres:gpu=4
```

nodes – number of compute nodes
ntasks-per-node – number of tasks per node
cpus-per-task – number of cpus to be assigned to each task
mem – RAM memory allocated for each node (max=494000 MB)

gpus-per-node – number of GPUs for each node

Slurm directives: walltime and partitions

#SBATCH --time=00:30:00, -t 00:30:00

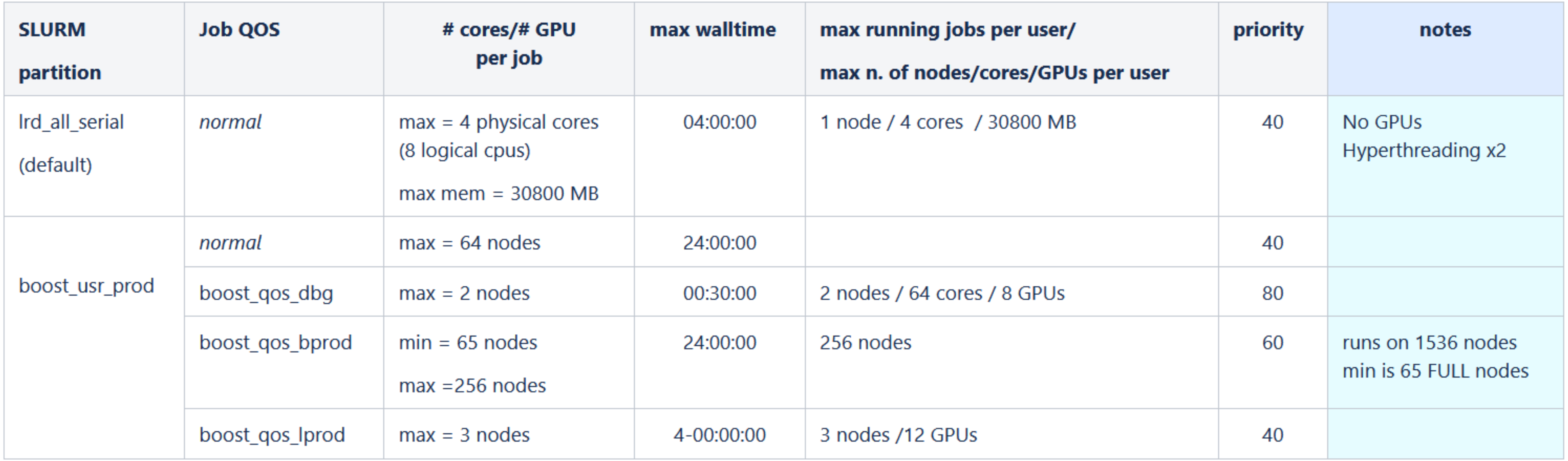
Specifies the maximum duration of the job. The maximum time allowed depends on the partition used

Pro-tip: the less walltime you ask, the faster your job will enter in execution. Think about it!

#SBATCH --partition=boost_usr_prod, -p boost_usr_prod
#SBATCH --qos=boost_qos_dbg, -q boost_qos_dbg (optional)

Specifies the “partition”, a.k.a. the specific set of nodes among which your job can search for resources. Optionally you can specify a QoS (Quality of Service) for jobs with particular purposes, like debugging or large production

Available partitions and QoS on LEONARDO Booster



SLURM partition	Job QOS	# cores/# GPU per job	max walltime	max running jobs per user/ max n. of nodes/cores/GPUs per user	priority	notes
lrd_all_serial (default)	*normal*	max = 4 physical cores (8 logical cpus) max mem = 30800 MB	04:00:00	1 node / 4 cores / 30800 MB	40	No GPUs Hyperthreading x2
boost_usr_prod	*normal*	max = 64 nodes	24:00:00		40	
	boost_qos_dbg	max = 2 nodes	00:30:00	2 nodes / 64 cores / 8 GPUs	80	
	boost_qos_bprod	min = 65 nodes max =256 nodes	24:00:00	256 nodes	60	runs on 1536 nodes min is 65 FULL nodes
	boost_qos_lprod	max = 3 nodes	4-00:00:00	3 nodes /12 GPUs	40	

Notes:
- the partition **lrd_all_serial** runs on front-end nodes, and as such it is not subject to accounting and can be used for free
- to use the QoS **boost_qos_bprod**, the minimum requirement is for cpus, gpus, nodes and mem to be over the regular limit…that's why the nodes have to be asked in full!

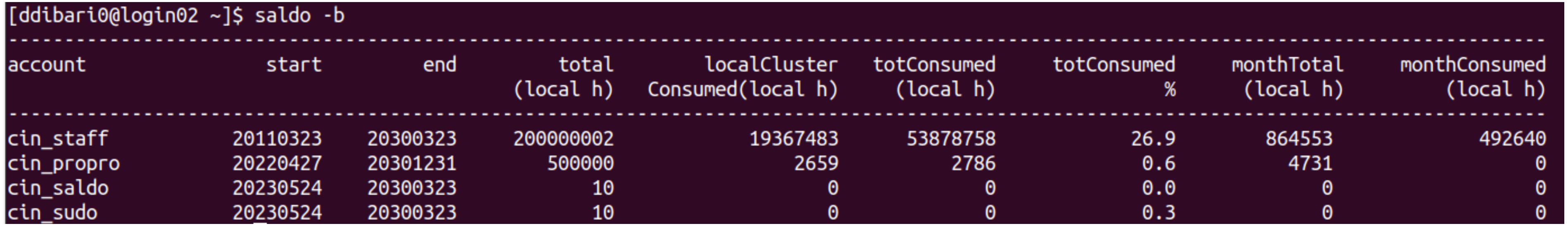
Slurm directives: accounting

#SBATCH --account=<my_account>, -A <my_account>

Specifies the account to use the CPU hours from.

As user, you have access to a limited number of CPU hours to spend. They are not assigned to users, but to **projects** and are shared between the users who are working on the same project (i.e. your research partners). Such projects are called **accounts** and are a different concept from your username.

You can check the status of your account with the command "*saldo -b*", which tells you how many CPU hours you have already consumed for each account you're assigned at (a more detailed report is provided by "*saldo -r*").



```
[ddibari0@login02 ~]$ saldo -b
---------------------------------------------------------------------------------------------------------------------------
account                start        end          total        localCluster  totConsumed    totConsumed   monthTotal  monthConsumed
                                            (local h)   Consumed(local h)    (local h)              %    (local h)      (local h)
---------------------------------------------------------------------------------------------------------------------------
cin_staff           20110323   20300323    200000002          19367483      53878758           26.9       864553         492640
cin_propro          20220427   20301231       500000              2659          2786            0.6         4731              0
cin_saldo           20230524   20300323           10                 0             0            0.0            0              0
cin_sudo            20230524   20300323           10                 0             0            0.3            0              0
```

Account for the school

The account provided for this school is “**tra25_advgpu**“
(you have to specify it on your job scripts).

It will expire on Sunday, April 20th (there is a big maintenance coming up from April 7th to 16th)

WARNING! The account is shared between all the students;
there are plenty of hours for everybody,
but don’t waste them!

```
#SBATCH --account=tra25_ictp_rom
```

Environment and execution line

```
srun [options]  ./myprogram
mpirun [options] ./myprogram
```

Your parallel executable is launched on the compute nodes via the command *"srun"* or *"mpirun",* with all the tasks requested via resource allocation.
If your executable is not parallel, a simple *./myprogram* will do.

WARNING:
For parallel jobs, **openmpi has to be loaded** inside the job script:

```
module load openmpi/<module_version>
```

Be sure to load the same version of the compiler that you used to compile your code!!

Simple job for using GPUs

```
#!/bin/bash
#SBATCH --time=1:00:00
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --mem=10GB
#SBATCH --partition=boost_usr_prod
#SBATCH --gres=gpu:1
#SBATCH --account=<my_account>

./mycudaexe.x
```

You can ask for up to 4 GPUs for each node (there are 4 A100 on each node).
If they aren't asked, SLURM won't allocate them for you even if they are in the node you are using.

You don't have to load the cuda module inside your jobscript!

Submitting a job

You have created your jobscript! …and now?

sbatch

```
sbatch <job_script>
```

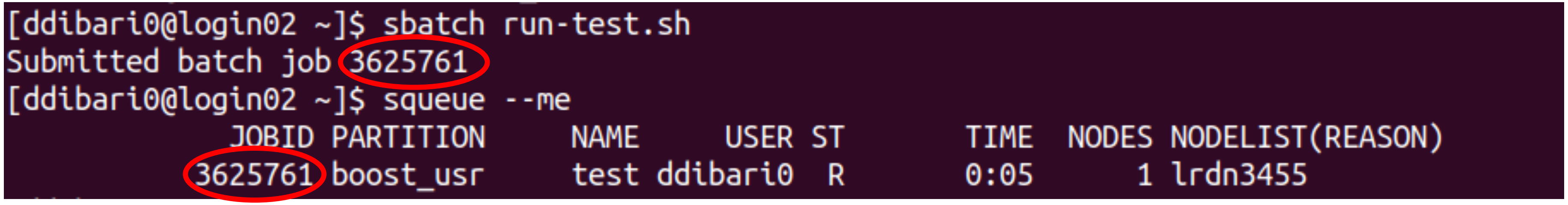
Your job will be submitted to the SLURM scheduler and executed when there will be nodes available (according to your priority and the partition you requested)

squeue

```
squeue -u <username>, --me
```

Shows the list of all your scheduled jobs, along with their status (idle, running, closing, …)
It also shows you the job id required for other SLURM commands

Other Slurm commands



```
[ddibari0@login02 ~]$ sbatch run-test.sh
Submitted batch job 3625761
[ddibari0@login02 ~]$ squeue --me
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           3625761 boost_usr     test ddibari0  R       0:05      1 lrdn3455
```

scontrol show job

scontrol show job <job_id>

Provides a long list of informations for the job requested.

In particular, if your job isn't running yet, you'll be notified about the reason it is not starting and, if it is scheduled with top priority, you will get an estimated start time

scancel

scancel <job_id>, --me

Removes the job (queued or running) from the scheduled job list by killing it

Interactive batch jobs

In case you need to “interact” with your running job (tuning of input parameters, debugging etc.)

and it needs more than 10 minutes, or many processes (not suitable on the login nodes) you can submit an

“Interactive” SLURM batch job

Ask for the needed resources (cores, gpus, memory, time) with **srun** or **salloc**.
The job is queued and scheduled as any other job but, when, executed, the standard input, output, and error streams are connected to the terminal session from which srun or salloc were launched.
You can then run your application from that terminal

Non MPI programs (single process or multi-threaded programs using one or more GPUs)

```
$ srun <options> --pty bash
```

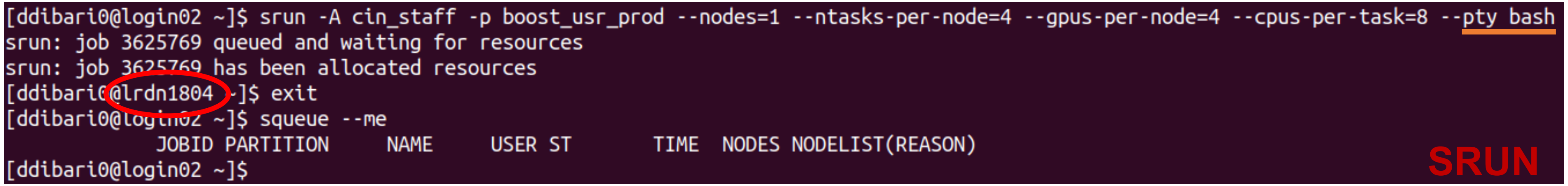
The session starts on the compute node (look at the prompt)

MPI programs using one or more GPUs

```
$ salloc <options>
```

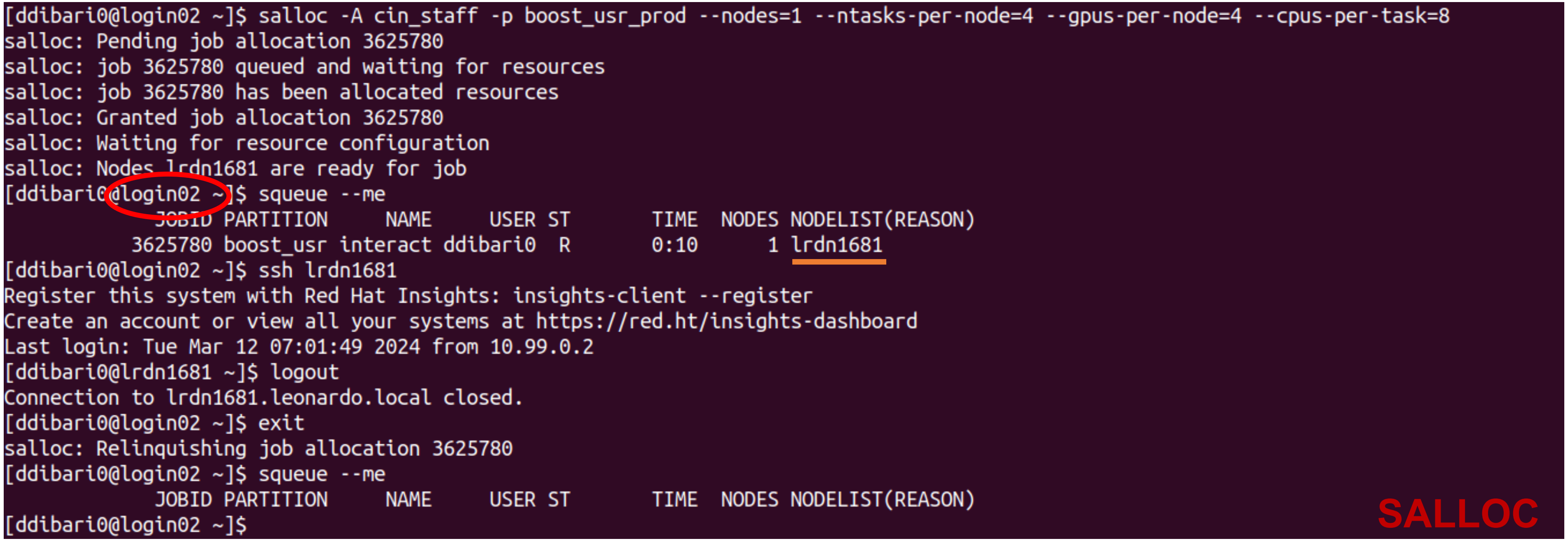
A new session is started on the login node

Interactive batch jobs



```
[ddibari0@login02 ~]$ srun -A cin_staff -p boost_usr_prod --nodes=1 --ntasks-per-node=4 --gpus-per-node=4 --cpus-per-task=8 --pty bash
srun: job 3625769 queued and waiting for resources
srun: job 3625769 has been allocated resources
[ddibari0@lrdn1804 ~]$ exit
[ddibari0@login02 ~]$ squeue --me
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
[ddibari0@login02 ~]$
```

SRUN



```
[ddibari0@login02 ~]$ salloc -A cin_staff -p boost_usr_prod --nodes=1 --ntasks-per-node=4 --gpus-per-node=4 --cpus-per-task=8
salloc: Pending job allocation 3625780
salloc: job 3625780 queued and waiting for resources
salloc: job 3625780 has been allocated resources
salloc: Granted job allocation 3625780
salloc: Waiting for resource configuration
salloc: Nodes lrdn1681 are ready for job
[ddibari0@login02 ~]$ squeue --me
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
           3625780 boost_usr interact ddibari0  R       0:10      1 lrdn1681
[ddibari0@login02 ~]$ ssh lrdn1681
Register this system with Red Hat Insights: insights-client --register
Create an account or view all your systems at https://red.ht/insights-dashboard
Last login: Tue Mar 12 07:01:49 2024 from 10.99.0.2
[ddibari0@lrdn1681 ~]$ logout
Connection to lrdn1681.leonardo.local closed.
[ddibari0@login02 ~]$ exit
salloc: Relinquishing job allocation 3625780
[ddibari0@login02 ~]$ squeue --me
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
[ddibari0@login02 ~]$
```

SALLOC

Interactive batch jobs

Keep in mind that you are using computing nodes, meaning that you are consuming compute hours!

To exit from an interactive session, just type "*exit*" or press Ctrl+D (easy to forget with salloc)

Exercise 1

1) Write a job script with "walltime" of 3 minutes that asks for 1 node and 1 core in Booster partition. Copy-paste the following in the execution section:

```
hostname
echo 'Hello World'
sleep 4
```

Now add the automatic sending of the email in case of ending and abort of the job.

2) Launch the job with sbatch

3) Check its state with squeue

4) Check its state again with squeue after having increased the sleep to 60, namely:

```
hostname
echo 'Hello World'
sleep 60
```

Documentation

Our userguide goes into more depth about all the aspects described during this presentation. In particular, we suggest:

Production environment on LEONARDO

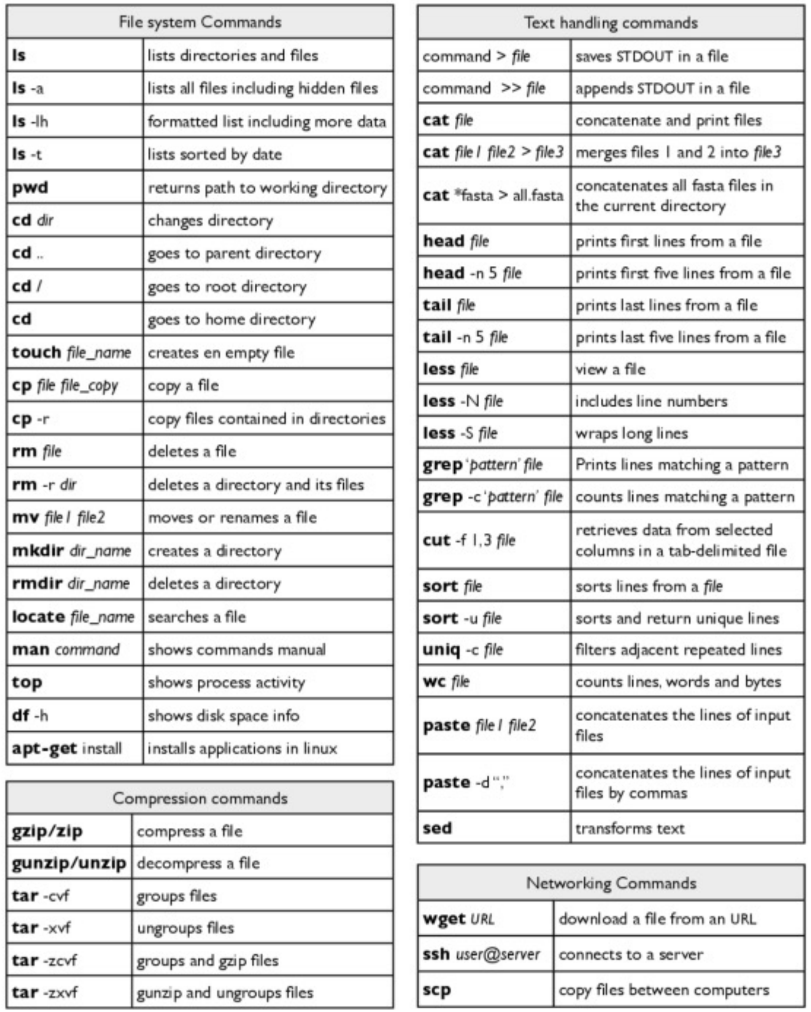
LEONARDO Booster partition specifics

Work areas and filesystem

Accounting and budget consumption

Batch scheduler Slurm

Linux commands



File system Commands	
ls	lists directories and files
ls -a	lists all files including hidden files
ls -lh	formatted list including more data
ls -t	lists sorted by date
pwd	returns path to working directory
cd *dir*	changes directory
cd ..	goes to parent directory
cd /	goes to root directory
cd	goes to home directory
touch *file_name*	creates en empty file
cp *file file_copy*	copy a file
cp -r	copy files contained in directories
rm *file*	deletes a file
rm -r *dir*	deletes a directory and its files
mv *file1 file2*	moves or renames a file
mkdir *dir_name*	creates a directory
rmdir *dir_name*	deletes a directory
locate *file_name*	searches a file
man *command*	shows commands manual
top	shows process activity
df -h	shows disk space info
apt-get install	installs applications in linux

Compression commands	
gzip/zip	compress a file
gunzip/unzip	decompress a file
tar -cvf	groups files
tar -xvf	ungroups files
tar -zcvf	groups and gzip files
tar -zxvf	gunzip and ungroups files

Text handling commands	
command > *file*	saves STDOUT in a file
command >> *file*	appends STDOUT in a file
cat *file*	concatenate and print files
cat *file1 file2* > *file3*	merges files 1 and 2 into *file3*
cat *fasta > all.fasta	concatenates all fasta files in the current directory
head *file*	prints first lines from a file
head -n 5 *file*	prints first five lines from a file
tail *file*	prints last lines from a file
tail -n 5 *file*	prints last five lines from a file
less *file*	view a file
less -N *file*	includes line numbers
less -S *file*	wraps long lines
grep '*pattern*' *file*	Prints lines matching a pattern
grep -c '*pattern*' *file*	counts lines matching a pattern
cut -f 1,3 *file*	retrieves data from selected columns in a tab-delimited file
sort *file*	sorts lines from a *file*
sort -u *file*	sorts and return unique lines
uniq -c *file*	filters adjacent repeated lines
wc *file*	counts lines, words and bytes
paste *file1 file2*	concatenates the lines of input files
paste -d ","	concatenates the lines of input files by commas
sed	transforms text

Networking Commands	
wget *URL*	download a file from an URL
ssh *user@server*	connects to a server
scp	copy files between computers