



UNIVERSITÄT
HEIDELBERG
ZUKUNFT
SEIT 1386



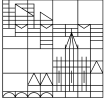
universität
uulm

universität freiburg



University of Stuttgart
Germany

Universität
Konstanz



UNIVERSITY OF
HOHENHEIM

ESSLINGEN
UNIVERSITY

Intermediate Filesystems

Begatim Bytyqi (KIT/SCC)

Motivation & Goals

■ In this lesson

- Understand the core mechanics of file systems, including key concepts like:
 - **File, file system, virtual file system, inodes, parallel file system**
- Compare different file systems by evaluating their pros & cons
- Select the most appropriate file system for your specific use case
- Exercises



Material: Slides & Scripts

- <https://indico.kit.edu/event/5742/>
- BwUniCluster 3.0: /opt/bwhpc/common/workshops/2026-10-21/
- HoreKa 2: /software/all/workshop/2026-10-21/

How to read the following slides

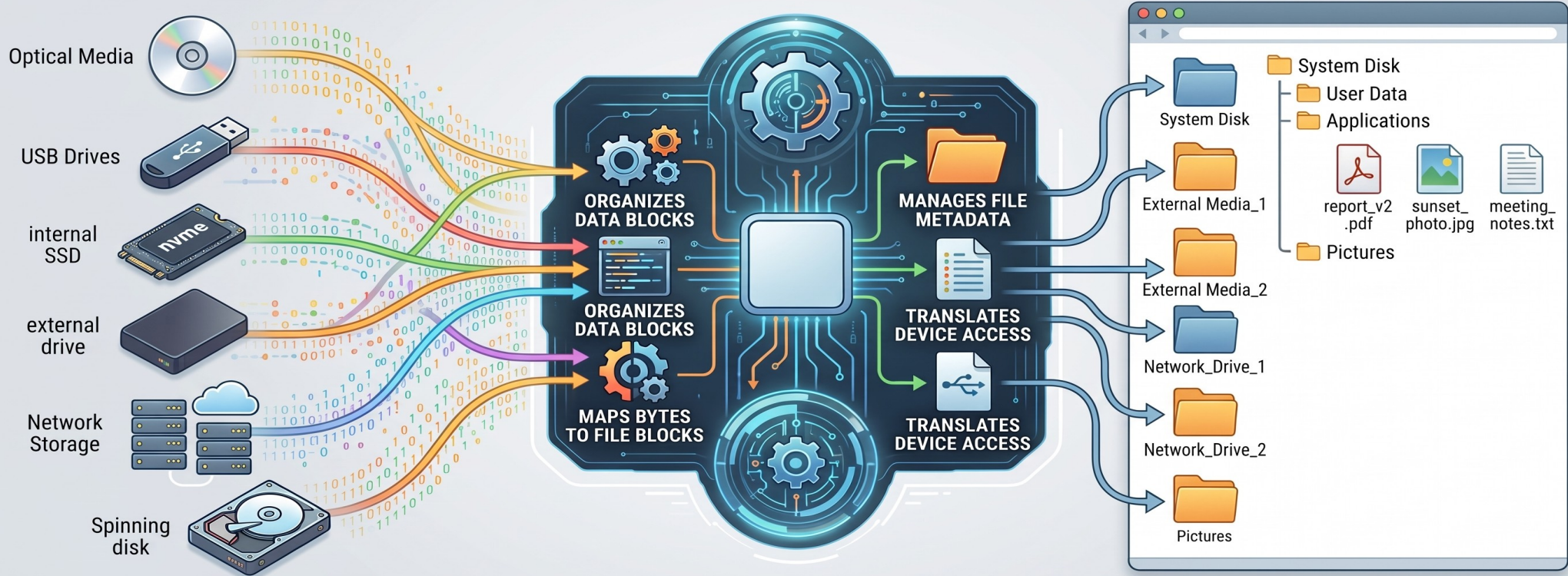
Abbreviation/Colour code	Full meaning
<code>\$ command -option value</code>	<code>\$</code> = prompt of the interactive shell The full prompt may look like: user@machine:path\$ The command has been entered in the interactive shell session
<code><integer></code> <code><string></code>	<code><></code> = Placeholder for integer, string etc
foo, bar	Metasyntactic variables

HOW A FILE SYSTEM ABSTRACTS HARDWARE COMPLEXITY

DIFFERENT HARDWARE STORAGE DEVICES (RAW DATA, OBLIVIOUS TO STRUCTURE)

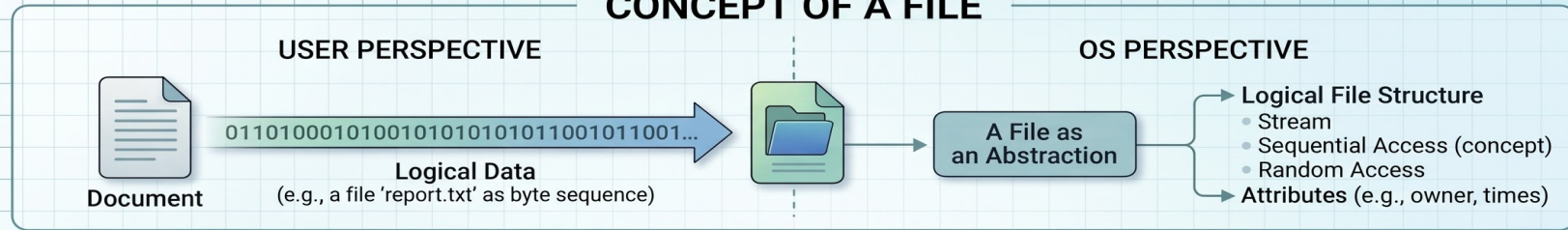
FILE SYSTEM ABSTRACTION LAYER (ORGANIZES AND MANAGES DATA)

UNIFIED USER VIEW (DIRECTORIES & FILES, COHERENT STRUCTURE)

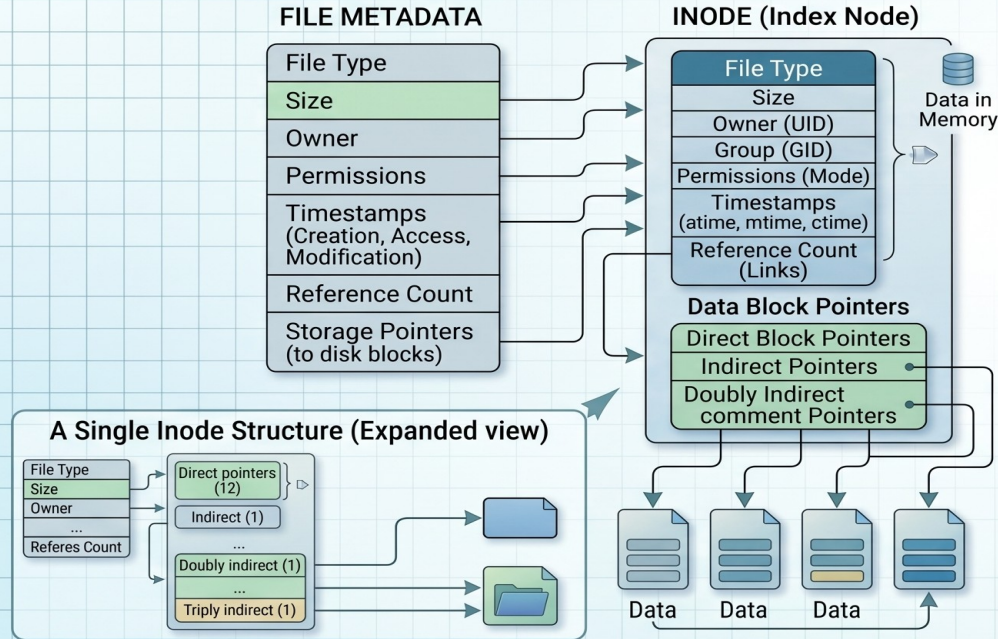


The File System simplifies and unifies diverse hardware into an organized, user-friendly view.

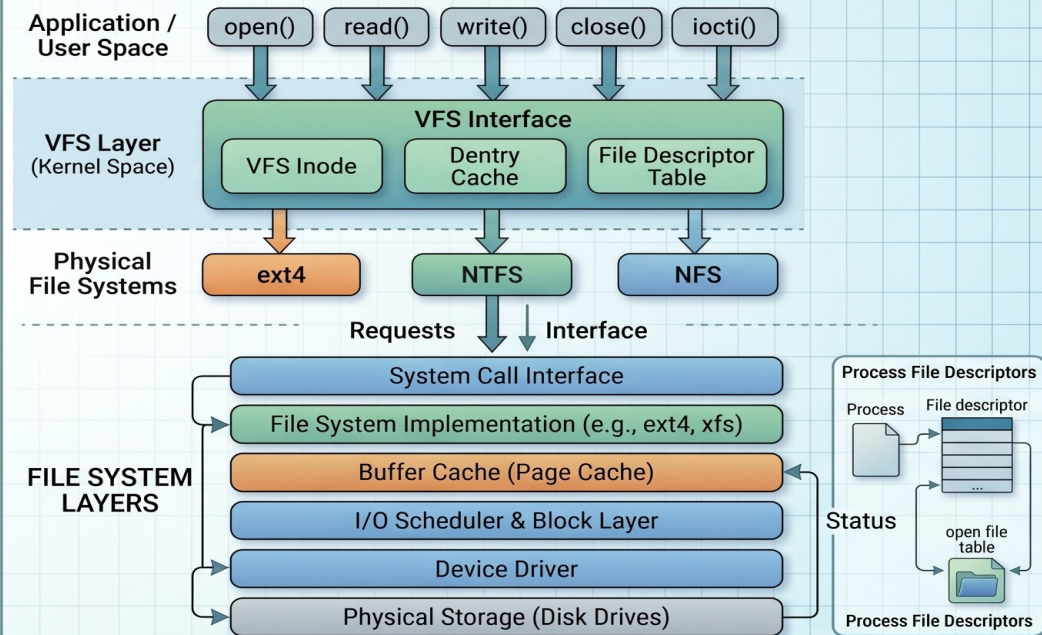
CONCEPT OF A FILE



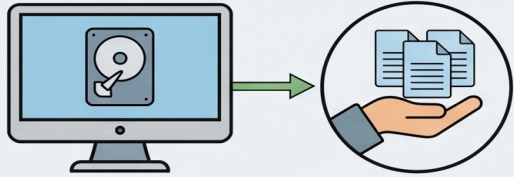
METADATA & INODES (INDEX NODES)



VFS (VIRTUAL FILE SYSTEM)

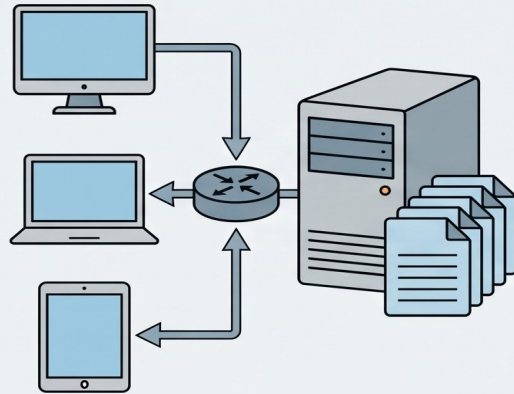


1. LOCAL FILE SYSTEM



Data stored locally, accessed by one computer only.

2. NETWORK FILE SYSTEM (NFS)

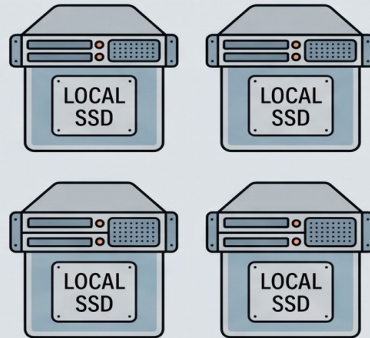


Data shared across a network, accessed by multiple computers.

THE DIFFERENCE: LOCAL NODE vs. PARALLEL FILE SYSTEM (In HPC)

LOCAL NODE FILE SYSTEM

Each node uses its own local SSD for high-speed but isolated storage.

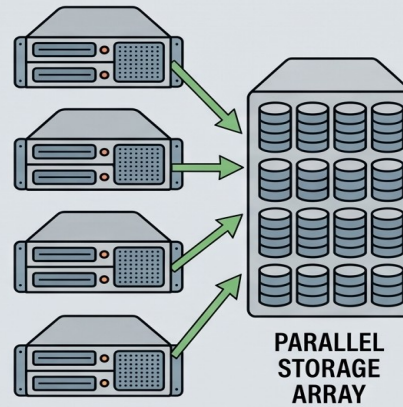


Each node uses its own local SSD for high-speed but isolated storage.

Limited capacity, difficult to share data between nodes.

PARALLEL FILE SYSTEM

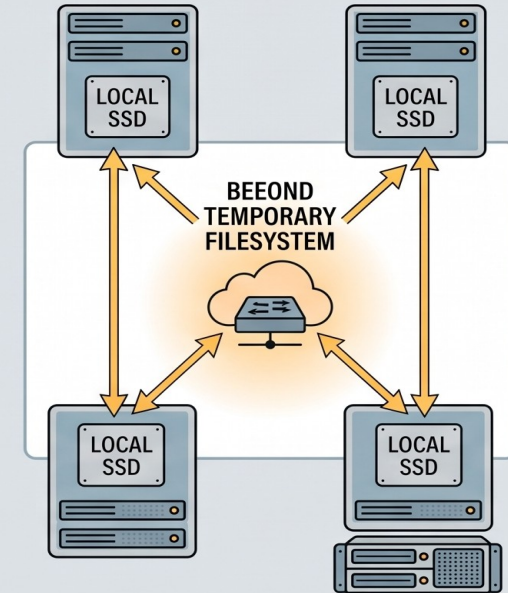
Single, shared namespace across all nodes, data striped across many storage targets for high performance.



High capacity, concurrent access, ideal for large datasets

SOMETHING IN-BETWEEN: BEEOND (BeeGFS On Demand)

A network filesystem but uses local SSDs for data storage.



Creates a performant, temporary parallel filesystem across local node resources for specific compute jobs.

MOST IMPORTANT PARALLEL FILE SYSTEMS

lustre™

Used on most of the largest HPC systems



**IBM Storage Scale
(aka GPFS)**

Used in industry and on many HPC systems



BeeGFS
with GPFS

Underlying file system for BeeGFS On Demand (BeeOND)

COMMON CHARACTERISTICS & CAPABILITIES



POSIX Compliant

All three file systems follow the POSIX standard.



Large Capacity & Parallel Access

Offers large capacity and parallel access from many nodes.



High Performance for Large Data

Good performance for huge files and access with large chunks.

LIMITATIONS & UNDERLYING REASONS



Limitations

Dislike small files, random I/O, or many metadata (open, close, stat, create, remove) operations



Reasons

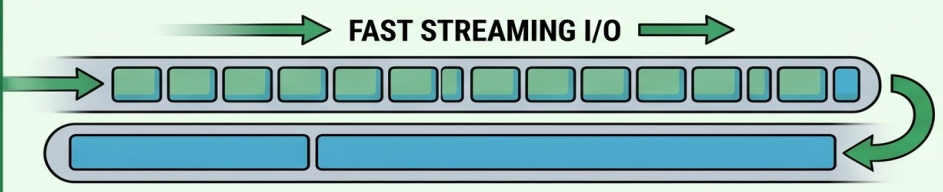


Communication over network and locking to guarantee consistency

PARALLEL FILE SYSTEM CONSIDERATIONS

 **1. SEQUENTIAL DATA IS IDEAL**  **OPTIMIZED PERFORMANCE**

FAST STREAMING I/O



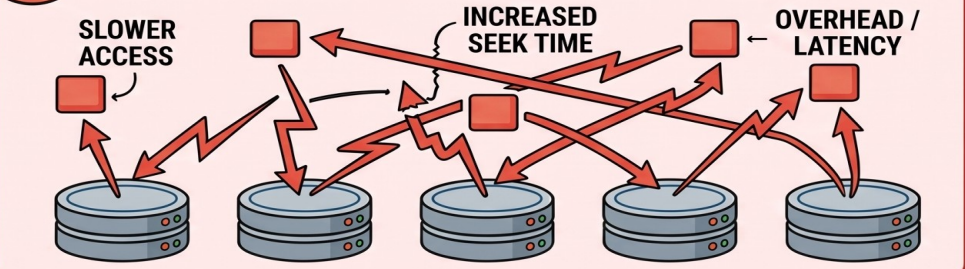
EX. SINGLE LARGE FILE (e.g., 1 GB)

 **2. WHAT IS RANDOM I/O?**  **POOR PERFORMANCE**

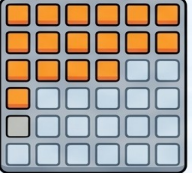
SLOWER ACCESS

INCREASED SEEK TIME

OVERHEAD / LATENCY



 **3. SMALL FILES vs. LARGE FILES (GPFS 16MB BLOCK)**

MANY SMALL FILES (Tiny 1KB, 10KB...)	FEW LARGE FILES (e.g., 20MB, 100MB+)
	
FREE FOR SUB-BLOCKS/ FRAGMENTS	MINIMAL OVERHEAD
HIGH METADATA OVERHEAD & FRAGMENTATION	HIGH BLOCK UTILIZATION / EFFICIENT

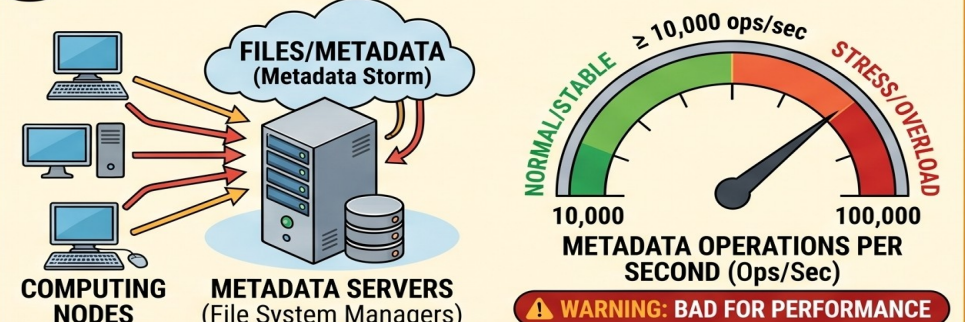
 **RECOMMENDED FILE SIZE: LARGE FILES FOR BEST I/O PATTERNS**

 **4. HOW MANY FILES? (METADATA STRESS)**

FILES/METADATA
(Metadata Storm)

COMPUTING
NODES

METADATA SERVERS
(File System Managers)



≥ 10,000 ops/sec

10,000 100,000

METADATA OPERATIONS PER
SECOND (Ops/Sec)

 **WARNING: BAD FOR PERFORMANCE**

Filesystem selection guide

Property	\$HOME	Workspace	\$TMPDIR	BeeOND ¹	LSDF ¹
ACCESS & RETENTION					
Visibility	All nodes	All nodes	Local node	Job nodes	Login + job nodes
Lifetime	Permanent	Weeks	Job runtime	Job runtime	Permanent
Backup	Yes*	No	No	No	Yes
SCALE					
Capacity	40 GB–10 TB	10–250 TB	128 GB–7 TB per node	N × 750 GB	Per project
Inodes	2M–unlimited	1M–unlimited	Unlimited	Unlimited	Per project
I/O CHARACTERISTICS					
Sequential I/O	Medium	High	Node-local	High (scales)	High
Random I/O	Low	Low	High	Medium	Low
Shared-system impact	High	Medium	Low	Low	High

* \$HOME backup excludes Helix. ¹ BeeOND and LSDF are only available on bwUniCluster 3.0 and HoreKa.

I/O ratings are qualitative.

File system on each node using local disks (\$TMPDIR)

■ Node local storage on SSDs

- Usage with environment variable \$TMPDIR
 - On JUSTUS 2 \$TMPDIR file system stores data in main memory and \$SCRATCH is on local SSD
- **Separate** temporary private directory **on each node** of a batch job
 - Make sure you have copied your data back to a workspace or \$HOME within your job
- HowTo: → [https://wiki.bwhpc.de/e/BwUniCluster3.0/Hardware_and_Architecture#\\$TMPDIR](https://wiki.bwhpc.de/e/BwUniCluster3.0/Hardware_and_Architecture#$TMPDIR)

■ Usage example

- Outside batch job create archive with compressed input dataset on a workspace:

```
$ tar -cvzf $(ws_find data-ssd)/dataset.tgz dataset/
```
- In batch script extract compressed input dataset to local SSD:

```
tar -C $TMPDIR/ -xvzf $(ws_find data-ssd)/dataset.tgz
```
- In batch script application reads data from dataset on SSD and writes results to SSD:

```
myapp -input $TMPDIR/dataset/myinput.csv -outputdir $TMPDIR/results
```
- In batch script save results to a workspace:

```
rsync -av $TMPDIR/results $(ws_find data-ssd)/results-${SLURM_JOB_ID}/
```

BeeOND = Private file system for batch job

■ BeeOND (BeeGFS on Demand)

- Available only on **bwUniCluster 3.0** and on **HoreKa 2**
- Private file system for batch job, created at job start and destroyed at job end
 - Data staging is **required**
- Request creation in job script or on command line:
- Exclusive allocation is **required**.

■ Request creation on job script or using the command line

```
#SBATCH --constraint=BEEOND  
#SBATCH --exclusive
```

```
$ sbatch --exclusive -C BEEOND ...
```

■ Use path below `/mnt/odfs/${SLURM_JOB_ID}` to access BeeOND, e.g.

```
$ cd /mnt/odfs/${SLURM_JOB_ID}/stripe_default
```

→ [https://wiki.bwhpc.de/e/BwUniCluster3.0/Hardware_and_Architecture#BeeOND_\(BeeGFS_On-Demand\)](https://wiki.bwhpc.de/e/BwUniCluster3.0/Hardware_and_Architecture#BeeOND_(BeeGFS_On-Demand))

LSDF Online Storage = External storage for special users

■ LSDF Online Storage

- Available only on bwUniCluster 3.0 and on HoreKa 2 **for special users**
 - Intended usage for scientific measurement data and data-intensive scientific simulation results
 - <https://www.scc.kit.edu/en/services/11228.php>
- **Visible** on login nodes and on batch job nodes **if access was requested**
 - External access from external with different protocols is also possible
- Request access in job script or on command line:

```
#SBATCH --constraint=LSDF
```

```
$ sbatch -C LSDF ...
```
- Use environment variables \$LSDF, \$LSDFPROJECTS, \$LSDFHOME to access, e.g.

```
cd ${LSDF}
```
- HowTo: → https://wiki.bwhpc.de/e/BwUniCluster3.0/Hardware_and_Architecture#LSDF_Online_Storage

Remarks for exercise

- Login to bwUnicluster 3.0 or HoreKa 2 and show list of commands for exercises:

- BwUniCluster 3.0:

```
$ cat /opt/bwhpc/common/workshops/2026-10-21/02/pfs_commands.txt
```

- HoreKa 2:

```
$ cat /software/all/workshop/2026-10-21/02/pfs_commands.txt
```

- Use Cut & Paste to execute the commands

- Start by creating the workspaces

