Focus on computing techniques

sara.vallero@to.infn.it

Smart workflows

Case studies

What about distributed computing?

Infrastructure explained

What about distributed computing?

I have a difficult problem that I cannot compute on my laptop…

- Maybe the problem is too complex
- Or maybe there is too much data to be processed

Strong scaling

Weak scaling

High performance or high throughput?

How to distribute data?

How to distribute the calculation?

How to distribute data?

Distributed filesystem

A Distributed Filesystem (DFS) is a physically distributed implementation of the classical time-sharing model of the traditional filesystem. It resides on different machines and/or sites and offers a **unified logical view of data**, whether local or remote.

- **TRANSPARENCY:**
  - Be able to perform same operations as on a local filesystem
  - Files must appear as they were put on a single location
  - The complexity of the underlying system must be hidden to users

- **FAULT TOLERANCE:**
  - System availability despite network or server failures
  - Data integrity and consistency when several users try to access them simultaneously

- **SCALABILITY:**
  - Efficiently leverage the dynamic addition of new servers
    - Centralized systems = single server = bottleneck (unless multithreading and/or caching)

Some common DFSs

	HDFS	Ceph	GlusterFS	Lustre
Architecture	Centralized	Distributed	Decentralized (client-server)	Centralized
Interfaces				
System availability				
Data availability				
Load balancing				
Storage type				

Architecture

- **Client-server:** several servers store and share metadata and data between multiple clients (global namespace)
- **Cluster-based:** data and metadata are decoupled
    - **Centralized** = only 1 metadata server
    - **Totally distributed** = distributed metadata servers

Some common DFSs

	HDFS	Ceph	GlusterFS	Lustre
Architecture	Centralized	Distributed	Decentralized (client-server)	Centralized
Interfaces	CLI, FUSE, REST, API	FUSE, mount, REST	FUSE, mount	FUSE
System availability				
Data availability				
Load balancing				
Storage type				

Interfaces

- **Command Line Interface (CLI):** access files with traditional unix command (cp, rm, mv…)
- **Application Programming Interface (API):** implemented in different programming languages or REST (web-based)
- **Mount:** attach remote directories to the local filesystem (using unix mount or FUSE)

Some common DFSs

hadoop ceph GlusterFS lustre

	HDFS	Ceph	GlusterFS	Lustre
Architecture	Centralized	Distributed	Decentralized (client-server)	Centralized
Interfaces	CLI, FUSE, REST, API	FUSE, mount, REST	FUSE, mount	FUSE
System availability	No failover	High	High	Failover
Data availability				
Load balancing				
Storage type				

System availability

- **High availability:** metadata are replicated and distributed across several servers
- **Failover:** several servers in standby periodically save the metadata to be ready to take control
- **No failover:** this is a Single Point of Failure (SPOF). In HDFS a second server periodically saves the metadata, but a system stop is required for it to take control.

Some common DFSs

	HDFS	Ceph	GlusterFS	Lustre
Architecture	Centralized	Distributed	Decentralized (client-server)	Centralized
Interfaces	CLI, FUSE, REST, API	FUSE, mount, REST	FUSE, mount	FUSE
System availability	No failover	High	High	Failover
Data availability	Replication	Replication	RAID-like	No
Load balancing				
Storage type				

Data availability

- **Replication:** several copies of the data are made. This might raise consistency issues solved by synchronization mechanisms and placement strategies.
- **RAID-like:** several copies of a whole storage device into other ones inside the same volume.
- **No replication:** rely on independent software

Some common DFSs

	HDFS	Ceph	GlusterFS	Lustre
Architecture	Centralized	Distributed	Decentralized (client-server)	Centralized
Interfaces	CLI, FUSE, REST, API	FUSE, mount, REST	FUSE, mount	FUSE
System availability	No failover	High	High	Failover
Data availability	Replication	Replication	RAID-like	No
Load balancing	Auto	Manual	Manual	No
Storage type				

Load balancing

- Overloaded servers can delay or abort request execution. These must be relieved and the data distributed on other servers in the cluster (possibly newly added empty ones).

Some common DFSs

	HDFS	Ceph	GlusterFS	Lustre
Architecture	Centralized	Distributed	Decentralized (client-server)	Centralized
Interfaces	CLI, FUSE, REST, API	FUSE, mount, REST	FUSE, mount	FUSE
System availability	No failover	High	High	Failover
Data availability	Replication	Replication	RAID-like	No
Load balancing	Auto	Manual	Manual	No
Storage type	File	Object	File	File (parallel)

Storage type

- **File store:** data is organized in directory trees, folders, and individual files (hierarchical). It functions similarly to a local hard drive.
- **Object store:** breaks files into pieces called objects. Each object receives a unique ID and also stores metadata about the file. Not hierarchical, scalable.
- **Block store:** breaks up data into blocks and then stores them as separate pieces, each with a unique ID.

Which one is best?

Small quantity of big files: HDFS, Lustre

Both small and big data: Ceph, GlusterFS

High throughput: Lustre

A closer look to HDFS

- for managing pools of big data and supporting related big data analytics applications

- rapid transfer of data between compute nodes

- closely coupled with MapReduce

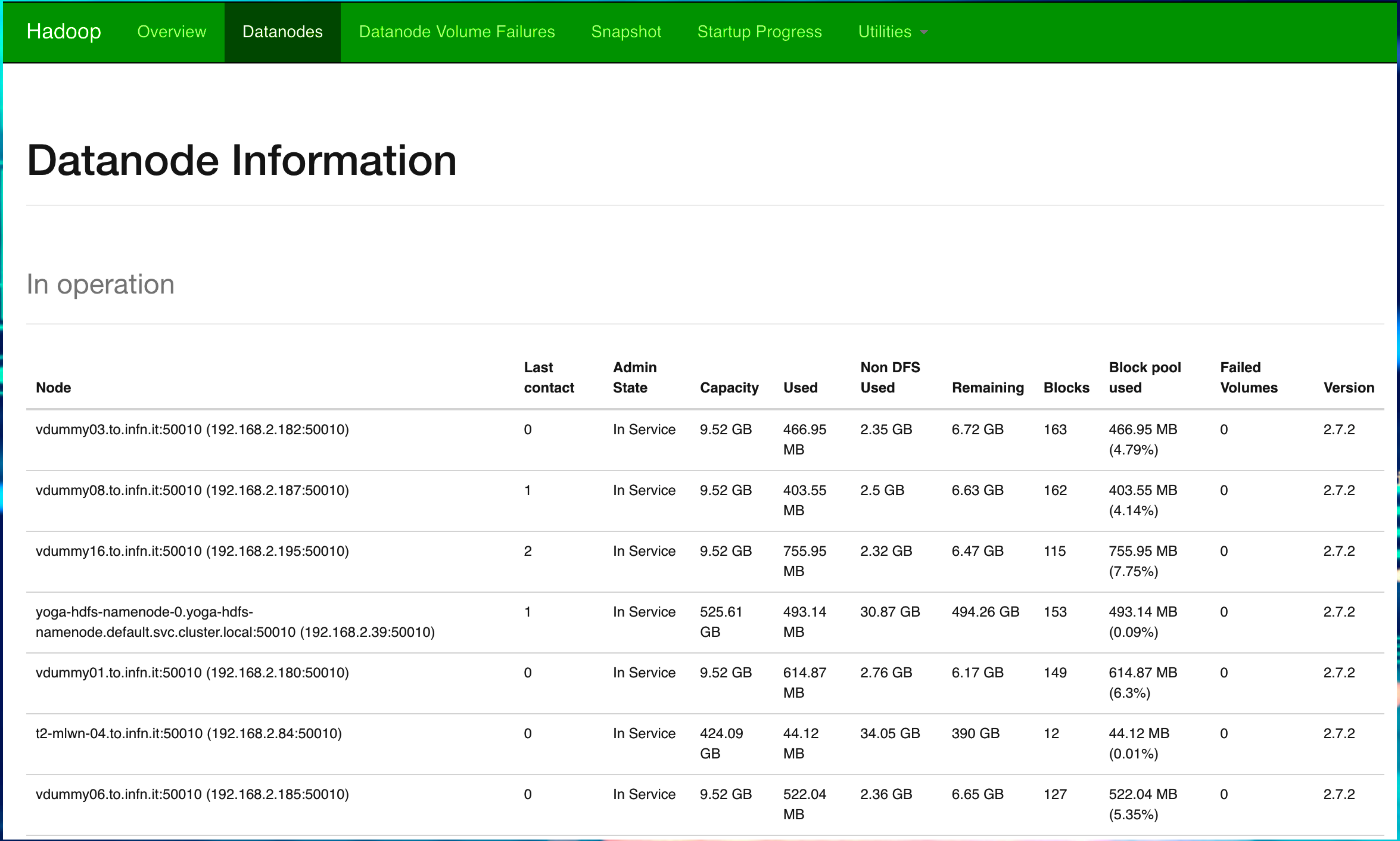
- breaks the information down into separate blocks and distributes them to different nodes in a cluster: highly efficient parallel processing

- highly fault-tolerant (data-wise). The file system replicates each piece of data multiple times and distributes the copies to individual nodes. In case of a node crash, processing can continue while data is recovered

- echoes POSIX design style in some aspects

- very large-scale implementations

- support for low-cost commodity hardware



Datanode Information

In operation

Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
vdummy03.to.infn.it:50010 (192.168.2.182:50010)	0	In Service	9.52 GB	466.95 MB	2.35 GB	6.72 GB	163	466.95 MB (4.79%)	0	2.7.2
vdummy08.to.infn.it:50010 (192.168.2.187:50010)	1	In Service	9.52 GB	403.55 MB	2.5 GB	6.63 GB	162	403.55 MB (4.14%)	0	2.7.2
vdummy16.to.infn.it:50010 (192.168.2.195:50010)	2	In Service	9.52 GB	755.95 MB	2.32 GB	6.47 GB	115	755.95 MB (7.75%)	0	2.7.2
yoga-hdfs-namenode-0.yoga-hdfs-namenode.default.svc.cluster.local:50010 (192.168.2.39:50010)	1	In Service	525.61 GB	493.14 MB	30.87 GB	494.26 GB	153	493.14 MB (0.09%)	0	2.7.2
vdummy01.to.infn.it:50010 (192.168.2.180:50010)	0	In Service	9.52 GB	614.87 MB	2.76 GB	6.17 GB	149	614.87 MB (6.3%)	0	2.7.2
t2-mlwn-04.to.infn.it:50010 (192.168.2.84:50010)	0	In Service	424.09 GB	44.12 MB	34.05 GB	390 GB	12	44.12 MB (0.01%)	0	2.7.2
vdummy06.to.infn.it:50010 (192.168.2.185:50010)	0	In Service	9.52 GB	522.04 MB	2.36 GB	6.65 GB	127	522.04 MB (5.35%)	0	2.7.2

Hadoop
Data
In ope
Hadoop
Overview
Datanodes
Snapshot
Startup Progress
Utilities
Go!

Node
vdummy03
vdummy08
vdummy16
yoga-hdfs-r namenode.
vdummy01
t2-mlwn-04
vdummy06

Browse Directory

/user

Hierarchical directory structure

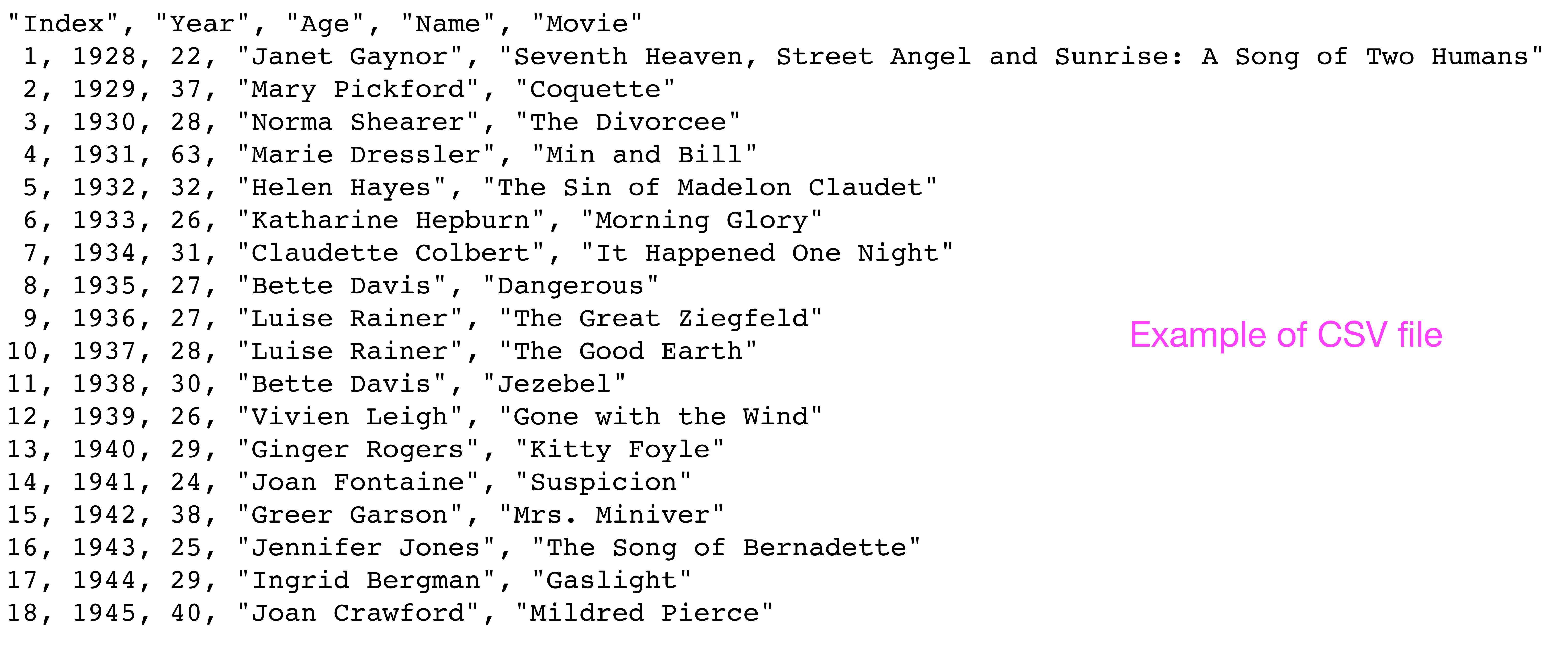
Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
drwxr-xr-x	jovyan	jovyan	0 B	11/15/2019, 12:12:08 PM	0	0 B	aliber92
drwxr-xr-x	jovyan	jovyan	0 B	11/14/2019, 6:48:25 PM	0	0 B	bagnasco
drwxr-xr-x	jovyan	jovyan	0 B	11/14/2019, 6:40:00 PM	0	0 B	gabrielefronze
drwxr-xr-x	jovyan	jovyan	0 B	11/14/2019, 6:53:12 PM	0	0 B	giorgiobar
drwxr-xr-x	jovyan	jovyan	0 B	11/14/2019, 6:36:21 PM	0	0 B	leggerf
drwxr-xr-x	jovyan	jovyan	0 B	11/15/2019, 12:13:42 PM	0	0 B	marco-ph
drwxr-xr-x	jovyan	jovyan	0 B	11/15/2019, 12:15:11 PM	0	0 B	obertino
drwxr-xr-x	jovyan	jovyan	0 B	11/14/2019, 6:49:07 PM	0	0 B	slusso
drwxr-xr-x	jovyan	jovyan	0 B	9/30/2019, 12:53:34 PM	0	0 B	svallero
drwxr-xr-x	jovyan	jovyan	0 B	10/2/2019, 11:37:29 AM	0	0 B	testuser
drwxr-xr-x	jovyan	jovyan	0 B	10/7/2019, 7:16:50 PM	0	0 B	testuser2

Some useful file format

Use tabular data structure with headers

CSV

- Comma separated values
- Data is organized in rows
- Human readable (text file)
- Used i.e. to export data from Microsoft Excel



```
"Index", "Year", "Age", "Name", "Movie"
 1, 1928, 22, "Janet Gaynor", "Seventh Heaven, Street Angel and Sunrise: A Song of Two Humans"
 2, 1929, 37, "Mary Pickford", "Coquette"
 3, 1930, 28, "Norma Shearer", "The Divorcee"
 4, 1931, 63, "Marie Dressler", "Min and Bill"
 5, 1932, 32, "Helen Hayes", "The Sin of Madelon Claudet"
 6, 1933, 26, "Katharine Hepburn", "Morning Glory"
 7, 1934, 31, "Claudette Colbert", "It Happened One Night"
 8, 1935, 27, "Bette Davis", "Dangerous"
 9, 1936, 27, "Luise Rainer", "The Great Ziegfeld"
10, 1937, 28, "Luise Rainer", "The Good Earth"
11, 1938, 30, "Bette Davis", "Jezebel"
12, 1939, 26, "Vivien Leigh", "Gone with the Wind"
13, 1940, 29, "Ginger Rogers", "Kitty Foyle"
14, 1941, 24, "Joan Fontaine", "Suspicion"
15, 1942, 38, "Greer Garson", "Mrs. Miniver"
16, 1943, 25, "Jennifer Jones", "The Song of Bernadette"
17, 1944, 29, "Ingrid Bergman", "Gaslight"
18, 1945, 40, "Joan Crawford", "Mildred Pierce"
```

Example of CSV file

Some useful file format

Use tabular data structure with headers

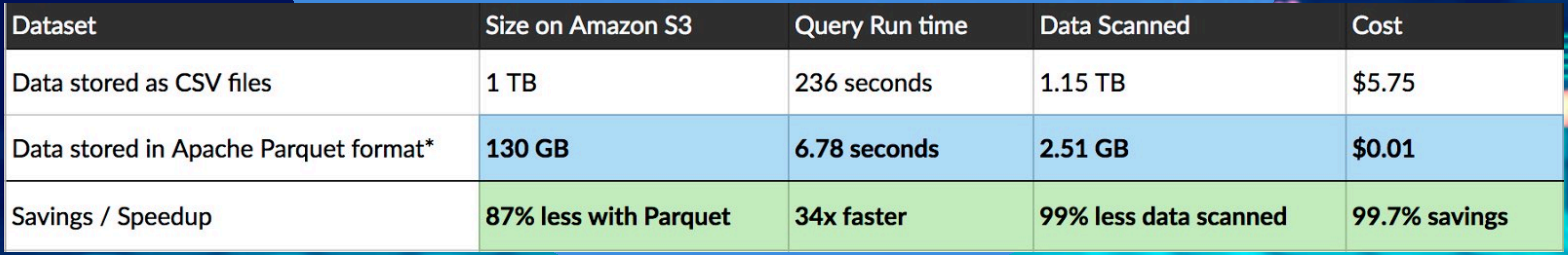
CSV

- Comma separated values
- Data is organized in rows
- Human readable (text file)
- Used i.e. to export data from Microsoft Excel

Apache Parquet

- Columnar storage format
- Self describing (schema is embedded within the data)
- Supports complex nested data structures
- Optimized for query performance
- Minimized I/O
- Compression and encoding (Snappy)

Commercial Clouds charge you by the amount of data stored or scanned per query



Dataset	Size on Amazon S3	Query Run time	Data Scanned	Cost
Data stored as CSV files	1 TB	236 seconds	1.15 TB	$5.75
Data stored in Apache Parquet format*	**130 GB**	**6.78 seconds**	**2.51 GB**	**$0.01**
Savings / Speedup	**87% less with Parquet**	**34x faster**	**99% less data scanned**	**99.7% savings**

Hadoop
Overview
Datanodes
Snapshot
Startup Progress
Utilities
Go!
×
Download
Close

Browse Directory

/data

Permission	Owner	Group	ck Size	Name
drwxr-xr-x	root	supergro		Higgs100k.parquet
drwxr-xr-x	root	supergro		Higgs10M.parquet
-rwxr-xr-x	root	supergro	MB	Higgs1M.csv
drwxr-xr-x	root	supergro		Higgs1M.parquet

Hadoop, 2015.

File information - Higgs1M.csv

Block information -- ✓ Block 0 / Block 1 / Block 2 / Block 3 / Block 4 / Block 5

Block ID: 10737442

Block Pool ID: BP-1561664661-192.135.19.9-1567437632293

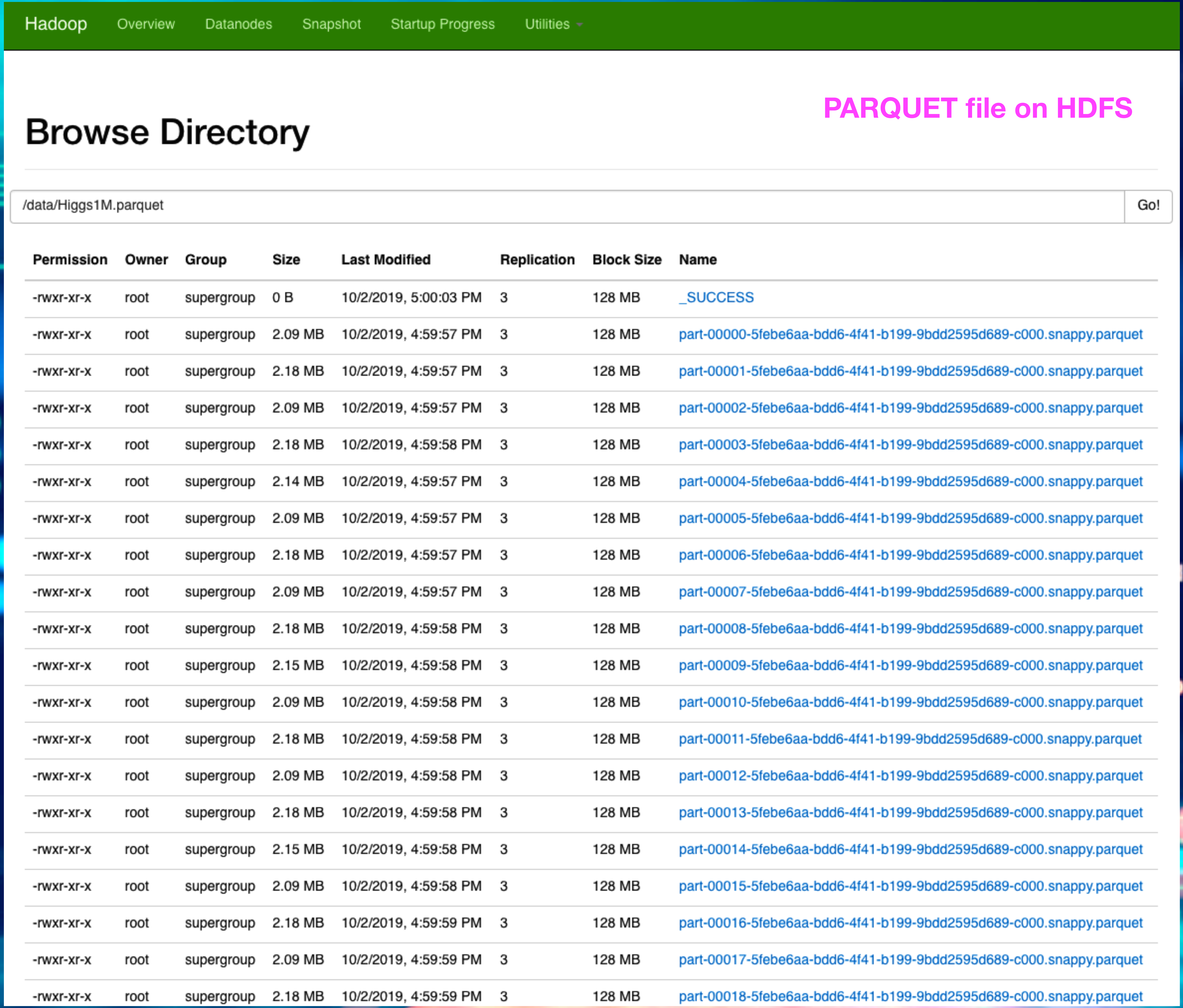
Generation Stamp: 3438

Size: 134217728

Availability:

- vdummy15.to.infn.it
- vdummy16.to.infn.it
- vdummy02.to.infn.it

CSV file on HDFS



PARQUET file on HDFS

Browse Directory

/data/Higgs1M.parquet

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rwxr-xr-x	root	supergroup	0 B	10/2/2019, 5:00:03 PM	3	128 MB	_SUCCESS
-rwxr-xr-x	root	supergroup	2.09 MB	10/2/2019, 4:59:57 PM	3	128 MB	part-00000-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.18 MB	10/2/2019, 4:59:57 PM	3	128 MB	part-00001-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.09 MB	10/2/2019, 4:59:57 PM	3	128 MB	part-00002-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.18 MB	10/2/2019, 4:59:58 PM	3	128 MB	part-00003-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.14 MB	10/2/2019, 4:59:57 PM	3	128 MB	part-00004-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.09 MB	10/2/2019, 4:59:57 PM	3	128 MB	part-00005-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.18 MB	10/2/2019, 4:59:57 PM	3	128 MB	part-00006-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.09 MB	10/2/2019, 4:59:57 PM	3	128 MB	part-00007-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.18 MB	10/2/2019, 4:59:58 PM	3	128 MB	part-00008-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.15 MB	10/2/2019, 4:59:58 PM	3	128 MB	part-00009-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.09 MB	10/2/2019, 4:59:58 PM	3	128 MB	part-00010-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.18 MB	10/2/2019, 4:59:58 PM	3	128 MB	part-00011-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.09 MB	10/2/2019, 4:59:58 PM	3	128 MB	part-00012-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.18 MB	10/2/2019, 4:59:58 PM	3	128 MB	part-00013-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.15 MB	10/2/2019, 4:59:58 PM	3	128 MB	part-00014-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.09 MB	10/2/2019, 4:59:58 PM	3	128 MB	part-00015-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.18 MB	10/2/2019, 4:59:59 PM	3	128 MB	part-00016-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.09 MB	10/2/2019, 4:59:59 PM	3	128 MB	part-00017-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet
-rwxr-xr-x	root	supergroup	2.18 MB	10/2/2019, 4:59:59 PM	3	128 MB	part-00018-5febe6aa-bdd6-4f41-b199-9bdd2595d689-c000.snappy.parquet

Lessons to take-home

- We know what a distributed filesystem is, why it's useful…
- …and we can even name a few
- File format matters