Data Storage in High Energy Physics with ROOT

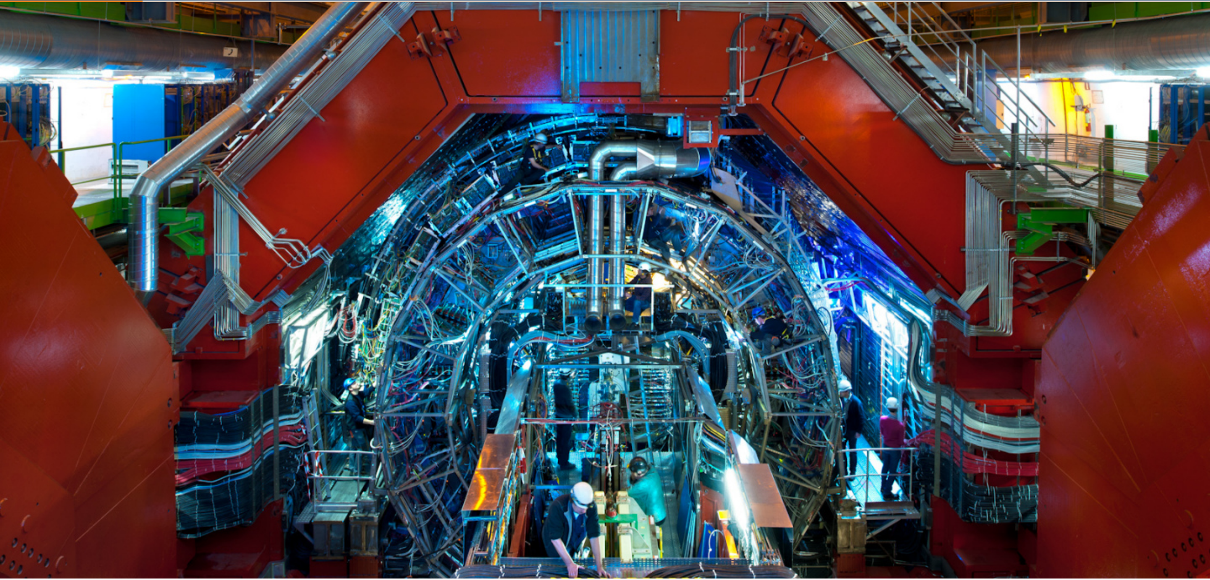
Jakob Blomer, CERN

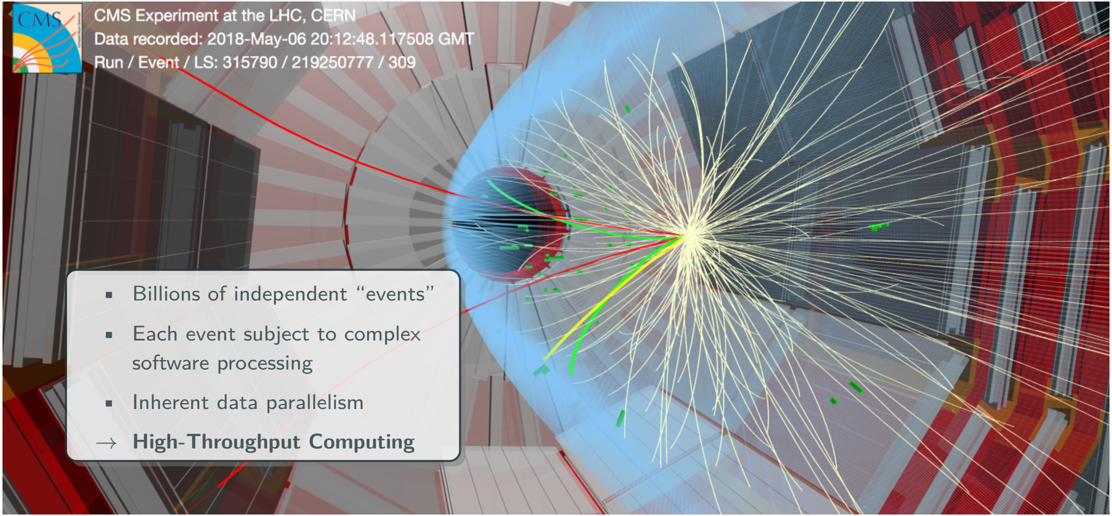
European HDF5 User Group Meeting, ITER, June 2022

Accelerate & Collide

Particle collisions at high energies give access to the physics of the smallest scales

Inside a Particle Detector

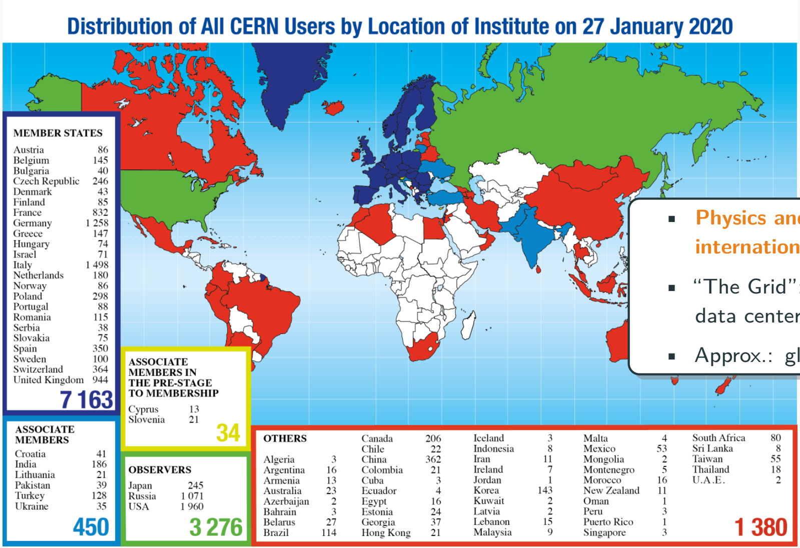




- Billions of independent “events”
- Each event subject to complex software processing
- Inherent data parallelism

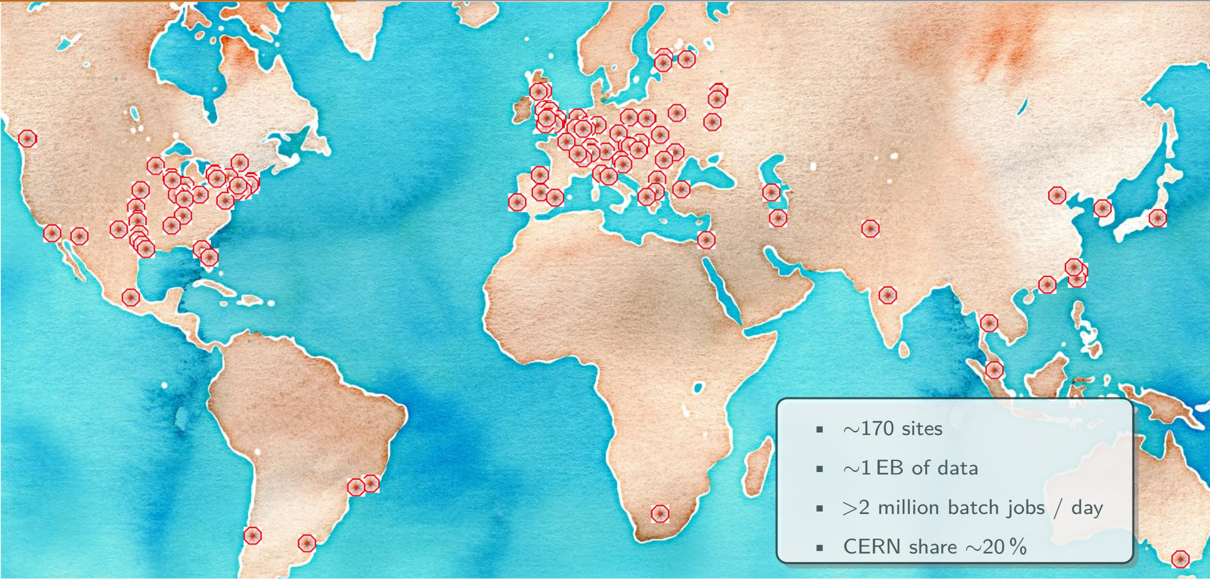
→ **High-Throughput Computing**

Federated Computing Resources



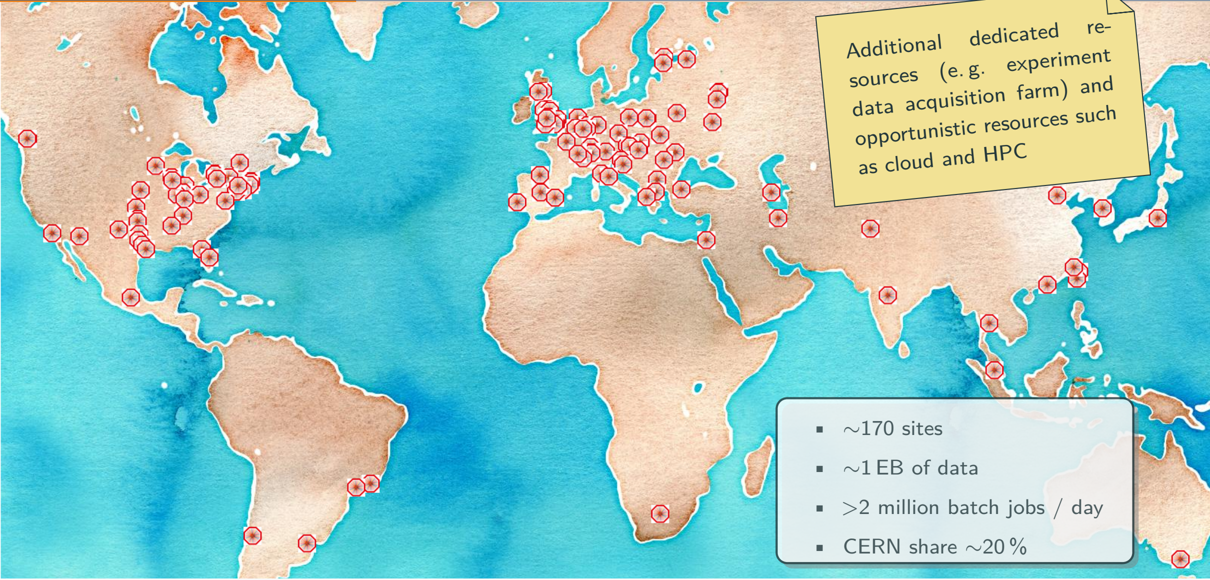
- **Physics and computing: international collaborations**
- “The Grid”: federating academic data centers
- Approx.: global batch system

Scale of the Worldwide LHC Computing Grid



- ∼170 sites
- ∼1 EB of data
- >2 million batch jobs / day
- CERN share ∼20 %

Scale of the Worldwide LHC Computing Grid



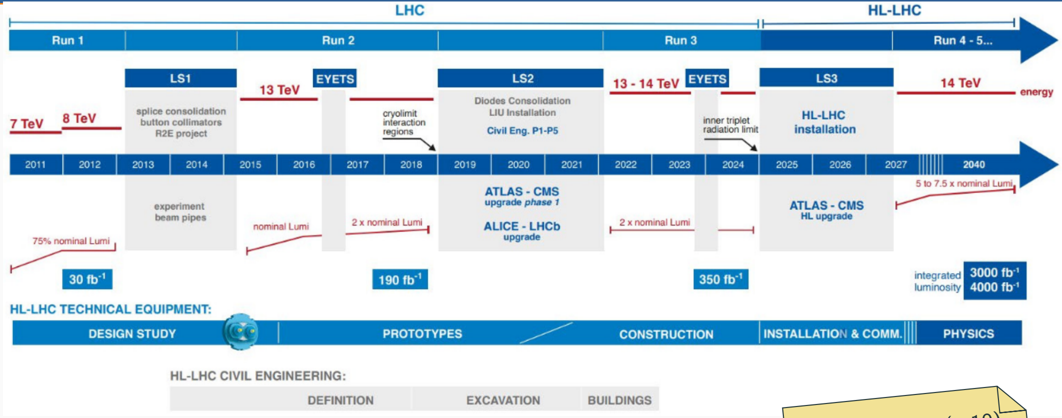
Additional dedicated resources (e. g. experiment data acquisition farm) and opportunistic resources such as cloud and HPC

- ∼170 sites
- ∼1 EB of data
- >2 million batch jobs / day
- CERN share ∼20 %

Experimental conditions from LHC to HL-LHC:

Luminosity:	$2 \times 10^{34} \mathrm{s}^{-1}\mathrm{cm}^{-2}$	$\rightarrow$	$5\text{–}7.5 \times 10^{34} \mathrm{s}^{-1}\mathrm{cm}^{-2}$
Radiation background:	$10^{14} \mathrm{neq/cm}^2$	$\rightarrow$	$\times 10^{16} \mathrm{neq/cm}^2$
Pile-up events	$\sim$40	$\rightarrow$	$\sim$140–200

Towards High-Luminosity LHC (HL-LHC)

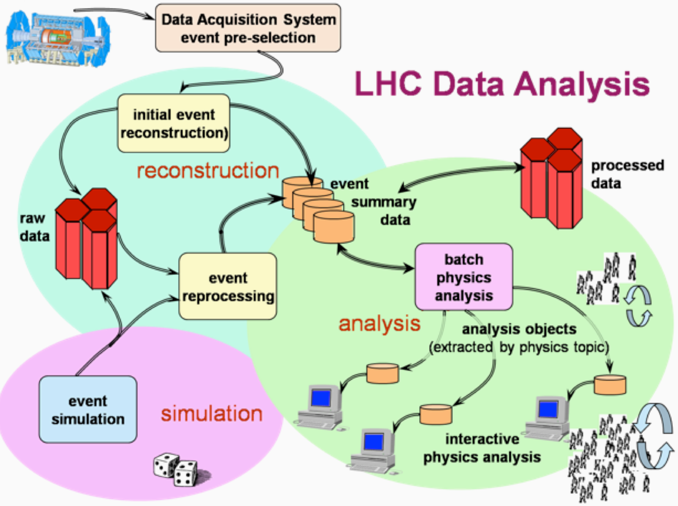


Experimental conditions from LHC to HL-LHC:

Luminosity:	$2 \times 10^{34}\mathrm{s}^{-1}\mathrm{cm}^{-2}$	$\rightarrow$	$5\text{–}7.5\times 10^{34}\mathrm{s}^{-1}\mathrm{cm}^{-2}$
Radiation background:	$10^{14}\mathrm{neq/cm}^2$	$\rightarrow$	$\times 10^{16}\mathrm{neq/cm}^2$
Pile-up events	$\sim$40	$\rightarrow$	$\sim$140–200

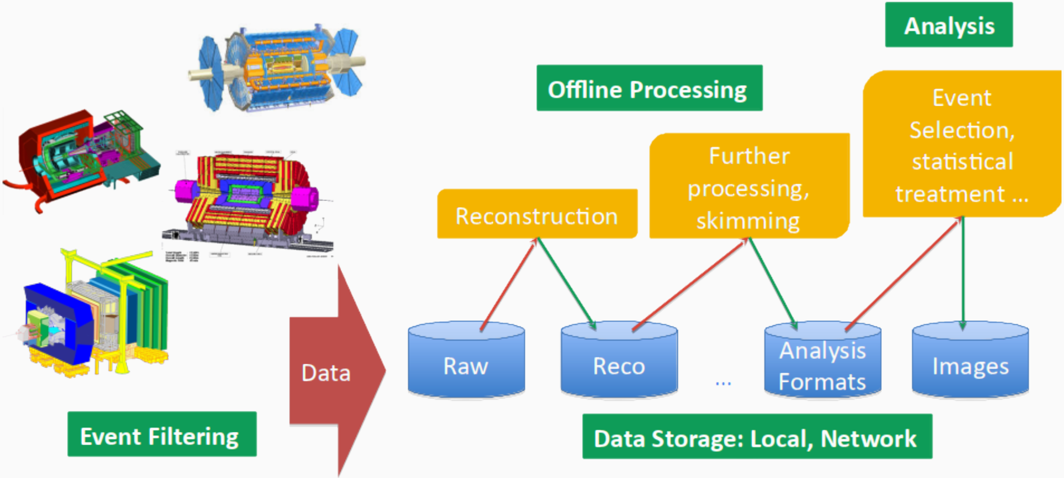
Until 2030: $\mathcal{O}(\times 10)$ bigger computing problem in many relevant metrics

Data Streams and Workflows

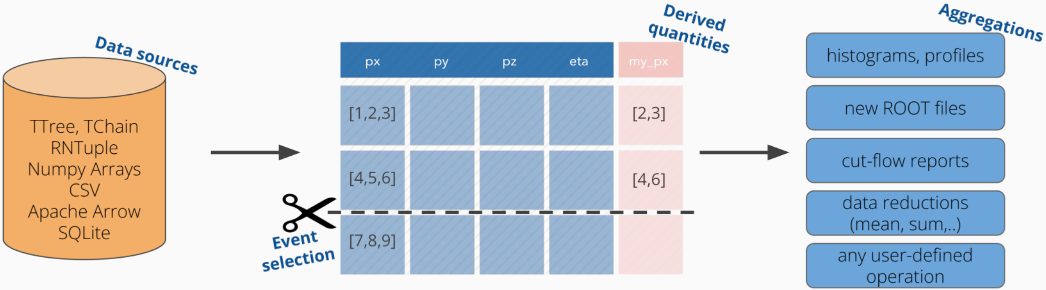


Source: Harvey et al.

Filtering, Skimming, Slimming, Derived Data



Data Analysis in High Energy Physics in a Nutshell

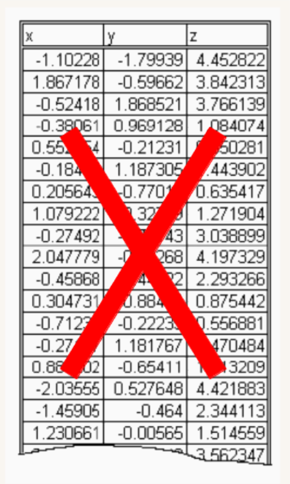


Some aspects particular to HEP

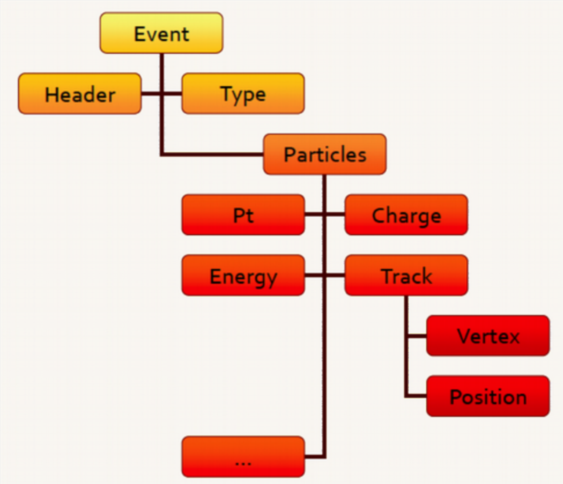
Input datasets are much larger than memory, entries are statistically independent.

Histograms, new ROOT files as common aggregations.

Event Data Structure

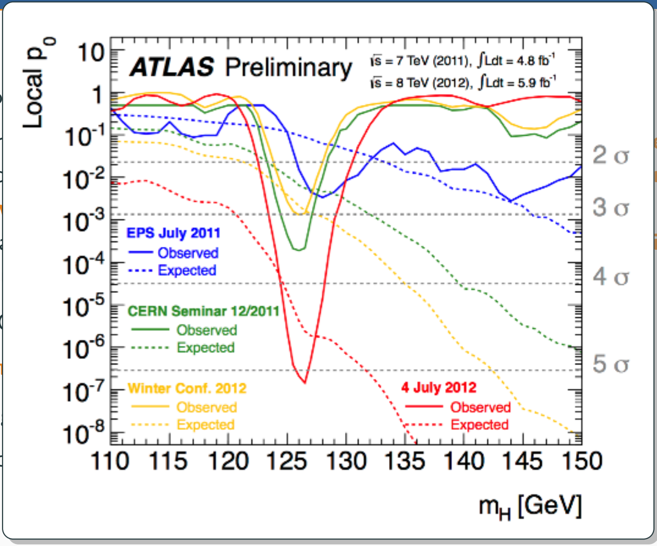


x	y	z
-1.10228	-1.79939	4.452822
1.867178	-0.59662	3.842313
-0.52418	1.868521	3.766139
-0.38061	0.969128	1.084074
[illegible]	-0.21231	[illegible]
[illegible]	1.187305	[illegible]
[illegible]	[illegible]	0.635417
1.079222	[illegible]	1.271904
-0.27492	[illegible]	3.038899
2.047779	[illegible]	4.197329
-0.45868	[illegible]	2.293266
0.304731	[illegible]	0.875442
[illegible]	[illegible]	0.556881
[illegible]	1.181767	[illegible]
[illegible]	-0.65411	[illegible]
-2.03555	0.527648	4.421883
-1.45905	-0.464	2.344113
1.230661	-0.00565	1.514559
[illegible]	[illegible]	3.562347



ROOT in a Nutshell

- ROOT is an o
  - Data ser
  - Data pro
    workflow
  - Data ana
  - Data vis
- ROOT has a C
  - with **dyn**
- Adopted in Hi
  - $\mathcal{O}(EB)$ o



ecords
ntegration, **parallel**
istics functions

ROOT in a Nutshell

- ROOT is an open source software framework with building blocks for
  - Data serialization and storage: **C++ reflection**, **column-wise nested records**
  - Data processing: **C++ interpretation and jitting**, **Jupyter notebook integration**, **parallel workflow management**
  - Data analysis: **Declarative analysis**, **machine learning**, **math and statistics functions**
  - Data visualization: **histograms**, **graphs**, **event viewer**
- ROOT has a C++ core
  - with **dynamic** bindings for Python
- Adopted in High Energy Physics and beyond (e. g. GeneROOT)
  - $\mathcal{O}(EB)$ of data in ROOT format

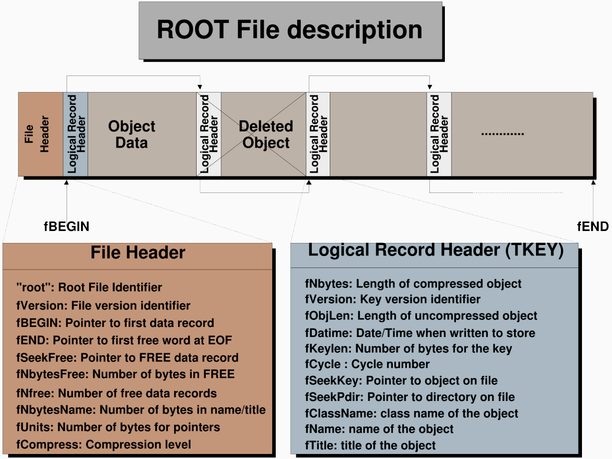
ROOT I/O Goals

Tailor-made persistency layer for High Energy Physics (HEP) event data, aiming at

- Smallest files and fastest read/write speed for typical HEP data
  - Transparent lossless and lossy compression
  - Async I/O
  - Multi-threaded reading and writing, support for accelerators (e. g. for decompression)
- Native support for object stores (e. g. DAOS)
- Careful API and data format design to prevent silent I/O errors
- Supporting decades-long lifetime of modern HEP experiments
  - Stable API and on-disk format
  - Automatic and customizable schema evolution

ROOT File

- Container for streamed, self-describing C++ objects
- Hierarchical internal structure, embedded file system
- Often used to combine data (column-wise events) with histograms (row-wise summary objects)



ROOT Tree/RNTuple

TTree / RNTuple provide column-wise storage of nested collections

- Only few other column-wise formats for nested records
  - Apache Parquet (Google Dremel) optimized for deep, sparse collections: our data is not sparse
  - Apache Arrow: transient, in-memory format
- ROOT's unique feature:
  seamless C++ integration
  Users do not need to maintain explicit data schema

Serialization of Nested Collections

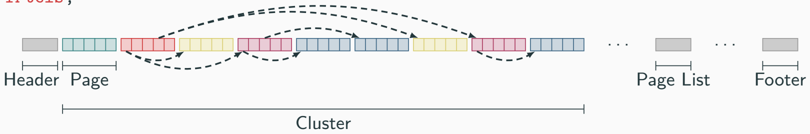
```
struct Event {
  int fId;
  std::vector<Particle> fParticles;
};

struct Particle {
  float fEnergy;
  std::vector<int> fCandidateIds;
};
```

Grossly simplified.

Think 10 k properties, 10 k physicists, 1 EB

Breakdown of the RNTuple on-disk format

```
struct Event {
    int fId;
    vector<Particle> fPtcls;
};
struct Particle {
    float fE;
    vector<int> fIds;
};
```



Cluster

- Block of consecutive complete events
- Defaults to 50 MB compressed

Page

- Unit of (de-)compression and (un-)packing
- Defaults to 64 kB uncompressed

Essential I/O features for High Energy Physics

- User interface
  - C++ and Python support
  - Transparent compression
  - Horizontal and vertical data combinations (joins, unions)
- Writing data
  - (Fast) merging of files without decompressing blocks
  - Schema extension during writing
- Efficient sparse streaming from remote storage (HTTP etc.)
- Automatic and customizable schema evolution

Summary

- ROOT I/O layer provides the persistency layer for High Energy Physics
- Designed for the HEP data model: columnar, nested collections
- Designed around distributed (federated) Big Data computing model (High Throughput Computing)
- Increasing share of resources, however, provided by HPC centers
  - Interested to learn from the HDF5 experience in High-Performance Data Analytics, for instance
    - distributed parallel writing
    - distributed merging of files
    - interplay between cluster file system and application I/O

https://lonelychairsatcern.tumblr.com

Thank you for your time!