

# Interactive Data Analysis for End Users on HN Science Cloud

---

## Status Report on Totem Test

**Enrico Bocchi**  
CERN IT, Storage



Helix Nebula Science Cloud Pilot Phase  
14<sup>th</sup> June 2018, CERN, Geneva

# Contributors

---

- **TOTEM Experiment // AGH Krakow**

V. Avati, M. Blaszkiewicz, L. Grzanka, M. Malawski, A. Mnich



- **EP, Software Development for Experiments**

D. Castro, D. Piparo, E. Tejedor



- **IT, Database Services**

L. Canali, P. Kothuri

- **IT, Storage**

E. Bocchi, J.T. Moscicki

# What is the Totem Test about

---

- **Goal:** Enable end users to run data analysis interactively
  - ✓ Analysis scenario: Totem experiment
  - ✓ End users: Totem collaborators and scientists
- **How:** Deployment of “Science Box” on Helix Nebula Cloud
  - ✓ EOS, CERNBox, SWAN + SPARK
  - ✓ Data transfer to import relevant datasets
- **Measure of success:**
  - Quantification of analysis done with SWAN
  - ✓ Real physics analysis with ROOT (RDataFrame) + SPARK
  - ✓ Functional tests + synthetic benchmarks via specialized tools

# First Tests (2017)

- “Science Box” on a single VM

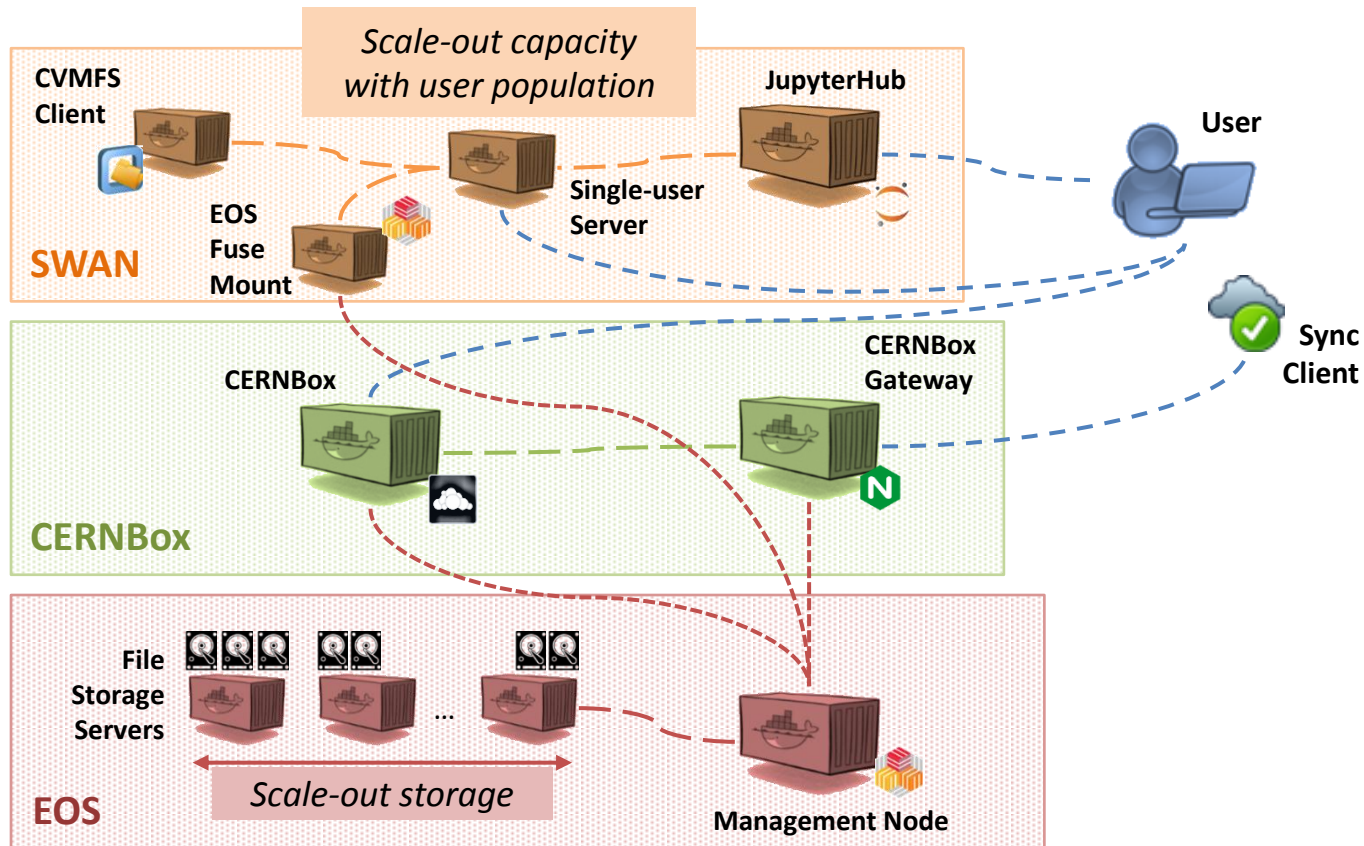


- ✓ Streamlined installation – No configuration required
- ✓ Automated functional tests for validation

- ✓ IBM
- ✓ RHEA
- ✓ T-Systems

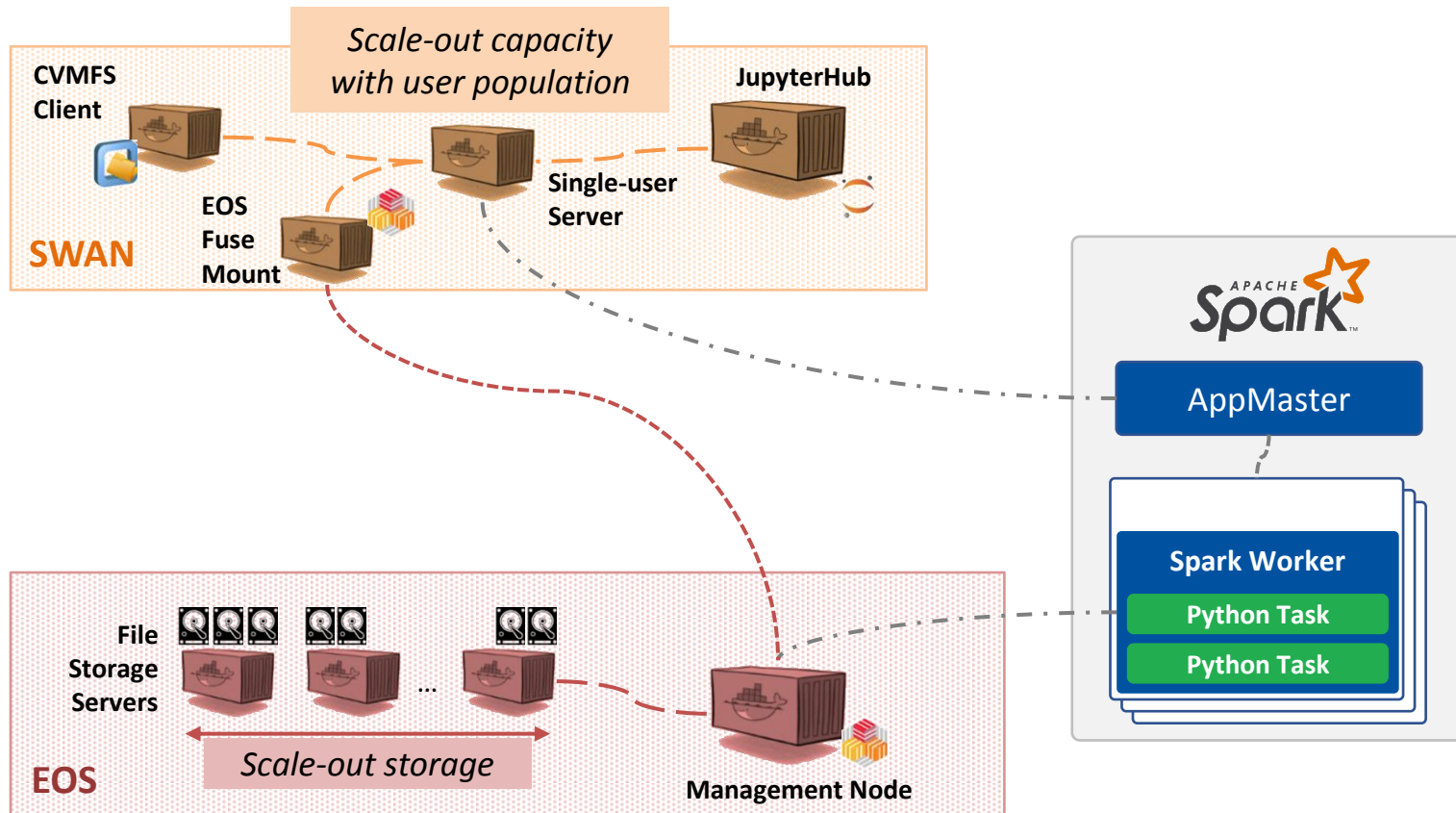
# Current Tests: Science Box

- Scalable deployment with Kubernetes



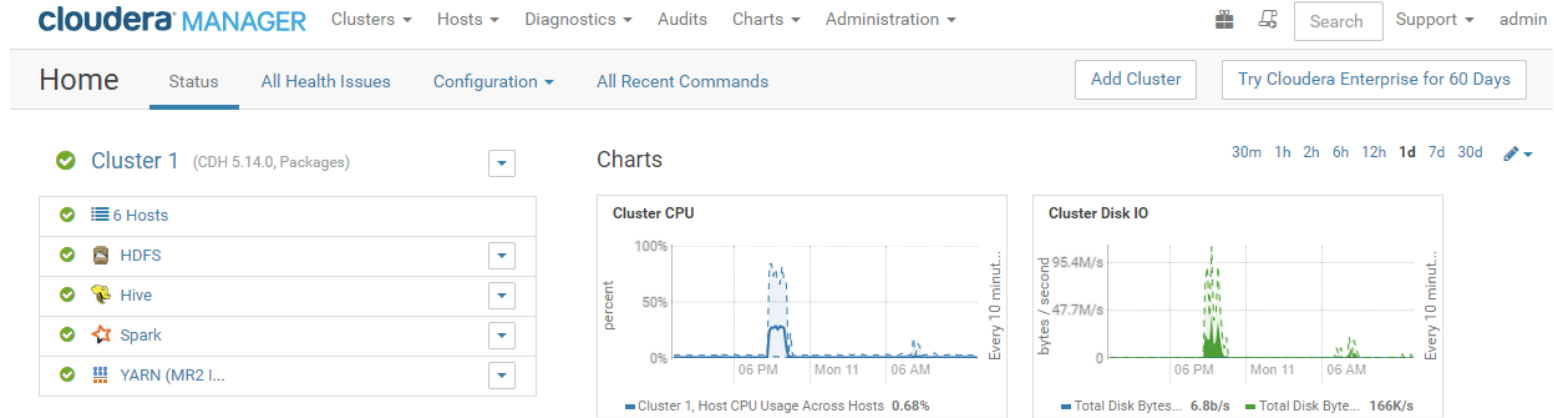
# Current Tests: Science Box

- Scalable deployment with Kubernetes
- SPARK as a computing engine



# Current Tests: Spark Cluster

- ✓ Hadoop and Spark installed via yum on plain VMs
- ✓ Configuration via Cloudera Manager
- ✓ Validation with industry standard TPC-DS benchmark + user workloads



## Current Installation

- ✓ 6 nodes
- ✓ ECS Type: s2.xlarge.8
- ✓ Hadoop: CDH\_5.14
- ✓ Spark: 2.2.1
- ✓ JAVA: 1.8.0\_161-b12

# Deployment Status in HN Cloud

- Deployment **OK** on OTC
- All resources managed as sub-tenant “*TOTEM\_Test*”
- Small scale for now
  - ✓ 22 VMs, 41 Block Devices
  - ✓ 128 CPUs, 488GB Memory
  - ✓ 22 System Volumes (950GB)
  - ✓ 19 Data Volumes (~12TB)
- Functional tests successful
  - ✓ Storage, Sync, Analysis
  - ✓ SWAN to Spark connection
  - ✓ Import small dataset

The screenshot displays the Open Telekom Cloud (OTC) dashboard. At the top, the navigation bar includes the OTC logo, the text "OPEN TELEKOM CLOUD", and a dropdown menu for the region "eu-de(T...)" which is circled in blue. Below the navigation bar, the "All Services" section shows a list of services: "Elastic Cloud Server (22)", "Cloud Server Backup Service (0)", "Elastic Volume Service (41)", and "Volume Backup Service (0)". A blue arrow points from the "Elastic Cloud Server (22)" service to a table below. The table lists the following resources:

Name/ID	AZ	Status	Specifications/Image
<input type="checkbox"/> eos-mgm 02fdd392-0b2b-41f0-82...	eu-de-02	→ Running	16 vCPUs   32 GB Standard_CentOS_7_latest
<input type="checkbox"/> swan 05d69a83-5426-4349-a...	eu-de-02	→ Running	32 vCPUs   128 GB Standard_CentOS_7_latest
<input type="checkbox"/> spark-0000 1fe61b77-e8e8-4e72-97...	eu-de-02	→ Running	4 vCPUs   32 GB Standard_CentOS_7_latest



# Deployment Status in HN Cloud

Demo at Lunchtime  
CERN, Restaurant 2

CERN Accelerating science

**EOS**  
Disk-based storage service.  
Docs More Info

**CERNBox**  
Cloud Storage with Sync & Share.  
Try it! More Info

**SWAN**  
Interactive Data Analysis in the Cloud.  
Try it! More Info

## More on Technology

<http://cernbox.cern.ch/cernbox/doc/boxed>

## OTC Resource Console

<https://myworkplace.t-systems.com/MyWorkplace/Login.aspx>

### FEEDBACK

Get in touch with us at:  
*science-box (at) cern (dot) ch*

# Experience with OTC as a User

---

## ■ Incidents:

- ✓ May 7<sup>th</sup> Outage:
  - 3 containers lost due to VMs reboot
  - Manual intervention required
- ✓ May 14<sup>th</sup>: VM reported IO errors on system disk – VM was rebuilt
- ✓ Early June: OTC web console unavailable for short periods

Dear user of the Open Telekom Cloud,

We do apologize for the partial downtime of our Open Telekom Platform on 7<sup>th</sup> may and we would like to provide you the final error analysis of Open Telekom Cloud's partial outage.

**Date / time:** 2018-05-07, 21:25 - 22:24 CEST (19:25 - 20:24 UTC)

## ■ Requests:

- ✓ Streamline VM name <==> hostname <==> IP address resolution  
E.g., Internal DNS to resolve VM name to its private IP address  
Worked-around with entries in /etc/hosts + dnsmasq

# Next Steps

---

- Application layer optimization for TOTEM use-case
  - ✓ Performance tuning, caching of frequently-used software packages, ...
  - ✓ Focus on relevant metrics for users, e.g., total execution time
- Scale-out storage and computing resources
  - ✓ Leverage on scalable architecture design (EOS, Swan, Spark)
  - ✓ Storage:
    - Import larger dataset (10-50TB) → Additional file servers and block devices
    - Investigate caching layer for Spark for improved performance
  - ✓ Computing:
    - Experiment and identify best VM flavor for Spark computing nodes
    - Additional spark workers to crunch bigger datasets

# Interactive Data Analysis for End Users on HN Science Cloud

---

## Status Report on Totem Test

**Enrico Bocchi**  
CERN IT, Storage

Helix Nebula Science Cloud Pilot Phase  
14<sup>th</sup> June 2018, CERN, Geneva

# Backup Slides

---

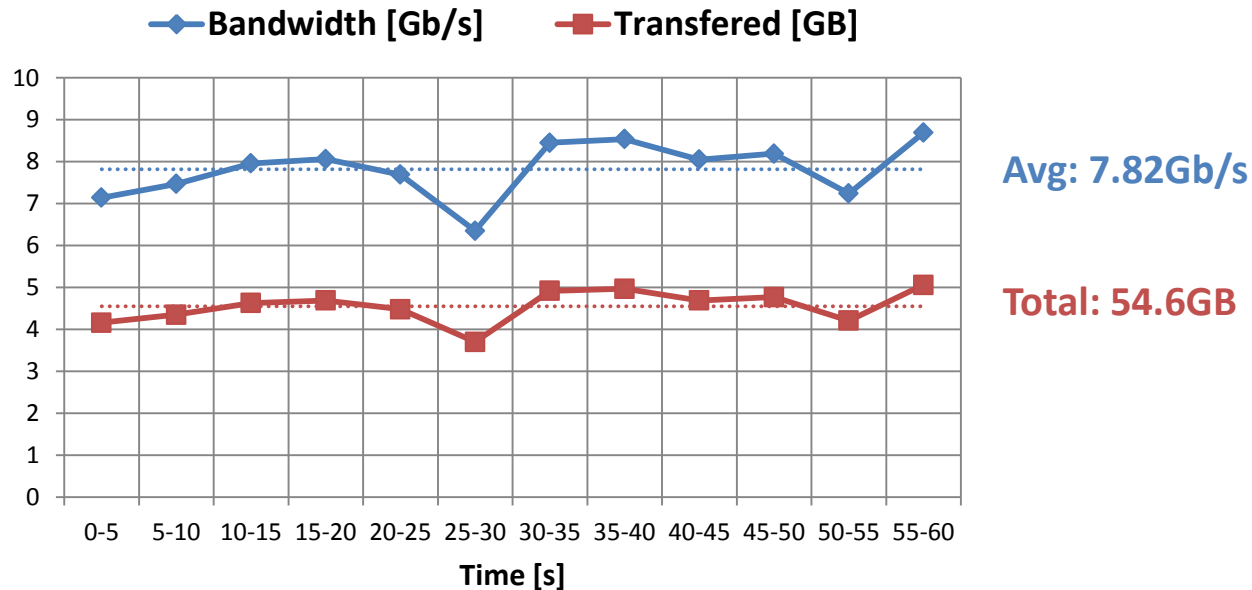
# Project Description

---

- This is a deployment test of the ScienceBox IT services (EOS, SWAN, CERNBox, and Spark) in a particular analysis scenario for TOTEM experiment (conducted jointly with SFT group and interested TOTEM collaborators).
- Phase 1 (April-September) is a feasibility study: (a) verification of the deployment of the services; (b) performance tuning and testing of a TOTEM data analysis example. If feasibility tests are successful then Phase 2 (September-November) is to use available resources opportunistically to understand the scaling limitations of the system and, if possible, to actually perform analysis for use-cases defined by the TOTEM collaborators.
- The amount of data currently foreseen is ~10-50TB. HN resource is treated as an extension of CERN resource. Any data stored in HN cloud will be automatically erased at the end of 2018.
- This project does not provide guarantee of service but will make opportunistic use of resources if feasible.

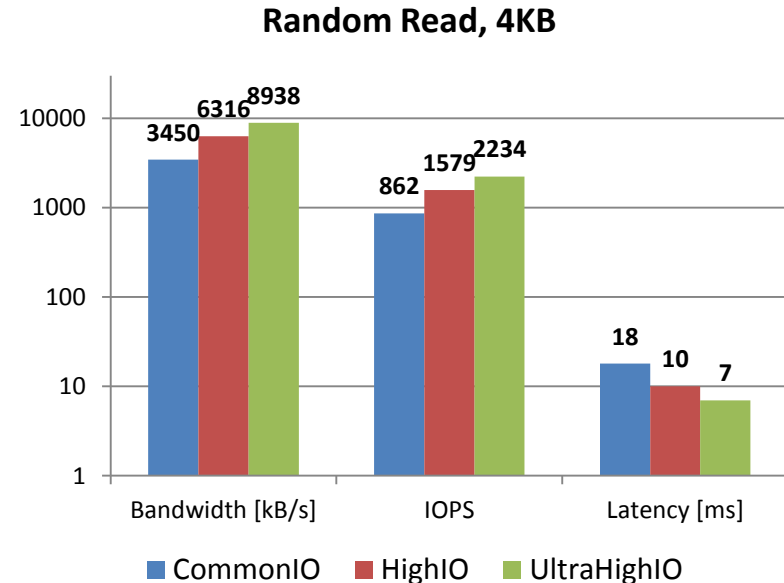
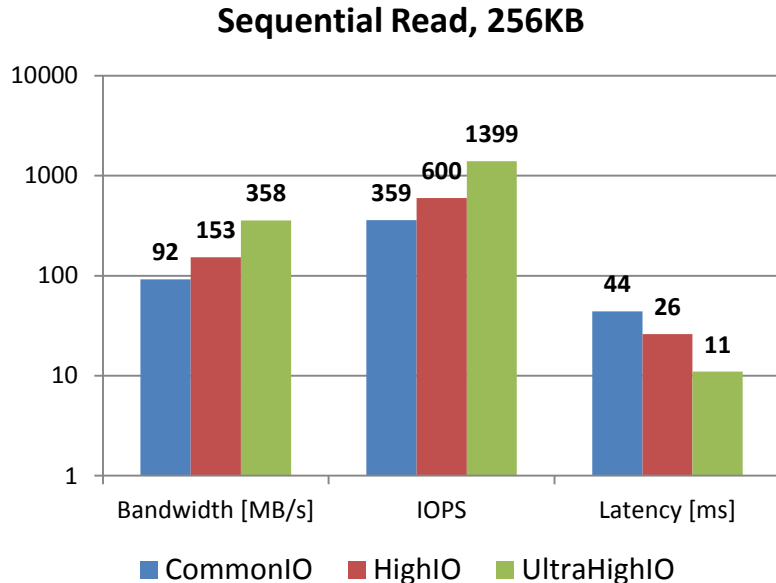
# Synthetic Benchmarks

- Network – iperf, ping
  - ✓ TCP connection, 128K buffer, 60s test (5s binning)
  - ✓ Average bandwidth: ~7.8 Gb/s (both directions)
  - ✓ Very low latency (~0.3ms) and jitter (sdev ~0.07ms)



# Synthetic Benchmarks (cont'd)

- Block Devices – fio
  - ✓ Three types of VDBs: Common-IO, High-IO, and UltraHigh-IO
  - ✓ No disk encryption, No disk sharing
  - ✓ Size form 10GB to 1TB (not related to performance)





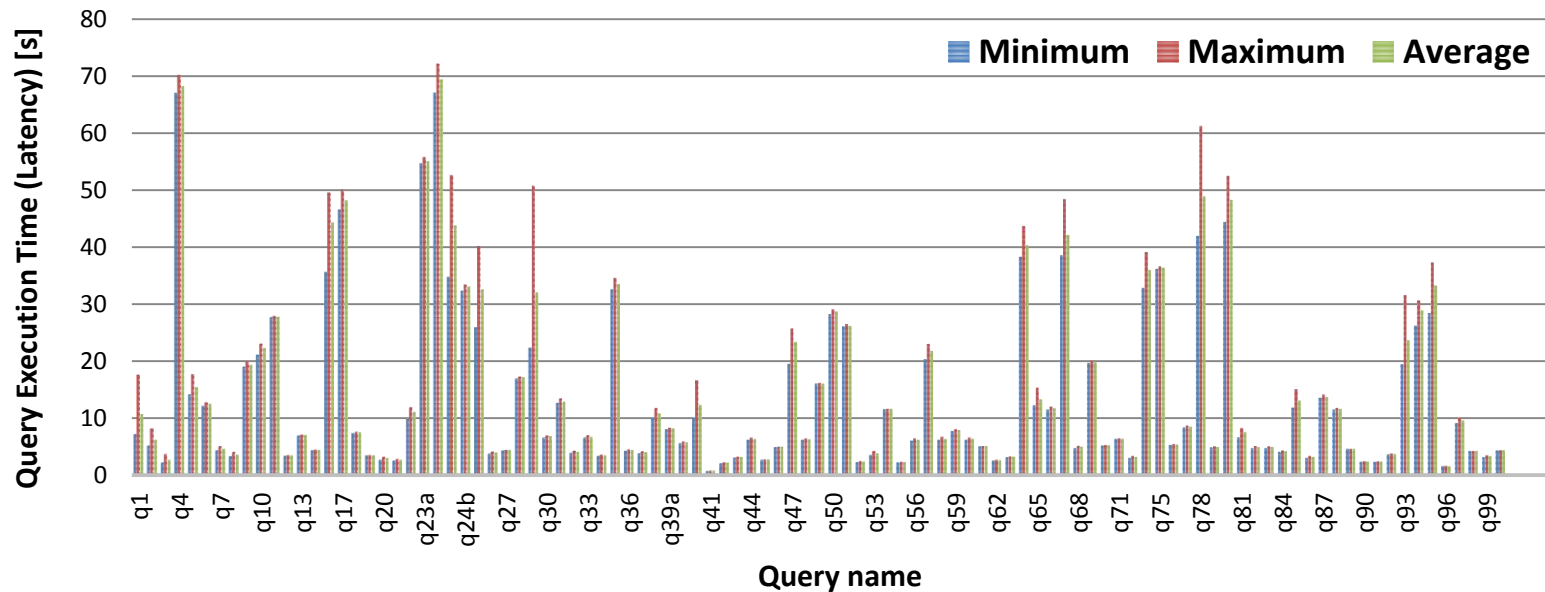
# Synthetic Benchmarks (cont'd)

---

- CPU and Memory – sysbench
  - ✓ CPU - Total events: 10'000
    - Avg. time per event: 1.11ms
    - 95<sup>th</sup> percentile: 1.12ms
  - ✓ Memory – Transfer (write) 10GB
    - IOPS: 1'614'325
    - Throughput: 1576.49 MB/s
- Two (main) flavor of VMs:
  - ✓ 4CPUs, 8GB memory – General purpose VM
  - ✓ 4CPUs, 32GB memory – Spark workers
  - ✓ Performance is consistent across VM sizes

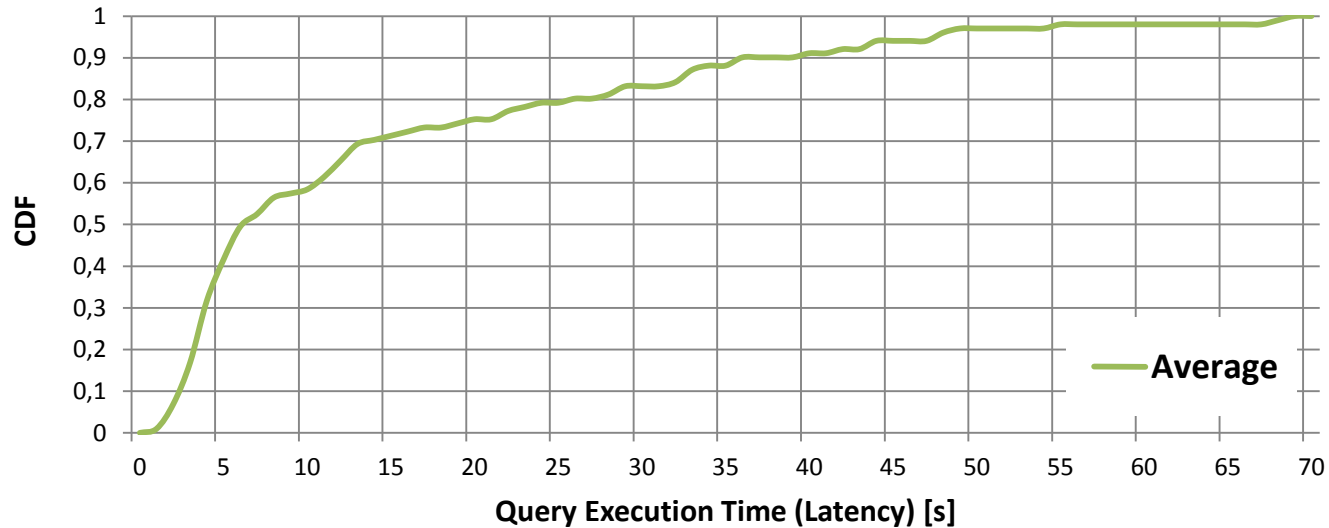
# Synthetic Benchmarks (cont'd)

- Spark Cluster – TPC-DS Benchmark
  - ✓ Small case test: Dataset size 20GB
  - ✓ 5 executors (2 cores, 7GB memory), Query Set v1.4, Spark 2.2.1



# Synthetic Benchmarks (cont'd)

- Spark Cluster – TPC-DS Benchmark
  - ✓ Small case test: Dataset size 20GB
  - ✓ 5 executors (2 cores, 7GB memory), Query Set v1.4, Spark 2.2.1



# Data Analysis with RDataFrame

---

```
df = RDataFrame('t', 'f.root')
hist = df.Filter('theta > 0') \
        .Histo1D('pt')
hist.Draw()
```

← - - - - Build data-frame object

← - - - - Apply transformation

← - - - - Apply actions

- Columnar dataset: ROOT, other formats
- Declarative analysis: **what**, not how
  - ✓ Transformations: cuts, definition of new columns
  - ✓ Actions: return a result (e.g., histogram)
- Implicit parallelisation
  - ✓ Multi-threaded, distributed

# RDataFrame: Distributed Execution

- RDataFrame supports distributed execution on top of Spark
  - ✓ No changes in the app will be required to go distributed!

The screenshot shows a Jupyter Notebook interface with a menu bar (FILE, EDIT, VIEW, INSERT, CELL, KERNEL, HELP) and a toolbar. The notebook contains the following content:

```
return myHist
```

In the current implementation, a second function is needed to merge partial results - in this case, partially filled histograms coming from the distributed parallelization. This requirement **will be removed** for the programming model to be simpler, and the implementation will automatically take care of the merge.

```
In [10]: def mergeHist(h1, h2):
         h1.Add(h2)
         return h1
```

The next cell launches the distributed computation with Spark, where a `DistTree` is constructed from three parameters:

- List of files of our dataset
- Tree name
- Desired number of partitions of our dataset (will potentially determine the amount of parallelism we will achieve)

Once the `DistTree` is created, we invoke the `ProcessAndMerge` method on it to trigger the distributed processing. The two arguments of this call are the two functions defined previously.

```
In [*]: dTree = DistTree(filelist = ['root://eostotem//eos/totem/data/cmstotem/2015/90m/TotemNtuple/version2/4495/TotemNtuple_9883.040.ntuple',
                                   #filelist = ["/eos/totem/data/cmstotem/2015/90m/TotemNtuple/version2/4495/TotemNtuple_9883.040.ntuple",
                                   treename = "TotemNtuple",
                                   npartitions = 2])

myHist = dTree.ProcessAndMerge(fillHist, mergeHist)
```

At the bottom, the Apache Spark status bar shows:

- Apache Spark: 2 EXECUTORS 2 CORES Jobs: 1 RUNNING

Job ID	Job Name	Status	Stages	Tasks	Submission Time	Duration
1	treeReduce	RUNNING	0/1 (1 active)	0 + 2/2	a few seconds ago	-

# RDataFrame: Spark Monitoring

- Automatic monitoring when an RDataFrame Spark job is spawned
- Job progress, task distribution, resource utilization
  - ✓ Dataset split in ranges under the hood, one task per range

```
In [4]: dTree = DistTree(filelist = ['root://eostotem//eos/totem/data/cmstotem/2015/90m/Totem/Ntuple/version2/4495/TotemNTuple_9883.040.ntuple.root'],  
                        treename = "TotemNtuple",  
                        npartitions = 4)  
  
myHist = dTree.ProcessAndMerge(fillHist, mergeHist)
```

