Systems and Architectures for Big Data AY 2017/2018
University of Rome Tor Vergata

Batch processing for smart plugs sensor data with Apache Spark on Google DataProc

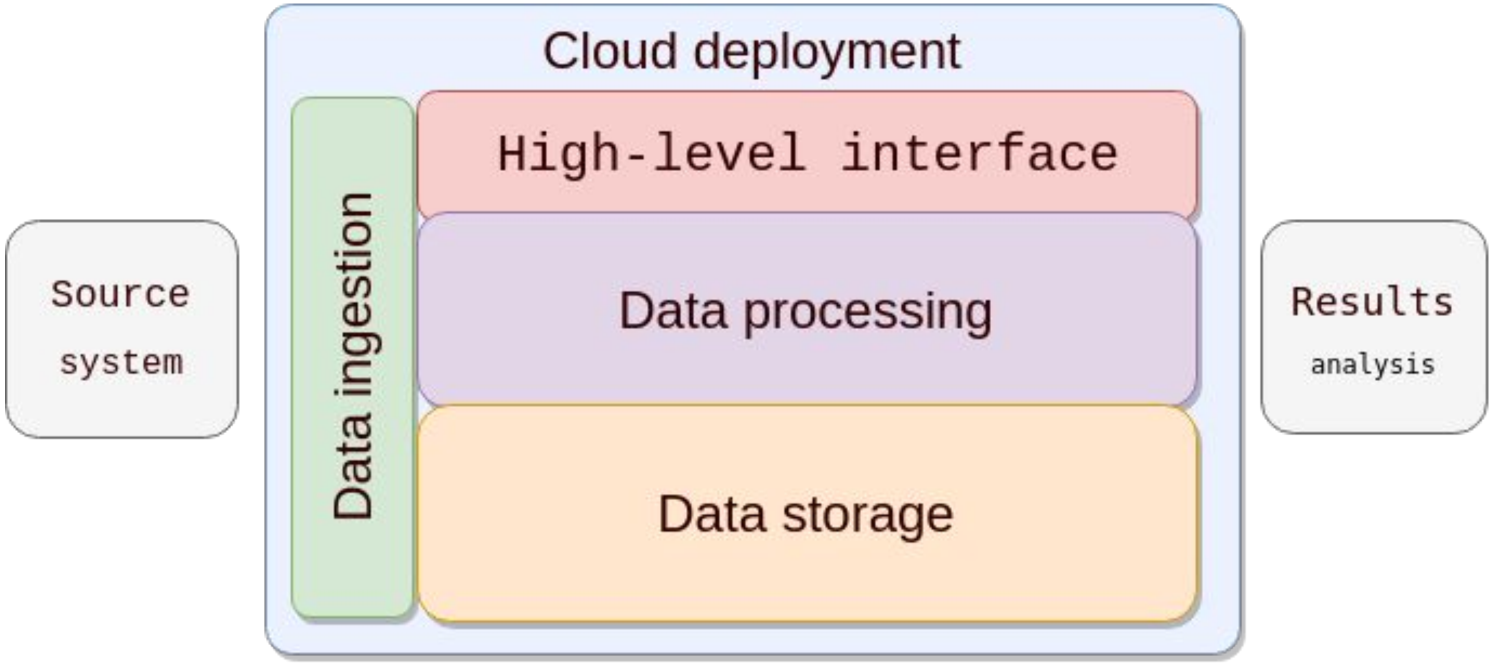
Ovidiu Daniel Barba
Laura Trivelloni
Emanuele Vannacci

1.

System Architecture

High Level Overview

System Layers



Data Processing Layer

- Basic RDD API
- Scala as the programming language
  a. Spark written in Scala
  b. Functional Programming Functions similar with RDD API

High Level Interface

SQL

- Support for structured and semi-structured data using DataFrame API
- SQL-like query language
- Our own UDF (User Defined Functions)

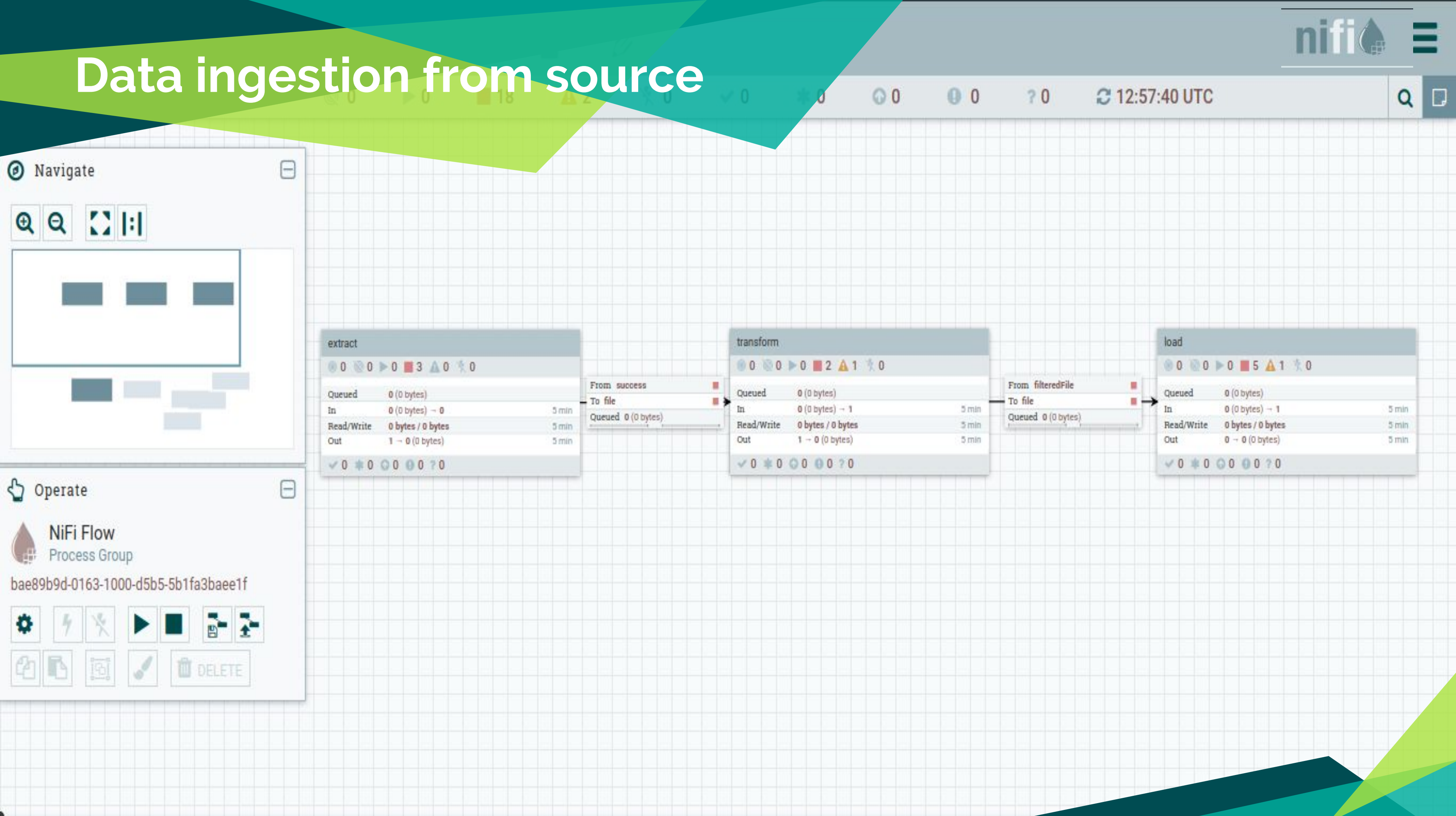
Data Storage

- HDFS stores and serves input, results and benchmark files
- Alluxio handles all data read, write and cache operations on HDFS theoretically improving performance up to 10x

Data Ingestion

- Inject data from external sources to data storage layer
- Transfer data between components in storage layer
- Data filtering

Data ingestion from source



NiFi Flow
Process Group
bae89b9d-0163-1000-d5b5-5b1fa3baee1f

extract
0 0 0 3 0 0

Queued	0 (0 bytes)	
In	0 (0 bytes) → 0	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	1 → 0 (0 bytes)	5 min

0 0 0 0 ? 0

From success
To file
Queued 0 (0 bytes)

transform
0 0 0 2 1 0

Queued	0 (0 bytes)	
In	0 (0 bytes) → 1	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	1 → 0 (0 bytes)	5 min

0 0 0 0 ? 0

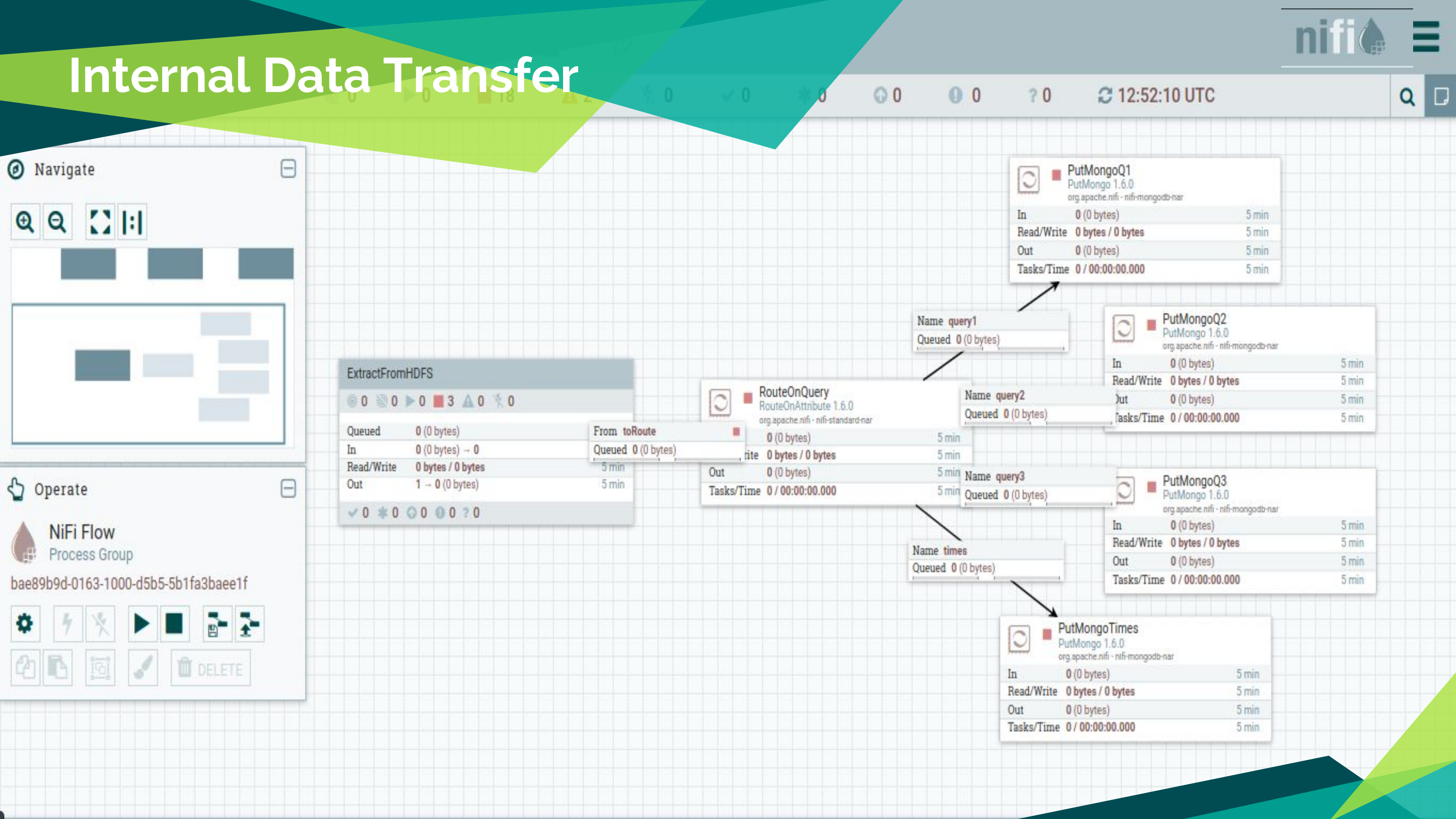
From filteredFile
To file
Queued 0 (0 bytes)

load
0 0 0 5 1 0

Queued	0 (0 bytes)	
In	0 (0 bytes) → 1	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 → 0 (0 bytes)	5 min

0 0 0 0 ? 0

Internal Data Transfer



ExtractFromHDFS

0 0 0 3 0 0

Queued	0 (0 bytes)	
In	0 (0 bytes) → 0	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	1 → 0 (0 bytes)	5 min

0 0 0 0 ? 0

From toRoute
Queued 0 (0 bytes)

RouteOnQuery
RouteOnAttribute 1.6.0
org.apache.nifi - nifi-standard-nar

	0 (0 bytes)	5 min
rite	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

Name query1
Queued 0 (0 bytes)

Name query2
Queued 0 (0 bytes)

Name query3
Queued 0 (0 bytes)

Name times
Queued 0 (0 bytes)

PutMongoQ1
PutMongo 1.6.0
org.apache.nifi - nifi-mongodb-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

PutMongoQ2
PutMongo 1.6.0
org.apache.nifi - nifi-mongodb-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

PutMongoQ3
PutMongo 1.6.0
org.apache.nifi - nifi-mongodb-nar

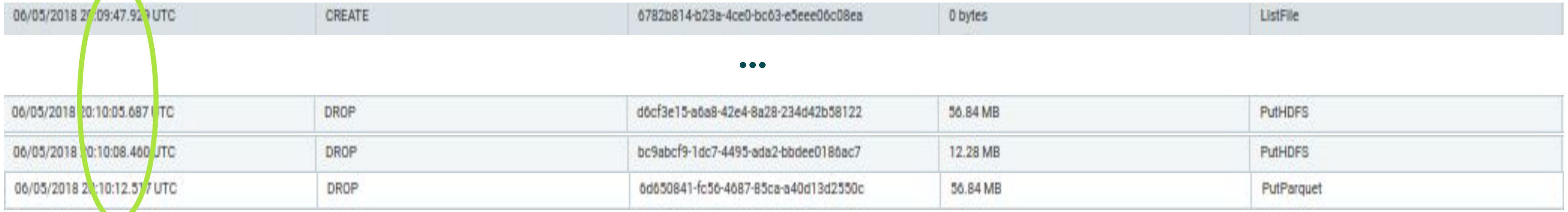
In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

PutMongoTimes
PutMongo 1.6.0
org.apache.nifi - nifi-mongodb-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

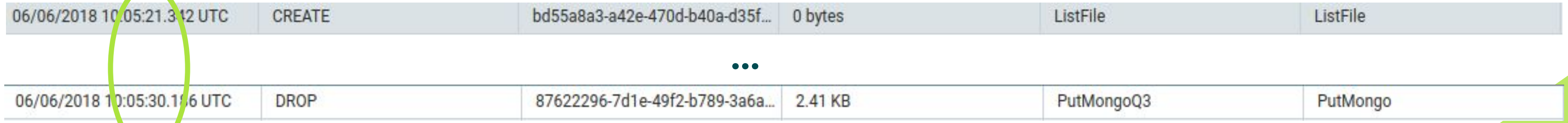
Data provenance lineage

Data ingestion from local to HDFS < 30 sec



06/05/2018 20:09:47.929 UTC	CREATE	6782b814-b23a-4ce0-bc63-e5eee06c08ea	0 bytes	ListFile
		…		
06/05/2018 20:10:05.687 UTC	DROP	d6cf3e15-a6a8-42e4-8a28-234d42b58122	56.84 MB	PutHDFS
06/05/2018 20:10:08.460 UTC	DROP	bc9abcf9-1dc7-4495-ada2-bbdee0186ac7	12.28 MB	PutHDFS
06/05/2018 20:10:12.517 UTC	DROP	6d650841-fc56-4687-85ca-a40d13d2550c	56.84 MB	PutParquet

Results collection from HDFS to MongoDB < 10 sec



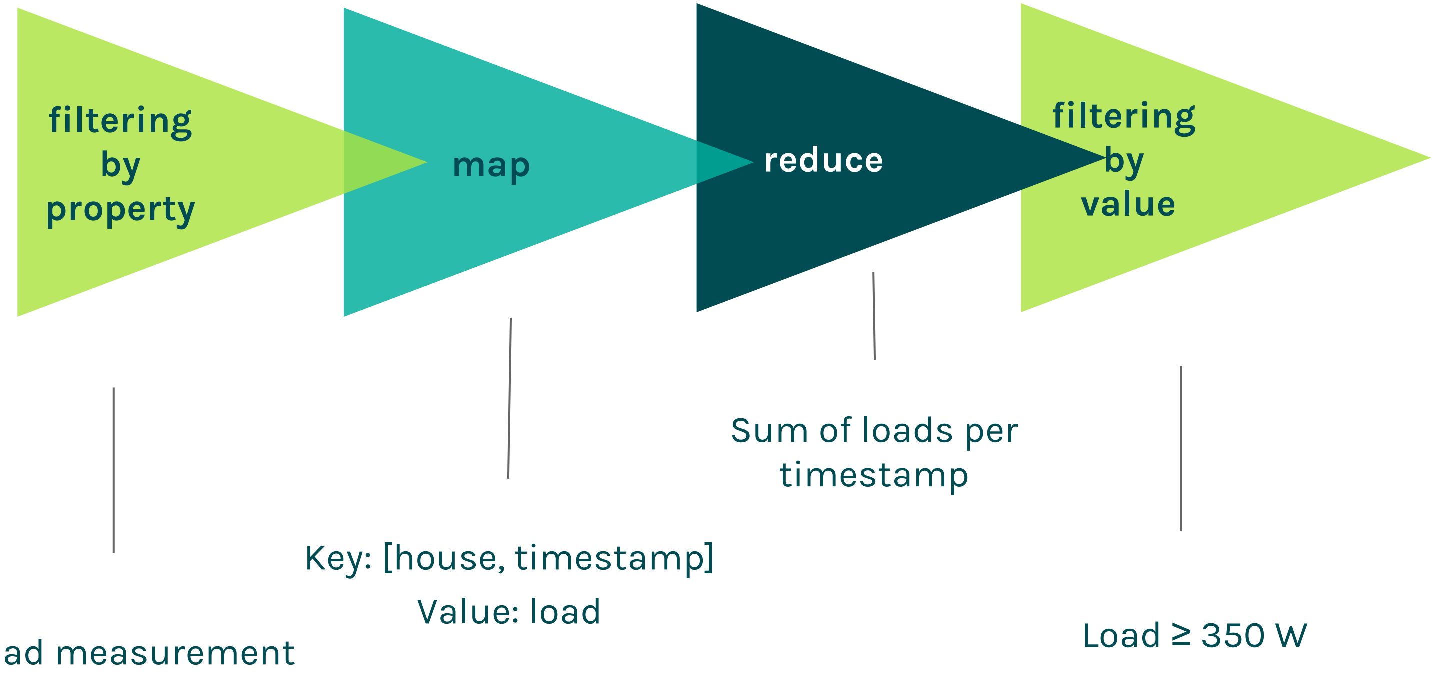
06/06/2018 10:05:21.342 UTC	CREATE	bd55a8a3-a42e-470d-b40a-d35f...	0 bytes	ListFile	ListFile
		…			
06/06/2018 10:05:30.186 UTC	DROP	87622296-7d1e-49f2-b789-3a6a...	2.41 KB	PutMongoQ3	PutMongo

2.

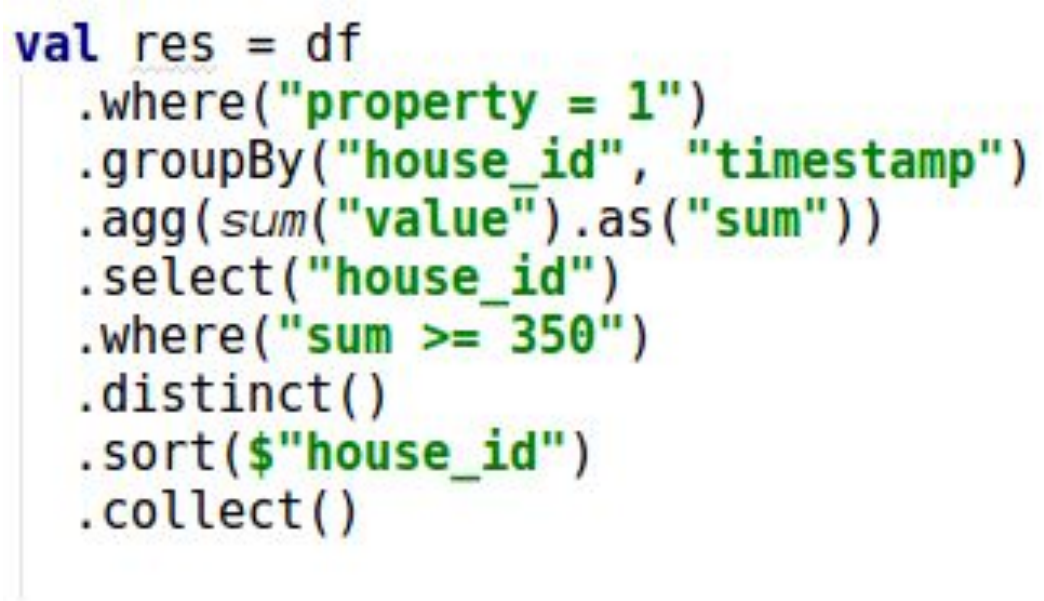
Queries

Detailed queries description

Query 1

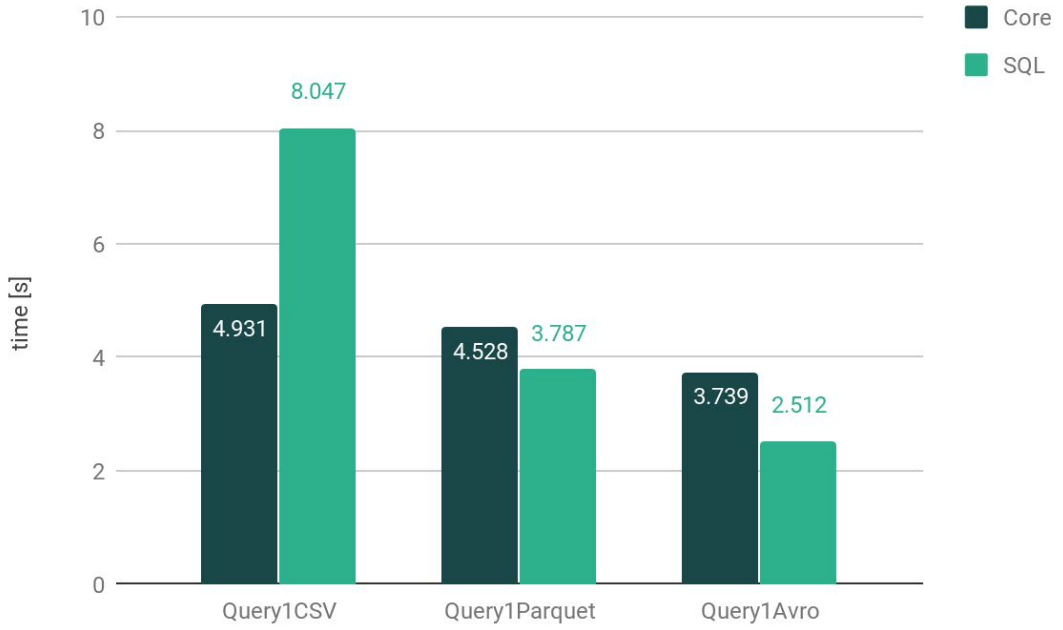

filtering by property
map
reduce
filtering by value
Load measurement
Key: [house, timestamp]
Value: load
Sum of loads per timestamp
Load ≥ 350 W

Query 1 Spark SQL



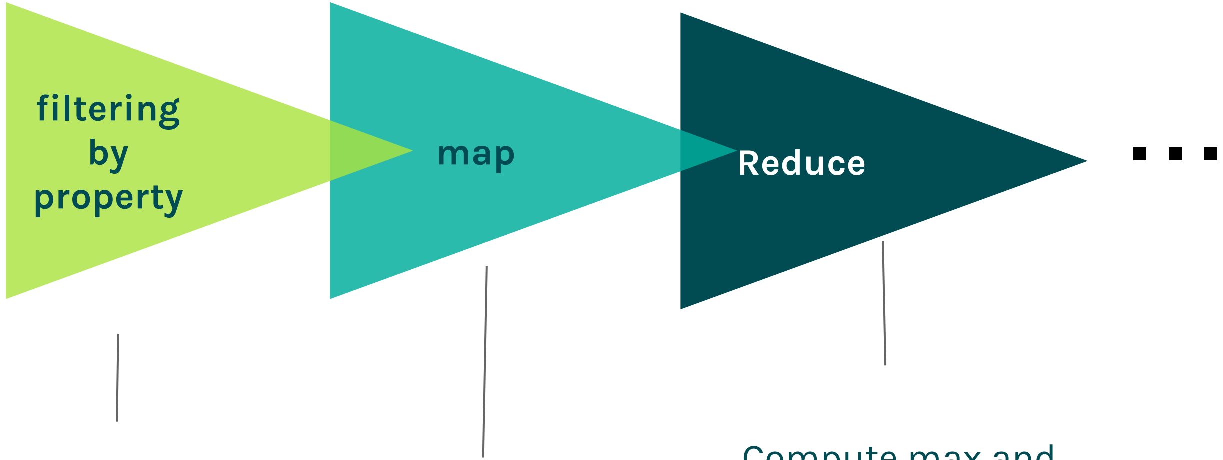
```
val res = df
    .where("property = 1")
    .groupBy("house_id", "timestamp")
    .agg(sum("value").as("sum"))
    .select("house_id")
    .where("sum >= 350")
    .distinct()
    .sort($"house_id")
    .collect()
```

Query 1 performances



Spark Core and Spark SQL implementations performances.

Query 2



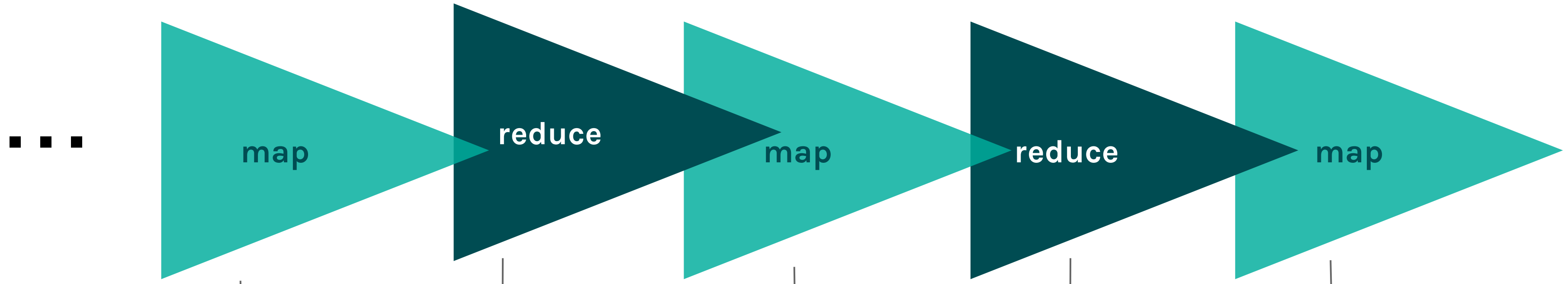
Work measurement

Key:
[house, household, plug, time slot, day, month]

Value: *MaxMinHolder* init by work

Compute max and min value per slot

Query 2



Sum of increments

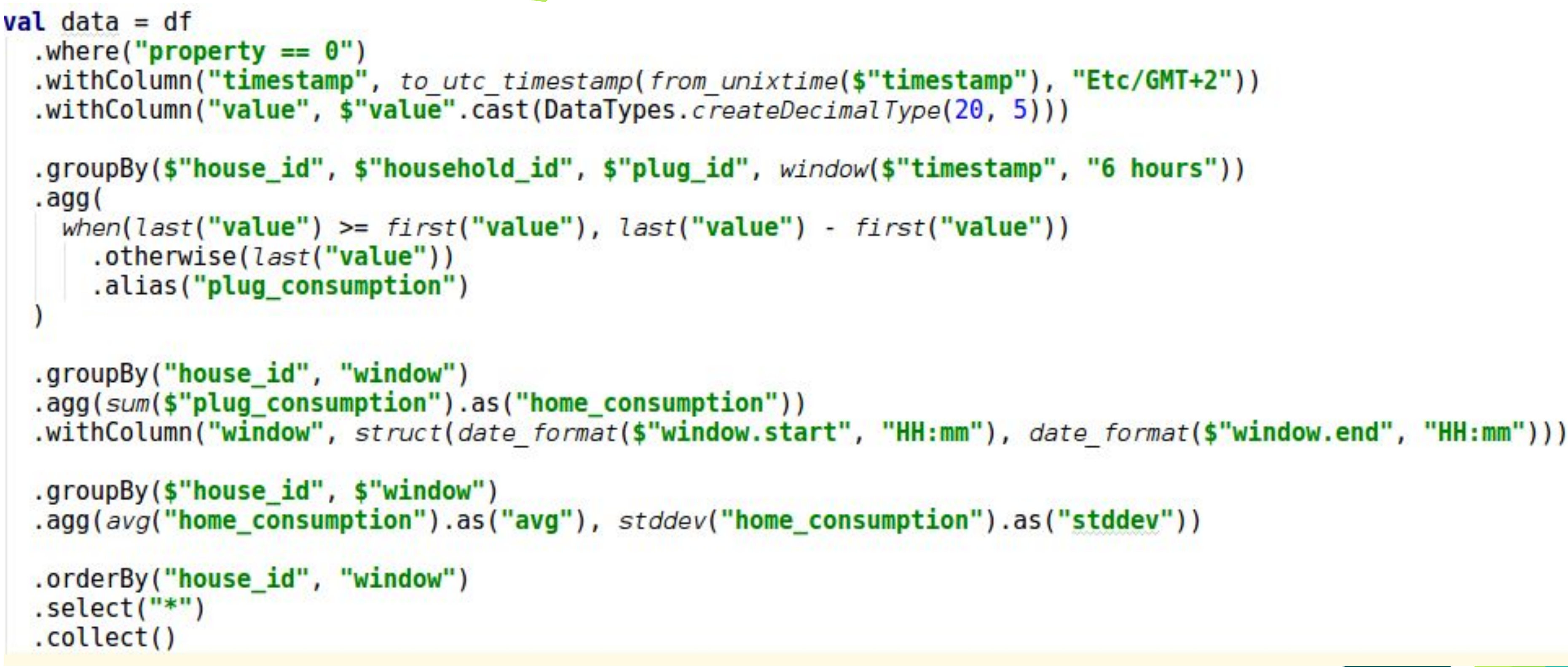
Sum of consumption, count of items, sum of consumption^2

Key:
[house, time slot, day, month]
Value: [consumption per slot]

Key:
[house, time slot]
Value: [consumption, 1, consumption^2]

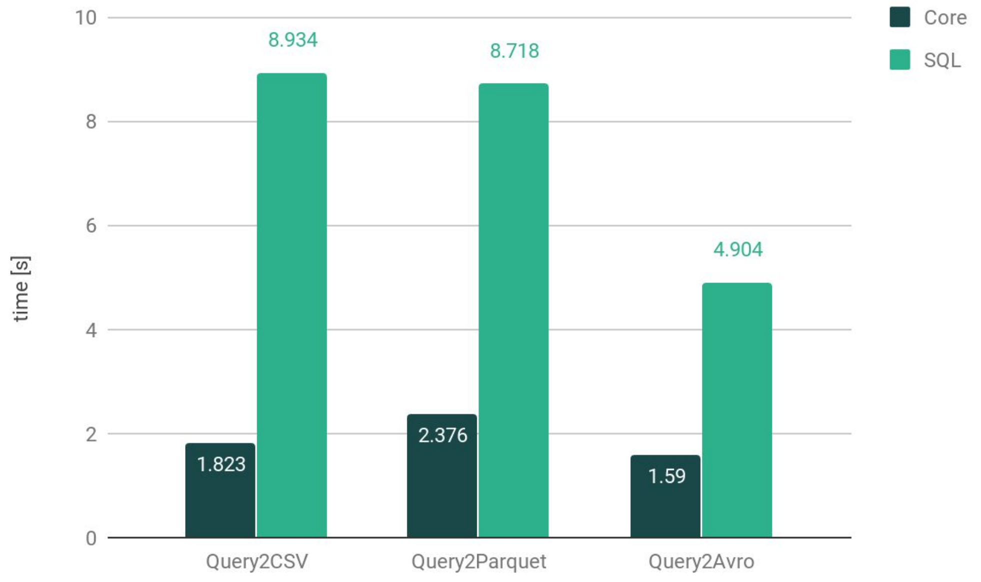
Key:
[house, time slot]
Value: [mean, standard deviation]

Query 2 Spark SQL



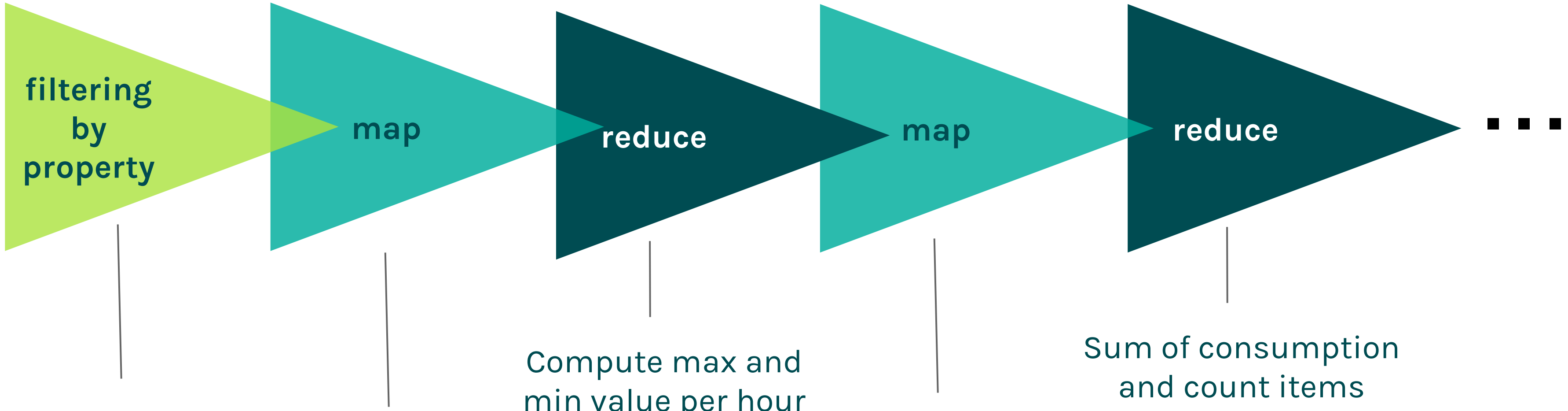
```
val data = df
  .where("property == 0")
  .withColumn("timestamp", to_utc_timestamp(from_unixtime($"timestamp"), "Etc/GMT+2"))
  .withColumn("value", $"value".cast(DataTypes.createDecimalType(20, 5)))

  .groupBy($"house_id", $"household_id", $"plug_id", window($"timestamp", "6 hours"))
  .agg(
    when(last("value") >= first("value"), last("value") - first("value"))
      .otherwise(last("value"))
      .alias("plug_consumption")
  )

  .groupBy("house_id", "window")
  .agg(sum($"plug_consumption").as("home_consumption"))
  .withColumn("window", struct(date_format($"window.start", "HH:mm"), date_format($"window.end", "HH:mm")))

  .groupBy($"house_id", $"window")
  .agg(avg("home_consumption").as("avg"), stddev("home_consumption").as("stddev"))

  .orderBy("house_id", "window")
  .select("*")
  .collect()
```

Query 2 performances



Spark Core and Spark SQL implementations performances.

Query 3



Load measurement

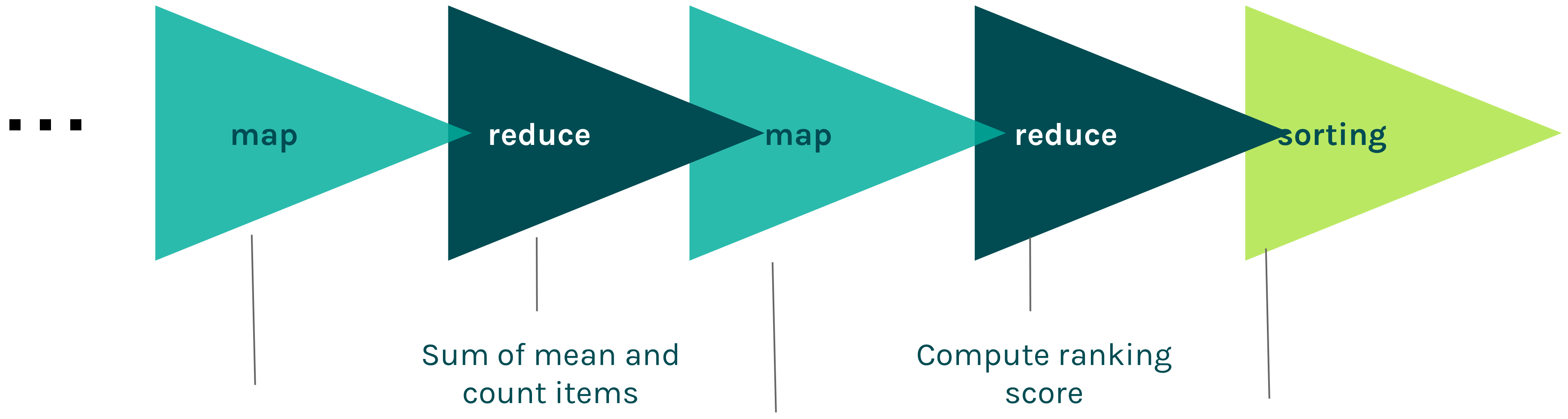
Key:
[house, household, plug, rate, hour, day, month]
Value: *MaxMinHolder* init by work]

Compute max and min value per hour

Key:
[house, houseold, plug,rate, day]
Value: [consumption, 1]

Sum of consumption and count items

Query 3



Key:
[house, household, plug, rate]
Value: [mean, 1]

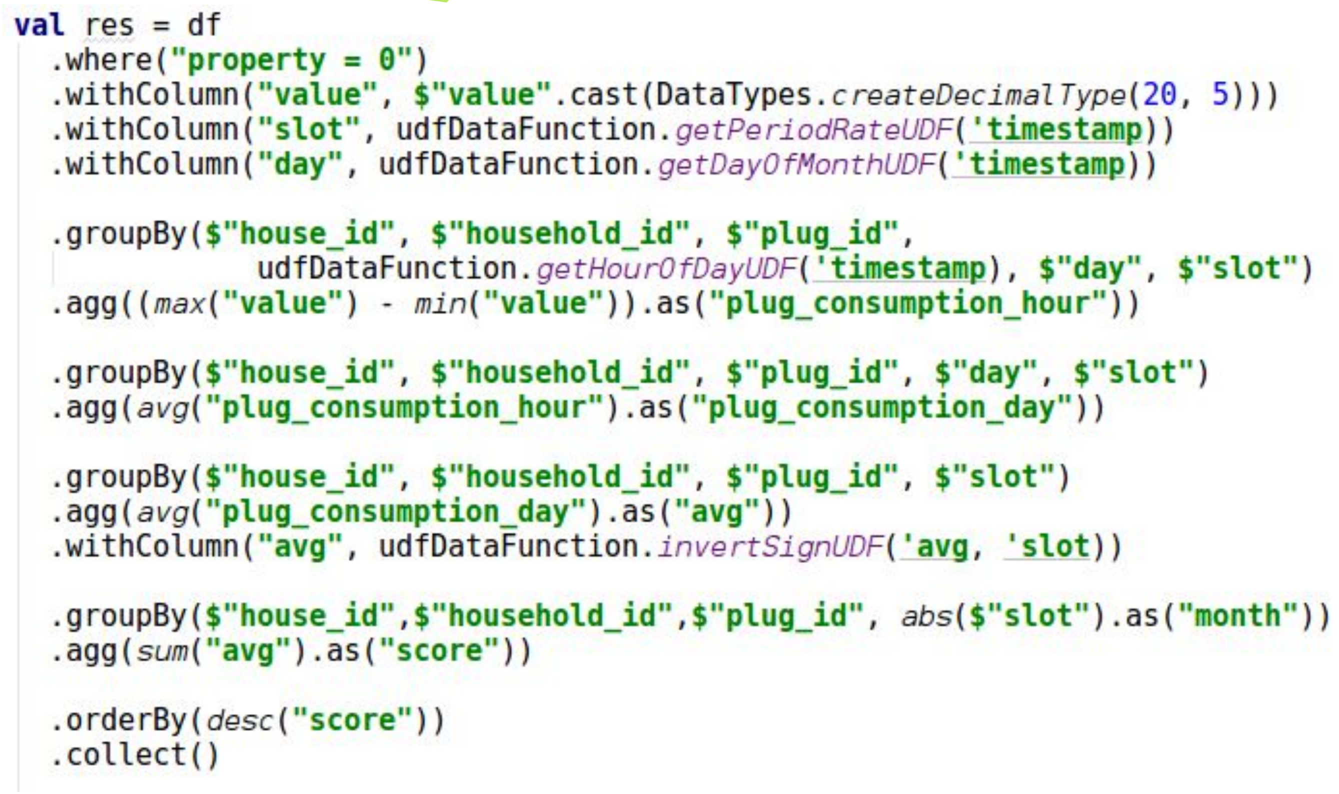
Sum of mean and count items

Key:
[house, household, plug, |rate|]
Value: [±mean]

Compute ranking score

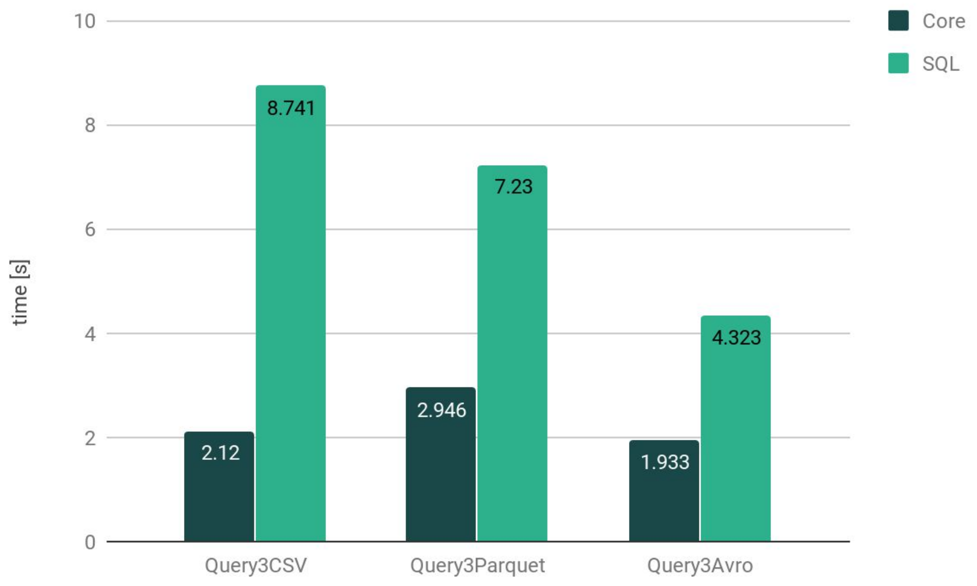
Sort by score

Query 3 Spark SQL



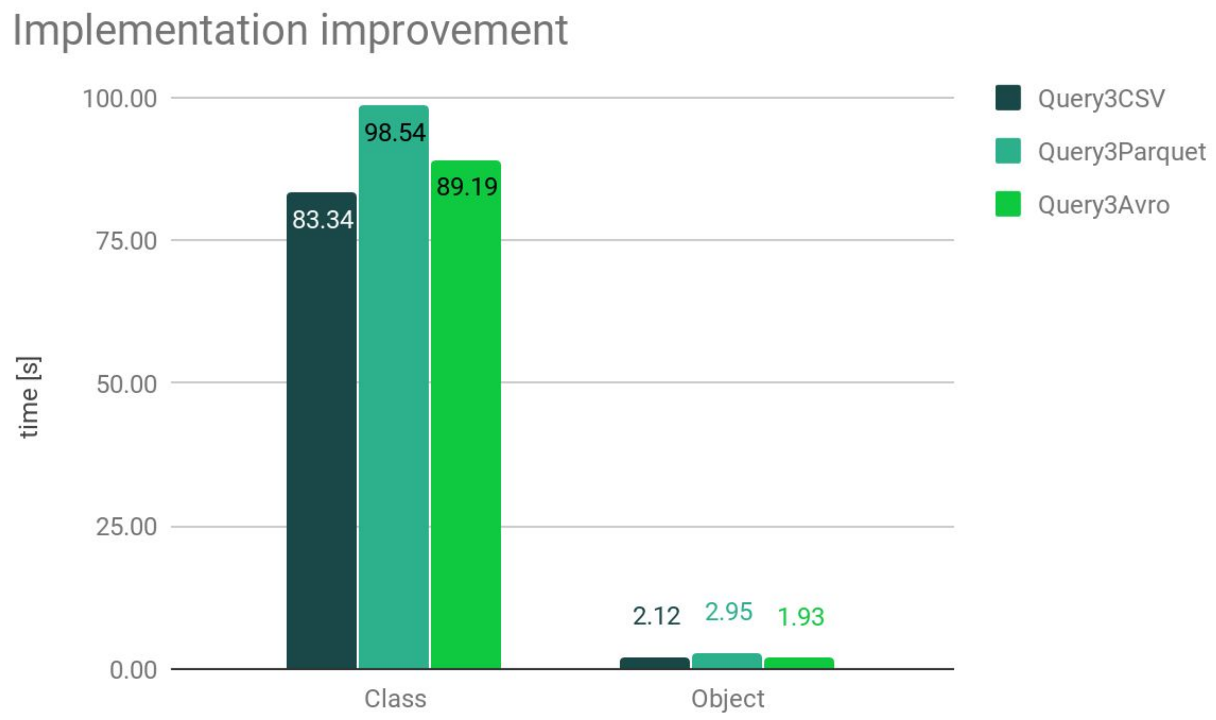
```
val res = df
    .where("property = 0")
    .withColumn("value", $"value".cast(DataTypes.createDecimalType(20, 5)))
    .withColumn("slot", udfDataFunction.getPeriodRateUDF('timestamp))
    .withColumn("day", udfDataFunction.getDayOfMonthUDF('timestamp))

    .groupBy($"house_id", $"household_id", $"plug_id",
                udfDataFunction.getHourOfDayUDF('timestamp), $"day", $"slot")
    .agg((max("value") - min("value")).as("plug_consumption_hour"))

    .groupBy($"house_id", $"household_id", $"plug_id", $"day", $"slot")
    .agg(avg("plug_consumption_hour").as("plug_consumption_day"))

    .groupBy($"house_id", $"household_id", $"plug_id", $"slot")
    .agg(avg("plug_consumption_day").as("avg"))
    .withColumn("avg", udfDataFunction.invertSignUDF('avg, 'slot))

    .groupBy($"house_id",$"household_id",$"plug_id", abs($"slot").as("month"))
    .agg(sum("avg").as("score"))

    .orderBy(desc("score"))
    .collect()
```

Query 3 performances



Spark Core and Spark SQL implementations performances.

Query 3 improvement



3.

Results

Queries output on given dataset

Query 1 results

House ID	0	1	2	3	4	5	6	7	8	9

Query 2 (partial) results

House ID	0	0	0	0	1	1	1	1	2	2	2	2
Time slot	00:00-5:59	6:00-11:59	12:00-17:59	18:00-23:59	00:00-5:59	6:00-11:59	12:00-17:59	18:00-23:59	00:00-5:59	6:00-11:59	12:00-17:59	18:00-23:59
Mean	0.14	0.41	0.19	0.25	0.37	0.69	0.79	0.68	0.30	0.30	0.31	0.33
Standard deviation	0.11	1.04	0.14	0.14	0.36	0.48	0.49	0.40	0.31	0.32	0.30	0.33

...

Query 3 (partial) results

Ranking	Plug	Month	Score
1°	(8,0,1)	9	0.115
2°	(8,0,0)	9	0.080
3°	(0,0,2)	9	0.077
4°	(1,0,1)	9	0.042
5°	(5,0,1)	9	0.011
6°	(7,0,1)	9	0.008

...

4.

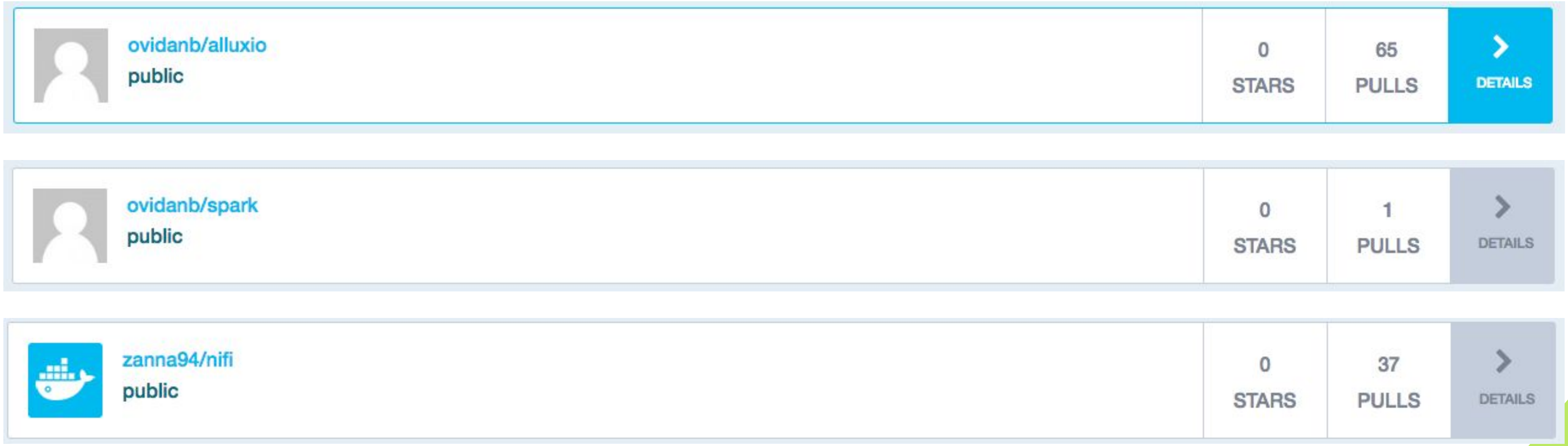
Deployment

Local and Cloud deployment

Local Deploy

- Every system component is built as a Docker Image
- Made from scratch (Spark, NiFi, Alluxio) or from existing images (HDFS, MongoDB)
- Some run in pseudo-distributed mode (HDFS, Spark and Alluxio) with 1 Master and N Workers
- All components run on same network

Docker Images



ovidanb/alluxio	public	0 STARS	65 PULLS	DETAILS
ovidanb/spark	public	0 STARS	1 PULLS	DETAILS
zanna94/nifi	public	0 STARS	37 PULLS	DETAILS

Cloud Deploy

Google Cloud Platform

Google Cloud Platform was used as the deploy environment.

In particular the DataProc and Kubernetes Engine services.

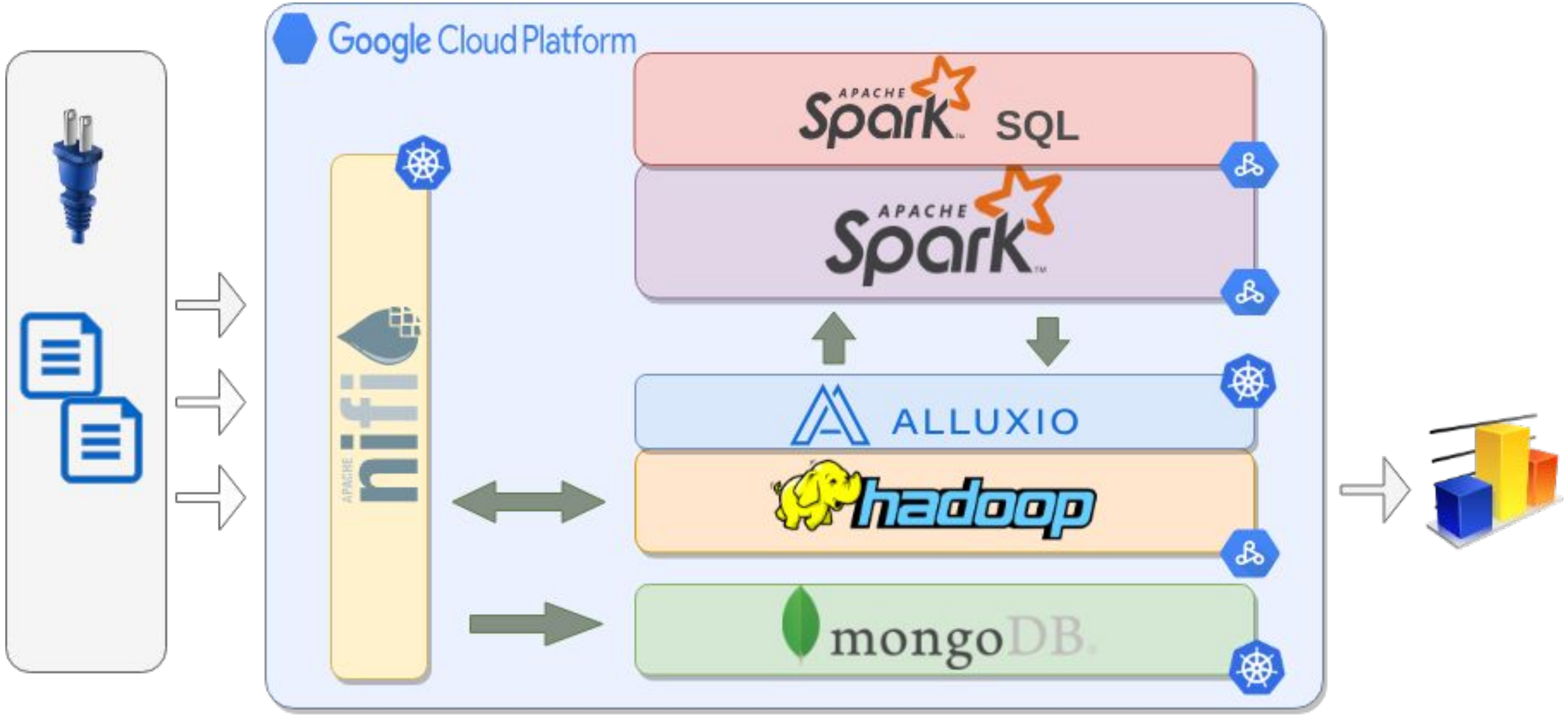
DataProc

- Fully-managed cloud service for running Spark and Hadoop Clusters
- Ready with Hadoop Ecosystem frameworks (HDFS, Hive, Pig and YARN)
- 3 or 5 node cluster configurations (1 Master only)

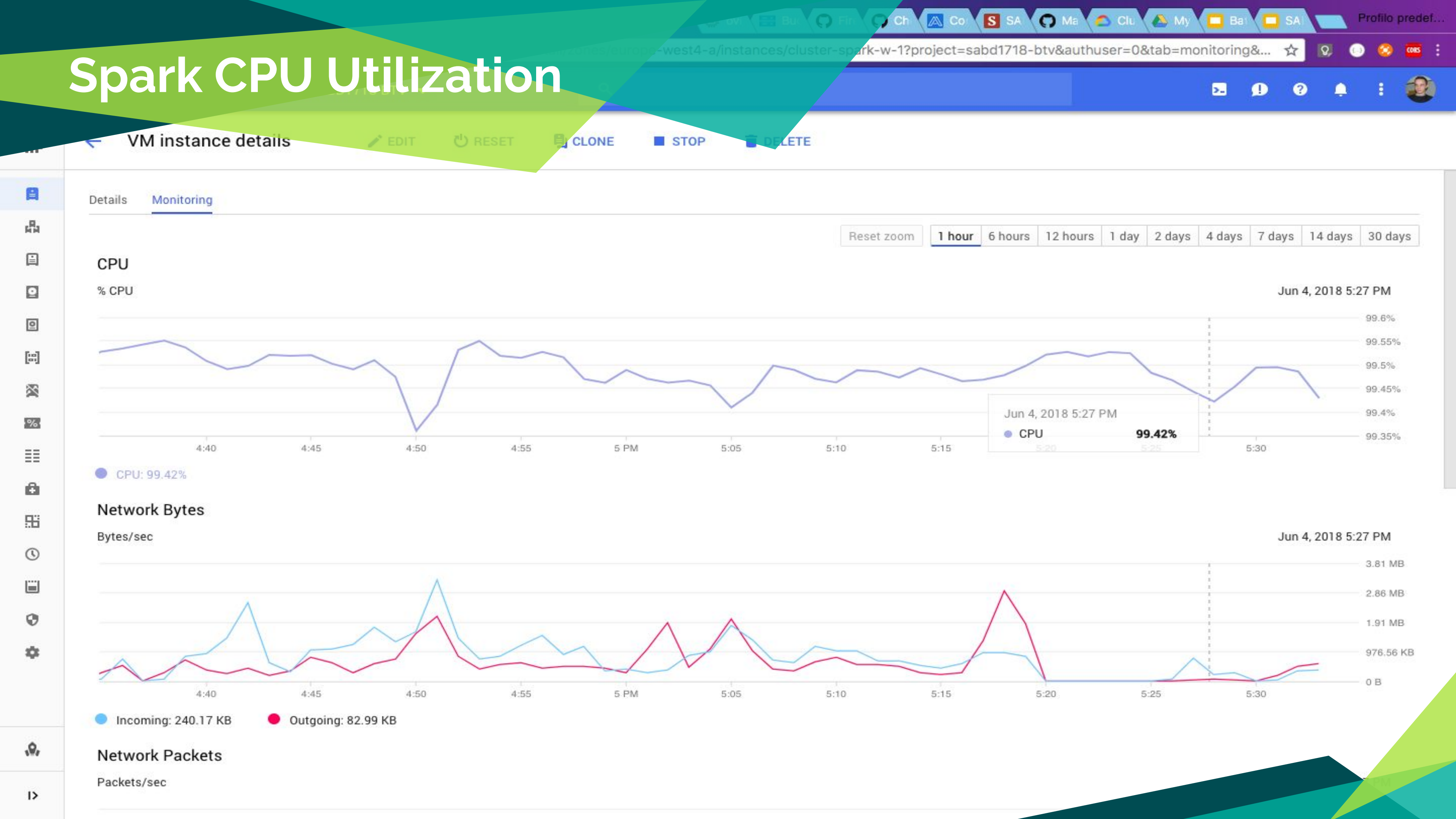
Kubernetes

- Used to deploy remaining System components (MongoDB, NiFi and Alluxio) as Docker containers
- 3-node Cluster (1 Master and 2 Slaves)
- On the same VPC network as DataProc cluster

Cloud Deployment



Spark CPU Utilization


VM instance details

EDIT RESET CLONE STOP DELETE

Details Monitoring

Reset zoom 1 hour 6 hours 12 hours 1 day 2 days 4 days 7 days 14 days 30 days

CPU

% CPU Jun 4, 2018 5:27 PM

Jun 4, 2018 5:27 PM
CPU 99.42%

● CPU: 99.42%

Network Bytes

Bytes/sec Jun 4, 2018 5:27 PM

● Incoming: 240.17 KB ● Outgoing: 82.99 KB

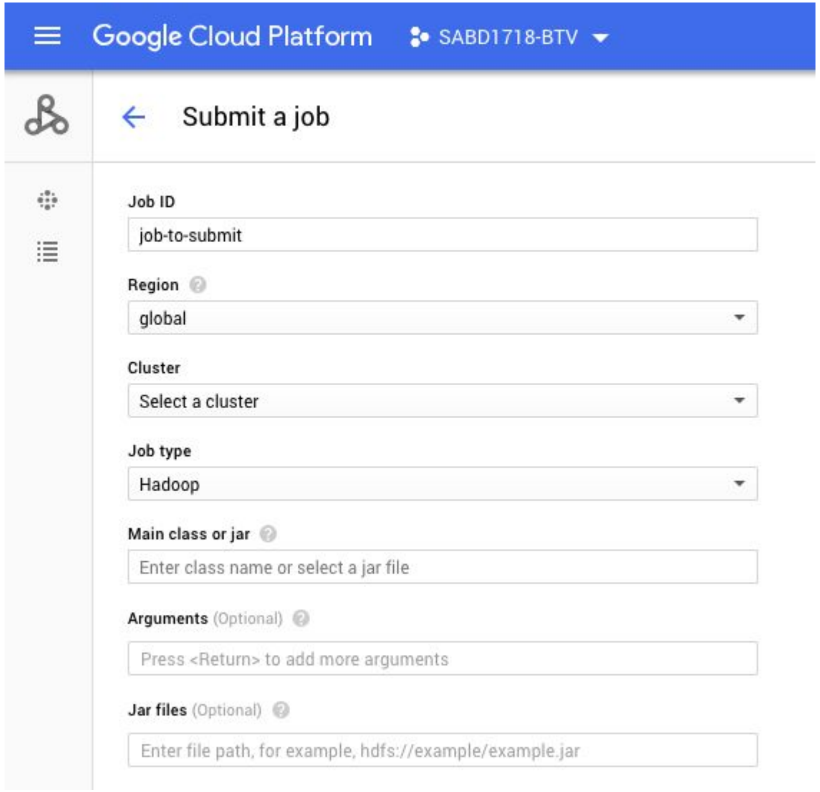
Network Packets

Packets/sec

DataProc Submit Job

Job Type:

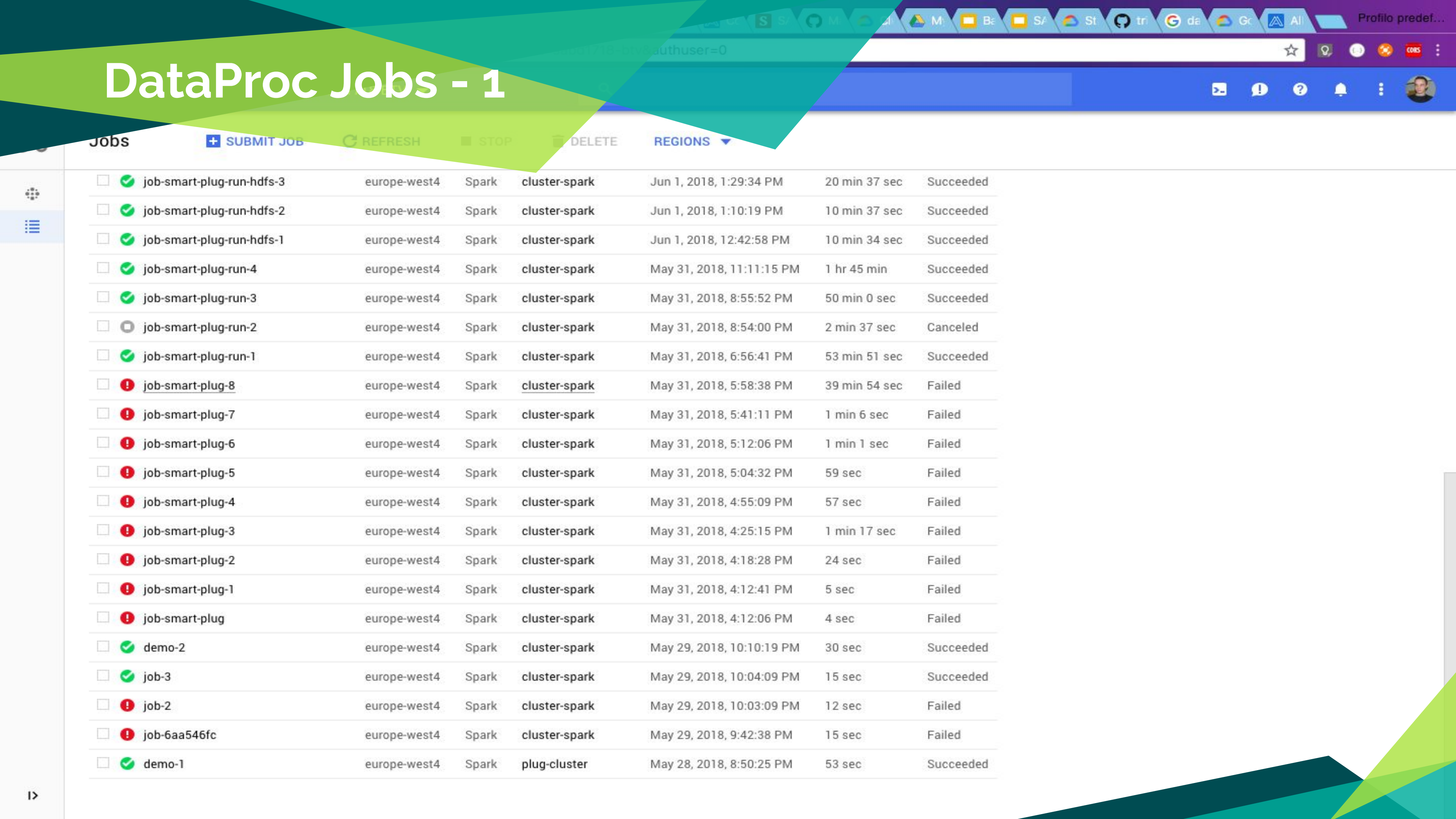
- Hadoop
- Spark
- PySpark
- Hive
- SparkSQL
- Pig



Submit a job

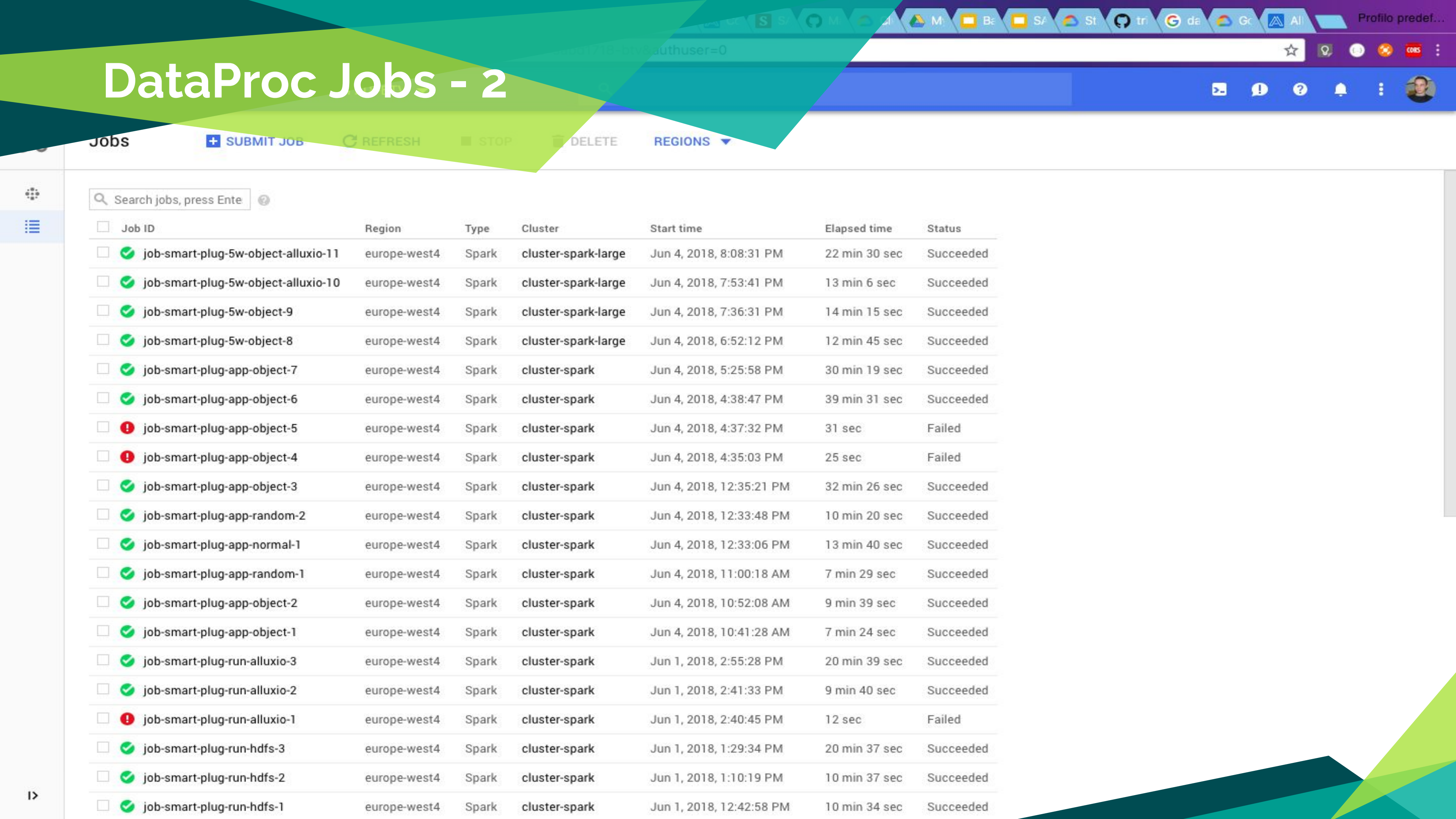
Job ID	job-to-submit
Region	global
Cluster	Select a cluster
Job type	Hadoop
Main class or jar	Enter class name or select a jar file
Arguments (Optional)	Press <Return> to add more arguments
Jar files (Optional)	Enter file path, for example, hdfs://example/example.jar

DataProc Jobs - 1



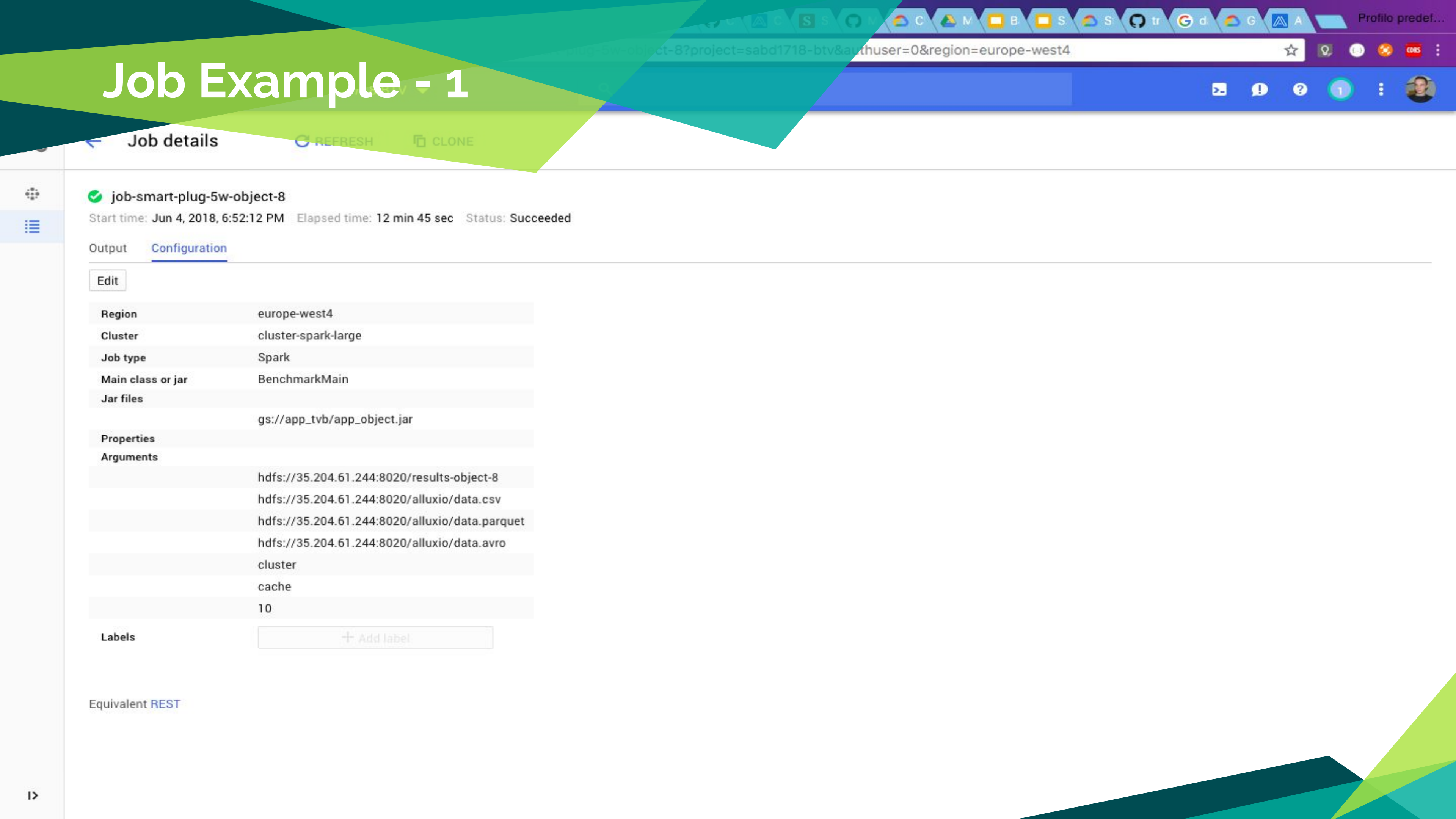
✓	job-smart-plug-run-hdfs-3	europe-west4	Spark	cluster-spark	Jun 1, 2018, 1:29:34 PM	20 min 37 sec	Succeeded
✓	job-smart-plug-run-hdfs-2	europe-west4	Spark	cluster-spark	Jun 1, 2018, 1:10:19 PM	10 min 37 sec	Succeeded
✓	job-smart-plug-run-hdfs-1	europe-west4	Spark	cluster-spark	Jun 1, 2018, 12:42:58 PM	10 min 34 sec	Succeeded
✓	job-smart-plug-run-4	europe-west4	Spark	cluster-spark	May 31, 2018, 11:11:15 PM	1 hr 45 min	Succeeded
✓	job-smart-plug-run-3	europe-west4	Spark	cluster-spark	May 31, 2018, 8:55:52 PM	50 min 0 sec	Succeeded
■	job-smart-plug-run-2	europe-west4	Spark	cluster-spark	May 31, 2018, 8:54:00 PM	2 min 37 sec	Canceled
✓	job-smart-plug-run-1	europe-west4	Spark	cluster-spark	May 31, 2018, 6:56:41 PM	53 min 51 sec	Succeeded
!	job-smart-plug-8	europe-west4	Spark	cluster-spark	May 31, 2018, 5:58:38 PM	39 min 54 sec	Failed
!	job-smart-plug-7	europe-west4	Spark	cluster-spark	May 31, 2018, 5:41:11 PM	1 min 6 sec	Failed
!	job-smart-plug-6	europe-west4	Spark	cluster-spark	May 31, 2018, 5:12:06 PM	1 min 1 sec	Failed
!	job-smart-plug-5	europe-west4	Spark	cluster-spark	May 31, 2018, 5:04:32 PM	59 sec	Failed
!	job-smart-plug-4	europe-west4	Spark	cluster-spark	May 31, 2018, 4:55:09 PM	57 sec	Failed
!	job-smart-plug-3	europe-west4	Spark	cluster-spark	May 31, 2018, 4:25:15 PM	1 min 17 sec	Failed
!	job-smart-plug-2	europe-west4	Spark	cluster-spark	May 31, 2018, 4:18:28 PM	24 sec	Failed
!	job-smart-plug-1	europe-west4	Spark	cluster-spark	May 31, 2018, 4:12:41 PM	5 sec	Failed
!	job-smart-plug	europe-west4	Spark	cluster-spark	May 31, 2018, 4:12:06 PM	4 sec	Failed
✓	demo-2	europe-west4	Spark	cluster-spark	May 29, 2018, 10:10:19 PM	30 sec	Succeeded
✓	job-3	europe-west4	Spark	cluster-spark	May 29, 2018, 10:04:09 PM	15 sec	Succeeded
!	job-2	europe-west4	Spark	cluster-spark	May 29, 2018, 10:03:09 PM	12 sec	Failed
!	job-6aa546fc	europe-west4	Spark	cluster-spark	May 29, 2018, 9:42:38 PM	15 sec	Failed
✓	demo-1	europe-west4	Spark	plug-cluster	May 28, 2018, 8:50:25 PM	53 sec	Succeeded

DataProc Jobs - 2



	Job ID	Region	Type	Cluster	Start time	Elapsed time	Status
✓	job-smart-plug-5w-object-alluxio-11	europe-west4	Spark	cluster-spark-large	Jun 4, 2018, 8:08:31 PM	22 min 30 sec	Succeeded
✓	job-smart-plug-5w-object-alluxio-10	europe-west4	Spark	cluster-spark-large	Jun 4, 2018, 7:53:41 PM	13 min 6 sec	Succeeded
✓	job-smart-plug-5w-object-9	europe-west4	Spark	cluster-spark-large	Jun 4, 2018, 7:36:31 PM	14 min 15 sec	Succeeded
✓	job-smart-plug-5w-object-8	europe-west4	Spark	cluster-spark-large	Jun 4, 2018, 6:52:12 PM	12 min 45 sec	Succeeded
✓	job-smart-plug-app-object-7	europe-west4	Spark	cluster-spark	Jun 4, 2018, 5:25:58 PM	30 min 19 sec	Succeeded
✓	job-smart-plug-app-object-6	europe-west4	Spark	cluster-spark	Jun 4, 2018, 4:38:47 PM	39 min 31 sec	Succeeded
!	job-smart-plug-app-object-5	europe-west4	Spark	cluster-spark	Jun 4, 2018, 4:37:32 PM	31 sec	Failed
!	job-smart-plug-app-object-4	europe-west4	Spark	cluster-spark	Jun 4, 2018, 4:35:03 PM	25 sec	Failed
✓	job-smart-plug-app-object-3	europe-west4	Spark	cluster-spark	Jun 4, 2018, 12:35:21 PM	32 min 26 sec	Succeeded
✓	job-smart-plug-app-random-2	europe-west4	Spark	cluster-spark	Jun 4, 2018, 12:33:48 PM	10 min 20 sec	Succeeded
✓	job-smart-plug-app-normal-1	europe-west4	Spark	cluster-spark	Jun 4, 2018, 12:33:06 PM	13 min 40 sec	Succeeded
✓	job-smart-plug-app-random-1	europe-west4	Spark	cluster-spark	Jun 4, 2018, 11:00:18 AM	7 min 29 sec	Succeeded
✓	job-smart-plug-app-object-2	europe-west4	Spark	cluster-spark	Jun 4, 2018, 10:52:08 AM	9 min 39 sec	Succeeded
✓	job-smart-plug-app-object-1	europe-west4	Spark	cluster-spark	Jun 4, 2018, 10:41:28 AM	7 min 24 sec	Succeeded
✓	job-smart-plug-run-alluxio-3	europe-west4	Spark	cluster-spark	Jun 1, 2018, 2:55:28 PM	20 min 39 sec	Succeeded
✓	job-smart-plug-run-alluxio-2	europe-west4	Spark	cluster-spark	Jun 1, 2018, 2:41:33 PM	9 min 40 sec	Succeeded
!	job-smart-plug-run-alluxio-1	europe-west4	Spark	cluster-spark	Jun 1, 2018, 2:40:45 PM	12 sec	Failed
✓	job-smart-plug-run-hdfs-3	europe-west4	Spark	cluster-spark	Jun 1, 2018, 1:29:34 PM	20 min 37 sec	Succeeded
✓	job-smart-plug-run-hdfs-2	europe-west4	Spark	cluster-spark	Jun 1, 2018, 1:10:19 PM	10 min 37 sec	Succeeded
✓	job-smart-plug-run-hdfs-1	europe-west4	Spark	cluster-spark	Jun 1, 2018, 12:42:58 PM	10 min 34 sec	Succeeded

Job Example - 1



Job details

REFRESH CLONE

job-smart-plug-5w-object-8

Start time: Jun 4, 2018, 6:52:12 PM Elapsed time: 12 min 45 sec Status: Succeeded

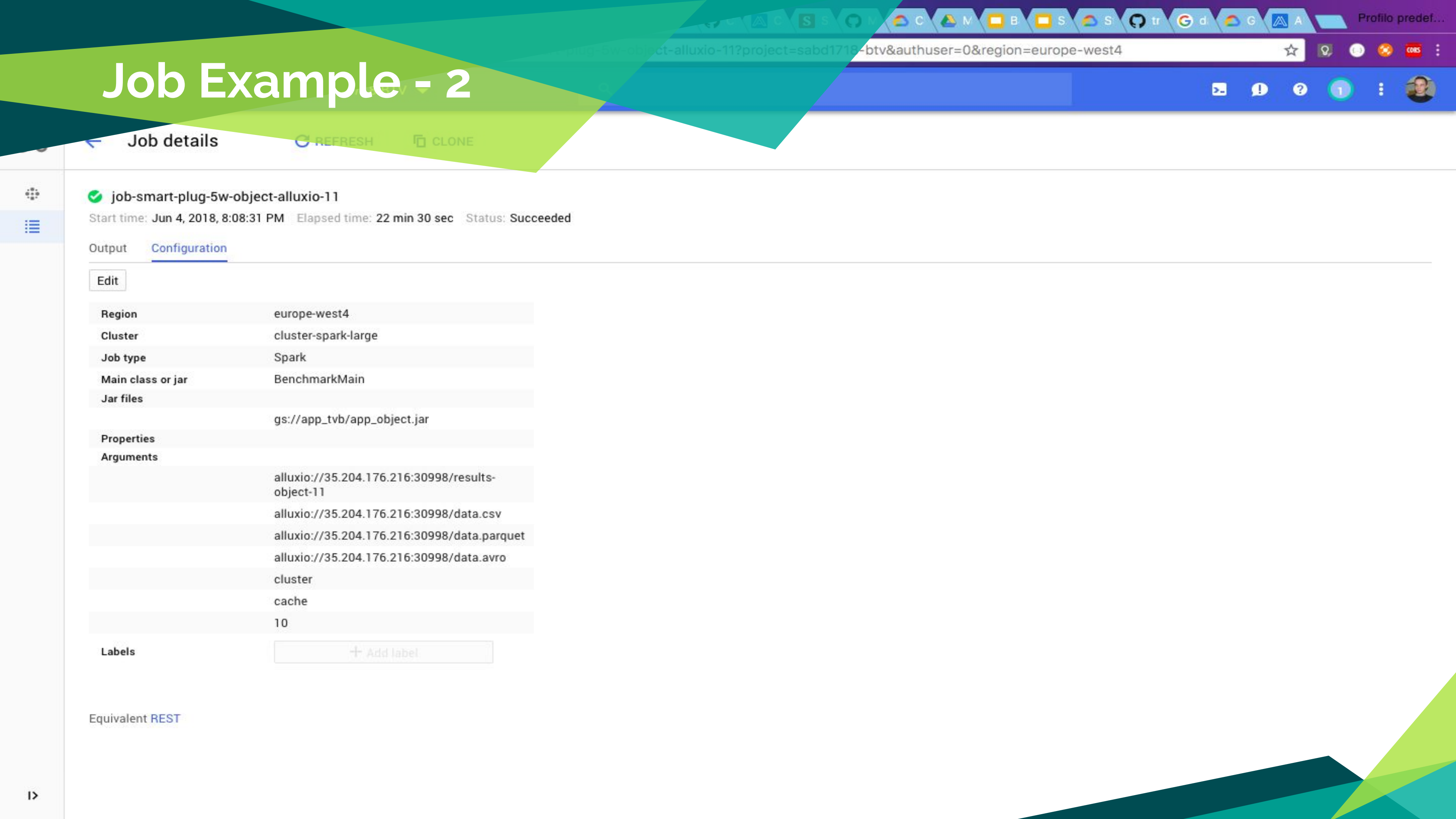
Output Configuration

Edit

Region	europe-west4
Cluster	cluster-spark-large
Job type	Spark
Main class or jar	BenchmarkMain
Jar files	
	gs://app_tvb/app_object.jar
Properties	
Arguments	
	hdfs://35.204.61.244:8020/results-object-8
	hdfs://35.204.61.244:8020/alluxio/data.csv
	hdfs://35.204.61.244:8020/alluxio/data.parquet
	hdfs://35.204.61.244:8020/alluxio/data.avro
	cluster
	cache
	10
Labels	+ Add label

Equivalent REST

Job Example - 2



job-smart-plug-5w-object-alluxio-11

Start time: Jun 4, 2018, 8:08:31 PM Elapsed time: 22 min 30 sec Status: Succeeded

Region	europe-west4
Cluster	cluster-spark-large
Job type	Spark
Main class or jar	BenchmarkMain
Jar files	
	gs://app_tvb/app_object.jar
Properties	
Arguments	
	alluxio://35.204.176.216:30998/results-object-11
	alluxio://35.204.176.216:30998/data.csv
	alluxio://35.204.176.216:30998/data.parquet
	alluxio://35.204.176.216:30998/data.avro
	cluster
	cache
	10
Labels	

Equivalent REST

5.

Queries Performance

Execution time with different formats on different architectures

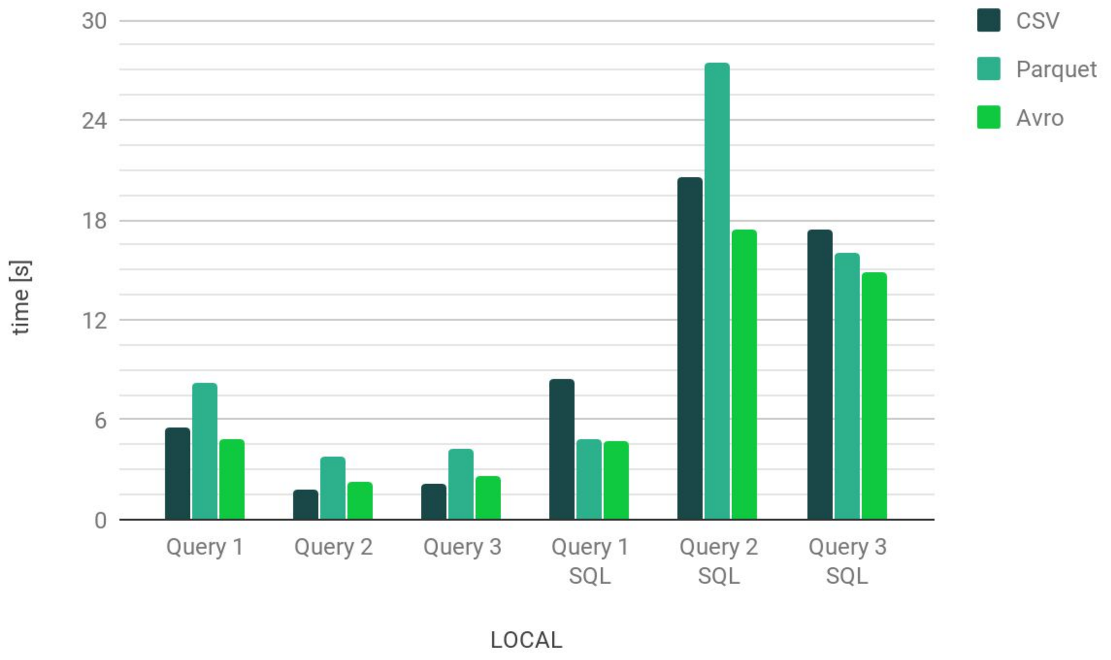
Local Deployment Performance

Specs :

- Macbook Air 2014
- 4 GB RAM
- 1,4 GHz Intel Core i5

Docker (same network):

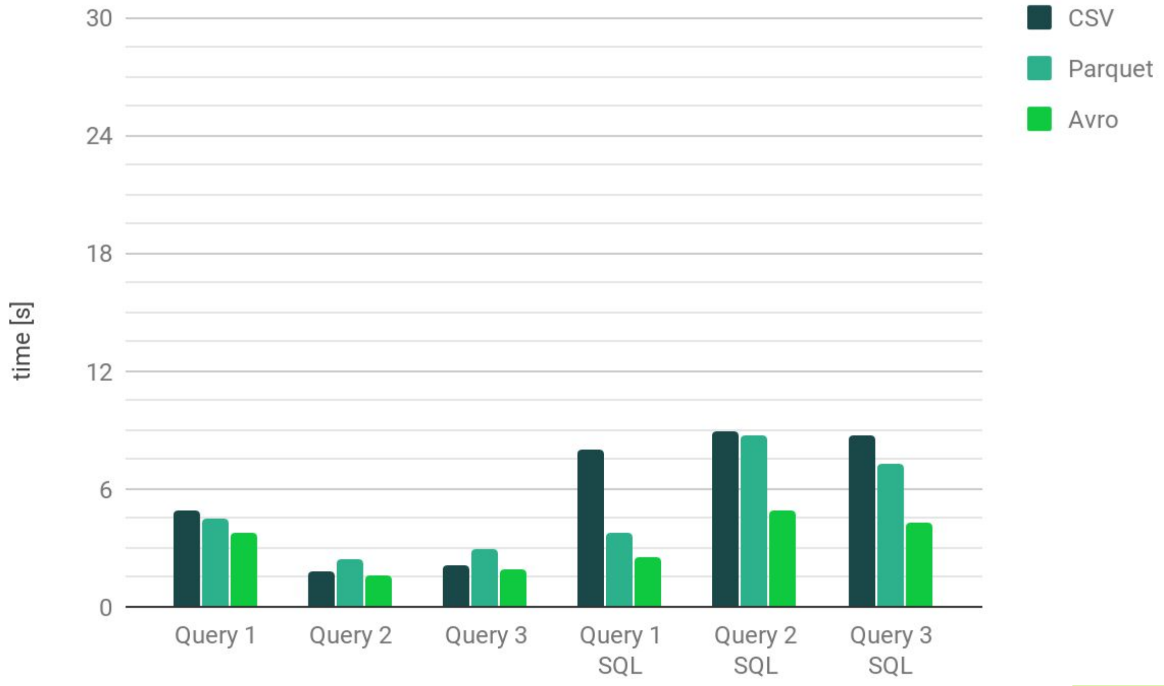
- 4 HDFS nodes
- 3 Spark nodes
- 2 Alluxio nodes



Cloud deployment using HDFS

DataProc :

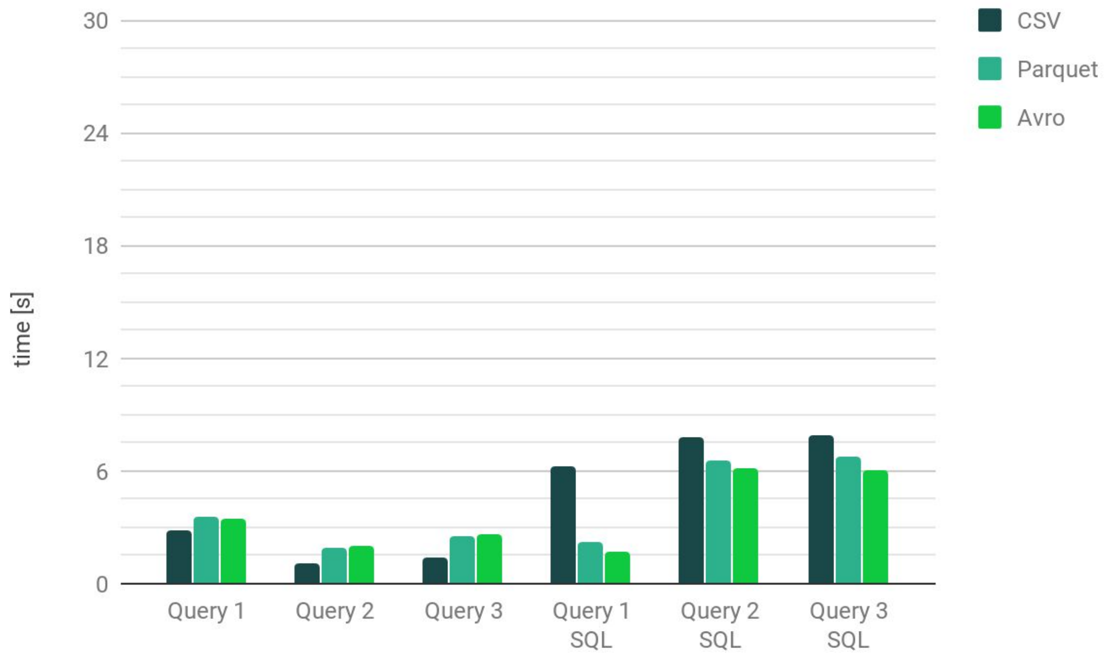
- 5 worker node cluster
- 1 master node
- node: n1-standard-2 (2 vCPUs with 7.5 GB RAM)



Cloud deployment using HDFS and cache

DataProc :

- 5 worker node cluster
- 1 master node
- node: n1-standard-2 (2 vCPUs with 7.5 GB RAM)



Cloud deployment using Alluxio

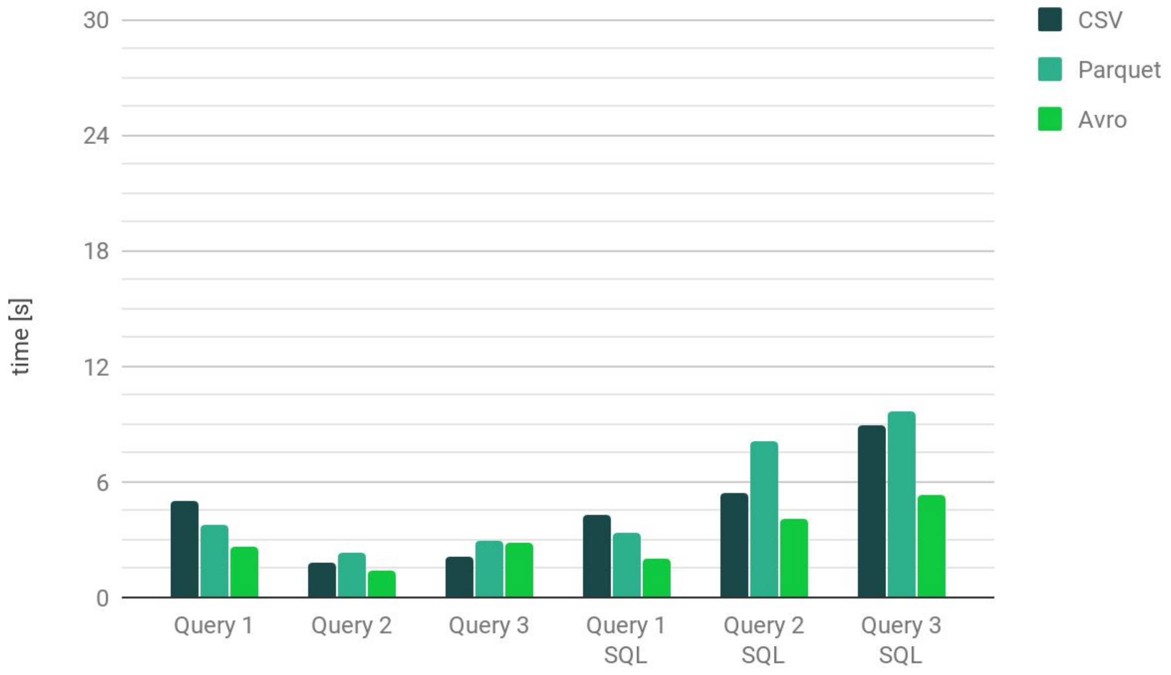
DataProc :

- 5 worker node cluster & 1 master
- node: n1-standard-2 (2 vCPUs with 7.5 GB RAM)

Kubernetes:

- 3 node n1-standard-2 cluster (each with Alluxio agent)

Both clusters on same VPC network



Best Performance

- Avro as dataset format
- Alluxio (on top of HDFS) as distributed file system
- No Spark caching

Best Performance

	Core	SQL
Query 1	2.37 s	2,01 s
Query 2	1.39 s	4.12 s
Query 3	2.84 s	5.31 s

Thank you for listening!