



# SISTEMI E ARCHITETTURE PER BIG DATA

Progetto 1

Analisi della diffusione di COVID-19

Marco Balletti

Francesco Marino



# Sommario

## Architettura:

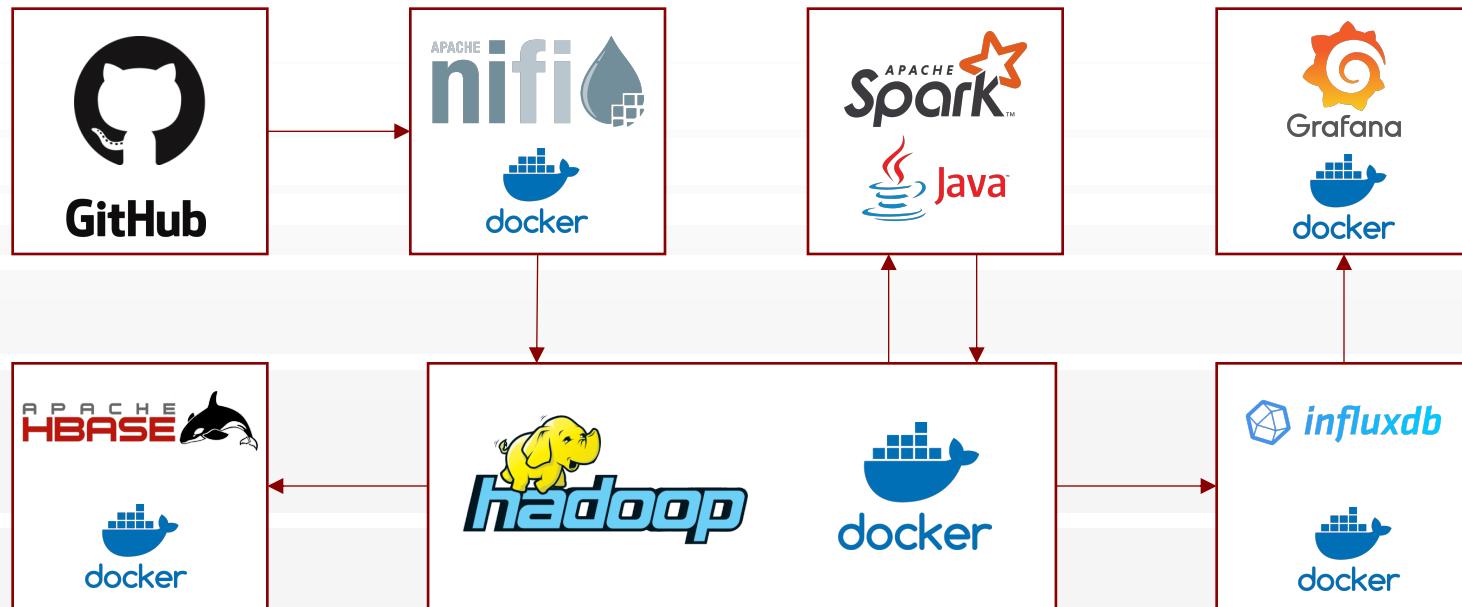
- NiFi
- HDFS
- Spark
- Hbase
- InfluxDB
- Grafana

## Queries:

- Query 1
- Query 2
- Query 3

## Benchmark

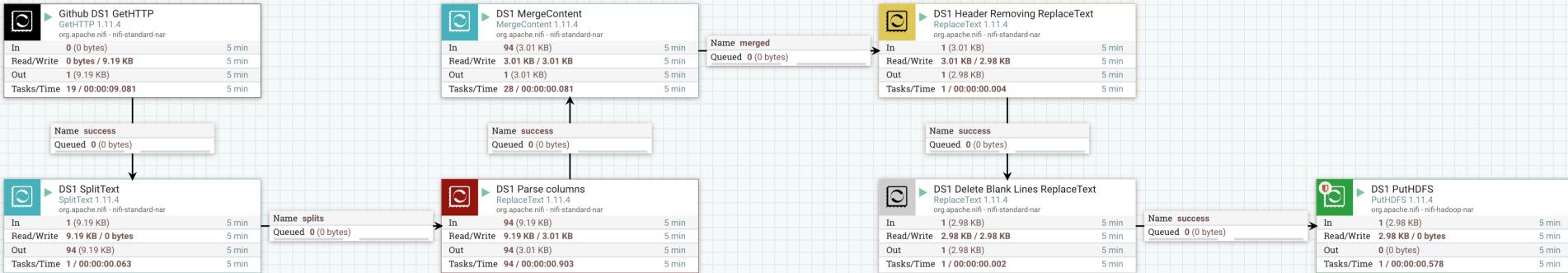
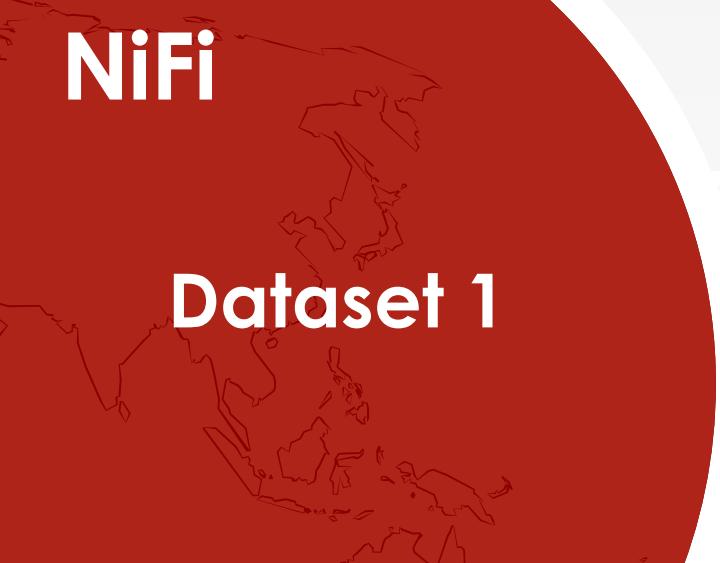
# Architettura



# I Container

<b>NiFi</b>	<b>HDFS</b>	<b>HBase</b>	<b>InfluxDB</b>	<b>Grafana</b>
<b>Immagine:</b> apache/nifi	<b>Immagine:</b> effeerre/hadoop	<b>Immagine:</b> harisekhon/hbase	<b>Immagine:</b> influxdb	<b>Immagine:</b> grafana/grafana
<b>Funzione:</b> Preprocessing ed Ingestion dei dati nell'HDFS	<b>Funzione:</b> Layer principale per lo storage di input ed output	<b>Funzione:</b> Datastore NoSQL per la memorizzazione dei risultati	<b>Funzione:</b> Datastore orientato a serie temporali per l'esportazione di dati da HDFS verso Grafana	<b>Funzione:</b> Rappresentazione grafica dei risultati ottenuti dal processamento
<b>Modalità di esecuzione:</b> Singolo container	<b>Configurazione:</b> Grado di replicazione pari a 2  <b>Modalità di esecuzione:</b> Molteplici container: <ul style="list-style-type: none"><li>• 1 master</li><li>• 3 worker</li></ul>	<b>Configurazione:</b> Memorizzazione di una singola versione del dato  <b>Modalità di esecuzione:</b> Singolo container	<b>Configurazione:</b> Retention time pari a 365 giorni  <b>Modalità di esecuzione:</b> Singolo container	<b>Modalità di esecuzione:</b> Singolo container

# Dataset 1

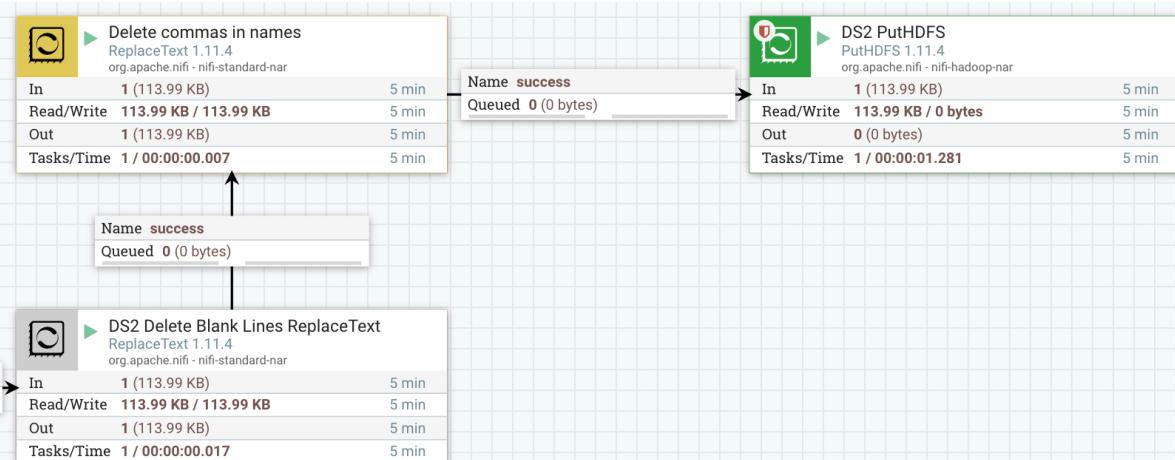


- Download del primo dataset aggiornato da GitHub [1]
- Split del .csv per righe
- Rimozione dei campi non necessari
- Ricomposizione del dataset in un unico .csv
- Rimozione dell'header
- Rimozione di eventuali righe vuote
- Scrittura del dataset pre-processato su HDFS

# Dataset 2



- Download del secondo dataset aggiornato da GitHub [2]
- Rimozione dell'header
- Rimozione di eventuali righe vuote
- Rimozione di virgole dai nomi
- Scrittura del dataset pre-processato su HDFS





## Query 1

- Import del dataset da HDFS (.textFile())
- Split dei dati e generazione delle tuple **[data, totale guariti, totale tamponi]** (.mapToPair())
- Conversione della chiave e propagazione dei dati alla settimana successiva (per la successiva conversione in valori puntuali) generando tuple **[data inizio settimana, totale guariti giornaliero, totale tamponi giornaliero]** (.flatMapToPair())
- Raggruppamento dei dati per settimana (.groupByKey())

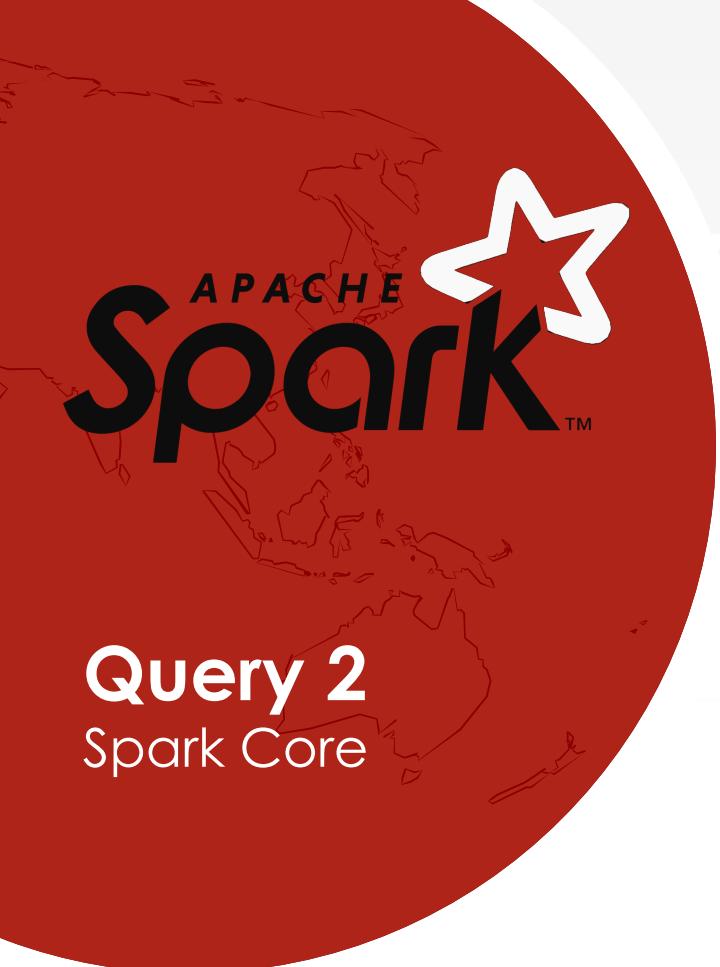
### Spark Core

- Conversione dei dati in valori puntuali e calcolo delle statistiche (.flatMapToPair())
- Export del risultato su HDFS ordinato temporalmente (.sortByKey().saveAsTextFile())

### Spark SQL

- Conversione dei dati in valori puntuali (.flatMapToPair())
- Creazione dello schema
- Esecuzione della query:

```
SELECT week, avg(cured),  
       avg(swabs)  
  FROM query1  
 GROUP BY week  
 ORDER BY week
```



Pre-processing

- Import del dataset da HDFS (`.textFile()`)
- Split dei dati, conversione a misurazioni puntuali, calcolo dei trendline coefficient e generazione delle tuple **[slope, coordinate geografiche, misurazioni giornaliere]** (`.mapToPair()`)
- Ordinamento decrescente rispetto al valore dello slope e riduzione del dataset ai primi 100 elementi (`.sortByKey().zipWithIndex().filter().keys()`)

Spark Core

- Identificazione del continente e suddivisione su base settimanale generando tuple **[continente e inizio settimana, lista dei valori per la settimana]** (`.flatMapToPair()`)
- Somma dei valori giorno per giorno per ogni continente (`.reduceByKey()`)
- Calcolo delle statistiche generando tuple **[continente e inizio settimana, media, deviazione standard, minimo, massimo]** (`.mapToPair()`)
- Export del risultato su HDFS ordinato temporalmente (`.sortByKey().saveAsTextFile()`)

...

## Spark SQL

- Identificazione del continente e suddivisione su base giornaliera generando tuple del tipo **[continente, settimana dell'anno, giorno della settimana, valore]** (.flatMapToPair())
- Creazione dello schema
- Aggregazione dei valori in base a giorno e continente:

```
SELECT continent, week,  
sum(positive) AS positive  
FROM query2  
GROUP BY continent, week, day
```

- Calcolo delle statistiche:

```
SELECT continent, week, mean(positive), stddev(positive),  
min(positive), max(positive)  
FROM query2  
GROUP BY continent, week  
ORDER BY continent, week
```



## Query 2

### Spark SQL

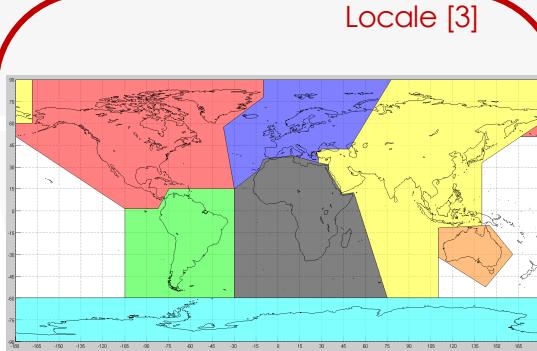


## Query 2

### Reverse Geocoding

Ottenere il continente a partire da: Latitudine-Longitudine

Obiettivo



- Utilizzo di fence (o poligoni) rappresentanti continenti
- Localizzazione della coppia di coordinate all'interno di uno dei poligoni in modo efficiente
- In caso di incertezza si passa all'utilizzo di REST API

REST API

- Servizi di reverse geocoding offerti in rete
- Nominatim [4]:
  - API gratuita
  - Prestazioni peggiori in termini di latenza e throughput
    - Tempo impiegato: 262936 ms
    - Richieste: 260
- Big Data Cloud [5]:
  - API gratuita
  - È richiesta una chiave per l'utilizzo
  - 50'000 richieste mensili
  - Prestazioni migliori in termini di latenza e throughput
    - Tempo impiegato: 21619 ms
    - Richieste: 260

- Viene preferita la risoluzione in locale ove possibile:

- Tempo impiegato: 213 ms
- Richieste: 260

Metodo Ibrido



## Query 3

### Pre-processing

- Import del dataset da HDFS (`.textFile()`)
- Split, conversione a valori puntuali e suddivisione dei dati su base mensile generando le tuple **[mese, stato/nazione, valori del mese]** (`.flatMapToPair()`)
- Calcolo del trendline coefficient per ogni coppia stato-mese generando le tuple **[mese, slope, stato/nazione, valori del mese]** (`.mapToPair()`)
- Raggruppamento dei dati per mese (`.groupByKey()`)
- Ordinamento decrescente rispetto al valore dello slope e riduzione del dataset ai primi 50 elementi per ogni mese (`.mapToPair()`)
- Ottenimento della lista di mesi (`.keys()`)
- Applicazione dell'algoritmo di clustering mese per mese

```
foreach month:  
    doClustering(RDD.filter(month))  
)
```

MLlib

Naive

## Algoritmo di Loyd [7]:

- Inizializzazione random dei centroidi
- Conversione dei dati nelle tuple **[slope, stato/nazione]** per il mese corrente (.flatMapToPair())
- Esecuzione distribuita del clustering:  
(  
**foreach iteration:**
  - Assegnamento dei punti al centroide più vicino (.mapToPair())
  - Calcolo dei nuovi centroidi (.groupByKey().mapToPair())
  - Aggiornamento dei centroidi (.collect())  
)- Valutazione dell'assegnamento e dei costi

# Query 3

## Clustering

# HBase

## Struttura delle tabelle

### Query 1

Chiave	Statistiche settimanali	
yyyy-MM-dd	Media di guariti	Media di tamponi

### Query 2

Chiave	Statistiche settimanali sui contagiati			
Continente - yyyy-MM-dd	Media	Deviazione standard	Minimo	Massimo

### Query 3

Chiave	Cluster mensili (elenco di paesi)			
MM-yyyy	Cluster 1	Cluster 2	Cluster 3	Cluster 4

# InfluxDB

## Struttura delle misurazioni



### Query 1

Tutti i punti fanno riferimento alla stessa misurazione

Tempo	Media di guariti	Media di tamponi
-------	------------------	------------------

### Query 2

Una misurazione per ogni statistica trovata

Tempo	Continente di riferimento	Statistica relativa alla misurazione
-------	---------------------------	--------------------------------------

### Query 3

Una misurazione per ogni cluster

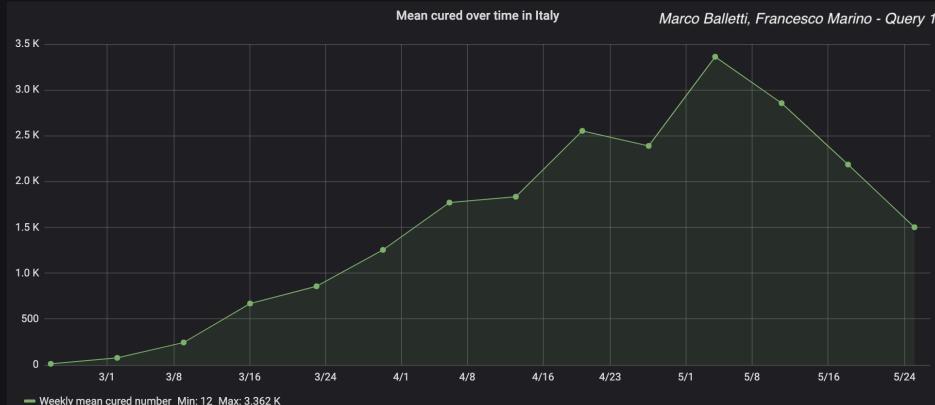
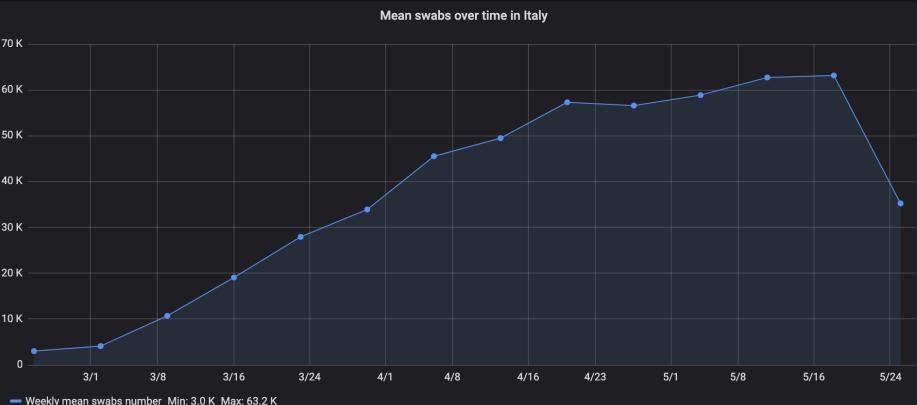
Tempo	Centroide del cluster	Lista di paesi
-------	-----------------------	----------------



## ▶ Presentazione dei risultati

Grafana

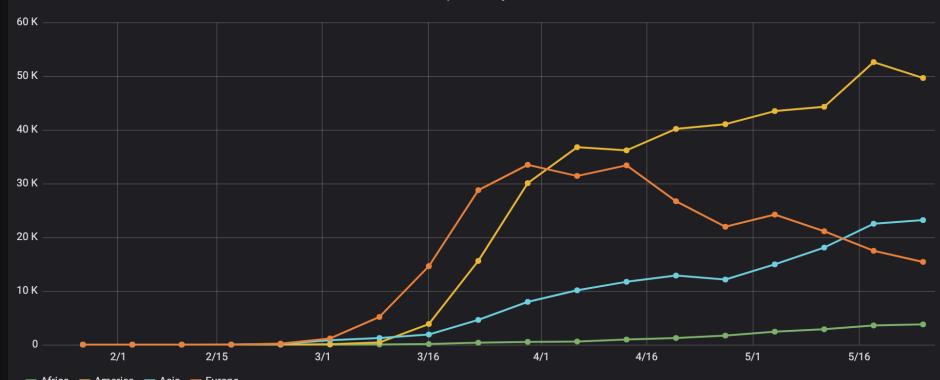
# Query 1



Time	Weekly mean cured number	Weekly mean swabs number
2020-02-24	11.857142857142858	3018.142857142853
2020-03-02	77	4115.714285714285
2020-03-09	244.71428571428572	10708.857142857143
2020-03-16	669.8571428571429	19071.85714285714
2020-03-23	858	27946.85714285714
2020-03-30	1255	33918.71428571428
2020-04-06	1770.857142857143	45533.142857142855
2020-04-13	1834.857142857143	49478.28571428572
2020-04-20	2553.285714285714	57302.57142857143
2020-04-27	2389.4285714285716	56587.57142857143
2020-05-04	3361.714285714286	58877.142857142855
2020-05-11	2855.714285714286	62721.142857142855

# Query 2

Mean of positive by continent

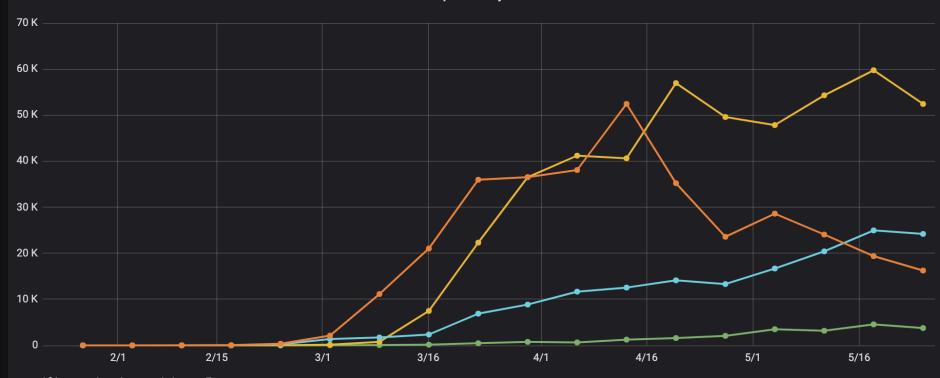


Standard deviation of positive by continent

Marco Balletti, Francesco Marino - Query 2

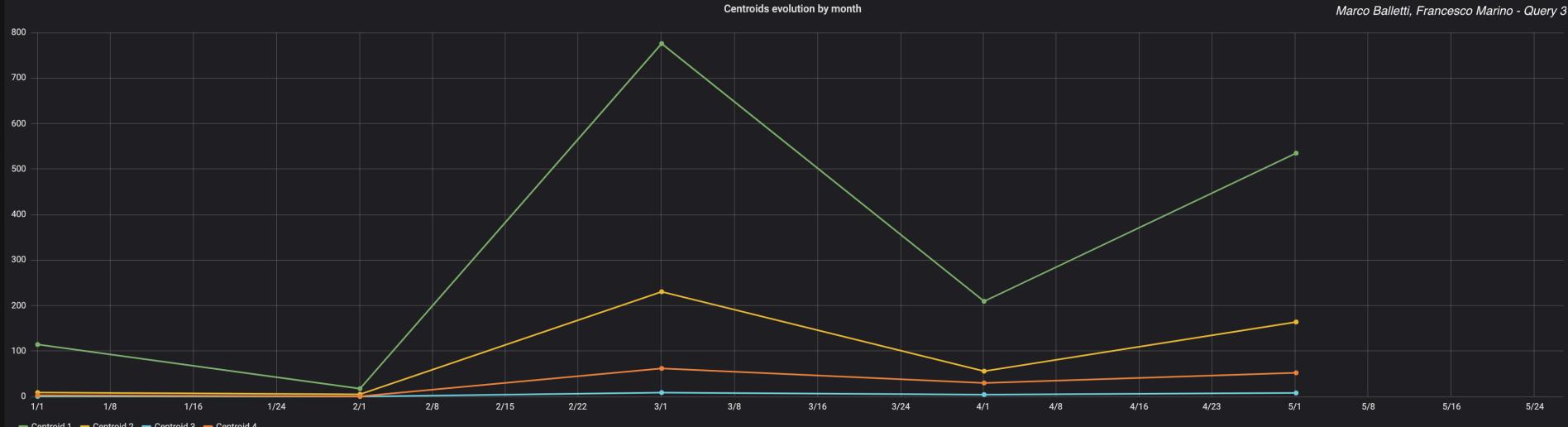


Min of positive by continent



# Query 3 - MLlib

Marco Balletti, Francesco Marino - Query 3



Cluster composition by month

Time	Cluster 1	Cluster 2	Cluster 3	Cluster 4
2020-05	[Brazil]	[India, Chile]	[Kuwait, Argentina, Colombia, Afghanistan, Egypt, United Arab Emirates, Indonesia, Oman, Armenia, Guatemala, Bolivia, Kazakhstan, Tajikistan, Cameroon, Sudan, Djibouti, Bahrain, Iraq, Honduras, Gabon, Azerbaijan, Venezuela, Congo (Kinshasa), Poland, South Sudan, Haiti, Belarus, Nepal, El Salvador, Ethiopia, Central African Republic, Moldova, Equatorial Guinea, Nigeria, Madagascar, Mauritania, Kenya, Algeria]	[Mexico, Peru, Bangladesh, Saudi Arabia, Iran, Qatar, Pakistan, South Africa]
2020-04	[Russia, Brazil]	[Peru, Saudi Arabia, India, Ecuador]	[Ukraine, United Arab Emirates, Chile, South Africa, Alberta, Kuwait, Indonesia, Quebec, Sweden, Egypt, Colombia, Nigeria, Kazakhstan, Ghana, Ontario, Afghanistan, Dominican Republic, Guinea, Oman, Bahrain, United Kingdom, Argentina, Panama, Morocco, Maldives, Senegal, Bolivia, Somalia, Sudan, Bulgaria, Sri Lanka, Armenia, Tanzania, Algérie, Moldova, Serbia, Guatemala]	[Mexico, Singapore, Belarus, Qatar, Pakistan, Bangladesh]
2020-03	[US]	[Spain, Germany, Italy, France]	[Austria, Portugal, Brazil, Israel, Quebec, Ireland, Sweden, Czechia, Chile, Russia, Norway, Philippines, Romania, Poland, Ecuador, Ontario, Luxembourg, Malaysia, New South Wales, Pakistan, Denmark, Thailand, Saudi Arabia, Indonesia, India, Dominican Republic, Panama, South Africa, Mexico, Argentina, Finland, Peru, Serbia, Colombia, Victoria, Greece, Croatia, Ukraine]	[United Kingdom, Iran, Turkey, Belgium, Switzerland, Netherlands]
2020-02	[Korea South]	[Italy, Iran]	[Switzerland, Norway, Iraq, Sweden, United Kingdom, Austria, Taiwan*, Ontario, United Arab Emirates, Israel, Netherlands, Croatia, Oman, Mexico, Pakistan, Hong Kong, Greece, Lebanon, Denmark, Queensland, Romania, Victoria, Western Australia, Brazil, Finland, Ireland, Luxembourg, Monaco, Qatar, Belarus, Quebec, Iceland, Lithuania, New Zealand, Nigeria, Estonia, San Marino, Georgia]	[Japan, France, Germany, Spain, Kuwait, Diamond Princess, US, Bahrain]
2020-01	[Hubei]	[Zhejiang, Guangdong, Henan, Hunan, Jiangxi]	[Heilongjiang, Guizhou, Guangxi, Shanxi, Beijing, Ningxia, Gansu, Liaoning, Japan, Korea South, Xinjiang, Singapore, Hainan, Inner Mongolia, Jilin, Thailand, United Arab Emirates, Queensland, Germany, Italy, Russia, United Kingdom, Ontario, Qinghai, Victoria, Malaysia, Sweden, Tibet, India, Philippines, Taiwan*, Finland, British Columbia]	[Anhui, Jiangsu, Sichuan, Shandong, Chongqing, Hebei, Shanghai, Fujian, Shaanxi, Yunnan]

# Benchmark

## Prestazioni delle query

Funzionalità testata	Tempo medio di esecuzione e varianza (ms)	
	Multi-start	Single start
Query 1 (Spark Core)	$35,6667 \pm 9,0779$	$1275,3333 \pm 92,5304$
Query 1 (Spark SQL)	$775,2727 \pm 24,5564$	$6229,0 \pm 147,6144$
Query 2 (Spark Core)	$114,8788 \pm 4,6187$	$1492,3333 \pm 21,3417$
Query 2 (Spark SQL)	$1905,2727 \pm 48,7015$	$8025,6667 \pm 130,3912$
Query 3 (Naive)	$1399,5657 \pm 48,1844$	$3705,6667 \pm 54,8075$
Query 3 (MLlib)	$764,4040 \pm 10,1789$	$2973,1667 \pm 80,0585$

# Benchmark

## Prestazioni degli algoritmi di clustering

Mese	Somma delle WSSSE dei cluster individuati	
	Naive	MLlib
Gennaio 2020	48,8278215766480 <sup>56</sup>	48,8278215766480 <sup>4</sup>
Febbraio 2020	4,2571098809791845	4,2190678645824695
Marzo 2020	16343,192726288 <sup>22</sup>	16343,192726288 <sup>196</sup>
Aprile 2020	4128,697916301173	4128,69791630117
Maggio 2020	3820,84645202067 <sup>17</sup>	3820,84645202067 <sup>44</sup>

## REFERENCES:

1. <https://github.com/pcm-dpc/COVID-19/blob/master/dati-andamento-nazionale/dpc-covid19-it-andamento-nazionale.csv>
2. [https://github.com/CSSEGISandData/COVID-19/blob/master/csse\\_covid\\_19\\_data/csse\\_covid\\_19\\_time\\_series/time\\_series\\_covid19\\_confirmed\\_global.csv](https://github.com/CSSEGISandData/COVID-19/blob/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_confirmed_global.csv)
3. <https://craiggrummitt.com/2019/02/27/get-a-continent-from-longitude-latitude/>
4. <https://nominatim.openstreetmap.org/>
5. <https://www.bigdatacloud.com/>
6. <http://theory.stanford.edu/~sergei/papers/vldb12-kmpar.pdf>
7. [https://en.wikipedia.org/wiki/Lloyd%27s\\_algorithm](https://en.wikipedia.org/wiki/Lloyd%27s_algorithm)