Esercitazione
Spark A.A. 2024/2025

Alessandro Mele
a.mele@pm.univpm.it

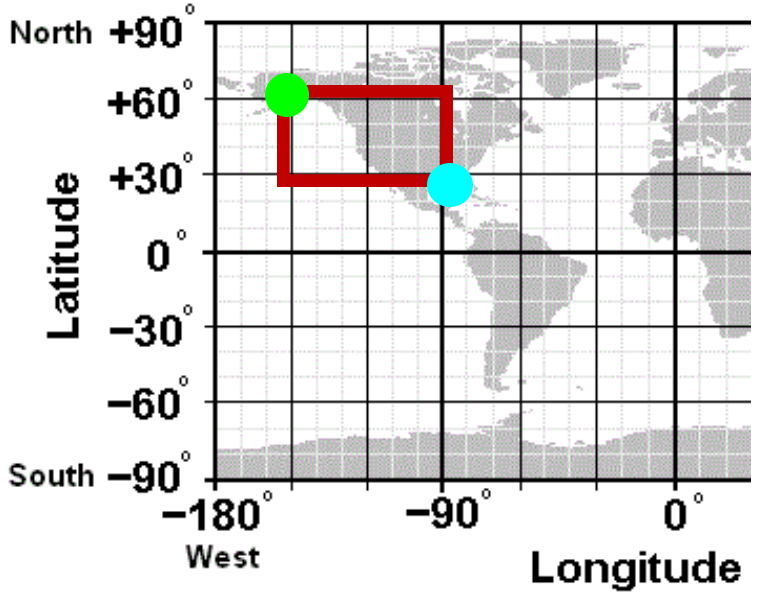
Challenge

- **Download:** link
  - Versione (molto) ridotta, qualche MB
  - Reale circa 15GB
- **Dataset**
  - Registrazioni di log e misurazioni relative a stazioni metereologiche
  - Una cartella per ogni anno
  - Per ogni anno, un file *csv* per ogni stazione (il nome del file è il nome della stazione)
    - Numero variabile di stazioni per ogni anno
    - NOTA: i file hanno alcuni campi non in comune, il programma deve gestire la situazione

Task 1

- Salvare su file le prime 10 temperature (TMP) con il maggior numero di occorrenze (ed il relativo conteggio registrate nell'area evidenziata (ordinate per temperatura)

COORDS	TMP	OCC
[(60,-135);(30,-90)]	21.1	50
[(60,-135);(30,-90)]	22.15	40
[(60,-135);(30,-90)]	22.7	60
[(60,-135);(30,-90)]	23.17	30



Task 2

- La colonna *WND* contiene le informazioni relative al vento in formato «150,1,N,0041,1».
  - L'elemento evidenziato esprime la velocità in nodi
- Salvare su file, per ogni velocità, la stazione in cui occorre più volte, riportare anche il relativo conteggio (ordinando per velocità)

VEL	STATION	OCC
1	s1	40
7	s2	45
9	s3	20

Task 3

- La colonna *REM* contiene osservazioni relative ad alcuni fenomeni
  - Tra questi, il campo *'HOURLY INCREMENTAL PRECIPITATION VALUES (IN)'* esprime, per ogni ora, la quantità di precipitazioni registrate
  - Es: '… HOURLY INCREMENTAL PRECIPITATION VALUES (IN):.00 .02 .00 T .01 .00 .00 …'
  - Nota: 'T' sta per trascurabile, quindi va considerato 0
- Salvare su file le prime 10 stazioni con il valor medio di precipitazioni più basso per ogni anno

VEL	STATION	AVG
2000	s1	0.01
2000	…	0.2
2001	s3	0.04
2001	…	0.06

Struttura script


```
from pyspark.context import SparkContext
from pyspark.sql.session import SparkSession

sc = SparkContext.getOrCreate()
spark = SparkSession(sc)

entry_point = ‘hdfs:///user/amircoli/BDAchallenge2324’
save_ point = ‘/home/amircoli/Scrivania/BDA/spark2425/results/gruppo_n’

# your code
```

Suggerimento lettura HDFS


```
# connect to hdfs
fs = spark._jvm.org.apache.hadoop.fs.FileSystem.get(spark._jsc.hadoopConfiguration())

# get files starting from dir_path
list_status = fs.listStatus(spark._jvm.org.apache.hadoop.fs.Path(dir_path))

# get file names
file_names = [files.getPath().getName() for files in list_status]
```

Note

- Entry point dati cluster
  - **`hdfs:///user/amircoli/BDAchallenge2324`**
- Path salvataggio risultati
  - **`/home/amircoli/Scrivania/BDA/spark2425/results/gruppo_n`** **(n numero gruppo)**
- **CONSEGNA**: unico file *gruppo_n.zip* con
  - Presentazione
  - Script *gruppo_n.py* con il codice della soluzione
  - Files *.csv* dei risultati, uno per ogni richiesta
    - *gruppo_n_r1.csv*, *gruppo_n_r2.csv*, …
- Punteggio
  - Efficacia: 0.8
  - Efficienza: 0.2
- **Testare il codice sulle versioni di Spark e HDFS delle macchine virtuali**