

# **Hard Disk Failure Data Processing**

**Corso di Sistemi ed Architetture per Big Data**

**Luca Falasca    Matteo Conti**  
**0334722            0323728**

Progetto 1 - Batch processing

**2024**

# Indice

## 1. Introduzione

- Obiettivi
- Dataset

## 2. Pipeline

- Data ingestion
- Data storage
- Data processing
- Analytical data storage
- Data visualization

## 3. Conclusioni

- Risultati
- Performance



# Introduzione

# Obiettivi

Il progetto verte sull'analisi di un dataset contenente dati riguardanti il monitoraggio di dischi rigidi installati all'interno di un cluster di server gestito da un cloud provider, in particolare si vuole:

- Realizzare una pipeline di elaborazione dati
- Eseguire le query richieste dalla specifica
- Visualizzare i risultati
- Analizzare le performance ottenute con i formati dati CSV e Parquet



# Dataset

Il dataset fornito è una versione ridotta di quello presentato nel Grand Challenge della conferenza ACM DEBS 2024. Delle numerose colonne presenti nel dataset, ne verranno selezionate solamente cinque, in particolare:

- *date*: data della misurazione nel formato 'YYYY-MM-DD'
- *serial\_number*: identificativo del disco rigido
- *model*: modello del disco rigido
- *s9\_power\_on\_hours*: tempo di accensione del disco rigido in ore
- *vault\_id*: identificativo del gruppo di storage server a cui il disco rigido appartiene
- *failure*: flag che indica se il disco rigido ha subito una failure o meno

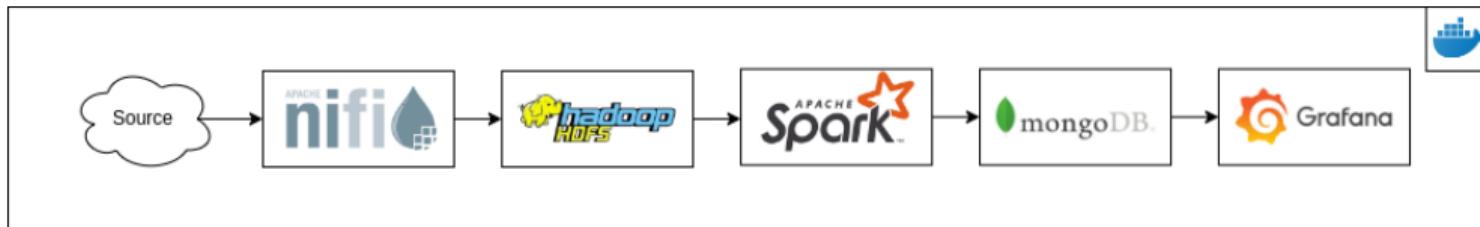


# Pipeline

# Pipeline

La pipeline di elaborazione dati è stata containerizzata e deployata tramite docker compose ed è composta da cinque componenti:

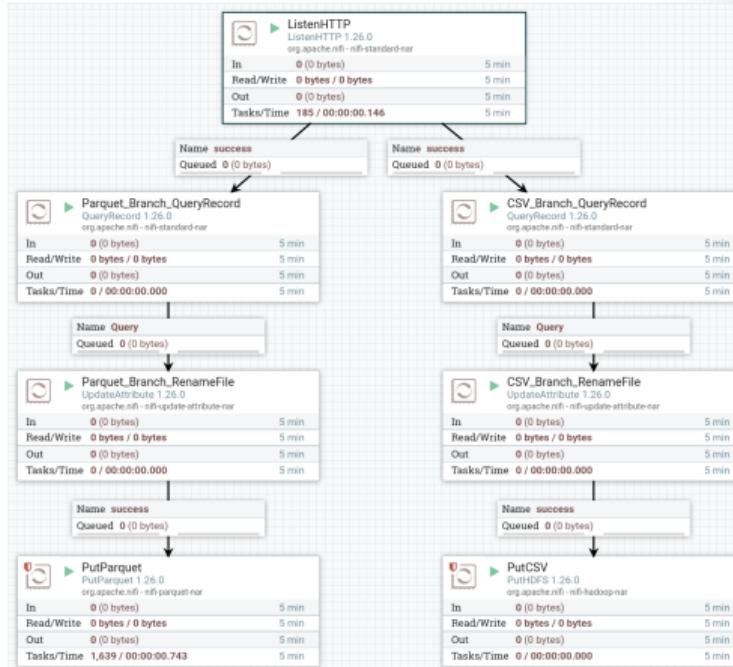
- Data ingestion
- Data storage
- Data processing
- Analytical data storage
- Data visualization



# Data ingestion-Apache NiFi

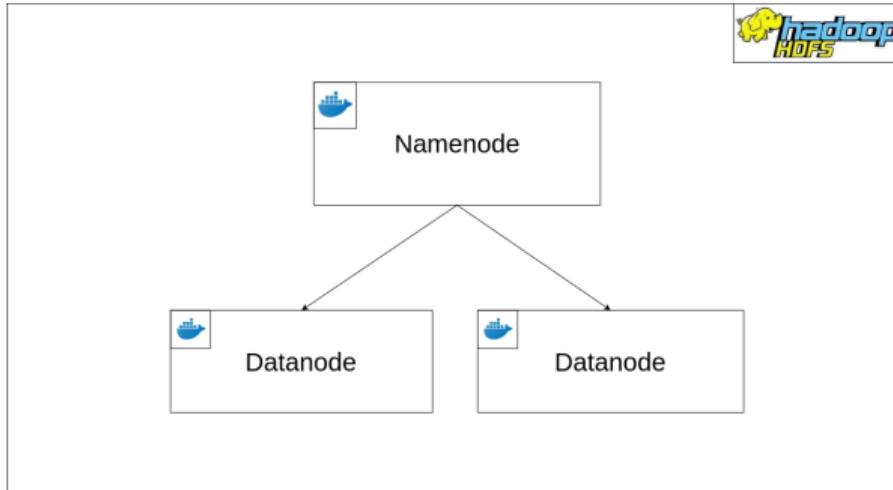
Per la data ingestion è stato utilizzato il framework Apache NiFi, il quale si occupa di:

- Ricevere tramite HTTP il file CSV contenente i dati
- Filtrare i dati
- Memorizzare i dati su HDFS in formato Parquet e CSV



# Data storage-HDFS

Per lo storage del dataset filtrato è stato utilizzato HDFS, non è stata utilizzata una particolare struttura del filesystem, in particolare i dati sono stati memorizzati nella root directory

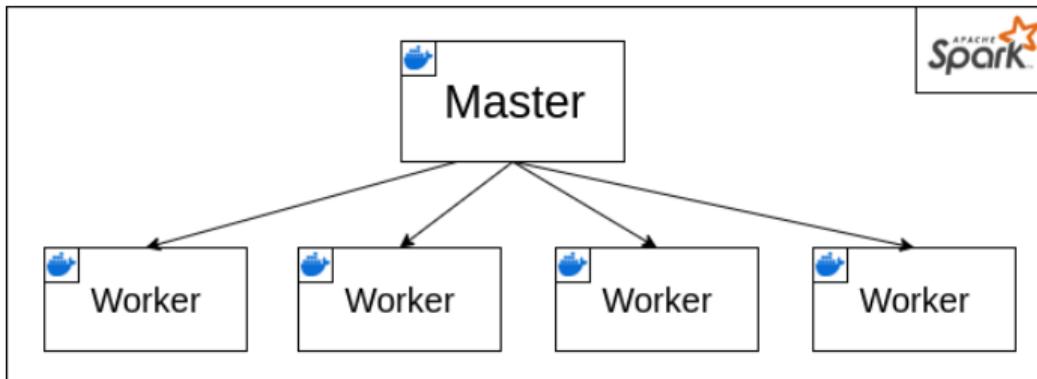


# Data Processing-Apache Spark



Per il processamento delle query è stato utilizzato **Apache Spark**

- Cluster formato da 1 nodo master e 4 nodi worker
- Replicazione docker compose



# Data Processing

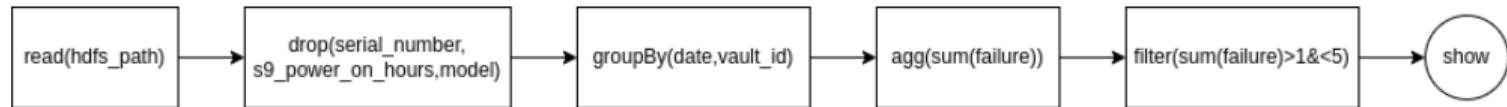
Considerazioni:

- Ogni query è stata eseguita sia in Parquet che in CSV
- Standardizzazione dell'esecuzione con show()
- Utilizzo di DataFrame per la manipolazione dei dati



# Data Processing-Query 1

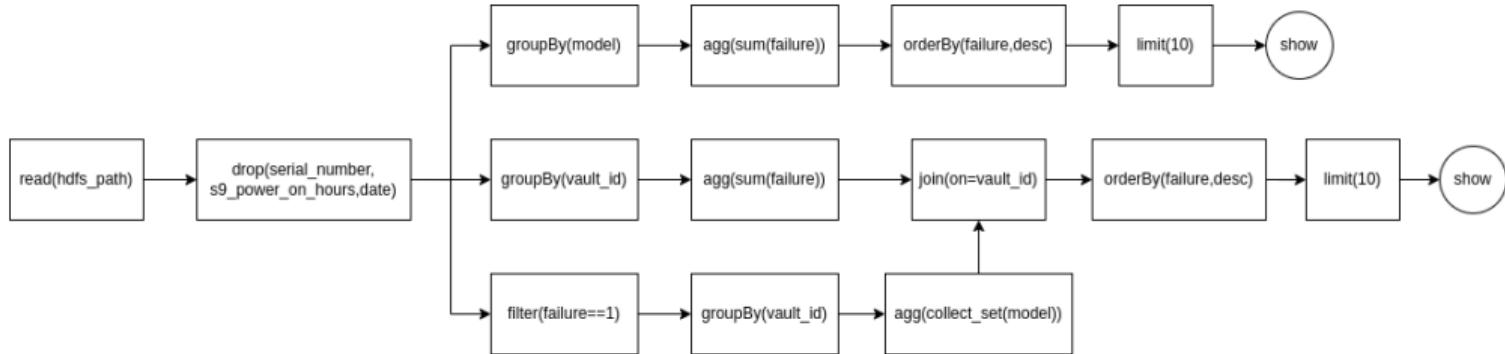
**Q1:** calcolare per ogni giorno, per ogni vault, il numero totale di fallimenti, in particolare determinare la lista di vault che hanno subito esattamente 4, 3 e 2 fallimenti



# Data Processing-Query 2

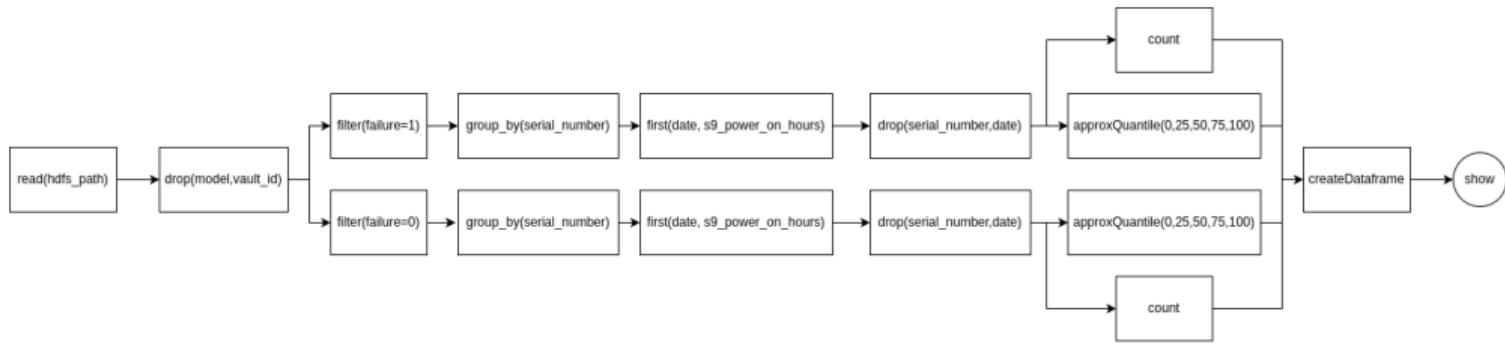
**Q2.1:** Calcolare la classifica dei 10 modelli di hard disk che hanno subito il maggior numero di fallimenti, riportando il modello di hard disk e il numero totale di fallimenti subiti dagli hard disk di quello specifico modello

**Q2.2:** Calcolare la classifica dei 10 vault che hanno registrato il maggior numero di fallimenti riportando, per ogni vault, il numero di fallimenti e la lista (senza ripetizioni) dei modelli di hark disk soggetti ad almeno un fallimento



# Data Processing-Query 3

**Q3:** Calcolare il minimo, 25-esimo, 50-esimo, 75-esimo percentile e massimo delle ore di funzionamento hark disk che hanno subito fallimenti e degli hard disk che non hanno subito fallimenti, indicando anche il numero totale di eventi utilizzati per il calcolo delle statistiche



# Analytical data storage-MongoDB

Per lo storage dei risultati è stato utilizzato  
**MongoDB**

La scrittura da Spark è stata effettuata tramite  
**il Spark-MongoDB connector**

Organizzazione dei dati:

- 1 collezione per ogni query + 1 per performance
- 1 documento per ogni riga



Viewing Database: results

Collections	Collection Name	+ Create collection			
 View	 Export	 [JSON]	 Import	performance	 Del
 View	 Export	 [JSON]	 Import	query1	 Del
 View	 Export	 [JSON]	 Import	query2.1	 Del
 View	 Export	 [JSON]	 Import	query2.2	 Del
 View	 Export	 [JSON]	 Import	query3	 Del

# Data visualization-Grafana

Per la visualizzazione dei dati è stato utilizzato  
**Grafana**



Pannelli realizzati:

- 5 pannelli per la visualizzazione dei risultati delle query
- 1 pannello per la visualizzazione delle performance

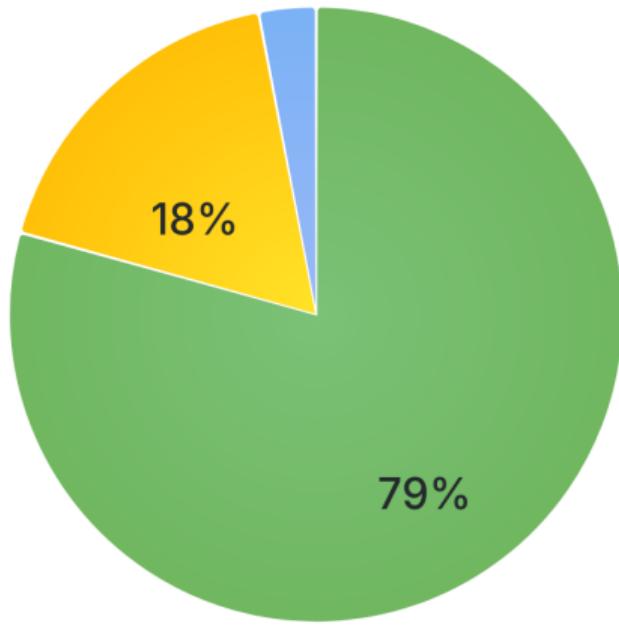




# Conclusioni

# Risultati-Query1

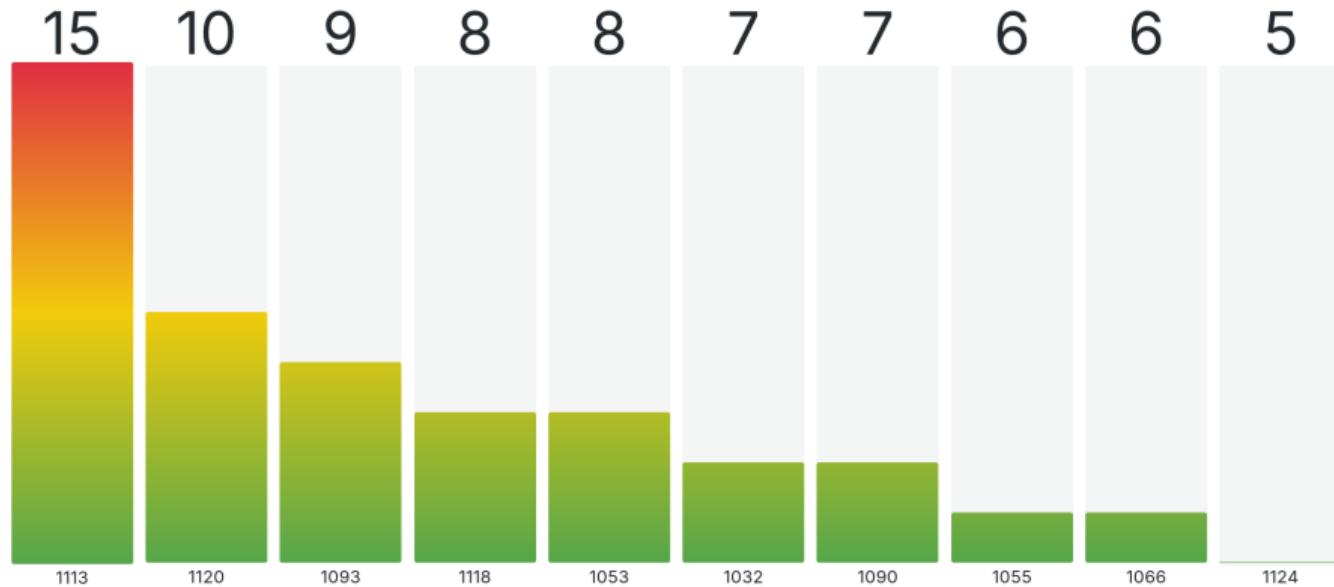
Number of Vault id for Failure



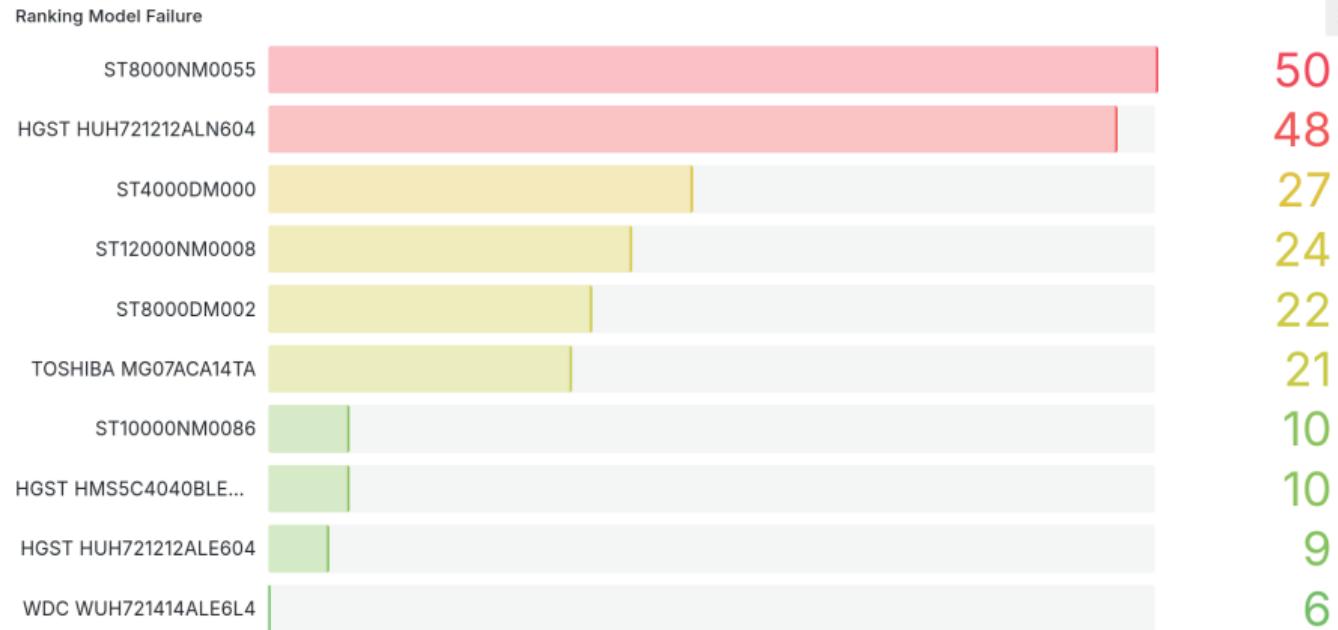
— 2 Failure — 3 Failure — 4 Failure

# Risultati-Query2

Vault id Failures



# Risultati-Query2

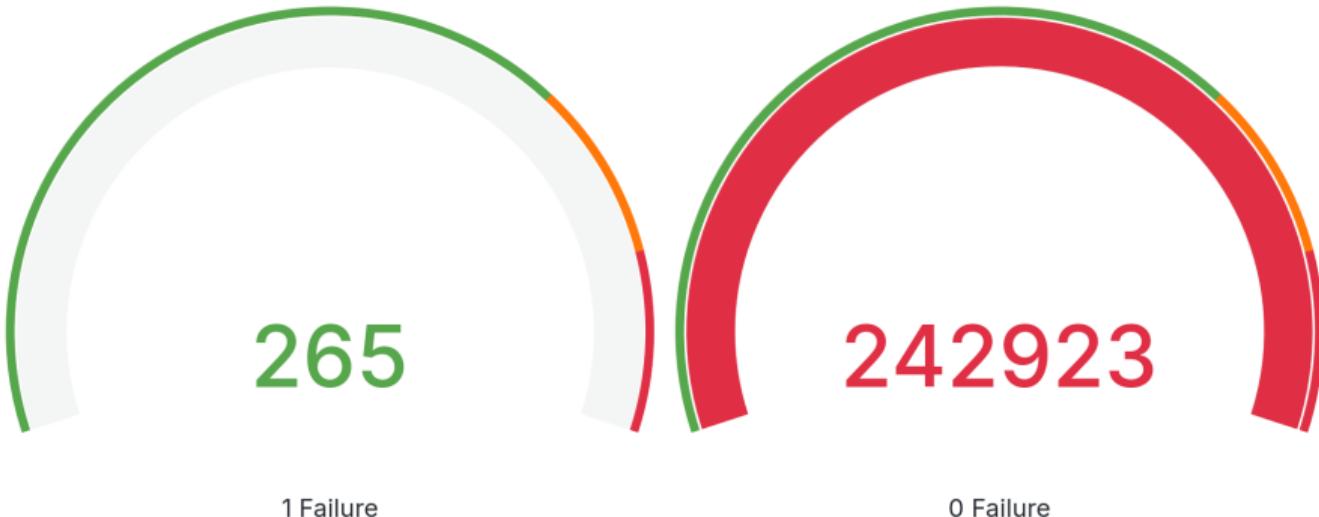


# Risultati-Query3

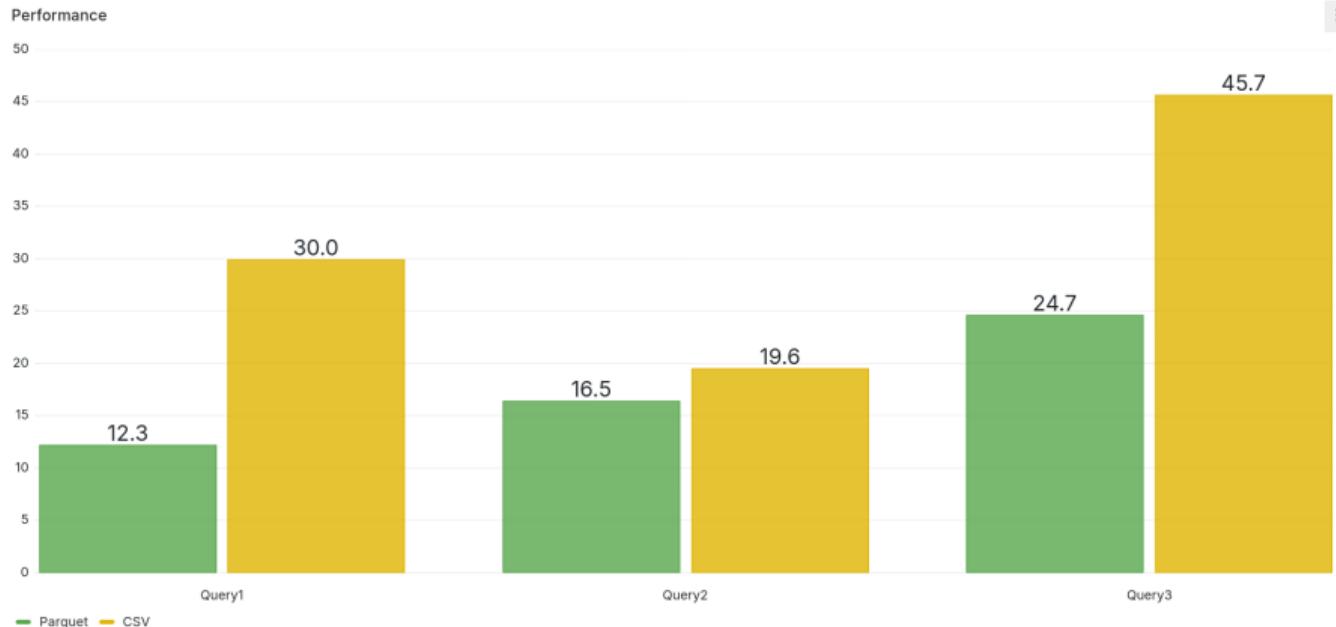


# Risultati-Query3

Count



# Performance





Grazie per l'attenzione