

# Relazione ex1

## 1. Introduzione

Abbiamo testato due algoritmi di ordinamento, MergeSort e QuickSort, utilizzando un dataset di 20 milioni di record.

Gli algoritmi sono stati testati su tre campi di ordinamento: una stringa (`string_field`), un intero (`integer_field`) e un float (`floating_field`).

L'obiettivo era misurare e confrontare i tempi di esecuzione per ciascun campo e algoritmo.

## 2. Risultati dei Test

MergeSort con `string_field`:

Tempo di ordinamento: 20 secondi

MergeSort con `integer_field`:

Tempo di ordinamento: 24 secondi

MergeSort con `floating_field`:

Tempo di ordinamento: 24 secondi

QuickSort con `integer_field`:

Tempo di ordinamento: 28 secondi

QuickSort con `floating_field`:

Tempo di ordinamento: 26 secondi

Nota:

Il test con QuickSort e il campo `string_field` ha superato il tempo limite di 10 minuti e quindi è stato interrotto.

## 3. Analisi dei Risultati

- MergeSort ha dimostrato di essere relativamente consistente in termini di tempi di esecuzione, con una media di circa 23 secondi per tutti i campi. Questo è atteso poiché MergeSort ha una complessità temporale di  $O(n \log n)$  sia nel caso migliore che peggiore, che è relativamente stabile indipendentemente dai dati specifici.
- QuickSort ha mostrato tempi di esecuzione più variabili, con un tempo di circa 24 secondi per `floating_field`, ma ha richiesto più tempo con `integer_field` (28 secondi). Questo può essere dovuto al fatto che QuickSort può peggiorare la sua prestazione fino a raggiungere una complessità temporale  $O(n^2)$  nel caso peggiore, e il campo

`integer_field` potrebbe avere una distribuzione di dati che ha portato a un numero maggiore di partizioni squilibrate.

## 4. Problemi Riscontrati:

Il test con QuickSort e `string_field` è stato interrotto perché ha superato il tempo limite di 10 minuti. Questo suggerisce che il QuickSort potrebbe essere meno efficiente su campi di tipo stringa con grandi dataset, probabilmente a causa della gestione delle comparazioni di stringhe che possono essere più costose rispetto a numeri interi o float.

## 5. Conclusione

I risultati confermano che MergeSort è più consistente e meno influenzato dalla natura dei dati, mentre QuickSort può variare notevolmente in base ai dati e alla distribuzione dei valori. In scenari con dataset molto grandi e campi di ordinamento complessi come le stringhe, MergeSort potrebbe essere preferibile per garantire tempi di esecuzione più prevedibili.