

BANCO DE DADOS

**BIG DATA**



**HADOOP**

Fabricio Rodrigues  
Fábio Junior  
Gabriel Batistuta  
Zezineto Mendes  
Joao Victor Freitas

# BIGDATA

## SOBRE

### Definição

Big Data refere-se a conjuntos de dados extremamente grandes e complexos que são difíceis de processar usando técnicas tradicionais de processamento de dados.

### Como surgiu

O Big Data surgiu no início dos anos 2000 em resposta ao crescimento exponencial da quantidade de dados gerados por empresas, dispositivos e interações online. Com o advento da internet e o aumento da capacidade de armazenamento digital, tornou-se necessário desenvolver novas técnicas e tecnologias para lidar com o volume massivo, a velocidade e a variedade de dados.



# BIGDATA

## DESAFIOS

### ALÉM DOS 5 Vs

- Segurança e Privacidade: Proteger os dados sensíveis contra acessos não autorizados.
- Escalabilidade: Manter a eficiência ao expandir a infraestrutura para lidar com mais dados.
- Armazenamento: Gerenciar o armazenamento eficiente de grandes volumes de dados.
- Análise: Ferramentas e técnicas para analisar e interpretar grandes conjuntos de dados

# BIGDATA

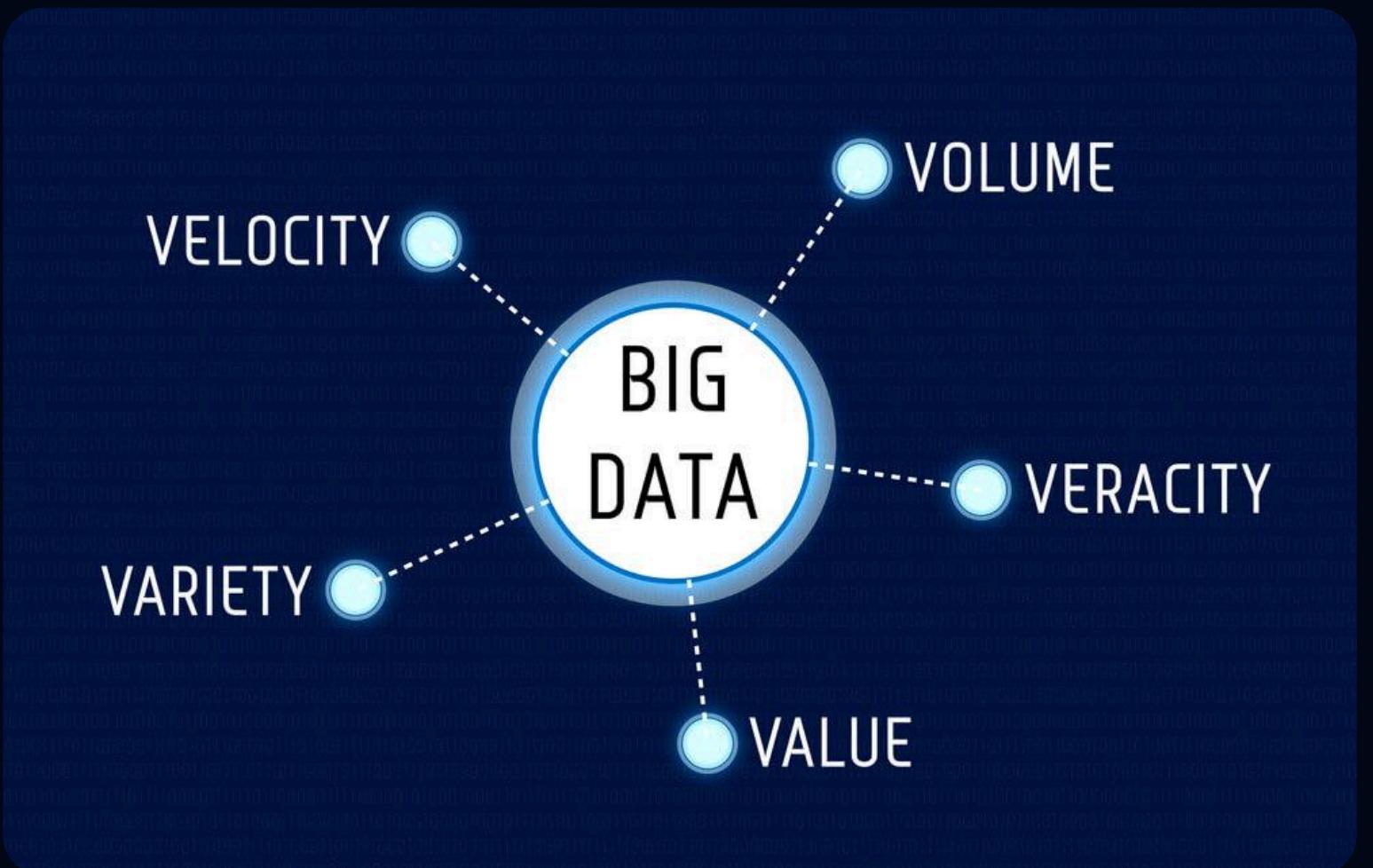
## OS Vs DO BIGDATA

### Volume

Refere-se à quantidade de dados gerada e armazenada. A quantidade de dados é enorme e está crescendo exponencialmente

### Variedade

Diferentes tipos de dados provenientes de várias fontes. Todos os tipos, formas e fontes de dados, estruturados ou não estruturados, que incluem documentos de texto, e-mails, dados de sensores, áudios, vídeos, imagens, bases de dados, etc.s estruturados, semi-estruturados e não estruturados.



### Valor

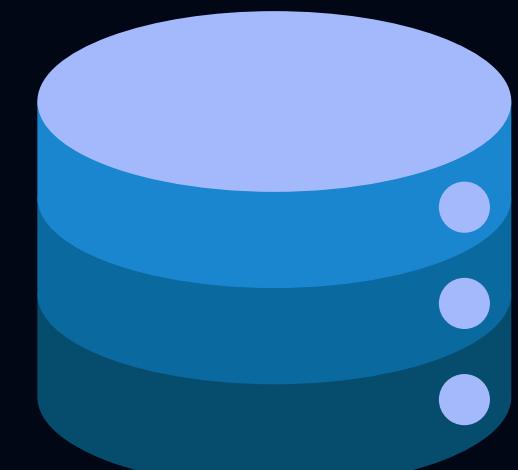
Transformar esses dados em insights açãoáveis que possam melhorar a tomada de decisões, otimizar processos, aumentar a eficiência operacional e criar novas oportunidades de negócio.

### Velocidadade

A velocidade em Big Data refere-se à rapidez com que os dados são gerados, coletados, processados e analisados. Isso inclui a taxa de chegada dos dados em tempo real ou quase em tempo real e a capacidade de processar e analisar esses dados rapidamente para obter insights imediatos.

### Veracidadade

A qualidade e a precisão dos dados. Garantir que os dados são confiáveis e corretos.



# BIGDATA

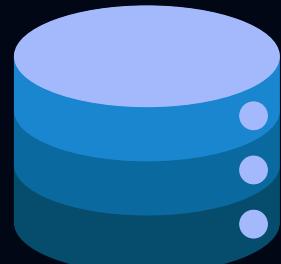
## SOBRE

- Ao introduzir Big Data, é crucial entender que não se trata apenas da quantidade de dados, mas também da diversidade, velocidade, qualidade e o potencial valor desses dados.
- Os 5 Vs são fundamentais para caracterizar e diferenciar Big Data dos dados tradicionais que muitas empresas já lidam.

# BIGDATA

## MEDIDA DE ARMAZENAMENTO

Unidade	Símbolo	Número de bytes
kilobyte	kB	$2^{10} = 1024$ bytes
megabyte	MB	$2^{20} = 1,048,576$ bytes
gigabyte	GB	$2^{30} = 1,073,741,824$ bytes
terabyte	TB	$2^{40} = 1,099,511,627,776$ bytes
petabyte	PB	$2^{50} = 1,125,899,906,842,624$ bytes
exabyte	EB	$2^{60} = 1,152,921,504,606,846,976$ bytes
zettabyte	ZB	$2^{70} = 1,180,591,620,717,411,303,424$ bytes
yottabyte	YB	$2^{80} = 1,208,925,819,614,629,174,706,176$ bytes



# BIGDATA

## Exemplos de uso

Diariamente encaramos o Big Data: por exemplo, as sugestões dos aplicativos como o Spotify, Netflix, Amazon, Twitter e Facebook se baseiam em algoritmos de aprendizagem automática (Machine Learning) que processam dados relacionados a nossas interações com essas plataformas.



# BIGDATA

## Exemplos de uso

### SAÚDE

No setor de saúde são gerados muitos dados diariamente, e a análise desses dados permite otimizar tanto a gestão clínica como o tratamento e o atendimento ao paciente. A análise de macrodados na investigação médica permite identificar fatores de risco, realizar diagnósticos mais precisos, detectar doenças em suas primeiras etapas, antecipar e combater epidemias.

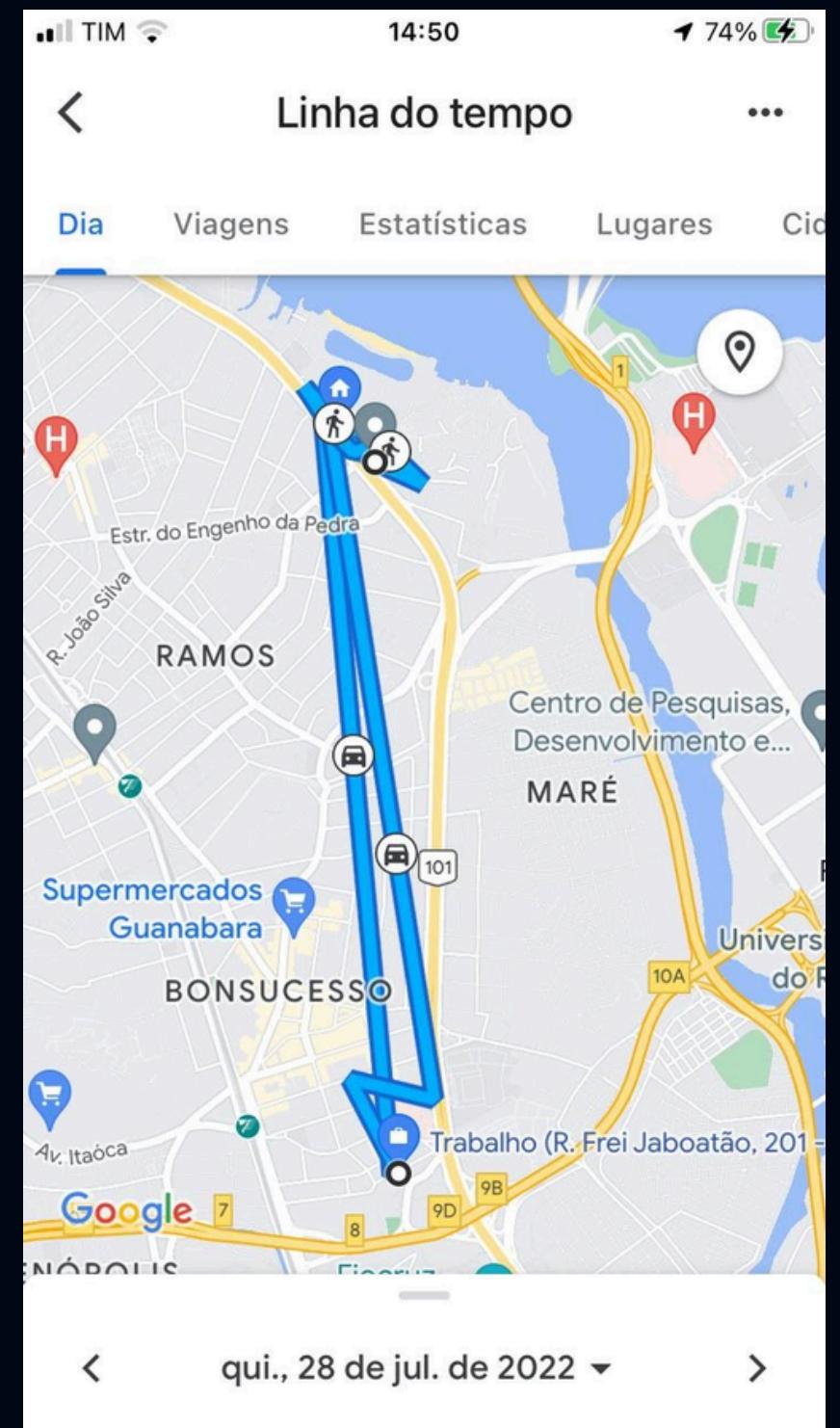


# BIGDATA

## Exemplos de uso

### COMPORTAMENTO DO USUÁRIO

- **Sites e aplicativos:** Rastreando cliques, visualizações, tempo na página e outros comportamentos online, as empresas podem entender como os usuários navegam e interagem com seus produtos e serviços.
- **Mídias sociais:** Analisando curtidas, compartilhamentos, comentários e menções, as empresas podem identificar tendências, avaliar a percepção da marca e se conectar com os clientes em um nível mais profundo.
- **Dispositivos móveis:** Coletando dados de geolocalização, uso de aplicativos e histórico de navegação, as empresas podem entender como os usuários interagem com seus produtos e serviços em seus dispositivos móveis.



# BIGDATA

Exemplos de uso

## GOVERNO

### Saúde

Monitoramento de surtos de doenças e pandemias em tempo real. Análise de dados de pacientes para melhorar os serviços de saúde e prever necessidades futuras.

### Segurança e Aplicação da Lei

Análise de dados de criminalidade para prever e prevenir crimes. Uso de dados de câmeras de vigilância e sensores para monitoramento de segurança.

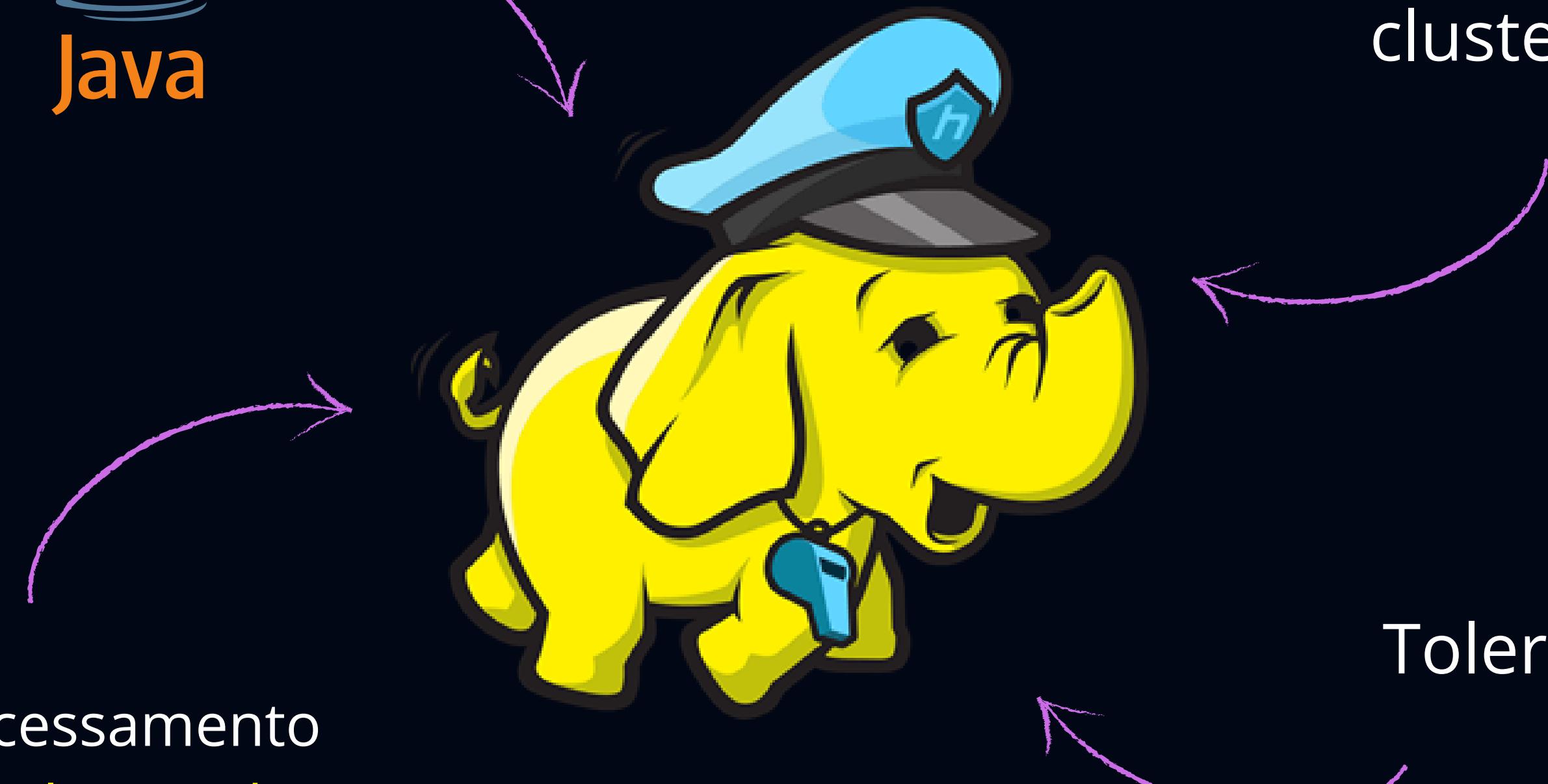
### Gestão de Recursos Naturais e Meio Ambiente

Monitoramento de recursos hídricos, florestas e biodiversidade. Análise de dados ambientais para promover a sustentabilidade e a conservação.

# BIGDATA

## SOBRE HADOOP

Utiliza  
  
Java



Lida com processamento  
de grandes volumes de  
dados

feita para computação  
distribuída voltada para  
clusters

Tolerante a falhas

# BIGDATA

## QUANDO O HADOOP FOI INVENTADO

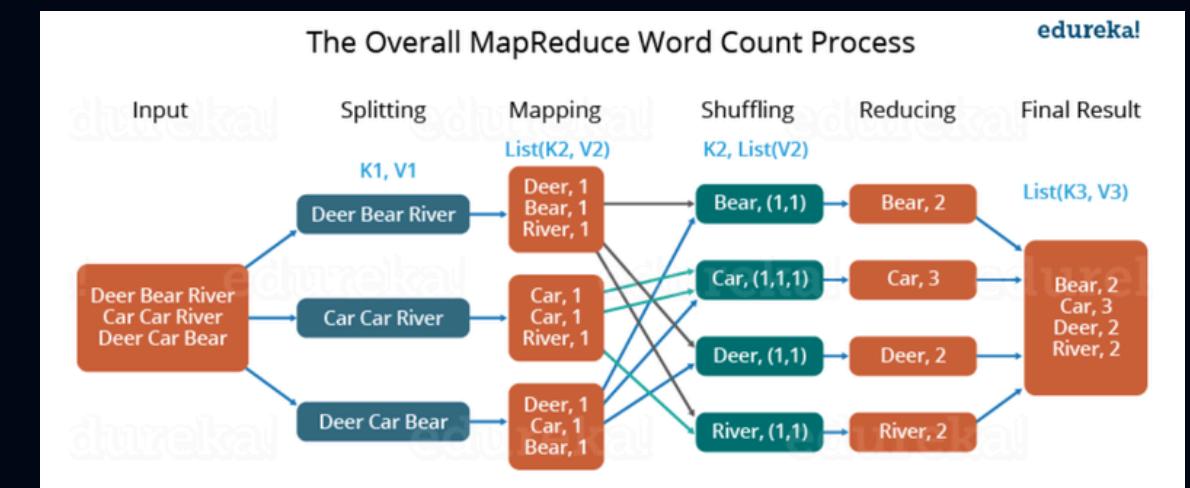


O Apache Hadoop nasceu da necessidade de processar volumes cada vez maiores de big data e fornecer resultados online mais rapidamente à medida que mecanismos de pesquisa como Yahoo e Google se tornavam cada vez mais populares.

Inspirado no MapReduce do Google, um modelo de programação que divide uma aplicação em pequenas frações para ser executada em diferentes nós, Doug Cutting e Mike Cafarella iniciaram o Hadoop em 2002 enquanto trabalhavam no projeto Apache Nutch. De acordo com um artigo do New York Times, o nome Hadoop é uma homenagem ao elefante de brinquedo do filho de Doug.



Alguns anos depois, o Hadoop separou-se do Nutch. O Nutch concentrou-se no elemento rastreador da web, e o Hadoop tornou-se a parte de processamento e computação distribuída. Dois anos depois de Cutting ingressar no Yahoo, o Hadoop foi lançado como um projeto de código aberto em 2008. A Apache Software Foundation (ASF) disponibilizou o Hadoop ao público em novembro de 2012 como Apache Hadoop.



# BIGDATA

## QUAIS SÃO OS PRINCIPAIS MODULOS DO HADOOP

- **HDFS** - Sistema de arquivos distribuídos Hadoop. O HDFS é um sistema baseado em Java que permite que grandes conjuntos de dados sejam armazenados em nós de um cluster de maneira tolerante a falhas.
- **YARN** - Sigla para Yet Another Resource Negotiator. O YARN é usado para gerenciamento de recursos de cluster, planejamento de tarefas e agendamento de jobs em execução no Hadoop.
- **MapReduce** - O MapReduce é um modelo de programação e mecanismo de processamento de big data usado no processamento paralelo de grandes conjuntos de dados. Originalmente, o MapReduce era o único mecanismo de execução disponível no Hadoop. Mais tarde, no entanto, o Hadoop adicionou suporte a outros, incluindo Apache Tez e Apache Spark.
- **Hadoop Common** - O Hadoop Common fornece um conjunto de serviços entre bibliotecas e utilitários para oferecer suporte aos outros módulos Hadoop.

# BIGDATA

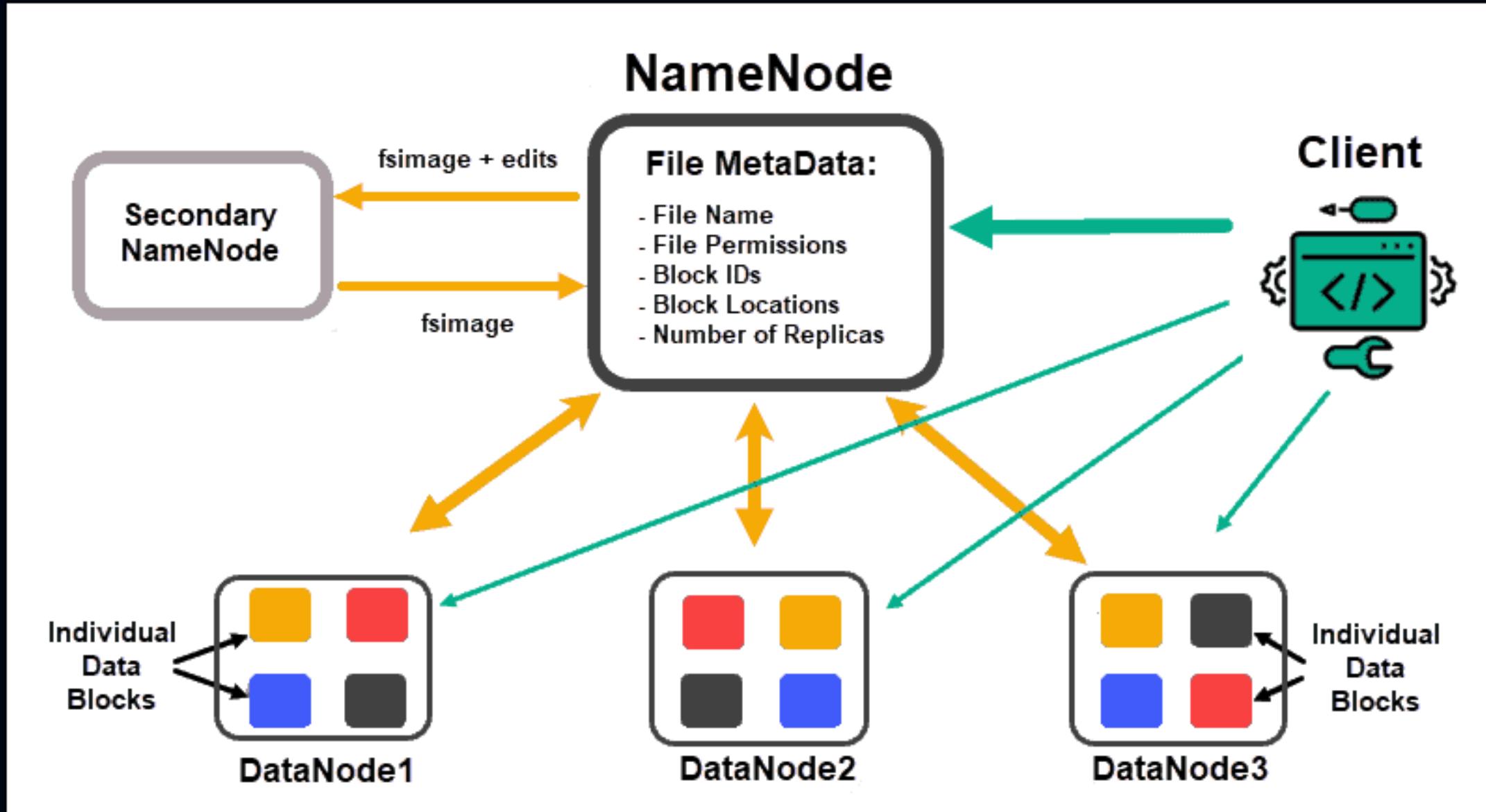
## CARACTERISTICAS DO HADOOP

- TOLERANTE A FALHAS
- CONSISTENCIA
- ESCALAVEL
- BAIXO CUSTO

# BIGDATA

## NAMENODE

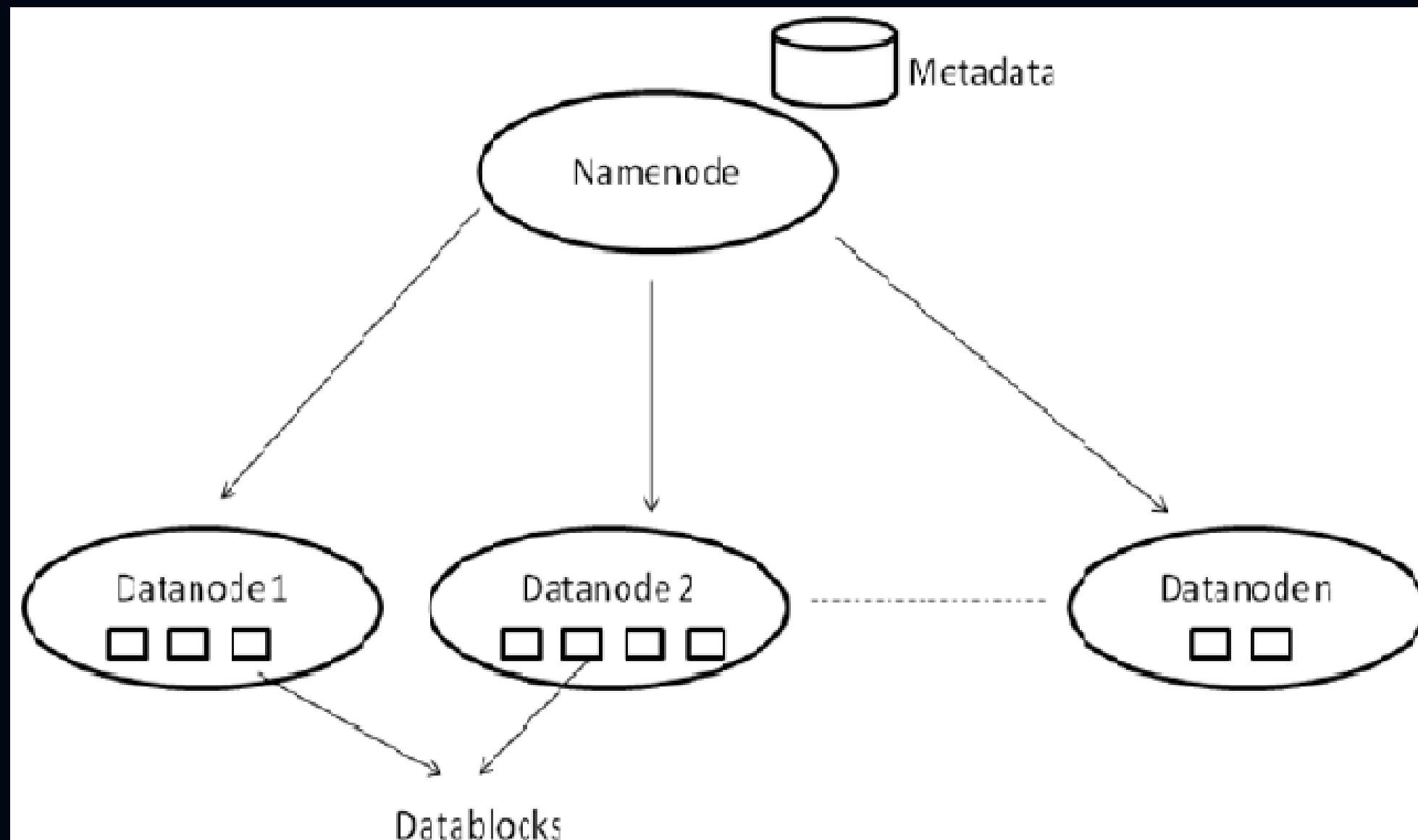
## Definição



**é um nó no cluster que sabe quais são os dados onde os blocos pertencem, qual é o tamanho dos blocos e para onde devem ir. É responsável pelo gerenciamento de arquivos do HDFS**

# BIGDATA

## DATA NODE



## Definição

localiza-se nos nós; são os dados que são distribuídos e replicados. Cada DataNode reporta-se ao NameNode, indicando quais blocos guarda e as atualizações realizadas neles.

# BIGDATA

## MAP REDUCE

```
public class InvertedIndexMapper extends Mapper<Object, Text, Text, Text> {  
    private static final Log LOG = LogFactory.getLog(InvertedIndexMapper.class);  
  
    @Override  
    protected void map(Object key, Text value, Context context) throws IOException,  
        InterruptedException {  
        StringTokenizer tokenizer = new StringTokenizer(value.toString(), ",; \\\\t\\\\n?!\\\\\"  
            /()[]$ %  
            ");  
        while (tokenizer.hasMoreTokens()) {  
            String token = tokenizer.nextToken();  
            Text keyOut = new Text(token);  
            Apache Hadoop Tutorial 12 / 18  
            String fileName = ((FileSplit) context.getInputSplit()).getPath().toString();  
            Text valueOut = new Text(fileName);  
            LOG.info("Key/Value: " + keyOut + "/" + valueOut);  
            context.write(keyOut, valueOut);  
        }  
    }  
}
```

## Definição

MapReduce é um modelo de programação desenhado para processar grandes volumes de dados em paralelo, dividindo o trabalho em um conjunto de tarefas independentes.

# BIGDATA

## MAP REDUCE

The screenshot shows the Docker interface with the following details:

- Container CPU usage:** 2.04% / 800% (8 CPUs available)
- Container memory usage:** 1.65GB / 7.51GB
- Show charts** button
- Search bar** with placeholder "Search"
- Filter:** Only show running containers
- Table Headers:** Name, Image, Status, Port(s), CPU (%), Last started, Actions
- Table Data:** A single row for the container "docker-hadoop".
  - Name: docker-hadoop
  - Image: docker-hadoop
  - Status: Running (5/5)
  - CPU (%): 1.88%
  - Last started: 42 seconds ago
  - Actions: A menu icon with three dots and a trash bin icon.
- Showing 1 item** message
- Walkthroughs** section:
  - Multi-container applications**: 8 mins
  - Containerize your application**: 3 mins

## Implementação

A implementação foi feita com containers do docker e as instruções e arquivos foram dados pelo terminal do windows.

# BIGDATA

## MAP REDUCE

```
1 √ [{"city": "AGAWAM", "loc": [-72.622739, 42.070206], "pop": 15338, "state": "MA", "_id": "01001"}  
2 , {"city": "CUSHMAN", "loc": [-72.51564999999999, 42.377017], "pop": 36963, "state": "MA", "_id": "01002"}  
3 , {"city": "BARRE", "loc": [-72.10835400000001, 42.409698], "pop": 4546, "state": "MA", "_id": "01005"}  
4 , {"city": "BELCHERTOWN", "loc": [-72.41095300000001, 42.275103], "pop": 10579, "state": "MA", "_id": "01007"}  
5 , {"city": "BLANDFORD", "loc": [-72.936114, 42.182949], "pop": 1240, "state": "MA", "_id": "01008"}  
6 , {"city": "BRIMFIELD", "loc": [-72.188455, 42.116543], "pop": 3706, "state": "MA", "_id": "01010"}  
7 , {"city": "CHESTER", "loc": [-72.988761, 42.279421], "pop": 1688, "state": "MA", "_id": "01011"}  
8 , {"city": "CHESTERFIELD", "loc": [-72.833309, 42.38167], "pop": 177, "state": "MA", "_id": "01012"}  
9 , {"city": "CHICOPEE", "loc": [-72.607962, 42.162046], "pop": 23396, "state": "MA", "_id": "01013"}  
10 , {"city": "CHICOPEE", "loc": [-72.576142, 42.176443], "pop": 31495, "state": "MA", "_id": "01020"}  
11 , {"city": "WESTOVER AFB", "loc": [-72.558657, 42.196672], "pop": 1764, "state": "MA", "_id": "01022"}  
12 , {"city": "CUMMINGTON", "loc": [-72.905767, 42.435296], "pop": 1484, "state": "MA", "_id": "01026"}  
13 , {"city": "MOUNT TOM", "loc": [-72.67992099999999, 42.264319], "pop": 16864, "state": "MA", "_id": "01027"}  
14 , {"city": "EAST LONGMEADOW", "loc": [-72.505565, 42.067203], "pop": 13367, "state": "MA", "_id": "01028"}  
15 , {"city": "FEEDING HILLS", "loc": [-72.675077, 42.07182], "pop": 11985, "state": "MA", "_id": "01030"}]
```

## Código

Os dados a serem analisados são  
uma lista de cidades que contém  
população e sigla do estado

# BIGDATA

## MAP REDUCE

```
root@25f506deea5b:/tmp# hadoop jar json-sort-actual-1.0-SNAPSHOT.jar JsonSortingJob input output
```

```
2024-07-15 10:20:22,477 INFO mapreduce.Job: Running job: job_1721038234513_0001
2024-07-15 10:20:29,543 INFO mapreduce.Job: Job job_1721038234513_0001 running in uber mode : false
2024-07-15 10:20:29,544 INFO mapreduce.Job: map 0% reduce 0%
2024-07-15 10:20:33,585 INFO mapreduce.Job: Task Id : attempt_1721038234513_0001_m_000000_0, Status : FAILED
Error: java.lang.RuntimeException: java.lang.NoSuchMethodException: JsonSortingJob$JsonSortingMapper.<init>()
        at org.apache.hadoop.util.ReflectionUtils.newInstance(ReflectionUtils.java:135)
        at org.apache.hadoop.mapred.MapTask.runNewMapper(MapTask.java:759)
        at org.apache.hadoop.mapred.MapTask.run(MapTask.java:347)
        at org.apache.hadoop.mapred.YarnChild$2.run(YarnChild.java:174)
        at java.security.AccessController.doPrivileged(Native Method)
        at javax.security.auth.Subject.doAs(Subject.java:422)
        at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1730)
        at org.apache.hadoop.mapred.YarnChild.main(YarnChild.java:168)
Caused by: java.lang.NoSuchMethodException: JsonSortingJob$JsonSortingMapper.<init>()
        at java.lang.Class.getConstructor0(Class.java:3082)
        at java.lang.Class.getDeclaredConstructor(Class.java:2178)
        at org.apache.hadoop.util.ReflectionUtils.newInstance(ReflectionUtils.java:129)
        ... 7 more
```

## Resultado

O resultado do código criado  
infelizmente não foi o desejado.

# BIGDATA

## MAP REDUCE

```
2024-07-15 10:20:44,660 INFO mapreduce.Job: map 100% reduce 100%
2024-07-15 10:20:45,672 INFO mapreduce.Job: Job job_1721038234513_0001 failed with state FAILED due to: Task failed task
_1721038234513_0001_m_000000
Job failed as tasks failed. failedMaps:1 failedReduces:0 killedMaps:0 killedReduces: 0

2024-07-15 10:20:45,723 INFO mapreduce.Job: Counters: 10
    Job Counters
        Failed map tasks=4
        Killed reduce tasks=1
        Launched map tasks=4
        Other local map tasks=3
        Rack-local map tasks=1
        Total time spent by all maps in occupied slots (ms)=21668
        Total time spent by all reduces in occupied slots (ms)=0
        Total time spent by all map tasks (ms)=5417
        Total vcore-milliseconds taken by all map tasks=5417
        Total megabyte-milliseconds taken by all map tasks=22188032
```

## Resultado

O resultado do código criado  
infelizmente não foi o desejado.