INTRODUÇÃO AO SPARK E RDDS

Uma visão sobre processamento distribuído eficiente

AUGUSTO MIGUEL FAUSTINO DOS SANTOS

POR QUE SPARK?

1. Problema:
    1. - MapReduce é ineficiente para processamento iterativo (ex: aprendizado de máquina) e consultas interativas (ex: consultas ad-hoc).
    2. - Muitas operações de I/O em disco (lentas).
2. Solução:
    1. - Spark mantém os dados na memória, acelerando o processamento em até 20x (exemplo: regressão logística).
    2. - Suporta DAGs (Grafos Acíclicos Dirigidos) que são mais eficientes do que o MapReduce linear.
    3. - Oferece uma API mais rica e flexível (SQL, streaming, ML).

O QUE É UM RDD?

1. **Conjunto de Dados Distribuídos Resilientes (RDD)** = Uma coleção imutável e particionada de dados distribuídos.
2. **Tolerância a falhas** via lineage – Recomputação de dados perdidos.
3. **Persistência** – Pode ser armazenado na memória ou no disco.
4. **Particionamento** – Otimiza operações como join e groupBy.
5. **Exemplo de código (Scala):**

```
1 val data = spark.textFile("hdfs://...")
2 val filtered = data.filter(_.contains("ERROR"))
3 filtered.persist() // Mantém na memória
```

APLICAÇÕES DO SPARK/RDDS

Processamento Iterativo

- Algoritmos de Aprendizado de Máquina (regressão logística, k-means).

1. - Exemplo: PageRank é 7.4x mais rápido que o Hadoop.

Consultas Interativas

2. - Análise de logs em tempo real (ex: filtrando erros em terabytes de dados).

Processamento de Gráficos com Pregel

3. - Implementação eficiente de algoritmos como PageRank.

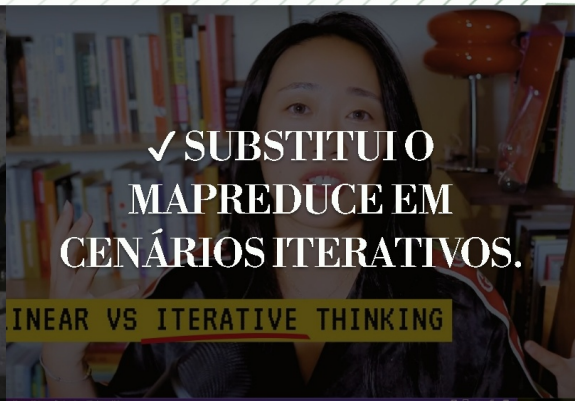
ETL e Pipelines de Dados

4. - Transformações complexas usando join, groupBy, etc.

CONCLUSÃO

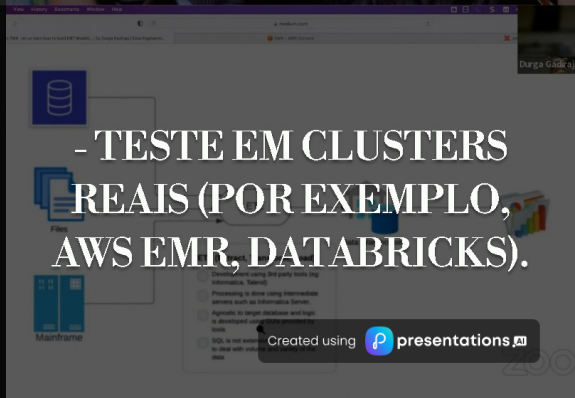
✓ SPARK É MAIS RÁPIDO (EVITA I/O DESNECESSÁRIO).

✓ RDDS PERMITEM UMA TOLERÂNCIA A FALHAS EFICIENTE.



✓ SUBSTITUI O MAPREDUCE EM CENÁRIOS ITERATIVOS.

✓ AMPLA GAMA DE APLICAÇÕES (ML, GRÁFICOS, STREAMING, SQL).



- TESTE EM CLUSTERS REAIS (POR EXEMPLO, AWS EMR, DATABRICKS).