



Hadoop e armazenamento de dados

Arquitetura do Hadoop, seu ecossistema e suas soluções; comparação entre sistema de arquivos distribuídos do Hadoop (HDFS) e sistema de gerência de banco de dados relacionais (RDBMS); conceitos de data lake.

Prof. Sérgio Assunção Monteiro

Propósito

Compreender a tecnologia Hadoop e os aspectos fundamentais do seu ecossistema cria oportunidades concretas para o profissional de tecnologia da informação gerenciar uma infraestrutura complexa e desenvolver aplicações para uma das áreas que mais cresce na atualidade: Big Data.

Preparação

Para reproduzir os exemplos que apresentaremos ao longo deste material, você vai precisar do sistema operacional Windows e dos programas PyCharm e Java 8 instalados em seu computador. Além disso, também precisará do Hadoop.

Objetivos

- Analisar a arquitetura do Hadoop
- Comparar os sistemas HDFS e RDBMS
- Reconhecer conceitos de Data Lake

Introdução

As aplicações de Big Data estão em praticamente todos os setores da sociedade moderna: educação, economia, serviços comerciais e públicos, entretenimento e muitas outras áreas. Esses sistemas combinam características relacionadas ao volume, à variedade e à velocidade de crescimento e processamento dos dados, o que torna os processos de desenvolvimento e gerenciamento um grande desafio.

Nesse cenário de oportunidades e desafios, foi desenvolvido pela Apache Foundation o framework Hadoop, uma biblioteca de software que permite o processamento distribuído de grandes conjuntos de dados por meio de clusters de computadores usando modelos de programação simples. As primeiras aplicações do Hadoop ocorreram em empresas de internet que naturalmente possuem diversos problemas de Big Data; devido ao seu sucesso, ele foi aplicado em muitas outras áreas.

Ao longo deste conteúdo, estudaremos os conceitos relacionados à tecnologia do Hadoop e trataremos dos aspectos de sua arquitetura e de seu ecossistema. Além disso, desenvolveremos alguns exemplos práticos que nos ajudarão a entender o funcionamento do Hadoop e, assim, adquirir uma visão mais clara sobre suas aplicações e limitações.

Introdução e contextualização

Atualmente, muitas aplicações envolvem grandes volumes de dados, como as transações financeiras on-line, a produção e o compartilhamento de conteúdo nas redes sociais e os estudos nas áreas da biologia genética.

Esses são apenas alguns exemplos que nos ajudam a ilustrar como situações semelhantes a essas estão inseridas no nosso cotidiano. Essas aplicações fazem parte do que conhecemos como Big Data.

Esse termo da língua inglesa foi incorporada ao nosso dia a dia para descrever um conjunto de tecnologias que gerencia aplicações complexas associadas à expressão **5Vs**, que descreve as características fundamentais das aplicações de Big Data (ISHWARAPPA; ANURADHA, 2015):

Volume

Trata da quantidade de dados gerada e coletada pelas aplicações. Normalmente, uma aplicação é classificada como Big Data quando trabalha com um volume de dados da ordem de Petabytes (PB), sendo que um 1 PB corresponde a 1.024 Terabytes.

Variedade

Os dados são encontrados em diversos formatos, podendo ser estruturados e não estruturados. É bastante comum trabalhar nesse tipo de aplicação com dados disponíveis em tabelas, arquivos texto e JSON, por exemplo.

Velocidade

Essa característica está relacionada tanto com a velocidade com a qual os dados são gerados quanto com a que são processados.

Veracidade

Trata da questão fundamental da qualidade dos dados. Em especial, nesse tipo de aplicação, com tantas variáveis para se controlar, é muito importante aplicar técnicas e usar ferramentas para garantir a integridade e a qualidade dos dados e evitar processamentos desnecessários.

Valor

Essa característica está relacionada à recompensa que se espera obter ao trabalhar com aplicações de Big Data. Dados em grandes volumes são muito úteis em estudos estatísticos para descobrir padrões e adquirir conhecimento.

As tecnologias que compõem as aplicações de Big Data são modernas e ainda estão se expandindo. Isso ocorre no caso concreto da popularização da computação em nuvem e da internet das coisas, que aumentam ainda mais a abrangência de Big Data.

Para que possamos lidar com toda a complexidade que envolve as aplicações de Big Data, precisamos de uma tecnologia que gerencie todos esses recursos computacionais de hardware e software. Uma das tecnologias que obteve mais sucesso com essa finalidade é o framework Hadoop, nosso foco neste conteúdo.

A arquitetura Hadoop

O Hadoop é uma tecnologia de framework de software livre desenvolvida pela Apache Foundation, sendo aplicado no armazenamento e no processamento de dados de grandes volumes, ou seja, em Big Data. Além da distribuição livre da Apache, o Hadoop possui outras distribuições, como:

- Cloudera;
- Hortonworks;
- MapR;
- IBM;
- Microsoft Azure;
- Amazon Web Services Elastic MapReduce Hadoop Distribution.

Todos esses fornecedores têm suas soluções baseadas no Hadoop Apache, fator que analisaremos mais adiante.



[...] a tecnologia Hadoop possui um sistema de cluster que funciona basicamente como uma arquitetura mestre-escravo. Essa estrutura permite armazenar e processar grandes volumes de dados em paralelo.

(WHITE, 2015, n.p.)

As grandes empresas da internet, como Facebook, Yahoo, Google, Twitter e LinkedIn, entre outras, usam o Hadoop pela natureza de suas aplicações, ou seja, pelos diferentes tipos de dados. Esses dados podem ser:

- Estruturados, como tabelas e planilhas;
- Não estruturados, como logs, corpo de e-mails e texto de blogs;
- Semiestruturados, como metadados de arquivos de mídia, XML e HTML.



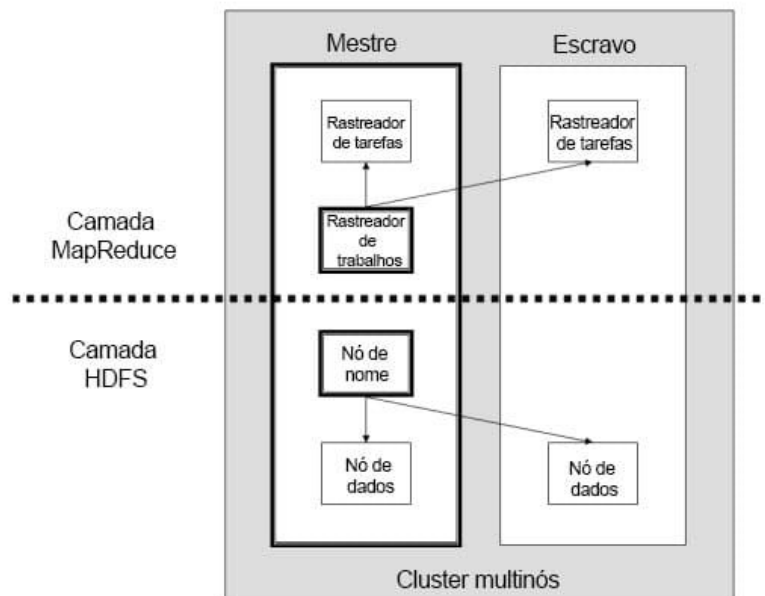
Resumindo

Essas aplicações possuem todas as características dos 5Vs de aplicações de Big Data. Isso ajudou bastante na divulgação e no aperfeiçoamento do framework Hadoop ao longo dos anos. Além disso, muitas organizações de grande porte, de modo geral, como empresas governamentais e prestadores de serviços financeiros e de saúde, por exemplo, também utilizam o Hadoop.

A arquitetura do Hadoop é formada por quatros componentes principais:

- MapReduce (modelo de programação paralela)
- HDFS (sistema de arquivos distribuídos do Hadoop)
- YARN (Yet Another Resource Negotiator)
- Utilitários comuns do Hadoop (Hadoop Common)

A imagem a seguir apresenta uma visão geral da arquitetura do Hadoop.



Visão geral da arquitetura do Hadoop

Ainda sobre a arquitetura do Hadoop, quando usamos o termo “cluster”, estamos nos referindo a um único nó do tipo mestre (*master*) com vários nós escravos (*slave*). O nó mestre inclui rastreador de trabalhos (*job tracker*), rastreador de tarefas (*task trackers*), Name Node e Data Node. Os nós escravos, por sua vez, possuem DataNode e Task Tracker. Vamos analisar esses componentes com mais detalhes.

MapReduce

Para realizar o processamento paralelo dos dados, o Hadoop utiliza um mecanismo chamado de MapReduce. Trata-se de uma estrutura que trabalha com duas etapas distintas.

Mapeamento dos dados

Essa etapa coleta os dados de entrada e os converte em um conjunto de dados que pode ser processado como um par do tipo valor e chave.

Redução dos dados

O resultado da fase de mapeamento é consumido pela tarefa de redução. Em seguida, os dados são processados até se atingir o objetivo da aplicação.

O MapReduce é a base fundamental para que o Hadoop possa ter um bom desempenho computacional. Quando se trabalha com aplicações de Big Data, não há como aplicar o processamento serial. Analisemos de forma mais detalhada as fases do MapReduce.

Fase de mapeamento

A entrada da fase de mapeamento é um conjunto de dados passado para a função Map. Essa função divide esses blocos de dados em tuplas, que são pares do tipo valor e chave. Tais pares são enviados para a fase de redução após a de mapeamento, que é dividida nas seguintes etapas:

Leitor de registros (record reader)

Divide os dados de entrada em pares de chave e valor para serem enviados como entrada para o mapeador. A chave a que nos referimos possui informações de localização, enquanto o valor são os dados associados a ela.

Mapeador

Função definida pelo usuário que faz o processamento das tuplas obtidas do leitor de registros. Essa função não gera nenhum par do tipo chave-valor.

Combinador (combiner)

É usado entre o mapeador e o redutor para reduzir o volume de transferência de dados entre ambos.

Particionador

Tem duas funções principais: buscar pares de chave-valor gerados nas fases do mapeador e gerar os fragmentos correspondentes a cada redutor.

Fase de redução

A função *reduce* combina as tuplas geradas pela fase de mapeamento e aplica as operações necessárias para processar os dados, que, em seguida, são enviados para o nó de saída final.

O processamento dos dados é sempre feito na fase de redução. Ela é dividida nas seguintes etapas:

Embaralhamento e classificação

O processo em que o mapeador gera os pares intermediários chave-valor e os transfere para a fase de redução é conhecido como embaralhamento (é bastante comum usar o termo em inglês *shuffle*). Por meio da aplicação do processo de embaralhamento, o sistema pode classificar os dados com o uso de seus pares chave-valor. Perceba aqui o benefício desse processo: à medida que algumas tarefas de mapeamento são concluídas, o embaralhamento já começa, ou seja, o processamento dos dados não espera pela conclusão da tarefa feita pelo mapeador.

Redução

Essa tarefa agrupa as tuplas geradas a partir do mapeamento e, em seguida, aplica os processos de classificação e agregação nesses pares de chaves e valores.

Gravação da saída

Quando todas as operações são executadas, os pares de chaves e valores são gravados em arquivo. Cada registro é gravado em uma nova linha, e a chave e o valor são separados por espaço.

HDFS (sistema de arquivos distribuídos Hadoop)

O sistema de arquivos distribuídos do Hadoop, mais conhecido como HDFS (*Hadoop Distributed File System*), é responsável pelo armazenamento de dados em um cluster do Hadoop. O HDFS foi projetado para trabalhar com grandes volumes de dados em hardware comum, isto é, em dispositivos baratos, como computadores pessoais.

Trata-se de um sistema que tem tolerância a falhas e, além disso, fornece alta disponibilidade para a camada de armazenamento e os outros dispositivos presentes nesse cluster do Hadoop.

A arquitetura do HDFS, por meio dos componentes NameNode e DataNode, utiliza um sistema de arquivos distribuídos que proporciona alto desempenho no acesso aos dados em clusters Hadoop. Esses arquivos podem ser expandidos, ou seja, são altamente escaláveis. Vamos entender os componentes do HDFS de maneira mais detalhada:

NameNode

Desempenha o papel de “mestre” em um cluster Hadoop que gerencia o Datanode (nós “escravos”). Ele tem como objetivo o armazenamento dos dados sobre os dados, ou seja, os metadados. Os logs de transações que servem para rastrear a atividade do usuário em um cluster Hadoop são um exemplo de metadados. Os NameNodes gerenciam os DataNodes por meio das operações de abrir, excluir, criar, replicar, renomear e fechar arquivos.



Metadados

Contêm informações sobre os arquivos, como nomes, tamanhos e informações sobre a localização (número ou ids do bloco) do DataNode em que o NameNode faz o armazenamento. Essas informações são úteis para encontrar o DataNode mais próximo. Como consequência, as operações de comunicação ficam mais rápidas.



DataNode

Desempenha o papel de “escravo”. A principal utilização dos DataNodes é armazenar os dados em um cluster Hadoop. A quantidade de DataNodes pode ser muito grande, aumentando, assim, a capacidade de armazenamento que o cluster Hadoop pode realizar.



YARN (Yet Another Resource Negotiator)

YARN é o componente estrutural sobre o qual funciona o MapReduce. Ele realiza duas operações distintas:

Agendamento de tarefas

O objetivo é dividir uma grande tarefa em tarefas menores para que elas possam ser atribuídas a vários nós escravos em um cluster do Hadoop. Dessa forma, o desempenho do processamento será maximizado. É o agendador de tarefas que faz o controle das prioridades de execução das tarefas, considerando aspectos, como sua importância e dependências em relação às demais tarefas e quaisquer outras informações, como o tempo para conclusão do trabalho, por exemplo.



Gerenciamento de recursos - ResourceManager

Faz o controle de todos os recursos disponibilizados para a execução de uma tarefa em um cluster Hadoop. Relacionado ao gerenciador de recursos está o gerenciador de dados (NodeManager), que é o agente de estrutura por máquina responsável pelos containers por meio do monitoramento do uso de recursos (CPU, memória, disco e rede) e do envio de relatórios de uso para o gerenciador de recursos.



Comentário

A ideia fundamental do YARN é dividir as funcionalidades de gerenciamento de recursos e agendamento/monitoramento de tarefas em daemons (processos que executam em background) separados.

Utilitários comuns do Hadoop (Hadoop Common)

Os utilitários comuns do Hadoop são as bibliotecas e aplicações que oferecem suporte para ele. Eles são usados para executar as aplicações no cluster Hadoop pelos componentes HDFS, YARN e MapReduce. De forma semelhante à que ocorre nos demais módulos do Hadoop, os utilitários assumem que as falhas de hardware são comuns e que devem ser tratadas automaticamente no software pelo Hadoop Framework.

Os utilitários Hadoop também são conhecidos como núcleo do Hadoop (Hadoop Core). De fato, o pacote de utilitários fornece serviços essenciais e processos básicos, como a abstração do sistema operacional e do seu respectivo sistema de arquivos.

Esse pacote também contém os arquivos Java Archive (JAR) e os scripts necessários para iniciar o Hadoop. Além disso, também podemos encontrar no pacote de utilitários códigos-fontes, documentação e informações sobre as contribuições de diferentes projetos da comunidade Hadoop.

Vantagens e desvantagens do Hadoop

Como vimos até agora, a arquitetura Hadoop é complexa: utilizá-lo para desenvolver um projeto exige bastante estudo. Por outro lado, as aplicações de Big Data fazem parte do nosso cotidiano; por isso, precisamos investir na compreensão das tecnologias para desenvolver soluções. Nesse sentido, vamos analisar as vantagens e as desvantagens da tecnologia do Hadoop para termos uma visão mais clara da aplicação dessa tecnologia na resolução de problemas práticos.

Vantagens do Hadoop

Entre as principais vantagens do Hadoop, temos:

Escalabilidade

O Hadoop foi projetado desde o início para trabalhar com grandes volumes de dados. Para isso, os componentes da sua arquitetura lidam com diferentes aspectos do armazenamento e do processamento de dados distribuídos em diferentes nós da infraestrutura que aplicamos na solução.

Redução de custos

A distribuição Apache do Hadoop é de um software livre. Além disso, ele não requer uma infraestrutura de hardware especial, podendo utilizar equipamentos comuns.

Flexibilidade

O Hadoop é capacitado para trabalhar com diferentes tipos de dados tanto estruturados como não estruturados. Dessa forma, as empresas podem aplicar suas estratégias para gerar valor a partir da composição e da análise desses dados.

Velocidade

Os componentes da arquitetura do Hadoop, como o HDFS e o MapReduce, são projetados respectivamente para gerenciar e processar dados com a aplicação de estruturas de dados e estratégias de algoritmos que otimizam a operação dos processos.

Tolerância a falhas

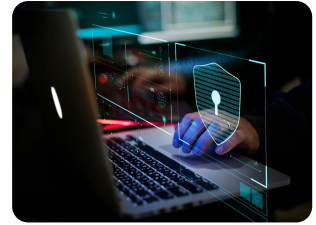
O Hadoop utiliza um processo de replicação dos dados entre os nós do cluster de modo que, se houver falha em algum nó, haverá outra cópia disponível para uso.

Desvantagens do Hadoop

Analisaremos agora algumas das principais desvantagens do Hadoop. Entre elas, estão:

Preocupações de segurança

Devido à complexidade das aplicações de Big Data de modo geral, os aspectos relacionados à segurança são um grande desafio. No caso do Hadoop, esse desafio está longe de ser trivial. Por exemplo, o modelo de segurança dele é desabilitado por padrão. Portanto, é da responsabilidade de quem vai gerenciar a infraestrutura da plataforma fazer a habilitação do módulo de segurança; caso contrário, os dados correrão um grande risco. Também é necessário tratar explicitamente de aspectos de criptografia dos dados.



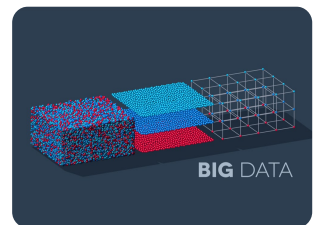
Vulnerabilidade intrínseca

O Hadoop foi desenvolvido na linguagem de programação Java. Existem diversos casos já catalogados de quebra de segurança do Hadoop, como escalonamento de privilégios e acesso não autorizado a senhas. Tudo isso ocorre devido à complexidade das aplicações de Big Data. O profissional que trabalha com os aspectos de segurança e controle de vulnerabilidades precisa conhecer muito bem a arquitetura do Hadoop e estudar constantemente os fóruns oficiais sobre esse tema, que é bastante dinâmico.



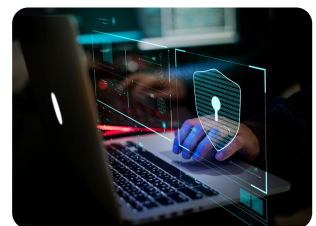
Não é adequado para dados pequenos

O Hadoop foi projetado para trabalhar com grandes volumes de dados. Infelizmente, isso significa que ele não é uma boa opção para trabalhar com pequenos volumes. Isso parece ser contraditório, mas, na verdade, não é. Os componentes HDFS e MapReduce utilizam técnicas eficientes para manipular muitos dados. Isso implica que as estruturas de dados e os algoritmos são dimensionados com essa finalidade e que, se essas técnicas forem usadas para trabalhar com pequenos volumes, serão ineficientes em relação a soluções mais simples.



Problemas de estabilidade

O Hadoop está em constante evolução e tem versões distribuídas por vários fornecedores. Por isso, não é raro que ocorram problemas relacionados à estabilidade da plataforma. Mais uma vez, isso reforça a necessidade de ter um profissional focado em aspectos da arquitetura e infraestrutura do Hadoop e que esteja se atualizando constantemente nos canais oficiais e fóruns de usuários e analistas.



Exemplo de MapReduce

Como vimos, o mecanismo de MapReduce é um componente fundamental na arquitetura do Hadoop para aplicar a computação distribuída e, assim, melhorar o desempenho dos programas.

Agora veremos um exemplo de programa desenvolvido em Python para ilustrar o funcionamento do MapReduce. Para isso, vamos precisar seguir estes passos:

- Instalar o Python e configurar o ambiente;
- Desenvolver o programa MapReduce;
- Executar o programa e analisar os resultados.

Instalar o Python e configurar o ambiente

O Python é uma linguagem de programação de propósito geral e pode ser utilizada para aplicações de bancos de dados, desenvolvimento web, internet das coisas e computação distribuída. Ele se tornou bastante popular, em especial, para aplicações de ciência de dados e de aprendizado de máquina. Trata-se de um software livre que possui bastante documentação disponível on-line, além de ter uma comunidade engajada que colabora na divulgação de soluções de problemas.

Para utilizar o Python, vamos empregar a IDE PyCharm Community. Ela está disponível no site da JetBrains (ver PyCharm). Nesse site, estão descritos inclusive os passos para fazer a instalação.

Depois de instalar o PyCharm, instalaremos o pacote mrjob. Para isso, basta abrir um terminal do Python e executar esta linha de comando:

```
python
pip install mrjob
```

O mrjob é uma biblioteca do Python que oferece recursos para escrever códigos MapReduce. Existe a facilidade de poder implementar código para as fases de mapeamento e de redução em uma única classe.

Desenvolver o programa MapReduce

Depois de instalar o mrjob, passamos a desenvolver nosso programa. O objetivo é receber um texto com diversas palavras e contar quantas vezes cada uma apareceu. Apresentamos a seguir o programa em Python:

```
python

from mrjob.job import MRJob
import re

palavra_regex = re.compile(r"[\w']+")

class QuantidadePalavras(MRJob):
    def mapper(self, _, linha):
        for p in palavra_regex.findall(linha):
            yield (p.lower(), 1)

    def reducer(self, p, qtd):
        yield (p, sum(qtd))

if __name__ == '__main__':
    QuantidadePalavras.run()
```

No início do programa, fazemos a importação da biblioteca mrjob e do pacote re. O mrjob, como vimos, é útil para desenvolver a aplicação MapReduce. Já o pacote re é usado para tratar expressões regulares, que são modelos computacionais muito úteis para representação de padrões e processamento de palavras.

Vamos entender com mais detalhes o que o código faz:

A linha de código a seguir recebe uma linha de texto como entrada e retorna uma lista de palavras sem os espaços.

```
python  
palavra_regex = re.compile(r"[\w']+")
```

Na próxima linha do programa, temos:

```
python  
class QuantidadePalavras(MRJob):
```

Trata-se de uma declaração para uma classe no Python. No caso, demos o nome da classe de “QuantidadePalavras”; além disso, ela herda características da classe “MRJob” da biblioteca que já havíamos instalado. Agora temos os métodos mapper e reducer, que são respectivamente responsáveis pela fase de mapeamento e redução.

No método mapper, processamos cada linha e geramos pares de palavras – que são as chaves – e do valor 1; no reducer, os pares do tipo chave e valor, obtendo a quantidade de vezes que a palavra apareceu no arquivo de entrada de dados;

Por fim, temos o seguinte bloco, que é responsável por iniciar o ciclo de vida do nosso processo:

```
python  
if __name__ == '__main__':  
    QuantidadePalavras.run()
```

Executar o programa e analisar os resultados

Para executar o nosso exemplo, vamos usar o arquivo “texto_exemplo.txt” com o seguinte conteúdo:

```
Este texto tem palavras repetidas  
mapreduce hadoop palavras  
palavras palavra hadoop texto python  
mrjob hadoop mapreduce hadoop mapreduce
```

Para executar o programa, devemos escrever e executar a linha de comando abaixo no terminal:

```
python  
python principal.py texto_exemplo.txt > qtd.txt
```

Precisamos que, no caso do nosso exemplo, o arquivo “texto_exemplo” esteja armazenado no mesmo local do arquivo “principal.py”. Quando executarmos a linha de comando, obteremos o arquivo de saída “qtd.txt” com o seguinte conteúdo:

```
python
```

```
'este' 1
'hadoop' 4
'mapreduce' 3
'mrjob' 1
'palavra' 1
'palavras' 3
'python' 1
'repetidas' 1
'tem' 1
'texto' 2
```

Que é exatamente o que queríamos obter: a quantidade de vezes que cada palavra apareceu no texto de entrada.

Visão geral do Hadoop

Neste vídeo, abordaremos os principais conceitos sobre o Hadoop, dando destaque para os aspectos da arquitetura.



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Vem que eu te explico!

Os vídeos a seguir abordam os assuntos mais relevantes do conteúdo que você acabou de estudar.

MapReduce



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

YARN (Yet Another Resource Negotiator)



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Verificando o aprendizado

Questão 1

As aplicações de Big Data fazem parte do nosso cotidiano. Essa afirmação é muito fácil de ser constatada. Até usuários que não conhecem aspectos da tecnologia, mas que já consumiram algum serviço da internet, já utilizaram, mesmo sem saber, serviços de Big Data. O Hadoop é uma tecnologia voltada exatamente para esse tipo de aplicação. Selecione a opção correta a respeito do Hadoop e das características dos projetos de Big Data.

A

Apesar da capacidade de tratar grandes volumes de dados, o framework Hadoop é limitado à infraestrutura de redes locais.

B

Um dos limitantes da utilização do Hadoop é que ele não pode ser usado para trabalhar com tabelas tradicionais de bancos de dados.

C

O Hadoop é um framework cuja distribuição é gratuita, estando voltado especificamente para tratar aplicações de grande volume de dados.

D

As apresentações de vídeos ao vivo nas redes sociais – conhecidas como lives – são uma das aplicações que podem ser tratadas por técnicas de Big Data.

E

O Hadoop foi desenvolvido em um modelo flexível em que todos os nós podem assumir papéis de mestre e escravo, dependendo do contexto da aplicação.



A alternativa D está correta.

O Hadoop Apache é um framework desenvolvido para trabalhar com aplicações de Big Data. Essas aplicações são caracterizadas pela complexidade que envolve seus dados. Uma dessas características é a variedade de dados, o que significa que eles podem ser estruturados, não estruturados e semiestruturados. O Hadoop está preparado para trabalhar com todas essas situações. No caso de vídeos, trata-se de uma situação de dados não estruturados.

Questão 2

As aplicações de Big Data são complexas. Isso ocorre devido à diversidade de tecnologias envolvidas para tratar de situações também complexas. Para o usuário final, o que importa é que a aplicação funcione bem. O Hadoop tem como objetivo tratar desse tipo de situação e dar transparência dessa complexidade para os usuários consumidores, porém, na prática, há vantagens e desvantagens no uso dele. Marque a opção correta a respeito das vantagens e desvantagens do Hadoop.

A

Projetos de Big Data precisam de muitos recursos computacionais e, portanto, podem ser muito caros. Devido à flexibilidade do Hadoop, não é necessário utilizar equipamentos especiais, o que ajuda a equilibrar a relação custo e benefício.

B

O fato de todas as distribuições do Hadoop terem licenças gratuitas reduz os custos dos projetos. Além disso, existe uma documentação com muitos exemplos que facilitam a aplicação do Hadoop.

C

O Hadoop é um framework muito complexo e que, apesar de ter algumas distribuições gratuitas, como no caso do Apache, não é uma boa escolha para a implementação de projetos de Big Data. O seu valor é histórico, pois foi uma das primeiras ferramentas de Big Data aplicadas para uso comercial.

D

Uma das vantagens do Hadoop é o seu sistema de armazenamento muito eficiente conhecido como MapReduce, que é capaz de gerenciar dados nos mais variados formatos.

E

O Hadoop é um framework muito seguro e fácil de ser configurado. Essas vantagens, aliadas aos baixos custos de implantação e manutenção, tornam o Hadoop uma das melhores escolhas para o desenvolvimento de projetos de Big Data.



A alternativa A está correta.

Projetos de Big Data são complexos. Tal complexidade se dá por vários motivos; por outro lado, o valor que pode ser extraído desses projetos justifica o investimento em tecnologias que otimizem a relação custo-benefício. Nesse sentido, o Hadoop contribui com uma arquitetura flexível que não exige a utilização de equipamentos especiais.

Introdução e contextualização

As aplicações de Big Data fazem parte do nosso cotidiano. Isso é um fato bem simples de se constatar ao observar as diversas situações práticas que demandam técnicas capazes de lidar com grandes volumes de dados, os quais podem crescer com grande velocidade e vir de diferentes fontes com formatos diversos.

Essas aplicações devem agregar valor à solução, com a identificação de padrões e a descoberta de conhecimento para dar suporte à tomada de decisão. Como vimos, o Hadoop é uma tecnologia de Big Data que possui uma arquitetura projetada para lidar com tais situações.

Os principais componentes da arquitetura do Hadoop são o sistema de gerenciamento de arquivos distribuídos, o HDFS e o mecanismo de processamento distribuído (conhecido como MapReduce).

Em relação ao HDFS, em especial, sabemos que ele pode tratar tanto de dados estruturados como dos não estruturados. Já os sistemas tradicionais de gerenciamento de dados, conhecidos como sistemas de gerenciamento de bancos de dados relacionais – do inglês *relational database management system*, embora sejam mais reconhecidos pela sigla RDBMS –, são mais restritos quanto ao gerenciamento dos dados que o HDFS.



Saiba mais

Essa restrição é um limitante para aplicações distribuídas, mas, por outro lado, é bastante segura no contexto em que se aplica. No caso do HDFS, sabemos que precisamos tratar da questão de segurança com bastante cuidado, pois o sistema pode conter vulnerabilidades que o expõem a ataques de cibercriminosos.

Muitas aplicações ainda usam RDBMS; no entanto, há muitas situações em que queremos migrar esses sistemas para um ambiente de Big Data, como o Hadoop, que oferece recursos para realizar esse processamento. Para isso, devemos conhecer a composição do ecossistema do Hadoop, que possui os seguintes componentes:

- HDFS: sistema de arquivos distribuídos do Hadoop;
- YARN: um negociador de recursos;
- MapReduce: processamento de dados usando programação paralela;
- Spark: processamento de dados na memória;
- PIG, HIVE: serviços de processamento de dados usando consulta (semelhante a SQL);
- HBase: banco de dados NoSQL;
- Mahout, Spark MLlib: aprendizado de máquina;
- Apache Drill: SQL no Hadoop;
- Zookeeper: gerenciamento de um cluster;
- Oozie: agendamento de trabalho;
- Flume, Sqoop: serviços de ingestão de dados;
- Solr & Lucene: pesquisa e indexação;
- Ambari: provisão, monitoramento e manutenção de um cluster.

Desses componentes, o Sqoop é o serviço do Hadoop para transferir dados de RDBMS para o HDFS. Ele pode fazer a importação e a exportação de dados de RDBMS para HDFS. Os dois processos são bem similares.

No caso da transferência de dados do RDBMS para o HDFS, quando realizamos o processamento do Sqoop, nossa tarefa principal é dividida em subtarefas. Essas subtarefas, por sua vez, são tratadas internamente como tarefas de mapa individuais.

Por fim, a subtarefa Map Task (tarefa de mapeamento) importa os dados para o ecossistema Hadoop. No caso do caminho contrário, ou seja, da transferência dos dados do HDFS para o RDBMS, o Sqoop é mapeado em tarefas de mapa que trazem o bloco de dados do HDFS. Esses blocos de dados são exportados para um destino de dados estruturados.

Esse processo é conhecido como ETL, que é a sigla para a expressão *extract, transform and load* (em português, extrair, transformar e carregar).

Diferenças entre o HDFS e o RDBMS

Os RDBMS são sistemas de gerenciamento de banco de dados relacionais. Como exemplos de sistemas RDBMS, temos o Oracle, o SQL Server da Microsoft, o MySQL e o PostgreSQL. Eles utilizam tabelas para fazer o armazenamento dos dados e das regras de integridade, que servem para relacionar as tabelas entre si e restringir as ações que podemos realizar sobre os dados.

Esses sistemas garantem as propriedades ACID das transações, que são:

A - Atomicidade

A execução das ações em um banco de dados, sequências de operações chamadas de transações, é indivisível.

C - Consistência

Um conceito fundamental em sistemas de banco de dados é o de integridade referencial, que trata das relações de dependências lógicas entre os dados. Então, ao final da execução de uma transação, o sistema deve manter a integridade dos dados, assim como respeitar todas as regras previamente definidas.

I - Isolamento

Faz a separação das execuções de transações para que elas não interfiram entre si e possam levar o sistema a um estado de inconsistência.

D - Durabilidade

Enquanto os dados são tratados pelas transações em execução, eles estão em um estado transiente. Ao final da execução da transação, o sistema deve garantir que os dados sejam gravados no banco de dados, isto é, que fiquem no estado de persistência.

Todas essas propriedades são fundamentais para um projeto de banco de dados. Portanto, podemos entender que os objetivos dos RDBMS são armazenar, gerenciar e recuperar os dados da forma mais rápida e confiável possível em um ambiente de arquitetura cliente-servidor.

No caso do HDFS, os dados estão contextualizados em um ambiente distribuído; devido às características intrínsecas das aplicações de Big Data, o gerenciamento deles é bem mais complexo. Isso nos mostra que há situações em que é mais adequado aplicar um modelo do que o outro, ou seja, o HDFS não é uma substituição do RDBMS.

Vamos analisar algumas das diferenças entre os dois sistemas em relação às seguintes características:

Volume de dados

O RDBMS funciona bem com o volume de dados na ordem de Gigabytes até Terabytes, enquanto o HDFS foi projetado para trabalhar com dados na ordem de Petabytes.

Taxa de transferência

Taxa de transferência significa o volume total de dados processados em determinado período. O RDBMS faz a operação de leitura muito rápida e a de escrita, lenta. Já o HDFS realiza as operações de leitura e escrita rapidamente.

Variedade de dados

O HDFS tem a capacidade de processar e armazenar dados estruturados, semiestruturados ou não estruturados. Já o RDBMS é usado apenas para gerenciar os estruturados e semiestruturados.

Tempo de resposta

Também conhecido como latência, tal tempo está relacionado à velocidade para recuperar informações. Apesar de o HDFS ter um rendimento alto para acessar lotes de grandes conjuntos de dados, o RDBMS é comparativamente mais rápido na recuperação das informações dos conjuntos de dados.

Escalabilidade

Para escalar uma solução no RDBMS, é necessário aumentar os recursos da máquina, processo chamado de escalabilidade vertical. No caso do HDFS, a escalabilidade é horizontal, ou seja, para expandir o sistema, basta adicionar mais computadores aos clusters existentes.

Processamento de dados

O Hadoop suporta OLAP (*online analytical processing*), que envolve consultas e agregações complexas. Dependendo da quantidade de dados, a velocidade de processamento de dados pode levar bastante tempo. No caso do RDBMS, as consultas e as agregações são feitas por meio do OLTP (*online transaction processing*), que comparativamente é mais rápido que o OLAP. Devemos ficar atentos para o fato de que, nas aplicações OLAP, as tabelas estão desnormalizadas, enquanto na OLTP elas estão normalizadas segundo as regras do modelo relacional de dados.

Custo

Além do Hadoop Apache ser um framework de software livre, ele permite aumentar a infraestrutura de um sistema sem a necessidade de equipamento especial. No caso do RDBMS, em muitos casos, temos de considerar questões relacionadas à licença de uso e aos custos de aquisição de recursos de hardware para a infraestrutura sobre a qual operam.

Instalação, configuração e teste do Hadoop

Vamos fazer a instalação e a configuração no Hadoop. Para isso, executaremos a sequência de passos abaixo:

1. Verificar os pré-requisitos de instalação;
2. Fazer o download do Hadoop;
3. Criar diretórios de dados;
4. Configurar arquivos do Hadoop;
5. Configurar a variável de ambiente do Hadoop;
6. Atualizar a pasta bin do Hadoop;
7. Verificar o funcionamento do Hadoop.

Pré-requisitos de instalação

Para utilizar o Hadoop, o Java 8 deve estar instalado. Além disso, a variável de ambiente JAVA_HOME tem de apontar para a pasta "bin" do Java.

Caso não tenha o Java instalado, vá para a página oficial do Java (ver Java Downloads). Escolha a versão adequada para o seu sistema operacional, baixe para sua máquina e faça a instalação. No final da instalação, não se esqueça de fazer a configuração da variável JAVA_HOME.

Para verificar se essa etapa funcionou corretamente, abra um prompt de comando e execute a linha abaixo:

```
plain-text  
java -version
```

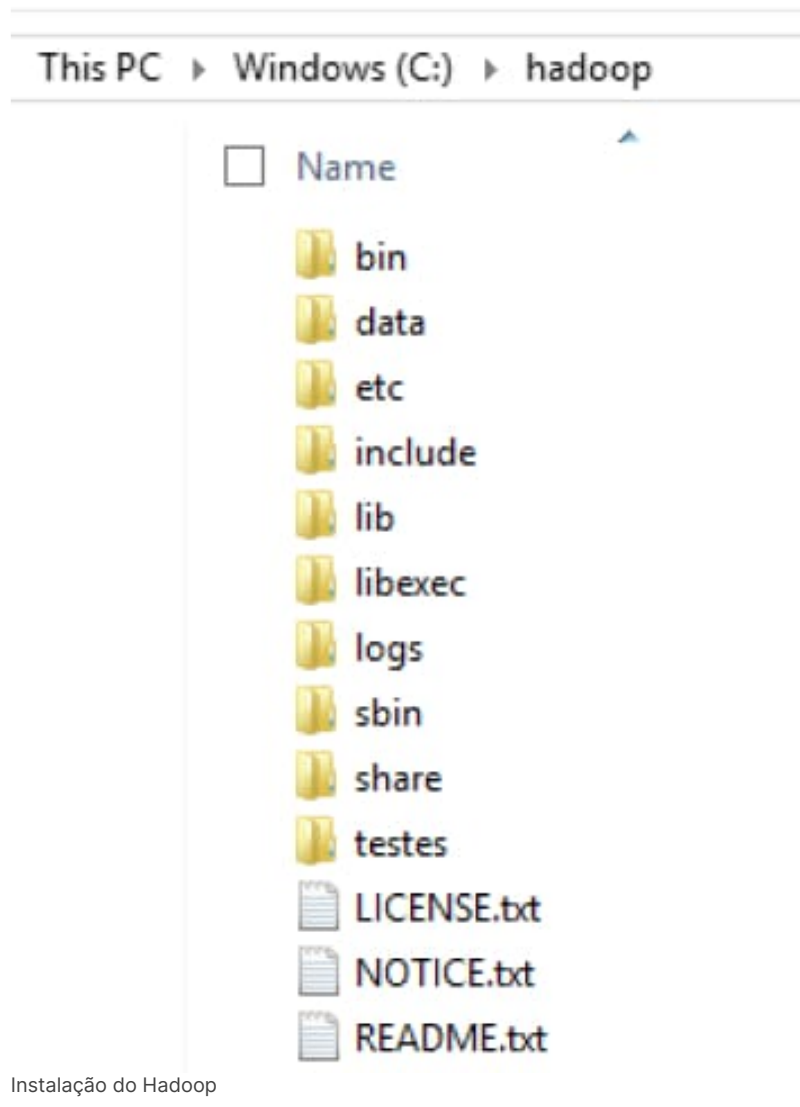
Se tudo funcionar corretamente, podemos seguir para a próxima etapa.

Fazer o download do Hadoop

A próxima etapa a ser realizada é baixar o Hadoop no site oficial do Apache. No nosso caso, escolheremos a versão 2.10.1, que pode ser baixada aqui: [Apache Hadoop Download](#).

Depois de salvar o arquivo, precisamos descompactá-lo. Em seguida, devemos copiá-lo para a pasta raiz do Windows. Além disso, temos de renomeá-lo para "hadoop" (sem as aspas).

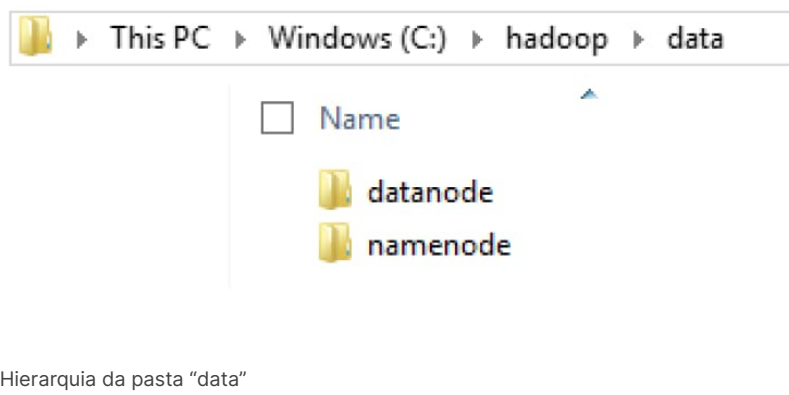
O resultado dessa etapa será o da imagem a seguir:



Criar diretórios de dados

O próximo passo é criar a pasta de dados. Para isso, vamos criar a pasta “data” dentro da pasta “hadoop”.

Em seguida, criaremos as pastas “datanode” e “namenode” dentro de “data”. Ao final, vamos obter este resultado:



Essas pastas são importantes, pois é dentro da pasta “datanode” que os arquivos do HDFS são hospedados.

Configurar arquivos do Hadoop

Agora vamos precisar configurar os arquivos do Hadoop. São eles:

mapred-site.xml

Lista de parâmetros para a configuração do MapReduce.

core-site.xml

Contém a configuração que substitui os parâmetros padrões do núcleo (core) do Hadoop.

hdfs-site.xml

Contém a definição de configuração para processos do HDFS, além de especificar a replicação de bloco padrão e a verificação de permissão no HDFS;

yarn-site.xml

Descreve as opções de configuração do YARN;

hadoop-env.cmd

Armazena as configurações globais usadas pelos comandos do Hadoop.

Esses arquivos estão localizados nesta pasta:

`C:\hadoop\etc\hadoop`

A configuração de cada arquivo deve ficar exatamente do modo como vamos mostrar.

Arquivo mapred-site.xml

Abra o arquivo “mapred-site.xml” e localize estas tags:

```
<configuration> </configuration>
```

Em seguida, dentro dessas tags de configuração, copie o código abaixo:

```
plain-text
```

```
mapreduce.framework.name  
yarn
```

Arquivo core-site.xml

Abra o arquivo “core-site.xml” e localize estas tags:

```
<configuration> </configuration>
```

Em seguida, dentro dessas tags de configuração, copie o código abaixo:

plain-text

```
fs.defaultFS
hdfs://localhost:9000
```

Arquivo hdfs-site.xml

Abra o arquivo “hdfs-site.xml” e localize estas tags:

`<configuration>` `</configuration>`

Em seguida, dentro dessas tags de configuração, copie o código abaixo:

plain-text

```
dfs.replication
1

dfs.namenode.name.dir
PATH TO NAMENODE

dfs.datanode.data.dir
PATH TO DATANODE
```

Arquivo yarn-site.xml

Abra o arquivo “yarn-site.xml” e localize estas tags:

`<configuration>` `</configuration>`

Em seguida, dentro dessas tags de configuração, copie o código abaixo:

plain-text

```
yarn.nodemanager.aux-services
mapreduce_shuffle

yarn.nodemanager.auxservices.mapreduce.shuffle.class
org.apache.hadoop.mapred.ShuffleHandler
```

Arquivo hadoop-env.cmd

Vamos verificar como a variável de ambiente JAVA_HOME está configurada. No nosso caso, usaremos a versão 1.8.0_131 do Java instalada nesta pasta:

`C:\Program Files\Java\jdk1.8.0_131`

Em seguida, copiaremos o nome do endereço e escreveremos uma das duas linhas no arquivo “hadoop-env.cmd”:

```
SET JAVA_HOME =% JAVA_HOME%
```

ou

```
SET JAVA_HOME = " C:\Progra~1\Java\jdk1.8.0_131\ "
```

Em nossa configuração, optaremos pela segunda linha. Realmente, precisamos escrever “Program Files” como “Progra~1” para evitar problemas com espaços.

Configurar a variável de ambiente do Hadoop

No painel de controle do Windows, vamos criar e configurar a variável de ambiente HADOOP_HOME para o endereço da pasta “bin” do Hadoop. No nosso caso, o endereço da variável HADOOP_HOME vai apontar para:

`c:\hadoop\bin`

Agora devemos configurar a variável de ambiente “Path”. Novamente acessaremos o painel de controle e selecionaremos a configuração de variáveis de ambiente.

Depois de selecionarmos a variável “Path”, devemos adicionar, ao final dela, a seguinte linha de comando:

```
;%HADOOP_HOME%
```

Esteja atento ao fato de que a variável de ambiente JAVA_HOME deve estar configurada desde o início do processo! Para testar se as variáveis de ambientes estão devidamente configuradas, basta abrir um terminal de comando – em qualquer pasta – e digitar os seguintes comandos:

```
plain-text
echo %JAVA_HOME%
```

e

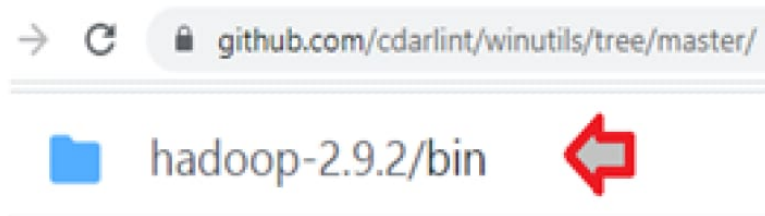
```
plain-text
echo %HADOOP_HOME%
```

Como resultado, será apresentado, conforme configuramos, o endereço completo do Java e do Hadoop.

Atualizar a pasta bin do Hadoop

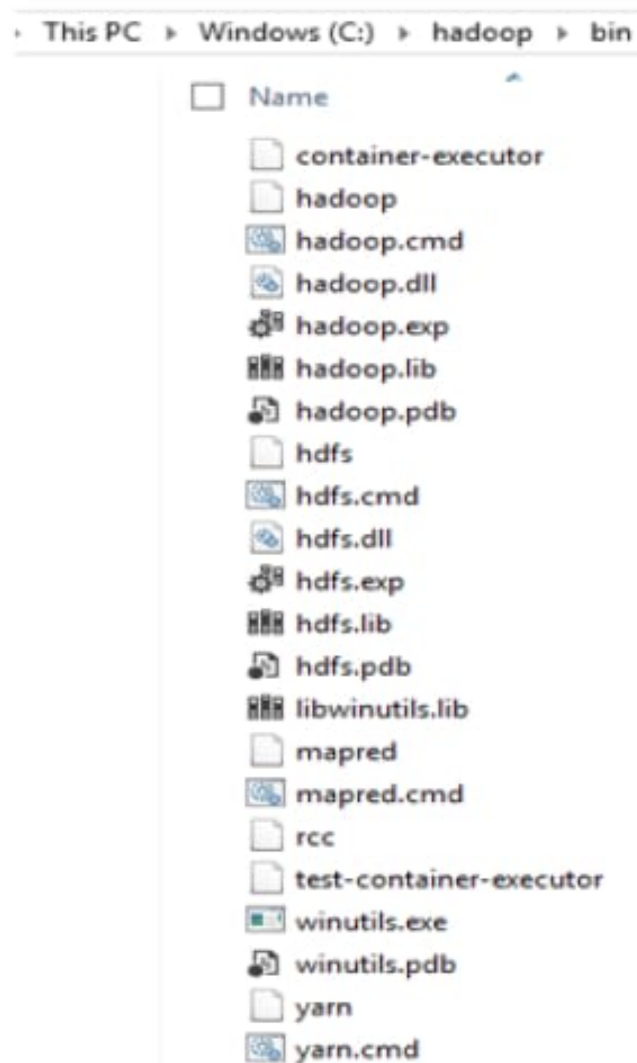
Nesta etapa, vamos copiar os arquivos do WinUtils, que são binários utilitários do Windows para o Hadoop. Esses arquivos estão hospedados na página do WinUtils, como demonstrado na imagem a seguir.

Na página do WinUtils, vamos baixar a versão do WinUtils compatível com a nossa instalação do Hadoop. No nosso caso, baixaremos a versão hadoop-2.9.2, como pode ser visto a seguir:



Versão do winutils

Em seguida, precisamos descompactar o pacote e copiar os arquivos – todos – para a pasta bin do Hadoop. A nossa pasta bin ficou assim:



Pasta bin atualizada com winutils.

Verificar o funcionamento do Hadoop

Agora provavelmente temos a parte mais aguardada: verificar se o Hadoop está funcionando.

Para isso, vamos seguir estes passos:

1. Formatar o namenode;
2. Iniciar o Hadoop;
3. Monitorar a execução do Hadoop.

Passo 1 - Formatar o namenode

Para começar o teste do Hadoop, vamos formatar o namenode. Para isso, devemos abrir um novo prompt de comando com o perfil de administrador e executar a linha de comando abaixo:

```
plain-text  
hadoop namenode -format
```

Dessa forma, formatamos todos os dados em namenode. Devemos usar esse comando apenas no início do processo para evitar a perda de dados.

Passo 2 - Iniciar o Hadoop

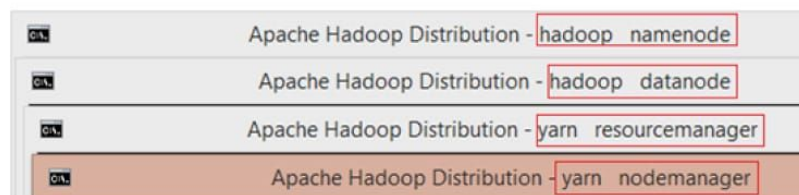
Agora podemos continuar no mesmo prompt de comando com o perfil de administrador para evitarmos problemas de permissão. Execute a linha de comando abaixo:

```
plain-text  
start-all.cmd
```

Se tudo funcionar corretamente, serão abertas automaticamente quatro janelas executando os seguintes *daemons* (programas que executam em background) do Hadoop:

- namenode
- datanode
- resourcemanager
- nodemanager

Na imagem a seguir, podemos ver a saída:



Daemons do Hadoop. A saída foi obtida com a execução do comando Hadoop: start-all.cmd

Passo 3 - Monitorar a execução do Hadoop

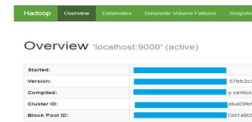
Vamos monitorar a execução dos serviços do Hadoop.

Namenode

Para monitorarmos o namenode, vamos abrir um browser e digitar este endereço:

localhost:50070

Podemos ver o resultado na imagem a seguir.



Resource manager

Vamos monitorar o resource manager. Para isso, devemos abrir um browser e digitar:

localhost:8088

Podemos ver o resultado na imagem a seguir.

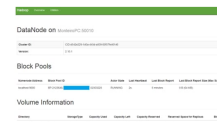


Datanode

Por fim, vamos monitorar o datanode. Para isso, devemos abrir um browser e digitar:

localhost:50075

Podemos ver o resultado na imagem a seguir.



Dicas importantes

Caso haja erro na execução do Hadoop, faça os seguintes procedimentos:

- Entre nas pastas do namenode e do datanode e exclua o conteúdo delas;
- Verifique a pasta tmp na raiz e elimine o conteúdo dela;
- Verifique se o conteúdo dos arquivos de configuração está exatamente como o que apresentamos.

HDFS versus RDBMS

No vídeo a seguir, abordaremos os conceitos sobre a tecnologia de HDFS e os compararemos com os sistemas de gerenciamento de banco de dados relacionais (RDBMS).



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Vem que eu te explico!

Os vídeos a seguir abordam os assuntos mais relevantes do conteúdo que você acabou de estudar.

Diferenças entre o HDFS e o RDBMS



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Diferenças entre o HDFS e o RDBMS



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Verificando o aprendizado

Questão 1

Os sistemas gerenciadores de bancos de dados relacionais, conhecidos pela sigla RDBMS, oferecem muitos recursos que garantem a eficiência e a confiabilidade dos dados. A partir da evolução das demandas por dados para os chamados problemas de Big Data, os RDBMS mostraram limitações. Dessa forma, foram necessárias outras soluções, como o Hadoop e o HDFS, por exemplo. Quanto às tecnologias de RDBMS e ao Hadoop/HDFS, marque a opção correta.

A

Devido à flexibilidade do Hadoop, ele pode trabalhar de forma equivalente com volumes de dados de grande e baixo porte, enquanto os RDBMS trabalham apenas com dados de baixo porte.

B

O HDFS é o sistema de gerenciamento de arquivos do Hadoop que o capacita a trabalhar com dados nos mais variados formatos, diferentemente do RDBMS, que trabalha apenas com dados relacionais.

C

Os sistemas do tipo RDBMS são muito eficientes para trabalhar com dados semiestruturados desde que estejam em um ambiente centralizado.

D

O sistema de gerenciamento de arquivos do Hadoop não está capacitado para trabalhar com tabelas relacionais. Caso um projeto demande dados estruturados, então ele deve usar um RDBMS.

E

Sistemas do tipo RDBMS são tão complexos quanto os sistemas de gerenciamento de arquivos do Hadoop. Basicamente o que os diferencia é o escopo da aplicação.



A alternativa B está correta.

Os problemas de Big Data são caracterizados, entre outras propriedades, pela variedade dos seus dados, que podem ser estruturados, semiestruturados e não estruturados. O HDFS, que é o sistema de gerenciamento de arquivos do Hadoop, foi projetado para trabalhar com essa variedade de dados. Já os sistemas do tipo RDBMS não têm essa característica, ficando limitados a trabalhar com tabelas em bancos de dados.

Questão 2

Os sistemas de gerenciamento de dados têm evoluído ao longo do tempo. Parte dessa evolução pode ser atribuída ao surgimento de tecnologias que os capacitam à resolução de problemas que até então não podiam ser tratados. Um exemplo concreto desse processo ocorre nos sistemas de gerenciamento de bancos de dados relacionais (RDBMS) e no sistema gerenciador de arquivos do Hadoop (HDFS). Em relação às tecnologias de RDBMS e HDFS, aponte a opção correta.

A

Considerado uma tecnologia antiga, o RDBMS deverá ser substituído pelo HDFS sempre que for possível.

B

Com a evolução da tecnologia, o RDBMS também evoluiu; atualmente, ele não apresenta nenhuma desvantagem em relação ao HDFS, a não ser para o tratamento de dados não estruturados.

C

Uma das vantagens do HDFS em relação ao RDBMS é a capacidade de expandir a infraestrutura de uma aplicação sem a necessidade de equipamentos especiais.

D

Um aspecto muito importante em um sistema de gerenciamento de dados é a velocidade com que realiza as operações de escrita e leitura dos dados, que são feitas mais eficazmente no RDBMS que no HDFS.

E

O RDMS e o HDFS são sistemas equivalentes para o gerenciamento de dados, diferenciando-se apenas pelo uso de tecnologias proprietárias, embora tenham os mesmos objetivos.



A alternativa C está correta.

O HDFS é o sistema de gerenciamento de arquivos do Hadoop com muitas vantagens em relação ao RDBMS, mas também possui desvantagens. Uma das vantagens está relacionada à capacidade de expansão de um sistema, que, no caso do HDFS, pode ser feita sem a necessidade de equipamentos especiais, o que é chamado de expansão horizontal. Já no caso do RDBMS, essa expansão é vertical, implicando maiores custos na aquisição de equipamentos.

Introdução e contextualização

As aplicações de Big Data são caracterizadas pelos 5Vs. Ainda existem outras definições que incluem outros “Vs”, como as propriedades de variabilidade e visualização. Isso ocorre porque essas aplicações ainda estão em processo de evolução.

Devido às características dessas aplicações, muitas soluções de Big Data exigem novas abordagens. Um desses casos ocorre em relação ao armazenamento dos dados. A solução aplicada pela tecnologia de Big Data é chamada de Data Lake.

O Data Lake – que, em português, seria traduzido como “lago de dados” – é um local da aplicação de Big Data que centraliza um grande volume de dados no formato original, sejam eles dados estruturados, não estruturados e até semiestruturados.

Esses dados são armazenados em objetos – conhecidos como *object storage* – que contêm tags de metadados e um identificador único. Essa estrutura de entidade dos dados permite que possamos analisá-los e buscar por padrões, pois as consultas são realizadas com bastante eficiência. Tais objetos de armazenamento podem ser consultados pelas demais aplicações de Big Data.

É natural compararmos os aspectos da arquitetura do Data Lake com os modelos dos bancos de dados tradicionais chamados de data warehouse (armazém de dados). A arquitetura do data warehouse é hierárquica, pois os dados são armazenados em arquivos ou pastas.

Já o Data Lake usa uma arquitetura plana por meio dos objetos de armazenamento de dados. De acordo com Singh e Ahmad (2019), os benefícios de seu uso são:

Escalabilidade

Para aumentar a capacidade de armazenamento do Data Lake, é necessário apenas acrescentar novos nós à infraestrutura, sem que haja a necessidade de um hardware especial.

Alta velocidade

O Data Lake usa programas utilitários para adquirir dados com alta velocidade e integrá-los com grandes volumes de dados históricos.

Estruturação

O Data Lake utiliza os objetos de armazenamento para guardar diferentes tipos de dados para que o acesso fique bastante eficiente.

Acesso aos dados originais

Os dados armazenados no Data Lake estão no formato original. Isso é considerado uma boa característica no processamento analítico (OLAP) para tentar descobrir padrões.

Acessibilidade

Os dados do Data Lake podem ser utilizados pelos usuários do sistema por meio de programas utilitários.

Aspectos essenciais do Data Lake

O Data Lake é composto por camadas e níveis. As camadas agrupam as funcionalidades comuns. São elas:

Camada de governança e segurança de dados

Responsável por controlar o acesso aos dados, esta camada utiliza mecanismos de segurança para que apenas usuários com perfis previamente mapeados tenham acesso e direito de manipulação dos dados.

Camada de metadados

Ela é responsável por marcar e identificar os dados com informações que auxiliem sobre a importância deles. Isso ajuda nos processos de análise para a extração de valor. Tal camada trata de dados nos diversos formatos: estruturados, não estruturados e semiestruturados.

Camada de gerenciamento do ciclo de vida da informação

Estabelece as regras de armazenamento dos dados e por quanto tempo devem permanecer no sistema, pois, com o tempo, é normal que eles percam seu valor.

Já os níveis são uma forma didática de agrupar aspectos semelhantes de uma funcionalidade. Podemos pensar da seguinte maneira: nos níveis, os dados fluem sequencialmente, como ocorre quando passamos parâmetros para uma função. Enquanto os dados se movem de camada para camada, as camadas realizam seu processamento nos dados em movimento.

1

Nível de admissão

Também conhecido como nível de ingestão, ele possui todos os serviços para a aquisição de dados de fontes externas. Esses dados podem vir em lotes (batch), microlotes (micro batch) e fluxo de dados de tempo real (real time streams).

2

Nível de gerenciamento

Aqui os dados são organizados por meio da aplicação de metadados e relacionamentos que auxiliem na sua identificação e localização.

3

Nível de consumo

Os dados do nível anterior são consumidos por aplicações relacionadas à análise de negócios.

Exemplo de execução no Hadoop

Vejamos um exemplo prático no Hadoop com um passo a passo. Para realizar os próximos passos, o Hadoop precisa estar instalado e configurado corretamente.

Em seguida, você deve:

1. Iniciar o Hadoop;
2. Criar um diretório no HDFS;
3. Copiar um arquivo no diretório do HDFS;
4. Verificar se o arquivo está no diretório do HDFS;
5. Verificar o conteúdo do arquivo;
6. Executar uma aplicação MapReduce;
7. Verificar o conteúdo do arquivo de saída;
8. Sair do Hadoop.

Iniciar o Hadoop

Primeiramente, abra um prompt de comando na pasta “sbin” onde o Hadoop foi instalado. Por exemplo, no nosso caso, o instalamos nesta pasta:

c:\hadoop

Execute a linha de comando abaixo:

```
plain-text
hadoop namenode -format
```

Em seguida, vamos iniciar a operação do Hadoop. Para isso, digite e execute este comando:

```
plain-text
start-all.cmd
```

Na imagem a seguir, podemos ver com mais detalhes como realizar esse passo.



```
C:\hadoop\sbin>start-all.cmd
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons
```

Iniciar o Hadoop no sbin

Em seguida, serão abertas, como vimos, quatro janelas. Não podemos fechá-las, pois vamos precisar que esses *daemons* estejam em execução para realizar as próximas etapas.

Criar um diretório no HDFS

Agora criaremos um diretório no HDFS onde vamos armazenar o nosso arquivo. Para criar o diretório, use o mesmo prompt de comando para digitar e executar o seguinte comando:

```
plain-text
hadoop fs -mkdir /dir_entrada
```

Na imagem a seguir, mostraremos, com mais detalhes, como realizar esse passo.



```
C:\hadoop\sbin>hadoop fs -mkdir /dir_entrada
```

Criação do diretório "dir_entrada".

Copiar um arquivo no diretório do HDFS

Vamos copiar um arquivo para o diretório que acabamos de criar no Hadoop. Para isso, criamos um arquivo texto chamado de “*texto_exemplo.txt*” (ele está no diretório “C”).

O conteúdo desse arquivo é:

*Este texto tem palavras repetidas,
como as palavras texto e muitas, por exemplo.
Dessa forma, o processamento do texto nos ajuda
a entender o pacote mrjob do python.*

Digite e execute o comando abaixo no prompt de comando:

```
plain-text  
hadoop fs -put C:/texto_exemplo.txt /dir_entrada
```

Verificar se o arquivo está no diretório do HDFS

Após copiarmos o arquivo para o diretório “dir_entrada” no HDFS, podemos verificar o conteúdo do diretório por meio da execução do comando:

```
plain-text  
hadoop fs -ls /dir_entrada
```

Na imagem abaixo, podemos ver a saída da execução com as informações sobre o arquivo que copiamos para o diretório “dir_entrada”.



```
C:\hadoop\sbin>hadoop fs -ls /dir_entrada  
Found 1 items  
-rw-r--r-- 1 supergroup 175 /dir_entrada/texto_exemplo.txt
```

Informações sobre o diretório “dir_entrada”

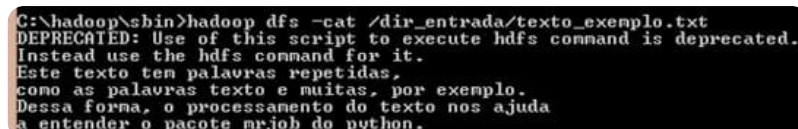
Verificar o conteúdo do arquivo

Já conseguimos visualizar o conteúdo do diretório que criamos. Agora você pode estar se perguntando: como vejo o conteúdo do arquivo dentro do diretório?

É simples! Para isso, basta digitar e executar este comando:

```
plain-text  
hadoop dfs -cat /dir_entrada/texto_exemplo.txt
```

Na imagem a seguir, conseguimos visualizar a saída.



```
C:\hadoop\sbin>hadoop dfs -cat /dir_entrada/texto_exemplo.txt  
DEPRECATED: Use of this script to execute hdfs command is deprecated.  
Instead use the hdfs command for it.  
Este texto tem palavras repetidas,  
como as palavras texto e muitas, por exemplo.  
Dessa forma, o processamento do texto nos ajuda  
a entender o pacote mrjob do python.
```

Conteúdo do arquivo de saída

Executar a aplicação MapReduce

O próximo passo é executar um programa no Hadoop. Para isso, devemos entrar neste diretório:

`"C:\hadoop\share\hadoop\mapreduce"`

E copiar o arquivo `"hadoop-mapreduce-examples-2.10.1.jar"` para a pasta `"C"`.

Agora basta executar esta linha de comando:

```
plain-text
```

```
hadoop jar C:/hadoop-mapreduce-examples-2.10.1.jar wordcount /dir_entrada /dir_saida
```

O resultado corresponde ao processamento do arquivo texto de entrada. Podemos ver o resultado na imagem a seguir.

A screenshot of a terminal window with a black background and white text. It displays the output of a Hadoop wordcount job. The statistics include: Reduce shuffle bytes=284, Reduce input records=23, Reduce output records=23, Spilled Records=46, Shuffled Maps =1, Failed Shuffles=0, Merged Map outputs=1, GC time elapsed (ms)=349, CPU time spent (ms)=3373, Physical memory (bytes) snapshot=427163648, Virtual memory (bytes) snapshot=597327872, and Total committed heap usage (bytes)=315097088. Under the 'Shuffle Errors' section, several error types are listed as 0: BAD_ID, CONNECTION, IO_ERROR, WRONG_LENGTH, WRONG_MAP, and WRONG_REDUCE. Finally, the 'File Input Format Counters' show Bytes Read=175, and the 'File Output Format Counters' show Bytes Written=186.

```
Reduce shuffle bytes=284
Reduce input records=23
Reduce output records=23
Spilled Records=46
Shuffled Maps =1
Failed Shuffles=0
Merged Map outputs=1
GC time elapsed (ms)=349
CPU time spent (ms)=3373
Physical memory (bytes) snapshot=427163648
Virtual memory (bytes) snapshot=597327872
Total committed heap usage (bytes)=315097088

Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0

File Input Format Counters
  Bytes Read=175
File Output Format Counters
  Bytes Written=186
```

Processamento do arquivo de entrada

Verificar o conteúdo do arquivo de saída

Para ver os detalhes do processamento do arquivo, basta executar a linha de comando:

```
plain-text
```

```
hadoop dfs -cat /dir_saida/*
```

Na imagem a seguir, apresentamos o resultado.

```

C:\hadoop\sbin>hadoop dfs -cat /dir_saida/*
DEPRECATED: Use of this script to execute hdfs
Instead use the hdfs command for it.
Dessa      1
Este       1
a          1
ajuda      1
as         1
como       1
do         2
e          1
entender   1
exemplo.   1
forma,     1
mrjob      1
muitas,    1
nos        1
o          2
pacote     1
palavras   2
por        1
processamento 1
python.    1
repetidas, 1
tem        1
texto      3

```

Detalhes do processamento do arquivo de entrada

Analisando a imagem, podemos ver a contagem de cada palavra do arquivo de entrada. Por exemplo, a palavra “processamento” tem apenas uma ocorrência no texto de entrada, enquanto “texto” aparece três vezes.

Sair do Hadoop

Conseguimos criar um diretório no HDFS do Hadoop. Também copiamos um arquivo para esse diretório e executamos um programa para processá-lo.

Agora vamos sair do Hadoop, excluir o arquivo do diretório que criamos no HDFS e excluir o próprio diretório. Para isso, execute a sequência de comandos a seguir no prompt de comando:

- Sair do Hadoop no modo de segurança:

```

plain-text
hadoop dfsadmin -safemode leave

```

- Excluir o arquivo do diretório no HDFS:

```

plain-text
hadoop fs -rm -r /dir_entrada/texto_exemplo.txt

```

- Excluir o diretório:

```

plain-text
hadoop fs -rm -r /dir_entrada

```

Algumas dicas e exemplo extra

Colocar o Hadoop para funcionar corretamente pode ser trabalhoso apesar dos passos que mostramos. Por isso, é preciso verificar vários aspectos desse processo.

Preparamos mais duas seções com dicas e mais um exemplo para ajudá-lo a consolidar melhor o conhecimento e obter mais segurança ao utilizar o Hadoop.

Dicas

O Hadoop exige bastante atenção, pois é preciso observar as muitas aplicações em execução simultaneamente, além das diversas parametrizações dos arquivos.

Desse modo, ao iniciar a execução de um exemplo, você deve observar com cuidado as seguintes situações:

- Verifique as versões do Hadoop e do Java instaladas no seu sistema.
- Verifique se as variáveis de ambiente JAVA_HOME e HADOOP_HOME estão configuradas corretamente.
- Garanta que o conteúdo dos diretórios “datanode” e “namenode” esteja vazio. Aqui cabe uma observação: só exclua esses dados se não for usá-los em outro momento. Tenha atenção redobrada com isso: você só poderá excluí-los se eles não forem realmente necessários.
- Verifique o conteúdo do diretório “tmp” que está na raiz de diretórios. Aqui vale o mesmo comentário do item anterior: se o conteúdo não for realmente necessário, ele deverá ser excluído.
- Os arquivos “jar” para aplicações de mapreduce normalmente estão neste endereço: C:\hadoop\share\hadoop\mapreduce.
- No começo da execução de um exemplo, lembre-se ainda de executar esta sequência de passos:

```
plain-text
```

```
hadoop namenode -format  
start-all.cmd
```

- Por fim, **nunca perca a paciência**. Se ainda assim tiver problemas, verifique se os arquivos estão configurados corretamente.

Exemplo extra

Vamos executar mais um exemplo de MapReduce. Para isso, siga todos os passos que mostramos anteriormente para iniciar o Hadoop.

Em seguida, copie o arquivo:

hadoop-mapreduce-examples-2.10.1.jar

Para o diretório:

C:/teste

Execute o programa no prompt de comando conforme a linha a seguir:

```
plain-text
```

```
hadoop jar C:/teste/hadoop-mapreduce-examples-2.10.1.jar
```

```
aggregatewordcount:
```

Programa de mapeamento e redução que conta as palavras nos arquivos de entrada;

aggregatewordhist:

Programa de mapeamento e redução baseado em agregação que calcula o histograma das palavras nos arquivos de entrada;

dbcount:

Programa que faz as contagens das visualizações de uma página de um banco de dados;

grep:

Programa de mapeamento e redução que conta as correspondências de um regex na entrada;

join:

Programa que efetua uma junção em conjuntos de dados classificados e igualmente particionados;

multifilewc:

Programa que conta palavras de vários arquivos;

randomtextwriter:

Programa que grava 10GB de dados textuais aleatórios por nó;

sudoku:

Um solucionador de sudoku;

wordmean:

Programa que calcula o comprimento médio das palavras nos arquivos de entrada;

wordmedian:

Programa que calcula a mediana do comprimento das palavras nos arquivos de entrada;

wordstandarddeviation:

Programa que calcula o desvio padrão do comprimento das palavras nos arquivos de entrada.

wordcount:

Programa que conta as palavras nos arquivos de entrada;

Agora vamos usar este parâmetro:

wordmean

A linha de comando ficará assim:

plain-text

```
hadoop jar C:/teste/hadoop-mapreduce-examples-2.10.1.jar wordmean /dir_entrada/  
texto_exemplo.txt /dir_saida2
```

Para ver o resultado da execução desse programa, digite e execute a linha de comando abaixo:

plain-text

```
hadoop dfs -cat /dir_saida2/*
```

Teremos como resultado a seguinte saída no terminal:

plain-text

```
C:\teste>hadoop dfs -cat /dir_saida2/*  
DEPRECATED: Use of this script to execute hdfs command is deprecated.  
Instead use the hdfs command for it.  
count    28  
length   138
```

O resultado contém a quantidade e a soma dos comprimentos da palavra do arquivo de entrada.

Aspectos fundamentais de um Data Lake

No vídeo a seguir, falaremos sobre os conceitos sobre um Data Lake, apontando ainda suas características.



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Vem que eu te explico!

Os vídeos a seguir abordam os assuntos mais relevantes do conteúdo que você acabou de estudar.

Aspectos Essenciais do Data Lake



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Dicas



Conteúdo interativo

Acesse a versão digital para assistir ao vídeo.

Verificando o aprendizado

Questão 1

Os dados das aplicações de Big Data têm muitas características especiais - entre elas, a variedade e o volume. Para gerenciar esses dados, o Hadoop utiliza o Data Lake, cuja tradução é lago de dados, o que dá uma boa ideia sobre seus objetivos. Marque a opção correta a respeito do Data Lake do Hadoop.

A

Os serviços que consomem dados no Data Lake garantem a qualidade deles, pois, antes de serem utilizados, eles passam por um processo de limpeza.

B

Antes que os dados fiquem salvos no Data Lake, é necessário que eles passem por um processo de transformação e padronização a fim de que possam ser acessados rapidamente nas próximas etapas.

C

Os dados que ficam no Data Lake não podem ser acessados rapidamente, pois o objetivo dessa etapa de gerenciamento de dados é a preservação dos dados originais.

D

O Data Lake é um repositório intermediário dos dados que tem como objetivo assegurar que apenas pessoas autorizadas tenham acesso a eles.

E

O Data Lake preserva os dados originais a fim de que eles possam ser analisados por diversos usuários consumidores por meio de programas com alto desempenho de resposta.



A alternativa E está correta.

O Hadoop disponibiliza programas utilitários que permitem que os dados sejam acessados com alta velocidade apesar do grande volume. Essa capacidade está associada às implementações de estruturas de dados que tornam esse processo muito eficiente.

Questão 2

O Data Lake é um componente fundamental da arquitetura do Hadoop. Os dados das aplicações de Big Data podem ter muitas características complexas que precisam ser tratadas. Selecione a opção correta a respeito do Data Lake.

A

Questões relacionadas à segurança da informação são tratadas por um programa utilitário específico do Data Lake que fica na camada de gerenciamento de ciclo de vida da informação.

B

O Data Lake pode ter dados estruturados, não estruturados e semiestruturados, mas o tratamento para cada um deles é realizado em diferentes camadas.

C

A ingestão dos dados é responsável por fazer marcações dos dados que posteriormente serão consumidos pelos programas utilitários do Hadoop de modo eficiente.

D

Um aspecto essencial em uma aplicação de Big Data é a consulta eficiente dos dados, que devem ser organizados para beneficiar as estruturas de dados de busca, o que ocorre na camada de metadados.

E

As camadas e os níveis do Data Lake são equivalentes em termos do tratamento dos dados, com apenas duas exceções, que são a implementação da política de segurança e a identificação dos dados.



A alternativa D está correta.

As camadas do Data Lake são as de governança, metadados e gerenciamento do ciclo de vida da informação. Cada uma delas é responsável por funcionalidades que visam a tratar a política de segurança de acesso, marcação dos dados e regras de permanência no sistema, que são atribuições respectivamente das camadas de governança, metadados e gerenciamento do ciclo de vida da informação.

Considerações finais

Neste conteúdo, estudamos a tecnologia Hadoop com ênfase em sua arquitetura e seu ecossistema. Também fizemos a instalação, a configuração e o monitoramento do Hadoop, além de testar alguns exemplos práticos de MapReduce.

Essa grande quantidade de conceitos e de tecnologias faz parte do universo das aplicações de Big Data. Trata-se de uma área dinâmica e que ainda pode se expandir muito. Certamente precisamos dominar os conceitos de todas as áreas do conhecimento e praticá-los, mas, nessa área, isso é mais sensível devido à dinâmica dos avanços que nela ocorrem.

O Hadoop é uma importante tecnologia de Big Data que ocupa um papel de destaque no mercado. Como vimos, ele pode ser aplicado em diferentes setores. Portanto, são muitas as oportunidades profissionais, como desenvolvimento de software, análise de dados, descoberta de conhecimento, infraestrutura e segurança da informação.

Podcast

Neste podcast, os principais conceitos relacionados ao Hadoop, com destaque para sua arquitetura, são abordados.



Conteúdo interativo

Acesse a versão digital para ouvir o áudio.

Explore +

Acesse o site oficial do Hadoop e aprofunde seu conhecimento nessa tecnologia.

Acesse o site da CVE (Common Vulnerabilities and Exposures) e pesquise sobre vulnerabilidades já catalogadas do Hadoop.

Referências

ISHWARAPPA, K.; ANURADHA, J. **A brief introduction on Big Data 5Vs characteristics and Hadoop technology.** Procedia computer science. v. 48. International Conference on Computer, Communication and Convergence. 2015. p. 319-324.

JETBRAINS. **PyCharm.** Consultado na internet em: 28 set. 2021.

ORACLE. **Java Downloads.** Consultado na internet em: 28 set. 2021.

SINGH, A.; AHMAD, S. **Architecture of Data Lake.** International journal of scientific research in computer science, engineering and information technology. v. 5. n. 2. 2019. p. 411-414.

THE APACHE SOFTWARE FOUNDATION. **Apache Hadoop Download.** Consultado na internet em 28 set. 2021.

WHITE, T. **Hadoop**: the definitive guide. O'Reilly Media/Yahoo! Press. Publicado em: abr. 2015.