Plataformas de Big Data

As plataformas de Big Data são conjuntos de tecnologias e ferramentas projetadas para lidar com volumes massivos de dados, geralmente chamados de "Big Data". Essas plataformas fornecem mecanismos para coletar, armazenar, processar, analisar e visualizar grandes volumes de dados de forma eficiente e escalável.

Uma característica fundamental das plataformas de Big Data é a capacidade de processar dados em paralelo, distribuindo tarefas de processamento em vários nós de computação em um cluster. Isso permite que os dados sejam processados em alta velocidade e com escalabilidade horizontal, aproveitando os recursos de vários servidores.

As plataformas de Big Data são construídas sobre uma infraestrutura distribuída e muitas vezes incluem componentes como:

1. Sistema de arquivos distribuídos: Permite armazenar e acessar dados em um ambiente distribuído. Exemplo: Hadoop Distributed File System (HDFS).
2. Frameworks de processamento distribuído: Fornecem uma camada de abstração para processar dados em paralelo. O Apache Hadoop MapReduce é um exemplo popular de framework de processamento distribuído.
3. Bancos de dados distribuídos: Oferecem recursos de armazenamento e consulta distribuídos para grandes volumes de dados. Exemplos incluem Apache HBase e Apache Cassandra.
4. Plataformas de processamento em tempo real: Permitem processar e analisar dados em tempo real. Exemplos populares são o Apache Storm e o Apache Flink.
5. Frameworks de análise distribuída: Permitem realizar análises complexas em conjuntos de dados distribuídos. O Apache Spark é um exemplo comumente utilizado.
6. Ferramentas de visualização e exploração de dados: Permitem visualizar e explorar grandes conjuntos de dados de forma interativa. Exemplos incluem Tableau e Apache Superset.

Essas plataformas de Big Data são amplamente utilizadas em diversas áreas, como ciência de dados, análise de negócios, pesquisa científica e tomada de decisões estratégicas. Elas oferecem recursos poderosos para lidar com os desafios do Big Data, como volume, velocidade, variedade e veracidade dos dados, permitindo extrair insights valiosos e obter vantagem competitiva.

Introdução ao ecossistema de Hadoop

O Hadoop é uma poderosa plataforma projetada para lidar com Big Data em um ambiente distribuído e com processamento paralelo. Composto por vários componentes interligados, o Hadoop oferece recursos essenciais para o gerenciamento eficiente de grandes volumes de dados. Entre os principais componentes do Hadoop, destacam-se o HDFS (Hadoop Distributed File System), o MapReduce e o YARN.

O HDFS, ou Hadoop Distributed File System, é responsável pelo armazenamento distribuído dos dados, permitindo que eles sejam divididos e replicados em diferentes nós de um cluster. Essa abordagem garante a redundância e a tolerância a falhas, essenciais para a confiabilidade e disponibilidade dos dados.

O MapReduce é um modelo de programação e processamento paralelo que permite a execução de tarefas em larga escala. Ele divide as tarefas em etapas de mapeamento e redução, distribuindo-as entre os nós do cluster para processamento simultâneo. Isso possibilita a rápida análise e transformação dos dados, facilitando a extração de insights valiosos.

O YARN (Yet Another Resource Negotiator) é um gerenciador de recursos computacionais em clusters. Ele controla e aloca os recursos disponíveis no cluster, garantindo que as tarefas do MapReduce e de outros aplicativos sejam executadas de forma eficiente e equilibrada. O YARN desempenha um papel fundamental na escalabilidade e no controle de recursos do Hadoop.

Com a combinação desses componentes, o Hadoop oferece uma solução robusta e escalável para o processamento de Big Data. Sua arquitetura distribuída e capacidade de processamento paralelo permitem lidar com grandes volumes de dados de forma eficiente, tornando-o amplamente utilizado em diversas áreas, como ciência de dados, análise de negócios e pesquisa científica.

Soluções comerciais de Hadoop

As soluções comerciais do Hadoop disponíveis atualmente oferecem uma ampla gama de recursos e funcionalidades para lidar com os desafios do processamento e análise de grandes volumes de dados. Essas soluções são desenvolvidas por empresas especializadas e geralmente oferecem pacotes completos que abrangem desde o armazenamento até a análise dos dados. Aqui está um resumo das principais soluções comerciais de Hadoop:

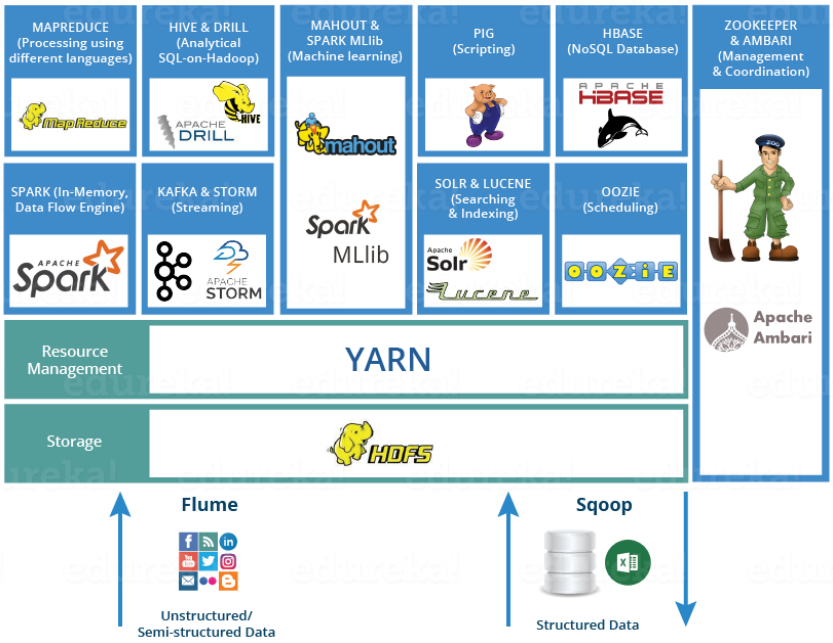
1. **Apache Hadoop**: O Apache Hadoop é uma plataforma de Big Data de código aberto que se tornou a base para muitas soluções comerciais. Ele oferece um ecossistema completo para o processamento e armazenamento distribuído de dados, incluindo o HDFS, MapReduce e YARN.
2. **Cloudera**: A Cloudera é uma das principais fornecedoras de soluções comerciais de Big Data. Sua plataforma inclui o Cloudera Distribution Hadoop (CDH), que oferece um conjunto abrangente de ferramentas para gerenciamento, processamento e análise de dados em larga escala. O Cloudera CDP (Cloudera Data Platform) é uma plataforma abrangente de Big Data desenvolvida pela Cloudera, uma das principais empresas no campo de soluções comerciais de Big Data. O CDP é projetado para permitir às organizações extrair valor de seus dados em escala, simplificando o gerenciamento e a análise de dados em ambientes híbridos e multicloud. O Cloudera CDP integra várias tecnologias-chave, incluindo Apache Hadoop, Apache Spark, Apache Hive, Apache HBase e outros componentes do ecossistema de Big Data. Essa integração fornece uma base sólida para o processamento, armazenamento e análise de grandes volumes de dados. Uma característica importante do Cloudera CDP é a capacidade de unificar diferentes ambientes de dados em uma única plataforma. Ele oferece uma experiência consistente de gerenciamento e governança de dados, independentemente de esses dados estarem localizados no local (on-premises), em nuvens públicas ou em ambientes híbridos. O CDP também oferece recursos avançados de segurança e governança, permitindo que as organizações protejam seus dados confidenciais e estejam em conformidade com regulamentações. Ele oferece controle granular de acesso aos dados, auditoria abrangente e criptografia para garantir a segurança dos dados em trânsito e em repouso.
3. **Hortonworks**: A Hortonworks é outra empresa líder em soluções de Big Data. Sua plataforma, o Hortonworks Data Platform (HDP), é construída com base no Hadoop e inclui recursos avançados de gerenciamento, segurança e integração de dados. A Hortonworks foi incorporada pela Cloudera.
4. **IBM BigInsights**: A IBM oferece o BigInsights, uma solução de Big Data que combina o poder do Hadoop com recursos adicionais, como análise preditiva e ferramentas de visualização de dados. Ele também inclui integração com outras tecnologias da IBM, como o IBM Watson.

5. **Amazon EMR**: A Amazon Web Services (AWS) oferece o Elastic MapReduce (EMR), uma solução de Big Data na nuvem. Ele permite o processamento escalável de dados usando o Hadoop, Spark e outras estruturas, facilitando a implantação e a escalabilidade dos clusters de Big Data.
6. **Microsoft Azure HDInsight**: O HDInsight é a oferta de Big Data da Microsoft no serviço Azure. Ele fornece uma plataforma gerenciada para processamento de Big Data usando o ecossistema Hadoop, Spark e outras tecnologias de código aberto.
7. **Google Cloud Dataproc**: O Cloud Dataproc é a solução de Big Data do Google Cloud. Ele permite o processamento e análise de grandes volumes de dados usando o Hadoop, Spark e outras ferramentas, com recursos de escalabilidade e integração com outros serviços do Google Cloud.

Essas são apenas algumas das soluções comerciais de Hadoop disponíveis no mercado. Cada uma delas tem seus recursos exclusivos, suporte adicional e integração com outras tecnologias, permitindo que as empresas escolham a solução mais adequada às suas necessidades de negócios e requisitos de dados.

Ecossistema Hadoop

O ecossistema Hadoop é um conjunto de ferramentas e tecnologias de código aberto projetado para processar e analisar grandes volumes de dados de forma distribuída e escalável. Ele é baseado em uma arquitetura distribuída e foi criado para lidar com os desafios do Big Data, como volume, velocidade, variedade e veracidade dos dados.



O Hadoop é composto por várias ferramentas interconectadas, sendo as principais:

1. **Hadoop Distributed File System (HDFS)**: É um sistema de arquivos distribuído projetado para armazenar grandes volumes de dados em clusters de servidores. O HDFS divide os dados em blocos que são replicados em vários nós do cluster, garantindo alta disponibilidade e tolerância a falhas.

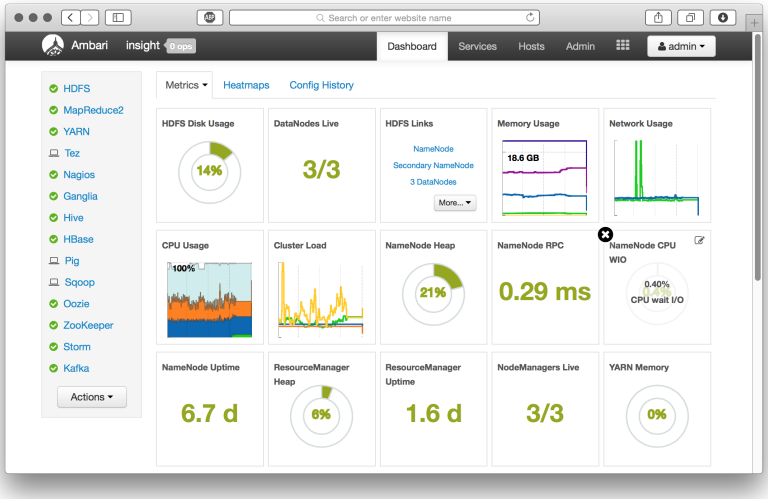
2. **MapReduce**: É um modelo de programação e um framework para processamento paralelo de grandes conjuntos de dados. O MapReduce divide as tarefas em etapas de mapeamento e redução, permitindo a execução distribuída e simultânea das tarefas em um cluster de servidores.
3. **YARN (Yet Another Resource Negotiator)**: É um gerenciador de recursos do Hadoop que permite a alocação eficiente de recursos de computação em um cluster. Ele é responsável por controlar e gerenciar a execução de aplicativos no cluster, garantindo a utilização otimizada dos recursos disponíveis.

Além dessas ferramentas principais, o ecossistema Hadoop inclui várias outras tecnologias e ferramentas que estendem suas capacidades e permitem diferentes casos de uso:

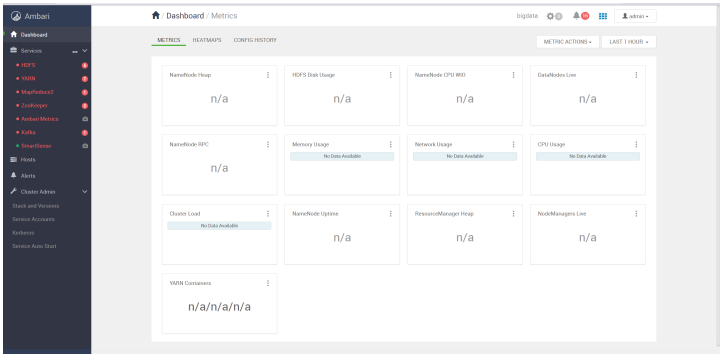
- **Apache Hive**: É uma camada de consulta de dados que permite executar consultas SQL-like no Hadoop, permitindo a análise de dados estruturados. Ele fornece uma abstração sobre o MapReduce, permitindo aos usuários consultar os dados usando uma linguagem familiar como o SQL.
- **Apache Pig**: É uma linguagem de script de alto nível para processamento de dados no Hadoop. Ele simplifica o desenvolvimento de tarefas complexas de processamento de dados ao fornecer uma sintaxe simples e expressiva. Esta ferramenta já não é muito usada.
- **Apache HBase**: É um banco de dados distribuído e escalável que fornece acesso aleatório e em tempo real a grandes volumes de dados no Hadoop. Ele é projetado para lidar com dados não estruturados e semiestruturados e é usado principalmente para aplicativos que exigem baixa latência de leitura e gravação.
- **Apache Spark**: É um framework de processamento de dados em memória que oferece velocidade e desempenho aprimorados em comparação com o MapReduce. O Spark suporta uma variedade de operações de processamento, incluindo processamento em lote, processamento em tempo real, aprendizado de máquina e processamento de gráficos.
- **Apache Kafka**: É uma plataforma de streaming distribuída usada para ingestão, armazenamento e processamento em tempo real de fluxos de dados em tempo real. O Kafka é altamente escalável e durável, permitindo a integração de fontes de dados em tempo real com os sistemas do Hadoop.
- **Tez**: O Apache Tez é um mecanismo de execução de alto desempenho projetado para processamento de dados no ecossistema Hadoop. Ele fornece uma estrutura flexível para executar tarefas complexas em clusters, otimizando a execução de consultas e fluxos de trabalho.
- **Superset**: O Apache Superset é uma plataforma de visualização de dados interativa e de código aberto. Ele permite explorar e visualizar dados de forma intuitiva, criando painéis interativos, gráficos e relatórios.
- **Flume**: O Apache Flume é uma ferramenta de ingestão de dados que permite coletar, agregar e mover grandes volumes de dados de forma confiável e escalável. Ele facilita a transferência de dados entre fontes de dados e destinos, tornando mais fácil a ingestão de dados no ecossistema Hadoop. Está ferramenta se encontra em desuso.
- **Sqoop**: O Apache Sqoop é uma ferramenta de importação/exportação de dados que facilita a transferência de dados entre bancos de dados relacionais e o Hadoop. Ele suporta a importação de dados do Hadoop para um banco de dados relacional e vice-versa.
- **Drill**: O Apache Drill é um mecanismo de consulta distribuído que permite executar consultas interativas e ad-hoc em diferentes fontes de dados, como Hadoop, bancos de dados NoSQL e sistemas de arquivos.

- **Solr**: O Apache Solr é uma plataforma de pesquisa e indexação de código aberto baseada no Apache Lucene. Ele fornece recursos avançados de pesquisa em texto completo e indexação para dados no Hadoop e outros sistemas.
- **Oozie**: O Apache Oozie é um sistema de agendamento e coordenação de fluxos de trabalho no ecossistema Hadoop. Ele permite definir e executar fluxos de trabalho complexos, que podem incluir várias tarefas, como processamento de dados, transferência de arquivos e execução de jobs.
- **ZooKeeper**: O Apache ZooKeeper é um serviço de coordenação e gerenciamento distribuído que fornece recursos de sincronização, eleição de líderes e gerenciamento de configurações para clusters distribuídos. Ele é amplamente usado para garantir a consistência e a confiabilidade dos serviços no ecossistema Hadoop.
- **Ambari**: O Apache Ambari é uma ferramenta de gerenciamento e provisionamento de clusters do Hadoop. Ele fornece uma interface gráfica e um conjunto de APIs para facilitar a instalação, configuração e monitoramento de clusters Hadoop.

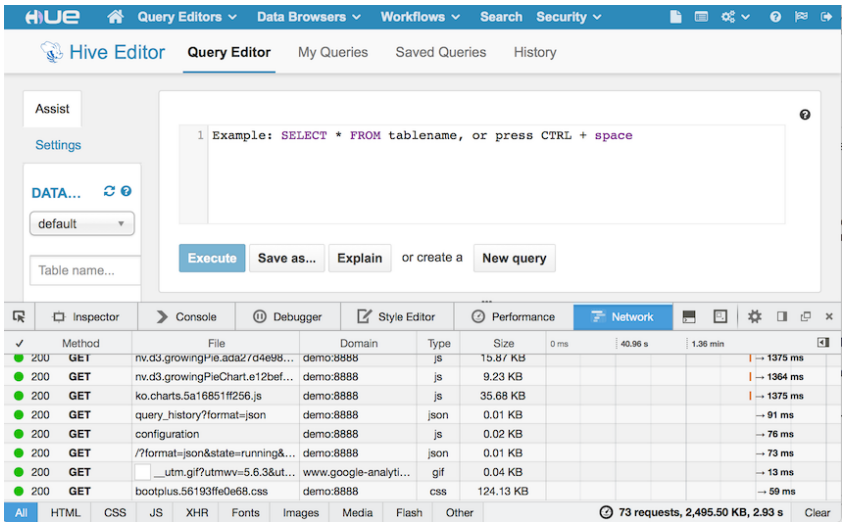
Ambari no HDP (Hortonworks Data Plataform) 2:



Ambari no HDP (Hortonworks Data Plataform) 3:



- **Hue**: O Apache Hue é uma interface de usuário para o ecossistema Hadoop que permite acessar e interagir com várias ferramentas e serviços, como Hive, Pig, Oozie e Impala. Ele oferece uma experiência simplificada para executar consultas, criar fluxos de trabalho e visualizar dados.



- **Impala**: O Apache Impala é um mecanismo de consulta de alto desempenho e de processamento em tempo real para o ecossistema Hadoop. Ele permite executar consultas SQL interativas diretamente em dados armazenados no Hadoop, oferecendo uma alternativa ao MapReduce para análise em tempo real.
- **Ranger**: O Apache Ranger é uma plataforma de segurança e governança de dados para o ecossistema Hadoop. Ele fornece recursos abrangentes de controle de acesso e políticas de segurança para proteger dados sensíveis e garantir conformidade com regulamentações. O Ranger oferece gerenciamento centralizado de políticas de acesso, auditoria de dados e integração com outros componentes do ecossistema Hadoop.
- **Atlas**: O Apache Atlas é uma plataforma de metadados para o ecossistema Hadoop. Ele permite catalogar, pesquisar e governar metadados de dados em toda a organização. O Atlas fornece uma visão abrangente dos ativos de dados, incluindo informações sobre a origem dos dados, seu significado, relacionamentos e impacto. Ele facilita a descoberta, rastreabilidade e governança de dados no ecossistema Hadoop.
- **Knox** é um componente do ecossistema Hadoop que fornece uma camada de segurança e acesso controlado para os serviços do Hadoop. Ele atua como um gateway de acesso seguro, permitindo que usuários externos acessem e interajam com os serviços do Hadoop de forma segura. O Knox oferece recursos de autenticação, autorização e auditoria para proteger o acesso aos serviços do Hadoop. Ele permite configurar políticas de segurança para controlar quais usuários ou grupos têm permissão para acessar cada serviço. Além disso, ele suporta diferentes métodos de autenticação, como Kerberos, LDAP e SAML, para garantir a autenticidade dos usuários. Ao usar o Knox, os administradores podem centralizar a segurança dos serviços do Hadoop em um único ponto de controle. Isso simplifica a administração e o gerenciamento de políticas de segurança, além de fornecer uma camada adicional de proteção para o ecossistema Hadoop. O Knox também fornece recursos de proxy reverso, que permitem ocultar a topologia e a infraestrutura subjacentes do Hadoop dos usuários externos. Isso ajuda a proteger a infraestrutura do Hadoop e a reduzir possíveis vulnerabilidades.
- **Phoenix**: O Apache Phoenix é uma camada de acesso a dados de código aberto para o ecossistema Hadoop que permite consultas rápidas e escaláveis em bancos de dados distribuídos. Ele fornece uma interface SQL para interagir com os dados armazenados no HBase, permitindo consultas em tempo real usando SQL padrão. O Phoenix otimiza as consultas SQL para aproveitar a estrutura de armazenamento do HBase, fornecendo desempenho e escalabilidade aprimorados para consultas analíticas e operacionais.

- **Storm**: O Apache Storm é uma plataforma de processamento de streaming em tempo real distribuída e tolerante a falhas. Ele permite a ingestão, processamento e análise de dados de streaming em tempo real em larga escala. O Storm é usado para casos de uso que exigem processamento contínuo de fluxos de dados, como detecção de fraudes, monitoramento de redes, análise de logs e processamento de eventos em tempo real.
- **Flink**: O Apache Flink é uma plataforma de processamento de dados em tempo real e em lote que oferece recursos de streaming de alto desempenho e processamento de dados em larga escala. Ele suporta a execução de fluxos contínuos de dados em tempo real e também pode processar dados em lote. O Flink oferece baixa latência, tolerância a falhas e recursos avançados de processamento, como janelas de tempo, agregações e suporte a algoritmos de aprendizado de máquina.
- **Zeppelin**: O Apache Zeppelin é uma ferramenta de notebook interativo de código aberto projetada para explorar, visualizar e colaborar em análise de dados. Ele fornece uma interface de usuário baseada em navegador que permite aos usuários escrever e executar código em várias linguagens, como Python, R, Scala e SQL, em um ambiente interativo. O Zeppelin é especialmente útil para análise de dados e ciência de dados, pois permite a criação de notebooks que combinam código executável, visualizações interativas, texto descritivo e gráficos em um único ambiente. Ele suporta a integração com várias ferramentas e tecnologias de Big Data, como Apache Spark, Hadoop, Hive, Pig, entre outros. Com o Zeppelin, os usuários podem escrever e executar trechos de código em tempo real, visualizar os resultados imediatamente e colaborar com outros usuários em notebooks compartilhados. Ele também oferece recursos avançados, como autocompletar código, histórico de comandos, geração de gráficos e exportação de resultados. Além disso, o Zeppelin permite a criação de painéis interativos, nos quais os usuários podem visualizar e explorar dados por meio de gráficos, tabelas dinâmicas e visualizações interativas. Esses painéis podem ser compartilhados e apresentados, facilitando a comunicação e a apresentação dos resultados da análise de dados. No geral, o Apache Zeppelin é uma ferramenta versátil para análise e exploração de dados, fornecendo um ambiente interativo e colaborativo para escrever código, executar consultas e criar visualizações interativas. Ele desempenha um papel fundamental na análise de dados, permitindo que os usuários interajam com dados de forma eficiente e intuitiva.

Essas ferramentas desempenham papéis específicos no ecossistema Hadoop, fornecendo recursos adicionais para processamento, análise, ingestão e gerenciamento de dados em larga escala.

Arquitetura do HDFS, YARN e MapReduce

A importância de Cloud Computing em Big Data

A ampla disponibilidade de enormes volumes de dados não estruturados a partir de diferentes domínios, tais como coleções de texto, imagens, vídeos e dados relacionados, criaram uma demanda para automatizar o processo de extração de informações a partir destas fontes. Neste cenário de "Big Data", o processamento de alto desempenho (PAD), aliado à técnicas de mineração de dados, está transformando o armazenamento, a manipulação e a análise de dados em tarefas mais baratas e mais rápidas. Isto faz com que grandes bases de dados possam ser consultadas pela comunidade científica e empresas, mudando a maneira como o conhecimento é produzido e como as empresas fazem negócios.

DADOS SÃO GERADOS CADA VEZ MAIS EM GRANDES VOLUMES POR DIFERENTES FONTES E EM DIFERENTES FORMATOS	O MUNDO ESTÁ CADA VEZ MENOR

O uso de algoritmos e computadores paralelos de alto desempenho tem permitido a redução do tempo de processamento dessas bases de dados, além de viabilizar seu armazenamento de maneira distribuída]. Isso tem permitido a "Recuperação de Informação" em dados não estruturados de maneira rápida, como é o caso, por exemplo, dos motores de busca (ex. Google). Além da busca, o uso de PAD tem tornado a "Mineração de Dados", isto é, a descoberta de padrões e conexões ocultas entre os dados, mais eficiente. Finalmente, o PAD tem contribuído para o refinamento de técnicas de "Aprendizagem de Máquina", permitindo a generalização de conhecimentos existentes com novos dados. Seu uso tem sido principalmente na engenharia de dados usadas no aprendizado e no uso combinado de várias técnicas (ensemble).

Quando o assunto é computação de alto desempenho, não é difícil pensarmos em servidores sofisticados e caros respondendo por este trabalho. No entanto, é possível obter resultados tão bons quanto ou superiores a partir de alguma solução de cluster - uma tecnologia capaz de fazer computadores mais simples trabalharem em conjunto, como se formassem uma máquina apenas. Cluster é o nome dado a um sistema que relaciona dois ou mais computadores para que estes trabalhem de maneira conjunta no intuito de processar uma determinada tarefa.

Os nós do cluster devem ser interconectados, preferencialmente, por uma tecnologia de rede conhecida, para fins de manutenção e controle de custos, como a Ethernet. É extremamente importante que o padrão adotado permita a inclusão ou a retirada de nós com o cluster em funcionamento, do contrário, o trabalho de remoção e substituição de um computador que apresenta problemas, por exemplo, faria a aplicação como um todo parar.

A computação em cluster se mostra muitas vezes como uma solução viável porque os nós podem até mesmo ser compostos por computadores simples, como PCs de desempenho mediano (“commodity”). Juntos, eles configuram um sistema de processamento com capacidade suficiente para dar conta de determinadas aplicações que, se fossem atendidas por supercomputadores ou servidores sofisticados, exigiriam investimentos muito maiores.

MapReduce

MapReduce é um modelo de programação desenhado para processar grandes volumes de dados em paralelo, dividindo o trabalho em um conjunto de tarefas independentes. Programas MapReduce são escritos em um determinado estilo influenciado por construções de programação funcionais, especificamente expressões idiomáticas para listas de processamento de dados. Este módulo explica a natureza do presente modelo de programação e como ela pode ser usada para escrever programas que são executados no ambiente Hadoop.

O MapReduce consiste das seguintes funções:

- **Map**: Responsável por receber os dados na forma de chave/valor representando de forma lógica cada registro dos dados de entrada, podendo ser, por exemlo, uma linha em um arquivo de log ou de uma tabela. A função map retorna uma lista com zero ou mais dados chave/valor e deve ser codificada pelo desenvolvedor, através de outras ferramentas ou da API Java;

- **Shuffle**: A etapa de shuffle é responsável por organizar o retorno da função Map, atribuindo para a entrada de cada Reduce todos os valores associados a uma mesma chave. Este etapa é realizada pela biblioteca do MapReduce;
- **Reduce**: Por fim, ao receber os dados de entrada a função Reduce retorna uma lista de chave/valor contendo zero ou mais registros, semelhante ao Map, deve ser codificada pelo desenvolvedor.

A arquitetura do MapReduce é semelhante ao do HDFS, master-slave. No MapReduce os componentes são:

- **JobTracker**: Ele recebe o job MapReduce e programa as tarefas map e reduce para execução, coordenando as atividades nos TaskTrackers;
- **TaskTracker**: Componente responsável por executar as tarefas de map e reduce e informar o progresso das atividades.

Características do MapReduce

- **Flexibilidade** – processa todos os dados independente do tipo e formato;
- **Confiabilidade** – permite que os Jobs sejam executados em paralelo;
- **Acessibilidade** – suporte a diversas linguagens de programação.

HDFS

Componente de storage do Hadoop, otimizado para alto throughtput e que funciona melhor na leitura e escrita de grande arquivos (Gigabytes para cima). Os dados no HDFS podem estar ou não estruturados.

Dentre as características do HDFS estão a escalabilidade e disponibilidade graças a replicação de dados e tolerância a falhas. O HDFS replica os arquivos um número configurado de vezes e é tolerante a falhas tanto em hardware quanto em software.

O HDFS segue a arquitetura master-slave. Seus dois componentes principais são:

- **NameNode**: tem como responsabilidade gerenciar os arquivos armazenados no HDFS. Suas funções incluem mapear a localização, realizar a divisão dos arquivos em blocos, encaminhar os blocos aos nós escravos, obter os metadados dos arquivos e controlar a localização de suas réplicas. Como o NameNode é constantemente acessado, por questões de desempenho, ele mantém todas as suas informações em memória. Todas as operações de manipulação de arquivos ou pastas do HDFS são tratadas pelo NameNode. Isso porque é ele quem contém as informações de em quais nós ao longo do cluster estão cada segmento de arquivo salvo no HDFS, bem como os diretórios existentes e suas permissões de acesso. Como todo dado salvo no HDFS é persistido em arquivo (binário ou texto), arquivos representando grandes DataSets (como, por exemplo, total de mensagens trocadas entre usuários de uma rede social) logo excederiam a capacidade física do HD. Para resolver tal desafio,o NameNode divide o arquivo em blocos de tamanho predefinido e define para quais DataNodes cada um desses segmentos será enviado. Além disso, a configuração do NameNode também define a quantidade de réplicas de cada bloco para mitigar possíveis falhas de nós. O NameNode definirá quais nós receberão cada uma das réplicas, balanceando a carga de uso de cada nó e levando em conta também o segmento de rede em que cada nó se encontra.É importante ressaltar que o NameNode em momento algum manipula os dados persistidos no HDFS. Ele apenas coordena as operações a serem realizadas, provendo informações e instruções para que os programas clientes possam

realizar essas operações diretamente com os nós onde os dados estão sendo gravados (ou de onde estão sendo lidos).

- **DataNode**: enquanto o NameNode gerencia os blocos de arquivos, são os DataNodes que efetivamente realizam o armazenamento dos dados. Como o HDFS é um sistema de arquivos distribuído, é comum a existência de diversas instâncias do DataNode em uma aplicação Hadoop, para que eles possam distribuir os blocos de arquivos em diversas máquinas. Um DataNode poderá armazenar múltiplos blocos, inclusive de diferentes arquivos. Além de armazenar, eles precisam se reportar constantemente ao NameNode, informando quais blocos estão guardando bem como todas as alterações realizadas localmente nesses blocos.
- **SecondaryNameNode**: utilizado para auxiliar o NameNode a manter seu serviço. Sua função é realizar pontos de checagem (checkpointing) do NameNode em intervalos pré-definidos, de modo a garantir a sua recuperação e atenuar o seu tempo de reinicialização. O SecondaryNameNode é o processo responsável por sincronizar os arquivos EditLogs com a imagem do FsImage, para gerar um novo FsImage mais atualizado. Essa atividade é executada pelo SecondaryNameNode de forma agendada, definindo-se um intervalo de tempo em segundos no arquivo core-site.xml. Também pode ser disparada sempre que o total de edições atingir um limite de tamanho em bytes, predeterminado no arquivo de configuração. O fato de o SecondaryNameNode ter que carregar todo o FsImage em memória, mais as edições, para combiná-los em um novo fsimage acarreta em um consumo elevado de RAM. Por este motivo, ele foi projetado para executar como um processo separado do NameNode, evitando que essas atividades pudessem comprometer o desempenho do NameNode. Em clusters grandes, é altamente recomendado manter o SecondaryNameNode em uma máquina exclusiva. É importante observar que o nome SecondaryNameNode pode induzir ao erro de se concluir que o este processo seja um NameNode de backup, ou mesmo uma redundância do serviço do NameNode, para o caso de uma eventual indisponibilidade. Tal avaliação está errada, não há no Hadoop serviço de redundância ou backup do NameNode (até a versão 2). Tais estratégias devem ser planejadas e implementadas sob demanda, pelo administrador do cluster. Para evitar essa confusão de conceitos, a versão 0.21 do Hadoop passou a denominá-lo como CheckpointNode.
- **NameNode Standby**: é um componente-chave no Hadoop que desempenha um papel fundamental na alta disponibilidade do Hadoop Distributed File System (HDFS). Sua função principal é fornecer uma cópia de backup do NameNode ativo, que é o principal ponto de controle para o sistema de arquivos distribuído. O NameNode é responsável por manter o namespace do HDFS, incluindo informações sobre a localização física dos blocos de dados e as permissões de acesso aos arquivos. No entanto, se o NameNode ativo falhar, isso resultaria em uma interrupção do serviço, pois todas as operações no sistema de arquivos dependeriam dele. Para mitigar esse risco, o HDFS implementa o conceito de NameNode Standby. O NameNode Standby é uma réplica em espera que mantém uma cópia atualizada dos metadados do HDFS em sincronia com o NameNode ativo. Ele monitora constantemente a saúde do NameNode ativo e pode assumir suas responsabilidades caso ocorra uma falha. Quando o NameNode ativo falha, o NameNode Standby é ativado e se torna o novo NameNode ativo. Ele assume a responsabilidade de responder às solicitações de clientes, manter os metadados do HDFS atualizados e coordenar as operações de leitura e gravação no sistema de arquivos.O uso do NameNode Standby no Hadoop garante que, mesmo em caso de falha do NameNode ativo, o sistema de arquivos continue operando sem interrupção significativa. Isso fornece alta disponibilidade e confiabilidade ao HDFS, essenciais para aplicações e serviços que dependem do armazenamento e processamento distribuídos fornecidos pelo Hadoop.

Existem arquivos, que devido ao seu grande tamanho, não podem ser armazenados em um único disco rígido. Esta situação fica mais evidente quando nos referimos aos arquivos de aplicações “Big Data”. Uma alternativa nesse caso é criar uma forma de dividir esses grandes arquivos e distribuí-los em um cluster de máquinas. Embora funcional, essa tarefa seria muito difícil de ser implementada sem a utilização de um recurso específico, como o HDFS.

O HDFS adota a estratégia de que antes de armazenar os arquivos, estes sejam submetidos a um procedimento de divisão em uma sequência de blocos de tamanho fixo. O tamanho padrão definido no framework atualmente é 128 Mb, podendo ser alterado se necessário. Esse tamanho é muito superior ao dos sistemas de arquivos tradicionais, que usa blocos de 512 bytes. Dessa forma, somente depois de dividido é que esses arquivos são distribuídos para os diversos nós escravos.

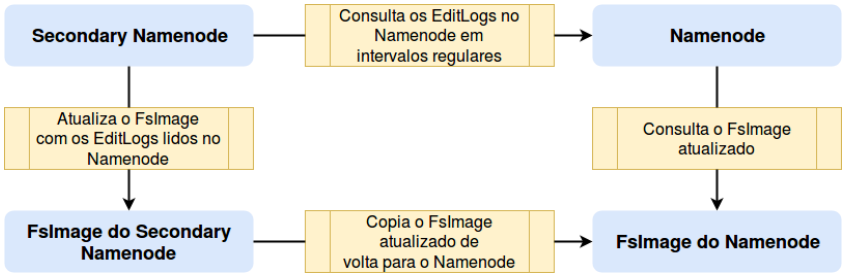
Além de dividir os arquivos em blocos, o HDFS ainda replica esses blocos na tentativa de aumentar a segurança. Por padrão, um bloco do HDFS possui três réplicas alocadas em diferentes nós, podendo essa quantidade ser configurada. Ainda existe uma recomendação, por questão de confiabilidade e desempenho, de alocar duas réplicas no mesmo rack, porém em nós distintos, e a outra réplica em um rack diferente. Como tipicamente a velocidade de comunicação entre máquinas de um mesmo rack é maior que em racks diferentes, por questão de desempenho, no momento de selecionar uma réplica para ser substituída em um processo, o HDFS dá preferência à réplica pertencente ao mesmo rack.

O maior benefício com a replicação é a obtenção de maior tolerância a falhas e confiabilidade dos dados, pois no caso de um nó escravo vir a falhar, o processamento passará a ser feito por outra máquina que contenha a réplica desse bloco, sem haver a necessidade de transferência de dados e a interrupção da execução da aplicação. Tudo isso é feito de forma transparente, pois o Hadoop oferece mecanismos para reiniciar o processamento sem que os demais nós percebam a falha ocorrida. No contexto de uma falha ocorrerá uma diminuição da quantidade de réplicas de um bloco. Então, para retomar a sua margem de confiabilidade, o NameNode consulta os metadados sobre os DataNodes falhos e reinicia o processo de replicação em outros DataNodes, para garantir o seu fator mínimo.

Arquivos FsImage e EditLogs

Imagine o seguinte exemplo no qual você tem arquivos relacionados a finanças e marketing e você deseja armazená-los no HDFS. Logo, será necessário criar 2 diretórios: uma para finanças e outro para marketing. Você colocará os arquivos de finanças no diretório finanças e os de marketing no diretório marketing.

Suponha que você tenha o arquivo finanças.txt no diretório /finanças/finanças.txt. No HDFS, o arquivo finanças.txt será armazenado em blocos em diferentes nos Datanodes. No entanto, colocando este arquivo no diretório /finanças você está apenas definindo uma localização lógica para o arquivo com o intuito de manter fácil a organização e gerenciamento dos arquivosl. Logo, o diretório ou namespace /finanças/finanças.txt é apenas uma abstração que agrupa os arquivos de finanças juntos, mas na realidade os blocos dos dados estão separados fisicamente no cluster HDFS.



O endereçamento dos dados que se encontram em cada nó ao longo do cluster é mantido em um arquivo do NameNode chamado FsImage. Por motivos de desempenho, quando o HDFS é iniciado, as informações do FsImage são carregadas para a memória do NameNode, evitando a necessidade de se realizar leitura de arquivo em disco sempre que tiver que tratar requisições de clientes.

Conforme o estado do HDFS vai sendo alterado (novos arquivos vão sendo inseridos, outros removidos, diretórios são alterados, arquivos têm dados acrescentados, nós inteiros são removidos ou adicionados ao cluster), as informações sobre que nó contém quais blocos também vai sendo atualizada na memória do NameNode, deixando o FsImage desatualizado. Para que essa diferença entre a memória do NameNode e o FsImage não seja perdido para sempre no caso de um desligamento, o estado do NameNode é sempre mantido também em arquivos temporários, chamados EditLogs (simplesmente escrever cada nova operação concorrente em um novo arquivo é menos custoso do que tentar sincronizar toda vez o mesmo arquivo).

Com o passar do tempo, o número de arquivos EditLog pode se tornar grande demais, tornando-se necessária uma atualização do fsimage. Essa é a função do SecondaryNameNode.

Introdução ao armazenamento de dados em Big Data

Nos últimos anos, o mundo tem testemunhado uma explosão sem precedentes no volume de dados gerados por várias fontes, como dispositivos conectados, mídias sociais, transações comerciais e sensores. Essa imensa quantidade de informações, conhecida como Big Data, apresenta um enorme potencial para revelar insights valiosos e impulsionar a inovação em diversos setores.

No entanto, lidar com Big Data requer uma abordagem diferenciada em relação ao armazenamento e gerenciamento de dados convencionais. Os sistemas tradicionais de armazenamento e bancos de dados são inadequados para lidar com o volume, a velocidade e a variedade dos dados do Big Data. É aqui que entra o armazenamento de dados em Big Data.

O armazenamento de dados em Big Data é um conjunto de tecnologias, técnicas e práticas que permitem capturar, armazenar, processar e analisar grandes volumes de dados de forma eficiente e escalável. Isso envolve a utilização de sistemas distribuídos que dividem os dados em várias máquinas interconectadas, permitindo o processamento paralelo e a capacidade de lidar com grandes cargas de trabalho.

Uma das principais características do armazenamento de dados em Big Data é a capacidade de lidar com dados não estruturados, semiestruturados e em tempo real. Esses dados podem incluir texto, imagens, vídeos, áudios e informações geradas por sensores. Os sistemas de armazenamento de Big Data são projetados para armazenar esses tipos de dados e facilitar sua análise posterior.

Além disso, o armazenamento de dados em Big Data também oferece mecanismos avançados de indexação, compressão, replicação e recuperação de dados, garantindo a integridade e a disponibilidade dos dados em todo o processo de armazenamento e análise.

Em suma, o armazenamento de dados em Big Data é essencial para enfrentar os desafios impostos pelo volume, velocidade e variedade dos dados gerados atualmente. Essa abordagem inovadora permite extrair insights significativos e tomar decisões informadas com base nas informações disponíveis, impulsionando a transformação digital e o progresso em várias áreas da

indústria, pesquisa e sociedade como um todo.

Data Lake

Com o crescimento exponencial no volume e na variedade de dados gerados atualmente, as organizações estão buscando maneiras eficazes de armazenar, gerenciar e analisar essas informações em larga escala. Nesse contexto, o conceito de Data Lake surgiu como uma solução promissora para lidar com os desafios do Big Data.

O Data Lake é um repositório centralizado e altamente escalável que armazena grandes quantidades de dados brutos em sua forma original, independentemente do formato ou estrutura. Diferentemente dos sistemas tradicionais de armazenamento de dados, o Data Lake não requer uma modelagem prévia dos dados ou esquemas rígidos, permitindo a inclusão de dados de várias fontes, como bancos de dados relacionais, logs de servidores, dispositivos IoT (Internet das Coisas) e mídias sociais.

Essa abordagem flexível e orientada a dados brutos oferece várias vantagens. Em primeiro lugar, o Data Lake permite a captura e o armazenamento de dados em sua forma bruta, preservando todos os detalhes e contextos originais. Isso é crucial para análises avançadas e descoberta de insights significativos, uma vez que os dados podem ser explorados e transformados posteriormente, conforme necessário.

Além disso, o Data Lake possibilita a integração e a consolidação de dados de diferentes fontes em um único repositório. Essa centralização simplifica a busca, o acesso e o compartilhamento de dados, eliminando silos e promovendo a colaboração entre equipes e departamentos.

Outra vantagem do Data Lake é sua capacidade de dimensionamento horizontal, permitindo a expansão do armazenamento e do processamento de dados de forma elástica, à medida que as necessidades da organização crescem. Isso garante a escalabilidade do ambiente e a capacidade de lidar com grandes volumes de dados em tempo real.

No entanto, é importante destacar que, embora o Data Lake ofereça flexibilidade e escalabilidade, a governança de dados e a qualidade dos dados são aspectos críticos a serem considerados. Um Data Lake bem projetado deve incluir mecanismos robustos de metadados, segurança e controle de acesso, garantindo a confiabilidade, a privacidade e a conformidade dos dados.

O Data Lake representa uma abordagem inovadora e eficaz para o armazenamento e o gerenciamento de dados em Big Data. Ao preservar a integridade dos dados brutos e promover a flexibilidade e a escalabilidade, o Data Lake capacita as organizações a explorar e extrair insights valiosos de seus enormes volumes de dados, impulsionando a tomada de decisões informadas e a obtenção de vantagens competitivas no cenário atual de dados em constante evolução.

Existem várias opções disponíveis no mercado para implementar um Data Lake, seja em ambiente on-premise (local) ou em nuvem. Essas opções oferecem diferentes recursos, escalabilidade e modelos de implementação, permitindo que as organizações escolham a melhor solução para atender às suas necessidades específicas. A seguir, vou apresentar algumas das opções populares de Data Lake disponíveis:

1. AWS S3 (Amazon Simple Storage Service): A Amazon Web Services (AWS) oferece o Amazon S3 como um serviço de armazenamento em nuvem altamente escalável e durável. O S3 é amplamente utilizado para construir Data Lakes, permitindo o armazenamento de grandes volumes de dados brutos. Ele oferece recursos avançados de segurança, criptografia, controle de acesso e integração com outras ferramentas e serviços da AWS, como AWS Glue e Amazon Athena.

2. Azure Data Lake Storage: O Azure Data Lake Storage é uma solução de armazenamento escalável fornecida pela Microsoft Azure. Ele oferece recursos para armazenar, processar e analisar grandes volumes de dados de forma eficiente. O Azure Data Lake Storage permite a integração com várias ferramentas e serviços do ecossistema do Azure, como Azure Databricks e Azure Data Factory, tornando-o uma opção popular para Data Lakes em nuvem.
3. Google Cloud Storage: O Google Cloud Storage é uma plataforma de armazenamento em nuvem oferecida pelo Google Cloud Platform (GCP). Ele fornece armazenamento escalável e durável para criar Data Lakes na nuvem. O Google Cloud Storage oferece recursos avançados de segurança, controle de acesso, integração com outras ferramentas do GCP, como BigQuery e Dataflow, e suporte para várias cargas de trabalho de Big Data.
4. Hadoop Distributed File System (HDFS): O Hadoop Distributed File System é um sistema de arquivos distribuído e altamente escalável projetado para lidar com grandes volumes de dados. Ele faz parte do ecossistema do Apache Hadoop e pode ser implantado tanto em ambientes on-premise quanto em nuvem. O HDFS é frequentemente usado como componente central em implantações de Data Lake baseadas em Hadoop, permitindo o armazenamento de dados brutos de várias fontes.
5. Cloudera Data Platform: A Cloudera Data Platform (CDP) é uma plataforma abrangente para gerenciar, processar e analisar dados em escala. Ela oferece opções flexíveis de implantação, permitindo a criação de Data Lakes tanto em ambientes on-premise quanto em nuvem. A CDP inclui recursos avançados de governança de dados, segurança e integração com outras ferramentas e serviços de Big Data.

Essas são apenas algumas das opções disponíveis no mercado para implementar um Data Lake. Cada uma delas tem suas próprias vantagens e características específicas, e a escolha depende das necessidades, requisitos e estratégias de uma organização em particular. É importante realizar uma avaliação detalhada dos recursos, custos, suporte e integração com outras tecnologias antes de selecionar a melhor opção para construir um Data Lake eficaz e escalável.

Data Warehouse em Big Data

À medida que as organizações lidam com a crescente quantidade de dados gerados diariamente, surge a necessidade de armazenar, organizar e analisar essas informações de maneira eficiente e estruturada. É nesse contexto que o conceito de Data Warehouse (DW) em Big Data ganha destaque.

Um Data Warehouse é um sistema centralizado de armazenamento de dados que coleta, organiza e integra informações de várias fontes diferentes dentro de uma organização. Ele é projetado para suportar a análise e a geração de relatórios, fornecendo uma visão unificada e consistente dos dados, independentemente da sua origem ou formato.

Em um ambiente de Big Data, o Data Warehouse desempenha um papel crucial na integração de dados estruturados e não estruturados, provenientes de diferentes fontes, como bancos de dados relacionais, sistemas de gestão de dados não relacionais, sensores IoT, mídias sociais e muito mais. Essa integração permite que as organizações obtenham uma visão holística e abrangente de seus dados, identificando padrões, tendências e insights relevantes para a tomada de decisões estratégicas.

Uma das principais características do Data Warehouse é a modelagem dimensional, que organiza os dados em torno de fatos (eventos de negócios) e dimensões (características relacionadas aos fatos). Essa estrutura facilita a análise multidimensional e a exploração dos dados por meio de consultas complexas e agregações.

Além disso, o Data Warehouse é projetado para oferecer desempenho e escalabilidade, permitindo o processamento eficiente de grandes volumes de dados. Ele utiliza técnicas como particionamento, indexação e otimização de consultas para garantir respostas rápidas e eficazes às consultas analíticas.

Outro aspecto importante do Data Warehouse em Big Data é a governança dos dados. Isso envolve a definição de políticas, padrões e processos para garantir a qualidade, a segurança e a conformidade dos dados. A governança adequada é essencial para manter a confiabilidade e a consistência dos dados em um ambiente de Big Data complexo e em constante evolução.

O DW em Big Data desempenha um papel fundamental na organização e análise de grandes volumes de dados. Ele oferece uma visão consolidada dos dados, facilitando a descoberta de insights valiosos para a tomada de decisões estratégicas. Com sua modelagem dimensional, desempenho escalável e governança de dados eficaz, o Data Warehouse em Big Data se torna uma ferramenta essencial para as organizações que buscam obter o máximo valor de seus dados em um cenário de Big Data em constante expansão.

Existem várias opções disponíveis no mercado para implementar um Data Warehouse para Big Data, tanto em ambientes on-premise (locais) quanto em nuvem. Essas opções oferecem diferentes recursos, escalabilidade e modelos de implementação, permitindo que as organizações escolham a solução mais adequada às suas necessidades específicas. A seguir, vou apresentar algumas das opções populares de Data Warehouse disponíveis:

1. Amazon Redshift: O Amazon Redshift é um serviço de Data Warehouse oferecido pela Amazon Web Services (AWS). Ele fornece um ambiente de armazenamento e análise altamente escalável para Big Data. O Redshift é projetado para executar consultas rápidas e complexas em grandes volumes de dados, com recursos avançados de compressão, particionamento e distribuição. Ele pode ser implantado em nuvem e oferece integração com outras ferramentas e serviços da AWS, como AWS Glue e Amazon S3.
2. Google BigQuery: O Google BigQuery é um serviço de Data Warehouse fornecido pelo Google Cloud Platform (GCP). Ele oferece armazenamento e análise escaláveis de Big Data, permitindo consultas rápidas em grandes conjuntos de dados. O BigQuery utiliza um modelo de pagamento por uso e oferece recursos avançados de processamento paralelo e distribuído. Ele integra-se perfeitamente com outros serviços do GCP, como Google Cloud Storage e Dataflow.
3. Azure Synapse Analytics: O Azure Synapse Analytics é uma plataforma de análise de Big Data fornecida pela Microsoft Azure. Ela combina recursos de Data Warehouse e Big Data em um único ambiente integrado. O Synapse Analytics oferece escalabilidade elástica, permitindo o processamento de grandes volumes de dados em tempo real. Ele integra-se com outras ferramentas e serviços do ecossistema do Azure, como Azure Data Lake Storage e Azure Databricks.
4. Snowflake: O Snowflake é uma plataforma de Data Warehouse na nuvem projetada para lidar com grandes volumes de dados. Ele oferece recursos escaláveis e flexíveis para armazenamento e análise de dados de várias fontes. O Snowflake utiliza uma arquitetura baseada em nuvem, permitindo que os usuários dimensionem seus recursos de armazenamento e computação conforme necessários. Ele suporta consultas SQL padrão e oferece recursos avançados de segurança e governança.

Essas são apenas algumas das opções disponíveis no mercado para implementar um Data Warehouse para Big Data. Cada uma delas tem suas próprias vantagens, recursos e modelos de implementação. A escolha depende das necessidades, requisitos e estratégias de uma organização em particular. É importante realizar uma avaliação detalhada dos recursos, custos,

suporte e integração com outras tecnologias antes de selecionar a melhor opção para construir um Data Warehouse eficaz e escalável para Big Data.

Introdução ao Apache Hive

O Apache Hive é um projeto de código aberto do ecossistema do Apache Hadoop que fornece uma interface de consulta e análise de dados em grandes conjuntos de dados armazenados no Hadoop Distributed File System (HDFS) ou em outros sistemas de armazenamento compatíveis com o Hadoop. O Hive oferece uma linguagem de consulta chamada HiveQL, que é semelhante à SQL, permitindo que os usuários escrevam consultas familiares e realizem análises em dados estruturados e semiestruturados.

Uma das principais características do Apache Hive é sua capacidade de trabalhar com esquemas de dados semelhantes a tabelas, chamados de tabelas Hive, que são mapeadas para arquivos ou diretórios no HDFS. Essas tabelas Hive podem ser criadas, consultadas e manipuladas usando a linguagem HiveQL. Além disso, o Hive oferece suporte a particionamento de tabelas, particionando os dados com base em colunas específicas para melhorar o desempenho das consultas.

O Hive também permite a criação de funções personalizadas e transformações complexas de dados por meio do uso de UDFs (User-Defined Functions) e UDTFs (User-Defined Table Functions). Isso oferece flexibilidade para personalizar e estender as capacidades do Hive para atender a necessidades específicas.

Uma das principais vantagens do Apache Hive é sua capacidade de fornecer uma camada de abstração sobre o Hadoop, permitindo que usuários familiarizados com SQL aproveitem a escalabilidade e a capacidade de processamento distribuído do Hadoop para análise de dados. O Hive traduz as consultas escritas em HiveQL em tarefas MapReduce ou em tarefas executadas no Apache Tez, um mecanismo de execução de alto desempenho para consultas no Hadoop.

Além disso, o Hive integra-se com outras ferramentas e serviços do ecossistema Hadoop, como o Apache Spark, permitindo a execução de consultas usando o Spark como mecanismo de processamento.

O Apache Hive é amplamente utilizado em diversas áreas, como análise de dados, processamento de logs, criação de painéis de controle e relatórios. Ele fornece uma maneira eficiente e familiar de consultar e analisar grandes volumes de dados armazenados no Hadoop, aproveitando as vantagens do processamento distribuído.

O Apache Hive é frequentemente classificado como um data warehouse, embora seja importante entender seu papel e suas características específicas nesse contexto. O Hive é projetado para fornecer uma camada de abstração e consulta sobre grandes conjuntos de dados armazenados no Hadoop Distributed File System (HDFS) ou em outros sistemas de armazenamento compatíveis com o Hadoop. Ele permite que os usuários escrevam consultas usando uma linguagem de consulta semelhante ao SQL, chamada HiveQL, para analisar e extrair insights dos dados.

Embora o Hive compartilhe algumas semelhanças com sistemas de data warehousing tradicionais, como a capacidade de consultar dados em esquemas de tabela, particionamento e otimizações de consulta, ele também possui algumas diferenças significativas. Ao contrário dos data warehouses tradicionais, o Hive não foi inicialmente projetado para oferecer um desempenho de consulta em tempo real ou recursos avançados de modelagem de dados.

Em vez disso, o Hive foi originalmente concebido para aproveitar a escalabilidade do Hadoop e do processamento distribuído para análise de dados em grande escala. Ele se baseia no mecanismo MapReduce ou no Apache Tez para executar consultas em paralelo em clusters de servidores.

Com o tempo, o Hive evoluiu para oferecer melhorias de desempenho e recursos adicionais, como otimização de consultas, suporte a índices, uso de colunas virtuais e integração com mecanismos de processamento mais rápidos, como o Apache Spark.

Portanto, embora o Hive seja considerado um data warehouse em alguns contextos, é importante entender suas características distintas e reconhecer que ele é uma solução que aproveita a escalabilidade e o poder do processamento distribuído do Hadoop para análise de dados em grande escala, com foco em eficiência e escalabilidade, em vez de consultas em tempo real.

Comandos HDFS

É importante conhecer os comandos do Hadoop Distributed File System (HDFS) por várias razões:

1. Gerenciamento eficiente de dados: Os comandos do HDFS permitem que você gerencie arquivos e diretórios de forma eficiente. Você pode criar, copiar, mover, renomear, excluir e visualizar informações sobre arquivos e diretórios no HDFS. Conhecer esses comandos permite que você execute tarefas de gerenciamento de dados de forma mais eficaz, economizando tempo e esforço.
2. Manipulação de permissões e propriedades: O HDFS possui recursos de controle de acesso e permissões para garantir a segurança dos dados armazenados. Comandos como `chmod`, `chown` e `chgrp` permitem que você defina e altere as permissões de arquivos e diretórios, bem como o proprietário e o grupo proprietário. Conhecer esses comandos é fundamental para garantir a segurança e a conformidade dos dados.
3. Solução de problemas: Ao trabalhar com o HDFS, é provável que você encontre situações em que precise solucionar problemas ou diagnosticar problemas. Os comandos do HDFS, como `ls`, `stat`, `cat` e `tail`, permitem que você visualize e obtenha informações sobre os arquivos e diretórios no HDFS, o que pode ajudar na solução de problemas e na identificação de problemas em potencial.
4. Automação de tarefas: Os comandos do HDFS podem ser usados em scripts e programas para automatizar tarefas de gerenciamento de dados. Por exemplo, você pode criar um script para copiar regularmente dados do HDFS para o sistema de arquivos local ou para realizar backups automatizados. Ter conhecimento dos comandos do HDFS permite que você automatize essas tarefas e torne seu fluxo de trabalho mais eficiente.
5. Preparação para certificações e entrevistas: Se você estiver buscando certificações relacionadas ao ecossistema Hadoop ou se estiver se preparando para entrevistas de emprego em que o conhecimento de Hadoop e do HDFS é relevante, conhecer os comandos do HDFS é essencial. Essas certificações e entrevistas podem incluir perguntas práticas sobre como usar os comandos do HDFS para realizar tarefas específicas.

Em resumo, conhecer os comandos do HDFS é fundamental para realizar tarefas de gerenciamento de dados eficientemente, garantir a segurança e a conformidade dos dados, solucionar problemas, automatizar tarefas e estar preparado para certificações e entrevistas relacionadas ao Hadoop e ao ecossistema do HDFS.

Segue alguns dos principais comandos do Hadoop Distributed File System (HDFS):

1. Listar diretórios e arquivos:

- `hdfs dfs -ls \<diretório\>`: Lista os arquivos e diretórios dentro do caminho especificado.
- `hdfs dfs -ls -R \<diretório\>`: Lista os arquivos e diretórios recursivamente, exibindo também os conteúdos dos subdiretórios.

2. Criar diretórios:

- `hdfs dfs -mkdir \<diretório\>`: Cria um novo diretório no caminho especificado.

3. Copiar arquivos:

- `hdfs dfs -put \<arquivo_local\> \<caminho_destino\>`: Copia um arquivo local para o HDFS.
- `hdfs dfs -copyFromLocal \<arquivo_local\> \<caminho_destino\>`: Copia um arquivo local para o HDFS (versão alternativa do comando anterior).
- `hdfs dfs -get \<caminho_origem\> \<diretório_destino\>`: Copia um arquivo do HDFS para o sistema de arquivos local.

4. Excluir arquivos ou diretórios:

- `hdfs dfs -rm \<caminho\>`: Exclui um arquivo do HDFS.
- `hdfs dfs -rm -r \<caminho\>`: Exclui um diretório e todo o seu conteúdo do HDFS.

5. Mover arquivos ou diretórios:

- `hdfs dfs -mv \<caminho_origem\> \<caminho_destino\>`: Move um arquivo ou diretório para um novo local no HDFS.

6. Visualizar o conteúdo de um arquivo:

- `hdfs dfs -cat \<arquivo\>`: Exibe o conteúdo de um arquivo no HDFS.
- `hdfs dfs -tail \<arquivo\>`: Exibe as últimas linhas de um arquivo no HDFS.

7. Obter informações sobre um arquivo ou diretório:

- `hdfs dfs -stat \<caminho\>`: Exibe informações detalhadas sobre um arquivo ou diretório.
- `hdfs dfs -du -s -h \<diretório\>`: Exibe o tamanho total de um diretório e seus subdiretórios em formato legível.

Comandos do HDFS para lidar com permissões de arquivos:

1. Definir permissões de arquivo ou diretório:

- `hdfs dfs -chmod \<permissões\> \<arquivo_ou_diretório\>`: Define as permissões para um arquivo ou diretório no HDFS. As permissões podem ser especificadas no formato octal (por exemplo, 755) ou simbólico (por exemplo, u=rwx,g=rx,o=r).

2. Alterar o proprietário de um arquivo ou diretório:

- `hdfs dfs -chown \<proprietário\> \<arquivo_ou_diretório\>`: Altera o proprietário de um arquivo ou diretório no HDFS.
- `hdfs dfs -chgrp \<grupo\> \<arquivo_ou_diretório\>`: Altera o grupo proprietário de um arquivo ou diretório no HDFS.

3. Visualizar permissões e propriedades de um arquivo ou diretório:

- `hdfs dfs -ls -d -R \<arquivo_ou_diretório\>`: Exibe as informações de permissões e propriedades para um arquivo ou diretório no HDFS.

4. Alterar permissões recursivamente em um diretório:

- `hdfs dfs -chmod -R \<permissões\> \<diretório\>`: Define as permissões recursivamente em um diretório e todos os seus subdiretórios e arquivos.

5. Alterar proprietário recursivamente em um diretório:

- `hdfs dfs -chown -R \<proprietário\> \<diretório\>`: Altera o proprietário recursivamente em um diretório e todos os seus subdiretórios e arquivos.
- `hdfs dfs -chgrp -R \<grupo\> \<diretório\>`: Altera o grupo proprietário recursivamente em um diretório e todos os seus subdiretórios e arquivos.

6. Verificar as permissões de um arquivo ou diretório:

- `hdfs dfs -test -[ezs] \<arquivo_ou_diretório\>`: Verifica se você tem permissões de leitura (e), gravação (w) ou execução (x) em um arquivo ou diretório.

Esses comandos permitem que você defina, altere e verifique permissões de arquivos e diretórios no HDFS. É importante observar que, para executar esses comandos, é necessário ter as permissões apropriadas no HDFS. Além disso, o HDFS também segue as políticas de segurança e autenticação configuradas no cluster do Hadoop.

Introdução ao Delta Lake

O Delta Lake é um sistema de armazenamento de dados open source projetado para oferecer confiabilidade, consistência e escalabilidade aprimoradas para ambientes de Big Data. Ele foi desenvolvido pela Databricks e é construído sobre o Apache Parquet e o Apache Spark, aproveitando o poder e a flexibilidade dessas tecnologias.

O Delta Lake aborda desafios comuns enfrentados no gerenciamento de dados em escala, como a garantia da integridade dos dados, a consistência transacional e a governança de dados. Ele fornece uma camada de abstração sobre o armazenamento subjacente, permitindo que os usuários trabalhem com dados em formatos tabulares semelhantes aos de um banco de dados relacional, mantendo a capacidade de processamento distribuído e a escalabilidade do Spark.

Uma das principais características do Delta Lake é o suporte à consistência transacional, permitindo operações atômicas e isoladas em dados. Isso significa que as operações de escrita e atualização de dados são tratadas como transações, garantindo que as alterações sejam aplicadas em sua totalidade ou não sejam aplicadas de forma alguma. Essa consistência transacional é particularmente valiosa em ambientes de Big Data, onde várias operações podem ocorrer simultaneamente.

Outra característica importante do Delta Lake é o suporte a esquemas evolutivos, permitindo que os dados sejam atualizados de forma incremental e flexível ao longo do tempo. Isso facilita a evolução e a manutenção dos esquemas à medida que os requisitos de negócios mudam, evitando a necessidade de reescrever e reprocessar todos os dados.

Além disso, o Delta Lake oferece recursos avançados de gerenciamento de versões e de recuperação de falhas, permitindo que os usuários acessem versões anteriores dos dados, acompanhem o histórico das alterações e restaurem dados em caso de falhas.

O Delta Lake também se integra perfeitamente ao ecossistema do Apache Spark, permitindo que os usuários aproveitem os recursos avançados de processamento de dados do Spark para consultas e análises avançadas em dados do Delta Lake.

O Delta Lake complementa o Hadoop Distributed File System (HDFS) de várias maneiras, proporcionando recursos adicionais que melhoram a confiabilidade, consistência e governança dos dados no ecossistema Hadoop. Aqui estão alguns pontos em que o Delta Lake se destaca em relação ao HDFS:

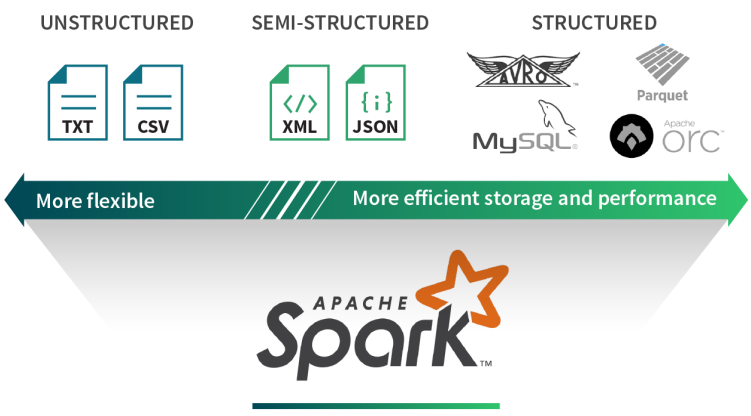
1. Consistência transacional: O HDFS é um sistema de arquivos distribuído que oferece alta escalabilidade, mas não possui suporte nativo para transações atômicas em nível de registro. Por outro lado, o Delta Lake fornece consistência transacional, permitindo que operações de gravação e atualização sejam tratadas como transações, garantindo que as alterações sejam aplicadas completamente ou não sejam aplicadas de forma alguma. Isso é fundamental para garantir a integridade dos dados em ambientes de Big Data.
2. Esquemas evolutivos: O HDFS é um sistema de armazenamento de dados bruto, sem um mecanismo integrado para gerenciar alterações no esquema dos dados. O Delta Lake, por sua vez, oferece suporte a esquemas evolutivos, permitindo que os dados sejam atualizados de forma incremental e flexível ao longo do tempo. Isso simplifica a evolução e a manutenção dos esquemas à medida que os requisitos de negócios mudam, evitando a necessidade de reescrever e reprocessar todos os dados.
3. Gerenciamento de versões: O HDFS não possui recursos nativos de gerenciamento de versões, o que dificulta o rastreamento das alterações nos dados ao longo do tempo. O Delta Lake, por sua vez, oferece um mecanismo de gerenciamento de versões embutido, permitindo que os usuários acessem versões anteriores dos dados, acompanhem o histórico das alterações e restaurem dados em caso de falhas ou necessidades de auditoria.
4. Otimização de consultas: Embora o HDFS seja eficiente em armazenar grandes volumes de dados, a otimização de consultas pode ser um desafio. O Delta Lake integra-se ao Apache Spark, um mecanismo de processamento distribuído de alto desempenho, permitindo que os usuários aproveitem recursos avançados de processamento e otimização de consultas do Spark para análises mais rápidas e eficientes em dados do Delta Lake.
5. Governança de dados: O Delta Lake oferece recursos de governança de dados, como validação de esquema, restrições de integridade e metadados enriquecidos. Esses recursos ajudam a garantir a qualidade dos dados, a conformidade com regulamentações e a consistência nos processos de ingestão e transformação de dados.

Em suma, o Delta Lake complementa o HDFS ao fornecer recursos adicionais, como consistência transacional, esquemas evolutivos, gerenciamento de versões e otimização de consultas, melhorando a confiabilidade, consistência e governança dos dados no ecossistema Hadoop. Ele oferece uma camada adicional de abstração sobre o HDFS, permitindo o gerenciamento e a análise mais eficientes de grandes volumes de dados em ambientes de Big Data.

Em resumo, o Delta Lake é uma solução inovadora para o gerenciamento de dados em ambientes de Big Data, oferecendo confiabilidade, consistência transacional e escalabilidade aprimoradas. Com seu suporte a operações transacionais, esquemas evolutivos e recursos de gerenciamento de versões, o Delta Lake ajuda as organizações a lidarem com os desafios do gerenciamento de dados em larga escala, permitindo o acesso confiável e eficiente a dados para análises avançadas e tomada de decisões informadas.

Ao trabalhar com Big Data, existem vários tipos de arquivos que são comumente usados para armazenar e processar grandes volumes de dados. Cada tipo de arquivo tem suas próprias características e é projetado para atender a diferentes necessidades de armazenamento, processamento e análise de dados. Aqui estão alguns dos tipos de arquivos mais populares usados no contexto do Big Data:

1. Arquivos CSV (Comma-Separated Values): Os arquivos CSV são arquivos de texto simples em que os dados são organizados em colunas separadas por vírgulas. Eles são amplamente usados devido à sua simplicidade e compatibilidade com várias ferramentas e plataformas de análise. Os arquivos CSV são facilmente legíveis por humanos e podem ser compactados para otimizar o uso de espaço de armazenamento. No entanto, eles podem não ser eficientes para análises complexas ou para dados semiestruturados.
2. Arquivos Parquet: O formato Parquet é um formato de armazenamento colunar projetado para otimizar o processamento e a análise de dados em Big Data. Ele é eficiente para leitura seletiva de colunas e compressão de dados, o que o torna ideal para análises rápidas e eficientes em grandes conjuntos de dados. O Parquet é amplamente utilizado em ecossistemas de Big Data, como o Apache Hadoop e o Apache Spark.
3. Arquivos Avro: O formato Avro é um formato de dados compacto e serializado que permite a definição de esquemas complexos. Ele oferece suporte à evolução de esquemas, o que significa que os dados podem ser atualizados sem a necessidade de modificar a estrutura do arquivo. O Avro é comumente usado em ambientes de Big Data onde a flexibilidade e a portabilidade dos dados são essenciais.
4. Arquivos ORC (Optimized Row Columnar): O formato ORC é outro formato colunar otimizado para consultas rápidas e eficientes em grandes volumes de dados. Ele oferece recursos avançados de compressão, particionamento e indexação, o que o torna eficiente para consultas seletivas em dados complexos. O ORC é frequentemente usado em conjunto com o Apache Hive e o Apache Impala para análise interativa de dados em tempo real.
5. Arquivos JSON (JavaScript Object Notation): O formato JSON é amplamente utilizado para representar dados semiestruturados. Ele fornece uma estrutura hierárquica e flexível para armazenar dados complexos, como documentos, registros e objetos. Os arquivos JSON são facilmente legíveis por humanos e são comumente usados em aplicativos web e em integrações de sistemas.
6. Arquivos XML (eXtensible Markup Language): O formato XML é usado para estruturar e representar dados em um formato legível por máquina e por humanos. Ele é amplamente utilizado para troca de dados entre sistemas e é flexível o suficiente para acomodar estruturas de dados complexas. No entanto, o XML pode ser mais verboso em comparação com outros formatos e pode não ser eficiente para análises em larga escala.



É importante selecionar o tipo de arquivo adequado com base nas características dos dados, nas necessidades de processamento e nas ferramentas e tecnologias utilizadas no ambiente de Big Data. Cada tipo de arquivo tem suas vantagens e desvantagens, e a escolha certa pode afetar o desempenho, a eficiência e a escalabilidade das operações de Big Data.

Até agora, abordamos vários tópicos relacionados a Big Data, armazenamento de dados e ferramentas associadas. Aqui está um resumo dos assuntos discutidos e como eles estão relacionados:

1. Armazenamento de dados em Big Data: Iniciamos discutindo a importância do armazenamento de dados em Big Data, destacando tecnologias como Hadoop Distributed File System (HDFS) e suas peculiaridades.
2. Soluções de armazenamento em nuvem: Em seguida, exploramos opções de armazenamento em nuvem, como Data Lake e Data Warehouse, destacando suas características e benefícios.
3. Tecnologias específicas: Falamos sobre o Apache Hive, uma ferramenta de consulta e análise de dados no ecossistema Hadoop, e suas peculiaridades. Em seguida, discutimos o Delta Lake, um sistema de armazenamento de dados construído sobre o Apache Parquet e o Apache Spark, que oferece confiabilidade, consistência transacional e governança aprimoradas.
4. Tipos de arquivos em Big Data: Também abordamos os diferentes tipos de arquivos usados para trabalhar com Big Data, como CSV, Parquet, Avro, ORC, JSON e XML, destacando suas características e usos específicos.

Esses tópicos estão interconectados, pois fazem parte do ecossistema do Big Data e abordam diferentes aspectos do gerenciamento, armazenamento, análise e processamento de grandes volumes de dados. As tecnologias, como HDFS, Hive e Delta Lake, fornecem soluções para lidar com os desafios enfrentados ao trabalhar com Big Data, enquanto os diferentes tipos de arquivos oferecem opções de armazenamento e estruturação de dados.

Compreender esses tópicos e suas interconexões é fundamental para aproveitar ao máximo o potencial do Big Data, realizar análises eficientes e tomar decisões informadas com base nos dados. Cada um desses elementos desempenha um papel importante no ciclo de vida do Big Data, desde o armazenamento inicial até a análise e extração de insights valiosos.

Como usar a sandbox Hortonworks HDP 2.6.4

A Sandbox HDP é um CentOS Linux com todo Hortonworks Data Plataform (HDP) funcional.

A sandbox da Hortonworks HDP (Hortonworks Data Platform) 2.6 é uma plataforma de aprendizado e desenvolvimento gratuita que permite que os usuários explorem e experimentem o ecossistema do Hadoop sem a necessidade de configurar um ambiente completo de produção. A sandbox é uma distribuição pré-configurada do HDP que oferece uma experiência pronta para uso em um ambiente virtualizado.

A sandbox do HDP 2.6 é baseada em uma máquina virtual, permitindo que os usuários a executem em seus próprios sistemas, independentemente do sistema operacional que estejam usando. Isso oferece uma abordagem prática para aprender e testar as várias tecnologias e componentes do ecossistema do Hadoop, como o HDFS (Hadoop Distributed File System), o YARN (Yet Another Resource Negotiator), o MapReduce, o Hive, o Pig, o Spark, o HBase e muitos outros.

Uma das principais vantagens da sandbox do HDP 2.6 é a sua facilidade de uso. Ela vem pré-configurada com todos os componentes necessários para executar um cluster do Hadoop, permitindo que os usuários iniciem rapidamente e se familiarizem com as diferentes tecnologias. Além disso, a sandbox inclui uma interface gráfica intuitiva, o Ambari, que facilita a administração e o monitoramento do cluster.

A sandbox do HDP 2.6 também oferece uma série de tutoriais e exemplos práticos que ajudam os usuários a entenderem e explorarem os recursos do Hadoop. Esses materiais educacionais abrangem desde noções básicas até casos de uso mais avançados, permitindo que os usuários aprendam e experimentem a análise de dados em larga escala, processamento de dados distribuídos e outras capacidades do Hadoop.

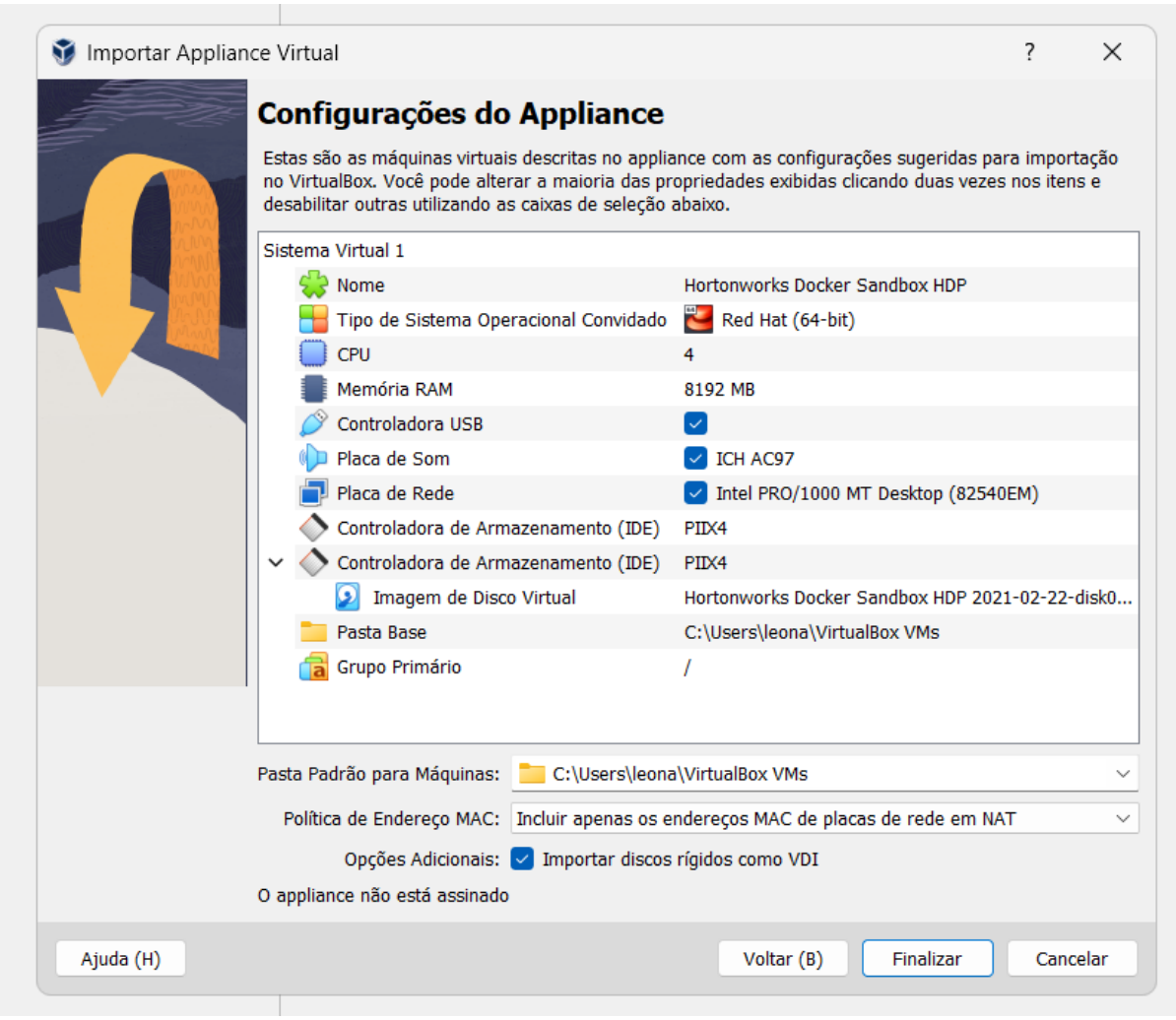
Além disso, a sandbox do HDP 2.6 é uma excelente ferramenta para desenvolvedores que desejam testar e depurar seus próprios aplicativos e integrações com o ecossistema do Hadoop. Ela oferece um ambiente seguro para desenvolvimento, permitindo que os desenvolvedores experimentem e validem suas soluções antes de implantá-las em um ambiente de produção.

Em resumo, a sandbox da Hortonworks HDP 2.6 é uma plataforma valiosa para aprender, explorar e desenvolver com o ecossistema do Hadoop. Ela oferece um ambiente pré-configurado e fácil de usar, tutoriais educacionais abrangentes e uma interface gráfica intuitiva, permitindo que os usuários mergulhem no mundo do Big Data e aproveitem os benefícios da análise de dados em escala.

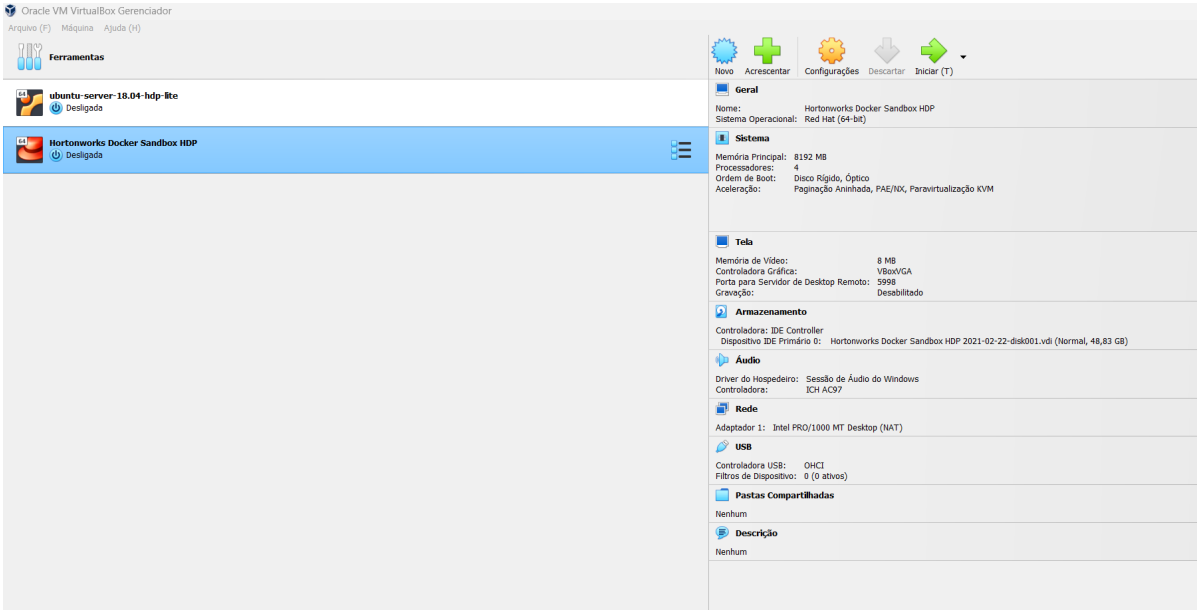
O link para download desta sandbox é: https://drive.google.com/file/d/1Ipb_81bU6Re2m-uQW5M_o9aOHpcAJ9qw/view?usp=sharing

Após o download do arquivo OVA, faça a instalação do Virtualbox versão 6 ou superior e faça a importação da máquina virtual. Abra o VirtualBox e vá para Arquivo -> Importar Appliance. Selecione a imagem da sandbox que você baixou e clique em Abrir.

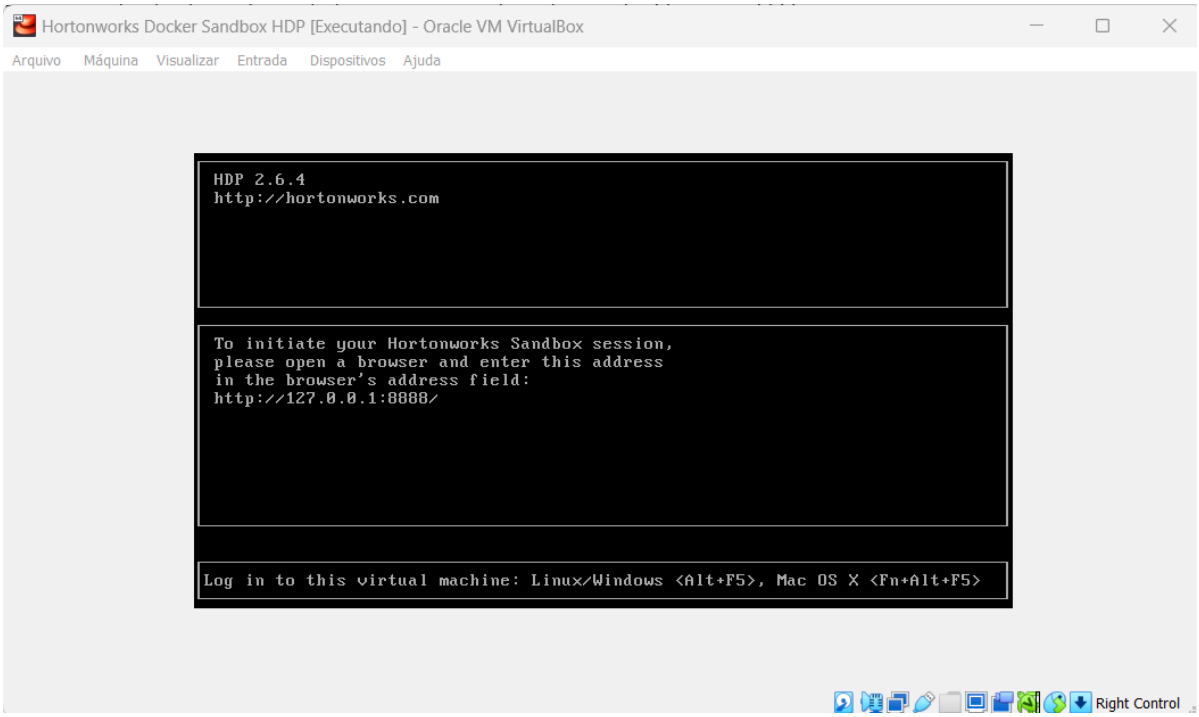
Você deve obter uma tela como esta:



Após importar a sandbox, inicie a máquina virtual clicando no botão com ícone de seta verde (Iniciar (T)):

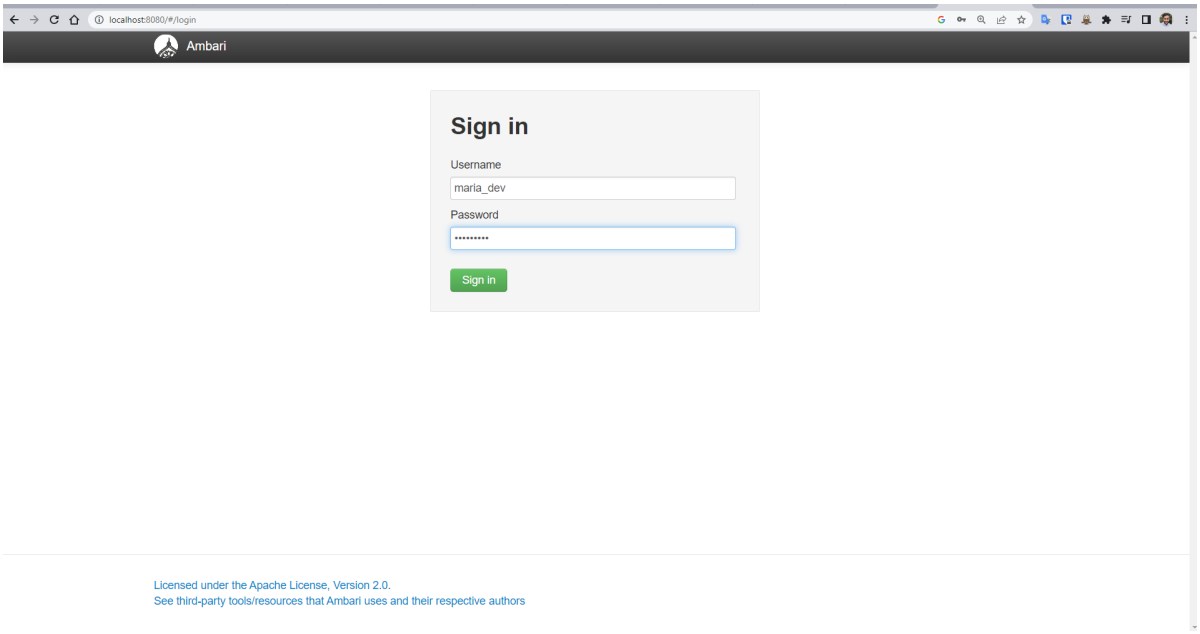


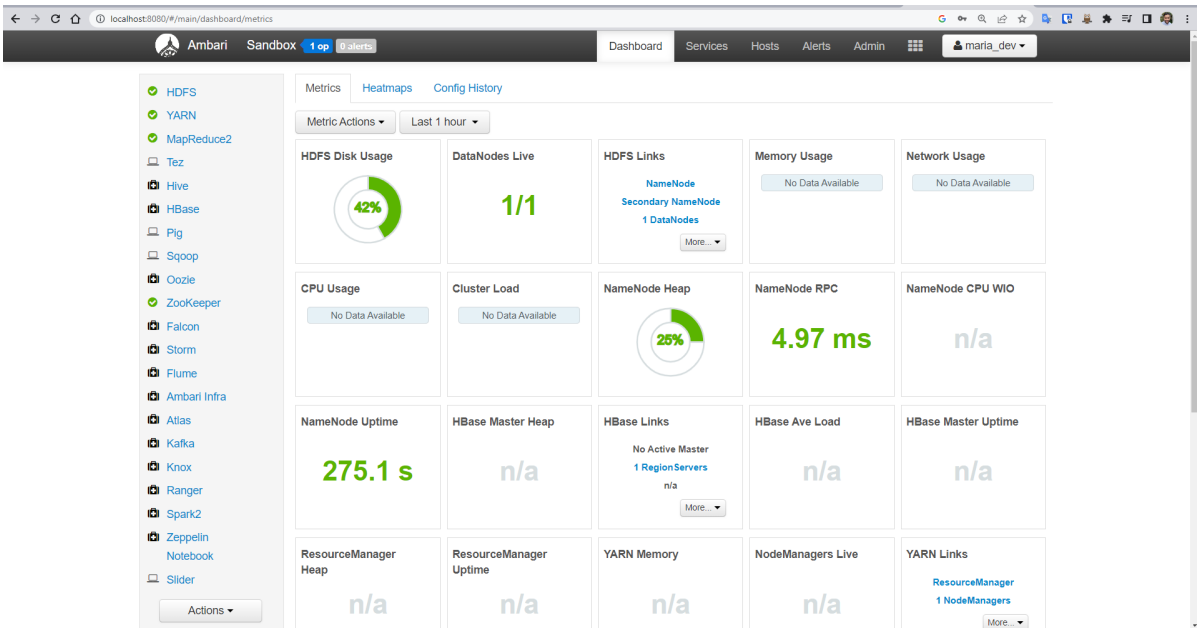
Uma janela do console se abre e exibe o processo de inicialização. Esse processo leva alguns minutos. Quando você ver a seguinte tela, você pode começar a usar a sandbox:



Você pode acessar via SSH ou usar a própria interface web que a distribuição oferece, acessível via http://127.0.0.1:4200

Para acessar o Ambari: http://127.0.0.1:8080





Credenciais:

- Login: maria_dev Senha: maria_dev
- Login: raj_ops Senha: raj_ops
- Login: holger_gov Senha: holger_gov
- Login: amy_ds Senha: amy_ds

1. admin – Administrador do sistema
2. maria_dev – Responsável por preparar e obter “insights” a partir de dados. Ela usa bastante as ferramentas como Hive, Pig, HBase.
3. raj_ops – Responsável pela construção de infraestrutura como pesquisa e desenvolvimento de atividades como projeto, instalação, configuração e administração do Hadoop. Possui conhecimentos sólidos de administração de sistemas operacionais.
4. holger_gov – Responsável pela governança de dados.
5. amy_ds – Cientista de dados.

Name id(s)	Role	Serviços
Sam Admin	Ambari Admin	Ambari
Raj (raj_ops)	Hadoop Warehouse Operator	Hive/Tez, Ranger, Falcon, Knox, Sqoop, Oozie, Flume, Zookeeper
Maria (maria_dev)	Spark and SQL Developer	Hive, Zeppelin, MapReduce/Tez/Spark, Pig, Solr, HBase/Phoenix, Sqoop, NiFi, Storm, Kafka, Flume
Amy (amy_ds)	Data Scientist	Spark, Hive, R, Python, Scala
Holger (holger_gov)	Data Steward	Atlas

OS Level Authorization

Name id(s)	HDFS Authorization	Ambari Authorization	Ranger Authorization
Sam Admin	Max Ops	Ambari Admin	Admin access
Raj (raj_ops)	Access to Hive, Hbase, Atlas, Falcon, Ranger, Knox, Sqoop, Oozie, Flume, Operations	Cluster Administrator	Admin Access
Maria (maria_dev)	Access to Hive, Hbase, Falcon, Oozie and Spark	Service Operator	Normal User Access
Amy (amy_ds)	Access to Hive, Spark and Zeppelin	Service Operator	Normal User Access
Holger (holger_gov)	Access to Atlas	Service Administrator	Normal User Access

Para começar abra um terminal na máquina hospedeira (sugestão instale o Moba na sua máquina) e execute os seguintes comandos: