Projeto final de Processamento Massivo de Dados com Apache HBase™

Sobre

Este repositório é parte integrante do projeto final da disciplina de Processamento Massivo de Dados do curso de Ciência da Computação da Universidade Federal de São Carlos Campus Sorocaba ministrado pela Profa. Dra. Sahudy Montenegro González.

Membros do Projeto:

Eduardo Rodrigues da Cruz - 587540
Renato Araujo Rizzo - 587788

Objetivo

O objetivo do projeto foi pesquisar, descrever e criar uma documentação em português do banco de dados NoSQL Apache HBase™, e detalhar suas características baseadas nos conceitos abordados em sala de aula.

Conteúdo

O projeto é composto por uma documentação em português brasileiro detalhando como o banco de dados Apache HBase™ implementa as principais características de bancos NoSQL e um tutorial de instalação com forma de uso e exercícios práticos aplicados ao banco.

Introdução

HBase é um banco de dados NoSQL orientado a colunas e de código aberto criado e distribuído pela Apache Software Foundation e que foi derivado do projeto BigTable do Google.
O HBase roda no Hadoop, que é uma plataforma de software em Java de computação distribuída voltada para clusters e processamento de grandes volumes de dados, também criado pela Apache e que usa modelos de programação simples.[1][2][3][4]

Modelo de Dados e Terminologia

No HBase, os dados são armazenados em tabelas que possuem linhas e colunas, mas que também são comumente chamadas de mapa multi-dimensional.

- **Tabela**: Uma tabela no HBase consiste de diversas linhas.
- **Linha**: Uma linha no HBase consiste de uma chave da linha e uma ou mais colunas com valores associados a ela. Linhas são organizadas alfabeticamente por chave de linha no momento de armazenamento. O objetivo é armazenar os dados de forma que linhas relacionadas fiquem próximas

umas as outras. Um padrão comum para a chave de linha é um domínio de website. Se as chaves de linhas são domínios você provavelmente vai armazená-las em reverso (org.apache.www, org.apache.mail, org.apache.jira). Dessa forma, todos os domínios Apache do exemplo vão estar próximos uns dos outros ao invés de espalhados se baseado pela primeira letra do subdomínio.

- **Coluna**: Uma coluna no HBase consiste de uma família de colunas e um qualificador de colunas, pelo qual são delimitados por um : (caractere de dois-ponto).
- **Famílias de Colunas**: Famílias de colunas reúne fisicamente um conjunto de colunas e seus valores, por questões de desempenho. Cada família de coluna tem um conjunto de propriedade de armazenamento, como por exemplo se os valores devem estar no cache de memória, como os dados estão comprimidos ou as chaves de linhas estão codificadas, entre outras. Cada linha de uma tabela possui as mesmas famílias de colunas, embora uma determinada linha possa não armazenar nada em uma determinada família de colunas.
- **Qualificadores de Coluna**: Um qualificador de coluna é adicionado ao conteúdo de uma família de coluna, um qualificador de coluna pode ser content:html, e outro pode ser content:pdf. Embora as famílias de colunas sejam fixas na criação da tabela, os qualificadores de coluna são mutáveis e podem diferir bastante entre as linhas.
- **Célula**: Uma célula é um combinação de linha, família de coluna e um qualificador de coluna e contém um valor e um timestamp, no qual representa a versão do valor.
- **Timestamp**: Um timestamp é escrito ao lado de cada valor para uma determinada versão do valor. Por padrão, o timestamp representa a hora no RegionServer quando os dados foram gravados, mas você pode especificar um valor diferente para registrar os registros na célula.

Arquitetura

[6] O HBase usa modelo de distribuição Master-Slave e fornece leituras e gravações aleatórias de baixa latência sobre o HDFS (Hadoop Distributed File System) que é um sistema de arquivos distribuído projetado para ser tolerante a falhas e suportar rápida transferência de dados entre nós. As tabelas são dinamicamente distribuídas pelo sistema sempre que se tornam grandes demais para serem manipuladas (Auto Sharding). A unidade mais simples e fundamental de escalabilidade horizontal no HBase é uma região que é um conjunto contínuo e ordenado de linhas que são armazenadas juntas (subconjunto de dados da tabela). A arquitetura do HBase possui um único nó mestre chamado HMaster e vários slaves, ou seja, servidores de região. Cada slave serve um conjunto de regiões e uma região pode ser servida apenas por um único servidor de região. Sempre que um cliente envia uma solicitação de gravação, o HMaster recebe a solicitação e a encaminha para o servidor da região correspondente.

Componentes da arquitetura e replicação

A arquitetura do HBase possui 3 componentes importantes responsáveis pela coordenação, manutenção e balanceamento de carga, são eles o HMaster, Region Server e o ZooKeeper.

HMaster

HMaster é um processo leve que atribui regiões a servidores de região no cluster para balanceamento de carga. Suas responsabilidades são:

- Gerenciar e monitorar o cluster
- Disponibiliza interface para administração de tabelas (criar, atualizar, e excluir tabelas).
- Controlar failover.
- Tratar operações DDL.
- Responsável pelas operações de alteração de esquema ou metadados.

Servidor de Região (Region Server)

Estes são os nós de trabalho que lidam com pedidos de leitura, gravação, atualização e exclusão de clientes. O processo do Region Server é executado em todos os nós do cluster hadoop. O Region Server é executado no HDFS DataNode e consiste nos seguintes componentes:

- Cache de bloco - esse é o cache de leitura. Os dados lidos com mais frequência são armazenados no cache de leitura e, sempre que o cache do bloco estiver cheio, os dados usados recentemente serão despejados.
- MemStore - Este é o cache de gravação e armazena novos dados que ainda não foram gravados no disco. Cada família de colunas em uma região possui um MemStore.
- Write Ahead Log (WAL) é um arquivo que armazena novos dados que não são mantidos no armazenamento permanente.
- HFile é o arquivo de armazenamento real que armazena as linhas como valores de chave classificados em um disco.

ZooKeeper

O HBase usa o ZooKeeper como um serviço de coordenação distribuído para atribuições de região e para recuperar qualquer falha no servidor de região, carregando-o em outros servidores de região que estão funcionando. O ZooKeeper é um servidor de monitoramento centralizado que mantém as informações de configuração e fornece sincronização distribuída. Sempre que um cliente deseja se comunicar com as regiões, ele precisa abordar o Zookeeper primeiro. Os servidores HMaster e Region são registrados no serviço ZooKeeper, o cliente precisa acessar o quorum do ZooKeeper para conectar-se aos servidores da região e ao HMaster. No caso de falha do nó em um cluster HBase, o ZKquoram acionará mensagens de erro e começará a reparar os nós com falha.

O serviço ZooKeeper mantém o controle de todos os servidores da região que estão em um cluster HBase - informações sobre quantos servidores da região existem e quais servidores da região estão mantendo qual DataNode. O HMaster entra em contato com o ZooKeeper para obter os detalhes dos servidores da região. Vários serviços que o Zookeeper fornece incluem:

- Estabelecendo comunicação do cliente com os servidores da região.
- Rastreando falhas de servidor e partições de rede.
- Manter informações de configuração
- Fornece nós efêmeros, que representam servidores de região diferentes.

Propriedades

[7] O HBase não é um banco de dados compatível com transações ACID. No entanto, garante certas propriedades específicas de Atomicidade, Consistência, Isolamento e Durabilidade.

Atomicidade

Todas as mutações são atômicas dentro de uma linha. Qualquer escrita terá sucesso total ou fracasso. Uma operação que retorna um código de “êxito” foi completamente bem-sucedida. Uma operação que retorna um código de “falha” falhou completamente. Uma operação que atinge o tempo limite pode ter sido bem-sucedida e pode ter falhado. No entanto, ele não será parcialmente bem-sucedido ou falhou. Isso é verdade mesmo se a mutação cruzar várias famílias de colunas em uma linha

Consistência e Isolamento

Todas as linhas retornadas por qualquer leitura consistirão em uma linha completa que existia em algum momento no histórico da tabela. Isso ocorre entre famílias de colunas, ou seja, obter uma linha completa que ocorre simultaneamente com algumas mutações retornará uma linha completa que existia em algum momento entre a mutação

Durabilidade

Todos os dados visíveis também são duráveis. Ou seja, uma leitura nunca retornará dados que não foram tornados duráveis no disco. Qualquer operação que retorne um código de “êxito” (por exemplo, não lança uma exceção) será tornada durável. Qualquer operação que retorne um código de “falha” não se tornará durável (sujeito às garantias de atomicidade). Todos os cenários razoáveis de falha não afetarão nenhuma das garantias deste documento.

Onde usar e onde não usar o HBase?

O HBase é usado no Facebook Messenger por suas propriedade de boas escalabilidade e bom desempenho.
O Facebook detalha que dois aspectos importantes ajudaram a usá-lo:
Um conjunto curto de dados temporais que tende a ser volátil. Um conjunto crescente de dados que raramente é acessado.[8]

Se você achar que seus dados estão armazenados em coleções, por exemplo, alguns metadados, dados de mensagens ou dados binários todos com o mesmo valor, então você deve considerar o HBase.

O HBase não é um bom substituto a banco de dados relacionais se o uso que você dará a ele for o mesmo que se espera de bancos desse tipo.
O HBase pode requerer uso intensivo de memória e CPU em momentos de leitura e escrita elevado o que pode exigir equipamentos robustos para manter as características importantes de desempenho do banco.[9]

Glossário

NameNode: Nome atribuído ao HMaster pela ferramenta Cloudera Manager.
DataNodes: Nome atribuído ao Slave (RegionServer) pela ferramenta Cloudera Manager.
VMs: Máquina virtual.

Hadoop: Plataforma de software em Java de computação distribuída voltada para clusters e processamento de grandes volumes de dados.
Timestamp: Registro de data e hora.
HDFS: Sistema de arquivos distribuídos do Haddop
Master: Nó principal. Mestre.
Slave: Nó de distribuição. Servidor de região.
Failover: Servidor que assume determinado serviço se outro servidor tem problema.
DDL: Linguagem de definição de dados. Linguagem para alterar os esquemas de tabelas no banco de dados. (Ex. Alter Table, Create Database, etc).

Referências

[1] https://hbase.apache.org/book.html
[2] https://www.dezyre.com/article/overview-of-hbase-architecture-and-its-components/295
[3] https://pplware.sapo.pt/informacao/apache-hbase-not-only-sql-nosql/
[4] https://www.youtube.com/watch?v=V1fXSCASVDc
[5] https://www.cloudera.com/products/open-source/apache-hadoop/apache-hbase.html
[6] https://hadoop.apache.org/
[7] https://mapr.com/blog/in-depth-look-hbase-architecture/
[8] https://www.facebook.com/note.php?note_id=454991608919
[9] https://blog.cloudera.com/apache-hbase-dos-and-donts/

Instalação e configuração do ambiente

[5] Para usar o HBase, utilizaremos máquinas virtuais com sistema operacional CentOS e a ferramenta Cloudera Manager, que implementa um ambiente com todos os recursos necessários para o funcionamento do HBase. Também utilizaremos o editor de texto Vi neste tutorial, procure se familiarizar com os seus comandos antes de prosseguir.

Passo 1 - Criar 4 máquinas virtuais (1 NameNode, 3 DataNodes):

Para criação das máquinas virtuais (VMs), utilizaremos o**VMware® Workstation 15 Player (15.5.0 build-14665864)** com as seguintes configurações de sistema operacional e hardware:

- Sistema Operacional: CentOS Linux release 7.7.1908 (CentOS-7-x86_64-DVD-1908.iso)
- Hardware (Namenode):

- Hardware (DataNodes):

Credenciais:

```
-   Senha do root: cloudera

-   Usuário: cloudera - (Deve ser administrador)

-   Senha: cloudera
```

Passo 2: Dar privilégio de “sudo sem senha” para o usuário “cloudera” nas 4 VMs

Isso deve ser feito para que o Cloudera Manager possa fazer alterações no sistema operacional, durante sua execução.

Abrir o terminal e digitar o comando:

```
[cloudera@localhost ~]$ sudo visudo
```

O arquivo de sudoers do sistema será aberto. Insira na última linha:

```
cloudera ALL=(ALL) NOPASSWD:ALL
```

Salve o arquivo e feche o editor. Isso dá ao usuário “cloudera” todas as permissões de sudo sem pedir senha.

Passo 3: Configurar os hostnames das 4 máquinas

Em cada uma das VMs, configure o hostname da máquina. Sugerimos os nomes master.hb, slave1.hb, slave2.hb e slave3.hb. Para configurar, utilize o comando abaixo:

```
[cloudera@localhost ~]$ sudo hostnamectl set-hostname <nome escolhido>
```

Edite o arquivo /etc/hosts para que cada VM conheça os hostnames das outras VMs. Adicione no começo do arquivo, o IP, o hostname completo e o hostname básico de cada VMs. O arquivo terá a mesma alteração nas 4 VMs. Exemplo:

```
172.16.217.131 master.hb master

172.16.217.132 slave3.hb slave3

172.16.217.129 slave2.hb slave2

172.16.217.128 slave1.hb slave1
```

Procure não alterar outras linhas já presentes no arquivo. Apenas adicione as linhas acima no começo.

Em cada VM, edite o arquivo etc/sysconfig/network com o respectivo hostname completo de cada máquina. Para isso, basta adicionar a linha no começo do arquivo. Por exemplo, na master.hb, adicionaremos ao arquivo, a linha:

```
HOSTNAME=master.hb
```

Passo 4: Desabilitar o Firewall em todas as VMs

Para desabilitar o Firewall, em cada uma das VMs, entre com os comandos:

```
$ sudo systemctl disable firewalld

$ sudo systemctl stop firewalld
```

Isso desabilitará o Firewall e impedirá que ele seja ativado, caso o host reinicie.

Passo 5: Sincronizar os hosts.

Em cada um dos hosts, faça o procedimento abaixo.

Para que o cluster funcione bem, todos os hosts devem estar sincronizados. Para isso usaremos o ntp, Network Time Protocol. Primeiro instale o serviço em cada host:

```
$ sudo yum install ntp -y
```

Edite o arquivo /etc/ntp.conf para adicionar os servidores NTP, como no exemplo:

server 0.pool.ntp.org

server 1.pool.ntp.org

server 2.pool.ntp.org

Ative o serviço e configure-o para que seja ativado sempre que o host bootar:

```
$ sudo systemctl enable ntpd

$ sudo systemctl start ntpd
```

Sincronize o relógio do sistema com o servidor ntp, e então sincronize o relógio do hardware também:

```
$ ntpdate -u 0.pool.ntp.org

$ hwclock --systohc
```

Passo 6: Instalar o Cloudera Manager no master.hb.

Execute os próximos passos apenas no host master.hb.

Primeiro, instale o wget, para baixar arquivos via terminal:

```
$ sudo yum install wget -y
```

Agora, baixe o instalador do Cloudera Manager:

```
$ wget https://archive.cloudera.com/cm6/6.2.1/cloudera-manager-installer.bin
```

Altere as permissões para executar o arquivo, e então, execute-o:

```
$ chmod u+x cloudera-manager-installer.bin

$ sudo ./cloudera-manager-installer.bin
```

Leia os termos de compromisso e clique em Aceitar -> Aceitar -> Sim. A instalação iniciará:

Ao final da instalação, clique em OK para encerrar.

Passo 7: Criar o Cluster

Fora das VMs, no browser de sua máquina, acesse a Web UI do Cloudera Manager. Para isso, digite no browser o IP do master.hb e a porta 7180 ou 7183. Por exemplo: 172.16.217.131:7180.

Ao ser redirecionado, entre com usuário “admin” e senha “admin”. Você será redirecionado para a Instalação de Cluster. Clique em Continuar, forneça o nome para o novo cluster e clique novamente em Continuar.

Agora especifique o host com o qual iniciará o Cluster, inserindo o ip do master.hb. Selecione-o e clique em Continuar:

Mantenha a opção de Repositório Público da Cloudera e clique em Continuar. Na tela seguinte, leia os termos de compromisso e selecione a opção de instalação do JDK e clique em Continuar.

A seguir, forneça as credenciais como estão na figura abaixo e clique em Continuar. Obs: senha “cloudera”.

A seguir, a instalação do cluster começará. Aguarde a conclusão.

Por último, escolha os serviços que serão instalados no seu host. Instalaremos o Hdfs, Yarn, Zookeeper e, logicamente, o HBase. A máquina master.hb ficará com os seguintes papéis:

Passo 8: Adicionar os outros 3 hosts

Clique em “Add Hosts” e será redirecionado para a página de registro de hosts. Selecione a opção “Add hosts to cluster” e selecione no dropbox o cluster criado anteriormente e clique em Continuar.

Na próxima tela, forneça os IPs dos outros 3 hosts, de forma semelhante ao feito no passo anterior. Nas próximas etapas, siga as mesmas configurações do passo anterior até o momento de escolher Serviços.

Agora daremos alguns papéis para os 3 hosts, de acordo com a figura abaixo:

Finalmente, teremos nosso cluster funcionando, com Hadoop e HBase.

Utilização HBase

Passo 1: Entrar no HBase Shell

Através do host master.hb, no terminal, digite:

```
$ hbase shell
```

Passo 2: Criar uma tabela

```
$ create '<nome da tabela>','<família de colunas>'
```

Passo 3: Inserir valores nas colunas da tabela

```
put '<nome da tabela>', '<row key>', '<familia de colunas:coluna>', '<valor>'
```

Exercícios

1 - Conecte no HBase via Hbase Shell.

2 - Crie uma tabela para Funcionários de uma empresa. Esse funcionário terá dados pessoais e dados profissionais (Dica: Famílias de Colunas).

3 - Popule esta tabela com informações de 3 funcionários.

a.) Um gerente, de 28 anos, chamado Roberto Ramos, do departamento de TI, com salário de R$1000.

b.) Um recepcionista, do departamento de Administração, com salário de R$500, 23 anos, chamado Marcos Souza.

c.) Uma supervisora, do departamento de Compras, 30 anos, salário de R$1500, chamada Celina Mendes.

4 - Retorne todos os dados da tabela de funcionários.

5 - Retorne os dados do recepcionista.

6 - Desligue uma das VMs (slave1, slave2 ou slave3) e verifique se ainda tem acesso aos dados.

7 - Desligue outra das VMs e verifique se, com a última restante, ainda tem acesso aos dados.

8 - Religue as outras duas VMs.

9 - Apague a tabela de funcionários