FIAP

Sistemas de informação

Enterprise analytics and data warehousing

Profº Fabiano J. Cury Marques

FIAP

NTERPRISE ANALYTICS AND DATA WAREHOUSING

NoSQL Column Oriented

FIAP

Agenda

- ✘ Introdução
- ✘ Cassandra
- ✘ Hands On
- ✘ Atividade de Projeto I
- ✘ Referências

Introdução

Introdução

FIAP

Antes de começas, por quê utilizar NoSQL?

Deixe a VM do cassandra instalando...

C:\>**mkdir vagrant_cassandra** 1

C:\ >**cd vagrant_cassandra**

C:\vagrant_cassandra>

C:\vagrant_ cassandra >**vagrant init tcthien/java-dev-server** 2

A `Vagrantfile` has been placed in this directory. You are now ready to `vagrant up` your first virtual environment! Please read the comments in the Vagrantfile as well as documentation on `vagrantup.com` for more information on using Vagrant.

C:\vagrant_cassandra >**vagrant up** 3

Utilizaremos a plataforma online do Cassandra

- Acesse:

https://www.datastax.com/

FIAP

Justificativas para o uso de NoSQL

- ✘ Dados com estrutura indefinida: A maioria das implementações de NoSQL oferecem representação de dados sem estrutura definida. Isso atribui uma responsabilidade maior à aplicação que consome os dados e menor ao repositório de armazenamento;
- ✘ Tempo de Desenvolvimento: O tempo tende a diminuir considerando a inexistência de queries complexas em SQL, muitas vezes necessárias, para relacionar um conjunto expressivo de tabelas (modelo normalizado);
- ✘ Velocidade: A velocidade tem se tornado fator vital para sistemas envolvendo a conexão de uma quantidade expressiva de dispositivos mobile;
- ✘ Planejamento de Escalabilidade: Muitas empresas oferecem serviços de processamento/armazenamento com preços variados levando em conta o número de nós utilizado;

Escalabilidade

- Escalonar verticalmente (scale up/down): Está relacionado ao uso de mais ou de menos recurso computacional (processador e memória) para um único nó de processamento;
- Escalonar de forma vertical também inclui o uso de tecnologias de virtualização;

- Escalonar horizontalmente (scale out/in): Envolve a utilização da quantidade de nós de processamento no sistema;
- Sistemas que permitem escalonamento horizontal suportam o uso de uma quantidade maior de computadores para realizar determinadas tarefas. A amazona oferece um serviço "Elástico" (EC2) para a utilização de mais máquinas de forma transparente.

Tabela de dispersão

- Uma tabela de dispersão (Hash Table) é uma estrutura de dados que associa uma chave à um valor;
- Em geral, tabelas hash permitem realizar buscas extremamente rápidas a partir da chave fornecida para encontrar o valor desejado;
- A estratégia principal se baseia em um algoritmo denominado função de dispersão (hash) que mapeia uma chave de tamanho variável para uma chave de tamanho fixo (geralmente numérica);
- Tipicamente, a complexidade da busca é de tempo constante O(1), independentemente do tamanho da tabela;

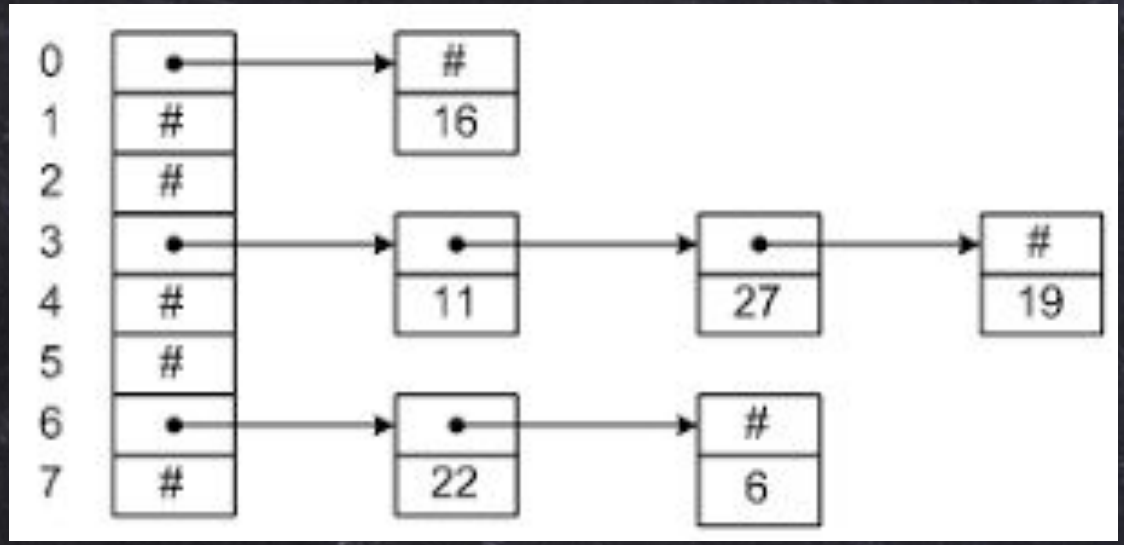
Tabela de dispersão

- Tipicamente as chaves são convertidas em números que correspondem aos índices de uma tabela:

001	**Maria | Arquiteto**
002	Adão | Vendedor
003	José | Alfaiate
004	Carla | Balconista
005	marta | joalheiro
. . .	. . .
875	thomas | carpinteiro
876	pedro | serralheiro

Tabela de dispersão

- Eventualmente a função de espalhamento não é perfeita e produz índices repetidos, tratados como conflito;



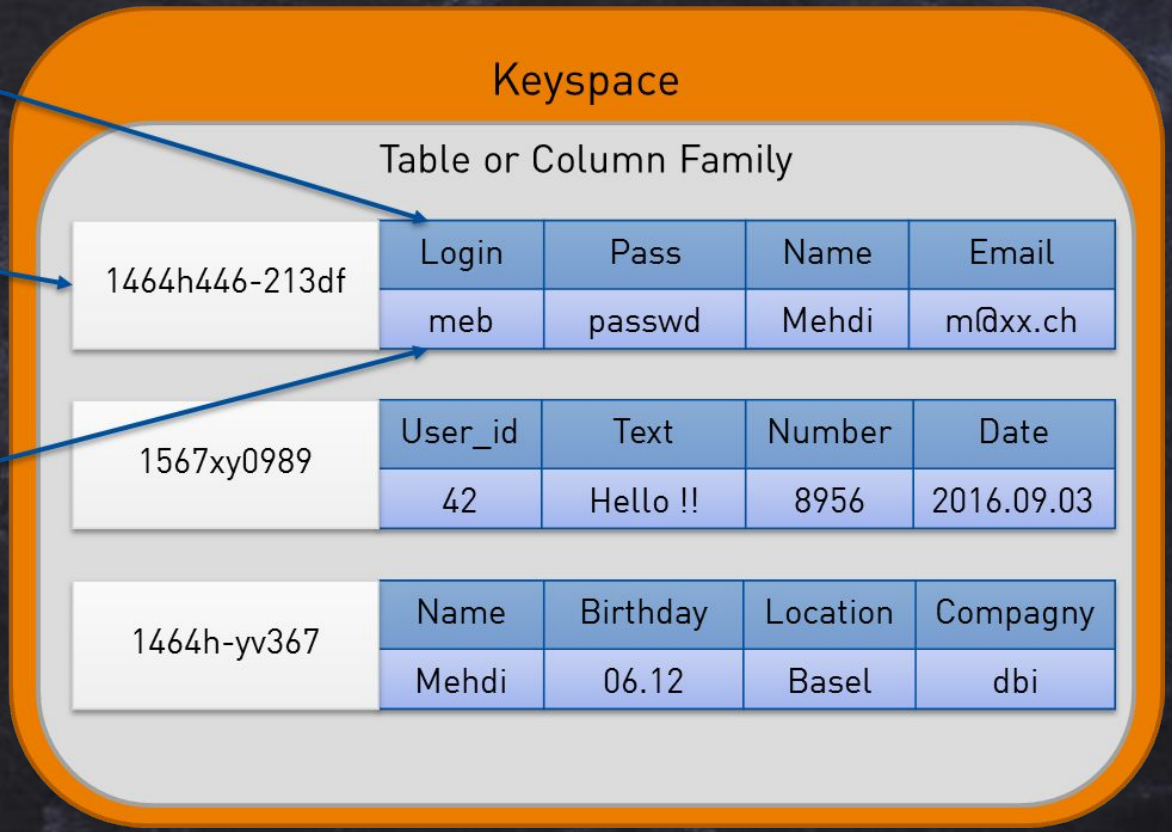
- Em geral, os conflitos são tratados por meio de listas ligadas;
- Quanto menor o número de conflitos melhor o desempenho;

Tabela de dispersão

Column key

Row key

Column value



1464h446-213df	Login	Pass	Name	Email
	meb	passwd	Mehdi	m@xx.ch

1567xy0989	User_id	Text	Number	Date
	42	Hello !!	8956	2016.09.03

1464h-yv367	Name	Birthday	Location	Compagny
	Mehdi	06.12	Basel	dbi

FIAP

Cassandra

Apache Cassandra Project

Cassandra

- Apache Cassandra é um banco de dados distribuído de segunda geração;
- Com design completamente distribuído permite alta escalabilidade;
- Apesar de ter sido desenhado para rodar em múltiplos servidores, Cassandra pode ser instalada em um computador para realizar alguns experimentos locais;

Apache CASSANDRA™

FIAP

Cases Netflix

- Substituição do DB Oracle pelo DB Cassandra:
  - Capacidade de 1.000.000 de escritas por segundo;
  - Latência – menos de 0.015 milissegundos;
  - Custo EC2 Amazon – $60/hora (cluster de 48 nós);
  - Custo por nó – $1.25/hora;
  - Incluindo armazenamento de 128.8 Tera bytes;
  - Incluindo Banda de rede read de 22 Mbps;
  - Incluindo Banda de rede escrita de 18.6 Mbps;

FIAP

Cassandra vs RDBMS

Banco de dados relacional	Cassandra
Recebimento de dados em baixa velocidade	Recebimento de dados em alta velocidade
Suporta dados provenientes de um ou poucos locais	Suporta dados provenientes de muitos lugares
Gerencia dados estruturados	Gerencia dados estruturados, não-estruturados e semiestruturados
Suporta transações complexas (joins)	Suporta transações simples
Ponto único de falha com failover	Sem ponto único de falha
Controla dados com volume moderado	Controla dados em volume expressivo
Implantações centralizadas	Implantações descentralizadas
Transações escritas em local único	Transações escritas em muitos locais
Escalabilidade de leitura	Escalabilidade de leitura e escrita
Suporta crescimento vertical	Suporta crescimento horizontal

Cassandra em aplicações

- Mensageria:
  - Aplicações com expressivo envio/recebimento de dados envolvendo dispositivos mobile;
- Intenet of things (IoT):
  - Dispositivos localizados em diferentes locais enviando informações de sensores em alta velocidade;
- Catálogo de produtos e aplicativos de varejo:
  - Expressiva entrada e saída de carrinhos de compra;
- Analytics de media social e motor de recomendações:
  - Empresas online de media social para prover analise e recomendação à seus clientes;

FIAP

Cassandra - Conceitos

- **Nó** : Local onde o dado é armazenado;
- **Data Center** : Uma conjunto de nós;
- **Cluster** : Um conjunto de Data Centers;
- **Commit Log** : Utilizado para recuperação depois de falha. Todo dado é escrito no commit log;
- **Mem-table** : Depois de escrito no commit log, o dado é escrito na mem-table;
- **SSTable** : Quando a men-table atinge um determinado tamanho, o dado é descarregado em um arquivo SSTable em disco rígido;

FIAP

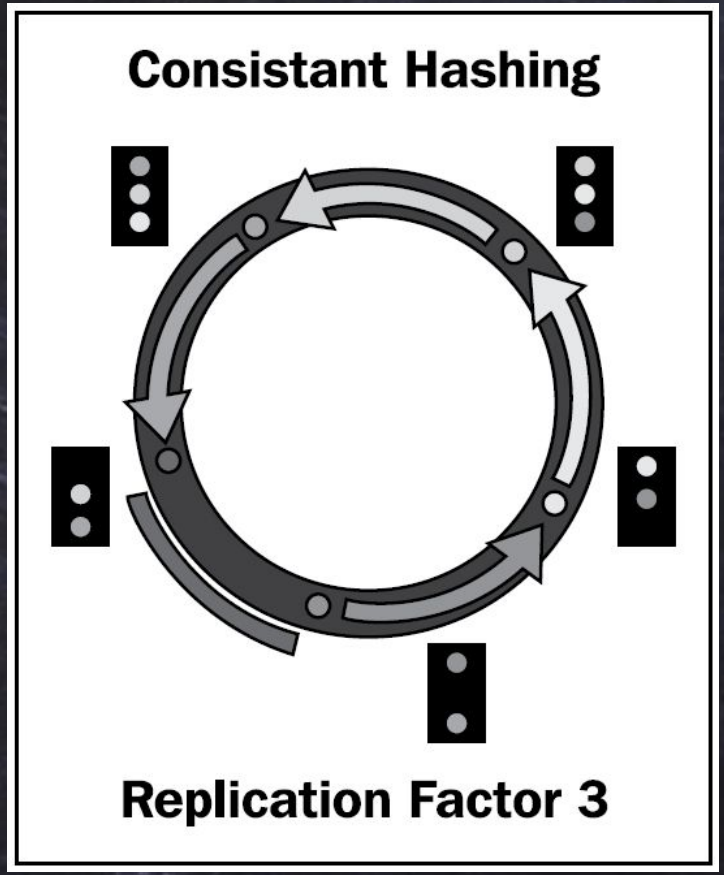
Cassandra – replicação de dados

- Para prevenir uma eventual falha de hardware o dado recebido é replicado para mais de um nó. Isso garante "*no single point of failure*";
- Cassandra determina a replica de dados a partir de duas definições:
  - Replication Strategy : Determina o local onde deve ser colocada a próxima replicação do dado;
  - Replication Factor : Determina o número total de replicas atribuídas à diferentes nós;
- Para um fator de replicação igual á um, deve haver uma cópia do dado no DB;
- Para um fator de replicação igual á três, deve haver três cópias do dado no DB;

Cassandra – Estratégia de replicação

- A estratégia de replicação simples (SimpleStrategy) é utilizada quando existe apenas um Data Center (um rack);

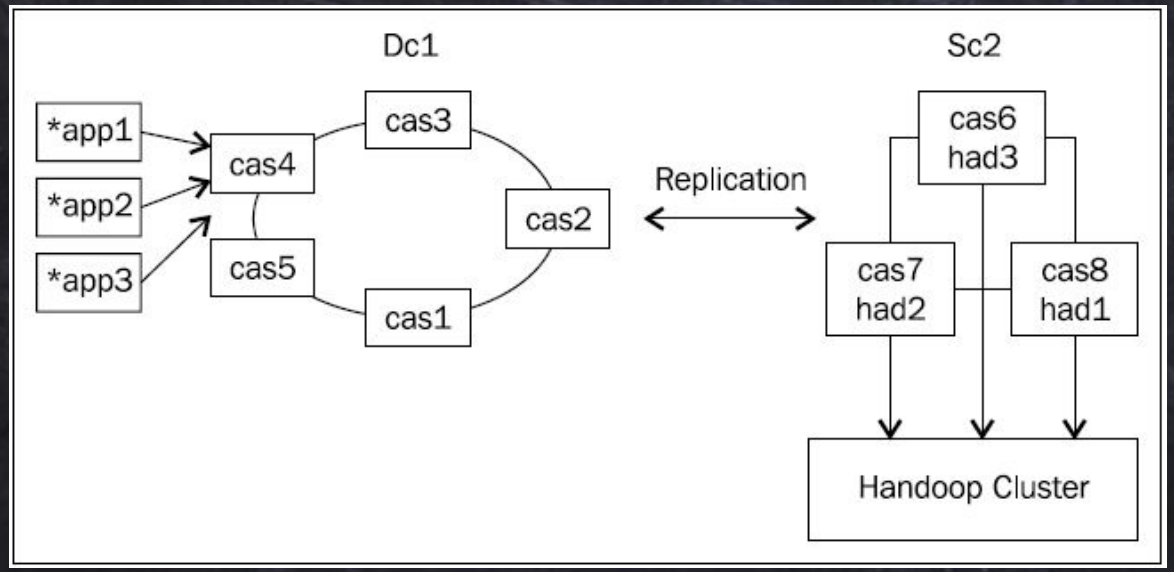
- As operações de replica seguem um sentido horário/anti-horário previamente definido em uma arquitetura anel;



Cassandra – Estratégia de replicação

- A estratégia de denominada NetworkTopologyStrategy é utilizada quando existe mais de um Data Center;

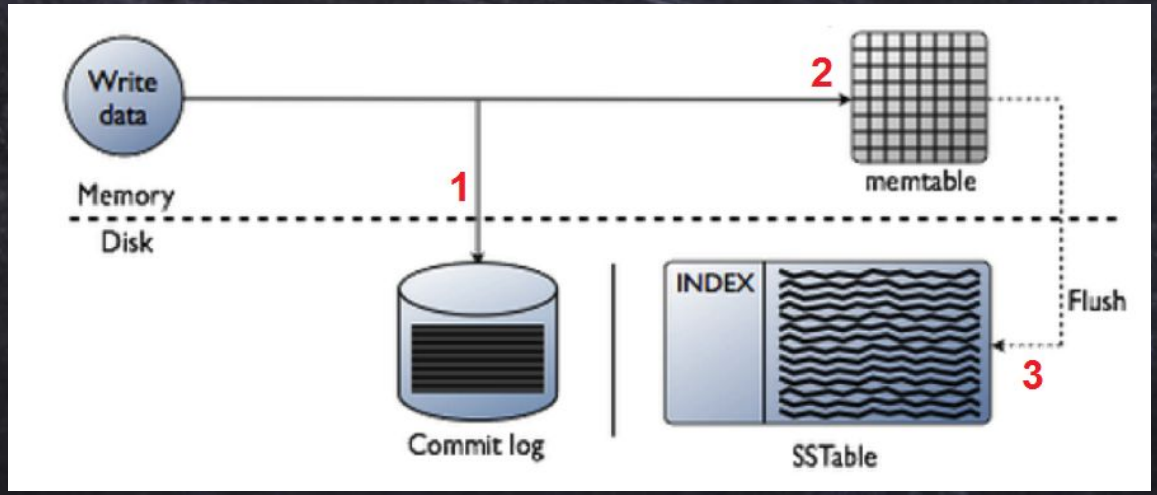
- A replicação ocorre a partir de um ponto do anel de um Data Center para outro;



Cassandra – Operação Escrita

- A escrita em um nó do DB segue a ordem apresentada abaixo:

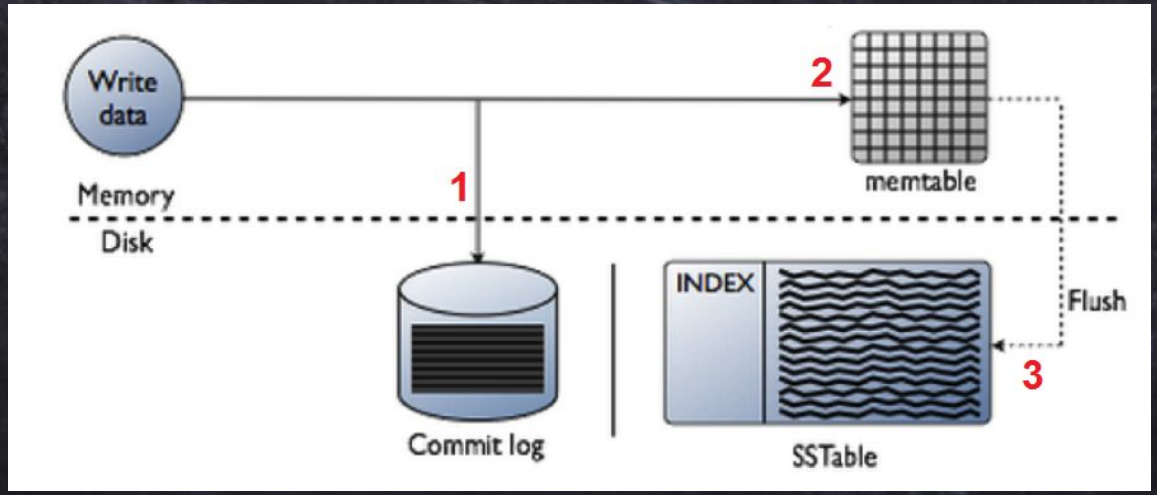
1. O dado é enviado para a região Commit log,
2. O dado é armazenado temporariamente na mem-table;
3. Os dados de uma mem-table cheia são armazenados em SSTable;



Cassandra – Operação Escrita

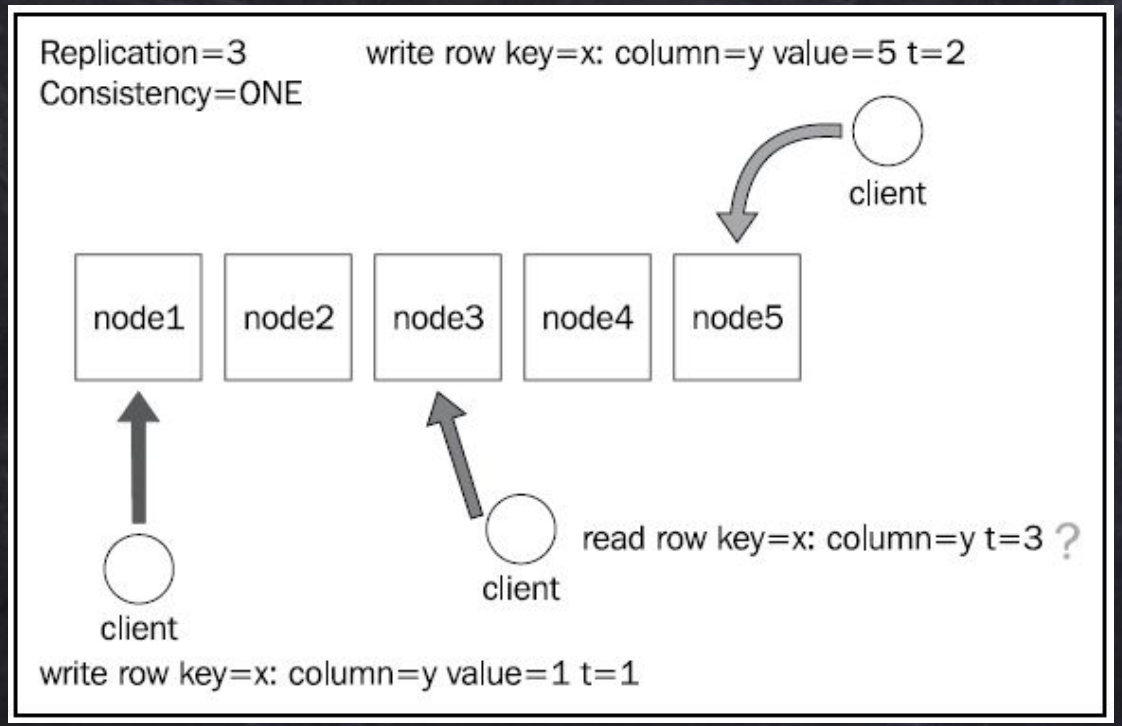
- Um valor de Nível de consistência (Consistency level) é definido para garantir um número mínimo de replicas necessárias para o "reconhecimento de sucesso";

- Uma réplica ativa responde a solicitação de escrita depois que escreveu na mem-table;



Cassandra – Operação Escrita

- Independentemente do nível de consistência, uma solicitação de escrita é envida para todas as replicas ativas;



Cassandra – Operação Leitura

- A operação de leitura apresenta um custo maior para o DB Cassandra;
- Isso ocorre porque um determinado nó precisa garantir que trata-se do último dado;
- Solicitações são feitas entre os nós levando em conta o Replication Factor;

FIAP

Servidor Cassandra

- ✘ Suba a instancia do vagrant a partir do prompt do dos:

  C:\vagrant_java-dev-server\ vagrant up

- ✘ Abra o terminal putty e acesse utilizando porta 127.0.0.1:2222 – usuário: vagrant, senha:vagrant
- ✘ No shell de comandos do putty vá ao caminho:

  ./home/vagrant/bin

- ✘ Inicialize o servidor cassandra:

  $ cassandra –f

- ✘ Abra um segundo terminal putty e acesse a utilizando porta 127.0.0.1:2222 – usuário: vagrant, senha:vagrant.
- ✘ Utilizando o segundo terminal putty, acesse o servidor Cassandra:

  $ cqlsh localhost –u cassandra –p cassandra

FIAP

Cassandra – Regras gerais do modelo

- A sintaxe de comando do DB Cassandra é parecida com a sintaxe SQL com algumas restrições;
- Cassandra não suporta JOINs, GROUP BY, clausulas OR, agregação;
- A criação de uma tabela, por exemplo, é feita utilizando o script ao lado:

```
CREATE TABLE PESSOA
(
    ID TEXT,
    EMAIL TEXT,
    NOME TEXT,
    SOBRENOME TEXT,
    PRIMARY KEY(ID)

);
```

Cassandra – Regras gerais do modelo

- A chave primária da tabela é utilizada para definir a partição (em qual nó um registro deve ficar);

- As demais colunas são utilizadas como clutering key e são utilizadas para um armazenamento ordenado;

```
CREATE TABLE PESSOA
(
    ID TEXT,
    EMAIL TEXT,
    NOME TEXT,
    SOBRENOME TEXT,
    PRIMARY KEY(ID)

);
```

Cassandra – Keyspace

- Uma keyspace é um objeto que suporta família de colunas, indexes, estratégias utilizadas, fator de replicação, etc.

- Uma keyspace pode ser criada a partir do comando exemplo abaixo:

```
CREATE KEYSPACE test with REPLICATION={'class':'SimpleStrategy','replication_factor': 3};
```

- Uma keyspace pode ser alterada a partir do comando exemplo abaixo:

```
ALTER KEYSPACE test with REPLICATION={'class':'NetworktopologyStrategy', 'DataCenter1':1};
```

- E pode ser destruída (destruindo todas as tabelas e dados)

```
DROP KEYSPACE test
```

Cassandra – CRUD

- **CREATE**: Insere um registro novo na tabela:

```
INSERT INTO KEYSPACENAME.TABLENAME(ColumnName1, ColumnName2, . . . .)
VALUES (Column1Value, Column2Value, . . . .)
```

- **READ**: faz a leitura de um registrou ou um conjunto de registros:

```
SELECT ColumnNames FROM KEYSPACENAME.TABLENAME
WHERE ColumnName1=Column1Value AND
      ColumnName2=Column2Value AND ... ;
```

- **UPDATE**: Atualiza o um ou mais campos de uma tabela:

```
UPDATE KEYSPACENAME.TABLENAME
SET ColumnName1 = Column1ValueNew,
    ColumnName2 = Column2ValueNew,
        . . .
WHERE ColumnName=ColumnValue;
```

Cassandra - CRUD

- DELETE: Remove um registro:

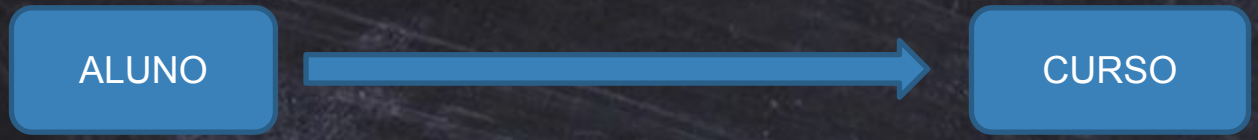
```
DELETE FROM KEYSPACENAME.TABLENAME
WHERE ColumnName1=ColumnValue;
```

- DELETE: Remove uma coluna de um determinado registro:

```
DELETE ColumnNames FROM KEYSPACENAME.TABLENAME
WHERE ColumnName1=ColumnValue;
```

Cassandra - relacionamentos

- UM PRA UM:




```
CREATE TABLE ALUNO_CURSO
(
    ALUNO_RM INT PRIMARY KEY,
    ALUNO_NOME TEXT,
    CURSO_NOME TEXT
);
```

Cassandra – relacionamentos

- UM PARA MUITOS:


```
CREATE TABLE ALUNO_CURSO
(
    ALUNO_RM INT,
    ALUNO_NOME TEXT,
    CURSO_NOME TEXT PRIMARY KEY
);
```

Cassandra – relacionamentos

- **MUITOS PARA MUITOS:**


```
CREATE TABLE ALUNO_CURSO
(
    ALUNO_RM INT PRIMARY KEY,
    ALUNO_NOME TEXT,
    CURSO_NOME TEXT
);
```

```
CREATE TABLE CURSO_ALUNO
(
    CURSO_NOME TEXT PRIMARY KEY,
    ALUNO_NOME TEXT,
    ALUNO_RM INT
);
```

Cassandra – Expiração de dados TTL

- É possível inserir um dado no DB Cassandra que expira depois de um tempo determinado em segundos;
- Essa funcionalidade é denominada Time to Live (ttl):
- Segue script utilizado:

```
INSERT INTO TABLENAME(ColumnNames)
        VALUES(ColumnValues) USING ttl TimeInseconds;
```

FIAP

Cassandra – Expiração de dados TTL

- Um índice pode ser construído para uma coluna da tabela exceto a coluna primary key. Exemplo;

```
CREATE INDEX INDEXNOME ON PESSOA (NOME);

CREATE INDEX INDEXSOBRENOME ON PESSOA (SOBRENOME);
```

- Um índice pode ser destruído. Exemplo:

```
DROP INDEX IF EXISTS INDEXNOME;

DROP INDEX IF EXISTS INDEXSOBRENOME;
```

FIAP

Tipos de dados

CQL Type	Constants	Description
ascii	Strings	US-Ascii character string
Bigint	Integers	64-bit signed long
Blob	Blobs	Arbitrary bytes in hexadecimal
Boolean	Booleans	True or false
Counter	Integers	Distributed counter values 64 bit
Decimal	Integers, floats	Variable precision decimal
Double	Integers, floats	64-bit floating point
Float	Integers, floats	32-bit floating point
Frozen	Tuples, collections, user	Stores cassandra types
Inet	Strings	IP address in IPV4 or IPV6 format
Int	Integers	32 bit signed integer
List		Collection of elements
Map		Json style collection of elements
Set		Collection of elements
Text	Strings	UTF-8 encoded strings
Timestamp	Integers, strings	Id generated with date plus time
Timeuuid	Uuids	Type 1 uuid
Tuple		A group of 2,3 fields
Uuid	Uuids	Standard uuid
Varchar	Strings	UTF-8 encoded string
Varint	Integers	Arbitrary precision integer

Cassandra – Tipo Set

- Um tipo Set é uma coleção de valores que são apresentados de forma ordenados, exemplo:

```
CREATE TABLE FUNCIONARIO
(
    ID TEXT,
    NOME TEXT,
    SOBRENOME TEXT,
    EMAIL SET<TEXT>,
    PRIMARY KEY(ID)
);

INSERT INTO FUNCIONARIO (ID, NOME, SOBRENOME, EMAIL) VALUES ('001','pedro da
silva','BONIFACIO',{'cllal@gmail.com', 'allal@hotmail.com', 'bllal@yahoo.com'}
);

SELECT * FROM FUNCIONARIO;
```

Cassandra – Tipo List

- Um tipo List é uma coleção de valores onde a ordem dos valores não tem importância, exemplo:

```
CREATE TABLE ALUNO
(
    ID TEXT,
    NOME TEXT,
    SOBRENOME TEXT,
    DISCIPLINA LIST<TEXT>,
    PRIMARY KEY(ID)
);

INSERT INTO ALUNO (ID, NOME, SOBRENOME, DISCIPLINA) VALUES ('007','Marta Rocha'
,'Silva',['calculo', 'algebra', 'bbb'] );

SELECT * FROM ALUNO;
```

Cassandra – Tipo Map

- Um tipo Map é uma coleção utilizada para armazenar pares chave/valores ordenados, exemplo:

```
CREATE TABLE CARRO
(
    ID TEXT,
    MODELO TEXT,
    DESCRICAO TEXT,
    COR_ACRESCIMO MAP<TEXT,float>,
    PRIMARY KEY(ID)
);

INSERT INTO CARRO (ID, MODELO, DESCRICAO, COR_ACRESCIMO) VALUES ('015','FIT',
'esportivo',{'preto':2.5, 'branco':0.0, 'amarelo':4.0} );

SELECT * FROM CARRO;
```

Atividade (opcional)

- ✘ Adapte o data mart de vendas para tabelas no Cassandra e realize uma carga de dados (fake) utilizando linguagem python a partir do ambiente repl.it.

Atividade Semanal

- A NAC da próxima semana de aula (dia 19/10) será uma atividade de seminário. Os alunos devem se dividir em grupos, máximo de 5 pessoas, para apresentar o resultado de uma atividade de pesquisa em no máximo 10 minutos. Cada grupo deve escolher um dos temas apresentados na lista abaixo:

1. Principais Aplicações envolvendo Big Data;
2. Aplicações utilizando NoSQL por chave valor;
3. Aplicações utilizando NoSQL por documento;
4. Aplicações utilizando NoSQL por representação de grafos;
5. Aplicações utilizando ElasticSearch;
6. Aplicações utilizando Tableau, PowerBI;
7. Aplicações utilizando AWS;
8. Principais Aplicações envolvendo Inteligência Artificial;

- A pesquisa deve ser realizada em torno das principais aplicações envolvendo esse tipo de armazenamento e seus principais desafios.
- O slide de cada grupo deve ser entregue até 30 minutos antes da próxima aula.

FIAP

Atividade Semanal (opcional)

- ✘ Defina um projeto de seu interesse utilizando o DB Cassandra. Para definir o seu projeto, considere as características, apresentadas em sala de aula, associadas à um problema que permitam justificar o uso de um banco de dados NoSQL.

- ✘ Deve ser entregue:

1. Arquivo texto contendo todos os scripts de criação das tabelas;
2. População das tabelas (ao menos dois registros) no formato INSERT;
3. Um documento Word, contendo:
   - Descrição do projeto/problema;
   - Justificativa do uso do DB Cassandra;
   - Descrição/Justificativa de cada tabela (mínimo três linhas)

Referências

- Peter A. Boncz, Martin L. Kersten, Stefan Manegold. Breaking the memory wall in MonetDB. communications of the acm, Dec. 2008, Vol. 51, N° 12.
- Introduction toMulticore architecture Tao Zhang Oct. 21, 2010
- Edward Capriolo. Cassandra High Performance Cookbook, packt publiching, Mumbai, 2011.
- Gaurav Vaish. Getting Started with NoSQL, packt publiching, Mumbai, 2013.
- https://www.dbi-services.com/blog/wp-content/uploads/sites/2/2022/04/Cassandra_dataModel.png acessador em 19/10/2023 às 19h00min

OBRIGADO!

https://br.linkedin.com/in/wakimsaba