APACHE CASSANDRA

Adriano Bonacin

Apresentação

yadax

Sobre este curso

- Cerca de 32 horas de muito Hands On
  - Necessário acesso à internet (porta 22)
- Introdução a NoSQL x Bigdata/Smartdata
- Instalando/Gerenciando o Serviço Apache Cassandra
- Camada de Dados - CQL
- Arquitetura Cassandra
- Cassandra Ops
- Cassandra Best Practices

yadax

Pré requisitos para o curso

- Conhecimento básico de Linux
  - Listar arquivos – ls
  - Ver conteúdo de um arquivo – cat
  - Editar arquivo – vi
  - Copiar/mover arquivos – cp/mv

- Não tem interface gráfica, somente tela preta ☺️

Introdução a NoSQL x Bigdata/Smartdata

yadax

Bancos Relacionais

- Por muitos anos foi a única solução
- Atendia praticamente todas demandas
- Escalar é caro, vertical
- Consistente, ACID
- Normalizado, Joins
- Integridade referencial
- Exemplos?

Vertical

Horizontal

yadax

ACID

- Atomic – Não existe meia transação
- Consistent – Todos veem o DB da mesma forma em dado momento
- Isolation – Privacidade nos registros alterados até o commit.
- Durable – Se o DB respondeu OK para seu commit, está OK mesmo em falha da instance.

yadax

ACID

- Atomic – Não existe meia transação
- Consistent – Todos veem o DB da mesma forma em dado momento
- Isolation – Privacidade nos registros alterados até o commit.
- Durable – Se o DB respondeu OK para seu commit, está OK mesmo em falha da instance.

yadax

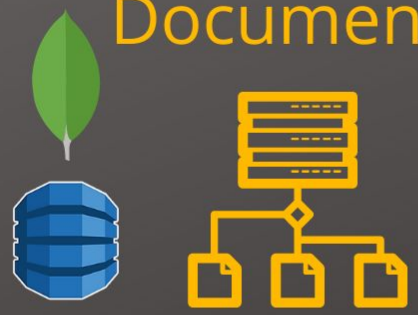
NoSQL

- Alta disponibilidade
- Escalabilidade
- Distribuídos
- Alto volume de dados
- Schema!?
- Normalização!?
- Relacionamento!?
- Not Only SQL

yadax

Tipos de NoSQL

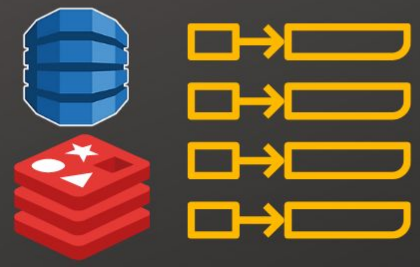
Documento



Grafo

yadax

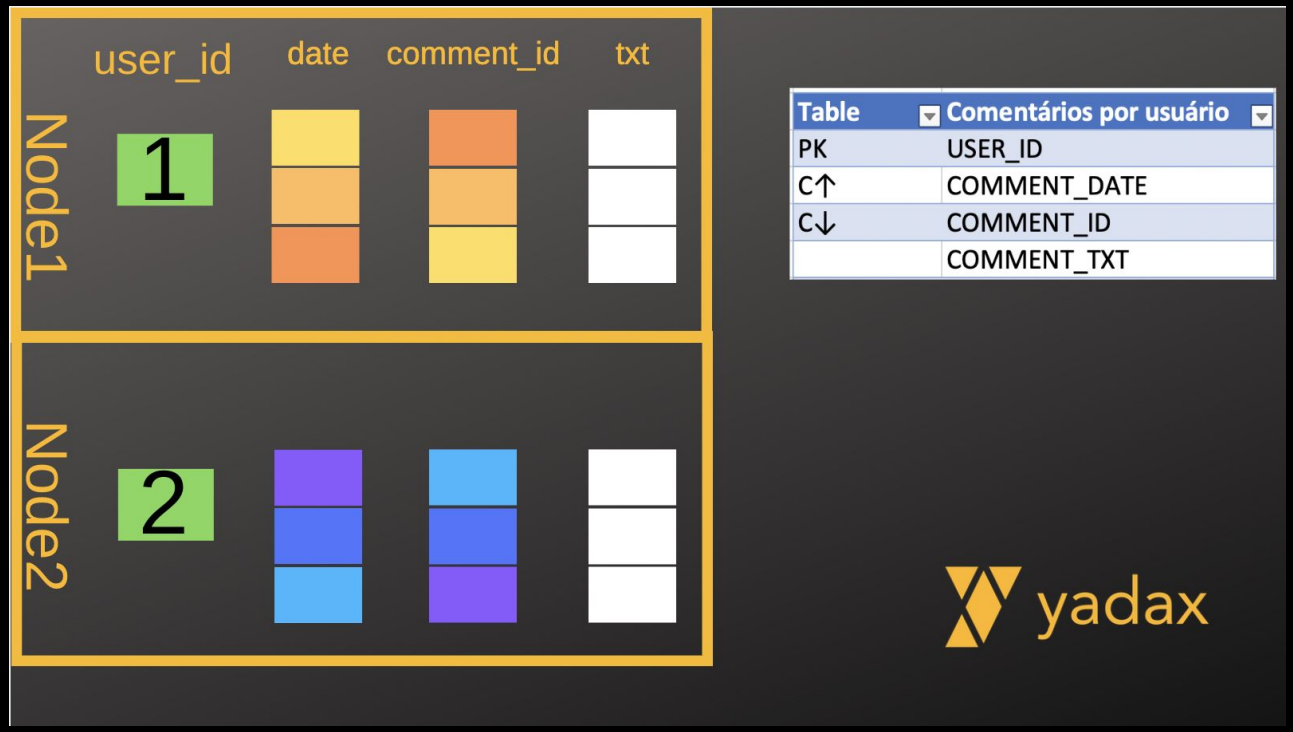
Chave Valor



Colunar

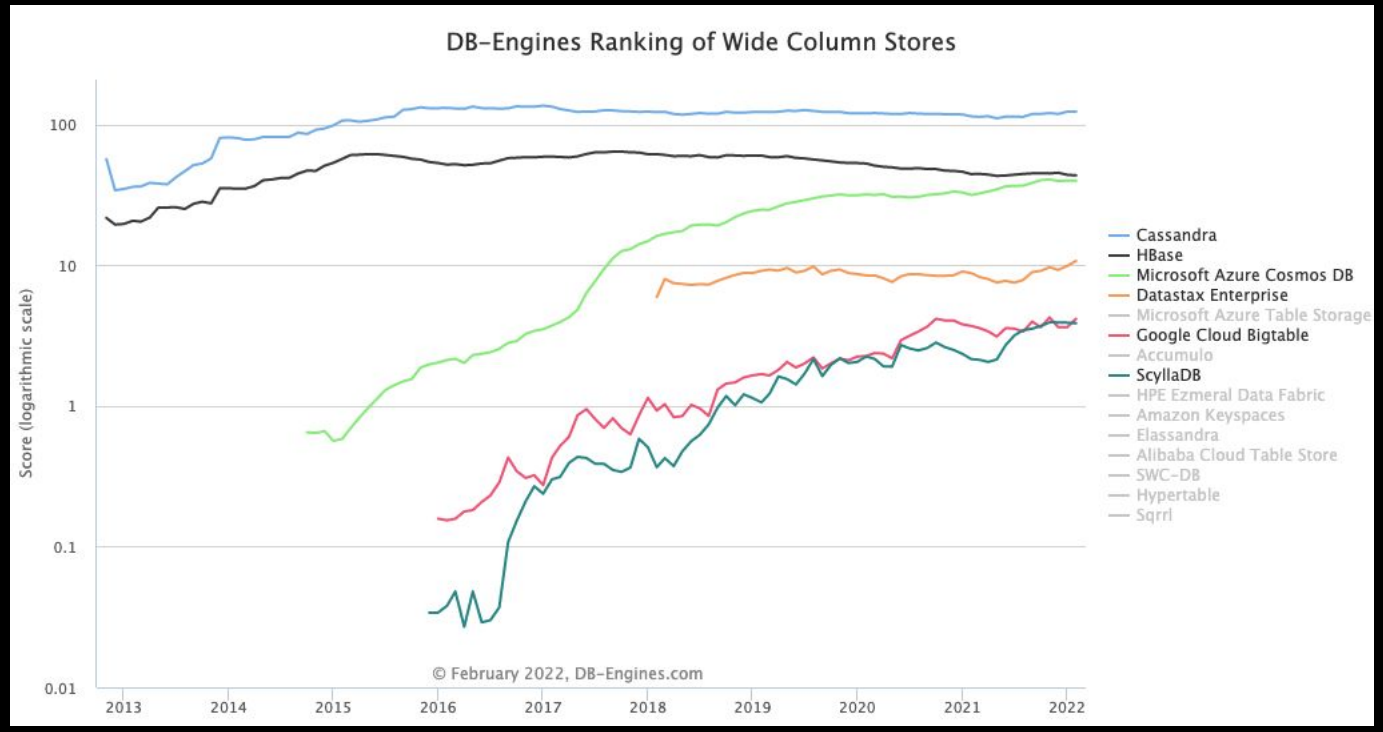
Wide column store

- Baseados BigTable do Google
- Ao invés de linhas, colunas (atributos)
- Suporta grande volume de requisições R/W
- VLDB
- Dados replicados
- Ex.: Cassandra, Hbase, Azure CosmosDB



yadax

Wide column store



yadax

Wide column store – Use Cases

- Spotify: Cassandra para armazenar User Profile e metadados de Músicas, Playlist, Artistas
- Facebook: Hbase para o guardar mensagens (Messenger) e para o Nearby Friends.
- Pagseguro: Cassandra para receber milhares de transações da Moderninha/minuto.

Key/Values

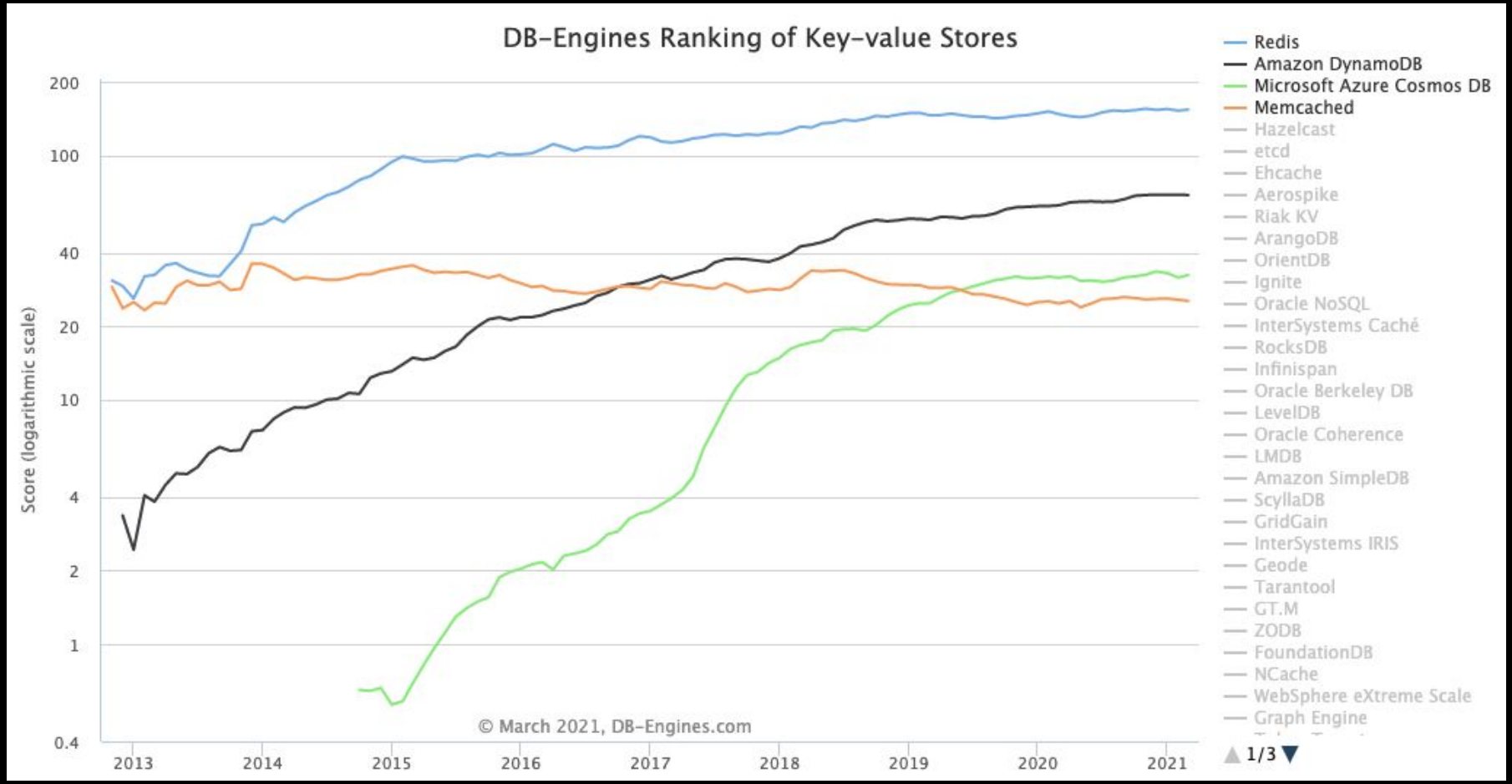
- Moledo mais simples
- Chave: Valor
- Maior taxa R/W entre os NoSQL
- Cache (resposta mais rápida)
- Ex.: MemCached, Redis, DynamoDb

KEY	VALUE
1	“https://yadax.com.br/imgs/foto.jpg”
2	123
3	{“id”: “abonacin”, “idade”: 36, ...}
4	[1, 23, 9, 1]
5	

Key/Values – Use Cases

- User profile
- Session Info
- Carrinho compra
- Detalhes top produtos
- IP x MAC Info
- Cache

Key/Values



yadax

Document

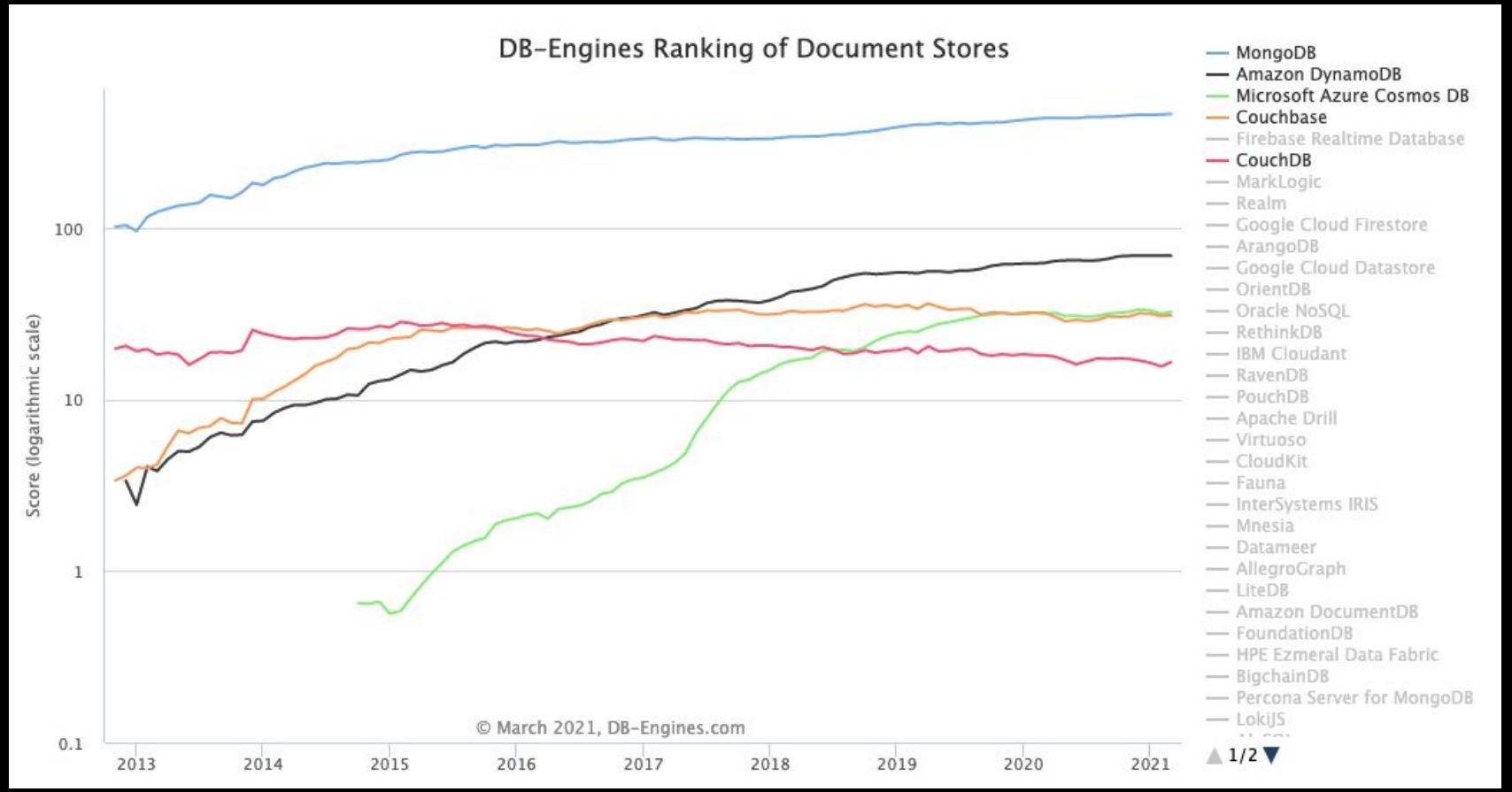
- Armazena docs
  - JSON
  - XML
  - YAML
  - ...
- Permite busca pelos atributos
- Permite doc dentro de doc
- Ex.: MongoDB, DynamoDB, CosmosDB

yadax

Document – Use Cases

- SEGA: MongoDB para gerenciar milhões de contas de usuário
- Weather Channel: usa MongoDB para notificar em tempo real mais de 40 milhões de usuários
- Genomics England usa MongoDB para permitir ciência de dados em 100,000 projetos de genomas
- Pagseguro: DynamoDB para tracking de emissão de cartões

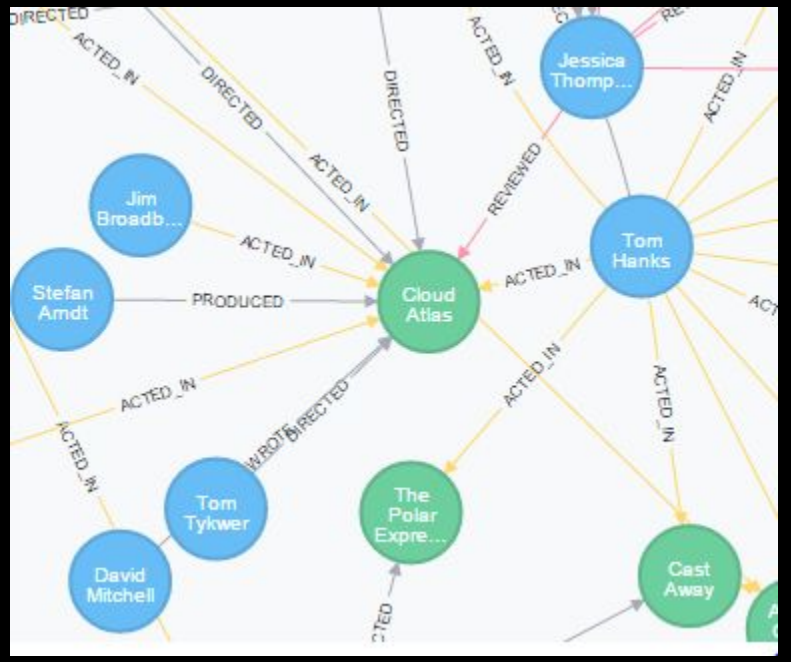
Document



yadax

Graph

- Armazenar elementos/relações
- Nós/Vértices: Elementos
- Arestas: Relacionamento
- Atributos: vértices e arestas
- Ex.: Neo4J, CosmosDB



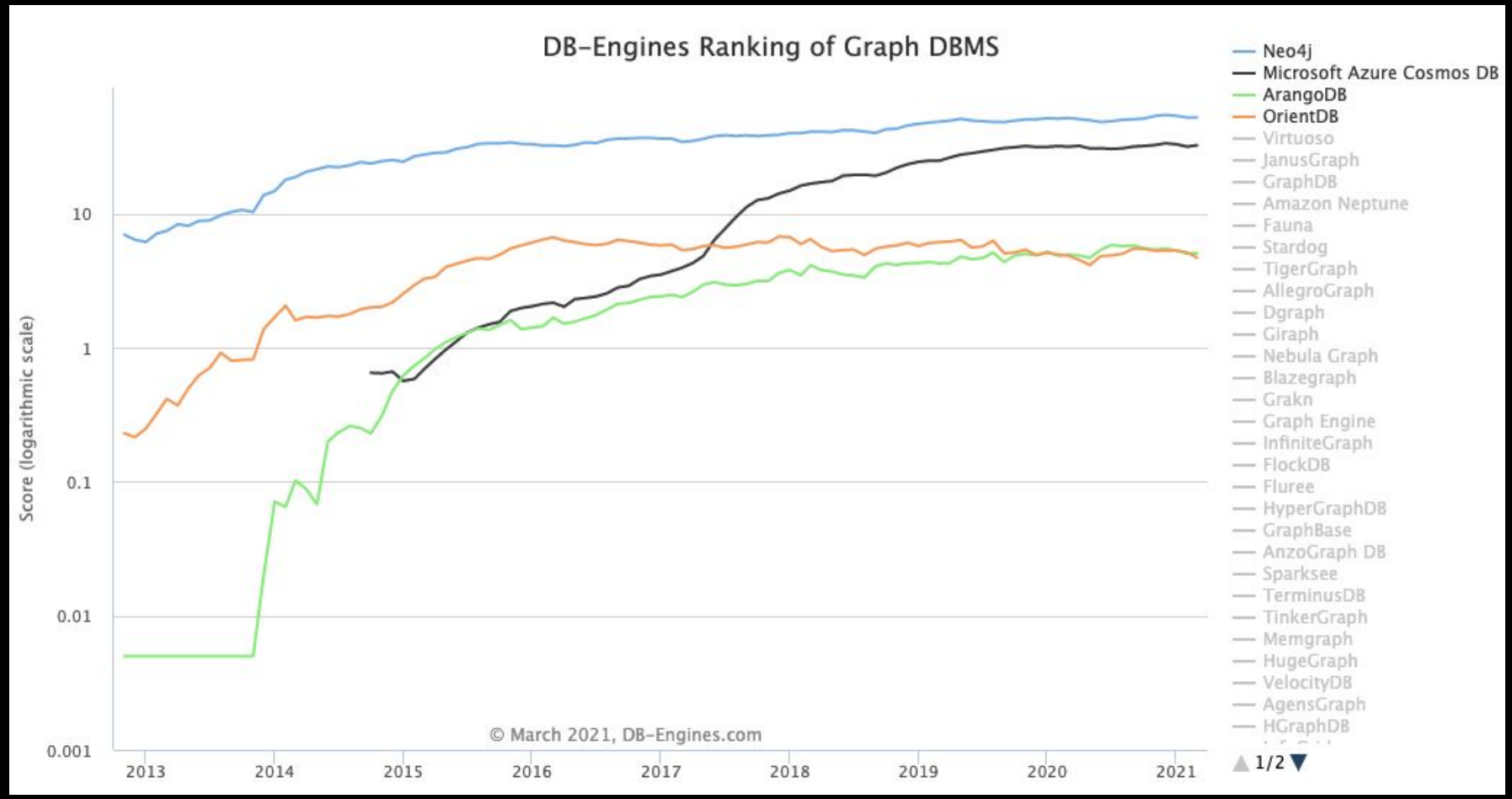
yadax

Graph - Use Cases

- Walmart: Neo4J para recomendação de produtos
- Medium: Neo4J para garantir conteúdo mais relevantes
- Cisco: Para mapear topologias e antecipar à falhas
- UOL: CMDB - para determinar relacionamento entre CIs

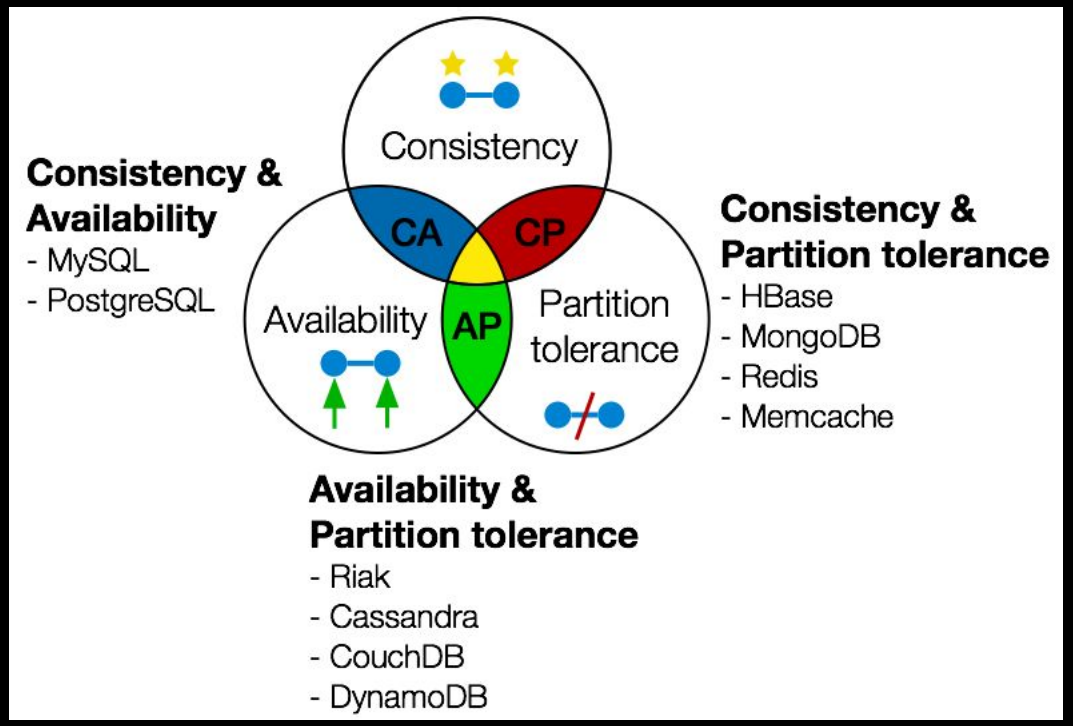
yadax

Graph


DB-Engines Ranking of Graph DBMS
Score (logarithmic scale)
10
1
0.1
0.01
0.001
2013
2014
2015
2016
2017
2018
2019
2020
2021
© March 2021, DB-Engines.com
Neo4j
Microsoft Azure Cosmos DB
ArangoDB
OrientDB
Virtuoso
JanusGraph
GraphDB
Amazon Neptune
Fauna
Stardog
TigerGraph
AllegroGraph
Dgraph
Giraph
Nebula Graph
Blazegraph
Grakn
Graph Engine
InfiniteGraph
FlockDB
Fluree
HyperGraphDB
GraphBase
AnzoGraph DB
Sparksee
TerminusDB
TinkerGraph
Memgraph
HugeGraph
VelocityDB
AgensGraph
HGraphDB
1/2

yadax

Teorema CAP

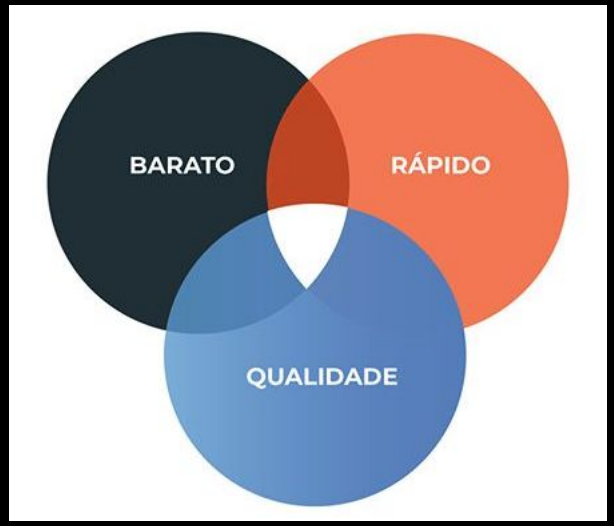


- Consistency: Todo mundo vê os dados da mesma forma
- Availability: Clientes conseguem ler e escrever
- Partition Tolerance: Vai continuar funcionando mesmo que tenha uma falha de comunicação entre os nodes

yadax

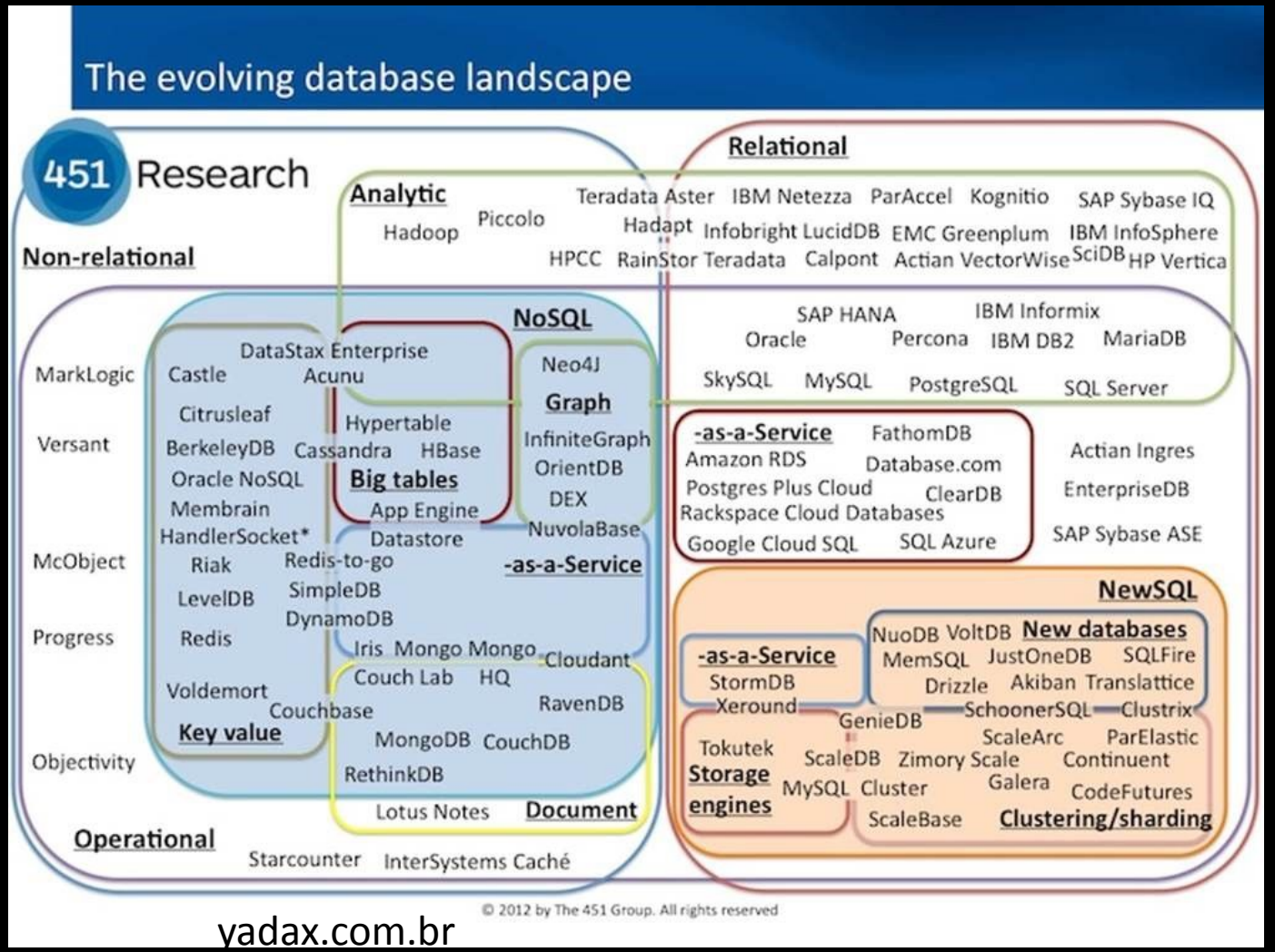
Teorema CAP

- Escolha dois



yadax

Uma infinidade



yadax.com.br

yadax

NoSQL x BigData

- Capacidade de escalar
- Volume da dados, TB/host
- Velocidade
- Nasceram em uma nova era

yadax

APACHE CASSANDRA

Vamos ao que interessa!

yadax

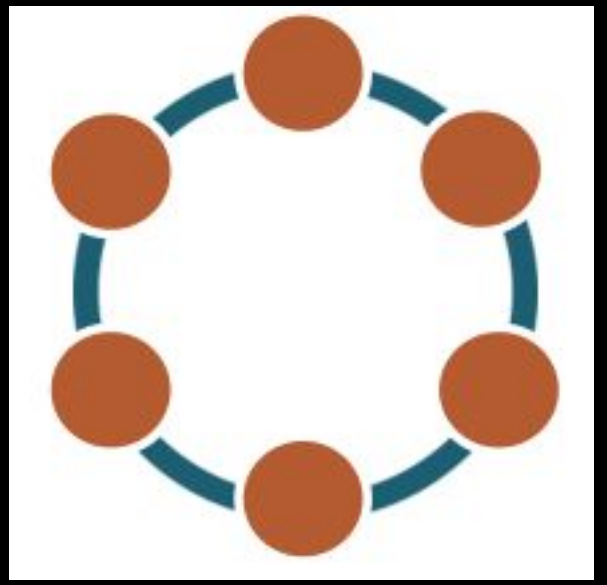
Distribuições Cassandra

- Apache – Open Source
  - 2.1
  - 2.2
  - 3.0
  - 3.11
  - 4.0
- Datastax – Enterprise
  - 6.8 (3.11)
  - 6.7 (3.11)
  - 6.0 (3.11)
  - 5.1 - EOSL
  - 5.0 - EOSL

yadax

Características do Cassandra

- Tolerância à falha
- Sem ponto único de falha
- Multi master
- Distribuído globalmente
- Escalonamento linear*
- Always writable*
- Linguagem similar ao SQL



yadax

Quando não usar

- Queries não usando PK, precisando de Secondary Indexes
- Agregação
- Join
- Locks
- Updates/Deletes
- Transação
- Consistência forte

yadax

Quando usar

- Escrita muito maior que Leitura
- Dados raramente atualizados
- Leitura apenas pela PK
- Dados devem ser particionados
- Sem Join
- Evite Index
- Precisa trabalhar globalmente
- TTL

yadax

Instalando/Gerenciando o Apache Cassandra

Primeiros passos!

yadax

Prereqs

- Última versão do Java 8
  - Oracle
  - Open-jdk
  - > 1.8.0_161
- Python + 2.7 (cqlsh)

yadax

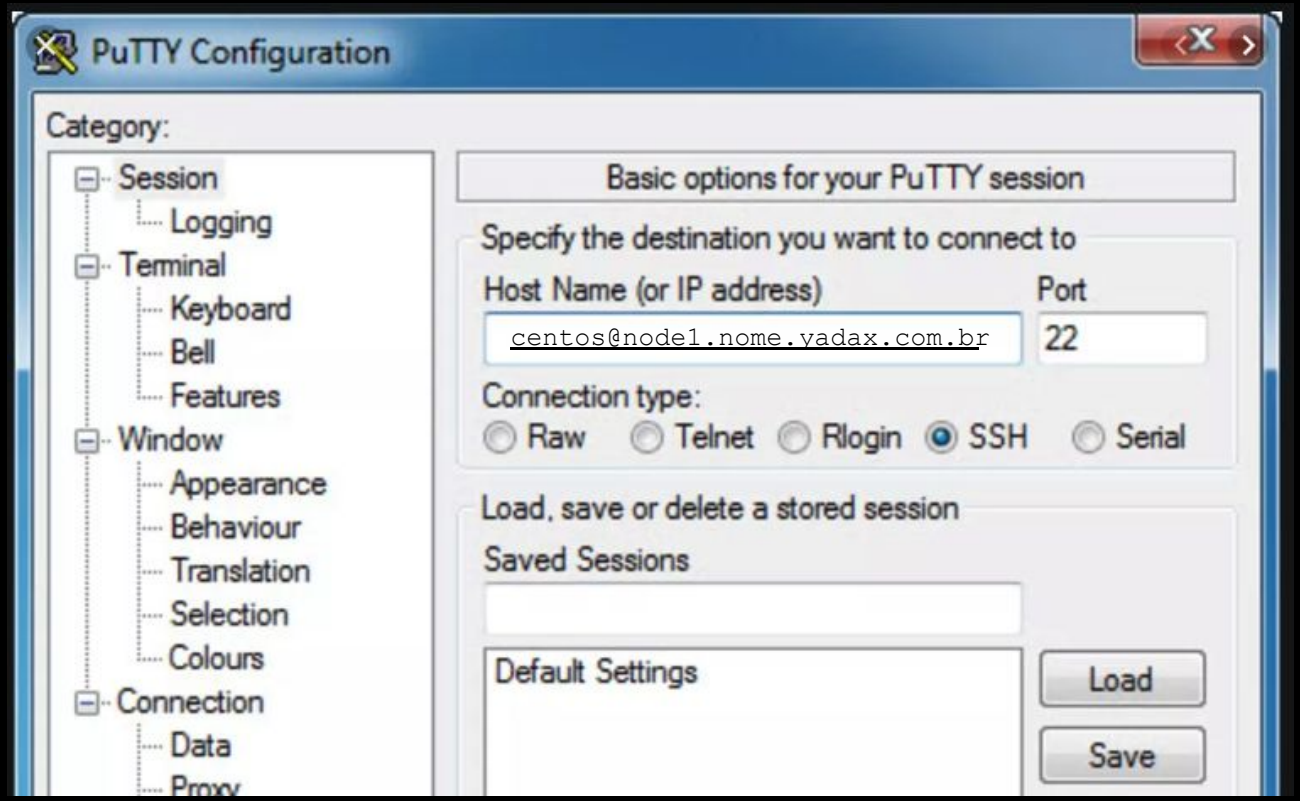
Sistema operacional

- Usaremos Centos7: Redhat like
  - Pacotes rpm
- Também são suportados:
  - Debian
  - Suse
  - Ubuntu
  - MacOS
  - Oracle Linux
  - Redhat
  - Amazon Linux

- $ -> Exec com o user cassandra
- # -> Exec com o root

Conectando

- `ssh centos@node1.dev.yadax.com.br`
- Utilize a senha do centos



yadax

Atualize seu SO

- `[centos@host ~]$ sudo su -`
- `[root@host ~]# yum update -y`
- 5 a 10 min

yadax

Python

O principal client do Cassandra (cqlsh) é desenvolvido em python.

```
# python -V
  Python 2.7.5
```

yadax

Java

Cassandra roda sobre uma JVM.

```
# java -version
  -bash: java: command not found
# yum install -y java-1.8.0-openjdk
# java -version
  openjdk version "1.8.0_222"
  OpenJDK Runtime Environment (build 1.8.0_222-b10)
  OpenJDK 64-Bit Server VM (build 25.222-b10, mixed mode)
```

yadax

Repositório Apache Cassandra

http://cassandra.apache.org/download/

- Com "vi", crie o arquivo:

```
# vi /etc/yum.repos.d/cassandra.repo
   [cassandra]
   name=Apache Cassandra
   baseurl=https://www.apache.org/dist/cassandra/redhat/22x/
   gpgcheck=1
   repo_gpgcheck=1
   gpgkey=https://www.apache.org/dist/cassandra/KEYS

   i -> começa escrever
   Esc + :wq -> Modo comandos + write + quit
```

yadax

Instalando o cassandra

```
# yum install -y cassandra
...
...

# rpm -qa | grep cassandra
  cassandra-2.2.19-1.noarch
```

yadax

Iniciando cassandra

```
# systemctl daemon-reload
# service cassandra start
```

ou

```
# systemctl start cassandra (Linux 7)
```

Devolve o prompt, mas leva um tempo +- 30 s

yadax

Iniciando cassandra

https://issues.apache.org/jira/browse/CASSANDRA-15273

https://yadax.com.br/cassandra/erro-ao-startar-apache-cassandra-2-2-16/

Problema devido à um patch de segurança. Corrigido no 2.2.19

```
[root@ip-172-31-89-81 ~]# service cassandra start
Reloading systemd:                                         [  OK  ]
Starting cassandra (via systemctl):  Job for cassandra.service failed because a
configured resource limit was exceeded. See "systemctl status cassandra.service" and
"journalctl -xe" for details.
                                                           [FAILED]
```

yadax

Acompanhe o log

```
tail -10f /var/log/cassandra/system.log
```

```
...
INFO  [main] 2021-06-30 18:59:59,948 Server.java:192 -
Starting listening for CQL clients on
localhost/127.0.0.1:9042...
```

tail -f -> vai jogando na tela tudo que ocorre no arquivo

CTRL + C devolve o prompt

Consultando processo java

```
# ps -ef | grep java
cassandra 2223 1 /usr/java/jdk1.8.0_144/jre/bin/java
-Ddse.server_process -ea -XX:+UseThreadPriorities -
... -XX:+UseTLAB -XX:+ResizeTLAB -XX:+UseNUMA
-Djava.net.preferIPv4Stack=true -Xms8G -Xmx8G ... ...
...
```

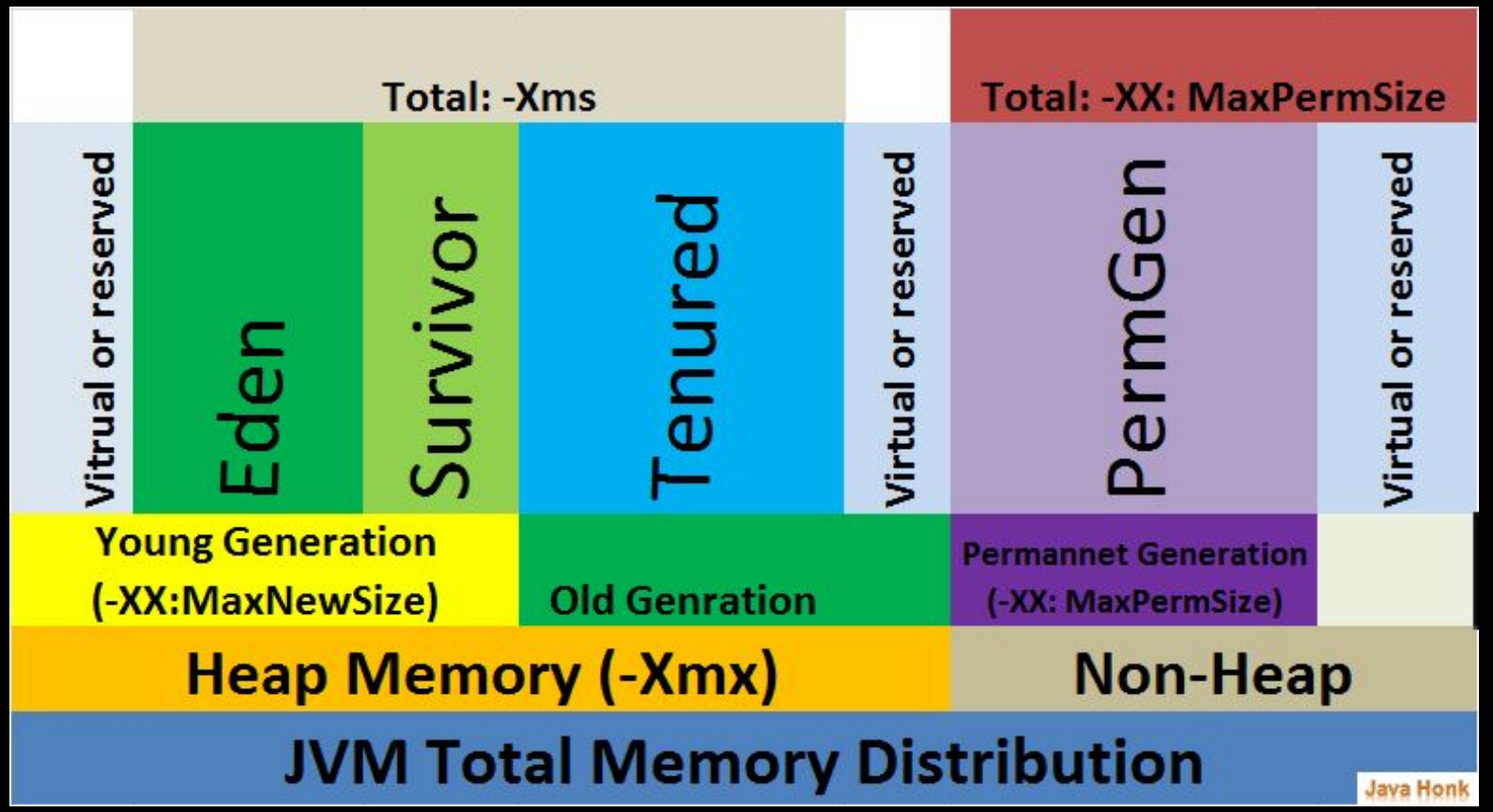
Xms e Xmx — são iguais e representam o tamanho da Heap

2233 — id do processo linux

yadax

Java Memory

- Xmx (max size) = Xms (size)

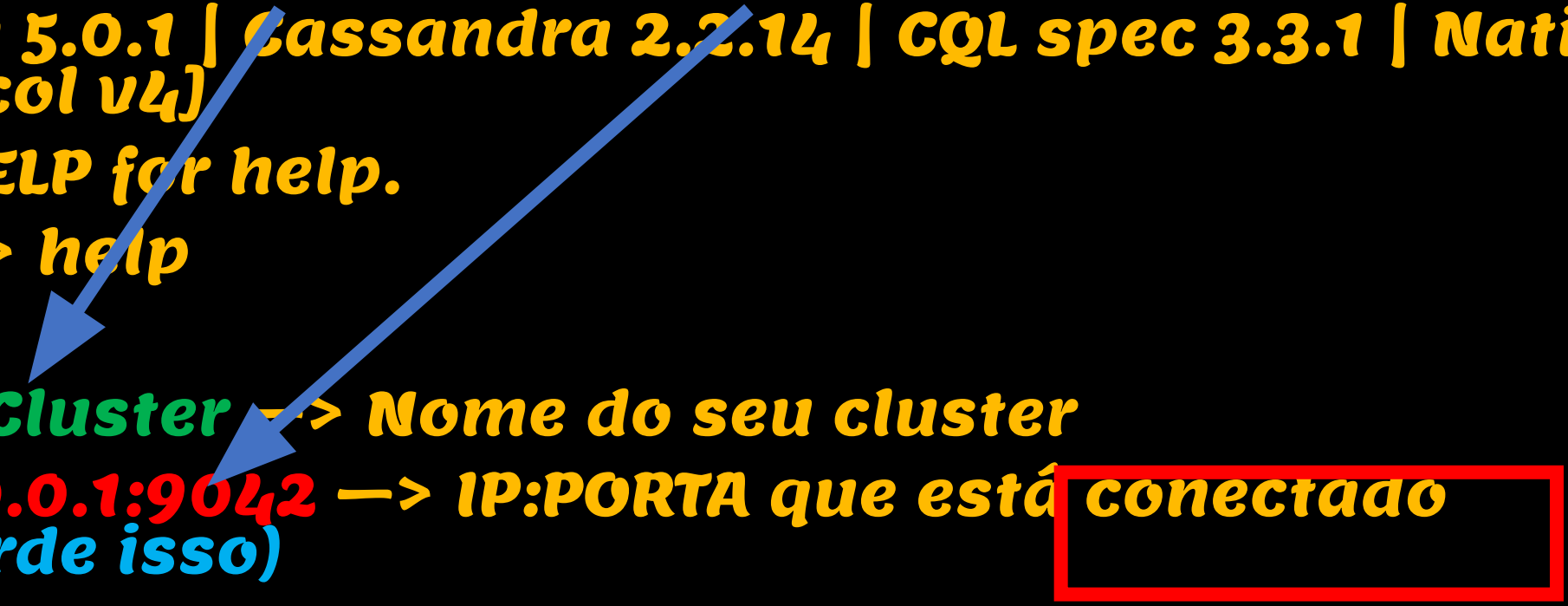


yadax

Conectando no cqlsh

```
# cqlsh
Connected to Test Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 2.2.14 | CQL spec 3.3.1 | Native protocol v4]
Use HELP for help.
cqlsh> help
```

- Test Cluster —> Nome do seu cluster
- 127.0.0.1:9042 —> IP:PORTA que está conectado (guarde isso)



yadax

Conectando no cqlsh

 # netstat –nltp

Mostra quais interface/portas o host está "ouvindo"

Guarde o resultado.

 # telnet 127.0.0.1 4321

O que acontece quando tentamos acessar uma po

```
[root@ip-10-191-18-63 ~]# telnet 127.0.0.1 4321
Trying 127.0.0.1...
telnet: connect to address 127.0.0.1: Connection refused
```

yadax

nodetool

- O nodetool é uma ferramenta que interage com o Cassandra via JMX (falaremos disso mais tarde)

- nodetool status

yadax

Parando cassandra

```
# service cassandra stop
```

ou

```
# systemctl stop cassandra
```

Nas versões antigas, nem sempre funciona. Busque pelo processo java para confirmar. Mate-o se necessário: kill -15 2223

. . . ou -9 se -15 não resolver

yadax

Parando cassandra

Embora o stop service pare o cassandra, há uma forma mais sutil de pará-lo

service cassandra start

$ nodetool drain – força descarregar todos os dados em memória (flush memtables) para disco (SSTables) e fecha todas as conexões abertas pelo cliente

service cassandra stop

yadax

Acompanhe o log

```
tail -10f /var/log/cassandra/system.log
```

```
...
INFO  [MemtableFlushWriter:3] 2021-06-30 19:18:57,442
Memtable.java:353 - Writing
Memtable-schema_columns@67241001(49.749KiB serialized
bytes, 967 ops, 0%/0% of on/off-heap limit)
INFO  [MemtablePostFlush:2] 2021-06-30 19:18:57,452
CompactionManager.java:1549 - Executor has been shut down,
not submitting background task
INFO  [MemtablePostFlush:2] 2021-06-30 19:18:57,453
CompactionManager.java:1549 - Executor has been shut down,
not submitting background task
INFO  [RMI TCP Connection(4)-127.0.0.1] 2021-06-30
19:18:57,459 StorageService.java:1210 - DRAINED
```

Habilitando cassandra no boot

```
# chkconfig cassandra on
```

ou

```
# systemctl enable cassandra
```

yadax

Logs do Cassandra

 # tail –f /var/log/cassandra/system.log

- ***Tem as principais informações sobre a saúde do Cassandra***
- ***Informações resumidas sobre GC (Garbage Collector)***

yadax

Outros Logs do Cassandra

```
# tail -f /var/log/cassandra/debug.log
```

Um pouco mais de informação - DEBUG

```
# tail -f /var/log/cassandra/gc.log.0.current
```

Logs da atividade de Gargabe Collector (Java)
GC é um dos maiores vilões para o Cassandra

Stop the world — O Java congela esperando pelo GC

yadax

Conhecendo o cassandra.yaml

- Yaml - yet another markup language
- Yadax - yet another database company (yadac), com x fica na moda
- Espaços controlam hierarquia
- Não deve iniciar por espaco

```
pai
  filho
    neto
     bisneto errado, com 1 espaco
```

yadax

Conhecendo o cassandra.yaml

- /etc/cassandra/conf/cassandra.yaml (default)
- Sempre traz uma explicação resumida do que o parâmetro representa
- Mantenha o Cassandra DOWN
- # service cassandra stop

```
# The name of the cluster. This is mainly used to prevent machines in
# one logical cluster from joining another.

cluster_name: 'Yadax'     <<< Faça esta alteração
```

yadax

Conhecendo o cassandra.yaml

```
# This defines the number of tokens randomly assigned to this node on the ring
# The more tokens, relative to other nodes, the larger the proportion of data
# that this node will store. You probably want all nodes to have the same number
# of tokens assuming they have equal hardware capability.

num_tokens: 16      <<< Faça esta alteração
```

yadax

Conhecendo o cassandra.yaml

Hinted Handoff: Se um node ficar down, outros nodes guardam dados para entregar quando o node voltar.

```
hinted_handoff_enabled: true
max_hint_window_in_ms: 10800000 # 3 hours
```

yadax

Conhecendo o cassandra.yaml

Por default, não é habilitado autenticação/autorização de usuário.

Todo mundo pode se conectar

Todo mundo pode fazer tudo.

```
authenticator: AllowAllAuthenticator
authorizer: AllowAllAuthorizer
```

yadax

Conhecendo o cassandra.yaml

Diretório onde ficarão nossos dados

```
data_file_directories:
  - /var/lib/cassandra/data
```

Estrutura de logs do cassandra para durabilidade de uma escrita.

```
commitlog_directory: /var/lib/cassandra/commitlog
commitlog_segment_size_in_mb: 32
```

yadax

Conhecendo o cassandra.yaml

É possível fazer ajustes nas estruturas de cache, chaves e/ou linhas inteiras

```
key_cache_size_in_mb: 100MB
row_cache_size_in_mb: 0
saved_caches_directory: /var/lib/cassandra/saved_caches
```

yadax

Conhecendo o cassandra.yaml

- Os seeds são nodes especiais, eles controlam quem entra e quem sai do cluster. Sempre será necessário ter um seed vivo.
- Em um novo cluster, sempre o primeiro node a startar precisa ser um seed.
- Mantenha pelo menos 2 seeds por datacenter e igual em todos os nodes

```
seed_provider:
- class_name: org.apache.cassandra.locator.SimpleSeedProvider
      parameters:
          - seeds: "172.31..." <<< Faça esta alteração
```

yadax

Conhecendo o cassandra.yaml

Você precisa informar ao cassandra como ele vai receber as requisições de seus "clientes". Pode ser via IP ou network interface.

```
# listen_address:  <<< Faça esta alteração
listen_interface: eth0 <<< Faça esta alteração
# rpc_address: <<< Faça esta alteração
rpc_interface: eth0      <<< Faça esta alteração
```

Para saber o nome da interface, execute:

```
$ ip a
```

yadax

Conhecendo o cassandra.yaml

- O cassandra precisa conhecer todos os membros do cluster.
- O parametro endpoint_snitch controla isso.
- Antigamente usava-se um arquivo com a lista de todos os membros x DCs. Desvantagem?
- Atualmente (com o GossipingPropertyFile -> cassandra-rackdc.properties) cada node informa a qual DC pertence.

```
endpoint_snitch: GossipingPropertyFileSnitch   << Faça esta alteração
```

yadax

Snitch

- SimpleSnitch – Single DC
- GossipingPropertyFileSnitch – For production
- PropertyFileSnitch - cassandra-topology.properties (todos os nodes)
- Ec2Snitch – Usa API EC2 para entender a topologia. Cada AZ é um RACK
- Ec2MultiRegionSnitch – O mesmo, cada REGIAO é um DC
- RackInferringSnitch – Baseia no range de IP para determinar o DC

yadax

Outros arquivos de conf

- cassandra-env.sh
  - Altera algumas configs de JVM nas versão mais antigas
  - Habilita JMX local/remoto
  - Autenticação do JMX
- logback.xml
  - Altera configs do log, como o tamanho do system.log
- jvm.options (Cassandra 3+)
  - Configurações específicas relacionadas a JVM nas versões mais novas
- cassandra-rackdc.properties
  - Usado pelo GossipingPropertyFileSnitch para informar Rack/DC

yadax

cassandra-rackdc.properties

- Aqui você determina para cada host em qual DC e Rack ele está
- O resto fica com a Gossip Protocol.
  - “Estou entrando no Cluster, estou no DCA e no RACK1”.

- No seu arquivo deixe:

```
dc=AWS      << Faça esta alteração
rack=Rack1  << Faça esta alteração
```

RackAwareness: Divide o dado entre Racks diferentes

yadax

Limpando nosso Server

```
# service cassandra stop
# ps -ef | grep java

# rm -rf /var/lib/cassandra/data/*
# rm -rf /var/lib/cassandra/commitlog/*
# rm -rf /var/lib/cassandra/saved_caches/*

ou
# rm -rf /var/lib/cassandra/*/*
```

yadax

Limpando nosso Server

Após limpar os arquivos, o cassandra sobe novinho. Lembre-se de iniciar por um SEED.

Em um cluster você terá que limpar todos os nodes antes de reiniciá-lo.

Inicie o cassandra e acompanhe o log

```
# service cassandra start
```

```
# tail -f /var/log/cassandra/system.log
```

yadax

Inicie o cassandra e acompanhe o log

```
INFO  [main] 2021-06-30 19:53:24,589
Server.java:192 - Starting listening for CQL
clients on /10.191.18.63:9042...
INFO  [main] 2021-06-30 19:53:24,615
CassandraDaemon.java:540 - ...
INFO  [main] 2021-06-30 19:53:24,615
CassandraDaemon.java:633 - Startup complete
INFO  [OptionalTasks:1] 2021-06-30 19:53:26,643
CassandraRoleManager.java:378 - Created default
superuser role 'cassandra'
```

yadax

Conecte no cassandra

```
$ cqlsh


Connection error: ('Unable to connect to any
servers', {'127.0.0.1': error(111, "Tried
connecting to [('127.0.0.1', 9042)]. Last error:
Connection refused")})
```

O que houve? Estava funcionando!

```
[root@ip-10-191-18-63 ~]# telnet 127.0.0.1 4321
Trying 127.0.0.1...
telnet: connect to address 127.0.0.1: Connection refused
```

yadax

Busque por ajuda!

```
$ cqlsh --help
Usage: cqlsh.py [options] [host [port]]

CQL Shell for Apache Cassandra
```

O default é localhost (127.0.0.1)

Mas o cassandra ouve(listen) na loopback?

yadax

Ouvindo na 9042

```
$ netstat -nltp | grep 9042
tcp    0    0 10.191.18.63:9042      0.0.0.0:*    LISTEN      15249/java
```

Connection Refused: Você tentou acessar por uma porta que o host/interface não está ouvindo

```
[root@ip-10-191-18-63 ~]# netstat -nltp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:9042          0.0.0.0:*               LISTEN      84143/java
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      1763/sshd
```

ANTES

```
[root@ip-10-191-18-63 ~]# telnet 127.0.0.1 4321
Trying 127.0.0.1...
telnet: connect to address 127.0.0.1: Connection refused
```

yadax

Conecte no cassandra

```
# cqlsh $HOSTNAME
Connected to Yadax at ip-172-31-93-254.ec2.internal:9042.
[cqlsh 5.0.1 | Cassandra 2.2.16 | CQL spec 3.3.1 | Native protocol v4]
Use HELP for help.
cqlsh>
```

yadax

Dados: CQL e cqlsh

yadax

Conecte no cassandra, crie sua 1ª keyspace

```
$ cqlsh
cassandra@cqlsh>  describe keyspaces;
   system  system_auth  ...

cassandra@cqlsh> CREATE KEYSPACE hello_world WITH
REPLICATION = { 'class' : 'SimpleStrategy',
'replication_factor' : 1 } AND DURABLE_WRITES = true;
```

Keyspace:

- Schema: no oracle
- Database: no mysql, mssql

yadax

Create Keyspace

Sintaxe:

```
CREATE KEYSPACE [IF NOT EXISTS] <keyspace_name>
 WITH replication =
 {'class': '<replication_strategy>',
  '<data_center_name>':'<replication_factor>'}
 AND durable_writes = <true/false>;
```

yadax

Create Keyspace – Single DC

```
CREATE KEYSPACE IF NOT EXISTS yadax
 WITH replication =
 {'class': 'NetworkTopologyStrategy', 'AWS':'1'}
 AND durable_writes = true;
```

Crie esta!

yadax

Create Keyspace – Multi DC

```
CREATE KEYSPACE yadax
 WITH replication = {'class': 'NetworkTopologyStrategy',
  'AWS':'2', 'Azure':'3', 'GCP':'3'}
 AND durable_writes = true;
```

Se multi DC / Cloud

yadax

Replication Strategy

- SimpleStrategy
  - Possível definir apenas o **número de réplicas.**
- NetworkTopologyStrategy
  - Possível definir **número de réplicas por datacenter.** Utilize mesmo que tenha um DC só.

REPLICATION FACTOR

yadax

Durable Writes

- True: Escreve no commit log
- False: Bypass no commit log
  - Não recomendado

yadax

Crie sua primeira tabela (column family)

```
cassandra@cqlsh> use yadax;

cassandra@cqlsh:yadax> create table my_table (id
int, nome text);

InvalidRequest: Error from server: code=2200
[Invalid query] message="No PRIMARY KEY specifed
(exactly one required)"
```

yadax

Crie sua primeira tabela (column family)

```
cassandra@cqlsh:yadax> create table users (
id int,
nome text,
primary key (id));

insert into users (id, nome)values (1,
'Adriano');

commit; ??
```

yadax

DDL da sua keyspace

```
cassandra@cqlsh> describe yadax;

CREATE KEYSPACE yadax WITH replication = {'class':
'NetworkTopologyStrategy', 'AWS': '1'}  AND
durable_writes = true;

CREATE TABLE yadax.users(
    id int PRIMARY KEY,
    nome text
)...
;
```

yadax

Dicionário de dados

- Também é possível buscar sua tabela no dicionário de dados.

```
SELECT columnfamily_name
FROM   system.schema_columnfamilies
WHERE  keyspace_name = 'yadax';

 columnfamily_name
-------------------
             users
```

yadax

Partition Key

- Determina qual node será responsável pelo dado
- Partitioner: Valor para o Cluster, não pode ser alterado
- Depende de um algoritmo de Hash - Partitioner
  - RandomPartitioner - Old
  - Murmur3Partitioner – Evolução do Random – DEFAULT (cassandra.yaml)
  - ByteOrderedPartitioner – Distribuído baseado texto

yadax

Partition Key

- Talvez o ponto mais importante do projeto
- Relacionado as maiores falhas de projetos
- Duas histórias tristes
  - Hot spot
  - Dados duplicados

yadax

Clustering Columns

Partition Key determina qual node é responsável pelo dado.

Clustering column determina a ordem que os registros serão ordenados

Útil para casos que deseja dados ordenados na consulta

```
CREATE TABLE comments_by_users
(id_user INT,
 dt_comment DATE,
 id_film TEXT,
 comment TEXT,
 PRIMARY KEY (id_user, dt_comment));
```

yadax

Clustering Columns

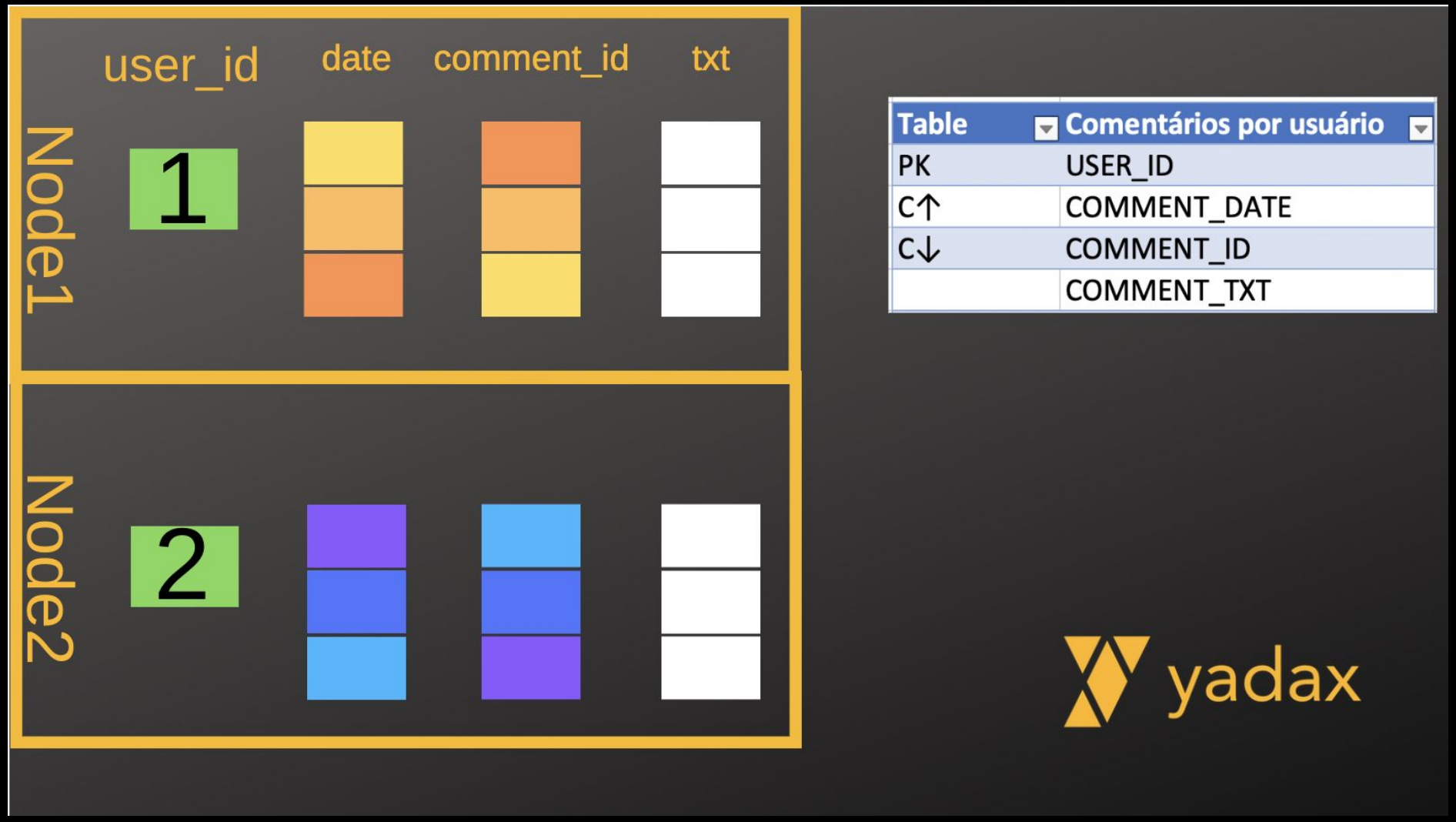


Table	Comentários por usuário
PK	USER_ID
C↑	COMMENT_DATE
C↓	COMMENT_ID
	COMMENT_TXT

yadax

Clustering Columns

```
cassandra@cqlsh:yadax> describe comments_by_users

CREATE TABLE yadax.comments_by_users (
    id_user int,
    dt_comment date,
    comment text,
    id_film text,
    PRIMARY KEY (id_user, dt_comment)
) WITH CLUSTERING ORDER BY (dt_comment ASC)
```

yadax

Clustering Columns

```
PRIMARY KEY (id_user, dt_comment)
```

Vamos inserir 4 registros, o último duplicado.

yadax

Clustering Columns

```
> insert into yadax.comments_by_users (id_user, dt_comment,
id_film, comment) values (1, '2019-02-02', 'A','Foi legal');

> insert into yadax.comments_by_users (id_user, dt_comment,
id_film, comment) values (1, '2019-01-01', 'B','Foi legal também');

> insert into yadax.comments_by_users (id_user, dt_comment,
id_film, comment) values (1, '2019-03-03', 'C','Foi bom');

> insert into yadax.comments_by_users (id_user, dt_comment,
id_film, comment) values (1, '2019-03-03', 'C','Muito bom');
```

yadax

Clustering Columns

```
> SELECT *
  FROM   comments_by_users
  WHERE  id_user = 1;

 id_user | dt_comment | comment                | id_film
---------+------------+------------------------+---------
       1 | 2019-01-01 | Foi legal também       |       B
       1 | 2019-02-02 |           Foi legal    |       A
       1 | 2019-03-03 |           Muito bom    |       C
```

O que houve?

Primary Key =
Partition Key + Clustering Columns

yadax

Allow Filtering (caca)

```
> select * FROM hello_world.comments_by_users where
dt_comment = '2019-02-02';
```

```
InvalidRequest: Error from server: code=2200 [Invalid
query] message="Cannot execute this query as it might
involve data filtering and thus may have unpredictable
performance. If you want to execute this query despite
the performance unpredictability, use ALLOW FILTERING"
```

yadax

Allow Filtering

```
> select * FROM hello_world.comments_by_users where
dt_comment = '2019-02-02' ALLOW FILTERING;

 id_user | dt_comment | comment    | id_film
---------+------------+------------+---------
       1 | 2019-02-02 | Foi legal |       A
```

yadax

REPITA COMIGO:
BUSQUE SEMPRE PELA PK!

Se você precisa buscar de forma diferente, você não deveria estar usando Cassandra.

yadax

Partition Key Composta

```
CREATE TABLE logs_by_location (
location_id TEXT,
day INT,
time_in TIMESTAMP,
log TEXT,
PRIMARY KEY ((location_id, day), time_in));
```

Precisa separar por vírgulas dentro dos ()

yadax

Clustering Order

```
CREATE TABLE logs_by_location (
location_id TEXT,
day INT,
time_in TIMESTAMP,
log TEXT,
PRIMARY KEY ((location_id, day), time_in))
WITH CLUSTERING ORDER BY (time_in DESC);
```

yadax

UPSERT (bruxaria)

```
SELECT *
FROM   yadax.logs_by_location;
```

```
 location_id | day | time_in                        | log
-------------+-----+--------------------------------+---------
             |     |                                |
```

yadax

UPSERT (bruxaria)

```
UPDATE  yadax.logs_by_location
SET     log = 'Tudo OK'
WHERE   location_id = 'Host 1'
AND     day = 1
AND     time_in = '2019-01-01 04:00';
```

yadax

UPSERT

- O que acontece agora que você fez um update em um registro que não existe?

- O que você espera?

yadax

UPSERT

```
SELECT *
FROM   yadax.logs_by_location;
```

```
 location_id | day | time_in                       | log
-------------+-----+-------------------------------+---------
      Host 1 |   1 | 2019-01-01 04:00:00+0000 | Tudo OK
```

yadax

Secondary Index

```
SELECT *
FROM   yadax.logs_by_location
WHERE  location_id = 'Host 1'
AND    day = 1;
```

Desta forma (location_id AND day), OK!

yadax

Secondary Index

```
SELECT *
FROM   yadax.logs_by_location
WHERE  day = 1;
```

```
InvalidRequest: Error from server: code=2200 [Invalid query]
message="Partition key parts: location_id must be restricted
as other parts are"
```

yadax

Secondary Index

```
> create index logs_day on yadax.logs_by_location(day);
```

```
> cassandra@cqlsh:yadax> select * from yadax.logs_by_location
where day = 1;
```

```
 location_id | day | time_in                         | log
-------------+-----+---------------------------------+---------
      Host 1 |   1 | 2019-01-01 04:00:00+0000 | Tudo OK
```

yadax

Secondary Index

- Devem ser bem planejados
- Evite baixa cardinalidade (poucos valores distintos)
- Evite alta cardinalidade (prefira Mview, mas não use Mview ☺)
- Não utilize em colunas que sofrem updates

yadax

Query Driven

- O Cassandra não é flexível
- O dado não será normalizado
- Uma tabela para cada consulta
- Pense primeiro na query, depois na tabela
- Sua app precisa popular todas as tabelas

yadax

Materialized View – 3+

- Veio para diminuir o problema de ser query driven
- Cria outra “tabela” com uma PK diferente, mantida pelo próprio Cassandra.
- Você joga para o banco uma responsabilidade da app.

yadax

Datatypes

Type	Supported	Description
ascii, text	strings	US-ASCII character string, UTF-8 encoded string
bigint, int	integers	64/32-bit signed long
blob	blobs	Arbitrary bytes (no validation), expressed as hexadecimal
boolean	booleans	true or false
counter	integers	Distributed counter value (64-bit long)
Date, time, timestamp	strings	yyyy-mm-dd, HH:MM:SS[.fff], yyyy-mm-dd[(T|)HH:MM:SS[.fff]][(+|-)NNNN]
Decimal, double, float	integers, floats	Java type
list		[literal, literal, literal]
map		{ key : value, key : value ... }
set		{ literal, literal, literal }

CONSISTENCY

- Determina o número mínimo de hosts envolvidos em uma operação de escrita/leitura

- Quorum = TRUNC((RF / 2)) + 1

- Principais
  - ONE
  - TWO
  - LOCAL_QUORUM
  - EACH_QUORUM
  - QUORUM
  - ALL

yadax

Arquitetura Cassandra

yadax

Ring ou Cluster

- Single instance -> Node
- Grupo de Nodes -> Cluster / Ring

- Cada Node é responsável por um range de Tokens

- Cada PartitionKey é hasheada para um token específico

Node – RF 3	Range Prim	Range Sec	Range Ter
1	-100 a -67	1 a 33	67 a 100
2	-33 a 0	-100 a -67	1 a 33
3	34 a 66	-33 a 0	-100 a -67
4	-66 a -34	34 a 66	-33 a 0
5	67 a 100	-66 a -34	34 a 66
6	1 a 33	67 a 100	-66 a -34

Token vs Ring

- Primary key: (nome, idade)
- Nome = '**AdrianO**'-> Token: 84
- Nome = 'Raul'-> Token: -49
- Nome = '**AdrianA**'-> Token: -3

Node – RF 3	Range Prim	Range Sec	Range Ter
1	-100 a -67	1 a 33	67 a 100
2	-33 a 0	-100 a -67	1 a 33
3	34 a 66	-33 a 0	-100 a -67
4	-66 a -34	34 a 66	-33 a 0
5	67 a 100	-66 a -34	34 a 66
6	1 a 33	67 a 100	-66 a -34

- Esta forma é conhecida como: single token range per node (antigo)

yadax

VNodes

- Com os vnodes, temos varios ranges por host
- Initial_tokens controlam a quantidade de ranges
- Por isso a quantidade de dados é proporcional ao # ranges

Node – RF 3	Range Prim	Range Sec	Range Ter
1	-100 a -67	1 a 33	67 a 100
2	-33 a 0	-100 a -67	1 a 33
3	34 a 66	-33 a 0	-100 a -67
4	-66 a -34	34 a 66	-33 a 0
5	67 a 100	-66 a -34	34 a 66
6	1 a 33	67 a 100	-66 a -34

Node – RF 3	Range Prim	Range Prim	Range Prim
1	-100 a -91	51 a 60	91 a 100
2	1 a 10	-90 a -81	-50 a -41
3	31 a 40	51 a 60	-40 a -31
4	-80 a -71	41 a 50	-70 a -61
5	-20 a -11	-60 a -51	-30 a -21
6	61 a 70	81 a 90	91 a 100
7		71 a 80	
8	-10 a 0	11 a 20	21 a 30

INITIAL_TOKENS

yadax

Partitioner

- É quem determina a forma de gerar os tokens
- Função Hash

- ByteOrderedPartitioner
- RandomPartitioner
- **Murmur3Partitioner (default)**

Estrutura de dados

- MemTable
  - Mem(ory) Table
  - Representação da tabela em memória
  - Toda escrita acontece na Memtable + commit log.
  - De tempos em tempos, a memtable é “descarregada” para disco

Memtable - Parâmetros

- memtable_flush_writers
- memtable_heap_space_in_mb
- memtable_cleanup_threshold
- memtable_flush_period_in_ms (per table)
- memtable_allocation_type

yadax

Memtable - Flush

- Commit log cheio
- Periodicamente, baseado no memtable_flush_period_in_ms
- Threshold atingido - memtable_cleanup_threshold
- Manualmente
  - nodetool flush

yadax

Estrutura de dados

- SSTable
  - Sorted String Table
- Conceito do Google BigTable
- Imutável
- Persistente, disco
- Criada com o Flush da Memtable
- Ordenada pelo Hash da PK

SSTables

```
# pwd
/var/lib/cassandra/data/yadax/scores-42b59f00dd7211e987a79d4050c7a079
# ll
-rw-r--r--. 1 cassandra cassandra   43 Sep 22 20:53 lb-1-big-CompressionInfo.db
-rw-r--r--. 1 cassandra cassandra  692 Sep 22 20:53 lb-1-big-Data.db
-rw-r--r--. 1 cassandra cassandra   10 Sep 22 20:53 lb-1-big-Digest.adler32
-rw-r--r--. 1 cassandra cassandra   24 Sep 22 20:53 lb-1-big-Filter.db
-rw-r--r--. 1 cassandra cassandra  127 Sep 22 20:53 lb-1-big-Index.db
-rw-r--r--. 1 cassandra cassandra 4516 Sep 22 20:53 lb-1-big-Statistics.db
-rw-r--r--. 1 cassandra cassandra   93 Sep 22 20:53 lb-1-big-Summary.db
-rw-r--r--. 1 cassandra cassandra   94 Sep 22 20:53 lb-1-big-TOC.txt
```

SSTables

- Data - Data.db
  - Dados da SSTable
- Primary Index – Index.db
  - Índice das Partitions Keys apontando para sua posição no datafile
- Bloom Filter – Filter.db
  - Réplica de uma estrutura de memória (bloom filter) que verifica se a linha pode existir naquela Memtable
- Compression Info - CompressrionInfo.db
  - Informações a respeito de dados não comprimidos, localização e outras infos referentes a compressão.

SSTables

- Statistics - Statistics.db
  - Metadados referentes a estatísticas envolvendo dados da SStable
- Digest - Digest.adler32
  - Arquivo contendo Checksum do datafile
- SSTable Index Summary – Summary.db
  - Porção (sample) do partition index
- Table of Contents – TOC.txt
  - Contém a lista dos arquivos em formato texto

SSTable Index

yadax

Bloom Filter

- Probabilistic data Structure
- Pode haver falsos positivos: Possivelmente existe a PK
- Não há falsos negativos: Não existe esta PK

yadax

CommitLog

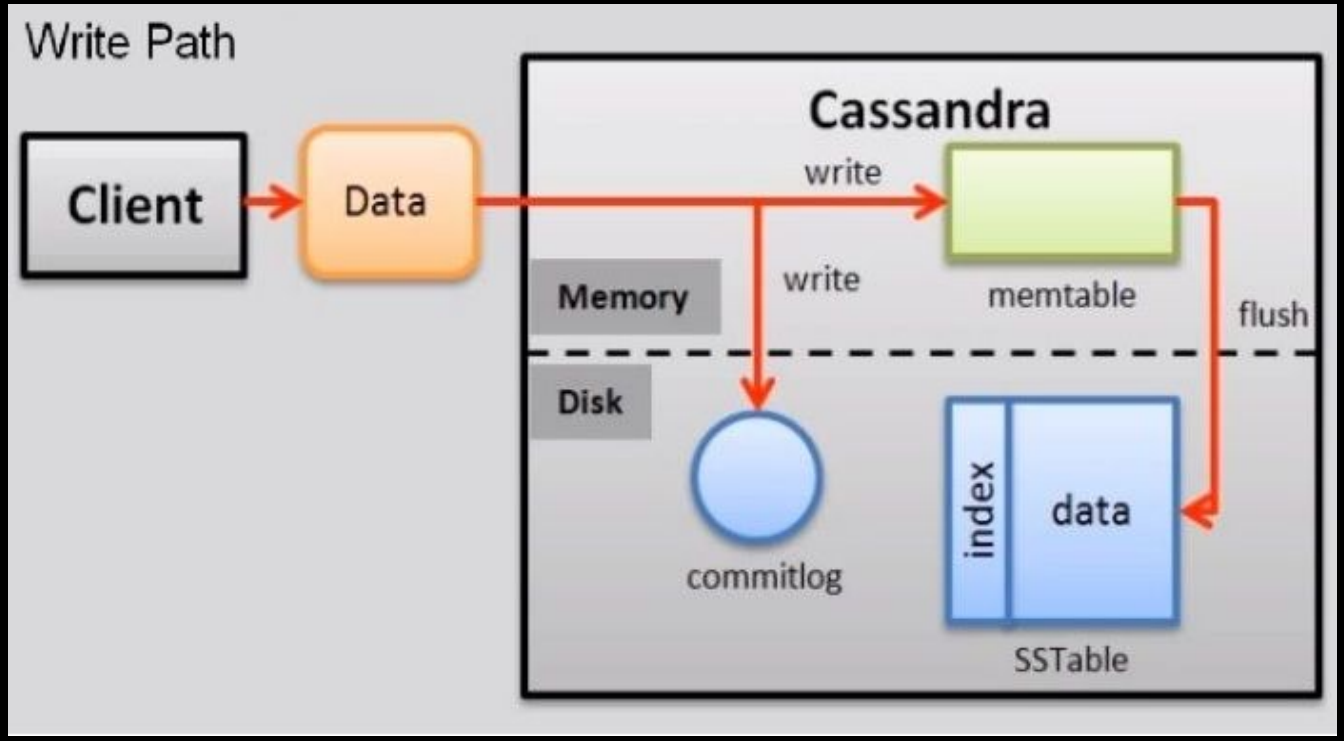
- Redo Log – Oracle
- Binany Log – Mysql
- Transaction Log – MSSql
- WAL - Postgres

- Garante a durabilidade da transação
- Utilizado caso o Host Falhe (crash).
- Escrito sequencialmente

yadax

CommitLog

- commitlog_directory
- commit_failure_policy
- commitlog_segment_size_in_mb
- commitlog_total_space_in_mb



yadax

Hinted Handoff

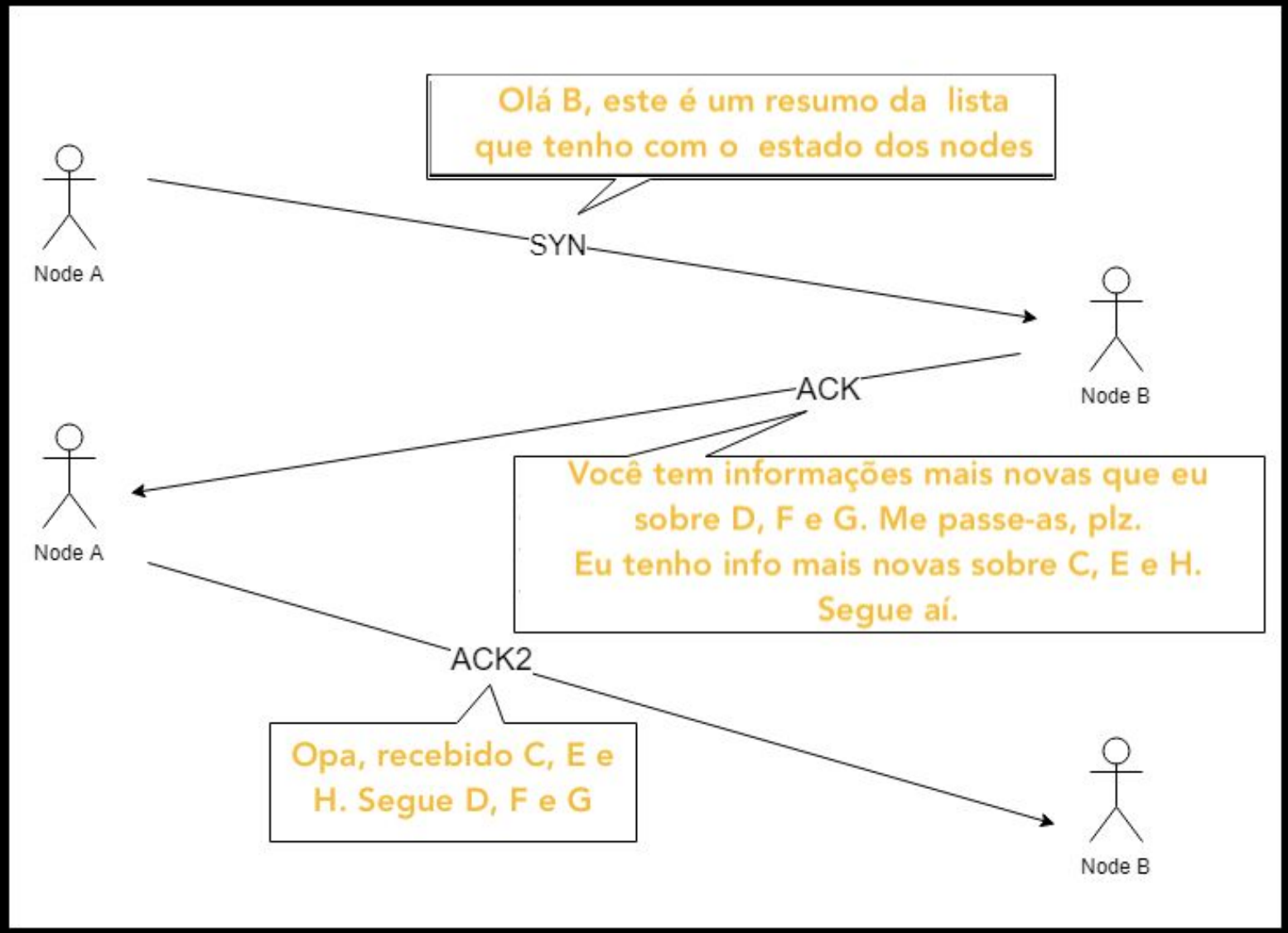
- O Cassandra é altamente disponível
- Isso leva pessoas escolherem ele, mesmo não se adaptando ao cases de uso
- Uma das peças chaves são as hinted handoff
- Quando um node falha, o outro assume suas obrigações
- Se o node 2 falha, o node 1 pode atuar como coordinator
- Ele guarda os registros que deveriam estar com o 2
- Entrega quando o node 2 voltar.

Hinted Handoff

- Quando o node volta, as Hints são entregues
- Se o node fica fora por mais de max_hint_window_in_ms, será necessário um repair para colocar a casa em ordem

- hinted_handoff_enabled
- max_hint_window_in_ms
- hinted_handoff_throttle_in_kb
- max_hints_delivery_threads

Gossip Protocol



yadax

Gossip Protocol

- Normalmente um node conversa com até três nodes a cada segundo
- Não há restrição/determinação sobre com quem conversar
- Não armazenam com quem falou anteriormente
- Modo eficiente de espalhar a topologia e saúde do cluster
- Espalha topologia – Rack e DC
- Espalha donos de tokens
- Status do colega: Normal, Joining, Down, Leaving
- Load do colega: CPU, Disco, ...

CLUSTER

yadax

Mãos a obra?

- Vamos criar nosso cluster.

yadax

Configuração Cluster - Overview

- O melhor do Cassandra é sua alta disponibilidade
- Multi Master
  - Isso é para poucos
- Multi Datacenters
- Sem ponto único de falha
- SEEDs controlam quem entra e quem sai
  - Use 2 ou 3 por DC
- Todos precisam ter o mesmo CLUSTER_NAME

Configuração Cluster - Overview

- Vamos usar o GossipingPropertyFileSnitch como snitch
- É preciso configurar o cassandra-rackdc.properties
- Os nodes precisam se falar na porta 7000 e 7001 (listen_address)
- A distribuição de dados entre nodes será proporcional ao num_tokens

yadax

Configuração Cluster – Mão a obra

- Acesse suas 6 maquinas para certificar que está tudo OK.
- Crie o repositório yum em todos os nodes
- Instale o cassandra 2.2 em todos os nodes
- Neste casos, ferramentas para executar a mesma ação em diversos nodes ajuda:
  - Win: Moba
  - Mac: iTerm
  - Linux: Terminator
- Pense como seria bom isso tudo isso ser automático

yadax

Configuração Cluster – Mão a obra

- Escolha 3 para o DCA e 3 para o DCB
- Escolha 2 do DCA e 2 do DCB para serem SEEDs

- Para o DCA, o cassandra-rackdc.properties deve ficar:

  dc=DCA

  rack=RACK1
- Faça o mesmo para o DCB

yadax

Configuração Cluster – Mão a obra

- Escolha o nome do seu cluster
- Mude o endpoint_snitch para GossipingPropertyFileSnitch
- Veja o IP dos hosts
- Limpe os dados dos nodes, caso já tenha sido utilizado

yadax

ARQUIVOS DE CONF

yadax

Configuração Cluster – Mão a obra

- Estes são os parâmetros que você vai precisar alterar no cassandra.yaml:
  - cluster_name: Yadax
  - num_tokens: 16
  - seeds: ip1, ip2, ip4, ip5
  - # listen_address -> comentar (# no começo da linha)
  - listen_interface -> eth0 (utilizar “ip a” para descobrir)
  - # rpc_address: -> comentar (# no começo da linha)
  - rpc_interface: eth0
  - endpoint_snitch: GossipingPropertyFileSnitch

yadax

Configuração Cluster – Mão a obra

- Estes são os parâmetros que você vai precisar alterar no cassandra-rackdc.properties:

  dc=DCA

  rack=RACK1

yadax

START DO CLUSTER

Configuração Cluster – Mão a obra

- Start primeiro um dos seeds
- Acompanhe o log
- Verifique a saúde do cluster, os datacenters onde a maquina se encontra.
- nodetool status

yadax

Crie um alias para o system.log

```
# echo "alias alert='tail -10f /var/log/cassandra/system.log'" >> ~/.bash_profile

# . ~/.bash_profile

# alert
```

yadax

Crie uma keyspace

```
cassandra@cqlsh>
CREATE KEYSPACE yadax
WITH replication = {'class': 'NetworkTopologyStrategy',
'DCA': '1'}  AND durable_writes = true;
```

yadax

COPY

yadax

COPY

- Verifique no node1 se existem arquivos na pasta /tmp/2msales.csv

- `# head -2 /tmp/2msales.csv`

COPY

```
CREATE KEYSPACE killrvideo  WITH replication = {'class':
'NetworkTopologyStrategy', 'dc1':1};

CREATE TABLE killrvideo.videos_by_actor (
   actor_name text,
   character_name text,
   video_id timeuuid,
   release_year int,
   title text,
   PRIMARY KEY (actor_name, character_name, video_id)
) WITH CLUSTERING ORDER BY (character_name ASC, video_id ASC);
```

COPY

```
CREATE TABLE killrvideo.actors_by_video (
   video_id timeuuid,
   actor_name text,
   character_name text,
   release_year int,
   title text,
   PRIMARY KEY (video_id, actor_name, character_name)
) WITH CLUSTERING ORDER BY (actor_name ASC, character_name ASC);
```

yadax

COPY – FROM (import)

```
yadax@cqlsh:yadax> COPY killrvideo.videos_by_actor (actor_name,
character_name, video_id , release_year, title) from
'/tmp/videos_by_actor.csv';
```

```
yadax@cqlsh:yadax> COPY killrvideo.actors_by_video (actor_name,
character_name, video_id , release_year, title) from
'/tmp/actors_by_video.csv'
```

yadax

COPY

```
cqlsh:yadax> select count(*) from sales;

OperationTimedOut: errors={'10.191.18.67': 'Client request timeout. See
Session.execute[_async](timeout)'}, last_host=10.191.18.67
```

```
cqlsh:yadax> select * from sales limit 1;

 order_id  | country   | item_type  | order_priority | region | sales_channel | total_cost | total_profit | total_revenue | unit_cost | unit_price | units_sold
-----------+-----------+------------+----------------+--------+---------------+------------+--------------+---------------+-----------+------------+------------
 658789432 | Macedonia | Vegetables |              L | Europe |        Online | 1430755.22 |    586288.31 |    1430755.22 |     90.93 |     154.06 |       9287
```

yadax

ARQUIVOS DE DADOS

yadax

Arquivo de dados

- ls -lhrt (h -> human readable)
- Liste os arquivos da pasta /var/lib/cassandra/data
- Liste os arquivos da pasta /var/lib/cassandra/data/killrvideo
- Liste os arquivos da pasta /var/lib/cassandra/data/killrvideo/actors...

FLUSH

yadax

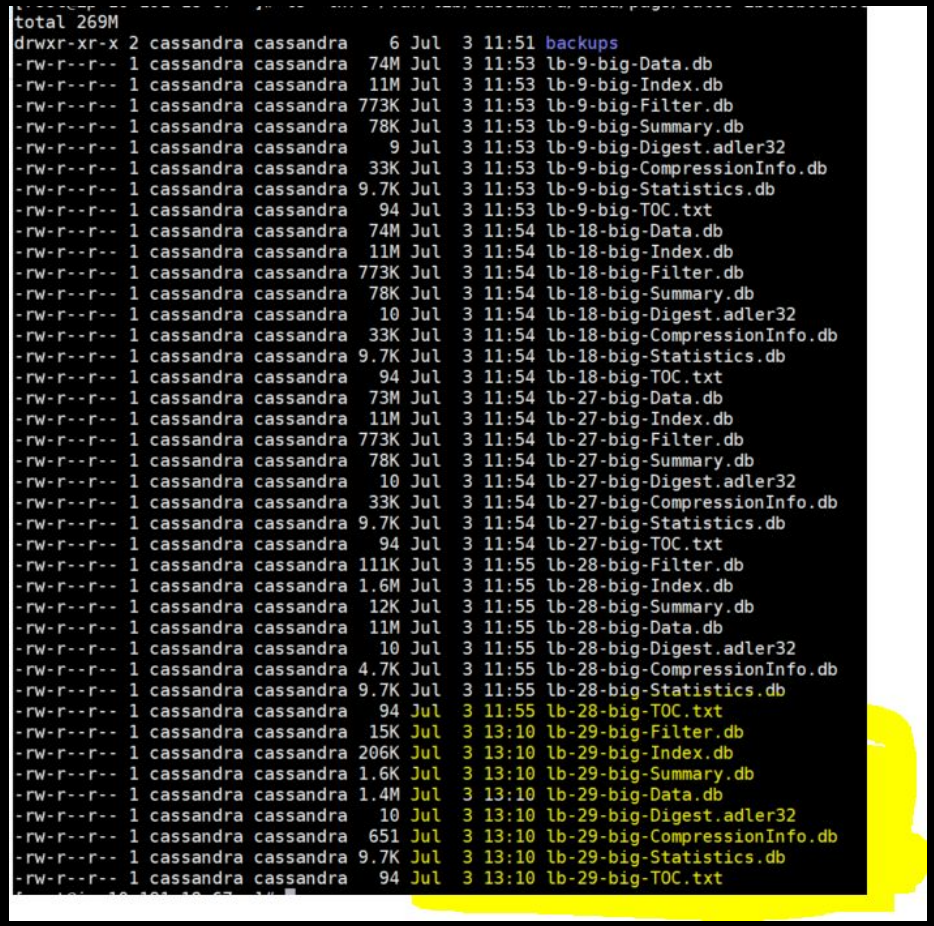
Force um flush

- nodetool flush killrvideo
- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/sales...
- O que houve?
- O que o nodetool flush faz mesmo?

yadax

FLUSH

```
total 267M
drwxr-xr-x 2 cassandra cassandra    6 Jul  3 11:51 backups
-rw-r--r-- 1 cassandra cassandra  74M Jul  3 11:53 lb-9-big-Data.db
-rw-r--r-- 1 cassandra cassandra  11M Jul  3 11:53 lb-9-big-Index.db
-rw-r--r-- 1 cassandra cassandra 773K Jul  3 11:53 lb-9-big-Filter.db
-rw-r--r-- 1 cassandra cassandra  78K Jul  3 11:53 lb-9-big-Summary.db
-rw-r--r-- 1 cassandra cassandra    9 Jul  3 11:53 lb-9-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra  33K Jul  3 11:53 lb-9-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 11:53 lb-9-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 11:53 lb-9-big-TOC.txt
-rw-r--r-- 1 cassandra cassandra  74M Jul  3 11:54 lb-18-big-Data.db
-rw-r--r-- 1 cassandra cassandra  11M Jul  3 11:54 lb-18-big-Index.db
-rw-r--r-- 1 cassandra cassandra 773K Jul  3 11:54 lb-18-big-Filter.db
-rw-r--r-- 1 cassandra cassandra  78K Jul  3 11:54 lb-18-big-Summary.db
-rw-r--r-- 1 cassandra cassandra   10 Jul  3 11:54 lb-18-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra  33K Jul  3 11:54 lb-18-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 11:54 lb-18-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 11:54 lb-18-big-TOC.txt
-rw-r--r-- 1 cassandra cassandra  73M Jul  3 11:54 lb-27-big-Data.db
-rw-r--r-- 1 cassandra cassandra  11M Jul  3 11:54 lb-27-big-Index.db
-rw-r--r-- 1 cassandra cassandra 773K Jul  3 11:54 lb-27-big-Filter.db
-rw-r--r-- 1 cassandra cassandra  78K Jul  3 11:54 lb-27-big-Summary.db
-rw-r--r-- 1 cassandra cassandra   10 Jul  3 11:54 lb-27-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra  33K Jul  3 11:54 lb-27-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 11:54 lb-27-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 11:54 lb-27-big-TOC.txt
-rw-r--r-- 1 cassandra cassandra 111K Jul  3 11:55 lb-28-big-Filter.db
-rw-r--r-- 1 cassandra cassandra 1.6M Jul  3 11:55 lb-28-big-Index.db
-rw-r--r-- 1 cassandra cassandra  12K Jul  3 11:55 lb-28-big-Summary.db
-rw-r--r-- 1 cassandra cassandra  11M Jul  3 11:55 lb-28-big-Data.db
-rw-r--r-- 1 cassandra cassandra   10 Jul  3 11:55 lb-28-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra 4.7K Jul  3 11:55 lb-28-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 11:55 lb-28-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 11:55 lb-28-big-TOC.txt
```



```
total 269M
drwxr-xr-x 2 cassandra cassandra    6 Jul  3 11:51 backups
-rw-r--r-- 1 cassandra cassandra  74M Jul  3 11:53 lb-9-big-Data.db
-rw-r--r-- 1 cassandra cassandra  11M Jul  3 11:53 lb-9-big-Index.db
-rw-r--r-- 1 cassandra cassandra 773K Jul  3 11:53 lb-9-big-Filter.db
-rw-r--r-- 1 cassandra cassandra  78K Jul  3 11:53 lb-9-big-Summary.db
-rw-r--r-- 1 cassandra cassandra    9 Jul  3 11:53 lb-9-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra  33K Jul  3 11:53 lb-9-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 11:53 lb-9-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 11:53 lb-9-big-TOC.txt
-rw-r--r-- 1 cassandra cassandra  74M Jul  3 11:54 lb-18-big-Data.db
-rw-r--r-- 1 cassandra cassandra  11M Jul  3 11:54 lb-18-big-Index.db
-rw-r--r-- 1 cassandra cassandra 773K Jul  3 11:54 lb-18-big-Filter.db
-rw-r--r-- 1 cassandra cassandra  78K Jul  3 11:54 lb-18-big-Summary.db
-rw-r--r-- 1 cassandra cassandra   10 Jul  3 11:54 lb-18-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra  33K Jul  3 11:54 lb-18-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 11:54 lb-18-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 11:54 lb-18-big-TOC.txt
-rw-r--r-- 1 cassandra cassandra  73M Jul  3 11:54 lb-27-big-Data.db
-rw-r--r-- 1 cassandra cassandra  11M Jul  3 11:54 lb-27-big-Index.db
-rw-r--r-- 1 cassandra cassandra 773K Jul  3 11:54 lb-27-big-Filter.db
-rw-r--r-- 1 cassandra cassandra  78K Jul  3 11:54 lb-27-big-Summary.db
-rw-r--r-- 1 cassandra cassandra   10 Jul  3 11:54 lb-27-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra  33K Jul  3 11:54 lb-27-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 11:54 lb-27-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 11:54 lb-27-big-TOC.txt
-rw-r--r-- 1 cassandra cassandra 111K Jul  3 11:55 lb-28-big-Filter.db
-rw-r--r-- 1 cassandra cassandra 1.6M Jul  3 11:55 lb-28-big-Index.db
-rw-r--r-- 1 cassandra cassandra  12K Jul  3 11:55 lb-28-big-Summary.db
-rw-r--r-- 1 cassandra cassandra  11M Jul  3 11:55 lb-28-big-Data.db
-rw-r--r-- 1 cassandra cassandra   10 Jul  3 11:55 lb-28-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra 4.7K Jul  3 11:55 lb-28-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 11:55 lb-28-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 11:55 lb-28-big-TOC.txt
-rw-r--r-- 1 cassandra cassandra  15K Jul  3 13:10 lb-29-big-Filter.db
-rw-r--r-- 1 cassandra cassandra 206K Jul  3 13:10 lb-29-big-Index.db
-rw-r--r-- 1 cassandra cassandra 1.6K Jul  3 13:10 lb-29-big-Summary.db
-rw-r--r-- 1 cassandra cassandra 1.4M Jul  3 13:10 lb-29-big-Data.db
-rw-r--r-- 1 cassandra cassandra   10 Jul  3 13:10 lb-29-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra  651 Jul  3 13:10 lb-29-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 13:10 lb-29-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 13:10 lb-29-big-TOC.txt
```

yadax

COMPACTION

yadax

Force um compact

- nodetool –u cassandra –pw compact killrvideo
- O que houve?
- O que nodetool compact faz?
- Qual o tamanho do arquivo de dados?

yadax

COMPACT

```
total 269M
drwxr-xr-x 2 cassandra cassandra    6 Jul  3 11:51 backups
-rw-r--r-- 1 cassandra cassandra  74M Jul  3 11:53 lb-9-big-Data.db
-rw-r--r-- 1 cassandra cassandra  11M Jul  3 11:53 lb-9-big-Index.db
-rw-r--r-- 1 cassandra cassandra 773K Jul  3 11:53 lb-9-big-Filter.db
-rw-r--r-- 1 cassandra cassandra  78K Jul  3 11:53 lb-9-big-Summary.db
-rw-r--r-- 1 cassandra cassandra    9 Jul  3 11:53 lb-9-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra  33K Jul  3 11:53 lb-9-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 11:53 lb-9-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 11:53 lb-9-big-TOC.txt
-rw-r--r-- 1 cassandra cassandra  74M Jul  3 11:54 lb-18-big-Data.db
-rw-r--r-- 1 cassandra cassandra  11M Jul  3 11:54 lb-18-big-Index.db
-rw-r--r-- 1 cassandra cassandra 773K Jul  3 11:54 lb-18-big-Filter.db
-rw-r--r-- 1 cassandra cassandra  78K Jul  3 11:54 lb-18-big-Summary.db
-rw-r--r-- 1 cassandra cassandra   10 Jul  3 11:54 lb-18-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra  33K Jul  3 11:54 lb-18-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 11:54 lb-18-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 11:54 lb-18-big-TOC.txt
-rw-r--r-- 1 cassandra cassandra  73M Jul  3 11:54 lb-27-big-Data.db
-rw-r--r-- 1 cassandra cassandra  11M Jul  3 11:54 lb-27-big-Index.db
-rw-r--r-- 1 cassandra cassandra 773K Jul  3 11:54 lb-27-big-Filter.db
-rw-r--r-- 1 cassandra cassandra  78K Jul  3 11:54 lb-27-big-Summary.db
-rw-r--r-- 1 cassandra cassandra   10 Jul  3 11:54 lb-27-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra  33K Jul  3 11:54 lb-27-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 11:54 lb-27-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 11:54 lb-27-big-TOC.txt
-rw-r--r-- 1 cassandra cassandra 111K Jul  3 11:55 lb-28-big-Filter.db
-rw-r--r-- 1 cassandra cassandra 1.6M Jul  3 11:55 lb-28-big-Index.db
-rw-r--r-- 1 cassandra cassandra  12K Jul  3 11:55 lb-28-big-Summary.db
-rw-r--r-- 1 cassandra cassandra  11M Jul  3 11:55 lb-28-big-Data.db
-rw-r--r-- 1 cassandra cassandra   10 Jul  3 11:55 lb-28-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra 4.7K Jul  3 11:55 lb-28-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 11:55 lb-28-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 11:55 lb-28-big-TOC.txt
-rw-r--r-- 1 cassandra cassandra  15K Jul  3 13:10 lb-29-big-Filter.db
-rw-r--r-- 1 cassandra cassandra 206K Jul  3 13:10 lb-29-big-Index.db
-rw-r--r-- 1 cassandra cassandra 1.6K Jul  3 13:10 lb-29-big-Summary.db
-rw-r--r-- 1 cassandra cassandra 1.4M Jul  3 13:10 lb-29-big-Data.db
-rw-r--r-- 1 cassandra cassandra   10 Jul  3 13:10 lb-29-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra  651 Jul  3 13:10 lb-29-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 13:10 lb-29-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 13:10 lb-29-big-TOC.txt
```

```
total 216M
drwxr-xr-x 2 cassandra cassandra    6 Jul  3 11:51 backups
-rw-r--r-- 1 cassandra cassandra 186M Jul  3 13:13 lb-30-big-Data.db
-rw-r--r-- 1 cassandra cassandra  28M Jul  3 13:13 lb-30-big-Index.db
-rw-r--r-- 1 cassandra cassandra 2.0M Jul  3 13:13 lb-30-big-Filter.db
-rw-r--r-- 1 cassandra cassandra 198K Jul  3 13:13 lb-30-big-Summary.db
-rw-r--r-- 1 cassandra cassandra   10 Jul  3 13:13 lb-30-big-Digest.adler32
-rw-r--r-- 1 cassandra cassandra  84K Jul  3 13:13 lb-30-big-CompressionInfo.db
-rw-r--r-- 1 cassandra cassandra 9.7K Jul  3 13:13 lb-30-big-Statistics.db
-rw-r--r-- 1 cassandra cassandra   94 Jul  3 13:13 lb-30-big-TOC.txt
```

yadax

ADD NODES NO CLUSTER

yadax

Inicie o segundo node do mesmo DC

- Inicie o serviço do Cassandra, em um node NÃO SEED
- Acompanhe o system.log
- É necessário que os nodes se falem na porta 7000 (listen_port)

yadax

Inicie o segundo node do mesmo DC

- O que você espera que aconteça com o volume de dados nos nodes?
- Confira o status do cluster
- O que houve?

STREAM

yadax

STREAM

- Sempre que add um node não seed (bootstrap)
- Acompanhe os alerts
- Verifique o status do cluster
- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/sales...
- SEEDs não fazem bootstrap automaticamente

yadax

Stream

- O que é esse Stream? Node 2:

```
INFO  [main]  StorageService.java:1212 - JOINING: schema complete, ready to bootstrap
INFO  [main]  StorageService.java:1212 - JOINING: waiting for pending range calculation
INFO  [main]  StorageService.java:1212 - JOINING: calculation complete, ready to bootstrap
INFO  [main]  StorageService.java:1212 - JOINING: getting bootstrap token
INFO  StreamResultFuture.java:169 - [Stream #7a128490-792b-11ea-8aab-d5f79dd6ff78 ID#0] Prepare completed. Receiving 1 files(2345074 bytes), sending 0 files(0 bytes)
INFO  StreamResultFuture.java:169 - [Stream #7a128490-792b-11ea-8aab-d5f79dd6ff78 ID#0] Prepare completed. Receiving 3 files(1912877 bytes), sending 0 files(0 bytes)
```

Start o terceiro node

- System.log Node 1

```
Creating new streaming plan for Bootstrap
Received streaming plan for Bootstrap
Prepare completed. Receiving 0 files(0 bytes), sending 2 files(1681762 bytes)
Session with /172.31.93.205 is complete
All sessions completed
```

yadax

CLEANUP

Limpeza dos dados

- Os nodes entregam os dados para o novo node, mas não os eliminam do próprio node.
- É necessário forçar essa limpeza no nodes “antigos”. Acompanhe o system.log
- nodetool –u cassandra –pw cassandra cleanup

yadax

Limpeza dos dados

- Qual o tamanho do arquivo de dados no node1 e node2?

- Sempre que um node for adicionado em um DC existente, é necessário fazer o cleanup nos nodes antigos.

yadax

Aumento do Replication Factor

- Altere o replication factor da keyspace yadax para dc1: 2

- ALTER KEYSPACE yadax WITH replication = {'class': 'NetworkTopologyStrategy', dc1': '2'};

yadax

Aumento do Replication Factor

- Qual volume de dados cada node deveria ter?

- Verifique o status do cluster

yadax

REPAIR

yadax

Repair

- nodetool repair
  - --full
  - --pr (partitioner range)
  - Default, incremental – Evite no cassadra < 4.0

- nodetool repair --full (acompanhe o alert)

yadax

Repair

- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/sales... no segundo node
- Qual tamanho do arquivo de dados?
- Best Practice: rodar –pr em sequencia em todos os nodes

yadax

Terceiro node

- Quando adicionarmos o terceiro node, com qual volume de dados ele deve ficar?

- 215M x 2 = 430M (duas cópias)
- 430 / 3 = 143M (em cada node)

- A distribuição não será exata.

yadax

Start o terceiro node -> SEED

- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/sales... no terceiro node
- O que aconteceu?
- No node3, nodetool repair --full

yadax

Start o terceiro node -> SEED

- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/business... no terceiro node
- Qual o tamanho do arquivo de dados?
- Se tiver mais de um arquivo de dados, compact:
- nodetool compact yadax

yadax

Limpeza dos dados

- Qual o tamanho do arquivo de dados no node1 e node2?
- Compacte a keyspace yadax
- Qual o tamanho do arquivo de dados no node1 e node2?
- Por que não diminuiu? Como diminuir?

yadax

Adicionar nodes

- Qual é o procedimento para adicionar nodes?
- Já o fizemos duas vezes

yadax

DECOMMISSION

yadax

Remover node

- E como removemos nodes?
- Se o node estiver vivo, podemos gentilmente pedir para ele sair.
- nodetool decommission + (stop cassandra ou kill java)
- É o método mais “sutil”, deve ser a primeira opção

yadax

Remover node 3

- Verifique o status do cluster
- Acompanhe o system.log
- Remova o node 3 e mate o processo java.
- Qual o tamanho do arquivo de dados no node1 e 2?

yadax

Remover node 3

- Node3: INFO [RMI TCP Connection(10)- StorageService.java:1212 - DECOMMISSIONED

- Node1 e 2:
  - Creating new streaming plan for Unbootstrap
  - Received streaming plan for Unbootstrap
  - Prepare completed. Receiving 1 files(1651921 bytes), sending 0 files(0 bytes)
  - StorageService.java:2262 - Removing tokens [-1763480119689959618, -2546962922378197620, -29360279323088258O5 ...

yadax

Remover node 3

- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/sales...
- Qual tamanho do arquivo de dados nos nodes 1 e 2?
- Compacte
- Qual o tamanho do arquivo de dados nos nodes 1 e 2?

yadax

Adicionar o node 3

- Verifique o processo java
- Mate o processo
- kill -9 <pid>

yadax

Adicionar o node 3

- Tentar add sem limpar
- Verifique o status do cluster
- O que houve?
- Qual tamanho do arquivo de dados no node 3?

yadax

Adicionar o node 3

- Por que o tamanho é este?

- Tente compacar.
  - O que houve?

- Tente limpar (cleanup).
  - O que houve?
  - Por que?

yadax

Remova o node 3

- Pare o node3.
  - Stop do serviço.
  - Confira se o processo java está rodando.
- No node 3, tente o nodetool decommission
- O que houve?

yadax

REMOVENODE

yadax

Remova o node 3

- Confira o status do cluster
- A partir de um node sobrevivente, remova o node 3.
- nodetool removenode <host_id>

yadax

Remova o node 3

- StorageService.java:2262 - Removing tokens [-8427423350248595549, -65704217364540608 05, ...
- HintedHandOffManager.java:225 - Deleting any stored hints for /172.31.87.71
- Gossiper.java:550 - Completing removal of /172.31.87.71

yadax

Adicione o node 3

- Limpe todos os dados do node 3
- rm -rf /var/lib/cassandra/*/*
- Adicione novamente o node 3
- Confira o status do cluster

yadax

ADD NOVO DC

yadax

Adicionando o segundo DC

- Start o node 1B.
- Acompanhe o alert
- Confira o status do cluster
- O que mudou?

yadax

Adicionando o segundo DC

- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/sales... no node 1B.

- Por que?

yadax

Adicionando o segundo DC

- Altere o replication factor da keyspace yadax
- ALTER KEYSPACE yadax WITH replication = {'class': 'NetworkTopologyStrategy', 'DCA': '2', 'DCB': '2'};
- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/business... no node 1B.
- O que precisamos fazer?

yadax

REBUILD

yadax

Adicionando o segundo DC

- Pode ser feito um repair full
- Ou reconstruir completamente o node
- Acompanhe o system.log
- nodetool rebuild

yadax

Adicionando o segundo DC

- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/sales... no node 1B.
- O que houve?
- Compacte
- Qual o tamanho do arquivo de dados?

yadax

Adicione o node 2B

- Start o node 2B
- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/sales... no node 2B.
- Confira o status do cluster
- O que houve?
- Qual o tamanho do arquivo de dados?

yadax

Adicione o node 3B

- Start o node 3B
- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/sales... no node 3B.
- O que houve?
- Qual o tamanho do arquivo de dados? Está correto?

yadax

Sincronize os dados

- O node 2B é responsável por dados que ele não tem.
- No node 2B, tente um nodetool repair --full
- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/sales... nos nodes 2B e 3B.
- O que houve?

yadax

Sincronize os dados

- No node 3B, tente um nodetool repair --full
- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/business... nos nodes 2B e 3B.
- O que houve?

yadax

Status do Cluster

- Verifique o status do cluster

- Limpe (cleanup) os dados nos nodes 1A, 1B, 2A, 2B

yadax

REMOVER DC

yadax

Remover DCA

- Verifique o status do cluster
- Remova todos os nodes do DCA
- Mate os processos java

yadax

Remover DCA

- Confira o status do cluster
- Altere a keyspace yadax para ter replicas apenas no DCB
- ALTER KEYSPACE yadax WITH replication = {'class': 'NetworkTopologyStrategy', 'DCB': '2'};

yadax

MIGRAÇÃO DC

yadax

Migração de DataCenter

- Este foi o procedimento para migração de DataCenter.

- Como é mesmo?

yadax

BACKUP

yadax

Backup

- O backup pode ser feito por keyspace
- Se você não informar a keyspace, todas terão backup
- O backup é um snapshot da keyspace (das SSTables)
- É um hardlink, por isso no mesmo Filesystem

yadax

Backup

- Você precisa copiar para outro local.

- nodetool -u cassandra -pw cassandra snapshot killrvideo

```
Requested creating snapshot(s) for [killrvideo] with snapshot name
[1586217410504]
Snapshot directory: 1586217410504
```

yadax

Backup

- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/sales... no node que fez o snapshot

- O que houve?

```
[root@ip-172-31-90-247 ~]# ls -lhrt /var/lib/cassandra/data/yadax/business_price-6c285960784d11ea9aa13b7649deea08/
total 5.9M
drwxr-xr-x. 2 cassandra cassandra    6 Apr  6 23:14 backups
-rw-r--r--. 2 cassandra cassandra 5.8M Apr  6 23:25 lb-5-big-Data.db
-rw-r--r--. 2 cassandra cassandra  33K Apr  6 23:25 lb-5-big-Index.db
-rw-r--r--. 2 cassandra cassandra 1008 Apr  6 23:25 lb-5-big-Filter.db
-rw-r--r--. 2 cassandra cassandra  262 Apr  6 23:25 lb-5-big-Summary.db
-rw-r--r--. 2 cassandra cassandra    9 Apr  6 23:25 lb-5-big-Digest.adler32
-rw-r--r--. 2 cassandra cassandra 4.1K Apr  6 23:25 lb-5-big-CompressionInfo.db
-rw-r--r--. 2 cassandra cassandra 6.9K Apr  6 23:25 lb-5-big-Statistics.db
-rw-r--r--. 2 cassandra cassandra   94 Apr  6 23:25 lb-5-big-TOC.txt
drwxr-xr-x. 3 cassandra cassandra   27 Apr  6 23:56 snapshots
```

yadax

Backup

- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/sales.../snapshots/ 1586217410504/

- O que houve?

```
[root@ip-172-31-90-247 ~]# ls -lhrt /var/lib/cassandra/data/yadax/business_price-6c285960784d11ea9aa13b76
49deea08/snapshots/1586217410504/
total 5.9M
-rw-r--r--. 2 cassandra cassandra 5.8M Apr  6 23:25 lb-5-big-Data.db
-rw-r--r--. 2 cassandra cassandra  33K Apr  6 23:25 lb-5-big-Index.db
-rw-r--r--. 2 cassandra cassandra 1008 Apr  6 23:25 lb-5-big-Filter.db
-rw-r--r--. 2 cassandra cassandra  262 Apr  6 23:25 lb-5-big-Summary.db
-rw-r--r--. 2 cassandra cassandra    9 Apr  6 23:25 lb-5-big-Digest.adler32
-rw-r--r--. 2 cassandra cassandra 4.1K Apr  6 23:25 lb-5-big-CompressionInfo.db
-rw-r--r--. 2 cassandra cassandra 6.9K Apr  6 23:25 lb-5-big-Statistics.db
-rw-r--r--. 2 cassandra cassandra   94 Apr  6 23:25 lb-5-big-TOC.txt
-rw-r--r--. 1 cassandra cassandra   31 Apr  6 23:56 manifest.json
```

yadax

Backup

- Liste os arquivos da pasta com a opção -i /var/lib/cassandra/data/yadax/sales.../snapshots/1586217410504/

- Liste os arquivos da pasta com a opção -i /var/lib/cassandra/data/yadax/sales.../

- O que houve?

yadax

Desastre

- Para o restore, a keyspace e as tabelas precisam existir.

- Descreva sua keyspace e guarde o resultado

yadax

TRUNCATE

yadax

Desastre

- Truncate table killrvideo.actors_by_video

```
select * from killrvideo.actors_by_video limit 10;
```

- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/sales... em todos os nodes do DCB.

yadax

RESTORE

yadax

Restore

- Copie os arquivos da pasta ...../snapshot/158637NNNNNNNN/* para /var/lib/cassandra/data/yadax/sales-....

- Em todos os nodes

- Se usou o root, modifique o owner dos arquivos:

  # chown cassandra:cassandra /var/lib/cassandra/data/yadax/sales-.../*

- `select * from yadax.sales limit 10;`

yadax

REFRESH

Restore

- Faça o reload do snapshot em todos os nodes
- nodetool refresh yadax sales (A tabela precisar existir)
- Acompanhe o alert
- `select * from yadax.sales limit 10 ;`

yadax

Restore

- Liste os arquivos da pasta /var/lib/cassandra/data/yadax/sales-.../snapshots/

- Por que tem dois diretórios?

- auto_snapshot: true

Whether or not a snapshot is taken of the data before keyspace truncation

or dropping of column families. The STRONGLY advised default of true

should be used to provide data safety. If you set this flag to false, you will

lose data on truncation or drop.

yadax

SNAPSHOTS

yadax

Snapshots

- Liste seus snapshots:
  - nodetool listsnapshots
- Limpe os snapshots:
  - nodetool clearsnapshot (todos)
    - -t <snapshot_name>
    - <keyspaces>
- Liste seus snapshots

yadax

Snapshots

- Crie um snapshot da keyspace yadax em um node apenas

- Qual o espaço ocupado pela pasta /var/lib/cassandra/data/yadax/sales...?

```
du -sh /var/lib/cassandra/data/yadax/*
```

yadax

Snapshots

- Qual o espaço ocupado apenas pela pasta /var/lib/cassandra/data/yadax/sales.../snapshots?

```
du -sh /var/lib/cassandra/data/yadax/sales-.../snapshots
```

- Tente:

```
du -sh /var/lib/cassandra/data/yadax/sales-.../*
```

- O que mudou?

yadax

UPGRADE

yadax

Adicione o DCA

- Verifique o status do cluster
- Limpe os dados dos nodes do DCA
- Adicione os todos os nodes no DCA

yadax

Adicione o DCA

- Mude o replication fator da yadax para DCA:2, DCB:2
- Faça um “nodetool repair --full” dos nodes do DCA
- Verifique o status do cluster
- Verifique o tamanho dos arquivos de dados

yadax

Upgrade

- Guarde a DDL de criação da sua keyspace, faça um backup em todos os nodes
- Verifique a compatibilidade da app com a nova versão
- Acompanhe sempre o system.log
- O formato da SSTable muda de versão para versão.

yadax

DESCRIBECLUSTER

yadax

UPGRADE - DESCRIBECLUSTER

- nodetool describecluster
- Limpe todos os snapshots antigos que existirem
- Faça um “nodetool repair -pr” em todos os nodes, em série
- Faça um snapshot (full) em todos os nodes

yadax

UPGRADE - CUIDADOS

- Durante o upgrade evite:
  - Novas features
  - Repair
  - Add/Remove nodes
  - DDLs

yadax

UPGRADE - CUIDADOS

- Durante o upgrade evite:
  - Habilitar CDC (change data capture)
  - Alterar credenciais e permissões.
- Sequencia
  - Primeiro DC: SEEDs, depois outros nodes
  - Segundo DC: SEEDs, depois outros nodes
- Faça uma nota da sequencia que seguirá

yadax

UPGRADE - DRAIN

- No primeiro SEED do DCA:
- Drain: Remove as conexões, faz flush das memtables.
- Shutdown friendly
- nodetool drain

yadax

UPGRADE

- Pare o serviço do cassandra
- Verifique se o processo java está down.
- Remova o cassandra 2.2 – yum remove cassandra
  - Preste bastante atenção no output
- O que aconteceu com os dados?

yadax

UPGRADE

- Altere o repositório do cassandra: /etc/yum.repos.d/cassandra.repo

```
[cassandra]
baseurl = https://www.apache.org/dist/cassandra/redhat/311x/
gpgcheck = 1
gpgkey = https://www.apache.org/dist/cassandra/KEYS
name = Apache Cassandra
repo_gpgcheck = 1
```

yadax

UPGRADE

- Instale o cassandra 3.11 – yum install cassandra
- Edite novamente o cassandra.yaml e cassandra-rackdc.properties

yadax

Ajuste - cassandra.yaml

- Estes são os parâmetros que você vai precisar alterar no cassandra.yaml:
  - cluster_name: Yadax
  - num_tokens: 16
  - seeds: ip1, ip2, ip4, ip5
  - # listen_address -> comentar (# no começo da linha)
  - listen_interface -> eth0 (utilizar “ip a” para descobrir)
  - # rpc_address: -> comentar (# no começo da linha)
  - rpc_interface: eth0
  - endpoint_snitch: GossipingPropertyFileSnitch

yadax

Ajuste - cassandra-rackdc.properties

- Estes são os parâmetros que você vai precisar alterar no cassandra-rackdc.properties:

  dc=DCA

  rack=RACK1

yadax

UPGRADE

- Inicie o service do Cassandra
- INFO [main] SystemKeyspace.java:1433 - Detected version upgrade from 2.2.NN to 3.11.N, snapshotting system keyspace
- nodetool describecluster
- Liste os arquivos de dados e repare seus nomes e size atuais
- nodetool upgradesstables

yadax

UPGRADE

- O que houve com os nomes dos arquivos?
- O que houve com o size?
- Como acompanhar o andamento do upgradesstables?

yadax

UPGRADE

- Faça o mesmo procedimento nos outros SEEDs do mesmo DC
- Faça o mesmo procedimento nos outros nodes do mesmo DC
- Faça o mesmo procedimento no outro DC, começando pelos SEEDs.

yadax

UPGRADE

- Drain / stop service
- Uninstall
- Repo
- Install
- Arquivos de conf
- Start
- Upgrade sstables
- Describe Cluster

yadax

CONSISTÊNCIA

yadax

CONSISTENCY

- No Cassandra temos consistência ajustável
- Por default, “always writable”. Se tiver um nó em pé, está escrevendo
- Se você aumentar a consistência, você perde a tolerância a falha
- Teorema CAP

yadax

CONSISTENCY

- Verifique o status do cluster

- Conecte no CQLSH no node 1A e verifique a consistência default.

```
cqlsh> CONSISTENCY
Current consistency level is ONE.
```

yadax

CONSISTENCY

- Crie uma Keyspace com RF=3 no DCA.

```
cqlsh> create keyspace consist WITH replication = {'class':
'NetworkTopologyStrategy','DCA': 3};
```

yadax

CONSISTENCY

- Crie uma Tabela na KS consist

```
cqlsh> USE consist;
cqlsh:consist> CREATE TABLE scores
(
  user TEXT,
  game TEXT,
  year INT,
  month INT,
  day INT,
  score INT,
  PRIMARY KEY (user, game, year, month, day)
);
```

yadax

CONSISTENCY

- Insira um registro

```
cqlsh:consist> INSERT INTO scores (user, game, year, month, day, score)
VALUES ('hack', 'Mario', 2015, 05, 01, 4000);
```

yadax

CONSISTENCY

- Stop os outros dois nodes do DCA, confira o status do cluster
- Você consegue ler o registro?
- Tente inserir um novo registro

```
cqlsh:consist> INSERT INTO scores (user, game, year, month, day, score)
VALUES ('abonacin', 'Mario', 2015, 05, 03, 1750);
```

- O que houve? Você consegue ler o novo registro?

yadax

CONSISTENCY

- Mude a consistência para Quorum. Tente ler novamente.

```
cqlsh:consist> CONSISTENCY QUORUM ;
cqlsh:consist> select * from consist.scores ;
```

- O que houve?

yadax

CONSISTENCY

- Tente inserir um registro.

```
cqlsh:consist> INSERT INTO scores (user, game, year, month,
day, score) VALUES ('japa', 'Mario', 2015, 05, 03, 2250);
```

- O que houve? Por que?

yadax

HINTED HANDOFF

yadax

HINTED HANDOFF

- Acompanhe o system.log do node 1A
- Start o node 2A, confira o status do cluster
- O que houve?

```
INFO  [HintedHandoff:1]  HintedHandOffManager.java:362 -
Started hinted handoff for host:
5b1078c3-804b-4ec7-ab26-bd4d4f358abb with IP:
/172.31.88.164
INFO  [HintedHandoff:1]  HintedHandOffManager.java:394 -
Finished hinted handoff of 1 rows to endpoint
/172.31.88.164
```

yadax

CONSISTENCY

- Conecte no CQLSH no node 1A, ajuste a consistência para Quorum.
- Tente ler os registros. Tente inserir um registro

```
cqlsh:consist> INSERT INTO scores (user, game, year, month, day, score)
VALUES ('japa', 'Mario', 2015, 05, 03, 2250);
```

- O que houve? Por que?

yadax

CONSISTENCY

- Conecte no CQLSH no node 1A, ajuste a consistência para ALL.
- Tente ler os registros. Tente inserir um registro

```
INSERT INTO scores (user, game, year, month, day, score) VALUES ('asia',
'Mario', 2015, 05, 03, 500);
```

- O que houve? Por que?

yadax

HINTED HANDOFF

- Acompanhe o system.log do node 1A
- Start o node 3A, confira o status do cluster
- O que houve?

```
INFO  [HintedHandoff:2] HintedHandOffManager.java:362 -
Started hinted handoff for host:
4dc6319c-8f88-4fe2-8709-267b7adc2b53 with IP:
/172.31.88.210
INFO  [HintedHandoff:2] HintedHandOffManager.java:394 -
Finished hinted handoff of 2 rows to endpoint
/172.31.88.210
```

yadax

CONSISTENCY

- Conecte no CQLSH no node 1A, ajuste a consistência para ALL.
- Tente ler os registros. Tente inserir um registro

```
INSERT INTO scores (user, game, year, month, day, score) VALUES ('asia',
'Mario', 2015, 05, 03, 500);
```

- O que houve? Por que?

yadax

READ REPAIR

yadax

READ REPAIR

- Desabilite o hinted handoff de todos nodes do DCA – cassandra.yaml
  - hinted_handoff_enabled: false
- Restarte o cassandra nos nodes do DCA
- Stop os nodes 2A e 3A
- Verifique o status do cluster

yadax

READ REPAIR

- Conecte no CQLSH no node 1A, ajuste a consistência para One.
- Tente ler os registros. Tente inserir um registro

```
cqlsh:consist> INSERT INTO scores (user, game, year, month, day, score) VALUES
('abonacin', 'MK3', 2015, 06, 01, 2500);
```

- Tente ler os registros. O que houve?

yadax

READ REPAIR

- Acompanhe o system.log do node 1A
- Starte os outros nodes
- Houve entrega de hints?

yadax

READ REPAIR

- Seu registro ficou no node 1A
- Stop o node 1A
- Conecte no CQLSH no node 2A, ajuste a consistência para One.
- Tente ler os registros. O que houve?

yadax

READ REPAIR

- Starte o node 1A
- Conecte no CQLSH no node 1A, ajuste a consistência para ONE.
- Tente ler os registros. O que houve?
- Conecte no CQLSH nos nodes 2A e 3A, ajuste a consistência para ONE.
- Tente ler os registros. O que houve?

yadax

READ REPAIR

- Conecte no CQLSH no node 2A, ajuste a consistência para ALL.
- Tente ler os registros. O que houve?
- Conecte no CQLSH no node 3A, ajuste a consistência para ONE.
- Tente ler os registros. O que houve?

yadax

AUTENTICAÇÃO

yadax

Autenticação

- As infos de autenticação encontram-se na system_auth.
- Altere a keyspace system_auth para ter 3 replicas em cada DC, com NetworkTopologyStrategy
- Faça repair da system_auth (--full) em todos os nodes (um de cada vez).

yadax

Autenticação

- Altere o cassandra.yaml em todos os nodes

- authenticator: PasswordAuthenticator
- authorizer: CassandraAuthorizer

- Restart o node

yadax

Conecte no cassandra

```
$ echo $HOSTNAME
ip-172-31-89-224.ec2.internal
$ cqlsh $HOSTNAME
Connection error: ('Unable to connect to any
servers', {'172.31.89.224':
AuthenticationFailed('Remote end requires
authentication.',)})
```

Agora precisamos de user/senha: cassandra/cassandra

yadax

Conectado com sucesso

```
$ cqlsh -u cassandra -p cassandra $HOSTNAME
Connected to ABonacin at ip-172-31-89-224.ec2.internal:9042.
[cqlsh 5.0.1 | Cassandra 2.2.14 | CQL spec 3.3.1 | Native protocol v4]
Use HELP for help.
cassandra@cqlsh>
```

AMEM!

yadax

Super user

- Crie um user “dba” como superuser

```
cqlsh> create user dba with password 'Yadax2020'   superuser;
```

yadax

User Cassandra x DBA

```
$ cqlsh -u cassandra -p cassandra $HOSTNAME
```

```
$ cqlsh -u dba -p Yadax2020 $HOSTNAME
```

- Baixe todo o DCB

yadax

User Cassandra x DBA

- Verifique o status do cluster

- Se conecte de novo

```
$ cqlsh -u cassandra -p cassandra $HOSTNAME
```

```
$ cqlsh -u dba -p Yadax2020 $HOSTNAME
```

yadax

User Cassandra x DBA

- O que houve?

- "Unable to perform authentication: Cannot achieve consistency level QUORUM"

yadax

Sendo preguiçoso

Crie um arquivo ~/.cassandra/cqlshrc
~ é um alias para o home do user: /home/cassandra

```
$ vi ~/.cassandra/cqlshrc
[authentication]
username = dba
password = Yadax2020

[connection]
hostname = ip-172-31-89-224
port = 9042
```

yadax

Sendo preguiçoso

Tente agora:

```
$ cqlsh
```

```
Parabéns, você é um preguiçoso!
```

yadax

JMX

- Verifique a saúde do cluster
- Você continua não precisando de user e senha.
- Por que?
- cassandra-env.sh -> onde alteramos as configs do JMX

yadax

JMX

- /etc/cassandra/conf/cassandra-env.sh
- Busque pelo trecho:

```
if [ "x$LOCAL_JMX" = "x" ]; then
```

- Adicione imediatamente antes:

```
LOCAL_JMX=no
```

```
LOCAL_JMX="no"
if [ "x$LOCAL_JMX" = "x" ]; then
    LOCAL_JMX=yes
fi
```

yadax

JMX

- Busque pelo trecho abaixo e ajuste:

```
JVM_OPTS="$JVM_OPTS
-Dcom.sun.management.jmxremote.password.file=/etc/cassandra/conf/jmxremote.password"

JVM_OPTS="$JVM_OPTS
-Dcom.sun.management.jmxremote.access.file=/etc/cassandra/conf/jmxremote.access"
```

yadax

JMX - jmxremote.access

Crie o arquivo /etc/cassandra/conf/jmxremote.access

```
# cat /etc/cassandra/conf/jmxremote.access
cassandra readonly
yadax readwrite
```

yadax

JMX - jmxremote.password

- Crie o arquivo /etc/cassandra/conf/jmxremote.password

```
# cat /etc/cassandra/conf/jmxremote.password
cassandra Yadax2020
yadax Yadax2020
```

- Restrinja o acesso ao arquivo

```
chmod 600 /etc/cassandra/conf/jmxremote.password
```

yadax

JMX

- Restarte todos os nodes
- Verifique a saúde do cluster

```
# nodetool -u cassandra -pw Yadax2020 -h $HOSTNAME status
```

- O que houve?

yadax

JMX

```
# nodetool -u yadax -pw Yadax2020 -h $HOSTNAME status
```

- O que houve?

yadax

USERS: JMX x DB

- Yadax é um user do DB?

cqlsh

```
dba@cqlsh> list users;
dba@cqlsh> list roles;
```

yadax

HEAP SIZE

Configuração HEAP

- Por default, a HEAP tem 1/4 do tamanho da RAM
- Se você tiver muito problema com GC, é possível ajustar este size
- Nem sempre HEAP grande é bom
- Arquivos conf
  - 2.2: cassandra-env.sh
  - 3.+: jvm.options

yadax

HEAP

- Veja o tamanho atual da sua HEAP

- ps –ef | grep java

-Xms1024M
-Xmx1024M

yadax

Ajuste no jvm.options

- Vamos ajustar o /etc/cassandra/conf/jvm.options

- Busque pelo trecho abaixo

```
#-Xms4G
#-Xmx4G
```

Descomente e altere para 1500M

```
-Xms1500M
-Xmx1500M
```

yadax

Ajuste no jvm.options

- Restart o node

- Verifique o resultado

yadax

BEST PRACTICES

(TA ACABANDO)

yadax

Best Practices

- User DBA – SuperUSER
- Keyspaces usando NetworkTopology
- Repair periódico
- Distribuição de Racks (Rackawareness)

yadax

Best Practices

- NTP
- Kernel e Limits
- Não usar SWAP

yadax

Best Practices

- Usar SSDs
- Scheduler IO: deadline ou noop
- Desabilitar Transp Huge Pages
- HEAP entre 1/4 e 1/2 da RAM, nunca maior que 32G

yadax

Obrigado!!!

yadax