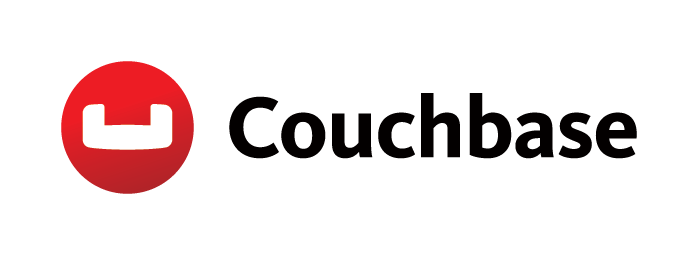


**MIT EM BIG DATA**

**BLOCO C – ARMAZENAMENTO HETEROGÊNEO DE DADOS**



**Alunos: Milton Bahia**

**Roberto Rio**

**Vinicius Rosa**

**Sumário**

[1 INTRODUÇÃO 2](#_Toc451807255)

[2 DETALHAMENTO TEÓRICO 3](#_Toc451807256)

[2.1 Aplicações 4](#_Toc451807257)

[2.2 Arquitetura 6](#_Toc451807258)

[2.2.1 Serviços 7](#_Toc451807259)

[2.2.2 Sharding 8](#_Toc451807260)

[2.2.3 Alta Disponibilidade e Arquitetura de Replicação 9](#_Toc451807262)

[2.2.4 Segurança 11](#_Toc451807263)

[2.3 Instalação 13](#_Toc451807264)

[2.4 Interação 14](#_Toc451807265)

[2.4.1 Acesso Chave-Valor 14](#_Toc451807266)

[2.4.2 Acesso através do N1QL 14](#_Toc451807267)

[2.4.3 Otimização de Consultas 16](#_Toc451807268)

[2.5 Tarefas Administrativas 17](#_Toc451807269)

[2.5.1 CouchBase Web Console 18](#_Toc451807270)

[2.5.2 Cluster Setup 19](#_Toc451807271)

[2.5.3 Backup e Restore 20](#_Toc451807272)

[3 EVIDÊNCIAS 22](#_Toc451807273)

[3.1 Realização de Operações CRUD a ferramenta N1QL 22](#_Toc451807274)

[4 CONCLUSÕES 24](#_Toc451807275)

[5 REFERÊNCIAS BIBLIOGRÁFICAS 25](#_Toc451807276)

[5.1 Documentação On-Line e Site de Referência 25](#_Toc451807277)

[5.2 Livros 25](#_Toc451807278)

# INTRODUÇÃO

Nos últimos anos houve uma explosão no número de banco de dados NoSQL, tendo atingido um número superior a 225 produtos segundo informações do site [nosql-database.org](http://nosql-database.org/).

Embora ainda representem uma pequena fatia no mercado total de banco de dados, a utilização de bancos NoSQL tem crescido graças à difusão da tecnologia Big Data. Neste cenário de Big Data, os bancos de dados NoSQL tem grande aderência em função da sua capacidade de armazenamento de altos volumes de dados a custos bem menores que as soluções de banco de dados relacionais, escalabilidade horizontal, esquemas flexíveis e da capacidade de trabalhar com dados não estruturados.

Diante do cenário de inovação e introdução do NoSQL ao mercado de banco de dados, aconteceram várias fusões de empresas, com a finalidade de criar produtos mais robustos e fortes no mercado. Uma dessas fusões foi a que envolveu o CouchDB e o Membase dando origem ao CouchBase em 2011.

O Membase foi desenvolvido por vários líderes do projeto memcached, os quais tinham fundado uma empresa, a NorthScale, para desenvolver um armazenamento de chave-valor com a simplicidade, velocidade e escalabilidade, mas também fornecendo o armazenamento, persistência e capacidades de consulta de um banco de dados. A CouchOne Inc. foi fundada em 2009 e desenvolveu uma distribuição comercial do projeto open source Apache CouchDB, um banco de dados NoSql orientado a documentos. O produto resultante dessa fusão foi o Couchbase Server, o qual reuniu os pontos fortes do MemBase e do CouchDB.

Em janeiro de 2012, foi lançado o Couchbase Server 1.8 e em dezembro de 2012, o Couchbase Server 2.0, com novos recursos, incluindo um novo armazenamento de documentos JSON, indexação e consulta, views MapReduce e replicação entre clusters. Atualmente o CouchBase está disponível na versão 4.1 e está em desenvolvimento a versão 4.5

Este trabalho tem como objetivo apresentar, em caráter exploratório, os conceitos, recursos e formas de utilização do banco de dados CouchBase Server.

# DETALHAMENTO TEÓRICO

O Couchbase Server é um projeto open-source de banco de dados NoSQL multi-modelo, podendo ser utilizado como armazenamento chave-valor, banco de dados orientado a documentos ou cache distribuído.

Todo os objetos ou itens no Couchbase são armazenados como Chave/Valor, sejam tipos de dados simples (string, número, booleano), documentos JSON ou dados no formato binário. A chave é tipicamente uma chave delegada (surrogate key), um identificador único e imutável, podendo ser gerada através de um contador ou UUID ou definida pelo usuário na hora da inserção.

A forma de acesso ao Couchbase é que define se ele está sendo utilizado como um Banco Chave/Valor ou um Banco de Dados Orientado a Documentos.

* **Armazenamento Chave / Valor**: Os valores são armazenados e recuperados através de um único atributo denominado de chave. Neste tipo de acesso, o conteúdo armazenado como valor é considerado “opaco” para o Couchbase, não podendo ser utilizado nos mecanismos de consulta.
* **Banco de Dados de Documento**: Quando são armazenados documentos JSON, os atributos dos documentos podem ser utilizados nas operações de consulta e atualização. Além disso é possível a criação de índices, views e agregações baseadas no conteúdo do documento.

Os itens (chave/valor ou documentos) são armazenados em contêineres lógicos denominados de Buckets. O conceito de Bucket no Couchbase é equivalente a definição de database (MySql) / esquema (Oracle) em bancos de dados relacionais ou até mesmo de uma tabela, dependendo da forma de utilização da ferramenta. Fazendo uma comparação dos componentes do Couchbase com um banco relacional temos o quadro abaixo:

| Relacional | Couchbase |
| --- | --- |
| Database / Esquema | Bucket |
| Tabela | Bucket ou itens (com atributo de tipo) |
| Linha | Documento ou Chave-Valor |
| Chave Primária | ID do Documento ou chave |
| Coluna | Atributo do Documento |
| Índice | Índice |
| View | View |
| Agregações | Map Reduce |

Tabela 1 - Comparação com o Banco de Dados Relacional

O Couchbase Server pode ser configurado para não realizar a persistência final dos dados em disco, utilizando apenas a memória. Neste modo de operação, o Couchbase funciona como um cache distribuído, proporcionando um acesso escalável e de baixa latência para grandes conjuntos de dados na memória, opcionalmente, por meio da *Memcache API*. A *Memcache API* provê uma *hash table* distribuída em múltiplas máquinas. Quando a tabela está cheia, inserções subsequentes fazem com que os dados antigos sejam expurgados utilizando um algoritmo LRU (Least Recently Used).

## Aplicações

Os casos de uso e aplicações do Couchbase são baseados nas suas principais características e vantagens em relação aos bancos de dados relacionais.

* **Modelo de dados flexível ou esquema dinâmico**: O armazenamento dos dados em documentos JSON permite um modelo de dados flexível ou dinâmico, com diferentes conjuntos de atributos e tipos em cada instância de documento. Os atributos podem ser inseridos ou retirados sem necessidade de migração de dados ou de alteração imediata de todos os documentos pré-existentes.
* **Escalabilidade**: O Couchbase é projetado para escalar horizontalmente de forma fácil em hardware comodity. A distribuição automática dos dados em múltiplos servidores (sharding), permite a balanceamento e distribuição da carga ao longo do cluster e a redução de gargalos no processamento.
* **Alta Disponibilidade**: A alta disponibilidade do Couchbase é implementada através de mecanismos de replicação de dados ao longo dos nós do cluster. Além disso, o Couchbase foi projetado para suportar operações de manutenção, tais como upgrade de software, build de índices, compactação e outras operações sem ficar offline.
* **Implementação em múltiplos Data Centers**: O Couchbase possui mecanismos de replicação de dados entre datacenters (XRDC) permitindo a operação em múltiplas posições geográficas com melhoria de desempenho e disponibilidade.

O Couchbase é vastamente utilizado, nos campos de Real-time Big Data, detecção de fraudes, Internet of Things (IoT), aplicativos móveis, gerenciamento de conteúdo, gerenciamento de profile entre outros. Essas aplicações em escala de internet, com constantes mudanças de requisitos, número crescente de usuários, alto volume de dados e alta velocidade, exploraram o seu alto desempenho, escalabilidade, baixa latência, modelo de dados flexível, e armazenamento de conteúdo semiestruturado e não estruturado.

Além disso, o Couchbase se integra ao contexto das tecnologias de Big Data por ser um banco NoSQL orientado a documentos com capacidade de armazenamento e processamento de alta escala. Arquiteturas modernas de Big Data combinam Hadoop e bancos NoSQL. O Hadoop é projetado para Big Data Analytics, mas ele não atende aos casos de Real Time. Os bancos NoSQL, tais como o Couchbase, são projetados para Big Data em tempo real, mas ele tem característica operacional ao invés de analítica.

Como caso de sucesso podemos citar o caso da empresa de Televisão DirecTV, que necessita servir canais de TV para milhões de usuários espalhados pelo mundo. Nesse caso eles precisam oferecer acesso aos serviços, mas somente aos serviços que o usuário tem acesso, seja para TV em sua casa ou até mesmo em canais acessados via Streaming em Smartphones, tablets e etc. Eles optaram por utilizar Couchbase para sua necessidade e conseguiram atingir seus objetivos.

Para exemplificar como o CouchBase atua no atendimento aos seus clientes no mercado de Big Data, citamos o caso de sucesso do Ebay. O Ebay selecionou o CouchBase para melhorar a experiência do usuário através de múltiplos canais, incluindo web e móvel, com o objetivo de apoiar mais de 2 bilhões de leituras por dia. Para este cliente, o CouchBase fornece baixa latência em acesso de leitura a um item, nos dados de comprador e vendedor, através de dois clusters implantados em dois data centers. Baixa latência em leitura e escrita com tokens de autenticação também são habilitados em clusters com três data centers.

O Couchbase também possui uma versão mobile, a qual é uma solução embarcada de banco de dados NoSQL que pode operar eventualmente de forma desconectada e realizar sincronismo com o Couchbase Server.

## Arquitetura

O CouchBase Server foi projetado para oferecer alta performance, escalabilidade e disponibilidade. Ele pode ser instalado em um único servidor ou num cluster abrangendo muitas máquinas, podendo operar também em múltiplos datacenters.

O cluster Couchbase tem uma arquitetura distribuída do tipo shared-nothing (nada compartilhado). Ele possui uma topologia ponto a ponto na qual todos os nós são iguais, com os mesmos componentes, serviços e interfaces. Não existem nós com funções específicas, tais como Master, Slave, Name Node e etc. Essa topologia ponto a ponto proporciona uma melhor performance e utilização de recursos, eliminando gargalos e pontos únicos de falha.

O Couchbase Server consiste nos seguintes componentes principais: **Cluster Manager, Data service, Index service, e Query service**. Os comportamentos de tempo de execução, tais como replicação, armazenamento, armazenamento em cache e assim por diante podem ser sintonizados com as necessidades dos diferentes serviços.

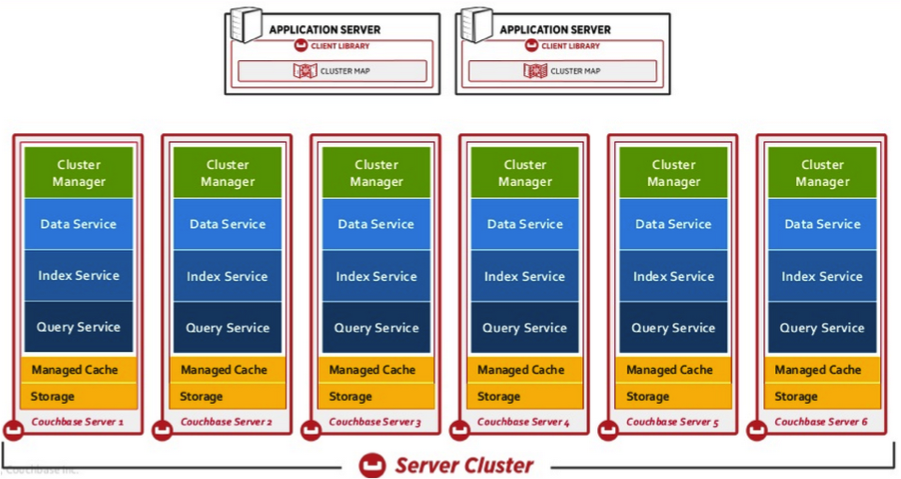


Figura 1 - Cluster, Nós e Serviços.

### Serviços

Os serviços do Couchbase são componentes que executam *processos* independentes e específicos dentro do cluster. O banco de dados lida com três *processos* distintos: **operações de dados, indexação e processamento de consultas**. O Couchbase Server possui serviços de dados, índice e queries para permitir a execução independente desses *processos* dentro de um cluster.

Cada nó pode executar todos os serviços ou um subconjunto dos serviços. O administrador pode criar topologias originais usando estes serviços independentes e dimensionar de forma independente os três *processos.*

* **Data Service**: Fornece a API chave-valor para execução de operações CRUD (criar, recuperar, atualizar e excluir) em itens com chaves dentro de “bucket”s.
* **Index Services**: Os índices fornecem acesso mais rápido aos dados em um “bucket”. O Couchbase provê três tipos de indexação: View MapReduce, View Espacial (Spatial View), Global Secondary Index (GSI)
* **Query Services:** Permite a recuperação dos dados de formas distintas no Couchbase Server, dependendo do tipo de consulta: Consulta chave/valor e consultas baseadas no valor ou atributos dos documentos.

### Sharding

No Couchbase, documentos logicamente relacionados são armazenados em buckets, os quais são equivalentes a esquemas em bancos de dados relacionais. O bucket é um espaço de chaves (Keyspace) único. Quando um bucket é criado é feito um provisionamento de recursos físicos específicos para ele, tais como a quantidade de memória RAM para cache e a configuração do número de réplicas.

Internamente, o Couchbase usa um mecanismo chamado vBuckets (equivalente a shard ou partição) para distribuir automaticamente os dados ao longo do cluster. Esse processo é denominado de autosharding. Os vBuckets habilitam a replicação de dados, failover e reconfiguração dinâmica do cluster. Usuários e aplicações não manipulam os vBuckets diretamente.

O Couchbase divide automaticamente cada Bucket em 1024 vBuckets ativos e 1024 vBuckets de réplica, e os distribui uniformemente ao longo dos nós do cluster. Cada vBucket possui um subconjunto de chaves (document ID).

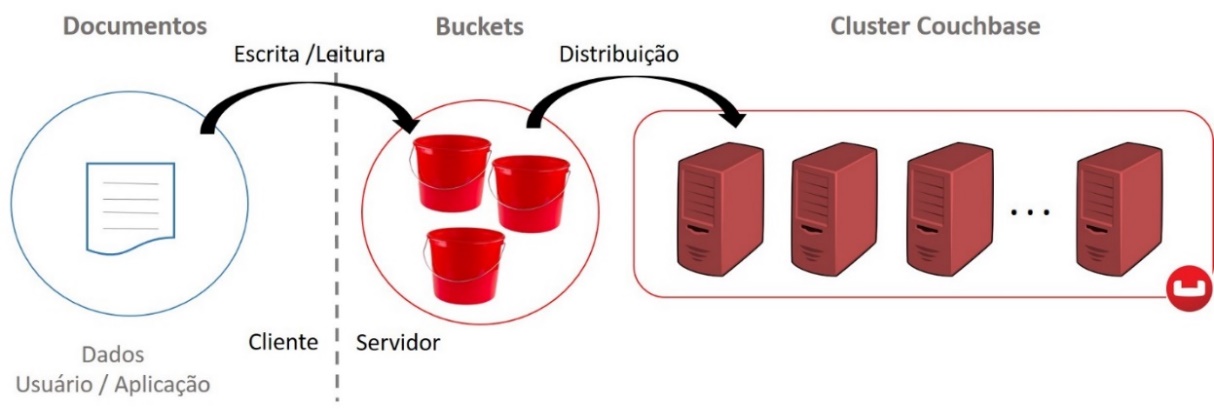


Figura 2 - Distribuição de Documentos no Couchbase

A localização de cada vBucket ao longo dos nós do cluster é armazenada no Cluster Map. O Cluster Map é copiado para cada nó do cluster e também para o Couchbase Client, o qual é o responsável pela distribuição dos documentos no cluster de forma automática e transparente.

O Couchbase client, também denominado de smartclient, determina o vBucket de cada documento aplicando uma função de hashing na chave (document ID). A partir do vBucket, o Coucbbase client determina o nó (servidor) utilizando o Cluster Map e então executa a gravação ou leitura do documento no respectivo nó (servidor). Essa distribuição é ilustrada na figura a seguir:

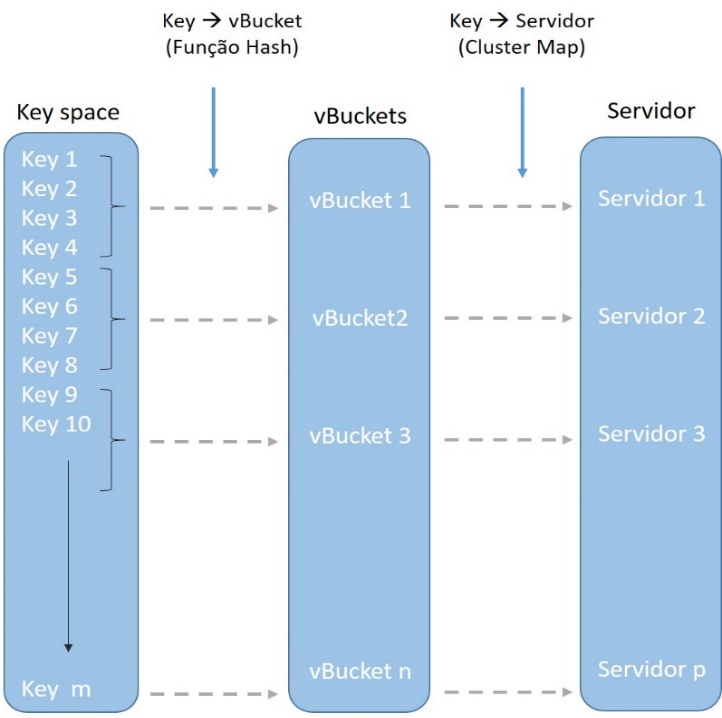


Figura 3- Mapeamento de vBuckets



### Alta Disponibilidade e Arquitetura de Replicação

A disponibilidade dos dados no Couchbase é garantida através de mecanismos de replicação interna ao cluster. Além disso também existe um mecanismo para replicar os dados em outro cluster.

No Couchbase existe apenas uma cópia ativa de cada documento. Através do mecanismo de replicação são criadas cópias (replicas) dos documentos ativos, as quais são distribuídas ao longo do cluster. Cada cópia é armazenada em um nó separado do cluster. Cada nó (servidor) replica uma fatia dos dados ativos (vBuckets) para múltiplos nós de forma que cada nó é uma cópia de qualquer outro nó do cluster. O Clouchbase suporta até 3 réplicas de cada documento, de forma que ao todo podem existir 4 cópias dos dados (1 ativo + 3 réplicas).

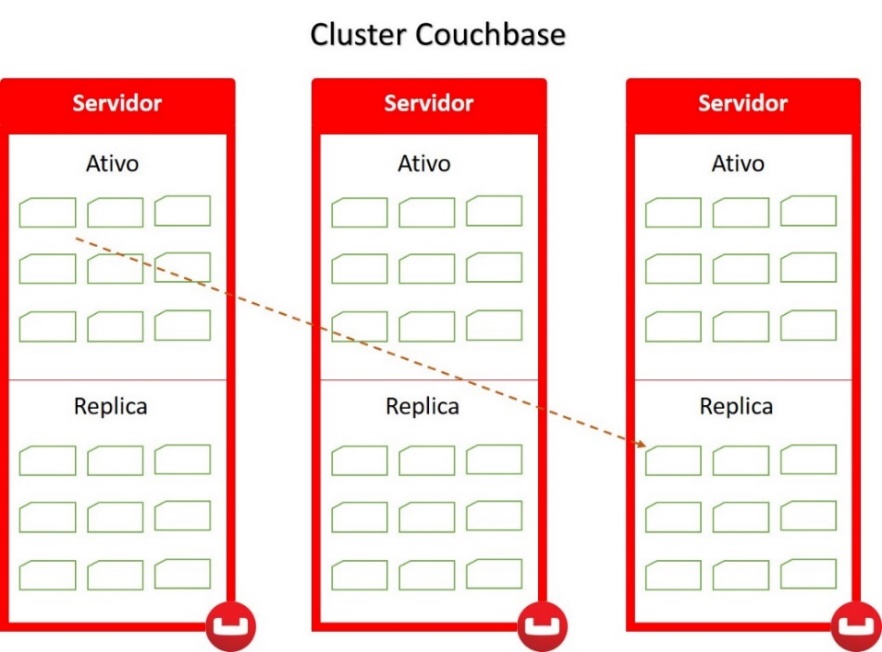


Figura 4 - Replicação interna no Couchbase

Em caso de falha de um nó do cluster, o Couchbase recupera os dados através da ativação das réplicas que estão nos outros nós do cluster. Esse processo é conhecido como failover, o qual pode ser automático ou manual.

Para assegurar a disponibilidade e a confiabilidade para aplicações corporativas, os dados ativos e suas réplicas devem ser armazenados em racks de servidores separados. O Couchbase implementa um mecanismo denominado de Rack-Zone Awareness, o qual permite um agrupamento lógico de servidores no cluster em racks ou zonas de disponibilidade distintas. Este tipo de replicação inteligente assegura que o dado esteja disponível apesar da possibilidade de falha de rack, ou switch ou energia.

É possível também implementar a replicação entre clusters através do XDCR (Cross Data Center Replication). Os clusters podem estar em diferentes datacenters e regiões geográficas distintas ou dentro do mesmo site. Através deste tipo de replicação é possível obter alta disponibilidade, distribuição de carga, performance ou recuperação de desastres. A replicação pode ser bidirecional ou unidirecional e configurada por bucket. Deste modo é possível configurar uma replicação bidirecional em um bucket e uma replicação unidirecional em outro bucket.

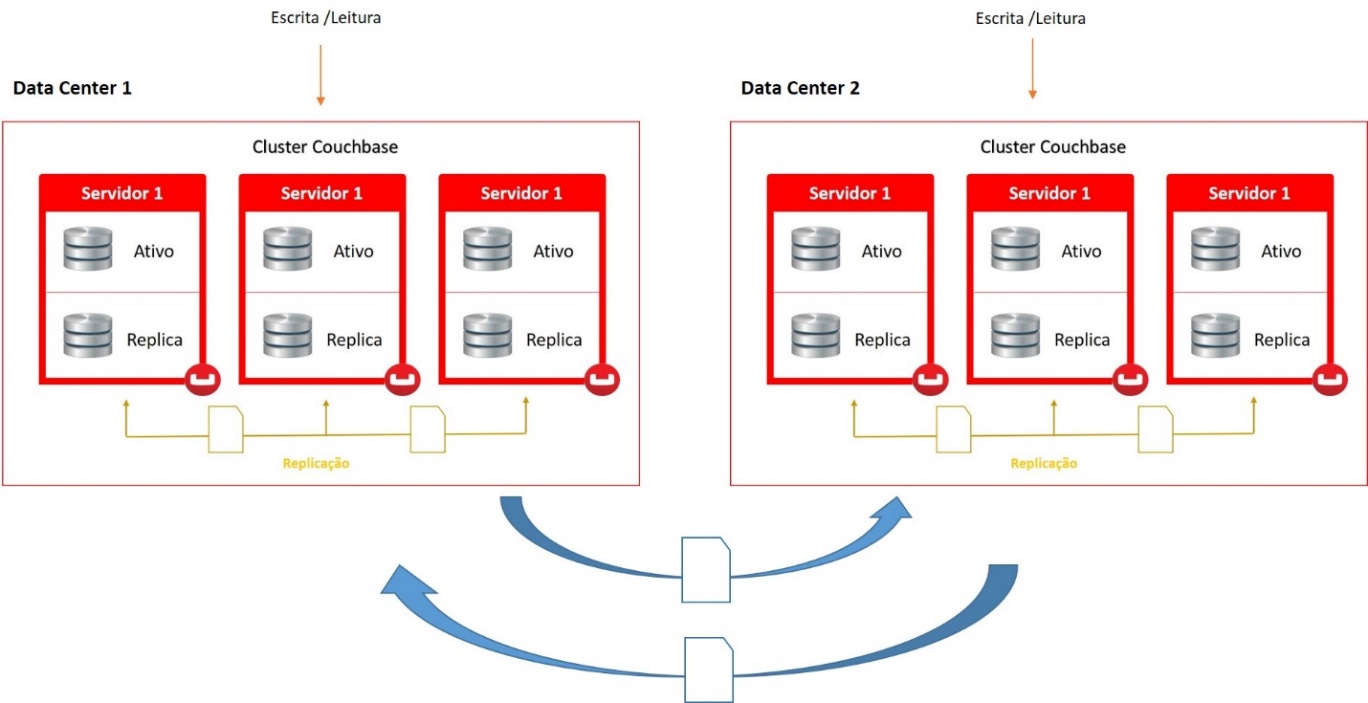


Figura 5- Replicação entre Clusters (XDCR)

Na linguagem do teorema CAP de Eric Brewer, o Couchbase é um sistema do tipo CP, o que significa que ele fornece consistência e tolerância a partição. No entanto, o Couchbase Server pode ser configurado como um sistema de AP, disponibilidade e tolerância a partição, com vários clusters usando XDCR (Cross Data Center Replication).

### Segurança

O Couchbase oferece alguns mecanismos de segurança que ajudam a proteger os dados contra ameaças e violações.

#### Autenticação:

O Couchbase verifica a identidade de administradores e aplicações usando os métodos de autenticação SASL e não SASL, incluindo mecanismos de autenticação do tipo desafio-resposta (CRAM) baseado no HMAC-MD5. Ele também suporta autenticação LDAP para administradores que se conectam utilizando o console web de administração.

A autenticação de aplicações é configurada no nível de bucket O controle de acesso é configurado através da console de administração Web durante a criação do bucket ou posteriormente através da edição das suas configurações

#### Encriptação

O Couchbase utiliza encriptação para proteger os dados do sistema. Ele provê mecanismos de segurança para os dados armazenados no disco e para o trafego de dados entre aplicações e clusters.

• **Trafego de Dados**: O Couchbase suporta trafego SSL fim-a-fim entre aplicações e cluster e entre clusters (XDCR). Esta encriptação cobre tanto o trafego de pacotes de dados como administrativos.

• **Dados armazenados**: O Couchbase trabalha com encriptação de disco baseada em LUCKS no Linux, criptografia de unidade de disco Bitlocker no Windows Server 2008 e 2012, e com a plataforma de segurança de dados da Vormetric, a qual faz encriptação no nível de disco, arquivo e aplicação.

#### Auditoria

A auditoria capacita usuários autorizados a monitorar as ações executadas pelos administradores no Cluster Couchbase. Apenas administradores com privilégio completo podem configurar a auditoria utilizando o console Web. A Tabela 2 apresenta os eventos passíveis de auditoria.

|  |  |  |
| --- | --- | --- |
| **Eventos Administrativos Auditáveis** | | |
| Tentativas de Login de administradores | Modificações nas configurações de auditoria | Habilitar ou desabilitar a auditoria |
| Adicionar um nó ao cluster | Remover um nó do cluster | Failover de um nó |
| Rebalanceamento do cluster | Shutdown/Startup do sistema iniciado por um administrador | Criação de bucket |
| Deleção de bucket | Flush do bucket | Modificação das configurações de um bucket |
| Adicionar administrador com permissão de leitura | Remover administrador com permissão de leitura | Adicionar um administrador |
| Remover um administrador | Configurar a referência para cluster remoto | Deletar a referência para um cluster remoto |
| Modificar a XDCR | Criar/deletar um profile XDCR | Parar ou reiniciar o stream XDCR |
| Modificar as regras de filtro de XDCR | Adicionar/remover um nó de query | Adicionar/remover um nó de indexação |
| Criar um grupo de servidores | Adicionar um nó para o grupo de servidores | Remover um nó de um grupo de servidores |
| Deletar o grupo de servidores | Alteração de senha administrativa/redefinição do backup |  |

Tabela 2 - Eventos para Auditoria

## Instalação

O Couchbase Server vem em duas edições diferentes: Enterprise Edition e Community Edition.

A instalação seleciona a plataforma e a edição de software correta para suas necessidades, faz o download do software e executa o instalador (<http://www.couchbase.com/nosql-databases/downloads>).

Para instalar Couchbase Servidor, é preciso levantar os seguintes pontos:

* Requisitos de hardware,
* Plataformas suportadas,
* Configuração de rede,
* Considerações de segurança e
* Considerações de implantação de produção.

Os requisitos mínimos para instalação são:

* Dual-core x86 CPU rodando em 2GHz.
* 4GB RAM (física).
* Um dispositivo de armazenamento baseado em blocos (disco rígido, SSD , EBS, iSCSI)
* Sistemas de arquivos de rede, tais como CIFS e NFS não são suportadas.

Os requisitos recomendados para instalação são:

* CPU Quad core 64-bit x86 CPU rodando em 3GHz;
* CPU de seis cores quando for utilizar Cross Datacenter Replication (XDCR) e Views.
* 16GB RAM (física).
* Um dispositivo de armazenamento baseado em blocos (disco rígido, SSD , EBS, iSCSI)
* Sistemas de arquivos de rede, tais como CIFS e NFS não são suportadas.

O Couchbase Server é suportado em vários sistemas operacionais populares e ambientes virtuais, sua instalação é fácil em qualquer que seja a plataforma. Para um maior detalhamento sobre a instalação e ou desinstalação, consulte a sessão de anexos deste trabalho.

## Interação

O acesso aos dados armazenados no Couchbase Server pode ser realizado através das seguintes formas:

* Acesso chave-valor
* Acesso através da linguagem N1QL

### Acesso Chave-Valor

Os documentos podem ser criados e recuperados usando o document\_id (chave). A recuperação de um documento através do seu ID (chave) é extremamente rápida.

As operações podem ser executadas através da ferramenta de comando de linha ou através de uma aplicação utilizando os SDK’s disponíveis em diversas linguagens de programação. Na Tabela 4 apresentamos a sintaxe para as operações CRUD na ferramenta de comando de linha CBC.

### Acesso através do N1QL

O N1QL (le-se “nickel”) provê uma linguagem de consulta semelhante ao SQL. Através do N1QL é possível realizar consultas baseadas no conteúdo (atributos) dos documentos JSON armazenados no Couchbase.

As operações CRUD podem ser executadas através da ferramenta de comando de linha CBQ ou através de uma aplicação utilizando os SDK’s disponíveis em diversas linguagens de programação. Na Tabela 4 apresentamos a sintaxe para as operações CRUD na ferramenta de comando de linha CBQ.

| **ACESSO** | **OPERAÇÃO** | **Comando de Linha** |
| --- | --- | --- |
| CHAVE/VALOR | INSERT | $ cbc-create document\_id -V '{"json":"value"}' $ cbc create --mode insert document\_id < arquivo.json |
| UPDATE | $ cbc create --mode replace document\_id < arquivo.json |
| READ | cbc cat document\_id |
| DELETE | cbc rm document\_id |
| N1QL  (DOCUMENTO) | INSERT | cbq> INSERT INTO product (KEY, VALUE) VALUES ("odwalla-juice1",   { "productId": "odwalla-juice1", "unitPrice": 4.40, "type": "product", "color":"green"}) RETURNING \*; |
| UPDATE | cbq> UPDATE product USE KEYS "odwalla-juice1” SET type = "product-juice" RETURNING product.type ; |
| UPSERT | cbq> UPSERT INTO product (KEY, VALUE) VALUES ("odwalla-juice1", { "productId": "odwalla-juice1",   "unitPrice": 5.40, "type": "product", "color":"red"}) RETURNING \* ; |
| READ | cbq> SELECT name, brewery\_id from `beer-sample` WHERE brewery\_id IS NOT MISSING LIMIT 2; |
| DELETE | cbq> DELETE FROM product p WHERE p.unitPrice = 5.25 RETURNING p.productId |
| INDICE PRIMARIO | cbq> create primary index on `beer-sample`; |
| INDICE SECUNDÁRIO | cbq> CREATE INDEX `beer-sample-type-index` ON `beer-sample`(type) USING GSI; |
| DROP INDICE | cbq> DROP PRIMARY INDEX ON `beer-sample` USING GSI; |

Tabela 4 - Exemplos de Interação com o Couchbase

### Otimização de Consultas

A forma de consulta mais eficiente no CouchBase é o acesso por chave/valor. As consultas baseadas no conteúdo dos documentos (N1QL) podem ser otimizadas através de indexação. O Couchbase provê três tipos de indexação: View MapReduce, View Espacial (Spatial View) e Global Secondary Index (GSI).

#### MapReduce View

As views permitem a extração de atributos específicos, informações dos documentos e a criação de um índice das informações selecionadas. Elas são definidas através de funções Map() e Reduce() escritas em Java Script.

As views são construídas de forma incremental e reindexadas apenas para inclusão dos documentos alterados após o último “index update”. O servidor Couchbase faz o pré-processamento e persiste os resultados da view antes de retornar a resposta para o cliente.

As views Map Reduce podem ser consultadas através do N1QL e da View API.

A criação de views é feita através da console web de administração do Couchbase.

#### View Espacial (Spatial View)

O Couchbase utiliza as views espaciais para consultar informações geoespaciais.

Views espaciais suportam uma variedade de diferentes casos de uso, incluindo análises multidimensionais, dados geográficos e uma combinação de ambos. Quando são analisados dados multidimensionais, views espaciais podem operar em dados numéricos e mapear categorias para números. Essas views suportam o formato GeoJson e permitem consultas do tipo bounding-box em geometrias complexas que são representadas como pontos, segmentos de linhas ou polígonos.

A principal diferença entre as views mapreduce e as views espaciais é que ao invés de utilizar duas funções (Map e Reduce), as views espaciais contém uma única função chamada de função espacial. As views espaciais também retornam mais do que uma única chave por documento, possibilitando consultas em múltiplos atributos em uma única query.

As views espaciais podem ser consultadas através da View API.

#### Global Secondary Index (GSI)

Estes índices permitem que as aplicações realizem consultas rápidas com uma alta taxa de transferência. Diferentemente dos índices de view, os quais indexam apenas um subconjunto dos dados locais em cada Data Node, um índice secundário global (GSI) indexa todos os dados dos buckets em um único local e reside nos Index nodes.

Os índices do tipo GSI são criados a partir das declarações do tipo “Create Index“ em N1QL. Eles podem ser do tipo primário ou secundário.

O índice primário contém o conjunto completo de chaves do bucket. O índice primário é opcional, sendo requerido apenas para permitir a execução de consultas adhocs em um bucket no qual não existem índices secundários.

Os índices secundários contêm um conjunto filtrado ou total de chaves em um bucket.

Para realização de consultas N1QL em um bucket é necessário criar pelo menos um (1) índice.

Na versão 4.5 do Couchbase, ainda em desenvolvimento, será oferecida a funcionalidade de Full Text Search. Esta funcionalidade permitirá consultas otimizadas baseadas em linguagem natural. Atualmente esta funcionalidade é implementada através de uma integração com o Elastic Search.

## Tarefas Administrativas

A administração de Couchbase Server é fornecida através de um número de diferentes ferramentas e sistemas. As ferramentas de administração usadas ​​com Couchbase Server são:

1. **Couchbase Web Console**: fornece uma interface completa para configurar, gerenciar e monitorar sua instalação Couchbase Server.
2. **Linha de comando de interface (CLI)**: fornece informações e controle sobre seu servidor Couchbase e instalação de cluster. As ferramentas CLI pode fazer uso da API REST. Você pode usar o CLI em combinação com seus scripts e procedimentos de gestão para fornecer funcionalidade adicional, como failover automatizado, backups e outros procedimentos.
3. **API REST**: Além do Couchbase Console Web, o Couchbase Server incorpora uma interface de gestão exposta através do protocolo HTTP REST padrão. Esta interface REST pode ser chamada a partir de seus scripts de gestão e administração personalizada para apoiar as operações diferentes.

### CouchBase Web Console

O Couchbase Web Console é a principal ferramenta para gerenciar o ambiente Couchbase.

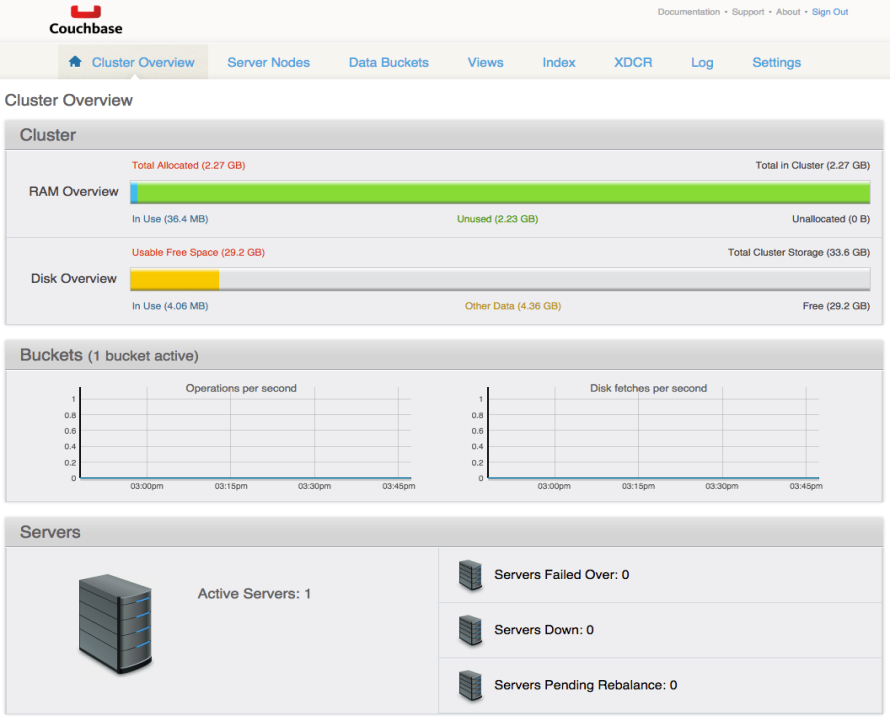


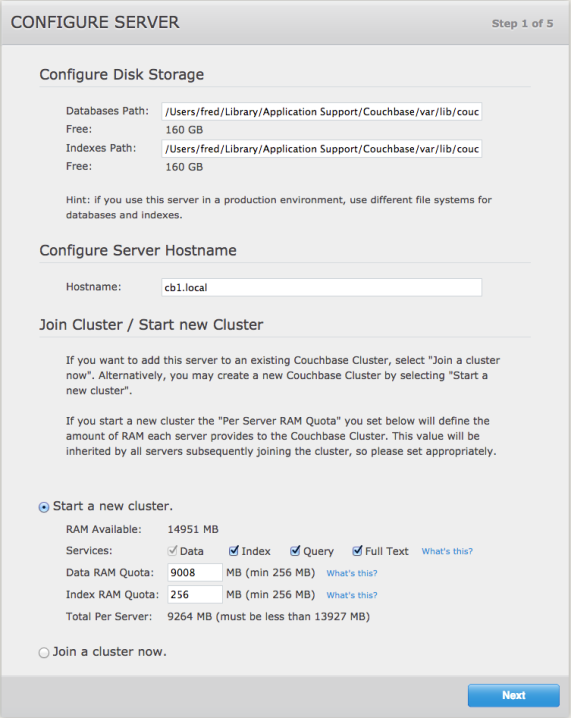
Figura 11 - CouchBase Web Console – Cluster Overview

A tela principal (vide Figura 11) apresenta as seguintes seções:

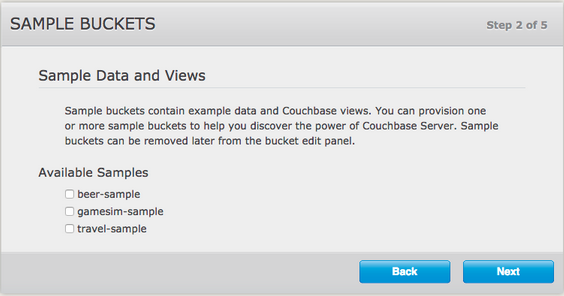
* **Seção Cluster:** fornece informações sobre o uso de RAM e disco para o cluster.
* **Seção Buckets:** fornece dois gráficos que mostram as operações por segundo e acesso a disco por segundo.
* **Seção Server:** indica as informações globais dos servidores no cluster.

### Cluster Setup

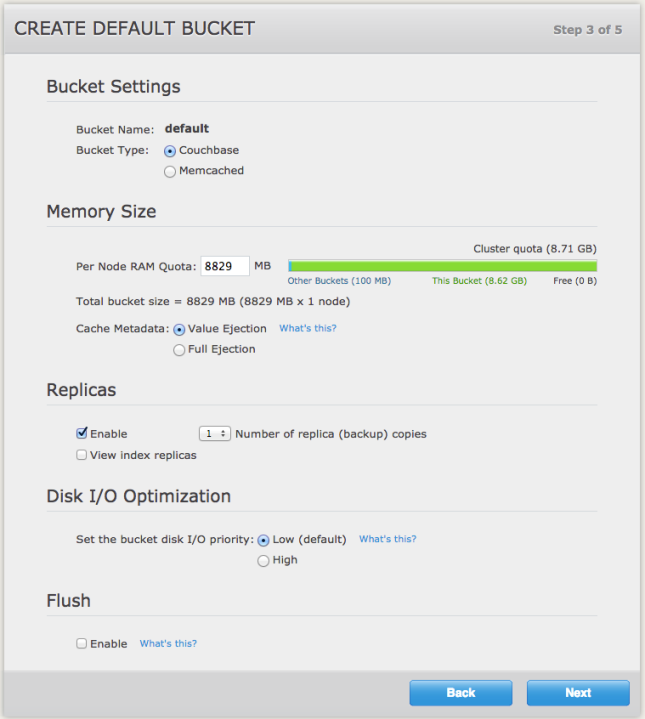
A criação de um cluster envolve a definição de locais dos arquivo e dos nomes dos hosts. O local padrão do Couchbase Web Console é http://< hostname>:8091. A configuração do cluster envolve 5 passos:



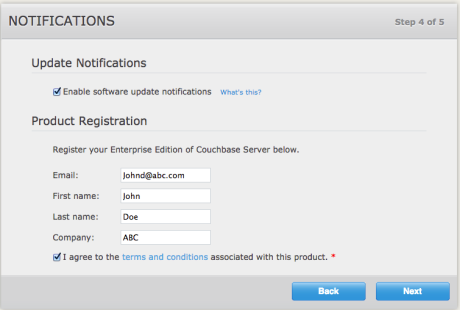
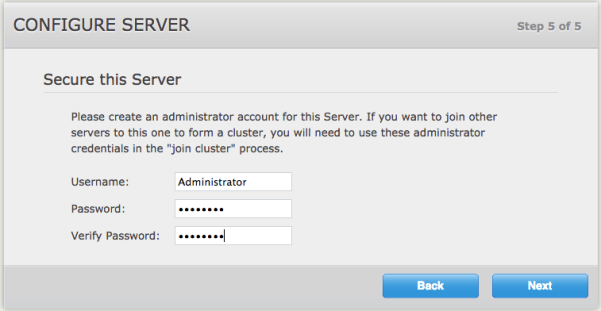
**Figura 15 – Cluster Setup – Parte 1.**



**Figura 16 – Cluster Setup – Parte 2.**



**Figura 17 – Cluster Setup – Parte 3.**

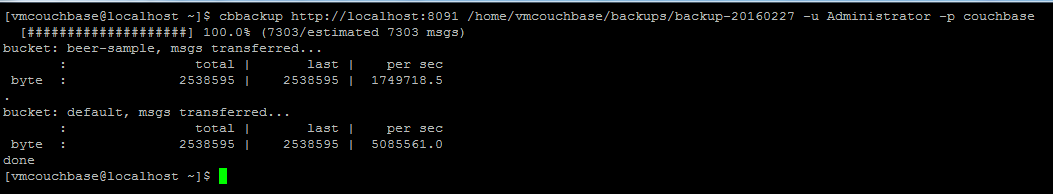
 

**Figura 18 – Cluster Setup – Parte 4 e 5.**

### Backup e Restore

O backup de todo o cluster deve ser feito periodicamente para minimizar a inconsistência de dados quando uma restauração é necessária. O backup de dados deve ser um processo regular do cluster para garantir que não ocorram perdas de informações no caso de uma falha grave de hardware ou instalação. Existem vários métodos para a realização de uma cópia de segurança, tais como:

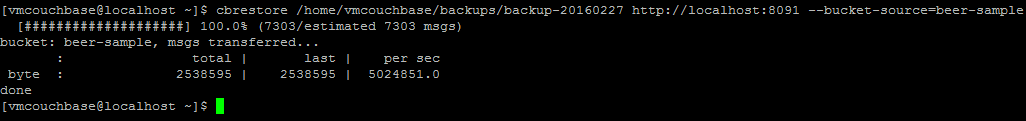
1. **Usando cbbackup**: O comando cbbackup permite fazer backup de um único nó, buckets individuais, ou todo o cluster em uma estrutura de backup flexível, que pode restaurar os dados para os mesmos ou diferentes, aglomerados e buckets. Todas as cópias de segurança podem ser realizadas num cluster vivo ou nó. O cbbackup é a ferramenta de backup mais flexível e recomendada.
2. **Usando cópias de arquivos**: Em execução ou em cluster off-line pode ser feito backup, copiando os arquivos em cada nó. Com este método, você só pode restaurar um cluster com a configuração idêntica.



**Figura 19 – Exemplo de backup.**

Ao restaurar um backup, é necessário selecionar a sequência de restauração adequada com base no tipo de restauração a ser executada. Os métodos disponíveis durante a restauração de um cluster dependem do método usado para fazer o backup do cluster.

A restauração dos dados de um bucket no qual foi feito backup usando o cbbackup é realizada através do comando cbrestore. A configuração do cluster não tem de coincidir com aquele usado quando os dados foram feitos por backup. O cbrestore pode ser usado para transferir informações para um novo cluster atualizado ou versão do cluster existente expandida em caso de recuperação de desastres.



**Figura 20 – Exemplo de restore do bucket beer-sample.**

# EVIDÊNCIAS

## Realização de Operações CRUD a ferramenta N1QL



Figura 12 - Criação de um novo documento

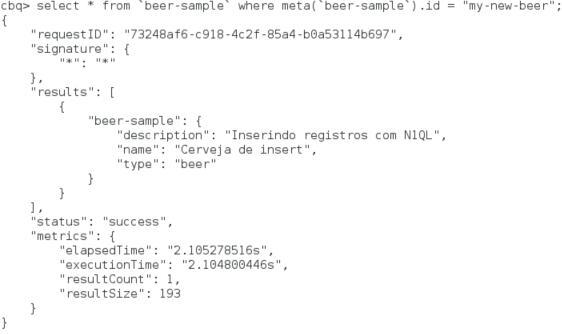


Figura 13 - Consulta de documento

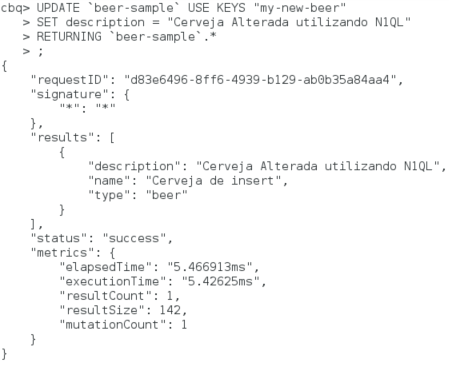


Figura 14 - Alteração de Documento



Figura 15 - Remoção de Documentos

# CONCLUSÕES

Ao estudar o Couchbase server pudemos notar que o mesmo é um banco de fácil interação, que entrega ferramentas que ajudam muito ao desenvolvedor e ao administrador do banco a construir e manter suas aplicações. Utiliza a linguagem N1QL(“Nickel”) que é bem próximo ao tão difundido SQL, o que diminui a curva de aprendizado das equipes ao lidar com a ferramenta.

O Couchbase é um banco veloz e confiável, uma vez que mantém boa parte de seus dados em memória, mas grava seus dados em disco de maneira assíncrona para manter a durabilidade desses dados, além de dar a possibilidade de configurar réplicas.

Facilmente escalável e tolerante a falhas de nó, possibilitando a inserção e remoção de nós em um cluster com o mínimo de configuração necessária, além de possibilitar o espalhamento dos cluster entre datacenters remotos através do protocolo XDCR.

Extensível, possibilitando integração com a ferramenta Elasticsearch, habilitando o full-text seaching de uma maneira fácil e descomplicada. Além disso, possui integração com Hadoop, Spark, Kafka e Talend.

Nos testes comparativos de stress vimos que ganha disparado de seu concorrente no quesito leitura e no quesito gravação se torna bem próximo, ao utilizar o engine MyISAM, que não dá suporte a transações e com isso ganha uma grande agilidade em operações de escrita.

Como ponto alto a se destacar apontamos a facilidade de criação e manutenção de ambientes utilizando Couchbase e sua agilidade para operações de leitura, o que o torna uma ótima escolha para ferramentas de reporting.

# REFERÊNCIAS BIBLIOGRÁFICAS

## Documentação On-Line e Sites de Referência

* <http://www.infoq.com/br/articles/State-of-NoSQL>
* <http://www.infoworld.com/article/2613970/nosql/nosql-showdown--mongodb-vs--couchbase.html>
* <http://www.couchbase.com/couchbase-vs-couchdb>
* <http://assist-software.net/blog/couchbase-vs-couchdb-vs-mongodb>
* <http://db-engines.com/en/system/CouchDB%3BCouchbase%3BMongoDB>
* <http://blog.totvs.com/totvs-e-couchbase-anunciam-parceria-para-suporte-de-big-data/>
* <http://adrianolisboa.com/o-que-e-o-couchbase-lite/>
* <http://dicasparasoftware.blogspot.com.br/2013/09/couchbase-relaxa-nosql-derriere-em.html>
* <http://www.guiaphp.com.br/couchbase/>
* <http://www.couchbase.com/nosql-databases/couchbase-server>
* <http://pythonhosted.org/couchbase/>
* <https://pypi.python.org/pypi/couchbase>

## Livros

POTSANGBAM, Henry. **Learning Couchbase**. Packt Publishing. Ebook: Novembro 2015, 248p

OSTROVSKY , David; RODENSKI , Yaniv. **Pro Couchbase Server**. Apress, Ebook: Agosto 2014, 348p.