Instituto Infnet - Pós-Graduação MIT em Big Data

Bruno Brandão
Leandro Coelho
Ortiz Araujo

Projeto de Bloco 1
NuoDB

Rio de Janeiro
06 de Outubro de 2016

Index

Introdução

Posicionando-se no mercado como uma companhia de banco de dados distribuídos, a NuoDB vende soluções de banco de dados relacionais em nuvem.

Compatível com SQL, o banco NuoDB vem sendo chamado de "NewSQL". Tem uma arquitetura distribuída voltada para a nuvem que, ao adicionar um novo servidor para "escalar" o banco de dados, tem um ganho no aumento de performance sem a necessidade de aplicação de sharding (particionamento de grandes arquivos em pedaços menores).

As tarefas do BD são distribuídas entre vários processadores para evitar gargalos de dados. Ele usa mensagens P2P para tarefas de roteamento de nós, e é compatível com ACID.

O banco de dados usa uma abordagem em camadas - que compreende várias camadas redundantes de *Transaction Engines* (TE) e *Storage Managers* (SM)." Esta abordagem permite adaptar os dados de maneira previsível na nuvem. O NuoDB consiste em diversas e redundantes *TE's* e *SM's*, que podem rodar na mesma plataforma.

O sistema foi projetado para se alinhar com as 12 regras (criadas por Edgar F. Codd, IBM) para bancos de dados relacionais. Ele adiciona a capacidade de executar em qualquer lugar; escalabilidade elástica; disponibilidade ininterrupta; um único banco de dados, lógica; e distribuído de segurança. O sistema pode processar mais de 1 milhão de transações por segundo. Ele está disponível em uma versão gratuita limitada, uma versão desenvolvedor livre, uma versão profissional paga e uma versão empresarial.

Aplicações

O NuoDB deve ser utilizado principalmente em projetos que necessitem de uma combinação entre o conceito ACID, alta disponibilidade, geo-distribuição e que possam se beneficiar da linguagem SQL, que é amplamente conhecida no mercado. Muitas indústrias vêm procurando essa combinação para atender suas necessidades de negócio, algumas delas são:

- Games;
- Telecomunicações;
- Serviços Financeiros.

Um bom exemplo é a indústria de serviços financeiros, composta por bancos, operadoras de cartão de crédito, grupos de investimento, entre outros, que buscam cada vez mais inovação para atender a crescente demanda de seus clientes, sendo eles, por muitas vezes, globais. Portanto essa indústria tem o desafio de atender cada vez mais clientes, realizando mais transações, a partir de mais lugares, exigindo mais agilidade nas transações e ainda deve estar em conformidade com as leis e regulamentações de cada lugar.

Assim, podemos dizer que existem 3 importantes características para alinhar a solução a esse tipo de negócio:

- Operações globais;
- Consistência transacional;
- Controle sobre a localização dos dados distribuídos.

Os bancos de dados NoSQL, conseguem realizar o serviço de distribuição global e até mesmo de controle sobre a localização dos dados, mas sacrificam a consistência transacional em troca de mais poder de geo-distribuição e escalabilidade. Migrar para esse tipo de banco de dados ainda apresenta grande custo e risco, pois é necessário repensar a arquitetura e a implementação das transações em código de aplicação, quando já se possui esse conhecimento em SQL.

O NuoDB pode atender as necessidades dessa indústria, pois combina a linguagem SQL, é totalmente distribuído, com controle sobre a localização dos dados e está em conformidade com o conceito ACID (o que garante a consistência transacional), além de ser horizontalmente escalável.

Arquitetura

Se baseando em replicação sob demanda, o NuoDB apresenta uma arquitetura distribuída utilizando de uma tecnologia chamada *durable distributed cache*, que consiste em um sistema *peer-to-peer* de memória central que assegura a lógica ACID, resiliência e desempenho *scale-out* em vários *data centers*.

Estrutura em níveis

Visando a escalabilidade do sistema, o NuoDB se divide em níveis, podendo assim separar a parte transacional (onde a velocidade é essencial) da parte de armazenamento (durabilidade).

- **Processamento**

  Através dos *Transaction Engines* (TEs), que são os motores de transação, é possível garantir o Atomicidade, Consistência e Isolamento da sigla ACID.

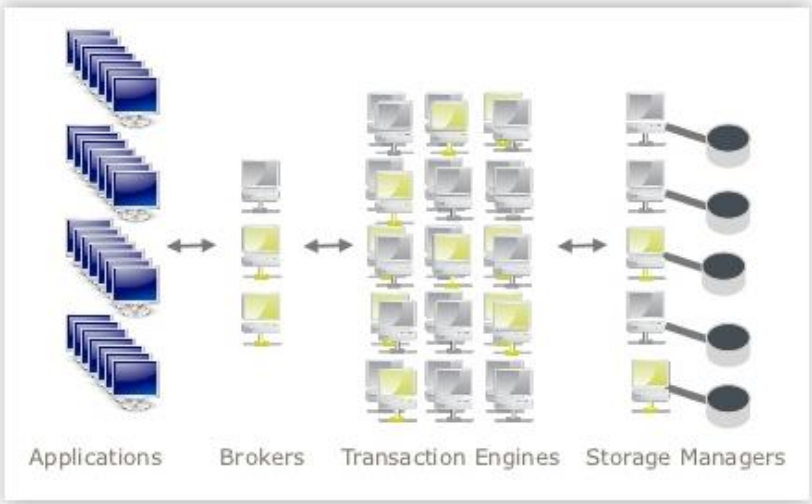
- **Armazenamento**

  Utilizando os *Storage Managers* (SMs) o banco faz uma cópia de toda a base de dados, podendo servir assim as engines de transação (TE). O nome utilizado para essa cópia é "arquivo".

- **Administrativo**

  Responsável por garantir escalabilidade horizontal e o balanceamento de carga. Cada partição (nó) conta com um agente que juntos, formam o que se chama de rede ponto-a-ponto (Domínio).

Cada Domínio conta com agentes chamados de *Brokers*, que são responsáveis por iniciar, parar ou migrar uma base de dados, interagir (iniciar, terminar e gerenciar logs) dos TEs e dos SMs e cuidam também da monitoria do sistema.



Átomos

São objetos auto-coordenados que podem representar qualquer tipo de informação como dados, metadados, esquemas, índices entre outros. Esses objetos facilitam a comunicação interna e a implementação do *cache*. O nível de processamento transacional é responsável por fazer o mapeamento entre o conteúdo SQL e os devidos Átomos.

Além do conteúdo da base de dados, existe o "Átomo de Catálogo", que é utilizado para buscar os demais átomos entre os processos correntes. Na criação da base de dados, e da ativação do primeiro TE, um objeto é diretamente colocado no seu *cache*, o Catálogo Mestre. O Catálogo Mestre indica onde todos os elementos da base de dados podem ser descobertos. Utilizando os catálogos as TE podem descobrir onde um átomo está disponível.

Para buscar um átomo, as TEs tentam primeiramente no *cache* do próprio TE, não encontrando busca no cache de outros TEs e, por final, se não encontrar solicita ao SM para buscar em espaço físico.

Essa abstração de todos os dados em Átomos serve para simplificar o modelo de durabilidade do NuoDB. Todo os acessos na arquitetura são feitos através dos átomos, e todos eles são guardados como pares de chave-valor.

Controle de Concorrência

Como mencionado anteriormente, os Átomos ficam responsáveis por toda a parte de durabilidade do sistema, mas isso não garante as propriedades ACID. Visando o gerenciamento de conflitos e consistência dos dados, o NuoDB utiliza um protocolo chamado MVCC (Multi-Version Concurrency Control) que é responsável por realizar o versionamento dos dados (atualizações/remoções do banco).

Os motores de transação (TEs) guardam diferentes versões de um mesmo dado. Após uma atualização, a nova versão fica pendente até que a transação associada a ela tenha sido realizada com sucesso.

Além do MVCC o NuoDB utiliza um modelo de visibilidade da base de dados, chamado de Isolamento Instantâneo (*Snapshot Isolation)*. Esse modelo permite obter uma visão constante da base de dados no exato momento em que uma transação se inicia.

Mas, visando garantir a persistência dos dados para múltiplas transações simultâneas, o NuoDB disponibiliza um motor de transação (TE) para intermediar e evitar conflitos. Esse intermediador se denomina "presidente" e tem como principal função definir o tempo em que cada transação vai ser realizada.

Tarefas Administrativas

Backup & Restore

Em versões anteriores do produto, fazer um backup consistente de banco em execução era bastante complexo. O administrador precisava escolher entre:

- Criar um snapshot "point-in-time" do seu arquivo usando um sistema de arquivos avançado como ZFS ou Btrfs
- Criar um snapshot "point-in-time" do servidor usando suas ferramentas de virtualização
- Dedicar um gerenciador de armazenamento (SM) para gerar um arquivo estático.

Cada uma das possibilidades acima tem seus desafios. Os dois primeiros itens exigem que o administrador disponha de uma infra-estrutura específica, quer se trate de uma máquina virtual ou um sistema de arquivos.

O terceiro item além de precisar de bastante recurso disponível, também requer alguma lógica adicional para garantir que o arquivo seja totalmente sincronizado. Uma vez que sincronizado, é preciso reiniciar o SM.

Visando sanar esse problema, o NuoDB criou um sistema de backup conhecido como HotCopy.

HotCopy usa um dos gerentes de armazenamento existentes (SM) para criar uma cópia transacional consistente do arquivo de um banco de dados sem interrupção do serviço. O banco de dados continua em execução e as transações continuam sendo realizadas, mesmo em um banco de dados com um único host.

Segurança

O NuoDB possui uma lógica hierárquica no que se diz respeito à permissão de acesso aoBanco de dados:

- **Administrador do Domínio**

  Responsável pela implementação do banco, tendo total acesso à criação de bancos, DBAs, usuários e schemas.

- **DBAS**

  Ao criar um banco de dados NuoDB, é preciso especificar um DBA (Database Administrator), incluindo nome de usuário e senha. O DBA é o superusuário do banco de dados e tem privilégios de SELECT (não UPDATE) e INSERT para todas as tabelas do banco. O DBA pode criar, alterar e deletar usuários e / ou regras e conceder e revogar privilégios de usuários.

- **Usuários**

  Possui permissões restritas de acordo com as configurações realizadas pelo DBA. Geralmente com acesso a tabelas específicas apenas com a permissão de SELECT.

  No quesito autenticação de usuários, é possível trabalhar em dois formatos:

- **Usuários Internos**: Onde o usuário não passa por processo de autenticação rebuscado, utilizando apenas usuário e senha definidos previamente

- **LDAP (Lightweight Directory Access Protocol):** Tem o objetivo de integrar com contas de sistemas já existentes, minimizando o trabalho de manutenção dos usuários dentro do Banco.

Glossário

ACID

Sigla para as propriedades de transação do banco de dados de atomicidade, consistência, isolamento e durabilidade.

Admin Center

A instalação do NuoDB inclui esta ferramenta baseada em navegador, que fornece acesso a ferramentas, demos, documentação e amostras.

Admin Center menu

Um menu do tipo Google, acessado no canto superior direito de qualquer ferramenta baseada em navegador NuoDB, que permite selecionar as ferramentas, demos, documentação e amostras.

Agent

Parte da camada de gerenciamento NuoDB. Em um host NuoDB, um agente é um processo que inicia e pára os motores de transação (TEs) e gerentes de armazenamento (SMs), fornece o estado e informações estatísticas sobre bancos de dados NuoDB e processos que estão em execução.

Album

Um gerente de armazenamento de snapshot usa álbuns para gerir de forma eficiente. Um album refere-se a um conjunto de snapshots tiradas durante um período de tempo.

Archive Directory

A pasta no disco para do qual o gestor de armazenamento (SM) escreve e lê os dados.

Atom

A estrutura do objeto interno que representa todos os dados em um banco de dados NuoDB. Os átomos são objetos auto-coordenados que representam tipos específicos de informação (como dados, índices ou esquemas).

Atomicity

Uma propriedade de transação que requer que, ou todo o processamento feito em uma transação é concluída com êxito ou nada do processamento na transação é feita. Em outras palavras, se uma transação é incapaz de completar toda a sua transformação, em seguida, o estado do banco de dados é retornado ao que era antes da operação começar.

Automation Console

É uma ferramenta de administração de bancos de dados do NuoDB. É baseada em navegador que permite criar bases de dados, gerenciar sua operação, monitorar indicadores de desempenho e visualizar outras estatísticas.

Broker

Um broker é um agente que permite o gerenciamento de acesso do cliente, requisições de rotas de conexão SQL ao TEs em qualquer lugar do domínio, armazena a configuração de domínio durável, executa o aplicador, que automatiza o ciclo de vida de processos de scale-out com base no modelo de cada banco de dados, monitora a atividade e dispara alarmes para todos os hosts, fornece segurança (nunca retornar um resultado incorreto) em caso de falha, retransmite informações sobre seus agentes ligado a todos os outros brokers. Cada broker tem um quadro completo de todo o domínio, incluindo todos os hosts e todos os bancos de dados e seus processos. Um domínio é funcional apenas com pelo menos um broker.

Consistency

Uma propriedade de transação do banco de dados que garante que o banco de dados está em um estado válido após uma transação. Por exemplo, considere uma transação que move o dinheiro da conta A para a conta B. O total de ambas as contas devem ser os mesmos antes da operação e depois da transação a.

Continuous availability

A capacidade de um banco de dados NuoDB para continuar a trabalhar, se um ou mais hosts forem desligados. Se processos e transações ativas falharem são abortados, e erros são relatados aos clientes. As transações ativas em hosts não são interrompidas.

Continuous data protection

Um gerente de armazenamento de snapshots salva um snapshot para cada transação confirmada e armazena esses snapshots em seu arquivo de instantâneo. Salvar e armazenar snapshots de banco de dados fornecendo proteção contínua de dados (CDP), porque cada snapshot fornece uma cópia virtual de todo o banco de dados. Um snapshot mostra o que o banco de dados parece nesse momento no tempo.

Database

Uma instância de um banco de dados NuoDB é composta por um número variável de processos em execução em um número variável de hosts, motores de Transação e gerentes de armazenamento que fazem parte do banco de dados. Um banco de dados pode ser servido por qualquer número de motores de transação e gerentes de armazenamento. Cada motor de transação e cada gestor de armazenamento serve um banco de dados.

Domínio (NuoDB)

Uma coleção de máquinas ou instâncias de nuvem que foram provisionados para trabalhar em conjunto para apoiar bancos de dados NuoDB. Um domínio pode conter vários bancos de dados e cada banco de dados podem ser executados em um ou mais hosts.

Cada máquina física no domínio executa um broker ou um agente. Brokers e agentes são ligados uns aos outros e são referidos como pares. Um domínio é um conjunto de pares.

Domain administrator

Um usuário responsável por:

- A provisão de um conjunto de hosts, onde NuoDB é executado
- Instalando NuoDB nesses hosts
- Usando o Console de automatização para criar bancos de dados com base em modelos internos
- Configurar brokers e agentes para formar uma rede onde o NuoDB está instalado
- Monitorar a saúde geral do domínio

Domain password

Esta é a senha especificada no momento da instalação NuoDB.

Durability

Uma propriedade transação banco de dados que garante que as alterações feitas durante uma transação são persistentes e podem ser recriados em caso de uma falha do sistema.

Enforcer

Processo que é executado em cada broker no domínio. O aplicador compara o estado do domínio contra os requisitos para a base (s) ou contra o molde (s) utilizado para criar a base de dados (s). Somente o corretor líder, inicia processos adicionais, conforme necessário.

Geo-distribuídos

Um modelo NuoDB que podem ser selecionados ao criar um banco de dados. O comportamento padrão é que começa um gerenciador de armazenamento (SM) e como muitos motores de transação (TEs) como possível em todas as regiões. Cada host vai ter no máximo um SM e um TE de modo que este modelo pode ser usado com qualquer número de hosts. Um processo é reiniciado em um novo host, se um estiver offline e outro estiver disponível.

Geo-distributed

A capacidade de atravessar várias localizações geográficas ou regiões, em centros de dados em todo o mundo.

Isolamento

Uma propriedade banco de dados que garante que mesmo em um sistema concorrente, as transações se comportam como se fossem executadas em série.

Journaling

Mantém o controle de alterações em um banco de dados de modo a que em caso de falha, o banco de dados pode ser mantido em um estado consistente.

Leader

O leader broker é eleito pelo Algoritmo de consenso Raft para determinar a broker majoritário. O leader broker executa as verificações vitalidade para o domínio e determina se um outro broker não está mais respondendo. Todas as operações de seguros exigem reconhecimentos de um broker majoritário.

Load balancing

O ato de distribuir a carga de trabalho entre vários processos. Em NuoDB, o broker gere o balanceamento de carga de conexões SQL para motores de transação (TEs).

Management tier

A camada de gerenciamento na arquitetura de NuoDB que consiste nos agentes e brokers , que são referidos como pares.

Minimally Redundant

O comportamento padrão é começar com dois gerentes de armazenamento (SM) e dois motores de transação (SE). Cada host vai ter no máximo um SM e um TE de modo que este modelo requer um mínimo de dois, mas pode utilizar até quatro hosts. O modelo minimamente redundante não garante disponibilidade contínua.

Multi Version Concurrency Control (MVCC)

É uma forma de controle de concorrência que utiliza várias versões de registros para fornecer uma visão consistente dos dados para cada transação e evitar transações simultâneas de substituir alterações um do outro. Sob MVCC, os leitores não bloqueiam escritores, e vice-versa.

Multi-Tenancy

Capacidade de um domínio NuoDB para hospedar mais de um banco de dados.

No Knobs Administration

"Não há Botões" é um termo que implica, mesmo para um sistema complicado, que existem suficientes propriedades padrão definidos de tal forma que o sistema irá funcionar "fora da caixa".

Node

Processos do gerenciamento de armazenamento e de motores de transação são considerados nós. Um banco de dados é um conjunto de nós.

NuoDB Check

Uma ferramenta de linha de comando que pode ser usado para validar os dados em disco e no diretório do arquivo. Ele também pode ser usado para limpar os arquivos adicionais no disco.

NuoDB Durable Cache Distribuído (DDC)

Arquitetura dimensionável de forma dinâmica, não tem nenhum ponto único de falha e proporciona propriedades de transação ACID.

NuoDB Loader

Uma ferramenta de linha de comando para importar e exportar dados.

NuoDB Manager

Uma ferramenta de linha de comando que serve funções semelhantes como o Console Automation NuoDB. No Gerenciador de NuoDB, você pode gerenciar o ciclo de vida dos processos em seu domínio NuoDB, e você pode monitorar, analisar e controlar o domínio. NuoDB Manager permite que você especifique os comandos de forma interativa ou não interativa.

NuoDB Migrator

Uma linha de comando Java que ajuda os administradores de banco de dados migrar esquemas e dados existentes para um banco de dados NuoDB.

NuoDB services

São os serviços que são necessários estar em execução, a fim de tirar o máximo proveito de todas as ferramentas NuoDB.

NuoDB SQL

A variação da linguagem SQL que NuoDB suporta. O utilitário sql fornece uma interface de linha de comando para a execução de SQL com NuoDB.

NuoDB system tray application

A ferramenta gráfica rodando em Mac OS X e Windows usado para iniciar a página NuoDB Admin Center e o Console Automation NuoDB e para verificar atualizações NuoDB.

Peer

Brokers e agentes são considerados pares. Um domínio é um conjunto de pares. Em um domínio, nunca há mais do que um ponto por host.

Pseudo system table

Pseudo tabelas que conter dados gerados pelo NuoDB cada vez que você consulta. Estes dados não existem nas tabelas, mas são apresentados como uma tabela SQL. Por exemplo, SYSTEM.QUERYSTATS e SYSTEM.NODES são pseudo tabelas do sistema.

Quorum

Em um domínio, o quorum é o número mínimo de brokers que deve confirmar a mudança para a configuração de domínio durável para que a mudança seja feita. Normalmente, esta é uma maioria simples. Muitas operações na camada de gerenciamento requer o reconhecimento da maioria dos corretores no domínio, a fim de garantir a segurança (nunca retornar um resultado incorreto) em caso de falha.

Redundancy

Uma propriedade sistema que significa duplicado. No caso de um banco de dados NuoDB, este refere-se a vários hosts provisionados no domínio para lidar com o caso de falha em qualquer um host do domínio.

Safety

Nunca retornar um resultado incorreto.

Scale-out Performance

Esta é a melhoria do desempenho alcançado pelo provisionamento automaticamente de novos hosts no domínio NuoDB.

SQL Explorer

Uma ferramenta baseada em navegador empacotado com NuoDB que permite acessar seu banco de dados e executar instruções SQL em seu banco de dados. Você pode iniciar SQL Explorer a partir do menu de NuoDB Admin Center.

Storage group

Um grupo de armazenamento representa uma unidade de armazenamento lógico que pode ser servida por múltiplos gerenciadores de armazenamento em um banco de dados.

Storage tier

Esta é a camada de arquitetura onde os dados são armazenados e gerenciados através de objetos chamados átomos. Gerentes de armazenamento são parte da camada de armazenamento em arquitetura NuoDB.

System tables

Em cada banco de dados NuoDB, tabelas do sistema são tabelas persistentes que são armazenados no esquema SYSTEM. Tabelas do sistema contém dados importantes compartilhados entre todos os processos no domínio NuoDB e disponível para todas as conexões com o banco de dados. Isso inclui informações sobre objetos de banco de dados, tais como tabelas, índices, esquemas, triggers, usuários, funções e privilégios.

Table partition

É um atributo que pode ser associado com uma tabela SQL. A partição contém um subconjunto das versões de linha que são armazenados em servidores por um grupo de armazenamento.

Templates

Um modelo NuoDB especifica um Acordo de Nível de Serviço (SLA) para um banco de dados.

Temporary table

Um objeto de banco de dados que reside na memória somente para a conexão atual. Você pode definir uma tabela temporária com uma declaração CREATE TABLE e você pode operá-lo com DML. A tabela temporária pode ser descartada durante uma sessão.

Instalação

Escolhemos a versão Community visando criar um ambiente que refletisse ao máximo uma instalação do NuoDB Enterprise, que apesar de ter mais funções que a versão Community, difere muito de uma instalação do Tour do NuoDB, que utiliza o Docker, não recomendado pela própria empresa para ambientes de produção. Nossa escolha afetou, no entanto, a possibilidade de escalar mais de um nó do NuoDB por questões de licença, que tentamos conseguir junto ao suporte do NuoDB, porém sem sucesso.

Requisitos Mínimos

A instalação do NuoDB para ambiente de produção está disponível apenas para o Linux.

O NuoDB suporta o Ubuntu 12.0.4 e 14.0.4, Red Hat Enterprise Linux 5.9, 5.10, 6.3 e 6.4 e CentOs 7.0.

É necessário ter o Java 1.7 ou Java 1.8 e o NuoDB já foi testado utilizando os seguintes softwares:

- Oracle Java 1.7 and 1.8
- Open JDK 1.7 and 1.8

O hardware mínimo recomendado é:

- x86_64 dual core CPU, 1.6GHz
- 4GB memory

É altamente recomendado que o sistema de arquivos utilizado suporte fallocate(). A chamada de sistema fallocate() é suportada pelos seguintes sistemas de arquivo:

- XFS
- ext4
- Btrfs
- tmpfs

A comunicação entre os processos do NuoDB em um domínio requer o Protocolo de Internet versão 4 (IPV4). IPV6 não é suportado.

Instruções de Instalação

Para realizar a instalação do NuoDB Community Edition versão 2.5.5.1.x86_64 no CentOS 7.0, que foi a distribuição escolhida, utilizando o pacote RPM (RedHat Package Manager), baixe o pacote nuodb-ce-2.5.5.1.x86_64.rpm diretamente do site do NuoDB.

Primeiro deve-se instalar o Java. Nesse caso, foi escolhido o Open JDK 8 (openjdk-8-jre). Para isso, utilize os comandos:

sudo yum install openjdk-8-jre

Após a instalação do Java, basta instalar o NuoDB:

sudo rpm --install nuodb-ce-2.5.5.1.x86_64.rpm

Tarefas de pós instalação

A única alteração que deve necessita ser realizada é alterar o arquivo default.properties, localizado no caminho NUODB_HOME/etc/default.properties.

Remova o comentário da linha da propriedade domainPassword e defina uma senha após o sinal de "=".

Esta será a senha que cada Broker e Agente que atuam neste domínio deve especificar.

Interações

CRUD

Uma das características mais relevantes do banco de dados NuoDb é a utilização da linguagem SQL, sendo por esta razão enquadrado em uma nova categoria de bancos de dados conhecida como NewSQL.

O acrônimo CRUD é utilizado frequentemente para definir as quatro operações básicas de um banco de dados. Seu significado éC(reate) R(etrieve) U(pdate) D(elete), ou melhor , são os comandos: INSERT, SELECT, UPDATE e DELETE.

SELECT

O comando SELECT, sem dúvida, é o comando mais utilizado em qualquer sistema de banco de dados relacional. Através dele consegue-se manipular grandes volumes de dados a nível de consulta.

Sintaxe:

SELECT campo1, campo2, ..., campoN, *

FROM tabela1, tabela2, ..., tabelaN

[WHERE condição]

[GROUPY BY ...]

[HAVING ...]

[ORDER BY ...]

Exemplo:

Using SELECT ALL vs SELECT DISTINCT

SELECT ALL playerid,year,stint FROM scoring

WHERE year = 2009 AND playerid LIKE 'boy%' ORDER BY playerid;

INSERT

O comando INSERT serve para inserir um registro em alguma tabela de um banco de dados. Todos os campos não nulos devem ser informados os demais são de caráter facultativo.

Sintaxe:

INSERT INTO

(campo1, campo2, ..., campoN)

VALUES(valor1, valor2, ..., valorN)

Exemplo:

```
INSERT INTO hockey.scoring
  VALUES ('aaltoan01', 2001, 1, 'ANA', 'C', 3, 0, 0, 0);
INSERT INTO hockey.scoring
  VALUES ('aaltoan01', 2002, 1, 'ANA', 'C', 3, 0, 0, 0), ('aaltoan01', 2003, 1, 'ANA', 'C', 3, 0, 0, 0);
```

UPDATE

O comando UPDATE serve para realizar a alteração de algum registro já inserido em alguma tabela de um banco de dados.

Sintaxe:

UPDATE

SET campo1 = valor1, campo2 = valor2, ..., campoN = valorN

[WHERE condição]

Exemplo:

```
UPDATE teams
  SET wins = wins + 1
 WHERE teamid IN ('COB','BOS','AND');
```

DELETE

O comando DELETE apaga um determinado registro em uma dada tabela.

Sintaxe

DELETE FROM

[WHERE condição]

Exemplo:

```
DELETE FROM teams
WHERE teamid IN ('COB','BOS','AND');
```

Otimização

O NuoDB tenta otimizar a execução de consultas, que possuem junções de tabelas. No entanto, a execução da consulta ainda pode ser inferior ao ideal e se for esse o caso, então você pode tentar qualquer uma das seguintes alternativas para melhorar o desempenho da consulta.

O primeiro passo na tentativa de diminuir o tempo das consultas é identificar quais consultas estão com tempo acima do esperado. Para isso o NuoDB disponibiliza as seguintes formas:

Consulta diretamente ao Histórico de Tabelas do Sistema

Para saber qual de suas consultas tomou uma quantidade excessiva de tempo, examinamos os registros na tabela SYSTEM.QUERYSTATS. A tabela SYSTEM.QUERYSTATS também é conhecida como o Log para Consultas Lentas. Por padrão, NuoDB armazena as dez consultas mais lentas, que levaram mais de dez segundos para executar. Isso pode ser modificado alterando valores MAX_QUERY_COUNT e MIN_QUERY_TIME em SYSTEM.PROPERTIES.

Para encontrar as consultas mais lentas, execute uma simples seleção a partir SYSTEM.QUERYSTATS.

Exemplo:

```
SET OUTPUT VERTICAL;
SELECT * FROM system.querystats;
================================== Row #1
==================================
SQLSTRING: select * from t where x=?;
COUNT: 1
RUNTIME: 13010424
USER: CLOUD
SCHEMA: USER
NUMPARAM: 1
PARAMS: 0/string/2
NODEID: 1
NROWS: 1
UNIQUEID: 4
TIMESTAMP: 2013-04-23 12:29:53.512637
EXPLAIN:
```

Console de Automação para consulta tabelas do sistema

É possível detectar consultas lentas consultando o histórico pelo Console de Automação, além de consultar a SYSTEM.QUERYSTATS. Para acessar as informações de consulta lenta, ir para a página de detalhes do banco de dados, clicando em bancos de dados no painel da esquerda e, em seguida, selecionar qual banco de dados você deseja investigar. Para ver dados históricos sobre as consultas que já tenha sido concluída, selecione a guia histórico consultas lentas.

As informações apresentadas aqui incluem um identificador único para a consulta, o tempo decorrido para a consulta, o nome do banco de dados, o host no qual a consulta está em execução e a instrução SQL. Isto é equivalente ao log de consultas lentas do NuoDB na SYSTEM.QUERYSTATS, tal como acontece com SYSTEM.QUERYSTATS, podem ser modificados os valores MAX_QUERY_COUNT e MIN_QUERY_TIME em SYSTEM.PROPERTIES.

Se você selecionar a ID para a consulta, você pode obter mais detalhes, incluindo a saída EXPLAIN para a consulta.

Colunas da QUERYSTATS

Column	Description
SQLString	SQL query string
Count	Number of times the query executed
Runtime	Execution time in microseconds; this is the last execution time and not an average
User	User who executed the query
Schema	Schema against which the query was run
NumParam	Number of parameters in the query
Params	Information about each parameter including location, type and value

NodeID	Internal ID for TE for this query
NRows	Number of rows returned by the query
Timestamp	When the query started
Explain	Shows the execution plan of the executed statement

Identificar a execução corrente de consultas lentas utilizando tabelas do sistema

NuoDB também fornece um mecanismo para análise de consulta em tempo real. SYSTEM.CONNECTIONS é outra tabela NuoDB. Informa todas as consultas atualmente em execução no seu banco de dados. Quando as consultas mostradas acima QUERYSTATS ainda estavam em execução, eles teriam sido relatados na tabela de ligações da seguinte forma:

```
SET OUTPUT VERTICAL;
SELECT sqlstring, count, runtime, user, schema, numparam, params, connid,
 open, handle, nodeid, execid FROM system.connections;
*************************** Row #1 ***************************
SQLSTRING: select * from t where x=?;
COUNT: 1
RUNTIME: 13010424
USER: CLOUD
SCHEMA: USER
NUMPARAM: 1
PARAMS: 0/string/2
CONNID: 2
OPEN: 1
HANDLE: 3
NODEID: 1
EXECID: 5534023222971858944 1
```

As informações são semelhantes a tabela de SYSTEM.QUERYSTATS. Ele identifica unicamente cada instrução SQL e podem ser utilizados como parâmetros para a instrução KILL.

```
SELECT sqlstring, runtime FROM system.connections;

SQLSTRING                                                    RUNTIME
------------------------------------                         --------
select * from t where x=?;                                   13010424
select s from t2 join t3 on t2.s=t3.s;                       14013823
```

Connections Columns

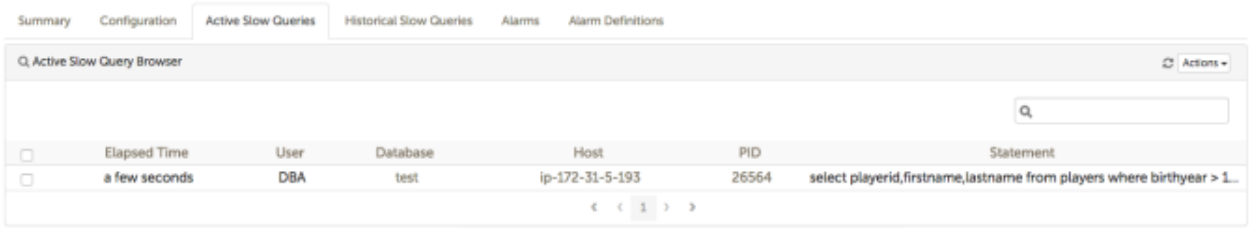
Column	Description
SQLString	SQL query string
Count	Number of times the statement executed
Runtime	Elapsed run time in microseconds of the transaction
User	User running the statement
Schema	Schema against which the statement is running
NumParam	Number of parameters in the statement
Params	Information about each parameter including location, type and value
ConnID	Connection ID number
Open	Number of open statements for the connection
Handle	Internal connection ID
OpenResults	Number of open result sets for the current connection
NodeID	Internal ID for TE for this connection

ExecID	Execution ID that can be used for KILL STATEMENT command
TransID	The identifier for the transaction being run on the connection. This value can be used to query SYSTEM.TRANSACTIONS with "WHERE id = *this_value*".
TransRunTime	Execution run time in microseconds of the transaction
AutoCommitFlags	Indicator for the value of the Autocommit system property setting when the statement was executed. 1=ON, 0=OFF
IsolationLevel	The isolation level defined for the connection when the statement was executed. See Transactions and Isolation Levels
ClientHost	Specifies the IP address for the connecting client.
ClientProcessID	Specifies the process ID (pid) for the connecting client.
ClientInfo	Specifies client-supplied information for the connecting client.
AutoCommitSPMode	
RollBackMode	

Identificar a execução corrente de consultas lentas usando o console Automation

É possível detectar a execução de consultas lentas no Console de automação. Para acessar as informações de consulta lenta, ir para a página de detalhes do banco de dados, clicando em bancos de dados no painel à esquerda e, em seguida, selecionar qual banco de dados você deseja investigar.

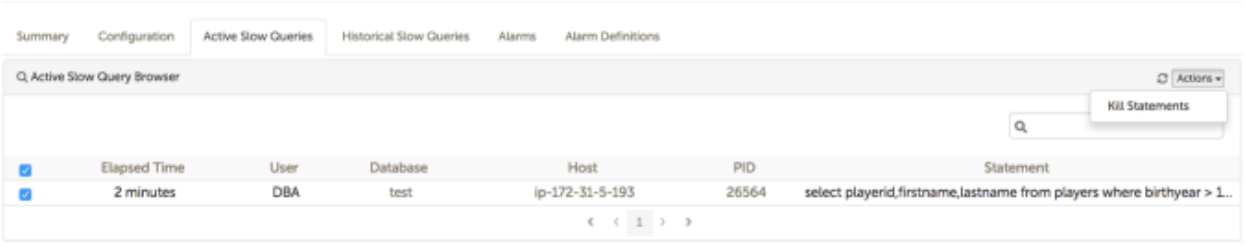
Para ver todas as consultas ativas que estão em execução no banco de dados, selecione a guia Ativo "Active Slow Queries"



Informações apresentadas no guia Ativo de Consultas lentas inclui o tempo decorrido da consulta, o usuário que executa a consulta, o banco de dados, o host no qual a consulta está sendo executado, e a identificação do processo do mecanismo de operação (TE). Isso é equivalente aos dados exibidos na SYSTEM.CONNECTIONS.

Ao contrário da maioria do Console Automation, por motivos de desempenho, a guia ativo de consultas lentos não é atualizada automaticamente.

Para matar uma consulta lenta, selecione a caixa de seleção ao lado de uma ou mais consultas e, em seguida, escolha Ações | Matar declarações.



Obtenção de Relatório SQL de Tempo Decorrido

Ferramenta útil na depuração de consultas lentas. Permite-lhe ligar ou desligar a notificação de tempo decorrido para cada instrução SQL que está sendo executada.

```
$ nuosql test --user dba --password goalie --timer full
SQL> select * from hockey.scoring limit 3000;

PLAYERID  YEAR  STINT  TEAMID  POSITION  GAMESPLAYED  GOALS  ASSISTS  PENALTYMINUTES
---------  -----  ------  -------  ---------  ------------  ------  --------  ---------------

aaltoan01  1997  1  ANA  C  3  0  0  0
aaltoan01  1998  1  ANA  C  73  3  5  24
...
bladoto01  1972  1  PHI  D  78  11  31  26
bladoto01  1973  1  PHI  D  70  12  22  37

Elapsed time 36ms
Server execution time 7ms
SQL>
```

Executando ANALYZE para obter os Índices de Estatísticas

A estatísticas de índice não são gerados por padrão. E são úteis tanto para o Query Optimizer e para o usuário. Eles nos dizem mais sobre o índice e sua eficácia com os dados específicos sobre os quais ele está sendo executado. O usuário deve executar ANALYZE para gerar estatísticas para uma tabela ou índice. Executar ANALYZE é necessária sempre que forem feitas alterações significativas ao conteúdo de uma tabela ou índice.

Usando sugestões de otimização

Dicas de otimização são um meio para indicar como o otimizador deve avaliar a instrução SELECT. Por exemplo, você pode saber, com base nos dados em uma determinada coluna, que um índice pode ser mais seletivo do que outro para a consulta particular que você está executando. Uma sugestão de otimização é útil nesta situação. Uma sugestão de otimização é incorporada na consulta SQL, logo após a palavra-chave SELECT.

Casos de Uso

Cenário Geral: Jogos em um mundo conectado

Com parte da indústria de jogos em alto crescimento. Todos estão fazendo malabarismos com as diversas plataformas devido ao crescimento explosivo no espaço móvel, e reconhecem o crescimento de um modelo orientado a serviços que depende de atualizações e da satisfação do jogador e do jogo em produzir incrementos através de compras no jogo e de conteúdo promocional.

Para cada novo jogo construído, está se pensando constantemente em como se adaptar a este ambiente conectado. Dados sobre jogadores, sua demografia, dispositivos e o estado de seus jogos salvos devem ser gerenciados e acessíveis imediatamente, onde quer que esses jogadores estejam, sem importar que hora do dia escolher para jogar, e independentemente de quantos jogadores estão on-line ao mesmo tempo.

Acima de tudo, todos esses dados devem ser usados para otimizar as decisões sobre conteúdo promocional (o que oferecer e quando a oferecê-lo) ou a implantação de novas funcionalidades. Enquanto isso, os jogadores querem em tempo real, acesso preciso para quadros de líderes, pontuação do torneio.

Todos esses requisitos e expectativas dos jogadores devem ser considerados na construção de uma plataforma de jogos e o banco de dados que serve de base fundamental para alcançar o sucesso.

Cliente: Start-Up que construiu uma plataforma multi-jogador

Uma Start-Up que possuía uma plataforma de jogos online implementou uma dimensão multi-jogador nos jogos single-player, aumentando a aderência e adicionando um componente competitivo nos jogos.

Este projeto requer um banco de dados que de forma rentável permanece ativo e disponível com o mínimo de latência, independentemente do volume de tráfego. Bancos de dados tradicionais são caros para realizar o pré-provisionamento para atender as demandas de pico. E os bancos de dados NoSQL não fornecem as transações necessárias para manter a consistência.

NuoDB

Por Que NuoDB?

- ACID transacional
- Consistência
- Elastic Scale-out
- Em data centers 2+

Benefícios:

- Solução escalável distribuídos geograficamente no NuoDB. Entrega capacidade a uma taxa económica.
- Simplicidade operacional
- Picos de demanda e uso global de padrões são acompanhadas por facilidade de gerenciamento e scale-out flexível.
- Facilidade de Migração

Suporte SQL NuoDB permite que simples e rápida migração de aplicativo de dados SQL existente.

Evidências

Preparamos um ambiente com os requisitos mínimos de instalação citados anteriormente, criando um nó do NuoDB em máquina virtual. Nesse capítulo, evidenciaremos por meio de imagens, algumas funções e comandos importantes do NuoDB, utilizando este ambiente.

Iniciando os serviços Agent, Rest e Web Console do NuoDB:

```
[root@localhost ~]# sudo service nuoagent start
Starting nuoagent (via systemctl):                         [  OK  ]
[root@localhost ~]# sudo service nuorestsvc start
Starting nuorestsvc (via systemctl):                       [  OK  ]
[root@localhost ~]# sudo service nuowebconsole start
Starting nuowebconsole (via systemctl):                    [  OK  ]
[root@localhost ~]# _
```

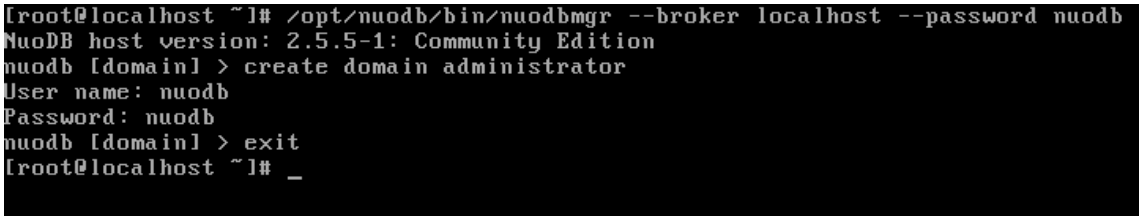
Abrindo o agent.log para verificar a inicialização do agent:

Aqui o objetivo foi verificar se o serviço do Agent havia iniciado sem falhas. Podemos perceber na imagem a seguir, com a mensagem "NuoAgent.log ready (main) startup completed; agent is ready.", que o objetivo foi atingido.

```
[root@localhost ~]# sudo service nuoagent start
Starting nuoagent (via systemctl):                         [  OK  ]
[root@localhost ~]# tail /var/log/nuodb/agent.log
2016-10-04T22:34:03.597-0300 INFORMAÇÕES PeerService.<init> (main) protocol vers
ion 4
2016-10-04T22:34:03.600-0300 INFORMAÇÕES DomainStateService.ready (main) Startin
g resync thread
2016-10-04T22:34:03.608-0300 INFORMAÇÕES LocalServer.start (main) Raft server st
arted
2016-10-04T22:34:03.608-0300 INFORMAÇÕES EventManager.notifyDomainEvent (main) J
oined Domain
2016-10-04T22:34:03.613-0300 INFORMAÇÕES SigarContainerImpl$ReconnectTimer.run (
Thread-8) found no running Nodes to reconnect
2016-10-04T22:34:03.617-0300 INFORMAÇÕES TagContainerImpl$DomainJoinedRunnable.r
un (Thread-9) TagContainer is ready
2016-10-04T22:34:03.619-0300 INFORMAÇÕES EventingContainerImpl$DomainListenerImp
l$1.run (evt-domain-list) EventContainer is ready (3 alarms read)
2016-10-04T22:34:03.621-0300 INFORMAÇÕES TagService$DomainJoinedRunnable.run (Th
read-10) Region is: DEFAULT_REGION
2016-10-04T22:34:03.623-0300 INFORMAÇÕES NuoAgent.logReady (main) startup comple
ted; agent is ready
2016-10-04T22:34:04.104-0300 INFORMAÇÕES SimpleAutoRestarter.restart (Thread-7)
no markers found at [/var/opt/nuodb/demo-archives]
[root@localhost ~]# _
```

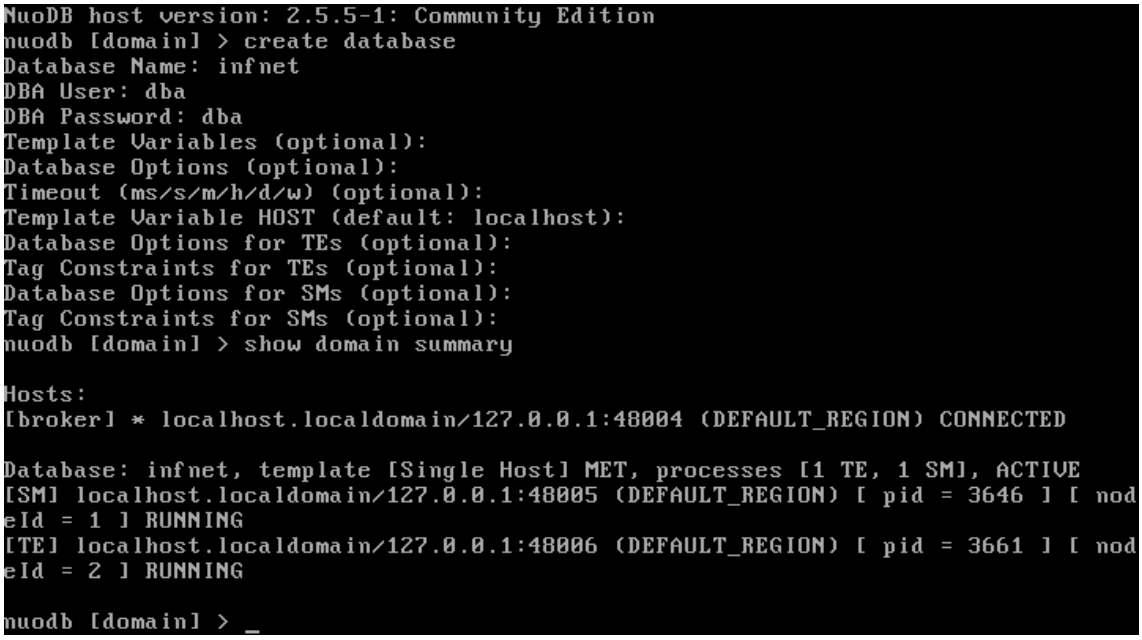
Criando um administrador do domínio:

Criamos um administrador do domínio utilizando o comando nuodbmgr, que deve ser acionado a partir do diretório <NUODB_INSTALL>/bin. Após isso, definimos o User Name e Password, conforme pode ser observado na imagem a seguir.



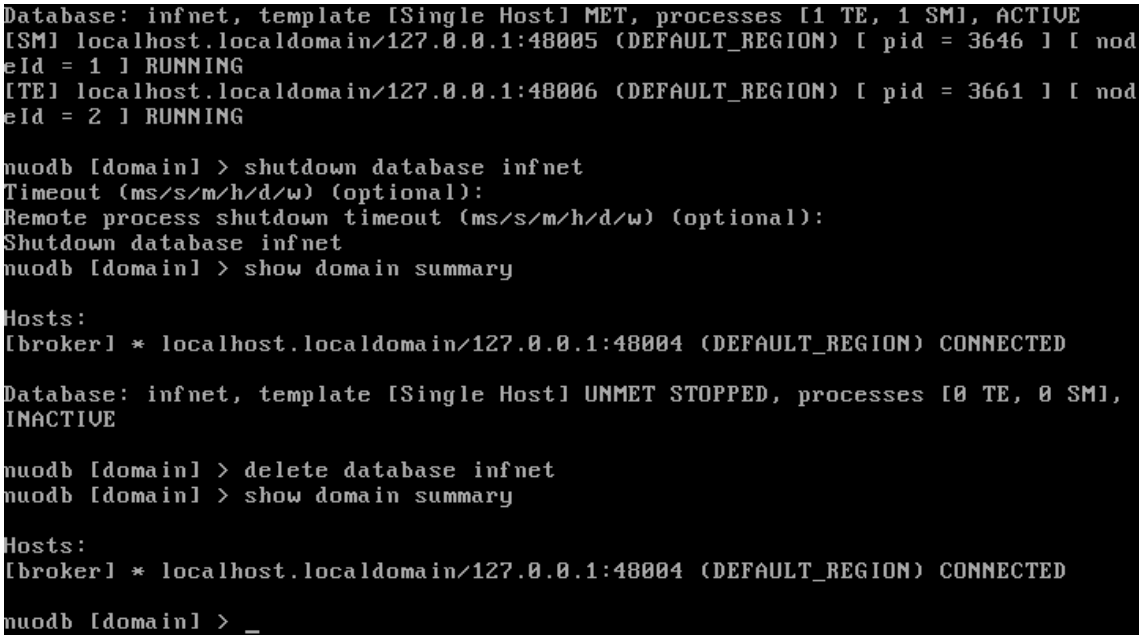
Criando uma base de dados:

Na seguinte imagem, evidenciamos a criação de uma nova base de dados de nome infnet, definimos o nome de usuário e a senha do usuário que irá administrar a base de dados e deixamos as demais opções com o padrão do NuoDB. Utilizamos o comando "show domain summary" para verificar se a base havia sido criada e suas características, assim como o status do SM e do TE.



Parando e apagando uma base de dados:

Pode-se perceber na imagem a seguir, que a base de dados infnet estava ativa, utilizamos o comando "shutdown database infnet" para desativá-la, verificamos a desativação e então deletamos a base, mostrando por fim, que ela já não constava mais na lista de bases de dados do domínio.

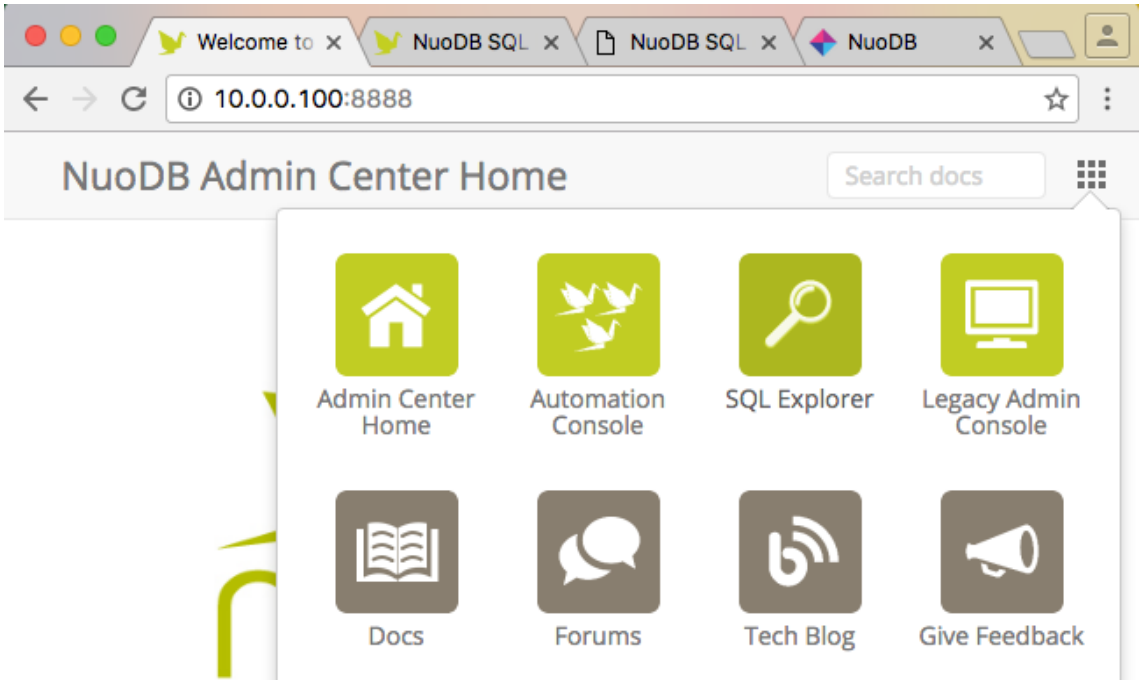


Criando um schema:

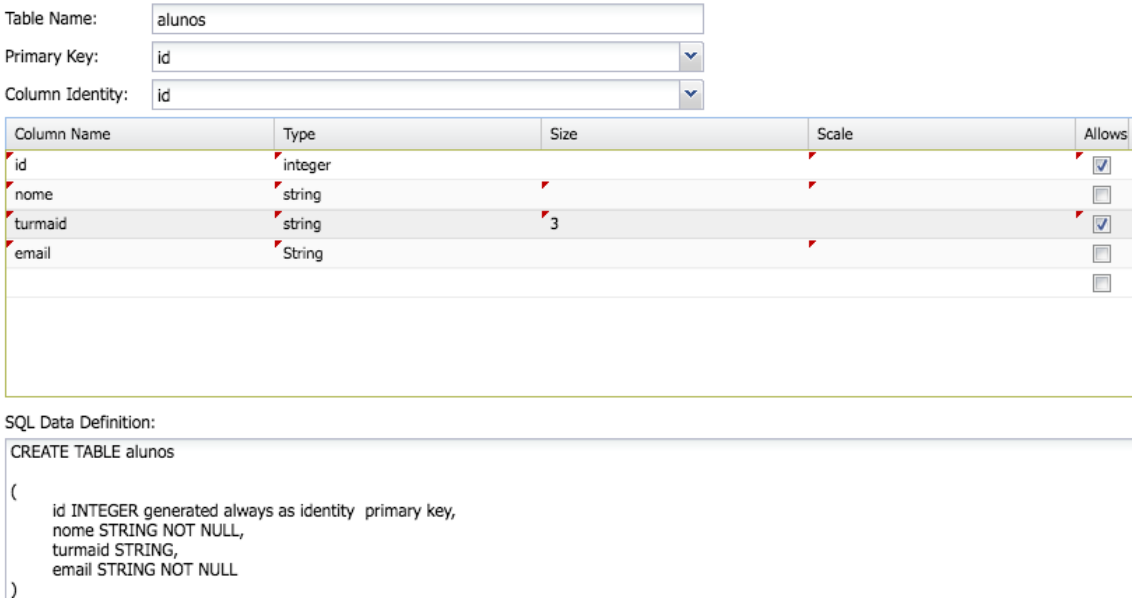
Na imagem a seguir, podemos ver o comando utilizado para criar um novo schema, de nome mit_bigdata. A ferramenta utilizada foi o nuosql, que assim como o nuodbmgr, deve ser executado a partir do diretório <NUODB_INSTALL>/bin. Com o comando "show schemas", evidenciamos a criação do segundo schema.

Criação de tabelas e inserção de dados:

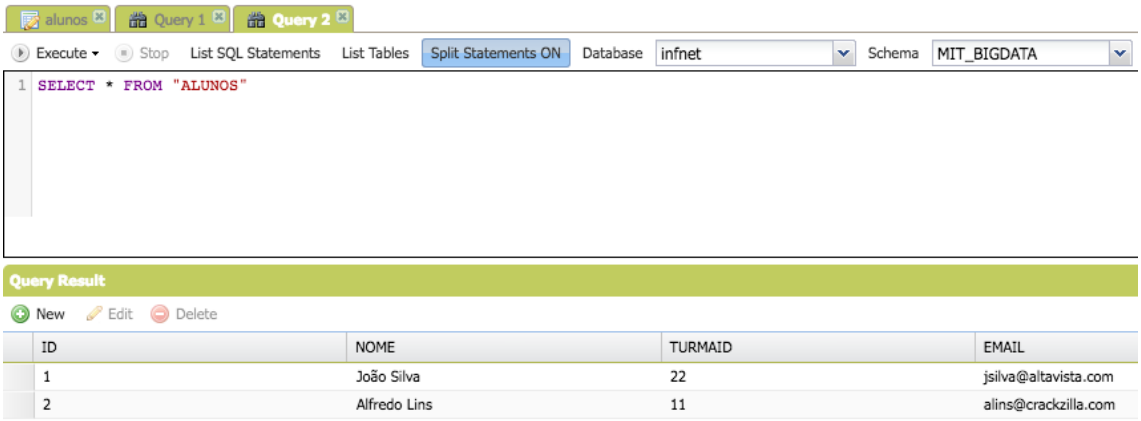
Escolhemos o NuoDB SQL Explorer como ferramenta para este objetivo, para reduzir o esforço administrativo das tarefas. O acesso é feito a partir de um botão no menu do NuoDB Admin Center, que deve ser acessado pelo navegador da internet, com a URL http://x.x.x.x:8888, sendo x.x.x.x o IP do servidor do NuoDB.



Após fazer login com a conta de DBA da base de dados, criamos uma tabela de nome alunos no schema mit_bigdata e configuramos as colunas e suas propriedades.



Após executar a tarefa, utilizamos a ferramenta para criação de queries, disponível no NuoDB SQL Explorer para inserir os dados de teste e depois evidenciamos com um comando select.



Conclusões

Neste trabalho, foi apresentado um breve estudo sobre as características do NuoDB, suas aplicações, arquitetura e algumas tarefas administrativas mais comuns. A partir de um estudo teórico, demonstramos que o NuoDB é um sistema de banco de dados que pode atender aplicações e serviços que necessitem garantir a consistência dos dados por meio do conceito ACID e ainda ter a possibilidade de escalar horizontalmente na nuvem. Mostramos os conceitos que podem ser utilizados com o NuoDB para que um sistema ou serviço atenda clientes em regiões diferentes do mundo, garantindo a geo-localização dos dados, alta disponibilidade e desempenho. Além disso, evidenciamos, que o NuoDB pode reduzir custos com contratações e treinamento, assim como reduzir o risco, quando se tratando de migrar as regras de negócio e/ou transações dos sistemas já existentes para código de aplicação, eliminando as opções de bancos de dados NoSQL para este fim.

Todas as evidencias apresentadas, provam a facilidade de instalação e configuração do NuoDB para fins de desenvolvimento e sua interface de administração simples e intuitiva, além do uso da linguagem SQL, já difundida no mercado.

Assim, espera-se angariar recursos e apoio para a pesquisa e desenvolvimento desta tecnologia e fomentar o uso do NuoDB por parte das empresas que tem as necessidades citadas neste trabalho.

Referências bibliográficas

http://www.nuodb.com/

http://doc.nuodb.com/