HORUS
FACULDADES

Introdução ao Uso de Banco de Dados

PROF. ESP. EZEQUIEL JULIANO MÜLLER

HORUS FACULDADES

CURSO DE SISTEMAS DE INFORMAÇÃO

PINHALZINHO - SC

Definições

Banco de Dados é uma coleção de dados inter-relacionados, representando informações sobre um domínio específico.

Sistema de Banco de Dados é um sistema de manutenção de registros por computador envolvendo quatro componentes principais, sendo eles dados, hardware, software e usuários.

O **SGBD (Sistema Gerenciador de Banco de Dados)** é um software com recursos específicos para facilitar a manipulação das informações dos dados e o desenvolvimento de programas aplicativos.

Definições

Os **sistemas de banco de dados** são projetados para gerenciar grandes grupos de informações.

O gerenciamento de dados envolve a definição de estruturas para armazenamento de informação e o fornecimento de mecanismos para manipulá-las.

Além disso, o **sistema de banco de dados** precisa fornecer segurança das informações armazenadas, caso o sistema dê problema, ou contra tentativas de acesso não-autorizado. Se os dados devem ser divididos entre diversos usuários, o sistema precisa evitar possíveis resultados anômalos.

A importância das informações na maioria das organizações e o consequente valor dos bancos de dados têm orientado o desenvolvimento de um grande corpo de conceitos e técnicas para o gerenciamento eficiente dos dados.

Sistemas de Banco de Dados Relacionais

O objetivo de um **sistema de banco de dados relacional** é isolar os usuários dos detalhes mais internos do banco de dados (abstração de dados) e também prover independência de dados ás aplicações (estrutura física de armazenamento e à estratégia de acesso).

As vantagens na sua utilização são a rapidez na manipulação e no acesso á informação, redução do esforço humano (desenvolvimento e utilização), disponibilização da informação no tempo necessário, controle integrado de informações distribuídas fisicamente, redução de redundância e de inconsistência de informações, compartilhamento de dados, aplicação automática de restrições de segurança, redução de problemas de integridade etc.

Sistema de Banco de Dados Relacionais – Papéis Humanos

Em um sistema de Banco de Dados Relacional possuímos a seguinte hierarquia: **“Programadores de Aplicações”, “Usuários Sofisticados”, “Usuários Especializados”** e **“Usuários Ingênuos”.**

Em um sistema de banco de dados também possuímos administradores, aquelas pessoas que são responsáveis pelo controle do sistema de banco de dados sendo o **“Administrador de Dados (AD)”** e o **“Administrador do SGBD (DBA)”.**

Na administração de sistemas de banco de dados relacionais o papel do **“Administrador de Dados (AD)”** é efetuar a definição e atualização do esquema do banco de dados e o papel do **“Administrador do SGBD (DBA)”** é a definição da estrutura de armazenamento e a estratégia de acesso, concessão de autorização para acesso a dados, definição de controles de integridade, definição de estratégias para copia de segurança (backup) e recuperação (recover), monitoramento do desempenho, execução de rotinas de desempenho, modificação da organização física, etc.

Pergunta...

1. Porque a informação e, consequentemente os bancos de dados, tem se tornado peças chave no atual momento do ambiente corporativo?

Modelo Relacional

Desde sua criação no início dos anos 1970, o Modelo Relacional de dados tem sido utilizado em larga escala pela grande maioria dos sistemas de gerenciamento de banco de dados.

Tendo surgido como sucessor dos modelos hierárquico e de rede, o modelo relacional tornou-se padrão para a grande maioria dos SGBDs (Sistemas Gerenciadores de Banco de Dados), tais como o SQL Server, Oracle, PostgreSQL, MySQL, FirebirdSQL etc. Seus elementos básicos são as relações (ou tabelas), as quais são compostas de linhas (ou tuplas) e colunas (ou atributos).

Outra característica fundamental desse modelo é a utilização de restrições de integridade. Esses elementos são utilizados para garantir que a integridade dos dados seja mantida. As restrições de integridade mais comuns são as chaves, mais especificamente, as chaves primárias e as chaves estrangeiras.

Modelo Relacional

Outra característica importante do Modelo Relacional é o processo de Normalização. Seu objetivo é a aplicação de certas regras sobre as tabelas do banco de dados, de forma a garantir o projeto adequado dessas tabelas. Uma característica básica da normalização consiste na separação dos dados referentes a elementos distintos em tabelas distintas, associadas através da utilização das chaves.

Modelo Relacional

Adicionalmente, o modelo relacional passou a adotar como linguagem de definição, manipulação e consulta de dados a SQL (Structured Query Language).

Desenvolvida originalmente pela IBM, o SQL é uma linguagem declarativa de para banco de dados relacional inspirada na álgebra relacional.

Sua simplicidade e alto poder de expressão fizeram do SQL a linguagem de consulta de dados mais utilizada no mundo e ajudou a consolidar a posição dominando do modelo relacional.

DDL - Data Definition Language

DCL - Data Control Language

DML - Data Manipulation Language

SGBD’s de Modelo Relacional

Características Gerais de um SGBD Relacional

Controle de Redundâncias: A redundância consiste no armazenamento de uma mesma informação em locais diferentes, provocando inconsistências. Em um banco de dados as informações só se encontram armazenadas em um único local, não existindo duplicação descontrolada dos dados. Quando existem replicações dos dados, estas são decorrentes do processo de armazenagem típica do ambiente Cliente-Servidor, totalmente sob controle do banco de dados.

Compartilhamento dos Dados: O SGBD deve incluir software de controle de concorrência ao acesso dos dados, garantindo em qualquer tipo de situação a escrita/leitura de dados sem erros.

Controle de Acesso: O SGBD deve dispor de recursos que possibilitem selecionar a autoridade de cada usuário. Assim um usuário poderá realizar qualquer tipo de acesso, outros poderão ler alguns dados e atualizar outros e outros ainda poderão somente acessar um conjunto restrito de dados para escrita e leitura.

Características Gerais de um SGBD Relacional

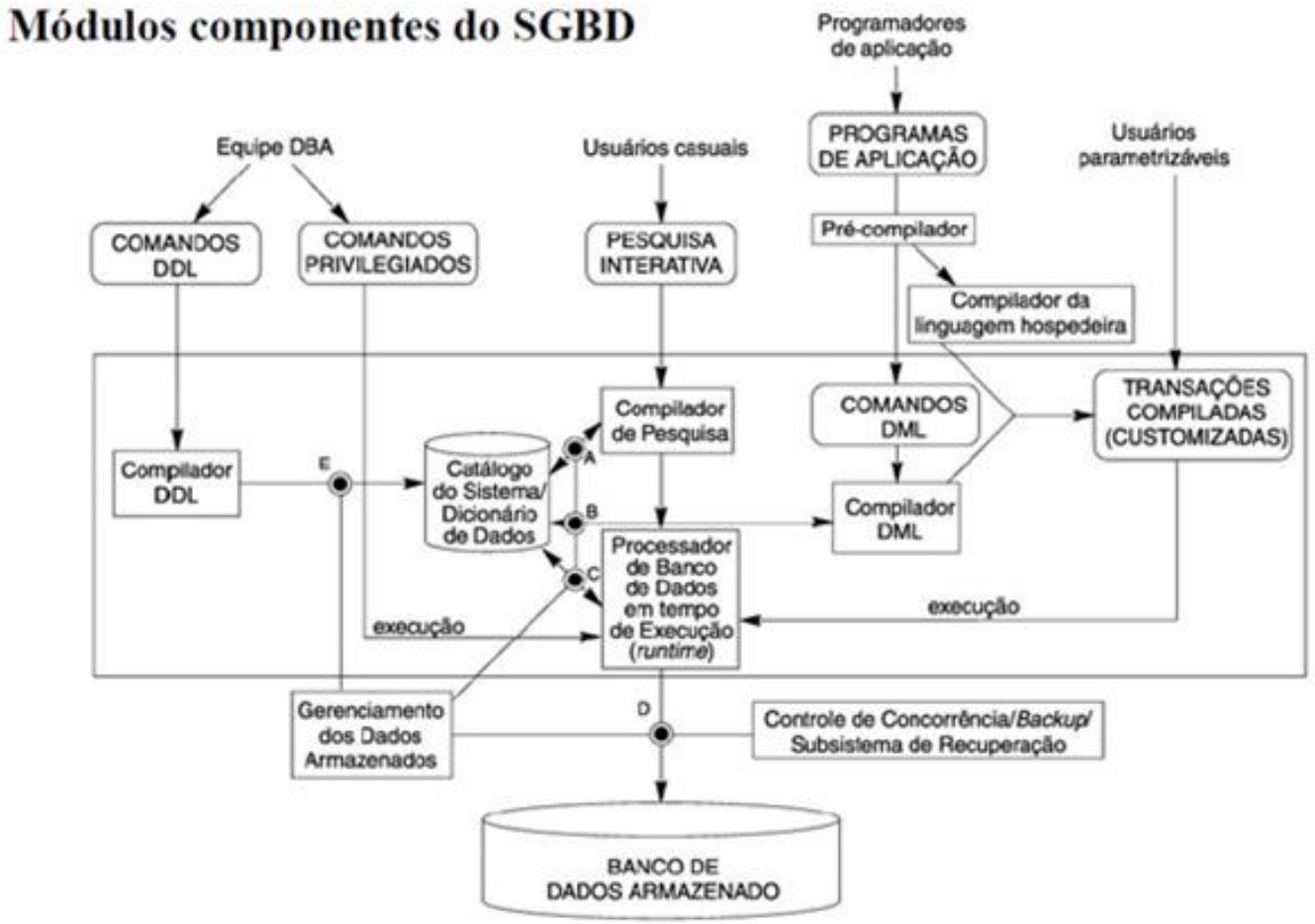
Interfaceamento: Um banco de dados deverá disponibilizar formas de acesso gráfico, em linguagem natural, em SQL ou ainda via menus de acesso, não sendo uma “caixa-preta” somente sendo passível de ser acessada por aplicações.

Esquematização: Um banco de dados deverá fornecer mecanismos que possibilitem a compreensão do relacionamento existente entre as tabelas e de sua eventual manutenção.

Controle de Integridade: Um banco de dados deverá impedir que aplicações ou acessos pelas interfaces possam comprometer a integridade dos dados.

Backups: O SGBD deverá apresentar facilidade para recuperar falhas de hardware e software, através da existência de arquivos de “pré-imagem” ou de outros recursos automáticos, exigindo minimamente a intervenção de pessoal técnico.

Módulos Componentes do SGBD



SGBD Relacional

Os SGBDs relacionais oferecem aos usuários processos de validação, verificação e garantias de integridade dos dados, controle de concorrência, recuperação de falhas, segurança, controle de transações, otimização de consultas, dentre outros.

A utilização de tais recursos facilitou a vida dos desenvolvedores de aplicações, possibilitando que estes pudessem se preocupar exclusivamente com o foco da aplicação.

SGBD Relacional

Como um dos conceitos mais básicos do modelo relacional, as chaves representam uma forma simples e eficaz de associação entre as tabelas do banco de dados.

A chave primária foi criada com o objetivo de identificar de forma única as tuplas da tabela e ainda de determinar a ordem física dessas tuplas.

A chave estrangeira permite uma relação de dependência entre atributos de tabelas distintas, de forma que os valores permitidos em um atributo dependam dos valores existentes em outro atributo.

Tais recursos são amplamente utilizados em bancos de dados relacionais e servem como base para a utilização de outros componentes, como é o caso dos índices. Estes elementos tornaram-se padrão para todo tipo de tabela por propiciarem um significativo ganho de performance no processamento de consultas.

SGBD Relacional

Além desses componentes, os SGDBs Relacionais possibilitam que múltiplos usuários possam acessar e manipular um mesmo banco de dados simultaneamente de forma eficiente, recurso indispensável para sistemas de grande porte.

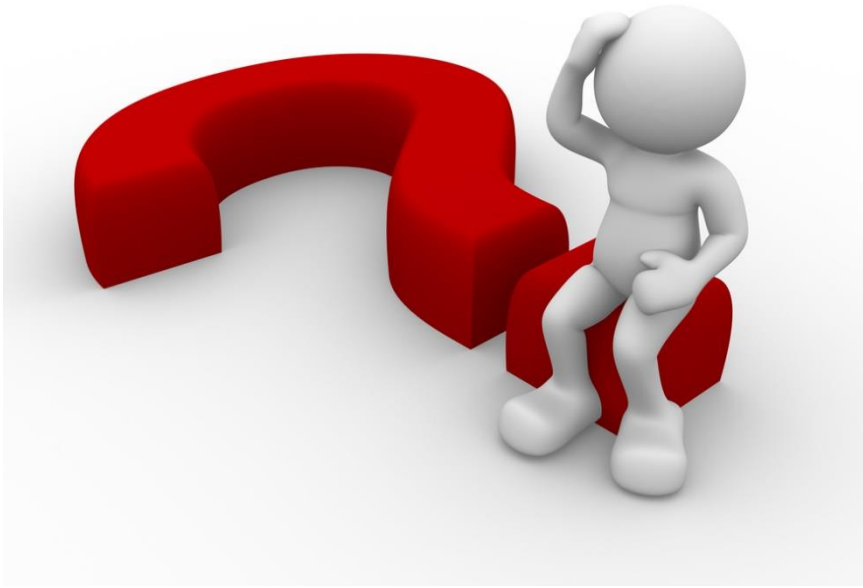
Outra característica importante dos SGDBs relacionais consiste na possibilidade do sistema se recuperar de forma adequada de possíveis falhas. O sistema tem a capacidade de retornar ao ponto anterior â falha ocorrida, garantindo que o banco permanecerá em um estado consistente.

Todos esses recursos ajudaram a manter os SGDBs Relacionais em posição de destaque entre os mais diversos tipos de ambientes computacionais, mas não impediram o surgimento de certos problemas, principalmente devido ao crescimento vertiginoso do volume de dados presentes nos bancos de certas organizações.

Perguntas...

1. O que é a DML? Cite exemplos.
2. O que é a DDL? Cite exemplos.
3. Para que serve a PK (chave primária) e a FK (chave estrangeira)?
4. Quais das características gerais de um SGBD você considera mais importante? Por quê?

Introdução a Banco de Dados Não-Relacionais



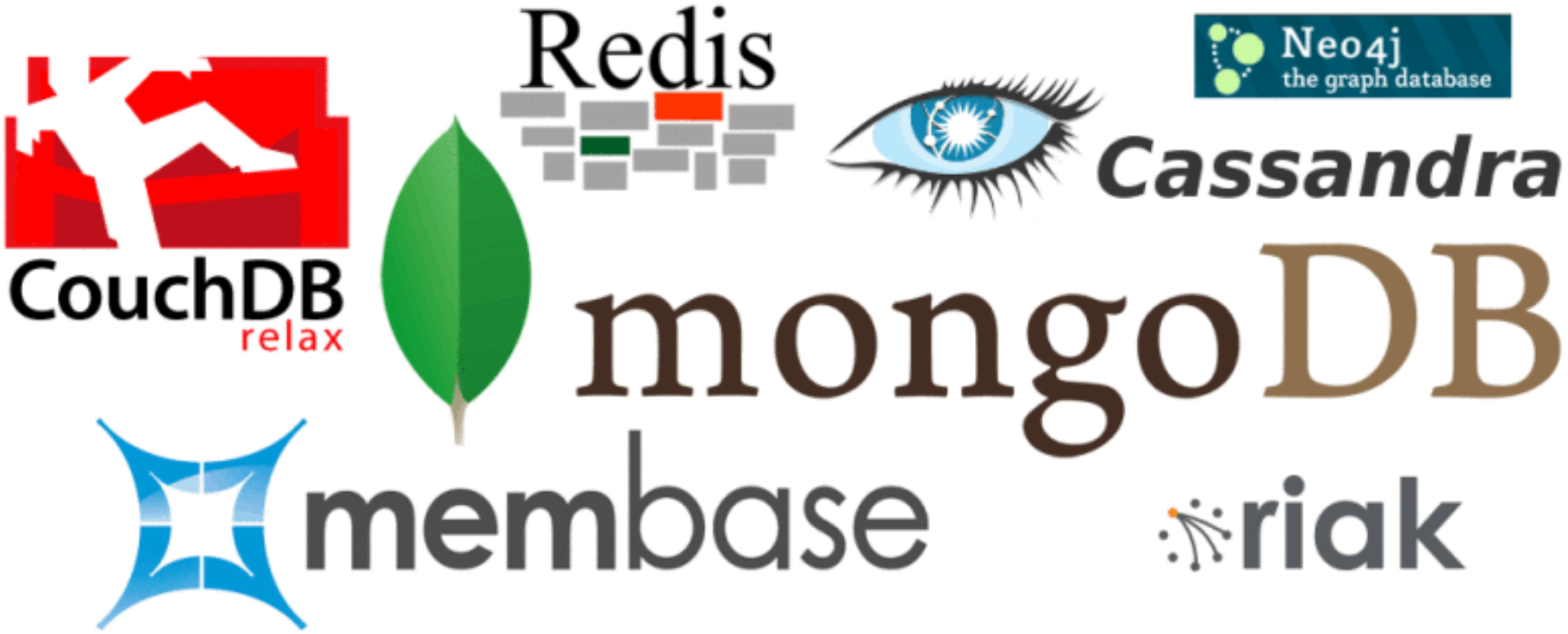
Introdução a Banco de Dados NoSQL

O Modelo Relacional tem sido amplamente utilizado em praticamente todos os tipos de sistemas de bancos de dados nas últimas décadas.

Porém, com o crescimento cada vez mais intenso do volume de dados de certas organizações, certos fatores limitantes têm propiciado que modelos alternativos de banco de dados sejam utilizados em tais cenários.

Motivados principalmente pela questão da escalabilidade do sistema, uma nova geração de bancos de dados – conhecidos como NoSQL – vem ganhando força e espaço tanto nas universidades quanto no mercado de trabalho.

Alguns Banco de Dados NoSQL

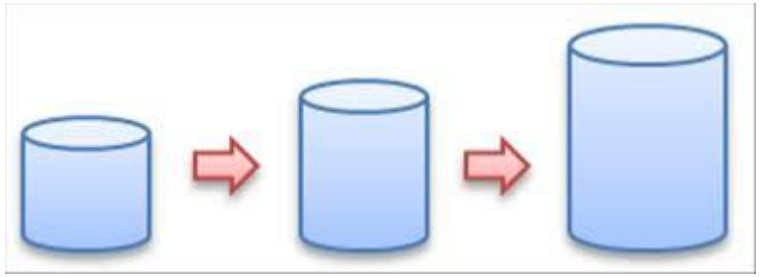


Modelo Relacional - Limitações

Devido ao intenso crescimento do número de aplicações, soluções, recursos e tudo o que se refere a sistemas computacionais nas últimas décadas, o volume de dados associados a tais tipos de sistemas também teve um ritmo de crescimento acelerado.

Atualmente, o volume de dados de determinadas organizações, como é o caso do Google, atinge o nível de petabytes. Em cenários nos quais o gerenciamento de grande volume de dados tem se mostrado problemático, a utilização de SGBDs relacionais, ou em outras palavras do próprio Modelo Relacional, já não se mostra tão eficiente.

De um modo geral, os principais problemas encontrados com a utilização do Modelo Relacional estavam concentrados na dificuldade em se conciliar tal modelo com a demanda por escalabilidade cada vez mais frequente.



Modelo Relacional - Limitações

Como exemplo, tomemos uma determinada aplicação web sendo executada sobre um SGBD relacional. Com o crescimento do número de usuários, o sistema começa a ter uma queda de performance. Para superar esse problema, restam duas alternativas promover um upgrade no servidor ou aumentar o número de servidores.

Tais opções não seriam suficientes se o número de usuários continuar a crescer em ritmo intenso porque o problema passa a se concentrar no próprio acesso à base de dados. A solução, portanto, passa a ser escalar o banco de dados, distribuindo-o em várias máquinas. Tal abordagem é possível em um SGBD relacional, porém, não é realizada de maneira tão simples.

Devido a sua natureza estruturada, os desenvolvedores começaram a perceber a dificuldade em se organizar os dados do Modelo Relacional em um sistema distribuído trabalhando com particionamento de dados. É justamente nesse ponto em que o foco das soluções não-relacionais estão concentradas.

Banco de Dados NoSQL

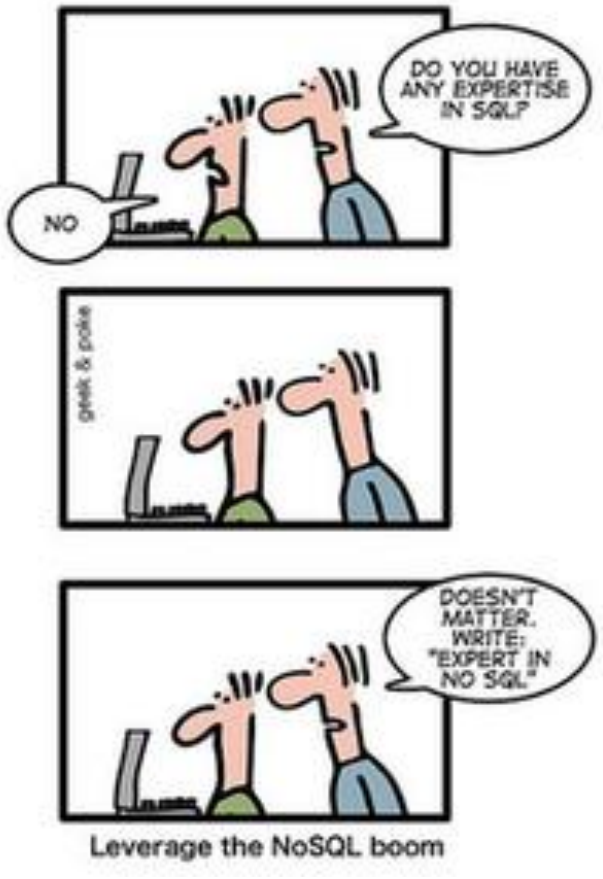
As mudanças ocorridas na tentativa de se propor alternativas ao uso do Modelo Relacional levaram os desenvolvedores a pensar em um modo alternativo de se modelar as bases de dados. A estrutura pouco flexível utilizada até então passou a ser um problema a ser contornado e as soluções propostas tinham como base a eliminação ou minimização dessa estruturação.

Se, por um lado, tais soluções perdiam todo o arcabouço de regras de consistência presentes no Modelo Relacional, por outro lado, poderiam ganhar em performance, flexibilizando os sistemas de banco de dados para as características particulares de cada organização.

Banco de Dados NoSQL

O termo NoSQL surgiu em 1998, a partir de uma solução de banco de dados que não oferecia uma interface SQL, mas esse sistema ainda era baseado na arquitetura relacional. Posteriormente, o termo passou a representar soluções que promoviam uma alternativa ao Modelo Relacional, tornando-se uma abreviação de **Not Only SQL (não apenas SQL), sendo utilizado principalmente em casos em que o Modelo Relacional não apresentava a performance adequada.**

O propósito, portanto, das soluções NoSQL não é substituir o Modelo Relacional como um todo, mas apenas em casos nos quais seja necessária uma maior flexibilidade da estruturação do banco.



Banco de Dados NoSQL

Uma das primeiras implementações de um sistema realmente não-relacional surgiu em 2004 quando o Google lançou o BigTable, um banco de dados proprietário de alta performance que tinha como objetivo promover maior escalabilidade e disponibilidade. A ideia central era justamente flexibilizar a forte estruturação utilizada pelo Modelo Relacional.

Outra implementação foi realizada em 2007, pela Amazon, com a apresentação do sistema Dynamo, o qual tinha como característica básica ser um banco de dados não-relacional, de alta disponibilidade, utilizado pelos web services da Amazon

Banco de Dados NoSQL

Em 2008, iniciou-se outro importante projeto. O Cassandra foi projetado para ser um sistema de banco de dados distribuído e de alta disponibilidade, desenvolvido pelo site Facebook para lidar com grandes volumes de dados. No início de 2010, o Cassandra desbancou o MySQL como banco de dados do Twitter, demonstrando sua importância cada vez mais crescente.

A partir dai começaram a surgir outras opções como o CouchDB, MongoDB etc...

NoSQL no Delphi ?

Sim é possível!

Existem algumas bibliotecas/drives para utilização de alguns bancos de dados NoSQL em Delphi.

TMongoWire = https://github.com/stijnsanders/TMongoWire

Pebongo = https://code.google.com/p/pebongo/

Mongo-Delphi-Driver = https://github.com/gerald-lindsly/mongo-delphi-driver

mORMot = http://synopse.info/fossil/wiki/Synopse+OpenSource

Análise Comparativa Modelo Relacional x NoSQL

	Relacional	NoSQL
Escalonamento	Possível, mas complexo. Devido à natureza estruturada do modelo, a adição de forma dinâmica e transparente de novos nós no grid não é realizada de modo natural.	Uma das principais vantagens desse modelo. Por não possuir nenhum tipo de esquema pré-definido, o modelo possui maior flexibilidade o que favorece a inclusão transparente de outros elementos.
Consistência	Ponto mais forte do modelo relacional. As regras de consistência presentes propiciam uma maior grau de rigor quanto à consistência das informações.	Realizada de modo eventual no modelo: só garante que, se nenhuma atualização for realizada sobre o item de dados, todos os acessos a esse item devolverão o último valor atualizado.
Disponibilidade	Dada a dificuldade de se conseguir trabalhar de forma eficiente com a distribuição dos dados, esse modelo pode não suportar a demanda muito grande de informações do banco.	Outro fator fundamental do sucesso desse modelo. O alto grau de distribuição dos dados propicia que um maior número de solicitações aos dados seja atendida por parte do sistema e que o sistema fique menos tempo não-disponível.

Modelo Relacional x NoSQL

A decisão de se realizar uma mudança dessa natureza – optar por um abordagem NoSQL em contraste com uma abordagem de um SGSBD tradicional – deve levar em consideração as necessidades do problema. Os fatores que devem ser utilizados como critérios de comparação são de tipos diversos, indo desde a escalabilidade do sistema, passando por questões de consistência de dados, até problemas de facilidade de uso ou existência ou não de linguagens de consulta.

Como pontos positivos dos SGBDs relacionais deve-se observar que tais sistemas são soluções muito mais maduras e experimentadas, enquanto que as implementações NoSQL ainda estão definindo um padrão próprio. Além disso, a consistência de dados continua sendo um fator a ser considerado com atenção e as transações dos SGBDs relacionais ainda são a melhor forma de se trabalhar com esse problema.

Modelo Relacional x NoSQL

Outro fator importante a ser considerado é a ausência de uma linguagem de consulta, como é o caso do SQL para os SGBDs relacionais. Não existe, em qualquer abordagem NoSQL, nada que se aproxime da simplicidade e expressividade oferecida pelo SQL. Adicionalmente, perde-se toda a funcionalidade oferecida pela linguagem, tais como funções, rotinas, etc.

Adicionalmente, deve-se ter em mente que a escalabilidade em alto grau faz-se necessária apenas em bancos de grande porte, nos quais a alta disponibilidade é imprescindível. Utilizar uma solução NoSQL para bancos de dados nos quais a disponibilidade não seja um fator imprescindível ainda é uma abordagem discutível.

Pergunta...

1. O que você considera mais importante: consistência ou escalabilidade? Por quê?

Referências

EMERSON S. GAUDENCIO. **Conceituando Banco de Dados e SGBD.** Disponível em: <http://certificacaobd.com.br/>. Acesso em: 14 ago. 2014.

BRITO, Ricardo W.. **Bancos de Dados NoSQL x SGBDs:** Relacionais:Análise Comparativa*. Disponível em: <http://www.infobrasil.inf.br/userfiles/27-05-S4-1-68840-Bancos de Dados NoSQL.pdf>. Acesso em: 14 ago. 2014.

Dúvidas?