Banco de dados NOSQL para integração de bases de dados de gases de efeito estufa

Thamires Dupre Guimarães[1]
Alan Massaru Nakai[2]
Luciano Vieira[2]

O objetivo deste trabalho[3] é estudar e desenvolver soluções para armazenar os dados pertinentes aos projetos citados. Uma vez que esses dados serão coletados em projetos de pesquisa em rede, envolvendo diversos grupos, poderão ocorrer mudanças nas necessidades referentes ao processo de experimentação que, no curso do experimento, naturalmente necessitam de ajustes. Isto pode trazer impactos ao sistema de informação que as suportam, o que inviabiliza o uso de um esquema fixo, como os existentes em banco de dados relacionais. Diante deste cenário, é altamente desejável que a solução computacional em desenvolvimento possua características adaptativas que permitam o atendimento ágil à evolução das necessidades e requisitos, como é o caso dos bancos de dados NoSQL (Bancos de Dados Não Relacionais) livres de esquema.

Os bancos de dados NoSQL (REDMOND; WILSON, 2012) foram criados para suprir carências que surgiram com o aumento da quantidade e diversidade de dados das aplicações atuais que não são completamente atendidas pelos bancos de dados relacionais, uma vez que exigem cada vez mais espaço de armazenamento. Segundo Brito (2010), os principais benefícios da abordagem NoSQL são: alta disponibilidade, menor tempo de resposta, paralelismo, flexibilidade de esquema e escalonamento horizontal. Grandes

[1] Universidade Estadual de Campinas - thamires.dupre@gmail.com

[2] Embrapa Informática Agropecuária - {alan.nakai, luciano.vieira}@embrapa.br

[3] Este trabalho está inserido no contexto dos projetos Fluxus e Saltus, da Embrapa, que visam desenvolver sistemas de informação para integração de bases de dados para o monitoramento da dinâmica da emissão de gases de efeito estufa e dos estoques de carbono nas culturas de grãos e nas florestas brasileiras, naturais e plantadas, respectivamente.

empresas como Google, Facebook e Youtube utilizam bancos de dados NoSQL para atender suas complexas necessidades de armazenamento de informação. Entretanto, a adoção deste tipo de tecnologia não é adequada para todos os casos. Nestes bancos, as operações de junção são bastante custosas e os mecanismos para consistência dos dados são limitados. Além disso, as diferenças de interfaces e formatos de dados dos diferentes gerenciadores NoSQL podem causar aprisionamento tecnológico (DE DIANA; GEROSA, 2010).

A primeira etapa do trabalho foi uma pesquisa sobre os diferentes tipos de bancos de dados NoSQL. As principais categorias existentes, segundo Toth (2011) são: (i) *Chave/Valor*, que armazenam os dados como chaves e valores (ex. Riak, Redis e DynamoDB); *Documentos*, que armazenam documentos (ex. CouchDB e MongoDB); *Grafos*, que permitem armazenar relacionamentos entre os dados (ex. Neo4J e *BigData*); e *Família de colunas*, que armazenam os dados como triplas contendo linha, coluna e rótulo de tempo (ex. Cassandra, HBase e Amazon SimpleDB).

O banco de dados NoSQL escolhido para esse projeto foi o MongoDB, um gerenciador de banco de dados voltado a documentos, que busca combinar as vantagens do armazenamento *chave-valor* (rápidos e escaláveis) com suporte a consultas complexas, típicas de BD relacionais (MONGODB, 2013). Um fator decisivo para essa escolha foi o fato de esse banco ser ideal para trabalhar com tabelas esparsas, onde muitos campos não são preenchidos. Esta característica é muito importante no contexto dos projetos Fluxus e Saltus, nos quais as definições sobre o que será armazenado poderá variar ao longo dos projetos.

No MongoDB o termo Coleção é usado para especificar um conjunto de dados, de forma análoga às tabelas do bancos de dados relacionais; um *Documento*, por sua vez, é a estrutura que contém os dados armazenados, como os registros do banco de dados relacional. Para exemplificar, uma coleção chamada *Biomassa Vegetal*, pode possuir documentos com campos como *Código*, *Data da Coleta*, *Responsável*, *Massa Verde Total* e *Quantidade de Carbono*. Entretanto, nem todos esses campos precisam estar preenchidos em todos os documentos. Além disso, no futuro pode ser necessário incluir outros campos em novos documentos.

O MongoDB possui bibliotecas de manipulação para várias linguagens de programação (Java, C++, Lua, .NET, entre outros). Esse banco é indicado para blogs, aplicações com muito conteúdo e informações estatísticas, pois

possui vários métodos que facilitam a replicação da informação e o armazenamento de grandes dados.

A segunda etapa do trabalho, que ainda está em andamento, é o desenvolvimento de uma API[4] Java para facilitar o desenvolvimento do sistema de informação para os projetos Saltus e Fluxus. A API em desenvolvimento permite realizar as quatro principais operações de um banco de dados: Inserir, Alterar, Remover e Consultar. Ela adiciona à API nativa do MongoDB funcionalidades como a inserção de documentos diretamente de arquivos e facilidades para construção de consultas complexas com combinações de restrições, semelhantes as consultas com a cláusula *WHERE* do SQL (ex. *Atributo1 = 100 e Atributo2 > 10*). A Tabela 1 mostra a sintaxe das principais funcionalidades da API.

Tabela 1. Principais Métodos da API desenvolvida.

Método	Descrição	Parâmentros
`static void remover_id( String nome_banco, String colecao, String id)`	Exclui um documento a partir de seu identificador	**nome_banco** - a base a ser usada; **colecao** *– nome da tabela a ser usada; **id** - o valor do identificador unico do que deseja excluir
`void insere_deArquivo(String nome_banco, String colecao, ArrayList<String> doc)`	Insere todos os documentos lidos em um arquivo	**doc** - É um conjunto documentos** a ser inserido lidos a partir de um arquivo de texto
`static ArrayList recuperar_lista (String nome_banco, String colecao, ArrayList <Query_parameters> pars)`	Retorna todos os documentos que atendem parametros passados	**pars** - Uma lista contendo as condições para a busca

Continua...

[4] API, de *Application Programming Interface* (ou Interface de Programação de Aplicativos), é um conjunto de rotinas estabelecidos por um software para a utilização das suas funcionalidades usando apenas seus serviços.

Tabela 1. Continuação.

Método	Descrição	Parâmentros
`static void atualiza_dados(String nome_banco, String colecao, String id, String campo, String novo_valor)`	Atualiza o valor de um campo do documento;	**Id** - O identificador único do documento a ser alterado; **campo** - Nome do campo que será atualizado; **novo_valor** - Valor atualizado

Espera-se que o desenvolvimento desta API facilite o uso do banco de dados por parte dos pesquisadores dos projetos, que poderão utilizar uma interface mais intuitiva do que a da API padrão.

Referências

BANKER, K. **MongoDB in action - manning**. Shelter Island: Manning, c2012. 287 p.

BRITO, R. W. **Bancos de Dados NoSQL x SGBDs relacionais**: análise Comparativa. Fortaleza, [S.n.], 2010. 6 p.

DE DIANA, M.; GEROSA, M. A. NOSQL na Web 2.0: um estudo comparativo de Bancos não-relacionais para gerenciamento de dados na Web 2.0. In: WORKSHOP DE TESES E DISSERTAÇÕES EM BANCO DE DADOS, 9.,2010, Belo Horizonte. **Anais**... Belo Horizonte, [S.n.], 2010.

MONGODB. 2013. Disponível em: <http://www.mongodb.org/>. Acesso em: 18 set. 2013.

REDMOND, E.; WILSON, J. R. **Seven databases in seven weeks**: a guide to modern databases and the NoSQL movement. Dallas: Pragmatic Bookshelf, 2012. 333 p. (Series Pragmatic