Revista Brasileira de Cartografia (2013) N^0 65/3: 479-492
Sociedade Brasileira de Cartografia, Geodésia, Fotogrametria e Sensoriamento Remoto
ISSN: 1808-0936

BANCOS DE DADOS GEOGRÁFICOS E SISTEMAS NOSQL: ONDE ESTAMOS E PARA ONDE VAMOS

Geographic Databases and NoSQL: Accomplishments and Future Directions

Gilberto Ribeiro de Queiroz, Antônio Miguel Vieira Monteiro & Gilberto Câmara

Instituto Nacional de Pesquisas Espaciais - INPE
Divisão de Processamento de Imagens – DPI
Avenida dos Astronautas, 1758. Jd. Granja - CEP 12227-010. São José dos Campos, SP, Brasil
{gribeiro, miguel, gilberto}@dpi.inpe.br

Recebido em 19 de julho, 2012/ Aceito em 01 de setembro, 2012
Received on july 19, 2012/ Accepted on september 01, 2012

RESUMO

A integração da tecnologia de sistemas de informações geográficas (SIGs) com os sistemas de bancos de dados relacionais (SGBD-R) e a busca por padrões de interoperabilidade entre sistemas geoespaciais foram assuntos que dominaram grande parte da agenda da academia, da indústria e da comunidade de usuários de dados geográficos em geral nas duas últimas décadas do século XX. Como resultado, presenciou-se o surgimento dos serviços geográficos na *web*, do suporte geoespacial integrado em quase todos os SGBD-R e o começo da criação das grandes infraestruturas de dados espaciais. No entanto, desde meados dos anos 90, no campo das tecnologias de banco de dados, um grande debate científico internacional vem crescendo sobre novos sistemas que sejam mais adequados às novas demandas de aplicações não-convencionais. Essas tecnologias nasceram com o uso extensivo da *internet* e a grande capacidade de capturar dados e informações em formato digital através de uma miríade de dispositivos e novos sensores, tanto na área das ciências como nas relações em sociedade, através dos dispositivos móveis de comunicação multimídia. Ciências como a Astronomia, as novas questões apresentadas a partir de vários recortes disciplinares distintos em um contexto de mudanças climáticas e ambientais e as novas demandas de um mundo fortemente conectado através da informação em rede e em tempo quase-real construíram novos desafios para o armazenamento e recuperação de informações, não imaginados quando os conceitos e tecnologias de SGBD-R foram estabelecidos e os sistemas para lidar com volumes de dados geográficos apareceram. Este trabalho procura apresentar uma revisão crítica da literatura de bancos de dados geográficos e das tecnologias de armazenamento desenvolvidas ao longo deste período, incluindo os sistemas NoSQL, uma nova classe de sistemas não-relacionais que vem sendo empregada em diversos domínios de aplicação. Esta revisão mostra como estas novas estratégias podem suprir lacunas entre os sistemas relacionais tradicionais e as necessidades das novas aplicações comerciais com forte uso da *internet* e das novas demandas da comunidade científica em diversos campos do conhecimento, em particular para os estudos dos processos no domínio geográfico.

Palavras chaves: SIG, SGBD-R, NoSQL, Bancos de Dados Matriciais.

ABSTRACT

The integration of geographical information systems (GIS) with relational databases (RDBMS) and the search for interoperable standards among geospatial systems were two major issues that dominated most of the agenda of academia, industry, and the users´ community of spatial data in general in the last two decades of the twentieth century. As a result, we witnessed the emergence of geographic web services, the integration of geospatial support in almost all RDBMS and the advent of large spatial data infrastructures. However, since the mid 1990s, in the field of database technologies, a new international scientific debate has been growing about new systems that are more responsive to new demands, mainly for non-conventional applications. These technologies were born with the extensive use of the internet and the great ability of gathering data and information in digital format through a myriad of new devices and sensors, in science and in the social field, through spatially aware mobile appliances. Science fields, such as Astronomy, reformulated questions from several different views in the context of climatic and environmental changes. New demands of a world strongly connected through a network information in near real time built new challenges for storage and retrieval of information. These advances were unimaginable when the DBMS-R concepts and technologies were established and the systems to deal with geographic data appeared. This paper presents a critical review of the literature of geographic databases and storage technologies developed over this period, including the NoSQL systems, a new class of non-relational systems that has been employed in several application domains. This review shows how these new strategies can fill gaps between the traditional relational systems and the needs of new commercial applications with heavy use of internet and the new demands of the scientific community in various fields of knowledge, particularly those linked to the study of geographical processes.

Keywords: GIS, RDBMS, NoSQL, Array Databases.

1. INTRODUÇÃO

Tradicionalmente, os sistemas gerenciadores de bancos de dados relacionais (SGBD-R) têm sido empregados para o armazenamento e gerenciamento de dados nos mais diversos tipos de aplicações (ELMASRI e NAVATHE, 2006). No entanto, somente na última década, estes sistemas se tornaram capazes de lidar de maneira ampla e eficiente com dados geográficos. A partir desta integração, a tecnologia relacional passou a fazer parte do cotidiano de usuários de SIGs, pois grande parte das ferramentas passou a ser capaz de acessar dados armazenados nos SGBD-R.

Neste mesmo período, houve uma grande preocupação com a padronização do suporte espacial incluído nos SGBD-R. Padrões abertos, como a *Simple Feature* (OGC, 2012a; OGC, 2012b) e a *SQL/MM* (MELTON e EISENBERG, 2001), foram amplamente adotados pelos fabricantes de tecnologia relacional. Como benefício, grande parte dos SIGs se tornou interoperável no nível de armazenamento.

Em paralelo a este desenvolvimento, começaram a surgir novas tecnologias de bancos de dados em resposta à crescente demanda por um gerenciamento mais eficaz de grandes volumes de dados, assim como à necessidade de realização de análises sobre os mesmos. Estes novos sistemas, denominados *NoSQL*, possuem raízes no uso extensivo da *internet*, na grande capacidade de capturar dados e informações em formato digital através de uma miríade de dispositivos e novos sensores e na profusão de dispositivos móveis de comunicação com capacidade multimídia e localização espacial. Esta tecnologia está começando a despertar a atenção de pesquisadores e profissionais da área de Geoinformática (KERR, 2009; BAPTISTA et al., 2011; MALONE, 2010; WANG e WANG, 2010).

É neste contexto que este trabalho apresenta como essas novas tecnologias têm sido utilizadas para suprir lacunas entre os sistemas relacionais tradicionais e as necessidades de aplicações com forte uso da *internet* bem como das novas demandas da comunidade científica em diversos campos do conhecimento, em particular para os estudos dos processos no domínio geográfico. Para isso, a Seção 2 apresenta uma breve revisão da tecnologia relacional. A Seção 3 mostra os resultados da integração entre as tecnologias SIG e SGBD-R. A Seçaõ 4 aborda em mais detalhes os principais sistemas NoSQL que têm sido empregados nas grandes organizações ligadas à *internet* bem como os que estão sendo desenvolvidos para atender demandas de comunidades científicas, como a de Sensoriamento Remoto. A Seção 5 discute o estágio destas tecnologias em relação aos sistemas relacionais para

o caso de dados espaciais. E, por último, na Seção 6, serão fornecidas algumas direções de como endereçar problemas de armazenamento de dados matriciais e atender à expectativa de novas aplicações espaciais com ênfase em modelagem computacional.

2. SISTEMAS RELACIONAIS

Os Sistemas Gerenciadores de Bancos de Dados Relacionais (SGBD-R) apoiam-se em uma fundamentação teórica sólida introduzida por Codd (1970), denominada de *Modelo Relacional de Dados*. Um dos principais objetivos deste modelo é prover independência física dos dados para as aplicações, de forma que estas não tenham que obrigatoriamente conhecer detalhes de como os dados encontram-se organizados no meio de armazenamento. Esta independência é obtida através da manutenção do esquema do banco de dados, isto é, de metadados contendo definições de tabelas, visões e índices, junto com o mapeamento destes elementos no meio de armazenamento.

A existência de esquemas para as tabelas do banco de dados é uma forte característica dos SGBD-R. Para cada tabela, é mantido um registro do seu nome e informações de suas colunas. Cada coluna possui um nome e o tipo de dados dos valores que podem estar presentes nas linhas destas colunas. Consequentemente, as linhas de uma mesma tabela compartilham a mesma definição, ou seja, os mesmos tipos de atributos.

Além do modelo relacional, várias funcionalidades ajudaram a popularizar este tipo de sistema, com destaque para:

a) métodos de indexação baseados em *árvores-b* (COMER, 1979) ou tabelas *hash* (CORMEN et al., 1990), capazes de acelerar a execução de consultas e operações de atualização;

b) existência de uma linguagem declarativa de alto nível para consulta e manipulação de dados, como a *SQL* (MELTON, 2003);

c) um modelo de programação baseado no conceito de transações que correspondem a unidades de trabalho, as quais podem agrupar uma ou mais operações e cuja execução deve ser realizada de forma atômica, satisfazendo algumas propriedades conhecidas por ACID: *atomicidade, consistência, isolamento e durabilidade* (ELMASRI e NAVATHE, 2006). A *atomicidade* garante que todas as operações encapsuladas pela transação sejam realizadas, ou, no caso de falhas, o efeito de nenhuma delas seja aplicado sobre o banco. A *consistência* garante que, ao final da execução de uma transação, as regras de integridade do banco sejam respeitadas. O *isolamento* garante que o resultado da execução de transações concorrentes seja o mesmo de quando executadas de forma isolada (seriais). A *durabilidade* garante que, após a confirmação de uma transação, os resultados sejam refletidos no banco de dados, mesmo se ocorrerem falhas após a confirmação de sua execução.

d) mecanismos de replicação de dados geralmente baseados em um modelo mestre/escravo, no qual consultas de leitura podem ser submetidas a qualquer nó, e de escrita, apenas no nó mestre.

3. SIG E SISTEMAS RELACIONAIS

Conforme mostrado em Casanova et al. (2005), várias tentativas de integração entre estas duas tecnologias ocorreram sem sucesso antes do final dos anos 90. Até então, o que se tinha eram SIGs que exploravam o uso de campos binários longos (BLOBs) ou esquemas de tabelas relacionais para armazenar a componente espacial do dado geográfico. Tais estratégias não foram bem sucedidas por várias razões, entre elas a falta de mecanismos eficientes de indexação espacial (GAEDE e GÜNTHER, 1998) e a incapacidade de uso da linguagem SQL para manipulação direta da componente espacial armazenada no banco.

Somente no final dos anos 90 e meados dos anos 2000, com a inclusão de tipos de dados, operadores e mecanismos de indexação espaciais nos SGBD-R, é que estas tecnologias começaram a apresentar maior sinergia. O setor comercial desenvolveu o *Oracle Spatial* (ORACLE, 2003), o *IBM DB2 Spatial Extender* (IBM, 2002) e, mais recentemente, a Microsoft introduziu suporte espacial ao *SQL Server* (MICROSOFT, 2008). A comunidade de software livre também criou extensões, como o *PostGIS* para o *PostgreSQL* (OBE et al., 2011) e o *SpatiaLite* para o *SQLite* (FURIERI, 2012).

O suporte espacial dos atuais SGBD-R é fortemente baseado em padrões abertos, como a *Simple Feature* (OGC, 2012a; OGC, 2012b) e a *SQL/MM* (MELTON e EISENBERG, 2001). Estas

especificações abrangem basicamente o modelo geométrico e as operações espaciais que devem ser suportadas pelas implementações. O modelo geométrico contém representações para pontos, linhas, polígonos, coleções homogêneas e heterogêneas de geometrias, como mostrado na Figura 1. Os operadores topológicos seguem o paradigma da matriz de 9-interseções estendida dimensionalmente, proposta por Clementini e Di Felice (1995).

Com a introdução dos tipos geométricos nos SGBD-R, tornou-se possível a criação de tabelas em que uma das colunas é capaz de armazenar geometrias. O exemplo mostrado na Figura 2 usou coleções homogêneas de polígonos para representar as áreas de cada unidade federativa. Cada objeto armazenado desta forma possui todos os pares de coordenadas da fronteira. Consequentemente, os relacionamentos espaciais são computados durante a execução das consultas, não sendo armazenados de forma explícita como em modelos topológicos.

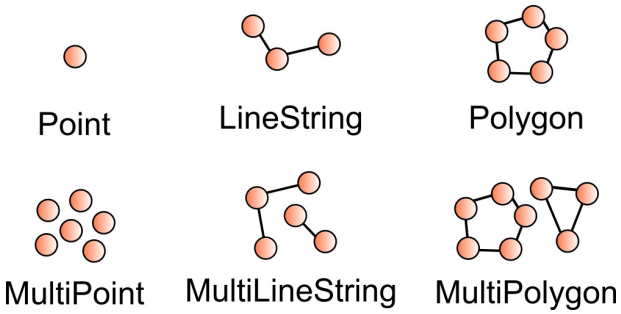

Fig.1 - Tipos Geométricos Básicos da SFS-SQL.

Tabela: unidades_federativas		
gid	sigla	geometria
1	MG	
2	SP	
3	RJ	

Fig.2 - Exemplo de tabela com coluna geométrica.

A inclusão de operadores espaciais possibilitou a criação de consultas SQL como a mostrada na Figura 3, em que a região com a aptidão agrícola do município de João Pinheiro no Estado de Minas Gerais é calculada a partir de dados armazenados em duas tabelas, uma com o mapa de municípios do Estado de Minas Gerais e outra com o de aptidão agrícola deste Estado.

O PostGIS, o Oracle Spatial e o SpatiaLite optaram por fornecer mecanismos de indexação espacial baseados nas *árvores-r* (GUTTMAN, 1984). Outros, como o SQL Server e o IBM DB2 Spatial Extender, utilizam uma estratégia baseada em *grades fixas multiníveis* (NIEVERGELT et al., 1984) e *space filling curves* (LAWDER e KING, 2000), de forma a não terem a necessidade de criar novas implementações de seus mecanismos de indexação sobrejacentes (*árvores-b*).

Cada vez mais, novas funcionalidades geoespaciais são integradas aos SGBD-R. Uma delas é o armazenamento e gerenciamento de dados na forma matricial (*raster*). Como exemplo, a extensão *PostGIS Raster* (OBE et al., 2011) lançada oficialmente em 2012, possibilita armazenar no banco de dados imagens de sensoriamento remoto. Esta extensão disponibiliza diversas estratégias de armazenamento da imagem, entre elas: (a) divisão da imagem em pequenos blocos, denominados *tiles*, que são armazenados em tabelas (Figura 4), e (b) armazenamento de referências (*links*) para as imagens gravadas em arquivos fora do banco. Também é possível criar pirâmides (ou *overviews*) para acelerar a visualização por parte das aplicações. O conjunto de operadores e funções presentes nesta extensão permite a realização de análises complexas nas imagens, assim como operações envolvendo dados vetoriais e matriciais como o recorte de uma imagem a partir de uma máscara vetorial.

Outras funcionalidades que vêm sendo incluídas aos SGBD-R são voltadas para representações topológicas. Extensões como o

```
SELECT m.nommuni, a.classe,
       ST_Intersection(m.the_geom,
                       a.the_geom)
  FROM mg_municipios m,
       mg_aptidao_agricola a
 WHERE ST_Intersects(m.the_geom,
                     a.the_geom)
   AND m.nommuni = 'João Pinheiro'
```

Fig.3 - Consulta espacial em SQL.

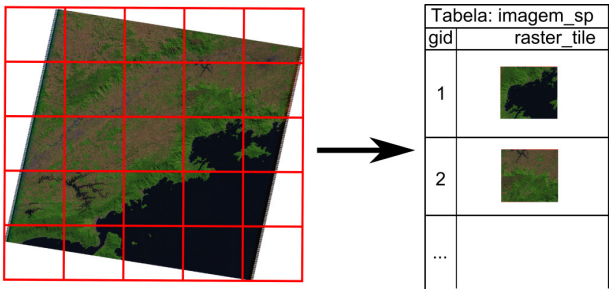

Fig.4 - Imagem armazenada em uma tabela.

PostGIS Topology introduzem tipos e operadores para trabalhar com os conceitos de nós, arestas e faces. Neste tipo de representação, os pontos das fronteiras dos objetos são armazenados uma única vez, sendo compartilhados entre as faces. Este suporte pode ser usado, entre outras coisas, para garantir a correta topologia das áreas geradas como resultado de processos de simplificação.

Por último, existem extensões dos SGBD-R para lidar com representações de rede. Apesar da existência da extensão *pgRouting* para o PostgreSQL, apenas o *Oracle Spatial Network Data Model* (ORACLE, 2008) possui um modelo mais amplo e com operadores capazes de atender a um maior domínio de aplicações geoespaciais.

Do exposto até aqui, observa-se que a tecnologia relacional integrou várias funcionalidades que antes eram encontradas apenas nos SIGs. Em consequência disso, os SGBD-R têm se mostrado uma solução eficiente para o gerenciamento de dados geoespaciais. Já a padronização deste suporte teve o efeito positivo de possibilitar o desenvolvimento de uma geração de aplicativos geográficos interoperáveis e capazes de utilizar os SGBD-R.

Atualmente, as principais bibliotecas de acesso a dados geoespaciais baseadas em software livre, GDAL (WARMERDAM, 2010), GeoTools (TURTON, 2010) e TerraLib (CÂMARA et al., 2010), possuem uma boa interface com sistemas relacionais. Aplicativos desktop, como o Quantum GIS (QUANTUM GIS DEVELOPMENT TEAM, 2012) e TerraView (CÂMARA et al., 2010), também são capazes de visualizar e analisar dados destes sistemas. O GeoServer (AIME, 2007), MapServer (KROPLA, 2005) e TerraOGC (CÂMARA et al., 2010) disponibilizam serviços *web* geográficos a partir de bases de dados relacionais.

Contudo, tem-se observado na última década o surgimento de novas tecnologias de armazenamento de dados com modelos e recursos bem diferentes dos sistemas relacionais tradicionais. Trata-se de um tópico emergente e pouco explorado no universo de SIGs.

4. NOVAS TECNOLOGIAS DE BANCOS DE DADOS: SISTEMAS NoSQL

Novas tecnologias de armazenamento, gerenciamento e processamento de grandes volumes de dados têm sido criadas por dois grupos distintos: a dos gigantes da *internet*, como *Google*, *Amazon* e *Facebook*, e a dos grupos de pesquisa em bancos de dados associados a comunidades científicas, principalmente, de *Astronomia* e *Sensoriamento Remoto*.

O primeiro grupo tem se preocupado com a criação de sistemas para atender aplicações *web* que lidam com grande número de usuários, como máquinas de busca, redes sociais e demais aplicações da *Web* 2.0. Nestas aplicações, existe a necessidade de realizar análises sobre petabytes de dados provenientes de fontes não estruturadas, como documentos HTML, PDF, arquivos texto, mensagens eletrônicas, entre outras.

O segundo grupo tem buscado criar sistemas mais adequados à comunidade científica, introduzindo novos modelos de dados e linguagens de consultas baseados em matrizes (*arrays*). Trata-se de uma tentativa de resposta à crescente demanda deste meio, que, com os avanços nos equipamentos de coletas de dados, como telescópios, sensores a bordo de satélites, Geo-Sensores e GPS, tem enfrentado dificuldades para o armazenamento, processamento e análise de quantidades massivas de dados. Alguns autores como Berriman e Groom (2011) já falam em tsunami de dados.

Esses novos sistemas estão sendo denominados de *NoSQL*, cuja tradução mais aceita no momento é *Not only SQL* ou *sistemas pós-relacionais* (POKORNY, 2011). Grande parte deles não utiliza a SQL como linguagem de consulta e, ao contrário dos sistemas relacionais, não há a necessidade de criação de esquemas rígidos para armazenamento dos dados.

Os sistemas que surgiram motivados pelos trabalhos publicados pela Google (GHEMAWAT et al., 2003; DEAN e GHEMAWAT, 2008; CHANG et al., 2008) funcionam de forma

distribuída e são altamente tolerantes à falhas. Estes sistemas encontram-se em operação em grandes centros de dados onde é comum a ocorrência de falhas em alguns nós devido a problemas de disco, CPU ou até mesmo de comunicação de rede. Em consequência, eles possuem recursos para replicação de dados em várias máquinas, aumentando a disponibilidade do sistema: quando um nó cai, outro passa a responder.

Os NoSQL estão sendo projetados desde o início para atender a uma maior *escalabilidade horizontal*, isto é, fazer com que a adição de novas máquinas ajudem a melhorar o desempenho do sistema de forma linear. Este tipo de escalabilidade contrasta com a *vertical*, onde um servidor precisa ser substituído por outro mais potente para aumentar a capacidade computacional do sistema. Desta forma, a primeira forma de escalabilidade possui menor custo de expansão.

Alguns sistemas NoSQL, como o *Cassandra* (LAKSHMAN e MALIK, 2010) e o *Dynamo* (DECANDIA et al., 2007), são descentralizados, ou seja, os dados encontram-se replicados por várias máquinas, e cada uma delas é autônoma, sendo capaz de responder a qualquer consulta. Este tipo de estratégia aumenta a disponibilidade do sistema, uma vez que não existe um único ponto capaz de comprometer o seu funcionamento. Esta é uma característica que contrasta com a maior parte das implementações relacionais, onde, geralmente, é empregado um modelo mestre/escravo. Como neste modelo a escrita é limitada ao mestre, esta última forma de replicação apresenta duas desvantagens imediatas: (a) maior susceptibilidade a falhas, uma vez que a inoperância do nó mestre pode deixar o sistema indisponível, e (b) maior dificuldade para obter escalabilidade horizontal, pois o nó mestre pode se tornar um gargalo para o sistema.

Outro ponto de contraste entre os NoSQL e os sistemas relacionais é o suporte a transações ACID. Muitos NoSQLs simplesmente não oferecem este recurso, considerado dispensável em algumas aplicações, principalmente pelo custo computacional envolvido para provê-las. Este é um dos pontos de muita polêmica neste debate entre NoSQL e sistemas relacionais, como mostrado por Stonebraker (2011).

Existe uma taxonomia dada aos NoSQL de acordo com o modelo de dados e a estratégia de armazenamento adotada. A seguir, detalha-se cada uma destas categorias.

4.1 Bancos de Dados Orientados a Documentos

Existem vários sistemas que se enquadram nesta família de NoSQL, também chamados de *Document Stores*. Destacam-se o *MongoDB* (CHODOROW e DIROLF, 2010) e o *Apache CouchDB* (ANDERSON et al., 2010), com modelos projetados para trabalhar com documentos semi-estruturados (sem um esquema pré-definido). Em geral, a representação dos documentos utiliza uma notação derivada da sintaxe *JSON* (CROCKFORD, 2006), acrônimo de *JavaScript Object Notation*, que é um formato leve para intercâmbio de dados baseado na linguagem *JavaScript* (ECMA INTERNATIONAL, 2011). Este formato se popularizou nas aplicações *web*. A capacidade de lidar com dados sem um esquema rígido combinado com eficientes sistemas de indexação tornam estes sistemas uma solução interessante para a construção de aplicativos da *Web* 2.0, como gerenciamento de conteúdos e redes sociais.

No *MongoDB*, um banco de dados é composto por coleções de documentos no formato *BSON*, uma versão binária dos documentos *JSON*. As coleções e documentos lembram, respectivamente, os conceitos de tabelas e linhas dos sistemas relacionais. A diferença encontra-se no fato de que os documentos de uma coleção não precisam obrigatoriamente ter o mesmo esquema. Assim, documentos de uma coleção podem ter campos muito diferentes uns dos outros (Figura 5).

No *CouchDB*, um banco de dados é formado por uma coleção independente de documentos expressos na notação *JSON*. Cada documento possui pelo menos dois atributos: um identificador único e um número de revisão usado para versionamento e resolução de conflitos durante operações de atualização. Assim como nos sistemas relacionais, é possível definir visões (*views*) sobre os dados dos documentos. As visões podem servir para filtrar documentos do banco de dados, para extrair partes mais específicas, para criar apresentações mais adequadas às aplicações e até mesmo como uma forma de criação de índices para acelerar a localização dos documentos. As visões são definidas através de funções de agregação

```
{
     "_id" : "001",
     "tipo": "hospital",
     "nome": "Santa Casa",
"endereco" : {
       "logradouro": "R. das Rosas",
           "numero": 95,
           "cidade": "Ouro Preto" }
},
{
        "_id": "999",
       "tipo": "foco_queimada",
   "long_lat": [-46.76, -20.54],
   "satelite": "GOES-12",
 "observacao": "2012-07-05 05:15:00"
}
```

Fig.5 - Coleção de documentos *JSON* para representação de pontos de interesse.

escritas em *JavaScript* que retornam pares de chave-valor. Assim, aplicações podem facilmente acessar individualmente uma informação ou recuperar todos os dados associados a uma visão.

O protocolo usado para leitura e atualização dos documentos é baseado em uma *API RESTful HTTP* (RICHARDSON e RUBY, 2008), isto é, a comunicação entre clientes e o servidor *CouchDB* é realizada através de requisições feitas no protocolo *HTTP* (FIELDING et al., 1999), usando métodos *GET*, *PUT*, *POST* e *DELETE*. Esta estratégia de comunicação é bem diferente das utilizadas na maioria dos sistemas de bancos de dados, que, geralmente, criam protocolos próprios com conexão via *TCP* (STEVENS, 1994). Outro ponto forte deste sistema é a sua capacidade de replicação de dados entre nós, que pode ser usada por sistemas que precisam manter a sincronização entre dispositivos móveis, computadores pessoais e servidores.

4.2 Armazenamento baseado em pares chave-valor

Também conhecidos por *Key-Value Stores*, os sistemas derivados do *DBM* (GNU, 2011), como *Oracle Berkeley DB* (OLSON et al., 1999), *Kyoto Cabinet* (FAL LABS, 2011) e *LevelDB* (DEAN e GHEMAWAT, 2012), são bibliotecas com rotinas para o gerenciamento de bancos de dados baseadas no armazenamento de pares chave-valor. A chave é usada para identificar um valor que pode ser um dado estruturado ou não. Estes sistemas oferecem vários métodos de acesso (B^+-tree, Hash, Heap, Queue) que são utilizados na organização primária dos registros do banco de dados, além de componentes configuráveis, como *cache* de dados, controle de concorrência, mecanismos de transação ACID, recuperação a falhas e até mesmo replicação. Em geral, podem ser embutidos nas aplicações, de forma a evitar gargalos de comunicação entre processos comuns nas arquiteturas cliente-servidor dos sistemas relacionais. Este modelo de biblioteca fornece um nível de programação apropriado para a criação de soluções especializadas de armazenamento. Um exemplo disso é o caso de modelos de dados complexos que não se enquadram bem no modelo relacional, como grafos e documentos não estruturados.

Tradicionalmente, estes sistemas têm sido amplamente utilizados de forma embarcada em dispositivos móveis (celulares) e roteadores de rede, em sistemas de autenticação de usuários e de gerenciamento de conteúdo. Outro exemplo de uso pode ser encontrado em Decandia et al. (2007), que descreve o sistema *Dynamo* desenvolvido pela *Amazon*, com o objetivo de suportar as demandas exigidas na manutenção de estado das suas aplicações: alta disponibilidade e escalabilidade.

4.3 Bancos baseados em Grafos

Estes sistemas se baseiam na Teoria dos Grafos (BOLLOBÁS, 1998) e são conhecidos como *Graph Databases*. Fornecem três construtores básicos em seu modelo de dados: nós (ou vértices), ligações (ou arestas) e propriedades. Os nós são usados para modelar objetos que existem de forma independente das ligações. Em geral, não possuem um esquema rígido. Assim, dois objetos representados por nós de um mesmo grafo podem ter atributos bem distintos. As arestas são usadas para materializar o relacionamento entre nós. As propriedades podem ser usadas tanto para descrever atributos de nós quanto de arestas.

Este modelo de dados tem se tornado popular em aplicações *web* de domínio social, fornecendo capacidades interessantes para a realização de consultas requeridas por este tipo de aplicação e com um bom desempenho. As implementações desta tecnologia, como o *OrientDB* e o *Neo4J* (HUNGER, 2010), optaram por integrar uma linguagem para travessia de grafos denominada de *Gremlin* (AVRAM, 2010), capaz de expressar travessias complexas nos grafos de forma concisa, ao invés do uso estrito da SQL.

Um bom exemplo de aplicação deste tipo de tecnologia foi dado pelo *Twitter*, que criou um sistema de bancos de dados batizado de *FlockDB* (TWITTER, 2010) para o gerenciamento eficiente de listas de adjacência que materializam as listas de seguidores de uma pessoa e de quem a está seguindo. O FlockDB possibilita uma rápida recuperação das pessoas para as quais uma mensagem (*tweet*) deve ser transmitida ou dos seguidores comuns a determinadas pessoas.

Segundo os engenheiros deste sistema, várias alternativas baseadas em sistemas relacionais foram experimentadas antes de criar uma solução personalizada como esta. Porém, todas acabavam esbarrando em alguns problemas ou dificuldades. Segundo o próprio site, em 2010, ele era usado para manter mais de 13 bilhões de ligações, sustentando taxas de 20 mil operações de escrita por segundo e 100 mil de leitura por segundo.

4.4 Armazenamento Orientado a Colunas

Um dos sistemas NoSQL que se enquadram nesta classe (*Column Store*) é o *Apache Cassandra* (LAKSHMAN e MALIK, 2010), desenvolvido inicialmente pelo *Facebook* e atualmente disponível como um projeto de software livre da fundação Apache. O modelo de dados do Cassandra inclui o conceito de *keyspace*, algo similar ao conceito de banco de dados nos sistemas relacionais. Analogamente às tabelas do modelo relacional, existe o conceito de famílias de colunas (*column families*), usadas para agrupar dados com estruturas similares. As colunas são formadas por pares *(nome da coluna, valor)*. Grupos de colunas associadas a uma mesma entidade ou objeto são identificados através de uma chave (*key*). Esses grupos de colunas formam uma linha (*row*), mas ao contrário dos sistemas relacionais onde todas elas possuem a mesma estrutura, em sistemas colunares como o Cassandra, as linhas podem ter pares bem diferentes umas das outras. Desta forma, um valor é identificado por uma tripla: <*nome da família de colunas, chave, nome da coluna*>.

Outra abstração fornecida no modelo do Cassandra são as chamadas super famílias de colunas (*super column families*), usadas para criar subgrupos de colunas. Uma supercoluna (*super column*) funciona como um índice para outras colunas, e assim os valores passam a ser identificados através de uma quadra: <*nome da família de colunas, chave, nome da supercoluna, nome da subcoluna*>.

O Cassandra faz uso de implementações de tabelas *hash* distribuídas, em que a estrutura de índice leva em consideração a replicação e particionamento dos dados em várias máquinas (nós), além da existência de diversos centros de dados (*data centers*).

4.5 Bancos de Dados Matriciais

Como alertado por Jim Gray e colegas (GRAY et al., 2005), sempre houve uma lacuna entre as tecnologias de bancos de dados relacionais e a comunidade científica. Entre os vários pontos deste descasamento, o modelo de dados e a linguagem de consulta se mostraram grandes barreiras para adoção desta tecnologia, pois os dados científicos geralmente se encontram na forma de matrizes multidimensionais.

Em consequência disso, vários grupos optaram por construir seus próprios sistemas de armazenamento e processamento de dados. Um exemplo desta prática é mostrado em Raoult (2012), que apresenta a arquitetura do *MARS*, um sistema desenvolvido pelo *Centro Europeu de Previsão a Médio Prazo* (ECMWF) para lidar com mais de 30.000 requisições diárias, acessando cerca de 1.500.000 imagens, correspondentes a mais de 100GBytes de dados movidos para processamento todos os dias.

No entanto, mais recentemente, um consórcio formado por diversos pesquisadores da área de bancos de dados e representantes das comunidades científicas de Astronomia e Sensoriamento Remoto criaram um sistema chamado *SciDB* (CUDRE-MAUROUX et al., 2009; BROWN, 2010). Voltado para as demandas destas comunidades, este sistema fornece um modelo de dados e uma linguagem de consulta baseados em matrizes multidimensionais (Figura 6). Este modelo tem dado origem à nomenclatura de *Array Databases*.

O SciDB usa uma estratégia de armazenamento, onde a matriz é quebrada em blocos que podem ser distribuídos em diversas máquinas para acelerar a realização de análises em matrizes muito grandes. Além disso, os blocos podem replicar partes das bordas para favorecer a computação de vizinhança. Estratégias de compactação com algoritmos que fazem um balanceamento entre capacidade de compressão e

J \ I	[0]	[1]	[2]	[3]
[0]	("A", 29)	("K", 32)	("B", 31)	("C", 32)
[1]	("F", 30)	("D", 31)	("E", 30)	("U", 32)
[2]	("O", 32)	("V", 32)	("L", 32)	("X", 32)
[3]	("R", 33)	("S", 33)	("Q", 32)	("T", 32)

Fig. 6 - Definição de uma matriz 4 x 4 com 2 atributos.

uso de CPU podem ser aplicadas a esses blocos. Os valores de cada célula são armazenados em arquivos separados, pois é comum, nas aplicações científicas, matrizes com células com grande número de valores e modelos, e processos que usam apenas alguns deles. Esta separação também tem como objetivo obter um melhor resultado de compressão.

5. NoSQL E DADOS GEOGRÁFICOS: ONDE ESTAMOS

Atualmente, alguns sistemas NoSQL estão introduzindo suporte ao armazenamento e recuperação de dados espaciais. A Tabela 1 apresenta um quadro comparativo entre os recursos existentes nos SGBD-R e as várias classes de sistemas NoSQL. Nesta tabela, usamos o PostgreSQL com sua extensão geográfica PostGIS como referência para as funcionalidades espaciais atualmente encontradas em sistemas relacionais baseados em software livre.

Como pode ser observado no quadro, o *MongoDB* possui índices espaciais para dados representados por pontos. Este mecanismo permite a realização de consultas de localização, como os *N* vizinhos mais próximos a um determinado ponto, e de consultas por intervalo, utilizando-se um retângulo de busca, um círculo ou polígono simples como filtro. A implementação deste mecanismo é bem diferente das fornecidas pelas extensões espaciais dos SGBD-R, usando-se técnicas de *hash* sobre as coordenadas geográficas dos pontos. Cada coleção de documentos está limitada a no máximo um índice espacial, que pode conter atributos extras colocados juntamente com a estrutura para acelerar consultas que envolvam tanto um predicado espacial quanto alfanumérico. É de conhecimento dos autores a existência apenas de um *plugin* para o aplicativo Quantum GIS chamado *MongoDBLayer* (MARKUS, 2012), que possibilita a visualização de dados armazenados no *MongoDB*.

O *CouchDB* encontra-se integrado nas bibliotecas GDAL e GeoTools. Também existe uma extensão espacial chamada *GeoCouch* (Thompson, 2011), que facilita a manipulação de documentos no formato *GeoJSON* (BUTLER et al., 2012), o qual é uma especialização da notação *JSON*, capaz de codificar geometrias, feições e coleções de feições. As geometrias suportadas são um subconjunto previsto na especificação OGC da *Simple Feature*. Esta extensão ainda acrescenta um mecanismo de indexação espacial.

Os primeiros sistemas com dados geoespaciais desenvolvidos sobre estas duas tecnologias de *Document Stores* lidam basicamente com o armazenamento de pontos de interesse (POI). O *CouchDB* tem sido usado na construção de servidores que funcionam como uma *cache* de *tiles* de imagens, comuns em aplicações de mapas na *web* ao estilo do Google Maps.

No caso dos *Graph Databases*, representados no quadro pelo Neo4J, existe uma biblioteca chamada *Neo4J Spatial* (NEUBAUER, 2011), que acrescenta métodos de indexação espacial e operadores espaciais a este sistema. Esta biblioteca funciona integrada ao GeoTools, possibilitando o acesso a dados armazenados no Neo4J por aplicações como o GeoServer e uDIG.

Em relação aos *Column Stores*, representados pelo *Apache Cassandra*, não é de conhecimento dos autores ferramentas de software livre para SIGs que possuem algum tipo de interface com estes sistemas. Alguns trabalhos iniciais têm tentado explorar o *Cassandr*a para a criação de uma infraestrutura de armazenamento de dados espaciais (MALONE, 2010). O fato é que existe uma série de questões abertas para a integração desta família de NoSQL em aplicações geográficas. Uma delas é conseguir atender de forma satisfatória consultas espaciais de intervalo e vizinhança, uma vez que estes sistemas são fortemente baseados em tabelas *hash* distribuídas, eficientes para consultas com casamento exato da chave de busca.

Os *Array Databases*, representados pelo SciDB, possuem características interessantes para armazenamento e análise de dados matriciais. Dentro desta classe de NoSQL, existe outro sistema chamado *Rasdaman* (BAUMANN et al., 1998), que também oferece uma linguagem baseada em

Tabela 1: Suporte espacial dos SGBD-R e NoSQL.

	PostgreSQL/ PostGIS	MongoDB	CouchDB/ GeoCouch	Neo4J/ Spatial	Apache Cassandra	SciDB	LevelDB, Berkeley DB
Classe	Relacional	NoSQL Orientado a Documentos	NoSQL Orientado a Documentos	NoSQL Baseado em Grafos	NoSQL Orientado a Colunas	NoSQL Orientado a Matrizes	NoSQL Baseado em Chave-Valor
linguagem consulta	SQL	x	x	SQL, Gremlin	x	AQL, AFL	x
transações ACID	x	x	x	✓	x	x	✓
tipos geométricos	SFS	Pontos 2D	SFS	SFS	x	x	x
operadores espaciais	✓	x	x	x	x	x	x
indexação espacial	árvore-r	hash	árvore-r	árvore-r	x	x	x
projeções cartográficas	✓	x	x	x	x	x	x
armazenamento matricial	✓	x	x	x	x	✓	x
operadores para dados matriciais	✓	x	x	x	x	✓	x
representação topológica	✓	x	x	x	x	x	x
representação de redes	x	x	x	✓	x	x	x
integração com bibliotecas de acesso a dados geoespaciais	GDAL/OGR, GeoTools, TerraLib	x	GDAL/OGR, GeoTools	GeoTools	x	x	x
aplicativos geográficos	QGIS, uDIG, gvSIG, Kosmo, TerraView	QGIS (Experimental)	x	uDIG	x	x	x
serviços geoespaciais	MapServer, GeoServer, TerraOGC	x	x	x	x	x	x

matrizes multidimensionais. Este último tornou-se um projeto de software livre encubado pela fundação OSGEO. A importância desta classe de sistemas é reforçada com novos trabalhos que vêm apresentando propostas de criação de outras linguagens de consultas baseadas em matrizes, como o *SciQL* (KERSTEN et al., 2011).

6. CONSIDERAÇÕES FINAIS: PARA ONDE VAMOS

O cenário atual das tecnologias de armazenamento e gerenciamento de dados está passando por um momento especial em que novos sistemas com novas arquiteturas e características começam a emergir. De certa forma, estamos presenciando uma fragmentação das tecnologias de bancos de dados em nichos cada vez mais específicos de aplicações (STONEBRAKER, 2005).

Dentre os vários nichos de tecnologia de armazenamento, consideramos os *Array Databases* a mais próxima e promissora para uma classe de problemas no domínio geoespacial. A forma de representação de dados destes sistemas é capaz de atender a vários tipos de usos do dado geográfico, incluindo a criação e manutenção de centros de dados de imagens de Sensoriamento Remoto e o uso como plataforma analítica de grandes bases de dados matriciais.

Tecnologias como o SciDB e o *Rasdaman* podem representar uma boa alternativa, como *backend* de serviços *web*, a exemplo do *Web Coverage Service* (*WCS)* e do *Web Coverage Processing Service* (*WCPS)* (OGC, 2012c; OGC, 2012d), que envolvem processamento em imagens do lado servidor. Este é um caso de uso prático para a construção de infraestruturas de dados espaciais que necessitem fornecer serviços de processamento a seus clientes.

Outras aplicações com forte interação com os SIGs que também podem explorar capacidades específicas das tecnologias NoSQL são as ferramentas para modelagem e simulação de dinâmicas espacialmente explícitas. Estas ferramentas trabalham com grandes volumes de dados socioambientais para a produção de cenários de apoio a tomada de decisões. A maioria delas utiliza as idéias de uma *geografia celular*, conceito proposto por Tobler (1979) e de mundos celulares discutido por Couclelis (1985), exploradas a partir de variações de estrutura de dados matriciais e estruturas híbridas, conhecidas como espaço celular, para representar o espaço geográfico através de um conjunto de células conectadas por relações de vizinhança.

Novamente, o modelo fornecido pelos *Array Databases* se encaixa bem a este tipo de representação. As matrizes multidimensionais são

capazes de suportar a criação de espaços celulares com vários atributos, incluindo valores numéricos, textuais e datas. O armazenamento especializado, com blocagem, compressão e replicação das bordas, possibilita explorar a travessia das células e da vizinhança de forma eficiente. A distribuição de partes da matriz por várias máquinas torna possível a obtenção de um alto grau de paralelismo nas operações de leitura e escrita das células.

Contudo, no caso das ferramentas de simulação computacional, onde é comum a realização de processamentos de forma iterativa sobre as matrizes, é preciso pensar em mecanismos que diminuam a necessidade de transferência de dados entre o servidor e a ferramenta de simulação a cada iteração do modelo. Uma possibilidade de contornar esta necessidade seria a escrita dos modelos em uma das linguagens de programação de alto nível suportadas pelos *array databases*. Com isso, todo o processamento se daria dentro do espaço de endereçamento do próprio banco, evitando potenciais gargalos de comunicação e transferência de dados.

Por último, ainda existem situações que demandam a criação de formas de armazenamento mais especializadas. Neste caso, os sistemas NoSQL da classe dos *Key-Value Stores* podem servir como um bom ponto de partida. No contexto das aplicações de modelagem baseadas na biblioteca TerraLib, estes sistemas NoSQL vêm sendo experimentados como alternativa para o armazenamento de dados de espaços celulares. O objetivo desse trabalho é prover uma solução que possibilite aos pesquisadores o uso de máquinas *desktop* convencionais para a condução de seus experimentos mediante volumes consideráveis de dados espaciais. Como benefício, isso possibilitaria a realização de ensaios mais elaborados sem a necessidade de se ter que lançar mão de ambientes computacionais mais complexos, como o de *clusters* de computadores.

Enfim, com tantas tecnologias de bancos de dados disponíveis e com a crescente evolução das mesmas, principalmente em relação aos sistemas matriciais, é de se esperar uma grande evolução no suporte espacial dessas tecnologias. Consequentemente, avanços significativos deverão ocorrer nas ferramentas de modelagem computacional em relação ao volume de dados que estas serão capazes de lidar.

AGRADECIMENTOS

Os autores gostariam de agradecer ao Instituto Nacional de Pesquisas Espaciais pelo apoio em suas pesquisas.

REFERÊNCIAS BIBLIOGRÁFICAS

AIME, A. **GeoServer, past, present and future**. 2007. Disponível em: <http://2007.foss4g.org/presentations>. Acesso: 01 junho 2012.

ANDERSON, J. C.; LEHNARDT, J.; SLATER, N. **CouchDB: The Definitive Guide**. O'REILLY, 2010. 272p.

AVRAM, A. Gremlin, a Language for Working with Graphs. **Info Q**, Jan, 2010. Disponível em: <http://www.infoq.com/news/2010/01/Gremlin>. Acesso: 01 junho 2012.

BAUMANN, P.; DEHMEL, A.; FURTADO, P.; RITSCH, R.; WIDMANN, N. The multidimensional database system RasDaMan. **SIGMOD Rec.**, v. 27, n. 2, p. 575-577, June 1998.

BAPTISTA, C. S.; LIMA JUNIOR, O. F.; OLIVEIRA, M. G.; ANDRADE, F. G.; SILVA, T. E.; PIRES, C. E. S. Using OGC Services to Interoperate Spatial Data Stored in SQL and NoSQL Databases. In: SIMPÓSIO BRASILEIRO DE GEOINFORMÁTICA (GEOINFO), 12, 2011, Campos do Jordão-SP. **Anais...** São José dos Campos: MCT/INPE, 2011. p. 61-72.

BERRIMAN, G. B.; GROOM, S. L. How Will Astronomy Archives Survive the Data Tsunami? **Communications of the ACM**, v. 54, n. 12, p. 52-56, December 2011.

BOLLOBÁS, B. **Modern graph theory**. New York, EUA: Springer-Verlag, 1998. 408p.

BROWN, P. G. Overview of sciDB: large scale array storage, processing and analysis. In: ACM SIGMOD International Conference on Management of data, 2010. **Proceedings...** ACM, New York, NY, USA, 2010. p. 963-968.

BUTLER, H.; DALY, M.; DOYLE, A.; GILLIES, S.; SCHAUB, T.; SCHMIDT, C. **The GeoJSON Format Specification**. Disponível em: <http://www.geojson.org/geojson-spec.html>. Acesso: 18 junho 2012.

CÂMARA, G.; VINHAS, L.; QUEIROZ, G. R.; FERREIRA, K. R.; MONTEIRO, A. M.;

CARVALHO, M.; CASANOVA, M. TerraLib: An open-source GIS library for large-scale environmental and socio-economic applications. In: HALL, B.; LEAHY, M (Eds). **Open Source Approaches in Spatial Data Handling:** Advances in Geographic Information Science. Springer, 2010. p. 247-270.

CASANOVA, M.; CÂMARA, G.; DAVIS, C.; VINHAS, L.; QUEIROZ, G. R. **Bancos de Dados Geográficos**. Curitiba: Editora Mundo Geo, 2005. 504p.

CHANG, F.; DEAN, J.; GHEMAWAT, S.; HSIEH, W. C.; WALLACH, D. A.; BURROWS, M.; CHANDRA, T.; FIKES, A.; GRUBER, R. E. Bigtable: A distributed storage system for structured data. **ACM Transactions on Computer Systems**, v. 26, n. 4, p. 1-26, June 2008.

CHODOROW, K.; DIROLF, M. **MongoDB: The Definitive Guide**. O'REILLY, 2010. 216p.

CLEMENTINI, E.; DI FELICE, P. A Comparison of Methods for Representing Topological Relationships. **Information Sciences - Applications**, v. 3, n. 3, p. 149-178, May 1995.

CODD, E. F. A relational model of data for large shared data banks. **Communications of the ACM**, v. 13, n. 6, p. 377-387, June 1970.

COMER, D. Ubiquitous B-Tree. **ACM Computing Surveys**, v. 11, n. 2, p. 121-137, June 1979.

CORMEN, T. T.; LEISERSON, C. E.; RIVEST, R. L. **Introduction to Algorithms**. Cambridge, MA, USA: MIT Press, 1990. 1048p.

COUCLELIS, H. Cellular worlds: a framework for modeling micro - macro dynamics. **Environment and Planning A**, v. 17 n. 5, p. 585-596, 1985.

CROCKFORD, D. **RFC 4627: The application/ json Media Type for JavaScript Object Notation (JSON)**. July, 2006. Disponível em: <http:// www.ietf.org/rfc/rfc4627.txt>. Acesso: 01 junho de 2012.

CUDRE-MAUROUX, P.; KIMURA, H.; LIM, K.-T.; ROGERS, J.; SIMAKOV, R.; SOROUSH, E.; VELIKHOV, P.; WANG, D. L.; BALAZINSKA, M.; BECLA, J.; DEWITT, D.; HEATH, B.; MAIER, D.; MADDEN, S.; PATEL, J.; STONEBRAKER, M.; ZDONIK, S. A demonstration of scidb: a science-oriented dbms. **Proc. VLDB Endow.**, v. 2, n. 2, p. 1534-1537, August, 2009.

DEAN, J.; GHEMAWAT, S. MapReduce: simplified data processing on large clusters. **Communications of the ACM**, v. 51, n. 1, p. 107-113, January, 2008.

DEAN, J.; GHEMAWAT, S. **LevelDB**. Disponível em: <http://code.google.com/p/leveldb>. Acesso: 08 agosto 2012.

DECANDIA, G.; HASTORUN, D.; JAMPANI, M.; KAKULAPATI, G.; LAKSHMAN, A.; PILCHIN, A.; SIVASUBRAMANIAN, S.; VOSSHALL, P.; VOGELS, W. Dynamo: Amazon's Highly Available Key-value Store. **ACM SIGOPS Operating System Review**, v. 41, n. 6, p. 205-220, December 2007.

ECMA INTERNATIONAL. **ECMAScript Language Specification**. ECMA-262, 5.1 Edition, June, 2011. Disponível em: <http://www.ecma-international.org>. Acesso: 10 junho 2012.

ELMASRI, R.; NAVATHE, S. B. **Fundamentals of database systems**. Addison Wesley, 2006.

FAL LABS. **Kyoto Cabinet: a straightforward implementation of DBM**. Disponível em: <http:// fallabs.com/kyotocabinet>. Acesso: 15 agosto 2011.

FIELDING, R. T.; GETTYS, J.; MOGUL, J. C.; NIELSEN, H. F.; MASINTER, L.; LEACH, P. J.; BERNERS-LEE, T. **RFC 2616: Hypertext Transfer Protocol – HTTP/1.1**. June, 1999. Disponível em: <http://www.ietf.org/rfc/rfc2616.txt>. Acesso: 01 junho 2012.

FURIERI, A. **Using SpatiaLite**. Disponível em: <http:/ /www.gaia-gis.it/gaia-sins>. Acesso: 02 maio 2012.

GAEDE, V.; GÜNTHER, O. Multidimensional access methods. **ACM Computing Surveys**, v. 30, n. 2, p. 170-231, June 1998.

GHEMAWAT, S.; GOBIOFF, H.; LEUNG, S.-T. The Google file system. **ACM SIGOPS Operating Systems Review**, v. 37, n. 5, p. 29-43, October 2003.

GNU. **GDBM – GNU dbm**. Disponível em: <http:// www.gnu.org/software/gdbm>. Acesso: 14 agosto 2011.

GRAY, J.; LIU, D. T.; NIETO-SANTISTEBAN, M.; SZALAY, A.; DEWITT, D. J.; HEBER, G. Scientific Data Management in the Coming Decade. **ACM SIGMOD**, v. 34, n. 4, December 2005.

GUTTMAN, A. R-trees: a dynamic index structure for spatial search. **ACM SIGMOD**, v. 14, n. 12, p. 47-57, June 1984.

HUNGER, M. Neo4j: Java-based NoSQL Graph Database. **Info Q**, Feb., 2010. Disponível em: <http://www.infoq.com/news/2010/02/neo4j-10>. Acesso: 09 junho 2012.

IBM. **IBM DB2 Spatial Extender: user's guide and reference**. Disponível em: <http://www.ibm.com.br>. Acesso: 03 junho 2002.

KERR, N. T. **Alternative Approaches to Parallel GIS Processing**. Arizona State University, December, 2009. Disponível em: <http://www.nathankerr.com/projects/parallel-gis-processing/alternative_approaches_to_parallel_gis_processing.html>. Acesso: 30 maio 2012.

KERSTEN, M.; ZHANG, Y.; IVANOVA, M.; NES, N. SciQL, a query language for science applications. In: Workshop on Array Databases (AD '11), 2011. **Proceedings...** ACM, New York, NY, USA, 2011. p. 1-12.

KROPLA, B. **Beginning MapServer:** Open Source GIS Developmen. Apress, 2005. 448p.

LAKSHMAN, A.; MALIK, P. Cassandra: a decentralized structured storage system. **ACM SIGOPS Operating System Review**, v. 44, n. 2, p. 35-40, April 2010.

LAWDER, J. K.; KING, P. J. H. Using Space-Filling Curves for Multi-dimensional Indexing. In: British National Conference on Databases: Advances in Databases, 17., 2000. **Proceedings...** London: Springer-Verlag, 2000. p. 20-35.

MALONE, M. **Video: How SimpleGeo Built a Scalable Geospatial Database with Apache Cassandra**. Disponível em: <http://www.readwriteweb.com/cloud/2011/02/video-simplegeo-cassandra.php>. Acesso: 18 junho 2012.

MARKUS. **MongoDB Plugin for Quantum GIS**. Disponível em: <http://geokoder.com/mongodb-plugin-for-quantum-gis>. Acesso: 01 junho 2012.

MELTON, J. **Advanced SQL: 1999 - Understanding Object-Relational and Other Advanced Features**. Morgan Kaufmann, 2003. 563p.

MELTON, J; EISENBERG, A. SQL multimedia and application packages (SQL/MM). **ACM SIGMOD**, v. 30, n. 4, p. 97-102, December 2001.

MICROSOFT. **White paper: delivering location intelligence with spatial data**. Disponível em: <http://www.microsoft.com>. Acesso: 12 dezembro 2008.

NEUBAUER, P. **Neo4J Spatial - GIS for the rest of us**. Disponível em: <http://www.slideshare.net/peterneubauer/2011-07oscon>. Acesso: 08 maio 2012.

NIEVERGELT, J.; HINTERBERGER, H.; SEVCIK, K. C. The Grid File: An Adaptable, Symmetric Multikey File Structure. **ACM Transactions on Database Systems**, v. , n. 1, p. 38-71, March 1984.

OBE, R.; HSU, L.; RAMSEY, P. **PostGIS in Action**. Manning Publications, 2011. 520p.

OGC. **OpenGIS Implementation Specification for Geographic information - Simple feature access - Part 1: Common architecture**. Disponível em: <http://www.opengeospatial.org>. Acesso: 17 abril 2012a.

OGC. **OpenGIS Implementation Specification for Geographic information - Simple feature access - Part 2: SQL option**. Disponível em: <http://www.opengeospatial.org>. Acesso: 17 abril 2012b.

OGC. **Web Coverage Service**. Disponível em: <http://www.opengeospatial.org>. Acesso: 17 abril 2012c.

OGC. **Web Coverage Processing Service**. Disponível em: <http://www.opengeospatial.org>. Acesso: 17 abril 2012d.

OLSON, M. A.; BOSTIC, K.; SELTZER, M. Berkeley DB. In: Annual conference on USENIX (ATEC '99). **Proceedings...** USENIX Association, Berkeley, CA, USA, 1999**.**

ORACLE. **Oracle Spatial guide**. Disponível em: <http://www.oracle.com>. Acesso: 29 maio de 2003.

ORACLE. **Oracle Spatial - topology and network data models developer's guide**. Disponível em: <http://www.oracle.com>. Acesso: 23 junho 2008.

POKORNY, J. NoSQL Databases: a step to database scalability in Web environment. In: International Conference on Information Integration and Web-based Applications and Services (iiWAS '11), 13., 2011.

Proceedings... ACM, New York, NY, USA, 2011. p. 278-283.

QUANTUM GIS DEVELOPMENT TEAM (2012). **Quantum GIS Geographic Information System**. Open Source Geospatial Foundation Project. Disponível em: <http://qgis.osgeo.org>. Acesso: 26 maio 2012.

RAOULT, B. **Architecture of the new MARS server**. Disponível em: <www.ecmwf.int/publications>. Acesso: 04 junho 2012.

RICHARDSON, L.; RUBY, S. **RESTful Web Services**. O'REILLY, 2008.

STEVENS, W. R. **TCP/IP Illustrated, Volume 1, The Protocols**. Addison-Wesley, 1994. 600p.

STONEBRAKER, M.; CETINTEMEL, U. 2005. "One Size Fits All": An Idea Whose Time Has Come and Gone. In: International Conference on Data Engineering (ICDE '05), 21. **Proceedings...** IEEE Computer Society, Washington, DC, USA,2005. p. 2-11.

STONEBRAKER, M. Stonebraker on NosQL and Enterprises. **Communications of the ACM**, v. 54, n. 8, p. 10-11, August 2011.

THOMPSON, M. **GEO, CouchDB & Node.j**s. O'REILLY, 2011. 66p.

TOBLER, W. R. Cellular Geography. In: GALE, S.; OLSSON, G. **Philosophy in Geography**. Dordrecht, The Netherlands: D. Reidel, 1979. p. 279-386.

TURTON, I. GeoTools. In: HALL, B.; LEAHY, M (Eds). **Open Source Approaches in Spatial Data Handling:** Advances in Geographic Information Science. Springer, 2010. p. 153-170.

TWITTER. **Introducing FlockDB**. Twitter Engineering Blog, Monday, May, 2010. Disponível em: <http://engineering.twitter.com>. Acesso: 01 maio 2012.

WANG, Y.; WANG, S. Research and implementation on spatial data storage and operation based on Hadoop platform. In: International Conference on Geoscience and Remote Sensing (IITA-GRS), 2010. **Proceedings...** IEEE, v. 2, 2010. p. 275 - 278.

WARMERDAM, F. The Geospatial Data Abstraction Library. In: HALL, B.; LEAHY, M (Eds). **Open Source Approaches in Spatial Data Handling:** Advances in Geographic Information Science. Springer, 2010. p. 87-104.