Revista Brasileira de Geomática
ISSN: 2317-4285

https://periodicos.utfpr.edu.br/rbgeo

Geoprocessamento com banco de dados NoSQL e MapReduce

Dayse Silveira de Almeida
daysesa@ufcat.edu.br
orcid.org/0000-0003-2718-5132
Universidade Federal de Catalão (UFCAT), Catalão, Goiás, Brasil.

Adriano Lopes Leão da Silva
adrianolls.dev@gmail.com
http://orcid.org/0000-0001-8318-8755
Universidade Federal de Catalão (UFCAT), Catalão, Goiás, Brasil.

RESUMO

Informação geoespacial tem sido de grande importância para diversas aplicações. No contexto de monitoramento de degradações ambientais, os dados geoespaciais, normalmente coletados por sensores presentes em satélites, mapeiam focos de queimadas, áreas desmatadas, entre outros. Esses dados são coletados continuamente por longos períodos de tempo e com riqueza de detalhes, gerando assim, grandes volumes. Ao longo das últimas décadas, esses dados foram armazenados principalmente em bancos de dados relacionais. Para atender aos requisitos atuais, de dados em larga escala advindos de diferentes fontes, de tempo real e de alta concorrência, os bancos de dados NoSQL vêm se mostrando como uma melhor alternativa. Esses sistemas de banco de dados normalmente são distribuídos, não requerem dados estruturados, e são desenvolvidos para escalabilidade horizontal. No entanto, há ainda certa deficiência dos bancos de dados NoSQL em termos de funções espaciais. Sendo assim, o objetivo neste artigo é fazer uma revisão sobre os bancos de dados NoSQL a fim de se verificar o suporte desses aos dados geoespaciais. Os bancos de dados NoSQL que têm se destacado na literatura e se mostrado mais adequados para o gerenciamento dos dados geográficos volumosos são os orientados aos documentos, devido ao maior suporte aos formatos de dados geométricos, aos índices, às funções geoespaciais, e devido também ao alto desempenho computacional. Além desses, se destaca também o modelo de processamento distribuído MapReduce pela possibilidade de se criar funções de mapeamento e de redução para dados geoespaciais e tirar proveito das plataformas de alto desempenho computacional que seguem esse modelo.

PALAVRAS-CHAVE: Dados Geoespaciais. Dados Abertos. Bancos de Dados NoSQL. MapReduce.

INTRODUÇÃO

As informações geoespaciais são de grande importância para diversas áreas e aplicações, como o planejamento urbano, o planejamento de tráfego, o gerenciamento de energia e o monitoramento de degradações ambientais e o planejamento de políticas públicas para a preservação do meio ambiente.

Os dados geoespaciais normalmente são coletados por sensores remotos presentes em satélites e, no contexto de degradação ambiental mapeiam queimadas, incêndios florestais e desmatamento e outras. Utilizando dados de focos de queimadas, por exemplo, é possível verificar onde eles ocorrem diariamente, de modo que se possa intervir; fazer análises mensais, com enfoque nos períodos menos chuvosos e com maior risco de fogo; fazer análises anuais, para comparar o crescimento ou a diminuição no número de queimadas e incêndios florestais, e; fazer análises das possíveis alterações em outro período de tempo de interesse.

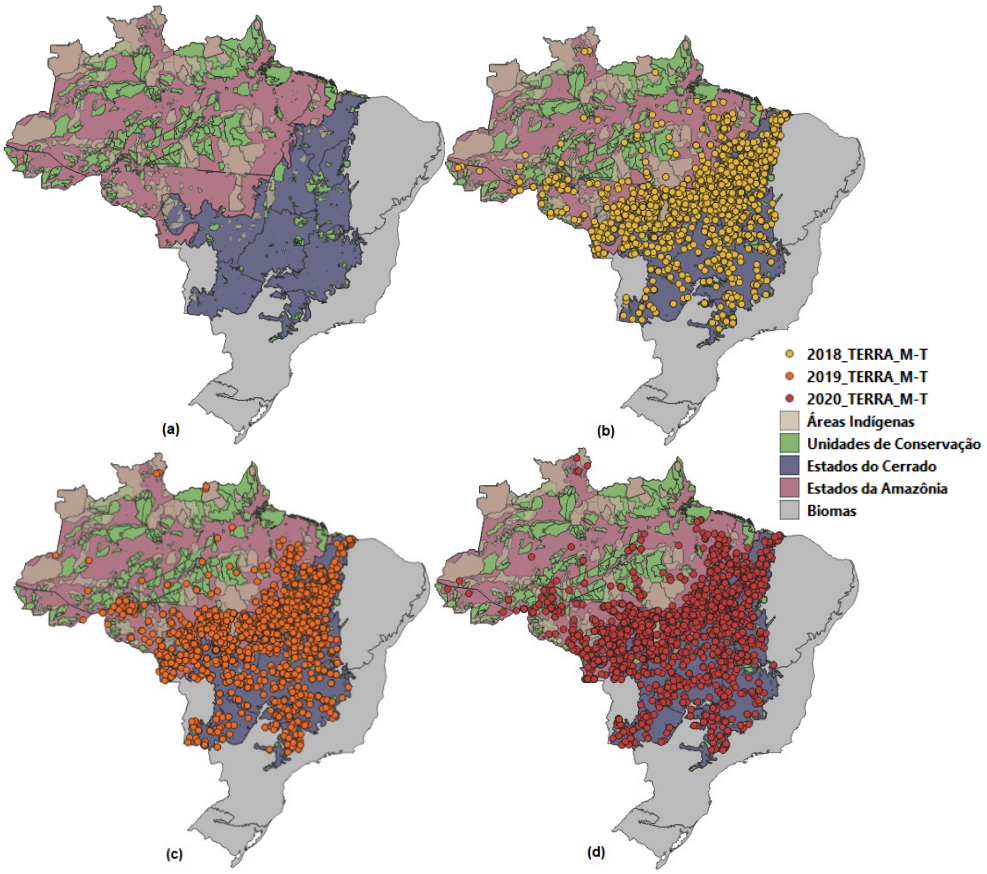
Nas análises estatísticas e visuais realizadas como motivação para este trabalho, foram utilizados dados históricos de focos de queimadas ocorridos na Amazônia e no Cerrado entre 2000 e 2020, no mês de junho, e, dados de desmatamento ocorridos entre 1988 e 2019 na Amazônia e, entre 2000 e 2019 no Cerrado. Para os dados de queimadas coletados, foi escolhido o mês de junho por esse estar dentro do período de seca.

Na Figura 1 são mostrados os dados de focos de queimadas ocorridos na Amazônia e no Cerrado no mês de junho dos anos de 2018, 2019 e 2020. Esses dados foram coletados pelo satélite TERRA M-T e disponibilizados na forma de dados abertos pelo Instituto Nacional de Pesquisas Espaciais (INPE) (INPE, 2022). Pode-se observar focos de queimadas muito próximos ou nas próprias unidades de conservação ambiental e nas áreas de terras indígenas.

Adicionalmente, os dados coletados por sete satélites no mês de junho de 2000 a 2020 mostram, em maior ou menor escala, uma redução gradativa no número de focos até 2017, e a partir de 2018 um crescimento nesse número. Esses satélites foram escolhidos para essa análise, por estarem em órbita durante a maior parte do período de tempo mencionado e terem seus dados disponibilizados ou, por contemplarem o período em que há dados de poucos satélites disponibilizados. No gráfico da Figura 2 é mostrada a variação no número de focos de queimadas na Amazônia e no gráfico da Figura 3, a variação no número de focos de queimadas no Cerrado.

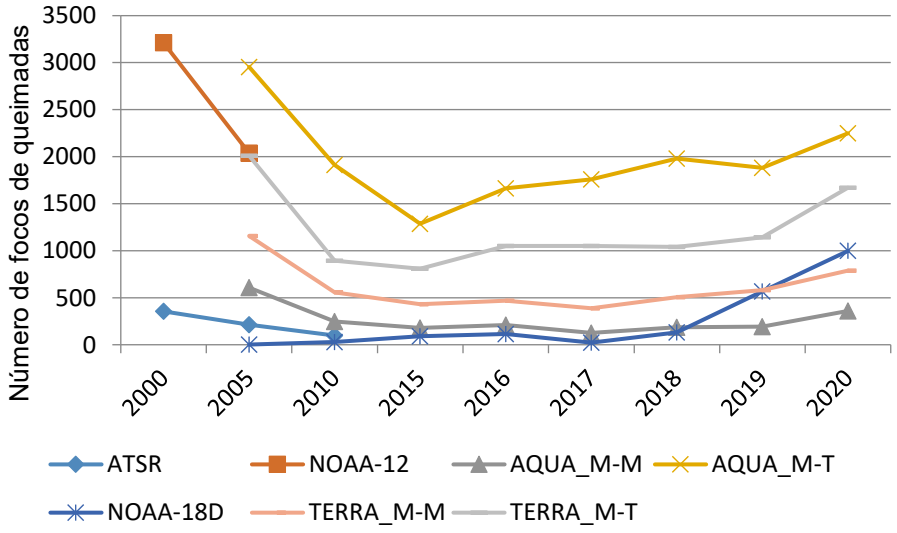
A análise dos dados permite observar um aumento no número de focos de queimadas na Amazônia, em junho de 2018, de 30,41% a 450% em relação a junho de 2017, dependendo do satélite considerado; exceto pelos dados coletados pelo satélite TERRA M-T, nos quais se observa uma redução de 0,86% no número de focos. No ano de 2019 os satélites registraram um aumento no número de focos de 4,87% a 329,55% em relação ao mesmo período do ano de 2018, exceto o satélite AQUA M-T que registrou uma redução de 5,05%. No ano de 2020, o aumento foi de 19,57% a 86,08% em relação ao mesmo período de 2019. Além disso, desde 2010, o maior número de focos registrado por cada um dos satélites ocorreu em 2020, quando comparado ao número de focos no mesmo período dos anos anteriores considerados.

Figura 1 – (a) Estados, unidades de conservação e áreas indígenas nos biomas Amazônia e Cerrado. Focos de queimadas coletados pelo satélite Terra M-T em junho de (b) 2018, (c) 2019 e (d) 2020



Fonte: Autoria própria, com dados do Programa Queimadas do INPE.

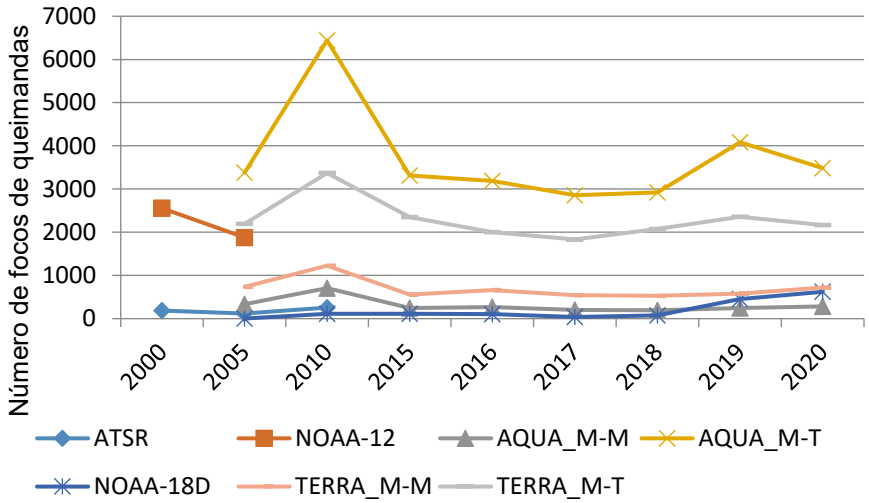
Figura 2 – Número de focos de queimadas na Amazônia, coletados por satélite no mês de junho de cada ano



Fonte: Autoria própria, com dados do Programa Queimadas do INPE.

No Cerrado, os dados por satélite revelam que o aumento de número de focos de queimadas em junho de 2018 em relação a junho de 2017 foi de 2,24% a 119,44%, exceto pelos dados dos satélites TERRA M-M e AQUA M-M que representam redução de 2,21% e 3%, respectivamente. No ano de 2019 houve aumento no número de focos de queimadas de 9,62% a 474,68% em relação a 2018. Em junho 2020, houve um aumento de 14,46% a 37% em relação ao mesmo período de 2019. Os dados dos satélites TERRA M-T e AQUA M-T, no entanto, apresentam uma redução no número de focos em 2020 de 8,14% e 14,70%, respectivamente, nos períodos comparados.

Figura 3 – Número de focos de queimadas no Cerrado, coletados por satélite no mês de junho de cada ano



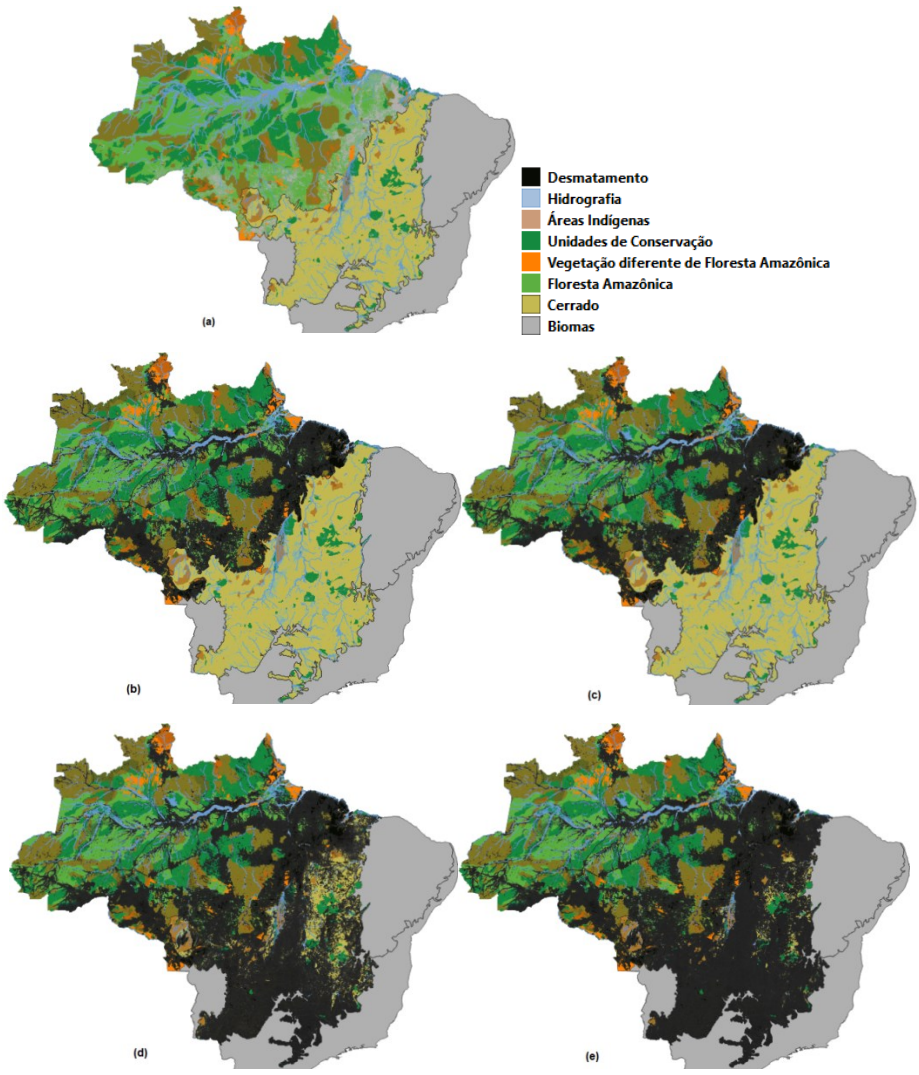
Fonte: Autoria própria, com dados do Programa Queimadas do INPE.

Além das queimadas, esses biomas vêm sendo degradados com o avanço do desmatamento, inclusive sobre as unidades de conservação ambiental e as áreas de terras indígenas. Na Figura 4(a) são destacados os biomas brasileiros Amazônia e Cerrado, com sua hidrografia, unidades de conservação e áreas indígenas, além das áreas com cobertura florestal e das áreas com tipos de vegetação que não são florestas na Amazônia. Na Figura 4(b) é mostrada a área desmatada no bioma Amazônia, entre 1988 e 2007, correspondendo a um total de aproximadamente 693.431 km2. Adicionalmente, na Figura 4(c) é mostrada, a supressão da vegetação nativa entre 2008 e 2019, que corresponde a uma área de aproximadamente 80.401 km2, sendo 17.532 km2 apenas nos anos de 2018 e 2019, uma área maior que a média dos doze anos considerados. Na Figura 4(d), além das áreas desmatadas na Amazônia, é mostrada a área de desmatamento acumulado no Cerrado até o ano de 2000, equivalente a aproximadamente 737.310 km2. Adicionalmente, na Figura 4(e), é mostrada a supressão da vegetação nativa a partir de 2001, até 2019, em uma área de aproximadamente 284.015 km2. Nos anos de 2018 e 2019 foram 13.129 km2 de vegetação nativa suprimida. As áreas desmatadas são mapeadas pelo projeto PRODES, um programa de monitoramento ambiental do INPE, utilizando imagens de satélite da classe Landsat (INPE, 2021).

Para armazenar, gerenciar e visualizar os dados utilizados nas análises realizadas para a motivação deste trabalho, foi utilizado o banco de dados

PostGIS (OBE; HSU, 2019; POSTGIS, 2022) em conjunto com o sistema de informação geográfica QGIS (QGIS, 2022). Os bancos de dados relacionais, tais como o PostGIS e o Microsoft Azure SQL (AU; RISCHPATER, 2015), foram bastante desenvolvidos ao longo dos anos para oferecer suporte aos dados geoespaciais. Esses bancos de dados suportam os principais tipos de dados espaciais e diferentes índices para consultas espaciais mais rápidas.

Figura 4 – (a) Hidrografia, áreas indígenas, unidades de conservação e tipos de vegetação nos biomas Amazônia e Cerrado, (b) desmatamento na Amazônia entre 1988 e 2007, (c) cumulativamente, desmatamento na Amazônia entre 2008 e 2019, (d) adicionalmente, desmatamento no Cerrado até 2000, (e) cumulativamente, desmatamento no Cerrado entre 2001 e 2019



Fonte: Autoria própria, com dados do Programa Queimadas do INPE.

A desvantagem dos bancos de dados relacionais é que eles adotam esquemas de dados estruturados e possuem escalabilidade limitada. Devido aos grandes volumes de dados gerados por sensores remotos presentes nos satélites coletados continuamente por longos períodos; à evolução das comunicações

móveis e o aumento da quantidade de serviços disponibilizados ao usuário que coletam informação geoespacial e possuem requisitos de tempo real com essas informações, tornou-se necessário o uso de esquemas não estruturados para dados, uma escalabilidade maior e mais flexível e tempos de resposta mais rápidos para consultas que aqueles proporcionados pelos bancos de dados relacionais espaciais. Para atender a esses novos requisitos e permitir um processamento uma análise adequada desses dados, os bancos de dados NoSQL vêm sendo desenvolvidos para suportar armazenamento e consultas de dados geoespaciais.

Os sistemas de banco de dados NoSQL normalmente são distribuídos, não requerem dados estruturados, e são desenvolvidos para escalabilidade horizontal. Para obter essa escalabilidade, os bancos de dados NoSQL promovem um relaxamento nas propriedades ACID (atomicidade, consistência, isolamento e durabilidade), presentes nos tradicionais bancos de dados relacionais. Isso provê a capacidade de armazenar, gerenciar e indexar grandes volumes de dados e suportar alta concorrência nas requisições. No entanto, uma desvantagem dos bancos de dados NoSQL em termos de dados geoespaciais, é que esses bancos possuem atualmente, apenas funções espaciais básicas e em menor número quando comparado ao número de funções oferecidas pelos bancos de dados relacionais, como discutido na Seção 3.

Dentre os bancos de dados NoSQL, os bancos de dados orientados a documentos têm se destacado na literatura para o gerenciamento dos dados geoespaciais volumosos, devido ao maior suporte aos formatos de dados geográficos, aos índices e às funções geoespaciais, além do alto desempenho (AMERI et al., 2014; ARIAS MUNOZ et al., 2016; BARALIS et al., 2017; MAIA; CAMARGOS; HOLANDA, 2018; BARTOSZEWSKI; PIORKOWSKI; LUPA, 2019; PIETROŃ, 2019; SVEEN, 2019; GUO; ONSTEIN, 2020).

O objetivo neste artigo é verificar o suporte oferecido pelos bancos de dados NoSQL e o modelo de processamento distribuído Hadoop MapReduce aos dados geoespaciais, diante da sua capacidade de gerenciar e processar grandes volumes de dados com bom desempenho computacional. O artigo está organizado da seguinte maneira. Na Seção 1 são apresentadas as características e os conceitos relacionados aos dados geoespaciais. Na Seção 2 são discutidos os conceitos relativos aos modelos de banco de dados NoSQL. Na Seção 3 são analisados os bancos de dados NoSQL mais populares bem como a plataforma de MapReduce mais popular, e verificado o suporte ao gerenciamento de dados geoespaciais de cada um. Na Seção 4 são colocadas as considerações finais e os resultados esperados.

CONCEITOS RELACIONADOS AOS DADOS GEOESPACIAIS

Há duas formas principais de representar dados geoespaciais (CÂMARA, 2005): dados vetoriais e dados matriciais. Os dados matriciais são compostos por uma matriz de células organizada em linhas e colunas, na qual cada célula tem um valor associado que representa uma informação, como por exemplo, de temperatura. Os dados vetoriais são compostos por pontos individuais que são armazenados como valores de (x, y) em representações bidimensionais e (x, y, z)

em representações tridimensionais. Os pontos são conectados por meio de regras para criar linhas, polígonos e sólidos.

Para representar um objeto do mundo real como um aeroporto, um edifício ou uma área de conservação ambiental, se utiliza os conceitos de recursos e geometrias. Recursos são abstrações do mundo real e podem possuir atributos, como atributos espaciais dando a localização ou a extensão do recurso, atributos temáticos fornecendo características descritivas do recurso, e outros atributos como metadados refletindo a qualidade do dado.

Uma geometria é qualquer forma geométrica que pode ser usada para representar o atributo espacial do recurso, como um ponto, uma linha, um polígono ou um sólido. As geometrias podem ser adicionadas em espaços dimensionais, bidimensionais ou tridimensionais. A dimensão da geometria deve ser menor ou igual à dimensão do espaço incorporado. Para a visualização usando mapas bidimensionais, como aquele mostrado na Figura 1, pontos, linhas e polígonos são suficientes para uma boa representação. Para representações mais complexas que exigem espaço tridimensional, também é necessário utilizar os sólidos.

Para lidar com a localização geoespacial dos recursos, bem como com os relacionamentos entre os recursos, é necessário utilizar um sistema de referência de coordenada (*Coordinate Reference System - CRS*). Sistemas de referência de coordenadas incluem o sistema de coordenadas geocêntricas, o sistema de coordenadas geográficas (WGS84 *datum*), o *Universal Transverse Mercator* (UTM) e o sistema de coordenadas cartesianas.

O EPSG *Geodetic Parameter Dataset*, desenvolvido pelo *European Petroleum Survey Group* (EPSG), é uma coleção pública das definições de sistemas de referência de coordenadas e transformações de coordenadas. O código EPSG é amplamente utilizado em sistemas e bibliotecas de informações geográficas. Já o *Well-known Text* (WKT) é uma linguagem de marcação de texto que representa os sistemas de referência de coordenadas e as conversões entre eles, conforme definido pelo *Open Geospatial Consortium* (OGC).

Algumas características dos dados geoespaciais como os grandes volumes e a multidimensionalidade tornam o seu processamento mais custoso. Esses dados requerem maior escalabilidade dos sistemas e resposta rápida para atualização e consulta. No entanto, nos bancos de dados NoSQL, que fornecem maior flexibilidade e alto desempenho, os principais tipos de geometria suportados são ponto, múltiplos pontos, linha, múltiplas linhas, polígono, múltiplos polígonos e coleção de geometrias, enquanto que os bancos de dados relacionais com base no padrão internacional ISO/IEC 13249: 2016 fornecem 27 tipos de geometrias diferentes (GUO; ONSTEIN, 2020).

BANCOS DE DADOS NOSQL

Para que o armazenamento e o gerenciamento de grandes volumes de dados desordenados sejam eficientes e o processamento de consultas seja realizado em tempo viável, se faz necessária o uso de modelos de banco de dados *Not Only SQL* (NoSQL). Esse termo refere-se a um grupo de bancos de dados não relacionais (MONIRUZZAMAN; HOSSAIN, 2013), em que os dados não são

exclusivamente armazenados em tabelas e a manipulação dos dados não é realizada apenas por meio da *Structured Query Language* (SQL).

Os bancos de dados NoSQL possuem esquemas de dados flexíveis ou não possuem esquemas, possuem alta escalabilidade horizontal e consequentemente se mostram mais adequados para suporte aos grandes volume de dados em termos de desempenho. Por serem sistemas distribuídos, oferecem também a replicação. Os principais modelos de bancos de dados NoSQL são chave-valor, orientado aos documentos, orientado às colunas, orientado aos grafos e o modelo de programação para processamento distribuído MapReduce.

Os bancos de dados chave-valor são constituídos por modelos de dados simples nos quais os dados armazenados correspondem a chaves e valores. Cada chave é única e as consultas por valores são feitas por meio de acesso direto utilizando a chave. O valor, por sua vez, pode ser tanto um dado simples quanto um dado complexo. Assim, esses bancos de dados são caracterizados pela alta capacidade de expansão e pelo acesso rápido aos dados, pois esses são armazenados primariamente em memória RAM (*Random Access Memory*) por meio da estrutura de armazenamento *Distributed Hash Table* (DHT), e acessados de forma direta em tempo de execução.

Ao longo dos últimos anos, muitos bancos de dados baseados no modelo chave-valor foram desenvolvidos motivados pelo sistema Dynamo da Amazon (DECANDIA et al., 2007), tais como o Voldemort, desenvolvido e utilizado pela LinkedIn (PROJECT VOLDEMORT, 2022), o Redis (REDIS, 2022) e o Scalaris (SCHÜTT; SCHINTKE; REINEFELD, 2008).

Os bancos de dados orientados aos documentos são similares aos bancos de dados chave-valor. No entanto, o valor único de cada chave é sempre vinculado a um documento, ou seja, o dado armazenado no campo valor é mais complexo. Esses documentos armazenam dados semiestruturados em formato *Extensible Markup Language* (XML), *Javascript Option Notation* (JSON) ou *Binary JSON* (BSON). Esse modelo é considerado eficiente para armazenar dados pois cada instância pode possuir atributos distintos, não há necessidade de realizar mapeamento de esquemas, e cada instância pode conter dados simples como textos, ou dados complexos como outros documentos aninhados. Assim, todas as informações relevantes a uma consulta são acomodadas em um único documento. Três importantes representantes do modelo de armazenamento de documentos são o MongoDB (BRADSHAW; BRAZIL; CHODOROW, 2019; MONGODB, 2022), o SimpleDB (MURTY, 2008) e o CouchDB (ANDERSON et al., 2010).

Os bancos de dados orientados às colunas armazenam e processam dados de acordo com colunas ao invés de linhas, como no modelo relacional. Assim, as colunas são armazenadas de maneira contígua no disco e em uma consulta, apenas os atributos necessários são lidos. Isso melhora a eficiência de consultas que acessam apenas um subconjunto de colunas de uma tabela grande. Além disso, essas colunas também podem ser segmentadas em vários nós distribuídos em uma rede para garantir a capacidade de expansão.

Os bancos de dados orientados às colunas são inspirados principalmente no BigTable da Google (CHANG et al., 2008). Esses sistemas distribuídos de armazenamento e gerenciamento de dados são o Cassandra, desenvolvido pela Facebook (LAKSHMAN; MALIK, 2009; CARPENTER; HEWITT, 2016; APACHE

CASSANDRA, 2022); o HBase (GEORGE, 2015; APACHE HBASE, 2022), uma versão programada em Java do BigTable e parte do Hadoop da Apache (WHITE, 2012; APACHE HADOOP, 2022) e; o HyperTable (HYPERTABLE INC., 2022).

O último modelo de banco de dados, orientado ao grafo, utiliza a estrutura de grafos para a representação dos dados, ou seja, os dados são modelados por meio de vértices e arestas. Os bancos de dados orientados aos grafos processam com eficiência grandes conjuntos de dados e permitem a construção de modelos preditivos e análise de correlações e padrões de dados. Como os vértices são ligados por relações, as travessias entre os vértices ao longo das arestas são rápidas (HWANG et al., 2015). Assim, esse modelo suporta linguagens de consulta transversais no grafo, além do processamento de consultas online (*Online Transaction Processing - OLTP*) (NOLÉ; SARTIANI, 2018) e vem sendo muito utilizado em redes sociais e sistemas de recomendação. No entanto, os bancos de dados orientados aos grafos normalmente possuem uma arquitetura centralizada e os sistemas de análise em grafo utilizam uma abordagem distribuída, necessitando assim de estratégias de fragmentação dos dados, o que é um desafio para a escalabilidade. Um importante representante desse paradigma é Neo4j (REDMOND; WILSON, 2012; NEO4J, 2022).

Um modelo de programação eficiente para o processamento de grandes volumes de dados é o MapReduce. Esse modelo de programação foi proposto pela Google para suportar computações paralelas de dados em clusters computacionais (DEAN; GHEMAWAT, 2008) e processa os dados em duas fases, inspiradas nas funções de *map* e de *reduce* usadas na programação funcional. Na primeira fase, a função *Map* tem como entrada um conjunto de pares chave/valor e processa esse conjunto de dados divididos em blocos, paralelamente. Como saída tem-se outro conjunto de pares chave/valor encaminhado para a fase intermediária do processo, responsável por organizar os dados e encaminhá-los para a fase de *Reduce*. Nessa fase de *Reduce*, um algoritmo de redução é executado paralelamente sobre os dados que foram mapeados anteriormente, e estes são gravados em um arquivo de saída. A plataforma de programação e execução de código aberto mais difundida do modelo de programação MapReduce, é o Hadoop MapReduce da Apache (DITTRICH; QUIANÉ-RUIZ, 2012; APACHE HADOOP, 2022).

PRINCIPAIS BANCOS DE DADOS NOSQL E O SUPORTE AOS DADOS GEOESPACIAIS

Nesta seção, será examinado o principal banco de dados NoSQL de cada um dos quatro modelos, com base na popularidade classificada pelo DB-Engines (DB-ENGINES RANKING, 2022), a fim de destacar o suporte fornecido para dados geoespaciais. Será analisado também, o suporte da plataforma Hadoop MapReduce para o processamento de dados geoespaciais.

MONGODB

O MongoDB (BRADSHAW; BRAZIL; CHODOROW, 2019; MONGODB, 2022) é um banco de dados de código aberto e gratuito que armazena dados em documentos *Binary JSON* (BSON). Os documentos JSON não possuem estrutura

fixa e suportam *arrays* e documentos embutidos como valores, sendo identificados por uma chave primária. Assim, os relacionamentos entre os documentos também podem ser feitos por referência à chave primária. Esses documentos são organizados em coleções dentro de um banco de dados.

As consultas no MongoDB são feitas por meio de uma linguagem de consulta que permite filtrar e classificar por qualquer campo, incluindo campos que armazenam *arrays* e documentos embutidos, em todos os documentos. As próprias consultas são em formato JSON. Além disso, o MongoDB também fornece suporte para agregações, busca por dados geoespaciais, busca em grafos e busca por texto. Para melhorar a eficiência das consultas, podem-se criar índices para os campos dos documentos que podem ser consultados.

A replicação no MongoDB é feita utilizando-se arquivos de *log* no nó primário, que atua como mestre. Durante a replicação, os nós secundários, que atuam como escravos, verificam quais as operações de escrita foram feitas no nó mestre desde a última sincronização e executam as operações nos arquivos de log dos bancos de dados locais. O MongoDB recupera-se de falhas no nó primário, elegendo um nó secundário como mestre sacrificando a disponibilidade e garantindo a consistência.

O MongoDB suporta escalabilidade horizontal distribuindo os dados entre milhares de nós e balanceando a carga entre esses nós. Além disso, o MongoDB gerencia clusters na nuvem por meio da MongoDB Atlas, na Amazon *Web Service* (AWS), na Microsoft Azure e na Google Cloud.

Suporte aos dados geoespaciais

O MongoDB suporta o armazenamento de dados geospaciais como objetos GeoJSON e pares de coordenadas legadas. Para representar um dado GeoJSON, o MongoDB usa um documento embutido com um tipo de objeto GeoJSON e um *array* de coordenadas (x, y) representando a longitude e a latitude do objeto. Os tipos de objetos GeoJSON suportados pelo MongoDB são: *Point, LineString, Polygon, MultiPoint, MultiLineString, MultiPolygon* e *GeometryCollection*. Para especificar um ponto GeoJSON em um campo chamado "*localizacao*", por exemplo, é necessário embutir o seguinte documento nesse campo:

localizacao: {

type: "Point",

coordinates: [-48.58708, -17.74356] (1)

}

O MongoDB usa o sistema de referência WGS84 para consultas geoespaciais em objetos GeoJSON. Essas consultas são calculadas sobre uma esfera.

Para representar um dado como um par de coordenadas legadas, o MonogDB usa um *array* ou um documento embutido. O ponto definido no exemplo (1) é especificado da seguinte maneira usando *array* e documento embutido, respectivamente:

localizacao: [-48.58708, -17.74356] (2)

localizacao: {

longitude: -48.58708, (3)

latitude: -17.74356

}

O MongoDB possui dois tipos de índices para consultas geoespaciais: o *2dsphere* que, suporta consultas que calculam geometrias em uma esfera como a terra e, o *2d* que, suporta consultas que calculam geometrias em um plano bidimensional. Além disso, o banco de dados em questão provê operadores de consulta geoespacial para verificar a intersecção de geometrias com uma geometria GeoJSON *($geoIntersects*), para selecionar geometrias contidas dentro dos limites de uma geometria GeoJSON (*$geoWithin*), para retornar objetos geoespaciais próximos a um ponto (*$near*) e para retornar objetos geoespaciais próximos a um ponto na esfera (*$nearSphere*). Para a agregação, o MongoDB fornece a operação *$geoNear* que retorna um fluxo ordenado e limitado de documentos com base na proximidade de um ponto geoespacial. Os documentos de saída incluem um campo de distância adicional e podem incluir um campo para identificação da localização.

As operações geoespaciais suportadas em coleções fragmentadas de documentos são *$geoNear, $near* e *$nearSphere*. Já um cluster fragmentado suporta também as operações *$geoWithin* e *$geoIntersect*.

REDIS

O Redis (REDIS, 2022) é um banco de dados do modelo chave-valor de código aberto e gratuito, que armazena não apenas dados simples como também dados de estruturas mais complexas, como *Strings, Lists, Sets, Sorted Sets, Hashes, HyperLogLogs* e *Bitmaps*. As operações com esses dados são feitas em memória RAM, porém, o Redis possui diferentes configurações para garantir a persistência dos dados em disco. O modo RDB permite cópias pontuais em um arquivo compactado em intervalos de tempo específicos, ideal para backups. O modo AOF possui três opções de escrita para que as modificações realizadas na memória sejam armazenadas em um log no disco, garantindo assim a durabilidade dos dados.

A replicação dos dados em um cluster é realizada no esquema mestre e escravo de modo assíncrono, oferecendo também suporte ao modo síncrono. O Redis também oferece gerenciamento dos dados em nuvem por meio da Google *Cloud*.

Suporte aos dados geoespaciais

O Redis possui seis comandos para indexar dados geoespaciais: *geoadd, geodist, geohash, geopos, georadius* e *georadiusbymember*. Esses comandos não possuem seu próprio tipo de dados, tirando proveito do conjunto ordenado de tipos de dados. Para inserir um objeto geoespacial a um conjunto, com uma chave específica, por exemplo, usa-se o comando *GEOADD <nomeconjunto> longitude latitude <nomeobjeto>*. A latitude e a longitude são codificadas como chave do conjunto ordenado usando um algoritmo de *geohash*, e os dados são armazenados por meio dessa chave nesse conjunto ordenado.

Como exemplo, considere a inserção dos objetos *"minhacasa"* e *"casajulia"* no conjunto *"construcao"*:

GEOADD construcao -115.17172 36.12196 minhacasa (4)

GEOADD construcao -115.171971 36.120609 casajulia

Se esses fossem objetos em movimento, para atualizar a localização de um objeto, seria necessário executar o comando novamente com as novas coordenadas. Isso é possível porque o índice geográfico é um conjunto e objetos repetidos não são permitidos.

Os demais comandos determinam a distância entre dois objetos em metros, pés, milhas ou quilômetros (*geodist*), retorna *strings* válidas representando a posição de um ou mais elementos no conjunto ordenado (*geohash*), retorna a posição em longitude e latitude de todos os objetos listados (*geopos*), determina quais objetos estão dentro de um raio a partir de certo ponto (*georadius*), sendo que esse ponto pode ser outro objeto do conjunto (*georadiusbymember*). Além disso, algumas operações podem ser realizadas usando outros comandos sobre o conjunto ordenado, como o comando *zrem* para remover um objeto do conjunto.

Como segundo exemplo, considere o comando *geodist* para determinar a distância entre *"minhacasa"* e *"casajulia"* em metros, unidade de medida padrão:

GEODIST construcao minhacasa casajulia (5)

que obtém como resultado "90.7082".

Além do Redis ser um banco de dados que armazena os dados em memória RAM, ele possui uma capacidade limitada para criar relacionamentos entre os objetos. Devido a isso, ele é mais adequado para sistemas de tempo real, que necessitam de processamento e tempo de resposta pequeno, como sistemas de rastreamento de veículos com carga.

CASSANDRA

O Cassandra (LAKSHMAN; MALIK, 2009; CARPENTER; HEWITT, 2016; APACHE CASSNDRA, 2022) é um sistema de armazenamento e gerenciamento distribuído de grandes volumes de dados, desenvolvido pela empresa Facebook, tornando-se posteriormente um sistema de código aberto.

O Cassandra gerencia os dados em tabelas de quatro dimensões, que incluem linhas, colunas, famílias de colunas e supercolunas. Uma linha é distinguida por uma chave do tipo string. As colunas, por sua vez, podem ser agrupadas constituindo as famílias de colunas. As supercolunas incluem colunas relacionadas aos mesmos nomes. E por fim, uma família de colunas inclui colunas e supercolunas, que podem ser inseridas continuamente na família da coluna em tempo de execução.

O Cassandra utiliza clusters em formato de anel para distribuir os dados. Esse modelo de cluster permite que máquinas sejam adicionadas e removidas do sistema a qualquer instante, tornado o sistema sempre disponível e tolerante a partição. No entanto, a consistência dos dados é realizada por meio de uma assembleia entre os nós do cluster por espalhamento, ou seja, os nós possuem uma área de atuação e não se comunicam com nós muito distantes. Assim,

apenas nós vizinhos se comunicam entre si verificando a consistência da informação e, em caso de divergência, a base de dados é atualizada por completo. Dessa forma, o Cassandra apresenta consistência fraca no espalhamento de alterações feitas por múltiplas atualizações simultaneamente. O Cassandra está disponível como um serviço na Google *Cloud Platform*.

Suporte aos dados geoespaciais

O Cassandra e a linguagem de consulta CQL (*Cassandra Query Language*) não suportam consultas geoespaciais (APACHE CASSANDRA, 2022). O principal meio para fornecer suporte aos dados geoespacial é por meio de um plug-in chamado Índice Lucene (GUO; ONSTEIN, 2020). Esse *plug-in* suporta a indexação de dados geoespaciais do tipo ponto, linha e polígono, transformações geoespaciais para recuperar a intersecção, a união, a diferença e o centróide e, operações geoespaciais para determinar a intersecção e o pertencimento. Os índices são uma extensão dos índices secundários do Cassandra e podem ser criados por meio do comando CQL CREATE CUSTOM INDEX.

Outro importante representante do modelo orientado à coluna, com popularidade superior a do Neo4j (próximo banco de dados a ser analisado) de acordo com o ranking DB-Engines, é o HBase, que faz parte do plataforma Hadoop. No entanto, o HBase não suporta dados e consultas geoespaciais (APACHE HBASE, 2022).

NEO4J

O Neo4j (REDMOND; WILSON, 2012; NEO4J, 2022) é um sistema de banco de dados em grafo de código aberto escrito em Java. O modelo utilizado é o de grafo de propriedades, no qual um grafo direcionado é modelado usando propriedades nos vértices e nas arestas. Os seus componentes - vértices, arestas e propriedades, são armazenados em três arquivos separados. Então, para reduzir a latência, o Neo4j fornece dois níveis de *cache*: *cache* do sistema de arquivos e cache do objeto. No primeiro, os arquivos são divididos em páginas e no segundo, os vértices e as arestas são mantidos como objetos Java no *heap*.

O Neo4j possui uma linguagem de consulta declarativa chamada Cypher. Essa linguagem permite criar, modificar e remover vértices, arestas e propriedades e, a sintaxe da consulta é composta pelas cláusulas *start, match, where* e *return*. *Start* é uma cláusula opcional para especificar o vértice inicial da análise usando o identificador do vértice. *Match* combina padrões permitindo encontrar subgrafos de interesse. E *return* retorna o resultado da consulta. A Cypher também suporta várias funções de agregação, como *sum*, *count* e *avg* e funções que podem ser utilizadas para avaliar expressões em uma consulta, tais como *foreach*, *with*, *has* e *nodes*.

O Neo4j é projetado para ser um banco de dados centralizado e consequentemente é limitado pelos recursos da máquina. Para superar essa limitação, pode-se utilizar um *cluster* no qual o modelo de replicação utilizado pelo Neo4j é o de mestre e escravo, sendo tolerante a falhas e fornecendo disponibilidade. A consistência oferecida, no entanto, é uma consistência eventual.

No modelo distribuído, o Neo4j usa uma estratégia de fragmentação dos dados em cache ao invés de fragmentar os dados em diferentes nós do cluster. Assim, cada nó do cluster armazena o grafo completo e coloca em cache uma parte dos dados, escolhida de acordo com o escalonamento das requisições. Essa estratégia limita a escalabilidade à memória de uma única máquina. Assim como o MongoDB, o Neoj gerencia dados nas nuvens Amazon *Web Services* (AWS), Google *Cloud* e Microsoft Azure.

Suporte aos dados geoespaciais

O Neo4j suporta apenas a geometria espacial do tipo ponto, bidimensional ou tridimensional. Os pontos podem ser especificados como um sistema de referência de coordenadas geográficas ou um sistema de referência de coordenadas cartesianas. O número de funções espaciais é limitado e essas são relativas ao único tipo de geometria suportado. As funções são: *distance()*, que retorna um valor numérico representando a distância geodésica entre dois pontos no mesmo sistema de referência de coordenadas; *point()-WGS 2D*, que retorna um ponto bidimensional no sistema de referência de coordenadas WGS84 correspondente aos valores de coordenadas fornecidos; *point()-WGS 84 3D*, que retorna um ponto tridimensional no mesmo sistema de referência de coordenadas que o anterior; *point()-Cartesian 2D*, que retorna um ponto bidimensional no sistema de referência de coordenadas cartesianas correspondente aos valores de coordenadas fornecidos; e, por fim, a *point()-Cartesian 3D*, que retorna um ponto tridimensional no sistema de referência de coordenadas cartesianas.

Considere como exemplo, uma consulta na linguagem Cypher que realiza cálculo de distância entre dois pontos bidimensionais no sistema de referência de coordenadas cartesianas:

```
WITH point({ longitude: 2.3, latitude: 4.5, crs: 'cartesian' }) AS ponto1,
point({longitude: 1.1, latitude: 5.4, crs: 'cartesian' }) AS ponto2
RETURN distance(ponto1,ponto2) AS distancia
```
(6)

O resultado para esse consulta é 1,5 na mesma unidade de medida dos pontos. Quando o sistema de referência de coordenadas WGS84 é utilizado, a unidade de medida do valor retornado é o metro. Para a indexação espacial, o Neo4j usa Árvore-B+ Generalizada.

Adicionalmente, uma biblioteca chamada *Neo4j Spatial* (NEO4J SPATIAL, 2022) suporta os tipos de geometria *Point, LineString, Polygon, MultiPoint, MultiLineString, MultiPolygon* e *GeometryCollection*, além de operações de topologia para verificar se uma geometria contém outras, se está contida em outra, se sobrepõem, se intersecta ou se está disjunta e, vinte e nove procedimentos espaciais. O índice utilizado por essa biblioteca para consultas sobre geometrias é a Árvore-R.

HADOOP MAPREDUCE

Apesar de não ser um banco de dados e sim um modelo de programação, o MapReduce ((DEAN; GHEMAWAT, 2004) ganha atenção devido à eficiência no processamento paralelo e distribuído de grandes volumes de dados. O Hadoop MapReduce (DITTRICH; QUIANÉ-RUIZ, 2012) é a implementação mais representativa de MapReduce, por ser de código aberto e apresentar as características de simplicidade, escalabilidade e tolerância a falhas do modelo.

Embora o Hadoop MapReduce seja a plataforma mais popular para o processamento distribuído de grandes volumes de dados, ele não é adequado para o processamento em tempo real pois os dados são enviados em lote para cada nó do cluster executar o algoritmo de mapeamento ou redução implementado. Para processamento de dados em tempo real tem-se o Spark (ZAHARIA et al., 2010; APACHE SPARK, 2022), um sistema distribuído de análise de dados com alto desempenho e baseado em memória primária. Essa implementação, por sua vez, requer significantemente mais recursos que o Hadoop MapReduce. Tanto o Hadoop quanto o Spark estão disponíveis como serviços para processamento de dados no *Dataproc* da *Google Cloud Platform*.

A maioria das plataformas para processamento distribuído, incluindo Hadoop MapReduce e Spark, não suportam nativamente dados geoespaciais, o que prejudica a eficiência no processamento desses dados. Diversos esforços vêm sendo feitos no sentido de criar extensões para o Hadoop MapReduce que suportem funcionalidades geoespaciais, ou de projetar algoritmos transformando operadores para dados geoespaciais em funções de Map e Reduce e tirar vantagem dessa plataforma de alto desempenho (AJI et al., 2013; ELDAWY, 2014; ELDAWY; MOKBEL, 2015a; ELDAWY; MOKBEL, 2015b; GAO ET AL., 2017; JO; LEE, 2018; HAN ET AL., 2019; ALKATHIRI; JHUMMARWALA; POTDAR, 2019; GIS TOOLS FOR HADOOP, 2022). Muitos desses trabalhos consideram o Hadoop MapReduce uma caixa preta, ou seja, não há integração com o seu núcleo e, portanto, são limitados pelas limitações do Hadoop MapReduce. Os sistemas que estendem o Hadoop MapReduce normalmente não suportam operações geoespaciais avançadas, como junções geoespaciais e operações geoestatísticas, importantes para análises geoespaciais.

O SpatialHadoop (ELDAWY; MOKBEL, 2013; ELDAWY et al., 2013; ELDAWY, 2014; ELDAWY; MOKBEL, 2015a; ELDAWY; MOKBEL, 2015b; SPATIALHADOOP, 2022), por outro lado, insere conhecimento sobre dados espaciais nas principais camadas do Hadoop MapReduce, tornando-se mais eficiente no processamento de consultas. Assim, o SpatialHadoop introduz índices espaciais que permitem que novas operações espaciais sejam implementadas no sistema. Os índices fornecidos são Arquivo Grid, Árvore-R e Árvore-R+.

O SpatialHadoop não provê uma nova linguagem, mas sim uma extensão da *Pig Latin Language* (OLSTON et al., 2008), chamada Pigeon, com suporte para tipos de dados espaciais, funções e operações. Os tipos de dados espaciais suportados incluem: *Point, LineString, Polygon* e *MultiPolygon*. Também provê (i) funções espaciais básicas como cálculo de área dos polígonos (*Area*); (ii) funções de análise espacial para realizar transformações espaciais nas geometrias de entrada, como cálculo de centroide (*Centroid*) ou intersecção (*Intersection*); predicados para verificar se uma linha está próxima de uma geometria (*IsClosed*) e para verificar se duas geometrias se tocam (*Touches*), além de *Overlaps* e

Contains; e, (iv) funções de agregação espacial que tomam como entrada um conjunto de geometrias e retornam como saída um único valor que resume a entrada, como *ConvexHull*, que retorna um polígono convexo mínimo que contém todas as geometrias de entrada, além das funções para realizar a união de polígonos, traçar a linha do horizonte e verificar o par mais distante e o par mais próximo. Também foram adicionadas consultas, para determinar os k-vizinhos mais próximo (KNN), por intervalo (FILTER) e, para oferecer suporte às junções espaciais (JOIN).

Considere como exemplo o uso da função *Distance* para obter a distância entre cada *casa* e uma farmácia cuja localização é dada por *farm_loc*:

casas_distancia = FOREACH casas GENERATE id, Distance(casa_loc, farm_loc); (7)

Uma consulta pelas 10 casas mais próximas de um ponto é dada por:

casas_proximas = KNN casas WITH_K=10 USING Distance(casa_loc, ponto_loc); (8)

CONSIDERAÇÕES FINAIS

Neste artigo foram revisados os principais bancos de dados NoSQL e a plataforma Hadoop MapReduce a fim de se discutir o suporte fornecido para tratar dados geoespaciais. Verificou-se que o banco de dados NoSQL orientado ao documento MongoDB possui o maior suporte aos dados geoespaciais com maior número de funções e métodos de indexação, bom desempenho no processamento de consultas, maior popularidade e a maior atenção acadêmica recebida nos últimos anos. A plataforma Hadoop MapReduce também se mostra viável, devido à possibilidade de se implementar funções de Map e de Reduce para diversos tipos de aplicações e dados, inclusive para os dados geoespaciais, além da existência de extensões implementadas que suportam esse tipo de dado, sendo que a mais abrangente é o SpatialHadoop.

Na Tabela 1 são resumidas as geometrias, as operações geoespaciais incluindo comandos, funções e procedimentos, e os índices geoespaciais suportados pelos bancos de dados NoSQL, pela plataforma Hadoop MapReduce e pelos *plug-ins* e bibliotecas discutidas nesta seção. Por meio da revisão realizada nesta seção é possível observar que os bancos de dados NoSQL não suportam cálculos de superfície geométrica e processamento de volume.

Como resultado deste artigo, espera-se contribuir para os avanços na pesquisa em banco de dados para tratamento de dados geoespaciais.

No contexto da motivação deste artigo, ou seja, a preservação ambiental, espera-se que diversos trabalhos sejam produzidos, contribuindo com (i) informações para pautar decisões político-administrativas acerca de medidas para a preservação do meio ambiente, e com (ii) a conscientização da sociedade a respeito da degradação ambiental e suas consequências para o ecossistema, a fim de que elejam governantes com propostas consistentes relativas à preservação dos biomas brasileiros e cobrem desses, bem como do Ministério do Meio Ambiente, medidas eficazes.

Tabela 1 – Suporte aos dados geoespaciais pelos bancos de dados NoSQL

Banco de Dados / Plataforma de Processamento	Geometrias		Operações	Índices
MongoDB	*Point, LineString, Polygon, MultiPoint, MultiLineString, MultiPolygon, GeometryCollection*		*$geoIntersects, $geoWithin, $near, $nearSphere*	*2dsphere 2d*
Redis	*Point*		*geoadd, geodist, geohash, geopos, georadius, georadiusbymember*	*geohash*
Cassandra	-	-	-	-
	Índice Lucene	*Point, LineString, Polygon*	*intersects, contains, is_within*	*Lucene index*
Neo4j	-	*Point*	*distance(), point()-WGS 84 2D, point()-WGS 84 3D, point()-Cartesian 2D, point()-Cartesian 3D*	Árvore-B+ Generalizada
	Neo4j Spatial	*Point, LineString, Polygon, MultiPoint, MultiLineString, MultiPolygon, GeometryCollection*	*contains, within, intersects, covers, disjoint*, etc. *spatial.addLayer, spatial.addNode, spatial.addWKTs, spatial.bbox, spatial.importOSM, spatial.importShapefile* e outros 23 procedimentos	Árvore-R
Hadoop MapReduce	-	-	-	-
	Spatial Hadoop	*Point, LineString, Polygon, MultiPolygon*	*Area, Distance, Centroid, Intersection, IsClosed, Touches, ConvexHull, MBR, Union, Skyline, ClosestPair, FarthestPair, DelaunayTriangulation, FILTER, KNN, JOIN*	Grid, Árvore-R e Árvore-R+

Fonte: Autoria própria.

Geoprocessing with NoSQL databases and MapReduce

ABSTRACT

Geospatial information has been relevant for several applications. In the context of monitoring environmental degradation, geospatial data normally collected by sensors present in satellites, map fires, deforested areas, and others. These data are collected continuously for long periods of time and in great detail, thus generating large volumes. Over the past few decades, this data has been stored mainly in relational databases. To attend the current requirements for large-scale data from different sources, processing in real time and with high concurrency, NoSQL databases are proving to be a better alternative. These database systems are normally distributed, do not require structured data, and are designed for horizontal scalability. However, there is still a deficiency of NoSQL databases in terms of spatial functions. Therefore, the purpose in this paper is to review the NoSQL databases in order to verify its support for geospatial data. The NoSQL databases that have been highlighted in the literature and have been shown to be more suitable for the management of large geographic data are those based on documents, due the wide support for geometric data formats, indexes, geospatial functions, and also due the high computational performance. In addition to these, the MapReduce distributed processing model also highlights for the possibility of creating mapping and reduction functions for geospatial data and taking advantage of the high computational performance platforms that follow this model.

KEYWORDS: Geospatial Data. Open Data. NoSQL Databases. MapReduce.

REFERÊNCIAS

AJI, A.; WANG, F.; VO, H.; LEE, R.; LIU, Q.; ZHANG, X.; SALTZ, J. Hadoop-GIS: A high performance spatial data warehousing system over MapReduce. Proceedings of the VLDB Endowment, v. 6. n. 11, p. 1009–1020, ago. 2013. https://doi.org/10.14778/2536222.2536227.

ALKATHIRI, M.; JHUMMARWALA, A.; POTDAR, M. B. Multi-dimensional geospatial data mining in a distributed environment using MapReduce. **Journal of Big Data**, 6, 82, 2019. https://doi.org/10.1186/s40537-019-0245-9.

AMERI, P.; GRABOWSKI, U.; MEYER, J.; STREIT, A. On the Application and Performance of MongoDB for Climate Satellite Data. In: IEEE International Conference on Trust, Security and Privacy in Computing and Communications, 13, 2014, Beijing, China. **Proceedings...** Beijing: IEEE, 2014. p. 652-659. https://doi.org/10.1109/TrustCom.2014.84.

ANDERSON, J. C.; LEHNARDT J.; SLATER N. CouchDB: the definitive guide. USA: O'Reilly Media Inc., 2010.

APACHE CASSANDRA. Disponível em: <https://cassandra.apache.org/>. Acesso em: 06 set. 2022.

APACHE HADOOP. Disponível em: <http://hadoop.apache.org/>. Acesso em: 06 set. 2022.

APACHE HBASE. Disponível em: <https://hbase.apache.org/>. Acesso em: 06 set. 2022.

APACHE SPARK. Disponível em: <https://spark.apache.org/>. Acesso em: 06 set. 2022.

ARIAS MUNOZ, C.; BROVELLI, M. A.; CORTI, S.; ZAMBONI, G. Big Geo Data Management: an Exploration with Social Media and Telecommunications Open Data. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, v. XLI-B2, p. 595–601, 2016. https://doi.org/10.5194/isprs-archives-XLI-B2-595-2016.

AU, C.; RISCHPATER, R. Geospatial Data with Azure SQL Database. In: AU, C.; RISCHPATER, R. (Ed). Microsoft Mapping: Geospatial Development in Windows 10 with Bing Maps and C#. Berkeley, CA, USA: Apress, 2015, p. 33–53.

BARALIS, E.; DALLA VALLE, A.; GARZA, P.; ROSSI, C.; SCULLINO, F. SQL versus NoSQL databases for geospatial applications. In: IEEE INTERNATIONAL CONFERENCE ON BIG DATA (BIG DATA), 2017, Boston, MA, USA. **Proceedings...** Boston, USA: IEEE, 2017. p. 3388-3397. https://doi.org/10.1109/BigData.2017.8258324.

BARTOSZEWSKI D.; PIORKOWSKI A.; LUPA M.. The Comparison of Processing Efficiency of Spatial Data for PostGIS and MongoDB Databases. In: KOZIELSKI S., MROZEK, D., KASPROWSKI P., MAŁYSIAK-MROZEK B., KOSTRZEWA D. (Ed) Beyond Databases, Architectures and Structures. Paving the Road to Smart Data Processing and Analysis. Communications in Computer and Information Science, v. 1018. Ustroń, Poland: Springer, Cham, 2019, p. 291-302. https://doi.org/10.1007/978-3-030-19093-4_22.

BRADSHAW, S.; BRAZIL, E.; CHODOROW, K. **MongoDB: the definitive guide**. 3. ed. USA: O'Reilly Media Inc., 2019.

CÂMARA, G. Representação computacional de dados geográficos. In: CASANOVA., M; CÂMARA, G.; DAVIS, C.; VINHAS, L.; QUEIROZ, G. R. (Org). Bancos de Dados Geográficos. Curitiba: MundoGEO, 2005. p. 1-44.

CARPENTER, J.; HEWITT E. **Cassandra: the definitive guide**. 2. ed. USA: O'Reilly Media Inc., 2016.

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**, New York, NY, USA, v. 26, n. 2, artigo nº 4, 26 p., jun. 2008. https://doi.org/10.1145/1365815.1365816.

DB-ENGINES RANKING. Disponível em: <https://db-engines.com/en/ranking>. Acesso em: 06 set. 2022.

DEAN, J.; GHEMAWAT, S. Mapreduce: simplified data processing on large clusters. ACM - 50th anniversary issue: 1958 – 2008, New York, NY, USA, v. 51, n. 1, p. 107-113, jan. 2008. https://doi.org/10.1145/1327452.1327492.

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 Systems Review**, New York, NY, USA, v. 41, n. 6, p. 205-220, dez. 2007. https://doi.org/10.1145/1323293.1294281.

DITTRICH, J.; QUIANÉ-RUIZ, J-A. Efficient big data processing in Hadoop MapReduce. Proceedings of the VLDB Endowment, v. 1, n. 12, ago. 2012. https://doi.org/10.14778/2367502.2367562.

ELDAWY, A.; MOKBEL, M. F.. 2013. A demonstration of SpatialHadoop: an efficient mapreduce framework for spatial data. Proceedings of the VLDB Endowment, v. 6, n. 12, p. 1230–1233, ago. 2013a. https://doi.org/10.14778/2536274.2536283.

ELDAWY; A.; LI, Y., MOKBEL, M. F.; JANARDAN, R. CG_Hadoop: computational geometry in MapReduce. In: ACM SIGSPATIAL INTERNATIONAL CONFERENCE ON ADVANCES IN GEOGRAPHIC INFORMATION SYSTEMS, 21, 2013, Orlando, Florida. **Proceedings...** New York, NY, USA: Association for Computing Machinery, 2013. p. 294–303. https://doi.org/10.1145/2525314.2525349.

ELDAWY, A. SpatialHadoop: Towards flexible and scalable spatial processing using mapreduce. In: SIGMOD PHD SYMPOSIUM, 2014, Snowbird, UT, USA. **Proceedings...** New York, NY, USA: ACM, 2014. p. 46–50. https://doi.org/10.1145/2602622.2602625.

ELDAWY, A.; MOKBEL, M. F.. 2015a. SpatialHadoop: A MapReduce framework for spatial data. In: INTERNATIONAL CONFERENCE DATA ENGINEERING (ICDE), 31, 2015, Seoul, South Korea. **Proceedings...** Seoul, South Korea: IEEE, 2015a, p. 1352–1363. Doi: 10.1109/ICDE.2015.7113382.

ELDAWY, A.; MOKBEL, M.. The ecosystem of SpatialHadoop. **SIGSPATIAL Special**, v. 6, n. 3, p. 3-10, abr. 2015b. https://doi.org/10.1145/2766196.2766198.

GAO, S.; LI, L.; LI, W.; JANOWICZ, K.; ZHANG, Y. Constructing gazetteers from volunteered Big Geo-Data based on Hadoop. **Computers, Environment and Urban Systems**, v. 61, parte B, p. 172-186, 2017. https://doi.org/10.1016/j.compenvurbsys.2014.02.004.

GEORGE, L. HBase: the definitive guide. 2. ed. USA: O'Reilly & Associates Incorporated, 2015.

GIS TOOLS FOR HADOOP. Disponível em: < https://esri.github.io/gis-tools-for-hadoop/>. Acesso em: 06 set. 2022.

GUO, D.; ONSTEIN, E. State-of-the-Art Geospatial Information Processing in NoSQL Databases. **ISPRS International Journal of Geo-Information**, v.9, n. 5, 331, mai. 2020. https://doi.org/10.3390/ijgi9050331.

HAN, Z.; QIN, F.; CUI, C.; LIU, Y.; WANG, L.; FU, P. Mr4Soil: A MapReduce-Based Framework Integrated with GIS for Soil Erosion Modelling. **ISPRS International Journal of Geo-Information**, v. 8, n. 3, p. 103, fev. 2019. https://doi.org/10.3390/ijgi8030103.

HYPERTABLE INC. Disponível em: <https://hypertable.org/>. Acesso em: 06 set. 2022.

HWANG, J. S.; LEE, S.; LEE, Y.; PARK, S. A Selection Method of Database System in Bigdata Environment: A Case Study From Smart Education Service in Korea. **International Journal of Advances in Soft Computing & Its Applications**, v.7, n. 1, mar. 2015. https://doi.org/10.1016/j.procs.2016.07.096.

INPE. Dados Abertos. Disponível em: <http://www.inpe.br/dados_abertos/>. Acesso em: 01 ago. 2022.

JO, J.; LEE, K.-W. High-Performance Geospatial Big Data Processing System Based on MapReduce. **ISPRS International Journal of Geo-Information**, v. 7, n. 10, p. 399, out. 2018. https://doi.org/10.3390/ijgi7100399.

LAKSHMAN, A.; MALIK, P. Cassandra: structured storage system on a p2p network. In: ACM SYMPOSIUM ON PRINCIPLES OF DISTRIBUTED COMPUTING, 28, 2009, Calgary, Alberta, Canada. **Proceedings...** New York, NY, USA: Association for Computing Machinery, 2009. p. 5. https://doi.org/10.1145/1582716.1582722.

MAIA, D. C. M.; CAMARGOS, B. D. C.; HOLANDA, M.. **Performance Analysis on Voluntary Geographic Information Systems with Document-Based NoSQL Database**. In: ROCHA Á., REIS L. (Ed) Developments and Advances in Intelligent Systems and Applications. Studies in Computational Intelligence, v. 718. Springer, Cham, 2018. p. 181-197. https://doi.org/10.1007/978-3-319-58965-7_13.

MONGODB. Disponível em: <https://www.mongodb.com/>. Acesso em: 06 set. 2022.

MONIRUZZAMAN, A.; HOSSAIN, S. A. Nosql database: new era of databases for big-data analytics classification, characteristics and comparison. **The Computing Research Repository – CoRR**, v. 23, n. 3, 2013. https://doi.org/10.48550/arXiv.1307.0191.

MURTY, J. Programming amazon web services: S3, EC2, SQS, FPS, and SimpleDB. USA: O'Reilly Media Inc., 2009.

NEO4J. Disponível em: <http://www.neo4j.org/>. Acesso em: 06 set. 2022.

NEO4J SPATIAL. Disponível em: <https://neo4j-contrib.github.io/spatial/>. Acesso em: 06 set. 2022.

NOLÉ, M.; SARTIANI, C. Graph management systems: a qualitative survey. **APTIKOM Journal on Computer Science and Information Technologies**, v. 3, n. 2, p. 66-76, 2018.

OBE, R. O.; HSU, L. S. PostGIS in Action, 3. ed. NY, USA: Manning Publications Co., 2019.

OLSTON, C.; REED, B.; SRIVASTAVA, U.; KUMAR, R.; TOMKINS, A. Pig Latin: A Not-so-foreign Language for Data Processing. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, 2008, Vancouver, Canada. Proceedings… New York, NY, USA: Association for Computing Machinery, 2008. p.1099–1110. https://doi.org/10.1145/1376616.1376726.

PIETROŃ, M.. Analysis of performance of selected geospatial analyses implemented on the basis of relational and NoSQL databases, **Polish Cartographical Review**, v. 51, n. 4, p. 167-179, 2019. https://doi.org/10.2478/pcr-2019-0014.

POSTGIS. Disponível em: <https://postgis.net/>. Acesso em: 06 set. 2022.

PROJECT VOLDEMORT. Disponível em: <http://www.project-voldemort.com/voldemort/>. Acesso em: 06 set. 2022.

QGIS. Disponível em: < https://qgis.org/pt_BR/site/>. Acesso em: 06 set. 2022.

REDIS. Disponível em: <https://redis.io/topics/introduction>. Acesso em: 06 set. 2022.

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

SVEEN, A. F.. Efficient storage of heterogeneous geospatial data in spatial databases. **Journal of Big Data**, v. 6, artigo nº 102, nov. 2019. https://doi.org/10.1186/s40537-019-0262-8.

SCHÜTT, T.; SCHINTKE, F; REINEFELD, A. Scalaris: reliable transactional p2p key/value store. In: ACM SIGPLAN WORKSHOP ON ERLANG, 7, 2008, Victoria, BC, Canada. **Proceedings…** New York, NY, USA: Association for Computing Machinery, 2008. p. 41-48, 2008. https://doi.org/10.1145/1411273.1411280.

SPATIALHADOOP. Disponível em: <http://spatialhadoop.cs.umn.edu/>. Acesso em: 06 set. 2022.

WHITE, T. Hadoop: the definitive guide. 3. ed. USA: O'Reilly & Associates Incorporated, 2012.

ZAHARIA, M.; CHOWDHURY, M.; FRANKLIN, M. J.; SHENKER, S.; STOICA, I. Spark: Cluster computing with working sets. HotCloud, v. 10, n. 1-7, p. 95, 2010. Disponível em: <usenix.org>

Recebido: 06 set. 2022
Aprovado: 09 nov. 2022
DOI: 10.3895/rbgeo.v11n2.15927
Como citar: ALMEIDA, D. S.; SILVA, A. L. L.. Geoprocessamento com banco de dados NoSQL e MapReduce. **R. bras. Geom.**, Curitiba, v. 11, n. 2, p. 441-464, abr./jun. 2023. Disponível em: <https://periodicos.utfpr.edu.br/rbgeo>. Acesso em: XXX.
Correspondência:
Dayse Silveira de Almeida
Av. Dr. Lamartine Pinto de Avelar, 1120, CEP 75704-020, Catalão, Goiás, Brasil.