

# Analyzing Performance of Geographic Databases to Enhance Social Analysis in Map Applications

Israel Silva

Universidade Federal Do Pará  
Marituba, Pará, Brasil  
israel.silva@icen.ufpa.br

André Neto

Universidade Federal Do Pará  
Belém, Pará, Brasil  
andre.neto@icen.ufpa.br

Marcelle Mota

Universidade Federal Do Pará  
Belém, Pará, Brasil  
mpmota@ufpa.br

## Abstract

**Context:** Social mapping applications are essential tools for understanding regional needs. In addition to physical geography, these maps integrate data on infrastructure, activities, and social demands—visualizing regions, points of interest, and relevant layers. This approach provides a clear and comprehensive view to support business, government, and social decisions. **Problem:** Managing social mapping data presents challenges, especially when storing sets of coordinates. This limits specialized database functions, impacting application performance in advanced geospatial functions. In particular, it is difficult to define which technologies are most appropriate to handle such data. **Solution:** We propose to use geographic databases such as PostgreSQL and MySQL Spatial. To do this, we propose to identify which functionalities are facilitated by these two technologies and compare their performance in executing these functionalities. **IS Theory:** Guided by the sociotechnical theory of Innovations Theory, this study examines how IS technologies interact in social contexts. **Method:** This study used a qualitative survey of professionals to propose functionalities considered useful for social analysis and an empirical experiment on the performance of the functionalities using PostGIS and MySQL Spatial databases. **Results:** Using a geographic database improved data processing, especially for large data. PostGIS showed better scalability and faster response times than MySQL Spatial as data volumes increased. **Contributions and Impact on the IS field:** This research improves the handling of geospatial data in IS, offering a practical solution to optimize queries, benefiting academia and industry.

## CCS Concepts

• Information systems → Geographic information systems.

## Keywords

Geographic databases, Social mapping, Spatial queries, PostGIS, MySQL Spatial

## 1 INTRODUÇÃO

Nos últimos anos, a utilização de aplicativos de mapas tem crescido significativamente, proporcionando aos usuários informações sobre a infraestrutura e os serviços disponíveis em suas regiões. Tais aplicações vão desde plataformas lucrativas como Uber [3] e iFood [31], até aquelas voltadas para o benefício social, como "Meu Lar de Volta" [21] e o "Fluss – Monitor de enchentes" [27]. Essas aplicações, além de fornecer mapas interativos, agregam dados detalhados sobre diversos pontos de interesse na área mapeada.

Os mapas sociais são representações cartográficas que destacam não apenas a geografia física de uma região, mas também as interações sociais e comunitárias que ocorrem dentro dela. Esses mapas vão além das tradicionais indicações de ruas e pontos de interesse, fornecendo informações detalhadas sobre os serviços, atividades e necessidades específicas de uma comunidade. Nesse contexto, aplicações de mapas sociais podem agregar séries de dados cartográficos (ou espaciais) sociais, que representam as coordenadas, ou conjunto de coordenadas, indicando pontos ou elementos geométricos no mapa [28].

Aplicações que contenham dados cartográficos sociais podem favorecer análises sociais diversas. Analistas podem encontrar tendências e padrões espaciais no mapa e chegar a diferentes conclusões [19]. A possibilidade de realizar operações com os dados cartográficos é um ponto fundamental para realizar tais análises. Por exemplo, encontrar pontos próximos, é uma operação que se utiliza do cálculo entre coordenadas de diferentes dados cartográficos. Operações como essa tendem a se tornar mais complexas ou demoradas de acordo com a quantidade de dados cartográficos disponíveis. Ressalta-se que uma grande quantidade e variedade de dados cartográficos sociais disponíveis tende a favorecer análises sociais mais ricas.

Assim, é importante considerar o projeto da base de dados e o Sistema Gerenciador de Banco de Dados (SGBD) a ser utilizado. No tratamento de dados cartográficos, as bases de dados geográficas ou geoespaciais oferecem vantagens para o armazenamento dos dados de coordenadas e a possibilidade de gerar consultas espaciais complexas [26]. Portanto, os bancos de geográficos podem potencializar análises sociais, permitindo a realização de operações decisivas com os dados cartográficos sociais. SGBDs relacionais tradicionais, como PostgreSQL, MySQL e Oracle, em muitos casos, são estendidos para que possam operar como base de dados geográficos.

Apesar da vantagem em utilizar os bancos de dados geográficos para aplicações de mapas sociais, se faz necessário realizar um projeto detalhado visando entender suas vantagens, desvantagens e melhores decisões de uso. Um aspecto importante, é entender quais operações de consultas são disponibilizadas (ou facilitadas) por SGBDs geográficos e como elas podem ser úteis para as análises sociais. Outro ponto crucial é comparar as soluções disponíveis no mercado, visando entender quais são mais vantajosas para as análises sociais.

O objetivo da pesquisa é propor uma análise comparativa entre a performance de duas soluções de banco de dados geográficos no contexto de aplicações de mapas sociais. Utiliza-se como estudo de caso um projeto de refatoração de uma aplicação desse tipo. O TerPaz Maps é resultado de um projeto social e agrega uma diversidade de dados cartográficos sociais como: escolas, hospitais,

praças, ruas com pavimentação e sem pavimentação, pontos de alagamento, pontos de descarte irregular de lixo, dentre outros [24]. Os dados foram disponibilizados para o trabalho e contam com um total aproximado de 13.208 pontos de interesse (como localização de escolas, hospitais, entre outros), 5.971 ruas e suas condições (como pavimentação, alagamento) a respeito de 8 bairros da região metropolitana de Belém.

O estudo propõe uma pesquisa inicial sobre os SGBDs geográficos e possíveis operações que poderiam ser realizadas com os dados cartográficos sociais. A partir disso, as funcionalidades já existentes no TerPaz Maps foram mantidas e novas funcionalidades foram propostas tendo em vista produzir análises sociais mais ricas. Para isto, foram consultados profissionais de análises sociais, visando entender quais funcionalidades seriam mais relevantes neste contexto.

Com as operações de análises definidas, foram realizados experimentos práticos, objetivando comparar as performances destas funcionalidades com dois dos SGBDs geográficos mais populares: PostGIS [13] e MySQL Spatial [4]. Como resultado o estudo apresenta uma análise detalhada sobre as performances considerando especificamente o contexto de análises sociais. Espera-se que os resultados possam contribuir para o projeto de bases de dados para aplicações de mapas sociais que não apenas otimizam o desempenho das operações, mas também permitam a implementação de funcionalidades mais sofisticadas, oferecendo uma experiência mais rica e informativa aos usuários, facilitando análises sociais e comunitárias.

## 2 REFERENCIAL TEÓRICO

Uma série de referências teóricas embasam o presente estudo e são necessárias para entender os métodos e conclusões da pesquisa. As referências abordam os conceitos de bases de dados e do aplicativo utilizado como objeto de estudo. É importante considerar que este estudo é guiado pela Teoria de Sistemas Sociotécnicos, que aborda como tecnologias em sistemas de informação interagem em contextos sociais [22]. Considera-se, nesse caso, como a tecnologia de banco de dados geográfico pode interagir na compreensão e análise de problemas sociais.

### 2.1 Sistemas de Informação Geográficas (SIG)

Sistemas de Informação Geográfica (SIG) são tecnologias para capturar, armazenar, analisar e visualizar dados geográficos, integrando informações espaciais com dados para apoiar decisões. Um SIG é composto por software, hardware, dados, pessoas e métodos para processar informações relacionadas à localização [20].

Embora tradicionalmente associados a sistemas pesados e complexos, os SIGs também estão presentes em plataformas web e móveis. O avanço das tecnologias de mapeamento, exemplificado por aplicativos como *Google Maps*, facilitou o desenvolvimento de soluções mais acessíveis para análise e visualização de dados geoespaciais.

### 2.2 SGBDs Geográficos

A estrutura, armazenamento e gerenciamento de dados são algumas das principais funções de um SGBD. Um dos principais papéis exercidos pelo SGBD é possibilitar a utilização dos dados por um SIG.

Em geral, **SGBDs relacionais** são as tecnologias mais comuns em sistemas de informação [6]. SGBDs relacionais são baseados no uso de tabelas (ou relações) para representar as entidades armazenadas. Um elemento importante nos SGBDs relacionais é o uso da SQL (Linguagem de Consulta Estruturada) para realizar a consulta dos dados armazenados.

Os **SGBDs geográficos** são estruturados para abranger os recursos existentes de sistemas de gerenciamento de bancos de dados relacionais, incluindo suporte para SQL [26]. Além desses recursos, eles também permitem o armazenamento de estruturas específicas de coordenadas que representam os dados cartográficos. Normalmente compreendem funções para análise básica de objetos espaciais e, mais importante, também suportam os esquemas de indexação espacial. Os índices são extremamente importantes para grandes tabelas espaciais, porque permitem a recuperação rápida de registros durante a consulta [26].

Devido abranger os recursos de SGBDs relacionais, muitas vezes SGBDs geográficos funcionam como extensões destes primeiros. Dentre as extensões, destaca-se por exemplo a possibilidade de incluir tipos específicos de dados espaciais para além dos já existentes. Dentre os principais tipos de dados novos, destacam-se como exemplo os tipos: **POINT**, que representa um ponto de interesse no mapa; **LINESTRING**, que representa uma linha no mapa; **POLYGON**, que representa um polígono no mapa; e **GEOMETRY**, que representa qualquer forma geométrica no mapa [13].

O uso dos tipos específicos permite que operações mais complexas com dados espaciais sejam realizadas, como calcular a distância entre dois pontos, a combinação entre múltiplas geometrias, a distância entre múltiplas geometrias e o cálculo de zonas de interesse (ou influência) ao redor de pontos. Dentre as soluções de SGBDs geográficos mais populares podem ser citados o PostGIS [13], uma extensão ao SGBD PostgreSQL [12]; o MySQL Spatial [4], uma extensão ao SGBD MySQL [23] [29].

### 2.3 Aplicações de Mapas Sociais

Mapas sociais são importantes ferramentas para analisar fenômenos sociais, compreender problemas e encontrar soluções [33]. Aplicações de mapas sociais são sistemas para diferentes plataformas que agregam dados cartográficos sociais e permitem a sua visualização em mapas. Tais aplicações podem ou não permitir a interação do usuário com os dados cartográficos. Entretanto, essa interação tende a favorecer a análise dos dados e a produção de ideias [2]. Além do mais, quanto maior for a quantidade de dados disponíveis maiores são as possibilidades de interação e de análises possíveis.

Dentre as aplicações de mapas sociais, o TerPaz Maps foi utilizado como objeto de estudo para este trabalho. Trata-se de uma aplicação *web* de mapas sociais. Os dados cartográficos disponíveis na aplicação foram obtidos em um processo de mapeamento social colaborativo [24]. Nesse processo, a construção de um mapa social é realizada por meio da colaboração de diversos participantes, residentes das áreas de estudo [32]. A captura de dados foi realizada em 8 bairros da região metropolitana de Belém. Diferentes dados foram capturados nos bairros, como: ruas pavimentadas e não pavimentadas, ruas sujeitas a alagamento, comércio locais, serviços oferecidos, atividades educacionais, entre outros [24].

Ressalta-se que os dados cartográficos disponíveis estavam armazenados em um SGBD relacional *MySQL* (sem extensão geográfica). Assim, os dados cartográficos foram armazenados na base em campos textuais, que englobaram as coordenadas no formato GeoJson. O GeoJSON é um formato de dados baseado em JSON (JavaScript Object Notation) utilizado para representar informações geográficas e seus atributos associados. Ele foi projetado para ser leve, legível por humanos e facilmente manipulável por sistemas e bibliotecas de software voltadas para dados espaciais [16]. Nesse modelo, o TerPaz Maps realiza consultas SQL as estruturas textuais.

Tecnologias de mapas para aplicações conseguem facilmente interpretar GeoJson e exibir as informações disponíveis no mapa. No entanto, a utilização no formato textual de armazenamento dificulta a realização de operações com os dados. Para isso, é necessário converter os valores GeoJson para uma linguagem de programação e realizar os cálculos necessários, o que pode comprometer a eficiência das operações e funcionalidades.

Para a realização desta pesquisa foi disponibilizada, além da base de dados *MySQL*, o código-fonte da aplicação, desenvolvido em PHP [14] com o *framework* *Laravel* [7]. O código-fonte representa a criação de Interface Programável de Aplicação (API) que pode ser utilizada por aplicativos móveis ou web para acessar os dados cartográficos sociais disponíveis.

### 3 TRABALHOS RELACIONADOS

Estudos relacionados oferecem importantes comparações de desempenho e aplicabilidade de bancos de dados relacionais e não relacionais, além de bancos espaciais. Além disso, outras pesquisas contribuem de diferentes formas para a definição dos métodos e análises realizadas na presente pesquisa.

Nguyen[25] aborda a importância da indexação em PostGIS para consultas espaciais, utilizando o esquema R-Tree-over-GiST. O estudo demonstra ganhos significativos em desempenho com a indexação, especialmente em grandes tabelas espaciais, mas também alerta sobre a sensibilidade dos parâmetros geométricos usados nas consultas. O trabalho não aborda a comparação com outros SGBDs espaciais.

Kolonko[17] explora OracleDB e MongoDB, demonstrando que o MongoDB apresenta melhor desempenho em latência e vazão, enquanto o OracleDB destaca-se pela estabilidade. O autor enfatiza a necessidade de critérios claros para escolha de tecnologias conforme as demandas específicas de escalabilidade e diversidade de dados. O trabalho aponta para estratégias para a comparação entre performances de dois SGBDs. Entretanto, não é abordado o uso de SGBDs geográficos, somente relacionais.

Da mesma forma, Fraczek e Plechawska-Wojcik[11] analisam PostgreSQL, MongoDB e Cassandra em uma aplicação de mídia social. Os resultados mostram o MongoDB como mais rápido para leitura e o PostgreSQL como mais eficiente para escrita, enquanto o Cassandra apresenta desafios de implementação. Esses achados oferecem subsídios para escolhas tecnológicas em sistemas web. Abdullah e Zhuge[1] comparam MongoDB e MySQL no contexto de big data, concluindo que o MongoDB supera o MySQL em inserção e recuperação de dados em cenários de alta escala. Eles destacam a inadequação dos SGBDs relacionais tradicionais para aplicações modernas que demandam escalabilidade e alta performance.

Shukla, Shivnani e Shah[30] comparam Oracle Spatial e PostGIS, destacando as suas funcionalidades de indexação espacial. Eles concluem que os bancos de dados espaciais superam os sistemas tradicionais em eficiência e flexibilidade ao lidar com dados geoespaciais, evidenciando as limitações dos SGBDs relacionais tradicionais para esse tipo de aplicação. Os autores também concluem que o Oracle Spatial demonstrou-se mais lento na consulta de dados espaciais em comparação ao PostGIS. O trabalho dos autores, no entanto, não cita o uso das operações específicas de cálculos com os dados espaciais. Além disso, não há contexto sobre a natureza dos dados utilizados nos experimentos.

Lamas et al[18] apresentam a proposta de uma arquitetura para o desenvolvimento de SIG para dispositivos móveis para acesso a informações armazenadas remotamente em um banco de dados geográficos. A pesquisa realizada pelos autores tem grande relevância ao tratar da experiência de uso de SGBDs geográficos dentro de determinados contextos. Entretanto, não é abordada a comparação entre o uso de diferentes SGBDs.

O trabalho de Guinelli et al [15] aborda a criação de uma metodologia MDA (Arquitetura Orientada por Modelos) para o projeto de banco de dados com uma extensão para projeto de banco de dados geográficos. A ferramenta auxilia na modelagem de bases de dados geográficos. O trabalho de modelagem se aproxima ao que foi realizado nesta pesquisa. No entanto não aborda comparações de desempenho entre tecnologias.

#### 3.1 Justificativa para a pesquisa

Embora os estudos revisados ofereçam uma análise aprofundada sobre o desempenho de bancos de dados relacionais, não relacionais e geoespaciais, poucos abordam de maneira detalhada a aplicação de bancos de dados geográficos para mapeamento social. Além disso, apresentam uma série de ideias para uso dos SGBDs geográficos e aplicações que podem se beneficiar fortemente do uso dessa tecnologia.

O presente trabalho se diferencia ao propor um estudo aplicado ao uso de banco de dados geográficos no mapeamento social. O estudo atual se concentra em uma análise mais abrangente de bancos de dados geográficos, incluindo as funcionalidades espaciais avançadas para aplicações de mapeamento social. O estudo enfatiza operações com aplicação específica para análises sociais, adaptando-os ao contexto de mapas sociais. Logo, espera-se que as contribuições dessa pesquisa possam ser relevantes para favorecer e estimular análises sociais.

O estudo também se alinha a aspectos da Teoria Sociotécnica, pois busca incentivar análises sociais. Além disso, a comparação entre as tecnologias pode ser caracterizada como um benchmarking. A Tabela 1 resume a relação da pesquisa atual com os trabalhos citados.

### 4 PROCEDIMENTOS METODOLÓGICOS

O método adotado para este estudo é estruturado em cinco etapas principais, com o objetivo de investigar funções específicas de bancos de dados geográficos, coletar opiniões sobre as funcionalidades mais relevantes, corrigir a base de dados inicial da aplicação, implementar funcionalidades por meio de endpoints na API e, por fim,

Tabela 1: Resumo dos trabalhos relacionados

Trabalho	Contribuição	Lacuna
[26]	Análise de performance de SGBD geográfico	Não comparado com outro SGBD geográfico
[17], [1], [11]	Compara performances de SGBDs relacionais e não relacionais	Não compara com SGBD geográfico
[30]	Comparação de performances entre SGBDs geográficos	Não explora funções especiais com dados geográficos
[18]	Utiliza SGBD geográfico em contexto	Não aborda contexto social
[15]	Modelagem de SGBD geográfico	Não aborda comparação de performance

realizar testes comparativos de desempenho com as funcionalidades nas duas versões do banco de dados.

#### 4.1 Etapa 1: Pesquisa de funções especiais

A primeira etapa do projeto consistiu em uma pesquisa detalhada sobre as funcionalidades geoespaciais do MySQL e PostgreSQL, com foco em suas extensões MySQL Spatial[4] e PostGIS[13] (PostgreSQL[12]), ambas amplamente utilizadas em SGBDs geográficos[29]. A escolha do MySQL Spatial se deve à sua compatibilidade com a estrutura existente na aplicação estudada, além de ambas as ferramentas serem gratuitas. O objetivo foi identificar e propor funcionalidades aplicáveis ao contexto da aplicação, explorando os recursos geoespaciais de cada tecnologia. Houve preocupação em selecionar funções com funcionalidades semelhantes em ambas as plataformas.

O PostGIS[13] oferece a função `ST_DistanceSphere()`, que calcula a distância entre dois pontos em uma esfera. No MySQL Spatial[4], a função equivalente é `ST_Distance_Sphere()`, que também calcula a distância esférica entre dois pontos. Para unir geometrias, o PostGIS[13] disponibiliza a função `ST_Union()`, que combina múltiplas geometrias em uma única entidade geométrica. No MySQL Spatial[4], a função correspondente é `ST_Union()`, que une múltiplas geometrias em uma única geometria composta.

A função `ST_Buffer()` no PostGIS[13] cria uma zona de buffer ao redor de uma geometria, útil para delimitar áreas de influência. De forma semelhante, o MySQL Spatial[4] oferece a função `ST_Buffer()`, que também gera uma área de buffer ao redor de uma geometria. Para verificar a proximidade entre geometrias, o PostGIS[13] possui a função `ST_DWithin()`, que retorna true se a distância entre duas geometrias estiver dentro de um limite especificado, otimizando a busca por proximidade. No MySQL Spatial[4], a função `ST_Distance()` cumpre uma função similar, calculando a distância entre duas geometrias e possibilitando a verificação de proximidade entre objetos espaciais. A tabela 2 resume as funções espaciais utilizadas neste trabalho.

#### 4.2 Etapa 2: Pesquisa de Funcionalidades relevantes

Com as funções especiais identificadas na etapa anterior, foram propostas algumas funcionalidades para o contexto da aplicação. Para validar sua relevância, foi realizada uma pesquisa de opinião por meio de um questionário online [5]. O público-alvo foi composto por 11 profissionais e estudantes de áreas relacionadas à análise social (Ciências Sociais, Ciências Geográficas ou áreas afins). No questionário, as funcionalidades foram propostas aos participantes acompanhadas de imagens ilustrativas de como seriam aplicadas

na prática e na sua forma final onde o usuário poderia usufruir. Em seguida, os participantes avaliaram a relevância de cada funcionalidade em uma escala de 0 a 10, onde notas próximas de 0 indicavam baixa relevância e notas próximas de 10 indicavam alta relevância. Além disso, os usuários também responderam a uma avaliação subjetiva sobre cada funcionalidade proposta, no mesmo formulário.

Em suma as funcionalidades propostas foram as seguintes: **Funcionalidade 1:** ver números no mapa (quantidades, distâncias) que auxiliem em análises de problemas sociais; **Funcionalidade 2:** pesquisar atividades dentro de uma área específica, como encontrar escolas no raio de 1000 metros a partir de um ponto; **Funcionalidade 3:** identificar atividades que tenham outras específicas próximas, como encontrar hospitais que tenham escolas próximas dentro de um raio de 500 metros; **Funcionalidade 4:** visualizar no mapa o comprimento total de trechos de ruas em determinadas condições (pavimentado/alagado/obstruído/etc.); **Funcionalidade 5:** consultar atividades em áreas de difícil acesso, como ruas não pavimentadas ou sujeitas a alagamentos; **Funcionalidade 6:** ver a área de influência total (raios de 100m, 200m, 300m e 400m) de hospitais e como ela mudaria com a construção de um novo hospital em um ponto específico.

A análise das respostas ao formulário, tanto qualitativa quanto quantitativa, destacou as funcionalidades mais relevantes no contexto das análises sociais. As funcionalidades escolhidas foram as 1, 3, 5 e 6, todas com média acima de 8,5 na análise quantitativa e menções positivas dos participantes a partir da análise qualitativa das respostas.

#### 4.3 Etapa 3: Refatoração da base de dados inicial

Para a realização dos experimentos propostos foi realizado um processo de refatoração do projeto inicial do aplicativo para a estrutura de base de dados geográfica. A Figura 1 apresenta o modelo de dados inicial da aplicação. Elementos espaciais nesse modelo eram armazenados com o tipo de dado textual `TEXT` ou `MEDIUMTEXT`, com o nome `geometry`. Estes campos são responsáveis por armazenar os dados espaciais no formato GeoJson.

A tabela `regions` continha dados sobre oito bairros mapeados; `classes` armazenava 14 classes de atividades, incluindo educação, comércio e saúde; `subclasses` possuía 178 subclasses, cada uma vinculada a uma classe específica, por exemplo, escolas públicas vinculada a educação; `activities` armazenava informações sobre 13.208 atividades, relacionadas a uma região e a uma subclasse; `road_networks` continha dados sobre as ruas dos bairros, organizados em 33 registros, que representavam as condições das ruas em cada região. Por fim, a tabela `system_files` era responsável pelo

**Tabela 2: Resumo das funções com dados espaciais identificados nos dois SGBDs**

Descrição	PostGIS	MySQL Spatial
Calcula distância esférica entre dois pontos	<b>ST_DistanceSphere()</b>	<b>ST_Distance_Sphere()</b>
Combina múltiplas geometrias	<b>ST_Union()</b>	<b>ST_Union()</b>
Cria zona de influência ao redor de ponto	<b>ST_Buffer()</b>	<b>ST_Buffer()</b>
Calcula proximidade entre elementos	<b>ST_DWithin()</b>	<b>ST_Distance()</b>

armazenamento de ícones para exibição dos pontos de atividades no mapa.

Ressalta-se que os arquivos textuais GeoJson, além das coordenadas, poderiam conter também informações não espaciais, que indicavam cores ou formas de exibição dos elementos espaciais. Por exemplo, poderiam conter um padrão pontilhado para representar uma linha de rua. O conteúdo GeoJson nessas colunas era estruturado como uma coleção de geometrias (GeometryCollection), que incluía tipos como polígonos, linhas (LineString) e pontos.

Iniciando pela refatoração da estrutura da tabela **regions**, a mesma teve o tipo de sua coluna geometry alterado para GEOMETRY, também foi criado uma nova coluna chamada **center** do tipo POINT, para armazenar o centro da geometria da região. A tabela **activities** teve um tratamento semelhante, tendo sua coluna geometry alterada do tipo TEXT para POINT. **road\_networks** era a tabela responsável pelas ruas de cada região juntamente com sua condição, foi renomeada para **streets**, sua coluna geometry teve o tipo alterado para GEOMETRY, com relação aos dados não espaciais, eles foram extraídos do GeoJson armazenados em uma nova coluna chamada **properties** do tipo LONGTEXT.

Já as condições de cada rua foram salvas em uma tabela chamada **street\_conditions**, nesse momento a tabela que antes tinha todas as ruas condensadas em 33 linhas, agrupadas por suas condições, passou a listar 5.971 registros, onde cada linha contém dados sobre uma rua específica, que se relaciona com uma região(region) e uma condição(street\_condition), o que será muito importante para futuras consultas e filtros, envolvendo ruas específicas de uma região. Por fim a tabela **system\_files** que continha dados sobre ícones foi renomeada para **icons**, onde cada ícone pertence a uma subclasse, ademais, foram mantidas apenas colunas mais relevantes(disk\_name, file\_name).

As novas colunas geoespaciais (geometry e center nas tabelas regions, streets e activities) foram indexadas espacialmente por padrão em ambos os bancos de dados. No PostgreSQL/PostGIS, utilizamos a estrutura GiST, baseada em R-Tree, conforme recomendado por Nguyen [25]. No MySQL Spatial, a indexação espacial foi aplicada com R-Tree sobre InnoDB (desde MySQL 8.0). Esses índices otimizam consultas espaciais, reduzindo escaneamentos completos, melhorando a eficiência de funções como ST\_DWithin e ST\_DistanceSphere.

Abaixo, exemplificamos a criação de índices espaciais em ambas as base de dados SGBD's MySQL Spatial e PostgreSQL/PostGIS, respectivamente:

```
CREATE SPATIAL INDEX indexname ON center(geometry);
CREATE INDEX indexname ON center USING GIST (geometry);
```

Após a mudança nos tipos das colunas, os dados armazenados em GeoJSON foram convertidos e inseridos corretamente no banco,

facilitando uso de filtros e operações espaciais subsequentes. A Figura 2 mostra o resultado da refatoração da base de dados.

#### 4.4 Etapa 4: Implementação das funcionalidades na API

Com as funções necessárias, funcionalidades desejadas e o banco de dados estruturado corretamente, iniciou-se o desenvolvimento das funcionalidades na API. A API original foi reestruturada e foram adicionadas novas funcionalidades tendo em vista testar o acesso aos dados cartográficos em ambos SGBDs.

Os códigos SRID (Spatial Reference System Identifier) definem o sistema de coordenadas usado nas consultas espaciais, influenciando a interpretação das coordenadas, especialmente em cálculos de proximidade e distância. O SRID 4326 (WGS 84) é o padrão para coordenadas geográficas, utilizando latitude e longitude em graus, sendo amplamente adotado em aplicativos de mapeamento. Já o SRID 3857 (Web Mercator) é usado em mapas na web, projetando coordenadas em metros, o que facilita cálculos de distância e área em superfícies planas. Esses sistemas impactam diretamente o funcionamento de consultas espaciais e operações métricas. O funcionamento interno das funcionalidades se dá da seguinte forma:

**Atividades próximas de um ponto:** ilustrada na Figura 3, recebe como parâmetros um bairro, um conjunto de subclasses de busca, uma coordenada de um ponto e o tamanho do raio de busca em metros. É realizada consulta de todas as atividades das subclasses, filtrando conforme os parâmetros enviados. A função espacial **ST\_Distance\_Sphere** é utilizada para retornar apenas as atividades dentro do raio especificado.

**Pontos de Interesse:** ilustrada na Figura 4, recebe como parâmetros um bairro, subclasse de onde partirá a busca (exemplo: hospitais), subclasse alvo (exemplo: escolas) e o raio de busca em metros. Para cada atividade da subclasse de partida, são buscadas atividades da subclasse alvo dentro do raio. Utiliza-se a função **ST\_DistanceSphere** para calcular a distância entre os pontos e aplicar o filtro de proximidade.

**Atividades de Difícil Acesso:** ilustrada na Figura 5, recebe como parâmetro um bairro, uma lista de subclasses e uma lista de condições das ruas próximas. Uma consulta é feita filtrando todas as ruas com as condições requisitadas e depois para cada rua é verificado se possui as atividades a 10 metros de distância que tenha pelo menos uma atividade da lista de subclasses buscadas. A lógica espacial utiliza funções como **ST\_DWithin** (no PostGIS) ou **ST\_Distance** (no MySQL Spatial) para determinar as atividades próximas às ruas filtradas.

**Atividades com Zona de Influência:** ilustrada na Figura 6, recebe como parâmetros um bairro, uma lista de subclasses de busca, uma lista de coordenadas de novas atividades e uma lista

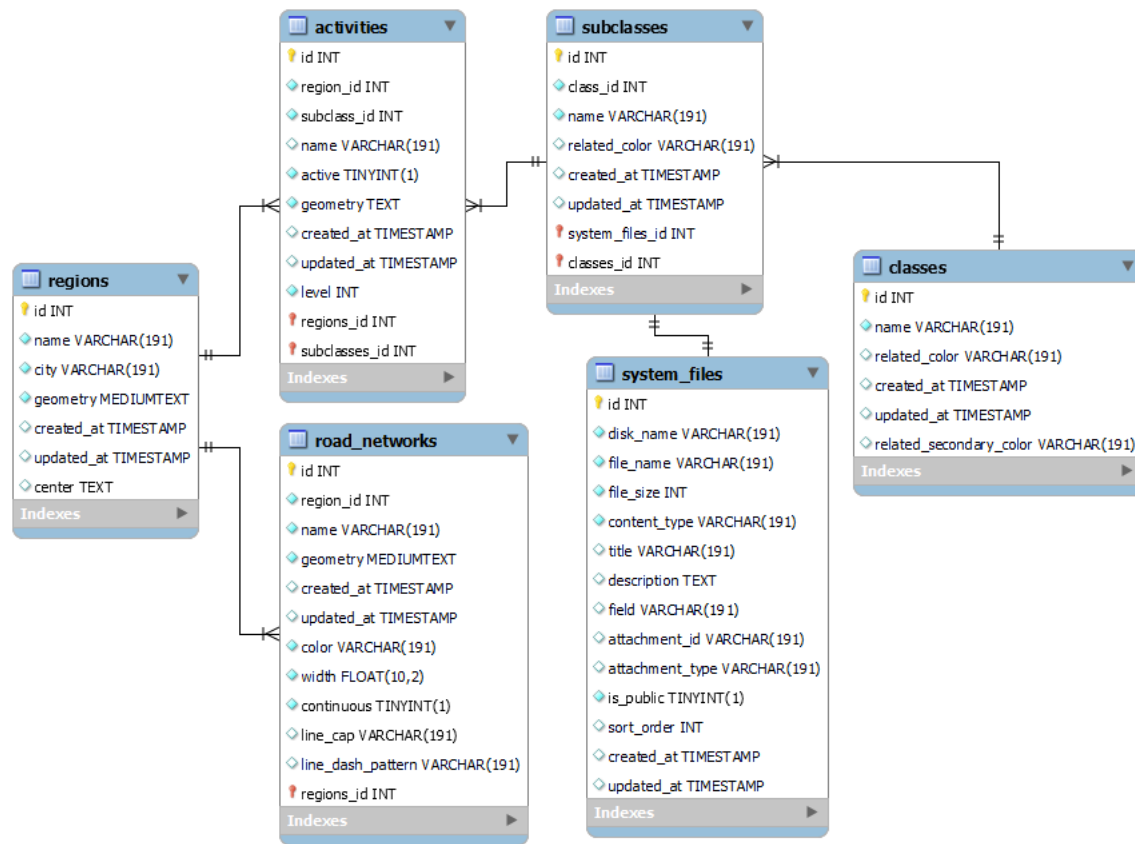


Figura 1: Modelagem da base de dados inicial da aplicação

de tamanhos de raio em metros. É feita a busca pelas atividades presentes no banco de dados, calculado suas geometrias de *buffer* para cada valor de raio, depois todas as geometrias são unidas de acordo com seus raios, a mesma logica é aplicada para a lista de atividades novas que não estão no banco de dados. Por fim tanto as geometrias de *buffer* das atividades do banco de dados, quanto das atividades novas, são unidas de acordo com seus raios, resultando em até 4 geometrias de *buffer*.

No caso do *PostGis*, todos os *buffers* de um raio podem ser calculados e unidos para todas as atividades filtradas do banco de dados com uso das seguintes operações combinadas, *ST\_Union* e *ST\_Buffer*. Contudo, quando se trata do *Mysql Spatial*, seguimos com adaptações devido a seguinte limitação: não é possível usar *ST\_Union* em várias geometrias de uma vez. Então nesse caso sempre precisamos unir dois *buffers* de cada vez.

#### 4.5 Etapa 5: MySQL Spatial VS PostGIS

Postman é uma plataforma amplamente utilizada para desenvolvimento e teste de APIs, com ele é possível configurar requisições HTTP, definir parâmetros e cabeçalhos, automatizar testes em lote e obter os tempos individuais e médios de resposta, facilitando a análise de desempenho e a verificação do funcionamento da API. Para esta etapa, utilizamos a ferramenta Postman para configurar e

enviar as requisições de teste em lotes de 100. Ao final, obtivemos a média do tempo de resposta para o lote, que é calculada pela ferramenta, comparando o desempenho entre endpoints que utilizam MySQL Spatial e PostGIS.

Foi definido o valor de cinco iterações para cada funcionalidade, utilizando o volume de atividades retornadas de 50, 250, 1000, 2000 e 3000 atividades. Os filtros de parâmetro (regiões, subclasses, raio) foram configurados de acordo com o objetivo de controlar a quantidade de atividades retornadas em cada iteração. Apenas na Figura 9, os limites de retorno foram configurados para volumes menores de atividades, com retornos de 50, 250, 500, 750 e 1.000 atividades, visto que a natureza da consulta resulta na comparação das tabelas mais numerosas da aplicação (streets e activites).

Logo, os filtros em cada funcionalidade foram: **Atividades próximas de um ponto (Figura 7)**: região = 7, subclasses = todas, raio = 1.800 metros; **Pontos de interesse (Figura 8)**: região = 7, subclasses = todas, raio = 500; **Atividades de difícil acesso (Figura 9)**: região = todas, ruas = 3 (alagadas) e 5 (obstruídas), Subclasses: 9, 28, 7, 19, 13, 6, 1, 3, 14, 31, 49, 60, 125; e **Atividades com zona de influência (Figura 10)**: região=todas, subclasses=todas, raio=250, 500, 750, e 1.000 metros, atividades novas: 2 localizações novas.

O computador para realização dos testes tem o seguinte hardware:

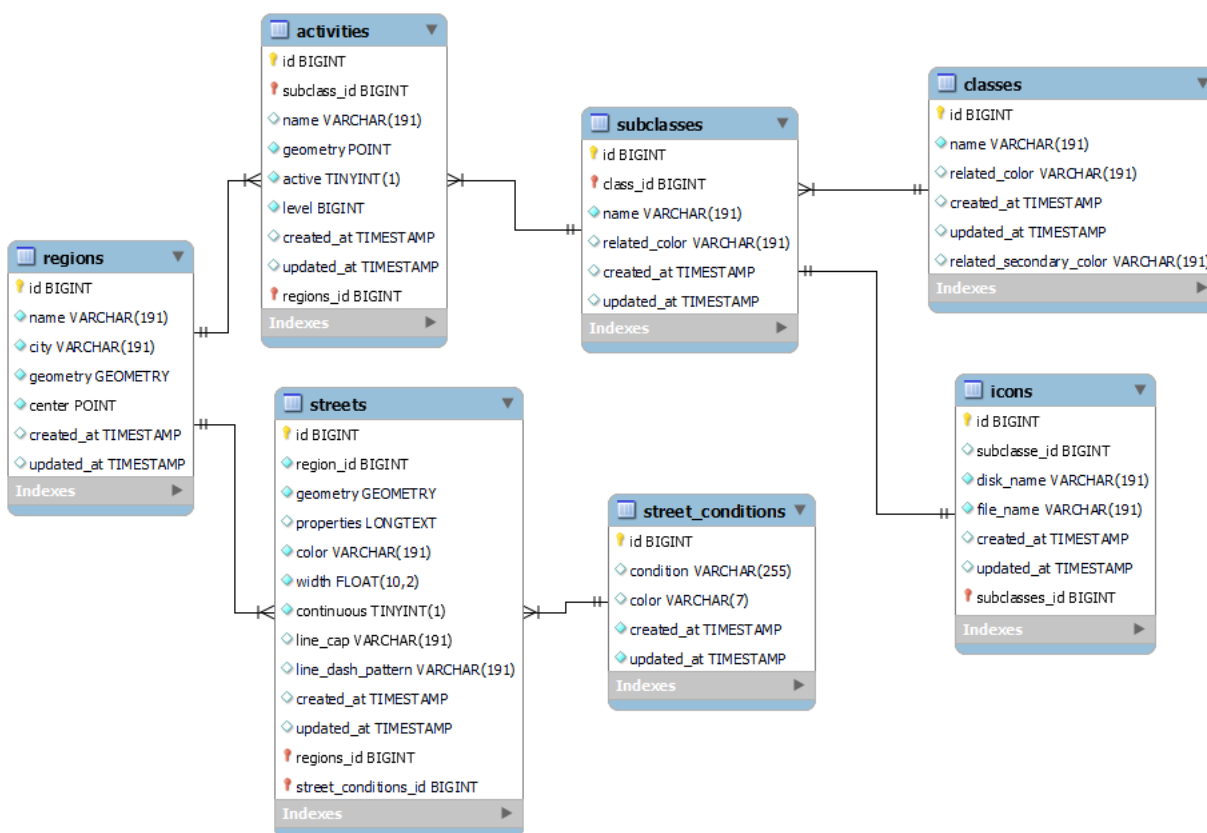


Figura 2: Modelagem da base de dados reestruturada

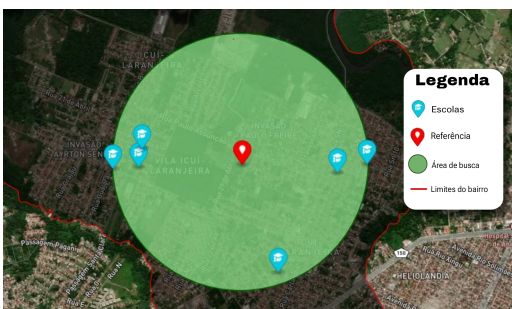


Figura 3: Ilustração das atividades próximas de um ponto

- Processador: AMD Ryzen 5 5600
- RAM: 64GB(4x16gb) DDR4 @3200MHz
- SSD: NVMe M.2 512gb
- SO: Windows 10 Pro 64bits

Banco de dados utilizados:

- PostgreSQL 16.3
- MySQL: 9.0.1

Os testes foram realizados em um ambiente controlado, utilizando condições específicas de hardware e software, o que pode não refletir exatamente cenários de uso prático. Este arranjo foi



Figura 4: Ilustração dos pontos de interesse

necessário para garantir consistência nos resultados e minimizar variáveis externas. Contudo, em futuros testes, planeja-se avaliar o desempenho das soluções em um ambiente mais próximo do uso real, com sistemas web e mobile, envolvendo usuários em testes,





Figura 5: Ilustração das atividades de difícil acesso

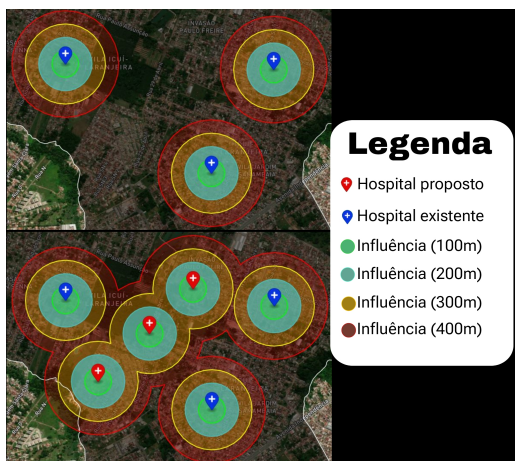


Figura 6: Ilustração das atividades com zona de influência

dependendo da aprovação de um comitê de ética. Esses testes permitirão analisar o comportamento das funcionalidades em infraestruturas heterogêneas, mais representativas de um cenário de aplicação real.

## 5 RESULTADOS

Esta seção apresenta os testes das duas versões do banco de dados, com gráficos de linha para visualizar os padrões de desempenho ao longo do tempo.

### 5.1 Apresentação dos dados

Pode ser observado na Figura 7, que o MySQL tem tempos de resposta menores que o PostgreSQL para até 250 atividades. Com 1000 registros, as médias se igualam (169 ms no PostgreSQL e 168 ms no MySQL), mas o MySQL se torna mais lento com 2000 e 3000 atividades.

Na Figura 8, observa-se que o PostgreSQL tem um desempenho significativamente melhor do que o MySQL, com tempos de resposta consideravelmente menores em todas as iterações. A diferença é

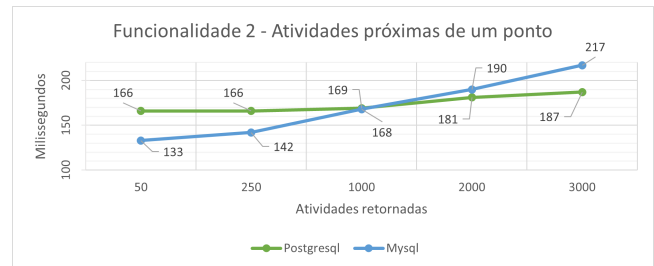


Figura 7: Tempo médio de resposta das iterações nas atividades próximas de um ponto

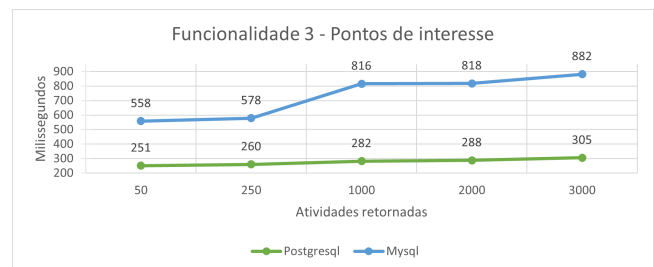


Figura 8: Tempo médio de resposta das iterações nos pontos de interesse

particularmente acentuada em conjuntos maiores de dados, onde o MySQL apresenta um aumento significativo no tempo de resposta (de 558 ms para 882 ms entre 50 e 3000 registros), enquanto o PostgreSQL mantém uma curva de crescimento mais moderada.

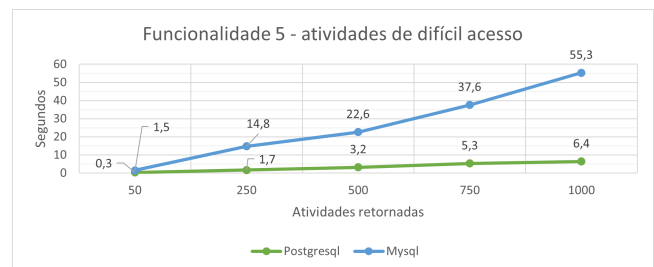
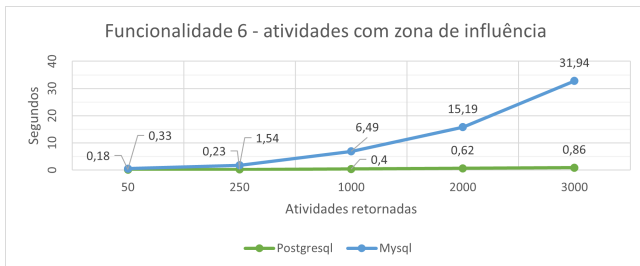


Figura 9: Tempo médio de resposta das iterações nas atividades de difícil acesso

Já na Figura 9, o PostgreSQL novamente supera o MySQL em todas as iterações. A diferença entre os dois bancos de dados aumenta conforme o volume de dados cresce, com o MySQL apresentando tempos muito mais elevados em relação ao PostgreSQL. Em particular, para 1000 registros, o PostgreSQL leva 6,4 segundos em comparação com os 55,3 segundos do MySQL.

Por fim, a Figura 10 mostra que o PostgreSQL continua a apresentar tempos de resposta menores em todas as iterações. A diferença é especialmente notável nos conjuntos maiores de dados, onde o MySQL tem um tempo de resposta significativamente mais alto (31,94 segundos para 3000 atividades), enquanto o PostgreSQL mantém um aumento mais gradual, atingindo 0,86 segundos.





**Figura 10: Tempo médio de resposta das iterações nas zonas de influência**

## 6 DISCUSSÃO

Em todas as funcionalidades testadas, o PostgreSQL apresentou tempos de resposta mais baixos e maior estabilidade no desempenho em comparação com o MySQL, especialmente à medida que o volume de dados aumentava. Nas Figuras 7, 9 e 10, o MySQL mostrou-se competitivo, com tempos de resposta próximos aos do PostgreSQL para conjuntos menores de dados, e até mesmo superior em um caso específico (Figura 7).

No entanto, essa vantagem do MySQL diminuiu rapidamente à medida que o número de atividades retornadas aumentou, e o PostgreSQL começou a se destacar com tempos de resposta menores em conjuntos maiores. Esse comportamento sugere que, embora o MySQL seja competitivo em pequenos volumes, ele perde eficiência conforme a carga de dados cresce. Além disso, em todos os gráficos resultantes, a estabilidade do PostgreSQL é evidente no crescimento do tempo de resposta, apresentando uma curva de aumento suave e gradual.

A Figura 9 mostra que o PostgreSQL com PostGIS teve melhor desempenho, levando 6,4 segundos para 1000 registros, enquanto o MySQL demorou 55,3 segundos. No início, ambos os bancos tiveram tempos próximos, mas à medida que o volume de dados aumentou, o MySQL tornou-se significativamente mais lento. Isso se deve à otimização do PostGIS para operações espaciais complexas, e ao uso mais eficiente de índices espaciais. Já o MySQL apresenta limitações que aumentam o tempo de execução em grandes volumes de dados, reforçando a superioridade do PostgreSQL para consultas espaciais intensivas.

A Figura 10 mostra que o PostgreSQL manteve tempos de resposta menores em todas as iterações, com uma diferença expressiva em grandes volumes de dados (0,86 segundos para 3000 atividades, contra 31,94 segundos no MySQL). Isso ocorre porque a funcionalidade envolve a criação de múltiplos buffers e a união de geometrias, e o MySQL possui uma limitação na função *ST\_Union*. Enquanto o PostgreSQL permite unir várias geometrias em uma única chamada de *ST\_Union*, o MySQL exige múltiplas chamadas sucessivas, o que torna o processo menos eficiente. A função *ST\_Buffer*, por outro lado, funciona da mesma forma nos dois bancos de dados.

A superioridade do PostgreSQL em algumas funcionalidades pode ser atribuída à sua capacidade de otimizar operações geoespaciais complexas, que é realizada de maneira mais eficiente. O MySQL ainda se mostra competitivo em cenários com menores volumes de dados e pode ser adequado para aplicações que não

demandam processamento intensivo de dados geoespaciais, apesar de ter limitações como visto em sua função *ST\_Union*.

## 7 CONSIDERAÇÕES FINAIS

Este estudo comparativo destacou as diferenças significativas de desempenho entre MySQL Spatial e PostgreSQL/PostGIS no processamento de operações geoespaciais, tornando nítida a superioridade do PostgreSQL em cenários com grandes volumes de dados. Os testes demonstraram que, embora o MySQL seja competitivo em conjuntos menores, ele perde eficiência com o aumento da carga de dados, enquanto o PostgreSQL mantém estabilidade e tempos de resposta consistentes, graças às suas otimizações internas para dados geoespaciais. Dessa forma, o PostGIS demonstra-se como uma opção mais viável para análises sociais mais complexas e ricas. Foi possível também observar a viabilidade das funcionalidades propostas através da aplicação TerPaz Maps. Com isso, pode-se observar que é possível otimizar análises sociais a partir da adoção de tecnologias adequadas, como é o caso do banco de dados geográfico PostGIS.

### 7.1 Contribuições da Pesquisa

Como contribuição, este trabalho apresenta uma comparação empírica da execução prática de quatro funcionalidades que podem ser implementadas em aplicações de mapas sociais com o uso de bancos de dados geográficos. A proposta das funcionalidades foi embasada por uma pesquisa dentre as possibilidades dos SGBDs e pelas opiniões de especialistas que atestaram a sua utilidade em análises sociais.

Além disso, considera-se como contribuição o relato do processo de refatoração de uma base de dados relacional para uma base de dados geográficos, o que pode contribuir para outras aplicações que possam realizar este processo. Por fim, considera-se como principal contribuição a comparação entre as performances em implementações práticas utilizando duas entre as principais tecnologias de SGBDs geográficos disponíveis gratuitamente. Em especial, o estudo demonstra resultados que podem ser valiosos para a criação de aplicações que possam suportar análises sociais profundas.

### 7.2 Limitações

O uso de PHP/Laravel se justificou por serem as tecnologias empregadas no projeto original. No entanto, a análise com outras linguagens de programação poderia produzir resultados diferentes. O uso dos SGBDs também é uma limitação devido ao fator financeiro necessário para uso de SGBDs privados. Além disso, a arquitetura e os dados da aplicação utilizada também são fatores que podem limitar a reprodução dos resultados em outros casos. Ademais, ressalta-se que houve um número limitado de funcionalidades que foram testadas devido a um fator de tempo e complexidade de desenvolvimento.

A avaliação do desempenho buscou considerar uma carga realista baseada em uma aplicação prática existente. No entanto, a simulação de cenários de alta concorrência e distribuição geográfica não foi o foco principal deste estudo, o que pode influenciar a análise de escalabilidade e robustez das soluções.

### 7.3 Trabalhos futuros

Como trabalhos futuros, pretende-se explorar a implementação da API com outras tecnologias, como Python [10], e frameworks geoespaciais, como Django [8] ou GeoDjango [9]. Além disso, deve-se ampliar a análise comparativa para outros SGBDs geoespaciais, como Snowflake, Esri Geodatabases, Microsoft SQL Server e SAP HANA, além de avaliar bancos NoSQL, como MongoDB, no contexto de mapas sociais [1, 11, 17]. Deve-se também investigar novas funções espaciais para aprimorar a aplicação e atender a demandas específicas. Em adição a estas melhorias, pretende-se realizar testes em ambientes mais realistas, considerando carga concorrente, alta disponibilidade e distribuição geográfica, para melhor avaliar o desempenho dos bancos de dados.

Outro ponto importante para exploração em trabalhos futuros é o de explorar a integração da API. Primeiramente com servidores de mapas, como GeoServer, ArcGIS e QGIS Server, ampliando a interoperabilidade. Ademais, avaliar como diferentes front-ends podem consumir a API e os desafios dessa flexibilidade. Ampliar a base de participantes na validação das funcionalidades, para obter maior representatividade dos resultados também é uma intenção para futuros estudos.

### 8 Declaração de Uso de IA

A ferramenta de Inteligência Artificial ChatGPT, versão 4, foi utilizada para identificar pontos de melhoria na escrita ao longo do texto em geral, com o objetivo de torná-lo mais claro.

### Referências

- [1] Qingfeng Zhuge Ahmad Abdullah. 2015. From Relational Databases to NoSQL Databases: Performance Evaluation. *Research Journal of Applied Sciences, Engineering and Technology* (2015). <https://doi.org/10.19026/rjaset.11.1799>
- [2] Natalia Andrienko, Gennady Andrienko, Hans Voss, Fatima Bernardo, Joana Hipolito, and Ursula Kretschmer. 2002. Testing the usability of interactive maps in CommonGIS. *Cartography and Geographic Information Science* 29, 4 (2002), 325–342.
- [3] Garrett Camp and Travis Kalanick. 2009. UBER. [https://play.google.com/store/apps/details?id=com.ubercab&hl=pt\\_BR](https://play.google.com/store/apps/details?id=com.ubercab&hl=pt_BR). Accessed: 2024-06-08.
- [4] Oracle Corporation. 2024. MySQL 8.4 Documentation: Spatial Types. <https://dev.mysql.com/doc/refman/8.4/en/spatial-types.html>. Acesso em: 16 nov. 2024.
- [5] Autores da Pesquisa. 2025. Formulário da Pesquisa de Funcionalidades relevantes. <https://form.typeform.com/to/CdsgVoTj>.
- [6] R Elmasri, Shamkant B Navathe, R Elmasri, and SB Navathe. 2015. Fundamentals of Database Systems. In *Advances in Databases and Information Systems*, Vol. 139. Springer.
- [7] Taylor Otwell et al. 2024. *Laravel: The PHP Framework for Web Artisans*. Disponível em: <https://laravel.com/>. Acesso em: 18 nov. 2024.
- [8] Django Software Foundation. 2024. *Django: The Web Framework for Perfectionists with Deadlines*. Disponível em: <https://www.djangoproject.com/>. Acesso em: 18 nov. 2024.
- [9] Django Software Foundation. 2024. *GeoDjango: GIS Extensions for Django*. Disponível em: <https://docs.djangoproject.com/en/stable/ref/contrib/gis/>. Acesso em: 18 nov. 2024.
- [10] Python Software Foundation. 2024. *Python: Programming Language*. Disponível em: <https://www.python.org/>. Acesso em: 18 nov. 2024.
- [11] Konrad Fraczek and Malgorzata Plechawska-Wojcik. 2017. Comparative Analysis of Relational and Non-relational Databases in the Context of Performance in Web Applications. 153–164. [https://doi.org/10.1007/978-3-319-58274-0\\_13](https://doi.org/10.1007/978-3-319-58274-0_13)
- [12] PostgreSQL Global Development Group. 1997. PostgreSQL. <https://www.postgresql.org/>. Accessed: 2024-06-13.
- [13] PostgreSQL Global Development Group. 2024. PostGIS. <https://postgis.net/>. Accessed: 2024-06-08.
- [14] The PHP Group. 2024. *PHP: Hypertext Preprocessor*. Disponível em: <https://www.php.net/>. Acesso em: 18 nov. 2024.
- [15] Joao Victor Guinelli, André de Souza Rocha, Carlos Eduardo Pantoja, and Ricardo Choren. 2014. Uma metodologia para apoio ao projeto de banco de dados geográficos utilizando a mda. In *Anais do X Simpósio Brasileiro de Sistemas de Informação*. SBC, 161–171.
- [16] IETF GeoJSON Working Group. 2016. RFC 7946: The GeoJSON Format. <https://datatracker.ietf.org/doc/html/rfc7946>. Accessed: 2024-11-16.
- [17] Kamil Kolonko. 2018. Performance comparison of the most popular relational and non-relational database management systems. <https://api.semanticscholar.org/CorpusID:208124360>
- [18] Anderson Resende Lamas, Jugurta Lisboa Filho, Ronoel Botelho Júnior, and Alcione Oliveira. 2009. Ontologias e Web Services aplicados ao desenvolvimento de Sistemas de Informação Geográfica Móveis Sensíveis ao Contexto. In *Anais do V Simpósio Brasileiro de Sistemas de Informação*. SBC, 157–168.
- [19] John R Logan. 2012. Making a place for space: Spatial thinking in social science. *Annual review of sociology* 38 (2012).
- [20] Paul A. Longley, Michael F. Goodchild, David J. Maguire, and David W. Rhind. 2015. *Geographic Information Science and Systems* (4th ed.). Wiley, Hoboken, NJ.
- [21] Felipe Machado, Gabriel Hipólito, Fábio Borges, and Felipe Menezes. 2024. MEU LAR DE VOLTA. <https://www.meulardevolta.com.br/>. Accessed: 2024-06-08.
- [22] Enid Mumford. 2000. A Socio-Technical Approach to Systems Design. *Requirements Engineering* 5, 2 (2000), 125–133. <https://doi.org/10.1007/PL00010345>
- [23] AB MySQL. 2001. MySQL.
- [24] André Avelino Da Silva Neto, Paulo Alves De Melo, Estêvão Barbosa, and Marcelle Mota. 2023. Usability Considerations in Social Mobile Maps to Encourage Social Change. In *Anais do XXII Simpósio Brasileiro sobre Fatores Humanos em Sistemas Computacionais*. SBC.
- [25] Thanh Nguyen. 2009. Indexing PostGIS databases and spatial Query performance evaluations. *International Journal of Geoinformatics* 5 (09 2009), 1–9.
- [26] Thanh Thi Nguyen. 2009. Indexing PostGIS databases and spatial Query performance evaluations. *International Journal of Geoinformatics* 5, 3 (2009), 1.
- [27] João Gabriel Pozzobon dos Santos. 2023. FLUSS - Monitor de Enchentes. <https://apps.apple.com/br/app/fluss-monitor-de-enchentes/id6468917253?l=en-GB>. Accessed: 2024-06-08.
- [28] Nohemi Ramirez Aranda, Jeroen De Waegemaeker, Viktor Venhorst, Wim Leendertse, Eva Kerselaers, and Nico Van de Weghe. 2021. Point, polygon, or marker? In search of the best geographic entity for mapping cultural ecosystem services using the online public participation geographic information systems tool, "My Green Place". *Cartography and Geographic Information Science* 48, 6 (2021), 491–511.
- [29] Philippe Rigaux, Michel Scholl, and Agnes Voisard. 2001. *Spatial databases: with application to GIS*. Elsevier.
- [30] Deepika Shukla, Chirag Shrivani, and Darshit Shah. 2016. Comparing Oracle Spatial and Postgres PostGIS. *International Journal of Computer Science and Communication (IJCSC)* 7, 2 (2016). <https://csjournals.com/IJCSC/PDF7-2/16.%20Deepika.pdf>
- [31] Patrick Sigrist, Eduardo Baer, Victor Júnior, Felipe Fioravante, and Simone Alves de Carvalho. 2011. IFOOD. [https://play.google.com/store/search?q=ifood&c=apps&hl=pt\\_BR](https://play.google.com/store/search?q=ifood&c=apps&hl=pt_BR). Accessed: 2024-06-08.
- [32] Brandon Oviedo Sánchez, Emmanuel Alfaro Brenes, Isaac Mena López, Adrián Amador Ávila, Diego Munguía Molina, and Jaime Gutiérrez Alfaro. 2019. Promoting community participation in thematic mapping processes by simplifying the free software tool OSMTracker for Android. In *2019 IV Jornadas Costarricenses de Investigación en Computación e Informática (JoCICI)*. 1–6. <https://doi.org/10.1109/JoCICI48395.2019.9105207>
- [33] Laura Vaughan. 2018. *Mapping society: The spatial dimensions of social cartography*. UCL Press.

Received 20 February 2024; revised 12 March 2024; accepted 5 June 2024