

# Análise Comparativa de LLMs na Otimização de Índices em Banco de Dados: Um Estudo com GPT-4, Claude 3.7 Sonnet e Gemini 2.0 Flash

Daniel Costa<sup>1</sup>, Danyllo Albuquerque<sup>1</sup>, Bruno Moreno<sup>1</sup>

<sup>1</sup>Instituto Federal da Paraíba (IFPB)

Programa de Pós-graduação em Gestão Tecnologia da Informação (PPGTI)

Laboratório de Inovação em Software (LIS)

{cunha.daniel, danyllo.albuquerque, bruno.moreno}@ifpb.edu.br,

**Abstract.** *This study presents a comparative analysis of three Large Language Models (LLMs) — GPT-4, Claude 3.7 Sonnet, and Gemini 2.0 Flash — focusing on their ability to suggest indexes for performance optimization in relational databases. Using a PostgreSQL system hosted on Supabase, indexing suggestions from each model were applied, and execution times of a standardized set of SQL queries were evaluated. The results showed that, although all models provided valid recommendations, only GPT-4 exhibited minimal performance impact, with the lowest average execution time increase (+0.097ms). Claude 3.7 Sonnet showed intermediate performance (+0.102ms), suggesting additional elements such as constraints, while Gemini 2.0 Flash resulted in the highest overhead (+0.149ms), indicating lower practical effectiveness. This research contributes to understanding LLMs as support tools in data engineering, providing empirical evidence of their applicability and limitations in automated database optimization scenarios.*

**Resumo.** *Este estudo apresenta uma análise comparativa entre três Grandes Modelos de Linguagem (LLMs) — GPT-4, Claude 3.7 Sonnet e Gemini 2.0 Flash — com foco em sua capacidade de sugerir índices para otimização de desempenho em bancos de dados relacionais. Utilizando um sistema PostgreSQL hospedado no Supabase, foram aplicadas sugestões de indexação fornecidas por cada modelo e avaliados os tempos de execução de um conjunto padronizado de consultas SQL. Os resultados demonstraram que, embora todos os modelos tenham gerado recomendações válidas, apenas o GPT-4 apresentou impacto mínimo sobre o desempenho, com o menor acréscimo médio de tempo (+0.097ms). Claude 3.7 Sonnet obteve desempenho intermediário (+0.102ms), incluindo sugestões adicionais como constraints, enquanto o Gemini 2.0 Flash apresentou o maior overhead (+0.149ms), indicando menor eficácia prática. A pesquisa contribui para a compreensão do uso de LLMs como ferramentas auxiliares na engenharia de dados, fornecendo evidências empíricas sobre sua aplicabilidade e limitações em cenários de otimização automatizada de bancos de dados.*

## 1. Introdução

O crescimento exponencial dos dados gerados e armazenados em sistemas computacionais impõe desafios cada vez maiores à eficiência no acesso e recuperação dessas informações [Date 2004]. No contexto de bancos de dados relacionais, o desempenho das consultas está diretamente relacionado à forma como os dados são organizados e acessados, sendo a criação de índices uma das principais estratégias de otimização adotadas por administradores de banco de dados (DBAs) e desenvolvedores [Heuser 2009]. No entanto, a definição de índices eficazes

é uma tarefa complexa, que exige conhecimento técnico aprofundado, análise dos padrões de acesso e experiência prática com sistemas de gerenciamento de banco de dados (SGBDs) [Pereira 2021].

Esse processo torna-se ainda mais desafiador em ambientes com alta demanda e grande diversidade de consultas, consumindo tempo e recursos das equipes técnicas. Nesse cenário, os avanços recentes em Inteligência Artificial, especialmente no Processamento de Linguagem Natural (PLN), abrem novas possibilidades para a automação e o suporte a decisões técnicas [de Menezes et al. 2023]. Grandes Modelos de Linguagem (LLMs) têm se destacado pela capacidade de compreender contextos, gerar código e propor soluções para problemas técnicos, incluindo tarefas relacionadas à modelagem e à otimização de bancos de dados [Zhou et al. 2024].

Diante desse contexto, este estudo investiga o potencial dos LLMs como ferramentas auxiliares na otimização do desempenho de bancos de dados relacionais, com ênfase na criação de índices. O objetivo principal é avaliar e comparar as recomendações de indexação fornecidas por três modelos — GPT-4, Claude 3.7 Sonnet e Gemini 2.0 Flash — quanto ao impacto no tempo de execução de um conjunto padronizado de consultas. Para isso, foi desenvolvido um ambiente experimental com um banco de dados PostgreSQL, hospedado no Supabase, estruturado a partir de um esquema representativo de biblioteca. A metodologia consistiu na execução das mesmas consultas em quatro cenários distintos: sem otimizações e com as sugestões de cada um dos três modelos, permitindo uma análise sistemática da eficácia de cada abordagem.

A relevância desta pesquisa reside no uso crescente de LLMs como assistentes técnicos em diversas áreas do desenvolvimento de software, incluindo atividades que exigem conhecimento especializado, como a administração de bancos de dados. Avaliar de forma sistemática o desempenho desses modelos em tarefas críticas, como a recomendação de índices, é essencial para validar sua aplicação em contextos reais. Este trabalho se destaca por adotar uma abordagem comparativa aplicada ao mesmo conjunto de consultas, preenchendo uma lacuna ainda pouco explorada na literatura acadêmica sobre o uso prático de LLMs na otimização automatizada de bancos de dados relacionais.

## 2. Fundamentação

Grandes Modelos de Linguagem (LLMs) representam um avanço notável no campo do Processamento de Linguagem Natural (PLN), caracterizando-se por sua habilidade de compreender, interpretar e gerar textos com alta coerência contextual [Yao et al. 2024]. Esses modelos são treinados em grandes volumes de dados e apresentam desempenho expressivo em tarefas complexas, como geração de código, resumo automático, análise semântica e recomendação de soluções técnicas.

Modelos como o GPT-4<sup>1</sup>, Claude 3.7 Sonnet<sup>2</sup> e Gemini 2.0 Flash<sup>3</sup> destacam-se como exemplos representativos dessa tecnologia, cada um com especificidades em relação à arquitetura, capacidade de raciocínio contextual e velocidade de resposta. A escolha desses três modelos para este estudo se justifica pelo seu amplo reconhecimento no cenário atual, por representarem abordagens distintas de engenharia de LLMs, e por estarem disponíveis em plataformas acessíveis que viabilizam sua experimentação prática. Com a crescente integração dessas ferramentas em fluxos de trabalho técnicos, surge a oportunidade de aplicá-las em

---

<sup>1</sup><https://openai.com/index/gpt-4/>

<sup>2</sup><https://www.anthropic.com/claude/sonnet>

<sup>3</sup>[https://aistudio.google.com/new\\_chat?model=gemini-2.0-flash-exp](https://aistudio.google.com/new_chat?model=gemini-2.0-flash-exp)

domínios tradicionalmente especializados, como a administração e otimização de bancos de dados [Hong et al. 2024], ampliando seu potencial de uso prático em ambientes de produção.

A otimização de bancos de dados relacionais é um aspecto essencial para garantir a eficiência na recuperação de informações, especialmente em aplicações com alto volume de dados e consultas frequentes [Pereira 2021]. Entre as diversas técnicas de otimização existentes, a criação de índices é uma das mais utilizadas devido ao seu impacto direto na redução do tempo de resposta das consultas.

Índices são estruturas auxiliares que permitem o acesso mais eficiente aos dados armazenados, organizando internamente as informações de modo a facilitar operações de busca, filtragem e junção [Heuser 2009]. No entanto, a criação de índices exige um equilíbrio cuidadoso: enquanto índices bem definidos podem acelerar significativamente as consultas, implementações inadequadas ou excessivas podem introduzir *overhead* no sistema, afetando operações de escrita e atualizações, além de consumir recursos adicionais de armazenamento. Nesse cenário, a adoção de soluções baseadas em inteligência artificial, como os LLMs, apresenta-se como uma alternativa promissora para apoiar a escolha automatizada de índices, minimizando a necessidade de intervenção manual e reduzindo os custos associados à análise e experimentação tradicionais.

### 3. Trabalhos Relacionados

Trabalhos recentes sobre otimização de bancos de dados com o uso de inteligência artificial, como demonstrado por [Sriram Dharwada 2025], têm se concentrado majoritariamente em otimizadores de consultas baseados em aprendizado de máquina tradicional. Essas abordagens, embora promissoras, requerem etapas adicionais de pré-processamento, engenharia de características e ajuste de hiperparâmetros, o que pode limitar sua aplicabilidade em ambientes de produção dinâmicos.

Entretanto, ainda são escassos os estudos que abordam de forma específica o uso de LLMs para a recomendação automatizada de índices. Essa lacuna é particularmente relevante, considerando o crescente uso dessas ferramentas por profissionais técnicos em tarefas antes restritas a especialistas em banco de dados. [Jingru Gan 2025] demonstraram a capacidade dos LLMs em gerar estruturas cristalinas, evidenciando seu potencial para compreensão e manipulação de estruturas complexas. Embora seu foco não tenha sido diretamente voltado à performance em bancos de dados, o estudo reforça o potencial dessas ferramentas em cenários estruturados. Na mesma linha, [Tan et al. 2025] conduziram um estudo sobre a aplicação de LLMs como otimizadores de consultas em bancos de dados relacionais. Seus resultados indicam que os modelos são capazes de sugerir planos de execução eficientes e propor recomendações de ajustes, como reescrita de consultas ou criação de índices.

Apesar desses avanços, o presente trabalho se diferencia por realizar uma análise experimental comparativa entre múltiplos LLMs — GPT-4, Claude 3.7 Sonnet e Gemini 2.0 Flash — aplicados a um mesmo ambiente de dados e conjunto padronizado de consultas. Em vez de focar apenas na viabilidade ou na geração de sugestões, esta pesquisa quantifica o impacto real das recomendações de índices no desempenho do banco de dados, utilizando métricas objetivas de tempo de execução. Com isso, oferece uma contribuição inédita à literatura, ao propor uma solução replicável de avaliação prática da eficácia dos LLMs na otimização automatizada de índices em bancos relacionais.

### 4. Configuração do Estudo

A fim de avaliar o impacto das recomendações de indexação feitas por diferentes LLMs sobre o desempenho de consultas SQL (do inglês, *Structured Query Language*), foi conduzido um

experimento controlado com abordagem quantitativa. O estudo foi estruturado para garantir reprodutibilidade e permitir uma comparação objetiva entre os modelos analisados. As etapas da configuração envolvem a definição do ambiente de testes, a estruturação do banco de dados, a elaboração de um conjunto fixo de consultas, e a execução dessas consultas em diferentes cenários de otimização.

#### 4.1. Ambiente de Teste

O ambiente de testes foi implementado utilizando o sistema de gerenciamento de banco de dados PostgreSQL, hospedado na plataforma Supabase<sup>4</sup>, que fornece uma infraestrutura moderna e acessível para experimentos com bancos relacionais. O uso do Supabase permitiu uma implantação rápida do ambiente e controle total das configurações necessárias para a avaliação de desempenho.

#### 4.2. Estrutura do Banco de Dados

Para o experimento, foi modelado um esquema simplificado de uma biblioteca, contendo as seguintes entidades principais: autores, livros, membros, categorias, empréstimos e avaliações. As tabelas foram inicialmente criadas sem chaves estrangeiras nem relacionamentos explícitos, com o objetivo de isolar o impacto dos índices recomendados pelos modelos e evitar interferências de mecanismos de otimização internos do SGBD.

A seguir, apresenta-se o script de criação das tabelas utilizadas:

```
CREATE TABLE public.authors (  
    id bigint primary key generated always as identity,  
    name text NOT NULL,  
    bio text  
);  
CREATE TABLE public.books (  
    id bigint primary key generated always as identity,  
    title text NOT NULL,  
    author_id bigint,  
    published_date timestamp with time zone,  
    genre text  
);  
CREATE TABLE public.members (  
    id bigint primary key generated always as identity,  
    name text NOT NULL,  
    email text NOT NULL,  
    join_date timestamp with time zone  
);  
CREATE TABLE public.categories (  
    id bigint primary key generated always as identity,  
    name text NOT NULL,  
    description text  
);  
CREATE TABLE public.loans (  
    id bigint primary key generated always as identity,  
    book_id bigint,  
    member_id bigint,  
    loan_date timestamp with time zone,  
    return_date timestamp with time zone  
);  
CREATE TABLE public.reviews (  
    id bigint primary key generated always as identity,  
    book_id bigint,  
    member_id bigint,  
    rating integer CHECK (rating >= 1 AND rating <= 5),  
    review_text text  
);
```

Essa estrutura representa um cenário realista e funcional de sistema de gerenciamento de biblioteca, sendo suficientemente complexo para gerar consultas com diferentes níveis de

---

<sup>4</sup><https://supabase.com/>

dificuldade, inclusive com agregações e junções, sem a interferência de otimizações internas prévias.

### 4.3. Coleta de Dados e Execução Experimental

A coleta de dados foi realizada a partir da execução de um conjunto de 10 consultas SQL padronizadas, representando diferentes operações típicas em um banco de dados: filtros simples, agregações, junções e subconsultas. Essas consultas foram projetadas para testar o impacto dos índices sugeridos em contextos variados de acesso a dados:

```
SELECT * FROM public.books WHERE author_id = 1;
SELECT COUNT(*) FROM public.loans;
SELECT * FROM public.reviews WHERE book_id = 1;
SELECT * FROM public.members WHERE id NOT IN
    (SELECT member_id FROM public.loans);
SELECT book_id, AVG(rating) AS average_rating
    FROM public.reviews GROUP BY book_id;
SELECT * FROM public.books WHERE genre = 'Fantasy';
SELECT m.* FROM public.members m, public.loans l
    WHERE m.id = l.member_id;
SELECT b.title, r.rating FROM public.books b
    LEFT JOIN public.reviews r ON b.id = r.book_id;
SELECT author_id, COUNT(*) FROM public.books GROUP BY author_id;
SELECT email, COUNT(*) FROM public.members
    GROUP BY email HAVING COUNT(*) > 1;
```

Essas consultas foram executadas em quatro cenários experimentais distintos:

- **Cenário 1 – Base de comparação:** execução sem qualquer índice adicional, representando o desempenho original do banco de dados;
- **Cenário 2 – GPT-4:** aplicação das recomendações de índices sugeridas pelo modelo GPT-4;
- **Cenário 3 – Claude 3.7 Sonnet:** aplicação das recomendações fornecidas por Claude 3.7 Sonnet, incluindo sugestões adicionais como *constraints* e reorganização de colunas;
- **Cenário 4 – Gemini 2.0 Flash:** aplicação dos índices sugeridos pelo modelo Gemini 2.0 Flash, seguindo os mesmos critérios de avaliação.

Em cada cenário, as consultas foram executadas sob as mesmas condições, e os tempos de resposta foram coletados utilizando ferramentas de análise de desempenho nativas do PostgreSQL. Os dados obtidos permitiram comparar o impacto direto de cada conjunto de índices sobre o tempo de execução das consultas, oferecendo uma base objetiva para avaliar a eficácia de cada modelo.

## 5. Avaliação Comparativa

A análise dos tempos de execução coletados permitiu avaliar o impacto prático das recomendações de indexação fornecidas por cada LLM sobre o desempenho do banco de dados. A Tabela 1 apresenta os tempos médios de resposta, em milissegundos, para cada uma das dez consultas, comparando os quatro cenários experimentais.

De forma geral, observou-se que nenhum dos modelos conseguiu reduzir os tempos de execução em relação à base de comparação, indicando que as sugestões de índices, embora sintaticamente corretas e conceitualmente coerentes, não resultaram em otimizações práticas perceptíveis nos testes realizados. Isso sugere que os modelos, embora capazes de identificar colunas relevantes para indexação, ainda não consideram com precisão o volume de dados, a cardinalidade das colunas, ou os planos de execução gerados pelo otimizador do PostgreSQL.

**Tabela 1. Comparação dos Tempos de Execução por Consulta e Modelo**

Consulta	Base (ms)	GPT-4 (ms)	Claude 3.7 (ms)	Gemini 2.0 (ms)
Livros por Autor	0.392	0.522 (+0.130)	0.500 (+0.108)	0.554 (+0.162)
Total de Empréstimos	0.389	0.431 (+0.042)	0.477 (+0.088)	0.513 (+0.124)
Avaliações por Livro	0.400	0.453 (+0.053)	0.509 (+0.109)	0.514 (+0.114)
Membros sem Empréstimos	0.541	0.583 (+0.042)	0.671 (+0.130)	0.734 (+0.193)
Média de Avaliações	0.458	0.562 (+0.104)	0.599 (+0.141)	0.600 (+0.142)
Livros por Gênero	0.400	0.523 (+0.123)	0.511 (+0.111)	0.587 (+0.187)
Membros com Empréstimos	0.679	0.753 (+0.074)	0.805 (+0.126)	0.917 (+0.238)
Livros sem Avaliações	0.796	0.869 (+0.073)	0.930 (+0.134)	0.933 (+0.137)
Total de Livros por Autor	0.504	0.594 (+0.090)	0.555 (+0.051)	0.626 (+0.122)
Membros com <i>E-mails</i> Duplicados	0.453	0.544 (+0.091)	0.528 (+0.075)	0.576 (+0.123)

Este achado pode ser atribuído a dois fatores principais. Primeiro, ao uso de um conjunto de dados relativamente pequeno, em que os planos de execução padrão já são suficientemente eficientes. Segundo, à limitação atual dos LLMs em interpretar metadados e estatísticas internas dos SGBDs, essenciais para decisões sofisticadas de otimização.

Apesar disso, os dados revelam diferenças importantes entre os modelos:

- **O GPT-4 apresentou o menor impacto médio sobre o tempo de execução**, com aumento médio de apenas +0.097ms por consulta. Suas recomendações foram conservadoras e focadas em colunas diretamente envolvidas em filtros e junções. Esse comportamento sugere que o modelo é mais cauteloso, evitando a criação de índices redundantes ou de baixo impacto, o que o torna promissor para uso assistido por humanos em ambientes reais.
- **Claude 3.7 Sonnet teve desempenho intermediário**, com aumento médio de +0.102ms. Notavelmente, Claude sugeriu não apenas índices, mas também constraints adicionais. Embora essas estruturas possam melhorar a integridade dos dados, sua inserção sem consideração do contexto pode elevar o tempo de execução de determinadas consultas, como aquelas que envolvem subconsultas ou filtros compostos.
- **Gemini 2.0 Flash apresentou o pior desempenho entre os modelos**, com aumento médio de +0.149ms. As sugestões desse modelo parecem ter incluído índices com baixa seletividade ou aplicados a colunas com pouca frequência de uso nos filtros, o que pode ter introduzido *overhead* adicional. Esse comportamento pode indicar que o modelo foi treinado para ser mais abrangente, mas carece de mecanismos de priorização ou ponderação de impacto real.

Analisando por tipo de consulta, nota-se que as operações de agregação simples (como `COUNT(*)` e `AVG(rating)`) e os filtros diretos (`WHERE author_id = 1`) foram pouco afetadas pelas otimizações, com variações discretas entre os modelos. Por outro lado, consultas que envolvem junções (`JOIN`), subconsultas aninhadas ou agregações condicionais (`GROUP BY` com `HAVING`) apresentaram maiores variações, sugerindo que os modelos ainda encontram desafios para recomendar índices eficazes nesses contextos mais complexos.

Além disso, a consistência nos aumentos de tempo observados com Gemini e Claude indica que sugestões automatizadas sem validação podem levar a degradações perceptíveis de performance, especialmente quando aplicadas indiscriminadamente. Tais achados reforçam a necessidade de envolvimento humano no processo de decisão, ao menos enquanto os modelos não tiverem acesso direto ao otimizador interno e às estatísticas do SGBD.

Esses resultados reforçam que, embora os LLMs tenham potencial como ferramentas de apoio para tarefas de otimização, seu uso deve ser complementado por testes práticos e

conhecimento de domínio. A adoção dessas tecnologias em ambientes produtivos exige não apenas validação técnica, mas também o desenvolvimento de mecanismos de avaliação automatizada das sugestões geradas, integrados ao ciclo de monitoramento e *tuning* dos bancos de dados.

## 6. Implicações

Os achados desta pesquisa, ao comparar o impacto prático das recomendações de indexação fornecidas por diferentes LLMs em um ambiente experimental controlado, geram implicações relevantes tanto para o meio acadêmico quanto para o setor produtivo. A seguir, discutem-se os principais desdobramentos deste estudo nesses dois contextos.

**Implicações na Academia.** Este estudo contribui de forma relevante para o campo acadêmico ao propor uma metodologia experimental sistemática e reprodutível para avaliar o desempenho de LLMs em tarefas específicas de otimização de bancos de dados. Diferentemente de abordagens anteriores que se concentram em geração de código SQL ou na reescrita de consultas, este trabalho foca na análise quantitativa do impacto prático das recomendações de indexação feitas por LLMs sobre o tempo de execução das consultas. Ao aplicar um conjunto padronizado de consultas em um ambiente controlado, a pesquisa permitiu mensurar de forma objetiva o *overhead* introduzido por cada modelo, evidenciando que o GPT-4 apresentou o menor impacto médio (+0.097ms), seguido por Claude 3.7 (+0.102ms) e Gemini 2.0 (+0.149ms). Além disso, foram identificados padrões recorrentes nas estratégias de indexação sugeridas, sobretudo em consultas com junções e agregações, indicando que os modelos possuem níveis distintos de sensibilidade à estrutura das consultas. Ao estabelecer métricas claras de avaliação, o trabalho amplia a base de conhecimento sobre a aplicabilidade de LLMs no contexto de engenharia de dados, oferecendo subsídios metodológicos para investigações futuras.

**Implicações na Indústria.** Do ponto de vista da prática profissional, os resultados desta pesquisa indicam que os LLMs podem ser úteis como ferramentas complementares no processo de otimização de desempenho de bancos de dados, desde que seu uso seja acompanhado por avaliação humana criteriosa. A análise demonstrou que, embora os modelos sejam capazes de identificar colunas relevantes para indexação, nem todas as recomendações geram ganhos reais de performance e, em alguns casos, podem até introduzir *overhead* desnecessário. Isso é especialmente relevante para equipes de engenharia de dados que consideram integrar LLMs em fluxos de automação de *tuning* de banco de dados. Entre os modelos avaliados, o GPT-4 destacou-se como a opção mais estável e conservadora, com impacto mínimo sobre o tempo de execução, o que o torna mais indicado para aplicações em ambientes de produção com consultas sensíveis à performance. A pesquisa também evidencia que o uso de LLMs pode ser particularmente promissor em consultas complexas, onde a interpretação de relacionamentos e agregações demanda conhecimento especializado. No entanto, reforça-se a importância da validação empírica das sugestões antes de sua aplicação em ambientes críticos, dado o comportamento variável dos modelos e a ausência de integração com metadados e estatísticas dos sistemas de gerenciamento de banco de dados.

## 7. Considerações Finais

Este estudo teve como objetivo investigar a capacidade de Grandes Modelos de Linguagem (LLMs) — especificamente GPT-4, Claude 3.7 Sonnet e Gemini 2.0 Flash — em fornecer recomendações eficazes de indexação para otimização do desempenho de consultas em bancos de dados relacionais. Por meio de um experimento controlado, foram aplicadas sugestões geradas por cada modelo em um banco PostgreSQL e avaliados os tempos de execução de um conjunto fixo de consultas, permitindo uma análise comparativa sistemática entre os modelos.

Os resultados indicam que, embora todos os modelos sejam capazes de identificar colunas potencialmente relevantes para indexação, nem todas as sugestões geraram ganhos efetivos de performance. Em alguns casos, observou-se inclusive um leve aumento no tempo de execução, evidenciando que as recomendações precisam ser interpretadas com cautela. O GPT-4 apresentou o menor *overhead* médio, o que sugere um comportamento mais conservador e tecnicamente adequado em comparação com os demais. Essa constatação reforça parcialmente a hipótese de que LLMs podem apoiar tarefas de *tuning* em bancos de dados, mas também evidencia limitações relacionadas à falta de acesso a metadados e estatísticas do otimizador do SGBD.

Como próximos passos, recomenda-se a realização de estudos com conjuntos de dados maiores e mais heterogêneos, bem como a replicação do experimento em diferentes sistemas gerenciadores de banco de dados. Além disso, o desenvolvimento de *prompts* específicos e contextualmente orientados para tarefas de *tuning* — como filtragem, ordenação ou agregação — pode ampliar o potencial dos LLMs como assistentes técnicos no processo de otimização de desempenho em ambientes de dados reais.

## Referências

- Date, C. J. (2004). *Introdução a sistemas de bancos de dados*. Elsevier Brasil.
- de Menezes, I. V. C. B., de Oliveira Barbosa, L. S., and Garzon, N. A. (2023). Aplicações de inteligência artificial na otimização de banco de dados. *RECIMA21-Revista Científica Multidisciplinar-ISSN 2675-6218*, 4(12):e4124516–e4124516.
- Heuser, C. A. (2009). *Projeto de banco de dados*, volume 6. Bookman Porto Alegre.
- Hong, Z., Yuan, Z., Zhang, Q., Chen, H., Dong, J., Huang, F., and Huang, X. (2024). Next-generation database interfaces: A survey of llm-based text-to-sql. *arXiv preprint arXiv:2406.08426*.
- Jingru Gan, Peichen Zhong, Y. D. Y. Z. C. D. H. W. C. P. G. K. A. P. D. S.-K. W. W. (2025). Large language models are innate crystal structure generators. *arXiv:2502.20933*.
- Pereira, P. C. (2021). *Introdução a banco de dados*. São Paulo: Editora Senac.
- Sriram Dharwada, Himanshu Devrani, J. H. H. D. (2025). Query rewriting via llms. *arXiv:2502.12918*.
- Tan, J., Zhao, K., Li, R., Yu, J. X., Piao, C., Cheng, H., Meng, H., Zhao, D., and Rong, Y. (2025). Can large language models be query optimizer for relational databases? *arXiv preprint arXiv:2502.05562*.
- Yao, Y., Duan, J., Xu, K., Cai, Y., Sun, Z., and Zhang, Y. (2024). A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing*, page 100211.
- Zhou, X., Sun, Z., and Li, G. (2024). Db-gpt: Large language model meets database. *Data Science and Engineering*, 9(1):102–111.