UNIVERSIDADE DO ESTADO DA BAHIA
DEPARTAMENTO DE CIÊNCIAS EXATAS E DA TERRA
CURSO DE GRADUAÇÃO EM SISTEMAS DE INFORMAÇÃO

JÉSSICA ROCHA CARDOSO

ANÁLISE DA EFICIÊNCIA ENERGÉTICA UTILIZANDO TÉCNICAS DE OTIMIZAÇÃO DE CONSULTAS SQL

SALVADOR, BAHIA, BRASIL
2023

JÉSSICA ROCHA CARDOSO

ANÁLISE DA EFICIÊNCIA ENERGÉTICA UTILIZANDO TÉCNICAS DE OTIMIZAÇÃO DE CONSULTAS SQL

Monografia apresentada ao curso de Sistemas de Informação do Departamento de Ciências Exatas e da Terra da Universidade do Estado da Bahia - UNEB, como requisito parcial à obtenção do grau de bacharel em Sistemas de Informação. Área de Concentração: Ciência da Computação

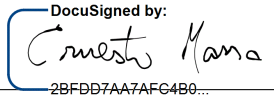
Orientador: Dr. Ernesto de Souza Massa Neto

SALVADOR, BAHIA, BRASIL

2023

Termo de Anuência do Orientador

Declaro para os devidos fins que li e revisei este trabalho e atesto sua qualidade como resultado final desta monografia. Confirmo que o referencial teórico apresentado é completo e suficiente para fundamentar os objetivos propostos e que a metodologia científica utilizada e os resultados finais são consistentes e com qualidade suficiente para submissão à banca examinadora final do Trabalho de Conclusão de Curso II do curso de Bacharelado em Sistemas de Informação.



Dr. Ernesto de Souza Massa Neto

JÉSSICA ROCHA CARDOSO

ANÁLISE DA EFICIÊNCIA ENERGÉTICA UTILIZANDO TÉCNICAS DE OTIMIZAÇÃO DE CONSULTAS SQL

Monografia apresentada ao curso de Sistemas de Informação do Departamento de Ciências Exatas e da Terra da Universidade do Estado da Bahia - UNEB, como requisito parcial à obtenção do grau de bacharel em Sistemas de Informação. Área de Concentração: Ciência da Computação

Aprovada em: 07 de Dezembro de 2023

BANCA EXAMINADORA

DocuSigned by:
Ernesto Massa
2BFDD7AA7AFC4B0...

Prof. Dr. Ernesto de Souza Massa Neto (Orientador)
Universidade do Estado da Bahia – UNEB

Prof. Dr. Cláudio Alves de Amorim
Universidade do Estado da Bahia – UNEB

Profª. Ma. Daniela Barreto Araújo
Universidade do Estado da Bahia – UNEB

A minha mãe e o meu padrasto, que mesmo com todas as dificuldades, sempre priorizaram os meus estudos e meu crescimento pessoal. Obrigada por sacrificarem tanto para garantir que eu tivesse as oportunidades que vocês não tiveram.

AGRADECIMENTOS

Dedico este trabalho com profunda gratidão àqueles que me cercaram e me apoiaram ao longo desta jornada acadêmica, bem como em minha vida pessoal.

À minha mãe, Erica, meu exemplo de força, resiliência e apreciação pela vida, e ao meu padrasto, Ubiraci, cuja presença constante em todos os momentos da minha vida trouxe conforto e estabilidade. Agradeço por serem meu alicerce, por estarem ao meu lado em todos os momentos, e por moldarem a pessoa que sou hoje. À minha família, especialmente aos meus tios e meus primos, que são a fonte constante de inspiração que me motiva a seguir em frente.

Ao meu noivo, João Gabriel, que acreditou em mim nos momentos em que eu mesma duvidava. Desde a inscrição no vestibular até a conclusão do curso, sua presença foi uma força essencial. Seu apoio, amor e confiança impulsionaram minha jornada.

A meu orientador, Professor Ernesto Massa, que me introduziu à área da computação verde e forneceu orientações importantes. Sua orientação, paciência e liberdade foram pilares essenciais para o desenvolvimento deste trabalho. À professora Daniela Araújo, que me apresentou a área de banco de dados e esteve sempre disponível com orientações e conselhos.

A minha amiga Graziele, que sempre esteve ao meu lado, compartilhando alegrias e tristezas, oferecendo apoio incondicional e amizade verdadeira. E aos meus amigos que conheci ao longo do curso, em especial a Filipe, Vitor e Reinilson. Com vocês ao meu lado, a jornada acadêmica se tornou uma experiência memorável, cheia de desafios superados e conquistas significativas.

Agradeço a todos por serem os pilares da minha trajetória e por tornarem essa jornada verdadeiramente especial.

“Não espere o futuro mudar tua vida, porque o futuro é a consequência do presente”

(Racionais MC’s)

RESUMO

Este estudo tem como objetivo analisar a eficiência energética em bancos de dados tradicionais, utilizando técnicas de otimização de consultas SQL. A pesquisa busca investigar se a aplicação dessas técnicas pode reduzir o consumo de energia e promover a sustentabilidade ambiental. Para isso, serão realizados estudos e experimentos que avaliarão o desempenho de consultas SQL em diferentes cenários, considerando métricas e indicadores relevantes. Espera-se que os resultados obtidos indiquem que a utilização de técnicas de otimização pode contribuir para a redução do consumo de energia em bancos de dados, o que pode ter um impacto significativo na sociedade e nas organizações. Com base nesses resultados esperados, este trabalho poderá fornecer avanços científicos e contribuir para o campo da otimização de consultas em SGBDs, promovendo melhores serviços, tomadas de decisões mais assertivas e um futuro mais sustentável.

Palavras-chave: Eficiência Energética. Otimização de Consultas. Sustentabilidade. Consultas SQL. Banco de Dados Verde. Computação Verde.

ABSTRACT

This study, entitled "GreenDB,"focuses on the analysis of energy efficiency in traditional database management systems (DBMS) through the implementation of SQL query optimization techniques. The research explores the feasibility of these techniques in reducing energy consumption, aiming to promote environmental sustainability. Through comprehensive studies and experiments, the performance of SQL queries was evaluated in various scenarios, taking into account both energy consumption and response time. The results highlight that strategies such as data partitioning and indexing play important roles in reducing energy consumption and improving performance. However, the creation of views proved to be less effective in terms of energy efficiency. This study contributes to the field of query optimization in DBMS, emphasizing the importance of personalized approaches that consider the specific characteristics of the database and prevalent query patterns. Additionally, it points to opportunities to extend this research to other DBMS and suggests the use of specific energy measurement tools for a more detailed analysis of energy consumption. The project represents a significant contribution to advancing energy efficiency in database management systems, aligning with the growing demand for sustainable practices in data management, promoting enhanced services, more informed decision-making, and a more sustainable future.

Keywords: Energy Efficiency. Query Optimization. Traditional Databases. Sustainability. SQL queries. Green Database. Green Computing.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE CÓDIGOS-FONTE

LISTA DE ABREVIATURAS E SIGLAS

AIE	Agência Internacional de Energia
CSV	*Comma-Separated Values*
DCL	*Data Control Language*
DDL	*Data Definition Language*
DML	*Data Manipulation Language*
DTL	*Data Transaction Language*
GPU	Unidade de Processamento Gráfico
JDK	*Java Development Kit*
JVM	*Java Virtual Machine*
PID	Identificadores de Processo
PROCEL	Programa Nacional de Conservação de Energia Elétrica
RAPL	*Running Average Power Limit*
RRAM	*Resistive Random Access Memory*
SGBD	Sistemas de Gerenciamento de Banco de Dados
SQL	*Structured Query Language*
TI	Tecnologia da Informação
TIC	Tecnologia da Informação e Comunicação

LISTA DE SÍMBOLOS

CO_2	Dióxido de carbono

SUMÁRIO

1 INTRODUÇÃO

As primeiras indicações das mudanças climáticas causadas pela ação humana surgiram durante a década de 1960, quando foram identificados aumentos na concentração de dióxido de carbono (CO2) na atmosfera, um dos principais gases responsáveis pelo efeito estufa. Em decorrência dessas descobertas, em 1979, foi realizada a primeira Conferência Mundial sobre o Clima em Genebra, na Suíça. Esse evento confirmou as evidências de que as atividades humanas tem impacto nas alterações climáticas, como a emissão de dióxido de carbono (BRASIL, 2010).

A partir deste marco inicial, as evidências científicas têm se acumulado, fornecendo uma compreensão cada vez mais clara de que o clima da Terra está passando por transformações sem precedentes. O aumento da temperatura média global tem levado a efeitos significativos, incluindo o derretimento acelerado das calotas polares e a consequente elevação do nível do mar, representando uma ameaça concreta para as comunidades costeiras. Essas mudanças têm o potencial de desencadear inundações, erosão do litoral e perda de habitats naturais. Além disso, observa-se um aumento na frequência e intensidade de eventos climáticos extremos, como tempestades e secas. Esses fenômenos climáticos têm impactos diretos abrangentes, afetando comunidades, agricultura, infraestrutura e saúde pública (MASSON-DELMOTTE et al., 2021).

Diante destes desafios, a comunidade global tem reconhecido a necessidade de agir de forma coletiva para enfrentar a mudança climática. Logo, o Acordo de Paris, assinado em 2015 por quase todos os países do mundo, estabeleceu metas ambiciosas para limitar o aquecimento global a 1,5 graus Celsius acima dos níveis pré-industriais (REI et al., 2017). Essas metas exigem ações concretas para reduzir as emissões de gases de efeito estufa, promover a transição para fontes de energia limpa e adotar práticas mais sustentáveis em todos os setores econômicos, incluindo a Tecnologia da Informação e Comunicação (TIC).

No contexto da Tecnologia da Informação e Comunicação (TIC), a expressão "verde" refere-se a estratégias e abordagens que visam conservar os recursos naturais e reduzir o impacto ambiental. Essa abordagem está diretamente relacionada à transição para uma economia de baixo carbono, que se caracteriza pela busca constante de minimizar as emissões de gases de efeito estufa e reduzir a pegada de carbono, promovendo práticas sustentáveis e

tecnologias mais limpas. No entanto, apesar de desempenhar um papel fundamental nessa transição, a TIC também é responsável por uma parcela significativa das emissões de gases de efeito estufa, devido ao consumo energético em **data centers**, redes de telecomunicações e dispositivos eletrônicos (MALMODIN; LUNDéN, 2018). Portanto, torna-se essencial explorar soluções sustentáveis que tornem a TIC mais eficiente e ecologicamente consciente.

Diante desses desafios, a busca por soluções tecnológicas que contribuam para a sustentabilidade ambiental ganha relevância. As TIC verdes envolvem as aplicações e práticas sustentáveis no desenvolvimento e uso de sistemas de tecnologia da informação e comunicação. Isso abrange desde a eficiência energética e o uso de fontes renováveis de energia em *data centers* até a adoção de design ecoeficiente em dispositivos eletrônicos. Autores como Smith e Koulamas (2017) ressaltam que as TIC verdes têm o potencial de reduzir significativamente as emissões de carbono e minimizar o impacto ambiental dos sistemas de tecnologia, otimizar recursos computacionais e prolongar a vida útil de dispositivos, contribuindo assim para a mitigação das mudanças climáticas e alinhando-se com metas globais de sustentabilidade (SMITH; KOULAMAS, 2017).

No entanto, um componente-chave da TIC que também requer atenção é a eficiência energética dos sistemas de gerenciamento de banco de dados, pois eles são essenciais para o armazenamento e processamento de informações em uma ampla variedade de setores, como finanças, saúde e transporte. Estima-se que a indústria de tecnologia seja responsável por aproximadamente 7% do consumo global de eletricidade, tornando-se um dos setores de maior crescimento no consumo de energia (COOK et al., 2017).

Em particular, a eficiência energética em sistemas de gerenciamento de banco de dados torna-se um desafio significativo, considerando especialmente o crescente volume de dados gerados e armazenados pela sociedade atual. Os sistemas de grande porte, como os *data centers*, consomem quantidades significativas de energia elétrica, resultando em altos custos operacionais e impactos ambientais consideráveis. Por exemplo, um centro de dados típico pode consumir energia equivalente a 25.000 casas, e consome 100-200 vezes mais energia do que um escritório padrão do mesmo tamanho (POESS; NAMBIAR, 2008). Diante desse cenário, a busca por soluções que permitam reduzir o consumo energético desses sistemas, sem comprometer seu desempenho e funcionalidades, é um desafio que tem motivado pesquisadores e profissionais da área a investigar e desenvolver abordagens inovadoras, visando reduzir o impacto ambiental e

contribuir para a sustentabilidade.

Uma abordagem promissora é a otimização de consultas em bancos de dados tradicionais. As consultas representam uma parte significativa da carga de trabalho dos bancos de dados, e melhorias nessa área podem resultar em redução do consumo de energia. Logo, a otimização de consultas envolve a busca por algoritmos e técnicas eficientes que possam executar as consultas de forma mais rápida e econômica em termos de energia (ROUKH et al., 2016). Essas estratégias visam minimizar o uso de recursos, resultando em uma diminuição do impacto ambiental causado pelos SGBD.

Neste cenário, este estudo teve como objetivo central investigar e implementar técnicas de otimização de consultas em SGBD com o intuito de reduzir o consumo de energia desses sistemas, promovendo, assim, a sustentabilidade ambiental. Para alcançar esse propósito, foram realizadas revisões da literatura, identificando desafios e lacunas existentes, e foram implementados algoritmos e estratégias que visaram não apenas melhorar o desempenho das consultas, mas também minimizar o uso de recursos energéticos. Por meio de experimentos e análises comparativas, avaliou-se o impacto dessas técnicas no tempo de resposta das consultas, no consumo de energia dos SGBD e na utilização eficiente dos recursos computacionais disponíveis.

Neste contexto, a pergunta de pesquisa que norteia este estudo é: "Quais estratégias e técnicas de otimização de consultas SQL podem ser mais eficazes na redução do consumo de energia em SGBD, levando em consideração diversos cenários de uso e tipos de consultas, e como essas abordagens impactam o desempenho das operações realizadas?"

Esta pergunta de pesquisa visa investigar de forma mais aprofundada como a otimização de consultas SQL pode ser aplicada para melhorar a eficiência energética em bancos de dados, considerando diferentes fatores que podem influenciar o consumo de energia e a qualidade das operações executadas. A execução deste trabalho se baseia na motivação de abordar o tema da otimização de consultas em SGBD e sua relevância social, econômica e acadêmica, visando contribuir para a área de conhecimento ao fornecer avanços científicos e acadêmicos, bem como abordar lacunas no conhecimento existente e proporcionar benefícios práticos para a sociedade.

Diversos estudos têm sido realizados com o objetivo de aprimorar as técnicas de otimização de consultas em SGBDs, evidenciando a importância do tema. Nesse contexto, a revisão sistemática realizada como parte deste projeto destaca a relevância de abordar a eficiência

energética como um critério importante na otimização de consultas em bancos de dados, a exemplo de (MAHAJAN; ZONG, 2018; MAHAJAN et al., 2019). A eficiência energética é uma preocupação crescente em sistemas de banco de dados, especialmente devido ao aumento no consumo de energia e às preocupações com a sustentabilidade ambiental (MACHINERY, 2015).

Além disso, estudos recentes têm enfatizado a necessidade de aprimorar as técnicas de otimização de consultas, levando em consideração o crescente volume de dados e a diversidade de consultas realizadas em diferentes domínios (MARCUS et al., 2021). Melhorar o desempenho dos SGBD e otimizar o processamento de consultas são fatores-chave para garantir a eficiência e a agilidade das operações de recuperação de informações.

Portanto, a realização deste projeto de pesquisa é uma oportunidade de envolvimento em uma área de conhecimento relevante, com impacto direto na sociedade e nas organizações. Ao fornecer avanços científicos, abordar lacunas no conhecimento existente e considerar aspectos como eficiência energética, busco contribuir de maneira significativa para o campo da otimização de consultas em SGBDs, promovendo melhores serviços, tomadas de decisões mais assertivas e um futuro mais sustentável.

Este trabalho visa abordar a sustentabilidade na área de Banco de Dados, com foco na eficiência energética. Após a introdução, onde se destaca a relevância da sustentabilidade diante dos desafios ambientais na Tecnologia da Informação, os capítulos subsequentes abrangem diversos aspectos do tema. "Banco de Dados Verde" oferece uma visão geral dos conceitos, com ênfase na eficiência energética e na redução de consumo. O "Projeto GreenDB" detalha a estratégia de otimização de consultas SQL, a coleta de dados, a preparação de dados com PySpark, a estrutura das tabelas e a mensuração do consumo de energia com ferramentas como Powerjoular e JoularJX. Os testes e resultados são discutidos no capítulo "Testes e Resultados", abordando particionamento, indexação, views e apresentando resultados de consumo energético e tempo de resposta. Por fim, as "Considerações Finais" recapitulam os principais resultados e apontam possíveis direções futuras de pesquisa. Este trabalho, embasado em uma abordagem multidisciplinar, contribui para a compreensão e aplicação de conceitos de eficiência energética, Banco de Dados e sustentabilidade, com o objetivo de promover um futuro mais sustentável no campo da Tecnologia da Informação.

2 BANCO DE DADOS VERDE

Os bancos de dados atuam no armazenamento e gerenciamento de informações em diversos setores da indústria. Conforme definido por Korth e Silberschatz (KORTH; SILBERSCHATZ, 1994), um banco de dados é uma compilação de dados entrelaçados, representando informações sobre um âmbito específico. Essa estrutura possibilita o agrupamento de informações relacionadas e aborda um mesmo tema, fornecendo uma base sólida para a tomada de decisões e a execução eficiente de operações.

No entanto, o crescimento exponencial da demanda por dados tem levado a um aumento significativo no consumo de energia dos *data centers* que abrigam esses bancos de dados. Esse alto consumo energético suscita preocupações sobre a sustentabilidade e o impacto ambiental desses sistemas.

É importante ressaltar que o custo da eletricidade tornou-se uma parte significativa dos custos operacionais dos *data centers*. Diversas projeções, incluindo as de (HINTEMANN; HINTERHOLZER, 2019), evidenciam a variabilidade nos cenários futuros. No "melhor caso", espera-se que o consumo de energia dos *data centers* permaneça constante; no entanto, se as tendências atuais persistirem, há a estimativa de que o consumo de energia dos *data centers* possa dobrar até 2030 em comparação com 2019. Essas perspectivas diversificadas destacam a importância de estratégias proativas para enfrentar os desafios energéticos e fomentar a sustentabilidade nos *data centers*. Nesse contexto, é imperativo buscar soluções que impulsionem a eficiência energética, visando a redução de gastos e a promoção de práticas que tornem as operações dos *data centers* mais sustentáveis. Além disso, a consideração cuidadosa do consumo de energia para alimentar e resfriar os servidores é essencial para alcançar uma abordagem verdadeiramente sustentável.

Para enfrentar esses desafios, a comunidade de pesquisa e desenvolvimento tem concentrado esforços na busca por soluções que tornem os bancos de dados mais eficientes e sustentáveis.

2.1 COMPUTAÇÃO VERDE

A Computação Verde surgiu como uma resposta aos crescentes problemas ambientais causados pelo avanço da TIC e pelo aumento significativo do consumo de energia relacionado a essas atividades (MURUGESAN, 2008). À medida que a computação se tornava cada vez mais presente em nossas vidas, surgiram desafios relacionados à eficiência energética, ao descarte de resíduos eletrônicos e à pegada de carbono gerada pelas atividades de Tecnologia da Informação (TI) (MOI et al., 2014).

A expressão "pegada de carbono"refere-se à quantidade total de gases de efeito estufa, principalmente dióxido de carbono, liberados diretamente ou indiretamente por uma pessoa, organização, evento ou produto ao longo de seu ciclo de vida. Em outras palavras, representa o impacto ambiental em termos de contribuição para o aquecimento global. No contexto da Computação Verde, a pegada de carbono está relacionada às emissões de gases de efeito estufa associadas às atividades de TI, como o uso de energia para alimentar servidores, dispositivos eletrônicos e centros de dados.

O rápido desenvolvimento tecnológico e o aumento no uso de dispositivos eletrônicos, como computadores, *smartphones* e servidores, levaram a um aumento substancial no consumo de energia desses dispositivos. Por exemplo, os *data centers*, que abrigam uma grande quantidade de servidores e sistemas de armazenamento de dados, são conhecidos por serem grandes consumidores de energia. De acordo com um estudo da Agência Internacional de Energia (AIE), os *data centers* são responsáveis por cerca de 1% do consumo global de eletricidade e esse número tende a aumentar significativamente nos próximos anos (Agência Internacional de Energia, 2022).

Além disso, o descarte inadequado de resíduos eletrônicos, como computadores obsoletos, telefones celulares e outros dispositivos eletrônicos descartados, tem um impacto significativo na poluição ambiental (BEULA; SURESHKUMAR, 2021). Esses resíduos contêm substâncias tóxicas, como chumbo, mercúrio e cádmio, que representam sérios riscos para o meio ambiente e para a saúde humana caso não sejam descartados corretamente (BEULA; SURESHKUMAR, 2021). Portanto, é essencial adotar práticas adequadas de gerenciamento e reciclagem de resíduos eletrônicos, a fim de minimizar os danos ambientais e proteger a saúde pública.

Outra questão é a dependência de fontes não renováveis de energia, como combustíveis fósseis, para alimentar os sistemas computacionais também apresenta problemas significativos. A queima desses combustíveis emite gases de efeito estufa, como dióxido de carbono, contribuindo para o aquecimento global e as mudanças climáticas.

A conscientização sobre esses problemas ambientais e a necessidade de uma abordagem mais sustentável para a computação levaram ao surgimento do termo "Computação Verde". A Computação Verde visa reduzir o impacto ambiental da tecnologia da informação, adotando práticas e soluções que promovam a eficiência energética, a utilização de fontes de energia renovável e a gestão adequada dos resíduos eletrônicos.

Para alcançar esses objetivos, a Computação Verde abrange uma ampla gama de áreas, incluindo o projeto de *hardware* e *software* energeticamente eficientes, o desenvolvimento de algoritmos otimizados para minimizar o consumo de energia, a virtualização de servidores para melhorar a utilização de recursos, o gerenciamento inteligente de energia em *data centers*, o uso de técnicas de refrigeração eficientes e a conscientização dos usuários finais sobre a importância da utilização responsável da tecnologia.

Sendo assim, diversas iniciativas têm sido adotadas para promover a Computação Verde, como organizações governamentais e não governamentais, juntamente com empresas de tecnologia, têm se engajado em projetos e programas que visam incentivar a adoção de práticas sustentáveis na área da computação. Além disso, pesquisas científicas e acadêmicas continuam a explorar novas soluções e abordagens para melhorar a eficiência energética e reduzir o impacto ambiental da computação.

Uma iniciativa governamental notável é o programa "Programa de Eficiência Energética", que visa promover práticas de redução de energia em agências governamentais do país (ANEEL, 2022). Sendo assim, esse programa estabeleceu metas ambiciosas de eficiência energética e redução de emissões de carbono nas operações de TI do governo brasileiro.

No setor não governamental, empresas líderes em tecnologia também têm desempenhado um papel fundamental na promoção da Computação Verde. A Apple, por exemplo, comprometeu-se a tornar toda a sua cadeia de suprimentos neutra em carbono até 2030 (APPLE, 2021). Além disso, o Google anunciou que, até 2030, suas operações serão 100% alimentadas por energia renovável (GOOGLE, 2022).

Além das iniciativas de Computação Verde mencionadas anteriormente, uma área específica que se destaca nesse contexto é a de Banco de Dados Verdes. Os bancos de dados desempenham um papel fundamental na maioria dos sistemas de informação, armazenando e gerenciando grandes quantidades de dados. No entanto, o armazenamento e processamento desses dados também consomem uma quantidade significativa de energia com impacto direto no consumo energético e na sustentabilidade dos sistemas de TI.

2.1.1 Eficiência energética e redução do consumo de energia

A eficiência energética e a redução do consumo de energia são temas relevantes, tanto no contexto brasileiro quanto globalmente. Diversas iniciativas têm sido estabelecidas para promover o uso racional de energia e combater o desperdício. Um exemplo é o Programa Nacional de Conservação de Energia Elétrica (PROCEL), criado em dezembro de 1985 e coordenado pelo Ministério de Minas e Energia, em parceria com a Eletrobras (PROCEL, 1985). Esse programa tem como objetivo principal fomentar a eficiência energética e conscientizar sobre a importância de utilizar os recursos energéticos de forma eficiente.

Além disso, a Lei de Eficiência Energética, estabelecida pela Lei nº 10.295/2001, é outra legislação relevante que busca promover a alocação eficiente dos recursos energéticos e a preservação do meio ambiente (BRASIL, 2001). Essa lei estabelece uma política nacional de conservação e uso racional de energia, determinando a criação de um Programa de Metas que impõe obrigações aos fabricantes e consumidores no sentido de atender aos requisitos de consumo de energia e eficiência energética.

No contexto específico dos bancos de dados, a eficiência energética desempenha um papel fundamental. Autores como Beloglazov et al. (BELOGLAZOV et al., 2012) e Yan et al. (YAN et al., 2018) destacam a importância de otimizar o consumo de energia em ambientes de *data centers*, onde os bancos de dados estão hospedados. Eles ressaltam que a adoção de estratégias eficientes, como o uso de algoritmos de economia de energia, o provisionamento dinâmico de recursos e a virtualização, pode resultar em reduções significativas nos custos operacionais e na pegada de carbono desses *data centers*.

Outros estudos, como os de Luo et al. (LUO et al., 2015) e Liu et al. (LIU et al., 2016), exploram técnicas específicas para melhorar a eficiência energética em bancos de dados. Eles propõem estratégias como a otimização da execução de consultas, a compactação de dados

e o escalonamento dinâmico de recursos, visando reduzir o consumo de energia durante as operações de processamento de dados. Essas abordagens demonstram um potencial significativo para alcançar ganhos em termos de eficiência energética e sustentabilidade.

Diante disso, a implementação de estratégias eficientes nos bancos de dados é essencial para minimizar o impacto ambiental e otimizar o uso de recursos energéticos. A conscientização sobre a importância dessas práticas é fundamental, assim como o cumprimento das legislações pertinentes. A combinação de iniciativas governamentais, como o PROCEL e a Lei de Eficiência Energética, juntamente com o conhecimento e a aplicação de técnicas eficientes pelos profissionais da área, contribuirá para promover a sustentabilidade e a redução do consumo de energia nos bancos de dados.

2.1.2 Iniciativas na área de Banco de Dados

Na busca por soluções sustentáveis, a área de banco de dados verde tem testemunhado iniciativas que abrangem otimizações no consumo de energia durante consultas, inovações em arquiteturas de armazenamento, virtualização de bancos de dados e técnicas avançadas de compressão de dados.

A otimização do consumo de energia durante consultas (POGODAEV; RYZHKOVA, 2020; CHOI et al., 2016) foca em estratégias para reduzir o tempo de execução, envolvendo a reorganização de consultas e o uso eficiente de Unidade de Processamento Gráfico (GPU)s. Arquiteturas inovadoras, como aquela baseada em memórias RRAM (SUN et al., 2018), destacam-se por oferecerem maior eficiência energética.

Outra área de pesquisa importante é a criação de arquiteturas de armazenamento mais eficientes, como abordado no artigo (SUN et al., 2018). Essa pesquisa propõe uma estrutura inovadora de armazenamento em memória baseada em memórias de acesso aleatório resistivas *Resistive Random Access Memory* (RRAM), que permite uma execução mais eficiente das consultas, resultando em menor consumo de energia. A utilização de RRAM como arquitetura de armazenamento oferece benefícios significativos, como menor latência e menor consumo de energia, o que pode contribuir para um processamento mais eficiente das consultas em bancos de dados.

A virtualização de bancos de dados também é uma área relevante de pesquisa,

conforme discutido no artigo (SALAMI et al., 2017). Essa técnica permite a consolidação de vários bancos de dados em um único servidor físico, reduzindo a necessidade de infraestrutura e energia nos *data centers*. A virtualização oferece vantagens, como melhor utilização dos recursos de *hardware*, maior flexibilidade e menor consumo de energia, contribuindo para a sustentabilidade dos bancos de dados.

Além das contribuições mencionadas, existem outras iniciativas relevantes no campo de bancos de dados verdes. Uma delas é o uso de técnicas avançadas de compressão de dados, conforme descrito no artigo (GOETZ et al., 2014). Essa abordagem busca reduzir o espaço de armazenamento necessário para os dados, resultando em menor consumo de energia tanto para o armazenamento em disco quanto para a transmissão de dados.

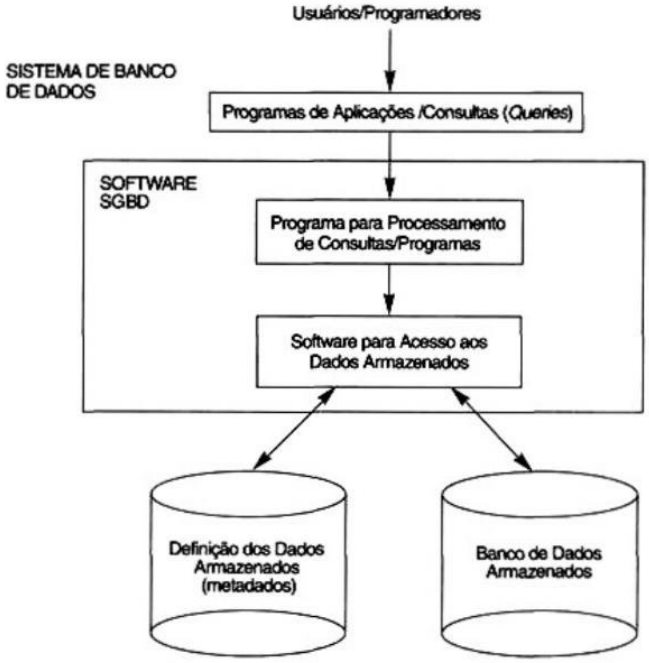
Ao adotar essas abordagens e tecnologias, é possível avançar em direção a um modelo de banco de dados mais eficiente e sustentável, reduzindo tanto os custos operacionais como as emissões de carbono associadas aos *data centers*. Logo, essas iniciativas refletem o compromisso em promover a eficiência energética no contexto dos bancos de dados, contribuindo para um futuro mais sustentável. Sendo assim, a contínua pesquisa e inovação nessa área são essenciais para enfrentar os desafios da demanda crescente por dados e para aprimorar a eficiência e a sustentabilidade dos bancos de dados.

2.2 SISTEMAS DE GERENCIAMENTO DE BANCO DE DADOS

Um componente essencial entre o banco de dados e os usuários é o SGBD. Ele tem como principal objetivo fornecer um ambiente conveniente e eficiente para o armazenamento e recuperação das informações do banco de dados (SILBERSCHATZ et al., 1999).

Sendo assim, o SGBD disponibiliza recursos para definir, construir, manipular, compartilhar, proteger e manter bancos de dados (ELMASRI; NAVATHE, 2011). A definição envolve a especificação das estruturas de armazenamento, como os elementos e tipos de dados que compõem os registros. A construção diz respeito ao armazenamento dos dados, incluindo registros e relacionamentos. A manipulação engloba a recuperação e atualização dos dados, como inclusões, exclusões e alterações. O compartilhamento refere-se à permissão de acesso concorrente a um banco de dados. A proteção está relacionada à segurança contra falhas de *hardware*, *software* e acesso não autorizado. Já a manutenção oferece suporte ao crescimento contínuo do banco de dados (ELMASRI; NAVATHE, 2011).

Figura 1 – Representação simplificada de um SGBD.



Fonte: (FERREIRA, 2018).

Dessa forma, o SGBD desempenha um papel fundamental na gestão de bancos de dados, como ilustrado na Figura 1. O SGBD atua como uma camada de abstração que isola os usuários e os programadores da complexidade subjacente do banco de dados. Ele oferece um ambiente seguro e eficiente para armazenar, recuperar e manipular informações, ao mesmo tempo, em que simplifica a interação dos usuários e programadores com o sistema. No diagrama, observa-se que os usuários se comunicam com o SGBD por meio de consultas (*queries*), e o SGBD, por sua vez, possui sistemas e mecanismos internos para processar e acessar os dados armazenados. Além disso, o SGBD gerencia as definições de dados armazenados, conhecidas como metadados, tornando mais eficiente o controle e a organização dos recursos do banco de dados.

2.2.1 Processamento de Consultas

A linguagem de consulta *Structured Query Language* (SQL) é uma linguagem padrão utilizada para interagir com bancos de dados relacionais. Ela permite a criação, manipulação e recuperação de dados armazenados em um banco de dados, oferecendo uma ampla gama de comandos e instruções (SHAR; TAN, 2012).

Dentro do escopo da linguagem SQL, seus comandos se dividem em categorias distintas: A *Data Manipulation Language* (DML) abrange comandos que focalizam a manipulação,

alteração e atualização de dados em um banco de dados. Por meio desses comandos, é possível inserir, atualizar ou excluir registros, bem como recuperar informações específicas. A *Data Definition Language* (DDL) abarca os comandos empregados na administração e estruturação das tabelas. Isso inclui a criação, alteração ou remoção de tabelas, além de definir restrições e índices. A *Data Control Language* (DCL) envolve os comandos que gerenciam permissões e acessos ao banco de dados. Ela assegura a segurança e o controle adequado sobre quem pode executar determinadas operações. A *Data Transaction Language* (DTL) aborda comandos que asseguram a consistência dos dados ao realizar alterações. Isso inclui a capacidade de iniciar, confirmar ou reverter transações, mantendo a integridade dos dados.

Por meio dessas divisões, a linguagem SQL oferece um conjunto robusto de ferramentas para a interação com bancos de dados relacionais, permitindo uma administração eficiente dos dados e operações.

A complexidade do processamento de consulta em um SGBD é evidenciada por uma série de etapas interconectadas que visam extrair informações de um banco de dados com base em consultas SQL. Como ilustrado na Figura 2. Essas atividades abrangem desde a tradução inicial da consulta em uma linguagem compreensível pelo banco de dados até a otimização da consulta e a definição de um plano de execução eficaz (SILBERSCHATZ et al., 2016).

Antes de iniciar o processamento da consulta, é necessário traduzi-la para uma linguagem utilizável pelo banco de dados relacional, uma vez que a linguagem SQL não é ideal para a representação interna do sistema de consulta. Para isso, é realizada uma análise sintática da consulta, similar ao analisador sintático de um compilador. O sistema constrói uma representação em forma de árvore de análise da consulta, que é então traduzida para uma expressão da álgebra relacional.

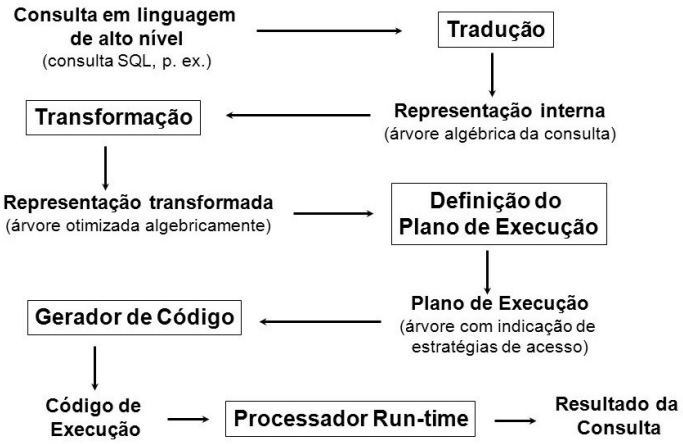
No entanto, apenas fornecer a expressão da álgebra relacional não é suficiente para especificar completamente como avaliar a consulta. É necessário adicionar instruções que indicam como avaliar cada operação da álgebra relacional, como algoritmos a serem utilizados. A representação da álgebra relacional de uma consulta orienta parcialmente como avaliá-la, mas para uma execução completa e eficaz, é necessário incluir instruções detalhadas sobre os métodos a serem empregados.(SILBERSCHATZ et al., 2016)).

Existem várias maneiras de avaliar as expressões da álgebra relacional, conhecidas

como planos de execução. Diferentes planos de execução para uma consulta podem ter diferentes custos, o que significa que o sistema deve escolher o plano mais eficiente em termos de desempenho para executar a consulta.

Dessa forma, o processamento de consulta envolve a tradução da consulta para uma forma utilizável pelo banco de dados, a otimização da consulta, a construção de um plano de execução eficiente e, finalmente, a execução real da consulta para obter os resultados desejados do banco de dados.

Figura 2 – Etapas no Processamento da Consulta.



Fonte: SlideToDoc. Disponível em: <https://slidetodoc.com/sumrio-1-sql-embutida-2-processamento-de-consultas/> Acesso em: 11 de setembro de 2023.

2.2.2 Otimização de Consulta

A otimização de consultas desempenha um papel essencial nos Sistemas de Gerenciamento de Bancos de Dados (SGBDs), visando aprimorar o desempenho e a eficiência das consultas. Além dos fatores tradicionais como tempo de resposta, consumo de recursos e custo computacional, há uma crescente necessidade de considerar o consumo de energia como um parâmetro vital nesse processo.

No cenário da otimização, a escolha do plano de execução mais eficaz entre diversas estratégias disponíveis para processar uma consulta específica é um passo crucial. Essa etapa se torna ainda mais vital ao lidar com consultas complexas, uma vez que as consultas mais simples já contam com otimização em seu processamento. Nesse contexto, o SGBD desenvolve um plano de execução destinado a minimizar os custos associados à execução da consulta em

questão (SILBERSCHATZ et al., 2016).

Ao receber uma consulta, o SGBD inicia um processo de otimização, que inclui a análise da consulta, a geração de planos de execução alternativos e a seleção do plano mais adequado, baseando-se em heurísticas e estimativas de custo. Nessa etapa, são empregadas técnicas como a estimativa de custo do plano de execução, considerando informações e estatísticas sobre as relações envolvidas na consulta. Adicionalmente, a escolha dos algoritmos de execução das operações e a utilização de índices específicos contribuem para melhorar o desempenho das consultas.

Portanto, a otimização de consultas busca aprimorar a eficiência global do sistema de banco de dados, buscando não apenas a redução do tempo de resposta das consultas, mas também a otimização do consumo de recursos e, de forma proeminente, do consumo de energia. Nas próximas seções, serão exploradas as técnicas de otimização de consulta, abrangendo tanto a otimização baseada em custo quanto a utilização de heurísticas, com um foco especial na otimização do consumo de energia.

2.2.2.1 Otimização Baseada em Custo

A técnica de otimização de consulta baseada em custo é empregada para estimar e comparar os custos de execução de uma consulta usando diferentes estratégias de execução, selecionando aquela com a menor estimativa de custo. Essa abordagem leva em consideração informações do dicionário de dados, estatísticas e parâmetros que não são influenciados pela sintaxe da consulta SQL(ELMASRI; NAVATHE, 2011).

O processo de otimização baseado em custo gera vários planos de execução de consulta a partir de uma determinada consulta, utilizando regras de equivalência, e seleciona aquele com o menor custo (SILBERSCHATZ et al., 2016). Conforme mencionado por Elmasri e Navathe, para estimar os custos das várias estratégias de execução, é necessário manter atualizadas as informações necessárias para as funções de custo, as quais podem ser armazenadas no catálogo do SGBD e acessadas pelo otimizador de consulta (ELMASRI; NAVATHE, 2011).

No entanto, caso as estatísticas estejam ausentes ou desatualizadas, o otimizador não terá as informações básicas necessárias para o processo de otimização da consulta. Por esse motivo, essa técnica é mais adequada para consultas compiladas, em que o processo de

otimização é realizado durante a compilação e a estratégia selecionada é armazenada e executada diretamente em tempo de execução (ELMASRI; NAVATHE, 2011).

Logo, a otimização baseada em custo utiliza algoritmos e técnicas avançadas para explorar o espaço de planos de execução possíveis e selecionar o mais eficiente. Essa abordagem leva em consideração a estrutura do banco de dados, as propriedades das consultas e as estatísticas disponíveis para tomar decisões inteligentes durante o processo de otimização. Por conta disso, é importante ressaltar que a otimização baseada em custo busca soluções para um problema e uma solução que minimize uma função de custo, no entanto, essas funções são apenas estimativas, e o otimizador pode escolher uma estratégia de solução que não seja a ótima.

2.2.2.2 Utilização de Heurísticas na Otimização de Consultas

A utilização de heurísticas na otimização de consultas apresenta vantagens em relação ao método baseado em custo. Enquanto o método baseado em custo requer a seleção do plano de execução com base em estimativas armazenadas nos catálogos do sistema de banco de dados, o que pode ser um processo custoso devido ao grande número de planos de execução possíveis, a abordagem heurística trabalha diretamente na álgebra relacional, aplicando regras definidas para gerar um plano de execução mais eficiente, sem depender de análises estatísticas (SILBERSCHATZ et al., 2016).

A otimização baseada em heurística, também conhecida como otimização algébrica, utiliza as regras de otimização na forma algébrica de uma consulta. A partir de uma forma algébrica gerada pela consulta SQL, são aplicadas as regras de equivalência da álgebra relacional para obter uma nova forma algébrica. No entanto, é fundamental garantir que as transformações realizadas resultem em uma expressão equivalente da consulta (SILBERSCHATZ et al., 2016).

Essa abordagem heurística permite explorar propriedades e características das consultas e das estruturas de dados, utilizando conhecimento prévio para tomar decisões eficientes durante o processo de otimização. As regras de otimização são aplicadas de forma iterativa, buscando melhorias incrementais no plano de execução da consulta.

É importante destacar que, embora as heurísticas sejam eficazes na otimização de consultas, elas não garantem a obtenção do plano de execução ótimo em todos os casos. No entanto, ao trabalhar diretamente com a álgebra relacional, essa abordagem oferece agilidade e

eficiência na geração de planos de execução, sem depender de análises estatísticas extensas.

2.2.2.3 Técnica de Indexação

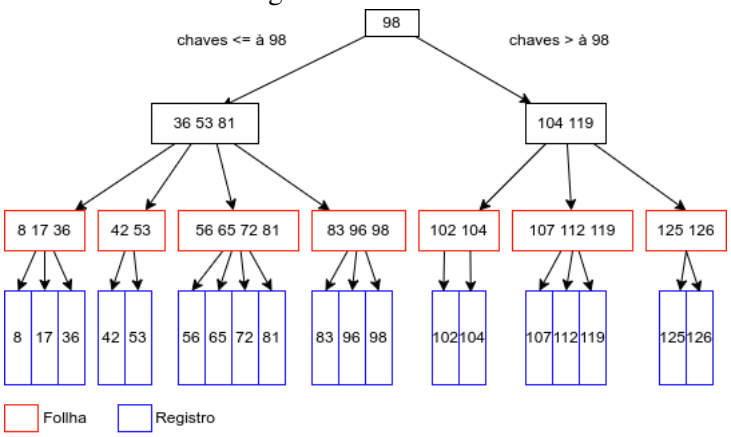
A técnica de indexação é um elemento fundamental no projeto e gerenciamento de bancos de dados, proporcionando um acesso rápido e eficiente aos dados armazenados. Diversos autores têm contribuído para o desenvolvimento e aprimoramento das técnicas de indexação, resultando em abordagens eficazes para otimizar as operações de busca e recuperação de dados (BAYER; MCCREIGHT, 1972; LITWIN et al., 1980; HÉMAN et al., 2012).

A indexação por meio de árvores B, originalmente proposta por Bayer e McCreight em 1972 (BAYER; MCCREIGHT, 1972), é uma técnica amplamente adotada para a organização eficiente de dados em sistemas de gerenciamento de bancos de dados. Esta abordagem utiliza uma estrutura de árvore balanceada que facilita operações como divisão e fusão de páginas, proporcionando buscas eficientes. A árvore B é especialmente eficaz em ambientes onde inserções e exclusões frequentes ocorrem, pois ela mantém automaticamente o equilíbrio, assegurando um desempenho consistente.

Na Figura 3, é possível observar a estrutura e a organização típica de uma árvore B. Os nós folhas estão destacados em vermelho, enquanto os nós azuis representam os registros contidos nesses nós. Importante notar que os nós folhas contêm os dados reais, armazenados em um vetor, enquanto os nós internos contêm exclusivamente chaves e ponteiros para os nós filhos. A característica fundamental de uma árvore B é a organização ordenada de suas chaves. No primeiro nó, identificado na Figura, a chave possui o valor 98. Notavelmente, à direita desse nó, encontram-se chaves com valores superiores a 98, enquanto à esquerda estão aquelas cujos valores são menores ou iguais a 98. Essa disposição estratégica permite uma rápida e eficiente busca binária, otimizando o processo de localização de chaves específicas na árvore.

Essa estratégia tem como objetivo otimizar operações de busca e gerenciamento de dados em sistemas de banco de dados, tornando as operações de inserção, exclusão e recuperação de informações mais eficientes e consistentes. O equilíbrio automático proporcionado pela árvore B é importante em ambientes dinâmicos, onde as modificações frequentes nos dados demandam uma estrutura que se adapte de forma eficaz, garantindo um desempenho otimizado em todas as operações.

Figura 3 – Árvore B.



Fonte: UnB/FGA - CAE. Disponível em: <https://sae.unb.br/cae/conteudo/unbfga/lbd/banco2_indices.html> Acesso em: 11 de setembro de 2023.

A técnica de indexação por *hashing*, proposta por Litwin et al. (1980), oferece uma abordagem distinta para a organização eficiente de dados. Nessa estratégia, cada valor de chave é diretamente mapeado para um endereço de memória física, permitindo a busca direta e instantânea dos dados correspondentes. A Figura 4 esquematiza o funcionamento desse método, destacando a associação de cada chave a um endereço de memória física, onde os dados são armazenados.

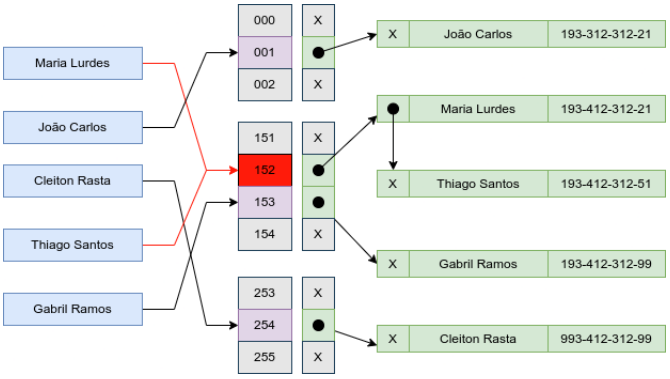
A indexação por *hashing* é especialmente eficaz em operações de busca por igualdade, quando o valor específico da chave a ser pesquisada é conhecido. No entanto, essa abordagem pode enfrentar desafios em situações de colisão, que ocorrem quando dois ou mais valores de chave são mapeados para o mesmo endereço de memória.

No exemplo apresentado na Figura 4, podemos observar uma colisão no índice 152, onde as chaves "Maria Lurdes"e "Thiago Santos"são mapeadas para o mesmo endereço. Para contornar essa situação, é implementado o encadeamento de endereços, uma técnica na qual os elementos que colidem são organizados em uma estrutura de lista encadeada. No caso específico, a chave de "Maria Lurdes"aponta para "Thiago Santos", estabelecendo uma relação que permite a resolução da colisão.

Essa estratégia de encadeamento de endereços é fundamental para preservar a integridade e a eficiência da indexação por *hashing* em cenários de colisão, garantindo que os dados associados a chaves conflitantes possam ser acessados e manipulados adequadamente.

Uma técnica mais recente é a indexação por árvores B+ com compressão, proposta por Héman et al. (2012). Essa técnica combina os benefícios da estrutura de árvore B+ com a

Figura 4 – Estrutura de índice de hashing.



Fonte: UnB/FGA - CAE. Disponível em: <https://sae.unb.br/cae/conteudo/unbfga/lbd/banco2_indices.html> Acesso em: 11 de setembro de 2023.

compressão de dados, visando reduzir o espaço de armazenamento necessário para os índices. A compressão permite que mais dados sejam armazenados em cada página da árvore, resultando em uma redução no número total de páginas e, consequentemente, em uma melhoria no desempenho das operações de busca.

Além disso, a indexação por bitmap, introduzida por Wu et al. (2001), é uma técnica eficiente para lidar com dados categóricos. Nessa abordagem, cada valor distinto de uma coluna é mapeado para um bitmap, que representa a presença ou ausência desse valor em cada registro. Esses bitmaps são combinados por operações lógicas para realizar consultas eficientes em dados categóricos.

Outra técnica relevante é a indexação invertida, proposta por Zobel e Moffat (2006). Essa técnica é amplamente utilizada em sistemas de recuperação de informações, como mecanismos de busca na Web. A indexação invertida consiste em construir um índice que mapeia termos encontrados nos documentos para os documentos em que eles ocorrem, permitindo uma recuperação rápida de informações com base em termos de busca.

Essas são algumas das principais técnicas de indexação utilizadas na otimização de consultas em bancos de dados. Cada uma delas possui características distintas e é mais adequada para determinados tipos de consultas e cenários de uso. Portanto, a escolha da técnica de indexação apropriada depende das características do banco de dados e dos requisitos específicos de desempenho e eficiência da aplicação.

2.2.2.4 Particionamento de Dados

O particionamento de dados é uma estratégia fundamental para otimizar consultas em sistemas de gerenciamento de bancos de dados (SGBDs). Essa abordagem envolve a divisão de extensos conjuntos de dados em unidades menores chamadas partições, visando aprimorar o desempenho e a eficiência das consultas. O particionamento de um banco de dados não apenas aprimora o desempenho, mas também simplifica a manutenção do sistema. Ao dividir uma tabela volumosa em tabelas menores e independentes, as consultas que acessam apenas uma fração dos dados podem ser executadas com maior rapidez, uma vez que há menos dados a serem processados (MICROSOFT, 2008).

O particionamento de dados oferece a vantagem de distribuir informações em diferentes locais de armazenamento, como discos ou servidores. Isso facilita a paralelização de consultas e operações de recuperação, tornando-se especialmente benéfico em sistemas de bancos de dados distribuídos, onde os dados estão distribuídos entre vários nós de um cluster.

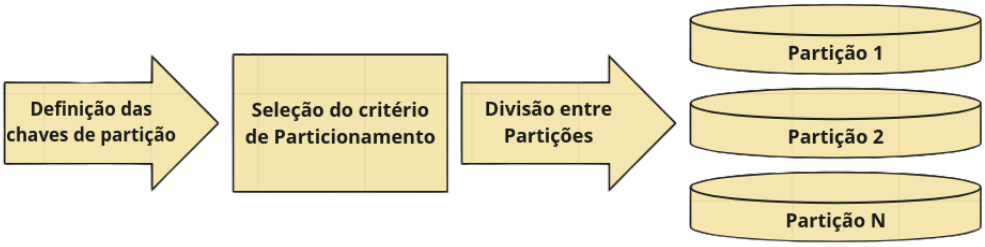
Dentre as estratégias amplamente adotadas de particionamento, destaca-se o Particionamento Horizontal, no qual os dados são divididos em partições com base em critérios de igualdade em uma coluna específica. Por exemplo, é possível particionar uma tabela de funcionários pelo departamento, aprimorando o desempenho de consultas que envolvem filtros nessas colunas particionadas. Outra estratégia relevante é o Particionamento Vertical, onde as colunas de uma tabela são divididas em partições separadas. Essa abordagem é particularmente útil quando diferentes partes de uma tabela são acessadas com frequência e de maneira independente. Cada uma dessas estratégias de particionamento oferece benefícios distintos, adaptando-se a diferentes cenários de uso e demandas de consulta.

Além dessas estratégias mencionadas, é importante destacar o Particionamento por Intervalo, uma técnica específica que organiza dados com base em faixas específicas de valores em uma coluna. Por exemplo, ao particionar uma tabela de vendas por intervalos de datas, é possível otimizar consultas que buscam informações em períodos específicos, aprimorando significativamente o desempenho.

A seleção da estratégia de particionamento adequada depende das características específicas do banco de dados, das consultas frequentes realizadas e dos requisitos de desempenho da aplicação em questão. Neste trabalho, a avaliação da técnica de particionamento de dados

incluirá uma análise específica do Particionamento por Intervalo, visando compreender seu impacto no desempenho e eficiência das consultas em um ambiente de banco de dados, seguindo as etapas da Figura 5. As etapas de análise comparativa e validação estatística contribuirão para demonstrar de forma sólida os benefícios e eventuais desafios associados ao uso dessa técnica específica de particionamento.

Figura 5 – Etapas na criação de um particionamento.



Fonte: elaborado pela autora.

2.3 TRABALHOS CORRELATOS

Nesta seção, serão apresentados alguns trabalhos relevantes que contribuem para o entendimento e avanço na área de pesquisa relacionada à eficiência energética em bancos de dados. Para a identificação desses trabalhos, foi realizada uma revisão sistemática da literatura, utilizando a seguinte estratégia de busca:

A pesquisa foi conduzida em diversas bases de dados, incluindo a Biblioteca Digital IEEE, Web of Science, Science Direct e Scopus, com acesso concedido através da Comunidade Acadêmica Federada (CAFe). A string de busca utilizada foi definida com base nos termos-chave e sinônimos relacionados à área de estudo, como mostrado abaixo:

String de Busca: *("Database"OR "Green Computing"OR "Green database") AND ("DBMS"OR "Query optimization"OR "Query-plan evaluation") AND ("Energy efficiency"OR "Energy saving"OR "performance metrics")*

Esta estratégia de busca foi elaborada com base nos termos PICOC (População, Intervenção, Comparação, Outcome e Contexto) e visou identificar estudos que avaliam a eficiência energética de diferentes tipos de sistemas de gerenciamento de banco de dados (SGBD), bem como as técnicas de otimização de consultas para economia de energia. Além disso, foi aplicado um filtro temporal para incluir estudos publicados nos últimos 10 anos, visto que a área é dinâmica e em evolução.

A pesquisa resultou em 103 artigos encontrados nas bases de dados selecionadas, dos quais 30 atenderam aos critérios de inclusão definidos. Os critérios de inclusão consideraram artigos que abordam a eficiência energética de bancos de dados, apresentam métricas de desempenho relacionadas à eficiência energética dos bancos de dados e descrevem e avaliam diferentes técnicas de otimização de consultas para bancos de dados.

Agora, apresentarei alguns dos trabalhos identificados durante essa revisão sistemática:

- Mahajan e Zong (2018) - *"Energy efficiency analysis of query optimizations on MongoDB and Cassandra"*

  Neste estudo, os autores focaram na análise da eficiência energética das otimizações de consultas nos bancos de dados MongoDB e Cassandra. Eles investigaram diferentes técnicas de otimização, como índices, cache e particionamento, e avaliaram o impacto dessas técnicas no consumo de energia e no desempenho das consultas. Os experimentos realizados mostraram que as otimizações de consultas tiveram um efeito positivo tanto na eficiência energética quanto no desempenho dos bancos de dados.
- Mahajan et al. (2019) - *"Improving the energy efficiency of relational and NoSQL databases via query optimizations"*

  Neste trabalho, os pesquisadores exploraram a melhoria da eficiência energética em bancos de dados relacionais e NoSQL por meio de otimizações de consultas. Eles utilizaram benchmarks amplamente aceitos, como o *Yahoo! Cloud Server Benchmark*, e conjuntos de dados personalizados, incluindo dados convertidos de aproximadamente 100GB do Twitter. O estudo avaliou uma variedade de técnicas de otimização e comparou o desempenho e a eficiência energética dos bancos de dados MySQL (relacional), MongoDB e Cassandra (NoSQL). Os resultados destacaram que as otimizações de consultas podem proporcionar economias significativas de energia, sem comprometer o desempenho, em diferentes tipos de bancos de dados.
- Gutowski et al. (2018) - *"Measuring the Energy Consumption of Massive Data Insertions: An Energy Consumption Assessment of the PL/SQL for LOOP and FORALL Methods"*

  Este trabalho concentrou-se na medição do consumo de energia durante a inserção em massa de dados em bancos de dados. Os autores avaliaram o consumo de energia dos métodos PL/SQL FOR LOOP e FORALL em um contexto de grande volume de dados.

O estudo destacou a importância de considerar o impacto energético durante operações de inserção em massa e revelou diferenças significativas no consumo de energia entre os métodos avaliados.

Os trabalhos mencionados fornecem uma base sólida para o desenvolvimento de estratégias e técnicas de otimização de consultas visando à eficiência energética em bancos de dados. Ao considerar essas contribuições, torna-se possível avançar no conhecimento e na conscientização sobre a importância de reduzir o consumo de energia e otimizar as operações nos sistemas de gerenciamento de banco de dados.

Em resumo, esses estudos destacam a relevância da eficiência energética em bancos de dados, evidenciando como as otimizações de consultas podem ser uma ferramenta valiosa para alcançar esse objetivo. Com as técnicas apropriadas, é viável obter economias significativas de energia sem prejudicar o desempenho, promovendo assim a sustentabilidade e a qualidade dos sistemas de banco de dados.

As limitações identificadas nos trabalhos correlatos foram um dos principais motores que impulsionaram a realização desta pesquisa. Um problema recorrente em muitas dessas pesquisas é a falta de padronização na medição do consumo de energia dos bancos de dados, o que torna difícil a comparação entre estudos e a obtenção de métricas consistentes de eficiência energética. Além disso, é notável a escassez de estudos que avaliam a eficiência energética em ambientes de produção do mundo real, uma vez que a maioria das pesquisas se baseia em conjuntos de dados simulados.

3 PROJETO GREENDB

O presente projeto tem como principal objetivo a otimização do consumo de energia em sistemas de gerenciamento de banco de dados que lidam com volumes substanciais de informações. Para atingir esse objetivo, foram aplicadas técnicas de otimização específicas, acompanhadas por uma análise do consumo de energia utilizando as ferramentas Powerjoular e Joularjx. A integração programática com o SGBD possibilitou a implementação das estratégias de otimização e a condução de experimentos automatizados. O resultado buscado era uma notável redução no consumo de energia, mantendo o desempenho e a qualidade das consultas realizadas.

Neste cenário, a pesquisa se destaca ao adotar uma abordagem inovadora. Em vez de concentrar-se diretamente nos sistemas de gerenciamento de banco de dados (SGBD), a pesquisa propõe a integração de estratégias de otimização de consultas com um programa intermediário entre a aplicação e o SGBD. Isso permite medir o consumo de energia no nível da chamada do método no programa, proporcionando um maior controle e uma granularidade que não é possível ao medir diretamente o SGBD. Essa abordagem é fundamental, uma vez que a maior parte da comunicação com o SGBD ocorre por meio de uma aplicação.

Adicionalmente, a pesquisa busca tornar essa integração acessível e econômica, eliminando a necessidade de dispositivos de *hardware* complexos para medição do consumo de energia. Essa abordagem não apenas amplia a viabilidade da pesquisa, mas também a torna aplicável em uma variedade de cenários de desenvolvimento de *software*.

Essa abordagem, que não requer dispositivos de *hardware* adicionais para medição, torna o projeto mais acessível e econômico, ampliando sua aplicabilidade em diversos contextos. Por meio dessa integração e medição, a pesquisa almeja avaliar a eficiência energética em bases de dados reais.

Além disso, essa pesquisa abre portas para novas propostas de pesquisa na área, especialmente relacionadas à otimização de consultas para melhorar a eficiência energética nesse contexto específico. Sendo assim, esse estudo representa uma iniciativa para contribuir para o avanço do conhecimento e da aplicação da eficiência energética em sistemas de gerenciamento

de banco de dados, oferecendo uma abordagem inovadora e acessível que pode ser aplicada em ambientes de produção do mundo real.

3.1 ESTRATÉGIA DE OTIMIZAÇÃO DE CONSULTAS SQL

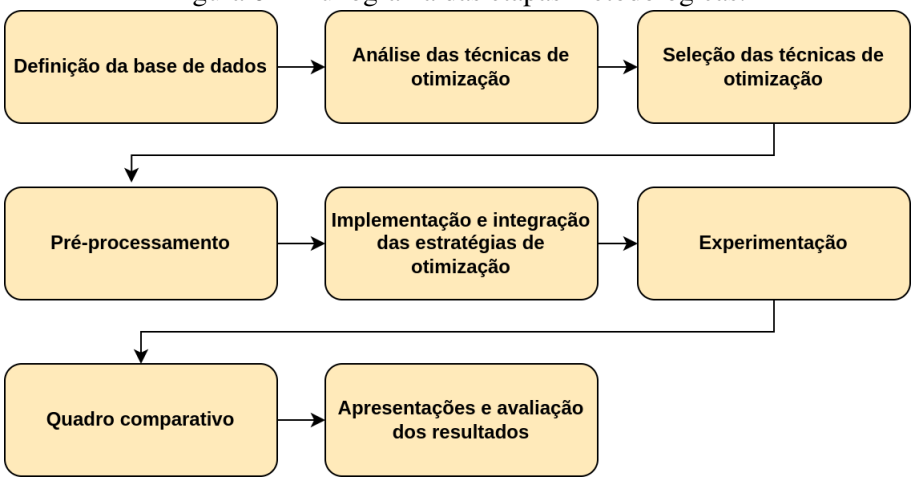
A abordagem escolhida para conduzir este estudo foi desdobrada em fases sequenciais, compreendendo a análise das técnicas de otimização, a seleção das estratégias, a implementação prática, a realização de experimentos e a validação dos resultados obtidos. A seguir, a metodologia utilizada é detalhada de forma abrangente:

- **Definição da base de dados:** Nesta etapa inicial, foi realizada a definição da base de dados que serviria como objeto de estudo e otimização. Foram considerados fatores como a estrutura, o tamanho e a representatividade da base de dados em relação ao ambiente de aplicação do projeto. A escolha criteriosa da base de dados foi fundamental para as etapas subsequentes de análise e otimização das consultas.
- **Análise das técnicas de otimização:** Nesta etapa, foram identificadas e analisadas as técnicas utilizadas no projeto, levando em consideração o contexto e as características do sistema em que as consultas foram otimizadas. A análise abrangeu critérios como desempenho, eficiência, segurança e escalabilidade.
- **Seleção das técnicas de otimização:** Com base na pesquisa teórica conduzida na revisão sistemática e nas características requeridas, foram selecionadas as técnicas de otimização de consultas mais adequadas ao contexto do projeto. As decisões consideraram fatores como a complexidade das consultas, a estrutura e o volume do banco de dados, e as restrições do ambiente de implementação. A preferência foi por estratégias consolidadas na literatura.
- **Implementação e integração das estratégias de otimização:** Nesta etapa, as técnicas de otimização de consultas selecionadas foram implementadas, e integradas no SGBD, utilizando linguagens de programação e ferramentas adequadas para o desenvolvimento e a integração das técnicas implementadas.
- **Experimentação:** Foram projetados e executados experimentos práticos para avaliar a eficiência energética e o desempenho das consultas otimizadas em comparação com as consultas não otimizadas. Foram definidos cenários de teste representativos do ambiente de aplicação do projeto, incluindo volumes de dados reais ou simulados. Além disso,

foram coletadas métricas relevantes, com uma ênfase na eficiência energética, seguido pelo tempo de resposta das consultas.

- **Quadro comparativo:** Os resultados obtidos nos experimentos foram analisados estatisticamente, realizando uma comparação entre os resultados das consultas otimizadas e não otimizadas, identificando possíveis ganhos de desempenho e eficiência.
- **Apresentações e avaliação dos resultados:** Com base nos resultados obtidos, foram elaborados relatórios técnicos detalhados, descrevendo as etapas da pesquisa, a seleção das técnicas, a implementação, os experimentos e os resultados obtidos.

Figura 6 – Fluxograma das etapas metodológicas.



Fonte: Elaborado pela autora.

Em última análise, a Figura 6 ilustra o fluxo das etapas metodológicas apresentadas.

3.1.1 Delimitações

Foram definidas algumas delimitações para orientar o escopo e os limites da pesquisa. Estas delimitações incluem:

- **Banco de Dados Utilizado:** O estudo foi realizado utilizando o SGBD MySQL, em sua versão 8.0.34. O banco de dados foi instalado e configurado em um servidor Linux Ubuntu 22.04. Portanto, não foram considerados outros SGBDs ou sistemas operacionais.
- **Técnicas de Otimização de Consultas SQL:** A escolha das técnicas de otimização de consultas SQL foi pautada na revisão da literatura existente e considerações de viabilidade temporal. Embora existam diversas técnicas disponíveis, a seleção foi cuidadosamente feita para atender aos objetivos deste estudo dentro do prazo estipulado.
- **Medição do Consumo de Energia e Tempo de Resposta:** O estudo incluiu a medição do

consumo de energia e do tempo de resposta das consultas SQL. Essa abordagem permitiu a análise da eficiência energética em relação ao desempenho. A relação entre o tempo de resposta e o consumo de energia foi investigada para identificar possíveis otimizações.

- **Implementação Prática Controlada:** A implementação prática das técnicas de otimização de consultas SQL foi realizada em um ambiente controlado, utilizando um conjunto de consultas e dados específicos. É importante observar que, em ambientes de produção reais, a complexidade das consultas e a carga de trabalho podem variar, o que pode afetar os resultados de eficiência energética e desempenho.

As delimitações foram definidas com o objetivo de estabelecer o escopo da pesquisa e fornecer uma base sólida para a análise de eficiência energética e otimização de consultas SQL. É importante reconhecer que essas delimitações ajudam a definir os limites do estudo e que outras abordagens ou contextos podem apresentar resultados diferentes.

3.2 COLETA DE DADOS E CRIAÇÃO DO CONJUNTO DE DADOS

Nesta seção, é apresentado o processo de aquisição das informações que serviram como base para as análises no âmbito do projeto GreenDB. A coleta de dados é uma etapa essencial, pois influencia diretamente a qualidade de todas as análises subsequentes. Logo, é detalhado as fontes de dados utilizadas, os métodos empregados para a obtenção das informações, bem como as estratégias adotadas para garantir a consistência e a relevância dos dados coletados. Além disso, exploramos a restrição temporal dos dados, justificando o período selecionado para análise. Após isso, a preparação e transformação dos dados, realizada com o auxílio da ferramenta PySpark, são discutidas em detalhes, incluindo exemplos de operações de limpeza e formatação. Esta seção oferece uma visão abrangente do processo de coleta de dados, destacando a importância de uma base sólida e confiável para análises posteriores.

3.2.1 Coleta de Dados

A coleta de dados hospitalares representa um pilar essencial para o nosso projeto. Optamos por obter dados do Sistema de Informações de Internações Hospitalares (SIH), uma fonte pública amplamente reconhecida por suas informações relacionadas ao atendimento hospitalar em todo o país. O SIH foi escolhido como nossa principal fonte de dados devido à sua natureza pública e à vasta quantidade de informações abrangentes disponíveis. Esta extensa base

de dados é fornece informações representativas que servirão de base para a análise da eficiência energética e otimização do consumo de energia em sistemas de gerenciamento de bancos de dados.

Além do SIH, outras fontes de dados externos foram integradas ao projeto. Isso inclui informações sobre estabelecimentos de saúde, características municipais, códigos internacionais de doenças (CID) e detalhes de procedimentos médicos. Essas fontes adicionais enriquecem a análise, permitindo um contexto mais abrangente.

Os dados foram restritos ao intervalo de tempo compreendido entre os anos de 2019 e 2022. Essa seleção baseou-se na necessidade de trabalhar com dados recentes e atuais, garantindo a relevância das análises realizadas.

3.2.2 Pré-processamento de Dados com PySpark

O PySpark foi escolhido como a ferramenta principal para lidar com os dados, uma decisão estratégica, dada a sua facilidade de uso com arquivos CSV, que é o formato comum das bases de dados, principalmente a do DataSUS. Usamos o PySpark para ler, limpar, transformar e, quando necessário, renomear colunas nos dados. A flexibilidade do PySpark garante que os dados estejam prontos para análises futuras, enquanto sua capacidade de integrar informações de diferentes fontes enriquece nossa base de dados e nos permite fazer análises abrangentes.

Aqui está um exemplo de código PySpark utilizado para a leitura de arquivos CSV:

```
1 df_mun = spark.read.csv('dados/municipios.csv', header=True)
```

Código-fonte 1 – Exemplo de Leitura de Arquivos CSV com PySpark

Em seguida, o código 2 é um exemplo de código PySpark para renomear colunas:

```
1 df_mun = spark.read.csv('dados/municipios.csv', header=True)
2 df_mun = df_mun.withColumnRenamed('NOME', 'NOME_MUN')
```

Código-fonte 2 – Exemplo de Limpeza e Renomeação de Colunas com PySpark

Após a preparação dos dados com o PySpark, o próximo passo foi o carregamento desses dados no banco de dados que serviu como fonte para consultas e análises. A eficiência desse processo é

fundamental para garantir que os dados estejam prontos para serem consultados de forma rápida e precisa.

No projeto GreenDB, o carregamento de dados no banco de dados foi realizado por meio do uso do PySpark e da funcionalidade de escrita em banco de dados. Cada conjunto de dados preparado foi inserido no banco de dados correspondente, garantindo a integridade e a coerência dos dados armazenados.

No código 3, é apresentado um trecho de código Python que demonstra a inserção dos dados no banco de dados:

```
1 def insert_data(csv, table_name):
2     df = spark.read.csv(f'dados/processado/{csv}.csv',
    inferSchema=True, header=True)
3     df.write.jdbc(DB_URL, table_name, properties=props, mode='
    overwrite')
```

Código-fonte 3 – Exemplo de inserção dos dados.

Nesse exemplo, a função *'insert_data'* é usada para carregar os dados de um arquivo CSV em uma tabela específica do banco de dados. O uso do PySpark tornou o processo de carregamento eficiente e automatizado.

3.3 PREPARAÇÃO DO BANCO DE DADOS E ESTRUTURA DAS TABELAS

Nesta seção, detalharemos as etapas de preparação do banco de dados, incluindo a estrutura das tabelas e as modificações necessárias para permitir a criação de chaves primárias e estrangeiras.

3.3.1 Preparação das Tabelas

Para criar um ambiente de banco de dados otimizado, realizamos modificações nas tabelas do banco de dados. O processo envolveu a definição de chaves primárias e estrangeiras para melhorar a integridade dos dados e otimizar a estrutura do banco de dados.

3.3.1.1 Criação de Chave Primária

Uma chave primária é fundamental para garantir a unicidade das entradas em uma tabela. No exemplo a seguir, demonstramos a alteração da tabela ‘municipios‘ para adicionar uma chave primária à coluna ‘MUNIC_RES‘. Essa alteração é necessária para garantir que cada município seja identificado de forma única.

```
1 ALTER TABLE `datasus`.`municipios`
2 CHANGE COLUMN `MUNIC_RES` `MUNIC_RES` INT NOT NULL ,
3 ADD PRIMARY KEY (`MUNIC_RES`);
```

Código-fonte 4 – Exemplo de criação de chave primária

3.3.1.2 Criação de Chave Estrangeira

As chaves estrangeiras são usadas para estabelecer relações entre tabelas. No contexto do projeto GreenDB, é essencial relacionar tabelas para permitir análises detalhadas. No exemplo a seguir, mostramos a criação de uma chave estrangeira na tabela ‘internamentos_ba‘, vinculando a coluna ‘MUNIC_RES‘ à tabela ‘municipios‘. Isso permite a referência a informações de municípios nas consultas sobre internamentos.

```
1 ALTER TABLE datasus.internamentos_ba
2 ADD CONSTRAINT fk_internamentos_municipio
3 FOREIGN KEY (MUNIC_RES) REFERENCES datasus.municipios(MUNIC_RES)
    ;
```

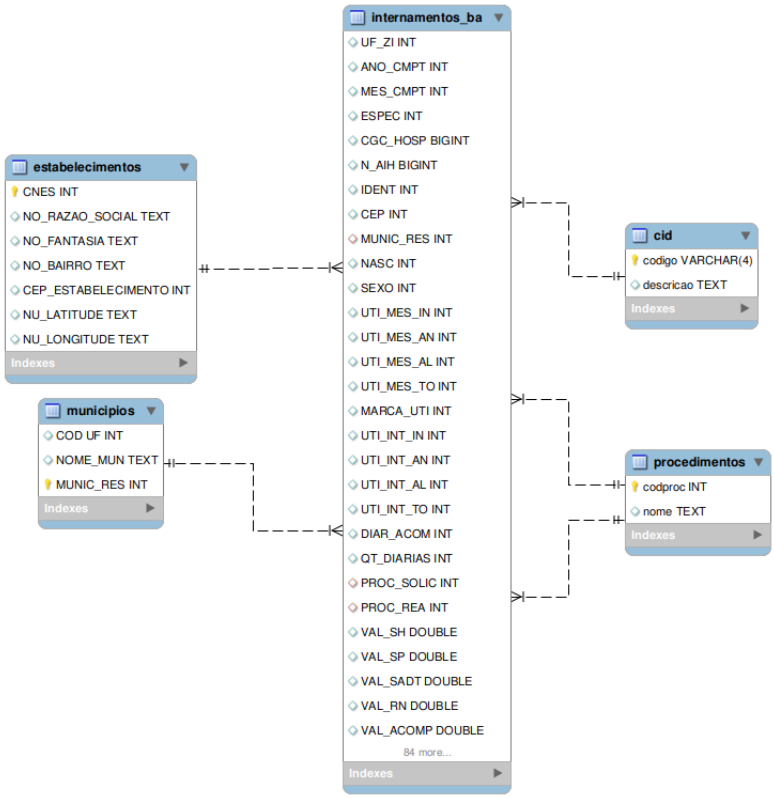
Código-fonte 5 – Exemplo de criação de chave estrangeira

3.3.2 Descrição das Tabelas

A seguir, apresentamos as tabelas do banco de dados e uma breve descrição do propósito de cada tabela:

A Figura 7 ilustra o relacionamento entre as tabelas Internamentos, Estabelecimentos de Saúde, Municípios, Procedimentos e Classificação Internacional de Doenças (CID), mostrando

Figura 7 – Relacionamento entre as tabelas Internamentos, Estabelecimentos de Saúde, Municípios, Procedimentos e CID



Fonte: Elaborado pela autora.

como essas tabelas se conectam para análises mais abrangentes.

A Tabela "CID" contém informações sobre a Classificação Internacional de Doenças, que é amplamente utilizada para codificar diagnósticos médicos, onde contém os campos de código, que é o código da CID e o campo descrição, sendo a descrição da condição médica

A Tabela "Procedimentos" engloba informações referentes aos procedimentos médicos realizados em pacientes. Esta tabela inclui campos que representam o código do procedimento médico, juntamente com o seu nome. Estes dados servem para a análise e compreensão dos procedimentos de saúde registrados.

A Tabela "Municípios" abrange informações relacionadas aos municípios, fornecendo um contexto geográfico para análises subsequentes. Ela inclui campos que representam o código da Unidade Federativa (UF) do município, o nome do município e um código específico para sua identificação.

A Tabela "Estabelecimentos de Saúde" contém informações detalhadas sobre os locais de atendimento médico. Ela engloba o Código Nacional de Estabelecimentos de Saúde

(CNES), que serve como identificador exclusivo, a Razão Social que representa o nome legal do estabelecimento, o Nome Fantasia, que é o nome pelo qual o local é conhecido publicamente, o Bairro onde está situado, o CEP (Código de Endereçamento Postal) correspondente, além das coordenadas geográficas de latitude e longitude. Esses dados são fundamentais para análises relacionadas à localização geográfica e para compreender a diversidade de estabelecimentos de saúde.

A Tabela de "Internamentos" abrange diversos campos essenciais para a análise da eficiência energética no contexto hospitalar. Esta tabela contém informações relevantes, incluindo o "CGC_HOSP" que identifica o hospital do internamento, bem como os valores associados ao internamento, como "Valores de SH" (Sistema Hospitalar) e "Valores de SP" (Serviço Profissional). Além disso, campos como "Morte", "UTI" (Unidade de Terapia Intensiva), "Dias de Permanência" e "Natureza" desempenham um papel importante e na gestão dos recursos hospitalares.

É importante ressaltar que, em conformidade com as diretrizes de privacidade e ética, esta tabela não contém informações que possam identificar os pacientes, mantendo a confidencialidade de dados sensíveis. Os campos presentes, como "Sexo" e "Idade", não fornecem dados de identificação pessoal, garantindo a privacidade dos pacientes e a integridade dos dados.

Essas modificações e estruturas de tabelas são essenciais para fornecer uma base sólida para a análise de eficiência energética no contexto do projeto GreenDB.

3.4 MENSURAÇÃO DO CONSUMO DE ENERGIA

A mensuração do consumo de energia desempenha um papel fundamental na otimização de sistemas e aplicativos, especialmente em um cenário cada vez mais voltado para a eficiência energética e sustentabilidade. Neste contexto, utilizamos dois *software* de monitoramento de energia: o PowerJoular e o JoularJX (NOUREDDINE, 2022).

A mensuração do consumo de energia desempenha um papel fundamental na otimização de sistemas e aplicativos, especialmente em um cenário cada vez mais voltado para a eficiência energética e sustentabilidade. Em um esforço para tornar nossos sistemas de gerenciamento de banco de dados mais eficientes e alinhados com as práticas de sustentabilidade,

optamos por utilizar duas ferramentas de monitoramento de energia: o PowerJoular e o JoularJX. Essas ferramentas foram escolhidas devido à eficácia em fornecer uma análise detalhada do consumo de energia, pela sua capacidade de identificar áreas de alto consumo e o fato de que não requerem dispositivos de *hardware* adicionais para realizar as medições.

3.4.1 Powerjoular

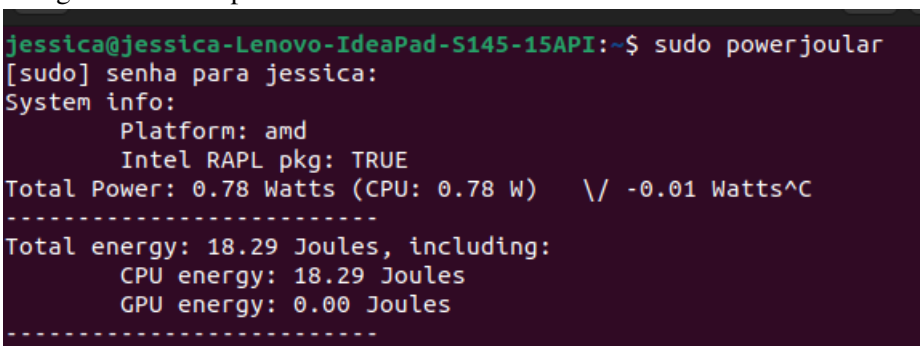
O PowerJoular é uma ferramenta de monitoramento de energia que fornece uma visão detalhada do consumo de energia de um sistema. Essa solução oferece a capacidade de obter dados precisos sobre o consumo de energia do processador, fornecendo métricas relevantes, como o total de energia consumida pelo sistema, a alocação de energia por componente e o gasto energético decorrente da execução de consultas específicas (NOUREDDINE, 2022).

Sua versatilidade é evidente ao ser aplicada em diversos cenários, desde o desenvolvimento de *software* até testes de desempenho e análise de consumo energético em computadores e dispositivos de placa única, como o Raspberry Pi (NOUREDDINE, 2022). O diferencial do PowerJoular é sua habilidade de oferecer análises em tempo real, permitindo que desenvolvedores identifiquem com precisão as áreas do código-fonte que estão demandando mais energia. Essa característica possibilita a otimização dessas seções específicas, resultando em uma redução global do consumo de energia.

No contexto do monitoramento de energia, o PowerJoular utiliza uma interface de linha de comando, como ilustrado na Figura 8. Essa visualização apresenta a saída padrão da interface, que foi projetada para oferecer flexibilidade e eficiência aos usuários que interagem por meio de um terminal. A interface disponibiliza em tempo real os dados de consumo de energia tanto da CPU quanto da GPU, além de oferecer a opção de registrar esses dados em arquivos no formato *Comma-Separated Values* (CSV). Além disso, os dados energéticos são coletados a cada segundo, permitindo análises de tendências ao longo do tempo e identificação de áreas que podem ser otimizadas.

Além disso, o PowerJoular tem a capacidade de monitorar o consumo energético de processos individuais por meio da inserção de seus Identificadores de Processo (PID) durante a execução. A Figura 9 ilustra essa funcionalidade, mostrando o monitoramento do consumo energético de um aplicativo Java usando a ferramenta JoularJX. A representação inclui o PID do aplicativo e os dados de consumo de energia em tempo real da CPU e GPU. Esse recurso é

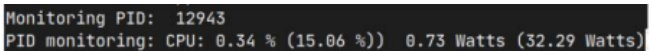
Figura 8 – Saída padrão da interface de linha de comando do PowerJoular



Fonte: elaborado pela autora.

importante para desenvolvedores em busca de aprimorar a eficiência energética de suas aplicações, permitindo a identificação precisa de processos de alto consumo e a otimização de seus códigos correspondentes.

Figura 9 – Saída padrão da interface de linha de comando do PowerJoular ao monitorar um PID



Fonte: (NOUREDDINE, 2022).

Sendo assim, a interface de linha de comando do PowerJoular e sua capacidade de monitorar processos individuais proporcionam uma importante fonte de informações, pois esses dados não apenas apoiam a análise e otimização do consumo de energia ao longo do tempo, mas também embasam decisões voltadas à eficiência energética e ao contínuo refinamento das aplicações desenvolvidas. Portanto, a presença do PowerJoular no campo de monitoramento de consumo energético emerge como uma alternativa sólida para a otimização eficaz de sistemas, alinhando-se à busca contínua por práticas sustentáveis na indústria da computação.

3.4.2 JoularJX

O JoularJX é uma ferramenta que fornece dados de consumo de energia em tempo de execução para métodos individuais em um programa, e para alcançar essa granularidade, o JoularJX colabora com o PowerJoular para monitorar o consumo de energia da CPU e GPU. Por meio desse processo, o JoularJX coleta e registra dados de energia para todos os métodos em um arquivo separado, bem como para métodos específicos em outro arquivo (NOUREDDINE, 2022).

Além disso, uma característica do JoularJX é sua capacidade de fornecer dados de consumo de energia para métodos em tempo real, a cada segundo durante a execução, esses

dados podem ser visualizados por meio de ferramentas de visualização. Essa funcionalidade permite o acompanhamento dinâmico do consumo de energia para métodos individuais enquanto o *software* é executado, proporcionando diferentes percepções. Esse recurso é particularmente vantajoso para a identificação de métodos que consomem quantidades excessivas de energia, permitindo a otimização precisa dessas porções de código. Além disso, sua natureza de código aberto também oferece a oportunidade de personalização para atender às necessidades específicas dos desenvolvedores.

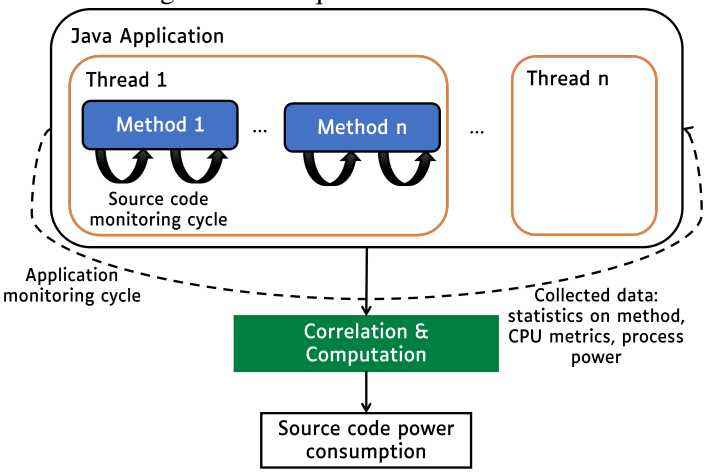
A Figura 10 ilustra a arquitetura central do JoularJX, destacando o processo de monitoramento, onde a cada ciclo de monitoramento do aplicativo *(Application monitoring cycle)*, que ocorre a cada um segundo por padrão, o JoularJX inicia sua coleta de dados *(Collected Data)*. Nesse estágio, ele foca na medição do uso da CPU pela *Java Virtual Machine* (JVM). Para isso, realiza o cálculo do consumo de energia de toda a JVM. Em sistemas Windows, essa medição é possível graças ao programa PowerMonitor.exe, utilizando a API Intel. Em sistema Linux x86_64, a interface *Running Average Power Limit* (RAPL) presente no kernel Linux é empregada para essa finalidade. Quando se trata de dispositivos como Raspberry Pi e Asus Tinker Board, modelos de potência de regressão desenvolvidos internamente entram em ação, fornecendo os dados necessários.

Em seguida, o JoularJX parte para a coleta de informações sobre o uso de CPU de cada *thread* ativa dentro da JVM, realizado por meio do método *getThreadCpuTime* fornecido pelo *Java Development Kit* (JDK). O objetivo é calcular o consumo de energia individual para cada *thread* em execução. No ciclo de monitoramento do código-fonte, que ocorre a cada dez milissegundos por padrão, o JoularJX direciona sua atenção para o *stacktrace* de cada *thread*, conforme exibido na Figura 10, onde contém informações vitais sobre os métodos em execução. Por fim, o JoularJX analisa cuidadosamente o *stacktrace* e identifica o método que está sendo executado no topo da pilha de chamadas.

Logo, ao término do ciclo de monitoramento do aplicativo, o JoularJX realiza uma análise estatística, implicando no cálculo da proporção de tempo gasto em cada método observado no *stacktrace*. Com base nessa análise, o JoularJX aloca o consumo de energia do código-fonte *(Source code power consumption)* conforme a contribuição de cada método. Esse processo detalhado possibilita uma compreensão precisa de como o consumo de energia é distribuído ao longo da execução do aplicativo, fornecendo informações valiosas para otimizações específicas e

aprimoramento da eficiência energética nos programas de *software* monitorados.

Figura 10 – Arquitetura do JoularJX



Fonte: (NOUREDDINE, 2022).

Na integração do JoularJX em nosso projeto, o uso do arquivo de configuração denominado **config.properties** permitiu a personalização eficiente do comportamento da ferramenta, oferecendo uma abordagem flexível para direcionar o monitoramento dos métodos de interesse. Como exemplo, configuramos a opção **filter-method-names=greendb**, direcionando o JoularJX para focar exclusivamente nos métodos cujo nome completo começa com "greendb". Essa configuração foi valiosa ao direcionar nossa análise para áreas específicas do código do projeto, facilitando a identificação de possíveis gargalos no consumo de energia.

Para iniciar o processo de monitoramento, utilizamos um script de lançamento. Esse script executou a classe Java do projeto em conjunto com o agente JoularJX. O uso desse agente possibilitou a coleta de dados de consumo de energia durante a execução do programa.

A linha de comando utilizada para executar o programa Java com o agente JoularJX foi a seguinte:

```
1 java -javaagent:$joularJX_jar -cp $CLASSPATH:$connector_jar $CLASS
```

Código-fonte 6 – Exemplo da Utilização do JoularJX.

Nesse contexto, as variáveis se desdobram da seguinte forma:

- `$joularJX_jar` corresponde ao caminho do arquivo JoularJX, que é o agente responsável pela coleta de dados de consumo de energia.

- `$CLASSPATH` representa o caminho para todas as classes necessárias ao projeto.
- `$connector_jar` indica o caminho para o arquivo do conector utilizado.
- `$CLASS` refere-se à classe Java a ser executada.

Ao concluir cada execução do projeto, o JoularJX gerou arquivos de dados detalhados, que forneceram informações sobre o consumo de energia de métodos específicos. Esses dados se mostraram essenciais para avaliar o impacto do consumo de energia no desempenho global do *software*.

Essa integração do JoularJX no projeto, com foco na configuração flexível, facilidade de implantação e análise de dados, revelou-se uma abordagem prática e eficaz para monitorar o consumo de energia e identificar áreas com potencial de melhoria no código-fonte. A personalização permitida pelo arquivo de configuração foi crucial para direcionar nossa análise com precisão, tornando-a altamente relevante para nossos objetivos de otimização energética.

Mais detalhes sobre a execução e a estrutura do script podem ser encontrados no Apêndice D: "Script de Lançamento".

3.5 INTEGRAÇÃO PROGRAMÁTICA COM O SGBD

A integração programática desempenha um papel essencial na implementação das técnicas de otimização de consultas em um Sistema de Gerenciamento de Banco de Dados (SGBD), no caso, o MySQL. Utilizamos a linguagem de programação Java juntamente com a biblioteca **mysql-connector-java** para estabelecer a conexão entre a aplicação Java e o MySQL.

Essa abordagem programática foi adotada, pois muitas consultas são executadas pelas aplicações que se conectam ao SGBD. Ela nos permite desenvolver funcionalidades adaptadas, ajustando as técnicas de otimização de consultas conforme as necessidades específicas do projeto e explorando os recursos avançados oferecidos pelo MySQL.

A biblioteca **mysql-connector-java** desempenha um papel central ao possibilitar uma conexão eficiente e segura com o MySQL. Por meio dessa integração, conseguimos realizar consultas customizadas, aproveitar recursos avançados do SGBD e implementar as técnicas de otimização previamente selecionadas.

Essa integração programática é utilizada para automatizar experimentos e execu-

tar consultas em larga escala, simplificando a criação de programas que executam consultas otimizadas, coletam dados relevantes e realizam análises estatísticas eficientes e consistentes.

4 TESTES E RESULTADOS

Neste capítulo, serão apresentados os testes realizados e os resultados obtidos no contexto do projeto "GreenDB." Descrevemos as técnicas de particionamento, indexação e a criação de views, bem como sua utilidade na promoção da eficiência energética em sistemas de gerenciamento de banco de dados.

Para avaliar a eficiência e o desempenho do banco de dados GreenDB, foi realizado testes utilizando a Tabela `internamentos_ba`. Durante os testes, foram consideradas duas configurações distintas da Tabela:

Tabela 1 – Configurações das Tabelas `internamentos_ba`

Tabela	**Número de Linhas**	**Tamanho**
Configuração Original	3.151.593	2 GB
Configuração Reduzida	1.575.797	1 GB

A configuração reduzida foi criada replicando a Tabela original com metade dos registros, mantendo as características essenciais para análise.

Esses testes visaram avaliar o desempenho do banco de dados em diferentes cenários, considerando o volume de dados. Os resultados destes testes são apresentados nas seções subsequentes.

4.0.1 Particionamento

O particionamento de dados por intervalo é uma técnica essencial para melhorar a eficiência energética e o desempenho do sistema. Ele envolve a divisão de Tabelas em subconjuntos menores com base em critérios como mês e ano, conforme ilustrado parcialmente no código 7 (o código completo pode ser encontrado no Apêndice A). A escolha desse particionamento baseia-se no fato de que, em sistemas de saúde, os dados são frequentemente acessados com base em datas. Portanto, ao particionar nossos dados dessa maneira, podemos significativamente melhorar o desempenho das consultas que buscam informações em um período específico.

```
CREATE TABLE datasus.internamentos_ba_partitioned
PARTITION BY RANGE COLUMNS (ANO_CMPT, MES_CMPT) (
        PARTITION p_2019_01 VALUES LESS THAN (2019, 2),
        PARTITION p_2019_02 VALUES LESS THAN (2019, 3),
        ...
        PARTITION p_max VALUES LESS THAN (MAXVALUE, MAXVALUE)
) AS
SELECT * FROM datasus.internamentos_ba WHERE 1 = 0;
```

Código-fonte 7 – Exemplo da Criação do particionamento.

4.0.1.1 Resultado do Consumo Energético e Tempo de Resposta

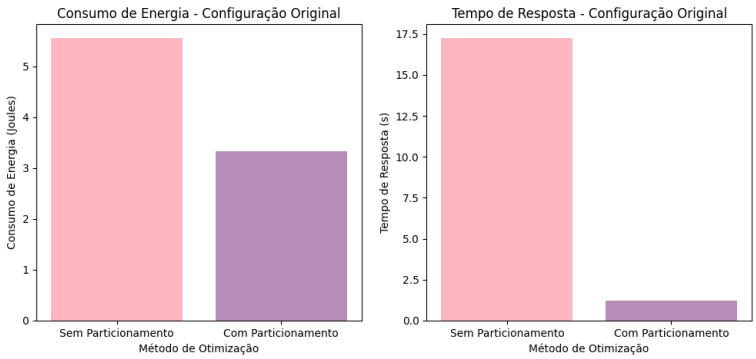
Tabela 2 – Consumo de Energia e Tempo de resposta na Criação do Particionamento

Configuração	Consumo de Energia (Joules)	Tempo de Resposta (s)
Configuração Original	15.20	68.68
Configuração Reduzida	12.10	2218.49

Tabela 3 – Resultado da consulta sem/com Particionamento na Configuração Original

Método de Otimização	Consumo de Energia (Joules)	Tempo de Resposta (s)
Sem Particionamento	5.55	17.24
Com Particionamento	3.33	1.23
Redução Percentual	40.00%	92.79%

Figura 11 – Consumo de Energia e Tempo de Resposta sem/com Particionamento na Configuração Original

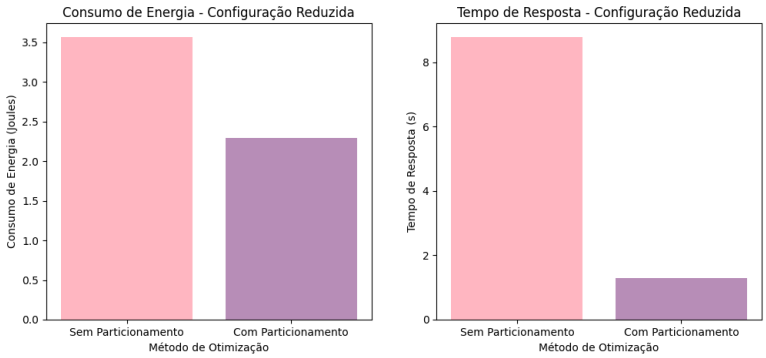


Fonte: Elaborado pela autora.

Tabela 4 – Resultado da consulta sem/com Particionamento na Configuração Reduzida

Método de Otimização	Consumo de Energia (Joules)	Tempo de Resposta (s)
Sem Particionamento	3.56	8.78
Com Particionamento	2.29	1.30
Redução Percentual	35.67%	85.19%

Figura 12 – Consumo de Energia e Tempo de Resposta sem/com Particionamento na Configuração Reduzida



Fonte: Elaborado pela autora.

A criação do particionamento na Tabela internamentos_ba envolveu a divisão da Tabela em subconjuntos menores com base nos critérios temporais de mês e ano. O processo de criação do particionamento consumiu energia e teve um impacto no tempo de resposta, conforme apresentado na Tabela 2.

Na configuração original, o consumo de energia durante a criação do particionamento foi de 15.20 Joules, e o tempo de resposta foi de 68.68 segundos. Já na configuração reduzida, esses valores foram de 12.10 Joules e 2218.49 segundos, respectivamente.

Comparando as configurações, observamos que a criação do particionamento demandou mais recursos na configuração reduzida devido à complexidade do processo, que envolve replicar a Tabela original com metade dos registros. Entretanto, é importante ressaltar que esse é um custo único associado à criação do particionamento e não afeta o desempenho contínuo das consultas uma vez que o particionamento está em vigor.

Apesar do aumento no consumo de energia durante a criação, os benefícios subsequentes do particionamento, evidenciados pelas reduções significativas no consumo de energia e tempo de resposta nas consultas, compensam amplamente esse custo inicial. Portanto, a estratégia de particionamento continua a ser uma escolha eficaz para otimizar o desempenho e a eficiência energética em sistemas de gerenciamento de banco de dados.

Os testes de particionamento revelaram resultados significativos em termos de eficiência energética e desempenho do sistema, como mostrado nas Figuras 11 e 12. Na configuração original, o particionamento proporcionou uma redução expressiva no consumo de energia, diminuindo de 5.55 Joules para 3.33 Joules, uma redução percentual de 40%. Além disso, o tempo de resposta para as consultas foi drasticamente reduzido, passando de 17.24 segundos para 1.23 segundos, representando uma melhoria de 92.79%, como visto na Tabela 3.

Na configuração reduzida da Tabela, os benefícios do particionamento também foram evidentes. O consumo de energia diminuiu de 3.56 Joules para 2.29 Joules, uma redução percentual de 35.67%. O tempo de resposta, por sua vez, experimentou uma diminuição notável, passando de 8.78 segundos para 1.30 segundos, uma melhoria de 85.19%, segundo a Tabela 4.

Esses resultados indicam que o particionamento de dados é uma estratégia eficaz, não apenas em configurações originais com grandes volumes de dados, mas também em cenários com uma quantidade reduzida de registros. A divisão de Tabelas com base em critérios temporais, como ano e mês, permitiu consultas mais eficientes, especialmente aquelas que buscam informações em períodos específicos. Essa abordagem mostrou-se valiosa tanto em termos de economia de energia quanto de aceleração das consultas.

4.0.2 Indexação

A indexação é uma técnica fundamental para aprimorar a eficiência do sistema. No contexto deste estudo, criamos índices por meio de árvores B, em colunas relevantes para acelerar consultas frequentes. Por exemplo, o código 17 mostra a criação do índice na coluna CGC (Cadastro Geral de Contribuintes) do hospital, denominado *idx_cgc_hosp* (o código completo pode ser visualizado no Apêndice C).

```
1 CREATE INDEX idx_cgc_hosp ON internamentos_ba (CGC_HOSP);
```

Código-fonte 8 – Criação do Índice idx_cgc_hosp

4.0.2.1 Resultado do Consumo Energético e Tempo de Resposta

A criação do Índice na coluna CGC_Hosp na configuração original apresentou um consumo de energia de 3.10 Joules e um tempo de resposta de 9.63 segundos durante a criação

do índice. Já na configuração reduzida, esses valores foram de 2.17 Joules e 4.78 segundos, respectivamente, conforme visto na Tabela 5.

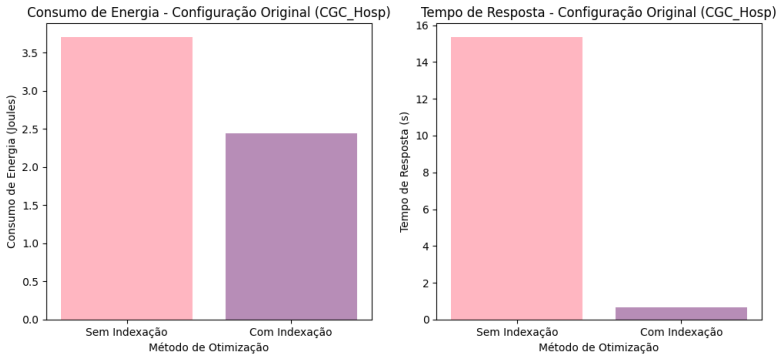
Tabela 5 – Consumo de Energia e tempo de resposta na Criação do Índice na coluna CGC_Hosp

Configuração	Consumo de Energia (Joules)	Tempo de Resposta (s)
Configuração Original	3.10	9.63
Configuração Reduzida	2.17	4.78

Tabela 6 – Resultado da consulta com/sem Indexação na coluna CGC_Hosp na Configuração Original

Método de Otimização	Consumo de Energia (Joules)	Tempo de Resposta (s)
Sem Indexação	3.70	15.34
Com Indexação	2.44	0.66
Redução Percentual	34.05%	95.69%

Figura 13 – Consumo de Energia e Tempo de Resposta com e sem Indexação na coluna CGC_Hosp na Configuração Original



Fonte: Elaborado pela autora.

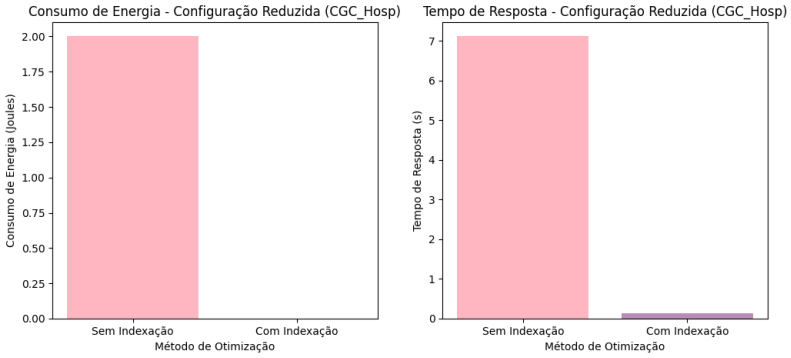
Tabela 7 – Resultado da consulta com/sem Indexação na coluna CGC_Hosp na Configuração Reduzida

Método de Otimização	Consumo de Energia (Joules)	Tempo de Resposta (s)
Sem Indexação	2.00	7.11
Com Indexação	0.0	0.14
Redução Percentual	100%	98.03%

A criação do índice na coluna CGC_Hosp teve um impacto expressivo nas consultas, como evidenciado nas Figuras 13 e 14. Nas duas configurações, a indexação resultou em uma redução percentual considerável tanto no consumo de energia quanto no tempo de resposta.

Na configuração original, a indexação resultou em uma redução de 34.05% no consumo de energia e uma melhoria de 95.69% no tempo de resposta. Na configuração reduzida,

Figura 14 – Consumo de Energia e Tempo de Resposta com e sem Indexação na coluna CGC_Hosp na Configuração Reduzida



Fonte: Elaborado pela autora.

os benefícios foram melhores, com uma redução de 100% no consumo de energia e uma melhoria de 98.03% no tempo de resposta. Esses resultados indicam que a indexação na coluna CGC_Hosp foi altamente eficaz, proporcionando ganhos expressivos de desempenho em ambas as configurações.

A criação do índice na coluna "sexo" resultou em um consumo de energia de 3.24 Joules e um tempo de resposta de 9.60 segundos na configuração original. Na configuração reduzida, esses valores foram de 1.99 Joules e 4.53 segundos, respectivamente.

Tabela 8 – Consumo de Energia e tempo de resposta na Criação do Índice na coluna Sexo

Configuração	Consumo de Energia (Joules)	Tempo de Resposta (s)
Configuração Original	3.24	9.60
Configuração Reduzida	1.99	4.53

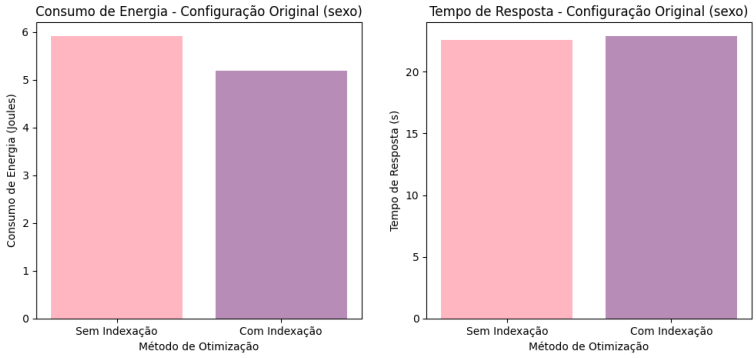
A criação do índice na coluna "sexo" proporcionou resultados distintos em comparação com a indexação na coluna CGC_Hosp. Os resultados detalhados são apresentados nas Tabelas 9 e 10.

Tabela 9 – Resultado da consulta com/sem Indexação na coluna sexo na Configuração Original.

Método de Otimização	Consumo de Energia (Joules)	Tempo de Resposta (s)
Sem Indexação	5.91	22.54
Com Indexação	5.19	22.86
Redução Percentual	12.17%	-1.42%

Na configuração original, a indexação na coluna "sexo" resultou em uma redução de 12.17% no consumo de energia, mas com uma ligeira diminuição de -1.42% no tempo de

Figura 15 – Consumo de Energia e Tempo de Resposta com e sem Indexação na coluna sexo na Configuração Original

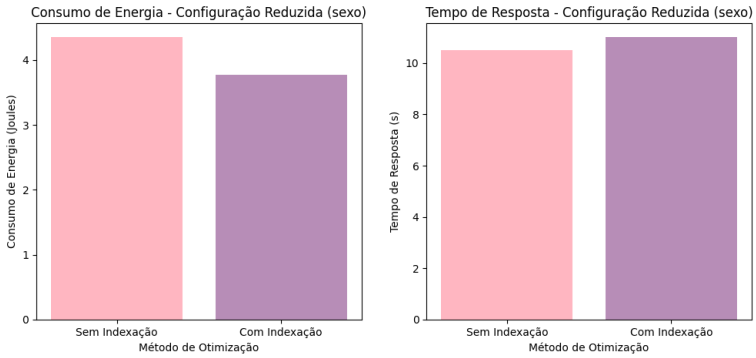


Fonte: Elaborado pela autora.

Tabela 10 – Resultado da consulta com/sem Indexação na coluna sexo na Configuração Reduzida

Método de Otimização	Consumo de Energia (Joules)	Tempo de Resposta (s)
Sem Indexação	4.35	10.51
Com Indexação	3.77	11.00
Redução Percentual	13.33%	-4.66%

Figura 16 – Consumo de Energia e Tempo de Resposta com e sem Indexação na coluna sexo na Configuração Reduzida



Fonte: Elaborado pela autora.

resposta em comparação com consultas sem indexação, como mostrado na Figura 15.

Já na configuração reduzida, a indexação apresentou uma redução mais significativa no consumo de energia, alcançando 13.33%. No entanto, o tempo de resposta teve uma variação negativa de -4.66%. Essa disparidade sugere que, em cenários com conjuntos de dados menores, o benefício da indexação na coluna "sexo" pode ser mais limitado e, em alguns casos, pode até resultar em um tempo de resposta ligeiramente maior em comparação com consultas sem indexação e pode ser vista na Tabela 10.

A baixa cardinalidade da coluna "sexo" (com apenas dois valores únicos: 1 e 3)

teve um impacto significativo na eficácia da indexação. Em colunas com alta cardinalidade, a implementação de índices pode ser altamente vantajosa, reduzindo consideravelmente o tempo necessário para recuperar dados específicos. No entanto, em colunas com baixa cardinalidade, como é o caso da coluna "sexo", os benefícios da indexação são mais limitados, uma vez que os valores únicos podem ser prontamente recuperados sem a necessidade de um índice. Vale ressaltar que a indexação não é uma operação isenta de custos, envolvendo tanto o armazenamento quanto a manutenção de índices, o que consome recursos significativos em termos de espaço em disco e capacidade de processamento.

Essa descoberta destaca a importância da análise detalhada do contexto e das características das colunas antes de optar pela indexação. A eficácia da indexação varia com base na natureza dos dados e no padrão de consultas do sistema. Ao otimizar sistemas de gerenciamento de banco de dados, é essencial considerar cuidadosamente a relação entre a cardinalidade da coluna, a técnica de indexação e o potencial impacto no desempenho das consultas. A escolha de quando e como aplicar a indexação deve ser guiada pela compreensão específica do cenário e das demandas das consultas envolvidas.

4.0.3 View

A criação de views é uma estratégia que visa simplificar consultas complexas. No escopo deste projeto, foi criada uma View chamada *soma_valores_por_mes_ano* para somar os valores por mês e ano, como demonstrado no trecho de código 9 (O código completo está disponível no Apêndice B):

```
1 CREATE VIEW soma_valores_por_mes_ano AS
2 SELECT
3     ANO_CMPT,
4     MES_CMPT,
5     SUM(VAL_TOT) AS total_valores
6 FROM
7     datasus.internamentos_ba
8 GROUP BY
9     ANO_CMPT, MES_CMPT;
```

Código-fonte 9 – Criação da Estratégia de View

4.0.3.1 Resultado do Consumo Energético e Tempo de Resposta

A criação da view resultou em dados notáveis, conforme a Tabela 11

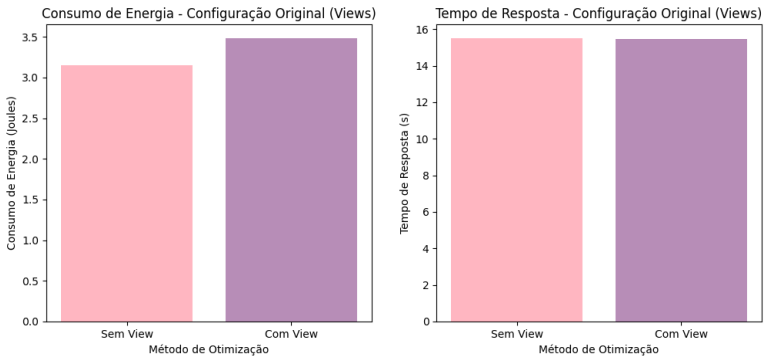
Tabela 11 – Consumo de Energia e tempo de resposta na Criação de Views

Configuração	Consumo de Energia (Joules)	Tempo de Resposta (s)
Configuração Original 1	4.03	0.15
Configuração Reduzida 2	3.55	0.10

Tabela 12 – Resultado da consulta sem/com Views na Configuração Original

Método de Otimização	Consumo de Energia (Joules)	Tempo de Resposta (s)
Sem View	3.15	15.49
Com View	3.48	15.46
Redução Percentual	-10.48%	0.19%

Figura 17 – Consumo de Energia e Tempo de Resposta sem/com Views na Configuração Original



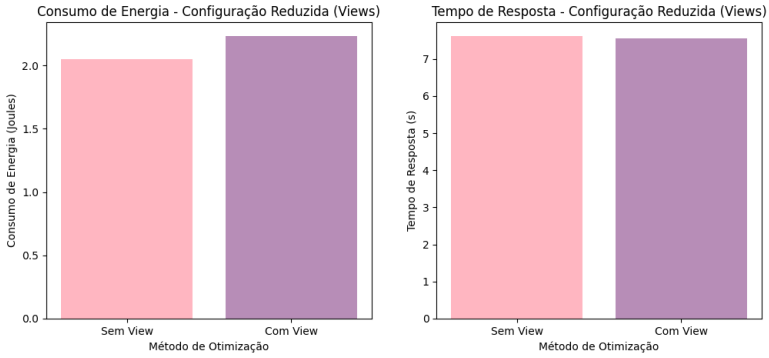
Fonte: Elaborado pela autora.

Tabela 13 – Resultado da consulta sem/com Views na Configuração Reduzida

Método de Otimização	Consumo de Energia (Joules)	Tempo de Resposta (s)
Sem View	2.05	7.61
Com View	2.23	7.56
Redução Percentual	-8.78%	0.65%

Na configuração original, a utilização da view revelou-se uma estratégia que teve um impacto negativo no consumo de energia, resultando em um aumento de 10.48%. O tempo de resposta praticamente não foi alterado, com uma variação de apenas 0.19%, conforme a Tabela 12. Já na configuração reduzida, a view contribuiu para um aumento de 8.78% no consumo de energia, enquanto o tempo de resposta teve uma leve melhora de 0.65%, de acordo com a Tabela 13.

Figura 18 – Consumo de Energia e Tempo de Resposta sem/com Views na Configuração Reduzida



Fonte: Elaborado pela autora.

5 CONSIDERAÇÕES FINAIS

No âmbito desse projeto, intitulado "GreenDB", exploramos estratégias e técnicas de otimização de consultas como uma abordagem para redução do consumo energético. Especificamente, concentramo-nos em técnicas de particionamento, indexação e criação de views em sistemas de gerenciamento de banco de dados, com o objetivo central de aprimorar a eficiência energética. Ao longo desse percurso, conduzimos uma série de testes e análises que nos permitiram aprofundar nosso entendimento acerca das notáveis reduções observadas não apenas no consumo de energia, mas também no tempo de resposta das consultas, em diversos cenários de otimização.

O particionamento de dados emergiu como uma estratégia altamente eficaz. A divisão temporal das tabelas não apenas acelerou consultas específicas, mas também resultou em uma redução notável no consumo de energia, destacando sua relevância para operações que exigem recuperação de informações em períodos específicos.

A indexação desempenhou um papel crucial, contudo, sua eficácia variou consideravelmente. A indexação na coluna de alta cardinalidade "CGC_Hosp"mostrou-se impactante, enquanto na coluna de baixa cardinalidade "sexo", os benefícios foram mais modestos. Esta discrepância ressalta a importância de considerar a natureza dos dados ao aplicar técnicas de indexação.

A criação de views não contribuiu positivamente para a eficiência energética. Pelo contrário, observamos um aumento no consumo de energia. Este resultado sublinha a necessidade de ponderar cuidadosamente a aplicação de views, reconhecendo que, em certos cenários, essa estratégia pode não ser vantajosa.

Ao avaliar o tempo de resposta das consultas, percebemos reduções consideráveis com o particionamento e a indexação. Contudo, a criação de views teve um impacto mínimo. Esses resultados reforçam a ideia de que, para além do consumo de energia, a análise do desempenho é fundamental na escolha e implementação de técnicas de otimização.

Este estudo proporcionou uma visão aprofundada das estratégias de otimização de

consultas SQL em SGBD, com foco na eficiência energética. As conclusões apontam para a necessidade de uma abordagem personalizada, considerando as características específicas do banco de dados e os padrões de consulta predominantes. As escolhas de técnicas de otimização devem ser guiadas pela compreensão do contexto do sistema, garantindo que o benefício energético seja equilibrado com o impacto no desempenho.

Embora este estudo tenha se concentrado no MySQL, há oportunidades substanciais para estender a pesquisa a outros SGBDs. Cada sistema possui peculiaridades únicas, e explorar técnicas de otimização em contextos diversos pode enriquecer nosso entendimento. Ferramentas de medição de energia específicas para SGBDs representam um passo importante em trabalhos futuros, proporcionando uma análise mais precisa e detalhada do consumo de energia.

Logo, este projeto oferece uma contribuição significativa para o campo em constante evolução da eficiência energética em sistemas de gerenciamento de banco de dados. Compreender e aplicar estratégias de otimização não apenas reduz o impacto ambiental, mas também impulsiona o desempenho, alinhando-se às crescentes demandas por práticas sustentáveis na gestão de dados.

REFERÊNCIAS

Agência Internacional de Energia. **Data Centers and Data Transmission Networks**. 2022. Disponível em: <https://www.iea.org/reports/data-centres-and-data-transmission-networks>.

ANEEL. **Programa de Eficiência Energética**. 2022. <https://www.gov.br/aneel/pt-br/acesso-a-informacao/acoes-e-programas/eficiencia-energetica>. Acessado em 18 de setembro de 2023.

APPLE. **Compromisso da Apple com a Sustentabilidade e a Computação Verde**. 2021. <https://nr.apple.com/dH8n2T1v6H>. Acessado em 18 de setembro de 2023.

BAYER, R.; MCCREIGHT, E. M. Organization and maintenance of large ordered indexes. **Acta Informatica**, Springer, v. 1, n. 3, p. 173–189, 1972.

BELOGLAZOV, A.; ABAWAJY, J.; BUYYA, R. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. **Future Generation Computer Systems**, Elsevier, v. 28, n. 5, p. 755–768, 2012.

BEULA, D.; SURESHKUMAR, M. A review on the toxic e-waste killing health and environment–today's global scenario. **Materials Today: Proceedings**, Elsevier, v. 47, p. 2168–2174, 2021.

BRASIL. **Lei nº 10.295, de 17 de outubro de 2001**. 2001. <http://www.procelinfo.com.br/>. Acessado em 18 de junho de 2023.

BRASIL, I. S. ambiental no. biodiversidade, economia e bem-estar humano/instituto de pesquisa econômica aplicada. **Brasília: Ipea**, 2010.

CHOI, W. G.; SHIN, M.; LEE, D.; PARK, H.; PARK, S. Optimization of a multiversion index on ssds to improve system performance. In: **2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)**. [S.l.: s.n.], 2016. p. 001620–001625.

ELMASRI, R.; NAVATHE, S. B. **Sistemas de Banco de Dados**. 6th. ed. São Paulo: Addison Wesley, 2011.

FERREIRA, C. de M. **RELATÓRIO DE ESTÁGIO SUPERVISIONADO DESENVOLVIMENTO DE UM MÓDULO PARA IMPORTAÇÃO DE DADOS**. 2018.

GOETZ, S.; ILSCHE, T.; CARDOSO, J.; SPILLNER, J.; KISSINGER, T.; ASSMANN, U.; LEHNER, W.; NAGEL, W. E.; SCHILL, A. Energy-efficient databases using sweet spot frequencies. In: **2014 IEEE/ACM 7TH INTERNATIONAL CONFERENCE ON UTILITY AND CLOUD COMPUTING (UCC)**. 345 E 47TH ST, NEW YORK, NY 10017 USA: IEEE, 2014. (International Conference on Utility and Cloud Computing). ISBN 978-1-4799-7881-6. ISSN 2373-6860.

GOOGLE. **Google 100% de Energia Renovável até 2030**. 2022. Acessado em 18 de setembro de 2023. Disponível em: <https://blog.google/intl/pt-br/novidades/iniciativas/google-for-brasil-renovando-nosso-compromisso-com-a-sustentabilidade-no-brasil/>.

GUTOWSKI, N.; CAMP, O.; CHAUVEAU, E. Measuring the energy consumption of massive data insertions: An energy consumption assessment of the pl/sql for loop and forall methods. In: . [S.l.: s.n.], 2018. v. 2018-January, p. 450–457. Cited By 0.

HÉMAN, S.; BONCZ, P.; KERSTEN, M. L. B+ tree compression. In: ACM. **Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data**. [S.l.], 2012. p. 113–124.

HINTEMANN, R.; HINTERHOLZER, S. Energy consumption of data centers worldwide. **Business, Computer Science (ICT4S)**, 2019.

KORTH, H.; SILBERSCHATZ, A. **Sistemas de Bancos de Dados**. 2a. edição revisada. ed. [S.l.]: Makron Books, 1994.

LITWIN, W.; NEIMAT, M.; SCHNEIDER, D. A. Linear hashing: A new tool for file and table addressing. In: **Proceedings of the 1980 ACM SIGMOD International Conference on Management of Data**. [S.l.: s.n.], 1980. p. 212–223.

LIU, X.; LI, W.; WANG, W.; WU, L.; CAO, J. Energy-efficient scheduling for big data processing in datacenters. **IEEE Transactions on Services Computing**, IEEE, v. 9, n. 4, p. 553–566, 2016.

LUO, B.; LI, B.; LIU, J.; CAO, J.; LUO, M. Energy efficiency optimization for large-scale data centers: A survey. **IEEE Transactions on Sustainable Computing**, IEEE, v. 1, n. 2, p. 83–98, 2015.

MACHINERY, A. f. C. O impacto da tecnologia da informação no consumo de energia e nas emissões de carbono. 2015. Disponível em: <https://dl.acm.org/doi/10.1145/2755977>.

MAHAJAN, D.; BLAKENEY, C.; ZONG, Z. Improving the energy efficiency of relational and nosql databases via query optimizations. **Sustainable Computing: Informatics and Systems**, v. 22, p. 120–133, 2019. ISSN 2210-5379. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2210537918301112>.

MAHAJAN, D.; ZONG, Z. Energy efficiency analysis of query optimizations on mongodb and cassandra. In: . [S.l.: s.n.], 2018. v. 2017-October, p. 1–6. Cited By 4.

MALMODIN, J.; LUNDéN, D. The energy and carbon footprint of the global ict and e&m sectors 2010–2015. **Sustainability (Basel, Switzerland)**, MDPI AG, Basel, v. 10, n. 9, p. 3027, 2018. ISSN 2071-1050.

MARCUS, R.; NEGI, P.; MAO, H.; TATBUL, N.; ALIZADEH, M.; KRASKA, T. Bao: Making learned query optimization practical. In: **Proceedings of the 2021 International Conference on Management of Data**. [S.l.: s.n.], 2021. p. 1275–1288.

MASSON-DELMOTTE, V.; ZHAI, P.; PIRANI, A.; CONNORS, S. L.; PÉAN, C.; BERGER, S.; CAUD, N.; CHEN, Y.; GOLDFARB, L.; GOMIS, M. et al. Climate change 2021: the physical science basis. **Contribution of working group I to the sixth assessment report of the intergovernmental panel on climate change**, Cambridge University Press Cambridge, UK, v. 2, 2021.

MICROSOFT. **Particionamento de Tabelas e Índices**. 2008. Acesso em 05 de outubro de 2023. Disponível em: <https://learn.microsoft.com/pt-br/previous-versions/sql/sql-server-2008-r2/ms178148(v=sql.105)?redirectedfrom=MSDN>.

MOI, P. C. P.; SOUZA, A. P. S. de; OLIVEIRA, M. M.; FAITTA, A. C. J.; REZENDE, W. B. de; MOI, G. P.; FREIRE, F. A. D. L. Lixo eletrônico: consequências e possíveis soluções. **Connection line-revista eletrônica do UNIVAG**, n. 7, 2014.

MURUGESAN, S. Harnessing green it: Principles and practices. **IT Professional**, v. 10, n. 1, p. 24–33, 2008.

NOUREDDINE, A. Powerjoular and joularjx: Multi-platform software power monitoring tools. In: **18th International Conference on Intelligent Environments (IE2022)**. Biarritz, France: [s.n.], 2022.

POGODAEV, A.; RYZHKOVA, D. Developing method to optimize queries in denormalized databases. In: **2020 2nd International Conference on Control Systems, Mathematical Modeling, Automation and Energy Efficiency (SUMMA)**. [S.l.: s.n.], 2020. p. 687–691.

PROCEL. 1985. <http://www.procelinfo.com.br/>. Acessado em 18 de junho de 2023.

REI, F. C. F.; GONÇALVES, A. F.; SOUZA, L. P. de. Acordo de paris: reflexões e desafios para o regime internacional de mudanças climáticas. **Veredas do Direito**, v. 14, n. 29, p. 81–99, 2017.

ROUKH, A.; BELLATRECHE, L.; ORDONEZ, C. Enerquery: Energy-aware query processing. In: **CIKM'16: PROCEEDINGS OF THE 2016 ACM CONFERENCE ON INFORMATION AND KNOWLEDGE MANAGEMENT**. 1515 BROADWAY, NEW YORK, NY 10036-9998 USA: ASSOC COMPUTING MACHINERY, 2016. ISBN 978-1-4503-4073-1.

SALAMI, B.; MALAZGIRT, G. A.; ARCAS-ABELLA, O.; YURDAKUL, A.; SONMEZ, N. Axledb: A novel programmable query processing platform on fpga. **MICROPROCESSORS AND MICROSYSTEMS**, ELSEVIER SCIENCE BV, PO BOX 211, 1000 AE AMSTERDAM, NETHERLANDS, v. 51, p. 142–164, JUN 2017. ISSN 0141-9331.

SHAR, L. K.; TAN, H. B. K. Defeating sql injection. **Computer**, IEEE, v. 46, n. 3, p. 69–77, 2012.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. **Sistema de Banco de Dados**. 3. ed. São Paulo: Makron Books, 1999.

SILBERSCHATZ, A.; SUNDARSHAN, S.; KORTH, H. F. **Sistema de banco de dados**. [S.l.]: Elsevier Brasil, 2016.

SMITH, M.; KOULAMAS, C. Green ict: a review, current trends and challenges. **Journal of Cleaner Production**, Elsevier, v. 142, p. 179–190, 2017.

SUN, Y.; WANG, Y.; YANG, H. Bidirectional database storage and sql query exploiting rram-based process-in-memory structure. **ACM TRANSACTIONS ON STORAGE**, ASSOC COMPUTING MACHINERY, 2 PENN PLAZA, STE 701, NEW YORK, NY 10121-0701 USA, v. 14, n. 1, SI, APR 2018. ISSN 1553-3077.

WU, K.; ZHU, Y.; LEE, D. L. Efficient approximate query processing in multimedia databases. **ACM Transactions on Database Systems (TODS)**, ACM, v. 26, n. 4, p. 388–423, 2001.

YAN, Z.; WEN, Y.; ZHANG, H.; JIANG, Z. A review of energy-efficient data management and processing in data centers. **Frontiers of Computer Science**, Springer, v. 12, n. 4, p. 713–730, 2018.

ZOBEL, J.; MOFFAT, A. Inverted files for text search engines. In: **ACM Computing Surveys (CSUR)**. [S.l.]: ACM, 2006. v. 38, n. 2, p. 6.

APÊNDICES

APÊNDICE A – Utilização do Particionamento

A seguir, é apresentado os códigos e descrições referentes à criação e utilização de particionamento no banco de dados do projeto.

No código fonte 10, é apresentado o SQL que foi utilizado para criar a tabela particionada, **internamentos_ba_partitioned**. Essa tabela é organizada de maneira eficiente, com base nos valores das colunas **ANO_CMPT** (ano de competência) e **MES_CMPT** (mês de competência). O particionamento é estabelecido utilizando a cláusula **PARTITION BY RANGE COLUMNS**, o que significa que os dados são distribuídos em partições com base no intervalo de valores dessas colunas. Várias partições são criadas, uma para cada mês e ano, com a última partição, **p_max**, designada para valores além do limite especificado.

```
1  CREATE TABLE datasus.internamentos_ba_partitioned
2      PARTITION BY RANGE COLUMNS (ANO_CMPT, MES_CMPT) (
3      PARTITION p_2019_01 VALUES LESS THAN (2019, 2),
4      PARTITION p_2019_02 VALUES LESS THAN (2019, 3),
5      PARTITION p_2019_03 VALUES LESS THAN (2019, 4),
6      PARTITION p_2019_04 VALUES LESS THAN (2019, 5),
7      PARTITION p_2019_05 VALUES LESS THAN (2019, 6),
8      PARTITION p_2019_06 VALUES LESS THAN (2019, 7),
9      PARTITION p_2019_07 VALUES LESS THAN (2019, 8),
10     PARTITION p_2019_08 VALUES LESS THAN (2019, 9),
11     PARTITION p_2019_09 VALUES LESS THAN (2019, 10),
12     PARTITION p_2019_10 VALUES LESS THAN (2019, 11),
13     PARTITION p_2019_11 VALUES LESS THAN (2019, 12),
14     PARTITION p_2019_12 VALUES LESS THAN (2020, 1),
15     PARTITION p_2020_01 VALUES LESS THAN (2020, 2),
16     PARTITION p_2020_02 VALUES LESS THAN (2020, 3),
17     PARTITION p_2020_03 VALUES LESS THAN (2020, 4),
18     PARTITION p_2020_04 VALUES LESS THAN (2020, 5),
19     PARTITION p_2020_05 VALUES LESS THAN (2020, 6),
20     PARTITION p_2020_06 VALUES LESS THAN (2020, 7),
```

```
21      PARTITION p_2020_07 VALUES LESS THAN (2020, 8),
22      PARTITION p_2020_08 VALUES LESS THAN (2020, 9),
23      PARTITION p_2020_09 VALUES LESS THAN (2020, 10),
24      PARTITION p_2020_10 VALUES LESS THAN (2020, 11),
25      PARTITION p_2020_11 VALUES LESS THAN (2020, 12),
26      PARTITION p_2020_12 VALUES LESS THAN (2021, 1),
27      PARTITION p_2021_01 VALUES LESS THAN (2021, 2),
28      PARTITION p_2021_02 VALUES LESS THAN (2021, 3),
29      PARTITION p_2021_03 VALUES LESS THAN (2021, 4),
30      PARTITION p_2021_04 VALUES LESS THAN (2021, 5),
31      PARTITION p_2021_05 VALUES LESS THAN (2021, 6),
32      PARTITION p_2021_06 VALUES LESS THAN (2021, 7),
33      PARTITION p_2021_07 VALUES LESS THAN (2021, 8),
34      PARTITION p_2021_08 VALUES LESS THAN (2021, 9),
35      PARTITION p_2021_09 VALUES LESS THAN (2021, 10),
36      PARTITION p_2021_10 VALUES LESS THAN (2021, 11),
37      PARTITION p_2021_11 VALUES LESS THAN (2021, 12),
38      PARTITION p_2021_12 VALUES LESS THAN (2022, 1),
39      PARTITION p_2022_01 VALUES LESS THAN (2022, 2),
40      PARTITION p_2022_02 VALUES LESS THAN (2022, 3),
41      PARTITION p_2022_03 VALUES LESS THAN (2022, 4),
42      PARTITION p_2022_04 VALUES LESS THAN (2022, 5),
43      PARTITION p_2022_05 VALUES LESS THAN (2022, 6),
44      PARTITION p_2022_06 VALUES LESS THAN (2022, 7),
45      PARTITION p_2022_07 VALUES LESS THAN (2022, 8),
46      PARTITION p_2022_08 VALUES LESS THAN (2022, 9),
47      PARTITION p_2022_09 VALUES LESS THAN (2022, 10),
48      PARTITION p_2022_10 VALUES LESS THAN (2022, 11),
49      PARTITION p_2022_11 VALUES LESS THAN (2022, 12),
50      PARTITION p_max VALUES LESS THAN (MAXVALUE, MAXVALUE)
51  ) AS
52  SELECT * FROM datasus.internamentos_ba WHERE 1 = 0;
```

Código-fonte 10 – Criação do particionamento.

Em seguida, em código fonte 11 é demostrado o processo de transferência de dados da tabela original, **internamentos_ba**, para a nova tabela particionada, **internamentos_ba_partitioned**. Isso é realizado com uma simples instrução de inserção, que copia todos os registros da tabela de origem para a tabela particionada. Esse processo ocorre para podermos começar a explorar a técnica de particionamento nas consultas subsequentes.

```
1 INSERT INTO datasus.internamentos_ba_partitioned
2 SELECT * FROM datasus.internamentos_ba;
```

Código-fonte 11 – Inserindo os dados na tabela particioanda.

Neste trecho de código 13 utilizando a linguagem Java, temos o método **InternamentsByYearMonth**. Esse método é responsável por realizar uma consulta SQL à tabela **datasus.internamentos_ba**. O objetivo dessa consulta é extrair informações específicas referentes ao ano de 2020 e ao mês de fevereiro. Os dados são agregados de acordo com critérios como procedimento, município de residência, diagnóstico principal, estabelecimento de saúde, sexo e faixa etária dos pacientes. Além disso, calculamos o total de internamentos e a média de dias de permanência para cada grupo.

```
1  public static void InternamentsByYearMonth(){
2      final String sqlQuery = "\n"
3              + "SELECT\n"
4              + "    B.nome AS procedimento,\n"
5              + "    C.nome_mun AS municipio_residencia,\n"
6              + "    D.descricao AS diag_principal,\n"
7              + "    E.NO_FANTASIA AS estabelecimento,\n"
8              + "    SEXO,\n"
9              + "    CASE\n"
10             + "        WHEN IDADE BETWEEN 0 AND 9 THEN '0-9'\n"
11             + "        WHEN IDADE BETWEEN 10 AND 19 THEN '10-19'\n"
12             + "        WHEN IDADE BETWEEN 20 AND 29 THEN '20-29'\n"
13             + "        WHEN IDADE BETWEEN 30 AND 39 THEN '30-39'\n"
14             + "        WHEN IDADE BETWEEN 40 AND 49 THEN '40-49'\n"
15             + "        WHEN IDADE BETWEEN 50 AND 59 THEN '50-59'\n"
16             + "        WHEN IDADE BETWEEN 60 AND 69 THEN '60-69'\n"
17             + "        ELSE '70+'\n"
18             + "    END AS faixa_etaria,\n"
```

```
19              + "     COUNT(*) AS total_internamentos ,\n"
20              + "     AVG(DIAS_PERM) AS media_dias_permanencia\n"
21              + "FROM datasus.internamentos_ba AS A\n"
22              + "INNER JOIN procedimentos AS B ON\n"
23              + "     A.PROC_REA = B.codproc\n"
24              + "INNER JOIN municipios AS C ON\n"
25              + "     A.MUNIC_RES = C.MUNIC_RES\n"
26              + "INNER JOIN cid AS D ON\n"
27              + "     A.DIAG_PRINC = D.CODIGO\n"
28              + "INNER JOIN estabelecimentos AS E ON\n"
29              + "     A.CNES = E.CNES\n"
30              + "WHERE ANO_CMPT = 2020\n"
31              + "     AND MES_CMPT = 2\n"
32              + "GROUP BY B.nome, C.nome_mun, D.descricao , E.NO_FANTASIA,
    SEXO, faixa_etaria\n"
33              + "ORDER BY SEXO, faixa_etaria , B.nome;";
34
35      conn.getConnection();
36      final ResultSet queryResultSet = conn.executeQuery(sqlQuery);
37      conn.closeConnection();
38  }
```

Código-fonte 12 – Método de consulta sem a utilização da técnica de particionamento.

O método **InternamentsByYearMonthPartition** do código 12 é responsável por executar consultas na tabela particionada **datasus.internamentos_ba_partitioned**. Assim como no código anterior, essa consulta busca informações referentes ao ano de 2020 e ao mês de fevereiro, mas se utiliza da estratégia de particionamento.

```
1  public static void InternamentsByYearMonthPartition(){
2      final String sqlQuery = "SELECT\n"
3              + "     B.nome AS procedimento ,\n"
4              + "     C.nome_mun AS municipio_residencia ,\n"
5              + "     D.descricao AS diag_principal ,\n"
6              + "     E.NO_FANTASIA AS estabelecimento ,\n"
7              + "     SEXO,\n"
8              + "     CASE\n"
9              + "         WHEN IDADE BETWEEN 0 AND 9 THEN '0-9'\n"
```

```
10              + "         WHEN IDADE BETWEEN 10 AND 19 THEN '10-19'\n"
11              + "         WHEN IDADE BETWEEN 20 AND 29 THEN '20-29'\n"
12              + "         WHEN IDADE BETWEEN 30 AND 39 THEN '30-39'\n"
13              + "         WHEN IDADE BETWEEN 40 AND 49 THEN '40-49'\n"
14              + "         WHEN IDADE BETWEEN 50 AND 59 THEN '50-59'\n"
15              + "         WHEN IDADE BETWEEN 60 AND 69 THEN '60-69'\n"
16              + "         ELSE '70+'\n"
17              + "     END AS faixa_etaria ,\n"
18              + "     COUNT(*) AS total_internamentos ,\n"
19              + "     AVG(DIAS_PERM) AS media_dias_permanencia\n"
20              + "FROM datasus.internamentos_ba_partitioned AS A\n"
21              + "INNER JOIN procedimentos AS B ON\n"
22              + "     A.PROC_REA = B.codproc\n"
23              + "INNER JOIN municipios AS C ON\n"
24              + "     A.MUNIC_RES = C.MUNIC_RES\n"
25              + "INNER JOIN cid AS D ON\n"
26              + "     A.DIAG_PRINC = D.CODIGO\n"
27              + "INNER JOIN estabelecimentos AS E ON\n"
28              + "     A.CNES = E.CNES\n"
29              + "WHERE ANO_CMPT = 2020\n"
30              + "     AND MES_CMPT = 2\n"
31              + "GROUP BY B.nome, C.nome_mun, D.descricao , E.NO_FANTASIA,
32               SEXO, faixa_etaria\n"
33              + "ORDER BY SEXO, faixa_etaria , B.nome;";
34
35      conn.getConnection();
36      final ResultSet queryResultSet = conn.executeQuery(sqlQuery);
37      conn.closeConnection();
38  }
```

Código-fonte 13 – Método de consulta com a utilização da técnica de particionamento.

APÊNDICE B – Utilização da View

Este apêndice apresenta os códigos e descrições relacionados à criação e utilização da estratégia de View.

No código fonte 14, é demonstrada a criação da view denominada **soma_valores_por_mes_ano**. Essa view é projetada para extrair informações agregadas da tabela **datasus.internamentos_ba**. A view realiza a soma dos valores totais (**VAL_TOT**) com base no ano de competência (**ANO_CMPT**) e mês de competência (**MES_CMPT**). Isso simplifica a obtenção de informações sobre o total de valores gastos em internamentos ao longo do tempo.

```
1 CREATE VIEW soma_valores_por_mes_ano AS
2 SELECT
3     ANO_CMPT,
4     MES_CMPT,
5     SUM(VAL_TOT) AS total_valores
6 FROM
7     datasus.internamentos_ba
8 GROUP BY
9     ANO_CMPT, MES_CMPT;
```

Código-fonte 14 – Criação da Estratégia de View

No código fonte 15, apresentamos um método em linguagem de programação Java denominado **SumValuesinternaments**. Esse método é responsável por realizar uma consulta direta na tabela **datasus.internamentos_ba** para calcular a soma dos valores totais de internamentos por mês e ano. Essa abordagem não utiliza a técnica de View.

```
1     public static int SumValuesinternaments() throws SQLException,
      IOException {
2         final String sqlQuery =
3             "SELECT\n" +
4             "    ANO_CMPT,\n" +
5             "    MES_CMPT,\n" +
```

```
6             "    SUM(VAL_TOT) AS total_valores\n" +
7             "FROM\n" +
8             "    datasus.internamentos_ba\n" +
9             "GROUP BY\n" +
10            "    ANO_CMPT, MES_CMPT;";
11
12        conn.getConnection();
13        final ResultSet queryResultSet = conn.executeQuery(sqlQuery);
14        conn.closeConnection();
15        return 0;
16    }
```

Código-fonte 15 – Método de consulta com a utilização da estratégia de view.

Já no código fonte 16, temos o método **SumValuesinternamentsView**. Este método utiliza a View **soma_valores_por_mes_ano** para calcular a soma dos valores totais de internamentos por mês e ano.

```
1     public static void SumValuesinternamentsView() throws SQLException,
      IOException {
2         final String sqlQuery =
3             "SELECT\n" +
4             "    ANO_CMPT,\n" +
5             "    MES_CMPT,\n" +
6             "    total_valores\n" +
7             "FROM\n" +
8             "    datasus.soma_valores_por_mes_ano;" +
9         conn.getConnection();
10        final ResultSet queryResultSet = conn.executeQuery(sqlQuery);
11        conn.closeConnection();
12    }
```

Código-fonte 16 – Método de consulta sem a utilização da estratégia de view

APÊNDICE C – Utilização de Indices

O código SQL no Código 17 é responsável pela criação de um índice denominado **idx_cgc_hosp** na tabela **internamentos_ba**.

```sql
CREATE INDEX idx_cgc_hosp ON internamentos_ba (CGC_HOSP);
```

Código-fonte 17 – Criação do índice idx_cgc_hosp

O trecho de código Java no **Código 20** implementa o método `InternamentsByHosp`. Esse método tem como objetivo realizar uma consulta SQL na tabela de internamentos do Sistema Único de Saúde (SUS). A consulta visa extrair informações estatísticas específicas relacionadas aos internamentos em um hospital identificado pelo número de CGC '13937131005887'

```java
public static void InternamentsByHosp() throws SQLException,
IOException {
    final String sqlQuery = "SELECT\n"
            + "    A.SEXO,\n"
            + "    B.nome AS procedimento,\n"
            + "    COUNT(*) AS total_internamentos\n"
            + "FROM datasus.internamentos_ba AS A\n"
            + "INNER JOIN procedimentos AS B ON A.PROC_REA = B.
codproc\n"
            + "WHERE A.CGC_HOSP = '13937131005887' \n"
            + "GROUP BY A.SEXO, procedimento\n"
            + "ORDER BY total_internamentos DESC;";

    conn.getConnection();
    final ResultSet queryResultSet = conn.executeQuery(sqlQuery);
    conn.closeConnection();
 }
```

Código-fonte 18 – Método de consulta de internamentos por gênero.

O código SQL 19 demonstra a criação de um índice denominado **idx_sexo** na tabela

internamentos_ba. Este índice é criado especificamente na coluna **SEXO** da tabela, com o objetivo de otimizar o desempenho das consultas que envolvem essa coluna.

```
1 CREATE INDEX idx_sexo ON internamentos_ba (SEXO);
```

Código-fonte 19 – Criação do índice idx_sexo

O trecho de código Java apresentado no Código 20 implementa o método **InternamentsBySexo**. Este método é responsável por executar uma consulta SQL na tabela de dados de internamentos do Sistema Único de Saúde (SUS) com o objetivo de extrair informações estatísticas específicas relacionadas ao gênero dos pacientes.

```
1   public static void InternamentsBySexo() throws SQLException,
    IOException {
2    final String sqlQuery = "SELECT\n"
3                + "    A.SEXO,\n"
4                + "    B.nome AS procedimento,\n"
5                + "    COUNT(*) AS total_internamentos\n"
6                + "FROM datasus.internamentos_ba AS A\n"
7                + "INNER JOIN procedimentos AS B ON A.PROC_REA = B.
    codproc\n"
8                + "WHERE A.SEXO = 1\n"
9                + "GROUP BY A.SEXO, B.procedimento\n"
10               + "ORDER BY total_internamentos DESC;";
11
12   conn.getConnection();
13   final ResultSet queryResultSet = conn.executeQuery(sqlQuery);
14   conn.closeConnection();
15   }
```

Código-fonte 20 – Método de consulta de internamentos por gênero.

APÊNDICE D – Script de Lançamento

Este apêndice apresenta o script de lançamento utilizado para executar a classe principal do projeto GreenDB. O script é projetado para coletar dados de eficiência energética usando o agente Java JoularJX em várias iterações. O número de iterações é fornecido como argumento ao executar o script. Durante cada iteração, o script executa a classe especificada e coleta os dados relevantes, movendo os resultados para diretórios apropriados.

```
trap 'rm -f joularJX-*.csv ; exit' INT

source <(grep = ../var.ini)

CLASS=greendb.Index

if [ $# -lt 1 ]; then
  echo 1>&2 "usage: bash $0 <number of iterations>"
  exit 2
fi

nb\\_iterations=$1
shift

now=$(date +%y.%m.%d-%Hh%Mm%Ss)

mkdir -p results/$now
mkdir -p graphs

for i in $(seq 1 $nb\\_iterations); do
    java -javaagent:$joularJX\\_jar -cp $CLASSPATH:$connector\\_jar
    $CLASS

    total\_energy="joularJX-*-methods-energy.csv"
    total\_filtered\_energy="joularJX-*-methods-energy-filtered.csv"
    power="joularJX-*-methods-power.csv"
    power\_filtered="joularJX-*-methods-filtered-power.csv"

```

```
28      filtered\_csv="./results/"$now"/"$i"\_"$now"\_joularJX-methods-energy
        -filtered.csv"
29      graph="./graphs/"$now"\_methods-consumption.png"
30
31      mv $total\_filtered\_energy $filtered\_csv
32      echo "JoularJX csv moved to $filtered\_csv"
33
34      rm -f $power
35      rm -f $power\_filtered
36      rm -f $total\_energy
37  done
38
39  echo "Creation of the graph..."
40  python3 jx\_plot.py "results/"$now $graph
41
42  eog $graph
```

Código-fonte 21 – Script de Lançamento da Classe com o JoularJX