UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO
DEPARTAMENTO DE ESTATÍSTICA E INFORMÁTICA
PROGRAMA DE PÓS–GRADUAÇÃO EM INFORMÁTICA APLICADA

HERDERSON GOMES COUTO

UMA ANÁLISE COMPARATIVA DE ENVELHECIMENTO DE SOFTWARE EM AMBIENTES COM BANCOS DE DADOS RELACIONAIS

RECIFE – PE

2023

HERDERSON GOMES COUTO

UMA ANÁLISE COMPARATIVA DE ENVELHECIMENTO DE SOFTWARE EM AMBIENTES COM BANCOS DE DADOS RELACIONAIS

Dissertação submetida à Coordenação do Programa de Pós-Graduação em Informática da Universidade Federal Rural de Pernambuco, como parte dos requisitos necessários para obtenção do grau de Mestre.

ORIENTADOR: Prof. Dr. Ermeson Carneiro de Andrade

RECIFE – PE

2023

Dados Internacionais de Catalogação na Publicação
Universidade Federal Rural de Pernambuco
Sistema Integrado de Bibliotecas
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

C871a Couto, Herderson Gomes
Uma análise comparativa de envelhecimento de software em ambientes com bancos de dados relacionais / Herderson Gomes Couto. - 2023.
113 f. : il.

Orientador: Ermeson Carneiro de Andrade.
Inclui referências.

Dissertação (Mestrado) - Universidade Federal Rural de Pernambuco, Programa de Pós-Graduação em Informática Aplicada, Recife, 2023.

1. Envelhecimento de software. 2. Análise estatística. 3. Banco de dados relacional. I. Andrade, Ermeson Carneiro de, orient. II. Título

CDD 004

HERDERSON GOMES COUTO

UMA ANÁLISE COMPARATIVA DE ENVELHECIMENTO DE SOFTWARE EM AMBIENTES COM BANCOS DE DADOS RELACIONAIS

Dissertação submetida à Coordenação do Programa de Pós-Graduação em Informática Aplicada da Universidade Federal Rural de Pernambuco, como parte dos requisitos necessários para obtenção do grau de Mestre.

Aprovada em: 27 de Outubro de 2023.

BANCA EXAMINADORA

Prof. Dr. Ermeson Carneiro de Andrade (Orientador)
Universidade Federal Rural de Pernambuco
Departamento de Computação

Prof. Dr. Gustavo Rau de Almeida Callou
Universidade Federal Rural de Pernambuco
Departamento de Computação

Prof. Dr. Rubens de Souza Matos Júnior
Instituto Federal de Sergipe
Campus Lagarto

Dedico este trabalho a todos aqueles que, de forma direta ou indireta, participaram ativamente de sua construção. O desfecho desta dissertação é a manifestação do esforço conjunto e das contribuições valiosas de diversas pessoas, tanto no âmbito pessoal quanto profissional.

Agradecimentos

Dedico esta seção de agradecimentos para expressar minha gratidão a todos que contribuíram para a realização desta dissertação.

Em primeiro lugar, a Deus, por me dar forças para poder enfrentar os desafios da vida. Agradeço a Ele por ter sido meu sustento e alento durante toda essa jornada.

À minha família e amigos, que sempre me apoiaram e encorajaram durante esta trajetória. Agradeço a minha amada esposa Milene Caroline pelos seus conselhos e palavras de incentivo que foram essenciais para eu não desistir em momentos difíceis.

Ao meu orientador, Ermeson Andrade, pelo suporte inestimável, paciência e orientação durante todo o processo de pesquisa. Suas sugestões e críticas foram fundamentais para o desenvolvimento deste trabalho.

À banca examinadora, pelos comentários, sugestões e contribuições à dissertação, que ajudaram a aprimorar o trabalho e enriquecer a discussão.

Aos professores do programa de pós-graduação, pelo conhecimento e ensinamentos compartilhados em sala de aula, que foram fundamentais para a minha formação acadêmica. Também agradeço ao Secretário do programa e a instituição por todo apoio durante o curso.

Este trabalho é fruto de um esforço coletivo, e sem o apoio de cada um, não teria sido possível.

Resumo

Em um contexto de sistemas computacionais que operam sem interrupção, o envelhecimento de software emerge como um desafio de grande relevância. Esse fenômeno pode afetar profundamente tais sistemas, resultando na degradação de recursos e possíveis quedas de desempenho, além de interrupções inesperadas durante o uso. O vazamento de memória e a degradação de desempenho frequentemente surgem como sintomas primordiais desse processo de envelhecimento. À medida que o software envelhece, sua eficiência e confiabilidade tendem a diminuir, tornando-se uma preocupação crítica, sobretudo quando o software é uma peça vital para inúmeras organizações. Consequentemente, a investigação minuciosa dos sintomas associados ao envelhecimento de software se torna essencial, permitindo identificar áreas problemáticas nos sistemas e facilitando a adoção de medidas proativas para mitigação de problemas. Os Sistemas de Gerenciamento de Banco de Dados (SGBDs) representam uma parcela particularmente vulnerável nesse cenário, dada sua importância crítica. Como componentes fundamentais de muitos sistemas modernos e serviços web, os SGBDs desempenham um papel essencial abrangendo uma variedade de setores. A notável prevalência dos SGBDs relacionais é resultado de suas exigências rigorosas de disponibilidade e operação contínua, frequentemente lidando com volumes substanciais de transações. Logo, abordar potenciais interrupções provenientes do envelhecimento de software é vital para garantir a satisfação dos usuários e a continuidade dos serviços. No ambiente dos SGBDs, a presença do envelhecimento de software acarreta riscos significativos, tais como a degradação de desempenho, instabilidade operacional e obstáculos à manutenção e evolução dos sistemas. O escopo deste trabalho abrange a investigação experimental e a comparação dos sintomas relacionados ao envelhecimento de software em ambientes de sistema que fazem uso dos SGBDs MySQL, MariaDB, PostgreSQL e SQL Server. Através de análises estatísticas do consumo de memória e da degradação do desempenho no tempo de resposta, nossos resultados apontam para a possível presença desse fenômeno nos ambientes desses SGBDs. Ademais, por meio de análises a nível de processos, conseguimos identificar quais processos contribuíram majoritariamente para o aumento no consumo de memória ao longo dos experimentos.

Palavras-chave: Envelhecimento de Software. Análise Estatística. Banco de dados Relacional. MySQL. MariaDB. PostgreSQL. SQL Server

Abstract

In a context of uninterrupted operating computer systems, software aging emerges as a highly relevant challenge. This phenomenon can profoundly affect such systems, resulting in resource degradation and potential performance drops, as well as unexpected interruptions during use. Memory leaks and performance degradation often manifest as primary symptoms of this aging process. As software ages, its efficiency and reliability tend to decrease, becoming a critical concern, especially when the software is a vital component for numerous organizations. Consequently, thorough investigation of symptoms associated with software aging becomes essential, enabling the identification of problematic areas in systems and facilitating the adoption of proactive measures to mitigate issues. Database Management Systems (DBMSs) represent a particularly vulnerable portion in this scenario, given their critical importance. As fundamental components of many modern systems and web services, DBMSs play an essential role spanning a variety of sectors. The notable prevalence of relational DBMSs is a result of their strict requirements for availability and continuous operation, often dealing with substantial transaction volumes. Therefore, addressing potential disruptions stemming from software aging is vital to ensure user satisfaction and service continuity. In the DBMS environment, the presence of software aging entails significant risks, such as performance degradation, operational instability, and hindrances to system maintenance and evolution. The scope of this work encompasses experimental investigation and comparison of symptoms related to software aging in system environments utilizing the MySQL, MariaDB, PostgreSQL, and SQL Server DBMSs. Through statistical analyses of memory consumption and performance degradation in response time, our results indicate the potential presence of this phenomenon in the environments of these DBMSs. Furthermore, through process-level analyses, we were able to identify which processes predominantly contributed to the increase in memory consumption throughout the experiments.

Keywords: Software Aging. Statistical Analysis. Relational Database. MySQL. MariaDB. PostgreSQL. SQL Server.

Lista de Figuras

Figura 1 – Estrutura experimental utilizada. 33
Figura 2 – Schema de banco de dados adotado nos experimentos. 35
Figura 3 – MySQL - Consumo de memória RAM - Carga Baixa. 42
Figura 4 – MySQL - Consumo de memória RAM - Carga Média. 43
Figura 5 – MySQL - Consumo de memória RAM - Carga Alta. 43
Figura 6 – MySQL - Tempo de Resposta - Carga Baixa. 45
Figura 7 – MySQL - Tempo de Resposta - Carga Média. 45
Figura 8 – MySQL - Tempo de Resposta - Carga Alta. 46
Figura 9 – MySQL - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Baixa. 48
Figura 10 – MySQL - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Média. 49
Figura 11 – MySQL - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Alta. 49
Figura 12 – Crescimento do processo do MySQL ao longo do tempo de teste - Carga Baixa. 53
Figura 13 – Crescimento do processo do MySQL ao longo do tempo de teste - Carga Média. 54
Figura 14 – Crescimento do processo do MySQL ao longo do tempo de teste - Carga Alta. 54
Figura 15 – MariaDB - Consumo de memória RAM - Carga Baixa. 56
Figura 16 – MariaDB - Consumo de memória RAM - Carga Média. 57
Figura 17 – MariaDB - Consumo de memória RAM - Carga Alta. 57
Figura 18 – MariaDB - Tempo de Resposta - Carga Baixa. 59
Figura 19 – MariaDB - Tempo de Resposta - Carga Média. 59
Figura 20 – MariaDB - Tempo de Resposta - Carga Alta. 60
Figura 21 – MariaDB - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Baixa. 62
Figura 22 – MariaDB - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Média 62

Figura 23 – MariaDB - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Alta. 63
Figura 24 – Crescimento do processo do MariaDB ao longo do tempo de teste - Carga Baixa 66
Figura 25 – Crescimento do processo do MariaDB ao longo do tempo de teste - Carga Média 67
Figura 26 – Crescimento do processo do MariaDB ao longo do tempo de teste - Carga Alta 68
Figura 27 – PostgreSQL - Consumo de memória RAM - Carga Baixa. 69
Figura 28 – PostgreSQL - Consumo de memória RAM - Carga Média. 70
Figura 29 – PostgreSQL - Consumo de memória RAM - Carga Alta. 71
Figura 30 – PostgreSQL - Tempo de Resposta - Carga Baixa. 72
Figura 31 – PostgreSQL - Tempo de Resposta - Carga Média. 73
Figura 32 – PostgreSQL - Tempo de Resposta - Carga Alta. 74
Figura 33 – PostgreSQL - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Baixa. 76
Figura 34 – PostgreSQL - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Média. 77
Figura 35 – PostgreSQL - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Alta. 77
Figura 36 – Crescimento dos processos do PosgreSQL ao longo do tempo de teste - Carga Baixa 81
Figura 37 – Crescimento dos processos do PosgreSQL ao longo do tempo de teste - Carga Média 82
Figura 38 – Crescimento dos processos do PosgreSQL ao longo do tempo de teste - Carga Alta 82
Figura 39 – SQL Server - Consumo de memória RAM - Carga Baixa. 84
Figura 40 – SQL Server - Consumo de memória RAM - Carga Média. 84
Figura 41 – SQL Server - Consumo de memória RAM - Carga Alta. 85
Figura 42 – SQL Server - Tempo de Resposta - Carga Baixa. 86
Figura 43 – SQL Server - Tempo de Resposta - Carga Média. 87
Figura 44 – SQL Server - Tempo de Resposta - Carga Alta. 88

Figura 45 – SQL Server - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Baixa. 90
Figura 46 – SQL Server - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Média. 91
Figura 47 – SQL Server - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Alta. 91
Figura 48 – Crescimento dos processos do SQL Server ao longo do tempo de teste - Carga Baixa 95
Figura 49 – Crescimento dos processos do SQL Server ao longo do tempo de teste - Carga Média 96
Figura 50 – Crescimento dos processos do SQL Server ao longo do tempo de teste - Carga Alta 96

Lista de Tabelas

Tabela 1 – Configuração de hardware dos computadores utilizados. 33
Tabela 2 – Parâmetros usados nos testes - Consultas retornando 4000 tuplas. . . . 34
Tabela 3 – MySQL - Teste de Mann-Kendall e Sen's slope para o uso de memória RAM. 44
Tabela 4 – MySQL - Teste de Mann-Kendall e Sen's slope para o tempo de resposta. 47
Tabela 5 – MySQL - Descrição dos processos que mais consumiram memória - Carga Baixa. 48
Tabela 6 – MySQL - Processos que mais cresceram juntamente com o SGBD - Carga Baixa. 51
Tabela 7 – MySQL - Descrição dos processos que mais cresceram juntamente com o SGBD - Carga Baixa 51
Tabela 8 – MySQL - Processos que mais cresceram juntamente com o SGBD - Carga Média. 52
Tabela 9 – MySQL - Processos que mais cresceram juntamente com o SGBD - Carga Alta. 52
Tabela 10 – MySQL - Teste de Mann-Kendall e Sen's slope para o processo do SGBD. 55
Tabela 11 – MariaDB - Teste de Mann-Kendall e Sen's slope para o uso de memória RAM. 58
Tabela 12 – MariaDB - Teste de Mann-Kendall e Sen's slope para o tempo de resposta. 60
Tabela 13 – MariaDB - Descrição dos processos que mais consumiram memória - Carga Baixa. 61
Tabela 14 – MariaDB - Processos que mais cresceram juntamente com o SGBD - Carga Baixa. 64
Tabela 15 – MariaDB - Descrição dos processos que mais cresceram juntamente com o SGBD - Carga Baixa 64
Tabela 16 – MariaDB - Processos que mais cresceram juntamente com o SGBD - Carga Média. 65
Tabela 17 – MariaDB - Processos que mais cresceram juntamente com o SGBD - Carga Alta. 65

Tabela 18 – MariaDB - Teste de Mann-Kendall e Sen's slope para o processo do SGBD. 68
Tabela 19 – PostgreSQL - Teste de Mann-Kendall e Sen's slope para o uso de memória RAM. 72
Tabela 20 – PostgreSQL - Teste de Mann-Kendall e Sen's slope para o tempo de resposta. 74
Tabela 21 – PostgreSQL - Descrição dos processos que mais consumiram memória - Carga Baixa. 76
Tabela 22 – PostgreSQL - Processos que mais cresceram juntamente com o SGBD - Carga Baixa. 79
Tabela 23 – PostgreSQL - Descrição dos processos que mais cresceram juntamente com o SGBD - Carga Baixa 79
Tabela 24 – PostgreSQL - Processos que mais cresceram juntamente com o SGBD - Carga Média. 80
Tabela 25 – PostgreSQL - Processos que mais cresceram juntamente com o SGBD - Carga Alta. 80
Tabela 26 – PostgreSQL - Teste de Mann-Kendall e Sen's slope para os processos do SGBD. 83
Tabela 27 – SQL Server - Teste de Mann-Kendall e Sen's slope para o uso de memória RAM. 86
Tabela 28 – SQL Server - Teste de Mann-Kendall e Sen's slope para o tempo de resposta. 88
Tabela 29 – SQL Server - Descrição dos processos que mais consumiram memória - Carga Baixa. 90
Tabela 30 – SQL Server - Processos que mais cresceram juntamente com o SGBD - Carga Baixa. 93
Tabela 31 – SQL Server - Descrição dos processos que mais cresceram juntamente com o SGBD - Carga Baixa 93
Tabela 32 – SQL Server - Processos que mais cresceram juntamente com o SGBD - Carga Média. 94
Tabela 33 – SQL Server - Processos que mais cresceram juntamente com o SGBD - Carga Alta. 94

Tabela 34 – SQL Server - Teste de Mann-Kendall e Sen's slope para os processos do SGBD. 97
Tabela 35 – Tabela comparativa de envelhecimento para o uso de RAM. 99
Tabela 36 – Tabela comparativa de envelhecimento para o Tempo de Resposta. . . 101
Tabela 37 – Tabela comparativa de envelhecimento dos processos dos SBGDs. . . . 102

Lista de Siglas

ACID	Atomicidade, Consistência, Isolamento, Durabilidade
CPU	Central Processing Unit
FTP	File Transfer Protocol
HD	Hard Disc
PID	Process Identifier
PSS	Proportion Set Size
RAM	Random Acess Memory
RSS	Resident Set Size
SGBD	Sistema de Gerenciamento de Banco de Dados
SQL	Structured Query Language
SSD	Solid State Drive
USS	Unique Set Size

Sumário

1 Introdução 16

1.1 Motivação e Justificativa 18

1.2 Trabalhos Relacionados 18

1.3 Objetivos 22

1.4 Organização da Dissertação 22

2 Fundamentação Teórica 24

2.1 Banco de Dados Relacionais 24

2.1.1 MySQL 25

2.1.2 MariaDB 26

2.1.3 PostgreSQL 26

2.1.4 SQL Server 27

2.2 Envelhecimento de Software 28

2.2.1 Vazamento de Memória 29

2.2.2 Degradação de Desempenho 30

3 Plano Experimental 32

3.1 Ambiente de Testes 32

3.2 Execução dos Experimentos 33

3.3 Coleta dos Dados e Análises 35

3.3.1 Coleta de Dados 35

3.3.2 Análises 37

3.3.2.1 Teste de Tendência 37

3.3.2.2 Teste de Mann-Kendall 38

3.3.2.3 Estimador Sen's Slope 39

4 Resultados 41

4.1 MySQL 41

4.1.1 Análise do Consumo de Memória 42

4.1.2 Análise do Tempo de Resposta 44

4.1.3 Análise dos Processos 47

4.1.3.1 Processos que mais consumiram Memória 47

4.1.3.2 Processos que mais cresceram juntamente com o SGBD . 50

4.1.3.3 Análise dos Processos do MySQL . . . 53
4.2 MariaDB . . . 55
4.2.1 Análise do Consumo de Memória . . . 55
4.2.2 Análise do Tempo de Resposta . . . 58
4.2.3 Análise dos Processos . . . 61
4.2.3.1 Processos que mais consumiram Memória . . . 61
4.2.3.2 Processos que mais cresceram juntamente com o SGBD . 63
4.2.3.3 Análise dos Processos do MariaDB . . . 66
4.3 PostgreSQL . . . 69
4.3.1 Análise do Consumo de Memória . . . 69
4.3.2 Análise do Tempo de Resposta . . . 72
4.3.3 Análise dos Processos . . . 75
4.3.3.1 Processos que mais consumiram Memória . . . 75
4.3.3.2 Processos que mais cresceram juntamente com o SGBD . 78
4.3.3.3 Análise dos Processos do PostgreSQL . . . 81
4.4 SQL Server . . . 83
4.4.1 Análise do Consumo de Memória . . . 83
4.4.2 Análise do Tempo de Resposta . . . 86
4.4.3 Análise dos Processos . . . 89
4.4.3.1 Processos que mais consumiram Memória . . . 89
4.4.3.2 Processos que mais cresceram juntamente com o SGBD . 92
4.4.3.3 Análise dos Processos do SQL Server . . . 95
4.5 Comparativos dos Efeitos do Envelhecimento de Software . . . 97
4.5.1 Comparativo do Consumo de Memória . . . 97
4.5.2 Comparativo do Tempo de Resposta . . . 99
4.5.3 Comparativo da Análise de Processos . . . 101
4.6 Ameaças à Validade . . . 103
4.7 Considerações Finais . . . 103
5 Conclusão e Trabalhos Futuros . . . 105
5.1 Contribuições . . . 106
5.2 Trabalhos Futuros . . . 107
Referências . . . 108

1 Introdução

Em um mundo cada vez mais digital e interconectado, os Sistemas de Gerenciamento de Banco de Dados (SGBDs) desempenham um papel vital. Esses sistemas estão presentes em vários setores da sociedade, desde as esferas empresariais e governamentais até as plataformas digitais de comércio e interação social. Eles desempenham um papel crucial ao possibilitar o armazenamento e acesso eficiente a uma diversidade de informações. Dentre os tipos de SGBDs existentes, os sistemas relacionais são os mais amplamente utilizados (DB-ENGINES, 2023).

Esses SGBDs, em sua concepção, são projetados para assegurar uma disponibilidade ininterrupta, demonstrando uma notável capacidade de operar continuamente mesmo sob cargas de trabalho substanciais (MINHAS et al., 2013). A operação ininterrupta, aliada à eficiência no processamento de transações, coloca esses sistemas como alicerces críticos para as operações diárias de diversas organizações. Entretanto, em um ambiente que demanda constante performance e confiabilidade, os riscos decorrentes de eventuais interrupções ganham relevância. Mesmo os sistemas mais robustos estão suscetíveis à ocorrência de falhas, muitas vezes na forma de *bugs* que podem resultar em travamentos, lentidão e instabilidade (HE, 2019).

Em adição aos desafios inerentes ao desenvolvimento de software, existe um fenômeno que merece atenção especial: o envelhecimento de software. Trata-se de um processo gradual que culmina na deterioração das características de um sistema, podendo causar impactos significativos no seu desempenho e na sua confiabilidade ao longo do tempo (PARNAS, 1994). O vazamento de memória e a degradação de desempenho figuram como um dos sintomas mais proeminentes do envelhecimento de software (GROTTKE et al., 2008). A presença desses sintomas em ambientes de execução de SGBDs pode acarretar consequências prejudiciais, tais como aumento nos tempos de resposta e ocorrências de falhas que ameaçam a continuidade operacional.

Atualmente, existe uma ampla gama de estudos na literatura que se dedica a analisar o envelhecimento de software em sistemas de computador. Nos estudos (PIETRANTUONO; RUSSO, 2020; COTRONEO et al., 2014), foram conduzidas várias pesquisas abrangentes para explorar o impacto do envelhecimento em diversos tipos de sistemas computacionais, incluindo ambientes de nuvem e o servidor web Apache. Apesar disso, são escassos os

estudos que investigaram os sintomas e os efeitos do envelhecimento de software em ambientes de banco de dados. Um dos estudos que abordaram esse tema é o trabalho de (BOVENZI et al., 2012), no qual os autores examinaram os efeitos do envelhecimento de software causado pela manifestação de erros no MySQL. No entanto, esses tipos de estudos costumam focalizar em um único banco de dados e apresentam carência de uma análise de processo abrangente para investigar as possíveis causas do envelhecimento de software.

Neste contexto, a presente dissertação tem por objetivo investigar os efeitos do envelhecimento de software em ambientes de banco de dados relacionais, com um foco específico na análise comparativa entre diferentes SGBDs, incluindo MySQL, MariaDB, PostgreSQL e SQL Server, implantados em um ambiente operacional Linux. Por meio de uma série de experimentos, foram conduzidas consultas em tabelas previamente definidas, variando intervalos entre requisições e diferentes quantidades de usuários simultâneos. Métricas, tais como o uso de memória RAM e tempo de resposta, foram coletadas e submetidas aos testes estatísticos de Mann-Kendall (MANN, 1945) (KENDALL, 1948) e do estimador Sen's slope (SEN, 1968), a fim de confirmar estatisticamente as suspeitas de envelhecimento observada nos dados. Além disso, o estudo identificou os processos potencialmente responsáveis por induzir o envelhecimento nos ambientes. Nossos achados indicaram a presença de potenciais efeitos e sintomas do envelhecimento do software nos ambientes examinados. Além disso, verificamos que outros processos do sistema operacional, além dos processos dos SGBDs, podem colaborar para esse envelhecimento. Por último, notamos que a deterioração do ambiente pode levar a um aumento no tempo de resposta.

Os resultados obtidos oferecem visões valiosas sobre a presença e a natureza do envelhecimento de software em SGBDs populares, demonstrando sua relevância prática e implicação na performance operacional. Adicionalmente, a metodologia de análise empregada neste estudo pode servir como um recurso valioso para profissionais de Tecnologia da Informação e desenvolvedores na identificação precoce e mitigação contínua de problemas de envelhecimento de software em ambientes de bancos de dados complexos e dinâmicos. Consequentemente, a investigação realizada contribui não apenas para uma compreensão mais profunda dos desafios enfrentados pelos SGBDs modernos, mas também para a construção de estratégias proativas que assegurem a confiabilidade e a performance contínuas desses sistemas essenciais.

1.1 Motivação e Justificativa

Os bancos de dados relacionais desempenham um papel crucial, armazenando e gerenciando os dados vitais que sustentam a funcionalidade de muitos sistemas. No entanto, diante de situações de alta demanda e à medida que os volumes de dados e a complexidade das operações aumentam, esses sistemas podem enfrentar problemas relacionados ao desempenho que podem levar ao surgimento de bugs, lentidão nas operações com tempo de respostas prolongados e ineficiências operacionais. Muitos desses problemas podem estar associados ao fenômeno de envelhecimento de software (PARNAS, 1994). Portanto, estudos relacionados a esse assunto são extremamente necessários, uma vez que SGBDs podem ser ativos críticos em uma organização (GROVER; TENG, 1992).

Na literatura, trabalhos como (MACHIDA et al., 2012b) e (MACHIDA et al., 2016) foram conduzidos para detecção e resolução de problemas relacionados ao envelhecimento de software em sistemas de armazenamento. No entanto, são escassos os trabalhos que investigaram o fenômeno do envelhecimento de software em ambientes de banco de dados, sendo um exemplo deles o estudo (BOVENZI et al., 2012). Assim, nosso trabalho surge devido à carência de pesquisas abrangentes que relacionam o envelhecimento de software aos ambientes de bancos de dados relacionais, buscando preencher essa lacuna por meio de uma análise comparativa detalhada. Ao empregar SGBDs amplamente reconhecidos, a presente pesquisa objetiva investigar como o envelhecimento do software afeta o desempenho em ambientes de bancos de dados, incluindo a degradação da memória ao longo do tempo. Essa compreensão é vital para orientar estratégias de manutenção e aprimoramento, contribuindo para a extensão da vida útil dos sistemas e garantindo a continuidade operacional.

1.2 Trabalhos Relacionados

Nesta seção, serão apresentados os estudos que utilizamos como referência no desenvolvimento do trabalho, os quais desempenharam um papel fundamental na compreensão abrangente do nosso campo de pesquisa. Nos últimos anos, a temática do envelhecimento de software tem atraído considerável interesse tanto no meio acadêmico quanto na indústria, resultando em uma profusão de publicações científicas e patentes registradas. À medida que novas tecnologias emergem e são adotadas, a pesquisa relacionada ao envelhecimento de software continua a se expandir. No entanto, é notável a

escassez significativa de estudos que explorem os indicadores de envelhecimento de software em ambientes exclusivamente baseados em SGBDs.

Como resultado, apresentamos inicialmente algumas estudos que se concentram exclusivamente na análise de desempenho em sistemas de banco de dados. Posteriormente, discutimos estudos abrangentes que se focalizam na análise do envelhecimento de software em contextos computacionais mais amplos e que em alguns casos fizeram comparativos de envelhecimento em ambientes computacionais. Em seguida, abordamos pesquisas que examinaram o envelhecimento de software em sistemas de armazenamento e SGBDs. Por fim, realizamos uma comparação entre os trabalhos aqui relacionados e o presente trabalho.

Diversos estudos conduziram análises de desempenho em sistemas de gerenciamento de bancos de dados. No estudo apresentado em (PIRES et al., 2006), uma análise comparativa entre os dois sistemas de gerenciamento de bancos de dados de código aberto mais amplamente utilizados globalmente, o MySQL e o PostgreSQL, foi conduzida por meio do benchmark OSDB. O propósito dessa pesquisa consistiu em coletar as métricas essenciais para possibilitar um confronto em termos de desempenho entre esses sistemas. Uma abordagem similar também foi empreendida em (KLIMEK; SKUBLEWSKA-PASZKOWSKA, 2021), onde uma análise comparativa de desempenho foi executada entre os SGBDs PostgreSQL e MySQL. Esse estudo englobou diversos elementos atinentes à performance desses sistemas, abarcando o tempo de resposta, a escalabilidade, a utilização de recursos e outros fatores relevantes. Por sua vez, em (ILIĆ et al., 2021), uma análise comparativa de desempenho entre os sistemas SQL Server e Oracle foi conduzida comparando diversos aspectos de desempenho.

No estudo de (SANTOS et al., 2019), uma avaliação de desempenho foi executada, focalizando a análise de sensibilidade em contêineres Docker que executavam os SGBDs relacionais MySQL, PostgreSQL e SQLServer. O principal objetivo desse estudo foi determinar qual sistema de gerenciamento de banco de dados relacional deveria ser escolhido ao ser utilizado em conjunto com o Docker, visando a obtenção do melhor desempenho. Outra investigação comparativa de desempenho entre sistemas de gerenciamento de banco de dados foi realizado em (POLITOWSKI; MARAN, 2014), no qual os sistemas comparados foram o MongoDB e o PostgreSQL. Os testes foram divididos em três categorias: inserção de dados, busca simples e busca complexa. Os resultados destacaram o MongoDB como o sistema com melhor desempenho, com base na análise do tempo de execução das operações

de leitura e escrita.

Na literatura, diversos estudos têm abordado o fenômeno do envelhecimento de software em ambientes computacionais. No contexto das plataformas de *Blockchain*, algumas investigações também foram conduzidas com relação a esse tipo de envelhecimento. Um exemplo é o estudo (DIAS; ANDRADE, 2022), no qual foi conduzida uma análise com o intuito de explorar os eventos que desencadeiam o envelhecimento de software na plataforma de *Blockchain* Cardano. Em outro estudo, (MELO et al., 2022) realizaram uma avaliação da plataforma de *Blockchain* Hyperledger Fabric focando na detecção de possíveis vazamentos de recursos.

Além das plataformas de *Blockchain*, também foram realizadas pesquisas sobre o envelhecimento de software em contêineres. As investigações conduzidas em (VINÍCIUS et al., 2022) e (OLIVEIRA et al., 2021) analisaram sintomas associados ao envelhecimento de software na plataforma de contêiner Docker. Não se limitando a esses domínios, outras investigações abordaram o envelhecimento de software no contexto do sistema operacional Android. O estudo (HUO et al., 2018) empregou algoritmos de *machine learning* para a detecção do envelhecimento de software no sistema Android. Por outro lado, em (QIAO et al., 2016) conduziram uma análise empírica das manifestações de *bugs* associados ao envelhecimento de software nesse sistema operacional.

Outras pesquisas focaram em ambientes web. No estudo (FICCO et al., 2018), uma análise foi empreendida para identificar os sintomas e as origens do envelhecimento do software no contexto do sistema de processamento de fluxo de eventos Apache Storm. No trabalho (GROTTKE et al., 2006), foi investigada a degradação do desempenho de um servidor web Apache sob cargas de trabalho aplicadas durante um período de 25 dias.

Também é possível encontrar estudos que realizaram comparações relativas ao envelhecimento do software em ambientes computacionais. Em (ANDRADE et al., 2020), os pesquisadores exploraram o fenômeno de envelhecimento do software em sistemas de classificação de imagens executados de forma contínua. Essas comparações foram efetuadas em dois contextos distintos: ambiente em nuvem e ambiente de borda. Em um estudo subsequente (ANDRADE et al., 2021b), os mesmos autores examinaram as suspeitas de envelhecimento em sistemas de classificação de imagens, abordando uma quantidade mais ampla de dados para as análises realizadas tanto em ambientes de computação em nuvem quanto em ambientes de borda.

Diversas pesquisas têm se concentrado em sistemas de armazenamento. No estudo (MACHIDA et al., 2012b), foi apresentada uma nova abordagem denominada "extensão da vida útil do software". Essa abordagem se baseia na alocação dinâmica de recursos e no controle da carga de trabalho, com o propósito de enfrentar o envelhecimento do software. A viabilidade e eficácia dessa estratégia são comprovadas por meio de experimentos envolvendo o Memcached, um sistema distribuído de armazenamento em memória. Uma ampliação desse trabalho foi realizada em um estudo posterior (MACHIDA et al., 2016). Nesse estudo, a eficácia da extensão da vida útil do software contra o envelhecimento do software é examinada em maior profundidade. Além disso, é identificado o intervalo ótimo para a aplicação dessa extensão, no qual a disponibilidade do sistema é otimizada por meio de modelagem estocástica.

Outro estudo relevante é o trabalho (MATOS et al., 2012), onde os autores analisaram o impacto do envelhecimento de software nos mecanismos de armazenamento elástico em nuvens privadas. Já o trabalho (MACHIDA et al., 2012a) investigou problemas relacionados ao envelhecimento em software de código aberto utilizado em sistemas de computação em nuvem, como Hadoop, MapReduce, Cassandra, Eucalyptus, entre outros. Em outra perspectiva, o estudo (OKAMURA et al., 2017) propôs um modelo estatístico baseado em um Modelo de Markov Oculto Contínuo para compreender e modelar o envelhecimento de software. Esse modelo estimou o processo de degradação de software utilizando dados experimentais provenientes de um sistema de banco de dados MySQL. No entanto, é notável a escassez de estudos que investigam os sintomas de envelhecimento de software em ambientes exclusivamente dedicados a SGBDs. Entre esses estudos, destaca-se a pesquisa (BOVENZI et al., 2012), na qual os efeitos do envelhecimento do software, resultantes da ativação de bugs relacionados à simultaneidade, foram examinados no contexto do MySQL.

Diferentemente das pesquisas previamente mencionadas, nosso objetivo principal consiste em investigar a manifestação do envelhecimento de software em SGBDs na forma de uma análise comparativa abrangente entre ambientes. Normalmente, os estudos sobre o envelhecimento de software em SGBDs se concentram em um único banco de dados, deixando lacunas na exploração das possíveis razões por trás desse envelhecimento. No âmbito do nosso trabalho, optamos por abordar esse desafio de maneira diferenciada. Utilizamos quatro SGBDs amplamente reconhecidos – MySQL, MariaDB, PostgreSQL e

SQL Server – para conduzir um estudo comparativo sobre o envelhecimento de software. Nossa abordagem envolveu a avaliação da degradação do uso de memória e do desempenho ao longo do tempo, empregando testes estáticos de Mann-Kendall e de Sen's slope para confirmar nossas suspeitas.

Adicionalmente, uma característica distintiva do nosso estudo é a análise abrangente dos processos envolvidos. Por meio desse enfoque, conseguimos confirmar, em certos casos, a presença de envelhecimento de software nos SGBDs analisados. Também pudemos identificar processos suspeitos que possivelmente contribuíram para esse envelhecimento nos ambientes investigados. Este trabalho é pioneiro na medida em que representa uma análise abrangente sobre o envelhecimento de software em ambientes de bancos de dados relacionais, englobando uma gama significativamente ampla de SGBDs.

1.3 Objetivos

O objetivo central deste estudo é investigar os sintomas associados ao envelhecimento de software e realizar uma análise comparativa desse fenômeno em ambientes que utilizam Sistemas de Gerenciamento de Bancos de Dados relacionais, dentro de uma arquitetura cliente-servidor. Mais especificamente, os objetivos deste trabalho são:

- Detectar e comparar indícios de envelhecimento de software em ambientes de banco de dados relacionais;
- Identificar os processos responsáveis pelo envelhecimento de software;
- Analisar o impacto do uso de diferentes cargas de trabalho no desempenho e no consumo de recursos dos SGBDs utilizados; e
- Conduzir testes estatísticos para confirmar as suspeitas de deterioração de recursos e desempenho nos ambientes analisados.

1.4 Organização da Dissertação

Esta dissertação encontra-se organizada da seguinte maneira. O Capítulo 2 apresenta a fundamentação do trabalho, no qual os principais conceitos são introduzidos com o objetivo de proporcionar ao leitor uma compreensão adequada do conteúdo apresentado. O Capítulo 3 detalha o plano experimental, fornecendo informações

minuciosas sobre o ambiente adotado e o estudo conduzido. No Capítulo 4, são apresentados os resultados obtidos e as análises realizadas. O Capítulo 5 traz as conclusões, juntamente com as contribuições deste trabalho, além de descrever possíveis direções para trabalhos futuros.

2 Fundamentação Teórica

O objetivo deste capítulo é apresentar os principais conceitos sobre os temas abordados para um melhor entendimento do trabalho. Primeiramente, são apresentados um breve histórico e os principais conceitos dos banco de dados relacionais, ressaltando sua importância no mundo moderno, além de fornecer informações sobre os SGBDs utilizados nos experimentos realizados. Por fim, são introduzidos conceitos sobre envelhecimento de software com ênfase no vazamento de memória e na degradação de desempenho.

2.1 Banco de Dados Relacionais

O modelo de banco de dados relacional foi proposto por Edgar F. Codd, um matemático e cientista da computação britânico, em um artigo publicado em 1970. Codd trabalhava na IBM Research na época e seu artigo, intitulado *A Relational Model of Data for Large Shared Data Banks*, descrevia um novo modelo de gerenciamento de dados que era baseado em uma estrutura de tabelas relacionais (CODD, 1970). O modelo relacional proposto por Codd introduziu a ideia de que as informações poderiam ser organizadas em tabelas com relações entre elas, em vez de em arquivos separados. Esse modelo foi uma mudança radical em relação aos modelos de gerenciamento de dados anteriores, como o modelo hierárquico e o modelo em rede, que eram baseados em estruturas de árvore e grafo, respectivamente.

Após a publicação do artigo de Codd, o modelo de banco de dados relacional começou a ganhar popularidade e rapidamente se tornou o padrão da indústria para os sistemas de gerenciamento de banco de dados. O primeiro sistema comercial de banco de dados relacional, chamado de Oracle, foi lançado pela Oracle Corporation em 1979. Desde então, o modelo relacional tem sido amplamente utilizado em aplicações empresariais e em muitos outros setores. Eles são projetados para armazenar e manipular grandes quantidades de informações de maneira estruturada e eficiente. Os bancos de dados relacionais são compostos por tabelas que possuem atributos/colunas e tuplas/linhas. Cada coluna representa um atributo ou característica da entidade, enquanto cada linha representa uma instância ou ocorrência da entidade (HEUSER, 2009). Por exemplo, uma tabela de clientes pode ter colunas para nome, endereço e telefone, e cada linha seria um

cliente diferente. A relação entre as tabelas é estabelecida por meio de chaves estrangeiras, que são colunas que se referem a outras tabelas. Isso permite que as informações sejam conectadas e relacionadas, facilitando a realização de consultas e a obtenção de resultados precisos.

Uma das vantagens dos bancos de dados relacionais é a sua capacidade de garantir a integridade dos dados por meio de regras e restrições (DATE, 2004). Isso significa que os dados são protegidos contra erros e inconsistências, ajudando a manter a qualidade dos dados ao longo do tempo. Além disso, os bancos de dados relacionais são altamente escaláveis e podem ser usados em uma ampla variedade de aplicativos, desde pequenos sistemas de gerenciamento de estoque até grandes sistemas corporativos.

Esses sistemas fornecem uma estrutura clara para relacionar os dados, permitindo que os usuários realizem consultas complexas e obtenham resultados precisos por meio da linguagem declarativa padrão SQL (*Structured Query Language*). Devido as suas capacidades de integridade de dados, escalabilidade e flexibilidade, os bancos de dados relacionais são os mais utilizados no mundo (DB-ENGINES, 2023). Neste trabalho, nossa ênfase recai sobre a avaliação de quatro proeminentes sistemas de gerenciamento de bancos de dados: MySQL, MariaDB, PostgreSQL e SQL Server.

2.1.1 MySQL

O MySQL é um sistema de gerenciamento de banco de dados relacional de código aberto (MYSQL, 2023). Ele foi criado em 1995 por uma empresa sueca chamada MySQL AB e posteriormente foi adquirido pela Oracle Corporation em 2010. O MySQL é amplamente utilizado em aplicativos da web, em sistemas de gerenciamento de conteúdo e em muitos outros aplicativos empresariais. Ele é conhecido por sua facilidade de uso, desempenho, escalabilidade e flexibilidade. Ele suporta muitos recursos avançados, como transações ACID (Atomicidade, Consistência, Isolamento e Durabilidade), replicação de banco de dados em tempo real e particionamento de banco de dados.

Uma grande vantagem do MySQL é sua compatibilidade com muitas linguagens de programação, incluindo PHP, Python, Ruby, C++, C#, Java e muitas outras. Isso significa que os desenvolvedores podem integrar facilmente o MySQL em seus aplicativos e sites da web, independentemente da linguagem de programação que estiverem usando. O

MySQL também é conhecido por sua escalabilidade, o que significa que pode lidar com grandes volumes de dados e muitos usuários simultâneos (MILANI, 2007). Por ser um software de código aberto, o MySQL emerge como uma escolha econômica para empresas de qualquer porte, uma vez que não envolve despesas relacionadas a licenças.

2.1.2 MariaDB

O MariaDB é um sistema de gerenciamento de banco de dados relacional de código aberto, criado como uma ramificação do MySQL após sua aquisição pela Oracle Corporation em 2010 (MARIADB, 2023). Esse SGBD foi desenvolvido pela MariaDB Corporation, uma empresa sediada na Finlândia e mantida por uma comunidade global de desenvolvedores. O MariaDB é conhecido por sua estabilidade, desempenho e segurança, bem como por sua escalabilidade e flexibilidade. Ele é amplamente utilizado em empresas de todos os tamanhos, desde pequenas startups até grandes corporações, em setores que vão desde tecnologia da informação até finanças, saúde e governo.

Uma das principais vantagens do MariaDB é sua comunidade de desenvolvedores ativa e engajada, que trabalha constantemente para melhorar e expandir o software. Como um projeto de código aberto, qualquer pessoa pode contribuir para o MariaDB, tornando-o uma opção altamente personalizável para atender às necessidades específicas de uma organização. Outra vantagem do MariaDB é seu baixo custo, já que é um software de código aberto e pode ser usado sem custos de licenciamento (DYER, 2015). Além disso, a MariaDB Corporation oferece suporte e serviços de consultoria para empresas que desejam adotar o MariaDB em suas operações.

2.1.3 PostgreSQL

O PostgreSQL é um sistema de gerenciamento de banco de dados relacional de código aberto, que é conhecido por sua confiabilidade, integridade e escalabilidade. O PostgreSQL teve suas origens em 1986 como parte do projeto POSTGRES da Universidade da Califórnia em Berkeley. Desde então, tem sido desenvolvido ativamente por mais de 35 anos (POSTGRESQL, 2023). Uma das principais características do PostgreSQL é sua capacidade de ser estendido e personalizado. Ele fornece uma estrutura poderosa para

desenvolvedores criarem seus próprios tipos de dados personalizados, funções e operadores. Isso significa que o PostgreSQL pode ser personalizado para atender às necessidades específicas de uma empresa ou organização.

Uma outra grande vantagem do PostgreSQL é sua segurança. Ele inclui recursos avançados de segurança, como autenticação de usuário, criptografia de dados em trânsito e em repouso, controle de acesso granular, etc (MILANI, 2008). Isso torna o PostgreSQL uma opção segura para empresas que desejam proteger seus dados e garantir que eles estejam em conformidade com as regulamentações governamentais. Além disso, o PostgreSQL é um software de código aberto, o que significa que é livre para uso e modificações.

O PostgreSQL pode ser considerado um sistema de gerenciamento de banco de dados poderoso, seguro e escalável, que é amplamente utilizado em aplicativos empresariais de missão crítica em todo o mundo. Ele suporta muitos recursos avançados e pode ser estendido e personalizado para atender às necessidades específicas de uma organização.

2.1.4 SQL Server

O SQL Server é um sistema de gerenciamento de banco de dados relacional desenvolvido pela Microsoft (SQLSERVER, 2023). Ele é conhecido por sua confiabilidade, escalabilidade e recursos avançados, que o tornam uma escolha popular para aplicativos empresariais de missão crítica em todo o mundo.

Uma das principais vantagens do SQL Server é sua integração com outras tecnologias Microsoft, como o .NET Framework e o Azure. Isso significa que os desenvolvedores podem criar aplicativos que aproveitam as tecnologias da Microsoft e integrá-los com o SQL Server para fornecer uma experiência de usuário integrada e confiável (MISTRY; MISNER, 2014). Além disso, o SQL Server possui uma ampla gama de recursos de segurança, incluindo autenticação de usuário, criptografia de dados em trânsito e em repouso, controle de acesso granular e muito mais. Isso torna o SQL Server uma escolha popular para empresas que desejam proteger seus dados e garantir que eles estejam em conformidade com as regulamentações governamentais.

O SQL Server também possui uma ampla gama de ferramentas de gerenciamento e monitoramento que permite aos administradores de banco de dados gerenciar, monitorar e otimizar o desempenho do banco de dados em tempo real. Isso ajuda empresas a manterem

seus aplicativos funcionando de forma rápida e eficiente, garantindo a satisfação do usuário e reduzindo o tempo de inatividade do sistema. Como um produto da Microsoft, o SQL Server oferece suporte empresarial e serviços de consultoria, o que significa que as empresas podem obter suporte especializado e serviços personalizados para ajudá-las a implementar e gerenciar seus bancos de dados SQL Server.

2.2 Envelhecimento de Software

O envelhecimento de software é um fenômeno que pode afetar os sistemas computacionais, resultando na deterioração do sistema, tendo potencial para diminuir o desempenho do software e até mesmo causar interrupções inesperadas durante o seu uso (PARNAS, 1994). Em 1995, Huang et al. estudaram o processo de degradação de software durante sua execução e exploraram a correlação entre o aumento das taxas de falha e o tempo de execução (HUANG et al., 1995). Esse trabalho inaugurou a área de pesquisa de Envelhecimento de Software, estando ela inserida na subárea de confiabilidade de software.

Esse fenômeno pode ser causado por vários fatores, como mudanças no ambiente operacional, uso contínuo, falhas de hardware ou software, erros de codificação e falta de manutenção adequada. Existem diversos sintomas que podem indicar o envelhecimento de software, sendo os mais comuns deles o vazamento de memória e a degradação de desempenho (GROTTKE et al., 2008). Esses sintomas geralmente aparecem gradualmente e vão piorando ao longo do tempo, podendo levar à perda de desempenho, falhas, aumento no tempo de resposta, aumento do consumo de energia e outros problemas (HE, 2019). O vazamento de memória e a degradação de desempenho serão detalhados nas próximas subseções.

Vale destacar que, conforme um software envelhece, ele pode se tornar menos eficiente e confiável, especialmente para aqueles que dependem dele como um ativo crítico. Por isso, examinar os sintomas do envelhecimento do software é fundamental para identificar áreas problemáticas do sistema, permitindo a aplicação de medidas proativas para corrigir esses problemas. Dessa forma, as organizações podem garantir que o software continue a funcionar corretamente e evitar possíveis prejuízos decorrentes de falhas ou mau desempenho (COTRONEO et al., 2014).

É crucial que os desenvolvedores estejam cientes desses sintomas e ajam para

prevenir ou mitigar esses problemas antes que eles afetem negativamente o desempenho e a segurança do software. Com uma abordagem proativa, os desenvolvedores podem evitar que o software envelhecido cause prejuízos financeiros e de reputação, além de garantir que ele continue atendendo às necessidades dos usuários de forma eficiente e segura. Para obter informações adicionais sobre a terminologia básica relacionada ao envelhecimento de software, o leitor pode fazer referência a (TRIVEDI et al., 2010).

2.2.1 Vazamento de Memória

O ato de analisar a memória principal para identificar suspeitas de envelhecimento de software é algo muito comum na literatura, como por exemplo em (GROTTKE et al., 2006), (MATIAS et al., 2009) e (ANDRADE et al., 2021a). Isso porque o vazamento de memória é um dos principais sintomas do envelhecimento de software (GROTTKE et al., 2008). Esse problema acontece quando um programa não libera corretamente a memória que foi alocada durante sua execução, fazendo com que o sistema operacional acredite que a memória ainda está em uso, mesmo que o programa já tenha encerrado (MACÊDO et al., 2010). Com o passar do tempo, esses vazamentos de memória podem se acumular e consumir cada vez mais recursos do sistema, comprometendo a estabilidade e o desempenho do software. Isso pode resultar em falhas, travamentos e outras consequências negativas que afetam a experiência do usuário (VALENTIM et al., 2016).

O vazamento de memória pode ocorrer por diversos motivos, como a má gestão de ponteiros, falha na liberação de memória alocada dinamicamente ou problemas com o ciclo de vida dos objetos. Embora o vazamento de memória possa ser corrigido com uma boa gestão de memória, a solução pode se tornar mais difícil à medida que o software envelhece. Isso ocorre porque o software pode ter sido projetado sem considerar a possibilidade de vazamento de memória ou pode ter sido criado com uma linguagem de programação que não fornece suporte adequado para o gerenciamento de memória. Para evitar o problema de vazamento de memória, testes e monitoramentos devem ser feitos regularmente (HAUSWIRTH; CHILIMBI, 2004). É importante garantir que todas as alocações de memória sejam liberadas corretamente após o término da execução de uma tarefa. O uso de ferramentas de diagnóstico e depuração também pode ajudar a identificar e corrigir vazamentos de memória em tempo hábil (XIE; AIKEN, 2005).

No entanto, mesmo com esses cuidados, é importante lembrar que o envelhecimento de software é um fenômeno que pode ocorrer ao longo do tempo (CASTELLI et al., 2001). Além disso, é fundamental que testes e monitoramentos regulares sejam realizados para identificar e corrigir quaisquer vazamentos de memória (GARG et al., 1998). Dessa forma, é possível garantir que o software permaneça estável e ofereça uma experiência positiva ao usuário, mesmo após anos de uso.

2.2.2 Degradação de Desempenho

Conforme o software envelhece, uma interação complexa de fatores começa a surgir, contribuindo para a deterioração do desempenho (HUANG et al., 1995). A degradação de desempenho em sistemas de computadores refere-se à diminuição gradual ou abrupta da eficiência, velocidade ou capacidade de processamento de um sistema, resultando em uma execução mais lenta ou menos eficaz das tarefas e operações que normalmente seriam realizadas de forma rápida e eficiente. Isso pode ser causado por vários fatores, incluindo o fenômeno de envelhecimento de software, saturação de recursos, fragmentação de software ou sistemas mal otimizados, resultando em uma experiência do usuário menos satisfatória e um impacto negativo nas operações computacionais.

Os softwares podem sofrer degradação ou perda de desempenho ao longo do tempo devido a uma série de fatores (YIGITBASI et al., 2010). Entre os principais motivos, temos o uso intensivo desses softwares ou sua obsolescência, onde a falta de compatibilidade com atualizações de sistemas operacionais e hardware pode resultar em conflitos e incompatibilidades (JANG et al., 2017). Além disso, a acumulação de fragmentação de dados, o crescimento do volume de dados, a saturação de recursos do sistema, as mudanças nos requisitos e a falta de manutenção regular são elementos que podem contribuir para a degradação do desempenho. Em SGBDs, as constantes operações podem resultar em acúmulos na memória RAM, causando aumento no tempo de resposta. O prejuízo para os usuários de softwares afetados pela degradação de desempenho pode ser significativo. Em primeiro lugar, há uma redução na eficiência e na velocidade de execução das tarefas, o que resulta em atrasos e perda de produtividade. Operações que eram anteriormente rápidas e fluidas podem se tornar lentas e frustrantes. Além disso, a experiência do usuário é prejudicada, pois os aplicativos podem travar ou responder de

maneira imprevisível, causando desconforto e insatisfação.

Em resumo, a complexa interação de fatores que contribui para a deterioração do desempenho dos softwares à medida que envelhecem destaca-se como um fenômeno de ampla relevância. Esse declínio na eficiência, velocidade e capacidade de processamento, que afeta tanto a experiência do usuário quanto as operações computacionais, sublinha a necessidade de ações proativas. Diante dessa realidade, incluindo o envelhecimento do software, a saturação de recursos e a otimização deficitária, é fundamental adotar abordagens de manutenção, atualização e otimização contínuas para atenuar as consequências adversas e garantir a funcionalidade efetiva e duradoura de sistemas computacionais críticos (ZHENG et al., 2014).

3 Plano Experimental

Neste capítulo são apresentados detalhes sobre o estudo realizado, incluindo desde a construção do experimento, os ambientes utilizados, bem como os procedimentos adotados para coletar e analisar os dados. Primeiro, é detalhado o ambiente de teste adotado, onde são descritos todos os recursos de hardware e software utilizados. Depois, é apresentado como o experimento foi executado, descrevendo os parâmetros usados nos testes e o *schema* de banco de dados adotado. Por fim, são explicados os procedimentos para a coleta dos dados e abordados os conceitos necessários para o entendimento das análises realizadas no trabalho.

3.1 Ambiente de Testes

Na montagem do ambiente foi utilizada uma estrutura experimental com dois computadores, um funcionando como cliente e o outro como servidor de banco de dados. Nessa configuração, o cliente envia consultas SQL ao servidor, que responde prontamente às solicitações. A conexão entre as duas máquinas foi estabelecida por meio de uma rede local cabeada (*Fast Ethernet*). Durante os testes, somente as duas máquinas foram mantidas conectadas para garantir um tráfego de rede isolado e evitar interferência de pacotes de outros hosts. Esse procedimento foi adotado para obter resultados mais precisos e sem ruído em relação ao comportamento do servidor de banco de dados sob as cargas de trabalho utilizadas.

Para conectar os dois computadores, utilizamos um roteador de rede convencional da fabricante Link1One, modelo L1-RW131. Os SGBDs utilizados nos experimentos foram o PostgreSQL 14.4, o MySQL 8.0.31, o MariaDB 10.10 e o SQL Server Developer 2019. Tanto o cliente quanto o servidor executaram o sistema operacional GNU/Linux Ubuntu na versão 20.04 de 64 bits. A Figura 1 ilustra a estrutura utilizada nos experimentos, enquanto que a Tabela 1 apresenta as configurações de hardware dos computadores utilizados.

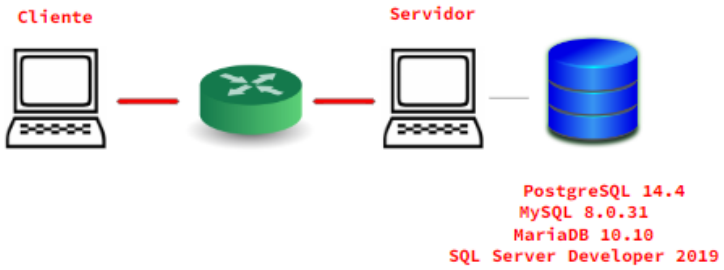

Figura 1 – Estrutura experimental utilizada.

Tabela 1 – Configuração de hardware dos computadores utilizados.

Computador	Hardware
Cliente	Processador Intel Core i7-4790 3,6 GHz, HD de 1 TB, Memória RAM de 8 GB
Servidor	Processador AMD A10 pro 7800b r7, HD de 250 GB, Memória RAM de 7 GB

3.2 Execução dos Experimentos

Por um período de 48 horas de modo ininterrupto, para cada uma das cargas de trabalho adotadas, foram executados os experimentos que consistiam no envio de consultas SQL pela máquina cliente à máquina designada como servidor de banco de dados. O mesmo experimento foi executado para todos os ambientes com os SGBDs utilizados neste trabalho. Cada consulta realizada pelo cliente retornou um total de quatro mil tuplas com o uso do comando SQL a seguir:

Código-fonte 3.1 – Comando SQL.

```
1 SELECT C.ID_cliente, C.Nome_Cliente,
2 C.Sobrenome_cliente, Pr.nome_produto,
3 Pr.preco_produto, P.qtde FROM
4 Clientes AS C JOIN Pedidos AS P
5 ON C.ID_cliente = P.ID_cliente
6 JOIN Produtos AS Pr ON
7 Pr.ID_produto = P.ID_produto;
```

Os parâmetros adotados, conforme descritos na Tabela 2, foram escolhidos no

intuito de acelerar o aparecimento de potenciais sintomas de envelhecimento de software no ambiente do servidor (COUTO et al., 2022). É importante ressaltar que o servidor passou por um processo de reinicialização após cada carga de trabalho. Assim, após executar a carga leve por 48 horas, o servidor foi reiniciado antes de seguir para a carga média e assim por diante. Essa reinicialização do sistema operacional garantiu a normalização do uso dos recursos de hardware e software do servidor, permitindo um novo teste com uma nova carga sem acúmulo de recursos computacionais anteriores e com a limpeza das áreas de cache. Também é importante destacar que somente um SGBD funcionou durante a realização de cada experimento. Por exemplo, foram realizados os teste com o PostgreSQL. Após a conclusão desses testes, o software foi desinstalado do servidor e o MySQL foi instalado para a realização de novos testes com o novo banco de dados. Foi obedecida essa regra para todos os SGBDs utilizados no experimento para eliminar possibilidade dos recursos computacionais serem consumidos por outro SGBD que não estivesse sob os efeitos das cargas adotadas.

Tabela 2 – Parâmetros usados nos testes - Consultas retornando 4000 tuplas.

Carga	**Intervalo entre as requisições (seg)**	**Número de usuários simultâneos**
Baixa	1	5
Média	0,5	50
Alta	0,2	100

Na realização dos experimentos, foi utilizado o *schema* de banco de dados apresentado na Figura 2. Esse *schema* é composto por três tabelas, cada uma contendo 4000 tuplas de registros inseridos. As tabelas estão relacionadas através das chaves estrangeiras "ID_cliente"e "ID_produto", presentes na tabela "Pedidos". Nenhuma estrutura de índice foi criada com o intuito de otimizar as consultas realizadas. Note que o objetivo principal deste estudo é detectar sinais de envelhecimento de software nos ambientes do servidor, considerando os bancos de dados relacionais utilizados. Além disso, comparar os resultados obtidos dos diferentes bancos de dados para verificar se houve alguma diferença significativa nas suspeitas de envelhecimento de software.

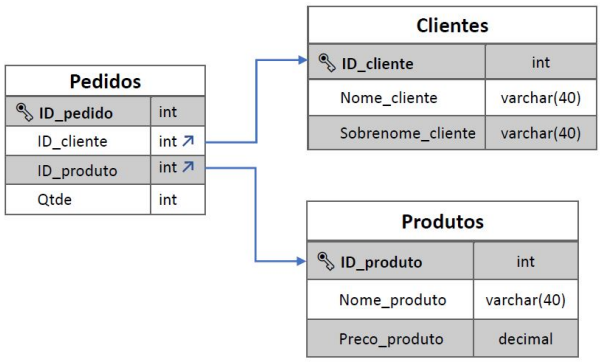

Figura 2 – Schema de banco de dados adotado nos experimentos.

O Apache JMeter versão 5.4.1 (JMETER™, 2023) foi utilizado para automatizar e manter o envio constante de requisições aos bancos de dados e para coletar os dados de tempo de resposta do servidor. O JMeter é um software de teste de desempenho de código aberto desenvolvido pela Apache Software Foundation. Ele é usado para testar a carga e o desempenho de aplicativos da web, bancos de dados, serviços de rede, servidores FTP (*File Transfer Protocol*) e outros tipos de serviços de rede. Além disso, permite que os usuários criem e executem testes de carga para medir o desempenho de um sistema, aplicativo ou serviço sob uma carga simulada. Os testes de carga podem ser realizados com um número específico de usuários virtuais ou com uma carga variável, simulando condições de pico ou picos de tráfego. Os resultados dos testes podem ser usados para otimizar o desempenho do sistema ou para determinar os limites de capacidade.

3.3 Coleta dos Dados e Análises

Nesta seção, são apresentados como os dados foram coletados e as análises realizadas.

3.3.1 Coleta de Dados

A coleta das informações necessárias para realização deste trabalho ocorreu por meio de *scripts* e também com o auxílio do JMeter. Na coleta de informações sobre o uso dos recursos do ambiente, foi criado um *Shell script* que capturou os dados a cada cinco segundos, por meio do comando *free*. O comando *free* é uma ferramenta comumente utilizada em sistemas Linux para exibir informações sobre o uso da memória do sistema. Ele fornece uma visão geral do estado atual da memória física e virtual, além de informações

sobre o uso do swap.

Já para a análise dos processos, foi desenvolvido um *script* em Python que coletou, a cada uma hora, as informações sobre o uso de memória pelos processos existentes no sistema operacional da máquina designada como servidor, o comando *smem* (SELENIC, 2023) foi empregado nesta tarefa. Os dados coletados abrangeram tanto os processos nativos do Ubuntu, quanto os processos relacionados à execução dos SGBDs utilizados e demais processos. O comando *smem* é uma ferramenta útil em sistemas Linux para visualizar e coletar informações detalhadas sobre o uso de memória por processos individuais (MACKALL, 2008-2009). Ele fornece estatísticas de uso de memória física, memória compartilhada, memória virtual e outras métricas relacionadas. Ao executar o comando *smem* sem argumentos, ele exibe um resumo geral do uso de memória em um sistema, incluindo informações sobre a memória física total, memória usada, compartilhada, em cache, swap, entre outras.

No entanto, o recurso mais poderoso do comando *smem* é sua capacidade de fornecer informações detalhadas sobre processos individuais. Pode-se usar opções como *-p* ou *–process* seguidas pelos identificadores de processo (PIDs) para obter informações sobre processos específicos. Por exemplo, ao executar o comando "*smem* -p 1234", é possível obter informações detalhadas sobre o processo com o PID 1234. Isso inclui o uso de memória física, memória compartilhada, memória virtual, entre outras. Essas informações podem ser úteis para identificar quais processos estão usando quantidades significativas de memória e podem ajudar na solução de problemas de desempenho ou na otimização do uso de memória. O comando *smem* também oferece várias outras opções, como a capacidade de classificar a saída com base em diferentes critérios, filtrar processos com base em determinados critérios e formatar a saída em vários estilos.

O *smem* também pode ser usado para detectar vazamentos de memória em um sistema e determinar a utilização de memória por diferentes usuários ou grupos. Algumas das métricas de uso de memória que são fornecidas pelo *smem* são: USS (tamanho do conjunto único), PSS (tamanho do conjunto proporcional) e RSS (tamanho do conjunto residente). Porém, neste trabalho, foram utilizados os valores de RSS e USS para realização das análises dos processos. O RSS é um indicador frequentemente utilizado em estudos de envelhecimento de software e é uma métrica que representa a quantidade de memória física atualmente alocada a um processo específico em um sistema. O RSS engloba tanto

a memória privada exclusiva do processo, quanto a memória compartilhada com outros processos. Já o USS, diferentemente do RSS, que inclui alocações de memória ou páginas de memória compartilhadas entre diferentes processos, informa o uso de memória utilizada unicamente por um determinado processo. Deste modo, advocamos que o USS é mais apropriada para detectar um possível vazamento de memória nos processos dos SGBDs utilizados ao longo do tempo. O RSS foi utilizado na análise que mostrou os processos que mais consumiram memória. Além disso, os dados de tempo de resposta, taxa erro e vazão foram coletados com o auxílio do JMeter. É importante destacar que coletamos diversas métricas. No entanto, neste trabalho, nosso foco está na investigação da presença de degradação de memória e de desempenho como indicadores do envelhecimento de software, uma vez que os demais indicadores não apresentaram evidências claras de envelhecimento.

3.3.2 Análises

Neste trabalho, empregamos o teste de Mann-Kendall e o estimador da inclinação de Sen's Slope, ambos métodos estatísticos, com o propósito de detectar indícios de envelhecimento de software em ambientes específicos. Abordaremos a descrição de ambos os testes a seguir. No entanto, antes disso, vamos apresentar os conceitos fundamentais relacionados aos testes de tendência, que servem como alicerce para essas análises.

3.3.2.1 Teste de Tendência

O teste de tendência é uma ferramenta estatística utilizada para avaliar se há uma tendência ou padrão linear em um conjunto de dados. Esse teste é frequentemente aplicado em análises de séries temporais e em estudos que buscam identificar uma relação de causa e efeito entre duas variáveis. Existem diferentes tipos de teste de tendência, cada um adequado para diferentes tipos de dados e hipóteses de pesquisa. Um dos testes mais comuns é o teste de Mann-Kendall, que compara as diferenças entre pares de observações em uma série temporal e avalia se há uma tendência crescente ou decrescente nos valores ao longo do tempo (MCLEOD, 2005).

Para realizar um teste de tendência, é necessário definir uma hipótese nula, que geralmente assume que não há tendência ou padrão linear nos dados. Em seguida, é

aplicado um teste estatístico que compara as diferenças entre as observações e calcula um valor de estatística de teste, que é comparado com um valor crítico para determinar se a hipótese nula deve ser rejeitada ou não (SIEGEL; JR, 1975). É importante lembrar que o teste de tendência não é uma prova definitiva de uma relação de causa e efeito entre duas variáveis, mas sim uma ferramenta estatística para avaliar a presença de um padrão linear em um conjunto de dados. É essencial que outros fatores sejam levados em consideração ao interpretar os resultados de um teste de tendência, como a qualidade dos dados, a presença de outras variáveis relevantes e a consistência dos resultados com outras pesquisas. Um teste de tendência é uma ferramenta estatística útil para avaliar se há um padrão linear em um conjunto de dados, mas é importante considerar outros fatores ao interpretar seus resultados. A escolha do teste adequado e a interpretação cuidadosa dos resultados são fundamentais para garantir que as conclusões da análise sejam confiáveis e precisas (NETER et al., 1996).

3.3.2.2 Teste de Mann-Kendall

O teste de Mann-Kendall é uma técnica estatística amplamente utilizada para analisar tendências em dados temporais (MANN, 1945) (KENDALL, 1948). Ele foi originalmente proposto por Mann e Kendall em 1945 e é amplamente utilizado em diversas áreas, como hidrologia, meteorologia, engenharia civil, economia, entre outras. O teste de Mann-Kendall é um teste não paramétrico, o que significa que não faz suposições sobre a distribuição dos dados. Ele avalia se há uma tendência significativa nos dados ao longo do tempo, mesmo que os dados não sejam distribuídos normalmente. A tendência pode ser crescente, decrescente ou não significativa. Uma das principais vantagens do teste de Mann-Kendall é sua robustez em relação a valores extremos e dados faltantes. Ele é capaz de detectar tendências em dados que apresentam uma variação ampla ou em séries temporais com lacunas de dados, o que pode ser um problema em outros testes estatísticos paramétricos (HIPEL; MCLEOD, 1994).

Ao realizar o teste de Mann-Kendall, duas hipóteses são consideradas: a hipótese nula e a hipótese alternativa. A hipótese nula afirma que não há tendência nos dados, enquanto a hipótese alternativa indica se há uma tendência de aumento ou diminuição nos dados. O valor de p (*p-value*) é um importante parâmetro estatístico que é utilizado para

avaliar a significância da hipótese alternativa. Se o valor de *p-value* for menor que 0,05, é geralmente aceito que há uma tendência nos dados, ou seja, a hipótese alternativa é aceita e a hipótese nula é rejeitada. Isso significa que os dados apresentam uma tendência de aumento ou diminuição ao longo do tempo. Outro parâmetro importante do teste de Mann-Kendall é o valor de S, que indica a magnitude e a direção da tendência observada nos dados. Se o valor de S for positivo, há uma tendência de aumento nos dados, enquanto se o valor de S for negativo, há uma tendência de diminuição nos dados.

Foi utilizado o teste de Mann-Kendall nesta pesquisa para analisar a tendência dos dados de uso de memória RAM e tempo de resposta. O objetivo específico da análise foi verificar se o envio constante de requisições de leitura aos ambientes de banco de dados utilizados poderia causar alguma degradação devido às cargas de trabalho aplicadas. Essa degradação seria detectada pelo aumento do consumo de memória e do tempo de resposta ao longo do tempo de teste. O teste foi realizado utilizando a linguagem R e o pacote "trend"(CRAN, 2023). O pacote "trend"do R é uma ferramenta estatística que permite a análise de tendências, fornecendo uma variedade de funções para implementar diferentes métodos de detecção de tendência, como o teste de Mann-Kendall, o teste de Sen´s slope, o teste de Cox-Stuart, entre outros.

3.3.2.3 Estimador Sen's Slope

O estimador Sen's Slope é uma técnica estatística utilizada para estimar a inclinação de uma reta em um conjunto de dados (SEN, 1968). Ele é frequentemente utilizado em conjunto com o Teste de Mann-Kendall para avaliar a tendência temporal de um conjunto de dados. Enquanto o teste de Mann-Kendall detecta em um conjunto de dados o aumento ou diminuição ao longo do tempo, o Sen's Slope permite estimar a magnitude dessa tendência. Para aplicar o Teste de Mann-Kendall em conjunto com o estimador Sen's Slope, primeiro é realizada a análise de tendência com o Teste de Mann-Kendall. Se for detectada uma tendência estatisticamente significativa na série temporal, o próximo passo é estimar a magnitude dessa tendência com o estimador Sen's Slope.

O cálculo do estimador Sen's slope é feito por meio da diferença de inclinação entre cada par de pontos de uma série temporal, considerando todas as inclinações possíveis. Com as diferenças de inclinação obtidas, calcula-se a mediana desses valores. O resultado

final é um único valor que representa a magnitude da inclinação da reta que melhor se ajusta aos pontos da série temporal, levando em conta a tendência temporal positiva ou negativa. O Estimador Sen's Slope é uma ferramenta não paramétrica, o que significa que não é necessário assumir uma distribuição específica dos dados para seu cálculo. Além disso, é uma medida robusta e eficiente para identificar tendências em séries temporais, mesmo na presença de valores extremos ou pontos discrepantes.

Neste trabalho, fizemos o uso do Estimador Sen's slope a um nível de confiança igual a 95% para determinar a magnitude da inclinação dos dados de uso de memória RAM e tempo de resposta em conjunto com o teste de Mann-Kendall. O teste foi realizado utilizando a linguagem R e o pacote "trend"(CRAN, 2023) e nos ajudou por meio do resultado, determinar qual ambiente foi mais ou menos afetado pelo envelhecimento de software em virtude das cargas de trabalho aplicadas.

4 Resultados

Neste capítulo, apresentaremos os resultados obtidos a partir da execução dos experimentos. Primeiramente, são exibidos os resultados dos experimentos realizados com o MySQL. Em seguida, são discutidos os resultados obtidos no ambiente com o MariaDB. Posteriormente, são apresentados os resultados dos experimentos executados no ambiente com o PostgreSQL. A seguir, são apresentados os resultados obtidos com os experimentos executados no ambiente do SQL Server. Por fim, realizaremos um comparativo entre os resultados obtidos com cada SGBD e apresentaremos as ameaças à validade deste trabalho.

Para os resultados e análises apresentados neste capítulo, é necessário esclarecer alguns pontos para um melhor entendimento do trabalho:

- No decorrer de todos os teste realizados, não ocorreu nenhuma ocorrência de erro. Isto é, todas as requisições enviadas ao servidor foram processadas;
- Os limites de taxa de transmissão da rede empregada não foram excedidos, uma vez que a máxima taxa de transferência alcançada durante os testes foi de 6948,55 KB/s;
- Nenhum dos experimentos resultou em utilização de memória superior à capacidade total disponível do sistema operacional. Assim, a memória Swap alocada permaneceu sem uso em todos os momentos, tornando desnecessária a consideração desse indicador de envelhecimento de software.;
- Para conduzir análises estatísticas nos dados de uso de memória, tornou-se essencial calcular a média de uso por hora devido à volumosa quantidade de informações, o que inviabilizaria a realização dos testes de outra forma; e
- A fim de facilitar a geração dos gráficos e a realização dos testes estatísticos, foi necessário calcular a média do tempo de resposta por hora, devido à grande quantidade de dados envolvidos.

4.1 MySQL

Nesta seção, serão apresentados os resultados obtidos nos testes realizados com o MySQL. Primeiramente, será apresentada a análise do consumo de memória RAM. Em seguida, será abordada a análise do tempo de resposta para verificar se houve alguma perda de desempenho do SGBD. Além disso, realizaremos uma análise dos processos,

identificando os principais responsáveis pelo consumo de memória, bem como os possíveis processos relacionados ao envelhecimento do software no ambiente.

4.1.1 Análise do Consumo de Memória

Foi realizada uma análise do uso de memória RAM usando testes estatísticos de Mann-Kendall e Sen's slope, com o objetivo de investigar indícios de envelhecimento de software no ambiente do MySQL sob diferentes cargas de trabalho. A Figura 3 ilustra o comportamento do consumo de memória RAM durante os testes com a carga baixa. Através da figura, é perceptível um uso inicial baixo da memória, que aumenta gradualmente com o passar do tempo. Durante a maior parte do período, observamos que o uso de memória se manteve abaixo de 17,5%. Além disso, foram identificados picos pontuais, sendo um deles com um uso de memória superior a 19%.

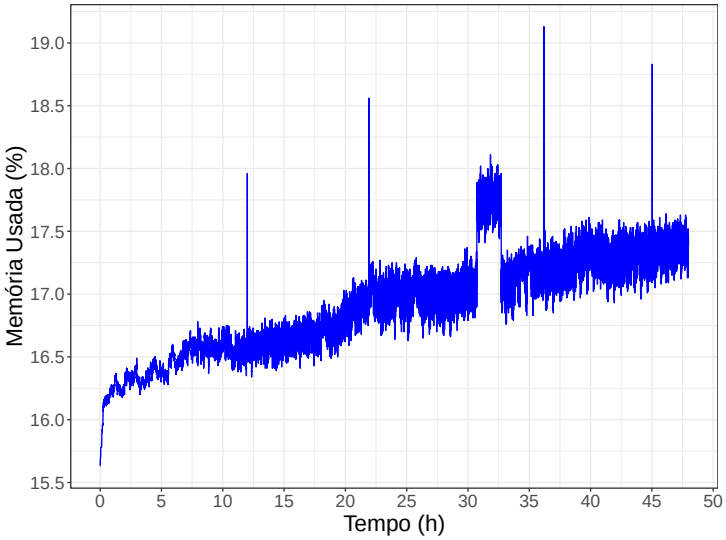

Figura 3 – MySQL - Consumo de memória RAM - Carga Baixa.

A Figura 4 apresenta o consumo de memória ao longo do teste com a carga média no servidor MySQL. Observa-se um patamar mais elevado de utilização de memória em comparação com o teste de carga baixa (Figura 3). É evidente que a maior parte do tempo o uso da memória esteve acima de 17%, o que pode ser considerado um nível de utilização relativamente baixo. Adicionalmente, percebe-se uma tendência de crescimento no uso da memória ao longo do tempo; no entanto, é importante realizar testes estatísticos para confirmar essa suspeita.

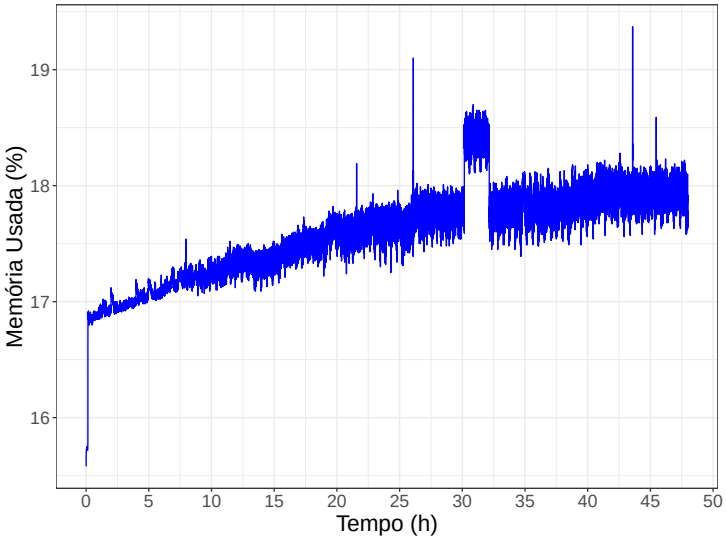

Figura 4 – MySQL - Consumo de memória RAM - Carga Média.

Na Figura 5, é apresentado o consumo de memória durante os testes com carga alta. Assim como nos gráficos das Figuras 3 e 4, também é possível observar um aumento percentual no uso de memória RAM ao longo do tempo. Também é notável que existem picos de uso de memória mais elevados e um nível geral de utilização mais alto quando a carga é mais elevada.

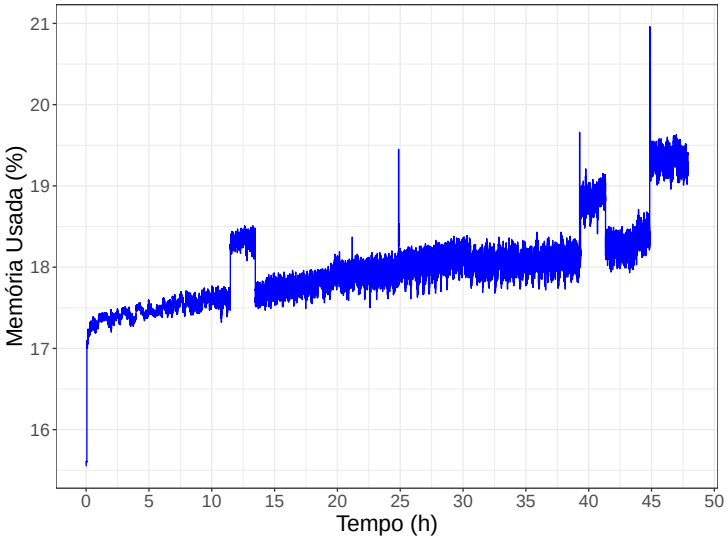

Figura 5 – MySQL - Consumo de memória RAM - Carga Alta.

Com base nos dados de uso de memória do servidor usando o MySQL, foram realizados testes estatísticos de Mann-Kendall e do estimador Sen's slope para confirmar suspeitas de degradação no ambiente. A Tabela 3 apresenta os resultados desses testes, confirmando estatisticamente a suspeita de envelhecimento do software devido à degradação

do uso da memória ao longo do tempo. Em todas as cargas utilizadas, foi observada uma tendência de crescimento nos dados. Os valores de *p-value* obtidos foram inferiores a 0,05 e os valores de S foram positivos em todos os casos. Além disso, os valores do Sen's slope indicam que a magnitude do crescimento é maior para a carga alta. No entanto, é importante observar que a carga baixa apresentou um crescimento ligeiramente superior ao da carga média, mesmo com a mesma configuração de hardware e software. Isso pode sugerir que essas cargas de trabalho podem não afetar diretamente a magnitude de crescimento dos dados.

Tabela 3 – MySQL - Teste de Mann-Kendall e Sen's slope para o uso de memória RAM.

SGBD	**Carga**	***p-value***	**S**	**Sen's slope (%/h)**	**Tendência**
MySQL	Baixa	2,2e-16	9,88e+02	0,024	Crescimento
	Média	2,2e-16	9,94e+02	0,023	Crescimento
	Alta	2,2e-16	9,4e+02	0,026	Crescimento

4.1.2 Análise do Tempo de Resposta

Com o objetivo de investigar uma possível perda de desempenho decorrente das cargas utilizadas e encontrar mais evidências de envelhecimento do software no ambiente, foi conduzida uma análise do tempo de resposta. Essa análise teve como propósito verificar se o uso contínuo do sistema de gerenciamento de banco de dados resultaria em uma deterioração do desempenho, evidenciada pelo aumento do tempo de resposta ao longo do período em que as cargas foram aplicadas.

Na Figura 6, é mostrado o comportamento do tempo de resposta ao longo da execução da carga baixa para o ambiente com o SGBD MySQL. Nesse teste, o tempo de resposta médio ficou em 47 ms e é possível notar muitas oscilações no decorrer do tempo. O gráfico apresentado não fornece uma indicação clara sobre se houve uma tendência de aumento ou diminuição nos dados ao longo do tempo. Deste modo, é fundamental a realização dos testes estatísticos para confirmar as suspeitas.

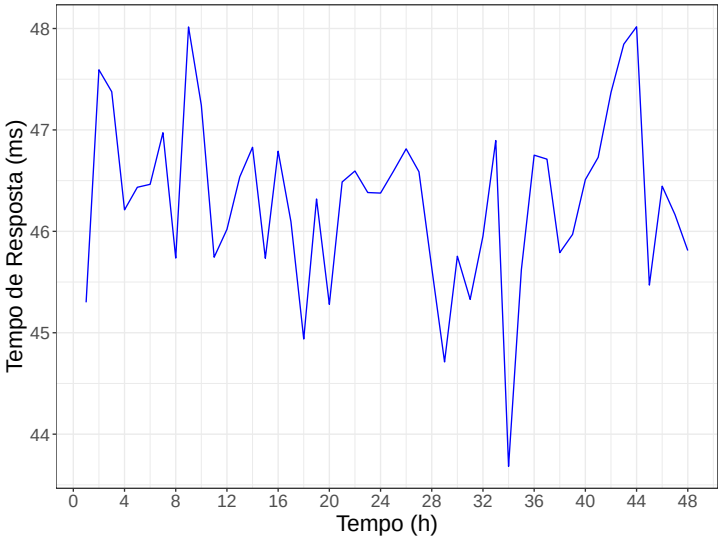

Figura 6 – MySQL - Tempo de Resposta - Carga Baixa.

A Figura 7 apresenta a evolução do tempo de resposta durante a execução dos experimentos com a carga média utilizando o MySQL. É possível observar um aumento no tempo de resposta ao longo do tempo. O tempo de resposta inicial, que variou entre 200 e 300 ms, aumentou para um valor superior a 500 ms ao final do teste. Isso indica que o sistema degradou com o tempo. Além disso, o tempo médio obtido com o uso dessa carga foi de 402 ms, o que representa um aumento significativo em comparação com os testes de carga baixa, onde o tempo de resposta médio foi de 47 ms.

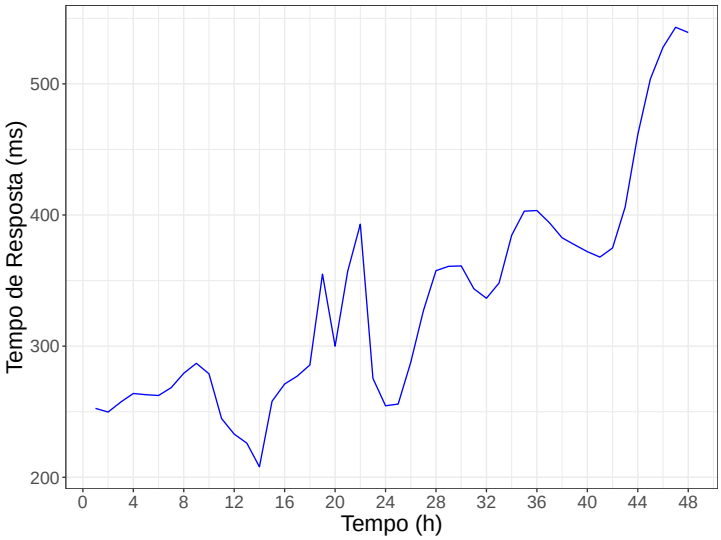

Figura 7 – MySQL - Tempo de Resposta - Carga Média.

O gráfico mostrado na Figura 8 ilustra os resultados do tempo de resposta para a carga alta. Verificou-se que os valores de tempo de resposta foram significativamente

maiores em comparação com os gráficos apresentados nas Figuras 6 e 7. No experimento em questão, obteve-se um tempo médio de resposta de 1623 ms. Além disso, é possível observar uma tendência de aumento nos dados ao longo do tempo. É interessante notar que o crescimento mais acentuado ocorreu perto das 16 horas de teste.

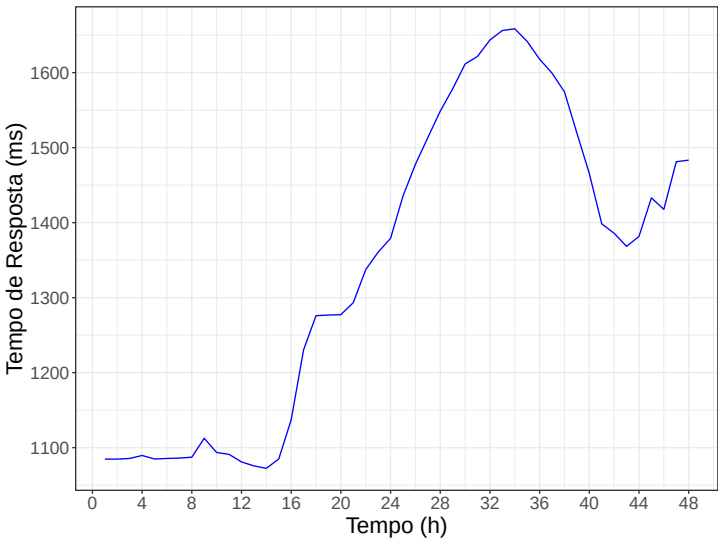

Figura 8 – MySQL - Tempo de Resposta - Carga Alta.

Como parte da análise, foram realizados o teste de Mann-Kendall e estimado o Sen's slope nos dados do tempo de resposta, a fim de confirmar estatisticamente as suspeitas de envelhecimento de software por degradação de desempenho. A Tabela 4 apresenta os resultados dos testes estatísticos realizados, revelando que, em cargas média e alta, o desempenho do MySQL apresentou perda. Por outro lado, mesmo com um valor de *p-value* maior que 0,05, indicando ausência de tendência estatística de crescimento ou decrescimento nos dados, observou-se que os valores de S e do Sen's slope foram negativos em carga baixa, sugerindo uma tendência de decrescimento nos dados. É importante ressaltar que essa tendência, embora não seja estatisticamente significativa de acordo com o teste de Mann-Kendall, indica uma possível queda no tempo de resposta do MySQL. Adicionalmente, vale destacar que as cargas média e alta têm impacto direto na degradação do ambiente, onde cargas maiores resultam em inclinações mais elevadas.

Tabela 4 – MySQL - Teste de Mann-Kendall e Sen's slope para o tempo de resposta.

SGBD	Carga	*p-value*	S	Sen's slope (ms/h)	Tendência
MySQL	Baixa	0,689	-4,6e+01	-0,0037	Decrescimento
	Média	5,65e-12	7,7e+02	4,66	Crescimento
	Alta	3,06e-09	6,68e+02	12,18	Crescimento

4.1.3 Análise dos Processos

Nesta seção, detalhamos as análises dos processos no ambiente em que o MySQL foi empregado. Inicialmente, analisaremos os cinco processos que mais consumiram memória RAM. Posteriormente, identificaremos aqueles que registraram os maiores aumentos no consumo de memória durante a execução dos experimentos. Por fim, investigaremos o consumo de memória do processo relacionado ao MySQL.

4.1.3.1 Processos que mais consumiram Memória

Nesta análise, conseguimos identificar quais processos estavam consumindo mais memória RAM no servidor. Além disso, pudemos avaliar se as requisições enviadas ao MySQL estavam contribuindo significativamente para o uso de memória. Para realizar essa análise, utilizamos o último arquivo coletado pelo *script* de monitoramento de processos, uma vez que nele encontramos uma fase em que o ambiente já está deteriorado devido à alta quantidade de requisições enviadas pelo cliente, próximo às 48 horas de teste para uma determinada carga.

Na Tabela 5, são descritos os cinco processos que mais consumiram memória em carga baixa. O gráfico da Figura 9 representa a quantidade de memória utilizada por esses cinco processos no experimento em que o MySQL foi utilizado como SGBD. Com essa carga, o processo responsável pela execução do MySQL utilizou mais memória do que os demais, seguido pelos processos nativos do Ubuntu: "gnome-shell", "Xorg vt1 -dis", "tracker-store"e "snapd", respectivamente.

Tabela 5 – MySQL - Descrição dos processos que mais consumiram memória - Carga Baixa.

PIDs	Descrição
904	**mysql** - Processo do MySQL.
1177	**gnome-shell** - Ambiente de desktop GNOME.
1109	**Xorg vt1 -dis** - Servidor de exibição padrão.
1655	**tracker-store** - Processo de indexação de arquivos.
637	**snapd** - Processo que permite instalação e execução de aplicativos empacotados no formato 'snap'.

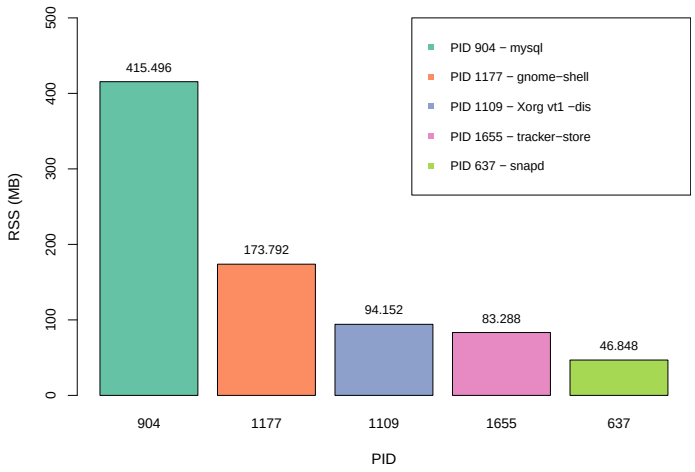

Figura 9 – MySQL - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Baixa.

Apesar de possuir PIDs diferentes, a descrição dos cinco processos que mais consumiram memória em carga média é apresentada na Tabela 5. A Figura 10 mostra o consumo desses processos. É possível observar que os mesmos processos que estavam entre os cinco principais consumidores de memória em uma carga baixa também aparecem nesse experimento. Além disso, o processo responsável pela execução do MySQL consumiu significativamente mais memória do que os demais, com um uso de memória superior a 400 MB.

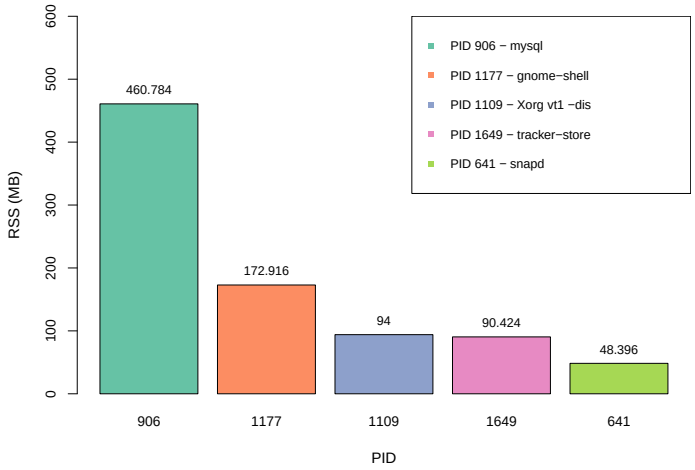

Figura 10 – MySQL - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Média.

Mesmo com PIDs diferentes, a descrição dos processos que tiveram o maior consumo de memória em carga alta é apresentada na Tabela 5. A Figura 11 ilustra graficamente o consumo de memória desses processos. Além disso, durante a carga alta, observou-se que o processo responsável pela execução do MySQL utilizou mais memória do que os demais. Também é importante mencionar que, ao compararmos o processo do MySQL em carga média e baixa, notamos um discreto aumento no consumo de memória, revelando assim o efeito da carga mais intensa sobre o SGBD utilizado.

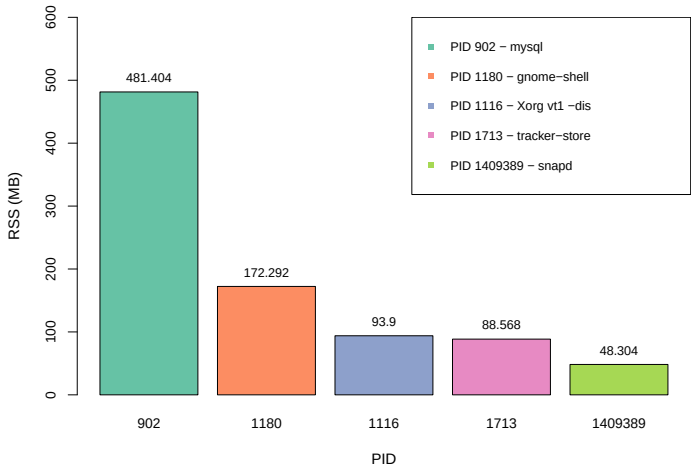

Figura 11 – MySQL - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Alta.

Após a análise realizada, fica evidente que as cargas de trabalho aplicadas resultaram no MySQL sendo o principal responsável pelo consumo de memória RAM, superando significativamente os demais processos em todas as situações. Além disso, observou-se consistentemente que os processos gnome-shell, Xorg vt1 -dis, tracker-store e snapd aparecem durante a execução do MySQL. Esses resultados sugerem que o SGBD pode desempenhar um papel significativo nos efeitos de envelhecimento de software no ambiente em questão.

4.1.3.2 Processos que mais cresceram juntamente com o SGBD

Nesta análise, ao contrário da análise realizada na Seção 4.1.3.1, foram identificados os processos que apresentaram o maior aumento no consumo de memória RAM durante os testes. Para calcular a porcentagem de crescimento, utilizamos os valores de USS do arquivo gerado pelo *script* de monitoramento de processos correspondente à primeira hora de teste, bem como o arquivo referente à última hora de teste. A análise abrange os quatro processos que registraram o maior aumento no consumo de memória, além do processo associado à execução do MySQL. O objetivo da análise foi identificar os possíveis processos responsáveis pelo envelhecimento do software no ambiente, devido ao acúmulo gerado nos mesmos.

A Tabela 6 apresenta o ranking dos processos com o maior crescimento no consumo de memória em carga baixa, enquanto a Tabela 7 fornece a descrição desses processos. O que chamou atenção nesse ranking é o percentual de crescimento do processo "tracker-store", atingindo 387%. Já ao analisarmos o processo do MySQL (PID 904), observamos um crescimento baixo de apenas 0,06%, colocando-o na 34ª posição nesse ranking. No entanto, podemos identificar um acúmulo no consumo de memória do SGBD, o que levanta suspeitas de um possível envelhecimento do software nesse componente.

Tabela 6 – MySQL - Processos que mais cresceram juntamente com o SGBD - Carga Baixa.

Ranking	**PID**	**Processo**	**Crescimento**
1º	1655	tracker-store	387%
2º	1406	tracker-miner	25%
3º	1663	tracker-extract	23%
4º	637	snapd	20%
34º	904	mysql	0,06%

Tabela 7 – MySQL - Descrição dos processos que mais cresceram juntamente com o SGBD - Carga Baixa

Processo	**Descrição**
tracker-store	Processo de indexação de arquivos
traker-miner	Processo responsável por indexar e pesquisar arquivos no sistema de arquivos.
tracker-extract	Processo que indexa e extrai metadados de arquivos no sistema de arquivos
snapd	Processo que permite instalação e execução de aplicativos empacotados no formato 'snap'
mysql	Processo do MySQL

Na Tabela 8, é apresentado o ranking dos processos que mais cresceram durante os experimentos com carga média. Apesar de possuírem PIDs diferentes, a descrição desses processos é apresentada na Tabela 7, com exceção do processo "gvfsd-metadata", que é responsável pela serialização de gravação dos metadados gvfs. Mais uma vez, o processo "tracker-store"ocupou a primeira posição do ranking, apresentando um crescimento maior (432%) em comparação com o experimento de carga baixa (387%). Além disso, um novo processo, o "gvfsd-metadata", também entrou no ranking. O processo do MySQL teve um acúmulo de apenas 0,05%, o que pode ser considerado baixo e bastante próximo ao valor de crescimento observado na carga baixa.

Tabela 8 – MySQL - Processos que mais cresceram juntamente com o SGBD - Carga Média.

Ranking	PID	Processo	Crescimento
1º	1649	tracker-store	432%
2º	641	snapd	25%
3º	1694	tracker-extract	21%
4º	1674	gvfsd-metadata	21%
27º	906	mysql	0,05%

Na Tabela 9, são apresentados os processos que apresentaram o maior crescimento em carga alta. Mesmo com PIDs diferentes, a descrição desses processos pode ser vista na Tabela 7, com exceção do processo "init splash", que é responsável pela inicialização do Ubuntu. Observa-se que o processo do MySQL (PID 902) registrou um aumento mais significativo de 0,77%, em comparação com os crescimentos de 0,06% e 0,05% durante as cargas baixa e média, respectivamente. Pode-se inferir que a intensificação da carga de trabalho pode ter contribuído para esse acúmulo maior, levando esse processo a ocupar a 27ª posição no ranking de maiores crescimentos em carga alta. Além disso, o processo "init slapsh", responsável pela inicialização do Ubuntu, passou a integrar essa lista, com um crescimento percentual de 162%.

Tabela 9 – MySQL - Processos que mais cresceram juntamente com o SGBD - Carga Alta.

Ranking	PID	Processo	Crescimento
1º	1713	tracker-store	416%
2º	1	init splash	162%
3º	1757	tracker-extract	22%
4º	1471	tracker-miner	20%
27º	902	mysql	0,77%

É importante observar que o processo "tracker-store"ocupou a primeira posição em todos os casos, sugerindo que esse processo desempenha um papel relevante na degradação dos ambientes. No entanto, os processos relacionados ao MySQL não podem ser negligenciados, pois também têm impacto na degradação dos ambiente adotado.

4.1.3.3 Análise dos Processos do MySQL

Como evidenciado pela análise anterior, os processos relacionados ao banco de dados podem ser um dos fatores responsáveis pela degradação dos ambientes do sistema adotado. Nesta análise, exploraremos mais detalhadamente a investigação desses processos.

A Figura 12 apresenta a evolução do processo responsável pela execução do MySQL em carga baixa. Através desse gráfico, é evidente o aumento no uso de memória pelo processo do SGBD. Conforme revelado na análise do crescimento dos processos (Seção 4.1.3.2), embora tenha ocorrido um crescimento ao longo do tempo, esse crescimento é considerado baixo, o que demonstra a eficiente capacidade do MySQL em gerenciar a memória.

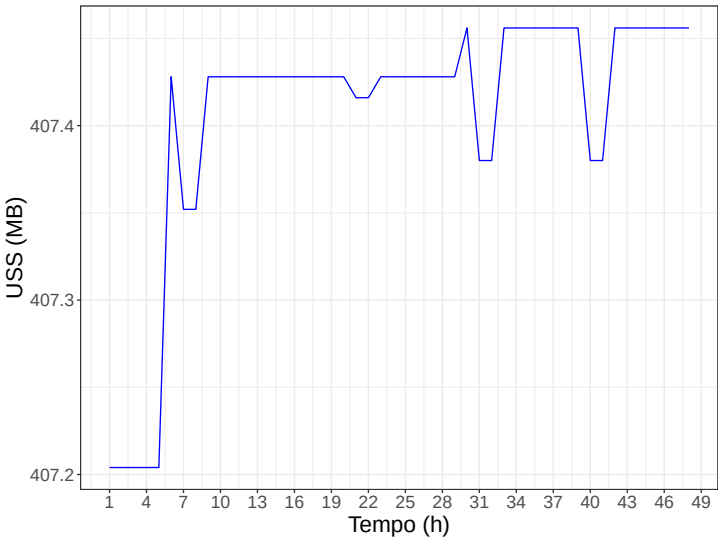

Figura 12 – Crescimento do processo do MySQL ao longo do tempo de teste - Carga Baixa.

Em relação à carga média, é possível observar um crescimento nos dados representados no gráfico da Figura 13. Assim como na carga baixa, nota-se um aumento relativamente baixo nos dados ao longo do tempo. Portanto, a realização de testes estatísticos se torna crucial para determinar a magnitude desse crescimento e identificar qual tendência está presente nos dados obtidos. Além disso, esses testes nos permitem obter uma análise mais aprofundada sobre os resultados.

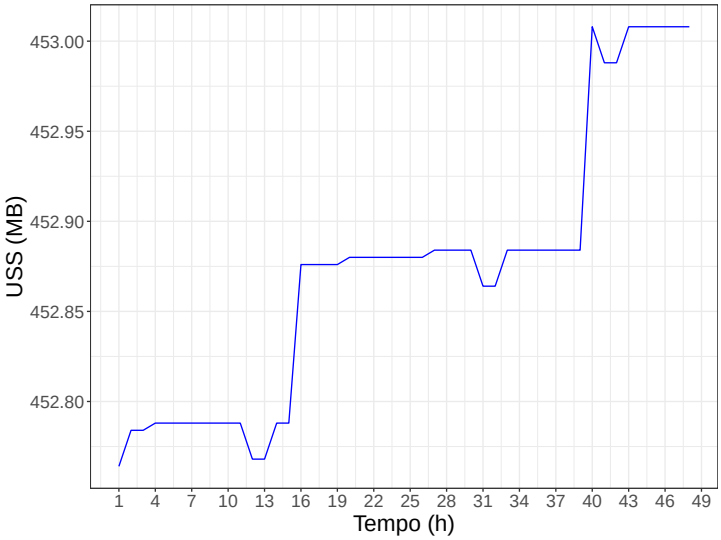

Figura 13 – Crescimento do processo do MySQL ao longo do tempo de teste - Carga Média.

A Figura 14 ilustra a evolução do desempenho do MySQL em condições de carga alta. Esses dados revelam que o crescimento permaneceu estável até as 37 horas de teste, mas a partir desse ponto houve um aumento abrupto que se manteve até o final do teste. Ao observar a figura, é possível notar um nível mais elevado de utilização de memória comparado com a carga baixa e média, o que sugere uma possível tendência de crescimento nos dados. Essa tendência pode ser confirmada por meio da aplicação de testes estatísticos, que fornecerão uma análise mais precisa dos resultados obtidos.

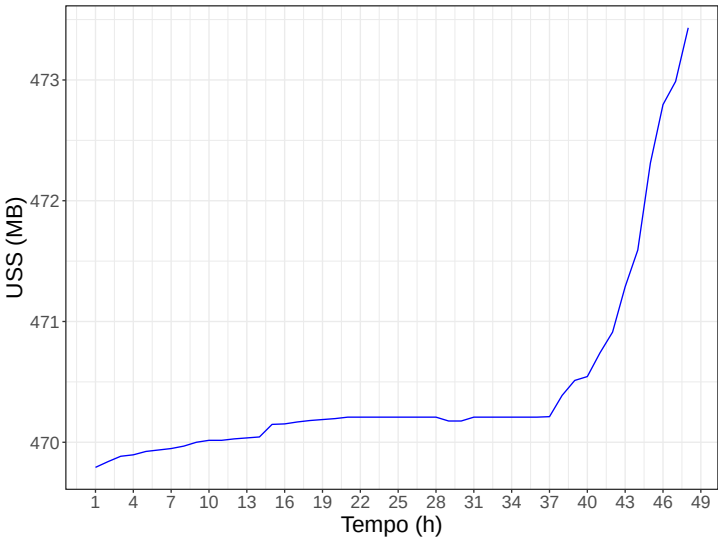

Figura 14 – Crescimento do processo do MySQL ao longo do tempo de teste - Carga Alta.

A Tabela 10 apresenta os resultados do teste de Mann-Kendall e do estimador

Sen's slope para analisar o crescimento do consumo de memória ao longo do tempo nos experimentos com o processo do MySQL. O objetivo desses testes foi verificar indicativos de envelhecimento de software no SGBD considerando diferentes cargas de trabalho. Os valores obtidos para o *p-value* foram inferiores a 0,05 em todos os casos, e os valores de S foram positivos, o que indicam uma tendência de crescimento nos dados, corroborando a suspeita de envelhecimento do software no SGBD. Além disso, o estimador Sen's slope nos fornece informações sobre a magnitude de inclinação nesses dados, destacando que as cargas de trabalho têm impacto direto na degradação do ambiente através do MySQL, onde cargas maiores resultam em degradações mais elevadas.

Tabela 10 – MySQL - Teste de Mann-Kendall e Sen's slope para o processo do SGBD.

SGBD	**Carga**	***p-value***	**S**	**Sen's slope (MB/h)**	**Tendência**
MySQL	Baixa	2,71e-07	5,47e+02	0,0011	Crescimento
	Média	3,72e-15	8,72e+02	0,0046	Crescimento
	Alta	2,2e-16	9,91e+02	0,017	Crescimento

4.2 MariaDB

Nesta seção, serão apresentados os resultados obtidos no experimento em que o MariaDB foi utilizado como SGBD. Primeiramente, será apresentada a análise do consumo de memória RAM. Em seguida, será abordada a análise do tempo de resposta para verificar se houve alguma perda de desempenho do SGBD. Além disso, realizaremos uma análise dos processos, identificando os principais responsáveis pelo consumo de memória, bem como os possíveis processos relacionados ao envelhecimento do software no ambiente.

4.2.1 Análise do Consumo de Memória

Uma análise de uso de memória RAM foi conduzida utilizando os testes estatísticos de Mann-Kendall e Sen's slope, a fim de investigar as suspeitas de envelhecimento de software em um ambiente onde o MariaDB foi utilizado como SGBD submetido a cargas de trabalho. A Figura 15 ilustra o consumo de memória RAM no servidor em carga baixa. O uso de memória se manteve relativamente baixo durante ao longo do tempo, porém é

possível observar um crescimento nos dados no decorrer do teste. Picos superiores a 14% de uso de memória também são observados, porém o uso se manteve abaixo de 13% na maior parte do tempo.

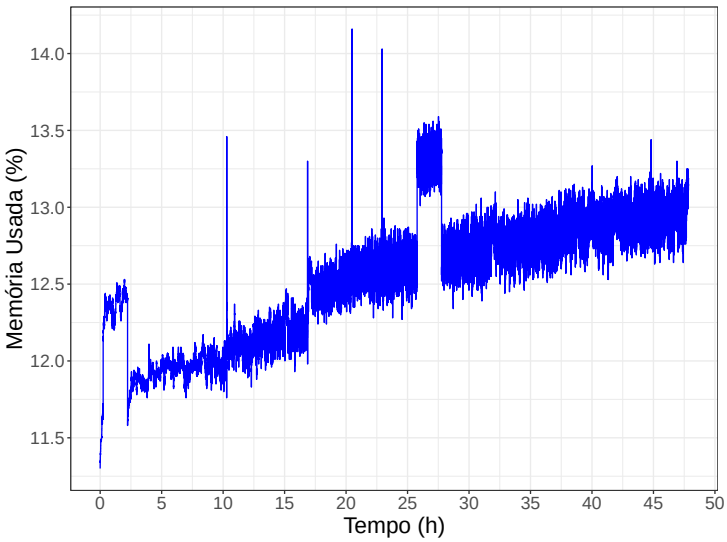

Figura 15 – MariaDB - Consumo de memória RAM - Carga Baixa.

Em relação à carga média, observou-se que o comportamento da memória ao longo do tempo permaneceu baixo, porém crescente, conforme demonstrado na Figura 16. Comparando com o uso de memória em carga baixa (Figura 15), percebemos que o patamar de utilização chegou em 13% de uso de memória mais rapidamente para esta carga. Embora uma tendência de crescimento seja visível nesses dados, é essencial realizar testes estatísticos para confirmar se houve degradação devido ao efeito da carga ao longo do tempo.

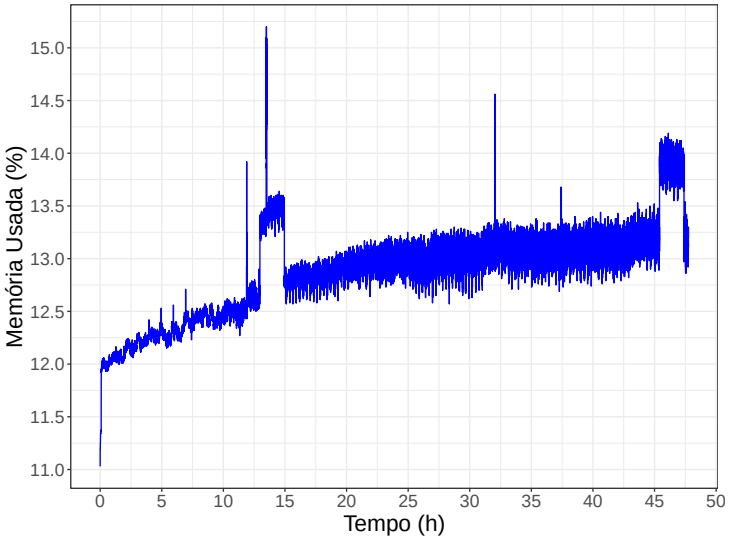

Figura 16 – MariaDB - Consumo de memória RAM - Carga Média.

Na Figura 17, é apresentado o gráfico do consumo de memória sob carga alta. Como podemos observar, ao longo do tempo de teste, o consumo de memória cresceu com o uso dessa carga. Também é possível notar picos de uso superiores a 14% e um patamar de uso um pouco mais elevado em comparação aos experimentos em carga baixa e média. Mesmo com a carga mais intensa, o consumo de memória se manteve relativamente baixo, com um crescimento constante na maior parte do tempo.

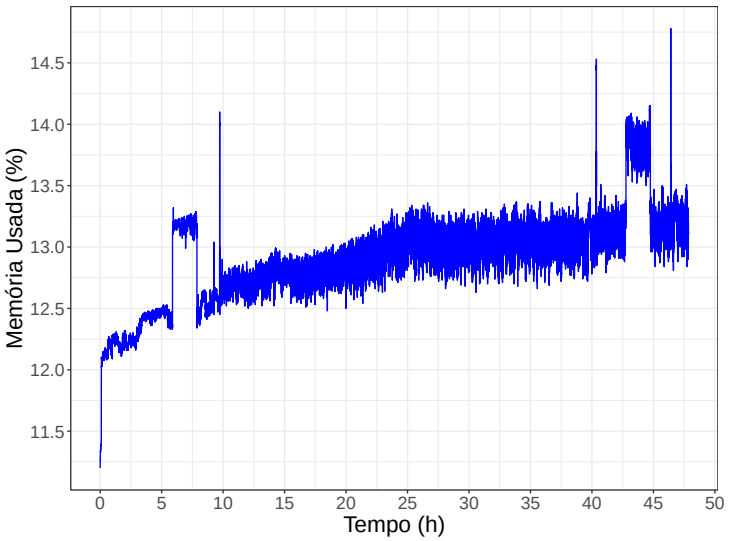

Figura 17 – MariaDB - Consumo de memória RAM - Carga Alta.

Foram realizados os testes estatísticos de Mann-Kendall e Sen's slope para confirmar as suspeitas de aumento no consumo de memória nos dados apresentados nas Figuras 15, 16 e 17. Os resultados desses testes, bem como as tendências observadas nos gráficos,

são apresentados na Tabela 11. Em todos os casos, o valor de *p-value* foi inferior a 0,05, indicando a existência de uma tendência estatisticamente significativa nos dados. Além disso, o valor de S foi positivo para todas as cargas, evidenciando uma tendência de crescimento. Esse crescimento também é corroborado pelos valores positivos obtidos para a inclinação de Sen's slope. Portanto, os resultados sugerem uma tendência de aumento no uso de memória ao longo do tempo, em todas as situações, o que pode ser interpretado como um indício de envelhecimento do software nesse componente devido à degradação da memória.

Tabela 11 – MariaDB - Teste de Mann-Kendall e Sen's slope para o uso de memória RAM.

SGBD	**Carga**	***p-value***	**S**	**Sen's slope (%/h)**	**Tendência**
MariaDB	Baixa	2,2e-16	9,52e+02	0,024	Crescimento
	Média	2,2e-16	9,38e+02	0,024	Crescimento
	Alta	1,13e-14	8,7e+02	0,017	Crescimento

4.2.2 Análise do Tempo de Resposta

Foi realizada uma análise do tempo de resposta com o intuito de investigar possíveis perdas de desempenho relacionadas às cargas utilizadas e obter mais evidências sobre o envelhecimento do software no ambiente com o MariaDB. O objetivo dessa análise foi verificar se a utilização contínua do sistema SGBD resultaria em uma deterioração do desempenho, conforme indicado pelo aumento do tempo de resposta ao longo do período em que as cargas foram aplicadas.

A Figura 18 ilustra o comportamento do tempo de resposta durante o teste com carga baixa. Embora sejam observadas oscilações, é possível notar uma tendência de queda ao analisar o ponto inicial e o ponto final do gráfico. Todavia, para determinar a tendência desses dados, é necessário realizar testes estatísticos. O tempo de resposta médio obtido com essa carga foi de 52 ms.

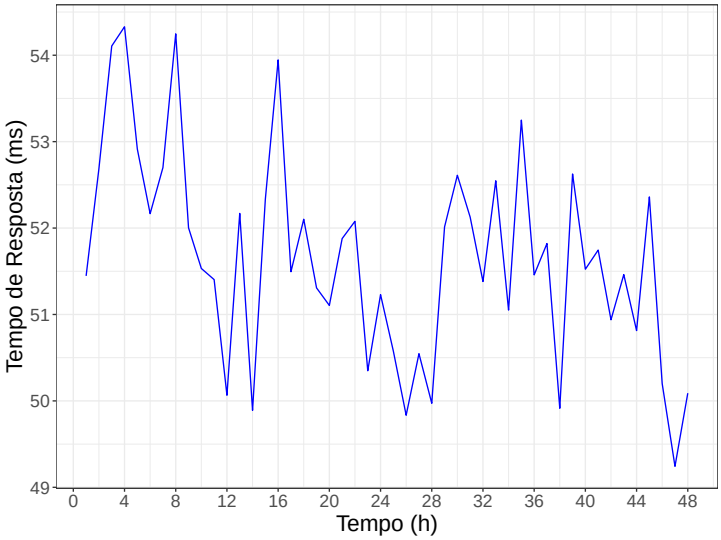

Figura 18 – MariaDB - Tempo de Resposta - Carga Baixa.

Com a utilização da carga média, o tempo de resposta médio foi de 382 ms. No entanto, na Figura 19, observamos picos que excedem os 500 ms. Não é possível determinar claramente se houve uma tendência de aumento no tempo de resposta no decorrer do teste. Portanto, os testes estatísticos de Mann-Kendall e do estimador Sen's slope são necessários para detectar se houve ou não uma perda de desempenho do MariaDB ao longo do tempo, e assim, obter mais um indício do envelhecimento de software pela perda desempenho ao longo do tempo.

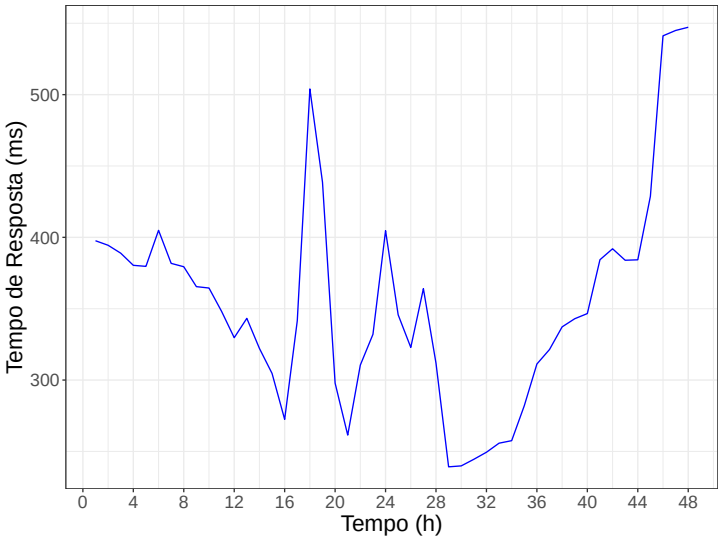

Figura 19 – MariaDB - Tempo de Resposta - Carga Média.

Na Figura 20, é apresentado o comportamento do tempo de resposta sob carga alta. Nesse gráfico, percebemos que o tempo de resposta se manteve relativamente estável até

as 22 horas de teste, quando começam a surgir oscilações. O tempo de resposta médio no experimento com essa carga apresentou um aumento significativo em comparação com as cargas mais baixas, chegando a 1484 ms. Além disso, é possível notar um crescimento nos resultados ao longo do tempo, porém são necessários testes estatísticos para confirmar essa suspeita.

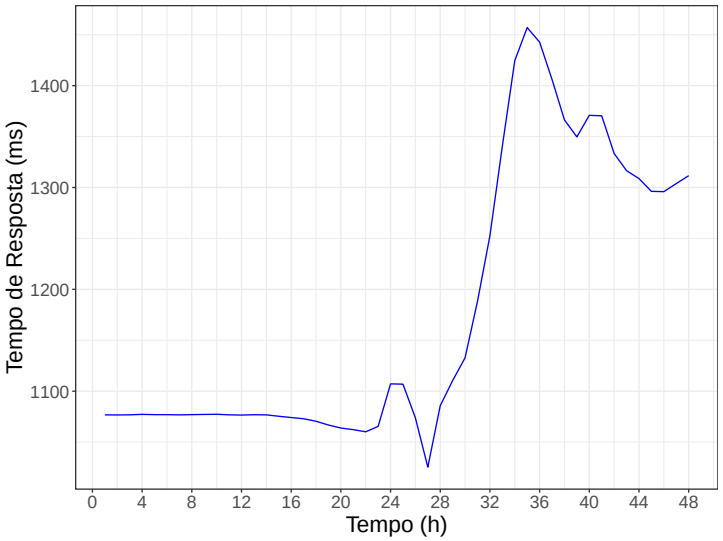

Figura 20 – MariaDB - Tempo de Resposta - Carga Alta.

A Tabela 12 apresenta os resultados dos testes estatísticos realizados nos dados de tempo de resposta. Ao analisar as cargas baixa e média, não foi encontrada evidência estatística de uma tendência de crescimento. No entanto, na carga média, embora o valor de *p-value* seja maior que 0,05 (indicando a falta de uma tendência estatisticamente significativa), observou-se que os valores de S e da magnitude de inclinação de Sen's slope foram negativos. Essa observação sugere uma tendência de decrescimento nos dados ao longo do tempo. Além disso, os resultados indicam que apenas na carga alta foi confirmada estatisticamente a degradação de desempenho, evidenciada pelo aumento do tempo de resposta durante o teste.

Tabela 12 – MariaDB - Teste de Mann-Kendall e Sen's slope para o tempo de resposta.

SGBD	**Carga**	***p-value***	**S**	**Sen's slope (ms/h)**	**Tendência**
MariaDB	Baixa	0,003	-332	-0,037	Decrescimento
	Média	0,676	-4,8e+01	-0,258	Decrescimento
	Alta	0,0002	4,26e+02	5,75	Crescimento

4.2.3 Análise dos Processos

Nesta seção, são descritas as análises dos processos no ambiente com o MariaDB. Nosso principal objetivo ao realizar essas análises é identificar possíveis indícios de envelhecimento de software no ambiente por meio dos processos do sistema operacional e do SGBD utilizado. Além disso, também identificamos os cinco processos que consumiram mais memória e revelamos aqueles que tiveram o maior aumento durante a execução dos experimentos, levando em consideração tanto o processo do SGBD quanto os demais.

4.2.3.1 Processos que mais consumiram Memória

Como parte da análise de processos, identificamos os cinco processos que mais consumiram memória RAM no servidor utilizando o MariaDB como SGBD. Semelhante ao MySQL, os dados dessa análise foram extraídos do último arquivo coletado pelo *script* de monitoramento de processos, uma vez que nele encontramos uma fase em que o ambiente já está degradado devido à alta quantidade de requisições enviadas pelo cliente, próximo às 48 horas de teste para uma determinada carga.

A Tabela 13 apresenta a descrição dos cinco processos que mais consumiram memória em carga baixa, enquanto a Figura 21 ilustra a quantidade de memória utilizada por esses processos. É possível observar que o processo responsável pela execução do MariaDB foi o segundo que mais consumiu memória no ambiente, ficando atrás apenas do processo gnome-shell. O valor de RSS do MariaDB foi de 105,46 MB, enquanto o gnome-shell utilizou 172,32 MB. Além disso, outros processos como Xorg vt1 -dis, tracker-store e snapd também estão incluídos nessa classificação.

Tabela 13 – MariaDB - Descrição dos processos que mais consumiram memória - Carga Baixa.

PIDs	Descrição
1203	**gnome-shell** - Ambiente de desktop GNOME.
844	**mariadb** - Processo do MariaDB.
11159	**Xorg vt1 -dis** - Servidor de exibição padrão.
1638	**tracker-store** - Processo de indexação de arquivos.
543820	**snapd** - Processo que permite instalação e execução de aplicativos empacotados no formato 'snap'.

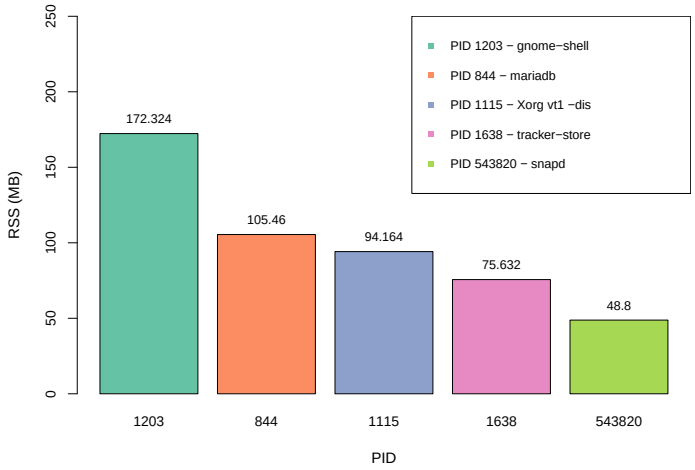

Figura 21 – MariaDB - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Baixa.

Na Tabela 22, são apresentados os cinco processos que mais consumiram memória em carga média. Embora esses processos tenham PIDs diferentes, as descrições podem ser encontradas na Tabela 13. Mais uma vez, o processo do MariaDB foi o segundo que mais consumiu memória no servidor, com um valor de RSS igual a 117,54 MB. Comparado ao seu valor durante a carga baixa, houve um aumento percentual de aproximadamente 11,46%. Além disso, é possível observar que os mesmos processos e a mesma ordem de classificação obtidos no experimento durante a carga baixa também foram observados na carga média.

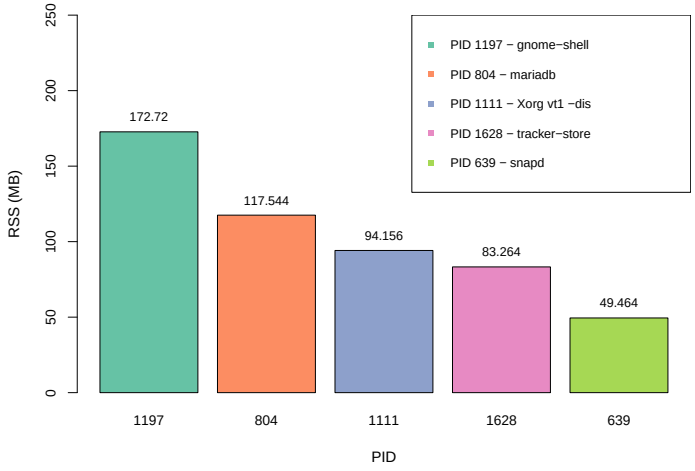

Figura 22 – MariaDB - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Média

A Figura 23 apresenta os processos que mais consumiram RAM durante os experimentos em carga alta. Além disso, embora esses processos tenham PIDs diferentes, suas descrições podem ser vistas na Tabela 13. Comparando com a carga média, houve um aumento percentual de aproximadamente 8,8%, enquanto em relação à carga baixa, o aumento foi de 21,26%. Também é importante observar que a mesma ordem de classificação e os mesmos processos que apareceram nos experimentos de carga baixa e média também são encontrados nessa classificação em carga alta.

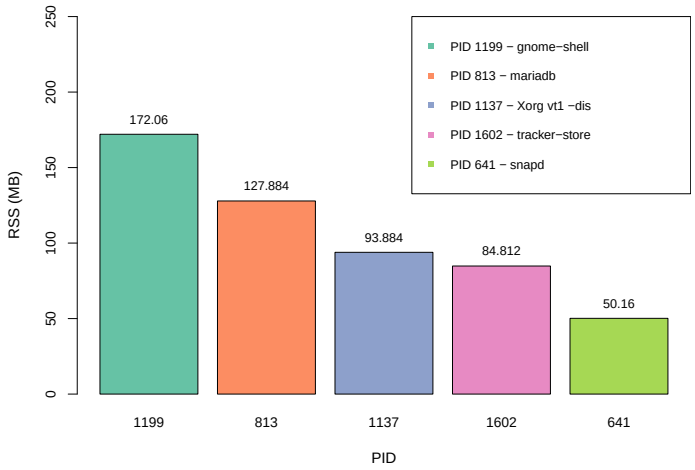

Figura 23 – MariaDB - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Alta.

Com base na análise realizada, pode-se concluir que o processo do MariaDB foi um dos principais consumidores de memória nos diferentes cenários de carga (baixa, média e alta). Durante a execução dos experimentos, ele ocupou a segunda posição, ficando atrás apenas do processo gnome-shell. Além disso, é importante destacar que a ordem de classificação dos processos e os mesmos processos foram observados nos diferentes cenários de carga. Isso indica que o MariaDB teve um impacto significativo no consumo de memória, independentemente da carga de trabalho, e pode ter contribuído para os efeitos de envelhecimento de software no ambiente analisado.

4.2.3.2 Processos que mais cresceram juntamente com o SGBD

Nesta análise, identificamos os quatro processos que apresentaram o maior aumento no consumo de memória RAM, juntamente com o processo relacionado à execução do

MariaDB durante as cargas. Semelhante ao MySQ, para calcular a porcentagem de crescimento, utilizamos os valores de USS do arquivo correspondente à primeira hora, bem como o referente a última hora de teste, ambos coletados pelo *script* de monitoramento de processos.

Na Tabela 14, é apresentado o ranking dos processos que tiveram o maior crescimento no consumo de memória durante os testes em carga baixa. A descrição desses processos pode ser encontrada na Tabela 15. Conforme observado na Tabela 14, o processo tracker-store ocupou a primeira posição nesse ranking, com um aumento percentual de 334% no consumo de memória. Em seguida, o processo de inicialização do Ubuntu, init splash, ficou em segundo lugar. O processo responsável pela execução do MariaDB ficou na 28ª posição, apresentando um crescimento baixo de apenas 0,01%.

Tabela 14 – MariaDB - Processos que mais cresceram juntamente com o SGBD - Carga Baixa.

Ranking	**PID**	**Processo**	**Crescimento**
1º	1638	tracker-store	334%
2º	1	init splash	142%
3º	1646	tracker-extract	22%
4º	1405	tracker-miner	21%
28º	844	mariadb	0,01%

Tabela 15 – MariaDB - Descrição dos processos que mais cresceram juntamente com o SGBD - Carga Baixa

Processo	**Descrição**
tracker-store	Processo de indexação de arquivos
init splash	Processo de inicialização do Ubuntu.
tracker-extract	Processo que indexa e extrai metadados de arquivos no sistema de arquivos
traker-miner	Processo responsável por indexar e pesquisar arquivos no sistema de arquivos.
mariadb	Processo do MariaDB

A Tabela 16 apresenta o ranking dos processos que tiveram maior crescimento no consumo de memória RAM em carga média. Embora esses processos tenham diferentes PIDs, a descrição de cada processo pode ser encontrada na Tabela 15, com exceção do

processo snapd, que é responsável pela instalação e execução de aplicativos empacotados no formato 'snap'. Novamente, o processo tracker-store ocupa a primeira posição no ranking com um crescimento de 377%. Por outro lado, o processo relacionado ao MariaDB apresentou um crescimento baixo de 0,37%, ocupando a 26ª posição. No entanto, quando comparado ao seu crescimento em carga baixa, esse aumento foi significativo.

Tabela 16 – MariaDB - Processos que mais cresceram juntamente com o SGBD - Carga Média.

Ranking	**PID**	**Processo**	**Crescimento**
1º	1628	tracker-store	377%
2º	639	snapd	30%
3º	1673	tracker-extract	22%
4º	1382	tracker-miner	20%
26º	804	mariadb	0,37%

Por fim, na Tabela 17, são apresentados os processos que tiveram o maior aumento no consumo de memória durante os testes de carga alta. Com uma carga mais elevada, o processo tracker-store registrou o maior crescimento, atingindo 397%, em comparação com as cargas baixa e média. Também é importante ressaltar o crescimento significativo do processo responsável pela execução do MariaDB, que atingiu um crescimento de 6%, bem maior que os valores atingidos por ele com as cargas baixa e média, passando a ocupar a sexta posição no nesse ranking.

Tabela 17 – MariaDB - Processos que mais cresceram juntamente com o SGBD - Carga Alta.

Ranking	**PID**	**Processo**	**Crescimento**
1º	1602	tracker-store	397%
2º	641	snapd	29%
3º	1659	tracker-extract	21%
4º	1	init splash	13%
6º	813	mariadb	6%

Semelhante ao MySQL, o processo "tracker-store"ocupou a primeira posição em todos os casos, sugerindo que esse processo desempenha um papel relevante na

degradação do ambiente. No entanto, os processos relacionados ao MariaDB não podem ser negligenciados, pois também têm impacto na degradação do ambiente adotado.

4.2.3.3 Análise dos Processos do MariaDB

Como evidenciado pela análise anterior, os processos relacionados ao MariaDB podem ser um dos fatores responsáveis pela degradação dos ambientes do sistema adotado. Nesta análise, exploraremos mais detalhadamente a investigação desses processos.

A Figura 24 ilustra a evolução do processo de execução do MariaDB em uma carga baixa. É possível observar no gráfico um aumento progressivo do processo ao longo do tempo. Inicialmente, o valor de USS foi de 98,716 MB, alcançando 98,724 MB ao final do teste, indicando um crescimento relativamente baixo ao comparar os dois valores. Destacam-se elevações mais significativas no uso de memória entre a primeira e a terceira hora do experimento, assim como entre a trigésima primeira e a trigésima segunda hora. Em ambos os casos, esses períodos de crescimento são seguidos por uma estabilização no uso de memória, em que o valor se mantém constante.

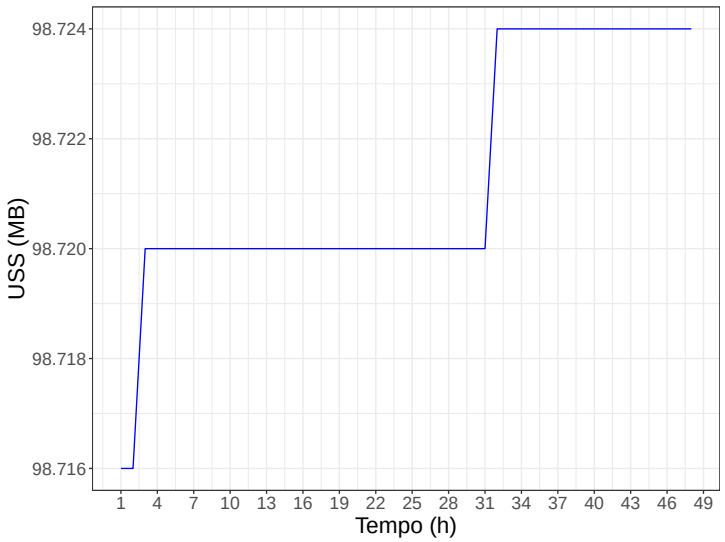

Figura 24 – Crescimento do processo do MariaDB ao longo do tempo de teste - Carga Baixa

Na Figura 25 é apresentada a evolução do uso de memória pelo processo do MariaDB em carga média. A partir dos resultados desse gráfico, podemos observar um aumento no uso de memória pelo processo do MariaDB. Ao contrário do comportamento observado em carga baixa (ver Figura 24), esse crescimento foi mais constante. No entanto, se

analisarmos os valores iniciais e finais de USS, percebemos que, embora tenha havido um crescimento, a diferença não é grande, o que reforça a necessidade de realizar testes estatísticos para confirmar as suspeitas de envelhecimento do software no banco de dados.

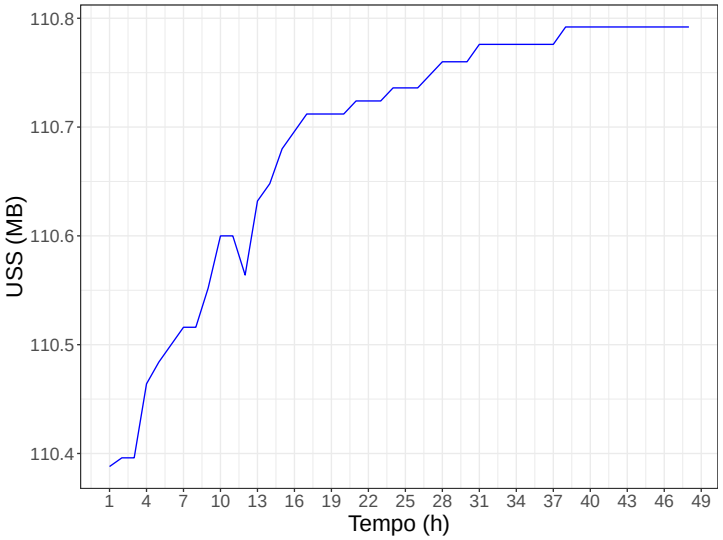

Figura 25 – Crescimento do processo do MariaDB ao longo do tempo de teste - Carga Média

Também em carga alta, como é visto na Figura 26, o processo responsável pela execução do MariaDB apresentou crescimento ao longo do tempo. Com o uso da carga mais intensa, observamos um aumento mais significativo no uso de memória, passando de aproximadamente 114 MB na primeira hora de teste para cerca de 121 MB no final do teste. Ao comparar esses resultados com os das Figuras 24 e 25, é possível notar um crescimento mais acentuado nos dados.

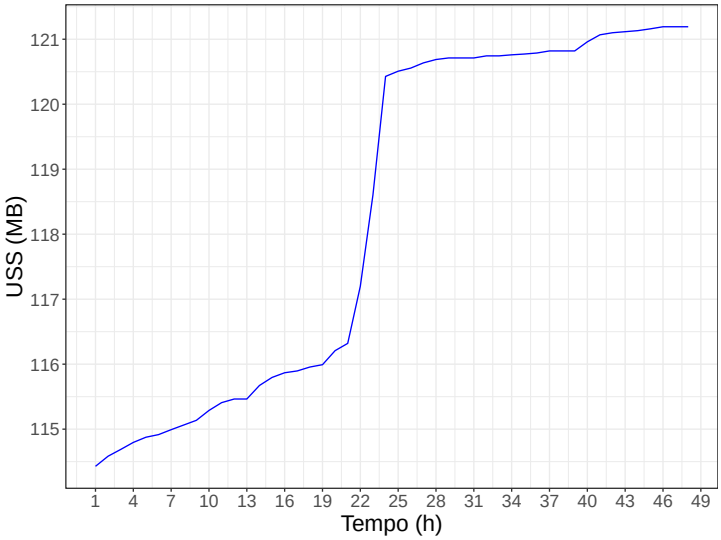

Figura 26 – Crescimento do processo do MariaDB ao longo do tempo de teste - Carga Alta

A Tabela 18 apresenta os resultados dos testes estatísticos realizados nos dados dos gráficos das Figuras 24, 25 e 26. O valor de *p-value* obtido foi inferior a 0,05, indicando a presença de uma tendência estatisticamente significativa nos dados. Além disso, os valores de S (estatística de Mann-Kendall) e do estimador Sen's slope foram positivos, sugerindo uma tendência de crescimento em todas as condições de carga utilizadas ao longo do tempo. Os resultados também demonstram que a carga alta apresentou o maior valor absoluto de Sen's slope, confirmando que o crescimento foi maior nessa condição em comparação às demais. Esses testes permitiram confirmar estatisticamente, as suspeitas de envelhecimento do software no MariaDB, evidenciando um possível acúmulo no processo responsável por sua execução no ambiente durante a aplicação das cargas de trabalho. Vale destacar que as cargas de trabalho têm impacto direto na degradação do ambiente através do MariaDB, onde cargas maiores resultam em degradações mais elevadas.

Tabela 18 – MariaDB - Teste de Mann-Kendall e Sen's slope para o processo do SGBD.

SGBD	**Carga**	***p-value***	**S**	**Sen's slope (MB/h)**	**Tendência**
MariaDB	Baixa	1,2e-09	5,85e+02	0,0001	Crescimento
	Média	2,2e-16	1,03e+03	0,006	Crescimento
	Alta	2,2e-16	1,12e+03	0,159	Crescimento

4.3 PostgreSQL

Nesta seção, serão apresentados os resultados obtidos no experimento em que o PostgreSQL foi utilizado como SGBD. Primeiramente, será apresentada a análise do consumo de memória RAM. Em seguida, será abordada a análise do tempo de resposta para verificar se houve alguma perda de desempenho do SGBD. Além disso, realizaremos uma análise dos processos, identificando os principais responsáveis pelo consumo de memória, bem como os possíveis processos relacionados ao envelhecimento do software no ambiente.

4.3.1 Análise do Consumo de Memória

Como parte das análises realizadas para investigar a presença de indícios de envelhecimento de software no ambiente do PostgreSQL, foram coletados dados de uso de memória RAM no servidor. A Figura 27 apresenta o monitoramento da memória com a carga baixa, onde é possível observar um crescimento gradual do consumo de memória mesmo com a carga constante durante todas as 48 horas de envio de requisições. Isso sugere que, mesmo com uma carga estável, o servidor gradualmente alocou mais memória para processar as requisições. O percentual máximo de memória usada chegou próximo dos 13,5% já perto do fim da execução do experimento.

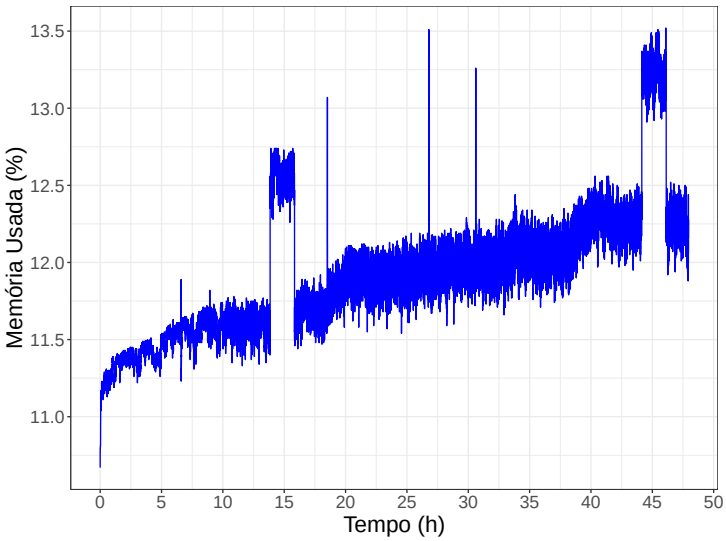

Figura 27 – PostgreSQL - Consumo de memória RAM - Carga Baixa.

No experimento com carga média, no qual houve um aumento no número de requisições por segundo enviadas ao servidor, os resultados obtidos na Figura 28 revelam

um patamar mais elevado de utilização da memória RAM em comparação com o gráfico apresentado na Figura 27. Ao analisar a Figura 28, fica evidente que o uso da memória foi aumentando progressivamente ao longo do tempo de teste. Esse comportamento demonstra que, à medida que mais requisições foram sendo processadas, houve uma necessidade maior de alocação de memória para atender à demanda. Dessa forma, a memória utilizada alcançou picos superiores a 15%, indicando que o sistema estava sendo submetido a um maior esforço em termos de consumo de recursos.

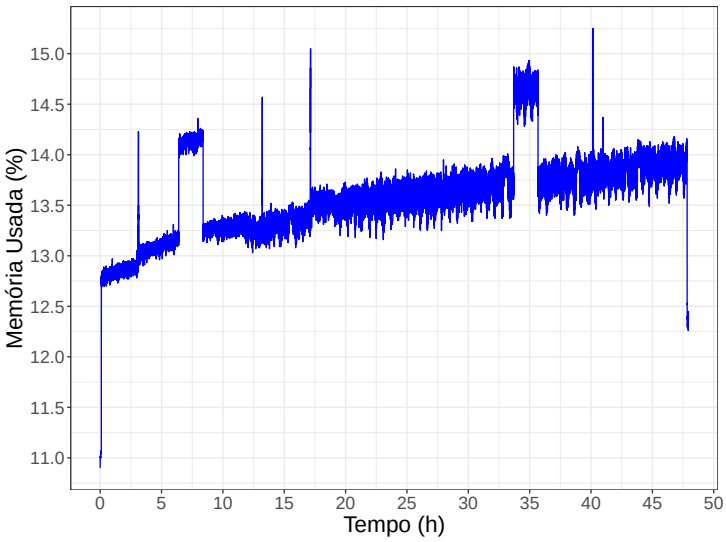

Figura 28 – PostgreSQL - Consumo de memória RAM - Carga Média.

Assim como nos gráficos anteriores com as cargas baixa e média, o gráfico apresentado na Figura 29 também revela um aumento progressivo no percentual de uso de memória RAM ao longo do tempo durante a execução de uma carga mais alta. Nesse caso, é possível observar um patamar mais elevado e picos mais altos de utilização de memória quando uma carga de trabalho mais intensa é aplicada ao banco de dados instalado no servidor. Analisando a Figura 29, podemos notar que logo no início do experimento houve um aumento abrupto no percentual de memória usada. O valor subiu de aproximadamente 10,5% para acima de 14% de utilização da memória. Esse aumento repentino sugere que, ao iniciar a carga de trabalho mais elevada, o sistema demandou imediatamente uma alocação maior de memória para lidar com o aumento das requisições. Após esse aumento inicial, o sistema se manteve estável, apresentando um leve crescimento ao longo do tempo.

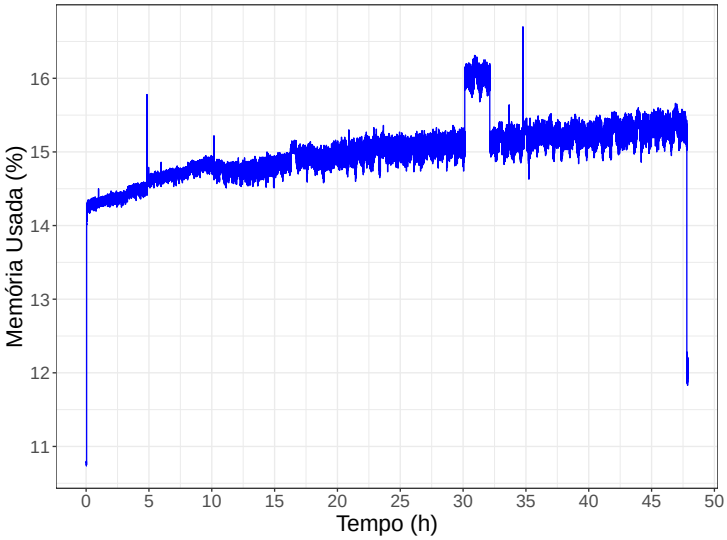

Figura 29 – PostgreSQL - Consumo de memória RAM - Carga Alta.

Para confirmar a suspeita de uma tendência de crescimento no uso da memória RAM ao longo do tempo nos gráficos apresentados nas Figuras 27, 28 e 29, foram realizados os testes de Mann-Kendall e obtido o estimador da inclinação de Sen's slope. Os resultados desses testes estão resumidos na Tabela 19, que apresenta os valores obtidos com as diferentes cargas de trabalho aplicadas ao servidor. Para todas as cargas de trabalho testadas, o valor do *p-value* foi inferior a 0,05, indicando que existe uma tendência nos dados, seja de crescimento ou decrescimento. Além disso, os valores de S e do estimador de Sen's slope foram positivos em todas as situações. Portanto, podemos concluir que houve uma tendência de crescimento no consumo de memória ao longo do tempo para todas as cargas de trabalho aplicadas ao servidor. Esses resultados sugerem que ocorreu uma degradação da memória no ambiente, resultando em um acúmulo progressivo ao longo do tempo durante a realização dos experimentos. No entanto, é importante observar que a carga baixa apresentou um crescimento ligeiramente superior as outras cargas, mesmo com a mesma configuração de hardware e software. Isso pode sugerir que as cargas de trabalho podem não afetar diretamente a magnitude da inclinação de Sen's slope.

Tabela 19 – PostgreSQL - Teste de Mann-Kendall e Sen's slope para o uso de memória RAM.

SGBD	Carga	*p-value*	S	Sen's slope (%/h)	Tendência
PostgreSQL	Baixa	7,54e-16	9,08e+02	0,02	Crescimento
	Média	1,01e-13	8,38e+02	0,018	Crescimento
	Alta	2,2e-16	9,36e+02	0,017	Crescimento

4.3.2 Análise do Tempo de Resposta

A fim de investigar a perda de desempenho do sistema devido à aplicação das cargas de trabalho, monitoramos o tempo de resposta durante a execução do experimento. Utilizamos o PostgreSQL no ambiente para investigar se o uso contínuo do SGBD levaria a uma degradação no desempenho, manifestada pelo aumento do tempo de resposta ao longo do período de aplicação das cargas.

A Figura 30 ilustra o comportamento do tempo de resposta ao longo do período de teste durante o experimento com carga baixa. Foi obtido um tempo de resposta médio de 31 ms nesse experimento, e uma tendência de queda nos dados pode ser observada. No entanto, para confirmar essas observações, os testes estatísticos de Mann-Kendall e Sen's slope são necessários.

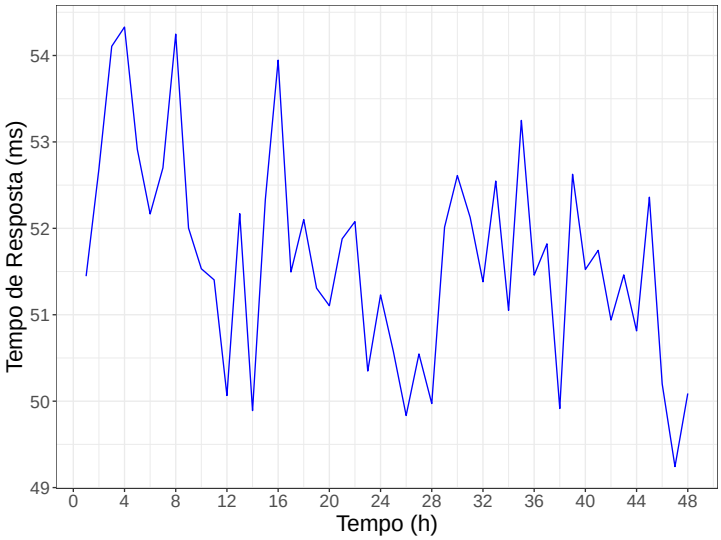

Figura 30 – PostgreSQL - Tempo de Resposta - Carga Baixa.

Na Figura 31, é apresentado o comportamento do tempo de resposta durante o

experimento com carga média. É possível observar que foram obtidos tempos de resposta consideravelmente maiores para essa carga em comparação com a carga baixa. Além disso, podemos notar um crescimento progressivo dos dados ao longo do tempo, com um aumento mais significativo ocorrendo após as 36 horas de teste. O tempo de resposta médio registrado para essa carga foi de 680 ms. Esses resultados podem indicar uma deterioração do desempenho do sistema sob a carga média, com tempos de resposta mais elevados e um aumento notável ao longo do período de teste.

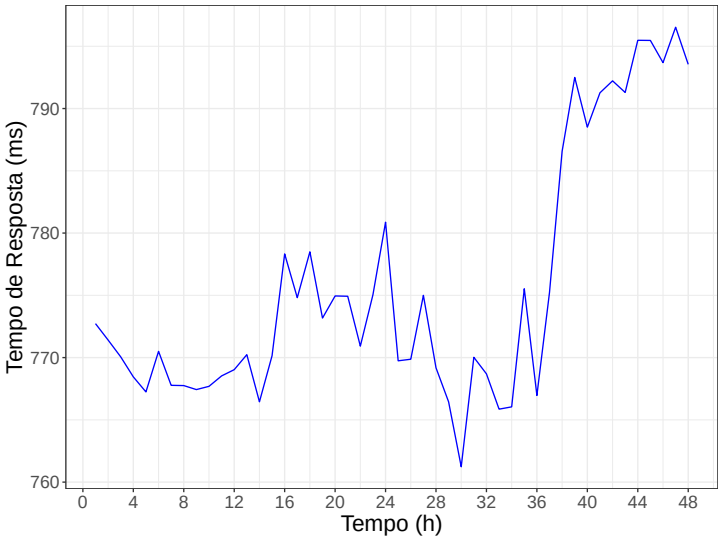

Figura 31 – PostgreSQL - Tempo de Resposta - Carga Média.

Por fim, durante a carga alta, obteve-se um tempo de resposta médio de 1932 ms. Ao analisar o gráfico representado na Figura 32, é possível observar que o tempo de resposta se mantém relativamente estável até as 42 horas de teste. A partir desse ponto, ocorre um crescimento mais acentuado, atingindo um pico por volta das 46 horas de teste, seguido por uma queda nos dados. Observando o gráfico, é evidente uma clara tendência de crescimento ao final dos testes, seguida por uma queda abrupta.

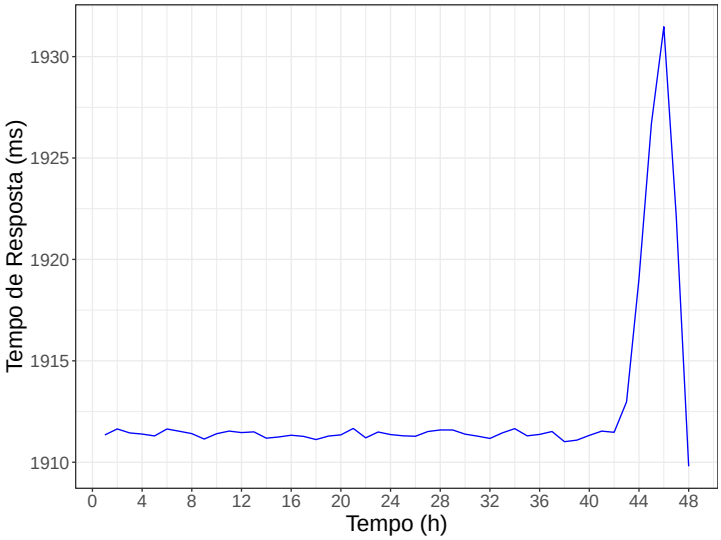

Figura 32 – PostgreSQL - Tempo de Resposta - Carga Alta.

A Tabela 20 apresenta os resultados dos testes estatísticos realizados para verificar a tendência nos dados de tempo de resposta. O valor do *p-value* foi superior a 0,05 para os experimentos de carga baixa e alta, o que indica uma ausência de tendência nos dados. No entanto, observou-se que os valores de S e a inclinação de Sen foram negativos para a carga baixa, indicando estatisticamente uma tendência de decrescimento nos dados. Já para a carga alta, os valores de S e a inclinação de Sen foram positivos, revelando uma tendência de crescimento. Para a carga média, a tendência de crescimento foi mais evidente, o que pode ser observado no gráfico da Figura 31, onde essa tendência é mais visível em comparação com as outras cargas. Para a carga alta, o valor do *p-value* foi inferior a 0,05, e os valores de S e a inclinação de Sen foram positivos, indicando uma tendência de crescimento. Vale destacar que o valor do slope é menor para a carga alta do que para a carga média. Isso pode ser decorrente da queda abrupta observada no tempo de resposta no gráfico de carga alta ao final do experimento.

Tabela 20 – PostgreSQL - Teste de Mann-Kendall e Sen's slope para o tempo de resposta.

SGBD	**Carga**	***p-value***	**S**	**Sen's slope (ms/h)**	**Tendência**
PostgreSQL	Baixa	0,29	-120	-0,0005	Decrescimento
	Média	7,654e-05	4,46e+02	0,48	Crescimento
	Alta	0,28	122	0,0031	Crescimento

4.3.3 Análise dos Processos

Nesta seção, são descritas as análises dos processos no ambiente com o PostgreSQL. O principal objetivo ao realizar essas análises é identificar possíveis sinais de envelhecimento de software no ambiente, observando os processos do sistema operacional e do SGBD utilizado. Além disso, identificamos os cinco processos que apresentaram maior consumo de memória e revelamos aqueles que apresentaram maior aumento durante a execução dos experimentos, levando em consideração tanto o processo do SGBD quanto os demais.

4.3.3.1 Processos que mais consumiram Memória

Com o uso do *script* de monitoramento, foi coletado ao longo da execução de cada teste, os processos que estavam sendo executados no ambiente de banco de dados. Com esses dados, foi possível verificar quais os processos estavam consumindo mais memória RAM do servidor e saber se as requisições enviadas ao PostgreSQL estavam fazendo com que SGBD fosse o principal responsável pelo uso da memória. Semelhante às análises realizadas dessa natureza nos outros bancos de dados, foi utilizado o último arquivo coletado pelo *script* de monitoramento de processos, pois nele podemos encontrar um estágio em que o ambiente já está degradado pela alta quantidade de requisições enviadas pelo cliente, próximo às 48 horas de teste para uma dada carga.

É importante destacar que para classificar os processos de modo a gerar os cinco processos que mais consumiram memória com os dados do monitoramento de cada carga de trabalho, foi feito um somatório dos valores de RSS de todos os processos relacionados a execução do PostgreSQL. Uma vez que somente pela execução do SGBD no Ubuntu, temos sete processos rodando no sistema operacional, mesmo sem nenhum usuário conectado. Além disso, para cada usuário conectado ao SGBD, um processo é criado no Ubuntu. Por exemplo, se tivermos cem usuários conectados fazendo requisições ao SGBD, iremos encontrar no arquivo de monitoramento dos processos, além dos sete processos que são padrão do sistema, mais cem referentes a usuários conectados. Deste modo, o PID 0, denominado de "postgres"na Tabela 21 e nas Figuras 33, 34, 35, representa de forma única os processos do PostgreSQL com os valores de RSS já somados. Isso foi necessário para diferenciá-lo dos demais processos do Ubuntu listados no arquivo coletado.

Na Tabela 21 temos os cinco processos que mais consumiram memória no ambiente

do servidor de banco de dados durante o teste com a carga baixa e a Figura 33 mostra a quantidade de memória usada por esses processos. Mesmo na segunda posição, o uso do PostgreSQL se destacou, pois chegou muito próximo de atingir o valor em megabyte de memória utilizada do processo "gnome-shell". Outros processos tiveram um uso bem mais discreto do que os dois primeiros, sendo eles o "Xorg vt1 -dis", o "tracker-store"e o "systemd-journald".

Tabela 21 – PostgreSQL - Descrição dos processos que mais consumiram memória - Carga Baixa.

PIDs	Descrição
1125	**gnome-shell** - Ambiente de desktop GNOME.
0	**postgres** - Soma dos Processos do PostgreSQL.
1060	**Xorg vt1 -dis** - Servidor de exibição padrão.
1592	**tracker-store** - Processo de indexação de arquivos.
282	**systemd-journald** - processo que coleta e armazena dados de log.

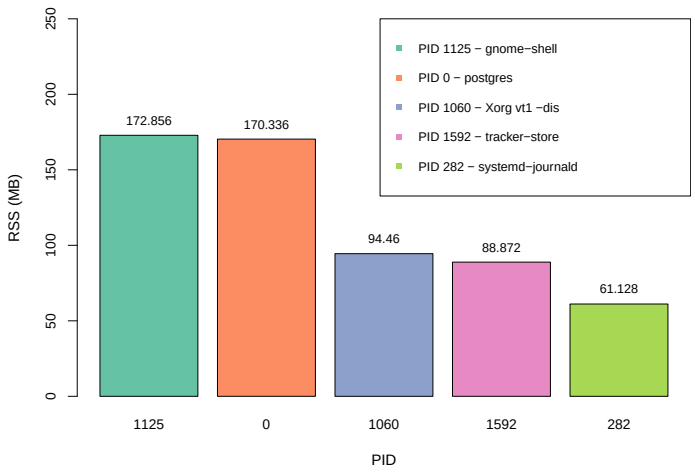

Figura 33 – PostgreSQL - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Baixa.

A Figura 34 representa os processos que apresentaram o maior consumo de memória durante a carga de trabalho média. Embora esses processos possuam PIDs diferentes, a descrição de cada um deles pode ser encontrada na Tabela 21. É importante ressaltar que os processos do PostgreSQL se destacaram pelo alto consumo de memória, como evidenciado na Figura 34, onde eles chegaram a utilizar quase 1000 MB da memória

RAM disponível. Além disso, é notável que o consumo de memória do PostgreSQL foi consideravelmente superior ao dos demais processos

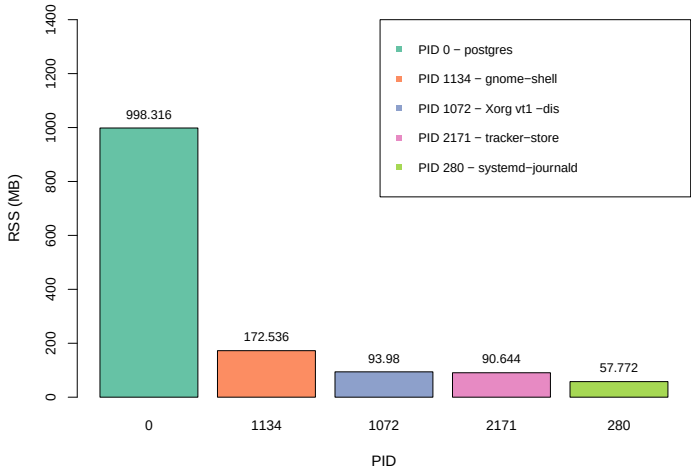

Figura 34 – PostgreSQL - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Média.

A Figura 35 exibe o consumo de memória dos cinco processos que mais utilizaram recursos durante a carga alta. Embora esses processos tenham PIDs diferentes, é possível encontrar a descrição de cada um deles na Tabela 21. Nessa carga específica, foi observado que o PostgreSQL foi o processo que consumiu a maior quantidade de memória, com um valor de RSS próximo a 2000 MB. Isso representa quase o dobro do consumo de memória em comparação com o valor atingido durante a carga média.

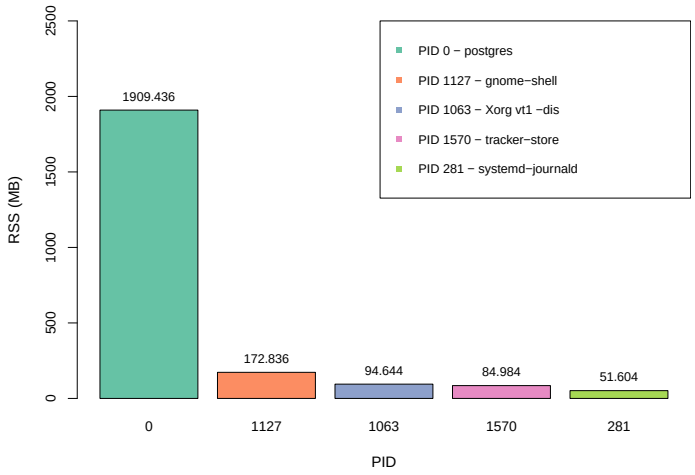

Figura 35 – PostgreSQL - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Alta.

Conforme observado nas análises, as cargas de trabalho aplicadas resultaram no PostgreSQL sendo um dos principais consumidores de memória RAM, superando significativamente outros processos tanto em carga média quanto em carga alta. Com base nisso, pode-se concluir que as cargas de trabalho empregadas no PostgreSQL fizeram com que ele consumisse a maior parte da memória do servidor na maioria dos casos. Essa observação sugere que o PostgreSQL possa ser um dos principais responsáveis pelos efeitos do envelhecimento de software no ambiente.

4.3.3.2 Processos que mais cresceram juntamente com o SGBD

Nesta análise, foram identificados os quatro processos que apresentaram o maior aumento no consumo de memória RAM, juntamente com os processos relacionados à execução do PostgreSQL durante as cargas de trabalho. Semelhante aos outros bancos de dados, para calcular a porcentagem de crescimento, utilizamos os valores de USS do arquivo correspondente à primeira hora, bem como o referente à última hora de teste. Ambos os valores foram coletados pelo *script* de monitoramento de processos.

Na Tabela 22, é apresentado o ranking dos processos que apresentaram o maior crescimento no consumo de memória durante os testes em carga baixa. A descrição desses processos pode ser encontrada na Tabela 23. Conforme observado na Tabela 22, o processo "tracker-store"ocupou a primeira posição nesse ranking, com um aumento percentual de 430% no consumo de memória. Em seguida, o processo "tracker-miner"ficou em segundo lugar. É interessante notar que as três primeiras posições foram ocupadas por processos do Tracker, uma ferramenta de indexação e pesquisa para o Linux. O processo responsável pela execução do PostgreSQL ocupou a 30ª posição, apresentando um crescimento de 0,16%.

Tabela 22 – PostgreSQL - Processos que mais cresceram juntamente com o SGBD - Carga Baixa.

Ranking	PID	Processo	Crescimento
1º	1592	tracker-store	430%
2º	1327	tracker-miner	26%
3º	1626	tracker-extract	23%
4º	1206	gsd-rfkill	8%
30º	0	postgres	0,16%

Tabela 23 – PostgreSQL - Descrição dos processos que mais cresceram juntamente com o SGBD - Carga Baixa

Processo	Descrição
tracker-store	Processo de indexação de arquivos
traker-miner	Processo responsável por indexar e pesquisar arquivos no sistema de arquivos.
tracker-extract	Processo que indexa e extrai metadados de arquivos no sistema de arquivos
gsd-rfkill	Processo específico para o controle de bloqueio de rádio.
postgres	Soma dos Processos do PostgreSQL

A Tabela 24 mostra o ranking dos processos que tiveram o maior crescimento no consumo de memória durante a execução do teste em carga média. Embora possuam PIDs diferentes, as descrições desses processos podem ser vistas na Tabela 23, com exceção do processo "init splash", responsável pela inicialização do Ubuntu, e do processo "sd-pam", responsável pela autenticação do usuário e gerenciamento de sessões. Novamente, o processo "tracker-store"ocupou a primeira posição, com crescimento de 440%, seguido pelo processo "init splash"com um crescimento de 68%. Em contraste, os processos relacionados ao PostgreSQL, com PID 0, tiveram um crescimento baixo, de apenas 0,02%, bem menor do que o crescimento apresentado em carga baixa.

Tabela 24 – PostgreSQL - Processos que mais cresceram juntamente com o SGBD - Carga Média.

Ranking	PID	Processo	Crescimento
1º	2171	tracker-store	440%
2º	1	init splash	68%
3º	1990	sd-pam	61%
4º	1998	tracker-miner	27%
31º	0	postgres	0,02%

O ranking dos processos que apresentaram o maior crescimento no consumo de memória em carga alta é apresentado na Tabela 25. Assim como na carga baixa, os processos relacionados ao Tracker retomaram as três primeiras posições, sendo que o processo "tracker-store"se destacou mais uma vez ao ocupar a primeira posição, registrando um crescimento de 395%. Em comparação com as cargas baixa e média, o processo do PostgreSQL teve seu maior crescimento, registrando 0,2%, o que o posicionou na 28ª posição do ranking.

Tabela 25 – PostgreSQL - Processos que mais cresceram juntamente com o SGBD - Carga Alta.

Ranking	PID	Processo	Crescimento
1º	1570	tracker-store	395%
2º	1332	tracker-miner	24%
3º	1615	tracker-extract	23%
4º	1	init splash	20%
28º	0	postgres	0,2%

Conforme os outros ambientes, aqui também foi possível observar que o processo "tracker-store"ocupou a primeira posição em todos os casos, sugerindo que esse processo desempenha um papel relevante na degradação do ambiente. No entanto, os processos relacionados ao PostgreSQL não podem ser negligenciados, pois também têm impacto na degradação do ambiente.

4.3.3.3 Análise dos Processos do PostgreSQL

Como evidenciado pela análise anterior, os processos relacionados ao PostgreSQL podem ser um dos fatores responsáveis pela degradação dos ambientes do sistema adotado. Nesta análise, exploraremos mais detalhadamente a investigação desses processos.

A Figura 36 apresenta a evolução dos processos responsáveis pela execução do PostgreSQL ao longo do tempo de teste em carga baixa. É possível observar uma utilização moderada da memória pelo SGBD durante todo o período de tempo. Na maior parte do tempo, a memória permaneceu estável, apresentando picos superiores a 15 MB. Também é perceptível uma tendência de crescimento nos dados, porém pouco perceptível, o que reforça a realização de testes estatísticos para confirmação da suspeita.

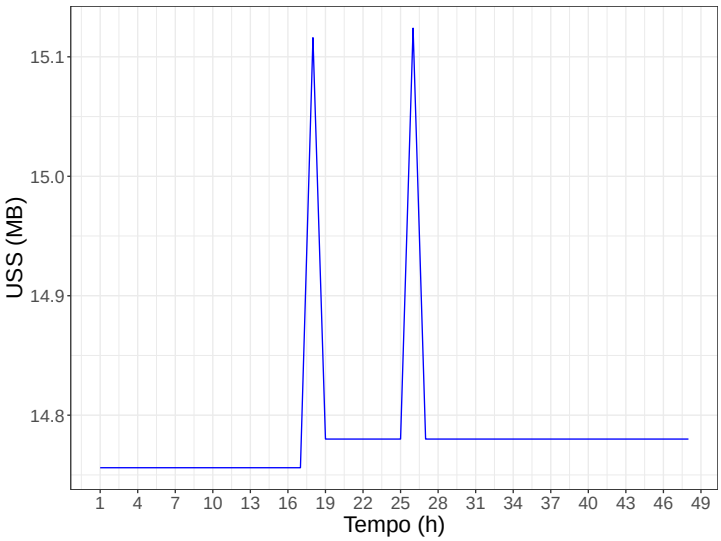

Figura 36 – Crescimento dos processos do PosgreSQL ao longo do tempo de teste - Carga Baixa

Na Figura 37 é mostrada a evolução do consumo de memória pelo o SGBD em carga média. Comparando com o gráfico de carga baixa (ver Figura 36), houve um aumento significativo no consumo da memória. Uma maior quantidade de picos de utilização de memória também foram observados ao longo do tempo. Ao observar a figura, não é possível perceber um aumento ou diminuição no consumo de memória pelo processo.

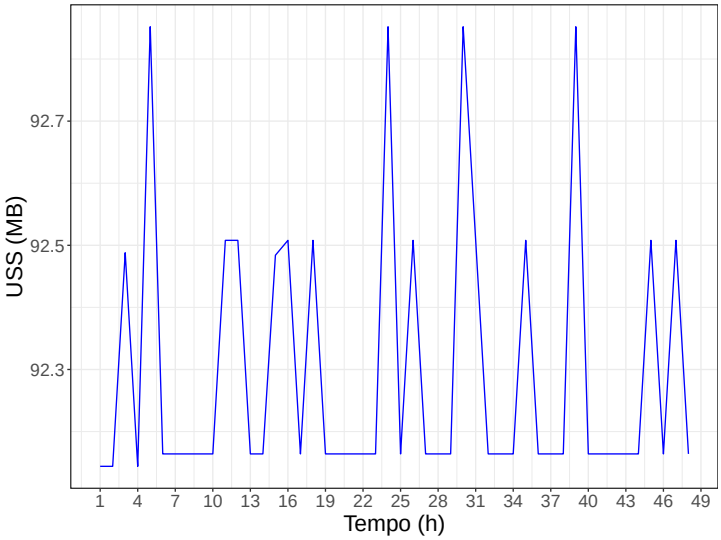

Figura 37 – Crescimento dos processos do PosgreSQL ao longo do tempo de teste - Carga Média

A Figura 38 apresenta a evolução do consumo de memória pelo PostgreSQL em condições de carga alta. Observa-se claramente no gráfico uma tendência de queda nos dados ao longo do tempo. Embora várias oscilações sejam visíveis ao longo do período analisado, é a partir da 28ª que se inicia uma trajetória de redução no consumo de memória.

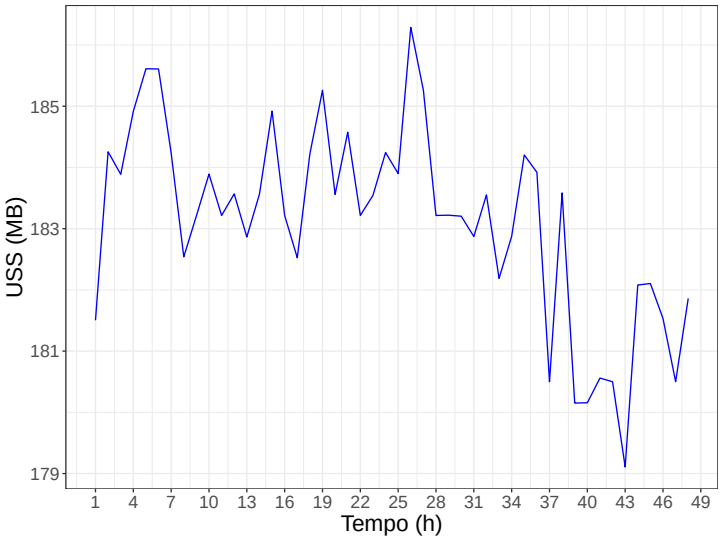

Figura 38 – Crescimento dos processos do PosgreSQL ao longo do tempo de teste - Carga Alta

A Tabela 26 apresenta os resultados dos testes estatísticos de Mann-Kendall e Sen's Slope para os dados de uso de memória pelos processos do PostgreSQL para as cargas de trabalho empregadas. Ao analisar a tabela, podemos observar que somente na carga alta é

possível afirmar estatisticamente a presença de uma tendência nos dados. Nesse caso, a tendência ao longo do tempo foi de queda, uma vez que os valores de S e a inclinação Sen são negativos, indicando uma diminuição no uso de memória com o passar do tempo.

Por outro lado, para as cargas baixa e média, não foi possível determinar uma tendência estatística significativa nos dados. Na carga baixa, embora os dados apresentem uma tendência estatisticamente significativa (*p-value* menor que 0,05), essa tendência é considerada ausente devido à inclinação Sen ser igual a zero. Isso indica que os valores não apresentam uma mudança sistemática ao longo do tempo, ou seja, não há uma direção clara de crescimento ou decrescimento no uso de memória. Na carga média também não foi encontrada uma tendência estatística nos dados, pois apresentou *p-value* maior que 0,05 e o valor de Sen's slope igual a zero.

Tabela 26 – PostgreSQL - Teste de Mann-Kendall e Sen's slope para os processos do SGBD.

SGBD	**Carga**	***p-value***	**S**	**Sen's slope (MB/h)**	**Tendência**
PostgreSQL	Baixa	4,95e-07	484	0	Ausente
	Média	0,39	8,4e+01	0	Ausente
	Alta	8,79e-05	-442	-0,065	Decrescimento

4.4 SQL Server

Nesta seção, serão apresentados os resultados obtidos nos testes realizados com o SQL Server. Primeiramente, será apresentada a análise do consumo de memória RAM. Em seguida, será abordada a análise do tempo de resposta para verificar se houve alguma perda de desempenho do SGBD. Além disso, realizaremos uma análise dos processos, identificando os principais responsáveis pelo consumo de memória, bem como os possíveis processos relacionados ao envelhecimento do software no ambiente.

4.4.1 Análise do Consumo de Memória

Foi monitorado o consumo de memória RAM ao longo do tempo a fim de verificar sintomas de envelhecimento de software no ambiente do SQL Server. A Figura 39 ilustra o comportamento da memória no servidor durante o experimento com a carga baixa. É

perceptível que, para essa carga, o consumo de memória aumentou gradualmente ao longo do tempo. Também foram observados picos de uso de memória superiores a 25%, e, na maior parte do tempo, o uso de memória se manteve entre 23,5% e 24,5%.

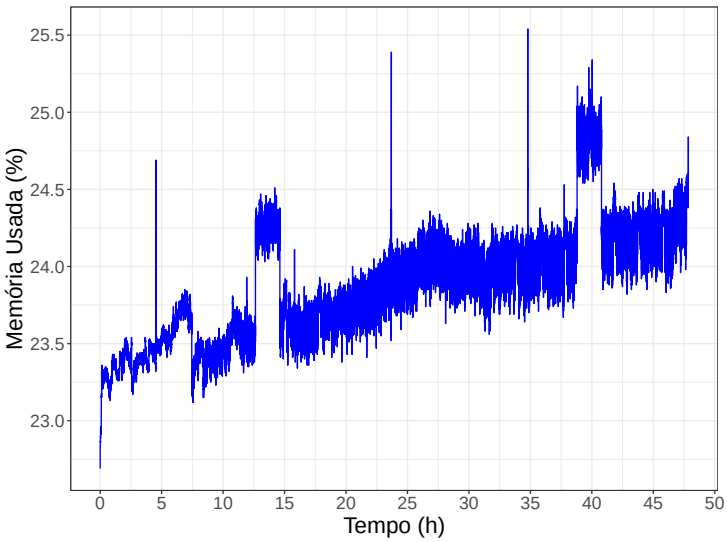

Figura 39 – SQL Server - Consumo de memória RAM - Carga Baixa.

Na Figura 40, é apresentado o consumo de memória no ambiente com o SQL Server durante a carga média. Em comparação com a carga baixa (ver Figura 39), observa-se um patamar mais elevado de utilização de memória, provavelmente devido a uma maior quantidade de requisições enviadas ao servidor. Além disso, é possível observar um crescimento nos dados ao longo do tempo. No entanto, é importante realizar testes estatísticos para confirmar as suspeitas de degradação da memória.

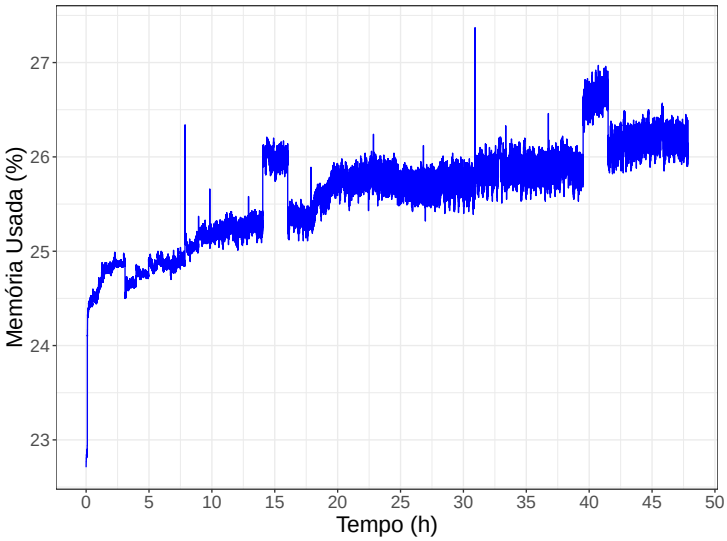

Figura 40 – SQL Server - Consumo de memória RAM - Carga Média.

A Figura 41 apresenta o comportamento do consumo de memória RAM ao longo do tempo durante o experimento com uma carga de trabalho alta. Inicialmente, observa-se um rápido crescimento na quantidade de memória utilizada, seguido por uma estabilização do consumo. Por volta das vinte horas de teste, é notado outro aumento abrupto no consumo. Esses dados indicam uma tendência geral de crescimento no consumo de memória ao longo do tempo com a utilização dessa carga de trabalho intensa. Além disso, ao compararmos com os gráficos da carga média e baixa, é evidente que o servidor precisou alocar mais memória para atender às demandas dessa carga de trabalho.

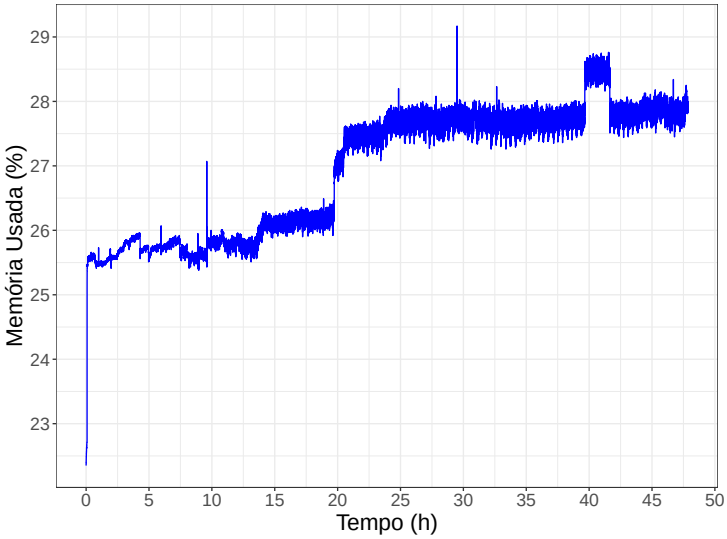

Figura 41 – SQL Server - Consumo de memória RAM - Carga Alta.

Foram conduzidos os testes estatísticos de Mann-Kendall e Sen's slope para confirmar as suspeitas de aumento no consumo de memória nos dados apresentados nas Figuras 39, 40 e 41. Os resultados destes testes, juntamente com as tendências observadas nos gráficos, são expostos na Tabela 27. Para todas as cargas aplicadas, o valor de *p-value* foi menor que 0,05, indicando a existência de uma tendência estatisticamente significativa nos dados. Adicionalmente, o valor de S foi positivo para todos os casos, demonstrando uma tendência de crescimento. Essa ascensão é também confirmada pelos valores positivos obtidos para a inclinação de Sen's slope. Portanto, os resultados sugerem uma inclinação de crescimento no uso de memória ao longo do tempo, em todas as circunstâncias, o que pode ser interpretado como um indício de envelhecimento do software nesse componente devido à degradação da memória. Além disso, vale destacar que as cargas de trabalho tiveram um impacto na degradação do ambiente, no qual cargas maiores resultaram em

inclinações (slopes) mais elevadas.

Tabela 27 – SQL Server - Teste de Mann-Kendall e Sen's slope para o uso de memória RAM.

SGBD	Carga	*p-value*	S	Sen's slope (%/h)	Tendência
SQL Server	Baixa	2,98e-14	8,56e+02	0,02	Crescimento
	Média	4,22e-15	8,84e+02	0,032	Crescimento
	Alta	2,2e-16	9,26e+02	0,06	Crescimento

4.4.2 Análise do Tempo de Resposta

Foi conduzida uma investigação sobre possíveis perdas de desempenho relacionadas às cargas utilizadas e para obter mais evidências sobre o envelhecimento do software no ambiente com o SQL Server. Na Figura 42, é apresentado a variação do tempo de resposta durante a execução da carga leve no ambiente em que foi utilizado o SGBD SQL Server. Durante esse teste, a média do tempo de resposta foi de 48 ms e é evidente a presença de flutuações significativas ao longo do período analisado. O gráfico exibido não oferece uma indicação clara quanto a uma possível tendência ascendente ou descendente dos dados ao longo do tempo. Portanto, a realização de análises estatísticas mais aprofundadas se torna crucial.

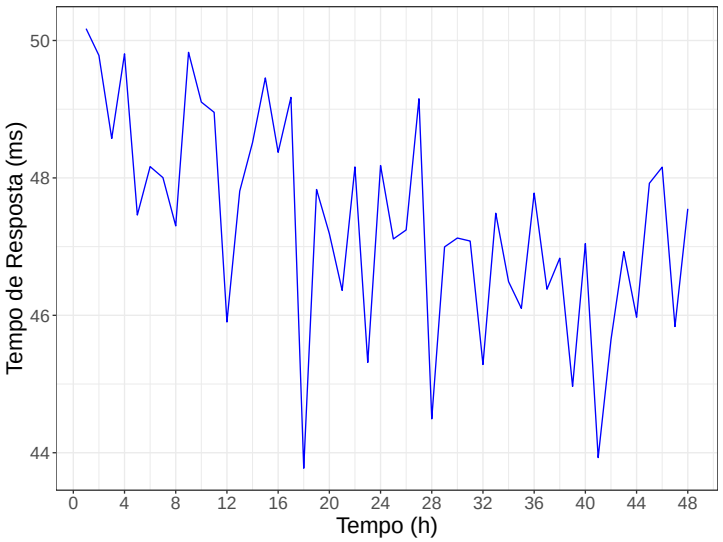

Figura 42 – SQL Server - Tempo de Resposta - Carga Baixa.

A Figura 43 ilustra como o tempo de resposta evolui com a utilização da carga

média. Os dados revelam um aumento gradual no tempo de resposta ao longo do tempo de execução da carga, indicando uma possível perda de desempenho do servidor. Até aproximadamente 12 horas de teste, o tempo de resposta permaneceu abaixo de 400 ms, porém, no final do teste, os valores subiram para mais de 800 ms. Comparando com a carga mais leve (ver Figura 42), é evidente um aumento significativo no tempo de resposta para essa carga, alcançando um valor médio de 645 ms.

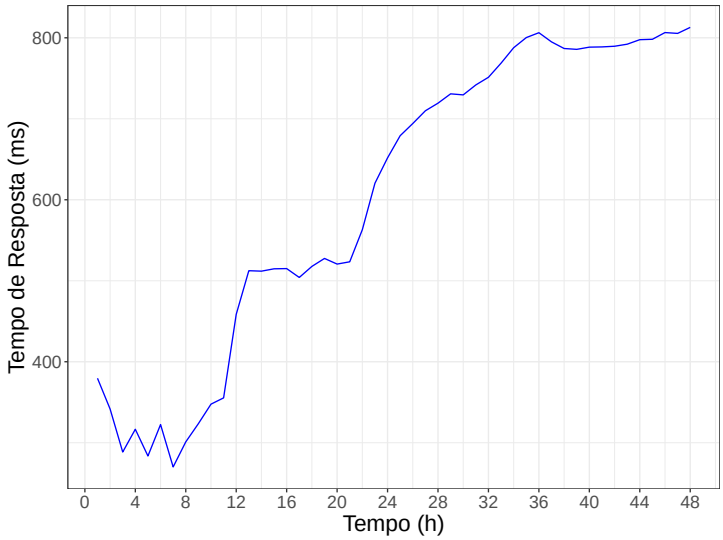

Figura 43 – SQL Server - Tempo de Resposta - Carga Média.

Finalmente, na Figura 44, é apresentada a evolução do tempo de resposta durante a aplicação da carga alta. Em comparação com as cargas baixa e média, é evidente que o SGBD levou significativamente mais tempo para lidar com a maior quantidade de requisições. O tempo de resposta médio para essa carga alta foi de 1583 ms. Até aproximadamente 22 horas de teste, o tempo de resposta manteve-se abaixo de 1250 ms, mas começou a apresentar aumento após esse período. No geral, é perceptível uma tendência de crescimento nos dados ao longo do tempo, o que pode mais uma vez indicar a deterioração do desempenho do sistema e servir como outro indício do envelhecimento do software no ambiente do SQL Server.

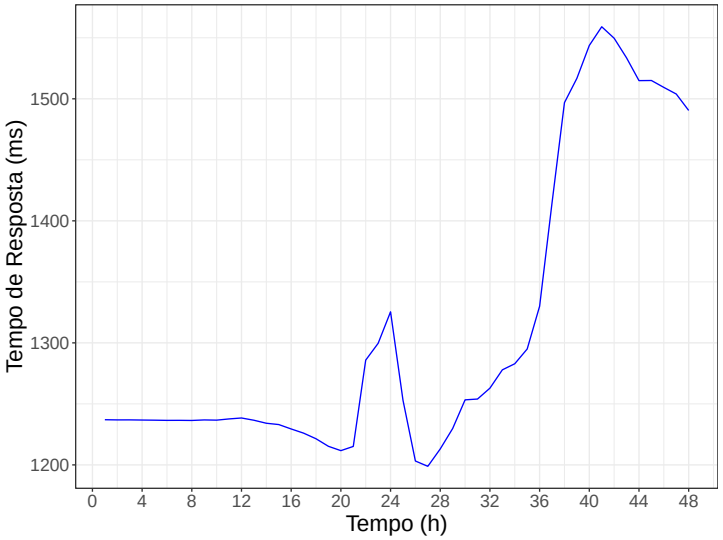

Figura 44 – SQL Server - Tempo de Resposta - Carga Alta.

Os dados de tempo de resposta foram analisados utilizando os testes de Mann-Kendall e Sen's slope para verificar de maneira estatística as suspeitas de redução de desempenho que foram observadas. Os resultados desses testes estão resumidos na Tabela 28 para as três diferentes cargas de trabalho aplicadas. Em todos os cenários, os valores de *p-value* foram inferiores a 0,05, o que indica uma tendência estatisticamente significativa nos dados. Além disso, os valores de S e a inclinação de Sen foram positivos para as cargas de trabalho média e alta, sugerindo uma tendência de crescimento nos dados. Por outro lado, para a carga de trabalho baixa, os valores de S e a inclinação de Sen foram negativos, indicando uma tendência diminuição nos dados. Portanto, com base nas análises estatísticas realizadas, podemos confirmar que o servidor experimentou uma perda de desempenho nas cargas de trabalho média e alta. Isso reforça a evidência de possíveis sinais de envelhecimento do software no ambiente utilizando o SQL Server. Vale destacar que, apesar de a degradação ser maior para a carga alta, o valor da inclinação foi menor para essa carga em comparação com a carga média. Isso pode ser devido aos picos encontrados na carga alta.

Tabela 28 – SQL Server - Teste de Mann-Kendall e Sen's slope para o tempo de resposta.

SGBD	**Carga**	***p-value***	**S**	**Sen's slope (ms/h)**	**Tendência**
SQL Server	Baixa	3,31e-05	-468	-0,0621	Decrescimento
	Média	2,2e-16	1,004e+03	12,67	Crescimento
	Alta	2,62e-05	4,75e+02	4,65	Crescimento

4.4.3 Análise dos Processos

Nesta seção, são descritas as análises dos processos no ambiente com o SQL Server. O principal objetivo ao realizar essas análises é identificar possíveis sinais de envelhecimento de software no ambiente, observando os processos do sistema operacional e do SGBD utilizado. Além disso, identificamos os cinco processos que apresentaram maior consumo de memória e revelamos aqueles que apresentaram maior aumento durante a execução dos experimentos, levando em consideração tanto o processo do SGBD quanto os demais.

4.4.3.1 Processos que mais consumiram Memória

Utilizando o *script* de monitoramento, durante a execução de cada teste, foram coletados os processos em execução no servidor. A partir desses dados, foi possível identificar os processos que mais estavam utilizando a memória RAM do servidor e verificar se as solicitações enviadas ao SQL Server estavam causando um aumento significativo na utilização de memória pelo SGBD. Para essa avaliação, conforme já apresentado anteriormente, empregou-se o último arquivo coletado pelo *script* de monitoramento de processos, uma vez que ele representa uma fase em que o ambiente já está comprometido devido à elevada carga de requisições do cliente, próximo das 48 horas de testes sob determinada carga.

É importante ressaltar que, a fim de classificar os processos e identificar os cinco que mais consumiram memória com base nos dados de monitoramento de cada carga de trabalho, foi feito um somatório dos valores de RSS dos processos relacionados à execução do SQL Server. Deste modo, o PID 0, denominado de "sqlserver"na Tabela 29 e nas Figuras 45, 46, 47, representa de forma única os processos do SQL Server já com os valores de RSS somados. Isso foi necessário para diferenciá-lo dos demais processos do Ubuntu, listados no arquivo coletado e identificar o consumo total do SQL Server.

A Tabela 29 apresenta a descrição dos cinco processos que mais consumiram memória em carga baixa, enquanto que a Figura 45 ilustra a quantidade de memória utilizada por esses processos. Podemos observar por meio da figura que o SQL Server foi o principal responsável pelo consumo de memória durante o experimento com essa carga, utilizando cerca de 995,7 MB. Os processos, "gnome-shell", "Xorg vti -dis", "tracker-store"e "snapd", aparecem em sequência com uma utilização de memória significativamente menor.

Tabela 29 – SQL Server - Descrição dos processos que mais consumiram memória - Carga Baixa.

PIDs	Descrição
0	**sqlserver** - Soma dos processos do SQL Server.
1215	**gnome-shell** - Ambiente de desktop GNOME.
1151	**Xorg vt1 -dis** - Servidor de exibição padrão.
3537	**tracker-store** - Processo de indexação de arquivos.
624	**snapd** - Processo que permite instalação e execução de aplicativos empacotados no formato 'snap'.

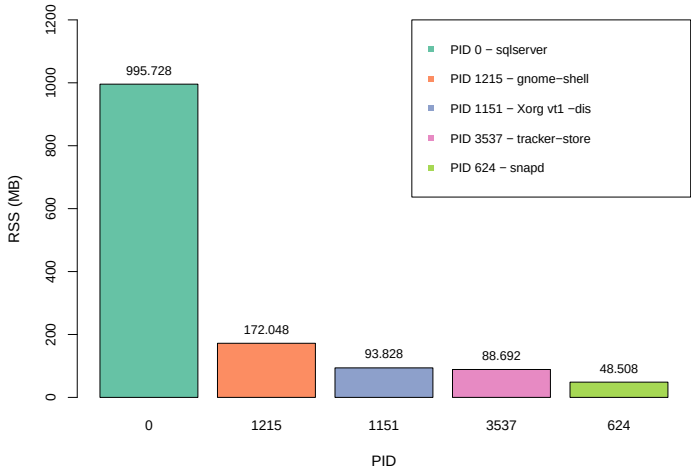

Figura 45 – SQL Server - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Baixa.

Na Figura 46, está representado o consumo de memória dos cinco processos que mais utilizaram RAM durante a execução em carga média. Apesar dos identificadores de processo serem diferentes, a Tabela 29 traz as descrições desses processos. Nesse contexto de carga, o SQL Server mantém sua posição como o principal consumidor de memória no servidor, utilizando 1104,1 MB. Em comparação com a situação de carga baixa, houve um aumento percentual de aproximadamente 10,8% no consumo de memória, indicando que o servidor teve que alocar mais memória RAM para atender às demandas da carga média.

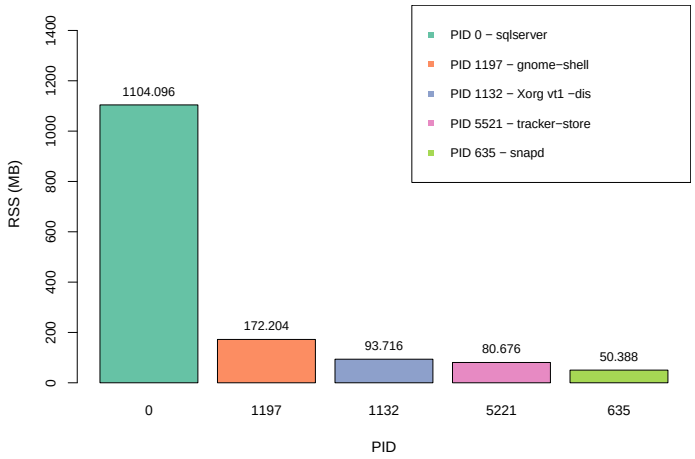

Figura 46 – SQL Server - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Média.

A Figura 47 apresenta os cinco processos que mais consumiram memória durante uma carga alta. Apesar dos identificadores de processo serem diferentes, a Tabela 29 fornece as descrições correspondentes a esses processos. Os processos responsáveis pela execução do SQL Server, PID 0, ocuparam novamente a primeira posição, demonstrando um consumo de memória significativamente maior em comparação com os demais. Durante a operação com carga alta, o SQL Server utilizou aproximadamente 1245 MB de memória, alocando mais RAM do que nas situações de carga baixa e média.

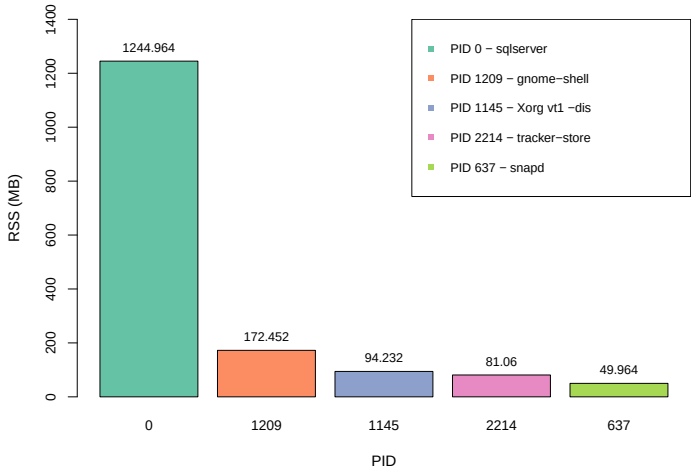

Figura 47 – SQL Server - Quantidade de memória utilizada pelos cinco processos que mais consumiram memória - Carga Alta.

Após a análise detalhada, fica claro que as diferentes cargas de trabalho aplicadas destacaram o SQL Server como o principal consumidor de memória RAM, ultrapassando consideravelmente os demais processos em todas as cargas aplicadas. Além disso, de forma consistente, os processos "gnome-shell", "Xorg vt1 -dis", "tracker-store"e "snapd"completaram o top 5 processos que mais consumiram memória RAM. Esses resultados podem sugerir que o SQL Server possa desempenhar um papel substancial nos efeitos de envelhecimento do software no ambiente analisado.

4.4.3.2 Processos que mais cresceram juntamente com o SGBD

Nesta análise, em contraste com a avaliação realizada na seção 4.4.3.1, foram identificados os processos que apresentaram o maior aumento no consumo de memória RAM durante os testes. Conforme já explicado anteriormente, para calcular a porcentagem de crescimento, utilizamos os valores de USS do arquivo gerado pelo *script* de monitoramento de processos correspondente à primeira hora de teste, assim como o arquivo referente à última hora de teste. A análise abrange os quatro processos que registraram o maior aumento no consumo de memória, além do processo que representa a soma dos processos relacionados à execução do SQL Server.

A Tabela 30 exibe a classificação dos processos com o maior aumento no uso de memória em situações de carga baixa, enquanto a Tabela 31 fornece a descrição desses processos. Nos três primeiros lugares, estão os processos associados ao tracker, sendo que o "tracker-store"se destaca ao apresentar um crescimento de 415%. Observa-se o processo que representa a execução do SQL Server ocupando a 15ª posição no ranking, com um aumento mais modesto de apenas 1%. No entanto, é possível identificar um acúmulo, o que suscita suspeitas sobre um possível envelhecimento do software no sistema no SGBD.

Tabela 30 – SQL Server - Processos que mais cresceram juntamente com o SGBD - Carga Baixa.

Ranking	PID	Processo	Crescimento
1º	3337	tracker-store	415%
2º	1580	tracker-miner	51%
3º	3582	tracker-extract	28%
4º	624	snapd	25%
15º	0	sqlserver	1%

Tabela 31 – SQL Server - Descrição dos processos que mais cresceram juntamente com o SGBD - Carga Baixa

Processo	Descrição
tracker-store	Processo de indexação de arquivos
traker-miner	Processo responsável por indexar e pesquisar arquivos no sistema de arquivos.
tracker-extract	Processo que indexa e extrai metadados de arquivos no sistema de arquivos
snapd	Processo que permite instalação e execução de aplicativos empacotados no formato 'snap'
sqlserver	Soma dos processos do SQL Server

A Tabela 32 apresenta a classificação dos processos com maior aumento no uso de memória durante a execução do teste com carga média. Embora possuam identificadores de processo diferentes, as descrições correspondentes podem ser encontradas na Tabela 31, com exceção do processo "init splash", responsável pela inicialização Ubuntu. Novamente, o processo "tracker-store"liderou a lista com um crescimento de 366%, seguido pelo processo "init splash"com um aumento de 153%. Em relação aos outros processos ranqueados, o crescimento do SQL Server foi relativamente baixo, ocupando a 9ª posição. No entanto, comparado ao mesmo processo durante a execução com a carga baixa, houve um aumento significativo, indo de 1% para 3% durante a carga média.

Tabela 32 – SQL Server - Processos que mais cresceram juntamente com o SGBD - Carga Média.

Ranking	PID	Processo	Crescimento
1º	5521	tracker-store	366%
2º	1	init splash	153%
3º	635	snapd	30%
4º	5529	tracker-extract	21%
9º	0	sqlserver	3%

Na Tabela 33, estão listados os processos que registraram os maiores aumentos no consumo de memória durante os testes de carga alta. Novamente, o processo "tracker-store"ocupou a primeira posição no ranking, com um crescimento de 363%. Já o acúmulo no SQL Server foi considerável maior quando comparado com a carga baixa e média, apresentando um crescimento de 11% e alcançando a quinta posição no ranking. Completando essa lista, encontramos os processos "snapd", "tracker-extract"e "init splash".

Tabela 33 – SQL Server - Processos que mais cresceram juntamente com o SGBD - Carga Alta.

Ranking	PID	Processo	Crescimento
1º	2214	tracker-store	363%
2º	637	snapd	33%
3º	222	tracker-extract	22%
4º	1	init splash	16%
5º	0	sqlserver	11%

De forma semelhante aos outros ambientes de bancos de dados estudados, o processo "tracker-store"ocupou a primeira posição em todos os casos, sugerindo que esse processo desempenha um papel relevante na degradação do ambiente. No entanto, os processos relacionados ao SQL Server não podem ser negligenciados, uma vez que também têm impacto na degradação do ambiente adotado.

4.4.3.3 Análise dos Processos do SQL Server

Como evidenciado pela análise anterior, os processos relacionados a bancos de dados podem ser um dos fatores responsáveis pela degradação dos ambientes do sistema adotado. Nesta análise, exploraremos mais detalhadamente a investigação desses processos.

A Figura 48 apresenta a evolução dos processos de execução do SQL Server sob uma carga de trabalho baixa. Notam-se várias oscilações ao longo do tempo, e não é possível identificar claramente uma tendência definida, seja de crescimento ou decrescimento. No entanto, ao compararmos o estado inicial e final do valor de USS, observa-se um aumento no valor de USS. A análise estatística desses dados é fundamental para determinar uma tendência significativa, na qual os testes de Mann-Kendall e inclinação desempenham um papel crucial.

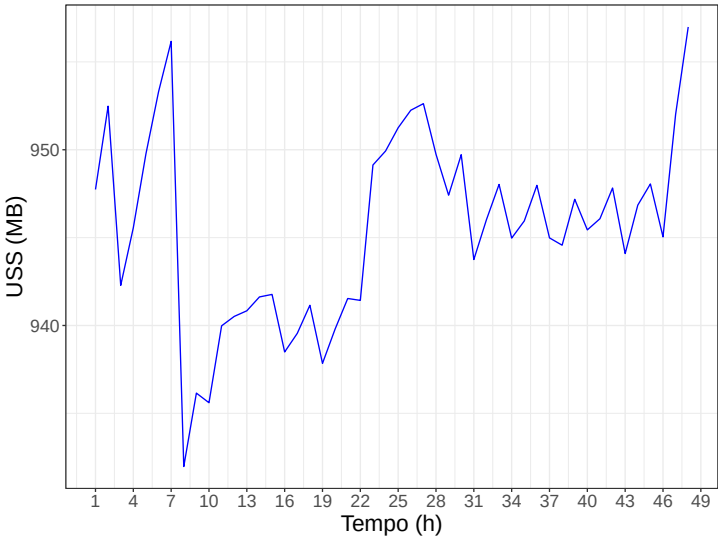

Figura 48 – Crescimento dos processos do SQL Server ao longo do tempo de teste - Carga Baixa

Na Figura 49, é apresentado como a memória utilizada pelo SQL Server evolui durante a aplicação da carga média. Durante essa carga, podemos notar um nível constante de uso de memória, que se mantém consistentemente acima de 1000 MB. Durante a primeira hora de teste, o consumo de memória foi cerca de 1031 MB, aumentando para aproximadamente 1065 MB ao final do teste. Além disso, ao analisar os dados ao longo do tempo, percebemos uma tendência de crescimento, indicando um gradual acúmulo na utilização de memória. Esse padrão levanta suspeitas de possíveis problemas de envelhecimento de software no SGBD.

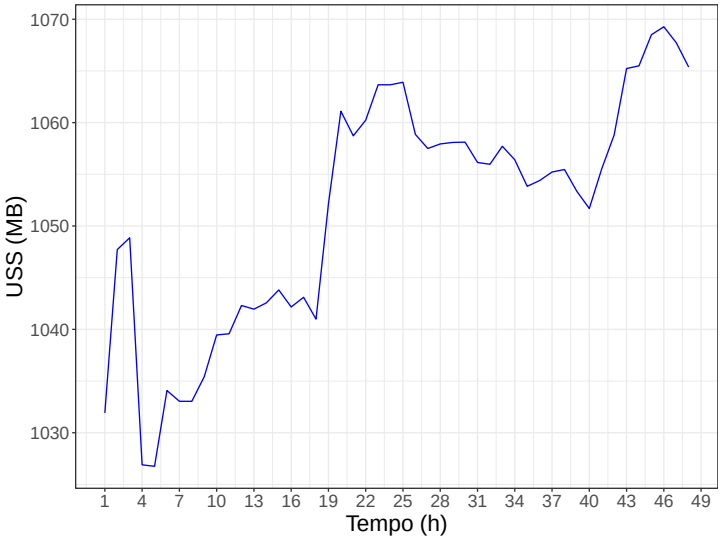

Figura 49 – Crescimento dos processos do SQL Server ao longo do tempo de teste - Carga Média

Ao analisar os experimentos realizados com uma carga alta (ver a Figura 50), é evidente uma notável tendência de crescimento nos dados. Inicialmente, a alocação de memória pelo SQL Server foi em torno de 1089 MB, aumentando para cerca de 1206 MB ao término do teste. Até aproximadamente as 19 horas, o aumento gradual não é tão proeminente; entretanto, a partir desse ponto, observa-se um rápida crescimento seguido por uma relativa estabilização no uso da memória. Novamente, é crucial recorrer a testes estatísticos para determinar a natureza dessa tendência nos dados e calcular uma inclinação, seja ela ascendente ou descendente.

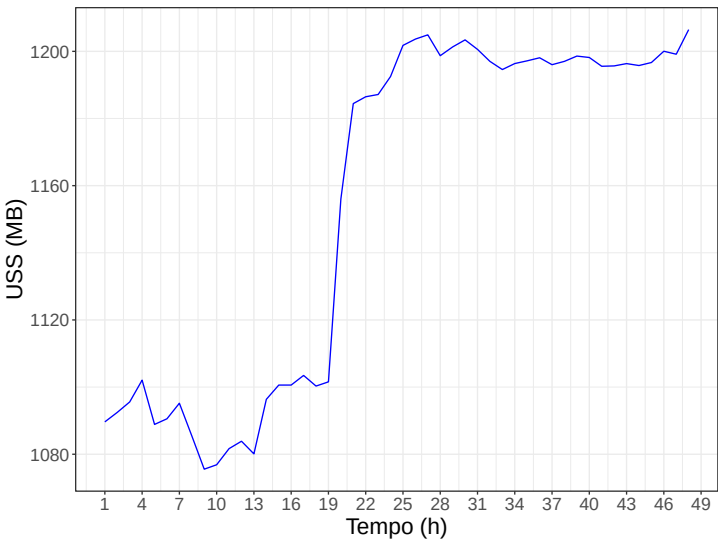

Figura 50 – Crescimento dos processos do SQL Server ao longo do tempo de teste - Carga Alta

A Tabela 34 apresenta os resultados advindos dos testes estatísticos de Mann-Kendall e da inclinação de Sen's slope. Esses testes foram aplicados aos dados relativos ao consumo de memória pelos processos do SQL Server. Com a análise dos resultados foi revelado que todos os valores de *p-value* encontram-se abaixo de 0,05 para todas as diferentes cargas utilizadas, o que sustenta a presença de uma tendência estatisticamente significativa nos dados. Além disso, destaca-se que tanto os valores de S quanto a magnitude das inclinações foram positivos. Esse cenário aponta para uma clara tendência estatística de crescimento no consumo de memória pelo SQL Server durante a aplicação das diversas cargas. Notavelmente, durante a fase de maior carga, é evidente que o valor de inclinação de Sen supera os demais. Isso indica que a carga de trabalho tem impacto na degradação do ambiente, onde cargas maiores resultaram em inclinações mais elevadas.

Tabela 34 – SQL Server - Teste de Mann-Kendall e Sen's slope para os processos do SGBD.

SGBD	**Carga**	***p-value***	**S**	**Sen's slope (MB/h)**	**Tendência**
SQL Server	Baixa	0,042	2,3e+02	0,165	Crescimento
	Média	3,95e-08	6,19e+02	0,7	Crescimento
	Alta	7,2e-09	6,52e+02	2,83	Crescimento

4.5 Comparativos dos Efeitos do Envelhecimento de Software

Nesta seção, é realizado um comparativo dos resultados obtidos durante as análises de envelhecimento de software realizada com cada SGBD. Primeiramente, realizamos um comparativo do consumo de memória RAM. Em seguida, o comparativo dos resultados do tempo de resposta. Por fim, o comparativo da análise de processos.

4.5.1 Comparativo do Consumo de Memória

Na Tabela 35 estão consolidados os resultados dos dados relativos ao uso de memória com as cargas e os SGBDs empregados nos experimentos. Isso nos possibilitou avaliar o ambiente mais impactado pelos efeitos do envelhecimento do software, além de obter uma compreensão da degradação do servidor perante diversas cargas de trabalho e diferentes

SGBDs utilizados. Para determinar qual SGBD sofreu mais com o envelhecimento de software ao longo do tempo, foi considerada a taxa de crescimento da utilização média de memória ao longo do tempo, representada pelo valor da inclinação de Sen's slope.

Com base nos dados apresentados na Tabela 35, é notável que, em situação de carga baixa, o SQL Server consumiu mais memória RAM do que os outros SGBDs. No entanto, destaca-se a notável semelhança nos valores das inclinações (representadas pelo Sen's slope). Ao analisarmos as inclinações para a carga baixa, fica claro que elas são muito próximas entre si. Deste modo, podemos dizer que todos os SGBDs estão sofrendo um crescimento semelhante na utilização média de memória ao longo do tempo.

Já em carga média, é mais fácil notar que o ambiente com o SQL Server foi o mais afetado pelo envelhecimento ao longo do tempo. Isso fica ressaltado na tabela, onde se destaca por possuir a inclinação de Sen's slope mais elevada entre os diferentes bancos, com um valor de 0,032. Além disso, esse ambiente também apresentou o uso médio mais alto de memória, atingindo 25,6%. Em contrapartida, os ambientes que utilizaram os SGBDs MariaDB, PostgreSQL e MySQL exibem inclinações de Sen's slope semelhantes, variando de 0,018 a 0,024, sendo que o PostgreSQL apresentou o menor valor.

Na análise da alta carga, o ambiente com o SQL Server destacou-se novamente como o mais impactado pelo envelhecimento de software ao longo do tempo. Isso ficou evidente tanto pelo uso médio mais elevado de memória, atingindo 26,9%, quanto pela maior magnitude de inclinação de Sen's slope, que registrou um valor de 0,06. O ambiente utilizando o MySQL posicionou-se como o segundo mais afetado, apresentando uma inclinação de Sen's de 0,026 e uso médio de memória de 18%. Por outro lado, os ambientes que empregaram o MariaDB e o PostgreSQL tiveram inclinações de Sen's slope idênticas, ambas com valor de 0,017, apresentando níveis distintos de uso médio de memória.

Tabela 35 – Tabela comparativa de envelhecimento para o uso de RAM.

Carga	SGBD	Uso médio de memória	Sen's slope (%/h)	Tendência
Baixa	MySQL	16,9%	0,024	Crescimento
	MariaDB	12,5%	0,024	Crescimento
	PostgreSQL	11,9%	0,02	Crescimento
	SQL Server	23,9%	0,02	Crescimento
Média	MySQL	17,6%	0,023	Crescimento
	MariaDB	12,9%	0,024	Crescimento
	PostgreSQL	13,6%	0,018	Crescimento
	SQL Server	25,6%	0,032	Crescimento
Alta	MySQL	18%	0,026	Crescimento
	MariaDB	12,9%	0,017	Crescimento
	PostgreSQL	15%	0,017	Crescimento
	SQL Server	26,9%	0,06	Crescimento

4.5.2 Comparativo do Tempo de Resposta

Na Tabela 36 estão consolidados os resultados relativos ao tempo de resposta, considerando as diferentes cargas e os SGBDs utilizados nos experimentos. Essa consolidação nos permitiu avaliar os ambientes que sofreram as maiores degradações nos desempenhos ao longo do tempo. Para identificar tais ambientes, utilizamos a taxa de crescimento. Essa taxa é determinada pela inclinação do coeficiente de Sen, o que permitiu identificar qual ambiente sofreu a maior deterioração de desempenho ao longo do período analisado. Adicionalmente, fornecemos os tempos de resposta médios de cada experimento como parte dos resultados apresentados.

Ao analisar os dados relacionados ao uso da carga de baixa, fica claramente evidente que todos os sistemas de gerenciamento de banco de dados apresentaram uma tendência de diminuição no tempo de resposta ao longo do período de análise, uma vez que os valores de Sen's slope foram todos negativos. Isso nos indica que não houve uma perda de desempenho durante a execução dessa carga de trabalho ao longo do tempo. Portanto, podemos concluir que os possíveis efeitos do envelhecimento de software não resultaram em uma perda de desempenho em nenhum dos ambientes quando a carga de trabalho de baixa intensidade foi aplicada.

Com o uso da carga média, é possível observar comportamentos distintos ao longo do tempo entre os ambientes. O MySQL se destaca ao demonstrar um aumento notável no tempo de resposta médio, alcançando 402 ms, em conjunto com uma inclinação de Sen's slope positiva de 4,66 ms/h, indicando um crescimento gradual no tempo de resposta e perda desempenho ao longo do tempo. Por outro lado, o MariaDB apresenta uma tendência contrária, com uma inclinação de -0,258 ms/h, não sofrendo perda desempenho com o uso dessa carga. Enquanto isso, o PostgreSQL e o SQL Server revelam tendências de crescimento em seus tempos de resposta médio, sustentadas por inclinações positivas de 0,48 ms/h e 12,67 ms/h, respectivamente, com tempos médios de resposta de 680 ms e 645 ms. Com base nas análises, entre os SGBDs listados, o SQL Server experimentou o maior aumento no tempo de resposta, refletindo um possível impacto negativo no desempenho ao longo do tempo.

Em cenários de alta carga, fica evidente que todos os ambientes experimentaram uma perda no desempenho, como refletido na tendência de crescimento nos dados. Dentre eles, o MySQL se destaca por apresentar a maior deterioração do desempenho, evidenciada por uma inclinação positiva de Sen's slope de 12,18 ms/h, juntamente com um tempo de resposta médio de 1623 ms. Similarmente, o MariaDB também mostra uma tendência de crescimento, com uma inclinação de 5,75 ms/h e um tempo de resposta médio de 1484 ms. Adicionalmente, o PostgreSQL exibe o maior tempo de resposta médio, embora seu crescimento ao longo do tempo seja modesto, com uma inclinação de apenas 0,0031 ms/h. O SQL Server também enfrentou os efeitos potenciais do envelhecimento de software, demonstrando uma perda no desempenho com um crescimento de 4,65 ms/h e um tempo de resposta médio de 1583 ms.

Tabela 36 – Tabela comparativa de envelhecimento para o Tempo de Resposta.

Carga	SGBD	Tempo de Resposta Médio	Sen's slope (ms/h)	Tendência
Baixa	MySQL	47 ms	-0,0037	Decrescimento
	MariaDB	52 ms	-0,037	Decrescimento
	PostgreSQL	31 ms	-0,0005	Decrescimento
	SQL Server	48 ms	-0,0621	Decrescimento
Média	MySQL	402 ms	4,66	Crescimento
	MariaDB	382 ms	-0,258	Decrescimento
	PostgreSQL	680 ms	0,48	Crescimento
	SQL Server	645 ms	12,67	Crescimento
Alta	MySQL	1623 ms	12,18	Crescimento
	MariaDB	1484 ms	5,75	Crescimento
	PostgreSQL	1932 ms	0,0031	Crescimento
	SQL Server	1583 ms	4,65	Crescimento

4.5.3 Comparativo da Análise de Processos

Na Tabela 37 estão resumidos os resultados relativos à utilização de memória pelos processos dos SGBDs. Para esta avaliação, especificamente, foi adotado o valor de USS. A análise foi conduzida levando em consideração diversas cargas de trabalho e os SGBDs empregados nos experimentos. A consolidação desses dados nos possibilitou identificar quais SGBDs demonstram os maiores aumentos ao longo do tempo em termos de consumo de memória. Isso foi alcançado ao avaliar o grau de inclinação da linha de tendência de Sen.

Na condição de carga baixa, o SQL Server se destacou ao exibir o USS médio mais elevado, alcançando 945,5 MB. Isso veio acompanhado por uma inclinação positiva na linha de tendência de Sen, registrando 0,165 MB/h. Esse resultado indica que o SQL Server foi o SGBD mais impactado pelos possíveis efeitos de envelhecimento de software ao longo do tempo, considerando essa carga específica. Tanto o MySQL quanto o MariaDB também revelaram um aumento no USS médio, com inclinações de Sen's slope de 0,0011 MB/h e 0,0001 MB/h, respectivamente. Isso sugere que esses sistemas também foram afetados por efeitos sutis de envelhecimento de software, embora em menor medida do que o SQL Server. No entanto, nessa análise em nível de processo, o PostgreSQL não apresentou uma tendência discernível nos dados. A inclinação de Sen's slope foi igual a

zero, o que sugere a ausência de vazamento de memória ao longo do tempo para essa carga específica.

Em carga de trabalho média, percebe-se que tanto o MySQL quanto o MariaDB demonstraram um crescimento no USS médio, apresentando inclinações de Sen's slope positivas de 0,0046 MB/h e 0,006 MB/h, respectivamente. Já o PostgreSQL revela uma tendência estável, refletida em sua inclinação de Sen's slope igual a zero e USS médio de 92,3 MB. O SQL Server se destaca por apresentar o maior crescimento, com uma inclinação de Sen's slope de 0,7 MB/h e um USS médio de 1051,5 MB.

Em contraste, sob cargas de trabalho mais intensas, as tendências persistem. O MySQL e o MariaDB continuam a apresentar crescimento, mas com inclinações de Sen's slope de 0,017 MB/h e 0,159 MB/h, respectivamente. O PostgreSQL, contudo, evidencia uma mudança para a direção oposta, com uma inclinação de Sen's slope negativa de -0,065 MB/h e um USS médio de 183,1 MB. Por sua vez, o SQL Server continua a apresentar o maior crescimento, exibindo uma inclinação de Sen's slope de 2,83 MB/h e um USS médio de 1154,6 MB. Vale destacar que em todas as cargas utilizadas, o SQL Server foi o SGBD mais afetado pelo fenômeno de envelhecimento de software.

Tabela 37 – Tabela comparativa de envelhecimento dos processos dos SBGDs.

Carga	**SGBD**	**USS Médio**	**Sen's slope (MB/h)**	**Tendência**
Baixa	MySQL	407,4 MB	0,0011	Crescimento
	MariaDB	98,7 MB	0,0001	Crescimento
	PostgreSQL	14,8 MB	0	Ausente
	SQL Server	945,5 MB	0,165	Crescimento
Média	MySQL	452,9 MB	0,0046	Crescimento
	MariaDB	110,7 MB	0,006	Crescimento
	PostgreSQL	92,3 MB	0	Ausente
	SQL Server	1051,5 MB	0,7	Crescimento
Alta	MySQL	470,4 MB	0,017	Crescimento
	MariaDB	118,3 MB	0,159	Crescimento
	PostgreSQL	183,1 MB	-0,065	Decrescimento
	SQL Server	1154,6 MB	2,83	Crescimento

4.6 Ameaças à Validade

Nesta seção são descritas as ameaças à validade deste trabalho.

Generalização do Estudo Experimental: Assim como ocorre em todo estudo experimental, nossos resultados podem não ser facilmente generalizados para outros ambientes de bancos de dados. No entanto, há uma convicção de que os ambientes escolhidos guardam notáveis semelhanças com aqueles comuns em empresas de porte médio e pequeno. Além disso, o método de análise detalhado neste artigo pode ser facilmente ajustado para encaixar em qualquer outro ambiente de banco de dados, permitindo que pesquisadores interessados reproduzam nosso estudo com a mesma abordagem.

Restrição na Duração e Quantidade de Execuções Experimentais: Os experimentos relacionados ao envelhecimento de software requerem uma quantidade significativa de tempo, e sua duração está naturalmente sujeita a limitações, assim como em qualquer estudo experimental. Ainda que testes com uma duração de 48 horas para cada configuração de carga, conforme descritos na Tabela 2, tenham sido conduzidos, não é possível assegurar a identificação de todas as possíveis tendências de degradação.

Representatividade da Carga de Trabalho Utilizada: As cargas de trabalho escolhidas estão alinhadas com o *schema* de banco de dados ilustrado na Figura 2. Consequentemente, é plausível que o leitor obtenha resultados distintos dos nossos, ao adotar um *schema* ou uma consulta diferente. Adicionalmente, as cargas de trabalho exploradas neste estudo compreendem três níveis de intensidades constantes (baixa, média e alta), os quais podem não refletir os comportamentos usuais dos usuários em ambientes reais. Considerando que o propósito primordial deste trabalho é identificar tendências de envelhecimento de software nos ambientes adotados, não consideramos cargas de trabalho variáveis nos experimentos.

4.7 Considerações Finais

Neste capítulo, foram expostos os resultados provenientes dos experimentos conduzidos. A partir das análises realizadas nos dados referentes ao consumo de memória, tempo de resposta e processos, foi possível traçar considerações pertinentes acerca do envelhecimento de software nos ambientes dos SGBDs estudados.

Primeiramente, no que se refere ao consumo de memória, observa-se que o ambiente

com o SQL Server emerge como o mais suscetível aos efeitos do envelhecimento de software ao longo do tempo. Em todas as situações, especialmente em cargas médias e altas, o ambiente do SQL Server utilizou mais RAM do servidor. Além disso, apresentou inclinações de Sen's slope mais elevadas, indicando um crescimento mais pronunciado na utilização da memória em comparação com outros ambientes. Essa tendência destaca um possível desafio para o SQL Server em manter a eficiência operacional com o passar do tempo.

Quanto ao tempo de resposta, observam-se variações significativas nos diferentes ambientes de SGBDs. O SQL Server, destaca-se especialmente em cargas médias, com o segundo maior tempo de resposta médio e maior inclinação de Sen's slope. O PostgreSQL, por outro lado, revelou ser o menos afetado, apresentando tendências mais estáveis de decrescimento e crescimento, com inclinações de Sen's slope próximas a zero. O MariaDB, por sua vez, sugere uma capacidade de manter ou aprimorar o desempenho ao longo do tempo, principalmente em cargas de trabalho mais leves. Já o MySQL se destacou negativamente em carga alta, exibindo uma inclinação de Sen's slope considerável.

Na análise específica dos processos associados aos bancos, os resultados confirmam a tendência observada nas análises anteriores. O SQL Server demonstrou consistentemente o maior crescimento no uso médio de memória e níveis mais elevados de utilização da mesma, indicando uma maior propensão a vazamentos de memória ou ineficiências associadas ao envelhecimento do software, seguido pelo MySQL. Em contrapartida, o PostgreSQL exibiu uma tendência mais estável, apresentando inclinações de Sen's slope iguais a zero nas cargas baixa e média e valor negativo em carga alta, o que pode indicar uma menor propensão a vazamentos de memória ao longo do tempo.

5 Conclusão e Trabalhos Futuros

Neste trabalho foi realizada uma análise comparativa dos sintomas de envelhecimento de software em ambientes que envolveram os sistemas de gerenciamento de banco de dados MySQL, MariaDB, PostgreSQL e SQL Server. Para avaliar as tendências, empregamos o teste estatístico de Mann-Kendall e o estimador Sen's slope. Os SGBDs utilizados nos experimentos são amplamente conhecidos e estão entre os mais utilizados no mundo. Porém, os resultados obtidos neste trabalho mostram que existem indicativos de envelhecimento de software nos ambientes adotados, revelados pelo crescimento do consumo de memória e a degradação ao longo do tempo.

Também analisamos os processos dos ambientes estudados, em todos os experimentos realizados o "tracker-store"apareceu como primeiro colocado entre os processos que tiveram o maior aumento no consumo de memória, sendo o processo do ambiente Linux que mais contribuiu para o envelhecimento de software. Além disso, na análise realizada, observou-se vazamento de memória em processos relacionados aos SGBDs, com destaque para o SQL Server, que apresentou os maiores acúmulos de consumo de memória. Por outro lado, ao analisar o crescimento dos processos ao longo do tempo, não identificamos uma tendência de aumento significativo no consumo de memória pelo PostgreSQL por meio de testes estatísticos. Isso sugere que o envelhecimento do software foi mais evidente no ambiente em geral do que especificamente no PostgreSQL.

Quanto à perda de desempenho, podemos afirmar que não foram detectadas em carga baixa, mas em carga média e alta, nossas suspeitas foram confirmadas estatisticamente na maioria dos casos. Em outras palavras, em situações de carga média e alta, observamos um aumento no tempo de resposta ao longo do tempo, indicando uma degradação no desempenho. Os resultados obtidos sugerem que o SQL Server foi o SGBD mais afetado pelo envelhecimento de software. Isso se deve não apenas ao fato de que ele consome a maior quantidade de RAM para atender às demandas, mas também porque apresentou as maiores magnitudes de crescimento nos dados de uso de memória ao longo do tempo.

Esses resultados destacam a importância de monitorar e gerenciar o envelhecimento de software em ambientes de banco de dados. É relevante destacar que o MySQL, o MariaDB, o PostgreSQL e o SQL Server são softwares amplamente adotados em ambientes de produção. Portanto, análises de envelhecimento de software são fundamentais para

garantir a eficiência, segurança e confiabilidade contínuas desses sistemas. As análises conduzidas neste estudo, bem como a metodologia utilizada, podem ser úteis para profissionais de Tecnologia da Informação e desenvolvedores no enfrentamento de problemas relacionados ao envelhecimento de software em ambientes de banco de dados.

5.1 Contribuições

São apresentadas nesta seção as contribuições desta pesquisa:

- O método de análise empregado neste estudo, juntamente com o modelo e toda a arquitetura experimental, pode ser facilmente ajustado para outros ambientes e outros SGBDs.
- Os resultados obtidos têm potencial significativo tanto para usuários quanto para especialistas em Tecnologia da Informação que necessitam tomar decisões a fim de superar os desafios ligados ao envelhecimento de software;
- Artigo publicado na X Escola Regional de Informática de Goiás (ERI-GO) (COUTO et al., 2022): *Herderson Couto, Francisco Airton Silva, Gustavo Callou e Ermeson Andrade. Uma Abordagem Experimental para Avaliar o Desempenho do Banco de Dados Open-Source PostgreSQL* - Artigo fruto do início da pesquisa, quando foi avaliado o desempenho do PostgreSQL diante de diferentes cargas de trabalho;
- Artigo publicado na revista IEEE Latin America Transactions (COUTO et al., 2023): *Herderson Couto, Ermeson Andrade, Francisco Airton Silva e Gustavo Callou. Analysis of Software Aging in a Database Environment* - Artigo que investigou os sintomas de envelhecimento de software em um ambiente de SGBD com o PostgreSQL, cujo parte dos resultados estão presentes nesta dissertação; e
- Submissão de artigo na revista IEEE Transactions on Reliability: *Herderson Couto, Fumio Machida, Gustavo Callou e Ermeson Andrade. A Comparative Analysis of Software Aging in Relational Database System Environments - Artigo submetido em 31 de julho de 2023 e que encontra-se em fase de revisão.* O referido artigo foi desenvolvido em colaboração com o Professor Fumio Machida do Department of Computer Science da University of Tsukuba. No mencionado artigo, foram identificados e comparados os sintomas de envelhecimento de software em ambientes que utilizam os SGBDs MySQL e SQL Server. Os resultados obtidos compõem esta

dissertação.

5.2 Trabalhos Futuros

Como trabalhos futuros, pretendemos realizar os experimentos por um tempo maior de execução, conduzir testes com um *schema* de banco de dados mais complexo, além de utilizar outras operações e diferentes cargas de trabalho. Além disso, considerar mais parâmetros para a análise de envelhecimento de software, tais como o uso da CPU e consumo de energia. Também está nos planos realizar experimentos com outros sistemas operacionais e diferentes SGBDs, tanto relacionais como não relacionais, e explorar experimentos com hardware de maior capacidade, incluindo uma maior quantidade de memória RAM e/ou um servidor equipado com SSD. Estudos sobre o rejuvenescimento de software e a investigação aprofundada da influência dos demais processos do Linux no envelhecimento de software também estão planejados para serem realizados.

Referências

ANDRADE, E.; MACHIDA, F.; PIETRANTUONO, R.; COTRONEO, D. Software aging in image classification systems on cloud and edge. In: IEEE. **2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)**. [S.l.], 2020. p. 342–348.

_____. Memory degradation analysis in private and public cloud environments. In: IEEE. **2021 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)**. [S.l.], 2021. p. 33–39.

ANDRADE, E.; PIETRANTUONO, R.; MACHIDA, F.; COTRONEO, D. A comparative analysis of software aging in image classifiers on cloud and edge. **IEEE Transactions on Dependable and Secure Computing**, IEEE, 2021.

BOVENZI, A.; COTRONEO, D.; PIETRANTUONO, R.; RUSSO, S. On the aging effects due to concurrency bugs: A case study on mysql. In: IEEE. **2012 IEEE 23rd International Symposium on Software Reliability Engineering**. [S.l.], 2012. p. 211–220.

CASTELLI, V.; HARPER, R. E.; HEIDELBERGER, P.; HUNTER, S. W.; TRIVEDI, K. S.; VAIDYANATHAN, K.; ZEGGERT, W. P. Proactive management of software aging. **IBM Journal of Research and Development**, IBM, v. 45, n. 2, p. 311–332, 2001.

CODD, E. F. A relational model of data for large shared data banks. **Communications of the ACM**, ACM New York, NY, USA, v. 13, n. 6, p. 377–387, 1970.

COTRONEO, D.; NATELLA, R.; PIETRANTUONO, R.; RUSSO, S. A survey of software aging and rejuvenation studies. **ACM Journal on Emerging Technologies in Computing Systems (JETC)**, Acm New York, NY, USA, v. 10, n. 1, p. 1–34, 2014.

COUTO, H.; ANDRADE, E.; SILVA, F. A.; CALLOU, G. Analysis of software aging in a database environment. **IEEE Latin America Transactions**, v. 21, n. 7, p. 821–828, 2023.

COUTO, H.; SILVA, F.; CALLOU, G.; ANDRADE, E. Uma abordagem experimental para avaliar o desempenho do banco de dados open-source postgresql. In: **Anais da X Escola Regional de Informática de Goiás**. [S.l.]: SBC, 2022. p. 12–23.

CRAN. **Non-Parametric Trend Tests and Change-Point Detection**. 2023. <https://cran.r-project.org/web/packages/trend/index.html>. [Online; accessed 28-set-2023].

DATE, C. J. **Introdução a sistemas de bancos de dados**. [S.l.]: Elsevier Brasil, 2004.

DB-ENGINES. **DB-Engines Ranking**. 2023. <https://db-engines.com/en/ranking>. [Online; accessed 28-set-2023].

DIAS, D.; ANDRADE, E. Análise de envelhecimento de software em uma plataforma de blockchain. In: SBC. **Anais do V Workshop em Blockchain: Teoria, Tecnologias e Aplicações**. [S.l.], 2022. p. 40–53.

DYER, R. J. **Learning MySQL and MariaDB: Heading in the right direction with MySQL and MariaDB**. [S.l.]: "O'Reilly Media, Inc.", 2015.

FICCO, M.; PIETRANTUONO, R.; RUSSO, S. Aging-related performance anomalies in the apache storm stream processing system. **Future Generation Computer Systems**, Elsevier, v. 86, p. 975–994, 2018.

GARG, S.; MOORSEL, A. V.; VAIDYANATHAN, K.; TRIVEDI, K. S. A methodology for detection and estimation of software aging. In: IEEE. **Proceedings Ninth International Symposium on Software Reliability Engineering (Cat. No. 98TB100257)**. [S.l.], 1998. p. 283–292.

GROTTKE, M.; LI, L.; VAIDYANATHAN, K.; TRIVEDI, K. S. Analysis of software aging in a web server. **IEEE Transactions on reliability**, IEEE, v. 55, n. 3, p. 411–420, 2006.

GROTTKE, M.; MATIAS, R.; TRIVEDI, K. S. The fundamentals of software aging. In: IEEE. **2008 IEEE International conference on software reliability engineering workshops (ISSRE Wksp)**. [S.l.], 2008. p. 1–6.

GROVER, V.; TENG, J. T. An examination of dbms adoption and success in american organizations. **Information & Management**, Elsevier, v. 23, n. 5, p. 239–248, 1992.

HAUSWIRTH, M.; CHILIMBI, T. M. Low-overhead memory leak detection using adaptive statistical profiling. In: **Proceedings of the 11th international conference on Architectural support for programming languages and operating systems**. [S.l.: s.n.], 2004. p. 156–164.

HE, H. Tuning backfired? not (always) your fault: Understanding and detecting configuration-related performance bugs. In: **Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering**. [S.l.: s.n.], 2019. p. 1229–1231.

HEUSER, C. A. **Projeto de banco de dados: Volume 4 da Série Livros didáticos informática UFRGS**. [S.l.]: Bookman Editora, 2009.

HIPEL, K. W.; MCLEOD, A. I. **Time series modelling of water resources and environmental systems**. [S.l.]: Elsevier, 1994.

HUANG, Y.; KINTALA, C.; KOLETTIS, N.; FULTON, N. D. Software rejuvenation: Analysis, module and applications. In: IEEE. **Twenty-fifth international symposium on fault-tolerant computing. Digest of papers**. [S.l.], 1995. p. 381–390.

HUO, S.; ZHAO, D.; LIU, X.; XIANG, J.; ZHONG, Y.; YU, H. Using machine learning for software aging detection in android system. In: IEEE. **2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)**. [S.l.], 2018. p. 741–746.

ILIĆ, M.; KOPANJA, L.; ZLATKOVIĆ, D.; TRAJKOVIĆ, M.; ĆURGUZ, D. Microsoft sql server and oracle: Comparative performance analysis. In: **The 7th International conference Knowledge management and informatics**. [S.l.: s.n.], 2021. p. 33–40.

JANG, E.; JOHNSON, M.; BURNELL, E.; HEIMERL, K. Unplanned obsolescence: Hardware and software after collapse. In: **Proceedings of the 2017 workshop on computing within limits**. [S.l.: s.n.], 2017. p. 93–101.

JMETER™, A. **Apache JMeter - What can I do with it?** 2023. <https://jmeter.apache.org/>. [Online; accessed 28-set-2023].

KENDALL, M. G. Rank correlation methods. Griffin, 1948.

KLIMEK, B.; SKUBLEWSKA-PASZKOWSKA, M. Comparison of the performance of relational databases postgresql and mysql for desktop application. **Journal of Computer Sciences Institute**, v. 18, p. 61–66, 2021.

MACÊDO, A.; FERREIRA, T. B.; MATIAS, R. The mechanics of memory-related software aging. In: IEEE. **2010 IEEE Second International Workshop on Software Aging and Rejuvenation**. [S.l.], 2010. p. 1–5.

MACHIDA, F.; XIANG, J.; TADANO, K.; MAENO, Y. Aging-related bugs in cloud computing software. In: IEEE. **2012 IEEE 23rd international symposium on software reliability engineering workshops**. [S.l.], 2012. p. 287–292.

_____. Software life-extension: a new countermeasure to software aging. In: IEEE. **2012 IEEE 23rd International Symposium on Software Reliability Engineering**. [S.l.], 2012. p. 131–140.

_____. Lifetime extension of software execution subject to aging. **IEEE Transactions on Reliability**, IEEE, v. 66, n. 1, p. 123–134, 2016.

MACKALL, M. **smem - Report memory usage with shared memory divided proportionally.** 2008–2009. <https://linux.die.net/man/8/smem>. [Online; accessed 28-set-2023].

MANN, H. B. Nonparametric tests against trend. **Econometrica: Journal of the econometric society**, JSTOR, p. 245–259, 1945.

MARIADB. **MariaDB**. 2023. <https://mariadb.org/>. [Online; accessed 28-set-2023].

MATIAS, R.; BARBETTA, P. A.; TRIVEDI, K. S.; FILHO, P. J. F. Accelerated degradation tests applied to software aging experiments. **IEEE Transactions on reliability**, IEEE, v. 59, n. 1, p. 102–114, 2009.

MATOS, R.; ARAUJO, J.; ALVES, V.; MACIEL, P. Characterization of software aging effects in elastic storage mechanisms for private clouds. In: IEEE. **2012 IEEE 23rd International Symposium on Software Reliability Engineering Workshops**. [S.l.], 2012. p. 293–298.

MCLEOD, A. I. Kendall rank correlation and mann-kendall trend test. **R Package Kendall**, Western Univ., v. 602, p. 1–10, 2005.

MELO, C.; OLIVEIRA, F.; DANTAS, J.; ARAUJO, J.; PEREIRA, P.; MACIEL, R.; MACIEL, P. Performance and availability evaluation of the blockchain platform hyperledger fabric. **The Journal of Supercomputing**, Springer, p. 1–23, 2022.

MILANI, A. **MySQL-guia do programador**. [S.l.]: Novatec Editora, 2007.

_____. **PostgreSQL-Guia do Programador**. [S.l.]: Novatec Editora, 2008.

MINHAS, U. F.; RAJAGOPALAN, S.; CULLY, B.; ABOULNAGA, A.; SALEM, K.; WARFIELD, A. Remusdb: Transparent high availability for database systems. **The VLDB Journal**, Springer, v. 22, n. 1, p. 29–45, 2013.

MISTRY, R.; MISNER, S. **Introducing Microsoft SQL Server 2014**. [S.l.]: Microsoft Press, 2014.

MYSQL. **MySQL**. 2023. <https://www.mysql.com/>. [Online; accessed 28-set-2023].

NETER, J.; KUTNER, M. H.; NACHTSHEIM, C. J.; WASSERMAN, W. et al. Applied linear statistical models. Irwin Chicago, 1996.

OKAMURA, H.; ZHENG, J.; DOHI, T. A statistical framework on software aging modeling with continuous-time hidden markov model. In: IEEE. **2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)**. [S.l.], 2017. p. 114–123.

OLIVEIRA, F.; ARAUJO, J.; MATOS, R.; MACIEL, P. Software aging in container-based virtualization: an experimental analysis on docker platform. In: IEEE. **2021 16th Iberian Conference on Information Systems and Technologies (CISTI)**. [S.l.], 2021. p. 1–7.

PARNAS, D. L. Software aging. In: IEEE. **Proceedings of 16th International Conference on Software Engineering**. [S.l.], 1994. p. 279–287.

PIETRANTUONO, R.; RUSSO, S. A survey on software aging and rejuvenation in the cloud. **Software Quality Journal**, Springer, v. 28, n. 1, p. 7–38, 2020.

PIRES, C. E.; NASCIMENTO, R. O.; SALGADO, A. C. Comparativo de desempenho entre bancos de dados de código aberto. **Escola Regional de Banco de Dados, Anais da ERBD06, Porto Alegre**, 2006.

POLITOWSKI, C.; MARAN, V. Comparação de performance entre postgresql e mongodb. **X Escola Regional de Banco de Dados. SBC**, p. 1–10, 2014.

POSTGRESQL. **PostgreSQL**. 2023. <https://www.postgresql.org/about/>. [Online; accessed 28-set-2023].

QIAO, Y.; ZHENG, Z.; QIN, F. An empirical study of software aging manifestations in android. In: IEEE. **2016 IEEE international symposium on software reliability engineering workshops (ISSREW)**. [S.l.], 2016. p. 84–90.

SANTOS, B.; ENDO, P. T.; SILVA, F. A. Uma avaliação de desempenho de contêineres docker executando diferentes sgbds relacionais. In: SBC. **Anais do XVIII Workshop em Desempenho de Sistemas Computacionais e de Comunicação**. [S.l.], 2019.

SELENIC. **smem memory reporting tool.** 2023. <https://www.selenic.com/smem/>. [Online; accessed 27-out-2023].

SEN, P. K. Estimates of the regression coefficient based on kendall's tau. **Journal of the American statistical association**, Taylor & Francis, v. 63, n. 324, p. 1379–1389, 1968.

SIEGEL, S.; JR, N. J. C. **Estatística não-paramétrica para ciências do comportamento**. [S.l.]: Artmed Editora, 1975.

SQLSERVER. **SQL Server**. 2023. <https://www.microsoft.com/pt-br/sql-server/sql-server-downloads>. [Online; accessed 28-set-2023].

TRIVEDI, K. S.; GROTTKE, M.; ANDRADE, E. Software fault mitigation and availability assurance techniques. **International Journal of System Assurance Engineering and Management**, Springer, v. 1, p. 340–350, 2010.

VALENTIM, N. A.; MACEDO, A.; MATIAS, R. A systematic mapping review of the first 20 years of software aging and rejuvenation research. In: IEEE. **2016 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)**. [S.l.], 2016. p. 57–63.

VINÍCIUS, L.; RODRIGUES, L.; TORQUATO, M.; SILVA, F. A. Docker platform aging: a systematic performance evaluation and prediction of resource consumption. **The Journal of Supercomputing**, Springer, p. 1–31, 2022.

XIE, Y.; AIKEN, A. Context-and path-sensitive memory leak detection. In: **Proceedings of the 10th European software engineering conference held jointly with 13th ACM SIGSOFT international symposium on Foundations of software engineering**. [S.l.: s.n.], 2005. p. 115–125.

YIGITBASI, N.; GALLET, M.; KONDO, D.; IOSUP, A.; EPEMA, D. Analysis and modeling of time-correlated failures in large-scale distributed systems. In: IEEE. **2010 11th IEEE/ACM International Conference on Grid Computing**. [S.l.], 2010. p. 65–72.

ZHENG, P.; QI, Y.; ZHOU, Y.; CHEN, P.; ZHAN, J.; LYU, M. R.-T. An automatic framework for detecting and characterizing performance degradation of software systems. **IEEE Transactions on Reliability**, IEEE, v. 63, n. 4, p. 927–943, 2014.