UNIVERSIDADE DO VALE DO TAQUARI - UNIVATES
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
CURSO DE SISTEMAS DE INFORMAÇÃO

UM ESTUDO COMPARATIVO SOBRE A PERFORMANCE DE DIFERENTES BASES NOSQL EM APLICAÇÕES DE BUSINESS INTELLIGENCE

Fernando Augusto Giordani

Lajeado, novembro de 2017

Fernando Augusto Giordani

UM ESTUDO COMPARATIVO SOBRE A PERFORMANCE DE DIFERENTES BASES NOSQL EM APLICAÇÕES DE BUSINESS INTELLIGENCE

Trabalho de Conclusão de Curso apresentado ao Centro de Ciências Exatas e Tecnológicas da Universidade do Vale do Taquari - UNIVATES, como parte dos requisitos para a obtenção do título de bacharel em Sistemas de Informação.

Área de concentração: Banco de Dados

Orientador: Pablo Dall'Oglio

Lajeado, novembro de 2017

Fernando Augusto Giordani

UM ESTUDO COMPARATIVO SOBRE A PERFORMANCE DE DIFERENTES BASES NOSQL EM APLICAÇÕES DE BUSINESS INTELLIGENCE

Este trabalho foi julgado adequado para a obtenção do título de bacharel em Sistemas de Informação do CETEC e aprovado em sua forma final pelo Orientador e pela Banca Examinadora.

Banca Examinadora:

Prof. Pablo Dall'Oglio, UNIVATES - Orientador

Mestre em Computação Aplicada pela UNISINOS – São Leopoldo, Brasil

Prof. Juliano Dertzbacher, UNIVATES

Mestre em Computação pela UFRGS – Porto Alegre, Brasil

Prof. Mouriac Halen Diemer, UNIVATES

Mestre em Computação pela UFRGS – Porto Alegre, Brasil

Lajeado, novembro de 2017

RESUMO

Conforme os anos passam, maior é a experiência adquirida por uma empresa enquanto está ativa no mercado e, consequentemente, maior o seu banco de dados tende a ficar, chegando a um certo ponto onde nem mesmo uma ótima modelagem dos dados evitará problemas como baixo desempenho, falta de escalabilidade e/ou baixa disponibilidade dos dados. Esses três fatores são essenciais para as ferramentas da área de Inteligência de Negócios, pois os níveis Estratégico e Tático de uma organização precisam que as informações estejam disponíveis no maior tempo possível e, sejam apresentadas de forma rápida. Geralmente, o modelo Relacional é utilizado na modelagem das estruturas de dados, porém quando um desses problemas ocorre, torna-se necessário realizar uma pesquisa por bancos de dados mais eficientes, conhecendo então os modelos Não-Relacional. Este trabalho visa realizar um comparativo de desempenho entre dois modelos Não-Relacionais, definindo uma estrutura a ser implementada em laboratório, em um ambiente simulado, e, através da utilização de uma ferramenta gratuita de Inteligência de Negócios, realizar testes obtendo métricas como tempo de execução, uso da memória e uso do processador.

Palavras-chave: Modelo Não-Relacional, Modelo NoSQL, Inteligência de Negócios.

ABSTRACT

As the years go by, greater is the experience gained by a company while it is active in the marketplace and, consequently, greater your database tends to be, reaching to a point where not even a great data modeling will avoid problems like low performance, lack of scalability and / or low data availability. These three factors are essential for business intelligence tools because an organization's Strategic and Tactical levels need the information to be available as long as possible and be presented in a fast way. Generally, the Relational model is used in the modeling of data structures, but when one of these problems occurs, it is necessary to perform a search for more efficient databases, knowing the Non-Relational models. This work aims to perform a performance comparison between two Non-Relational models, defining a structure to be implemented in laboratory, in a simulated environment, and, through the use of a free Business Intelligence tool, perform tests obtaining metrics such as execution time, memory usage and processor usage.

Keywords: Non-Relational Model, NoSQL Model, Business Intelligence.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE CÓDIGOS

LISTA DE QUADROS

LISTA DE ABREVIATURAS

BD	Bancos de Dados
BI	Business Intelligence
CEO	Chief Executive Officer
CPU	Central Processing Units
CSV	Comma-Separated Values
DCL	Data Control Language
DDL	Data Definition Language
DRM	Digital Rights Management
DML	Data Manipulation Language
DW	Data Warehouse
ERP	Enterprise Resource Planning
ETL	Extract, Transform and Load
FIFO	First-In First-Out
HP	Hewlett Packard
HTML	HyperText Markup Language
IBM	International Business Machines
JDBC	Java Database Connectivity
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
MAL	MonetDB Assembly Language
MPP	Massively Parallel Processing
MSSIS	Microsoft SQL Server Integration Services
NoSQL	Not Only SQL

ODBC	Open Database Connectivity
OLAP	Online Analytic Processing
ORM	Object-Relational Mapper
PBA	Pentaho Business Analytics
PCRE	Perl Compatible Regular Expressions
PDI	Pentaho Data Integration
PHP	PHP: Hypertext Preprocessor
RHEL	Red Hat Enterprise Linux
RoR	Ruby on Rails
RTF	Rich Text Format
SGBD	Sistema Gerenciador de Banco de Dados
SAD	Sistemas de Apoio à Decisão
SI	Sistemas de Informação
SIE	Sistemas de Informação Executiva
SIG	Sistemas de Informação Gerencial
SLES	SUSE Linux Enterprise Server
SPT	Sistemas de Processamento de Transações
SQL	Structured Query Language
XML	eXtensible Markup Language

SUMÁRIO

1. INTRODUÇÃO

O mercado competitivo exige que as empresas estejam sempre à frente de seus concorrentes e, para isso acontecer, elas precisam dar importância à análise de um dos bens mais preciosos de sua posse, a informação. De acordo com Kimball e Ross (2002), a informação é um bem intangível e valioso, é considerado um ativo da empresa.

Contudo, somente com a informação não é possível tomar uma decisão de forma inteligente. Para isso acontecer, o gerente da empresa precisa ter como compromisso a aptidão da tomada de decisão eficiente e correta, utilizando os Sistemas de Informação (SI), mais especificamente, o Sistemas de Apoio à Decisão (SAD).

Um Sistema de Informação pode ser classificado como um grupo de elementos inter-relacionados e, capazes de extrair, controlar e disseminar dados e informações, proporcionando o melhor feedback possível para atender um determinado objetivo. Todos os tipos de sistemas são projetados para auxiliar na automatização das tarefas de uma empresa ou, auxiliar na tomada de decisão.

Os principais tipos de SI são: os Sistemas de Processamento de Transações (SPT), utilizados especialmente pelo nível Operacional de uma empresa; os Sistemas de Informação Gerencial (SIG), usados pelo nível Tático da organização; os Sistemas de Apoio à Decisão, que atende os níveis Tático e Estratégico; e, por último, os Sistemas de Informação Executiva (SIE), que são empregados pelo nível Estratégico.

Devido ao alto volume de informações relevantes para uma empresa, torna-se inviável para o ser humano gerenciá-las sem ter registro em uma base. Surge então, a necessidade de implantar um Banco de Dados (BD), para registrar essas informações e, consequentemente, um Sistema Gerenciador de Banco de Dados (SGBD), para manipulá-las. Esses dois itens tornaram-se importantes para as empresas, ou seja, não existem mais empresas que não tenham um BD implantado pois, ele é fundamental para o registro das operações e para a comunicação com os Sistemas de Informação (SI), se tornando o cérebro do negócio, onde todas as informações de cunho importante estão armazenadas.

Toda e qualquer estrutura do banco utiliza um modelo de armazenamento e, entre os mais famosos, está o modelo Relacional, que é tradicionalmente utilizado pelas empresas e, o modelo *Not Only SQL* (NoSQL) ou, Não-Relacional, que possui uma estrutura diferente do modelo anterior e propõe maior desempenho.

O modelo NoSQL pode ser: Orientado a Documentos, este modelo respeita a definição de dados e documentos autocontidos, dispensando descrições e definindo desde o início como ele será apresentado e, qual será sua estrutura de armazenamento; Orientado a Chave-Valor, esse modelo é composto de uma chave e, cada chave, possui seu respectivo valor, podendo haver duplicação de uma chave; Orientado a Grafo, este modelo tende a ser o mais complexo de todos os modelos NoSQL pois, os relacionamentos entre as tabelas são feitos através de grafos; e, por fim, Orientado a Coluna, esse modelo é parecido com o Modelo Relacional, entretanto, no lugar de registrar os dados agrupando por linhas, ele grava os dados agrupando por colunas, reduzindo o tempo de consulta e gravação dos dados.

Um dos softwares mais requisitados pelas empresas é o *Business Intelligence* (BI), que é classificado como um SAD, ou seja, possui como objetivo principal, analisar os dados de uma determinada empresa, apresentando a ela, análises, gráficos e indicadores de desempenho para assim, ampará-la na tomada de decisão.

Na maioria dos casos, a empresa possui uma base de dados colossal e, como sua necessidade e interesse de encontrar informações relevantes tende a crescer cada vez mais, torna-se necessário pesquisar alternativas de sistemas que proponham melhor desempenho, agilidade e, quando tivermos uma base mais complexa, saibam lidar com *Big Data*, que é um termo muito utilizado na atualidade para nomear conjuntos gigantescos e complexos de dados.

1.1. Motivação

Conforme a necessidade de explorar maiores quantidades de dados cresce, maior será a demanda por desempenho e disponibilidade dos dados. Entretanto, o modelo relacional chega a um certo ponto em que não consegue mais dispor de um alto desempenho, de uma alta disponibilidade ou, dispor juntamente dos dois, sendo necessário pesquisar por modelos de banco de dados que atendam esses dois requisitos necessários.

Ao realizar uma busca por modelos de banco de dados mais eficientes, é possível perceber que o modelo NoSQL Orientado a Colunas é bastante recomendado, principalmente, por causa de seu desempenho obtido nas consultas. Comparando esse modelo com o modelo Relacional, notamos que o modelo Não-Relacional não é tão utilizado pelas empresas, universidades e faculdades, e isso acaba despertando uma curiosidade quanto a sua organização e, o quão eficiente ele pode ser comparado com o modelo tradicionalmente utilizado.

Ao longo do tempo, vários comparativos de desempenho entre o modelo Relacional e NoSQL foram desenvolvidos, então para essa pesquisa serão comparadas duas bases NoSQL. Além disso, como o fator desempenho é um ponto crucial para as ferramentas de BI, utilizar nos testes uma ferramenta dessa categoria pode gerar resultados mais interessantes para uma empresa do segmento avaliar se vale a pena migrar para uma das bases estudadas.

1.2. Objetivo geral

É objetivo principal desta pesquisa analisar comparativamente a utilização de diferentes Bases de dados NoSQL para *Data Warehouses* (DW), a fim de avaliar sua performance em aplicações de *Business Intelligence*. Para esse comparativo, utilizaremos um cenário construído em laboratório, em um ambiente simulado.

1.3. Objetivos específicos

São objetivos específicos desta pesquisa:

- Pesquisar sobre diferentes modelos de bases NoSQL;

- Modelar uma estrutura de *Data Warehouse* a ser utilizada na avaliação;
- Selecionar diferentes bases de dados NoSQL;
- Criar estrutura utilizando as duas bases NoSQL;
- Escolher uma ferramenta de *Business Intelligence* para avaliações;
- Realizar testes de desempenho sobre as bases criadas;
- Analisar comparativamente os resultados obtidos.

1.4. Organização do trabalho

Como base para a realização desta pesquisa, os capítulos foram estruturados da seguinte maneira:

O primeiro capítulo descreve o nível de importância da informação para as empresas e o que elas precisam fazer para estar sempre a frente de seus concorrentes, desde o registro de suas operações até a aquisição de ferramentas para auxiliar a tomada de decisão. Além disso, são descritos a motivação e os objetivos do presente estudo.

O segundo capítulo apresenta os referenciais teóricos sobre os tópicos abordados no presente trabalho, dentre eles estão: a caracterização de cada um dos tipos de sistemas de informação, a definição de um banco de dados e de seus modelos mais utilizados e, alguns conceitos importantes para a área de Inteligência de Negócios.

O terceiro capítulo informa qual foi a metodologia adotada para a realização desse comparativo, descrevendo os métodos de pesquisa e procedimentos executados.

O quarto capítulo apresenta a análise feita com base em alguns bancos de dados não-relacionais, com o objetivo de escolher dois deles para utilizar no comparativo e, a análise feita com base em algumas ferramentas de BI, com o objetivo de escolher uma para posterior avaliação. Também apresenta o resultado da análise com base em alguns critérios adotados.

O quinto capítulo, além de apresentar o estudo desenvolvido, apresenta o cenário e os motivos para utilizá-lo nesse estudo e ainda, detalha como os comparativos entre os dois bancos não-relacionais foram elaborados.

O sexto capítulo demonstra os resultados obtidos a partir dos testes realizados perante os critérios de avaliação determinados no capítulo anterior.

O sétimo capítulo apresenta as considerações finais desse estudo, analisando todos os resultados alcançados.

2. REFERENCIAL TEÓRICO

Este capítulo busca apresentar os principais conceitos relacionados ao trabalho, utilizando informações obtidas através da realização da pesquisa bibliográfica. Inicialmente será apresentada a principal ferramenta utilizada nas operações frequentes de uma empresa e, cuja comunicação com os Bancos de Dados (BD) é indispensável: os Sistemas de Informação (SI). Em seguida, serão abordadas breves descrições dos tipos de sistemas, com exceção do Sistema de Apoio à Decisão (SAD), que será utilizado no estudo. Também abordará os conceitos e principais modelos de BD, destacando os modelos *Not Only SQL* (NoSQL) Orientado a Coluna, modelo escolhido para essa pesquisa.

E, por fim, visto que o trabalho utiliza uma ferramenta *Open Source* de *Business Intelligence* (BI) para os testes de desempenho, serão abordados também alguns conceitos relacionados à área, como: *Data Warehouse* (DW), *Extract, Transform and Load* (ETL), o principal modelo utilizado para criação de DW, *Big Data* e o conjunto de funções chamado de *Slice and Dice* que, dentre suas funcionalidades, disponibiliza o *Drill Down* e *Drill Up*.

2.1. Sistemas de Informação

Sistemas de Informação (SI) pode ser especificado como um conjunto de elementos interligados entre si que organizam e compartilham informações de uma organização, para assim, prestar o devido apoio à coordenação e controle da empresa e, também as tomadas de decisão (LAUDON e LAUDON, 2011 apud JOÃO, 2012, p. 6).

Segundo Audy, de Andrade e Cidral (2005), os SI fornecem a união entre as áreas e processos de negócio da organização, também fornecem informações sobre os ambientes externo e interno de uma empresa, visando planejar a longo prazo e, por último, auxiliam as empresas na visualização de seus problemas e oportunidades, criando ou melhorando seus produtos.

As organizações sempre buscam a excelência profissional, procurando aperfeiçoar suas operações diárias; buscam também inovação em seus produtos ou serviços, assim como em seus modelos de negócio, que definem como a empresa deve fabricar, entregar e comercializar seu produto ou, como um serviço deve ser prestado para assim, aumentar seu valor. Além disso, procuram manter ou melhorar seu relacionamento com os seus fornecedores e, principalmente, com sua carteira de clientes. Para isso, necessitam da informação mais segura o possível para tomar uma decisão e cumprirem um ou mais desses objetivos, para adquirirem uma vantagem competitiva no ramo de atuação.

2.2. Níveis de Tomada de Decisão

Para garantir que a empresa alcance os objetivos preestabelecidos, é preciso unir as pessoas dos diversos níveis organizacionais e, através de uma comunicação clara, garantir que saibam os seus objetivos e organizem as atividades da empresa para levá-la adiante (PERADELLES, 2016).

O conceito dos níveis organizacionais é um dos mais citados na área de Administração e, cada nível, possui um foco (RENNÓ, 2010). Segundo Peradelles (2016) e Rennó (2010), a empresa é dividida em três níveis organizacionais, veja as suas diferenças a seguir.

2.2.1. Nível Estratégico

O nível Estratégico é onde ocorre o planejamento a longo prazo (5 a 10 anos), ou seja, visa o futuro da organização tendo como base os fatores externos e internos de seu ambiente, onde definimos informações como os valores, visões e o principal, o propósito da sua existência. A tomada de decisão é realizada pelo nível mais alto da empresa, ou seja, pela alta administração da empresa que, na maior parte dos casos, é o proprietário, o *Chief Executive Officer* (CEO), o presidente ou a diretoria (PERADELLES, 2016).

De acordo com o Portal Administração (2014), esse nível é responsável por indicar as estratégias de um futuro não próximo, colaborando na definição dos principais princípios da empresa e, na análise dos ambientes internos e externos. Resumindo esse nível, é um processo que possibilita determinar o rumo que a empresa seguirá com objetivo de conquistar um nível de otimização entre a organização e seu ambiente.

Segundo Rennó (2010), é o primeiro nível da hierarquia, onde os gestores da alta cúpula determinam os temas que afetam a organização como um todo. Temas como regularização governamental, desempenho dos concorrentes e, variações nos hábitos dos consumidores são responsabilidades desse nível.

2.2.2. Nível Tático

O nível Tático é responsável pelo planejamento a médio prazo (1 a 3 anos), ou seja, tem um envolvimento mais restrito, se comparado com o nível Estratégico, e é o encarregado de criar as metas e condições para que os procedimentos definidos no nível superior sejam atendidos. As pessoas que ocupam os cargos executivos ou de gerentes são os encarregados pela tomada de decisão (PERADELLES, 2016).

Segundo o Portal Administração (2014), é onde há intermediação entre os níveis estratégico e operacional da empresa, geralmente é projetado a médio prazo e atinge cada unidade da organização, traduzindo e representando as decisões do nível acima e, transformando-as em planos precisos dentro de cada unidade.

De acordo com Rennó (2010), esse nível é composto por chefes de divisão e, nesta posição, temos a responsabilidade de uma área específica da organização, Gerência de Recursos Humanos por exemplo e, além disso, temos a missão de implementar as estratégias e políticas estabelecidas pelo nível Estratégico. Gerentes desse nível devem estabelecer boas parcerias entre si e seus subordinados, resolver conflitos e motivar seus funcionários.

2.2.3. Nível Operacional

O nível Operacional tem como responsabilidade, concretizar os procedimentos executados em um curto prazo (3 a 6 meses). É nesse nível que há participação dos três níveis,

pois é necessária a garantia da realização de todas as ações planejadas desde a Alta Direção (PERADELLES, 2016).

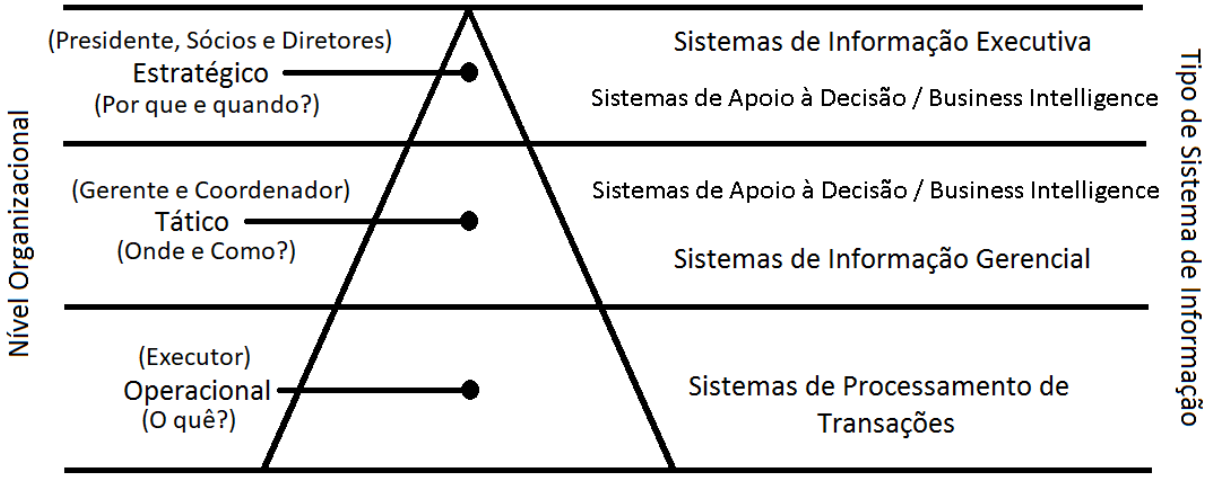
Em concordância com o Portal Administração (2014), esse nível é a execução das ações anteriormente especificadas pelos baixos níveis de gestão (nível tático). Tem como principal objetivo transformar os planos táticos de cada setor em planos operacionais para cada atividade, pensando a curto prazo e, no fato de estar diretamente ligado com a área responsável pela execução do plano de ação.

Conforme Rennó (2010), esse nível possui foco em tarefas específicas, geralmente de curto prazo, como a fabricação de bens e serviços. É composto de supervisores e chefes de equipe e, as regras e diretrizes especificadas pelo nível Tático devem ser continuadas, sem afetar a motivação e a eficiência de seus funcionários.

2.3. Tipos de Sistemas de Informação

Atualmente, esses sistemas são classificados de acordo com sua finalidade e, pelo nível organizacional. Na Figura 1, será apresentado como cada um deles foi classificado e, em seguida, as definições para cada tipo de SI.

Figura 1 – Tipos de SI em relação aos níveis organizacionais



Fonte: Adaptado pelo autor com base em Audy, Andrade e Cidral (2009, p.118) e, Blog Adm Inside (2015).

2.3.1. Sistemas de Processamento de Transações

Uma transação pode ser classificada como uma troca de informações executada quando duas partes participam de uma mesma atividade. As transações, basicamente, representam os eventos básicos executados por uma organização durante seu período de atividade. Para cada evento executado, uma quantidade de dados é gerada e, esses são utilizados pelos SI. Os Sistemas de Processamento de Transações (SPT) são responsáveis pela execução e registro das transações rotineiras e de nível operacional de uma empresa. Sendo assim, podem ser chamados também de sistemas operativos ou transacionais (AUDY, DE ANDRADE, E CIDRAL, 2005).

Segundo Florenzano (2015), esses sistemas são responsáveis pela vigilância, coleta, estocagem e computação de dados gerados a partir das transações empresariais e, inseridos no banco de dados empresarial. Esses dados podem ser coletados por pessoas ou sensores conectados ao computador mediante a um dispositivo de entrada e, em seguida, podem ser processados de duas formas: a partir de um processamento em lote, onde as informações são coletadas e submetidas a um processamento posterior; e, a partir de um processamento online, onde as informações são processadas no mesmo momento em que são registradas.

São alguns exemplos de tipos de aplicação: Processamento de pedidos, Fatura, Controle de estoque, Contas a pagar, Contas a receber, e, Compras.

2.3.2. Sistemas de Informação Gerencial

Os Sistemas de Informação Gerencial (SIG) sumarizam e relatam a situação atual das operações organizacionais, satisfazendo grande parte dos gerentes do nível tático através de relatórios cuja composição são indicadores de desempenho obtidos para uma determinada área ou divisão. São métodos para controlar as atividades cotidianas da organização e, seu desenvolvimento e utilização é assegurado quando a organização dispõe de um SPT já implementado e, de uma cultura de gerência regrada no uso de indicadores ou na avaliação dos resultados (AUDY, DE ANDRADE, E CIDRAL, 2005).

Conforme Florenzano (2015), esses sistemas oferecem aos gerentes do nível tático, informações em forma de relatórios, como auxílio na organização e controle das operações. São cinco os principais tipos de relatórios gerados a partir desse sistema: Rotina, produzidos com

pausas programadas; Detalhados, apresentando um maior detalhamento; Indicadores principais, resumindo a performance de atividades cruciais; Comparativos, comparando a performance de diferentes filiais ou, faixas de tempo; e, de Exceção, onde o gerente pode obter informações específicas.

São alguns exemplos de relatórios: Relatório de vendas diário ou semanal, demonstrativos financeiros mensais e relatório com dados dos clientes que ultrapassaram os limites de crédito.

2.3.3. Sistemas de Informação Executiva

Esses sistemas são conhecidos por auxiliarem os executivos nas decisões não-estruturadas, ou seja, quando as situações não possuem um nível mínimo de entendimento ou, quando o procedimento adotado não é aceito por todos. Eles, os Sistemas de Informação Executiva (SIE), prestam esse apoio disponibilizando um ambiente computacional, comunicável e, de simples acesso aos dados internos e externos da organização. A partir desse ambiente, a ferramenta proporciona ao executivo uma visão da própria situação atual e, das tendências na área de atuação da empresa (AUDY, DE ANDRADE, E CIDRAL, 2005).

De acordo com Florenzano (2015), esse sistema foi elaborado para atender necessidades específicas da alta direção, fornecendo um rápido acesso as informações atuais e, aos relatórios gerenciais. Além disso, deve ser descomplicado e utilizar gráficos como base. Para o nível Estratégico, o que determina a importância desse sistema é a habilidade de gerar os relatórios de Exceção e de Expansão. Esse tipo de sistema pode variar em termos de competências e privilégios pois, podem ser refinados com análise e apresentação multidimensionais, acesso simples aos dados, simples interface gráfica, acesso à intranet, entre outros.

2.3.4. Sistemas de Apoio à Decisão

Como as situações cotidianas enfrentadas pelos gerentes dos níveis tático e estratégico sempre possuem uma característica incomum, dificilmente, essas serão previstas ou planejadas e, para resolver essas situações, decisões semiestruturadas precisam ser tomadas. Essas decisões são situações moderadamente compreendidas e, que possuem algum procedimento conhecido para utilização (AUDY, DE ANDRADE, E CIDRAL, 2005).

Sabendo que uma decisão pode levar uma organização à falência, os Sistemas de Apoio à Decisão (SAD) prestam o devido apoio aos gerentes, dos níveis estratégico e tático da organização, no momento exato de tomar uma decisão semiestruturada e, isso é feito com base nos dados adquiridos a partir dos SIG, SPT e fontes externas (AUDY, DE ANDRADE, E CIDRAL, 2005).

Esse sistema relaciona modelos e dados, tentando eliminar os problemas semiestruturados e não-estruturados, com grande participação do usuário. Os SADs podem: Examinar várias alternativas de forma rápida, realizar análises de riscos sistemáticos, incorporar-se aos sistemas de comunicação e bancos de dados e, auxiliar o trabalho em equipe (FLORENZANO, 2015).

2.3.5. Business Intelligence

Também conhecido por Inteligência de Negócios, é um software desenvolvido com o intuito de representar, através de gráficos, análises e indicadores de desempenho, as informações extraídas do sistema da empresa e/ou do ambiente externo. Além disso, é uma ferramenta que auxilia na descoberta do conhecimento, nas simulações das possíveis situações e, torna possível para o usuário analisar melhor os dados (COLAÇO JÚNIOR, 2004).

Segundo Ausland (2015), o conceito original foi proposto na década de 90 e descreve como um conjunto de técnicas utilizadas para classificar, extrair, avaliar, supervisionar e distribuir as informações que sustentam a gestão de negócios, por exemplo dados dos fornecedores, dos clientes e, concorrentes. Resumindo, é uma etapa onde os dados brutos de uma empresa são transformados em informações claras e significativas para uma futura análise do negócio.

De acordo com o site Software Advice (2017), no ramo atual de BI, existem mais de 130 ferramentas pagas e, realizando outra busca pela internet, foi identificado pelo menos 40 ferramentas *Open Source*. Dentre elas, a ferramenta Superset, disponibilizada a partir do Github; Pentaho Business Analytics, disponibilizada pela Pentaho; e, SpagoBI, disponibilizada pela OW2 Corsortium, uma comunidade de software de código aberto.

2.4. Banco de Dados

Ainda existem pessoas que confundem Banco de Dados (BD) com Sistema Gerenciador de Banco de Dados (SGBD), entretanto, ambos não são a mesma coisa. Um banco pode ser denominado como um conjunto de dados interligados. Geralmente, são compostos pelos dados das operações de uma empresa, mas em sua composição pode haver informações sobre seu ramo de atuação e até de seus concorrentes. Também pode haver registros dos conhecimentos e experiências passadas pela empresa, e até mesmo das decisões tomadas até então (COLAÇO JÚNIOR, 2004).

Segundo Elmasri e Navathe (2011), Banco de Dados é um grupo de fatos conhecidos, com significado contido e que podem ser catalogados. Podemos citar como exemplo, o nome de uma pessoa que conhecemos, o endereço dessa pessoa, os números de telefone da mesma, entre outros dados.

Todo e qualquer banco utiliza um modelo de Banco de dados, podendo ser: Relacional, modelo tradicionalmente utilizado; ou, Não-relacional, também conhecido pelo termo *Not Only SQL* (NoSQL). Esses dois termos são detalhados mais adiante.

2.5. Sistema Gerenciador de Banco de Dados

O Sistema Gerenciador de Banco de Dados (SGBD) é classificado como um conjunto de programas utilizados pelos usuários para modelar, criar e gerenciar BD com base em um determinado objetivo. Os dados armazenados precisam estar íntegros e sendo compartilhados. Sendo assim, podem também representar a centralização de arquivos com informações redundantes, minimizando a redundância entre os mesmos (COLAÇO JÚNIOR, 2004).

2.6. Modelo Relacional

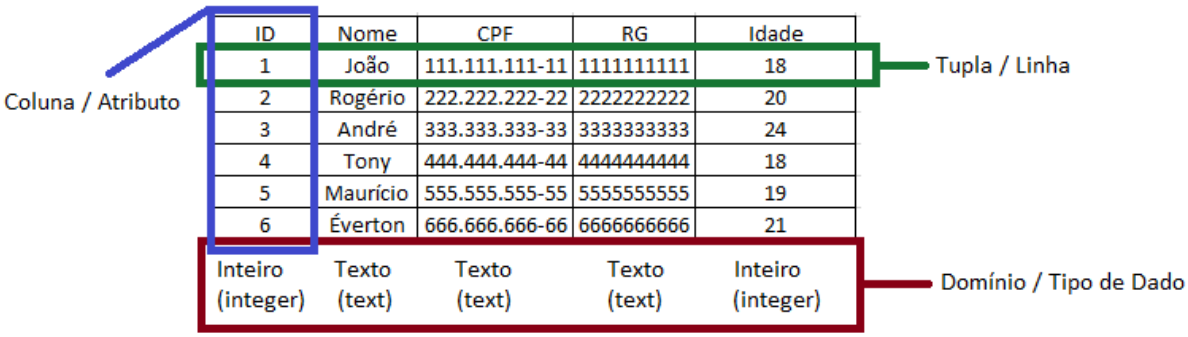
De acordo com Macário e Baldo (2005), esse modelo foi proposto na década de 70 pelo matemático britânico Edgar Frank Codd, como um novo modelo de representação dos dados e, apesar desse modelo ser predominante até os dias atuais, Codd não teve seus minutos de fama. Segundo o blog *Password* (2010), como a ideia original não foi totalmente compreendida pelos

desenvolvedores da *International Business Machines* (IBM), empresa onde Codd trabalhava, várias alterações foram feitas, originando a linguagem de consulta *Structured Query Language* (SQL).

A estrutura SQL é composta das seguintes linguagens: *Data Definition Language* (DDL, ou Linguagem de Definição de Dados): utilizada para criar, editar e remover tabelas e/ou índices; *Data Manipulation Language* (DML, ou Linguagem de Manipulação de Dados): usada para consultar, inserir, alterar ou remover uma informação dentro da base de dados; e, *Data Control Language* (DCL, ou Linguagem de Controle de Dados): que é utilizada para alterar as permissões de acesso a base de dados.

O modelo de Codd é conhecido como um conjunto de tabelas, ou relações, de nome único. Cada tabela possui um determinado número de colunas, ou atributos e, um determinado número de linhas, ou tuplas. Para cada atributo é definido um tipo de dado, ou domínio. Na Figura 2 é apresentado um exemplo de tabela que utiliza o modelo de Codd e, na Figura 3, são listados os 10 bancos Relacionais mais utilizados.

Figura 2 – Exemplo de tabela utilizando o modelo Relacional



Fonte: Autor.

Figura 3 – Os 10 bancos Relacionais mais utilizados

137 systems in ranking, October 2017

Rank			DBMS	Database Model	Score		
Oct 2017	Sep 2017	Oct 2016			Oct 2017	Sep 2017	Oct 2016
1.	1.	1.	Oracle	Relational DBMS	1348.80	-10.29	-68.30
2.	2.	2.	MySQL	Relational DBMS	1298.83	-13.78	-63.82
3.	3.	3.	Microsoft SQL Server	Relational DBMS	1210.32	-2.23	-3.86
4.	4.	4.	PostgreSQL	Relational DBMS	373.27	+0.91	+54.58
5.	5.	5.	DB2	Relational DBMS	194.59	-3.75	+14.03
6.	6.	6.	Microsoft Access	Relational DBMS	129.45	+0.64	+4.78
7.	7.	7.	SQLite	Relational DBMS	111.98	-0.05	+3.41
8.	8.	8.	Teradata	Relational DBMS	80.08	-0.83	+3.85
9.	9.	9.	SAP Adaptive Server	Relational DBMS	67.24	+0.48	-2.25
10.	10.	10.	FileMaker	Relational DBMS	61.06	+0.07	+6.11

Fonte: Retirado da página *DB-Engines*.

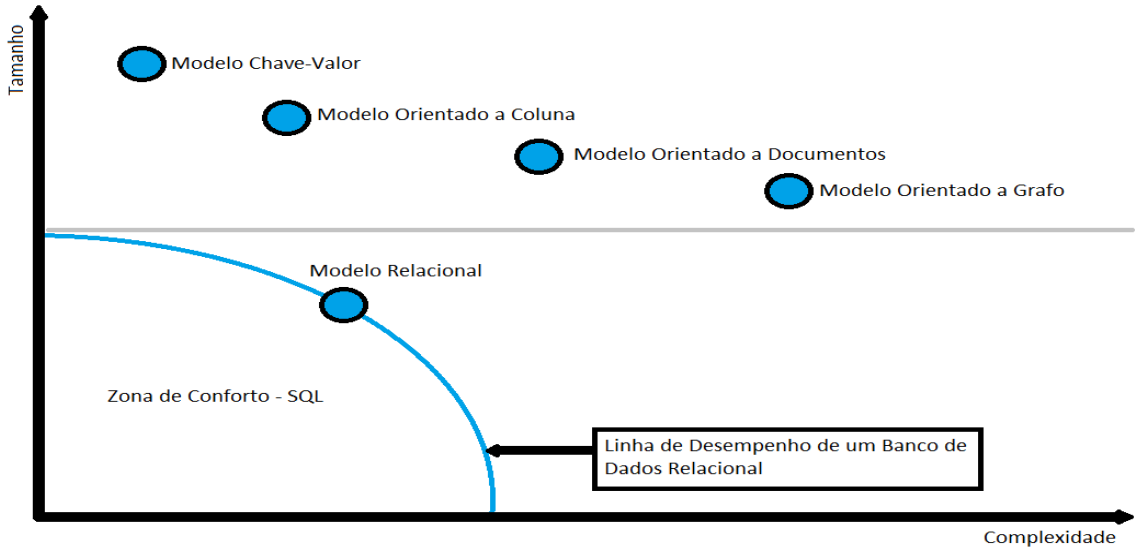
2.7. Modelo NoSQL

Conforme o fluxo de dados cresce, o modelo Relacional tende a sofrer cada vez mais com a escalabilidade, a indisponibilidade de dados, a falta de desempenho e/ou, a confiabilidade. Surge então uma alternativa ao modelo Relacional, o modelo NoSQL, também chamado de modelo Não-Relacional.

Esse modelo, diferente do descrito no item 2.6, atende aos requisitos do ambiente de computação compartilhada em escala expandida, resolvendo assim os problemas encontrados para o modelo Relacional.

Atualmente, existem quatro modelos NoSQL e eles podem ser orientados a: Documentos, Chave-Valor, Grafo ou a Colunas. Esse trabalho explicará cada um dos modelos mas terá foco principal no modelo Orientado a Colunas ou, modelo Colunar. Na Figura 4 são apresentados, através dos níveis de Complexidade e Tamanho, as diferenças entre os cinco modelos citados até então.

Figura 4 – Gráfico comparativo dos níveis de complexidade e tamanho



Fonte: Adaptado pelo Autor com base em Vardanyan (2011).

Dentre os quatro principais modelos NoSQL, o modelo Chave-Valor tende a ser o menos complexo de implementar pois ele é, basicamente, uma grande tabela *hash*, composta de uma chave e, cada chave, possui um valor. Apesar de sua facilidade de implementação, esse modelo é incapaz de consultar ou alterar o trecho de um valor armazenado (VARDANYAN, 2011). Esse modelo é utilizado em aplicações cujo foco é lidar com carregamento de imensas quantidades de dados, tem como ponto forte o fato das pesquisas serem rápidas, mas os dados registrados não possuem um *schema*.

O modelo Colunar é um pouco mais difícil que o citado anteriormente, pois foi criado pensando em armazenar, através de uma boa organização dos dados, e processar quantidades de dados mais robustas, desempenhando pesquisas mais eficientes. Esse modelo é utilizado por sistemas de arquivos distribuídos, seus pontos fortes são as pesquisas rápidas e, o bom compartilhamento de armazenamento de dados, porém sua API não é de qualidade, sendo seu único ponto fraco (VARDANYAN, 2011).

Já o modelo Orientado a Documentos é similar ao padrão Chave-Valor pois é, basicamente, um conjunto de documentos versionados cuja composição são aglomerações de grupos de chave-valor, são flexíveis quanto a dados incompletos, porém não possuem uma sintaxe padrão de *query* (VARDANYAN, 2011). Utilizado para aplicações Web, ele é muito parecido com o modelo Chave-Valor, entretanto ele sabe qual é o valor. É flexível quanto aos dados incompletos e, como ponto negativo, não possui uma sintaxe padrão de *query*.

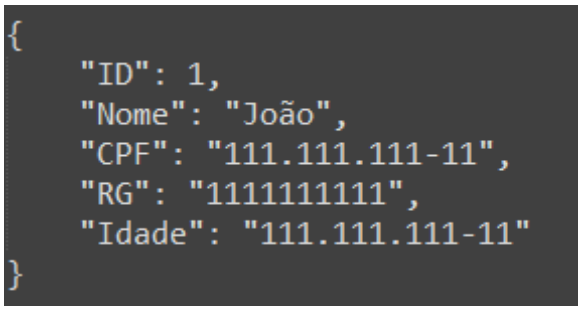
E, por último, o modelo Orientado a Grafo, que é considerado o modelo mais complexo pelo fato de não utilizar a estrutura SQL e nem organizar os dados entre tabelas, colunas e linhas. Esse modelo é composto de vértices, onde cada vértice representa uma entidade, e arestas que, além de representar a ligação realizada entre os vértices, pode conter informações sobre os relacionamentos (SATO, 2014).

As redes sociais são as principais aplicações que utilizam esse modelo. O lado positivo dele é que utiliza algoritmos gráficos, caminho mais curto e conectividade, entretanto, seu lado negativo é quando se torna necessário percorrer todo o gráfico em busca de uma única resposta, além de o agrupamento ser complicado (VARDANYAN, 2011).

2.7.1. Modelo NoSQL – Orientado a Documentos

Nesse modelo, os documentos representam as tabelas, sendo possível, compará-las com as tabelas convencionais. Uma das diferenças entre esse modelo e o modelo Relacional é que esse possui uma estrutura mais flexível e não fica atado às colunas pré-definidas. Na prática, isso significa que um número variável de documentos ligados a uma coleção podem contar com formatos versáteis. Muitos sistemas, que adotam esse modelo, utilizam o padrão *JavaScript Object Notation* (JSON) para armazenar os dados. Na Figura 5 é apresentado um exemplo desse documento (GROFFE, 2016).

Figura 5 – Exemplo de tabela utilizando o modelo Orientado a Documentos com JSON



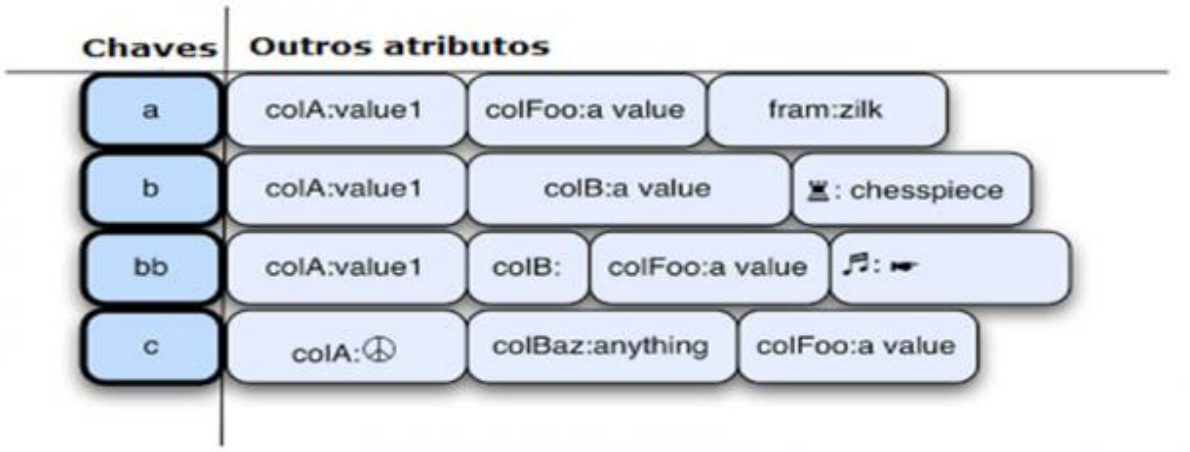
Fonte: Autor.

2.7.2. Modelo NoSQL - Orientado a Chave-Valor

Os bancos que utilizam esse modelo também são mais flexíveis, se comparados com o modelo Relacional, e são compostos por grupos de chaves onde, cada chave, possui seu

respectivo valor. As chaves funcionam como identificador único, mas nem sempre a duplicação de uma chave é evitada. Na Figura 6 é apresentado um exemplo desse modelo (GROFFE, 2016).

Figura 6 – Exemplo de tabela utilizando o modelo Orientado a Chave-Valor

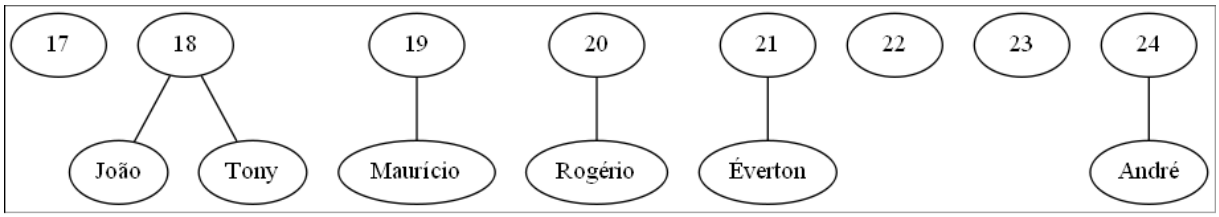


Fonte: Retirado da página *iMasters*.

2.7.3. Modelo NoSQL - Orientado a Grafo

Os bancos que utilizam esse modelo, representam os relacionamentos entre as tabelas através de grafos. Dentre os modelos NoSQL existentes até então, esse tende a ser o mais complexo (GROFFE, 2016). Na Figura 7 é apresentado um exemplo desse modelo.

Figura 7 – Exemplo de tabela utilizando o modelo Orientado a Grafo



Usuários e suas respectivas idades

Fonte: Autor.

2.7.4. Modelo NoSQL - Orientado a Coluna

No modelo Relacional, os grupos de dados são representados pelas linhas de uma tabela, entretanto no modelo colunar, os agrupamentos são representados pelas colunas e, os dados de

colunas distintas, quando representam o mesmo agrupamento, utilizam as mesmas posições do banco (GROFFE, 2016).

Esse modelo tende a ser muito mais eficiente, se comparado com o modelo Relacional, pois a maneira com que os dados são armazenados apresenta, como uma das vantagens, um nível de compressão superior a 60% mais eficiente (BARROSO, 2012).

Na Figura 8 é apresentado um exemplo da organização sem o identificador de registros e, em seguida, para identificar um registro, é feita uma alteração na estrutura conforme apresentado na Figura 9.

Figura 8 – Exemplo de BD Orientado a Colunas (sem identificador de registro)

ID	Nome	CPF	RG	Idade
1	João	111.111.111-11	1111111111	18
2	Rogério	222.222.222-22	2222222222	20
3	André	333.333.333-33	3333333333	24
4	Tony	444.444.444-44	4444444444	18
5	Maurício	555.555.555-55	5555555555	19
6	Éverton	666.666.666-66	6666666666	21

Fonte: Autor.

Figura 9 – Exemplo de BD Orientado a Colunas (com identificador de registro)

ID		Nome		CPF		RG		Idade	
ID	Value	ID	Value	ID	Value	ID	Value	ID	Value
1	1	1	João	1	111.111.111-11	1	1111111111	1	18
2	2	2	Rogério	2	222.222.222-22	2	2222222222	2	20
3	3	3	André	3	333.333.333-33	3	3333333333	3	24
4	4	4	Tony	4	444.444.444-44	4	4444444444	4	18
5	5	5	Maurício	5	555.555.555-55	5	5555555555	5	19
6	6	6	Éverton	6	666.666.666-66	6	6666666666	6	21

Fonte: Autor.

Outro exemplo que pode se notar uma grande diferença é, por exemplo, fazer uma média da idade. No modelo Relacional, todos os registros serão recuperados, carregando todos os campos, para em seguida, executar a operação, enquanto que no modelo Colunar, apenas a coluna “Idade” será utilizada nessa operação, aproveitando melhor os recursos disponíveis (BARROSO, 2012).

Na Figura 10 é apresentada uma lista dos 10 bancos que utilizam esse modelo de BD.

Figura 10 – Os 10 bancos Colunares mais utilizados

10 systems in ranking, October 2017

Rank Oct 2017	Rank Sep 2017	Rank Oct 2016	DBMS	Database Model	Score Oct 2017	Score Sep 2017	Score Oct 2016
1.	1.	1.	Cassandra	Wide column store	124.79	-1.41	-10.27
2.	2.	2.	HBase	Wide column store	64.39	+0.05	+6.20
3.	3.	↑ 4.	Microsoft Azure Cosmos DB	Multi-model	12.63	+1.40	+9.74
4.	4.	↓ 3.	Accumulo	Wide column store	4.00	+0.12	+0.57
5.	5.		Microsoft Azure Table Storage	Wide column store	3.29	+0.03	
6.	6.	6.	Google Cloud Bigtable	Wide column store	0.70	-0.12	+0.35
7.	↑ 8.		MapR-DB	Multi-model	0.64	+0.16	
8.	↓ 7.	↓ 7.	Sqrrl	Multi-model	0.50	-0.01	+0.24
9.	9.	↓ 8.	ScyllaDB	Wide column store	0.34	-0.01	+0.19
10.	10.		Alibaba Cloud Table Store	Wide column store	0.00	±0.00	

Fonte: Retirado da página *DB-Engines*.

Conforme a necessidade de explorar maiores quantidades de dados cresce, maiores serão as demandas por desempenho e o espaço ganho pelos bancos NoSQL. Na Figura 11 é apresentado um ranking atual dos bancos de Dados mais utilizados, dentre os dez primeiros, quatro não utilizam o modelo Relacional. De acordo com DB-Engines (2017), os bancos a serem utilizados nesse comparativo são classificados como relacionais. Entretanto, de acordo com Vertica (2017) e MonetDB (2017), eles são classificados como orientados a colunas. Ambos foram destacados na Figura 11.

Figura 11 – Ranking dos bancos de dados mais utilizados

334 systems in ranking, October 2017

Rank Oct 2017	Rank Sep 2017	Rank Oct 2016	DBMS	Database Model	Score Oct 2017	Score Sep 2017	Score Oct 2016
1.	1.	1.	Oracle	Relational DBMS	1348.80	-10.29	-68.30
2.	2.	2.	MySQL	Relational DBMS	1298.83	-13.78	-63.82
3.	3.	3.	Microsoft SQL Server	Relational DBMS	1210.32	-2.23	-3.86
4.	4.	↑ 5.	PostgreSQL	Relational DBMS	373.27	+0.91	+54.58
5.	5.	↓ 4.	MongoDB	Document store	329.40	-3.33	+10.60
6.	6.	6.	DB2	Relational DBMS	194.59	-3.75	+14.03
7.	7.	↑ 8.	Microsoft Access	Relational DBMS	129.45	+0.64	+4.78
8.	8.	↓ 7.	Cassandra	Wide column store	124.79	-1.41	-10.27
9.	9.	9.	Redis	Key-value store	122.05	+1.65	+12.51
10.	10.	↑ 11.	Elasticsearch	Search engine	120.23	+0.23	+21.12
			...				
25.	↓ 24.	↓ 23.	Memcached	Key-value store	27.64	-1.30	-1.45
26.	26.	26.	Vertica	Relational DBMS	22.06	+0.05	-0.25
27.	27.	↑ 28.	Microsoft Azure SQL Database	Relational DBMS	21.85	+0.25	+1.60
			...				
103.	↑ 110.	↓ 102.	VoltDB	Relational DBMS	1.59	+0.25	+0.22
104.	↑ 106.	↓ 95.	MonetDB	Relational DBMS	1.57	+0.13	-0.01
105.	↓ 104.	↓ 82.	mSQL	Relational DBMS	1.53	+0.03	-0.41

Fonte: Adaptada de *DB-Engines*.

2.8. Data Warehouse

De acordo com Kimball e Ross (2002), na maioria das vezes, a informação é preservada de duas formas: através de um sistema de nível operacional, onde os dados são armazenados e, através de um *Data Warehouse* (DW), onde os dados são disponibilizados externamente.

O *Data Warehouse* (DW) é classificado como uma base de dados histórica isolada do ambiente de produção e, a mesma é configurada para armazenar os dados do sistema da empresa. Antes dos dados serem inseridos nessa base, eles sofrem um processo de seleção, tratamento e organização dos dados, para assim, aumentar o desempenho da ferramenta de BI (COLAÇO JÚNIOR, 2004). Esse processo é denominado de *Extract, Transform and Load* (ETL) e será abordado mais adiante.

Com base nas experiências adquiridas por Kimball e Ross (2002) e, com os objetivos de organizar os dados para uma análise posterior, auxiliar no processo de compreensão e visualização dos dados mais significativos, resolver discrepância dos indicadores e, principalmente, auxiliar as empresas no processo de tomada de decisão com base em informações mais confiáveis, foram definidas as seguintes obrigações de um DW:

- **Disponibilizar, de forma fácil, a informação da organização:** os dados armazenados em um DW precisam ser de fácil absorção e, indiscutíveis para os clientes. O mesmo vale para as nomenclaturas, elas devem ser precisas e conhecidas pela empresa. Além disso, a ferramenta utilizada para analisar os dados registrados no DW deve ser de simples manipulação, evitando dificuldades que podem ser encontradas pelos usuários finais, ao realizar um cruzamento.

- **Apresentar, de forma consistente, a informação da organização:** o conteúdo do DW deve ser consolidado e, para acontecer isso, os dados das diferentes fontes da empresa precisam ser minuciosamente integrados, limpos e, assegurar da própria qualidade, para depois serem disponibilizados aos usuários finais. As informações de cada processo de negócio precisam estar relacionadas entre si, ou seja, caso duas medidas apresentem a mesma nomenclatura, elas devem ter o mesmo significado, caso contrário, devem possuir nomenclaturas distintas.

- **Ser flexível a mudanças:** cedo ou tarde, uma mudança dentro da empresa será feita e o DW precisa estar preparado para ela, garantindo com que os dados armazenados até o momento não sejam prejudicados com essa alteração. A alteração pode ser em uma necessidade do cliente ou usuário final, numa condição de negócio, numa tecnologia e até mesmo nos dados pois, ambos passam por um processo de evolução. Quando novas questões precisam ser respondidas ou, novos dados são adicionados ao DW, as aplicações e dados existentes precisam ser ajustados de acordo com as mudanças.

- **Oferecer segurança quanto a informação da organização:** o DW sempre será composto de informações de extrema importância para a empresa, informações significativas sobre o negócio da empresa, portanto precisa controlar o acesso a essas informações e dados.

- **Auxiliar como embasamento para uma melhor decisão:** o DW deve ser composto dos dados corretos para o apoio a tomada de decisão. A única saída verdadeira de um *Data Warehouse* são as decisões realizadas após a apresentação de suas comprovações. Essas decisões geram impacto e valor ao negócio, por isso ele é considerado a base para o Sistema de Apoio a Decisão.

- **Precisa ser aceito pela empresa:** diferente de um *Enterprise Resource Planning* (ERP), onde o usuário é obrigado a utilizar o sistema independente de aceitá-lo ou não, o uso do DW torna-se opcional e, dentre os fatores de aceitação, a simplicidade é o principal. O teste de aceitação de um DW dura em torno de 6 meses consecutivos e quando falha, o tempo consumido para modelar o DW, pensando em atender os objetivos descritos anteriormente, acaba sendo desperdiçado.

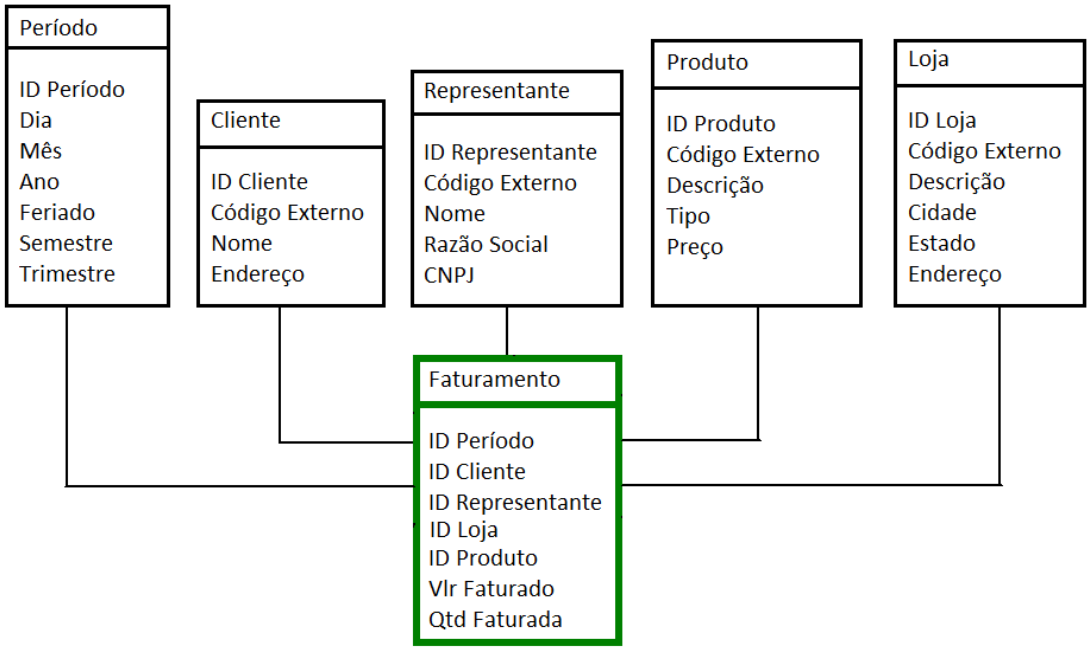
Para construirmos um DW que atenda os objetivos descritos acima não basta termos conhecimento das técnicas e modelagem de bancos de dados, precisamos ter total conhecimento do próprio negócio, dos processos realizados dentro da empresa e daqueles que necessitam da informação.

2.9. Modelo Estrela

De acordo com Colaço Júnior (2004), é uma forma de organizar as tabelas do modelo tradicional (Relacional) para possibilitar a simulação de um DW. Esse modelo foi criado por Dr. Ralph Kimball e, como aspecto básico, tem a forte presença de dados repetitivos, proporcionando um melhor desempenho.

Ele recebeu esse nome porque seu modelo de representação, composto de uma tabela principal chamada de tabela Fato, e várias tabelas de apoio denominadas tabelas Dimensão, lembra muito uma estrela. Veja na Figura 12, um exemplo desse modelo.

Figura 12 – Exemplo de um banco de dados utilizando o modelo Estrela (*Star Schema*)



Fonte: Autor.

A expressão Fato é utilizada na representação de uma medida de negócio, no exemplo apresentado na Figura 12, seria a tabela de Faturamento, onde poderíamos mensurar quanto foi vendido de um produto, em um determinado período, numa determinada loja, por um representante e, para um cliente exclusivo. Essas medições, representadas na Figura 12 pelas colunas “Vlr Faturado” e “Qtd Faturada”, são chamadas de métricas e, são obtidas a partir do cruzamento de todas as dimensões disponíveis (KIMBALL e ROSS, 2002).

Há três operações que podem ser feitas sobre as métricas, são elas: Soma, a operação mais rápida dentre as três, onde os valores são simplesmente somados; Contagem, onde as

métricas são contadas, independente de seus valores; e, Média, onde será apresentada uma média para a métrica. As duas últimas operações exigem mais tempo para serem processadas e, isso dependerá do tamanho do DW disponibilizado (KIMBALL e ROSS, 2002).

2.10. Extract, Transform and Load

Como citado anteriormente, BI é um conjunto de técnicas utilizadas para classificar, extrair, avaliar, supervisionar e distribuir as informações que a gestão de negócios utiliza como base. A técnica, utilizada pelo BI, para extração dos dados é chamada de *Extract, Transform and Load* (ETL) e é nessa etapa que os dados são tratados antes de serem apresentados dentro da ferramenta, ou seja, nessa etapa é realizada a padronização dos dados.

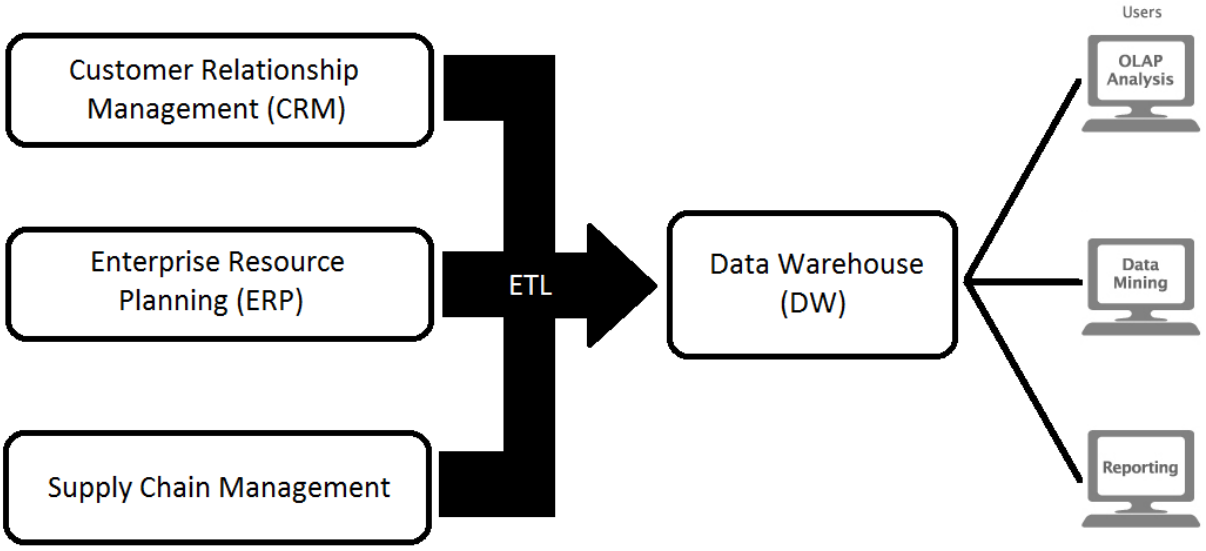
Segundo Ribeiro (2011), o ETL visa trabalhar com qualquer tipo de importação, exportação, migração e transformação dos dados, sendo muito utilizado no processo de carga do *Data Warehouse* (DW), onde é necessário integrar mais de uma fonte de dados. São exemplos de ferramentas de ETL: *Pentaho Data Integration* (PDI) e, *Microsoft SQL Server Integration Services* (MSSIS).

De acordo com Ribeiro (2011), a sigla ETL representa:

- ***Extraction* (Extração):** etapa responsável por extrair os dados dos sistemas de origem;
- ***Transform* (Transformação):** etapa onde é realizada a limpeza dos dados (remoção dos espaços em branco, converter para maiúsculo, entre outras operações);
- ***Load* (Carregamento):** é a etapa final do ETL onde, após a limpeza dos dados, os mesmos são carregados para o DW.

A Figura 13 apresenta a forma de comunicação entre os sistemas da empresa, o processo de ETL e a base do BI.

Figura 13 – Comunicação entre os sistemas da empresa, o ETL e o BI



Fonte: Autor.

2.11. Big Data

De acordo com a empresa SAS (2017), esse termo é atribuído às bases que possuem imensas massas de dados e que, independentemente de sua estruturação, impactam no cotidiano de uma organização. A relevância desse termo se dá em torno do que você pode fazer com seus dados, por exemplo: após analisar os dados de uma origem qualquer, encontrar respostas para diminuir seus custos, desenvolver ou aperfeiçoar seus produtos e, até mesmo tomar decisões mais seguras, analisando melhor as possibilidades.

Conforme Alecrim (2013), em poucas palavras, *Big Data* é utilizado quando há análise de uma imensa quantidade de dados para obtenção de respostas que, seriam mais difíceis de encontrar em análises mais inferiores.

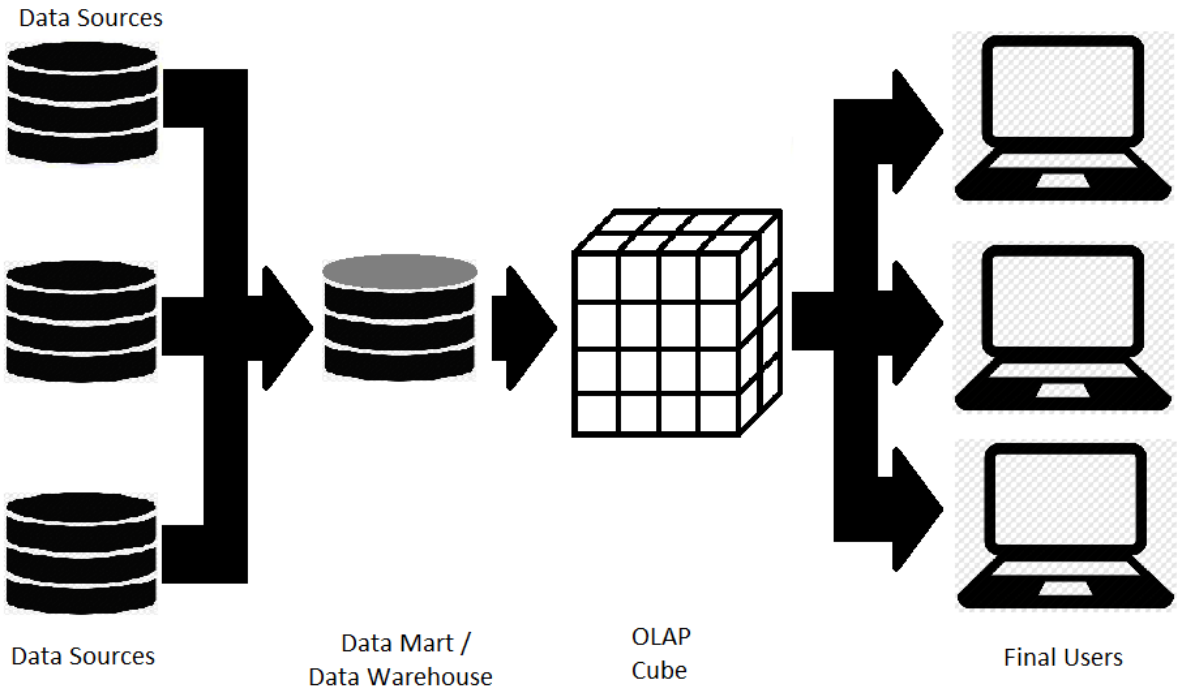
Se uma empresa descobrir como utilizar ao máximo, os seus dados, ela terá em mãos um item muito poderoso, chamado informação. Com este item, ela poderá: entender como aperfeiçoar os seus produtos ou serviços, criar novas e eficientes estratégias de marketing ou aperfeiçoar as já existentes, reduzir custos, reduzir o tempo de produção, ter um melhor aproveitamento dos recursos disponíveis, superar os demais concorrentes de seu ramo de atuação e, entre outras opções, aperfeiçoar seus serviços disponíveis para os clientes (ALECRIM, 2013).

2.12. Online Analytic Processing

Inicialmente, como solução para atender a demanda pela produção de consultas descrevendo, de forma precisa e resumida, as informações sobre os negócios da empresa, foram desenvolvidas ferramentas para produção de relatórios. Entretanto, assim que a informação se tornou o recurso mais poderoso para as empresas e surgiu a infraestrutura dos *Data Warehouse*, veio também a necessidade de uma nova ferramenta capaz de analisar melhor uma maior quantidade de informações, que até então era uma demanda que os relatórios tradicionais não conseguiram atender (COLAÇO JÚNIOR, 2004).

Ao se deparar com grandes volumes de dados, dados históricos, bases não normalizadas e, por consequência, pelo impedimento do uso das ferramentas geradoras de relatórios, a indústria de software se uniu com pesquisadores da área para investir em uma nova ferramenta capaz de lidar com os três primeiros itens. Surge então a tecnologia *Online Analytic Processing* (OLAP) que é um grupo de métodos utilizados para lidar com as informações mantidas em um DW (COLAÇO JÚNIOR, 2004). Na Figura 14 é demonstrado como os dados são processados antes de serem apresentados aos usuários finais.

Figura 14 – Processo de preparação dos dados antes de serem apresentados



Fonte: Autor.

2.12.1. Slice and Dice

É um conjunto de funções utilizadas para formação de dados complexos. Também é considerado um dos principais traços de uma ferramenta OLAP, permitindo ao usuário: alterar a ordem de apresentação das dimensões; possibilitar a troca dos eixos, facilitando a compreensão por parte dos usuários; e analisar melhor as informações de diferentes pontos de vista, permitindo ao usuário a investigação de diferentes inter-relacionamentos. Ele é dividido em quatro operações, são elas: *Ranging*, *Drilling*, *Rotation* e *Ranking* (COLAÇO JÚNIOR, 2004).

2.12.1.1. Ranging

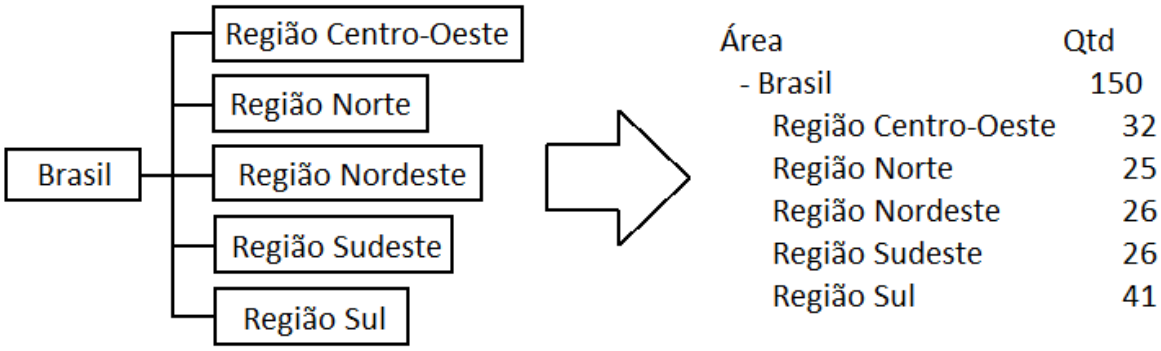
Essa operação é utilizada quando um elemento é inserido ou deletado de um cubo, tornando-se necessário ajustar todas as consultas que foram realizadas a partir desse cubo. Por exemplo: atualmente, tenho uma análise que está me apresentando o valor faturado diário e, após uma nova carga de dados, os valores do dia atual foram disponibilizados, porém não estão sendo apresentados ainda, necessitando ajustes na análise (COLAÇO JÚNIOR, 2004).

2.12.1.2. Drilling

É um conjunto de operações que permite o usuário alternar entre os níveis de granularidade da informação. De acordo com Colaço Júnior (2004), existem três operações OLAP que fazem isso, são elas: *Drill Down*, *Drill Up* e *Drill Across*.

Drill Down é a operação que permite o usuário descer um nível de granularidade, trazendo dados mais atômicos (COLAÇO JÚNIOR, 2004). Supondo que você esteja vendo os dados à nível de país e deseja ver os mesmos dividido entre as regiões. Veja na Figura 15, um exemplo de como funcionaria.

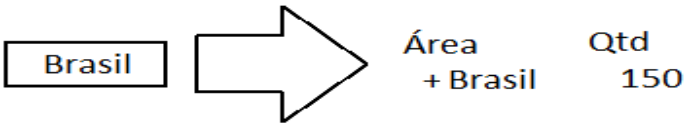
Figura 15 – Demonstração do uso do *Drill Down*



Fonte: Autor.

Drill Up é simplesmente o inverso do *Drill Down*, ou seja, com essa operação vemos os dados agrupados como um todo (COLAÇO JÚNIOR, 2004). A Figura 16 representa a apresentação dos dados caso o usuário utilizasse essa operação.

Figura 16 – Demonstração do uso do *Drill Up*



Fonte: Autor.

Drill Across é a operação utilizada para navegar inclinadamente na estrutura da árvore hierárquica, permitindo a comparação entre membros pertencentes a um mesmo nível (COLAÇO JÚNIOR, 2004). Utilizando o exemplo anterior, a Figura 17 representa o resultado dessa função.

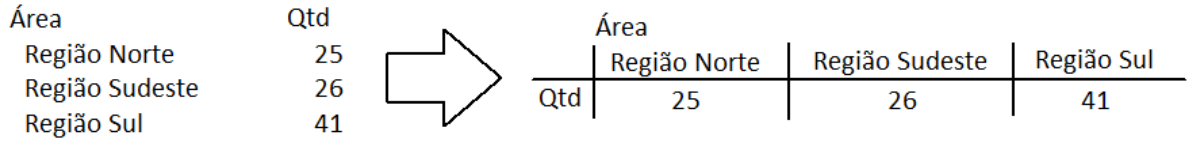
Figura 17 – Demonstração do uso do *Drill Across*

Fonte: Autor.

2.12.1.3. Rotation

De acordo com Colaço Júnior (2004), essa operação possui uma diferença pois, diferente das citadas anteriormente, essa não adiciona e nem remove membros de uma análise e, não utiliza operações de disco do servidor. A função dela é mudar o modo como os dados estão sendo apresentados, por exemplo: a troca de eixos, conforme apresentado na Figura 18.

Figura 18 – Demonstração do uso do *Rotation*

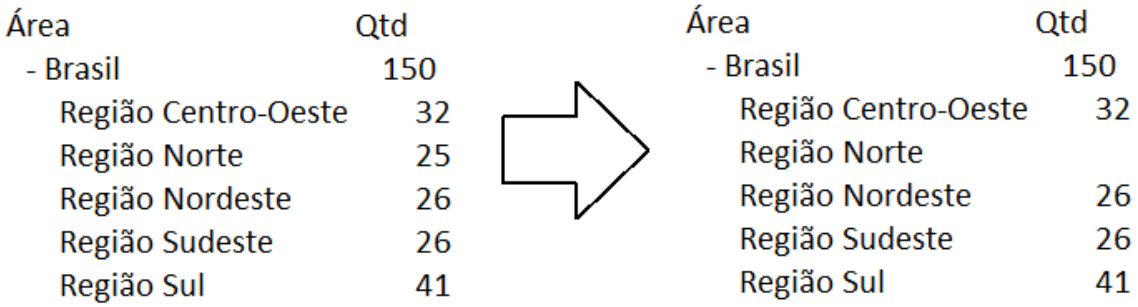


Fonte: Autor.

2.12.1.4. Ranking

Conforme Colaço Júnior (2004) aponta, essa operação é utilizada quando o usuário precisa filtrar as informações que deseja visualizar. Através de uma classificação dos dados, operando diretamente no valor das células, será possível analisar quais foram as três regiões com maior quantidade, podendo assim desconsiderar informações desnecessárias. A Figura 19 apresenta o resultado após a aplicação do filtro das três regiões com maior quantidade.

Figura 19 – Demonstração do uso do *Ranking*



Fonte: Autor

3. METODOLOGIA

Os trabalhos desenvolvidos pela área da informática são caracterizados pela produção de novos acontecimentos, geralmente, envolvendo a construção de um *software*. Contudo, para se obter resultados plenamente satisfatórios, é necessária a aplicação de métodos científicos e, a definição dos critérios de avaliação (WAINER, 2007). Neste capítulo será apresentada a natureza da pesquisa, a abordagem utilizada, as fontes de dados e o procedimento de coleta adotado.

3.1. Delineamento de pesquisa

Realizar testes de performance associados à utilização de diferentes Bancos de Dados (BD) é objetivo principal deste trabalho. Com isso, faz-se necessária a definição de uma estrutura de dados e, implementação da mesma utilizando, para essa pesquisa, as duas bases a serem testadas para assim, avaliarmos qual banco desempenhou os melhores resultados. Como as ferramentas de *Business Intelligence* (BI) precisam propor alto desempenho e alta disponibilidade, nada mais justo do que utilizá-la na avaliação dos dois bancos de dados estudados. Sendo assim, podemos caracterizar esse experimento como uma pesquisa quantitativa.

O objetivo principal desse tipo de pesquisa é assegurar a clareza dos estudos realizados. Para isso o método se baseia na comparação de resultados medidos. É muito comum o uso de dados sintéticos obtidos através da realização de experimentos como, por exemplo, simulações em ambientes controlados ou artificiais (WAINER, 2007).

Após determinado os objetivos desta pesquisa, a mesma pode ser classificada como exploratória pois, como os modelos estudados não são tão explorados pelas empresas e universidades, os mesmos acabam despertando a curiosidade de saber o quão melhor podem ser.

A pesquisa exploratória equivale a produção de um estudo sobre temas relacionados ao projeto de pesquisa com a ideia de familiarizar-se com os mesmos e, além disso, permitir o andamento do trabalho. Este tipo de pesquisa está relacionado com fontes de informação duradouras como, por exemplo, bibliografias e conhecimentos práticos (WAINER, 2007).

Todas as informações citadas para a realização deste trabalho foram obtidas através de pesquisas em livros, sites referentes ao assunto e artigos. Após a realização do referencial teórico foi obtida a base de conhecimento necessária para entendimento dos modelos comparados, caracterizando esse estudo como uma pesquisa bibliográfica.

Segundo Pizzani, Silva, Bello e Hayashi (2012), esta pesquisa refere-se ao levantamento e revisão da literatura existente sobre um determinado tema. Por intervenção da análise e discussão de várias contribuições científicas, teóricas e metodológicas, é levantada a base do conhecimento sobre um específico assunto.

Para a realização do comparativo entre os dois bancos NoSQL, será elaborado um cenário de avaliação, que possibilitará ambas bases serem avaliadas segundo os critérios de avaliação estabelecidos.

Os cenários utilizados nessa pesquisa, serão desenvolvidos em laboratório, ou seja, em um ambiente simulado, onde as variáveis estarão em observação e, em seguida, serão coletadas para qualificação. Este ambiente permitirá saber qual dos dois bancos desempenhou os melhores resultados.

Segundo Marconi e Lakatos (2003), o objetivo desse tipo de pesquisa é relatar e examinar fenômenos desenvolvidos em ambientes controlados, em um laboratório podemos mensurar e observar as variáveis do estudo, o que nos permite chegar aos resultados. Além disso, esse trabalho também utiliza do procedimento experimental, que é o fato do objeto de pesquisa ser exposto às variáveis e exigências controladas dentro do laboratório, permitindo ao pesquisador examinar as saídas geradas sobre o instrumento de pesquisa.

4. FERRAMENTAS AVALIADAS

Durante a busca por ferramentas a serem utilizadas nesse comparativo, foram levantadas algumas opções de Bancos de Dados NoSQL e, algumas opções de ferramentas de BI. Neste capítulo serão apresentados o levantamento dos bancos de dados, o levantamento das ferramentas de BI e, por fim, os critérios que auxiliaram na escolha dos bancos e, da ferramenta de BI.

4.1. Bancos de Dados

Dentre os vários SGBD's NoSQL orientados a colunas e disponibilizados atualmente, três opções foram testadas durante a realização dessa pesquisa, pelo fato de suportarem a linguagem SQL, são elas: Greenplum, MonetDB e, Vertica, que serão apresentadas a seguir:

4.1.1. Greenplum

Greenplum é uma plataforma *Open Source* de banco de dados baseado no famoso PostgreSQL (GREENPLUM, 2017). De acordo com Greenplum (2017), as seguintes inovações foram realizadas:

- **Arquitetura de processamento massivamente paralelo:** a arquitetura do Greenplum fornece paralelização automática de todos os dados e consultas em escala, arquitetura nada compartilhada;

- **Carregamento em escala Petabyte:** O carregamento de alta performance utiliza a tecnologia *Massively Parallel Processing* (MPP), onde há uma escala de velocidades de carga para nós adicionais maiores que 10 terabytes por hora;

- **Otimizador de consulta inovador:** disponibiliza o primeiro otimizador de consulta baseado em custos da indústria para grandes cargas de trabalho de dados;

- **Armazenamento e execução de dados polimórficos:** usuários podem escolher entre o armazenamento orientado a linha, modelo relacional, ou armazenamento orientado a coluna, modelo não relacional;

- **Avançada máquina de aprendizagem:** Fornecido pelo Apache MADlib, uma biblioteca para análise escalável no banco de dados, estendendo os recursos SQL através de funções definidas pelo usuário;

- **Acessibilidade a dados externos:** acesse e consulte todos os seus dados por meio da sintaxe de tabela externa.

4.1.2. Vertica

Vertica foi fundado em 2005 por Michael Stonebraker e Andrew Palmer e, desde o início, foi construído pensando em analisar enormes quantidades de dados, em outras palavras, considerando o uso do termo *Big Data*. No dia 22 de março de 2011, a *Hewlett Packard* (HP) adquiriu os direitos desse banco, o que expandiu seu portfólio de *software* disponibilizado para empresas corporativas e, a partir do dia 1° de setembro de 2017, esse portfólio foi fundido com o *Micro Focus*.

O guia disponibilizado a partir de sua documentação apresenta os conceitos básicos para você começar efetivamente a modelar, construir, operar e manter uma base de dados Vertica. Também assume que você esteja familiarizado com os conceitos básicos e a terminologia dos SGBD Relacionais e, consultas SQL (VERTICA, 2017). De acordo com Vertica (2017), seus principais recursos são:

- **Armazenamento e execução colunares:** o armazenamento colunar oferece ganhos significativos na performance, operações de entrada e saída, espaço de

armazenamento, e eficiência quando se trata de cargas de dados analíticas. Além disso, apenas as colunas necessárias para responder à consulta são utilizadas;

- **Carregamento e consulta em tempo real:** com a alta concorrência de consultas e a capacidade de carregar simultaneamente novos dados no sistema, o Vertica pode carregar dados até 10 vezes mais rápido do que os bancos de dados tradicionais (modelo Relacional);

- **Análise Avançada do Banco de Dados:** dentro do banco, há um conjunto de análises avançadas que permitem a realização dos cálculos de análise mais próximos dos dados. Isso fornece resultados imediatos de um único local sem ter que extrair dados de um ambiente separado;

- **Ferramentas de Administração e Designer de Bancos de Dados:** esses recursos permitem o controle e ajuste das bases de dados do Vertica, com um mínimo esforço de administração;

- **Compressão avançada:** a codificação e compactação agressivas permitem que o Vertica melhore drasticamente o desempenho analítico ao reduzir a CPU, a memória e as entradas e saídas de disco no tempo de processamento. Vertica pode reduzir o tamanho de dados original para até 1/10 do seu tamanho original;

- **Processamento paralelo e massivo:** uma solução robusta e escalável de processamento paralelo que fornece redundância ativa, replicação automática, *failover* (tolerância a falhas) e recuperação.

4.1.3. MonetDB

MonetDB pode ser considerado uma solução pois ele, diferente de um Sistema Gerenciador de Banco de Dados (SGBD) comum, inovou em todas as camadas do SGBD trazendo alguns benefícios como: a utilização da fragmentação vertical para o armazenamento dos dados, uma avançada estrutura para realização de consultas combinadas por *Central Processing Units* (CPU), índices espontâneos e adaptáveis, otimização no tempo de execução das consultas e uma estrutura de *software* modular (MONETDB, 2017).

Baseado no padrão SQL 2003, ele possui suporte completo para *foreign keys* (chaves estrangeiras), *joins* (junções), *views* (visões), *triggers* (gatilhos) e, *stored procedures* (procedimentos armazenados). Além disso, é totalmente compatível com as propriedades ACID e, dentre as interfaces de programação que ele suporta, estão o *Java Database Connectivity* (JDBC), *Open Database Connectivity* (ODBC), *PHP: Hypertext Preprocessor* (PHP), Python, *Ruby on* Rails (RoR), C/C++ e Perl (MONETDB, 2017). De acordo com MonetDB (2017), seus principais recursos disponibilizados são:

- **Kernel de banco de dados para armazenamento colunar:** MonetDB é construído de acordo com a representação das relações de banco de dados, como colunas. São consideráveis entidades trocadas na memória pelo sistema operacional e comprimidas no disco após a necessidade;

- **Sistema de alta performance:** MonetDB distingue-se em aplicativos em que o conjunto de dados produzidos pode ser amplamente mantido na memória principal ou em que algumas colunas de uma tabela relacional ampla são suficientes para lidar com solicitações. Uma maior exploração de algoritmos conscientes do cache provou a validade dessas decisões de design;

- **Motor com poder multi-núcleo:** MonetDB é projetado para execução paralela de vários núcleos, para assim, reduzir o tempo de resposta das consultas complexas em execução;

- **Kernel de banco de dados algébrico e versátil:** MonetDB é projetado para acomodar diferentes linguagens de consulta através de sua linguagem algébrica proprietária, chamada de *MonetDB Assembly Language* (MAL);

- **Um tamanho para todos:** O tamanho do banco de dados máximo suportado pelo MonetDB depende da plataforma de processamento, ou seja, se é um processador de 32 ou 64 bits e, do dispositivo de armazenamento, por exemplo, a tecnologia de armazenamento no disco;

- **Plataforma extensível:** MonetDB tem sido fortemente influenciado pelas experiências científicas para entender a interação entre os algoritmos e requisitos de aplicação, e isso, transformou-o em um sistema de banco de dados extensível com ganchos em todos os níveis na pilha de *software* permitindo por

exemplo, o tradicional encapsulamento de operações extraídas das bibliotecas científicas existentes;

- **Amplo escopo de aplicação:** O MonetDB oferece suporte a uma ampla paleta de domínios de aplicativos, bibliotecas externas como: *Perl Compatible Regular Expressions* (PCRE), Raptor, libxml e GEOS. Vários formatos de arquivo estão sendo encapsulados dentro de cofres de dados, o que cria uma ligação, uma ponte natural entre processamento de banco de dados e processamento baseado em arquivos legados, predominante em alguns domínios científicos;

- **Solução de código aberto:** MonetDB foi desenvolvido durante muitos anos de pesquisa da CWI, estatuto que garante aos usuários, o fácil acesso aos resultados.

4.2. Ferramentas de BI

Como um dos objetivos específicos dessa pesquisa é a escolha de uma ferramenta de BI para avaliações, foi realizada uma busca por essas ferramentas e, em seguida, foi realizada uma análise sobre elas. São três as ferramentas analisadas nesse estudo: Superset, Pentaho Business Analytics e SpagoBI.

4.2.1. Superset

Superset é uma aplicação Web para exploração e visualização dos dados. Além disso, fala muitos dialetos SQL através do SQLAlchemy, um Python *Object-Relational Mapper* (ORM) que é compatível com a maioria dos bancos de dados comuns (GITHUB, 2017). Segundo Github (2017), fornece:

- Uma interface intuitiva para explorar e visualizar conjuntos de dados, e criar painéis interativos;

- Uma grande variedade de visualizações lindas para mostrar seus dados;

- Fácil, de código-livre, o usuário consegue detalhar e fatiar os dados expostos em painéis implícitos. Os painéis e gráficos atuam como ponto de partida para uma análise mais profunda;

- Um editor SQL / IDE de última geração, expondo um rico navegador de metadados e um fluxo de trabalho fácil para criar visualizações de qualquer conjunto de resultados;

- Um modelo de segurança extensível e de alta granularidade que permite regras complexas sobre quem pode acessar quais recursos do produto e conjuntos de dados. Integração com backends de autenticação principais, como: banco de dados, OpenID, LDAP, OAuth e REMOTE_USER;

- Uma camada semântica leve, permitindo controlar como as fontes de dados são expostas ao usuário, definindo dimensões e métricas;

- Suporte para a maioria dos bancos de dados que utilizam a linguagem SQL;

- A integração profunda com o Druid permite que o Superset permaneça rápido durante a exploração dos dados;

- Painéis de controle de carregamento rápido e, com cache configurável.

4.2.2. Pentaho Business Analytics

Pentaho Business Analytics, distribuído pela Pentaho, é uma ferramenta Open-Source que possui uma abordagem moderna, simples e interativa, deixando os usuários empresariais mais capacitados quanto ao processo de análise e cruzamento dos dados e, menos dependentes do setor da TI. De acordo com Pentaho (2017), são algumas características dele:

- **Análise visual e interativa:** Permita que os usuários empresariais sejam autônomos com acesso imediato para analisar e visualizar todos os dados;

- **Dashboards gráficos e responsivos:** Melhora o desempenho organizacional com painéis interativos que podem oferecer aos usuários empresariais indicadores-chave de desempenho, em uma interface visual amigável;

- **Soluções completas para relatórios:** Simplifica o relatório de dados mesclados com recursos de relatório que abrangem todo o *continuum*, desde relatórios interativos de autoatendimento a relatórios corporativos de alto volume e altamente formatados;

- **Gerenciamento de usuários em escala e, incorpora qualquer análise:** Minimiza o número de aplicativos e ganhe mais controle sobre a adoção de análises que incorporam análises. Pentaho Analytics pode ser facilmente administrado com um conjunto de ferramentas para o desenvolvimento eficiente, implantação e gerenciamento de ambientes de usuário.

4.2.3. SpagoBI

SpagoBI é uma suíte inteiramente Open-Source para projetos de BI. Abrange todas as áreas analíticas de projetos de Business Intelligence, com temas e motores inovadores (SPAGOBI, 2017). De acordo com SpagoBI (2017), uma ampla gama de ferramentas analíticas é oferecida, são elas:

- **Relatórios:** Realiza relatórios estruturados e permite exportá-los para os formatos *HyperText Markup* Language (HTML), PDF, XLS, *eXtensible Markup Language* (XML), TXT, *Comma-Separated Values* (CSV) e *Rich Text* Format (RTF);

- **Análise Multidimensional (OLAP):** Permite explorar os seus dados em diferentes níveis de detalhamento e de diferentes perspectivas, através das funções do *Slice and Dice*;

- **Gráficos e Indicadores de Desempenho:** Permite a criação de diversos tipos de gráficos, entre eles: histogramas, diagramas de dispersão e gráficos de pizza. Além disso, oferece um conjunto de ferramentas para criar e gerenciar *Key Performance Indicator* (KPI);

- **Cockpits interativos:** Permite criar visões compostas de várias análises / gráficos, definindo caminhos de navegação e, uma melhor exploração dos dados apresentados;

- **Relatórios específicos:** Permite aos usuários a criação de relatórios compostos de várias folhas, incluindo várias tabelas e gráficos;

- **Inteligência local:** Visualize seus dados de negócio a partir de catálogos de mapas estáticos ou serviços de mapeamento da Web e, interaja dinamicamente para adquirir visualizações instantâneas;

- **Inquérito livre:** Um mecanismo de *Query by Example* (QbE) torna a exploração e a navegação de dados particularmente intuitiva e fácil, graças a uma interface totalmente gráfica e baseada na Web;

- **Mineração de Dados:** Análises avançadas dos dados permitem que você extraia conhecimento a partir de grandes volumes de dados, para melhorar suas tomadas de decisão e, suas estratégias de negócio;

- **Análise da Rede:** Visualize e interprete as relações entre entidades através de vistas especializadas. Uma entidade pode ser animada (por exemplo, usuários de mídia social) ou inanimados (por exemplo, países, empresas, projetos);

- **Integração com o Talend:** Possui integração com o Talend, ferramenta *Open Source* de ETL, permitindo carregar os dados para dentro do DW e, gerenciá-los de acordo com seu interesse;

- **Colaboração:** Crie um portfólio de relatórios estruturados, enriqueça sua análise com anotações pessoais e comentários postados pelos usuários e, em seguida, compartilhe essas análises através de um fluxo de trabalho colaborativo;

- **Automação do escritório:** Publique seus documentos pessoais dentro de seu ambiente de BI, integrando ferramentas comuns de escritório como Open Office e Microsoft Office;

- **Gerenciamento MasterData:** Os usuários podem voltar a escrever no banco de dados e modificar os dados da tabela por meio de uma interface intuitiva, cujo comportamento pode ser definido por meio de uma simples configuração de parâmetros ou, usando modelos pré-definidos;

- **Processos Externos:** Gerencie seus processos analíticos, executando-os em segundo plano ou, informando um período de início e fim de execução.

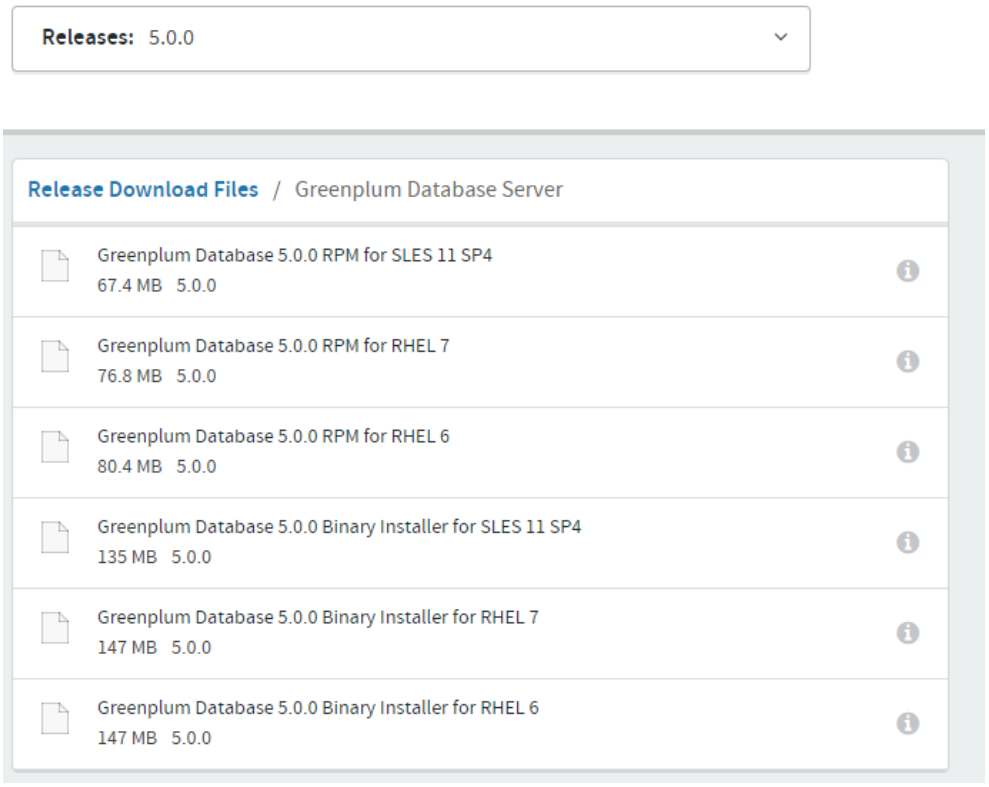
4.3. Avaliação

Numa primeira avaliação, o Superset havia sido a ferramenta escolhida por apresentar uma interface mais simples e bonita. Após a escolha da ferramenta, foi realizada a escolha dos bancos de dados a serem comparados.

Os bancos escolhidos inicialmente foram o Greenplum e o MonetDB. Greenplum foi um dos escolhidos porque é baseado no PostgreSQL, banco de dados aberto com ampla comunidade. O Greenplum permite, além do armazenamento orientado a linha (característica do modelo Relacional), o armazenamento orientado a coluna (característica presente nos modelos NoSQL). Já o MonetDB foi escolhido principalmente por ser o precursor no armazenamento colunar, além de ser um banco *Open Source* e possuir uma documentação rica.

Na etapa de download dos bancos, o Greenplum teve de ser substituído porque não suportava o sistema operacional Ubuntu Linux como servidor do banco, prejudicando a isonomia do *benchmark*. Conforme é apresentado na Figura 20, apenas os sistemas operacionais *SUSE Linux Enterprise Server*, representado pela sigla SLES, e *Red Hat Enterprise Linux*, representado pela sigla RHEL, eram suportados como servidor do banco. Outro fator que influenciou para a troca é o fato do mesmo estar perdendo sua participação no mercado pois, seus clientes acabaram adotando outras tecnologias de banco.

Figura 20 – Servidores suportados pelo Greenplum



Fonte: Retirado da página de download do Greenplum.

Após serem validados os bancos de dados, foi o momento de validar a ferramenta de BI Superset pré-selecionada. Esta ferramenta apresentou erros de código-fonte durante a criação de análises sobre o banco substituto do Greenplum (Vertica) e precisou ser substituída.

No item 4.3.1 serão apresentados a análise e os critérios adotados para a escolha dos bancos de dados. E, no item 4.3.2, serão apresentados a avaliação e os critérios adotados para a escolha da ferramenta de BI.

4.3.1. Critérios de Avaliação dos Bancos de Dados

No primeiro momento, a partir da documentação da ferramenta de BI pré-selecionada (Superset), foram levantados os bancos NoSQL suportados. Em seguida, os mesmos foram classificados de acordo com o tipo de licença disponível, podendo ser apenas Comercial, *Community*, *Open Source* ou, *Community* e *Open Source*.

Logo depois, foi verificado se os bancos possuíam uma documentação e, como a lista de bancos foi extraída da documentação da ferramenta, era garantido o suporte aos bancos. Por

último, com base no ranking gerado a partir do site *DB-Engines* (2017), foram verificadas as posições atuais para cada banco listado, com exceção do Athena que não estava presente.

Figura 21 – Bancos NoSQL suportados pelo Superset

Banco de Dados	Comercial	Community / Open-Source	Doc.	Suporte da ferramenta	Ranking DB-Engines (03/10/2017)
Presto		X	X	X	128°
Redshift	X		X	X	32°
Impala		Open-Source	X	X	31°
Greenplum		Open-Source	X	X	38°
Athena	X		X	X	-
Vertica	X	Community	X	X	26°
ClickHouse		Open-Source	X	X	142°
MonetDB		Open-Source	X	X	104°

Fonte: Autor.

Após levantamento de algumas informações relacionadas aos bancos de dados listados na Figura 21, foram definidas as seguintes restrições para a rejeição de algumas opções apresentadas:

- Licença gratuita e sem um período de tempo definido para os testes, o banco precisa ser *Open Source* ou, pelo menos, disponibilizar uma versão *Community*;
- Sem pesquisas recentes, apenas bancos que não foram utilizados em pesquisas acadêmicas realizadas recentemente, no último semestre mais precisamente;
- Com suporte ao sistema operacional Ubuntu Linux 16.04.

Conforme informado no item 4.3, Greenplum não foi selecionado para essa pesquisa porque não suportava o sistema operacional Linux Ubuntu 16.04, utilizado durante o comparativo. Outro banco que apresentou a mesma situação foi o Impala que, de acordo com o site Confluence (2017), o banco tem suporte apenas para as versões 14.04 e 15.04.

Os bancos Redshift e Athena só possuíam versões Comerciais ou, tinham um período definido de testes, então foram descartados. O banco de dados Clickhouse foi descartado em virtude da existência de um estudo acadêmico recente, no qual ele era comparado com o banco de dados Relacional PostgreSQL, sendo este estudo bastante completo.

Após a eliminação de alguns bancos conforme as restrições acima, três alternativas ainda estavam disponíveis, são elas: Presto, Vertica e, MonetDB. Com base no ranking extraído do site *DB-Engines*, dois bancos foram selecionados: o MonetDB, escolhido desde o início pelo fato de ser o pioneiro no armazenamento colunar e, o Vertica, banco que apresentou a melhor posição do ranking, dentre os bancos listados.

Figura 22 – Análise dos bancos escolhidos

Banco de Dados	Comercial	Community / Open-Source	Doc.	Suporte da ferramenta	Ranking DB-Engines (03/10/2017)
Presto		X	X	X	128º
Redshift	X		X	X	32º
Impala		Open-Source	X	X	31º
Greenplum		Open-Source	X	X	38º
Athena	X		X	X	-
Vertica	**X**	**Community**	**X**	**X**	**26º**
ClickHouse		Open-Source	X	X	142º
MonetDB		**Open-Source**	**X**	**X**	**104º**

Fonte: Autor.

Atualmente, existem muitos bancos de dados NoSQL, MonetDB e Vertica são dois deles que, além de adotarem o modelo Orientado a Colunas, são *Open Source* ou, possuem uma versão *Community*. Quando uma empresa, que possui uma base de dados colossal, utiliza um sistema que trabalhe com essa massa de dados, é normal encontrar problemas de processamento dos dados, sendo necessário realizar uma busca por modelos de bancos de dados de maior performance para tentar resolver esse tipo de problema.

Segundo Tiwari (2011), a forma de armazenamento e manipulação dos dados, adotada por esse modelo de banco de dados, é adequado e traz vantagens para aplicações que utilizam mineração de dados ou analisam e processam quantidades imensas de dados.

4.3.2. Critérios de Avaliação das Ferramentas de BI

Conforme informado no item 4.3, a ferramenta escolhida inicialmente (Superset) conseguiu se conectar com o Vertica, porém apresentou erro ao criar uma nova análise. Sendo assim, foi necessário analisar as outras duas ferramentas de BI escolhidas, comparando os seguintes critérios:

- **Interface amigável:** a ferramenta precisa apresentar uma interface bem intuitiva, ou seja, o usuário não deve encontrar dificuldades durante a utilização;
- **Suporte ao Sistema Operacional:** a ferramenta precisa estar disponível para o sistema operacional utilizado na pesquisa, Ubuntu Linux 16.04;
- **Configuração:** ela precisa ser de fácil implementação e não exigir muitas configurações;
- **Documentação:** precisa disponibilizar a seus usuários uma documentação detalhada e simples de entender;
- **Possuir suporte aos dois bancos de dados:** de nada adianta implantar uma ferramenta se ela não suporta os bancos de dados utilizados nessa pesquisa.

Com base nesses critérios, foram criadas as tabelas apresentadas na Figura 23 e Figura 24. Onde, a primeira apresenta a tabela utilizada para análise e, a segunda, apresenta os resultados da análise.

Figura 23 – Ferramentas de BI analisadas

Critérios	Pentaho Business Analytics	SpagoBI
Interface amigável		
Suporte ao Sistema Operacional		
Configuração		
Documentação		
Suporte ao MonetDB		
Suporte ao Vertica		

Fonte: Autor.

Analisando algumas telas dos dois sistemas, o Pentaho Business Analytics apresentou uma interface de melhor usabilidade e mais bonita, o que garantiu a sua aprovação, nessa exigência.

As duas ferramentas suportam o sistema operacional Linux Ubuntu, portanto ambas foram avaliadas positivamente nesse quesito. No critério de Configuração, quem levou a melhor foi o Pentaho Business Analytics. pois ele só precisa do Java instalado no servidor. Após essa instalação, basta baixar a ferramenta, descompactá-la e iniciar seu serviço.

No critério de Documentação, as duas ferramentas possuem uma documentação, portanto as duas foram aprovadas nesse requisito. Quanto ao Suporte aos dois bancos (MonetDB e Vertica) apenas o Pentaho Business Analytics possui suporte, o que garantiu a sua aprovação nesses dois quesitos.

Em adição a essa análise, de acordo com o ranking do site IT Central Station (2017), dentre as duas ferramentas, o Pentaho Business Analytics obteve as melhores estatísticas, recebendo a 9° posição do ranking e, uma classificação média de 7.9. Já o SpagoBI ficou com a 17° posição e, uma classificação de 7.4.

Com base nessas informações, a tabela apresentada na Figura 24 foi gerada.

Figura 24 – Análise da ferramenta escolhida

Critérios	Pentaho Business Analytics	SpagoBI
Interface amigável	X	
Suporte ao Sistema Operacional	X	X
Configuração	X	
Documentação	X	X
Suporte ao MonetDB	X	
Suporte ao Vertica	X	
Ranking de Ferramentas de BI (IT Central Station)	9°	17°
Classificação Média (IT Central Station)	7.9	7.4

Fonte: Autor.

E, a partir dos resultados apresentados, foi escolhida a nova ferramenta de BI a ser utilizada no comparativo, o Pentaho Business Analytics, que, como destaques, teve uma posição superior no ranking, além de obter uma classificação média superior.

5. O ESTUDO

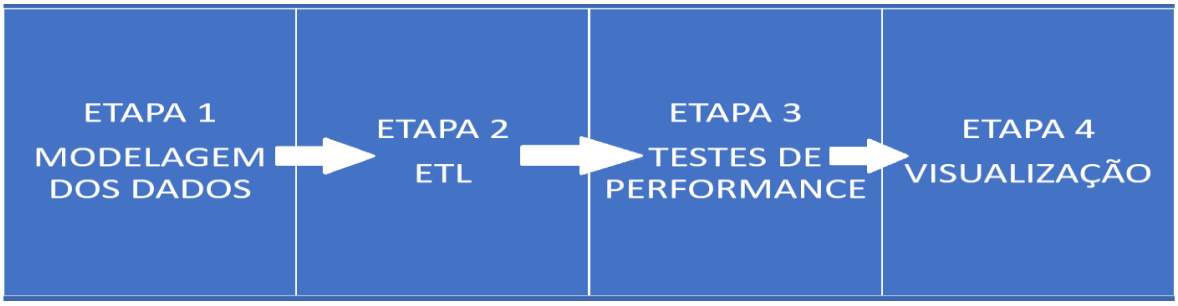
Neste capítulo será apresentado o projeto de pesquisa desenvolvido, apresentando o cenário utilizado, os aspectos que serão avaliados e, as ferramentas utilizadas durante a realização dessa pesquisa.

5.1. Visão Geral

Essa pesquisa tem como objetivo comparar dois bancos de dados NoSQL para avaliar qual banco apresenta os melhores resultados. Como as ferramentas de BI precisam dispor de um maior desempenho e maior disponibilidade, nada mais justo do que modelar um DW utilizando os dois bancos de dados e, com base em alguns critérios fundamentais na modelagem e utilização de um DW, avaliar os mesmos.

O cenário proposto nessa pesquisa fez o uso de um conjunto de dados que foi carregado nos dois bancos. A primeira avaliação foi com base na modelagem de dados, em seguida, a alimentação dos dados. Após essas duas etapas, avaliou-se a performance de cada banco e, por último, a partir da ferramenta de BI, qualificou-se a visualização das informações. Na Figura 25 é apresentado a visão geral desse cenário.

Figura 25 – Cenário de avaliação

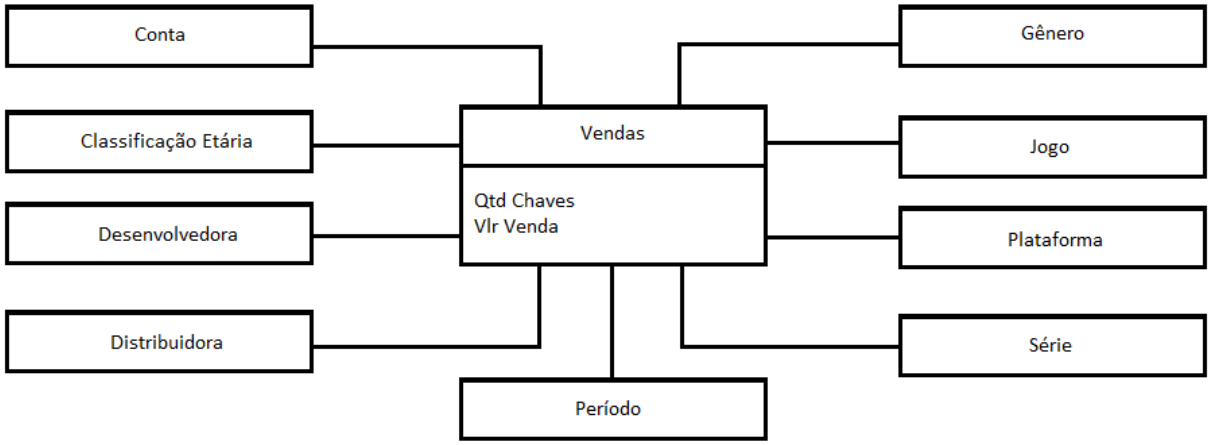


Fonte: Autor.

5.2. Cenário desenvolvido

O cenário de avaliação utilizado foi criado em laboratório, onde teve-se a oportunidade de testar os bancos. A massa de dados inserida dentro das duas bases é fictícia, representando um cubo de vendas específico de uma das áreas de entretenimento digital, a área de jogos. Na Figura 26 são apresentadas as medidas e dimensões utilizadas nesse comparativo.

Figura 26 – Estrutura criada para a pesquisa



Fonte: Autor.

O modelo retratado na Figura 26 contém as duas principais medidas geralmente utilizadas na análise de vendas, são elas: valor e quantidade, onde a quantidade é representada pelo o número de chaves. Estas medidas ficam registradas na tabela fato.

As dimensões expostas serão utilizadas na categorização das medidas, possibilitando detalhar por Conta, Desenvolvedora, Plataforma, Gênero, entre outros. A seguir, será apresentada uma descrição de cada dimensão:

- Classificação Etária: classificação dada para um jogo de acordo com o sistema de classificação adotado pelo país. Segundo O Globo (2017), no Brasil, o conteúdo pode ser classificado como Livre (L), Não recomendado para menores de dez anos (10+), Não recomendado para menores de doze anos (12+), Não recomendado para menores de catorze anos (14+), Não recomendado para menores de dezesseis anos (16+) e, Não recomendado para menores de dezoito anos (18+);

- Conta: a conta representa um cadastro que o usuário precisa realizar em um sistema do tipo *Digital Rights Management* (DRM) ou, sistema gerenciador de direitos autorais. O exemplo real mais famoso atualmente é a Steam, que já registrou um pico de mais de 16 milhões de usuários simultâneos;

- Desenvolvedora: é responsável pelo desenvolvimento do jogo;

- Distribuidora: é responsável por distribuir o jogo;

- Gênero: uma categorização feita de acordo com o conteúdo do jogo. Exemplos reais: Ação, Corrida, Aventura, Simulador, entre outros;

- Jogo: como o próprio nome diz, é o jogo que foi vendido;

- Período: contém informações de quando a venda foi efetuada;

- Plataforma: plataforma a qual o jogo pertence. Exemplos reais: Computador, Xbox 360, Xbox One, entre outros;

- Série: normalmente quando um jogo possui uma sequência, a mesma acaba pertencendo a mesma série de jogos.

Para todas as dimensões foram gerados dados fictícios, exceto para a dimensão de Classificação Etária, que foi preenchida de acordo com o sistema brasileiro de classificação. Com base na estrutura apresentada na Figura 26, as tabelas do DW foram criadas e, logo em seguida, foram carregados os dados das dimensões e, da tabela fato.

Para gerar os dados das Dimensões, foram utilizados o site GenerateData e, para a dimensão de Período e o cubo de Vendas, códigos na linguagem de programação Java. Nessa

etapa, foi possível avaliar algumas formas de carregar os dados nas tabelas e, o tempo utilizado para importação dos dados.

Em seguida, foram avaliadas a performance dos dois bancos e, a partir de uma ferramenta de BI, avaliou-se a visualização dos dados. Para avaliar a performance, consultas foram executadas diretamente nas duas bases e, quanto a visualização, dashboards foram criados a partir de uma ferramenta que tivesse suporte aos bancos comparados, avaliando o desempenho obtido e a compatibilidade. Algumas análises foram elaboradas com o objetivo de reproduzir algumas situações de análise, são elas:

- **Ranking de jogos mais rentáveis:** apresenta ao usuário, os jogos ordenados de acordo com o valor de suas vendas com as vendas em um certo período;
- **Evolução das vendas:** apresenta ao usuário, o desempenho obtido durante os últimos dois anos;
- **Desenvolvedoras menos lucrativas:** apresenta as desenvolvedoras com menores lucros.

5.3. Ferramentas utilizadas

Conforme informado no capítulo 1, o objetivo desta pesquisa é analisar comparativamente a utilização de diferentes Bases de dados NoSQL para *Data Warehouses* (DW), a fim de avaliar sua performance em aplicações de *Business Intelligence*. Logo, para a execução do mesmo, foram selecionados os bancos de dados e, a ferramenta de BI.

Conforme declarado no capítulo 4, foram realizadas análises sobre alguns bancos de dados NoSQL e, de algumas ferramentas *Open Source* de BI, o que auxiliou no processo de escolha dos dois bancos de dados e da ferramenta a serem utilizados no comparativo.

Um dos bancos escolhidos foi o MonetDB, por ser o pioneiro no armazenamento colunar e, o outro foi o Vertica por apresentar a melhor posição dentre os bancos avaliados. Para a ferramenta de BI, o Pentaho Business Analytics foi a escolhida para esse comparativo.

5.4. Aspectos avaliados

Após a elaboração e explicação do cenário a ser utilizado nessa pesquisa, os dois bancos de dados, utilizados no comparativo, foram avaliados em aspectos fundamentais de *Data Warehouses*, são eles:

- **Estrutura:** com base nos fundamentos do modelo estrela, os bancos foram avaliados quanto a organização dos dados a fim de criar um DW;
- **Carga de dados:** os bancos foram avaliados pelos métodos de importação dos dados gerados a partir de sites e, em alguns casos, códigos Java;
- **Performance:** avaliou-se o desempenho obtido através da execução de consultas definidas no item 5.2;
- **Visualização:** através da ferramenta de BI, dashboards foram criados e, cada banco foi avaliado quanto o desempenho obtido e a compatibilidade.

5.4.1. Estrutura

A estrutura de um DW é muito importante pois, com base nas necessidades do cliente, são extraídas as dimensões e métricas que constituíram uma ou mais tabelas fatos. Nas duas bases de dados buscou-se seguir o modelo estrela que, conforme é apresentado no item 2.7.1, é composto de uma tabela central, chamada de Fato e, essa tabela é cercada de suas Dimensões.

Conforme apresentado no item 5.2, a estrutura será composta de nove dimensões, sendo elas: Classificação Etária, Conta, Desenvolvedora, Distribuidora, Gênero, Jogo, Plataforma, Período e Série. Além disso, também fará parte de sua composição, um fato representando as vendas de jogos, onde teremos as medidas Valor Venda e Quantidade de Chaves. O Modelo Entidade-Relacionamento dos dois DWs implementados será apresentado no item 6.2.1.

5.4.2. Carga de dados

O processo de carga das duas bases não foi realizado a partir de uma ferramenta de ETL pois, como os dados são fictícios, não precisaram passar por um processo de limpeza dos dados.

A partir de um site gerador de dados e, códigos na linguagem Java, foram gerados os dados utilizados nessa pesquisa. Esses dados, posteriormente, foram inseridos nos DWs a partir dos comandos de importação disponibilizados pelos próprios bancos de dados. Nessa etapa, foi possível avaliar os tipos de dados disponíveis e, o tempo de importação dos registros.

5.4.3. Performance

Nessa etapa, foi avaliado o tempo de execução de algumas análises definidas no item 5.2, para facilitar na apresentação dos resultados (capítulo 6), elas foram denominadas e classificadas da seguinte maneira:

- **Consulta 1:** Ranking de jogos mais rentáveis;
- **Consulta 2:** Evolução das vendas;
- **Consulta 3:** Desenvolvedoras menos lucrativas.

O Ranking de jogos mais rentáveis tem o objetivo de apresentar os jogos com maior faturamento em um determinado período, ordenados a partir do valor de venda. O resultado desta consulta apresenta o jogo, o valor das vendas e, a quantidade de chaves vendidas.

A evolução das vendas tem como princípio analisar as métricas, retornando por mês e ano, o valor e quantidades faturadas. Assim, os gerentes podem analisar a situação atual da empresa, se ela está ou não aumentando seu faturamento e, em quanto ela está aumentando.

E, a terceira e última análise, apresenta as desenvolvedoras que tiveram os menores lucros. O resultado dessa consulta será composto do nome da desenvolvedora, o valor e quantidades faturadas, apresentando esses dados na ordem crescente, ou seja, os menores faturamentos serão os primeiros apresentados. Essa análise servirá de apoio para os gerentes saberem se vale ou não continuar a investir em jogos dessa desenvolvedora.

5.4.4. Visualização

Nesse tópico, a partir da ferramenta de BI, foi avaliado sua capacidade de usufruir os dados dos dois bancos e, apresentá-los de forma intuitiva. A usabilidade do sistema também foi

avaliada, pensando nos usuários que não possuem conhecimento na área de Tecnologia da Informação (TI) e, precisam criar análises e gráficos para, posteriormente, apresentá-los em uma reunião ou, a seus superiores.

Para facilitar sua usabilidade, a ferramenta deve disponibilizar ao usuário, vários formatos de visualização como: gráficos de barra, pizza, linhas, dentre outros formatos. Além disso, caso o usuário não queira utilizar um formato específico de visualização, a ferramenta deve disponibilizar a opção de apresentar os dados em uma simples listagem, contendo apenas as informações que o usuário necessita.

6. AVALIAÇÃO

Neste capítulo serão demonstrados os resultados coletados durante o desenvolvimento do presente comparativo. Todos os passos realizados serão descritos e, em seguida, seus resultados e, o comportamento das variáveis serão analisados.

6.1. Cenário de testes

O cenário de testes utilizado durante esta pesquisa foi elaborado de forma com que os dois bancos de dados não relacionais fossem comparados sem distinção, ou seja, utilizando a mesma configuração de servidor e, o mesmo conjunto de dados. As configurações do servidor utilizado neste estudo são apresentadas no Quadro 1.

Quadro 1 – Configurações do servidor do cenário de testes

Processador	Intel Core i3 2348M 2.3GHz
Núcleos	2
Memória RAM	4GB
Disco	500 GB Seagate 5400RPM
Sistema Operacional	Ubuntu 16.04 LTS

Fonte: Autor.

Além disso, para a realização do comparativo, foram instaladas as versões 11.27.5 do MonetDB e, 8.1.1-0 do Vertica. Para os dois bancos de dados, foram adotadas suas configurações padrão, ou seja, apenas foram instalados seguindo um passo-a-passo disponibilizado em seus sites.

Dentre os dois bancos instalados, o menos complexo foi o MonetDB, pois sua instalação não apresentou problemas. Já o Vertica, apresentou muitos erros durante sua instalação, mas felizmente, ele possui uma documentação que informa e explica cada erro e, como resolvê-lo. No Quadro 2 são apresentadas as dificuldades encontradas durante a instalação do Vertica e, uma breve explicação de cada uma.

Cada inconveniente apresentado durante a instalação, possuía um link da documentação do banco. A diferença entre esses links era apenas o erro, que ganhou papel de identificador. O link era composto do trecho "https://my.vertica.com/docs/8.1.x/HTML/index.htm#cshid=" e, no final dele, era informado o código do erro. Por exemplo, o link do erro (S0305) era https://my.vertica.com/docs/8.1.x/HTML/index.htm#cshid=S0305.

Quadro 2 – Erros encontrados durante a instalação do Vertica

Erro	**Descrição**
HINT (S0305)	A Micro Focus recomenda que o pacote *tzdata* esteja atualizado antes da instalação do Banco de Dados; O pacote *tzdata* é um banco de dados de fuso horário de domínio público que está pré-instalado na maioria dos sistemas Linux;
HINT (S0040)	Vertica sugere que a seguinte ferramenta esteja instalada, de modo que o suporte possa ajudar a solucionar seu sistema se surgirem proble; O pacote *pstack* (ou *gstack*) exibe o *Stack Trace* ou, Mapa de Execução de um programa / processo em execução;
HINT (S0041)	Outra ferramenta sugerida pelo banco é o pacote *mcelog*, identificado pelo número S0041; Esse pacote é responsável por coletar e decodificar exceções;
HINT (S0045)	Outra sugestão do banco é ter instalado o pacote *sysstat*, que é responsável pelo monitoramento da performance do seu sistema Linux;
WARN (S0112)	O parâmetro do kernel swappiness define a quantidade e a frequência com que o kernel copia os conteúdos da RAM para um espaço de troca; Vertica recomenda um valor de 1;
FAIL (S0140)	Este tópico detalha os vários métodos de escala de frequência da CPU suportados pelo Vertica, porém esse erro pode ser ignorado;
FAIL (S0190)	Vertica requer, no mínimo, 1GB de memória RAM por processador lógico, porém esse erro pode ser ignorado;

FAIL (S0150)	Esse tópico detalha como alterar o escalonamento do disco e, informa que o Vertica requer o escalonamento *deadline*, tentando fornecer uma latência garantida para as requisições, ou *noop*, que utiliza o conceito de fila ou, *First-In First-Out* (FIFO);
FAIL (S0020)	Para a resolução desse problema, o usuário deve alterar a chamada do sistema do kernel Linux que carrega o conteúdo de um arquivo para o cache da página; Vertica requer, no mínimo, 2 GB;
FAIL (S0030)	Antes de instalar o Vertica, é necessário habilitar um dos seguintes sincronizadores de tempo no seu sistema: *chrony* ou NTPD; Para esse comparativo, foi habilitado o NTPD;
FAIL (S0310)	Esse erro informa que é necessário desabilitar as *transparent hugepages*, uma camada de abstração que automatiza a maioria dos aspectos da criação, gerenciamento e uso de páginas enormes;

Fonte: Autor.

Após a instalação dos dois bancos de dados, foram elaborados dois códigos na linguagem de programação Java. Um para geração dos dados da dimensão de Período e, o outro para a geração dos dados do cubo de Vendas. Os dois códigos não possuem parâmetros de entrada pois, todos os dados necessários foram definidos dentro do código.

Num primeiro momento, o objetivo inicial dos *scripts* era gerar um arquivo no formato SQL, composto de uma quantidade de comandos DML de inserção, porém, notou-se que os dois bancos de dados não conseguiam executar eficientemente uma quantidade imensa de comandos DML de inserção, então os scripts foram ajustados para gerar arquivos no formato CSV.

O código para geração dos dados da dimensão de Período gera um arquivo no formato CSV composto das informações a serem armazenadas dentro da dimensão de Período. As informações são geradas de acordo com as datas inicial e final, que são definidas no início do código. Enquanto as duas datas não forem iguais, novas informações são geradas e registradas dentro do arquivo CSV. Além disso, para que o código não entre em um laço de repetição infinito e, não gere vários registros para a mesma data, é somado um dia à data inicial. No Código 1 é apresentado o script utilizado para a dimensão de Período.

Para essa pesquisa, a dimensão de Período foi carregada com dados dentre as datas 01/01/2010 e 31/07/2017, o que resultou na geração de 2.769 registros e, um arquivo com tamanho aproximado de 155 KB.

Código 1 – Script Java para geração dos dados da dimensão de Período

```
package geradimperiodov2;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.GregorianCalendar;

public class GeraDimPeriodoV2 {
    public static void main(String[] args) throws IOException {
        File script1 = new File("dim_periodo.csv");
        script1.createNewFile();

        FileWriter fw_script1 = new FileWriter (script1);
        BufferedWriter bw_script1 = new BufferedWriter (fw_script1);

        Calendar d_inicial = new GregorianCalendar(2010, 0, 1);
        Calendar d_final = new GregorianCalendar(2017, 7, 1);

        SimpleDateFormat formatacao_data = new SimpleDateFormat("yyyy-MM-dd");
        SimpleDateFormat formatacao_ano = new SimpleDateFormat("yyyy");
        SimpleDateFormat formatacao_mes = new SimpleDateFormat("M");
        SimpleDateFormat formatacao_dia = new SimpleDateFormat("d");

        int dia, mes, ano, semana;

        String meses[] = new String[12];
        meses[0] = "JAN";   meses[1] = "FEV";   meses[2] = "MAR";   meses[3] = "ABR";   meses[4] = "MAI";   meses[5] = "JUN";
        meses[6] = "JUL";   meses[7] = "AGO";   meses[8] = "SET";   meses[9] = "OUT";   meses[10] = "NOV";  meses[11] = "DEZ";

        while ( !formatacao_data.format(d_inicial.getTime()).equals( formatacao_data.format(d_final.getTime()) )){
            dia = (Integer) Integer.parseInt(formatacao_dia.format(d_inicial.getTime()));
            mes = (Integer) Integer.parseInt(formatacao_mes.format(d_inicial.getTime()));
            ano = (Integer) Integer.parseInt(formatacao_ano.format(d_inicial.getTime()));
            semana = d_inicial.get(Calendar.WEEK_OF_YEAR);

            bw_script1.write("'" + formatacao_data.format(d_inicial.getTime()) + "'; " + ano + "; " + mes + "; " + dia + "; "
                    + semana + "; " + "'"+meses[mes - 1] + "/" + ano + "'; " + "'SEMANA "+semana+"';");
            bw_script1.newLine();
            d_inicial.add(Calendar.DATE, 1);
        }
        bw_script1.close();
    }
}
```

Fonte: Autor.

Para cada dimensão, foi gerada uma quantidade distinta de registros, conforme é apresentado no Quadro 3. A partir do GenerateData, um site utilizado para gerar dados de acordo com as configurações definidas pelo usuário, foram gerados os comandos DML de inserção para as Dimensões, com exceção das Dimensões Classificação Etária e Período.

Quadro 3 – Quantidade de registros inseridos em cada tabela Dimensão

Dimensão	**Quantidade de Registros**
Classificação Etária	6 registros
Conta	10 registros
Desenvolvedora	75 registros

Distribuidora	100 registros
Gênero	25 registros
Jogo	350 registros
Período	2.769 registros
Plataforma	10 registros
Série	250 registros

Fonte: Autor.

O *script* do cubo de Vendas é um pouco diferente do script apresentado no Código 1. Nele, você define quantos registros devem ser gerados. Cada registro representará uma venda e, será composto das informações a serem armazenadas dentro do cubo de Vendas. Os valores para as chaves estrangeiras foram gerados aleatoriamente com base na quantidade de registros que cada dimensão possui. Assim como o script anterior, esses dados serão registrados em um arquivo no formato CSV. Enquanto a quantidade de registros não for atendida, novos dados serão gerados e inseridos no arquivo. No Código 2 é apresentado o *script* utilizado para o cubo de Vendas.

Código 2 – *Script* Java para geração dos dados do cubo de Vendas

```
package gerafatvendasv2;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.text.DecimalFormat;

public class GeraFatVendasV2 {
    public static void main(String[] args) throws IOException {
        File script2 = new File("fat_vendas.csv");
        script2.createNewFile();

        FileWriter fw_script2 = new FileWriter (script2);
        BufferedWriter bw_script2 = new BufferedWriter (fw_script2);

        int dim_periodo_id, dim_jogo_id, dim_classificacao_etaria_id, dim_serie_id, dim_conta_id,
            dim_distribuidora_id, dim_desenvolvedora_id, dim_genero_id, dim_plataforma_id, qtd_chaves;
        float vlr_venda;

        for(int i=0; i < 100000000; i++){
            dim_classificacao_etaria_id = (int) ((Math.random() * 6) + 1);
            dim_conta_id = (int) ((Math.random() * 10) + 1);
            dim_desenvolvedora_id = (int) ((Math.random() * 75) + 1);
            dim_distribuidora_id = (int) ((Math.random() * 100) + 1);
            dim_genero_id = (int) ((Math.random() * 25) + 1);
            dim_jogo_id = (int) ((Math.random() * 350) + 1);

            dim_periodo_id = (int) ((Math.random() * 2769) + 1);

            dim_plataforma_id = (int) ((Math.random() * 10) + 1);
            dim_serie_id = (int) ((Math.random() * 250) + 1);

            qtd_chaves = (int) ((Math.random() * 10000) + 1);
            DecimalFormat df = new DecimalFormat("#.##");
            vlr_venda = Float.parseFloat(df.format((float) Math.random() * 25000).replace(",", "."));

            bw_script2.write(""+ dim_periodo_id + "; " + dim_jogo_id + "; " + dim_classificacao_etaria_id + "; " + dim_serie_id + "; "
                    + dim_conta_id + "; " + dim_distribuidora_id + "; " + dim_desenvolvedora_id + "; " + dim_genero_id + "; "
                    + dim_plataforma_id + "; " + qtd_chaves + "; " + vlr_venda + "");
            bw_script2.newLine();
        }
        bw_script2.close();
    }
}
```

Fonte: Autor.

Os dados que viabilizaram esse comparativo não foram extraídos de nenhum sistema, e sim, foram gerados de duas formas, conforme é apresentado no Quadro 4. Somente a dimensão de Classificação Etária possui dados reais, pois esses foram gerados manualmente de acordo com o sistema brasileiro de classificações.

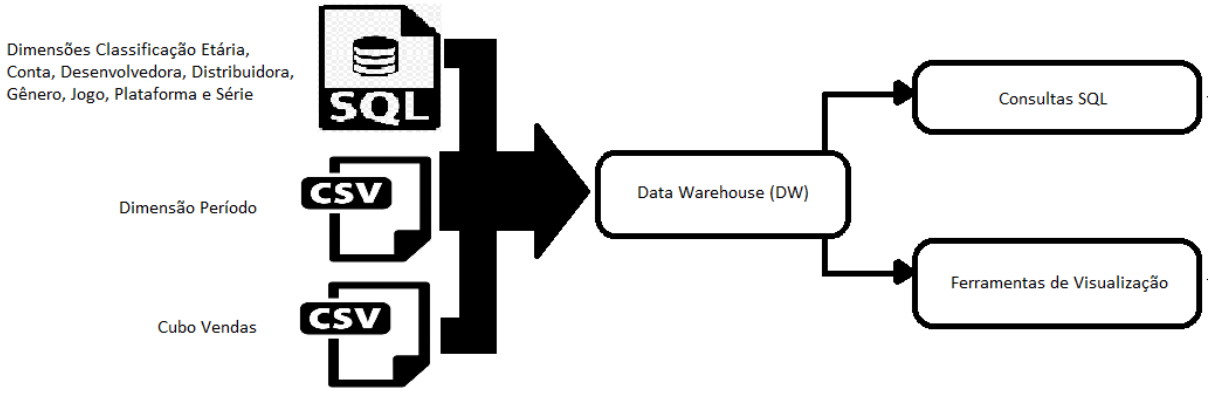
Quadro 4 – Métodos utilizados para geração dos dados

Dimensão	**Método**
Conta, Desenvolvedora, Distribuidora, Gênero, Jogo, Plataforma, Série	Os comandos INSERT's foram gerados a partir do site GenerateData.com
Classificação Etária	Os comandos INSERT's foram gerados manualmente, de acordo com o sistema brasileiro de classificações
Período	A partir de um Script na linguagem de programação Java, foi gerado um arquivo CSV com os dados dessa dimensão
Fato:	
Vendas	A partir de um Script na linguagem de programação Java, foi gerado um arquivo CSV com os dados das vendas

Fonte: Autor.

Após a geração dos dados, o cenário de testes estava preparado para as avaliações, começando pelas etapas de modelagem dos DWs, passando para a etapa de importação dos dados, avançando para os testes de desempenho e, por último, a etapa de visualização. Na Figura 27 é apresentado o cenário de testes dessa pesquisa. Como pode ser visto, foram utilizadas origens no formato SQL, e CSV. O resultado deste Data Warehouse será consumido por consultas SQL e ferramentas de visualização (Pentaho).

Figura 27 – Cenário de testes



Fonte: Autor.

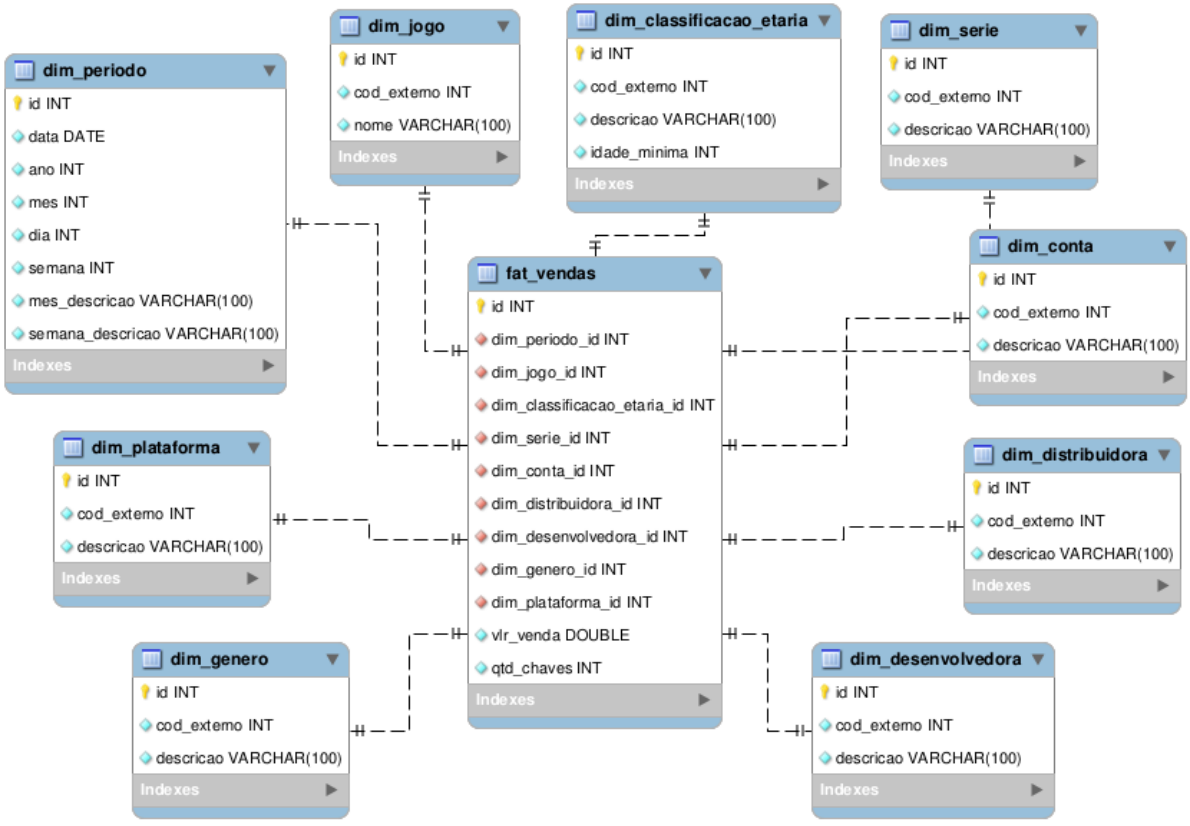
6.2. Análise dos resultados

Neste item serão apresentados os resultados obtidos a partir desse comparativo entre os dois bancos de dados, com base nos aspectos de avaliação determinados no capítulo 5.

6.2.1. Modelagem

Seguindo o modelo especificado no item 5.2, foi modelado o DW para as duas bases NoSQL Orientadas a Colunas, seguindo os conceitos do modelo estrela, onde foram criadas as dimensões Classificação Etária, Conta, Desenvolvedora, Distribuidora, Gênero, Jogo, Plataforma, Período e Série. No meio dessas tabelas, foi criado o cubo Vendas, que está interligada com as demais dimensões por meio de chaves de identificação e, contém as principais métricas de uma venda, que são a Quantidade e o Valor. Na Figura 28 é apresentado o modelo Entidade Relacionamento do DW criado nas duas bases.

Figura 28 – Modelo do DW



Fonte: Autor.

Na etapa de modelagem dos dois DWs, não foram encontradas dificuldades durante a execução, apenas duas alterações precisaram ser feitas para que os comandos rodassem no MonetDB, as mesmas foram destacadas no Quadro 5.

Quadro 5 – Diferenças encontradas no comando DDL dos dois bancos

MonetDB 11.27.5	**Vertica 8.1.1-0**
Comandos DDL	
DROP TABLE IF EXISTS **sys**.dim_classificacao_etaria; CREATE TABLE sys.dim_classificacao_etaria (id **INTEGER AUTO_INCREMENT PRIMARY KEY NOT NULL** , cod_externo INTEGER NOT NULL , descricao VARCHAR(100) NOT NULL , idade_minima INTEGER NOT NULL);	DROP TABLE IF EXISTS **public**.dim_classificacao_etaria; CREATE TABLE public.dim_classificacao_etaria (id **AUTO_INCREMENT PRIMARY KEY NOT NULL** , cod_externo INTEGER NOT NULL , descricao VARCHAR(100) NOT NULL , idade_minima INTEGER NOT NULL);

Fonte: Autor.

Como as alterações feitas para o MonetDB não são tão impactantes a ponto de o usuário perder horas procurando uma solução, isso não acabou lhe afetando na avaliação, mantendo-se lado a lado no comparativo com o Vertica.

Em uma pesquisa realizada por Kober (2017), em que o objetivo do estudo era similar ao do presente trabalho, realizando o comparativo entre o Clickhouse e PostgreSQL, foi descoberto que o banco de dados Clickhouse não suportava várias junções em uma mesma consulta. Para verificar se os dois bancos de dados utilizados nessa pesquisa não tinham essa limitação, uma consulta composta por junções em todas as tabelas Dimensão do modelo do DW foi criada e executada, comprovando que as duas bases não possuem essa limitação.

Para avaliar a modelagem, nas duas bases de dados, foram coletados os tempos de execução dos comandos de criação das tabelas do DW. Os resultados são apresentados no Quadro 6.

Quadro 6 – Tempo de criação das tabelas do DW

Dimensão	**MonetDB 11.27.5**	**Vertica 8.1.1-0**
dim_classificação_etaria	0,053 segundos	0,069 segundos
dim_conta	0,072 segundos	0,039 segundos
dim_desenvolvedora	0,036 segundos	0,043 segundos
dim_distribuidora	0,09 segundos	0,065 segundos
dim_genero	0,07 segundos	0,039 segundos
dim_jogo	0,051 segundos	0,035 segundos
dim_periodo	0,078 segundos	0,049 segundos
dim_plataforma	0,051 segundos	0,042 segundos
dim_serie	0,039 segundos	0,037 segundos
fat_vendas	0,093 segundos	0,078 segundos
TOTAL	**0,633 segundos**	**0,496 segundos**

Fonte: Autor.

Para validar os dados inseridos nas duas bases, duas visões (*views*) foram criadas. Uma delas é a *view* ‘v_confere_dimensoes’ que apresenta, para cada tabela Dimensão, o nome dela, o número de registros inseridos, o menor e, o maior identificador encontrado.

A tabela Fato também ganhou uma *view* para validação. Ela apresenta a quantidade de registros inseridos e, para cada chave estrangeira, são realizadas as operações de mínimo e máximo para identificar possíveis inconsistências. Os comandos de criação das visões são apresentados no Quadro 7.

Quadro 7 – Visões criadas nos dois DWs

v_confere_dimensoes	**v_confere_fato**
Comandos DDL de criação das Views	
CREATE VIEW v_confere_dimensoes AS (SELECT 'dim_classificacao_etaria' AS tab, count(*), min(id), max(id) FROM dim_classificacao_etaria UNION ALL	CREATE VIEW v_confere_fato AS (SELECT 'id' AS info, count(*) AS qtd, min(id) AS min, max(id) AS max FROM fat_vendas UNION ALL SELECT 'dim_classificacao_etaria_id' AS info, NULL AS qtd, min(dim_classificacao_etaria_id) AS in ,

```
SELECT 'dim_conta' AS tab, count(*), min(id), max(id) FROM dim_conta
UNION ALL
SELECT 'dim_desenvolvedora' AS tab, count(*), min(id), max(id) FROM dim_desenvolvedora
UNION ALL
SELECT 'dim_distribuidora' AS tab, count(*), min(id), max(id) FROM dim_distribuidora
UNION ALL
SELECT 'dim_genero' AS tab, count(*), min(id), max(id) FROM dim_genero
UNION ALL
SELECT 'dim_jogo' AS tab, count(*), min(id), max(id) FROM dim_jogo
UNION ALL
SELECT 'dim_periodo' AS tab, count(*), min(id), max(id) FROM dim_periodo
UNION ALL
SELECT 'dim_plataforma' AS tab, count(*), min(id), max(id) FROM dim_plataforma
UNION ALL
SELECT 'dim_serie' AS tab, count(*), min(id), max(id) FROM dim_serie);
```

```
max(dim_classificacao_etaria_id) AS max FROM fat_vendas
UNION ALL
SELECT 'dim_conta_id' AS info, NULL AS qtd, min(dim_conta_id) AS min, max(dim_conta_id) AS max FROM fat_vendas
UNION ALL
SELECT 'dim_desenvolvedora_id' AS info, NULL AS qtd, min(dim_desenvolvedora_id) AS min, max(dim_desenvolvedora_id) AS max FROM fat_vendas
UNION ALL
SELECT 'dim_distribuidora_id' AS info, NULL AS qtd, min(dim_distribuidora_id) AS min, max(dim_distribuidora_id) AS max FROM fat_vendas
UNION ALL
SELECT 'dim_genero_id' AS info, NULL AS qtd, min(dim_genero_id) AS min, max(dim_genero_id) AS max FROM fat_vendas
UNION ALL
SELECT 'dim_jogo_id' AS info, NULL AS qtd, min(dim_jogo_id) AS min, max(dim_jogo_id) AS max FROM fat_vendas
UNION ALL
SELECT 'dim_periodo_id' AS info, NULL AS qtd, min(dim_periodo_id) AS min, max(dim_periodo_id) AS max FROM fat_vendas
UNION ALL
SELECT 'dim_plataforma_id' AS info, NULL AS qtd, min(dim_plataforma_id) AS min,
```

	max(dim_plataforma_id) AS max FROM fat_vendas UNION ALL SELECT 'dim_serie_id' AS info, NULL AS qtd, min(dim_serie_id) AS min, max(dim_serie_id) AS max FROM fat_vendas);

Fonte: Autor.

Em cada banco de dados utilizado no comparativo, também foram coletados os tempos de execução dos comandos de criação das visões do DW. Os resultados são apresentados no Quadro 8.

Quadro 8 – Tempo de criação das visões do DW

Visão	**MonetDB 11.27.5**	**Vertica 8.1.1-0**
v_confere_dimensoes	0,125 segundos	0,078 segundos
v_confere_fato	0,444 segundos	0,081 segundos
TOTAL	**0,569 segundos**	**0,159 segundos**

Fonte: Autor.

Com base nos resultados obtidos, notou-se que o Vertica obteve os melhores resultados, apresentando uma redução de 21,64% no tempo de criação das tabelas e, uma redução de 72,06% no tempo de criação das visões.

6.2.2. Carga de Dados

A carga de dados buscou avaliar alguns métodos de importação e o tempo de importação para inserir os dados dentro dos dois DWs. Este processo pode ser executado a partir de ferramentas de ETL, scripts de uma determinada linguagem de programação, comandos DML de inserção ou até mesmo, importação de arquivos de formato texto ou CSV.

Para todas as tabelas, o método de importação foi a execução de comandos DML de inserção, com exceção da dimensão de Período e, do cubo de Vendas, que utilizaram o método

de importação a partir de arquivos CSV. No Quadro 9 são apresentados os tempos de execução dos comados DML de inserção.

Quadro 9 – Tempo de inserção dos dados nas tabelas Dimensão

Dimensão	**MonetDB 11.27.5**	**Vertica 8.1.1-0**	**Nro de Registros**
dim_classificação_etaria	0,177 segundos	0,268 segundos	6 registros
dim_conta	0,301 segundos	0,250 segundos	10 registros
dim_desenvolvedora	1,847 segundos	1,233 segundos	75 registros
dim_distribuidora	2,496 segundos	1,658 segundos	100 registros
dim_genero	0,617 segundos	0,413 segundos	25 registros
dim_jogo	8,648 segundos	5,214 segundos	350 registros
dim_plataforma	0,229 segundos	0,177 segundos	10 registros
dim_serie	6,682 segundos	3,724 segundos	250 registros
TOTAL	**20,997 segundos**	**12,937 segundos**	**826 registros**

Fonte: Autor.

Como esse comparativo pretende avaliar os dois bancos de dados utilizando três quantidades distintas de registros para o cubo de Vendas, três arquivos foram gerados durante a pesquisa. Assim, será possível verificar se o tamanho das bases influencia nos resultados da pesquisa. No item 6.2.3 são apresentados os resultados obtidos para as três quantidades analisadas.

É interessante citar que, para o MonetDB, não foi possível fazer a importação dos dois arquivos CSV, sem antes, remover a coluna chave da tabela em questão (coluna ‘id’). Ao acessar sua própria documentação, não foi encontrado um exemplo que atendesse esse caso e, no Stackoverflow (2016), foram identificadas duas maneiras de solucionar esse imprevisto, são elas:

- **Remover a coluna chave:** essa opção necessitava que a coluna chave da tabela fosse removida antes da importação dos arquivos e, após a importação, fosse executado um comando DDL de alteração na tabela, adicionando a coluna chave;

- **Criar uma tabela temporária:** a outra opção encontrada seria a de criar uma tabela temporária, limitada à sessão do usuário, composta dos dados do arquivo e, em seguida, fazer a migração da tabela temporária para a tabela em questão;

Após a realização desse pequeno levantamento, o primeiro método foi o escolhido porque os comandos DDL de alteração foram executados em menos de dois minutos. No Quadro 10 são apresentados os comandos utilizados para importação dos dados e, no Quadro 11 são apresentados os tempos de importação de cada arquivo, a quantidade de registros inseridos e, o tamanho dos arquivos gerados.

Quadro 10 – Comandos de importação dos dados da dimensão Período e Cubo Vendas

Tabela	**MonetDB 11.27.5**
dim_periodo	COPY INTO dim_periodo FROM '/home/giordani/NetBeansProjects/GeraDimPeriodoV2/dim_periodo.csv' USING DELIMITERS '; ','\n','\'';
fat_vendas (1.000.000 registros)	COPY 1000000 RECORDS INTO fat_vendas FROM '/home/giordani/NetBeansProjects/GeraFatVendasV2/fat_vendas3.csv' USING DELIMITERS '; ','\n';
fat_vendas (50.000.000 registros)	COPY 50000000 RECORDS INTO fat_vendas FROM '/home/giordani/NetBeansProjects/GeraFatVendasV2/fat_vendas2.csv' USING DELIMITERS '; ','\n';
fat_vendas (100.000.000 registros)	COPY 100000000 RECORDS INTO fat_vendas FROM '/home/giordani/NetBeansProjects/GeraFatVendasV2/fat_vendas.csv' USING DELIMITERS '; ','\n';
Tabela	**Vertica 8.1.1-0**
dim_periodo	COPY dim_periodo (data, ano, mes, dia, semana, mes_descricao, semana_descricao) FROM '/home/giordani/NetBeansProjects/GeraDimPeriodoV2/dim_periodo.csv' DELIMITER ';';
fat_vendas (1.000.000 registros)	COPY fat_vendas (dim_periodo_id, dim_jogo_id, dim_classificacao_etaria_id, dim_serie_id, dim_conta_id, dim_distribuidora_id, dim_desenvolvedora_id, dim_genero_id, dim_plataforma_id, qtd_chaves, vlr_venda) FROM

	'/home/giordani/NetBeansProjects/GeraFatVendasV2/fat_vendas3.csv' DELIMITER ';';
fat_vendas (50.000.000 registros)	COPY fat_vendas (dim_periodo_id, dim_jogo_id, dim_classificacao_etaria_id, dim_serie_id, dim_conta_id, dim_distribuidora_id, dim_desenvolvedora_id, dim_genero_id, dim_plataforma_id, qtd_chaves, vlr_venda) FROM '/home/giordani/NetBeansProjects/GeraFatVendasV2/fat_vendas2.csv' DELIMITER ';';
fat_vendas (100.000.000 registros)	COPY fat_vendas (dim_periodo_id, dim_jogo_id, dim_classificacao_etaria_id, dim_serie_id, dim_conta_id, dim_distribuidora_id, dim_desenvolvedora_id, dim_genero_id, dim_plataforma_id, qtd_chaves, vlr_venda) FROM '/home/giordani/NetBeansProjects/GeraFatVendasV2/fat_vendas.csv' DELIMITER ';';

Fonte: Autor.

No Quadro 10 podemos notar algumas diferenças encontradas entre os comandos de importação utilizados pelo MonetDB e, pelo Vertica. A primeira delas é que, para o MonetDB, não é possível informar as colunas que os dados do arquivo representam, ou seja, a tabela de destino precisa ser composta apenas das colunas que compõem o arquivo a ser importado e, além disso, precisam ter a mesma ordem dos campos, caso contrário não será possível importar os dados.

A segunda diferença encontrada é que, quando o arquivo a ser importado pelo MonetDB possui uma quantidade imensa de dados, é altamente recomendável informar a quantidade aproximada de registros que compõe o arquivo. Essa superestimação permite que o servidor de banco de dados aloque espaço suficiente para a tabela, com antecedência.

E, os delimitadores, que para importar os dados da Dimensão de Período, no MonetDB, foi necessário informar o delimitador de coluna (representado pelo trecho ‘; ‘), o delimitador de registro (representado pelo trecho ‘\n’) e, além desses dois, foi utilizado um terceiro delimitador para os campos cujo dado era representado entre aspas simples (representado pelo trecho ‘\’’). Durante a importação do arquivo das vendas, apenas os delimitadores de registro e, de coluna foram utilizados pois, o arquivo de vendas não possuía nenhuma informação entre aspas simples.

Quadro 11 – Tempo de importação dos dados na dimensão Período e Cubo Vendas

Tabela	**MonetDB 11.27.5**	**Vertica 8.1.1-0**	**Nro de Registros**	**Tamanho do Arquivo**
dim_periodo	0,546 segundos	0,277 segundos	2.769 registros	153,9 KB
fat_vendas	5,4 segundos	2,755 segundos	1.000.000 registros	49,8 MB
fat_vendas	2.665 segundos	234,59 segundos	50.000.000 registros	2,5 GB
fat_vendas	5.424 segundos	662,28 segundos	100.000.000 registros	5,0 GB

Fonte: Autor.

Após a importação de cada arquivo gerado com os dados do cubo de Vendas, foi verificado o tamanho que as duas bases estavam ocupando em disco. No Quadro 12 são apresentados os métodos de verificação do tamanho dos bancos e, no Quadro 13, são apresentados os resultados obtidos.

Quadro 12 – Métodos utilizados para verificar o espaço ocupado

MonetDB 11.27.5
Dentro do diretório onde o banco de dados está armazenado, foi executado o comando '**du -sh**'. Durante a pesquisa, o banco esteve armazenado no diretório '**/path/to/mydbfarm**'
Vertica 8.1.1-0
Para verificar o espaço em disco utilizado por esse banco, foi necessária a execução da consulta abaixo: **SELECT /*+ label(estimated_raw_size)*/ pj.anchor_table_schema, pj.used_compressed_gb, pj.used_compressed_gb * la.ratio AS raw_estimate_gb FROM (SELECT ps.anchor_table_schema, SUM(used_bytes) / (1024^3) AS used_compressed_gb FROM v_catalog.projections p JOIN v_monitor.projection_storage ps ON ps.projection_id = p.projection_id WHERE p.is_super_projection = 't' GROUP BY ps.anchor_table_schema) pj CROSS JOIN (SELECT (SELECT database_size_bytes FROM v_catalog.license_audits WHERE audited_data = 'Regular' ORDER BY audit_start_timestamp DESC LIMIT 1) / (SELECT**

SUM(used_bytes) FROM V_MONITOR.projection_storage) AS ratio) la ORDER BY pj.used_compressed_gb DESC;

Fonte: Autor.

Quadro 13 – Tamanho das bases de dados, após a importação de cada arquivo

Tabela	**MonetDB 11.27.5**	**Vertica 8.1.1-0**
fat_vendas (1.000.000 reg.)	118 MB	97,220 MB
fat_vendas (50.000.000 reg.)	5,6 GB	1,016 GB
fat_vendas (100.000.000 reg.)	12 GB	2,022 GB

Fonte: Autor.

Com base nesses resultados, novamente o Vertica foi o melhor qualificado. No Quadro 14 é apresentado um resumo de seus resultados, onde o cálculo do percentual de redução é quanto o tempo obtido pelo Vertica representa sobre o tempo obtido pelo MonetDB, subtraindo-se um para saber qual a porcentagem exata de redução.

Quadro 14 – Resultados obtidos pelo Vertica

Operação	**MonetDB 11.27.5**	**Vertica 8.1.1-0**	**% de Redução**
(em segundos)			
Inserção dos dados nas dimensões	20,997 segundos	12,937 segundos	38,39%
Importação dos dados da dimensão de Período	0,546 segundos	0,277 segundos	49,27%
Importação dos dados do cubo de Vendas (1.000.000 registros)	5,4 segundos	2,755 segundos	48,98%
Importação dos dados do cubo de Vendas (50.000.000 registros)	2665 segundos	234,59 segundos	91,20%
Importação dos dados do cubo de Vendas (100.000.000 registros)	5424 segundos	662,28 segundos	87,79%
(em Megabytes - MB)			
Tamanho das bases de Dados (1.000.000 registros)	118 MB	97,220 MB	17,61%

Tamanho das bases de Dados (50.000.000 registros)	5.600 MB	1.016 MB	81,86%
Tamanho das bases de Dados (100.000.000 registros)	12.000 MB	2.022 MB	83,15%

Fonte: Autor.

Também se percebe que o poder de compressão dos dados disponível pelo Vertica é superior ao disponibilizado pelo MonetDB, já que o tamanho das bases tem uma diferença de 83,15%.

6.2.3. Performance

A etapa de avaliação da performance dos dois bancos de dados foi dividida em três sub-etapas, onde foram coletados o tempo de execução, o uso da memória e, o uso do processador. Na primeira sub-etapa, os tempos de execução das consultas definidas nessa pesquisa serão coletados logo após a inserção de 100 milhões de registros. Na segunda sub-etapa, os tempos também serão coletados, porém, será utilizada uma base com 50 milhões de registros. E, por último, será realizada uma nova coleta dos tempos utilizando uma base com 1 milhão de registros.

No item 5.4.3 foram definidas as consultas a serem utilizadas nesse comparativo. Nenhuma delas precisou ser ajustada antes da execução, ou seja, a mesma consulta foi executada nas duas bases. Todas as consultas pertencem a linguagem SQL e, possuem um filtro de dados, com o objetivo de representar análises reais das informações. No Quadro 15, são apresentadas as consultas executadas durante a avaliação.

Quadro 15 – Consultas executadas durante a avaliação

Consulta 1 - Ranking dos Jogos mais rentáveis
SELECT j.nome AS jogo, SUM(f.vlr_venda) AS vlr_total_vendido, SUM(f.qtd_chaves) AS qtd_total_vendido FROM fat_vendas f INNER JOIN dim_jogo j ON (f.dim_jogo_id = j.id) INNER JOIN dim_periodo p ON (f.dim_periodo_id = p.id) WHERE p.data >= '2017-01-01' GROUP BY j.nome ORDER BY 2 DESC;
Consulta 2 - Evolução das Vendas

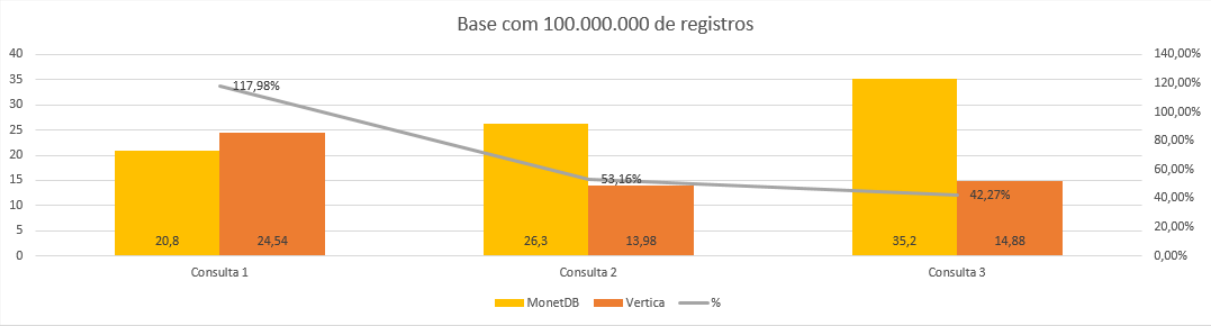
SELECT p.mes_descricao AS mês, SUM(f.vlr_venda) AS vlr_total_vendido, SUM(f.qtd_chaves) AS qtd_total_vendido, p.ano, p.mes FROM fat_vendas f INNER JOIN dim_periodo p ON (f.dim_periodo_id = p.id) WHERE p.data >= '2016-01-01' GROUP BY p.mes_descricao, p.ano, p.mes ORDER BY 4, 5;
Consulta 3 - Desenvolvedoras menos lucrativas
SELECT d.descricao AS desenvolvedora, SUM(f.vlr_venda) AS vlr_total_vendido, SUM(f.qtd_chaves) AS qtd_total_vendido FROM fat_vendas f INNER JOIN dim_desenvolvedora d ON (f.dim_desenvolvedora_id = d.id) INNER JOIN dim_periodo p ON (f.dim_periodo_id = p.id) WHERE p.data >= '2016-01-01' GROUP BY d.descricao ORDER BY 2;

Fonte: Autor.

A coleta dos tempos de execução de cada consulta foi realizada diretamente dos clientes de cada banco de dados. Todas as vezes que foi necessário conectar no cliente do Vertica para coletar um tempo de execução, também foi necessário executar o comando '**\t**', para habilitar o contador. Enquanto que, para o MonetDB o contador já vinha habilitado sem a execução de nenhum comando.

Na Figura 29 são apresentados os resultados obtidos após a execução das três consultas em uma base com 100 milhões de registros. As colunas do gráfico representam os tempos, em segundos, de execução em cada banco e, a linha representa o percentual do tempo que o Vertica consumiu com base no tempo consumido pelo MonetDB.

Figura 29 – Tempos das Consultas Analíticas MonetDB X Vertica



Fonte: Autor.

Notou-se que o Vertica, na maioria das consultas, apresentou os melhores tempos, tendo destaque na consulta 2, que utilizou menos da metade do tempo obtido para o MonetDB. Já na consulta 3, o Vertica obteve um tempo 57,73% menor e, na consulta 1, ele acabou obtendo um resultado pior por conta da cláusula de ordenação utilizada na consulta e, pela quantidade de registros presentes na dimensão de Jogos.

Para melhor avaliar esse critério, três novas consultas foram criadas e executadas, com o intuito de lerem a base completa que, na primeira sub-etapa da avaliação, era composta de 100 milhões de registros. A primeira consulta é, basicamente, uma contagem dos registros presentes na tabela Fato, agrupados por gênero, a segunda uma contagem dos registros, a soma dos valores e a soma das chaves vendidas, agrupados por série, e a terceira uma contagem dos registros, soma dos valores, soma das chaves vendidas, agrupados por gênero e ordenados pela soma das chaves.

No Quadro 16, são apresentadas as novas consultas utilizadas durante a avaliação e, na Figura 30 são apresentados os tempos de execução das consultas que usufruíram da base completa. As colunas, novamente representam os tempos em segundos, obtidos pelos dois bancos de dados desse comparativo e, a linha representa o percentual do tempo que o Vertica utilizou com base no tempo consumido pelo MonetDB.

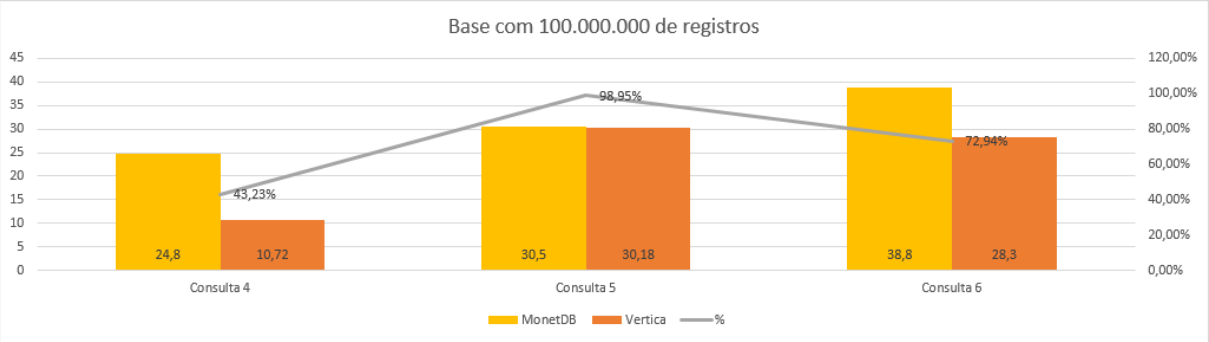
Quadro 16 – Consultas criadas durante a avaliação

Consulta 4 – Contagem, agrupado por Genêro
SELECT g.descricao AS genero, COUNT(f.id) FROM fat_vendas f INNER JOIN dim_genero g ON (f.dim_genero_id = g.id) INNER JOIN dim_periodo p ON (f.dim_periodo_id = p.id) WHERE p.data >= '2017-01-01' GROUP BY g.descricao
Consulta 5 – Contagem e Soma das duas métricas, agrupados por Série
SELECT s.descricao AS serie, COUNT(f.id), SUM(f.vlr_venda), SUM(f.qtd_chaves) FROM fat_vendas f INNER JOIN dim_serie s ON (f.dim_serie_id = s.id) INNER JOIN dim_periodo p ON (f.dim_periodo_id = p.id) WHERE p.data >= '2015-01-01' GROUP BY s.descricao

Consulta 6 – Contagem e Soma das duas métricas, agrupados por Genêro e ordenados pela soma das chaves
SELECT g.descricao AS genero, COUNT(f.id), SUM(f.vlr_venda), SUM(f.qtd_chaves) FROM fat_vendas f INNER JOIN dim_genero g ON (f.dim_genero_id = g.id) INNER JOIN dim_periodo p ON (f.dim_periodo_id = p.id) WHERE p.data >= '2016-10-01' AND p.data <= '2017-03-31' GROUP BY g.descricao ORDER BY 4

Fonte: Autor.

Figura 30 – Tempos das Consultas MonetDB X Vertica com 100 milhões de registros

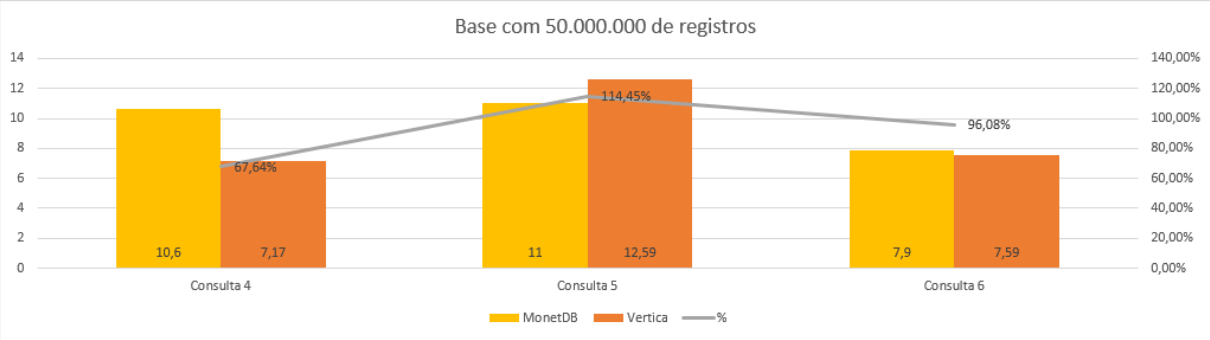


Fonte: Autor.

Novamente podemos notar que o Vertica, na maioria das consultas, apresentou os melhores tempos, tendo destaque na consulta 4, que utilizou menos da metade do tempo obtido para o MonetDB. Já na consulta 6, o Vertica obteve um tempo 27,06% menor e, na consulta 5, ele acabou obtendo um resultado muito próximo do obtido pelo MonetDB, com uma redução de 1,05%.

Conforme relatado no início desse item (6.2.3), os mesmos testes foram realizados em bases com menor número de registros. Na Figura 31 são apresentados os resultados obtidos a partir de uma base com 50 milhões de registros.

Figura 31 – Tempos das Consultas MonetDB X Vertica com 50 milhões de registros

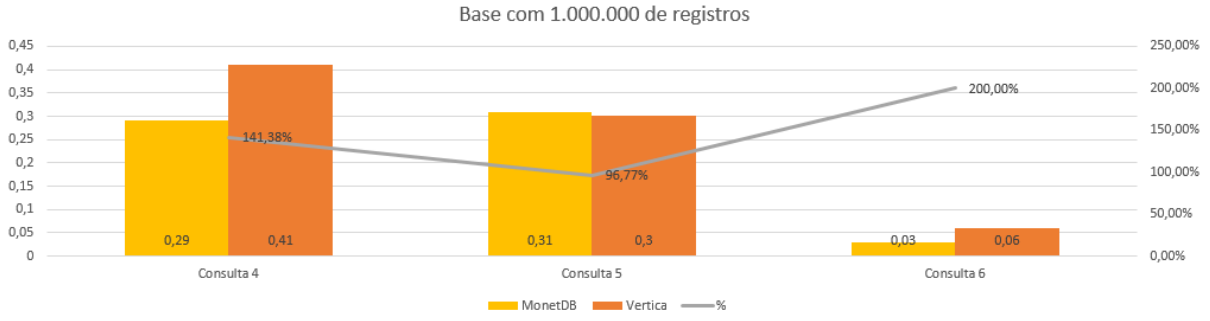


Fonte: Autor.

Dessa vez, podemos notar que o Vertica não foi tão bem, tendo o maior destaque na consulta 4, com uma redução de 32,36% com base no tempo obtido para o MonetDB. Já na consulta 6, o Vertica obteve um tempo 1,92% menor e, na consulta 5, ele acabou obtendo um tempo de execução 14,45% superior ao tempo obtido pelo MonetDB.

Após a realização dos testes em uma base de 50 milhões de registros, foram realizados novos testes utilizando uma base com 1 milhão de registros. Na Figura 32 são apresentados os resultados obtidos a partir dessa nova base.

Figura 32 – Tempos das Consultas MonetDB X Vertica com 1 milhão de registros



Fonte: Autor.

Dessa vez, podemos notar que o MonetDB obteve os melhores resultados, tendo o maior destaque na consulta 6, com uma redução de 50% com base no tempo obtido para o Vertica. Já na consulta 4, o MonetDB obteve um tempo 41,38% inferior e, na consulta 5, ele acabou obtendo um tempo de execução 3,23% superior ao tempo obtido pelo Vertica.

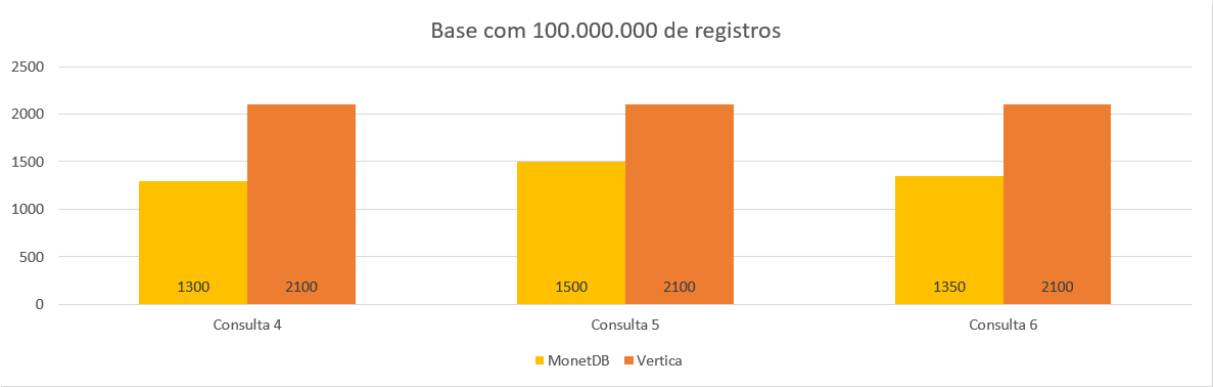
Durante a análise dos resultados obtidos até então, na maioria dos testes notou-se que, quanto maior o tamanho da base, menor é o tempo de execução obtido no Vertica. Utilizando como base os resultados da consulta 6, notamos que a primeira análise apresentou uma redução

de 27,06%, o segundo percentual apresentou uma redução de 3,92% e, no terceiro, acabou obtendo um crescimento de 100%.

Isso pode ser justificado por conta do seu nível de compressão dos dados, já que em termos de espaço em disco, o Vertica utiliza menos da metade do espaço usado pelo MonetDB. Resumindo, como o nível de compressão dos dados é superior, mais tempo ele deve utilizar para descomprimir e apresentar os dados.

Durante a realização das três sub-etapas, também foram coletados o consumo da memória disponível no servidor e, o consumo da CPU, observando os pontos de pico da memória do servidor (em *megabytes*), do processador como um todo e, do indicador médio de carga do sistema (*load average*). Na Figura 33, são apresentados os consumos de memória do servidor, obtidos durante os testes com a base de 100 milhões de registros.

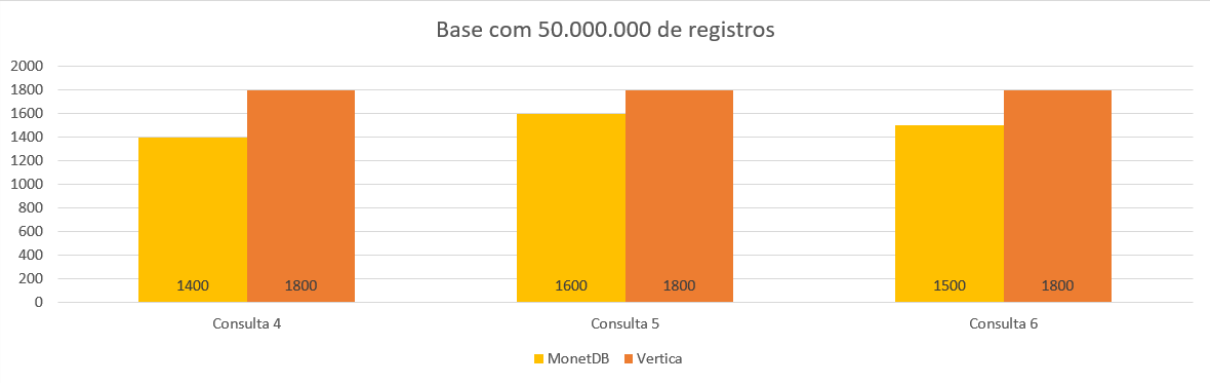
Figura 33 – Consumo da memória MonetDB X Vertica com 100 milhões de registros



Fonte: Autor.

Na Figura 34 são apresentados os consumos de memória do servidor, obtidos durante os testes realizados nos dois bancos de dados, após a base ser preenchida com 50 milhões de registros.

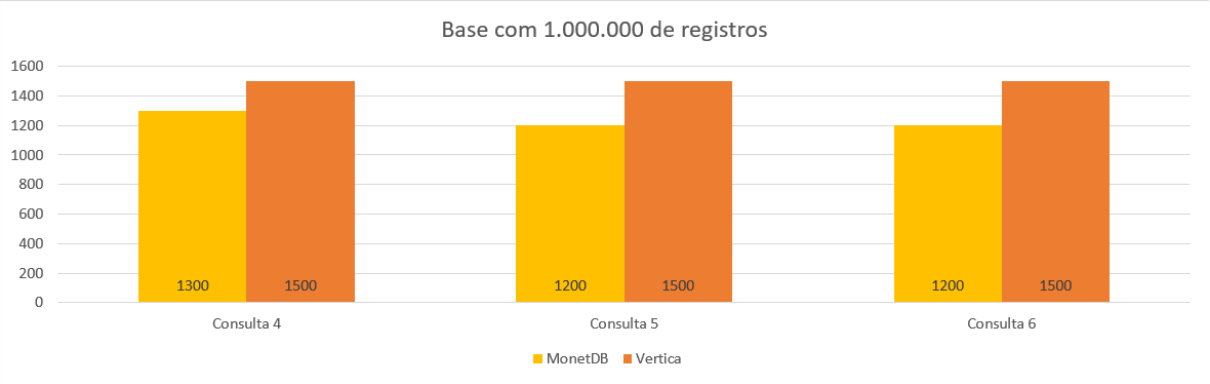
Figura 34 – Consumo da memória MonetDB X Vertica com 50 milhões de registros



Fonte: Autor.

E, na Figura 35 são apresentadas as mesmas métricas (consumos de memória) porém, obtidas durante os testes realizados nos dois bancos de dados, após a base ser preenchida com 1 milhão de registros.

Figura 35 – Consumo da memória MonetDB X Vertica com 1 milhão de registros



Fonte: Autor.

Após coletar o uso da memória do servidor durante a execução das três consultas, nas três sub-etapas de avaliação, notou-se que o Vertica acaba alocando mais memória do que o MonetDB e, a mesma não acaba sendo realocada conforme a consulta executada. No Quadro 17 são apresentadas as médias de consumo da memória obtidas em cada base e, no modo geral.

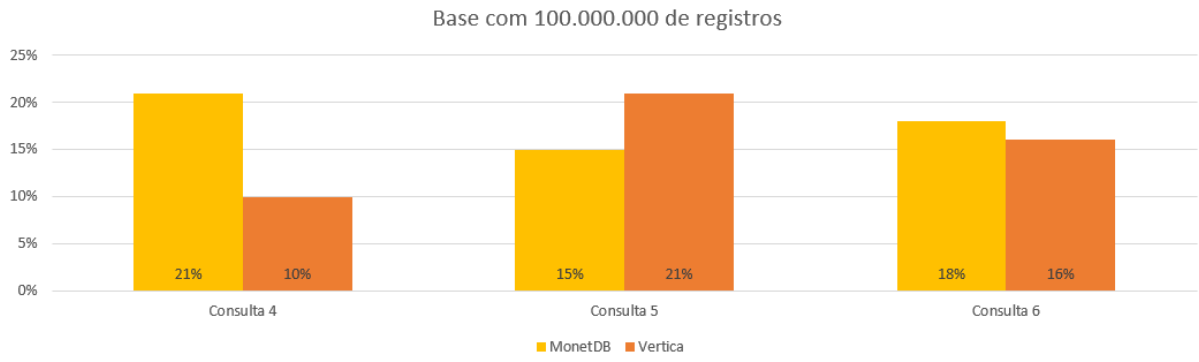
Quadro 17 – Médias de consumo de memória

	MonetDB 11.27.5	**Vertica 8.1.1-0**
100 milhões de registros	1.833,33 MB	2100 MB
50 milhões de registros	1.500 MB	1800 MB
1 milhão de registros	1.233,33 MB	1500 MB
Média geral	**1.522,22 MB**	**1.800 MB**

Fonte: Autor.

Após a apresentação dos resultados coletados com base no uso da memória do servidor, iniciou-se a coleta das métricas que representam o consumo do processador. Na Figura 36 são apresentados os percentuais de uso da CPU, obtidos nos testes com a base de 100 milhões de registros.

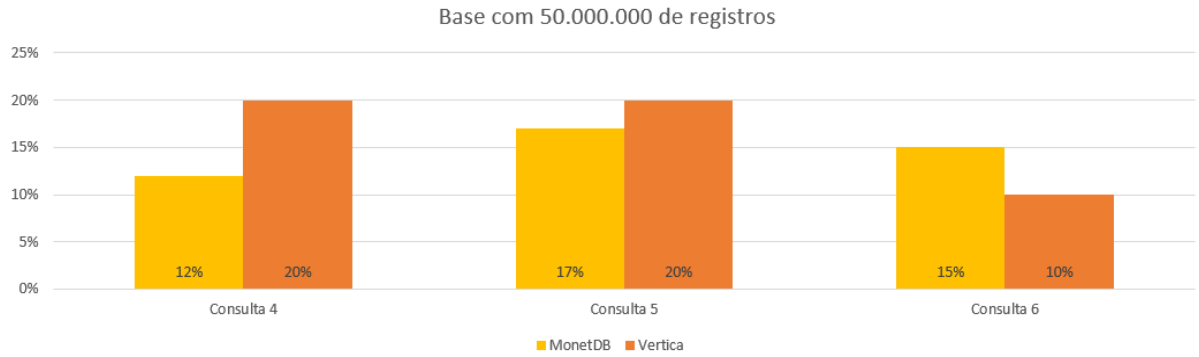
Figura 36 – Uso da CPU MonetDB X Vertica com 100 milhões de registros



Fonte: Autor.

Na Figura 37 são apresentados os percentuais de uso da CPU, obtidos durante os testes realizados nos dois bancos de dados, após a base ser preenchida com 50 milhões de registros de vendas.

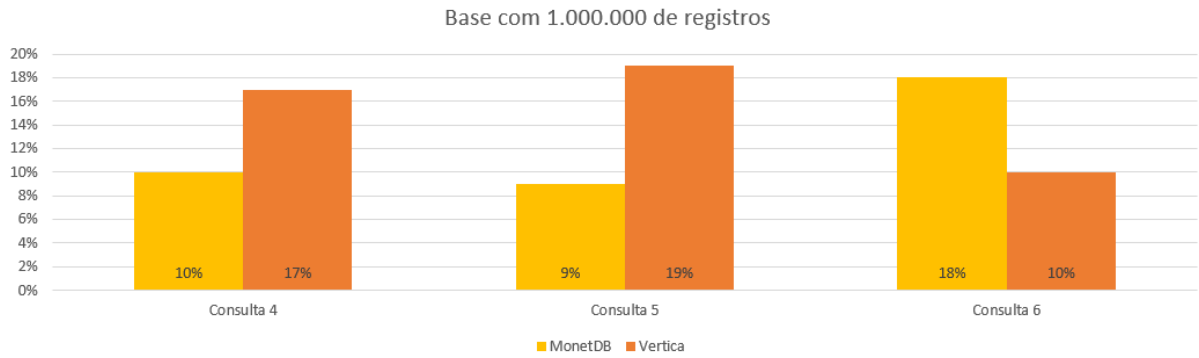
Figura 37 – Uso da CPU MonetDB X Vertica com 50 milhões de registros



Fonte: Autor.

E, na Figura 38 são apresentadas as mesmas métricas porém, obtidas durante os testes realizados nos dois bancos de dados, após as duas bases serem preenchidas com 1 milhão de registros.

Figura 38 – Uso da CPU MonetDB X Vertica com 1 milhão de registros



Fonte: Autor.

Durante a coleta das métricas de uso do processador, em nenhum momento, o processador ocupou mais do que 30% de sua capacidade, visto que os dois bancos de dados utilizados nesse comparativo permaneceram com suas configurações padrão. A consulta 6 acaba utilizando mais do processador porque, em sua composição, estão sendo utilizadas as operações de soma, contagem e ordenação.

Já na consulta 4, os resultados encontrados na maioria dos casos é justamente o inverso dos resultados encontrados para a consulta 6, isso porque a consulta 4 não está utilizando as operações de soma e ordenação. No Quadro 18, são apresentados os percentuais médio de processamento obtidos pelos dois bancos de dados utilizados nesse comparativo.

Quadro 18 – Percentuais médio de processamento

	MonetDB 11.27.5	**Vertica 8.1.1-0**
100 milhões de registros	18%	15,67%
50 milhões de registros	14,67%	16,67%
1 milhão de registros	12,33%	15,33%
Média geral	**15%**	**15,89%**

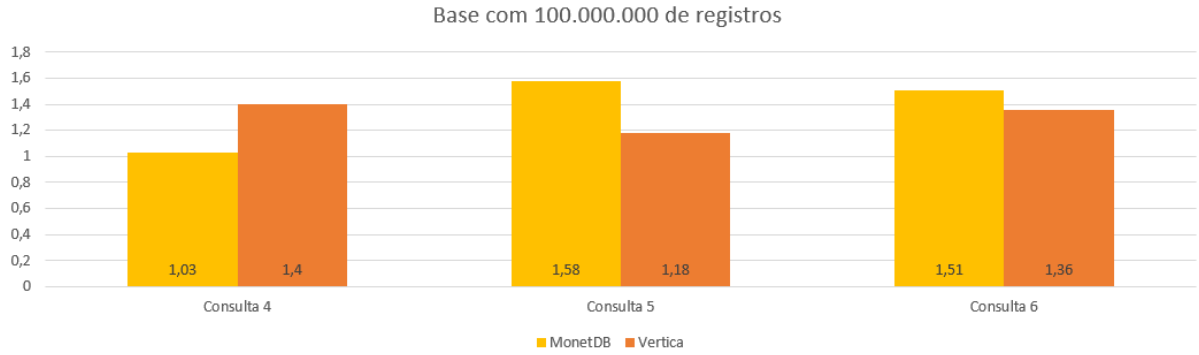
Fonte: Autor.

Nota-se que o Vertica manteve um nível estável de processamento (em torno de 15%), mesmo após aumentar a quantidade de registros inseridos em sua base. Já o MonetDB aumentava o processamento conforme o volume de dados envolvido, variando entre 12% e 18%.

Após a obtenção dos percentuais médio de processamento, obtidos em cada sub-etapa, notou-se que o Vertica obteve um melhor aproveitamento do processador do servidor, que, em média foi utilizado 15,89%, diferença de 0,89%, se comparado com a média obtida pelo MonetDB.

Por fim, foi realizada a coleta do indicador médio de carga do sistema, também conhecido pelo termo *load average*. Esse indicador avalia o índice de ocupação do servidor. Sendo assim, como o servidor utilizado durante o comparativo possui 2 núcleos, se o *load average* passar de 2, significa que ele está sobrecarregado, com uma fila de processos aguardando sua própria execução. Na Figura 39 são apresentados os indicadores médio de carga do sistema, obtidos nos testes com a base de 100 milhões de registros.

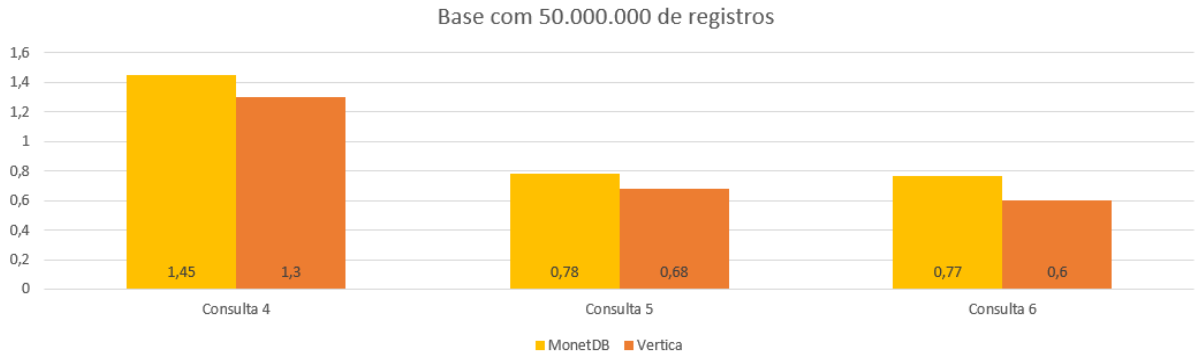
Figura 39 – Load Average MonetDB X Vertica com 100 milhões de registros



Fonte: Autor.

Na Figura 40 são apresentados os indicadores médio de carga do sistema, obtidos durante os testes realizados nos dois bancos de dados, após suas bases serem preenchidas com 50 milhões de registros.

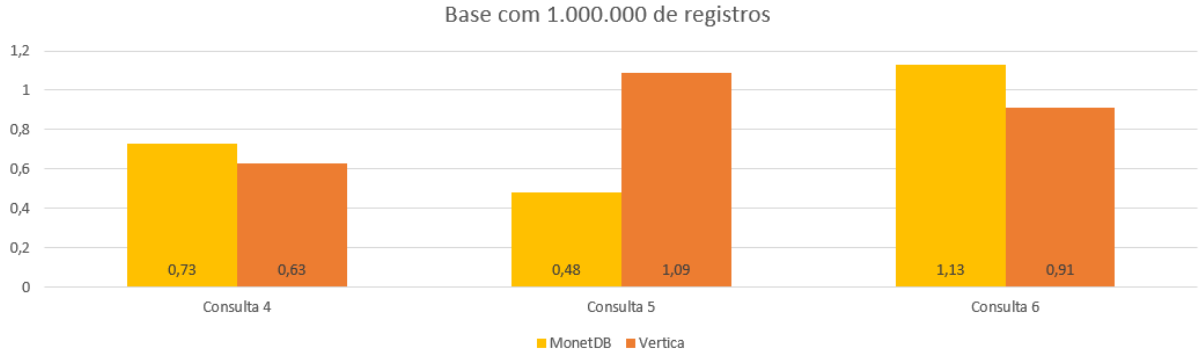
Figura 40 – Load Average MonetDB X Vertica com 50 milhões de registros



Fonte: Autor.

E, na Figura 41 são apresentados os indicadores médio de carga do sistema, obtidos durante os testes realizados nos dois bancos dedados, após as bases serem preenchidas com 1 milhão de registros.

Figura 41 – Load Average MonetDB X Vertica com 1 milhão de registros



Fonte: Autor.

Durante a coleta dos indicadores médio de carga do sistema, em nenhum momento, o sistema ficou sobrecarregado, operando normalmente, lembrando que os dois bancos de dados utilizados nesse comparativo permaneceram com suas configurações padrão. No Quadro 19, são apresentadas as médias dos indicadores de *load average*, obtidas pelos dois bancos de dados.

Quadro 19 – Médias de load average

	MonetDB 11.27.5	**Vertica 8.1.1-0**
100 milhões de registros	1,37	1,31
50 milhões de registros	1	0,86
1 milhão de registros	0,78	0,88
Média geral	**1,05**	**1,02**

Fonte: Autor.

Após a obtenção das médias de *load average*, obtidas em cada sub-etapa, notou-se que o Vertica obteve as menores médias de *load average*, com exceção da última sub-etapa, onde obteve uma média de 12,82% superior à encontrada para o MonetDB. No geral, a média de *load average* foi de 2,86% inferior à média geral do MonetDB.

6.2.4. Visualização

A etapa de avaliação da visualização consiste em validar a conexão entre a ferramenta de BI e, os dois bancos de dados utilizados no comparativo. Além disso, a ferramenta de BI registra em um arquivo texto, todas ou, boa parte das operações feitas na ferramenta, dentre elas, a abertura de um dashboard, que é um conjunto de análises.

Essa etapa, assim como a etapa 6.2.3, também foi dividida em três sub-etapas, onde para cada uma delas, será requisitado a abertura de quatro dashboards, onde dois se comunicaram com o MonetDB e, os outros dois, com o Vertica. Cada dashboard possui em sua composição, três objetos e, cada um, utiliza um tipo de gráfico.

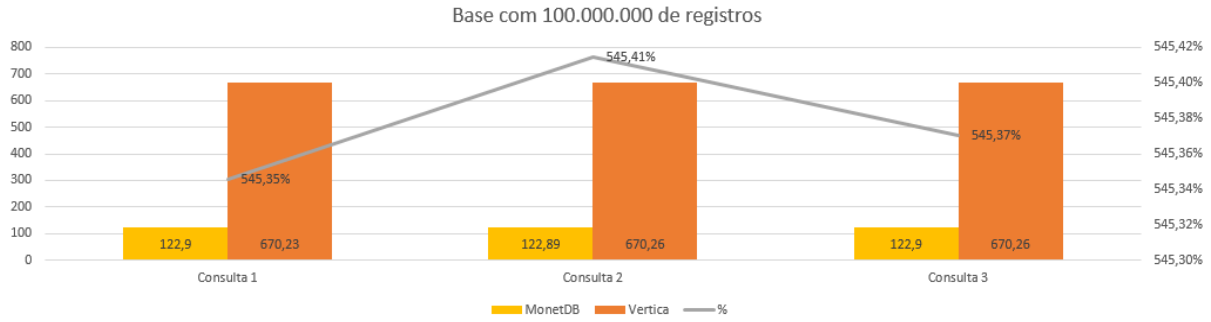
Quanto a conexão com os dois bancos de dados, apenas para o Vertica foi necessário importar o driver de conexão JDBC. Para o MonetDB, o driver já vinha incluso no pacote com a ferramenta de BI.

Na etapa de criação do dashboard, foi adicionado um gráfico de barras, um gráfico de linhas e, um gráfico de pizza. Esses três gráficos, foram configurados para acessar um dos bancos do comparativo, utilizando as consultas apresentadas no Quadro 15. Após a criação do primeiro dashboard, o mesmo foi replicado, substituindo as consultas do Quadro 15 pelas

consultas do Quadro 16. E, mais adiante, os dois dashboards foram replicados, alterando apenas os dados de conexão do MonetDB pelos dados do Vertica.

Para ter certeza de que nenhum processo estava influenciando ou impactando nos resultados da pesquisa, o servidor foi reiniciado e, após a reinicialização, apenas a ferramenta de BI e um dos bancos foram iniciados. Realizada a coleta dos tempos de execução, o servidor foi reiniciado novamente para coletar as métricas com base no outro banco de dados utilizado no comparativo. Na Figura 42, são apresentados os tempos de geração dos objetos do dashboard, utilizando a base com 100 milhões de registros e, as consultas do Quadro 15.

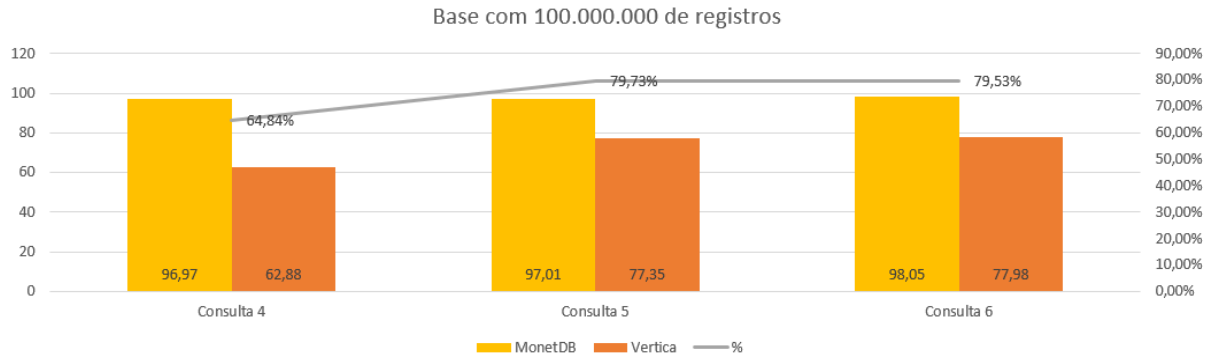
Figura 42 – Geração do Dashboard Analítico – 100 milhões de registros



Fonte: Autor.

Na Figura 43, são apresentados os tempos de geração dos objetos do dashboard, após as duas bases serem preenchidas com 100 milhões de registros, porém, as consultas do Quadro 16.

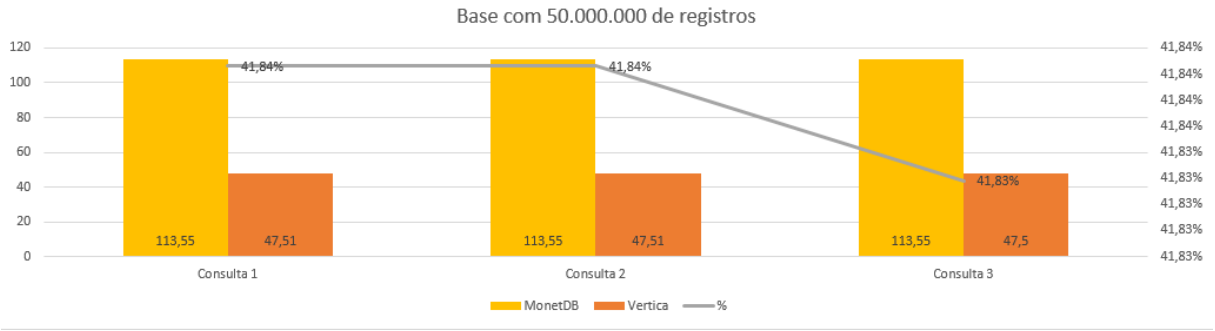
Figura 43 – Geração do Dashboard – 100 milhões de registros



Fonte: Autor.

Na Figura 44 são apresentados os tempos de geração dos objetos do dashboard, após as duas bases de dados serem preenchidas com 50 milhões de registros e, as consultas do Quadro 15.

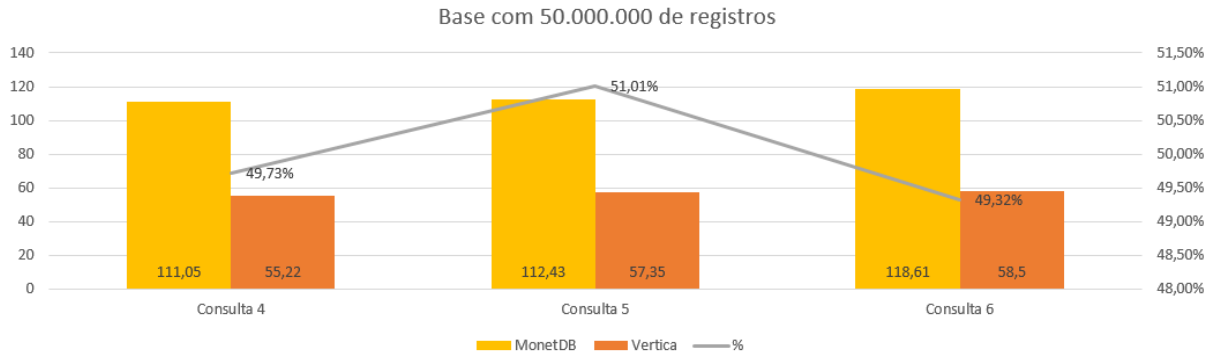
Figura 44 – Geração do Dashboard Analítico – 50 milhões de registros



Fonte: Autor.

Na Figura 45, são apresentados os tempos de geração dos objetos do dashboard, após as duas bases de dados serem preenchidas com 50 milhões de registros, porém, as consultas do Quadro 16.

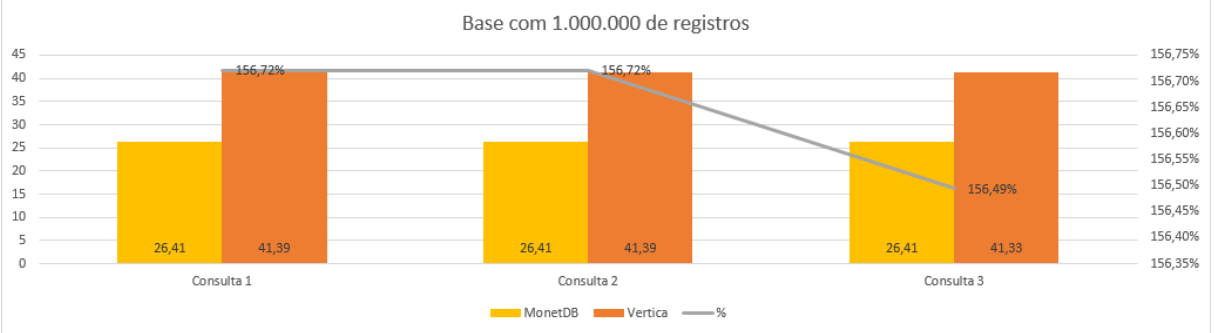
Figura 45 – Geração do Dashboard – 50 milhões de registros



Fonte: Autor.

Na Figura 46 são apresentados os tempos de geração dos objetos do dashboard, obtidos após o preenchimento das duas bases dados com 1 milhão de registros e, as consultas do Quadro 15.

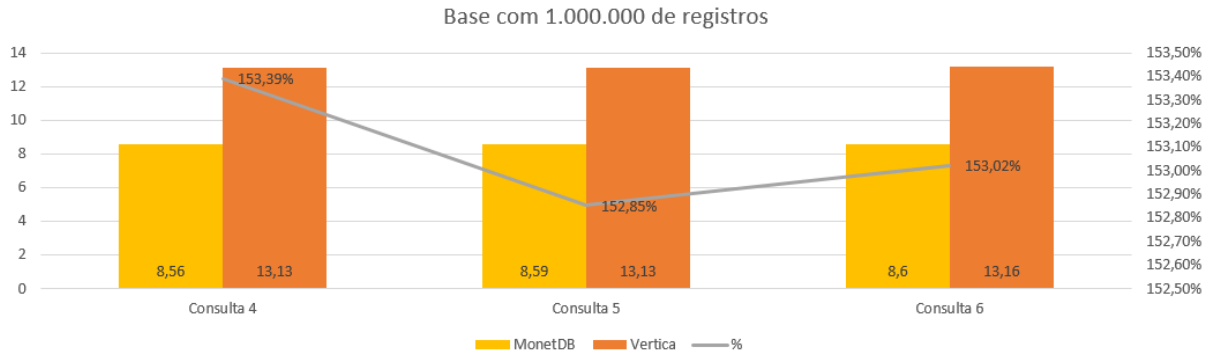
Figura 46 – Geração do Dashboard Analítico – 1 milhão de registros



Fonte: Autor.

E, na Figura 47 são apresentados os tempos de geração dos objetos do dashboard, após as duas bases de DW serem preenchidas com 1 milhão de registros, porém, as consultas do Quadro 16.

Figura 47 – Geração do Dashboard – 1 milhão de registros



Fonte: Autor.

Após a apresentação dos resultados obtidos com base nos quatro dashboards gerados a partir das três quantidades distintas de registros, notou-se que o Pentaho Business Analytics obteve bons resultados com os dois bancos.

Na Figura 42, ao gerar o dashboard analítico a partir de uma base com 100 milhões de registros, o MonetDB apresentou os melhores resultados nas 3 consultas, obtendo um percentual de redução superior a 400%. Na Figura 43, ao gerar o dashboard a partir da mesma base (com 100 milhões de registros), o Vertica acabou apresentando os melhores resultados nas 3 consultas, onde o maior percentual de redução foi de 35,16%.

Na Figura 44, após gerar o dashboard analítico a partir de uma base com 50 milhões de registros, novamente, o Vertica apresentou os melhores resultados nas 3 consultas, obtendo um percentual de redução acima de 50%, para cada consulta. Na Figura 45, após gerar o dashboard a partir da mesma base (com 50 milhões de registros), o Vertica apresentou os melhores resultados nas 3 consultas, onde o menor percentual de redução foi de 48,99%.

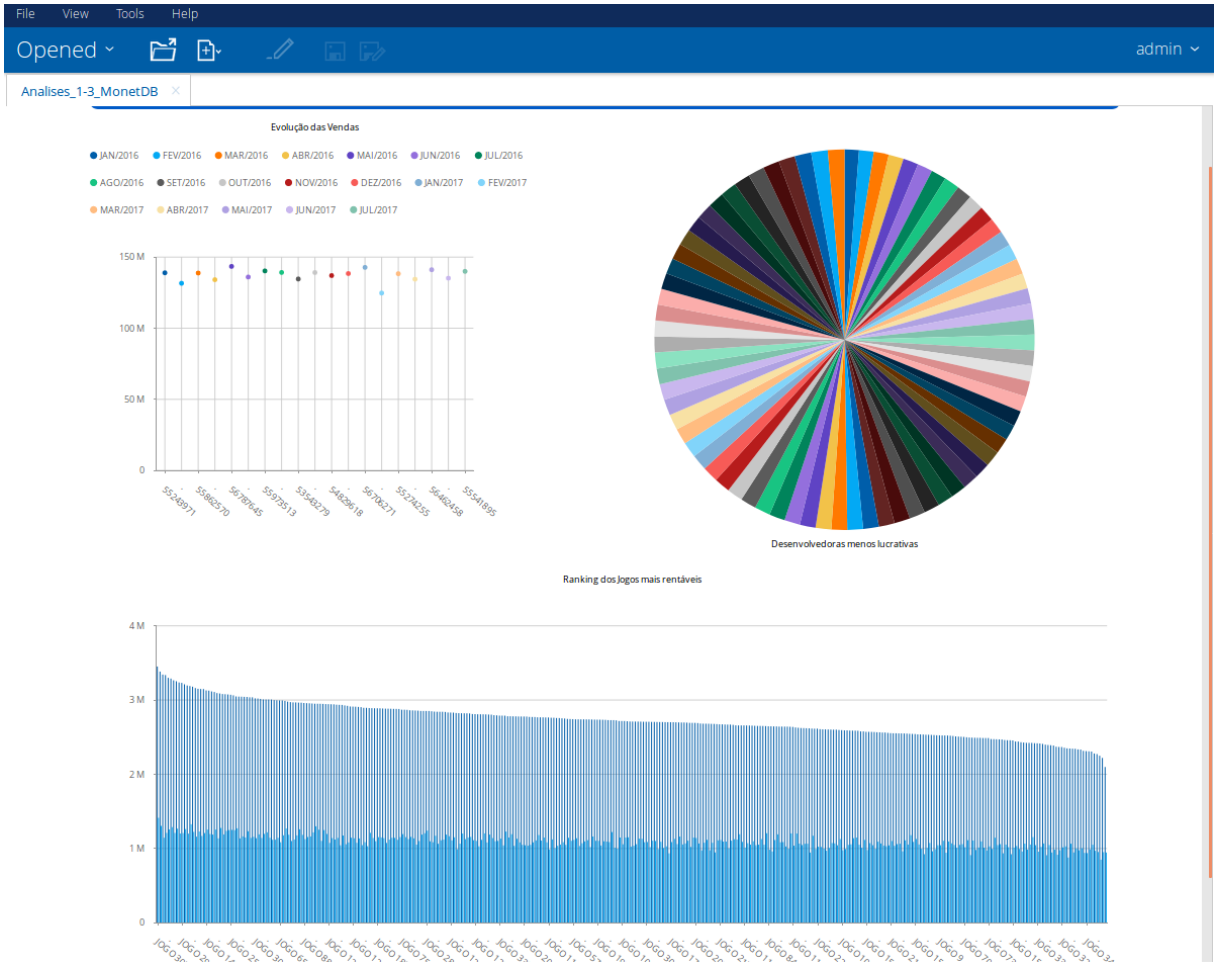
Tanto na Figura 46, quanto na Figura 47, quem demonstrou os melhores resultados foi o MonetDB, apresentando uma redução acima de 50%, para cada consulta. Como o foco principal desse critério de avaliação era validar a conexão entre a ferramenta de BI e os dois

bancos e, além disso, comparar os tempos de geração dos dashboards, a etapa de criação dos objetos não teve muito foco.

Como a escolha do banco vai depender muito do tipo de contexto que a empresa se encontra, ou seja, qual problema ela quer resolver, foi criado o Quadro 20, com objetivo de apresentar, com base em uma escala de 1 a 5, um resumo dos resultados obtidos nesse comparativo.

Na Figura 48 é apresentado o dashboard gerado com base nas consultas do Quadro 15, lembrando que esse dashboard foi replicado e, suas consultas foram substituídas pelas consultas do Quadro 16.

Figura 48 – Dashboard gerado durante o comparativo



Fonte: Autor.

6.3. Resultado

Com a análise nos aspectos de modelagem, carga de dados, performance e visualização, foi possível verificar algumas diferenças entre os dois bancos de dados, desde a etapa de instalação dos dois bancos, até a etapa de geração e visualização dos dados a partir da ferramenta de *Business Intelligence* (BI). Com base nos experimentos realizados, a escala apresentada na Tabela 1 será utilizada para estabelecer um critério de comparação entre os dois bancos de dados nos mais diversos critérios.

Tabela 1 – Lista de Escalas utilizadas

Nível	**Descrição**
1	Péssimo
2	Ruim
3	Indiferente
4	Bom
5	Excelente

Fonte: Autor.

No Quadro 20, com base na escala apresentada na Tabela 1, é apresentado um resumo dessa avaliação e, mais abaixo, foi realizada a explicação do Quadro 20

Quadro 20 – Resumo dos resultados obtidos

	MonetDB 11.27.5	**Vertica 8.1.1-0**
Instalação	5 (★★★★★)	2 (★★)
Modelagem dos dados	2 (★★)	5 (★★★★★)
Importação dos dados	2 (★★)	5 (★★★★★)
Tamanho das Bases	2 (★★)	5 (★★★★★)
Testes de Performance	3 (★★★)	4 (★★★★)
Consumo de Memória	5 (★★★★★)	4 (★★★★)
Consumo de CPU	4 (★★★★)	3 (★★★)
Indicador médio de Carga do Sistema	3 (★★★)	5 (★★★★★)
Visualização	3 (★★★)	3 (★★★)

Fonte: Autor.

Na etapa de instalação dos dois bancos de dados, vimos que o MonetDB possuía uma instalação mais simplificada, se comparada com a do Vertica, o que lhe garantiu 5 pontos, contra 2 do Vertica. Entretanto, na etapa de modelagem dos dados, o Vertica começou a apresentar seus benefícios após uma instalação cheia de problemas a serem corrigidos, o que lhe assegurou 5 pontos, contra 2 do MonetDB.

Na etapa de importação dos dados, tanto via comandos DML de inserção, quanto via importação dos arquivos, o MonetDB não conseguiu ser mais eficiente do que o Vertica, deixando de apresentar os melhores resultados e lhe garantindo apenas 2 pontos, contra 5 do Vertica. Ao verificar o tamanho das duas bases, após a criação e importação dos dados, Vertica novamente obteve os melhores resultados, ganhando mais 5 pontos a seu favor, enquanto o MonetDB ganhou apenas 2 pontos.

Ao iniciar a avaliação com base na performance dos dois bancos, em alguns momentos, o MonetDB apresentou os melhores resultados, mas, na maioria das situações, o Vertica obteve os melhores resultados. Das dezoito análises executadas, treze demonstram o Vertica como a melhor opção de banco, contra quatro do MonetDB e, um empate. Sendo assim, o MonetDB conseguiu adquirir 3 pontos a seu favor, enquanto que o Vertica ficou na frente com a diferença de 1 ponto.

Lembrando que, para avaliar o consumo de memória e do processador do servidor, o número de consultas foi reduzido para nove, dando foco às análises apresentadas no Quadro 16.

Ao avaliar o consumo de memória do servidor, o MonetDB foi o que menos utilizou a memória para trazer as mesmas informações que o Vertica, o que lhe garantiu 5 pontos, enquanto que o Vertica teve 1 ponto a menos. E, referente ao consumo de CPU, novamente, o MonetDB foi o banco que, na maioria dos casos, obteve os melhores resultados, obtendo assim, 4 pontos a seu favor, enquanto que o Vertica, novamente, teve uma diferença de 1 ponto. Das nove análises executadas, cinco beneficiam o MonetDB, contra quatro do Vertica.

Ao avaliar o indicador médio de carga do sistema, o Vertica obteve os melhores resultados, com um total de sete das nove análises executadas a seu favor. O MonetDB acabou obtendo os melhores resultados apenas em duas análises. Dessa forma, o Vertica conquistou mais 5 pontos, enquanto que o MonetDB conquistou 3 pontos.

E, por fim, com base no critério de visualização, houve um empate entre os dois bancos de dados, pois, das dezoito análises executadas, metade favoreceram o Vertica e, a outra metade favoreceu o MonetDB, garantindo 3 pontos para ambos bancos de dados.

Por fim, visto que diferentes critérios possuem diferentes pesos de avaliação, não faz sentido realizar somas ou médias. Enquanto que para um contexto, o tamanho de bases é mais relevante, para outro o uso de CPU pode ser muitas vezes mais importante.

7. CONSIDERAÇÕES FINAIS

No primeiro momento foi definido como objetivo principal dessa pesquisa, a execução de um estudo comparativo entre duas bases NoSQL Orientadas a Colunas e, projetadas para atender a demanda de uma ferramenta de BI. Este objetivo foi cumprido, através da pesquisa sobre diferentes modelos de bases NoSQL, modelagem de uma estrutura de *Data Warehouse* a ser utilizada na pesquisa, escolha de diferentes bases de dados NoSQL, criação das duas bases para posterior avaliação, escolha de uma ferramenta de *Business Intelligence* para uma posterior avaliação e, por fim, analisar os resultados obtidos nessa pesquisa.

A partir da revisão bibliográfica, foi permitido conhecer e detalhar os principais conceitos utilizados nessa pesquisa, desde o detalhamento dos níveis de uma empresa e dos tipos de sistemas de informação existentes até a compreensão dos fundamentos de cunho importantes para o pleno funcionamento de um *Data Warehouse*.

Assim, foi possível definir o tipo de base NoSQL utilizada nessa pesquisa, o modelo Orientado a Coluna, que, dentre as várias opções disponíveis, estão o MonetDB e o Vertica, modelos escolhidos para essa pesquisa. Após a escolha dos dois bancos de dados a serem utilizados e comparados, foi implementado o ambiente de testes dessa pesquisa, onde foram avaliados os critérios de modelagem dos dados, carga dos dados, performance e, visualização dos dados.

Após a realização do comparativo entre os dois bancos de dados, foi possível observar algumas diferenças e semelhanças entre eles, bem como as situações em que cada um obteve ou não vantagens. Como semelhanças, pode-se observar o fato dos dois bancos suportarem a linguagem SQL, além de utilizarem, da mesma forma, os conceitos de tabelas e atributos

Dos critérios avaliados, o que mais se destacou foi o critério de carga de dados, pois os resultados obtidos pelo Vertica, durante a importação do arquivo de vendas, composto de 100 milhões de registros, utilizou apenas de 12,21% do tempo utilizado pelo MonetDB.

Dentre os dois bancos comparados, o que se destacou na maioria das situações, em que o desempenho foi avaliado, foi o Vertica, conforme foi citado no capítulo 6. O MonetDB também ganhou destaque, principalmente nos cenários que envolviam a geração dos dois dashboards utilizando uma base com 1 milhão registros. Além disso, obteve os menores consumos de memória e processador. Ele foi o banco que menos utilizou a memória para trazer os mesmos dados apresentados pelo Vertica.

Dentre as três ferramentas de BI analisadas, apenas o Pentaho Business Analytics foi capaz de se comunicar com os dois bancos, não apresentando problemas de conexão com as duas bases testadas.

O estudo cumpriu todos os objetivos definidos nessa pesquisa, apresentando resultados interessantes entre as duas bases de dados. Os resultados mostraram que, mesmo após uma instalação complicada e cheia de problemas, um banco de dados pode apresentar os melhores resultados ao compara-lo com outro banco, com base em um grupo de critérios de avaliação, como foi o caso do Vertica, que obteve boa parte dos resultados a seu favor.

Para trabalhos futuros, novos comparativos poderiam ser realizados, comparando outros bancos de dados ou até mesmo diferentes modelos de bancos de dados, como por exemplo: um comparativo entre bancos relacionais, não-relacionais orientados a colunas e, não relacionais orientados a documentos, podendo novamente analisar os critérios de avaliação dessa pesquisa.

REFERÊNCIAS

40 OPEN Source and Free Business Intelligence Solutions. Predictive Analytics Today, 2017. Disponível em: <http://www.predictiveanalyticstoday.com/open-source-free-business-intelligence-solutions/>. Acesso em: 9 mai. 2017.

AIRBNB Projects. Airbnb, 2017. Disponível em: <http://airbnb.io/projects/superset/>. Acesso em: 24 mai. 2017.

ALECRIM, Emerson. O que é Big Data? InfoWester, 2013. Disponível em: <https://www.infowester.com/big-data.php>. Acesso em: 4 mai. 2017.

AUDY, Jorge L. N.; DE ANDRADE, Gilberto K.; CIDRAL, Alexandre. Fundamentos de Sistemas de Informação. Porto Alegre: Bookman, 2005.

BARROSO, Isaias. Banco de dados orientado a colunas. Meu canto para assuntos sobre Tecnologia da Informação… ou não, 2012. Disponível em: <https://isaiasbarroso.wordpress.com/2012/06/20/banco-de-dados-orientado-a-colunas/>. Acesso em: 27 mai. 2017.

BIG Data O que é e por que é importante? SAS, 2017. Disponível em: <https://www.sas.com/pt_br/insights/big-data/what-is-big-data.html>. Acesso em: 4 mai. 2017.

CHEMIN, Beatris F. Manual da Univates para trabalhos acadêmicos: planejamento, elaboração e apresentação. 3. ed. Lajeado: Univates, 2015. E-book. Disponível em: <http://www.univates.br/biblioteca/>.

COLAÇO JÚNIOR, Methanias. Projetando Sistemas de Apoio à Decisão Baseados em Data Warehouse. Rio de Janeiro: Axcel Books, 2004.

COMO funciona o sistema de classificação indicativa. O Globo, 2017. Disponível em: <https://oglobo.globo.com/cultura/filmes/como-funciona-sistema-de-classificacao-indicativa-19988509>. Acesso em: 14 nov. 2017.

CONCEITO de Business Intelligence. AUSLAND, 2017. Disponível em: <http://ausland.com.br/conceito-de-business-intelligence/>. Acesso em: 9 mai. 2017.

CONFLUENCE. Confluence, 2017. Disponível em: <https://cwiki.apache.org/confluence/display/IMPALA/Building+Impala>. Acesso em: 16 out. 2017.

DB-ENGINES Ranking. DB-Engines, 2017. Disponível em: <https://db-engines.com/en/ranking>. Acesso em: 3 out. 2017.

ELMASRI, Ramez; NAVATHE, Shamkant B. Sistemas de banco de dados. 6. ed. São Paulo: Pearson Addison Wesley, 2011.

FLORENZANO, Cláudio. Tipos de Sistemas de Informação nas organizações. CBSI, 2015. Disponível em: <http://www.cbsi.net.br/2015/04/tipos-de-sistemas-de-informacao-nas-organizacoes.html>. Acesso em: 6 out. 2017.

GITHUB. Github, 2017. Disponível em: <https://github.com/apache/incubator-superset>. Acesso em: 16 out. 2017.

GRANDES desconhecidos da Computação – Edgar Frank Codd. Password, 2010. Disponível em: <https://blogpassword.wordpress.com/2010/10/03/grandes-desconhecidos-da-computacao-%E2%80%93-edgar-frank-codd/>. Acesso em: 8 mai. 2017.

GREENPLUM. Greenplum, 2017. Disponível em: <http://greenplum.org/>. Acesso em: 16 out. 2017.

GROFFE, Renato. Bancos de dados NoSQL: uma visão geral. iMasters, 2016. Disponível em: <https://imasters.com.br/banco-de-dados/bancos-de-dados-nosql-uma-visao-geral/?trace=1519021197&source=single>. Acesso em: 13 mai. 2017.

JOÃO, Belmiro L. Sistemas de Informação. São Paulo: Pearson Education do Brasil, 2012. E-book. Disponível em: <http://www.univates.br/biblioteca/>.

KIMBALL, Ralph; ROSS, Margy. The data warehouse toolkit: the complete guide to dimensional modeling. 2. ed. São Paulo: Wiley Computer, 2002.

KOBER, Marcel. Um Estudo Comparativo entre o Uso de Bases de Dados Relacionais e Não Relacionais para Datawarehouses. Lajeado. Dissertação de bacharel da Centro Universitário – Univates, 2017.

MACÁRIO, Carla G. do N.; Baldo, Stefano M. O Modelo Relacional. Instituto de Computação da Unicamp, Campinas, 2005.

MONETDB. MonetDB, 2017. Disponível em: <https://www.monetdb.org/content/column-store-features>. Acesso em: 16 out. 2017.

ORGANIZAÇÃO, Organograma e Níveis Organizacionais. Adm Inside, 2015. Disponível em: <https://adminsideblogs.wordpress.com/2015/11/08/organizacao-organograma-e-niveis-organizacionais/>. Acesso em: 6 out. 2017.

PENTAHO. Pentaho, 2017. Disponível em: <http://www.pentaho.com>. Acesso em: 16 out. 2017.

PENTAHO vs SpagoBI. IT Central Station, 2017. Disponível em: <https://www.itcentralstation.com/products/comparisons/pentaho_vs_spagobi>. Acesso em: 13 out. 2017.

PERADELLES, Marcella. A diferença entre o planejamento estratégico, tático e operacional. Blog da Qualidade, 2016. Disponível em: <http://www.blogdaqualidade.com.br/a-diferenca-entre-o-planejamento-estrategico-tatico-e-operacional/>. Acesso em: 16 mai. 2017.

PERFORMANCE CO-PILOT. Performance Co-Pilot, 2017. Disponível em: <http://pcp.io/>. Acesso em: 01 jun. 2017.

PIZZANI, Luciana; DA SILVA, Rosemary C.; BELLO, Suzelei F.; HAYASHI, Maria C. P. I. A ARTE DA PESQUISA BIBLIOGRÁFICA NA BUSCA DO CONHECIMENTO. Revista Digital de Biblioteconomia e Ciência da Informação, São Carlos, v. 10, n.1, p. 53-66, jul. 2012.

PLANEJAMENTO Estratégico, Tático e Operacional. Portal Administração, 2014. Disponível em: <http://www.portal-administracao.com/2014/07/planejamento-estrategico-tatico-operacional.html>. Acesso em: 6 out. 2017.

RENNÓ, Rodrigo. Níveis Organizacionais. Eu vou Passar, 2010. Disponível em: <https://www.euvoupassar.com.br/artigos/detalhe?a=administracao-niveis-organizacionais>. Acesso em: 06 out. 2017.

RIBEIRO, Viviane. O que é ETL?. Data Analytics to Data Science, 2011. Disponível em: <https://vivianeribeiro1.wordpress.com/2011/06/28/o-que-e-etl-2/>. Acesso em: 09 out. 2017.

SATO, Priscila M. GraphDB Series: o que é um banco de dados de grafos. iMasters, 2014. Disponível em: <https://imasters.com.br/banco-de-dados/graphdb-series-o-que-e-um-banco-de-dados-de-grafos/?trace=1519021197&source=single>. Acesso em: 27 mai. 2017.

SOFTWARE ADVICE. Software Advice, 2017. Disponível em <http://www.softwareadvice.com/bi/?deployment_id=&market_products_sort_order=&market_products_sortby=great_fit&more=true&price_ranges=&stars=&segment_id=&platforms=&int_site_code=&subsize1_id=#buyers-guide>. Acesso em: 9 mai. 2017.

SPAGOBI. SpagoBI, 2017. Disponível em: <http://www.spagobi.org/product/>. Acesso em: 16 out. 2017.

STACKOVERFLOW. Stackoverflow, 2016. Disponível em: <https://stackoverflow.com/questions/34879615/monetdb-bulk-load-with-auto-increment>. Acesso em: 17 nov. 2017.

TIWARI, Shashank. Professional NoSQL. Indianopolis: John Wiley e Sons, 2011.

VARDANYAN, Mikayel. Escolhendo a ferramenta certa para o banco de dados NoSQL. iMasters, 2011. Disponível em: <https://imasters.com.br/artigo/21781/banco-de-dados/escolhendo--a-ferramenta-certa-para-o-banco-de-dados-nosql/?trace=1519021197&source=single>. Acesso em: 13 mai. 2017.

VERTICA. Vertica, 2017. Disponível em: <https://my.vertica.com/docs/8.1.x/HTML/index.htm#Authoring/ConceptsGuide/Other/Conc

eptsGuide.htm%3FTocPath%3DVertica%2520Concepts%7C_____0>. Acesso em: 16 out. 2017.

WAINER, Jacques. Métodos de pesquisa quantitativa e qualitativa para a Ciência da Computação. Campinas: Unicamp, 2007.