UNIVERSIDADE DE SÃO PAULO

LUCIANO DO NASCIMENTO MIRANDA

ESTUDO COMPARATIVO ENTRE BANCO DE DADOS RELACIONAIS E NOSQL NO PROCESSO DE EXTRAÇÃO DE VALOR DOS DADOS

Monografia apresentada ao Programa de Educação Continuada da Escola Politécnica da Universidade de São Paulo, para obtenção do título de Especialista, pelo Programa de MBA USP Tecnologias Digitais e Inovação Sustentável.

SÃO PAULO

2021

UNIVERSIDADE DE SÃO PAULO

LUCIANO DO NASCIMENTO MIRANDA

ESTUDO COMPARATIVO ENTRE BANCO DE DADOS RELACIONAIS E NOSQL NO PROCESSO DE EXTRAÇÃO DE VALOR DOS DADOS

Monografia apresentada ao Programa de Educação Continuada da Escola Politécnica da Universidade de São Paulo, para obtenção do título de Especialista, pelo Programa de MBA USP Tecnologias Digitais e Inovação Sustentável.

Orientador: Profa. Ma. Rosangela de Fátima Pereira Marquesone.

SÃO PAULO

2021

FICHA CATALOGRÁFICA

MIRANDA, LUCIANO

Estudo comparativo entre banco de dados relacionais e NoSQL no processo de extração de valor dos dados / L. MIRANDA -- São Paulo, 2021.

55 p.

Monografia (MBA em Tecnologias Digitais e Sustentabilidade) – Escola Politécnica da Universidade de São Paulo. PECE – Programa de Educação Continuada em Engenharia.

1. NoSQL e Big Data 2. Sistemas Gerenciadores de Banco de Dados Relacionais 3. Ferramentas de Big Data 4. Grande massa de dados 5. Extração de valor das informações I Universidade de São Paulo. Escola Politécnica. PECE – Programa de Educação Continuada em Engenharia II.t.

AGRADECIMENTOS

Agradeço a todos os professores pela atenção e colaboração em conjunto com a coordenação e todos os funcionários do PECE/USP que foram de grande importância para o andamento deste projeto de curso. Agradeço em especial a coordenadora Tereza que sempre foi muito atenciosa, muito prestativa e sempre nos incentivou, a professora Rosangela pela paciência, perseverança, ajuda, atenção, generosidade onde na qual sem suas análises e explicações seria inviável.

A construção desta escrita e deste trabalho colocando as ideias estudadas em prática e pôr fim a todos os amigos estudando do curso, no qual aprendi muito com todos observando, trabalhando em conjunto, vendo suas apresentações, analisando seus comportamentos, discutindo e aceitando opiniões, agradeço demais a todos pela oportunidade, pelo aprendizado em um momento tão complicado de pandemia todos vocês me ajudaram demais para o crescimento pessoal e profissional.

Encerro este curso muito certo de que foi um investimento ímpar a nível de vida, mais uma vez o meu muito obrigado a todos.

RESUMO

Fatores como a compreensão dos dados, o melhoramento da descentralização destas informações, a forma de ingestão e a integração de ferramentas têm proporcionado benefícios às organizações, tais como eficiência em análise, melhor organização e menor desperdício de tempo. Entretanto, alcançar esses objetivos são desafios a serem sempre alcançados pelas empresas. Nesse contexto, torna-se necessário analisar como as ferramentas de Big Data podem ser utilizadas, visando agregar valor ao negócio a partir da identificação da tecnologia adequada para cada solução. A partir desse cenário, esse trabalho apresenta uma comparação entre a tecnologia Elasticsearch e o banco de dados relacional MySQL, com a finalidade de identificar diferenças, características positivas e negativas e sintaxes de utilização, para assim identificar como tais soluções podem ser utilizadas no processo de extração de valor dos dados.

Palavras-chave: Big Data, Elasticsearch, Banco de Dados Relacional.

ABSTRACT

Factors such as understanding the data, improving the decentralization of this information, the form of ingestion and the integration of tools have provided benefits to organizations, such as efficiency in analysis, better organization, and less waste of time. However, achieving these goals are challenges to be always pursued by companies. In this context, it is necessary to analyze how Big Data tools can be used, aiming to add value to the business by identifying the appropriate technology for each solution. From this scenario, this work presents a comparison between the Elasticsearch technology and the MySQL relational database, to identify differences, positive and negative characteristics, and usage syntax, to identify how such solutions can be used in the process. extraction of value from data.

Keywords: Big Data, Elasticsearch, Relational Database.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O processo de análise de informações, das mais variadas fontes de dados, tipos e formatos do ponto de vista do analisador, não é realizado de forma racional. O sentimento, a intuição e as experiências vividas em análises passadas serão determinantes para o sucesso mesmo que feitos de forma consciente ou inconsciente (NONOHAY, 2012; CHOO, 2006; PEREIRA, 2010).

Segundo Lohr (2012), o crescente volume de dados gerado pelas redes sociais, e-mails, sensores, arquivos de servidores e registros de auditoria são a base para a extração de gigantescas quantidades de informações e a obtenção de conhecimento sobre o que estes dados significam.

Atualmente, diversas empresas ainda utilizam primariamente o modelo de banco de dados relacional, porém, isto tende a mudar com o aumento do fluxo de informações a serem analisadas. Diante deste novo cenário, os bancos de dados NoSQL têm se tornado parte das aquisições, com o objetivo de proporcionar ganhos de eficiência nos negócios.

Com o crescente volume de informações, surgiu o termo Big Data, como um novo conceito de inovação, a justificativa para o estudo de Big Data é a tomada de decisão orientada a dados. Segundo citado pela consultoria Internacional Data Corporation (IDC) (IDC, 2021), Big Data é visto como a nova geração de tecnologia e arquitetura que possui como finalidade extrair economicamente um alto valor e uma grande variedade de dados, em alta velocidade de captura e análise fazendo com que os gastos mundiais cheguem a US $ 215,7 bilhões em 2021.

Diante deste cenário, torna-se necessário e relevante um estudo comparativo entre as tecnologias de banco de dados relacionais e NoSQL, identificando como o *Big Data* pode ser utilizado no processo de análise das informações.

1.1 Motivação

De acordo com De Souza et. al (2014), o tema banco de dados relacional ainda é abordado pela grande maioria dos profissionais de tecnologia e gestores como um modelo de bom funcionamento em grandes aplicações. Porém, quando entra-se no mérito de aplicações extremamente grandes, na ordem de milhares de bytes em que

a disponibilidade precisa ser confiável e sem restrições, se questiona a manipulação das simples operações que os banco de dados realizam diariamente, tais como deleção, ordenação, inserção, atualização, podendo sofrer complicações se trabalhados da forma tradicional com se trabalha com banco de dados relacionais tradicionais.

O desafio de Big Data tem exigido mudanças no desenvolvimento de tecnologias, na forma de pensar na coleta de dados, processamento, armazenamento e a infraestrutura (DEMCHENKO et. al., 2013). O entendimento dos dados traduzidos em informações de forma rápida independente do volume, modo e fonte de dados em um mercado acirrado são critérios relevantes para o sucesso dos negócios.

1.2 Objetivo

Dado o crescente interesse em novas tecnologias de armazenamento de dados e a popularização do conceito de NoSQL, esse trabalho tem como objetivo realizar uma comparação inicial entre a solução Elastiscsearch e o sistemas de gerenciamento de banco de dados relacional MySQL. Para esse objetivo, será realizado um experimento a partir da instalação e configuração das duas ferramentas em uma máquina local, avaliando o desempenho de ambas as soluções em aspectos como: tempo de carga e de consulta dos dados, além de ser avaliado empiricamente as funcionalidades de cada uma das soluções.

Essas tecnologias e aplicativos foram discutidas nos seguintes aspectos: características de armazenamento, estrutura, processamento e manipulação de dados, justamente com o objetivo de encontrar melhores formas de coletar, armazenar, analisar e processar utilizando padrões para a melhor compreensão dos 3V’s de Big Data (Velocidade, Volume e Variedade).

1.3 Contribuição

O processo de organização de dados capturados das mais variadas fontes são contribuições que rompem o uso do modelo tradicional relacional de armazenamento dos dados. Para a comunidade e o meio acadêmico, estudar novas tecnologias e formas de se trabalhar com grande variedade, velocidade e volume de dados é mudar

o modo de pensar em como armazenar, extrair e gerar valor de um imenso volume de dados.

O estudo contribui para uma melhor compreensão sobre o processo de extração dos dados, de como é possível unir diferentes tecnologias e extrair valor desta junção, bem como determinar qual pode ser a de melhor utilização, dependendo do cenário de atuação.

1.4 Metodologia

O método de pesquisa utilizado nesse trabalho foi realizado de acordo com a seguinte estratégia:

1. Na primeira etapa foi realizado o levantamento bibliográfico, por meio do estudo de artigos, páginas web, livros, fóruns e resumos de documentações de tecnologia.
2. Na segunda etapa, foi realizado o experimento referente ao uso da ferramenta Elasticsearch, comparando com o sistema de gerenciamento de banco de dados relacional MySQL.
3. Na terceira etapa, foi realizada uma discussão sobre as tecnologias estudadas, comparando-as, identificando pontos de vantagem e desvantagem.

1.5 Organização do trabalho

Este trabalho está dividido da seguinte forma:

Capítulo 1 - são apresentados conceitos necessários para a compreensão do uso de Big Data. Também é descrito o motivo pelo qual melhorar a compreensão das informações que cresce de maneira exponencial necessita de atenção dos gestores e profissionais de tecnologia.

Capítulo 2 - é realizada a revisão bibliográfica das tecnologias de Big Data, NoSQL e banco de dados relacionais, explicando suas principais características.

Capítulo 3 - é apresentada as ferramentas utilizadas no estudo comparativo entre MySQL e Elasticsearch, características da modelagem de tabelas, diferenças de conceitos, testes de desempenho e de armazenamento.

Capítulo 4 - conclui o trabalho, resumindo os pontos principais da utilização da tecnologia NoSQL e trabalhos futuros.

2 REVISÃO BIBLIOGRÁFICA

2.1 Banco de Dados Relacionais

O modelo de banco de dados relacional tem representação na matemática, conforme definição apresentada a seguir.

> "O Modelo Relacional (MR) é um modelo de dados representativo (ou de implementação) que foi proposto por Codd, em (1970). O Modelo fundamenta-se em conceitos da matemática - teoria dos conjuntos e lógica de predicado. Os primeiros sistemas comerciais baseados no MR foram disponibilizados por volta de 1980 e desde então ele vem sendo implementado em muitos sistemas, tais como *Access, Oracle, MySql,* entre outros." (ELMASRI; NAVATHE, 2011, p.38 apud COSTA 2011, p.33).

Atualmente o modelo de banco de dados relacional é amplamente utilizado e, mesmo diante da evolução de novas tecnologias, sua adoção ainda permanece, por fornecer mecanismos para recuperar falhas, manter a integridade, prover consultas rápidas, garantir segurança, tendo suas transações baseadas na propriedade ACID, referentes a atomicidade, consistência, isolamento e durabilidade (BRITO, 2010). De acordo com Date (2004, p. 67), o modelo relacional refere-se a três aspectos principais dos dados: a estrutura de dados, a integridade de dados e a manipulação de dados.

O modelo relacional é matematicamente representado por relações. Sua terminologia denomina as tuplas como sendo suas linhas, as colunas são os atributos e a tabela como o relacionamento das linhas e colunas. (ELMASRI; NAVATHE, 2011, p. 39). Já segundo Silberschatz, Korth e Sudarshan (1999, p. 62), “uma relação é definida matematicamente como um subconjunto do produto cartesiano de uma lista de domínios”.

Em relação à integridade dos dados, tem-se: chave candidata, chave primária, chave estrangeira e integridade referencial. Uma relação pode possuir mais de uma chave, sendo cada uma delas chamada de chave candidata, conforme afirma Elmasri e Navathe (2011, p.45).

A integridade referencial, onde integridade é conhecida como a precisão ou correção de dados no banco de dados, é considerada o valor designado na chave primária da entidade. Neste sentido, simplificando o que é a integridade referencial, é

válido afirmar que ela se refere a negação que implica ao banco de dados não poder ter chaves estrangeiras sem valores condizentes (DATE, 2004).

2.2 Big Data

Para o mercado de trabalho, o termo Big Data é considerado um fenômeno por uma perspectiva relacionada à grande massa de dados, entretanto, a base do termo para muitas empresas ainda é divergente (FRANK, 2012).

Manyika et al. (2011) definem Big Data como um agrupamento de dados em que seu volume supera as condições normais de captura, armazenagem e análise. Em complemento, este termo também é relacionado a enormes quantidades de dados, com dados e características diferentes, fornecidos de fontes diversas e heterogêneas, com conceitos de computação distribuída e sem estruturação (MCAFEE; BRYNJOLFSSON, 2012).

A variedade mencionada se refere à diversidade de dados oriundos de diferentes fontes e em formatos de texto, áudio, vídeos sendo eles estruturados, semiestruturados e não estruturados. Por fim, a velocidade se refere à rapidez da criação dos dados e da forma com que se multiplicam na Internet em referência a estrutura de Big Data (C4PPR4 D4T4 5C13NC3, 2017).

Big Data refere-se, portanto, a um conjunto de informações cujo tamanho ultrapassa a capacidade de ferramentas tradicionais de banco de dados para capturar, armazenar, gerenciar e analisar os dados. Assim, segundo McKinsey (2011), pode-se concluir que a definição é subjetiva, por não haver um tamanho exato para se definir o contexto de grande volume de dados.

2.3 NoSQL

A tecnologia NoSQL, segundo Porcelli (2016), se enquadra em soluções para armazenar dados não relacionais. Dessa forma, a utilização de NoSQL pode ser considerada no contexto de tecnologias de Big Data. Entretanto, apesar de muitas associações entre NoSQL e Big Data, não há uma definição que restrinja essa tecnologia somente a grande volume de dados.

A necessidade de gerenciar grandes volumes de dados já é uma realidade atualmente por inúmeras empresas. Soluções NoSQL normalmente são utilizadas com foco em proporcionar maior desempenho e escalabilidade, por possuírem uma série de alternativas de armazenamento dos dados, serem escaláveis e flexíveis, dado que não exigem restritamente um esquema predefinido dos dados.

Segundo Vieira et. al (2012), a quantidade de dados gerada diariamente da Web, redes sociais, sensores, entre muitos outros dispositivos, apresentam-se na métrica de terabytes. Diante deste cenário, a adoção ao movimento NoSQL tem aumentado pelas empresas.

A seguir são apresentadas características em comum de soluções de banco de dados NoSQL (NÄSHOLM, 2012):

- Distribuídos: há o compartilhamento de recursos entre os sistemas dessa categoria;
- Escalabilidade horizontal: permitem aumentar o desempenho do sistema por meio da adição de recursos (novos servidores) ao cluster;
- Grandes volumes: possibilitam o armazenamento de dados em grande volume, na ordem de terabytes e petabytes;
- BASE ao invés de ACID: nesse contexto, aspectos como a consistência e isolamento muitas vezes são descartados, de forma que a tecnologia NoSQL prevalece a abordagem BASE - Basicamente disponível, Estado leve, eventualmente consistente, do que a propriedade ACID.
- Modelos de dados não relacionais: permite que a modelagem dos dados seja feita de diversas outras formas, além do modelo relacional;
- Sem definições de esquema: não requer uma estrutura pré-definida, possibilitando o armazenamento dos dados sem a rigidez de um banco de dados relacional.
- Sem suporte à SQL: Muitos permitem o uso do SQL como linguagem de consulta, porém cada sistema oferece sua própria interface de consulta (JACKSON, 2011).

2.4 Diferenças Entre Banco de Dados Relacionais e NoSQL

De acordo com Sousa Junior (2014), o modelo NoSQL trabalha com a propriedade BASE (Basicamente Disponível, Eventualmente Consistente) e nesta denominação, atribui-se a obrigatoriedade de disponibilidade o tempo todo, mesmo que impacte sua consistência, no modelo NoSQL os dados inseridos já entram indexados, isso torna a consulta mais rápida para os dados armazenados.

Para Aniceto, Xavier (2014, p. 8)

> Uma definição de *NoSQL* é que ele é um conjunto de conceitos que permitem o processamento de dados de forma rápida e eficiente com um foco em performance. É um modelo alternativo pensado para se modelar os dados sem ter que seguir os padrões rígidos criados para o modelo relacional. Como medida para se contornar esse problema, ele foi proposto trazendo consigo planos de eliminar ou reduzir essa estruturação [8]. Para evitar a perda de informações, o NoSQL tem uma arquitetura distribuída tolerante a falhas que se baseia na redundância de dados em vários servidores.

Uma das características de soluções NoSQL é a não obrigatoriedade de um esquema. Nesse contexto, o desenvolvedor passa a ser o responsável por realizar a estruturação dos dados de uma aplicação, conforme sua necessidade.

A partir das características apresentadas, a Tabela 1 ilustra as principais diferenças identificadas entre banco de dados relacionais e NoSQL.

Tabela 1 - Diferenças entre banco de dados relacionais e NoSQL.

Caracteristicas	Bancos Relacionais	NoSQL
Variedade	Único Tipo (Relacional)	Quatro Tipos: Documental, Colunar, chave-valor e grafos
Estrutura	Pré-Definida	Dinâmica e Flexível
Escalabilidade	Comumente Vertical	Normalmente Horizontal
Propósito	Integridade dos Dados	Alto Desempenho e Disponibilidade

Fonte: Próprio Autor.

Em complemento, a Tabela 2 ilustra a estrutura entre NoSQL e Banco de Dados Relacional. Para isso, utilizou-se como parâmetro a comparação do Elasticsearch e do MySQL.

Tabela 2 - Comparação de estrutura do Elastiscsearch x MySQL

Elasticsearch	MySQL
INDICE	ESQUEMA
TIPO	TABELA
DOCUMENTO	LINHA
CAMPO	COLUNA
MAPEAMENTO	ESTRUTURA DA TABELA

Fonte: Próprio Autor.

Outra diferença observada na literatura é que o armazenamento NoSQL oferece uma variedade de formatos de armazenamento. Atualmente estes bancos de dados podem ser baseados em chave-valor, documentos, colunas e grafos. Vieira et al. (2012, p.23) dizem que a organização do modelo de armazenamento NoSQL podem ser descritos da maneira a seguir:

- Chave-Valor: é considerado o modelo de armazenamento mais simples, dentre as categorias de NoSQL. Sua estrutura pode ser compreendida como sendo uma tabela *hash*, no qual o campo valor permite o armazenamento de diferentes tipos de dados, que por sua vez são acessados por um campo chave, conforme ilustrado na Figura 1.

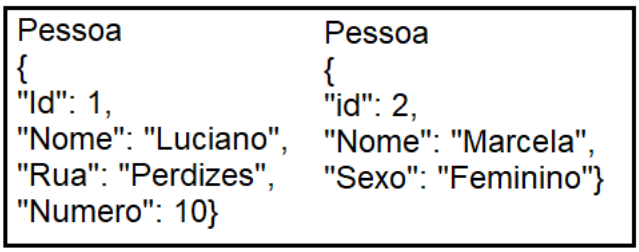
Figura 1. Modelo orientado a chave-valor

NOME	LUCIANO
IDADE	30
SEXO	MASCULINO
FONE	(11)99999-7777

Fonte: Próprio Autor.

- Documentos: Documento é um conjunto de campos e valores com nomes. Seus dados podem ter várias maneiras, sendo, JSON, BSON, XML, YAML e até em formato de texto. Pode-se definir um documento como o nome dado a um conjunto de itens de dados relacionados que constituem uma entidade (MCMURTRY et al., 2013), conforme exemplo da Figura 2.

Figura 2 – Modelo orientado a documentos.

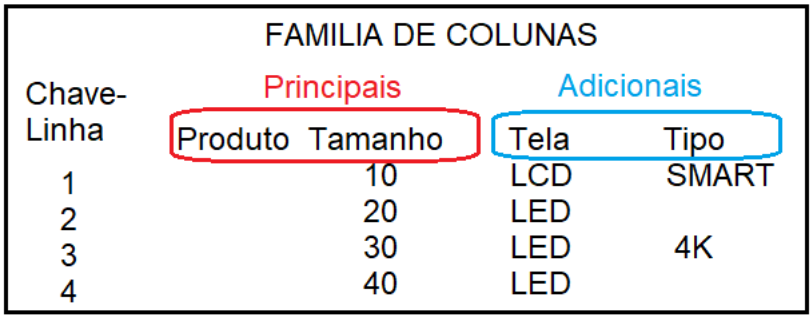


Fonte: Próprio Autor.

Cada documento pode ter os mesmos campos, o que torna a consulta mais fácil, mas como não existe um esquema fixo, cada documento pode ter um número e um tipo de campo diferente.

- Coluna: O banco de dados NoSQL organiza os dados em linhas e colunas. Existe uma semelhança nesta organização com os SGBD's relacionais, no entanto, o modelo de colunas possui estrutura desnormalizada, onde sua exploração de dados permite a divisão de colunas em grupos denominados famílias de colunas (MCMURTRY et al., 2013), conforme ilustrado na Figura 3.

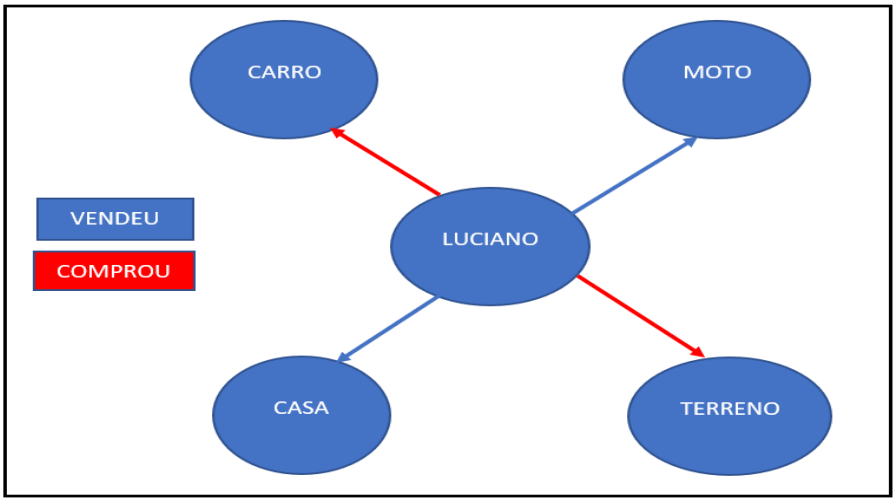
Figura 3 – Modelo orientado a colunas.



Fonte: Próprio Autor.

- Grafos: Nesta composição existem três tipos de elementos, conforme descrito por Mcmurtry et al. (2013): nós, arestas e propriedades. Os nós se referem às instâncias de dados, as arestas são utilizadas para definir os relacionamentos mantidos entre as instâncias, e as propriedades são valores de dados existentes nas instâncias, conforme demonstra a Figura 4.

Figura 4 – Modelo orientado a grafos



Fonte: Próprio Autor.

Visando complementar esse estudo, na próxima seção é apresentado um experimento com foco em comparar características de desempenho e de funcionalidades entre essas duas formas de armazenamento de dados (relacional e NoSQL). Para isso, foi selecionado para o estudo a solução MySQL para armazenamento de dados relacionais e a solução Elasticsearch, para armazenamento em forma de documentos.

3 A SOLUÇÃO ELASTICSEARCH

3.1 Características do Elasticsearch

O Elasticsearch é uma ferramenta com características de Big Data que, ao ser comparada com outros modelos de banco de dados NoSQL, apresenta desempenho superior nas operações de leitura que os demais (ABUBAKAR, ADEYI e AUTA, 2014). O mecanismo de dados distribuído com orientação a documentos possibilita esta tecnologia tratar informações na proporção de terabytes. Além dessa característica de armazenamento em formato de documentos, o Elasticsearch também é caracterizado como uma engine de busca aos dados, com baixo tempo de resposta.

O Elasticsearch oferece 3 funcionalidades básicas em sua composição: o armazenamento, a indexação e a busca de dados. O armazenamento é realizado em formato orientado a documentos, utilizando nativamente o padrão de arquivo JSON, conforme descrito por Baeza-Yates; Ribeiro Neto (2013).

Segundo Pragmateek (2013), ao comparar o JSON com XML, é possível observar que o JSON oferece maior desempenho por ter menor tempo de processamento e por possuir integração a sistemas web como o JavaScript que utiliza JSON de forma nativa.

A arquitetura escalável possibilita a elaboração por um *cluster* de nós realizando a comunicação entre os mesmos e fornecendo uma API REST (*Representation State Transfer,* no inglês, Transferência de Estado Representacional) responsável pelas consultas e armazenamento de documentos (KONONENKO et al., 2014). O Elasticsearch oferece também diversas outras maneiras de se trabalhar com buscas, tais como:

- Suporte a reservatório de threads, agrupamentos, monitoramento;
- Escalabilidade horizontal;
- Alta disponibilidade;
- Schema-less (não exige declaração do esquema a ser utilizado).

Os arquivos utilizados no Elasticsearch são salvos em formato CSV. Seu motor de busca é definido por Roy Thomas (2000) como uma ferramenta de arquitetura distribuída a qual é utilizada para análise de informações por meio da *API REST*, com

o intuito de melhorar a troca de informações com o protocolo de navegação de páginas web HTTP.

3.2 Componentes do Elasticsearch

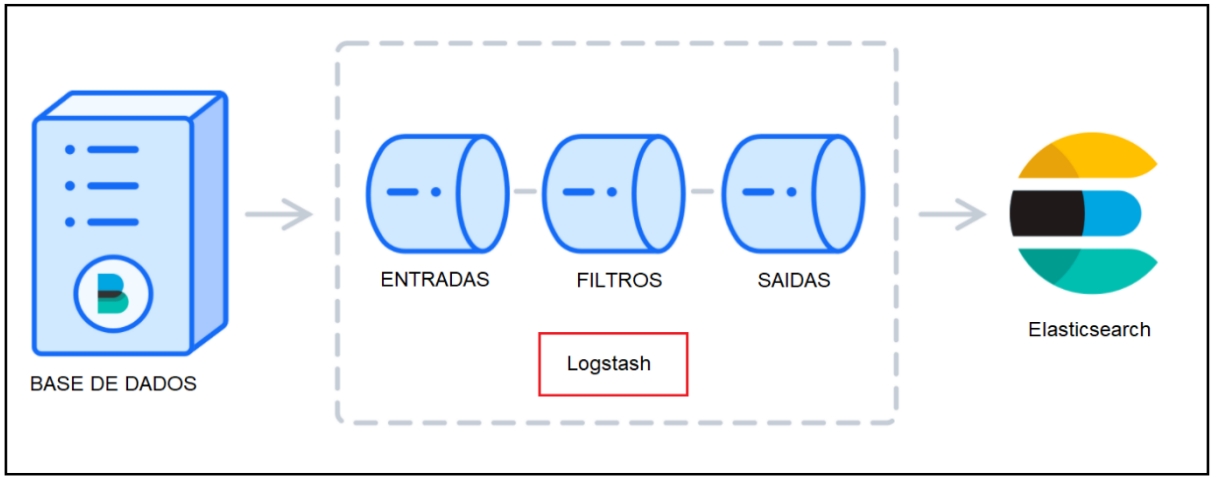
O Elastisearch atua com documentos e sua infraestrutura possui as descrições datalhadas a seguir (KONONENKO et al., 2014):

- Cluster: conjunto de nós;
- Nós: armazem de dados;
- Shard: setor das partições de dados dentro dos nós;
- Réplica Shard: cópia idêntica do shard padrão inserido em outro nó;
- Indice: própria base de dados;
- Tipos: tabelas da base de dados;
- Documentos: arquivos em JSON, informações na base de dados;
- Campos: colunas.

Além do motor de busca, o Elasticsearch possui uma composição referenciada pela sigla ELK, referente às soluções Elasticsearch, Logstash e Kibana, conforme será detalhado a seguir.

Logstash é o responsável pelo processamento de dados do lado do servidor, sua principal função é a ingestão de dados de diversas fontes, tendo utilidade no processo de envio de dados ao Elasticsearch. Possui como funcionalidades as etapas de obter e preparar os dados onde estes são lidos a partir de uma fonte em formato de texto ou extraída por agentes instalados. A Figura 5 apresenta um exemplo de sua utilização por meio da entrada das informações, filtragem e saída para um índice do Elasticsearch.

Figura 5 - Estrutura do Logstash.



Fonte: Logstash (2020).

O Modelo do Logstash representado na Figura 6 é apresentado no formato de arquivo (.conf).

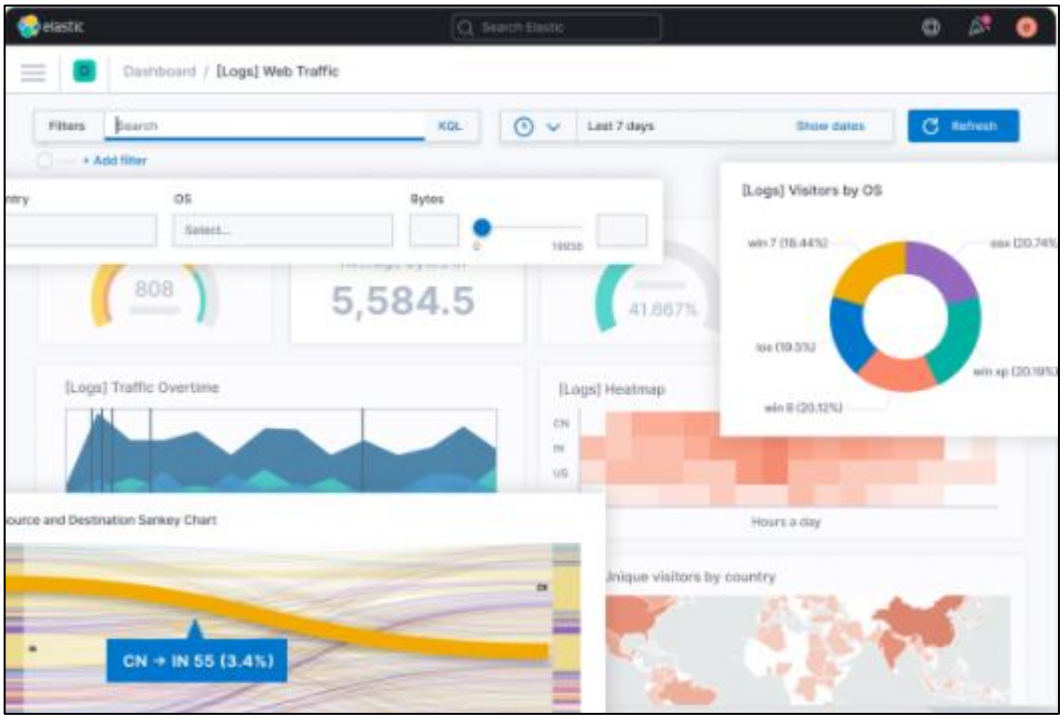
Figura 6 - Logstash *sintaxe* arquivo (.conf).

```
input {
   filet {
          path => "C:/Logstash/logstash-7.13.4-windows-x86_64/logstash-7.13.4/bin/total_load.csv"
          start_position => "beginning"
          }
       }
filter  {
     csv {
         separator => ","
         columns => ["Id","Name","Link","Organ","Superior Organ","Status","Salary"]
        }
         }
output  {
    elasticsearch {
           hosts => "localhost:9200"
           index => "carga_total"
                  }
         }
```

Fonte: Próprio autor.

Kibana é responsável pela construção de painéis de visualização (KIBANA, 2020). É a interface gráfica responsável pela criação de painéis, possibilitando a visualização dos dados que estão armazenados e que estão sendo coletados pelo Logstash e armazenados no ElasticSearch. É uma ferramenta de gerenciamento de informação com diversas possibilidades de tabelas, mapas, histogramas e gráficos. A Figura 7 apresenta um exemplo de sua interface.

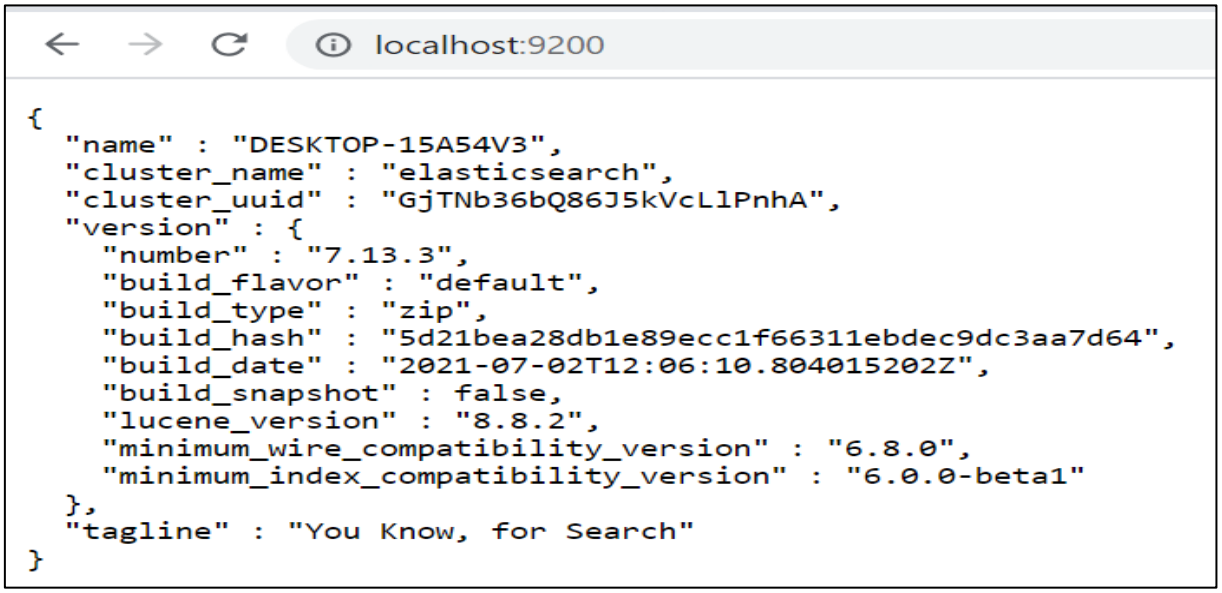
Figura 7 - Página de início do Kibana



Fonte: https://www.elastic.co/pt/kibana/

A Figura 8 ilustra o Elasticsearch já inicializado no navegador e pronto para utilização.

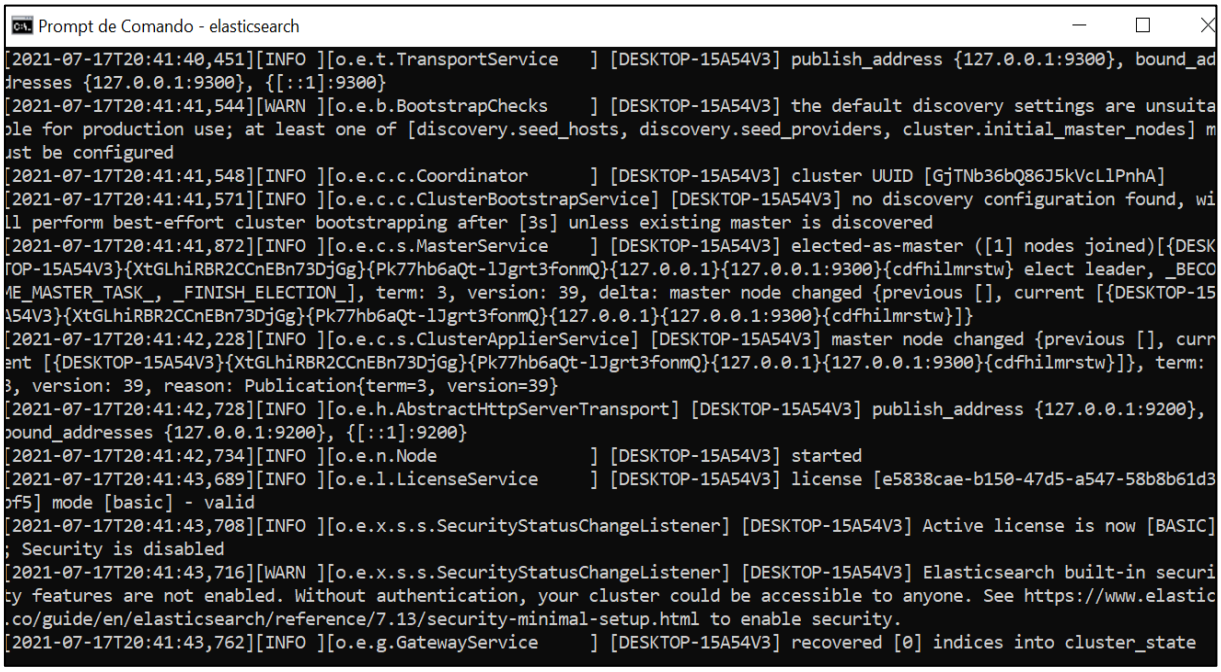
Figura 8 – Elasticsearch Iniciado



Fonte : Próprio autor.

Elasticsearch em execução fornecendo informações do nó, conforme Figura 9.

Figura 9 - Elasticsearch em execução.



Fonte: Próprio autor.

Conforme comentado anteriormente, a arquitetura proporciona o trabalho em cluster de nós, que é um conjunto de nós responsável pela distribuição da carga de dados. Esta implementação em cluster melhora o desempenho nas consultas de dados e a segurança, evitando assim a perda do acesso aos dados caso algum nó falhe.

3.3 Vantagens e desvantagens do Elasticsearch

O Elasticsearch oferece benefícios para busca de dados. Esta ferramenta NoSQL possui funcionalidades referentes aos requisitos tradicionais de grande volume de dados, tais como o tempo de resposta.

No mercado de tecnologia, empresas como a Netflix, Facebook, Dell, Airbus e GitHub utilizam o Elasticsearch. A necessidade de encontrar informações por milhões de usuários, de forma simultânea, das mais variadas fontes que consigam melhorar o atendimento e a experiência do cliente é a grande vantagem para que empresas citadas anteriormente busquem a solução de busca elástica.

A partir do estudo realizado, pode-se observar que a escolha de uma estratégia de armazenamento de dados depende dos requisitos da aplicação. Cada modelo possui suas vantagens e desvantagens, inexistindo situação ótima para todos os

cenários. Fatores como atualizações em termos de dados arquivados, mudança anual de versões, caso exista muita personalização, automação ou o volume de dados devem ser avaliados para se tomar a decisão.

No que se refere ao Elasticsearch, a dificuldade no entendimento da linguagem utilizada pela ferramenta, no caso Domain Specific Language (DSL), pode ser um fator impeditivo, por ser pouco difundida no mercado e com poucos profissionais que compreendem a linguagem.

Além disso, configurar índices ou pontos de dados para a consulta dos dados também pode ser desafiador. O suporte ao cliente ainda não é considerado proativo para quem enfrenta prazos e requer atualização dos índices a cada nova versão, o que não é uma tarefa fácil, caso não se tenha uma equipe de infraestrutura adequada.

O Elasticsearch, como motor de busca na sua característica, pode ser utilizado como ferramenta de armazenamento NoSQL orientada a documentos, mas pesquisar e juntar documentos diferentes é desafiador, o que gera na maior parte das tentativas o trabalho de desnormalizar documentos a fim de aperfeiçoar o modelo da pesquisa. Também foi identificado que a documentação das APIs pode ser melhorada, pois não é intuitivo a pesquisa de uma funcionalidade para determinada implementação.

3.4 Metodologia

Para atingir o objetivo do estudo de comparar as tecnologias apresentadas, pode-se destacar três funções a serem utilizadas: coleta, armazenamento e visualização. Para coletar informações, é possível utilizar scripts de inserção de informações na base de dados e/ou agentes instalados.

A apresentação das informações pode ser realizada por documentos JSON, DSL e por meio da exploração dos painéis que a ferramenta Kibana fornece em conjunto ao Elasticsearch. Além disso, para continuidade dos testes referentes às funcionalidades das soluções, foram realizadas as seguintes etapas:

- Carga de dados no Elasticsearch e no MySQL;
- Realização de consultas com linguagem DSL e SQL;
- Listagem de dados para testes de desempenho.

3.5 Descrição do Experimento

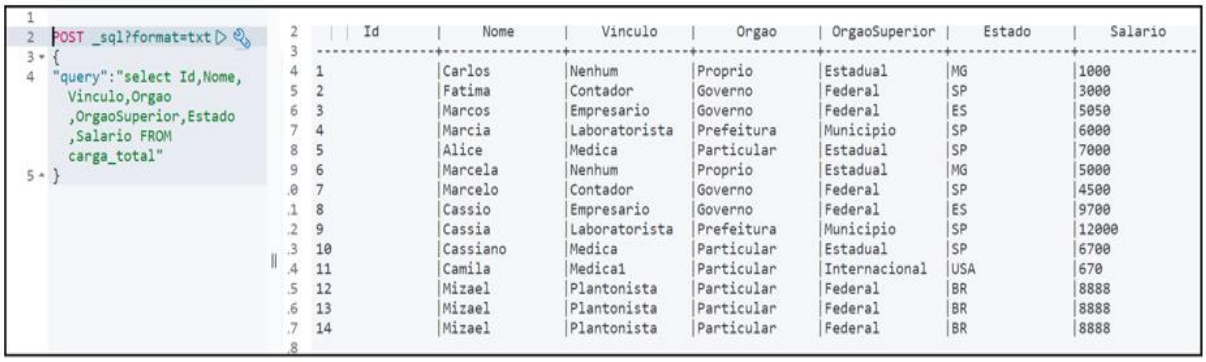
A parte inicial dos testes foi a inserção de informações utilizando um arquivo CSV. A partir deste arquivo, o Logstash realiza o tratamento e o envio das informações ao Elasticsearch. Para o banco de dados relacional foi realizando a inserção de informações por meio da console do próprio MySQL direto no *schema* do banco de dados.

O segundo passo foi testar os comandos utilizando a linguagem DSL a fim de validar a sintaxe e sua usabilidade, conforme representado na Figura 10, mostrando a listagem de todos os registros utilizados na simulação.

O arquivo de carga (.csv) é armazenado em um determinado diretório e configurado sua ingestão por meio de um agente que cria o índice no Elasticsearch. A instalação utilizada é uma estação de trabalho, com sistema operacional Windows e MySQL.

A seguir é apresentado o experimento para avaliação da sintaxe e usabilidade do modulo de desenvolvimento do Elasticsearch utilizando a linguagem DSL, mas que também suporte à linguagem *SQL*.

Figura 10 - Listagem Geral do Índice (Carga_total).



```
POST _sql?format=txt
{
"query":"select Id,Nome,
  Vinculo,Orgao
  ,OrgaoSuperior,Estado
  ,Salario FROM
  carga_total"
}
```

Id	Nome	Vinculo	Orgao	OrgaoSuperior	Estado	Salario
1	Carlos	Nenhum	Proprio	Estadual	MG	1000
2	Fatima	Contador	Governo	Federal	SP	3000
3	Marcos	Empresario	Governo	Federal	ES	5050
4	Marcia	Laboratorista	Prefeitura	Municipio	SP	6000
5	Alice	Medica	Particular	Estadual	SP	7000
6	Marcela	Nenhum	Proprio	Estadual	MG	5000
7	Marcelo	Contador	Governo	Federal	SP	4500
8	Cassio	Empresario	Governo	Federal	ES	9700
9	Cassia	Laboratorista	Prefeitura	Municipio	SP	12000
10	Cassiano	Medica	Particular	Estadual	SP	6700
11	Camila	Medica1	Particular	Internacional	USA	670
12	Mizael	Plantonista	Particular	Federal	BR	8888
13	Mizael	Plantonista	Particular	Federal	BR	8888
14	Mizael	Plantonista	Particular	Federal	BR	8888

Fonte: Próprio autor.

- Index (Índice): Tem função de inserir documentos JSON ao índice ou determinado fluxo de dados mantendo-o consultável, caso já exista o índice especificado este será atualizado com as novas informações ou incrementado com novas informações. A seguir é apresentado um exemplo da inserção de um registro no fluxo de documentos com apenas os dados de Id e Nome. A sintaxe POST

/carga_total/_doc/15 { “Id”: “15”,“Nome”: [“Joao da Silva1”]} é ilustrada conforme Figura 11.

Figura 11 - Documento *JSON* do Registro Inserido no ElasticSearch.

Fonte: Próprio autor.

- GET (Listagem): No ElasticSearch a paginação para a exibição dos registros é de dez registros por página. No caso do exemplo existem 15 registros, sendo assim, para a exibição de todos os registros ao invés de utilizar GET carga_total/_search que exibe apenas os dez primeiros registros, pode-se utilizar GET carga_total/_*search*?size=50&q=*:* que consiste na paginação de cinquenta registros por consulta realizada, conforme ilustração da Figura 12.

Figura 12 – Listagem de Registros com Paginação.

Fonte: Próprio autor.

- Delete (Exclusão): A deleção dos registros da API remove o documento JSON do índice, no caso o índice “carga_total” terá seu documento JSON de Id=15

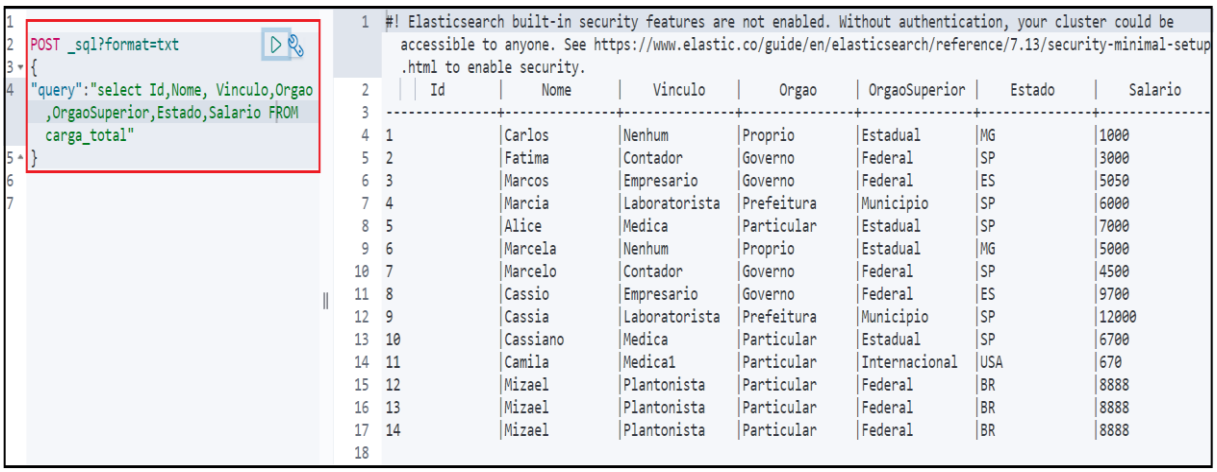
removido conforme comando: DELETE /carga_total/_doc/15, realizado com sucesso, conforme Figura 13.

Figura 13 – Exclusão de documento do Índice.

Fonte: Próprio autor.

A Figura 14 exemplifica a listagem de todos os registros novamente a fim de verificar que o registro do Id=15 foi excluído com sucesso.

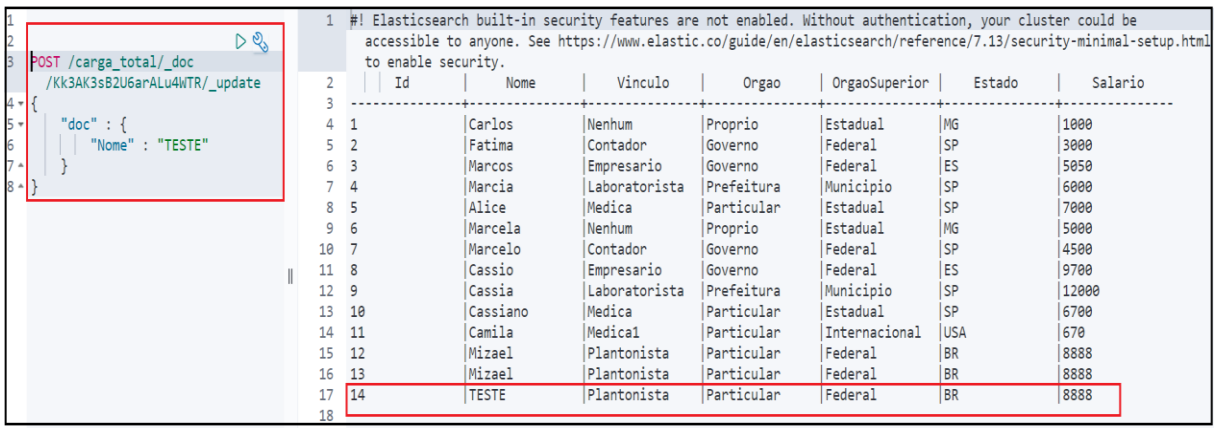
Figura 14 - Listagem de todos os registros.



Fonte: Próprio autor.

- Update (Atualização): A atualização dos registros da API altera os valores já armazenados no índice, abaixo segue exemplo da sintaxe comando de atualização conforme Figura 15, apenas o campo (Nome) no documento para o ID 14 foi alterado.

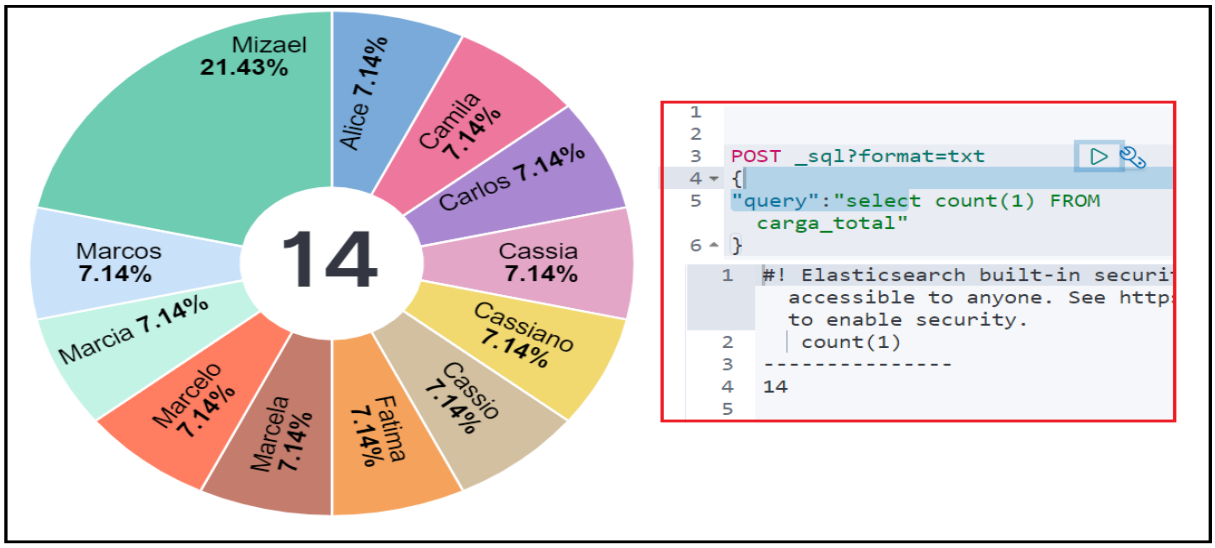
Figura 15 - Atualização de Nome de um Registro.



Fonte: Próprio autor.

Finalizando o contexto de APIs, segue o resumo geral da exibição de todos os dados utilizando a ferramenta de consulta e desenvolvimento do ElasticSearch e a visualização em forma de painel contabilizando os registros armazenados por nome e representatividade percentual exemplificado na Figura 16.

Figura 16 - Painel x Query Console de Desenvolvimento do ElasticSearch.



Fonte: Próprio autor.

Após a experimentação da sintaxe DSL, conclui-se que sua utilização é efetiva em usabilidade, pois pode ser utilizada para consultas tanto utilizando o modulo de desenvolvimento do Elasticsearch como o próprio navegador e possibilita a inserção da linguagem SQL, evitando assim a necessidade de acesso a console de

gerenciamento do banco de dados relacional MySQL para executar determinadas tarefas de inserção, deleção, atualização e leitura.

Após compreender melhor a sintaxe utilizada pela ferramenta NoSQL, o experimento continua com a análise de desempenho. Para isso, foi utilizada uma tabela com 4.817 registros. Primeiro foi realizado o teste de inserção destes registros e em seguida a medição de tempo de apresentação das informações por meio de um filtro confeccionado de forma igualitária para ambas as tecnologias.

No MySQL foi utilizada a inserção por linha de comando diretamente pelo console de administração do banco de dados. Para o NoSQL, a ingestão foi realizada pelo console de administração da ferramenta Elasticsearch.

Primeiramente realiza-se a carga dos dados para o MySQL, conforme Tabela 3.

Tabela 3 - Modelo de Carga de Dados no MySQL.

```
INSERT INTO clientes (id,nome,vinculo,orgao,orgaosuperior,estado,salario) values (1,'Sonia','SP','SSP','SSSP','SP',5000);
INSERT INTO clientes (id,nome,vinculo,orgao,orgaosuperior,estado,salario) values (2,'Sonia','SP','SSP','SSSP','SP',5000);
INSERT INTO clientes (id,nome,vinculo,orgao,orgaosuperior,estado,salario) values (3,'Sonia','SP','SSP','SSSP','SP',5000);
INSERT INTO clientes (id,nome,vinculo,orgao,orgaosuperior,estado,salario) values (4,'Sonia','SP','SSP','SSSP','SP',5000);
INSERT INTO clientes (id,nome,vinculo,orgao,orgaosuperior,estado,salario) values (5,'Sonia','SP','SSP','SSSP','SP',5000);
INSERT INTO clientes (id,nome,vinculo,orgao,orgaosuperior,estado,salario) values (6,'Sonia','SP','SSP','SSSP','SP',5000);
INSERT INTO clientes (id,nome,vinculo,orgao,orgaosuperior,estado,salario) values (4817,'TESTE','SP','SSP','SSSP','SP',194787);
```

Fonte: Próprio autor.

As cargas realizadas para o MySQL foram executadas com três amostras, a primeira com (4817) registros, a segunda com (9634) e a terceira com (14451) conforme Tabela 4 e Figuras 20, 21 e 22.

Tabela 4 - Tempo de Inserção de Dados (MySQL).

Teste do Tempo de Inserção de Dados (MySQL)							
Bancos	Tecnologias	Numero de Registros1		Numero de Registros2		Numero de Registros3	
Relacional	MySQL	4817		9634		14451	
	Tempos	Tempo Inicial	Tempo Final	Tempo Inicial	Tempo Final	Tempo Inicial	Tempo Final
		18:38:52	18:39:45	18:49:36	18:56:10	19:03:07	19:20:00
	Tempo Total	00:00:53		00:06:34		00:16:53	

Fonte: Próprio autor.

Em seguida, realiza-se a carga dos dados para o Elasticsearch conforme Tabela 5.

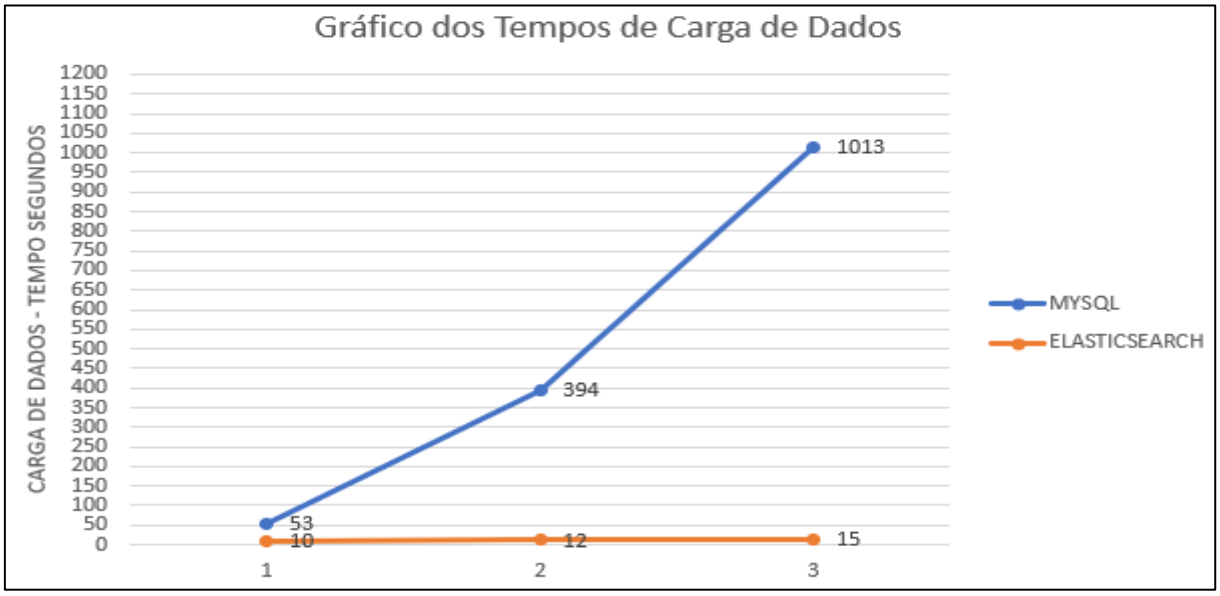
Tabela 5 - Tempo de Inserção de Dados (Elasticsearch)

Teste do Tempo de Inserção de Dados (Elasticsearch)							
Bancos	Tecnologias	Numero de Registros1		Numero de Registros2		Numero de Registros3	
NoSQL	Elasticsearch	4817		9634		14451	
	Tempos	Tempo Inicial	Tempo Final	Tempo Inicial	Tempo Final	Tempo Inicial	Tempo Final
		17:55:42	17:55:52	21:25:38	21:25:50	21:44:59	21:45:14
	Tempo Total	00:00:10		00:00:12		00:00:15	

Fonte: Próprio autor.

A partir desses dados, é possível observar, conforme apresentado na Figura 17, que a carga de dados no Elasticsearch apresentou desempenho superior ao do MySQL, em todas as amostras utilizadas.

Figura 17 - Gráfico comparativo referente a carga de dados x tempo.



Fonte: Próprio autor.

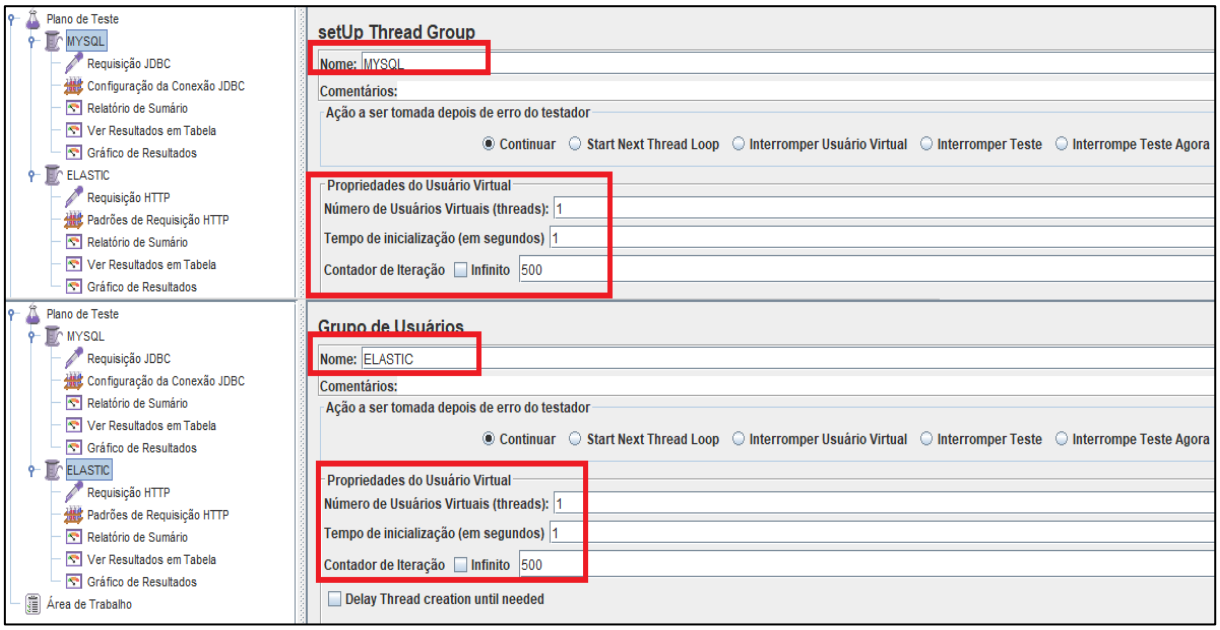
Assim, utilizando as ferramentas de dados MySQL e Elasticsearch, foi possível observar o comportamento de ambas as tecnologias utilizando amostras diferentes para o processo de inserção de dados. O Elasticsearch se mostrou mais efetivo na questão tempo.

A próxima comparação entre MySQL e Elasticsearch será no âmbito de consultas de dados, em que os registros utilizados são os mesmos do teste de carga de dados. Para este teste, foi utilizada a ferramenta JMeter com a intenção de medir suas variáveis, tempo da amostra que se refere ao tempo total da requisição em m/s e

vazão (*throughput*) que se refere ao número de operações que o sistema é capaz de completar em um determinado período.

Na Figura 18 é apresentado que o número de iterações para o teste de desempenho para o MySQL e o Elasticsearch foram sempre os mesmos, para a finalidade de observar e entender melhor seu comportamento.

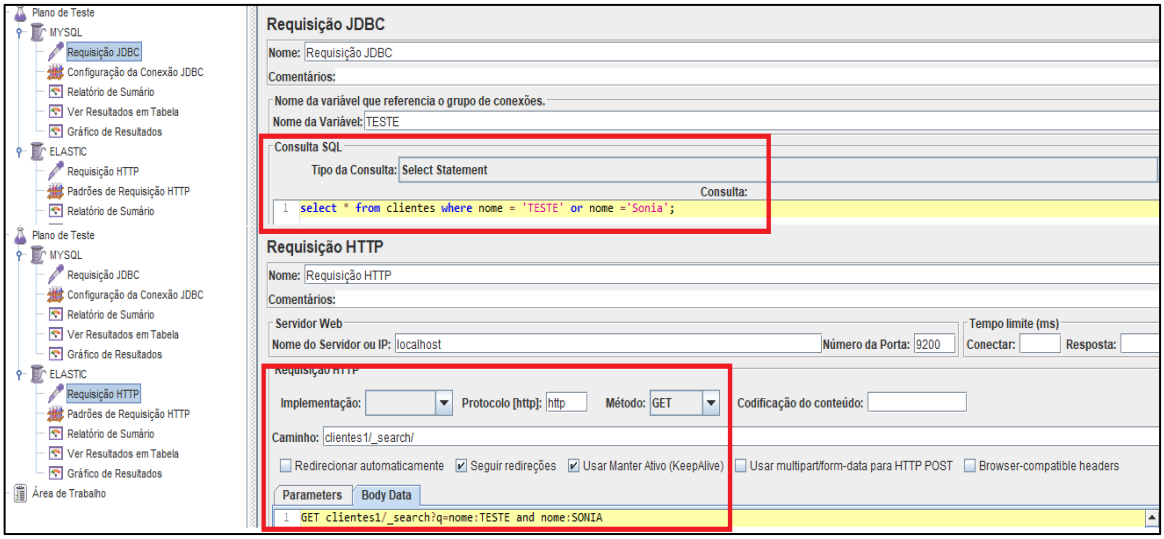
Figura 18– Parâmetros do MySQL e Elasticsearch no JMeter.



Fonte: Próprio autor.

Após definidos os valores de iteração, a definição da consulta a ser realizada para o MySQL e o Elasticsearch também são as mesmas, porém cada uma na sua característica. O MySQL com a linguagem SQL e o Elasticsearch com a linguagem DSL, conforme Figura 19.

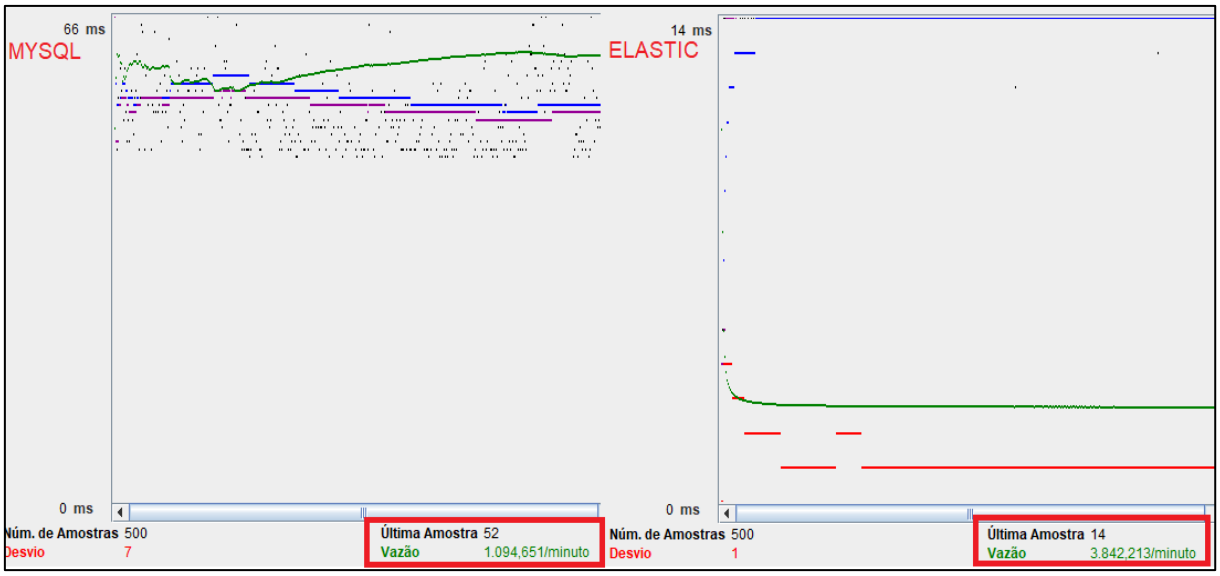
Figura 19 – Parâmetros de requisição MySQL e Elasticsearch no JMeter.



Fonte: Próprio autor.

Com os parâmetros já configurados, inicia-se os testes começando com a primeira rodada de 500 amostras, a segunda com 1000 e a terceira com 2000 iterações. Para a primeira etapa de testes, as interações com 500 amostras no MySQL apresentaram maior tempo de resposta que o Elasticsearch e menor vazão, sendo assim o Elasticsearch consegue completar com menor tempo mais requisições, conforme Figura 20.

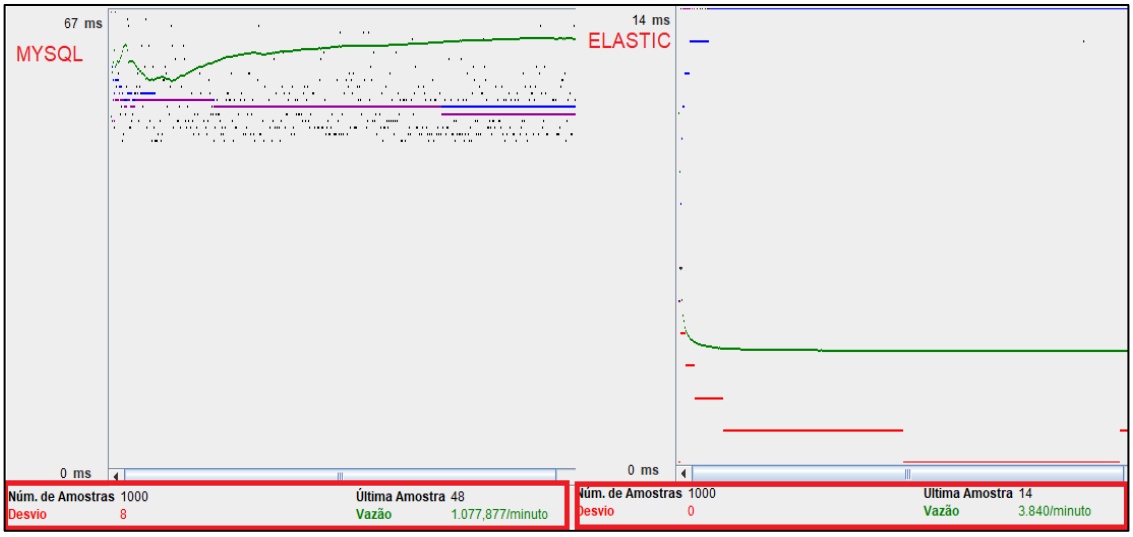
Figura 20 – Tempos e vazão (500) com requisições no MySQL e Elasticsearch com JMeter.



Fonte: Próprio autor.

Para a segunda etapa de testes as iterações, com 1000 amostras, o MySQL apresentou maior tempo de resposta que o Elasticsearch e menor vazão, sendo assim o Elasticsearch consegue completar com menos tempo mais requisições, conforme Figura 21.

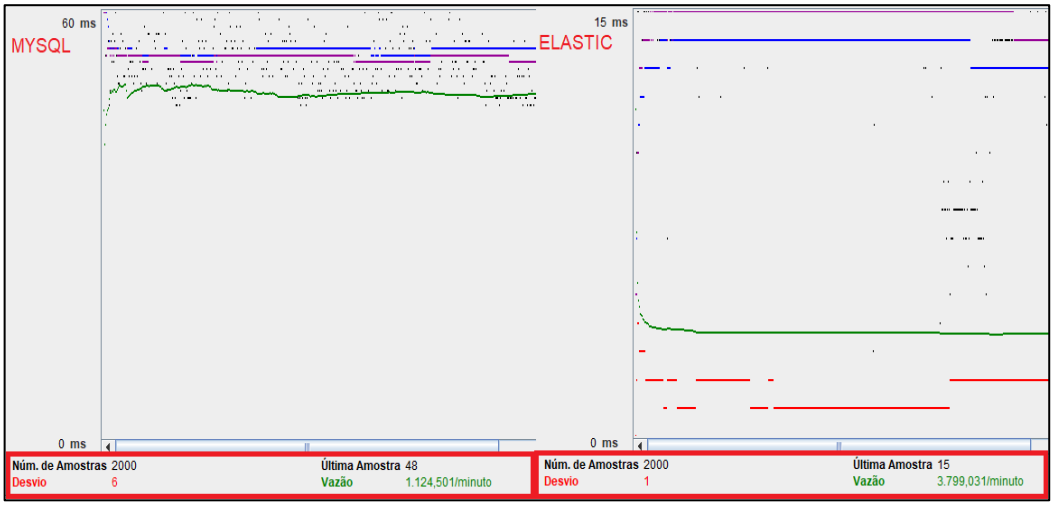
Figura 21 – Tempos e vazão (1000) com requisições no MySQL e Elasticsearch com JMeter.



Fonte: Próprio autor.

Para a terceira e última etapa de testes as iterações com 2000 amostras o MySQL apresentou maior tempo de resposta que o Elasticsearch e menor vazão, completando com menor tempo mais requisições (Figura 22).

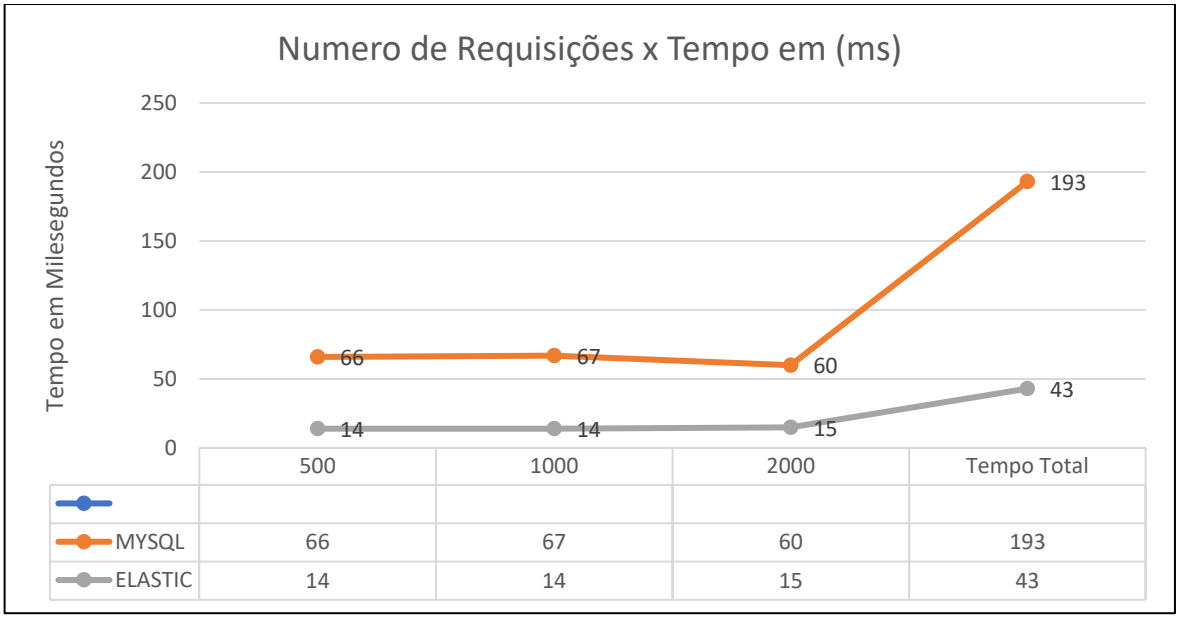
Figura 22 - Tempos e vazão (2000) com requisições no MySQL e Elasticsearch com JMeter.



Fonte: Próprio autor.

Finalizados todos os testes de carga e desempenho de consultas, o gráfico da Figura 23 apresenta o número de amostras de requisição completadas pelo tempo.

Figura 23 – Gráfico de desempenho das consultas.



Fonte: Próprio autor.

Finalizados os testes de carga e desempenho, conclui-se que o Elasticsearch obteve melhor tempos no processo de inserção de dados, tendo resposta mais rápida para os mesmos números de registros utilizados no MySQL, a parte de desempenho de consultas também foram mais bem sucedidas no Elasticsearch, uma vez que para todas as amostragens testadas o tempo foi menor, com mais requisições sendo completadas em menor espaço de tempo.

4 CONCLUSÃO

Este trabalho apresentou um estudo comparativo entre o MySQL e o Elasticsearch, comparando-os através da revisão da literatura e de testes práticos, visando compreender como essas soluções são utilizadas na gestão de dados.

Do ponto de vista de implementação, ambas as ferramentas avaliadas possuem vantagens. Para o caso do Elasticsearch, os dados são inseridos e já entram indexados, já no MySQL, a indexação pode ser criada no momento da criação da tabela ou após ela já ter sido criada, o que gera melhor desempenho na realização de consultas. Nesse trabalho, portanto, os testes realizados não são completos, sendo necessário uma análise mais aprofundada, considerando o recurso de indexação.

Empresas que buscam alta escalabilidade tendem a escolher a tecnologia NoSQL para trabalhar com enorme volume de dados e manter sua alta disponibilidade para seus clientes com menor tempo de resposta para as consultas. Outro ponto importante a se destacar é que as tecnologias NoSQL não foram criadas para substituir as de modelo de banco de dados relacionais, que por sua vez já possuem sua aderência consolidada. Sua principal motivação é o gerenciamento de grandes volumes de dados, o que nem sempre é possível com o modelo relacional.

As duas tecnologias possuem pontos favoráveis e desfavoráveis, dependendo do sistema que se deseja criar ou implementar. Para ambos os casos, foi possível compreender melhor pontos que geram valor no processo de extração dos dados.

4.1 Trabalhos Futuros

Para trabalhos futuros, são projetados testes mais elaborados em ambientes empresariais, com volumes de dados maiores para testar fatores como confiabilidade, desempenho, escalabilidade, utilização de hardware e processamento em nuvem.

Outro trabalho futuro é apresentado no Apêndice B, com a proposta de criar uma arquitetura protótipo que já se encontra em desenvolvimento, com foco na atuação na área da saúde, onde as informações são armazenadas em banco de dados relacional MySQL. Nesse cenário, o Elasticsearch trabalha em conjunto com o MySQL e terá a responsabilidade de indexar as informações e ingestão em tempo real com a

utilização do agente de coleta Logstash, o Kibana será a ferramenta de Big Data que permitirá a geração de painéis das informações.

Também foi criado um modelo responsável pelas funções de inserção, leitura, atualização e deleção criado através da plataforma JavaNetbeans, integrando-se com a exibição de páginas em PHP e servidor HttpApacherServer, com finalidade de gerar painéis que venham a ser utilizados e customizadas por empresas, áreas, gerências, tornando assim a visualização e identificação dos dados e comunicação mais versátil e centralizada.

Nesse trabalho a proposta visa a validação de informações do setor com o objetivo de mensurar e identificar problemas em atendimento de chamados de analistas e para verificação de informações de dados de pacientes caso estejam apresentando problemas em cálculos de análises médicas como o mostrado do índice de massa corporal (IMC).

REFERÊNCIAS BIBLIOGRÁFICAS

ABUBAKAR, Y.; ADEYI, T. S.; AUTA, I. G. Performance Evaluation of NoSQL Systems Using YCSB in a resource Austere Environment. **International Journal of Applied Information Systems (IJAIS) – ISSN: 2249-0868,** v. 7, n. 8, p. 23-27, 2014. em: < https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.671.7275&rep=rep1&type=pdf> Acesso em: 15 jun de 2021.

ANICETO, R. C; XAVIER, R. F. **Um Estudo Sobre a Utilização do Banco de Dados *NoSQL* Cassandra em Dados Biológicos.** Disponível em: http://bdm.unb.br/bitstream/10483/7927/1/2014_RodrigoCardosoAniceto_Rene FreireXavier.pdf. Acesso em: 07 nov. 2021.

BAEZA-YATES, Ricardo; RIBEIRO-NETO, Berthier. **Recuperação de Informação: conceitos e tecnologias das máquinas de busca**. (2nd. Ed.) Porto Alegre: Bookman, 2013. Cap.3 p.106-122.

BRITO, Ricardo W. **Bancos de Dados NoSQL x SGBDs Relacionais: Análise Comparativa**. Faculdade Farias Brito e Universidade de Fortaleza, 2010. Disponível em: < http://www.infobrasil.inf.br/pagina/anais-2010 > Acesso em: 15 set. 2021.

CHOO, Chun Wei. **A Organização do conhecimento: como as organizações usam a informação para citar significado, construir conhecimento e tomar decisões**. 2 ed. São Paulo: SENAC São Paulo, 2006.
.
C4PPR4 D4T4 5C13NC3, 2017. **Afinal, o que é Big Data**. Disponível em: <https://medium.com/opsocial/afinal-o-que-%C3%A9-big-data-7ec0094dfb4c> Acesso em: 10 jun 2021.

DATE, C. J. **INTRODUÇÃO A SISTEMAS DE BANCOS DE DADOS**. 8. ed. Rio de Janeiro: Editora Campus, 2004.

DEMCHENKO, Y., GROSSO, P., DE LAAT, C. & Membrey, P. **Addressing Big Data issues in scientific data infrastructure.** Colaboration Technologies and Systems (CTS), 2013.

ELMASRI, Ramez; NAVATHE, Shamkant B. **SISTEMAS DE BANCO DE DADOS**.6. ed. São Paulo: Addison Wesley, 2011.

ELASTIC. **Elasticsearch como um banco de dados NoSQL**. Publicado em: 17 de janeiro de 2019. Disponível em: <https://www.semantix.com.br/elasticsearch-como-um-banco-de-dados-nosql/>. Acesso em: 16 de ago de 2021.

ELASTICSEARCH 2021. **O que é Elasticsearch**. Disponível em: <https://www.elastic.co/pt/what-is/elasticsearch> Acesso em: 18 de mai de 2021.
FRANK, C. **Improving Decision Making in the World of Big Data**. Disponível em: <https://www.forbes.com/sites/christopherfrank/2012/03/25/improving-decision-making-in-the-world-of-big-data/?sh=1a595d7b1e85> Acesso em: 10 jun 2021.

IDC. **Global Spending on Big Data and Analytics Solutions Will Reach $215.7 Billion in 2021, According to a New IDC Spending Guide.** Disponível em: < https://www.idc.com/getdoc.jsp?containerId=prUS48165721> Acesso em: 18 jul 2021.

JACKSON, J. **CouchBase, SQLite launch unified NoSQL query language**. Accessed January, v. 25, p. 2012, 2011.

KIBANA. **A sua janela para o Elastic Stack**. 2020. Disponível em: < https://www.elastic.co/pt/kibana/> Acesso em: 21 de jul de 2021.

KONONENKO, O. et al. Mining Modern Repositories with Elasticsearch**. Proceedings of the 11th Working Conference on Mining Software Repositories**. ACM. [S.l.]: [s.n.]. 2014. p. 328-331.

LOGSTASH. ***Logstash* 7.4.1 Release Notes**. 2020. Disponível em: < https://www.elastic.co/guide/en/logstash/7.4/logstash-7-4-1.html>. Acesso em: 21 de jul de 2021.

LOHR, S. ***The age of big data***. *The New York Times*, v. 161, n. 55679, p. 1, 2012.

MANYIKA, J.; CHUI, M. Big Data: **A Próxima Fronteira para Inovação, Concorrência e Produtividade**. McKinsey Global Institute, mai 2011.

MCAFEE, Andrew; BRYNJOLFSSON, Erik. Big Data. The management revolution. **Harvard Business Review**, v. 90, n. 10, 2012 p. 61–68. Disponível em: <https://hbr.org/2012/10/big-data-the-management-revolution>. Acesso em: 22 set. 2021.

MCMURTRY, D., OAKLEY, A., SHARP, J., SUBRAMANIAN, M., and ZHANG, H. **Data access for highly-scalable solutions: Using sql, nosql, and polyglot persistence**. Microsoft patterns & practices, 2013.

NONOHAY, Roberto Guedes. **Tomada de decisão e os sistemas cerebrais: primeiros diálogos entre administração, psicologia e neurofisiologia**. Porto Alegre, 2012.

NÄSHOLM, P. **Extracting Data from NoSQL Databases-A Step towards Interactive Visual Analysis of NoSQL Data**. Chalmers University of Technology, 2012.

PEREIRA, Júlio Cesar R. **Análise de Dados Qualitativos: Estratégias Metodológicas para as Ciências da Saúde, Humanas e Sociais**. 3 ed. 1 reimpr. São Paulo: Editora da Universidade de São Paulo, 2004.

PORCELLI, ALEXANDRE. **O que é NoSQL?**. Java Magazine, 86, jan. 2011.
PRAGMATEEK. **JSON vs. XML: Some Hard Numbers About Verbosity**. [S.l.], 2013. Disponível em: <https://www.codeproject.com/Articles/604720/JSON-vsXML-Somehard-numbers-about-verbosity>. Acesso em: 20 mai. 2021.

SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSHAN, S. **SISTEMA DE BANCO DE DADOS**. 3. ed. São Paulo: Makron Books, 1999.

SOUSA JUNIOR, Luis Carlos. **Replicação SQL e Replicação NoSQL**. Publicado em: 16 de setembro de 2014. Disponível em: <http://luiscarlossousaj.blogspot.com/2014/09/replicacao-sql-e-replicacao-nosql.html> Acesso em: 14 de mai de 2021.

SOUZA, Alexandre Morais de; FANTINATO, Marcelo; PRADO, Edmir Parada Vasques; SUN, Violeta. **Critérios para seleção de SGBD NoSQL**: o ponto de vista de especialistas com base na literatura. Anais. Londrina: UEL, 2014.

VIEIRA, M. R. et al. **Bancos de dados NoSql: conceitos, ferramentas, linguagens e estudos de casos no contexto de Big Data**. Simpósio Brasileiro de Bancos de Dados, 2012.

APÊNDICE A - CONFIGURAÇÃO DO LOGSTASH

Para início é preciso configurar um arquivo no formato (.conf) que contém as informações necessários para que o ElasticSearch identifique as configurações e extraia de onde vem os dados e para qual índice será direcionado, o (.conf) possui basicamente três parametrizações, o ponto inicial de (entrada), a função de filtragem e o ponto final que envia os dados transformados e cria automaticamente o índice.

O formato do arquivo com as informações é do tipo (.csv), com suas colunas separadas por (,) no formato (Id, Nome, Vinculo, Orgao, OrgaoSuperior, Estado, Salário) conforme segue conforme Figura 24.

Figura 24 – Arquivo (.csv) para ingestão de dados.

1	Id,Nome,Vinculo,Orgao,OrgaoSuperior,Estado,Salario
2	1,Carlos,Nenhum,Proprio,Estadual,MG,1000
3	2,Fatima,Contador,Governo,Federal,SP,3000
4	3,Marcos,Empresario,Governo,Federal,ES,5050
5	4,Marcia,Laboratorista,Prefeitura,Municipio,SP,6000
6	5,Alice,Medica,Particular,Estadual,SP,7000

Fonte: Próprio autor.

A configuração de ingestão do LogStash inicia com a entrada dos dados e com o apontamento da localização do arquivo conforme Figura 25.

Figura 25 - Função entrada de dados.

```
input {
        file {
            path => "C:/Logstash/logstash-7.13.4-windows-x86_64/logstash-7.13.4/bin/carga_total.csv"
            start_position => "beginning"
             }
       }
```

Fonte: Próprio autor.

As informações são configuradas na parte de filtragem, começando pela definição dos campos que serão as colunas separadas por (,) conforme figura 26.

Figura 26 – Colunas de dados.

```
filter {
      csv {
           separator => ","
            skip_header => true
            columns => ["Id","Nome","Vinculo","Orgao","OrgaoSuperior","Estado","Salario"]
            }
```

Fonte: Próprio autor.

Existe a possibilidade de conversão de data e hora para formato de campo de tempo (*timestamp*) que será utilizado pelo *Kibana*, conforme Figura 27.

Figura 27 – Conversão data e hora.

```
Date {
        match => ["Data_e_Hora","yyyy-MM-dd' 'HH:mm:ss'.'SSS"]
        targer => "@timestamp"
       }
```

Fonte: Próprio autor.

Figura 28 – Índice de saída das informações.

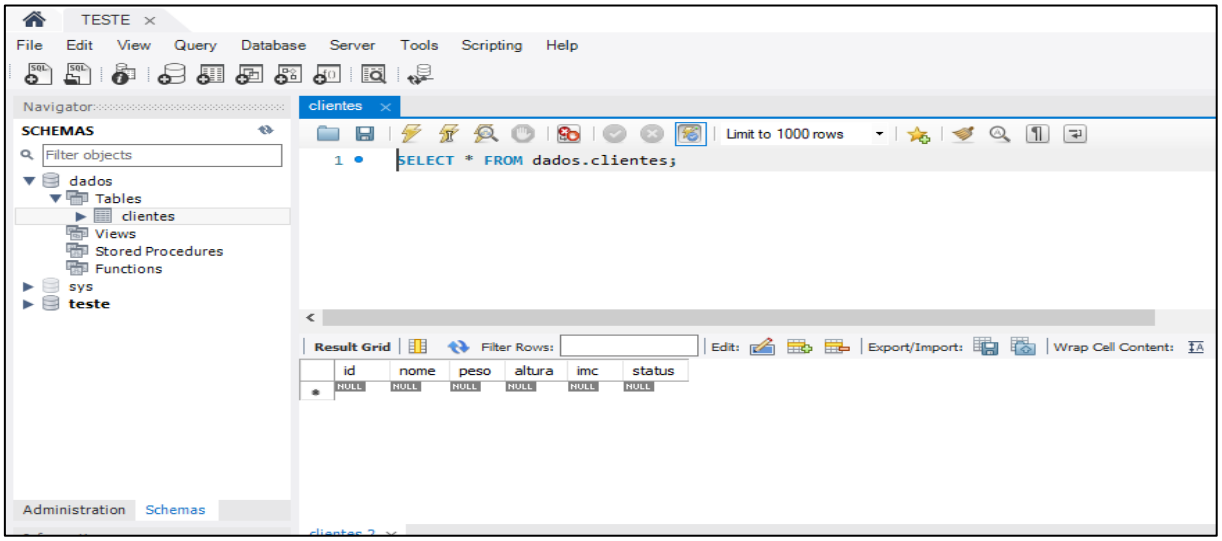
```
output {
   elasticsearch {
      index => "carga_total"
      hosts => "localhost:9200"
   }

}
```

Fonte: Próprio autor.

APÊNDICE B - DESENVOLVIMENTO DO PROTOTIPO

Inicialmente foi criado o banco de dados *MySQL* de nome (dados) com a tabela (clientes) conforme Figura 29:

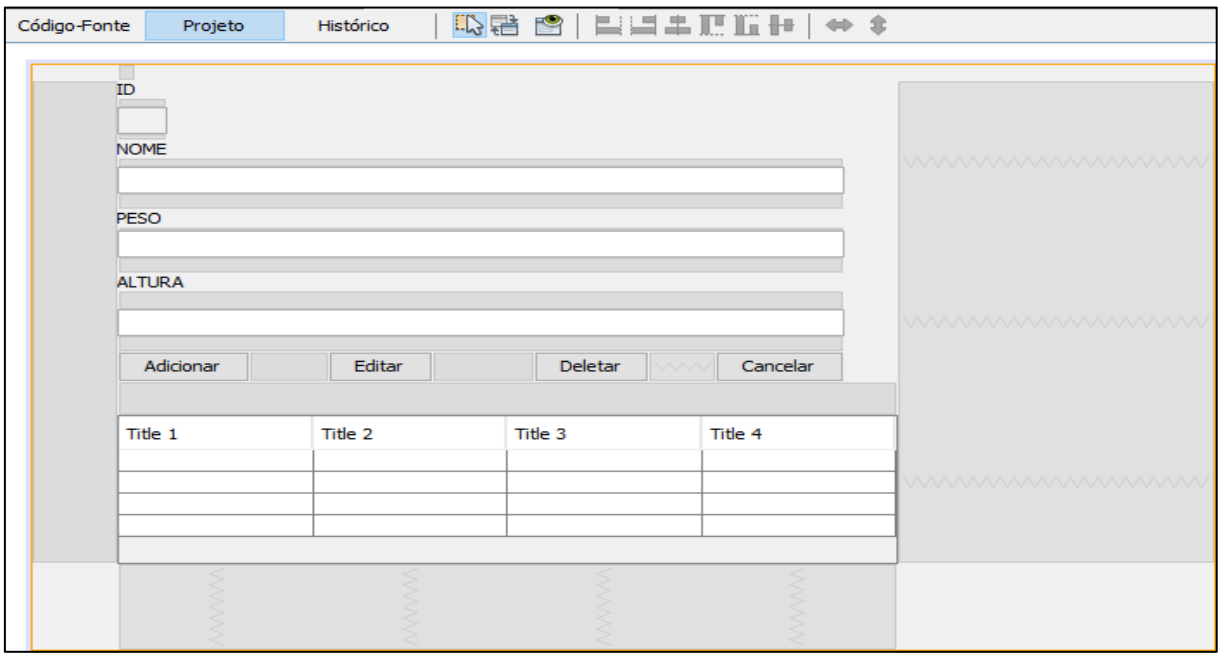
Figura 29 - Tabela (Clientes)



Fonte: Próprio autor.

A conexão do banco de dados da Figura 32 faz a conexão com o sistema (CRUD) desenvolvido em JavaNetbeans conforme Figura 30:

Figura 30 - (CRUD) em Java *NetBeans* IDE 8.1.



Fonte: Próprio autor.

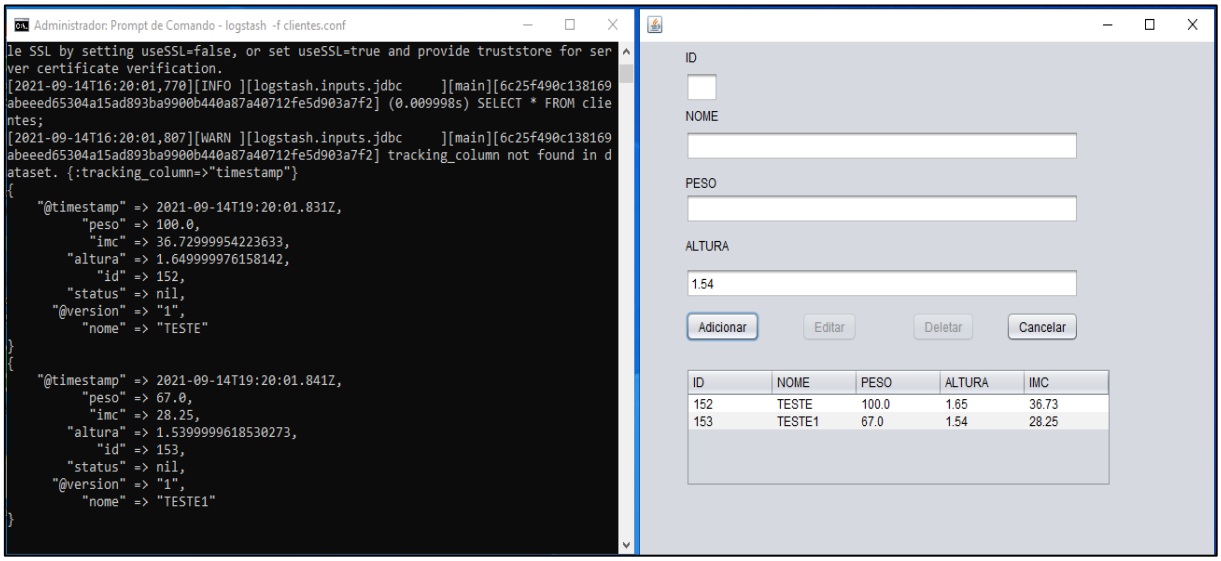
Após testar a conexão do sistema (CRUD) com o banco de dados MySQL foi instalado o Elasticsearch, o Logstash e o Kibana, o Elasticsearch será responsável pela indexação dos dados da tabela (clientes), o Logstash faz a conexão de integração entre o MySQL realizando a ingestão das informações inseridas e gravados no banco de dados MySQL através da ferramenta (CRUD), a Figura 31 mostra o código de integração feito por meio de JDBC:

Figura 31 - Código de conexão de Integração MySQL com Elasticsearch.

```
input {
  jdbc {
    jdbc_driver_library => "C:\DOC_LUCIANO\sistema\lib\mysql-connector-java-5.1.48-bin.jar"
    jdbc_driver_class => "com.mysql.jdbc.Driver"
    jdbc_connection_string => "jdbc:mysql://localhost:3306/dados"
    jdbc_user => root
    jdbc_password => teste1234
    tracking_column => "timestamp"
    use_column_value => true
statement => "SELECT * FROM clientes;"
    schedule => "*/2 * * * *"
  }
}
output {
  elasticsearch {
    hosts => ["localhost:9200"]
    index => "clientes"
document_id=> "%{id}"
  }
  stdout {
    codec => rubydebug
  }
}
```

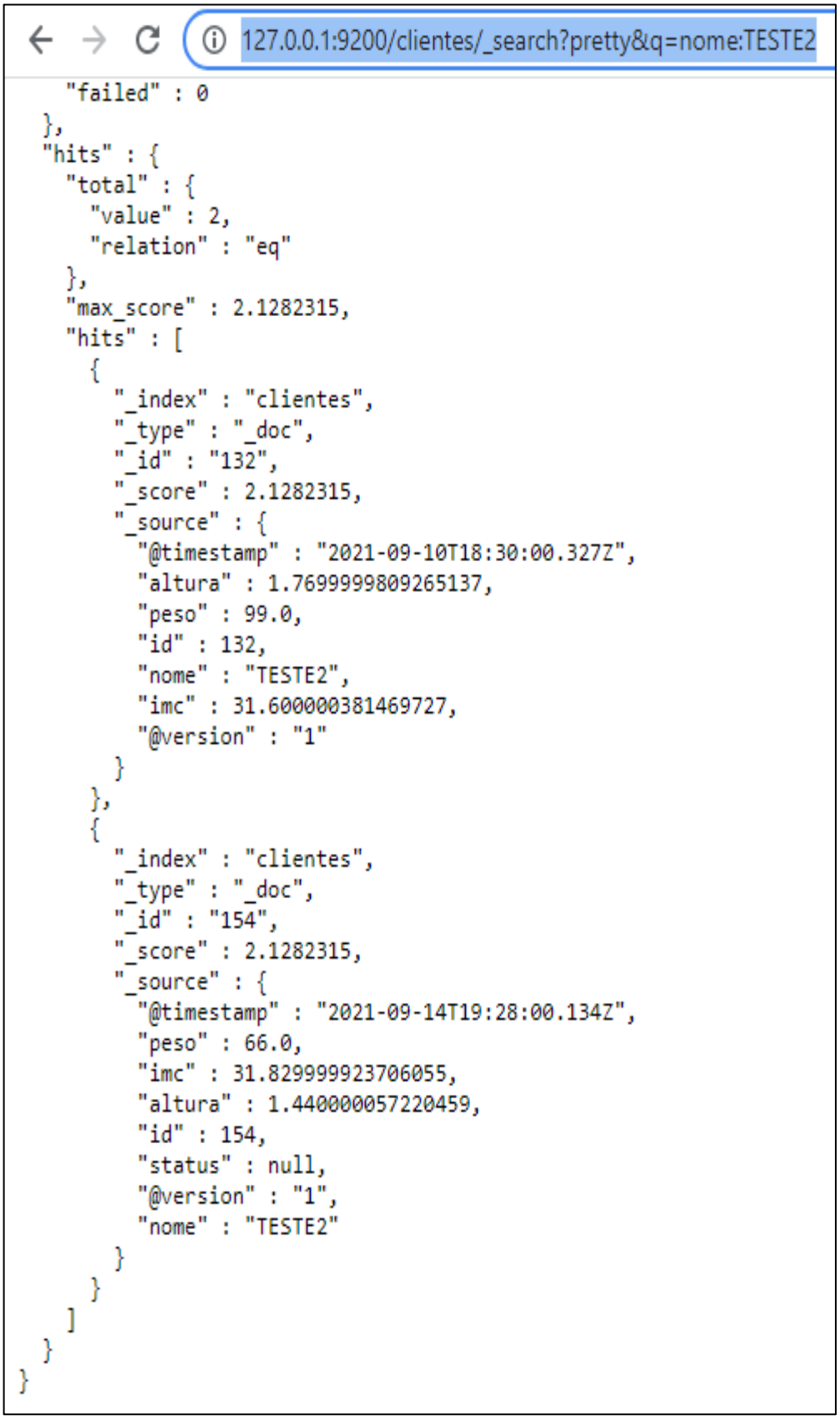
Fonte: Próprio autor.

Figura 32 - Logstash sincronizando dados.



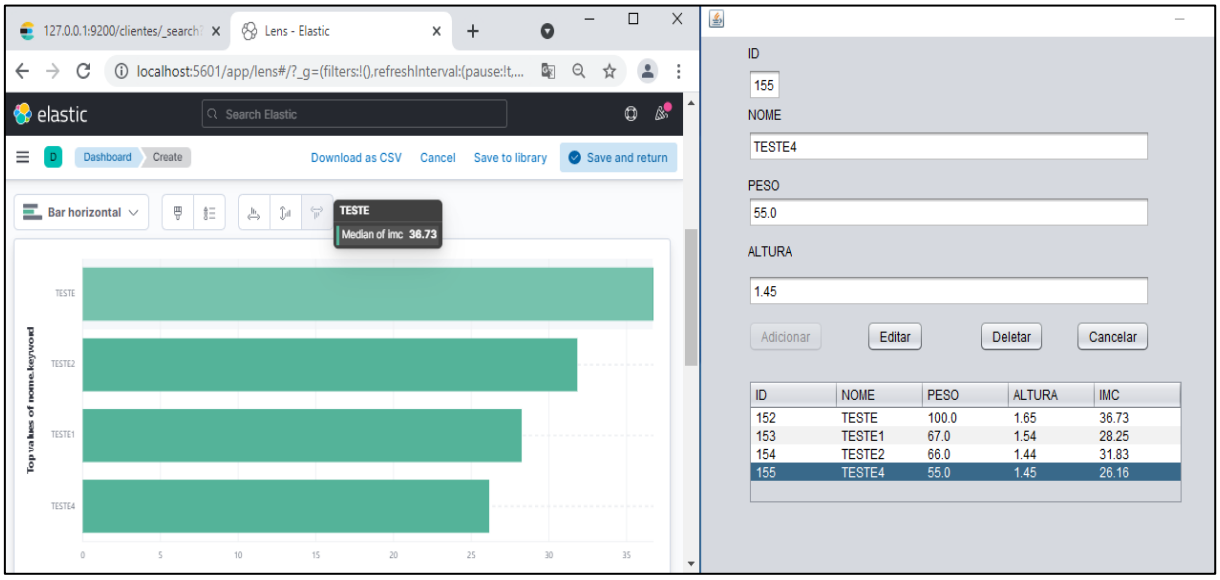
Fonte: Próprio autor.

Figura 33 - Dados listados pelo *browser* no Elasticsearch.



Fonte: Próprio autor.

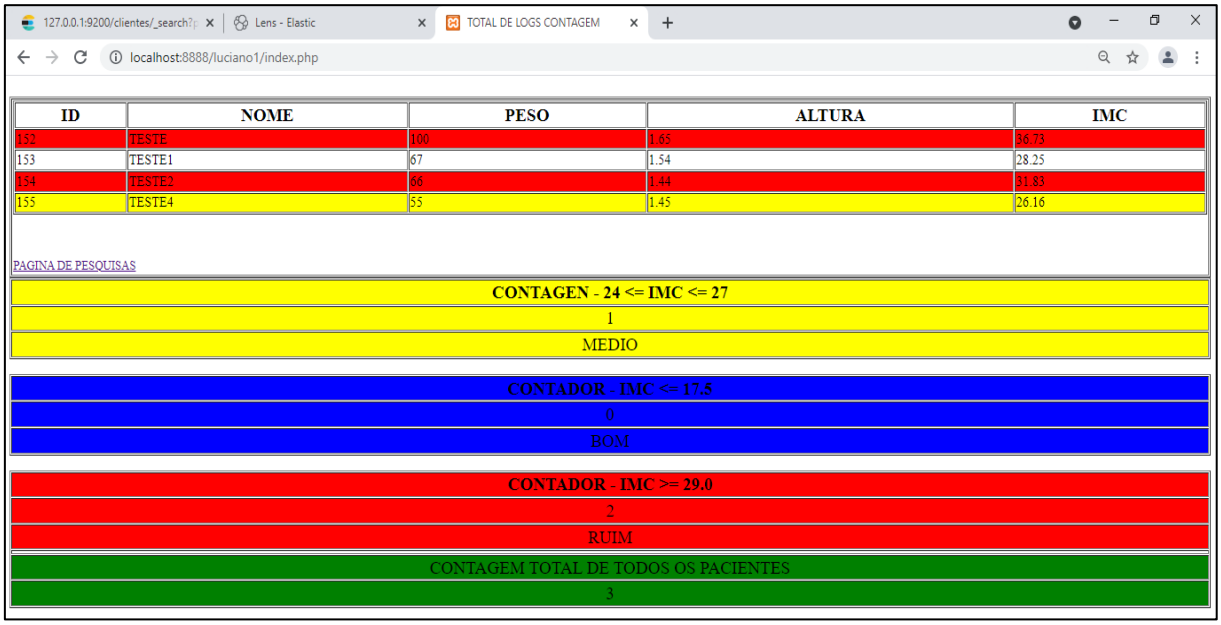
Figura 34 - Painel no Kibana.



Fonte: Próprio autor.

Os dados podem ser verificados tanto no Elasticsearch como no PHP e abaixo foi configurada uma aplicação que simula cores para diferenciar o nível de IMC dos pacientes pela coloração, sendo (vermelho) mais preocupante, (amarelo) sendo sinal de atenção, (azul) em nível bom sem muitas preocupações, na cor (branco) não se enquadra como nível algum de preocupação e a cor (verde) faz uma contagem total da somatória de todos os status de IMC alarmados conforme Figura 35.

Figura 35 - Aplicação configurada para *Web* com *PHP*.



Fonte: Próprio autor.

A arquitetura a seguir exemplifica como ambiente de protótipo visa mostrar a implementação prática de uma solução que pode gerar alertas caso algum dado esteja mais fora do que se pretende em um tratamento médico, podendo assim armazenar grandes volumes e utilizar de todo o poder de Big Data para análise dos dados e abaixo a Figura 36 ilustra os componentes necessários para a sua implementação.

Figura 36 - Exemplo da Arquitetura Proposta.

-

Fonte: Próprio autor.

A seguir o endereço disponível para acesso do um vídeo para compreensão do protótipo: https://www.youtube.com/watch?v=rKDDMUNx8Y8

APÊNDICE C - CARGA DE DADOS VIA LOGSTASH

Tabela 6 **–** Carga de Dados utilizando Logstash

	A	B	C	D	E	F	G
1	Id	Nome	Vinculo	Orgao	OrgaoSup	Estado	Salario
2	1	Carlos	Nenhum	Proprio	Estadual	MG	1000
3	2	Fatima	Contador	Governo	Federal	SP	3000
4	3	Marcos	Empresari	Governo	Federal	ES	5050
5	4	Marcia	Laborator	Prefeitura	Municipio	SP	6000
6	5	Alice	Medica	Particular	Estadual	SP	7000

Fonte : Próprio autor.

Abaixo o agente de ingestão do Logstash Figura 40 será executado, o comando de execução realizada um chamado no arquivo (.csv) que é a tabela acima e o arquivos de configuração (.conf) configurado com as parametrizações das Figuras do Apêndice A.

Figura 37 – Execução (.conf) para ingestão de dados.

```
C:\Logstash\logstash-7.13.4-windows-x86_64\logstash-7.13.4\bin>logstash -f carga_dados.conf
```

Fonte: Próprio autor.

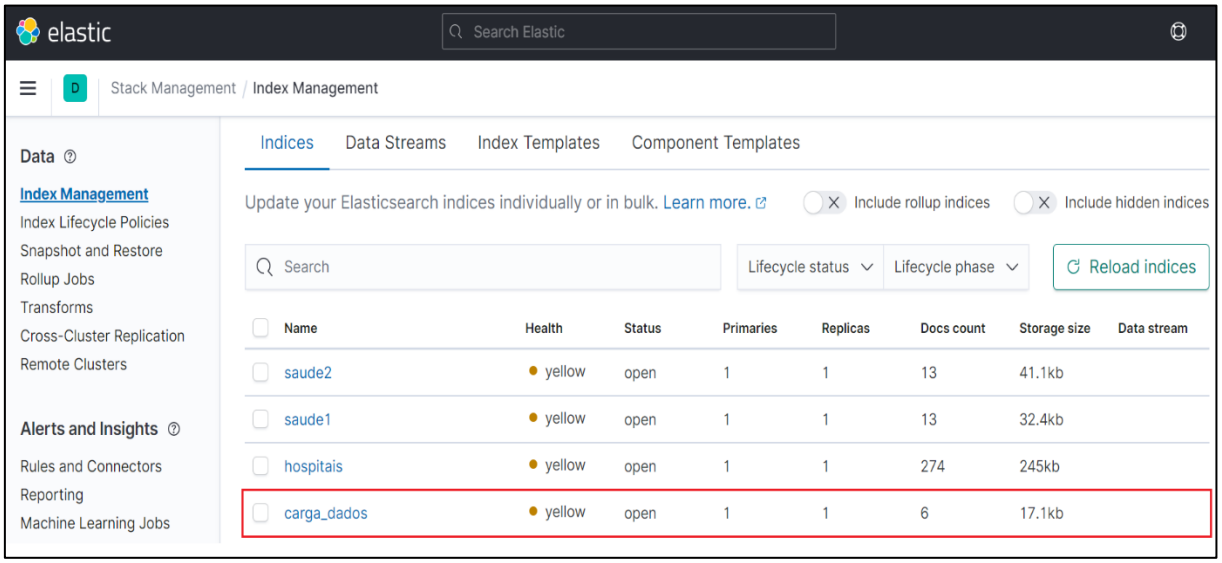
Figura 38 – Logstash em execução.

```
Administrador: Prompt de Comando - logstash  -f carga_dados.conf

C:\Logstash\logstash-7.13.4-windows-x86_64\logstash-7.13.4\bin>logstash -f carga_dados.conf
"Using bundled JDK: ""
OpenJDK 64-Bit Server VM warning: Option UseConcMarkSweepGC was deprecated in version 9.0 and will likely be removed in
a future release.
Sending Logstash logs to C:/Logstash/logstash-7.13.4-windows-x86_64/logstash-7.13.4/logs which is now configured via log
4j2.properties
[2021-08-05T20:04:26,866][INFO ][logstash.runner          ] Log4j configuration path used is: C:\Logstash\logstash-7.13.
4-windows-x86_64\logstash-7.13.4\config\log4j2.properties
[2021-08-05T20:04:26,881][INFO ][logstash.runner          ] Starting Logstash {"logstash.version"=>"7.13.4", "jruby.vers
ion"=>"jruby 9.2.16.0 (2.5.7) 2021-03-03 f82228dc32 OpenJDK 64-Bit Server VM 11.0.11+9 on 11.0.11+9 +indy +jit [mswin32-
x86_64]"}
[2021-08-05T20:04:27,006][WARN ][logstash.config.source.multilocal] Ignoring the 'pipelines.yml' file because modules or
 command line options are specified
[2021-08-05T20:04:28,800][INFO ][logstash.agent           ] Successfully started Logstash API endpoint {:port=>9600}
[2021-08-05T20:04:29,113][INFO ][org.reflections.Reflections] Reflections took 78 ms to scan 1 urls, producing 24 keys a
nd 48 values
[2021-08-05T20:04:31,639][INFO ][logstash.outputs.elasticsearch][main] New Elasticsearch output {:class=>"LogStash::Outp
uts::ElasticSearch", :hosts=>["//localhost:9200"]}
[2021-08-05T20:04:32,286][INFO ][logstash.outputs.elasticsearch][main] Elasticsearch pool URLs updated {:changes=>{:remo
ved=>[], :added=>[http://localhost:9200/]}}
[2021-08-05T20:04:32,603][WARN ][logstash.outputs.elasticsearch][main] Restored connection to ES instance {:url=>"http:/
/localhost:9200/"}
[2021-08-05T20:04:32,728][INFO ][logstash.outputs.elasticsearch][main] Elasticsearch version determined (7.13.3) {:es_ve
rsion=>7}
[2021-08-05T20:04:32,744][WARN ][logstash.outputs.elasticsearch][main] Detected a 6.x and above cluster: the `type` even
t field won't be used to determine the document _type {:es_version=>7}
[2021-08-05T20:04:32,931][INFO ][logstash.outputs.elasticsearch][main] Using a default mapping template {:es_version=>7,
 :ecs_compatibility=>:disabled}
[2021-08-05T20:04:32,963][INFO ][logstash.javapipeline    ][main] Starting pipeline {:pipeline_id=>"main", "pipeline.wor
```

Fonte: Próprio autor.

Após a execução do Logstash que pode ser verificado também no Apendice A, o Kibana já cria o indice que foi indicado no arquivo (carga_dados.conf), exemplo Figura 39:

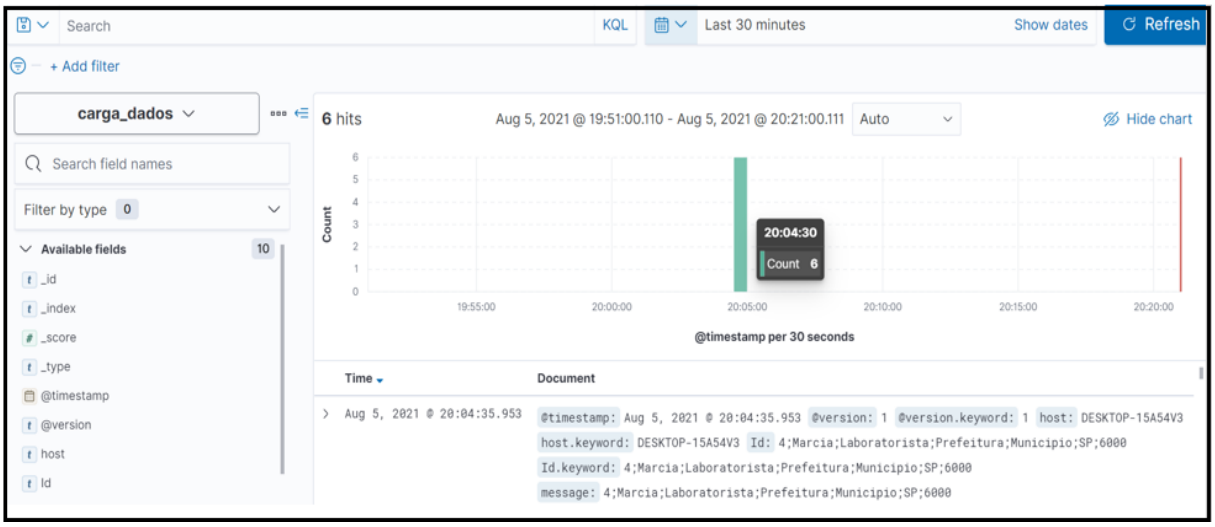
Figura 39 – Índice no Kibana.



Fonte: Próprio autor.

Criado o indice no gerenciamento de indices do Kibana já é possivel visualizar os dados ingeridos pelo ElasticSearch e manipular os mesmos, podendo realizar suas consultas, deleções, inserções ou atualizações, criação de paines para melhor traduzir as informações dos dados coletados conforme Figura 40.

Figura 40 – Painel Dados Ingeridos Tempo Real.



Fonte: Próprio autor.

Confome o arquivo contendo os dados é atualizado ou alimentado com novas informações o agente realiza a coleta e o encaminhamento para o Elasticsearch, isso facilita muito o acompanhamento em tempo real das informações que estão sendo ingeridas e a tabela abaixo exemplifica esta atualização.

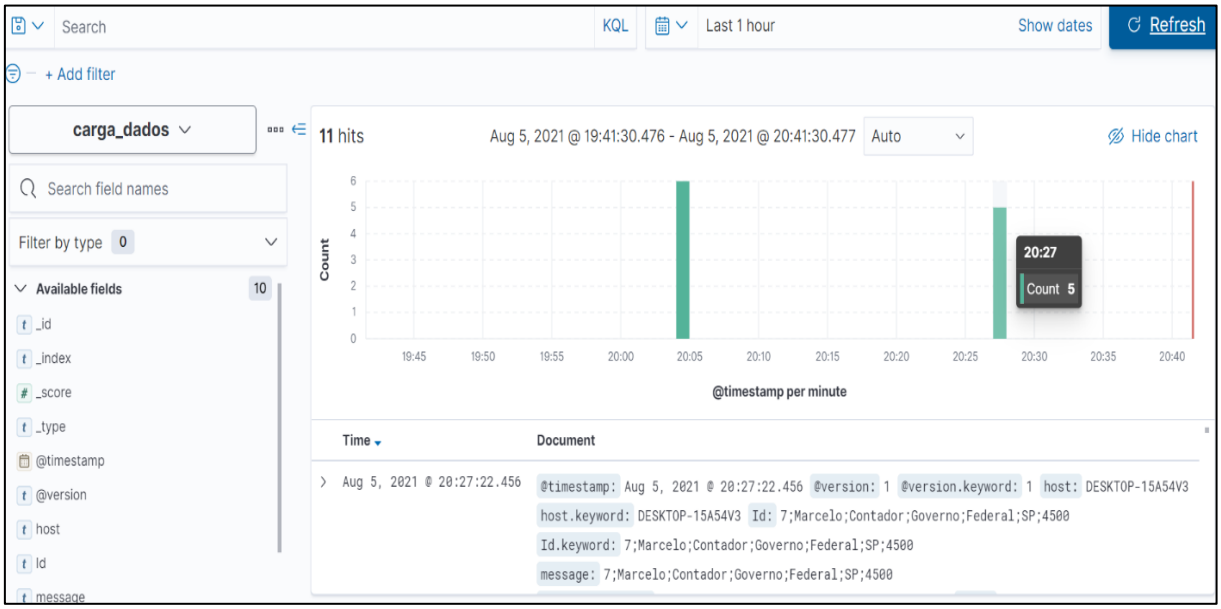
Na primeira execução do agente a parte na cor (verde) foi a primeira coleta de dados e após a planilha ter sido alimentada com novas informações que são as referênciadas na cor (amarelo) o agente de ingestão Logstash identificou as novas informações e já as indexou realizando o envio para o Elasticsearch conforme Tabela (7) e Figura (41):

Tabela 7 – Novos Dados para Ingestão

Id	Nome	Vinculo	Orgao	OrgaoSuperior	Estado	Salario
1	Carlos	Nenhum	Proprio	Estadual	MG	1000
2	Fatima	Contador	Governo	Federal	SP	3000
3	Marcos	Empresar	Governo	Federal	ES	5050
4	Marcia	Laborator	Prefeitura	Municipio	SP	6000
5	Alice	Medica	Particular	Estadual	SP	7000
6	Marcela	Nenhum	Proprio	Estadual	MG	5000
7	Marcelo	Contador	Governo	Federal	SP	4500
8	Cassio	Empresar	Governo	Federal	ES	9700
9	Cassia	Laborator	Prefeitura	Municipio	SP	12000
10	Cassiano	Medica	Particular	Estadual	SP	6700

Fonte: Próprio autor.

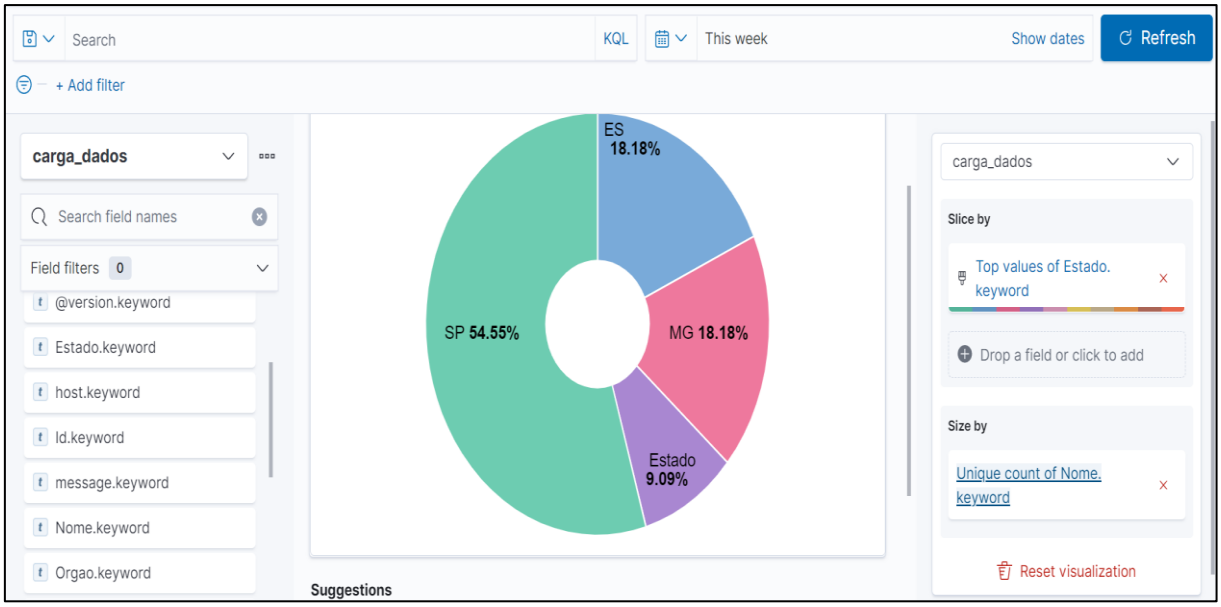
Figura 41 – Painel de dados inseridos em tempo real.



Fonte: Próprio autor.

Após a ingestão das informações a ferramentas de paines do ElasticSearch proporcionar a analise de diversar maneiras, abaixo um exemplo de analise para saber o percentual de funcionários divididos por seus estados conforme Figuras 42 e 43.

Figura 42 - Painel com percentual funcionários x estado no Kibana.



Fonte: Próprio autor.

Figura 43 - Painel salários x nomes no Kibana.

Fonte: Próprio autor.