

**FACULDADE DOCTUM DE CARATINGA**

**ANDRÉ LUIZ GOMES DA SILVA**

**USO DO PROCESSO DE EXTRAÇÃO, TRANSFORMAÇÃO E CARGA (ETL)  
VOLTADO A VISÃO DE ATRIBUTOS QUALIFICANTES,  
ASSOCIADOS AO AUXÍLIO À QUALIFICAÇÃO DO PROFISSIONAL DE TI**

**CARATINGA  
2019**

**ANDRÉ LUIZ GOMES DA SILVA  
FACULDADE DOCTUM DE CARATINGA**

**PROPOSTA DE USO DO PROCESSO DE EXTRAÇÃO, TRANSFORMAÇÃO E  
CARGA (ETL) VOLTADO A VISÃO DE ATRIBUTOS QUALIFICANTES,  
ASSOCIADOS AO AUXÍLIO À QUALIFICAÇÃO DO PROFISSIONAL DE TI**

Trabalho de Conclusão de Curso  
apresentado ao Curso de Ciência da  
Computação da Faculdade Doctum de  
Caratinga, como requisito parcial à obtenção  
do título de Bacharel em Ciência da  
Computação

Área de Concentração: Business  
Intelligence

Orientador: Prof. Maicon **Vinicius**  
Ribeiro

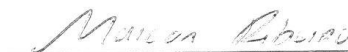
**CARATINGA  
2019**

**TERMO DE APROVAÇÃO**

O Trabalho de Conclusão de Curso intitulado: USO DO PROCESSO DE EXTRAÇÃO, TRANSFORMAÇÃO E CARGA (ETL) VOLTADO A VISÃO DE ATRIBUTOS QUALIFICANTES, ASSOCIADOS AO AUXÍLIO À QUALIFICAÇÃO DO PROFISSIONAL DE TI, elaborado pelo(s) aluno(s) ANDRÉ LUIZ GOMES DA SILVA foi aprovado por todos os membros da Banca Examinadora e aceito pelo curso de CIÊNCIA DA COMPUTAÇÃO das FACULDADES DOCTUM DE CARATINGA, como requisito parcial da obtenção do título de

**BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.**

Caratinga 04/12/2019



MAICON RIBEIRO  
Prof. Orientador



FABRÍCIA PIRES  
Prof. Avaliador 1



RICARDO BOTELHO  
Prof. Examinador 2

## DEDICATÓRIA

Dedicado a Thalita de Paula Gomes da Silva, para sempre comigo, presente na minha história, parte da minha vida, viva na minha memória, dentro da minha alma e do meu coração.

## AGRADECIMENTOS

Agradeço primeiramente a Deus por ter me dado saúde e forças pra superar todas as dificuldades e obstáculos.

A minha mãe Maria Do Rosário e meu padrasto José Geraldo, por terem me dado a oportunidade de estar aqui, que sempre fizeram tudo que podiam para que eu pudesse ter as melhores condições para estudar, me garantindo tudo que precisei, e às vezes sem perceber, com pequenas ações e palavras me motivando e dando forças, por todas as noites que ela deixou minha comida preparada, minha cama arrumada com carinho pois eu chegaria de madrugada e cansado. Por continuamente reconhecer minha luta e insistir que o melhor caminho era o que eu estava tomando. Obrigado Mãe pelo amor e cuidado.

Agradeço também aos professores, Elias Gonçalves e Fabricia Pires pois sempre que precisei, estavam dispostos a me ajudar e esclarecer minhas dúvidas. Quero agradecer especialmente ao meu orientador Maicon Ribeiro não apenas pelo conhecimento racional, mas a manifestação do caráter e afetividade, por ter me aconselhado e me guiado e por ser o exemplo de profissional e pessoa a qual me inspiro e desejo me tornar.

Aos meus amigos e companheiros Cleiton e Jordan, que fizeram parte da minha formação e que continuarão presentes na minha vida, *gracias mis amigos!*

Por fim, a mais importante, minha Pitanga. Obrigado por ter tido paciência e compreensão, por ter me mostrado suas tantas virtudes nos momentos de maior pressão, por ter me dado paz e serenidade quando o mundo estava desabando a minha volta. Você foi, e sempre será, a parte mais bela de mim. O que aprendi com você talvez nunca tenha nem ao menos percebido. Obrigado pelo amor, carinho, cuidado e atenção. Desde que apareceu na minha vida você se tornou essencial. Thaís Maryane, amo você.

A todos que fazem parte de minha vida, do meu meio, dos meus dias.

Meus sinceros agradecimentos!

*“Carry on, carry on  
As if nothing really matters”*

Freddie Mercury

## **ABREVIATURAS E SIGLAS**

ETL – Extract, Transform and Load (Extração, transformação e carga)

DW – Data Warehouse (Armazém de dados)

DM – Data Mart (Repositório de dados)

HTML – Hypertext Markup Language (Linguagem de Marcação de Hipertexto)

DOM – Document Object Model (Modelo de Objeto de Documento)

URL – Uniform Resource Locator (Localizador Uniforme de Recursos)

XML – Extensible Markup Language (Linguagem de Marcação Extensível)

SGBD - Sistemas de Gestão de Base de Dados

SQL – Structured Query Language (Linguagem de consulta estruturada)

PDI – Pentaho Data Integration

BI – Business Intelligence (Inteligência de negócios)

IDE – Integrated Development Environment (Ambiente de Des. Integrado)

HTTP – Hypertext Transfer Protocol (Protocolo de Transferência de Hipertexto)

## Lista de Figuras

Figura 1: Processo de Extração, transformação e carga .....	18
Figura 2: Web Scraping.....	22
Figura 3: Imagem simplificada de um sistema de banco de dados .....	26
Figura 4: Tela SGBD Workbench 8.0 CE. ....	27
Figura 5: Código de criação de tabela em SQL.....	28
Figura 6: Tela Inicial Pentaho Data Integration 8.3 (Kettle).....	29
Figura 7: Data Mart Infraestrutura .....	33
Figura 8: Esquema modelo Snow Flake (Floco de neve).....	35
Figura 9: Esquema modelo Star Schema (Esquema Estrela) .....	35
Figura 10: Composição elementar de uma tabela fato.....	37
Figura 11: Dashboard Criada com <i>Power BI</i> .....	39
Figura 12: Caminho dos dados .....	40
Figura 13: Estruturação de blocos em <i>empregos.com.br</i> .....	45
Figura 14: Estruturação de blocos em <i>empregos.profissionaisti.com.br</i> .....	46
Figura 15: Estrutura site <i>catho.com</i> .....	47
Figura 16: Padrão de disposição dos links .....	48
Figura 17: Padrão de nomenclatura de urls .....	49
Figura 18: Esquema Scraping em <i>empregos.com.br</i> .....	53
Figura 19: Esquema Scraping Catho .....	54
Figura 20: Esquema Scraping Profissionais TI.....	55
Figura 21: Estrutura das tabelas no banco de dados .....	58
Figura 22: (Data Integration - Kettle) Transformações em <i>empregos.com</i> .....	60
Figura 23: (Data Integration - Kettle) Transformações em <i>catho.com</i> .....	61
Figura 24: (Data Integration - Kettle) Transformações em <i>empregos.profissionaisti.com</i> .....	61
Figura 25: Estrutura das tabelas após transformações .....	62
Figura 26: (Catho) exemplo de atributos para reconhecimento .....	63
Figura 27: Exemplo de criação de <i>tokens</i> .....	66
Figura 28: Derivação dos campos pai em <i>empregos.com.br</i> .....	68
Figura 29: Derivação dos campos pai em <i>empregos.profissionaisti.com.br</i> .....	69
Figura 30: Derivação dos campos pai em <i>catho.com</i> .....	70
Figura 31: Modelagem <i>Snow Flake</i> .....	72



Figura 32: Definição das dimensões .....	73
Figura 33: Fato vaga .....	74
Figura 34: Criação do fato vaga. ....	75
Figura 35: Relacionamento das tabelas no Data Mart .....	76
Figura 36: Conexão Power Bi com base de dado .....	77
Figura 37: Top 10 Ferramentas e quantidade de ocorrências.....	83
Figura 38: Número de benefícios reconhecidos .....	84
Figura 39: Top 10 perfil e quantidades reconhecidas.....	85
Figura 40: Carga do Data Mart no Power Bi.....	87
Figura 41: Top 10 ferramentas solicitadas para o cargo desenvolvedor .....	88

## **Lista de Quadros**

Quadro 1: Padrão observado de nomenclatura de elementos .....	57
Quadro 2: Quantidade de cargos categorizados .....	65

## Lista de Gráficos

Gráfico 1: Número de vagas relacionadas à tecnologia .....	43
Gráfico 2: Porcentagem de vagas relacionadas a TI .....	44
Gráfico 3: Porcentagem de classificação por área .....	82
Gráfico 4: Top 10 ferramentas solicitadas para o cargo desenvolvedor.....	88

## RESUMO

Com a alta demanda de virtualização de produtos e serviços, atualmente um dos meios mais solicitados de se procurar emprego é online. Essa modalidade permitiu uma maior acessibilidade das vagas presentes no mercado, entretanto, não é tão fácil acompanhar o que as empresas procuram, no âmbito das habilidades e competências exigidas para a qualificação dessas oportunidades de trabalho. E quem sofre diretamente com isso são os recém-formados e profissionais da área de tecnologia da informação, que em meio a uma enormidade de possibilidades, há uma desorientação sobre os atributos necessários para sua qualificação. Pensando nisso, o presente trabalho, voltado para o campo de *Business Intelligence*, decorre sobre o uso das ferramentas e técnicas de ETL (Extração, transformação e carga), para a extração, tratamento, classificação e reconhecimento de atributos, e carga dos dados disponíveis em plataformas que ofertam vagas online.

**Palavras-Chave:** Business Intelligence. ETL. Qualificação.

## **ABSTRACT**

With a high demand for product and service virtualization, it is currently one of the most sought-after ways to look for jobs online. This modality has allowed for greater accessibility of vacancies present in the market, however, it is not so easy to follow or that companies select, within the scope of skills and competences required for an assessment of these job opportunities. And who suffer directly from this are the recent graduates and professionals in the area of information technology, which in the midst of a multitude of possibilities, there is a disorientation about the attributes customized for their qualification. With this in mind, the present work, focused on the field of Business Intelligence, deals with the use of ETL (Extraction, Transformation and Load) tools and techniques, for the extraction, treatment, classification and recognition of resources, and the loading of data available in platforms that offer online vacancies.

**Keywords:** Business Intelligence. ETL. Qualification.

## Sumário

### **1 INTRODUÇÃO16**

### **2 REFERENCIAL TEÓRICO18**

#### **2.1 ETL – Extract, Transform e Load (Extração, transformação e carga)18**

2.1.1 Extract (Extração)20

2.1.2 Transform (Transformação)20

2.1.3 Load (Carga)21

#### **2.2 Raspagem de dados na web21**

2.2.1 Web Crawler23

2.2.2 Web Scraping24

2.2.3 BeautifulSoup25

2.2.4 Web Scraping/Crawler é legal?25

#### **2.3 Banco de Dados26**

2.3.1 Sistema gerenciador de banco de dados (SGBD)27

2.3.2 SQL – Structured Query Language28

#### **2.4 Pentaho Data Integration (PDI)29**

#### **2.5 Data Warehouse30**

2.5.1 Data Mart32

2.5.2 Modelo Dimensional33

#### **2.6 Power BI38**

### **3 METODOLOGIA40**

#### **3.1 Levantamento dos dados41**

3.1.1 Análise das fontes e da legalidade da extração dos dados41

3.1.2 Caracterização dos dados44

#### **3.2 Web Crawler48**

#### **3.3 Web Scraping51**

3.3.1 Scraping Plataforma 152

3.3.2 Scraping Plataforma 253

3.3.3 Scraping Plataforma 355

3.3.4 Extração do conteúdo56

### **3.4 Pentaho Data Integration – Transformações58**

### **3.5 Classificação e reconhecimento de atributos63**

3.5.1 Classificações e reconhecimento dos Cargos e áreas65

3.5.2 Reconhecimento Atribuições dos Cargos, ferramentas, certificações, benefícios e perfil.67

3.5.3 Reconhecimento Jornada, regime, cidade e estado.70

3.5.4 Atributos Salário e número de vagas71

### **3.6 Implementação do Data Mart71**

3.6.1 Modelo Dimensional71

3.6.2 Escolha das dimensões72

3.6.3 Criação da tabela fato73

3.6.4 Implementação Física74

### **3.7 Power BI76**

## **4 APRESENTAÇÃO DOS RESULTADOS78**

**4.1 Crawler/Scraping78**

**4.2 Pentaho Data Integration – Kettle80**

**4.3 Classificação e reconhecimento de Atributos81**

**4.4 Implementação do Data Mart e carga86**

**4.5 Geração dos Relatórios86**

## **5 CONSIDERAÇÕES FINAIS90**

## **6 TRABALHOS FUTUROS92**

## **7 REFERÊNCIAS93**

## 1 INTRODUÇÃO

Graças ao desenvolvimento da tecnologia e da internet, hoje uma das formas mais comuns e eficientes de se procurar emprego é online. Com a alta demanda, esse mercado cresceu e se modernizou, e hoje em dia existem inúmeros sites especializados que divulgam vagas de emprego em diversos setores, e também em nichos específicos como a TI (Tecnologia da Informação).

O mercado de TI está em constante crescimento e evolução, diante disso é notável a falta de profissionais qualificados para preenchimento das vagas ofertadas, o perfil esperado do profissional da tecnologia do século XXI, é de um profissional com múltiplas habilidades técnicas, habilidades comportamentais específicas, e que apresente uma inovação contínua na velocidade em que produtos e serviços são lançados.

Mas quais competências e ferramentas, e que tipo de conhecimento é solicitado aos profissionais nas vagas divulgadas pelas empresas? O que de fato se precisa saber para se inserir no mercado de trabalho, ou conseguir uma boa colocação?

As respostas aos questionamentos supracitados são possíveis pois com o crescimento exponencial dos dados gerados pela web e o surgimento de novas tecnologias capazes de manipular grandes volumes de dados, tem permitido explorar e extrair informações valiosas num universo de dados disponíveis que são gerados nos mais diversos meios de comunicação, sobretudo nas páginas da internet.

Essa informação gera ganhos de competitividade e eficiência nas mais diversas áreas, o valor intrínseco das informações retiradas desse “mar de dados” é diretamente proporcional a maneira como são tratados. Com o Business Intelligence e suas ferramentas de ETL (Extração, transformação e carga), esses dados são transformados em informação e através da descoberta, a informação é transformada em conhecimento.

Embora o conceito de Business Intelligence seja nativo a uma necessidade comercial, seu uso não se restringe a essa área. Muito pelo contrário, o Business Intelligence e as ferramentas de ETL podem ser usados em qualquer ambiente em que seja necessária uma tomada de decisão informada, desde que existam tecnologias e dados relevantes, organizados e confiáveis. Portanto, é possível desenvolver novos trabalhos investigando a eficácia do uso de ferramentas de ETL em outros ambientes não comerciais.



No decorrer do trabalho, será visto a exploração de ferramentas de ETL, passando por cada uma de suas fases, com o intento de extrair informações relevantes de plataformas que ofertam vagas online, promovendo o tratamento e “lapidação” desses dados, a fim de realizar o reconhecimento e classificação de atributos e competências inerentes a cada vaga. Conjuntamente foi abordado técnicas de modelação da informação para criação de *Data Warehouses*, a fim de prover o acesso aos dados de forma intuitiva e com alta performance.

## 2 REFERENCIAL TEÓRICO

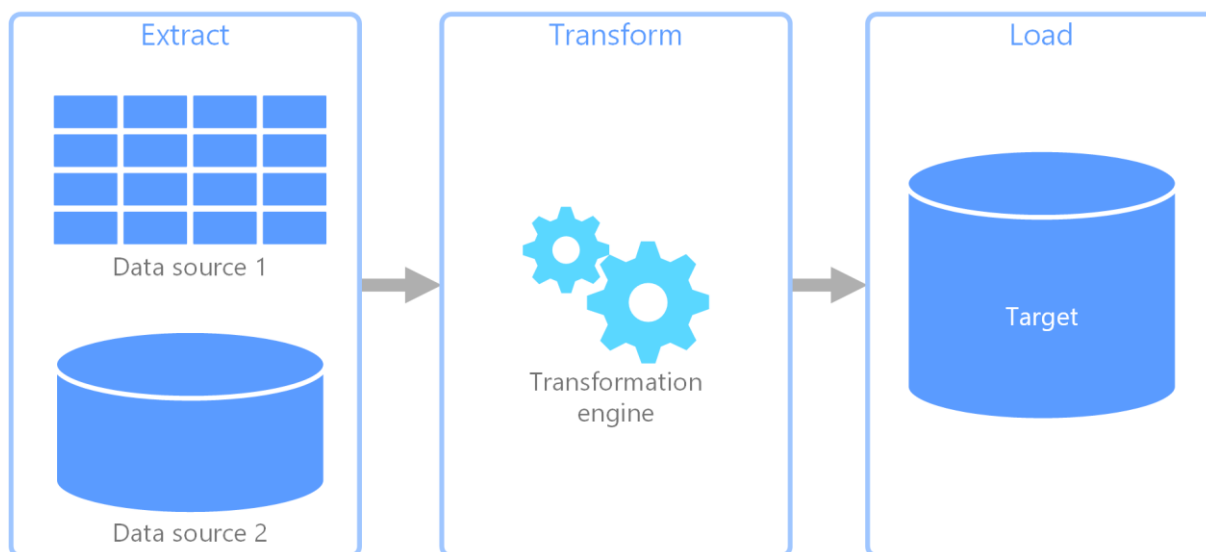
Nesta seção será descrito o referencial teórico deste trabalho, o qual apresenta as principais definições e características dos temas abordados. Para tanto, foi realizado um estudo de ferramentas e técnicas que serão aplicadas no trabalho, permitindo assim uma melhor compreensão de conceitos relevantes para o entendimento e realização deste estudo

### 1.1 ETL – Extract, Transform e Load (Extração, transformação e carga)

O processo ETL extrai os dados da origem sistemas, transforma os dados de acordo com as regras de negócios e carrega os resultados no diretório *Data Warehouse* de destino.

O sistema Extract-Transform-Load (ETL) é a base dos Data Warehouses. Um sistema ETL projetado adequadamente extrai dados de diversas fontes, aplica padrões de qualidade e consistência dos dados, integra dados para que fontes separadas possam ser usadas juntas e, finalmente, fornece dados em um formato pronto para apresentação, para que os desenvolvedores de aplicativos possam criar aplicativos e usuários finais possam tomar decisões. (KIMBALL; CASERTA, 2004).

Figura 1: Processo de Extração, transformação e carga



Fonte: Microsoft (2019)

Na Figura 1, é exemplificado o processo de ETL, onde os dados são extraídos de fontes de diversos formatos, após e feito um tratamento desses dados conforme as regras de negócios e por fim o carregamento desses dados.

Conforme observa-se na Figura 1, em concordância com Bergamaschi (2011) descreve algumas atividades detalhadas de ETL:

- a) A observação e identificação de informações relevantes na fonte de dados;
- b) A extração dessa informação estruturada em formação;
- c) Integração de várias fontes de diversos formatos em um formato único e comum;
- d) A limpeza do conjunto de dados resultante com base em banco de dados e regras de negócios;
- e) A propagação dos dados para o *Data Warehouse* e/ou para *Datamarts*.

O sistema ETL cria ou quebra o *Data Warehouse*. Embora a construção de um sistema de ETL ser uma atividade de bastidores, que não é muito visível para os usuários finais, de acordo com KIMBALL e CASERTA (2004) pode consumir facilmente 70% dos recursos necessários para a implementação e manutenção de um *Data Warehouse* típico.

ETL é um assunto simples e complicado. Quase todo mundo entende a missão básica do sistema ETL: obter dados da fonte e carrega-los no armazém de dados. E a maioria dos observadores está cada vez mais apreciando a necessidade de limpar e transformar dados ao longo do caminho. (INMON 2002).

O sistema ETL agrega valor significativo aos dados. É muito mais do que um encanamento para obter dados dos sistemas de origem. Especificamente, o sistema ETL remove erros e corrige dados ausentes, fornece medidas documentadas de confiança nos dados, captura o fluxo de dados transacionais para proteção, ajusta dados de várias fontes para serem usados juntos e estrutura os dados para serem utilizados pelas ferramentas do usuário final. Portanto, se faz necessário uma abordagem descrevendo cada fase.

## 1.1 Extract (Extração)

Retirar os dados de uma variedade de sistemas de origem corretamente é geralmente o aspecto mais desafiador do ETL, pois é essa etapa que prepara o terreno para como serão os processos subsequentes.

Em geral, o objetivo da fase de extração é converter os dados em um único formato para o processo de transformação. O sistema também pode usar dados de diferentes organizações ou de diferentes formatos. Fonte de dados em formatos comuns, bancos de dados relacionais e arquivos simples, mas pode incluir banco de dados não relacional e estruturas como gerenciamento de informações ou outras estruturas de dados, como Método de Acesso ao Armazenamento Virtual ou Método de acesso sequencial indexado ou mesmo buscando de fontes externas, como através de *Spider-web* ou raspagem de tela. (YULIANTO, 2019)

O objetivo do processo de extração de dados é coletar dados úteis de várias fontes de dados heterogêneas. A complexidade da extração de dados é geralmente determinada pela complexidade da fonte de dados (SUN; LAN, 2012).

## 1.2 Transform (Transformação)

A transformação de dados, geralmente envolve diversas operações, como filtragem, classificação, agregação, junção de dados, limpeza de dados, eliminação de duplicação e validação de dados.

Transformação, limpeza e conformidade são as principais etapas de um sistema ETL. As outras etapas de extração e entrega são obviamente necessárias, mas elas apenas movem e reformatam dados. Limpar e tratar realmente altera os dados e fornece orientação se os dados podem ser usados para os propósitos pretendidos. As etapas de limpeza e conformidade geram metadados potentes. Olhando de volta para as fontes originais, esses metadados são um diagnóstico do que é errado nos sistemas de origem. (KIMBALL; CASERTA, 2004).

Também chamada de tratamento ou limpeza de dados, a etapa de transformação trata da detecção e remoção de erros e inconsistências de dados para melhorar a qualidade dos dados. As abordagens de limpeza de dados envolvem três etapas: análise de dados, refinamento de dados e verificação de dados (YULIANTO, 2019).

### 1.3 Load (Carga)

O último processo no ETL, que é a fase de carregamento, é responsável por carregar os dados processados pelas duas fases anteriores no DW. De acordo com SUN (2012), existem dois métodos principais para carga de dados, *refreshing* e *updating*. O método *refreshing* é usado principalmente para carregar os dados no banco de dados durante a criação de um DW, enquanto o método *updating* é usado para inserção de novos dados no DW.

Carga é a etapa final do ETL. Nesta etapa, dados limpos e conformes são gravados nas estruturas dimensionais que são realmente acessadas pelos usuários finais e sistemas de aplicativos. Nos *Data warehouses* menores consistem em um único espaço de tabela para acesso do usuário final, as tabelas dimensionais não são gravadas neste espaço de tabela, mas em todos os grandes *Data warehouses*, variando de vários espaços de tabela a amplamente distribuídos e autônomas redes de *Data marts*. (KIMBALL; CASERTA, 2004).

Deste modo, temos que a etapa de carga ocorre na sequência da transformação. Após a efetuação dos tratamentos necessários nos dados, a carga no *Data Warehouse* é iniciada. Essa etapa se resume na persistência dos dados na base de dados.

### 1.2 Raspagem de dados na web

A raspagem de dados da Web é uma técnica de *software* para extrair informações de *sites*, seu funcionamento consiste principalmente na transformação de dados não estruturados (formato *HTML*) da *web* em dados estruturados (banco de dados ou planilha eletrônica). (RAY, 2015).

No presente trabalho, a raspagem de dados fara parte da primeira etapa do ETL, a etapa de extração, PICHILIANI (2011), afirma que:

A técnica de raspagem de dados pode ser aplicada nas mais diversas aplicações, em especial, em situações em que as ferramentas de ETL não desempenham a contento a tarefa, devido a peculiaridade, à variação de formato e até a forma de disponibilizar esses dados na internet.

Algumas técnicas mais comuns de coleta de dados atualmente são: Cópia Humana; “*Grepping*” de texto; Programação HTTP; Análise Dom e Scraping/Crawler, cada uma das técnicas são abordadas a seguir:

Cópia humana, ou cópia manual, podemos elucidar como “copiar e colar”, menos indicada e mais custosa, essa técnica é usada quando a página a qual os dados serão coletados possui um nível alto de proteção contra os softwares de coleta.

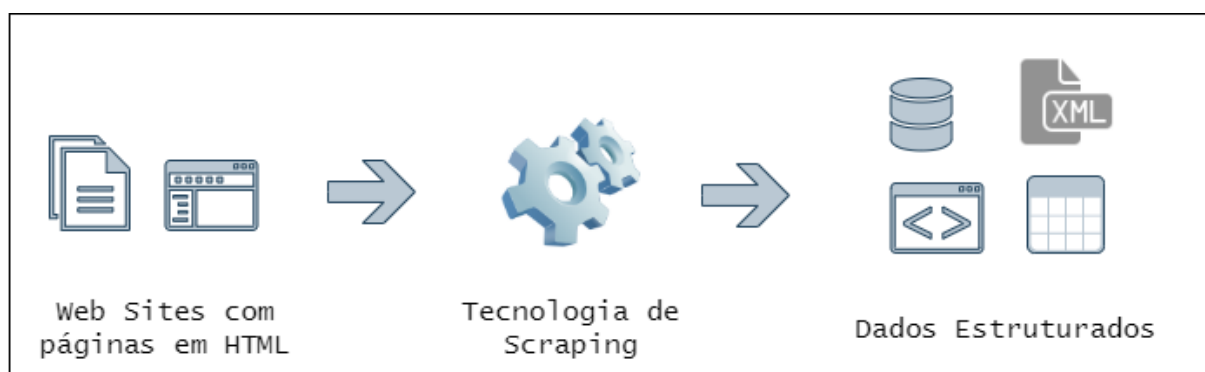
"*Grepping*" de texto e correspondência de expressões regulares: que utiliza o comando *grep* do *UNIX*, ou realiza correspondências de expressões regulares que podem ser em diversas linguagens de programação, como Python e Perl. (BULL, 2002)

Também muito visualizado na comunidade, raspagem de dados utilizando programação *HTTP*: Páginas *web* dinâmicas e estáticas podem ser acessadas através de solicitações *HTTP* para o servidor *web* remoto usando programação de socket.

Os autores JENSEN, MADSEN e MOLLER (2011), abordam a raspagem de dados utilizando Análise DOM: Incorporando um navegador, como o Internet Explorer ou o Mozilla, essa técnica pode recuperar o conteúdo dinâmico gerado pelos scripts do lado do cliente. Também é possível analisar páginas *web* em uma árvore DOM.

Existem também inúmeros *softwares* de coleta, essas ferramentas conhecidas como *Scraping*, *Crawler*, *Robot*, podem ser usados para personalizar soluções de coleta de dados *web* (Figura 2). Algumas ferramentas de alcance mundial já possuem nativamente funções de extração de dados na *web*, pode-se citar por exemplo os *softwares* da Microsoft: Excel, com o suplemento *Power Query*, e o *Power Bi*, que utiliza a ferramenta em conjunto com inteligência artificial, e pode reconhecer automaticamente a estrutura de dados de uma página ou fazer a extração através de padrões observados e descrito pelo usuário.

Figura 2: Web Scraping



Fonte: O autor

Na Figura 2, é exemplificado o processo de Scraping, que extrai dados desestruturados de diversas fontes e faz a persistência desses dados em banco de dados, planilhas eletrônicas e arquivos XML de maneira estruturada.

Muito utilizada para criação de algoritmos de *Scraping* a linguagem Python, contém várias bibliotecas para raspagem de dados da Internet, tais como *BeautifulSoup*, *Django*, *Selenium*, *Requests*, *Mechanize* e *Urllib2*.

Cada ferramenta possui vantagens e desvantagens relacionadas a facilidade de aprendizagem, documentação, desempenho e possibilidade de sobrecarga dos sites. Como o Python é uma linguagem aberta, foi importante considerar a estabilidade dessas ferramentas no que diz respeito à atualidade da documentação e ausência de erros de execução. Geralmente isto é garantido quando há uma grande comunidade de usuários regulares da linguagem, reportando os erros de execução e trabalhando cooperativamente para encontrar novas soluções. Este é o caso da biblioteca BeautifulSoup 4 no Python.

#### 1.4 Web Crawler

Também conhecido como *Spider* ou *Bot* (robô), *Web Crawler*, é uma forma eficiente de automatizar a pesquisa em páginas da internet (LIU, 2011), indexando páginas e armazenando-as na base de dados, obtendo informações vista como importantes. Podemos citar como exemplos de rastreadores do tipo mencionado *JSpider*, *Crawljax* e também o *Googlebot* o *Search Engine* do Google.

JARMUL e LAWSON (2017), afirma que os *Crawlers* geralmente são construídos de uma maneira genérica, segmentando sites de uma série de domínios de nível superior, capturando pequenas e genéricas informações específicas e links para outras páginas.

Pode-se entender basicamente, que *Crawler* é um *software* desenvolvido que percorrem a Web, visitando um conjunto inicial de urls, em busca de outras urls nestas páginas. Ao encontrá-los, visita-os de maneira recursiva, até que uma condição de parada seja satisfeita. Os motores de busca análogos aos citados acima, o utilizam para criar uma cópia das páginas visitadas para um pós-processamento e indexação, conseguindo oferecer dessa maneira uma busca mais rápida.

Como nem sempre é interessante percorrer toda a Web, os *Crawlers* podem ser configurados para atuar em um domínio específico, ou seja, para navegar apenas

em sites de interesse do usuário. Os *Crawlers* que são configurados para navegar desta forma são chamados de *Focused Crawler* ou *Crawlers* focados (LIU, 2011).

Geralmente a ferramenta *Web Crawler* é usada em conjunto com *Web Scraping*, coletando os links e passando para o *Web Scraping* realizar a “raspagem” desses dados. No decorrer do trabalho será utilizado um *Focused Crawler* extraíndo urls específicas de paginação de sites e passando para o *Scraping* realizar a raspagem da página, extraíndo seu conteúdo.

## 1.5 Web Scraping

Ainda que os navegadores sejam úteis para a execução de *JavaScript*, exibindo imagens e organizando objetos em um formato mais legível para o usuário (entre outras coisas), *Web Scraping* é excelente para coletar e processar enormes quantidades de dados, em vez de visualizar uma página de cada vez através do navegador, você pode visualizar milhares ou até milhões de páginas de uma só vez.

“Se você pensa que a única maneira de acessar a Internet é através de um navegador, você está perdendo uma enorme variedade de possibilidades” (MITCHELL, 2015).

*Web Scraping* é quase tão antigo, quanto a própria internet. Embora não seja um termo novo, nos últimos anos, essa prática vem sendo relacionada mais como raspagem de dados na web, mineração de dados, coleta na web ou termos similares. Hoje o consenso geral parece favorecer a *Web Scraping* e raspagem de dados na web (MITCHELL, 2015).

Para DALE (2016):

Web Scraping, ou raspagem da Web, é o processo de aquisição automatizada de dados não estruturados de sites da Internet, seguido do armazenamento estruturado dos mesmos, para posterior análise. A motivação para esta prática é que muitos dados disponíveis na Internet “não foram projetados para ser consumidos programaticamente pela Web”

Na prática, *web Scraping* abrange uma ampla variedade de técnicas de programação e tecnologias voltados a extração e estruturação da dados obtidos em páginas web, além da análise de dados e segurança da informação. Como será visto no decorrer do trabalho.



## 1.6 BeautifulSoup

*BeautifulSoup* 4 é uma biblioteca Python fundamentada na base do mecanismo de análise HTML/XML, usado para extração, análise e edição de informações na árvore DOM de páginas da web, ajudando os desenvolvedores a criar rapidamente um protótipo do sistema e obter dados. Além disso, possui alta flexibilidade entre plataformas.

BeautifulSoup 4 é uma biblioteca Python para extrair dados de arquivos HTML e XML. Ele trabalha com seu analisador favorito para fornecer maneiras idiomáticas de navegar, pesquisar e modificar a árvore de análise. Geralmente, economiza horas ou dias de trabalho para os programadores. (RICHARDSON, 2019)

Os desenvolvedores que trabalham com a biblioteca *BeautifulSoup* 4 podem, de acordo com suas necessidades, instalar mecanismos específicos de análise HTML, como por exemplo, *lxml*, *html5lib*. Tomando o *lxml* como exemplo, os usuários podem inicializar o *BeautifulSoup* 4 com a seguinte chamada: *BeautifulSoup(markup, "lxml")*. Após a inicialização, o *BeautifulSoup* 4 obtém todas as estruturas correspondentes da árvore *DOM* para documentos *HTML*. Em seguida, usando uma série interna de funções de interface, ele visita, obtém e edita os valores ou conjuntos de propriedades dos nós designados da árvore *DOM*. Obtendo por exemplo o conteúdo dentro da *tag* `<b>`.

## 1.7 Web Scraping/Crawler é legal?

Ao utilizar ferramentas de *Crawler/Scraping* deve-se a princípio examinar as declarações de divulgação de cada plataforma, que estão contidos nos termos de uso e política de privacidade. Isso se torna necessário para avançar legalmente e eticamente.

Dessa maneira RODRIGUES (2015) detalha:

A raspagem da Web passa por uma linha tênue entre a coleta de informações e o roubo de informações. A maioria dos sites tem uma declaração de divulgação de direitos autorais que protege legalmente as informações do seu site e trazem informações a respeito da utilização de dados e a extensão em que esta é permitida.

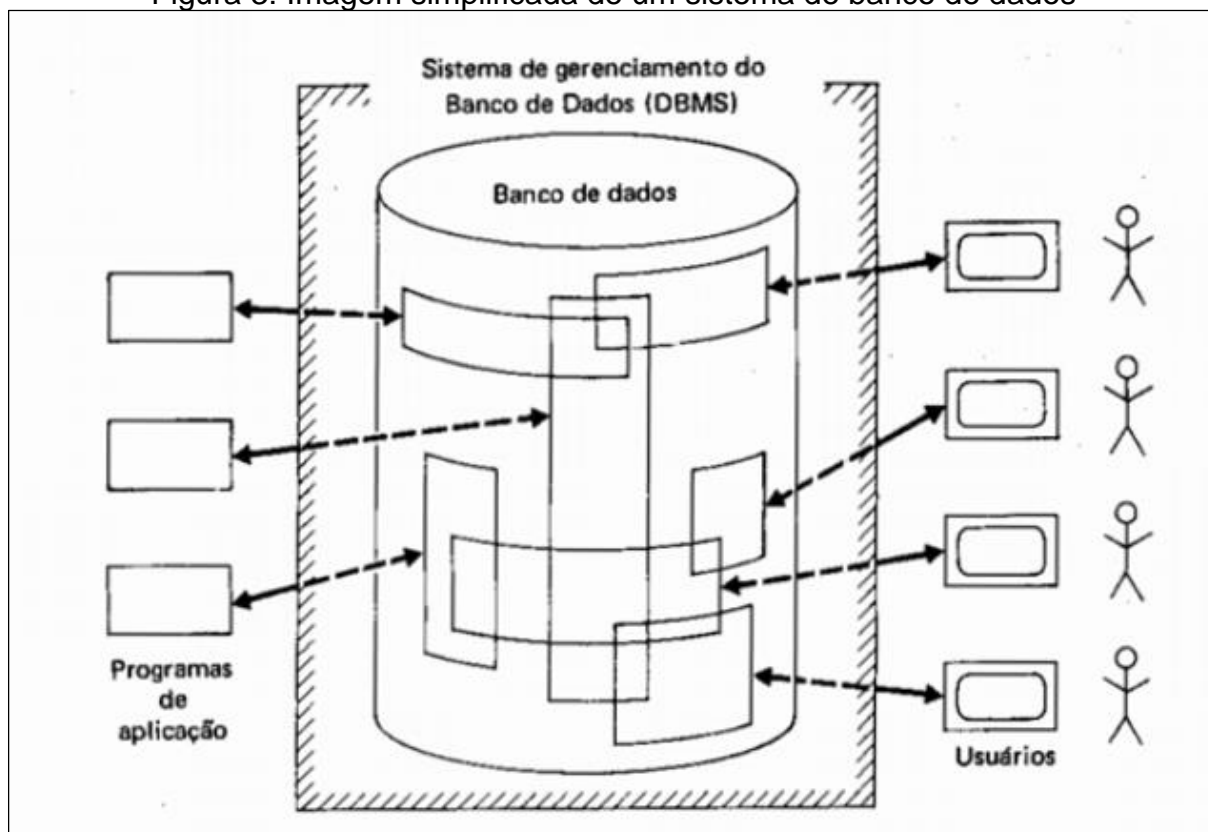
Do mesmo modo LAWSON (2015), declara que o *Scrap/Crawler*, e o que é legalmente permitido na raspagem da Web, ainda estão sendo estabelecidos. Apesar de numerosas decisões nas últimas duas décadas, se os dados raspados estiverem sendo usados para uso pessoal e privado e dentro do uso justo das leis de direitos autorais, geralmente não há problema.

No entanto, se os dados estiverem sendo utilizados para fins comerciais, se a raspagem for agressiva o suficiente para derrubar o site ou se o conteúdo for protegido por direitos autorais, o raspador estará violando os termos de serviço, e para isso existem vários precedentes legais para impedimento.

### 1.3 Banco de Dados

Banco de dados é um sistema de armazenamento de dados baseado em computador, DATE (2004) o descreve como um sistema cujo objetivo global é registrar e manter informação. Esta informação pode ser qualquer uma considerada significativa ao usuário servido pelo sistema.

Figura 3: Imagem simplificada de um sistema de banco de dados

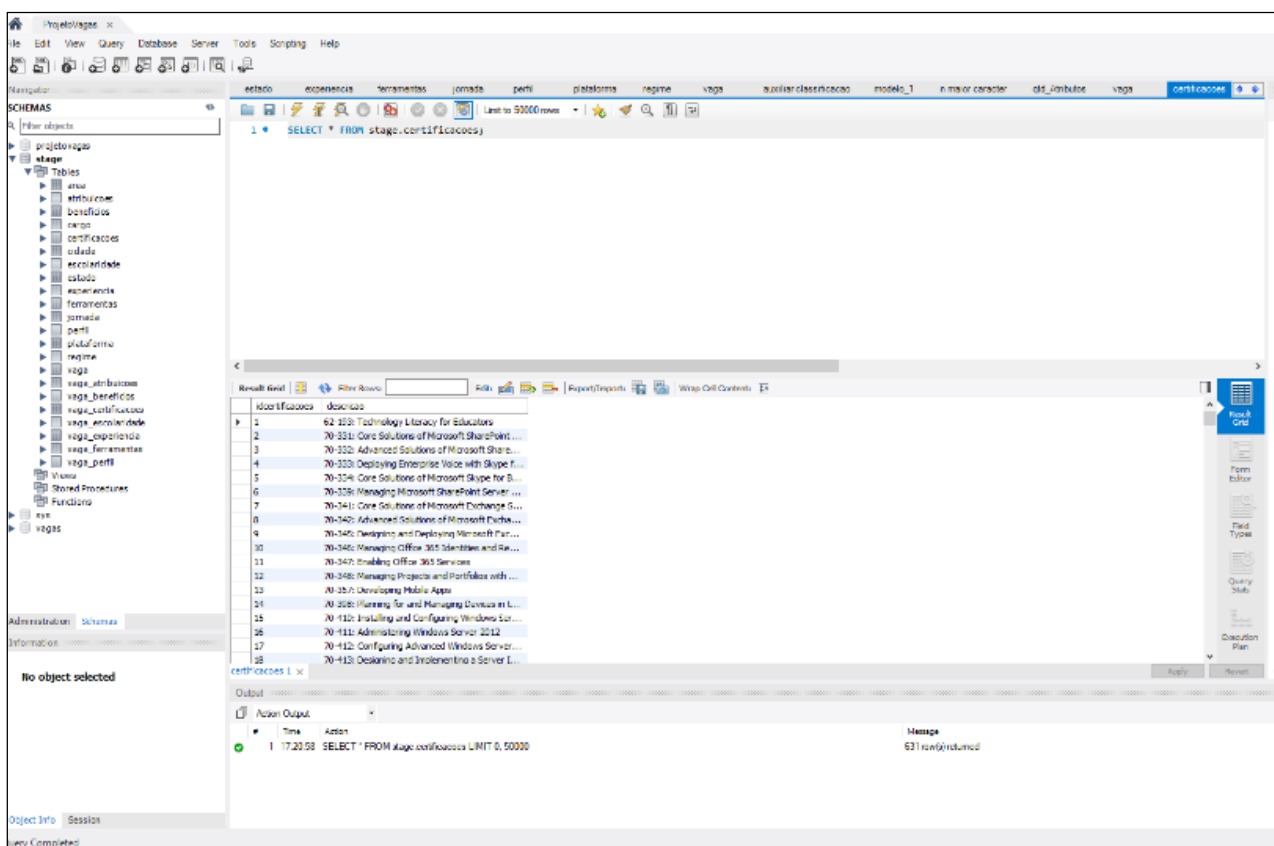


Fonte: DATE (2004).

## 1.8 Sistema gerenciador de banco de dados (SGBD)

Um sistema gerenciador de banco de dados (SGBD) é uma coleção de programas que permite aos usuários criar e manter um banco de dados, ELMASRI e NAVATHE (2006), define SGBD como um sistema de software de propósito geral que facilita os processos de definição, construção, manipulação e compartilhamento de bancos de dados entre vários usuários e aplicações.

Figura 4: Tela SGBD Workbench 8.0 CE.



Fonte: O autor

Na Figura 4, é visto a tela inicial do Workbench, e sua disposição dos elementos presentes.

Dentre tantos outros no mercado, pode se citar como exemplos de Sistemas gerenciador de banco de dados: *Oracle, MysqI, SQL Server, PostgreSQL*.

## 1.9 SQL – Structured Query Language

*Structured Query Language* ou Linguagem Estruturada de Consulta, é a linguagem padrão para manipulação de bancos de dados relacionais (ELMASRI; NAVATHE, 2006) desse modo, pode se dizer que SQL é uma linguagem que permite definir, modificar e consultar dados em bancos de dados baseados em tabelas, criação de relacionamentos, e *triggers* (eventos), são duas dentre outras tantas funcionalidades.

Desta forma, DATE (2004) a retrata da seguinte maneira:

É uma linguagem de fácil aprendizado, poderosa e de alto nível, a qual permite ao usuário especificar, na declaração SQL, como será o resultado da consulta, deixando a cargo do SGBD a tarefa de escolher qual a melhor forma de executar a query do usuário.

Figura 5: Código de criação de tabela em SQL

```
35
36
37  -----
38  -- Table `stage`.`cargo`
39  -----
40  CREATE TABLE IF NOT EXISTS `stage`.`cargo` (
41    `idcargo` INT NOT NULL AUTO_INCREMENT,
42    `descricao` VARCHAR(150) NOT NULL,
43    `area_idarea` INT NOT NULL,
44    PRIMARY KEY (`idcargo`),
45    INDEX `fk_cargo_area_idx` (`area_idarea` ASC) VISIBLE,
46    CONSTRAINT `fk_cargo_area`
47      FOREIGN KEY (`area_idarea`)
48      REFERENCES `stage`.`area` (`idarea`)
49      ON DELETE NO ACTION
50      ON UPDATE NO ACTION)
51  ENGINE = InnoDB;
```

Fonte: O autor

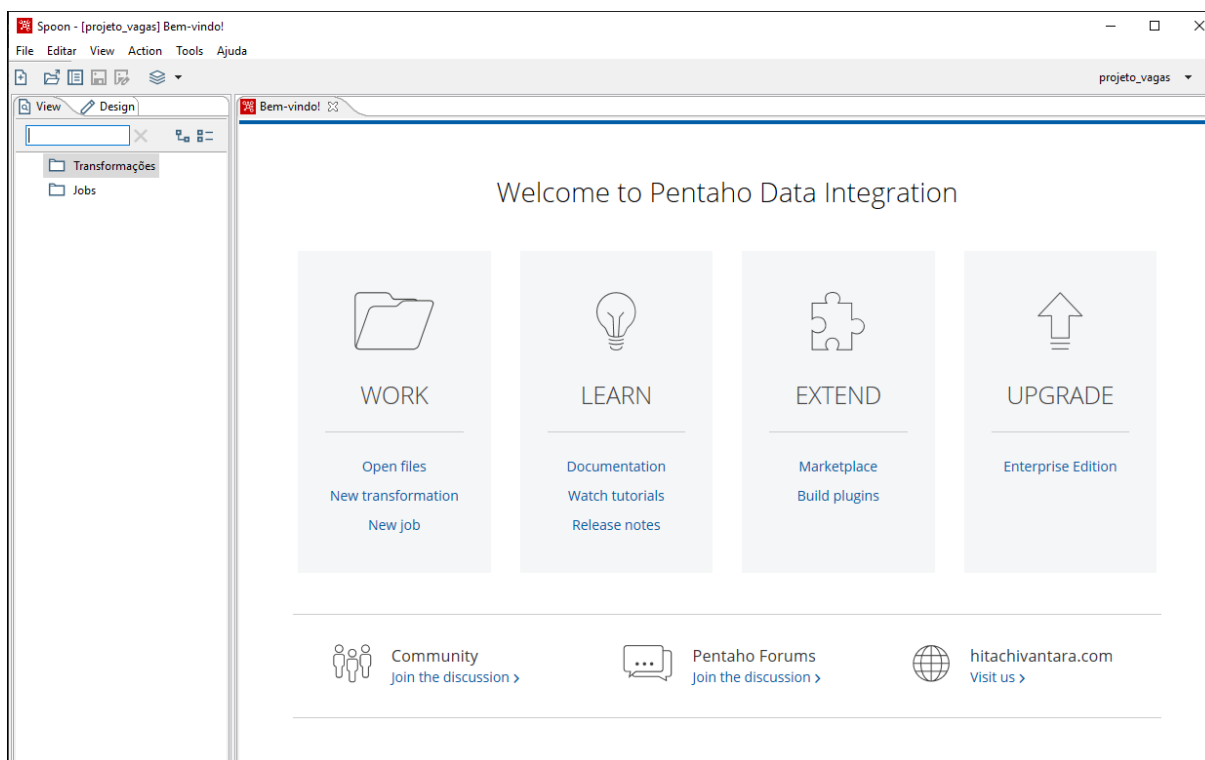
Na Figura 5, é visto um exemplo de uma *query* em SQL, que cria uma tabela, faz a indexação, cria uma chave estrangeira e coloca restrições de exclusão de dados.

## 1.4 Pentaho Data Integration (PDI)

O *Pentaho Data Integration* Figura 6, é uma das soluções ETL mais utilizadas e com maior valor no mercado (TARNAVEANU, 2012). Tornou-se uma ferramenta altamente recomendada, por possuir uma longa história, solidez que inspira credibilidade e por sua robustez. Em 2008 o *software* foi considerado uma das melhores aplicações para inteligência empresarial pela revista InfoWorld (DINELEY, 2008).

O Pentaho Data Integration (também conhecido como Kettle) é um mecanismo, juntamente com um conjunto de ferramentas responsáveis para os processos de *Extracting, transforming, e Loading* mais conhecido como processo ETL. O PDI não serve apenas como uma ferramenta ETL, mas também é usado para outros fins, como migração de dados entre aplicativos ou bancos de dados, exportação de dados de bancos de dados para arquivos, limpeza de dados e muito mais. O PDI possui um design intuitivo, ambiente gráfico com o recurso de arrastar e soltar, e seus recursos de ETL são amplamente poderosos. (ROLDÁN, 2010)

Figura 6: Tela Inicial Pentaho Data Integration 8.3 (Kettle)



Fonte: O autor

O pacote *Pentaho Business Analytics* está disponível em duas versões: *Community Edition* e *Enterprise Edition*. O primeiro é mantido por uma comunidade de voluntários e possui todo o seu código aberto. A segunda, versão comercial do software, tem como mantenedora, que fornece suporte técnico e recursos extras aos clientes a Pentaho Corporation, sediada em Orlando, Flórida, com escritórios em San Francisco, Califórnia e em toda a Europa, e é vendida sobre o modelo de assinatura com três níveis: *Enterprise*, *Premium* e *Standart* que inclui suporte, serviços e aprimoramentos de produtos entre outros benefícios.

## 1.5 Data Warehouse

*Data warehouse* é um banco de dados dedicado ao processamento analítico de dados. Eles são usados como apoio para atividades de tomada de decisão nos mais modernos processos de negócios, quando conjuntos de dados volumosos e complexos precisam ser estudados e analisados.

O DW é definido como um banco de dados, destinado a sistemas de apoio à tomada de decisão e cujos dados foram armazenados em estruturas lógicas dimensionais, possibilitando o seu processamento analítico por ferramentas especiais (BARBIERI, 2001).

A tecnologia para processamento analítico pressupõe que os dados sejam apresentados na forma de *Datamarts*, definido como subconjunto de um DW focalizado em funções ou atividades específicas.

Um dos precursores do assunto, William H. Inmon que ganhou o apelido de “pai do Data Warehouse” sustenta que o Data Warehouse é uma coleção de dados orientados por assunto, integrado, variável com o tempo e não-volátil, que tem por objetivo dar suporte aos processos de tomada de decisão (INMON, 2002).

Ou seja, os dados de um *Data Warehouse* são extraídos dos sistemas transicionais e otimizados para processamento de consulta. Em geral, um *Data Warehouse* requer a consolidação de outros recursos de dados além dos armazenados em banco de dados relacionais, incluindo informações provenientes de planilhas eletrônicas, documentos textuais.

Segundo KIMBALL (2008), a implantação de um Data Warehouse tem alguns objetivos principais, dentre eles: permitir uma combinação e separação dos dados de várias maneiras; fornece acesso aos dados de qualquer nível de informação necessário; fornecer dados consistentes; ajudar na reengenharia de negócios através da qualidade dos dados armazenados no DW.

Diante desses atributos pode-se realizar uma comparação entre as principais características de um *Data Warehouse* com um banco de dados operacional, exemplificando no Quadro 1 que relaciona os atributos de banco de dados operacionais e *Data Warehouses*.

Quadro 1: Comparação entre banco de dados operacionais e data Warehouse

<b>Características</b>	<b>Banco de dados Operacionais</b>	<b>Data Warehouse</b>
Objetivo	Operações diárias do negócio	Analisar o negócio
Uso	Operacional	Informativo
Tipo de processamento	OLTP	OLAP
Unidade de Trabalho	Inclusão, alteração, exclusão	Carga e consulta
Número de Usuários	Milhares	Centenas
Tipo de Usuário	Operadores	Comunidade Gerencial
Interação do usuário	Somente predefinida	Predefinida e ad-hoc
Condição dos dados	Dados operacionais	Dados Analíticos
Volume	Megabytes - gigabytes	Gigabytes – Terabytes
Histórico	60 a 90 dias	5 a 10 anos
Granularidade	Detalhados	Detalhados e resumidos
Redundância	Não Ocorre	Ocorre
Estrutura	Estática	Variável
Manutenção desejada	Mínima	Constante
Acesso a registros	Dezenas	Milhares
Atualização	Continua (Tempo real)	Periódica (em batch)
Integridade	Transação	A cada atualização
Número de índices	Poucos/simples	Muitos/complexos
Intenção dos índices	Localizar um registro	Aperfeiçoar consultas

Fonte: VERZOLA (2008)

## 1.10 Data Mart

Um *Data Warehouse* não precisa ser construído todo de uma vez, devido a sua complexidade, pode-se criar um processo de negócios que são concebidos aos poucos. Um subconjunto do DW, com propósito específico relacionado, são os *Data Marts*, sistemas de suporte a decisão, que podem incorporar dados substanciais, mas contêm muito menos dados que teria um *Data Warehouse* desenvolvido para o mesmo projeto.

*Data Marts* são focalizados em propósitos específicos do negócio, o planejamento do sistema e a análise dos requerimentos são mais facilmente gerenciáveis, e o projeto, implementação, fase de testes e instalação são bem mais baratos que para um *Data Warehouse* (INMON, WELCH e GLASSEY, 1999).

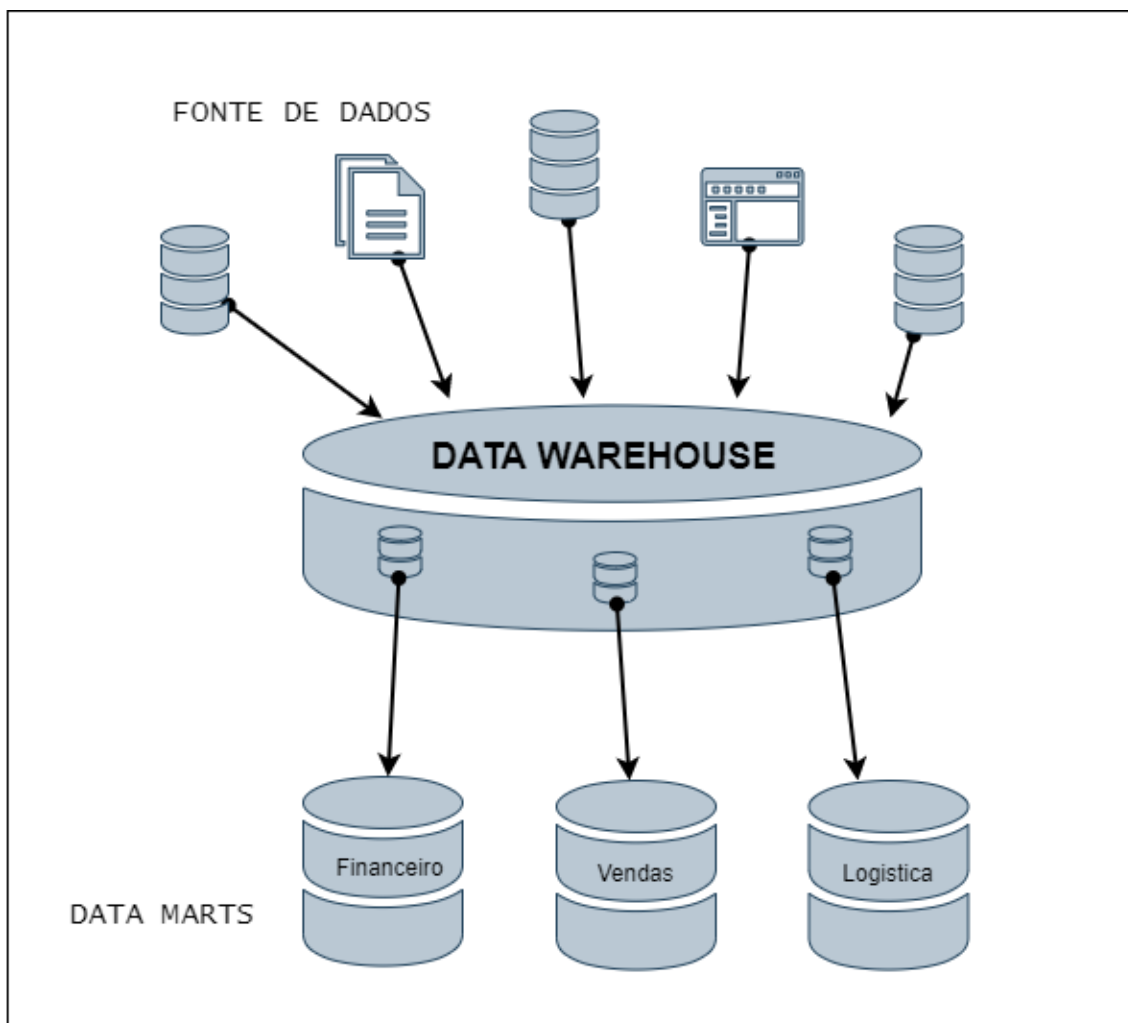
*Data Marts* podem ser entendidos como extensões do DW. O aprendizado que é obtido na conclusão do projeto de cada *Data Mart* traz contribuições para revisões periódicas nos objetivos do projeto do *Data Warehouse* corporativo, servindo de apoio para a própria revisão dos objetivos da organização e seu plano estratégico e tático. (PENNA, 2003).

Os projetos de *Data Marts* devem ser inicialmente simples e úteis para que possam atingir seus objetivos de forma rápida e clara. Não é desejável para uma empresa investir uma quantia em dinheiro e tempo de seus funcionários em um projeto que pode levar meses para ser concluído e que durante o processo de implantação possa terminar por gerar controvérsias e até mesmos problemas para os setores (KIMBALL, CASERTA 2004).

KIMBALL e ROSS (2002) também **explica** que um *Data Mart* é como se fosse um *Data Warehouse* com menor capacidade e complexidade, e são usados para atender a uma unidade específica de negócios. Portanto, são tradicionalmente mais fáceis de construir e manter.



Figura 7: Data Mart Infraestrutura



Fonte: O autor

Portanto pode se dizer que um Data Mart, possui as mesmas características de um DW, **porém** é criado para atender a um setor ou objetivo menor, como exposto na Figura 7, com os Data Marts dos setores financeiro, vendas e logística, e possui uma complexidade de criação e manutenção menor pois o seu volume de dados e abrangência é menor.

### 1.11 Modelo Dimensional

Segundo KIMBALL (2008), se o dado for numérico e variar continuamente a cada amostragem, ele será considerado um fato. Do contrário, se for uma descrição praticamente constante de um item, será considerada um atributo de dimensão.

A modelagem dimensional resulta em um design de banco de dados consistente, criando uma redundância de dados útil, que evita cálculos repetitivos que inibem o desempenho toda vez que um relatório é preparado.

#### 1.11.1 *Esquema Floco de Neve (Snow Flake)*

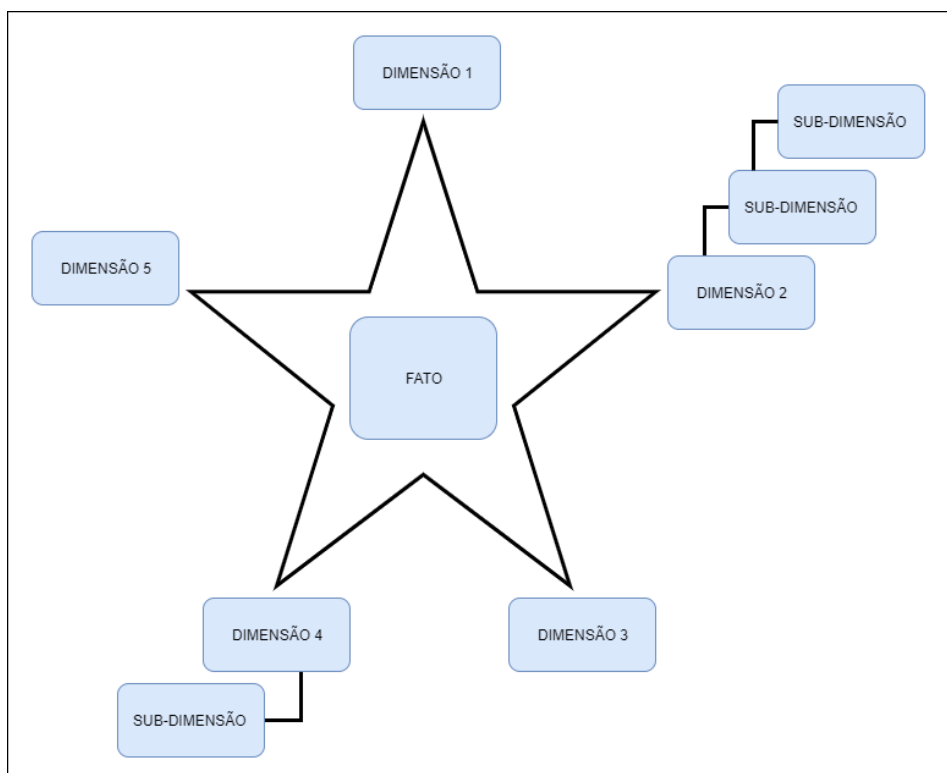
Um esquema de floco de neve consiste em uma tabela de fatos central contendo as medidas ou algum outro conteúdo de interesse, e é cercado por entidades de dimensão contendo o contexto ajustado para a análise do fato. As dimensões geralmente se relacionam com os fatos em um relacionamento um para muitos. (KIMBALL; CASERTA, 2004).

O método do esquema de floco de neve, é um método de modelagem de dados aplicado no *Data warehouse*. É desenvolvido a partir do método *Star Schema*. No esquema de floco de neve, cada tabela de dimensão pode ter outra sub tabela de dimensão. A dimensão dos dados tem uma informação assunto que funciona como material para tomada de decisão, isso porque em cada dimensão de dados é possível gerar uma descrição mais detalhada. (PEYRAVI, 2012)

Uma vantagem do esquema floco de neve é a clara exposição da hierarquia dimensional, que frequentemente se relaciona diretamente aos níveis de agregação que podem ser aplicados ao fato. Por exemplo, a dimensão do tempo pode consistir em uma hierarquia representando ano, trimestre, mês, semana e dia. Essa hierarquia está diretamente relacionada as agregações temporais que podem ser aplicadas ao fato, ou seja, visualizar os fatos consolidado por ano, trimestre, mês, semana ou dia.

A principal desvantagem do esquema floco de neve é o grande número de tabelas que precisam ser unidas para suportar até as mais consultas básicas.

Figura 8: Esquema modelo Snow Flake (Floco de neve)



Fonte: O autor

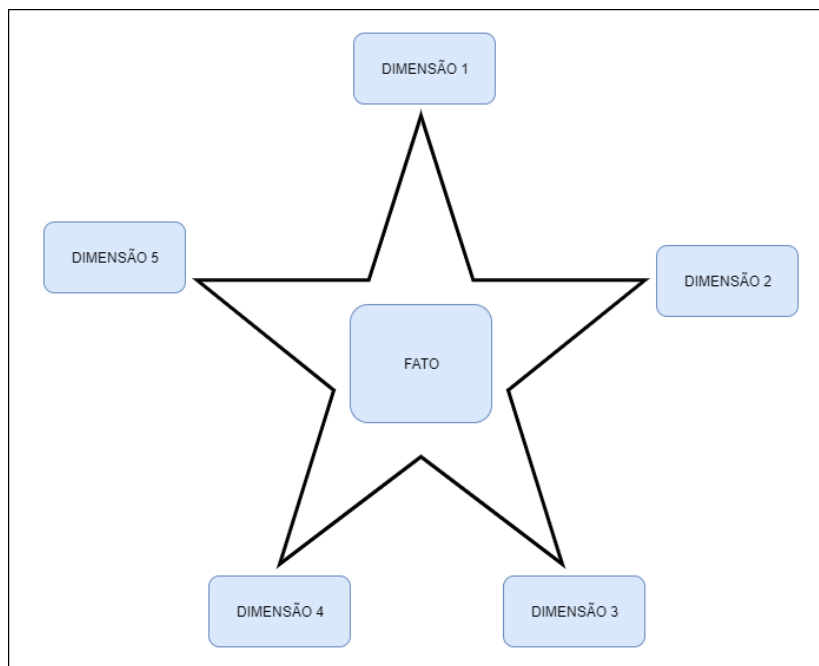
O esquema floco de neve, ilustrado pela Figura 8, é uma variação do esquema estrela e consiste na decomposição de uma ou mais dimensões que devem ser normalizadas. Esse tipo de esquema é utilizado quando se tem dimensões de grandes tamanhos, portanto, sua principal desvantagem é a complexidade nas consultas sobre o número de tabelas relacionadas. (BARBIERI, 2001).

#### 1.11.2 Esquema Estrela (Star Schema)

O esquema estrela, ilustrado pela Figura 9, é uma maneira de dispor as tabelas do modelo relacional para o modelo dimensional, podendo ser implementado em banco de dados relacionais e principalmente, em banco de dados dimensional (KIMBALL, 2008).

É uma estrutura simples, com poucas tabelas e ligações bem definidas, assemelha-se ao modelo de negócio. Permite a criação de um banco de dados que facilita a execução de consultas complexas, podendo ser realizadas de modo eficiente e intuitivo pelo usuário (BARBIERI, 2011).

Figura 9: Esquema modelo Star Schema (Esquema Estrela)



Fonte: O autor

Portanto, o esquema estrela consiste simplesmente em um fato cercado por um único nível de entidades de dimensão recolhidas ou consolidadas. Voltando ao exemplo dado em 1.11.1, todas as informações que representam ano, trimestre, mês, semana e dia teriam que ser representadas de forma menos efetiva em uma única entidade. Isso geralmente leva a um melhor desempenho da consulta, já que as consultas agora se juntam a menos tabelas, porém não fornece explicitamente suporte para hierarquias de atributos e as tabelas dimensionais são um problema.

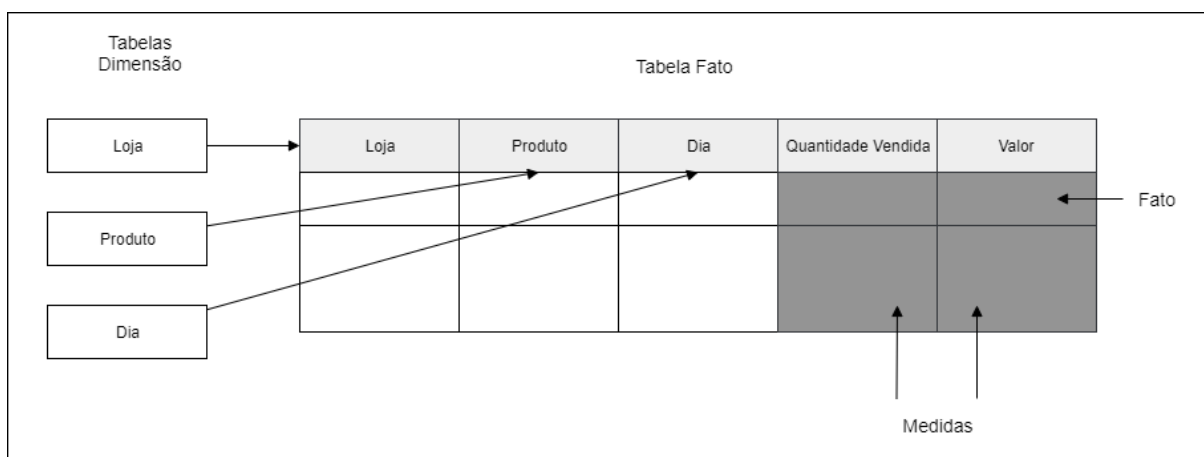
Qual técnica de modelagem é mais adequada, portanto, irá depender do projeto em questão.

### 1.11.3 Fato

Originando-se das entidades presente no modelo relacional que representam ações, eventos, acontecimentos, enfim, fatos que se pretende registrar, daí a origem do seu nome (BARBIERI, 2001). Na Figura 10 é apresentada a estruturação desse modelo.

Para a representação de um fato são utilizados atributos numéricos, designados medidas. Uma medida é determinada pela combinação das dimensões que participam de um fato e, por meio disso, é possível analisar o desempenho de um indicador de negócios. Portanto, desenvolver um DW é uma questão de unir as necessidades dos seus usuários com a realidade dos dados disponíveis (KIMBALL, 2002).

Figura 10: Composição elementar de uma tabela fato.



Fonte: BARBIERI (2001)

Na Figura 10, é exposto a estruturação de uma tabela fato, composta pelos valores quantitativos, e pelas chaves estrangeiras das tabelas de dimensões.

### 1.11.4 Dimensão

No que se refere às tabelas dimensão, constituem as estruturas de entradas, Barbieri (2001), estabelece que representam entidades de negócios e servem para armazenar informações como geografia (estado, cidade), tempo (dia, mês, ano), etc. As tabelas dimensão têm uma relação 1:N com as tabelas fato e armazenam um número, substancialmente, menor de linhas que elas.

As tabelas dimensões devem ser compreendidas como um recurso que realiza a filtragem de valores aplicados na manipulação dos fatos, por onde as consultadas entram no ambiente do *Data Warehouse* (KIMBALL, 2002).

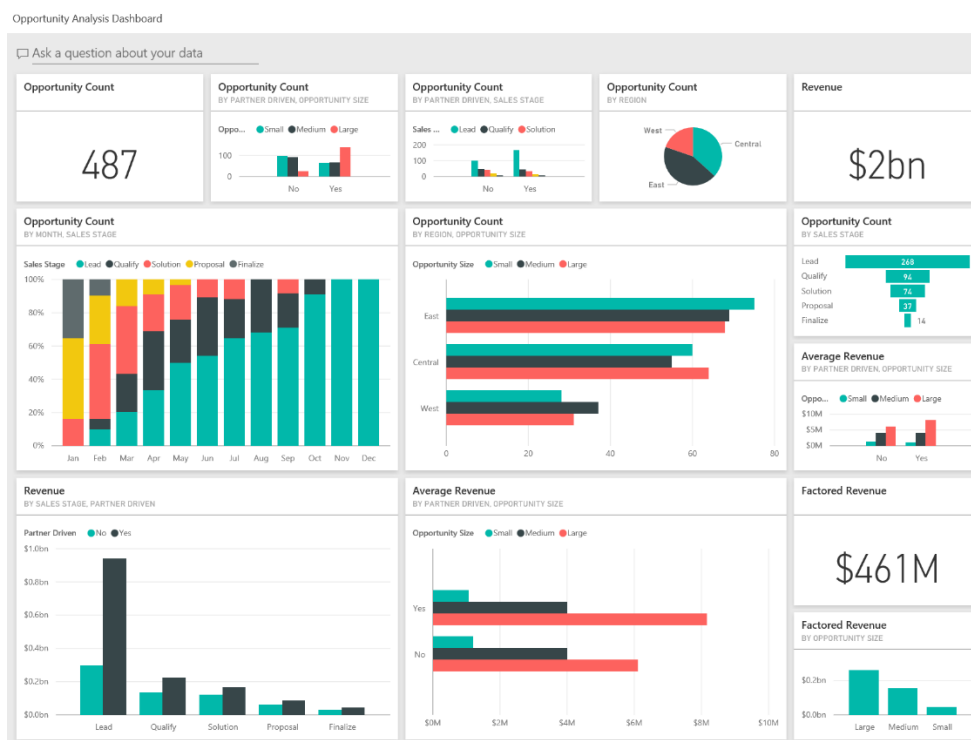
Normalmente, cada dimensão é um ponto de entrada ou mecanismo independente para filtrar os dados. Por exemplo, uma modelação para um *Data Warehouse* clássico possui as três seguintes dimensões: produto, geografia e tempo. Uma modelação em um contexto médico, por outro lado poderia ter tempo, diagnóstico e tratamento como dimensões.

## 1.6 Power BI

O *Microsoft Power BI* é um conjunto de ferramentas e serviços de análise de negócios, que consiste em aplicativos e serviços projetados para fornecer recursos visuais e insights interativos e coerentes sobre dados.

O *Power BI* é um conjunto de ferramentas de análise de negócios para analisar dados e compartilhar ideias. Os painéis do *Power BI* fornecem uma visão de 360 graus para os usuários corporativos com suas métricas mais importantes em um só lugar, atualizadas em tempo real e disponíveis em todos os seus dispositivos. Com um clique, os usuários podem explorar os dados em seu painel usando ferramentas intuitivas que facilitam encontrar as respostas. Criar um dashboard é simples graças a conexões com aplicativos corporativos populares, completas com dashboards predefinidos, que ajudam você a começar a trabalhar rapidamente. E você pode acessar seus dados e relatórios em qualquer lugar com os aplicativos Móveis *Power BI*, que se atualizam automaticamente com qualquer alteração em seus dados. (POWELL, 2017)

O *Power BI* permite que qualquer pessoa se conecte facilmente a qualquer um dos seus dados, crie painéis e relatórios ao vivo e explore dados por meio de visualizações interativas a qualquer momento. O *Power BI*, torna todos os seus dados visíveis em um único local, independentemente de onde os dados reside, permitindo uma visão consolidada das operações comerciais. (FERRARI; RUSSO, 2016).

Figura 11: Dashboard Criada com *Power BI*

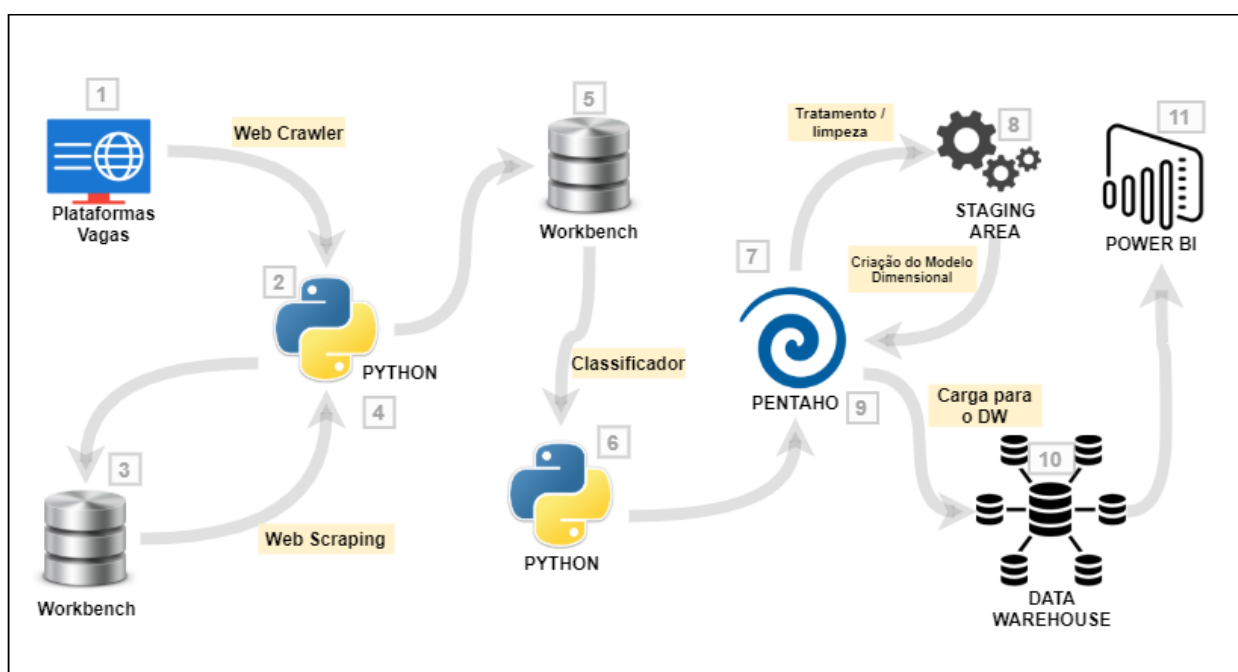
Fonte: MICROSOFT (2019)

O *Power BI* inclui dois aplicativos complementares. O primeiro é o *Power BI Desktop*, uma exploração visual de dados e ferramenta de relatórios. O segundo é um conjunto de aplicativos móveis interativos nativos para Windows, iOS e Dispositivos Android, fornecendo acesso seguro a painéis e relatórios ativos do *Power BI* a partir de qualquer dispositivo. Além disso, o *Power BI* pode ser estendido com um conjunto de *APIs REST* que permitem que os desenvolvedores integrem soluções de cliente e Web com o *Power BI* ou para criar visualizações personalizadas (POWELL, 2019), como o exemplo mostrado na Figura 11, onde os gráficos são combinados em uma única tela, formando uma *dashboard*.

### 3 METODOLOGIA

Para a conclusão do objetivo de utilizar o processo de ETL (Extração, transformação e carga) para reconhecimento e análise dos atributos que as empresas mais solicitam foi efetuada uma análise criteriosa da escolha das fontes para a extração dos dados, cada objeto vaga é submetido a diversas transformações e tratamentos para que seja feita carga do volume, utilizado conceitos de modelagem dimensional, para assim viabilizar a **análise** final do conjunto completo, para isso foi proposto um caminho ao qual cada objeto vaga será exposto.

Figura 12: Caminho dos dados



Fonte: O autor

A Figura 12, apresenta cada etapa do caminho de dados, inicialmente em (1) são extraídos as URL's das vagas, através do *Web Crawler* com Python (2), e são salvos no banco de dados em (3), logo após, as URL's são enviadas de volta ao Python (4) para efetivação da extração do conteúdo com *Web Scraping*, o resultado é novamente persistindo no banco de dados (5), e enviado ao Algoritmo de reconhecimento e classificação (6), feito em Python, após classificados os dados são enviados ao Pentaho (7) para o tratamento e limpeza, e enviados a *Staging área* (8), após tratados os dados, são submetidos a modelagem dimensional (9), para carga no



Data Warehouse (10), para finalmente serem cruzados e analisados com o Microsoft Power BI (11)

## 1.7 Levantamento dos dados

Antes de executar o algoritmo para a extração dados, foi feita uma análise das plataformas que ofertam vagas online, nessa análise foi verificada o número de vagas que cada plataforma ofertava, a disposição dessas vagas e a política de disponibilidade dos dados de cada empresa.

O processo de obtenção dos dados foi realizado por meio de ferramentas de extração de dados na web – abordados na seção 3.1 Web Crawler e 3.2 Web Scraping. Um ponto importante a ser destacado é a análise da legalidade dos dados, onde foi verificada a política de privacidade de cada plataforma, observando destaques sobre a prescrição de mecanismos de extração de conteúdo.

### 1.12 Análise das fontes e da legalidade da extração dos dados

A fim de definição de escopo, foram pesquisadas apenas plataformas que dispunham de vagas destinadas a profissionais da tecnologia da informação. Essa análise foi efetuada no dia 23 de maio de 2019, e foram escolhidas as plataformas que mais se encaixaram no perfil da pesquisa, a análise foi efetuada sobre os critérios da política de privacidade e termos de uso individuais de cada plataforma, também foi analisado o número total de vagas ofertadas e o número de vagas ofertadas relacionadas a TI.

Quadro 1: Nível de restrição sites quanto a utilização de Scraping/Crawler

PLATAFORMA	URL	Nível política privacidade para uso dos dados	Uso de Crawler/Scraping
empregos.com.br	empregos.com.br	Não Descrito	Não Descrito
Vagas	vagas.com.br	Alta	Proibido
Empregos TI	empregos.professionaisti.com.br	Não Descrito	Não Descrito
InfoJobs	infojobs.com.br	Alta	Proibido
Catho	catho.com.br	Não Descrito	Não Descrito
ApInfo	apinfo.com	Alta	Não Descrito
Trabalha Brasil	trabalhabrasil.com.br	Não Descrito	Não Descrito

Fonte: O autor

A plataforma *empregos.com.br*, possuía em seus termos de uso cinco tópicos principais: Normas e condições para utilização dos serviços; obrigações da plataforma; obrigações dos usuários; disposições gerais e restrições – nas quais normalmente as plataformas descrevem a proibição de uso de mecanismos de extração automatizada – em sua política de privacidade, é descrito primeiramente que a empresa realiza uma captura de dados e registro de atividades dos usuários do site, logo após, é feita uma breve descrição do armazenamento dos dados possuídos pela empresa, por fim aponta que faz o uso interno das informações dos candidatos e empresas, e que parte desses são acessíveis publicamente sem danos as partes envolvidas.

Em nenhum dos cinco tópicos dos termos de uso, nem na política de privacidade ficou evidenciado intenção de posição contrária da empresa quanto à utilização dos dados publicados pela sua plataforma.

Em seu portal, a plataforma *catho.com* descreve em sua política de privacidade, uma estrutura com quatro informações principais, sobre como o site coleta e utiliza as informações dos usuários que utilizam os serviços da *Catho*, sobre a comunicação do site com os usuários, sobre a segurança que o site oferece após o usuário fazer login, e sobre o conteúdo do site. A plataforma não possui um termo de uso geral para o site, mas sim dois termos de usos específicos, que são eles: “Termo de uso – recrutamento perfeito” e “Termos de usos PcD – Pessoas com deficiência”. Novamente não foi observado qualquer impedimento quanto ao uso de Crawler/Scrap em sua plataforma.

A plataforma *apinfo.com* não possui termos de uso, e na sua política de privacidade a empresa discorre apenas sobre a coleta e utilização das informações dos usuários, garantindo o uso desses dados exclusivamente para prestação dos serviços descritos no site. Entretanto no rodapé da sua página inicial, é expresso a frase: - “É proibida a reprodução do conteúdo deste site, em qualquer meio, eletrônico ou impresso, sem autorização escrita.” – o suficiente para impossibilitar a utilização da plataforma no projeto.

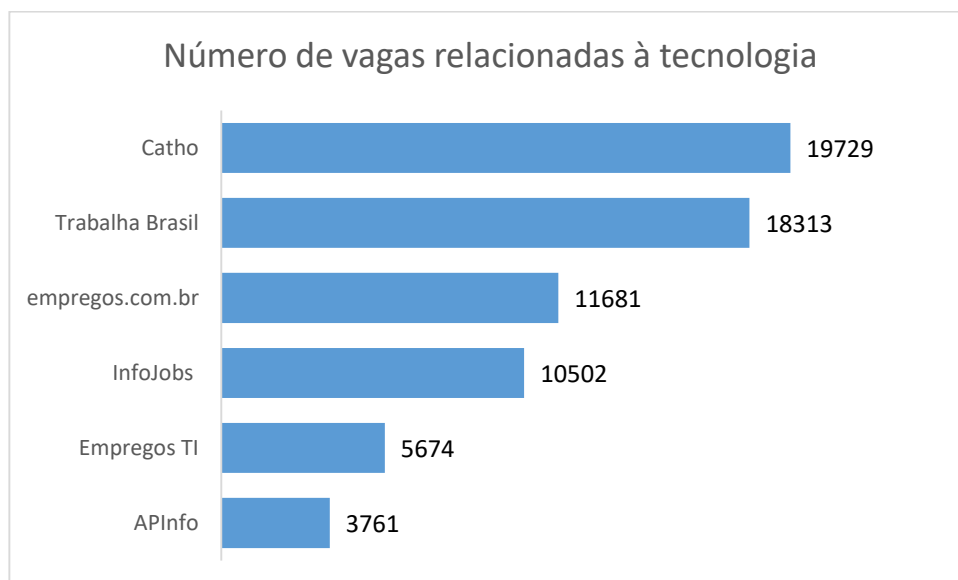
Os sites *empregos.profissionaisti.com.br* e *trabalhabrasil.com.br* não possuem termos de uso nem política de privacidade, tampouco aviso nem advertência visível em sua plataforma quanto a utilização dos *Crawler/Scraping*.

Já as plataformas *vagas.com.br* e *infojobs.com.br* explicitaram em sua política de privacidade a proibição dos mecanismos de extração de dados *Crawler/Scraping*, por conseguinte não farão parte das plataformas utilizadas para montagem do *dataset* da pesquisa.

Para definição de quais plataformas serão usadas no projeto, também foi analisado o número de vagas que cada plataforma conta em seu catálogo. As plataformas *empregos.profissionaisti.com* e *ApInfo.com* divulgam vagas exclusivamente da área de tecnologia da informação, e seu número de vagas ofertadas foi relativamente baixo, o site *vagas.com* não divulga o número de vagas existente em seus bancos de dados.

As plataformas *catho.com*, *catalogo*, *empregos.com* e *infojobs.com.br*, apresentaram um ótimo número de vagas para o projeto, somado, as quatro plataformas ofertaram no dia da análise mais de 60 mil vagas (Gráfico 1).

Gráfico 1: Número de vagas relacionadas à tecnologia

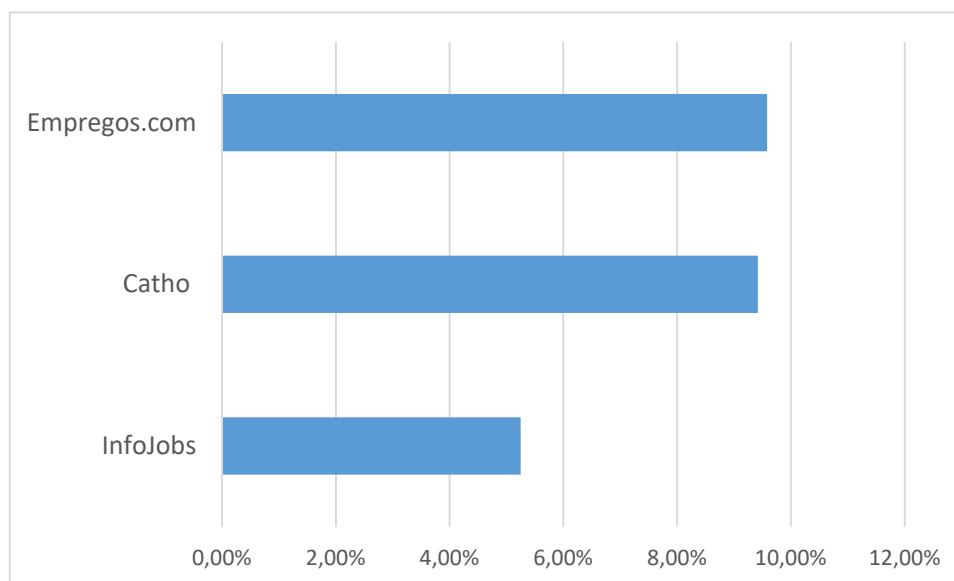


Fonte: O autor

O Site *trabalhabrasil.com.br* apesar de apresentar um bom número de vagas, a sua estrutura de disposição das vagas conhecida como rolagem infinita, impede o uso da biblioteca de extração *BeautifulSoup 4*, usada no projeto. Para realizar a extração, seria necessário a utilização de bibliotecas específicas para o tipo de sintaxe citado, por esse motivo as vagas ofertadas pela plataforma citada não fizeram parte da montagem do *dataset*.

As plataformas *empregos.com.br*, *empregos.profissionaisti.com* e *catho.com* não especificam em sua política de privacidade tampouco nos termos de uso contrariedade acerca da utilização de mecanismos de extração para fins não comerciais e de pesquisa. Também apresentaram bons números de vagas ofertadas, 37.084 no total, por conseguinte foram escolhidas para fazer parte das plataformas utilizadas para montagem do *dataset* da pesquisa.

Gráfico 2: Porcentagem de vagas relacionadas a TI



Fonte: O autor

Outro dado relevante da análise realizada, se trata do número de vagas da área de tecnologia em relação ao número total de vagas ofertadas nas três plataformas que ofertam vagas para todas as áreas. Nos três portais (Catho, InfoJobs e *empregos.com.br*) essa relação teve uma média de 8,08%, destacando como as vagas de TI atualmente ocupam uma grande porção no mercado de trabalho, como visto no Gráfico 2.

### 1.13 Caracterização dos dados

Após a análise anterior, três plataformas se encaixaram perfeitamente no perfil da pesquisa, *empregos.profissionaisti.com*, *catho.com*, e *empregos.com*. O primeiro passo a ser executado foi a verificação da disposição das vagas em cada plataforma, para preparação do algoritmo de extração pois a sua funcionalidade se dá de forma automatizada e única para cada plataforma.

Figura 13: Estruturação de blocos em empregos.com.br

**Programador Web** ← 1  
IT4FINANCE | São Paulo/SP - 1 vaga  
Publicado em 24/10/2019

Seja o primeiro a avaliar 2 ↗

---

**Dados da Vaga** 3

4 ↗

**Descrição:** Irá fazer atuação em solicitações de manutenções e requisições do ambiente de produção do sistema.  
Local de trabalho: Pinheiros-SP.

**Qualificação:** Desejável HTML 5, Ajax, Devexpress, Crystal Reports, SQL Server, conhecimento de programação de WebServices (WCF).  
Desenvolvimento de Software, Imprescindível Visual Studio 2015 (C#, ASP.Net)

**Formação:** Ensino superior. ← 5

**Local de Trabalho:** São Paulo / SP - 1 vaga ← 6

**Dados da Empresa** 7

**Nome:** It4finance

**Ramo:** Informática/ Tecnologia

**Descrição:** Desenvolvimento de programas de computador sob encomenda.

**Salário:** ← 8  
**R\$3.000,00** C

Veja média salarial

**Forma de Contratação:**  
Efetivo (CLT) ← 9

**Benefícios:**  
A Combinar ← 10

**Horário de Trabalho:**  
De segunda a sexta-feira das 09h às 18h.  
↑ 11

Fonte: O autor

Na plataforma *empregos.com.br* as vagas são dispostas em 4 blocos, A, B, C e D (Figura 13). No bloco A foi extraído o título da vaga (que posteriormente será submetida para classificação do cargo e área), a quantidade de vagas que está sendo ofertadas, a cidade, e a data de publicação da vaga.

O bloco B, observam-se as informações consideradas mais importantes sobre a vaga, o campo descrição que servirá para reconhecimento de 3 atributos, a qualificação, formação, local de trabalho e número de vagas.

No bloco C, foi retirado o salário, benefícios, regime de contratação e a jornada de trabalho.

O bloco D, exibe apenas informações inerentes a empresa que está disponibilizando a vaga, a plataforma *empregos.com.br* não disponibiliza essas informações para todas as vagas, apenas para um número pequeno de vagas patrocinadas, portanto esse bloco não foi extraído.

Figura 14: Estruturação de blocos em *empregos.profissionaisti.com.br*

1 vaga | Código: 5201

## Programador / Desenvolvedor Delphi Pleno

5 LABOR SERVICOS ADMINIS... Cachoeirinha (RS) Desenvolvimento de Softwa... 03/11/2019

Efetivo (CLT) Período Integral

---

PERFIL DESEJADO: Agilidade nas ações, atenção de detalhes, confiabilidade e clareza nas informações, espírito de equipe, iniciativa, proatividade, auto gerenciável, senso crítico e dinamismo.

REQUISITOS:

- Curso superior na área de TI (completo ou em andamento)
- Experiência mínima de 2 anos
- Conhecimento em SQL
- Conhecimentos em C# serão um diferencial

ATIVIDADES:

- Desenvolver e dar manutenção seguindo os requisitos definidos para a atividade e garantir a atividade desenvolvida.
- Enviar currículo para [oportunidades@cooperacaorh.com.br](mailto:oportunidades@cooperacaorh.com.br)

**Salário:** à combinar

**Responsável:** Patrícia

**Exigências**

- Ensino Superior

Fonte: O autor

Na plataforma *empregos.profissionaisti.com.br* as vagas foram divididas em dois blocos principais, A e B (Figura 14), o bloco B representa apenas o campo descrição, e foi submetido ao algoritmo de classificação para reconhecimento de outros atributos que já existem no texto, mas não estão separadas por tags HTML como será visto no próximo capítulo.

A partir do bloco A, foram extraídos os seguintes campos: quantidade de vagas, título da vaga, cidade, data de publicação, regime de contratação e jornada de trabalho.

Figura 15: Estrutura site *catho.com*

**Analista de Infraestrutura** ← 1  
De R\$ 5.001,00 a R\$ 6.000,00 | Ipatinga - MG (1) ← 2

3 → Realiza backups de servidores e estações de coleta de dados. Otimizar os recursos de servidores e estações de coleta de dados. Instalar e configurar host e máquinas virtuais; Instalar e configurar drives de comunicação padrão OPC, 4 configuração DCOM. Instalar e configurar interfaces de coleta de dados (PIMS). Instalar e configurar câmeras de CFTV. Elaborar descritivo funcional, projeto, manuais de manutenção e todos outros documentos e desenhos necessários ao projeto. Participar de todas as etapas do projeto: levantamento de requisitos, desenvolvimento, implantação e testes (testes individuais e integrados, testes de performance). Participar das reuniões de acompanhamento nas diversas fases do projeto. Treinar a equipe de manutenção e usuário final, outras atividades correlatas. Infraestrutura de redes e padrão de comunicação OPC (configuração DCOM). Conhecimento de processos siderúrgicos. Redes de CFTV. Conhecimento e experiências das ferramentas Acronis, Vcenter e VMWare. Inglês para leitura de manuais.

← 5

---

**BENEFÍCIOS** ← 6  
Assistência Médica / Medicina em grupo, Restaurante na empresa

**HORÁRIO** ← 7  
De segunda a sexta, das 8:00 às 17:00.

**REGIME DE CONTRATAÇÃO** ← 8  
CLT (Efetivo), Prestador de serviços (PJ)

---

**DADOS DA EMPRESA**  
Exclusivo para Assinantes

**QUALIFICAÇÃO**  
Utiliza a Catho há 6 anos e 4 dias.

Fonte: O autor

No site *catho.com*, as vagas também estão dispostas por 4 blocos principais, no bloco A, considerado como cabeçalho, foram extraídos o título, salário, local da vaga e quantidade de vagas ofertadas. No bloco B temos apenas o campo descrição, que serve de base para reconhecimento de importantes atributos para análise da vaga. Na Figura 15 pode-se observar três campos no bloco C, que são eles: Benefícios, horário e regime de contratação. Contudo o número de campos do bloco varia de vaga para vaga, podendo ter ainda mais um campo denominado “Informações adicionais”.

Explorada as peculiaridades de cada *template*, termina a reunião das informações necessárias para a extração em si, a próxima etapa foi a abordagem dos algoritmos de *Crawler* e *Scraping* que realizam a raspagem.

## 1.8 Web Crawler

O papel do Crawler no projeto, foi extrair as URL's de todas as vagas ofertadas pelas plataformas e salvá-las num banco de dados. Para isso ele foi capaz de visitar todas as páginas do site, e para cada página, salvar todas as URL's de vagas que estavam presentes. Em cada *site* o número de URL's de vagas por página é diferente, para tanto se fez necessário uma análise do código html e a busca por padrões, a linguagem Python provê de uma biblioteca poderosa para busca de tags na sintaxe html, a biblioteca BeautifulSoup 4, que contém métodos de pesquisa para busca por tags específicas ou ainda pelas nomenclatura de seus ids e classes.

Toda a escrita do código foi feita utilizando o PyCharm Professional 2019.2.4. O PyCharm é uma IDE com suporte ao Python e ótimos recursos como autopreenchimento e detecção de erros de código além de pesquisa inteligente saltando entre classes, arquivos ou símbolos (Jetbrains, 2019). Utilizando o Python na sua versão 3.8.0.

Ao analisar o código html do site *empregos.com.br* percebe-se que o padrão das urls de páginas, são tags "a" com o conteúdo do atributo "href" terminadas com os caracteres "/informática-ti-e-internet/p" precedido da numeração da página que será redirecionada (Figura 16).

Figura 16: Padrão de disposição dos links

```

▶ <div id="ct100_ContentBody_divPaiMioloBusca" class="emp-grid">...</div>
▶ <div class="emp-paginacao">...</div>
▼ <div id="ct100_ContentBody_divMoldePaginador" class="paginacao">
  ▼ <div class="emp-paginacao">
    ▼ <div class="pagination-list area-paginacao">
      ▼ <ul>
        ▶ <li>...</li>
        ▶ <li>...</li>
        ▼ <li == $0
          ▶ <a id="rptPaginas_ct103_lkbPagina" href="https://www.empregos.com.br/vagas/informatica-ti-e-internet/p3" data-valor="3">3</a>
          </li>
        ▶ <li>...</li>
        ▶ <li>...</li>
        ▶ <li>...</li>
        ▶ <li>...</li>
        ▶ <li>...</li>
      </ul>
      ▶ <a id="lkbProximo" title="Próximo" class="next" href="https://www.empregos.com.br/vagas/informatica-ti-e-internet/p2" data-valor="2">...</a>
    </div>
  </div>
</div>

```

Fonte: *empregos.com.br* (2019)



Para a extração dessas URL's foi desenvolvido um método que percorre por todas as tags "a" com o atributo "href" preenchido, e em seguida executa um condicional, que verifica se em cada tag possui os caracteres "/informática-ti-e-internet/p", e caso os critérios dessa busca retornem positivo, insere essa url na lista url\_paginas (Apêndice 1).

Para cada iteração do algoritmo de Crawler, é extraído também as URL's direta das vagas, para isso o método extract\_links que se utiliza dos mesmos passos do método anterior, se diferenciando no modo da busca, que utiliza de uma expressão regular (Apêndice 2), para encontrar os links terminado com uma combinação de sete caracteres numéricos (Figura 17).

Figura 17: Padrão de nomenclatura de urls

```

▶ <div class="selo-vaga-patrocinada vagas-list">...</div>
▼ <div class="grid-13-16 col-esquerda" data-client-id="colEsq">
  ▶ <div class="empresa grid-4-16 txtcenter logo-item" data-client-id="logoItem">...</div>
  ▼ <div class="descricao grid-12-16" data-client-id="descricaoItem">
    ▼ <h2>
      <a href="https://www.empregos.com.br/vagas/analista-de-suporte-tecnico/santo-andre/sp/6315390" onclick="
      window.open('https://www.empregos.com.br/vagas/analista-de-suporte-tecnico/santo-andre/sp/6315390/
      modal','XXX', 'scrollbars=1, width=960, height=768, location=0' ); return false;" data-vaga="6315390">
      Analista de suporte técnico
      </a> == $0
    </h2>
  </div>
  ▶ <span>...</span>
</h2>

```

Fonte: *empregos.com.br* (2019)

Como o scraper executou de forma automatizada, foi definido um método (Apêndice 3) que descreve os passos necessários para a extração dos dados:

1. Insere na lista "available\_urls\_page" a URL "https://www.empregos.com.br/vagas/informatica-ti-e-internet/"
2. Inicia uma iteração nos itens da lista "available\_urls\_page" com critério de parada condicionado a presença de itens na mesma.
3. Retira da lista "available\_urls\_page" uma url aleatória e a salva na variável "url"
4. Verifica se a url já existe na lista "seen\_urls\_page"

5. Caso não exista, dentro de uma estrutura de captura de erros “try catch”, extrai para a variável “content” o conteúdo HTML da url, com o método get da biblioteca Requests, com o parâmetro “timeout” configurado em três segundos, para que o processo de *Crawler* não sobrecarregue o site com requisições e corra o risco de derrubá-lo.

6. Caso apresente algum erro na instrução de requisição, a iteração é cortada e segue para o próximo laço.

7. Chama a função “extract\_links” salvando as urls extraídas na lista “url\_vagas”.

8. Sempre que a lista “url\_vagas” conter 100 ou mais links, chama o método “insert.crawl\_links\_empregoscombr” que faz a persistência desses links no banco de dados na tabela “crawl\_links”.

9. Adiciona a URL na lista “seen\_urls\_page”

10. Chama a função “extract\_newpages” inserindo as URL’s extraídas na lista “avaible\_urls\_page”

Após a execução, a tabela “crawl\_links\_empregoscombr” estará povoada com o montante de URL’s de vagas extraído pelo algoritmo, a performance de extração, bem como a porcentagem de urls extraídos com sucesso será detalhado na seção Apresentação dos Resultados.

Finalizado o processo na primeira plataforma, sequencialmente é chamado os métodos correspondentes ao *Crawler* das páginas *catho.com* e *empregos.profissionaisti.com.br*, para cada plataforma foi explorado seus padrões de nomenclatura de URL’s, e adequação do algoritmo para extração das URL’s de páginas e URL’s de vagas (Apêndices 4 e 5). A partir dessa etapa, temos o necessário para finalmente realizar a extração dos dados definidos anteriormente.

## 1.9 Web Scraping

Essa seção relata a execução do processo de raspagem de dados dos sites. Para tal, será usado técnicas de *Web Scraping* conforme as peculiaridades e disposição de elementos de cada site, com o uso da biblioteca *BeautifulSoup 4*.

A ferramenta foi executada em um notebook Dell *Alienware* com processador Intel Core i7-8700 de 4.5GHz, 16GB de memória RAM, 128GB de SSD, Windows 10 Pro e uma conexão a cabo, da SIGNET, de 50Mb/s. Os dados foram armazenados localmente, por esse motivo não houve problemas quanto ao alto consumo de memória, nem de elevada demanda por processamento.

Para melhor estruturação, o algoritmo de *Scraping* foi modelado para extrair todo o código-fonte das páginas, e ignorar o conteúdo irrelevante para a pesquisa, persistindo no banco de dados somente os dados com valor para pesquisa. Os métodos “find()” e “find\_all” da biblioteca *BeautifulSoup 4* faz a captura de pequenas partes do conteúdo do código-fonte, filtrando por tags, ou até mesmo a classe ou id dessas tags.

Foi observado dois tipos de campos na estrutura da vaga divulgada, os campos com informações subjetivas e implícitas na qual chamaremos de campos pai, e um campo mais objetivo e explícito, onde as informações são manifestadas de forma mais concisa, sem a presença de itens que não caracterize outro atributo além daquele anunciado, chamaremos esses de campos filho. Um exemplo de campo pai é o campo “Informações adicionais”, onde através desse campo posteriormente serão reconhecidas diversas informações que compõem dados dos campos filhos de um ou mais atributos.

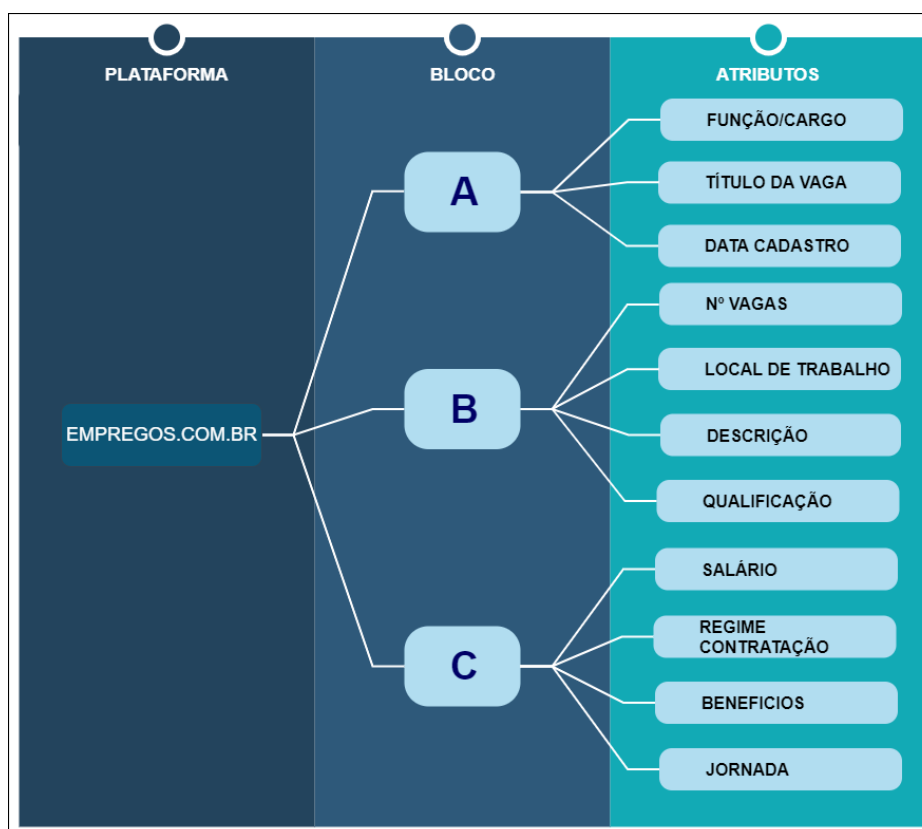
Cada plataforma possui uma disposição e presença de elementos único, portanto, para cada plataforma, será necessário um esquema de extração diferente, que será retratado nos próximos três subcapítulos.

### 1.14 Scraping Plataforma 1

No site *empregos.com.br*, para cada bloco foi feito um método que buscam as tags que abrigam cada elemento. Para o bloco A, o método “extract\_box\_header” (Apêndice 5), extrai as informações do atributo “data\_cadastro”, além do atributo “funcao” que posteriormente foi submetido ao algoritmo de classificação e reconhecimento para categorização da área de atuação e cargo.

O bloco B contém o atributo de maior valor para a pesquisa, o campo descrição, que através desse campo o algoritmo de classificação será capaz de reconhecer outros atributos inerentes a cada vaga. Para isso o método “Extract\_box\_body” (Apêndice 6) extrai além da descrição, os campos “n\_vaga”, “local\_trabalho” e “qualificacao”.

Figura 18: Esquema Scraping em *empregos.com.br*



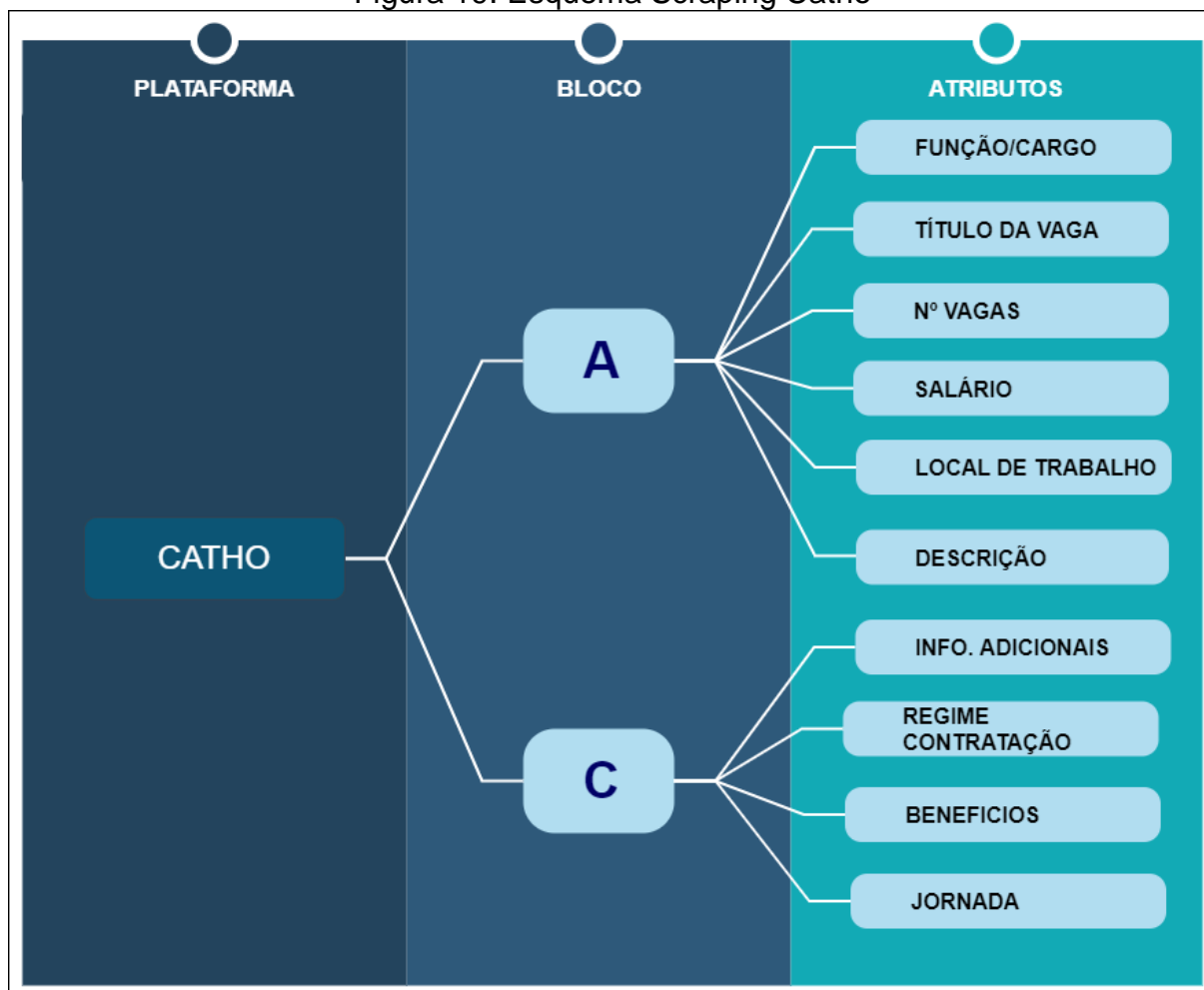
Fonte: O autor

O Bloco C, é um bloco estático, pois nas vagas analisadas sempre mostrou os mesmos tipos de informações: “salario”, “regime\_contratacao”, “beneficios” e “jornada”, a extração de seu conteúdo é efetuada com o método “Extract\_box\_salario” (Apêndice 7). Na Figura 18 é mostrado o conteúdo de cada bloco.

### 1.15 Scraping Plataforma 2

No site *catho.com*, a extração também foi feita através de três blocos principais. Para o bloco A, local de maior concentração de atributo, foi criado o método chamado de “extract\_box\_header” (Apêndice 8), que extrai os dados “data\_cadastro”, “qtd\_vagas”, “salario”, “local\_trab”, “descricao” e “cargo”.

Figura 19: Esquema Scraping Catho



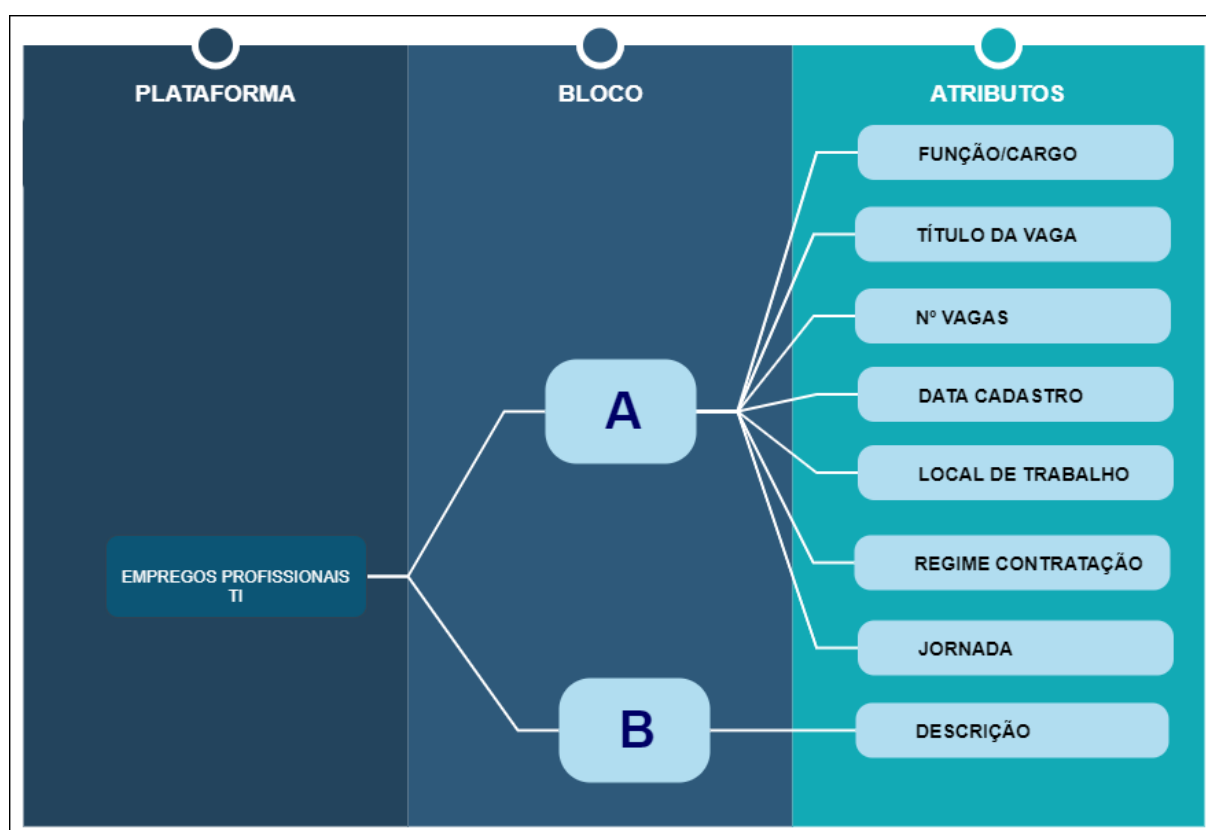
Fonte: O autor

No bloco B pode-se ter presente até quatro atributos, dependendo da qualidade de informação de cada oferta, os atributos são: "info\_adicional", "regime", "jornada" e "beneficios", o método criado para a extração do bloco denomina-se "extract\_box\_body" (Apêndice 9). A exemplificação de sua extração é retratada na Figura 19.

### 1.16 Scraping Plataforma 3

Já o site *empregos.professionaisti.com.br*, concentra a maioria das informações no bloco A, ficando no bloco B apenas o atributo “descricao”, os métodos de extração são respectivamente para A e B: “extract\_box\_header” (Apêndice 10) e “extract\_box\_body” (Apêndice 11).

Figura 20: Esquema Scraping Profissionais TI



Fonte: O autor

No bloco B estão presentes os atributos: “funcao”, “n\_vagas”, “local”, “regime”, “jornada” e “data\_cad”. A imagem da Figura 20 resume o esquema de extração criado para o site *empregos.professionaisti.com.br*.

### 1.17 Extração do conteúdo

A criação de um algoritmo genérico serviu como base para a personalização da extração do conteúdo de cada plataforma (Apêndices 12, 13 e 14), esse código descreve os passos necessários para a extração dos dados:

1. Importa a lista de urls da tabela "craw\_links" (extraído anteriormente pelo *Web Crawler*)
2. Inicia uma iteração nos itens da lista "urls" com critério de parada condicionado a presença de itens na mesma.
3. Dentro de uma estrutura de captura de erros "try catch", extrai para a variável "content" o conteúdo HTML da URL, com o método get da biblioteca *Requests*, com o parâmetro "timeout" configurado em três segundos, para que o processo de *Crawler* não sobrecarregue o site com requisições e corra o risco de derrubá-lo. Também dentro da estrutura as instruções para extração de atributos definidos por blocos.
4. Caso apresente algum erro na instrução de requisição, a iteração é cortada e segue para o próximo laço.
5. Chama o método "insert.scrap<nomeplataforma>" que faz a persistência desses links no banco de dados, na tabela "scrap<nomedaplataforma>".



Importante salientar que para cara plataforma, foi necessária reconhecer o padrão de nomenclatura de suas tags, bem como o seu id e a classe, que foram responsáveis pela automação do procedimento de extração. Esse padrão de nomenclatura é demonstrado no

Quadro 1.

Quadro 1: Padrão observado de nomenclatura de elementos

PLATAFORMA	ATRIBUTO	BASE PARA EXTRAÇÃO (TAGS)
Empregos.com.br	Cargo	"h1" dentro da tag de classe "info-empresa"
	Data Cadastro	"p" dentro da tag de classe "info-empresa"
	Salário	classe "valor itens" dentro da tag de classe "salario-vaga grid-4-16"
	Regime de Contração	Id "ctl00_ContentBody_divContratacao" dentro da tag de classe "salario-vaga grid-4-16"
	Benefícios	id "ctl00_ContentBody_divBeneficios" dentro da tag de classe "salario-vaga grid-4-16"
	Jornada	id "ctl00_ContentBody_divHorTrab" dentro da tag de classe "salario-vaga grid-4-16"
	Descrição	Id "ctl00_ContentBody_trDescricaoVaga"
	Qualificação	Id "ctl00_ContentBody_trQualificacao"
	Formação	Id "ctl00_ContentBody_trFormacao"
	Local	Primeiro "p" dentro da tag de classe "conteudo-vaga grid-12-16"
	Número de vagas	Segundo "p" dentro da tag de classe "conteudo-vaga grid-12-16"
Catho	Benefícios	"p" dentro da tag "section" de id "beneficios"
	Info Adicionais	"p" dentro da tag "section" de id "informacoesAdicionais"
	Regime	"p" dentro da tag "section" de id "regimeContratacao"
	Jornada	"p" dentro da tag "section" de id "horario"
	Cargo	Primeira tag "p" dentro da tag de id "anchorTituloVaga"
	Salário	Segunda tag "p" dentro da tag de id "anchorTituloVaga"
	Número de vagas	Terceira tag "p" dentro da tag de id "anchorTituloVaga"
	Data Cadastro	Quarta tag "p" dentro da tag de id "anchorTituloVaga"
	Local Trabalho	Quinta tag "p" dentro da tag de id "anchorTituloVaga"
	Descrição	"p" dentro da tag "section" de id "descricao-da-vaga"
Profissionais TI	Descrição	Tag de classe "job-content"
	Regime	Primeiro "span" dentro da tag "div" de id "content"
	Jornada	Segundo "span" dentro da tag "div" de id "content"
	Local de Trabalho	Primeira tag "a" dentro da tag de classe "job-clearfix"
	Função	Segunda tag "a" dentro da tag de classe "job-clearfix"
	Data Cadastro	Terceira tag "a" dentro da tag de classe "job-clearfix"
	Cargo	Tag de classe "page-title"
	Número de vagas	Tag "p" dentro da tag de classe job-header"

Fonte: O autor

Para a persistência das informações no banco de dados foi utilizada a biblioteca *pymysql* que com seus métodos, permite com poucos linhas de código-fonte a conexão e execução de *querrys* no sgbd mysql.

Ao final da execução do algoritmo nas três plataformas, se obteve as três tabelas com suas modelações descritas na Figura 21, na plataforma *empregos.com.br* se obteve 12 atributos, em *catho.com* foram 10 e *profissionaisti.com.br* 9 atributos.

Figura 21: Estrutura das tabelas no banco de dados

Tabela	Atributo	Tipo	Tamanho
scrap_empregoscombr	id	INT	11
	descricao	TEXT	
	qualificacao	TEXT	
	funcao	VARCHAR	500
	local_trabalho	VARCHAR	500
	titulo	VARCHAR	500
	n_vagas	VARCHAR	500
	salario	VARCHAR	500
	regime_e_contratacao	VARCHAR	500
	beneficios	VARCHAR	500
	jornada	VARCHAR	500
	link	VARCHAR	500
data_cadastro	VARCHAR	500	
scrap_catho	id	INT	11
	qtd_vagas	VARCHAR	50
	data_cadastro	VARCHAR	50
	salario	VARCHAR	50
	local_trab	VARCHAR	150
	descricao	TEXT	
	cargo	VARCHAR	500
	beneficios	VARCHAR	500
	jornada	VARCHAR	500
	info_adicional	VARCHAR	500
regime_e	VARCHAR	500	
scrap_profissionaisti	id	INT	11
	descricao	TEXT	
	regime_e	VARCHAR	100
	jornada	VARCHAR	100
	local	VARCHAR	100
	funcao	VARCHAR	100
	data_cad	VARCHAR	100
	titulo	VARCHAR	150
	n_vagas	VARCHAR	50
link	VARCHAR	500	

Fonte: O autor

Após finalizado o processo de extração da informação, se obteve subsídios para iniciar a etapa de transformação e limpeza dos dados. O volume de vagas extraídas pelo algoritmo, a performance de extração, bem como a porcentagem de URL's extraídos com sucesso será detalhado no tópico Apresentação de resultados.

## 1.10 Pentaho Data Integration – Transformações

Após a execução do algoritmo de *Scraping*, os dados ainda não estavam prontos para serem submetidos ao reconhecimento e classificação de atributos, necessitavam de uma limpeza. Nesse tratamento, são identificadas as anomalias e discrepâncias que poderiam vir a comprometer os procedimentos posteriores, podendo também prejudicar a análise.

Para tal, foi necessário partir de uma inspeção criteriosa da base de dados, identificando as inconsistências e procedendo com as medidas necessárias para sua adequação.

A transformação e carga foi realizado em uma das ferramentas do pacote Pentaho, o Data Integration ou Kettle em sua versão 8.3 A ferramenta forneceu o suporte necessário para a execução de todas as tarefas deste processo.

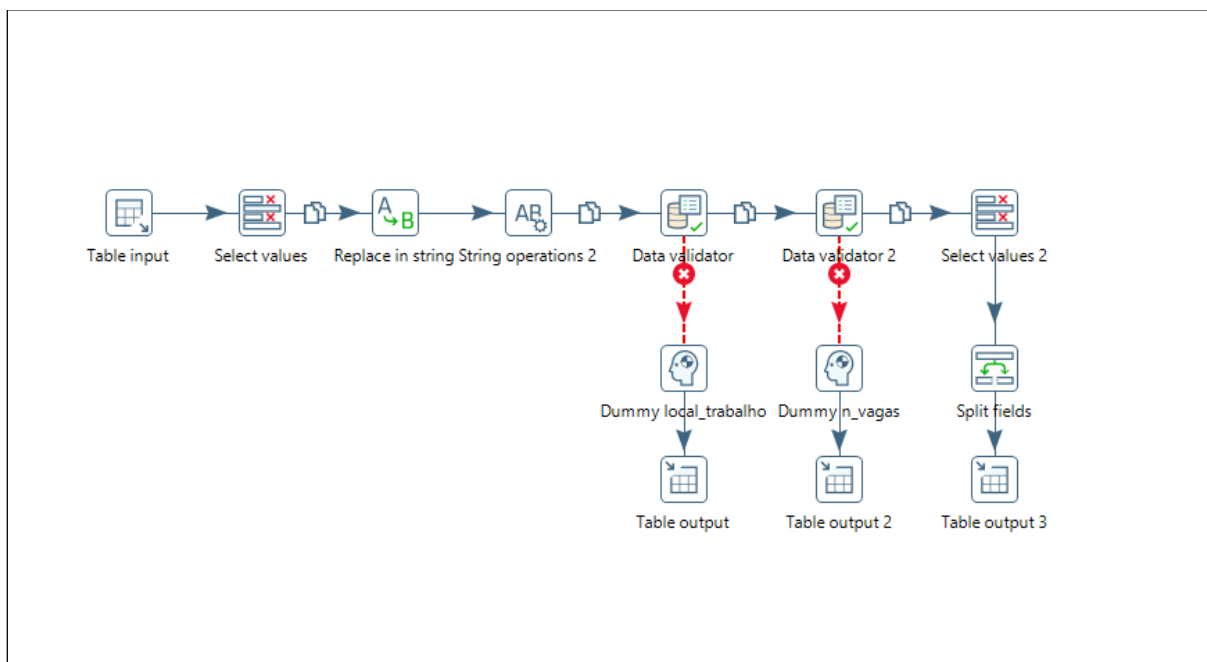
A etapa de transformação foi responsável por fazer uma limpeza nos dados úteis para o estudo, retirando ou acrescentando caracteres, criando campos que não existem, corrigindo entradas erradas e violação de restrições de integridade. Agindo também na identificação, exclusão ou tratamento de valores nulos.

É importante salientar que a limpeza dos dados também faz parte do controle de qualidade e, por isso, podemos concluir que este pré-processamento figura como uma etapa essencial para que os resultados de uma análise sejam confiáveis.

Os dados extraídos são então enviados para uma área de armazenamento temporário chamada *staging* área, uma área anterior ao processo de transformação e limpeza. Isto é feito para evitar a necessidade de extrair dados novamente, caso algum problema ocorra. Depois disso, os dados passaram pela transformação e processo de limpeza.

Para cada plataforma, foi necessária uma combinação de transformações e tratamentos, conforme a necessidade específica. Na Figura 22, é apresentada os *Steps* das transformações efetuadas na plataforma *empregos.com*.

Figura 22: (Data Integration - Kettle) Transformações em *empregos.com*



Fonte: O autor

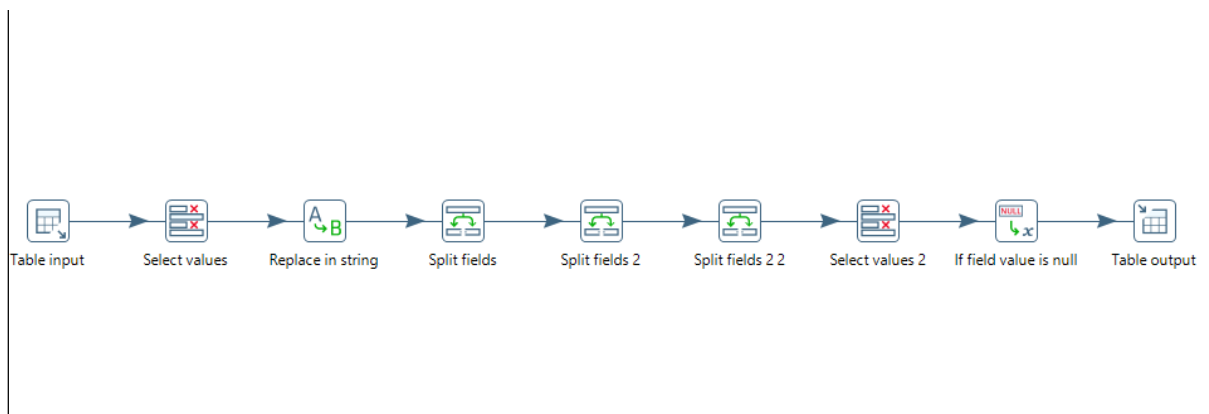
O primeiro *step*, é um *table input*, que é responsável por fazer a conexão e importação dos dados no banco de dados, após essa importação os dados são passados para o *step select values* que seleciona ou exclui valores, atuando ainda em seus metadados, renomeando colunas, modificando tipos, configurando *encoding* e várias outras opções do gênero.

O *step replace in string* atua para retirar caracteres indesejados em quatro campos: qualificação, *n\_vagas*, *salario* e *data\_cadastro*, logo após o *step String Operations*, realiza limpeza de espaço em branco e quebra de linhas nos campos: *n\_vagas*; *salario*; *data\_cadastro* e *qualificação*. Os *steps* de *Data validator* fazem uma verificação de anomalias nos dados, buscando por valores que por algum motivo, não foram extraídos corretamente pelo *Scraping*, após encontrado, a linha toda é retirada do fluxo normal dos dados e salvo em uma tabela auxiliar denominada “*tab\_auxiliar*”.

Por fim é utilizado o *step Select Values* novamente para edição nos metadados dos campos *n\_vagas* e *data\_cadastro*, e *Split Fields* responsável pela divisão do campo *local\_trabalho* em cidade e estado, o delimitador utilizado foi o caracter “/”.

O último *step Table output*, finaliza as transformações enviando de volta para o banco *plus* dentro da *staging área*.

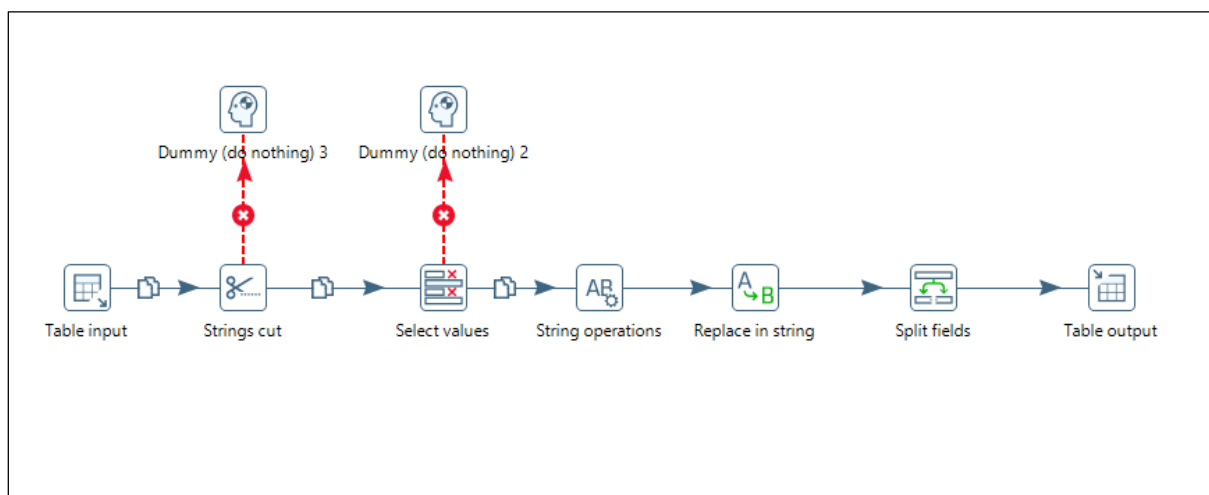
Figura 23: (Data Integration - Kettle) Transformações em *catho.com*



Fonte: O autor

Os dados provenientes do Scraping na plataforma Catho, apresentaram necessidades semelhantes de transformações, e o seu fluxo dentro do pentaho (Figura 23) foi feito com os seguintes *steps*: *Table output*, e *Select Values* com a mesma utilidade da última transformação; *Replace in Strings* excluindo caracteres indesejados nos campos *qtd\_vagas* e *salario*; *Split fields* 1 e 2 para eliminar pedaços indesejados nos campos *data\_cadastro* e *local\_trabalho* e 3 dividindo o campo *local\_trabalho* em dois novos campos, *cidade* e *estado*; *Select Values* elimina três campos auxiliares gerados pelos *steps* anteriores; *Is field value is null* que faz o tratamento de nulos nos campos: *benefícios*, *jornada*, *info\_adicional* e *regime*; e finaliza com *Table output*, enviando o resultado para a tabela “*trans\_catho*” na *staging área*.

Figura 24: (Data Integration - Kettle) Transformações em *empregos.profissionaisti.com*



Fonte: O autor

Dos Steps necessários para as transformações dos dados provenientes da plataforma *empregos.professionaisti.com.br* (Figura 24), apenas o *step String cut*, que realiza o corte dos dois primeiros caracteres no campo *n\_vagas*, não foi utilizado anteriormente, o restante foi utilizado de forma similar ao seu uso nas transformações anteriores.

Figura 25: Estrutura das tabelas após transformações

Tabela	Coluna	Tipo
vagas_empregoscombr	id	INT(11)
	descricao	TEXT
	qualificacao	TEXT
	funcao	VARCHAR(150)
	cidade	VARCHAR(100)
	estado	VARCHAR(2)
	titulo	VARCHAR(150)
	n_vagas	INT(11)
	salario	VARCHAR(50)
	regime_contratacao	VARCHAR(50)
	beneficios	TEXT
	jornada	VARCHAR(200)
	link	VARCHAR(200)
	data_cadastro	DATE
vagas_catho	id	INT(11)
	qtd_vagas	INT(11)
	data_cadastro	VARCHAR(50)
	salario	VARCHAR(50)
	cidade	VARCHAR(50)
	estado	VARCHAR(2)
	descricao	TEXT
	cargo	TEXT
	beneficios	TEXT
	jornada	TEXT
info_adicional	TEXT	
regime	TEXT	
vagas_professionaisti	id	INT(11)
	descricao	TEXT
	regime	VARCHAR(100)
	jornada	VARCHAR(100)
	cidade	VARCHAR(100)
	estado	VARCHAR(2)
	funcao	VARCHAR(100)
	data_cad	DATE
	titulo	VARCHAR(150)
	n_vagas	INT(11)
link	TEXT	

Fonte: O autor

O ETL é a espinha dorsal da arquitetura DW, então seu desempenho e qualidade são extremamente relevantes para a precisão, operabilidade e usabilidade de *Data Warehouses*. Os *steps* de *Table output* utilizados nas transformações criaram tabelas estruturalmente preparadas para a próxima fase do projeto (Figura 25), a classificação e reconhecimento de atributos.

### 1.11 Classificação e reconhecimento de atributos

Essa seção aborda uma etapa muito importante para o trabalho, a etapa de reconhecimento e classificação dos atributos inerentes a cada vaga, presentes nos campos pai.

De uma maneira geral sempre que uma empresa efetua a divulgação de uma vaga em um portal online, o seu objetivo é levar aquela vaga ao alcance de um profissional que contenha o perfil desejado pela empresa e que possua o conhecimento e atributos necessários a vaga.

Como abordado nos capítulos anteriores, todas as plataformas possuem um campo pai, que foi chamado de descrição, onde livremente é descrito as informações sobre a vaga, cada empresa preenche o campo conforme as necessidades e peculiaridades de cada cargo. Ao examinar esse campo percebe-se algumas informações, que mesmo sem uma estruturação, foi notado um padrão de repetição nas vagas divulgadas. Usando como exemplo a vaga divulgada na Figura 26, é possível categorizar diversos atributos como por exemplo os inerentes a *atribuições do cargo*, onde temos: “prestar suporte ao cliente”; “registrar chamados”; “esclarecer dúvidas”; “efetuar a gestão dos chamados”, entre outros. Pode-se também categorizar os atributos referentes ao *perfil e relacionamento interpessoal do candidato*, no exemplo citado temos: “ter foco no cliente”, “bom relacionamento interpessoal”, “trabalho em equipe”, “boa argumentação” e “bom humor”.

Figura 26: (Catho) exemplo de atributos para reconhecimento

#### **Analista de Suporte e Implantação**

A Combinar | Ipatinga - MG (1)

Prestar suporte ao cliente, registrar chamados e esclarecer dúvidas através dos canais. Efetuar a gestão dos chamados e dar feedback ao cliente. Planejar a implantação dos sistemas em novos clientes. Realizar treinamentos e orientar os clientes garantindo a melhor utilização e entendimento dos sistemas. Efetuar a conferência de dados migrados, configuração e manutenção dos sistemas. Mapear as rotinas de negócio do cliente e identificar as principais necessidades de melhorias nos sistemas. Elaborar relatórios e documentações técnicas relativas aos sistemas. Ensino Superior completo ou cursando na área de Tecnologia da Informação: TI, Ciências da Computação, Sistemas de Informação, Engenharia da Informação, ou outros relacionados. Experiência em atendimento a clientes e implantação de sistemas. Conhecimento de Normas e Legislações Públicas. Ter foco no cliente, bom relacionamento interpessoal, trabalho em equipe, boa argumentação, proatividade e bom humor.

Fonte: O Autor

Além dos atributos citados, foi efetuado o reconhecimento e categorização de mais dez atributos designados como campos filhos, esse reconhecimento foi feito utilizando como base todo o conteúdo da vaga, que no momento se encontravam pré-estruturados em campos pais e filhos após ter passado pelo algoritmo de *Scraping* e tratado com o *Kettle*, abaixo uma breve descrição desses atributos:

1. Área de Atuação: Área de atuação dentro do universo da TI, para o projeto essas áreas foram divididas em 8 áreas, exemplo: Desenvolvedor, Infraestrutura e Banco de Dados. Posteriormente será abordado os critérios para classificação dessas áreas, bem como uma descrição individual.
2. Cargo: Descrição resumida do cargo que será exercido, também abordado os critérios para classificação posteriormente.
3. Benefícios: Descrição dos benefícios ofertados pela empresa para o cargo, exemplo: Assistência Médica, convênio com Farmácia, participação nos lucros.
4. Jornada: Período durante o qual o trabalhador está à disposição da sua empresa.
5. Certificações: Declaração formal de comprovação que ateste o nível de conhecimento técnico e a especialização do profissional. Exemplo: Oracle Certified Professional Advanced PL/SQL, Certificação ITIL, CISSP.
6. Ferramentas: Tecnologias e ferramentas utilizadas pelo profissional para realização de determinado fim, pode ser uma linguagem de programação, metodologia, Framework, sistema, entre outros.
7. Atribuições do cargo: Atividades e rotinas que o candidato irá exercer conforme o cargo.
8. Perfil: Relacionamento interpessoal do candidato, refere-se à interação entre pessoas, grupos e times.
9. Escolaridade: Cumprimento de um determinado ciclo de estudos.
10. Experiencia: Descrição de suas qualificações, habilidades, competências e experiências anteriores.
11. Salário: Valor do vencimento oferecido.
12. Regime de contratação: Formas que definem o vínculo empregatício.

Após a escolha da metodologia utilizada para o reconhecimento desses atributos, foi constatado que o melhor caminho seria a criação de um algoritmo próprio para reconhecimento e classificação dos atributos, para isso foi levado em consideração a utilização de uma rede neural, mas para tanto não foi encontrado nenhum modelo disponível para treinamento da rede.

O método utilizado para o reconhecimento foi a comparação de strings, com a utilização de “bag of words”, em tradução literal “saco de palavras”.



Para cada atributo foi criado inúmeros *tokens* – palavras ou sentenças que representa determinado atributo – e importados em uma lista, que iterando sobre o texto de entrada (campo pai), realiza a busca dos *tokens* dentro do campo recebido, e ao fazer o reconhecimento desse *token*, faz a inserção do mesmo na tabela que o relaciona com a vaga em questão. Criando assim um relacionamento um para muitos. Na sequência e exemplificação do reconhecimento de cada atributo.

### 1.18 Classificações e reconhecimento dos Cargos e áreas

De modo geral, a Tecnologia da Informação é uma área da Informática que envolve elementos como *hardware*, *software*, redes e bancos de dados, por ser um setor em constante evolução e com uma grande necessidade de especialistas, são várias as áreas em que um profissional do ramo pode atuar.

Para melhor visualização e categorização, foi efetuada uma classificação das áreas de atuação em oito categorias: Desenvolvimento de *Software*, Gestão/Projetos/Comercial, Big Data/I.A./Data Science, Banco de Dados, Suport/Help Desk/Telecon, Infraestrutura/Redes/Cloud, Segurança Da Informação e Marketing Digital/E-commerce. Para cada uma dessas categoria foi indexado os cargos a ela pertencentes, essa indexação ocorre no método “reconhece\_cargo” e pertence ao processo de reconhecimento de cargos.

O Quadro 2 retrata a quantidade de cargos categorizados para cada área. A listagem dos cargos classificados bem como a relação com a área está exposta nos anexos 15 a 21.

Quadro 2: Quantidade de cargos categorizados

Área	Quantidade
Desenvolvimento de Software	17
Gestão/Projetos/Comercial	23
Big Data/I.A./Data Science	10
Banco de Dados	8
Suport/Help Desk/Telecon	10
Infraestrutura/Redes/Cloud	21
Segurança Da Informação	4
Marketing Digital/E-commerce	6
Total	99

Fonte: o autor

Para a categorização do cargo de cada vaga, foi utilizado o campo filho proveniente do *Scraping* Título/Cargo, esse campo é inserido no código-fonte em forma de *string* (cadeia ou sequência de caracteres), e submetido a um método que verifica a existência de cada *token* do “bag of words” nessa *string*, de modo que, se algum *token* for encontrado, a função retorna *true* imediatamente finalizando a sua iteração nos *tokens*.

Para criação dos *tokens* dentro da “bag of words”, foi feita uma análise de termos e sentenças relacionados a cada cargo categorizado. Na Figura 27 é visualizado um exemplo de “bag of words” com os *tokens* responsáveis pelo reconhecimento dos cargos de Analista de Sistemas e Desenvolvedor. No total foram criados 862 *tokens*.

Figura 27: Exemplo de criação de *tokens*

```
def analista_sistemas(titulo):
    tokens = ["analista de sistema", "analista de sistemas", "analista de desenvolvimento", "analista desenvolvedor",
             "analista desenvolvedor", "analista de dot net", "analista de dotnet", "analista .net",
             "analista java", "analista back-end", "analista backend", "análise e programação", "analise e programação",
             "especialista ios", "analista sap", "analista desenvolver", "consultor sistemas", "analista asp net",
             "front end react", "analistas de sistemas", "analista full stack", "ou fullstack", "analista mobile",
             "analista c#", "analista fullstack", "analista de front end", "analista de frontend", "analista jr",
             "analista junior", "analista de software", "front-end junior", "front-end pleno", "fullstack web",
             "consultor de sistema", "técnico php", "estágio em apis", "analista node.", "analista programador java"]

    for token in tokens:
        if token in titulo:
            return 1
    return 0

def desenvolvedor(titulo):
    tokens = ["desenvolvedor", "programador", "developer", "dev", "desenvolvedor(a)",
             "desenvolvimento ti", "desenvolvimento", "desevolvedor asp.net", "desenvolvedor .net", "desenvolvedor aps",
             "java back-end", "java backend", "php - back end", "programação java", "fullstack php", "ms visual",
             "estagiário(a) de php", "estagiário de php", "c# pleno", "programação em c#", "programaador php",
             "programação php", "back-end -", "full stack ruby", "desenvolvedor java", "codificadores javascript", "java web",
             "programação c#", "desenvolvedor front end", "back end sênior", "desenvolver .net", "c# / node",
             "front-end -", "especialista android", "java especialista"]

    for token in tokens:
        if token in titulo:
            return 1
    return 0
```

Fonte: O autor

Como cada cargo está indexado a área que ele pertence, então ao efetuar o reconhecimento do cargo, eventualmente se obtém a área de atuação.

### 1.19 Reconhecimento Atribuições dos Cargos, ferramentas, certificações, benefícios e perfil.

Para esses cinco atributos, a relação do relacionamento é de um para muitos, ou seja, cada vaga pode conter zero ou várias ocorrências de cada atributos. Como por exemplo as ferramentas, uma vaga de desenvolvedor pode especificar o uso de várias linguagens de programação, como Delphi e Python.

Para a criação dos *tokens* de reconhecimentos desses atributos, foi desenvolvida uma abordagem semelhante ao reconhecimento dos cargos e áreas, porém como o número de vagas extraídas foi relativamente grande, e a captura desses atributos foi executada de forma manual, foi separada uma amostragem de 10% das vagas de cada cargo, e observando cada vaga foi retirada e categorizada cada *token*. No Quadro 2 **Erro! Fonte de referência não encontrada.** e exposto a quantidade de *tokens* reconhecido e persistidos para cada atributo. Nos apêndices 23 a 26 são listados cada um dos 1024 *tokens* encontrados.

Quadro 2: Número de *tokens* categorizados.

Atributo	Quantidade
Atribuições do cargo	142
Ferramentas	132
Certificações	632
Benefícios	20
Perfil	142
Total	1024

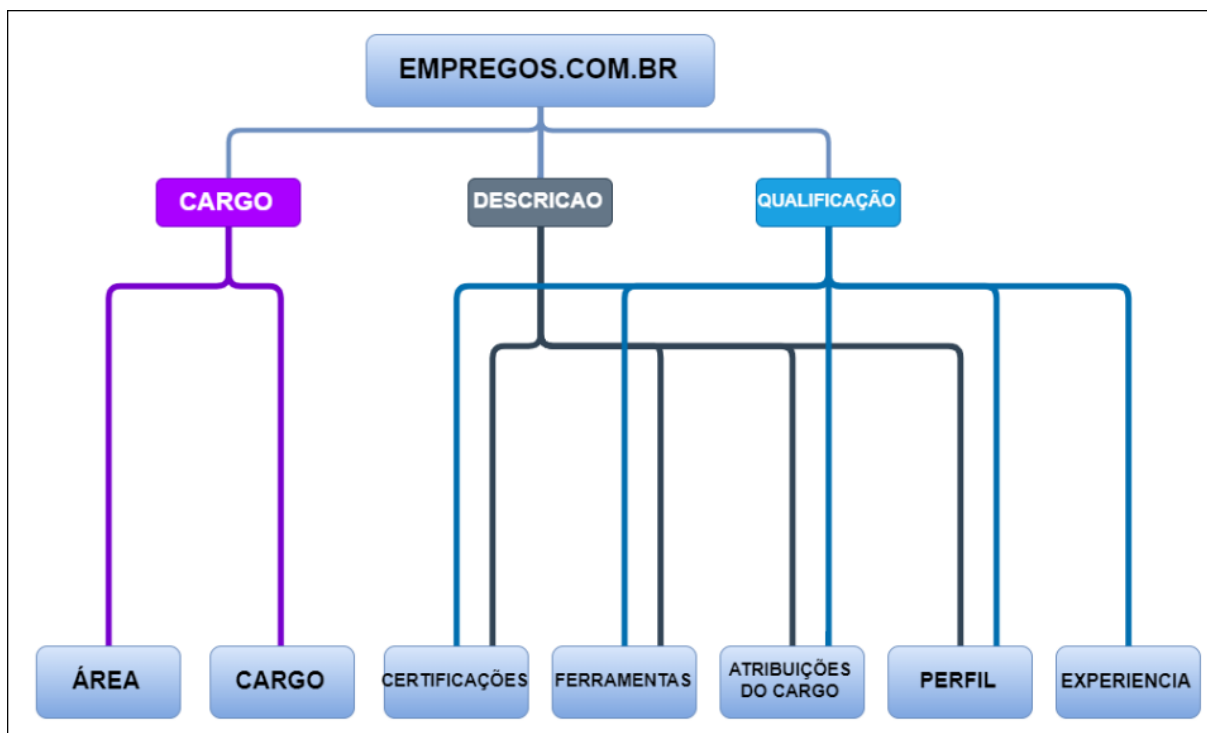
Fonte: O autor

O funcionamento do algoritmo de classificação segue a mesma lógica do anteriormente descrito, por intermédio da comparação de *strings*, é passada a *string* correspondente ao método de reconhecimento, e ele verifica a presença de cada *token* naquela *string* e caso esteja presente, adiciona-se o *token* na tabela de relação correlacionando com a tabela vaga.

Importante ressaltar que para cada plataforma existe um esquema diferente para reconhecimento dos atributos em campos pai, para tanto se fez necessário um mapeamento do fluxo de dados de acordo com o conteúdo que cada campo pai possui.

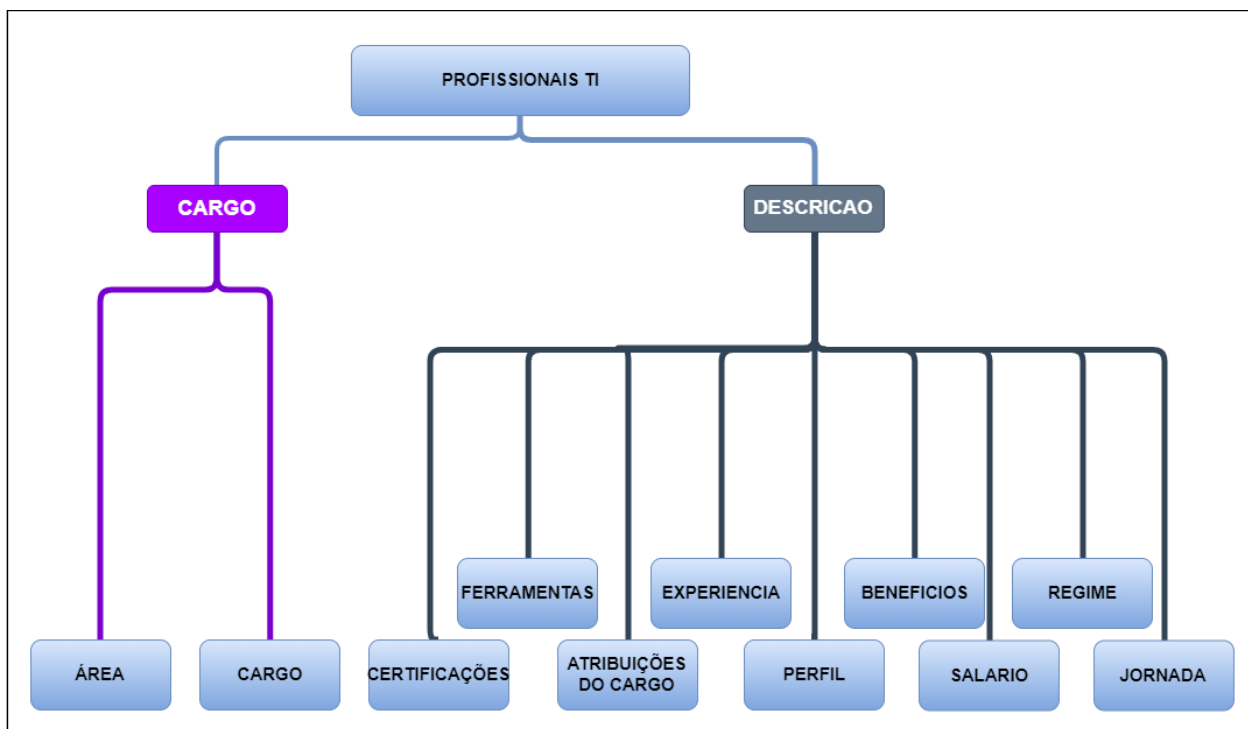
No site *empregos.com.br*, é apresentado três campos pai (Figura 28), o campo cargo, que deriva para área e cargo, o campo descrição, que reconhece itens de certificações, ferramentas, atribuições do cargo e perfil, e o campo qualificação que pode conter os mesmos itens do campo descrição e ainda experiência.

Figura 28: Derivação dos campos pai em *empregos.com.br*



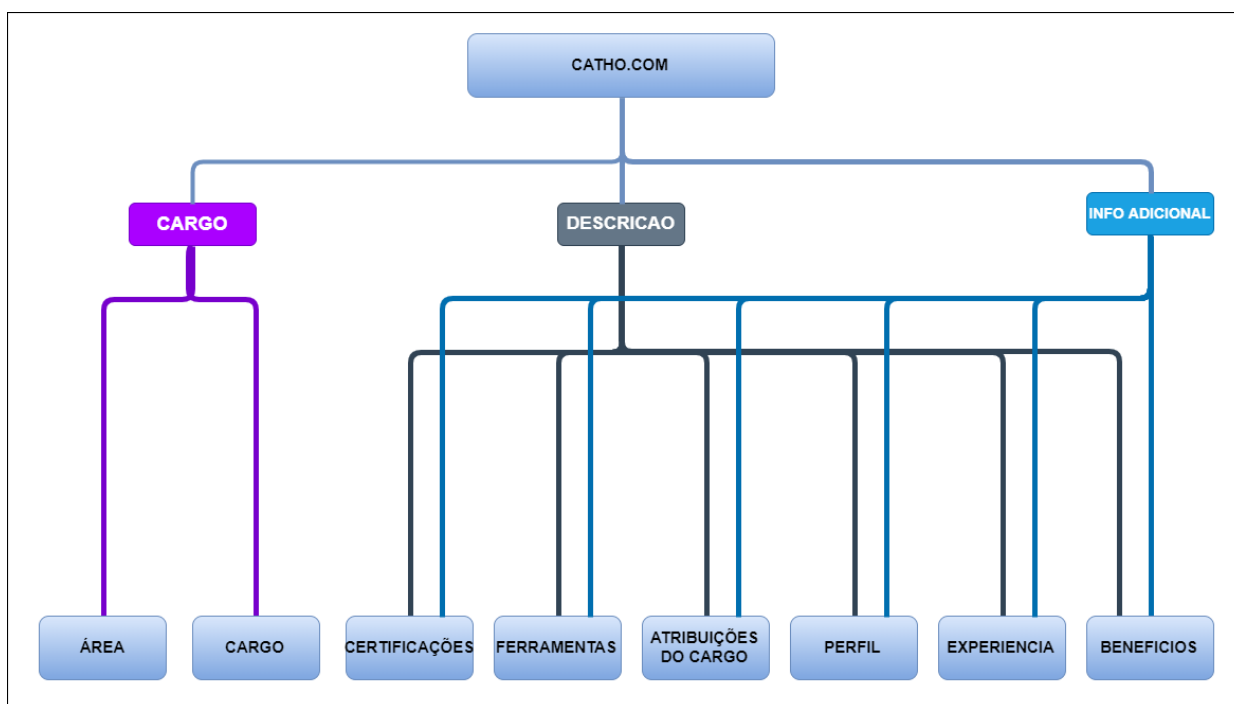
Fonte: O Autor

Já na plataforma *empregos.profissionaisti.com.br* (Figura 29), o campo pai descrição é responsável por fazer a maior parte das derivações, através dele se reconhece itens de certificações, ferramentas, atribuições do cargo, perfil e experiência. Também presente, o campo pai cargo deriva para área e cargo.

Figura 29: Derivação dos campos pai em *empregos.professionaisti.com.br*

Fonte: O Autor

No site *catho.com* (Figura 30) os campos pai descrição e info adicional, podem derivar certificações, ferramentas, atribuições do cargo, perfil, experiência e benefícios, porém em menor número no campo Informações adicionais.

Figura 30: Derivação dos campos pai em *catho.com*

Fonte: O autor

### 1.20 Reconhecimento Jornada, regime, cidade e estado.

Os atributos jornada, regime, cidade e estado, são campos filhos, e no geral são atributos que não existem muitas variações, e como nas plataformas essas informações já estavam dispostas de maneira isolada, isto é, estavam fora de um campo pai, contendo um campo dedicado para cada um, não foi necessário a criação de *tokens* para reconhecimento na *string*.

O trecho do algoritmo responsável pela sua categorização e inserção no banco é descrita da seguinte maneira:

1. Verifica se o item já existe no banco.
2. Caso exista retorna o id.
3. Caso não exista, realiza a inserção e retorna o id.
4. Insere novo relacionamento na tabela atributo-vaga.

### 1.21 Atributos Salário e número de vagas

Os atributos salário e vagas, são atributos quantitativos e campos filhos, portanto não houve a necessidade de submetê-los ao algoritmo de classificação e reconhecimento.

Após o processo de classificação, os dados estão prontos para serem carregados no DW, para isso é necessário a definição do modelo dimensional, bem como a relação entre as tabelas. Ao efetuar o reconhecimento e classificação, o algoritmo devolve os dados para a *staging* área, de maneira a prepará-los para o modelo dimensional, como será visto na sequência.

## 1.12 Implementação do Data Mart

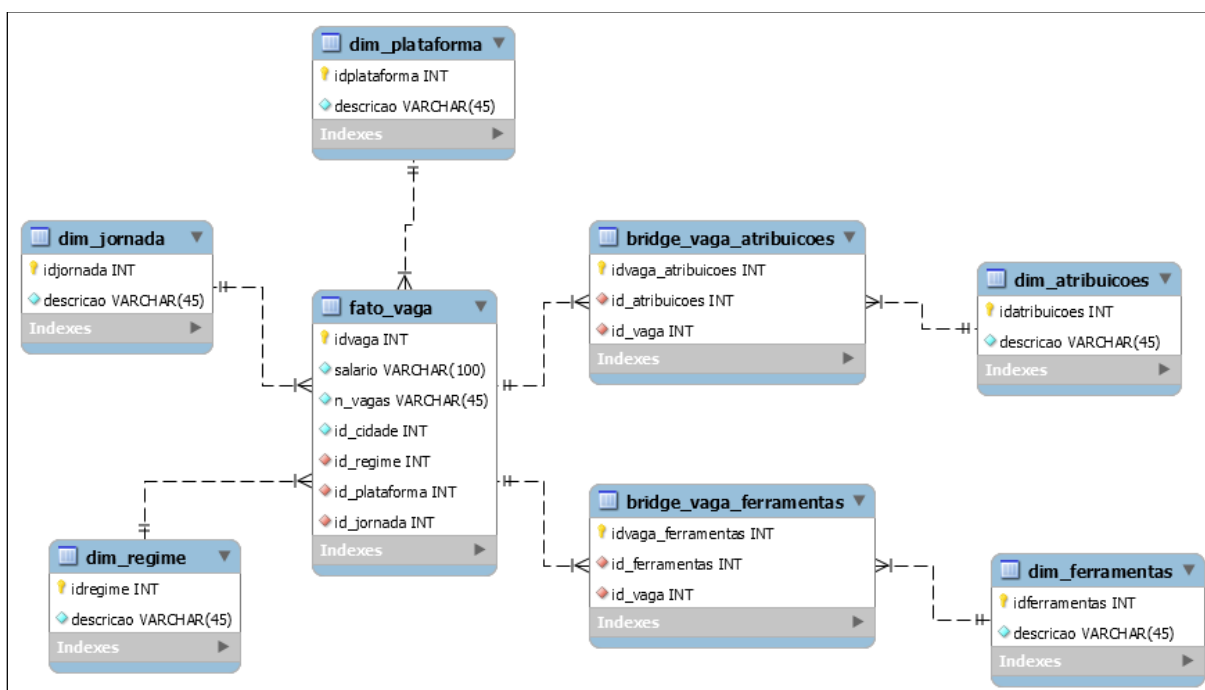
Com os dados provenientes do passo de classificação, é construído o DW, que é o artefato final do processo de ETL. Para a busca das informações que porventura estariam contidas de maneira implícita no banco de dados, é no DW que serão carregados os dados das vagas e realizadas as consultas.

Primeiramente foi definido como seria o modelo dimensional e quais informações de cada tabela seriam utilizadas para a implementação de suas dimensões, posteriormente, definiu-se como seria a relação entre as tabelas no DW, além da criação de uma nova tabela, que reúne em si as chaves primárias de todas as outras e seus dados quantificáveis, conhecida como Tabela Fato, ela é considerada a principal tabela de um modelo dimensional (KIMBALL, 2008).

### 1.22 Modelo Dimensional

Para a modelagem dimensional do DW, foi escolhida a modelagem descrito por KIMBALL (1998) como *Snow flake*.

Uma dimensão é dita estar snowflaked quando segundo KIMBALL (1998), os campos de baixa cardinalidade de uma dimensão compuseram uma outra tabela, a qual foi ligada com a tabela original com chaves artificiais. Um exemplo disso se mostra na Figura 31.

Figura 31: Modelagem *Snow Flake*

Fonte: O autor

A matriz dimensional é composta, essencialmente, por dois tipos de tabelas: as de fato e as de dimensão. A tabela de fato é um grande inventário central, composto basicamente pelas ocorrências do negócio, por exemplo: de vagas ofertadas. Já a tabela de dimensão armazena basicamente os atributos do negócio, como dados sobre as ferramentas utilizadas, cargo (cargo e área de atuação) e localização (cidade e estado). As tabelas fatos normalmente apresentam um grande volume de dados, enquanto as tabelas de dimensões um volume comparativamente menor.

### 1.23 Escolha das dimensões

As dimensões do *Data Mart*, serão todas as tabelas de atributos referente as vagas (Figura 32), e servirão como filtros nas consultas, descrevendo o fato.

Cada dimensão é definida com uma única chave primária. Essa chave é a base da integridade referencial no relacionamento com a tabela de fatos. Cada dimensão será composta também de um atributo, textual e distinto.



Figura 32: Definição das dimensões



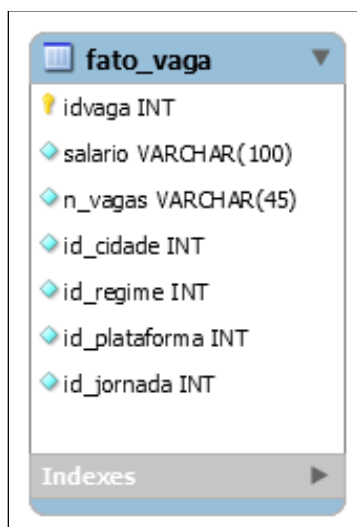
Fonte: O autor

#### 1.24 Criação da tabela fato

A tabela de fatos é a principal tabela de um modelo dimensional, onde as medições numéricas de interesse estão armazenadas (KIMBALL, 2002).

A tabela de fatos irá registrar os fatos que serão analisados. Foi composta por uma chave primária (formada por uma combinação única de valores de chaves de dimensão) e pelas métricas quantitativas “salario” e “número de vagas”. De maneira simplista, a tabela fato é responsável por fazer a ligação com as dimensões.

Figura 33: Fato vaga



Fonte: O autor

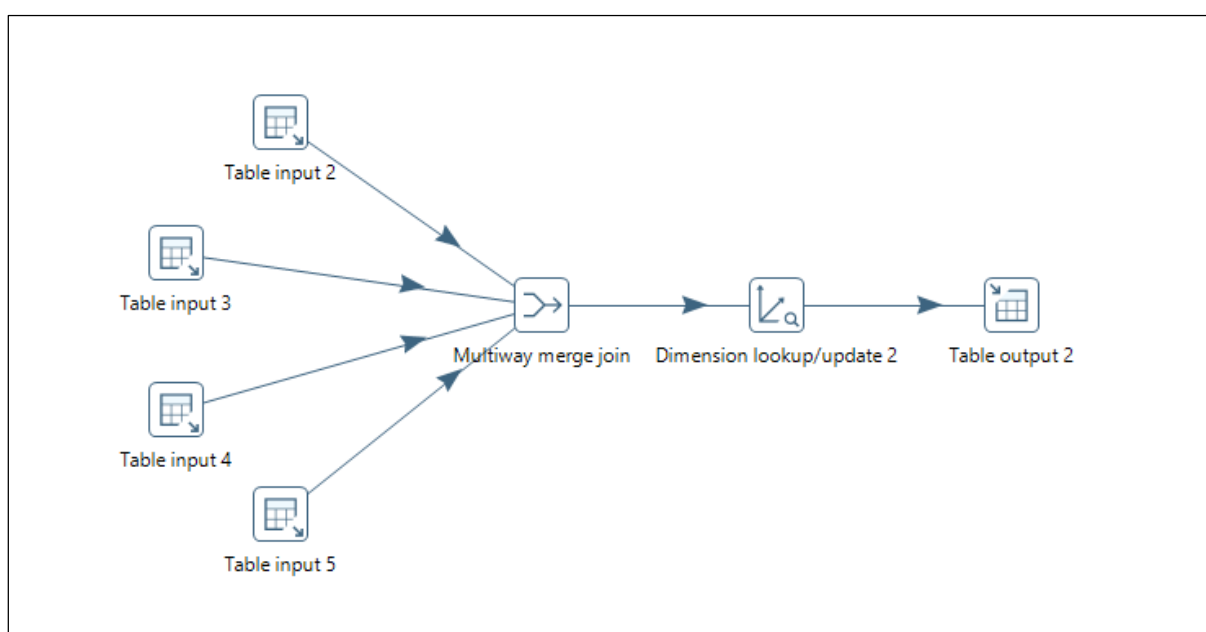
A Figura 33 mostra, a estrutura da tabela fato vaga, para implementação física do *Data Mart*.

### 1.25 Implementação Física

A criação da tabela fato e das dimensões bem como as cargas para o *Data Mart*, foram feitas novamente utilizando o PDI. O armazenamento do *Data Mart* ficou a cargo do sbgd MySQL com o *Workbench* onde se criou uma nova base de dados com o nome plus\_DW.

Primeiramente, foi realizada a conexão e importação dos dados da dimensão utilizando o componente *Table Input*, que carrega todos os dados de origem para o *step Dimension Lookup / Update*, onde foram feitas as configurações de acordo com as necessidades do sistema, como por exemplo a criação da *Surrogate key*. Posteriormente, os dados foram gravados em seu respectivo destino (*Data Mart*), esse procedimento foi replicado para cada uma das dimensões.

Figura 34: Criação do fato vaga.

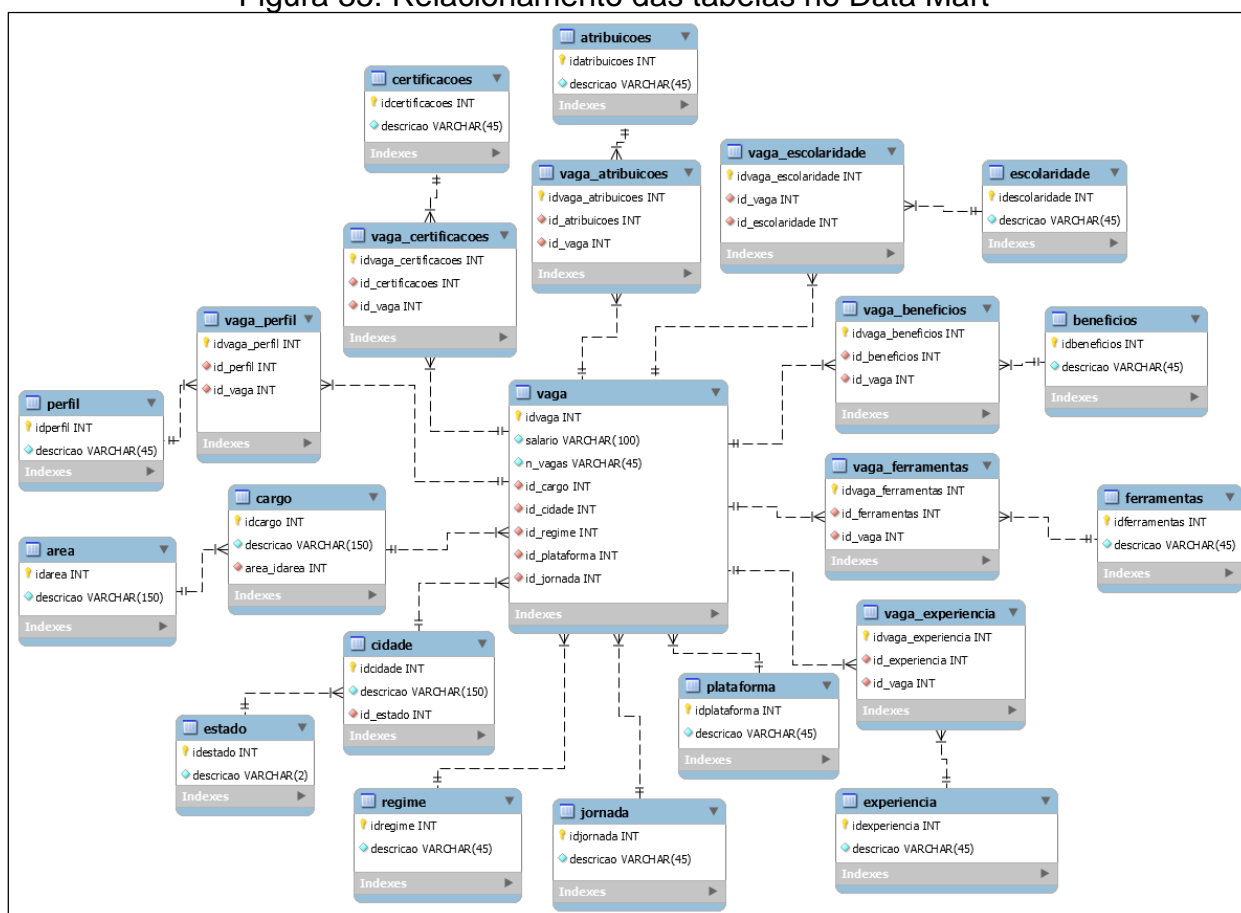


Fonte: O autor

A tabela fato irá conter o registro de cada ocorrência de vagas disponibilizada, para sua criação, foram definidos os dados quantificáveis que estariam presentes, definidos pelos campos salário e número de vagas, posteriormente, se iniciou o processo de montagem da tabela, onde foi adicionado o componente *Table Input* para a entrada de dados de cada registro da tabela vagas, seguido pelo componente *Multiway Merge Join*, responsável por realizar a junção dessas tabelas através das suas respectivas chaves. Continuando o processo, foram inseridos os componentes *Dimension Lookup / Update*, responsável por jogar para a tabela fato as chaves que vem de cada dimensão. Por último, todos os dados foram carregados na tabela fato através do componente *Table Output*. Esse processo pode ser visto na Figura 34.

Após a criação do modelo dimensional e a carga dos dados no *Data Mart*, a próxima etapa será a confecção dos relatórios e análises. Para melhor visualização desses recursos, o *Data Mart* criado foi carregado e interpretado pelo Microsoft Power BI.

Figura 35: Relacionamento das tabelas no Data Mart



Fonte: O autor

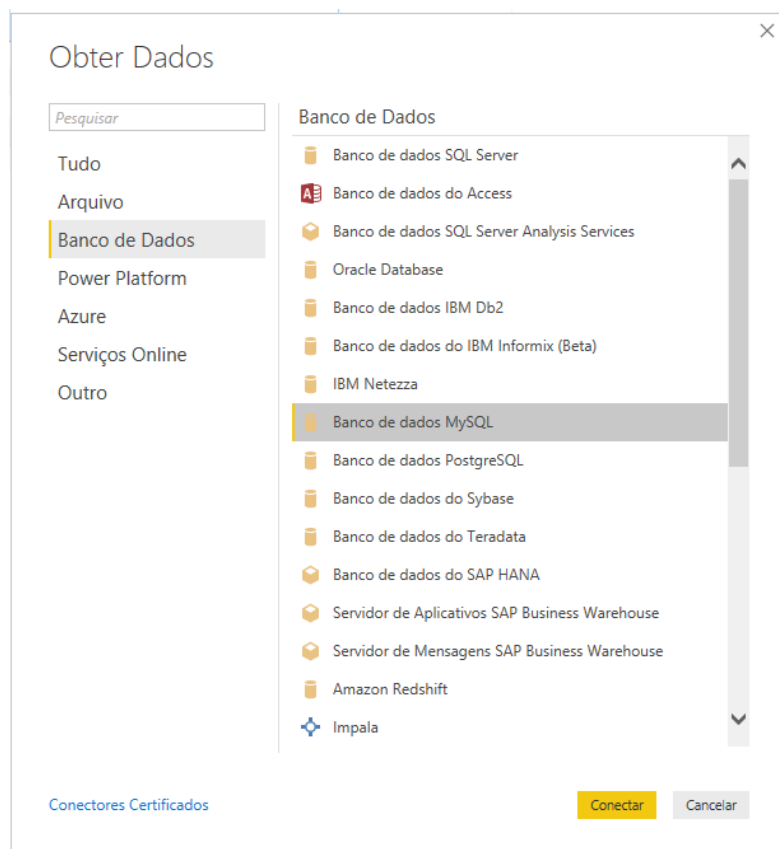
A Figura 35, mostra a estrutura dos dados após a modelação dimensional *Snow Flake*, nesse momento os dados estão prontos para a carga no *Data Mart*.

### 1.13 Power Bi

Com os dados modelados e armazenados no *Data Mart*, pode-se passar para a última etapa do projeto, a criação de relatórios e gráficos, para essa etapa foi utilizado o Microsoft Power Bi Desktop versão 2.75.5649.582.

A conexão do DM com o Power Bi é bem simples, por já contar nativamente com um conector MySQL, e com uma interface intuitiva, em poucos segundos o Power Bi faz a conexão e importação dos dados para o *Power Query* (Figura 36).

Figura 36: Conexão Power Bi com base de dado



Fonte: O autor

Após a conexão com o *Data Mart*, foi realizado o cruzamento dos dados para obtenção dos relatórios, o próximo capítulo irá abordar os resultados obtidos com o uso das ferramentas aplicadas no projeto, e também será apresentado o produto final do trabalho, a análise das ferramentas e técnicas utilizadas para análise das vagas ofertadas na web e seus atributos.

## 4 APRESENTAÇÃO DOS RESULTADOS

A apreciação da proposta apresentada, foi feita em cima dos métodos e técnicas abordados no capítulo anterior, primeiramente serão informados os resultados provenientes das etapas de extração da informação, os passos do processo de ETL são descritos logo após, na sequência são apresentados os resultados do Algoritmo de classificação e reconhecimento de atributos, por fim é feita uma análise sobre os dados inseridos no DW, bem como os relatórios que podem ser gerados pelo mesmo.

### 1.14 Crawler/Scraping

Com utilização da biblioteca *Beautifulsoup 4*, ganhou-se em agilidade e simplicidade na codificação, bem como em suporte a documentação e comunidade atuante, a sua simplicidade facilitou a construção do *Web Crawler* e *Web Scraping*. A opção pelo seu uso, possibilita a extração de dados da Web bastando, apenas, conhecimentos de *HTML* e da linguagem Python. Porém, o seu uso não se mostrou efetivo em sites com a técnica de rolagem infinita.

Ao executado, o Algoritmo de *Crawler*, extraiu com sucesso 83,86% das URL's de vagas publicadas pelas plataformas. Somadas as três plataformas de estudo dispunham de 37.084 vagas, dispostas em páginas com 10, 15 e 10 URL's de vagas respectivamente em empregos.com, catho.com. e profissionaisti.com.

Como o Algoritmo estava configurado para não exceder o tempo de 3 segundos de resposta em cada requisição enviada ao site, para assim não o sobrecarregar de requisições e derrubá-lo, houve uma perda de 5.984 URL's que estavam em páginas cujo seu tempo de resposta excedeu esse limite.

Em âmbito geral cada requisição teve uma média de tempo de execução (período total em que cada iteração do objeto vaga levou para executar seu loop no algoritmo) de 0,066 segundos, esse tempo de resposta baixo foi possível graças a natureza objetiva e simples do algoritmo, 07 minutos e 26 segundos foi o tempo necessário para que fosse finalizado o processo em todas as três plataformas. O Quadro 3 descreve os resultados em cada plataforma, para o registro do tempo de execução de todo o projeto de software, foi adicionado ao método correspondente uma função que contabiliza o tempo total de execução.

Quadro 3 : Resultados etapa de Crawler

Plataforma	Entrada (Em páginas)	Saída (Url de vagas)	Tempo médio (Em segundos)	Tempo Total (Em Minutos)
empregos.com.br	1.121	11.432	0,057	01,04
catho.com	1.083	16.258	0,044	00,48
profissionaisti.com.br	3.410	3.410	0,098	05,34
Total	5.614	31.100	0,066	07,26

Fonte: O autor

Devido uma estruturação do código-fonte um pouco maior, na etapa seguinte de *Scraping*, pode-se observar um tempo médio de execução maior para cada URL de vaga, 0,071 segundos, esse tempo médio ficou dentro do esperado, tendo em vista que sua execução foi incrementada com mais processos em relação ao *Crawler*. Levaram 35 minutos e 36 segundos para que todas as 31.100 URL'S de vagas fossem expostas ao Algoritmo.

A sua taxa de URL'S extraídas com sucesso também ficou dentro do esperado, do total de URL'S submetidas, se obteve êxito em 23.799 delas, gerando uma taxa onde 76,52% das URL's foram extraídas com sucesso.

Quadro 4: Resultados etapa de Scraping

Plataforma	Entrada (Url vagas)	Saída (Url vagas)	Tempo médio (Em segundos)	Tempo Total (Em Minutos)
empregos.com.br	11.432	8.449	0,078	14,52
catho.com	16.258	12.084	0,061	16,32
profissionaisti.com.br	3.410	3.266	0,074	04,12
Total	31.100	23.799	0,071	35,36

Fonte: O autor

O Quadro 4 descreve o processo, expondo o número de URL's que foram expostas ao algoritmo, o quantidade de URL's que o algoritmo conseguiu extrair, bem como o tempo médio gasto em cada URL e também o tempo total.

### 1.15 Pentaho Data Integration – Kettle (PDI)

Após a etapa de transformações pode-se concluir que o PDI é uma poderosa ferramenta de integração e transformação, altamente personalizável e que recebe atualizações constantes de seus desenvolvedores, além de oferecer inúmeros conectores poderosos, a sua versão de código-fonte aberto possui todas as ferramentas necessárias para a limpeza dos dados do projeto.

À medida que as funcionalidades necessárias para as transformações foram mais usadas, a sua usabilidade ficou mais inteligíveis. Esta curva de aprendizado foi crescente, mas de inclinação suave, pois apesar da ferramenta ser muito intuitiva, possui uma enorme variedade de *Steps* e funcionalidades.

O uso do Pentaho Data Integration para os processos de transformação e modelagem dimensional, trouxe maior flexibilidade, menor tempo de desenvolvimento e melhor estruturação para tarefas como as discutidas neste artigo.

Após a carga do dataset para limpeza, foram encontradas diversas inconsistências oriundas do Scraping, como por exemplo: Linhas que não possuíam dados, ou seus elementos não eram suficientes para análise, essas linhas que apresentaram dados fora do padrão foram excluídas do montante, de 23.799 vagas submetidas a limpeza, 407 não foram aproveitadas.



### 1.16 Classificação e reconhecimento de Atributos

Com a utilização da linguagem python, a agilidade no desenvolvimento do código-fonte foi relevante. Como o projeto foi dividido em módulos, a sua manutenção foi clara e descomplicada. Já o tempo total de execução do algoritmo, foi bem acima dos algoritmos anteriormente citados *Crawler* e *Scraping*. Muito em virtude dos complexos métodos de reconhecimento, onde em cada iteração dos objetos vaga, os *tokens* de reconhecimento são expostos há milhares de comparações de termos, também agrega ao tempo final, os segundos necessários para a conexão e inserção da informação no banco de dados.

O algoritmo foi configurado para a execução em lotes, para que em caso de alguma adversidade, não ocorresse a perda de dados já processados.

Quadro 5: Resultados etapa de classificação

Plataforma	Entrada (Objeto vaga)	Tempo médio (Em segundos)	Tempo Total
empregos.com.br	6.120	1,623	2:45:33
catho.com	8.552	1,987	4:43:13
profissionaisti.com.br	2.720	1,554	1:10:27
Total	17.392	1,721	8:39:12

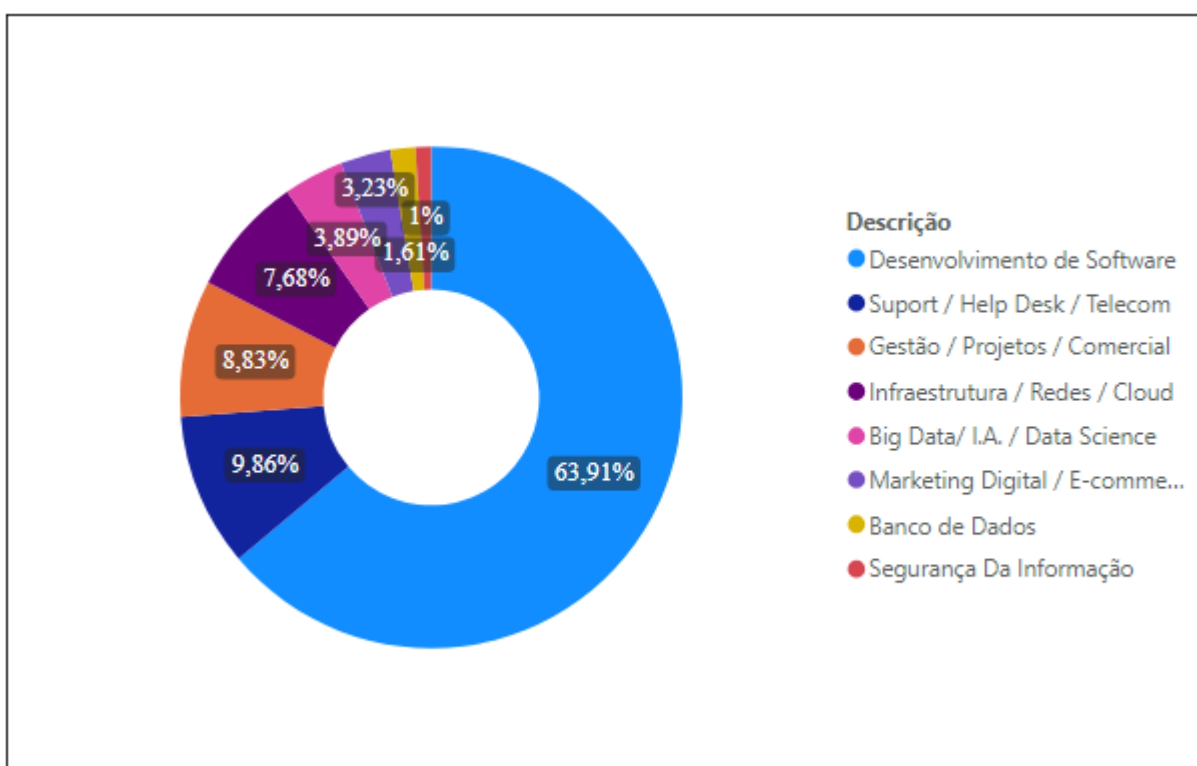
Fonte: O autor

O Quadro 5, descreve as informações sobre o tempo de execução do algoritmo. O tempo médio de execução de cada objeto vaga, foi de 0,721 segundos, esse tempo médio ficou dentro do esperado, tendo em vista a complexidade de seu código-fonte e o elevado número de processos. Levaram 8 horas 39 minutos e 12 segundos para que todos os 17.392 objetos, fossem expostos ao algoritmo.

Como resultados da etapa de classificação e reconhecimento de atributos, é apresentado o número de atributos reconhecidos, das categorias: Área e cargo; atribuições do cargo; ferramentas; certificações; Perfil; e benefícios.

Foram processadas com êxito todas as 17.392 vagas oriundas do tratamento, o algoritmo de classificação foi capaz de fazer o reconhecimento e classificação sem perder nenhuma vaga. Inicialmente foi feita a classificação de cada vaga quanto ao seu cargo, e a que área pertencia. No Gráfico 3, fica explícito o amplo predomínio das vagas para desenvolvimento de software, revelando a escassez de profissionais qualificados e o aumento da demanda em função da capacidade de formação dos mesmos.

Gráfico 3: Porcentagem de classificação por área



Fonte: O autor

Bastante satisfatório, o reconhecimento das ferramentas obteve 40.362 ocorrências em 17.392 vagas submetidas, porém desse montante de vagas, 5.632 não teve ferramentas reconhecidas, desse modo se faz necessário a criação de mais tokens de ferramentas. Portanto, pode-se dizer que 3,43 é a média de ferramentas solicitadas por vagas, tendo como critério somente as vagas que descreveram pelo menos uma ferramenta. Na Figura 37 é apresentado as dez ferramentas que mais apareceram como requisitos para candidatos.

Figura 37: Top 10 Ferramentas e quantidade de ocorrências

Descrição	Número de ocorrências
SQL	5762
Java	5587
JavaScript	3124
CSS	2843
SQL Server	2028
C#	2008
angular	1973
PHP	1538
Scrum	1483
APIs	966
<b>Total</b>	<b>27312</b>

Fonte: O autor

O algoritmo foi capaz de reconhecer 20.316 atribuições do cargo, a média de 1,16 atribuições do cargo por vaga, foi bastante insatisfatória, no entanto justificável pelo fato do processo de criação dos *tokens* ter sido feito sob uma análise de apenas 5% do montante geral das vagas e como resultado apenas 141 *tokens* de atribuições do cargo.

Dos 19 *tokens* de benefícios, foram reconhecidas 35.863 ocorrências (Figura 38), média de 2,06 benefícios por vaga submetida ao reconhecimento, porém ao refinar o processo de análise, verificou-se que apenas 23,17% das vagas teve um ou mais benefícios reconhecidos, ou 8.311 vagas. Portanto a média de benefícios levando em consideração apenas as vagas que tiveram algum benefício reconhecido ficou em 4,3. Um número bastante satisfatório e próximo da realidade.

Figura 38: Número de benefícios reconhecidos

Descrição	Número de Ocorrências
Vale Transporte	7281
Assistência Médica	5810
Assistência Odontológica	4465
Tíquete Refeição	3528
Seguro de Vida	3445
Medicina em grupo	3321
Tíquete Alimentação	1369
Participação nos lucros	1346
Auxílio Creche	950
Estacionamento	754
Cesta Básica	728
Restaurante na empresa	675
Convênio com Farmácia	653
Ambiente Informal	466
Horário Flexível	458
Plano de Carreira	353
Celular fornecido pela empresa	142
Transporte Fornecido pela empresa	108
meritocracia	11
<b>Total</b>	<b>35863</b>

Fonte: O autor

Foram reconhecidos 5.372 *tokens* de perfil em 3.498 das 17.392 vagas submetidas ao algoritmo. Levando em consideração apenas as vagas com ocorrências, a média de 1,53 *tokens* de perfil por vaga condiz com a realidade, onde pode-se perceber um menor foco nesse quesito.

Figura 39: Top 10 perfil e quantidades reconhecidas

Descricao	Número de Ocorrências
racional	522
organização	483
inovador	292
Trabalho em equipe	284
Liderança	236
Inovação	235
proatividade	207
eficiente	188
proativo	181
determinado	158
<b>Total</b>	<b>2786</b>

Fonte: O autor

É apresentado na Figura 39 os dez perfis com maior número de ocorrências.

O número de certificações reconhecidas, foi excepcionalmente baixo, 92 ocorrências no total, isso ocorreu pelo fato de que a descrição de cada certificação ser bem composta, “MB2-715: Microsoft Dynamics 365 customer engagement Online Deployment” é um exemplo, e na descrição das vagas os ofertantes colocarem como “MB2 – Microsoft 365”, portanto ficou identificado a necessidade da criação de *tokens* mais elaborados.

### 1.17 Implementação do Data Mart e carga

A modelagem multidimensional dos dados foi fundamental na construção do Data Mart, com a utilização do modelo *Snow Flake* foi possível obter-se um desempenho superior, no que se refere às consultas e análise de grandes volumes de dados. Pois esse modelo permite, que as consultas sejam realizadas de maneira mais intuitiva e flexível.

A sua composição fornece uma estrutura desnormalizada, e mais adequada a consultas e exploração de informações, realizando a visualização e cruzamento dos dados, de maneira simples.

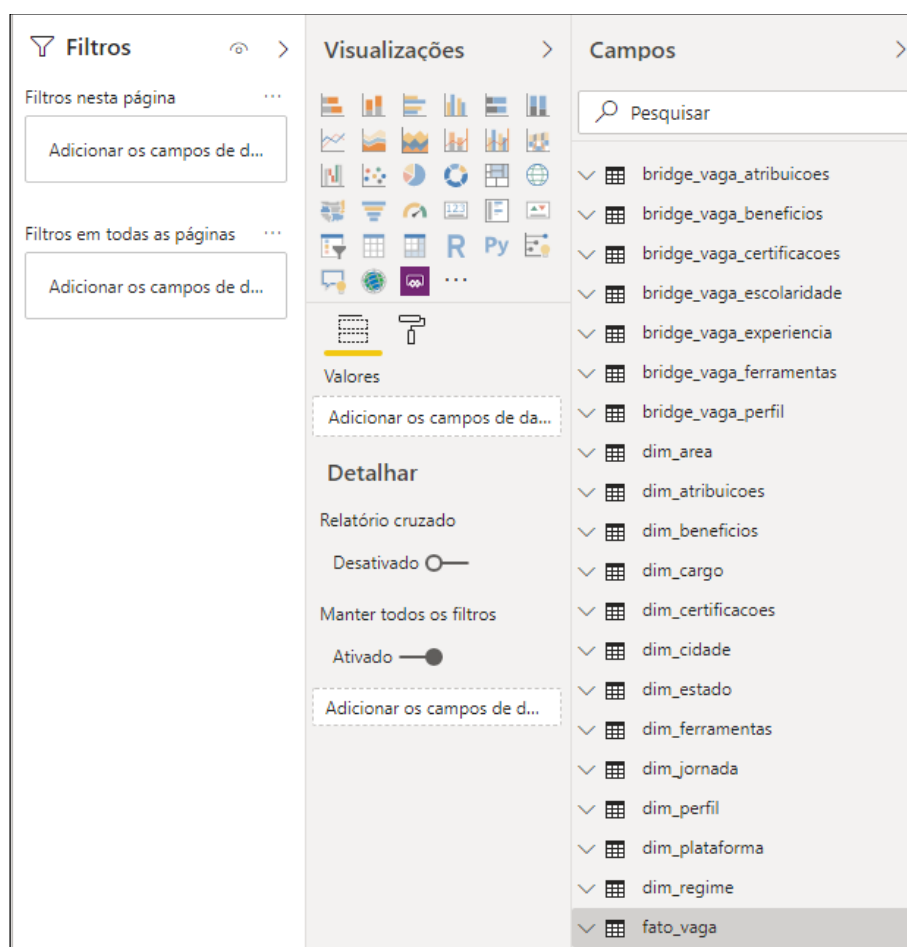
Os esforços empenhados nesse trabalho, foram imensamente recompensados, já que foi possível desenvolver todas as etapas de construção de um *Data Warehouse*, contribuindo com o discente para com a sua experiência acadêmica.

### 1.18 Geração dos Relatórios

A geração dos relatórios se dá com a utilização do *Microsoft Power BI*, o uso da ferramenta foi onde as informações são transformadas em conhecimento.

O sistema é simples e convencional, e por se tratar de uma ferramenta de *self service BI* facilitou a sua usabilidade, a sua comunicação com o Data Mart, foi rápida e eficaz, após a carga do *Data Mart*, a tabela fato e as dimensões são apresentadas através de campos (Figura 40), e todo o cruzamento e geração dos relatórios foi feito no modelo “arrasta e solta”.

Figura 40: Carga do Data Mart no Power Bi



Fonte: O autor

Para exemplificar uma análise, foi utilizado o seguinte caso: um relatório para verificar quais são as dez ferramentas mais solicitadas para o cargo de Desenvolvedor, para tal foi selecionado na dimensão `dim_cargo` o campo descrição, cruzando com o campo `id_vaga_ferramenta` da dimensão `bridge_vaga_ferramenta`, que retorna o número de ocorrências desse atributo no fato vaga, para filtrar por cargo bastou acrescentar novamente o campo descrição da dimensão `dim_cargo` em filtros.

Dessa maneira pode-se obter as ferramentas mais utilizadas para o cargo de desenvolvedor, e para filtrar apenas as 10 ferramentas com maiores ocorrências bastou-se configurar um campo presente na aba filtro. O resultado é ilustrado na Figura 41.

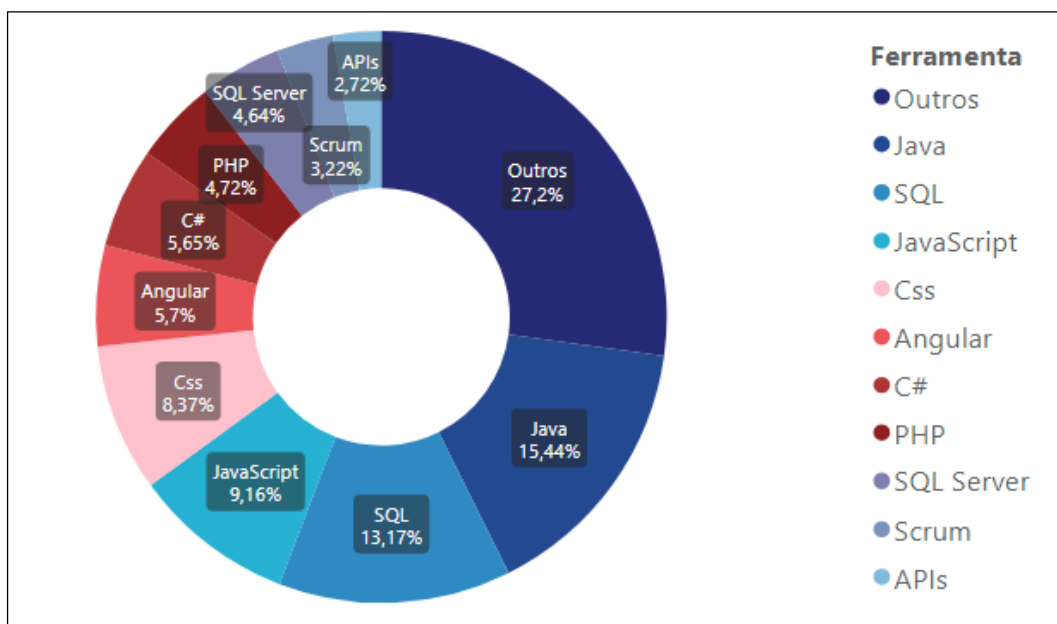
Figura 41: As 10 ferramentas mais solicitadas para o cargo desenvolvedor

Ferramenta	Nº de Ocorrências
Java	3952
SQL	3371
JavaScript	2344
CSS	2143
angular	1458
C#	1446
PHP	1209
SQL Server	1187
Scrum	823
APIs	697
<b>Total</b>	<b>18630</b>

Fonte: O autor

Com o Power Bi, também é possível uma representação gráfica, o exemplo mostrado no Gráfico 4, utiliza dos mesmos critérios da análise anterior, porém foi adicionado o agrupamento das ferramentas restantes (abaixo do top 10) em “outros”, além de cada elemento ser rotulado com a sua porcentagem sobre o total de ocorrências.

Gráfico 4: As 10 ferramentas mais solicitadas para o cargo desenvolvedor



Fonte: O autor



Com 12 dimensões para serem analisadas, torna-se possível quantidade expressiva de cruzamentos suscetíveis a análise. Em poucos segundos se pode, por exemplo, extrair um relatório que indique quais são as ferramentas mais utilizadas por um determinado cargo, cruzando com cidade, plataforma ou qualquer uma das dimensões existentes. Assim sendo, o conjunto de relatórios factíveis é extremamente extenso, e valoroso ao profissional de TI, pois o auxilia a ter a visão do que o mercado solicita.

## 5 CONSIDERAÇÕES FINAIS

Na fundamentação teórica foi revisada a literatura acadêmica com artigos que tratam do processo de ETL, extração de informação e seu uso, mecanismos de *Scraping* e *Crawler*, *Data Warehouses*, e sobre a ferramenta de análise de dados *Power Bi*. Foram identificadas as principais abordagens para a modelagem dimensional de dados e as atividades necessárias ao tratamento dos dados não estruturados, que embora não sejam objeto de estudo deste trabalho possibilitam análise posterior em outros estudos.

Na etapa da coleta dos dados, a análise da legalidade da extração dos dados foi de extrema significância, para que o projeto prossiga legalmente. O *BeautifulSoup 4* demonstrou ser uma biblioteca efetiva para extração de dados web utilizando o código HTML das páginas, a sua simplicidade facilitou a construção do *Web Crawler* e *Web Scraper*.

Com ela, foi possível extrair os dados da Web bastando, apenas, conhecimentos de HTML e da linguagem de programação Python. Após ter sido programado corretamente, a sua execução propiciou a extração da informação de forma rápida e precisa, porém um ponto importante a ser destacado, é a configuração do tempo máximo de resposta para cada requisição enviada, no projeto esse tempo foi de três segundos, das 36.714 requisições enviadas (5.614 requisições do *Crawler* e 31.100 do *Scraping*), 9.275 excederam o tempo limite, ocasionando uma perda de 13.285 vagas ou 25,01% do montante total de 37.084 disponibilizado pelas plataformas. Uma proposta para resolução do problema, seria a programação de reenvio das URL's cujo tempo de resposta exceda o limite preestabelecido.

A ferramenta Pentaho, conseguiu realizar todo o processo de transformação dos dados com qualidade e eficiência, a sua utilização foi de grande valia, pois realizou todo o processo de limpeza dos dados assim como a carga para as tabelas dimensões e fato para o *Data Mart*.

Sobre a etapa de reconhecimento e classificação dos atributos, pode-se concluir que o algoritmo criado foi deveras satisfatório, reconhecendo com maestria todos os *tokens* criados que estavam presentes nas vagas, entretanto, o dissabor foi por conta do baixo número de *tokens* criados para reconhecimento dos atributos, devido a forma manual de classificá-los, haja vista que a criação manual dos tokens

demanda tempo considerável, o que em relação a quantidade de vagas abordadas neste trabalho, corresponde há tempo superior ao ideal.

Os elementos e técnicas da modelagem dimensional para o *Data Mart* possibilitaram uma melhor organização e visualização dos dados, as informações extraídas do fato vaga, foram apreciadas sob diversas óticas diferentes, através das dimensões, que permitiu a visualização do conjunto de informações de forma consistente.

O uso da ferramenta *Power Bi* proporcionou uma implementação rápida, e a criação de relatórios e indicadores de uma maneira simples e intuitiva, além de possibilitar o desenvolvimento das análises mesmo para usuários sem conhecimentos técnicos específicos.

Este trabalho demonstrou os conhecimentos necessários para a utilização do conceito de ETL e as principais técnicas utilizadas em suas várias etapas. A utilização das diversas ferramentas de *Business Intelligence* nesse trabalho foi de grande relevância, pois realizou os processos de extração, tratamento, e visualização dos dados, transformando vários dados desestruturados de páginas web, em informações relevantes para o auxílio do profissional de Tecnologia da Informação a uma visão basilar de atributos, ferramentas, conhecimento e perfil interpessoal, que é solicitado pelo mercado, para assim se qualificar através das singularidades exigidas.

## 6 TRABALHOS FUTUROS

Para novos trabalhos, com a continuação desta obra, destacasse a criação e classificação mais elaborada de *tokens* para reconhecimento dos atributos ferramentas, atribuições do cargo e perfil interpessoal.

Também é visado o desenvolvimento de uma plataforma online, para que o público geral tenha acesso aos resultados do projeto.

Nessa plataforma será possível que o usuário insira quais são as habilidades e competências que já possui, e o sistema faça um relatório, retornando os cargos que o usuário mais se aproxime com as suas habilidades atuais, e também mostre quais as habilidades são necessárias e o candidato não possui.

## 7 REFERÊNCIAS

BARBIERI, Carlos. *Business Intelligence: Modelagem e Qualidade*. Rio de Janeiro: Elsevier, 2011. 392 p

BERGAMASCHI, Sonia. *Keyword search over relational databases: a metadata approach*. In: Proceedings of the 2011 ACM SIGMOD International Conference on Management of data. ACM, 2011.

BULL, Robert Ian et al. *Semantic grep: Regular expressions+ relational abstraction*. In: Ninth Working Conference on Reverse Engineering, 2002. Proceedings. IEEE, 2002.

DALE, Kyre. *Data Visualization with Python and JavaScript: Scrape, Clean, Explore & Transform your Data*, O'Reilly Media Inc., California, 2016.

DATE, Christopher J. *Introdução a sistemas de bancos de dados*. Elsevier Brasil, Tradução da 8<sup>o</sup> Edição Americana. Edição: 1; São Paulo, 2004.

DINELEY D., *Best Of Open Soucer Software Awards 2008*, - California, 2008. Disponível em <[https://www.infoworld.com/slideshow/2008/11/183-2008\\_infoworld\\_-1.html](https://www.infoworld.com/slideshow/2008/11/183-2008_infoworld_-1.html)> Acesso em 02 de novembro de 2019.

ELMASRI, Ramez; NAVATHE, Shamkant. *Sistemas de banco de dados*, Pearson Education do Brasil, 4<sup>a</sup> edição, São Paulo, 2006.

FERRARI, Alberto; RUSSO, Marco. *Introducing Microsoft Power BI*. Microsoft Press, 2016.

INMON, Willian H. *Building the Data Warehouse*. 3th ed.. John Wiley and Sons, Inc. 2002.

INMON, Willian. H.; WELCH, JD; GLASSEY, KL *Gerenciando Data Warehouse*. Tradução: Ana de Sá Woodward, São Paulo: Makron Books, 1999.

JARMUL, Katharine; LAWSON, Richard. *Python Web Scraping*, 2ª edição, John Wiley & Sons, 2011. Birmingham, 2017.

JENSEN, Simon Holm; MADSEN, Magnus; MOLLER, Anders. *Modeling the HTML DOM and browser API in static analysis of JavaScript web applications*. In: Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering. ACM, 2011.

JETBRAINS. *PyCharm Python IDE for Professional Developers* <<https://www.jetbrains.com/help/pycharm/>> Acesso em 02 out. 2019.

KIMBALL, Ralph. CASERTA, Joe. *The Data Warehouse ETL Toolkit: Practical Techniques for Extracting, Cleaning, Conforming and Delivering Data*. Indianápolis: Wiley, 2004.

KIMBALL, Ralph. e ROSS, Margy. *The Data Warehouse Toolkit: the complete guide to dimensional modeling*, 2nd ed. John Wiley and Sons, Inc. (2002)

KIMBALL, Ralph. *The Data Warehouse lifecycle toolkit*. 2. ed. Indianapolis, Indiana (USA): Wiley Publishing Inc., 2008.

LAWSON, Richard. *Web scraping with Python*. Packt Publishing Ltd, 2015.

LIU, Bing. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data (Data-Centric System and Applications)*, 2nd ed. Heidelberg, New York, Springer, 2011.

MICROSOFT. 2019 *ETL (extrair, transformar e carregar)*. Disponível em: <<https://docs.microsoft.com/pt-br/azure/architecture/data-guide/relational-data/etl>>. Acesso em: 1 nov. 2019.

MITCHELL, Ryan. *Web Scraping with Python / Ryan Mitchell*. Boston: O'Reilly Media, 2015. 256 c.

PENNA, Rogério Adriano Castelpoggi et al. *O Data Warehouse como Suporte à Inteligência de Negócio*. 2003.

PEYRAVI M. H., *A Schema Selection Framework for Data Warehouse Design*, International Journal of Machine Learning and Computing, Vol. 2, No. 3, June 2012,

PICHILIANI, Mauro Carlos. *Conversando sobre banco de dados*. Clube de Autores managed, 2011.

POWELL, Brett. *Microsoft power bi cookbook: creating business intelligence solutions of analytical data models, reports, and dashboards*. Packt publishing ltd, 2017.

RAY, Sunil., 2015. *Beginner's guide to Web Scraping in Python using BeautifulSoup*; Analytics Vidhya. Disponível em: <<https://www.analyticsvidhya.com/blog/2015/10/beginner-guideweb-scraping-beautiful-soup-python>> Acesso em 01 de novembro de 2019.

RICHARDSON, Leonard. *Beautiful soup documentation*. April, 2007.

RODRIGUES C. *Web Scraping e Web Crawling Tecnicas de raspagem de dados na inteligência fiscal*, - Curitiba, 2015. Disponível em <<http://sinfisco.org.br/wp-content/uploads/2017/12/Web-Scraping-e-Web-Crowling-t%C3%A9cnicas-de-raspagem-de-dados-na-Intelig%C3%Aancia-Fiscal.pdf>> Acesso em 01 de novembro de 2019.

ROLDÁN, María Carina. *Pentaho 3.2 Data Integration: Beginner's Guide*. Packt Publishing Ltd, 2010.

SUN, Kunjian & LAN, Yuqing. (2012). *SETL: A scalable and high performance ETL system*. 3rd International Conference on System Science, 2012

TARNAVEANU, Diana. *Pentaho Business Analytics: a Business Intelligence Open Source Alternative*, *Database System Journal*, vol. III, nº 3/2012, 2012.



VERZOLA I. 2008 - DATA WAREHOUSE. Disponível em:  
<<http://www.pontes.inf.br/noticia.asp?id=24>>. Acesso em: 25 out. 2019

YULIANTO, Ardhian Agung. *Extract Transform Load (ETL) Process in Distributed Database Academic Data Warehouse*. APTIKOM Journal on Computer Science and Information Technologies, v. 4, n. 2, p. 61-68, 2019.

**Apêndice 1 – Código-fonte do método extract\_newpages()**

```
def extract_newpages(content):  
    soup = BeautifulSoup(content, "lxml")  
    pages = set()  
    for tag in soup.find_all("a", href=True):  
        if tag["href"].startswith("http"):  
            if re.search("/vagas/informatica-ti-e-internet/p)", tag["href"]):  
                pages.add(tag["href"])  
    return pages
```

**Apêndice 2 – Código-fonte do método extract\_links()**

```
def extract_links(content):
    soup = BeautifulSoup(content, "lxml")
    links = set()
    # percorre todas tags "a" que tenha o atributo href preenchido
    for tag in soup.find_all("a", href=True):
        if tag["href"].startswith("http"):
            if re.search("\d{7}", tag["href"]):
                links.add(tag["href"])
    return links
```

**Apêndice 3 – Código-fonte método `crawl()` em *empregos.com.br***

```
def crawl(start_url):
    seen_urls_page = set()
    seen_urls_vagas = set()
    available_urls_page = set([start_url])
    available_urls_vagas = set()
    pages = set()
    url_vagas = set()
    links_vagas = set()
    print("Iniciando Crawl no link : ", start_url)
    while available_urls_page:
        url = available_urls_page.pop()
        if url not in seen_urls_page:
            try:
                contentPage = requests.get(url,timeout=3).text;
            except :
                continue
            url_vagas = url_vagas.union(extract_links(contentPage));
            if len(url_vagas) >= 100:
                print("Inserindo "+ str(len(url_vagas)) + " links no banco : ")
                insert.crawl_links_empregoscombr(url_vagas)
                url_vagas.clear()
            seen_urls_page.add(url);
            pages = extract_newpages(contentPage)
            available_urls_page = available_urls_page.union(pages)
    return url_vagas
```

**Apêndice 4 - Código-fonte método crawl() em *catho.com***

```
def crawl(start_url):
    seen_urls_page = set()
    seen_urls_vagas = set()
    available_urls_page = set([start_url])
    available_urls_vagas = set()
    pages = set()
    url_vagas = set()
    links_vagas = set()
    print("Iniciando Crawl no link : ", start_url)
    while available_urls_page:
        url = available_urls_page.pop()
        if url not in seen_urls_page:
            try:
                contentPage = requests.get(url,timeout=3).text;
            except :
                continue
            url_vagas = url_vagas.union(extract_links(contentPage));
            if len(url_vagas) >= 100:
                print("Inserindo "+ str(len(url_vagas)) + " links no banco : ")
                insert.crawl_links_catho(url_vagas)
                url_vagas.clear()
            seen_urls_page.add(url);
            pages = extract_newpages(contentPage)
            available_urls_page = available_urls_page.union(pages)
    return url_vagas
```

**Apêndice 4 - Código-fonte método crawl() em *empregos.profissionaisti.com.br***

```
def crawl(start_url):
    seen_urls_page = set()
    seen_urls_vagas = set()
    available_urls_page = set([start_url])
    available_urls_vagas = set()
    pages = set()
    url_vagas = set()
    links_vagas = set()
    print("Iniciando Crawl no link : ", start_url)
    while available_urls_page:
        url = available_urls_page.pop()
        if url not in seen_urls_page:
            try:
                contentPage = requests.get(url,timeout=3).text;
            except :
                continue
            url_vagas = url_vagas.union(extract_links(contentPage));
            if len(url_vagas) >= 100:
                print("Inserindo "+ str(len(url_vagas)) + " links no banco : ")
                insert.crawl_links_empregosprofissionaisti(url_vagas)
                url_vagas.clear()
            seen_urls_page.add(url);
            pages = extract_newpages(contentPage)
            available_urls_page = available_urls_page.union(pages)
    return url_vagas
```

**Apêndice 5 – Código-fonte do método `Extract_box_header()` em *empregos.com.br***

```
def extract_box_header(content):  
    soup = BeautifulSoup(content, "lxml")  
    box = soup.find(class_="info-empresa")  
  
    titulo = box.find("h1").text  
    data_public = box.find_all("p")  
  
    return titulo.strip(), data_public[1].text.strip()
```

**Apêndice 6 – Código-fonte do método Extract\_box\_body() em *empregos.com.br***

```
def extract_box_body(content):
    soup = BeautifulSoup(content,"lxml")

    desc =
soup.find(id="ctl00_ContentBody_trDescricaoVaga").text.replace("Descrição:", "").replace(" ", "").strip()

    qual =
soup.find(id="ctl00_ContentBody_trQualificacao").text.replace("Qualificação:", "").replace(" ", "").strip()

    form =
soup.find(id="ctl00_ContentBody_trFormacao").text.replace("Formação:", "").replace("Formação", "").replace(" ", "").strip()

    table = soup.find(class_="conteudo-vaga grid-12-16")
    p = table.find_all("p")
    string = p[3].text.replace(" ", "").strip()
    x = string.split("-")
    local = x[0]
    n_vagas = x[1]

    return desc, qual, form, local, n_vagas
```



**Apêndice 7 – Código-fonte do método `extract_box_salario()` em *empregos.com.br***

```
def extract_box_salario(content):
    soup = BeautifulSoup(content,"lxml")
    box = soup.find(class_="salario-vaga grid-4-16")

    salario = box.find(class_="valor itens").text.replace("
", "").replace("Salário:", "").strip()
    reg_cont = box.find(id="ctl00_ContentBody_divFormaDeContratacao").text.replace("
", "").replace("Forma de Contratação:", "").strip()
    beneficios = box.find(id="ctl00_ContentBody_divBeneficios").text.replace("
", "").replace("Benefícios:", "").strip()
    hor_trab = box.find(id="ctl00_ContentBody_divHorarioDeTrabalho").text.replace("
", "").replace("Horário de Trabalho:", "").strip()

    return salario, reg_cont, beneficios, hor_trab
```

**Apêndice 8 – Código-fonte do método `extract_box_header()` em *catho.com***

```
def extract_box_header(content):  
    soup = BeautifulSoup(content, "lxml")  
    box = soup.find(class_="headerVaga")  
    titulo = box.find(id="anchorTituloVaga").text.replace(" ", "").strip()  
    print(titulo)  
    return box.text.replace(" ", "").strip(), titulo
```

**Apêndice 9 – Código-fonte do método `extract_box_body()` em *catho.com***

```
def extract_box_body(content):
    soup = BeautifulSoup(content, "lxml")
    section = soup.find(id="descricao-da-vaga")
    desc = section.find("p").text.strip()
    print(desc)
    print(desc.encode("utf-8"))
    section = soup.find(class_="groupDadosVaga observacoes")
    try:
        sec_bene = section.find(class_="beneficios")
        beneficios = sec_bene.find("article").text.strip().replace(" ", "")
    except:
        beneficios = ""
    try:
        sec_info = section.find(class_="informacoesAdicionais")
        info_adici = sec_info.find("article").text.strip().replace(" ", "")
    except:
        info_adici = ""
    try:
        sec_regi = section.find(class_="regimeContratacao")
        regime = sec_regi.find("article").text.strip().replace(" ", "")
    except:
        regime = ""
    try:
        sec_jorn = section.find(class_="horario")
        jornada = sec_jorn.find("article").text.strip().replace(" ", "")
    except:
        jornada = ""
    return desc, beneficios, jornada, info_adici, regime
```

**Apêndice 10 – Código-fonte do método `extract_box_header ()` em *empregos.profissionaisti.com.br***

```
def extract_box_header(content):  
    soup =BeautifulSoup(content, "lxml")  
    box = soup.find(class_="job-header")  
    n_vagas = box.find("p").text  
    titulo = box.find(class_="page-title").text  
    div = box.find(class_="job-icons clearfix")  
    div_a = div.find_all("a")  
    local = div_a[1].text  
    funcao = div_a[2].text  
    data_cad = div_a[3].text  
    return local, funcao, data_cad, titulo, n_vagas
```

**Apêndice 11 – Código-fonte do método `extract_box_body ()` em *empregos.professionaisti.com.br***

```
def extract_box_body(content):  
    soup = BeautifulSoup(content, "lxml")  
    box = soup.find(id="content")  
    sessao = box.find_all("span", class_="label")  
    regime = sessao[0].text  
    jornada = sessao[1].text  
    desc = box.find(class_="job-content").text  
  
    return regime, jornada, desc
```

**Apêndice 12 – Código-fonte do método scrap() em  
*empregos.profissionaisti.com.br***

```
def scrap(urls):
    print("Iniciando scrap em : " + str(len(urls)) + " Links")
    count = 1
    while urls:
        url = urls.pop()
        print(str(count) + " Links Extraídos")
        #verificar se o link já existe no banco
        #Sim-> verifica se a vaga fechou e vai pro proximo laço
        try:
            content =get(url, timeout=3).text;
            local, funcao, data_cad, titulo, n_vagas = extract_box_header(content)
            regime, jornada, desc = extract_box_body(content)
        except:
            print("erro ao extrair content")
            continue

insert.scrap_empregosprofissionaisti(desc,regime,jornada,local,funcao,data_cad,titulo,n_vagas,url)
    count = count + 1
```

### Apêndice 13 – Código-fonte do método scrap() em *catho.com*

```

def scrap(urls):

    print("Iniciando scrap em : " + str(len(urls)) + " Links")
    count = 1
    while urls:
        url = urls.pop()
        print(str(count) + " Links Extraídos")
        #verificar se o link já existe no banco
        #Sim-> verifica se a vaga fechou e vai pro proximo laço
        try:
            content =get(url, timeout=3).text;
            cabecalho,cargo = extract_box_header(content)
            desc,      beneficios,      jornada,      info_adici,      regime      =
extract_box_corpo(content)
        except:
            continue
        insert.scrap_catho(cabecalho, desc, cargo, beneficios, jornada, info_adici,
regime)
        count = count + 1

```

**Apêndice 14 – Código-fonte do método scrap() em *empregos.com.br***

```
def scrap(urls):

    print("Iniciando scrap em : " + str(len(urls)) + " Links")
    count = 1
    while urls:
        url = urls.pop()
        print(str(count) + " Links Extraídos")
        try:
            content =get(url, timeout=3).text;
            salario, reg_cont, beneficios, hor_trab = extract_box_salario(content)
            titulo, data_public = extract_box_header(content)
            desc, qual, form, local, n_vagas = extract_box_corpo(content)
        except:
            continue

        insert.scrap_empregoscombr(
            desc, qual, titulo, local, titulo, n_vagas,
            salario, reg_cont, beneficios, hor_trab, url, data_public
        )
        count = count + 1
```



**Apêndice 15 – Cargos classificados para a área de Desenvolvimento de *software***

Analista de Arquitetura
Analista de Experiência do Usuário - UX
Analista de Integração de Sistemas
Analista de Processos
Analista de Requisitos
Analista de Sistemas
Analista de Teste
Analista DevOps
Arquiteto de Software
Assistente de Desenvolvimento
Desenvolvedor
Designer de Interfaces
Engenheiro de Software
Engenheiro de Testes
Gerente de Desenvolvimento
Gerente de Testes
Supervisor de Desenvolvimento

**Apêndice 16 – Cargos classificados para a área de Gestão/Projetos/Comercial**

Agile Master
Analista Comercial
Analista de Implantação
Analista de Inovação
Analista de Negócios
Analista de Projetos
Analista de Qualidade
Analista de TI
Assistente Comercial
Assistente TI
Comprador
Consultor Comercial
Consultor de Ferramenta
Consultor de Planejamento
Coordenador de Outsourcing
Engenheiro de Qualidade
Executivo de Vendas
Product Owner
Scrum Master
Supervisor Comercial
Supervisor de Contas
Supervisor de Projetos
Supervisor de TI

**Apêndice 17 – Cargos classificados para a área de Big Data/I.A./Data Science**

Analista de BI
Analista de Big Data
Analista de Dados
Analista de IOT
Analista ETL
Arquiteto de Dados
Cientista de Dados
Consultor de Ferramenta
Engenheiro de Dados
Pesquisador

**Apêndice 18 – Cargos classificados para a área de Banco de Dados**

Administrador de Banco de Dados - DBA
Analista de Banco de Dados
Analista de Gestão de Dados
Analista MIS
Arquiteto de Banco de Dados
Assistente MIS
Desenvolvedor de Banco de Dados
Engenheiro de Banco de Dados

**Apêndice 19 – Cargos classificados para a área de Suport/Help Desk/Telecon**

Analista de Atendimento
Analista de Relacionamentos
Analista de Service Desk
Analista de Suporte - Help Desk
Analista de Telecon
Assistente de Suporte - Help Desk
Coordenador de Suporte - Help Desk
Técnico de Service Desk
Técnico de Suporte - Help Desk
Técnico de Telecon

**Apêndice 20 – Cargos classificados para a área de Infraestrutura/Redes/Cloud**

Administrador de Redes
Analista Cloud
Analista de Infraestrutura
Analista de Middleware
Analista de Monitoramento
Analista de Redes
Analista de Servidores
Arquiteto Cloud
Arquiteto de Soluções
Arquiteto Redes
Assistente de Infraestrutura
Assistente de Redes
Auxiliar Técnico
Consultor Cloud
Consultor de Soluções
Coordenador de Infraestrutura
Engenheiro de Middleware
Técnico de Impressora
Técnico de Infraestrutura
Técnico de Redes
Técnico em Informática

**Apêndice 21 – Cargos classificados para a área de Segurança Da Informação**

Analista de Segurança da Informação
Coordenador de segurança da informação
Engenheiro de Segurança da Informação
Técnico de Segurança da Informação

**Apêndice 22 – Cargos classificados para a área de Marketing Digital/E-commerce**

Analista Adwords
Analista de Marketing
Gerente de Marketing
Web Designer
Wordpress
Assistente de Marketing



## Apêndice 23 – Tokens categorizados para reconhecimento de ferramentas

C	C#	Inglês Básico	processos siderúrgicos
R	C++	Inglês Fluente	Programação
Ruby	clicktale	Inglês técnico	projetos ágeis
Swift	cloud AWS	Invision	projetos de implementação
Visual Basic .NET	Cloud Computing	Invision Studio	Prototipação
Agile Coach	Cobol	Italiano - Fluente	protótipos
jAX- WS	Controle de fluxo	Java	Psicologia voltada para o design
acessibilidade	CSS	java web	Python
Acronis	Cumulative Flow Diagram	JavaScript	REACT
Adobe XD	Delphi	JSON	Redes de CFTV
Ágil/SCRUM	desenho de fluxos	kanban	Rest
Agile Coach	Design gráfico	leituras de dados	SaaS
Agile Master	design responsivo	máquinas virtuais	SAP
Amazon Web Services	Design Sprint	MATLAB	Scratch
Amazon WS	Design Thinking	metodologias ágeis	Scrum
Análise de métricas	DynamoDb	Métodos de pesquisa pela internet	Scrum
Análise Heurística	estudos etnográficos	métricas	servidores de aplicação
angular	Experiência na área	Microsoft Azure	sistemas legados
APIs	eyetracking	mockups navegáveis	Sketch
aplicações nuvem	Figma	MongoDb	SOAP
Arquitetura SOA	frameworks	navegabilidade	Sprint MVC
Assembly	Francês - Intermediário	NODEJS	SQL
autocad	FRONT-END	NOSQL	SQL Server
axis	gerenciamento de projetos	Objective-C	Telecom
axure	Go	outsourcing	testes de usabilidade
Azure	Google Cloud	Perl	usabilidade
BACK-END	Histograma	persistência com hibernate	Vcenter
background técnico	HTML5	Photoshop	Visual Basic
banco de dados relacional	Illustrator	PHP	VMWare
Bancos de dados relacional	Inglês - Avançado	Plataforma de Integrações	Vue.js
boot e security	Inglês - Básico	PMI	webAnalytics
bootstrap	Inglês - Fluente	princípio	wireframes
brainstorm	Inglês avançado	Processo de Discovery	

**Apêndice 24 – Tokens categorizados para reconhecimento de benefícios**

<b>Ambiente Informal</b>
<b>Assistência Médica</b>
<b>Assistência Odontológica</b>
<b>Auxílio Creche</b>
<b>Celular fornecido pela empresa</b>
<b>Cesta Básica</b>
<b>Convênio com Farmácia</b>
<b>Estacionamento</b>
<b>Horário Flexível</b>
<b>Medicina em grupo</b>
<b>meritocracia</b>
<b>Participação nos lucros</b>
<b>Plano de Carreira</b>
<b>Restaurante na empresa</b>
<b>Seguro de Vida</b>
<b>Tíquete Alimentação</b>
<b>Tíquete Refeição</b>
<b>Transporte Fornecido pela empresa</b>
<b>Vale Transporte</b>

## Apêndice 25 – Tokens categorizados para reconhecimento de atribuições do cargo

Ajudar equipes	auxiliei na estratégia do produto	Explorar novas tecnologias
apresentar os resultados	busca pela alta performance	fábrica de software
definição das soluções	Capacidade de analisar problemas	Facilitar as reuniões
identifica atividades críticas	Colaborar com times criativos	Facilitar cerimônias
implantação e testes	Coletar informações	facilitar sessões de brainstorm
levantamento de requisitos	Compreender o objetivo do negócio	Fazer Benchmark
participar da condução do processo	Comunicação com o time	Fornecer KPIs de alerta
Realizar a convergência das validações	comunicação entre os sistemas	garantindo a melhor experiência
acompanhar os principais indicadores	comunicar ideias de design	gerenciar o seu tempo para cumprir as suas tarefas
acompanhar os procedimentos	configuração DCOM	Gerenciar os milestones
adoção de práticas ágeis	conhecer a respeito do público	Identificar e propor melhorias
adquirir novos conhecimentos	construir sistemas distribuídos	Identificar problemas de usabilidade
ajudar os nossos times	criação de interfaces para apps mobile	Influenciar as tomadas de decisão
alcance os objetivos planejados	Criar hipóteses e testes	Infraestrutura de redes
Análise e desenvolve projetos	criar interfaces	Instalar e configurar câmeras de CFTV
análise estatística	criar itens do backlog	Instalar e configurar drives
Análise Heurística	descrever os objetivos	Instalar e configurar host
aperfeiçoar os métodos	desenho em autocad	Instalar e configurar interfaces de coleta de dados
Aplicar testes de usabilidade	Desenvolver a transparência	integração de processos
Aplicar treinamentos	desenvolver licenciamento	interagir com demais áreas
apoiando as áreas internas	desenvolver soluções	interpretar indicadores de performance
apoiar as decisões	desenvolvimento	manter coesão entre design
apoiar na produção de conteúdo	desenvolvimento de modelagem	manter um bom relacionamento com o time
apoio a criação de novas soluções	Desenvolvimento de sistemas distribuídos	manutenção dos sistemas
aprendizado contínuo	desenvolvimento de soluções	Mapear Storyboards
apresentar conceitos	elaboração de croqui	maximizar o valor do trabalho
aprimorar as soluções	elaborar arquitetura	modelagem de sistemas
aprimorar o desempenho da equipe	Elaborar descritivo funcional	Monitorar e controlar o escopo
atingir os objetivos	encontrar o melhor caminho	montagens de processos
atuando em projetos de clientes	Entender as necessidades do cliente	otimização
Atuará em projetos de grandes clientes	entender problemas técnicos	otimizar os esforços de implantação
Aumentar o número de conversões	entendimento de problemas	Otimizar os recursos de servidores
Aumentar o tempo de permanência nos sites	entregar personas	Participar das reuniões
Auxiliar o time	estimar as situações do mundo real	Participar de debates com os demais times
Auxiliar o time de Produtos	Estimular o time	Participar de reuniões técnicas
passagem de seus conhecimentos	propor correções e melhorias	reuniões de planejamento
pesquisas com usuários	prototipar	revisar e implementar políticas unificadas
planejamento e desenvolvimento	Prototipar projetos	Saber lidar com falta de informação
Planejar a experiência	Realizar a documentação	Seguir análises de problemas
planos de ação	Realizar análise financeira	Seguir evoluções do produto
práticas organizacionais	Realizar backups de servidores	sugerir melhorias nos processos
Preparar e documentar as conclusões	realizar o mapeamento	testes de performance
Projetar fluxos	Realizar pesquisas de design	testes de usabilidade
Promover um ambiente de entrega	Realizar testes A/B	Trabalhar ativamente com o time
propor a criação de interfaces	Resolução de problemas	trabalhar na jornada do cliente
Treinar equipe	workshops com equipe	trazer uma perspectiva
utilizar técnicas de user experience		Treinar a equipe

## Apêndice 26 – Tokens categorizados para reconhecimento de perfil interpessoal

Abordagem metódica	comunicativo	feliz	perspicaz
Adaptabilidade	conectadas e atualizadas	fiel	Persuasão
Agilidade para tomar decisões	Confiabilidade	Flexibilidade	Planejamento futuro
agradável	confiante	habilidoso	pontual
alegre	confiável	honesto	preocupado
altruísta	Controle do estresse	honrado	preparado
amigável	cooperação entre a equipe	humilde	prestativo
apaixonados por tecnologia	corajoso	imparcial	prestável
aplicado	cordial	influência sob a equipe	proatividade
Aprendizado contínuo	cortês	Iniciativa	proativo
arguente e negocie	Crescimento na carreira	iniciativa	produtivo
assertivo	Criatividade	Inovação	Prudência
atencioso	criativo	inovador	prudente
atento	craterioso	Integridade	racional
autêntico	cuidadoso	íntegro	respeitador
Autoconfiança	Cumprir prazos	Inteligência emacional	responsável
autoconhecimento	Curioso	Inteligência social	sábio
Autocontrole	decente	inteligente	sagaz
autodesenvolvimento	dedicado	Intuição	sensato
aventureiro	descontraído	inventivo	senso crítico
Bom humor	determinado	justo	simpático
Busca por conhecimentos	digno	leal	sincero
calmo	disciplinado	legal	sinta prazer em aprender
Capacidade crítica	divertido	Liderança	tolerante
Capacidade de ouvir	educado	Motivação	Trabalho colaborativo
carismático	eficiente	Negociação	Trabalho em equipe
coerência	Empatia	observador	transparente
colaborar com o aprendizado	empenhado	organização	valente
Competitividade	empreendedor	organizado	valoroso
Compreensão	engraçado	otimista	verdadeiro
compreensivo	entusiasta	ousado	Visão do negócio
Compromisso com a excelência	Equilíbrio emocional	paciente	Visão no cliente
Comunicação	esforçado	Pensamento estruturado	visão sistêmica
Comunicação efetiva	Ética	perfeccionista	zeloso
comunicação eficaz	Falar em público	perseverante	Fazer mais com menos