UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Instituto de Ciência e Tecnologia
Câmpus de Sorocaba

DOUGLAS MARCELO CONSTANTINI BUENO

Automação e engenharia de dados para a análise das disparidades salariais de gênero no mercado de trabalho brasileiro: um estudo de 2012 a 2021

Sorocaba
2023

DOUGLAS MARCELO CONSTANTINI BUENO

Automação e engenharia de dados para a análise das disparidades salariais de gênero no mercado de trabalho brasileiro: um estudo de 2012 a 2021

Trabalho de graduação apresentado ao Instituto de Ciência e Tecnologia, da Universidade Estadual Paulista “Júlio de Mesquita Filho” Campus Sorocaba, como parte dos requisitos para obtenção do grau de bacharel em Engenharia de Controle e Automação.

Orientador: Prof. Dr. Everson Martins

Sorocaba

2023

B928a

Bueno, Douglas Marcelo Constantini

Automação e engenharia de dados para a análise das disparidades salariais de gênero no mercado de trabalho brasileiro: um estudo de 2012 a 2021 / Douglas Marcelo Constantini Bueno. -- Sorocaba, 2023

53 p.

Trabalho de conclusão de curso (Bacharelado - Engenharia de Controle e Automação) - Universidade Estadual Paulista (Unesp), Instituto de Ciência e Tecnologia, Sorocaba

Orientador: Everson Martins

1. Automação por software. 2. Python (computer program language). 3. PostgreSQL. 4. Power BI. 5. ETL- Extract, Load, Transform. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca do Instituto de Ciência e Tecnologia, Sorocaba.
Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Instituto de Ciência e Tecnologia
Câmpus de Sorocaba

AUTOMAÇÃO E ENGENHARIA DE DADOS PARA A ANÁLISE DAS DISPARIDADES SALARIAIS DE GÊNERO NO MERCADO DE TRABALHO: UM ESTUDO DE 2012 A 2021

DOUGLAS MARCELO CONSTANTINI BUENO

ESTE TRABALHO DE GRADUAÇÃO FOI JULGADO ADEQUADO COMO PARTE DO REQUISITO PARA A OBTENÇÃO DO GRAU DE **BACHAREL EM ENGENHARIA DE CONTROLE E AUTOMAÇÂO**

Prof. Dr. Everson Martins
Coordenador

BANCA EXAMINADORA:

Prof. Dr. EVERSON MARTINS
Orientador/UNESP-Campus de Sorocaba

Profª. Drª. MARIA GLÓRIA CAÑO DE ANDRADE
UNESP- Campus de Sorocaba

Prof. Dr. NILSON CRISTINO DA CRUZ
UNESP – Campus de Sorocaba

Outubro de 2023

AGRADECIMENTOS

Primeiramente, agradeço aos meus pais, que mesmo sem educação formal, sempre se esforçaram ao máximo para me proporcionar uma educação de qualidade. Sou imensamente grato por sempre me apoiarem e nunca me deixarem desistir dos meus sonhos.

Em segundo lugar, agradeço à minha namorada, que me mostrou o mundo dos estudos e me encorajou a acreditar em mim mesmo, sempre me ajudando e estando ao meu lado nos momentos mais desafiadores.

Em terceiro lugar, agradeço ao Professor Dr. Everson Martins por dedicar seu tempo em me auxiliar no desenvolvimento do projeto.

Por fim, gostaria de agradecer a todos os funcionários da UNESP Sorocaba e aos laços de amizade que eu criei ao decorrer da graduação, pois sem eles não teria chegado até aqui.

RESUMO

Este trabalho teve como objetivo a utilização de softwares para automatizar os processos de extração e análise de dados públicos fornecidos pelo Ministério do Trabalho. O foco principal do projeto é sintetizar informações sobre o mercado de trabalho brasileiro nos últimos 10 anos e observar as discrepâncias salariais por gênero, tipo de ocupação e sua evolução ao longo do tempo. No desenvolvimento deste trabalho, foi empregada a linguagem Python e suas bibliotecas de análise e obtenção de dados, juntamente com o PostgreSQL para o armazenamento das informações obtidas. Além disso, utilizou-se o Power BI para a visualização e análise dos dados. A implementação dessas ferramentas permitiu a criação de um sistema de monitoramento de dados empregatícios com atualização automática, possibilitando a identificação de diferenças salariais entre municípios, estados, sexos e ocupações ao longo dos anos. Dessa forma, as ferramentas utilizadas forneceram uma solução eficiente para a análise de dados públicos do Ministério do Trabalho, permitindo a obtenção de conclusões relevantes sobre as discrepâncias salariais no mercado de trabalho brasileiro.

Palavras-chave: automação de processos; Python; PostgreSQL; *business intelligence*; dados abertos.

ABSTRACT

This work aimed to utilize software to automate the processes of extracting and analyzing open data provided by the Ministry of Labor. The main focus of the project is to synthesize information about the Brazilian labor market over the past 10 years and observe pay gaps related to gender and occupation as well as their evolution over time. This work used Python language and its data analysis and retrieval libraries along with PostgreSQL to store the collected information. Furthermore, Power BI was used for data visualization and analysis.

The use of these tools allowed the creation of a data monitoring system for employment database with automatic updates, enabling the identification of wage differences between cities, states, genders, and occupations over the years. Thus, the tools used in this work provided an efficient solution for analyzing open available data provided by the Ministry of Labor, allowing important analysis the related to pay gaps in the Brazilian labor market.

Keywords: process automation; Python; PostgreSQL; business intelligence; open data.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ETL *Extract, Load, Transform* - Extrai, carrega, transforma

APIs *Application Programming Interface* - Interface de Programação de Aplicação

SQL *Structured Query Language* - Linguagem de Consulta Estruturado

NoSQL *Not Only Structured Query Language* - Linguagem de Consulta Não somente estruturada

CSV *Comma-separated values* - Valores separados por vírgula

JSON *JavaScript Object Notation* - Notação de objetos JavaScript

BI *Business Intelligence* - Inteligência de negócios

DAX *Data Analysis Expressions* - Expressão de Análise de dados

CLT consolidação das leis trabalhistas

FTP *File Transfer Protocol* - Protocolo de transferência de arquivos

SUMÁRIO

1 INTRODUÇÃO

Na sociedade contemporânea, os dados de informação, também conhecidos como o *novo petróleo* segundo Clive Humby, desempenham um papel fundamental no processo de tomada de decisões. No entanto, quando estão em sua forma bruta, não é possível aproveitar todo o seu potencial. Por esse motivo, áreas como engenharia, ciência de dados e inteligência de negócios foram desenvolvidas com o objetivo de extrair o máximo potencial desse recurso valioso.

Neste setor, ocorre o constante desenvolvimento e aprimoramento de novas tecnologias, o que possibilita tornar os processos, como a tomada de decisões, mais eficientes e ágeis. Anteriormente, tarefas que demandam horas de trabalho agora podem ser facilmente executadas com o auxílio de ferramentas e linhas de código.

Além disso, a transparência nas informações é um aspecto destacado pela engenharia de dados e o *Business Intelligence*, que permitem que empresas coletem e apresentem informações essenciais de forma clara e objetiva, contribuindo para evitar fraudes e imprevistos.

Nesse sentido, visando promover a transparência das informações, a Comissão de Finanças e Tributação da Câmara dos Deputados do Brasil incentivou a implementação da Lei de Dados Abertos (Agência Câmara de Notícias, 2023). Essa lei permite que a população tenha acesso a dados produzidos por órgãos públicos, como índices de remuneração por município e renda anonimizada de cada cidadão registrado em regime empregatício.

Com base nessas informações, este trabalho teve como objetivo automatizar a obtenção de uma base de dados públicos, permitindo a coleta de informações frequentes para analisar a situação da remuneração em diferentes regiões do Brasil ao longo de um período de dez anos. A partir desses dados, foi possível analisar as diferenças salariais entre homens e mulheres em diferentes cargos.

Para realizar esse trabalho, foram aplicados conhecimentos de ETL (*Extract,Transform, Load*) para enriquecimento e processamento de grandes volumes de dados, utilização de bancos de dados relacionais para armazenar de forma organizada os dados extraídos, e *Business Intelligence* para visualização dos resultados finais.

1.1 Objetivos

O objetivo deste trabalho foi automatizar a obtenção de uma base pública de dados, permitindo a coleta de informações fornecidas frequentemente para analisar a situação salarial em diferentes regiões do Brasil. Com base nas informações obtidas é possível analisar as diferenças salariais entre homens e mulheres em diferentes cargos.

Para alcançar esse objetivo, foram aplicados conhecimentos de ETL (*Extract*, *Transform*, *Load*) para enriquecer e processar grandes volumes de dados, utilizando um banco de dados relacional para armazenar os dados extraídos. Além disso, foram empregadas técnicas de *Business Intelligence* para a visualização dos dados finais, possibilitando uma análise clara e objetiva das informações coletadas.

Ao automatizar a obtenção desses dados e realizar a análise das diferenças salariais, busca-se contribuir para a transparência das informações e fornecer subsídios para a identificação de disparidades salariais entre gêneros em diferentes estados e regiões do Brasil.

2 REVISÃO BIBLIOGRÁFICA

Inicialmente, é necessário esclarecer os conceitos teóricos utilizados para que o funcionamento do projeto possa ser adequadamente compreendido.

2.1 *Big data*

O termo *Big Data* foi criado para descrever conjuntos de dados que possuem volumetria e complexidade tão grandes que não podem ser gerenciados e analisados utilizando metodologias tradicionais. As caracterizações desses dados é baseado em três *Vs*:

- Volume: Refere-se a grandes quantidades de dados que são gerados diariamente por dispositivos de captura, como sensores industriais, redes sociais e websites.
- Velocidade: Indica a grande velocidade que estes dados são produzidos diariamente.
- Variedade: Diz respeito à diversidade de tipos de dados que são produzidos, como texto, imagens, vídeos e áudios.

Devido ao imenso volume de dados, as análises de big data tornaram-se uma atividade complexa e desafiadora. No entanto, ferramentas como Apache Spark tornaram-se fundamentais para lidar com essas informações e gerar valor a partir delas. (Oracle, [201–?])

2.2 ETL - *Extract, Transform, Load*

O resultado direto da 4ª revolução industrial é o aumento na geração de dados pelas suas tecnologias. Estas inovações permitem que empresas coletem, criem e analisem dados em tempo real, resultando em um crescimento exponencial na quantidade de dados gerados ao longo dos anos. No entanto, o manuseio dessas informações tem enfrentado desafios relacionados ao armazenamento, manutenção e análise.

Para lidar com essas dificuldades, foram desenvolvidas metodologias, como o ETL (Extração, Transformação e Carga), que permitem a manipulação dessas informações de maneira estruturada.

A etapa de extração envolve a coleta de dados de várias fontes, como arquivos de texto, banco de dados e *APIs.* Essa coleta pode ser realizada por meio

de técnicas como leitura de arquivos em suas diversas estruturas, como CSV e JSON, consulta a banco de dados usando *query* em SQL ou obtenção por requisições a *APIs*. Neste projeto, a extração da informação ocorrerá via requisição de uma *API* que fornecerá arquivos em formato CSV.

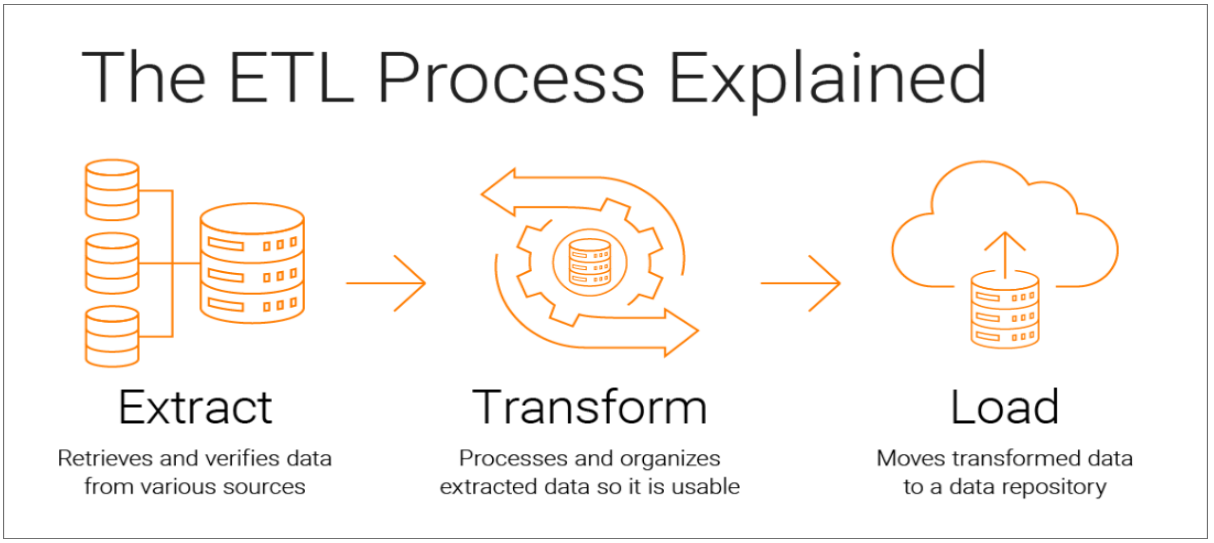
A etapa de transformação envolve a manipulação dos dados, sendo essencialmente o processo de tornar as informações obtidas úteis para o usuário final. Essa etapa é responsável por realizar a limpeza dos dados, removendo dados inconsistentes ou indesejáveis. É fundamental para garantir a qualidade e consistência dos dados finais. Neste projeto a transformação será realizada utilizando *scripts* na linguagem de programação Python, utilizando bibliotecas especializadas em ETL.

A última etapa, carga, envolve o envio das informações trabalhadas nos processos anteriores para um destino final unificado, como banco de dados. Neste projeto esta etapa gerará um arquivo em formato texto que posteriormente será exportado para um banco de dados relacional.

Em suma, o processo ETL é um método para integrar dados de várias fontes em um local centralizado e transformá-los em um formato útil para análise. Esse processo é fundamental para garantir a qualidade e consistência dos dados em um ambiente de análise de dados.

A Figura 1 exemplifica as três etapas do método ETL.

Figura 1 - Esquema do funcionamento de um processo de ETL.



Fonte: Informatica ([2022?]).

2.2.1 Shell Script e Linux

Em prol de manter a base sempre atualizada, foi utilizado o Shell Script no sistema operacional Linux para automatizar os downloads das bases e executar os *pipelines* em Python. A escolha da linguagem *Shell* deve-se à sua capacidade de criar comandos para o Linux, permitindo a automação de processos, agendamento de tarefas e manipulação de arquivos. Essa abordagem garantiu a atualização contínua da base de dados, melhorando a eficiência do sistema.

A Figura 2 ilustra o processo de download dos arquivos, movimentação da base, criação de diretórios e execução do *pipeline*.

Figura 2 - Shell Script que executa todos os comandos para a execução do *pipeline*.

```
#! /bin/bash

YEAR_DOWNLOAD="$(date +"%Y" -d "2 year ago")"


export directory=$YEAR_DOWNLOAD

if [[ ! -e $directory ]]; then
    mkdir $directory
elif [[ ! -d $directory ]]; then
    echo "$dir already exists but is not a directory" 1>&2
fi

cd  $directory

while true; do
    if wget -q --reject-regex 'RAIS_ESTAB_' -r "ftp://ftp.mtps.gov.br/pdet/microdados/RAIS/$YEAR_DOWNLOAD/"; then
            echo "url is working"
            break
    else
            echo "url is not working "
            sleep  200h
    fi
done


mkdir input

mv ftp.mtps.gov.br/pdet/microdados/RAIS/$(YEAR_DOWNLOAD)/* input
rm -r ftp.mtps.gov.br

mv input/$YEAR_DOWNLOAD input/RAIS

cd input/RAIS

7z x "*.7z"

rm *.7z

cd ../..

if [[ ! -e saida_municipio ]]; then
    mkdir saida_municipio
elif [[ ! -d saida_municipio ]]; then
    echo "$dir already exists but is not a directory" 1>&2
fi

if [[ ! -e saida ]]; then
    mkdir saida
elif [[ ! -d saida ]]; then
    echo "$dir already exists but is not a directory" 1>&2
fi

export diretorio_entrada=$YEAR_DOWNLOAD


python3 ./pipeline_enriquecimento_base.py
python3 ./media_rais_cbo_municipio.py
```

Fonte: Autoria própria.

Outro ponto importante a ser destacado é a utilização do *Cron,* um comando em Linux que permite agendar ações em diferentes períodos de tempo, bastando apenas informar qual é a data de realização, frequência e código a ser rodado.

Na Figura 3 é possível observar o processo de agendamento de execução do *pipeline*.

Figura 3 **-** Agendamento de rotina para cada dois anos.

```
# m h  dom mon dow   command
0 0 0 1 1 ? 2023/2 /mnt/d/tcc_versao_final/execute.sh > /mnt/d/tcc_versao_final/log.txt
```

Fonte: Autoria própria.

O agendamento é configurado de acordo com a regra apresentada na Tabela 1, e o período de agendamento é determinado pela posição dos números nos campos correspondentes.

Tabela 1 - Tabela descritiva de cada campo do comando crontab.

Campo	Significado	Valores
1	Minutos	0 a 59
2	Hora	0 a 23
3	Dia do mês	1 a 31
4	Mês	1 a 12
5	Dia da semana	0 a 7, sendo que os números 0 e 7 indicam o domingo
6	Comando a ser executado	qualquer comando válido do sistema

Fonte:Almeida(2018).

2.3 Python

Python é uma linguagem de programação orientada a objetos amplamente utilizada para manipulação de dados. Sua popularidade nesse contexto se deve à disponibilidade de diversas bibliotecas especializadas que facilitam o

desenvolvimento de *pipelines*. Essas bibliotecas oferecem uma ampla gama de funções para a manipulação e análise dos dados, o que torna Python muito popular entre engenheiros de dados e profissionais que lidam com grandes volumes de informações (Shubhnoor, 2021).

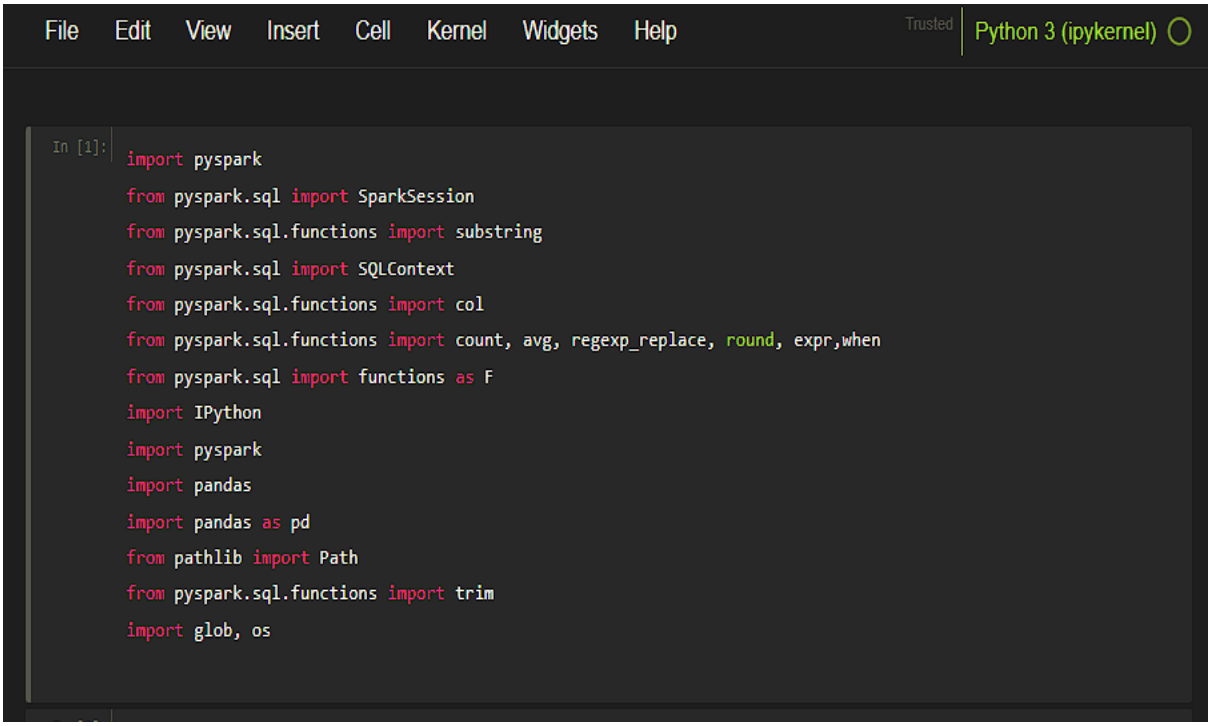
2.4 Jupyter Notebook

O Jupyter Notebook é uma interface visual interativa que permite a execução de código de forma modular, utilizando os seus *notebooks*. Ele oferece recursos para visualizar dados, testar funções e códigos de maneira simples e rápida.

Além disso, o Jupyter Notebook é uma ferramenta poderosa para explorar e visualizar dados, testar hipóteses, desenvolver e compartilhar projetos de *software*, bem como colaborar entre equipes de programação e análise de dados.

Na Figura 4, é possível observar a interface gráfica do Jupyter Notebook, que está sendo executado no ambiente Kernel Python 3. Neste exemplo existem dois notebooks em execução, nos quais é possível executar cada parte separadamente, sem a necessidade de executar todos os passos de uma vez.

Figura 4 - Interface gráfica do Jupyter Notebook.



Fonte: Autoria própria.

2.5 Apache Spark

O Apache Spark é um *framework* de código aberto amplamente utilizado para processar grandes volumes de dados. Em comparação com outras *frameworks* de processamento de dados, o Spark se destaca por sua velocidade, escalabilidade, flexibilidade e comunidade ativa de usuários.

1. **Velocidade**: Uma das principais características do Spark é sua capacidade de processar dados utilizando memória RAM, o que o torna significativamente mais rápido do que outros sistemas de processamento de dados distribuídos que fazem uso frequente de disco. Esta arquitetura resulta em processamentos até 100 vezes mais rápidos que outras ferramentas.
2. **Escalabilidade**: O Spark é altamente escalável, permitindo a utilização de *clusters* de computadores para processar grandes volumes de dados. O *Spark* distribui as tarefas em diferentes nós do *cluster* para que o processamento ocorra em paralelo, como se vários computadores estivessem trabalhando em conjunto. Isso possibilita dividir o processamento em tarefas menores, que são executadas simultaneamente em diferentes nós do *cluster*. Como resultado, o Spark pode processar grandes quantidades de dados de forma muito mais rápida do que um único computador ou servidor. Além disso, o Spark pode ser facilmente dimensionado adicionando mais nós ao *cluster* para aumentar sua capacidade de processamento.
3. **Flexibilidade**: O Spark é altamente flexível, pois pode trabalhar com diversas fontes de dados e linguagens de programação. Isso permite que empresas lidem com diferentes tipos de projetos e equipes utilizando uma única ferramenta.
4. **Comunidade ativa de usuários**: Devido a sua popularidade, o Spark possui uma comunidade muito ativa, o que facilita a troca de informações e o suporte entre os usuários. (Gimino , 2019)

2.6 Banco de dados

Banco de dados relacional (SQL) e não relacionais (NoSQL) são duas maneiras diferentes de armazenar informações em um banco de dados. Geralmente, os bancos SQL são armazenados em um único servidor, tornando sua

escalabilidade vertical. Por outro lado, os bancos de dados não relacionais possuem escalabilidade horizontal, ou seja, em vez de aprimorar um único *cluster*, se pode distribuir o processamento em vários servidores, tornando o processo mais barato. (Ferreira, [ca. 2020])

2.6.1 Banco de dados não relacional (NoSQL)

Os bancos de dados NoSQL (*Not Only SQL*) são baseados em modelos de dados não estruturados, usando outros formatos em vez de tabela para armazenar informações, como documentos, chave-valor e gráficos. (Schafer, [201--?])

1. **Armazenamento em formato de documento**s: Os dados são armazenados em formato semelhante a objetos JSON, com cada dado contendo pares de campos e valores. Cada documento pode ter uma estrutura diferente.
   A Figura 5 mostra o exemplo deste modelo de armazenamento.

Figura 5 - Exemplo de estrutura de um armazenamento de dados de documentos.

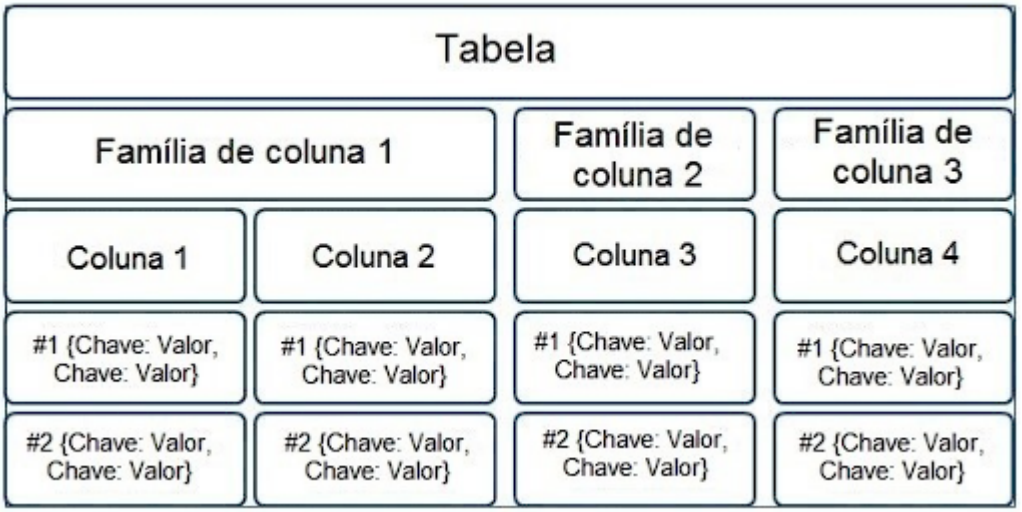
```
[
    {
        "year" : 2013,
        "title" : "Turn It Down, Or Else!",
        "info" : {
            "directors" : [ "Alice Smith", "Bob Jones"],
            "release_date" : "2013-01-18T00:00:00Z",
            "rating" : 6.2,
            "genres" : ["Comedy", "Drama"],
            "image_url" : "http://ia.media-imdb.com/images/N/O9ERW.jpg",
            "plot" : "A rock band plays their music at high volumes, annoying the neighbors.",
            "actors" : ["David Matthewman", "Jonathan G. Neff"]
        }
    },
    {
        "year": 2015,
        "title": "The Big New Movie",
        "info": {
            "plot": "Nothing happens at all.",
            "rating": 0
        }
    }
]
```

Fonte: Amazon Web Services ([201–?]).

2. **Modelo colunar:** Neste modelo, os dados são armazenados em colunas, diferente do modelo SQL *em* que os dados são armazenados em linhas. Isso

possibilita consultas mais eficientes em grandes conjuntos de dados. Na Figura 6, é apresentada uma tabela NOSQL com dados em formato coluna.

Figura 6 - Exemplo de tabela NOSQL com dados em formato coluna.



Fonte: Tipos [...] (2017).

3. **Dados chave/valor:** Nesse formato, cada informação é armazenada com uma chave e um valor, o que torna o modelo muito rápido para leitura. Esses dados são independentes entre si, o que permite inserção de dados em tempo de execução sem conflitar o banco de dados. Na Figura 7, é apresentada uma tabela NOSQL com dados em formato chave/valor.

Figura 7 - Exemplo de tabela NOSQL com dados em formato chave/valor.

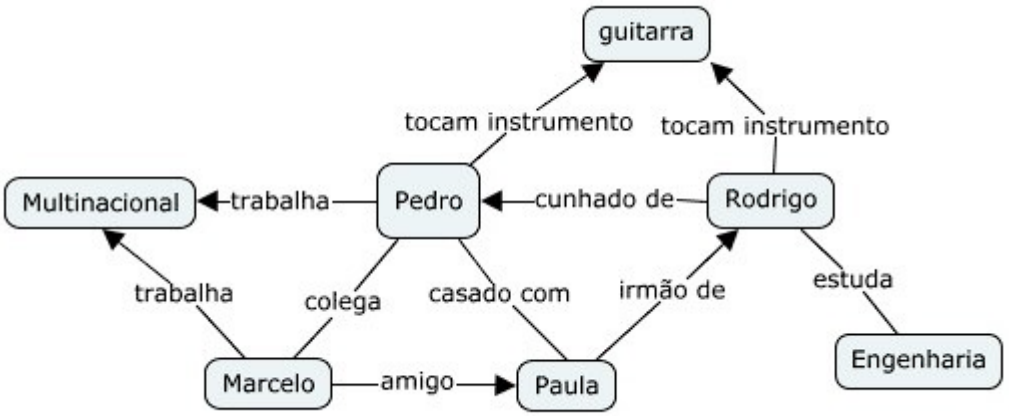
Chave	Valor
carro_3345_cor	preto
carro_3345_pneu	17
carro_3365_cor	branco
carro_3365_pneu	15
carro_4560_peso	1215
carro_4715_ano	2016

Fonte: Tipos [...] (2017).

4. **Modelo representado por Grafos**: Este modelo é utilizado onde exigem dados fortemente ligados e consultas complexas. Nele são utilizados os seguintes componentes básicos: um grafo, que representa o dado, e as

arestas, que representam as relações entre os nós do grafo e suas propriedades. Na Figura 8, é apresentado um exemplo de organização de dados NOSQL orientado a grafos.

Figura 8 - Exemplo de organização de banco de dados NoSQL orientado a grafos.



Fonte: Tipos [...] (2017).

2.6.2 Banco de dados relacional (SQL)

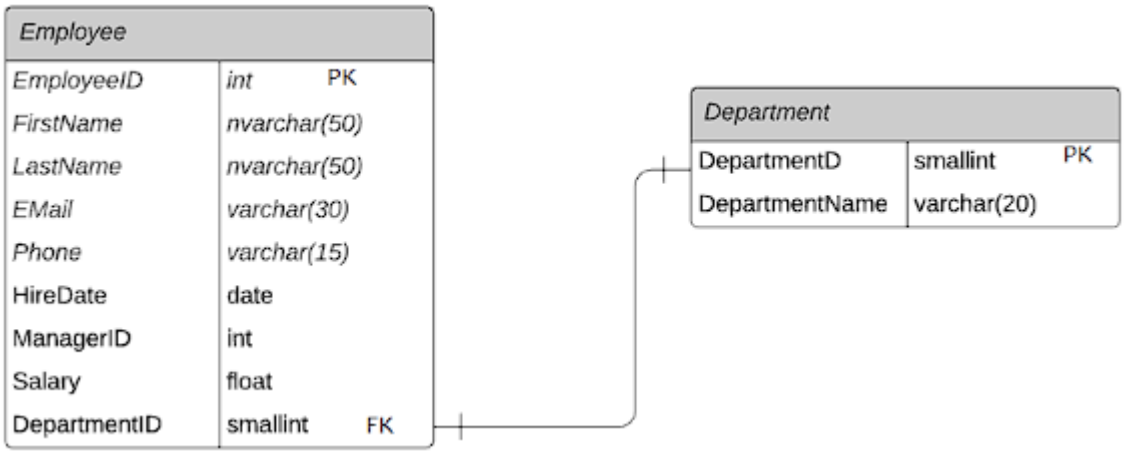
Banco de dados relacional é uma maneira de arranjar a arquitetura das informações em uma relação pré-definida de linhas e colunas. Estas informações são agrupadas em uma ou mais tabelas, o que facilita a visualização e o entendimento da informação ali armazenada.

No banco de dados relacional qualquer tabela consegue ser conectada a outra utilizando um atributo comum nomeado como chaves primárias e estrangeiras, sendo que a tabela que possui a restrição de chaves estrangeiras é nomeada tabela pai e a tabela referenciada é chamada tabela filha. (*Google Cloud,* 2023)

Além disso, ambas as chaves devem coexistir para a tabela filha, ou seja, não é permitido adicionar um valor de chave estrangeira na tabela filha caso este valor não exista na tabela pai.

Na Figura 9 é apresentada a ligação entre duas tabelas pela sua relação de chave estrangeira.

Figura 9 - Representação de conexão entre duas tabelas via Primary Key.



Fonte: Foreign[...]([2021?]).

2.7 Business Intelligence

Business Intelligence (BI) é o nome dado ao conjunto de estratégias, técnicas e tecnologias aplicadas que permitem tomada de decisões ágeis e precisas. As ferramentas de BI permitem acessar e obter informações a partir de uma grande quantidade de dados, apresentando-os em relatórios analíticos e gráficos, facilitando a compreensão das informações obtidas.

Com o objetivo de obter os melhores resultados, as estratégias de BI têm incluído mais processos para melhorar o seu resultado final. Esses processos podem ser listados a seguir.

Mineração de dados: Obtenção de dados de diversas fontes descentralizadas com a finalidade de obter informações relevantes sobre algum assunto. Essas fontes de dados não precisam se limitar apenas às informações internas da empresa, podendo incluir outras fontes externas, como redes sociais, relatórios públicos do governo e sites de notícias, entre outros tipos de sites que fornecem informações.

Tratamento dos dados e ETL: Após a obtenção das informações, esses dados precisam ser tratados para evitar redundância e ruídos que possam atrapalhar na análise da informação final.

Armazenamento de dados: Após o tratamento, as informações são enviadas para um banco de dados para centralizá-las e torná-las acessíveis.

Visualização: O BI pega dados descentralizados e desconexos e transforma-os em informações relevantes para o usuário final. Esses dados precisam ser apresentados de maneira clara e didática. Para isso, existem ferramentas de visualização que utilizam diversos métodos como gráficos para facilitar a compreensão do usuário final.

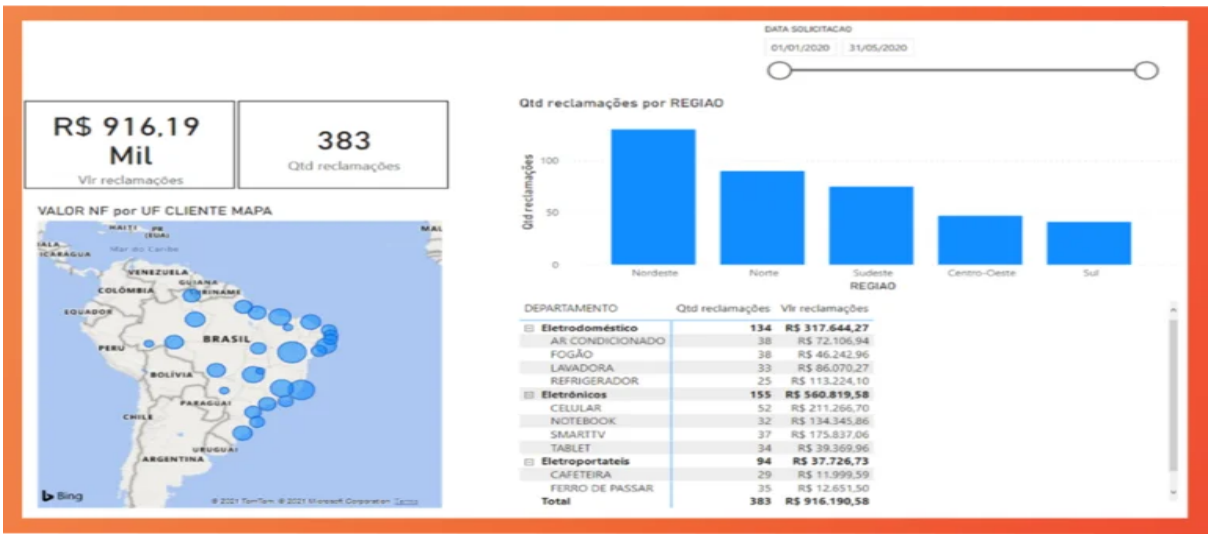
Dashboard: É o resumo dos dados em forma de interface gráfica, geralmente é a parte do dado que o usuário final terá contato. (Know Solution, [201–?])

2.7.1 Power Bi

O Power BI é uma plataforma de *Business Intelligence* (*BI*) que permite compartilhar informações através de relatórios interativos. Com uma interface gráfica intuitiva, o *software* é acessível para usuários de diferentes níveis de habilidade, possibilitando a geração rápida e fácil de informações. Além disso, o Power BI oferece diversas opções de compartilhamento de dados, permitindo que os *dashboards* sejam disponibilizados como arquivos ou via *web* em formato de sites. (Maia , 2021)

Na Figura 10 a seguir é possível observar um dashboard em Power BI.

Figura 10 - Relatório em Power BI.



Fonte: Maia (2021).

2.7.2 *Data Analysis Expressions*

O *Data Analysis Expressions* (DAX) é uma linguagem utilizada para manipular dados no Power BI. Com o DAX, é possível criar fórmulas personalizadas que permitem realizar cálculos e análises de dados de maneira fácil, sem necessitar manipular o dado de origem.

O DAX se baseia em três conceitos fundamentais: sintaxe, funções e contexto.

Sintaxe: Refere-se à estrutura da programação da fórmula, ou seja, as regras que a linguagem segue para funcionar corretamente. Por exemplo, a linguagem DAX sempre se inicia com um sinal de igual (=).

Funções: São códigos encapsulados que ao serem chamados executam certas ações especificadas. O DAX possui diversas funções pré-definidas que permitem ao usuário realizar diferentes ações sem a necessidade de criar a lógica por trás delas.

Contexto: O contexto pode ser dividido em três categorias: contexto de linha, contexto de consulta e contexto de filtro. (Nascimento, 2021)

- Contexto de linha: Responsável por analisar linhas isoladas em relação a tabela em geral. Por exemplo, é possível utilizar o contexto de linha para encontrar a maior quantidade de um produto em uma tabela.
- Contexto de consulta: Responsável pela análise inteira da tabela como um todo. Se esse tipo de contexto for aplicado a uma coluna, o resultado será o mesmo para todas as linhas.
- Contexto de filtro: Neste tipo de contexto, é possível especificar filtros para as fórmulas, permitindo calcular cada célula com base em critérios específicos.

2.8 Dados abertos

Os dados abertos governamentais são informações disponibilizadas gratuitamente por instituições públicas governamentais com a finalidade de manter a transparência das instituições perante os cidadãos. Esses dados são geralmente disponibilizados em sites do governo ou portais de transparência, permitindo fácil acesso da população para essas informações. Além disso, estes dados são licenciados de maneira que seu uso e redistribuição sejam livres, possibilitando análises e criações de soluções inovadoras por parte da população. (Ceweb, [202–?])

2.8.1 Base de RAIS

A base de Relação Anual de Informações Sociais (RAIS) é uma fonte de coleta de dados trabalhistas levantando dados estatísticos sobre atividades trabalhistas no Brasil. Ela é usada para obter informações como número de empregos com carteira de trabalho assinada, número de demissões, quais setores com maior crescimento, média salarial da população, entre outras informações relevantes referente a trabalho.

Na RAIS existem informações anonimizadas de todos os contribuintes contratados pelo regime de consolidação das leis trabalhistas (CLT) e contratos de trabalho por tempo indeterminado ou determinado. (Torres, 2023)

Esses dados são valiosos para análises e estudos sobre o mercado de trabalho e sua evolução ao longo do tempo.

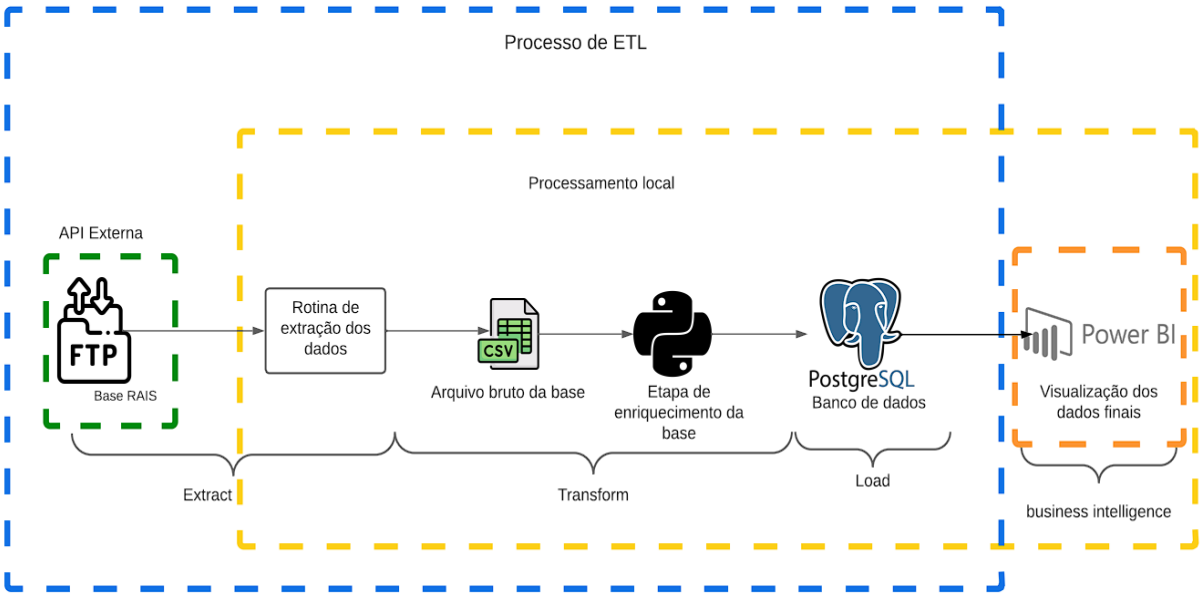
3 MATERIAIS e MÉTODOS

Ao longo deste trabalho, foram abordadas as etapas e os conceitos envolvidos no desenvolvimento do projeto. Nesse sentido, para a implementação do *pipeline* de dados, foram utilizadas diversas ferramentas e linguagens de programação, tais como Python, Spark, PostgreSQL e Power BI.

3.1 Arquitetura do projeto

Para a criação do projeto, foi utilizada a arquitetura representada na Figura 11.

Figura 11 - Arquitetura do projeto.



Fonte: Autoria própria.

Na Figura 11 é possível observar duas estruturas distintas: uma em verde, que representa um ambiente externo acessado por meio de uma *API*, e outra em amarelo, que representa o ambiente de processamento local, ou seja, o computador utilizado.

A execução de todo o processo ETL, destacada na estrutura azul, se dará pela chamada do comando ***./execute.sh*** localizado no agendador de rotina dentro do *crontab*. Esse comando será responsável por invocar uma linha de comando em Shell Script, que executa a criação de diretórios, o *download* dos documentos a serem trabalhados, a chamada dos códigos em Python e enviar os dados enriquecidos para uma tabela no banco de dados PostgreSQL. Em seguida, na

estrutura laranja, os dados serão consumidos por dashboards utilizando a ferramenta Power BI.

3.2 *Download* automático da base de dados

A fim de automatizar completamente o processo, foi necessário reproduzir todos os passos que um ser humano executaria para processar a base de dados. Isso inclui o *download* dos arquivos, a criação de diretório para armazenamento e o enriquecimento dos dados obtidos. Para alcançar esta automação, foi criado um arquivo em Shell Script que irá realizar todas essas operações quando executado.

3.2.1 Funcionamento do código de orquestração.

O código de orquestração foi dividido em 5 etapas, nessas etapas são responsáveis por orquestrar o download automático da base, preparar diretórios para armazenar os arquivos e também executar os *pipelines* de enriquecimentos:

Etapa 1 (Figura 12): É feita a referência a uma variável chamada *YEAR_DOWNLOAD* para armazenar o penúltimo ano de referência. Em seguida, é criado um diretório com o nome do ano para armazenar a base de entrada daquele ano. Além disso, essa etapa verifica se o diretório definido por *directory* existe. Se o diretório não existir, o *script* cria o diretório. Se o diretório existir, mas não for um diretório, o *script* emite uma mensagem de erro.

Figura 12 -Processo de criação de diretório.

```
YEAR_DOWNLOAD="$(date +"%Y" -d "2 year ago")"


export directory=$YEAR_DOWNLOAD

if [[ ! -e $directory ]]; then
    mkdir $directory
elif [[ ! -d $directory ]]; then
    echo "$dir already exists but is not a directory" 1>&2
fi

cd  $directory
```

Fonte: Autoria própria.

Etapa 2 (Figura 13): O *script* acessa o diretório definido pela variável *directory* e inicia um loop infinito usando o comando *while*. Dentro do loop, o *script* verifica se a URL do servidor FTP está funcionando. Se sim, o *script* utiliza o comando *wget* para baixar arquivos do servidor, excluindo aqueles que contenham a nomenclatura RAIS_ESTAB_. Caso a URL não esteja funcionando, o *script* aguarda por 200 horas e tenta novamente, repetindo esse processo até que o arquivo seja baixado com sucesso.

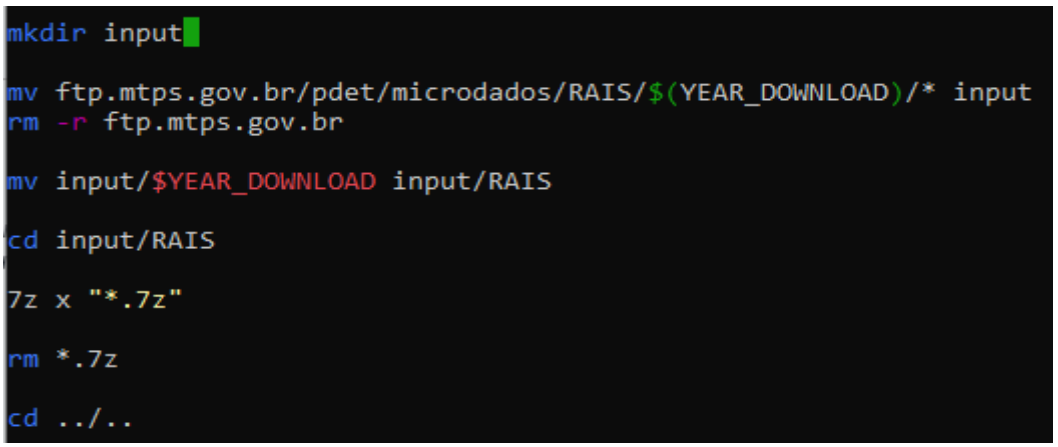
Figura 13 - Download da base de dados que será trabalhada.

```
while true; do
    if wget -q --reject-regex 'RAIS_ESTAB_' -r "ftp://ftp.mtps.gov.br/pdet/microdados/RAIS/$YEAR_DOWNLOAD/"; then
            echo "url is working"
            break
    else
            echo "url is not working "
            sleep  200h
    fi
done
```

Fonte: Autoria própria.

Etapa 3 (Figura 14): Etapa responsável por extrair os arquivos baixados, excluir as redundâncias e movê-los para um diretório de entrada padronizado.

Figura 14 - Preparação e padronização do diretório de entrada.



```
mkdir input

mv ftp.mtps.gov.br/pdet/microdados/RAIS/$(YEAR_DOWNLOAD)/* input
rm -r ftp.mtps.gov.br

mv input/$YEAR_DOWNLOAD input/RAIS

cd input/RAIS

7z x "*.7z"

rm *.7z

cd ../..
```

Fonte: Autoria própria.

Etapa 4 (Figura 15): Verifique se o diretório de saída do *pipeline* existe. Se não existir, o *script* cria o diretório. Se existir, mas não for um diretório, o *script* emite uma mensagem de erro.

Figura 15 - Preparação do diretório de saída.

```
if [[ ! -e saida_municipio ]]; then
    mkdir saida_municipio
elif [[ ! -d saida_municipio ]]; then
    echo "$dir already exists but is not a directory" 1>&2
fi

if [[ ! -e saida ]]; then
    mkdir saida
elif [[ ! -d saida ]]; then
    echo "$dir already exists but is not a directory" 1>&2
fi

export diretorio_entrada=$YEAR_DOWNLOAD
```

Fonte: Autoria própria.

Etapa 5 (Figura 16): Executa os *scripts* python responsáveis pelo enriquecimento do arquivo.

Figura 16 - Execução dos *pipelines*.

```
python3 ./pipeline_enriquecimento_base.py
python3 ./media_rais_cbo_municipio.py
```

Fonte: Autoria própria.

3.3 Enriquecimento do arquivo de entrada.

Com a finalidade de deixar o arquivo pronto para visualização utilizando os *dashboards*, é necessário realizar uma limpeza dos dados. Essa etapa é responsável por remover informações desnecessárias, eliminar ruídos e dados irrelevantes que possam atrapalhar na análise *BI*.

Para realizar esta etapa, foram criados dois *scripts* em Python para o enriquecimento dos dados. O primeiro *script* é responsável pela higienização dos dados, enquanto o segundo *script* é responsável por agrupar as informações e enviar os dados aprimorados para o banco de dados.

3.3.1 Higienização da base de entrada

O *script* Python a ser abordado por esta etapa utiliza PySpark para processar os arquivos armazenados no diretório de entrada. O objetivo inicial é extrair

informações de remuneração média por ocupação e sexo de cada município brasileiro.

O código é dividido em 4 etapas, sendo essas:

Etapa 1 (Figura 17)**:** Importa as bibliotecas necessárias para o funcionamento do *script*, tais como PySpark, Pandas e OS (responsável pela comunicação com o sistema operacional). Logo após, uma sessão Spark é criada para o processamento dos dados. Estes dados estão localizados em um diretório de entrada importado de uma variável ambiente exportada pelo *script* de orquestração.

Figura 17 **-** *Script* em Python de importação de bibliotecas.

```
import pyspark
from pyspark.sql import SparkSession
from pyspark.sql.functions import substring
from pyspark.sql import SQLContext
from pyspark.sql.functions import col
from pyspark.sql.functions import count, avg, regexp_replace, round, expr,when
from pyspark.sql import functions as F
import IPython
import pyspark
import pandas
import pandas as pd
from pathlib import Path
from pyspark.sql.functions import trim
import glob, os
```

Fonte: Autoria própria.

Etapa 2 (Figura 18)**:** Nesta etapa o *script* lista todos os arquivos em formato *.txt* que estão localizados no diretório de entrada e armazena seus caminhos em uma lista chamada *fileList*. Para cada elemento da lista, o *script* lê o arquivo usando o PySPark, definindo algumas opções de formatação.

Figura 18 - *Script* responsável pela leitura dos arquivos de entrada.

```
fileList=[]
for file in glob.glob(f"{diretorio}\\input\\RAIS\\*.txt"):
    fileList.append(file)

for arquivo in fileList:
    df = (spark.read
                        .option("header", "true")
                        .option("encoding", "ISO-8859-1")
                        .option("sep", ";")
                        .option("recursiveFileLookup", "true")
                        .csv(path=arquivo)
                )
```

Fonte: Autoria própria.

Etapa 3 (Figura 19 e Figura 20): Nesta etapa o código realiza uma série de transformações nos dados lidos pela Etapa 2, essas transformações incluem a remoção de espaços em branco, conversão de casas decimais, filtragem de ruídos como remuneração igual a zero e agrupamentos dos dados por UF, municípios, ocupação e sexo. Em seguida, é calculada a média de remuneração e o número de trabalhadores em cada grupo.

Figura 19 - Higienização da base de entrada e retirada de ruídos.

```
df = df.withColumn('Município',when(col('Mun Trab') =='000000', col('Município'))
                                            .otherwise(col('Mun Trab')))

df = df.withColumn('UF', substring('Município',1, 2))

for coluna in df.columns:
    df = df.withColumn(coluna, trim(col(f'`{coluna}`')))



df_filtrado_campos_uteis=df.select(['UF','Município',
                  'CBO Ocupação 2002','Vl Remun Média Nom', 'Sexo Trabalhador'])
```

Fonte: Autoria própria.

Figura 20 - Formatação de campos e agrupamento de características.

```
df=(df.withColumn("remuneracao",regexp_replace('Vl Remun Média Nom', ',','.'))
      .withColumn('remuneracao_media',col('remuneracao').cast('double'))
   )

df_grouped = (df.groupBy('UF','Município', 'CBO Ocupação 2002','Sexo Trabalhador')
                        .agg(F.mean('remuneracao_media').alias('Media_remuneracao'),
                             F.count('remuneracao_media').alias('SomaAgrupamento')))

df_grouped = df_grouped.withColumn('MediaRemuneracaoCBO', round(col('Media_remuneracao'), 2))

df_final = df_grouped.select(['UF',
                            'Município',
                            'CBO Ocupação 2002',
                            'Sexo Trabalhador',
                            'MediaRemuneracaoCBO',
                            'SomaAgrupamento'])
```

Fonte: Autoria própria.

Etapa 4 (Figura 21): Por fim, os resultados são salvos em arquivos CSV dentro do diretório de saída para cada arquivo de entrada processado.

Figura 21 - Salvamento do arquivo final.

```
path_result = f".\\{diretorio}\\saida_municipio\\"+arquivo.split('\\')[-1]
df_final.toPandas().to_csv(path_result)
```

Fonte: Autoria própria.

O arquivo de saída ficará com formato apresentado na Figura 22 e por isso passará por mais um tratamento.

Figura 22 - Exemplo de saída do arquivo final deste *pipeline*.

```
,UF,Município,CBO Ocupação 2002,Sexo Trabalhador,MediaRemuneracaoCBO,SomaAgrupamento
0,11,110001,141305,01,2515.18,1
1,11,110001,142105,01,907.35,2
2,11,110001,142210,02,1229.79,1
3,11,110001,142305,02,1433.34,1
4,11,110001,223605,02,2475.31,2
5,11,110001,223710,02,5515.26,1
6,11,110001,231205,01,837.22,1
7,11,110001,232105,01,1362.66,1
8,11,110001,232110,01,601.51,1
9,11,110001,232110,02,600.63,1
10,11,110001,232130,01,1627.78,1
```

Fonte: Autoria própria.

3.3.2 Padronização do arquivo e processamento na base de dados

O *script* Python abordado nesta etapa lê os arquivos processados pela etapa anterior e faz transformações e agregações dos dados utilizando a biblioteca PySpark. Além disso, ele é responsável por realizar o *deploy* do arquivo a uma base de dados PostgreSQL.

As duas primeiras etapas se repetem em relação ao arquivo anterior, tendo como diferença o uso da biblioteca *SQLAlchemy* responsável por comunicar com o banco de dados e também o arquivo que será lido como entrada, Figura 23.

Figura 23 - *Script* em Python de importação de bibliotecas.

```
import pandas as pd
import glob
import pyspark
import pyspark.sql.functions as f
from pyspark.sql.functions import col
from pyspark.sql import SparkSession, DataFrame
from functools import reduce
import sys, os
from sqlalchemy import create_engine

spark  = (SparkSession.builder.appName('RAIS')
          .getOrCreate())
diretorio = os.environ["diretorio_entrada"]
df_list = []
path_input = f".\\{diretorio}\\saida_municipio\\"
for file in glob.glob(path_input+"*.txt"):
    ano = path_dir.split('_')[1]
    df = (spark.read
                    .option("header", "true")
                    .option("encoding", "UTF-8")
                    .option("sep", ",")
                    .option("recursiveFileLookup", "true")
                    .csv(path=file)
              )
    df = df.withColumn('ano', f.lit(ano))
    df_list.append(df)
df = reduce(DataFrame.unionAll, df_list)
cwd = os.getcwd()
arquivo = cwd +'\CBO_2.txt'
arquivo_sexo = cwd +'\sexo.txt'
arquivo_municipio = cwd + '\Municipio.txt'
```

Fonte: Autoria própria.

Após isto, é adicionada uma coluna *ano* com o ano correspondente ao arquivo. Todos os *DataFrames* carregados são agregados em um único *DataFrame*. Logo após, o código carrega outros arquivos CSV responsáveis por vincular a informação em texto aos códigos de município, sexo e Classificação brasileira de ocupação, e assim deixando o dado mais fácil de se compreender, Figura 24.

Figura 24 - Importação dos arquivos para agrupamento.

```
df_sexo =  (spark.read
                        .option("header", "true")
                        .option("encoding", "UTF-8")
                        .option("sep", ",")
                        .option("recursiveFileLookup", "true")
                        .csv(path=arquivo_sexo)
                )

df_municipio =  (spark.read
                        .option("header", "true")
                        .option("encoding", "UTF-8")
                        .option("sep", ":")
                        .option("recursiveFileLookup", "true")
                        .csv(path=arquivo_municipio)
                )
df_cbo = (spark.read
                        .option("header", "true")
                        .option("encoding", "ISO-8859-1")
                        .option("sep", ";")
                        .option("recursiveFileLookup", "true")
                        .csv(path=arquivo)
                )

df = df.orderBy(col('Município').asc(),col('CBO Ocupação 2002').asc(),col('Sexo Trabalhador').asc())
df = df.join(df_cbo, df["CBO Ocupação 2002"] == df_cbo["Codigo_CBO"], "inner")
df = df.join(df_municipio, df["Município"] == df_municipio["codigo_municipio"], "inner")
df = df.join(df_sexo, df["Sexo Trabalhador"] == df_sexo["id_sexo"], "inner")

df_final = df[['ano','codigo_municipio','Municipio',' sexo','Codigo_CBO','CBO 2002
Ocupação','MediaRemuneracaoCBO','SomaAgrupamento']]
```

Fonte: Autoria própria.

Por fim, o arquivo final é salvo em um CSV para futuros *backups* e também é enviado para a tabela de dados no banco de dados PostgreSQL, Figura 25.

Figura 25 - *Script* em Python de importação de bibliotecas.

```
path_result = f'.\\{path_dir}\\saida\\arquivo_final.txt'
df_final.toPandas().to_csv(path_result, index= False)
engine = create_engine('postgresql://postgres:admin@localhost:5432/postgres')
df_final.to_sql("cbo_por_municipio", engine, schema="rais", if_exists="append", index=False)
```

Fonte: Autoria própria.

3.4 Banco de dados PostgreSQL

Com a finalidade de inserir novos dados com uma certa frequência, tornou-se inviável modificar um arquivo de texto sempre que novos dados surgissem para serem inseridos. Deste modo, optou-se pelo uso de um banco de dados relacional chamado PostgreSQL. Esse banco de dados permite com que dados sejam inseridos de maneira fácil e prática, também permite com que buscas personalizadas sejam realizadas somente ao especificar os campos necessários.

Assim, foi estruturada uma tabela utilizando a interface gráfica do banco de dados PostgreSQL chamada PgAdmin, Tabela 2.

Tabela 2 - Estrutura da tabela cbo_por_municipio.

tabela: cbo_por_municipio		
coluna	descrição	tipo
ano	ano de referência do dado	bigint
codigo_municipio	código IBGE do município	bigint
municipio	nome do município referente ao código	character varying
sexo	sexo do grupo	character varying
codigo_cbo	Classificação Brasileira de Ocupações	bigint
cbo_2002_ocupacao	Descrição da ocupação	character varying
media_remuneracao_cbo	Média da remuneração referente ao sexo, municipio, ano e cbo	numeric
soma_servidores	soma dos trabalhadores utilizados para realizar a média	bigint

Fonte: Autoria própria.

Após o *insert* dos arquivos trabalhados, a tabela ficará conforme a Tabela 3.

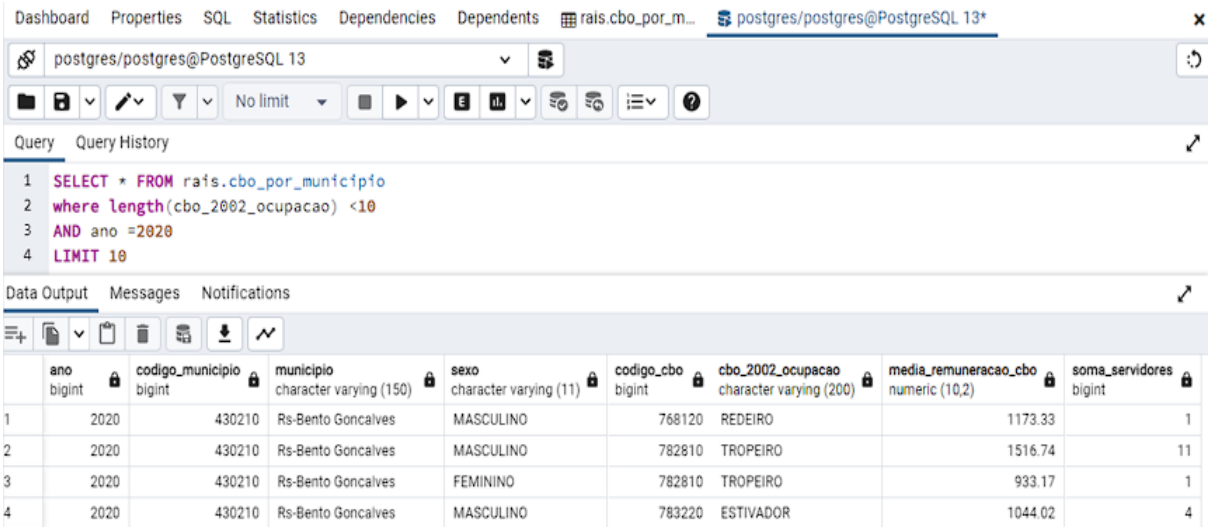
Tabela 3 - Seleção de 10 linhas da tabela PostgreSQL.

ano	codigo_municipio	municipio	sexo	codigo_cbo	cbo_2002_ocupacao	media_remuneracao_cbo	soma_servidores
2020	430210	Rs-Bento Goncalves	MASCULINO	768120	REDEIRO	1173.33	1
2020	430210	Rs-Bento Goncalves	MASCULINO	782810	TROPEIRO	1516.74	11
2020	430210	Rs-Bento Goncalves	FEMININO	782810	TROPEIRO	933.17	1
2020	430210	Rs-Bento Goncalves	MASCULINO	783220	ESTIVADOR	1044.02	4
2020	430210	Rs-Bento Goncalves	MASCULINO	841745	XAROPEIRO	2752.43	1
2020	430210	Rs-Bento Goncalves	MASCULINO	848305	PADEIRO	2373.83	69
2020	430210	Rs-Bento Goncalves	FEMININO	848305	PADEIRO	1801.55	29
2020	430210	Rs-Bento Goncalves	MASCULINO	848520	MAGAREFE	2704.68	3
2020	430215	Rs-Boa Vista das Missoes	MASCULINO	241005	ADVOGADO	924.16	1
2020	430215	Rs-Boa Vista das Missoes	MASCULINO	252210	CONTADOR	7166.29	1

Fonte: Autoria própria.

Além do mais, o PostgreSQL possui uma ferramenta que facilita a manipulação de dados chamada PgAdmin, possibilitando com que o usuário acesse determinados tipos de dados apenas comando chamados *Query*, como pode se observar na figura 26.

Figura 26 - Interface gráfica da ferramenta PgAdmin.



	ano bigint	codigo_municipio bigint	municipio character varying (150)	sexo character varying (11)	codigo_cbo bigint	cbo_2002_ocupacao character varying (200)	media_remuneracao_cbo numeric (10,2)	soma_servidores bigint
1	2020	430210	Rs-Bento Goncalves	MASCULINO	768120	REDEIRO	1173.33	1
2	2020	430210	Rs-Bento Goncalves	MASCULINO	782810	TROPEIRO	1516.74	11
3	2020	430210	Rs-Bento Goncalves	FEMININO	782810	TROPEIRO	933.17	1
4	2020	430210	Rs-Bento Goncalves	MASCULINO	783220	ESTIVADOR	1044.02	4

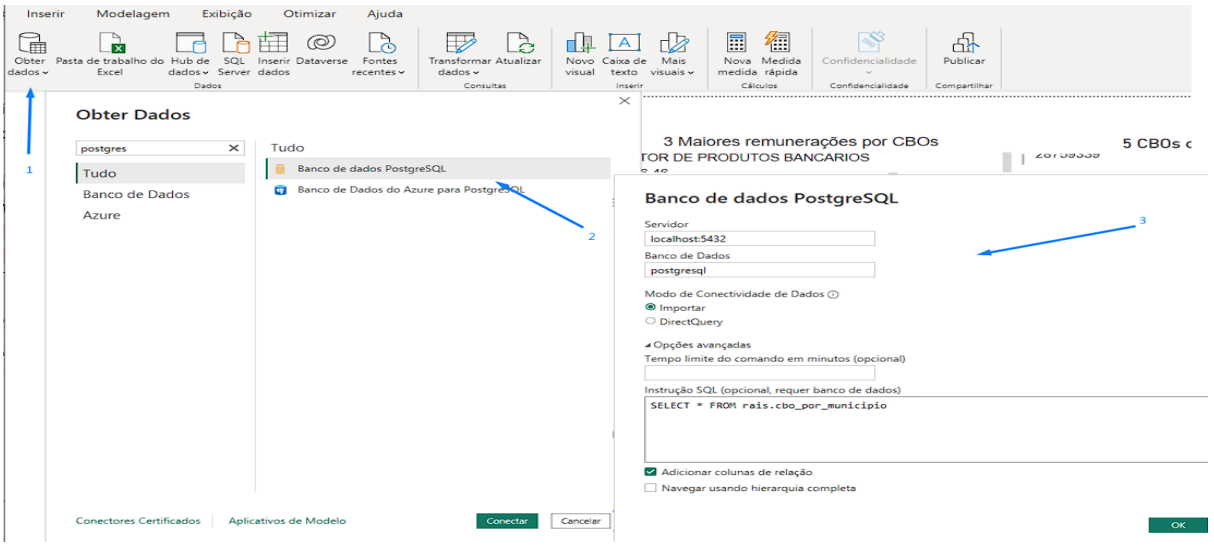
Fonte: Autoria própria.

3.5 Criação de dashboard em *Power BI*

Para a visualização dos dados inseridos no banco, utilizou-se a ferramenta Power BI para a criação dos *dashboard* para apresentação de informações relevantes sobre as informações salariais dos trabalhadores brasileiros.

Para o funcionamento do *dashboard* foi realizada uma conexão com o banco de dados PostgreSQL, para isso o Power BI fornece ferramentas que facilitam esta conexão**.** Sendo que o processo é realizado por etapas mostradas pelas setas indicadas na figura 27:

Figura 27 **–** Conexão da ferramenta Power bi com o banco de dados PostgreSQL.



Fonte: Autoria própria.

Etapa 1 e 2: Responsável por abrir o campo de conexão do Power BI.

Etapa 3: Responsável por passar as informações que farão a conexão ao banco de dados, tais como servidor e *query* de seleção.

Com as três etapas realizadas o dado em sua forma natural será disponibilizado para visualização e utilizando a linhagem *DAX* o BI irá fazer manipulações para a retornar novas informações a partir dos dados originais.

Depois de estabelecida a conexão com a base de dados, deu-se início ao processo de criação dos gráficos e indicadores analíticos.

As informações mostradas pelo relatório BI são gerenciadas a partir de um filtro dinâmico. Nesse filtro possui os campos de ano de referência do dado, Estado e município e CBO.

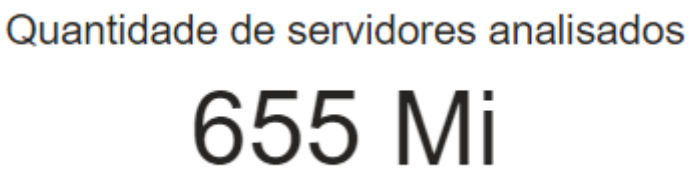
Figura 28- Filtro dinâmico do relatório BI.

Fonte: Autoria própria.

Além do mais, o relatório BI possui 6 indicadores e 4 gráficos que possuem funções distintas a fim de levantar informações relevantes para o leitor. Sendo essas informações mostradas a seguir:

Indicador 1 (Figura 29) - Responsável por mostrar a soma de trabalhadores analisados com base nos filtros aplicados.

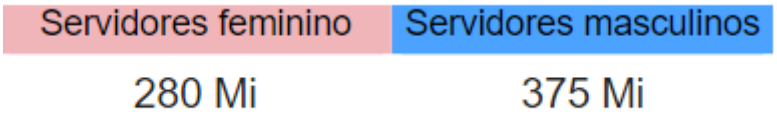
Figura 29- Soma de servidores analisados.



Fonte: Autoria própria.

Indicador 2 (Figura 30) - Responsável por mostrar a soma dos servidores masculinos e femininos com base nos filtros aplicados.

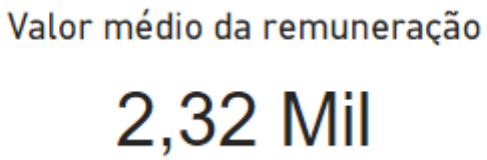
Figura 30**- Soma de servidores discriminados por sexos.**



Fonte: Autoria própria.

Indicador 3 (Figura 31) - Responsável por analisar a média salarial dos servidores com base nos filtros selecionados.

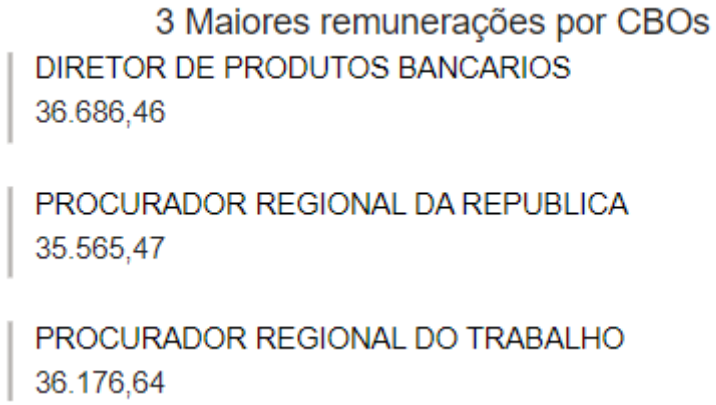
Figura 31 **-** Média salarial dos servidores.



Fonte: Autoria própria.

Indicador 4 (Figura 32) - Responsável por indicar as 3 maiores remunerações de classificação brasileira de ocupação com base nos filtros aplicados.

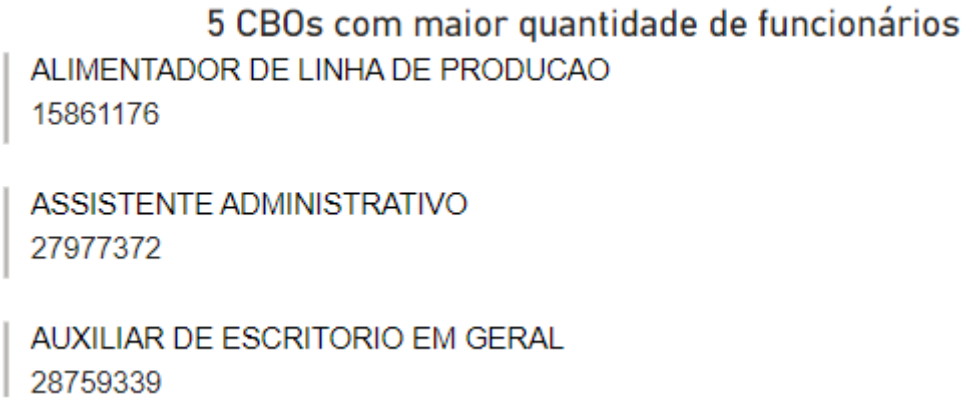
Figura 32 - Maiores remunerações por CBO.



Fonte: Autoria própria.

Indicador 5 (Figura 33) - Responsável por indicar os cinco CBOs com maior quantidade de funcionários com base nos filtros aplicados.

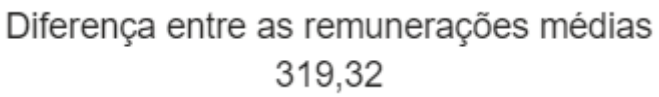
Figura 33 - Classificação de categorias com maior quantidade de funcionários.



Fonte: Autoria própria.

Indicador 6 (Figura 34) - Diferença entre remuneração de homens e mulheres com base nos filtros aplicados.

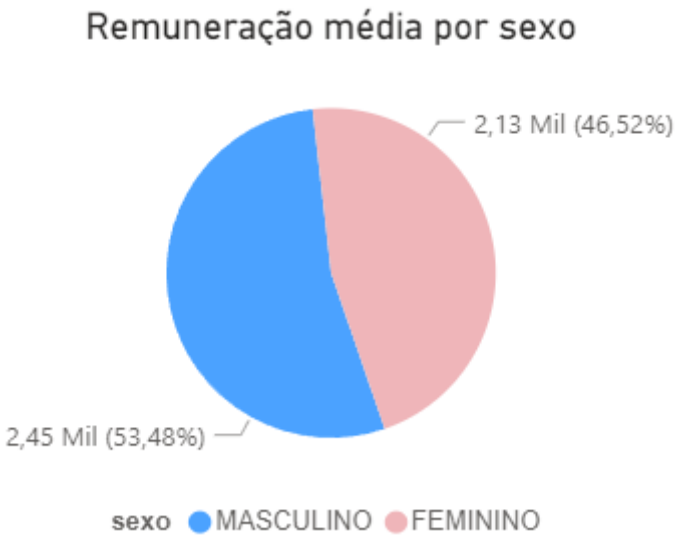
Figura 34 -Diferença entre remuneração média entre homens e mulheres.



Fonte: Autoria própria.

Gráfico 1 (Figura 35): Gráfico de pizza responsável por mostrar visualmente a diferença salarial entre homens e mulheres.

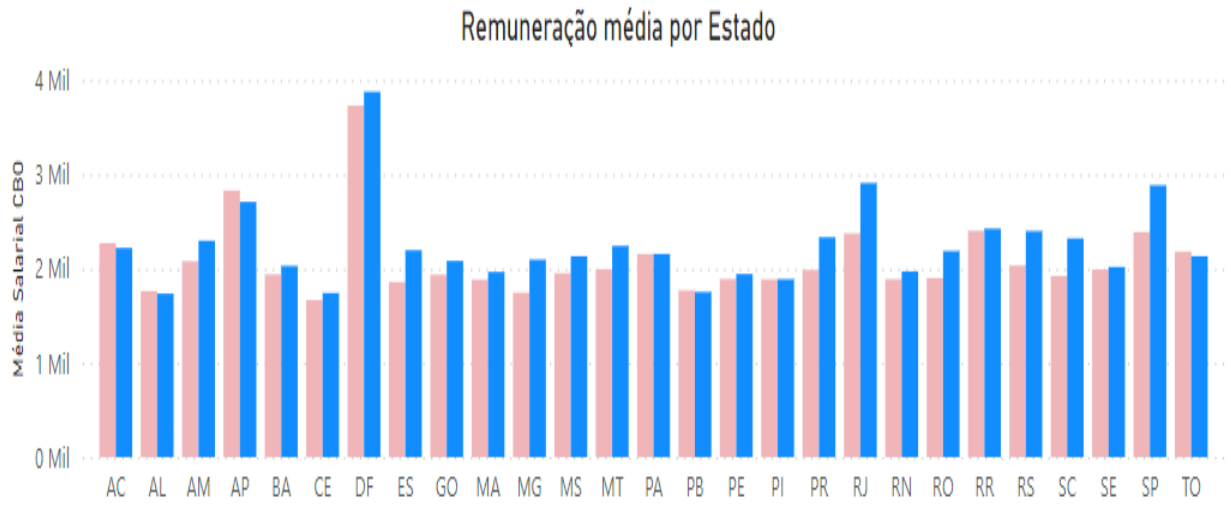
Figura 35- Gráfico de pizza.



Fonte: Autoria própria.

Gráfico 2 (Figura 36): Gráfico de colunas clusterizadas responsável por mostrar a remuneração de homens e mulheres de acordo com os estados de referências, alterando os resultados de acordo com os filtros de CBO e ano de referência.

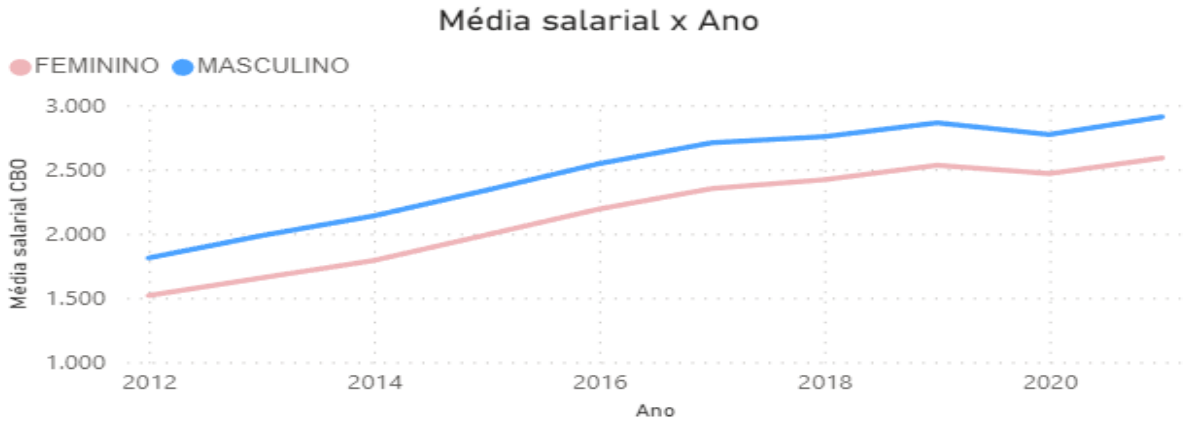
Figura 36 - Gráfico de colunas clusterizadas.



Fonte: Autoria própria.

Gráfico 3 (Figura 37): Gráfico de linha responsável por mostrar a diferença salarial entre homem e mulher ao decorrer dos anos, podendo mostrar diferentes resultados ao selecionar outros CBOs e municípios.

Figura 37 - Gráfico de linha.



Fonte: Autoria própria.

Gráfico 4 (Figura 38): Gráfico de mapas responsável por mostrar a localização dos servidores ao redor do Brasil. Neste gráfico o tamanho da bolha irá representar a quantidade de trabalhadores para determinadas regiões. É válido destacar que o gráfico demonstrado é interativo e mostra as volumetrias à medida que se aproxima da visualização das áreas, logo, em sua visão geral como observamos abaixo, pode-se observar apenas as bolhas nas áreas com maior volume de trabalhadores, enquanto nas com menor são observadas à medida que se aproxima da imagem dentro da sua interação do *BI*.

Figura 38 - Gráfico de mapa.

Fonte: Autoria própria.

4 RESULTADOS e DISCUSSÕES

O resultado é trazido para melhor entendimento dos indicadores e o que cada tela significa. Além do mais permite expor entendimentos e contextualizações dos dados obtidos.

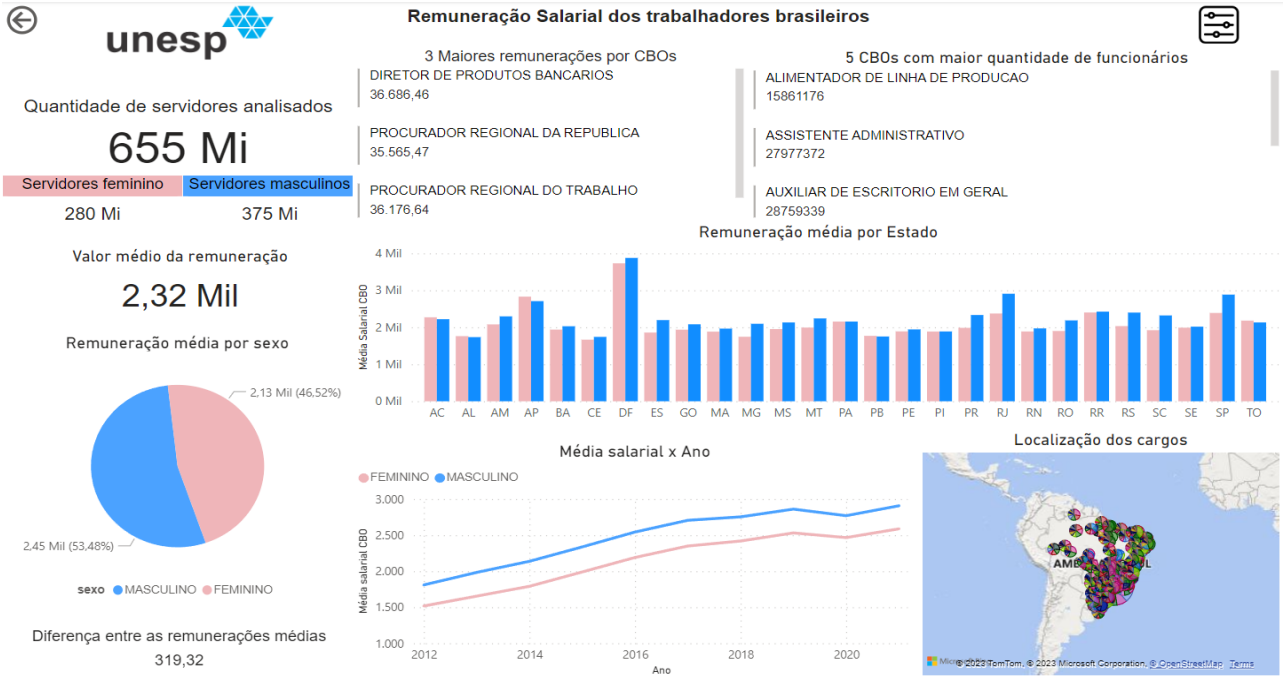
4.1 Análise dos dados utilizando o *Power BI*

Com os dados tratados e integrados via banco de dados ao *Power BI*, desenvolveu-se um relatório com todas as informações necessárias para a análise e conclusões sobre a diferença salarial entre homens e mulheres no Brasil em relação a diversas categorias profissionais, anos e municípios.

4.1.1 Visão macro do relatório *BI*

Na análise macro do relatório apresentado na Figura 39, nenhum filtro de seleção foi aplicado e todos os dados foram incluídos na análise. O período de referência foi de 2012 a 2021 e um total de 655 milhões de dados de trabalhadores foram analisados, dos quais 280 milhões de dados eram de mulheres e 375 milhões eram de homens. Durante esse período, a remuneração média mensal no Brasil foi de 2.320,00 reais e a diferença entre as remunerações médias foi de aproximadamente 320,00 reais. Também é possível observar as três maiores médias de remunerações por CBO e os cinco CBOs possuem mais dados de trabalhadores cadastrados.

Figura 39 - Visão macro do relatório na ferramenta Power BI.



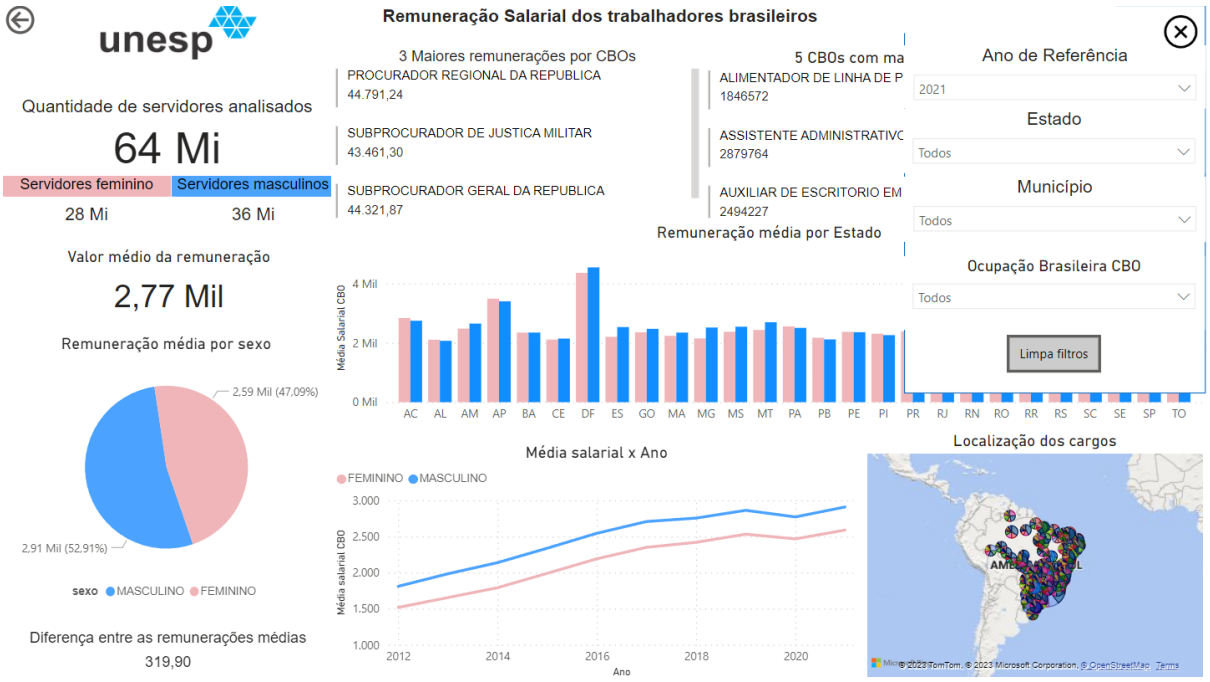
Fonte: Autoria própria.

Além disso, pode-se perceber, ao analisar o gráfico de colunas, que há uma diferença salarial entre os estados, sendo que o Distrito Federal apresenta a maior média salarial para seus servidores, enquanto os demais estados apresentam uma consistência em seus valores. Por fim, o gráfico de linhas mostra claramente que, em geral, a média salarial dos servidores brasileiros do sexo masculino é sempre superior à média salarial dos servidores do sexo feminino.

4.1.2 Relatório com base no ano de 2021.

Ao examinar o relatório referente a 2021, conforme ilustrado na Figura 40, observa-se que foram analisados um total de 64 milhões de servidores. Dentre esses servidores, cerca de 28 milhões são mulheres e 36 milhões são homens. Percebe-se que as tendências anteriores se mantêm, com os servidores do sexo masculino continuando a receber salários mais altos em grande parte dos estados. A média salarial para os servidores, considerando ambos os sexos, é de aproximadamente 2.770,00 reais.

Figura 40 - Visão dos indicadores e gráficos do relatório ao utilizar o ano de 2021 como referência.



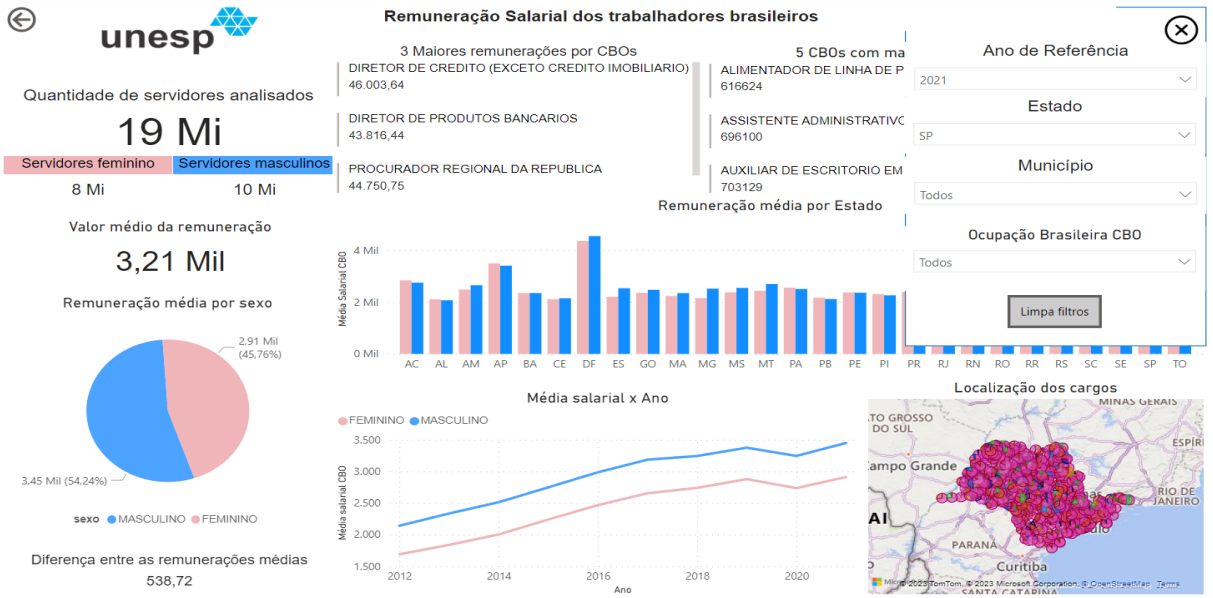
Fonte: Autoria própria.

4.1.3 Relatório com base no ano de 2021 e Estado de São Paulo.

Ao analisar o relatório utilizando o ano de 2021 como referência e focando no Estado de São Paulo, conforme mostrado na Figura 41, é possível observar que a média salarial para esse estado é maior em comparação com a média geral de todos os outros estados. Em São Paulo, a média salarial é de aproximadamente 3.210,00 reais, enquanto a média geral calculada foi de 2.770,00 reais.

Além disso, ao analisar a diferença salarial entre homens e mulheres em São Paulo, nota-se que essa diferença é maior em comparação com a média geral. A diferença salarial entre os sexos em São Paulo é de cerca de 540,00 reais, o que representa um acréscimo de 220,00 reais em relação à média geral, que é de 320,00 reais.

Figura 41 - Visão dos indicadores e gráficos do relatório ao utilizar o ano de 2021 e Estado de São Paulo como referência.



Fonte: Autoria própria.

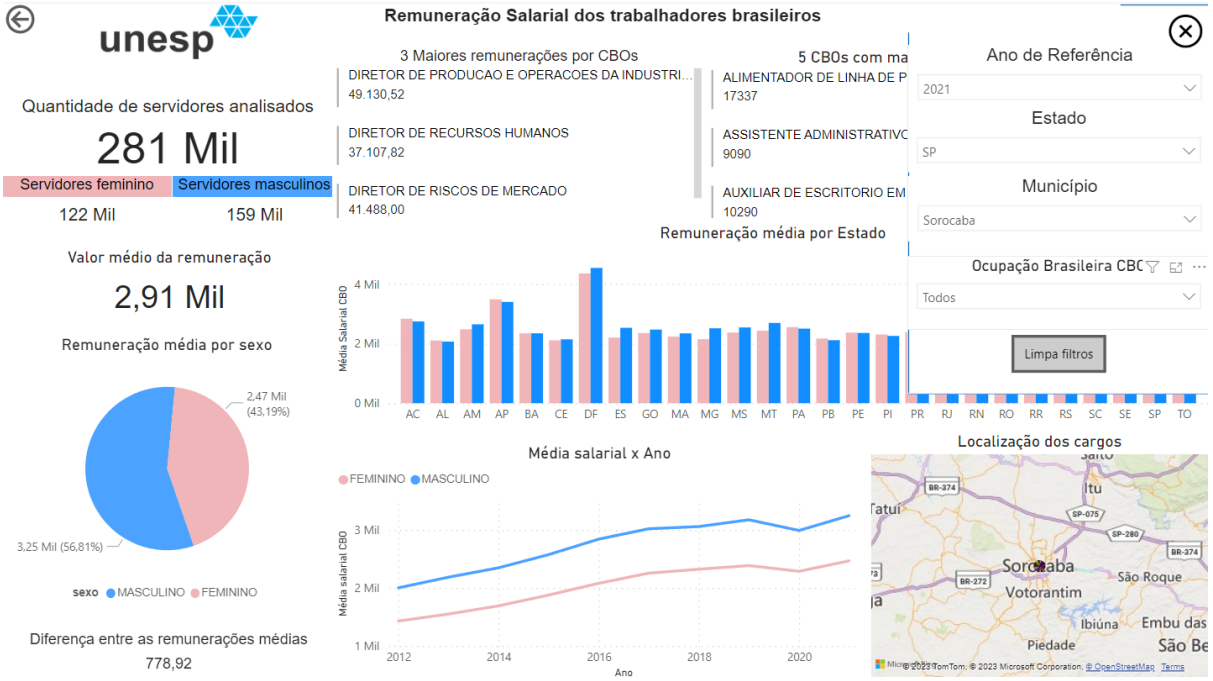
4.1.4 Relatório com base no ano de 2021 e município de Sorocaba.

Ao analisar o Relatório com filtros de ano igual a 2021, e município de Sorocaba, conforme mostrado na Figura 42 percebe-se que a média salarial para Sorocaba, que seria de 2.910,00 reais é inferior à média geral do Estado de São Paulo, de 3.210,00, havendo uma diferença de aproximadamente 300,00 reais.

Pode-se também analisar que a diferença salarial entre homens e mulheres é muito maior em Sorocaba em comparação à média geral de todos os outros Estados, sendo esta diferença de aproximadamente 780,00 reais, 460,00 reais de diferença em comparação a média geral do Brasil.

Para o município de Sorocaba, o crescimento salarial, ao comparar o ano de 2012 e 2021, foi de aproximadamente 1.000,00 reais para as mulheres e 1.200,00 reais para os homens, sendo que a média salarial das mulheres sempre foi inferior à dos homens analisados.

Figura 42 - Visão dos indicadores e gráficos do relatório ao utilizar o ano de 2021 e município de Sorocaba como referência.



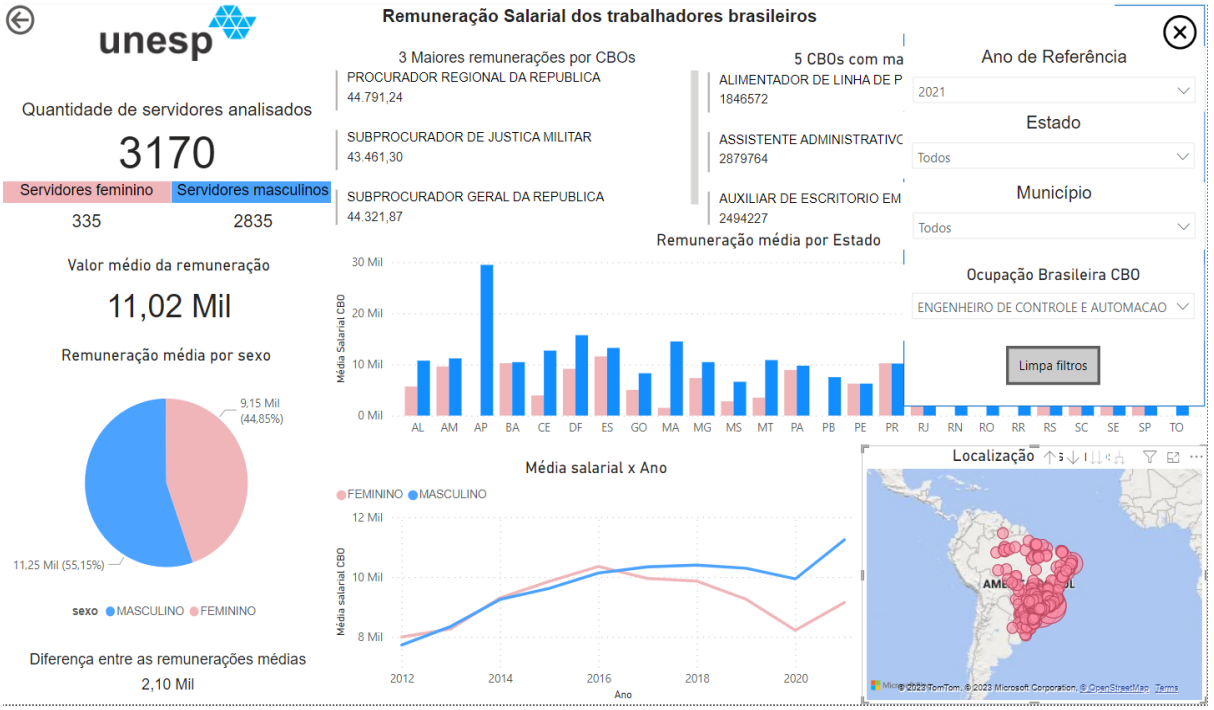
Fonte: Autoria própria.

4.1.5 Salário de pessoas engenheiras de controle e automação com base no ano de 2021.

Ao examinar os trabalhadores inscritos sob o CBO de controle e automação no ano de 2021, conforme apresentado na Figura 43, foi identificada uma disparidade salarial significativa de 2.100,00 reais entre homens e mulheres. A remuneração média para esses profissionais foi de aproximadamente 11.000,00 reais.

Além disso, é importante ressaltar que o número de registros de trabalhadores nessa categoria foi menor do que o esperado, totalizando apenas 3.170 profissionais. Essa baixa quantidade pode ser atribuída, em parte, ao fato de que muitos engenheiros de controle e automação trabalham nessa área, mas são registrados com outros CBOs (Classificação Brasileira de Ocupações) para que as empresas não precisem cumprir o piso salarial exigido pelos sindicatos.

Figura 43 - Visão dos indicadores e gráficos do relatório ao utilizar o ano de 2021 e CBO de engenheiro de controle e automação como filtros.



Fonte: Autoria própria.

4.1.6 Engenharia de dados na compreensão da informação

A partir dos resultados apresentados, fica evidente que este projeto oferece a capacidade de analisar várias variáveis e combinar essas informações para obter uma ampla gama de resultados diferentes. Isso se tornou possível graças às novas tecnologias disponíveis no campo da engenharia de dados. O projeto conseguiu lidar com um grande volume de dados dispersos, aproximadamente 700 milhões, utilizando um computador com configurações medianas. Esse tipo de processamento seria impossível de ser realizado no passado.

As combinações geradas permitiram obter diversos resultados relacionados à diferença salarial por sexo, região, classificação de ocupação e ano no Brasil. Essas informações foram obtidas a partir de um grande conjunto de dados, o que possibilitou a realização de análises diversas e a compreensão dessas informações de maneira mais profunda.

5 CONCLUSÃO

O desenvolvimento deste projeto possibilitou aprofundar os conhecimentos relacionados a big data, ETL (Extração, Transformação e Carga) e *Business Intelligence*. O resultado final do trabalho permitiu visualizar a diferença salarial entre homens e mulheres, revelando como esse cenário se comporta em diferentes regiões e profissões.

Uma análise temporal mostrou que, ao comparar as médias salariais, os homens ganharam mais do que as mulheres em todos os anos e em quase todos os estados brasileiros. A diferença salarial geral se manteve constante ao longo do tempo, variando muito pouco.

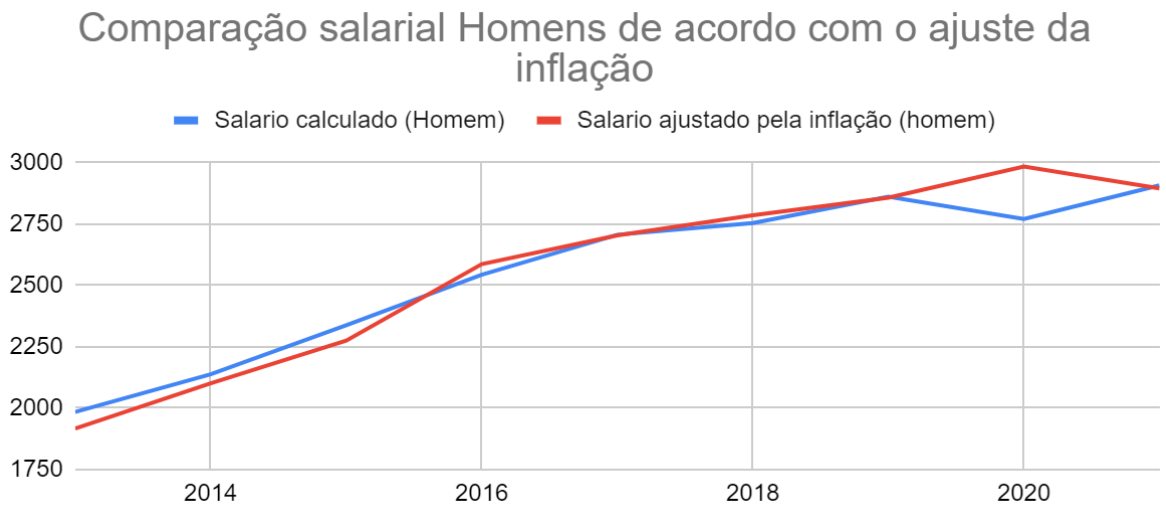
Outro ponto importante abordado pelo projeto é a análise do comportamento salarial em relação ao ajuste da inflação (IPCA), Tabela 4. Observou-se que o salário tende a acompanhar a inflação, mas no ano de 2020, devido à pandemia do coronavírus, a inflação foi relativamente maior do que o salário calculado. Esse comportamento é evidenciado no gráfico da Figura 44 para o salário masculino, sendo que a mesma tendência é observada para o salário feminino. (Tabela[...], 2023)

Tabela 4 - Comparação entre salário calculado e ajuste salarial pelo Índice Nacional de Preços ao Consumidor Amplo (IPCA).

Comparação salarial dos anos em relação ao IPCA				
Ano de referência	Salário calculado (Homem)	Salário calculado (Mulher)	Salário ajustado pela inflação (homem)	Salário ajustado pela inflação (Mulheres)
2021	2908,72	2588,81	2896,03	2578,52
2020	2770,79	2467,01	2985,46	2640,98
2019	2862,1	2531,86	2858,03	2510,19
2018	2754,73	2419,46	2786,75	2419,45
2017	2706,9	2350,12	2704,23	2328,7
2016	2544,2	2190,89	2586,93	2202,26
2015	2337,52	1989,93	2274,82	1905,35
2014	2137,79	1790,57	2100,84	1752,24
2013	1983,61	1654,46	1915,66	1606,34

Fonte: Autoria própria.

Figura 44 - Análise de comparação entre salário calculado e inflação ao decorrer dos anos.



Fonte: Autoria própria.

Ao analisar o gráfico da Figura 45, observou-se um aumento do desemprego após 2014, possivelmente devido à crise econômica brasileira que ocorreu neste ano.

Figura 45 - Análise de comparação entre salário calculado e inflação ao decorrer dos anos.

Fonte: Autoria própria.

Esses resultados destacam a importância da engenharia de dados e a facilidade que ela proporciona na análise de grandes volumes de dados usando programação. Essa abordagem permite explorar e compreender informações relevantes sobre o mercado de trabalho, incluindo a disparidade salarial de gênero e outras análises comparativas ao longo do tempo.

REFERÊNCIAS

ALMEIDA, R. Q. Automatização de tarefas com crontab e cron. [S.l]: Dicas l, 2018. Disponível em: https://www.dicas-l.com.br/arquivo/automatizacao_de_tarefas_com_crontab_e_cron.php. Acesso em: 12 mar. 2023.

AMAZON WEB SERVICES. O banco de dados de documentos definidos.[S.l]: Amazon, [201–?]. Disponível em: https://aws.amazon.com/pt/nosql/document/. Acesso em: 20 mar. 2023.

CEWEB. Dados Abertos: Capítulo 2 - O que são Dados Abertos?. [S.l]:Ceweb, [202–?], Disponível em: https://ceweb.br/guias/dados-abertos/capitulo-2/

COMISSÃO de Finanças aprova projeto que cria Lei de Dados Abertos. Distrito Federal: Agência câmara de notícias, 2022. Disponível em: https://www.camara.leg.br/noticias/924163-COMISSAO-DE-FINANCAS-APROVA-PROJETO-QUE-CRIA-LEI-DE-DADOS-ABERTOS. Acesso em: 14 mar. 2023.

FERREIRA J.P. :Bancos relacionais x Bancos não relacionais: quando usar cada um?. [S.l]: Natahouse, [ca 2020]. Disponível em: https://www.natahouse.com/blog/bancos-relacionais-x-bancos-nao-relacionais-quando-usar-cada-um. Acesso em: 20 mar. 2023.

FOREIGN Keys in SQL Server. [S.l]: Tutorials teacher, [201--?] . Disponível em: https://www.tutorialsteacher.com/sqlserver/create-foreign-keys. Acesso em: 27 mar. 2023.

GIMINO A. Spark vs. Hadoop MapReduce: Qual estrutura de big data escolher. [S.l]: Medium , 2019. Disponível em: https://medium.com/mangue-data/spark-vs-hadoop-mapreduce-qual-estrutura-de-big-data-escolher-b8927de07f7e. Acesso em: 17 mar. 2023.

GOOGLE Cloud. O que é um banco de dados relacional. [S.l]: Google cloud, 2023. Disponível em: https://cloud.google.com/learn/what-is-a-relational-database?hl=pt-br#:~:text=Um%20banco%20de%20dados%20relacional%20%C3%A9%20um%20conjunto%20de%20informa%C3%A7%C3%B5es,estruturas%20de%20dados%20se%20relacionam. Acesso em: 25 mar. 2023.

INFORMATICA. What Is ETL (Extract, Transform, Load). [S.l]: Informatica, [2022?]. Disponível em: https://www.informatica.com/nl/resources/articles/what-is-etl.html Acesso em: 12 mar. 2023.

KNOW Solution. Conheça os principais componentes de Business Intelligence.Rio de Janeiro: Know soluction, [201–?] Disponível em: https://www.knowsolution.com.br/conheca-os-principais-componentes-de-business-intelligence/. Acesso em: 27 mar. 2021.

MAIA F. : O que é Power BI: para que serve e como utilizar?. São Paulo: Lean solutions, 2021. Disponível em: https://www.leansolutions.com.br/blog/power-bi/#. Acesso em: 27 mar. 2023.

NASCIMENTO C. O que é a Linguagem DAX no Power BI?. [S.l]: Ninja do excel, 2021. Disponível em: https://ninjadoexcel.com.br/o-que-linguagem-dax-no-power-bi/. Acesso em: 4 abr. 2023.

ORACLE. What Is Big Data? [S.l]: Oracle, [201--?]. Disponível em: https://www.oracle.com/big-data/what-is-big-data/ Acesso em: 14 mar. 2023

SCHAFER L. What is NoSQL. [S.l]: Mongodb, [201--?]. Disponível em: https://www.mongodb.com/nosql-explained. Acesso em: 20 mar. 2023.

SHUBHNOOR G.: Role of Python for Data Engineering: 4 Critical Aspects. San Francisco: Hevo data, 2021. Disponível em: https://hevodata.com/learn/python-for-data-engineering/. Acesso em: 12 mar. 2023.

TABELA ipca: Índice Nacional de Preços ao Consumidor – Amplo. [S.l] : Idinheiro, 2023. Disponível em: https://www.idinheiro.com.br/tabelas/tabela-ipca/. Acesso em: 19 jun. 2023.

TIPOS de bancos de dados NoSQL. [S.l]: Micreiros, 2017. Disponível em: https://micreiros.com/tipos-de-bancos-de-dados-nosql/. Acesso em: 24 mar. 2023.

TORRES V. Entenda o que é RAIS e RAIS Negativa em 2023: a Relação Anual de Informações Sociais. [S.l]: Contabilizei, 2023. Disponível em: https://www.contabilizei.com.br/contabilidade-online/rais/. Acesso em: 20 abr. 2023