pr

Projeto extensionista

Tópicos de Big Data em Python

Matheus da Rocha Costa

Sumário

[Diagnóstico e Teorização: 2](#_Toc182501517)

[Identificação das Partes Interessadas e parceiros. 2](#_Toc182501518)

[Problemática e/ou problemas identificados. 2](#_Toc182501519)

[Dados para análise: 2](#_Toc182501520)

[Objetivos/resultados/efeitos a serem alcançados 2](#_Toc182501521)

[Ferramentas Utilizadas: 2](#_Toc182501522)

[Referencial teórico: 2](#_Toc182501523)

[Como Python foi utilizado para tratar o problema real: 2](#_Toc182501524)

[Resultados alcançados: 2](#_Toc182501525)

[Script 2](#_Toc182501526)

[Anomalias Detectadas 2](#_Toc182501527)

[Modelos de Notebooks e suas Quantidades (antes do tratamento) 2](#_Toc182501528)

[Modelos com Duplicatas 2](#_Toc182501529)

[Modelos de Notebooks e suas Quantidades (após o tratamento) 2](#_Toc182501530)

[Resultados alcançados (Gráfico em Barras) 2](#_Toc182501531)

[Script 2](#_Toc182501532)

[Resultados da Análise 2](#_Toc182501533)

[Gráfico de Resultados 2](#_Toc182501534)

[Resultados alcançados (Gráfico em pizza) 2](#_Toc182501535)

[Script 2](#_Toc182501536)

[Resultados da Análise 2](#_Toc182501537)

[Gráfico de Comparação 2](#_Toc182501538)

[Carta de apresentação 2](#_Toc182501539)

# Diagnóstico e Teorização:

## Identificação das Partes Interessadas e parceiros.

1. **Compreendendo as Partes Interessadas da Baru Offshore**

A Baru Offshore Navegação LTDA, como empresa atuante no setor de serviços de apoio marítimo offshore, possui um leque diversificado de partes interessadas. Cada uma delas possui um papel crucial no sucesso e na sustentabilidade das operações da empresa.

1. **Perfil Socioeconômico das Partes Interessadas:**

* **Colaboradores:**
* **Perfil:** Profissionais altamente qualificados, com experiência em operações marítimas, engenharia, manutenção e áreas afins.
* **Escolaridade:** Predominantemente ensino técnico e superior, com destaque para cursos na área naval e offshore.
* **Gênero:** Embora o setor historicamente tenha sido predominantemente masculino, a Baru Offshore, como empresa moderna, busca promover a diversidade e a equidade de gênero.
* **Faixa Etária:** Variada, abrangendo desde jovens profissionais recém-formados até profissionais com longa experiência no setor.
* **Quantidade Estimada:** Entre 400 a 500 colaboradores.
* Clientes:
* **Perfil:** Grandes empresas do setor de petróleo e gás, tanto nacionais quanto internacionais. Predominantemente a Petrobras.
* **Escolaridade:** Alta escolaridade, com profissionais altamente especializados em engenharia, geologia e áreas relacionadas.
* **Gênero:** Predominantemente masculino, mas com crescente participação de mulheres em cargos de liderança.
* **Faixa Etária:** Profissionais com experiência no setor, geralmente com idade acima de 30 anos.
* **Quantidade Estimada:** Depende do tamanho da empresa e do número de contratos em vigor.
* Fornecedores:
* **Perfil:** Profissionais altamente qualificados em áreas como engenharia, tecnologia da informação, gestão de projetos e análise de dados.
* **Escolaridade:** Predominantemente ensino superior, com pós-graduação e especializações em áreas relacionadas à indústria e tecnologia.
* **Gênero:** A empresa busca promover a diversidade de gênero, mas a área de tecnologia ainda pode apresentar um predomínio masculino.
* **Faixa Etária:** Variada, com uma concentração maior de profissionais com experiência e conhecimento técnico.
* **Quantidade Estimada:** Entre 51 a 100 colaboradores.

## Problemática e/ou problemas identificados.

A empresa Baru Offshore estava enfrentando problemas com a consistência e padronização dos dados de seu inventário de TI. Havia diversas anomalias e inconsistências nos registros dos ativos alugados e comprados, tais como:

* + **Anomalias de Cadastro**: Os números de série dos equipamentos apresentavam padrões inconsistentes, não condizentes com as especificações dos fabricantes (como Dell, HP e Lenovo), o que dificultava a identificação precisa dos ativos.
  + **Duplicação de Registros**: Alguns ativos tinham números de série duplicados, o que indicava possíveis registros redundantes ou repetidos. Isso comprometia a confiabilidade do inventário e aumentava a possibilidade de erros na gestão dos ativos.
  + **Equipamentos Incorretamente Classificados**: Equipamentos como monitores estavam listados na categoria incorreta, como "notebooks". Esse problema dificultava a categorização e alocação adequada dos ativos, impactando a gestão e a alocação de equipamentos conforme as necessidades dos usuários.
  + **Falta de Padronização nos Modelos**: Existia uma variedade muito grande de modelos e configurações de notebooks e outros equipamentos, o que não permitia uma padronização eficaz e dificultava a análise de custo-benefício, manutenção e substituição.

Esses problemas resultavam em desafios operacionais e financeiros, uma vez que dificultavam o controle preciso do inventário, aumentavam os custos de manutenção e dificultavam a tomada de decisões estratégicas para o parque tecnológico.

## Dados para análise:

* Anomalias de Cadastro: Verificar se os números de série dos equipamentos seguem os padrões de cada fabricante (Dell, HP, Lenovo), detectando irregularidades.
* Duplicação de Equipamentos: Identificar equipamentos com números de série repetidos que possam indicar registros duplicados.
* Distribuição de Modelos: Analisar a quantidade de modelos antes e depois do tratamento dos dados para verificar a padronização.
* Identificação de Equipamentos Incorretamente Classificados: Confirmar se os monitores estão corretamente listados e se há erros na categoria "notebook".

## Objetivos/resultados/efeitos a serem alcançados

* Recomendação de Padronização: Sugerir o modelo e configuração de equipamentos mais eficientes para cada perfil de usuário, alinhados com as necessidades específicas de cada área da empresa.
* Redução de Custos: Apontar oportunidades de economia por meio de padronização e eliminação de duplicatas, além de sugerir negociações de contratos e ajustes de inventário.
* Melhoria de Performance Operacional: Assegurar que os ativos estejam ajustados às demandas de uso, reduzindo a necessidade de suporte técnico e maximizando a produtividade.
* **Metodologia Proposta:**

1. Coleta e Integração de Dados:

* Fontes de Dados: Serão coletados dados de inventário, relatórios de anomalias e registros de manutenção.
* Unificação e Tratamento com Python: Os dados serão carregados e manipulados com o auxílio da biblioteca Pandas para leitura e análise de bases CSV.

1. **Tratamento e Limpeza dos Dados:**

* Padronização de Registros: Números de série serão verificados para garantir que estejam no formato correto, identificando e marcando anomalias quando necessário.
* Tratamento de Duplicatas: Equipamentos com números de série duplicados serão identificados e tratados para garantir a integridade dos registros.
* Correção de Classificações: Serão detectados e corrigidos registros incorretos, como monitores listados na categoria "notebook".

1. **Análise Exploratória e Visualização de Dados:**

* Gráficos de Barras e Pizza: Serão criados gráficos de barras para visualizar a distribuição de anomalias, duplicações, e categorização incorreta dos monitores, e gráficos de pizza para comparar a quantidade de modelos únicos antes e depois do tratamento.
* Agrupamento e Estatísticas Descritivas: A análise incluirá a contagem de equipamentos por categoria, marca e tipo de anomalia para identificar padrões de inconsistência e tendências.

## Ferramentas Utilizadas:

Durante o desenvolvimento do projeto para a empresa Baru Offshore, foi utilizado Python para lidar com grandes volumes de dados e resolver um problema relacionado ao tratamento de dados em uma base de inventário de equipamentos de TI. O objetivo era corrigir falhas na base de dados, como a falta de tratamento de anomalias, inconsistências nos registros e dados incompletos.

Para isso, utilizei algumas bibliotecas do Python, como:

1. **Pandas**: Foi a principal ferramenta para manipulação e análise dos dados. Com o Pandas, consegui carregar as bases de dados, processá-las, aplicar filtros e transformações, e realizar análises detalhadas, como a contagem de modelos únicos de notebooks e a detecção de duplicatas. Utilizei comandos como read\_csv para carregar os arquivos, apply para aplicar funções de detecção de anomalias em cada linha e value\_counts para contar a quantidade de modelos por descrição.
2. **Matplotlib** e **Seaborn**: Essas bibliotecas foram essenciais para a visualização dos dados. Criei gráficos de barras e gráficos de pizza para comparar a quantidade de modelos antes e depois do tratamento. O matplotlib foi utilizado para gráficos mais simples, como o gráfico de barras, e o seaborn foi utilizado para criar visualizações mais sofisticadas e customizadas, com uma estética mais refinada.
3. **Tkinter**: Para proporcionar uma melhor experiência ao usuário, usei o Tkinter para criar uma interface gráfica simples que permite ao usuário escolher o diretório onde deseja salvar o arquivo CSV após a atualização dos dados. Essa abordagem facilitou o processo de interação com a ferramenta, permitindo que a pessoa que fosse utilizar o sistema escolhesse o local para salvar o arquivo de maneira intuitiva.

## Referencial teórico:

Frameworks vistos, como o **IT Asset Management (ITAM)**, o **Data Management Body of Knowledge (DAMA-DMBOK)**, e os princípios de governança de dados de empresas como **SAS** e **PwC**.

Ambos utilizados para base sobre como melhorar o tratamento de dados, monitoramento do ciclo de vida do ativo, gerenciamento de dados e governança de dados.

# Como Python foi utilizado para tratar o problema real:

A base de dados da empresa estava repleta de inconsistências, como números de série inválidos e a inclusão de monitores listados como notebooks. Utilizando o Python, apliquei uma série de tratamentos automatizados para detectar essas anomalias. A função detectar\_anomalias foi criada para identificar problemas específicos, como números de série inválidos para marcas como Dell, HP e Lenovo. Além disso, o Python permitiu que eu verificasse duplicatas e inconsistências na base de dados, como a presença de monitores errados no lugar de notebooks.

Depois de identificar as anomalias, forneci uma solução interativa, permitindo que o responsável pela atualização dos dados corrigisse os registros manualmente. O Python, junto com o uso do input, tornou isso possível de

maneira simples e direta. Após todas as correções, os dados foram exportados de volta para um novo arquivo CSV.

Com o uso dessas bibliotecas e técnicas, consegui automatizar o processo de detecção e correção de problemas, além de fornecer uma maneira eficiente de analisar os dados antes e depois do tratamento. Esse processo resultou em uma base de dados mais confiável, melhorando a qualidade e a integridade das informações, o que é essencial para a tomada de decisões na empresa.

Com base nos resultados obtidos, serão propostas ações concretas para otimizar a gestão de ativos e reduzir custos.

# Resultados alcançados:

## Script

**import pandas as pd**

**import os**

**import tkinter as tk**

**from tkinter import filedialog**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**# Configuração do pandas para exibir todas as linhas no terminal**

**pd.set\_option('display.max\_rows', None)  # Remove qualquer limitação de exibição de linhas**

**# Obter o diretório atual onde o script está localizado**

**diretorio\_atual = os.path.dirname(os.path.abspath(\_\_file\_\_))**

**# Carregar os dados do arquivo CSV (usando caminho relativo)**

**caminho\_base\_inventario = os.path.join(diretorio\_atual, "BaseInventario.csv")**

**df = pd.read\_csv(caminho\_base\_inventario)**

**# Converter todas as descrições para minúsculo (mantendo a comparação case insensitive)**

**df['Descricao'] = df['Descricao'].str.lower()**

**# Filtrar para incluir apenas Notebooks**

**df\_notebooks = df[df['Tipo'].str.lower() == "notebook"]**

**# Função para verificar anomalias**

**def detectar\_anomalias(row):**

**descricao = row['Descricao'].lower()**

**num\_serie = str(row['NumSerie'])**

**if "dell" in descricao and not (len(num\_serie) == 7 and num\_serie.isalnum()):**

**return "Anomalia: DELL com NumSerie inválido"**

**if "hp" in descricao and not (num\_serie.startswith("BR") and len(num\_serie) == 10 and num\_serie[2:].isalnum()):**

**return "Anomalia: HP com NumSerie inválido"**

**if "lenovo" in descricao and not (num\_serie.startswith("PE") and len(num\_serie) == 8 and num\_serie[2:].isalnum()):**

**return "Anomalia: Lenovo com NumSerie inválido"**

**if "monitor" in descricao:**

**return "Anomalia: Monitor listado como Notebook"**

**return None**

**# Aplica a função de detecção de anomalias a cada linha**

**df\_notebooks['Anomalia'] = df\_notebooks.apply(detectar\_anomalias, axis=1)**

**# Filtra as linhas que possuem anomalias**

**anomalias = df\_notebooks[df\_notebooks['Anomalia'].notnull()]**

**# Exibe as anomalias encontradas**

**if not anomalias.empty:**

**print("\nAnomalias detectadas:")**

**print(anomalias[['Descricao', 'NumSerie', 'Anomalia']])**

**else:**

**print("\nNenhuma anomalia detectada.")**

**# Conta a quantidade de notebooks por descrição (antes do tratamento)**

**model\_counts\_before = df\_notebooks['Descricao'].value\_counts().reset\_index()**

**model\_counts\_before.columns = ['Descricao', 'Quantidade']**

**model\_counts\_before = model\_counts\_before.sort\_values(by='Quantidade', ascending=False)**

**# Exibe todos os modelos e quantidades antes do tratamento**

**print("\nModelos de notebooks e suas quantidades (antes do tratamento):")**

**print(model\_counts\_before)**

**# Verificar duplicatas (caso haja inconsistências em NumSerie)**

**duplicates = df\_notebooks[df\_notebooks.duplicated(subset='NumSerie', keep=False)]**

**if not duplicates.empty:**

**print("\nModelos com duplicatas:")**

**print(duplicates[['Descricao', 'NumSerie']].drop\_duplicates())**

**# while para permitir múltiplas correções manuais**

**while True:**

**palavra\_chave = input("\nDigite a palavra-chave do modelo que deseja corrigir (ex: 'g7', 'xps', 'g9') ou 'sair' para finalizar: ").strip().lower()**

**if palavra\_chave == 'sair':**

**print("Finalizando a atualização dos dados.")**

**break**

**# Encontra os modelos que contêm a palavra-chave**

**modelos\_para\_corrigir = df\_notebooks[df\_notebooks['Descricao'].str.contains(palavra\_chave)]**

**if modelos\_para\_corrigir.empty:**

**print(f"Nenhum modelo encontrado com a palavra-chave '{palavra\_chave}'.")**

**continue**

**print(f"\nModelos encontrados para a palavra-chave '{palavra\_chave}':")**

**print(modelos\_para\_corrigir[['Descricao', 'NumSerie']])**

**# Pergunta o novo valor da descrição**

**novo\_valor = input(f"Digite o novo valor para os modelos encontrados: ").strip()**

**# Atualiza a base de dados corrigida com o novo valor de descrição**

**df\_notebooks.loc[df\_notebooks['Descricao'].str.contains(palavra\_chave), 'Descricao'] = novo\_valor**

**# Pergunta se deseja continuar**

**continuar = input("\nDeseja corrigir outro modelo? (sim/não): ").strip().lower()**

**if continuar != 'sim':**

**print("Finalizando a atualização dos dados.")**

**break**

**# Contar a quantidade de notebooks por descrição após o tratamento**

**model\_counts\_after = df\_notebooks['Descricao'].value\_counts().reset\_index()**

**model\_counts\_after.columns = ['Descricao', 'Quantidade']**

**model\_counts\_after = model\_counts\_after.sort\_values(by='Quantidade', ascending=False)**

**# Exibe todos os modelos e quantidades após o tratamento**

**print("\nModelos de notebooks e suas quantidades (após o tratamento):")**

**print(model\_counts\_after)**

**# Função para selecionar diretório de destino**

**def selecionar\_diretorio():**

**root = tk.Tk()**

**root.withdraw()  # Ocultar a janela principal**

**caminho = filedialog.askdirectory()  # Abrir o explorador de arquivos**

**return caminho**

**# Solicitar ao usuário um diretório para salvar o arquivo**

**diretorio = selecionar\_diretorio()**

**if diretorio:**

**caminho\_arquivo = os.path.join(diretorio, "BaseInventario\_atualizado.csv")**

**df\_notebooks.to\_csv(caminho\_arquivo, index=False)**

**print(f"\nBase de dados atualizada com sucesso! O arquivo foi salvo em '{caminho\_arquivo}'.")**

**else:**

**print("Nenhum diretório selecionado. O arquivo não foi salvo.")**

**"""**

**\*\*tratamento.py\*\***

**Este script Python realiza o processamento de um dataset de inventário de notebooks, com o objetivo de identificar e corrigir inconsistências nos dados.**

**\*\*Funcionalidades:\*\***

**\* \*\*Carregamento:\*\* Lê um arquivo CSV com informações sobre notebooks.**

**\* \*\*Pré-processamento:\*\***

**\* Converte descrições para minúsculas.**

**\* Filtra apenas registros de notebooks.**

**\* \*\*Detecção de anomalias:\*\***

**\* Verifica inconsistências em números de série e classificações.**

**\* Identifica notebooks classificados incorretamente (e.g., monitores como notebooks).**

**\* \*\*Correção manual:\*\***

**\* Permite ao usuário corrigir manualmente descrições de modelos.**

**\* \*\*Geração de relatório:\*\***

**\* Gera um novo arquivo CSV com os dados corrigidos.**

**\*\*Uso:\*\***

**1. \*\*Configuração:\*\***

**\* Certifique-se de que o arquivo `BaseInventario.csv` esteja na mesma pasta que o script.**

**\* Configure os critérios de detecção de anomalias para cada marca de notebook.**

**2. \*\*Execução:\*\***

**\* Execute o script.**

**\* Siga as instruções para corrigir as anomalias e selecionar o diretório de saída.**

**\*\*Dependências:\*\***

**\* pandas: Para manipulação de dados.**

**\* os: Para interagir com o sistema operacional.**

**\* tkinter: Para criar interfaces gráficas simples (seleção de diretório).**

**\* matplotlib, seaborn: Para visualização de dados (opcional).**

**\*\*Autor:\*\* [Seu nome]**

**\*\*Data:\*\* [Data de criação]**

**\*\*Versão:\*\* [Versão do script]**

**\*\*Histórico de alterações:\*\***

**\* [Data]: [Descrição da alteração]**

**\*\*Observações:\*\***

**\* A detecção de anomalias é baseada em regras específicas para cada marca de notebook.**

**\* A correção manual é realizada de forma interativa, permitindo ao usuário verificar e corrigir cada caso.**

**\* O script gera um relatório detalhado com as anomalias encontradas e as correções realizadas.**

**"""**

## Anomalias Detectadas

Anomalias detectadas:

Descricao NumSerie Anomalia  
notebook dell inspiron 14 - 5458 1LCY9B92 Anomalia: DELL com NumSerie inválido  
 notebook hp g6 240 BRJ848ZWPH Anomalia: HP com NumSerie inválido  
 notebook dell latitude 3490 NBT8 Anomalia: DELL com NumSerie inválido  
 notebook dell latitude 3490 NBT7 Anomalia: DELL com NumSerie inválido  
 notebook lenovo v15 g2 SPE08DN1L Anomalia: Lenovo com NumSerie inválido  
 notebook lenovo thinkpad(ap) SPE06RY9J Anomalia: Lenovo com NumSerie inválido  
 notebook hp g8 250 BRJ2514JB7 Anomalia: HP com NumSerie inválido  
 notebook dell latitude 3420 D7KKW3 Anomalia: DELL com NumSerie inválido  
 notebook dell latitude 5490 PTL9 Anomalia: DELL com NumSerie inválido  
 notebook dell vostro 3510 2QL31LQ3 Anomalia: DELL com NumSerie inválido  
 monitor dell p2222h BR-0VW2GP-TVB00-271-4JRB-A01 Anomalia: DELL com NumSerie inválido  
 notebook dell latitude 5420 FP69V9V3 Anomalia: DELL com NumSerie inválido  
 notebook dell latitude 5420 BRP69VQ3 Anomalia: DELL com NumSerie inválido

## Modelos de Notebooks e suas Quantidades (antes do tratamento)

Descricao Quantidade  
 notebook dell latitude 3420 55  
 notebook dell latitude 5420 29  
 notebook hp 14  
 notebook dell latitude 5490 12  
 notebook hp g8 250 11  
 notebook hp 250 g8 11  
 notebook hp g8 250 16gb 10  
 notebook hp g7 240 8  
 notebook lenovo thinkpad t14 7  
 notebook dell vostro 15 3510 6  
 notebook dell latitude 3410 5  
 notebook dell latitude 3490 5  
 notebook hp g9 250 4  
 notebook hp 250 g9 3  
 notebook hp g8 240 3  
 notebook hp com carregador 3  
 notebook dell inspiron 14 - 5458 3  
 notebook hp 240 g7 3  
 notebook hp g6 240 3  
 notebook hp 3  
 notebook hp com fonte 2  
 notebook dell inspiron 14 - 7460 2  
 notebook dell inspiron 7472 com carregador 2  
 dell latitude 5490 2  
 notebook hp g8 2  
 dell inspiron 14 - 5458 2  
 notebook lenovo b330 2  
 notebook hp g7 2  
 notebook dell inspiron 7472 2  
 notebook hp 240 g6 (com carregador) 1  
 notebook dell inspiron 15 / 3567 1  
 notebook dell inspiron 14 5458 1  
dell latitude 7480 com carregador + kit teclado e mouse 1  
 notebook dell inspiron 5480 1  
 notebook dell inspiron 5458 1  
 notebook dell vostro 5470 1  
 dell inspiron 14 5458 1  
 notebook hp 240 g6 (manatech) 1  
 notebook dell vostro 5481 1  
 dell inspiron 7472 1  
 notebook hp 240 g6 1  
 notebook dell p92g + teclado e mouse 1  
 notebook dell latitude 5400 + teclado e mouse 1  
 notebook dell inspiron 5480 com carregador 1  
 notebook dell latitude 7480 1  
 notebook hp 240 g7 com carregador 1  
 notebook dell vostro 5470 1  
 notebook dell latitude 9480 1  
 notebook dell xps 13 - 9320 1  
 notebook dell latitude 5490 (selbetti) 1  
 notebook dell latitude 5480 1  
 notebook g8 250 1  
 notebook dell xps 13 1  
 notebook dell xps 13 plus 9320 1  
 notebook dell latitude 3490(selbetti) 1  
 notebook dell inspiron 3501 1  
 notebook 240 g7 (apnetworks) 1  
 notebook lenovo v15 g2 1  
 notebook hp g7 240 (apnetworks) 1  
 notebook dell vostro 14-5480 1  
 notebook lenovo thinkpad(ap) 1  
 notebook dell 3420 1  
 notebook dell vostro 3510 1  
 monitor dell p2222h 1  
 notebook dell inspiron 5420 1  
 lenovo thinkpad e14 gen2 1  
 notebook dell latitude 5420 1  
 notebook lenovo thinkpad t14 1

## Modelos com Duplicatas

Descricao NumSerie  
notebook dell latitude 9480 CJ9PDQ2  
notebook dell latitude 7480 CJ9PDQ2

## Modelos de Notebooks e suas Quantidades (após o tratamento)

Descricao Quantidade  
 notebook dell latitude 3420 55  
 notebook dell latitude 5420 29  
 HP 240 G7 16  
 notebook hp 14  
 notebook dell latitude 5490 12  
 notebook hp g8 250 11  
 notebook hp 250 g8 11  
 notebook hp g8 250 16gb 10  
 notebook lenovo thinkpad t14 7  
 notebook dell vostro 15 3510 6  
 notebook dell latitude 3490 5  
 notebook dell latitude 3410 5  
 notebook hp g9 250 4  
 notebook hp g8 240 3  
 notebook hp g6 240 3  
 notebook hp 3  
 notebook hp com carregador 3  
 notebook hp 250 g9 3  
 notebook dell inspiron 14 - 5458 3  
 dell latitude 5490 2  
 notebook lenovo b330 2  
 notebook dell inspiron 7472 2  
 notebook dell inspiron 14 - 7460 2  
 notebook hp g8 2  
 notebook dell inspiron 7472 com carregador 2  
 notebook hp com fonte 2  
 dell inspiron 14 - 5458 2  
 notebook dell vostro 5470 1  
 notebook dell inspiron 14 5458 1  
 notebook dell inspiron 5480 1  
 notebook dell latitude 9480 1  
dell latitude 7480 com carregador + kit teclado e mouse 1  
 notebook hp 240 g6 (manatech) 1  
 notebook hp 240 g6 1  
 dell inspiron 7472 1  
 notebook dell vostro 5481 1  
 notebook dell inspiron 5480 com carregador 1  
 notebook dell latitude 7480 1  
 notebook dell p92g + teclado e mouse 1  
 notebook dell latitude 5400 + teclado e mouse 1  
 notebook dell inspiron 15 / 3567 1  
 notebook dell inspiron 5458 1  
 dell inspiron 14 5458 1  
 notebook hp 240 g6 (com carregador) 1  
 notebook dell vostro 5470 1  
 notebook dell latitude 5480 1  
 notebook dell xps 13 - 9320 1  
 notebook dell xps 13 1  
 notebook dell xps 13 plus 9320 1  
 notebook dell latitude 3490(selbetti) 1  
 notebook dell inspiron 3501 1  
 notebook lenovo v15 g2 1  
 notebook dell latitude 5490 (selbetti) 1  
 notebook g8 250 1  
 notebook dell vostro 14-5480 1  
 notebook lenovo thinkpad(ap) 1  
 notebook dell 3420 1  
 notebook dell vostro 3510 1  
 monitor dell p2222h 1  
 notebook dell inspiron 5420 1  
 lenovo thinkpad e14 gen2 1  
 notebook dell latitude 5420 1  
 notebook lenovo thinkpad t14 1

# Resultados alcançados (Gráfico em Barras):

## Script

import pandas as pd

import matplotlib.pyplot as plt

import os

# Obter o diretório atual onde o script está localizado

diretorio\_atual = os.path.dirname(os.path.abspath(\_\_file\_\_))

# Carregar os dados da base CSV (usando caminho relativo)

caminho\_base\_inventario\_pronta = os.path.join(diretorio\_atual, "BaseInventario\_pronta.csv")

df\_atualizado = pd.read\_csv(caminho\_base\_inventario\_pronta)

# Função para verificar anomalias

def detectar\_anomalias(row):

    descricao = row['Descricao'].lower()

    num\_serie = str(row['NumSerie'])

    if "dell" in descricao and not (len(num\_serie) == 7 and num\_serie.isalnum()):

        return "Anomalia: DELL com NumSerie inválido"

    if "hp" in descricao and not (num\_serie.startswith("BR") and len(num\_serie) == 10 and num\_serie[2:].isalnum()):

        return "Anomalia: HP com NumSerie inválido"

    if "lenovo" in descricao and not (num\_serie.startswith("PE") and len(num\_serie) == 8 and num\_serie[2:].isalnum()):

        return "Anomalia: Lenovo com NumSerie inválido"

    if "monitor" in descricao:

        return "Anomalia: Monitor listado como Notebook"

    return None

# Função de detecção de anomalias a cada linha

df\_atualizado['Anomalia'] = df\_atualizado.apply(detectar\_anomalias, axis=1)

# Conta as anomalias

anomalias = df\_atualizado['Anomalia'].notnull().sum()

# Filtra para incluir apenas Notebooks

df\_notebooks = df\_atualizado[df\_atualizado['Tipo'].str.lower() == "notebook"]

# Verifica duplicatas com base na chave primária

duplicatas = df\_notebooks[df\_notebooks.duplicated(subset='NumSerie', keep=False)].shape[0]

# Verifica monitores na base de dados errada

monitores\_errados = df\_atualizado[df\_atualizado['Tipo'].str.lower() == "monitor"].shape[0]

# Prepara os dados para o gráfico de barras

categorias = ['Anomalias', 'Notebooks com Duplicata de NumSerie', 'Monitores Errados']

quantidades = [anomalias, duplicatas, monitores\_errados]

# Exibir o gráfico de barras

plt.figure(figsize=(8, 6))

plt.bar(categorias, quantidades, color=['#66c2a5', '#fc8d62', '#8c6bb1'])

plt.title('Quantidade de Anomalias, Duplicatas e Monitores Errados')

plt.ylabel('Quantidade')

plt.show()

## Resultados da Análise

Total de Anomalias Detectadas: 13

Notebooks com Duplicatas de Número de Série: 2

Monitores listados incorretamente como Notebooks: 1

## Gráfico de Resultados

O gráfico a seguir exibe a quantidade de anomalias, duplicatas e registros incorretos:

Gráfico, Gráfico de barras

Descrição gerada automaticamente

# Resultados alcançados (Gráfico em pizza):

Este documento apresenta a análise comparativa entre a quantidade de modelos únicos  
no inventário original e no inventário após o tratamento de dados.

## Script

import pandas as pd

import matplotlib.pyplot as plt

import os

# Obter o diretório atual onde o script está localizado

diretorio\_atual = os.path.dirname(os.path.abspath(\_\_file\_\_))

# Carregar os dados das bases CSV (usando caminho relativo)

caminho\_base\_inventario = os.path.join(diretorio\_atual, "BaseInventario.csv")

caminho\_base\_inventario\_pronta = os.path.join(diretorio\_atual, "BaseInventario\_pronta.csv")

df\_original = pd.read\_csv(caminho\_base\_inventario)

df\_atualizado = pd.read\_csv(caminho\_base\_inventario\_pronta)

# Contar a quantidade de modelos únicos na coluna "Descricao" em cada arquivo

modelos\_unicos\_original = df\_original['Descricao'].nunique()

modelos\_unicos\_atualizado = df\_atualizado['Descricao'].nunique()

# Prepara os dados para o gráfico de pizza

dados = [modelos\_unicos\_original, modelos\_unicos\_atualizado]

labels = ['Modelos Originais', 'Modelos Após Tratamento']

# Exibe o gráfico de pizza

plt.figure(figsize=(8, 6))

plt.pie(dados, labels=labels, startangle=140, colors=['#66c2a5', '#fc8d62'],

        wedgeprops={'edgecolor': 'black'}, autopct=lambda p: f'{int(p/100.\*sum(dados))}')  # Mostrar os valores reais

plt.title('Comparativo de Quantidade de Modelos Únicos Antes e Depois do Tratamento')

plt.show()

## Resultados da Análise

Quantidade de Modelos Únicos no Inventário Original: 196

Quantidade de Modelos Únicos no Inventário Após o Tratamento: 27

## Gráfico de Comparação

O gráfico a seguir compara a quantidade de modelos únicos antes e depois do tratamento de dados:

Gráfico, Gráfico de pizza

Descrição gerada automaticamente

# Carta de apresentação:

Vimos por desta apresentar o grupo de acadêmicos da Universidade Estácio de Sá listado na tabela a final deste documento, a fim de convidá-lo a participar de uma atividade extensionista associada à disciplina Tópicos de Big Data em Python, sob responsabilidade da/do Prof. Luis Alexandre Estevão da Silva.

Em consonância ao Plano Nacional de Educação e demais normativas educacionais vigentes, a Universidade Estácio de Sá desenvolve atividade extensionista que, norteados pela metodologia de aprendizagem baseada em projetos, tem por princípios fundantes o diagnóstico dos problemas/demandas/necessidades, a participação ativa dos interessados/públicos participantes, a construção dialógica, coletiva e experiencial de conhecimentos, o planejamento de ações, o desenvolvimento e avaliação das ações, a sistematização dos conhecimentos, a avaliação das ações desenvolvidas.

Nesse contexto, a disciplina acima mencionada tem como principal escopo os temas relacionados à Análise de Dados em Python com Pandas, Hadoop e Armazenamento de Dados, Princípios de Big Data, Princípios de Desenvolvimento de Spark com Python e Big Data Analytics.

Sendo assim, pedimos o apoio dessa organização/entidade/coletivo/associação/outro, que aqui chamaremos de parte interessada, para a realização das seguintes atividades: diagnósticos, análises, entrevistas, levantamentos, projetos ou qualquer outra metodologia de estudo de caso que auxilie no desenvolvimento das competências de nossos acadêmicos e ao mesmo tempo possa contribuir para a comunidade em que estamos inseridos.

Como se trata de atividade de ensino/aprendizagem de caráter extensionista, prevista no Projeto Pedagógico do Curso, salientamos que:

• não há cobrança de remuneração de qualquer natureza por parte da Universidade Estácio de Sá, seus alunos ou o docente da disciplina, à parte interessada;

• as atividades desenvolvidas no âmbito do projeto extensionista não configuram relação de trabalho entre os alunos e o docente da Universidade Estácio de Sá, disciplina Tópicos de Big Data em Python, e a parte interessada;

• os resultados do projeto só poderão ser implantados para uso efetivo mediante Anotação de Responsabilidade Técnica de um profissional habilitado;

• os resultados do projeto podem ser implantados pela parte interessada para fins lucrativos, sem a necessidade de pagamento de quaisquer benefícios aos alunos, ao docente da disciplina e à Universidade Estácio de Sá;

• quaisquer custos relativos à implantação e operação contínua do projeto fora do escopo das atividades do presente projeto serão arcados pela parte interessada.

Aproveitamos a oportunidade e solicitamos que, em caso de aceite, seja formalizado, mediante assinatura da Carta de Autorização, as atividades e informações que o(s) aluno(s) poderá(ão) ter acesso.

Desde já nos colocamos à sua disposição para quaisquer esclarecimentos.

|  |
| --- |
| Grupo de Alunos |
| Matheus da Rocha Costa  Matrícula: 202309922177 |
| Rio de Janeiro, 14 de novembro de 2024. |
| Luis Alexandre Estevão da Silva.  Tópicos de Big Data em Python.  2024.2 |

Texto

Descrição gerada automaticamenteMarcos Gabriel dos Reis Mendes.

Coordenador Administrativo e de Tecnologia da Informação. Baru Offshore Navegação LTDA.

CNPJ: 14.426.327/0001-34