

Relatório Técnico – Engenharia de Dados

1. Objetivo

O notebook tem como finalidade tratar, padronizar e validar os dados de vendas de produtos de uma confeitaria, garantindo consistência entre valores de custo, faturamento e lucro, além de corrigir divergências nos nomes de produtos e categorias.

2. Carregamento e Inspeção dos Dados

2.1 Leitura da Base

```
df =  
pd.read_excel("/content/relatorio_items_vendas20251016-1-rwrl26.xlsx")
```

O dataset foi importado de um arquivo Excel contendo informações de vendas.

As colunas principais incluem:

- Produto;
- Categoria;
- Quantidade Vendida;
- Faturamento Total;
- Preço Médio;
- Custo Médio;
- Custo Total;
- Lucro;
- Margem;

2.2 Análise Inicial

Comandos como `df.head()` e `df.info()` foram utilizados para:

- Verificar os tipos de dados.
- Identificar possíveis inconsistências de tipo (e.g. colunas numéricas salvas como texto).

3. Correção de Tipos de Dados

As colunas "Custo Médio" e "Custo Total" foram convertidas para o tipo float:

```
df[['Custo Médio', 'Custo Total']] = df[['Custo Médio', 'Custo Total']].astype(float)
```

Essa etapa assegura que cálculos posteriores de lucro e margens sejam válidos numericamente.

4. Padronização dos Nomes de Produtos

4.1 Normalização Geral

Foi criada uma nova coluna `Produto_corrigido` com as seguintes transformações:

```
df['Produto_corrigido'] = df['Produto'].str.strip().str.upper()
```

- Remoção de espaços extras no início e no final.
- Conversão para maiúsculas para uniformizar a comparação entre nomes.

4.2 Correção de Erros Específicos

Foram aplicadas substituições pontuais para corrigir variações manuais de escrita:

```
df['Produto_corrigido'] = df['Produto_corrigido'].str.replace(
    'NAKED. RED VELVET - 10 FATIAS - ZERO',
    'NAKED - RED VELVET (10 FATIAS - ZERO AÇÚCAR)'
)
```

Outros exemplos de ajustes:

- "FATIA -CHOCOMELO (150G)" → "FATIA - CHOCOMELO (150G)"
- "FOCACCIA- PESTO & FRANGO" → "FOCACCIA - PESTO & FRANGO"
- "POTE DA FELICIDADE -MOUSE DE CHOCOLATE BRANCO E MORANGO" → "POTE DA FELICIDADE - MOUSE DE CHOCOLATE BRANCO E MORANGO"

- "BOLO COM RASPAS DE CHOCOLATE(8 FATIAS) ZERO AÇÚCAR" → "BOLO COM RASPAS DE CHOCOLATE (8 FATIAS) ZERO AÇÚCAR"

Essas correções eliminam duplicidades de produtos causadas por erros de digitação ou formatação.

5. Padronização das Categorias

5.1 Normalização Geral

Foi criada uma coluna `Categoria_corrigido` com o mesmo padrão de limpeza:

```
df['Categoria_corrigido'] = df['Categoria'].str.strip().str.upper()
```

5.2 Correções Específicas

Algumas categorias foram ajustadas manualmente:

```
df['Categoria_corrigido'] = df['Categoria_corrigido'].replace('CESTA E PRESENTES', 'CESTAS E PRESENTES')
df['Categoria_corrigido'] = df['Categoria_corrigido'].str.replace('LANCHES - SEM GLÚTEN E SEM LACTOSE !', 'LANCHES - SEM GLÚTEN E SEM LACTOSE')
df['Categoria_corrigido'] = df['Categoria_corrigido'].str.replace('TORTAS - AGENDAMENTO 24H - RETIRADA EM LOJA -', 'TORTAS - AGENDAMENTO 24H - RETIRADA EM LOJA')
```

6. Substituição das Colunas Originais

Após validação das colunas corrigidas:

```
df['Produto'] = df['Produto_corrigido']
df['Categoria'] = df['Categoria_corrigido']
df = df.drop(columns=["Produto_corrigido", "Categoria_corrigido"])
```

As colunas intermediárias são removidas, mantendo apenas as versões finais padronizadas.

7. Identificação de Produtos em Múltiplas Categorias

O código busca produtos cadastrados em mais de uma categoria:

```
duplicados = df.groupby("Produto")["Categoria"].nunique()
produtos_multicat = duplicados[duplicados > 1].index
df_multicat =
df[df["Produto"].isin(produtos_multicat)].sort_values(["Produto",
"Categoria"])
tabela_resumo = df_multicat[["Produto", "Categoria", "Preço Médio"]]
```

Essa análise identifica possíveis inconsistências na categorização de produtos.

8. Validação dos Cálculos Financeiros

8.1 Faturamento

Verificação se o faturamento total corresponde ao cálculo:

```
df['Faturamento_calculado'] = df['Quantidade Vendida'] * df['Preço
Médio']
```

8.2 Lucro

Verificação do lucro com base nos custos:

```
df['Lucro_calculado'] = df['Faturamento Total'] - (df['Custo Médio'] *
df['Quantidade Vendida'])
```

8.3 Limpeza Final

Após validação, as colunas auxiliares são removidas:

```
df = df.drop(columns=["Faturamento_calculado", "Lucro_calculado"])
```

9. Exportação dos Dados Tratados

O dataset final, devidamente limpo e validado, é exportado para dois formatos:

```
df.to_csv('relatorio_item_vendas_limpo.csv', index=False)
df.to_excel('relatorio_item_vendas_limpo.xlsx', index=False)
```

10. Resumo das Principais Transformações

| Tipo de Ação | Descrição |
|----------------------------------|--|
| Conversão de tipos | “Custo Médio” e “Custo Total” convertidos para float |
| Padronização textual | Remoção de espaços e aplicação de maiúsculas em produtos e categorias |
| Correções manuais | Uniformização de nomes de produtos e categorias com erros de digitação |
| Identificação de inconsistências | Verificação de produtos em múltiplas categorias |
| Validação de cálculos | Conferência de faturamento e lucro com base em quantidade e custos |
| Exportação final | Criação de versões limpas em CSV e Excel |

11. Conclusão

O processo de tratamento garantiu integridade e consistência nos dados de vendas, eliminando redundâncias e erros de digitação, além de assegurar coerência entre os cálculos de faturamento, custo e lucro.

O resultado final é uma base de dados pronta para análises financeiras e de desempenho de produtos da confeitaria.