Projeto Final – AgroData

Análise da Produção de Origem Animal e Produção de Aquicultura

Jéssica Bento

Leandro Nascimento

Thais Telles

Thamires Pires

Victor Santos

Vinicius Menezes

SUMÁRIO

1. PROBLEMA

Todos os processos são executados manualmente, o que implica que em cada etapa - extração, armazenamento, limpeza, transformação, carregamento e modelagem - requer a intervenção humana sempre que necessário. Esse procedimento é suscetível a demoras e erros, podendo prejudicar a tomada de decisões e a eficiência empresarial.

2. OBJETIVOS

Este estudo visa analisar e comparar a produção de dois setores agropecuários distintos: Origem Animal e Aquicultura. O objetivo é investigar a distribuição da produção por estado, identificando as áreas de maior concentração em cada setor. Além disso, pretende-se examinar a variação dessas produções ao longo do intervalo de anos selecionado.

3. SOBRE OS DADOS

A base de dados utilizada neste estudo foi coletada pelo Instituto Brasileiro de Geografia e Estatística (IBGE) e consiste nos dados de Pesquisa da Pecuária Municipal (PPM).

> *"A Pesquisa da Pecuária Municipal fornece informações sobre os efetivos da pecuária existentes no município na data de referência do levantamento, bem como a produção de origem animal, e o valor da produção durante o ano de referência."*

FONTE: *BASE DOS DADOS. Pesquisa Pecuária Municipal (PPM). Disponível em: <https://basedosdados.org/dataset/f7df4160-7a6f-4658-a287-3a73d412ed10?table=ff7d91a7-4482-4097-8b73-82d02f17a8c0>. Acesso em: 29 de mai. de 2024.*

A partir da base de dados em questão, optamos por utilizar duas de suas tabelas: "Produção da Aquicultura" e "Produção de Origem Animal". A seguir, apresentaremos uma descrição detalhada de cada uma dessas tabelas.

A produção de Aquicultura engloba as produções da piscicultura, carcinocultura e malacocultura. Seu conteúdo inclui:

Nome	Tipo No BigQuery	Descrição
ano	INT64	Ano
sigla_uf	STRING	Sigla da Unidade da Federação
id_municipio	STRING	ID Município IBGE 7 dígitos
produto	STRING	Produto
quantidade	INT64	Quantidade da produção
valor	INT64	Valor da produção

FONTE: *BASE DOS DADOS. Pesquisa Pecuária Municipal (PPM). Disponível em: <https://basedosdados.org/dataset/f7df4160-7a6f-4658-a287-3a73d412ed10?table=ff7d91a7-4482-4097-8b73-82d02f17a8c0>. Acesso em: 29 de mai. de 2024.*

A produção de Origem Animal contempla a produção de leite, ovos de galinha, ovos de codorna, mel, lã bruta e casulos do bicho-da-seda. Seu conteúdo inclui:

Nome	Tipo No BigQuery	Descrição
ano	INT64	Ano
sigla_uf	STRING	Sigla da Unidade da Federação
id_municipio	STRING	ID Município IBGE 7 dígitos
produto	STRING	Produto
unidade	STRING	Unidade de medida do produto
quantidade	INT64	Quantidade da produção
valor	INT64	Valor da produção

FONTE: *BASE DOS DADOS. Pesquisa Pecuária Municipal (PPM). Disponível em: <https://basedosdados.org/dataset/f7df4160-7a6f-4658-a287-3a73d412ed10?table=ff7d91a7-4482-4097-8b73-82d02f17a8c0>. Acesso em: 29 de mai. de 2024.*

Além disso, foi utilizado uma outra base de dados para fazer o cruzamento entre ambas as bases unindo-as por municípios. Essa base de dados, denominada 'Malha Municipal' também coletada pelo IBGE, consiste em:

> *"Os arquivos desta disseminação representam a Malha Municipal Digital da Divisão Político-Administrativa Brasileira, de acordo com a estrutura político-administrativa vigente. No ano de 2022, a Malha Municipal Digital (MMD) da Divisão Político-Administrativa Brasileira é constituída por 5572 geocódigos."*

FONTE: *IBGE. Malha Municipal. Disponível em: <https://www.ibge.gov.br/geociencias/todos-os-produtos-geociencias/15774-malhas.html?=&t=acesso-ao-produto>. Acesso em: 29 de mai. de 2024.*

A Malha Municipal engloba 5568 Municípios, 1 Distrito Federal (Brasília – DF), 1 Distrito Estadual (Fernando de Noronha – PE), 2 Áreas Estaduais Operacionais (Lagoa dos Patos e Lagoa Mirim, ambas atribuídas ao Rio Grande do Sul). Seu conteúdo inclui:

CD_MUN	Código município
NM_MUN	Nome município
SIGLA_UF	Sigla unidade federativa
AREA_KM2	Área quilometro quadrado
GEOMETRY	Coordenadas

4. METODOLOGIA

A metodologia de Descoberta de Conhecimento em Bancos de Dados (KDD - Knowledge Discovery in Databases) é um processo sistemático para extrair informações úteis e significativas a partir de grandes conjuntos de dados. Neste contexto, para aplicar a metodologia KDD à análise da Pesquisa da Pecuária Municipal (PPM) e a Malha Municipal, ambos do IBGE, usamos as seguintes etapas:

1. ***Seleção dos Dados*** - Inicialmente, foram selecionados os conjuntos de dados relevantes da PPM, focando nas informações relacionadas à produção de Origem Animal e Aquicultura, bem como dados geográficos da Malha Municipal para identificação dos municípios.

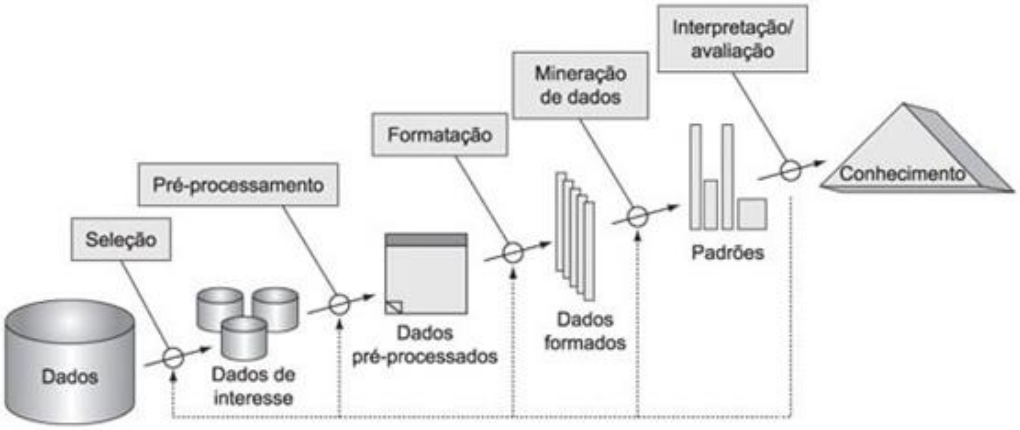
2. ***Pré-processamento dos Dados*** - Os dados foram submetidos a um processo de limpeza e pré-processamento para tratar valores ausentes e inconsistências. Isso garantiu a qualidade dos dados antes da análise.

3. ***Transformação dos Dados*** - Os dados foram transformados conforme necessário para torná-los adequados para análise. Foi realizado a inserção de novas colunas e a união das 3 bases de dados citadas anteriormente.

4. ***Mineração de Dados*** - Utilizando técnicas de mineração de dados, como análise estatística, foram explorados os padrões e inconsistências presentes nos dados. Foram investigadas também a distribuição da produção de Origem Animal e Aquicultura por estado.

5. ***Avaliação e Interpretação dos Resultados*** - Os resultados da análise foram avaliados quanto à sua relevância e significância. Foram interpretados de acordo com os objetivos do estudo e do contexto do setor, buscando extrair insights e conclusões úteis.

6. ***Apresentação dos Resultados*** - Por fim, os resultados foram apresentados de forma clara e objetiva, por meio de gráficos, tabelas e relatórios descritivos com o uso das ferramentas Power BI e Looker Studio. Foram destacadas as principais descobertas e insights obtidos durante o processo de análise dos dados, proporcionando uma compreensão mais profunda da produção agropecuária no Brasil.



FONTE: *MEDIUM. Knowledge Discovery in Databases (KDD). Disponível em: <https://medium.com/blog-do-zouza/knowledge-discovery-in-databases-kdd-462ea2775715>. Acesso em: 3 de jun. de 2024.*

5. FLUXO DE TRABALHO (WORKFLOW)

Nesta seção, apresentamos a origem dos dados utilizados neste estudo, detalhando o processo de extração, transformação e carregamento (ETL) dos mesmos. Os dados foram inicialmente acessados através do site Base Dos Dados e IBGE, que são repositórios confiáveis de dados públicos.

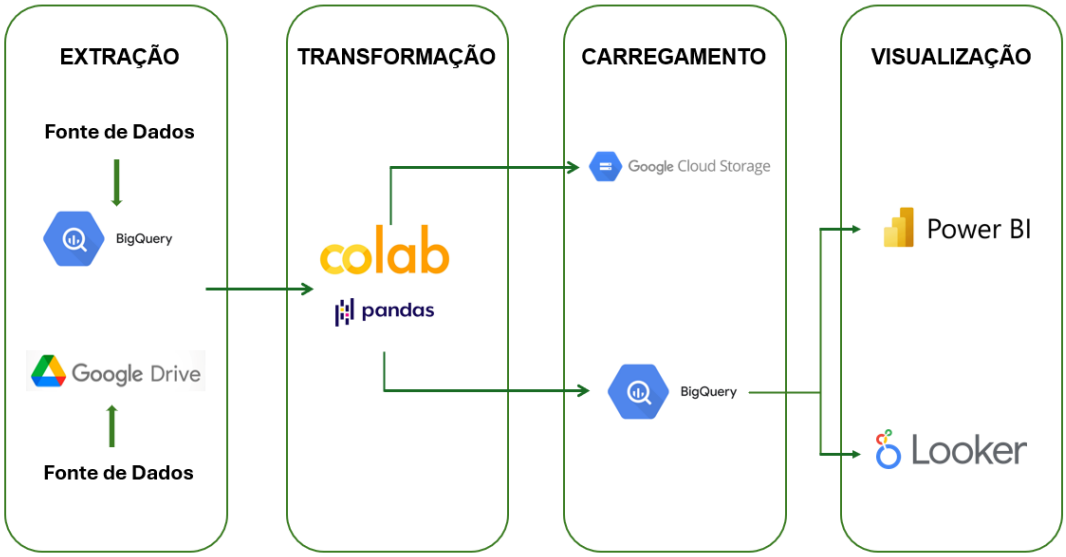
A extração dos dados foi realizada utilizando a BigQuery, um serviço de armazenamento e análise de dados do Google. O BigQuery permite acessar grandes volumes de dados de forma eficiente e escalável, tornando-o uma escolha adequada para lidar com conjuntos de dados complexos como a PPM. Além disso, os dados da Malha Municipal foi extraída do Google Drive, uma plataforma de armazenamento na nuvem amplamente utilizada.

Após a extração dos dados, foram aplicadas diversas etapas de transformação utilizando a biblioteca Python Pandas. Esta etapa incluía limpeza dos dados e formatação de variáveis com o objetivo de preparar os dados para análise.

Uma vez que os dados foram devidamente transformados, eles foram carregados para o Google Cloud Storage e para o BigQuery. O Cloud Storage é um serviço de armazenamento em nuvem escalável e altamente disponível, enquanto o BigQuery é um serviço de banco de dados e análise de dados em nuvem, que permite consultas rápidas e eficientes em grandes conjuntos de dados. Este armazenamento

na nuvem permite fácil acesso aos dados a partir de diferentes plataformas e facilita o compartilhamento deles.

Por fim, após todo o processo de ETL (Extração, Transformação e Carregamento) a base de dados a partir do carregamento é conectada a duas plataformas de visualização de dados, Power BI e Looker Studio. O primeiro é uma ferramenta da Microsoft para criação de relatórios e dashboards de modo a reunir informações visuais e dinâmicas do conjunto de dados. Por sua vez, o Looker Studio é uma ferramenta do Google também usada para criação de dashboards, além de transformar seus dados em relatórios e painéis informativos totalmente personalizáveis.



FONTE: *ELABORADO PELOS AUTORES.*

6. ANÁLISE DE DADOS

As perguntas de negócio que tentaremos responder com as ferramentas de visualização são:

- Quais são os principais produtos de Origem Animal e Aquicultura em termos de valor e quantidade produzida?
- Quais estados se destacam na produção de Origem Animal e aquicultura em termos de valor total de produção?

- Qual a evolução da produção total (valor) de Origem Animal e Aquicultura ao longo dos anos (2020-2022)?
- Na produção de Origem Animal quais produtos tem maior volume de produção por unidade?
- Na produção de Aquicultura quais produtos tem maior volume de produção?
- Qual o produto que mais vale a pena, em relação a custo e produção?

I. POWER BI

Abaixo seguem os modelos de dashboards construídos pela equipe AgroData para análise da produção de Origem Animal e Aquicultura com a ferramenta Power BI:

Figura: Análise de Aquicultura

Figura: Análise de Origem Animal

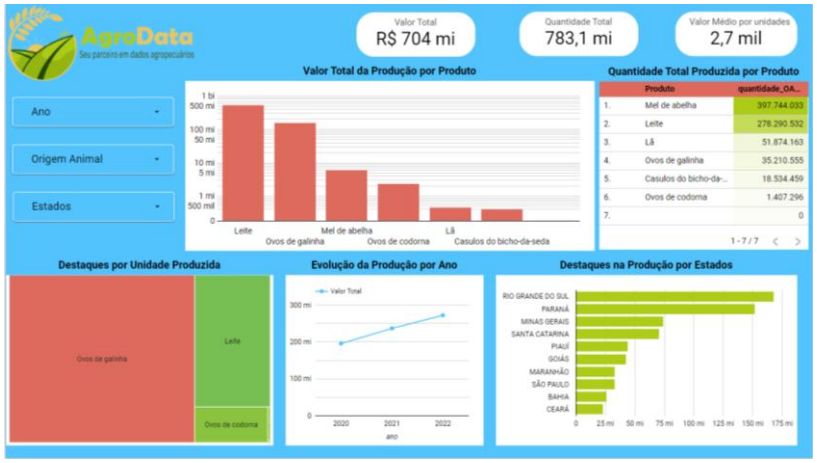
Link Dashboard Power BI:
https://app.powerbi.com/view?r=eyJrIjoiNzJiMjFkZTctMzVhZC00OWJkLTgzMTgtYjRkOWZlNzRhN2E4IiwidCI6IjdiODMwMTM0LWU4ODAtNGNlYS05OGFkLTk5NTg5YjZjODZhOSJ9

II. Looker Studio

Abaixo seguem os modelos de dashboards construídos pela equipe AgroData para análise da produção de Origem Animal e Aquicultura com a ferramenta Looker Studio:

Figura: Análise de Aquicultura

Figura: Análise de Origem Animal



Link Dashboard Looker Studio:
https://lookerstudio.google.com/u/0/reporting/75e47586-8bfa-47cc-bcdd-f798f8714628/page/p_wgm4buczhd

7. ESTRUTURA DO CÓDIGO ETL

Neste tópico será mostrado a estrutura do Script da ETL deste trabalho.

EXTRAÇÃO

- Instalação de pacotes e dependências
- Importação das bibliotecas
- Abertura das bases de dados

TRATAMENTO

- Reduzindo o DataFrame (2020-2022)
- Analisando valores nulos
- Inserindo nova coluna
- Analisando tipos de dados
- Removendo colunas
- Renomeando Colunas
- Unindo DataFrames
- Removendo dados nulos da união
- Verificando inconsistências dos dados

CARREGAMENTO

- Big Query
- Cloud Storage

Link do Colab:
https://colab.research.google.com/drive/1a4LKi5mDp-WiIceVwuI4IVXw0YIXU6E1?usp=sharing#scrollTo=efLAKFIujx15

8. CÓDIGOS ETL

Detalhamento dos códigos da ETL (Extração, Transformação e Carregamento).

Extração

Instalação de pacotes e dependências

```
[ ] # Instalando pacotes necessários para execução da visualização
    !pip install geopandas --quiet
    !pip install plotly --quiet
```

Importações das bibliotecas

```
[ ] # Importando bibliotecas
    from google.cloud import bigquery
    import pandas as pd
    import pandas_gbq
    import geopandas as gpd
    import matplotlib.pyplot as plt
    import plotly.express as px
    import numpy as np
    import plotly.graph_objs as go
    import plotly.io as pio
    import seaborn as sns
```

Abertura das bases de dados

```
[3] # Passo 1: Autenticando o usuário no Google Colab
    from google.colab import auth
    auth.authenticate_user()
```

```
[4] # Passo 2: Comando de abertura para Google Drive
    from google.colab import drive
    drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
```

```
# Passo 3: Importando as bibliotecas do Google BigQuery em um ambiente Python
from google.cloud import bigquery
import pandas as pd
import pandas_gbq
import geopandas as gpd
import matplotlib.pyplot as plt
```

```
[6] # Passo 4: Informando o nome do projeto
    project_id = 'integral-surfer-422200-k6'
    client = bigquery.Client(project=project_id)
```

```
[7] # Passo 5: realizando a conexação da consulta com a bigquery

    # Produção Aquicultura
    query_extracao_PA = client.query('''SELECT * FROM `integral-surfer-422200-k6.basedados.producaoAquicultura`''')


    # Produção de origem Animal
    query_extracao_POA = client.query('''SELECT *  FROM `integral-surfer-422200-k6.basedados.producaoOrigemAnimal`''')


    # Base de dados dos municípios do Brasil
    df_mapa = gpd.read_file('/content/drive/MyDrive/Colab Notebooks/Soul Code/Semana 9/Municipios/BR_Municipios_2022.shp')
```

```
[8] # Lendo o Conjunto de dados
    df_ProducaoAquicultura = query_extracao_PA.to_dataframe()
    df_ProducaoOrigemAnimal = query_extracao_POA.to_dataframe()
```

Tratamento

Reduzindo DataFrame

```
[ ] # Reduzindo o dataframe (anos 2020-2022)
    df_ProducaoAquicultura = df_ProducaoAquicultura[(df_ProducaoAquicultura['ano'] >= 2020) & (df_ProducaoAquicultura['ano'] <= 2022)]

    df_ProducaoOrigemAnimal =  df_ProducaoOrigemAnimal[(df_ProducaoOrigemAnimal['ano'] >= 2020) & (df_ProducaoOrigemAnimal['ano'] <= 2022)]

    print(pd.unique(df_ProducaoAquicultura['ano']))
    print(pd.unique(df_ProducaoOrigemAnimal['ano']))
```

```
[ ] # Removendo linhas com valor zero nas colunas 'quantidade' e 'valor'
    df_ProducaoAquicultura = df_ProducaoAquicultura[(df_ProducaoAquicultura['quantidade'] > 0) & (df_ProducaoAquicultura['valor'] > 0)]

    df_ProducaoOrigemAnimal =  df_ProducaoOrigemAnimal[(df_ProducaoOrigemAnimal['quantidade'] > 0) & (df_ProducaoOrigemAnimal['valor'] > 0)]
```

```
[ ] # Visualizando dataframe Aquicultura
    df_ProducaoAquicultura.head(5)
```

```
[ ] # Visualizando dataframe Origem Animal
    df_ProducaoOrigemAnimal.head(5)
```

```
[ ] # Verificando se há valores zeros
    #df_ProducaoOrigemAnimal[df_ProducaoOrigemAnimal['quantidade'] > 0]
    df_ProducaoOrigemAnimal[df_ProducaoOrigemAnimal.isin([0]).any(axis=1)]
```

```
[ ] # Informações do dataframe
    df_ProducaoOrigemAnimal.info()
    print("\n----------------------------------\n")
    df_ProducaoAquicultura.info()
    print("\n----------------------------------\n")
    df_mapa.info()
```

Analisando valores nulos

```
[ ] # checagem dos dados nulos Origem Animal
    df_ProducaoOrigemAnimal.isnull().sum()
```

```
[ ] # checagem dos dados nulos Aquicultura
    df_ProducaoAquicultura.isnull().sum()
```

```
[ ] # checagem dos dados nulos Mapa
    df_mapa.isnull().sum()
```

Inserindo nova coluna

```
# Inserindo uma nova coluna 'valor de produção por unidade'
df_ProducaoAquicultura['vp_unidade_AC'] = round(df_ProducaoAquicultura['valor']*1000/df_ProducaoAquicultura['quantidade'], 2)

df_ProducaoOrigemAnimal['vp_unidade_OA'] = round(df_ProducaoOrigemAnimal['valor']*1000/df_ProducaoOrigemAnimal['quantidade'], 2)
```

```
# Checando inserção da nova coluna Origem Animal
df_ProducaoAquicultura.sample(5)
```

```
# Checando inserção da nova coluna Aquicultura
df_ProducaoOrigemAnimal.sample(5)
```

Analisando tipos de dados

```
# Analisando os tipos de dados Aquicultura
df_ProducaoAquicultura.dtypes
```

```
# Analisando os tipos de dados Origem Animal
df_ProducaoOrigemAnimal.dtypes
```

```
# Analisando os tipos de dados Mapa
df_mapa.dtypes
```

```
# Alterando o tipo da coluna 'Ano'
df_ProducaoAquicultura['ano'] = df_ProducaoAquicultura['ano'].astype(str)

df_ProducaoOrigemAnimal['ano'] = df_ProducaoOrigemAnimal['ano'].astype(str)
```

```
# Conferindo os tipos de dados Aquicultura
df_ProducaoAquicultura.dtypes
```

```
# Conferindo os tipos de dados Origem Animal
df_ProducaoOrigemAnimal.dtypes
```

Removendo colunas

```
# Informações do dataframe Origem Animal
df_ProducaoOrigemAnimal.head(5)
```

```
# Informações do dataframe Aquicultura
df_ProducaoAquicultura.head(5)
```

```
# Informações do dataframe Mapa
df_mapa.head(5)
```

```
# Remoção das colunas 'sigla_uf' e 'AREA_KM2'
df_ProducaoAquicultura.drop(['sigla_uf'], axis=1, inplace=True)

df_ProducaoOrigemAnimal.drop(['sigla_uf'], axis=1, inplace=True)

df_mapa.drop(['AREA_KM2'], axis=1, inplace=True)
```

```
# Chechando remoção Aquicultura
df_ProducaoAquicultura.head(5)
```

```
# Chechando remoção Origem Animal
df_ProducaoOrigemAnimal.head(5)
```

```
# Chechando remoção Mapa
df_mapa.head(5)
```

Renomeando as colunas

```
# Renomeação
df_ProducaoAquicultura = df_ProducaoAquicultura.rename({'id_municipio': 'CODIGO'}, axis = 1)
df_ProducaoAquicultura = df_ProducaoAquicultura.rename({'produto': 'produto_AC'}, axis = 1)
df_ProducaoAquicultura = df_ProducaoAquicultura.rename({'quantidade': 'quantidade_AC'}, axis = 1)
df_ProducaoAquicultura = df_ProducaoAquicultura.rename({'valor': 'valor_AC'}, axis = 1)
```

```
# Renomeação
df_ProducaoOrigemAnimal = df_ProducaoOrigemAnimal.rename({'id_municipio': 'CODIGO'}, axis = 1)
df_ProducaoOrigemAnimal = df_ProducaoOrigemAnimal.rename({'produto': 'produto_OA'}, axis = 1)
df_ProducaoOrigemAnimal = df_ProducaoOrigemAnimal.rename({'unidade': 'unidade_OA'}, axis = 1)
df_ProducaoOrigemAnimal = df_ProducaoOrigemAnimal.rename({'quantidade': 'quantidade_OA'}, axis = 1)
df_ProducaoOrigemAnimal = df_ProducaoOrigemAnimal.rename({'valor': 'valor_OA'}, axis = 1)
```

```
# Renomeação
df_mapa = df_mapa.rename({'CD_MUN': 'CODIGO'}, axis = 1)
df_mapa = df_mapa.rename({'NM_MUN': 'NOME_MUNICIPIO'}, axis = 1)
```

```
# Chechando renomeação Aquicultura
df_ProducaoAquicultura.head(5)
```

```
# Chechando renomeação Origem Animal
df_ProducaoOrigemAnimal.head(5)
```

```
# Chechando renomeação Mapa
df_mapa.head(5)
```

Unindo DataFrames

```
# Verificando o tamanho do dataframe Aquicultura
df_ProducaoAquicultura.shape
```

```
# Verificando o tamanho do dataframe Origem Animal
df_ProducaoOrigemAnimal.shape
```

```
# Verificando o tamanho do dataframe Mapa
df_mapa.shape
```

```
# União dos dataframe Origem Animal e Aquicultura
uniao_OA_AC = df_ProducaoOrigemAnimal.merge(df_ProducaoAquicultura, on = ['CODIGO', 'ano'], how = "outer")
```

```
# Chechando união dos dataframe
uniao_OA_AC.head(5)
```

```
# Verificando o tamanho do dataframe
uniao_OA_AC.shape
```

```
# Verifcando valores nulos do cruzamento
uniao_OA_AC.isnull().sum()
```

```
# Retorno das linhas do DataFrame onde a coluna 'produto_OA' contém valores nulos
uniao_OA_AC[uniao_OA_AC['produto_OA'].isna()]
```

```
# Retorno das linhas do DataFrame onde a coluna 'produto_AC' contém valores nulos
uniao_OA_AC[uniao_OA_AC['produto_AC'].isna()]
```

```
[ ] # Substituindo os dados nulos por 0
    # Mudança para zero, no casos em que tem dados de Origem Animal, mas não tem de AquiCultura

    uniao_OA_AC['produto_OA'] = uniao_OA_AC['produto_OA'].fillna('')
    uniao_OA_AC['unidade_OA'] = uniao_OA_AC['unidade_OA'].fillna('')
    uniao_OA_AC['produto_AC'] = uniao_OA_AC['produto_AC'].fillna('')

    uniao_OA_AC.fillna(0, inplace=True)
```

```
[ ] # Checagem para verificar se ainda tem valores nulos
    uniao_OA_AC[uniao_OA_AC['produto_OA'].isna()]
```

```
[ ] # Checando a mudança
    uniao_OA_AC.isnull().sum()
```

```
[ ] # União do novo dataframe uniao_oficial a base de dados Mapa
    uniao_oficial = df_mapa.merge(uniao_OA_AC, how = "left", on = 'CODIGO')
```

```
[ ] # Chechando união
    uniao_oficial.head(5)
```

```
[ ] # Verificando o tamanho do dataframe
    uniao_oficial.shape
```

```
[ ] # Vericando tipos de dados
    uniao_oficial.dtypes
```

Removendo dados nulos da União

```
[ ] # checagem dos dados nulos
    uniao_oficial.isnull().sum()
```

```
[ ] # Filtrando os valores vazios
    uniao_oficial[uniao_oficial['ano'].isna()]
```

```
[ ] # Eliminando os valores vazios
    uniao_oficial = uniao_oficial.dropna()
```

```
[ ] # checagem dos dados nulos
    uniao_oficial.isnull().sum()
```

```
[ ] # Verificando o tamanho do dataframe
    uniao_oficial.shape
```

```
[ ] # Iterando sobre as colunas do DataFrame
    for column in uniao_oficial.columns:

        # Verificando se a coluna contém valores inteiros
        if uniao_oficial[column].dtype == 'int':
            print(f"Valores inteiros únicos na coluna '{column}':")
            print(sorted(uniao_oficial[column].unique()))
            print("-" * 50)  # Linha de separação

        # Verificando se a coluna contém valores de ponto flutuante
        elif uniao_oficial[column].dtype == 'float':
            print(f"Valores decimais únicos na coluna '{column}':")
            print(sorted(uniao_oficial[column].unique()))
            print("-" * 50)  # Linha de separação

        # Verificando se a coluna contém valores de texto (string)
        elif uniao_oficial[column].dtype == 'object':
            print(f"Valores de texto únicos na coluna '{column}':")
            print(sorted(uniao_oficial[column].unique()))
            print("-" * 50)  # Linha de separação
```

Verificando inconcistências dos dados

```
[ ] # instalação do pacote
    !pip install pandera --quiet
```

```
[ ] # importando o  pacote pandera
    import pandera as pa
```

```
[ ] # Tipos de dados
    df = uniao_oficial.drop(['geometry'], axis=1)
    df.dtypes
```

```
[ ] # Definição do esquema de validação
    schema = pa.DataFrameSchema({'CODIGO': pa.Column(pa.String),
                                 'NOME_MUNICIPIO': pa.Column(pa.String),
                                 'SIGLA_UF': pa.Column(pa.String),
                                 'ano': pa.Column(pa.String),
                                 'produto_OA': pa.Column(pa.String),
                                 'unidade_OA': pa.Column(pa.String),
                                 'quantidade_OA': pa.Column(pa.Int64),
                                 'valor_OA': pa.Column(pa.Int64),
                                 'vp_unidade_OA': pa.Column(pa.Float64),
                                 'produto_AC': pa.Column(pa.String),
                                 'quantidade_AC': pa.Column(pa.Int64),
                                 'valor_AC': pa.Column(pa.Int64),
                                 'vp_unidade_AC': pa.Column(pa.Float64)
                                })
```

```
[ ] # Validar o DataFrame
    schema.validate(df)
```

Carregamento

BigQuery

```
# Carregando dados na Bigquery
table_id = 'integral-surfer-422200-k6.basedados.dados_tratados'
project_id = 'integral-surfer-422200-k6'
pandas_gbq.to_gbq(df, table_id, project_id=project_id)
```

Cloud Storage

```
# Instalação de pacote
!pip install gcsfs
```

```
# Importando bibliotecas
import os
import pandas as pd
from google.cloud import storage
```

```
# Configurando a chave de segurança - Acesso ao projeto
serviceAccount = '/content/drive/MyDrive/Meu PC/chaves/agrodata.json'
os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = serviceAccount
```

```
# Configurando Google Cloud Storage - Acesso ao Bucket
client = storage.Client()                                    # Autenticação do cliente
bucket = client.get_bucket('agropop')                        # nome do bucket
bucket.blob('dados_tratados.csv')                            # nome do arquivo
path = 'gs://agropop/dados_tratados.csv'                     # caminho do arquivo
```

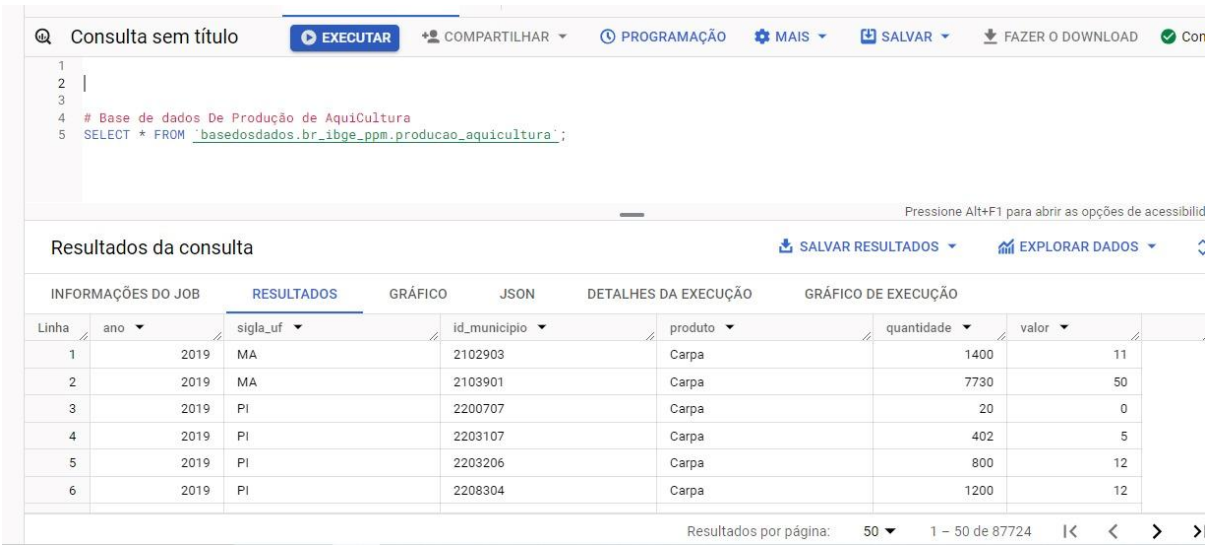
```
# Fase de Carregamento na Bucket
df.to_csv(path, index=True)
```

9. CONSULTAS REALIZADAS NA BIG QUERY

O BigQuery é um serviço de data warehouse totalmente gerenciado e altamente escalável oferecido pela Google Cloud Platform. Ele permite armazenar, consultar e analisar grandes conjuntos de dados usando SQL.

Durante o projeto foram realizadas algumas consultas na Big Query, elas serão apresentadas a seguir:

Figura: Extraindo Base de Dados



Linha	ano	sigla_uf	id_municipio	produto	quantidade	valor
1	2019	MA	2102903	Carpa	1400	11
2	2019	MA	2103901	Carpa	7730	50
3	2019	PI	2200707	Carpa	20	0
4	2019	PI	2203107	Carpa	402	5
5	2019	PI	2203206	Carpa	800	12
6	2019	PI	2208304	Carpa	1200	12

Figura: Extraindo Base de Dados

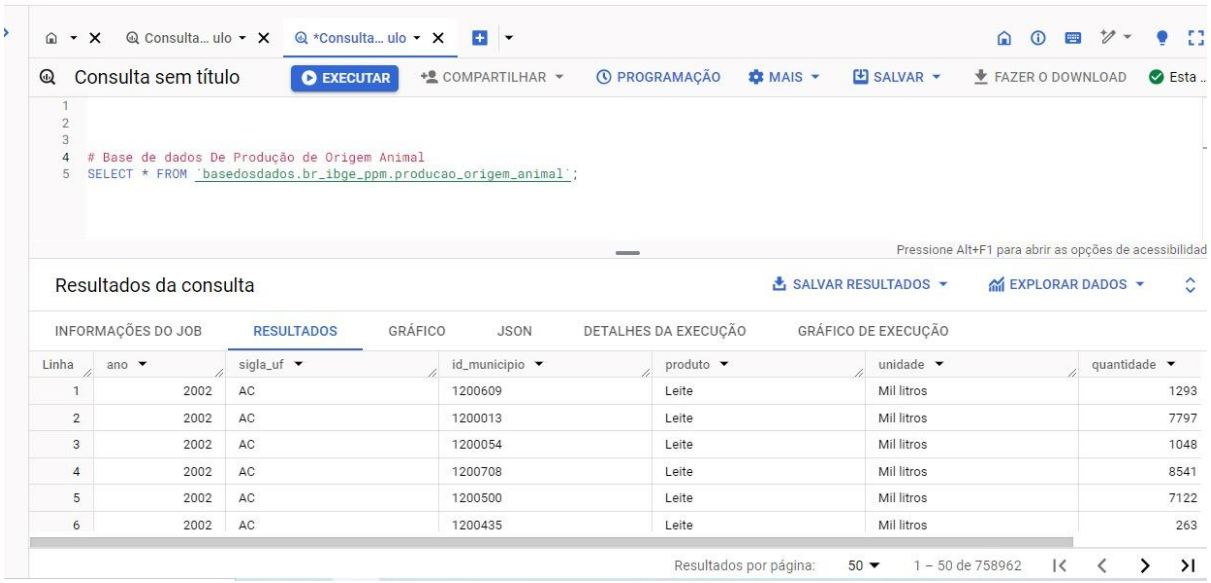


Figura: Update após tratamento

```
UPDATE integral-surfer-422200-k6.basedados.dados_tratados
SET SIGLA_UF = CASE
        WHEN SIGLA_UF = 'AC' THEN 'ACRE'
        WHEN SIGLA_UF = 'AL' THEN 'ALAGOAS'
        WHEN SIGLA_UF = 'AP' THEN 'AMAPÁ'
        WHEN SIGLA_UF = 'AM' THEN 'AMAZONAS'
        WHEN SIGLA_UF = 'BA' THEN 'BAHIA'
        WHEN SIGLA_UF = 'CE' THEN 'CEARÁ'
        WHEN SIGLA_UF = 'DF' THEN 'DISTRITO FEDERAL'
        WHEN SIGLA_UF = 'ES' THEN 'ESPÍRITO SANTO'
        WHEN SIGLA_UF = 'GO' THEN 'GOIÁS'
        WHEN SIGLA_UF = 'MA' THEN 'MARANHÃO'
        WHEN SIGLA_UF = 'MT' THEN 'MATO GROSSO'
        WHEN SIGLA_UF = 'MS' THEN 'MATO GROSSO DO SUL'
        WHEN SIGLA_UF = 'MG' THEN 'MINAS GERAIS'
        WHEN SIGLA_UF = 'PA' THEN 'PARÁ'
        WHEN SIGLA_UF = 'PB' THEN 'PARAÍBA'
        WHEN SIGLA_UF = 'PR' THEN 'PARANÁ'
        WHEN SIGLA_UF = 'PE' THEN 'PERNAMBUCO'
        WHEN SIGLA_UF = 'PI' THEN 'PIAUÍ'
        WHEN SIGLA_UF = 'RJ' THEN 'RIO DE JANEIRO'
        WHEN SIGLA_UF = 'RN' THEN 'RIO GRANDE DO NORTE'
        WHEN SIGLA_UF = 'RS' THEN 'RIO GRANDE DO SUL'
        WHEN SIGLA_UF = 'RO' THEN 'RONDÔNIA'
        WHEN SIGLA_UF = 'RR' THEN 'RORAIMA'
        WHEN SIGLA_UF = 'SC' THEN 'SANTA CATARINA'
        WHEN SIGLA_UF = 'SP' THEN 'SÃO PAULO'
        WHEN SIGLA_UF = 'SE' THEN 'SERGIPE'
        WHEN SIGLA_UF = 'TO' THEN 'TOCANTINS'
    END
WHERE SIGLA_UF != ''
```