Extração e Tratamento dos Dados

In [2]:

```
#pip install pyodbc
# importar bibliotecas
import pandas as pd
import numpy as np
import io
import pyodbc

```

In [2]:

```
# Conexão direta com o banco SQLServer Importação da tabela de expedientes Situações
conn = pyodbc.connect('Driver={SQL Server};Server=anvssdf522;UID=alesandre.santos;PW
query = 'SELECT * FROM ta_historico_peticao'
df = pd.read_sql_query(query, conn)
#df = df0.head(1000)
```

...

In [13]:

```
# Leitura do arquivo congelado em 01/07/2023
df = pd.read_excel(r'C:\Users\alesa\OneDrive - ANVISA - Agencia Nacional de Vigilanc
                           header=0, sheet_name="Planilha1")

```

In [3]:

```
#df.head()
#df.info()
#base_a = base_a.dropna()
#base_a.info()
#base_a.isnull().sum()
#base_a.head()
```

In [9]:

```
# Importação das tabelas auxiliares e seleção de colunas
assuntos = pd.read_excel(r'C:\Users\alesa\OneDrive - ANVISA - Agencia Nacional de Vi
                         header=0, sheet_name="ASSUNTOS")
status   = pd.read_excel(r'C:\Users\alesa\OneDrive - ANVISA - Agencia Nacional de Vi
                         header=0, sheet_name="STATUS_NIVEL_2")

colunas_para_excluir0 = ['ContarDeCO_ASSUNTO2', 'MáxDeDATA_SITUACAO', 'STATUS_3_GGTC
                         'ciclo' ]
status = status.drop(colunas_para_excluir0, axis=1)

crit_fila   = pd.read_excel(r'C:\Users\alesa\OneDrive - ANVISA - Agencia Nacional de
                         header=0, sheet_name="FILA_ANALISE_ASSUNTO_SITUACAO")
crit_fila = crit_fila[['CO_ASSUNTO','DS_SITUACAO_ASSUNTO_DOC','TIPO_LISTA_FILA', 'FI
#crit_fila.info()
```

In [16]:

```
# Transformando datas do formato texto para o formato data e excluindo expediente co
df = pd.DataFrame(df[(df['NU_EXPEDIENTE'] != 977988114) &
                     (df['NU_EXPEDIENTE'] != 36605116)])
df['DT_ENTRADA'] = pd.to_datetime(df['DT_RECEBIMENTO_ANVISA_TOX'],format="%Y-%m-%d %
df['DT_INICIO_SITUACAO'] = pd.to_datetime(df['DT_INICIO_SITUACAO'],format="%Y-%m-%d
df['DT_FIM_SITUACAO'] = pd.to_datetime(df['DT_FIM_SITUACAO'],format="%Y-%m-%d %H:%M:
df['DT_PUBLICACAO'] = pd.to_datetime(df['DT_PUBLICACAO'],format="%Y-%m-%d %H:%M:%S.%
df['DT_CARGA'] = pd.to_datetime(df['DT_CARGA'],format="%Y-%m-%d %H:%M:%S.%f")
df['CO_ASSUNTO'] = df['CO_ASSUNTO'].astype('int64')
#df.info()
```

In [17]:

```
# Define a função para formatar o CNPJ e Expediente
def formatar_cnpj(NU_CNPJ_EMPRESA):
    return "{}.{}.{}/{}-{}".format(NU_CNPJ_EMPRESA[:2], NU_CNPJ_EMPRESA[2:5], NU_CNP

def formatar_exped(NU_EXPEDIENTE):
    return "{}/{}-{}".format(NU_EXPEDIENTE[:7],  NU_EXPEDIENTE[7:8], NU_EXPEDIENTE[8

# Cria a nova coluna com o CNPJ formatado
df['NU_CNPJ_EMPRESA'] = df['NU_CNPJ_EMPRESA'].astype('str')
df['NU_CNPJ_EMPRESA'] = df['NU_CNPJ_EMPRESA'].apply(formatar_cnpj)

df['NU_EXPEDIENTE'] = df['NU_EXPEDIENTE'].astype('str')
df['NU_EXPEDIENTE_f'] = df['NU_EXPEDIENTE'].apply(formatar_exped)

df['NU_EXPEDIENTE'] = df['NU_EXPEDIENTE'].astype('int64')

#df.head()
```

In [18]:

```
# Criando df de Recursos
df_r = pd.DataFrame((df[(df['CO_ASSUNTO'] == 5062)]))

# Criando df de Cancelados
df_c = pd.DataFrame(df[(df['DS_SITUACAO_ASSUNTO_DOC'] == "Cancelado a pedido da empr
                        (df['DS_SITUACAO_ASSUNTO_DOC'] == "Desistência a pedido")|
                        (df['DS_SITUACAO_ASSUNTO_DOC'] == "Petição encerrada")|
                        (df['DS_SITUACAO_ASSUNTO_DOC'] == "Arquivado")|
                        (df['DS_SITUACAO_ASSUNTO_DOC'] == "Arquivado a pedido")])

#df_c.head()
#Agrupando df de Recursos e Cancelados por Número do Processo
df_ra = df_r.groupby('NU_PROCESSO').size().reset_index(name='RECURSO')
df_ca = df_c.groupby('NU_PROCESSO').size().reset_index(name='CANCELADO')
```

In [19]:

```
# Relacionando Tabela principal com tabela de Recursos e Cancelados
df_001 = df.merge(df_ra, on="NU_PROCESSO", how="left").merge(df_ca, on="NU_PROCESSO"

# Relacionando Tabela principal com tabela de Assuntos
df_002 = df_001.merge(assuntos, on="CO_ASSUNTO", how="left")

# Exclui colunas que não são de interesse
colunas_para_excluir = ['Fato_Gerador', 'obs',  'Tipo_agrofit', 'DT_RECEBIMENTO_ANVI
                        'prazo_336']
df_002 = df_002.drop(colunas_para_excluir, axis=1)

# Seleciona somente expedientes de Registro e Publicação
df_002 = pd.DataFrame(df_002[(df_002['TIPO_PUBLICACAO'] == "1. Registro") |
                             (df_002['TIPO_PUBLICACAO'] == "2. Pós-Registro")])

#df_002.info()
```

In [22]:

```
# Criando a informação de ultima situação  no processo do expediente mais recente
ult_status = df_002.groupby(['NU_PROCESSO', 'NU_EXPEDIENTE'])['DT_INICIO_SITUACAO'].

base_dv_0  = df_002.merge(ult_status, on=['NU_PROCESSO', 'NU_EXPEDIENTE','DT_INICIO_

ult_status_r = base_dv_0.groupby(['NU_PROCESSO', 'NU_EXPEDIENTE','DT_INICIO_SITUACAO

ult_status_r = pd.DataFrame(ult_status_r[(ult_status_r['QTD_ERRO'] >= 2)])

base_dv_1  = base_dv_0.merge(ult_status_r, on=['NU_PROCESSO', 'NU_EXPEDIENTE','DT_IN
base_dv_1['MARCADOR'] = np.where((base_dv_1['QTD_ERRO'] >= 2) & (base_dv_1['DT_FIM_S
base_dv_1 = pd.DataFrame(base_dv_1[(base_dv_1['MARCADOR'] == 0)])
base_dv_1['ULT_SIT'] = base_dv_1['DS_SITUACAO_ASSUNTO_DOC']
base_dv_1['DT_INICIO_UL_SITUACAO'] = base_dv_1['DT_INICIO_SITUACAO']

base_dv_1 = base_dv_1[['NU_PROCESSO', 'NU_EXPEDIENTE','ULT_SIT','DT_INICIO_UL_SITUAC
#base_dv_1.info()
df_003  = df_002.merge(base_dv_1, on=['NU_PROCESSO', 'NU_EXPEDIENTE'], how="left")
df_004 = df_003.merge(status, left_on="ULT_SIT", right_on="DS_SITUACAO_ASSUNTO_DOC",

df_004 = df_004.drop(['DS_SITUACAO_ASSUNTO_DOC_y'], axis=1)
df_004['DS_SITUACAO_ASSUNTO_DOC'] = df_004['DS_SITUACAO_ASSUNTO_DOC_x']
df_004['STATUS_2_GGTOX_P'] = df_004['STATUS_2_GGTOX']

df_004 = df_004.drop(['DS_SITUACAO_ASSUNTO_DOC_x','STATUS_2_GGTOX', 'CICLO_GGTOX','c

df_005 = df_004.merge(status, left_on="DS_SITUACAO_ASSUNTO_DOC", right_on="DS_SITUAC
df_005['data_p_nula'] = df_005['DT_PUBLICACAO'].isnull().astype(int)

# Se Finalizado e data de publicação = nulo então data finalização = data situação e
df_005['DT_FINALIZACAO'] = np.where((df_005['data_p_nula'] == 1) & (df_005['STATUS_2
                                    df_005['DT_INICIO_UL_SITUACAO'], df_005['DT_PUBL
df_005['DATA_FINALIZACAO'] = df_005['DT_FINALIZACAO'].dt.date
df_005['DATA_ENTRADA'] = df_005['DT_ENTRADA'].dt.date
df_005= df_005.drop(['data_p_nula'], axis=1)

df_006 = df_005.merge(crit_fila, on=['CO_ASSUNTO', 'DS_SITUACAO_ASSUNTO_DOC'], how="
df_006 = pd.DataFrame(df_006[(df_006['ULT_SIT'] == df_006['DS_SITUACAO_ASSUNTO_DOC']
                             (df_006['DT_INICIO_UL_SITUACAO'] == df_006['DT_INICIO_S
```

In [23]:

```
# Criando outputs: Andamento, Saída e Entrada

ANDAMENTO = pd.DataFrame(df_006[(df_006['STATUS_2_GGTOX_P'] == "Em Processamento") &
                                (df_006['CO_ASSUNTO'] != 5015) &
                                (df_006['ULT_SIT'] == df_006['DS_SITUACAO_ASSUNTO
                                (df_006['DT_INICIO_UL_SITUACAO'] == df_006['DT_IN

FINALIZADOS = pd.DataFrame(df_006[(df_006['STATUS_2_GGTOX_P'] == "Finalizados") &
                                (df_006['ULT_SIT'] == df_006['DS_SITUACAO_ASSUNTO
                                (df_006['DT_INICIO_UL_SITUACAO'] == df_006['DT_IN

SAIDA_DV = FINALIZADOS.groupby(['CO_ASSUNTO','DATA_FINALIZACAO']).size().reset_index

ENTRADA_DV = df_006.groupby(['CO_ASSUNTO','DATA_ENTRADA']).size().reset_index(name='
```

In [24]:

```
# Criando outputs: Indicadores finalizados (Situações, fases, dias)

IND_FINALIZADOS = pd.DataFrame(df_005[(df_005['STATUS_2_GGTOX_P'] == "Finalizados")]

IND_FINALIZADOS = IND_FINALIZADOS.sort_values(by=['NU_PROCESSO', 'NU_EXPEDIENTE', 'C
                           ascending=[True,True,True, False, True])

IND_FINALIZADOS['DT_FIM_SITUACAO_LAG'] = IND_FINALIZADOS['DT_INICIO_SITUACAO'].shift

IND_FINALIZADOS = pd.DataFrame(IND_FINALIZADOS[(IND_FINALIZADOS['DT_INICIO_SITUACAO'

IND_FINALIZADOS['DIAS'] = np.where((IND_FINALIZADOS['NU_PROCESSO'] == IND_FINALIZADO
                                   (IND_FINALIZADOS['NU_EXPEDIENTE'] == IND_FINALIZA
                                   IND_FINALIZADOS.apply(lambda row: (row['DT_FIM_SI
                                   0)
IND_FINALIZADOS = pd.DataFrame(IND_FINALIZADOS[(IND_FINALIZADOS['DIAS'] > 0)])

IND_FINALIZADOS['CICLO_GGTOX'] = np.where((IND_FINALIZADOS['CICLO_GGTOX'].isnull()),
IND_FINALIZADOS['SAIDA_A'] = IND_FINALIZADOS['DT_FINALIZACAO'].dt.year
IND_FINALIZADOS['SAIDA_M'] = IND_FINALIZADOS['DT_FINALIZACAO'].dt.month
IND_FINALIZADOS['ENTRADA_A'] = IND_FINALIZADOS['DT_ENTRADA'].dt.year
IND_FINALIZADOS['ENTRADA_M'] = IND_FINALIZADOS['DT_ENTRADA'].dt.month

# Criando Fase '6. Tempo Total'
IND_FINALIZADOS_T = IND_FINALIZADOS.groupby(['SAIDA_A','SAIDA_M','ENTRADA_A','ENTRAD
                                   'CO_ASSUNTO','NU_PROCESSO','NU_EXPEDIENTE']).agg({




IND_FINALIZADOS_T['DT_INICIO_SITUACAO'] = IND_FINALIZADOS_T['DT_ENTRADA']
IND_FINALIZADOS_T['DT_FIM_SITUACAO'] = IND_FINALIZADOS_T['DT_FINALIZACAO']
IND_FINALIZADOS_T['CICLO_GGTOX'] = '6.Tempo_total'

IND_FINALIZADOS_T['DIAS'] = IND_FINALIZADOS_T.apply(lambda row: (row['DT_FINALIZACAO

IND_FINAL = pd.concat([IND_FINALIZADOS_T,IND_FINALIZADOS])
IND_FINAL = pd.DataFrame(IND_FINAL[(IND_FINAL['DIAS'] > 0)])
IND_FINAL_1 = IND_FINAL[['SAIDA_A','SAIDA_M','ENTRADA_A','ENTRADA_M','CO_ASSUNTO','N
                         'CANCELADO','DT_FINALIZACAO','DT_ENTRADA','DT_INICIO_SITUACAO
                         'CICLO_GGTOX','DIAS','NU_CNPJ_EMPRESA','NO_RAZAO_SOCIAL_EMPRE
#IND_FINAL.info()
```

In [30]:

```
# Exportando saidas para Excel

ANDAMENTO.to_excel('ANDAMENTO.xlsx', sheet_name='Planilha1', index=False)
SAIDA_DV.to_excel('SAIDA_DV.xlsx', sheet_name='Planilha1', index=False)
ENTRADA_DV.to_excel('ENTRADA_DV.xlsx', sheet_name='Planilha1', index=False)
IND_FINAL_1.to_excel('IND_FINAL.xlsx', sheet_name='Planilha1', index=False)
```