analise

September 4, 2023

1 1. Configurações

[ ]:
```
# Importando bibliotecas do python necessárias
import pandas as pd
import numpy as np
from dbfread import DBF
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.graph_objects as go
import colorsys
import warnings
```

[ ]:
```
# Configurando o notebook
warnings.filterwarnings('ignore')
```

[ ]:
```
# Setando os paths dos datasets
PATH_SIH = "/mnt/c/Users/gcmas/OneDrive/Área de Trabalho/S3 Biotech/
 ↪Desenvolvimento/ETL Growth/Datasets/sih_rd_pi.csv"
PATH_DEMOGRAPHIC = "/mnt/c/Users/gcmas/OneDrive/Área de Trabalho/S3 Biotech/
 ↪Desenvolvimento/ETL Growth/Datasets/DemografiaPI.csv"
PATH_CID = '/mnt/c/Users/gcmas/OneDrive/Área de Trabalho/S3 Biotech/
 ↪Desenvolvimento/ETL Growth/Datasets/cid10.dbf'
PATH_ESPEC_LEITOS = "/mnt/c/Users/gcmas/OneDrive/Área de Trabalho/S3 Biotech/
 ↪Desenvolvimento/ETL Growth/Datasets/leitos_espec.csv"
```

2 2. Aquisição de Dados

[ ]:
```
# Lendo o dataset publico do SIH de Janeiro de 2022 a Junho de 2023
df_sih = pd.read_csv(PATH_SIH, sep=";")
df_sih.head()
```

[ ]:

	UF_ZI	ANO_CMPT	MES_CMPT	ESPEC	CGC_HOSP	N_AIH	IDENT	\
0	220000	2022	1	3	6.097574e+13	2222101157530	1	
1	220000	2022	1	3	6.097574e+13	2222101157540	1	
2	220000	2022	1	3	6.097574e+13	2222101157551	1	
3	220000	2022	1	2	NaN	2222101081519	1	

```
4  220000      2022         1      2           NaN  2222101081574      1

        CEP  MUNIC_RES      NASC  …  TPDISEC1  TPDISEC2  TPDISEC3  TPDISEC4  \
0  64253000     220635  19930106  …       0.0       0.0       0.0       0.0
1  64258000     220557  19500501  …       0.0       0.0       0.0       0.0
2  64258000     220557  19740209  …       0.0       0.0       0.0       0.0
3  64110000     220550  19860416  …       0.0       0.0       0.0       0.0
4  64110000     220550  20040424  …       0.0       0.0       0.0       0.0

   TPDISEC5  TPDISEC6  TPDISEC7  TPDISEC8  TPDISEC9  DataRefAquivo
0       0.0       0.0       0.0       0.0       0.0           2201
1       0.0       0.0       0.0       0.0       0.0           2201
2       0.0       0.0       0.0       0.0       0.0           2201
3       0.0       0.0       0.0       0.0       0.0           2201
4       0.0       0.0       0.0       0.0       0.0           2201

[5 rows x 114 columns]
```

[ ]:
```
# Verificando o dados nulos nos Municípios de Residência
df_sih["MUNIC_RES"].isnull().sum()
```

[ ]:
```
0
```

[ ]:
```
# Verificando o dados nulos nos Municípios de Atendimento
df_sih["MUNIC_MOV"].isnull().sum()
```

[ ]:
```
6
```

[ ]:
```
# Dropando os dados nulos nos Municípios de Atendimento
df_sih.dropna(subset=["MUNIC_MOV"], inplace=True)
```

2.1 2.1 Enriquecimento de Dados

2.1.1 2.1.1 Dados de Demografia

[ ]:
```
# Lendo o dataset de features demograficas do Piauí
df_demographic = pd.read_csv(PATH_DEMOGRAPHIC, sep=",")
df_demographic.head()
```

[ ]:
```
   CO_MUNICIPIO_GESTOR          Municipio              Mesoregiao  \
0               220005              Acauã      Sudeste Piauiense
1               220010       Agricolândia  Centro Norte Piauiense
2               220020        Água Branca  Centro Norte Piauiense
3               220025  Alagoinha do Piauí      Sudeste Piauiense
4               220027   Alegrete do Piauí      Sudeste Piauiense

        RegionaldeSaude DensidadeDemografica_2010_  PopulacaoEstimada_2021_  \
0  Vale Do Rio Guaribas                       5,27                     7119
```

```
1            Entre Rios                    45,35                  5123
2            Entre Rios                   169,53                 17525
3  Vale Do Rio Guaribas                    13,77                  7678
4  Vale Do Rio Guaribas                    18,23                  4921

  IdadeMediaPop PopRelPediatrica PopRelIdoso  IDHM_2010_ PIBPerCapita_2020_
0         33,16            28,22       10,72         528            8819,73
1         36,04            25,95       15,45         599            8453,12
2         34,26             27,9       12,64         639           12537,11
3         36,04            23,29       12,69         531            8772,48
4         34,28            26,31       11,96         585           11207,05
```

[ ]:
```
# Selecionando as colunas necessárias para o estudo
df_mun_name = df_demographic[["CO_MUNICIPIO_GESTOR", "Municipio", "IDHM_2010_"]]
df_mun_name
```

[ ]:
```
    CO_MUNICIPIO_GESTOR           Municipio  IDHM_2010_
0                220005               Acauã         528
1                220010        Agricolândia         599
2                220020         Água Branca         639
3                220025  Alagoinha do Piauí         531
4                220027   Alegrete do Piauí         585
..                  …                   …         …
219              221135       Várzea Branca         553
220              221140       Várzea Grande         571
221              221150         Vera Mendes         503
222              221160  Vila Nova do Piauí         565
223              221170         Wall Ferraz         544

[224 rows x 3 columns]
```

[ ]:
```
# Enriquecendo informações sobre o município de gestor
df_mun_name_ufzi = df_mun_name.copy()
df_mun_name_ufzi.columns = ["UF_ZI", "MUNIC_UFZI_NOME", "IDHM_UFZI"]
df_sih = pd.merge(df_sih, df_mun_name_ufzi, on="UF_ZI", how="left")
```

[ ]:
```
# Enriquecendo informações sobre o município de residência
df_mun_name_res = df_mun_name.copy()
df_mun_name_res.columns = ["MUNIC_RES", "MUNIC_RES_NOME", "IDHM_RES"]
df_sih = pd.merge(df_sih, df_mun_name_res, on="MUNIC_RES", how="left")
```

[ ]:
```
# Enriquecendo informações sobre o município de atendimento
df_mun_name_mov = df_mun_name.copy()
df_mun_name_mov.columns = ["MUNIC_MOV", "MUNIC_MOV_NOME", "IDHM_MOV"]
df_sih = pd.merge(df_sih, df_mun_name_mov, on="MUNIC_MOV", how="left")
```

[ ]:
```
# Substituindo valores nulos em municipios não reconhecidos como Fora do Piauí
df_sih["MUNIC_MOV_NOME"].fillna("Fora do Piauí", inplace=True)
df_sih["MUNIC_RES_NOME"].fillna("Fora do Piauí", inplace=True)
```

2.1.2 2.1.2 Dados de Especialidade

[ ]:
```
# Lendo o dataset de especialidades de leitos
df_espec_leitos = pd.read_csv(PATH_ESPEC_LEITOS, delimiter=';',␣
 ↪encoding='latin-1')
df_espec_leitos
```

[ ]:

	COD_ESPEC	DESC_ESPEC
0	1	Cirurgico
1	2	Obstetricos
2	3	Clinico
3	4	Cronicos
4	5	Psiquiatria
5	6	Pneumologia Sanitaria (Tisiologia)
6	7	Pediatricos
7	8	Reabilitacao
8	9	Leito Dia / Cirurgicos
9	10	Leito Dia / Aids
10	11	Leito Dia / Fibrose Cistica
11	12	Leito Dia / Intercorrencias / Transplante
12	13	Leito Dia / Geriatria
13	14	Leito Dia / Saude Mental
14	51	UTI II Adulto COVID
15	52	UTI II Pediatrica COVID
16	64	Unidade Intermediaria
17	65	Unidade Intermediaria Neonatal
18	74	UTI I
19	75	UTI Adulto II
20	76	UTI Adulto III
21	77	UTI Infantil I
22	78	UTI Infantil II
23	79	UTI Infantil III
24	80	UTI Neonatal I
25	81	UTI Neonatal II
26	82	UTI Neonatal III
27	83	UTI Queimados
28	84	Acolhimento Noturno
29	85	UTI Coronariana-UCO tipo II
30	86	UTI Coronariana-UCO tipo III
31	87	Saude Mental (Clinico)
32	88	Queimado Adulto (Clinico)
33	89	Queimado Pediatrico (Clinico)
34	90	Queimado Adulto (Cirurgico)

```
35         91                    Queimado Pediatrico (Cirurgico)
36         92  UCI Unidade de Cuidados Intermediarios Neonata…
37         93  UCI Unidade de Cuidados Intermediarios Neonata…
38         94  UCI Unidade de Cuidados Intermediarios Pediatrico
39         95      UCI Unidade de Cuidados Intermediarios Adulto
40         96           Suporte Ventilati¿½rio Pulmonar COVID-19
```

[ ]:
```
# Enriquencendo informações sobre a especialidade
df_espec_leitos.columns = ["ESPEC", "DESC_ESPEC"]
df_sih["ESPEC"] = df_sih["ESPEC"].astype(int)
df_sih = pd.merge(df_sih, df_espec_leitos, on="ESPEC", how="left")
```

2.1.3 2.1.3 Dados de CID10

[ ]:
```
# Lendo o dataset dicionário de CID 10
table_cid = DBF(PATH_CID, encoding='iso-8859-1')
df_cid = pd.DataFrame(iter(table_cid))
df_cid
```

[ ]:
```
      CD_COD                                          CD_DESCR
0        A00                                      A00   Colera
1       A000  A00.0 Colera dev Vibrio cholerae 01 biot cholerae
2       A001    A00.1 Colera dev Vibrio cholerae 01 biot El Tor
3       A009                                 A00.9 Colera NE
4        A01              A01   Febres tifoide e paratifoide
…       …                                                …
14524   Z991                 Z99.1 Dependencia de respirador
14525   Z992               Z99.2 Dependencia de dialise renal
14526   Z993            Z99.3 Dependencia de cadeira de rodas
14527   Z998     Z99.8 Depend outr maq e aparelhos capacitantes
14528   Z999     Z99.9 Depend maquina e aparelho capacitante NE

[14529 rows x 2 columns]
```

[ ]:
```
# Enriquencendo informações sobre a CID do diagnóstico principal
df_cid.columns = ["DIAG_PRINC", "DESC_DIAG_PRINC"]
df_sih = pd.merge(df_sih, df_cid, on="DIAG_PRINC", how="left")
```

2.2 2.2 Visualização do Dataset Final

[ ]:
```
# Visualizando o dataset final
df_sih.head()
```

[ ]:
```
    UF_ZI  ANO_CMPT  MES_CMPT  ESPEC      CGC_HOSP          N_AIH  IDENT  \
0  220000      2022         1      3  6.097574e+13  2222101157530      1
1  220000      2022         1      3  6.097574e+13  2222101157540      1
2  220000      2022         1      3  6.097574e+13  2222101157551      1
```

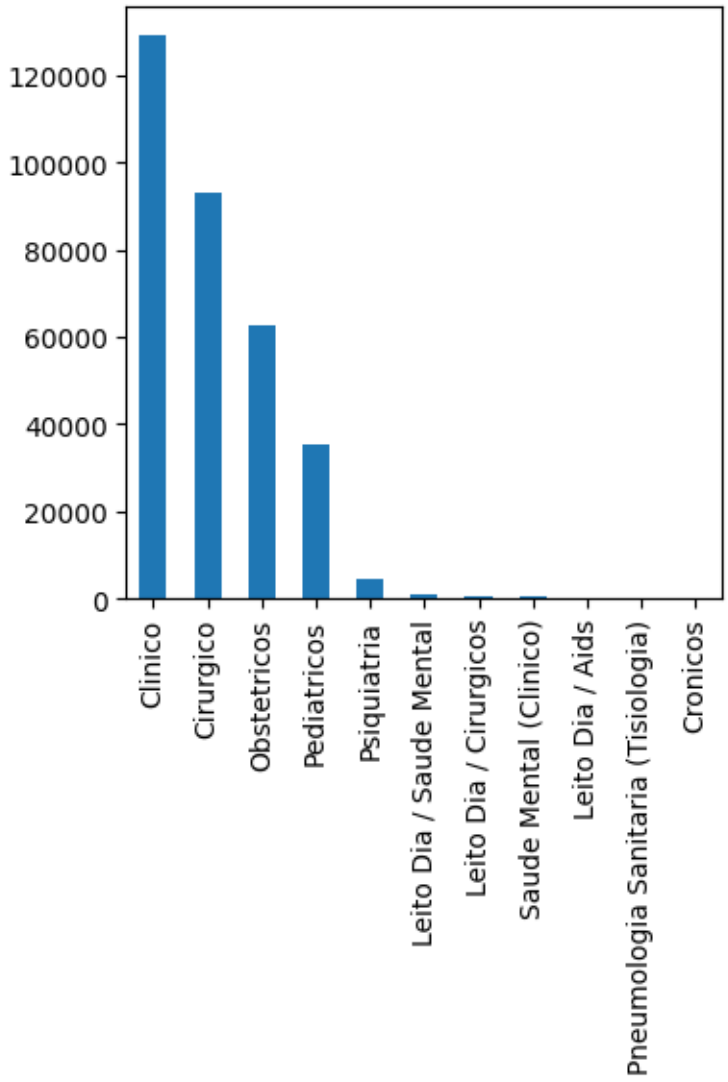
```
3  220000      2022         1      2           NaN  2222101081519      1
4  220000      2022         1      2           NaN  2222101081574      1

        CEP  MUNIC_RES      NASC  …  TPDISEC9  DataRefAquivo  \
0  64253000     220635  19930106  …       0.0           2201
1  64258000     220557  19500501  …       0.0           2201
2  64258000     220557  19740209  …       0.0           2201
3  64110000     220550  19860416  …       0.0           2201
4  64110000     220550  20040424  …       0.0           2201

   MUNIC_UFZI_NOME  IDHM_UFZI          MUNIC_RES_NOME  IDHM_RES  \
0              NaN        NaN          Milton Brandão     508.0
1              NaN        NaN  Lagoa de São Francisco     529.0
2              NaN        NaN  Lagoa de São Francisco     529.0
3              NaN        NaN         José de Freitas     618.0
4              NaN        NaN         José de Freitas     618.0

    MUNIC_MOV_NOME  IDHM_MOV   DESC_ESPEC  \
0         Pedro II       571      Clinico
1         Pedro II       571      Clinico
2         Pedro II       571      Clinico
3  José de Freitas       618  Obstetricos
4  José de Freitas       618  Obstetricos

                                  DESC_DIAG_PRINC
0      N39.0 Infecc do trato urinario de localiz NE
1  B95.5 Estreptococo NE causa doenc class outr cap
2  B95.5 Estreptococo NE causa doenc class outr cap
3                   O80.0 Parto espontaneo cefalico
4                   O80.0 Parto espontaneo cefalico

[5 rows x 122 columns]
```

[ ]:
```python
# Visualizando as especialidades de leitos mais frequentes
espec = df_sih["DESC_ESPEC"].value_counts()
print(espec)
espec.plot(kind="bar", figsize=(4, 4))
```

```
Clinico                             129390
Cirurgico                            93131
Obstetricos                          62915
Pediatricos                          35393
Psiquiatria                           4526
Leito Dia / Saude Mental               840
Leito Dia / Cirurgicos                 297
Saude Mental (Clinico)                 296
Leito Dia / Aids                        45
Pneumologia Sanitaria (Tisiologia)       8
```

```
Cronicos                                3
Name: DESC_ESPEC, dtype: int64
```

[ ]: <Axes: >

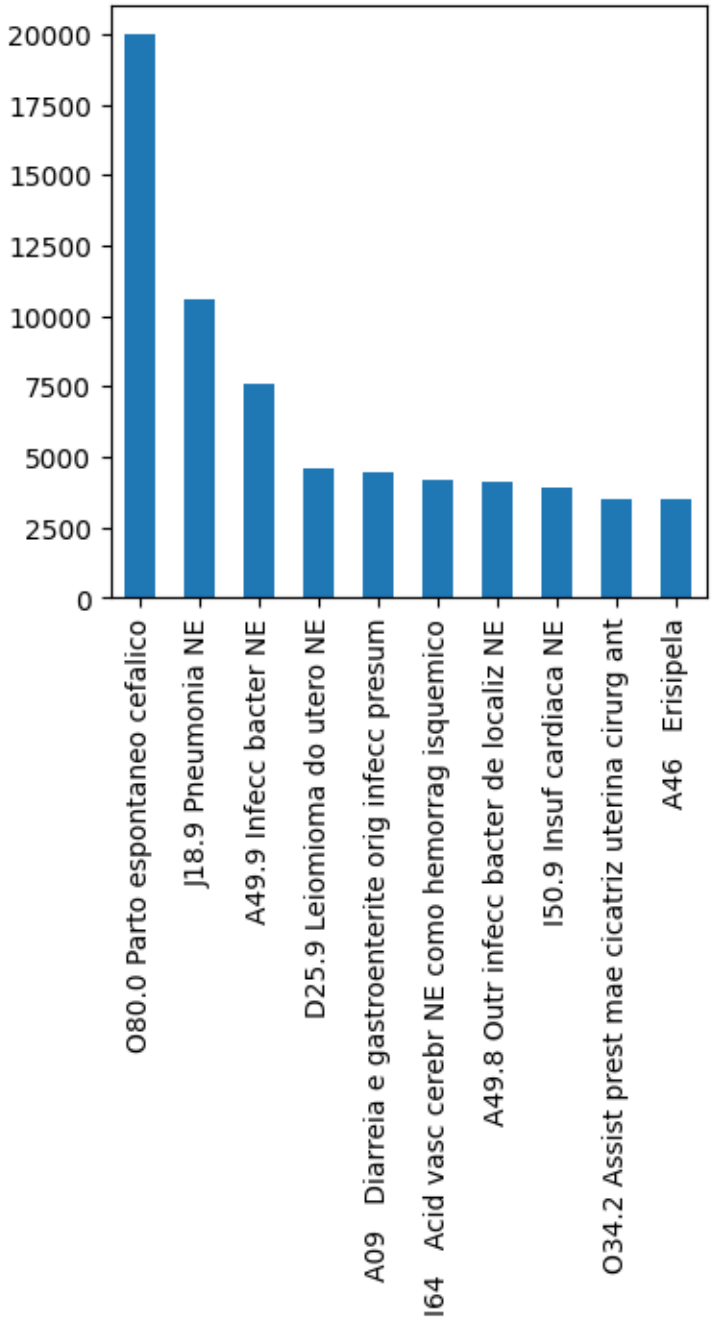


[ ]:
```
# Visualizando os diagnósticos principais mais frequentes
diag_princ = df_sih["DESC_DIAG_PRINC"].value_counts()[0:10]
print(diag_princ)
diag_princ.plot(kind="bar", figsize=(4, 4))
```

```
O80.0 Parto espontaneo cefalico          19988
J18.9 Pneumonia NE                       10608
A49.9 Infecc bacter NE                    7565
D25.9 Leiomioma do utero NE               4620
```

```
A09   Diarreia e gastroenterite orig infecc presum     4427
I64   Acid vasc cerebr NE como hemorrag isquemico      4196
A49.8 Outr infecc bacter de localiz NE                 4144
I50.9 Insuf cardiaca NE                                3923
O34.2 Assist prest mae cicatriz uterina cirurg ant     3510
A46   Erisipela                                        3503
Name: DESC_DIAG_PRINC, dtype: int64
```

[ ]: <Axes: >



3 3. Análise de Movimentação de Pacientes

3.1 3.1 Redução para apenas a variaveis de interesse

[ ]:
```
# Selecionando as colunas necessárias para o estudo
integer_columns = ["IDHM_RES", "IDHM_MOV", 'IDHM_UFZI']
category_columns = ["MUNIC_RES_NOME", "MUNIC_MOV_NOME", "DESC_DIAG_PRINC",␣
 ↪"DESC_ESPEC", "MUNIC_UFZI_NOME"]
all_target_columns = category_columns + integer_columns
df_rd = df_sih[all_target_columns]
```

[ ]:
```
# Convertendo as colunas adequadas para o tipo inteiro
for col in integer_columns:
    df_rd[col] = df_rd[col].astype("Int64")
```

[ ]:
```
# Visualizando o dataset
df_rd
```

[ ]:
```
                MUNIC_RES_NOME   MUNIC_MOV_NOME  \
0                Milton Brandão         Pedro II
1       Lagoa de São Francisco         Pedro II
2       Lagoa de São Francisco         Pedro II
3                José de Freitas  José de Freitas
4                José de Freitas  José de Freitas
…                            …               …
326839           Fora do Piauí         Teresina
326840                  Altos          Teresina
326841               Teresina          Teresina
326842               Teresina          Teresina
326843               Teresina          Teresina

                                        DESC_DIAG_PRINC   DESC_ESPEC  \
0           N39.0 Infecc do trato urinario de localiz NE      Clinico
1       B95.5 Estreptococo NE causa doenc class outr cap      Clinico
2       B95.5 Estreptococo NE causa doenc class outr cap      Clinico
3                       O80.0 Parto espontaneo cefalico  Obstetricos
4                       O80.0 Parto espontaneo cefalico  Obstetricos
…                                                    …          …
326839                   S82.2 Frat da diafise da tibia    Cirurgico
326840                          S02.6 Frat de mandibula    Cirurgico
326841                    S82.6 Frat do maleolo lateral    Cirurgico
326842         S52.5 Frat da extremidade distal do radio    Cirurgico
326843          S02.4 Frat dos ossos malares e maxilares    Cirurgico
```

```
       MUNIC_UFZI_NOME  IDHM_RES  IDHM_MOV  IDHM_UFZI
0                  NaN       508       571       <NA>
1                  NaN       529       571       <NA>
2                  NaN       529       571       <NA>
3                  NaN       618       618       <NA>
4                  NaN       618       618       <NA>
…                    …         …         …         …
326839        Teresina      <NA>       751        751
326840        Teresina       614       751        751
326841        Teresina       751       751        751
326842        Teresina       751       751        751
326843        Teresina       751       751        751

[326844 rows x 8 columns]
```

[ ]:
```
# Visualizando as informações do dataset
df_rd.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 326844 entries, 0 to 326843
Data columns (total 8 columns):
 #   Column           Non-Null Count   Dtype
---  ------           --------------   -----
 0   MUNIC_RES_NOME   326844 non-null  object
 1   MUNIC_MOV_NOME   326844 non-null  object
 2   DESC_DIAG_PRINC  326839 non-null  object
 3   DESC_ESPEC       326844 non-null  object
 4   MUNIC_UFZI_NOME  233439 non-null  object
 5   IDHM_RES         316392 non-null  Int64
 6   IDHM_MOV         326844 non-null  Int64
 7   IDHM_UFZI        233439 non-null  Int64
dtypes: Int64(3), object(5)
memory usage: 23.4+ MB
```

[ ]:
```
# Visualizando as estatísticas do dataset
df_rd.describe(include="all")
```

[ ]:
```
       MUNIC_RES_NOME MUNIC_MOV_NOME                  DESC_DIAG_PRINC  \
count          326844         326844                           326839
unique            225             63                             4845
top          Teresina       Teresina  O80.0 Parto espontaneo cefalico
freq            78815         140295                            19988
mean              NaN            NaN                              NaN
std               NaN            NaN                              NaN
min               NaN            NaN                              NaN
25%               NaN            NaN                              NaN
50%               NaN            NaN                              NaN
75%               NaN            NaN                              NaN
```

```
max                NaN            NaN                                  NaN

       DESC_ESPEC MUNIC_UFZI_NOME    IDHM_RES    IDHM_MOV   IDHM_UFZI
count      326844          233439    316392.0    326844.0    233439.0
unique         11              15        <NA>        <NA>        <NA>
top       Clinico         Teresina        <NA>        <NA>        <NA>
freq       129390          140295        <NA>        <NA>        <NA>
mean          NaN             NaN  638.306525  688.531758  718.783815
std           NaN             NaN   79.356441   66.016419   46.46781
min           NaN             NaN       485.0       497.0       524.0
25%           NaN             NaN       575.0       642.0       687.0
50%           NaN             NaN       619.0       698.0       751.0
75%           NaN             NaN       700.0       751.0       751.0
max           NaN             NaN       751.0       751.0       751.0
```

3.2 3.2 Fluxo de Pacientes

[ ]:
```python
# Definindo funções para contabilização
def most_frequent(x):
    return x.value_counts().index[0]

def list_top_cities_with_capital(df, city_column, top=11):
    return list(df[city_column].value_counts()[:top].index)
```

[ ]:
```python
# Definição de funções utilizadas para agrupamento e filtro
def group_info(df, origin_column, target_column):
    df.reset_index(inplace=True)
    df.rename(columns={"index": "Count"}, inplace=True)
    df = df.groupby([origin_column, target_column]).agg({"Count": "count",␣
 ↪"IDHM_RES": "last", "IDHM_MOV": "last", "DESC_DIAG_PRINC": most_frequent,␣
 ↪"DESC_ESPEC": most_frequent}).reset_index()
    df = df.rename(columns={"DESC_DIAG_PRINC": "DIAG_PRINC_FREQ", "DESC_ESPEC":␣
 ↪"ESPEC_FREQ"})
    return df.sort_values(by=['Count'], ascending=False).reset_index(drop=True)

def group_total_fluxo(df, city_column):
    if city_column == "MUNIC_MOV_NOME":
        idhm = "IDHM_MOV"
    elif city_column == "MUNIC_RES_NOME":
        idhm = "IDHM_RES"
    return df.groupby([city_column]).agg({"Count": "sum", idhm: "last",␣
 ↪"DIAG_PRINC_FREQ": most_frequent, "ESPEC_FREQ": most_frequent}).
 ↪sort_values(by=["Count"], ascending=False).reset_index()
```

[ ]:
```python
# Definição de funções utilizadas para filtragem
def filter_same_city(df, origin_column="MUNIC_MOV_NOME",␣
 ↪target_column="MUNIC_RES_NOME"):
```

```
    return df[df[origin_column] != df[target_column]].reset_index(drop=True)

def filter_outside_piaui(df, city_column):
    return df[df[city_column] != "Fora do Piauí"].reset_index(drop=True)

def filter_cities(df, city_column, list_cities):
    return df[df[city_column].isin(list_cities)].reset_index(drop=True)
```

[ ]:
```
# Definição de funções utilizadas para plotagem de gráficos
def plot_pacientes_vs_idhm(df_with_capital, df_without_capital=None,␣
 ↪coluna_municipio="MUNIC_RES_NOME"):
    if df_without_capital is None:
        fig, axes = plt.subplots(figsize=(20, 5))
        ax_left_1 = axes
    else:
        fig, axes = plt.subplots(1,2,figsize=(20, 5))
        ax_left_1 = axes[0]

    if coluna_municipio == "MUNIC_MOV_NOME":
        idhm = "IDHM_MOV"
    else:
        idhm = "IDHM_RES"

    ax_left_2 = ax_left_1.twinx()
    sns.barplot(x=coluna_municipio, y="Count", data=df_with_capital,␣
 ↪ax=ax_left_1, color="orange")
    sns.lineplot(x=coluna_municipio, y=idhm, data=df_with_capital,␣
 ↪ax=ax_left_2, color="blue")
    ax_left_1.set_xticklabels(ax_left_1.get_xticklabels(), rotation=90)
    ax_left_1.set_title("Fluxo de residencia de pacientes COM TERESINA")
    ax_left_1.set_xlabel("Município de residencia")
    ax_left_1.set_ylabel("Quantidade de pacientes")
    ax_left_2.set_ylabel("IDHM do município de residencia")
    # retirar a linha de contorno do gráfico e colocar linhas pontilhadas no␣
 ↪eixo y do gráfico
    ax_left_1.spines['top'].set_visible(False)
    ax_left_1.spines['right'].set_visible(False)
    ax_left_1.spines['left'].set_visible(False)
    ax_left_1.spines['bottom'].set_visible(False)
    ax_left_1.spines['bottom'].set_visible(False)
    ax_left_2.spines['top'].set_visible(False)
    ax_left_2.spines['right'].set_visible(False)
    ax_left_2.spines['left'].set_visible(False)
    # Colocar os valores do grafico de barras no topo das barras em cor laranja␣
 ↪negrito
    for p in ax_left_1.patches:
```

```
        ax_left_1.annotate('{:.0f}'.format(p.get_height()), (p.get_x()+0.2, p.
↪get_height()+100), color='orange', weight='bold')
    # Colocar os valores de cada ponto do grafico de linhas no topo de cada␣
↪ponto das linhas em cor azul negrito
    for line in ax_left_2.lines:
        for x,y in zip(line.get_xdata(), line.get_ydata()):
            ax_left_2.annotate('{:.0f}'.format(y), xy=(x,y), xytext=(x-0.
↪1,y+5), color='blue', weight='bold')

    if df_without_capital is None:
        return plt.show()
    else:
        ax_right_1 = axes[1]
        ax_right_2 = ax_right_1.twinx()
        sns.barplot(x=coluna_municipio, y="Count", data=df_without_capital,␣
↪ax=ax_right_1, color="orange")
        sns.lineplot(x=coluna_municipio, y=idhm, data=df_without_capital,␣
↪ax=ax_right_2, color="blue")
        ax_right_1.set_xticklabels(ax_right_1.get_xticklabels(), rotation=90)
        ax_right_1.set_title("Fluxo de residencia pacientes SEM TERESINA")
        ax_right_1.set_xlabel("Município de residencia")
        ax_right_1.set_ylabel("Quantidade de pacientes")
        ax_right_2.set_ylabel("IDHM do município de residencia")
        for p in ax_right_1.patches:
            ax_right_1.annotate('{:.0f}'.format(p.get_height()), (p.get_x()+0.
↪2, p.get_height()+100), color='orange', weight='bold')
        # Colocar os valores de cada ponto do grafico de linhas no topo de cada␣
↪ponto das linhas em cor azul negrito
        for line in ax_right_2.lines:
            for x,y in zip(line.get_xdata(), line.get_ydata()):
                ax_right_2.annotate('{:.0f}'.format(y), xy=(x,y), xytext=(x-0.
↪1,y+5), color='blue', weight='bold')
        return plt.show()

def plot_diag_princ_list_df(df, lista, coluna_municipio, coluna_analise):
    df_base = df.loc[df[coluna_municipio].isin(lista)].reset_index(drop=True)
    df_base = filter_same_city(df_base)
    df_base = filter_outside_piaui(df_base, coluna_municipio)
    n_plots = len(lista)
    n_cols = int(np.ceil(n_plots / 2))
    n_rows = int(n_plots/n_cols)

    fig, axes = plt.subplots(n_rows, n_cols, figsize=(20, n_plots*1),␣
↪gridspec_kw={'hspace':2.8})
    i = 0
    for municipio in lista:
```

```
        df = df_base.loc[df_base[coluna_municipio] == municipio]
        df = df.groupby([coluna_analise]).agg({coluna_municipio: "count"}).
↪reset_index()
        df.columns = [coluna_analise, "Count"]
        df = df.sort_values(by=['Count'], ascending=False).
↪reset_index(drop=True).head(10)

        row_index, col_index = divmod(i, n_cols)
        axis = axes[row_index, col_index]

        sns.barplot(x=df[coluna_analise], y=df["Count"], ax=axis,␣
↪color="orange")
        axis.set_xticklabels(axis.get_xticklabels(), rotation=85)
        axis.set_title(f'{municipio}')
        i = i + 1

    for i in range(n_plots, n_rows * n_cols):
        fig.delaxes(axes.flatten()[i])

    plt.show()
```

[ ]:
```
# Visualização do fluxo bruto de pacientes por município de residência e␣
↪município de atendimento
df_fluxo = group_info(df_rd, "MUNIC_RES_NOME", "MUNIC_MOV_NOME")
df_fluxo
```

[ ]:
```
              MUNIC_RES_NOME       MUNIC_MOV_NOME  Count  IDHM_RES  IDHM_MOV  \
0                   Teresina             Teresina  78583       751       751
1                   Parnaíba             Parnaíba  18351       687       687
2       São Raimundo Nonato  São Raimundo Nonato   9667       661       661
3                      Picos                Picos   4929       698       698
4             Fora do Piauí             Teresina   4827      <NA>       751
…                         …                    …    …       …         …
1422      Floresta do Piauí                Jaicós      1       538       524
1423  Santa Cruz dos Milagres          Campo Maior      1       577       656
1424                Floriano            Bom Jesus      1       700       668
1425                Floriano             Corrente      1       700       642
1426                   Acauã          Campo Maior      1       528       656

                                      DIAG_PRINC_FREQ   ESPEC_FREQ
0                     O80.0 Parto espontaneo cefalico      Clinico
1                     O80.0 Parto espontaneo cefalico    Cirurgico
2                              A49.9 Infecc bacter NE      Clinico
3     O75.9 Complic do trabalho de parto e do parto NE      Clinico
4                           T00.9 Traum superf mult NE    Cirurgico
…                                                   …           …
1422                           A49.9 Infecc bacter NE      Clinico
```

```
1423  K40.9 Hernia inguinal unilat NE s/obstr gangrena    Cirurgico
1424          J40   Bronquite NE como aguda ou cronica      Clinico
1425                    O80.0 Parto espontaneo cefalico  Obstetricos
1426                         M75.8 Outr lesoes do ombro    Cirurgico

[1427 rows x 7 columns]
```

[ ]:
```
# Filtrando os pacientes que residem e são atendidos no mesmo município
df_fluxo_without_same_city = filter_same_city(df_fluxo, "MUNIC_RES_NOME",␣
 ↪"MUNIC_MOV_NOME")
df_fluxo_without_same_city
```

[ ]:
```
               MUNIC_RES_NOME MUNIC_MOV_NOME  Count  IDHM_RES  IDHM_MOV  \
0                Fora do Piauí       Teresina   4827      <NA>       751
1                Fora do Piauí       Parnaíba   3017      <NA>       687
2                       União       Teresina   2938       577       751
3              José de Freitas       Teresina   2754       618       751
4                        Altos       Teresina   2742       614       751
…                        …             …    …       …       …
1359        Floresta do Piauí          Jaicós      1       538       524
1360  Santa Cruz dos Milagres    Campo Maior      1       577       656
1361                 Floriano      Bom Jesus      1       700       668
1362                 Floriano       Corrente      1       700       642
1363                    Acauã    Campo Maior      1       528       656

                                      DIAG_PRINC_FREQ   ESPEC_FREQ
0                          T00.9 Traum superf mult NE    Cirurgico
1                       O80.0 Parto espontaneo cefalico  Obstetricos
2                       O80.0 Parto espontaneo cefalico    Cirurgico
3                       O80.0 Parto espontaneo cefalico    Cirurgico
4                       O80.0 Parto espontaneo cefalico    Cirurgico
…                                                  …         …
1359                           A49.9 Infecc bacter NE      Clinico
1360  K40.9 Hernia inguinal unilat NE s/obstr gangrena    Cirurgico
1361          J40   Bronquite NE como aguda ou cronica      Clinico
1362                    O80.0 Parto espontaneo cefalico  Obstetricos
1363                         M75.8 Outr lesoes do ombro    Cirurgico

[1364 rows x 7 columns]
```

[ ]:
```
# Filtrando pacientes que residem fora do Piauí
df_fluxo_without_same_city_and_outside =␣
 ↪filter_outside_piaui(df_fluxo_without_same_city, "MUNIC_RES_NOME")
df_fluxo_without_same_city_and_outside
```

[ ]:
```
               MUNIC_RES_NOME MUNIC_MOV_NOME  Count  IDHM_RES  IDHM_MOV  \
0                       União       Teresina   2938       577       751
```

```
1             José de Freitas       Teresina   2754       618       751
2                       Altos       Teresina   2742       614       751
3                  Luís Correia       Parnaíba   1899       541       687
4                  Miguel Alves       Teresina   1755       539       751
…                           …           …    …       …       …
1326         Floresta do Piauí         Jaicós      1       538       524
1327  Santa Cruz dos Milagres    Campo Maior      1       577       656
1328                 Floriano      Bom Jesus      1       700       668
1329                 Floriano       Corrente      1       700       642
1330                    Acauã    Campo Maior      1       528       656

                                      DIAG_PRINC_FREQ   ESPEC_FREQ
0                       O80.0 Parto espontaneo cefalico    Cirurgico
1                       O80.0 Parto espontaneo cefalico    Cirurgico
2                       O80.0 Parto espontaneo cefalico    Cirurgico
3                       O80.0 Parto espontaneo cefalico  Obstetricos
4                       O80.0 Parto espontaneo cefalico    Cirurgico
…                                                   …           …
1326                            A49.9 Infecc bacter NE      Clinico
1327  K40.9 Hernia inguinal unilat NE s/obstr gangrena    Cirurgico
1328          J40   Bronquite NE como aguda ou cronica      Clinico
1329                    O80.0 Parto espontaneo cefalico  Obstetricos
1330                         M75.8 Outr lesoes do ombro    Cirurgico

[1331 rows x 7 columns]
```

3.2.1 3.2.1 Municipios de Residencia

[ ]:
```
# Agrupando o fluxo de pacientes por município de residência
df_res_total = group_total_fluxo(df_fluxo_without_same_city_and_outside,␣
 ↪"MUNIC_RES_NOME")
```

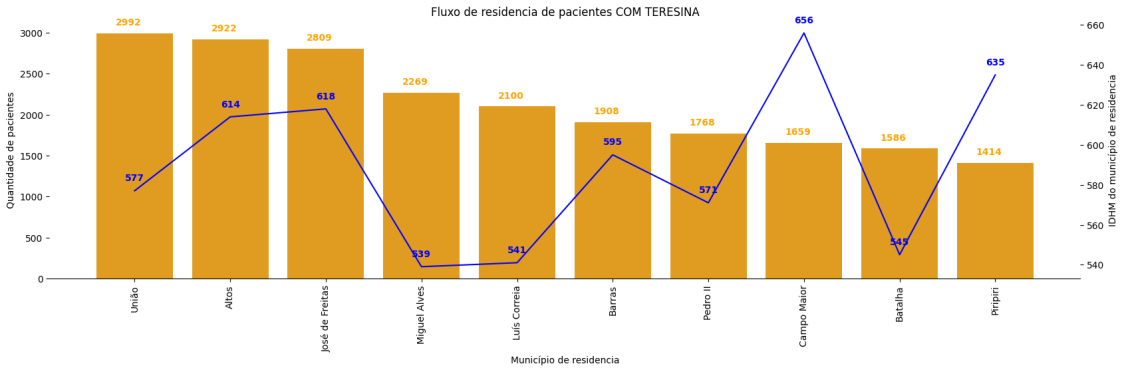
[ ]:
```
# Selecionando os 10 municípios com maior envio de pacientes
top10_df_res_total = df_res_total.head(10)
top10_df_res_total
```

[ ]:
```
    MUNIC_RES_NOME  Count  IDHM_RES  \
0            União   2992       577
1            Altos   2922       614
2  José de Freitas   2809       618
3     Miguel Alves   2269       539
4     Luís Correia   2100       541
5           Barras   1908       595
6        Pedro II   1768       571
7      Campo Maior   1659       656
8          Batalha   1586       545
9        Piripiri   1414       635
```

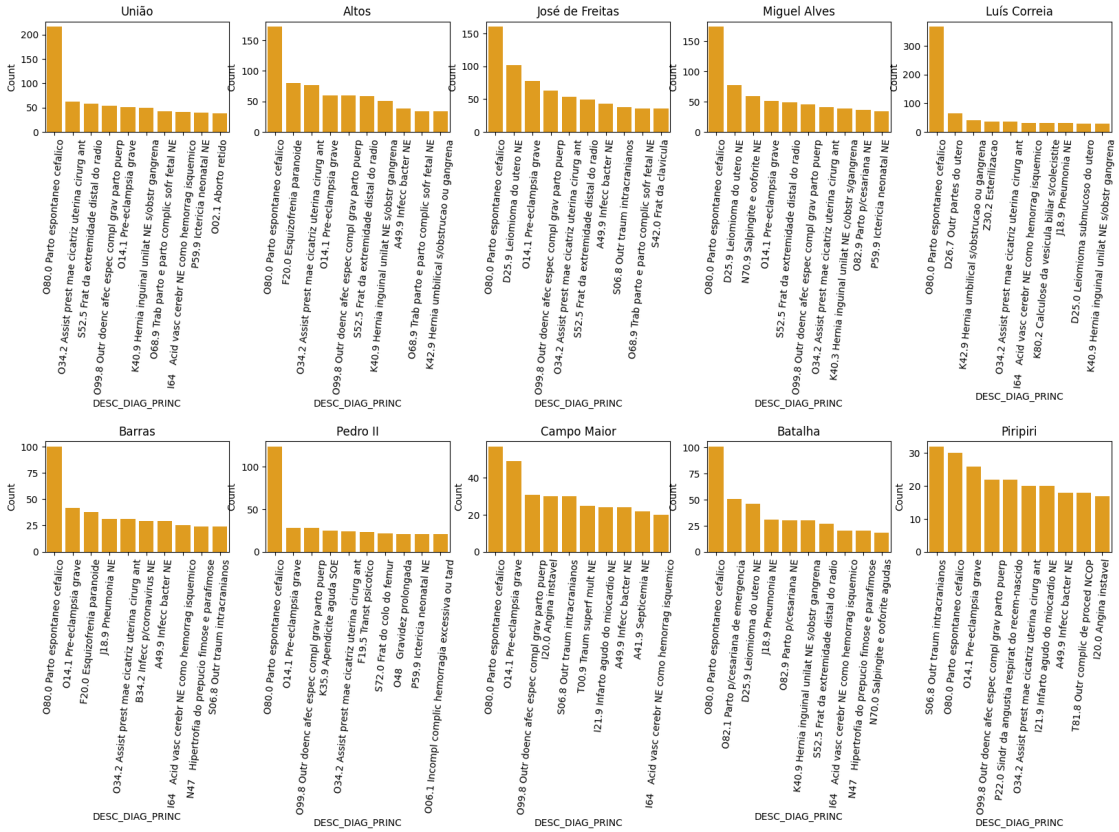
	DIAG_PRINC_FREQ	ESPEC_FREQ
0	D25.9 Leiomioma do utero NE	Cirurgico
1	O80.0 Parto espontaneo cefalico	Cirurgico
2	O80.0 Parto espontaneo cefalico	Cirurgico
3	O80.0 Parto espontaneo cefalico	Cirurgico
4	O80.0 Parto espontaneo cefalico	Obstetricos
5	O80.0 Parto espontaneo cefalico	Clinico
6	O80.0 Parto espontaneo cefalico	Cirurgico
7	O80.0 Parto espontaneo cefalico	Cirurgico
8	O80.0 Parto espontaneo cefalico	Cirurgico
9	P22.0 Sindr da angustia respirat do recem-nascido	Cirurgico

[ ]:
```
# Criando lista com os 10 municipios que mais enviam pacientes
list_top10_res = top10_df_res_total["MUNIC_RES_NOME"].to_list()
```

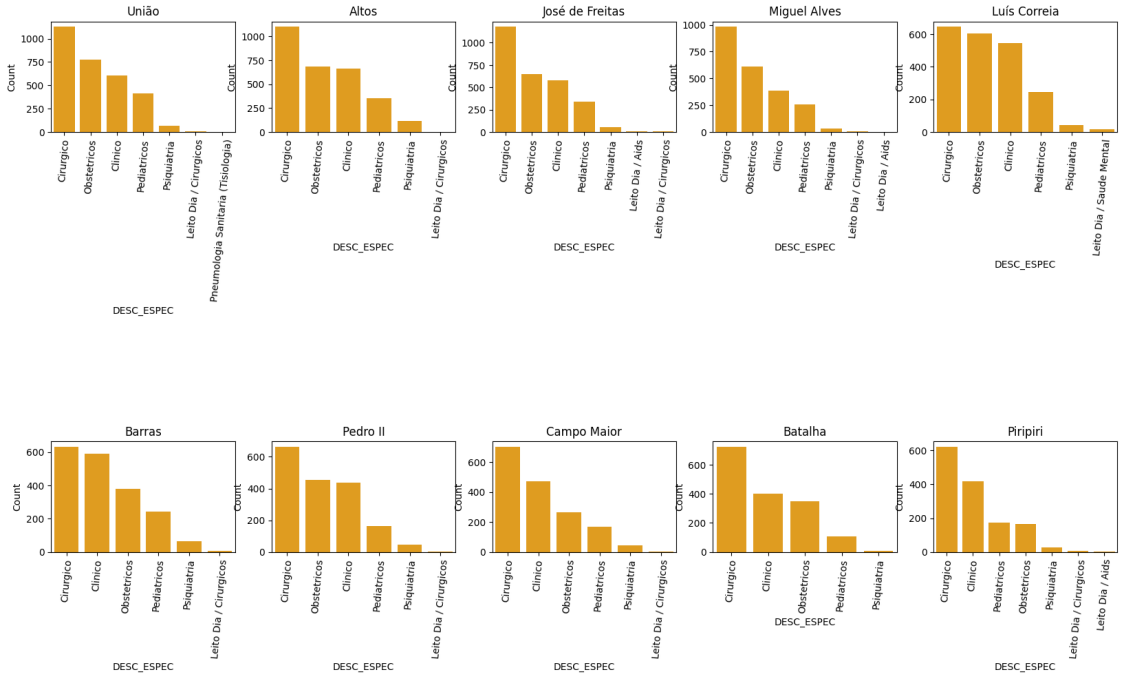
[ ]:
```
# Visualização da quantidade de pacientes enviados em relação ao IDHM do␣
 ↪municipio
plot_pacientes_vs_idhm(top10_df_res_total)
```



[ ]:
```
# Visualização das principais CIDs de diagnostico dos pacientes em relação ao␣
 ↪municipio de residencia que o enviou
plot_diag_princ_list_df(df_rd, list_top10_res, "MUNIC_RES_NOME",␣
 ↪"DESC_DIAG_PRINC")
```



[ ]:
```
# Visualização das especialidade de leito dos pacientes em relação ao municipio de residencia que o enviou
plot_diag_princ_list_df(df_rd, list_top10_res, "MUNIC_RES_NOME", "DESC_ESPEC")
```



3.2.2 3.2.2 Municipios de Atendimento

[ ]:
```
# Agrupando o fluxo de pacientes por município de atendimento
df_mov_total = group_total_fluxo(df_fluxo_without_same_city_and_outside,␣
 ↪"MUNIC_MOV_NOME")
```

[ ]:
```
# Selecionando os 10 municípios com maior recebimento de pacientes
top10_df_mov_total = df_mov_total.head(10)
top10_df_mov_total
```

[ ]:

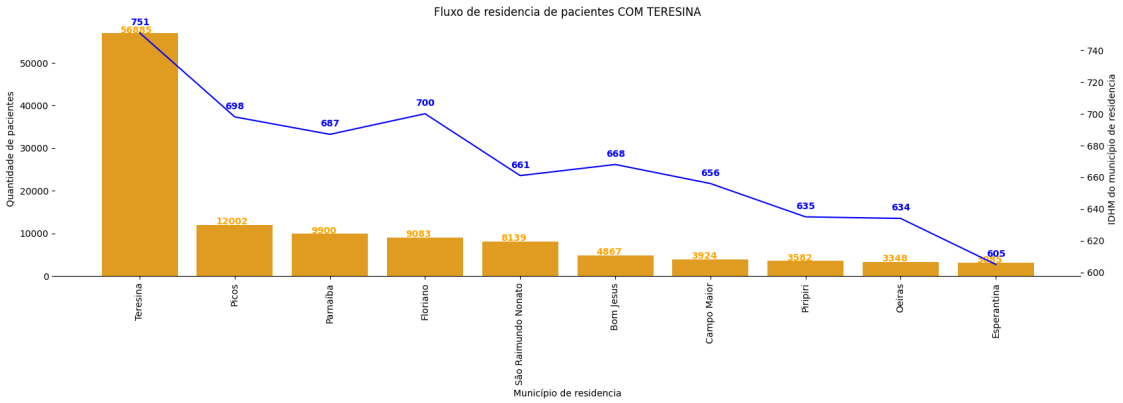
	MUNIC_MOV_NOME	Count	IDHM_MOV	\
0	Teresina	56885	751	
1	Picos	12002	698	
2	Parnaíba	9900	687	
3	Floriano	9083	700	
4	São Raimundo Nonato	8139	661	
5	Bom Jesus	4867	668	
6	Campo Maior	3924	656	
7	Piripiri	3582	635	
8	Oeiras	3348	634	
9	Esperantina	3085	605	

	DIAG_PRINC_FREQ	ESPEC_FREQ
0	O80.0 Parto espontaneo cefalico	Cirurgico
1	O75.9 Complic do trabalho de parto e do parto NE	Clinico

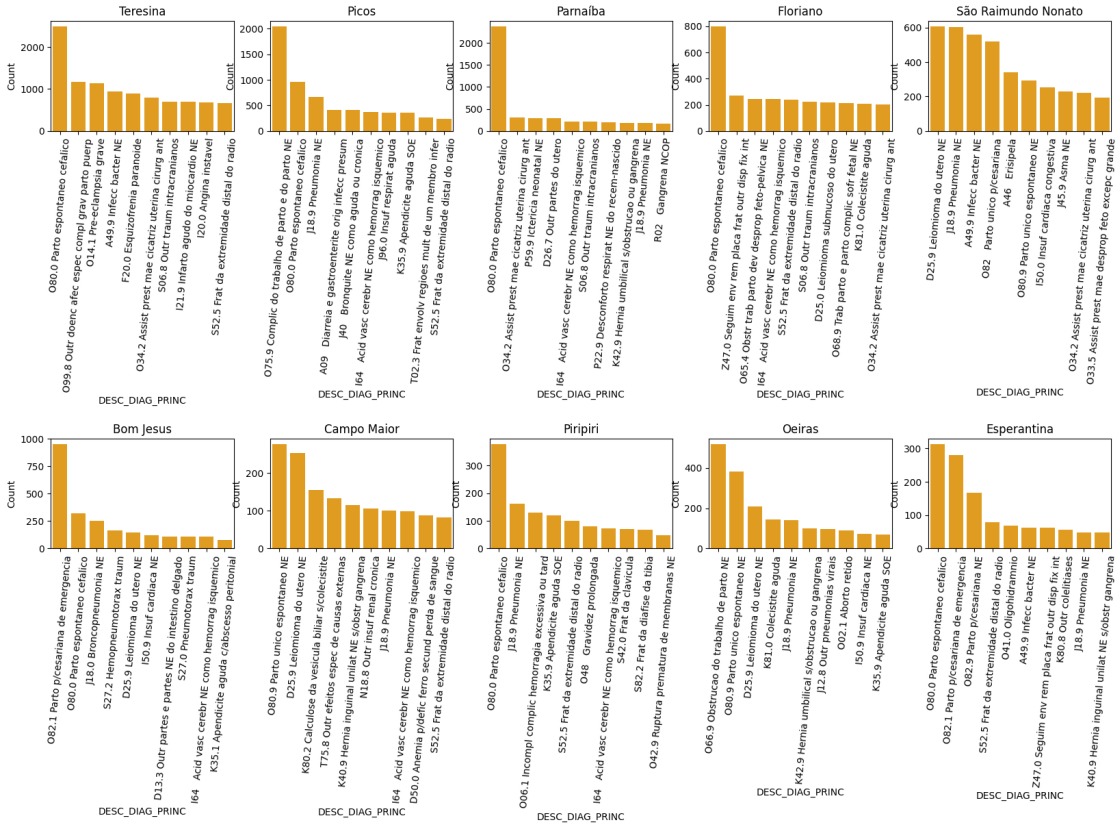
```
2                  O80.0 Parto espontaneo cefalico    Cirurgico
3                  O80.0 Parto espontaneo cefalico    Cirurgico
4                      D25.9 Leiomioma do utero NE      Clinico
5            O82.1 Parto p/cesariana de emergencia      Clinico
6                      D25.9 Leiomioma do utero NE    Cirurgico
7                  O80.0 Parto espontaneo cefalico  Obstetricos
8          O66.9 Obstrucao do trabalho de parto NE      Clinico
9                  O80.0 Parto espontaneo cefalico    Cirurgico
```

[ ]:
```
# Criando lista com os 10 municipios que mais recebem pacientes
list_top10_mov = top10_df_mov_total["MUNIC_MOV_NOME"].to_list()
```

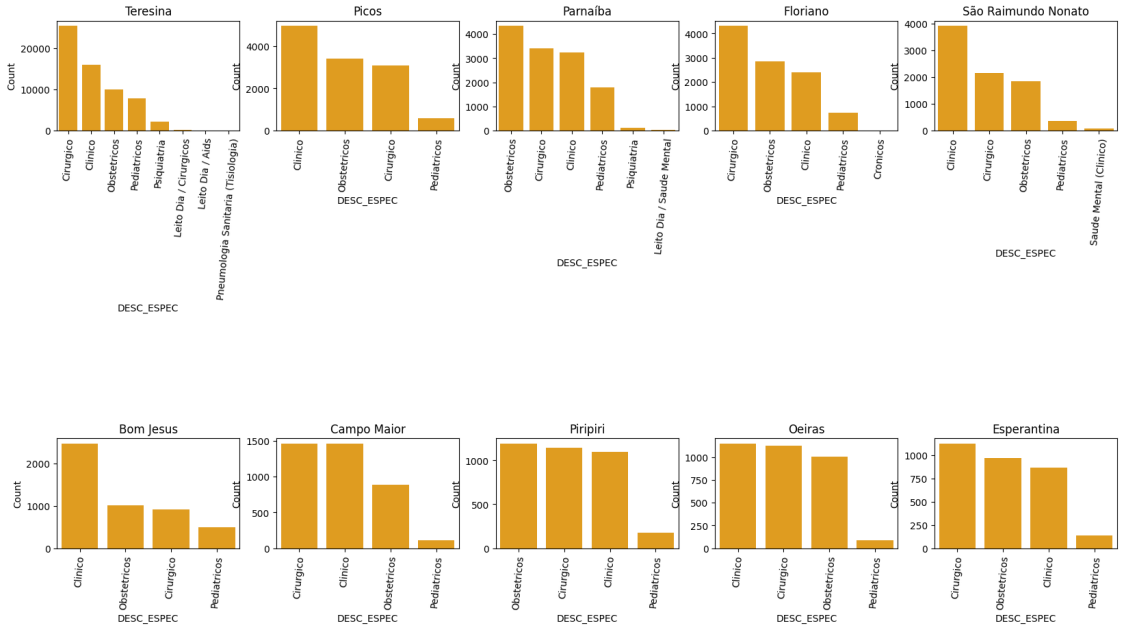
[ ]:
```
# Visualização da quantidade de pacientes enviados em relação ao IDHM do␣
↪municipio
plot_pacientes_vs_idhm(top10_df_mov_total, coluna_municipio="MUNIC_MOV_NOME")
```



[ ]:
```
# Visualização das principais CIDs de diagnostico dos pacientes em relação ao␣
↪municipio de atendeminto
plot_diag_princ_list_df(df_rd, list_top10_mov, "MUNIC_MOV_NOME",␣
↪"DESC_DIAG_PRINC")
```



[ ]:
```
# Visualização das especialidade de leito dos pacientes em relação ao municipio de atendimento
plot_diag_princ_list_df(df_rd, list_top10_mov, "MUNIC_MOV_NOME", "DESC_ESPEC")
```



3.2.3 3.2.3 Visualização do Fluxo de Pacientes

[ ]:
```
# Definindo função para plotagem de diagrama de sankey
def sankey_plot(df):
    df["MUNIC_RES_NOME"] = df["MUNIC_RES_NOME"].apply(lambda x: "origem: " +␣
 ↪str(x))
    df["MUNIC_MOV_NOME"] = df["MUNIC_MOV_NOME"].apply(lambda x: "destino: " +␣
 ↪str(x))

    unique_nodes = pd.unique(df[['MUNIC_RES_NOME', 'MUNIC_MOV_NOME']].values.
 ↪ravel('K'))

    # Create the graph
    fig = go.Figure(go.Sankey(
        arrangement='perpendicular',
        node=dict(
            pad=10,
            thickness=50,
            line=dict(color="black", width=0.5),
            label=unique_nodes,
        ),
        link=dict(
            source=pd.Categorical(df['MUNIC_RES_NOME'],␣
 ↪categories=unique_nodes).codes,
            target=pd.Categorical(df['MUNIC_MOV_NOME'],␣
 ↪categories=unique_nodes).codes,
            value=df['Count'],
        )
    ))

    fig.update_layout(
        width=1500,  # Largura desejada
        height=1500  # Altura desejada
    )

    # Show the graph
    fig.show()
```

3.2.3.1 Fluxo completo para 10 principais municipios de atendimento

[ ]:
```
# Filtrando dataset
df_to_sankey_mov = df_fluxo_without_same_city_and_outside.copy()
df_to_sankey_mov = df_to_sankey_mov.loc[df_to_sankey_mov['MUNIC_MOV_NOME'].
 ↪isin(list_top10_mov)].reset_index(drop=True)
```

[ ]:
```
# Visualizando fluxo
sankey_plot(df_to_sankey_mov)
```

[ ]:
```
# filtrando dataset para municipios que enviaram ao menos 400 pacientes
df_to_sankey_mov_filtered = df_to_sankey_mov.loc[df_to_sankey_mov["Count"] >=␣
 ↪400].sort_values(by="Count", ascending=False)
```

[ ]:
```
# Visualizando fluxo
sankey_plot(df_to_sankey_mov_filtered)
```

3.2.3.2 Fluxo dos 10 principais municipios de residencia

[ ]:
```
# Filtrando dataset
df_to_sankey_res = df_fluxo_without_same_city_and_outside.copy()
df_to_sankey_res = df_to_sankey_res.loc[df_to_sankey_res['MUNIC_RES_NOME'].
 ↪isin(list_top10_res)].reset_index(drop=True)
```

[ ]:
```
# Visualizando fluxo
sankey_plot(df_to_sankey_res)
```

3.2.3.3 Fluxo entre os 10 principais municipios de residencia e os 10 principais municipios de atendimento

[ ]:
```
# Filtrando dataset
df_to_sankey_comp = df_fluxo_without_same_city_and_outside.copy()
df_to_sankey_comp = df_to_sankey_comp.loc[df_to_sankey_comp['MUNIC_RES_NOME'].
 ↪isin(list_top10_res) & df_to_sankey_comp['MUNIC_MOV_NOME'].
 ↪isin(list_top10_mov)].reset_index(drop=True)
```

[ ]:
```
# Visualizando fluxo
sankey_plot(df_to_sankey_comp)
```

4 4. Exportação das principais tabelas

4.1 4.1 Tabela Reduzida

[ ]:
```
# Exportando dataset final
df_rd.to_excel("sih_rd_pi.xlsx", index=False)
```

4.2 4.2 Tabela de Fluxo Filtrada

[ ]:
```
# Exportando tabela
df_fluxo_without_same_city_and_outside.to_excel("fluxo_sih_rd_pi.xlsx",␣
 ↪index=False)
```

4.3 4.3 Tabela dos Principais Municipios de Residencia

[ ]:
```
# Exportando tabela
top10_df_res_total.to_excel("top10_df_res_total.xlsx", index=False)
```

4.4 4.4 Tabela dos Principais Municipios de Atendimento

[ ]:
```
# Exportando tabela
top10_df_mov_total.to_excel("top10_df_mov_total.xlsx", index=False)
```