JPNB1_Fornecedores

October 17, 2023

INTRODUÇÃO À DATA ANALYTICS PARA PESQUISA EM CONTABILIDADE

Projeto: Análise de dados de contratos do Poder Executivo de Santa Catarina

JPNB 01 - ETL DE DADOS DE FORNECEDORES

Elaborado por: Maurício Vasconcellos Leão Lyrio, Dr. | Página Oficial: www.vll.adm.br

[1]:
```
# Versão da linguagem Python e arquitetura do Jupyter Notebook
import platform
print('Versão da linguagem Python utilizada neste notebook:', platform.
  ↪python_version())
print('Arquitetura do Jupyter utilizada neste notebook:', platform.
  ↪architecture()[0])
```

```
Versão da linguagem Python utilizada neste notebook: 3.8.5
Arquitetura do Jupyter utilizada neste notebook: 64bit
```

---

1 Instalação das bibliotecas

[2]:
```
# Manipulação de dados
import pandas as pd

# Ignorar warnings
import warnings
warnings.filterwarnings('ignore')

# Versões dos pacotes utilizados neste Jupyter notebook
#!pip install -q -U watermark
%reload_ext watermark
%watermark -a "Mauricio Vasconcellos Leão Lyrio | vll.adm.br" --iversions
```

```
Author: Mauricio Vasconcellos Leão Lyrio | vll.adm.br

pandas  : 1.5.3
platform: 1.0.8
```

---

2 Carregamento dos datasets

Para iniciar nosso processo de análise de dados iremos **carregar o dataset com os dados cadastrais de fornecedores** recebido por meio de solicitação via LAI. Esse procedimento carrega o dataset do arquivo .csv original e o armazena em um dataframe pandas denominado ***df***, para que possamos manipulá-lo posteriormente.

[3]:
```
# Carregando o dataset de fornecedores
df = pd.read_csv('datasets/ELIC_fornecedores_cadastro.csv')
```

---

3 Análise exploratória dos dados

Após carregar o dataset damos início ao processo de análise exploratório, buscando **analisar a qualidade e integridade dos dados**. O processo de análise exploratório nos ajuda a ter uma visão geral do dataset e que tipo de pré-processamento precisaremos realizar nos dados a fim de deixá-los prontos para as etapas posteriores.

[4]:
```
# Verificando se o dataset foi carregado corretamente e seu tipo
type(df)
```

[4]:
```
pandas.core.frame.DataFrame
```

[5]:
```
# Verificando o formato do dataframe
df.shape
```

[5]:
```
(358232, 8)
```

[6]:
```
# Listando as colunas do dataframe
df.columns
```

[6]:
```
Index(['cnpj', 'razao_social', 'porte', 'cidade', 'uf', 'pais', 'situacao',
       'produtos_habilitados'],
      dtype='object')
```

[7]:
```
# Listando as informações gerais do dataframe
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 358232 entries, 0 to 358231
```

```
Data columns (total 8 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   cnpj                  358232 non-null  int64
 1   razao_social          358232 non-null  object
 2   porte                 358232 non-null  object
 3   cidade                358101 non-null  object
 4   uf                    358101 non-null  object
 5   pais                  358101 non-null  object
 6   situacao              358232 non-null  object
 7   produtos_habilitados  357273 non-null  object
dtypes: int64(1), object(7)
memory usage: 21.9+ MB
```

Até o momento conseguimos carregar os dados, verificar o tamanho do dataframe, suas colunas, o tipo de dado de cada coluna e se existem valores ausentes. Pelo info do dataframe é possível perceber que existem campos com valores nulos. Precisaremos definir o que fazer com esses campos e que tipo de tratamento iremos dar para os valores nulos, voltaremos a discutir essa questão na fase do pré-processamento de dados.

Pelo info do dataframe também é possível perceber que quase todos os campos são não-numéricos. O único campo numérico é o ***cnpj***, que, na verdade, não é uma variável quantitativa e sim um código que representa o cadastro de pessoa jurídica do fornecedor. Posteriormente iremos ajustar o tipo de dado desse campo, por hora iremos somente descrever os demais campos de nosso dataset e visualizar uma amostra dos dados.

[8]:
```
# Descrevendo os dados não-numéricos
df.describe(include=object)
```

[8]:

	razao_social	porte	cidade	uf	pais	situacao	produtos_habilitados
count	358232	358232	358101	358101	358101	358232	357273
unique	20121	3	1181	29	3	5	859
top	HAKA COMERCIAL DO BRASIL LTDA	ME/EPP	FLORIANOPOLIS	Santa Catarina	Brasil	Empresa ativa	1301 - Equipamentos, programas e suprimentos d…
freq	851	236367	31761	173193	358082	256961	2697

[9]:
```
df.head()
```

[9]:
```
             cnpj                                    razao_social  \
0   6942591000100                          MCG AGUIAR CARTUCHOS ME
1   5325332000160  Global Multimídia Comércio de Eletroeletrônico…
2   9349162000104            TEXAS INFORMATICA E PRODUTOS LTDA. EPP
3  13000035000172                         TARCIANE LOHN BOECHAT EPP
4  73977480000119                       COMERCIAL STORINNY LTDA EPP

           porte                      cidade             uf    pais  \
0         ME/EPP    SAO JOSE DO RIO PRETO       São Paulo  Brasil
1         ME/EPP                 SAO PAULO       São Paulo  Brasil
2  NÃO INFORMADO                   VITORIA  Espírito Santo  Brasil
3         ME/EPP  SANTO AMARO DA IMPERATRIZ  Santa Catarina  Brasil
4         ME/EPP                PORTO BELO  Santa Catarina  Brasil

        situacao                                produtos_habilitados
0  Empresa ativa  1303 - Equipamentos, programas e suprimentos d…
1  Empresa ativa  1303 - Equipamentos, programas e suprimentos d…
2  Empresa ativa  1303 - Equipamentos, programas e suprimentos d…
3  Empresa ativa  1303 - Equipamentos, programas e suprimentos d…
4  Empresa ativa  1303 - Equipamentos, programas e suprimentos d…
```

Com a visualização de uma amostra do dataset finalizamos a análise exploratória. Outros tipos de análise poderiam ser feitos nessa fase, porém, para nosso objetivo de preparar o dataset o que vimos até agora é suficiente. Passemos então à próxima fase do processo, o pré-processamento dos dados.

---

4 Pré-processamento

Na fase de análise exploratória identificamos que nosso dataset possui campos nulos e também que um dos campos foi definido de forma equivocada como numérico. Vamos agora tratar esses problemas e também analisar a necessidade de outros tipos de transformação de dados. Comecemos com a limpeza dos dados.

4.1 Limpeza

Conforme visto anteriormente, nosso dataset possui uma série de campos com valores nulos. Vamos analisar melhor essa situação e definir o que fazer com esses valores. para isso criaremos uma nova **tabela com a distribuição percentual de valores nulos por coluna**.

[10]:
```
# Criando uma lista vazia para armazenar as informações de nome e tipo de␣
 ↪coluna.
colunas_info = []

# Iterando pelas colunas do dataframe
for coluna in df.columns:
    coluna_nome = coluna
```

```
    coluna_tipo = df[coluna].dtype
    coluna_nulos = df[coluna].isnull().sum()
    coluna_nulos_perc = (coluna_nulos/len(df))*100
    colunas_info.
 ↪append((coluna_nome,coluna_tipo,coluna_nulos,coluna_nulos_perc))

# Criando um novo dataframe e exibindo as informações das colunas
df_colunas_info = pd.DataFrame(colunas_info, columns=['Coluna','Tipo','Q Nulo',␣
 ↪'% Nulo'])
print(df_colunas_info)
```

```
                Coluna    Tipo  Q Nulo    % Nulo
0                 cnpj   int64       0  0.000000
1         razao_social  object       0  0.000000
2                porte  object       0  0.000000
3               cidade  object     131  0.036568
4                   uf  object     131  0.036568
5                 pais  object     131  0.036568
6            situacao  object       0  0.000000
7  produtos_habilitados  object     959  0.267704
```

Com a nova tabela fica mais fácil evidenciar os valores ausentes do dataframe. No caso, as colunas ***cidade***, ***uf***, ***pais*** e ***produtos_habilitados*** apresentam valores ausentes. Em projetos de data-science, em geral, utiliza-se como regra para tratamento de valores ausentes as seguintes opções:

- Para valores ausentes >= 50%, descartamos a variável;
- Para valores ausentes < 50%, tratar os valores ausentes;
- Para valores ausentes < 2%, descartar os valores ausentes.

Apesar dessa regra geral, é importante analisar o dataframe e verificar a forma mais adequada para tratamento dos valores ausentes e, principalmente, justificar as escolhas feitas no decorrer do tratamento dos dados. Em nosso caso, **como os valores ausentes são menos de 2% dos valores dos campos vamos excluí-los**. Porém, vale salientar que, ao excluir os registros cujos campos estão ausentes perdemos parte da informação no dataset, vale refletir sobre a relevância da perda dessa informação.

Nesse caso, como os registros no dataset estão com granularidade definida em nível de produtos habilitados, acredita-se que os dados básicos dos fornecedores não serão perdidos devido à exclusão desses registros.

[11]:
```
# Criando uma cópia do dataframe original e excluindo os registros com valores␣
 ↪nulos.
df1 = df.dropna()
```

[12]:
```
df1.isnull().sum()
```

[12]:
```
cnpj                  0
razao_social          0
porte                 0
```

```
cidade                  0
uf                      0
pais                    0
situacao                0
produtos_habilitados    0
dtype: int64
```

Temos agora um novo dataframe ***df1*** com os registros com valore ausentes excluídos. Mantivemos em memória o dataset original para o caso de queremos retornar à essa versão em outro momento. Passemos agora à tranformação dos dados.

4.2 Transformação

4.2.1 Ajustando a coluna de CNPJ

Iniciaremos nossa fase de transformação de dados ajustando o tipo de dado referente à coluna ***cnpj*** do fornecedor. Conforme verificamos anteriormente, essa ***coluna foi definida como numérica***, do tipo int64, porém, por se tratar de um código de referência de cada fornecedor, ela na verdade é categórica. Além disso, ***o registro do CNPJ não está no padrão normal***, faltando os separadores. Vamos ajustar esses valores agora.

[13]:
```
'''
Alterando o tipo de dado da coluna CNPJ, normalizando os registros,
criando uma máscara para os valores e atualizando o dataframe
'''
for i, cnpj in enumerate(df1['cnpj']):
    # Convertendo para string antes de verificar o comprimento
    cnpj_str = str(cnpj)

    # Se o registro tiver menos de 14 caracteres
    if len(cnpj_str) < 14:
        # Incluir '0' no início até que o registro tenha 14 caracteres
        df1.at[i, 'cnpj'] = '0' * (14 - len(cnpj_str)) + cnpj_str

# Transformar o padrão do registro para padrão de CNPJ
df1['cnpj'] = df1['cnpj'].astype(str).apply(lambda x: f'{x[:2]}.{x[2:5]}.{x[5:
 ↪8]}/{x[8:12]}-{x[12:]}')
df1.head()
```

[13]:

	cnpj	razao_social	\
0	06.942.591/0001-00	MCG AGUIAR CARTUCHOS ME	
1	05.325.332/0001-60	Global Multimídia Comércio de Eletroeletrônico…	
2	09.349.162/0001-04	TEXAS INFORMATICA E PRODUTOS LTDA. EPP	
3	13.000.035/0001-72	TARCIANE LOHN BOECHAT EPP	
4	73.977.480/0001-19	COMERCIAL STORINNY LTDA EPP	

	porte	cidade	uf	pais	\
0	ME/EPP	SAO JOSE DO RIO PRETO	São Paulo	Brasil	

```
1         ME/EPP                   SAO PAULO       São Paulo  Brasil
2  NÃO INFORMADO                     VITORIA  Espírito Santo  Brasil
3         ME/EPP  SANTO AMARO DA IMPERATRIZ  Santa Catarina  Brasil
4         ME/EPP                 PORTO BELO  Santa Catarina  Brasil

        situacao                                produtos_habilitados
0  Empresa ativa  1303 - Equipamentos, programas e suprimentos d…
1  Empresa ativa  1303 - Equipamentos, programas e suprimentos d…
2  Empresa ativa  1303 - Equipamentos, programas e suprimentos d…
3  Empresa ativa  1303 - Equipamentos, programas e suprimentos d…
4  Empresa ativa  1303 - Equipamentos, programas e suprimentos d…
```

[14]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 357184 entries, 0 to 334110
Data columns (total 8 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   cnpj                  357184 non-null  object
 1   razao_social          357143 non-null  object
 2   porte                 357143 non-null  object
 3   cidade                357143 non-null  object
 4   uf                    357143 non-null  object
 5   pais                  357143 non-null  object
 6   situacao              357143 non-null  object
 7   produtos_habilitados  357143 non-null  object
dtypes: object(8)
memory usage: 32.6+ MB
```

4.2.2 Criando novas colunas para produtos habilitados

Conforme evidenciado acima, agora todos os campos de nosso dataset são campos categóricos e poderemos utilizar os métodos de string para a coluna. Dando sequência á transformação de dados, analisando a coluna ***produtos_habilitados*** percebemos que essa coluna apresenta a informação dos produtos concatenando duas informações, o grupo e a classe do produto.

Esse tipo de conhecimento é o que chamamos de conhecimento de negócio, ou seja, para perceber esses nuances nos dados o analista precisa conhecer pelo menos um pouco da área de negócio sobre a qual está trabalhando, dessa forma a possibilidade de ter insights úteis em relação ao tópico aumenta, aprimorando as possibilidades de análise.

Iremos separar a coluna de ***produtos_habilitados*** em duas colunas, a primeira para armazenar os grupos de produtos e a segunda para armazenar a classe dos produtos.

[17]:
```
# Setando a saída para mostrar todo o comprimento da linha
pd.set_option('display.max_colwidth', None)
# Imprimindo a amostra de dados da coluna 'produtos_habilitados'
df1['produtos_habilitados'].head()
```

[17]:
```
0    1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras
1    1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras
2    1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras
3    1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras
4    1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras
Name: produtos_habilitados, dtype: object
```

[19]:
```
# Criando as novas colunas e separando os registros da coluna original
df1[['Grupo','Classe','Descrição']] = df1['produtos_habilitados'].str.split(' -
 ↪',2,expand=True)
df1.head(2)
```

[19]:
```
                 cnpj                                          razao_social  \
0  06.942.591/0001-00                              MCG AGUIAR CARTUCHOS ME
1  05.325.332/0001-60  Global Multimídia Comércio de Eletroeletrônicos Ltda

    porte                 cidade         uf    pais       situacao  \
0  ME/EPP  SAO JOSE DO RIO PRETO  São Paulo  Brasil  Empresa ativa
1  ME/EPP              SAO PAULO  São Paulo  Brasil  Empresa ativa

        produtos_habilitados  \
0  1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras
1  1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras

  Grupo                                                Classe  \
0  1303  Equipamentos, programas e suprimentos de informática
1  1303  Equipamentos, programas e suprimentos de informática

                                  Descrição
0  Cartuchos de tinta e toner para impressoras
1  Cartuchos de tinta e toner para impressoras
```

[20]:
```
# Dividindo a coluna ***Grupo*** em duas para separar os códigos de grupo e de
 ↪classe.
df1['Grupo1'] = df1['Grupo'].str[:2]
df1['Grupo2'] = df1['Grupo'].str[2:]

# Renomeando a coluna de grupo/classe original
df1.rename(columns={'Grupo':'Código GC'},inplace=True)
```

```
# Criando as colunas de descrição de grupo e de classe
df1['Grupo_desc'] = df1['Grupo1']+' - '+df1['Classe']
df1['Classe_desc'] = df1['Grupo2']+' - '+df1['Descrição']
df1.head()
```

[20]:

```
                 cnpj                                        razao_social  \
0  06.942.591/0001-00                          MCG AGUIAR CARTUCHOS ME
1  05.325.332/0001-60  Global Multimídia Comércio de Eletroeletrônicos Ltda
2  09.349.162/0001-04          TEXAS INFORMATICA E PRODUTOS LTDA. EPP
3  13.000.035/0001-72                       TARCIANE LOHN BOECHAT EPP
4  73.977.480/0001-19                     COMERCIAL STORINNY LTDA EPP

           porte                     cidade             uf    pais  \
0         ME/EPP      SAO JOSE DO RIO PRETO       São Paulo  Brasil
1         ME/EPP                  SAO PAULO       São Paulo  Brasil
2  NÃO INFORMADO                    VITORIA  Espírito Santo  Brasil
3         ME/EPP  SANTO AMARO DA IMPERATRIZ  Santa Catarina  Brasil
4         ME/EPP                 PORTO BELO  Santa Catarina  Brasil

        situacao  \
0  Empresa ativa
1  Empresa ativa
2  Empresa ativa
3  Empresa ativa
4  Empresa ativa

        produtos_habilitados  \
0  1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras
1  1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras
2  1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras
3  1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras
4  1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras

  Código GC                                                Classe  \
0      1303  Equipamentos, programas e suprimentos de informática
1      1303  Equipamentos, programas e suprimentos de informática
2      1303  Equipamentos, programas e suprimentos de informática
3      1303  Equipamentos, programas e suprimentos de informática
4      1303  Equipamentos, programas e suprimentos de informática

                                  Descrição Grupo1 Grupo2  \
0  Cartuchos de tinta e toner para impressoras     13     03
```

```
1  Cartuchos de tinta e toner para impressoras     13     03
2  Cartuchos de tinta e toner para impressoras     13     03
3  Cartuchos de tinta e toner para impressoras     13     03
4  Cartuchos de tinta e toner para impressoras     13     03

                                                Grupo_desc  \
0  13 - Equipamentos, programas e suprimentos de informática
1  13 - Equipamentos, programas e suprimentos de informática
2  13 - Equipamentos, programas e suprimentos de informática
3  13 - Equipamentos, programas e suprimentos de informática
4  13 - Equipamentos, programas e suprimentos de informática

                                      Classe_desc
0  03 - Cartuchos de tinta e toner para impressoras
1  03 - Cartuchos de tinta e toner para impressoras
2  03 - Cartuchos de tinta e toner para impressoras
3  03 - Cartuchos de tinta e toner para impressoras
4  03 - Cartuchos de tinta e toner para impressoras
```

[21]:
```
# Excluindo as colunas que não serão necessárias
df1.drop(columns=['Classe','Descrição','Grupo1','Grupo2'],inplace=True)
df1.head()
```

[21]:
```
                cnpj                                        razao_social  \
0  06.942.591/0001-00                                MCG AGUIAR CARTUCHOS ME
1  05.325.332/0001-60  Global Multimídia Comércio de Eletroeletrônicos Ltda
2  09.349.162/0001-04                TEXAS INFORMATICA E PRODUTOS LTDA. EPP
3  13.000.035/0001-72                              TARCIANE LOHN BOECHAT EPP
4  73.977.480/0001-19                            COMERCIAL STORINNY LTDA EPP

           porte                     cidade              uf    pais  \
0         ME/EPP      SAO JOSE DO RIO PRETO       São Paulo  Brasil
1         ME/EPP                   SAO PAULO       São Paulo  Brasil
2  NÃO INFORMADO                     VITORIA  Espírito Santo  Brasil
3         ME/EPP  SANTO AMARO DA IMPERATRIZ  Santa Catarina  Brasil
4         ME/EPP                 PORTO BELO  Santa Catarina  Brasil

        situacao  \
0  Empresa ativa
1  Empresa ativa
2  Empresa ativa
3  Empresa ativa
4  Empresa ativa

        produtos_habilitados  \
0  1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras
```

```
1  1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras
2  1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras
3  1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras
4  1303 - Equipamentos, programas e suprimentos de informática - Cartuchos de
tinta e toner para impressoras

  Código GC                                                Grupo_desc  \
0      1303  13 - Equipamentos, programas e suprimentos de informática
1      1303  13 - Equipamentos, programas e suprimentos de informática
2      1303  13 - Equipamentos, programas e suprimentos de informática
3      1303  13 - Equipamentos, programas e suprimentos de informática
4      1303  13 - Equipamentos, programas e suprimentos de informática

                                      Classe_desc
0  03 - Cartuchos de tinta e toner para impressoras
1  03 - Cartuchos de tinta e toner para impressoras
2  03 - Cartuchos de tinta e toner para impressoras
3  03 - Cartuchos de tinta e toner para impressoras
4  03 - Cartuchos de tinta e toner para impressoras
```

[22]:
```
# Ajustando os nomes das novas colunas
df1.rename(columns={'Grupo_desc':'Grupo','Classe_desc':'Classe'},inplace=True)
```

[23]:
```
df1.isnull().sum()
```

[23]:
```
cnpj                      0
razao_social             41
porte                    41
cidade                   41
uf                       41
pais                     41
situacao                 41
produtos_habilitados     41
Código GC                41
Grupo                    41
Classe                12522
dtype: int64
```

Ao gerar o info do novo dataset percebemos que ao transformar o dataset original acabamos por gerar uma nova coluna que contém valores nulos. Vamos agora analisar a situação e decidir o que fazer com os dados inconsistentes.

[24]:
```
# Comparando a descrição dos dataframes
describe_df = df.describe(include=object)
describe_df1 = df1.describe(include=object)
```

```
# Adicionando os nomes dos dataframes como índices
describe_df.index=['df'] * len(describe_df)
describe_df1.index=['df1'] * len(describe_df)

# Concatenando os resultados em um único dataframe
df_comparacao = pd.concat([describe_df,describe_df1])

# Listando a tabela resultante
df_comparacao
```

[24]:

```
                                                razao_social   porte  \
df                                                    358232  358232
df                                                     20121       3
df                                HAKA COMERCIAL DO BRASIL LTDA  ME/EPP
df                                                       851  236367
df1                                                   357143  357143
df1                                                    19194       3
df1  BAGATINI & GUANDALINI EQUIPAMENTOS TOPOGRÁFICOS LTDA  ME/EPP
df1                                                      851  235879

            cidade              uf    pais       situacao  \
df          358101          358101  358101         358232
df            1181              29       3              5
df   FLORIANOPOLIS  Santa Catarina  Brasil  Empresa ativa
df           31761          173193  358082         256961
df1         357143          357143  357143         357143
df1           1162              29       3              5
df1  FLORIANOPOLIS  Santa Catarina  Brasil  Empresa ativa
df1          31605          172572  357125         256814

produtos_habilitados  \
df
357273
df
859
df   1301 - Equipamentos, programas e suprimentos de informática - Equipamentos
de informática
df
2697
df1
357143
df1
859
df1  1301 - Equipamentos, programas e suprimentos de informática - Equipamentos
de informática
df1
```

```
2697

                    cnpj Código GC  \
df                   NaN       NaN
df                   NaN       NaN
df                   NaN       NaN
df                   NaN       NaN
df1               357184    357143
df1                25140       859
df1   00.004.774/0999-70      1301
df1                  847      2697

                                                Grupo  \
df                                                NaN
df                                                NaN
df                                                NaN
df                                                NaN
df1                                            357143
df1                                               143
df1   66 - Materiais de uso em enfermaria e cirurgia
df1                                             28279

                               Classe
df                                NaN
df                                NaN
df                                NaN
df                                NaN
df1                            344662
df1                               784
df1   01 - Equipamentos de informática
df1                              2697
```

Comparando a descrição do ***df1*** com a descrição do ***df*** percebemos que no processo de limpeza e transformação de dados foram perdidos registros de fornecedores (o df original tinha 20.121 cnpjs únicos e o novo df1 possui 19.194). Conforme dito anteriormente, a decisão de manter ou não determinadas informações do dataset é do analista, porém, é importante ter em conta as justificativas para as escolhas feitas no decorrer do processo. No caso, optaremos por manter o dataframe com os registros de *Classe* nulos, dado que já havíamos perdido informações de 927 fornecedores na primeira etapa do processo de limpeza.

Porém, para que o dataframe se torne mais "amigável", vamos substituir os campos nos quais existem valores nulos por um valor categórico para fins de uso na etapa de análise.

[25]:
```
# Lista das colunas que precisam ter valores nulos preenchidos
colunas_para_preencher = ['razao_social', 'porte', 'cidade', 'uf', 'pais',␣
 ↪'situacao', 'produtos_habilitados', 'Código GC', 'Grupo', 'Classe']

# Preenchendo valores nulos nas colunas selecionadas
```

```
df1[colunas_para_preencher] = df1[colunas_para_preencher].fillna('Não definido')

# Verificando o resultado
df1.isnull().sum()
```

[25]:
```
cnpj                    0
razao_social            0
porte                   0
cidade                  0
uf                      0
pais                    0
situacao                0
produtos_habilitados    0
Código GC               0
Grupo                   0
Classe                  0
dtype: int64
```

Com a finalização do processo de tranformação de dados nosso dataframe está pronto para ser carregado em banco de dados ou exportado em formatos de arquivos para análise posterior. É o que iremos fazer agora, na fazer de geração de dados de saída.

---

5 Geração de dados de saída (Data output)

Uma vez finalizada a fase de limpeza e transformação, agora iremos dar saída ao dataset gerado para fins de análise. Faremos isso em forma de arquivos e em registros em banco de dados.

5.1 Gravação em arquivos

[26]:
```
# Gravando em formato .csv
df1.to_csv('datasets/df1_fornecedores.csv',index=False)
```

[27]:
```
# Gravando em formato .json
df1.to_json('datasets/df1_fornecedores.json')
```

[ ]:
```
# Gravando em formato .xls
#df1.to_excel('datasets/df1_fornecedores.xlsx',index=False)
```

5.2 Gravação em DB relacional

Gravando em banco de dados SQLite

[28]:
```
# Importando a biblioteca para interação com o SQLite
import sqlite3

# Criando uma conexão ao banco de dados SQLite
```

```
cnn=sqlite3.connect('database/bdContratosSC.db')

# Copiando nosso dataframe para o banco de dados
df1.to_sql('Fornecedores',cnn)
```

[28]: 357184

Abaixo o código para gravação em banco de dados ***MySQL***, não executaremos esse script porque necessitamos ter o SGBD instalado.

[ ]:
```
# Importando a biblioteca para interação com o MySQL
#import mysql.connector

# Configurando os parâmetros da conexão
#config={
#    'user':'seu usuário';
#    'password':'sua_senha';
#    'host':'localhost';    # ou endereço do servidor MySQL
#    'database':'seu_banco_de_dados'
#}

# Criando uma conexão ao banco de dados
#try:
#    conn=mysql.connector.connect(**config)
#    if conn.is_connected():
#        print('Conexão ao banco de dados bem sucedida')
#except mysql.connector.Error as err:
#    print(f'Erro ao conectar ao banco de dados: {err}')

# Copiando nosso dataframe para o banco de dados

# Fechando a conexão (ao terminar de utilizar)
#conn.close()
```

5.3 Gravação em datalake (BD não-relacional)

Abaixo o código para gravação em banco de dados ***MongoDB***, não executaremos esse script porque necessitamos ter SGBD instalado.

[ ]:
```
# Importando a biblioteca para interação com o MongoDB
#from pymongo import MongoClient

# Configurando os parâmetros de conexão
#client=MongoClient('mongodb://localhost:27017/')

# Acessando um banco de dados específico
#db=client['appdb']
```

```
# Criando uma coleção no banco de dados
#collection = db['Fornecedores']

# Carregando o dataframe
#data = df1.to_dict(orient='records')

# Inserindo os registros no MongoDB
#collection.insert_many(data)

# Listando as coleções disponíveis
#print(db.list_collection_names())
```

[ ]:
```
#fornecedores=collection.find()
#for fornecedor in fornecedores:
#    print(fornecedor)
```

[ ]:
```
# Fechando a conexão ao MongoDB
#client.close()
```

Com a geração dos dados de saída, nosso trabalho de ***ETL*** de dados terminou. Fizemos a limpeza, transformação e carga de dados para arquivos de saíde e bancos de dados relacionais e não-relacionais. Agora passaremos à etapa de visualização de dados (dataviz) que faremos utilizando o Microsoft Power BI como ferramenta.

FIM