COVID - 19

Edgard Joseph Kiriyama

Definições

COVID - 19: "A Covid-19 é uma infecção respiratória aguda causada pelo coronavírus SARS-CoV-2, potencialmente grave, de elevada transmissibilidade e de distribuição global" (Ministério da Saúde, 2021).

PNAD (Pesquisa Nacional por Amostra de Domicílios) - COVID 19

"Objetiva estimar o número de pessoas com sintomas referidos associados à síndrome gripal e monitorar os impactos da pandemia da COVID-19 no mercado de trabalho brasileiro" (IBGE.2020).

A coleta da Pesquisa Nacional por Amostra de Domicílios - PNAD COVID19 teve início em 4 de maio de 2020, com entrevistas realizadas por telefone em, aproximadamente, 48 mil domicílios por semana, totalizando cerca de 193 mil domicílios por mês, em todo o Território Nacional. A amostra é fixa, ou seja, os domicílios entrevistados no primeiro mês de coleta de dados permanecerão na amostra nos meses subsequentes, até o fim da pesquisa.

O questionário se divide em duas partes, sendo uma direcionada a questões de saúde, especificamente sobre sintomas associados à síndrome gripal e outra, a questões de trabalho. Nas questões de saúde, investiga-se a ocorrência de alguns dos principais sintomas da COVID19 no período de referência da pesquisa, considerando-se todos os moradores do domicílio.

Critérios:

Parte 1 - Identificação e Controle

1 - A unidade territorial (UF).

2 - Mês da pesquisa para este estudo serão Julho, Agosto, Setembro, Outubro e Novembro de 2020 (V1013).

3 - Situação do domicílio - Urbana e Rural (V1022)

Parte A - Característica gerais dos Moradores

4 - Idade do Morador (A002)

5 - Sexo (A003)

Parte B - COVID19 - Todos os Moradores

Sintomas:

6 - Na semana passada teve febre? * (B0011)

7 - Na semana passada teve tosse? * (B0012)

8 - Na semana passada teve dificuldade para respirar? * (B0014)

9 - Na semana passada teve perda de cheiro ou sabor? * (B00111)

10 - O(A) Sr(a) fez algum teste para saber se estava infectado(a) pelo coronavírus? * (B008)

Plano de Saúde

11 - Tem algum plano de saúde médico, seja particular, de empresa ou de órgão público (B007)

Parte C - Características de trabalho das pessoas acima de 14 anos.

Trabalho

12 - Há quanto tempo está afastado desse trabalho? (C005)

13 - No trabalho (único ou principal) que tinha nessa semana, era: (C007)

14 - Qual o principal motivo deste afastamento temporário? (C003)

Aplicação do Estudo

Importação da biblioteca PANDAS para leitura e manipulação dos dados

In [1]:
```
import pandas as pd
import numpy as np
```

Visualização dos dados no período Julho, Agosto, Setembro, Outubro e Novembro de 2020

In [2]:
```
jul2020 = pd.read_csv('dados/PNAD_COVID_072020.csv', sep = ',')
ago2020 = pd.read_csv('dados/PNAD_COVID_082020.csv', sep = ',')
set2020 = pd.read_csv('dados/PNAD_COVID_092020.csv', sep = ',')
out2020 = pd.read_csv('dados/PNAD_COVID_102020.csv', sep = ',')
nov2020 = pd.read_csv('dados/PNAD_COVID_112020.csv', sep = ',')
```

In [3]:
```
#Visualiza os 10 primeiros registros.
jul2020.head()
```

Out[3]:

	Ano	UF	CAPITAL	RM_RIDE	V1008	V1012	V1013	V1016	Estrato	UPA	...	F001	F0021	F0022	F002A1	F002A2	F002A3	F002A4	F(
0	2020	11	11.0	NaN	1	4	7	3	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
1	2020	11	11.0	NaN	1	4	7	3	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
2	2020	11	11.0	NaN	1	4	7	3	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
3	2020	11	11.0	NaN	1	4	7	3	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
4	2020	11	11.0	NaN	2	1	7	3	1110011	110015970	...	1	NaN	NaN	1	1	1	2	

5 rows × 145 columns

In [4]:
```
#Visualiza os 10 primeiros registros.
ago2020.head()
```

Out[4]:

	Ano	UF	CAPITAL	RM_RIDE	V1008	V1012	V1013	V1016	Estrato	UPA	...	F001	F0021	F0022	F002A1	F002A2	F002A3	F002A4	F(
0	2020	11	11.0	NaN	1	4	8	4	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
1	2020	11	11.0	NaN	1	4	8	4	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
2	2020	11	11.0	NaN	1	4	8	4	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
3	2020	11	11.0	NaN	1	4	8	4	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
4	2020	11	11.0	NaN	2	1	8	4	1110011	110015970	...	1	NaN	NaN	1	1	1	2	

5 rows × 145 columns

In [5]:
```
#Visualiza os 5 primeiros registros.
set2020.head()
```

Out[5]:

	Ano	UF	CAPITAL	RM_RIDE	V1008	V1012	V1013	V1016	Estrato	UPA	...	F001	F0021	F0022	F002A1	F002A2	F002A3	F002A4	F(
0	2020	11	11.0	NaN	1	4	9	5	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
1	2020	11	11.0	NaN	1	4	9	5	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
2	2020	11	11.0	NaN	1	4	9	5	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
3	2020	11	11.0	NaN	1	4	9	5	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
4	2020	11	11.0	NaN	2	1	9	5	1110011	110015970	...	1	NaN	NaN	1	1	1	2	

5 rows × 145 columns

In [6]:
```
#Visualiza os 5 primeiros registros.
out2020.head()
```

Out[6]:

	Ano	UF	CAPITAL	RM_RIDE	V1008	V1012	V1013	V1016	Estrato	UPA	...	F001	F0021	F0022	F002A1	F002A2	F002A3	F002A4	F(
0	2020	11	11.0	NaN	1	4	10	6	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
1	2020	11	11.0	NaN	1	4	10	6	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
2	2020	11	11.0	NaN	1	4	10	6	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
3	2020	11	11.0	NaN	1	4	10	6	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
4	2020	11	11.0	NaN	2	1	10	6	1110011	110015970	...	1	NaN	NaN	1	1	1	2	

5 rows × 145 columns

In [7]:
```
#Visualiza os 5 primeiros registros.
nov2020.head()
```

Out[7]:

	Ano	UF	CAPITAL	RM_RIDE	V1008	V1012	V1013	V1016	Estrato	UPA	...	F001	F0021	F0022	F002A1	F002A2	F002A3	F002A4	F(
0	2020	11	11.0	NaN	1	4	11	7	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
1	2020	11	11.0	NaN	1	4	11	7	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
2	2020	11	11.0	NaN	1	4	11	7	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
3	2020	11	11.0	NaN	1	4	11	7	1110011	110015970	...	1	NaN	NaN	1	1	1	2	
4	2020	11	11.0	NaN	2	1	11	7	1110011	110015970	...	1	NaN	NaN	1	1	1	2	

5 rows × 148 columns

Tratando e Modelando os dados que serão utilizados no estudo

--> Julho/2020

In [8]:
```
# Constroi o DataFrame com as colunas selecionadas para o mês de Julho.
df_jul2020 = pd.DataFrame(jul2020,columns = ['UF', 'V1013', 'V1022', 'A002', 'A003', 'B0011', 'B0012', 'B0014',
                                             'B00111','B007','B008', 'C005', 'C007', 'C003'])

# Renomear as colunas
df_jul2020 = df_jul2020.rename(columns = {'UF':'Estado',
                                          'V1013':'Mes',
                                          'V1022':'Area',
                                          'A002':'Idade_Morador',
                                          'A003':'Sexo',
                                          'B0011':'Febre',
                                          'B0012':'Tosse',
                                          'B0014':'Dific_respiratoria',
                                          'B00111':'Cheiro_sabor',
                                          'B007':'Plano_Saude',
                                          'B008':'Teste_Covid',
                                          'C005':'Tempo_afastamento',
                                          'C007':'Carac_trabalho',
                                          'C003':'Motivo_afastamento'})

# Modelagem e Tratamento dos dados

## Substitui os valores numéricos dos Estados para a legenda apropriada a leitura do dado.
df_jul2020['Estado'] = df_jul2020['Estado'].map({11:'Rondônia', 12:'Acre', 13:'Amazonas',
                                                 14:'Roraima', 15:'Pará',16:'Amapá',
                                                 17:'Tocantins', 21:'Maranhão', 22:'Piauí',
                                                 23: 'Ceará', 24:'Rio Grande do Norte', 25:'Paraíba',
                                                 26:'Pernambuco', 27: 'Alagoas', 28: 'Sergipe',
                                                 29: 'Bahia', 31:'Minas Gerais', 32: 'Espírito Santo',
                                                 33:'Rio de Janeiro',35:'São Paulo', 41: 'Paraná',
                                                 42: 'Santa Catarina', 43: 'Rio Grande do Sul', 50:'Mato Gr
                                                 51:'Mato Grosso',52:'Goiás', 53: 'Distrito Federal'}, na_a

## Substitui os valor numérico do Mês para a legenda apropriada a leitura do dado.
df_jul2020['Mes'] = df_jul2020['Mes'].map({7:'Julho'}, na_action=None)

## Substitui os valor da Área do município(Urbana ou Rural) para a legenda apropriada a leitura do dado.
df_jul2020['Area'] = df_jul2020['Area'].map({1:'Urbana', 2:'Rural'}, na_action=None)


## Substitui os valores numéricos do Sexo para a legenda apropriada a leitura do dado.
```

```
df_jul2020['Sexo'] = df_jul2020['Sexo'].map({1:'Homem', 2:'Mulher'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Febre para a legenda apropriada a leitura do dado.
df_jul2020['Febre'] = df_jul2020['Febre'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Tosse para a legenda apropriada a leitura do dado.
df_jul2020['Tosse'] = df_jul2020['Tosse'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Dificuldade respiratoria para a legenda apropriada a leitura
df_jul2020['Dific_respiratoria'] = df_jul2020['Dific_respiratoria'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_acti


## Substitui os valores numéricos sobre sintoma de perda de Cheiro ou sabor para a legenda apropriada a leitura
df_jul2020['Cheiro_sabor'] = df_jul2020['Cheiro_sabor'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre Plano de Saúde para a legenda apropriada a leitura do dado.
df_jul2020['Plano_Saude'] = df_jul2020['Plano_Saude'].map({1:'Sim', 2:'Não', 9:'Ignorado'}, na_action=None)

## Substitui os valores numéricos sobre Teste de Covid para a legenda apropriada a leitura do dado.
df_jul2020['Teste_Covid'] = df_jul2020['Teste_Covid'].map({1:'Sim', 2:'Não', 9:'Ignorado'}, na_action=None)


## Substitui os valores numéricos sobre Tempo de afastamento para a legenda apropriada a leitura do dado.
df_jul2020['Tempo_afastamento'] = df_jul2020['Tempo_afastamento'].map({1:'Menos de 1 mês',
                                                                       2:'De 1 mês a menos de 1 ano',
                                                                       3:'De 1 ano a menos de 2 anos',
                                                                       4:'2 anos ou mais',
                                                                      np.nan:'Não aplicável'}, na_action=None)

## Substitui os valores numéricos sobre Carac para a legenda apropriada a leitura do dado.
df_jul2020['Carac_trabalho'] = df_jul2020['Carac_trabalho'].map({1:'Trabalhador doméstico (empregado doméstico,
                                                                 2:'Militar do exercito, marinha ou aeronáutica',
                                                                 3:'Policial militar ou bombeiro mlilitar',
                                                                 4:'Empregado do setor privado',
                                                                 5:'Empregado do setor público (inclusive empresas de
                                                                 6:'Empregador',
                                                                 7:'Conta própria',
                                                                 8:'Trabalhador familiar não remunerado em ajuda a memb
                                                                 9:'Estava fora do mercado de trabalho (fazia apenas a
                                                                np.nan:'Não aplicável'}, na_action=None)

## Substitui os valores numéricos sobre Motivo do afastamento para a legenda apropriada a leitura do dado.
```

```
df_jul2020['Motivo_afastamento'] = df_jul2020['Motivo_afastamento'].map({1:'Estava em quarentena, isolamento, dis
                                                                        2:'Férias, folga ou jornada de trabalho
                                                                        3:'Licença maternidade ou paternidade',
                                                                        4:'Licença remunerada por motivo de saúd
                                                                        5:'Outro tipo de licença remunerada (est
                                                                        6:'Afastamento do próprio negócio/empres
                                                                        7:'Fatores ocasionais (mau tempo, paral
                                                                        8:'Outro motivo',
                                                                        np.nan:'Não aplicável'}, na_action=None

# Cria o DataFrame do mês
df_jul2020
```

Out[8]:

	Estado	Mes	Area	Idade_Morador	Sexo	Febre	Tosse	Dific_respiratoria	Cheiro_sabor	Plano_Saude	Teste_Covid	Tempo_afastame
0	Rondônia	Julho	Urbana	35	Homem	Não	Não	Não	Não	Sim	Não	Não aplica
1	Rondônia	Julho	Urbana	29	Mulher	Não	Não	Não	Não	Não	Não	De 1 mês a me de 1
2	Rondônia	Julho	Urbana	13	Homem	Não	Não	Não	Não	Sim	Não	Não aplica
3	Rondônia	Julho	Urbana	10	Homem	Não	Não	Não	Não	Sim	Não	Não aplica
4	Rondônia	Julho	Urbana	57	Mulher	Não	Não	Não	Não	Não	Não	Não aplica
...	...	...	...	...	...	...	...	...	...	...	...	
384161	Distrito Federal	Julho	Rural	45	Mulher	Não	Não	Não	Não	Não	Não	Não aplica
384162	Distrito Federal	Julho	Rural	22	Mulher	Não	Não	Não	Não	Não	Não	Não aplica
384163	Distrito Federal	Julho	Rural	16	Mulher	Não	Não	Não	Não	Não	Não	Não aplica
384164	Distrito Federal	Julho	Rural	83	Homem	Não	Não	Não	Não	Não	Não	Não aplica
384165	Distrito Federal	Julho	Rural	75	Mulher	Não	Não	Não	Não	Não	Não	Não aplica

384166 rows × 14 columns

--> Agosto / 2020

In [9]:
```
# Constroi o DataFrame com as colunas selecionadas para o mês de Julho.
df_ago2020 = pd.DataFrame(ago2020,columns = ['UF', 'V1013', 'V1022', 'A002', 'A003', 'B0011', 'B0012', 'B0014',
                                             'B00111','B007','B008', 'C005', 'C007', 'C003'])

# Renomear as colunas
df_ago2020 = df_ago2020.rename(columns = {'UF':'Estado',
                                          'V1013':'Mes',
                                          'V1022':'Area',
                                          'A002':'Idade_Morador',
                                          'A003':'Sexo',
                                          'B0011':'Febre',
                                          'B0012':'Tosse',
                                          'B0014':'Dific_respiratoria',
                                          'B00111':'Cheiro_sabor',
                                          'B007':'Plano_Saude',
                                          'B008':'Teste_Covid',
                                          'C005':'Tempo_afastamento',
                                          'C007':'Carac_trabalho',
                                          'C003':'Motivo_afastamento'})

# Modelagem e Tratamento dos dados

## Substitui os valores numéricos dos Estados para a legenda apropriada a leitura do dado.
df_ago2020['Estado'] = df_ago2020['Estado'].map({11:'Rondônia', 12:'Acre', 13:'Amazonas',
                                                 14:'Roraima', 15:'Pará',16:'Amapá',
                                                 17:'Tocantins', 21:'Maranhão', 22:'Piauí',
                                                 23: 'Ceará', 24:'Rio Grande do Norte', 25:'Paraíba',
                                                 26:'Pernambuco', 27: 'Alagoas', 28: 'Sergipe',
                                                 29: 'Bahia', 31:'Minas Gerais', 32: 'Espírito Santo',
                                                 33:'Rio de Janeiro',35:'São Paulo', 41: 'Paraná',
                                                 42: 'Santa Catarina', 43: 'Rio Grande do Sul', 50:'Mato
                                                 51:'Mato Grosso',52:'Goiás', 53: 'Distrito Federal'}, na_

## Substitui os valor numérico do Mês para a legenda apropriada a leitura do dado.
df_ago2020['Mes'] = df_ago2020['Mes'].map({8:'Agosto'}, na_action=None)

## Substitui os valor da Área do município(Urbana ou Rural) para a legenda apropriada a leitura do dado.
df_ago2020['Area'] = df_ago2020['Area'].map({1:'Urbana', 2:'Rural'}, na_action=None)


## Substitui os valores numéricos do Sexo para a legenda apropriada a leitura do dado.
```

```
df_ago2020['Sexo'] = df_ago2020['Sexo'].map({1:'Homem', 2:'Mulher'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Febre para a legenda apropriada a leitura do dado.
df_ago2020['Febre'] = df_ago2020['Febre'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Tosse para a legenda apropriada a leitura do dado.
df_ago2020['Tosse'] = df_ago2020['Tosse'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Dificuldade respiratoria para a legenda apropriada a leitura
df_ago2020['Dific_respiratoria'] = df_ago2020['Dific_respiratoria'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_act


## Substitui os valores numéricos sobre sintoma de perda de Cheiro ou sabor para a legenda apropriada a leitura
df_ago2020['Cheiro_sabor'] = df_ago2020['Cheiro_sabor'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre Plano de Saúde para a legenda apropriada a leitura do dado.
df_ago2020['Plano_Saude'] = df_ago2020['Plano_Saude'].map({1:'Sim', 2:'Não', 9:'Ignorado'}, na_action=None)

## Substitui os valores numéricos sobre Teste de Covid para a legenda apropriada a leitura do dado.
df_ago2020['Teste_Covid'] = df_ago2020['Teste_Covid'].map({1:'Sim', 2:'Não', 9:'Ignorado'}, na_action=None)


## Substitui os valores numéricos sobre Tempo de afastamento para a legenda apropriada a leitura do dado.
df_ago2020['Tempo_afastamento'] = df_ago2020['Tempo_afastamento'].map({1:'Menos de 1 mês',
                                                                       2:'De 1 mês a menos de 1 ano',
                                                                       3:'De 1 ano a menos de 2 anos',
                                                                       4:'2 anos ou mais',
                                                                      np.nan:'Não aplicável'}, na_action=None)

## Substitui os valores numéricos sobre Carac para a legenda apropriada a leitura do dado.
df_ago2020['Carac_trabalho'] = df_ago2020['Carac_trabalho'].map({1:'Trabalhador doméstico (empregado doméstico,
                                                                 2:'Militar do exercito, marinha ou aeronáutica',
                                                                 3:'Policial militar ou bombeiro mlilitar',
                                                                 4:'Empregado do setor privado',
                                                                 5:'Empregado do setor público (inclusive empresas de
                                                                 6:'Empregador',
                                                                 7:'Conta própria',
                                                                 8:'Trabalhador familiar não remunerado em ajuda a mer
                                                                 9:'Estava fora do mercado de trabalho (fazia apenas a
                                                                np.nan:'Não aplicável'}, na_action=None)

## Substitui os valores numéricos sobre Motivo do afastamento para a legenda apropriada a leitura do dado.
```

```
df_ago2020['Motivo_afastamento'] = df_ago2020['Motivo_afastamento'].map({1:'Estava em quarentena, isolamento, di
                                                                         2:'Férias, folga ou jornada de trabalho
                                                                         3:'Licença maternidade ou paternidade',
                                                                         4:'Licença remunerada por motivo de sau
                                                                         5:'Outro tipo de licença remunerada (es
                                                                         6:'Afastamento do próprio negócio/empre
                                                                         7:'Fatores ocasionais (mau tempo, paral
                                                                         8:'Outro motivo',
                                                                        np.nan:'Não aplicável'}, na_action=None

# Cria o DataFrame do mês
df_ago2020
```

Out[9]:

	Estado	Mes	Area	Idade_Morador	Sexo	Febre	Tosse	Dific_respiratoria	Cheiro_sabor	Plano_Saude	Teste_Covid	Tempo_afastam
0	Rondônia	Agosto	Urbana	36	Homem	Não	Não	Não	Não	Sim	Não	Não apli
1	Rondônia	Agosto	Urbana	30	Mulher	Não	Não	Não	Não	Não	Não	Não apli
2	Rondônia	Agosto	Urbana	13	Homem	Não	Não	Não	Não	Sim	Não	Não apli
3	Rondônia	Agosto	Urbana	11	Homem	Não	Não	Não	Não	Sim	Não	Não apli
4	Rondônia	Agosto	Urbana	57	Mulher	Não	Não	Não	Não	Não	Não	Não apli
...	...	...	...	...	...	...	...	...	...	...	...	
386515	Distrito Federal	Agosto	Rural	75	Mulher	Não	Não	Não	Não	Não	Não	Não apli
386516	Distrito Federal	Agosto	Rural	57	Mulher	Não	Não	Não	Não	Não	Sim	Não apli
386517	Distrito Federal	Agosto	Rural	52	Homem	Não	Não	Não	Não	Não	Não	Não apli
386518	Distrito Federal	Agosto	Rural	23	Homem	Não	Não	Não	Não	Não	Não	Não apli
386519	Distrito Federal	Agosto	Rural	5	Mulher	Não	Não	Não	Não	Não	Não	Não apli

386520 rows × 14 columns

--> Setembro / 2020

In [10]:
```
# Constroi o DataFrame com as colunas selecionadas para o mês de Julho.
df_set2020 = pd.DataFrame(set2020,columns = ['UF', 'V1013', 'V1022', 'A002', 'A003', 'B0011', 'B0012', 'B0014',
                                             'B00111','B007','B008', 'C005', 'C007', 'C003'])

# Renomear as colunas
df_set2020 = df_set2020.rename(columns = {'UF':'Estado',
                                          'V1013':'Mes',
                                          'V1022':'Area',
                                          'A002':'Idade_Morador',
                                          'A003':'Sexo',
                                          'B0011':'Febre',
                                          'B0012':'Tosse',
                                          'B0014':'Dific_respiratoria',
                                          'B00111':'Cheiro_sabor',
                                          'B007':'Plano_Saude',
                                          'B008':'Teste_Covid',
                                          'C005':'Tempo_afastamento',
                                          'C007':'Carac_trabalho',
                                          'C003':'Motivo_afastamento'})

# Modelagem e Tratamento dos dados

## Substitui os valores numéricos dos Estados para a legenda apropriada a leitura do dado.
df_set2020['Estado'] = df_set2020['Estado'].map({11:'Rondônia', 12:'Acre', 13:'Amazonas',
                                                 14:'Roraima', 15:'Pará',16:'Amapá',
                                                 17:'Tocantins', 21:'Maranhão', 22:'Piauí',
                                                 23: 'Ceará', 24:'Rio Grande do Norte', 25:'Paraíba',
                                                 26:'Pernambuco', 27: 'Alagoas', 28: 'Sergipe',
                                                 29: 'Bahia', 31:'Minas Gerais', 32: 'Espírito Santo',
                                                 33:'Rio de Janeiro',35:'São Paulo', 41: 'Paraná',
                                                 42: 'Santa Catarina', 43: 'Rio Grande do Sul', 50:'Mato
                                                 51:'Mato Grosso',52:'Goiás', 53: 'Distrito Federal'}, na_

## Substitui os valor numérico do Mês para a legenda apropriada a leitura do dado.
df_set2020['Mes'] = df_set2020['Mes'].map({9:'Setembro'}, na_action=None)

## Substitui os valor da Área do município(Urbana ou Rural) para a legenda apropriada a leitura do dado.
df_set2020['Area'] = df_set2020['Area'].map({1:'Urbana', 2:'Rural'}, na_action=None)


## Substitui os valores numéricos do Sexo para a legenda apropriada a leitura do dado.
```

```
df_set2020['Sexo'] = df_set2020['Sexo'].map({1:'Homem', 2:'Mulher'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Febre para a legenda apropriada a leitura do dado.
df_set2020['Febre'] = df_set2020['Febre'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Tosse para a legenda apropriada a leitura do dado.
df_set2020['Tosse'] = df_set2020['Tosse'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Dificuldade respiratoria para a legenda apropriada a leitura
df_set2020['Dific_respiratoria'] = df_set2020['Dific_respiratoria'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_act


## Substitui os valores numéricos sobre sintoma de perda de Cheiro ou sabor para a legenda apropriada a leitura
df_set2020['Cheiro_sabor'] = df_set2020['Cheiro_sabor'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre Plano de Saúde para a legenda apropriada a leitura do dado.
df_set2020['Plano_Saude'] = df_set2020['Plano_Saude'].map({1:'Sim', 2:'Não', 9:'Ignorado'}, na_action=None)

## Substitui os valores numéricos sobre Teste de Covid para a legenda apropriada a leitura do dado.
df_set2020['Teste_Covid'] = df_set2020['Teste_Covid'].map({1:'Sim', 2:'Não', 9:'Ignorado'}, na_action=None)


## Substitui os valores numéricos sobre Tempo de afastamento para a legenda apropriada a leitura do dado.
df_set2020['Tempo_afastamento'] = df_set2020['Tempo_afastamento'].map({1:'Menos de 1 mês',
                                                                       2:'De 1 mês a menos de 1 ano',
                                                                       3:'De 1 ano a menos de 2 anos',
                                                                       4:'2 anos ou mais',
                                                                      np.nan:'Não aplicável'}, na_action=None)

## Substitui os valores numéricos sobre Carac para a legenda apropriada a leitura do dado.
df_set2020['Carac_trabalho'] = df_set2020['Carac_trabalho'].map({1:'Trabalhador doméstico (empregado doméstico,
                                                                 2:'Militar do exercito, marinha ou aeronáutica',
                                                                 3:'Policial militar ou bombeiro mlilitar',
                                                                 4:'Empregado do setor privado',
                                                                 5:'Empregado do setor público (inclusive empresas de
                                                                 6:'Empregador',
                                                                 7:'Conta própria',
                                                                 8:'Trabalhador familiar não remunerado em ajuda a mer
                                                                 9:'Estava fora do mercado de trabalho (fazia apenas a
                                                                np.nan:'Não aplicável'}, na_action=None)

## Substitui os valores numéricos sobre Motivo do afastamento para a legenda apropriada a leitura do dado.
```

```
df_set2020['Motivo_afastamento'] = df_set2020['Motivo_afastamento'].map({1:'Estava em quarentena, isolamento, di
                                                                         2:'Férias, folga ou jornada de trabalho
                                                                         3:'Licença maternidade ou paternidade',
                                                                         4:'Licença remunerada por motivo de sau
                                                                         5:'Outro tipo de licença remunerada (es
                                                                         6:'Afastamento do próprio negócio/empre
                                                                         7:'Fatores ocasionais (mau tempo, paral
                                                                         8:'Outro motivo',
                                                                        np.nan:'Não aplicável'}, na_action=None

# Cria o DataFrame do mês
df_set2020
```

Out[10]:

	Estado	Mes	Area	Idade_Morador	Sexo	Febre	Tosse	Dific_respiratoria	Cheiro_sabor	Plano_Saude	Teste_Covid	Tempo_afast
0	Rondônia	Setembro	Urbana	36	Homem	Não	Não	Não	Não	Sim	Não	Não a
1	Rondônia	Setembro	Urbana	30	Mulher	Não	Não	Não	Não	Não	Não	Não a
2	Rondônia	Setembro	Urbana	13	Homem	Não	Não	Não	Não	Sim	Não	Não a
3	Rondônia	Setembro	Urbana	11	Homem	Não	Não	Não	Não	Sim	Não	Não a
4	Rondônia	Setembro	Urbana	57	Mulher	Não	Não	Não	Não	Não	Não	Não a
...	...	...	...	...	...	...	...	...	...	...	...	
387293	Distrito Federal	Setembro	Rural	45	Mulher	Sim	Sim	Não	Não	Não	Sim	Não a
387294	Distrito Federal	Setembro	Rural	22	Mulher	Sim	Sim	Não	Não	Não	Sim	Não a
387295	Distrito Federal	Setembro	Rural	16	Mulher	Sim	Sim	Não	Não	Não	Sim	Não a
387296	Distrito Federal	Setembro	Rural	83	Homem	Não	Não	Não	Não	Não	Não	Não a
387297	Distrito Federal	Setembro	Rural	75	Mulher	Não	Não	Não	Não	Não	Não	Não a

387298 rows × 14 columns

--> Outubro / 2020

In [11]:
```
# Constroi o DataFrame com as colunas selecionadas para o mês de Julho.
df_out2020 = pd.DataFrame(out2020,columns = ['UF', 'V1013', 'V1022', 'A002', 'A003', 'B0011', 'B0012', 'B0014',
                                             'B00111','B007','B008', 'C005', 'C007', 'C003'])

# Renomear as colunas
df_out2020 = df_out2020.rename(columns = {'UF':'Estado',
                                          'V1013':'Mes',
                                          'V1022':'Area',
                                          'A002':'Idade_Morador',
                                          'A003':'Sexo',
                                          'B0011':'Febre',
                                          'B0012':'Tosse',
                                          'B0014':'Dific_respiratoria',
                                          'B00111':'Cheiro_sabor',
                                          'B007':'Plano_Saude',
                                          'B008':'Teste_Covid',
                                          'C005':'Tempo_afastamento',
                                          'C007':'Carac_trabalho',
                                          'C003':'Motivo_afastamento'})

# Modelagem e Tratamento dos dados

## Substitui os valores numéricos dos Estados para a legenda apropriada a leitura do dado.
df_out2020['Estado'] = df_out2020['Estado'].map({11:'Rondônia', 12:'Acre', 13:'Amazonas',
                                                 14:'Roraima', 15:'Pará',16:'Amapá',
                                                 17:'Tocantins', 21:'Maranhão', 22:'Piauí',
                                                 23: 'Ceará', 24:'Rio Grande do Norte', 25:'Paraíba',
                                                 26:'Pernambuco', 27: 'Alagoas', 28: 'Sergipe',
                                                 29: 'Bahia', 31:'Minas Gerais', 32: 'Espírito Santo',
                                                 33:'Rio de Janeiro',35:'São Paulo', 41: 'Paraná',
                                                 42: 'Santa Catarina', 43: 'Rio Grande do Sul', 50:'Mato
                                                 51:'Mato Grosso',52:'Goiás', 53: 'Distrito Federal'}, na_

## Substitui os valor numérico do Mês para a legenda apropriada a leitura do dado.
df_out2020['Mes'] = df_out2020['Mes'].map({10:'Outubro'}, na_action=None)

## Substitui os valor da Área do município(Urbana ou Rural) para a legenda apropriada a leitura do dado.
df_out2020['Area'] = df_out2020['Area'].map({1:'Urbana', 2:'Rural'}, na_action=None)


## Substitui os valores numéricos do Sexo para a legenda apropriada a leitura do dado.
```

```
df_out2020['Sexo'] = df_out2020['Sexo'].map({1:'Homem', 2:'Mulher'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Febre para a legenda apropriada a leitura do dado.
df_out2020['Febre'] = df_out2020['Febre'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Tosse para a legenda apropriada a leitura do dado.
df_out2020['Tosse'] = df_out2020['Tosse'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Dificuldade respiratoria para a legenda apropriada a leitura
df_out2020['Dific_respiratoria'] = df_out2020['Dific_respiratoria'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_act


## Substitui os valores numéricos sobre sintoma de perda de Cheiro ou sabor para a legenda apropriada a leitura
df_out2020['Cheiro_sabor'] = df_out2020['Cheiro_sabor'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre Plano de Saúde para a legenda apropriada a leitura do dado.
df_out2020['Plano_Saude'] = df_out2020['Plano_Saude'].map({1:'Sim', 2:'Não', 9:'Ignorado'}, na_action=None)

## Substitui os valores numéricos sobre Teste de Covid para a legenda apropriada a leitura do dado.
df_out2020['Teste_Covid'] = df_out2020['Teste_Covid'].map({1:'Sim', 2:'Não', 9:'Ignorado'}, na_action=None)


## Substitui os valores numéricos sobre Tempo de afastamento para a legenda apropriada a leitura do dado.
df_out2020['Tempo_afastamento'] = df_out2020['Tempo_afastamento'].map({1:'Menos de 1 mês',
                                                                       2:'De 1 mês a menos de 1 ano',
                                                                       3:'De 1 ano a menos de 2 anos',
                                                                       4:'2 anos ou mais',
                                                                      np.nan:'Não aplicável'}, na_action=None)

## Substitui os valores numéricos sobre Carac para a legenda apropriada a leitura do dado.
df_out2020['Carac_trabalho'] = df_out2020['Carac_trabalho'].map({1:'Trabalhador doméstico (empregado doméstico,
                                                                 2:'Militar do exercito, marinha ou aeronáutica',
                                                                 3:'Policial militar ou bombeiro mlilitar',
                                                                 4:'Empregado do setor privado',
                                                                 5:'Empregado do setor público (inclusive empresas de
                                                                 6:'Empregador',
                                                                 7:'Conta própria',
                                                                 8:'Trabalhador familiar não remunerado em ajuda a mer
                                                                 9:'Estava fora do mercado de trabalho (fazia apenas a
                                                                np.nan:'Não aplicável'}, na_action=None)

## Substitui os valores numéricos sobre Motivo do afastamento para a legenda apropriada a leitura do dado.
```

```
df_out2020['Motivo_afastamento'] = df_out2020['Motivo_afastamento'].map({1:'Estava em quarentena, isolamento, di
                                                                         2:'Férias, folga ou jornada de trabalho
                                                                         3:'Licença maternidade ou paternidade',
                                                                         4:'Licença remunerada por motivo de sau
                                                                         5:'Outro tipo de licença remunerada (es
                                                                         6:'Afastamento do próprio negócio/empre
                                                                         7:'Fatores ocasionais (mau tempo, paral
                                                                         8:'Outro motivo',
                                                                        np.nan:'Não aplicável'}, na_action=None

# Cria o DataFrame do mês
df_out2020
```

Out[11]:

	Estado	Mes	Area	Idade_Morador	Sexo	Febre	Tosse	Dific_respiratoria	Cheiro_sabor	Plano_Saude	Teste_Covid	Tempo_afastar
0	Rondônia	Outubro	Urbana	36	Homem	Não	Não	Não	Não	Sim	Não	Não ap
1	Rondônia	Outubro	Urbana	30	Mulher	Não	Não	Não	Não	Não	Não	Não ap
2	Rondônia	Outubro	Urbana	13	Homem	Não	Não	Não	Não	Sim	Não	Não ap
3	Rondônia	Outubro	Urbana	11	Homem	Não	Não	Não	Não	Sim	Não	Não ap
4	Rondônia	Outubro	Urbana	57	Mulher	Não	Não	Não	Não	Não	Não	Não ap
...	...	...	...	...	...	...	...	...	...	...	...	
380456	Distrito Federal	Outubro	Rural	45	Mulher	Não	Não	Não	Não	Não	Não	Não ap
380457	Distrito Federal	Outubro	Rural	22	Mulher	Não	Não	Não	Não	Não	Não	Não ap
380458	Distrito Federal	Outubro	Rural	16	Mulher	Não	Não	Não	Não	Não	Não	Não ap
380459	Distrito Federal	Outubro	Rural	83	Homem	Não	Não	Não	Não	Não	Não	Não ap
380460	Distrito Federal	Outubro	Rural	75	Mulher	Não	Não	Não	Não	Não	Não	Não ap

380461 rows × 14 columns

--> Novembro / 2020

In [12]:
```
# Constroi o DataFrame com as colunas selecionadas para o mês de Julho.
df_nov2020 = pd.DataFrame(nov2020,columns = ['UF', 'V1013', 'V1022', 'A002', 'A003', 'B0011', 'B0012', 'B0014',
                                             'B00111','B007','B008', 'C005', 'C007', 'C003'])

# Renomear as colunas
df_nov2020 = df_nov2020.rename(columns = {'UF':'Estado',
                                          'V1013':'Mes',
                                          'V1022':'Area',
                                          'A002':'Idade_Morador',
                                          'A003':'Sexo',
                                          'B0011':'Febre',
                                          'B0012':'Tosse',
                                          'B0014':'Dific_respiratoria',
                                          'B00111':'Cheiro_sabor',
                                          'B007':'Plano_Saude',
                                          'B008':'Teste_Covid',
                                          'C005':'Tempo_afastamento',
                                          'C007':'Carac_trabalho',
                                          'C003':'Motivo_afastamento'})

# Modelagem e Tratamento dos dados

## Substitui os valores numéricos dos Estados para a legenda apropriada a leitura do dado.
df_nov2020['Estado'] = df_nov2020['Estado'].map({11:'Rondônia', 12:'Acre', 13:'Amazonas',
                                                 14:'Roraima', 15:'Pará',16:'Amapá',
                                                 17:'Tocantins', 21:'Maranhão', 22:'Piauí',
                                                 23: 'Ceará', 24:'Rio Grande do Norte', 25:'Paraíba',
                                                 26:'Pernambuco', 27: 'Alagoas', 28: 'Sergipe',
                                                 29: 'Bahia', 31:'Minas Gerais', 32: 'Espírito Santo',
                                                 33:'Rio de Janeiro',35:'São Paulo', 41: 'Paraná',
                                                 42: 'Santa Catarina', 43: 'Rio Grande do Sul', 50:'Mato
                                                 51:'Mato Grosso',52:'Goiás', 53: 'Distrito Federal'}, na_

## Substitui os valor numérico do Mês para a legenda apropriada a leitura do dado.
df_nov2020['Mes'] = df_nov2020['Mes'].map({11:'Novembro'}, na_action=None)

## Substitui os valor da Área do município(Urbana ou Rural) para a legenda apropriada a leitura do dado.
df_nov2020['Area'] = df_nov2020['Area'].map({1:'Urbana', 2:'Rural'}, na_action=None)


## Substitui os valores numéricos do Sexo para a legenda apropriada a leitura do dado.
```

```
df_nov2020['Sexo'] = df_nov2020['Sexo'].map({1:'Homem', 2:'Mulher'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Febre para a legenda apropriada a leitura do dado.
df_nov2020['Febre'] = df_nov2020['Febre'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Tosse para a legenda apropriada a leitura do dado.
df_nov2020['Tosse'] = df_nov2020['Tosse'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre sintoma de Dificuldade respiratoria para a legenda apropriada a leitura
df_nov2020['Dific_respiratoria'] = df_nov2020['Dific_respiratoria'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_act

## Substitui os valores numéricos sobre sintoma de perda de Cheiro ou sabor para a legenda apropriada a leitura
df_nov2020['Cheiro_sabor'] = df_nov2020['Cheiro_sabor'].map({1:'Sim', 2:'Não', 3:'Não sabe'}, na_action=None)

## Substitui os valores numéricos sobre Plano de Saúde para a legenda apropriada a leitura do dado.
df_nov2020['Plano_Saude'] = df_nov2020['Plano_Saude'].map({1:'Sim', 2:'Não', 9:'Ignorado'}, na_action=None)

## Substitui os valores numéricos sobre Teste de Covid para a legenda apropriada a leitura do dado.
df_nov2020['Teste_Covid'] = df_nov2020['Teste_Covid'].map({1:'Sim', 2:'Não', 9:'Ignorado'}, na_action=None)


## Substitui os valores numéricos sobre Tempo de afastamento para a legenda apropriada a leitura do dado.
df_nov2020['Tempo_afastamento'] = df_nov2020['Tempo_afastamento'].map({1:'Menos de 1 mês',
                                                                       2:'De 1 mês a menos de 1 ano',
                                                                       3:'De 1 ano a menos de 2 anos',
                                                                       4:'2 anos ou mais',
                                                                      np.nan:'Não aplicável'}, na_action=None)

## Substitui os valores numéricos sobre Carac para a legenda apropriada a leitura do dado.
df_nov2020['Carac_trabalho'] = df_nov2020['Carac_trabalho'].map({1:'Trabalhador doméstico (empregado doméstico,
                                                                 2:'Militar do exercito, marinha ou aeronáutica',
                                                                 3:'Policial militar ou bombeiro mlilitar',
                                                                 4:'Empregado do setor privado',
                                                                 5:'Empregado do setor público (inclusive empresas de
                                                                 6:'Empregador',
                                                                 7:'Conta própria',
                                                                 8:'Trabalhador familiar não remunerado em ajuda a mer
                                                                 9:'Estava fora do mercado de trabalho (fazia apenas a
                                                                np.nan:'Não aplicável'}, na_action=None)

## Substitui os valores numéricos sobre Motivo do afastamento para a legenda apropriada a leitura do dado.
df_nov2020['Motivo_afastamento'] = df_nov2020['Motivo_afastamento'].map({1:'Estava em quarentena, isolamento, di
```

```
                                                    2:'Férias, folga ou jornada de trabalho
                                                    3:'Licença maternidade ou paternidade',
                                                    4:'Licença remunerada por motivo de saú
                                                    5:'Outro tipo de licença remunerada (es
                                                    6:'Afastamento do próprio negócio/empre
                                                    7:'Fatores ocasionais (mau tempo, paral
                                                    8:'Outro motivo',
                                                    np.nan:'Não aplicável'}, na_action=None

# Cria o DataFrame do mês
df_nov2020
```

Out[12]:

	Estado	Mes	Area	Idade_Morador	Sexo	Febre	Tosse	Dific_respiratoria	Cheiro_sabor	Plano_Saude	Teste_Covid	Tempo_afas
0	Rondônia	Novembro	Urbana	36	Homem	Não	Não	Não	Não	Sim	Não	Não
1	Rondônia	Novembro	Urbana	30	Mulher	Não	Não	Não	Não	Não	Não	Não
2	Rondônia	Novembro	Urbana	13	Homem	Não	Não	Não	Não	Sim	Não	Não
3	Rondônia	Novembro	Urbana	11	Homem	Não	Não	Não	Não	Sim	Não	Não
4	Rondônia	Novembro	Urbana	57	Mulher	Não	Não	Não	Não	Não	Não	Não
...	...	...	...	...	...	...	...	...	...	...	...	
381433	Distrito Federal	Novembro	Rural	45	Mulher	Não	Não	Não	Não	Não	Não	Não
381434	Distrito Federal	Novembro	Rural	22	Mulher	Não	Não	Não	Não	Não	Não	Não
381435	Distrito Federal	Novembro	Rural	16	Mulher	Não	Não	Não	Não	Não	Não	Não
381436	Distrito Federal	Novembro	Rural	83	Homem	Não	Não	Não	Não	Não	Não	Não
381437	Distrito Federal	Novembro	Rural	75	Mulher	Não	Não	Não	Não	Não	Não	Não

381438 rows × 14 columns

Análise Descritiva

--> Por Estado

In [13]:
```
## Descrever qual foi o município que mais foi realizado as entrevistas
df_jul2020['Estado'].describe()
```

Out[13]:
```
count        384166
unique           27
top       São Paulo
freq          35020
Name: Estado, dtype: object
```

In [14]:
```
## Descrição da coluna Estado. Calcula a porcentagem de pessoas de que foram entrevistada no mês de Jullho
df_jul2020['Estado'].value_counts(normalize=True).round(2)*100
```

Out[14]:
```
São Paulo              9.0
Minas Gerais           9.0
Rio de Janeiro         8.0
Santa Catarina         6.0
Rio Grande do Sul      6.0
Paraná                 5.0
Maranhão               5.0
Bahia                  5.0
Ceará                  4.0
Pernambuco             4.0
Espírito Santo         4.0
Goiás                  4.0
Alagoas                3.0
Mato Grosso            3.0
Pará                   3.0
Paraíba                3.0
Amazonas               3.0
Mato Grosso do Sul     2.0
Piauí                  2.0
Rio Grande do Norte    2.0
Sergipe                2.0
Acre                   2.0
Distrito Federal       2.0
Rondônia               1.0
Roraima                1.0
Tocantins              1.0
Amapá                  1.0
Name: Estado, dtype: float64
```

In [15]:
```
## Descrever qual foi o município que mais foi realizado as entrevistas
df_ago2020['Estado'].describe()
```

Out[15]:
```
count           386520
unique              27
top       Minas Gerais
freq             34810
Name: Estado, dtype: object
```

In [16]:
```
## Descrição da coluna Estado. Calcula a porcentagem de pessoas de que foram entrevistada no mês de Agosto
df_ago2020['Estado'].value_counts(normalize=True).round(2)*100
```

Out[16]:
```
Minas Gerais           9.0
São Paulo              9.0
Rio de Janeiro         8.0
Santa Catarina         6.0
Rio Grande do Sul      6.0
Maranhão               5.0
Paraná                 5.0
Ceará                  5.0
Bahia                  4.0
Pernambuco             4.0
Espírito Santo         4.0
Goiás                  4.0
Mato Grosso            3.0
Alagoas                3.0
Pará                   3.0
Amazonas               3.0
Paraíba                3.0
Mato Grosso do Sul     2.0
Piauí                  2.0
Rio Grande do Norte    2.0
Sergipe                2.0
Acre                   2.0
Distrito Federal       2.0
Roraima                1.0
Tocantins              1.0
Rondônia               1.0
Amapá                  1.0
Name: Estado, dtype: float64
```

In [17]:
```
## Descrever qual foi o município que mais foi realizado as entrevistas
df_set2020['Estado'].describe()
```

Out[17]:
```
count           387298
unique              27
top       Minas Gerais
freq             34527
Name: Estado, dtype: object
```

In [18]:
```
## Descrição da coluna Estado. Calcula a porcentagem de pessoas de que foram entrevistada no mês de Setembro
df_set2020['Estado'].value_counts(normalize=True).round(2)*100
```

Out[18]:
```
Minas Gerais           9.0
São Paulo              9.0
Rio de Janeiro         8.0
Santa Catarina         6.0
Rio Grande do Sul      6.0
Maranhão               5.0
Paraná                 5.0
Ceará                  5.0
Bahia                  5.0
Pernambuco             4.0
Espírito Santo         4.0
Goiás                  4.0
Pará                   3.0
Mato Grosso            3.0
Alagoas                3.0
Amazonas               3.0
Paraíba                3.0
Mato Grosso do Sul     2.0
Piauí                  2.0
Sergipe                2.0
Rio Grande do Norte    2.0
Acre                   2.0
Distrito Federal       2.0
Roraima                1.0
Tocantins              1.0
Rondônia               1.0
Amapá                  1.0
Name: Estado, dtype: float64
```

In [19]:
```
## Descrever qual foi o município que mais foi realizado as entrevistas
df_out2020['Estado'].describe()
```

Out[19]:
```
count           380461
unique              27
top       Minas Gerais
freq             34230
Name: Estado, dtype: object
```

In [20]:
```
## Descrição da coluna Estado. Calcula a porcentagem de pessoas de que foram entrevistada no mês de Outrubro
df_out2020['Estado'].value_counts(normalize=True).round(2)*100
```

Out[20]:
```
Minas Gerais           9.0
São Paulo              9.0
Rio de Janeiro         8.0
Santa Catarina         6.0
Rio Grande do Sul      6.0
Maranhão               5.0
Paraná                 5.0
Pernambuco             4.0
Ceará                  4.0
Bahia                  4.0
Espírito Santo         4.0
Goiás                  4.0
Pará                   3.0
Alagoas                3.0
Mato Grosso            3.0
Amazonas               3.0
Paraíba                3.0
Mato Grosso do Sul     2.0
Piauí                  2.0
Sergipe                2.0
Rio Grande do Norte    2.0
Distrito Federal       2.0
Acre                   2.0
Roraima                1.0
Rondônia               1.0
Tocantins              1.0
Amapá                  1.0
Name: Estado, dtype: float64
```

In [21]:
```
## Descrever qual foi o município que mais foi realizado as entrevistas
df_nov2020['Estado'].describe()
```

Out[21]:
```
count           381438
unique              27
top       Minas Gerais
freq             34339
Name: Estado, dtype: object
```

In [22]:
```
## Descrição da coluna Estado. Calcula a porcentagem de pessoas de que foram entrevistada no mês de Novembro
df_nov2020['Estado'].value_counts(normalize=True).round(2)*100
```

Out[22]:
```
Minas Gerais           9.0
São Paulo              8.0
Rio de Janeiro         8.0
Santa Catarina         6.0
Rio Grande do Sul      6.0
Maranhão               5.0
Paraná                 5.0
Pernambuco             4.0
Bahia                  4.0
Ceará                  4.0
Espírito Santo         4.0
Goiás                  4.0
Pará                   3.0
Alagoas                3.0
Mato Grosso            3.0
Amazonas               3.0
Paraíba                3.0
Mato Grosso do Sul     2.0
Piauí                  2.0
Sergipe                2.0
Rio Grande do Norte    2.0
Distrito Federal       2.0
Acre                   2.0
Rondônia               2.0
Roraima                1.0
Tocantins              1.0
Amapá                  1.0
Name: Estado, dtype: float64
```

--> Area (Urbana ou Rural)

In [23]:
```
## Descrição da coluna Area
df_jul2020['Area'].value_counts(normalize=True).round(4)*100
```

Out[23]:
```
Urbana    76.29
Rural     23.71
Name: Area, dtype: float64
```

In [24]:
```
## Descrição da coluna Area
df_ago2020['Area'].value_counts(normalize=True).round(4)*100
```

Out[24]:
```
Urbana    76.03
Rural     23.97
Name: Area, dtype: float64
```

In [25]:
```
## Descrição da coluna Area
df_set2020['Area'].value_counts(normalize=True).round(4)*100
```

Out[25]:
```
Urbana    76.01
Rural     23.99
Name: Area, dtype: float64
```

In [26]:
```
## Descrição da coluna Area
df_out2020['Area'].value_counts(normalize=True).round(4)*100
```

Out[26]:
```
Urbana    76.14
Rural     23.86
Name: Area, dtype: float64
```

In [27]:
```
## Descrição da coluna Area
df_nov2020['Area'].value_counts(normalize=True).round(4)*100
```

Out[27]:
```
Urbana    76.33
Rural     23.67
Name: Area, dtype: float64
```

--> Idade por Mês

In [28]:
```
df_jul2020.describe()
```

Out[28]:

	Idade_Morador
count	384166.000000
mean	36.738118
std	21.781616
min	0.000000
25%	18.000000
50%	36.000000
75%	53.000000
max	111.000000

In [29]:
```
# Contagem de pessoas entrevistadas agrupados por idade.
classe = [0,9,19,29,59,120]
labels = ['0 a 9 anos', '10 a 19 anos', '20 a 29 anos', '30 a 59 anos','Acima de 60']
idade = pd.cut(df_jul2020.Idade_Morador, classe, labels=labels, include_lowest = True)
pd.value_counts(idade)
```

Out[29]:
```
30 a 59 anos    161974
Acima de 60      66253
10 a 19 anos     56983
20 a 29 anos     53004
0 a 9 anos       45952
Name: Idade_Morador, dtype: int64
```

In [30]:
```
df_ago2020.describe()
```

Out[30]:

	Idade_Morador
count	386520.000000
mean	36.787915
std	21.821777
min	0.000000
25%	18.000000
50%	36.000000
75%	54.000000
max	111.000000

In [31]:
```
# Contagem de pessoas entrevistadas agrupados por idade.
classe = [0,9,19,29,59,120]
labels = ['0 a 9 anos', '10 a 19 anos', '20 a 29 anos', '30 a 59 anos','Acima de 60']
idade = pd.cut(df_ago2020.Idade_Morador, classe, labels=labels, include_lowest = True)
pd.value_counts(idade)
```

Out[31]:
```
30 a 59 anos    162746
Acima de 60      67106
10 a 19 anos     57229
20 a 29 anos     53094
0 a 9 anos       46345
Name: Idade_Morador, dtype: int64
```

In [32]:
```
df_set2020.describe()
```

Out[32]:

	Idade_Morador
count	387298.000000
mean	36.862261
std	21.836474
min	0.000000
25%	18.000000
50%	36.000000
75%	54.000000
max	111.000000

In [33]:
```
# Contagem de pessoas entrevistadas agrupados por idade.
classe = [0,9,19,29,59,120]
labels = ['0 a 9 anos', '10 a 19 anos', '20 a 29 anos', '30 a 59 anos','Acima de 60']
idade = pd.cut(df_set2020.Idade_Morador, classe, labels=labels, include_lowest = True)
pd.value_counts(idade)
```

Out[33]:
```
30 a 59 anos    162986
Acima de 60      67656
10 a 19 anos     57302
20 a 29 anos     53150
0 a 9 anos       46204
Name: Idade_Morador, dtype: int64
```

In [34]:
```
df_out2020.describe()
```

Out[34]:

	Idade_Morador
count	380461.000000
mean	36.999611
std	21.849184
min	0.000000
25%	18.000000
50%	36.000000
75%	54.000000
max	111.000000

In [35]:
```
# Contagem de pessoas entrevistadas agrupados por idade.
classe = [0,9,19,29,59,120]
labels = ['0 a 9 anos', '10 a 19 anos', '20 a 29 anos', '30 a 59 anos','Acima de 60']
idade = pd.cut(df_out2020.Idade_Morador, classe, labels=labels, include_lowest = True)
pd.value_counts(idade)
```

Out[35]:
```
30 a 59 anos    160349
Acima de 60      67093
10 a 19 anos     56196
20 a 29 anos     51917
0 a 9 anos       44906
Name: Idade_Morador, dtype: int64
```

df_nov2020.describe()

In [36]:
```
# Contagem de pessoas entrevistadas agrupados por idade.
classe = [0,9,19,29,59,120]
labels = ['0 a 9 anos', '10 a 19 anos', '20 a 29 anos', '30 a 59 anos','Acima de 60']
idade = pd.cut(df_nov2020.Idade_Morador, classe, labels=labels, include_lowest = True)
pd.value_counts(idade)
```

Out[36]:
```
30 a 59 anos    161061
Acima de 60      67587
10 a 19 anos     56284
20 a 29 anos     51744
0 a 9 anos       44762
Name: Idade_Morador, dtype: int64
```

Sexo por mês

In [37]:
```
## Descrição da coluna Sexo
df_jul2020['Sexo'].value_counts(normalize=True).round(4)*100
```

Out[37]:
```
Mulher    51.99
Homem     48.01
Name: Sexo, dtype: float64
```

In [38]:
```
## Descrição da coluna Sexo
df_ago2020['Sexo'].value_counts(normalize=True).round(4)*100
```

Out[38]:
```
Mulher    51.99
Homem     48.01
Name: Sexo, dtype: float64
```

In [39]:
```
## Descrição da coluna Sexo
df_set2020['Sexo'].value_counts(normalize=True).round(4)*100
```

Out[39]:
```
Mulher    52.04
Homem     47.96
Name: Sexo, dtype: float64
```

In [40]:
```
## Descrição da coluna Sexo
df_out2020['Sexo'].value_counts(normalize=True).round(4)*100
```

Out[40]:
```
Mulher    52.08
Homem     47.92
Name: Sexo, dtype: float64
```

In [41]:
```
## Descrição da coluna Sexo
df_nov2020['Sexo'].value_counts(normalize=True).round(4)*100
```

Out[41]:
```
Mulher    52.13
Homem     47.87
Name: Sexo, dtype: float64
```

Sintomas de Febre por mês.

In [42]:
```
## Descrição da coluna Febre
df_jul2020['Febre'].value_counts(normalize=True).round(4)*100
```

Out[42]:
```
Não         98.40
Sim          1.52
Não sabe     0.08
Name: Febre, dtype: float64
```

In [43]:
```
## Descrição da coluna Febre
df_ago2020['Febre'].value_counts(normalize=True).round(4)*100
```

Out[43]:
```
Não         98.64
Sim          1.27
Não sabe     0.09
Name: Febre, dtype: float64
```

In [44]:
```
## Descrição da coluna Febre
df_set2020['Febre'].value_counts(normalize=True).round(4)*100
```

Out[44]:
```
Não         99.15
Sim          0.80
Não sabe     0.06
Name: Febre, dtype: float64
```

In [45]:
```
## Descrição da coluna Febre
df_out2020['Febre'].value_counts(normalize=True).round(4)*100
```

Out[45]:
```
Não         99.21
Sim          0.73
Não sabe     0.06
Name: Febre, dtype: float64
```

In [46]:
```
## Descrição da coluna Febre
df_nov2020['Febre'].value_counts(normalize=True).round(4)*100
```

Out[46]:
```
Não         99.11
Sim          0.83
Não sabe     0.05
Name: Febre, dtype: float64
```

Sintomas de Tosse por mês

In [47]:
```
## Descrição da coluna Tosse
df_jul2020['Tosse'].value_counts(normalize=True).round(4)*100
```

Out[47]:
```
Não         97.70
Sim          2.22
Não sabe     0.08
Name: Tosse, dtype: float64
```

In [48]:
```
## Descrição da coluna Tosse
df_ago2020['Tosse'].value_counts(normalize=True).round(4)*100
```

Out[48]:
```
Não          97.87
Sim           2.04
Não sabe      0.08
Name: Tosse, dtype: float64
```

In [49]:
```
## Descrição da coluna Tosse
df_set2020['Tosse'].value_counts(normalize=True).round(4)*100
```

Out[49]:
```
Não          98.51
Sim           1.44
Não sabe      0.06
Name: Tosse, dtype: float64
```

In [50]:
```
## Descrição da coluna Tosse
df_out2020['Tosse'].value_counts(normalize=True).round(4)*100
```

Out[50]:
```
Não          98.74
Sim           1.21
Não sabe      0.06
Name: Tosse, dtype: float64
```

In [51]:
```
## Descrição da coluna Tosse
df_nov2020['Tosse'].value_counts(normalize=True).round(4)*100
```

Out[51]:
```
Não          98.56
Sim           1.38
Não sabe      0.06
Name: Tosse, dtype: float64
```

Sintomas de dificuldade respiratórias por mês

In [52]:
```
## Descrição da coluna Dificuldade Respiratória
df_jul2020['Dific_respiratoria'].value_counts(normalize=True).round(4)*100
```

Out[52]:
```
Não         99.12
Sim          0.80
Não sabe     0.08
Name: Dific_respiratoria, dtype: float64
```

In [53]:
```
## Descrição da coluna Dificuldade Respiratória
df_ago2020['Dific_respiratoria'].value_counts(normalize=True).round(4)*100
```

Out[53]:
```
Não         99.25
Sim          0.66
Não sabe     0.09
Name: Dific_respiratoria, dtype: float64
```

In [54]:
```
## Descrição da coluna Dificuldade Respiratória
df_set2020['Dific_respiratoria'].value_counts(normalize=True).round(4)*100
```

Out[54]:
```
Não         99.47
Sim          0.47
Não sabe     0.06
Name: Dific_respiratoria, dtype: float64
```

In [55]:
```
## Descrição da coluna Dificuldade Respiratória
df_out2020['Dific_respiratoria'].value_counts(normalize=True).round(4)*100
```

Out[55]:
```
Não         99.56
Sim          0.38
Não sabe     0.06
Name: Dific_respiratoria, dtype: float64
```

In [56]:
```
## Descrição da coluna Dificuldade Respiratória
df_nov2020['Dific_respiratoria'].value_counts(normalize=True).round(4)*100
```

Out[56]:
```
Não         99.54
Sim          0.40
Não sabe     0.06
Name: Dific_respiratoria, dtype: float64
```

Sintomas de falta de paladar e olfato por mês

In [57]:
```
## Descrição da coluna Cheiro e Sabor
df_jul2020['Cheiro_sabor'].value_counts(normalize=True).round(4)*100
```

Out[57]:
```
Não         99.02
Sim          0.89
Não sabe     0.09
Name: Cheiro_sabor, dtype: float64
```

In [58]:
```
## Descrição da coluna Cheiro e Sabor
df_ago2020['Cheiro_sabor'].value_counts(normalize=True).round(4)*100
```

Out[58]:
```
Não         99.23
Sim          0.67
Não sabe     0.10
Name: Cheiro_sabor, dtype: float64
```

In [59]:
```
## Descrição da coluna Cheiro e Sabor
df_set2020['Cheiro_sabor'].value_counts(normalize=True).round(4)*100
```

Out[59]:
```
Não         99.51
Sim          0.42
Não sabe     0.07
Name: Cheiro_sabor, dtype: float64
```

In [60]:
```
## Descrição da coluna Cheiro e Sabor
df_out2020['Cheiro_sabor'].value_counts(normalize=True).round(4)*100
```

Out[60]:
```
Não         99.59
Sim          0.35
Não sabe     0.06
Name: Cheiro_sabor, dtype: float64
```

In [61]:
```
## Descrição da coluna Cheiro e Sabor
df_nov2020['Cheiro_sabor'].value_counts(normalize=True).round(4)*100
```

Out[61]:
```
Não         99.56
Sim          0.39
Não sabe     0.06
Name: Cheiro_sabor, dtype: float64
```

O Entrevistado tem Plano de Saúde por mês

In [62]:
```
## Descrição da coluna Plano de Saúde
df_jul2020['Plano_Saude'].value_counts(normalize=True).round(4)*100
```

Out[62]:
```
Não         76.87
Sim         22.61
Ignorado     0.51
Name: Plano_Saude, dtype: float64
```

In [63]:
```
## Descrição da coluna Plano de Saúde
df_ago2020['Plano_Saude'].value_counts(normalize=True).round(4)*100
```

Out[63]:
```
Não         77.45
Sim         22.06
Ignorado     0.49
Name: Plano_Saude, dtype: float64
```

In [64]:
```
## Descrição da coluna Plano de Saúde
df_set2020['Plano_Saude'].value_counts(normalize=True).round(4)*100
```

Out[64]:
```
Não         77.25
Sim         22.46
Ignorado     0.29
Name: Plano_Saude, dtype: float64
```

In [65]:
```
## Descrição da coluna Plano de Saúde
df_out2020['Plano_Saude'].value_counts(normalize=True).round(4)*100
```

Out[65]:
```
Não         77.11
Sim         22.70
Ignorado     0.19
Name: Plano_Saude, dtype: float64
```

In [66]:
```
## Descrição da coluna Plano de Saúde
df_nov2020['Plano_Saude'].value_counts(normalize=True).round(4)*100
```

Out[66]:
```
Não         76.99
Sim         22.85
Ignorado     0.17
Name: Plano_Saude, dtype: float64
```

Teste de COVID-19 por mês

In [67]:
```
## Descrição da coluna Teste de COVID-19
df_jul2020['Teste_Covid'].value_counts(normalize=True).round(4)*100
```

Out[67]:
```
Não         93.32
Sim          6.16
Ignorado     0.52
Name: Teste_Covid, dtype: float64
```

In [68]:
```
## Descrição da coluna Teste de COVID-19
df_ago2020['Teste_Covid'].value_counts(normalize=True).round(4)*100
```

Out[68]:
```
Não         91.29
Sim          8.20
Ignorado     0.51
Name: Teste_Covid, dtype: float64
```

In [69]:
```
## Descrição da coluna Teste de COVID-19
df_set2020['Teste_Covid'].value_counts(normalize=True).round(4)*100
```

Out[69]:
```
Não         89.61
Sim         10.10
Ignorado     0.29
Name: Teste_Covid, dtype: float64
```

In [70]:
```
## Descrição da coluna Teste de COVID-19
df_out2020['Teste_Covid'].value_counts(normalize=True).round(4)*100
```

Out[70]:
```
Não         88.12
Sim         11.71
Ignorado     0.18
Name: Teste_Covid, dtype: float64
```

In [71]:
```
## Descrição da coluna Teste de COVID-19
df_nov2020['Teste_Covid'].value_counts(normalize=True).round(4)*100
```

Out[71]:
```
Não         86.77
Sim         13.07
Ignorado     0.16
Name: Teste_Covid, dtype: float64
```

Afastamento do trabalho por mês

In [72]:
```
## Descrição da coluna Afastamento do Trabalho
df_jul2020['Tempo_afastamento'].value_counts(normalize=True).round(4)*100
```

Out[72]:
```
Não aplicável                 96.79
De 1 mês a menos de 1 ano      2.70
Menos de 1 mês                 0.42
2 anos ou mais                 0.05
De 1 ano a menos de 2 anos     0.04
Name: Tempo_afastamento, dtype: float64
```

In [73]:
```
## Descrição da coluna Afastamento do Trabalho
df_ago2020['Tempo_afastamento'].value_counts(normalize=True).round(4)*100
```

Out[73]:
```
Não aplicável                 97.54
De 1 mês a menos de 1 ano      2.06
Menos de 1 mês                 0.32
2 anos ou mais                 0.05
De 1 ano a menos de 2 anos     0.03
Name: Tempo_afastamento, dtype: float64
```

In [74]:
```
## Descrição da coluna Afastamento do Trabalho
df_set2020['Tempo_afastamento'].value_counts(normalize=True).round(4)*100
```

Out[74]:
```
Não aplicável                 97.98
De 1 mês a menos de 1 ano      1.66
Menos de 1 mês                 0.28
2 anos ou mais                 0.05
De 1 ano a menos de 2 anos     0.03
Name: Tempo_afastamento, dtype: float64
```

In [75]:
```
## Descrição da coluna Afastamento do Trabalho
df_out2020['Tempo_afastamento'].value_counts(normalize=True).round(4)*100
```

Out[75]:
```
Não aplicável                 98.33
De 1 mês a menos de 1 ano      1.36
Menos de 1 mês                 0.23
2 anos ou mais                 0.05
De 1 ano a menos de 2 anos     0.03
Name: Tempo_afastamento, dtype: float64
```

In [76]:
```
## Descrição da coluna Afastamento do Trabalho
df_nov2020['Tempo_afastamento'].value_counts(normalize=True).round(4)*100
```

Out[76]:
```
Não aplicável                 98.46
De 1 mês a menos de 1 ano      1.22
Menos de 1 mês                 0.24
2 anos ou mais                 0.05
De 1 ano a menos de 2 anos     0.03
Name: Tempo_afastamento, dtype: float64
```

Caracteristica do Trabalho por mês

In [77]:
```
## Descrição da coluna Caracteristica do Trabalho
df_jul2020['Carac_trabalho'].value_counts(normalize=True).round(4)*100
```

Out[77]:
```
Não aplicável
61.58
Empregado do setor privado
15.93
Conta própria
11.30
Empregado do setor público (inclusive empresas de economia mista)
5.68
Trabalhador doméstico (empregado doméstico, cuidados, babá)
1.91
Estava fora do mercado de trabalho (fazia apenas afazeres domésticos, cuidados de pessoas ou produção para pr
óprio consumo)     1.25
Empregador
1.16
Trabalhador familiar não remunerado em ajuda a membro do domicílio ou parente
0.86
Policial militar ou bombeiro mlilitar
0.18
Militar do exercito, marinha ou aeronáutica
0.16
Name: Carac_trabalho, dtype: float64
```

In [78]:
```
## Descrição da coluna Caracteristica do Trabalho
df_ago2020['Carac_trabalho'].value_counts(normalize=True).round(4)*100
```

Out[78]:
```
Não aplicável
61.36
Empregado do setor privado
15.95
Conta própria
11.36
Empregado do setor público (inclusive empresas de economia mista)
5.71
Trabalhador doméstico (empregado doméstico, cuidados, babá)
1.91
Estava fora do mercado de trabalho (fazia apenas afazeres domésticos, cuidados de pessoas ou produção para pr
óprio consumo)     1.34
Empregador
1.17
Trabalhador familiar não remunerado em ajuda a membro do domicílio ou parente
0.88
Policial militar ou bombeiro mlilitar
0.17
Militar do exercito, marinha ou aeronáutica
0.15
Name: Carac_trabalho, dtype: float64
```

In [79]:
```
## Descrição da coluna Caracteristica do Trabalho
df_set2020['Carac_trabalho'].value_counts(normalize=True).round(4)*100
```

Out[79]:
```
Não aplicável
61.07
Empregado do setor privado
16.07
Conta própria
11.48
Empregado do setor público (inclusive empresas de economia mista)
5.72
Trabalhador doméstico (empregado doméstico, cuidados, babá)
1.91
Estava fora do mercado de trabalho (fazia apenas afazeres domésticos, cuidados de pessoas ou produção para pr
óprio consumo)     1.35
Empregador
1.17
Trabalhador familiar não remunerado em ajuda a membro do domicílio ou parente
0.90
Policial militar ou bombeiro mlilitar
0.18
Militar do exercito, marinha ou aeronáutica
0.15
Name: Carac_trabalho, dtype: float64
```

In [80]:
```
## Descrição da coluna Caracteristica do Trabalho
df_out2020['Carac_trabalho'].value_counts(normalize=True).round(4)*100
```

Out[80]:
```
Não aplicável
60.62
Empregado do setor privado
16.36
Conta própria
11.62
Empregado do setor público (inclusive empresas de economia mista)
5.73
Trabalhador doméstico (empregado doméstico, cuidados, babá)
1.94
Estava fora do mercado de trabalho (fazia apenas afazeres domésticos, cuidados de pessoas ou produção para pr
óprio consumo)     1.36
Empregador
1.16
Trabalhador familiar não remunerado em ajuda a membro do domicílio ou parente
0.89
Policial militar ou bombeiro mlilitar
0.17
Militar do exercito, marinha ou aeronáutica
0.15
Name: Carac_trabalho, dtype: float64
```

In [81]:
```
## Descrição da coluna Caracteristica do Trabalho
df_nov2020['Carac_trabalho'].value_counts(normalize=True).round(4)*100
```

Out[81]:
```
Não aplicável
60.43
Empregado do setor privado
16.43
Conta própria
11.68
Empregado do setor público (inclusive empresas de economia mista)
5.74
Trabalhador doméstico (empregado doméstico, cuidados, babá)
1.97
Estava fora do mercado de trabalho (fazia apenas afazeres domésticos, cuidados de pessoas ou produção para pr
óprio consumo)     1.34
Empregador
1.15
Trabalhador familiar não remunerado em ajuda a membro do domicílio ou parente
0.91
Policial militar ou bombeiro mlilitar
0.18
Militar do exercito, marinha ou aeronáutica
0.15
Name: Carac_trabalho, dtype: float64
```

Trabalho Remoto por mês

In [82]:
```
## Descrição da coluna Caracteristica do Trabalho
df_jul2020['Motivo_afastamento'].value_counts(normalize=True).round(4)*100
```

Out[82]:
```
Não aplicável
93.75
Estava em quarentena, isolamento, distanciamento social ou férias coletivas
4.66
Férias, folga ou jornada de trabalho variável
0.49
Licença remunerada por motivo de saúde ou acidente da própria pessoa
0.40
Afastamento do próprio negócio/empresa por motivo de gestação, saúde, acidente, etc., sem ser remunerado por
instituto de previdência     0.22
Outro motivo
0.22
Licença maternidade ou paternidade
0.13
Fatores ocasionais (mau tempo, paralisação nos serviços de transportes, etc.)
0.10
Outro tipo de licença remunerada (estudo, paternidade, casamento, licença prêmio, etc.)
0.03
Name: Motivo_afastamento, dtype: float64
```

In [83]:
```
## Descrição da coluna Caracteristica do Trabalho
df_ago2020['Motivo_afastamento'].value_counts(normalize=True).round(4)*100
```

Out[83]:
```
Não aplicável
95.13
Estava em quarentena, isolamento, distanciamento social ou férias coletivas
3.44
Licença remunerada por motivo de saúde ou acidente da própria pessoa
0.39
Férias, folga ou jornada de trabalho variável
0.36
Afastamento do próprio negócio/empresa por motivo de gestação, saúde, acidente, etc., sem ser remunerado por
instituto de previdência     0.21
Outro motivo
0.19
Licença maternidade ou paternidade
0.13
Fatores ocasionais (mau tempo, paralisação nos serviços de transportes, etc.)
0.10
Outro tipo de licença remunerada (estudo, paternidade, casamento, licença prêmio, etc.)
0.04
Name: Motivo_afastamento, dtype: float64
```

In [84]:
```
## Descrição da coluna Caracteristica do Trabalho
df_set2020['Motivo_afastamento'].value_counts(normalize=True).round(4)*100
```

Out[84]:
```
Não aplicável
95.91
Estava em quarentena, isolamento, distanciamento social ou férias coletivas
2.72
Licença remunerada por motivo de saúde ou acidente da própria pessoa
0.38
Férias, folga ou jornada de trabalho variável
0.32
Afastamento do próprio negócio/empresa por motivo de gestação, saúde, acidente, etc., sem ser remunerado por
instituto de previdência     0.21
Outro motivo
0.20
Licença maternidade ou paternidade
0.12
Fatores ocasionais (mau tempo, paralisação nos serviços de transportes, etc.)
0.09
Outro tipo de licença remunerada (estudo, paternidade, casamento, licença prêmio, etc.)
0.04
Name: Motivo_afastamento, dtype: float64
```

In [85]:
```
## Descrição da coluna Caracteristica do Trabalho
df_out2020['Motivo_afastamento'].value_counts(normalize=True).round(4)*100
```

Out[85]:
```
Não aplicável
96.49
Estava em quarentena, isolamento, distanciamento social ou férias coletivas
2.16
Licença remunerada por motivo de saúde ou acidente da própria pessoa
0.39
Férias, folga ou jornada de trabalho variável
0.31
Afastamento do próprio negócio/empresa por motivo de gestação, saúde, acidente, etc., sem ser remunerado por
instituto de previdência     0.21
Outro motivo
0.19
Licença maternidade ou paternidade
0.11
Fatores ocasionais (mau tempo, paralisação nos serviços de transportes, etc.)
0.09
Outro tipo de licença remunerada (estudo, paternidade, casamento, licença prêmio, etc.)
0.04
Name: Motivo_afastamento, dtype: float64
```

In [86]:
```
## Descrição da coluna Caracteristica do Trabalho
df_nov2020['Motivo_afastamento'].value_counts(normalize=True).round(4)*100
```

Out[86]:
```
Não aplicável
96.74
Estava em quarentena, isolamento, distanciamento social ou férias coletivas
1.91
Licença remunerada por motivo de saúde ou acidente da própria pessoa
0.40
Férias, folga ou jornada de trabalho variável
0.30
Afastamento do próprio negócio/empresa por motivo de gestação, saúde, acidente, etc., sem ser remunerado por
instituto de previdência     0.23
Outro motivo
0.19
Licença maternidade ou paternidade
0.11
Fatores ocasionais (mau tempo, paralisação nos serviços de transportes, etc.)
0.09
Outro tipo de licença remunerada (estudo, paternidade, casamento, licença prêmio, etc.)
0.03
Name: Motivo_afastamento, dtype: float64
```

Exportar arquivo para CSV para Construção de Dashboard no PowerBI

In [87]:
```
df_jul2020.to_csv(r'dados/df_jul2020.csv', index = False)
```

In [88]:
```
df_ago2020.to_csv(r'dados/df_ago2020.csv', index = False)
```

In [89]:
```
df_set2020.to_csv(r'dados/df_set2020.csv', index = False)
```

In [90]:
```
df_out2020.to_csv(r'dados/df_out2020.csv', index = False)
```

In [91]:
```
df_nov2020.to_csv(r'dados/df_nov2020.csv', index = False)
```

Bibliografia

BRASIL. IBGE. . Pesquisa Nacional por Amostra de Domicílios. 2020. Disponível em: https://www.ibge.gov.br/estatisticas/sociais/saude/27947-divulgacao-mensal- (https://www.ibge.gov.br/estatisticas/sociais/saude/27947-divulgacao-mensal-) pnadcovid2.html?edicao=28351&t=downloadsArquivos:. Acesso em: 27 fev. 2022.

BRASIL. MINISTÉRIO DA SAÚDE. . Coronavirus. 2021. Disponível em: https://www.gov.br/saude/pt-br/coronavirus/o-que-e-o-coronavirus (https://www.gov.br/saude/pt-br/coronavirus/o-que-e-o-coronavirus). Acesso em: 27 fev. 2022.

WHO - World Health Organization. Coronavirus disease (COVID-19): symptoms. Symptoms. 2022. Disponível em: https://www.who.int/health-topics/coronavirus#tab=tab_3 (https://www.who.int/health-topics/coronavirus#tab=tab_3). Acesso em: 02 mar. 2022.