Autor : Mauro Duarte

e-mail: mausteph2@gmail.com (mailto:mausteph2@gmail.com)

Levantamento dos Dados ENEM 2021

Este notebook foi desenvolvido com o objetivo de fazer um levantamento dos DADOS do ENEM 2021, por região/estado e assim, realizar um comparativo entre dados de alunos que frequentam escola públicas e privadas. Com isso, foi realizado um dataset, este foi tratado por região e estado, afim de uma melhor compreensão dos dados coletados no ENEM 2021.

Índice

Preparação, Coleta e Organização dos dados

Voltar ao índice

In [1]:
```
#Importando a biblioteca do Pandas e Numpy
import pandas as pd
```

In [2]:
```
#Carregando os dados
df = pd.read_csv("../semantix projeto/microdados_enem_2021/dados/MICRODADOS_EN
```

In [3]:

Out[3]:

	NU_INSCRICAO	NU_ANO	TP_FAIXA_ETARIA	TP_SEXO	TP_ESTADO_CIVIL	TP_COR_RACA
0	210053865474	2021	5	F	1	1
1	210052384164	2021	12	M	1	1
2	210052589243	2021	13	F	3	1
3	210052128335	2021	3	M	1	3
4	210051353021	2021	2	F	1	3

5 rows × 76 columns

In [4]:

In [5]:

Out[5]:

	NU_INSCRICAO	NU_ANO	TP_FAIXA_ETARIA	TP_SEXO	TP_ESTADO_CIVIL	TP_COR_RACA
0	210053865474	2021	5	F	1	1
1	210052384164	2021	12	M	1	1
2	210052589243	2021	13	F	3	1
3	210052128335	2021	3	M	1	3
4	210051353021	2021	2	F	1	3

In [6]:
```
#listando as colunas
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3389832 entries, 0 to 3389831
Data columns (total 76 columns):
 #   Column                  Dtype
---  ------                  -----
 0   NU_INSCRICAO            int64
 1   NU_ANO                  int64
 2   TP_FAIXA_ETARIA         int64
 3   TP_SEXO                 object
 4   TP_ESTADO_CIVIL         int64
 5   TP_COR_RACA             int64
 6   TP_NACIONALIDADE        int64
 7   TP_ST_CONCLUSAO         int64
 8   TP_ANO_CONCLUIU         int64
 9   TP_ESCOLA               int64
 10  TP_ENSINO               float64
 11  IN_TREINEIRO            int64
 12  CO_MUNICIPIO_ESC        float64
 13  NO_MUNICIPIO_ESC        object
 14  CO_UF_ESC               float64
 15  SG_UF_ESC               object
 16  TP_DEPENDENCIA_ADM_ESC  float64
 17  TP_LOCALIZACAO_ESC      float64
 18  TP_SIT_FUNC_ESC         float64
 19  CO_MUNICIPIO_PROVA      int64
 20  NO_MUNICIPIO_PROVA      object
 21  CO_UF_PROVA             int64
 22  SG_UF_PROVA             object
 23  TP_PRESENCA_CN          int64
 24  TP_PRESENCA_CH          int64
 25  TP_PRESENCA_LC          int64
 26  TP_PRESENCA_MT          int64
 27  CO_PROVA_CN             float64
 28  CO_PROVA_CH             float64
 29  CO_PROVA_LC             float64
 30  CO_PROVA_MT             float64
 31  NU_NOTA_CN              float64
 32  NU_NOTA_CH              float64
 33  NU_NOTA_LC              float64
 34  NU_NOTA_MT              float64
 35  TX_RESPOSTAS_CN         object
 36  TX_RESPOSTAS_CH         object
 37  TX_RESPOSTAS_LC         object
 38  TX_RESPOSTAS_MT         object
 39  TP_LINGUA               int64
 40  TX_GABARITO_CN          object
 41  TX_GABARITO_CH          object
 42  TX_GABARITO_LC          object
 43  TX_GABARITO_MT          object
 44  TP_STATUS_REDACAO       float64
 45  NU_NOTA_COMP1           float64
 46  NU_NOTA_COMP2           float64
 47  NU_NOTA_COMP3           float64
 48  NU_NOTA_COMP4           float64
```

```
 49  NU_NOTA_COMP5           float64
 50  NU_NOTA_REDACAO         float64
 51  Q001                    object
 52  Q002                    object
 53  Q003                    object
 54  Q004                    object
 55  Q005                    float64
 56  Q006                    object
 57  Q007                    object
 58  Q008                    object
 59  Q009                    object
 60  Q010                    object
 61  Q011                    object
 62  Q012                    object
 63  Q013                    object
 64  Q014                    object
 65  Q015                    object
 66  Q016                    object
 67  Q017                    object
 68  Q018                    object
 69  Q019                    object
 70  Q020                    object
 71  Q021                    object
 72  Q022                    object
 73  Q023                    object
 74  Q024                    object
 75  Q025                    object
dtypes: float64(22), int64(17), object(37)
memory usage: 1.9+ GB
```

In [7]:
```
#   apenas as colunas que serão utilizadas
colunas = [
    'NU_INSCRICAO',
    'TP_FAIXA_ETARIA',
    'TP_SEXO',
    'TP_ESTADO_CIVIL',
    'TP_COR_RACA',
    'TP_ST_CONCLUSAO',
    'TP_ESCOLA',
    'TP_ENSINO',
    'IN_TREINEIRO',
    'TP_LOCALIZACAO_ESC',
    'SG_UF_PROVA',
    'TP_PRESENCA_CN',
    'TP_PRESENCA_CH',
    'TP_PRESENCA_LC',
    'TP_PRESENCA_MT',
    'NU_NOTA_CN',
    'NU_NOTA_CH',
    'NU_NOTA_LC',
    'NU_NOTA_MT',
    'TP_LINGUA',
    'TP_STATUS_REDACAO',
    'NU_NOTA_COMP1',
    'NU_NOTA_COMP2',
    'NU_NOTA_COMP3',
    'NU_NOTA_COMP4',
    'NU_NOTA_COMP5',
    'NU_NOTA_REDACAO',
    'Q005',
    'Q006',
    'Q008',
    'Q009',
    'Q012',
    'Q014',
    'Q019',
    'Q024',
    'Q025'
]
```

In [8]:
```
df = pd.read_csv('../semantix projeto/microdados_enem_2021/dados/MICRODADOS_EN
```

In [9]:
```
#Acerto nome das colunas
df = df.rename(columns={
    'NU_INSCRICAO':'INSCRICAO',
    'TP_FAIXA_ETARIA':'FAIXA_ETARIA',
    'TP_SEXO':'SEXO',
    'TP_ESTADO_CIVIL':'ESTADO_CIVIL',
    'TP_COR_RACA':'COR_RACA',
    'TP_ST_CONCLUSAO':'CONCLUSAO_ENSINO',
    'TP_ESCOLA':'TIPO_ESCOLA',
    'TP_ENSINO':'ENSINO',
    'IN_TREINEIRO':'TREINEIRO',
    'TP_LOCALIZACAO_ESC':'LOCALIZACAO_ESCOLA',
    'SG_UF_PROVA':'UF_PROVA',
    'TP_PRESENCA_CN':'PRESENCA_CIEN_NATUREZA',
    'TP_PRESENCA_CH':'PRESENCA_CIEN_HUMANAS',
    'TP_PRESENCA_LC':'PRESENCA_LING_CODIGOS',
    'TP_PRESENCA_MT':'PRESENCA_MATEMATICA',
    'NU_NOTA_CN':'NOTA_CIEN_NATUREZA',
    'NU_NOTA_CH':'NOTA_CIEN_HUMANAS',
    'NU_NOTA_LC':'NOTA_LING_CODIGOS',
    'NU_NOTA_MT':'NOTA_MATEMATICA',
    'TP_LINGUA':'LINGUA_ESTRANGEIRA',
    'TP_STATUS_REDACAO':'SITUACAO_REDACAO',
    'NU_NOTA_COMP1':'NOTA_COMPETENCIA_1',
    'NU_NOTA_COMP2':'NOTA_COMPETENCIA_2',
    'NU_NOTA_COMP3':'NOTA_COMPETENCIA_3',
    'NU_NOTA_COMP4':'NOTA_COMPETENCIA_4',
    'NU_NOTA_COMP5':'NOTA_COMPETENCIA_5',
    'NU_NOTA_REDACAO':'NOTA_REDACAO',
    'Q005':'PESSOAS_POR_RESIDENCIA',
    'Q006':'RENDA_MENSAL_FAMILIA',
    'Q008':'RESID_TEM_BANHEIRO',
    'Q009':'RESID_TEM_QUARTOS',
    'Q012':'RESID_TEM_GELADEIRA',
    'Q014':'RESID_TEM_LAVA_ROUPAS',
    'Q019':'RESID_TEM_TV_CORES',
    'Q024':'RESID_TEM_COMPUTADOR',
    'Q025':'RESID_TEM_INTERNET'
})
```

In [10]:
```
df.FAIXA_ETARIA = df.FAIXA_ETARIA.replace(
{
    1:"Menor 17 Anos",
    2:"17 anos",
    3:"18 anos",
    4:"19 anos",
    5:"20 anos",
    6:"21 anos",
    7:"22 anos",
    8:"23 anos",
    9:"24 anos",
    10:"25 anos",
    11:"Entre 26 e 30 anos",
    12:"Entre 31 e 35 anos",
    13:"Entre 36 e 40 anos",
    14:"Entre 41 e 45 anos",
    15:"Entre 46 e 50 anos",
    16:"Entre 51 e 55 anos",
    17:"Entre 56 e 60 anos",
    18:"Entre 61 e 65 anos",
    19:"Entre 66 e 70 anos",
    20:"Maior de 70 anos"
})
```

In [11]:
```
df.shape
```

Out[11]: (3389832, 36)

In [12]:
```
#Identificando o estado Civil
df.ESTADO_CIVIL = df.ESTADO_CIVIL.replace(
{
    0:"Não Informado",
    1:"Solteiro(a)",
    2:"Casado(a)/Mora com companheiro(a)",
    3:"Divorciado(a)/Desquitado(a)/Separado(a)",
    4:"Viúvo(a)"
})
```

In [13]:
```
#Identificando COR/RAÇA
df.COR_RACA = df.COR_RACA.replace({
    0:"Não declarado",
    1:"Branca",
    2:"Preta",
    3:"Parda",
    4:"Amarela",
    5:"Indígena",
    6:"Não dispõe da informação"
})
```

In [14]:
```
# Identificando a coluna Conclusão Ensino
df.CONCLUSAO_ENSINO = df.CONCLUSAO_ENSINO.replace({
    1:"Já concluí o Ensino Médio",
    2:"Estou cursando e concluirei o Ensino Médio em 2021",
    3:"Estou cursando e concluirei o Ensino Médio após 2021",
    4:"Não concluí e não estou cursando o Ensino Médio"
})
```

In [15]:
```
# Identificando a coluna Tipo Escola
df.TIPO_ESCOLA = df.TIPO_ESCOLA.replace({
    1:"Não Respondeu",
    2:"Pública",
    3:"Privada"
```

In [16]:
```
# Identificando a coluna Ensino
df.ENSINO = df.ENSINO.replace({
    1:"Tipo Instituição - Ensino Regular",
    2:"Tipo Instituição - Educação Especial - Modalidade Substitutiva"
```

In [17]:
```
#Identificando a coluna Treineiro
df.TREINEIRO = df.TREINEIRO.replace({
    0:"Não - Treineiro",
    1:"Treineiro"
```

In [18]:
```
# Criando a coluna Região
```

In [19]:

Out[19]:

	INSCRICAO	FAIXA_ETARIA	SEXO	ESTADO_CIVIL	COR_RACA	CONCLUSAO_EN
0	210053865474	20 anos	F	Solteiro(a)	Branca	Já concluí o E
1	210052384164	Entre 31 e 35 anos	M	Solteiro(a)	Branca	Já concluí o E
2	210052589243	Entre 36 e 40 anos	F	Divorciado(a)/Desquitado(a) /Separado(a)	Branca	Já concluí o E
3	210052128335	18 anos	M	Solteiro(a)	Parda	Estou cursa concluirei o E Médio
4	210051353021	17 anos	F	Solteiro(a)	Parda	Estou cursa concluirei o E Médio

In [20]:
```
# Definindo as informações da coluna Região
df.Regiao = df.UF_PROVA.replace({
    "AC":"Norte",
    "AM":"Norte",
    "RO":"Norte",
    "RR":"Norte",
    "PA":"Norte",
    "AP":"Norte",
    "TO":"Norte",
    "MA":"Nordeste",
    "PI":"Nordeste",
    "CE":"Nordeste",
    "RN":"Nordeste",
    "PB":"Nordeste",
    "PE":"Nordeste",
    "AL":"Nordeste",
    "SE":"Nordeste",
    "BA":"Nordeste",
    "MT":"Centro-Oeste",
    "MS":"Centro-Oeste",
    "GO":"Centro-Oeste",
    "DF":"Centro-Oeste",
    "MG":"Sudeste",
    "SP":"Sudeste",
    "ES":"Sudeste",
    "RJ":"Sudeste",
    "PR":"SUL",
    "SC":"SUL",
    "RS":"SUL"
})
```

In [21]:

Out[21]:

	INSCRICAO	FAIXA_ETARIA	SEXO	ESTADO_CIVIL	COR_RACA	CONCLUSAO_EN
0	210053865474	20 anos	F	Solteiro(a)	Branca	Já concluí o E
1	210052384164	Entre 31 e 35 anos	M	Solteiro(a)	Branca	Já concluí o E
2	210052589243	Entre 36 e 40 anos	F	Divorciado(a)/Desquitado(a) /Separado(a)	Branca	Já concluí o E
3	210052128335	18 anos	M	Solteiro(a)	Parda	Estou cursa concluirei o E Médio
4	210051353021	17 anos	F	Solteiro(a)	Parda	Estou cursa concluirei o E Médio

In [22]:
```
#Identificando a coluna Localização Escola
df.LOCALIZACAO_ESCOLA = df.LOCALIZACAO_ESCOLA.replace({
    1:"Urbana",
    2:"Rural"
})
```

In [23]:
```
#Identificando as colunas Presença Prova e Redação
df.PRESENCA_CIEN_NATUREZA = df.PRESENCA_CIEN_HUMANAS.replace({
    0:"Faltou Prova",
    1:"Presente Prova",
    2:"Eliminado Prova"
})
df.PRESENCA_CIEN_HUMANAS = df.PRESENCA_CIEN_HUMANAS.replace({
    0:"Faltou Prova",
    1:"Presente Prova",
    2:"Eliminado Prova"
})
df.PRESENCA_LING_CODIGOS = df.PRESENCA_LING_CODIGOS.replace({
    0:"Faltou Prova",
    1:"Presente Prova",
    2:"Eliminado Prova"
})
df.PRESENCA_MATEMATICA = df.PRESENCA_MATEMATICA.replace({
    0:"Faltou Prova",
    1:"Presente Prova",
    2:"Eliminado Prova"
})
df.LINGUA_ESTRANGEIRA = df.LINGUA_ESTRANGEIRA.replace({
    0:"Inglês",
    1:"Espanhol",
})
df.SITUACAO_REDACAO = df.SITUACAO_REDACAO.replace({
    1:'Sem problemas',
    2:'Anulada',
    3:'Cópia Texto Motivador',
    4:'Em Branco',
    6:'Fuga ao tema',
    7:'Não atendimento ao tipo textual',
    8:'Texto insuficiente',
    9:'Parte desconectada'
})
```

In [24]:
```
# Identificando algumas colunas selecionadas do questionário Socioeconomico
df.RENDA_MENSAL_FAMILIA = df.RENDA_MENSAL_FAMILIA.replace({
    'A':'Nenhuma Renda',
    'B':'Ate R$ 1.100,00',
    'C':'De R$ 1.100,01 - R$ 1.650,00.',
    'D':'De R$ 1.650,01 - R$ 2.200,00.',
    'E':'De R$ 2.200,01 - R$ 2.750,00.',
    'F':'De R$ 2.750,01 - R$ 3.300,00.',
    'G':'De R$ 3.300,01 - R$ 4.400,00.',
    'H':'De R$ 4.400,01 - R$ 5.500,00.',
    'I':'De R$ 5.500,01 - R$ 6.600,00.',
    'J':'De R$ 6.600,01 - R$ 7.700,00.',
    'K':'De R$ 7.700,01 - R$ 8.800,00.',
    'L':'De R$ 8.800,01 - R$ 9.900,00.',
    'M':'De R$ 9.900,01 - R$ 11.000,00.',
    'N':'De R$ 11.000,01 - R$ 13.200,00.',
    'O':'De R$ 13.200,01 - R$ 16.500,00.',
    'P':'De R$ 16.500,01 - R$ 22.000,00.',
    'Q':'Acima de R$ 22.000,00.'
})
df.RESID_TEM_BANHEIRO = df.RESID_TEM_BANHEIRO.replace({
    'A':'Não.',
    'B':'Sim, um.',
    'C':'Sim, dois.',
    'D':'Sim, três.',
    'E':'Sim, quatro ou mais.'
})
df.RESID_TEM_QUARTOS = df.RESID_TEM_QUARTOS.replace({
    'A':'Não.',
    'B':'Sim, um.',
    'C':'Sim, dois.',
    'D':'Sim, três.',
    'E':'Sim, quatro ou mais.'
})
df.RESID_TEM_GELADEIRA = df.RESID_TEM_GELADEIRA.replace({
    'A':'Não.',
    'B':'Sim, um.',
    'C':'Sim, dois.',
    'D':'Sim, três.',
    'E':'Sim, quatro ou mais.'
})
df.RESID_TEM_LAVA_ROUPAS = df.RESID_TEM_LAVA_ROUPAS.replace({
    'A':'Não.',
    'B':'Sim, um.',
    'C':'Sim, dois.',
    'D':'Sim, três.',
    'E':'Sim, quatro ou mais.'
})
df.RESID_TEM_TV_CORES = df.RESID_TEM_TV_CORES.replace({
    'A':'Não.',
    'B':'Sim, um.',
    'C':'Sim, dois.',
    'D':'Sim, três.',
    'E':'Sim, quatro ou mais.'
})
```

```
df.RESID_TEM_COMPUTADOR = df.RESID_TEM_COMPUTADOR.replace({
    'A':'Não.',
    'B':'Sim, um.',
    'C':'Sim, dois.',
    'D':'Sim, três.',
    'E':'Sim, quatro ou mais.'
})
df.RESID_TEM_INTERNET = df.RESID_TEM_INTERNET.replace({
    'A':'Não.',
    'B':'Sim.'
})
```

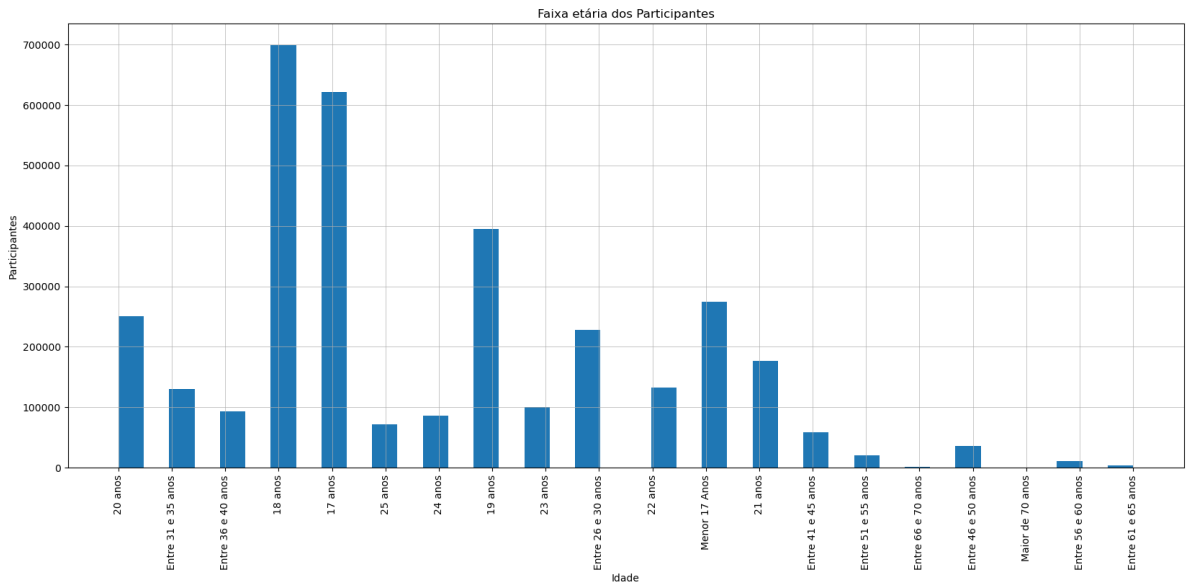
In [ ]:
```
# Salvando tabelas por região para consutlas futuras
regiao = sorted(df['Regiao'].unique())
for r in regiao:
    caminho_arquivo = "../semantix projeto/microdados_enem_2021/dados/enem_reg
    df[df['Regiao']== r].to_csv(caminho_arquivo)
```

In [25]:
```
import matplotlib
```

Análise dos dados - Faixa Etária x Estados

Voltar ao índice

In [26]:
```
#Gráfico Faixa etária participação por iodade
idade = df['FAIXA_ETARIA']
idade.hist(figsize=(20,8),bins=40)
plt.ylabel('Participantes')
plt.xlabel('Idade')
plt.grid(True, lw = 0.50)
plt.xticks(rotation = 90)
plt.title('Faixa etária dos Participantes')
```



In [27]:
```
#Participação por estados
estados = df['UF_PROVA']

estados.hist(figsize=(15,5),bins=60, color = 'red')
plt.ylabel('Participantes')
plt.yticks([50000, 100000, 150000, 200000, 250000, 300000, 350000, 400000, 450
plt.xlabel('Estados')
plt.grid(True, lw = 0.50)
plt.title('Participantes por Estados')
print()
```

In [28]:
```
uf = df['UF_PROVA'].value_counts()
```

Out[28]:
```
SP    509954
MG    327829
BA    266194
RJ    238347
CE    220517
PE    193616
PA    185978
RS    150484
PR    144282
GO    136915
MA    127905
PB    102002
AM     89778
RN     80820
SC     80765
PI     79969
DF     67501
ES     64181
AL     56584
MT     56085
SE     53796
MS     42490
RO     32801
TO     30873
AP     21774
AC     20336
RR      8056
Name: UF_PROVA, dtype: int64
```

In [29]:
```
#Filtrando os dados.
dados = df.filter(['INSCRICAO','Regiao']).groupby('Regiao').count().sort_value
dados
```

Out[29]:

Regiao	INSCRICAO
Nordeste	1181403
Sudeste	1140311
Norte	389596
SUL	375531
Centro-Oeste	302991

In [30]:
```
dados = df.filter(['TIPO_ESCOLA','RESID_TEM_BANHEIRO']).groupby('TIPO_ESCOLA')
dados
```

Out[30]:

TIPO_ESCOLA	RESID_TEM_BANHEIRO
Não Respondeu	2238975
Pública	958611
Privada	192244

In [31]:

In [32]:

Out[32]:

	INSCRICAO	FAIXA_ETARIA	SEXO	ESTADO_CIVIL	COR_RACA	CONCLUSAO_EN
0	210053865474	20 anos	F	Solteiro(a)	Branca	Já concluí o E
1	210052384164	Entre 31 e 35 anos	M	Solteiro(a)	Branca	Já concluí o E
2	210052589243	Entre 36 e 40 anos	F	Divorciado(a)/Desquitado(a) /Separado(a)	Branca	Já concluí o E
3	210052128335	18 anos	M	Solteiro(a)	Parda	Estou cursa concluirei o E Médio
4	210051353021	17 anos	F	Solteiro(a)	Parda	Estou cursa concluirei o E Médio

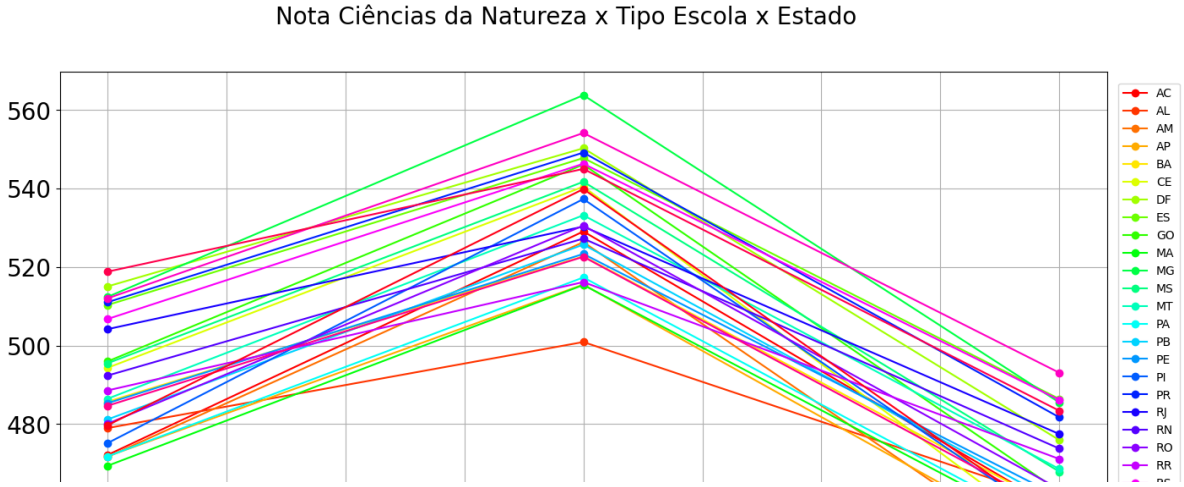
Análise dos dados - Prova e Redação

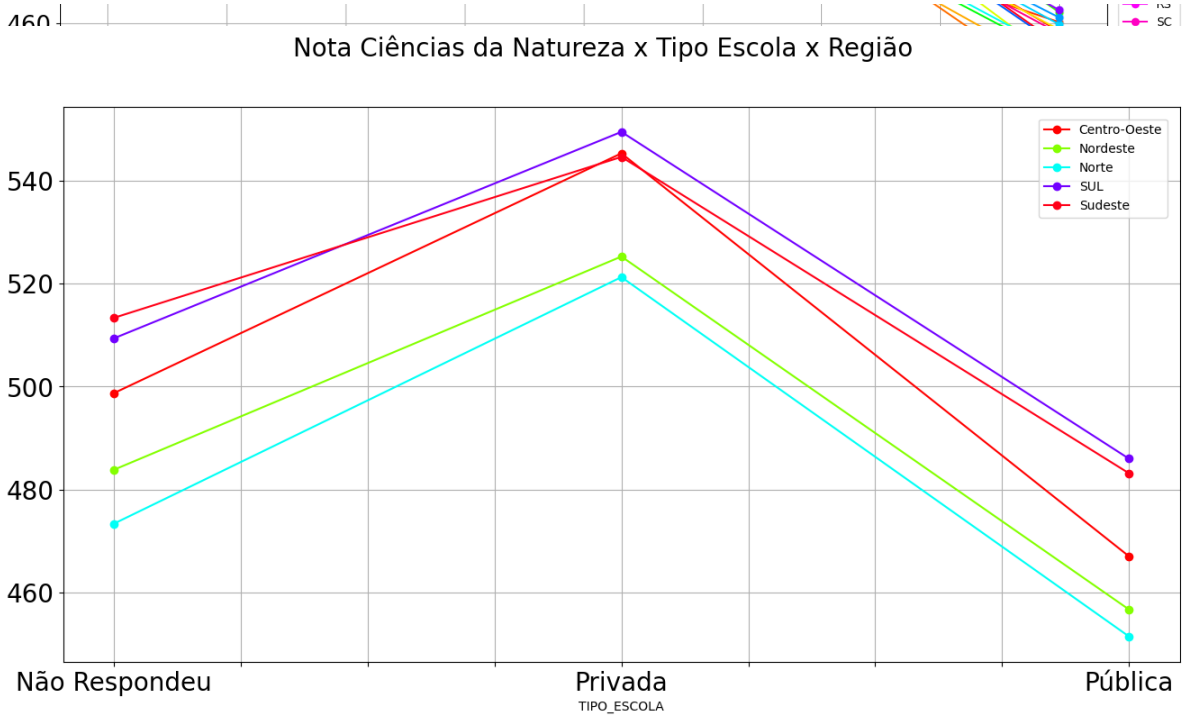
Voltar ao índice

Uma análise detalhada dos índice de nota de Redação por Estado e Região, entre as escolas Públicas e Privadas.

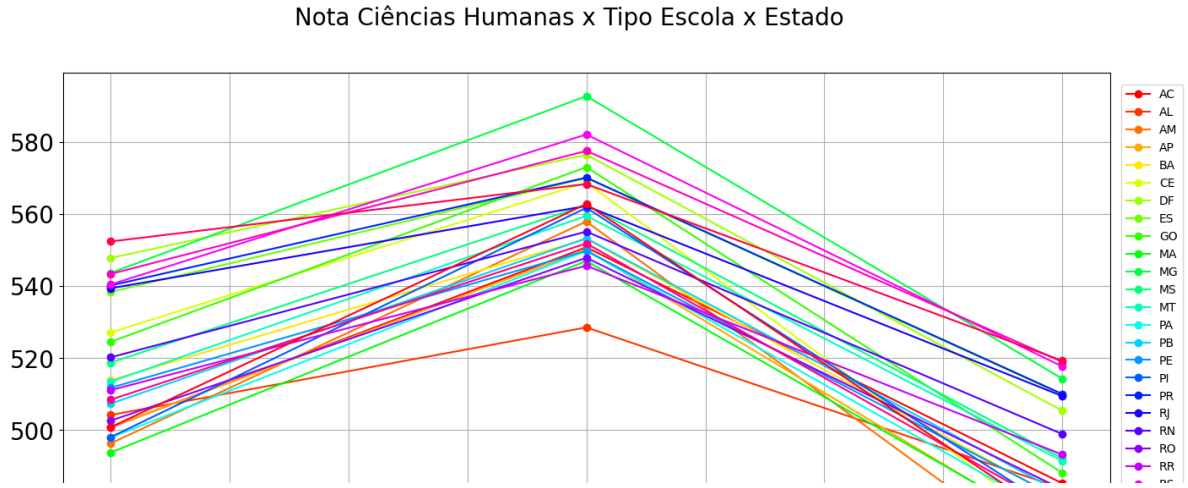
In [33]:
```
import plotly.graph_objects as go
```

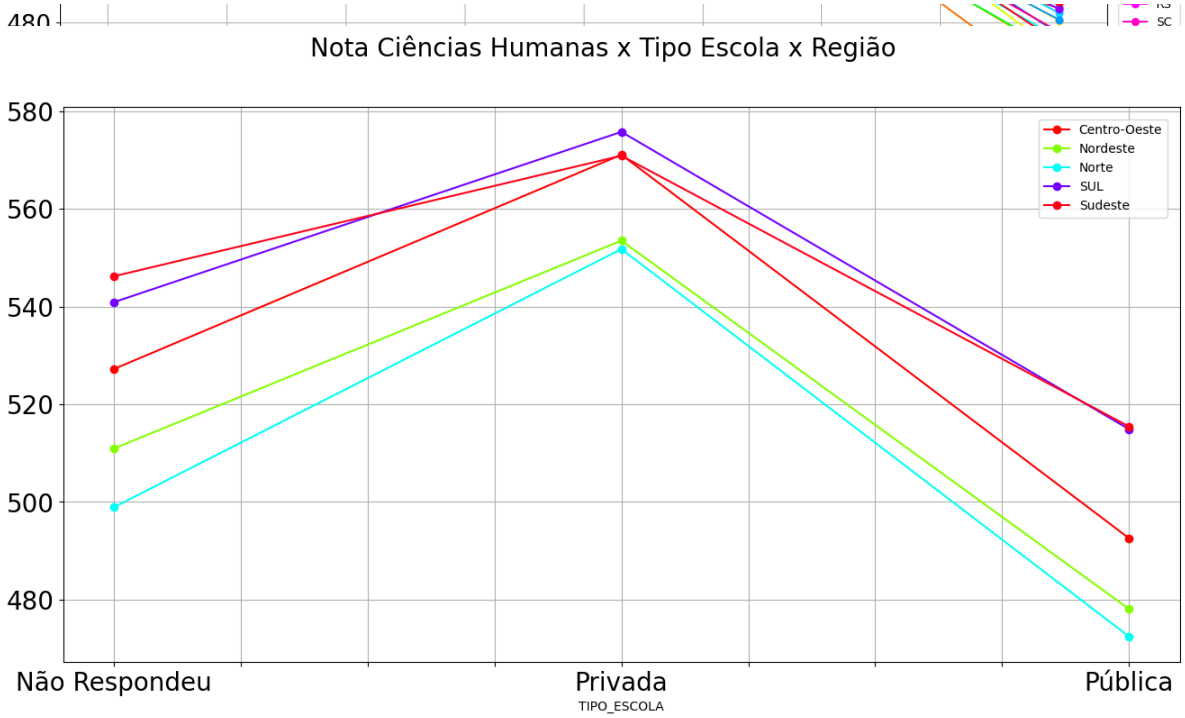
In [34]:
```
fig,ax = plt.subplots(figsize=(16,8))
plt.suptitle('Nota Ciências da Natureza x Tipo Escola x Estado',fontsize=20)

ax = df.filter(items=['NOTA_CIEN_NATUREZA','TIPO_ESCOLA','UF_PROVA'])\
.groupby(['TIPO_ESCOLA','UF_PROVA'])\
.mean().sort_values(by='NOTA_CIEN_NATUREZA',ascending=False)\
.unstack().plot(ax=ax,fontsize=20, colormap='hsv', grid=True, marker='o' ) #Se
                                                                            #au
                                                                            #mu
                                                                            #co
                                                                            #ma


#Modifica as legendas dos estados
handles, labels = ax.get_legend_handles_labels()
import re
edited_labels = [re.search(',\s(.+?)\)',label).group(1) for label in labels]
ax.legend(edited_labels, bbox_to_anchor=(1,1), loc=0, borderaxespad= 1)

print()
fig,ax = plt.subplots(figsize=(16,8))
plt.suptitle('Nota Ciências da Natureza x Tipo Escola x Região',fontsize=20)

ax = df.filter(items=['NOTA_CIEN_NATUREZA','TIPO_ESCOLA','Regiao'])\
.groupby(['TIPO_ESCOLA','Regiao'])\
.mean().sort_values(by='NOTA_CIEN_NATUREZA',ascending=False)\
.unstack().plot(ax=ax,fontsize=20, colormap='hsv', grid=True, marker='o' ) #Se
                                                                            #au
                                                                            #mu
                                                                            #co
                                                                            #ma


#Modifica as legendas dos estados
handles, labels = ax.get_legend_handles_labels()
import re
edited_labels = [re.search(',\s(.+?)\)',label).group(1) for label in labels]
ax.legend(edited_labels, bbox_to_anchor=(1,1), loc=0, borderaxespad= 1)

print()
```



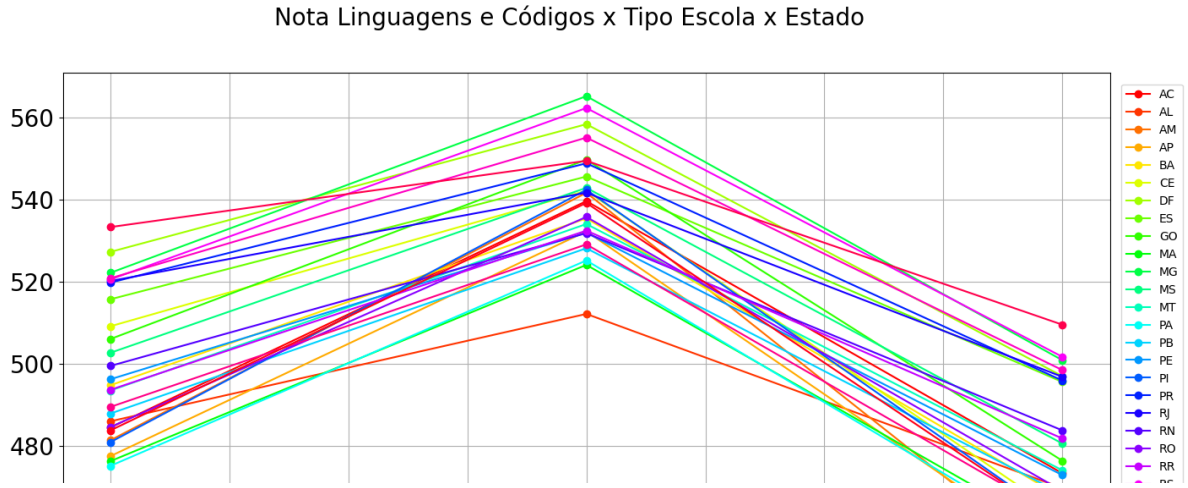

Nota Ciências da Natureza x Tipo Escola x Região
Centro-Oeste
Nordeste
Norte
SUL
Sudeste
540
520
500
480
460
Não Respondeu
Privada
Pública
TIPO_ESCOLA

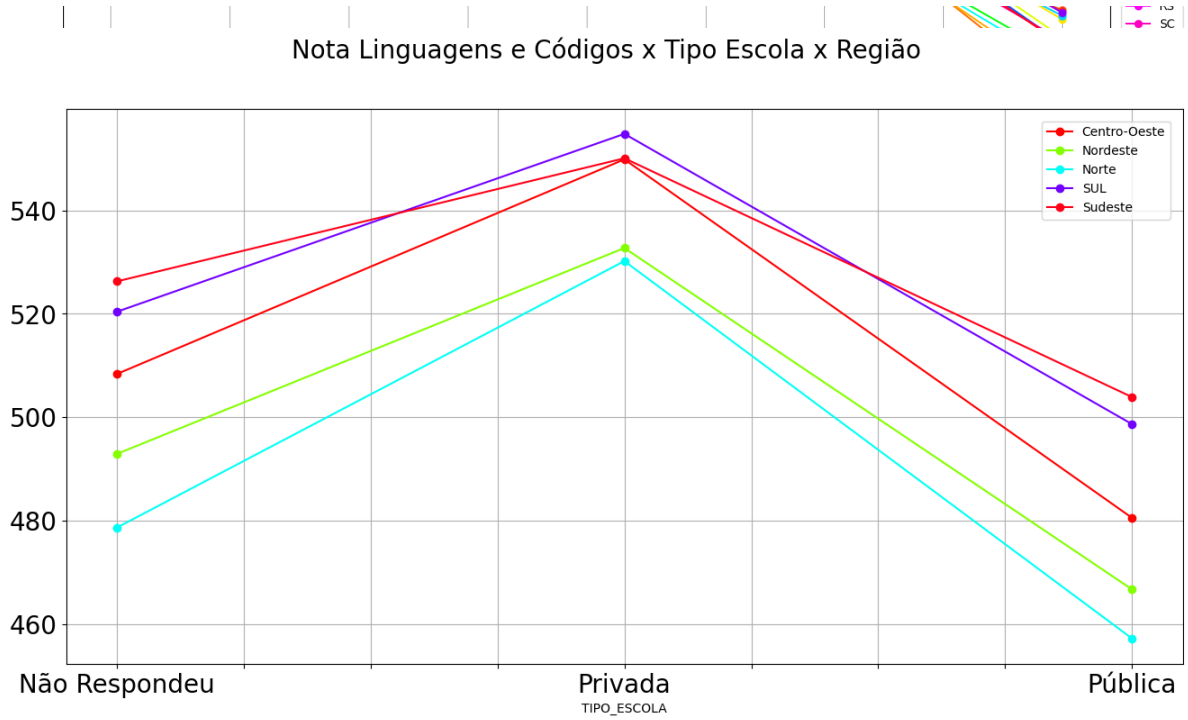
In [35]:
```
fig,ax = plt.subplots(figsize=(16,8))
plt.suptitle('Nota Ciências Humanas x Tipo Escola x Estado',fontsize=20)

ax = df.filter(items=['NOTA_CIEN_HUMANAS','TIPO_ESCOLA','UF_PROVA'])\
.groupby(['TIPO_ESCOLA','UF_PROVA'])\
.mean().sort_values(by='NOTA_CIEN_HUMANAS',ascending=False)\
.unstack().plot(ax=ax,fontsize=20, colormap='hsv', grid=True, marker='o' ) #Se
                                                                            #au
                                                                            #mu
                                                                            #co
                                                                            #ma


#Modifica as legendas dos estados
handles, labels = ax.get_legend_handles_labels()
import re
edited_labels = [re.search(',\s(.+?)\)',label).group(1) for label in labels]
ax.legend(edited_labels, bbox_to_anchor=(1,1), loc=0, borderaxespad= 1)

print()
fig,ax = plt.subplots(figsize=(16,8))
plt.suptitle('Nota Ciências Humanas x Tipo Escola x Região',fontsize=20)

ax = df.filter(items=['NOTA_CIEN_HUMANAS','TIPO_ESCOLA','Regiao'])\
.groupby(['TIPO_ESCOLA','Regiao'])\
.mean().sort_values(by='NOTA_CIEN_HUMANAS',ascending=False)\
.unstack().plot(ax=ax,fontsize=20, colormap='hsv', grid=True, marker='o' ) #Se
                                                                            #au
                                                                            #mu
                                                                            #co
                                                                            #ma


#Modifica as legendas dos estados
handles, labels = ax.get_legend_handles_labels()
import re
edited_labels = [re.search(',\s(.+?)\)',label).group(1) for label in labels]
ax.legend(edited_labels, bbox_to_anchor=(1,1), loc=0, borderaxespad= 1)

print()
```



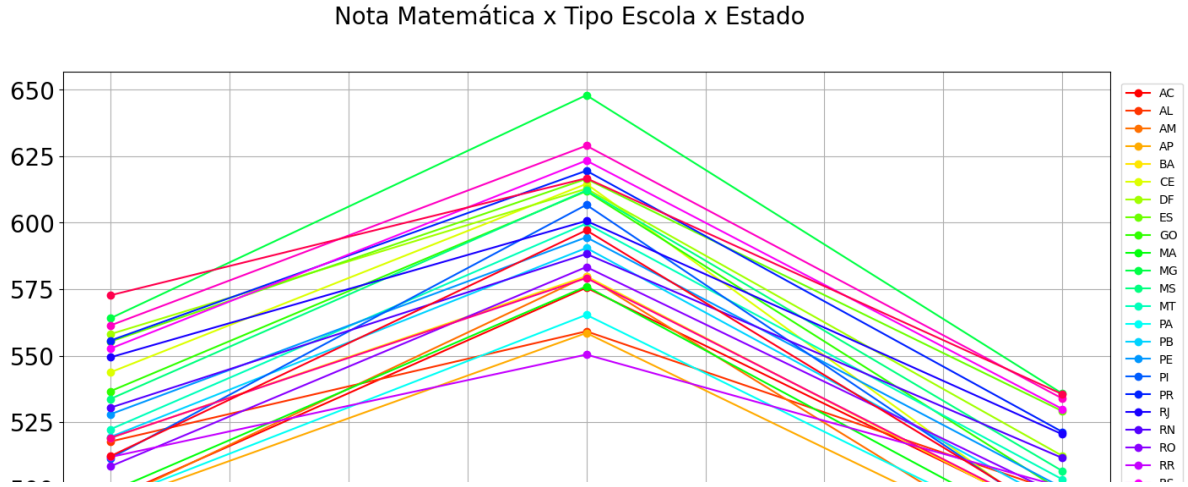

Nota Ciências Humanas x Tipo Escola x Região
580
560
540
520
500
480
Não Respondeu
Privada
Pública
TIPO_ESCOLA
Centro-Oeste
Nordeste
Norte
SUL
Sudeste

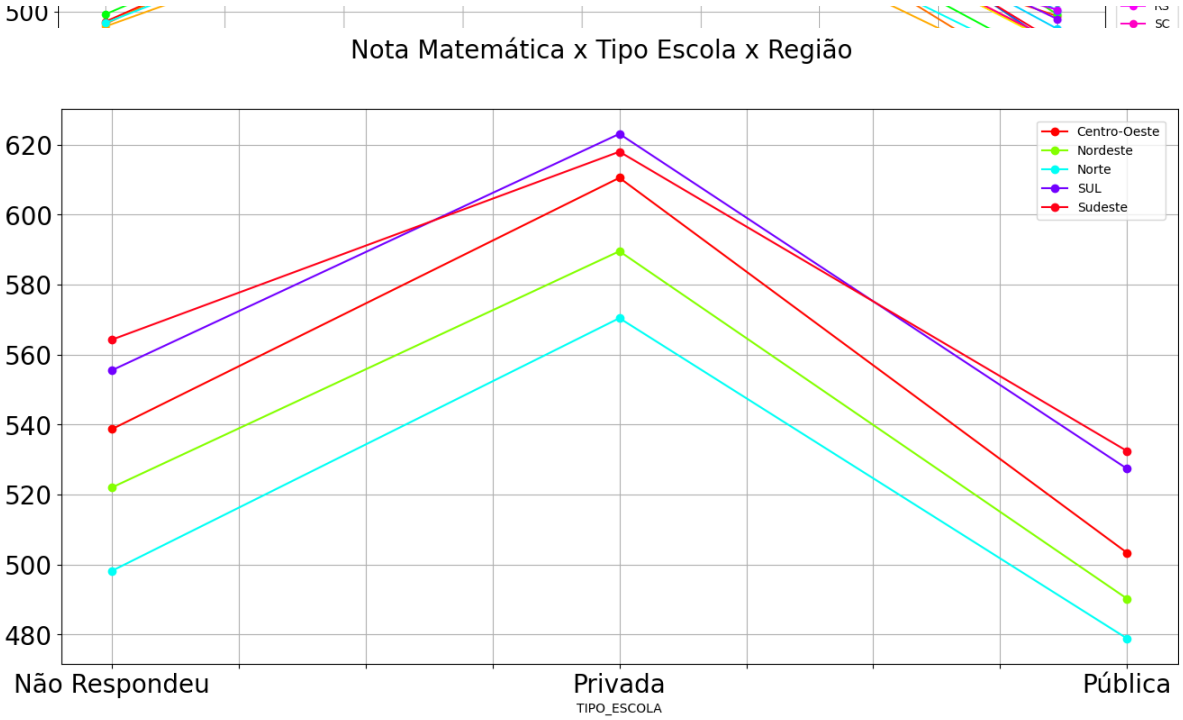
In [36]:
```
fig,ax = plt.subplots(figsize=(16,8))
plt.suptitle('Nota Linguagens e Códigos x Tipo Escola x Estado',fontsize=20)

ax = df.filter(items=['NOTA_LING_CODIGOS','TIPO_ESCOLA','UF_PROVA'])\
.groupby(['TIPO_ESCOLA','UF_PROVA'])\
.mean().sort_values(by='NOTA_LING_CODIGOS',ascending=False)\
.unstack().plot(ax=ax,fontsize=20, colormap='hsv', grid=True, marker='o' ) #Se
                                                                            #au
                                                                            #mu
                                                                            #co
                                                                            #ma


#Modifica as legendas dos estados
handles, labels = ax.get_legend_handles_labels()
import re
edited_labels = [re.search(',\s(.+?)\)',label).group(1) for label in labels]
ax.legend(edited_labels, bbox_to_anchor=(1,1), loc=0, borderaxespad= 1)

print()
fig,ax = plt.subplots(figsize=(16,8))
plt.suptitle('Nota Linguagens e Códigos x Tipo Escola x Região',fontsize=20)

ax = df.filter(items=['NOTA_LING_CODIGOS','TIPO_ESCOLA','Regiao'])\
.groupby(['TIPO_ESCOLA','Regiao'])\
.mean().sort_values(by='NOTA_LING_CODIGOS',ascending=False)\
.unstack().plot(ax=ax,fontsize=20, colormap='hsv', grid=True, marker='o' ) #Se
                                                                            #au
                                                                            #mu
                                                                            #co
                                                                            #ma


#Modifica as legendas dos estados
handles, labels = ax.get_legend_handles_labels()
import re
edited_labels = [re.search(',\s(.+?)\)',label).group(1) for label in labels]
ax.legend(edited_labels, bbox_to_anchor=(1,1), loc=0, borderaxespad= 1)

print()
```



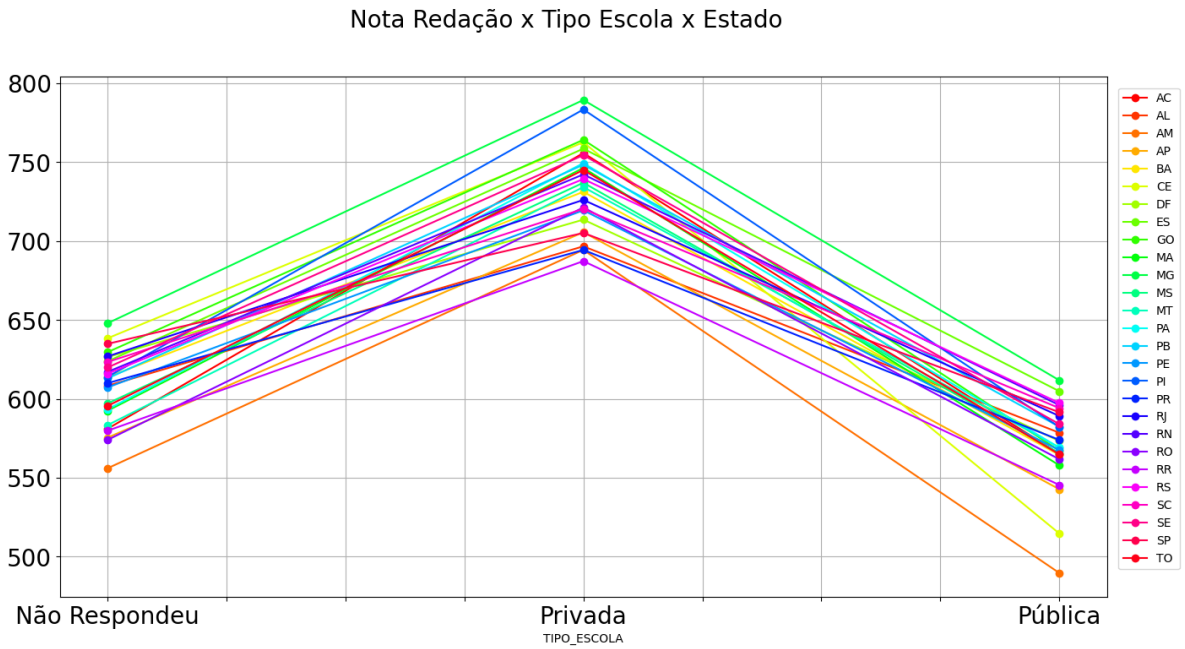

Nota Linguagens e Códigos x Tipo Escola x Região
Centro-Oeste
Nordeste
Norte
SUL
Sudeste
540
520
500
480
460
Não Respondeu
Privada
Pública
TIPO_ESCOLA

In [37]:
```
fig,ax = plt.subplots(figsize=(16,8))
plt.suptitle('Nota Matemática x Tipo Escola x Estado',fontsize=20)

ax = df.filter(items=['NOTA_MATEMATICA','TIPO_ESCOLA','UF_PROVA'])\
.groupby(['TIPO_ESCOLA','UF_PROVA'])\
.mean().sort_values(by='NOTA_MATEMATICA',ascending=False)\
.unstack().plot(ax=ax,fontsize=20, colormap='hsv', grid=True, marker='o' ) #Se
                                                                            #au
                                                                            #mu
                                                                            #co
                                                                            #ma


#Modifica as legendas dos estados
handles, labels = ax.get_legend_handles_labels()
import re
edited_labels = [re.search(',\s(.+?)\)',label).group(1) for label in labels]
ax.legend(edited_labels, bbox_to_anchor=(1,1), loc=0, borderaxespad= 1)

print()
fig,ax = plt.subplots(figsize=(16,8))
plt.suptitle('Nota Matemática x Tipo Escola x Região',fontsize=20)

ax = df.filter(items=['NOTA_MATEMATICA','TIPO_ESCOLA','Regiao'])\
.groupby(['TIPO_ESCOLA','Regiao'])\
.mean().sort_values(by='NOTA_MATEMATICA',ascending=False)\
.unstack().plot(ax=ax,fontsize=20, colormap='hsv', grid=True, marker='o' ) #Se
                                                                            #au
                                                                            #mu
                                                                            #co
                                                                            #ma


#Modifica as legendas dos estados
handles, labels = ax.get_legend_handles_labels()
import re
edited_labels = [re.search(',\s(.+?)\)',label).group(1) for label in labels]
ax.legend(edited_labels, bbox_to_anchor=(1,1), loc=0, borderaxespad= 1)

print()
```

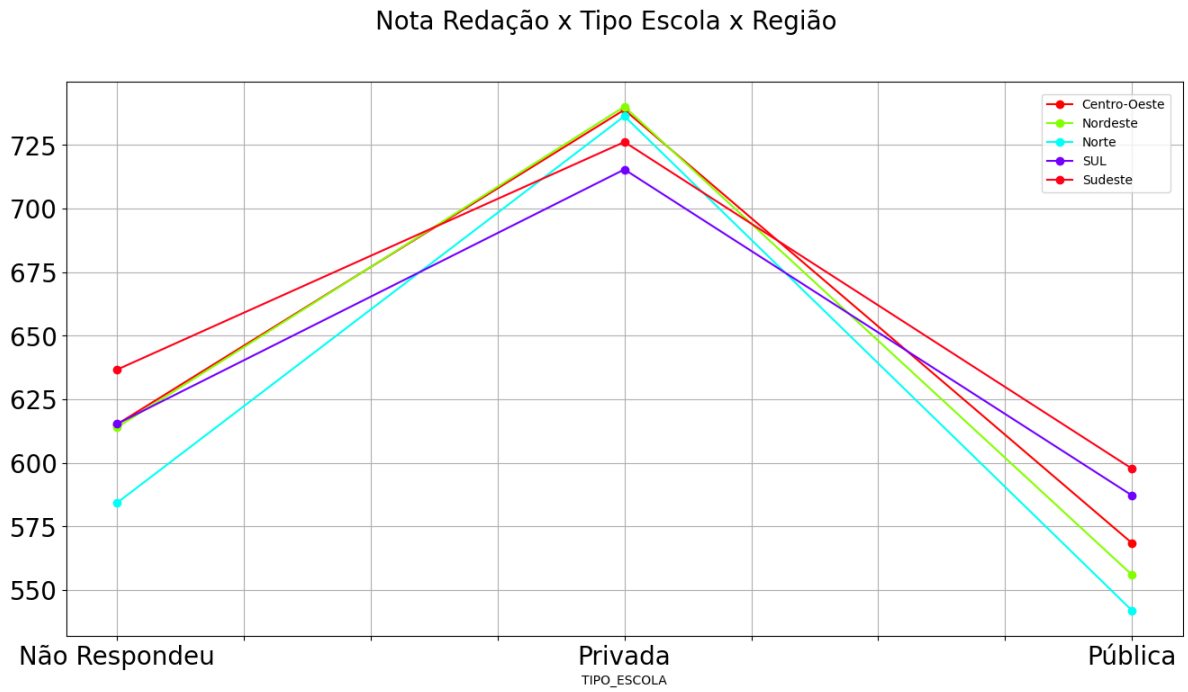



500
SC
Nota Matemática x Tipo Escola x Região
620
600
580
560
540
520
500
480
Centro-Oeste
Nordeste
Norte
SUL
Sudeste
Não Respondeu
Privada
Pública
TIPO_ESCOLA

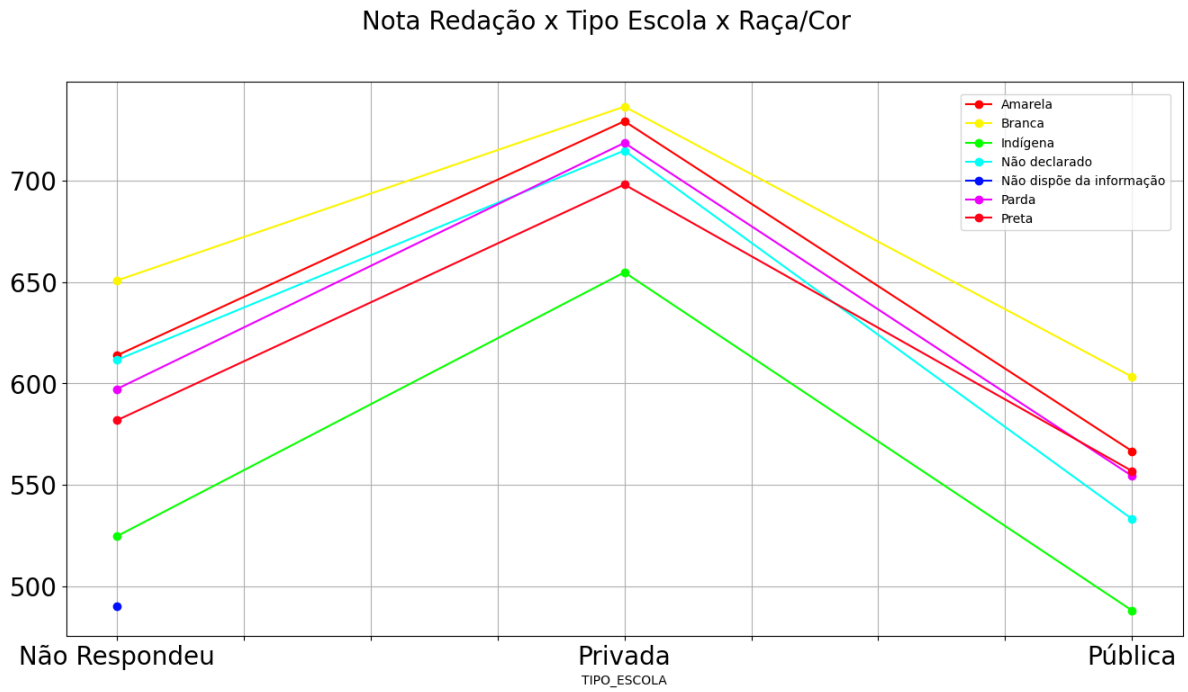
In [38]:
```
fig,ax = plt.subplots(figsize=(16,8))
plt.suptitle('Nota Redação x Tipo Escola x Estado',fontsize=20)

ax = df.filter(items=['NOTA_REDACAO','TIPO_ESCOLA','UF_PROVA'])\
.groupby(['TIPO_ESCOLA','UF_PROVA'])\
.mean().sort_values(by='NOTA_REDACAO',ascending=False)\
.unstack().plot(ax=ax,fontsize=20, colormap='hsv', grid=True, marker='o' ) #Se
                                                                            #au
                                                                            #mu
                                                                            #co
                                                                            #ma


#Modifica as legendas dos estados
handles, labels = ax.get_legend_handles_labels()
import re
edited_labels = [re.search(',\s(.+?)\)',label).group(1) for label in labels]
ax.legend(edited_labels, bbox_to_anchor=(1,1), loc=0, borderaxespad= 1)

print()
```



In [39]:
```
fig,ax = plt.subplots(figsize=(16,8))
plt.suptitle('Nota Redação x Tipo Escola x Região',fontsize=20)

ax = df.filter(items=['NOTA_REDACAO','TIPO_ESCOLA','Regiao'])\
.groupby(['TIPO_ESCOLA','Regiao'])\
.mean().sort_values(by='NOTA_REDACAO',ascending=False)\
.unstack().plot(ax=ax,fontsize=20, colormap='hsv', grid=True, marker='o' ) #Se
                                                                            #au
                                                                            #mu
                                                                            #co
                                                                            #ma


#Modifica as legendas dos estados
handles, labels = ax.get_legend_handles_labels()
import re
edited_labels = [re.search(',\s(.+?)\)',label).group(1) for label in labels]
ax.legend(edited_labels, bbox_to_anchor=(1,1), loc=0, borderaxespad= 1)

print()
```



In [40]:
```
fig,ax = plt.subplots(figsize=(16,8))
plt.suptitle('Nota Redação x Tipo Escola x Raça/Cor',fontsize=20)

ax = df.filter(items=['NOTA_REDACAO','TIPO_ESCOLA','COR_RACA'])\
.groupby(['TIPO_ESCOLA','COR_RACA'])\
.mean().sort_values(by='NOTA_REDACAO',ascending=False)\
.unstack().plot(ax=ax,fontsize=20, colormap='hsv', grid=True, marker='o' ) #Se
                                                                           #au
                                                                           #mu
                                                                           #co
                                                                           #ma


#Modifica as legendas dos estados
handles, labels = ax.get_legend_handles_labels()
import re
edited_labels = [re.search(',\s(.+?)\)',label).group(1) for label in labels]
ax.legend(edited_labels, bbox_to_anchor=(1,1), loc=0, borderaxespad= 1)

print()
```



In [41]:
```
#df_nota = df.NU_NOTA_REDACAO.value_counts()
```

Análise dos dados - Socioeconômico

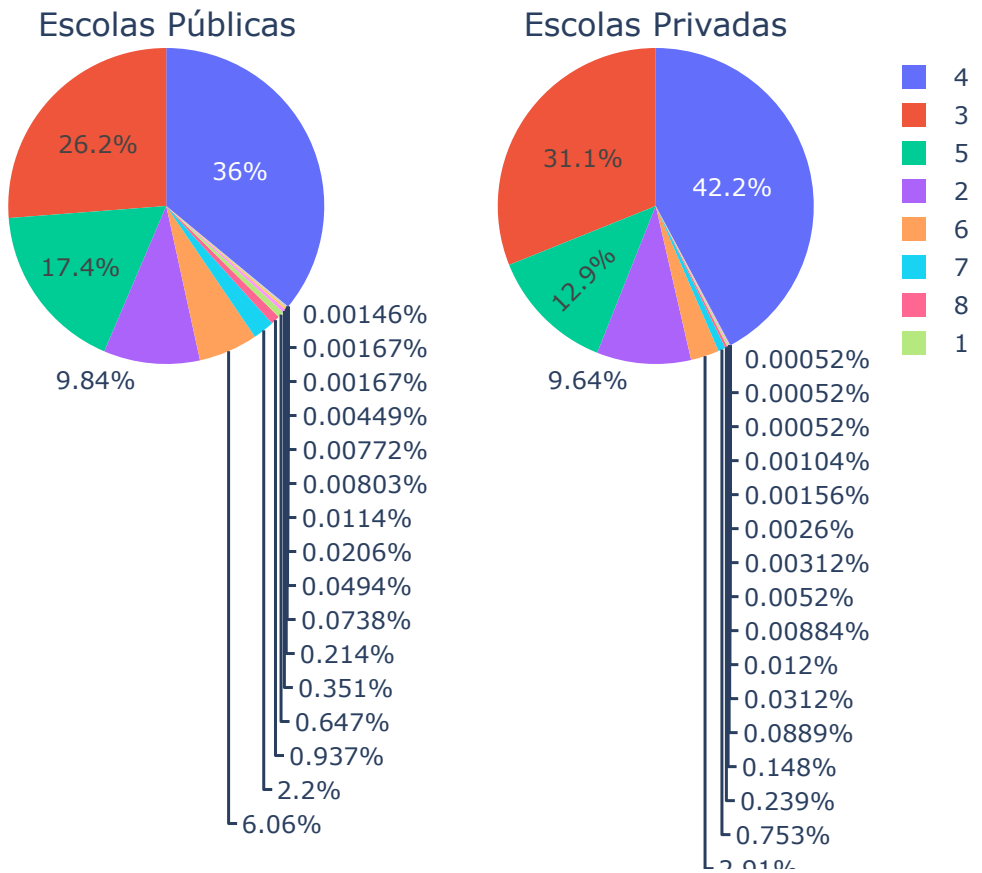
Voltar ao índice

Comparativo do questionário Socioeconômico entre os alunos de escolas públicas e privadas.

In [42]:
```
import plotly.graph_objects as go
```

In [63]:
```
pessoas_pub = df.query("TIPO_ESCOLA == 'Pública'")["PESSOAS_POR_RESIDENCIA"].v
label_pub = pessoas_pub.index
value_pub = pessoas_pub.values

pessoas_priv = df.query("TIPO_ESCOLA == 'Privada'")["PESSOAS_POR_RESIDENCIA"].
label_priv = pessoas_priv.index
value_priv = pessoas_priv.values

grafico_pessoas = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'}, {'t
print("Quantas pessoas moram na mesma residência?")
grafico_pessoas.add_trace(go.Pie(labels=label_pub, values=value_pub), 1, 1)
grafico_pessoas.add_trace(go.Pie(labels=label_priv, values=value_priv), 1, 2)
```

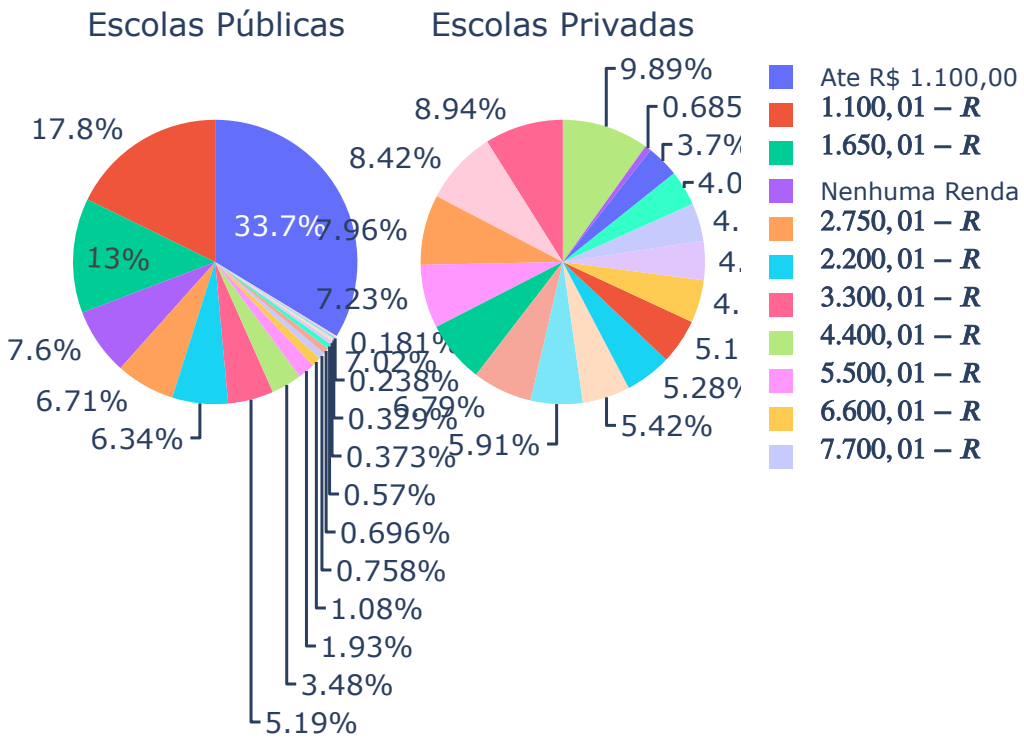
```
Quantas pessoas moram na mesma residência?
```



Nesse comparativo é possível verificar o percentual de pessoas por residência, entre alunos de escolas públicas e privadas.

In [55]:
```
renda_pub = df.query("TIPO_ESCOLA == 'Pública'")["RENDA_MENSAL_FAMILIA"].value
label_pub = renda_pub.index
value_pub = renda_pub.values

renda_priv = df.query("TIPO_ESCOLA == 'Privada'")["RENDA_MENSAL_FAMILIA"].valu
label_priv = renda_priv.index
value_priv = renda_priv.values

grafico_renda = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'},{'type

grafico_renda.add_trace(go.Pie(labels=label_pub, values=value_pub), 1, 1)
grafico_renda.add_trace(go.Pie(labels=label_priv, values=value_priv), 1, 2)
print("Renda mensal da família?")
grafico_renda.update_layout(uniformtext_minsize=14, uniformtext_mode='hide')
```

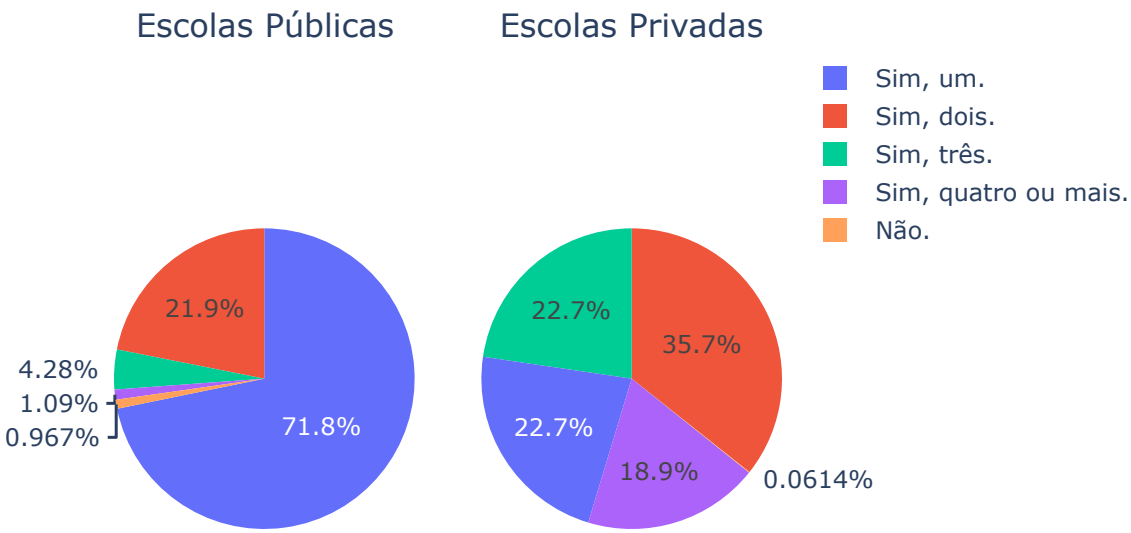
```
Renda mensal da família?
```




```
De acordo, com a renda mensal das famílias, nota-se que grande parte dos aluno
de escolas públicas a renda é até de R$ 1.000 e para alunos de escolas privada
```

In [56]:
```
escola_pub = df.query("TIPO_ESCOLA == 'Pública'")["RESID_TEM_BANHEIRO"].value_
label_pub = escola_pub.index
value_pub = escola_pub.values
escola_priv = df.query("TIPO_ESCOLA == 'Privada'")["RESID_TEM_BANHEIRO"].value
label_priv = escola_priv.index
value_priv = escola_priv.values

grafico_banheiro = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'}, {'
print("Residência tem Banheiro?")
grafico_banheiro.add_trace(go.Pie(labels=label_pub, values=value_pub), 1, 1)
grafico_banheiro.add_trace(go.Pie(labels=label_priv, values=value_priv), 1, 2)
```

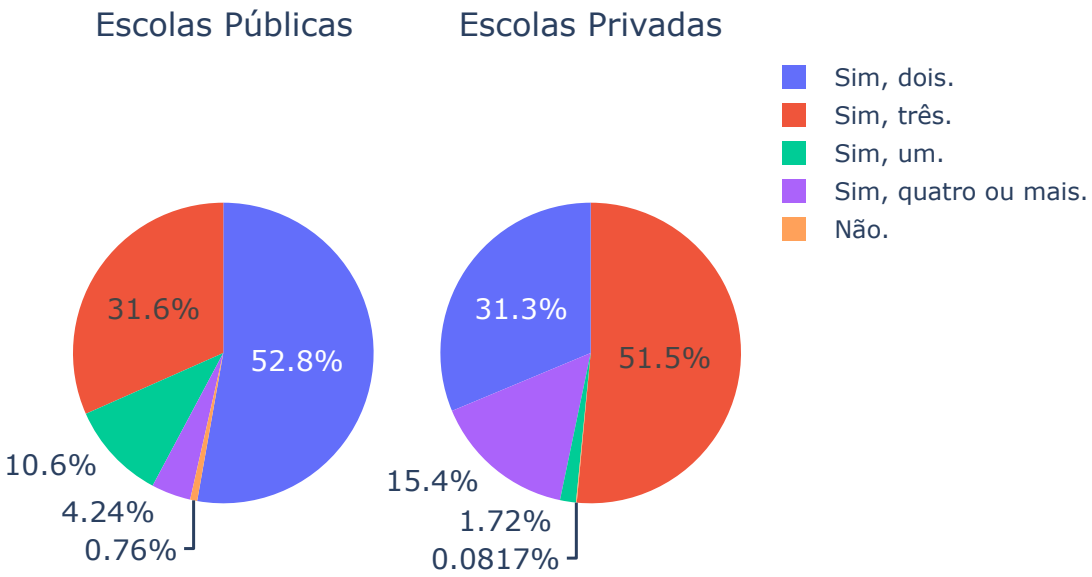
```
Residência tem Banheiro?
```



Embora, seja um percentual pequeno, alunos de escola pública não constam banheiros em suas redidências, já para alunos de escola privada, esse percentual é quase inexistente.

In [57]:
```
quartos_pub = df.query("TIPO_ESCOLA == 'Pública'")["RESID_TEM_QUARTOS"].value_
label_pub = quartos_pub.index
value_pub = quartos_pub.values

quartos_priv = df.query("TIPO_ESCOLA == 'Privada'")["RESID_TEM_QUARTOS"].value
label_priv = quartos_priv.index
value_priv = quartos_priv.values

grafico_quartos = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'},{'ty

grafico_quartos.add_trace(go.Pie(labels=label_pub, values=value_pub), 1, 1)
grafico_quartos.add_trace(go.Pie(labels=label_priv, values=value_priv), 1, 2)
print("A residência tem quartos?")
grafico_quartos.update_layout(uniformtext_minsize=14, uniformtext_mode='hide')
```

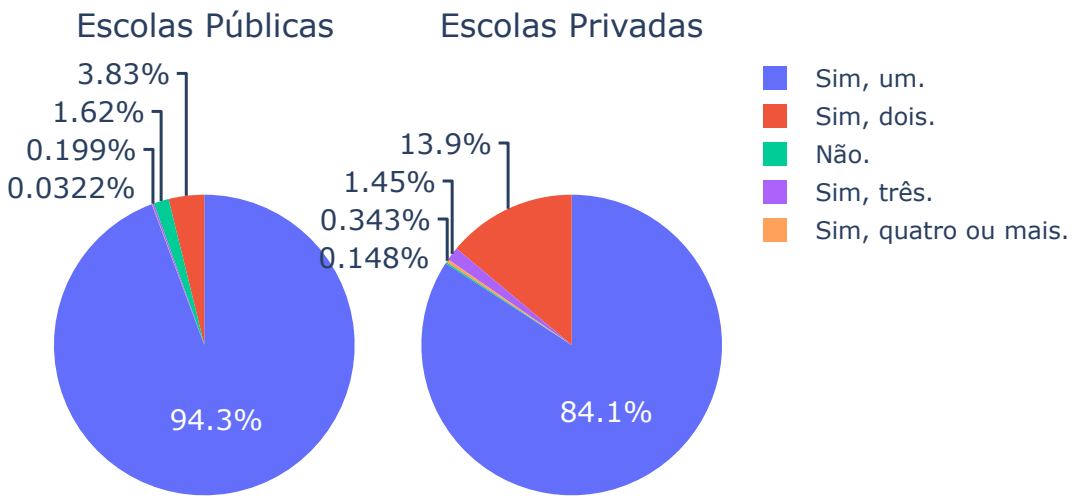
```
A residência tem quartos?
```



Neste gráfico, também é possível deduzir que 0.76% dos alunos de escola pública, não possui quarto, para alunos de escola privada, esse percentual é quase inexistente.

In [58]:
```
geladeira_pub = df.query("TIPO_ESCOLA == 'Pública'")["RESID_TEM_GELADEIRA"].va
label_pub = geladeira_pub.index
value_pub = geladeira_pub.values

geladeira_priv = df.query("TIPO_ESCOLA == 'Privada'")["RESID_TEM_GELADEIRA"].v
label_priv = geladeira_priv.index
value_priv = geladeira_priv.values

grafico_geladeira = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'},{'

grafico_geladeira.add_trace(go.Pie(labels=label_pub, values=value_pub), 1, 1)
grafico_geladeira.add_trace(go.Pie(labels=label_priv, values=value_priv), 1, 2
print("A residência tem geladeira?")
grafico_geladeira.update_layout(uniformtext_minsize=14, uniformtext_mode='hide
```

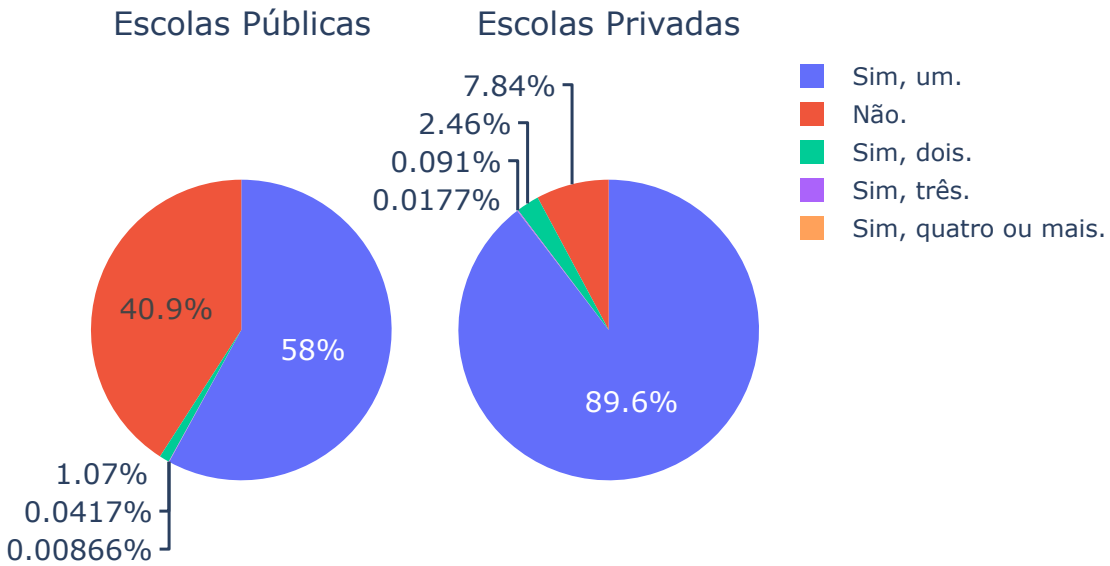
```
A residência tem geladeira?
```



Em ambas as residências, tanto para alunos de escola pública e privada, possuem apenas uma geladeira em casa. No entato, há uma pequena porcentagem de alunos de escola pública que não possuem geladeira em suas residências.

In [59]:
```
lavroupas_pub = df.query("TIPO_ESCOLA == 'Pública'")["RESID_TEM_LAVA_ROUPAS"].
label_pub = lavroupas_pub.index
value_pub = lavroupas_pub.values

lavroupas_priv = df.query("TIPO_ESCOLA == 'Privada'")["RESID_TEM_LAVA_ROUPAS"]
label_priv = lavroupas_priv.index
value_priv = lavroupas_priv.values

grafico_lavroupas = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'},{'

grafico_lavroupas.add_trace(go.Pie(labels=label_pub, values=value_pub), 1, 1)
grafico_lavroupas.add_trace(go.Pie(labels=label_priv, values=value_priv), 1, 2
print("A residência tem lava-roupas")
grafico_lavroupas.update_layout(uniformtext_minsize=14, uniformtext_mode='hide
```

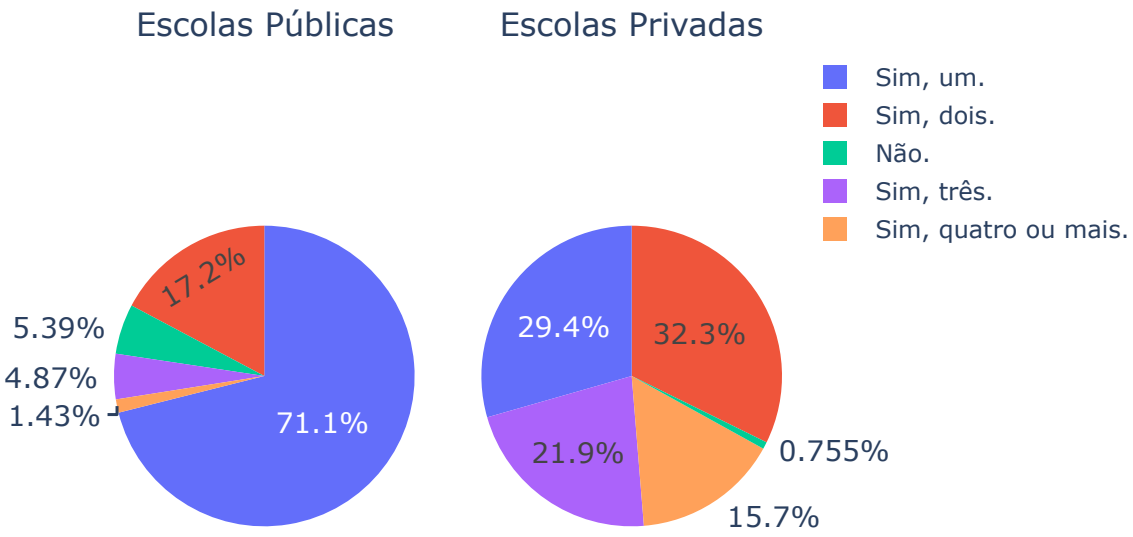
```
A residência tem lava-roupas
```



Neste gráfico, nota-se que um pouco mais da metade dos alunos de escola pública, possuem lava-roupas em suas residências, porém há uma porcentagem considerável que não possuem. Já para alunos de escolas particulares, alunos que não possuem lava-roupas em suas residências é um valor pequeno.

In [60]:
```
tv_pub = df.query("TIPO_ESCOLA == 'Pública'")["RESID_TEM_TV_CORES"].value_coun
label_pub = tv_pub.index
value_pub = tv_pub.values

tv_priv = df.query("TIPO_ESCOLA == 'Privada'")["RESID_TEM_TV_CORES"].value_cou
label_priv = tv_priv.index
value_priv = tv_priv.values

grafico_tv = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'},{'type':'

grafico_tv.add_trace(go.Pie(labels=label_pub, values=value_pub), 1, 1)
grafico_tv.add_trace(go.Pie(labels=label_priv, values=value_priv), 1, 2)
print("A residência tem TV em Cores?")
grafico_tv.update_layout(uniformtext_minsize=14, uniformtext_mode='hide')
```

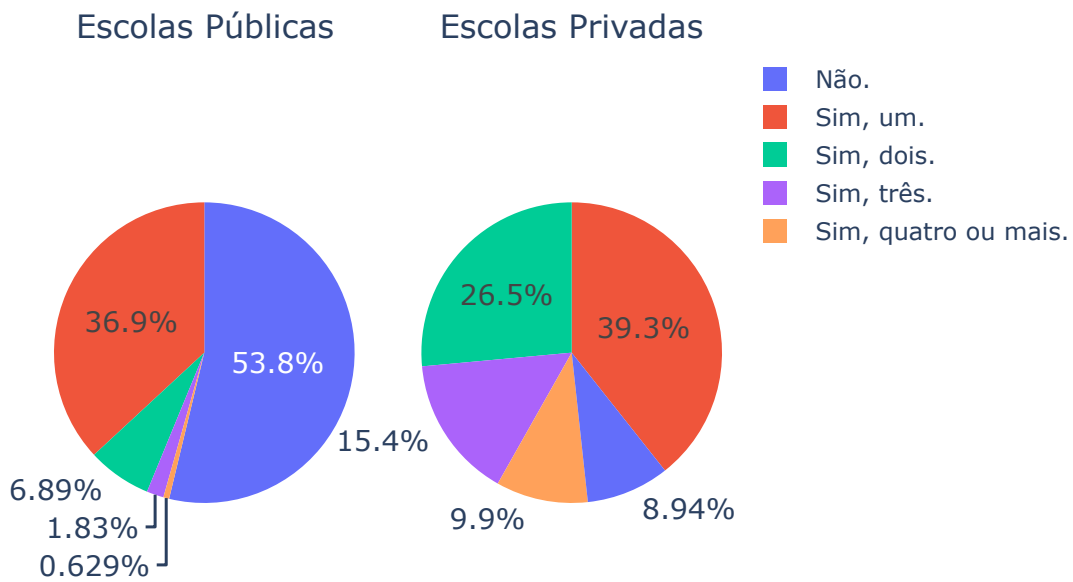
```
A residência tem TV em Cores?
```



Com relação a televisão, percebe-se que há uma porcentagem pequena para ambos alunos de escola pública e privada, que não possuem televisor em cores em suas residências. O comparativo também vale, para a questão de mais de um televisor, que ambas porcentagens, embora diferentes, apresentam maior proporção.

In [61]:
```
pc_pub = df.query("TIPO_ESCOLA == 'Pública'")["RESID_TEM_COMPUTADOR"].value_co
label_pub = pc_pub.index
value_pub = pc_pub.values

pc_priv = df.query("TIPO_ESCOLA == 'Privada'")["RESID_TEM_COMPUTADOR"].value_c
label_priv = pc_priv.index
value_priv = pc_priv.values

grafico_pc = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'},{'type':'

grafico_pc.add_trace(go.Pie(labels=label_pub, values=value_pub), 1, 1)
grafico_pc.add_trace(go.Pie(labels=label_priv, values=value_priv), 1, 2)

print("A residência tem Computador?")
grafico_pc.update_layout(uniformtext_minsize=14, uniformtext_mode='hide')
```

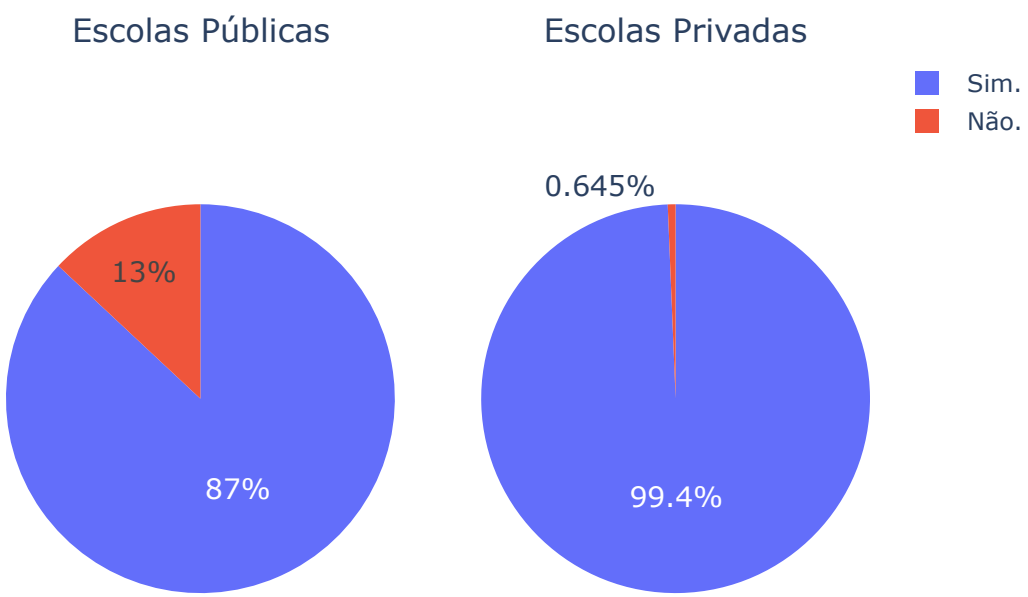
```
A residência tem Computador?
```



Nota-se, uma porcentagem maior que 50% dos alunos de escolas públicas, não possuírem computadores em suas residência. Já os alunos de escolas particulares, pelo menos 39,3 % possuem um computador, e 26,5% mais de um.

In [62]:
```
net_pub = df.query("TIPO_ESCOLA == 'Pública'")["RESID_TEM_INTERNET"].value_cou
label_pub = net_pub.index
value_pub = net_pub.values

net_priv = df.query("TIPO_ESCOLA == 'Privada'")["RESID_TEM_INTERNET"].value_co
label_priv = net_priv.index
value_priv = net_priv.values

grafico_net = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'},{'type':

grafico_net.add_trace(go.Pie(labels=label_pub, values=value_pub), 1, 1)
grafico_net.add_trace(go.Pie(labels=label_priv, values=value_priv), 1, 2)

print("A residência tem internet?")
grafico_net.update_layout(uniformtext_minsize=14, uniformtext_mode='hide')
```

```
A residência tem internet?
```



Alunos de escolas privadas, tem quase 100% internet em suas residências. Já 13% de alunos de escola pública, não possuem internet em suas residências.

Conclusão

Voltar ao índice

Ao final da análise dos dados coletados, foi possível realizar um comparativo, por estados e regiões, entre alunos de escolas públicas e privadas.Foram levantados, dados de teor socioeconômico e índice de notas de redação, afim de maior compreensão e conhecimento, para fins de pesquisas e estudos mais aprofundados, em relação a educação no país.