In [61]:
```
from pyspark.sql import SparkSession

spark = SparkSession.builder \
    .appName("Exemplo PySpark") \
    .getOrCreate()
```

In [62]:
```
df = spark.read.csv("./dataset/MICRODADOS_ENEM_2022.csv", header=True,inferSchema=True,
salvador_df = df.filter(df['NO_MUNICIPIO_PROVA'] == 'Salvador')
```

In [63]:
```
n_participantes_salvador = salvador_df.count()
n_participantes_salvador
```

Out[63]: 54604

In [64]:
```
import pandas as pd
ausencias_redacao_df = salvador_df.filter(salvador_df['NU_NOTA_REDACAO'].isNull())
n_ausencias_redacao = ausencias_redacao_df.count()
```

In [65]:
```
n_ausencias_redacao
```

Out[65]: 15818

In [66]:
```
zeros_redacao_df = salvador_df.filter(salvador_df['NU_NOTA_REDACAO'] == 0)
n_zeros_redacao = zeros_redacao_df.count()
```

In [67]:
```
n_zeros_redacao
```

Out[67]: 1565

In [68]:
```
categorias = ['<b>Não zerou</b>','<b>Ausentes</b>', '<b>Zeros</b>']
valores = [n_participantes_salvador-n_ausencias_redacao-n_zeros_redacao, n_ausencias_re
import plotly.express as px
fig = px.pie(values = valores, names= categorias, title = "Redação ENEM 2022 <b>Salvado
fig.add_annotation(
    text=f"<b>Total</b>: {n_participantes_salvador} inscritos",
    x=0.95,
    y=0.05,
    showarrow=False,
    font_size=14,
    xanchor="right",  # Alinhar a anotação à direita
    yanchor="bottom"  # Alinhar a anotação na parte inferior
)
fig.show()
```

In [69]:
```
salvador_df_sem_linhas_nulas = salvador_df.dropna(subset = ['NU_NOTA_REDACAO','NU_NOTA_
                                                            'NU_NOTA_CH','NU_NOTA_CN','
```

In [70]:
```
from pyspark.sql.functions import avg
import numpy as np
salvador_df_sem_linhas_nulas_media = np.round(salvador_df_sem_linhas_nulas.agg(avg('NU_
media_redacao = np.round(salvador_df_sem_linhas_nulas_media,2)
media_mt = np.round(salvador_df_sem_linhas_nulas.agg(avg('NU_NOTA_MT')).collect()[0][0]
media_cn = np.round(salvador_df_sem_linhas_nulas.agg(avg('NU_NOTA_CN')).collect()[0][0]
media_ch = np.round(salvador_df_sem_linhas_nulas.agg(avg('NU_NOTA_CH')).collect()[0][0]
media_lc = np.round(salvador_df_sem_linhas_nulas.agg(avg('NU_NOTA_LC')).collect()[0][0]
```

In [71]:
```
fig = px.bar(x=['Redação', 'Matemática', 'Ciências humanas', 'Ling e Códigos', 'Ciência
             y= [media_redacao,media_mt,media_ch,media_lc,media_cn], color = ['Redaçã
fig.update_traces(text=[media_redacao,media_mt,media_ch,media_lc,media_cn], textpositio
fig.update_layout(xaxis_title='<b>Áreas</b>', yaxis_title='Médias', title='<b>Médias</b
fig
```

In [72]:
```
from pyspark.sql.functions import col, when
salvador_df_sem_linhas_nulas = salvador_df_sem_linhas_nulas.withColumn("TP_COR_RACA", w
                    .when(col("TP_COR_RACA") == 2, "Preta")
                    .when(col("TP_COR_RACA") == 3, "Parda")
                    .when(col("TP_COR_RACA") == 4, "Amarela")
                    .when(col("TP_COR_RACA") == 5, "Indígena")
                    .when(col("TP_COR_RACA") == 6, "Não dispõe da informação")
                    .when(col("TP_COR_RACA") == 0, "Não Declarada")
                    .otherwise(col("TP_COR_RACA")))
```

In [73]:
```
fig = px.box(salvador_df_sem_linhas_nulas, x = 'TP_COR_RACA' , y = 'NU_NOTA_REDACAO',
             color = 'TP_COR_RACA', title = "BOXPLOT: Notas redação x Raça - <b>Salvad
fig.update_layout(xaxis_title = "<b>Raça</b>", yaxis_title = "Nota Redação" , legend_ti
#fig.update_traces(showlegend=False)
```

In [74]:
```
#valores_nulos_tratados = salvador_df.fillna({'NU_NOTA_REDACAO': media_redacao, 'NU_NOT
#                                              'NU_NOTA_CN' : media_cn, 'NU_NOTA_LC' :
#valores_nulos_tratados.show(10)
#df_valores_nulos_tratados = valores_nulos_tratados.toPandas()
```

In [75]:
```
salvador_df.createOrReplaceTempView("table_faixa_etaria")
faixa_etaria_df = spark.sql("SELECT TP_FAIXA_ETARIA,AVG(NU_NOTA_REDACAO) AS MEDIA_NOTA_
```

```
faixa_etaria_df = faixa_etaria_df.withColumn('TP_FAIXA_ETARIA', when(col('TP_FAIXA_ETAR
.when(col('TP_FAIXA_ETARIA') == 2,      '17')
.when(col('TP_FAIXA_ETARIA') == 3,      '18')
.when(col('TP_FAIXA_ETARIA') == 4,      '19')
.when(col('TP_FAIXA_ETARIA') == 5,      '20')
.when(col('TP_FAIXA_ETARIA') == 6,      '21')
.when(col('TP_FAIXA_ETARIA') == 7,      '22')
.when(col('TP_FAIXA_ETARIA') == 8,      '23')
.when(col('TP_FAIXA_ETARIA') == 9,      '24')
.when(col('TP_FAIXA_ETARIA') == 10,     '25')
.when(col('TP_FAIXA_ETARIA') == 11,     '26 - 30')
.when(col('TP_FAIXA_ETARIA') == 12,     '31 - 35')
.when(col('TP_FAIXA_ETARIA') == 13,     '36 - 40')
.when(col('TP_FAIXA_ETARIA') == 14,     '41 - 45')
.when(col('TP_FAIXA_ETARIA') == 15,     '46 - 50')
.when(col('TP_FAIXA_ETARIA') == 16,     '51 - 55')
.when(col('TP_FAIXA_ETARIA') == 17,     '56 - 60')
.when(col('TP_FAIXA_ETARIA') == 18,     '61 - 65')
.when(col('TP_FAIXA_ETARIA') == 19,     '66 - 70')
.when(col('TP_FAIXA_ETARIA') == 20,     '> 70')
)
```

In [76]:
```
faixa_etaria_df.createOrReplaceTempView("table_faixa_etaria")
faixa_etaria_df = spark.sql("SELECT TP_FAIXA_ETARIA, MEDIA_NOTA_REDACAO FROM table_faix
```

In [77]:
```
fig = px.bar(faixa_etaria_df, x = 'TP_FAIXA_ETARIA', y = "MEDIA_NOTA_REDACAO", title =
fig.update_layout(xaxis_title = '<b>Faixa Etária</b>', yaxis_title = "Média Redação")
```

In [78]:
```
salvador_df.createOrReplaceTempView("table_renda")
renda_df = spark.sql("SELECT Q006,ROUND(AVG(NU_NOTA_REDACAO), 2) AS MEDIA_NOTA_REDACAO
renda_df = renda_df.withColumn('Q006',
                               when(col('Q006')== 'A',   'Nenhuma Renda')
.when(col('Q006')== 'B',         '<= R$ 1.212,00')
.when(col('Q006')== 'C',         'R$ 1.212,01 - R$ 1.818,00')
.when(col('Q006')== 'D',         'R$ 1.818,01 - R$ 2.424,00')
.when(col('Q006')== 'E',         'R$ 2.424,01 - R$ 3.030,00')
.when(col('Q006')== 'F',         'R$ 3.030,01 - R$ 3.636,00')
.when(col('Q006')== 'G',         'R$ 3.636,01 - R$ 4.848,00')
.when(col('Q006')== 'H',         'R$ 4.848,01 - R$ 6.060,00')
.when(col('Q006')== 'I',         'R$ 6.060,01 - R$ 7.272,00')
.when(col('Q006')== 'J',         'R$ 7.272,01 - R$ 8.484,00')
.when(col('Q006')== 'K',         'R$ 8.484,01 - R$ 9.696,00')
.when(col('Q006')== 'L',         'R$ 9.696,01 - R$ 10.908,00')
.when(col('Q006')== 'M',         'R$ 10.908,01 - R$ 12.120,00')
.when(col('Q006')== 'N',         'R$ 12.120,01 - R$ 14.544,00')
.when(col('Q006')== 'O',         'R$ 14.544,01 - R$ 18.180,00')
.when(col('Q006')== 'P',         'R$ 18.180,01 - R$ 24.240,00')
.when(col('Q006')== 'Q',          '> R$ 24.240,00')
                              )
```

In [79]:
```
fig = px.bar(renda_df, y = 'Q006', x = "MEDIA_NOTA_REDACAO", title = "MEDIA REDAÇÃO X R
             text="MEDIA_NOTA_REDACAO", color_discrete_sequence=['green'], opacity= 0.
fig.update_traces(textposition ='outside')
fig.update_layout(xaxis_title = '<b>Média Redação</b>',
                  yaxis_title = "<b>Renda Familiar</b>",
                  )
```

In [80]:
```
#df_valores_nulos_tratados.to_excel("df_enem_salvador_nulos_tratados.xlsx")
```

In [81]:
```
fig = px.histogram(salvador_df_sem_linhas_nulas , x = ['NU_NOTA_CH', 'NU_NOTA_LC', 'NU_
                    barmode='overlay', title = "Distribuição das Notas por Área - <b>Sa
fig.update_layout(
    xaxis_title='<b>Notas</b>',
    yaxis_title='Contagem',
    legend_title_text = "Áreas",
)
legendas_personalizadas = {
    'NU_NOTA_CH': 'Ciências Humanas',
    'NU_NOTA_LC': 'Linguagens e Códigos',
    'NU_NOTA_CN': 'Ciências da Natureza',
    'NU_NOTA_MT': 'Matemática',
}
for coluna in fig.data:
    coluna.name = legendas_personalizadas.get(coluna.name, coluna.name)
fig
```

In [82]:
```
fig = px.box(salvador_df_sem_linhas_nulas, y="NU_NOTA_REDACAO", x = 'TP_SEXO', color =
fig.update_layout(xaxis_title = "<b>Sexo</b>", yaxis_title = "Nota Redação" , legend_ti
fig.update_traces(opacity = 0.6)
fig.show()
```

In [83]:
```
media_area_escola_df = spark.sql("SELECT TP_ESCOLA,ROUND(AVG(NU_NOTA_REDACAO),2) AS MED
media_area_escola_df = media_area_escola_df.withColumn('TP_ESCOLA',
                            when(col('TP_ESCOLA')== '1',      'Não Respondeu')
                           .when(col('TP_ESCOLA')== '2',      'Pública')
                           .when(col('TP_ESCOLA')== '3',      'Privada'))
```

```
media_area_escola_df.show(10)
fig = px.bar(media_area_escola_df, x = 'TP_ESCOLA',  y =['MEDIA_REDACAO','MEDIA_MT','ME

legendas_personalizadas = {
    'MEDIA_CH': 'Ciências Humanas',
    'MEDIA_LC': 'Linguagens e Códigos',
    'MEDIA_CN': 'Ciências da Natureza',
    'MEDIA_MT': 'Matemática',
    'MEDIA_REDACAO': 'Redação'
}
for coluna in fig.data:
    coluna.name = legendas_personalizadas.get(coluna.name, coluna.name)
fig.update_layout(
    xaxis_title='<b>Tipo Escola</b>',
    yaxis_title='Média',
    legend_title_text = "Áreas",
)
```

```
+-------------+-------------+--------+--------+--------+--------+
|    TP_ESCOLA|MEDIA_REDACAO|MEDIA_MT|MEDIA_CH|MEDIA_CN|MEDIA_LC|
+-------------+-------------+--------+--------+--------+--------+
|Não Respondeu|       626.87|  545.23|  539.18|  504.34|  530.48|
|      Privada|       749.75|  593.31|  568.32|  532.73|  553.89|
|      Pública|       569.46|  506.94|  509.93|  477.92|  503.84|
+-------------+-------------+--------+--------+--------+--------+
```

In [84]:
```
df_treineiro_areas = spark.sql("SELECT IN_TREINEIRO,ROUND(AVG(NU_NOTA_REDACAO),2) AS ME
df_treineiro_areas = df_treineiro_areas.withColumn('IN_TREINEIRO',
                                when(col('IN_TREINEIRO')== '1',   'Treineiro')
                               .when(col('IN_TREINEIRO')== '0',   'Outros'))
```

In [85]:
```
df_treineiro_areas.show()
```

```
+------------+-------------+--------+--------+--------+--------+
|IN_TREINEIRO|MEDIA_REDACAO|MEDIA_MT|MEDIA_CH|MEDIA_CN|MEDIA_LC|
+------------+-------------+--------+--------+--------+--------+
|   Treineiro|       675.97|  565.49|  543.97|  503.17|  537.76|
|      Outros|       621.72|  539.78|  535.63|  502.61|  526.28|
+------------+-------------+--------+--------+--------+--------+
```

In [86]:
```
df_treineiro = salvador_df.filter(salvador_df["IN_TREINEIRO"] == 1)
df_outros = salvador_df.filter(salvador_df["IN_TREINEIRO"] == 0)
df_treineiro.count()
fig = px.pie( values= [df_treineiro.count(),salvador_df.count()], names = ['<b>Treineir
fig.add_annotation(
    text=f"<b>Total</b>: {n_participantes_salvador} inscritos",
    x=0.95,
    y=0.05,
    showarrow=False,
    font_size=14,
    xanchor="right",  # Alinhar a anotação à direita
    yanchor="bottom"  # Alinhar a anotação na parte inferior
)
```

In [87]:
```
fig = px.bar(df_treineiro_areas, x = 'IN_TREINEIRO',  y =['MEDIA_REDACAO','MEDIA_MT','M
legendas_personalizadas = {
    'MEDIA_CH': 'Ciências Humanas',
    'MEDIA_LC': 'Linguagens e Códigos',
    'MEDIA_CN': 'Ciências da Natureza',
    'MEDIA_MT': 'Matemática',
    'MEDIA_REDACAO': 'Redação'
}
for coluna in fig.data:
    coluna.name = legendas_personalizadas.get(coluna.name, coluna.name)
fig.update_layout(
    xaxis_title='<b>Tipo de Inscrito</b>',
    yaxis_title='Média',
    legend_title_text = "Áreas",
)
```

In [89]:
```
import nbformat
from nbconvert import HTMLExporter
from nbconvert.writers import FilesWriter

# Nome do arquivo .ipynb de entrada
input_notebook = 'teste.ipynb'

# Nome do arquivo HTML de saída
output_html = 'enem_codigo.html'

# Abra e leia o arquivo .ipynb
with open(input_notebook, 'r', encoding='utf-8') as notebook_file:
    notebook_content = nbformat.read(notebook_file, as_version=4)
```

```
# Configure o exportador HTML
html_exporter = HTMLExporter()
html_exporter.template_name = 'basic'  # Use o template "basic"

# Converta o notebook para HTML
(html_output, resources) = html_exporter.from_notebook_node(notebook_content)

# Salvar o HTML em um arquivo
writer = FilesWriter()
writer.build_directory = 'output_dir'  # Especifique o diretório de saída se desejar
writer.write(html_output, resources, notebook_name=output_html)
```

```
c:\Users\isnan\AppData\Local\Programs\Python\Python311\share\jupyter\nbconvert\templates
\base\display_priority.j2:32: UserWarning:

Your element with mimetype(s) dict_keys(['application/vnd.plotly.v1+json']) is not able
to be represented.
```

Out[89]: WindowsPath('output_dir/enem_codigo.html.html')