Análise de Dados de acões com PySpark, Matplotlib e Seaborn no GCP

Código rodando na cloud function api-acoes-gcp

In [ ]:

```
import requests
import logging
import os
from google.cloud import storage
from flask import Flask, Request
from datetime import datetime

# Usamos Flask para criarmos um servidor que irá gerenciar requisições HTTP.
app = Flask(__name__)

PROJECT_ID = "estudos-448118"
BUCKET_NAME = "dados_input"

# Responde requisições do tipo POST
@app.route("/", methods=["POST"])
def fechamento_gerdau(request: Request):
    """
    Função para baixar dados e salvar no GCS.
    """
    logging.basicConfig(level=logging.INFO)
    logging.info("Função iniciada")

    try:
        chave_api = os.environ.get("API_KEY")
        if not chave_api:
            logging.error("Erro: Variável de ambiente API_KEY não definida.")
            return "Erro: Variável de ambiente API_KEY não definida.", 500

        url = f'https://www.alphavantage.co/query?function=TIME_SERIES_DAILY&symbol=
        logging.info(f"URL da API: {url}")

        r = requests.get(url)
        r.raise_for_status()
        logging.info(f"Código de status da API: {r.status_code}")

        nome_arquivo = "api-acoes/fechamento_gerdau.csv"

        # Salvar o arquivo no GCS
        # blob é usado para manipular arquivos no GCS, veio através da biblioteca go
        storage_client = storage.Client()
        bucket = storage_client.bucket(BUCKET_NAME)
        blob = bucket.blob(nome_arquivo)
        blob.upload_from_string(r.text, content_type="text/csv")

        logging.info(f"Arquivo salvo com sucesso no GCS: gs://{BUCKET_NAME}/{nome_ar
        return f"Arquivo salvo no GCS: gs://{BUCKET_NAME}/{nome_arquivo}", 200

    except requests.exceptions.RequestException as e:
        logging.error(f"Erro ao fazer requisição para API: {e}")
        return f"Erro ao fazer requisição para API: {e}", 500

    except Exception as e:
        logging.error(f"Erro inesperado: {e}")
        return f"Erro inesperado: {e}", 500
```

In [1]:
```
from pyspark.sql import SparkSession
from pyspark.sql.functions import *
import pandas as pd
from io import StringIO
import time
```

In [2]:
```
##### Criando a sessão Spark
spark = SparkSession.builder.appName("ProcessamentoCSV").getOrCreate()

##### Lendo o arquivo CSV
df_spark = spark.read.csv("gs://dados_input/api-acoes/fechamento_gerdau.csv", header
df_spark.show()
```

```
25/02/24 21:33:22 WARN SparkSession: Using an existing Spark session; only runtime SQ
L configurations will take effect.
[Stage 2:>                                                          (0 + 1) / 1]
+----------+-----+-----+-----+-----+--------+
| timestamp| open| high|  low|close|  volume|
+----------+-----+-----+-----+-----+--------+
|2025-02-21| 16.6|16.77|16.09|16.22|18343500|
|2025-02-20|17.43|17.58|16.43|16.43|26719700|
|2025-02-19|17.32| 17.5|17.17|17.36| 8894400|
|2025-02-18|17.46|17.64|17.27|17.44|11477300|
|2025-02-17|17.55|17.71|17.36|17.37| 8584700|
|2025-02-14|17.58|17.72|17.34|17.58| 6849900|
|2025-02-13| 17.3| 17.5|17.13|17.45| 5875700|
|2025-02-12|17.54|17.66|17.22|17.37|14205900|
|2025-02-11|17.54|18.13|17.51|17.74|18838100|
|2025-02-10| 17.1|17.74| 17.0|17.57|20915200|
|2025-02-07|17.21|17.21|16.63| 16.7| 8164900|
|2025-02-06|16.94|17.16|16.79|17.09| 6745400|
|2025-02-05|17.11|17.11| 16.8| 16.9|10004300|
|2025-02-04|17.35|17.46|17.03|17.09|10929300|
|2025-02-03| 17.1| 17.6|17.06|17.48|13393700|
|2025-01-31|17.73|17.92|16.96|17.22|18725900|
|2025-01-30| 17.5|17.88|17.48|17.77| 8594400|
|2025-01-29| 17.7| 17.8|17.37|17.47|14489200|
|2025-01-28|17.66|17.88|17.55|17.67| 7886500|
|2025-01-27|17.53|17.78|17.46|17.75| 8151600|
+----------+-----+-----+-----+-----+--------+
only showing top 20 rows
```

In [3]:
```
df_spark.printSchema()
```

```
root
 |-- timestamp: date (nullable = true)
 |-- open: double (nullable = true)
 |-- high: double (nullable = true)
 |-- low: double (nullable = true)
 |-- close: double (nullable = true)
 |-- volume: integer (nullable = true)
```

In [4]:
```
df_spark = df_spark.withColumnRenamed("timestamp", "date")
df_spark.show()
```

```
[Stage 3:>                                                          (0 + 1) / 1]
```

```
+----------+-----+-----+-----+-----+--------+
|      date| open| high|  low|close|  volume|
+----------+-----+-----+-----+-----+--------+
|2025-02-21| 16.6|16.77|16.09|16.22|18343500|
|2025-02-20|17.43|17.58|16.43|16.43|26719700|
|2025-02-19|17.32| 17.5|17.17|17.36| 8894400|
|2025-02-18|17.46|17.64|17.27|17.44|11477300|
|2025-02-17|17.55|17.71|17.36|17.37| 8584700|
|2025-02-14|17.58|17.72|17.34|17.58| 6849900|
|2025-02-13| 17.3| 17.5|17.13|17.45| 5875700|
|2025-02-12|17.54|17.66|17.22|17.37|14205900|
|2025-02-11|17.54|18.13|17.51|17.74|18838100|
|2025-02-10| 17.1|17.74| 17.0|17.57|20915200|
|2025-02-07|17.21|17.21|16.63| 16.7| 8164900|
|2025-02-06|16.94|17.16|16.79|17.09| 6745400|
|2025-02-05|17.11|17.11| 16.8| 16.9|10004300|
|2025-02-04|17.35|17.46|17.03|17.09|10929300|
|2025-02-03| 17.1| 17.6|17.06|17.48|13393700|
|2025-01-31|17.73|17.92|16.96|17.22|18725900|
|2025-01-30| 17.5|17.88|17.48|17.77| 8594400|
|2025-01-29| 17.7| 17.8|17.37|17.47|14489200|
|2025-01-28|17.66|17.88|17.55|17.67| 7886500|
|2025-01-27|17.53|17.78|17.46|17.75| 8151600|
+----------+-----+-----+-----+-----+--------+
only showing top 20 rows
```

In [5]:

```python
# Analisando alguns dados de nosso df.

menor_preco = df_spark.orderBy(col("low").asc()).select("date", "low").first()

maior_preco = df_spark.orderBy(col("high").desc()).select("date", "high").first()

min_vol = df_spark.orderBy(col("volume").asc()).select("date", "volume" ).first()

max_vol = df_spark.orderBy(col("volume").desc()).select("date", "volume" ).first()

media_fechamento = df_spark.agg(round(avg("close"), 2).alias("media_close")).collect

# Exibir os resultados
print(f"Menor preço: {menor_preco['low']} em {menor_preco['date']}")
print(f"Maior preço: {maior_preco['high']} em {maior_preco['date']}")
print(f"Menor volume: {min_vol['volume']} em {min_vol['date']}")
print(f"Maior volume: {max_vol['volume']} em {max_vol['date']}")
print("Média de fechamento dos últimos 100 dias:", media_fechamento[0]["media_close"
```

```
[Stage 8:>                                                          (0 + 1) / 1]
Menor preço: 16.09 em 2025-02-21
Maior preço: 21.3 em 2024-12-09
Menor volume: 5875700 em 2025-02-13
Maior volume: 45316700 em 2024-11-06
Média de fechamento dos últimos 100 dias: 18.64
```

Vamos calcular a média móvel simples (SMA) dos últimos 10 dias, para tentarmos identificar alguma tendência.

In [6]:

```python
from pyspark.sql import functions as F
from pyspark.sql.window import Window
```

```
# Window.orderBy("date"): Organiza os dados pela data.
# rowsBetween(-9, 0): Define uma janela de 10 dias, incluindo o dia atual (índice 0)

df_spark = df_spark.withColumn("SMA_10",F.round(F.avg("close").over(Window.orderBy(F
)
df_spark.show()
```

```
25/02/24 21:36:21 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:36:21 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:36:21 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:36:23 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:36:23 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
[Stage 13:>                                                         (0 + 1) / 1]
+----------+-----+-----+-----+-----+--------+------+
|      date| open| high|  low|close|  volume|SMA_10|
+----------+-----+-----+-----+-----+--------+------+
|2025-02-21| 16.6|16.77|16.09|16.22|18343500| 16.22|
|2025-02-20|17.43|17.58|16.43|16.43|26719700| 16.33|
|2025-02-19|17.32| 17.5|17.17|17.36| 8894400| 16.67|
|2025-02-18|17.46|17.64|17.27|17.44|11477300| 16.86|
|2025-02-17|17.55|17.71|17.36|17.37| 8584700| 16.96|
|2025-02-14|17.58|17.72|17.34|17.58| 6849900| 17.07|
|2025-02-13| 17.3| 17.5|17.13|17.45| 5875700| 17.12|
|2025-02-12|17.54|17.66|17.22|17.37|14205900| 17.15|
|2025-02-11|17.54|18.13|17.51|17.74|18838100| 17.22|
|2025-02-10| 17.1|17.74| 17.0|17.57|20915200| 17.25|
|2025-02-07|17.21|17.21|16.63| 16.7| 8164900|  17.3|
|2025-02-06|16.94|17.16|16.79|17.09| 6745400| 17.37|
|2025-02-05|17.11|17.11| 16.8| 16.9|10004300| 17.32|
|2025-02-04|17.35|17.46|17.03|17.09|10929300| 17.29|
|2025-02-03| 17.1| 17.6|17.06|17.48|13393700|  17.3|
|2025-01-31|17.73|17.92|16.96|17.22|18725900| 17.26|
|2025-01-30| 17.5|17.88|17.48|17.77| 8594400| 17.29|
|2025-01-29| 17.7| 17.8|17.37|17.47|14489200|  17.3|
|2025-01-28|17.66|17.88|17.55|17.67| 7886500|  17.3|
|2025-01-27|17.53|17.78|17.46|17.75| 8151600| 17.31|
+----------+-----+-----+-----+-----+--------+------+
only showing top 20 rows
```

Comentário a respeito do aviso WARN WindowExec:

Ao usar a operação de janela 'Window' sem especificar uma partição, faz com que o PySpark mova todos os dados para uma única partição, algo que pode prejudicar o desempenho. Como nossa base de dados é pequena, vou ignorar esse aviso.

Importando as bibliotecas Matplotlib e Seaborn para construção dos gráficos

In [8]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:
```
# Converte o DataFrame Spark para Pandas
df_pandas = df_spark.select("date", "SMA_10").toPandas()
```

In [10]:

```
# Aqui é importante converter para Pandas e garantir que a coluna date seja datetime

df_plot = df_spark.toPandas()
df_plot["date"] = pd.to_datetime(df_plot["date"], format="%d-%m-%Y")

plt.figure(figsize=(12, 6))
sns.lineplot(data=df_plot, x="date", y="SMA_10", marker="o", linestyle="-", color="b

# Ajustes visuais
plt.xlabel("Data")
plt.ylabel("SMA 10")
plt.title("Média Móvel de 10 Períodos (SMA 10)")
plt.xticks(rotation=45)
plt.grid()

plt.show()
```

```
25/02/24 21:38:29 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:38:29 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:38:29 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:38:30 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:38:30 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
```

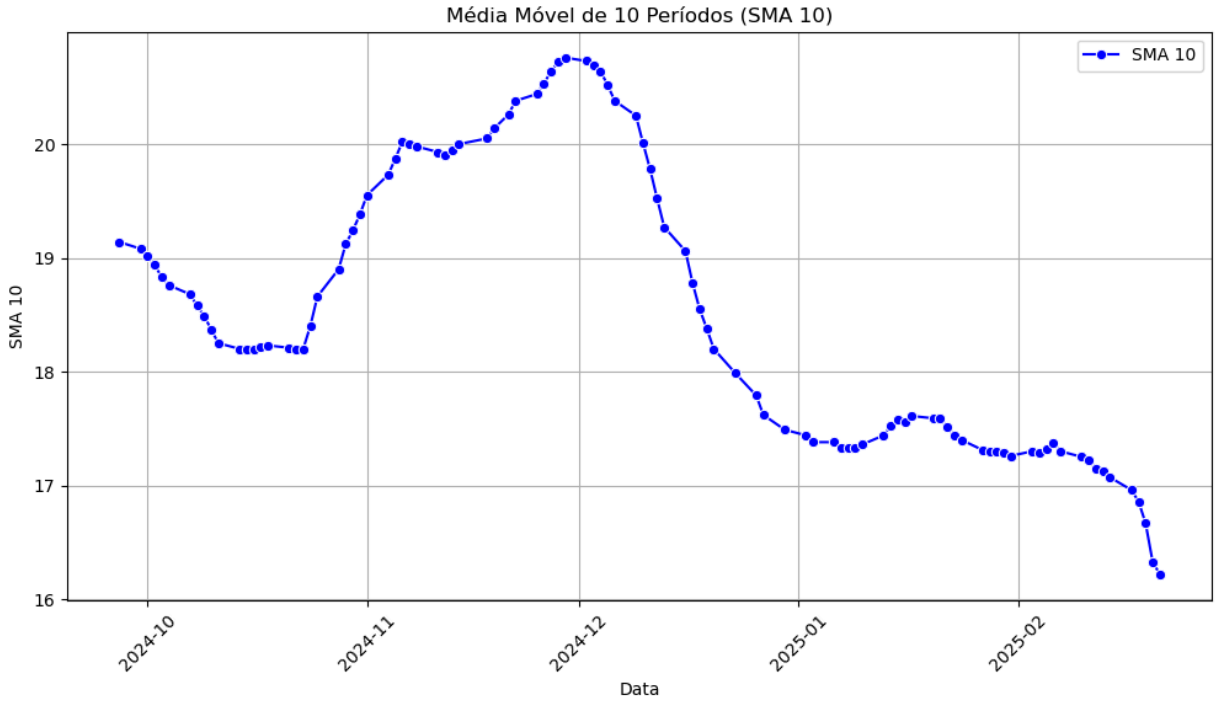
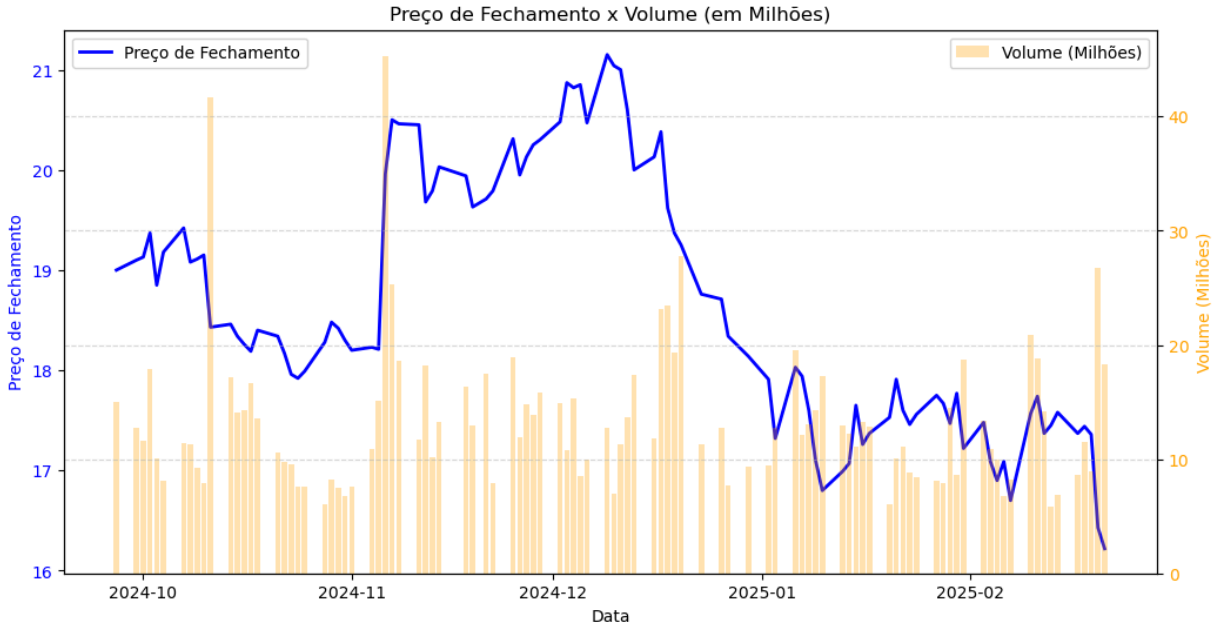


Gráfico Preço de fechamento x Volume (em milhões)

In [11]:

```
df_pandas2 = df_spark.select("date", "close", "volume").toPandas()

df_pandas2["date"] = pd.to_datetime(df_pandas2["date"], format="%d-%m-%Y")

# Ajustar escala do volume (milhões)
df_pandas2["volume_milhoes"] = df_pandas2["volume"] / 1_000_000

# Criar figura e eixo
```

```
fig, ax1 = plt.subplots(figsize=(12, 6))

# Plotar o preço de fechamento (linha)
ax1.plot(df_pandas2["date"], df_pandas2["close"], color="blue", label="Preço de Fech
ax1.set_xlabel("Data")
ax1.set_ylabel("Preço de Fechamento", color="blue")
ax1.tick_params(axis="y", labelcolor="blue")

# Criar segundo eixo para volume (barras)
ax2 = ax1.twinx()
ax2.bar(df_pandas2["date"], df_pandas2["volume_milhoes"], color="orange", alpha=0.3,
ax2.set_ylabel("Volume (Milhões)", color="orange")
ax2.tick_params(axis="y", labelcolor="orange")

# Adicionar legendas
ax1.legend(loc="upper left")
ax2.legend(loc="upper right")

# Layout
plt.title("Preço de Fechamento x Volume (em Milhões)")
plt.grid(True, linestyle="--", alpha=0.5)
plt.xticks(rotation=45)

plt.show()
```



Análise de Correlação entre Preço e Volume

In [13]:
```
df_spark.select(round(F.corr("volume", "close"), 2).alias("corr_volume_close")).show
```

```
[Stage 24:>                                                         (0 + 1) / 1]
+-----------------+
|corr_volume_close|
+-----------------+
|             0.17|
+-----------------+
```

O coeficiente de correlação de 0.17 entre o volume de negociações e o preço de fechamento (close) indica uma correlação fraca e positiva.

Interpretação:

- Próximo de 1: Forte correlação positiva (quando o volume aumenta, o preço tende a subir).
- Próximo de -1: Forte correlação negativa (quando o volume aumenta, o preço tende a cair).
- Próximo de 0: Pouca ou nenhuma correlação linear entre volume e preço.

Comparação entre Variação de Preço e Volume

Cálculo da Variação de Preço (diff_price):

- Objetivo: Verificar se grandes variações no preço são acompanhadas por aumento no volume de transações.
- Como fazer: Calcule a diferença percentual do preço e a diferença no volume e compare.
- diff_price: Mostra a variação percentual de preço de fechamento em relação ao preço de abertura no mesmo dia.

Cálculo da Variação de Volume (diff_volume):

- Objetivo: Calcular a variação percentual do volume de transações entre o dia atual e o dia anterior. A função lag é usada para pegar o valor do volume do dia anterior.
- diff_volume: Mostra a variação percentual do volume de transações de um dia para o próximo, comparando com o volume do dia anterior.

In [15]:

```
df_spark = df_spark.withColumn(
    "diff_price(%)", F.round((df_spark["close"] - df_spark["open"]) / df_spark["open
)

df_spark = df_spark.withColumn(
    "diff_volume(%)", F.round(
        (df_spark["volume"] - F.lag(df_spark["volume"]).over(Window.orderBy(F.col("d
        F.lag(df_spark["volume"]).over(Window.orderBy(F.col("date"))) * 100, 2
    )
)

df_spark = df_spark.orderBy(F.col("date").desc())
df_spark.show()
```

```
25/02/24 21:45:12 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:45:12 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:45:12 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:45:12 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:45:12 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:45:13 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:45:13 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
25/02/24 21:45:13 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
```

```
25/02/24 21:45:13 WARN WindowExec: No Partition Defined for Window operation! Moving
all data to a single partition, this can cause serious performance degradation.
+----------+-----+-----+-----+-----+--------+------+-------------+--------------+
|      date| open| high|  low|close|  volume|SMA_10|diff_price(%)|diff_volume(%)|
+----------+-----+-----+-----+-----+--------+------+-------------+--------------+
|2025-02-21| 16.6|16.77|16.09|16.22|18343500| 16.22|        -2.29|        -31.35|
|2025-02-20|17.43|17.58|16.43|16.43|26719700| 16.33|        -5.74|        200.41|
|2025-02-19|17.32| 17.5|17.17|17.36| 8894400| 16.67|         0.23|         -22.5|
|2025-02-18|17.46|17.64|17.27|17.44|11477300| 16.86|        -0.11|         33.69|
|2025-02-17|17.55|17.71|17.36|17.37| 8584700| 16.96|        -1.03|         25.33|
|2025-02-14|17.58|17.72|17.34|17.58| 6849900| 17.07|          0.0|         16.58|
|2025-02-13| 17.3| 17.5|17.13|17.45| 5875700| 17.12|         0.87|        -58.64|
|2025-02-12|17.54|17.66|17.22|17.37|14205900| 17.15|        -0.97|        -24.59|
|2025-02-11|17.54|18.13|17.51|17.74|18838100| 17.22|         1.14|         -9.93|
|2025-02-10| 17.1|17.74| 17.0|17.57|20915200| 17.25|         2.75|        156.16|
|2025-02-07|17.21|17.21|16.63| 16.7| 8164900|  17.3|        -2.96|         21.04|
|2025-02-06|16.94|17.16|16.79|17.09| 6745400| 17.37|         0.89|        -32.57|
|2025-02-05|17.11|17.11| 16.8| 16.9|10004300| 17.32|        -1.23|         -8.46|
|2025-02-04|17.35|17.46|17.03|17.09|10929300| 17.29|         -1.5|         -18.4|
|2025-02-03| 17.1| 17.6|17.06|17.48|13393700|  17.3|         2.22|        -28.47|
|2025-01-31|17.73|17.92|16.96|17.22|18725900| 17.26|        -2.88|        117.88|
|2025-01-30| 17.5|17.88|17.48|17.77| 8594400| 17.29|         1.54|        -40.68|
|2025-01-29| 17.7| 17.8|17.37|17.47|14489200|  17.3|         -1.3|         83.72|
|2025-01-28|17.66|17.88|17.55|17.67| 7886500|  17.3|         0.06|         -3.25|
|2025-01-27|17.53|17.78|17.46|17.75| 8151600| 17.31|         1.25|         -3.24|
+----------+-----+-----+-----+-----+--------+------+-------------+--------------+
only showing top 20 rows
```

Considerações sobre diff_price(%) e diff_volume(%)

- No dia 31-01-2025 ocorreu uma variação negativa do preço, e se observarmos a variação do volume com relação ao dia anterior, houve um aumento de 117%.
- Seguindo, no dia 03-02-2025, ocorre também uma variação relevante do preço, porém positiva, e para surpresa, com -28% de volume comparando com o dia anterior.
- No dia 10-02-2025, temos mais uma vez uma boa variação do preço positiva, acompanhado de 156% de volume a mais, quando comparado com o dia anterior.
- No dia 20-02-2025 tivemos uma forte variação do preço, acompanhado de um relevante aumento do volume de 200% com relação ao dia anterior.

Podemos concluir que, na maioria das vezes que o preço varia acima de 2%, seja positivamente ou negativamente, a variação de volume com relação ao dia anterior, aumenta, sendo que tem dias que o volume é consideravelmente maior do que no dia anterior.