A. Thu thập dữ liệu

1. Ngữ cảnh

Vàng là một kim loại quý và được sử dụng như một phương tiện để thanh toán hay lưu trữ giá trị. Trong quá khứ vàng đã được sử dụng như một đồng tiền vàng và hiện nay vẫn được sử dụng như một loại tiền tệ.

Vàng đóng vai trò quan trọng trong cuộc sống của chúng ta:

- Vàng có vai trò tiền tệ trên thế giới
- Vàng có ứng dụng chính trong ngành trang sức
- Vàng có ứng dụng trong ngành công nghiệp điện tử:
  - Vàng được sử dụng rộng rãi trong ngành công nghiệp điện tử để chế tạo các đầu nối điện có khả năng chống ăn mòn. Những đầu nối điện này được biết là được sử dụng trong nhiều thiết bị điện như máy tính. Thật thú vị khi lưu ý rằng một chiếc điện thoại thông minh trung bình được biết là chứa khoảng 50 miligam vàng. Cũng có thể lưu ý rằng vàng thường được sử dụng làm lớp phủ trong các đầu nối được sử dụng trong nhiều dạng dây quan trọng, chẳng hạn như dây USB, cáp video và cáp âm thanh
  - Nhờ khả năng chống ăn mòn, vàng cũng lý tưởng để sử dụng trong các điểm tiếp xúc điện. Vàng là một chất dẫn nhiệt tốt, không độc hại trong tự nhiên và là một trong những kim loại dễ uốn nhất mà con người biết đến. Do đó, các tiếp điểm của công tắc (thường rất dễ bị ăn mòn) thường được làm bằng vàng. Hơn nữa, các thiết bị bán dẫn thường được kết nối với các gói của chúng thông qua dây vàng cực tốt. Quá trình sử dụng dây vàng mịn để kết nối các thiết bị thường được gọi là liên kết dây.
- Có ứng dụng trong y học

⇒ Vàng có rất nhiều tác động lên đời sống xung quanh chúng ta. Chính vì thế cho nên hôm nay nhóm chúng em sẽ tìm hiểu về giá vàng.

2. Chủ đề

Giá vàng từ năm 1996 đến năm 2023.

3. License

CCO: Public Domain

4. Người ta thu thập dữ liệu này như thế nào?

B. Khám phá dữ liệu

1. Ý nghĩa của mỗi dòng

Mỗi dòng sẽ cung cấp số liệu liên quan đến cổ phiếu vàng trên sàn chứng khoán.

Không có dòng nào khác ý nghĩa với các dòng còn lại.

2. Ý nghĩa và kiểu dữ liệu của mỗi cột

Dataset này gồm 13 cột

Tên cột	Kiểu dữ liệu	Ý nghĩa
Date	String	Ngày của phiên giao dịch
Open	Decimal	Opening price (giá mở cửa) của **gold share**
High	Decimal	Highest price (giá cao nhất) của gold share
Low	Decimal	Lowest price (giá thấp nhất) của gold share
Close	Decimal	Closing price (giá đóng cửa) của gold share
WAP	Decimal	(Weighted Average Price) Là giá trung bình mà tại mức đó, các giao dịch cổ phiếu hoặc giao dịch tài sản được thực hiện
No. of Shares	Integer	Tổng số cổ phiếu được giao dịch trong ngày
No. of Trades	Integer	Tổng số giao dịch được thực hiện trong ngày
Total Turnover	Integer	Tổng giá trị của các giao dịch được thực hiện trong ngày
Deliverable Quantity	Integer	Tổng số cổ phiếu mà thực sự được chuyển giao cho người mua trong ngày
% Deli. Qty to Traded Qty	Decimal	Tỉ lệ phần trăm của số cổ phiếu mà thực sự được chuyển giao cho người mua trong ngày
Spread H-L	Decimal	Chênh lệch giữa highest price và lowest price của gold share trong ngày
Spread C-O	Decimal	Chênh lệch giữa closing price và opening price của gold share trong ngày

Chú thích

- Gold share: giá cổ phiếu vàng

3. Với mỗi cột, các giá trị được phân bố như thế nào?

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

Đọc dữ liệu

In [2]:
```
path = '../datasets/deccan gold mines ltd eod price.csv'
df = pd.read_csv(path)
df.head()
```

Out[2]:

	Date	Open	High	Low	Close	WAP	No. of Shares	No. of Trades	Total Turnover	Deliverable Quantity	% Deli. Qty to Traded Qty	Spread H-L	Sprea C-
0	13-04-23	50.47	50.47	47.86	49.15	49.35	122650	413	6053293	99475.0	81.10	2.61	-1.3
1	12-04-23	49.98	51.00	49.30	50.07	49.99	193359	417	9665903	166630.0	86.18	1.70	0.0
2	11-04-23	48.87	50.05	47.76	49.90	49.21	270964	509	13335488	208486.0	76.94	2.29	1.0
3	10-04-23	51.49	51.49	48.00	48.16	49.90	339774	517	16954583	280083.0	82.43	3.49	-3.3
4	06-04-23	48.40	49.04	47.65	49.04	48.80	245835	441	11996688	203526.0	82.79	1.39	0.6

In [3]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4812 entries, 0 to 4811
Data columns (total 13 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Date                       4812 non-null   object
 1   Open                       4812 non-null   float64
 2   High                       4812 non-null   float64
 3   Low                        4812 non-null   float64
 4   Close                      4812 non-null   float64
 5   WAP                        4812 non-null   float64
 6   No. of Shares              4812 non-null   int64
 7   No. of Trades              4812 non-null   int64
 8   Total Turnover             4812 non-null   int64
 9   Deliverable Quantity       4669 non-null   float64
 10  % Deli. Qty to Traded Qty  4669 non-null   float64
 11  Spread H-L                 4812 non-null   float64
 12  Spread C-O                 4812 non-null   float64
dtypes: float64(9), int64(3), object(1)
memory usage: 488.8+ KB
```

Nhận xét

- Cột Ngày(Date) có dạng object là phù hợp cho các bước tính toán sau.
- Các cột còn lại (Open, Close,...) đều ở dạng Numerical.
- Các cột đều có kiểu dữ liệu phù hợp nên không cần xử lý thêm.

Với mỗi cột dữ liệu:

- Tỉ lệ % (từ 0 đến 100) các giá trị NaN (giá trị bị thiếu)
- Giá trị min
- Giá trị lower quartile Q1 (phân vị 25)
- Giá trị median Q2 (phân vị 50)
- Giá trị upper quartile Q3 (phân vị 75)
- Giá trị max

In [4]:

```
def get_val(col):
    NaN_Value = col.isna().sum()
    value = col.dropna()
    arr = []
    arr.append((NaN_Value/len(col))*100)
    arr.append(np.percentile(value, 0))
    arr.append(np.percentile(value, 25))
    arr.append(np.percentile(value, 50))
    arr.append(np.percentile(value, 75))
    arr.append(np.percentile(value, 100))
    return np.array(arr).round(2)

index_info=["missing_value", "min", "lower_quartile", "median", "upper_quartile",
info = {}
for i in df.columns:
    if df[i].dtype == np.float64:
        info[i] = get_val(df[i])
columm_info_df =pd.DataFrame(info,index_info)
columm_info_df
```

Out[4]:

	Open	High	Low	Close	WAP	Deliverable Quantity	% Deli. Qty to Traded Qty	Spread H-L	Spread C-O
missing_value	0.00	0.00	0.00	0.00	0.00	2.97	2.97	0.00	0.00
min	1.92	1.92	1.92	1.92	1.92	17.00	2.67	0.00	-13.60
lower_quartile	16.83	17.29	16.24	16.71	16.78	27524.00	71.63	0.75	-0.60
median	22.50	23.10	21.85	22.40	22.45	51465.00	81.42	1.15	-0.11
upper_quartile	33.90	34.61	32.82	33.70	33.79	96866.00	96.30	1.80	0.24
max	142.90	142.90	136.10	136.10	136.10	1460255.00	100.00	13.60	9.80

Nhận xét

- Đối với các cột dữ liệu dạng số, ta thấy min và max của mỗi cột cách nhau rất lớn, dữ liệu sẽ phân bố trong khoảng khá lớn.
- Các cột đa số đều không có giá trị bị thiếu, chỉ có 2 cột là **Deliverable Quantity** và **% Deli. Qty to Traded Qty** => Ta cần tiền xử lý để xử lý dữ liệu bị thiếu (missing data)

4. Có cần phải tiền xử lý dữ liệu hay không? Nếu có thì cần xử lý như thế nào ?

- Tiền xử lí dữ liệu (Data preprocessing) là quá trình chuẩn bị và biến đổi dữ liệu trước khi áp dụng các thuật toán máy học hoặc khai thác dữ liệu. Mục đích của tiền xử lí dữ liệu là làm cho dữ liệu trở nên chuẩn hóa và dễ dàng xử lý hơn, để có thể tạo ra các mô hình dự đoán chính xác hơn.Tiền xử lí dữ liệu là một bước quan trọng và không thể thiếu trong quá trình xây dựng các mô hình máy học hoặc khai thác dữ liệu, vì nó đảm bảo chất lượng và tính tin cậy của kết quả dự đoán.
- Các kỹ thuật tiền xử lý dữ liệu:
  - Làm sạch dữ liệu (data cleaning/cleansing)
    - Tóm tắt hoá dữ liệu: nhận diện đặc điểm chung của dữ liệu và sự hiện diện của nhiễu hoặc các phần tử kì dị (outliers)
    - Xử lý dữ liệu bị thiếu (missing data)
    - Xử lý dữ liệu bị nhiễu (noisy data)
  - Tích hợp dữ liệu (data integration)
    - Tích hợp lược đồ (schema integration) và so trùng đối tượng (object matching)
    - Vấn đề dư thừa (redundancy)
    - Phát hiện và xử lý mâu thuẫn giá trị dữ liệu (detection and resolution of data value conflicts)

- Biến đổi dữ liệu (data transformation)
  - Làm trơn dữ liệu (smoothing)
  - Kết hợp dữ liệu (aggregation)
  - Tổng quát hóa dữ liệu (generalization)
  - Chuẩn hóa dữ liệu (normalization)
  - Xây dựng thuộc tính (attribute/feature construction)
- Thu giảm dữ liệu (data reduction)
  - Kết hợp khối dữ liệu (data cube aggregation)
  - Chọn tập con các thuộc tính (attribute subset selection)
  - Thu giảm chiều (dimensionality reduction)
  - Thu giảm lượng (numerosity reduction)
  - Tạo phân cấp ý niệm (concept hierarchy generation) và rời rạc hóa (discretization)

Xử lý dữ liệu bị thiếu

- Các cột đa số đều không có giá trị bị thiếu, chỉ có 2 cột là **Deliverable Quantity** và **% Deli. Qty to Traded Qty** => Ta cần tiền xử lý để xử lý dữ liệu bị thiếu (missing data).
- Do khoảng cách giữa min và max của 2 cột này khá lớn, ta không nên thay giá trị bị thiếu bằng mean vì sẽ làm cho kết quả tính toán có thể không chính xác.
- Thay các giá trị bị thiếu bằng median để tránh ảnh hưởng nhiều đến tính toán sau này.

In [5]:
```
# Deliverable Quantity
med_Deli_Quantity = df["Deliverable Quantity"].median()
df["Deliverable Quantity"] = df["Deliverable Quantity"].fillna(med_Deli_Quantity)

# % Deli. Qty to Traded Qty
med_Deli_Qty = df["% Deli. Qty to Traded Qty"].median()
df["% Deli. Qty to Traded Qty"] = df["% Deli. Qty to Traded Qty"].fillna(med_Deli
```

Các cột sau khi xử lý

In [6]:
```
def get_val(col):
    NaN_Value = col.isna().sum()
    value = col.dropna()
    arr = []
    arr.append((NaN_Value/len(col))*100)
    arr.append(np.percentile(value, 0))
    arr.append(np.percentile(value, 25))
    arr.append(np.percentile(value, 50))
    arr.append(np.percentile(value, 75))
    arr.append(np.percentile(value, 100))
    return np.array(arr).round(2)

index_info=["missing_value", "min", "lower_quartile", "median", "upper_quartile",
info = {}
for i in df.columns:
    if df[i].dtype == np.float64:
        info[i] = get_val(df[i])
columm_info_df =pd.DataFrame(info,index_info)
columm_info_df
```

Out[6]:

	Open	High	Low	Close	WAP	Deliverable Quantity	% Deli. Qty to Traded Qty	Spread H-L	Spread C-O
missing_value	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
min	1.92	1.92	1.92	1.92	1.92	17.00	2.67	0.00	-13.60
lower_quartile	16.83	17.29	16.24	16.71	16.78	28199.25	71.90	0.75	-0.60
median	22.50	23.10	21.85	22.40	22.45	51465.00	81.42	1.15	-0.11
upper_quartile	33.90	34.61	32.82	33.70	33.79	94606.00	94.90	1.80	0.24
max	142.90	142.90	136.10	136.10	136.10	1460255.00	100.00	13.60	9.80

Ghi df ra file ./deccan gold mines ltd eod price(da_xu_ly).csv

In [7]:
```
# df.to_csv('../datasets/deccan gold mines ltd eod price(da_xu_ly).csv')
```

C. Khám phá mối quan hệ trong dữ liệu

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

In [9]:
```
# Đọc dataset đã được xử lý
df = pd.read_csv('../datasets/deccan gold mines ltd eod price(da_xu_ly).csv', ind
```

In [10]:

```
# Sắp xếp dataset theo thứ tự thời gian
df["Date"] = pd.to_datetime(df["Date"], dayfirst=True)
df = df.sort_values(by='Date')

# Thêm dữ liệu 'năm' vào dataset
df['Year'] = [d.year for d in df['Date']]

# Lấy số liệu từ năm 2004 trở đi
df = df[df['Year'] >= 2004]

df
```

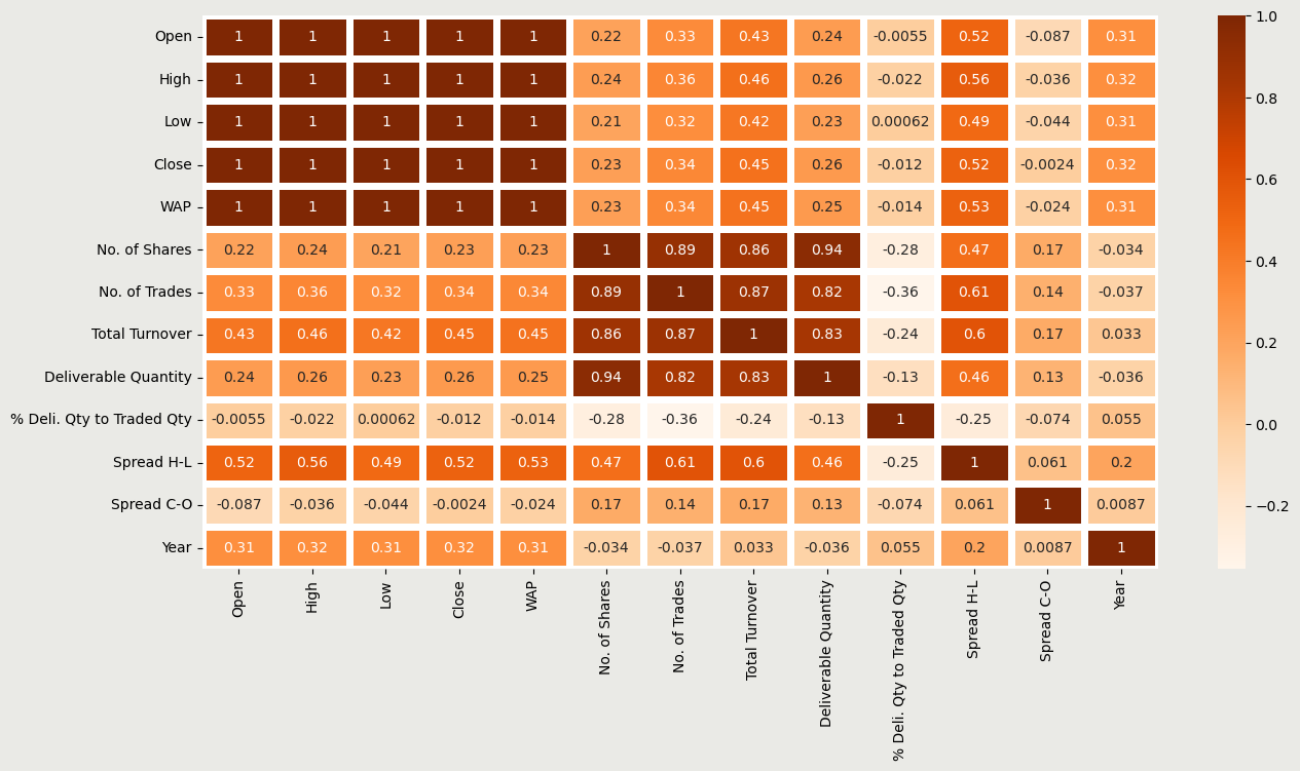
Out[10]:

	Date	Open	High	Low	Close	WAP	No. of Shares	No. of Trades	Total Turnover	Deliverable Quantity	% Deli. Qty to Traded Qty	Spread H-L	S
4774	2004-01-20	1.92	1.92	1.92	1.92	1.92	50	1	96	51465.0	81.42	0.00	
4773	2004-01-21	2.30	2.30	2.30	2.30	2.30	50	1	115	51465.0	81.42	0.00	
4772	2004-01-22	2.76	2.76	2.76	2.76	2.76	100	1	276	51465.0	81.42	0.00	
4771	2004-01-23	3.31	3.31	3.31	3.31	3.31	200	1	662	51465.0	81.42	0.00	
4770	2004-01-27	3.97	3.97	3.97	3.97	3.97	200	1	794	51465.0	81.42	0.00	
...	...	...	...	...	...	...	...	...	...	...	...	...	
4	2023-04-06	48.40	49.04	47.65	49.04	48.80	245835	441	11996688	203526.0	82.79	1.39	
3	2023-04-10	51.49	51.49	48.00	48.16	49.90	339774	517	16954583	280083.0	82.43	3.49	
2	2023-04-11	48.87	50.05	47.76	49.90	49.21	270964	509	13335488	208486.0	76.94	2.29	
1	2023-04-12	49.98	51.00	49.30	50.07	49.99	193359	417	9665903	166630.0	86.18	1.70	
0	2023-04-13	50.47	50.47	47.86	49.15	49.35	122650	413	6053293	99475.0	81.10	2.61	

4775 rows × 14 columns

Mối tương quan giữa Open price và Close price

In [11]:
```
df.set_index('Date',inplace=True)

#Correlation Map
plt.figure(figsize = [15, 7], clear = True, facecolor = '#EAEAE6')
sns.heatmap(df.corr(), annot = True, square = False, linewidths = 5,
            linecolor = "white", cmap = "Oranges");

df.reset_index(inplace=True)
```



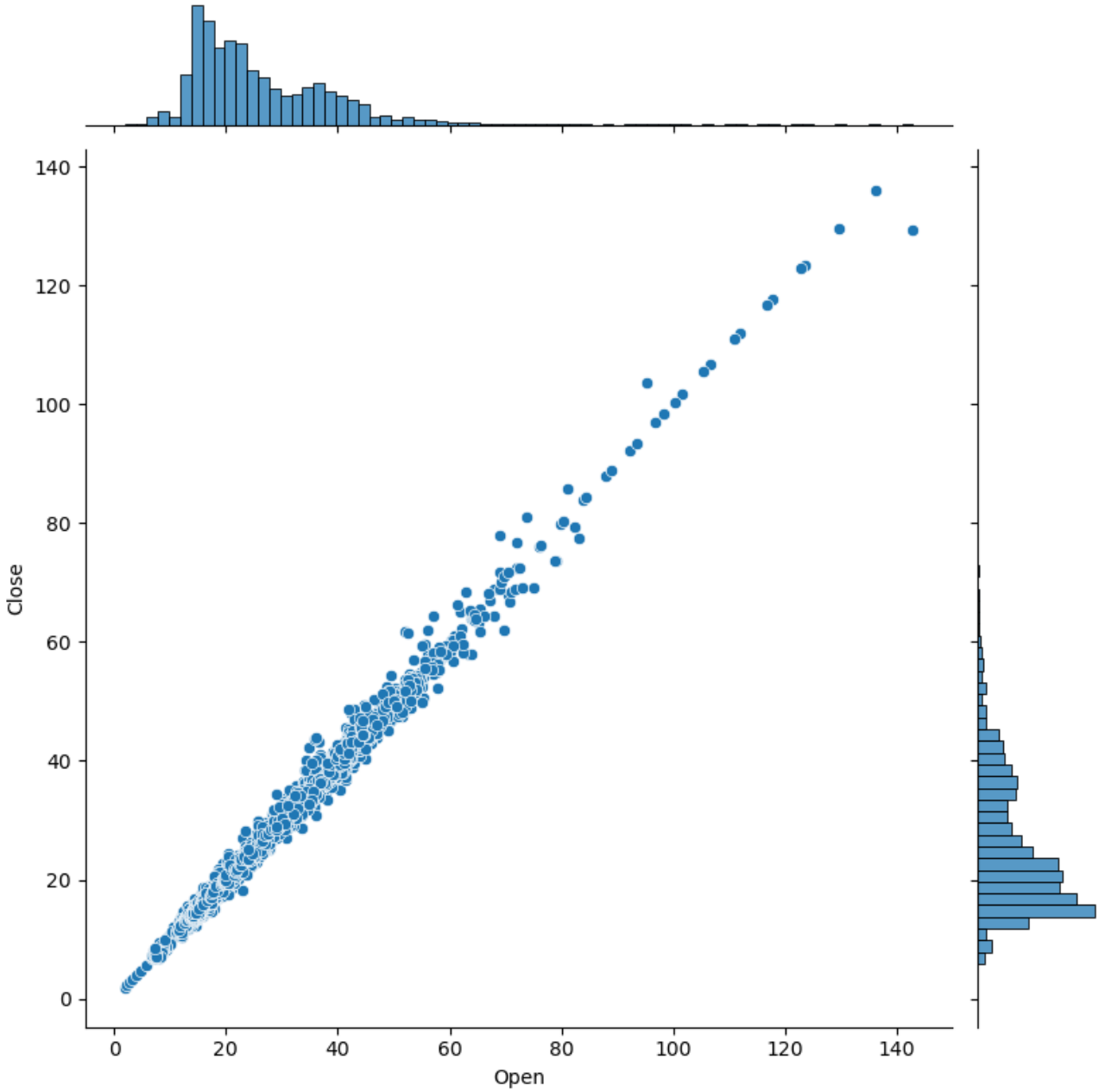
Nhận xét

- Từ đồ thị có thể thấy thuộc tính Open và Close có mối tương quan thuận với nhau.

Sau khi sử dụng heatmap để hình dung sơ bộ sự tương quan của Open price và Close price, dưới đây là thể hiện rõ ràng sự tương quan của 2 thuộc tính

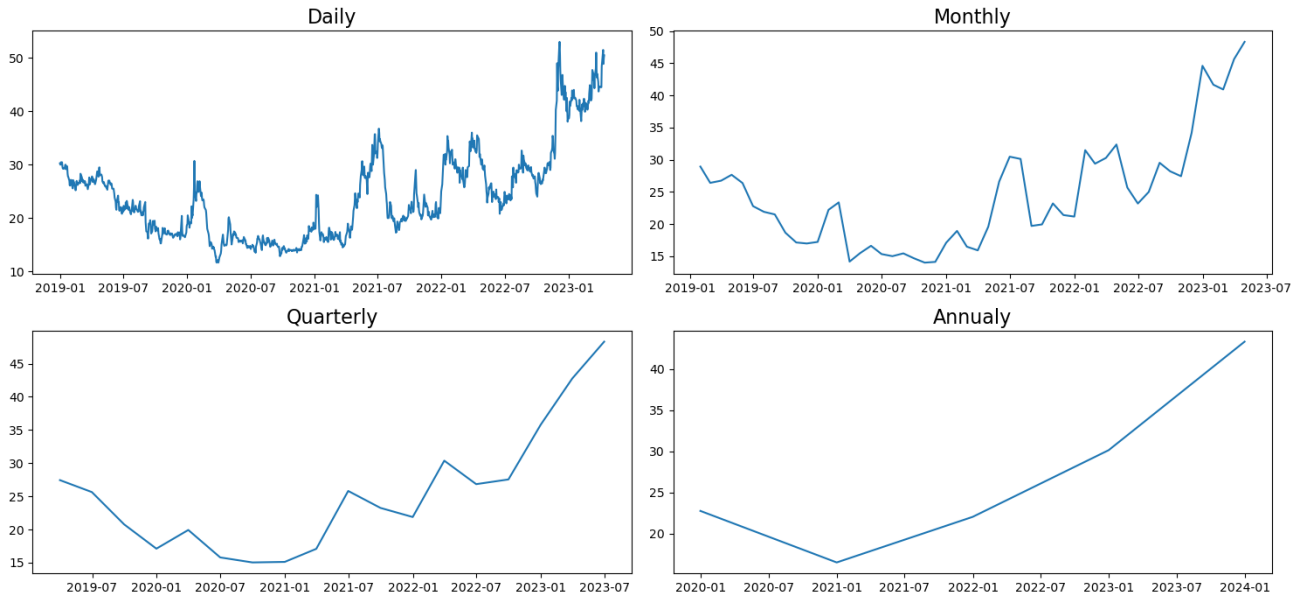
In [12]:

```
df.set_index('Date',inplace=True)
sns.jointplot(x = "Open", y = "Close", data = df, height = 8, ratio = 6, kind = "
df.reset_index(inplace=True)
```



1. Open price

In [13]:
```
df.set_index('Date',inplace=True)

## targeted period
data = df[df.index >= '2019']

fig,axes = plt.subplots(2,2,figsize=[15,7])

## resampling to daily freq (original data)
axes[0,0].plot(data.Open)
axes[0,0].set_title("Daily",size=16)

## resampling to monthly freq
axes[0,1].plot(data.Open.resample('M').mean())
axes[0,1].set_title("Monthly",size=16)

## resmapling to quarterly freq
axes[1,0].plot(data.Open.resample('Q').mean())
axes[1,0].set_title('Quarterly',size=16)

## resampling to annualy freq
axes[1,1].plot(data.Open.resample('A').mean())
axes[1,1].set_title('Annualy',size=16)

plt.tight_layout()
plt.show()

df.reset_index(inplace=True)
```

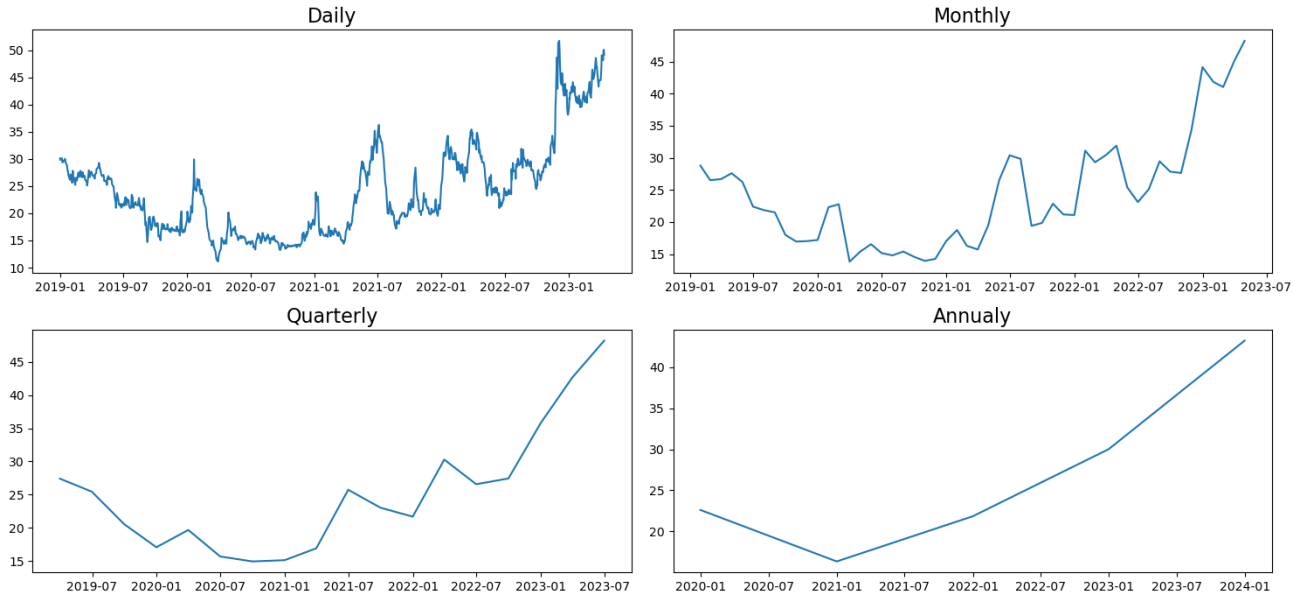


Nhật xét

- Giai đoạn trước năm 2020, Open price có dấu hiệu sụt giảm tới năm 2021 là chạm đáy.
- Sau đó thì Open price bắt đầu tăng trưởng ổn định trở lại
- Các kỹ thuật được áp dụng: Facet (Thể hiện biểu đồ qua từng ngày, từng tháng, từng quý, từng năm). Khi ta áp dụng kỹ thuật này sẽ cho ta dễ thấy được sự biến động cụ thể Open price của vàng qua từng khung thời gian mà ta thể hiện.
- Từ việc trực quan hóa, biểu đồ trên giúp ta thấy được Open price đang phục hồi khá nhiều sau Covid-19 từ đó giúp ích cho việc đầu tư.

2. Close price

In [14]:
```
df.set_index('Date',inplace=True)

fig,axes = plt.subplots(2,2,figsize=[15,7])

## resampling to daily freq (original data)
axes[0,0].plot(data.Close)
axes[0,0].set_title("Daily",size=16)

## resampling to monthly freq
axes[0,1].plot(data.Close.resample('M').mean())
axes[0,1].set_title("Monthly",size=16)

## resmapling to quarterly freq
axes[1,0].plot(data.Close.resample('Q').mean())
axes[1,0].set_title('Quarterly',size=16)

## resampling to annualy freq
axes[1,1].plot(data.Close.resample('A').mean())
axes[1,1].set_title('Annualy',size=16)

plt.tight_layout()
plt.show()

df.reset_index(inplace=True)
```



Nhận xét

- Giai đoạn trước năm 2020, Close price có dấu hiệu sụt giảm tới năm 2021 là chạm đáy.
- Sau đó thì Close price bắt đầu tăng trưởng ổn định trở lại
- Các kỹ thuật được áp dụng: Facet (Thể hiện biểu đồ qua từng ngày, từng tháng, từng quý, từng năm). Khi ta áp dụng kỹ thuật này sẽ cho ta dễ thấy được sự biến động cụ thể Close price của vàng qua từng khung thời gian mà ta thể hiện.
- Từ việc trực quan hóa, biểu đồ trên giúp ta thấy được Close price đang phục hồi khá nhiều sau Covid-19 từ đó giúp ích cho việc đầu tư.
- Có thể thấy sự giống nhau của Open price và Close price bởi vì chúng tương quan thuận dựa vào biểu đồ heat map ở trên

3. Lowest price

'Lowest price' là giá thấp nhất trong một phiên giao dịch hoặc trong chu kỳ theo dõi biến động giá.

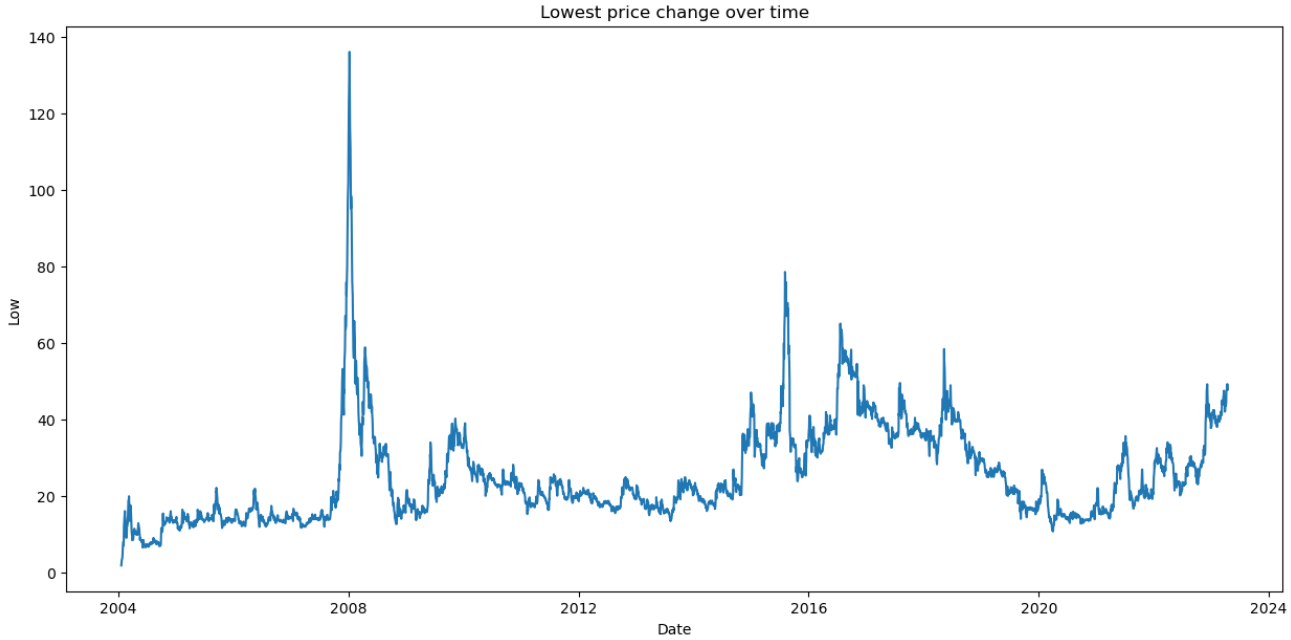
Giá thấp nhất thường được sử dụng như một chỉ số để đánh giá tình hình giá cả của một cổ phiếu trong ngắn hạn. Nhà đầu tư sẽ quan tâm đến giá thấp nhất của một cổ phiếu để đưa ra quyết định mua bán cổ phiếu. Nếu giá cổ phiếu tiệm cận hoặc vượt qua giá thấp nhất, nhà đầu tư có thể quyết định bán cổ phiếu để giảm thiểu rủi ro, đặc biệt nếu giá cổ phiếu tiếp tục giảm.

Giá thấp nhất thường được so sánh với giá cao nhất (highest price) để đưa ra các dự đoán về xu hướng giá cổ phiếu trong tương lai và hỗ trợ trong việc quyết định mua bán cổ phiếu. Trong một số trường hợp, giá thấp nhất cũng có thể cho thấy các điểm mua cổ phiếu tiềm năng, đặc biệt đối với các nhà đầu tư có chiến lược đầu tư dài hạn.

In [15]:
```
lowest = df["Low"]
highest = df["High"]
```

Sử dụng biểu đồ đường để dễ dàng quan sát xu hướng của 'lowest price' của giá vàng theo thời gian.

In [16]:
```
plt.figure(figsize=(15, 7))
sns.lineplot(data=df, x='Date', y='Low')
plt.title('Lowest price change over time')
plt.show()
```



Nhận xét:

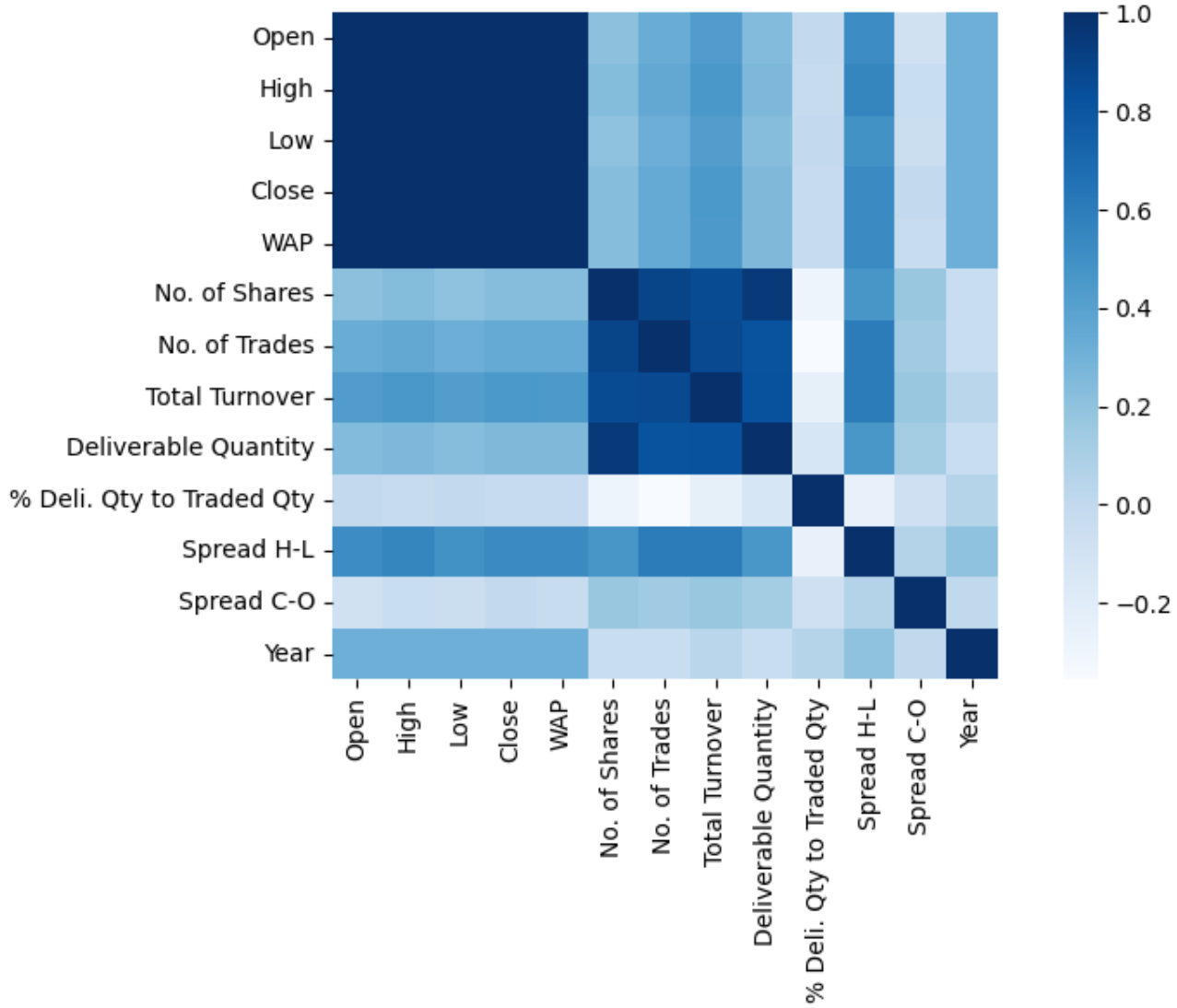
- Lowest price của vàng chạm đỉnh vào năm 2018, sau đó giảm dần về mức trung bình, rồi lại tăng nhẹ vào khoảng cuối năm 2015.
- Năm 2008, lowest price của vàng là cao nhất vì vào năm đó, vàng được coi là vịnh tránh bão an toàn số một trong mắt giới đầu tư trong bối cảnh lạm phát leo thang, đồng USD trượt giá và các tổ chức tài chính lần lượt đổ vỡ.
- Sau năm 2016, lowest price của vàng có xu hướng giảm dần, thấp nhất là vào năm 2020. Sau đại dịch Covid-19, nền kinh tế dần dần phục hồi, chính phủ các nước cũng chú trọng việc kích thích kinh tế, điều này khiến giới đầu tư dành sự chú ý tới vàng - một loại tài sản trú ẩn an toàn. Vậy nên sau năm 2020, lowest price của vàng có xu hướng tăng dần dần.

4. Sự tương quan giữa 3 thuộc tính High, No. of Shares, No. of Trade

- Hệ số tương quan là chỉ số thống kê đo lường mức độ mạnh yếu của mối quan hệ giữa hai biến số. Trong đó:
  - Hệ số tương quan có giá trị từ -1.0 đến 1.0. Kết quả được tính ra lớn hơn 1.0 hoặc nhỏ hơn -1 có nghĩa là có lỗi trong phép đo tương quan.
  - Hệ số tương quan có giá trị âm cho thấy hai biến có mối quan hệ nghịch biến hoặc tương quan âm (nghịch biến tuyệt đối khi giá trị bằng -1).
  - Hệ số tương quan có giá trị dương cho thấy mối quan hệ đồng biến hoặc tương quan dương (đồng biến tuyệt đối khi giá trị bằng 1).
  - Tương quan bằng 0 cho hai biến độc lập với nhau.
- Tạo biểu đồ Heatmap thể hiện độ tương quan giữa các biến với nhau:

In [17]:
```
# Correlation Map
corr = df.corr(method = "pearson") # Sử dụng hệ số tương quan Pearson
f, ax = plt.subplots(figsize = (10, 5))
sns.heatmap(corr, mask = np.zeros_like(corr, dtype = np.bool_), cmap = "Blues", s
```

Out[17]: <AxesSubplot:>



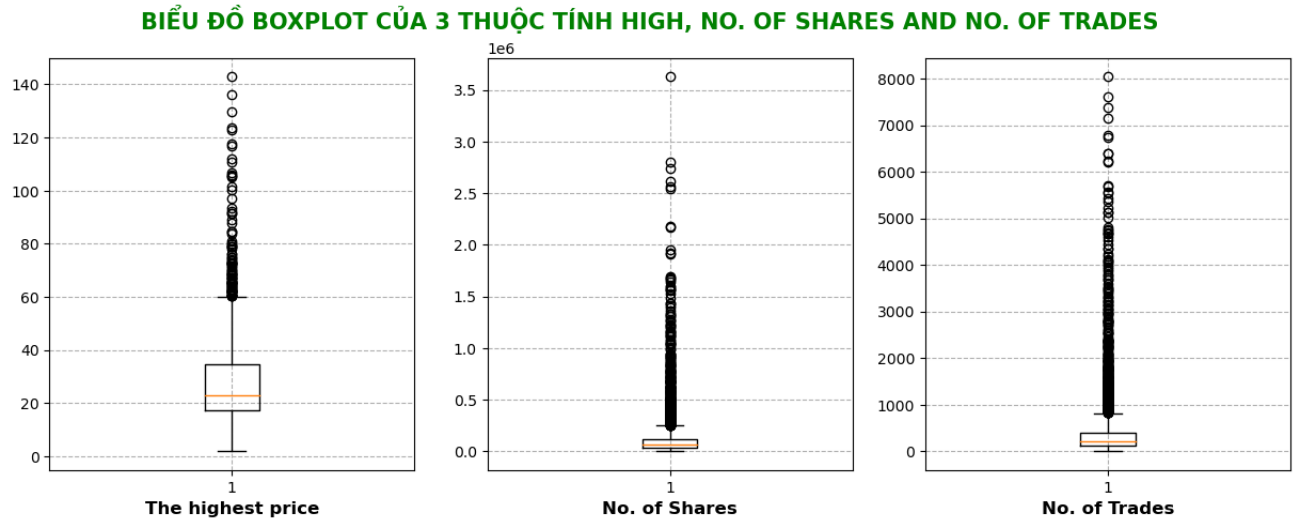
Nhận Xét:

- Từ đồ thị có thể thấy thuộc tính High có mối tương quan khá ít với No. of Shares và No. of Trades.
- Ngược lại, No. of Shares và No. of Trades có mối tương quan thuận với nhau.

Biểu đồ Boxplot thể hiện sự phân phối dữ liệu của 3 thuộc tính

In [18]:
```
fig, (ax1, ax2, ax3) = plt.subplots(nrows = 1, ncols = 3, figsize = (15,5))
ax1.boxplot(df['High'])
ax1.grid(True, linestyle = "--")

ax2.boxplot(df['No. of Shares'])
ax2.grid(True, linestyle = "--")

ax3.boxplot(df['No. of Trades'])
ax3.grid(True, linestyle = "--")

fig.suptitle(
    "BIỂU ĐỒ BOXPLOT CỦA 3 THUỘC TÍNH HIGH, NO. OF SHARES AND NO. OF TRADES",
    fontsize = 15, color = "green", fontweight = "bold"
)
ax1.set_xlabel("The highest price", fontsize = 12, fontweight = "bold")
ax2.set_xlabel("No. of Shares", fontsize = 12, fontweight = "bold")
ax3.set_xlabel("No. of Trades", fontsize = 12, fontweight = "bold")
```

Out[18]: Text(0.5, 0, 'No. of Trades')



Nhận Xét:

- Cả 3 thuộc tính đều xuất hiện nhiều outliers nằm ở phía trên.
- Cả 3 thuộc tính đều có xu hướng lệch phải.

Sự thay đổi của 3 thuộc tính High, No. of Shares, No. of Trade qua các năm

A. Highest Price qua các năm

Highest Price của giá vàng là giá cao nhất mà vàng đạt được trong ngày giao dịch.

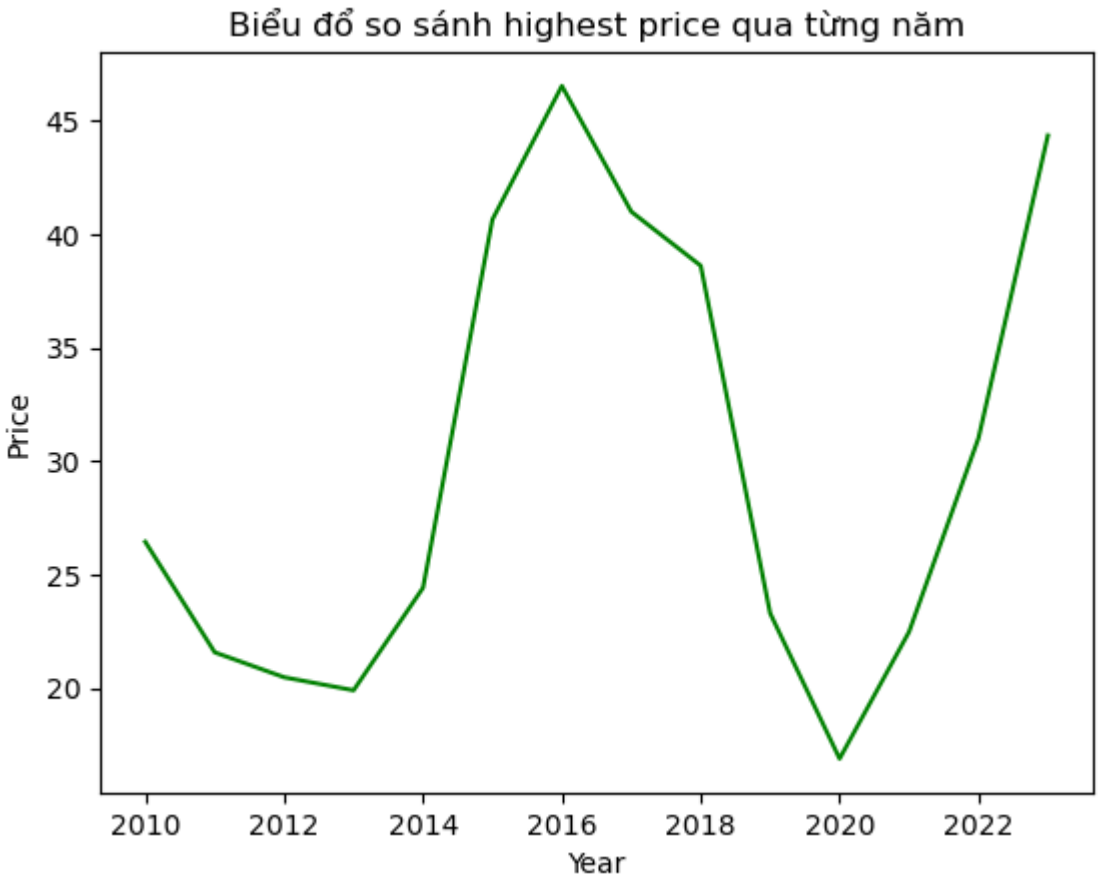
In [19]:
```
# Lấy dữ liệu các năm gần đây để dễ so sánh sự thay đổi
df_year = df[df['Year'] >= 2010]
df_high_year = df_year.groupby('Year')['High'].mean()
```

Vẽ biểu đồ:

- Biểu đồ được chọn để trực quan: Biểu đồ đường

- Nguyên nhân: Biểu đồ đường khá hữu dụng cho việc biểu diễn sự thay đổi của 1 biến theo thời

In [20]:
```
plt.plot(df_high_year, color='green')
plt.gca().set(title= "Biểu đổ so sánh highest price qua từng năm", xlabel= "Year"
plt.show()
```



Nhận Xét:

- Giai đoạn 2010 - 2013, Highest price của vàng có sự biến động khá lớn.
- Từ năm 2010 - 2016, Highest price của vàng có xu hướng tăng nhiều hơn. Từ năm 2016 - 2018 thì giảm nhẹ.
- Giai đoạn Covid-19 (2019 - 2020), Highest price của vàng tụt dốc chỉ phục hồi lại và tăng cao sau Covid-19 (2021 - nay)
- Các kỹ thuật được áp dụng: Manipulate View (Thay đổi màu sắc, định dạng biểu đồ, thay đổi tiêu đề). Khi ta áp dụng kỹ thuật này sẽ cho ta dễ thấy được sự thay đổi Highest price của vàng qua từng năm.
- Từ việc trực quan hóa, biểu đồ trên giúp ta thấy được Highest price đang phục hồi khá nhiều sau Covid-19 từ đó giúp ích cho việc đầu tư.

So sánh giữa Highest Price, Lowest Price, Open Price, Close Price qua các năm

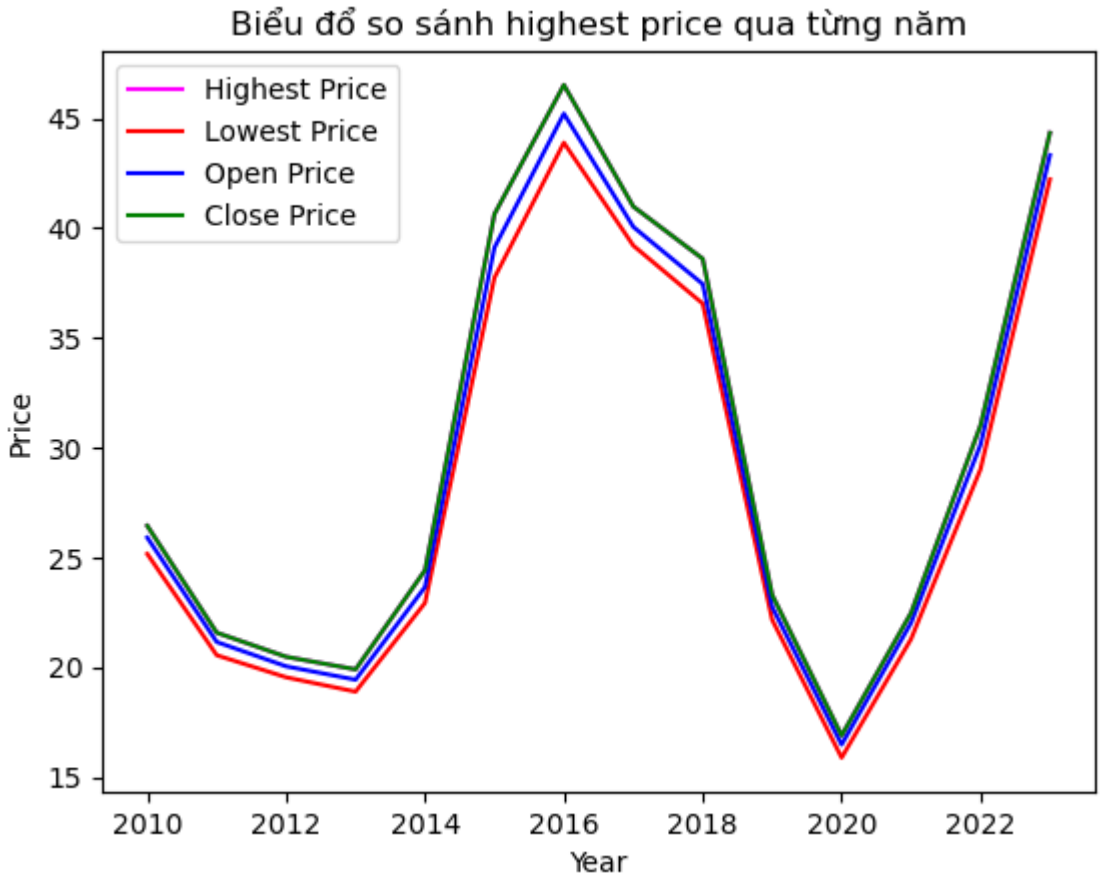
- Lấy trung bình theo các năm

In [21]:
```
df_open_year = df_year.groupby('Year')['Open'].mean()
df_close_year = df_year.groupby('Year')['High'].mean()
df_low_year = df_year.groupby('Year')['Low'].mean()
```

- Vẽ biểu đồ

In [22]:

```
years = [2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021,
plt.plot(years, df_high_year, color='magenta')
plt.plot(years, df_low_year, color='red')
plt.plot(years, df_open_year, color='blue')
plt.plot(years, df_close_year, color='green')
plt.gca().set(title= "Biểu đổ so sánh highest price qua từng năm", xlabel= "Year"
plt.legend(["Highest Price", "Lowest Price", "Open Price", "Close Price"])
plt.show()
```



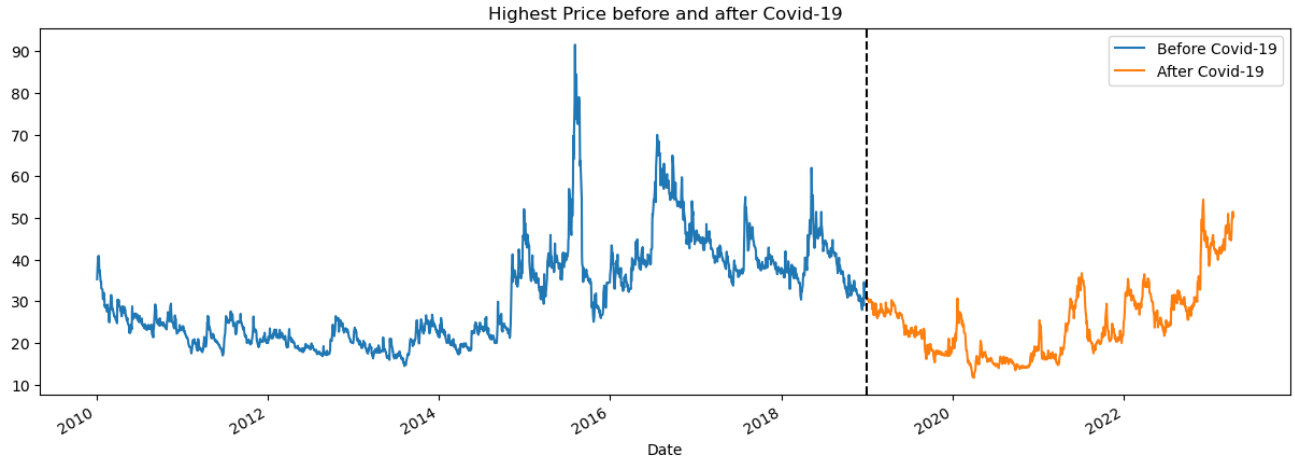
Nhận Xét:

- Giữa 4 thuộc tính có mối quan hệ tương quan đồng biến với nhau.
- Sự thay đổi của 4 thuộc tính tương tự biểu đồ Highest Price qua từng năm ở phía trên.
- Các kỹ thuật được áp dụng: Manipulate View (Sử dụng màu sắc khác nhau, định dạng biểu đồ, thay đổi tiêu đề, chú thích biểu đồ). Sử dụng màu sắc khác nhau giúp phân biệt các đường với nhau, chú thích biểu đồ để biểu đồ dễ hiểu.
- Sự tương quan giữa 4 thuộc tính là những yếu tố quan trọng ảnh hưởng đến giá vàng, hiểu được các thuộc tính này giúp ích cho ta trong việc đầu tư.

Sự thay đổi của Highest Price trước và sau Covid-19

In [23]:
```
before_Covid = df_year[df["Date"] < '01-01-2019']
after_Covid = df_year[df["Date"] >= '01-01-2019']

before_Covid_high = before_Covid[["Date", "High"]].set_index('Date')
after_Covid_high = after_Covid[["Date", "High"]].set_index('Date')

fig, ax = plt.subplots(figsize = (15,5))

before_Covid_high.plot(ax = ax, label = "Before Covid-19", title = "Highest Price
after_Covid_high.plot(ax = ax, label = "After Covid-19")

ax.axvline('2019-01-01', color = 'black', ls = '--')
ax.legend(['Before Covid-19', 'After Covid-19'])
plt.show()
```

```
C:\Users\Dat Phan\AppData\Local\Temp\ipykernel_9172\494278812.py:1: UserWarning: Boo
lean Series key will be reindexed to match DataFrame index.
  before_Covid = df_year[df["Date"] < '01-01-2019']
C:\Users\Dat Phan\AppData\Local\Temp\ipykernel_9172\494278812.py:2: UserWarning: Boo
lean Series key will be reindexed to match DataFrame index.
  after_Covid = df_year[df["Date"] >= '01-01-2019']
```



Nhận Xét:

- Ta sử dụng 2 màu để phân biệt giữa 2 giai đoạn trước và sau Covid-19.
- Highest Price giảm trong giai đoạn dịch và bắt đầu phục hồi sau dịch.
- Các kỹ thuật được áp dụng: Manipulate View (Thay đổi màu sắc, định dạng biểu đồ, thay đổi tiêu đề, chú thích). Ta chia 2 giai đoạn thành 2 màu khác nhau và ngăn cách bởi 1 mốc thời gian giúp ta so sánh rõ hơn sự thay đổi của 2 giai đoạn.
- Từ biểu đồ, ta có thể thấy được sự thay đổi giữa 2 giai đoạn giúp ích nhiều cho việc đầu tư.

B. No. of Shares và No. of Trades qua các năm

Tính trung bình của No. of Shares và No. of Trades qua từng năm

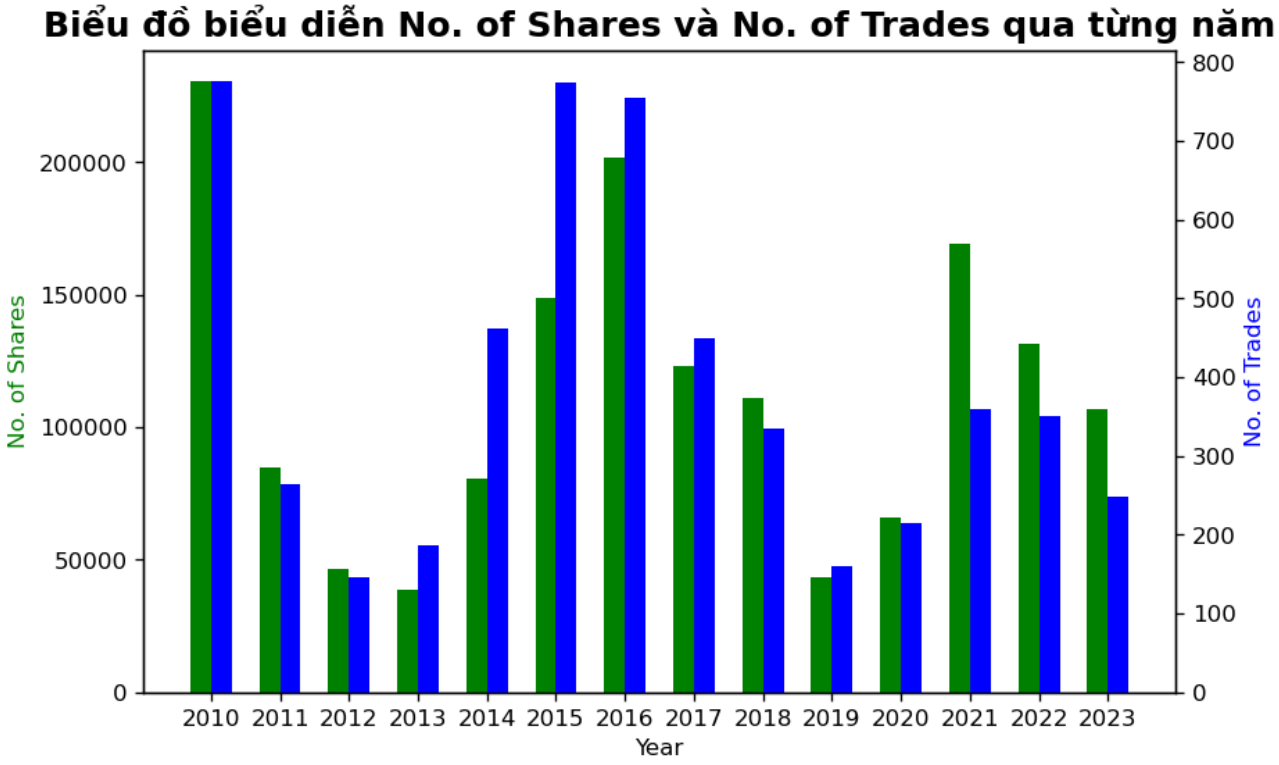
In [24]:
```
# No. of Shares qua tùng năm
df_shares_year = df_year.groupby('Year')['No. of Shares'].mean()

# No. of Trades
df_trades_year = df_year.groupby('Year')['No. of Trades'].mean()
```

Vẽ biểu đồ so sánh

- Biểu đồ biểu diễn: Biểu đồ cột.
- Lý do: Biểu đồ cột giúp ta biểu diễn được khối lượng của 1 biến và dễ so sánh với các biến khác về khối lượng.

In [25]:
```
fig, ax1 = plt.subplots(1,1,figsize = (8,5), dpi = 120)
years = ['2010', '2011', '2012', '2013', '2014', '2015', '2016', '2017', '2018',

# Set the width of the bars
bar_width = 0.3

# Set the positions of the bars on the x-axis
r1 = range(len(years))
r2 = [x + bar_width for x in r1]

ax1.bar(r1 ,df_shares_year, color = "green", width=bar_width)
ax1.set_xlabel('Year', fontsize = 10)
ax1.set_ylabel('No. of Shares', color = "green", fontsize = 10)

ax2  = ax1.twinx()
ax2.bar(r2 , df_trades_year, color = "blue", width=bar_width)
ax2.set_ylabel('No. of Trades', color = "blue", fontsize = 10)

plt.xticks([r + (bar_width / 2) for r in r1], years)
plt.title("Biểu đồ biểu diễn No. of Shares và No. of Trades qua từng năm", fontsi
```

Out[25]: Text(0.5, 1.0, 'Biểu đồ biểu diễn No. of Shares và No. of Trades qua từng năm')



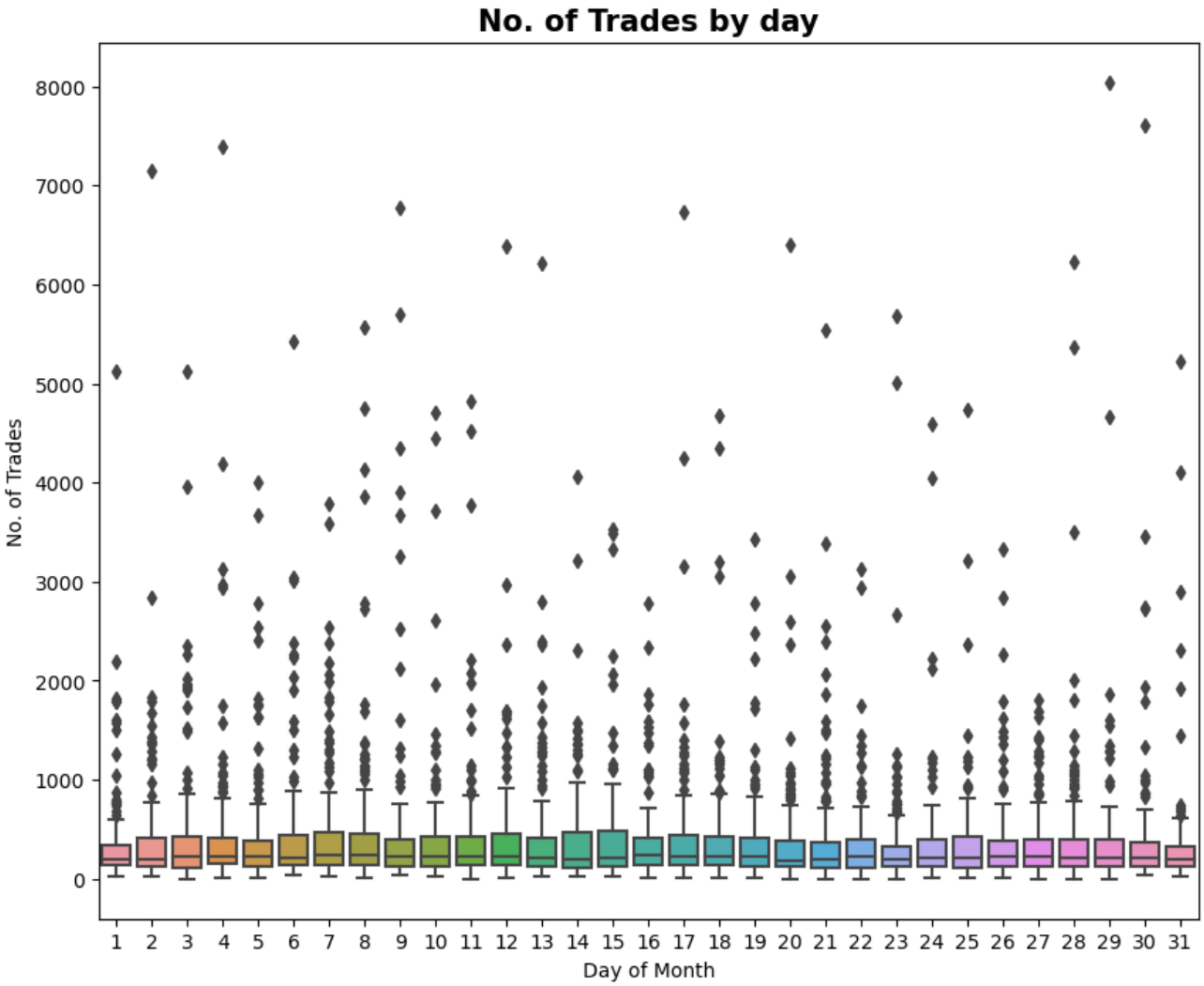
Nhận Xét:

- Ta có thể thấy No. of Shares và No. of Trades có mối tương quan đồng biến với nhau khi cả 2 cùng tăng hoặc cùng giảm.
- No. of Shares và No. of Trades có sự thay đổi nhiều qua từng năm.
- Tương tự, ta thấy sự thay đổi trước và sau Covid-19 khi khối lượng cả No. of Shares và No. of Trades đều giảm trong giai đoạn dịch và hồi phục lại sau đại dịch.
- Kỹ thuật Maniputulate View: Sử dụng nhiều màu sắc, định dạng kích thước biểu đồ, thay đổi tiêu đề, chú thích biểu đồ. Sử dụng 2 màu để phân biệt 2 biểu đồ, chú thích 2 trục tung vì 2 thuộc tính có đại lượng khác nhau.
- Từ sự thay đổi tương quan của 2 thuộc tính giúp việc đầu tư.

Số giao dịch của ngày nào trong tháng là cao nhất

In [26]:
```
df['Day of Month'] = [d.day for d in df['Date']]

fig, ax = plt.subplots(figsize = (10,8))
sns.boxplot(data = df, x = df["Day of Month"], y = df["No. of Trades"])
ax.set_title("No. of Trades by day", fontsize = 15, fontweight = "bold")
```

Out[26]:
```
Text(0.5, 1.0, 'No. of Trades by day')
```



Nhận xét:

- Dữ liệu có khá nhiều outliers.
- Từ biểu đồ, ta thấy khối lượng giao dịch ở đầu tháng là cao nhất. Cùng với đó ta biết nhà đầu tư thường đầu tư ở thời điểm nào trong tháng từ đó hỗ trợ cho đầu tư.
- Kỹ thuật Maniputulate View: Sử dụng nhiều màu sắc, chú thích tiêu đề. Sử dụng nhiều màu để phân biệt các ngày trong tháng.

5. Total Turnover

Là tổng giá trị của các giao dịch được thực hiện trong ngày.

Thường được tính bằng cách nhân số lượng cổ phiếu được giao dịch với giá của chúng trong ngày giao dịch đó.

Đây là một chỉ số quan trọng để đánh giá hoạt động của thị trường chứng khoán và độ thanh khoản của một cổ phiếu cụ thể. Nó cũng được sử dụng để đo lường sự quan tâm của các nhà đầu tư đối với một cổ phiếu hoặc thị trường chứng khoán cụ thể. Khi có nhiều giao dịch diễn ra, các nhà đầu tư có thể dễ dàng mua bán cổ phiếu mà không phải lo lắng về việc không tìm được người mua hoặc người bán.

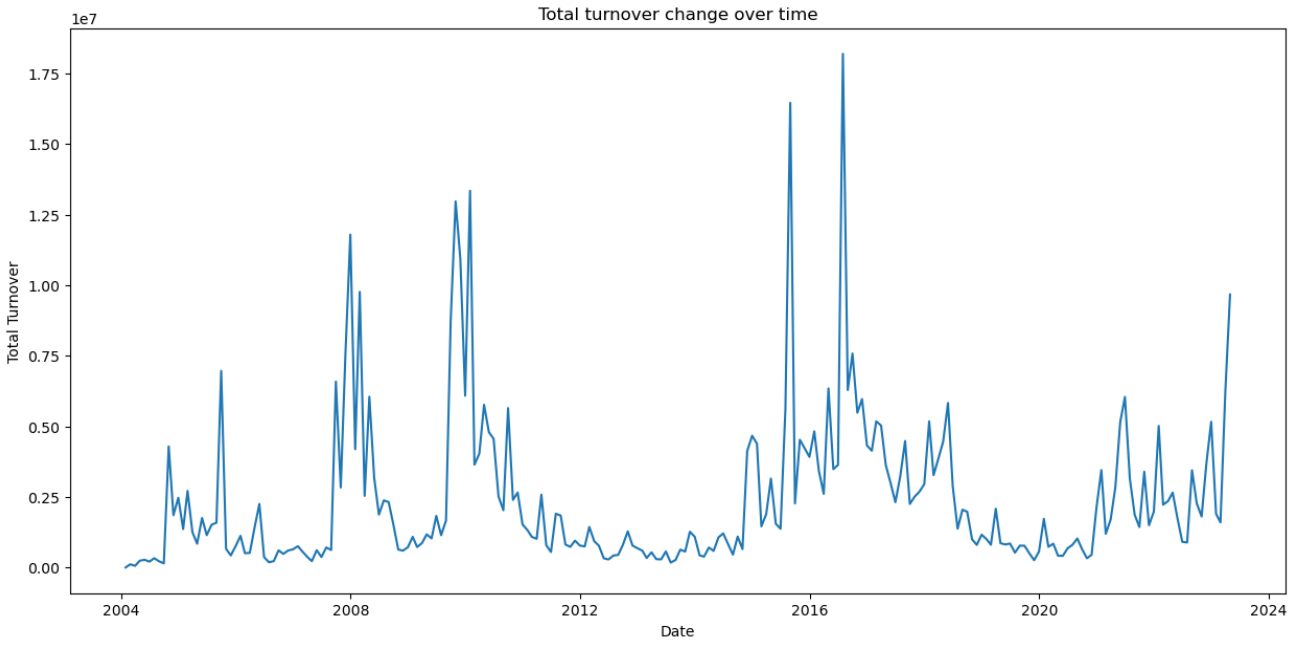
'Total turnover' cũng cung cấp cho các nhà đầu tư một số thông tin về một cổ phiếu trong quá khứ. Một cổ phiếu có 'total turnover' cao trong một khoảng thời gian nhất định cho thấy rằng cổ phiếu này nhận

In [27]:
```
total_turnover = df["Total Turnover"]
```

Dùng kỹ thuật Reduce để gom nhóm dữ liệu theo từng tháng, giúp giảm số lượng điểm dữ liệu trên biểu đồ. Sau đó trực quan bằng biểu đồ đường để dễ dàng quan sát xu hướng của 'total turnover' của vàng theo thời gian.

In [28]:
```
df.set_index("Date", inplace=True)
monthly_data = df.resample('M').median().reset_index()
```

In [29]:
```
plt.figure(figsize=(15, 7))
plt.title('Total turnover change over time')
sns.lineplot(x='Date', y='Total Turnover', data=monthly_data)
df.reset_index(inplace=True)
```



Dự đoán

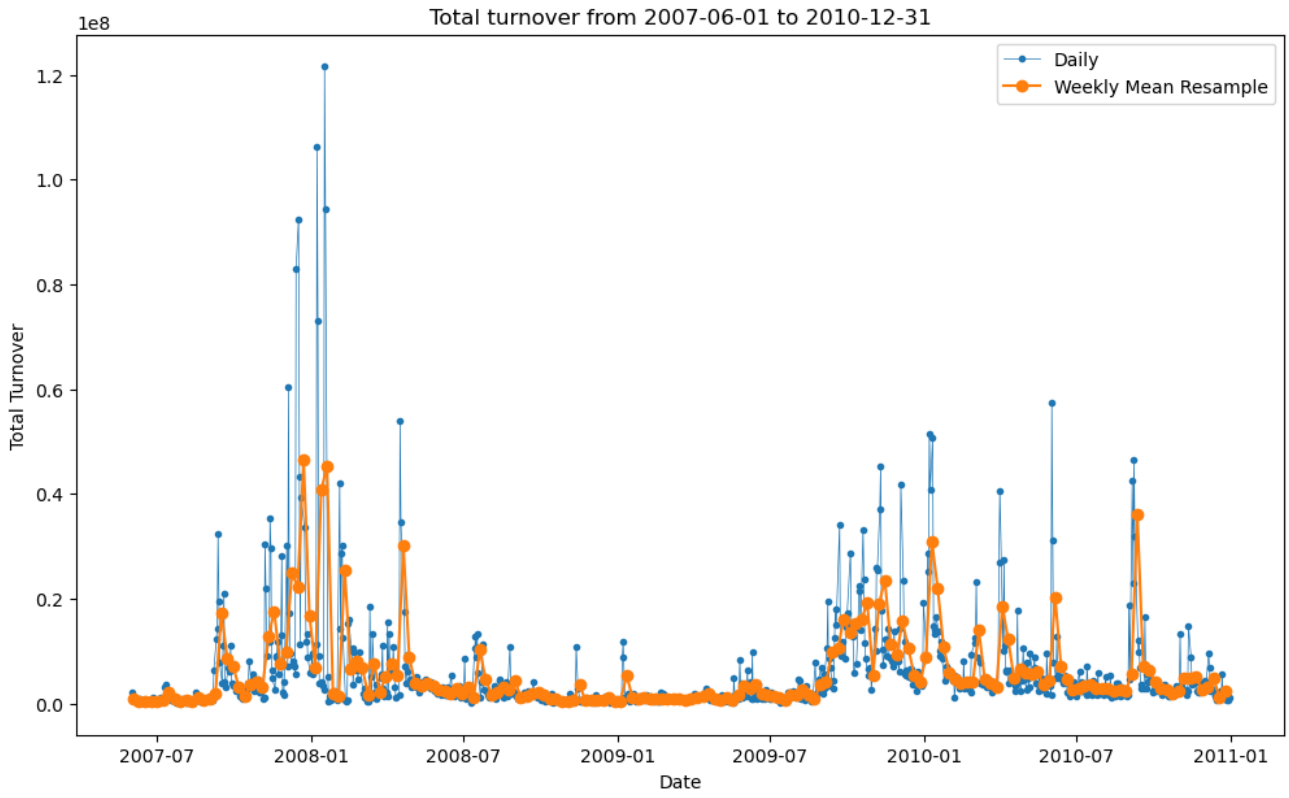
Dựa vào biểu đồ , có thể thấy rằng Total turnover có sự biến động mạnh vào những khoảng thời gian 2008-2011 và 2015-2017

a. Phân tích giai đoạn 2008-2011

Trực quan 'total turnover' theo ngày và theo tuần trên cùng 1 biểu đồ để thấy được lợi ích của việc áp dụng phương pháp **Reduce**, cụ thể là **Downsampling**, trong trực quan hóa dữ liệu:

In [30]:

```
df.set_index('Date', inplace=True)

# Sử dụng kỹ thuật Reduce để gom nhóm dữ liệu theo từng tuần, giúp giảm số lượng d
weekly_data = df.resample('W').mean()

# Ngày bắt đầu và kết thúc của khoảng thời gian cần quan tâm
start_date = '2007-06-01'
end_date = '2010-12-31'

fig, ax = plt.subplots(figsize=(12, 7))
ax.plot(df.loc[start_date:end_date, 'Total Turnover'],
        marker='.', linestyle='-', linewidth=0.5, label='Daily')

ax.plot(weekly_data.loc[start_date:end_date, 'Total Turnover'],
        marker='o', markersize=6, linestyle='-', label='Weekly Mean Resample')

ax.set_title('Total turnover from {} to {}'.format(start_date, end_date))
ax.set_xlabel('Date')
ax.set_ylabel('Total Turnover')
ax.legend()

plt.show()
```

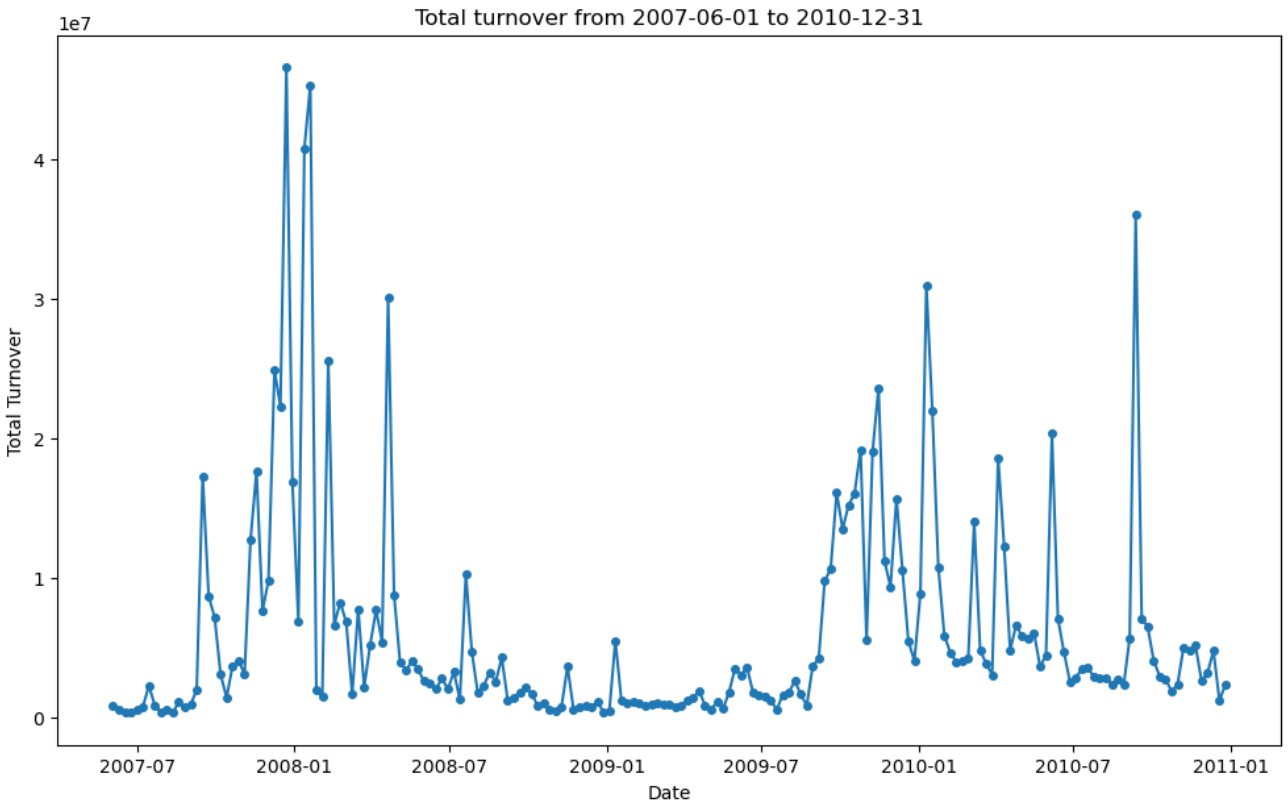


Khi áp dụng phương pháp **Reduce**, có thể thấy rằng khi vẽ biểu đồ theo dữ liệu đã được gom nhóm theo tuần, biểu đổ sẽ dễ nhìn hơn do các biến thiên có tần suất dao động cao đã được loại bỏ trong quá trình gom nhóm.

Bây giờ ta sẽ vẽ lại biểu đồ theo dữ liệu đã được gom nhóm theo tuần để dễ quan sát và phân tích:

In [31]:

```
fig, ax = plt.subplots(figsize=(12, 7))
ax.plot(weekly_data.loc[start_date:end_date, 'Total Turnover'],
        marker='o', markersize=4, linestyle='-')

ax.set_title('Total turnover from {} to {}'.format(start_date, end_date))
ax.set_xlabel('Date')
ax.set_ylabel('Total Turnover')

plt.show()
```



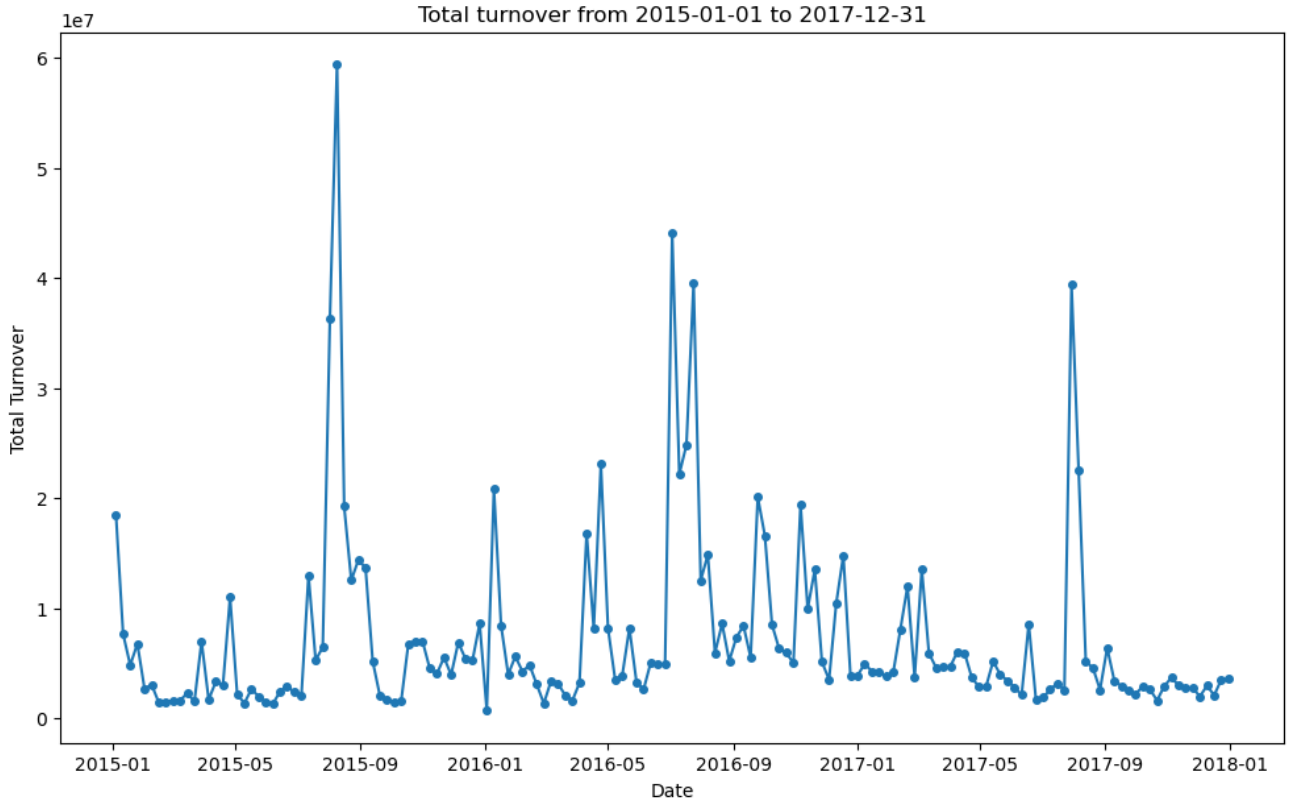
Nhận xét

- Năm 2008 là một năm đầy biến động với những thăng trầm của kinh tế thế giới và giá vàng cũng nằm trong vòng xoáy của những biến động đó.
- Một số nguyên nhân làm tăng giá vàng trong năm 2008 bao gồm sự biến động của kinh tế toàn cầu, sự tăng trưởng của cung và cầu và các chính sách của các chính phủ và ngân hàng trung ương.
- Năm 2008 cũng là năm đánh dấu sự ra đời và phát triển mạnh mẽ của các sàn giao dịch vàng. Thời điểm năm 2019 đã có hơn 10 sàn giao dịch vàng đã được đi vào hoạt động. Việc ra đời các sàn vàng, cùng với cơn sốt giá đợt đầu năm 2018 đã khiến một lượng tiền lớn chảy từ chứng khoán sang. Tuy nhiên, sự khốc liệt của loại hình đầu tư này cũng đã khiến nhiều nhà đầu tư phải trắng tay.
- Trong những năm 2010, giá vàng có xu hướng tăng nhanh. Năm 2010 thị trường vàng biến động bất thường với những mức giá kỷ lục liên tiếp được thiết lập.
- Khủng hoảng kinh tế năm 2008 vẫn có tác động đến thị trường sau 2 năm. Ngân hàng trung ương của các nước lớn đã tìm cách tăng chi tiêu đồng thời in thêm tiền mặt khiến lạm phát gia tăng. Các yếu tố quốc tế đầy rủi ro sau khủng hoảng khiến vàng lại trở thành tài sản được ưa chuộng để tích trữ.
- Bên cạnh các yếu tố cơ bản khiến giá vàng luôn theo xu thế đi lên, thì động thái găm giữ vàng của các nhà đầu tư, các chính phủ đã khiến nhu cầu vàng năm 2010 tăng vọt.

b. Phân tích giai đoạn 2015-2017

In [32]:

```
start_date = '2015-01-01'
end_date = '2017-12-31'

fig, ax = plt.subplots(figsize=(12, 7))

ax.plot(weekly_data.loc[start_date:end_date, 'Total Turnover'],
        marker='o', markersize=4, linestyle='-')

ax.set_title('Total turnover from {} to {}'.format(start_date, end_date))
ax.set_xlabel('Date')
ax.set_ylabel('Total Turnover')

plt.show()

df.reset_index(inplace=True)
```



Nhận xét

- Vào khoảng tháng 8 năm 2015, 'total turnover' của vàng tăng đột ngột.
- Những biến động trên thị trường tài chính, tiền tệ thế giới; giá dầu thô liên tiếp giảm và chưa có dấu hiệu tăng khiến cho giá vàng năm 2015 chỉ ở mức khiêm tốn. Tuy nhiên, việc giá vàng không quá cao đã thu hút mạnh những nhà đầu tư vàng trên toàn thế giới.
- Theo Báo cáo Xu hướng Nhu cầu Vàng của Hội đồng Vàng Thế giới (WGC) công bố tháng 11/2015, nhu cầu vàng toàn cầu trong quý III/2015 đạt 1,121 tấn, tăng 8% so với cùng kỳ năm 2014; chủ yếu là do nhu cầu vàng tại các thị trường lớn được cải thiện.
- Làn sóng mua vàng bắt nguồn từ hai nhân tố quan trọng là giá vàng tương đối rẻ và những biến động trên các thị trường tài chính, khiến các nhà đầu tư tìm đến kênh gửi tiền an toàn hơn.

6. Weighted Average Price (WAP)

Weighted Average Price (WAP) là một thuật ngữ tài chính dùng để chỉ mức giá trung bình mà tại đó tất cả các giao dịch của một số cổ phiếu hoặc tài sản nào đó được thực hiện trong một ngày giao dịch nhất định.

Tính toán chỉ số WAP này tính đến khối lượng của mỗi giao dịch, do đó các giao dịch có khối lượng lớn sẽ tác động lớn hơn đến giá trung bình chung.

Vai trò của WAP trong giá vàng:

- WAP giúp xác định giá vàng trung bình trong ngày đó. Cụ thể, WAP của giá vàng được tính bằng cách lấy giá mua bán vàng tại các giao dịch trong ngày đó, sau đó tính trung bình dựa trên khối lượng vàng được giao dịch tại mỗi giá.
- Việc tính toán WAP cho giá vàng hàng ngày giúp giảm thiểu sự biến động của giá vàng trong ngày, từ đó đưa ra được giá trung bình của vàng trong ngày đó. Điều này rất hữu ích đối với những người tham gia thị trường vàng, giúp họ có cái nhìn tổng quan về xu hướng giá vàng trong ngày và đưa ra quyết định giao dịch phù hợp.
- Ngoài ra, việc sử dụng WAP còn giúp đánh giá mức độ thanh khoản của thị trường vàng trong ngày đó. Nếu WAP cao hơn so với giá trung bình trong các ngày trước đó, điều này cho thấy có sự tăng trưởng trong giao dịch vàng trong ngày. Ngược lại, nếu WAP thấp hơn so với giá trung bình trong các ngày trước đó, điều này cho thấy có sự giảm sút trong giao dịch vàng trong ngày.

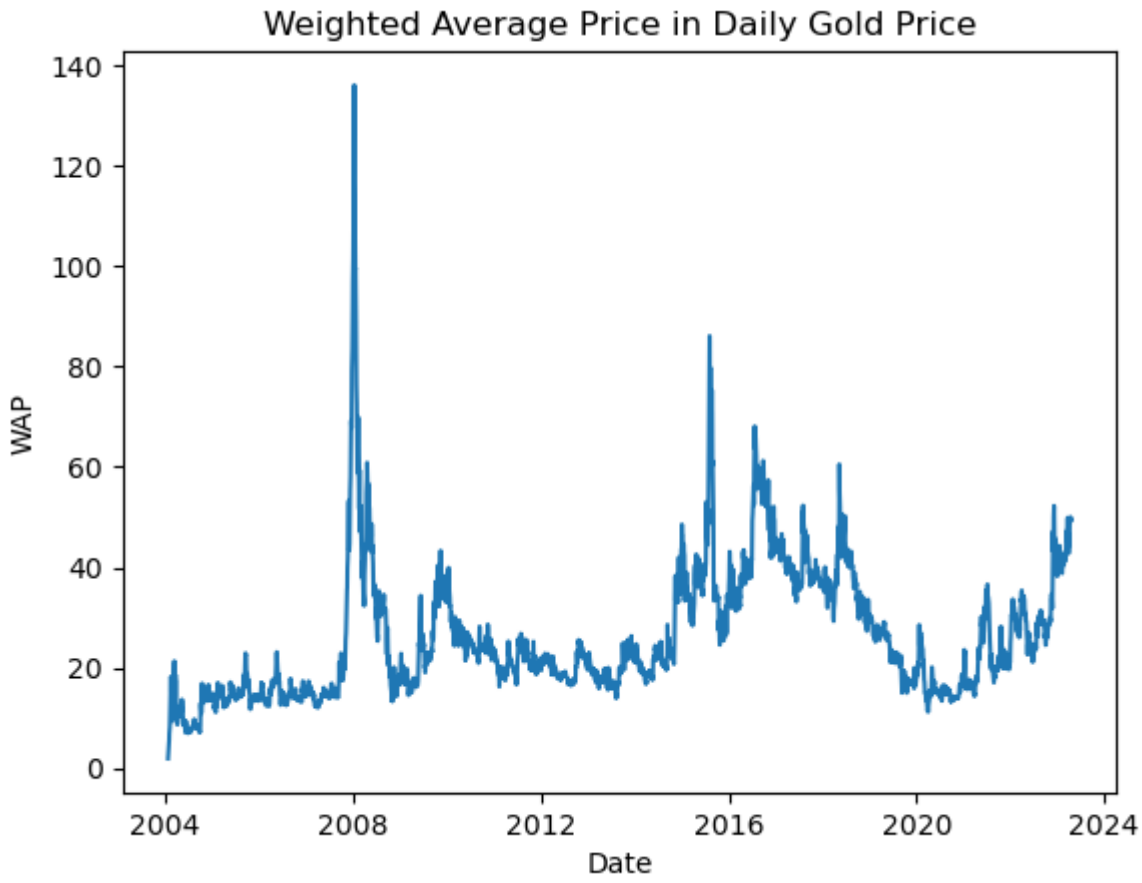
In [33]:
```
1 WAP_df = df[['Date','WAP','Year','Open']]
2 print(WAP_df)
```

```
           Date    WAP  Year   Open
0    2004-01-20   1.92  2004   1.92
1    2004-01-21   2.30  2004   2.30
2    2004-01-22   2.76  2004   2.76
3    2004-01-23   3.31  2004   3.31
4    2004-01-27   3.97  2004   3.97
...         ...    ...   ...    ...
4770 2023-04-06  48.80  2023  48.40
4771 2023-04-10  49.90  2023  51.49
4772 2023-04-11  49.21  2023  48.87
4773 2023-04-12  49.99  2023  49.98
4774 2023-04-13  49.35  2023  50.47

[4775 rows x 4 columns]
```

In [34]:

```
# Đặt cột "Date" làm chỉ mục của dataframe
WAP_df.set_index("Date", inplace=True)

print(WAP_df)

# Vẽ biểu đồ line plot cho WAP
plt.plot(WAP_df.index, WAP_df["WAP"])

# Đặt tiêu đề và tên trục
plt.title("Weighted Average Price in Daily Gold Price")
plt.xlabel("Date")
plt.ylabel("WAP")

# Hiển thị biểu đồ
plt.show()

WAP_df.reset_index(inplace=True)
```

```
              WAP  Year   Open
Date
2004-01-20   1.92  2004   1.92
2004-01-21   2.30  2004   2.30
2004-01-22   2.76  2004   2.76
2004-01-23   3.31  2004   3.31
2004-01-27   3.97  2004   3.97
...           ...   ...    ...
2023-04-06  48.80  2023  48.40
2023-04-10  49.90  2023  51.49
2023-04-11  49.21  2023  48.87
2023-04-12  49.99  2023  49.98
2023-04-13  49.35  2023  50.47

[4775 rows x 3 columns]
```



Nhận xét

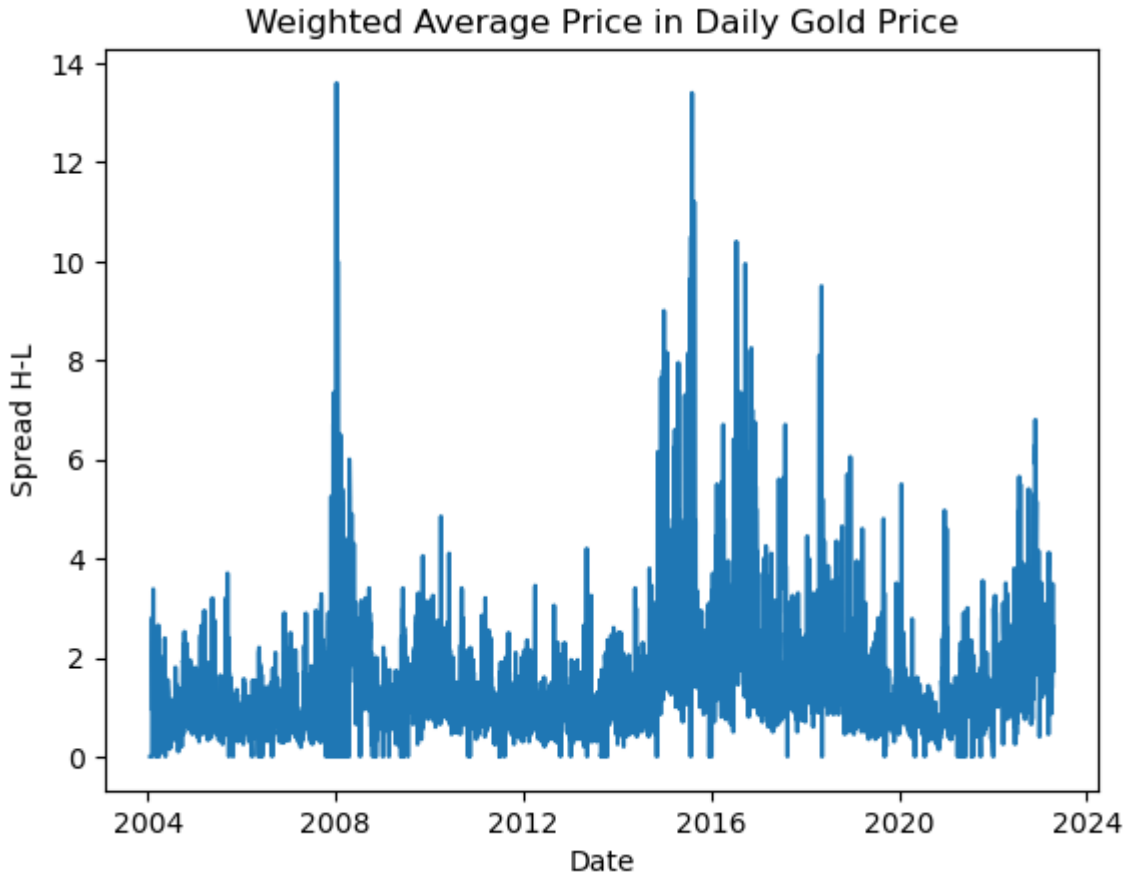
Dựa trên biểu đồ Line Chart của WAP ta có thể rút ra một số nhận xét như sau:

- Từ giữa năm 2015 đến năm 2016 giá vàng tăng mạnh và khá liên tục, tuy nhiên ngay sau đó giá vàng cũng đã có xu hướng giảm mạnh sau một khoảng thời gian ngắn và lại tiếp tục tăng tuy nhiên dao động tăng nhẹ hơn rất nhiều so với nữa đầu năm 2015
- Dựa vào WAP chúng ta có thể thấy được có những thời điểm trong năm giá vàng giảm như khoảng từ tháng 2 đến tháng 3 hoặc giữa tháng 4 đến đầu tháng 5 thì đó là những tháng mà giá vàng có xu hướng giảm
- Từ khoảng năm 2017 cho đến đầu năm 2020 thì giá vàng có xu hướng giảm đều qua các năm và biên độ dao động của nó cũng khá cao tuy nhiên từ sau nữa đầu năm 2020 thì giá vàng lại đang có xu hướng tăng trưởng trở lại
- Đây là một yếu tố quan trọng để người đầu tư có thể đánh giá được khoảng thời gian mà giá vàng thường xuyên dao động mạnh hay những khoảng thời gian mà nó tăng để đưa ra quyết định phù hợp
- Tuy nhiên WAP chỉ là một phần để đánh giá thôi, là người đầu tư thì cần phải dựa trên nhiều yếu tố khác như các chỉ báo kỹ thuật, yếu tố kinh tế, yếu tố chính trị và tình hình thị trường toàn cầu

7. The difference between the highest and lowest prices of the gold share on that day (Spread H-L)

- Đây là chỉ số thể hiện tính biến động của giá vàng trong ngày
- Nếu chỉ số Spread H-L lớn thì nó có nghĩa là giá vàng dao động mạnh trong ngày, có thể là do nhiều yếu tố ảnh hưởng đến giá, ví dụ như thông tin trong ngành, các sự kiện toàn cầu, hoặc sự thay đổi của tỷ giá.
- Thông qua việc phân tích Spread H-L, nhà đầu tư có thể đưa ra quyết định mua hoặc bán vàng vào khi nào. Nếu Spread H-L lớn thì nhà đầu tư có thể quyết định mua vào với giá thấp và bán ra với giá cao hơn, tận dụng được sự giao động của giá để kiếm về lợi nhuận.Ngược lại nếu Spread H-L thấp có thể là do giá vàng ổn định và ít có sự biến động nhà đầu tư có thể quyết định giữ cổ phiếu trong thời gian dài để đạt được lợi nhuận ổn định hơn.

In [35]:
```
Spread_H_L_df = df[['Date','Spread H-L']]

# Đặt cột "Date" làm chỉ mục của dataframe
Spread_H_L_df.set_index("Date", inplace=True)

# Vẽ biểu đồ line plot cho WAP
plt.plot(Spread_H_L_df.index, Spread_H_L_df["Spread H-L"])

# Đặt tiêu đề và tên trục
plt.title("Weighted Average Price in Daily Gold Price")
plt.xlabel("Date")
plt.ylabel("Spread H-L")

# Hiển thị biểu đồ
plt.show()
Spread_H_L_df.reset_index(inplace=True)
```



Nhận xét

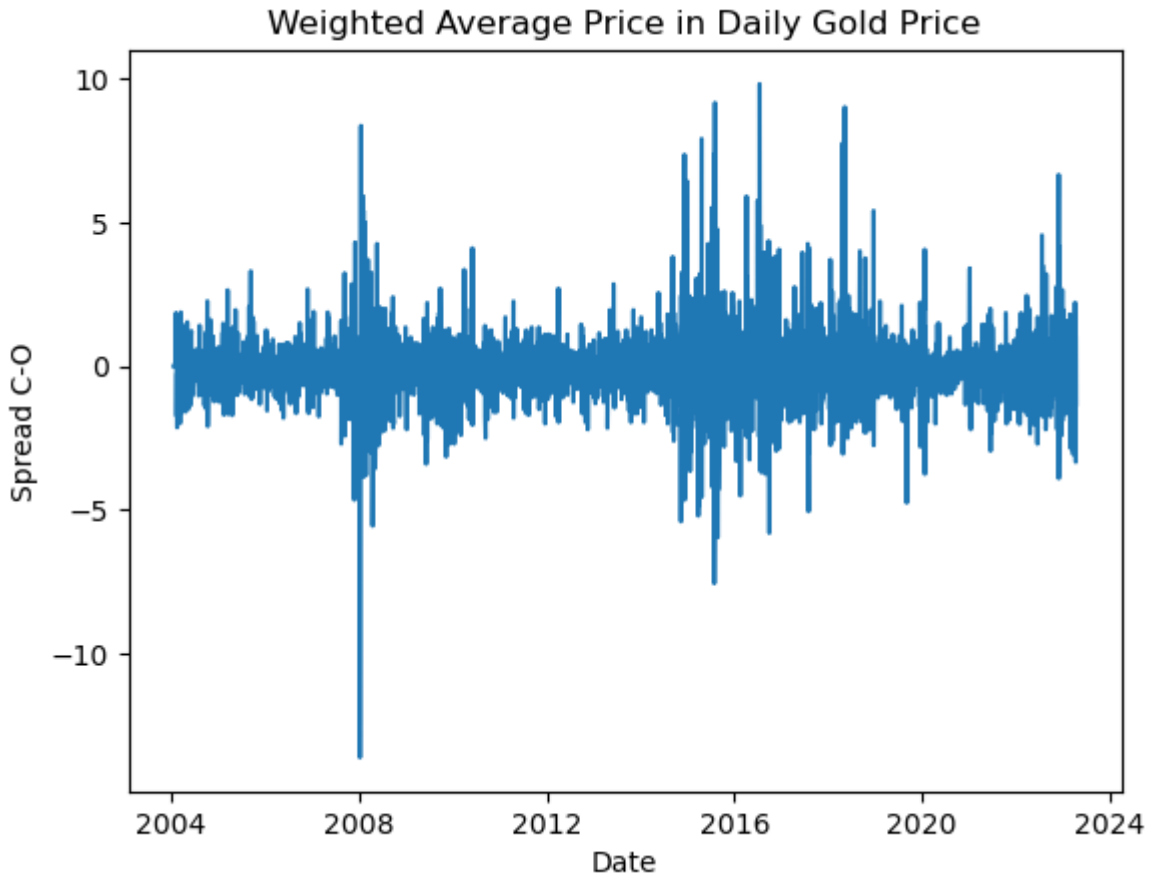
- Việc biểu diễn Spread H-L bằng line chart giúp chúng ta có thể dễ dàng quan sát và phân tích sự thay đổi của độ chênh lệch giá trong thời gian. Chẳng hạn, nếu giá trị của Spread H-L tăng lên đột ngột trong một ngày cụ thể, chúng ta có thể xem xét các yếu tố gây ra sự thay đổi này, ví dụ như thông tin về các biến động kinh tế, chính sách hay sự kiện xảy ra trong ngành đó. Điều này giúp chúng ta có thể đưa ra quyết định đầu tư hợp lý và hiệu quả hơn.
- Giá trị Spread H-L càng lớn thì chênh lệch giá giữa ngày cao nhất và thấp nhất càng lớn, điều này thường cho thấy sự biến động lớn trong giá cổ phiếu. Điều này có thể là do thị trường đang trong tình trạng không ổn định, hoặc có những tin tức, thông tin tác động đến giá cổ phiếu, tạo ra sự chênh lệch giá lớn. Tuy nhiên, cần lưu ý rằng chênh lệch giá lớn không đồng nghĩa với giá trị cổ phiếu tăng hoặc giảm, và ngược lại, giá trị cổ phiếu có thể tăng hoặc giảm mà chênh lệch giá không lớn. Do đó, cần phân tích và đánh giá nhiều yếu tố khác nhau để đưa ra quyết định đầu tư chính xác.
- Nếu giá trị Spread H-L nhỏ, có nghĩa là chênh lệch giá giữa ngày cao nhất và thấp nhất không quá lớn. Điều này có thể cho thấy thị trường đang ổn định hoặc có ít tác động từ thông tin và tin tức.

Tuy nhiên, cần phân tích kỹ các yếu tố khác như thanh khoản, xu hướng chung của thị trường để

8. The difference between the closing and opening prices of the gold share on that day. (Spread C-O)

- Spread C-O là một thuộc tính trong phân tích kỹ thuật thị trường tài chính.
- Spread C-O được tính bằng công thức: **Spread C-O = Giá đóng cửa - Giá mở cửa**
- Spread C-O thường được sử dụng để đánh giá sự khác biệt giữa giá đóng cửa và giá mở cửa của một tài sản trong một ngày giao dịch
  - Nếu Spread C-O có giá trị dương, nghĩa là giá trị đóng cửa cao hơn giá trị mở cửa nghĩa là tài sản đó tăng giá trong ngày đó
  - Nếu Spread C-O có giá trị âm, nghĩa là giá trị đóng cửa thấp hơn giá trị mở cửa nghĩa là tài sản đó giảm giá trong ngày đó

In [36]:
```
Spread_C_O_df = df[['Date','Spread C-O']]

# Đặt cột "Date" làm chỉ mục của dataframe
Spread_C_O_df.set_index("Date", inplace=True)

# Vẽ biểu đồ line plot cho WAP
plt.plot(Spread_C_O_df.index, Spread_C_O_df["Spread C-O"])

# Đặt tiêu đề và tên trục
plt.title("Weighted Average Price in Daily Gold Price")
plt.xlabel("Date")
plt.ylabel("Spread C-O")

# Hiển thị biểu đồ
plt.show()
Spread_C_O_df.reset_index( inplace=True)
```



Nhận xét

- Dựa vào biểu đồ line chart ta có thể nhận thấy được sự biến động của Spread C-O từng ngày trong thời gian nghiên cứu. Trong biểu đồ line chart ở trên chúng ta có thể thấy có nhiều ngày mà Spread C-O rất lớn thì những ngày đó là những ngày biến động về giá cổ phiếu.
- Tuy nhiên biểu đồ này không cho thấy sự tương quan hoặc ảnh hưởng của các yếu tố khác trong dữ liệu lên giá trị Spread C-O, do đó cần phải có sự kết hợp với các biểu đồ khác để có cái nhìn

Trực quan WAP (time series)

In [37]:

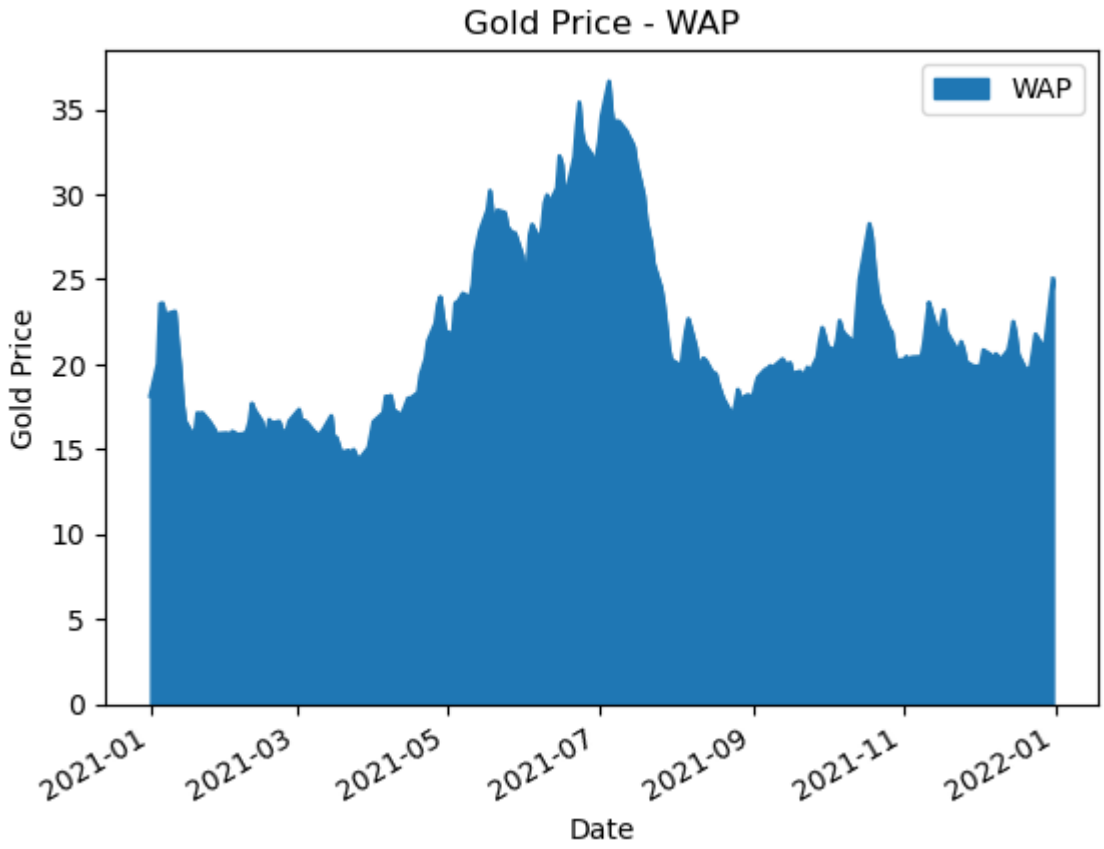
```
# tạo bảng vẽ với 2 khung, 1 hàng và 2 cột

WAP_df_2021 = WAP_df[WAP_df['Year']==2021]

WAP_df_2022 = WAP_df[WAP_df['Year']==2022]

# 2021
WAP_df_2021['month'] = WAP_df_2021['Date'].dt.month
WAP_df_2021['day'] = WAP_df_2021['Date'].dt.day
# 2022
WAP_df_2022['month'] = WAP_df_2022['Date'].dt.month
WAP_df_2022['day'] = WAP_df_2022['Date'].dt.day

# Trực quan năm 2021
# Select the columns to plot
columns = ["Date", "WAP"]

# Create a new dataframe with only the selected columns
subset = WAP_df_2021[columns]

# Set the "Date" column as the index
subset = subset.set_index("Date")

# Create the area chart
ax = subset.plot.area()

# Add labels and title
ax.set_xlabel("Date")
ax.set_ylabel("Gold Price")
ax.set_title("Gold Price - WAP")


# Trực quan time series
merged_df = pd.merge(WAP_df_2021, WAP_df_2022, on=['month', 'day'], how='inner')
merged_df = merged_df.dropna(subset=['day', 'month'])
merged_df['index'] = merged_df['Date_x'].dt.strftime('%m-%d')
# Sử dụng phương thức rename để đổi tên cột
merged_df = merged_df.rename(columns={'WAP_x': 'WAP_2021'})
merged_df = merged_df.rename(columns={'WAP_y': 'WAP_2022'})
merged_df = merged_df.iloc[::-1]
print(merged_df)

columns = ['index','WAP_2021','WAP_2022']
subset = merged_df[columns]
# Set the "Date" column as the index
subset = subset.set_index("index")

# Create the area chart
ax = subset.plot.area()

# Add labels and title
ax.set_xlabel("Date")
ax.set_ylabel("Gold Price")
ax.set_title("Gold Price - WAP_2021 vs WAP_2022")

# Show the chart
plt.show()
```

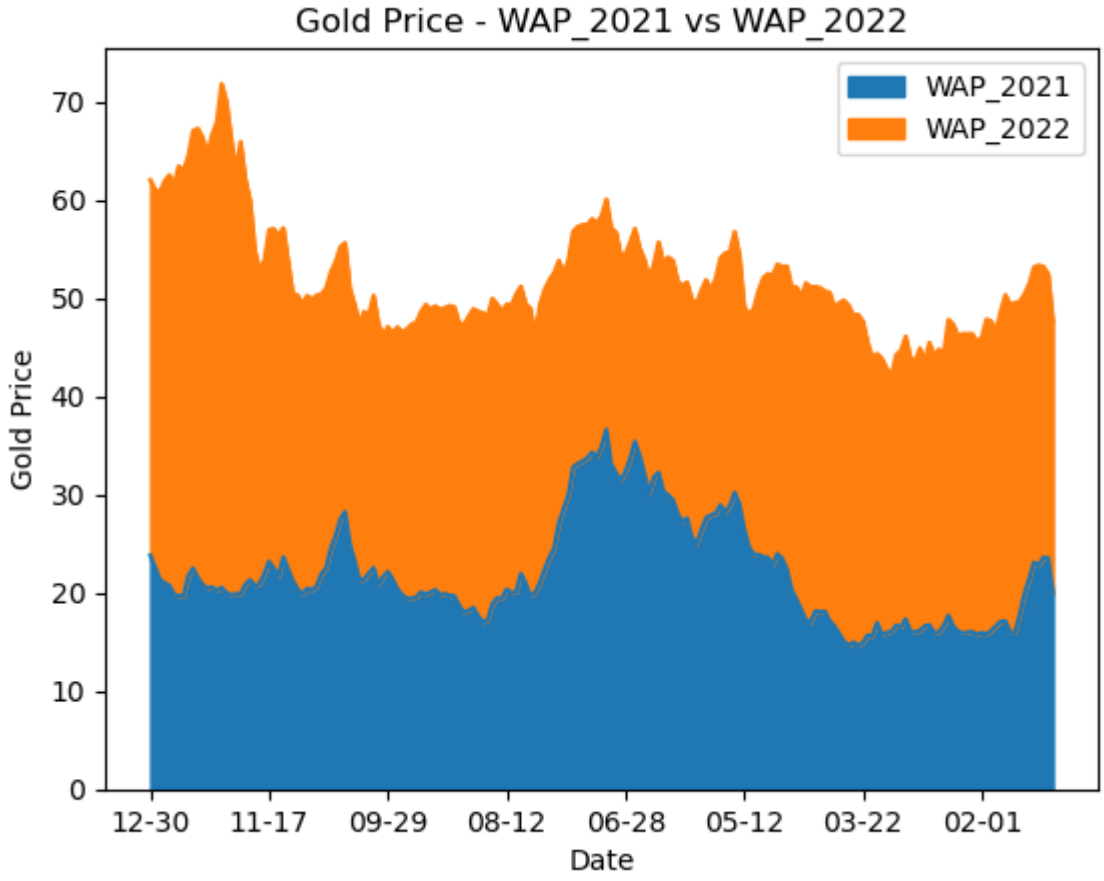
```
C:\Users\Dat Phan\AppData\Local\Temp\ipykernel_9172\4039575239.py:8: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  WAP_df_2021['month'] = WAP_df_2021['Date'].dt.month
C:\Users\Dat Phan\AppData\Local\Temp\ipykernel_9172\4039575239.py:9: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  WAP_df_2021['day'] = WAP_df_2021['Date'].dt.day
C:\Users\Dat Phan\AppData\Local\Temp\ipykernel_9172\4039575239.py:11: SettingWithCop
yWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  WAP_df_2022['month'] = WAP_df_2022['Date'].dt.month
C:\Users\Dat Phan\AppData\Local\Temp\ipykernel_9172\4039575239.py:12: SettingWithCop
yWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  WAP_df_2022['day'] = WAP_df_2022['Date'].dt.day
```

	Date_x	WAP_2021	Year_x	Open_x	month	day	Date_y	WAP_2022	\
190	2021-12-30	23.81	2021	23.45	12	30	2022-12-30	38.24	
189	2021-12-29	22.62	2021	22.00	12	29	2022-12-29	38.42	
188	2021-12-28	21.35	2021	20.90	12	28	2022-12-28	39.46	
187	2021-12-27	20.99	2021	21.55	12	27	2022-12-27	40.93	
186	2021-12-23	20.74	2021	20.00	12	23	2022-12-23	41.81	
..	...	...	...	...	...	...	...	...	
4	2021-01-11	23.09	2021	24.25	1	11	2022-01-11	30.06	
3	2021-01-07	22.82	2021	22.05	1	7	2022-01-07	30.53	
2	2021-01-06	23.58	2021	24.35	1	6	2022-01-06	29.64	
1	2021-01-05	23.54	2021	20.45	1	5	2022-01-05	28.95	
0	2021-01-04	19.99	2021	18.00	1	4	2022-01-04	27.60	

	Year_y	Open_y	index
190	2022	38.50	12-30
189	2022	38.05	12-29
188	2022	40.50	12-28
187	2022	42.50	12-27
186	2022	43.60	12-23
..	...	...	...
4	2022	32.00	01-11
3	2022	31.85	01-07
2	2022	30.35	01-06
1	2022	28.95	01-05
0	2022	27.60	01-04

```
[191 rows x 11 columns]
```





Nhận xét

- Cái nhìn đầu tiên có thể thấy rõ rằng mặt bằng chung của WAP năm 2022 cũng có sự thay đổi so với năm 2021
- Có những tháng mà biên độ của nó tăng rõ rệt so với cùng tháng năm trước điều đó chứng tỏ ví dụ như đầu tháng 12 của năm 2021 và năm 2022 điều đó cho thấy rằng khoảng thời điểm đó có sự thay đổi lớn có thể trong chính sách về tiền tệ hay về cung và cầu, hoặc khoảng thời gian đó có biến động về kinh tế, từ đó mà nhà kinh tế có thể cân nhắc khi đầu tư vào vàng.

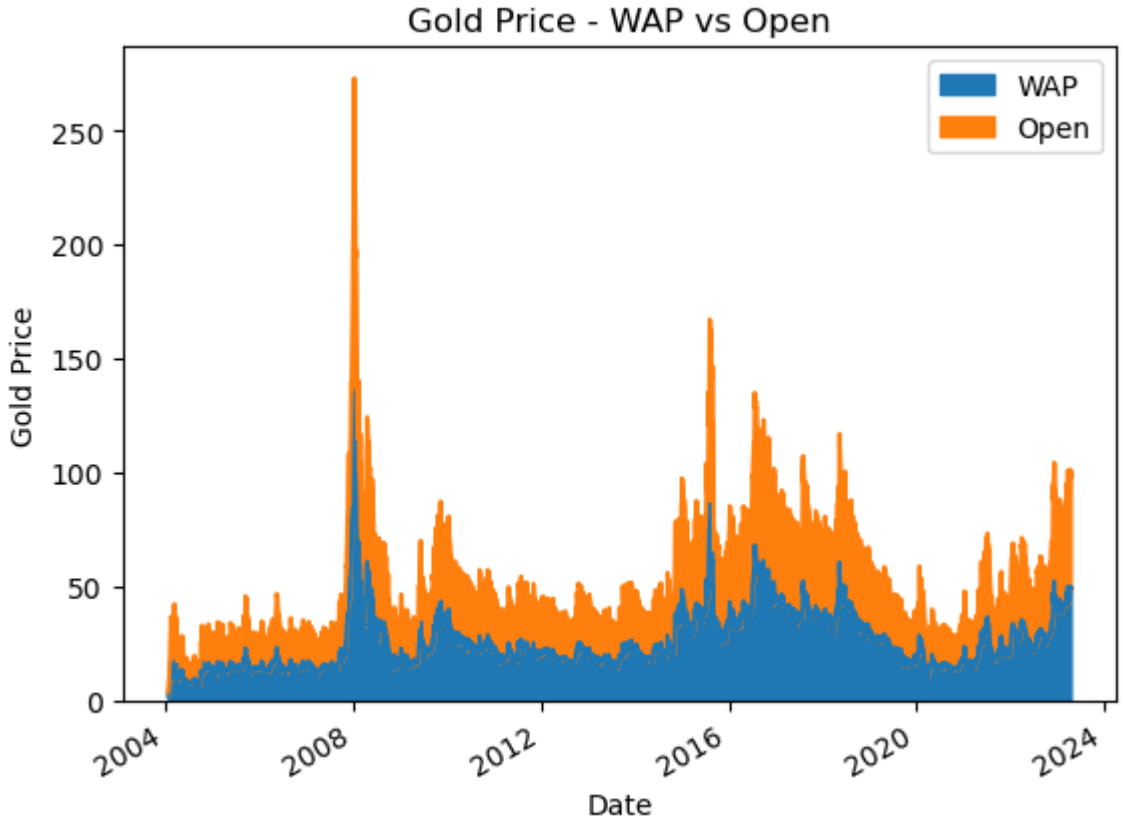
- Từ biểu đồ trên chúng ta sẽ có cái nhìn tổng thể rằng trong năm 2022 có bước phát triển trong giá vàng.
    - **Nhu cầu mua vàng của người dân**: tăng điều đó chứng tỏ rằng họ tin tưởng của người dân vào giá trị lưu trữ của vàng.
    - **Sự ổn định về kinh tế**: WAP tăng đều so với cùng tháng năm ngoái thì điều này cũng cho thấy được sự ổn định của kinh tế. Việc giá vàng tăng cũng là thường cho thấy được sự lo ngại về tương lai và sự không ổn định về nền kinh tế, tuy nhiên ở đây là chỉ số WAP của vàng tăng điều này cho thấy sự tin tưởng vào tương lai và sự ổn định của nền kinh tế
    - **Ảnh hưởng của chính sách tiền tệ**: Giá vàng có thể bị ảnh hưởng bởi các chính sách tiền tệ của các quốc gia. Nếu WAP vàng tăng, điều này có thể phản ánh sự giảm giá của tiền tệ, sự nới lỏng chính sách tiền tệ hoặc các yếu tố khác liên quan đến chính sách tiền tệ.
    - Tuy nhiên, việc đánh giá tình hình kinh tế chỉ dựa trên một chỉ số như WAP vàng là không đầy đủ. Cần phải kết hợp với các chỉ số kinh tế khác để có cái nhìn toàn diện hơn về tình hình kinh

Trực quan WAP với Open

In [38]:

```
# Area chart

# Select the columns to plot
columns = ["Date","WAP", "Open"]

# Create a new dataframe with only the selected columns
print(WAP_df)
subset = WAP_df[columns]

# Set the "Date" column as the index
subset = subset.set_index("Date")

# Create the area chart
ax = subset.plot.area()

# Add labels and title
ax.set_xlabel("Date")
ax.set_ylabel("Gold Price")
ax.set_title("Gold Price - WAP vs Open")

# Show the chart
plt.show()
```

```
           Date    WAP  Year   Open
0    2004-01-20   1.92  2004   1.92
1    2004-01-21   2.30  2004   2.30
2    2004-01-22   2.76  2004   2.76
3    2004-01-23   3.31  2004   3.31
4    2004-01-27   3.97  2004   3.97
...         ...    ...   ...    ...
4770 2023-04-06  48.80  2023  48.40
4771 2023-04-10  49.90  2023  51.49
4772 2023-04-11  49.21  2023  48.87
4773 2023-04-12  49.99  2023  49.98
4774 2023-04-13  49.35  2023  50.47

[4775 rows x 4 columns]
```


Gold Price - WAP vs Open
WAP
Open
Gold Price
250
200
150
100
50
0
2004
2008
2012
2016
2020
2024
Date

Nhận xét

- Việc so sánh giữa hai biến WAP và Open giúp chúng ta hiểu hơn về sự biến động của giá vàng trong ngày.
- Ở đây chúng ta có thể thấy Giá mở cửa (Open) luôn luôn lớn hơn giá trị trung bình trong ngày (WAP) điều đó có nghĩa là giá vàng luôn tăng trong một thời gian ngắn và giảm trong khoảng thời gian còn lại của một phiên giao dịch.
- Bên cạnh đó việc so sánh giá mở cửa (Open) và giá trị trung bình (WAP) trong ngày cũng có thể giúp chúng ta phát hiện ra những điều khác biệt giữa các giao dịch vàng, giúp các nhà đầu tư vàng đưa ra quyết định đầu tư vàng hiệu quả hơn.

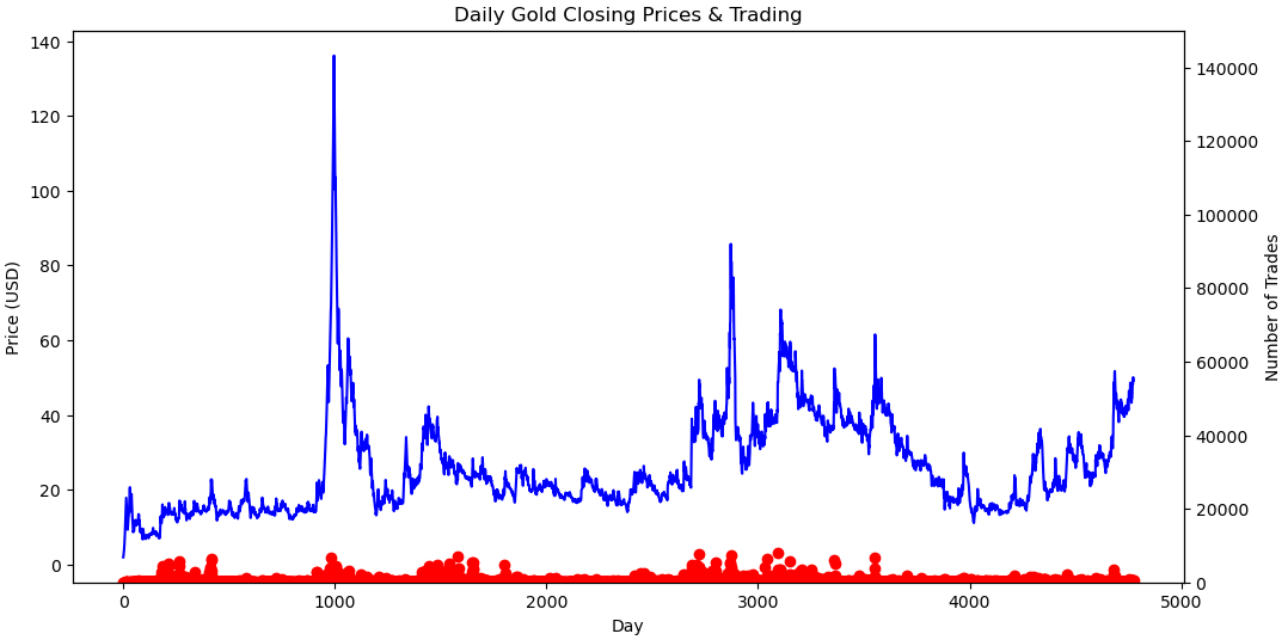
Dùng kỹ thuật Elide Data để lược bỏ bớt Data

- Loại bỏ một phần dữ liệu và giá trị trong tập dữ liệu mà không ảnh hưởng tới tính toàn vẹn và ý nghĩa của dataset đó
- Elide data có thể thực hiện bằng cách lược bỏ số hàng hoặc số cột trong dataset hoặc loại bỏ các giá trị ngoại lai
- Loại bỏ nhiều dữ liệu có thể khiến mất mát thông tin quan trọng
- Trong dataset mà bọn em đã chọn đó là về giá vàng, thì trong dataset này nếu có elide data chúng em sẽ lược bỏ một số ngày mà những ngày đó là những ngày mà thị trường vàng trên thế giới không hoạt động, thường thì đó sẽ là những ngày lễ lớn
- Tuy nhiên trong dataset này là dataset của những ngày có xảy ra giao dịch trên thị trường vì thế việc Elide data là không cần thiết

Trực quan Closing Prices and Gold Trading Volumne

In [39]:

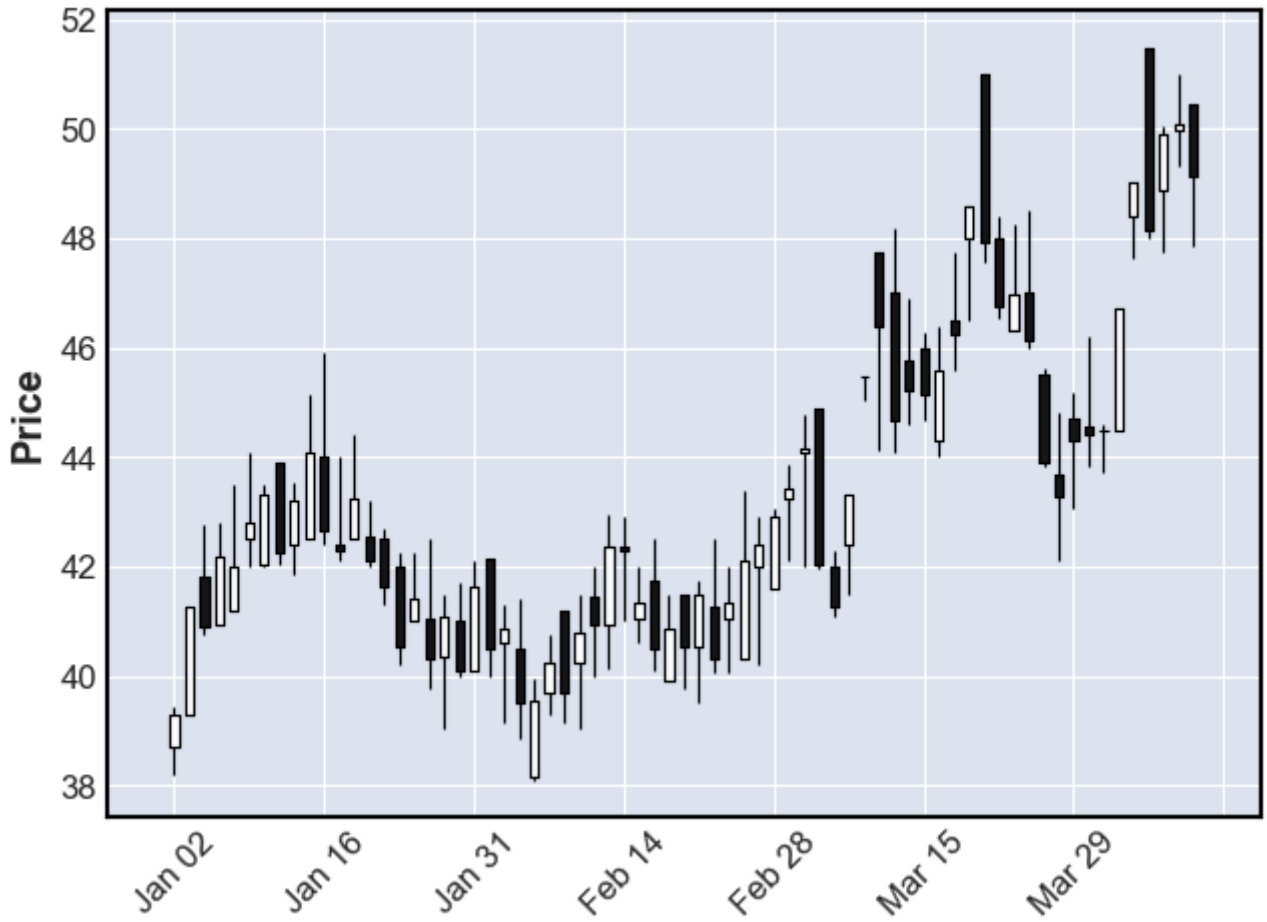
```
# extract the relevant columns
close_prices_df = df["Close"]
trading_df = df["No. of Trades"]

# create a figure and axis object
fig, ax1 = plt.subplots(figsize=(12, 6))

# plot the daily closing prices of gold as a line on the first axis
ax1.plot(close_prices_df, color="blue")
ax1.set_xlabel("Day")
ax1.set_ylabel("Price (USD)")
ax1.set_title("Daily Gold Closing Prices & Trading")

# create a second axis object with a twin y-axis for the scatter plot
ax2 = ax1.twinx()
ax2.set_ylim(0, 150000)

# overlay the daily trading volumes of gold as a scatter plot on the second axis
ax2.scatter(df.index, trading_df, color="red")
ax2.set_ylabel("Number of Trades")

plt.show()
```


Daily Gold Closing Prices & Trading
Price (USD)
Number of Trades
Day
0
20
40
60
80
100
120
140
0
20000
40000
60000
80000
100000
120000
140000
0
1000
2000
3000
4000
5000

In [40]:

```
import mplfinance as mpf

# Set the index to the Date column
df_2021 = df[df['Year']==2023]
df_2021.reset_index(inplace=True)
df_2021['month'] = df_2021['Date'].dt.month
# df_2021 = df_2021[df_2021['month']>9]
# df_2021.reset_index(inplace=True)
df_2021["Date"] = pd.to_datetime(df_2021["Date"])
df_2021.set_index("Date", inplace=True)

# Create a new DataFrame with just the columns we need
ohlc = df_2021[["Open", "High", "Low", "Close", "WAP"]]
wap = df_2021['WAP']

# Set the style options for the WAP line chart
line_style = {'color': 'blue', 'width': 0.5}

mpf.plot(ohlc, type="candle", volume=False, show_nontrading=False)

mpf.plot(ohlc, type="candle", volume=False, show_nontrading=False, addplot=mpf.mal

df_2021.reset_index(inplace=True)
```

```
C:\Users\Dat Phan\AppData\Local\Temp\ipykernel_9172\4075578471.py:6: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df_2021['month'] = df_2021['Date'].dt.month
C:\Users\Dat Phan\AppData\Local\Temp\ipykernel_9172\4075578471.py:9: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
ser_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df_2021["Date"] = pd.to_datetime(df_2021["Date"])
```



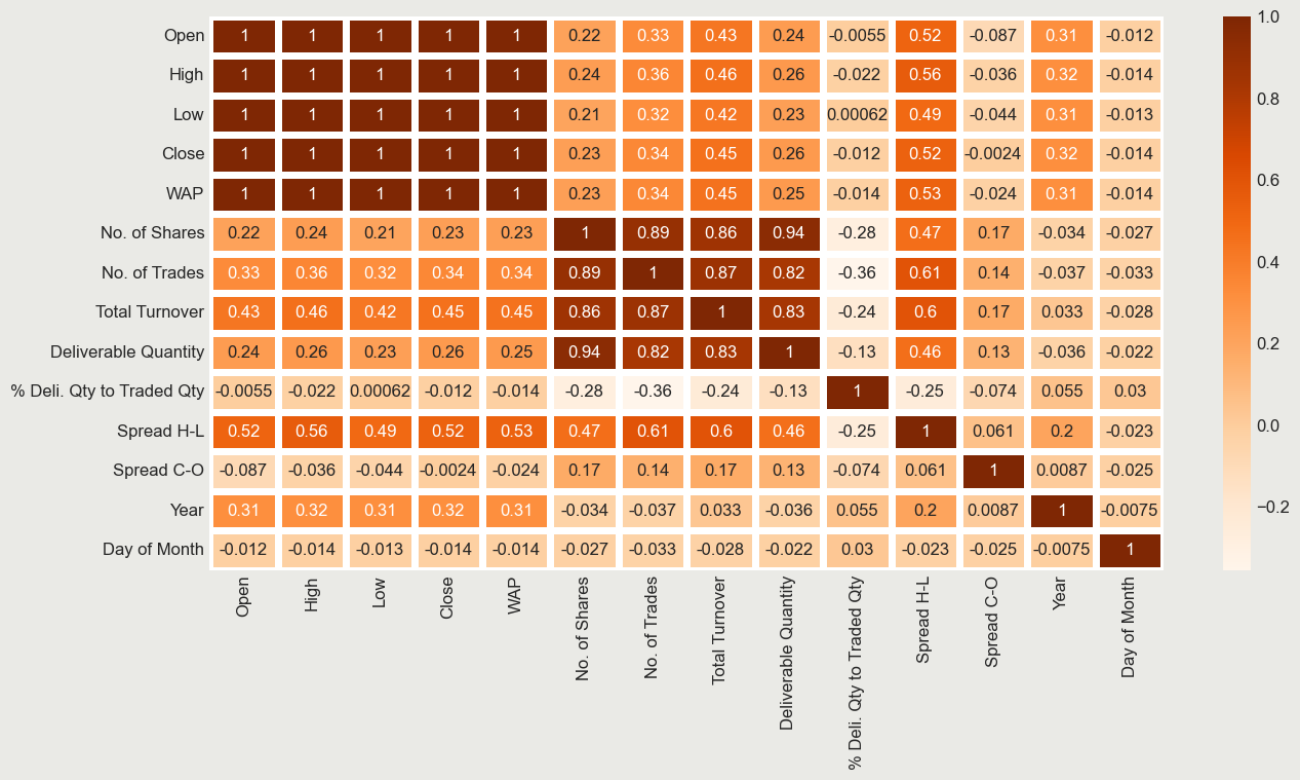
Nhận xét

- Biểu đồ thứ nhất là biểu đồ hình nến, biểu đồ này thể hiện sự biến động giá của tài sản trong một khoản thời gian nào đó

- Đầu tiên nhìn vào biểu đồ thứ 1 chúng ta có thể thấy được trục x hiển thị khoảng thời gian, trong khi trục y hiển thị các mức giá. Mỗi thanh nến riêng lẻ trên biểu đồ đại diện cho giá mở cửa, cao nhất, thấp nhất và đóng cửa trong một khoảng thời gian cụ thể.
- Màu sắc của nến cũng có thể cung cấp thông tin quan trọng về biến động giá.Nếu một thanh nến có màu trắng điều đó thường có nghĩa là giá đóng cửa cao hơn giá mở cửa, cho thấy rằng người mua đang kiểm soát và đẩy giá lên cao hơn. Nếu một thanh nến có màu đen, điều đó thường có nghĩa là giá đóng cửa thấp hơn giá mở cửa, cho thấy rằng người bán đang kiểm soát và đẩy giá xuống thấp hơn.
- Biểu đồ thứ 2 cũng là biểu đồ hình nến tuy nhiên ở đây em đã có sử dụng kỹ thuật Superimpose Layer, nghĩa là sẽ chồng các biểu đồ lên nhau, em đã sử dụng biểu đồ **line chart**, line chart trong biểu đồ thứ 2 thể hiện chỉ số WAP của giá vàng cùng trên một trục thời gian
- Việc tích hợp line chart của chỉ số WAP vào trong biểu đồ này giúp người đầu tư có thể dễ dàng so sánh mức giá trung bình trên mỗi phiên giao dịch. Với chỉ mỗi biểu đồ hình nến thôi thì chúng ta sẽ khó mà có thể nhận biết được mức giá trung bình của các phiên giao dịch. Tuy nhiên với việc kết hợp hai loại biểu đồ này lại với nhau thì giúp người đầu tư vàng có cái nhìn tổng quan hơn về mức giá của mỗi phiên giao dịch
- Ví dụ nếu line chart của WAP đang tăng dần và giá đóng cửa của mỗi cây nến trong biểu đồ hình nến đó cũng tăng, thì điều này cho thấy người dân đang có nhu cầu mua cao lên,ngược lại nếu đường WAP đang giảm dần trong khi giá đóng cửa của hình nến đang tăng thì điều này cho thấy sự bán ra đang tăng lên.

Mối tương quan giữa 'Deliverable Quantity' và '% Deli. Qty to Traded Qty'

In [41]:
```
df.set_index('Date', inplace=True)
#Correlation Map
plt.figure(figsize = [15, 7], clear = True, facecolor = '#EAEAE6')
sns.heatmap(df.corr(), annot = True, square = False, linewidths = 5,
            linecolor = "white", cmap = "Oranges");
df.reset_index(inplace=True)
```



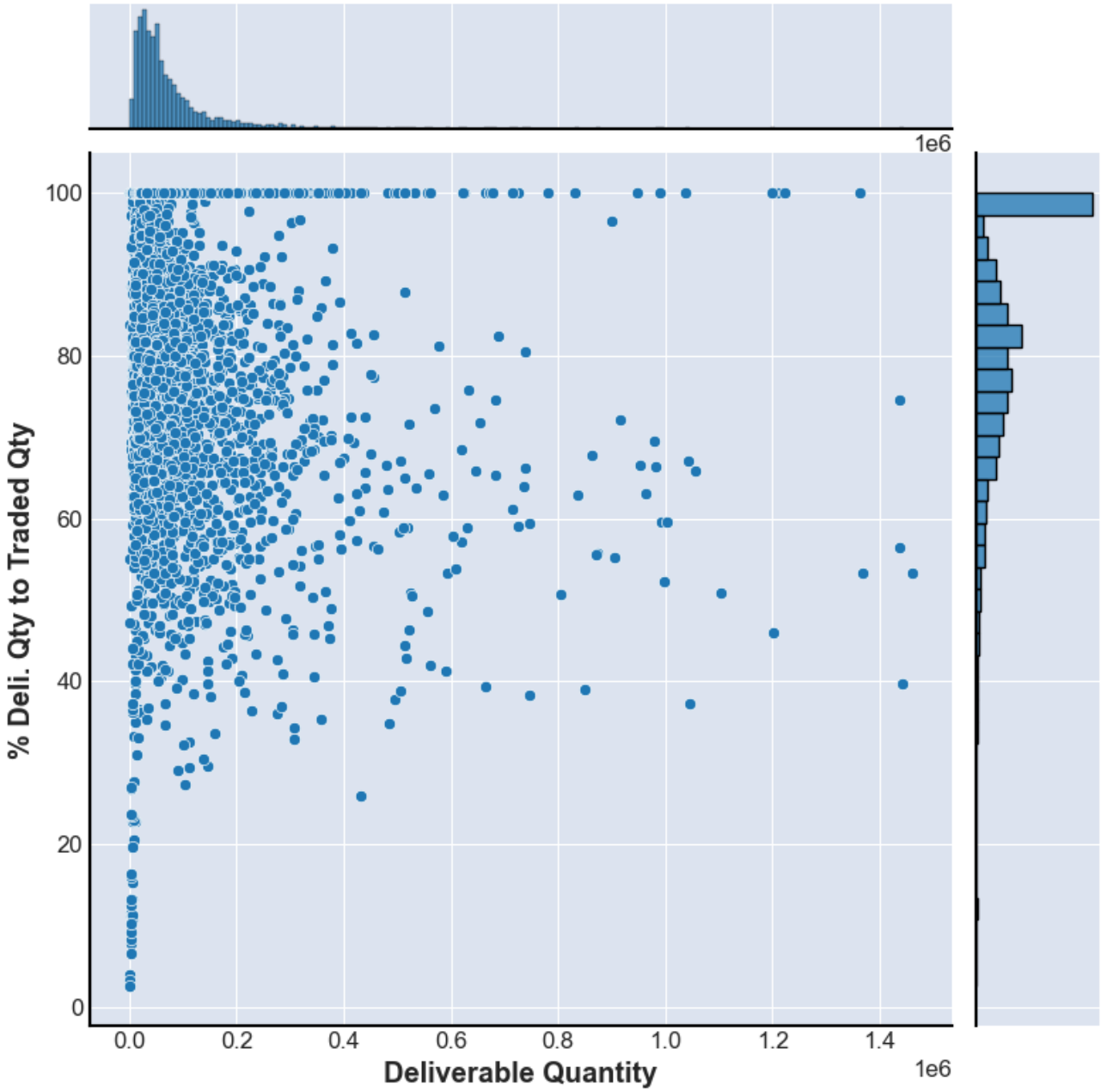
Nhận Xét:

- Deliverable Quantity và % Deli. Qty to Traded Qty có mối tương quan khá ít

Sau khi sử dụng heatmap để hình dung sơ bộ sự tương quan của 'Deliverable Quantity' và '% Deli. Qty to Traded Qty', dưới đây là thể hiện rõ ràng sự tương quan của 2 thuộc tính:

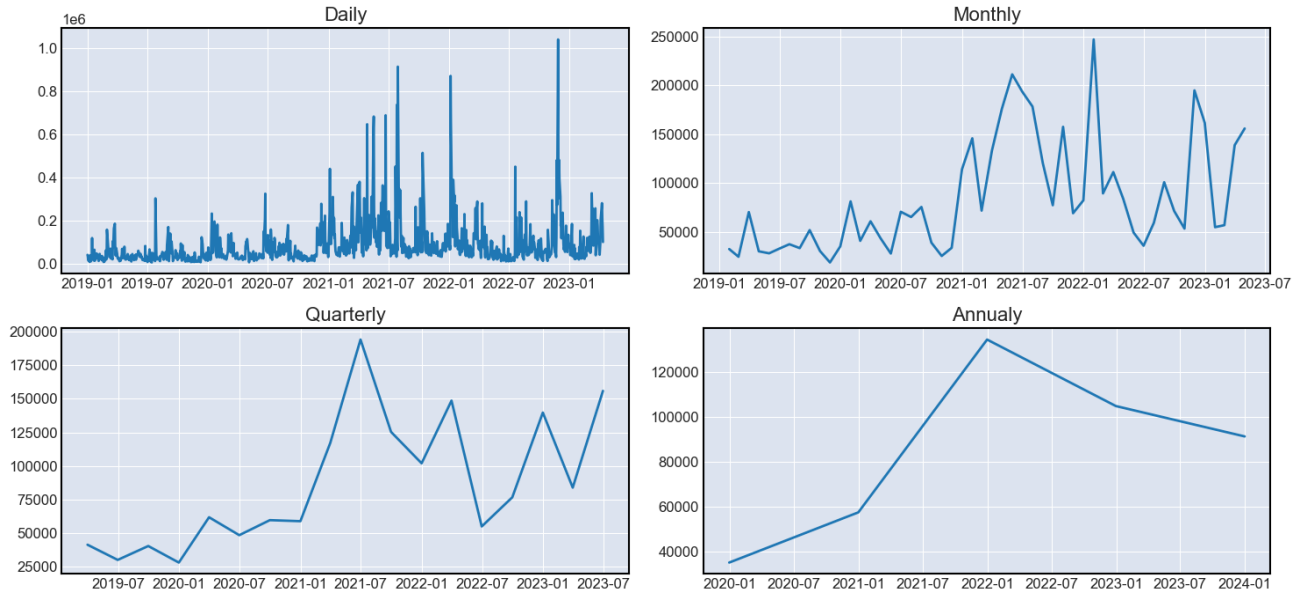
In [42]:

```
df.set_index('Date', inplace=True)
sns.jointplot(
    x = "Deliverable Quantity",
    y = "% Deli. Qty to Traded Qty",
    data = df, height = 8, ratio = 6, kind = "scatter"
);
df.reset_index(inplace=True)
```



9. Deliverable Quantity

In [43]:
```
df.set_index('Date', inplace=True)

fig,axes = plt.subplots(2,2,figsize=[15,7])

## resampling to daily freq (original data)
axes[0,0].plot(data['Deliverable Quantity'])
axes[0,0].set_title("Daily",size=16)

## resampling to monthly freq
axes[0,1].plot(data['Deliverable Quantity'].resample('M').mean())
axes[0,1].set_title("Monthly",size=16)

## resmapling to quarterly freq
axes[1,0].plot(data['Deliverable Quantity'].resample('Q').mean())
axes[1,0].set_title('Quarterly',size=16)

## resampling to annualy freq
axes[1,1].plot(data['Deliverable Quantity'].resample('A').mean())
axes[1,1].set_title('Annualy',size=16)

plt.tight_layout()
plt.show()

df.reset_index(inplace=True)
```

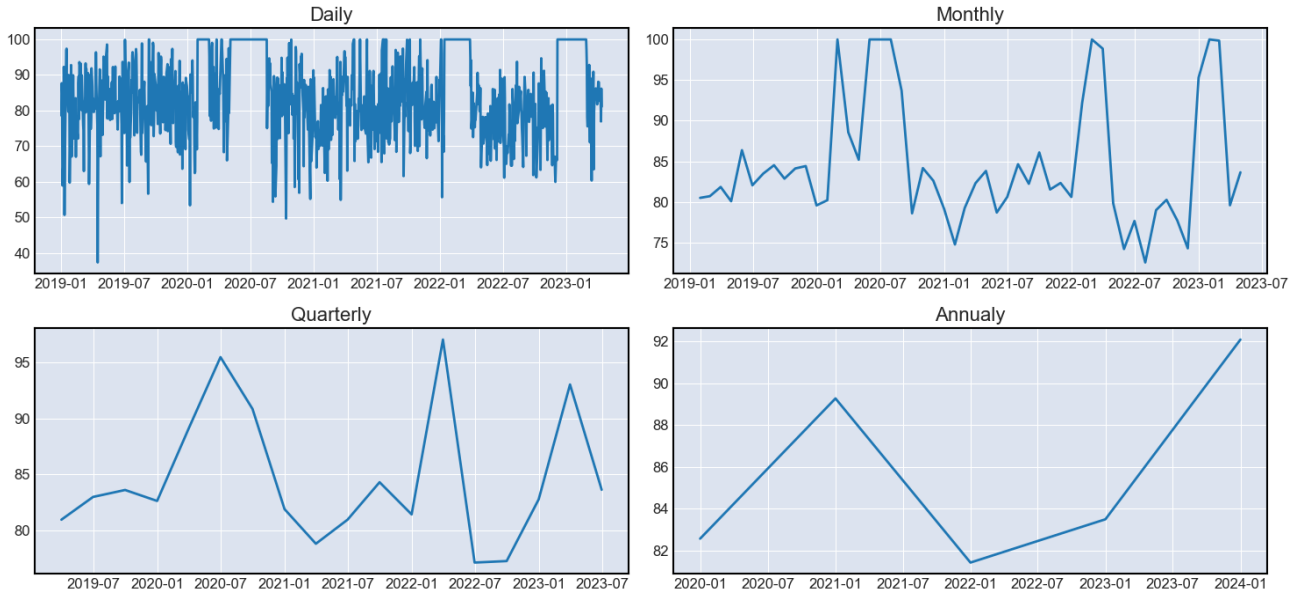


Nhận Xét:

- Ta có thể thấy được sự tăng dần của Deliverable Quantity chạm đỉnh vào năm 2022
- Sau đó thì Deliverable Quantity bắt đầu sụt giảm trở lại
- Các kỹ thuật được áp dụng: Facet (Thể hiện biểu đồ qua từng ngày, từng tháng, từng quý, từng năm). Khi ta áp dụng kỹ thuật này sẽ cho ta dễ thấy được sự biến động cụ thể Deliverable Quantity của vàng qua từng khung thời gian mà ta thể hiện.
- Từ việc trực quan hóa, biểu đồ trên giúp ta thấy được Deliverable Quantity đang sụt giảm khá nhiều ở thời điểm hiện tại

10. % Deli. Qty to Traded Qty

In [44]:
```
df.set_index('Date', inplace=True)

fig,axes = plt.subplots(2,2,figsize=[15,7])

## resampling to daily freq (original data)
axes[0,0].plot(data['% Deli. Qty to Traded Qty'])
axes[0,0].set_title("Daily",size=16)

## resampling to monthly freq
axes[0,1].plot(data['% Deli. Qty to Traded Qty'].resample('M').mean())
axes[0,1].set_title("Monthly",size=16)

## resmapling to quarterly freq
axes[1,0].plot(data['% Deli. Qty to Traded Qty'].resample('Q').mean())
axes[1,0].set_title('Quarterly',size=16)

## resampling to annualy freq
axes[1,1].plot(data['% Deli. Qty to Traded Qty'].resample('A').mean())
axes[1,1].set_title('Annualy',size=16)

plt.tight_layout()
plt.show()

df.reset_index(inplace=True)
```



Nhận Xét:

- Ta có thể thấy được sự tăng dần của % Deli. Qty to Traded Qty trước 2019 tới năm 2021
- Sau đó thì % Deli. Qty to Traded Qty bắt đầu sụt giảm và chạm đáy vào 2022
- Từ đó thì tăng dần trở lại và đang ở mức cao
- Các kỹ thuật được áp dụng: Facet (Thể hiện biểu đồ qua từng ngày, từng tháng, từng quý, từng năm). Khi ta áp dụng kỹ thuật này sẽ cho ta dễ thấy được sự biến động cụ thể Deliverable Quantity của vàng qua từng khung thời gian mà ta thể hiện.
- Từ việc trực quan hóa, biểu đồ trên giúp ta thấy được % Deli. Qty to Traded Qty đang có giá trị rất lớn ở thời điểm hiện tại