Import Libraries

In [2]:
```
#import necessary libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

Load of Data

In [4]:
```
#Load dataset
df = pd.read_csv("nigeria_food_prices (1).csv")
```

To View all Information

In [6]:
```
df
```

Out[6]:

	date	cmname	unit	category	price	currency	country
0	#date	#item+name	#item+unit	#item+type	#value	#currency	#country+name
1	15/01/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
2	15/02/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
3	15/03/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
4	15/04/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
...	...	...	...	...	...	...	..
13273	15/02/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	9880	NGN	Nigeria
13274	15/03/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	11310	NGN	Nigeria
13275	15/04/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	12110	NGN	Nigeria
13276	15/05/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	11148	NGN	Nigeria
13277	15/01/2018	Gari (white) - Retail	KG	cereals and tubers	165.913	NGN	Nigeria

13278 rows × 17 columns

Data Cleaning and Preprocessing

• Remove Duplicates and irrelevant columns

In [8]:
```
#Remove Duplicate rows
df = df.drop_duplicates()
```

In [9]:
```
df
```

Out[9]:

	date	cmname	unit	category	price	currency	country
0	#date	#item+name	#item+unit	#item+type	#value	#currency	#country+name
1	15/01/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
2	15/02/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
3	15/03/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
4	15/04/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
...	...	...	...	...	...	...	..
13273	15/02/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	9880	NGN	Nigeria
13274	15/03/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	11310	NGN	Nigeria
13275	15/04/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	12110	NGN	Nigeria
13276	15/05/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	11148	NGN	Nigeria
13277	15/01/2018	Gari (white) - Retail	KG	cereals and tubers	165.913	NGN	Nigeria

13278 rows × 17 columns

To remove a specific column

In [10]:
```
df = df.drop('default',axis=1).
```

In [11]:
```
df
```

Out[11]:

	date	cmname	unit	category	price	currency	country
0	#date	#item+name	#item+unit	#item+type	#value	#currency	#country+name
1	15/01/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
2	15/02/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
3	15/03/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
4	15/04/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
...	...	...	...	...	...	...	..
13273	15/02/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	9880	NGN	Nigeria
13274	15/03/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	11310	NGN	Nigeria
13275	15/04/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	12110	NGN	Nigeria
13276	15/05/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	11148	NGN	Nigeria
13277	15/01/2018	Gari (white) - Retail	KG	cereals and tubers	165.913	NGN	Nigeria

13278 rows × 16 columns

To remove specific columns

In [12]:
```
df = df.drop('adm1id',axis=1)
df = df.drop('mktid',axis=1)
df = df.drop('cmid',axis=1)
df = df.drop('ptid',axis=1)
df = df.drop('umid',axis=1)
df = df.drop('catid',axis=1)
```

In [13]:
```
df
```

Out[13]:

	date	cmname	unit	category	price	currency	country
0	#date	#item+name	#item+unit	#item+type	#value	#currency	#country+name
1	15/01/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
2	15/02/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
3	15/03/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
4	15/04/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria
...	...	...	...	...	...	...	..
13273	15/02/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	9880	NGN	Nigeria
13274	15/03/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	11310	NGN	Nigeria
13275	15/04/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	12110	NGN	Nigeria
13276	15/05/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	11148	NGN	Nigeria
13277	15/01/2018	Gari (white) - Retail	KG	cereals and tubers	165.913	NGN	Nigeria

13278 rows × 10 columns

to remove a row

In [14]:
```
df = df.drop(0,axis=0)
```

In [15]:
```
df
```

Out[15]:

	date	cmname	unit	category	price	currency	country	admname	mkt
1	15/01/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria	Adamawa	
2	15/02/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria	Adamawa	
3	15/03/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria	Adamawa	
4	15/04/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria	Adamawa	
5	15/05/2015	Bread - Retail	Unit	cereals and tubers	140	NGN	Nigeria	Adamawa	
...	...	...	...	...	...	...	...	...	
13273	15/02/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	9880	NGN	Nigeria	Zamfara	Nar
13274	15/03/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	11310	NGN	Nigeria	Zamfara	Nar
13275	15/04/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	12110	NGN	Nigeria	Zamfara	Nar
13276	15/05/2018	Gari (white) - Wholesale	100 KG	cereals and tubers	11148	NGN	Nigeria	Zamfara	Nar
13277	15/01/2018	Gari (white) - Retail	KG	cereals and tubers	165.913	NGN	Nigeria	Zamfara	Nar

13277 rows × 10 columns

Handle missing values

In [17]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13277 entries, 1 to 13277
Data columns (total 10 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   date      13277 non-null  object
 1   cmname    13277 non-null  object
 2   unit      13277 non-null  object
 3   category  13277 non-null  object
 4   price     13277 non-null  object
 5   currency  13277 non-null  object
 6   country   13277 non-null  object
 7   admname   13277 non-null  object
 8   mktname   13277 non-null  object
 9   sn        13277 non-null  object
dtypes: object(10)
memory usage: 1.0+ MB
```

to calculate the number of unique values in each column

In [18]:
```
df.nunique()
```

Out[18]:
```
date         166
cmname        37
unit           5
category       4
price       6258
currency       1
country        1
admname       14
mktname       21
sn           483
dtype: int64
```

To change the datatype

In [74]:
```
df['date'] = pd.to_datetime(df['date'])

df['price'] = df['price'].astype(float)
```

to view the datatype

In [76]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13277 entries, 1 to 13277
Data columns (total 11 columns):
 #   Column          Non-Null Count  Dtype         
---  ------          --------------  -----         
 0   date            13277 non-null  datetime64[ns]
 1   unit            13277 non-null  object        
 2   category        13277 non-null  object        
 3   price           13277 non-null  float64       
 4   currency        13277 non-null  object        
 5   country         13277 non-null  object        
 6   admin_name      13277 non-null  object        
 7   market_name     13277 non-null  object        
 8   serial_number   13277 non-null  object        
 9   commodity_name  13277 non-null  object        
 10  order_type      13277 non-null  object        
dtypes: datetime64[ns](1), float64(1), object(9)
memory usage: 1.1+ MB
```

is used to split a string in the cmname column and create two new columns: commodity_name and order_type

In [21]:
```
df[['commodity_name', 'order_type']] = df['cmname'].str.split(" - ", n=1, expand=Tr
df
```

Out[21]:

	date	cmname	unit	category	price	currency	country	admname	mktnam
1	2015-01-15	Bread - Retail	Unit	cereals and tubers	140.000	NGN	Nigeria	Adamawa	Mu
2	2015-02-15	Bread - Retail	Unit	cereals and tubers	140.000	NGN	Nigeria	Adamawa	Mu
3	2015-03-15	Bread - Retail	Unit	cereals and tubers	140.000	NGN	Nigeria	Adamawa	Mu
4	2015-04-15	Bread - Retail	Unit	cereals and tubers	140.000	NGN	Nigeria	Adamawa	Mu
5	2015-05-15	Bread - Retail	Unit	cereals and tubers	140.000	NGN	Nigeria	Adamawa	Mu
...	...	...	...	...	...	...	...	...	
13273	2018-02-15	Gari (white) - Wholesale	100 KG	cereals and tubers	9880.000	NGN	Nigeria	Zamfara	Kau Namoo
13274	2018-03-15	Gari (white) - Wholesale	100 KG	cereals and tubers	11310.000	NGN	Nigeria	Zamfara	Kau Namoo
13275	2018-04-15	Gari (white) - Wholesale	100 KG	cereals and tubers	12110.000	NGN	Nigeria	Zamfara	Kau Namoo
13276	2018-05-15	Gari (white) - Wholesale	100 KG	cereals and tubers	11148.000	NGN	Nigeria	Zamfara	Kau Namoo
13277	2018-01-15	Gari (white) - Retail	KG	cereals and tubers	165.913	NGN	Nigeria	Zamfara	Kau Namoo

13277 rows × 12 columns

is used to create a new DataFrame containing the unique values from the cmname column in the original df DataFrame

In [22]:
```
pd.DataFrame(df['cmname'].unique(), columns=['unique_cmname'])
```

Out[22]:

	unique_cmname
0	Bread - Retail
1	Oil (palm) - Wholesale
2	Oil (palm) - Retail
3	Rice (imported) - Wholesale
4	Maize (white) - Wholesale
5	Rice (local) - Wholesale
6	Rice (local) - Retail
7	Millet - Wholesale
8	Yam - Wholesale
9	Yam - Retail
10	Groundnuts (shelled) - Wholesale
11	Sorghum (white) - Wholesale
12	Maize (yellow) - Wholesale
13	Maize (yellow) - Retail
14	Fuel (diesel) - Retail
15	Fuel (petrol-gasoline) - Retail
16	Cowpeas (white) - Wholesale
17	Cowpeas (brown) - Wholesale
18	Sorghum (brown) - Wholesale
19	Sorghum (brown) - Retail
20	Gari (yellow) - Wholesale
21	Gari (white) - Wholesale
22	Maize - Wholesale
23	Sorghum - Wholesale
24	Wheat - Wholesale
25	Rice (imported) - Retail
26	Maize (white) - Retail
27	Millet - Retail
28	Groundnuts (shelled) - Retail
29	Sorghum (white) - Retail

	unique_cmname
30	Cowpeas (white) - Retail
31	Cowpeas (brown) - Retail
32	Gari (yellow) - Retail
33	Gari (white) - Retail
34	Beans (niebe) - Wholesale
35	Yam (Abuja) - Wholesale
36	Yam (Abuja) - Retail

to remove a column

In [23]:
```
df=df.drop("cmname",axis=1)
```

In [24]:
```
df
```

Out[24]:

	date	unit	category	price	currency	country	admname	mktname	
1	2015-01-15	Unit	cereals and tubers	140.000	NGN	Nigeria	Adamawa	Mubi	1980_55
2	2015-02-15	Unit	cereals and tubers	140.000	NGN	Nigeria	Adamawa	Mubi	1980_55
3	2015-03-15	Unit	cereals and tubers	140.000	NGN	Nigeria	Adamawa	Mubi	1980_55
4	2015-04-15	Unit	cereals and tubers	140.000	NGN	Nigeria	Adamawa	Mubi	1980_55
5	2015-05-15	Unit	cereals and tubers	140.000	NGN	Nigeria	Adamawa	Mubi	1980_55
...	...	...	...	...	...	...	...	...	
13273	2018-02-15	100 KG	cereals and tubers	9880.000	NGN	Nigeria	Zamfara	Kaura Namoda	1977_48
13274	2018-03-15	100 KG	cereals and tubers	11310.000	NGN	Nigeria	Zamfara	Kaura Namoda	1977_48
13275	2018-04-15	100 KG	cereals and tubers	12110.000	NGN	Nigeria	Zamfara	Kaura Namoda	1977_48
13276	2018-05-15	100 KG	cereals and tubers	11148.000	NGN	Nigeria	Zamfara	Kaura Namoda	1977_48
13277	2018-01-15	KG	cereals and tubers	165.913	NGN	Nigeria	Zamfara	Kaura Namoda	1977_48

13277 rows × 11 columns

to rename columns

In [25]:
```
df = df.rename(columns={'admname': 'admin_name', 'mktname': 'market_name','sn': 'se
```

to view the first 20

In [26]:
```
df.head(20)
```

Out[26]:

	date	unit	category	price	currency	country	admin_name	market_name	serial_nu
1	2015-01-15	Unit	cereals and tubers	140.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
2	2015-02-15	Unit	cereals and tubers	140.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
3	2015-03-15	Unit	cereals and tubers	140.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
4	2015-04-15	Unit	cereals and tubers	140.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
5	2015-05-15	Unit	cereals and tubers	140.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
6	2015-08-15	Unit	cereals and tubers	140.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
7	2015-09-15	Unit	cereals and tubers	140.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
8	2015-10-15	Unit	cereals and tubers	140.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
9	2015-11-15	Unit	cereals and tubers	140.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
10	2015-12-15	Unit	cereals and tubers	140.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
11	2016-01-15	Unit	cereals and tubers	140.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
12	2016-02-15	Unit	cereals and tubers	220.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
13	2016-03-15	Unit	cereals and tubers	220.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
14	2016-04-15	Unit	cereals and tubers	220.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1

	date	unit	category	price	currency	country	admin_name	market_name	serial_nu
15	2016-05-15	Unit	cereals and tubers	220.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
16	2016-06-15	Unit	cereals and tubers	300.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
17	2016-07-15	Unit	cereals and tubers	300.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
18	2016-08-15	Unit	cereals and tubers	300.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
19	2016-09-15	Unit	cereals and tubers	300.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1
20	2016-10-15	Unit	cereals and tubers	300.0	NGN	Nigeria	Adamawa	Mubi	1980_55_1

is used to sort the data by two columns (commodity_name and price), and then it retrieves the top 10 rows after sorting

In [27]:
```
df.sort_values(by=['commodity_name','price'],ascending=False).head(10)
```

Out[27]:

	date	unit	category	price	currency	country	admin_name	market_name	seria
9746	2016-07-15	100 KG	cereals and tubers	21250.0	NGN	Nigeria	Oyo	Ibadan	1975
9745	2016-06-15	100 KG	cereals and tubers	18500.0	NGN	Nigeria	Oyo	Ibadan	1975
9752	2017-03-15	100 KG	cereals and tubers	16000.0	NGN	Nigeria	Oyo	Ibadan	1975
9747	2016-08-15	100 KG	cereals and tubers	15000.0	NGN	Nigeria	Oyo	Ibadan	1975
9751	2017-02-15	100 KG	cereals and tubers	13000.0	NGN	Nigeria	Oyo	Ibadan	1975
9748	2016-11-15	100 KG	cereals and tubers	11000.0	NGN	Nigeria	Oyo	Ibadan	1975
9744	2016-05-15	100 KG	cereals and tubers	10750.0	NGN	Nigeria	Oyo	Ibadan	1975
9749	2016-12-15	100 KG	cereals and tubers	10500.0	NGN	Nigeria	Oyo	Ibadan	1975
9750	2017-01-15	100 KG	cereals and tubers	9500.0	NGN	Nigeria	Oyo	Ibadan	1975
9743	2016-04-15	100 KG	cereals and tubers	9375.0	NGN	Nigeria	Oyo	Ibadan	1975

is used to sort the data by two columns (commodity_name and price) in ascending order and then retrieve the first 10 rows of the sorted Data

In [28]:
```
df.sort_values(by=['commodity_name','price']).head(10)
```

Out[28]:

	date	unit	category	price	currency	country	admin_name	market_name	se
12478	2002-12-15	KG	pulses and nuts	112.3621	NGN	Nigeria	Sokoto	Illela (CBM)	10
12477	2002-11-15	KG	pulses and nuts	127.3570	NGN	Nigeria	Sokoto	Illela (CBM)	10
6660	2002-11-15	KG	pulses and nuts	130.0000	NGN	Nigeria	Katsina	Mai Adoua (CBM)	10
6661	2002-12-15	KG	pulses and nuts	135.0945	NGN	Nigeria	Katsina	Mai Adoua (CBM)	10
6093	2002-11-15	KG	pulses and nuts	150.0000	NGN	Nigeria	Katsina	Jibia (CBM)	10
6092	2002-10-15	KG	pulses and nuts	157.5000	NGN	Nigeria	Katsina	Jibia (CBM)	10
6094	2002-12-15	KG	pulses and nuts	169.6491	NGN	Nigeria	Katsina	Jibia (CBM)	10
6091	2002-09-15	KG	pulses and nuts	171.5594	NGN	Nigeria	Katsina	Jibia (CBM)	10
12476	2002-10-15	KG	pulses and nuts	181.6382	NGN	Nigeria	Sokoto	Illela (CBM)	10
6083	2002-01-15	KG	pulses and nuts	196.8692	NGN	Nigeria	Katsina	Jibia (CBM)	10

is used to group the data by the commodity_name column and then find the maximum price within each commodity group

In [29]:
```
df.groupby('commodity_name', as_index=False)['price'].max()
```

Out[29]:

	commodity_name	price
0	Beans (niebe)	419.1441
1	Bread	609.7600
2	Cowpeas (brown)	41500.0000
3	Cowpeas (white)	38960.0000
4	Fuel (diesel)	700.0000
5	Fuel (petrol-gasoline)	371.0000
6	Gari (white)	32500.0000
7	Gari (yellow)	48600.0000
8	Groundnuts (shelled)	52000.0000
9	Maize	276.6509
10	Maize (white)	18000.0000
11	Maize (yellow)	19080.0000
12	Millet	30000.0000
13	Oil (palm)	19375.0000
14	Rice (imported)	28140.0000
15	Rice (local)	53600.0000
16	Sorghum	273.8752
17	Sorghum (brown)	30600.0000
18	Sorghum (white)	29200.0000
19	Wheat	307.6923
20	Yam	90000.0000
21	Yam (Abuja)	21250.0000

is used to group the Data by the commodity_name column, and then aggregate values in other columns with specific functions

In [30]:
```
df.groupby('commodity_name', as_index=False).agg({
    'price': 'max',             # Maximum value for each category
    'date': 'first',
    'category': 'first',
     'admin_name': 'first'# Select the first 'other_column' value for each category
})
```

Out[30]:

	commodity_name	price	date	category	admin_name
0	Beans (niebe)	419.1441	2002-01-15	pulses and nuts	Katsina
1	Bread	609.7600	2015-01-15	cereals and tubers	Adamawa
2	Cowpeas (brown)	41500.0000	2015-01-15	pulses and nuts	Adamawa
3	Cowpeas (white)	38960.0000	2015-01-15	pulses and nuts	Adamawa
4	Fuel (diesel)	700.0000	2015-01-15	non-food	Adamawa
5	Fuel (petrol-gasoline)	371.0000	2015-01-15	non-food	Adamawa
6	Gari (white)	32500.0000	2015-01-15	cereals and tubers	Adamawa
7	Gari (yellow)	48600.0000	2015-08-15	cereals and tubers	Adamawa
8	Groundnuts (shelled)	52000.0000	2015-01-15	pulses and nuts	Adamawa
9	Maize	276.6509	2003-02-15	cereals and tubers	Borno
10	Maize (white)	18000.0000	2015-01-15	cereals and tubers	Adamawa
11	Maize (yellow)	19080.0000	2015-02-15	cereals and tubers	Adamawa
12	Millet	30000.0000	2015-01-15	cereals and tubers	Adamawa
13	Oil (palm)	19375.0000	2015-01-15	oil and fats	Adamawa
14	Rice (imported)	28140.0000	2015-01-15	cereals and tubers	Adamawa
15	Rice (local)	53600.0000	2015-01-15	cereals and tubers	Adamawa
16	Sorghum	273.8752	2003-02-15	cereals and tubers	Borno
17	Sorghum (brown)	30600.0000	2015-03-15	cereals and tubers	Adamawa
18	Sorghum (white)	29200.0000	2015-01-15	cereals and tubers	Adamawa
19	Wheat	307.6923	2003-02-15	cereals and tubers	Borno
20	Yam	90000.0000	2015-01-15	cereals and tubers	Adamawa
21	Yam (Abuja)	21250.0000	2015-12-15	cereals and tubers	Oyo

is used to filter the Data and select rows where the date column matches the exact value '2015-01-15'

In [31]:
```
df[df['date']=='2015-01-15']
```

Out[31]:

	date	unit	category	price	currency	country	admin_name	market_name	seri
1	2015-01-15	Unit	cereals and tubers	140.0	NGN	Nigeria	Adamawa	Mubi	198
35	2015-01-15	100 KG	oil and fats	7000.0	NGN	Nigeria	Adamawa	Mubi	19
69	2015-01-15	50 KG	cereals and tubers	9500.0	NGN	Nigeria	Adamawa	Mubi	198
103	2015-01-15	100 KG	cereals and tubers	4000.0	NGN	Nigeria	Adamawa	Mubi	19
137	2015-01-15	100 KG	cereals and tubers	8000.0	NGN	Nigeria	Adamawa	Mubi	19
...	...	...	...	...	...	...	...	...	
13049	2015-01-15	100 KG	pulses and nuts	8560.0	NGN	Nigeria	Zamfara	Kaura Namoda	197
13096	2015-01-15	100 KG	pulses and nuts	10571.0	NGN	Nigeria	Zamfara	Kaura Namoda	197
13143	2015-01-15	100 KG	cereals and tubers	4775.0	NGN	Nigeria	Zamfara	Kaura Namoda	197
13190	2015-01-15	100 KG	cereals and tubers	7325.0	NGN	Nigeria	Zamfara	Kaura Namoda	197
13237	2015-01-15	100 KG	cereals and tubers	6415.0	NGN	Nigeria	Zamfara	Kaura Namoda	197

257 rows × 11 columns

is used to filter the Data and select rows where the commodity_name column matches the exact word 'Bread'

In [32]:
```
df[df['commodity_name']=='Bread']
```

Out[32]:

	date	unit	category	price	currency	country	admin_name	market_name	se
1	2015-01-15	Unit	cereals and tubers	140.0000	NGN	Nigeria	Adamawa	Mubi	19
2	2015-02-15	Unit	cereals and tubers	140.0000	NGN	Nigeria	Adamawa	Mubi	19
3	2015-03-15	Unit	cereals and tubers	140.0000	NGN	Nigeria	Adamawa	Mubi	19
4	2015-04-15	Unit	cereals and tubers	140.0000	NGN	Nigeria	Adamawa	Mubi	19
5	2015-05-15	Unit	cereals and tubers	140.0000	NGN	Nigeria	Adamawa	Mubi	19
...	...	...	...	...	...	...	...	...	
12521	2018-01-15	Unit	cereals and tubers	292.0755	NGN	Nigeria	Zamfara	Kaura Namoda	19
12522	2018-02-15	Unit	cereals and tubers	296.4286	NGN	Nigeria	Zamfara	Kaura Namoda	19
12523	2018-03-15	Unit	cereals and tubers	296.4286	NGN	Nigeria	Zamfara	Kaura Namoda	19
12524	2018-04-15	Unit	cereals and tubers	300.0000	NGN	Nigeria	Zamfara	Kaura Namoda	19
12525	2018-05-15	Unit	cereals and tubers	346.4286	NGN	Nigeria	Zamfara	Kaura Namoda	19

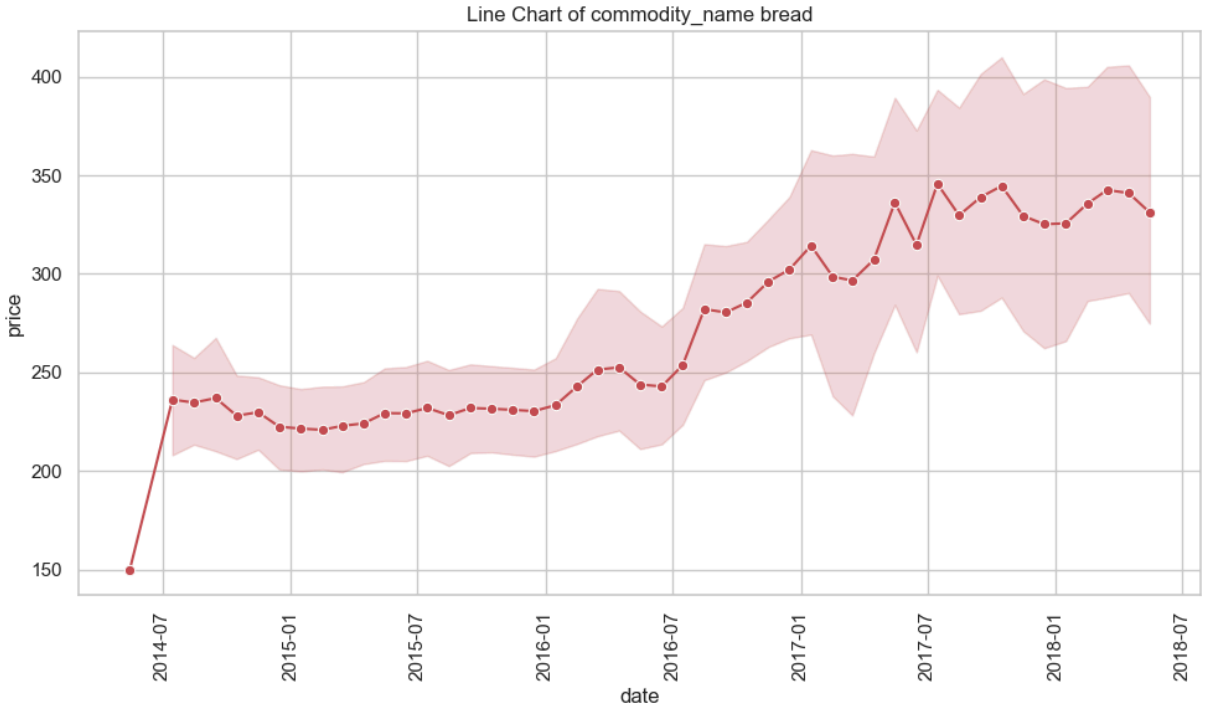
668 rows × 11 columns

Graphical representation of dataframe of 'commodity_name'== 'Bread'

In [117…

```
# Set the Seaborn style (optional)
sns.set(style="whitegrid")

# Increase figure size
plt.figure(figsize=(10, 6))
```

```
# Plotting the line chart using Seaborn
sns.lineplot(x='date', y='price', data=df[df['commodity_name']=='Bread'], marker='o

# Rotate the x-axis labels to make them readable
plt.xticks(rotation=90)

# Adding title and labels
plt.title('Line Chart of commodity_name bread')
plt.xlabel('date')
plt.ylabel('price')

# Automatically adjust layout
plt.tight_layout()

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

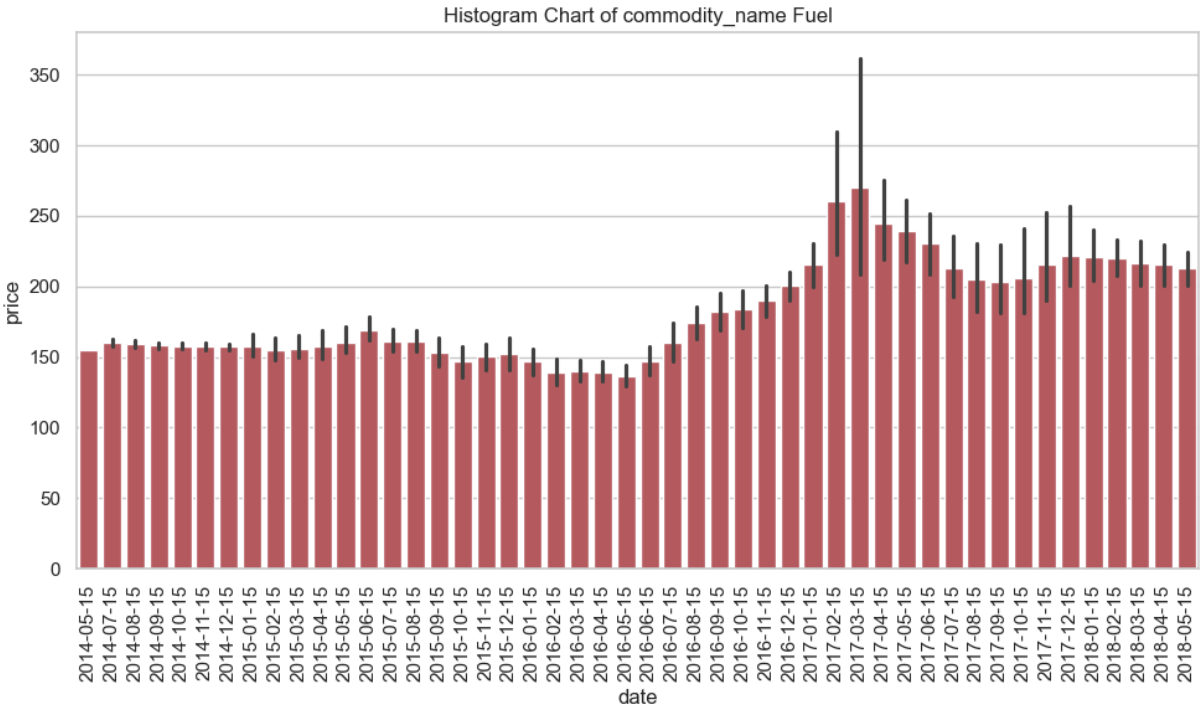


Graphical representation of dataframe of 'commodity_name'== 'Fuel(diesel)'

In [111…

```
# Set the Seaborn style (optional)
sns.set(style="whitegrid")

# Increase figure size
plt.figure(figsize=(10, 6))

# Plotting the line chart using Seaborn
sns.barplot(y='price', x='date',data = df[df['commodity_name'] == 'Fuel (diesel)'].

# Rotate the x-axis labels to make them readable
plt.xticks(rotation=90)

# Adding title and labels
```

```
plt.title('Histogram Chart of commodity_name Fuel')
plt.xlabel('date')
plt.ylabel('price')

# Automatically adjust layout
plt.tight_layout()

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

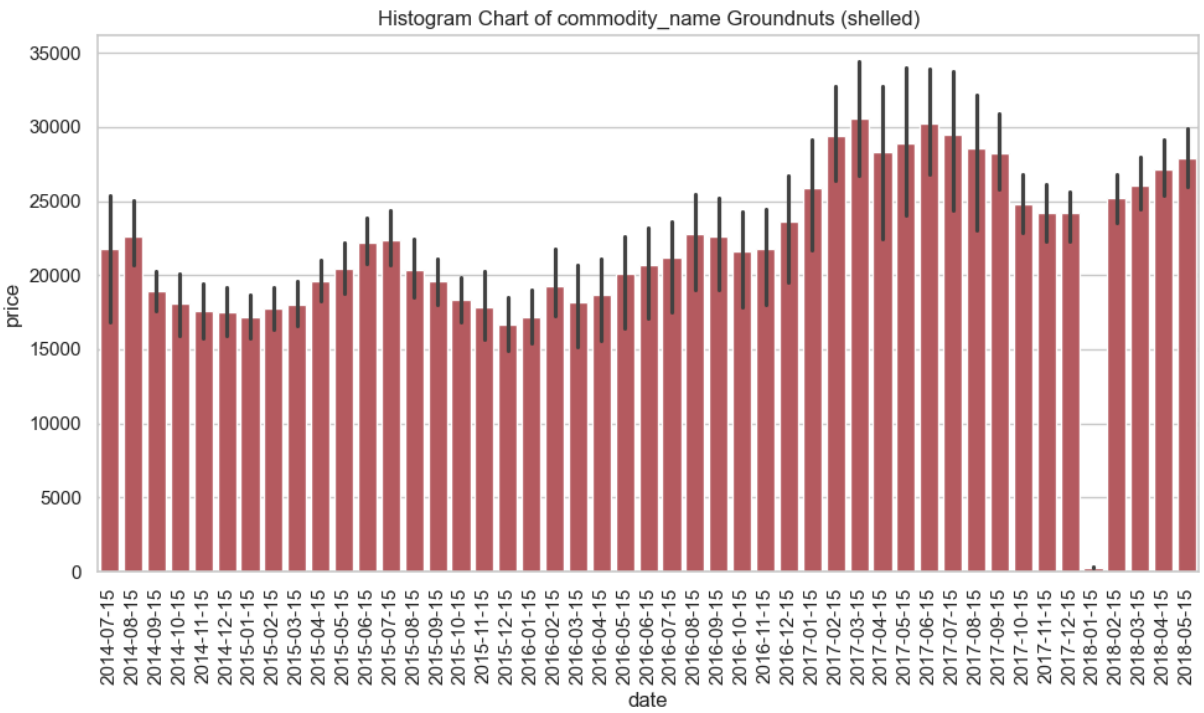


Graphical representation of dataframe of 'commodity_name'== 'Groundnuts (shelled)'

In [109…

```
# Set the Seaborn style (optional)
sns.set(style="whitegrid")

# Increase figure size
plt.figure(figsize=(10, 6))

# Plotting the line chart using Seaborn
sns.barplot(y='price', x='date',data = df[df['commodity_name'] == 'Groundnuts (shel

# Rotate the x-axis labels to make them readable
plt.xticks(rotation=90)

# Adding title and labels
plt.title('Histogram Chart of commodity_name Groundnuts (shelled)')
plt.xlabel('date')
plt.ylabel('price')

# Automatically adjust layout
plt.tight_layout()
```

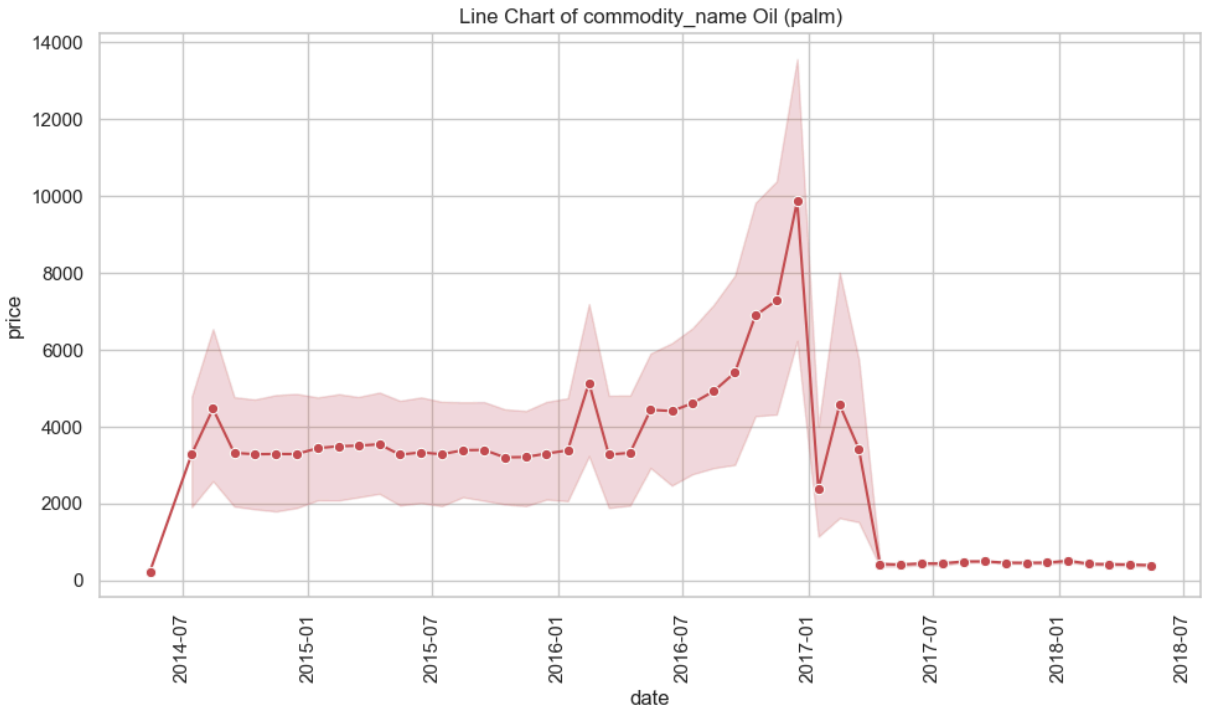
```
# Show the plot
plt.show()
```



Graphical representation of dataframe of 'commodity_name' == 'Oil (palm)'

In [113…

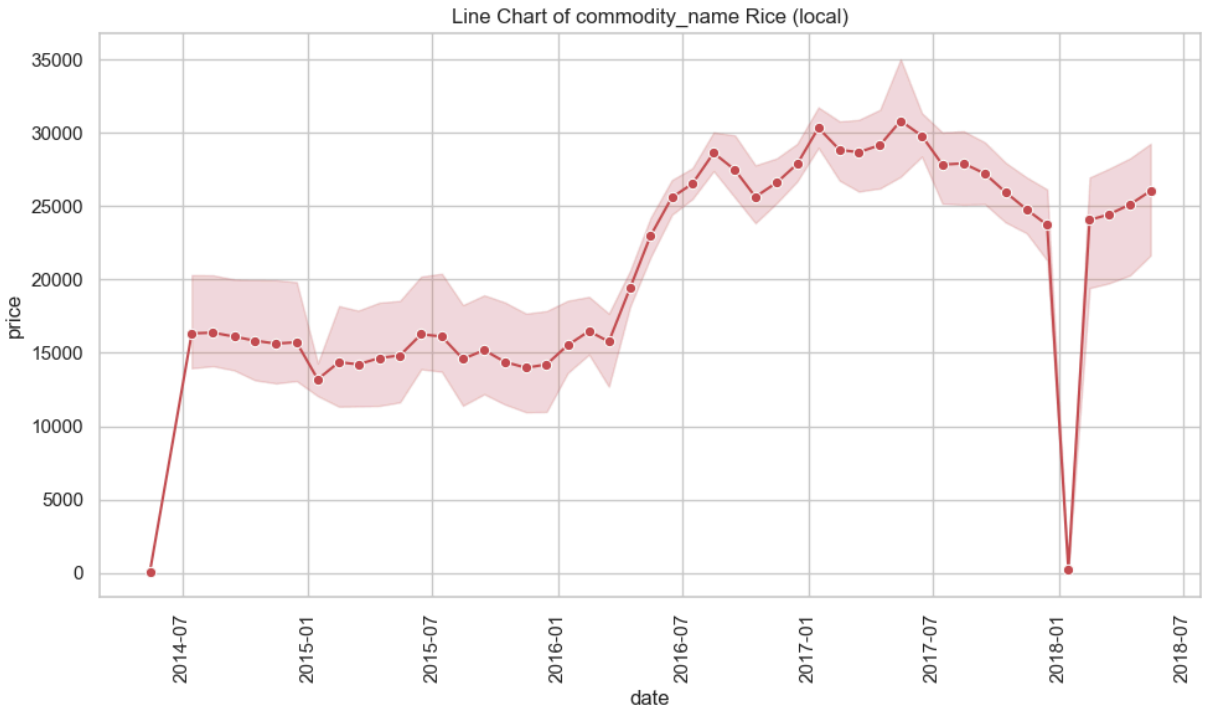
```
sns.set(style="whitegrid")

# Increase figure size
plt.figure(figsize=(10, 6))

# Plotting the line chart using Seaborn
sns.lineplot(x='date', y='price', data=df[df['commodity_name']=='Oil (palm)'], mark

# Rotate the x-axis labels to make them readable
plt.xticks(rotation=90)

# Adding title and labels
plt.title('Line Chart of commodity_name Oil (palm)')
plt.xlabel('date')
plt.ylabel('price')

# Automatically adjust layout
plt.tight_layout()

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



Graphical representation of dataframe of 'commodity_name' == 'Rice (local)'

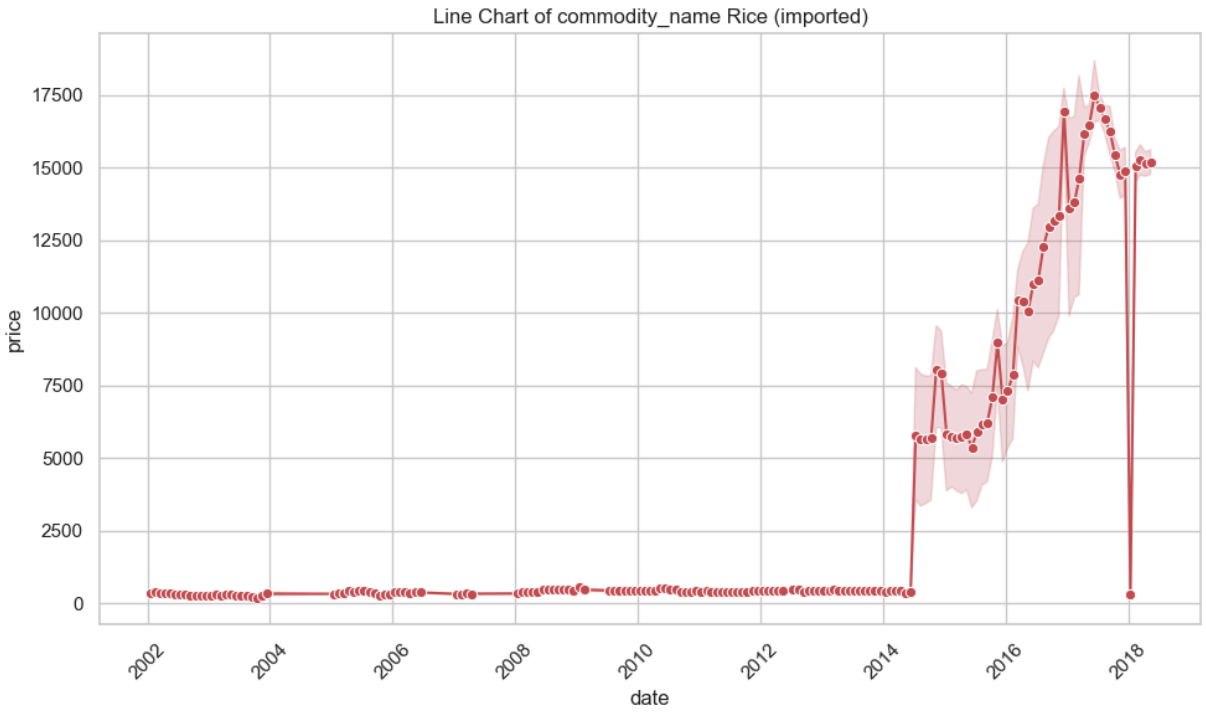
In [115…

```
sns.set(style="whitegrid")

# Increase figure size
plt.figure(figsize=(10, 6))

# Plotting the line chart using Seaborn
sns.lineplot(x='date', y='price', data=df[df['commodity_name']=='Rice (local)'], ma

# Rotate the x-axis labels to make them readable
plt.xticks(rotation=90)

# Adding title and labels
plt.title('Line Chart of commodity_name Rice (local)')
plt.xlabel('date')
plt.ylabel('price')

# Automatically adjust layout
plt.tight_layout()

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



Graphical representation of dataframe of 'commodity_name' == 'Rice (imported)

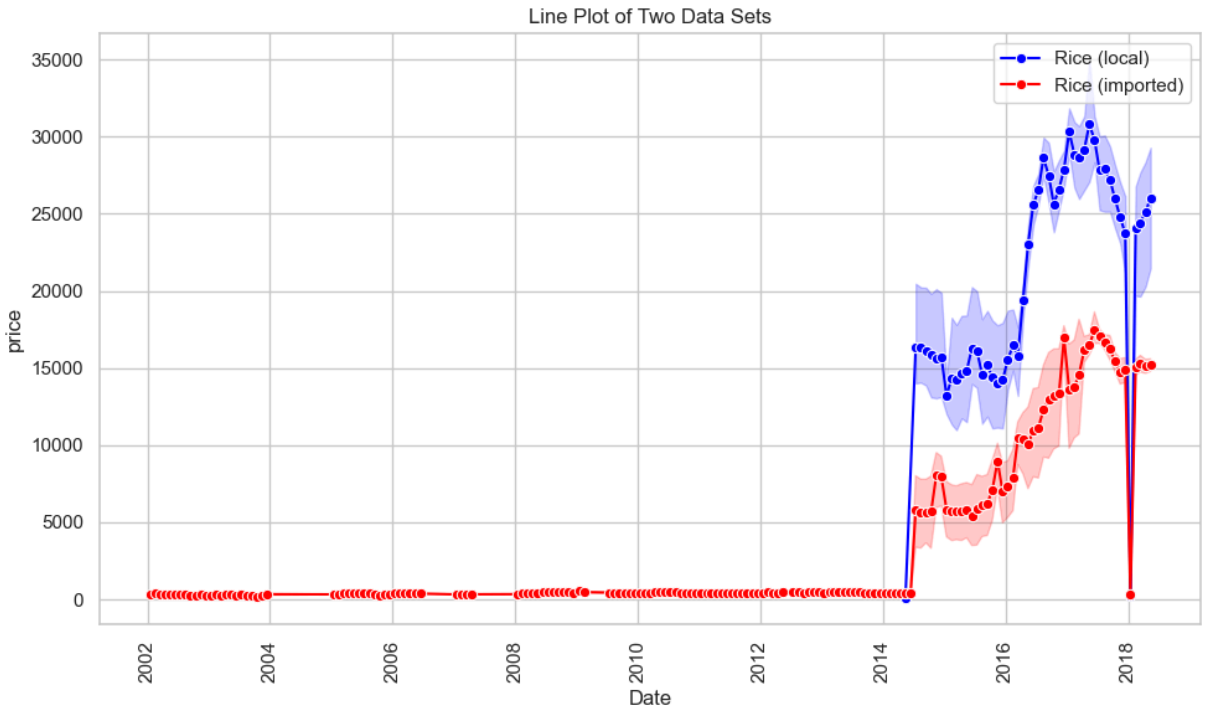
In [38]:
```
sns.set(style="whitegrid")

# Increase figure size
plt.figure(figsize=(10, 6))

# Plotting the line chart using Seaborn
sns.lineplot(x='date', y='price', data=df[df['commodity_name']=='Rice (imported)'],

# Rotate the x-axis labels to make them readable
plt.xticks(rotation=45)

# Adding title and labels
plt.title('Line Chart of commodity_name Rice (imported)')
plt.xlabel('date')
plt.ylabel('price')

# Automatically adjust layout
plt.tight_layout()

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



Graphical representation of dataframe of 'commodity_name' == 'Rice (imported) and commodity_name' == 'Rice(local)'

In [134…

```
# Create a line plot for the two datasets
plt.figure(figsize=(10, 6))

# Line plot for Data1
sns.lineplot(x='date', y='price' ,data=df[df['commodity_name']=='Rice (local)'], la

# Line plot for Data2
sns.lineplot(x='date', y='price' ,data=df[df['commodity_name']=='Rice (imported)'],

# Adding title and labels
plt.title('Line Plot of Two Data Sets')
plt.xlabel('Date')
plt.ylabel('price')

# Display legend
plt.legend()

# Show the plot
plt.xticks(rotation=90)  # Rotate x-axis labels for better readability
plt.tight_layout()  # Adjust layout to avoid overlap
plt.show()
```


Line Plot of Two Data Sets
Rice (local)
Rice (imported)
price
Date
35000
30000
25000
20000
15000
10000
5000
0
2002
2004
2006
2008
2010
2012
2014
2016
2018

In [ ]: