Découverte des données

In [1]:
```
# Importer les packages
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
```

In [2]:
```
# Afficher le nom de mes fichiers à utiliser
files = [file for file in os.listdir(r'C:\Users\BAKI\Desktop\DONI\Sales_Data')]
for file in files:
    print(file)
```

```
all_data.csv
Sales_April_2019.csv
Sales_August_2019.csv
Sales_December_2019.csv
Sales_February_2019.csv
Sales_January_2019.csv
Sales_July_2019.csv
Sales_June_2019.csv
Sales_March_2019.csv
Sales_May_2019.csv
Sales_November_2019.csv
Sales_October_2019.csv
Sales_September_2019.csv
```

In [3]:
```
path=r'C:\Users\BAKI\Desktop\DONI\Sales_Data'


# Créons une base de données vide
all_data = pd.DataFrame()

for file in files:
    current_data= pd.read_csv(path+'/'+ file)
    all_data= pd.concat([all_data, current_data])

print(all_data)
```

```
      Order ID                     Product Quantity Ordered Price Each  \
0       176558        USB-C Charging Cable                2      11.95
1          NaN                         NaN              NaN        NaN
2       176559  Bose SoundSport Headphones                1      99.99
3       176560                Google Phone                1        600
4       176560            Wired Headphones                1      11.99
...        ...                         ...              ...        ...
11681   259353      AAA Batteries (4-pack)                3       2.99
11682   259354                      iPhone                1        700
11683   259355                      iPhone                1        700
11684   259356      34in Ultrawide Monitor                1     379.99
11685   259357        USB-C Charging Cable                1      11.95

           Order Date                         Purchase Address
0      04/19/19 08:46            917 1st St, Dallas, TX 75001
1                 NaN                                     NaN
2      04/07/19 22:30        682 Chestnut St, Boston, MA 02215
3      04/12/19 14:38     669 Spruce St, Los Angeles, CA 90001
4      04/12/19 14:38     669 Spruce St, Los Angeles, CA 90001
...                ...                                     ...
11681  09/17/19 20:56   840 Highland St, Los Angeles, CA 90001
11682  09/01/19 16:00  216 Dogwood St, San Francisco, CA 94016
11683  09/23/19 07:39     220 12th St, San Francisco, CA 94016
11684  09/19/19 17:30   511 Forest St, San Francisco, CA 94016
11685  09/30/19 00:18   250 Meadow St, San Francisco, CA 94016

[560550 rows x 6 columns]
```

In [4]:
```
donnees_janvier=pd.read_csv(path+'/'+ 'Sales_January_2019.csv')
```

In [5]:
```
print(donnees_janvier)
```

```
     Order ID                   Product Quantity Ordered Price Each  \
0      141234                    iPhone                1        700
1      141235  Lightning Charging Cable                1      14.95
2      141236          Wired Headphones                2      11.99
3      141237          27in FHD Monitor                1     149.99
4      141238          Wired Headphones                1      11.99
...       ...                       ...              ...        ...
9718   150497              20in Monitor                1     109.99
9719   150498          27in FHD Monitor                1     149.99
9720   150499           ThinkPad Laptop                1     999.99
9721   150500    AAA Batteries (4-pack)                2       2.99
9722   150501              Google Phone                1        600

          Order Date                      Purchase Address
0     01/22/19 21:25        944 Walnut St, Boston, MA 02215
1     01/28/19 14:15       185 Maple St, Portland, OR 97035
2     01/17/19 13:33  538 Adams St, San Francisco, CA 94016
3     01/05/19 20:33     738 10th St, Los Angeles, CA 90001
4     01/25/19 11:59          387 10th St, Austin, TX 73301
...             ...                                    ...
9718  01/26/19 19:09            95 8th St, Dallas, TX 75001
9719  01/10/19 22:58    403 7th St, San Francisco, CA 94016
9720  01/21/19 14:31        214 Main St, Portland, OR 97035
9721  01/15/19 14:21      810 2nd St, Los Angeles, CA 90001
9722  01/13/19 16:43         428 Cedar St, Boston, MA 02215

[9723 rows x 6 columns]
```

In [6]:
```
all_data.to_csv(r'C:\Users\BAKI\Desktop\DONI\Sales_Data'+'/all_data.csv', index=False)
```

In [7]:
```
all_data.dtypes
```

Out[7]:
```
Order ID            object
Product             object
Quantity Ordered    object
Price Each          object
Order Date          object
Purchase Address    object
dtype: object
```

In [8]:
```
all_data.head()
```

Out[8]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001
1	NaN	NaN	NaN	NaN	NaN	NaN
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215
3	176560	Google Phone	1	600	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001

In [9]:
```
all_data.isnull().sum()
```

Out[9]:
```
Order ID            1635
Product             1635
Quantity Ordered    1635
Price Each          1635
Order Date          1635
Purchase Address    1635
dtype: int64
```

In [10]:
```
# Supprimons les valeurs manquantes
all_data=all_data.dropna(how='all')
```

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

Out[11]:
```
(558915, 6)
```

Quel est le mois durant lequel l'entreprise a réalisé le meilleur chiffre d'affaires?

In [12]:
```
# Créons une fonction de récupération du mois
def month_recup(x):
    return x.split('/')[0]
```

In [13]:
```
# Créons une nouvelle colonne 'Month' avec les mois récupérés
all_data['Month']=all_data['Order Date'].apply(month_recup)
all_data.head()
```

Out[13]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	04
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	04
3	176560	Google Phone	1	600	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	04
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	04
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	04

In [14]:
```
all_data['Month'].unique()
```

Out[14]:
```
array(['04', '05', 'Order Date', '08', '09', '12', '01', '02', '03', '07',
       '06', '11', '10'], dtype=object)
```

In [15]:
```
all_data= all_data[all_data['Month']!='Order Date']
```

In [16]:
```
all_data
```

Out[16]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	04
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	04
3	176560	Google Phone	1	600	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	04
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	04
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	04
...	...	...	...	...	...	...	...
11681	259353	AAA Batteries (4-pack)	3	2.99	09/17/19 20:56	840 Highland St, Los Angeles, CA 90001	09
11682	259354	iPhone	1	700	09/01/19 16:00	216 Dogwood St, San Francisco, CA 94016	09
11683	259355	iPhone	1	700	09/23/19 07:39	220 12th St, San Francisco, CA 94016	09
11684	259356	34in Ultrawide Monitor	1	379.99	09/19/19 17:30	511 Forest St, San Francisco, CA 94016	09
11685	259357	USB-C Charging Cable	1	11.95	09/30/19 00:18	250 Meadow St, San Francisco, CA 94016	09

557850 rows × 7 columns

In [17]:
```
all_data['Month'].unique()
```

Out[17]:
```
array(['04', '05', '08', '09', '12', '01', '02', '03', '07', '06', '11',
       '10'], dtype=object)
```

In [18]:
```
all_data.dtypes
```

Out[18]:
```
Order ID            object
Product             object
Quantity Ordered    object
Price Each          object
```

```
Order Date          object
Purchase Address    object
Month               object
dtype: object
```

Nous remarquons que toutes les variables de notre dataset sont actuellement de type object alors que certaines variables ne devraient pas etre de ce type.

In [19]:
```
# Changeons le type de données de la variable 'Month' qui est actuellement 'object' en variable de type 'int'
all_data['Month']=all_data['Month'].astype('int')
all_data['Month'].dtypes
```

Out[19]:
```
dtype('int32')
```

In [20]:
```
# Changeons le type de données de la variable 'Quantity Ordered' qui est actuellement 'object' en variable de typ
all_data['Quantity Ordered']=all_data['Quantity Ordered'].astype('int')
all_data['Quantity Ordered'].dtypes
```

Out[20]:
```
dtype('int32')
```

In [21]:
```
# Changeons le type de données de la variable 'Price Each' qui est actuellement 'object' en variable de type 'flo
all_data['Price Each']=all_data['Price Each'].astype('float')
all_data['Price Each'].dtypes
```

Out[21]:
```
dtype('float64')
```

In [22]:
```
# Créons une nouvelle colonne 'Sales_amount' qui sera le Quantity Ordered multiplié par le Price Each
all_data['Sales_amount'] =all_data['Quantity Ordered']* all_data['Price Each']
```

In [23]:
```
all_data.head()
```

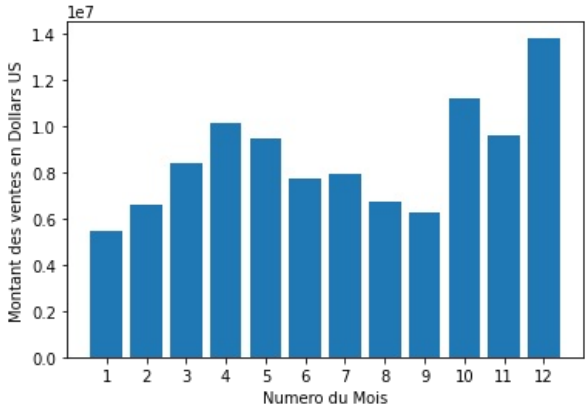
Out[23]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales_amount
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	4	11.99

In [24]:
```
# Les montants des ventes par mois

all_data.groupby('Month')['Sales_amount'].sum()
```

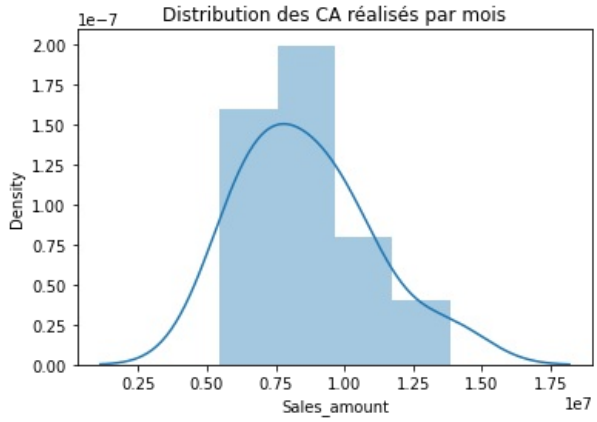
Out[24]:
```
Month
1     5.466770e+06
2     6.606067e+06
3     8.421301e+06
4     1.017201e+07
5     9.457820e+06
6     7.733407e+06
7     7.943327e+06
8     6.733404e+06
9     6.292680e+06
10    1.121018e+07
11    9.598810e+06
12    1.384033e+07
Name: Sales_amount, dtype: float64
```

In [25]:
```
months=range(1,13)
plt.bar(months,all_data.groupby('Month')['Sales_amount'].sum())
plt.xticks(months)
plt.ylabel('Montant des ventes en Dollars US')
plt.xlabel('Numero du Mois')
plt.show()
```



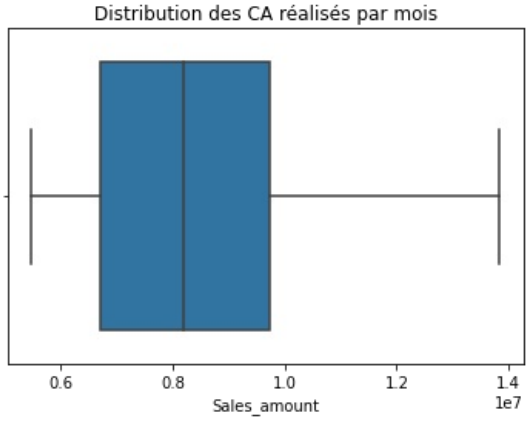
In [26]:
```
sns.distplot(
    all_data.groupby('Month')['Sales_amount'].sum(),
    kde = True).set_title("Distribution des CA réalisés par mois");
```

```
C:\Users\BAKI\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



In [27]:
```
sns.boxplot(all_data.groupby('Month')['Sales_amount'].sum()).set_title("Distribution des CA réalisés par mois");
```

```
C:\Users\BAKI\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



En observant notre **box-plot**, nous remarquons au niveau des CA réalisés par mois:

- **Une distribution positivement asymétrique**, c'est à dire la longueur du 1er et du 2ème quartile est plus petite que celle du 3ème et 4ème quartile.
- la tranche de **3 à 6 millions** représente **75%** des CA mensuels.
- la tranche de **7 à 9 millions** représente **25%** des CA mensuels.

In [28]:
```
# Etendue de la variable 'CA par mois'

all_data.groupby('Month')['Sales_amount'].sum().max() - all_data.groupby('Month')['Sales_amount'].sum().min()
```

Out[28]:
```
8373559.829999997
```

In [29]:
```
# Variance de la variable 'CA par mois'

all_data.groupby('Month')['Sales_amount'].sum().var()
```

Out[29]:
```
5662489213613.615
```

In [30]:
```
# Ecart-type de la variable 'CA par mois'

all_data.groupby('Month')['Sales_amount'].sum().std()
```

Out[30]:
```
2379598.5404293756
```

In [31]:
```
# Vérifions que l'écart-type est bel et bien la racine caréée de la variance

all_data.groupby('Month')['Sales_amount'].sum().std() == np.sqrt(all_data.groupby('Month')['Sales_amount'].sum().
```

Out[31]:
```
True
```

In [32]:
```
all_data.groupby('Month')['Sales_amount'].sum().describe()
```

Out[32]:
```
count    1.200000e+01
mean     8.623009e+06
std      2.379599e+06
min      5.466770e+06
25%      6.701570e+06
50%      8.182314e+06
75%      9.742110e+06
max      1.384033e+07
Name: Sales_amount, dtype: float64
```

In [33]:
```
# Quartiles de la variable 'CA par mois' : méthode .quantile()

print('1er quartile :',all_data.groupby('Month')['Sales_amount'].sum().quantile(0.25))

print('2è quartile :',all_data.groupby('Month')['Sales_amount'].sum().quantile(0.5))

print('3è quartile :',all_data.groupby('Month')['Sales_amount'].sum().quantile(0.75))
```

```
1er quartile : 6701569.5450028945
2è quartile : 8182314.210003907
3è quartile : 9742109.880003601
```

In [34]:
```
# Importation du module stats

from scipy import stats

# IQR de la variable 'CA par mois'

stats.iqr(all_data.groupby('Month')['Sales_amount'].sum())
```

Out[34]: 3040540.335000707

In [35]:
```
Sales_amount_per_month=all_data.groupby('Month')['Sales_amount'].sum()
```

In [36]:
```
fig = px.histogram(all_data, x="Month", y="Sales_amount")
fig.show()
```

En observant attentivement notre histogramme, nous pouvons voir l'entreprise a réalisé son meilleur CA dans le mois de **Décembre (4.613.443)**. Cela pourrait s'expliquer par le fait qu'en Décembre les gens achètent beaucoup de gadgets, appareils électroniques pendant les fetes de la Noel et de fin d'années.

Tandis que le mois de **Janvier** enregistre le CA le plus faible **(1.822.256)**.

In [37]:
```
# CA mensuel minimum
Sales_amount_per_month.min()
```

Out[37]: 5466770.1900020735

In [38]:
```
# CA mensuel maximum
Sales_amount_per_month.max()
```

Out[38]: 13840330.02000207

In [39]:
```
# CA mensuel moyen
Sales_amount_per_month.mean()
```

Out[39]: 8623008.992503123

In [40]:
```
# CA mensuel median
Sales_amount_per_month.median()
```

Out[40]: 8182314.210003907

Dans quelle ville l'entreprise a réalisé le maximum de commandes ?

In [41]:
```
# Créons une fonction de récupération du nom de la ville
def city(x):
    return x.split(',')[1]
```

In [42]:
```
# Créons une nouvelle colonne 'City' avec les noms de villes récupérés et affichons la dataframe
all_data['City'] = all_data['Purchase Address'].apply(city)
all_data
```

Out[42]:

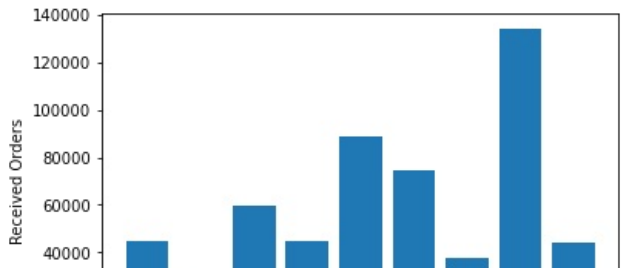
	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales_amount	City
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90	Dallas
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99	Boston
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles
...	...	...	...	...	...	...	...	...	...
11681	259353	AAA Batteries (4-pack)	3	2.99	09/17/19 20:56	840 Highland St, Los Angeles, CA 90001	9	8.97	Los Angeles
11682	259354	iPhone	1	700.00	09/01/19 16:00	216 Dogwood St, San Francisco, CA 94016	9	700.00	San Francisco
11683	259355	iPhone	1	700.00	09/23/19 07:39	220 12th St, San Francisco, CA 94016	9	700.00	San Francisco
11684	259356	34in Ultrawide Monitor	1	379.99	09/19/19 17:30	511 Forest St, San Francisco, CA 94016	9	379.99	San Francisco
11685	259357	USB-C Charging Cable	1	11.95	09/30/19 00:18	250 Meadow St, San Francisco, CA 94016	9	11.95	San Francisco

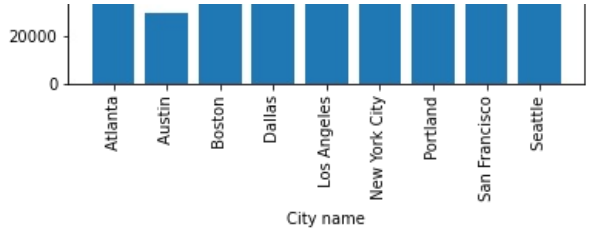
557850 rows × 9 columns

In [43]:
```
# Comptons le nombre de commandes en les groupant par ville sachant qu'une ligne ou observation de la dataframe e
all_data.groupby('City')['City'].count()
```

Out[43]:
```
City
 Atlanta          44643
 Austin           29715
 Boston           59802
 Dallas           44460
 Los Angeles      88815
 New York City    74628
 Portland         37395
 San Francisco   134196
 Seattle          44196
Name: City, dtype: int64
```

In [44]:
```
# Visualisons nos résultats à l'aide d'un graphique à barre
plt.bar(all_data.groupby('City')['City'].count().index,all_data.groupby('City')['City'].count())
plt.xticks(rotation='vertical')
plt.ylabel('Received Orders')
plt.xlabel('City name')
plt.show()
```





En observant attentivement notre graphique à barre, nous pouvons voir que l'entreprise a réalisé le plus de commandes **(89464)** dans la ville de **San Francisco**. Tandis que la ville d'**Austin** enregistre le moins de commandes **(19810)**.

On pourrait se poser plusieurs questions comme:

- Qu'est ce qui fait que la ville d'Austin enregistre un faible nombre de commande par rapport aux autres villes?
- Est-ce du fait d'un mauvais emplacement de la boutique? Est-ce du fait du Commercial?

En quel moment doit on faire une campagne publicitaire pour avoir plus de ventes ?

In [45]:
```
# Créons une nouvelle colonne 'Hour' dans laquelle nous allons insérer les heures de ventes extraites de la color
all_data['Hour']= pd.to_datetime(all_data['Order Date']).dt.hour
```

In [46]:
```
# Affichons notre dataframe pour vérifier notre colonne 'Hour'
all_data.head()
```

Out[46]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	Month	Sales_amount	City	Hour
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	4	23.90	Dallas	8
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	4	99.99	Boston	22
3	176560	Google Phone	1	600.00	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	600.00	Los Angeles	14
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	4	11.99	Los Angeles	14
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001	4	11.99	Los Angeles	9

Nous allons créer une fonction dans laquelle nous aurons une liste comprenant toutes les heures de commande et une autre liste comprenant le nombre de commandes réalisé pour chacune des heures listées. Nous allons d'abord grouper les lignes de commande par heure et par la suite compter le nombre commandes affecté à chaque heure.

Méthode 1 : A l'aide d'une fonction

In [47]:
```
keys=[]
hours=[]
for key, hour in all_data.groupby('Hour'):
    keys.append(key)
    hours.append(len(hour))

hours
```

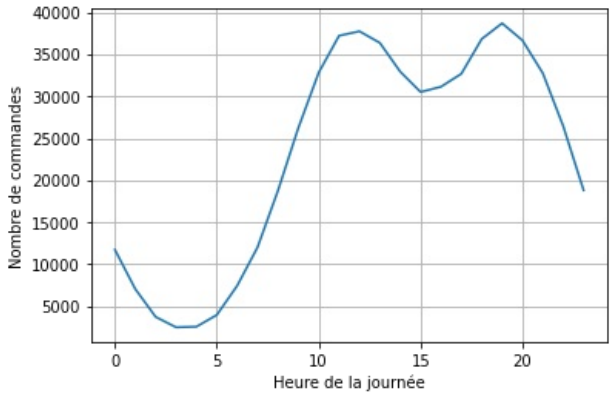
Out[47]:
```
[11730,
 7050,
 3729,
 2493,
 2562,
 3963,
 7446,
 12033,
 18768,
 26244,
 32832,
 37233,
 37761,
 36387,
 32952,
```

```
 30525,
 31152,
 32697,
 36840,
 38715,
 36684,
 32763,
 26466,
 18825]
```

In [48]:
```
keys
```

Out[48]:
```
[0,
 1,
 2,
 3,
 4,
 5,
 6,
 7,
 8,
 9,
 10,
 11,
 12,
 13,
 14,
 15,
 16,
 17,
 18,
 19,
 20,
 21,
 22,
 23]
```

In [49]:
```
# Visualisons nos quantités de commandes groupées par heures
plt.grid()
plt.plot(keys,hours)
plt.xlabel('Heure de la journée')
plt.ylabel('Nombre de commandes')

plt.show()
```



En observant notre graphique en courbe, nous pouvons constater que l'entreprise réalise plus de ventes entre **12h** et **19h**. Cela peut s'expliquer par le fait que ces heures soient des **créneaux** qui permettent aux clients de laisser leurs **occupations quotidiennes (boulot, famille, etc...)** pour faire leurs courses dans nos boutiques. Pour repondre à notre question de savoir à quelle heure serait profitable pour l'entreprise de méner une campagne publicitaire? Nous allons repondre entre **12h** et **19h** comme étant le meilleur créneau pour lancer une campagne publicitaire.

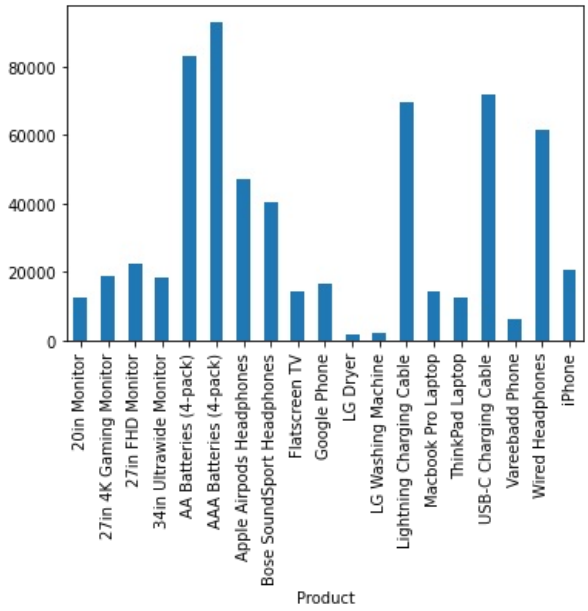
Quel produit se vend le plus ?

In [57]:
```
# Déterminons les produits qui ont le plus de commandes
all_data.groupby('Product')['Quantity Ordered'].sum()
```

Out[57]:
```
Product
20in Monitor                  12387
27in 4K Gaming Monitor        18732
27in FHD Monitor              22650
34in Ultrawide Monitor        18597
AA Batteries (4-pack)         82905
AAA Batteries (4-pack)        93051
Apple Airpods Headphones      46983
Bose SoundSport Headphones    40371
Flatscreen TV                 14457
Google Phone                  16596
LG Dryer                       1938
LG Washing Machine             1998
Lightning Charging Cable      69651
Macbook Pro Laptop            14184
ThinkPad Laptop               12390
USB-C Charging Cable          71925
Vareebadd Phone                6204
Wired Headphones              61671
iPhone                        20547
Name: Quantity Ordered, dtype: int32
```

In [56]:
```
# Visualisons
all_data.groupby('Product')['Quantity Ordered'].sum().plot(kind='bar')
```

Out[56]:
```
<AxesSubplot:xlabel='Product'>
```



En observant notre graphique à barre, nous pouvons voir que les produits comme:

AAA Batteries (4-pack) (93051),

AA Batteries (4-pack) (82905),

USB-C Charging Cable (71925),

Lightning Charging Cable (69651) et

Wired Headphones (61671) génèrent respectivement plus de ventes.

Certes, ces produits se vendent le plus, mais est ce que cela est lié à leurs prix unitaires respectifs?

Essayerons de voir l'impact de leurs prix **(prix moyens)** sur leurs quantités commandées.

In [58]:
```
all_data.groupby('Product')['Price Each'].mean()
```

Out[58]:
```
Product
20in Monitor                  109.99
27in 4K Gaming Monitor        389.99
27in FHD Monitor              149.99
34in Ultrawide Monitor        379.99
```

```
AA Batteries (4-pack)            3.84
AAA Batteries (4-pack)           2.99
Apple Airpods Headphones       150.00
Bose SoundSport Headphones      99.99
Flatscreen TV                  300.00
Google Phone                   600.00
LG Dryer                       600.00
LG Washing Machine             600.00
Lightning Charging Cable        14.95
Macbook Pro Laptop            1700.00
ThinkPad Laptop                999.99
USB-C Charging Cable            11.95
Vareebadd Phone                400.00
Wired Headphones                11.99
iPhone                         700.00
Name: Price Each, dtype: float64
```

In [53]:

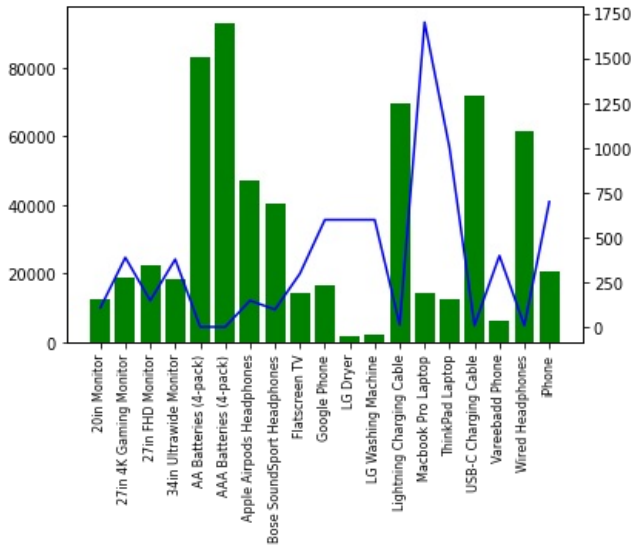
```
# Definissons les variables qui nous serviront à tracer notre graphique
products=all_data.groupby('Product')['Quantity Ordered'].sum().index
quantity=all_data.groupby('Product')['Quantity Ordered'].sum()
prices=all_data.groupby('Product')['Price Each'].mean()
```

In [55]:

```
# Visualisons l'impact des prix, des quantités vendues sur
plt.figure(figsize=(40,24))
fig,ax1=plt.subplots()
ax2=ax1.twinx()

ax1.bar(products,quantity,color='g')
ax2.plot(products,prices, 'b-')
ax1.set_xticklabels(products,rotation='vertical',size=8)

plt.show()
```

```
<ipython-input-55-5ee27999bf92>:7: UserWarning:

FixedFormatter should only be used together with FixedLocator
```

```
<Figure size 2880x1728 with 0 Axes>
```



Observons de plus près encore notre graphique à barre, et nous pouvons constater que les produits qui génèrent le plus de commandes sont les produits qui ont un prix moyen faible (prix moyen illustré en bleu sur le graphique).

A l'instar de :

AAA Batteries (4-pack) **(Nbre de commandes = 93051, prix moyen= 2.99)**,

AA Batteries (4-pack) **(Nbre de commandes = 82905, prix moyen= 3.84)**,

USB-C Charging Cable **(Nbre de commandes = 71925, prix moyen= 11.95)**,

Lightning Charging Cable **(Nbre de commandes = 69651, prix moyen= 14.95)** et

Wired Headphones **(Nbre de commandes = 61671, prix moyen= 11.99)**

Tandis que ceux qui ont un faible nombre de commandes ont un prix moyen élevé (prix moyen illustré en bleu sur le graphique).

A l'instar de :

Macbook Pro Laptop **(Nbre de commandes = 14184, prix moyen= 1700.00)**,

ThinkPad Laptop **(Nbre de commandes = 12390, prix moyen= 999.99)**,

LG Dryer **(Nbre de commandes = 1938, prix moyen= 600.00)**,

LG Washing Machine **(Nbre de commandes = 1998, prix moyen= 600.00)** et

iPhone **(Nbre de commandes = 20547, prix moyen= 700.00)**.

Fin de l'analyse !!!