Analyzing Sales Report

- KeithGalli's Sales Analysis Youtube
- KeithGalli's Jupyter notebook
- dataset

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
```

In [2]:
```
pd.set_option('mode.chained_assignment', None) # ignore warning when try to set values to existing data frame
```

Read data files & Clean up data

In [3]:
```
def fxdatafileReader(filesastrisk=None, filelist=[], ext='csv', skiprows=None):
    import pandas as pd
    from glob import glob

    if filesastrisk!=None:
        datafiles = sorted(glob(filesastrisk + '*.' + ext))
        print(datafiles)
    elif len(filelist)!=0:
        datafiles = [filename + '.' + ext for filename in filelist]
        print(datafiles)

    if ext=='csv':
        df = pd.concat((pd.read_csv(file).assign(filename=file)
             for file in datafiles), ignore_index=True)

    elif ext=='xlsx' or ext=='xls':
        df = pd.concat((pd.read_excel(file, skiprows=skiprows,
                  engine='openpyxl').assign(filename=file)
                  for file in datafiles), ignore_index=True)
    return df
```

In [4]:
```
data = fxdatafileReader(filesastrisk='Sales_', ext='csv')
```

```
['Sales_April_2019.csv', 'Sales_August_2019.csv', 'Sales_December_2019.csv', 'Sales_February_2019.cs
v', '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.cs
v']
```

In [5]:
```
data
```

Out[5]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	filename
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001	Sales_April_2019.csv
1	NaN	NaN	NaN	NaN	NaN	NaN	Sales_April_2019.csv
2	176559	Bose SoundSport Headphones	1	99.99	04/07/19 22:30	682 Chestnut St, Boston, MA 02215	Sales_April_2019.csv
3	176560	Google Phone	1	600	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	Sales_April_2019.csv
4	176560	Wired Headphones	1	11.99	04/12/19 14:38	669 Spruce St, Los Angeles, CA 90001	Sales_April_2019.csv
...	...	...	...	...	...	...	...
186845	259353	AAA Batteries (4-pack)	3	2.99	09/17/19 20:56	840 Highland St, Los Angeles, CA 90001	Sales_September_2019.csv
186846	259354	iPhone	1	700	09/01/19 16:00	216 Dogwood St, San Francisco, CA 94016	Sales_September_2019.csv
186847	259355	iPhone	1	700	09/23/19 07:39	220 12th St, San Francisco, CA 94016	Sales_September_2019.csv
186848	259356	34in Ultrawide Monitor	1	379.99	09/19/19 17:30	511 Forest St, San Francisco, CA 94016	Sales_September_2019.csv
186849	259357	USB-C Charging Cable	1	11.95	09/30/19 00:18	250 Meadow St, San Francisco, CA 94016	Sales_September_2019.csv

186850 rows × 7 columns

In [6]:
```
data_nan = data[data.isna().any(axis=1)]
data_nan
```

Out[6]:

	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address	filename
1	NaN	NaN	NaN	NaN	NaN	NaN	Sales_April_2019.csv
356	NaN	NaN	NaN	NaN	NaN	NaN	Sales_April_2019.csv
735	NaN	NaN	NaN	NaN	NaN	NaN	Sales_April_2019.csv
1433	NaN	NaN	NaN	NaN	NaN	NaN	Sales_April_2019.csv
1553	NaN	NaN	NaN	NaN	NaN	NaN	Sales_April_2019.csv
...	...	...	...	...	...	...	...
185176	NaN	NaN	NaN	NaN	NaN	NaN	Sales_September_2019.csv
185438	NaN	NaN	NaN	NaN	NaN	NaN	Sales_September_2019.csv
186042	NaN	NaN	NaN	NaN	NaN	NaN	Sales_September_2019.csv
186548	NaN	NaN	NaN	NaN	NaN	NaN	Sales_September_2019.csv
186826	NaN	NaN	NaN	NaN	NaN	NaN	Sales_September_2019.csv

545 rows × 7 columns

In [7]:
```
data.drop('filename', axis=1, inplace=True) # drop filename col
data.dropna(how='all', inplace=True)
data
```

Out[7]:

	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
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
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001
...	...	...	...	...	...	...
186845	259353	AAA Batteries (4-pack)	3	2.99	09/17/19 20:56	840 Highland St, Los Angeles, CA 90001
186846	259354	iPhone	1	700	09/01/19 16:00	216 Dogwood St, San Francisco, CA 94016
186847	259355	iPhone	1	700	09/23/19 07:39	220 12th St, San Francisco, CA 94016
186848	259356	34in Ultrawide Monitor	1	379.99	09/19/19 17:30	511 Forest St, San Francisco, CA 94016
186849	259357	USB-C Charging Cable	1	11.95	09/30/19 00:18	250 Meadow St, San Francisco, CA 94016

186305 rows × 6 columns

In [8]:
```
cols = [col.replace(' ', '') for col in data.columns.tolist()]
data.columns = cols
```

In [9]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 186305 entries, 0 to 186849
Data columns (total 6 columns):
 #   Column           Non-Null Count   Dtype
---  ------           --------------   -----
 0   OrderID          186305 non-null  object
 1   Product          186305 non-null  object
 2   QuantityOrdered  186305 non-null  object
 3   PriceEach        186305 non-null  object
 4   OrderDate        186305 non-null  object
 5   PurchaseAddress  186305 non-null  object
dtypes: object(6)
memory usage: 9.9+ MB
```

In [10]:
```
# check data consitency
qty = ['1', '2', '3', '4', '5', '6', '7', '8', '9']
goodq = data.QuantityOrdered.isin(qty)
data[~goodq]
```

Out[10]:

	OrderID	Product	QuantityOrdered	PriceEach	OrderDate	PurchaseAddress
519	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
1149	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
1155	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
2878	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
2893	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
...	...	...	...	...	...	...
185164	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
185551	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
186563	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
186632	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address
186738	Order ID	Product	Quantity Ordered	Price Each	Order Date	Purchase Address

355 rows × 6 columns

In [11]:
```
data = data[data.OrderID != 'Order ID']
```

Converting data type

In [12]:
```
data
```

Out[12]:

	OrderID	Product	QuantityOrdered	PriceEach	OrderDate	PurchaseAddress
0	176558	USB-C Charging Cable	2	11.95	04/19/19 08:46	917 1st St, Dallas, TX 75001
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
5	176561	Wired Headphones	1	11.99	04/30/19 09:27	333 8th St, Los Angeles, CA 90001
...	...	...	...	...	...	...
186845	259353	AAA Batteries (4-pack)	3	2.99	09/17/19 20:56	840 Highland St, Los Angeles, CA 90001
186846	259354	iPhone	1	700	09/01/19 16:00	216 Dogwood St, San Francisco, CA 94016
186847	259355	iPhone	1	700	09/23/19 07:39	220 12th St, San Francisco, CA 94016
186848	259356	34in Ultrawide Monitor	1	379.99	09/19/19 17:30	511 Forest St, San Francisco, CA 94016
186849	259357	USB-C Charging Cable	1	11.95	09/30/19 00:18	250 Meadow St, San Francisco, CA 94016

185950 rows × 6 columns

In [13]:
```
data.QuantityOrdered = pd.to_numeric(data.QuantityOrdered, downcast='integer', errors='coerce')
data.PriceEach = pd.to_numeric(data.PriceEach, downcast='float', errors='coerce')
data.OrderDate = pd.to_datetime(data.OrderDate, errors='coerce')
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 185950 entries, 0 to 186849
Data columns (total 6 columns):
 #   Column           Non-Null Count   Dtype
---  ------           --------------   -----
 0   OrderID          185950 non-null  object
 1   Product          185950 non-null  object
 2   QuantityOrdered  185950 non-null  int8
 3   PriceEach        185950 non-null  float32
 4   OrderDate        185950 non-null  datetime64[ns]
 5   PurchaseAddress  185950 non-null  object
dtypes: datetime64[ns](1), float32(1), int8(1), object(3)
memory usage: 8.0+ MB
```

Generate Sales by area, month, week & quarter

get area from address

In [14]:
```
data['area'] = data.PurchaseAddress.str.split(',').str.get(1).str.strip()
data
```

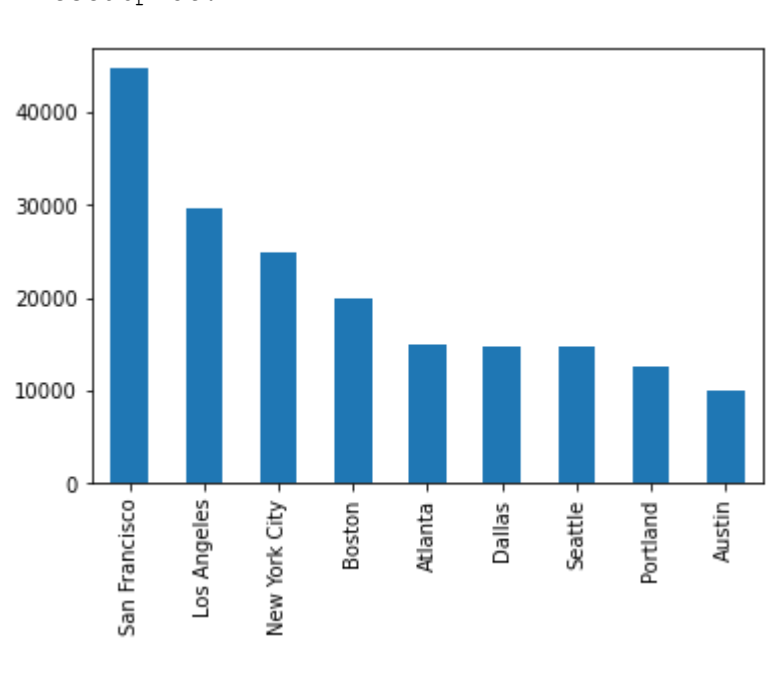
Out[14]:

	OrderID	Product	QuantityOrdered	PriceEach	OrderDate	PurchaseAddress	area
0	176558	USB-C Charging Cable	2	11.950000	2019-04-19 08:46:00	917 1st St, Dallas, TX 75001	Dallas
2	176559	Bose SoundSport Headphones	1	99.989998	2019-04-07 22:30:00	682 Chestnut St, Boston, MA 02215	Boston
3	176560	Google Phone	1	600.000000	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	Los Angeles
4	176560	Wired Headphones	1	11.990000	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	Los Angeles
5	176561	Wired Headphones	1	11.990000	2019-04-30 09:27:00	333 8th St, Los Angeles, CA 90001	Los Angeles
...	...	...	...	...	...	...	...
186845	259353	AAA Batteries (4-pack)	3	2.990000	2019-09-17 20:56:00	840 Highland St, Los Angeles, CA 90001	Los Angeles
186846	259354	iPhone	1	700.000000	2019-09-01 16:00:00	216 Dogwood St, San Francisco, CA 94016	San Francisco
186847	259355	iPhone	1	700.000000	2019-09-23 07:39:00	220 12th St, San Francisco, CA 94016	San Francisco
186848	259356	34in Ultrawide Monitor	1	379.989990	2019-09-19 17:30:00	511 Forest St, San Francisco, CA 94016	San Francisco
186849	259357	USB-C Charging Cable	1	11.950000	2019-09-30 00:18:00	250 Meadow St, San Francisco, CA 94016	San Francisco

185950 rows × 7 columns

In [15]:
```
data.area.value_counts().plot(kind='bar') # by number of sales
```

Out[15]: <AxesSubplot:>



In [16]:
```
df = data.copy()
```

break down the date into sub-components like month, week no, day & quarter

In [17]:
```
df['year'] = df.OrderDate.dt.year
df['mth'] = df.OrderDate.dt.month
#df.mth = df.mth.map(lambda x: str(x) if x > 9 else '0'+str(x))
df['dayname'] = df.OrderDate.dt.strftime("%A")
df['week'] = df.OrderDate.dt.isocalendar().week
df['quarter'] = df.OrderDate.dt.quarter
df.head()
```

Out[17]:

	OrderID	Product	QuantityOrdered	PriceEach	OrderDate	PurchaseAddress	area	year	mth	dayname	week	quarter
0	176558	USB-C Charging Cable	2	11.950000	2019-04-19 08:46:00	917 1st St, Dallas, TX 75001	Dallas	2019	4	Friday	16	2
2	176559	Bose SoundSport Headphones	1	99.989998	2019-04-07 22:30:00	682 Chestnut St, Boston, MA 02215	Boston	2019	4	Sunday	14	2
3	176560	Google Phone	1	600.000000	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	Los Angeles	2019	4	Friday	15	2
4	176560	Wired Headphones	1	11.990000	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	Los Angeles	2019	4	Friday	15	2
5	176561	Wired Headphones	1	11.990000	2019-04-30 09:27:00	333 8th St, Los Angeles, CA 90001	Los Angeles	2019	4	Tuesday	18	2

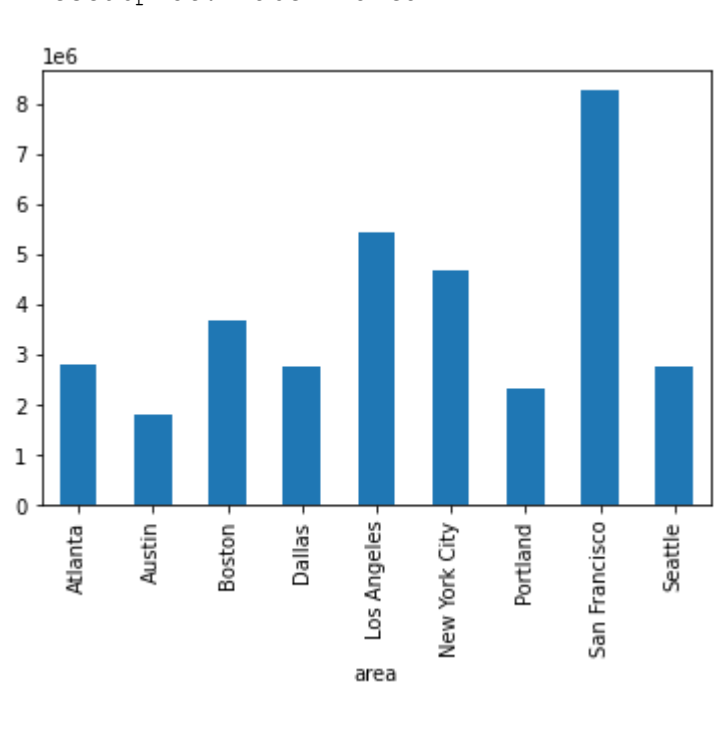
In [18]:
```
df['subtotal'] = df.QuantityOrdered * df.PriceEach
df.head()
```

Out[18]:

	OrderID	Product	QuantityOrdered	PriceEach	OrderDate	PurchaseAddress	area	year	mth	dayname	week	quarter	sub
0	176558	USB-C Charging Cable	2	11.950000	2019-04-19 08:46:00	917 1st St, Dallas, TX 75001	Dallas	2019	4	Friday	16	2	23.90
2	176559	Bose SoundSport Headphones	1	99.989998	2019-04-07 22:30:00	682 Chestnut St, Boston, MA 02215	Boston	2019	4	Sunday	14	2	99.98
3	176560	Google Phone	1	600.000000	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	Los Angeles	2019	4	Friday	15	2	600.00
4	176560	Wired Headphones	1	11.990000	2019-04-12 14:38:00	669 Spruce St, Los Angeles, CA 90001	Los Angeles	2019	4	Friday	15	2	11.99
5	176561	Wired Headphones	1	11.990000	2019-04-30 09:27:00	333 8th St, Los Angeles, CA 90001	Los Angeles	2019	4	Tuesday	18	2	11.99

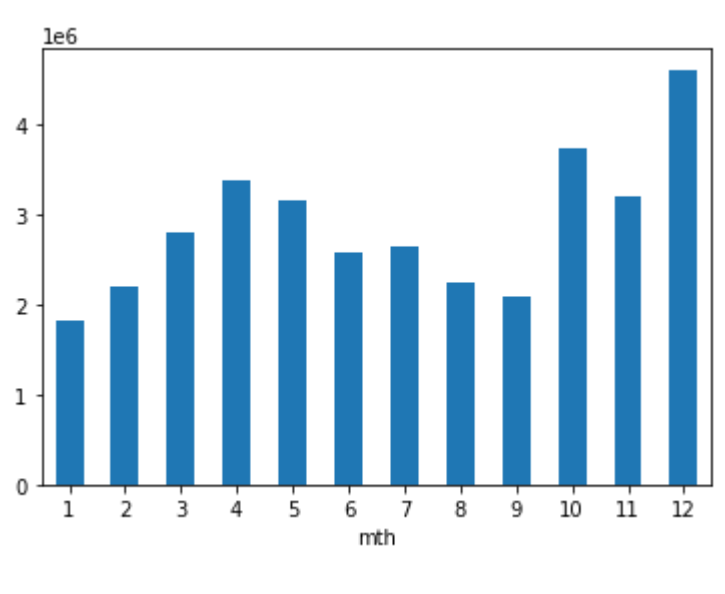
In [19]:
```
df.groupby(by='area').subtotal.sum().plot(kind='bar')
```

Out[19]: <AxesSubplot:xlabel='area'>



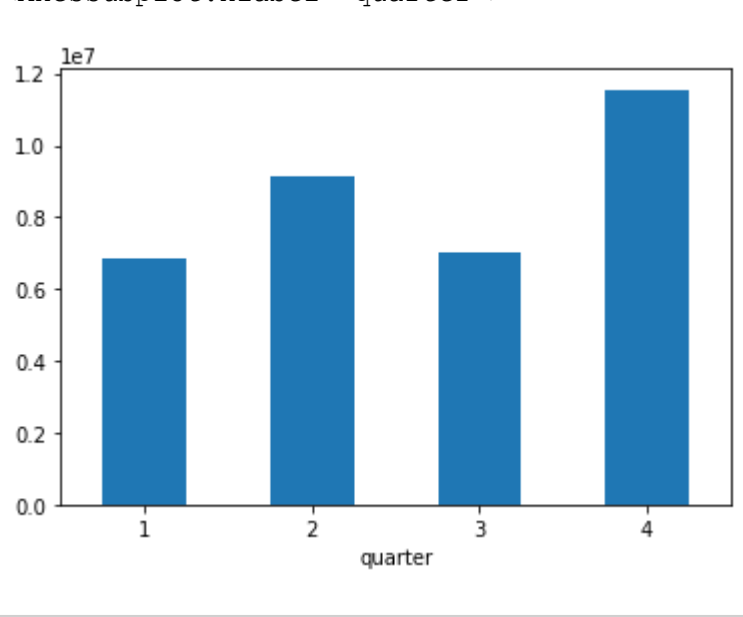
In [20]:
```
df.groupby(by='mth').subtotal.sum().plot(kind='bar', rot=0)
```

Out[20]: <AxesSubplot:xlabel='mth'>



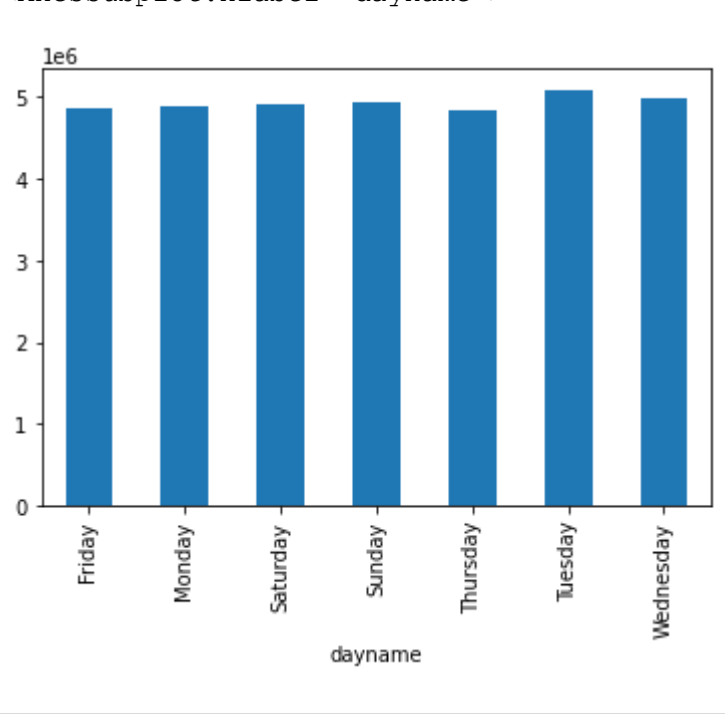
In [21]:
```
df.groupby(by='quarter').subtotal.sum().plot(kind='bar', rot=0)
```

Out[21]: <AxesSubplot:xlabel='quarter'>



In [22]:
```
df.groupby(by='dayname').subtotal.sum().plot(kind='bar')
```

Out[22]: <AxesSubplot:xlabel='dayname'>

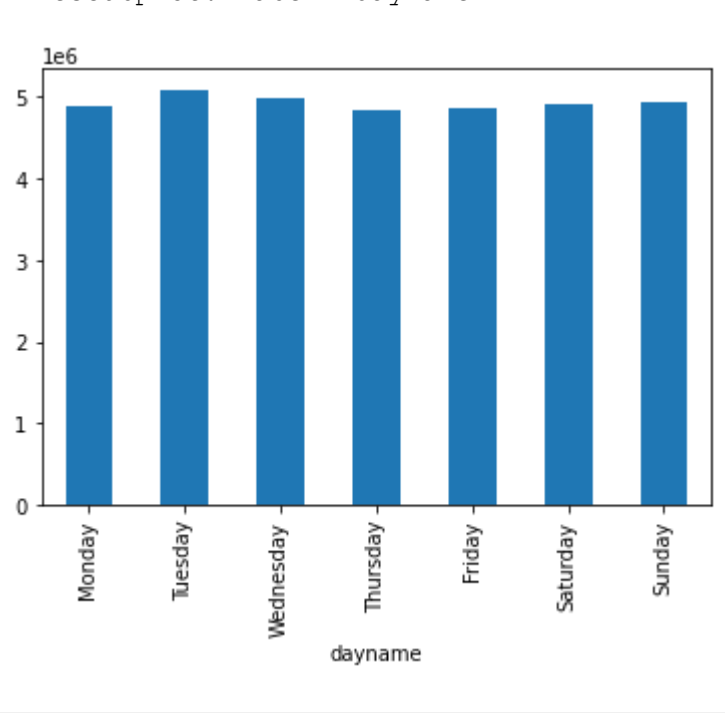


In [23]:
```
daysales = df.groupby(by='dayname').subtotal.sum()
daysales
```

Out[23]:
```
dayname
Friday       4855938.5
Monday       4883326.5
Saturday     4904357.0
Sunday       4932169.5
Thursday     4839465.0
Tuesday      5087957.0
Wednesday    4988822.0
Name: subtotal, dtype: float32
```

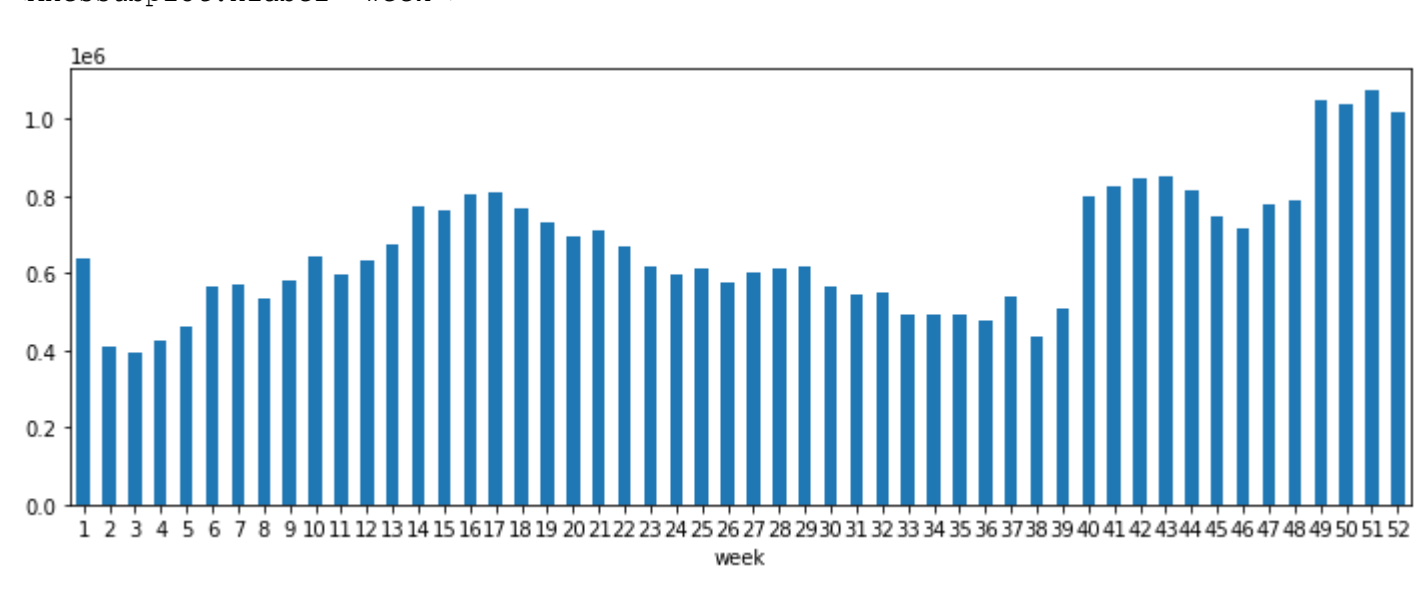
In [24]:
```
daysales = daysales.reindex(['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunda
y'])
daysales.plot(kind='bar')
```

Out[24]: <AxesSubplot:xlabel='dayname'>



In [25]:
```
df.groupby(by='week').subtotal.sum().plot(kind='bar', figsize=(12, 4), rot=0)
```

Out[25]: <AxesSubplot:xlabel='week'>



Assign sub-category by keywords

In [26]:
```
df.Product.value_counts()
```

Out[26]:
```
USB-C Charging Cable          21903
Lightning Charging Cable      21658
AAA Batteries (4-pack)        20641
AA Batteries (4-pack)         20577
Wired Headphones              18882
Apple Airpods Headphones      15549
Bose SoundSport Headphones    13325
27in FHD Monitor               7507
iPhone                         6842
27in 4K Gaming Monitor         6230
34in Ultrawide Monitor         6181
Google Phone                   5525
Flatscreen TV                  4800
Macbook Pro Laptop             4724
ThinkPad Laptop                4128
20in Monitor                   4101
Vareebadd Phone                2065
LG Washing Machine              666
LG Dryer                        646
Name: Product, dtype: int64
```

In [27]:
```
def fxkws2cat(df=None, kwscol=None, kwsdict=None, newcol='sub_category', nomatch='others', casesensitiv
e=False):
    for k, v in kwsdict.items():
        df.loc[df[kwscol].str.contains(k, case=casesensitive, regex=True), newcol] = v
    df[newcol].fillna(nomatch, inplace=True)
    return df
```

In [28]:
```
keywords = {'Cable': 'cable', 'Phone': 'Phone', 'Batteries': 'batteries', 'Monitor': 'Screen',
            'Headphones': 'Headphones', 'Laptop': 'Laptop'}
keywords
```

Out[28]:
```
{'Cable': 'cable',
 'Phone': 'Phone',
 'Batteries': 'batteries',
 'Monitor': 'Screen',
 'Headphones': 'Headphones',
 'Laptop': 'Laptop'}
```

In [29]:
```
df = fxkws2cat(df=df, kwscol='Product', kwsdict=keywords)
```

In [30]:
```
df.sub_category.value_counts()
```

Out[30]:
```
Headphones    47756
cable         43561
batteries     41218
Screen        24019
Phone         14432
Laptop         8852
others         6112
Name: sub_category, dtype: int64
```

In [31]:
```
df_salescat = df.groupby('sub_category').agg({'sub_category': ['count'], 'subtotal': 'sum'})
df_salescat
```

Out[31]:

	sub_category	subtotal
	count	sum
sub_category		
Headphones	47756	3.941194e+06
Laptop	8852	1.216756e+07
Phone	14432	8.940700e+06
Screen	24019	6.377228e+06
batteries	41218	1.988592e+05
cable	43561	6.335954e+05
others	6112	2.232900e+06

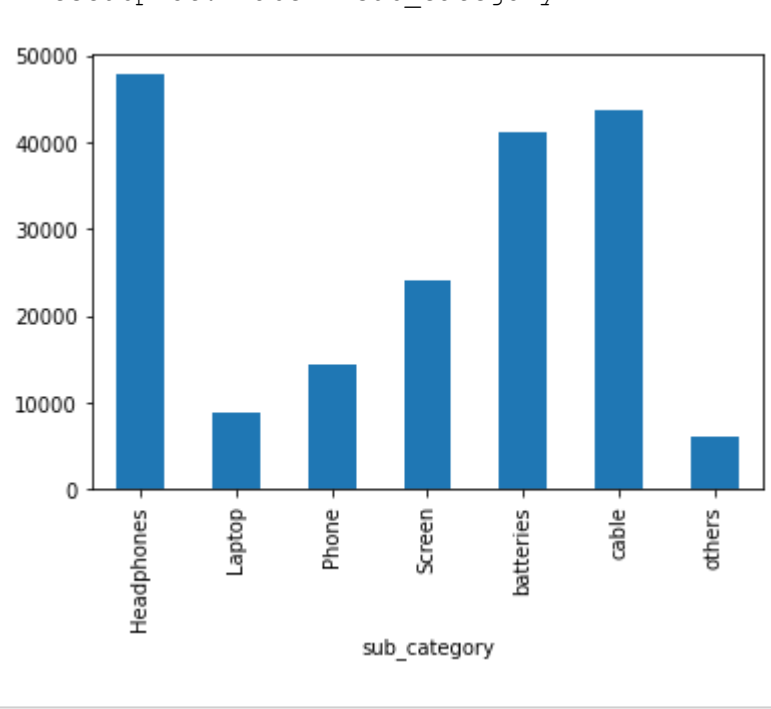
In [32]:
```
df_salescat.columns = ['item', 'value']
df_salescat.sort_values('item', ascending=False)
```

Out[32]:

	item	value
sub_category		
Headphones	47756	3.941194e+06
cable	43561	6.335954e+05
batteries	41218	1.988592e+05
Screen	24019	6.377228e+06
Phone	14432	8.940700e+06
Laptop	8852	1.216756e+07
others	6112	2.232900e+06

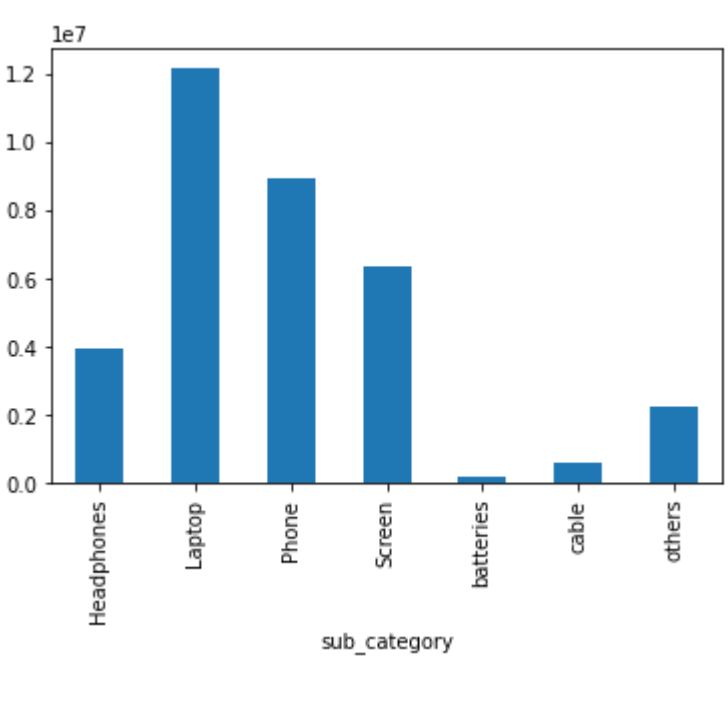
In [33]:
```
df_salescat.item.plot(kind='bar')
```

Out[33]: <AxesSubplot:xlabel='sub_category'>



In [34]:
```
df_salescat.value.plot(kind='bar')
```

Out[34]: <AxesSubplot:xlabel='sub_category'>



Find purchase-together items

In [35]:
```
def PurchaseTogether(df, RefCol, RefItems, n=10):
    # Summarized from Keith's work
    from itertools import combinations
    from collections import Counter

    df_ref = df[df[RefCol].duplicated(keep=False)]
    df_ref['Grouped'] =   df_ref.groupby(RefCol)[RefItems].transform(lambda x: ','.join(x))
    df_idxitems = df_ref[[RefCol, 'Grouped']].drop_duplicates()
    #print(df_idxitems)

    count = Counter()

    for row in df_idxitems['Grouped']:
        row_list = row.split(',')
        count.update(Counter(combinations(row_list, 2)))

    for key,value in count.most_common(n):
        print(key, value)
```

In [36]:
```
PurchaseTogether(df, 'OrderID', 'Product')
```

```
('iPhone', 'Lightning Charging Cable') 1005
('Google Phone', 'USB-C Charging Cable') 987
('iPhone', 'Wired Headphones') 447
('Google Phone', 'Wired Headphones') 414
('Vareebadd Phone', 'USB-C Charging Cable') 361
('iPhone', 'Apple Airpods Headphones') 360
('Google Phone', 'Bose SoundSport Headphones') 220
('USB-C Charging Cable', 'Wired Headphones') 160
('Vareebadd Phone', 'Wired Headphones') 143
('Lightning Charging Cable', 'Wired Headphones') 92
```

In [ ]: