1. import libraries, dataframe

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

In [3]:
```
# Define path
path = r'C:\Users\Personal Computer\Documents\09.2022_Instacart_Basket_Analysis'
```

In [4]:
```
# Import order_products_merged.pkl
# Note -- I would have named this differently, but I already did this before doing Part
ords_prods_merge = pd.read_pickle(os.path.join(path, '02 Data', 'Prepared Data', 'insta
```

2. do all the tasks in the exercise, create charts

In [4]:
```
ords_prods_merge.head(10)
```

Out[4]:

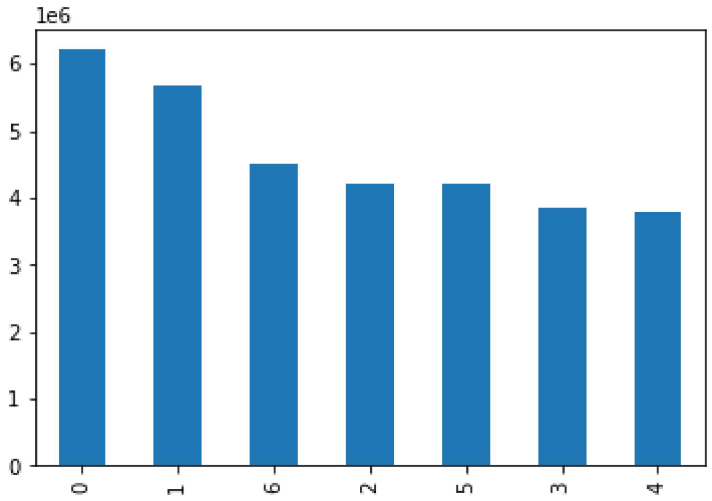
	order_id	user_id	user_order_number	orders_day_of_week	order_hour_of_day	days_since_prior_order
0	2539329	1	1	2	8	NaN
1	2398795	1	2	3	7	15.0
2	473747	1	3	3	12	21.0
3	2254736	1	4	4	7	29.0
4	431534	1	5	4	15	28.0
5	3367565	1	6	2	7	19.0
6	550135	1	7	1	9	20.0
7	3108588	1	8	1	14	14.0
8	2295261	1	9	1	16	0.0
9	2550362	1	10	4	8	30.0

10 rows × 34 columns

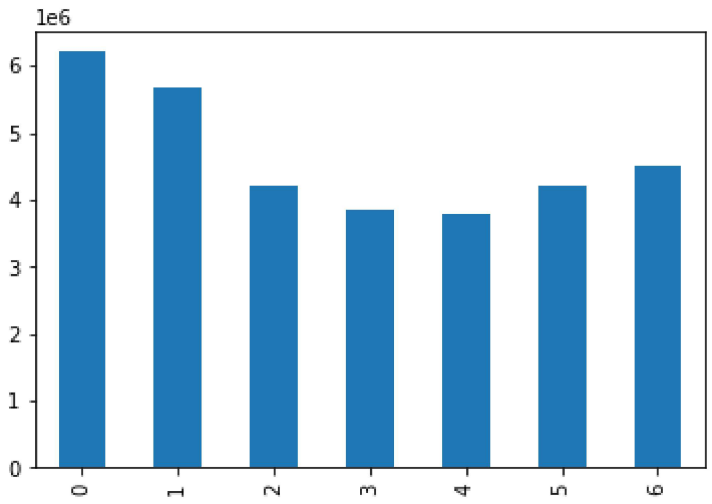
creating bar charts

In [5]:
```
# create a bar chart
ords_prods_merge['orders_day_of_week'].value_counts().plot.bar()
```

Out[5]: <AxesSubplot:>



In [6]:
```
# create a sorted bar chart
bar = ords_prods_merge['orders_day_of_week'].value_counts().sort_index().plot.bar()
```



In [12]:
```
sns.set(rc = {'figure.figsize':(15,5)})
baragain = ords_prods_merge['orders_day_of_week'].value_counts().sort_index().plot.bar(
plt.title('Orders By Day', fontsize = 20)
plt.xlabel('Day of Week', fontsize = 20)
plt.ylabel('Number of Customers', fontsize = 20)
plt.legend(fontsize=20)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
```

```
C:\Users\PERSON~1\AppData\Local\Temp/ipykernel_12864/1782417961.py in <module>
      5 plt.ylabel('Number of Customers', fontsize = 20)
      6 plt.legend(fontsize=20)
----> 7 baragain.to_clipboard()

AttributeError: 'AxesSubplot' object has no attribute 'to_clipboard'
```

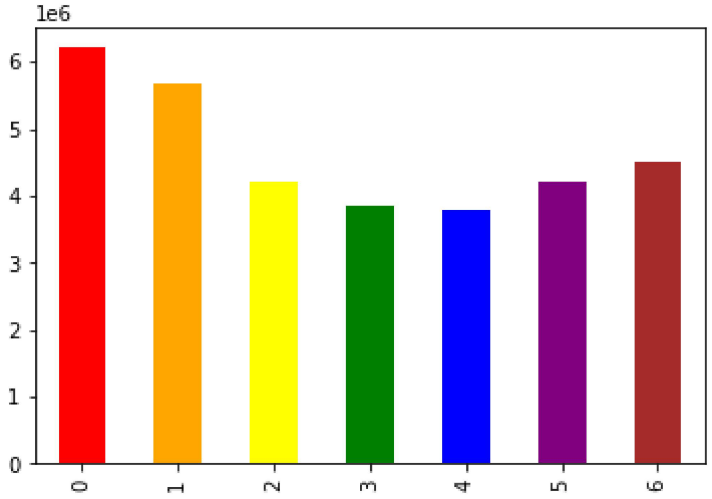
In [7]:
```
ords_prods_merge['orders_day_of_week'].value_counts()
```

Out[7]:
```
0    6204182
1    5660230
6    4496490
2    4213830
5    4205791
3    3840534
4    3783802
Name: orders_day_of_week, dtype: int64
```

In [13]:
```
bar = ords_prods_merge['orders_day_of_week'].value_counts().sort_index()
```

In [14]:
```
bar.to_clipboard()
```

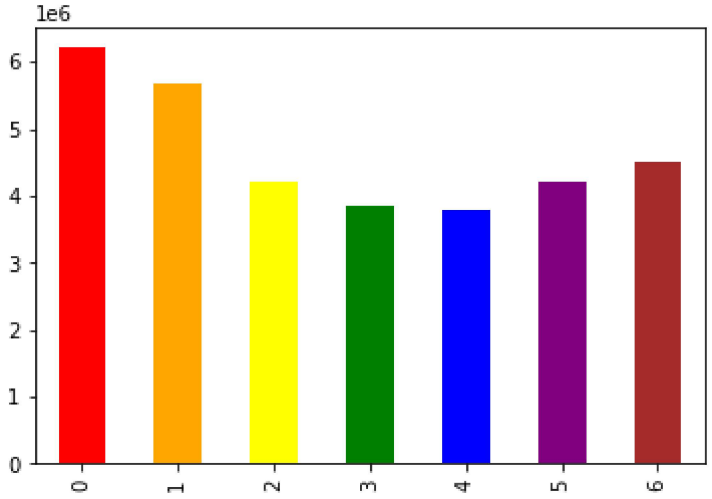
In [75]:
```
# create a sorted bar chart with colors
bar = ords_prods_merge['orders_day_of_week'].value_counts().sort_index().plot.bar(color
```



exporting charts

In [10]:
```
bar.figure
```

Out[10]:

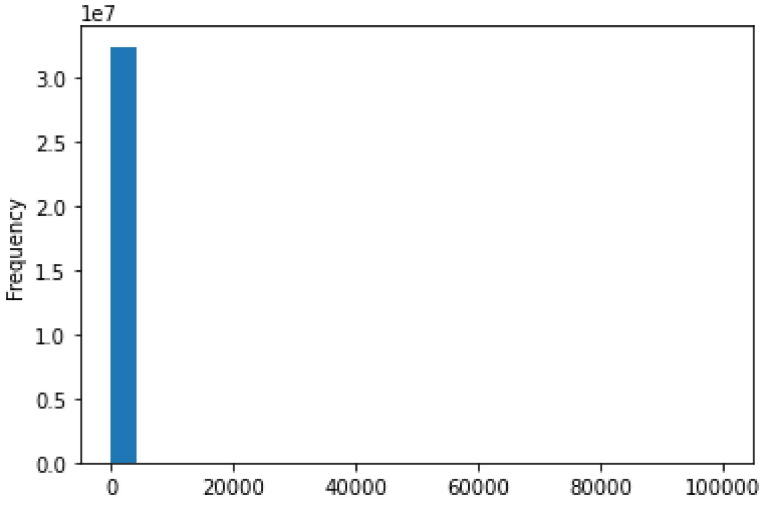


In [11]:
```
# save to folder
bar.figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'bar_orders_dow.
```

creating histograms and scatterplots

In [12]:
```
# create a histogram
ords_prods_merge['prices'].plot.hist(bins = 25)
```

Out[12]:
```
<AxesSubplot:ylabel='Frequency'>
```



In [13]:
```
ords_prods_merge['prices'].describe()
```

Out[13]:
```
count    3.240486e+07
mean     1.198023e+01
std      4.956554e+02
min      1.000000e+00
25%      4.200000e+00
50%      7.400000e+00
75%      1.130000e+01
max      9.999900e+04
Name: prices, dtype: float64
```

In [14]:
```
ords_prods_merge['prices'].mean()
```

Out[14]:
```
11.98022563865405
```

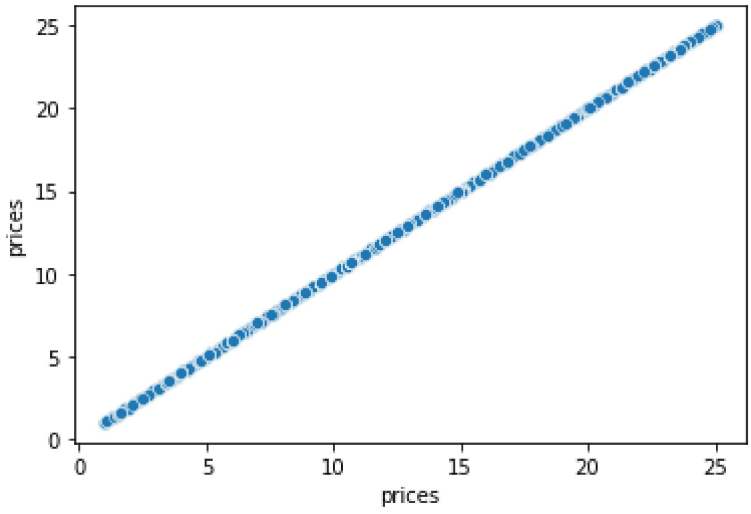
In [15]:
```
ords_prods_merge['prices'].median()
```

Out[15]:
```
7.4
```

In [16]:
```
ords_prods_merge['prices'].max()
```

Out[16]:
```
99999.0
```

In [73]:
```
# create a scatterplot
sccatter_1 = sns.scatterplot(x = 'prices', y = 'prices',data = ords_prods_merge)
```



In [18]:
```
ords_prods_merge.loc[ords_prods_merge['prices'] > 100]
```

Out[18]:

	order_id	user_id	user_order_number	orders_day_of_week	order_hour_of_day	days_since_prio
7805	1435153	1519	26	2	11	
7806	1066435	1519	32	3	10	
15143	1697208	2409	33	1	19	
20264	965345	3531	27	2	19	
20497	2943740	3793	1	2	9	
...	...	...	...	...	...	
32303799	3265389	51346	4	1	17	
32307882	1179092	131671	12	4	9	
32310810	1226705	39667	7	2	15	
32330048	3000037	95105	2	4	19	
32330049	2741225	95105	4	5	12	

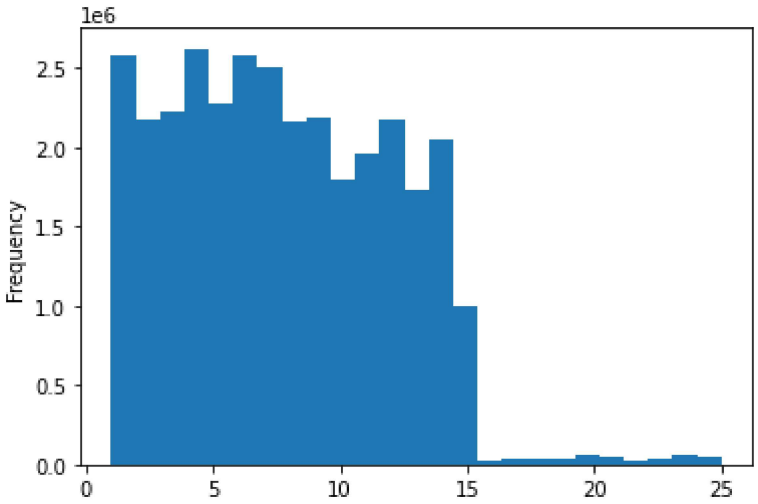
5127 rows × 34 columns

In [19]:
```
ords_prods_merge.loc[ords_prods_merge['prices'] > 100, 'prices'] = np.nan
```

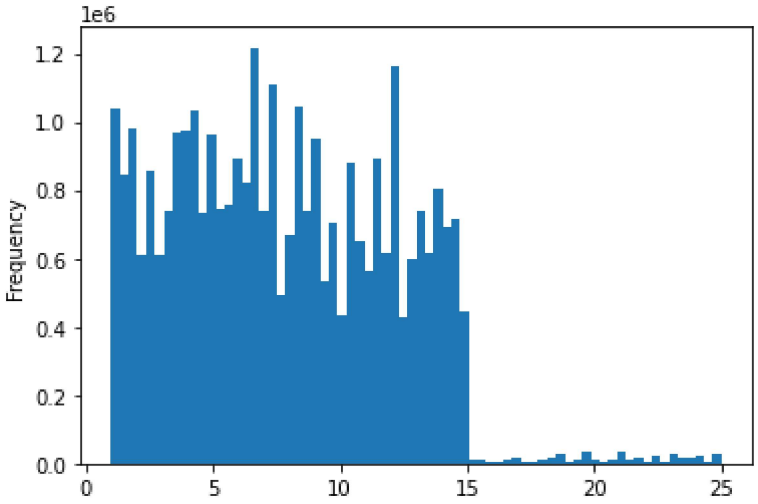
In [20]:
```
ords_prods_merge['prices'].max()
```

Out[20]:
```
25.0
```

In [21]:
```
# create a histogram (25 bins)
hist = ords_prods_merge['prices'].plot.hist(bins = 25)
```



In [22]:
```
# create a histogram (70 bins)
hist = ords_prods_merge['prices'].plot.hist(bins = 70)
```



In [23]:
```
# save to folder
hist.figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'hist_prices.pn
```

creating line charts; sampling rows

In [24]:
```
# sample with 70/30 split
np.random.seed(4)
dev = np.random.rand(len(ords_prods_merge)) <= 0.7
```

In [25]:
```
dev
```

Out[25]:
```
array([False,  True, False, ...,  True,  True,  True])
```

In [26]:
```
np.random.rand(10)
```

Out[26]:
```
array([0.93546686, 0.06557465, 0.85698584, 0.24456371, 0.22683171,
       0.17068366, 0.27008946, 0.52534234, 0.83151571, 0.78153402])
```

In [27]:
```
# split dataframe; big = 70% / small = 30%
big = ords_prods_merge[dev]
small = ords_prods_merge[~dev]
```

In [28]:
```
# confirm split = dataframe
len(ords_prods_merge)
len(big) + len(small)
```

Out[28]:
```
32404859
```

In [29]:
```
len(ords_prods_merge)
```

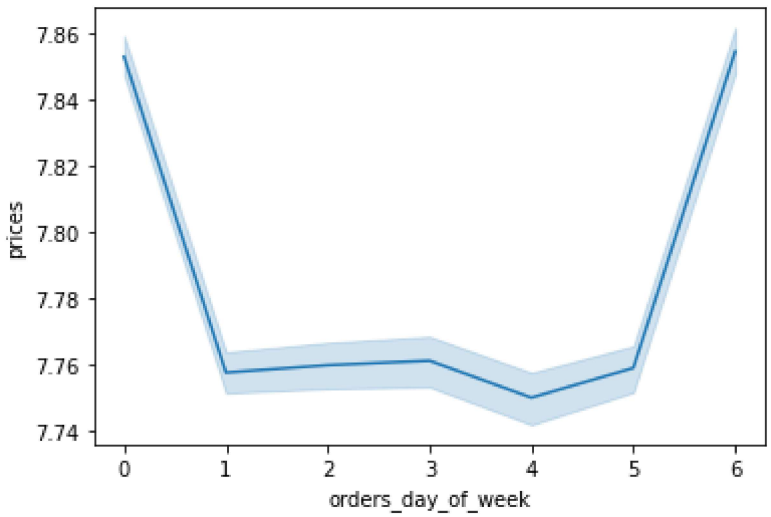
Out[29]:
```
32404859
```

In [30]:
```
len(big) + len(small)
```

Out[30]:
```
32404859
```

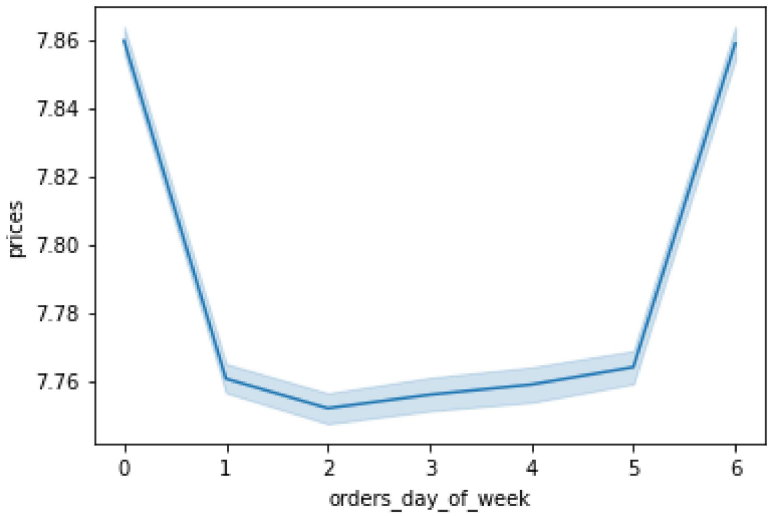
In [31]:
```
# sample orders dow, price columns - SMALL
df2 = small[['orders_day_of_week', 'prices']]
```

In [32]:
```
line = sns.lineplot(data = df2, x = 'orders_day_of_week', y ='prices')
```



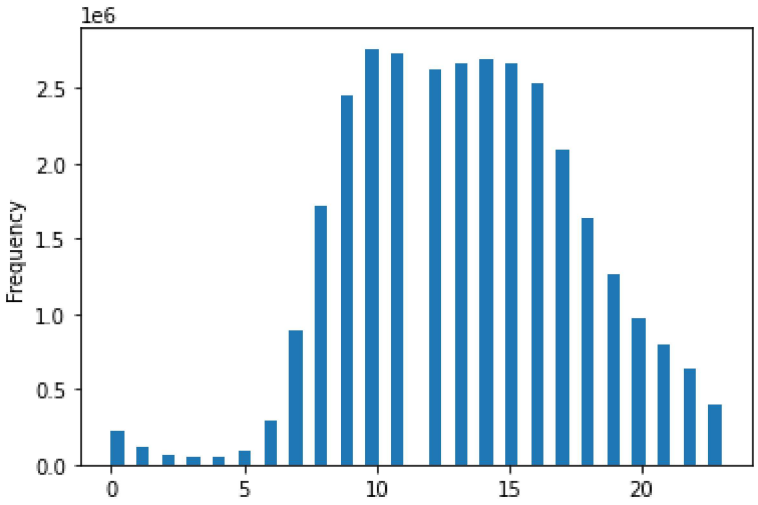
In [33]:
```
# sample orders dow, price columns - BIG
df3 = big[['orders_day_of_week', 'prices']]
```

In [34]:
```
line2 = sns.lineplot(data = df3, x = 'orders_day_of_week', y ='prices')
```



3. Mgmt wants an order_hour_of_day sales findings

In [42]:
```
# create a order_hour histogram (48 bins) [24 hours in a day, went with half hour bins]
hist_2 = ords_prods_merge['order_hour_of_day'].plot.hist(bins = 48)
```



In [16]:
```
# create a order_hour histogram (48 bins) [24 hours in a day, went with half hour bins]
sns.set(rc = {'figure.figsize':(15,5)})
hist_2 = ords_prods_merge['order_hour_of_day'].plot.hist(bins = 48)
plt.title('Orders By Hours of Day', fontsize = 20)
plt.xlabel('Hour', fontsize = 20)
plt.ylabel('Number of Order', fontsize = 20)
plt.legend(fontsize=20)
```

Out[16]:
```
<matplotlib.legend.Legend at 0x1b80d205580>
```

In [19]:
```
hours = ords_prods_merge['order_hour_of_day'].value_counts().sort_index()
plt.title('Orders By Hours of Day', fontsize = 20)
plt.legend(fontsize=20)
```

In [ ]:
```
```

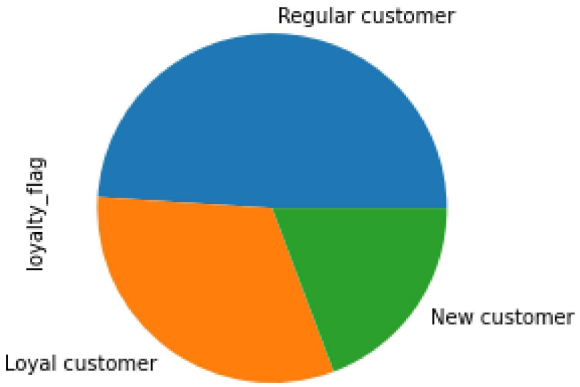
In [20]:
```
hours.to_clipboard()
```

orders tend to spike in late morning (before 10a) and start go down in the afternoon (after 4p)

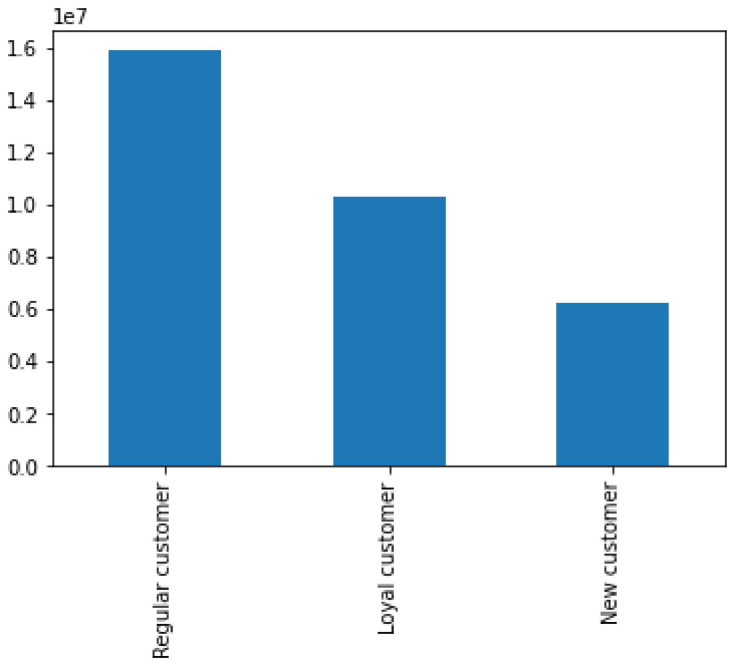
4. Marketing wants to know about customer loyalty

In [43]:
```
# Make a pie chart to show
sns.set(rc = {'figure.figsize':(15,5)})

pieloy = ords_prods_merge['loyalty_flag'].value_counts().plot.pie()
```



In [74]:
```
# Maybe a bar chart would be better
bar2 = ords_prods_merge['loyalty_flag'].value_counts().plot.bar()
```

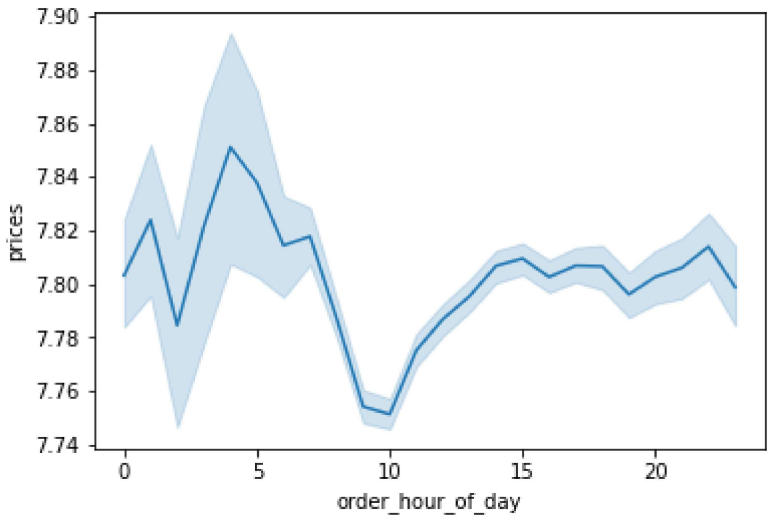


Regular customers comprise the group that makes up nearly half of orders overall

5. Is there a relationship between prices and hour of day?

In [48]:
```
# sample order_hour, price columns - BIG
df4 = big[['order_hour_of_day', 'prices']]
```

In [49]:
```
line_q_5 = sns.lineplot(data = df4, x = 'order_hour_of_day', y ='prices')
```

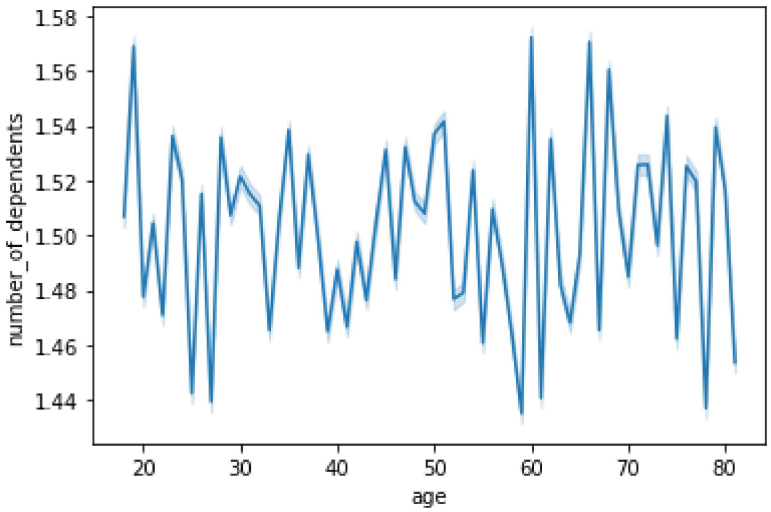


it looks like there are more high prices orders happening earler in the day and start to drop around 7 and 8, bottoming out at 10, and the rising back up throughout the afternoon and maining a relatively consistent level through the rest of the day

6. Exploratory analysis: age v. family status, number of dependents

In [50]:
```
# sample age and family_status - BIG
df5 = big[['age', 'number_of_dependents']]
```

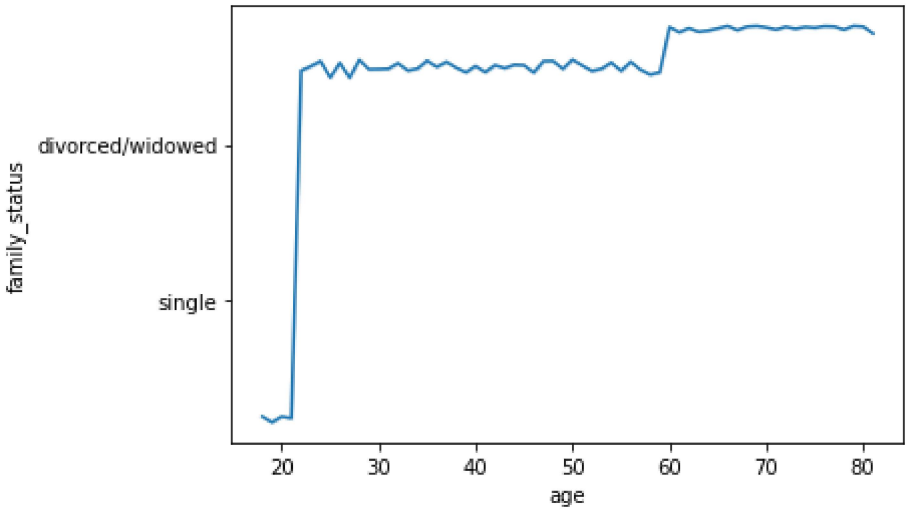
In [51]:
```
line_q_6 = sns.lineplot(data = df5, x = 'age', y ='number_of_dependents')
```



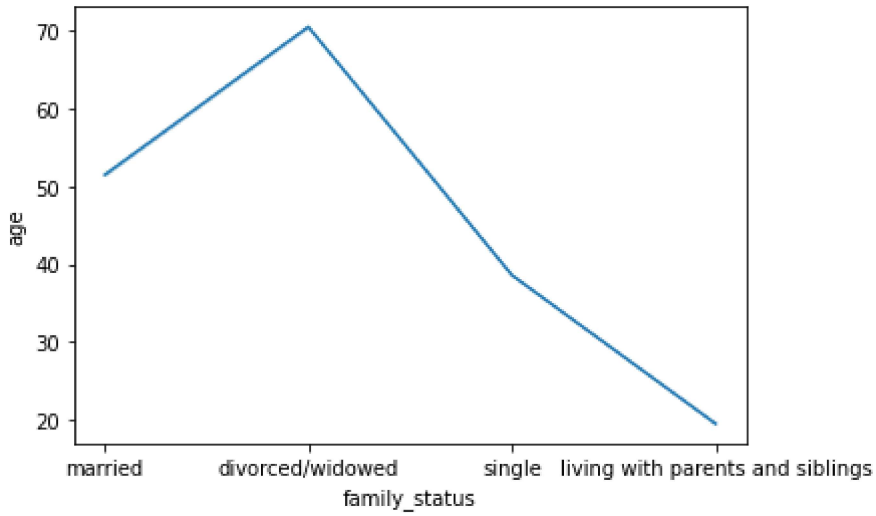
I think there is too much fluctuation to say there is any trend with respect to age and number of dependents

In [52]:
```
# sample age and family_status - BIG
df6 = big[['age', 'family_status']]
```

In [79]:
```
line_q_6_1 = sns.lineplot(data = df6, x = 'age', y ='family_status')
```



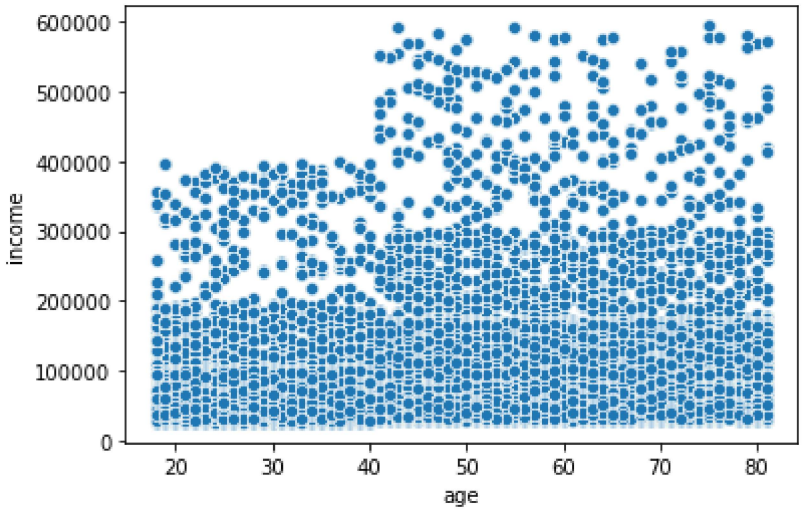
In [76]:
```
line_q_6_2 = sns.lineplot(data = df6, x = 'family_status', y ='age')
```



it would make sense that the older people get, particularly above twenty, the more dpenedents they would have; however, I do not know this information is actionable with respect to customer demographics

7. Is there a connection between age and income?

In [72]:
```
# create a scatterplot
scatter_2 = sns.scatterplot(x = 'age', y = 'income',data = ords_prods_merge)
```



there is a noticeable division in terms of age relative to spending potential is around age 40; while each age/age group has its outliers/higher earners, it is right around the 40 year mark that there seems to be a demographic shift with increased power through the 80s.

8. Export Charts

In [80]:
```
# all charts in this achievement
bar.figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'final_bar_order
sccatter_1 .figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'final_s
hist.figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'final_hist_pri
line.figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'final_line_ord
line2.figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'final_line_or
hist_2.figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'final_hist_o
pie.figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'final_pie_loyat
bar2.figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'final_bar_loya
line_q_5.figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'final_pric
line_q_6.figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'final_age_
line_q_6_1.figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'final_ag
line_q_6_2.figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'final_fa
scatter_2.figure.savefig(os.path.join(path, '04 Analysis', 'Visualizations', 'final_age
```

In [ ]: