In [1]:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns
color = sns.color_palette()

%matplotlib inline
pd.options.mode.chained_assignment = None  # default='warn'
```

```
C:\Users\vikas.rana\AppData\Local\Continuum\Anaconda3\envs\tensorflow\lib\site-packages\IPython\html.py:14: S
himWarning: The `IPython.html` package has been deprecated since IPython 4.0. You should import from `noteboo
k` instead. `IPython.html.widgets` has moved to `ipywidgets`.
  "`IPython.html.widgets` has moved to `ipywidgets`.", ShimWarning)
```

In [2]:

```
import os
os.chdir('D:\\Kaggle_Compi\\Insta_cart\\instacart_online_grocery_shopping_2017_05_01\\Data\\Data_all')
```

In [3]:

```
import os
cwd = os.getcwd()
cwd
```

Out[3]:

```
'D:\\Kaggle_Compi\\Insta_cart\\instacart_online_grocery_shopping_2017_05_01\\Data\\Data_all'
```

In [4]:

```
order_products_train_df = pd.read_csv("D:\\Kaggle_Compi\\Insta_cart\\instacart_online_grocery_shopping_2017_0
5_01\\Data\\Data_all\\order_products__train.csv")
order_products_prior_df = pd.read_csv("D:\\Kaggle_Compi\\Insta_cart\\instacart_online_grocery_shopping_2017_0
5_01\\Data\\Data_all\\order_products__prior.csv")
orders_df = pd.read_csv("D:\\Kaggle_Compi\\Insta_cart\\instacart_online_grocery_shopping_2017_05_01\\Data\\Da
ta_all\\orders.csv")
products_df =
pd.read_csv("D:\\Kaggle_Compi\\Insta_cart\\instacart_online_grocery_shopping_2017_05_01\\Data\\Data_all\\prod
ucts.csv")
aisles_df = pd.read_csv("D:\\Kaggle_Compi\\Insta_cart\\instacart_online_grocery_shopping_2017_05_01\\Data\\Da
ta_all\\aisles.csv")
departments_df = pd.read_csv("D:\\Kaggle_Compi\\Insta_cart\\instacart_online_grocery_shopping_2017_05_01\\Dat
a\\Data_all\\departments.csv")
```

In [6]:

```
orders_df.head(10)
```

Out[6]:

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

In [7]:

```
order_products_prior_df.head()
```

Out[7]:

	order_id	product_id	add_to_cart_order	reordered
0	2	33120	1	1
1	2	28985	2	1
2	2	9327	3	0
3	2	45918	4	1
4	2	30035	5	0

In [9]:
```
order_products_train_df.head()
```

Out[9]:

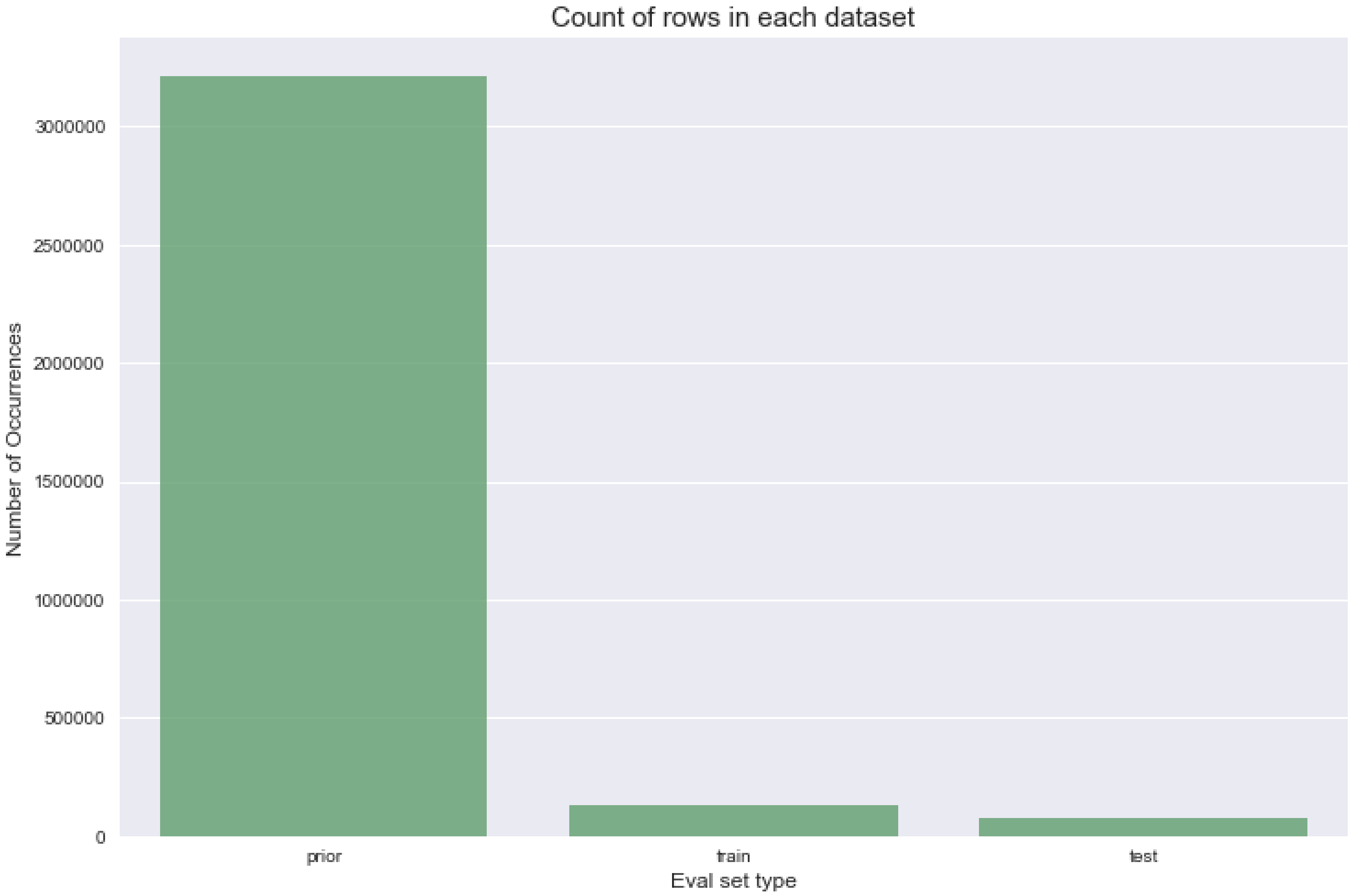
	order_id	product_id	add_to_cart_order	reordered
0	1	49302	1	1
1	1	11109	2	1
2	1	10246	3	0
3	1	49683	4	0
4	1	43633	5	1

In [31]:
```
cnt_srs
```

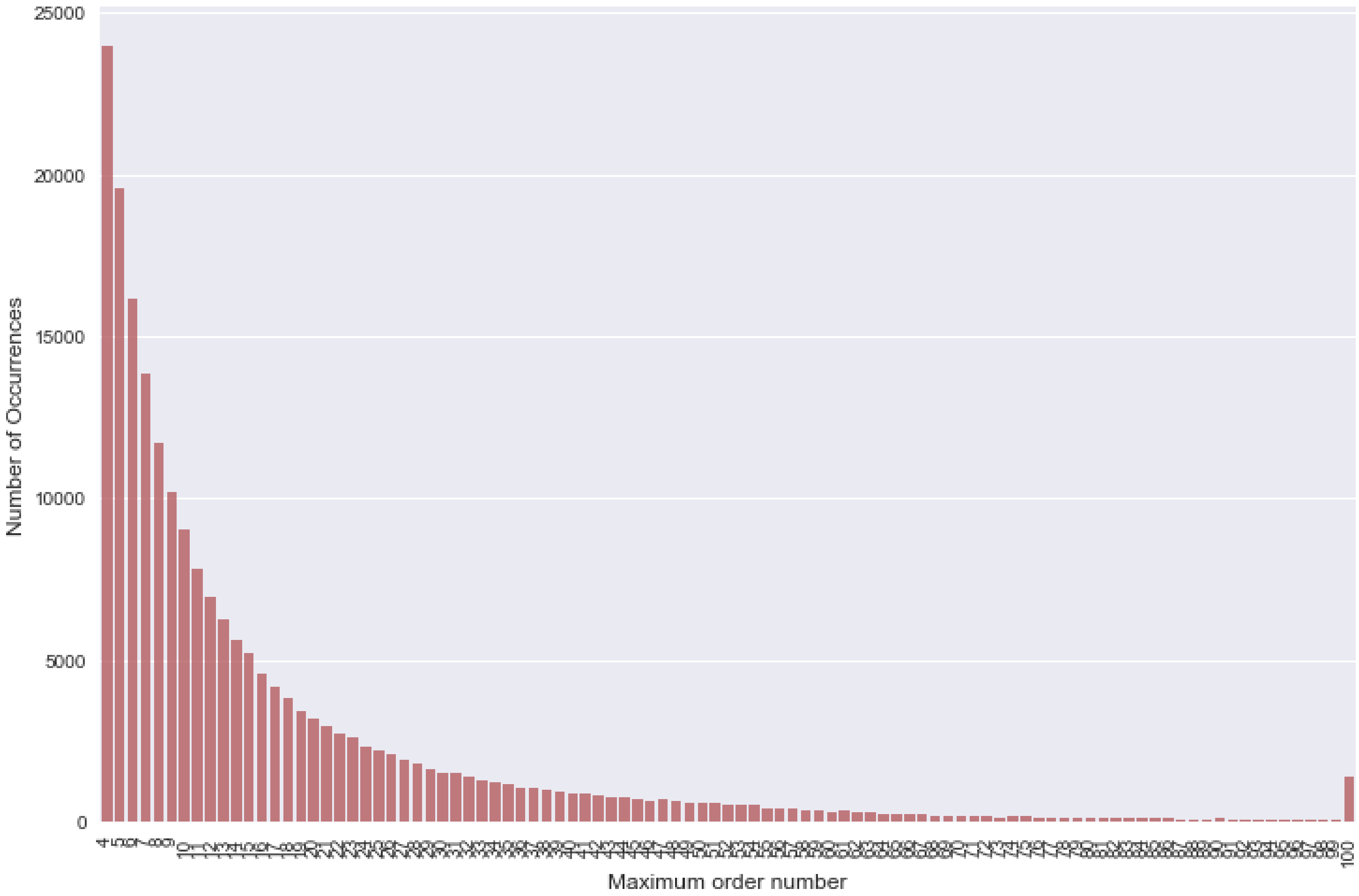
Out[31]:
```
fresh fruits                     3642188
fresh vegetables                 3418021
packaged vegetables fruits       1765313
yogurt                           1452343
packaged cheese                   979763
milk                              891015
water seltzer sparkling water     841533
chips pretzels                    722470
soy lactosefree                   638253
bread                             584834
refrigerated                      575881
frozen produce                    522654
ice cream ice                     498425
crackers                          458838
energy granola bars               456386
eggs                              452134
lunch meat                        395130
frozen meals                      390299
baby food formula                 382456
fresh herbs                       377741
Name: aisle, dtype: int64
```

In [8]:

```
cnt_srs = orders_df.eval_set.value_counts()

plt.figure(figsize=(12,8))
sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8, color=color[1])
plt.ylabel('Number of Occurrences', fontsize=12)
plt.xlabel('Eval set type', fontsize=12)
plt.title('Count of rows in each dataset', fontsize=15)
plt.xticks(rotation='horizontal')
plt.show()
```

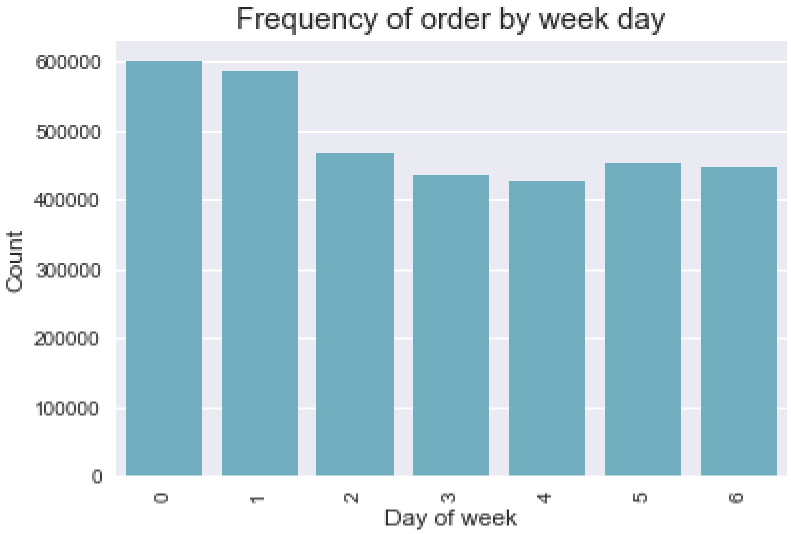


In [9]:
```
cnt_srs = orders_df.groupby("user_id")["order_number"].aggregate(np.max).reset_index()
cnt_srs = cnt_srs.order_number.value_counts()

plt.figure(figsize=(12,8))
sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8, color=color[2])
plt.ylabel('Number of Occurrences', fontsize=12)
plt.xlabel('Maximum order number', fontsize=12)
plt.xticks(rotation='vertical')
plt.show()
```



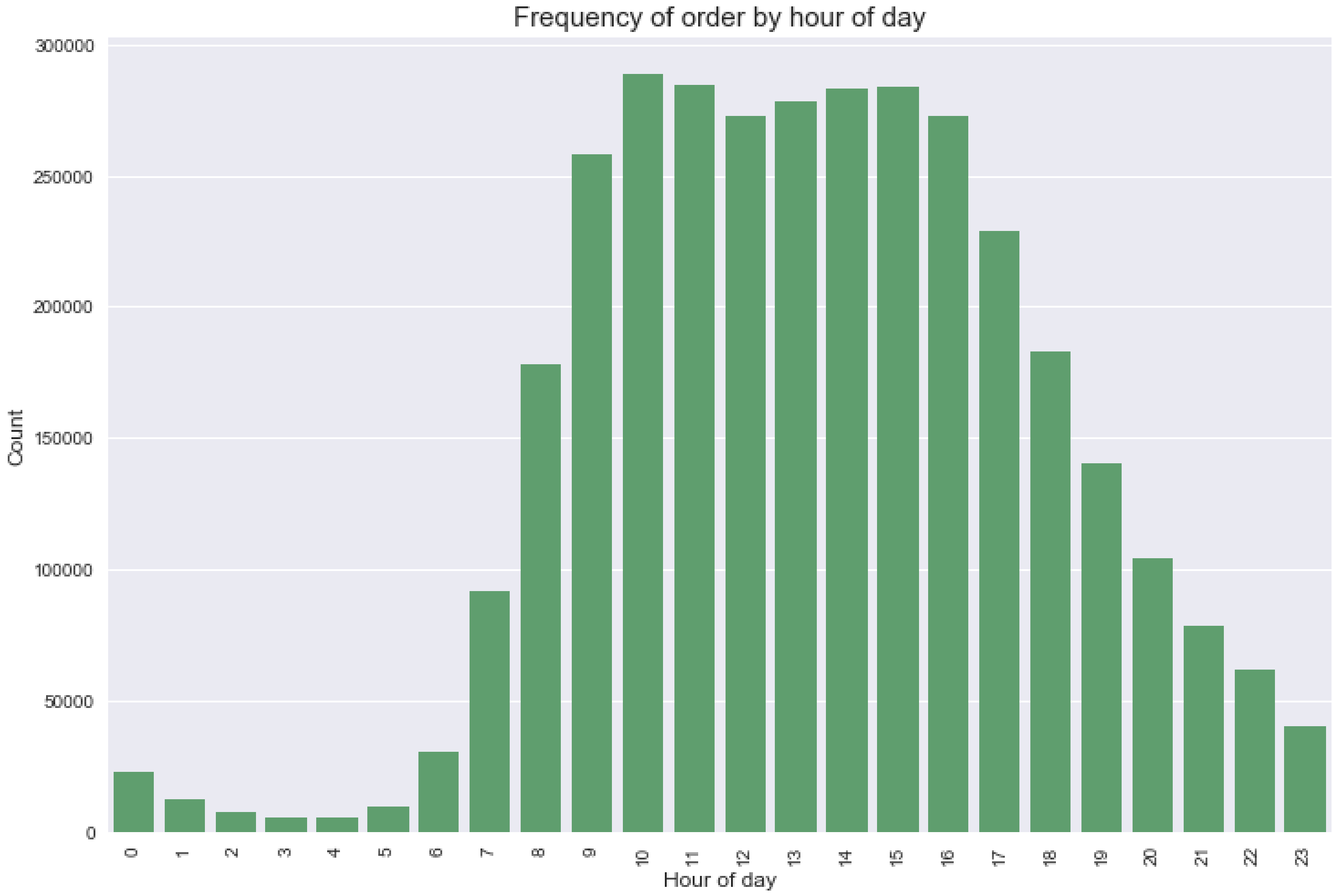
In [12]:

```
# plt.figure(figsize=(12,8))
sns.countplot(x="order_dow", data=orders_df, color=color[5])
plt.ylabel('Count', fontsize=12)
plt.xlabel('Day of week', fontsize=12)
plt.xticks(rotation='vertical')
plt.title("Frequency of order by week day", fontsize=15)
plt.show()
```



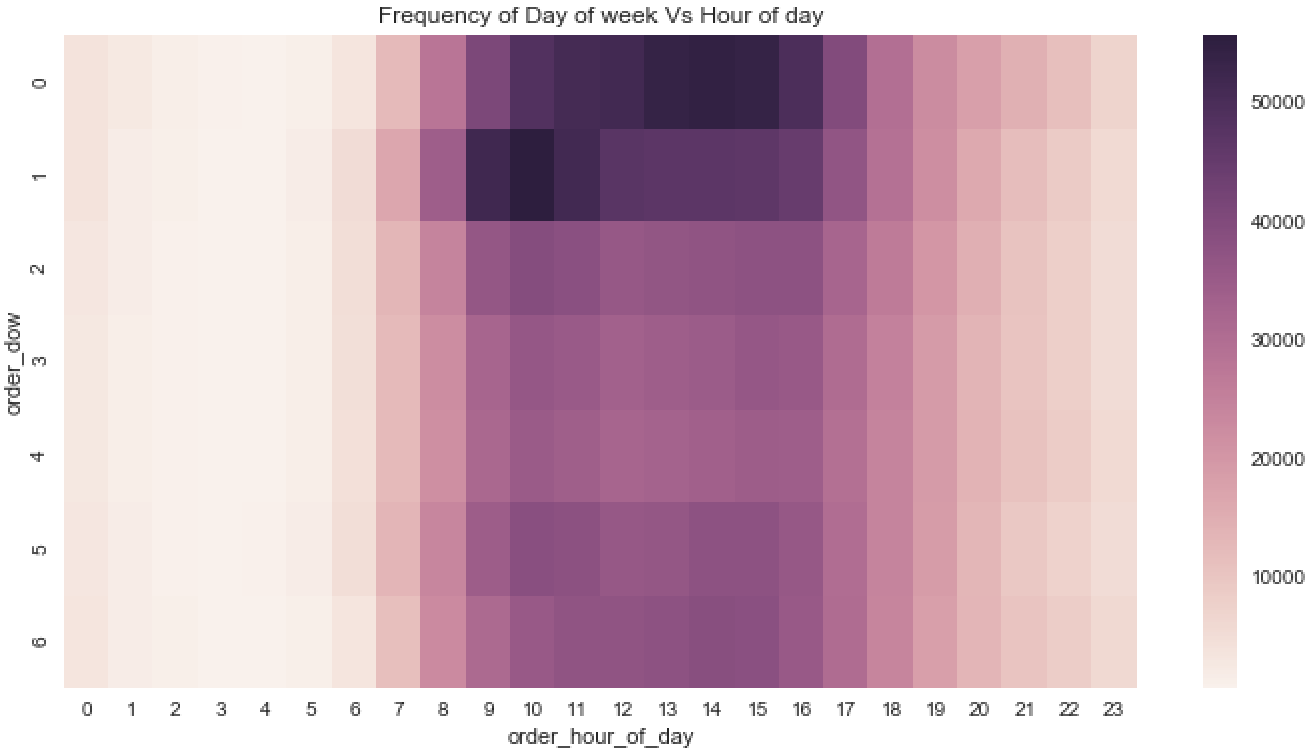
In [36]:

```
plt.figure(figsize=(12,8))
sns.countplot(x="order_hour_of_day", data=orders_df, color=color[1])
plt.ylabel('Count', fontsize=12)
plt.xlabel('Hour of day', fontsize=12)
plt.xticks(rotation='vertical')
plt.title("Frequency of order by hour of day", fontsize=15)
plt.show()
```



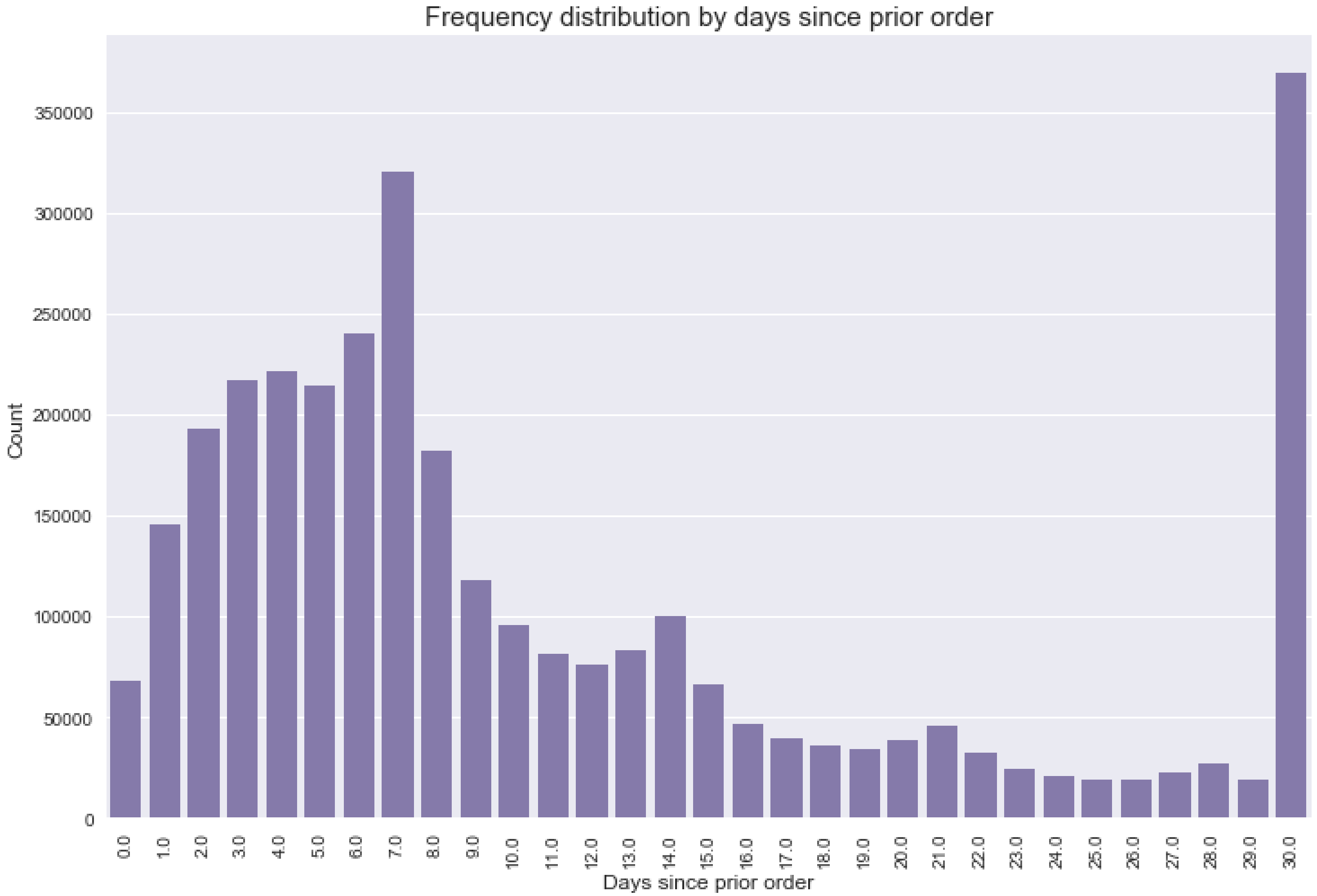
In [14]:

```
grouped_df = orders_df.groupby(["order_dow", "order_hour_of_day"])["order_number"].aggregate("count").reset_index()
grouped_df = grouped_df.pivot('order_dow', 'order_hour_of_day', 'order_number')

plt.figure(figsize=(12,6))
sns.heatmap(grouped_df)
plt.title("Frequency of Day of week Vs Hour of day")
plt.show()
```



In [15]:

```
plt.figure(figsize=(12,8))
sns.countplot(x="days_since_prior_order", data=orders_df, color=color[3])
plt.ylabel('Count', fontsize=12)
plt.xlabel('Days since prior order', fontsize=12)
plt.xticks(rotation='vertical')
plt.title("Frequency distribution by days since prior order", fontsize=15)
plt.show()
```



In [41]:

```
#percentage of ordered products in the prior set
order_products_prior_df.reordered.sum()/order_products_prior_df.shape[0]
```

Out[41]:

```
0.58969746679221613
```

In [ ]:

```
order_products_train_df.shape
```

In [44]:

```
# percentage of ordered products in the train set
order_products_train_df.reordered.sum()/order_products_train_df.shape[0]
```

Out[44]:

```
0.59859441275096292
```

In [17]:

```
grouped_df =order_products_prior_df.groupby("order_id")["reordered"].aggregate("sum").reset_index()
```

In [18]:

```
grouped_df.head(4)
```

Out[18]:

	order_id	reordered
0	2	6
1	3	8
2	4	12
3	5	21

In [19]:

```
grouped_df["reordered"].loc[grouped_df["reordered"]>1] = 1
grouped_df.reordered.value_counts() / grouped_df.shape[0]
```

Out[19]:

```
1    0.879151
0    0.120849
Name: reordered, dtype: float64
```

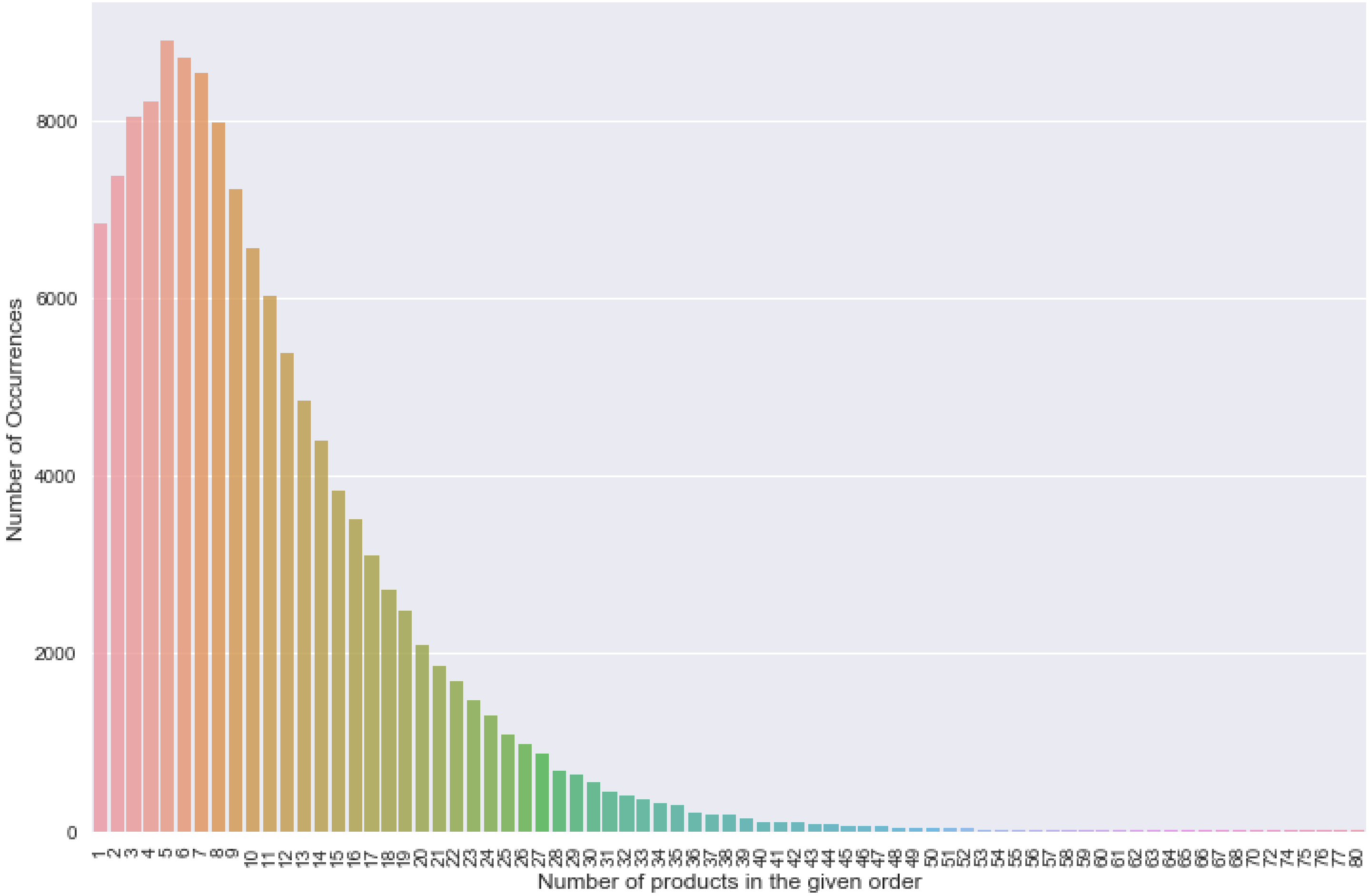
In [20]:

```
grouped_df = order_products_train_df.groupby("order_id")["reordered"].aggregate("sum").reset_index()
grouped_df["reordered"].loc[grouped_df["reordered"]>1] = 1
grouped_df.reordered.value_counts() / grouped_df.shape[0]
```

Out[20]:

```
1    0.93444
0    0.06556
Name: reordered, dtype: float64
```

In [21]:
```
grouped_df = order_products_train_df.groupby("order_id")["add_to_cart_order"].aggregate("max").reset_index()
cnt_srs = grouped_df.add_to_cart_order.value_counts()

plt.figure(figsize=(12,8))
sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8)
plt.ylabel('Number of Occurrences', fontsize=12)
plt.xlabel('Number of products in the given order', fontsize=12)
plt.xticks(rotation='vertical')
plt.show()
```



In [22]:
```
products_df.head(6)
```

Out[22]:

	product_id	product_name	aisle_id	department_id
0	1	Chocolate Sandwich Cookies	61	19
1	2	All-Seasons Salt	104	13
2	3	Robust Golden Unsweetened Oolong Tea	94	7
3	4	Smart Ones Classic Favorites Mini Rigatoni Wit...	38	1
4	5	Green Chile Anytime Sauce	5	13
5	6	Dry Nose Oil	11	11

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

Out[23]:

	aisle_id	aisle
0	1	prepared soups salads
1	2	specialty cheeses
2	3	energy granola bars
3	4	instant foods
4	5	marinades meat preparation

In [24]:

```
departments_df.head(5)
```

Out[24]:

	department_id	department
0	1	frozen
1	2	other
2	3	bakery
3	4	produce
4	5	alcohol

In [25]:

```
order_products_prior_df.head(2)
```

Out[25]:

	order_id	product_id	add_to_cart_order	reordered
0	2	33120	1	1
1	2	28985	2	1

In [26]:
```
order_products_prior_df = pd.merge(order_products_prior_df, products_df, on='product_id', how='left')
order_products_prior_df = pd.merge(order_products_prior_df, aisles_df, on='aisle_id', how='left')
order_products_prior_df = pd.merge(order_products_prior_df, departments_df, on='department_id', how='left')
order_products_prior_df.head()
```

Out[26]:

	order_id	product_id	add_to_cart_order	reordered	product_name	aisle_id	department_id	aisle	department
0	2	33120	1	1	Organic Egg Whites	86	16	eggs	dairy eggs
1	2	28985	2	1	Michigan Organic Kale	83	4	fresh vegetables	produce
2	2	9327	3	0	Garlic Powder	104	13	spices seasonings	pantry
3	2	45918	4	1	Coconut Butter	19	13	oils vinegars	pantry
4	2	30035	5	0	Natural Sweetener	17	13	baking ingredients	pantry

In [27]:
```
order_products_prior_df.describe()
```

Out[27]:

	order_id	product_id	add_to_cart_order	reordered	aisle_id	department_id
count	3.243449e+07	3.243449e+07	3.243449e+07	3.243449e+07	3.243449e+07	3.243449e+07
mean	1.710749e+06	2.557634e+04	8.351076e+00	5.896975e-01	7.121430e+01	9.921906e+00
std	9.873007e+05	1.409669e+04	7.126671e+00	4.918886e-01	3.820302e+01	6.281156e+00
min	2.000000e+00	1.000000e+00	1.000000e+00	0.000000e+00	1.000000e+00	1.000000e+00
25%	8.559430e+05	1.353000e+04	3.000000e+00	0.000000e+00	3.100000e+01	4.000000e+00
50%	1.711048e+06	2.525600e+04	6.000000e+00	1.000000e+00	8.300000e+01	9.000000e+00
75%	2.565514e+06	3.793500e+04	1.100000e+01	1.000000e+00	1.070000e+02	1.600000e+01
max	3.421083e+06	4.968800e+04	1.450000e+02	1.000000e+00	1.340000e+02	2.100000e+01

In [28]:

```
cnt_srs = order_products_prior_df['product_name'].value_counts().reset_index().head(20)
cnt_srs.columns = ['product_name', 'frequency_count']
cnt_srs
```

Out[28]:

	product_name	frequency_count
0	Banana	472565
1	Bag of Organic Bananas	379450
2	Organic Strawberries	264683
3	Organic Baby Spinach	241921
4	Organic Hass Avocado	213584
5	Organic Avocado	176815
6	Large Lemon	152657
7	Strawberries	142951
8	Limes	140627
9	Organic Whole Milk	137905
10	Organic Raspberries	137057
11	Organic Yellow Onion	113426
12	Organic Garlic	109778
13	Organic Zucchini	104823
14	Organic Blueberries	100060
15	Cucumber Kirby	97315
16	Organic Fuji Apple	89632
17	Organic Lemon	87746
18	Apple Honeycrisp Organic	85020
19	Organic Grape Tomatoes	84255

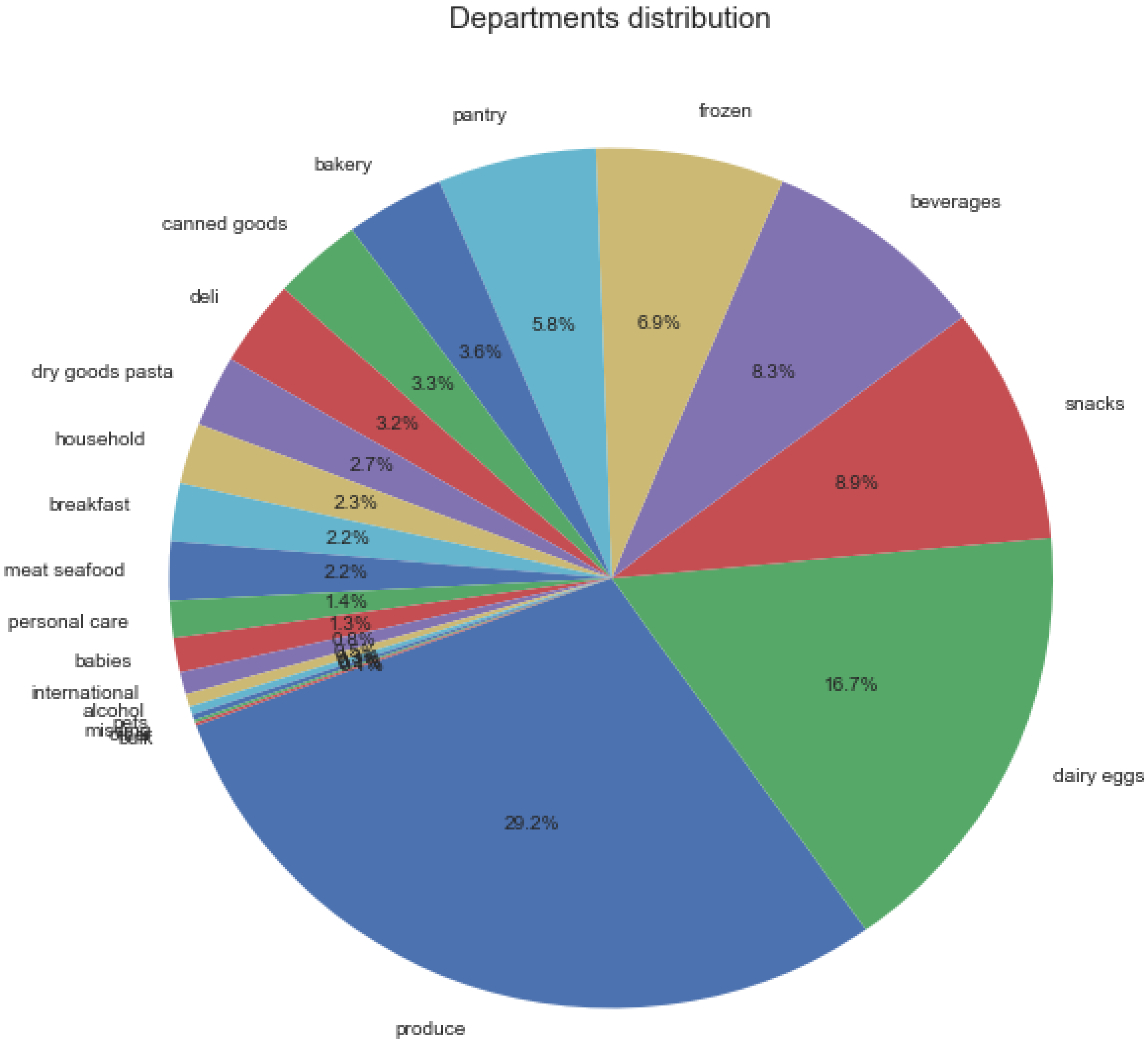
In [3]:

```
cnt_srs = order_products_prior_df['aisle'].value_counts().head(20)
plt.figure(figsize=(12,8))
sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8, color=color[5])
plt.ylabel('Number of Occurrences', fontsize=12)
plt.xlabel('Aisle', fontsize=12)
plt.xticks(rotation='vertical')
plt.show()
```

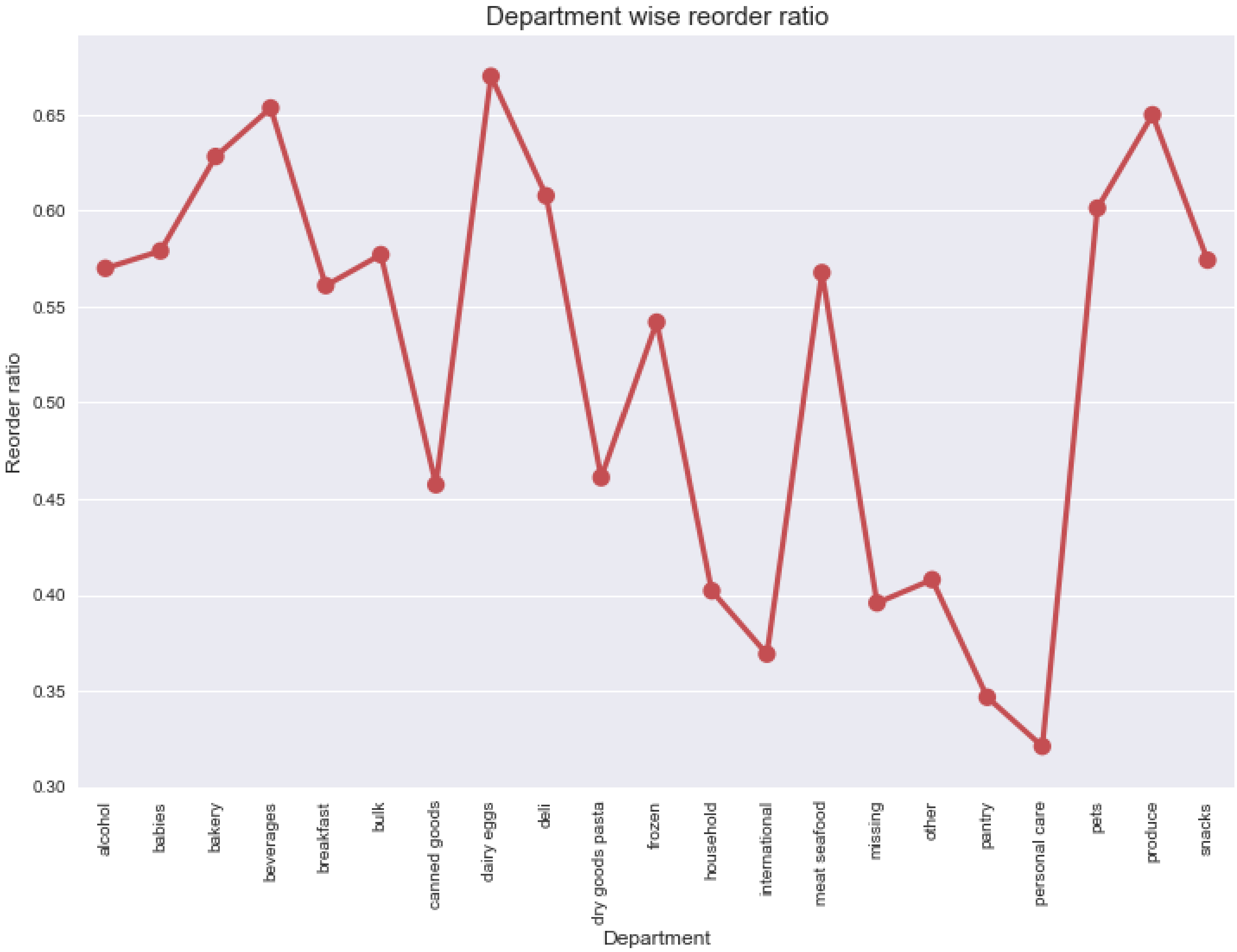
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-3-59a1127a5a4d> in <module>()
----> 1 cnt_srs = order_products_prior_df['aisle'].value_counts().head(20)
      2 plt.figure(figsize=(12,8))
      3 sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8, color=color[5])
      4 plt.ylabel('Number of Occurrences', fontsize=12)
      5 plt.xlabel('Aisle', fontsize=12)

NameError: name 'order_products_prior_df' is not defined
```

In [30]:

```
plt.figure(figsize=(10,10))
temp_series = order_products_prior_df['department'].value_counts()
labels = (np.array(temp_series.index))
sizes = (np.array((temp_series / temp_series.sum())*100))
plt.pie(sizes, labels=labels,
        autopct='%1.1f%%', startangle=200)
plt.title("Departments distribution", fontsize=15)
plt.show()
```

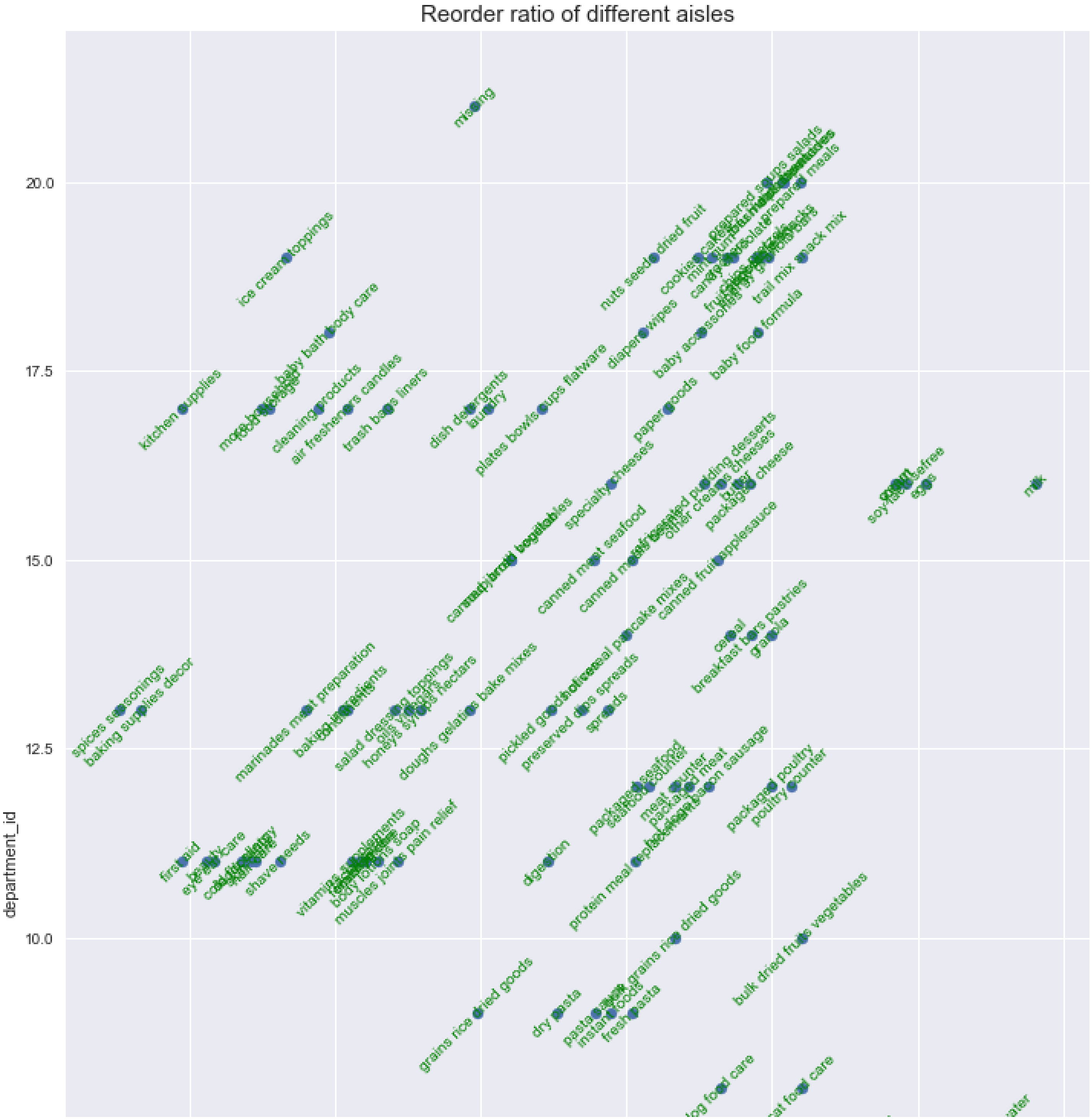

Departments distribution
pantry
5.8%
frozen
6.9%
bakery
3.6%
beverages
8.3%
canned goods
3.3%
deli
3.2%
snacks
8.9%
dry goods pasta
2.7%
household
2.3%
breakfast
2.2%
meat seafood
2.2%
personal care
1.4%
1.3%
babies
0.8%
international
alcohol
pets
missing
other
bulk
dairy eggs
16.7%
29.2%
produce

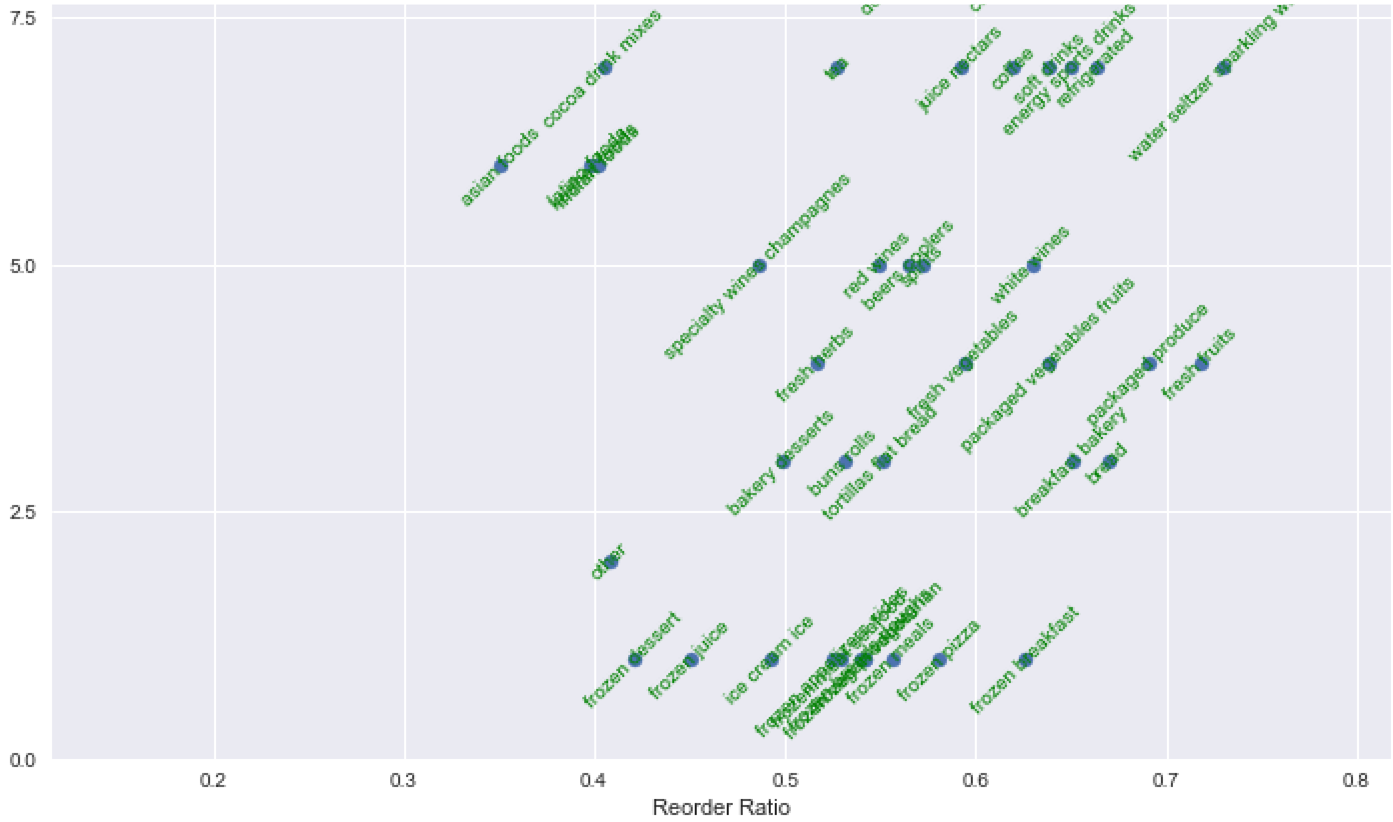
In [32]:
```
grouped_df = order_products_prior_df.groupby(["department"])["reordered"].aggregate("mean").reset_index()
plt.figure(figsize=(12,8))
sns.pointplot(grouped_df['department'].values, grouped_df['reordered'].values, alpha=0.8, color=color[2])
plt.ylabel('Reorder ratio', fontsize=12)
plt.xlabel('Department', fontsize=12)
plt.title("Department wise reorder ratio", fontsize=15)
plt.xticks(rotation='vertical')
plt.show()
```



In [33]:

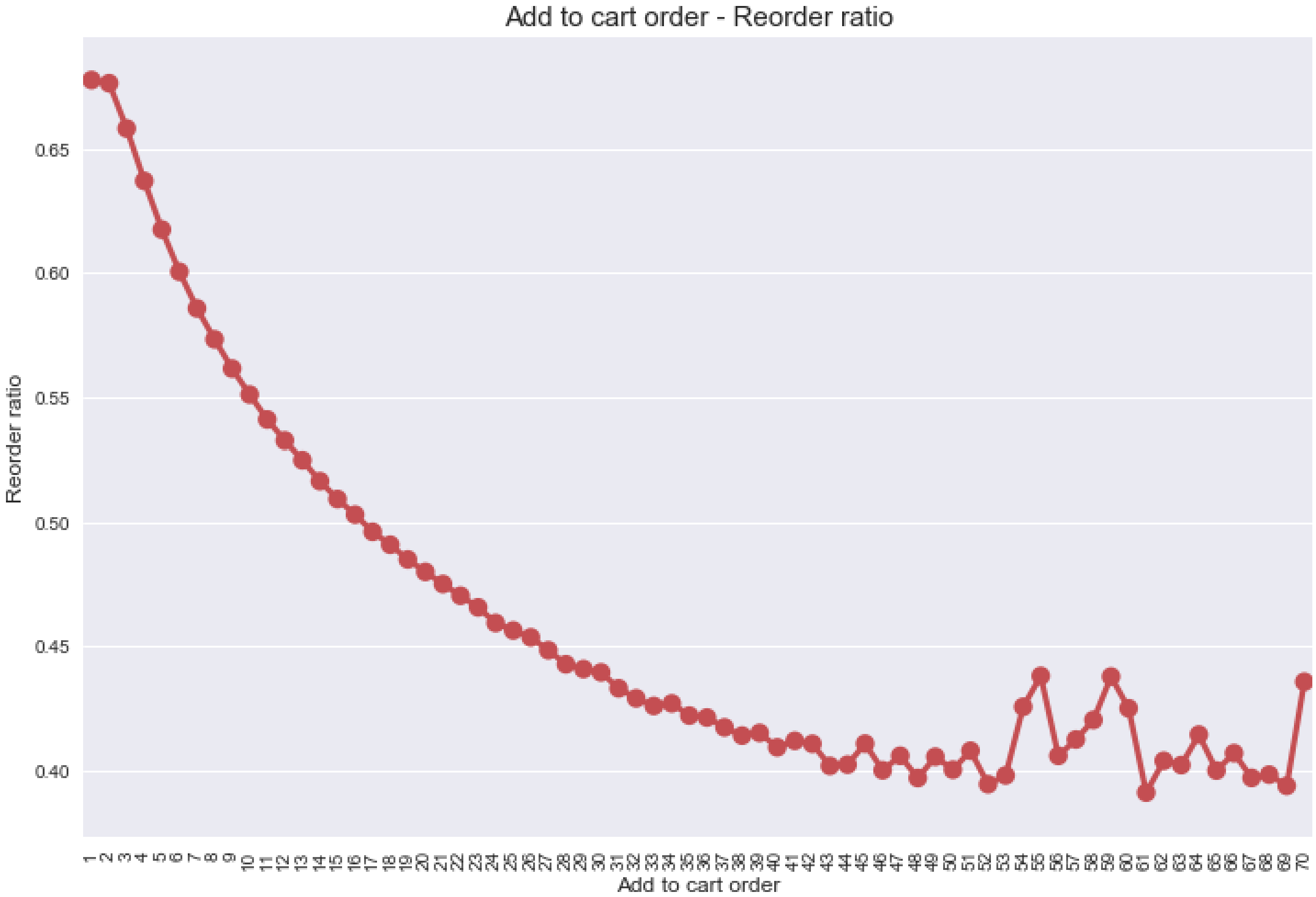
```
grouped_df = order_products_prior_df.groupby(["department_id", "aisle"])["reordered"].aggregate("mean").reset_index()
fig, ax = plt.subplots(figsize=(12,20))
ax.scatter(grouped_df.reordered.values, grouped_df.department_id.values)
for i, txt in enumerate(grouped_df.aisle.values):
ax.annotate(txt, (grouped_df.reordered.values[i], grouped_df.department_id.values[i]), rotation=45, ha='center', va='center', color='green')
plt.xlabel('Reorder Ratio')
plt.ylabel('department_id')
plt.title("Reorder ratio of different aisles", fontsize=15)
plt.show()
```


Reorder ratio of different aisles
department_id
20.0
17.5
15.0
12.5
10.0
missing
ice cream toppings
baby bath body care
kitchen supplies
cleaning products
air fresheners candles
trash bags liners
dish detergents
laundry
plates bowls cups flatware
nuts seeds dried fruit
prepared soups salads
meals
trail mix snack mix
diapers wipes
baby accessories
baby food formula
paper goods
specialty cheeses
refrigerated pudding desserts
other creams cheeses
butter
packaged cheese
canned meat seafood
canned fruit applesauce
pancake mixes
cereal
breakfast bars pastries
granola
soy lactosefree
eggs
milk
spices seasonings
baking supplies decor
marinades meat preparation
baking ingredients
salad dressing toppings
honeys syrups nectars
doughs gelatins bake mixes
pickled goods olives
preserved dips spreads
spreads
packaged seafood
seafood counter
meat counter
packaged meat
hot dogs bacon sausage
packaged poultry
poultry counter
protein meal replacements
first aid
beauty
eye ear care
shave needs
vitamins supplements
body lotions soap
muscles joints pain relief
digestion
grains rice dried goods
dry pasta
pasta sauce
instant foods
fresh pasta
bulk grains rice dried goods
bulk dried fruits vegetables
dog food care
cat food care
water

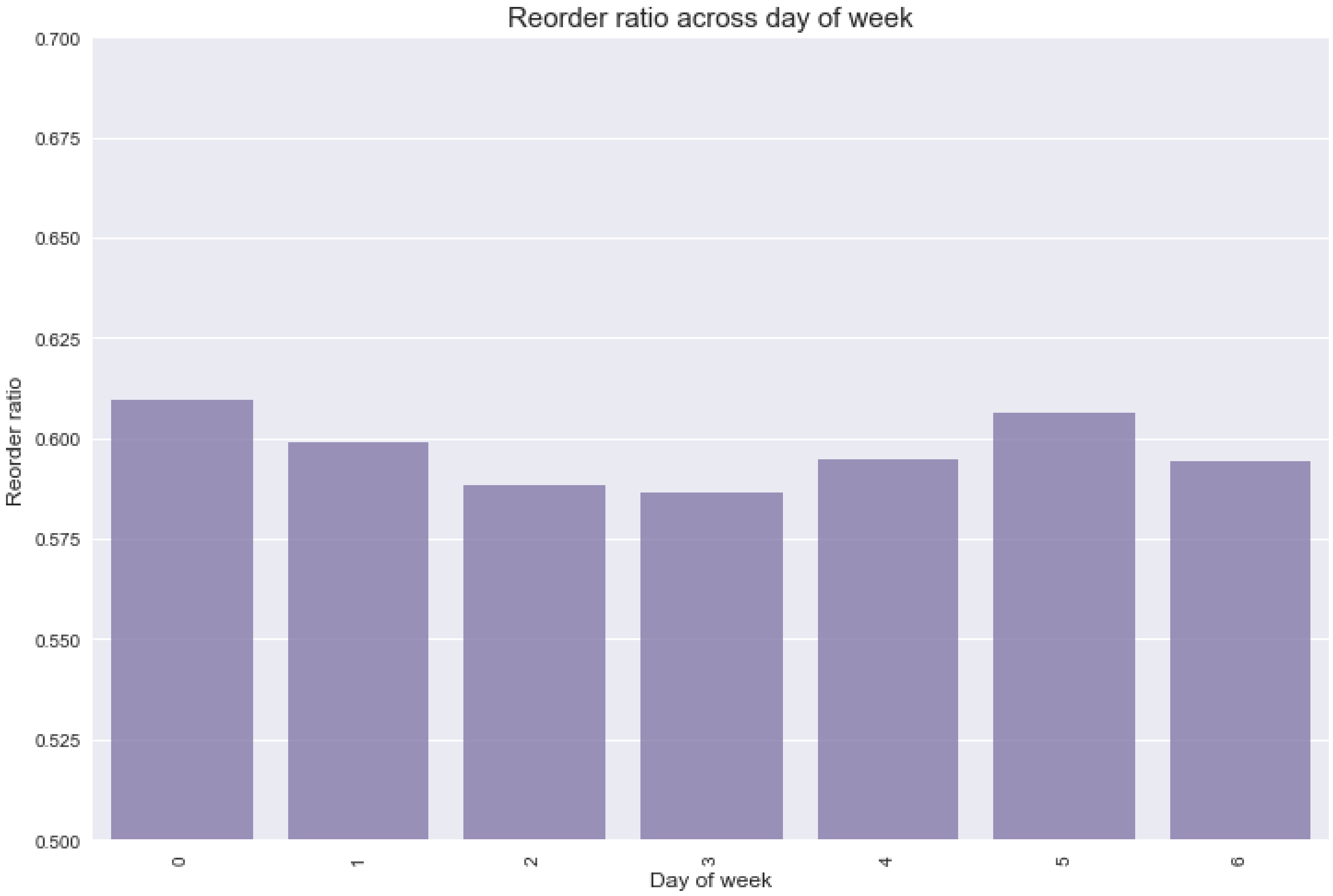

7.5
5.0
2.5
0.0
0.2
0.3
0.4
0.5
0.6
0.7
0.8
Reorder Ratio
asian foods
cocoa drink mixes
tea
juice nectars
coffee
soft drinks
energy sports drinks
refrigerated
water seltzer sparkling w
specialty wines champagnes
red wines
beers coolers
spirits
white wines
fresh herbs
fresh vegetables
packaged vegetables fruits
packaged produce
fresh fruits
bakery desserts
buns rolls
tortillas flat bread
breakfast bakery
bread
other
frozen dessert
frozen juice
ice cream ice
frozen meals
frozen pizza
frozen breakfast

In [35]:

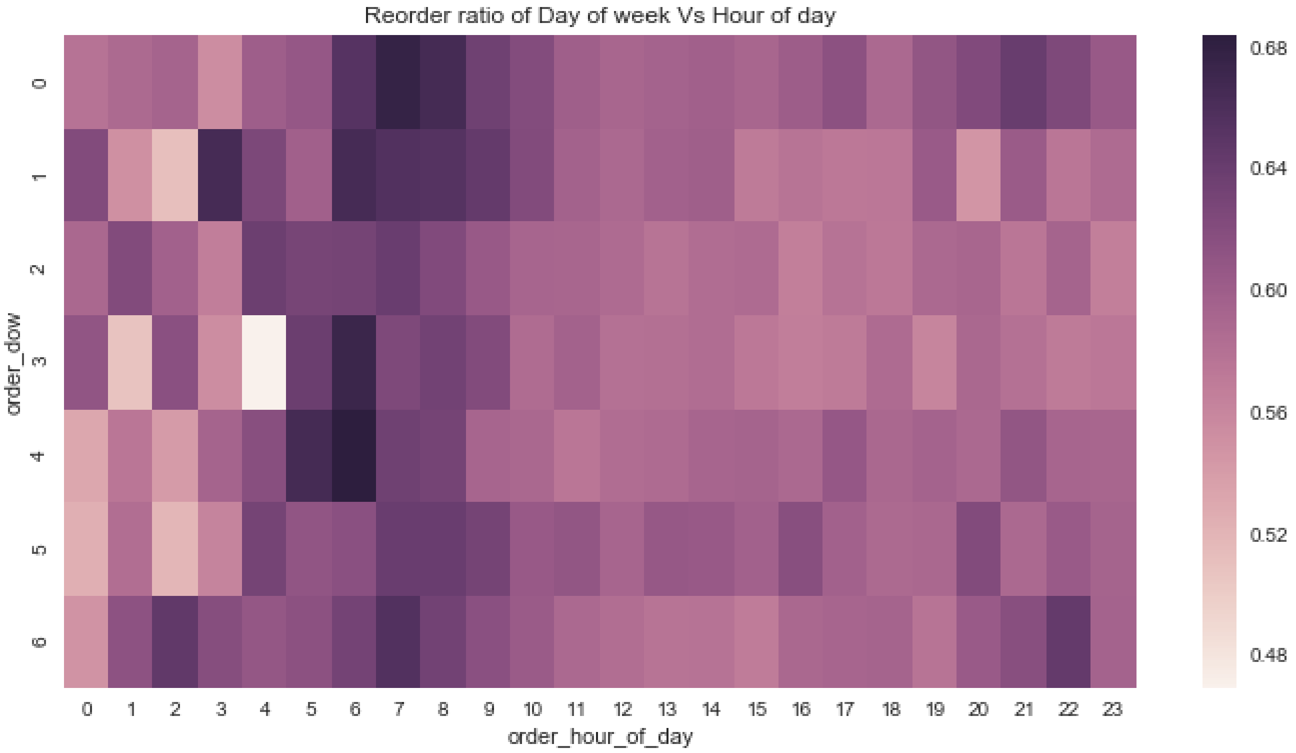
```
order_products_prior_df["add_to_cart_order_mod"] = order_products_prior_df["add_to_cart_order"].copy()
order_products_prior_df["add_to_cart_order_mod"].loc[order_products_prior_df["add_to_cart_order_mod"]>70] = 70
grouped_df = order_products_prior_df.groupby(["add_to_cart_order_mod"])["reordered"].aggregate("mean").reset_index()
plt.figure(figsize=(12,8))
sns.pointplot(grouped_df['add_to_cart_order_mod'].values, grouped_df['reordered'].values, alpha=0.8, color=color[2])
plt.ylabel('Reorder ratio', fontsize=12)
plt.xlabel('Add to cart order', fontsize=12)
plt.title("Add to cart order - Reorder ratio", fontsize=15)
plt.xticks(rotation='vertical')
plt.show()
```


Add to cart order - Reorder ratio
Reorder ratio
0.65
0.60
0.55
0.50
0.45
0.40
Add to cart order
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70

```
In [36]: order_products_train_df = pd.merge(order_products_train_df, orders_df, on='order_id', how='left')
         grouped_df = order_products_train_df.groupby(["order_dow"])["reordered"].aggregate("mean").reset_index()

         plt.figure(figsize=(12,8))
         sns.barplot(grouped_df['order_dow'].values, grouped_df['reordered'].values, alpha=0.8, color=color[3])
         plt.ylabel('Reorder ratio', fontsize=12)
         plt.xlabel('Day of week', fontsize=12)
         plt.title("Reorder ratio across day of week", fontsize=15)
         plt.xticks(rotation='vertical')
         plt.ylim(0.5, 0.7)
         plt.show()
```



In [38]:
```
grouped_df = order_products_train_df.groupby(["order_dow", "order_hour_of_day"])["reordered"].aggregate("mea
n").reset_index()
grouped_df = grouped_df.pivot('order_dow', 'order_hour_of_day', 'reordered')

plt.figure(figsize=(12,6))
sns.heatmap(grouped_df)
plt.title("Reorder ratio of Day of week Vs Hour of day")
plt.show()
```



In [1]:

In [ ]:

In [ ]: