EXPLORATORY DATA ANALYSIS ON GROCERY DATASET

WHAT IS EDA:

```
 Exploratory Data Analysis is a data analytic process that aims to understand the data in
depth and learn its different characterstics.
 EDA refers to the critical process of performing initial investigation on data to discover
patterns, to spot anomalies to test hypothesis
 and to check assumptions with the help of summary ststistics and graphical representations.
```

STEPS INVOLVED IN EXPLORATORY DATA ANALYSIS:

```
1.Understand the data :  understanding the variables in the dataset,and on what kind of data
you are working with.
2.Clean the data      : cleaning data from redundancy,irregularity and deleting unnecessary
columns,outliers which causes noise in the data.
3.Analysis of Relationship between variables : analysing the relationship between the
variables in the dataset.
4.Data visualisation  : visualizing the relationship in different patterns to understand
easily.
```

In [17]:
```
from IPython.display import Image
Image("groceries..jpeg")
```

Out[17]:

Grocery dataset:

This dataset tells about the Grocery purchased and this data is taken from kaggle website. we will try to understand the dataset by using pandas ,numpy for data storing and processing and for visualization we use matplotlib and seaborn The data contains 7 columns.

1. Member_number: unique number id.
2. Date: Date of transaction.
3. itemDescription: name of item.
4. year:year of transaction.
5. month:month of transaction.
6. day:day of transaction.
7. day_of_week:day of transaction in week.

we are importing pandas,numpy,seaborn,matplotlib and warnings to ignore warnings.This dataset tells about the grocery details and this data is taken feom kaggle website and we will try to understand the dataset by using pandas,numpy for data storing and processing and for visualisation we use matplotlib and seaborn.

In [21]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
import matplotlib.pyplot as plt
%matplotlib inline
```

Here we are Reading the groceries.csv file with the help of pandas

In [23]:
```
groceries_df=pd.read_csv('groceries.csv')
```

1.Understanding the data

we use the head() command to retrieve the first 5 rows of our groceries data

In [26]:
```
groceries_df.head()
```

Out[26]:

	Member_number	Date	itemDescription	year	month	day	day_of_week
0	1808	21-07-2015	tropical fruit	2015	7	21	1
1	2552	01-05-2015	whole milk	2015	5	1	4
2	2300	19-09-2015	pip fruit	2015	9	19	5
3	1187	12-12-2015	other vegetables	2015	12	12	5
4	3037	02-01-2015	whole milk	2015	1	2	4

we use the tail() command to retrieve the last 5 rows of our groceries data

In [28]:
```
groceries_df.tail()
```

Out[28]:

	Member_number	Date	itemDescription	year	month	day	day_of_week
38760	4471	10-08-2014	sliced cheese	2014	8	10	6
38761	2022	23-02-2014	candy	2014	2	23	6
38762	1097	16-04-2014	cake bar	2014	4	16	2
38763	1510	12-03-2014	fruit/vegetable juice	2014	3	12	2
38764	1521	26-12-2014	cat food	2014	12	26	4

This info() command is used to retrieve the information i.e., whether the data has null values or not and datatype of each columns

In [30]:
```
groceries_df.info();
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 38765 entries, 0 to 38764
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   Member_number    38765 non-null  int64 
 1   Date             38765 non-null  object
 2   itemDescription  38765 non-null  object
 3   year             38765 non-null  int64 
 4   month            38765 non-null  int64 
 5   day              38765 non-null  int64 
 6   day_of_week      38765 non-null  int64 
dtypes: int64(5), object(2)
memory usage: 2.1+ MB
```

This describe() command is used to display statistics values of our data(i.e.,mean,standard deviation,min,max) but this doesnot shows the statistics of objects in our data

In [32]:
```
groceries_df.describe()
```

Out[32]:

	Member_number	year	month	day	day_of_week
count	38765.000000	38765.000000	38765.000000	38765.000000	38765.000000
mean	3003.641868	2014.528518	6.477570	15.753231	3.014498
std	1153.611031	0.499193	3.431561	8.801391	1.987669
min	1000.000000	2014.000000	1.000000	1.000000	0.000000
25%	2002.000000	2014.000000	4.000000	8.000000	1.000000
50%	3005.000000	2015.000000	6.000000	16.000000	3.000000
75%	4007.000000	2015.000000	9.000000	23.000000	5.000000
max	5000.000000	2015.000000	12.000000	31.000000	6.000000

This describe(include='object') command used to display the count,unique values,top ,freq of

each object in our grocery data

In [34]:
```
groceries_df.describe(include='object')
```

Out[34]:

	Date	itemDescription
count	38765	38765
unique	728	167
top	21-01-2015	whole milk
freq	96	2502

This shape command is used to display the number of rows and columns of our data

In [36]:
```
groceries_df.shape
```

Out[36]:
```
(38765, 7)
```

columns command shows the column names

In [38]:
```
groceries_df.columns
```

Out[38]:
```
Index(['Member_number', 'Date', 'itemDescription', 'year', 'month', 'day',
       'day_of_week'],
      dtype='object')
```

nunique() command displays the unique values in each column.

In [40]:
```
groceries_df.nunique()
```

Out[40]:
```
Member_number      3898
Date                728
itemDescription     167
year                  2
month                12
day                  31
day_of_week           7
dtype: int64
```

It defines the unique values of particular column.

In [42]:
```
groceries_df['itemDescription'].unique()
```

Out[42]:
```
array(['tropical fruit', 'whole milk', 'pip fruit', 'other vegetables',
       'rolls/buns', 'pot plants', 'citrus fruit', 'beef', 'frankfurter',
       'chicken', 'butter', 'fruit/vegetable juice',
       'packaged fruit/vegetables', 'chocolate', 'specialty bar',
       'butter milk', 'bottled water', 'yogurt', 'sausage', 'brown bread',
       'hamburger meat', 'root vegetables', 'pork', 'pastry',
       'canned beer', 'berries', 'coffee', 'misc. beverages', 'ham',
       'turkey', 'curd cheese', 'red/blush wine',
       'frozen potato products', 'flour', 'sugar', 'frozen meals',
       'herbs', 'soda', 'detergent', 'grapes', 'processed cheese', 'fish',
       'sparkling wine', 'newspapers', 'curd', 'pasta', 'popcorn',
       'finished products', 'beverages', 'bottled beer', 'dessert',
       'dog food', 'specialty chocolate', 'condensed milk', 'cleaner',
       'white wine', 'meat', 'ice cream', 'hard cheese', 'cream cheese ',
       'liquor', 'pickled vegetables', 'liquor (appetizer)', 'UHT-milk',
       'candy', 'onions', 'hair spray', 'photo/film', 'domestic eggs',
       'margarine', 'shopping bags', 'salt', 'oil', 'whipped/sour cream',
       'frozen vegetables', 'sliced cheese', 'dish cleaner',
       'baking powder', 'specialty cheese', 'salty snack',
       'Instant food products', 'pet care', 'white bread',
       'female sanitary products', 'cling film/bags', 'soap',
       'frozen chicken', 'house keeping products', 'spread cheese',
       'decalcifier', 'frozen dessert', 'vinegar', 'nuts/prunes',
       'potato products', 'frozen fish', 'hygiene articles',
       'artif. sweetener', 'light bulbs', 'canned vegetables',
       'chewing gum', 'canned fish', 'cookware', 'semi-finished bread',
       'cat food', 'bathroom cleaner', 'prosecco', 'liver loaf',
       'zwieback', 'canned fruit', 'frozen fruits', 'brandy',
       'baby cosmetics', 'spices', 'napkins', 'waffles', 'sauces', 'rum',
       'chocolate marshmallow', 'long life bakery product', 'bags',
       'sweet spreads', 'soups', 'mustard', 'specialty fat',
       'instant coffee', 'snack products', 'organic sausage',
       'soft cheese', 'mayonnaise', 'dental care', 'roll products ',
       'kitchen towels', 'flower soil/fertilizer', 'cereals',
       'meat spreads', 'dishes', 'male cosmetics', 'candles', 'whisky',
       'tidbits', 'cooking chocolate', 'seasonal products', 'liqueur',
       'abrasive cleaner', 'syrup', 'ketchup', 'cream', 'skin care',
       'rubbing alcohol', 'nut snack', 'cocoa drinks', 'softener',
       'organic products', 'cake bar', 'honey', 'jam', 'kitchen utensil',
       'flower (seeds)', 'rice', 'tea', 'salad dressing',
       'specialty vegetables', 'pudding powder', 'ready soups',
       'make up remover', 'toilet cleaner', 'preservation products'],
      dtype=object)
```

Till now we understood the data next step is to clean the unnecessary data,outliers,and removing null values.

2.Cleaning the data

By using isnull() we can identify null values and by using sum() we can find sum.

In [46]:
```
groceries_df.isnull().sum()
```

Out[46]:
```
Member_number      0
Date               0
itemDescription    0
year               0
month              0
day                0
day_of_week        0
dtype: int64
```

groceries_df.loc[groceries_df.itemDescription=='whole milk'] by using this we can access the total record of the wholewheat

In [48]:
```
itemdescripition_df=groceries_df.loc[groceries_df.itemDescription=='whole milk']
itemdescripition_df
```

Out[48]:

	Member_number	Date	itemDescription	year	month	day	day_of_week
1	2552	01-05-2015	whole milk	2015	5	1	4
4	3037	02-01-2015	whole milk	2015	1	2	4
8	2762	20-03-2015	whole milk	2015	3	20	4
21	2867	11-12-2015	whole milk	2015	12	11	4
53	1061	09-05-2015	whole milk	2015	5	9	5
...	...	...	...	...	...	...	...
38667	3667	05-11-2014	whole milk	2014	11	5	2
38672	4211	04-03-2014	whole milk	2014	3	4	1
38688	2049	04-02-2014	whole milk	2014	2	4	1
38689	4855	16-06-2014	whole milk	2014	6	16	0
38745	3082	22-07-2014	whole milk	2014	7	22	1

2502 rows × 7 columns

Here date is unnecessary because day,month,year is available so date column is dropped.

In [54]:

```
grocery=groceries_df.drop(['Date'],axis=1)
```

we are displaying the grocery data to check whether the date is dropped or not by head()

In [59]:

```
grocery.head()
```

Out[59]:

	Member_number	itemDescription	year	month	day	day_of_week
0	1808	tropical fruit	2015	7	21	1
1	2552	whole milk	2015	5	1	4
2	2300	pip fruit	2015	9	19	5
3	1187	other vegetables	2015	12	12	5
4	3037	whole milk	2015	1	2	4

Since they are no null values and outliers we skip this steps.

now,we are defining the relationship between variables.

3.Relationship analysis

we are defining a variable num_df which has date related to year,month,day,day_of_week to find correlation

In [69]:

```
num_df=groceries_df[['year','month','day','day_of_week']]
num_df
```

Out[69]:

	year	month	day	day_of_week
0	2015	7	21	1
1	2015	5	1	4
2	2015	9	19	5
3	2015	12	12	5
4	2015	1	2	4
...	...	...	...	...
38760	2014	8	10	6
38761	2014	2	23	6
38762	2014	4	16	2
38763	2014	3	12	2
38764	2014	12	26	4

38765 rows × 4 columns

corelation is used to find the relationship between two columns.

we have created a variable corelation to find correlation of num_df by corr() command

In [73]:
```
corelation=num_df.corr()
corelation
```

Out[73]:

	year	month	day	day_of_week
year	1.000000	-0.000248	0.004209	0.000415
month	-0.000248	1.000000	0.006896	-0.012917
day	0.004209	0.006896	1.000000	-0.004957
day_of_week	0.000415	-0.012917	-0.004957	1.000000

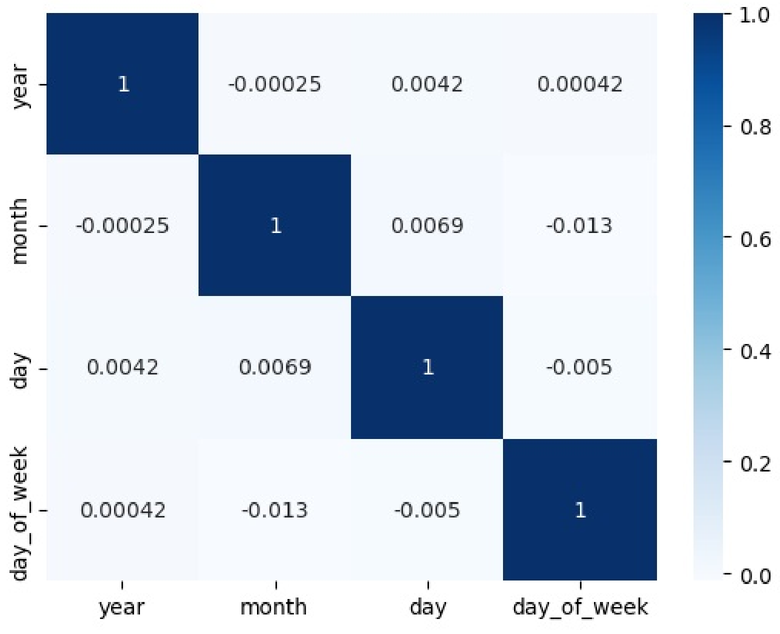
sns is a shorthand of seaborn.

Here we are plotting heatmap.

Heatmap is defined as a graphical representation of data using colors to visualize the value of the matrix.

In [75]:
```
sns.heatmap(corelation,xticklabels=corelation.columns,yticklabels=corelation.columns,
            annot=True,cmap='Blues')
```

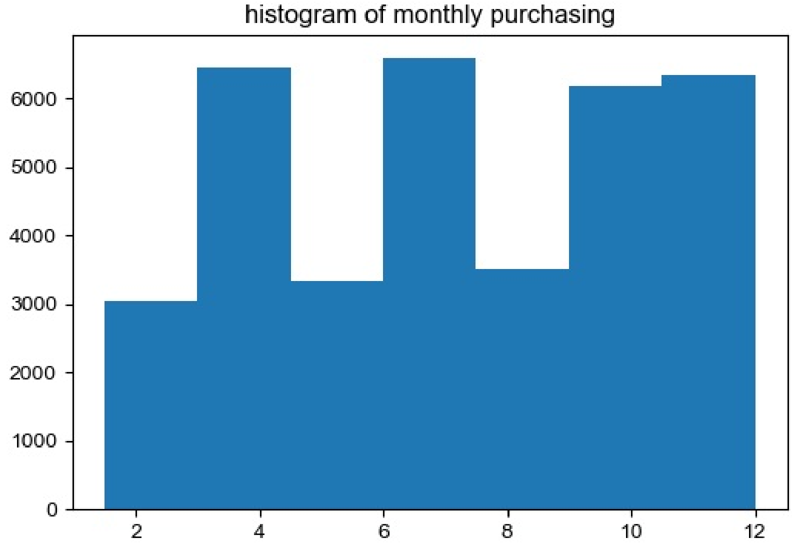
Out[75]: <Axes: >



4.Data visualization

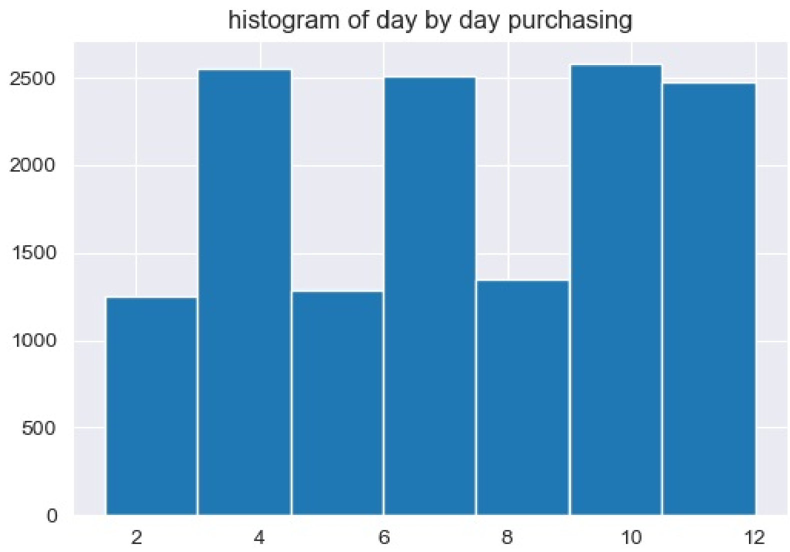
This is the histogram graph that shows the data related to which month is having high demand.

In [80]:
```
plt.figure(figsize=(6,4))
plt.hist(x=groceries_df.month,bins=[1.5,3.0,4.5,6.0,7.5,9.0,10.5,12.0]);
sns.set_style("darkgrid")
plt.title("histogram of monthly purchasing");
```



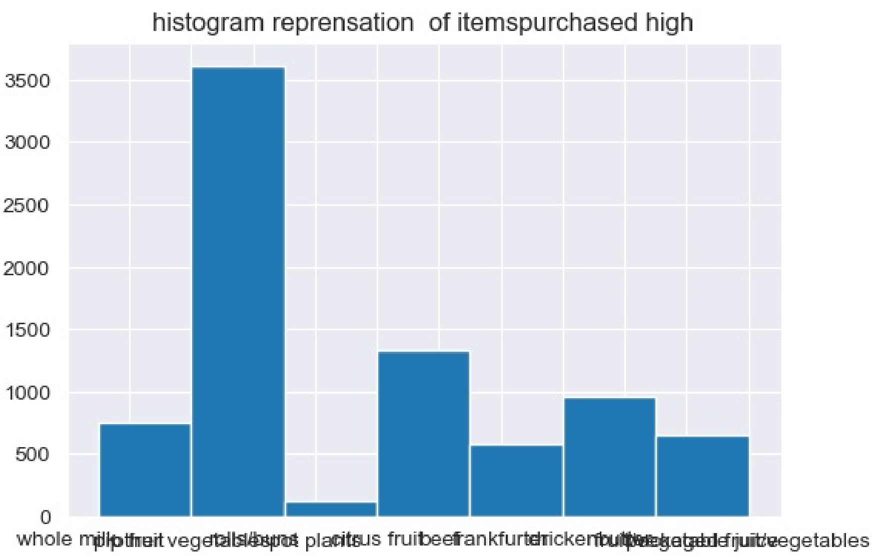
This is the histogram graph that shows the data related to which day is having high demand.

In [82]:
```
plt.figure(figsize=(6,4))
plt.hist(groceries_df.day,bins=[1.5,3.0,4.5,6.0,7.5,9.0,10.5,12.0]);
plt.title("histogram of day by day purchasing");
```



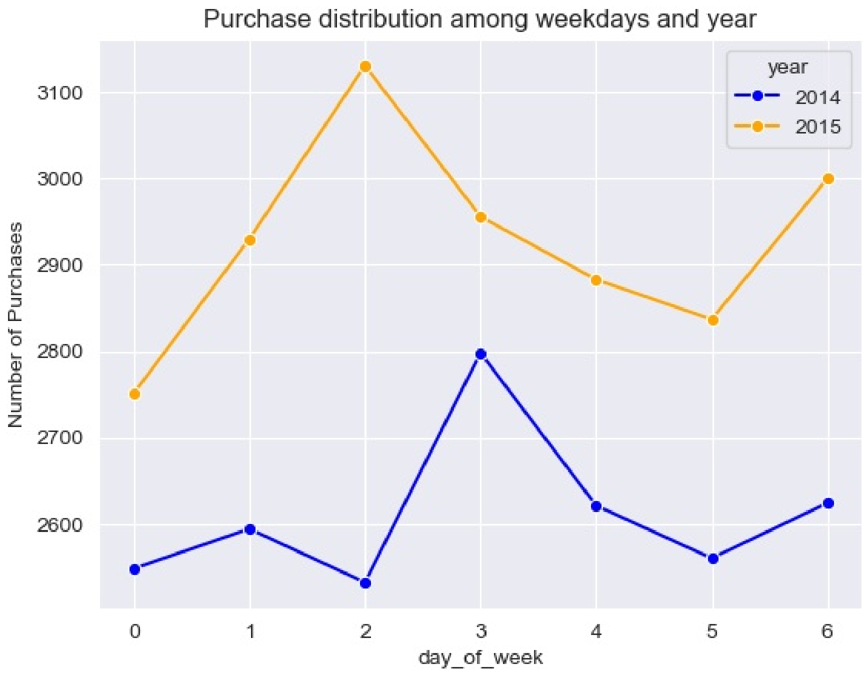
This is the histogram graph that shows the data related to which item is having high demand.

In [85]:
```
plt.figure(figsize=(6,4))
plt.hist(groceries_df.itemDescription,bins=[1.5,3.0,4.5,6.0,7.5,9.0,10.5,12.0]);
plt.title("histogram reprensation  of itemspurchased high");
```



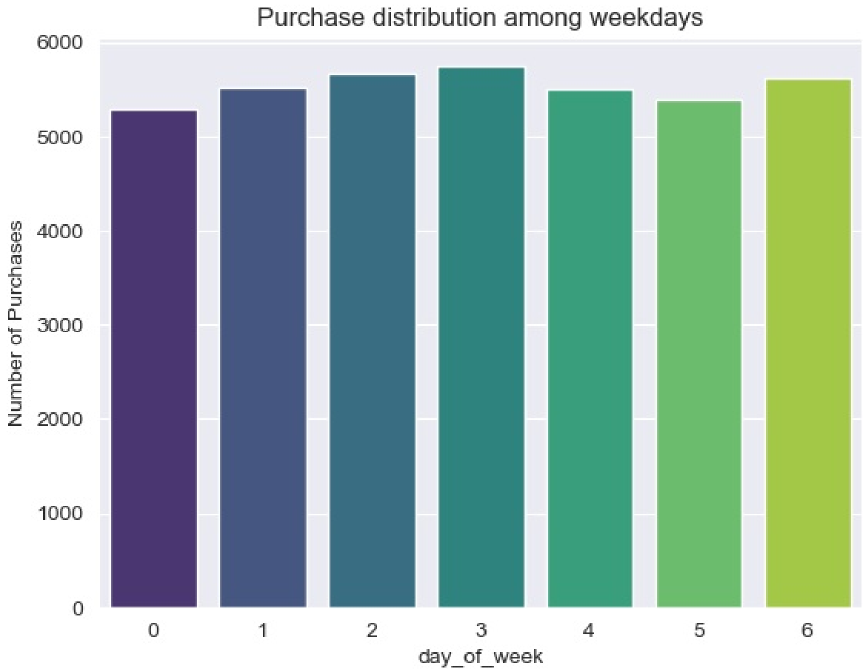
This is the lineplot of purchase distribution among weekdays and year

In [87]:
```
weekdays_year_dist = groceries_df.groupby(['year','day_of_week'],as_index =False).size()

sns.lineplot(data= weekdays_year_dist,x='day_of_week',y='size',hue='year',palette = ['blue','orange'], marker='
plt.ylabel('Number of Purchases')
plt.title('Purchase distribution among weekdays and year')
plt.show()
```



This is the bargraph related to purchase distribution among weekdays and xaxis we have taken the no.of purchases and on yaxis we have taken purchase distribution among weekdays

In [90]:
```
weekdays_dist = groceries_df.groupby(['day_of_week'],as_index =False).size()

sns.barplot(data= weekdays_dist,x='day_of_week',y='size',palette = 'viridis')
plt.ylabel('Number of Purchases')
plt.title('Purchase distribution among weekdays')
plt.show()
```



CONCLUSION:

The exploratory data analysis on the grocery dataset provided valuable insights in ,which item has high demand and on which month and on which day and day_of_week

HAPPY LEARNING