In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import copy
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [2]:

```
df=pd.read_excel("AR OUTPUT.xlsx")
```

In [3]:

```
df.head()
```

Out[3]:

	Support	Confidence	Lift	Consequent	implies	Items
0	0.020193	0.851852	2.349296	paper towels	<---	[eggs, dinner rolls, ice cream, pasta, lunch m...
1	0.020193	0.851852	2.266961	mixes	<---	[yogurt, dishwashing liquid/detergent, all- pu...
2	0.020193	0.821429	2.265393	paper towels	<---	[eggs, dinner rolls, poultry, ice cream, pasta]
3	0.022827	0.838710	2.258370	ketchup	<---	[tortillas, coffee/tea, juice, soap]
4	0.021949	0.833333	2.243893	pasta	<---	[paper towels, dishwashing liquid/detergent, e...

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 39 entries, 0 to 38
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Support     39 non-null     float64
 1   Confidence  39 non-null     float64
 2   Lift        39 non-null     float64
 3   Consequent  39 non-null     object 
 4   implies     39 non-null     object 
 5   Items       39 non-null     object 
dtypes: float64(3), object(3)
memory usage: 2.0+ KB
```

In [5]:

```
df.shape
```

Out[5]:

```
(39, 6)
```

In [6]:

```
df.describe().T
```

Out[6]:

	count	mean	std	min	25%	50%	75%	max
Support	39.0	0.022962	0.002463	0.020193	0.021071	0.022827	0.024583	0.028973
Confidence	39.0	0.825376	0.018606	0.800000	0.812500	0.821429	0.833333	0.884615

	count	mean	std	min	25%	50%	75%	max
Lift	39.0	2.101930	0.118810	1.898333	2.002049	2.132959	2.189456	2.349296

In [7]:

```
cat_df = df.select_dtypes(include=['object']).copy()
```

In [8]:

```
cat_df.head()
```

Out[8]:

	Consequent	implies	Items
0	paper towels	<---	[eggs, dinner rolls, ice cream, pasta, lunch m...
1	mixes	<---	[yogurt, dishwashing liquid/detergent, all- pu...
2	paper towels	<---	[eggs, dinner rolls, poultry, ice cream, pasta]
3	ketchup	<---	[tortillas, coffee/tea, juice, soap]
4	pasta	<---	[paper towels, dishwashing liquid/detergent, e...

In [9]:

```
print(cat_df.isnull().values.sum())
```

```
0
```

In [10]:

```
print(cat_df['Consequent'].value_counts())
```

```
poultry                         11
soda                             3
soap                             2
ice cream                        2
cheeses                          2
paper towels                     2
yogurt                           2
ketchup                          2
beef                             2
spaghetti sauce                  1
mixes                            1
eggs                             1
milk                             1
bagels                           1
coffee/tea                       1
waffles                          1
dinner rolls                     1
dishwashing liquid/detergent     1
pasta                            1
lunch meat                       1
Name: Consequent, dtype: int64
```

In [11]:

```
cat=[]
num=[]
for i in df.columns:
    if df[i].dtype=="object":
        cat.append(i)
    else:
        num.append(i)
print(cat)
print(num)
```

```
['Consequent', 'implies', 'Items']
['Support', 'Confidence', 'Lift']
```

```
In [12]:
```

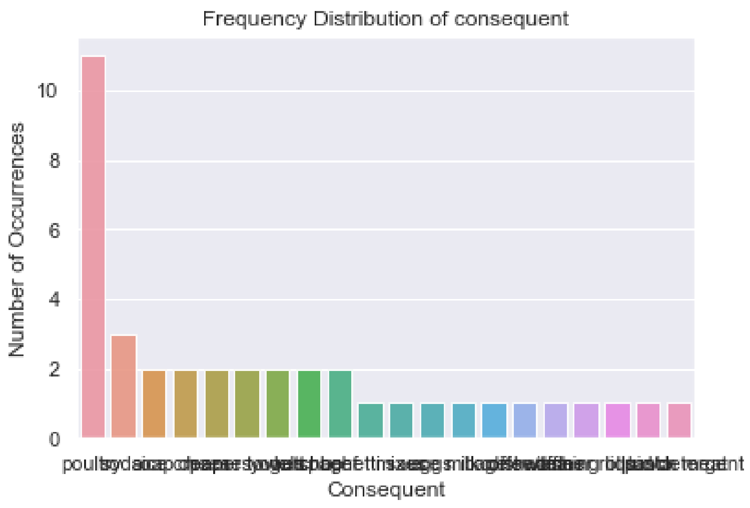
```python
for column in df.columns:
    if df[column].dtype == 'object':
        print(column.upper(),': ',df[column].nunique())
        print(df[column].value_counts().sort_values())
        print('\n')
```

```
CONSEQUENT :  20
spaghetti sauce                 1
dishwashing liquid/detergent    1
dinner rolls                    1
waffles                         1
coffee/tea                      1
bagels                          1
milk                            1
eggs                            1
mixes                           1
lunch meat                      1
pasta                           1
beef                            2
ketchup                         2
yogurt                          2
paper towels                    2
cheeses                         2
ice cream                       2
soap                            2
soda                            3
poultry                        11
Name: Consequent, dtype: int64


IMPLIES :  1
<---    39
Name: implies, dtype: int64


ITEMS :  39
[toilet paper, mixes, coffee/tea, soap]                                         1
[paper towels, dishwashing liquid/detergent, dinner rolls, ice cream, pasta]    1
[all- purpose, waffles, laundry detergent, juice]                               1
[shampoo, fruits, lunch meat, pork]                                             1
[eggs, dinner rolls, ice cream, pasta, lunch meat]                              1
[spaghetti sauce, poultry, waffles, laundry detergent]                          1
[dinner rolls, spaghetti sauce, beef, sugar]                                    1
[eggs, tortillas, coffee/tea, sugar]                                            1
[spaghetti sauce, laundry detergent, mixes, sugar]                              1
[paper towels, laundry detergent, soda, sugar]                                  1
[yogurt, dishwashing liquid/detergent, all- purpose, hand soap]                 1
[poultry, fruits, hand soap, sugar]                                             1
[dinner rolls, spaghetti sauce, sandwich loaves, soap]                          1
[all- purpose, flour, soda, ketchup]                                            1
[butter, spaghetti sauce, ice cream, lunch meat]                                1
[paper towels, yogurt, pasta, lunch meat]                                       1
[sandwich loaves, fruits, toilet paper, juice]                                  1
[paper towels, cereals, sandwich bags, sugar]                                   1
[dishwashing liquid/detergent, eggs, juice, sandwich bags]                      1
[paper towels, spaghetti sauce, milk, laundry detergent]                        1
[waffles, laundry detergent, mixes, soap]                                       1
[dinner rolls, spaghetti sauce, ice cream, beef]                                1
[paper towels, milk, individual meals, coffee/tea]                              1
[shampoo, hand soap, juice, sugar]                                              1
[paper towels, eggs, dinner rolls, pasta, lunch meat]                           1
[dinner rolls, spaghetti sauce, sandwich loaves, hand soap]                     1
[dinner rolls, spaghetti sauce, hand soap, soap]                                1
[ice cream, waffles, milk, pork]                                                1
[dinner rolls, spaghetti sauce, hand soap, coffee/tea]                          1
[spaghetti sauce, all- purpose, sandwich bags, ketchup]                         1
[tortillas, coffee/tea, juice, soap]                                            1
[paper towels, dishwashing liquid/detergent, eggs, dinner rolls, ice cream]     1
[butter, cheeses, sandwich loaves, laundry detergent]                           1
[yogurt, ice cream, tortillas, cereals]                                         1
[dinner rolls, spaghetti sauce, hand soap, sugar]                               1
```

```
[dinner rolls, spaghetti sauce, hand soap, sugar]                      1
[bagels, pasta, individual meals, pork]                                1
[cheeses, all- purpose, tortillas, coffee/tea]                         1
[eggs, poultry, beef, sandwich bags]                                   1
[eggs, dinner rolls, poultry, ice cream, pasta]                        1
Name: Items, dtype: int64
```

In [13]:

```python
import seaborn as sns
import matplotlib.pyplot as plt
product_count = cat_df['Consequent'].value_counts()
sns.set(style="darkgrid")
sns.barplot(product_count.index, product_count.values, alpha=0.9)
plt.title('Frequency Distribution of consequent')
plt.ylabel('Number of Occurrences', fontsize=12)
plt.xlabel('Consequent', fontsize=12)
plt.show()
```

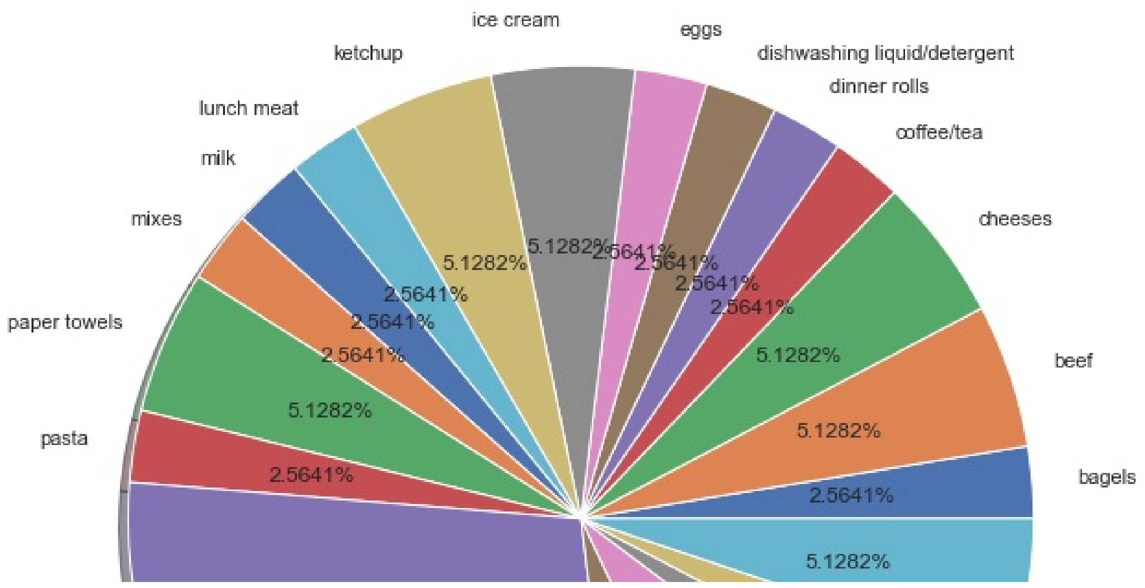


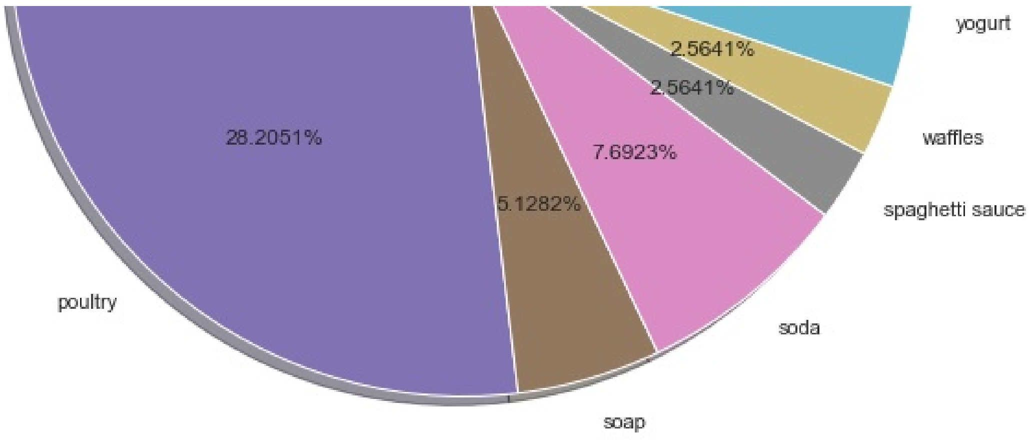
In [14]:

```python
labels = cat_df['Consequent'].astype('category').cat.categories.tolist()
```

In [15]:

```python
labels = cat_df['Consequent'].astype('category').cat.categories.tolist()
counts = cat_df['Consequent'].value_counts()
sizes = [counts[var_cat] for var_cat in labels]
fig1, ax1 = plt.subplots(figsize=(12,10))
ax1.pie(sizes, labels=labels, autopct='%1.4f%%', shadow=True) #autopct is show the % on plot
ax1.axis('equal')
plt.show()
```




yogurt
2.5641%
2.5641%
waffles
28.2051%
7.6923%
spaghetti sauce
5.1282%
poultry
soda
soap

```
In [ ]:
```

```
In [ ]:
```