ACTIVITY 5

Using the Apriori Algorithm for Market Basket Analysis

In [1]:

```
import matplotlib.pyplot as plt; plt.rcdefaults()
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
df = pd.read_csv('Workshop-5-dataset.zip', sep='\t',dtype=np.str)
```

In [3]:

```
# Question 1: First five rows of the dataset
df.head(5)
```

Out[3]:

	transaction_ID	Date	Time	item_0	item_1	item_2	item_3	
0	536365	01/12/2010	08:26	WHITE HANGING HEART T-LIGHT HOLDER	WHITE METAL LANTERN	CREAM CUPID HEARTS COAT HANGER	KNITTED UNION FLAG HOT WATER BOTTLE	W
1	536366	01/12/2010	08:28	HAND WARMER UNION JACK	HAND WARMER RED POLKA DOT	NaN	NaN	
2	536367	01/12/2010	08:34	ASSORTED COLOUR BIRD ORNAMENT	POPPY'S PLAYHOUSE BEDROOM	POPPY'S PLAYHOUSE KITCHEN	FELTCRAFT PRINCESS CHARLOTTE DOLL	K
3	536368	01/12/2010	08:34	JAM MAKING SET WITH JARS	RED COAT RACK PARIS FASHION	YELLOW COAT RACK PARIS FASHION	BLUE COAT RACK PARIS FASHION	
4	536369	01/12/2010	08:35	BATH BUILDING BLOCK WORD	NaN	NaN	NaN	

5 rows × 44 columns

In [4]:

```
# Question2: rows and columns in the dataset:
df.shape
```

Out[4]:

```
(31941, 44)
```

There are 31941 rows and 44 columns in the dataset

In [5]:

```
STUDENT_NAME = 'AbidemiAleem'
STUDENT_NO = '8712'
```

In [6]:

```
np.random.seed(int(STUDENT_NO))
unique_id = int('2' + STUDENT_NO)
rows = np.random.choice(df.index.values, unique_id)
data = df.loc[rows]
```

In [7]:

```
file_name = STUDENT_NAME + "_" + STUDENT_NO + ".csv"
data.to_csv(file_name)
```

In [8]:

```
# Question 3: Displaying Unique dates in the dataset
data['Date'] .nunique()
```

Out[8]:

```
305
```

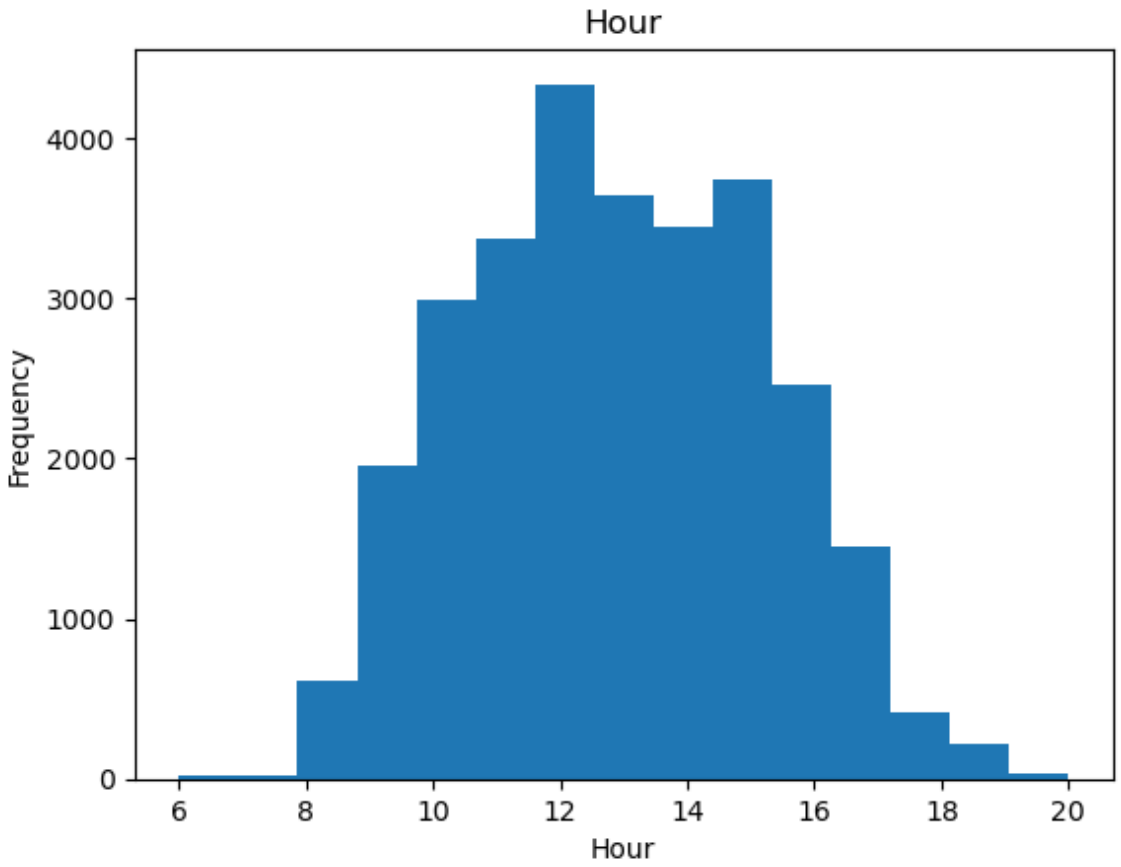
Answer: In the dataset, we have 305 unique dates.

In [9]:

```
data['Hour'] = pd.to_datetime(data['Time'], format='%H:%M').dt.hour
```

In [10]:

```
hour_hist = data.hist(column="Hour", bins=15, grid=False)
for ax in hour_hist.flatten():
 ax.set_xlabel("Hour")
 ax.set_ylabel("Frequency")
```



In [11]:

```
# import apyori
from apyori import apriori
```

In [12]:

```
data.head(1)
```

Out[12]:

	transaction_ID	Date	Time	item_0	item_1	item_2	item_3
15531	558837	04/07/2011	11:58	JUMBO BAG TOYS	JUMBO BAG WOODLAND ANIMALS	BLUE CALCULATOR RULER	SET/20 RED RETROSPOT PAPER NAPKINS

1 rows × 45 columns

In [13]:

```
items_df=data[data.columns[3:44]]
```

In [14]:

```
items_df.head()
```

Out[14]:

	item_0	item_1	item_2	item_3	item_4	item_5	item
15531	JUMBO BAG TOYS	JUMBO BAG WOODLAND ANIMALS	BLUE CALCULATOR RULER	SET/20 RED RETROSPOT PAPER NAPKINS	SET/6 RED SPOTTY PAPER CUPS	SET/6 RED SPOTTY PAPER PLATES	B DOUGHNU FRIDG MAGNE
20056	RED WOOLLY HOTTIE WHITE HEART	NaN	NaN	NaN	NaN	NaN	Na
23859	WHITE SOAP RACK WITH 2 BOTTLES	FIRST AID TIN	RED DINER WALL CLOCK	SET OF 4 PANTRY JELLY MOULDS	60 CAKE CASES VINTAGE CHRISTMAS	36 DOILIES VINTAGE CHRISTMAS	BOX OF MI VINTAG CRACKER
	SPACEBOY	WRAP	FANCY FONTS	SKULLS AND	WRAP I	WRAP BILLBOARD	RE

In [15]:

```
baskets = items_df.T.apply(lambda x: x.dropna().tolist()).tolist()
```

In [16]:

```
for i in baskets[:5]:
 print(i)
```

```
['JUMBO BAG TOYS', 'JUMBO BAG WOODLAND ANIMALS', 'BLUE CALCULATOR RULER',
'SET/20 RED RETROSPOT PAPER NAPKINS', 'SET/6 RED SPOTTY PAPER CUPS', 'SET/
6 RED SPOTTY PAPER PLATES', 'BIG DOUGHNUT FRIDGE MAGNETS', 'VINTAGE SNAP C
ARDS', 'JUMBO STORAGE BAG SUKI', 'SET OF 36 PAISLEY FLOWER DOILIES', 'PACK
OF 20 SKULL PAPER NAPKINS', 'SPACEBOY BIRTHDAY CARD', 'MINI LADLE LOVE HEA
RT RED', 'MINI LADLE LOVE HEART PINK', 'RED DINER WALL CLOCK', 'SMALL HEAR
T MEASURING SPOONS', 'LOVEBIRD HANGING DECORATION WHITE', 'GLASS JAR DAISY
FRESH COTTON WOOL', 'RECYCLING BAG RETROSPOT', 'TOY TIDY PINK POLKADOT',
'JUMBO BAG SPACEBOY DESIGN', 'JUMBO BAG PINK POLKADOT', 'JUMBO SHOPPER VIN
TAGE RED PAISLEY', 'PICNIC BASKET WICKER LARGE', 'TRAVEL SEWING KIT', 'PIN
K BABY BUNTING', 'DOORMAT WELCOME TO OUR HOME', 'PINK REGENCY TEACUP AND S
AUCER', 'RABBIT NIGHT LIGHT', 'SINGLE ANTIQUE ROSE HOOK IVORY', 'JUMBO BAG
APPLES', 'TEA TIME PARTY BUNTING', 'PINK HAWAIIAN PICNIC HAMPER FOR 2', 'S
ET/4 RED MINI ROSE CANDLE IN BOWL', 'JUMBO BAG RED RETROSPOT', 'DOTCOM POS
TAGE']
['RED WOOLLY HOTTIE WHITE HEART']
['WHITE SOAP RACK WITH 2 BOTTLES', 'FIRST AID TIN', 'RED DINER WALL CLOC
K', 'SET OF 4 PANTRY JELLY MOULDS', '60 CAKE CASES VINTAGE CHRISTMAS', '36
DOILIES VINTAGE CHRISTMAS', 'BOX OF 6 MINI VINTAGE CRACKERS', 'JAM MAKING
SET WITH JARS', 'TROPICAL  HONEYCOMB PAPER GARLAND', 'MULTICOLOUR HONEYCOM
B PAPER GARLAND', 'WHITE BELL HONEYCOMB PAPER', 'RED PAPER PARASOL', 'SET
OF 3 CAKE TINS PANTRY DESIGN', 'CLASSIC CAFE SUGAR DISPENSER', 'SET OF TEA
COFFEE SUGAR TINS PANTRY', 'ENAMEL FLOWER JUG CREAM', 'GREEN METAL BOX ARM
Y SUPPLIES', 'CHRISTMAS LIGHTS 10 VINTAGE BAUBLES', 'SET OF 6 SPICE TINS P
ANTRY DESIGN', 'VINTAGE SNAP CARDS', 'PAPER CHAIN KIT VINTAGE CHRISTMAS']
['SPACEBOY GIFT WRAP', 'WRAP PAISLEY PARK', 'FANCY FONTS BIRTHDAY WRAP',
'SKULLS AND CROSSBONES WRAP', 'WRAP I LOVE LONDON', 'WRAP BILLBOARD FONTS
DESIGN', 'RED RETROSPOT WRAP', 'PINK POLKADOT WRAP', 'BLUE POLKADOT WRAP',
'EMPIRE GIFT WRAP', 'LUNCH BAG RED RETROSPOT', 'LUNCH BAG ALPHABET DESIG
N', 'LUNCH BAG SUKI DESIGN', 'LUNCH BAG APPLE DESIGN', 'LUNCH BAG  BLACK S
KULL', 'LUNCH BAG DOLLY GIRL DESIGN', 'PINK OWL SOFT TOY', 'BLUE OWL SOFT
TOY', 'SHELF WITH 4 HOOKS HOME SWEET HOME', 'METAL 4 HOOK HANGER FRENCH CH
ATEAU', 'ENCHANTED BIRD COATHANGER 5 HOOK', 'AGED GLASS SILVER T-LIGHT HOL
DER', 'CINAMMON SET OF 9 T-LIGHTS', 'ORANGE SCENTED SET/9 T-LIGHTS', 'CHIL
LI LIGHTS', 'ROCKING HORSE GREEN CHRISTMAS', 'ROCKING HORSE RED CHRISTMA
S', 'FELTCRAFT PRINCESS CHARLOTTE DOLL', 'FELTCRAFT PRINCESS LOLA DOLL',
'FELTCRAFT CUSHION RABBIT', 'FELTCRAFT CUSHION OWL', 'FELTCRAFT CUSHION BU
TTERFLY', 'SINGLE ANTIQUE ROSE HOOK IVORY', 'HANGING HEART WITH BELL', 'CH
RISTMAS HANGING STAR WITH BELL', 'FAIRY TALE COTTAGE NIGHT LIGHT', 'CHILDR
ENS TOY COOKING UTENSIL SET', 'SPACEBOY BIRTHDAY CARD', 'FANCY FONT BIRTHD
AY CARD', 'CARD CAT AND TREE', 'CARD CIRCUS PARADE']
['CERAMIC STRAWBERRY CAKE MONEY BANK', 'RETROSPOT TEA SET CERAMIC 11 PC',
'BAG 250g SWIRLY MARBLES', 'VINTAGE SNAKES & LADDERS', 'HOLIDAY FUN LUDO',
'4 TRADITIONAL SPINNING TOPS', 'TRADTIONAL ALPHABET STAMP SET', 'ICE CREAM
SUNDAE LIP GLOSS', 'MINI PAINT SET VINTAGE', 'SET OF 3 WOODEN HEART DECORA
TIONS', 'WOODEN BOX OF DOMINOES', 'COLUMBIAN CANDLE RECTANGLE', 'PACK OF 1
2 TRADITIONAL CRAYONS', 'MR ROBOT SOFT TOY', 'MRS ROBOT SOFT TOY']
```

In [17]:

```
association_rules = apriori(baskets, min_support=0.01, min_confidence=0.2,
 min_lift=3, min_length=2)
association_results = list(association_rules)
```

In [18]:

```
print('Rules generated: ', len(association_results))
```

```
Rules generated:  96
```

In [19]:

```
print(association_results[0])
```

```
RelationRecord(items=frozenset({'PACK OF 72 RETROSPOT CAKE CASES', '60 TEA
TIME FAIRY CAKE CASES'}), support=0.010692393424352187, ordered_statistics
=[OrderedStatistic(items_base=frozenset({'60 TEATIME FAIRY CAKE CASES'}),
items_add=frozenset({'PACK OF 72 RETROSPOT CAKE CASES'}), confidence=0.409
3333333333333, lift=9.777686078757627), OrderedStatistic(items_base=frozen
set({'PACK OF 72 RETROSPOT CAKE CASES'}), items_add=frozenset({'60 TEATIME
FAIRY CAKE CASES'}), confidence=0.25540765391014975, lift=9.77768607875762
5)])
```

In [20]:

```
def display_rules(association_results):
    for item in association_results:
        pair = item[0]
        items = [x for x in pair]
        print("Rule: " + items[0] + " -> " + items[1])
        print("Support: " + str(item[1]))
        print("Confidence: " + str(item[2][0][2]))
        print("Lift: " + str(item[2][0][3]))
        print("====================================")
```

In [21]:

```
display_rules(association_results[:10])
```

```
Rule: PACK OF 72 RETROSPOT CAKE CASES -> 60 TEATIME FAIRY CAKE CASES
Support: 0.010692393424352187
Confidence: 0.4093333333333333
Lift: 9.777686078757627
====================================
Rule: ALARM CLOCK BAKELIKE GREEN -> ALARM CLOCK BAKELIKE PINK
Support: 0.012155196433546948
Confidence: 0.3856353591160221
Lift: 16.25897566951428
====================================
Rule: ALARM CLOCK BAKELIKE GREEN -> ALARM CLOCK BAKELIKE RED
Support: 0.01856366675954305
Confidence: 0.5889502762430939
Lift: 17.343528545119707
====================================
Rule: ALARM CLOCK BAKELIKE IVORY -> ALARM CLOCK BAKELIKE RED
Support: 0.011179994427417107
Confidence: 0.6269531249999999
Lift: 18.46264423076923
====================================
Rule: ALARM CLOCK BAKELIKE RED -> ALARM CLOCK BAKELIKE PINK
Support: 0.014628030091947618
Confidence: 0.6167400881057269
Lift: 18.16188862532475
====================================
Rule: CHARLOTTE BAG SUKI DESIGN -> CHARLOTTE BAG PINK POLKADOT
Support: 0.012085539147394818
Confidence: 0.5225903614457832
Lift: 18.320652573664624
====================================
Rule: RED RETROSPOT CHARLOTTE BAG -> CHARLOTTE BAG PINK POLKADOT
Support: 0.010762050710504319
Confidence: 0.4653614457831326
Lift: 14.005720997196333
====================================
Rule: RED RETROSPOT CHARLOTTE BAG -> CHARLOTTE BAG SUKI DESIGN
Support: 0.010100306492059069
Confidence: 0.35409035409035405
Lift: 10.656857700882856
====================================
Rule: GIN + TONIC DIET METAL SIGN -> COOK WITH WINE METAL SIGN
Support: 0.010378935636667596
Confidence: 0.4232954545454546
Lift: 16.490717898112745
====================================
Rule: PLEASE ONE PERSON METAL SIGN -> COOK WITH WINE METAL SIGN
Support: 0.010058790749512399
Confidence: 0.4318181818181818
Lift: 15.287748010312745
====================================
```

In [22]:

```
from collections import Counter

counter = Counter(baskets[0])
for i in baskets[1:]:
    if i != 'nan':
        counter.update(i)

del counter['nan']
counter.most_common(10)
```

Out[22]:

```
[('WHITE HANGING HEART T-LIGHT HOLDER', 2206),
 ('JUMBO BAG RED RETROSPOT', 1980),
 ('REGENCY CAKESTAND 3 TIER', 1966),
 ('PARTY BUNTING', 1537),
 ('LUNCH BAG RED RETROSPOT', 1492),
 ('SET OF 3 CAKE TINS PANTRY DESIGN', 1318),
 ('ASSORTED COLOUR BIRD ORNAMENT', 1310),
 ('LUNCH BAG  BLACK SKULL', 1268),
 ('PACK OF 72 RETROSPOT CAKE CASES', 1230),
 ('JUMBO BAG PINK POLKADOT', 1147)]
```

In [23]:

```
# Question 4a: Generating how many common items found in the rules displayed
most_common = [('WHITE HANGING HEART T-LIGHT HOLDER', 2206),
               ('JUMBO BAG RED RETROSPOT', 1980),
               ('REGENCY CAKESTAND 3 TIER', 1966),
               ('PARTY BUNTING', 1537),
               ('LUNCH BAG RED RETROSPOT', 1492),
               ('SET OF 3 CAKE TINS PANTRY DESIGN', 1318),
               ('ASSORTED COLOUR BIRD ORNAMENT', 1310),
               ('LUNCH BAG  BLACK SKULL', 1268),
               ('PACK OF 72 RETROSPOT CAKE CASES', 1230),
               ('JUMBO BAG PINK POLKADOT', 1147)]
most_common_items = [item[0] for item in most_common]
count = 0
for item in most_common_items:
    found = False
    for rule in association_results[:10]:
        pair = rule[0]
        items = [x for x in pair]
        if item in items:
            found = True
            break
    if found:
        count += 1
print("Number of most common items found in the rules:", count)
```

```
Number of most common items found in the rules: 1
```

Answer: Only one item can be found in the rules created which is,

('PACK OF 72 RETROSPOT CAKE CASES', 1230),

Question 4b:

Due to the fact that only 10 rules out of the 96 created rules generated were displayed, not all of the top 10 items were included. Also, if the top items are especially common or appear frequently in the data because they are not closely associated to any one item, they may not appear as strong association rules. On the other hand, since they don't frequently occur alongside other goods, the top items might not be included in many association rules if they are incredibly rare or only happen sometimes.

In conclusion, the threshold values, the properties of the data collection, and the intricacy of the underlying patterns all affect whether or not the top items in the association rules appear. As a result, it's crucial to carefully select appropriate threshold values and to consider the association rule mining findings in the context of the particular issue at hand.

In [24]:

```
# Setting 1
association_rules = apriori(baskets, min_support=0.015, min_confidence=0.7,
 min_lift=3, min_length=2)
association_results = list(association_rules)
```

In [25]:

```
print('Rules generated: ', len(association_results))
```

```
Rules generated:  4
```

In [26]:

```
print(association_results[0])
```

```
RelationRecord(items=frozenset({'GREEN REGENCY TEACUP AND SAUCER', 'PINK R
EGENCY TEACUP AND SAUCER'}), support=0.019643354694901086, ordered_statist
ics=[OrderedStatistic(items_base=frozenset({'PINK REGENCY TEACUP AND SAUCE
R'}), items_add=frozenset({'GREEN REGENCY TEACUP AND SAUCER'}), confidence
=0.7866108786610878, lift=24.129456782176447)])
```

In [27]:

```
def display_rules(association_results):
    for item in association_results:
        pair = item[0]
        items = [x for x in pair]
        print("Rule: " + items[0] + " -> " + items[1])
        print("Support: " + str(item[1]))
        print("Confidence: " + str(item[2][0][2]))
        print("Lift: " + str(item[2][0][3]))
        print("====================================")
```

In [28]:

```
display_rules(association_results[:10])
```

```
Rule: GREEN REGENCY TEACUP AND SAUCER -> PINK REGENCY TEACUP AND SAUCER
Support: 0.019643354694901086
Confidence: 0.7866108786610878
Lift: 24.129456782176447
=====================================
Rule: ROSES REGENCY TEACUP AND SAUCER -> GREEN REGENCY TEACUP AND SAUCER
Support: 0.02354416271942045
Confidence: 0.7222222222222222
Lift: 20.861614129219763
=====================================
Rule: ROSES REGENCY TEACUP AND SAUCER -> PINK REGENCY TEACUP AND SAUCER
Support: 0.01845918083031485
Confidence: 0.7391910739191073
Lift: 21.351764702580898
=====================================
Rule: ROSES REGENCY TEACUP AND SAUCER -> GREEN REGENCY TEACUP AND SAUCER
Support: 0.016160490387294512
Confidence: 0.822695035460993
Lift: 23.76380267420124
=====================================
```

In [29]:

```
from collections import Counter

counter = Counter(baskets[0])
for i in baskets[1:]:
    if i != 'nan':
        counter.update(i)

del counter['nan']
counter.most_common(10)
```

Out[29]:

```
[('WHITE HANGING HEART T-LIGHT HOLDER', 2206),
 ('JUMBO BAG RED RETROSPOT', 1980),
 ('REGENCY CAKESTAND 3 TIER', 1966),
 ('PARTY BUNTING', 1537),
 ('LUNCH BAG RED RETROSPOT', 1492),
 ('SET OF 3 CAKE TINS PANTRY DESIGN', 1318),
 ('ASSORTED COLOUR BIRD ORNAMENT', 1310),
 ('LUNCH BAG  BLACK SKULL', 1268),
 ('PACK OF 72 RETROSPOT CAKE CASES', 1230),
 ('JUMBO BAG PINK POLKADOT', 1147)]
```

In [30]:

```
# Generating how many common items found in the rules displayed
most_common = [('WHITE HANGING HEART T-LIGHT HOLDER', 2206),
               ('JUMBO BAG RED RETROSPOT', 1980),
               ('REGENCY CAKESTAND 3 TIER', 1966),
               ('PARTY BUNTING', 1537),
               ('LUNCH BAG RED RETROSPOT', 1492),
               ('SET OF 3 CAKE TINS PANTRY DESIGN', 1318),
               ('ASSORTED COLOUR BIRD ORNAMENT', 1310),
               ('LUNCH BAG  BLACK SKULL', 1268),
               ('PACK OF 72 RETROSPOT CAKE CASES', 1230),
               ('JUMBO BAG PINK POLKADOT', 1147)]
most_common_items = [item[0] for item in most_common]
count = 0
for item in most_common_items:
    found = False
    for rule in association_results[:10]:
        pair = rule[0]
        items = [x for x in pair]
        if item in items:
            found = True
            break
    if found:
        count += 1
print("Number of most common items found in the rules:", count)
```

```
Number of most common items found in the rules: 0
```

In [31]:

```
# Setting 2
association_rules = apriori(baskets, min_support=0.009, min_confidence=0.5,
 min_lift=3, min_length=2)
association_results = list(association_rules)
```

In [32]:

```
print('Rules generated: ', len(association_results))
```

```
Rules generated:  48
```

In [33]:

```
print(association_results[0])
```

```
RelationRecord(items=frozenset({'ALARM CLOCK BAKELIKE GREEN', 'ALARM CLOCK
BAKELIKE IVORY'}), support=0.009334076344385623, ordered_statistics=[Order
edStatistic(items_base=frozenset({'ALARM CLOCK BAKELIKE IVORY'}), items_ad
d=frozenset({'ALARM CLOCK BAKELIKE GREEN'}), confidence=0.5234375, lift=1
6.606560773480663)])
```

In [34]:

```
def display_rules(association_results):
    for item in association_results:
        pair = item[0]
        items = [x for x in pair]
        print("Rule: " + items[0] + " -> " + items[1])
        print("Support: " + str(item[1]))
        print("Confidence: " + str(item[2][0][2]))
        print("Lift: " + str(item[2][0][3]))
        print("====================================")
```

In [35]:

```
display_rules(association_results[:10])
```

```
Rule: ALARM CLOCK BAKELIKE GREEN -> ALARM CLOCK BAKELIKE IVORY
Support: 0.009334076344385623
Confidence: 0.5234375
Lift: 16.606560773480663
====================================
Rule: ALARM CLOCK BAKELIKE GREEN -> ALARM CLOCK BAKELIKE PINK
Support: 0.012155196433546948
Confidence: 0.5124816446402349
Lift: 16.25897566951428
====================================
Rule: ALARM CLOCK BAKELIKE GREEN -> ALARM CLOCK BAKELIKE RED
Support: 0.01856366675954305
Confidence: 0.5889502762430939
Lift: 17.343528545119707
====================================
Rule: ALARM CLOCK BAKELIKE IVORY -> ALARM CLOCK BAKELIKE RED
Support: 0.011179994427417107
Confidence: 0.6269531249999999
Lift: 18.46264423076923
====================================
Rule: ALARM CLOCK BAKELIKE ORANGE -> ALARM CLOCK BAKELIKE RED
Support: 0.009368904987461688
Confidence: 0.6862244897959183
Lift: 20.20807953950811
====================================
Rule: ALARM CLOCK BAKELIKE RED -> ALARM CLOCK BAKELIKE PINK
Support: 0.014628030091947618
Confidence: 0.6167400881057269
Lift: 18.16188862532475
====================================
Rule: BAKING SET SPACEBOY DESIGN -> BAKING SET 9 PIECE RETROSPOT
Support: 0.009647534132070215
Confidence: 0.6309794988610479
Lift: 20.752214629207796
====================================
Rule: RED  HARMONICA IN BOX -> BLUE HARMONICA IN BOX
Support: 0.009717191418222346
Confidence: 0.5294117647058824
Lift: 24.796852509356107
====================================
Rule: CHARLOTTE BAG SUKI DESIGN -> CHARLOTTE BAG PINK POLKADOT
Support: 0.012085539147394818
Confidence: 0.5225903614457832
Lift: 18.320652573664624
====================================
Rule: SPACEBOY LUNCH BOX -> DOLLY GIRL LUNCH BOX
Support: 0.016613262747283367
Confidence: 0.6259842519685039
Lift: 22.693509902171318
====================================
```

In [36]:

```
from collections import Counter

counter = Counter(baskets[0])
for i in baskets[1:]:
    if i != 'nan':
        counter.update(i)

del counter['nan']
counter.most_common(10)
```

Out[36]:

```
[('WHITE HANGING HEART T-LIGHT HOLDER', 2206),
 ('JUMBO BAG RED RETROSPOT', 1980),
 ('REGENCY CAKESTAND 3 TIER', 1966),
 ('PARTY BUNTING', 1537),
 ('LUNCH BAG RED RETROSPOT', 1492),
 ('SET OF 3 CAKE TINS PANTRY DESIGN', 1318),
 ('ASSORTED COLOUR BIRD ORNAMENT', 1310),
 ('LUNCH BAG  BLACK SKULL', 1268),
 ('PACK OF 72 RETROSPOT CAKE CASES', 1230),
 ('JUMBO BAG PINK POLKADOT', 1147)]
```

In [37]:

```
# Setting 3
association_rules = apriori(baskets, min_support=0.015, min_confidence=0.5,
 min_lift=9, min_length=2)
association_results = list(association_rules)
```

In [38]:

```
print('Rules generated: ', len(association_results))
```

```
Rules generated:  10
```

In [39]:

```
print(association_results[0])
```

```
RelationRecord(items=frozenset({'ALARM CLOCK BAKELIKE GREEN', 'ALARM CLOCK
BAKELIKE RED'}), support=0.01856366675954305, ordered_statistics=[OrderedS
tatistic(items_base=frozenset({'ALARM CLOCK BAKELIKE GREEN'}), items_add=f
rozenset({'ALARM CLOCK BAKELIKE RED'}), confidence=0.5889502762430939, lif
t=17.343528545119707), OrderedStatistic(items_base=frozenset({'ALARM CLOCK
BAKELIKE RED'}), items_add=frozenset({'ALARM CLOCK BAKELIKE GREEN'}), conf
idence=0.5466666666666667, lift=17.343528545119707)])
```

In [40]:

```
def display_rules(association_results):
    for item in association_results:
        pair = item[0]
        items = [x for x in pair]
        print("Rule: " + items[0] + " -> " + items[1])
        print("Support: " + str(item[1]))
        print("Confidence: " + str(item[2][0][2]))
        print("Lift: " + str(item[2][0][3]))
        print("====================================")
```

In [41]:

```
display_rules(association_results[:10])
```

```
Rule: ALARM CLOCK BAKELIKE GREEN -> ALARM CLOCK BAKELIKE RED
Support: 0.01856366675954305
Confidence: 0.5889502762430939
Lift: 17.343528545119707
====================================
Rule: SPACEBOY LUNCH BOX -> DOLLY GIRL LUNCH BOX
Support: 0.016613262747283367
Confidence: 0.6259842519685039
Lift: 22.693509902171318
====================================
Rule: GARDENERS KNEELING PAD CUP OF TEA -> GARDENERS KNEELING PAD KEEP CAL
M
Support: 0.016926720534967958
Confidence: 0.6952789699570816
Lift: 24.05162624747919
====================================
Rule: GREEN REGENCY TEACUP AND SAUCER -> PINK REGENCY TEACUP AND SAUCER
Support: 0.019643354694901086
Confidence: 0.6025641025641026
Lift: 24.12945678217645
====================================
Rule: ROSES REGENCY TEACUP AND SAUCER -> GREEN REGENCY TEACUP AND SAUCER
Support: 0.02354416271942045
Confidence: 0.7222222222222222
Lift: 20.861614129219763
====================================
Rule: JUMBO BAG RED RETROSPOT -> JUMBO BAG STRAWBERRY
Support: 0.016926720534967958
Confidence: 0.6262886597938144
Lift: 9.188553909044456
====================================
Rule: ROSES REGENCY TEACUP AND SAUCER -> PINK REGENCY TEACUP AND SAUCER
Support: 0.01845918083031485
Confidence: 0.7391910739191073
Lift: 21.351764702580898
====================================
Rule: WOODLAND CHARLOTTE BAG -> RED RETROSPOT CHARLOTTE BAG
Support: 0.01542908888269713
Confidence: 0.5775749674054759
Lift: 17.38294807562476
====================================
Rule: WOODEN PICTURE FRAME WHITE FINISH -> WOODEN FRAME ANTIQUE WHITE
Support: 0.016160490387294512
Confidence: 0.5155555555555555
Lift: 14.526625231708648
====================================
Rule: ROSES REGENCY TEACUP AND SAUCER -> GREEN REGENCY TEACUP AND SAUCER
Support: 0.016160490387294512
Confidence: 0.6471408647140865
Lift: 27.486255188862206
====================================
```

In [42]:

```
from collections import Counter

counter = Counter(baskets[0])
for i in baskets[1:]:
    if i != 'nan':
        counter.update(i)

del counter['nan']
counter.most_common(10)
```

Out[42]:

```
[('WHITE HANGING HEART T-LIGHT HOLDER', 2206),
 ('JUMBO BAG RED RETROSPOT', 1980),
 ('REGENCY CAKESTAND 3 TIER', 1966),
 ('PARTY BUNTING', 1537),
 ('LUNCH BAG RED RETROSPOT', 1492),
 ('SET OF 3 CAKE TINS PANTRY DESIGN', 1318),
 ('ASSORTED COLOUR BIRD ORNAMENT', 1310),
 ('LUNCH BAG  BLACK SKULL', 1268),
 ('PACK OF 72 RETROSPOT CAKE CASES', 1230),
 ('JUMBO BAG PINK POLKADOT', 1147)]
```

In [44]:

```
 #Generating how many common items found in the rules displayed
most_common = [('WHITE HANGING HEART T-LIGHT HOLDER', 2206),
               ('JUMBO BAG RED RETROSPOT', 1980),
               ('REGENCY CAKESTAND 3 TIER', 1966),
               ('PARTY BUNTING', 1537),
               ('LUNCH BAG RED RETROSPOT', 1492),
               ('SET OF 3 CAKE TINS PANTRY DESIGN', 1318),
               ('ASSORTED COLOUR BIRD ORNAMENT', 1310),
               ('LUNCH BAG  BLACK SKULL', 1268),
               ('PACK OF 72 RETROSPOT CAKE CASES', 1230),
               ('JUMBO BAG PINK POLKADOT', 1147)]
most_common_items = [item[0] for item in most_common]
count = 0
for item in most_common_items:
    found = False
    for rule in association_results[:10]:
        pair = rule[0]
        items = [x for x in pair]
        if item in items:
            found = True
            break
    if found:
        count += 1
print("Number of most common items found in the rules:", count)
```

```
Number of most common items found in the rules: 1
```

Question 5: Number of rules for each setting are;

Setting 1: Rules generated = 4

Setting 2: Rules generated = 48

Setting 3: Rules generated = 10

The first set of rules are related to tea cups and saucers, while the second and third sets of rules are related to different colors of alarm clocks and lunch boxes, and a gardening kneeling pad. The lift value for all rules is higher than 1, indicating a positive association between the items. The confidence values for the first set of rules are higher than those for the other sets of rules, indicating a stronger association between the items. Overall, the lift and confidence values suggest that the rules are meaningful and may be useful for targeted marketing or inventory management.

Question 6a

Filter the transactions on the 'day' of the week to perform analysis on two durations. We will be considering Tuesday and Sunday.

In [45]:

```
df['Date'] = pd.to_datetime(df['Date'])
```

In [46]:

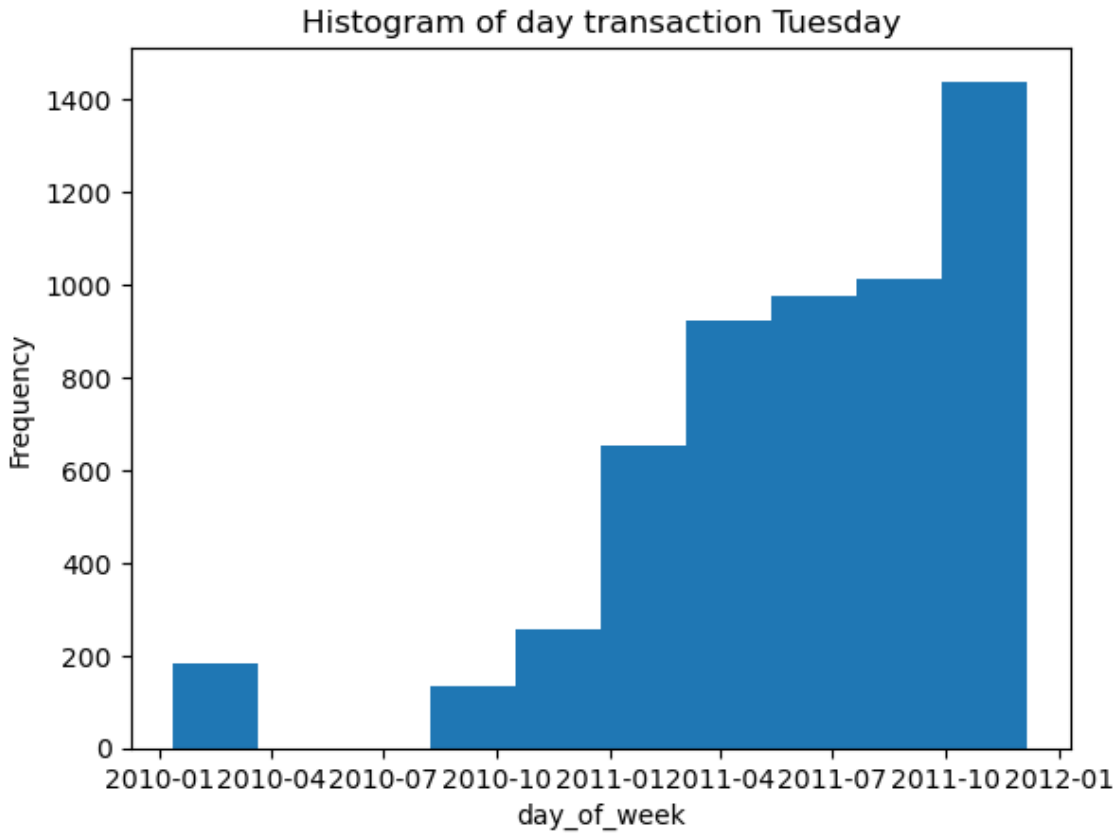
```
# converting the column to datetime format
df['Date'] = pd.to_datetime(df['Date'], format='%d-%m-%Y')
```

In [47]:

```
# Choosing the day of the week I want to filter
day_of_week = 'Tuesday'
# Filter transactions for the day of the week picked
df_filtered = df[df['Date'].dt.day_name() == day_of_week]
```

In [48]:

```
# Plot a histogram of the filtered data
plt.hist(df_filtered['Date'], bins=10)
plt.xlabel('day_of_week')
plt.ylabel('Frequency')
plt.title('Histogram of day transaction {}'.format(day_of_week))
plt.show()
```



Observation:

We can see that transactions occur from 2010-01 (January 2010) to 2012-01 but no significant transaction made between 2010-04 to 2010-07 and 2012-01. We also noticed that sales was increasing from 2010-10 to 2011-10(which has the highest transaction made)

In [49]:

```
items_df.head()
```

Out[49]:

	item_0	item_1	item_2	item_3	item_4	item_
15531	JUMBO BAG TOYS	JUMBO BAG WOODLAND ANIMALS	BLUE CALCULATOR RULER	SET/20 RED RETROSPOT PAPER NAPKINS	SET/6 RED SPOTTY PAPER CUPS	SET/6 REI SPOTT PAPEI PLATE
20056	RED WOOLLY HOTTIE WHITE HEART	NaN	NaN	NaN	NaN	Nal
23859	WHITE SOAP RACK WITH 2 BOTTLES	FIRST AID TIN	RED DINER WALL CLOCK	SET OF 4 PANTRY JELLY MOULDS	60 CAKE CASES VINTAGE CHRISTMAS	36 DOILIE VINTAG CHRISTMA
22275	SPACEBOY GIFT WRAP	WRAP PAISLEY PARK	FANCY FONTS BIRTHDAY WRAP	SKULLS AND CROSSBONES WRAP	WRAP I LOVE LONDON	WRA BILLBOARI FONT DESIGI
30890	CERAMIC STRAWBERRY CAKE MONEY BANK	RETROSPOT TEA SET CERAMIC 11 PC	BAG 250g SWIRLY MARBLES	VINTAGE SNAKES & LADDERS	HOLIDAY FUN LUDO	TRADITIONA SPINNIN TOP

5 rows × 41 columns

In [50]:

```
baskets = items_df.T.apply(lambda x: x.dropna().tolist()).tolist()
```

In [51]:

```
for i in baskets[:5]:
 print(i)
```

```
['JUMBO BAG TOYS', 'JUMBO BAG WOODLAND ANIMALS', 'BLUE CALCULATOR RULER',
'SET/20 RED RETROSPOT PAPER NAPKINS', 'SET/6 RED SPOTTY PAPER CUPS', 'SET/
6 RED SPOTTY PAPER PLATES', 'BIG DOUGHNUT FRIDGE MAGNETS', 'VINTAGE SNAP C
ARDS', 'JUMBO STORAGE BAG SUKI', 'SET OF 36 PAISLEY FLOWER DOILIES', 'PACK
OF 20 SKULL PAPER NAPKINS', 'SPACEBOY BIRTHDAY CARD', 'MINI LADLE LOVE HEA
RT RED', 'MINI LADLE LOVE HEART PINK', 'RED DINER WALL CLOCK', 'SMALL HEAR
T MEASURING SPOONS', 'LOVEBIRD HANGING DECORATION WHITE', 'GLASS JAR DAISY
FRESH COTTON WOOL', 'RECYCLING BAG RETROSPOT', 'TOY TIDY PINK POLKADOT',
'JUMBO BAG SPACEBOY DESIGN', 'JUMBO BAG PINK POLKADOT', 'JUMBO SHOPPER VIN
TAGE RED PAISLEY', 'PICNIC BASKET WICKER LARGE', 'TRAVEL SEWING KIT', 'PIN
K BABY BUNTING', 'DOORMAT WELCOME TO OUR HOME', 'PINK REGENCY TEACUP AND S
AUCER', 'RABBIT NIGHT LIGHT', 'SINGLE ANTIQUE ROSE HOOK IVORY', 'JUMBO BAG
APPLES', 'TEA TIME PARTY BUNTING', 'PINK HAWAIIAN PICNIC HAMPER FOR 2', 'S
ET/4 RED MINI ROSE CANDLE IN BOWL', 'JUMBO BAG RED RETROSPOT', 'DOTCOM POS
TAGE']
['RED WOOLLY HOTTIE WHITE HEART']
['WHITE SOAP RACK WITH 2 BOTTLES', 'FIRST AID TIN', 'RED DINER WALL CLOC
K', 'SET OF 4 PANTRY JELLY MOULDS', '60 CAKE CASES VINTAGE CHRISTMAS', '36
DOILIES VINTAGE CHRISTMAS', 'BOX OF 6 MINI VINTAGE CRACKERS', 'JAM MAKING
SET WITH JARS', 'TROPICAL  HONEYCOMB PAPER GARLAND', 'MULTICOLOUR HONEYCOM
B PAPER GARLAND', 'WHITE BELL HONEYCOMB PAPER', 'RED PAPER PARASOL', 'SET
OF 3 CAKE TINS PANTRY DESIGN', 'CLASSIC CAFE SUGAR DISPENSER', 'SET OF TEA
COFFEE SUGAR TINS PANTRY', 'ENAMEL FLOWER JUG CREAM', 'GREEN METAL BOX ARM
Y SUPPLIES', 'CHRISTMAS LIGHTS 10 VINTAGE BAUBLES', 'SET OF 6 SPICE TINS P
ANTRY DESIGN', 'VINTAGE SNAP CARDS', 'PAPER CHAIN KIT VINTAGE CHRISTMAS']
['SPACEBOY GIFT WRAP', 'WRAP PAISLEY PARK', 'FANCY FONTS BIRTHDAY WRAP',
'SKULLS AND CROSSBONES WRAP', 'WRAP I LOVE LONDON', 'WRAP BILLBOARD FONTS
DESIGN', 'RED RETROSPOT WRAP', 'PINK POLKADOT WRAP', 'BLUE POLKADOT WRAP',
'EMPIRE GIFT WRAP', 'LUNCH BAG RED RETROSPOT', 'LUNCH BAG ALPHABET DESIG
N', 'LUNCH BAG SUKI DESIGN', 'LUNCH BAG APPLE DESIGN', 'LUNCH BAG  BLACK S
KULL', 'LUNCH BAG DOLLY GIRL DESIGN', 'PINK OWL SOFT TOY', 'BLUE OWL SOFT
TOY', 'SHELF WITH 4 HOOKS HOME SWEET HOME', 'METAL 4 HOOK HANGER FRENCH CH
ATEAU', 'ENCHANTED BIRD COATHANGER 5 HOOK', 'AGED GLASS SILVER T-LIGHT HOL
DER', 'CINAMMON SET OF 9 T-LIGHTS', 'ORANGE SCENTED SET/9 T-LIGHTS', 'CHIL
LI LIGHTS', 'ROCKING HORSE GREEN CHRISTMAS', 'ROCKING HORSE RED CHRISTMA
S', 'FELTCRAFT PRINCESS CHARLOTTE DOLL', 'FELTCRAFT PRINCESS LOLA DOLL',
'FELTCRAFT CUSHION RABBIT', 'FELTCRAFT CUSHION OWL', 'FELTCRAFT CUSHION BU
TTERFLY', 'SINGLE ANTIQUE ROSE HOOK IVORY', 'HANGING HEART WITH BELL', 'CH
RISTMAS HANGING STAR WITH BELL', 'FAIRY TALE COTTAGE NIGHT LIGHT', 'CHILDR
ENS TOY COOKING UTENSIL SET', 'SPACEBOY BIRTHDAY CARD', 'FANCY FONT BIRTHD
AY CARD', 'CARD CAT AND TREE', 'CARD CIRCUS PARADE']
['CERAMIC STRAWBERRY CAKE MONEY BANK', 'RETROSPOT TEA SET CERAMIC 11 PC',
'BAG 250g SWIRLY MARBLES', 'VINTAGE SNAKES & LADDERS', 'HOLIDAY FUN LUDO',
'4 TRADITIONAL SPINNING TOPS', 'TRADTIONAL ALPHABET STAMP SET', 'ICE CREAM
SUNDAE LIP GLOSS', 'MINI PAINT SET VINTAGE', 'SET OF 3 WOODEN HEART DECORA
TIONS', 'WOODEN BOX OF DOMINOES', 'COLUMBIAN CANDLE RECTANGLE', 'PACK OF 1
2 TRADITIONAL CRAYONS', 'MR ROBOT SOFT TOY', 'MRS ROBOT SOFT TOY']
```

In [52]:

```
association_rules = apriori(baskets, min_support=0.01, min_confidence=0.2,
 min_lift=3, min_length=2)
association_results = list(association_rules)
```

In [53]:

```
print('Rules generated: ', len(association_results))
```

```
Rules generated:  96
```

In [54]:

```
print(association_results[0])
```

```
RelationRecord(items=frozenset({'PACK OF 72 RETROSPOT CAKE CASES', '60 TEA
TIME FAIRY CAKE CASES'}), support=0.010692393424352187, ordered_statistics
=[OrderedStatistic(items_base=frozenset({'60 TEATIME FAIRY CAKE CASES'}),
items_add=frozenset({'PACK OF 72 RETROSPOT CAKE CASES'}), confidence=0.409
3333333333333, lift=9.777686078757627), OrderedStatistic(items_base=frozen
set({'PACK OF 72 RETROSPOT CAKE CASES'}), items_add=frozenset({'60 TEATIME
FAIRY CAKE CASES'}), confidence=0.25540765391014975, lift=9.77768607875762
5)])
```

In [55]:

```
def display_rules(association_results):
    for item in association_results:
        pair = item[0]
        items = [x for x in pair]
        print("Rule: " + items[0] + " -> " + items[1])
        print("Support: " + str(item[1]))
        print("Confidence: " + str(item[2][0][2]))
        print("Lift: " + str(item[2][0][3]))
        print("====================================")
```

In [56]:

```
display_rules(association_results[:10])
```

```
Rule: PACK OF 72 RETROSPOT CAKE CASES -> 60 TEATIME FAIRY CAKE CASES
Support: 0.010692393424352187
Confidence: 0.4093333333333333
Lift: 9.777686078757627
====================================
Rule: ALARM CLOCK BAKELIKE GREEN -> ALARM CLOCK BAKELIKE PINK
Support: 0.012155196433546948
Confidence: 0.3856353591160221
Lift: 16.25897566951428
====================================
Rule: ALARM CLOCK BAKELIKE GREEN -> ALARM CLOCK BAKELIKE RED
Support: 0.01856366675954305
Confidence: 0.5889502762430939
Lift: 17.343528545119707
====================================
Rule: ALARM CLOCK BAKELIKE IVORY -> ALARM CLOCK BAKELIKE RED
Support: 0.011179994427417107
Confidence: 0.6269531249999999
Lift: 18.46264423076923
====================================
Rule: ALARM CLOCK BAKELIKE RED -> ALARM CLOCK BAKELIKE PINK
Support: 0.014628030091947618
Confidence: 0.6167400881057269
Lift: 18.16188862532475
====================================
Rule: CHARLOTTE BAG SUKI DESIGN -> CHARLOTTE BAG PINK POLKADOT
Support: 0.012085539147394818
Confidence: 0.5225903614457832
Lift: 18.320652573664624
====================================
Rule: RED RETROSPOT CHARLOTTE BAG -> CHARLOTTE BAG PINK POLKADOT
Support: 0.010762050710504319
Confidence: 0.4653614457831326
Lift: 14.005720997196333
====================================
Rule: RED RETROSPOT CHARLOTTE BAG -> CHARLOTTE BAG SUKI DESIGN
Support: 0.010100306492059069
Confidence: 0.35409035409035405
Lift: 10.656857700882856
====================================
Rule: GIN + TONIC DIET METAL SIGN -> COOK WITH WINE METAL SIGN
Support: 0.010378935636667596
Confidence: 0.4232954545454546
Lift: 16.490717898112745
====================================
Rule: PLEASE ONE PERSON METAL SIGN -> COOK WITH WINE METAL SIGN
Support: 0.010058790749512399
Confidence: 0.4318181818181818
Lift: 15.287748010312745
====================================
```

In [57]:

```
from collections import Counter

counter = Counter(baskets[0])
for i in baskets[1:]:
    if i != 'nan':
        counter.update(i)

del counter['nan']
counter.most_common(10)
```

Out[57]:

```
[('WHITE HANGING HEART T-LIGHT HOLDER', 2206),
 ('JUMBO BAG RED RETROSPOT', 1980),
 ('REGENCY CAKESTAND 3 TIER', 1966),
 ('PARTY BUNTING', 1537),
 ('LUNCH BAG RED RETROSPOT', 1492),
 ('SET OF 3 CAKE TINS PANTRY DESIGN', 1318),
 ('ASSORTED COLOUR BIRD ORNAMENT', 1310),
 ('LUNCH BAG  BLACK SKULL', 1268),
 ('PACK OF 72 RETROSPOT CAKE CASES', 1230),
 ('JUMBO BAG PINK POLKADOT', 1147)]
```

In [59]:

```
 #Generating how many common items found in the rules displayed
most_common = [('WHITE HANGING HEART T-LIGHT HOLDER', 2206),
               ('JUMBO BAG RED RETROSPOT', 1980),
               ('REGENCY CAKESTAND 3 TIER', 1966),
               ('PARTY BUNTING', 1537),
               ('LUNCH BAG RED RETROSPOT', 1492),
               ('SET OF 3 CAKE TINS PANTRY DESIGN', 1318),
               ('ASSORTED COLOUR BIRD ORNAMENT', 1310),
               ('LUNCH BAG  BLACK SKULL', 1268),
               ('PACK OF 72 RETROSPOT CAKE CASES', 1230),
               ('JUMBO BAG PINK POLKADOT', 1147)]
most_common_items = [item[0] for item in most_common]
count = 0
for item in most_common_items:
    found = False
    for rule in association_results[:10]:
        pair = rule[0]
        items = [x for x in pair]
        if item in items:
            found = True
            break
    if found:
        count += 1
print("Number of most common items found in the rules:", count)
```

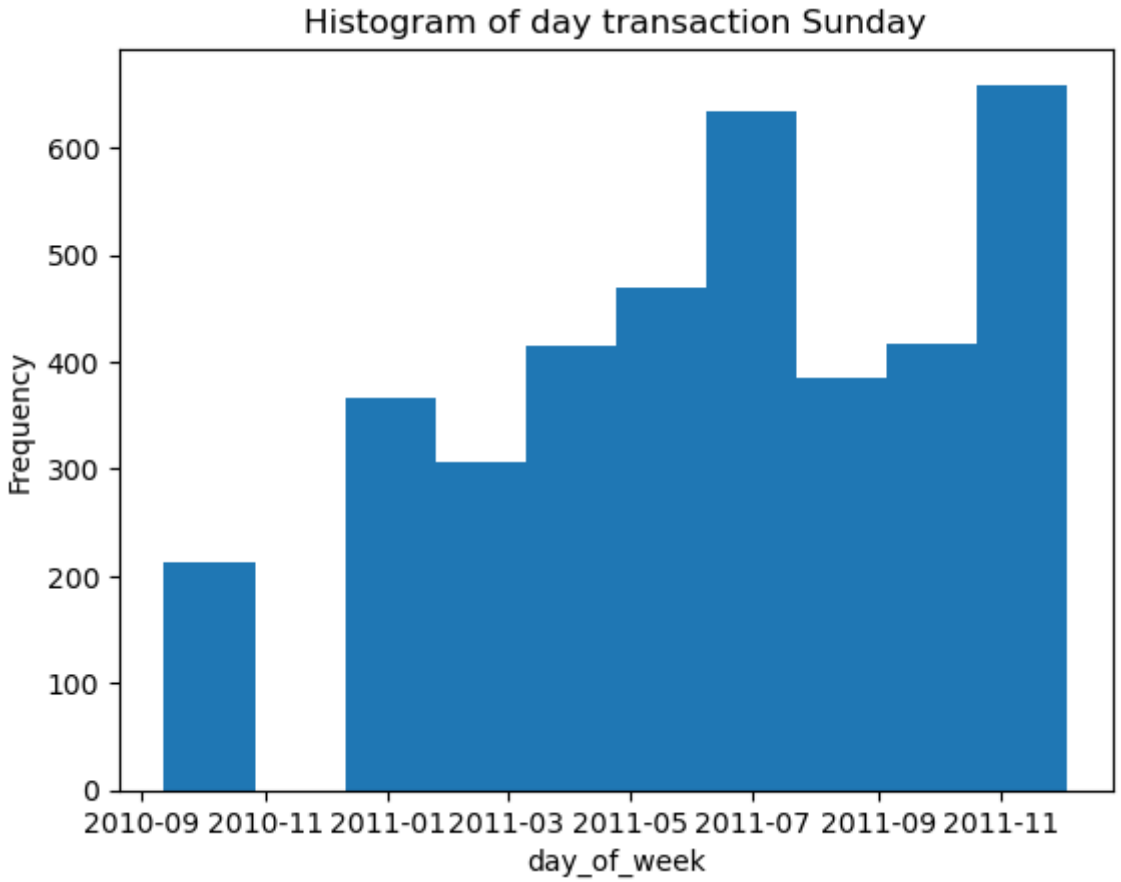
```
Number of most common items found in the rules: 1
```

In [60]:

```
# Choosing the day of the week I want to filter
day_of_week = 'Sunday'
# Filter transactions for the day of the week chosen
df_filtered = df[df['Date'].dt.day_name() == day_of_week]
```

In [61]:

```
# Plot a histogram of the filtered data
plt.hist(df_filtered['Date'], bins=10)
plt.xlabel('day_of_week')
plt.ylabel('Frequency')
plt.title('Histogram of day transaction {}'.format(day_of_week))
plt.show()
```



Observation

We can see that here transactions occur from 2010-09 (September 2010) to 2011-11. We also noticed that sales increade and decreased, highest transaction made occur in 2011-07 and 2011-11

In [62]:

```
baskets = items_df.T.apply(lambda x: x.dropna().tolist()).tolist()
```

In [63]:

```
for i in baskets[:5]:
 print(i)
```

```
['JUMBO BAG TOYS', 'JUMBO BAG WOODLAND ANIMALS', 'BLUE CALCULATOR RULER',
'SET/20 RED RETROSPOT PAPER NAPKINS', 'SET/6 RED SPOTTY PAPER CUPS', 'SET/
6 RED SPOTTY PAPER PLATES', 'BIG DOUGHNUT FRIDGE MAGNETS', 'VINTAGE SNAP C
ARDS', 'JUMBO STORAGE BAG SUKI', 'SET OF 36 PAISLEY FLOWER DOILIES', 'PACK
OF 20 SKULL PAPER NAPKINS', 'SPACEBOY BIRTHDAY CARD', 'MINI LADLE LOVE HEA
RT RED', 'MINI LADLE LOVE HEART PINK', 'RED DINER WALL CLOCK', 'SMALL HEAR
T MEASURING SPOONS', 'LOVEBIRD HANGING DECORATION WHITE', 'GLASS JAR DAISY
FRESH COTTON WOOL', 'RECYCLING BAG RETROSPOT', 'TOY TIDY PINK POLKADOT',
'JUMBO BAG SPACEBOY DESIGN', 'JUMBO BAG PINK POLKADOT', 'JUMBO SHOPPER VIN
TAGE RED PAISLEY', 'PICNIC BASKET WICKER LARGE', 'TRAVEL SEWING KIT', 'PIN
K BABY BUNTING', 'DOORMAT WELCOME TO OUR HOME', 'PINK REGENCY TEACUP AND S
AUCER', 'RABBIT NIGHT LIGHT', 'SINGLE ANTIQUE ROSE HOOK IVORY', 'JUMBO BAG
APPLES', 'TEA TIME PARTY BUNTING', 'PINK HAWAIIAN PICNIC HAMPER FOR 2', 'S
ET/4 RED MINI ROSE CANDLE IN BOWL', 'JUMBO BAG RED RETROSPOT', 'DOTCOM POS
TAGE']
['RED WOOLLY HOTTIE WHITE HEART']
['WHITE SOAP RACK WITH 2 BOTTLES', 'FIRST AID TIN', 'RED DINER WALL CLOC
K', 'SET OF 4 PANTRY JELLY MOULDS', '60 CAKE CASES VINTAGE CHRISTMAS', '36
DOILIES VINTAGE CHRISTMAS', 'BOX OF 6 MINI VINTAGE CRACKERS', 'JAM MAKING
SET WITH JARS', 'TROPICAL  HONEYCOMB PAPER GARLAND', 'MULTICOLOUR HONEYCOM
B PAPER GARLAND', 'WHITE BELL HONEYCOMB PAPER', 'RED PAPER PARASOL', 'SET
OF 3 CAKE TINS PANTRY DESIGN', 'CLASSIC CAFE SUGAR DISPENSER', 'SET OF TEA
COFFEE SUGAR TINS PANTRY', 'ENAMEL FLOWER JUG CREAM', 'GREEN METAL BOX ARM
Y SUPPLIES', 'CHRISTMAS LIGHTS 10 VINTAGE BAUBLES', 'SET OF 6 SPICE TINS P
ANTRY DESIGN', 'VINTAGE SNAP CARDS', 'PAPER CHAIN KIT VINTAGE CHRISTMAS']
['SPACEBOY GIFT WRAP', 'WRAP PAISLEY PARK', 'FANCY FONTS BIRTHDAY WRAP',
'SKULLS AND CROSSBONES WRAP', 'WRAP I LOVE LONDON', 'WRAP BILLBOARD FONTS
DESIGN', 'RED RETROSPOT WRAP', 'PINK POLKADOT WRAP', 'BLUE POLKADOT WRAP',
'EMPIRE GIFT WRAP', 'LUNCH BAG RED RETROSPOT', 'LUNCH BAG ALPHABET DESIG
N', 'LUNCH BAG SUKI DESIGN', 'LUNCH BAG APPLE DESIGN', 'LUNCH BAG  BLACK S
KULL', 'LUNCH BAG DOLLY GIRL DESIGN', 'PINK OWL SOFT TOY', 'BLUE OWL SOFT
TOY', 'SHELF WITH 4 HOOKS HOME SWEET HOME', 'METAL 4 HOOK HANGER FRENCH CH
ATEAU', 'ENCHANTED BIRD COATHANGER 5 HOOK', 'AGED GLASS SILVER T-LIGHT HOL
DER', 'CINAMMON SET OF 9 T-LIGHTS', 'ORANGE SCENTED SET/9 T-LIGHTS', 'CHIL
LI LIGHTS', 'ROCKING HORSE GREEN CHRISTMAS', 'ROCKING HORSE RED CHRISTMA
S', 'FELTCRAFT PRINCESS CHARLOTTE DOLL', 'FELTCRAFT PRINCESS LOLA DOLL',
'FELTCRAFT CUSHION RABBIT', 'FELTCRAFT CUSHION OWL', 'FELTCRAFT CUSHION BU
TTERFLY', 'SINGLE ANTIQUE ROSE HOOK IVORY', 'HANGING HEART WITH BELL', 'CH
RISTMAS HANGING STAR WITH BELL', 'FAIRY TALE COTTAGE NIGHT LIGHT', 'CHILDR
ENS TOY COOKING UTENSIL SET', 'SPACEBOY BIRTHDAY CARD', 'FANCY FONT BIRTHD
AY CARD', 'CARD CAT AND TREE', 'CARD CIRCUS PARADE']
['CERAMIC STRAWBERRY CAKE MONEY BANK', 'RETROSPOT TEA SET CERAMIC 11 PC',
'BAG 250g SWIRLY MARBLES', 'VINTAGE SNAKES & LADDERS', 'HOLIDAY FUN LUDO',
'4 TRADITIONAL SPINNING TOPS', 'TRADTIONAL ALPHABET STAMP SET', 'ICE CREAM
SUNDAE LIP GLOSS', 'MINI PAINT SET VINTAGE', 'SET OF 3 WOODEN HEART DECORA
TIONS', 'WOODEN BOX OF DOMINOES', 'COLUMBIAN CANDLE RECTANGLE', 'PACK OF 1
2 TRADITIONAL CRAYONS', 'MR ROBOT SOFT TOY', 'MRS ROBOT SOFT TOY']
```

In [64]:

```
association_rules = apriori(baskets, min_support=0.01, min_confidence=0.2,
 min_lift=3, min_length=2)
association_results = list(association_rules)
```

In [65]:

```
print('Rules generated: ', len(association_results))
```

```
Rules generated:  96
```

In [66]:

```
def display_rules(association_results):
    for item in association_results:
        pair = item[0]
        items = [x for x in pair]
        print("Rule: " + items[0] + " -> " + items[1])
        print("Support: " + str(item[1]))
        print("Confidence: " + str(item[2][0][2]))
        print("Lift: " + str(item[2][0][3]))
        print("====================================")
```

In [67]:

```
display_rules(association_results[:10])
```

```
Rule: PACK OF 72 RETROSPOT CAKE CASES -> 60 TEATIME FAIRY CAKE CASES
Support: 0.010692393424352187
Confidence: 0.4093333333333333
Lift: 9.777686078757627
====================================
Rule: ALARM CLOCK BAKELIKE GREEN -> ALARM CLOCK BAKELIKE PINK
Support: 0.012155196433546948
Confidence: 0.3856353591160221
Lift: 16.25897566951428
====================================
Rule: ALARM CLOCK BAKELIKE GREEN -> ALARM CLOCK BAKELIKE RED
Support: 0.01856366675954305
Confidence: 0.5889502762430939
Lift: 17.343528545119707
====================================
Rule: ALARM CLOCK BAKELIKE IVORY -> ALARM CLOCK BAKELIKE RED
Support: 0.011179994427417107
Confidence: 0.6269531249999999
Lift: 18.46264423076923
====================================
Rule: ALARM CLOCK BAKELIKE RED -> ALARM CLOCK BAKELIKE PINK
Support: 0.014628030091947618
Confidence: 0.6167400881057269
Lift: 18.16188862532475
====================================
Rule: CHARLOTTE BAG SUKI DESIGN -> CHARLOTTE BAG PINK POLKADOT
Support: 0.012085539147394818
Confidence: 0.5225903614457832
Lift: 18.320652573664624
====================================
Rule: RED RETROSPOT CHARLOTTE BAG -> CHARLOTTE BAG PINK POLKADOT
Support: 0.010762050710504319
Confidence: 0.4653614457831326
Lift: 14.005720997196333
====================================
Rule: RED RETROSPOT CHARLOTTE BAG -> CHARLOTTE BAG SUKI DESIGN
Support: 0.010100306492059069
Confidence: 0.35409035409035405
Lift: 10.656857700882856
====================================
Rule: GIN + TONIC DIET METAL SIGN -> COOK WITH WINE METAL SIGN
Support: 0.010378935636667596
Confidence: 0.4232954545454546
Lift: 16.490717898112745
====================================
Rule: PLEASE ONE PERSON METAL SIGN -> COOK WITH WINE METAL SIGN
Support: 0.010587907495123 99
Confidence: 0.4318181818181818
Lift: 15.287748010312745
====================================
```

In [68]:

```
from collections import Counter

counter = Counter(baskets[0])
for i in baskets[1:]:
    if i != 'nan':
        counter.update(i)

del counter['nan']
counter.most_common(10)
```

Out[68]:

```
[('WHITE HANGING HEART T-LIGHT HOLDER', 2206),
 ('JUMBO BAG RED RETROSPOT', 1980),
 ('REGENCY CAKESTAND 3 TIER', 1966),
 ('PARTY BUNTING', 1537),
 ('LUNCH BAG RED RETROSPOT', 1492),
 ('SET OF 3 CAKE TINS PANTRY DESIGN', 1318),
 ('ASSORTED COLOUR BIRD ORNAMENT', 1310),
 ('LUNCH BAG  BLACK SKULL', 1268),
 ('PACK OF 72 RETROSPOT CAKE CASES', 1230),
 ('JUMBO BAG PINK POLKADOT', 1147)]
```

In [70]:

```
 #Generating how many common items found in the rules displayed
most_common = [('WHITE HANGING HEART T-LIGHT HOLDER', 2206),
               ('JUMBO BAG RED RETROSPOT', 1980),
               ('REGENCY CAKESTAND 3 TIER', 1966),
               ('PARTY BUNTING', 1537),
               ('LUNCH BAG RED RETROSPOT', 1492),
               ('SET OF 3 CAKE TINS PANTRY DESIGN', 1318),
               ('ASSORTED COLOUR BIRD ORNAMENT', 1310),
               ('LUNCH BAG  BLACK SKULL', 1268),
               ('PACK OF 72 RETROSPOT CAKE CASES', 1230),
               ('JUMBO BAG PINK POLKADOT', 1147)]
most_common_items = [item[0] for item in most_common]
count = 0
for item in most_common_items:
    found = False
    for rule in association_results[:10]:
        pair = rule[0]
        items = [x for x in pair]
        if item in items:
            found = True
            break
    if found:
        count += 1
print("Number of most common items found in the rules:", count)
```

```
Number of most common items found in the rules: 1
```

Question6:

Rules generated = 96 for the two durations of the analysis, which was done on Tuessday and Sunday. Between the selected durations, there are no significant differnces in the purchasing patterns, they contain same items in the same order. Between the two sets, there is no distinction.

REFRENCES:

[1] Dr Vinita Nahar Workshop 5 - Market Basket Analysis using Apriori Algorithm notebbok.

[2] Lim, Y. (2022). Data Mining: Market Basket Analysis with Apriori Algorithm. [online] Medium. Available at: https://towardsdatascience.com/data-mining-market-basket-analysis-with-apriori-algorithm-970ff256a92c#:~:text=To%20perform%20a%20Market%20Basket%20Analysis%20implementation%20with (https://towardsdatascience.com/data-mining-market-basket-analysis-with-apriori-algorithm-970ff256a92c#:~:text=To%20perform%20a%20Market%20Basket%20Analysis%20implementation%20with) [Accessed 3 Mar. 2023].

[3] Khan, T.A. (2021). Market basket analysis using Apriori algorithm. [online] Medium. Available at: https://medium.com/@takhan.11/market-basket-analysis-using-apriori-algorithm-68c57af5cd9f (https://medium.com/@takhan.11/market-basket-analysis-using-apriori-algorithm-68c57af5cd9f) [Accessed 3 Mar. 2023].

[4] practicaldatascience.co.uk. (2021). How to use the Apriori algorithm for Market Basket Analysis. [online] Available at: https://practicaldatascience.co.uk/data-science/how-to-use-the-apriori-algorithm-for-market-basket-analysis (https://practicaldatascience.co.uk/data-science/how-to-use-the-apriori-algorithm-for-market-basket-analysis).

In [ ]: