In [1]:
```
import time
import os
import findspark
findspark.init()
from pyspark.sql.types import *
import pyspark
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
import pyspark.sql.functions as func
from pyspark.sql.functions import col
sc = SparkContext.getOrCreate()
spark = SparkSession.builder.getOrCreate()
print(sc.version)
print(spark.version)
```

```
2.4.5
2.4.5
```

In [2]:
```
from IPython.core.display import display, HTML
display(HTML("<style>.container { width:90% !important; }</style>"))
```

In [3]:
```
# Import data from CVS and Remove null
dirpathname = "C:/BigData/~data/590project/"
filename = "UserBehavior.csv"
# filename='ub.csv'
df=spark.read.option("sep", ",").option("header", "false").csv(dirp
# name of each line
df=df.select(col("_c0").alias('user_id'),col("_c1").alias('item_id'
# print(df.count())
#delete the null row
df=df.dropna(thresh=5)
#delete the duplication
df=df.distinct().orderBy(['user_id'],ascending=True)
df.show(10)
```

```
+-------+-------+-----------+-----------+----------+
|user_id|item_id|category_id|behavior_id| timestamp|
+-------+-------+-----------+-----------+----------+
|      1| 266784|    2520771|         pv|1511909676|
|      1|4615417|    4145813|         pv|1511870864|
|      1|2087357|    2131531|         pv|1511975142|
|      1|3682069|    4690421|         pv|1512059832|
|      1|5002615|    2520377|         pv|1511839385|
|      1|4198227|    1320293|         pv|1512149929|
|      1|4954999|     411153|         pv|1512061318|
|      1|1531036|    2920476|         pv|1511733732|
|      1|2104483|    4756105|         pv|1512194830|
|      1| 929177|    4801426|         pv|1512252443|
+-------+-------+-----------+-----------+----------+
only showing top 10 rows
```

In [4]:

```
# transfer timestamp to Date and time
from pyspark.sql.functions import split,explode
df.timestamp.cast('float')
# select the date form 2017-11-25~2017-12-03
df=df.filter(df.timestamp>=1511586000)\
     .filter(df.timestamp<=1512363599)
df=df.withColumn('time',func.from_unixtime(df.timestamp,'yyyyMMdd H
df=df.withColumn('s',split(df['time'],' '))
df=df.withColumn('time',df['s'].getItem(1))
df=df.withColumn('Date',df['s'].getItem(0))
df=df.drop('timestamp','s')
df.show(10)
```

```
+-------+-------+-----------+-----------+---------+--------+
|user_id|item_id|category_id|behavior_id|     time|    Date|
+-------+-------+-----------+-----------+---------+--------+
|      1| 266784|    2520771|         pv|22:54:036|20171128|
|      1|4615417|    4145813|         pv|12:07:044|20171128|
|      1|2087357|    2131531|         pv|17:05:042|20171129|
|      1|2104483|    4756105|         pv|06:07:010|20171202|
|      1|3682069|    4690421|         pv|16:37:012|20171130|
|      1|4954999|     411153|         pv|17:01:058|20171130|
|      1|5002615|    2520377|         pv|03:23:005|20171128|
|      1|4198227|    1320293|         pv|17:38:049|20171201|
|      1| 929177|    4801426|         pv|22:07:023|20171202|
|      1|1531036|    2920476|         pv|22:02:012|20171126|
+-------+-------+-----------+-----------+---------+--------+
only showing top 10 rows
```

In [7]:
```
# Basic data statistics
print('Total number of behaviors :{0}'.format(df.count()))

# the number of customers
df_user=df.select('user_id').distinct()
print('There are total {0} customers.'.format(df_user.count()))

# the number of days
df_day=df.select('Date').distinct()
print('Total record:{0} days.'.format(df_day.count()))

# Daily behavior statistics
data_daily=df.groupBy("Date").count().orderBy(["count"], ascending=
data_daily.show()
```

```
Total number of behaviors :3704006
There are total 37375 customers.
Total record:9 days.
+--------+------+
|    Date| count|
+--------+------+
|20171202|540471|
|20171203|463170|
|20171201|430425|
|20171130|402767|
|20171126|401888|
|20171129|393249|
|20171127|381776|
|20171128|376521|
|20171125|313739|
+--------+------+
```

In [8]:
```
# the total number of caterages
categ=df.dropDuplicates(['category_id']).count()
print('There are total {0} category.'.format(cater))
# The number of items which were bought
buy_categ=df.dropDuplicates(['category_id']).filter(df.behavior_id=
print('Buying category:{0}.'.format(buy_categ.count()))


# The total number of items
item=df.dropDuplicates(['item_id']).count()
print('There are total {0} items.'.format(item))
# The number of items which were bought
buy_item=df.dropDuplicates(['item_id']).filter(df.behavior_id=='buy
print('Buying items:{0}.'.format(buy_item.count()))

df_user=df.groupBy("user_id").count().orderBy(["count"], ascending=
df_user.show()
```

```
There are total 7081 category.
Buying category:262.
There are total 912758 items.
Buying items:12749.
+-------+-----+
|user_id|count|
+-------+-----+
| 115477|  755|
| 221012|  729|
| 164127|  714|
| 116139|  703|
| 114912|  689|
| 115906|  667|
|1010419|  665|
| 142071|  663|
| 213141|  655|
| 144929|  629|
| 167945|  615|
| 182888|  614|
| 190128|  611|
| 227503|  597|
|  19116|  593|
| 243436|  592|
|1014799|  587|
| 121233|  582|
|  18326|  582|
| 189865|  578|
+-------+-----+
only showing top 20 rows
```

In [9]:
```
# the number of behaviors and the proportion, which allowing the sa

# the number of pv(page view), cart(add in shopping cart), fav(purc
df_pv=df.filter(df.behavior_id=='pv')
pv=df_pv.count()
print('Total behavior of page view: {0}.'.format(pv))
print('percent:{:.2%}.'.format(pv/3704006))

df_cart=df.filter(df.behavior_id=='cart')
cart=df_cart.count()
print('Total behavior of adding in shopping cart: {0},.'.format(car
print('percent:{:.2%}.'.format(cart/3704006))

df_fav=df.filter(df.behavior_id=='fav')
fav=df_fav.count()
print('Total behavior of favor an item: {0}.'.format(fav))
print('percent:{:.2%}.'.format(fav/3704006))

df_buy=df.filter(df.behavior_id=='buy')
buy=df_buy.count()
print('Total behavior of buying a item: {0}.'.format(buy))
print('percent:{:.2%}.'.format(buy/3704006))
```

```
Total behavior of page view: 3316230.
percent:89.53%.
Total behavior of adding in shopping cart: 206584,.
percent:5.58%.
Total behavior of favor an item: 107060.
percent:2.89%.
Total behavior of buying a item: 74132.
percent:2.00%.
```

In [10]:
```
#Remove duplicates, the same operation on the same item by the same
df_dic=df.dropDuplicates(['user_id','behavior_id','item_id'])
dic=df_dic.count()
print('Remove repetitive operations, the total operation is Shared:
```

```
Remove repetitive operations, the total operation is Shared:301106
5.
```

In [11]:
```
# If the same person only counts the same action of agreeing item o

df_pv1=df_dic.filter(df_dic.behavior_id=='pv')
pv1=df_pv1.count()
print('Total behavior of page view: {0}.'.format(pv1))
print('percent:{:.2%}.'.format(pv1/dic))

df_cart1=df_dic.filter(df_dic.behavior_id=='cart')
cart1=df_cart1.count()
print('Total behavior of adding in shopping cart: {0},.'.format(car
print('percent:{:.2%}.'.format(cart1/dic))

df_fav1=df_dic.filter(df_dic.behavior_id=='fav')
fav1=df_fav1.count()
print('Total behavior of favor a item: {0}.'.format(fav1))
print('percent:{:.2%}.'.format(fav1/dic))

df_buy1=df_dic.filter(df_dic.behavior_id=='buy')
buy1=df_buy1.count()
print('Total behavior of buying a item: {0}.'.format(buy1))
print('percent:{:.2%}.'.format(buy1/dic))
```

```
Total behavior of page view: 2632816.
percent:87.44%.
Total behavior of adding in shopping cart: 201221,.
percent:6.68%.
Total behavior of favor a item: 106419.
percent:3.53%.
Total behavior of buying a item: 70609.
percent:2.34%.
```

In [12]:
```
# If the same person only counts the same action of agreeing item o
# calculate the conversion rate of a product at each stage from cli

# the conversion rate for adding a item in cart after viewing it
pv_cart=cart1/pv1
print('The conversion rate for adding a item in cart after viewing
# the conversion rate for buying a item after adding it in cart
cart_buy=buy1/cart1
print('The conversion rate for buying a item after favor it in cart
# the conversion rate for adding a item in cart after viewing it
pv_fav=fav1/pv1
print('The conversion rate for favor a item in cart after viewing i
# the conversion rate for buying a item after adding it in cart
fav_buy=buy1/fav1
print('The conversion rate for buying a item after adding it in car
```

```
The conversion rate for adding a item in cart after viewing it: 7.
64%.
The conversion rate for buying a item after favor it in cart: 35.0
9%.
The conversion rate for favor a item in cart after viewing it: 4.0
4%.
The conversion rate for buying a item after adding it in cart: 66.
35%.
```

In [13]:
```
# calculate the number of behaviors for different categories

df_categ_pv=df.withColumn('beha',func.when(df['behavior_id']=='pv',
             .groupBy("category_id").sum('beha').orderBy(['sum(beha
             .withColumn('count_pv',col("sum(beha)").cast("int")).s
print('The Top 10 categorys of vering page.')
df_categ_pv.show(10)


df_categ_cart=df_cart.groupBy('category_id').count().orderBy(["coun
                     .withColumn("count_cart",col('count')).select
print('The Top 10 categorys of adding in cart.')
df_categ_cart.show(10)


df_categ_fav=df_fav.groupBy('category_id').count().orderBy(["count"
                    .withColumn("count_fav",col('count')).select('c
print('The Top 10 categorys of favoring.')
df_categ_fav.show(10)


# df_categ_buy=df_buy.groupBy('category_id').count().orderBy(["coun
df_categ_buy=df.withColumn('beha',func.when(df['behavior_id']=='buy
             .groupBy("category_id").sum('beha').orderBy(['sum(beha
             .withColumn('count_buy',col("sum(beha)").cast("int")).
print('The Top 10 categorys of buying.')
df_categ_buy.show(10)

#Compare categorys that have been viewed more than categorys that h
df_categ=df_categ_buy.join(df_categ_pv,'category_id')\
                     .select('category_id','count_pv','count_buy').s
```

```
The Top 10 categorys of vering page.
+-----------+--------+
|category_id|count_pv|
+-----------+--------+
|    4756105|  181114|
|    4145813|  119676|
|    2355072|  117063|
|    3607361|  111408|
|     982926|  104926|
|    2520377|   73510|
|    4801426|   70522|
|    1320293|   64251|
|    2465336|   57359|
|    3002561|   53595|
+-----------+--------+
only showing top 10 rows

The Top 10 categorys of adding in cart.
+-----------+----------+
```

In [14]:
```
# the Top 10 items for different behaviors
df_item_pv=df.withColumn('beha',func.when(df['behavior_id']=='pv',1
            .groupBy("item_id").sum('beha').orderBy(['sum(beha)'],
            .withColumn('count_pv',col("sum(beha)").cast("int")).s
# df_item_pv=df.filter(df.behavior_id=='pv').groupBy("item_id").cou
#                          .orderBy(["count"], ascending=False)\
#                          .withColumn("count_pv",col('count')).show
print('The Top 10 items of vering page.')
df_item_pv.show(10)

df_item_cart=df.filter(df.behavior_id=='cart').groupBy('item_id').c
                        .orderBy(['count'],ascending=False)\
                        .withColumn("count_cart",col('count')).sele
print('The Top 10 items of adding in cart.')
df_item_cart.show(10)

df_item_fav=df.filter(df.behavior_id=='fav').groupBy("item_id").cou
                        .orderBy(["count"], ascending=False).withCo
                        .select('item_id','count_fav')

print('The Top 10 items of favoring.')
df_item_fav.show(10)

df_item_buy=df.withColumn('beha',func.when(df['behavior_id']=='buy'
            .groupBy("item_id").sum('beha').orderBy(['sum(beha)'],
            .withColumn('count_buy',col("sum(beha)").cast("int")).
# df_item_buy=df.filter(df.behavior_id=='buy').groupBy("item_id").c
#                          .withColumn('count_buy',col('count')).sho
print('The Top 10 items of buying.')
df_item_buy.show(10)

# Compare items that have been viewed more than items that have bee
df_item=df_item_buy.join(df_item_pv,'item_id')\
                    .select('item_id','count_pv','count_buy').show(
```

```
The Top 10 items of vering page.
+-------+--------+
|item_id|count_pv|
+-------+--------+
| 812879|    1129|
|3845720|     907|
|2032668|     803|
|2331370|     799|
| 138964|     755|
|3031354|     698|
|1535294|     690|
|3371523|     672|
|2338453|     664|
|4211339|     653|
+-------+--------+
only showing top 10 rows

The Top 10 items of adding in cart.
+-------+----------+
|item_id|count_cart|
```

```
+-------+----------+
|3031354|        66|
|2331370|        63|
|2560262|        57|
|2818406|        50|
| 812879|        49|
|1535294|        48|
| 705557|        45|
|2453685|        44|
|2279428|        41|
| 138964|        39|
+-------+----------+
only showing top 10 rows

The Top 10 items of favoring.
+-------+---------+
|item_id|count_fav|
+-------+---------+
| 812879|       40|
|2279428|       40|
|2818406|       38|
|2364679|       30|
|2331370|       30|
|2887571|       26|
| 138964|       26|
|1419997|       25|
|1783990|       25|
|1535294|       25|
+-------+---------+
only showing top 10 rows

The Top 10 items of buying.
+-------+---------+
|item_id|count_buy|
+-------+---------+
|3122135|       58|
|3031354|       31|
|2964774|       27|
|2560262|       25|
|1910706|       24|
|1116492|       23|
| 257772|       23|
|3964583|       22|
|1042152|       22|
|1034594|       21|
+-------+---------+
only showing top 10 rows

+-------+--------+---------+
|item_id|count_pv|count_buy|
+-------+--------+---------+
| 100010|       1|        0|
|1000240|       1|        0|
| 100140|       1|        0|
|1002185|       1|        0|
|1004266|       2|        0|
|1005483|       1|        0|
```

```
|1007636|       1|         0|
| 100768|       1|         0|
|1009080|       1|         0|
|1009129|       5|         0|
|1010103|       3|         0|
|1010262|       3|         0|
|1010503|       1|         0|
|1010896|       1|         0|
|1012418|      11|         0|
|1015551|       5|         0|
|1018899|      23|         0|
|1020260|       1|         0|
|1020987|       1|         0|
|1022204|       1|         0|
+-------+--------+----------+
only showing top 20 rows
```

In [15]:

```
# according to the RFM model to calculate the value of custums
# R-Last buying time: the range or date is 9, using 0-8 points to s
df=df.withColumn('Date',col("Date").cast("int"))
# ta=df.groupBy("user_id").max("Date").collect()[0].orderBy(['max(D
df_1=df.groupBy("user_id").max("Date").orderBy(['max(Date)'])
df_R=df_1.withColumn("R",func.when(df_1['max(Date)']==20171125,0).w
                                .when(df_1['max(Date)']==20171128,3).w
                                .when(df_1['max(Date)']==20171201,6).w
                                .when(df_1['max(Date)']==20171203,8))
df_R.show()
```

```
+-------+---------+---+
|user_id|max(Date)|  R|
+-------+---------+---+
| 229333| 20171125|  0|
|  22649| 20171128|  3|
|  11693| 20171129|  4|
| 216393| 20171129|  4|
| 108670| 20171129|  4|
| 139194| 20171130|  5|
| 127992| 20171130|  5|
| 149606| 20171130|  5|
| 125767| 20171130|  5|
| 126295| 20171130|  5|
| 131717| 20171130|  5|
| 109147| 20171201|  6|
| 108296| 20171201|  6|
| 126583| 20171201|  6|
| 122031| 20171201|  6|
| 125192| 20171201|  6|
| 125700| 20171201|  6|
```

In [23]:

```
# F-frequency of the buying
df_2=df.withColumn('beha',func.when(df['behavior_id']=='buy',1).oth
df_F=df_2.groupBy("user_id").sum('beha').orderBy(['sum(beha)'])\
                                        .withColumn('fre',col("sum(beh
print(df_F.select('fre').rdd.min()[0])
print(df_F.select('fre').rdd.max()[0])
# During this period, the maximum number of users purchased 84 time
# The number of users used to purchase was divided into 0-8 grades
df_F=df_F.withColumn('F',func.when(df_F['fre']<=9,0)
                                .when((col('fre')>=10) & (col('fre')<=
                                .when((df_F['fre']>=20) & (df_F['fre']
                                .when((df_F['fre']>=30) & (df_F['fre']
                                .when((df_F['fre']>=40) & (df_F['fre']
                                .when((df_F['fre']>=50) & (df_F['fre']
                                .when((df_F['fre']>=60) & (df_F['fre']
                                .when((df_F['fre']>=70) & (df_F['fre']
                                .when((df_F['fre']>=80) & (df_F['fre']

df_RF=df_R.join(df_F,'user_id').select('user_id','R','F')
df_RF=df_RF.withColumn('sum',df_RF['R']+df_RF['F']).orderBy(-col("s
df_RF.show()
```

```
0
84
+-------+---+---+---+
|user_id|  R|  F|sum|
+-------+---+---+---+
| 234304|  8|  8| 16|
| 107932|  8|  7| 15|
| 190873|  8|  6| 14|
| 242650|  8|  6| 14|
| 122504|  8|  6| 14|
| 128379|  8|  6| 14|
|1008380|  8|  5| 13|
| 165222|  8|  4| 12|
| 158803|  8|  4| 12|
|1003983|  8|  4| 12|
| 140047|  8|  4| 12|
|  23206|  8|  3| 11|
| 253135|  8|  3| 11|
| 235399|  8|  3| 11|
| 108865|  8|  3| 11|
```