Problem Statement :-

Analysis of Black Friday Sales

The analysis focuses on the sales of different products based on their Purchases, City, Gender, Age, Marital Status using *Hadoop, map-reduce, hive, pig, mongo dB, shell-script* and used *Excel* for graphical representation.

Dataset - https://www.kaggle.com/mehdidag/black-friday

Map-reduce Patterns used :-

- Secondary Sorting pattern
- Numerical Summarizations pattern
- Counting with Counters pattern
- Filtering pattern
- Top ten pattern
- Bottom ten pattern
- Distinct pattern
- Join Pattern
- Partitioning pattern

Java Map-Reduce Programs :-

1. **Pro_ProdSumPriceCityWise** - It calculates the total purchase amount of products city wise.

    hadoop jar Pro_ProdSumPriceCityWise-0.0.1-SNAPSHOT.jar
    Project/Pro_ProdSumPriceCityWise/Driver /Project/Input/BlackFriday.csv
    /Project/Output/SecSort/Input/.

    Output :-

    We can see for productid P00000142 city = A collected 2700335$ of purchase amount, city = B collected 10192027$ of purchase amount whereas city = C collected 22784190$

2. **Pro_SecSortProdPrice** - Secondary Sorting pattern. It uses the output file produced by the Pro_ProdSumPriceCityWise map-reduce program and provides output for each product with city having highest purchase total at top and lowest at bottom.

hadoop jar Pro_SecSortProdPrice-0.0.1-SNAPSHOT.jar Project/Pro_SecSortProdPrice/Driver /Project/Output/SecSort/Input/part-r-00000 /Project/Output/SecSort/Output/.

Output

```
akash@akash-VirtualBox:~/BigData/Project_jar$ hadoop fs -cat /Project/Output/SecSort/Output/* | head -20
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.security.authentication.util.KerberosUtil (file:/usr/l
b5.Config.getInstance()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.security.authentication.util.K
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
prodID=P00000142, city=C, Amount=22784190|
prodID=P00000142, city=B, Amount=10192027|
prodID=P00000142, city=A, Amount=2700335|
prodID=P00000242, city=C, Amount=67942452|
prodID=P00000242, city=B, Amount=51435388|
prodID=P00000242, city=A, Amount=36191966|
prodID=P00000342, city=C, Amount=119868254|
prodID=P00000342, city=B, Amount=102099807|
prodID=P00000342, city=A, Amount=84768684|
prodID=P00000442, city=C, Amount=174145483|
prodID=P00000442, city=B, Amount=155935863|
prodID=P00000442, city=A, Amount=137805778|
prodID=P00000542, city=C, Amount=230346271|
prodID=P00000542, city=B, Amount=211345432|
prodID=P00000542, city=A, Amount=192547543|
prodID=P00000642, city=C, Amount=302033993|
prodID=P00000642, city=B, Amount=275397576|
prodID=P00000642, city=A, Amount=251308449|
prodID=P00000742, city=C, Amount=384785625|
prodID=P00000742, city=B, Amount=356713249|
cat: Unable to write to output stream.
akash@akash-VirtualBox:~/BigData/Project_jar$
```

We can see that for ProductID = P00000142 , City = C has highest purchases whereas City = A has lowest.

3. **Pro_ProdSummary** – Numerical Summarizations pattern. It provides Summary of purchases for each product like Average, Mean, Standard deviation, Max, Min, Total and count of that product.

hadoop jar Pro_ProdSummary-0.0.1-SNAPSHOT.jar Project/Pro_ProdSummary/Driver /Project/Input/BlackFriday.csv /Project/Output/ProdSummary/.

```
akash@akash-VirtualBox:~/BigData/Project_jar$ hadoop fs -cat /Project/Output/ProdSummary/* | head -20
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.security.authentication.util.KerberosUtil (file:/usr/local/bi
b5.Config.getInstance()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.security.authentication.util.Kerberos
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
        0.0      0.0     NaN      0.0      0.0      0.0      1
P00000142        11143.507079646017        10925.0 2307.94279510495        13716.0 2725.0  1.2592163E7        1130
P00000242        10552.293800539084        10023.0 3179.231273299195       16503.0 3214.0  3914901.0          371
P00000342        5299.928571428572         5286.0  1834.6178297875344      8896.0  1727.0  1261383.0          238
P00000442        4795.358695652174         5194.0  1664.088124921866       8891.0  1764.0  441173.0           92
P00000542        5419.308219178082         5375.0  1742.058891142271       8903.0  1730.0  791219.0           146
P00000642        14913.23828125  15510.0 3633.1425203600147       19705.0 3950.0  7635578.0          512
P00000742        6033.441176470588         5444.0  1631.819453430037       8902.0  1723.0  1435959.0          238
P00000842        10008.722222222223        9987.0  2664.654462843017       13322.0 3347.0  360314.0           36
P00000942        10614.5 9921.0  4371.054282176767       19154.0 4022.0  573183.0          54
P00001042        13739.238866396761        15230.0 3346.6481344668687      19670.0 3847.0  6787184.0          494
P00001142        6681.475524475524         6973.0  1513.412060279189       8903.0  1814.0  3821804.0          572
P00001242        4255.828282828283         3735.0  2027.1855831619548      8841.0  1720.0  421327.0           99
P00001342        8498.688524590163         8387.0  2762.266065757024       13693.0 2813.0  518420.0           61
P00001442        7383.481927710844         7883.0  1617.6387968605459      9945.0  3925.0  612829.0           83
P00001542        9243.911764705883         8146.0  4743.403239300427       19411.0 3811.0  628586.0           68
P00001642        7570.851002865329         7908.0  1738.2658208521943      10075.0 2135.0  2642227.0          349
P00001742        7985.144766146993         7999.0  1801.1111411668269      10081.0 2013.0  3585330.0          449
P00001842        14420.315789473685        16180.0 4012.9336773671416      20956.0 4089.0  1095944.0          76
P00001942        4314.588516746411         3729.0  2076.661256463911       8857.0  1713.0  901749.0           209
cat: Unable to write to output stream.
akash@akash-VirtualBox:~/BigData/Project_jar$ 
```

We can see productid P00000142 on an average was purchased at 11143.50$ with max purchase amount of 13716$ and min of 2725 and so on.

4. **Pro_BestCity4Product** – It provides the city with the highest purchases for all the respective products along with the purchase amount.

hadoop jar Pro_BestCity4Product-0.0.1-SNAPSHOT.jar Project/Pro_BestCity4Product/Driver /Project/Output/SecSort/Input/part-r-00000 /Project/Output/BestCity4Prod/.

```
akash@akash-VirtualBox:~/BigData/Project_jar$ hadoop fs -cat /Project/Output/BestCity4Prod/* | tail -20
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.security.authentication.util.KerberosUtil (file:/u
b5.Config.getInstance()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.security.authentication.ut
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
P0097942        C       27941735423867
P0098042        C       27956730094720
P0098142        C       27971725666701
P0098242        C       27986727653943
P0098342        C       28001738997116
P0098442        C       28016756229612
P0098542        C       28031775068803
P0098642        C       28046794802707
P0098742        C       28061815448422
P0098842        B       28071829575603
P0098942        C       28086851436144
P0099042        C       28101875303001
P0099142        C       28116900038153
P0099242        C       28131928515571
P0099342        C       28146964741095
P0099442        C       28162009560687
P0099642        C       28177057207071
P0099742        C       28192106784680
P0099842        C       28207158756398
P0099942        C       28222211717239
akash@akash-VirtualBox:~/BigData/Project_jar$
```

For ProductId P0097942 was very popular in City C.

5. **getDetailsOfProduct.sh** – It is a shell-script that takes input ProductID from user and uses the output file generated from above map-reduce programs to give the summary details of the given Product.

```
akash@akash-VirtualBox:~/BigData/Project_jar$ cat getDetailsOfProduct.sh
echo "Enter the product id"
read prodID
aaa=`hadoop fs -cat /Project/Output/ProdSummary/* | grep $prodID`
bbb=`hadoop fs -cat /Project/Output/SecSort/Output/* | grep $prodID`
ccc=`hadoop fs -cat /Project/Output/BestCity4Prod/* | grep $prodID`
if [ $? -eq 0 ]
then
    echo " "
    echo "$(tput setaf 1)--------------------------------Summarry of Product-------------------------------------"
    echo " "
    echo "$(tput setaf 1)ProductID Avg_Price         Median  StdDev_in_Price    Max     Min     Total    Count"
    echo $aaa
    echo " "
    echo "-----------------------------City Wise Collection---------------------------------"
    echo " "
    echo $bbb
    echo " "
    echo "----------------------Best City for Sale of this Product----------------------------"
    echo " "
    echo $ccc | awk '{print "City : " $2 ", Amount : " $3}'
else
    echo "********************Product of the given ID doesnt exist as of now*********************"
fi
akash@akash-VirtualBox:~/BigData/Project_jar$
```

Output

We can see numerical summary as well as city wise collections and the best city for selling the product with productid P00069042.

6. **Pro_SummationCounters** – Counting with Counters pattern. It uses in build counters of map-reduce frame work to give total sales collections of each city.

hadoop jar Pro_SummationCounters-0.0.1-SNAPSHOT.jar Project/Pro_SummationCounters/Driver /Project/Input/BlackFriday.csv /Project/Output/SummationCounter/.
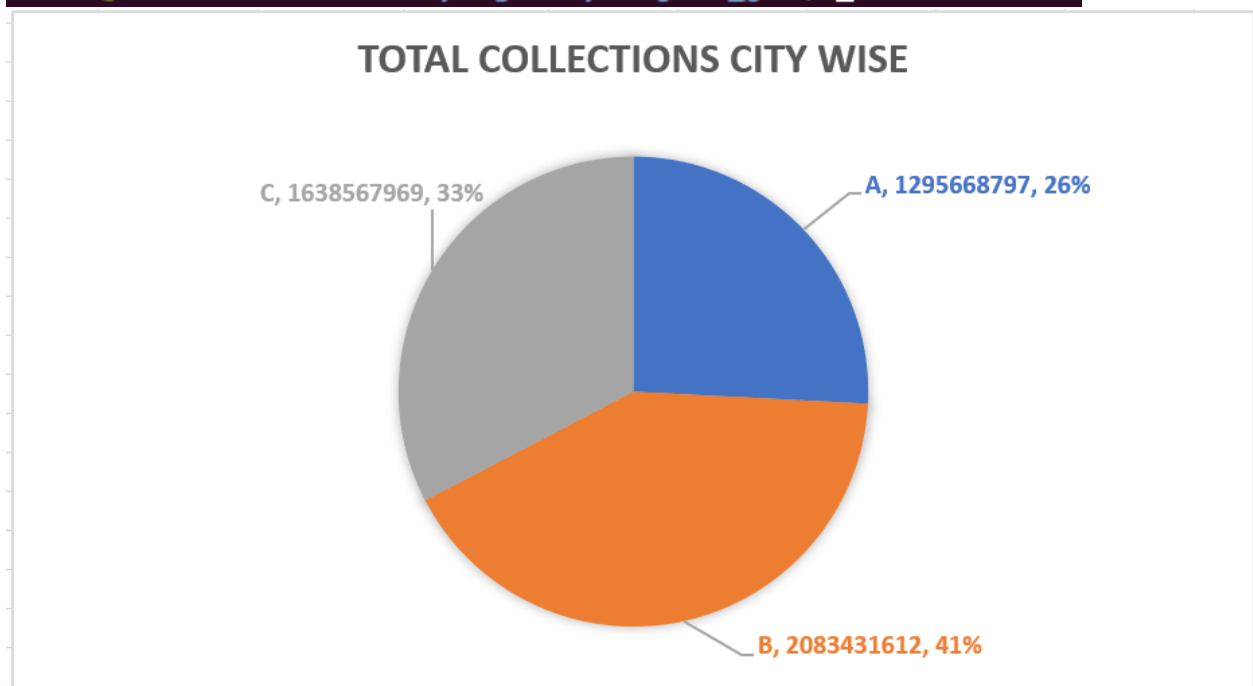
We can see city B made the highest collections during black Friday sale.

7. **Pro_PartitionerCity** – Partitioning pattern. It partitions the black Friday sales data based on the city so that we can carry out fast and detailed analysis of product city wise.

hadoop jar Pro_PartitionerCity-0.0.1-SNAPSHOT.jar Project/Pro_PartitionerCity/Driver /Project/Input/BlackFriday.csv /Project/Output/CityPartitioner/.

City A data

City B data

```
akash@akash-VirtualBox:~/BigData/Project_jar/Project-OutPut$ hadoop fs -cat /Project/Output/CityPartitioner/part-r-00001 | egrep -h "A|C" | wc -l
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.security.authentication.util.KerberosUtil (file:/usr/local/bin/hadoop-2.9.1/share/hadoop/commo
b5.Config.getInstance()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.security.authentication.util.KerberosUtil
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
0
akash@akash-VirtualBox:~/BigData/Project_jar/Project-OutPut$
```

City C data

```
akash@akash-VirtualBox:~/BigData/Project_jar$ hadoop fs -cat /Project/Output/CityPartitioner/part-r-00002 | head -20
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.security.authentication.util.KerberosUtil (file:/usr/local/bin/hadoo
b5.Config.getInstance()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.security.authentication.util.KerberosUtil
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
C       1004670,P00177242,M,36-45,5,C,2,0,12,,,712
C       1004669,P00112542,M,36-45,7,C,0,0,1,11,15,11836
C       1004669,P00178042,M,36-45,7,C,0,0,8,14,,7825
C       1004669,P00251242,M,36-45,7,C,0,0,5,11,,6942
C       1001693,P00031142,M,26-35,12,C,1,1,16,,,16162
C       1001693,P00274042,M,26-35,12,C,1,1,1,,,19054
C       1001693,P00120242,M,26-35,12,C,1,1,8,,,9894
C       1001693,P0097242,M,26-35,12,C,1,1,1,16,,19428
C       1001693,P00115642,M,26-35,12,C,1,1,8,14,,7981
C       1001693,P00252442,M,26-35,12,C,1,1,8,,,9833
C       1001693,P00351142,M,26-35,12,C,1,1,8,17,11679
C       1001693,P00289842,M,26-35,12,C,1,1,1,16,,3796
C       1000214,P00257042,M,18-25,20,C,1,0,5,14,,5182
C       1000214,P00184842,M,18-25,20,C,1,0,5,6,,3673
C       1000215,P00259942,M,36-45,14,C,1,1,16,,,20194
C       1000215,P00350442,M,36-45,14,C,1,1,2,3,15,15957
C       1000218,P00328442,M,36-45,14,C,3,1,5,14,,8709
C       1000218,P00331042,M,36-45,14,C,3,1,5,8,,7168
C       1000226,P00115542,M,36-45,1,C,1,0,8,,,9740
C       1000229,P00184942,M,18-25,10,C,1,0,1,8,17,15786
cat: Unable to write to output stream.
akash@akash-VirtualBox:~/BigData/Project_jar$ hadoop fs -lsr /Project/Output/CityPartitioner/
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.security.authentication.util.KerberosUtil (file:/usr/local/bin/hadoo
b5.Config.getInstance()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.security.authentication.util.KerberosUtil
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
lsr: DEPRECATED: Please use 'ls -R' instead.
-rw-r--r--   1 akash supergroup          0 2018-12-07 05:33 /Project/Output/CityPartitioner/_SUCCESS
-rw-r--r--   1 akash supergroup    6846799 2018-12-07 05:33 /Project/Output/CityPartitioner/part-r-00000
-rw-r--r--   1 akash supergroup   10740780 2018-12-07 05:33 /Project/Output/CityPartitioner/part-r-00001
-rw-r--r--   1 akash supergroup    7905943 2018-12-07 05:33 /Project/Output/CityPartitioner/part-r-00002
akash@akash-VirtualBox:~/BigData/Project_jar$
```

```
akash@akash-VirtualBox:~/BigData/Project_jar/Project-OutPut$ hadoop fs -cat /Project/Output/CityPartitioner/part-r-00002 | egrep -h "A|B" | wc -l
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.security.authentication.util.KerberosUtil (file:/usr/local/bin/hadoop-2.9.1/share/hadoop/commo
b5.Config.getInstance()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.security.authentication.util.KerberosUtil
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
0
akash@akash-VirtualBox:~/BigData/Project_jar/Project-OutPut$
```

We can see that the black Friday sales data is now be partitioned into 3 files based on the City i.e. A, B and C.

Pig Map-Reduce Programs :-

1. **pig_FilterMaleFemale.pig** – Filtering pattern. It filters the male and female data and saves into separate files.

```
akash@akash-VirtualBox:~/BigData/Project_jar$ cat pig_FliterMaleFemale.pig
a = LOAD '/Project/Input/BlackFriday.csv' using PigStorage(',')
as (userID:chararray,prodID:chararray,gender:chararray,age:chararray,
    ocp:chararray,city:chararray,stay:chararray,married:int,catg1:chararray,
    catg2:chararray,catg3:chararray,price:long);

b = Filter a by gender=='M';
c = Filter a by gender=='F';

store b into '/Project/Output/Pig/Male';
store c into '/Project/Output/Pig/Female';
akash@akash-VirtualBox:~/BigData/Project_jar$
```

pig pig_FliterMaleFemale.pig

Output

Male data

```
akash@akash-VirtualBox:~/BigData/Project_jar$ hadoop fs -cat /Project/Output/Pig/Male/* | head -20
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.security.authentication.util.KerberosUtil (file:/usr/local/bin/hadoop
b5.Config.getInstance()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.security.authentication.util.KerberosUtil
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
1000002 P00285442    M    55+    16    C    4+    0    8              7969
1000003 P00193542    M    26-35  15    A    3     0    1    2         15227
1000004 P00184942    M    46-50  7     B    2     1    1    8    17   19215
1000004 P00346142    M    46-50  7     B    2     1    1    15        15854
1000004 P0097242     M    46-50  7     B    2     1    1    16        15686
1000005 P00274942    M    26-35  20    A    1     1    8              7871
1000005 P00251242    M    26-35  20    A    1     1    5    11        5254
1000005 P00014542    M    26-35  20    A    1     1    8              3957
1000005 P00031342    M    26-35  20    A    1     1    8              6073
1000005 P00145042    M    26-35  20    A    1     1    1    2    5    15665
1000007 P00036842    M    36-45  1     B    1     1    1    14   16   11788
1000008 P00249542    M    26-35  12    C    4+    1    1    5    15   19614
1000008 P00220442    M    26-35  12    C    4+    1    5    14        8584
1000008 P00156442    M    26-35  12    C    4+    1    8              9872
1000008 P00213742    M    26-35  12    C    4+    1    8              9743
1000008 P00214442    M    26-35  12    C    4+    1    8              5982
1000008 P00303442    M    26-35  12    C    4+    1    1    8    14   11927
1000009 P00135742    M    26-35  17    C    0     0    6    8         16662
1000009 P00039942    M    26-35  17    C    0     0    8              5887
1000009 P00161442    M    26-35  17    C    0     0    5    14        6973
cat: Unable to write to output stream.
akash@akash-VirtualBox:~/BigData/Project_jar$
```

Female Data

```
akash@akash-VirtualBox:~/BigData/Project_jar$ hadoop fs -cat /Project/Output/Pig/Female/* | head -20
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.security.authentication.util.KerberosUtil (file:/usr/local/bin/h.
b5.Config.getInstance()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.security.authentication.util.KerberosUti
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
1000001 P00069042    F    0-17   10    A    2     0    3              8370
1000001 P00248942    F    0-17   10    A    2     0    1    6    14   15200
1000001 P00087842    F    0-17   10    A    2     0    12             1422
1000001 P00085442    F    0-17   10    A    2     0    12   14        1057
1000006 P00231342    F    51-55  9     A    1     0    5    8    14   5378
1000006 P00190242    F    51-55  9     A    1     0    4    5         2079
1000006 P0096642     F    51-55  9     A    1     0    2    3    4    13055
1000006 P00058442    F    51-55  9     A    1     0    5    14        8851
1000010 P00085942    F    36-45  1     B    4+    1    2    4    8    16352
1000010 P00118742    F    36-45  1     B    4+    1    5    11        8886
1000010 P00297942    F    36-45  1     B    4+    1    8              5875
1000010 P00266842    F    36-45  1     B    4+    1    5              8854
1000010 P00058342    F    36-45  1     B    4+    1    3    4         10946
1000010 P00032442    F    36-45  1     B    4+    1    5              5152
1000010 P00105942    F    36-45  1     B    4+    1    5              7089
1000010 P00182642    F    36-45  1     B    4+    1    2    4    9    12909
1000010 P00186942    F    36-45  1     B    4+    1    5    12        8770
1000010 P00155442    F    36-45  1     B    4+    1    1    11   15   15212
1000010 P00221342    F    36-45  1     B    4+    1    1    2    5    15705
1000010 P00087242    F    36-45  1     B    4+    1    14             7947
cat: Unable to write to output stream.
akash@akash-VirtualBox:~/BigData/Project_jar$
```

We can see the dataset is filtered based on Gender i.e. M and F and then stored into 2 different file.

2. **pig_JoinMaleFemaleProduct.pig** – Distinct pattern and Join Pattern. It is used to get the list of products purchased by both the genders.

```
akash@akash-VirtualBox:~/BigData/Project_jar$ cat pig_JoinMaleFemaleProduct.pig
m = LOAD '/Project/Output/Pig/Male/part-m-00000'
as (userID:chararray,prodID:chararray,gender:chararray,age:chararray,
    ocp:chararray,city:chararray,stay:chararray,married:int,catg1:chararray,
    catg2:chararray,catg3:chararray,price:long);

f = LOAD '/Project/Output/Pig/Female/part-m-00000'
as (userID:chararray,prodID:chararray,gender:chararray,age:chararray,
    ocp:chararray,city:chararray,stay:chararray,married:int,catg1:chararray,
    catg2:chararray,catg3:chararray,price:long);

m1 = DISTINCT(foreach m generate prodID);
f1 = DISTINCT(foreach f generate prodID);
joinMF = JOIN m1 by prodID , f1 by prodID;

store joinMF into '/Project/Output/Pig/JoinMFProd';
akash@akash-VirtualBox:~/BigData/Project_jar$
```

pig pig_JoinMaleFemaleProduct.pig

Output

```
akash@akash-VirtualBox:~/BigData/Project_jar$ hadoop fs -cat /Project/Output/Pig/JoinMFProd/* | head -30
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.hadoop.security.authentication.util.KerberosUtil (file:/usr
b5.Config.getInstance()
WARNING: Please consider reporting this to the maintainers of org.apache.hadoop.security.authentication.util
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
P0009842        P0009842
P0009942        P0009942
P0093742        P0093742
P0093842        P0093842
P0093942        P0093942
P0094042        P0094042
P0094142        P0094142
P0094242        P0094242
P0094342        P0094342
P0094442        P0094442
P0094542        P0094542
P0094642        P0094642
P0094742        P0094742
P0094842        P0094842
P0094942        P0094942
P0095042        P0095042
P0095142        P0095142
P0095242        P0095242
P0095342        P0095342
P0095442        P0095442
P0095542        P0095542
P0095642        P0095642
P0095742        P0095742
P0095842        P0095842
P0095942        P0095942
P0096042        P0096042
P0096142        P0096142
P0096342        P0096342
P0096442        P0096442
P0096542        P0096542
cat: Unable to write to output stream.
akash@akash-VirtualBox:~/BigData/Project_jar$
```

This can be used to know which products are unisex

3. To get the total collections of the product in the descending order of their collections.

a = LOAD '/Project/Input/BlackFriday.csv' using PigStorage(',') as
(userID:chararray,prodID:chararray,gender:chararray,age:chararray,ocp:chararray,city:chararray,stay:
chararray,married:int,catg1:chararray,catg2:chararray,catg3:chararray,price:long);

b = foreach a generate prodID , price;

c = Group b by prodID;

d = foreach c Generate group, SUM(b.price) as totPrice;

e = order d by totPrice DESC;

store e into '/Project/Output/Pig/Top10Prodcuts/';

Output

We can see ProductID P00025442 has made the highest sales.

Hive map-reduce

1. **Top 10 products based on their total collections -** Top ten pattern

create table BlackFriday (UserID String, ProductID String, Gender String, Age String,
Occupation Int, City String, Stay_In_Current_City_Years String, Maritial_Status Int,
Product_Category_1 Int, Product_Category_2 Int, Product_Category_3 Int, Purchase Int) row
format delimited fields terminated by ',';

```
hive> create table BlackFriday (UserID String, ProductID String, Gender String, Age String, Occupation Int, City String, Stay_In_Current_City_Years String, Maritial_Status Int, Product_Category_1 Int, Pro
duct_Category_2 Int, Product_Category_3 Int, Purchase Int) row format delimited fields terminated by ',';
OK
Time taken: 0.805 seconds
```

LOAD Data local inpath '/home/akash/Downloads/BlackFriday.csv' overwrite into table
BlackFriday;

```
hive> LOAD Data local inpath '/home/akash/Downloads/BlackFriday.csv' overwrite into table BlackFriday;
Loading data to table default.blackfriday
OK
Time taken: 1.694 seconds
```

select Distinct productid, SUM(purchase) over (partition by productid ROWS BETWEEN
UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) as TotalPurchase from
BlackFriday order by TotalPurchase DESC limit 10;

```
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 5.89 sec   HDFS Read: 24966949 HDFS Write: 112888 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 2.93 sec   HDFS Read: 117672 HDFS Write: 416 SUCCESS
Stage-Stage-3: Map: 1  Reduce: 1   Cumulative CPU: 1.96 sec   HDFS Read: 5875 HDFS Write: 397 SUCCESS
Total MapReduce CPU Time Spent: 10 seconds 780 msec
OK
P00025442       27532426
P00110742       26382569
P00255842       24652442
P00184942       24060871
P00059442       23948299
P00112142       23882624
P00110942       23232538
P00237542       23096487
P00057642       22493690
P00010742       21865042
Time taken: 83.154 seconds, Fetched: 10 row(s)
hive>
```

To Store it in a file –
insert overwrite local directory '/home/akash/Downloads/Hive-out' select Distinct productid,
SUM(purchase) over (partition by productid ROWS BETWEEN UNBOUNDED PRECEDING
AND UNBOUNDED FOLLOWING) as TotalPurchase from BlackFriday order by
TotalPurchase DESC limit 10;

TOP 10 PRODUCTS

Product	Value
P00025442	27532426
P00110742	26382569
P00255842	24652442
P00184942	24060871
P00059442	23948299
P00112142	23882624
P00110942	23232538
P00237542	23096487
P00057642	22493690
P00010742	21865042
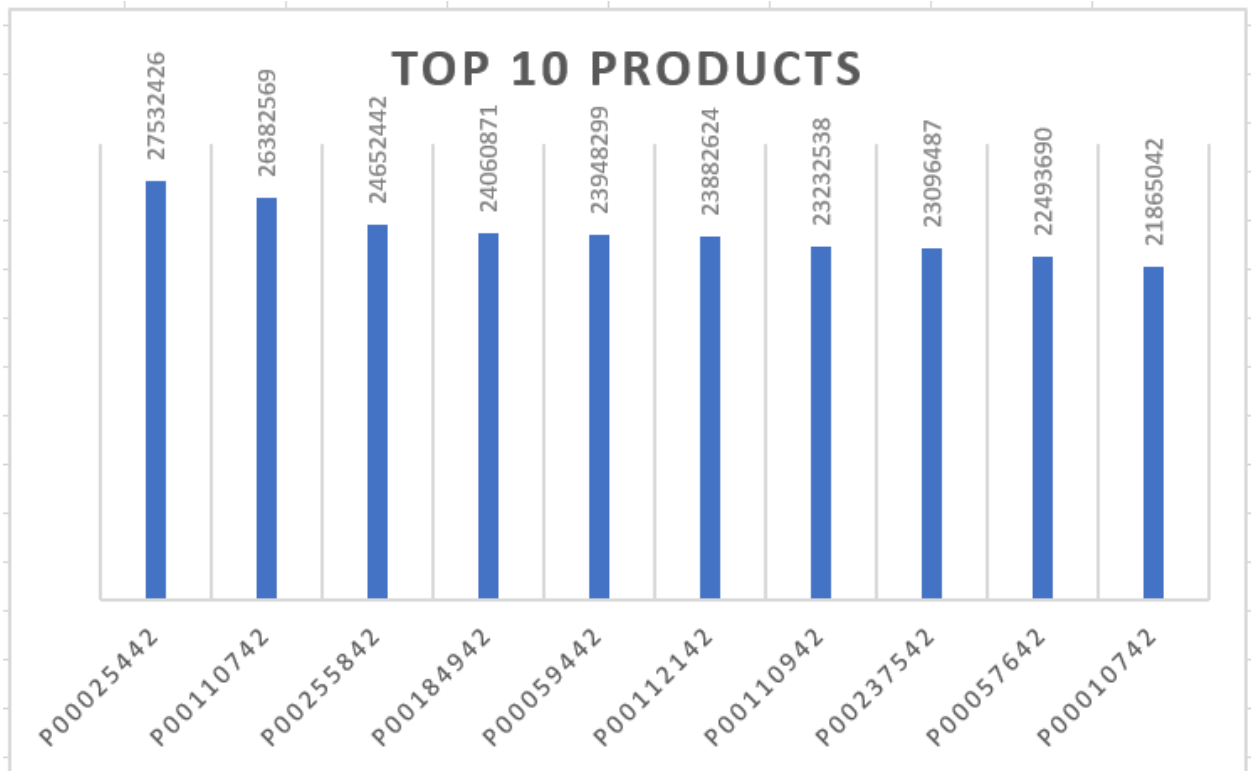
We can see ProductID P00025442 has made the highest sales.

2. **Bottom 10 products based on their total collections -** Bottom ten pattern

select Distinct productid, SUM(purchase) over (partition by productid ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) as TotalPurchase from BlackFriday order by TotalPurchase limit 10;

```
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1   Cumulative CPU: 6.36 sec   HDFS Read: 24966902 HDFS Write: 112888 SUCCESS
Stage-Stage-2: Map: 1  Reduce: 1   Cumulative CPU: 2.22 sec   HDFS Read: 117672 HDFS Write: 394 SUCCESS
Stage-Stage-3: Map: 1  Reduce: 1   Cumulative CPU: 2.17 sec   HDFS Read: 5853 HDFS Write: 354 SUCCESS
Total MapReduce CPU Time Spent: 10 seconds 750 msec
OK
Product_ID      NULL
P00091742       405
P00309042       726
P00353042       1545
P00325342       1656
P00012942       1717
P00287042       1739
P00104042       1745
P00357542       1791
P00063442       1866
Time taken: 90.003 seconds, Fetched: 10 row(s)
hive>
```

insert overwrite local directory '/home/akash/Downloads/Hive-out/Bottom/' select Distinct productid, SUM(purchase) over (partition by productid ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) as TotalPurchase from BlackFriday order by TotalPurchase limit 10;

BOTTOM 10 PRODUCTS

Product	Value
P00091742	405
P00309042	726
P00353042	1545
P00325342	1656
P00012942	1717
P00287042	1739
P00104042	1745
P00357542	1791
P00063442	1866
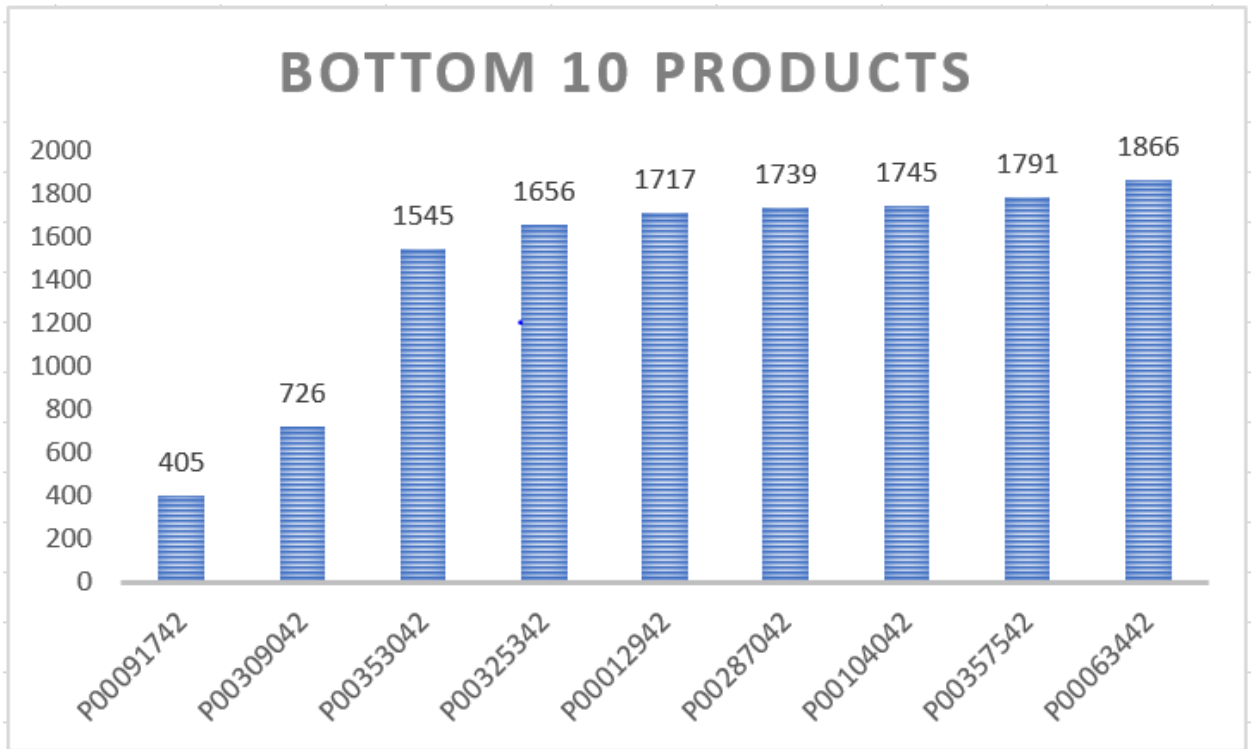
We can see ProductID P00091742 has made the lowest sales.

MongoDB map-reduce

Dumping BlackFriday.csv into mongoDB

./mongoimport --db sales --collection blackfriday --type csv --headerline --file
/c/Users/Akash/Downloads/BlackFriday.csv

```
Akash@DESKTOP-BBTTUHF MINGW64 /c/Program Files (x86)/mongodb-win32-x86_64-2008pl
us-ssl-3.4.17-12-g4fce55e175/bin
$ ./mongoimport --db sales --collection blackfriday --type csv --headerline --file /c/Users/Akash/Downloads/BlackFriday.csv
2018-12-13T09:35:30.100-0500    connected to: localhost
2018-12-13T09:35:32.083-0500    [###....................] sales.blackfriday    3.32MB/23.8MB (14.0%)
2018-12-13T09:35:35.083-0500    [#######................] sales.blackfriday    8.11MB/23.8MB (34.1%)
2018-12-13T09:35:38.084-0500    [############...........] sales.blackfriday    13.2MB/23.8MB (55.6%)
2018-12-13T09:35:41.083-0500    [#################......] sales.blackfriday    17.9MB/23.8MB (75.0%)
2018-12-13T09:35:43.937-0500    [#######################] sales.blackfriday    23.8MB/23.8MB (100.0%)
2018-12-13T09:35:43.937-0500    imported 537577 documents

Akash@DESKTOP-BBTTUHF MINGW64 /c/Program Files (x86)/mongodb-win32-x86_64-2008plus-ssl-3.4.17-12-g4fce55e175/bin
```

Dumping BlackFridayA.csv, BlackFridayB.csv, BlackFridayC.csv into mongoDB i.e. the data partitioned city wise using Pro_PartitionerCity

```
Akash@DESKTOP-BBTTUHF MINGW64 /c/Program Files (x86)/mongodb-win32-x86_64-2008pl
us-ssl-3.4.17-12-g4fce55e175/bin
$ ./mongoimport --db sales --collection blackfridayA --type csv --headerline --f
ile /c/Users/Akash/Project/Project-OutPut/CityARecords.csv
2018-12-13T11:47:10.147-0500    connected to: localhost
2018-12-13T11:47:12.129-0500    [###########...........] sales.blackfridayA 3
.51MB/6.94MB (50.5%)
2018-12-13T11:47:13.853-0500    [#######################] sales.blackfridayA 6
.94MB/6.94MB (100.0%)
2018-12-13T11:47:13.853-0500    imported 144638 documents

Akash@DESKTOP-BBTTUHF MINGW64 /c/Program Files (x86)/mongodb-win32-x86_64-2008pl
us-ssl-3.4.17-12-g4fce55e175/bin
$ ./mongoimport --db sales --collection blackfridayB --type csv --headerline --file /c/Users/Akash/Project/Project-OutPut/CityBRecords.csv
2018-12-13T11:47:37.981-0500    connected to: localhost
2018-12-13T11:47:39.969-0500    [########...............] sales.blackfridayB    3.90MB/10.9MB (35.8%)
2018-12-13T11:47:42.970-0500    [###################....] sales.blackfridayB    9.86MB/10.9MB (90.5%)
2018-12-13T11:47:43.473-0500    [#######################] sales.blackfridayB    10.9MB/10.9MB (100.0%)
2018-12-13T11:47:43.473-0500    imported 226493 documents

Akash@DESKTOP-BBTTUHF MINGW64 /c/Program Files (x86)/mongodb-win32-x86_64-2008plus-ssl-3.4.17-12-g4fce55e175/bin
$ ./mongoimport --db sales --collection blackfridayC --type csv --headerline --file /c/Users/Akash/Project/Project-OutPut/CityCRecords.csv
2018-12-13T11:48:01.516-0500    connected to: localhost
2018-12-13T11:48:03.509-0500    [###########...........] sales.blackfridayC    4.05MB/8.02MB (50.5%)
2018-12-13T11:48:05.634-0500    [#######################] sales.blackfridayC    8.02MB/8.02MB (100.0%)
2018-12-13T11:48:05.634-0500    imported 166446 documents

Akash@DESKTOP-BBTTUHF MINGW64 /c/Program Files (x86)/mongodb-win32-x86_64-2008plus-ssl-3.4.17-12-g4fce55e175/bin
$
```

1. **Count the number of products purchased by males and females**

Over all the cities

```
> mapfunc
function () { emit(this.Gender,1); }
> redufunc
function (key, values){
var count = Array.sum(values);
return count;
}
> db.blackfriday.mapReduce(mapfunc,redufunc,{out:"ProductCount4MaleFemale"});
{
        "result" : "ProductCount4MaleFemale",
        "timeMillis" : 5961,
        "counts" : {
                "input" : 537577,
                "emit" : 537577,
                "reduce" : 10551,
                "output" : 2
        },
        "ok" : 1
}
> db.ProductCount4MaleFemale.find();
{ "_id" : "F", "value" : 132197 }
{ "_id" : "M", "value" : 405380 }
>
```


NUMBER OF PRODUCTS PURCHASED

F, 132197, 25%
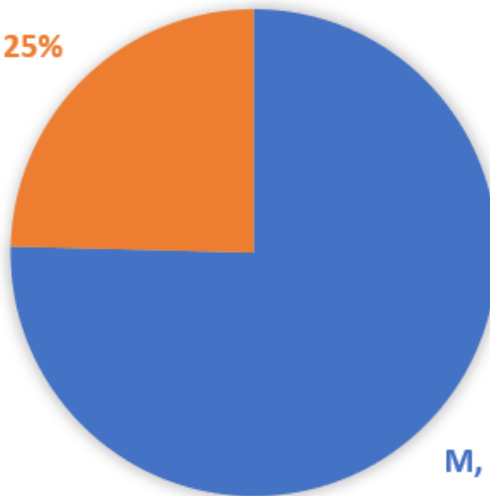
M, 405380, 75%

For City = A

```
> db.blackfridayA.mapReduce(mapfunc,redufunc,{out:"ProductCount4MaleFemale_CityA"});
{
        "result" : "ProductCount4MaleFemale_CityA",
        "timeMillis" : 711,
        "counts" : {
                "input" : 144638,
                "emit" : 144638,
                "reduce" : 2859,
                "output" : 2
        },
        "ok" : 1
}
> db.ProductCount4MaleFemale_CityA.find()
{ "_id" : "F", "value" : 34807 }
{ "_id" : "M", "value" : 109831 }
>
```

CITY A - NUMBER OF PRODUCTS PURCHASED

F, 34807, 24%

M, 109831, 76%

For City B

```
> db.blackfridayB.mapReduce(mapfunc,redufunc,{out:"ProductCount4MaleFemale_CityB"});
{
        "result" : "ProductCount4MaleFemale_CityB",
        "timeMillis" : 1122,
        "counts" : {
                "input" : 226493,
                "emit" : 226493,
                "reduce" : 4370,
                "output" : 2
        },
        "ok" : 1
}
> db.ProductCount4MaleFemale_CityB.find()
{ "_id" : "F", "value" : 56494 }
{ "_id" : "M", "value" : 169999 }
>
```
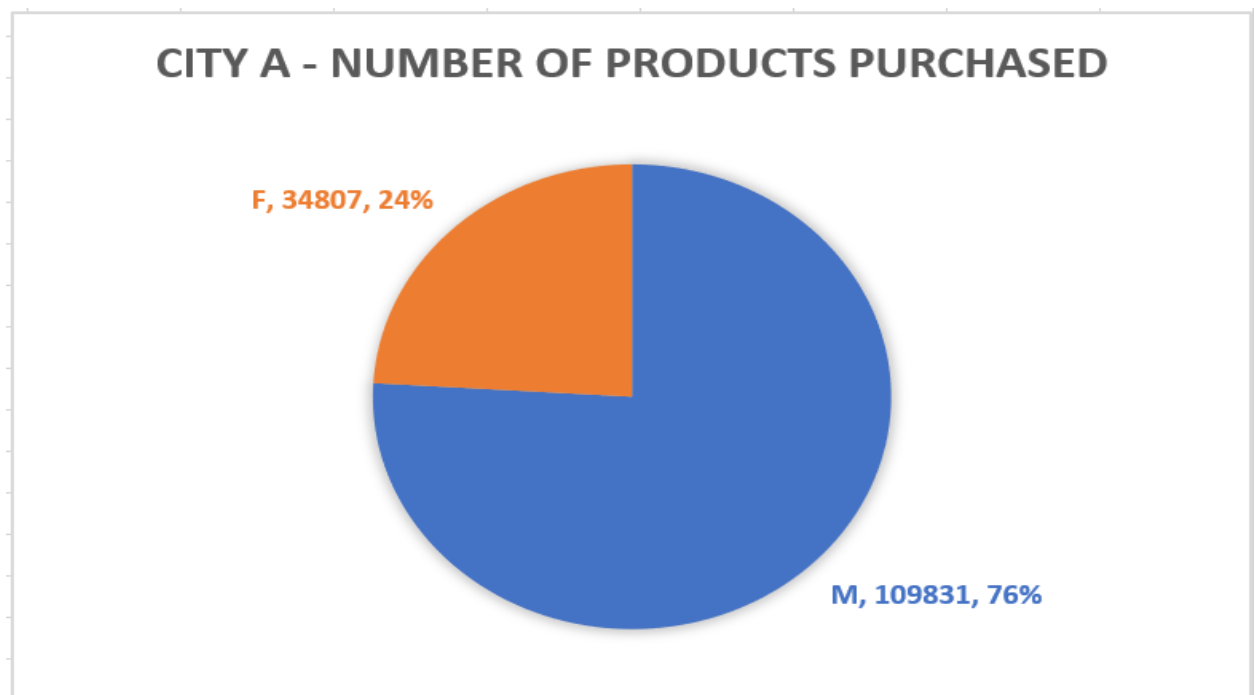
CITY B - NUMBER OF PRODUCTS PURCHASED

F, 56494, 25%

M, 169999, 75%
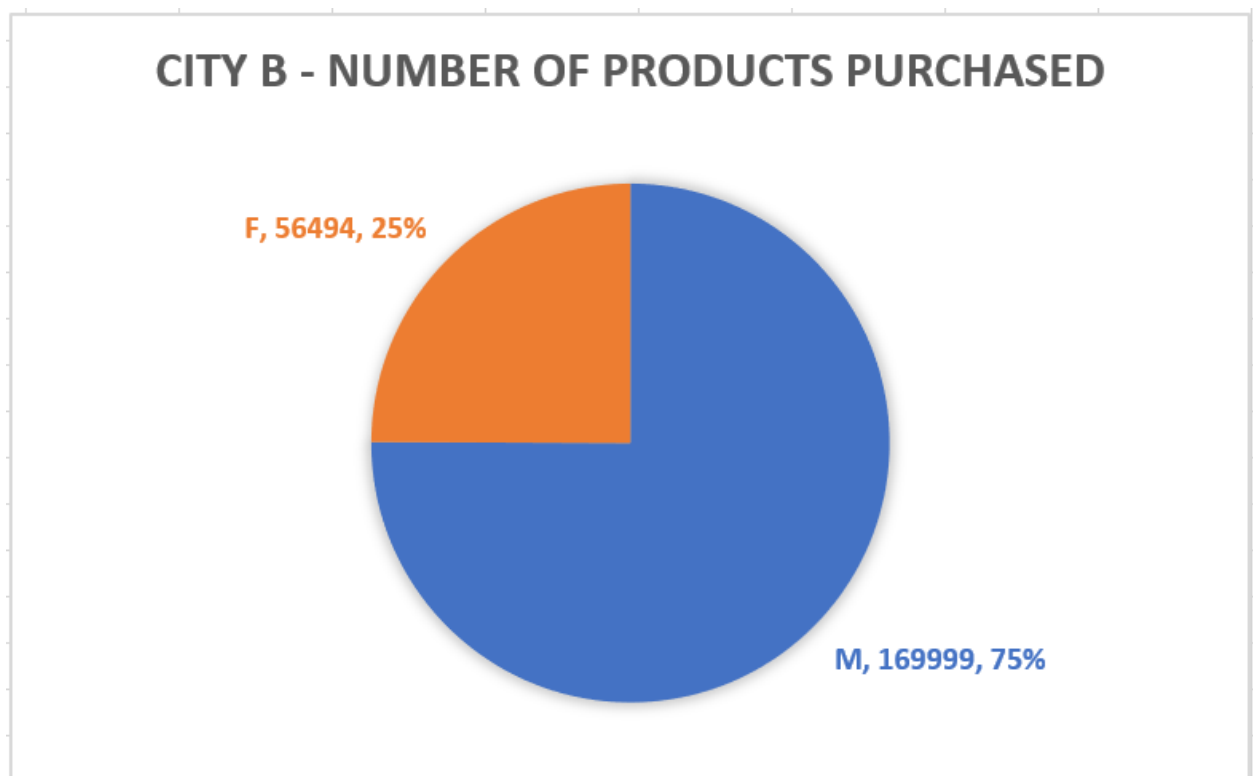
For City C

```
> db.blackfridayC.mapReduce(mapfunc,redufunc,{out:"ProductCount4MaleFemale_CityC"});
{
        "result" : "ProductCount4MaleFemale_CityC",
        "timeMillis" : 769,
        "counts" : {
                "input" : 166446,
                "emit" : 166446,
                "reduce" : 3329,
                "output" : 2
        },
        "ok" : 1
}
> db.ProductCount4MaleFemale_CityC.find()
{ "_id" : "F", "value" : 40896 }
{ "_id" : "M", "value" : 125550 }
>
```

We can see that in overall as well as city wise the percentage of products brought by males and females is almost constant i.e around M-75% and F-25%

2. Total amount spent made by males and females

Overall collections

```
> mapfunc1
function () { emit(this.Gender,this.Purchase); }
> redufunc
function (key, values){
var count = Array.sum(values);
return count;
}
> db.blackfriday.mapReduce(mapfunc1,redufunc,{out:"PuchasesMadeByMaleFemale"});
{
        "result" : "PuchasesMadeByMaleFemale",
        "timeMillis" : 7307,
        "counts" : {
                "input" : 537577,
                "emit" : 537577,
                "reduce" : 10551,
                "output" : 2
        },
        "ok" : 1
}
> db. PuchasesMadeByMaleFemale.find();
{ "_id" : "F", "value" : 1164624021 }
{ "_id" : "M", "value" : 3853044357 }
>
```
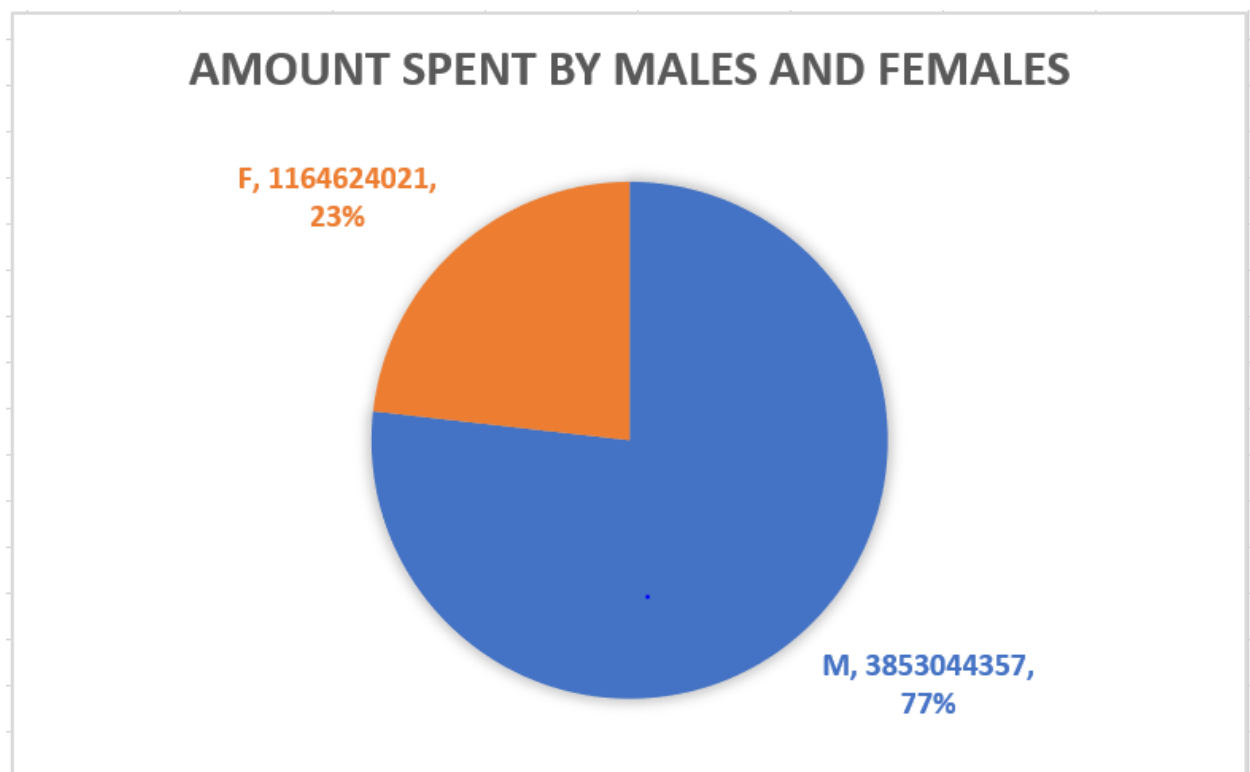

AMOUNT SPENT BY MALES AND FEMALES

F, 1164624021, 23%

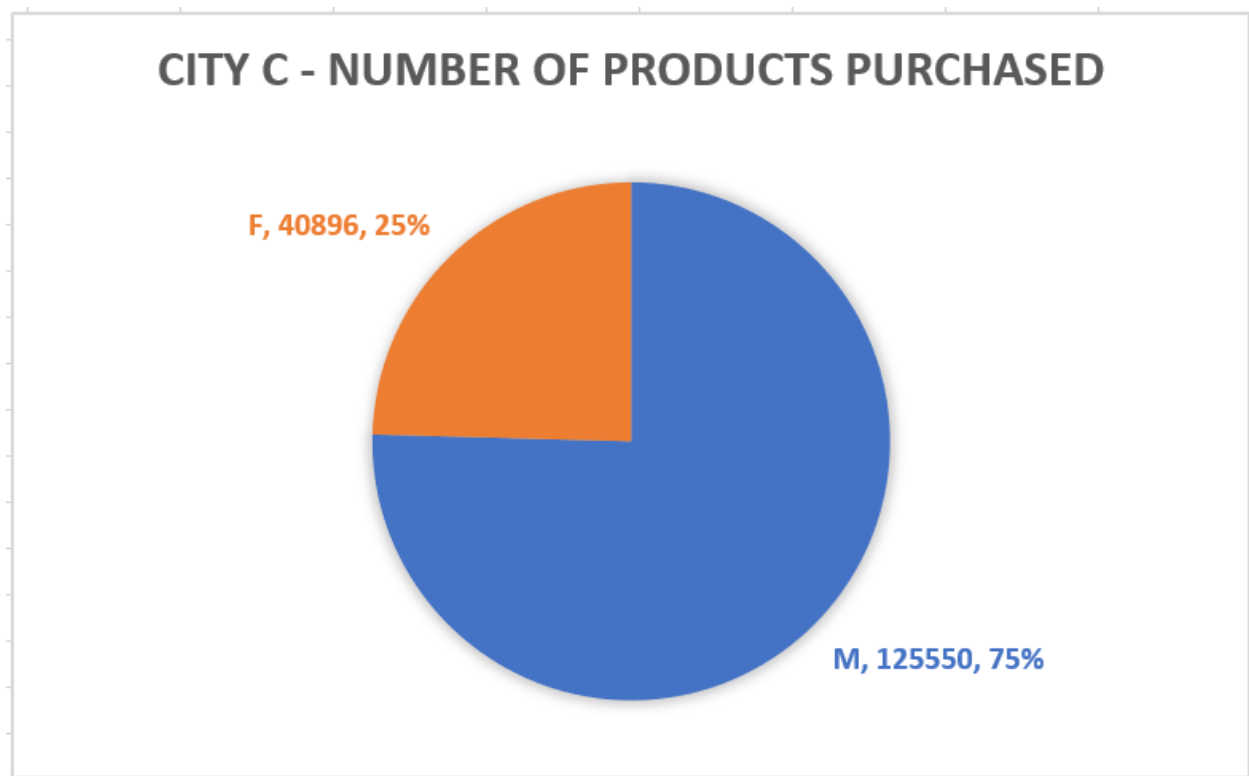M, 3853044357, 77%

In City A

```
> db.blackfridayA.mapReduce(mapfunc1,redufunc,{out:"PurchasesMadeByMaleFemale_CityA"});
{
        "result" : "PurchasesMadeByMaleFemale_CityA",
        "timeMillis" : 814,
        "counts" : {
                "input" : 144638,
                "emit" : 144638,
                "reduce" : 2859,
                "output" : 2
        },
        "ok" : 1
}
> db.PurchasesMadeByMaleFemale_CityA.find()
{ "_id" : "F", "value" : 300411276 }
{ "_id" : "M", "value" : 995257521 }
>
```

CITY A-AMOUNT SPENT BY MALES AND FEMALES

F, 300411276, 23%

M, 995257521, 77%

In City B

```
> db.blackfridayB.mapReduce(mapfunc1,redufunc,{out:"PurchasesMadeByMaleFemale_CityB"});
{
        "result" : "PurchasesMadeByMaleFemale_CityB",
        "timeMillis" : 1256,
        "counts" : {
                "input" : 226493,
                "emit" : 226493,
                "reduce" : 4370,
                "output" : 2
        },
        "ok" : 1
}
> db.PurchasesMadeByMaleFemale_CityB.find()
{ "_id" : "F", "value" : 485312751 }
{ "_id" : "M", "value" : 1598118861 }
>
```

CITY B-AMOUNT SPENT BY MALES AND FEMALES
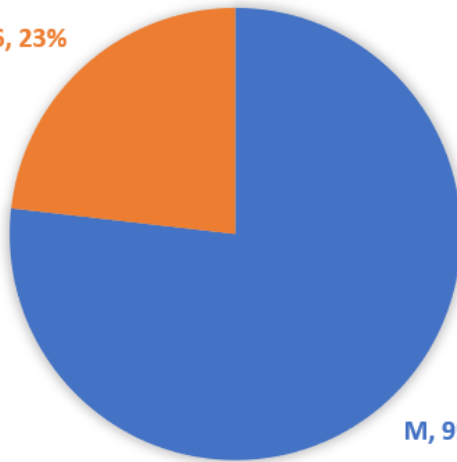

F, 485312751, 23%

M, 1598118861, 77%

In City C

```
> db.blackfridayC.mapReduce(mapfunc1,redufunc,{out:"PurchasesMadeByMaleFemale_CityC"});
{
        "result" : "PurchasesMadeByMaleFemale_CityC",
        "timeMillis" : 941,
        "counts" : {
                "input" : 166446,
                "emit" : 166446,
                "reduce" : 3329,
                "output" : 2
        },
        "ok" : 1
}
> db.PurchasesMadeByMaleFemale_CityC.find()
{ "_id" : "F", "value" : 378899994 }
{ "_id" : "M", "value" : 1259667975 }
>
```

CITY C-AMOUNT SPENT BY MALES AND FEMALES

F, 378899994, 23%
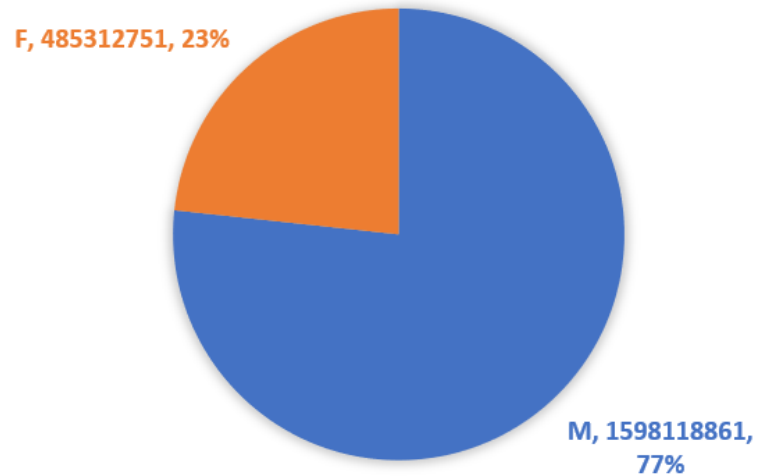
M, 1259667975, 77%
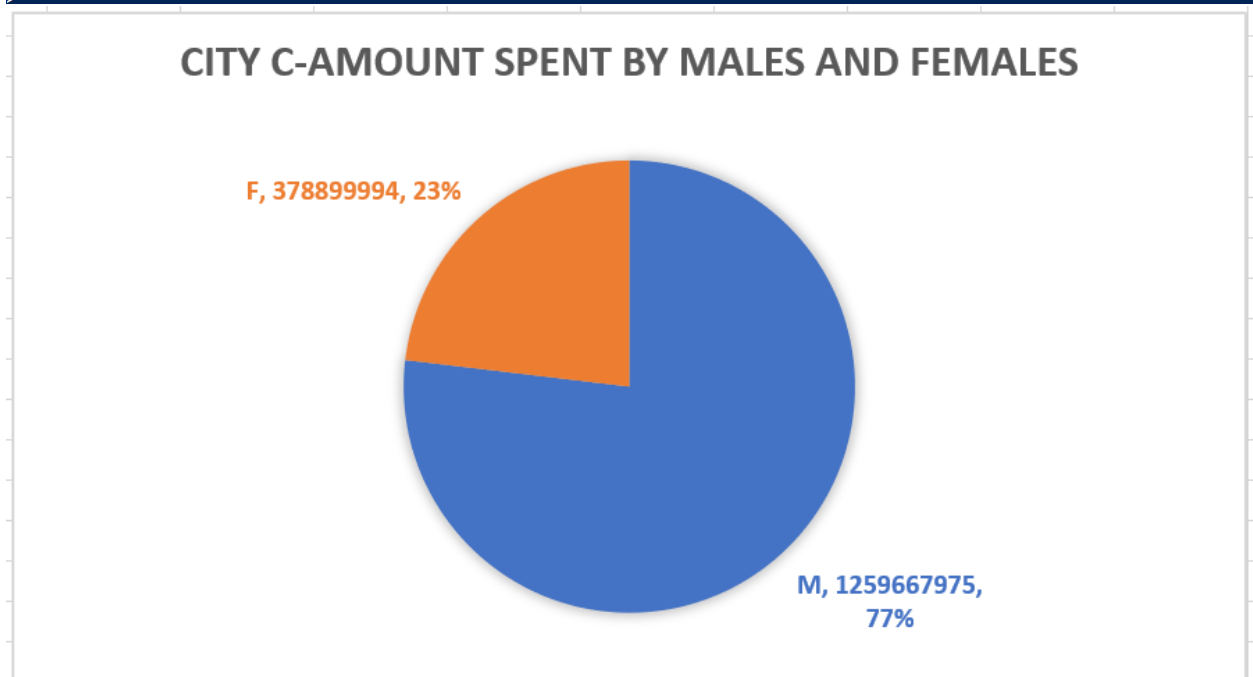
We can see Male spent more as compared to females in all the cities.

3. Total amount spent by different age groups

Overall

```
> mapfunc2
function (){emit(this.Age,this.Purchase);}
> db.blackfriday.mapReduce(mapfunc2,redufunc,{out:"AmountSpentByDiffAgeGroups"});
{
        "result" : "AmountSpentByDiffAgeGroups",
        "timeMillis" : 3196,
        "counts" : {
                "input" : 537577,
                "emit" : 537577,
                "reduce" : 26862,
                "output" : 7
        },
        "ok" : 1
}
> db.AmountSpentByDiffAgeGroups.find()
{ "_id" : "0-17", "value" : 132659006 }
{ "_id" : "18-25", "value" : 901669280 }
{ "_id" : "26-35", "value" : 1999749106 }
{ "_id" : "36-45", "value" : 1010649565 }
{ "_id" : "46-50", "value" : 413418223 }
{ "_id" : "51-55", "value" : 361908356 }
{ "_id" : "55+", "value" : 197614842 }
>
```
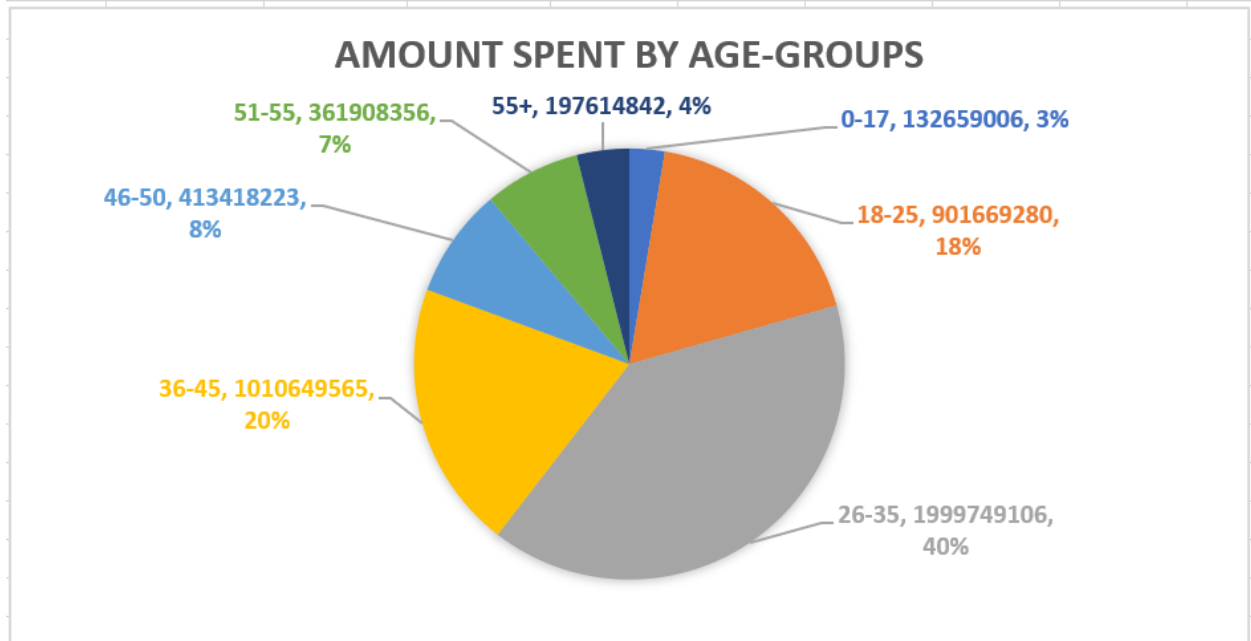
AMOUNT SPENT BY AGE-GROUPS

51-55, 361908356, 7%
55+, 197614842, 4%
0-17, 132659006, 3%
46-50, 413418223, 8%
18-25, 901669280, 18%
36-45, 1010649565, 20%
26-35, 1999749106, 40%

In City A

```
> db.blackfridayA.mapReduce(mapfunc2,redufunc,{out:"AmountSpentByDiffAgeGroups_CityA"});
{
        "result" : "AmountSpentByDiffAgeGroups_CityA",
        "timeMillis" : 859,
        "counts" : {
                "input" : 144638,
                "emit" : 144638,
                "reduce" : 6191,
                "output" : 7
        },
        "ok" : 1
}
> db.AmountSpentByDiffAgeGroups_CityA.find()
{ "_id" : "0-17", "value" : 21657219 }
{ "_id" : "18-25", "value" : 240169988 }
{ "_id" : "26-35", "value" : 647728672 }
{ "_id" : "36-45", "value" : 236366595 }
{ "_id" : "46-50", "value" : 62621435 }
{ "_id" : "51-55", "value" : 57155885 }
{ "_id" : "55+", "value" : 29969003 }
>
```
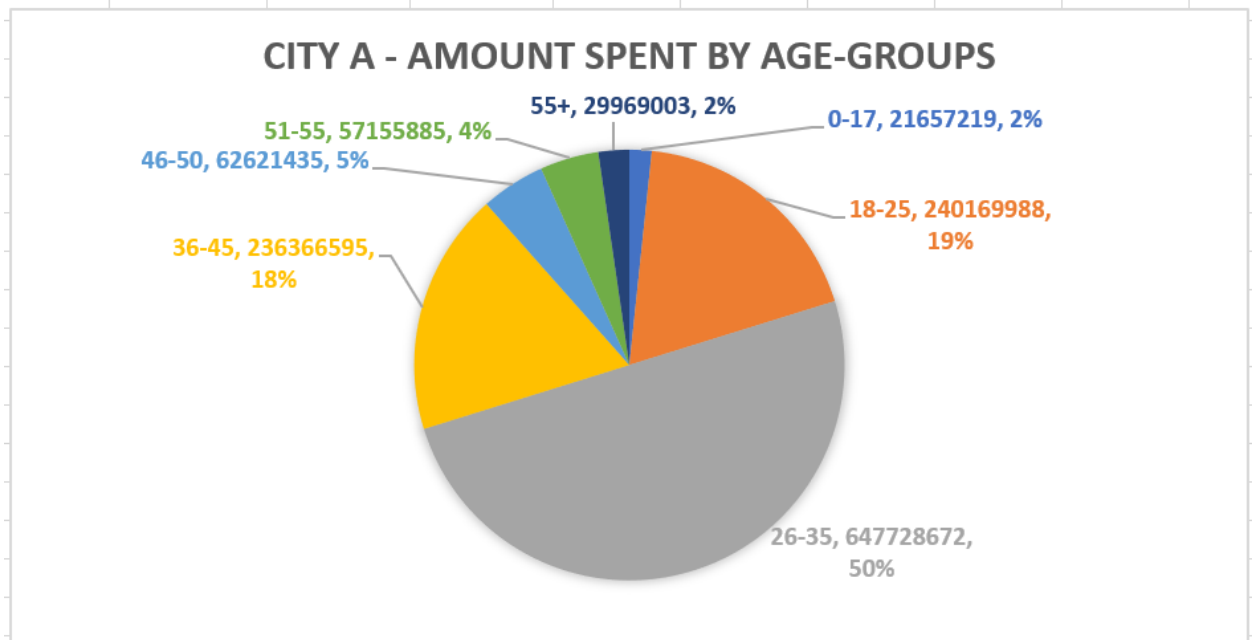
CITY A - AMOUNT SPENT BY AGE-GROUPS

55+, 29969003, 2%
0-17, 21657219, 2%
51-55, 57155885, 4%
46-50, 62621435, 5%
18-25, 240169988, 19%
36-45, 236366595, 18%
26-35, 647728672, 50%

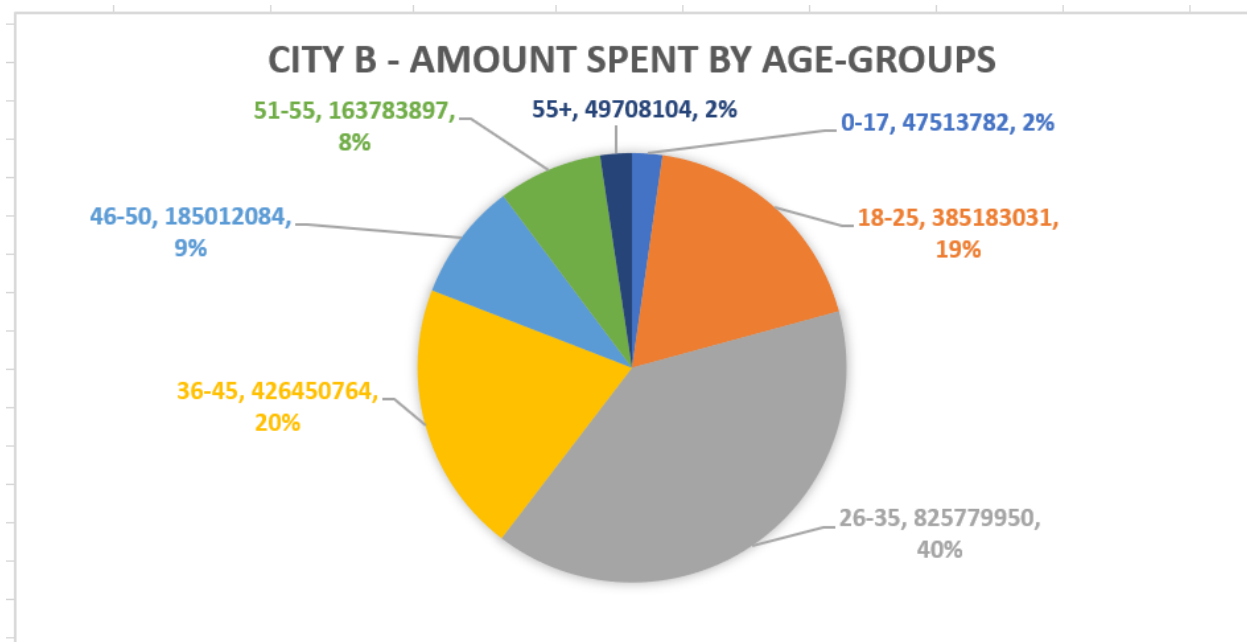In City B

```
> db.blackfridayB.mapReduce(mapfunc2,redufunc,{out:"AmountSpentByDiffAgeGroups_CityB"});
{
        "result" : "AmountSpentByDiffAgeGroups_CityB",
        "timeMillis" : 1343,
        "counts" : {
                "input" : 226493,
                "emit" : 226493,
                "reduce" : 9769,
                "output" : 7
        },
        "ok" : 1
}
> db.AmountSpentByDiffAgeGroups_CityB.find()
{ "_id" : "0-17", "value" : 47513782 }
{ "_id" : "18-25", "value" : 385183031 }
{ "_id" : "26-35", "value" : 825779950 }
{ "_id" : "36-45", "value" : 426450764 }
{ "_id" : "46-50", "value" : 185012084 }
{ "_id" : "51-55", "value" : 163783897 }
{ "_id" : "55+", "value" : 49708104 }
>
```


CITY B - AMOUNT SPENT BY AGE-GROUPS

51-55, 163783897, 8%
55+, 49708104, 2%
0-17, 47513782, 2%
46-50, 185012084, 9%
18-25, 385183031, 19%
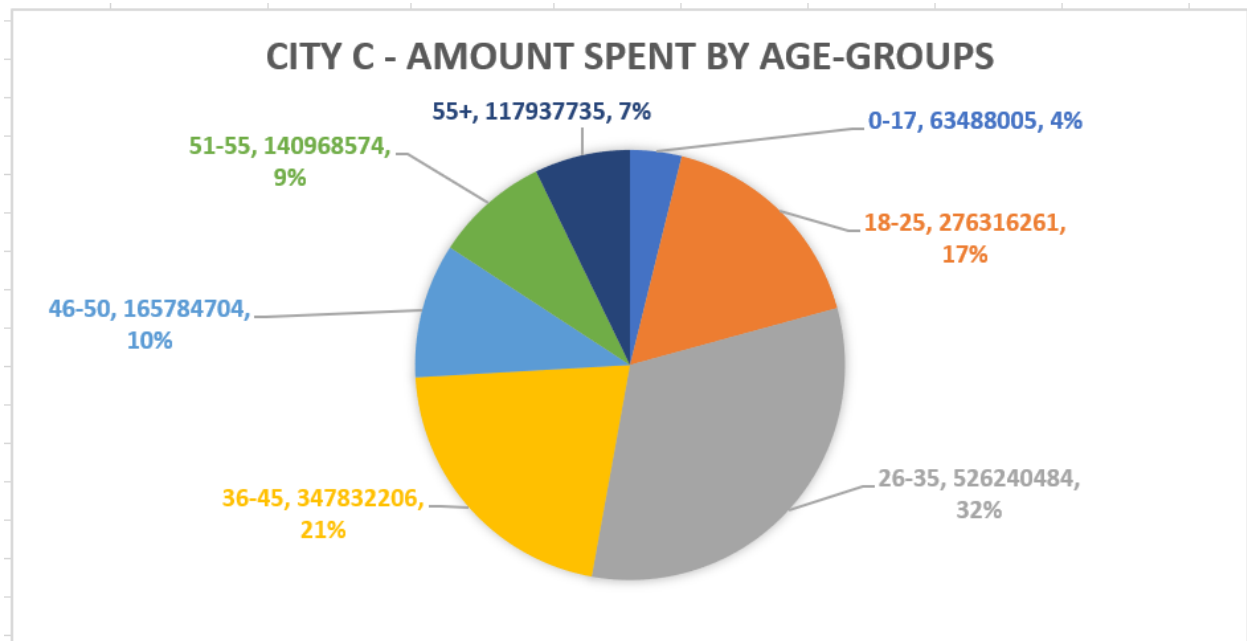36-45, 426450764, 20%
26-35, 825779950, 40%

In City C

```
> db.blackfridayC.mapReduce(mapfunc2,redufunc,{out:"AmountSpentByDiffAgeGroups_CityC"});
{
        "result" : "AmountSpentByDiffAgeGroups_CityC",
        "timeMillis" : 971,
        "counts" : {
                "input" : 166446,
                "emit" : 166446,
                "reduce" : 10538,
                "output" : 7
        },
        "ok" : 1
}
> db.AmountSpentByDiffAgeGroups_CityC.find()
{ "_id" : "0-17", "value" : 63488005 }
{ "_id" : "18-25", "value" : 276316261 }
{ "_id" : "26-35", "value" : 526240484 }
{ "_id" : "36-45", "value" : 347832206 }
{ "_id" : "46-50", "value" : 165784704 }
{ "_id" : "51-55", "value" : 140968574 }
{ "_id" : "55+", "value" : 117937735 }
>
```

CITY C - AMOUNT SPENT BY AGE-GROUPS

- 55+, 117937735, 7%
- 0-17, 63488005, 4%
- 51-55, 140968574, 9%
- 18-25, 276316261, 17%
- 46-50, 165784704, 10%
- 36-45, 347832206, 21%
- 26-35, 526240484, 32%

We can see that most of the amount came from age group 26-35 and with highest contribution from City A for this age group i.e 50%.

4. Total Amount spread by married and unmarried customers

```
> mapfunc3
function (){emit(this.Marital_Status,this.Purchase);}
> db.blackfriday.mapReduce(mapfunc3,redufunc,{out:"AmountSpentByMaritalStatus"});
{
        "result" : "AmountSpentByMaritalStatus",
        "timeMillis" : 2826,
        "counts" : {
                "input" : 537577,
                "emit" : 537577,
                "reduce" : 10718,
                "output" : 2
        },
        "ok" : 1
}
> db.AmountSpentByMaritalStatus.find()
{ "_id" : 0, "value" : 2966289500 }
{ "_id" : 1, "value" : 2051378878 }
>
```

AMOUNT SPENT AS PER MARITAL STATUS

1, 2051378878, 41%

0, 2966289500, 59%
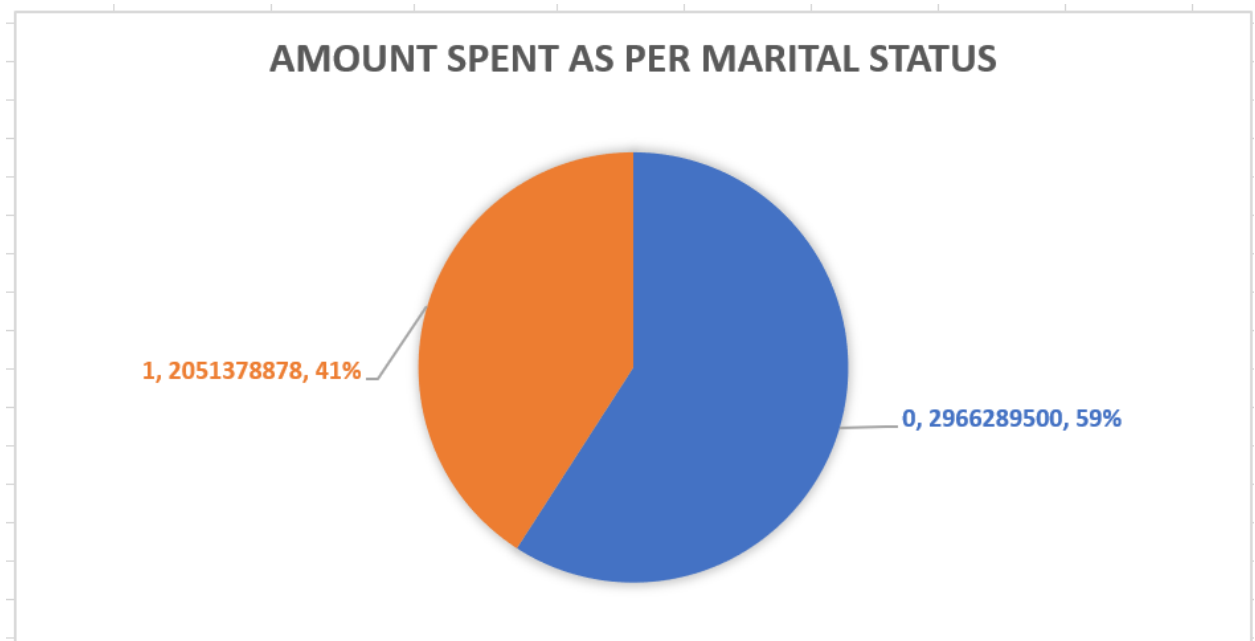
City Wise

```
> mapfunc4
function (){emit({city:this.City_Category,marital_stat:this.Marital_Status},this.Purchase);}
> db.blackfriday.mapReduce(mapfunc4,redufunc,{out:"AmountSpentByMaritalStatusCityWise"});
{
        "result" : "AmountSpentByMaritalStatusCityWise",
        "timeMillis" : 4204,
        "counts" : {
                "input" : 537577,
                "emit" : 537577,
                "reduce" : 27166,
                "output" : 6
        },
        "ok" : 1
}
> db.AmountSpentByMaritalStatusCityWise.find()
{ "_id" : { "city" : "A", "marital_stat" : 0 }, "value" : 806390391 }
{ "_id" : { "city" : "A", "marital_stat" : 1 }, "value" : 489278406 }
{ "_id" : { "city" : "B", "marital_stat" : 0 }, "value" : 1233719389 }
{ "_id" : { "city" : "B", "marital_stat" : 1 }, "value" : 849712223 }
{ "_id" : { "city" : "C", "marital_stat" : 0 }, "value" : 926179720 }
{ "_id" : { "city" : "C", "marital_stat" : 1 }, "value" : 712388249 }
>
```

We can see that unmarried people from city B have spent the most whereas married people of City A have spent the least of all

Conclusion

- City B made the highest sales collection followed by City C and City A.
- Males and Unmarried people spent more compared to Females and married people respectively.
- People belonging to 55+ Age group spent the least whereas people of age group 26-35 took more interest in buying stuff.
- Product-Id P00025442 made the highest sales collection whereas Product-Id P00091742 made the lowest.