Analysis of Big Mart Sales

Group: Starry Sky

Members: 20191003063 Vicky

20191003113 Erica

20191003208 Seven

20191002869 Lawrence

20191003140 Steve

20191002895 Lotty

20191003115 Chanera

20191003023 Russell

Professor：Kanoksak Wattanachote

Content

1. Data Information

1.1 Dataset Introduction

The data scientists at BigMart have collected 2013 sales data for 1559 products across 10 stores in different cities. Also, certain attributes of each product and store have been defined. The aim is to build a predictive model and predict the sales of each product at a particular outlet.

Therefore, we hope to get some useful information from the data

1.2 Atrribute Information

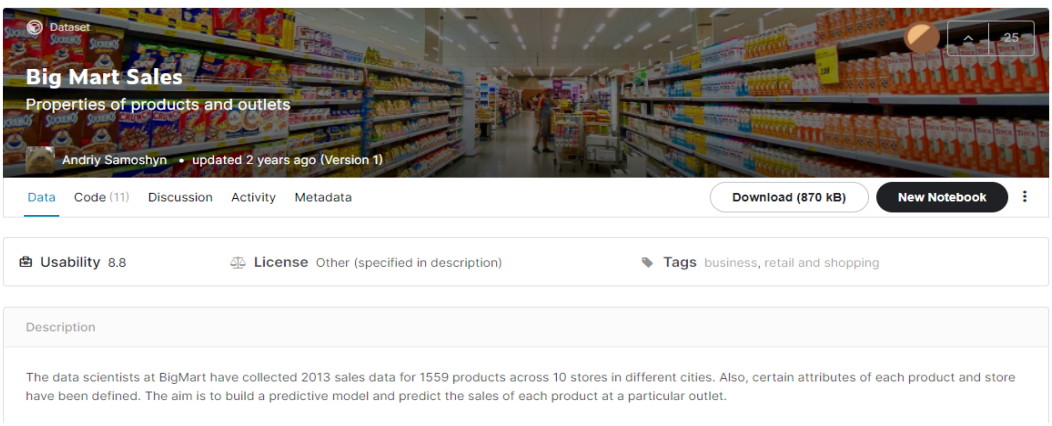
Attribute	**Meaning**	**Datatype**
Item_Identifier	Unique product ID	string
Item_Weight	Weight of product	float
Item_Fat_Content	Whether the product is low fat or not	string
Item_Visibility	The % of total display area of all products in a store allocated to the particular product	float
Item_Type	The category to which the product belongs	string
Item_MRP	Maximum Retail Price (list	float

	price) of the product	
Outlet_Identifier	Unique store ID	string
Outlet_Establishment_Year	The year in which store was established	int
Outlet_Location_Type	The type of city in which the store is located	string
Outlet_Type	Whether the outlet is just a grocery store or some sort of supermarket	string
Item_Outlet_Sales	Sales of the product in the particular store. This is the outcome variable to be predicted.	float

1.3 Data Source

Our data is from kaggle.

https://www.kaggle.com/mrmorj/big-mart-sales



2 Data Preprocessing

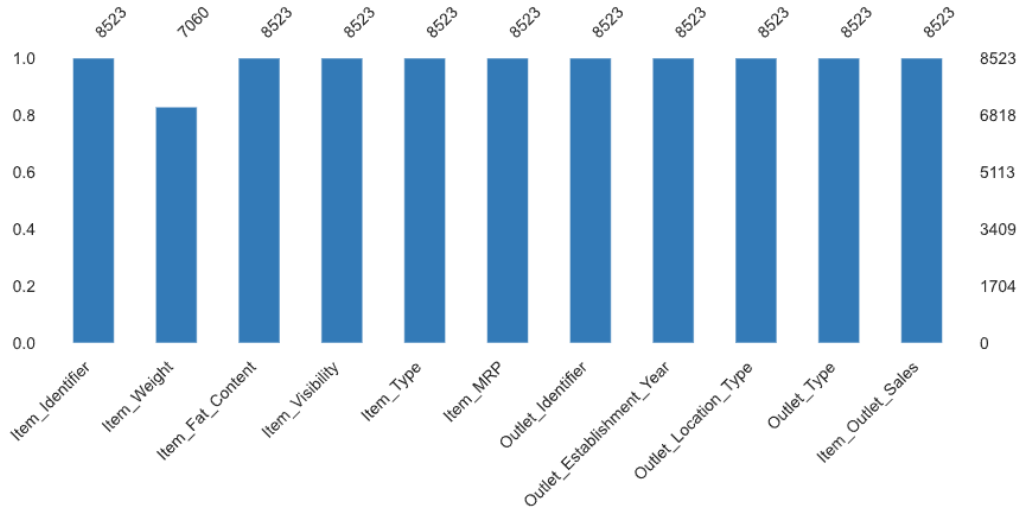
raw data

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Item_Ide	Item_Wi	Item_Fa	Item_Vis	Item_Ty	Item_MI	Outlet_I	Outlet_E	Outlet_L	Outlet_T	et_Sales		
2	FDA15	9.3	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Tier 1	Supermark	3735.138		
3	DRC01	5.92	Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Tier 3	Supermark	443.4228		
4	FDN15	17.5	Low Fat	0.01676	Meat	141.618	OUT049	1999	Tier 1	Supermark	2097.27		
5	FDX07	19.2	Regular	0	Fruits and	182.095	OUT010	1998	Tier 3	Grocery St	732.38		
6	NCD19	8.93	Low Fat	0	Household	53.8614	OUT013	1987	Tier 3	Supermark	994.7052		
7	FDP36	10.395	Regular	0	Baking Goo	51.4008	OUT018	2009	Tier 3	Supermark	556.6088		
8	FDO10	13.65	Regular	0.012741	Snack Foo	57.6588	OUT013	1987	Tier 3	Supermark	343.5528		
9	FDP10		Low Fat	0.12747	Snack Foo	107.7622	OUT027	1985	Tier 3	Supermark	4022.764		
10	FDH17	16.2	Regular	0.016687	Frozen Foo	96.9726	OUT045	2002	Tier 2	Supermark	1076.599		
11	FDU28	19.2	Regular	0.09445	Frozen Foo	187.8214	OUT017	2007	Tier 2	Supermark	4710.535		
12	FDY07	11.8	Low Fat	0	Fruits and	45.5402	OUT049	1999	Tier 1	Supermark	1516.027		
13	FDA03	18.5	Regular	0.045464	Dairy	144.1102	OUT046	1997	Tier 1	Supermark	2187.153		
14	FDX32	15.1	Regular	0.100014	Fruits and	145.4786	OUT049	1999	Tier 1	Supermark	1589.265		
15	FDS46	17.6	Regular	0.047257	Snack Foo	119.6782	OUT046	1997	Tier 1	Supermark	2145.208		
16	FDF32	16.35	Low Fat	0.068024	Fruits and	196.4426	OUT013	1987	Tier 3	Supermark	1977.426		
17	FDP49	9	Regular	0.069089	Breakfast	56.3614	OUT046	1997	Tier 1	Supermark	1547.319		
18	NCB42	11.8	Low Fat	0.008596	Health and	115.3492	OUT018	2009	Tier 3	Supermark	1621.889		
19	FDP49	9	Regular	0.069196	Breakfast	54.3614	OUT049	1999	Tier 1	Supermark	718.3982		
20	DRI11		Low Fat	0.034238	Hard Drink	113.2834	OUT027	1985	Tier 3	Supermark	2303.668		
21	FDU02	13.35	Low Fat	0.102492	Dairy	230.5352	OUT035	2004	Tier 2	Supermark	2748.422		
22	FDN22	18.85	Regular	0.13819	Snack Foo	250.8724	OUT013	1987	Tier 3	Supermark	3775.086		
23	FDW12		Regular	0.0354	Baking Goo	144.5444	OUT027	1985	Tier 3	Supermark	4064.043		
24	NCB30	14.6	Low Fat	0.025698	Household	196.5084	OUT035	2004	Tier 2	Supermark	1587.267		
25	FDC37		Low Fat	0.057557	Baking Goo	107.6938	OUT019	1985	Tier 1	Grocery St	214.3876		
26	FDR28	13.85	Regular	0.025896	Frozen Foo	165.021	OUT046	1997	Tier 1	Supermark	4078.025		
27	NCD06	13	Low Fat	0.099887	Household	45.906	OUT017	2007	Tier 2	Supermark	838.908		
28	FDV10	7.645	Regular	0.066693	Snack Foo	42.3112	OUT035	2004	Tier 2	Supermark	1065.28		
29	DRJ59	11.65	Low Fat	0.019356	Hard Drink	39.1164	OUT013	1987	Tier 3	Supermark	308.9312		
30	FDE51	5.925	Regular	0.161467	Dairy	45.5086	OUT010	1998	Tier 3	Grocery St	178.4344		
31	FDC14		Regular	0.072222	Canned	43.6454	OUT019	1985	Tier 1	Grocery St	125.8362		
32	FDV38	19.25	Low Fat	0.170349	Dairy	55.7956	OUT010	1998	Tier 3	Grocery St	163.7868		
33	NCS17	18.6	Low Fat	0.080829	Health and	96.4436	OUT018	2009	Tier 3	Supermark	2741.764		
34	FDP33	18.7	Low Fat	0	Snack Foo	256.6672	OUT018	2009	Tier 3	Supermark	3068.006		

We used Python to perform Data preprocessing. In the data preprocessing, we completed two tasks. First, we checked the missing values, and then we checked outlier, corrected outlier.

(1) Missing values

We can see just column “Item_Weight” has some missing value, then we don’t need to build a model in this report, and the data is very useful to us, so we decide to retain this data.



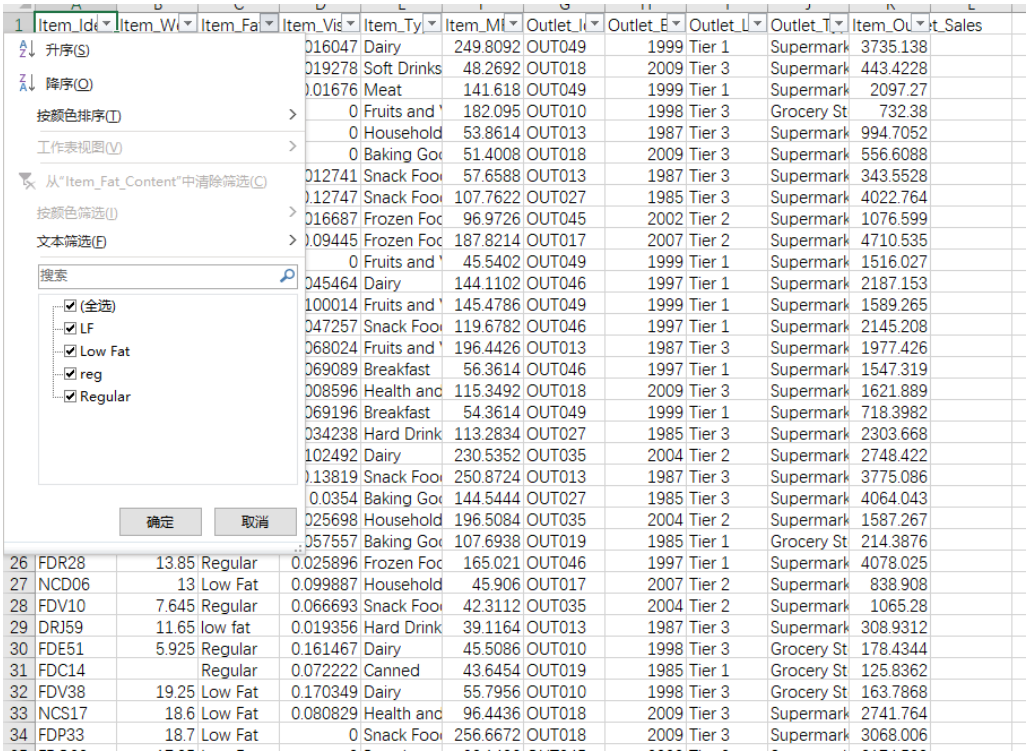
(2) Outlier

After observation, we can see the column “Item_Fat_Content” has some outliers. ‘low fat’, ‘LF’ and ‘Low Fat’ mean the same, and ‘reg’ and ‘Regular’ mean the same. So we used Python to replace ‘low fat’, ‘LF’ into ‘Low Fat’ and ‘reg’ into ‘Regular’.

Here are codes:

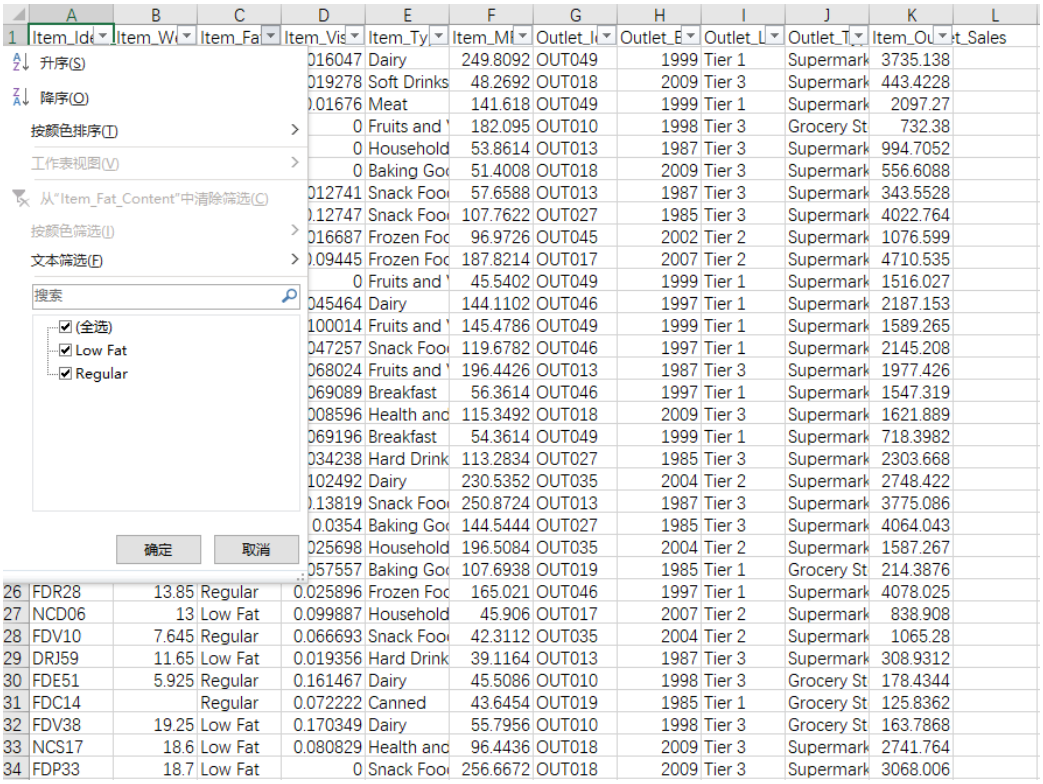
```
import pandas as pd
df = pd.read_csv("ccm.csv")
df.replace('LF', 'Low Fat', inplace=True)
df.replace('reg', 'Regular', inplace=True)
df.replace('low fat', 'Low Fat', inplace=True)
df.to_csv('ccm.csv', index=False)
```

Before processing



	Item_Ide	Item_W	Item_Fa	Item_Vis	Item_Ty	Item_MI	Outlet_I	Outlet_E	Outlet_L	Outlet_T	Item_Ou	t_Sales
1				016047	Dairy	249.8092	OUT049	1999	Tier 1	Supermark	3735.138	
				019278	Soft Drinks	48.2692	OUT018	2009	Tier 3	Supermark	443.4228	
				0.01676	Meat	141.618	OUT049	1999	Tier 1	Supermark	2097.27	
				0	Fruits and	182.095	OUT010	1998	Tier 3	Grocery St	732.38	
				0	Household	53.8614	OUT013	1987	Tier 3	Supermark	994.7052	
				0	Baking Goo	51.4008	OUT018	2009	Tier 3	Supermark	556.6088	
				012741	Snack Foo	57.6588	OUT013	1987	Tier 3	Supermark	343.5528	
				0.12747	Snack Foo	107.7622	OUT027	1985	Tier 3	Supermark	4022.764	
				016687	Frozen Foo	96.9726	OUT045	2002	Tier 2	Supermark	1076.599	
				0.09445	Frozen Foo	187.8214	OUT017	2007	Tier 2	Supermark	4710.535	
				0	Fruits and	45.5402	OUT049	1999	Tier 1	Supermark	1516.027	
				045464	Dairy	144.1102	OUT046	1997	Tier 1	Supermark	2187.153	
				100014	Fruits and	145.4786	OUT049	1999	Tier 1	Supermark	1589.265	
				047257	Snack Foo	119.6782	OUT046	1997	Tier 1	Supermark	2145.208	
				068024	Fruits and	196.4426	OUT013	1987	Tier 3	Supermark	1977.426	
				069089	Breakfast	56.3614	OUT046	1997	Tier 1	Supermark	1547.319	
				008596	Health and	115.3492	OUT018	2009	Tier 3	Supermark	1621.889	
				069196	Breakfast	54.3614	OUT049	1999	Tier 1	Supermark	718.3982	
				034238	Hard Drink	113.2834	OUT027	1985	Tier 3	Supermark	2303.668	
				102492	Dairy	230.5352	OUT035	2004	Tier 2	Supermark	2748.422	
				0.13819	Snack Foo	250.8724	OUT013	1987	Tier 3	Supermark	3775.086	
				0.0354	Baking Goo	144.5444	OUT027	1985	Tier 3	Supermark	4064.043	
				025698	Household	196.5084	OUT035	2004	Tier 2	Supermark	1587.267	
				057557	Baking Goo	107.6938	OUT019	1985	Tier 1	Grocery St	214.3876	
26	FDR28	13.85	Regular	0.025896	Frozen Foo	165.021	OUT046	1997	Tier 1	Supermark	4078.025	
27	NCD06	13	Low Fat	0.099887	Household	45.906	OUT017	2007	Tier 2	Supermark	838.908	
28	FDV10	7.645	Regular	0.066693	Snack Foo	42.3112	OUT035	2004	Tier 2	Supermark	1065.28	
29	DRJ59	11.65	low fat	0.019356	Hard Drink	39.1164	OUT013	1987	Tier 3	Supermark	308.9312	
30	FDE51	5.925	Regular	0.161467	Dairy	45.5086	OUT010	1998	Tier 3	Grocery St	178.4344	
31	FDC14		Regular	0.072222	Canned	43.6454	OUT019	1985	Tier 1	Grocery St	125.8362	
32	FDV38	19.25	Low Fat	0.170349	Dairy	55.7956	OUT010	1998	Tier 3	Grocery St	163.7868	
33	NCS17	18.6	Low Fat	0.080829	Health and	96.4436	OUT018	2009	Tier 3	Supermark	2741.764	
34	FDP33	18.7	Low Fat	0	Snack Foo	256.6672	OUT018	2009	Tier 3	Supermark	3068.006	

After processing

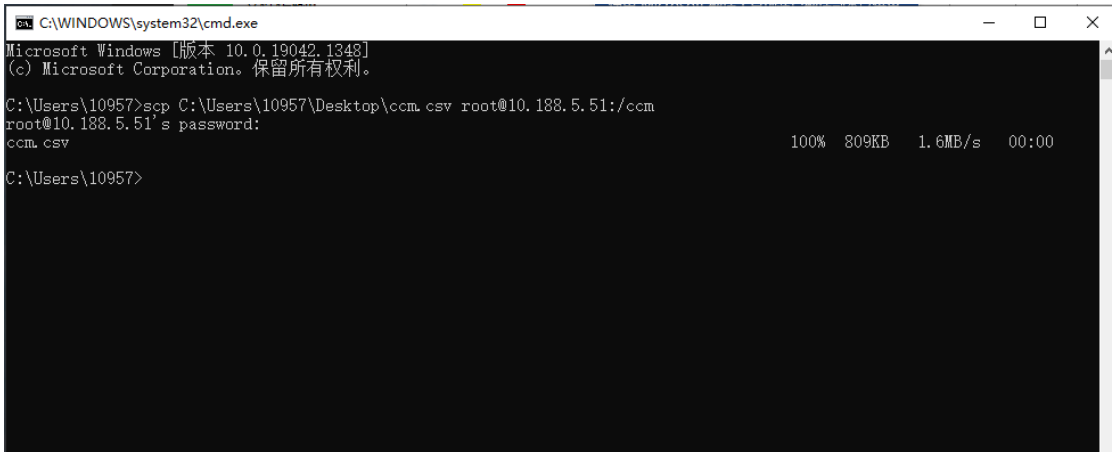


	A	B	C	D	E	F	G	H	I	J	K	L
1	Item_Ide	Item_W	Item_Fa	Item_Vis	Item_Ty	Item_MI	Outlet_I	Outlet_E	Outlet_L	Outlet_T	Item_Ou	t_Sales
				016047	Dairy	249.8092	OUT049	1999	Tier 1	Supermark	3735.138	
				019278	Soft Drinks	48.2692	OUT018	2009	Tier 3	Supermark	443.4228	
				0.01676	Meat	141.618	OUT049	1999	Tier 1	Supermark	2097.27	
				0	Fruits and	182.095	OUT010	1998	Tier 3	Grocery St	732.38	
				0	Household	53.8614	OUT013	1987	Tier 3	Supermark	994.7052	
				0	Baking Goo	51.4008	OUT018	2009	Tier 3	Supermark	556.6088	
				012741	Snack Foo	57.6588	OUT013	1987	Tier 3	Supermark	343.5528	
				0.12747	Snack Foo	107.7622	OUT027	1985	Tier 3	Supermark	4022.764	
				016687	Frozen Foo	96.9726	OUT045	2002	Tier 2	Supermark	1076.599	
				0.09445	Frozen Foo	187.8214	OUT017	2007	Tier 2	Supermark	4710.535	
				0	Fruits and	45.5402	OUT049	1999	Tier 1	Supermark	1516.027	
				045464	Dairy	144.1102	OUT046	1997	Tier 1	Supermark	2187.153	
				100014	Fruits and	145.4786	OUT049	1999	Tier 1	Supermark	1589.265	
				047257	Snack Foo	119.6782	OUT046	1997	Tier 1	Supermark	2145.208	
				068024	Fruits and	196.4426	OUT013	1987	Tier 3	Supermark	1977.426	
				069089	Breakfast	56.3614	OUT046	1997	Tier 1	Supermark	1547.319	
				008596	Health and	115.3492	OUT018	2009	Tier 3	Supermark	1621.889	
				069196	Breakfast	54.3614	OUT049	1999	Tier 1	Supermark	718.3982	
				034238	Hard Drink	113.2834	OUT027	1985	Tier 3	Supermark	2303.668	
				102492	Dairy	230.5352	OUT035	2004	Tier 2	Supermark	2748.422	
				0.13819	Snack Foo	250.8724	OUT013	1987	Tier 3	Supermark	3775.086	
				0.0354	Baking Goo	144.5444	OUT027	1985	Tier 3	Supermark	4064.043	
				025698	Household	196.5084	OUT035	2004	Tier 2	Supermark	1587.267	
				057557	Baking Goo	107.6938	OUT019	1985	Tier 1	Grocery St	214.3876	
26	FDR28	13.85	Regular	0.025896	Frozen Foo	165.021	OUT046	1997	Tier 1	Supermark	4078.025	
27	NCD06	13	Low Fat	0.099887	Household	45.906	OUT017	2007	Tier 2	Supermark	838.908	
28	FDV10	7.645	Regular	0.066693	Snack Foo	42.3112	OUT035	2004	Tier 2	Supermark	1065.28	
29	DRJ59	11.65	Low Fat	0.019356	Hard Drink	39.1164	OUT013	1987	Tier 3	Supermark	308.9312	
30	FDE51	5.925	Regular	0.161467	Dairy	45.5086	OUT010	1998	Tier 3	Grocery St	178.4344	
31	FDC14		Regular	0.072222	Canned	43.6454	OUT019	1985	Tier 1	Grocery St	125.8362	
32	FDV38	19.25	Low Fat	0.170349	Dairy	55.7956	OUT010	1998	Tier 3	Grocery St	163.7868	
33	NCS17	18.6	Low Fat	0.080829	Health and	96.4436	OUT018	2009	Tier 3	Supermark	2741.764	
34	FDP33	18.7	Low Fat	0	Snack Foo	256.6672	OUT018	2009	Tier 3	Supermark	3068.006	

3 Pig in Data Processing

3.1 Data import using Hadoop

(1) Use command ‘scp’ to put the csv file in the root package.



3.2 Using pig

(1) Import the csv file into pig

A = load 'ccm.csv' using PigStorage(',') as(

Item_Identifier: chararray,

Item_Weight: float,

Item_Fat_Content: chararray,

Item_Visibility: float,

Item_Type: chararray,

Item_MRP: float,

Outlet_Identifier: chararray,

Outlet_Establishment_Year: int,

Outlet_Location_Type: chararray,

Outlet_Type: chararray,

Item_Outlet_Sales: float

);

```
(NCI54,15.2,Low Fat,0.0,Household,110.4912,OUT017,2007,Tier 2,Supermarket Type1,1637.868)
(FDE22,9.695,Low Fat,0.029567217,Snack Foods,160.492,OUT035,2004,Tier 2,Supermarket Type1,4314.384)
(FDJ57,7.42,Regular,0.021695673,Seafood,185.3582,OUT017,2007,Tier 2,Supermarket Type1,3715.164)
(FDT08,13.65,Low Fat,0.049209192,Fruits and Vegetables,150.005,OUT035,2004,Tier 2,Supermarket Type1,2247.075)
(NCP54,15.35,Low Fat,0.035292856,Household,124.573,OUT018,2009,Tier 3,Supermarket Type2,1601.249)
(NCK53,11.6,Low Fat,0.03757414,Health and Hygiene,100.0042,OUT035,2004,Tier 2,Supermarket Type1,2976.126)
(NCQ42,20.35,Low Fat,0.0,Household,125.1678,OUT017,2007,Tier 2,Supermarket Type1,1907.517)
(FDW21,5.34,Regular,0.005997615,Snack Foods,100.4358,OUT017,2007,Tier 2,Supermarket Type1,1508.037)
(NCH43,8.42,Low Fat,0.07071203,Household,216.4192,OUT045,2002,Tier 2,Supermarket Type1,3020.0688)
(FDQ44,20.5,Low Fat,0.036133464,Fruits and Vegetables,120.1756,OUT035,2004,Tier 2,Supermarket Type1,3392.9167)
(NCN18,,Low Fat,0.124110736,Household,111.7544,OUT027,1985,Tier 3,Supermarket Type3,4138.613)
(FDB46,10.5,Regular,0.09414582,Snack Foods,210.8244,OUT018,2009,Tier 3,Supermarket Type2,2117.244)
(DRF37,17.25,Low Fat,0.08467618,Soft Drinks,263.191,OUT018,2009,Tier 3,Supermarket Type2,3944.865)
(FDN28,5.88,Regular,0.030242184,Frozen Foods,101.799,OUT035,2004,Tier 2,Supermarket Type1,515.995)
(FDW31,11.35,Regular,0.04324563,Fruits and Vegetables,199.4742,OUT045,2002,Tier 2,Supermarket Type1,2587.9646)
(FDG45,8.1,Low Fat,0.21430613,Fruits and Vegetables,213.9902,OUT010,1998,Tier 3,Grocery Store,424.7804)
(FDN58,13.8,Regular,0.05686164,Snack Foods,231.5984,OUT035,2004,Tier 2,Supermarket Type1,7182.6504)
(FDF05,17.5,Low Fat,0.026980352,Frozen Foods,262.591,OUT018,2009,Tier 3,Supermarket Type2,4207.856)
(FDR26,20.7,Low Fat,0.04280113,Dairy,178.3028,OUT013,1987,Tier 3,Supermarket Type1,2479.4392)
(FDH31,12.0,Regular,0.020407297,Meat,99.9042,OUT035,2004,Tier 2,Supermarket Type1,595.2252)
(FDA01,15.0,Regular,0.054488532,Canned,57.5904,OUT045,2002,Tier 2,Supermarket Type1,468.7232)
(FDH24,20.7,Low Fat,0.021518435,Baking Goods,157.5288,OUT018,2009,Tier 3,Supermarket Type2,1571.288)
(NCJ19,18.6,Low Fat,0.118661426,Others,58.7588,OUT018,2009,Tier 3,Supermarket Type2,858.882)
(FDF53,20.75,Regular,0.08360656,Frozen Foods,178.8318,OUT046,1997,Tier 1,Supermarket Type1,3608.636)
(FDF22,6.865,Low Fat,0.05678339,Snack Foods,214.5218,OUT013,1987,Tier 3,Supermarket Type1,2778.3833)
(FDS36,8.38,Regular,0.04698243,Baking Goods,108.157,OUT045,2002,Tier 2,Supermarket Type1,549.285)
(NCJ29,10.6,Low Fat,0.035186272,Health and Hygiene,85.1224,OUT035,2004,Tier 2,Supermarket Type1,1193.1136)
(FDN46,7.21,Regular,0.14522065,Snack Foods,103.1332,OUT018,2009,Tier 3,Supermarket Type2,1845.5977)
(DRG01,14.8,Low Fat,0.04487828,Soft Drinks,75.467,OUT046,1997,Tier 1,Supermarket Type1,765.67)
```

(2) show the structure of A

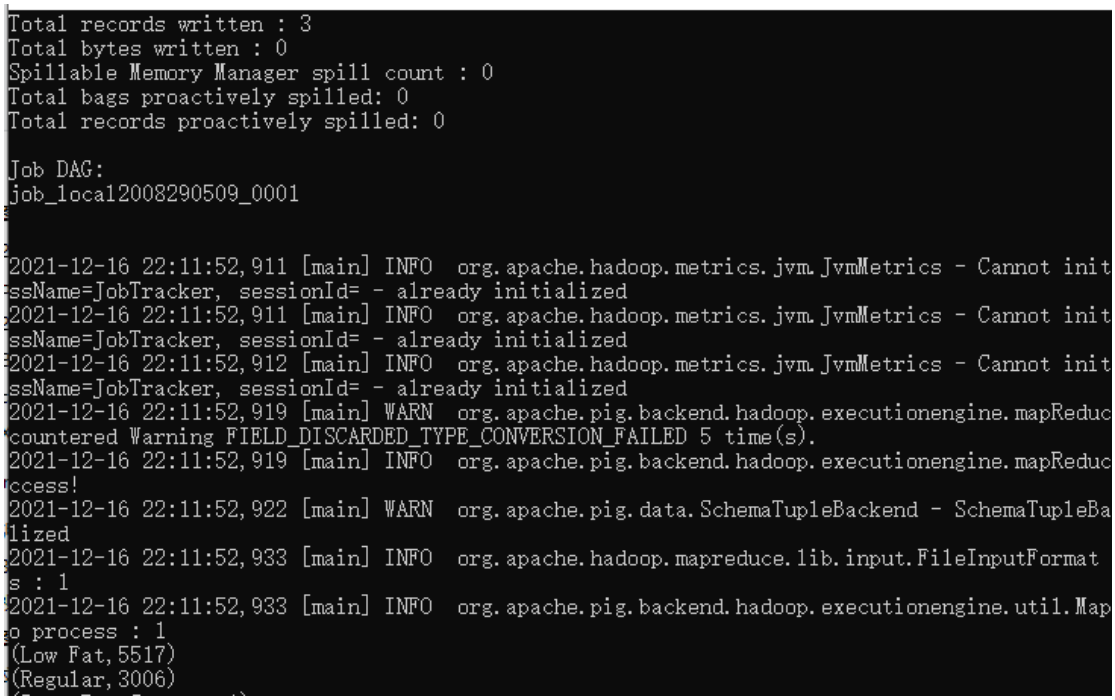
describe A;

```
grunt> describe A;
A: {Item_Identifier: chararray,Item_Weight: float,Item_Fat_Content: chararray,Item_Visibility: float,Item_Type: chararra
y,Item_MRP: float,Outlet_Identifier: chararray,Outlet_Establishment_Year: int,Outlet_Location_Type: chararray,Outlet_Typ
e: chararray,Item_Outlet_Sales: float}
```

(3) Data distribution

B = group A by Item_Fat_Content;

C = foreach B generate group, COUNT(A);



import matplotlib.pyplot as plt

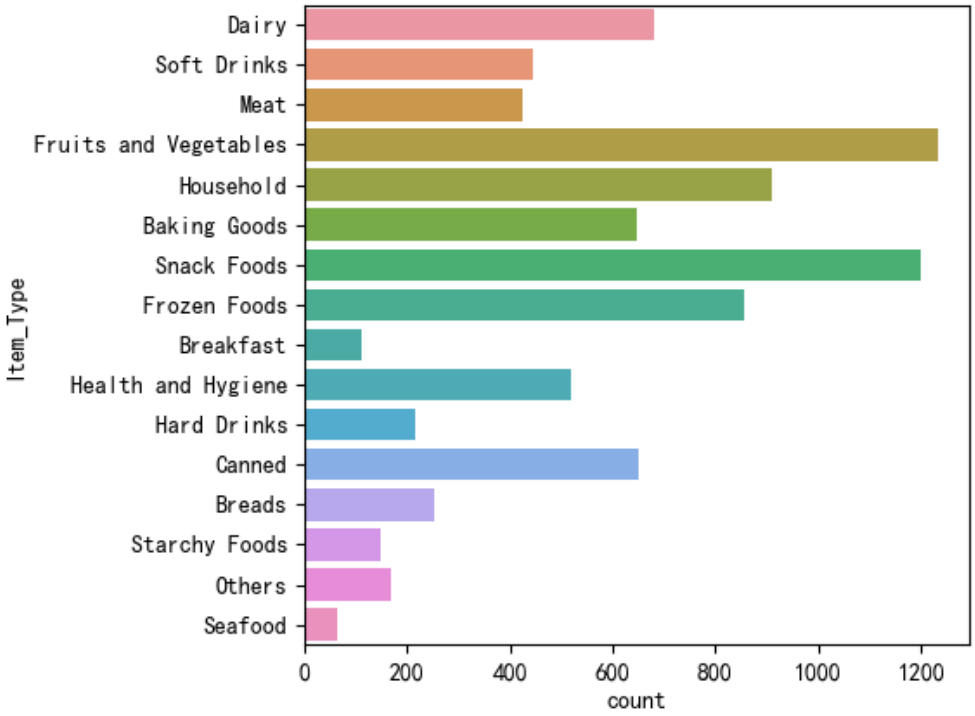
import seaborn as sns

import pandas as pd

df = pd.read_csv("ccm.csv")

sns.countplot(y=df["Item_Type "], data=df)

plt.show()



Here are the codes of python:

import matplotlib.pyplot as plt

import seaborn as sns

import pandas as pd

data = pd.read_csv("ccm.csv")

commutes = data['Item_Outlet_Sales']

commutes.plot.hist(grid=True, bins=20, rwidth=0.9, color='#607c8e')

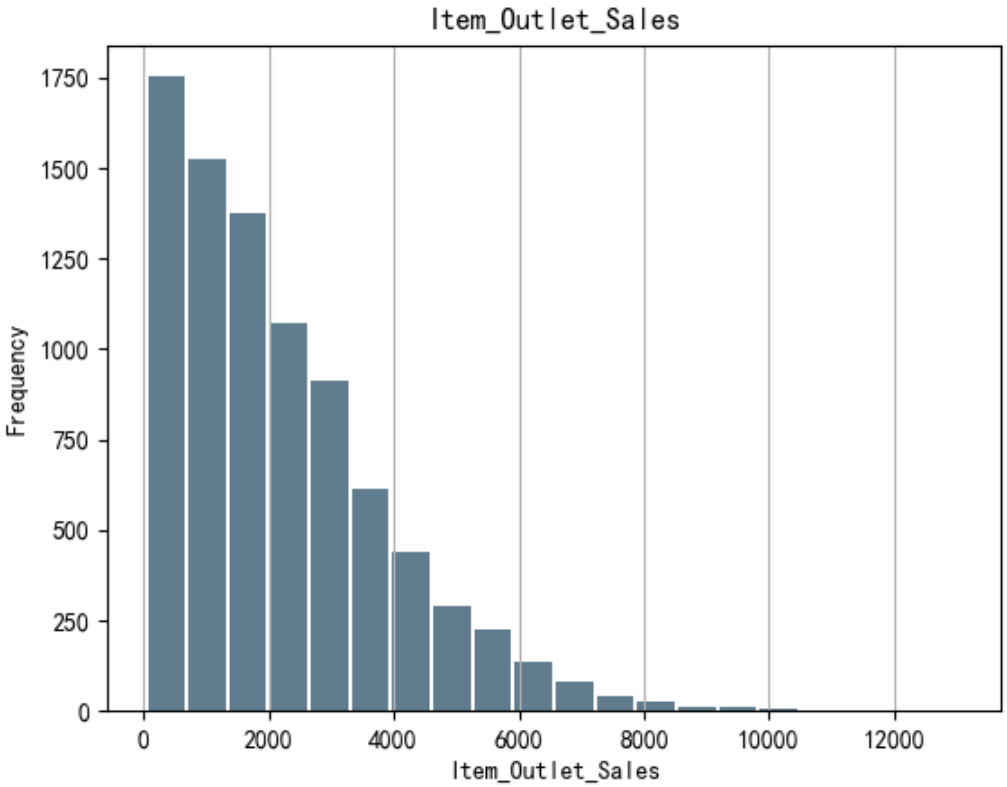
plt.title('Item_Outlet_Sales')

plt.xlabel('Item_Outlet_Sales')

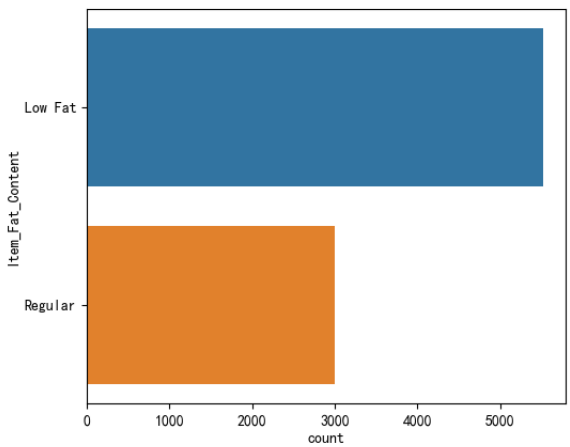
plt.ylabel('Frequency')

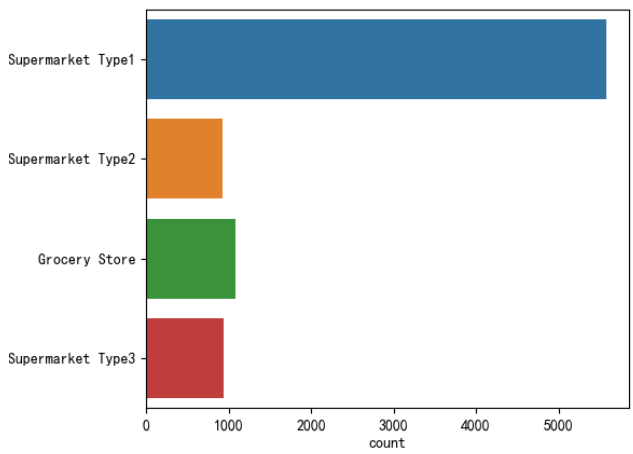
plt.grid(axis='y', alpha=0.75)

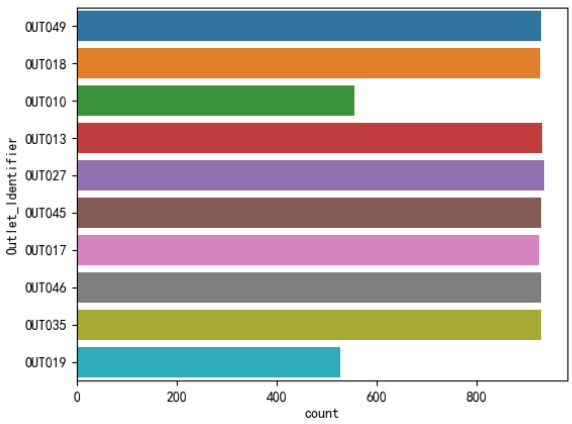
plt.show()

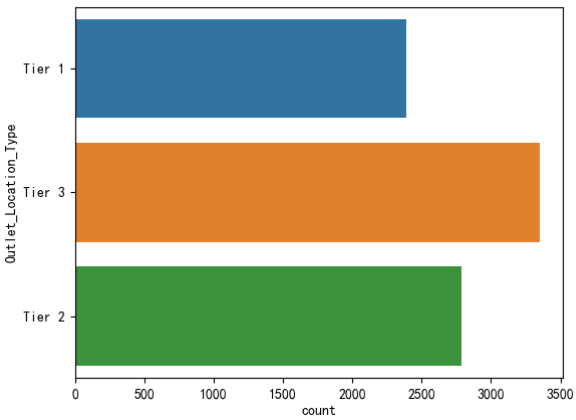


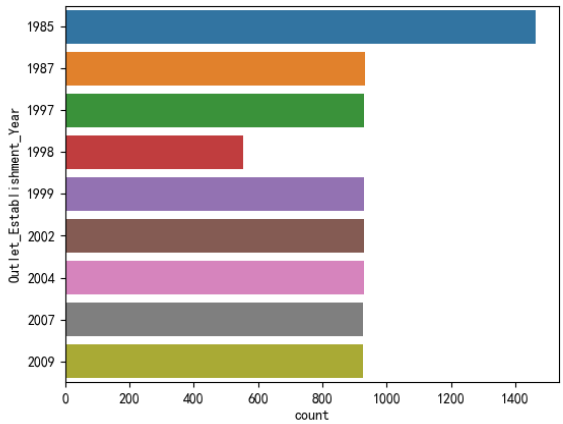
Use the same way we can get the data distribution of each column. Here are other figures.

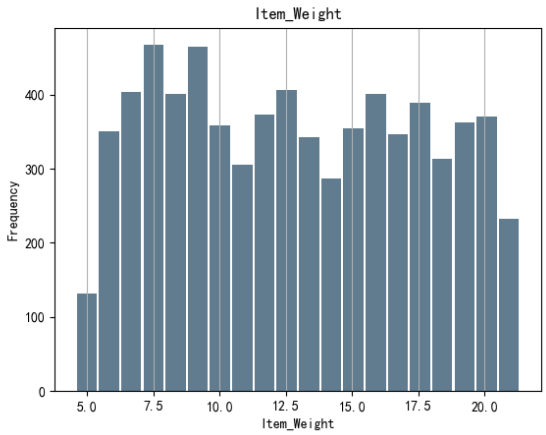


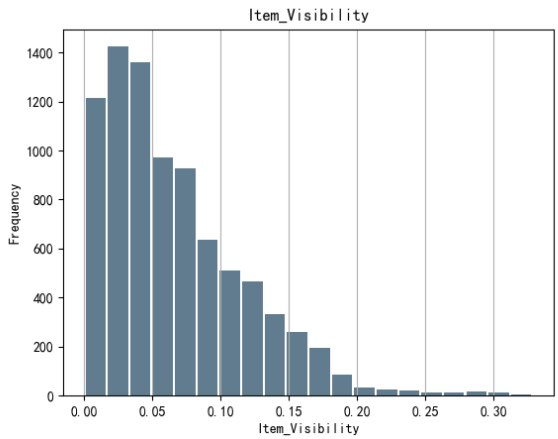


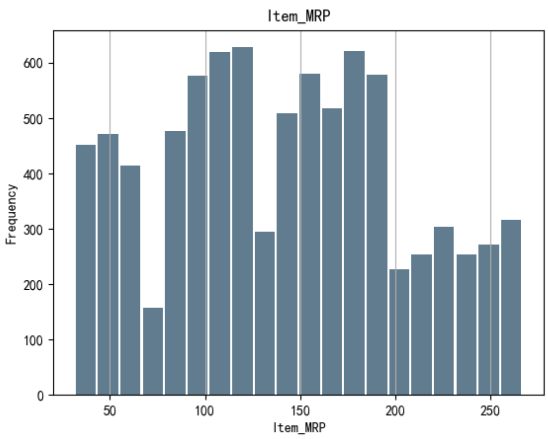

OUT049
OUT018
OUT010
OUT013
OUT027
OUT045
OUT017
OUT046
OUT035
OUT019
Outlet_Identifier
0
200
400
600
800
count


Tier 1
Tier 3
Tier 2
Outlet_Location_Type
0
500
1000
1500
2000
2500
3000
3500
count


1985
1987
1997
1998
1999
2002
2004
2007
2009
Outlet_Establishment_Year
0
200
400
600
800
1000
1200
1400
count


Item_Weight
Frequency
0
100
200
300
400
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Item_Weight


Item_Visibility
Frequency
0
200
400
600
800
1000
1200
1400
0.00
0.05
0.10
0.15
0.20
0.25
0.30
Item_Visibility


Item_MRP
Frequency
0
100
200
300
400
500
600
50
100
150
200
250
Item_MRP

(4) Retail Price Information

1. The average MPR of each item.

We can get information about the most expensive item. The most expensive item's ID is FDR25. After filtering, we can know the item is canned goods and it only sales in two types of supermarket.

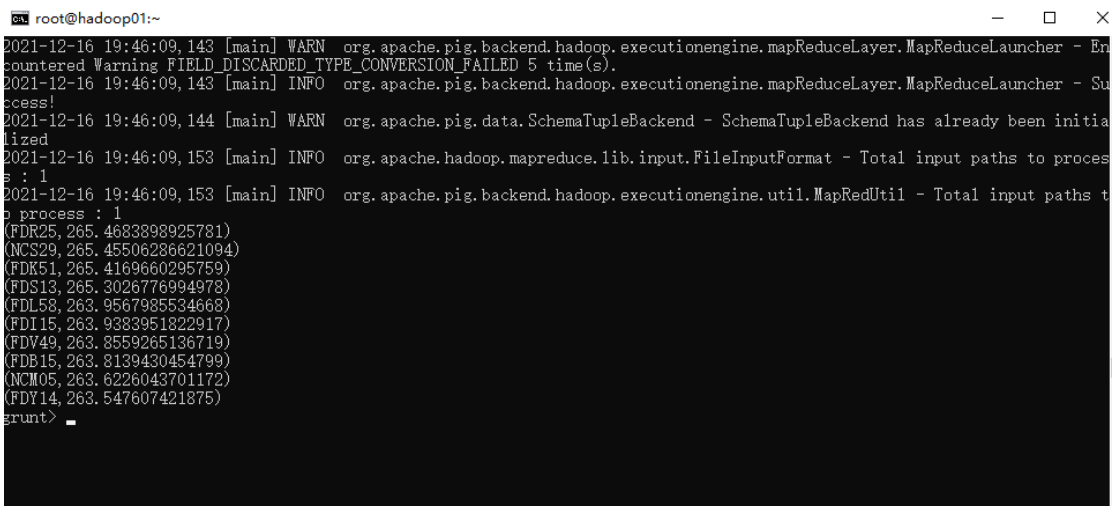
B = group A by Item_Identifier;

C = FOREACH B GENERATE group, AVG(A.Item_MRP);

D = order C by $1 DESC;

E = LIMIT D 10;

dump E

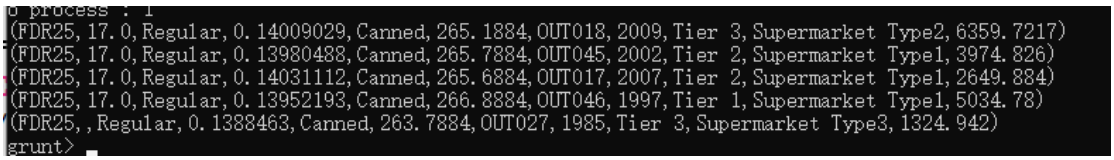


Select the most expensive item — ‘FDR25’.

B = FILTER A BY Item_Identifier == 'FDR25';

dump

B



2. The average MPR of each Item_Type.

In this part, we can get information about which type of item is more expensive. The result of analysis shown us that household goods are the most expensive in all types of item.

B = group A by Item_Type;

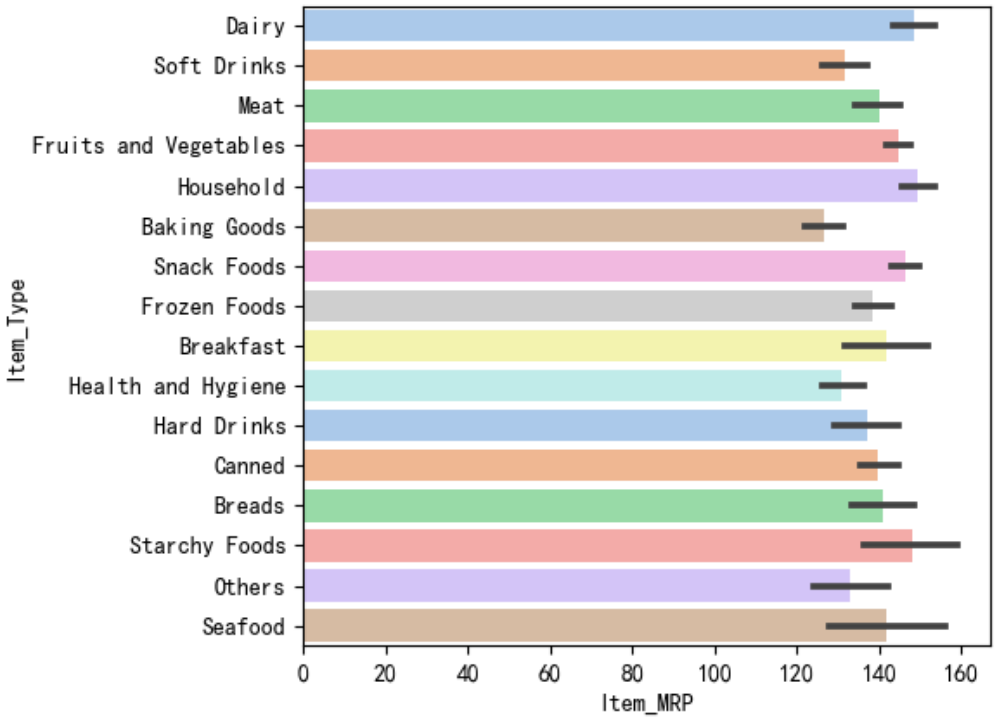
C = FOREACH B GENERATE group, AVG(A.Item_MRP);

D = order C by $1 DESC;

E = LIMIT D 10;

dump E

```
ssName=JobTracker, sessionId= - already initialized
2021-12-16 22:57:54,722 [main] INFO  org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM M
ssName=JobTracker, sessionId= - already initialized
2021-12-16 22:57:54,728 [main] INFO  org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM M
ssName=JobTracker, sessionId= - already initialized
2021-12-16 22:57:54,729 [main] INFO  org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM M
ssName=JobTracker, sessionId= - already initialized
2021-12-16 22:57:54,730 [main] INFO  org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initialize JVM M
ssName=JobTracker, sessionId= - already initialized
2021-12-16 22:57:54,735 [main] WARN  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapRe
countered Warning FIELD_DISCARDED_TYPE_CONVERSION_FAILED 5 time(s).
2021-12-16 22:57:54,735 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapRe
ccess!
2021-12-16 22:57:54,735 [main] WARN  org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has al
lized
2021-12-16 22:57:54,744 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total inpu
s : 1
2021-12-16 22:57:54,744 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - To
o process : 1
(Household,149.4247529752962)
(Dairy,148.49920726451705)
(Starchy Foods,147.83802297953014)
(Snack Foods,146.19493369420368)
(Fruits and Vegetables,144.5812345139392)
(Seafood,141.84171825647354)
(Breakfast,141.78814992037687)
(Breads,140.95266816340595)
(Meat,139.88203234504252)
(Canned,139.76383177488353)
```



3. The average MPR of each Item_Fat_Content.

Item_Fat_content means whether the product is low fat or not, so we can know the MPR of regular product and the MPR of low fat product are similar.

B = group A by Item_Fat_Content;

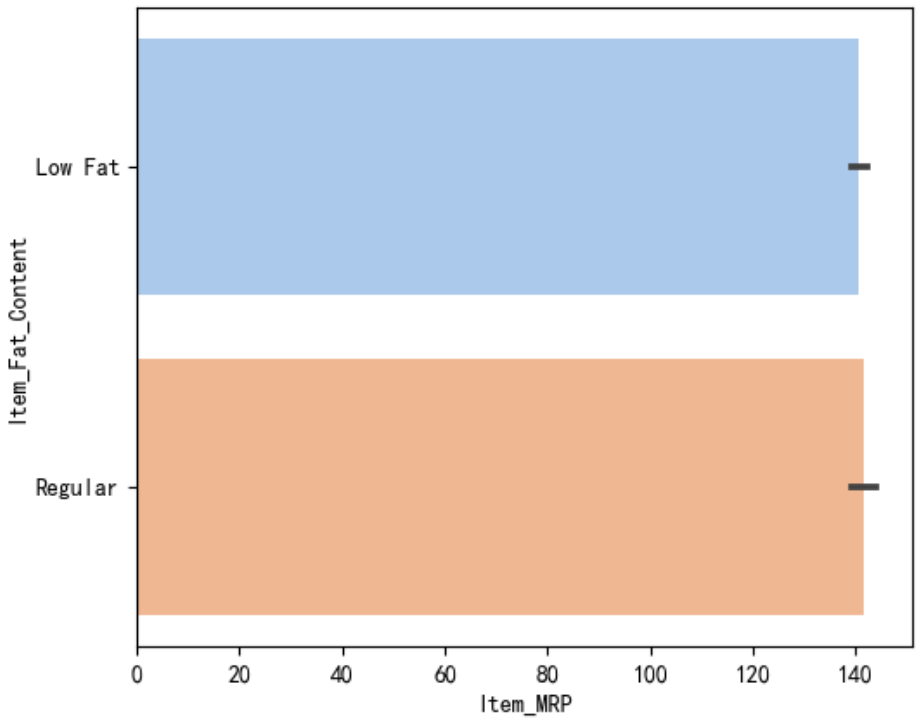
C = FOREACH B GENERATE group, AVG(A.Item_MRP);

D = order C by $1 DESC;

E = LIMIT D 10;

dump E

```
ssName=JobTracker, sessionId= - already initialized
2021-12-16 22:58:20,812 [main] INFO  org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initial
ssName=JobTracker, sessionId= - already initialized
2021-12-16 22:58:20,816 [main] INFO  org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initial
ssName=JobTracker, sessionId= - already initialized
2021-12-16 22:58:20,816 [main] INFO  org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initial
ssName=JobTracker, sessionId= - already initialized
2021-12-16 22:58:20,816 [main] INFO  org.apache.hadoop.metrics.jvm.JvmMetrics - Cannot initial
ssName=JobTracker, sessionId= - already initialized
2021-12-16 22:58:20,820 [main] WARN  org.apache.pig.backend.hadoop.executionengine.mapReduceLa
countered Warning FIELD_DISCARDED_TYPE_CONVERSION_FAILED 5 time(s).
2021-12-16 22:58:20,820 [main] INFO  org.apache.pig.backend.hadoop.executionengine.mapReduceLa
ccess!
2021-12-16 22:58:20,820 [main] WARN  org.apache.pig.data.SchemaTupleBackend - SchemaTupleBacke
lized
2021-12-16 22:58:20,828 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - T
s : 1
2021-12-16 22:58:20,828 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRed
o process : 1
(Regular,141.50425892295)
(Low Fat,140.71409781488666)
```



4. The relationship between MPR and other attributes

Using the same way to analyze and use python to draw figures.

From the analysis results, we can know Item_Weight and Item_Visibility don’t have a strong relationship with MPR. Obviously, there is a positive correlation between Item_Outlet_Sales and MPR.

Here are the code:

```
df = pd.read_csv("ccm.csv")
plt.scatter(df["Item_Outlet_Sales"], df["Item_MRP"], color="#607c8e")
plt.xlabel("Item_Outlet_Sales")
plt.ylabel("Item_MRP")
plt.show()
plt.scatter(df["Item_Weight"], df["Item_MRP"], color="#607c8e")
plt.xlabel("Item_Weight")
plt.ylabel("Item_MRP")
plt.show()
plt.scatter(df["Item_Visibility"], df["Item_MRP"], color="#607c8e")
plt.xlabel("Item_Visibility")
plt.ylabel("Item_MRP")
plt.show()
```

(5) Visibility Information

The average visibility of each item.

It can be concluded to the sales and popularity of items. In this part, we have got the average visibility of each item with pig. With these tuples, we can learn that which item would take more place in a store, and can further know that the sales and visibility are positively correlated. Finally, we can know that as the visibility increases, the popularity of items increases in turn.

Here are codes:

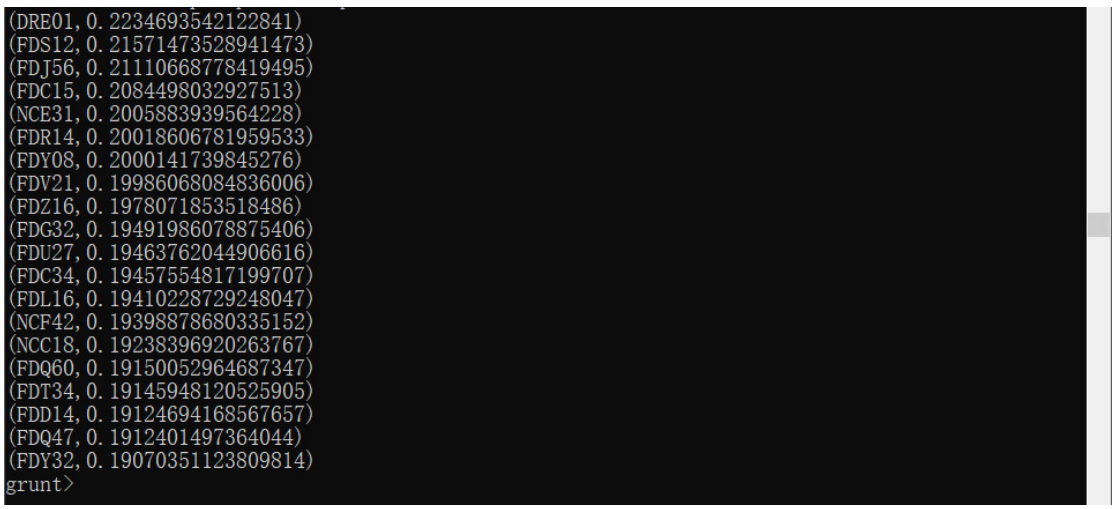
B = group A by Item_Identifier;

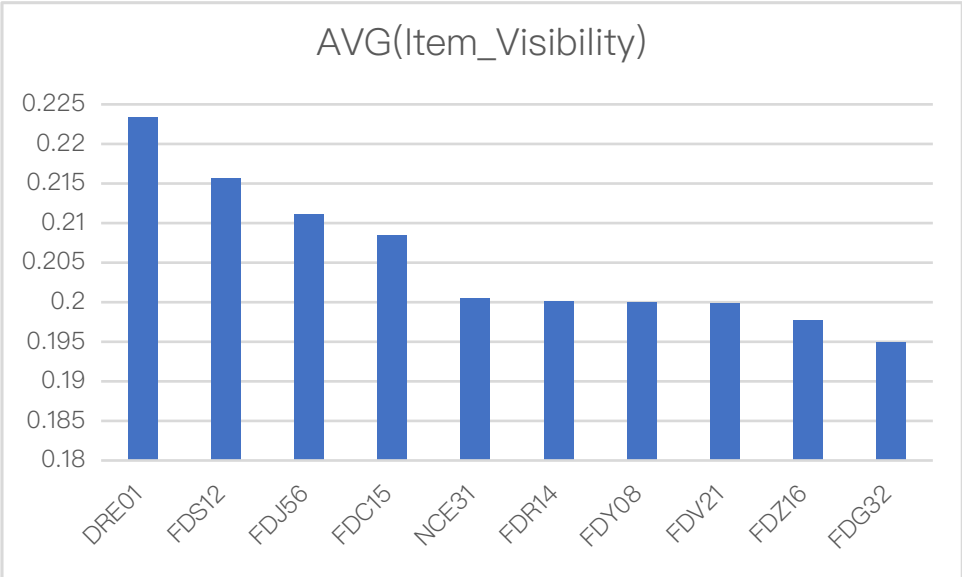
C = foreach B generate group, AVG(A.Item_Visibility);

D = order C by $1 DESC;

E = LIMIT D 20;

DUMP E;





The average visibility of each item.

It can be concluded to the sales and popularity of item types. It shows that in all kinds of items, people come to consume most probably for breakfast. We can tell that the first few are in high demand or inconvenient cooking in the family.

Here are codes:

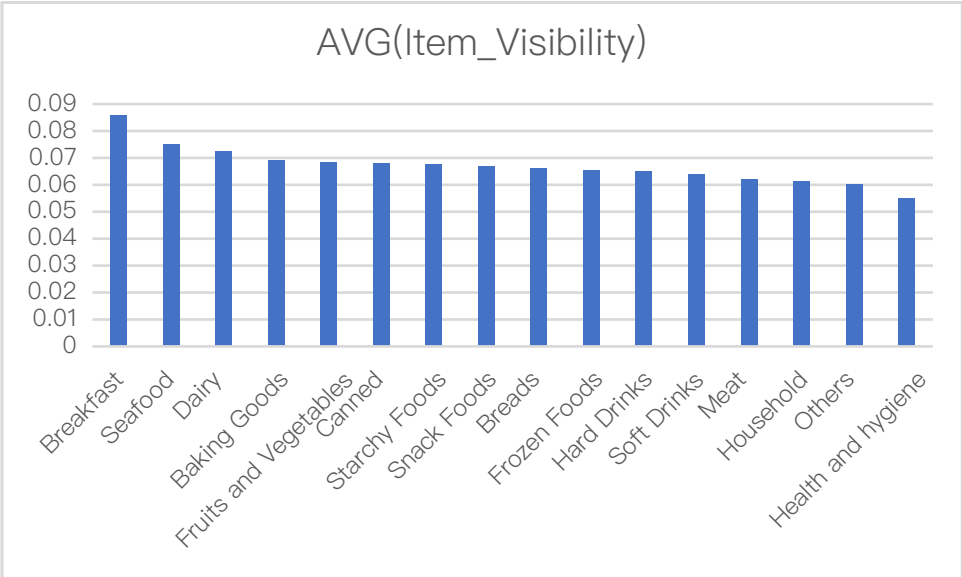
B = group A by Item_Type;

C = foreach B generate group, AVG(A.Item_Visibility);

D = order C by $1 DESC;

DUMP D;

```
(Breakfast,0.08572300958701155)
(Seafood,0.07497607974801213)
(Dairy,0.0724271983461307)
(Baking Goods,0.06916929967845158)
(Fruits and Vegetables,0.06851294274478183)
(Canned,0.06812931543706884)
(Starchy Foods,0.06756356396913729)
(Snack Foods,0.06685022279658975)
(Breads,0.06625509794905367)
(Frozen Foods,0.06564523899714489)
(Hard Drinks,0.06494255598512626)
(Soft Drinks,0.06397224776028247)
(Meat,0.062283811065204)
(Household,0.061322312902884334)
(Others,0.06024103182693117)
(Health and Hygiene,0.055215979741814615)
(Item_Type,)
grunt>
```



The average visibility of each fat content.

It is obvious that regular fat is a few more popular than low fat items. Being same as the mentioned, according to the visibility, this number of regular fat items is a little higher than the low fat, which means the number of goods prepared by merchant is different. It shows that there are more customers prefer regular fat food.

Here are codes:

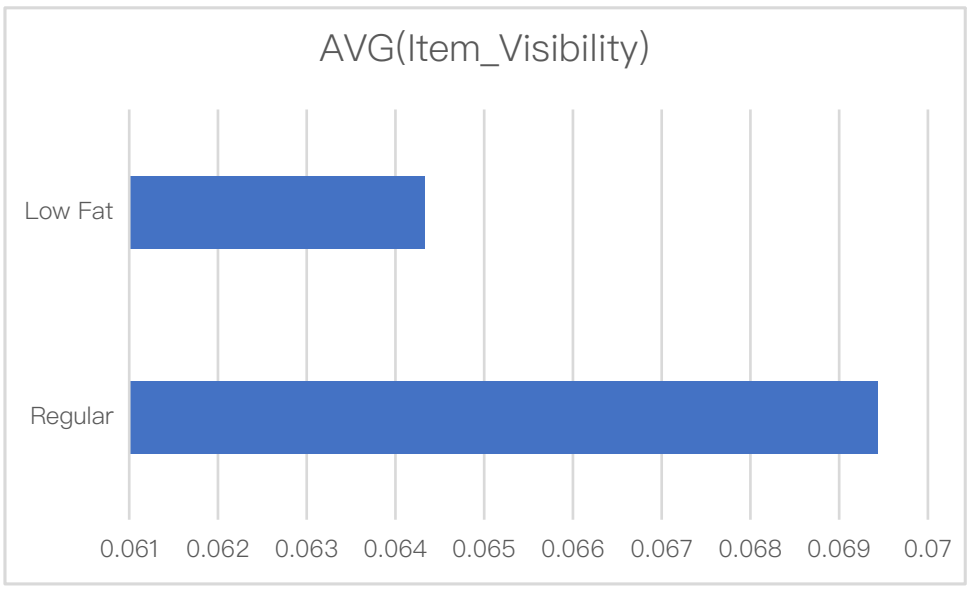
B = group A by Item_Fat_Content;

C = foreach B generate group, AVG(A.Item_Visibility);

D = order C by $1 DESC;

DUMP D;

```
(Regular,0.06943919373777183)
(Low Fat,0.0643300824614303 5)
(Item_Fat_Content,)
grunt>
```



(6)Sales Information

The sales of each outlet type

We can know that the average sales of supermarket type3 is the highest, the sum sales of type1 is the highest, and we can see grocery stores are the least profitable. So sales are related to the outlet type.

grunt> t = group D by Outlet_Type;

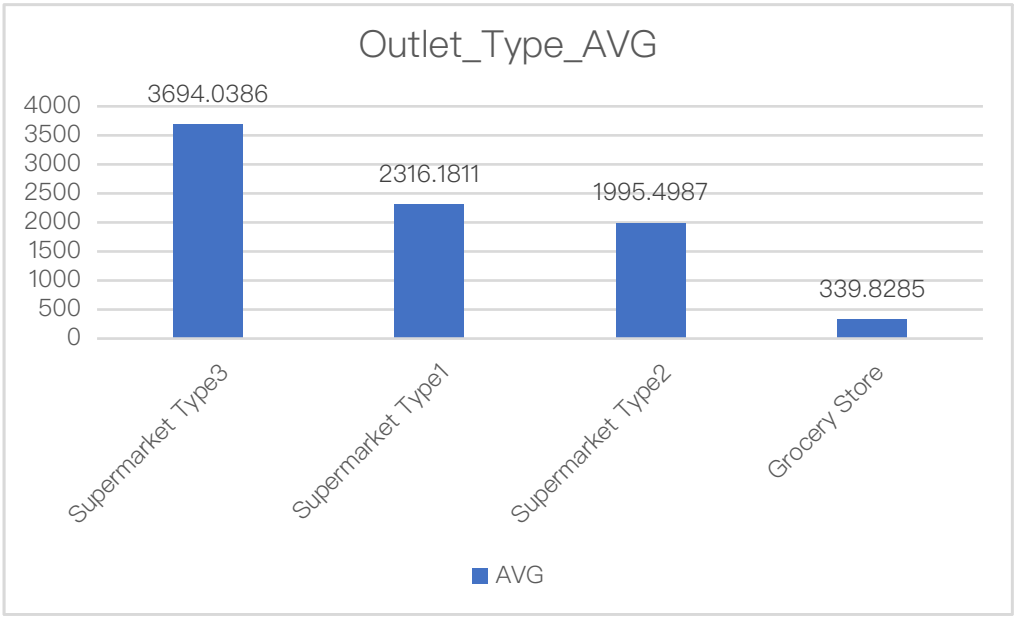
grunt> at = foreach t generate

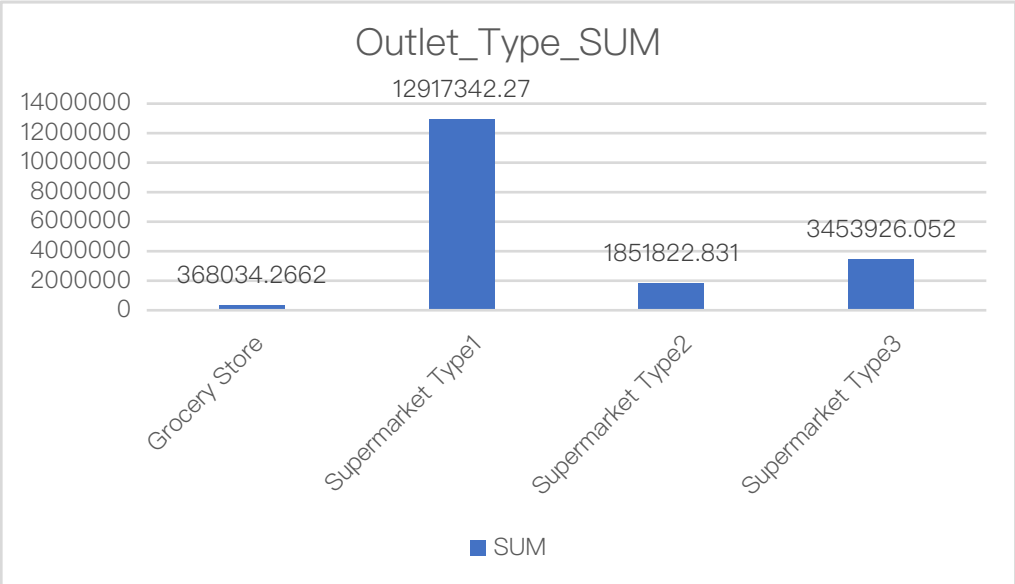
group,AVG(D.Item_Outlet_Sales),SUM(D.Item_Outlet_Sales);

grunt> x = order at by $1 DESC;

grunt> dump x

```
(Supermarket Type3,3694.038558029746,3453926.0517578125)
(Supermarket Type1,2316.181149360119,1.2917342269981384E7)
(Supermarket Type2,1995.4987403031053,1851822.8310012817)
(Grocery Store,339.828500668428,368034.26622390747)
```





The sales of each year of establishment

We can see that the sales situation was the best in 1985, the sales in 1998 was very low, and the sales situations in other years were similar.

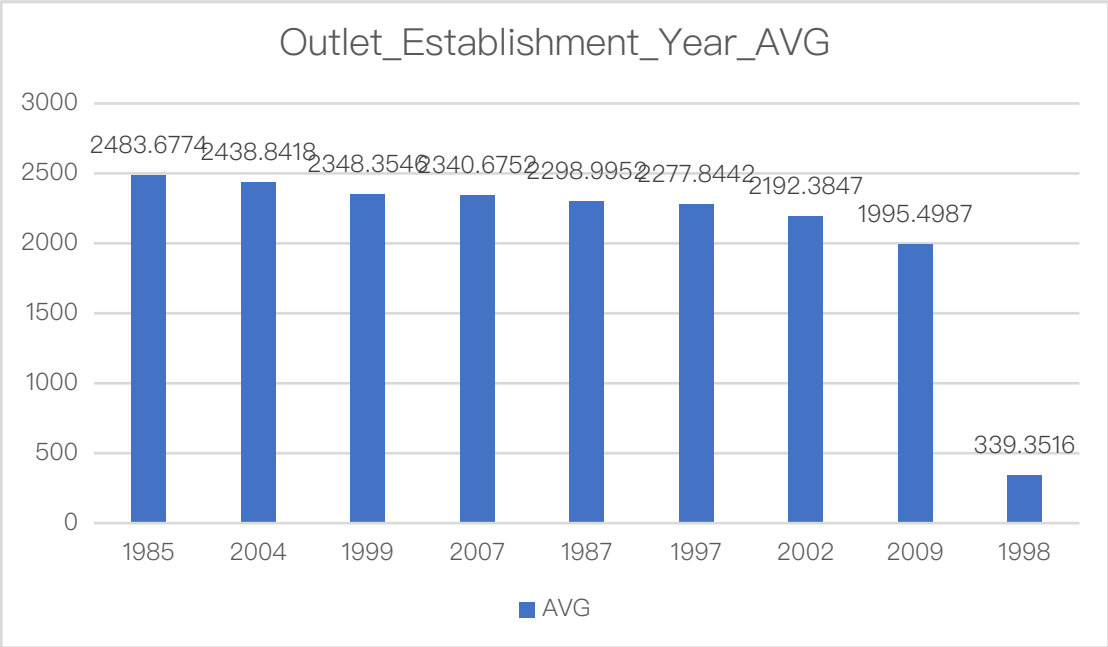
grunt> esy = group D by Outlet_Establishment_Year;

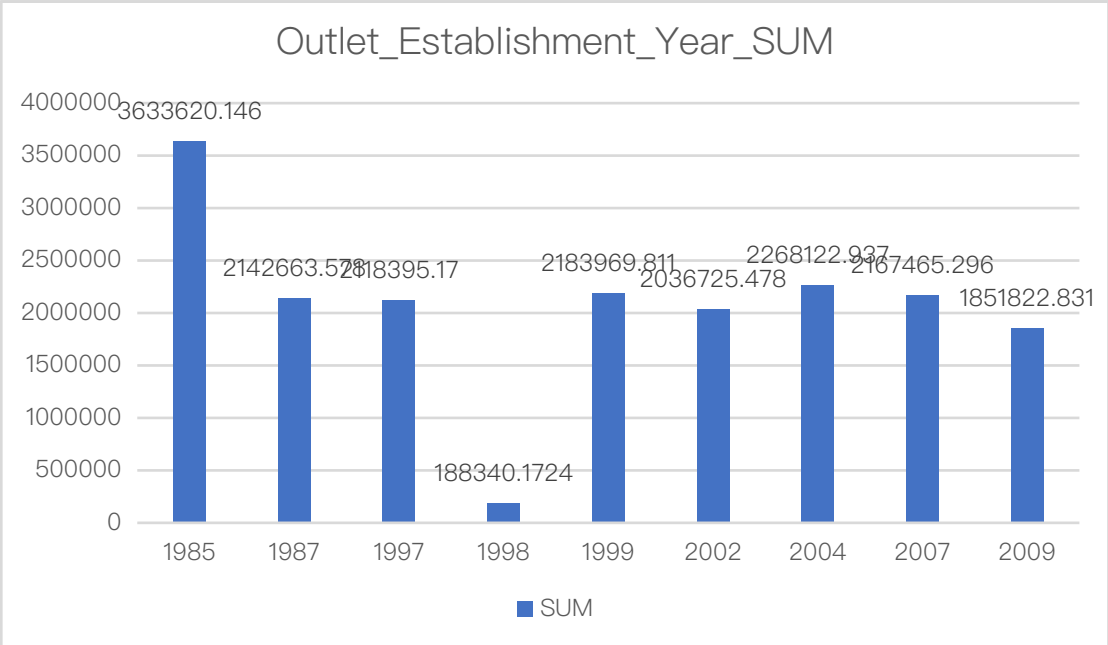
grunt> aesy = foreach esy generate group, AVG(D.Item_Outlet_Sales), SUM(D.Item_Outlet_Sales);

grunt> x = order aesy by $1 DESC;

grunt> dump x

```
(1985, 2483. 6774747513455, 3633620. 1455612183)
(2004, 2438. 8418682713664, 2268122. 9374923706)
(1999, 2348. 354635341193, 2183969. 8108673096)
(2007, 2340. 675265334852, 2167465. 2957000732)
(1987, 2298. 9952554252527, 2142663. 5780563354)
(1997, 2277. 844268405053, 2118395. 169616699)
(2002, 2192. 3847989758838, 2036725. 4782485962)
(2009, 1995. 4987403031053, 1851822. 8310012817)
(1998, 339. 351662018922, 188340. 1724205017)
```





The sales of each location type

The average sales of Tier2 is the highest, the sum sales of Tier3 is the highest, but the sales of Tier1 are much worse than others. So sales are related to the location type.

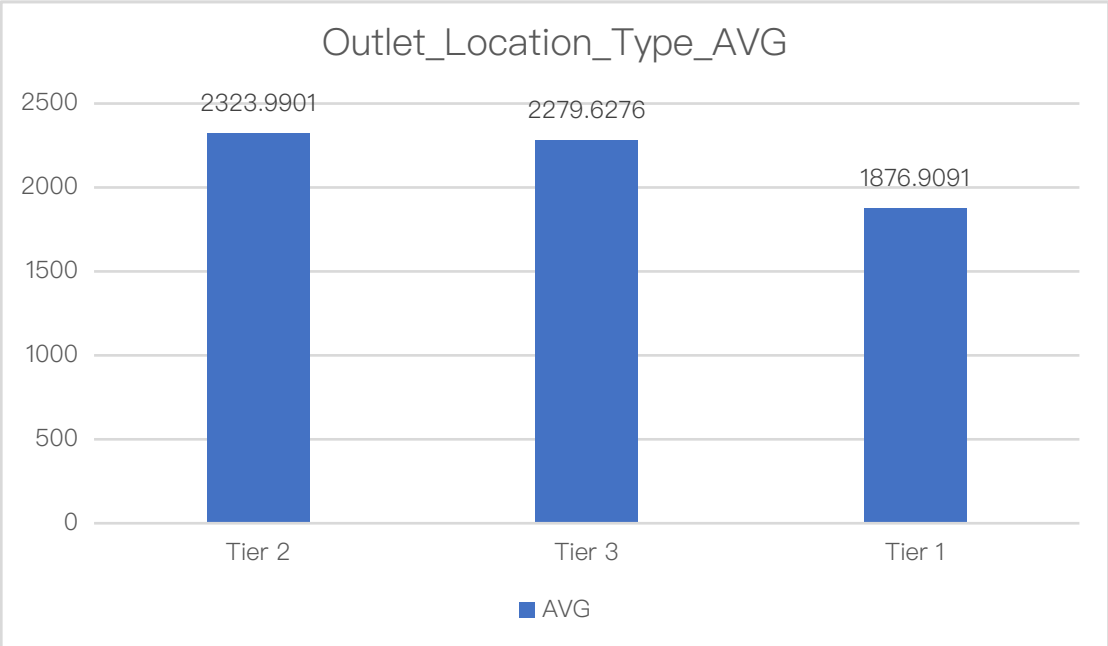
grunt> loc = group D by Outlet_Location_Type;

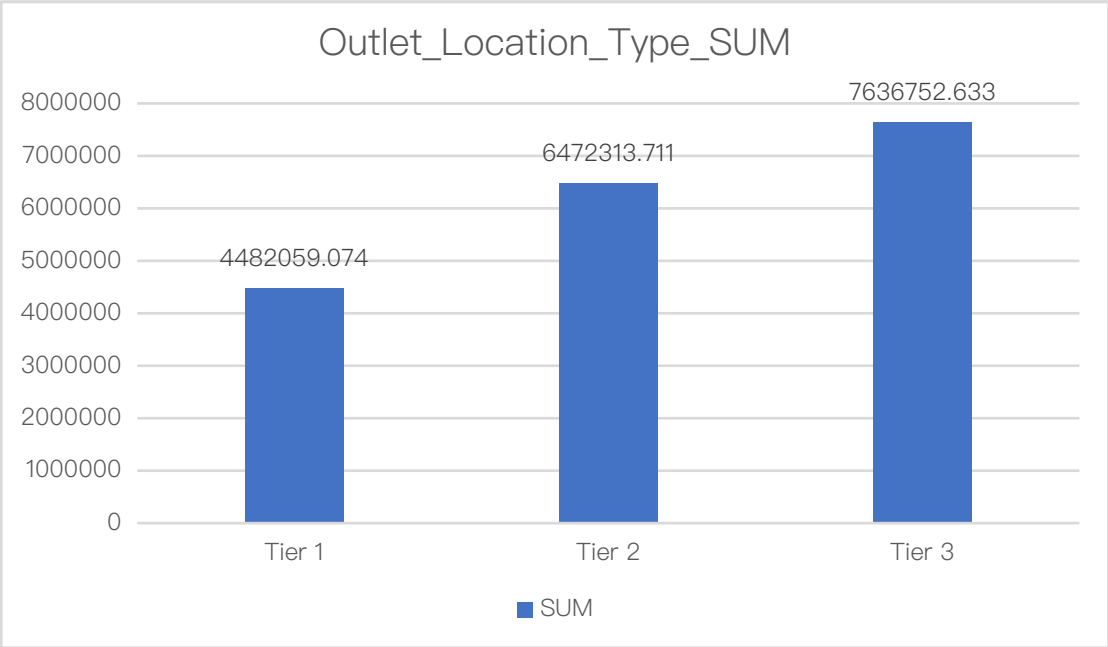
grunt> aloc = foreach loc generate group, AVG(D.Item_Outlet_Sales), SUM(D.Item_Outlet_Sales);

grunt> x = order aloc by $1 DESC;

grunt> dump x

```
(Tier 2,2323.9905606610555,6472313.71144104)
(Tier 3,2279.6276517122183,7636752.633235931)
(Tier 1,1876.9091600868571,4482059.074287415)
(Outlet_Location_Type,,)
grunt>
```





4 Pig in Data Processing

· Business Intelligence(BI):

Simply, BI is often understood as a tool to turn existing data in the enterprise into knowledge to help the business make informed business.

· OLAP(On-Line Analytical Processing):

OLPA is the main application of data warehouse systems, support complex analysis operations, focusing on decision support, and provides intuitive to understand the results of the query.

4.1 Introduction

We have two datasets: item.csv and sales.csv.

The datasets item has three columns:

Item_Identifier (Unique product ID),

Item_Type (The category to which the product belongs)

Item_Weight (Weight of product).

The datasets sales has five columns:

Item_Identifier (Unique product ID)

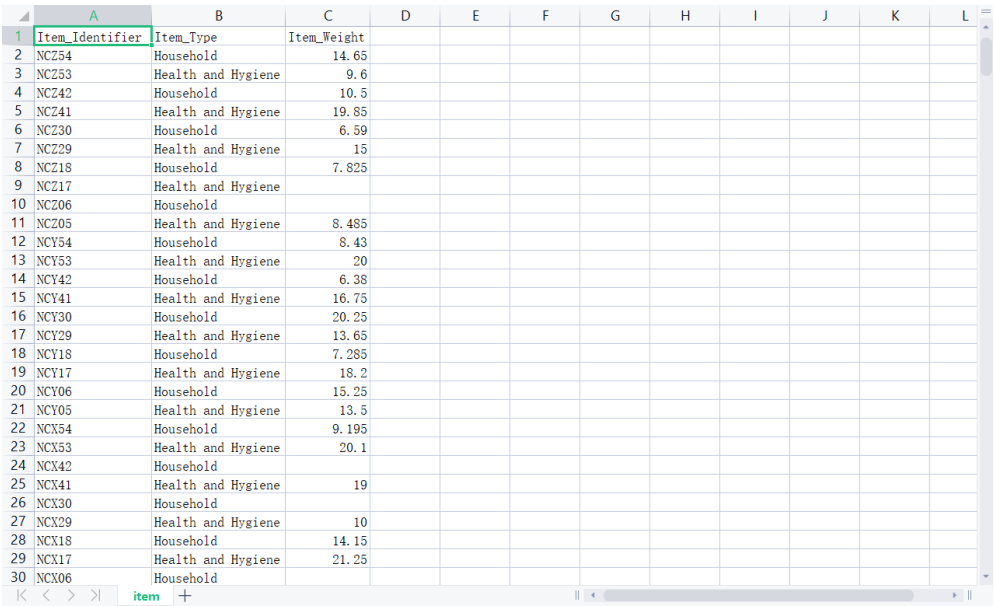
Outlet_Identifier (unique store ID)

Outlet_Location_Type (the type of city in which the store is located)

Outlet_Type (the type of the outlet)

Item_Outlet_Sales (Sales of the product in the particular store)

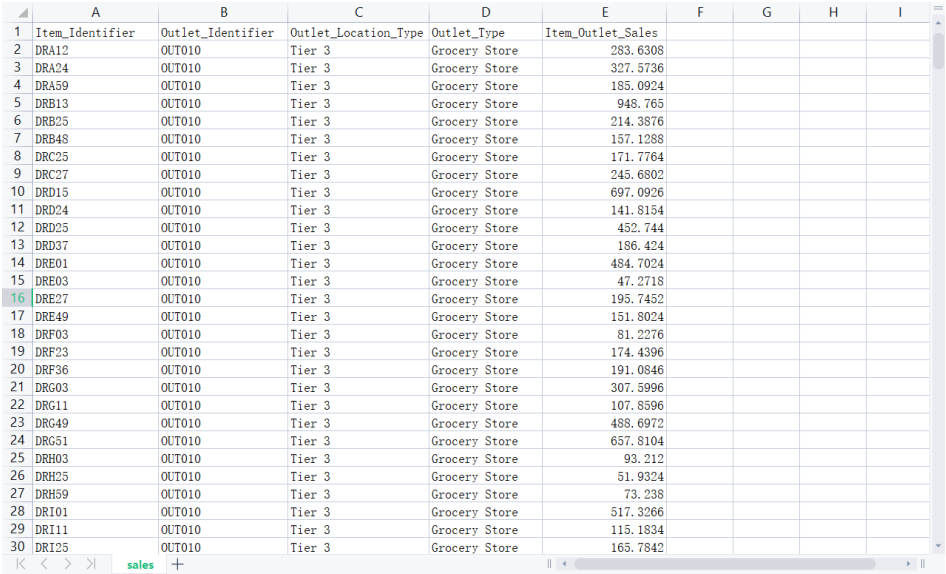
Item.csv



	A	B	C
1	Item_Identifier	Item_Type	Item_Weight
2	NCZ54	Household	14.65
3	NCZ53	Health and Hygiene	9.6
4	NCZ42	Household	10.5
5	NCZ41	Health and Hygiene	19.85
6	NCZ30	Household	6.59
7	NCZ29	Health and Hygiene	15
8	NCZ18	Household	7.825
9	NCZ17	Health and Hygiene	
10	NCZ06	Household	
11	NCZ05	Health and Hygiene	8.485
12	NCY54	Household	8.43
13	NCY53	Health and Hygiene	20
14	NCY42	Household	6.38
15	NCY41	Health and Hygiene	16.75
16	NCY30	Household	20.25
17	NCY29	Health and Hygiene	13.65
18	NCY18	Household	7.285
19	NCY17	Health and Hygiene	18.2
20	NCY06	Household	15.25
21	NCY05	Health and Hygiene	13.5
22	NCX54	Household	9.195
23	NCX53	Health and Hygiene	20.1
24	NCX42	Household	
25	NCX41	Health and Hygiene	19
26	NCX30	Household	
27	NCX29	Health and Hygiene	10
28	NCX18	Household	14.15
29	NCX17	Health and Hygiene	21.25
30	NCX06	Household	

item

Sales.csv



	A	B	C	D	E	F	G	H	I
1	Item_Identifier	Outlet_Identifier	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales				
2	DRA12	OUT010	Tier 3	Grocery Store	283.6308				
3	DRA24	OUT010	Tier 3	Grocery Store	327.5736				
4	DRA59	OUT010	Tier 3	Grocery Store	185.0924				
5	DRB13	OUT010	Tier 3	Grocery Store	948.765				
6	DRB25	OUT010	Tier 3	Grocery Store	214.3876				
7	DRB48	OUT010	Tier 3	Grocery Store	157.1288				
8	DRC25	OUT010	Tier 3	Grocery Store	171.7764				
9	DRC27	OUT010	Tier 3	Grocery Store	245.6802				
10	DRD15	OUT010	Tier 3	Grocery Store	697.0926				
11	DRD24	OUT010	Tier 3	Grocery Store	141.8154				
12	DRD25	OUT010	Tier 3	Grocery Store	452.744				
13	DRD37	OUT010	Tier 3	Grocery Store	186.424				
14	DRE01	OUT010	Tier 3	Grocery Store	484.7024				
15	DRE03	OUT010	Tier 3	Grocery Store	47.2718				
16	DRE27	OUT010	Tier 3	Grocery Store	195.7452				
17	DRE49	OUT010	Tier 3	Grocery Store	151.8024				
18	DRF03	OUT010	Tier 3	Grocery Store	81.2276				
19	DRF23	OUT010	Tier 3	Grocery Store	174.4396				
20	DRF36	OUT010	Tier 3	Grocery Store	191.0846				
21	DRG03	OUT010	Tier 3	Grocery Store	307.5996				
22	DRG11	OUT010	Tier 3	Grocery Store	107.8596				
23	DRG49	OUT010	Tier 3	Grocery Store	488.6972				
24	DRG51	OUT010	Tier 3	Grocery Store	657.8104				
25	DRH03	OUT010	Tier 3	Grocery Store	93.212				
26	DRH25	OUT010	Tier 3	Grocery Store	51.9324				
27	DRH59	OUT010	Tier 3	Grocery Store	73.238				
28	DRI01	OUT010	Tier 3	Grocery Store	517.3266				
29	DRI11	OUT010	Tier 3	Grocery Store	115.1834				
30	DRI25	OUT010	Tier 3	Grocery Store	165.7842				

sales

4.2 Data loading

（1）Create database and table

create database big_mart_sale;

create table Item (

itemID varchar(5),

itemType varchar(30),

itemWeight float,

primary key (itemID)

)

row format delimited fields terminated by ','

TBLPROPERTIES ('skip.header.line.count'='1');

create table shopSales (

itemID varchar(10) primary key,

shopID varchar(10) primary key,

shopLocation varchar(10),

shopType varchar(20),

itemSales float

)

row format delimited fields terminated by ','

TBLPROPERTIES ('skip.header.line.count'='1');

```
hive> show databases;
OK
default
finance
financials_ip
human_resources
jacob
practiceip
sales
school
Time taken: 0.427 seconds, Fetched: 8 row(s)
hive> create table Item (
    > itemID varchar(5),
    > itemType varchar(30),
    > itemWeight float
    > )
    > row format delimited fields terminated by ','
    > TBLPROPERTIES ('skip.header.line.count'='1');
OK
Time taken: 1.299 seconds
```

```
hive> create table shopSales (
    > itemID varchar(10),
    > shopID varchar(10),
    > shopLocation varchar(10),
    > shopType varchar(20),
    > itemSales float
    > )
    > row format delimited fields terminated by ','
    > TBLPROPERTIES ('skip.header.line.count'='1');
OK
Time taken: 0.087 seconds
hive> load data local inpath 'sales.csv' overwrite into table shopSales;
Loading data to table default.shopsales
Table default.shopsales stats: [numFiles=1, numRows=0, totalSize=407427, rawDataSize=0]
OK
Time taken: 0.331 seconds
hive>
```

（2） Load from hdfs

Use command:

load data local inpath 'item.csv' overwrite into table Item;

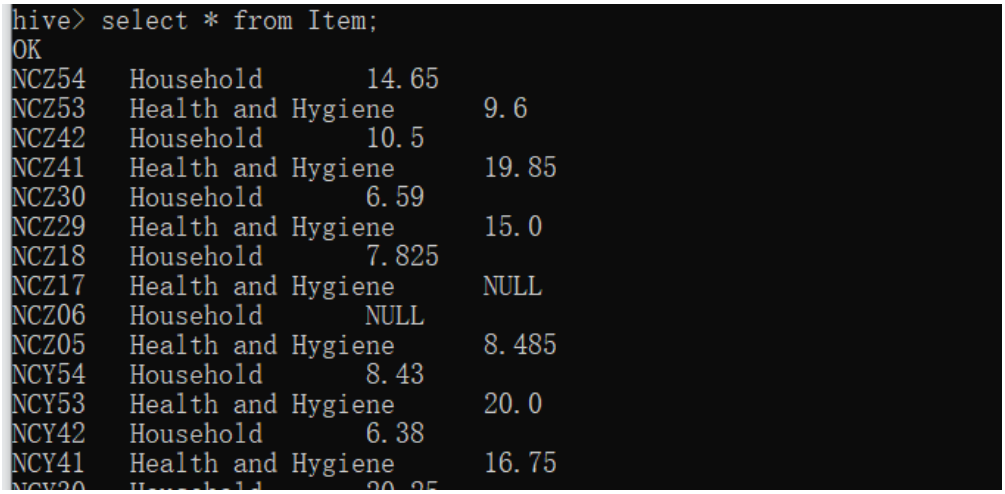
load data local inpath 'sales.csv' overwrite into table shopSales;

Then use command :

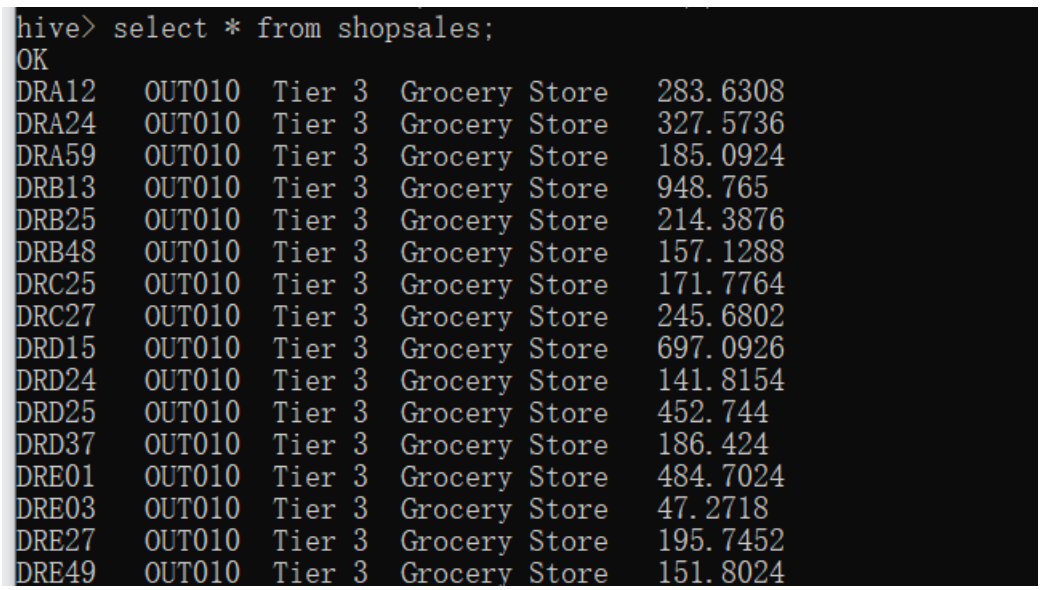
*select * from item;*

*select * from shopsales;*

We can see the data.



```
hive> select * from Item;
OK
NCZ54   Household       14.65
NCZ53   Health and Hygiene      9.6
NCZ42   Household       10.5
NCZ41   Health and Hygiene      19.85
NCZ30   Household       6.59
NCZ29   Health and Hygiene      15.0
NCZ18   Household       7.825
NCZ17   Health and Hygiene      NULL
NCZ06   Household       NULL
NCZ05   Health and Hygiene      8.485
NCY54   Household       8.43
NCY53   Health and Hygiene      20.0
NCY42   Household       6.38
NCY41   Health and Hygiene      16.75
```



```
hive> select * from shopsales;
OK
DRA12   OUT010  Tier 3  Grocery Store   283.6308
DRA24   OUT010  Tier 3  Grocery Store   327.5736
DRA59   OUT010  Tier 3  Grocery Store   185.0924
DRB13   OUT010  Tier 3  Grocery Store   948.765
DRB25   OUT010  Tier 3  Grocery Store   214.3876
DRB48   OUT010  Tier 3  Grocery Store   157.1288
DRC25   OUT010  Tier 3  Grocery Store   171.7764
DRC27   OUT010  Tier 3  Grocery Store   245.6802
DRD15   OUT010  Tier 3  Grocery Store   697.0926
DRD24   OUT010  Tier 3  Grocery Store   141.8154
DRD25   OUT010  Tier 3  Grocery Store   452.744
DRD37   OUT010  Tier 3  Grocery Store   186.424
DRE01   OUT010  Tier 3  Grocery Store   484.7024
DRE03   OUT010  Tier 3  Grocery Store   47.2718
DRE27   OUT010  Tier 3  Grocery Store   195.7452
DRE49   OUT010  Tier 3  Grocery Store   151.8024
```

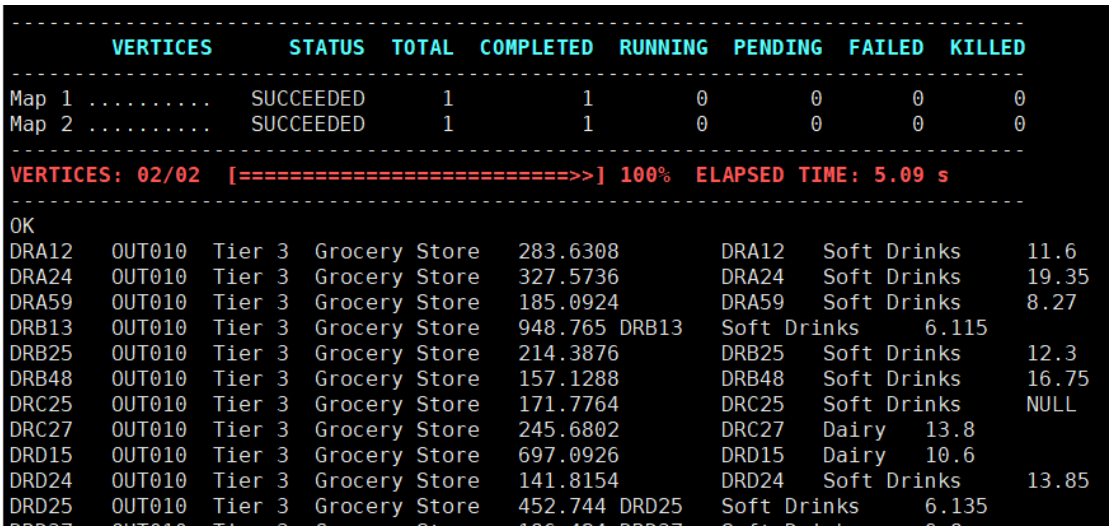
4.3 ETL & OLAP

（1） Join Tables

*select * from shopSales S LEFT JOIN Item I*

on S.itemID = I.itemID

limit 20;



```
----------------------------------------------------------------------------------
        VERTICES      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Map 2 ..........   SUCCEEDED      1          1        0        0       0       0
----------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 5.09 s
----------------------------------------------------------------------------------
OK
DRA12   OUT010  Tier 3  Grocery Store   283.6308        DRA12   Soft Drinks     11.6
DRA24   OUT010  Tier 3  Grocery Store   327.5736        DRA24   Soft Drinks     19.35
DRA59   OUT010  Tier 3  Grocery Store   185.0924        DRA59   Soft Drinks     8.27
DRB13   OUT010  Tier 3  Grocery Store   948.765 DRB13   Soft Drinks     6.115
DRB25   OUT010  Tier 3  Grocery Store   214.3876        DRB25   Soft Drinks     12.3
DRB48   OUT010  Tier 3  Grocery Store   157.1288        DRB48   Soft Drinks     16.75
DRC25   OUT010  Tier 3  Grocery Store   171.7764        DRC25   Soft Drinks     NULL
DRC27   OUT010  Tier 3  Grocery Store   245.6802        DRC27   Dairy   13.8
DRD15   OUT010  Tier 3  Grocery Store   697.0926        DRD15   Dairy   10.6
DRD24   OUT010  Tier 3  Grocery Store   141.8154        DRD24   Soft Drinks     13.85
DRD25   OUT010  Tier 3  Grocery Store   452.744 DRD25   Soft Drinks     6.135
```

（2） Use select

Use select to get the total sales of each category of goods in each shop：

select S.shopID, I.itemType, SUM(S.itemSales) as Totalsales

from shopSales S left join Item I

on S.itemID = I.itemID

group by S.shopID,I.itemType

order by S.shopID,totalsales desc;

The sales of snack foods in shop OUT010 is the highest.

（3） With Rollup

select S.shopID, I.itemType, SUM(S.itemSales) as Totalsales

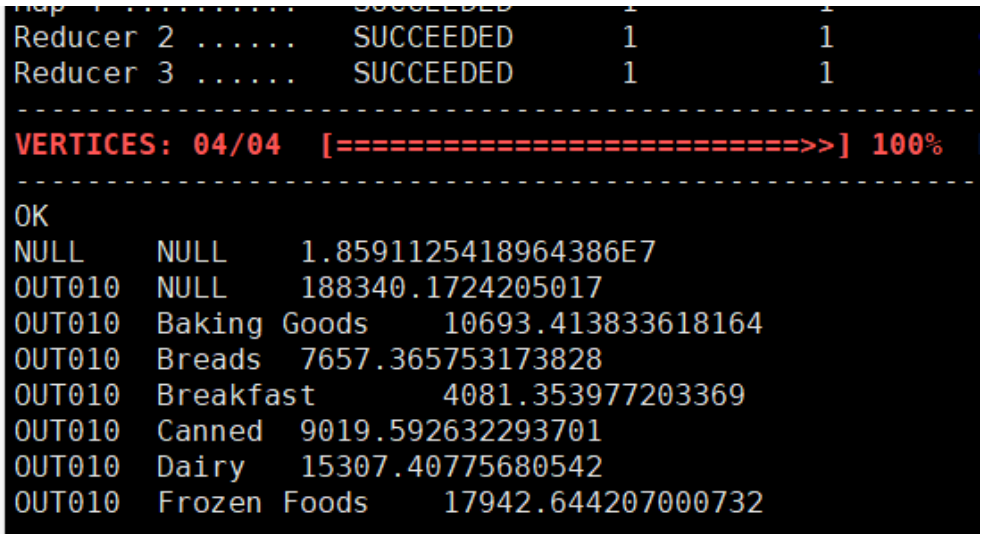
from shopSales S LEFT JOIN Item I

on S.itemID = I.itemID

group by S.shopID,I.itemType with rollup

order by S.shopID,I.itemType;

from this command ,we can see the totalsales of each itemType each shop:



There are the null value in the result.

This is the hierarchical aggregation with shopID dimension. And line2 mean the total sales of all categroy in shop OUT010. If we change the position of group by, such as:

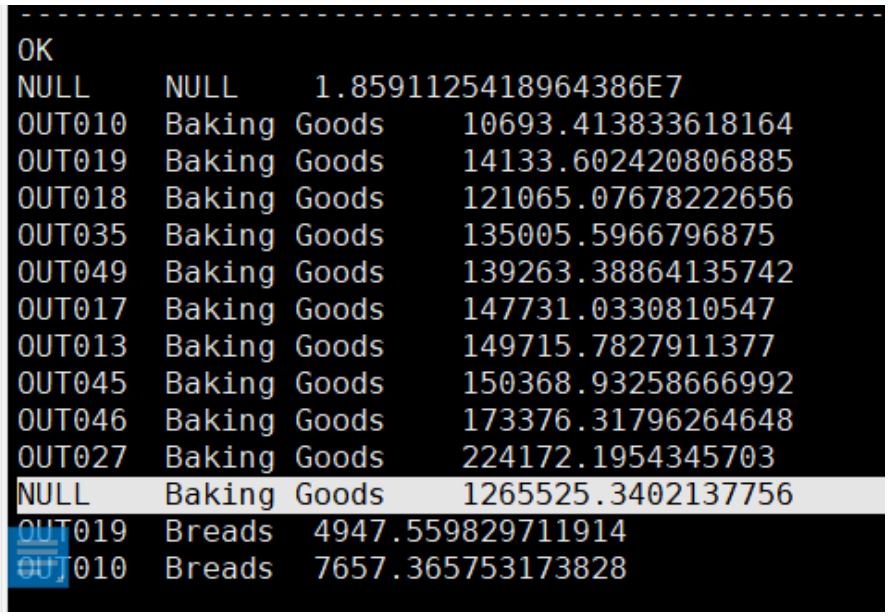
select S.shopID, I.itemType, SUM(S.itemSales) as Totolsales

from shopSales S LEFT JOIN Item I

on S.itemID = I.itemID

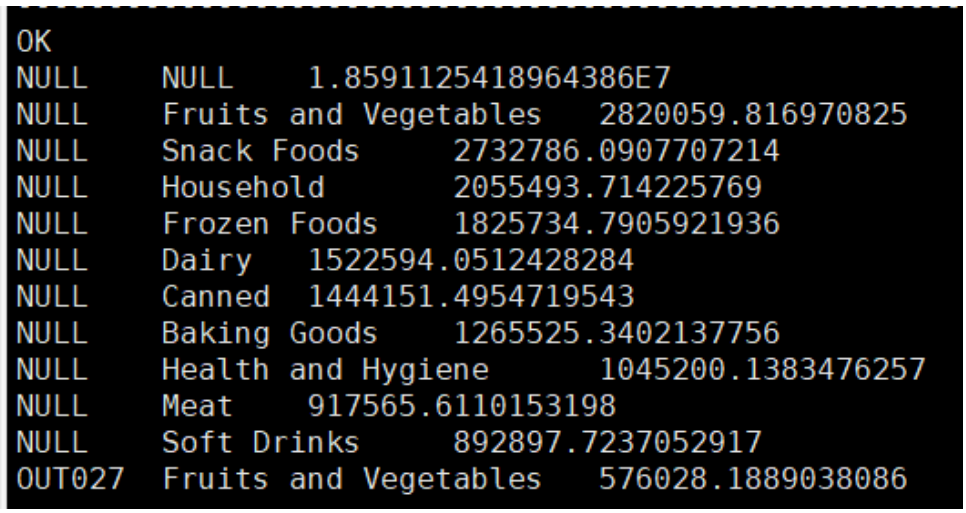
group by I.itemType,S.shopID with rollup

order by I.itemType,Totolsales;



This is the hierarchical aggregation with itemType dimension. And this line mean the total sales of baking goods in all the shop.

Order the data by the totalsales and get the result: we can see the Fruits and Vegetables are the best sell.



Rollup other dimension to get the different result, such as:

the totalsales of each item each shop：

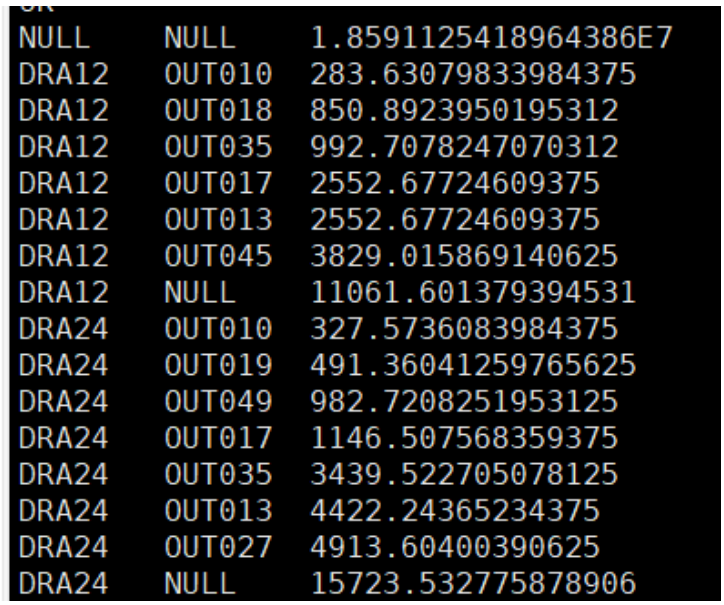
select I.itemID, S.shopID, SUM(S.itemSales) as Totolsales

from shopSales S LEFT JOIN Item I

on S.itemID = I.itemID

group by I.itemID,S.shopID with rollup

order by I.itemID,Totolsales;



NULL	NULL	1.8591125418964386E7
DRA12	OUT010	283.63079833984375
DRA12	OUT018	850.8923950195312
DRA12	OUT035	992.7078247070312
DRA12	OUT017	2552.67724609375
DRA12	OUT013	2552.67724609375
DRA12	OUT045	3829.015869140625
DRA12	NULL	11061.601379394531
DRA24	OUT010	327.5736083984375
DRA24	OUT019	491.36041259765625
DRA24	OUT049	982.7208251953125
DRA24	OUT017	1146.507568359375
DRA24	OUT035	3439.522705078125
DRA24	OUT013	4422.24365234375
DRA24	OUT027	4913.60400390625
DRA24	NULL	15723.532775878906

Information: The item ‘DRA14’ is best sell in shop ‘OUT045’ and the item ‘DRA24’ is best sell in shop ‘OUT027’.

Use the where satement to get the detail that you want to know,such as “where shopID=?” or “where itemID =?”

the totalsales of each item each itemType：

select i.itemType,I.itemID, SUM(S.itemSales) as Totolsales

from shopSales S LEFT JOIN Item I

on S.itemID = I.itemID

group by I.itemType,I.itemID with rollup

order by I.itemType,Totolsales desc;

```
NULL    NULL    1.8591125418964386E7
Baking Goods    NULL    1265525.3402137756
Baking Goods    FDU12   29793.21856689453
Baking Goods    FDL24   26038.7724609375
Baking Goods    FDJ48   25936.238845825195
Baking Goods    FDB37   25717.8564453125
Baking Goods    FDX36   25429.565185546875
Baking Goods    FDV60   24158.552673339844
Baking Goods    FDE36   24076.65966796875
Baking Goods    FDT12   23247.738372802734
Baking Goods    FDG60   22967.436279296875
Baking Goods    FDP48   22337.58984375
Baking Goods    FDN60   19966.010162353516
Baking Goods    FDZ23   19947.368103027344
Baking Goods    FDQ24   19630.447052001953
Baking Goods    FDO12   18791.538818359375
Baking Goods    FDX11   18721.630249023438
Baking Goods    FDH60   18659.044891357422
```

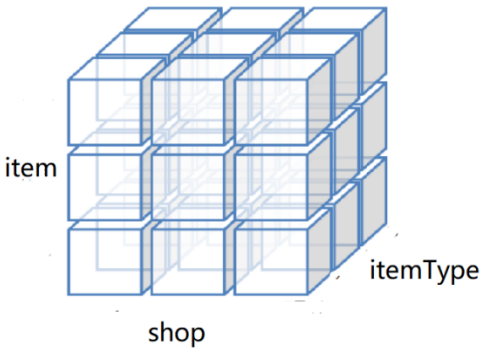
```
OK
NULL    NULL    2732786.0907707214
Snack Foods     NULL    2732786.0907707214
Snack Foods     FDT21   31873.177307128906
Snack Foods     FDN58   31279.283203125
Snack Foods     FDL58   30584.188903808594
Snack Foods     FDV57   28223.262084960938
Snack Foods     FDX58   27830.440002441406
Snack Foods     FDW34   26974.887268066406
Snack Foods     FDL57   26866.36181640625
Snack Foods     FDP33   26845.056121826172
Snack Foods     FDC46   26557.43051147461
Snack Foods     FDI22   26550.772399902344
Snack Foods     FDF22   25860.337768554688
Snack Foods     FDJ21   25225.16455078125
Snack Foods     FDL34   25124.628662109375
Snack Foods     FDV45   24971.494995117188
Snack Foods     FDR34   24735.80224609375
Snack Foods     FDO09   24721.153930664062
```

Information:in all baking goods, the item ‘FDU12’ is best sell. And in all snack foods, the item ‘FDT21’ is best sell.

（4）Cube

Cube can query multiple dimensions of hive. With cube, You don't have to write a lot of select statements and union them.

When building the data cube. For instance, dicing to face the data cube of shopID, itemType and itemID.



select S.shopID,I.itemType,I.itemID, SUM(S.itemSales) as Totolsales

from shopSales S LEFT JOIN Item I on S.itemID = I.itemID

group by S.shopID,I.itemType,I.itemID with cube;

OUT049	Seafood	NULL	11827.271102905273
OUT049	Seafood	FDG21	2247.074951171875
OUT049	Seafood	FDG33	3263.751708984375
OUT049	Seafood	FDH09	473.3837890625
OUT049	Seafood	FDH21	1267.6832275390625
OUT049	Seafood	FDI09	2396.8798828125
OUT049	Seafood	FDI57	1970.7679443359375
OUT049	Seafood	FDJ45	207.72959899902344

OUT049	NULL	NCL30	2173.171142578125
OUT049	NULL	NCL31	5296.43896484375
OUT049	NULL	NCL41	311.5943908691406
OUT049	NULL	NCL42	5635.3310546875
OUT049	NULL	NCM05	3964.839111328125
OUT049	NULL	NCM06	1853.587158203125
OUT049	NULL	NCM07	1006.6895751953125
OUT049	NULL	NCM18	1114.5491943359375
OUT049	NULL	NCM26	2450.14404296875
OUT049	NULL	NCM30	784.3123779296875

NULL	NULL	FDT46	3997.463165283203
NULL	NULL	FDT47	6707.2691650390625
NULL	NULL	FDT48	18451.98126220703
NULL	NULL	FDT49	13102.944274902344
NULL	NULL	FDT50	7957.6416015625
NULL	NULL	FDT51	2348.9424438476562
NULL	NULL	FDT52	17151.008056640625
NULL	NULL	FDT55	26982.876708984375
NULL	NULL	FDT56	4170.571197509766
NULL	NULL	FDT57	20384.132843017578

```
NULL    Breads  NULL    553237.1875305176
NULL    Breads  FDN23   17127.0390625
NULL    Breads  FDO11   17430.644454956055
NULL    Breads  FDO23   10872.513916015625
NULL    Breads  FDP11   20029.926818847656
NULL    Breads  FDP23   18288.194702148438
NULL    Breads  FDP59   6958.94157409668
NULL    Breads  FDQ11   13877.935089111328
NULL    Breads  FDQ23   8612.7890625
```

Selecting slices of the data cube to answer the OLAP query:

It only focuses on some dimensions,For instance: itemID ='FDR12',

the totalsales of item 'FDR12' in each shop then slicing to select the itemID ='FDR12'.

*select * from*

(select S.shopID,I.itemType,I.itemID, SUM(S.itemSales) as Totalsales

from shopSales S LEFT JOIN Item I on S.itemID = I.itemID

group by S.shopID,I.itemType,I.itemID with cube) a

where a.itemID ='FDR12'

order by a.Totalsales desc;

```
--------------------------------------------------------------------------------
Map 1 ..........   SUCCEEDED      1          1        0        0       0       0
Map 4 ..........   SUCCEEDED      1          1        0        0       0       0
Reducer 2 ......   SUCCEEDED      1          1        0        0       0       0
Reducer 3 ......   SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 04/04  [==========================>>] 100%  ELAPSED TIME: 7.18 s
--------------------------------------------------------------------------------
OK
NULL    NULL    NULL    11852.571655273438
NULL    NULL    FDR12   11852.571655273438
NULL    Baking Goods    NULL    11852.571655273438
NULL    Baking Goods    FDR12   11852.571655273438
OUT013  NULL    FDR12   3779.080810546875
OUT013  Baking Goods    NULL    3779.080810546875
OUT013  Baking Goods    FDR12   3779.080810546875
OUT013  NULL    NULL    3779.080810546875
OUT027  Baking Goods    FDR12   3091.97509765625
OUT027  Baking Goods    NULL    3091.97509765625
OUT027  NULL    FDR12   3091.97509765625
OUT027  NULL    NULL    3091.97509765625
OUT045  Baking Goods    FDR12   2233.09326171875
```

Or the totalsales of each item of ‘Fruits and vegetables’ in each shop then slicing to select the itemType = “Fruits and vegetables”.

· apply UNION command together with ROLLUP command for cube view

It equal to:

select S.shopID,I.itemType,I.itemID, SUM(S.itemSales) as Totolsales

from shopSales S LEFT JOIN Item I

on S.itemID = I.itemID

group by S.shopID,I.itemType,I.itemID with rollup;

UNION

select Null,I.itemType,I.itemID, SUM(S.itemSales) as Totolsales

from shopSales S LEFT JOIN Item I

on S.itemID = I.itemID

group by I.itemType,I.itemID with rollup;

UNION

select S.shopID,Null,I.itemID, SUM(S.itemSales) as Totolsales

from shopSales S LEFT JOIN Item I

on S.itemID = I.itemID

group by S.shopID,I.itemID with rollup;

5. Conclusion

According to query, we can know which kind of goods in the same type sell better and the best-selling goods in various stores and so on.

So we can selectively increase the amount of hot goods in the store and

select the best-sell good of the same type when purchasing.