In [1]:

```
import gzip
import shutil
import os
import pandas as pd
from ast import literal_eval
import json
from datetime import datetime
from sqlalchemy import create_engine
```

In [2]:

```
for i in os.listdir():
 if 'json' in i:
  with gzip.open(i, 'rb') as f_in:
     with open(i.replace('.gz',''), 'wb') as f_out:
      shutil.copyfileobj(f_in, f_out)
```

In [3]:

```
receipts = pd.read_json('receipts.json',lines=True)
brands = pd.read_json('brands.json',lines=True)
users = pd.read_json('users.json',lines=True)
```

In [15]:

```
receipts.head(10)
```

Out[15]:

	_id	bonusPointsEarned	bonusPointsEarnedReason	createDate	dateScanned	finishedDate	
0	{'$oid': '5ff1e1eb0a720f0523000575'}	500.0	Receipt number 2 completed, bonus point schedu...	{'$date': 1609687531000}	{'$date': 1609687531000}	{'$date': 1609687531000}	160
1	{'$oid': '5ff1e1bb0a720f052300056b'}	150.0	Receipt number 5 completed, bonus point schedu...	{'$date': 1609687483000}	{'$date': 1609687483000}	{'$date': 1609687483000}	160
2	{'$oid': '5ff1e1f10a720f052300057a'}	5.0	All-receipts receipt bonus	{'$date': 1609687537000}	{'$date': 1609687537000}	NaN	160
3	{'$oid': '5ff1e1ee0a7214ada100056f'}	5.0	All-receipts receipt bonus	{'$date': 1609687534000}	{'$date': 1609687534000}	{'$date': 1609687534000}	160
4	{'$oid': '5ff1e1d20a7214ada1000561'}	5.0	All-receipts receipt bonus	{'$date': 1609687506000}	{'$date': 1609687506000}	{'$date': 1609687511000}	16
5	{'$oid': '5ff1e1e40a7214ada1000566'}	750.0	Receipt number 1 completed, bonus point schedu...	{'$date': 1609687524000}	{'$date': 1609687524000}	{'$date': 1609687525000}	160
6	{'$oid': '5ff1e1cd0a720f052300056f'}	5.0	All-receipts receipt bonus	{'$date': 1609687501000}	{'$date': 1609687501000}	{'$date': 1609687502000}	160
7	{'$oid': '5ff1e1a40a720f0523000569'}	500.0	Receipt number 2 completed, bonus point schedu...	{'$date': 1609687460000}	{'$date': 1609687460000}	{'$date': 1609687461000}	16
8	{'$oid': '5ff1e1ed0a7214ada100056e'}	5.0	All-receipts receipt bonus	{'$date': 1609687533000}	{'$date': 1609687533000}	{'$date': 1609687534000}	160
9	{'$oid': '5ff1e1eb0a7214ada100056b'}	250.0	Receipt number 3 completed, bonus point schedu...	{'$date': 1609687531000}	{'$date': 1609687531000}	{'$date': 1609687531000}	160

In [11]:

```
brands.head()
```

Out[11]:

	_id	barcode	category	categoryCode	cpg	name	topBra
0	{'$oid': '601ac115be37ce2ead437551'}	511111019862	Baking	BAKING	{'$id': {'$oid': '601ac114be37ce2ead437550'}, ...	test brand @1612366101024	
1	{'$oid': '601c5460be37ce2ead43755f'}	511111519928	Beverages	BEVERAGES	{'$id': {'$oid': '5332f5fbe4b03c9a25efd0ba'}, ...	Starbucks	
2	{'$oid': '601ac142be37ce2ead43755d'}	511111819905	Baking	BAKING	{'$id': {'$oid': '601ac142be37ce2ead437559'}, ...	test brand @1612366146176	
3	{'$oid': '601ac142be37ce2ead43755a'}	511111519874	Baking	BAKING	{'$id': {'$oid': '601ac142be37ce2ead437559'}, ...	test brand @1612366146051	
4	{'$oid': '601ac142be37ce2ead43755e'}	511111319917	Candy & Sweets	CANDY_AND_SWEETS	{'$id': {'$oid': '5332fa12e4b03c9a25efd1e7'}, ...	test brand @1612366146827	

In [12]:

```
users.head()
```

Out[12]:

	_id	active	createdDate	lastLogin	role	signUpSource	state
0	{'$oid': '5ff1e194b6a9d73a3a9f1052'}	True	{'$date': 1609687444800}	{'$date': 1609687537858}	consumer	Email	WI
1	{'$oid': '5ff1e194b6a9d73a3a9f1052'}	True	{'$date': 1609687444800}	{'$date': 1609687537858}	consumer	Email	WI
2	{'$oid': '5ff1e194b6a9d73a3a9f1052'}	True	{'$date': 1609687444800}	{'$date': 1609687537858}	consumer	Email	WI
3	{'$oid': '5ff1e1eacfcf6c399c274ae6'}	True	{'$date': 1609687530554}	{'$date': 1609687530597}	consumer	Email	WI
4	{'$oid': '5ff1e194b6a9d73a3a9f1052'}	True	{'$date': 1609687444800}	{'$date': 1609687537858}	consumer	Email	WI

In [21]:

```
receipts.to_csv('receipts.csv')
brands.to_csv('brands.csv')
users.to_csv('users.csv')
```

Checking for null values

In [25]:

```
receipts.isnull().sum()
```

Out[25]:

```
_id                          0
bonusPointsEarned          575
bonusPointsEarnedReason    575
createDate                   0
dateScanned                  0
finishedDate               551
modifyDate                   0
pointsAwardedDate          582
pointsEarned               510
purchaseDate               448
purchasedItemCount         484
rewardsReceiptItemList     440
rewardsReceiptStatus         0
totalSpent                 435
userId                       0
dtype: int64
```

In [26]:

```
brands.isnull().sum()
```

Out[26]:

```
_id               0
barcode           0
category        155
categoryCode    650
cpg               0
name              0
topBrand        612
brandCode       234
dtype: int64
```

In [27]:

```
users.isnull().sum()
```

Out[27]:

```
_id              0
active           0
createdDate      0
lastLogin       62
role             0
signUpSource    48
state           56
dtype: int64
```

For receipts data

Fraction of missing values

In [32]:

```
percentage = receipts.isnull().mean()
for key, value in percentage.items():
    if value>0:
        print(key," : ",value*100)
```

```
bonusPointsEarned  :  51.385165326184094
bonusPointsEarnedReason  :  51.385165326184094
finishedDate  :  49.240393208221626
pointsAwardedDate  :  52.01072386058981
pointsEarned  :  45.57640750670242
purchaseDate  :  40.03574620196604
purchasedItemCount  :  43.25290437890974
rewardsReceiptItemList  :  39.32082216264522
totalSpent  :  38.8739946380697
```

This table has a lot of missing data values

For this data, a lot more analysis could be done, but for that the data have to be cleaned. Especially the 'rewardsReceiptItemList' column.

For user data

In [33]:

```
percentage = users.isnull().mean()
for key, value in percentage.items():
    if value>0:
        print(key," : ",value*100)
```

```
lastLogin  :  12.525252525252526
signUpSource  :  9.696969696969697
state  :  11.313131313131313
```

In [34]:

```
users['state'].unique()
```

Out[34]:

```
array(['WI', 'KY', 'AL', 'CO', 'IL', nan, 'OH', 'SC', 'NH'], dtype=object)
```

We can capture the state distribution

In [35]:

```
frequency = 100*(users['state'].value_counts()/len(users))
print(frequency)
```

```
WI    80.000000
NH     4.040404
AL     2.424242
OH     1.010101
IL     0.606061
KY     0.202020
CO     0.202020
SC     0.202020
Name: state, dtype: float64
```

We can see here that the majority distribution is from a single state

For brand data

In [36]:

```
percentage = brands.isnull().mean()
for key, value in percentage.items():
    if value>0:
        print(key," : ",value*100)
```

```
category  :  13.281919451585262
categoryCode  :  55.69837189374465
topBrand  :  52.44215938303341
brandCode  :  20.051413881748072
```

Brand table has a lot of missing values too

Note

This data has a lot of scope for analysis, but the majority of it depends on the demands. In the receipts table alone, there are too many attributes that can be analysed and plotted into a visualizations. It can also be broken off into two tables for a more in depth analysis.