


```
import pandas as pd
import numpy as np
import json

file_name = '/content/receipts.json'

# Load the JSON file
with open(file_name, 'r') as file:
    data = [json.loads(line) for line in file]

# Convert to DataFrame
df = pd.DataFrame(data)

df.head()
```




	_id	bonusPointsEarned	bonusPointsEarnedReason	createDate	dateScanned	finishedDate	modifyDate
0	{'Soid': '5ff1e1eb0a720f0523000575'}	500.0	Receipt number 2 completed, bonus point schedu...	{'\$date': 1609687531000}	{'\$date': 1609687531000}	{'\$date': 1609687531000}	{'\$date': 1609687536000}
1	{'Soid': '5ff1e1bb0a720f052300056b'}	150.0	Receipt number 5 completed, bonus point schedu...	{'\$date': 1609687483000}	{'\$date': 1609687483000}	{'\$date': 1609687483000}	{'\$date': 1609687488000}
2	{'Soid': '5ff1e1f10a720f052300057a'}	5.0	All-receipts receipt bonus	{'\$date': 1609687537000}	{'\$date': 1609687537000}	NaN	{'\$date': 1609687542000}
3	{'Soid': '5ff1e1ee0a7214ada100056f'}	5.0	All-receipts receipt bonus	{'\$date': 1609687534000}	{'\$date': 1609687534000}	{'\$date': 1609687534000}	{'\$date': 1609687539000}
4	{'Soid': '5ff1e1d20a7214ada1000561'}	5.0	All-receipts receipt bonus	{'\$date': 1609687506000}	{'\$date': 1609687506000}	{'\$date': 1609687511000}	{'\$date': 1609687511000}

Next steps:

Generate code with df

 View recommended plots

```
# Check for missing values
missing_values = df.isnull().sum()
print("Missing Values:\n", missing_values)
```




Missing Values:	
_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	

```
# Count the total number of rows
total_rows = len(df)

# Count the number of duplicate rows based on '_id'
duplicate_rows = df.duplicated(subset=['_id']).sum()

# Calculate the percentage of duplicate records
percentage_duplicates = (duplicate_rows / total_rows) * 100

print(f"Percentage of Duplicate Records: {percentage_duplicates:.2f}%")
```



Percentage of Duplicate Records: 0.00%

```
# Check data types
data_types = df.dtypes
print("Data Types:\n", data_types)
```

```
↗ Data Types:
   _id                object
bonusPointsEarned    float64
bonusPointsEarnedReason  object
createDate           object
dateScanned          object
finishedDate         object
modifyDate           object
pointsAwardedDate    object
pointsEarned         object
purchaseDate         object
purchasedItemCount    float64
rewardsReceiptItemList  object
rewardsReceiptStatus  object
totalSpent           object
userId              object
dtype: object
```

```
items_df = pd.json_normalize(df['rewardsReceiptItemList'])
```

```
# Print the normalized DataFrame for 'rewardsReceiptItemList'
print("\nNormalized rewardsReceiptItemList DataFrame:")
print(items_df)
```

```
↗ Normalized rewardsReceiptItemList DataFrame:
```

	0	1	2	3	4	
0	{'barcode': '4011', 'description': 'ITEM NOT F...	None	None	None	None	
1	{'barcode': '4011', 'description': 'ITEM NOT F...	None	None	None	None	
2	{'needsFetchReview': False, 'partnerItemId': '...	None	None	None	None	
3	{'barcode': '4011', 'description': 'ITEM NOT F...	None	None	None	None	
4	{'barcode': '4011', 'description': 'ITEM NOT F...	None	None	None	None	
...	
1114	{'barcode': 'B076FJ92M4', 'description': 'muel...	None	None	None	None	
1115	None	None	None	None	None	
1116	None	None	None	None	None	
1117	{'barcode': 'B076FJ92M4', 'description': 'muel...	None	None	None	None	
1118	None	None	None	None	None	
...	
0	None	None	None	None	None	
1	{'barcode': '028400642255', 'description': 'DO...	None	None	None	None	
2	None	None	None	None	None	
3	None	None	None	None	None	
4	{'barcode': '1234', 'finalPrice': '2.56', 'ite...	None	None	None	None	
...	
1114	{'barcode': 'B07BRRLSVC', 'description': 'thin...	None	None	None	None	
1115	None	None	None	None	None	
1116	None	None	None	None	None	
1117	{'barcode': 'B07BRRLSVC', 'description': 'thin...	None	None	None	None	
1118	None	None	None	None	None	
...	
0	None	None	None	None	None	
1	None	None	None	None	None	
2	None	None	None	None	None	
3	None	None	None	None	None	
4	None	None	None	None	None	
...	
1114	None	None	None	None	None	
1115	None	None	None	None	None	
1116	None	None	None	None	None	
1117	None	None	None	None	None	
1118	None	None	None	None	None	
...	
0	None	None	None	None	None	
1	None	None	None	None	None	
2	None	None	None	None	None	
3	None	None	None	None	None	
4	None	None	None	None	None	
...	
1114	None	None	None	None	None	
1115	None	None	None	None	None	
1116	None	None	None	None	None	
1117	None	None	None	None	None	
1118	None	None	None	None	None	

[1119 rows x 459 columns]

```
# Normalize rewardsReceiptItemList into a DataFrame
items_list = [item.get('rewardsReceiptItemList', []) for item in data]
items_df = pd.concat([pd.json_normalize(item) for item in items_list], ignore_index=True)

# Print the normalized DataFrame for 'rewardsReceiptItemList'
print("Normalized rewardsReceiptItemList DataFrame:")
print(items_df)

# If you want to analyze categorical values in normalized data:
# Identify columns with categorical data
categorical_columns = items_df.select_dtypes(include=['object']).columns

# Print list of categorical columns
print("\nCategorical columns in rewardsReceiptItemList:")
print(categorical_columns)

# Count distinct values in each categorical column
distinct_counts = items_df[categorical_columns].apply(pd.Series.nunique)

# Print distinct counts
print("\nNumber of distinct values in each categorical column in rewardsReceiptItemList:")
print(distinct_counts)
```



```
originalreceiptitemext    1/38
itemNumber                47
pointsEarned              277
targetPrice               2
competitiveProduct        2
originalFinalPrice        2
originalMetaBriteItemPrice 2
deleted                   1
priceAfterCoupon          334
metabriteCampaignId       75
dtype: int64
```