Data Model Overview

This data model is designed to support the analysis of receipt data, focusing on user activity, brand interactions, and temporal trends. The schema consists of one fact table and three dimension tables, which together provide a comprehensive structure for querying and reporting.The star schema, with its central fact table (FactReceipts) and surrounding dimension tables (DimUsers, DimBrands, DimDate), simplifies complex queries and enhances performance for analytical operations. This structure ensures efficient data retrieval, aggregation, and consistency, supporting scalable and flexible business intelligence analyses.

Cleaning and Data Quality check

Firstly, data from json files were read and converted to dataframe.For data cleaning na values of each data frame was checked. Coming to receipts dataframe columns that were not very useful such as 'bonusPointsEarned', 'bonusPointsEarnedReason', 'pointsEarned', 'pointsAwardedDate.$date', 'purchaseDate.$date','finishedDate.$date'were dropped inorder to preserve number of datapoints. But substantial number of data points were dropped based on 'purchasedItemCount', 'rewardsReceiptItemList'. As these columns were important for querying and data model. They cannot be dopped. It was noteworthy to see some duplicates in 'rewardsReceiptItemList' which were identified based on time and were terminated. While cleaning for brands dataframe 'categoryCode','topBrand','category' columns were dopped. And brandcodes were extracted from brandname. Hence there was no loss of datapoints for df_brands. Similarly for df_users 'active', 'role','state', 'signUpSource','lastLogin.$date' where dropped and no datapoints were lost.

To construct facts and dimensions table firstly create_dim_table function was created in order to structure data dimensions and start, end date were extracted in order to define date range. In order to develop the fact receipts table after much of evaluation it was noticed that pointsPayerId in 'rewardsReceiptItemList' is common with cog_id_oid in df_brands. It was noteworthy to see that 'rewardsReceiptItemList' was fully unstructured and hence pointsPayerId was extracted from it. It was used to left join on df_brands. After renaming and standardising columns , by leveraging sqlilite dataframes were converted into tables and stored in recipes db.

Queries

"Top 5 brands by receipts scanned for the most recent month":

```
    """
    SELECT b.name, COUNT(*) as scan_count
    FROM FactReceipts fr
    JOIN DimDate dd ON fr.dateScannedId = dd.dateId
    JOIN FactReceiptItems fri ON fr.receiptId = fri.receiptId
    JOIN DimBrands b ON fri.brandId = b.brandId
    WHERE dd.date >= DATE('now', '-1 month')
```

```
GROUP BY b.name
ORDER BY scan_count DESC
LIMIT 5
""",

"Ranking of the top 5 brands by receipts scanned for the previous month":
"""
SELECT b.name, COUNT(*) as scan_count
FROM FactReceipts fr
JOIN DimDate dd ON fr.dateScannedId = dd.dateId
JOIN FactReceiptItems fri ON fr.receiptId = fri.receiptId
JOIN DimBrands b ON fri.brandId = b.brandId
WHERE dd.date >= DATE('now', '-2 months') AND dd.date < DATE('now', '-1 month')
GROUP BY b.name
ORDER BY scan_count DESC
LIMIT 5
""",

"Average spend from receipts with 'Accepted' or 'Rejected' status":
"""
SELECT rewardsReceiptStatus, AVG(totalSpent) as avg_spent
FROM FactReceipts
WHERE rewardsReceiptStatus IN ('Accepted', 'Rejected')
GROUP BY rewardsReceiptStatus
""",

"Total number of items purchased from receipts with 'Accepted' or 'Rejected' status":
"""
SELECT fr.rewardsReceiptStatus, SUM(fri.quantity) as total_items
FROM FactReceipts fr
JOIN FactReceiptItems fri ON fr.receiptId = fri.receiptId
WHERE fr.rewardsReceiptStatus IN ('Accepted', 'Rejected')
GROUP BY fr.rewardsReceiptStatus
""",

"Brand with the most spend among users created within the past 6 months":
"""
SELECT b.name, SUM(fri.price * fri.quantity) as total_spent
FROM FactReceipts fr
JOIN DimUsers du ON fr.userId = du.userId
JOIN FactReceiptItems fri ON fr.receiptId = fri.receiptId
JOIN DimBrands b ON fri.brandId = b.brandId
WHERE du.createdDateId >= strftime('%Y%m%d', 'now', '-6 months')
GROUP BY b.name
```

```
ORDER BY total_spent DESC
LIMIT 1
""",

"Brand with the most transactions among users created within the past 6 months":
"""
SELECT b.name, COUNT(*) as transaction_count
FROM FactReceipts fr
JOIN DimUsers du ON fr.userId = du.userId
JOIN FactReceiptItems fri ON fr.receiptId = fri.receiptId
JOIN DimBrands b ON fri.brandId = b.brandId
WHERE du.createdDateId >= strftime('%Y%m%d', 'now', '-6 months')
GROUP BY b.name
ORDER BY transaction_count DESC
LIMIT 1
"""
```