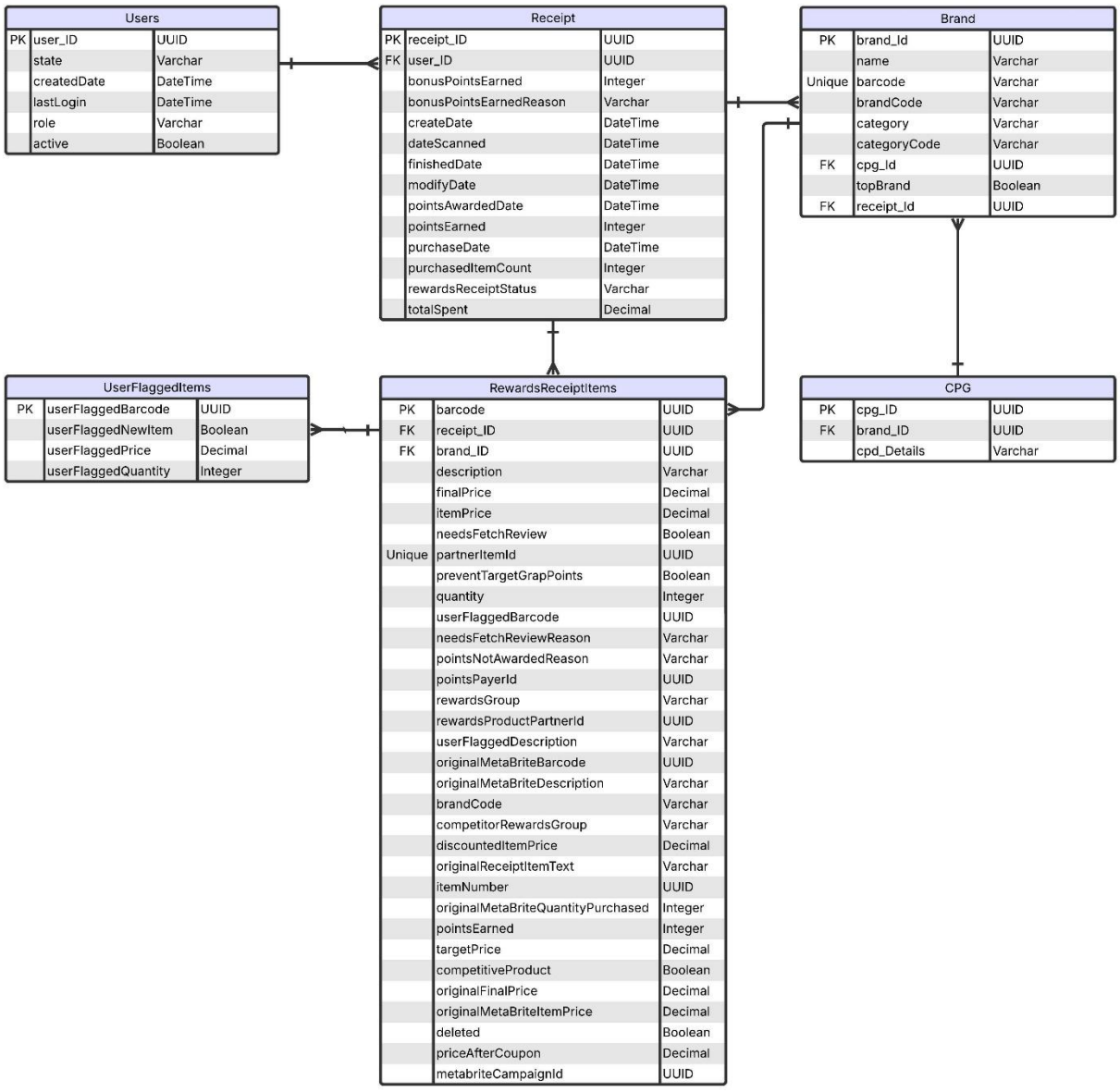
Fetch Rewards-Analytical Engineer Internship

First: Review Existing Unstructured Data and Diagram a New Structured Relational Data Model



Second: Write queries that directly answer predetermined questions from a business stakeholder

1. What are the top 5 brands by receipts scanned for most recent month?

```
SELECT rri.brandCode, COUNT(*) AS brand_count
FROM Receipts r
Inner JOIN RewardsReceiptItems rri ON r.receipt_id = rri.receipt_id
WHERE strftime('%Y-%m', r.dateScanned) = (
SELECT strftime('%Y-%m', MAX(dateScanned))
FROM Receipts)
GROUP BY rri.brandCode
ORDER BY brand_count DESC
LIMIT 5;
```

In the most recent calendar month from the data, the brandCode is None for all the entries. So, inorder to fetch the top 5 brands from the last 28 days (February 2021 has only 28 days), I slightly modified the query to fetch data from 30 days before the last scanned date. So, by executing this query, the top brands I extracted are BRAND, MISSION, and VIVA.

```
SELECT rri.brandCode, COUNT(*) AS brand_count
FROM Receipts r
INNER JOIN RewardsReceiptItems rri ON r.receipt_id = rri.receipt_id
WHERE strftime('%Y-%m-%d %H:%M:%S', dateScanned) >= strftime('%Y-%m-%d %H:%M:%S',
               date((SELECT MAX(dateScanned) FROM df_receipts), '-28 days'))
FROM Receipts)
GROUP BY rri.brandCode
ORDER BY brand_count DESC
LIMIT 5;
```

2. How does the ranking of the top 5 brands by receipts scanned for the recent month compare to the ranking for the previous month?

```
SELECT rri.brandCode, COUNT(*) AS brand_count
FROM Receipts r
INNER JOIN RewardsReceiptItems rri ON r.receipt_id = rri.receipt_id
WHERE strftime('%Y-%m', r.dateScanned) = (
    SELECT strftime('%Y-%m', date(MAX(dateScanned), '-1 month'))
FROM Receipts)
GROUP BY rri.brandCode
ORDER BY brand_count DESC
LIMIT 5;
```

This shows the top brands from the previous calendar month based on the number of receipts scanned. But, due to the brandCode being None for all the entries in the last calendar month, I modify the query to extract data from the previous month used in the above query.

```
SELECT rri.brandCode, COUNT(*) AS brand_count
FROM Receipts r
INNER JOIN RewardsReceiptItems rri ON r.receipt_id = rri.receipt_id
WHERE strftime('%Y-%m-%d %H:%M:%S', dateScanned) <= strftime('%Y-%m-%d %H:%M:%S',
date((SELECT MAX(dateScanned) FROM df_receipts), '-28 days'))
    AND   strftime('%Y-%m-%d %H:%M:%S', dateScanned) >= strftime('%Y-%m-%d
%H:%M:%S', date((SELECT MAX(dateScanned) FROM df_receipts), '-62 days'))
GROUP BY rri.brandCode
ORDER BY brand_count DESC
LIMIT 5;
```

3. When considering average spend from receipts with 'rewardsReceiptStatus' of 'Accepted' or 'Rejected', which is greater?

```
SELECT rewardsReceiptStatus, AVG(totalSpent) AS average_spend
FROM Receipts
WHERE RewardsReceiptStatus IN ('FINISHED', 'REJECTED')
GROUP BY rewardsReceiptStatus;
```

rewardsReceiptStatus average_spend

0 FINISHED 80.854305

1 REJECTED 23.326056

So, the average spend from the receipts from "ACCEPTED" rewardsReceiptStatus is higher than "REJECTED" status.

4. When considering total number of items purchased from receipts with 'rewardsReceiptStatus' of 'Accepted' or 'Rejected', which is greater?

```
SELECT rewardsReceiptStatus, SUM(purchasedItemCount) AS total_items_purchased
FROM Receipts
WHERE rewardsReceiptStatus IN ('FINISHED', 'REJECTED')
GROUP BY rewardsReceiptStatus;
```

rewardsReceiptStatus total_items_purchased

0 FINISHED 8184.0

1 REJECTED 173.0

Comparing the total number of items purchased, the receipts with "ACCEPTED" rewardsReceiptStatus is higher.

5. Which brand has the most spend among users who were created within the past 6 months?

```
SELECT rri.brandCode, SUM(r.totalSpent) as totalSpent
FROM RewardsReceiptItems rri
JOIN Receipts r ON rri.receipt_id = r.receipt_id
WHERE r.user_id IN (
      SELECT DISTINCT user_id
      FROM Users
      WHERE createdDate >= (
      SELECT strftime('%Y-%m-%d', MAX(createdDate), '-6 months') FROM Users)
    )
GROUP BY rri.brandCode
ORDER BY totalSpent DESC
LIMIT 1;
```

	brandCode	totalSpent
0	None	2930282.42
1	BEN AND JERRYS	197337.68

Since most of the brandcode entries are None, ignoring them, the brand with most spend among the users in the last 6 months is BEN AND JERRYS.

6. Which brand has the most transactions among users who were created within the past 6 months?

```
SELECT rew.brandCode, COUNT(*) AS transaction_count
FROM RewardsReceiptItems rew
JOIN df_receipts r ON rew.receipt_id = r.receipt_id
WHERE r.user_id IN (
      SELECT DISTINCT user_id
      FROM Users
      WHERE createdDate >= (
           SELECT strftime('%Y-%m-%d', MAX(createdDate), '-6 months') FROM Users)
      )
GROUP BY rew.brandCode
ORDER BY transaction_count DESC
LIMIT 2;
```

brandCode	transaction_count
None	2412
HY-VEE	291

Since most of the brandcode entries are None, ignoring them, the brand with most number transactions among the users in the last 6 months is HY-VEE.

Third: Evaluate Data Quality Issues in the Data Provided

1. On analyzing the data provided, I figured out that the data is very **noisy, unstructured** and contains a lot **of missing and dumped** values. The proportion of missing and dumped data were really high in Receipts and Brands Tables when compared to Users table.

   Depending on the use-case scenarios and nature of the null-value columns, statistical data imputation and predictive imputation techniques can be employed.

2. The date formats were **not consistent** across the different tables. I wrote methods to make all the date formats consistent across the tables.
3. All the UUID key columns were given as a **nested json objects**. So, I wrote methods to extract the data and store them in separate columns and also created new tables as required to improve performance and scalability.
4. The Users table contains **>50% of duplicate rows**.
5. The columns 'pointsAwardedDate' and 'pointsEarned' are **not consistent** with the 'rewardsReceiptStatus'. The "REJECTED" receipt status also earned points to the user which is logically wrong.
6. The **data inconsistencies lead to slow query matching and increased data processing time**. As the scale of the data grows, this can escalate to be a critical impact on the data pipeline.
7. The duplicate rows tampers the **scalability of the storage systems** and also delays query processing.
8. In distributed processing and Machine learning data pipelines, the addressing the above issues play a significant factor in improving the performance of the systems.
9. **Standard data governance policies and data format standardization** has to be mandated to get rid of data quality issues.
10. **Outlier detection methods** can be implemented to streamline data profiling on customer-centric monetization data.

Data Quality evaluation analysis can be found in the attached .ipynb file.

Fourth: Communicate with Stakeholders

Hello,

This is Arun Vignesh Malarkkan, an Analytics Engineer Intern at Fetch Rewards. I hope you are doing well. I have been working on building a new streamlined data pipeline for our Analytics Engine product and I have some important insights and concerns regarding the data we are using currently. I conducted a detailed statistical and code-based checks on the provided tables and I identified a few crucial quality issues that may impact our decision-making and the reliability of our framework. I wanted to share these findings and also discuss the potential solutions that can be implemented moving forward.

- **Data Completeness**: I have identified few key columns have missing values in them like the “brandCode” in Brands table, “pointsEarned” and “bonusPointsEarned” in the Receipts table, “brandCode” and “finalPrice” in the RewardSReceiptItemList table. These could significantly lower the accuracy of the analysis when we are trying to assess the top brands and the most reward points earned by the users.
- **Data Consistency**: There is a significant need of data standardization protocols across the tables as there are a lot of data inconsistencies and they significantly impact the query performance and their reliability. For eg: The columns ‘pointsAwardedDate’ and ‘pointsEarned’ are **not consistent** with the ‘rewardsReceiptStatus’. The “REJECTED” receipt status also earned points to the user which is logically wrong. These aspects need more data validation strategies implied on them.
- There are also some illogical values across different columns in the **RewardsReceiptItems table** extracted from the Receipts table which needs more cleaning to avoid misleading insights on the reward points.
- Also, the **“brandCode” value is “None”** for the majority of the entries which invalidates our analysis results on crucial business questions.

Given these observations, I would also appreciate your assistance on some important questions I have.

1. Do we have any existing validation systems in-place to prevent missing or inconsistent data to enter the pipeline?
2. As brand attribution to reward points is an important aspect of our business model, how do we tackle the “brandCode” value being “None” and is the Financial and Vendor teams aware of these issues?
3. Do we have any “Date&Time” standardization policies in-place? If so, what is the standard timezone we utilize across our products?

It will be really helpful if I can know the additional following information to make the framework optimized and scalable with respect to memory and computational overhead.

1. Full lifecycle of the Backend Software System handling Transactions and Reward Point Assignments.
2. Current Data caching and Distributed data warehouses.
3. Current Data Indexing and Query Optimization strategies.
4. Existing Key metrics and System Performance.
5. Expected database usage rate.

I'd like to schedule a brief meeting to discuss these findings and align on the next steps. Please let me know what time works best for you. Thank you.

Regards,
Arun Vignesh Malarkkan