Analytics Engineer Coding Exercise

Part 1: Review Existing Unstructured Data and Diagram a New Structured Relational Data Model

Assuming the data model is designed for analytics purposes (OLAP). Here we have 3 datasets: Receipts, Users, and Brands.

For Receipts data, each receipt is an individual transaction by a user, we could use the **receipts** table to store receipts information. A receipt could contain multiple items/brands, so the most granular data is each purchased item. To support millions of transactions efficiently, avoid data redundancy and ensure data integrity, we could use the **receipt_items** bridge table to handle the one-to-many relationship between receipt and purchased items

For Users data, we could use the **users** table to store user information. One user can have multiple receipts (one-to-many relationship between users and receipts).

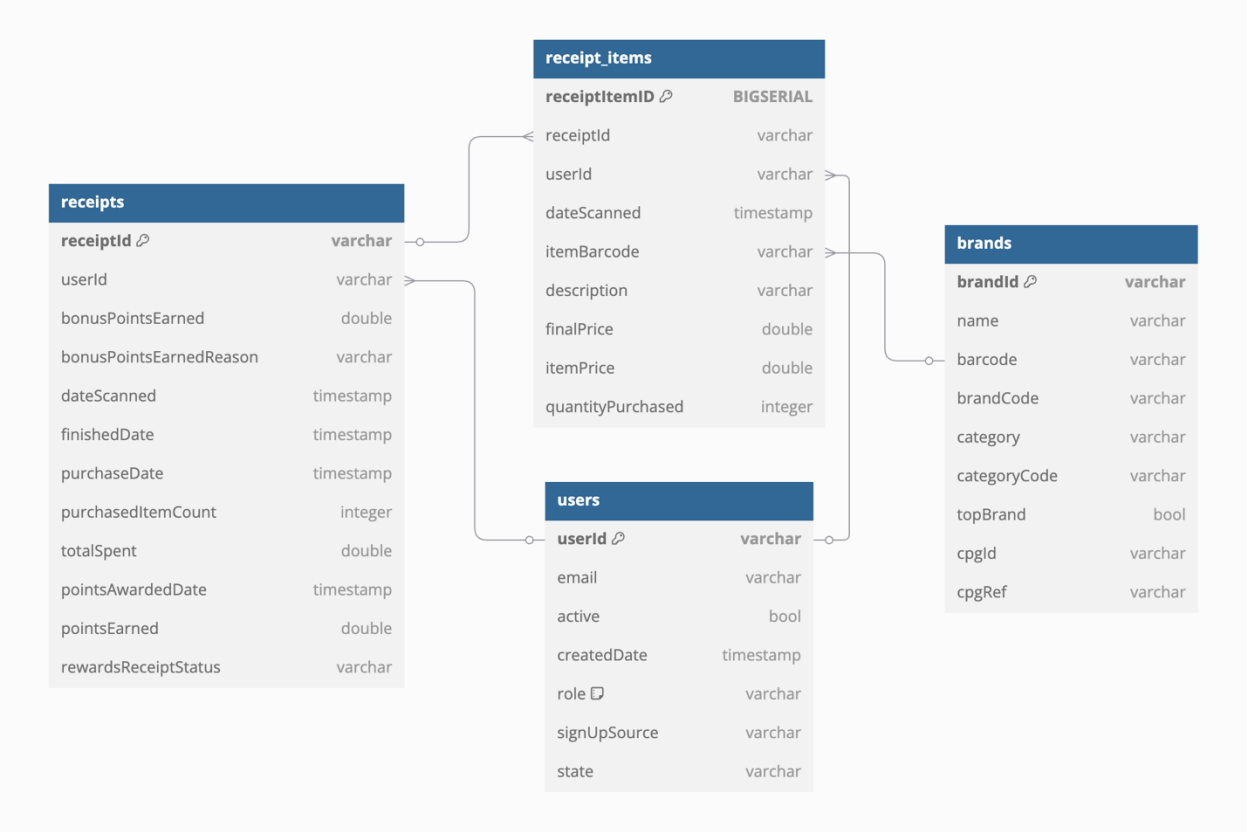
For Brands data, we could use the **brands** table to store brand information. One brand can be in multiple receipts, and one receipt can contain multiple brands (many-to-many relationship between receipts and brands)

Hence, the data model has four tables: **users**, **brands**, **receipts**, **receipt_items**.

```
Table receipts {
  receiptId varchar [pk]
  userId varchar [ref: > users.userId]
  bonusPointsEarned double
  bonusPointsEarnedReason varchar
  dateScanned timestamp
  finishedDate timestamp
  purchaseDate timestamp
  purchasedItemCount integer
  totalSpent double
  pointsAwardedDate timestamp
  pointsEarned double
  rewardsReceiptStatus varchar
}
Table receipt_items {
  receiptItemID BIGSERIAL PK  // Surrogate key
  receiptId varchar [ref: > receipts.receiptId]
  userId varchar [ref: > users.userId]
  dateScanned timestamp
  itemBarcode varchar [ref: > brands.barcode]
  description varchar
  finalPrice double
  itemPrice double
  quantityPurchased integer
}
```

```
Table users {
  userId varchar [pk]
  email varchar [unique]
  active bool
  createdDate timestamp
  role varchar [default: 'CONSUMER']
  signUpSource varchar
  state varchar
}
Table brands {
  brandId varchar [pk]
  name varchar
  barcode varchar [unique]
  brandCode varchar
  category varchar
  categoryCode varchar
  topBrand bool
  cpgId varchar
  cpgRef varchar
}
```

The ERD diagram is as follows:



Note:

- The dates in Json files are unixtime. Convert them to timestamp without time zone when loading to above tables.
- Use BIGSERIAL (Auto-Incremented Integer) as surrogate key for the **Receipt_Items** bridge table.
- Though there's no email information in the Users dataset, it's better to add an email column as a dimension table for users.

Part 2: Write Queries that Directly Answer Predetermined Questions from a Business Stakeholder

Note: here use PostgreSQL to write the queries.

```sql
-- What are the top 5 brands by receipts scanned for most recent month?
WITH recent_month AS (
  SELECT
    receiptId,
    dateScanned,
    itemBarcode
  FROM receipt_items
  WHERE dateScanned BETWEEN
        DATE_TRUNC('month', CURRENT_DATE) - INTERVAL '1 month' AND
        DATE_TRUNC('month', CURRENT_DATE)
)
SELECT
  b.name AS brand_name,
  COUNT(DISTINCT r.receiptId) AS receipts_scanned
FROM recent_month r
JOIN brands b ON r.itemBarcode = b.barcode
GROUP BY b.name
ORDER BY receipts_scanned DESC
LIMIT 5;



-- How does the ranking of the top 5 brands by receipts scanned for the recent month
compare to the ranking for the previous month?
WITH recent_month AS (
  SELECT receiptId, dateScanned, itemBarcode
  FROM receipt_items
  WHERE dateScanned BETWEEN
        DATE_TRUNC('month', CURRENT_DATE) - INTERVAL '1 month' AND
        DATE_TRUNC('month', CURRENT_DATE)
),

previous_month AS (
```

```
  SELECT receiptId, dateScanned, itemBarcode
  FROM receipt_items
  WHERE dateScanned BETWEEN
        DATE_TRUNC('month', CURRENT_DATE) - INTERVAL '2 month' AND
        DATE_TRUNC('month', CURRENT_DATE) - INTERVAL '1 month'
),

recent_rank AS (
SELECT *
FROM (
  SELECT b.name AS brand_name,
         COUNT(DISTINCT r.receiptId) AS receipts_scanned,
         RANK() OVER (ORDER BY COUNT(DISTINCT r.receiptId) DESC) AS recent_rank
  FROM recent_month r
  JOIN brands b ON r.itemBarcode = b.barcode
  GROUP BY b.name
) t
WHERE recent_rank <= 5
),

previous_rank AS (
SELECT *
FROM (
  SELECT b.name AS brand_name,
         COUNT(DISTINCT r.receiptId) AS receipts_scanned,
         RANK() OVER (ORDER BY COUNT(DISTINCT r.receiptId) DESC) AS previous_rank
  FROM previous_month r
  JOIN brands b ON r.itemBarcode = b.barcode
  GROUP BY b.name
) t
WHERE previous_rank <= 5
)

SELECT r.brand_name,
       r.receipts_scanned AS recent_receipts_scanned,
       p.receipts_scanned AS previous_receipts_scanned,
       r.recent_rank,
       p.previous_rank
FROM recent_rank r
FULL OUTER JOIN previous_rank p ON r.brand_name = p.brand_name
ORDER BY r.recent_rank;


-- When considering average spend from receipts with 'rewardsReceiptStatus’ of
‘Accepted’ or ‘Rejected’, which is greater?
-- This query returns the avg spend by status and order by the avg spend in descending
order
SELECT
```

```sql
  rewardsReceiptStatus,
  AVG(totalSpent) AS avg_spend
FROM receipts
WHERE rewardsReceiptStatus IN ('Accepted', 'Rejected')
GROUP BY rewardsReceiptStatus
ORDER BY avg_spend DESC;


-- When considering total number of items purchased from receipts with
'rewardsReceiptStatus’ of ‘Accepted’ or ‘Rejected’, which is greater?
-- This query returns the total number of items purchased by status and order by the
total number of items purchased in descending order
SELECT
  rewardsReceiptStatus,
  SUM(purchasedItemCount) AS total_items_purchased
FROM receipts
WHERE rewardsReceiptStatus IN ('Accepted', 'Rejected')
GROUP BY rewardsReceiptStatus
ORDER BY total_items_purchased DESC;


-- Which brand has the most spend among users who were created within the past 6
months?
WITH cte AS (
  SELECT
    b.name AS brand_name,
    SUM(r.finalPrice) AS total_spend
  FROM users u
  JOIN receipt_items r ON u.userId = r.userId
  AND u.createdDate >= CURRENT_DATE - INTERVAL '6 months'
  JOIN brands b ON r.itemBarcode = b.barcode
  GROUP BY b.name
)

SELECT *
FROM cte
WHERE total_spend = (SELECT MAX(total_spend) FROM cte);


-- Which brand has the most transactions among users who were created within the past
6 months?
-- note: I'm assuming transactions refer to the number of receipts scanned
WITH cte AS (
  SELECT
    b.name AS brand_name,
    COUNT(DISTINCT r.receiptId) AS total_transactions
  FROM users u
  JOIN receipt_items r ON u.userId = r.userId
```

```
  AND u.createdDate >= CURRENT_DATE - INTERVAL '6 months'
  JOIN brands b ON r.itemBarcode = b.barcode
  GROUP BY b.name
)

SELECT *
FROM cte
WHERE total_transactions = (SELECT MAX(total_transactions) FROM cte);
```

Part 3: Evaluate Data Quality Issues in the Data Provided

Below is a list of checked items, please refer to the data_evaluation.ipynb for details.

Quality Checklist:

- Duplicate Records: check for duplicate rows or duplicate values in primary key columns
- Missing Values: check for nulls or blanks in critical fields
- Outliers: check extreme values in numerical variables such as pointsEarned, bonusPointsEarned, totalSpent, and purchasedItemCount
- Logical Accuray:
    - createdDate should be before or equal to lastLogin in Users
    - numerical fields such as pointsEarned, bonusPointsEarned, and totalSpent should >= 0
    - purchasedItemCount in Receipts should match the sum of quantityPurchased for the corresponding receiptId in Receipt_Items
    - totalSpent should match the sum of finalPrice for the corresponding receiptId in Receipt_Items

Data Quality Issues Found:

1. **Users**
    - More than half (283 out of 495) of the user records are duplicate records.
    - There are a few missing values in lastLogin, signUpSource, and state columns.

2. **Brands**
    - There are large number of missing values in topBrand and categoryCode columns.

3. **Receipts**
    - There are large numbers of missing values in bonusPointsEarned, bonusPointsEarnedReason, pointsEarned, purchasedItemCount, rewardsReceiptItemList, totalSpent, finishedDate, pointsAwardedDate, purchaseDate.

However, most missing values are in the receipts of status equal to ‘SUBMITTED’ or ‘PENDING’, which indicates these missing values are due to the receipt processing is not finished yet.

For receipts with status of 'FINISHED', there are still a few missing values in bonusPointsEarned, it is beacuse there is no bonus points, i.e. the value is 0? If it is, suggest replacing null with 0.

- For some receipts with a status of ‘REJECTED’, there are bonusPointsEarned. What does the Status mean? It sounds like it should be 0 points to me if rejected.
- There are a lot of extreme large values in purchasedItemCount, totalSpent, and pointsEarned. These outliers need to be reviewed to make sure they are not errors.

- More than half of the purchased items in the rewardsReceiptItemList have no barcode. For some items have barcode, the description is ‘ITEM NOT FOUND’. It could be the barcode is not correct, or the item is missing in our brands data.

- There are 40 records found that the sum of quantityPurchased of all items doesn’t match the purchasedItemCount of the receipt. There are 13 records found that the sum of finalPrice of all items doesn’t match the totalSpent of the receipt.

Part 4: Communicate with Stakeholders

Hi [Stakeholder's Name],

I hope you're doing well. As part of our ongoing data quality evaluation, I have analyzed the users, brands, and receipts datasets. Below is a summary of the key issues identified and suggestions for improvement. I would appreciate your feedback and any additional insights you might have.

Summary of Data Quality Issues:

1. Users

- **Duplicate Records:**
  - More than half (283 out of 495) of the user records are duplicates.
- **Missing Values:**
  - Missing values were found in lastLogin, signUpSource, and state columns.

2. Brands

- **Incomplete Data:**
  - A large number of missing values in topBrand and categoryCode columns.

3. Receipts

- **Missing Values:**
  - Significant missing values in bonusPointsEarned, bonusPointsEarnedReason, pointsEarned, purchasedItemCount, rewardsReceiptItemList, totalSpent, finishedDate, pointsAwardedDate, and purchaseDate.
  - Most of these are associated with receipts in 'SUBMITTED' or 'PENDING' status, suggesting they are still being processed.
  - For receipts with a status of 'FINISHED', there are a few missing values in bonusPointsEarned. Is this due to no bonus points being awarded? If so, I suggest replacing null values with 0.
- **Status Inconsistencies:**
  - Some receipts marked as 'REJECTED' have bonusPointsEarned. Could you clarify if this is expected? If rejected receipts should not earn points, this may need correction.
- **Outliers:**
  - Extreme values were observed in purchasedItemCount, totalSpent, and pointsEarned. These outliers should be reviewed to confirm they are not data errors.
- **Incomplete Item Information:**
  - Over half of the purchased items in rewardsReceiptItemList are missing barcodes. Additionally, some items with barcodes have a description of 'ITEM NOT FOUND'. This may indicate incorrect barcodes or missing data in our brands database.
- **Inconsistent Summation:**
  - 40 records show a mismatch between the sum of quantityPurchased of all items and the purchasedItemCount of the receipt.
  - 13 records show a discrepancy between the sum of finalPrice of all items and the totalSpent of the receipt.

Suggestions for Improvement:

Assuming our data model is designed for analytics purposes (OLAP), here are some recommended actions:

Receipts Data:

- Store all receipt statuses (from 'SUBMITTED' to 'FINISHED'/'REJECTED') in a data lake for auditing and historical tracking.
- Assuming the final status of a receipt is either FINISHED' or 'REJECTED'. Only load receipts with statuses of 'FINISHED' or 'REJECTED' into the proposed receipts and receipt_items tables, as these statuses indicate that the processing is complete.
- Implement checks to ensure that the sum of quantityPurchased matches purchasedItemCount, and the sum of finalPrice matches totalSpent for each finished receipt.

- Extreme values observed in totalSpent, pointsEarned, and bonusPointsEarned are skewing data distributions and potentially affecting reporting accuracy. Define and implement rules for detecting and handling outliers, possibly by setting reasonable thresholds based on historical trends.
- For inconsistencies and missing values in key fields such as rewardsReceiptStatus and rewardsReceiptItemList, implement data validation checks at the point of entry or during data ingestion to improve consistency.

Users Table:

- As this is a dimension table, avoid storing lastLogin information. If tracking user logins is needed, consider creating a separate fact table to capture user activities, including columns like user_id, user_activity, and timestamp.
- Ensure no missing values and consider enriching user information by adding columns such as user_email and zip_code to support demographic analysis and user segmentation.

Brands Table:

- Ensure there are no missing values in this dimension table.
- Verify that all purchased items in the receipts can be mapped to a record in the brands table, allowing accurate analysis of rewards by brand category.

Performance and Scaling Considerations:

As the receipts and receipt_items tables continue to grow, we need to maintain query performance by:

- Creating appropriate indexes.
- Implementing partitioning strategies.
- Designing pre-aggregated tables (e.g., receipts/purchased items statistics by month).

Questions and Next Steps:

1. Can you clarify whether the missing bonusPointsEarned for 'FINISHED' receipts means no bonus points were awarded? If so, I will proceed with replacing null values with 0.
2. Should receipts marked as ‘REJECTED’ have any points awarded? This will guide how we handle those records.
3. Are there specific business rules governing the population of topBrand, categoryCode, and other missing fields?
4. Are there any other performance considerations or reporting requirements that we should be aware of before implementing these changes?

I would appreciate your feedback on the issues and suggestions outlined above. If needed, I am happy to discuss this in more detail.

Looking forward to your insights.