ER Model Documentation

VERTICA SQL

Introduction

This document outlines the Entity-Relationship (ER) model designed for the provided unstructured JSON data - Receipt, Users and Brand. The aim of the model is to structure the data into a relational database format, optimizing it for analysis and reporting purposes.

Key Understandings

Data Sources

The ER model is based on three primary data sources that are provided with the assessment:

1. **Receipts Data (receipts.json.gz)**: This dataset contains detailed information about each user's receipt, including products, bonuses, points earned, dates, status, total spent, and user ID etc.
2. **Users Data (users.json.gz)**: This dataset contains user specific information, including account details and account status.
3. **Brands Data (brands.json.gz)**: Contains information about brands, associated products, including brand codes, categories, and names.

Objective

The goal of structuring this data into a relational model is to:

- Facilitate efficient querying and analysis.
- Ensure data integrity and consistency.
- Provide a scalable framework for future data additions.

Assumptions

SQL Dialect: The model and queries are based on Vertica SQL.

Business logic assumptions:

1. **State Information**: The attribute `stateId` is incorporated in `User_Log` assuming that this information is recorded by session. If it is a static value associated with a user upon account registration, it should not be in the same table.
2. **Item Pricing**: `item_price` is assumed to be partner/brand provided information, while `final_price` could be an adjusted amount from Fetch.
3. **Brand-Category Relationship**: I am assuming that one brand can have many categories.
4. **User Active Status**: The `userActiveStatus.active` attribute is assumed to be account status and not login status.
5. **Brand Data**: The provided brand data is assumed to be partial, as no direct link exists between brand attributes and receipt or user data in the provided dataset.

Relationship/Data Type assumptions:

6. **Attributes Not Null**: Attributes are assigned as not null based on my current understanding of the data. This can be revised according to business needs and how the data is ingested, stored in raw tables, and received from partners.
7. **Receipt Data Breakdown**: Receipt data is broken down into two fact tables: `Receipt_Fact` and `Bonus_Fact`.
8. **Detailed Receipt Information**: Detailed receipt information is captured in the `Detailed_Receipt` table, linked via `receipt_id`.
9. **Nested Information**: Nested information within the `rewardReceiptItemList` attribute is captured in a separate `rewardReceiptItemList` table.
10. **Fetch Review**: The `Fetch_Review` table captures whether a receipt needs review, linked via `needsFetchReviewId`.
11. **User Reported Items**: User-related metrics and attributes from `rewardReceiptItemList` are captured in the `User_Receipt_Reported_Items` table.
12. **Receipt Status**: Receipt status information is captured in the `Receipt_Status` table, linked via `rewardsReceiptStatusId`.
13. **Bonus Points Details**: Bonus points details are captured in the `Bonus_Detail` table, linked via `bonusPointsEarnedReasonId`.
14. **User Dimension**: User-related information is captured in the `Users_Dimension` table, which can be joined to other dimension tables for additional details.
15. **User Log**: The `User_Log` table captures session-based state information, assuming it is recorded per session rather than a static account attribute.

ER Diagram assumptions:

16. **ER Diagram Grouping**: Rectangular boxes in the ER diagram group tables of the same or related entities.

Model Design

Table Structures

1. **Receipts Data -**
    a. **Reciepts_Fact** - this summarized contains receipt associated facts
    b. **Detailed_Receipt** - this contains detailed qualitative receipt information
    c. **Receipt_ItemList** - contains nested receipt item information on product level granularity
    d. **User_Receipt_Reported_Items** - supplements Receipt_ItemList table with user-reported metrics and attributes
    e. **Fetch_Review** - Fetch associated data for receipt review
    f. **Receipt_Status** - this table contains receipt status information
    g. **Bonus_Fact** - this fact table contains bonus-related information
    h. **Bonus_Detail** - this table supplements additional granularity to bonus information
2. **Users Data -**
    a. **Users_Dimension** - this table contains user-specific information and links to additional tables with user related information
    b. **User_ActiveStatus** - this table is designed assuming active status is associated to account and not session
    c. **User_Log** - this table is designed to provide user session information (here stateId is assumed to be recorded for each session, if otherwise, it stateId should not be considered for this table)
    d. **User_Role** - this table defines user role, per the data description, always set to 'consumer'
    e. **User_SignUpSource** - this table provides information about user's sign up source.
3. **Geo data -**
    a. **State_Dimension** - this table is added to supplement the stateId information in the User_Log table
4. **Brand -**
    a. **Brand_Dimension** - this table identifies brand associated information and maps to other qualitative information
    b. **Brand_Information** - provides details related to brand name
    c. **Brand_Category** - provides details related to brand category
    d. **Top_Brands** - this table provides information whether a brand is categorized as top brand or not
    e. **Product_Information** - this table details product associated qualitative information.

Design Considerations

There are three primary considerations from my end while designing the model:

1. Normalization

The model has been normalized to reduce redundancy and improve data integrity.

2. Indexing

All Primary Key and Foreign Key attributes will be indexed to improve query performance and joins.

3. Data Quality

Following steps are crucial to ensure a high data quality standard in the model:

- Setting strict validation rules during data ingestion.
- Regular audits and checks to identify and rectify data anomalies.
- Standardization of data formats across the warehouse.

4. Detailed and Aggregated Data

The model I designed is detailed and granular for the warehouse, which is essential for deep and thorough analysis and custom reporting. However, based on the needs of the data and analytical teams, creating aggregated schemas or pipelines may be beneficial. These aggregated schemas provide pre-processed, summarized data can provide a quick and efficient way to get data that is queried most frequently for analysis and reporting.

Potential Enhancements

1. **Additional Tables**: Future inclusion of more detailed tables, such as a `ReceiptItems` table to capture individual items on each receipt.
2. **Enhanced Relationships**: Defining more complex relationships, such as many-to-many relationships, if needed for advanced analysis.
3. **Historical Data Handling**: Implementing strategies to handle historical data changes without losing historical accuracy.

Performance and Scalability

1. Indexing and Partitioning

Indexing these tables on PK+FKs to speed up query processing. If needed, for large datasets, partitioning can also be utilized.

2. Query Optimization

Query Plans and frequent query requests can help pave way to revise the data model

Conclusion

While the model I designed is based on my assumptions, it should serve as a solid foundation for iterative development. By adhering to and following best practices, using indexing, normalization and data modeling, the model presented ensures data integrity, consistency and scalability.

Future iterations of this model should incorporate additional business needs and data insights to further improve and refine the model. At the end of the day, the flexibility to adapt the model to meet business needs is important for driving data-driven growth in the company. Based on the needs of the data and analytical teams, creating aggregated schemas or pipelines may be beneficial