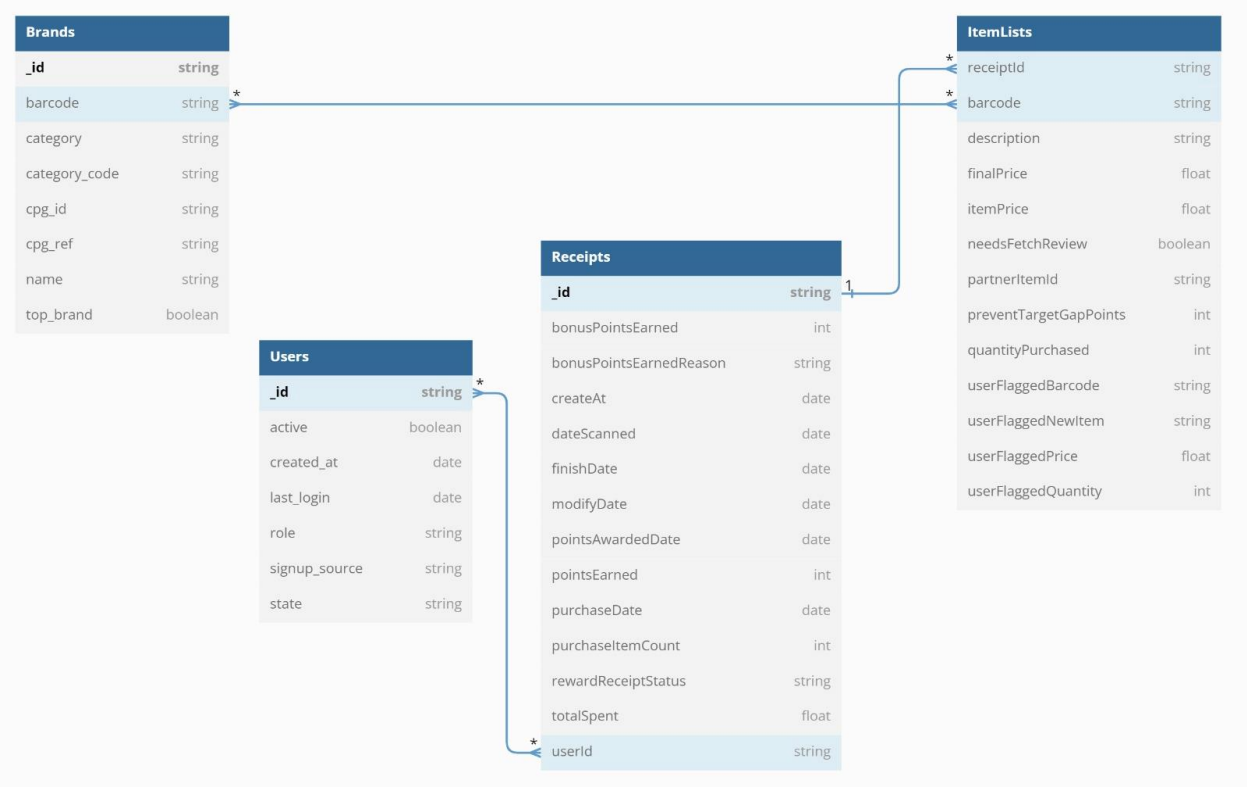
Fetch Take-home

Snowflake Creds:
Username – susreddy2503
Password – Susmitha123
Database - FETCH_REWARDS
Schema – USERS_DATA
Link - https://app.snowflake.com/ap-southeast-1/bq03971/w1g5EFYFBviL#query

First: ERD Diagram



CREATE INTERNAL STAGE AND LOAD JSON DATA

```
use fetch_rewards;
use schema users_data;

list @fetch_internal_stage;

create or replace file format json_format
   type = json
   strip_outer_array = true
;

create or replace table json_users (
Json_data variant
```

```
);

create or replace table json_brands (
Json_data variant
);

create or replace table json_receipts (
Json_data variant
);

copy into json_users
from @fetch_internal_stage/users.json
file_format = (format_name = JSON_FORMAT);

copy into json_receipts
from @fetch_internal_stage/receipts.json
file_format = (format_name = JSON_FORMAT);

copy into json_brands
from @fetch_internal_stage/brands.json
file_format = (format_name = JSON_FORMAT);
```

CREATE THE TABLES

```
create or replace table brands as
select
   flatten_id.VALUE::string as _id
   , JSON_DATA:barcode::string as barCode
   , JSON_DATA:brandCode::string as brandCode
   , JSON_DATA:category::string as category
   , JSON_DATA:categoryCode::string as categoryCode
   , JSON_DATA:cpg::string as cpg
   , JSON_DATA:name::string as name
   , JSON_DATA:topBrand::string as topBrand
from json_brands
,LATERAL FLATTEN (input => JSON_DATA:_id) as flatten_id

create or replace table users as
select
   flatten_id.VALUE::string as _id
   , JSON_DATA:state::string as state
   , JSON_DATA:createdDate as createdDate
   , JSON_DATA:lastLogin as lastLogin
   , JSON_DATA:role::string as role
   , JSON_DATA:signUpSource::string as signUpSource
   , JSON_DATA:active::boolean as active
from json_users
```

```
,LATERAL FLATTEN (input => JSON_DATA:_id) as flatten_id;

create or replace table receipts as
select
   flatten_id.VALUE::string as _id
   , JSON_DATA:bonusPointsEarned::float as bonusPointsEarned
   , JSON_DATA:bonusPointsEarnedReason::string as bonusPointsEarnedReason
   , JSON_DATA:dateScanned as dateScanned
   , JSON_DATA:finishedDate as finishedDate
   , JSON_DATA:modifyDate as modifyDate
   , JSON_DATA:pointsAwardedDate as pointsAwardedDate
   , JSON_DATA:pointsEarned::float as pointsEarned
   , JSON_DATA:purchaseDate as purchaseDate
   , JSON_DATA:purchasedItemCount::int as purchasedItemCount
   , JSON_DATA:rewardsReceiptItemList as rewardsReceiptItemList
   , JSON_DATA:rewardsReceiptStatus::string as rewardsReceiptStatus
   , JSON_DATA:totalSpent::float as totalSpent
   , JSON_DATA:userId::string as userId
from json_receipts
,LATERAL FLATTEN (input => JSON_DATA:_id) as flatten_id;
```

Second: Write a query that directly answers a predetermined question from a business stakeholder

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

- Last scanned data was on 2021-02-26

```
with receipts_scanned_data as (
   select
      _id as receipt_id
      , (flatten_itemlist.value):barcode::string as item_barcode
      , dateScanned
   from receipts
   , LATERAL FLATTEN (input => parse_json(rewardsReceiptItemList)) as flatten_itemlist
   where item_barcode is not null
)
, brands_data as (
   select
      item_barcode
      , month(dateScanned) || '-' || year(dateScanned) as scanMonth
      , name
      , count(*) as totalScans
   from receipts_scanned_data
   left join brands
```

```
   on receipts_scanned_data.item_barcode = brands.barcode
   where dateScanned >= add_months('2021-02-01'::date, -1)
   group by scanMonth, name, item_barcode
)
, ranked_data as (
   select
      totalScans
      , name
      , item_barcode
      , scanMonth
      , dense_rank() over (partition by scanMonth order by totalScans desc) as brand_rank
   from brands_data
   order by scanMonth desc, 5
)
select * from ranked_data
where brand_rank <= 5;
```

Observation: The above query should retrieve the top 5 brands according to total scans for the current and past months. This answers the first two bullet points i.e., top 5 brands in the current month and their comparison with the previous month. Since the last available data is in the month Feb-2021, I used this date to retrieve the data for the last month. However, the data provided is not sufficient to make the comparisons because there are null values for most brand names when joined with brands table on barcodes.

Alternative (using barcodes instead of brand names): When we rank the data based on barcodes, we can clearly see that there are products that share the ranking with the same number of scans in a month. However, the product with barcode 4011 topped the list in both the months. One question to the product team/ stake holders here could be - What/ how they want to see the ranking. Can the products share rank or they just want one product in a place.

The data quality issue here is - All barcodes scanned by the users do not have the corresponding brands/ brand names associated with them. This can be a big gap to understand user behavior of purchasing the products.

Third: Research on Data Quality Issues

Since the _id column in the three tables is expected to be primary key, it should not contain any duplicates or null values. The tables receipts and brands pass this integrity check but the users table holds duplicates for the _id columns which are expected to be unique to a user. For example, the user id '5ff1e194b6a9d73a3a9f1052' has almost 11 records in the users table. This can be QA-ed using the following queries:

```
select count(_id) duplicate_count _id from users
group by _id;

select * from users where _id = '5ff1e194b6a9d73a3a9f1052'
```

Another quality issue I identified is:
The purchase date(purchaseDate) is expected to be before the receipt scan date(dateScanned). There are few entries in receipts table that violate this data check by having scanDate before purchaseDate. These scans were also awarded points for the receipt scan. This needs to be investigated and validated by the backend before we accept the receipts. These results can be viewed using the following query:

```
select * from receipts
where dateScanned < purchaseDate
```

Fourth: Communicate with Stakeholders

- What questions do you have about the data?

- How are the receipts data and the brands data related? Is there a field in the receipts data schema that corresponds to the '_id' field in the brands data schema?
- What does the 'rewardsReceiptStatus' field in the receipts data schema represent? Are there specific values that it can take and what do they indicate?
- How is the 'bonusPointsEarned' field in the receipts data schema related to the 'pointsEarned' field? Are they calculated in the same way or are they different types of points?
- What is the meaning of "status of the receipt through receipt validation and processing" in the receipts data schema? How do this status impact the users scanning the receipts?
- How is the 'active' field in the users data schema used? Are inactive users removed from the system after certain period or just flagged?
- Are 'topBrand' and 'categoryCode' fields in the Brand Data Schema related, and if yes, how ?
- Can you give more details about 'cpg' reference in the Brand Data Schema, how is it related to the brand and how is it used ?
- Are there any missing fields or fields that are not present in the schema that would be useful to have in order to analyze the data?
- Are there any constraints or rules that govern the input of data into these tables?
- Is there any temporal dependency between the dates fields in the different tables, such as purchase date in receipts data should always come before points awarded date.
- How is the data generated, is it by application, by admin, or through some external source, and in the case of external source, can you explain the source?
- How often is the data updated, is it real-time, daily, weekly, monthly?
- Are there any data quality issues to be aware of in the data, such as missing or null fields, or duplicate records?
- Are there any other tables related to the three tables you provided, and if so, how do they relate?
- Are there any known gaps in data?

- How did you discover the data quality issues?

I have discovered a data quality issue due to which you might be seeing some increased values on the total number of users scanning the receipts. This is because of the duplicate data in one of our users table, I will filter them out and send you back the results as soon as possible.

Also, there are few receipt scans happening before the purchase of the products and were also awarded points for the scan. I suspect this to be a failure of validating the temporal dependency check on the backend while accepting the receipt scans. I will get in touch with the engineering team or who so ever is responsible to fix these bugs. For time being, I will eliminate these records from the data results I hand over to you.

- What do you need to know to resolve the data quality issues?
The duplicate values in users table can be easily filtered out. The other data quality issue could be missing brand names/ barcodes in the brands table. There are huge number of scans for the unknown brand names in the data provided:

I worked on the data request to get the top 5 brands of the current month and their comparison to the previous month based on total scans. But there is an issue to get the top 5 brands with the maximum scans in a month because most of these scans are from unknown brand names. However, I pulled some data with the reference on the barcodes, and it is as following: (Put the results here)

Let me know if you need anything else!

- What other information would you need to help you optimize data asset you're trying to create?
One thing I could think of is the rewardsReceiptItemList column in the receipts table. Currently, it is in the form list of json data, but we could think of flattening it into a sperate table rather than doing a lateral flatten every time we want to use that data. This totally depends on the usage of the data(items list in scanned receipt) for analytics keeping in mind the tradeoff between creating a new table vs flattening the column every time we need it.

- What performance and scaling concerns do you anticipate in production and how do you plan to address them?
The scans data in the receipts can increase exponentially with increasing users who use the Fetch Rewards app. There could be possible bottleneck when we have multiple views or tables with complex logic/ aggregations are built on this table. One way to scale the performance here is to increase the warehouse size. But increasing the warehouse/ cluster size is not optimal always, so one way to tackle this is to add indexes on top the views/tables. Since Snowflake doesn't support indices, clustering keys can be used to improve the performance.

The other option could be to understand how the analytics are using the data – analytics rarely use the whole raw events data most of the times, so this can be filtered to last 30-90 days. Also, we can have two different clusters, one for reading the data by dashboards (smaller warehouse) and the other for performing complex aggregations on the total events data(larger warehouse) with minimal views/tables for just that piece of data.