**Take Home Assignment Documentation**

* **Data Model**
  + 4 different tables, i.e. RECEIPTS, RECEIPT\_ITEMS, USERS, and BRANDS can be created in a data warehouse.
  + ID is the primary key in RECEIPTS table which is the foreign key in RECEIPT\_ITEMS table and is named as REWARDS\_RECEIPT\_ID
  + REWARDS\_RECEIPT\_ITEM\_ID is the primary key in RECEIPT\_ITEMS table
  + ID is the primary key in USERS table which is the foreign key in RECEIPTS table and is named as USER\_ID
  + ID is the primary key in the BRANDS table. BRANDS and RECEIPT\_ITEMS table can be joined on the basis of BRAND\_CODE. BRAND\_CODE is not the primary key or the foreign key as this column contains null values.
  + 1 RECEIPT can have n items in it, therefore, RECEIPTS has a 1:n relationship with RECEIPT\_ITEMS
  + 1 User can have n receipts, therefore, USERS has a 1:n relationship with RECEIPTS
  + A product of 1 Brand can be present in multiple receipt items, therefore, BRANDS has a 1:n relationship with RECEIPT\_ITEMS
* **SQL Queries:** I have ingested the data and wrote queries on PostgreSQL
  + **Question 1 - Which brand saw the most dollars spent in the month of June?**
    - While identifying the brand which spent the most dollars in June, the year was not mentioned, hence the query will return the results for the month of June in all the possible years in the data
    - Receipt\_Items table was used to compute the total spent at brand level using the total\_final\_price column and purchase\_date column from the **RECEIPTS** table was used to calculate the year and month
  + **Question 2 - Which user spent the most money in the month of August?**
    - While identifying the user which spent the most money in August, the year was not mentioned, hence the query will return the results for the month of June in all the possible years in the data
    - Purchase\_Date and Total\_Spent column from the **RECEIPTS** table was used calculate the most money spent in the month of August
  + **Question 3 - What user bought the most expensive item?**
    - The most expensive item was identified using the **RECEIPT\_ITEMS** table and based on the receipt Id we could join with the **RECEIPTS** table and identify the user\_id of the user who bought the most expensive item
  + **Question 4 - What is the name of the most expensive item purchased?**
    - The name of the most expensive item can be identified using the **RECEIPT\_ITEMS** table using the description column
  + **Question 5 - How many users scanned in each month?**
    - **RECEIPTS** table can be used to identify the number of users scanned in each month with the help of the columns date\_scanned and user\_id
* **Business Insights:** I have uploaded link to a google sheet and created the dashboard to show key business insights. Also, the SQL queries to derive the data for the dashboard and some data quality issues have been mentioned in the ‘Data Quality & SQL Queries’ sheet. Link: <https://docs.google.com/spreadsheets/d/1QfUlTLaXj7phh9LIU2A2Uroz7bHRPNpE35279klW6pc/edit#gid=1434986193>
  + Gender Distribution:
    - Created a pie chart that shows ~30% of the users don’t prefer to reveal their gender, females and transgenders occupy a share of 25% each and males being the lowest at ~21%
  + Sign Up Source:
    - 1/3rd of the total users prefer to sign-up using their apple-id, the other 2/3rd of the users are almost equally split into google, facebook, and login via email
  + Money Spent Over Time:
    - The money spent by the users is increasing over time which is a good sign for Fetch as it will encourage the users to consistently log their receipts on the application to earn rewards.
    - There are two outliers both in the month of Aug for the year 2021 and 2022 and I suspect a sale which might have motivated the users to purchase things in bulk
  + Money Spent By Brands(Top 10):
    - Starbucks, Kirkland Signature, Great Value, and more are some of the brands which the users are spending most of the money, therefore, Fetch can display these brands on the home page and even run special offers on these