PART 1:

There are 3 data sets given to us:

1. Receipts
2. Users
3. Brands.

I have made a couple of assumptions and taken my best judgement over this in terms of data exploration, cleaning and performing analytics for the business stakeholders.

To begin with, before building the foundation of data models and performing analysis on data, I always start with answering the business goal and the use case of it. Keeping in mind the long term developments and scalability, I personally believe having data in a highly normalized form or at least in terms of having them normalized at staging level. We can later create transformation models addressing specific business teams and needs.

Before I deep dive into the snow-flake model, let me explain and show my understanding of the data through ER Diagram.

Assuming the following:

User Table:

1. user_id is always unique and not null in the users table. (Currently that's not the case – I do get duplicate rows with various information which is present at times and at times information for some columns is missing for the same user).

Receipt Table:

1. receipt_id (PK) not null
2. user_id (FK) not null

Brands Table:

1. barcode on the item should always be present and should be unique for each item. I have assumed the barcode is always string.
   A. Items can have different sku / variation (there could still be unique barcodes for them).
      - If we do not intend to have unique barcodes for the sku or variations of item, then we ought to have the **brand_id** as a **foreign key** in the **rewards_receipt_item** table (explained below)

Cpg Table: This table is separated out from Brand Table that has the column 'cpg' as json
1. cpg_id (PK) not null
2. brand_id (FK) not null
3. Ref_type (either 'cpgs' or 'cogs')

Rewards Receipt Item Table:

This is a critical table, as it contains information related to the receipt as well as the brand and acts as a joining connection between them. Again, this is extracted and separated out as a table from the json object in the receipt table. This is to ensure more data transparency, and efficiency to perform analytical queries.

1. receipt_item_id (PK) not null
2. receipt_id (FK) not null
3. brand_id (FK) not null (assumed it should be included for better clarity on data and correctness, when the need arises to handle different skus / variants of an item)
4. barcode (can be used to join the brand table again) to get more insights

ER Diagram:

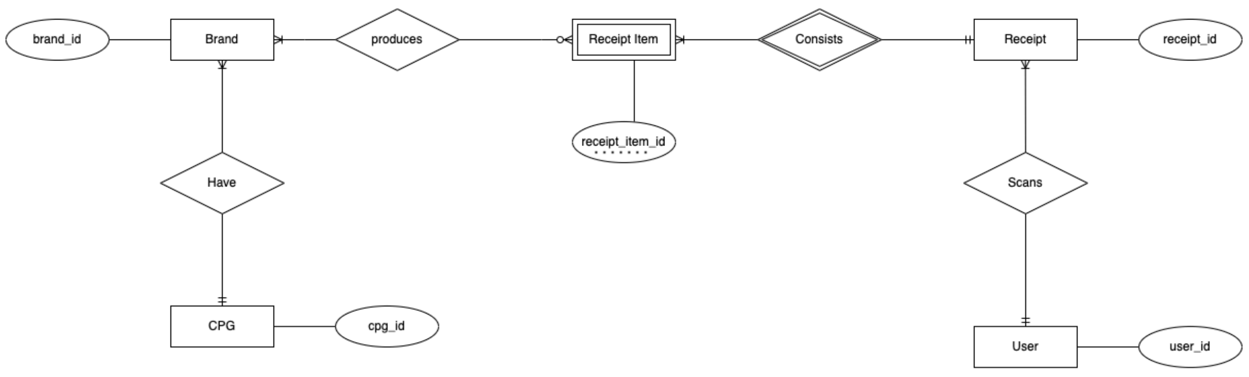
Entity - Relations and Assumptions:

1. Every receipt must be scanned by a user. Either one or multiple receipts can be scanned by a user. (Since it has been given that user_id in the receipt table ties back to the user collection, hence one is the minimum cardinality

2. A receipt can consist of one or many item lines. An item or many items however only belong to one receipt at a time.
Note: Receipt item is considered weak, because it does not exist without a receipt. If receipt is deleted, we also lose out on receipt_item info. Hence it's a weak entity and has an existential dependency.

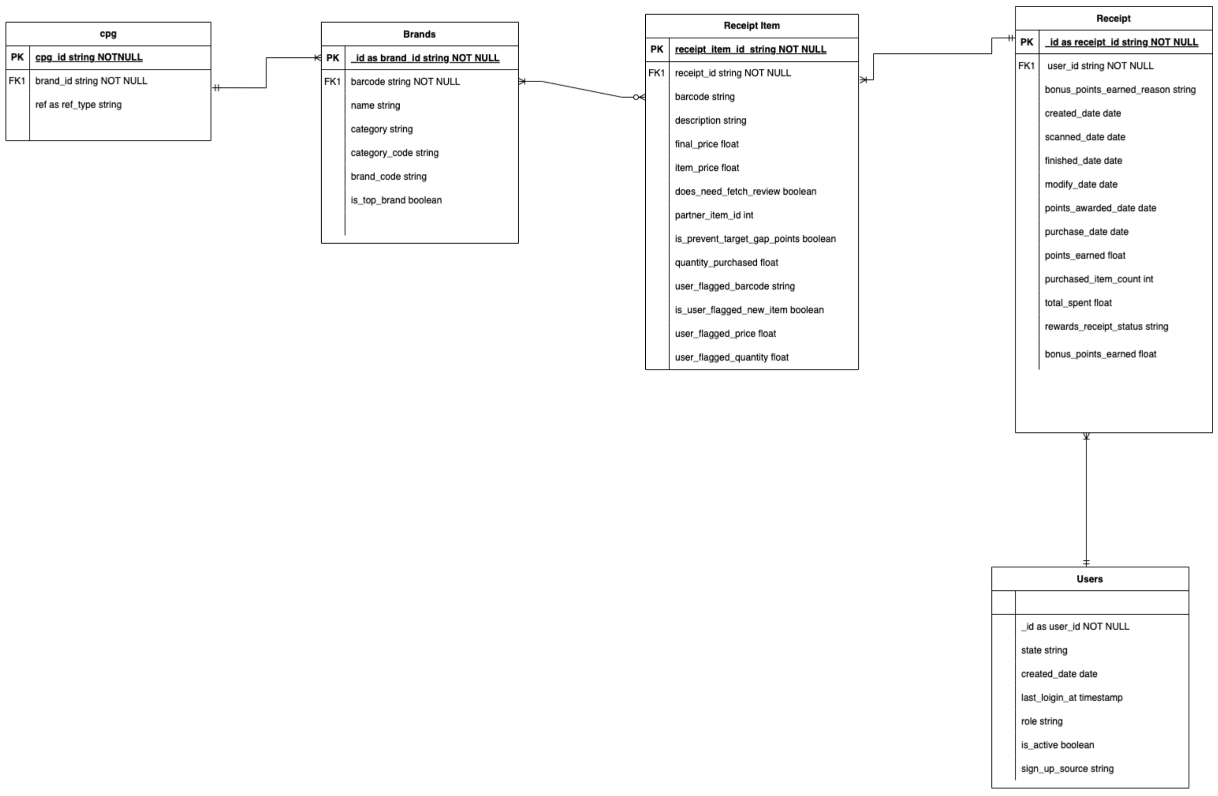
3. A brand can produce / sell no items or multiple items. However an item or multiple items need to be produced by a brand.

4. A consumer packaged goods (cpg) table can have one or multiple brands associated with it. A brand can have exactly 1 consumer packaged product associated. This piece is not very relevant for us to answer the business needs. However, just adding it to express my understanding.



<u>Note:</u> The above ER only shows PK for each entity. There are many (refer to the relational model) If they are not present in our data-set, I would create a surrogate key or have an auto-increment unique key for each entity to keep data unique and normalized.

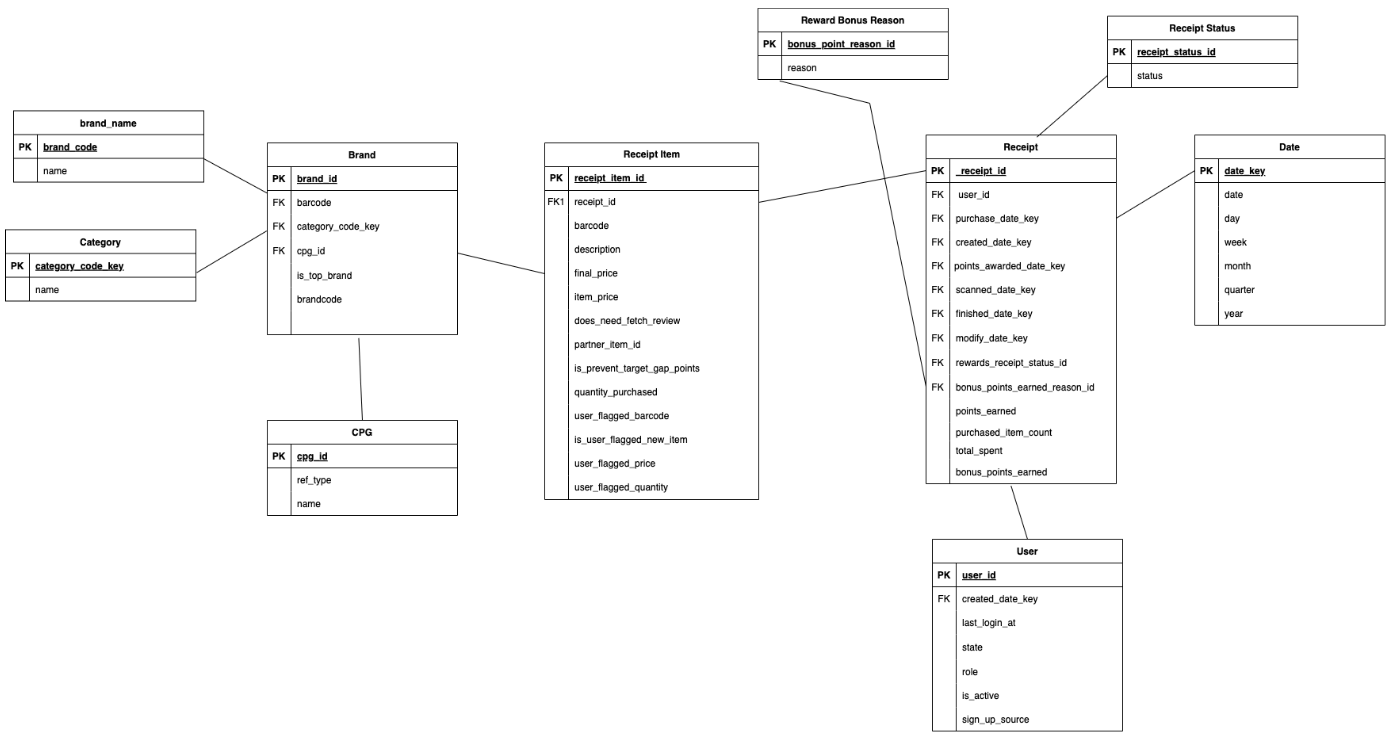
<u>Relational Model:</u>



Majority but all important attributes are added here in the model. Note, Dates can be pulled out as a separate dimension and we can have date_id, however to keep it concise, I am not expanding.

I have assumed there exists or should be a primary key within each dataset. (E.g Receipt Item, Cpg, do not have their own pk, but we could have a surrogate key for them)

Snowflake Schema:



Please note: The above is a pseudo model design. They can further be reduced to smaller dimensions. But the overall idea is to get facts and dimensions separated to get highly normalized design, which will be very useful for write operations. We can further break down receipt_item tables and other tables (users) into facts and dimensions.

PART 2:

Queries / Answers to Business Questions:

I will be using postgreSQL to answer these questions. To keep queries concise and avoid a lot of joins, I will use the design (referring to relational model not snowflake) to get the query and keep it simple.

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

```
with

most_recent_month_year as (
        select
                month(max(receipt.scanned_date)) as most_recent_month,
                year(max(receipt.scanned_date)) as most_recent_year
        from receipt
),

most_recent_receipts as (
        select
                receipt.receipt_id,
        from receipt
        where
                month(receipt.scanned_date) in (
                        select most_recent_month
                        from most_recent_month_year
                )
                and
                year(receipt.scanned_date) in (
                        select most_recent_year
                        from most_recent_month_year
                )
),

 most_recent_month_receipts_stats as (
        select
                brand.name as brand_name,
                sum(receipt.total_spent) as total_sales
        from receipt
        inner join receipt_item
        on receipt.receipt_id = receipt_item.receipt_id
        left join brand //this can be debated if all brands have barcodes or not. I assume they don’t
        on receipt_item.bar_code = brand.barcode and receipt_item.brand_id = brand.brand_id
        where brand.is_top_brand and receipt.receipt_id in (select receipt_id from
        most_recent_receipts)
        group by brand.brand_name
        order by total_sales DESC
),
overall_top_brand_rank as (
	select
		most_recent_month_receipts_stats.brand_name,
		most_recent_month_receipts_stats.total_sales,
		dense_rank() over (order by most_recent_month_receipts_stats.total_sales desc) as
top_brand_rank
	from most_recent_month_receipts_stats
)

select brand_name
from overall_top_brand_rank
where top_brand_rank <=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?

```
with

most_recent_month_year as (
	select
		month(max(receipt.scanned_date)) as most_recent_month,
		year(max(receipt.scanned_date)) as most_recent_year
	from receipt
),

previous_month_year as (
	select
		case
			when most_recent_month = 1 then 12
			else most_recent_month - 1
		end as previous_month,
		case
			when most_recent_month = 1 then most_recent_year - 1
			else most_recent_year
		end as most_recent_year
	from most_recent_month_year
),

most_recent_previous_receipts as (
	select
		receipt_id,
	from receipt
	where
		month(receipt.scanned_date) in (
			select previous_month
			from previous_month_year
		)
```

```
                    and
                    year(scanned_date) in (
                              select most_recent_year
                              from previous_month_year
                    )
),

 most_recent_previous_month_receipts_stats as (
          select
                    brand.name as brand_name,
                    sum(receipt.total_spent) as total_sales_previous_month
          from receipt
          inner join receipt_item
          on receipt.receipt_id = receipt_item.receipt_id
          left join brand //this can be debated if all brands have barcodes or not. I assume they don't
          on receipt_item.bar_code = brand.barcode and receipt_item.brand_id = brand.brand_id
          where brand.is_top_brand and receipt.receipt_id in (select receipt_id from
          most_recent_previous_receipts)
          group by brand.brand_name
          order by total_Sales DESC
),

overall_top_brand_rank_previous_month as (

          select
                    most_recent_previous_month_receipts_stats.brand_name,
                    most_recent_previous_month_receipts_stats.total_sales,
                    dense_rank() over (order by most_recent_previous_month_receipts_stats.total_sales
desc) as top_brand_rank_previous_month
          from most_recent_previous_month_receipts_stats
)

select
          overall_top_brand_rank_previous_month.brand_name
from overall_top_brand_rank_previous_month
where overall_top_brand_rank_previous_month.top_brand_rank_previous_month <=5;
```

Combining query from Q1 to get top 5 brands for recent month:
We can compare → using brand_name and their difference between total_sales for recent and previous to recent month. I will order by highest positive change to negative change.

```
top_brand_recent_month_sales as (
select
          brand_name,
          total_sales,
from overall_top_brand_rank
where top_brand_rank <=5
),
top_brand_previous_month_sales as (
	select
		brand_name,
		total_sales_previous_month
	from overall_top_brand_rank_previous_month
),

recent_top_brand_compare_to_previous as (
	select
		top_brand_recent_month_sales.brand_name,
		coalesce((top_brand_recent_month_sales.total_sales -
top_brand_previous_month_sales.total_sales_previous_month),0) as change_in_sales,
	from top_brand_recent_month_sales
	left join top_brand_previous_month_sales
	on top_brand_recent_month_sales.brand_name =
top_brand_previous_month_sales.brand_name
)

select
	brand_name
	change_in_sales
from recent_top_brand_compare_to_previous
order by change_in_sales desc;
```

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

```
with

avg_spend_accepted as (
        select
                avg(receipts.total_spent) as avg_spent
        from receipts
        where receipts.rewards_receipt_status in (‘Accepted’)
),

avg_spend_rejected as (
        select
                avg(receipts.total_spent) as avg_spent
        from receipts
        where receipts.rewards_receipt_status in (‘Rejected’)
),

overall_result_union as (
        (select
                avg_spent,
                “Accepted” as recepit_status
        from avg_spent_accepted)
        union all
        (select
                avg_spent,
                “Rejected” as receipt_status
        from avg_spent_rejected)
        order by avg_spent DESC
),

overall_rank as (
        select
                receipt_status,
                avg_spent,
                dense_rank() over (order by avg_spent desc) as top_rank
        from overall_result_union
)

select
        receipt_status
from overall_rank
where top_rank = 1;
```

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

```
with

total_items_accepted as (
        select
                sum(receipts.purchased_item_count) as total_items_purchased
        from receipts
        where receipts.rewards_receipt_status in (‘Accepted’)
),

total_items_rejected as (
        select
                sum(receipts.purchased_item_count) as total_items_purchased
        from receipts
        where receipts.rewards_receipt_status in (‘Rejected’)
),

overall_result_union as (
        (select
                total_items_accepted.total_items_purchased,
                “Accepted” as recepit_status
        from total_items_accepted)
        union all
        (select
                total_items_rejected.total_items_purchased,
                “Rejected” as receipt_status
        from total_items_rejected)
        order by total_items_purchased desc
),

overall_rank as (
        select
                receipt_status,
                total_items_purchased,
                dense_rank() over (order by total_items_purchased desc) as top_rank
        from overall_result_union
)

select
        receipt_status
from overall_rank
where top_rank = 1;
```

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

```
with

users_created_past_six_months as (
        select
                users.user.id
        from users
        where users.created_date >= (CURRENT_DATE - INTERVAL ‘6 months’)
),


brand_with_most_spend as (
        select
                brand.name as brand_name,
                sum(receipt.total_spent) as total_sales
        from receipt
        inner join receipt_item
        on receipt.receipt_id = receipt_item.receipt_id
        left join brand //this can be debated if all brands have barcodes or not. I assume they don’t
        on receipt_item.bar_code = brand.barcode and receipt_item.brand_id = brand.brand_id
        where receipt.user_id in (
                select users.user_id from users_created_past_six_months)
        group by brand.brand_name
),

overall_user_brand_spend as (
        select
                brand_name,
                total_sales,
                dense_rank() over (order by total_sales DESC) as top_spent_rank
        from brand_with_most_spend
)

select
        brand_name
from overall_user_brand_spend
where top_spent_rank = 1;
```

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

```
with

users_created_past_six_months as (
	select
		users.user.id
	from users
	where users.created_date >= (CURRENT_DATE - INTERVAL ‘6 months’)
),

brand_with_most_transactions as (
	select
		brand.name as brand_name,
		count(distinct receipt.receipt_id) as total_transactions
	from receipt
	inner join receipt_item
	on receipt.receipt_id = receipt_item.receipt_id
	left join brand //this can be debated if all brands have barcodes or not. I assume they don’t
	on receipt_item.bar_code = brand.barcode and receipt_item.brand_id = brand.brand_id
	where receipt.user_id in (
		select user_id from users_created_past_six_months)
	group by brand_name
),

overall_user_brand_transactions as (
	select
		brand_name,
		total_transactions,
		dense_rank() over (order by total_transactions DESC) as top_transactions_rank
	from brand_with_most_transactions
)

select
	brand_name
from overall_user_brand_transactions
where top_transactions_rank = 1;
```

PART 3:

I have used jupyter notebook and python3 to address and find data quality issues. You will find the .ipynb file in the repo alongside with the pdf of the code and the output too!

Part 4:

Hello Team,

I write this message to you as we strive to keep our data clean and healthy across all our systems. We have identified a few concerns with our current data that is collected from the receipts, data based on users as well as brands we have partnered with. These questions and issues arise after I reviewed the data through exploratory analysis and understood the specific answers that were needed.

Data Quality Concerns:

1. In our receipt items, we have discovered missing barcodes. I believe this could be due to unreadable barcodes (stickers or print issues) or if there are any certain products that don't have barcodes? This is an important feature for us, as we would like to associate every product that is being sold by a brand to be linked with our receipts through the barcode. This will give us more accurate information and allow us to extract actionable insights.

2. I have also noticed, there are a lot of duplicate entries of users created in our system that have the same information and details. Is there a way, we could set up some authentication on our user-input, so that we can prevent duplicates in our system

3. Brands are also tied up with Consumer Packaged Goods – Are there any cases where a brand might not have a CPG or is tied up with multiple CPGs. This again helps us give accurate metrics at the CPG level (of-course when the need arises).

Data Optimization / Query Performance:

1. I would also recommend as we aim for scalability and growth in the near future, we should start managing our storage of data to avoid latency and performance issues for our backend queries. This piece is essential and we can start off with partitioning historical data and index metrics that have active usage based on our most frequent reports that are needed by the business and product team. So, maybe having definitions of certain reports and their needs, we can manage and store data effectively.

2. As far as data freshness and availability goes, I would personally like to understand which teams use what kind of reports and how often. For example: Are we okay to review reports that do not have real-time data. Reports that have data which is synced every 12 hours, or even maybe a report that is viewed only once in a month? This would help us to design our data pipeline and transformation process to manage our costs.

Last but not the least, in order to address more data quality issues and optimize our data set, do we have or can the business team provide more information / documentation about different business requirements? I think the above answers and solutions will certainly help the data and analytics to cater needs of various teams