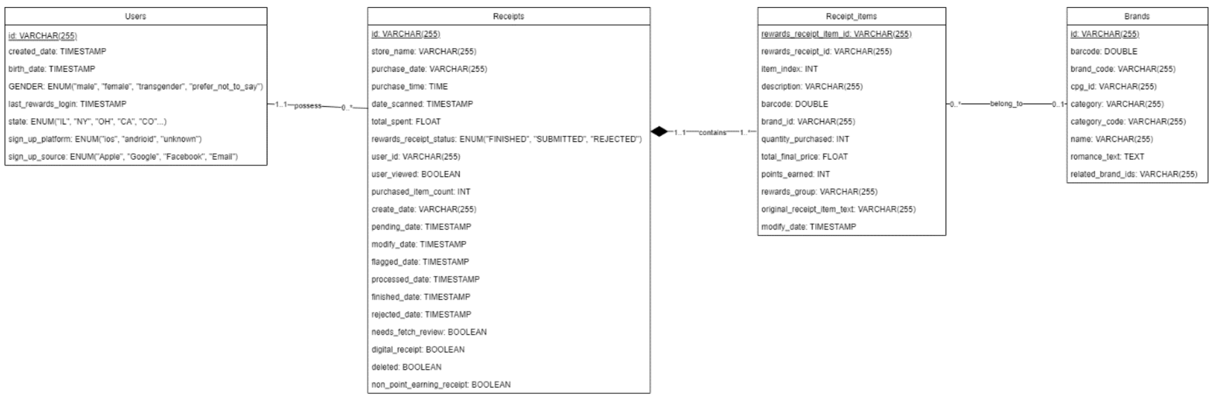
Applicant: Zhaokuan Chen zc56@illinois.edu

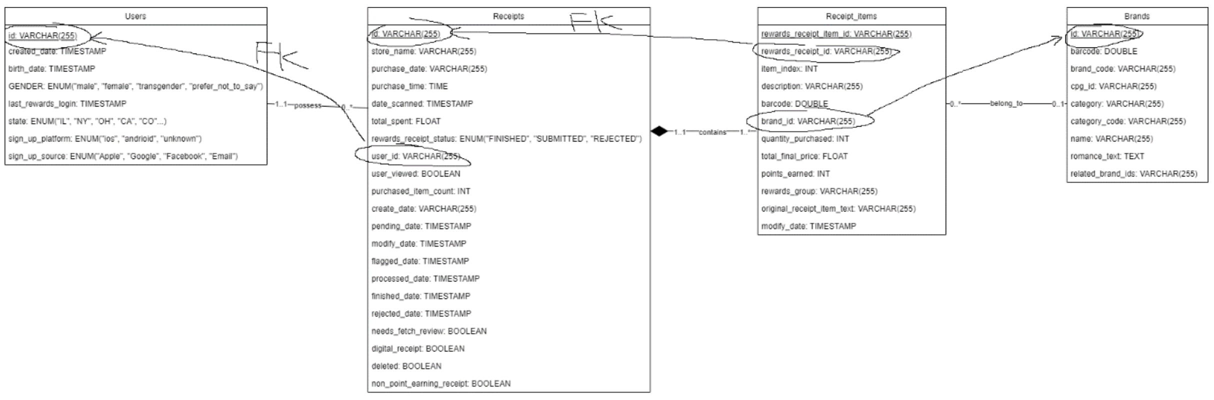
First: Review Existing Data and Diagram a New Structured Relational Data Model

For better view of the diagram, I included both pictures in the repository.

UML Diagram:



UML Diagram after marking foreign keys:



Database: MySQL

Why I draw a UML diagram like this?

In the UML diagram that I draw, the relation between two entities is merely there to show that there is a relation between the two entities. The relation table does not contain any attribute because the relation table has already merged with one table in this relationship.

To be more specific, in the design phase, the entity table "Receipts" does not have attribute "user_id" and the relation table "possess" may look like this:

```
CREATE TABLE possess(
    user_id VARCHAR(255),
    receipt_id VARCHAR(255),
    PRIMARY KEY (receipts_id),
    FOREIGN KEY (user_id) REFERENCES User(id),
    FOREIGN KEY (receipt_id) REFERENCES Receipts(id)
);
```

Because it is a one-to-many relationship between "Users" and "Receipts", the "possess" table merged with "Receipts" table. The merge add "user_id" into "Receipts" and form the "Receipts" table in the UML.

The same idea apply to "contains" and "belong_to".

In the design phase, the “Receipt_items” table does not contain the attribute “rewards_receipt_id” and the “contains” table may look like this:

```
CREATE TABLE contains(
	receipt_id VARCHAR(255),
	receipt_item_id VARCHAR(255),
	PRIMARY KEY (receipt_item_id),
	FOREIGN KEY (receipt_id) REFERENCES Receipts(id),
	FOREIGN KEY (receipt_item_id) REFERENCES Receipt_items(rewards_receipt_item_id),
);
```

Because it is a one-to-many relationship between “Receipts” and “Receipt_items”, the “contains” table merged with the “Receipt_items”. The merge add “rewards_receipt_id” into table “Receipt_items” and form the “Receipt_items” in the UML diagram.

I removed the attribute “barcode” from “Receipt_items” because “barcode” can be empty in “Brands” and hence “barcode” is not a primary key. Putting a non primary key in “Receipt_items” does not really help the connection between two table. I replaced “barcode” with “brand_id” which is a foreign key referencing to the attribute “Brand.id”

In the design phase, the “Receipt_items” table does not contain the attribute “brand_id” and the “belong_to” table may look like this:

```
CREATE TABLE belong_to (
	receipt_item_id VARCHAR(255),
	brand_id VARCHAR(255),
	PRIMARY KEY (receipt_item_id),
	FOREIGN KEY (receipt_item_id) REFERENCES Receipt_items(rewards_receipt_item_id),
	FOREIGN KEY (brand_id) REFERENCES Brands(id)
);
```

Because it is a many-to-one relationship between “Receipt_items” and “brands”, the “belong_to” table merged with the “Receipt_items”. The merge add “brand_id” into “Receipt_items” and form the “Receipt_items” in the UML diagram.

In conclusion, the four CSV file are four tables “Users”, “Receipts”, “Receipt_items”, “Brands”. Table “Users” and table “Receipts” are connected by the “user_id” in table “Receipts”. Table “Receipts” and table “Receipt_items” are connected by the “rewards_receipt_id” in table “Receipt_items”. Table “Receipt_items” and table “Brands” are connected by the “brand_code” in table “Receipt_item”. Additionally, I draw a composition relationship between “Receipts” and “Receipt_items” because “Receipt_items” cannot exist without “Receipts”.

Second: Write a query that directly answers questions from a business stakeholder.

Database: MySQL

1: Which brand saw the most dollors spent in the month of June

```
SELECT b.id AS Brand_ID, SUM(ri.total_final_price) AS Total_Gain
FROM Receipt_items AS ri JOIN Brands AS b ON (ri.brand_id = b.id)
WHERE month(modify_date) = 6
GROUP BY b.id
ORDER BY Total_Gain DESC
LIMIT 1;
```

2: Which user spent the most money in the month of August

```
SELECT u.id AS User_ID, SUM(r.total_spent) AS Total_Spent
FROM Users AS u JOIN Receipts AS r ON (u.id = r.user_id)
WHERE month(create_date)= 8
GROUP BY u.id
ORDER BY Total_Spent DESC
LIMIT 1;
```

3: What user bought the most expensive item

```
SELECT u.id
FROM Users AS u JOIN Receipts AS r ON (u.id = r.user_id) JOIN
(SELECT ri.rewards_receipt_id AS rewards_receipt_id, MAX(ri.total_final_price) AS max_price
FROM Receipt_items AS ri) AS temp ON (r.id = temp.rewards_receipt_id)
```

4: What is the name of the most expensive item purchased

```
SELECT original_receipt_item_text AS Item_Name, MAX(ri.total_final_price / ri.quantity_purchased) AS
Max_Price
FROM Receipt_items
```

5: How many users scanned in each month

```
SELECT month(date_scanned) AS Month, COUNT(DISTINCT user_id) AS User_Counts
FROM Receipts
GROUP BY month(date_scanned)
```

Third: Choose something noteworthy about the data and share with a non-technical stakeholder

The data quality is not that good. There are way too many empty entries in the table. And for some attributes, such as “barcode”, sometime it is using scientific notation and sometime it is not.
Additionally, in the original table “Receipt_items”, there is an attribute “brand_code”. I believe it is intended to serve as a connection between “Receipt_items” and “Brands”, maybe the designer want “brand_code” to be a foreign key in “Receipt_items” referring to “Brands.brand_code”. However, “brand_code” is not a key in “Brands” because “brand_code” can be empty (inferred by the fact that lots of entries under “brand_code” attribute are empty). Because “brand_code” is not a key, it cannot be a foreign key in “Receipt_item”. Therefore, the connection between “Receipt_items” and “Brands” is not correctly captured. Thus, I replaced “brand_code” with “brand_id” because “brand_id” is the primary key in “Brands”.