Part Three: Evaluate Data Quality Issues in the Data Provided

Language Used: Transact-SQL

USING Table 1: Receipts Table

1. Checking for Negative Values

```
SELECT BonusPointsEarned, PointsEarned, PurchasedItemCount, TotalSpent

WHERE (BonusPointsEarned < 0)

OR (PointsEarned < 0)

OR (PurchasedItemCount < 0)

OR (TotalSpent < 0);
```

2. Checking for NULL values in whatever column it is not allowed to have a NULL value

- for columns like 'BonusPointsEarned' or 'BonusPointsEarnedReason', that indicate added points that not all users may receive, we would allow/expect NULL values here. But for columns like 'ReceiptID', or DateTimes tracking timelines of the receipts being created to when they were uploaded onto Fetch Rewards, the unique ID given to each receipt and a Primary Key for our table, we cannot allow NULL values here.

```
SELECT * FROM RECEIPTS

WHERE (ReceiptID is NULL)

OR (CreateDate IS NULL)

OR (DateScanned IS NULL)

OR (FinishedDate IS NULL)

OR (PurchasedItemCount IS NULL)

OR (TotalSpend IS NULL)

OR (UserID IS NULL);
```

3. **Checking for issues in logic with recorded dates. For example, DateScanned should always be greater than or equal to CreateDate or PurchaseDate**

```
SELECT * FROM Receipts

WHERE (DateScanned < PurchaseDate)

OR (DateScanned < CreateDate)

OR (ModifyDate < CreateDate)

OR (FinishedDate < CreateDate);
```

4. **Making sure that the number of UNIQUE rows in Receipt_Item table = number of rows in Receipt table. You cannot have any Receipt Items that are not associated with a listed Receipt. Run the following queries, the output number should be the same, if not, there is a discrepancy between the data load of Receipt data and Receipt Item data.**

```
SELECT (COUNT(Receipts.ReceiptID));

SELECT (COUNT(DISTINCT(Receipts.ReceiptID)));
```