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

Task: To identify the top 5 brands based on the number of receipts scanned during the most recent month recorded in the receipts_main table.

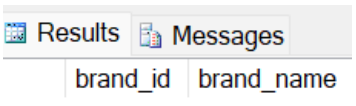
```
WITH recent AS (
   SELECT TOP 1
      YEAR(r.scanned_date) AS year, MONTH(r.scanned_date) AS month
   FROM receipts_main r
   ORDER BY r.scanned_date DESC
),
res AS (
   SELECT TOP 5
      i.barcode
   FROM receipts_main r
   JOIN receipt_items i ON r.receipt_id = i.receipt_id
   WHERE YEAR(r.scanned_date) = (SELECT year FROM recent)
     AND MONTH(r.scanned_date) = (SELECT month FROM recent)
   GROUP BY i.barcode
   ORDER BY COUNT(i.barcode) DESC
)
SELECT b.brand_id, b.brand_name FROM brands_data b
  JOIN res ON b.barcode = res.barcode;
```

Explanation:

1. **CTE - recent**:
- This CTE retrieves the most recent year and month based on the scanned_date from the receipts_main table.
- It sorts the records by scanned_date in descending order and selects the top entry, ensuring that we capture the latest month for analysis.
2. **CTE - res**:
- This CTE finds the top 5 barcodes from the receipt_items table, which are associated with the receipts scanned in the most recent month.
- It joins receipts_main with receipt_items on receipt_id, filters the records by the year and month obtained from the first CTE, groups the results by barcode, and orders them by the count of receipts scanned in descending order.
3. **Final Selection**:
- The main query retrieves brand_id and brand_name from the brands_data table by joining it with the results from the res CTE based on the barcode.

Results:

- Upon executing the query, the most recent data indicates that the most recent year is **2021** and the month is **March**
- The query identifies a list of the top 5 barcodes scanned during March 2021. However, it is important to note that the barcodes obtained from this analysis are not present in the brands_data table.
- As a result, the final output of the query is **empty**, indicating that there are no matching brands for the barcodes scanned in the most recent month.



Question 3: When considering *average spend* from receipts with 'rewardsReceiptStatus' of 'Accepted' or 'Rejected', which is greater?

Task: To evaluate the average spending of customers for receipts categorized by their rewardsReceiptStatus.
Notably, the provided dataset does not include any receipts with a status of 'Accepted.'
As a result, the query is written n order to extracts the total spent and average spent for the various statuses present in the dataset, which includes 'FLAGGED,' 'FINISHED,' 'PENDING,' 'REJECTED,' and 'SUBMITTED.'

```
SELECT
   rewards_receipt_status AS STATUS,
   SUM(total_spent) AS TotalSpent,
   (SUM(total_spent) / COUNT(rewards_receipt_status)) AS AvgSpent
FROM receipts_main
GROUP BY rewards_receipt_status
ORDER BY AvgSpent DESC;
```

Explanation:

1. **SELECT Statement**:
- rewards_receipt_status AS STATUS: Retrieves the status of the rewards receipt, aliased as STATUS for clarity.
- SUM(total_spent) AS TotalSpent: Calculates the total amount spent for each rewards receipt status, labeled as TotalSpent.
- (SUM(total_spent) / COUNT(rewards_receipt_status)) AS AvgSpent: Computes the average spending by dividing the total spent by the count of receipts for each status, labeled as AvgSpent.

2. **FROM Clause**:
- FROM receipts_main: Indicates that the data is pulled from the receipts_main table.

3. **GROUP BY Clause**:
- GROUP BY rewards_receipt_status: Groups the results by the rewards_receipt_status, enabling separate calculations for each status present in the data.

4. **ORDER BY Clause**:
- ORDER BY AvgSpent DESC: Orders the results in descending order based on average spending, allowing for easy identification of which status has a higher average spend.

Results:
The output of the query:

Results | Messages

	STATUS	TotalSpent	AvgSpent
1	FLAGGED	8300.78	180.451739
2	FINISHED	41882.53	80.854305
3	PENDING	1373.59	27.471800
4	REJECTED	1656.15	23.326056
5	SUBMITTED	NULL	NULL

- The analysis of the average spending across different rewards receipt statuses highlights that, among the available data, the 'FLAGGED' status shows the highest average spend per receipt.
- However, it is important to note that there is no data for the 'ACCEPTED' status, which limits the completeness of this analysis.

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

Task: To evaluate the total number of items purchased from receipts categorized by their rewardsReceiptStatus.
Notably, the provided dataset does not include any receipts with a status of 'Accepted.'
As a result, the query extracts the total number of purchased items for the various statuses present in the dataset.

```
SELECT
   rewards_receipt_status AS STATUS,
   SUM(purchased_item_count) AS ItemsCount
FROM receipts_main
GROUP BY rewards_receipt_status
ORDER BY ItemsCount DESC;
```

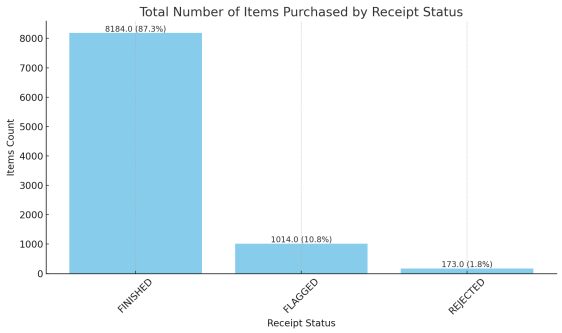
Explanation:

1. **SELECT Statement**:
- rewards_receipt_status AS STATUS: This retrieves the status of the rewards receipt, aliased as STATUS for clarity in the output.
- SUM(purchased_item_count) AS ItemsCount: This calculates the total number of items purchased for each rewards receipt status, labelled as ItemsCount.
2. **FROM Clause**:
- FROM receipts_main: This indicates that the data is being pulled from the receipts_main table.
3. **GROUP BY Clause**:
- GROUP BY rewards_receipt_status: This groups the results by the rewards_receipt_status, allowing for separate calculations for each status present in the data.
4. **ORDER BY Clause**:
- ORDER BY ItemsCount DESC: This orders the final results in descending order based on the total number of items purchased, allowing for easy identification of which status has a higher count.

Result:
The output of the query:

Results | Messages

	STATUS	ItemsCount
1	FINISHED	8184
2	FLAGGED	1014
3	REJECTED	173
4	PENDING	NULL
5	SUBMITTED	NULL



- From the analysis, it is evident that the 'FINISHED' status has the highest total number of items purchased, significantly surpassing other statuses.
- The 'FLAGGED' status also shows a notable count, while 'REJECTED' has a much lower total.

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

Task: to identify the brand that has the highest total spending from users who were created within the last six months. By analyzing the purchasing behavior of these new users, businesses can tailor their marketing strategies and product offerings to better engage this demographic.

It is important to note that instead of using the current date to fetch data regarding the past six months, we are basing our analysis on the most recent date available in the dataset. This means we are calculating the last six months from the latest recorded user creation date

```
WITH recent AS (
   SELECT user_id AS users FROM users_data
   WHERE created_date > DATEADD(MONTH, -6, (SELECT TOP 1 created_date FROM users_data
    ORDER BY created_date DESC))
),
barcode_spending AS (
   SELECT i.barcode, SUM(r.total_spent) AS TotalSpent FROM receipts_main r
   JOIN receipt_items i ON r.receipt_id = i.receipt_id
   WHERE r.user_id IN (SELECT users FROM recent)
   GROUP BY i.barcode
)
SELECT bs.barcode, bs.TotalSpent, b.brand_name FROM barcode_spending bs
LEFT JOIN brands_data b ON bs.barcode = b.barcode
ORDER BY bs.TotalSpent DESC;
```

Explanation

1. Common Table Expression (CTE) - recent:
- This CTE identifies users who have been created in the last six months by comparing their created_date to the most recent date recorded in the users_data table.
2. Common Table Expression (CTE) - barcode_spending:
- This CTE calculates the total amount spent by these recent users for each barcode associated with their purchases. It aggregates the spending from the receipts_main and receipt_items tables.
3. Final Selection:
- The final query selects the barcode, total spending, and the corresponding brand_name by performing a LEFT JOIN with the brands_data table. This ensures that all barcodes with total spending are displayed, even if there is no matching brand name.
4. Ordering:
- The results are ordered by TotalSpent in descending order, allowing for the quick identification of which brand has the highest spending.

Result:

- Upon executing the query, the output will provide a list of barcodes along with their total spending and associated brand names.
- This query effectively identifies the brand with the highest total spending among new users.

Results Messages

	barcode	TotalSpent	brand_name
1	3.11E+11	112930.69	NULL
2	5.11E+11	73579.49	NULL
3	7.55E+11	65021.98	NULL
4	76840580750	56036.89	NULL
5	36000391718	54967.20	NULL
6	21834033316	53838.20	NULL
7	12000809941	52192.11	NULL
8	21000658831	51941.45	NULL
9	41000022623	31101.10	NULL
10	34000056644	27272.25	NULL

383	21000068159	0.99	NULL
384	21000051885	0.99	NULL
385	43000066706	0.99	NULL
386	43000035818	0.99	NULL

	UniqueBarcodeCount
1	386

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

Task: To identify the brand that has the highest number of transactions from users who were created within the last six months. By analyzing the transaction behavior of these new users, businesses can gain insights into brand engagement and customer preferences, allowing them to optimize marketing strategies and product offerings.

It is important to note that instead of using the current date to fetch data regarding the past six months, we are basing our analysis on the most recent date available in the dataset.

```
WITH recent AS (
   SELECT user_id AS users FROM users_data
   WHERE created_date > DATEADD(MONTH, -6, (SELECT TOP 1 created_date FROM users_data
 ORDER BY created_date DESC))
)
SELECT i.barcode, COUNT(i.barcode) AS Transactions, b.brand_name FROM receipts_main r
JOIN receipt_items i ON r.receipt_id = i.receipt_id
LEFT JOIN brands_data b ON i.barcode = b.barcode
WHERE r.user_id IN (SELECT users FROM recent)
GROUP BY i.barcode, b.brand_name
ORDER BY Transactions DESC;
```

Explanation:

1. **Common Table Expression (CTE) - recent**:
- This CTE identifies users who have been created in the last six months by comparing their created_date to the most recent date recorded in the users_data table.
2. **Main Query**:
- The main query selects the barcode, counts the number of transactions for each barcode, and retrieves the corresponding brand_name.
- The query joins the receipts_main and receipt_items tables to aggregate the transaction data and uses a LEFT JOIN with the brands_data table to include brand names.
3. **WHERE Clause**:
- This filters the records to include only those transactions associated with users created in the last six months.
4. **GROUP BY Clause**:
- The results are grouped by both barcode and brand_name to ensure accurate counting of transactions for each unique barcode-brand combination.
5. **Ordering**:
- The results are ordered by the Transactions count in descending order, allowing for quick identification of which brand has the most transactions.

Results

Results | Messages

	barcode	Transactions
1	4011	115
2	7.55E+11	107
3	36000320893	92
4	36000391718	69
5	12000809941	63

382	21000648863	1
383	21000069293	1
384	21000070787	1
385	21908236865	1
386	22	1

- This query effectively identifies the brand with the highest number of transactions among new users.
- By focusing on users created within the last six months, businesses can gain insights into which brands are engaging new customers most effectively.