FETCH REWARDS TAKE-HOME EXAMINATION

First part:

1. Are there any data quality issues?

PRODUCT_TAKEHOME

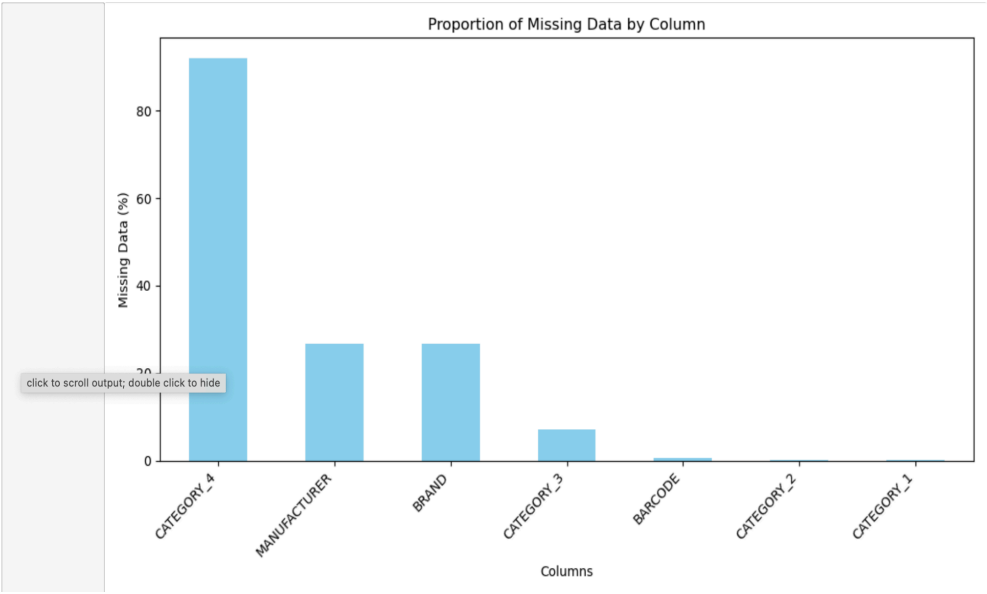
FIRST STEP: Checking for null/missing values

After a thorough analysis of the PRODUCT_TAKEHOME table, I infer the following:

- CATEGORY_1 has 111 null values
- CATEGORY_2 has 1424 null values
- CATEGORY_3 has 60566 null values
- CATEGORY_4 has 778093 null values
- MANUFACTURER has 86902 Placeholder values apart from 226474 null values
- BRAND has 226472 null values
- BRAND has 4025 null values and 4 duplicates including nulls.

```
In [9]: print(product_df.isnull().sum())
```

```
CATEGORY_1         111
CATEGORY_2        1424
CATEGORY_3       60566
CATEGORY_4      778093
MANUFACTURER    226474
BRAND           226472
BARCODE           4025
dtype: int64
```



SQLite:

```
SELECT barcode, COUNT(*) AS count
FROM PRODUCTS_TAKEHOME
GROUP BY barcode
HAVING COUNT(*) > 1;
```

	BARCODE	count
1	*NULL*	4025
2	017000329260	2
3	052336919068	2

CATEGORY_4 has the most null values, with 778093 accounting for 92% of missing values. This number is very high and accounts for severe data quality issues. Since this is hierarchical data, CATEGORY_4, assuming it is an optional field, must be very specific about the product implying that the data quality shouldn't be compromised because of this column.

```
In [52]: #GroupBy
         brand_counts = product_df.groupby('CATEGORY_1')['BRAND'].nunique()
         print(brand_counts)
```

```
CATEGORY_1
Alcohol                      4
Animals & Pet Supplies       1
Apparel & Accessories        7
Arts & Entertainment         3
Baby & Toddler               2
Beauty                       2
Beverages                    5
Dairy                        4
Deli & Bakery                3
Electronics                  3
Frozen                       5
Health & Wellness         5130
Home & Garden                8
Household Supplies           5
Luggage & Bags               1
Mature                       2
Meat & Seafood               1
Media                        2
Needs Review                11
Office & School              4
Pantry                       6
Produce                      3
Restaurant                   5
Snacks                    4094
Sporting Goods               3
Toys & Games                 3
Vehicles & Parts             1
Name: BRAND, dtype: int64
```

According to the above image, most of the brands belong to the category Health and Wellness and Snacks.

- BARCODE is a field that should be unique to each product. This field has duplicates and null values posing an issue over data quality.
  Missing BARCODE values can pose an issue in identifying the product.
  Most of the duplicate BARCODE values are associated with the same values of categories, manufacturers, and brands implying that these rows are redundant and it is safe to say that these values can be dropped.
  However, a few values showcase mismatches for the same barcode values.
  4025 - Total Null Values
  4209 - Total Duplicate Values
  Difference = 185, duplicate values excluding null.

  Null BARCODE values can be dropped because it is the only value that can be used to identify the product. If there is no value in the BARCODE, the row becomes redundant.

SECOND STEP: Dropping all the null/missing values present in BARCODE

```
In [10]: barcode_df = product_df.dropna(subset=['BARCODE'])
         print(barcode_df.isnull().sum())
```

```
CATEGORY_1          111
CATEGORY_2          661
CATEGORY_3        58714
CATEGORY_4       774291
MANUFACTURER     226227
BRAND            226225
BARCODE               0
dtype: int64
```

```
In [11]: print(barcode_df['BARCODE'].duplicated().sum())
```

```
185
```

```
In [32]: #GroupBy duplicate barcode values
duplicate_counts = product_df['BARCODE'].value_counts()
# print(duplicate_counts[duplicate_counts > 1])

#Investigating Duplicate Barcodes
for barcode in duplicate_counts[duplicate_counts>1].index:
    print(f"Duplicate Barcode {barcode}")
    print(product_df[product_df['BARCODE'] == barcode])
    print("-"*50)
```

```
Duplicate Barcode 3423905.0
       CATEGORY_1 CATEGORY_2       CATEGORY_3 CATEGORY_4         MANUFACTURER  \
612573     Snacks      Candy  Chocolate Candy        NaN  THE HERSHEY COMPANY
827242     Snacks      Candy  Chocolate Candy        NaN  THE HERSHEY COMPANY

            BRAND    BARCODE
612573  HERSHEY'S  3423905.0
827242  HERSHEY'S  3423905.0
--------------------------------------------------
Duplicate Barcode 3416105.0
       CATEGORY_1 CATEGORY_2       CATEGORY_3 CATEGORY_4         MANUFACTURER  \
227647     Snacks      Candy  Chocolate Candy        NaN  THE HERSHEY COMPANY
476008     Snacks      Candy  Chocolate Candy        NaN  THE HERSHEY COMPANY

          BRAND    BARCODE
227647  REESE'S  3416105.0
476008  REESE'S  3416105.0
--------------------------------------------------
```

Bad data:

```
--------------------------------------------------
Duplicate Barcode 3422007.0
       CATEGORY_1 CATEGORY_2          CATEGORY_3 CATEGORY_4  \
36017      Snacks      Candy  Candy Variety Pack        NaN
422809     Snacks      Candy     Chocolate Candy        NaN

                MANUFACTURER      BRAND    BARCODE
36017   THE HERSHEY COMPANY  HERSHEY'S  3422007.0
422809  THE HERSHEY COMPANY  HERSHEY'S  3422007.0
--------------------------------------------------
--------------------------------------------------
Duplicate Barcode 50426171.0
       CATEGORY_1 CATEGORY_2       CATEGORY_3 CATEGORY_4 MANUFACTURER   BRAND  \
184572     Snacks      Candy  Chocolate Candy        NaN          NaN     NaN
287404     Snacks      Candy  Chocolate Candy        NaN       NESTLE  NESTLE

          BARCODE
184572  50426171.0
287404  50426171.0
--------------------------------------------------
```

```
In [57]: # Identify duplicated barcodes (keep=False to include all occurrences of duplicates except null)
duplicated_barcodes = barcode_df[barcode_df["BARCODE"].duplicated(keep=False)]

# Create a list of tuples for each row with a duplicated barcode
duplicate_tuples = list(duplicated_barcodes.itertuples(index=False, name=None))

# Display the list of tuples
print(duplicate_tuples)
```

```
[('Health & Wellness', 'Hair Removal', 'Shaving Gel & Cream', "Women's Shaving Gel & Cream", 'PLACEHOLDER MANUFACTU
RER', 'PRORASO', 80199137.0), ('Snacks', 'Candy', 'Mints', nan, 'THE HERSHEY COMPANY', 'ICE BREAKERS', 3400203.0),
('Snacks', 'Candy', 'Chocolate Candy', nan, 'THE HERSHEY COMPANY', 'WHATCHAMACALLIT', 3429907.0), ('Snacks', 'Cand
y', 'Mints', nan, 'THE HERSHEY COMPANY', 'ICE BREAKERS', 3409800.0), ('Snacks', 'Chips', 'Crisps', nan, "TRADER JO
E'S", "TRADER JOE'S", 952811.0), ('Snacks', 'Candy', 'Confection Candy', nan, 'THE HERSHEY COMPANY', "REESE'S", 344
7505.0), ('Snacks', 'Dessert Toppings', 'Ice Cream Sauces & Syrups', nan, 'THE HERSHEY COMPANY', "HERSHEY'S", 34845
00.0), ('Snacks', 'Candy', 'Gum', nan, 'THE HERSHEY COMPANY', 'BUBBLE YUM', 3481103.0), ('Snacks', 'Candy', 'Chocol
ate Candy', nan, 'THE HERSHEY COMPANY', 'CADBURY', 3454206.0), ('Snacks', 'Candy', 'Chocolate Candy', nan, "KINDE
R'S", "KINDER'S", 80177609.0), ('Snacks', 'Fruit & Vegetable Snacks', 'Fruit Snacks', nan, 'LIDL US, LLC', 'LIDL',
20744571.0), ('Snacks', 'Candy', 'Chocolate Candy', nan, 'THE HERSHEY COMPANY', "REESE'S", 3448007.0), ('Snacks',
'Candy', 'Chocolate Candy', nan, 'LIDL US, LLC', 'LIDL', 20146900.0), ('Snacks', 'Candy', 'Chocolate Candy', nan,
'LIDL US, LLC', 'LIDL', 20433871.0), ('Snacks', 'Candy', 'Chocolate Candy', nan, 'THE HERSHEY COMPANY', "REESE'S",
3472705.0), ('Snacks', 'Candy', 'Chocolate Candy', nan, 'NESTLE', 'MILKYBAR', 59939498.0), ('Snacks', 'Candy', 'Min
ts', nan, 'THE HERSHEY COMPANY', 'BREATH SAVERS', 3433706.0), ('Snacks', 'Dessert Toppings', 'Ice Cream Sauces & Sy
rups', nan, 'THE HERSHEY COMPANY', "REESE'S", 3408704.0), ('Health & Wellness', 'Hair Care', 'Hair Color', nan, 'HE
NKEL', 'SCHWARZKOPF', 52336919068.0), ('Snacks', 'Candy', 'Confection Candy', nan, 'THE HERSHEY COMPANY', 'TWIZZLER
S', 3420003.0), ('Snacks', 'Candy', 'Chocolate Candy', nan, "KINDER'S", "KINDER'S", 80177616.0), ('Snacks', 'Cand
y', 'Gum', nan, 'THE HERSHEY COMPANY', 'ICE BREAKERS', 3464502.0), ('Snacks', 'Candy', 'Candy Variety Pack', nan,
'THE HERSHEY COMPANY', "HERSHEY'S", 3422007.0), ('Snacks', 'Candy', 'Chocolate Candy', nan, 'THE HERSHEY COMPANY',
```

The data quality in this table is compromised given that BARCODE is not unique to each product and has duplicates and null values which can lead to issues in referencing.

CATEGORY_4 has the most null values but does not seem to be a potential barrier assuming this field is optional and needs to be very specific giving more description about the product.

TRANSACTION_TAKEHOME:

FIRST STEP: Identifying the null/missing values in all the columns

```
In [65]: print(transaction_df.isnull().sum())

RECEIPT_ID            0
PURCHASE_DATE         0
SCAN_DATE             0
STORE_NAME            0
USER_ID               0
BARCODE            5762
FINAL_QUANTITY        0
FINAL_SALE            0
dtype: int64
```

SELECT * FROM TRANSACTION_TAKEHOME

	RECEIPT_ID	PURCHASE_DATE	SCAN_DATE	STORE_NAME	USER_ID	BARCODE	FINAL_QUANTITY	FINAL_SALE
1	0000d256-4041-4a3e-adc4-5623fb6e0c99	2024-08-21	2024-08-21 14:19:06.539 Z	WALMART	63b73a7f3d310dceeabd4758	015300014978	1.00	*NULL*
2	0001455d-7a92-4a7b-a1d2-c747af1c8fd3	2024-07-20	2024-07-20 09:50:24.206 Z	ALDI	62c08877baa38d1a1f6c211a	*NULL*	zero	1.49
3	00017e0a-7851-42fb-bfab-0baa96e23586	2024-08-18	2024-08-19 15:38:56.813 Z	WALMART	60842f207ac8b7729e472020	078742229751	1.00	*NULL*
4	000239aa-3478-453d-801e-66a82e39c8af	2024-06-18	2024-06-19 11:03:37.468 Z	FOOD LION	63fcd7cea4f8442c3386b589	783399746536	zero	3.49
5	00026b4c-dfe8-49dd-b026-4c2f0fd5e6a1	2024-07-04	2024-07-05 15:56:43.549 Z	RANDALLS	6193231ae9b3d75037b0f928	047900501183	1.00	*NULL*
6	0002d8cd-1701-4cdd-a524-b70402e2dbc0	2024-06-24	2024-06-24 19:44:54.247 Z	WALMART	5dcc6c510040a012b8e76924	681131411295	zero	1.46
7	000550b2-1480-4c07-950f-ff601f242152	2024-07-06	2024-07-06 19:27:48.586 Z	WALMART	5f850bc9cf9431165f3ac175	049200905548	1.00	*NULL*

SECOND STEP: Cross-verifying that if the duplicate barcode value present in PRODUCTS_TAKEHOME is present in the TRANSACTION_TAKEHOME table

--Testing to see if duplicate barcodes are present in the TRANSACTION TABLE, which is NOT present in the PRODUCT table.

```
SELECT * FROM TRANSACTION_TAKEHOME
WHERE BARCODE in ("017000329260", "052336919068")
```

RECEIPT_ID	PURCHASE_DATE	SCAN_DATE	STORE_NAME	USER_ID	BARCODE	FINAL_QUANTITY	FINAL_SALE

```
Execution finished without errors.
Result: 0 rows returned in 11ms
At line 34:
SELECT * FROM TRANSACTION_TAKEHOME
  WHERE BARCODE in ("017000329260", "052336919068")
```

THIRD STEP: Investigating more on the primary key value of the table, RECEIPT_ID

→ After running the SQL query, out of 50000 values of RECEIPT_ID, only 24440 are distinct. There are zero null values in this column implying that the rest of the count of receipt_id's are duplicates.

```
 -- DISTINCT  count of RECEIPT_ID
SELECT count (DISTINCT RECEIPT_ID) FROM
TRANSACTION_TAKEHOME
```

	count (DISTINCT RECEIPT_ID)
1	24440

FOURTH STEP:

-- Eliminating null values from FINAL_SALE and zero value from FINAL_QUANTITY. Assuming that a transaction has been completed, the final quantity and final sale cannot be zero or null. We can safely drop these values.

```
SELECT *
```

FROM TRANSACTION_TAKEHOME
WHERE FINAL_SALE is not null and FINAL_QUANTITY <> "zero"
ORDER BY RECEIPT_ID

	RECEIPT_ID	PURCHASE_DATE	SCAN_DATE	STORE_NAME	USER_ID	BARCODE	FINAL_QUANTITY	FINAL_SALE
1	0000d256-4041-4a3e-adc4-5623fb6e0c99	2024-08-21	2024-08-21 14:19:06.539 Z	WALMART	63b73a7f3d310dceeabd4758	015300014978	1.00	1.54
2	0001455d-7a92-4a7b-a1d2-c747af1c8fd3	2024-07-20	2024-07-20 09:50:24.206 Z	ALDI	62c08877baa38d1a1f6c211a	*NULL*	1.00	1.49
3	00017e0a-7851-42fb-bfab-0baa96e23586	2024-08-18	2024-08-19 15:38:56.813 Z	WALMART	60842f207ac8b7729e472020	078742229751	1.00	2.54
4	000239aa-3478-453d-801e-66a82e39c8af	2024-06-18	2024-06-19 11:03:37.468 Z	FOOD LION	63fcd7cea4f8442c3386b589	783399746536	1.00	3.49
5	00026b4c-dfe8-49dd-b026-4c2f0fd5c6a1	2024-07-04	2024-07-05 15:56:43.549 Z	RANDALLS	6193231ae9b3d75037b0f928	047900501183	1.00	5.29
6	0002d8cd-1701-4cdd-a524-b70402e2dbc0	2024-06-24	2024-06-24 19:44:54.247 Z	WALMART	5dcc6c510040a012b8e76924	681131411295	1.00	1.46
7	000550b2-1480-4c07-950f-ff601f242152	2024-07-06	2024-07-06 19:27:48.586 Z	WALMART	5f850bc9cf9431165f3ac175	049200905548	1.00	3.12

FIFTH STEP:

--Checking receipt_id's that have barcode value as NULL, this data can be redundant
SELECT *
FROM TRANSACTION_TAKEHOME
WHERE FINAL_SALE is not null and FINAL_QUANTITY <> "zero" and BARCODE is null
ORDER BY RECEIPT_ID

	RECEIPT_ID	PURCHASE_DATE	SCAN_DATE	STORE_NAME	USER_ID	BARCODE	FINAL_QUANTITY	FINAL_SALE
1	0001455d-7a92-4a7b-a1d2-c747af1c8fd3	2024-07-20	2024-07-20 09:50:24.206 Z	ALDI	62c08877baa38d1a1f6c211a	*NULL*	1.00	1.49
2	0010d87d-1ad2-4e5e-9a25-cec736919d15	2024-08-04	2024-08-04 18:01:47.787 Z	ALDI	66686fc2e04f743a096ea808	*NULL*	1.00	2.29
3	002ee298-d907-40ca-921a-586468571f76	2024-07-15	2024-07-16 16:42:19.211 Z	ALDI	63de64b1dcb50fbd3084f142	*NULL*	1.00	2.49
4	00326689-e763-4b27-9ad5-202fc93609e2	2024-06-19	2024-06-20 08:59:38.397 Z	ALDI	6158642597d737581b5d30ee	*NULL*	1.00	1.89
5	00a9e033-e49d-45d6-990e-90631f82776e	2024-09-05	2024-09-05 11:10:54.831 Z	ALDI	5d4f08e962fb4a4a58574e7f	*NULL*	1.00	2.09
6	00b63114-f9dc-433e-a50a-daebd81a27c3	2024-07-27	2024-08-06 18:00:36.816 Z	HOBBY LOBBY	6403b86180552327896fca11	*NULL*	1.00	1.79
7	00b827a6-e7b9-4eaa-8c85-6a11383e1885	2024-08-06	2024-08-13 17:09:48.281 Z	ALDI	61e33a9494e276361d8a6bf0	*NULL*	1.00	0.39

SIXTH STEP:

--Inner Join with user table, 12 records. We can drop these 12 transactions that have no barcode.
Assumption: 12 records without barcodes can be discarded as the count is pretty less.

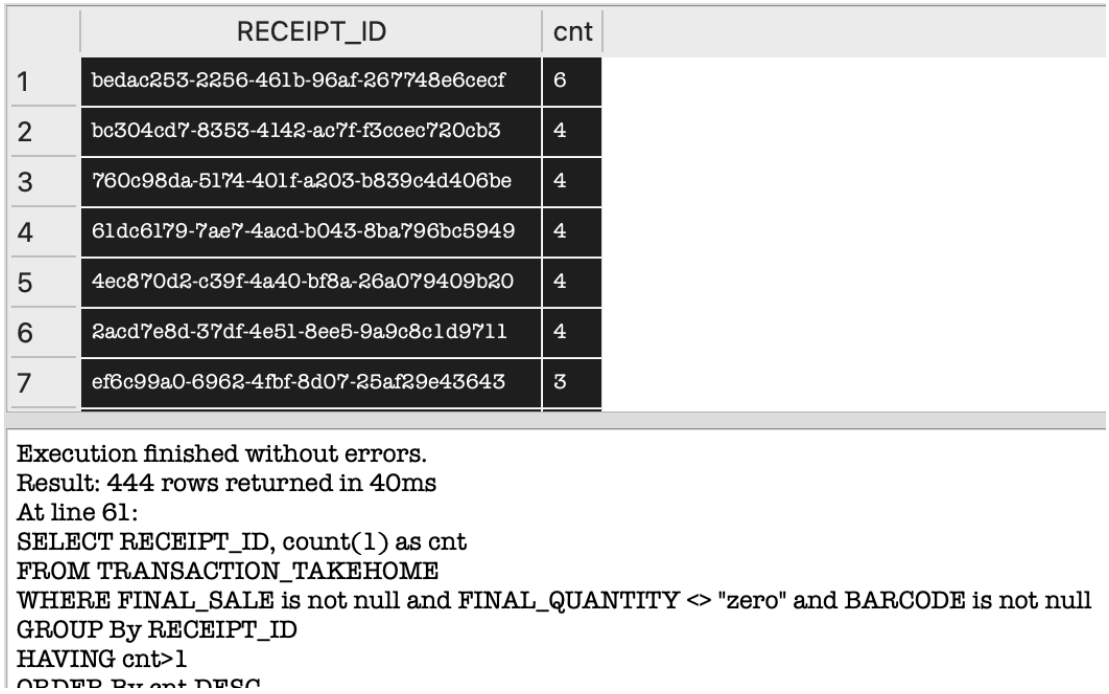
SELECT t.*, u.*
FROM TRANSACTION_TAKEHOME as t
INNER JOIN USER_TAKEHOME as u on u.ID = t.USER_ID

WHERE t.FINAL_SALE is not null and t.FINAL_QUANTITY <> "zero" and t.BARCODE is null

	RECEIPT_ID	PURCHASE_DATE	SCAN_DATE	STORE_NAME	USER_ID	BARCODE	FINAL_QUANTITY	FINAL_SALE	ID	CREATED_DATE	BIRTH_DATE
1	b94eb3c1-88aa-4785-be4c-f54b6679b816	2024-07-15	2024-07-16 12:02:24.171 Z	CVS	61730bba65abe727fff3fcf7	*NULL*	1.00	9.11	61730bba65abe727fff3fcf7	2021-10-22 19:06:34.000 Z	1954-07-12 00:00:00.00
2	9c47ad1e-08df-4c21-944a-d7ae1f1fc8a3	2024-09-06	2024-09-07 18:02:11.141 Z	WALMART	5f6518d1bf3f5a43fdd0c9a5	*NULL*	1.00	2.62	5f6518d1bf3f5a43fdd0c9a5	2020-09-18 20:30:09.000 Z	1955-02-19 05:00:00.00
3	9c47ad1e-08df-4c21-944a-d7ae1f1fc8a3	2024-09-06	2024-09-07 18:02:11.141 Z	WALMART	5f6518d1bf3f5a43fdd0c9a5	*NULL*	1.00	2.62	5f6518d1bf3f5a43fdd0c9a5	2020-09-18 20:30:09.000 Z	1955-02-19 05:00:00.00
4	e642e1e3-f463-4385-8e69-8e79d52e7241	2024-07-05	2024-07-07 11:03:59.418 Z	SAVE A LOT	59725916e4b01bd2053089fe	*NULL*	1.00	2.99	59725916e4b01bd2053089fe	2017-07-21 19:42:14.000 Z	1977-11-02 00:00:00.00
5	485d2ad5-e57e-4ec8-8483-11ac2fcac205	2024-06-30	2024-07-01 16:10:53.883 Z	ALDI	5fcd25273f814c1271450841	*NULL*	1.00	2.95	5fcd25273f814c1271450841	2020-12-06 18:38:31.000 Z	1972-11-15 00:00:00.00
6	1b81676d-750e-42d5-970c-b9432b09934f	2024-07-08	2024-07-08 23:27:59.639 Z	LODE	610a8541ca1fab5b417b5d33	*NULL*	1.00	4.29	610a8541ca1fab5b417b5d33	2021-08-04 12:17:05.000 Z	1977-01-12 00:00:00.00
7	6a4bcdd5-7469-45cb-8e32-a1f744ee3ab9	2024-07-09	2024-07-11 13:31:13.819 Z	ALDI	63faaa3e38f010745ba05f9f	*NULL*	1.00	1.75	63faaa3e38f010745ba05f9f	2023-02-26 00:39:26.000 Z	1980-10-24 00:00:00.00

SEVENTH STEP:

--Checking for duplicate receipt_id's
SELECT RECEIPT_ID, count(1) as cnt
FROM TRANSACTION_TAKEHOME
WHERE FINAL_SALE is not null and FINAL_QUANTITY <> "zero" and BARCODE is not null
GROUP By RECEIPT_ID
HAVING cnt>1
ORDER By cnt DESC



	RECEIPT_ID	cnt
1	bedac253-2256-461b-96af-267748e6cecf	6
2	bc304cd7-8353-4142-ac7f-f3ccec720cb3	4
3	760c98da-5174-401f-a203-b839c4d406be	4
4	61dc6179-7ae7-4acd-b043-8ba796bc5949	4
5	4ec870d2-c39f-4a40-bf8a-26a079409b20	4
6	2acd7e8d-37df-4e51-8ee5-9a9c8c1d9711	4
7	ef6c99a0-6962-4fbf-8d07-25af29e43643	3

```
Execution finished without errors.
Result: 444 rows returned in 40ms
At line 61:
SELECT RECEIPT_ID, count(1) as cnt
FROM TRANSACTION_TAKEHOME
WHERE FINAL_SALE is not null and FINAL_QUANTITY <> "zero" and BARCODE is not null
GROUP By RECEIPT_ID
HAVING cnt>1
ORDER By cnt DESC
```

EIGHTH STEP:

--Checking a particular RECEIPT_ID
select * from TRANSACTION_TAKEHOME where RECEIPT_ID =
"bedac253-2256-461b-96af-267748e6cecf"

	RECEIPT_ID	PURCHASE_DATE	SCAN_DATE	STORE_NAME	USER_ID	BARCODE	FINAL_QUANTITY	FINAL_SALE
1	bedac253-2256-461b-96af-267748e6cecf	2024-09-08	2024-09-08 20:00:42.348 Z	KROGER	614f7e8081627974a57c8a9e	011110121141	zero	0.89
2	bedac253-2256-461b-96af-267748e6cecf	2024-09-08	2024-09-08 20:00:42.348 Z	KROGER	614f7e8081627974a57c8a9e	011110121141	1.00	*NULL*
3	bedac253-2256-461b-96af-267748e6cecf	2024-09-08	2024-09-08 20:00:42.348 Z	KROGER	614f7e8081627974a57c8a9e	011110121141	zero	0.89
4	bedac253-2256-461b-96af-267748e6cecf	2024-09-08	2024-09-08 20:00:42.348 Z	KROGER	614f7e8081627974a57c8a9e	011110121141	1.00	*NULL*
5	bedac253-2256-461b-96af-267748e6cecf	2024-09-08	2024-09-08 20:00:42.348 Z	KROGER	614f7e8081627974a57c8a9e	011110121141	zero	0.89
6	bedac253-2256-461b-96af-267748e6cecf	2024-09-08	2024-09-08 20:00:42.348 Z	KROGER	614f7e8081627974a57c8a9e	011110121141	1.00	*NULL*
7	bedac253-2256-461b-96af-267748e6cecf	2024-09-08	2024-09-08 20:00:42.348 Z	KROGER	614f7e8081627974a57c8a9e	011110121141	1.00	0.89

Execution finished without errors.
Result: 12 rows returned in 11ms
At line 69:
select * from TRANSACTION_TAKEHOME where RECEIPT_ID = "bedac253-2256-461b-96af-267748e6cecf"

NINTH STEP:

--Checking availability in product and transaction table for a particular RECEIPT_ID
SELECT t.*, p.*
FROM TRANSACTION_TAKEHOME as t
INNER JOIN PRODUCTS_TAKEHOME as p
ON t.BARCODE = p.BARCODE
WHERE t.FINAL_SALE is not null and t.FINAL_QUANTITY <> "zero" and
t.BARCODE is not null and RECEIPT_ID =
"eb8b58c3-182a-4623-8492-0b8231b85135"

	RECEIPT_ID	PURCHASE_DATE	SCAN_DATE	STORE_NAME	USER_ID	BARCODE	FINAL_QUANTITY	FINAL_SALE	CATEGORY_1	CATEGORY_2
1	7b3ec72d-9d30-40b8-b185-0bfb538942a9	2024-08-20	2024-08-20 11:17:29.633 Z	DOLLAR GENERAL STORE	60fc1e6deb7585430ff52ee7	745527114884	1.00	1.65	Snacks	Cookies
2	04859b68-29e3-4e8d-9bdb-980046fc3473	2024-08-05	2024-08-09 16:06:00.570 Z	DOLLAR GENERAL STORE	654cf234a825ea102b81072e	745527114884	1.00	1.65	Snacks	Cookies
3	7ee1798e-fd2e-4278-858b-f417fdcafe08	2024-08-30	2024-09-04 12:53:31.478 Z	DOLLAR GENERAL STORE	65c29b137050d0a6206cd24f	012000504051	1.00	5.25	Beverages	Carbonated Soft Drinks
4	30977cbc-1d29-4f2d-851c-110443769d0	2024-09-01	2024-09-01 09:40:16.103 Z	WALMART	5baf733455806419c416c3be	037000779704	1.00	2.20	Health & Wellness	Oral Care
5	48c7720b-7097-4cee-995e-721e52c623bd	2024-06-25	2024-06-25 17:56:43.654 Z	COSTCO	66c5b9a416cc39173210ae18	000009697867	1.00	9.69	Snacks	Cookies
6	e70b5591-92a5-4d9f-8d52-5525ef91cfaf	2024-06-20	2024-06-21 11:32:23.957 Z	WALMART	62f069014e73e2db30ecab93	017000132556	1.00	8.76	Health & Wellness	Bath & Body
7	58ed121-a5cd-4b92-bbdd-9c8524ff9c3f	2024-06-12	2024-06-12 14:36:43.256 Z	WALGREENS	5ed4fbc746278a13f7d2fd93	049022697195	1.00	4.79	Snacks	Cookies

Execution finished without errors.
Result: 12413 rows returned in 912ms
At line 72:
SELECT t.*, p.*
FROM TRANSACTION_TAKEHOME as t
INNER JOIN PRODUCTS_TAKEHOME as p
ON t.BARCODE = p.BARCODE
WHERE t.FINAL_SALE is not null and t.FINAL_QUANTITY <> "zero" and t.BARCODE is not null --and RECEIPT_ID = "eb8b58c3-182a-4623-8492-0b8231b85135"

	RECEIPT_ID	PURCHASE_DATE	SCAN_DATE	STORE_NAME	USER_ID	BARCODE	FINAL_QUANTITY	FINAL_SALE	CATEGORY_1	CATEGORY_2	CATEGORY_3	CATEGORY_
1	eb8b58c3-182a-4623-8492-0b8231b85135	2024-09-07	2024-09-07 13:25:13.203 Z	WALMART	6612d57d940386b19ee22088	888109010089	1.00	1.68	Snacks	Snack Cakes	Cakes & Truffles Snack Cakes	*NULL*
2	eb8b58c3-182a-4623-8492-0b8231b85135	2024-09-07	2024-09-07 13:25:13.203 Z	WALMART	6612d57d940386b19ee22088	888109010089	1.00	1.68	Snacks	Snack Cakes	Cakes & Truffles Snack Cakes	*NULL*
3	eb8b58c3-182a-4623-8492-0b8231b85135	2024-09-07	2024-09-07 13:25:13.203 Z	WALMART	6612d57d940386b19ee22088	888109010089	1.00	1.68	Snacks	Snack Cakes	Cakes & Truffles Snack Cakes	*NULL*

```
Execution finished without errors.
Result: 3 rows returned in 887ms
At line 72:
SELECT t.*, p.*
FROM TRANSACTION_TAKEHOME as t
INNER JOIN PRODUCTS_TAKEHOME as p
ON t.BARCODE = p.BARCODE
WHERE  t.FINAL_SALE is not null and t.FINAL_QUANTITY <> "zero" and t.BARCODE is not null and RECEIPT_ID = "eb8b58c3-182a-4623-8492-0b8231b85135"
```

TENTH STEP:

--Checking the final duplicate RECEIPT_ID
SELECT RECEIPT_ID, count(1) as cnt
FROM TRANSACTION_TAKEHOME as t
INNER JOIN PRODUCTS_TAKEHOME as p
ON t.BARCODE = p.BARCODE
WHERE t.FINAL_SALE is not null and t.FINAL_QUANTITY <> "zero" and t.BARCODE is not null
GROUP By RECEIPT_ID
HAVING cnt>1
ORDER By cnt DESC

	RECEIPT_ID	cnt
1	61dc6179-7ae7-4acd-b043-8ba796bc5949	4
2	eb8b58c3-182a-4623-8492-0b8231b85135	3
3	de9f2ef7-2975-4cfd-a860-c06d89935e35	3
4	682cb059-74a1-4c47-abd8-5fd6541d88bf	3
5	67043629-5fbc-4f7d-ad0f-b7c14ea6aa37	3
6	43955b35-6fbc-4909-a4de-1a0de0dc387f	3
7	431fe612-ed55-470e-939c-043ad31f33f3	3

```
Execution finished without errors.
Result: 125 rows returned in 462ms
At line 79:
SELECT RECEIPT_ID, count(1) as cnt
FROM TRANSACTION_TAKEHOME as t
INNER JOIN PRODUCTS_TAKEHOME as p
ON t.BARCODE = p.BARCODE
WHERE  t.FINAL_SALE is not null and t.FINAL_QUANTITY <> "zero" and t.BARCODE is not null
GROUP By RECEIPT_ID
```

ELEVENTH STEP:

```
--CTE (Info table with all the relevant transactions and products, final table)
WITH info AS (
   SELECT t.*, p.*
         FROM TRANSACTION_TAKEHOME as t
         INNER JOIN PRODUCTS_TAKEHOME as p
         ON t.BARCODE = p.BARCODE
         WHERE  t.FINAL_SALE is not null and t.FINAL_QUANTITY <> "zero" and
t.BARCODE is not null
)
SELECT *
FROM info;
```

	RECEIPT_ID	PURCHASE_DATE	SCAN_DATE	STORE_NAME	USER_ID	BARCODE	FINAL_QUANTITY	FINAL_SALE	CATEGORY_1	CATEGORY_2
1	7b3ec72d-9d30-40b8-b185-0bfb638942a9	2024-08-20	2024-08-20 11:17:29.633 Z	DOLLAR GENERAL STORE	60fc1e6deb7585430ff52ee7	745527114884	1.00	1.65	Snacks	Cookies
2	04869b68-29e3-4e8d-9bdb-950046fc3473	2024-08-05	2024-08-09 16:06:00.570 Z	DOLLAR GENERAL STORE	654cf234a225ea102b81072e	745527114884	1.00	1.65	Snacks	Cookies
3	7ee1798e-fd2e-4278-838b-f417fdcafe08	2024-08-30	2024-09-04 12:53:31.478 Z	DOLLAR GENERAL STORE	66c29b137050d0a6206cd24f	012000504081	1.00	5.25	Beverages	Carbonated Soft Drinks
4	30977cbc-1d29-4f2d-881c-110443276d0	2024-09-01	2024-09-01 09:40:16.103 Z	WALMART	5baf7334586206419c416c3be	037000779704	1.00	2.20	Health & Wellness	Oral Care
5	48c7720b-7097-4cee-995e-721e52c623bd	2024-06-25	2024-06-25 17:56:43.664 Z	COSTCO	65cfb9a416cc39173210ae15	000009697867	1.00	9.69	Snacks	Cookies
6	c70b5591-92a5-4d9f-8d62-5525cf91cfaf	2024-06-20	2024-06-21 11:32:23.957 Z	WALMART	62f069014e73e2db30ecab93	017000132556	1.00	8.76	Health & Wellness	Bath & Body
7	58edc121-a5cd-4b92-bbdd-9c8524ff9e3f	2024-06-12	2024-06-12 14:38:43.256 Z	WALGREENS	5ed4fbc746278a13f7d2fd93	049022697195	1.00	4.79	Snacks	Cookies

```
Execution finished without errors.
Result: 12413 rows returned in 1080ms
At line 89:
WITH info AS (
   SELECT t.*, p.*
         FROM TRANSACTION_TAKEHOME as t
         INNER JOIN PRODUCTS_TAKEHOME as p
         ON t.BARCODE = p.BARCODE
         WHERE  t.FINAL_SALE is not null and t.FINAL_QUANTITY <> "zero" and t.BARCODE is not null
```

USER TAKEHOME

FIRST STEP:

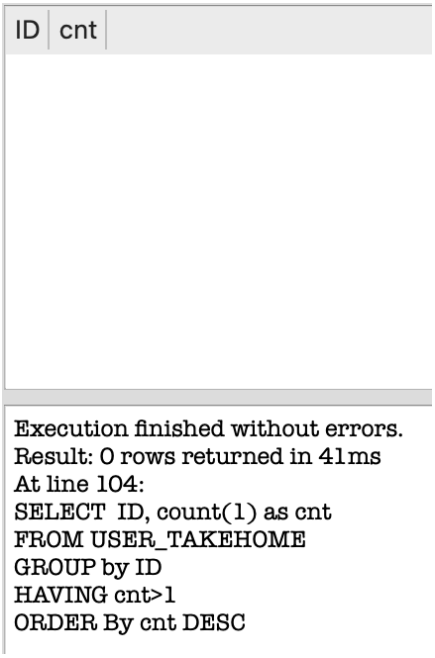
```
SELECT * FROM USER_TAKEHOME
```

	ID	CREATED_DATE	BIRTH_DATE	STATE	LANGUAGE	GENDER
1	5ef3b4f17053ab141787697d	2020-06-24 20:17:54.000 Z	2000-08-11 00:00:00.000 Z	CA	es-419	female
2	5ff220d383fcfc12622b96bc	2021-01-03 19:53:55.000 Z	2001-09-24 04:00:00.000 Z	PA	en	female
3	6477950aa55bb77a0e27ee10	2023-05-31 18:42:18.000 Z	1994-10-28 00:00:00.000 Z	FL	es-419	female
4	658a306e99b40f103b63ccf8	2023-12-26 01:46:22.000 Z	*NULL*	NC	en	*NULL*
5	653cf5d6a225ea102b7ecdc2	2023-10-28 11:51:50.000 Z	1972-03-19 00:00:00.000 Z	PA	en	female
6	5fe2b6f3ad416a1265c4ab68	2020-12-23 03:18:11.000 Z	1999-10-27 04:00:00.000 Z	NY	en	female
7	6512105468l6bb4d035b1ead	2023-09-25 22:57:24.000 Z	1983-09-25 22:57:25.000 Z	FL	es-419	male

```
Execution finished without errors.
Result: 100000 rows returned in 75ms
At line 101:
SELECT * FROM USER_TAKEHOME
```

SECOND STEP:

```
-- Checking for duplicates in ID
SELECT  ID, count(1) as cnt
FROM USER_TAKEHOME
GROUP by ID
HAVING cnt>1
ORDER By cnt DESC
```



ID	cnt

```
Execution finished without errors.
Result: 0 rows returned in 41ms
At line 104:
SELECT  ID, count(1) as cnt
FROM USER_TAKEHOME
GROUP by ID
HAVING cnt>1
ORDER By cnt DESC
```

THIRD STEP:

```
--Checking if there are null VALUES
SELECT *
FROM USER_TAKEHOME
WHERE BIRTH_DATE is null
```

	ID	CREATED_DATE	BIRTH_DATE	STATE	LANGUAGE	GENDER
1	658a306e99b40f103b63ccf8	2023-12-26 01:46:22.000 Z	*NULL*	NC	en	*NULL*
2	6568861a9cb6a4f05dc9ccda	2023-11-30 12:54:50.000 Z	*NULL*	NC	en	*NULL*
3	64fa0e8c791f8099cbbd1d27	2023-09-07 17:55:24.000 Z	*NULL*	*NULL*	en	*NULL*
4	655fe9a1dd170fff1657a048	2023-11-24 00:09:05.000 Z	*NULL*	LA	en	*NULL*
5	65206ec67933bcf6ee794eaf	2023-10-06 20:32:06.000 Z	*NULL*	TX	en	*NULL*
6	659788b0abb3371350c97e94	2024-01-05 04:42:24.000 Z	*NULL*	CA	en	*NULL*
7	6590f16299b40f103b64b0c0	2023-12-31 04:43:14.000 Z	*NULL*	*NULL*	en	female

```
Execution finished without errors.
Result: 3675 rows returned in 35ms
At line 111:
SELECT *
FROM USER_TAKEHOME
WHERE BIRTH_DATE is null
```

FOURTH STEP:

```
--Checking for particular userID
SELECT *
FROM USER_TAKEHOME
WHERE ID = "60fc1e6deb7585430ff52ee7"
```

ID	CREATED_DATE	BIRTH_DATE	STATE	LANGUAGE	GENDER

```
Execution finished without errors.
Result: 0 rows returned in 10ms
At line 116:
SELECT *
FROM USER_TAKEHOME
WHERE ID = "60fc1e6deb7585430ff52ee7"
```

FIFTH STEP:

-- INNER JOIN TRANSACTION and USER. Only 262 people have done legible transactions

```
SELECT t.*, u.*
FROM TRANSACTION_TAKEHOME as t
LEFT JOIN USER_TAKEHOME as u
ON t.USER_ID = u.ID
WHERE ID is not null
```

	RECEIPT_ID	PURCHASE_DATE	SCAN_DATE	STORE_NAME	USER_ID	BARCODE	FINAL_QUANTITY	FINAL_SALE	ID	CREATED_DATE	
1	cdcc0ef0-9f72-45d4-aecf-701ce553f792	2024-06-12	2024-06-16 10:00:31.951 Z	WALMART	62c303aebaa38d1a1f6d0d51	078742222349	1.00	*NULL*	62c303aebaa38d1a1f6d0d51	2022-07-04 15:13:50.000 Z	195
2	cdcc0ef0-9f72-45d4-aecf-701ce553f792	2024-06-12	2024-06-16 10:00:31.951 Z	WALMART	62c303aebaa38d1a1f6d0d51	078742222349	1.00	3.24	62c303aebaa38d1a1f6d0d51	2022-07-04 15:13:50.000 Z	195
3	8c6b21b7-4f9e-4316-98e5-8542c5491871	2024-08-30	2024-09-03 12:35:17.267 Z	TARGET	5fd4fb485f410d44bae3a776	022400643366	2.00	*NULL*	5fd4fb485f410d44bae3a776	2020-12-12 17:18:00.000 Z	197
4	8c6b21b7-4f9e-4316-98e5-8542c5491871	2024-08-30	2024-09-03 12:35:17.267 Z	TARGET	5fd4fb485f410d44bae3a776	022400643366	2.00	14.58	5fd4fb485f410d44bae3a776	2020-12-12 17:18:00.000 Z	197
5	a7e6adf6-3dac-497f-90f1-7f371c639a1f	2024-07-22	2024-07-22 09:49:41.406 Z	TARGET	5b441360be53340f289b0795	270071433990	1.00	7.99	5b441360be53340f289b0795	2018-07-10 02:01:04.000 Z	198
6	a7e6adf6-3dac-497f-90f1-7f371c639a1f	2024-07-22	2024-07-22 09:49:41.406 Z	TARGET	5b441360be53340f289b0795	270071433990	zero	7.99	5b441360be53340f289b0795	2018-07-10 02:01:04.000 Z	198
7	c7b129ac-0599-464c-948c-5ba783930982	2024-07-08	2024-07-09 05:34:12.331 Z	WALMART	5b441360be53340f289b0795	026700129155	1.00	*NULL*	5b441360be53340f289b0795	2018-07-10 02:01:04.000 Z	198

```
Execution finished without errors.
Result: 262 rows returned in 98ms
At line 121:
SELECT t.*, u.*
FROM TRANSACTION_TAKEHOME as t
LEFT JOIN USER_TAKEHOME as u
ON t.USER_ID = u.ID
WHERE ID is not null
```

SIXTH STEP:

Created Common Table Expressions

```
WITH info AS (
   SELECT t.*, p.*
        FROM TRANSACTION_TAKEHOME as t
        INNER JOIN PRODUCTS_TAKEHOME as p
        ON t.BARCODE = p.BARCODE
        WHERE  t.FINAL_SALE is not null and t.FINAL_QUANTITY <> "zero" and
t.BARCODE is not null
)
SELECT i.*, u.*
FROM info as i
LEFT JOIN USER_TAKEHOME as u
ON i.USER_ID = u.ID
```

2. Are there any fields that are challenging to understand?

→ In the table PRODUCTS_TAKEHOME, I found the field BARCODE challenging to understand. The field Barcode should be unique to every category. However, multiple null values and duplicates were found for various products of the same category. Based on a few assumptions, the null barcodes were dropped to clean up the data and ensure that the barcodes were unique and referred to the right product.

→ I found two duplicate values in the BARCODE field. However, when I checked the transaction table for the same values, they were not found, implying that the product associated with these barcodes was not purchased.

→ In the table TRANSACTION_TAKEHOME, the field receipt_id is the primary key used to identify the transactions made for a product by the user. Initially, I found duplicates of these receipt_id's. After researching the transaction table, I concluded that one receipt_id is associated with multiple transactions of the same user.

→ The column, FINAL_QUANTITY and FINAL_SALE seemed to be tricky. Some values in the FINAL_QUANTITY were 'zero,' and some values in FINAL_SALE were null. For any transaction to be successful, neither of these values should be empty. Hence, the null values for FINAL_QUANTITY, FINAL_SALE, and BARCODE were dropped.

→ In the table USER_TAKEHOME, I did not find any columns to be challenging. The ID is the primary key and has no null values. Based on the ID present in both the transaction table and user table, only 262 users are legible users who have had transactions made.