Q1 What are the top 5 brands by receipts scanned among users 21 and over?

A screenshot of a computer program

Description automatically generated

-- Common Table Expression (CTE) to filter out transactions with a final sale value of zero

WITH transaction\_data\_sale\_not\_null AS (

SELECT \* FROM TRANSACTION\_TAKEHOME

WHERE CAST(FINAL\_SALE AS FLOAT) != 0.00),

-- CTE to concatenate receipt\_id and barcode (primary key) and grouping them to remove duplicates

transaction\_data\_sale\_no\_duplicates AS (

SELECT

(RECEIPT\_ID || COALESCE(BARCODE, '0')) AS combined\_pk,

RECEIPT\_ID,

BARCODE,

USER\_ID,

FINAL\_SALE

FROM

transaction\_data\_sale\_not\_null

GROUP BY combined\_pk)

-- Main select query

SELECT

p.BRAND,

COUNT(t.RECEIPT\_ID) AS receipt\_count -- Using count of receipt id to check count of receipts scanned

FROM

transaction\_data\_sale\_no\_duplicates AS t

JOIN

USER\_TAKEHOME AS u ON t.USER\_ID = u.ID

JOIN

PRODUCTS\_TAKEHOME AS p ON t.BARCODE = p.BARCODE

WHERE

strftime('%Y', 'now') - strftime('%Y', u.BIRTH\_DATE) >= 21 -- Filtering for users > 21

AND p.BRAND <> '' -- Removing cases where brand is blank

GROUP BY p.BRAND ORDER BY receipt\_count DESC LIMIT 5;

Q2 What are the top 5 brands by sales among users that have had their account for at least six months?

A screenshot of a computer program

Description automatically generated

-- Common Table Expression (CTE) to filter out transactions with a final sale value of zero

WITH transaction\_data\_sale\_not\_null AS (

SELECT

\*

FROM

TRANSACTION\_TAKEHOME

WHERE

CAST(FINAL\_SALE AS FLOAT) != 0.00

),

-- CTE to concatenate receipt\_id and barcode (primary key) and grouping them to remove duplicates

transaction\_data\_sale\_no\_duplicates AS (

SELECT

(RECEIPT\_ID || COALESCE(BARCODE, '0')) AS combined\_pk,

RECEIPT\_ID,

BARCODE,

USER\_ID,

FINAL\_SALE

FROM

transaction\_data\_sale\_not\_null

GROUP BY

combined\_pk

)

-- Main select query

SELECT

p.BRAND,

SUM(CAST(t.FINAL\_SALE AS FLOAT)) AS total\_sales -- Calculating total sales per brand

FROM

transaction\_data\_sale\_no\_duplicates AS t

JOIN

USER\_TAKEHOME AS u ON t.USER\_ID = u.ID -- Join the filtered transactions with the USER\_TAKEHOME table on user ID

JOIN

PRODUCTS\_TAKEHOME AS p ON t.BARCODE = p.BARCODE -- Join the resulting table with PRODUCTS\_TAKEHOME on barcode to access product details

WHERE

DATE(u.CREATED\_DATE) <= DATE('now', '-6 months') -- Filter to include transactions where the user was created at least 6 months ago

AND p.BRAND IS NOT NULL -- Filter out any records where the product brand is null

GROUP BY

p.BRAND

ORDER BY

total\_sales DESC

LIMIT 5;

Q3 What is the percentage of sales in the Health & Wellness category by generation? – **I have defined generations based on age**

A screenshot of a computer program

Description automatically generated

Output:

A screenshot of a computer

Description automatically generated

-- Common Table Expression (CTE) to filter out transactions with a final sale value of zero

WITH transaction\_data\_sale\_not\_null AS (

SELECT \* FROM TRANSACTION\_TAKEHOME WHERE CAST(FINAL\_SALE AS FLOAT) != 0.00),

-- CTE to concatenate receipt\_id and barcode (primary key) and grouping them to remove duplicates

transaction\_data\_sale\_no\_duplicates AS (

SELECT

(RECEIPT\_ID || COALESCE(BARCODE, '0')) AS combined\_pk, RECEIPT\_ID, BARCODE, USER\_ID, FINAL\_SALE

FROM transaction\_data\_sale\_not\_null GROUP BY combined\_pk)

-- Main select query

SELECT CASE -- Creating generations based on birth date

WHEN strftime('%Y', 'now') - strftime('%Y', u.birth\_date) >= 76 THEN 'Silent Generation'

WHEN strftime('%Y', 'now') - strftime('%Y', u.birth\_date) BETWEEN 57 AND 75 THEN 'Baby Boomers'

WHEN strftime('%Y', 'now') - strftime('%Y', u.birth\_date) BETWEEN 42 AND 56 THEN 'Gen X'

WHEN strftime('%Y', 'now') - strftime('%Y', u.birth\_date) BETWEEN 27 AND 41 THEN 'Millennials'

WHEN strftime('%Y', 'now') - strftime('%Y', u.birth\_date) <= 26 THEN 'Gen Z'

END AS Generation,

SUM(CAST(t.FINAL\_SALE AS FLOAT)) AS generation\_sales,

ROUND(SUM(CAST(t.FINAL\_SALE AS FLOAT)) \* 100.0 / (

SELECT

SUM(CAST(tr.FINAL\_SALE AS FLOAT))

FROM

transaction\_data\_sale\_no\_duplicates AS tr

JOIN

PRODUCTS\_TAKEHOME AS pr ON tr.BARCODE = pr.BARCODE -- Join transaction and product tables on barcode.

JOIN

USER\_TAKEHOME AS ur ON tr.USER\_ID = ur.ID -- Join transaction and user tables on user ID

WHERE

pr.CATEGORY\_1 = 'Health & Wellness' -- Filter for 'Health & Wellness' category products only.

), 2) AS percentage\_of\_sales

FROM

transaction\_data\_sale\_no\_duplicates AS t

JOIN

USER\_TAKEHOME AS u ON t.USER\_ID = u.ID

JOIN

PRODUCTS\_TAKEHOME AS p ON t.BARCODE = p.BARCODE

WHERE p.CATEGORY\_1 = 'Health & Wellness' GROUP BY Generation ORDER BY percentage\_of\_sales DESC;

Q4. Who are Fetch’s power users?

There are two ways to check power users and I have pasted results for both queries. In the first query I have defined power users as users who have 10 top users who have the highest sales value with fetch. In the second query I have defined power users as users who have 10 top users who have the most number of transactions with fetch

A screenshot of a computer program

Description automatically generated

Output:

A screenshot of a computer

Description automatically generated

Query:

-- Here I have defined power users as users who have 10 top users who have the highest sales value with fetch

-- Common Table Expression (CTE) to filter out transactions with a final sale value of zero

WITH transaction\_data\_sale\_not\_null AS (

SELECT

\*

FROM

TRANSACTION\_TAKEHOME

WHERE CAST(FINAL\_SALE AS FLOAT) != 0.00),

-- CTE to concatenate receipt\_id and barcode (primary key) and grouping them to remove duplicates

transaction\_data\_sale\_no\_duplicates AS (

SELECT

(RECEIPT\_ID || COALESCE(BARCODE, '0')) AS combined\_pk, -- Using a default value for NULL BARCODEs

RECEIPT\_ID,

BARCODE,

USER\_ID,

FINAL\_SALE,

FINAL\_QUANTITY

FROM

transaction\_data\_sale\_not\_null

GROUP BY

combined\_pk

)

-- Main SELECT query to calculate the total sales value per user

SELECT

t.USER\_ID,

SUM(CAST(t.FINAL\_SALE AS FLOAT)) AS total\_sales\_value -- Summing up the total sales value

FROM

transaction\_data\_sale\_no\_duplicates AS t

JOIN

PRODUCTS\_TAKEHOME AS p

ON

t.BARCODE = p.BARCODE

GROUP BY

t.USER\_ID

ORDER BY

total\_sales\_value DESC LIMIT 10; -- Only retrieving the top 10 users by total sales value

Here I have defined power users as users who have 10 top users who have the most number of transactions with fetch.

A screenshot of a computer program

Description automatically generated

Output:

A screenshot of a computer program

Description automatically generated

Query:

WITH transaction\_data\_sale\_not\_null AS (

SELECT

\*

FROM

TRANSACTION\_TAKEHOME

WHERE CAST(FINAL\_SALE AS FLOAT) != 0.00),

-- CTE to concatenate receipt\_id and barcode (primary key) and grouping them to remove duplicates

transaction\_data\_sale\_no\_duplicates AS (

SELECT

(RECEIPT\_ID || COALESCE(BARCODE, '0')) AS combined\_pk, -- Using a default value for NULL BARCODEs

RECEIPT\_ID,

BARCODE,

USER\_ID,

FINAL\_SALE,

FINAL\_QUANTITY

FROM

transaction\_data\_sale\_not\_null

GROUP BY

combined\_pk

)

-- Main SELECT query to calculate the total sales value per user

SELECT

t.USER\_ID,

count(t.RECEIPT\_ID) as total\_transactions -- Counting the number of transactions (receipts) per user

FROM

transaction\_data\_sale\_no\_duplicates AS t

JOIN

PRODUCTS\_TAKEHOME AS p

ON

t.BARCODE = p.BARCODE

GROUP BY

t.USER\_ID

ORDER BY

total\_transactions DESC LIMIT 10; -- Only retrieving the top 10 users by total sales value

Q5 Which is the leading brand in the Dips & Salsa category?

Here I have only considered sales which are > 0 and are unique, removing any double counting and have joined with the products table to get the brand. Category 2 had dips and salsa so I have used category 2 in the where clause.

A screenshot of a computer program

Description automatically generated

-- Initial CTE to filter out transactions with non-null final sales

WITH transaction\_data\_sale\_not\_null AS (

SELECT

\*

FROM

TRANSACTION\_TAKEHOME

WHERE

CAST(FINAL\_SALE AS FLOAT) != 0.00),

-- Second CTE to ensure distinct transactions by concatenating receipt\_id and barcode

transaction\_data\_sale\_no\_duplicates AS (

SELECT

(RECEIPT\_ID || COALESCE(BARCODE, '0')) AS combined\_pk, -- Concatenating with COALESCE to handle possible NULL values in BARCODE

RECEIPT\_ID,

BARCODE,

USER\_ID,

FINAL\_SALE,

FINAL\_QUANTITY

FROM

transaction\_data\_sale\_not\_null )

-- Main SELECT statement to analyze sales data, summing final sale values by brand

SELECT

p.BRAND,

SUM(t.FINAL\_SALE) AS TOTAL\_SALES\_VALUE,

SUM(t.FINAL\_QUANTITY) AS TOTAL\_QTY

FROM

transaction\_data\_sale\_no\_duplicates AS t

JOIN

PRODUCTS\_TAKEHOME AS p

ON

t.BARCODE = p.BARCODE

WHERE

p.CATEGORY\_2 LIKE '%Dips & Salsa%'

GROUP BY

p.BRAND ORDER BY TOTAL\_SALES\_VALUE DESC LIMIT 1;

Q6. At what percent has Fetch grown year over year? – **Answering based on YOY User Signup Growth**

Assumption: I have taken the users table to answer this question since the transactions table only had data for only 2024 year. Since we do not have complete data for 2024 yet (only until August), the number for 2024 can be misleading

A screenshot of a computer

Description automatically generated

-- Extracting year and users registered for each year

WITH Yearly\_USERS AS (

SELECT strftime('%Y', created\_date) AS year, COUNT(\*) AS users

FROM USER\_TAKEHOME

GROUP BY year),

-- using lead function to get the users for next year

YOY\_Growth\_CALC AS (

SELECT

year,

users,

LAG(users) OVER (ORDER BY year) AS last\_year\_users

FROM Yearly\_USERS)

--calculating users and growth

SELECT

year, users,

round(((users - last\_year\_users) \* 100.0 / last\_year\_users),2) AS YOY\_GROWTH

FROM YOY\_Growth\_CALC

WHERE last\_year\_users IS NOT NULL;