**Fetch Rewards Coding Exercise - Analytics Engineer**

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

Review the 3 sample data files provided below. Develop a simplified, structured, relational diagram to represent how you would model the data in a data warehouse. The diagram should show each table’s fields and the joinable keys. You can use pencil and paper, readme, or any digital drawing or diagramming tool with which you are familiar. If you can upload the text, image, or diagram into a git repository and we can read it, we will review it!

**ANSWER**:

In response to the 3 sample data files located in the directory file “json\_data” I have constructed a structured relational data model that articulates how I would store the data in a RBDMS database.

I chose a relational database model for its ability to enforce data integrity, ensure efficient joins across normalized tables, and support scalable analytical queries. This structured approach allows for robust data transformations and streamlined reporting

A screenshot of a computer screen

AI-generated content may be incorrect.

For the purposes of this exercise and enabling robust data analysis and engineering, I executed this database in a local SQLite database instance. In this repository you will find the .db file if you navigate to the sqlite\_db folder.

A quick and convenient way to explore this database and bounce the queries in later parts of this exercise is to install [DB Browser for SQLite](https://sqlitebrowser.org/) and navigate to the Execute SQL tab within the application.

**Second: Write queries that directly answer predetermined questions from a business stakeholder**

Write SQL queries against your new structured relational data model that answer at least two of the following bullet points below of your choosing. Commit them to the git repository along with the rest of the exercise.

Note: When creating your data model be mindful of the other requests being made by the business stakeholder. If you can capture more than two bullet points in your model while keeping it clean, efficient, and performant, that benefits you as well as your team.

* What are the top 5 brands by receipts scanned for most recent month?
* How does the ranking of the top 5 brands by receipts scanned for the recent month compare to the ranking for the previous month?
* When considering *average spend* from receipts with 'rewardsReceiptStatus’ of ‘Accepted’ or ‘Rejected’, which is greater?
* When considering *total number of items purchased* from receipts with 'rewardsReceiptStatus’ of ‘Accepted’ or ‘Rejected’, which is greater?
* Which brand has the most *spend* among users who were created within the past 6 months?
* Which brand has the most *transactions* among users who were created within the past 6 months?

**ANSWER:**

--What are the top 5 brands by receipts scanned for most recent month?

--How does the ranking of the top 5 brands by receipts scanned for the recent month compare to the ranking for the previous month?

WITH month\_ranked AS (

    SELECT

        strftime('%Y-%m', date\_scanned) AS purchase\_month,

        RANK() OVER (ORDER BY MAX(date\_scanned) DESC) AS month\_rank

    FROM silver\_receipts

    GROUP BY purchase\_month

),

brand\_ranking AS (

    SELECT

        sb.brand\_id,

        sb.name AS brand\_name,

        strftime('%Y-%m', sr.date\_scanned) AS purchase\_month,

        COUNT(sr.receipt\_id) AS total\_receipts,

        RANK() OVER (PARTITION BY strftime('%Y-%m', sr.date\_scanned) ORDER BY COUNT(sr.receipt\_id) DESC) AS rank

    FROM silver\_receipts sr

    JOIN silver\_receipt\_items sri ON sr.receipt\_id = sri.receipt\_id

    JOIN silver\_brands sb ON sri.barcode = sb.barcode

    GROUP BY sb.brand\_id, sb.name, purchase\_month

)

SELECT \*

FROM brand\_ranking

ORDER BY purchase\_month DESC, rank;

-- When considering average spend from receipts with 'rewardsReceiptStatus’ of ‘Accepted’ or ‘Rejected’, which is greater?

-- When considering total number of items purchased from receipts with 'rewardsReceiptStatus’ of ‘Accepted’ or ‘Rejected’, which is greater?

SELECT

    strftime('%Y-%m', sr.date\_scanned) as scanned\_month,

    sr.rewards\_receipt\_status,

    COALESCE(AVG(NULLIF(sr.total\_spent, 0)), 0) AS avg\_spent,

    COALESCE(SUM(sri.quantity\_purchased), 0) AS total\_items\_purchased

FROM silver\_receipts sr

LEFT JOIN silver\_receipt\_items sri ON sr.receipt\_id = sri.receipt\_id

GROUP BY strftime('%Y-%m', sr.date\_scanned),sr.rewards\_receipt\_status;

--Which brand has the most spend among users who were created within the past 6 months?

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

SELECT

    sb.brand\_id,

    sb.name AS brand\_name,

    COUNT(sr.receipt\_id) AS total\_transactions,

    SUM(sr.total\_spent) AS total\_spent

FROM silver\_users su

JOIN silver\_receipts sr ON su.user\_id = sr.user\_id

JOIN silver\_receipt\_items sri ON sr.receipt\_id = sri.receipt\_id

JOIN silver\_brands sb ON sri.barcode = sb.barcode

WHERE su.created\_date >= date('now', '-6 months')

GROUP BY sb.brand\_id, sb.name

ORDER BY total\_spent DESC, total\_transactions DESC

**Third: Evaluate Data Quality Issues in the Data Provided**

Using the programming language of your choice (SQL, Python, R, Bash, etc...) identify as many data quality issues as you can. We are not expecting a full blown review of all the data provided, but instead want to know how you explore and evaluate data of questionable provenance.

Commit your code and findings to the git repository along with the rest of the exercise.

**ANSWER:**

--null barcodes result in missing opportunities to tie to brands. Every receipt item barcode that does not tie to a brand in the brand database equates to missed information about a purchase decision

SELECT

    strftime('%Y-%m', sr.purchase\_date) AS purchase\_month,

    CASE

        WHEN sri.barcode IS NULL THEN 'is\_null'

        WHEN sri.barcode IS NOT NULL and sb.brand\_id IS NULL THEN 'barcode\_no\_brand'

        WHEN sri.barcode IS NOT NULL and sb.brand\_id IS NOT NULL THEN 'barcode\_w\_brand'

    END AS barcode\_status,

    COUNT(\*) AS total\_count

FROM silver\_receipts AS sr

LEFT JOIN silver\_receipt\_items AS sri ON sr.receipt\_id = sri.receipt\_id

LEFT JOIN silver\_brands AS sb ON sri.barcode = sb.barcode

GROUP BY

    strftime('%Y-%m', sr.purchase\_date),

    CASE

        WHEN sri.barcode IS NULL THEN 'is\_null'

        WHEN sri.barcode IS NOT NULL and sb.brand\_id IS NULL THEN 'barcode\_no\_brand'

        WHEN sri.barcode IS NOT NULL and sb.brand\_id IS NOT NULL THEN 'barcode\_w\_brand'

    END

ORDER BY purchase\_month DESC;

--There are 55.48% of scanned items missing barcodes, this is a missed opportunity

SELECT

    COUNT(\*) AS total\_receipt\_items,

    SUM(CASE WHEN sri.barcode IS NULL THEN 1 ELSE 0 END) AS missing\_barcodes,

    ROUND(SUM(CASE WHEN sri.barcode IS NULL THEN 1 ELSE 0 END) \* 100.0 / COUNT(\*), 2) AS percent\_missing

FROM silver\_receipt\_items sri;

--there appears to be a very low amount of receipt\_item barcodes that match the brand database. Is there a reason for this?

select DISTINCT

        sri.barcode,

        count(sb.brand\_code) as count

from silver\_receipt\_items sri

inner join silver\_brands sb on sri.barcode = sb.barcode

group by sri.barcode

order by count desc

--There are discrepancies between date\_scanned and purchase\_month on the receipts data. Wondering how these different dates are being gathered in the system?

select

    strftime('%Y-%m', sr.purchase\_date) AS purchase\_month,

    count(\*)

from silver\_receipts sr

group by strftime('%Y-%m', sr.purchase\_date)

order by strftime('%Y-%m', sr.purchase\_date) desc;

select

    strftime('%Y-%m', sr.date\_scanned) AS date\_scanned,

    count(\*)

from silver\_receipts sr

group by strftime('%Y-%m', sr.date\_scanned)

order by strftime('%Y-%m', sr.date\_scanned) desc

**Fourth: Communicate with Stakeholders**

Construct an email or slack message that is understandable to a product or business leader who isn’t familiar with your day to day work. This part of the exercise should show off how you communicate and reason about data with others. Commit your answers to the git repository along with the rest of your exercise.

* What questions do you have about the data?
* How did you discover the data quality issues?
* What do you need to know to resolve the data quality issues?
* What other information would you need to help you optimize the data assets you're trying to create?
* What performance and scaling concerns do you anticipate in production and how do you plan to address them?

**ANSWER:**

Hello [insert name of business leader here],

Firstly, thank you for sharing this data with me. I’ve spent time reviewing it and would like an opportunity to better understand how the data is being gathered and what controls are in place for that data. Would you be willing to meet with me or put me in contact with subject matter experts on these topics?

Questions I have are:

* What is the lifecycle of a receipt from the moment of being scanned to a final “resting” place in our data? What different timestamps are being recorded in the system and how are they recorded?
* How is the receipt data being scanned? What kind of controls are in place to ensure that the data ingested is clean?
* How are we gathering our barcode database that maintains the barcode associations? Are we doing any sort of Quality Assurance to make sure the relationship of barcode/brand is 1:1. Is there an automated effort in place to update these relationships and retire previous relationships in a historical table? I have identified that 55.48% of receipt items have missing barcodes, impacting our ability to track brand performance.
* Executing this workflow into production seems feasible from my initial analysis of the data provided. Concurrent read/writes on the database will be a concern. I will need more information on what scaling the platform offers and if we can find a way to scale our compute with the amount of traffic we are receiving from the application.

Appreciate your time on this matter and look forward to hearing back.

Best,

Nicholas Kreuziger