Fetch Rewards Coding Exercise - Analytics Engineer

In this exercise you will:

Demonstrate how you reason about data and how you communicate your understanding of a specific data set to others.

What are the requirements?

1. Review unstructured JSON data and diagram a new structured relational data model
2. Generate a query that answers a predetermined business question
3. Generate a query to capture data quality issues against the new structured relational data model
4. Write a short email or Slack message to the business stakeholder

Please let us know which SQL dialect you are using and include any code, notes, etc.. that helped you develop your answers. Showing your work can only help you!

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!

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?

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.

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?

The Data

Receipts Data Schema

Download receipts.json.gz

- **_id:** uuid for this receipt
- **bonusPointsEarned:** Number of bonus points that were awarded upon receipt completion
- **bonusPointsEarnedReason:** event that triggered bonus points
- **createDate:** The date that the event was created
- **dateScanned:** Date that the user scanned their receipt
- **finishedDate:** Date that the receipt finished processing
- **modifyDate:** The date the event was modified
- **pointsAwardedDate:** The date we awarded points for the transaction
- **pointsEarned:** The number of points earned for the receipt
- **purchaseDate:** the date of the purchase
- **purchasedItemCount:** Count of number of items on the receipt
- **rewardsReceiptItemList:** The items that were purchased on the receipt
- **rewardsReceiptStatus:** status of the receipt through receipt validation and processing
- **totalSpent:** The total amount on the receipt
- **userId:** string id back to the User collection for the user who scanned the receipt

Users Data Schema

Download users.json.gz

- **_id:** user Id
- **state:** state abbreviation
- **createdDate:** when the user created their account
- **lastLogin:** last time the user was recorded logging in to the app
- **role:** constant value set to 'CONSUMER'
- **active:** indicates if the user is active; only Fetch will de-activate an account with this flag

Brand Data Schema

Download brands.json.gz

- **_id:** brand uuid
- **barcode:** the barcode on the item
- **brandCode:** String that corresponds with the brand column in a partner product file
- **category:** The category name for which the brand sells products in
- **categoryCode:** The category code that references a BrandCategory
- **cpg:** reference to CPG collection
- **topBrand:** Boolean indicator for whether the brand should be featured as a 'top brand'
- **name:** Brand name

How do I submit my exercise?

Provide a link to a public repository (i.e., GitHub, Bitbucket) to your recruiter. Please do not send files directly via email.

FAQs

How will this exercise be evaluated?

An engineer will review the code and documentation you submit. At a minimum ER diagrams should be legible and SQL must be runnable. While your solution does not need to be fully production ready, you are being evaluated so put your best foot forward!

I have questions about the problem statement.

For any requirements not specified above, use your best judgement to determine expected result. You can elaborate on your decisions via the documentation you provide in your repo.

Can I provide a private repository?

If at all possible, we prefer a public repository because we do not know which engineer will be evaluating your submission. Providing a public repository ensures a speedy review of your submission. If you are still uncomfortable providing a public repository, you can work with your recruiter to provide access to the reviewing engineer.

How long do I have to complete the exercise?

There is no time limit for the exercise. Out of respect for your time, we designed this exercise with the intent that it should take you a few hours. But, please take as much time as you need to complete the work.