Fourth: Communicate with Stakeholders

Here is my slack message that encompasses the questions that I had in the following domains:

- 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 message would be directed to Data Entry/Analysts team, Software/Analytics Engineers.

Email:

Hi Team,

I am reaching out to seek clarity on a few key areas, including data quality issues, optimizing data assets, and addressing potential performance and scaling concerns.

Firstly, Talking about Data Quality Issues:

I have noticed some significant data quality issues during the exploratory data analysis, particularly in the process where users scan receipts containing branded products. I have reviewed the data related to Users, Brands, and Receipts and developed a series of quality checks to ensure the data is consistent, complete, valid, and reliable. However, during this process, I identified several concerning issues that I would like to highlight to better understand the causes and discuss potential solutions.

1) **Users Data:**

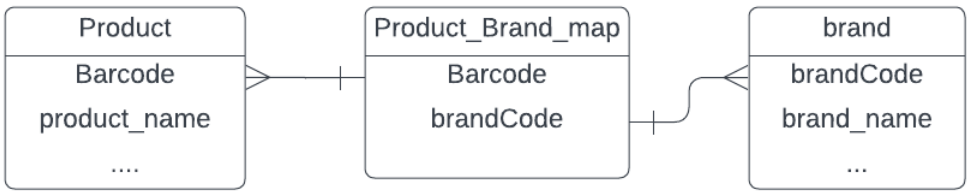
   Firstly, I observed a significant number of duplicate records, approximately **57.17%**. I want to confirm if I'm overlooking any potential reasons for this, as I'm almost certain that such a high volume of duplicates will negatively impact the downstream analysis. Could this be an error in the data entry process? Another issue I found in the table is related to the missing last login dates, which seems suspicious. Specifically, I identified **57** users whose last login is missing, yet they have made valid purchases. Initially, I assumed the missing login data was a technical glitch and that these users were likely inactive. However, after checking their purchase activity, I discovered that **57** of them had completed transactions. I believe this anomaly should be investigated further to ensure that the login date information is being properly captured. Another Data Quality Issue that I believe needs attention is the distribution of user creation dates. As you can see, several months are missing (for example, the years 2016, 2018, 2019, etc.), and the distribution seems abrupt. Could there be some details I'm missing, such as a decision to exclude data for accounts created during certain periods? This irregularity seems worth exploring.

```
Created counts by month:
 2014-12      1
2015-04       1
2017-07       1
2017-09       1
2017-12       1
2020-01       1
2020-07       1
2020-11       4
2020-12       1
2021-01     170
2021-02      30
```

2) Brands Data:

In the **Brands** table, I understand that '*brandCode*' could serve as a unique identifier for brands. Upon conducting some checks for unique identifiers, I've noticed some spelling and punctuation errors associated with the brandCodes, leading to inconsistencies. This made me curious about how the *brandCode* information is collected. Is it entered manually? If so, I believe there may be some potential for human error. We could explore ways to automate this process for better consistency. Additionally, I wanted to clarify the definition of 'TopBrand.' When do we classify a brand as 'Top'? If there's a standard formula, I believe we could create queries to automate this process, especially since I observed inconsistencies where the same brand is both categorized as a TopBrand and not.

I'm also curious about the role of the barcode in the Brands table. Since barcodes are typically product-specific, is there a way to extract brand information from the barcode? Additionally, having a separate product-brand mapping table, where each product (identified by its barcode) is linked to a unique brand identifier (like brandCode), would be highly useful. Something like this:



This would ensure that barcode and brand information are well-connected, helping us better analyze brand performance. If such a table or guidelines for developing one exist, it would be extremely beneficial for us to work with.

3) Receipts Table:

The linkage between the Receipts table and the Users table is established via user_id (the Foreign Key). During the Foreign Key integrity checks, I found that **45.3%** of the user_id present in the **Receipts** table are missing from the Users table. This raises some concerns about how often the Users table is updated. How does the refresh frequency of the Users table compare to the Receipts data? What methods are currently in place for refreshing the

Users table? Are there any pipeline bottlenecks that I might be unaware of? Could this even be an indicator of fraudulent transactions? What steps are currently taken to flag users as potential fraud? I believe maintaining foreign key integrity is crucial, as it ensures that only users who are registered with us are allowed to scan receipts.

Another primary concern is the lack of a proper way to connect the Receipts table and the Brands table, as I previously discussed. Both potential foreign keys: barcode and brandCode, are nearly **50%** missing. Could this be an issue with scanning? How does the team handle non-legible barcode information on receipts? Is there any scope for human intervention in these cases? Due to the high volume of missing values, we were unable to achieve 3NF compliance and had to stop at 2NF. It would be helpful if you could point me to any documentation that discusses possible imputation techniques for handling missing brandCodes.

Information needed to optimize the data assets:

One of the first bottlenecks I encountered while normalizing and writing data quality scripts was the importance of maintaining 3NF compliance. Updating the data, for example, to fix spelling errors, is much more optimized in 3NF compared to 2NF or 1NF. For example:

Scenario 1

column1	Category	column2	CategoryCode	column3
...	helth n well	...	HW	...
...	...	...		...
...	helth n well	...	HW	...
...	NaN	...	NaN	...

Scenario 2

CategoryCode	Category
HW	Health_Wellness
...	...
...	...
...	...

Consider two scenarios: Scenario 1 is 2NF compliant, and Scenario 2 is 3NF compliant. If a mistake is found in a column in Scenario 1, several rows across the table need to be updated, consuming both time and resources. In Scenario 2, with 3NF compliance, the update needs to happen only once in a single location. This shows the efficiency of maintaining 3NF in optimizing processes. However, due to a high volume of missing values, further normalization from 2NF to 3NF wasn't possible. Therefore, the information I would need is how to avoid huge missing values and potential imputers.

To better optimize the data flow into the databases while ensuring data quality checks, I would like to understand a few factors. For example, how frequently do the tables need to be updated, and what is the volume of the new data arriving? Additionally, what are the projected data growth trends? This information will help me design appropriate data quality pipelines.

For instance, in Apache Airflow DAGs handling large volumes of fast-flowing data, I could focus on performing checks only on the new incremental data. By partitioning the data in cloud storage (e.g., S3 or Hive) based on a timestamp, I can ensure that only the latest partitions undergo the necessary DAG processing, optimizing performance. On the other hand, smaller, less frequent data flows can be processed comprehensively without significant performance concerns.

Performance and Scaling Issues I anticipate in production:

One key concern in production is data volume growth. As the volume and frequency of incoming data increases, it will put pressure on storage and processing resources, potentially slowing down the data ingestion and pipeline execution. High data volumes, without proper partitioning or resource allocation, could lead to bottlenecks, especially during critical data quality checks. Another challenge involves scaling computer resources. If the current infrastructure relies heavily on vertical scaling, adding more power to a single machine could become inefficient as data grows. Lastly, pipeline failures may occur due to resource limitations, such as insufficient memory or CPU, leading to incomplete tasks and delays in downstream processes.

To address these concerns, horizontal scaling using technologies like Kubernetes can distribute the workload across multiple machines, reducing pressure on a single system. Airflow can be configured with Kubernetes to dynamically scale workers based on real-time data loads, which helps manage increasing data volume efficiently. Partitioning datasets based on timestamps or relevant criteria also optimizes query performance, reducing the strain on resources by processing only the required data. This approach ensures that the system adapts to both increasing data volume and processing complexity.

Additionally, implementing CI/CD pipelines ensures smooth and reliable updates in production. Continuous integration enables automated testing of pipeline changes and validations of data checks, ensuring that scaling adjustments or new features are integrated without disrupting current operations. CI/CD also supports automated deployment, making the data pipeline more robust and reducing the likelihood of failure during production scaling.

Looking forward to your insights and suggestions on these points. Please feel free to share any further details that could help the team.

Best,
Amarthya