**Fetch Rewards Coding Exercise - Analytics Engineer**

**Task 1:**

Since, the data has to be loaded into data warehouse, I created stage tables and data warehouse table for each of the dataset. For users and brands dimension tables are created and receipts is created as fact table which holds all the analytics related data.

I followed a star schema to create data warehouse tables and dividing into dimensions and fact table as required. The stage tables were loaded with data parsed from the JSON dataset without any transformation to avoid any data loss during the parsing and loading the data. Once, the stage table is loaded with the data, I created the data warehouse table and loaded the data from stage table to corresponding DW table by performing the required transformation and including the correct datatypes for the corresponding columns to make sure there is no overhead of memory or space in the data warehouse tables. The entire data warehouse loading process included the creation of stage table, loading the stage table with JSON dataset, performing the transformation on the stage table before finally loading into DW table. Later on, the stage tables were truncated to have the loading process repeated again for the next set of datasets.

ER model has been attached along with other files.

**Handling of early arriving facts:**

**Missing users and brands records:**

For the dimUsers and dimBrands table, there were certain records available in the fact receipts table which were missing in the users and Brands tables. I inserted those missing users and brands records accordingly into the dimusers and dimBrands table to have the data consistency.

**Duplicate brands data:**

Fixed the duplicate issue - with no information available I randomly picked a record with duplicate barcode and deleted it.

**Task 2:**

SQL query to pull related information for the predeterimined questions has been added as Task2\_SQL Query file.

**Task 3:**

**Data Quality issues (observed):**

1. Special character in the name of the key - $
2. Duplicate data - users with same (users data)
3. All the users objects need to be part of an parent level array (Child-parent level closing parenthesis missing)
4. Each user objects need to be separated by comma (,)
5. Ambiguous column names (Category code having some description values instead of code)
6. Data redundancy in certain columns (Category & Category code) with same data
7. Early arrived facts records

**Task 4: (Communications with stakeholders)**

Hi [Name],

I would like to thank you for providing the data in the required format for data loading and creating insights from the data to answer a few pre-determined questions from a business perspective. The loading and other data analysis have been completed and I am attaching the required files for your reference.

I would also like to bring few points for your consideration on the data sources and quality of data to improve our existing approach and make the entire process very effective in the future. It would also help us to be more productive and intuitive about the data and providing better analysis results in the production and make the entire approach scalable in the future based on the business needs.

Please refer to the below points for your reference which include questions on data, quality issues faced with the current set of datasets, approaches followed to resolve data quality issues, performance and scalability concerns in the productions, and plan to address these issues. Please let me know in case if you have any questions on any of the points:

* **What questions do you have about the data?**

1. What is the source of data gatherings?
2. What is the final outcome that is being expected from the data?
3. How the data quality is being ensured? Is there any expected set of instructions for each type of dataset?
4. How often this data change? It will help while designing a data pipeline and generating reports from a data warehouse?
5. Who are the final users of these analysis results and what are the main business insights are expected from these data?

* **How did you discover the data quality issues?**

1. By identifying the error and issues while trying to load the data into SQL Server and during data pre-processing
2. Understanding of JSON object formatting
3. Understanding the data and its requirement

* **What do you need to know to resolve the data quality issues?**

1. The correct JSON object format and correct structure
2. Nested object information contains the list of records that may have to be linked with correct columns in the other table.
3. JSON Object structure
4. Correct schema format

* **What other information would you need to help you optimize the data assets you're trying to create?**

1. The correct field mapping documents.
2. Data source and its frequency.
3. Higher volume of the data to create better insights and reports.

* **What performance and scaling concerns do you anticipate in production and how do you plan to address them?**

1. **Serialization and deserialization larger vs smaller JSON Objects-** In the case of a large object array, it is better to break up the large array into smaller prices and this optimization logic helps to reduce the server CPU usage at the production level.
2. **Longer fields name-** In case if the JSON objects have larger field names, it is always better to make fields name smaller and it can give at least a 10% parsing performance boost.
3. **Sparse columns –** In case if the columns in the table have a very high number of NULL values, the performance gets hit while querying and result in slower processing. Instead of Sparse columns, the schema can be defined to have a sparse table where NULL values spread across different columns in the table rather than only at one column.
4. Since JSON objects are stored by default as NVARCHAR(MAX). It is always good to define own schema with valid datatypes to avoid space and memory issue in the database.
5. I have created a surrogate to minimize the storage space. Further, we can create a column store index for optimal analytical query performance.
6. Create an additional index for better performance.
7. Based on the volume of the data, we can partition the table either by daily, monthly, or yearly. If the data is volume is huge, we can consider building a data warehouse on a distributed system like Azure Synapse.

**Scalability options in the Production environment:**

1. Based on the volume of the data arriving, we can consider ingesting data in near real-time by using technology like Event Hubs as message queue and Azure stream analytics or Spark Structured streaming to process the data in near real-time.
2. For data ingestion, we can automate the process on-premise by using technology like SSIS or Informatica or in Azure technology like Azure data factory for automated data ingestion instead of manually running the scripts. This pipeline will also have fault-tolerance built into it so that in case of any failure it can safely start from its point of failure.