CS 378 Project 3, due Thursday, 09/26.

Recall our acceptance criteria introduced in Project 2, whose relevant sections are copied below for ease of reference. This project assumes that your data satisfies anomaly types 5, 6, and 7. Observe that all three types concern issues with the data at the field level. This is an important point to keep in mind throughout this project as we are not attempting to remodel the entities at this stage, only the fields within the entities as they exist in the raw layer.

The high-level goals for this project are to construct a staging layer in BigQuery that fixes several field-level anomalies, including the three types mentioned below.

Anomaly Type	Description	Applies to Project	Air Travel Examples
5	There exists a field in any table of the warehouse whose assigned data type does not best fit its domain of values.	3	airports.timezone stores a numeric value as a string. tsa_traffic.date is stored as a string instead of date
6	There exists a field in any table of the warehouse whose null values are represented as empty strings, "\n" or something similar.	3	source_airport_id in the flight_routes table
7	There exists a field in any table of the warehouse that stores the values of multiple attributes in a single cell. The values represent different attributes.	3	flight_delays.airport_name is composed of city, state, and airport. All three attributes are stored in the same column

Data Transformation Strategies

- Create a staging area in BigQuery and populate it with the results from your data transformations. The transformations should be applied to the raw tables and materialized to the staging dataset. **Do NOT mutate the raw layer.** All mutations for this project should be applied to the staging layer only.
- Implement transformations in SQL and Python that resolve at least one anomaly type 5, 6, and 7 in the staging layer:
    - Resolve anomaly type 5 by casting the field in question to a more suitable type (e.g. integer instead of double, datetime instead of string, etc.).
    - Resolve anomaly type 6 by replacing empty strings or "\n", etc. with proper null values. You should also take this opportunity to scrub others fields that have unwanted values. For example, ", $, %, etc. This is not an exhaustive list, please use best judgment to decide what an unwanted character means within the context of your dataset.

    - Resolve anomaly type 7 by splitting the values of multiple attributes which are embedded into a single cell. You want to split them into their own fields within the same table.
- In addition to these three anomaly types, you should also perform these transformations:
    - Rename any non-descriptive columns in your staging tables.
    - Exclude any fields which don't store useful data from your staging tables.
    - Standardize any inconsistent names and/or categories. Leverage the LLM to resolve the standardizing issues (see code samples for details).

Implementation Guidelines

- Create a new Colab notebook for your data transformations. Name this notebook `3-[your-domain]-stg.ipynb`.
- Annotate your notebook with section headers and short Markdown comments to improve its readability.
- The staging tables should be stored in their own dataset in BigQuery. The name of the dataset should follow the convention of `[your_domain]_stg` where stg is short for staging. For example, air_travel_stg.
- When transforming the data within a field, be sure to materialize only the output from your transformations (i.e. don't carry over both the transformed values and original values to the staging area).
- If you need to split up a complex transformation into multiple steps, you can materialize the intermediate results as a temporary table. Be sure to create the temp table in the staging area as well and drop it right after you have created the final staging table.
- Make sure of the BigQuery built-in functions (e.g. string functions and cast functions) when converting fields from one type to another.
- When working with the LLM, employ these strategies: 1) inject sufficient context into the prompt to reduce hallucinations; and 2) send the input records in batches of a few hundred rows to avoid truncated results.
- Table names should remain in lowercase and it's best to use an underscore between multiple words. For example, airport_businesses instead of airport-businesses. The reason for this is because hyphens in dataset and table names require enclosing the name with single quotes.
- Column names should remain in lowercase across the board.
- Both table and column names should be descriptive. If the original names are not descriptive, rename them while creating the staging table.
- Don't forget to carry over the `_data_source` and `_load_time` fields into the staging area!
- Create an ERD that reflects the design of your staging tables. Bold all the fields which have changed from the raw layer. You do not need to prepare a data dictionary.
- Create a `project3` folder and add your artifacts to it (i.e. `3-[your-domain]-stg.ipynb` and `[your-domain]-erd-stg.pdf`).
- Create a standard submission.json file and upload it to Canvas by the submission deadline.

CS 378 Project 3 Rubric
Due Date: 09/26/24

`3-[your-domain]-stg.ipynb` is thorough and meets all requirements **-10** incorrectly used SQL commands and/or Python code **-10** incorrect data type conversion (anomaly type 5) **-10** incorrect replacement of empty strings and other unwanted characters (anomaly type 6) **-10** incorrect field splitting into separate fields (anomaly type 7) **-5** at least one staging table contains non-descriptive columns **-7** at least one staging table contains inconsistent name or category values **-7** staging dataset, tables or fields don't follow our naming convention **-5** notebook lacks Markdown annotations and is hard to follow **-30** did not create staging tables or staging tables missing in BQ dataset **-80** missing file	80
`[your-domain]-erd-stg.pdf` accurately depicts staging table schema and logical relationships between tables **-2** for file named incorrectly **-3** for each missing staging table **-3** for each missing relationship **-10** ERD not aligned with staging tables **-20** missing file	20

`submission.json` submitted into Canvas. Your project **will not** be graded without this submission. The file should have the following schema:
 	
`{`
`  "commit-id": "your most recent commit ID from Github",`
`  "project-id": "your project ID from GCP"`
`}`
 	
Example:
 	
`{`
`  "commit-id": "dab96492ac7d906368ac9c7a17cb0dbd670923d9",`
`  "project-id": "some-project-id"`
`}`
	Required

Total Credit:	**100**