CS 329E Project 3, due Thursday, 02/13.

This project assumes that your data satisfies the validation criteria which we introduced in Project 2. If this is not the case and you have not received a signed off from Prof. Cohen, please consult with the teaching staff before starting on this assignment.

The goals of this project are to address the data anomalies described in criteria 5, 6, and 7. These criteria are reproduced below for convenience:

Criteria	Description	Project Applicability	Air Travel Examples
5	There exists a table in the raw layer of the warehouse whose **assigned data types do not best fit its domain of values**.	Project 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 table in the raw layer of the warehouse whose **null values are represented as empty strings, "\n" or something similar**.	Project 3	`source_airport_id` in the `flight_routes` table and `icao_code` in the `aircrafts` table store the value "\N" for null
7	There exists a table in the raw layer of the warehouse that **stores the values of multiple attributes into a single field**. These values represent distinct attributes, but they are packed into a single field.	Project 3	The field `flight_delays.airport_name` contains a city, state, and airport. All three values are stored in the same field

Observe that all three types of data anomalies are concerned with field-level inconsistencies. This is an important point to keep in mind when it comes to the scope of this assignment as we are **not** looking to remodel the entities at this stage, only the fields within these entities as they exist in the raw layer of our warehouse.

Methodology

- 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, there are some additional transformations you should make to your staging tables if applicable:
    - Rename any non-descriptive fields
    - Remove (or exclude) any fields which don't store any useful data
    - If your tables contain any category or name fields and the values of those fields are inconsistent, you should standardize them. To do that, you need to assess the scale of the problem. If it's just a few types of inconsistencies, you can hard code the update statements. If you have more than a few types of inconsistencies, you should leverage ML. In our case, we will prompt the LLM by giving it the inputs from our tables and asking it to map them to one or more outputs.

Implementation Plan

- Create a new Colab notebook for your data transformations. Name this notebook `3-[your-domain]-stg.ipynb`.
- Create the staging tables in a new 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.
- For each data transformation, materialize only its output (i.e. don't carry over both the transformed values and original values into the staging area).
- If you need to split up a complex data transformation into multiple steps, you should materialize the intermediate results into a temporary table. Be sure to create the temp table in the staging area as well and drop it after you have created the final staging table.
- Make use of BigQuery's built-in functions (e.g. string functions and cast functions), especially when converting fields from one type to another.
- When working with the LLM, employ these strategies: 1) include sufficient context into the prompt to reduce hallucinations; and 2) send the input records in small batches to avoid truncated results.
- Don't forget to carry over the `_data_source` and `_load_time` fields into the staging area!

- The table names in the staging layer should remain in lowercase. Please use underscores between words rather than hyphens (e.g. `airport_businesses` instead of `airport-businesses`). The reason for this convention is because BQ requires you to escape table names with hyphens by enclosing them in quotes and the DBT BQ connector (which we will be using in the second half of the term) does not support hyphens at all.
- Column names should remain in lowercase across the board. Again, use underscores instead of hyphens between words.
- Both table and column names should be descriptive. If the original names are not descriptive, rename them while creating the staging table.
- Annotate your notebook with section headers and short Markdown comments. This not only helps others understand your logic, but it may help you remember it in a few weeks!
- 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 329E Project 3 Rubric
Due Date: 02/13/25

`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 (criteria 5) **-10** incorrect replacement of empty strings or other unwanted characters (criteria 6) **-10** incorrect field splitting into separate fields (criteria 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 **-5** did not follow naming convention for BQ dataset, tables or columns **-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**