CS 378 Project 4, due Thursday, **02/27**.

The goals of this project are to address the data anomalies described in criteria 8, 9, and 10. These criteria are reproduced below for convenience:

Criteria	Description	Project Applicability	Air Travel Examples
8	There exists a table in the raw layer of the warehouse that stores a **list of elements in a cell**. In contrast to criteria 7, these elements represent multiple values for the same attribute.	Project 4	`flight_routes` sometimes stores a list of equipments in the same cell, `airport_businesses` sometimes stores a list of menu items in the same cell.
9	There exists two tables in the raw layer of the warehouse which originated from different sources and which have similar data. These tables use **two different identifier systems to refer to the same entity**.	Project 4	airport information is repeated across multiple tables in a non-standard way. See for example `airportRef` and `aiportIdent` versus `airport_id` and `airport_code`.
10	There exists a table in the raw layer of the warehouse that models **more than one logical entity in the same table**. This leads to data redundancy and storing repeated values.	Project 4	The `flight_delays` table has information about airports, airlines, and flight delays. Fields like `carrier_name` and `airport_name` shouldn't exist in the table because they are redundant.

Observe that all three criteria are concerned with the design of the entities in the warehouse and resolving the data anomalies may lead to widespread changes, including merging and splitting entities. This is in contrast to the more localized changes we made in the previous assignment when we addressed the anomalies exhibited in criteria 4, 5, and 6.

Methodology

- Create an intermediate area in BigQuery and populate it with the results from your transformations. The transformations should be applied to the staging tables and

materialized in the intermediate dataset. **Do NOT mutate the staging area**. All mutations should be applied only to the intermediate area of the warehouse.

- Implement transformations in SQL and Python that resolve at least one anomaly type 8, 9, and 10 in the intermediate layer:
    - Resolve anomaly type 8 by converting the list of embedded values to either an array type in BQ within the same table or by flattening the values into an auxiliary table with proper references to the main table.
    - Resolve anomaly type 9 by creating a universal identifier for the shared entity and ensuring that dependent tables are updated to reference the new identifier.
    - Resolve anomaly type 10 by decomposing the multi-entity table into separate tables. To determine the design of these tables, think about the relationship between them.
- In addition to the three anomaly types, you should also perform these transformations:
    - Enrich at least one entity with a new attribute using the LLM.
    - Ensure that referential integrity holds on all the final tables in this layer. This may require you to remove duplicate records, look up missing values, delete orphan records, and filter out unwanted records.

Implementation Plan

- Create a Colab notebook that implements the intermediate layer with the appropriate transformations. Name your notebook `4-[your-domain]-int.ipynb`.
- Create the staging tables in a new dataset in BigQuery. The name of the dataset should follow the convention of `[your_domain]_int` where int is short for intermediate. For example, air_travel_int.
- Copy all the tables from the staging area into the intermediate area, even those which don't exhibit any anomalies.
- If you need to split up a complex transformation, you can materialize the intermediate results as a temporary table. When you create the temp table, place it in the intermediate area and drop it after you have finished creating the final table.
- Each table should have a well-defined primary key and child tables should also have proper foreign keys. Create the key constraints both in code as well as in your ERD.
- Reorder the table columns such that the primary key fields appear at the top.
- The first letter of the intermediate table names should be in uppercase. If the name has multiple words, capitalize the first letter of each word and use an underscore between the words. The only exception to this rule are the temporary tables. These should be all in lowercase letters and contain the prefix `tmp_`.
- Most intermediate table names should be in singular form except for junction tables, which can be either in singular or plural. For example, in the Air Travel warehouse, the Airport_Businesses table is a junction table because it models a many-to-many relationship.
- Column names should remain in lowercase across the board.

- Carry over the two system columns, `_data_source` and `_load_time`, from the staging tables into their respective intermediate tables. For each intermediate table that is derived from more than one data source, update its `_data_source` value to be the union of its data source names and store it as a string array type. For example, `["openflight", "transtats"]`.
- Create an ERD that reflects the design of your final intermediate tables. Bold all the fields, data types, and entities which have changed since the staging layer. Denote the keys for each entity and draw the proper relationships between the entities. Those relationships should have solid line edges. You do not need to prepare a data dictionary.
- Create a new folder in your repo and name it `project4`. Store all your artifacts for this project into the `project4` folder.
- Create a submission.json file and upload it to Canvas by the deadline. Only one person per group needs to do this step.

CS 378 Project 4 Rubric
Due Date: 02/27/25

`4-[your-domain]-int.ipynb` is thorough and meets all requirements **-10** incorrectly used SQL commands and/or Python code **-15** missing or incorrect resolution of anomaly type 8 **-15** missing or incorrect resolution of anomaly type 9 **-15** missing or incorrect resolution of anomaly type 10 **-5** missing or incorrect data enrichment **-5** missing `_data_source` and `_load_time` fields **-7** missing primary key constraints and/or logical checks **-7** missing foreign key constraints and/or logical checks **-5** notebook lacks Markdown annotations and is hard to follow **-5** did not follow naming convention for dataset, tables or columns **-20** empty intermediate tables in BQ dataset **-30** did not create intermediate tables in BQ dataset **-80** missing file	80
`[your-domain]-erd-int.pdf` accurately depicts the intermediate tables schema and logical relationships between tables **-2** for file named incorrectly **-3** for each missing intermediate table **-3** for each missing relationship or key (primary or foreign) **-2** did not bold the changed fields, types, or entities to this layer **-10** ERD not aligned with intermediate 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**