The first implementation hint for this project is out of date. More details below in the comment. Be sure to update before Fall 2025

CS 378 Project 2, due Thursday, **02/06**.

### Part 1: Goals

This project has three main goals:

- extract some structured data of interest from your text, pdf or image dataset with Gemini
- load your structured data files (csv, json) from GCS to BQ
- analyze your data against a set of validation criteria to ensure that your data collection is sufficiently diverse

### Part 2: Code Samples

The `project2` folder of the snippets repo has code samples and other artifacts that are relevant for this project.

Note: You should probably not run the two extraction notebooks (`1-air-travel-extract-*.ipynb`) because they take hours to complete. However, you can run the data load notebook (`2-air-travel-load.ipynb`) in your own project. This will load the air travel data to your BQ instance and can help you follow along in future projects.

### Part 3: Validation Criteria

Your data collection needs to meet certain criteria for subsequent projects in this course to make sense and have sufficient scope. The first 4 criteria are common to many data warehousing environments. The last 6 criteria are a series of data anomaly types, which occur frequently . You want your data collection as a whole to suffer from all 10 anomaly types. This means that you have sufficiently messy data to work with, which is what we are aiming for.

Please review the list of criteria below and evaluate which elements of your data collection satisfy each one. Note that criteria 5-10 represent different anomaly types that should be present in the data.

| Criteria | Description | Project Applicability | Air Travel Examples |
|---|---|---|---|
| 1 | The warehouse data must be made up of multiple data sources such that there are **at** | All projects | Airport Guide, Open Flights, BTS, TSA, etc. |

| | | | |
|---|---|---|---|
| | **least 4 sources that are independently produced**. | | |
| 2 | The warehouse data must be composed of **at least one source whose type is unstructured**. This can be text, pdf, or images. | All projects | The `airport_businesses` and `tsa_traffic` datasets were both created from pdf files. |
| 3 | The warehouse data must be composed of **multiple logical entities**. | All projects | Airports, Airlines, Airport Businesses, Airport Reviews, Flights, Routes, Countries, Aircraft |
| 4 | The warehouse data must be consistent such that its **functional dependencies hold** within the records of a table and across records. | All projects | The values of an airport record need to be consistent. For example, if I have an airport named JFK, its location should be NY. This is mostly to guard against synthetic data that is randomly generated. |
| 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 |
| 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. |

What should you do if your warehouse does not meet one or more of these criteria?

- Think of ways to broaden your data domain and start looking for additional related datasets.
- Consider choosing a different domain that has more available data. If you go down this route, please note that you'll need to redo most of Project 1 in a relatively short timespan.
- Given that we have two weeks to complete Project 2, I expect your data collection to satisfy all 10 criteria, including the six anomaly types. If you have attempted to find additional data and are still missing some criteria, please be sure to go to Professor Cohen's office hours and get sign off.

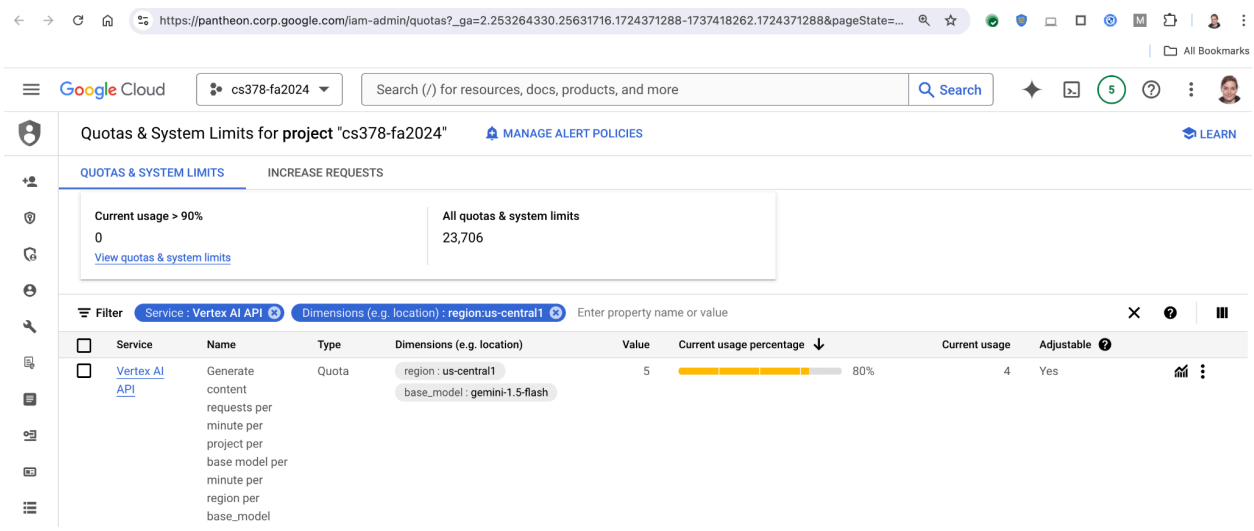
**Part 4: Implementation Plan**

- Create a new folder in your repo and name it `project2`. Store all of your artifacts for this project in the `project2` folder.
- Develop a Colab notebook that extracts some interesting data from your unstructured dataset and save the results as csv or json files stored in your bucket on GCS. Name your notebook `1-[your-domain]-extract-[data-source].ipynb`. If you are using more than one unstructured dataset, you will create a notebook for each.
- Develop a Colab notebook that loads the data files into BQ. Load each file into its own table in the raw area. The only exception to this is if you have a collection of files which represent the same type of data and are split into multiple files by date. In that case, you want to load all the files into the same table. Name your notebook `2-[your-domain]-load.ipynb`.

- Annotate your notebooks with section headers and short Markdown comments to improve their readability.
- Store your BQ tables in a raw dataset. The dataset name should follow the naming convention of `[your-domain]_raw`.
- When creating the BQ tables, add two new columns to the end of each table as follows:
    - `_data_source (STRING)`: should default to the name of the data source. Choose a descriptive name to identify each data source (e.g. "openflights").
    - `_load_time (TIMESTAMP)`: should default to the current timestamp and represent the time in which the records were loaded into the table.
- Choose descriptive table names. Note that the name of your table can be different from the name of the file from which it is sourced.
- Lowercase both the table and column names. Use underscores (instead of hyphens or camel case) to name a table name or column name with multiple words in its name.
- Update your data dictionary from Project 1 with the work that you've done in this project. For example, update the attribute list and any other details which have changed.
- Update your ERD with the work you've done in this project. You do not need to add the `_data_source` and `_load_time` fields to the diagram as those fields are understood and would only end up cluttering the diagram.
- Review the validation criteria in Part 3 and ensure that your data collection as a whole meets the 10 criteria. In a Markdown file, document how your data satisfies each one using a specific example. If your data is missing one or more criteria or if you're unsure if it satisfies a criteria, make note of that and speak to the Professor or TAs. Name your file `criteria-analysis.md`.
- Publish to your repo: `1-[your-domain]-extract-[data-source].ipynb`, `2-[your-domain]-load.ipynb`, `[your-domain]-data-dict-v2.xlsx`, and `[your-domain]-erd-v2.pdf`, and `criteria-analysis.md`. Remember that all artifacts need to go into your `project2` folder.
- Create a submission.json file and upload it to Canvas by the deadline. Only one person per group needs to do this step.

**Part 5: Implementation Hints**

- Once you start working with Gemini, you will probably hit a quota limit. The quota is set ridiculously low by default (e.g. 5 requests / minute). You can apply for a quota increase from here. You should ask for 200 requests / minute. The request should get approved automatically within 5 minutes unless you are on the free trial. If on the free trial, you should switch billing accounts before requesting the quota increase. Please speak to the Professor if you face any issues.




- If you get the error `[Errno 2] No such file or directory` when trying to download a file from the bucket or write a file that will be uploaded to the bucket. This is happening because the local file system on the Colab VM does not have a matching directory path. To fix the problem, open the terminal in Colab Enterprise and create a folder structure for storing the CSV files using the `mkdir` command. Then get the complete path with the `pwd` command. For example: `/contents/out-csv`. Be sure to update the folder variables in your code so that they match this path. Note: the paths on the local file system do not need to be the same as the paths in your bucket.

- When loading the data into BQ, you may need to relax the table schema constraints if you run into problems. For example, if you have defined a field as mandatory, you may need to redefine it as nullable. If you have defined a field as a DATE type, you may need to redefine it as a STRING. The goal here is to get the data into BQ and massage it later. This is known as an “ELT” approach (as opposed to “ETL”).

**Grading Rubric**

Due Date: 02/06/25

| | |
|---|---|
| `1-[your-domain]-extract-[data-source].ipynb` correctly written and indicates successful extraction<br>**-5** for each missing output<br>**-10** did not extract structured data with LLM<br>**-10** did not store output from extraction in csv or json<br>**-15** unable to verify the output from extraction process with GCS commands<br>**-30** missing file<br>**-20** didn't run the code | 30 |
| `2-[your-domain]-load.ipynb` correctly written and indicates successful table creation<br>**-5** for each missing output<br>**-5** column names and/or table names not lowercase<br>**-5** did not include data_source or load_time fields in tables<br>**-5** did not update variable values (project_id, bucket_name, etc)<br>**-15** unable to verify loads through bigquery commands<br>**-30** missing file<br>**-20** didn't run the code | 30 |
| `criteria-analysis.md` is thorough and has all information. All 10 criteria, including the six anomaly types should be met unless you have received a sign off from the Professor.<br>**-5** for each criteria explanation not thorough<br>**-5** if crucial misunderstanding of each criteria<br>**-30** missing file | 30 |
| `[your-domain]-data-dict-v2.xlsx` and `[your-domain]-erd-v2.pdf`<br>**-5** for each missing important link in ERD<br>**-5** data dictionary does not contain essential information of a table<br>**-5** ERD not aligned with data dictionary columns<br>**-10** missing ERD or data dictionary | 10 |
| `submission.json` submitted into Canvas. Your project **will not** be graded without this submission. The file should have the following schema:<br><br>`{`<br>`  "commit-id": "your most recent commit ID from Github",`<br>`  "project-id": "your project ID from GCP"`<br>`}`<br><br>Example:<br><br>`{`<br>`  "commit-id": "dab96492ac7d906368ac9c7a17cb0dbd670923d9",`<br>`  "project-id": "some-project-id"`<br>`}` | Required |

| | |
|---|---|
| **Total Credit:** | **100** |