Evaluation Project for stage one

Task Instructions and Step-by-Step Process:

Follow these steps to complete the project:

Step 1: Data Acquisition and Understanding

- Go to the [Kaggle Olympics History Dataset] page.
- Download the dataset files (e.g., `athlete_events.csv`, `noc_regions.csv`). You might need a free Kaggle account.
- Spend time exploring the CSV files. Understand the columns, data types, and the relationships between the files (e.g., how athlete information links to event results).
- *Think:* How should this data be structured in a relational database? Which columns will be primary keys? Which will be foreign keys? How can you avoid data redundancy?

Step 2: Database Design (ERD & Schema)

- Based on your understanding, design a relational database schema. Identify the necessary tables (e.g., `athletes`, `events`, `games`, `countries`, `results`).
- Create an Entity-Relationship Diagram (ERD) to visually represent your database design, showing tables, columns, primary keys, and relationships (one-to-one, one-to-many, many-to-many). You can draw this manually, use an online tool (like draw.io), or a database-specific tool.
- Define the data types for each column in your schema (e.g., `INTEGER`, `VARCHAR`, `REAL`, `DATE`).
- Consider normalization principles (up to 3NF) to ensure your design is efficient and avoids anomalies.

Step 3: Database Setup

- Choose your database system (SQLite is recommended for ease of setup).
- If using SQLite, decide on a name for your database file (e.g., `olympics.db`).
- Write the SQL `CREATE TABLE` statements based on your schema design from Step 2. Include primary key and foreign key constraints.
- Execute the `CREATE TABLE` statements using your chosen database tool (like a SQLite browser, pgAdmin, MySQL Workbench) or a Python script using the appropriate database library to create the database and tables.

Step 4: Data Loading and Initial Cleaning (Python)

- Use Python (with the `pandas` library) to read the downloaded CSV files into DataFrames.
- Perform necessary data cleaning and initial transformations in Python:
  - Handle missing values (e.g., in 'Age', 'Medal'). Decide on a strategy (imputation, dropping rows).
  - Convert data types if necessary (e.g., ensure numerical columns are treated as numbers).
  - Standardize categorical data if needed (e.g., consistent country names).
  - Prepare the data structure to match your database schema. This might involve splitting data from one CSV into multiple DataFrames that correspond to your database tables.

- Use a Python database library (`sqlite3` for SQLite, `psycopg2` for PostgreSQL, `mysql.connector` for MySQL) to connect to your database.
- Write Python code to iterate through your cleaned DataFrames and insert the data into the corresponding tables in your database. Use appropriate methods for bulk insertion if available (e.g., `df.to_sql()` in pandas for SQLite). Be mindful of foreign key constraints - you might need to load parent tables before child tables.

Step 5: Data Analysis using SQL (via Python)

- Formulate at least five analytical questions about the Olympics data that require querying your database. Examples:
  - Which country won the most gold medals in the Winter Olympics?
  - What is the distribution of athlete ages in different sports?
  - List the top 10 athletes with the most total medals.
  - Which cities have hosted both the Summer and Winter Olympics?
  - Analyze the trend of athlete participation over the years.
- Write SQL queries to answer these questions. Ensure your queries utilize JOINS, Aggregate Functions, GROUP BY, and potentially subqueries or window functions as appropriate.
- Use Python to connect to your database.
- Execute your SQL queries from within your Python script or notebook using the database connector library.
- Retrieve the results of your queries into Python (e.g., into pandas DataFrames for further manipulation or display).

Step 6: Python Data Analysis & Interpretation

- Use Python (pandas) to perform further analysis on the data retrieved from your database queries.
- Calculate descriptive statistics, perform aggregations, and identify trends or patterns in the results.
- Optionally, use a Python visualization library (like `matplotlib` or `seaborn`) to create charts or graphs based on your query results.
- Interpret your findings. What insights did you gain from the data based on your analysis?

Step 7: Documentation and Presentation

- Document your entire project process clearly. This can be in a report, a detailed README file in a GitHub repository, or a series of notes. Include:
  - Your database ERD and final schema (SQL `CREATE TABLE` statements).
  - A description of the data cleaning and transformation steps performed in Python.
  - The analytical questions you asked and the corresponding SQL queries used to answer them.
  - The Python code used for data loading, database interaction, and executing queries.
  - A summary of the key findings and insights you derived from the SQL and Python analysis.
- Prepare a brief presentation (e.g., 5-10 slides or a short report) summarizing your project. This should cover your database design, the analytical questions, key findings (perhaps with visualizations), and what you learned. This artifact is crucial for your portfolio.