Part 1: Python

Create an API using Python, preferably with FastAPI. This API should have two endpoints, each returning JSON data for the following tables:

1. `games` with the following schema:
   - `game_date`: The date of the game
   - `home_team_id`: The ID of the home team
   - `home_team_score`: The score of the home team
   - `away_team_id`: The ID of the away team
   - `away_team_score`: The score of the away team
2. `teams` with the following schema:
   - `id`: The ID of the team
   - `name`: The name of the team
   - `city`: The city of the team
   - `primary_color`: The primary color of the team

You should generate your own data for these tables.

Part 2: Data Pipeline

Using Airflow, create a data pipeline that inserts new data into a PostgreSQL database every day. Each endpoint in the API should correspond to a table in the database. For the `games` endpoint, only new data should be inserted.

Additionally, create a “power-ranking” data pipeline. This pipeline should create and update a table that ranks all the teams in the league based on the following rules:

- Each win earns a team two points.
- Each loss earns a team one point.
- Games do not end in a draw.
- If two teams have the same number of points, the team with the higher score differential should be ranked higher. If they also have the same score differential, their ranking can be random.

The `power_ranking` table should have the following schema:

- `rank`: The rank of the team
- `team_id`: The ID of the team
- `team_name`: The name of the team
- `wins`: The number of wins
- `losses`: The number of losses
- `points`: The total points
- `score_differential`: The score differential

This pipeline should run immediately after the first pipeline finishes successfully.

Part 3: SQL

Write the following SQL queries:

1. A query that calculates the average number of wins for teams grouped by their primary color.
2. A query that identifies the “high scoring” game, defined as the game where the combined score of the two teams is the highest.

3. Write a query that returns the team with the highest number of home game wins.
4. Write a query that calculates the total score for each team over all games.
5. Write a query that identifies the team with the most significant improvement in wins between two consecutive years.
6. Write a query that returns the game with the largest point difference between the home and away teams.
7. Write a query that finds the team that played the most games in their home city.
8. Write a query that calculates the win percentage for each team (number of wins divided by total games played).
9. Write a query that identifies the team with the most losses in away games.
10. Write a query that shows the distribution of home and away wins for each team.