Week 7 ETL Pipeline

Part 4 - Documentation & Reflection

Summary of ETL Steps

Extract

1) First we extracted our cleaned data from the original PWHL game event datasets by re-connecting to the database and running the query from week 6. This included tables for players, teams, games, shots, goals, assists, penalties, periods, and locations. We also incorporated external/enhanced data.
2) We then enriched the data by ingesting one of the external data sources from the PWHL HockeyTech API (via LeagueStat). Our original database was created referencing the same data that this API provides and therefore is a perfect source to continue building out this project. This API was reached via the URL (https://lscluster.hockeytech.com/feed/index.php), along with specific parameters passed to access a vast amount of data. Our thought process revolved around easily joining the data so we could obtain more detailed information (i.e. pick players who have scored/ blocked/assisted in a given game and then join other info about them).

Transformation

To clean and format the external data, the external player summary dataset obtained from the HockeyTech API was transformed to ensure consistency, usability, and compatibility with our original shot-level dataset.

1) First we handled missing values and inspected the dataset using `api_df.info()`. We confirmed that all 173 records were non-null. Because the API returned complete season-level statistics for all players, no imputation or row removal was required at this stage.
2) Next we standardized the data types. Although the dataset had no missing values, many numeric fields were initially stored as generic object types. To ensure reliability and proper operations, we converted columns into what we felt were more appropriate data types.
3) We then renamed columns and dropped irrelevant columns for clarity and merging purposes. To align the external dataset with our internal shot-level dataset, `player_id` was renamed to `player_key` and `name` was renamed to `player_name`. Because the API data contains season-level summary statistics, we wanted to keep only the fields relevant for comparison with in-game shot data, so we removed 4 columns (player_name, active, position, and team_code).
4) We then merged the cleaned API dataset with the original shot-level dataset using a left join on `player_key`. A left join was selected to preserve all shot events, avoid dropping players without API matches, and maintain shot-level granularity. After merging, the resulting dataset contained 53 columns combining shot-level event data and player-level season summary stats.
5) Finally we created a derived column combining `year_of_birth` and `birthday` to allow for potential demographic and age-based analysis without modifying the original birthdate field,

and performed a final validation to ensure the final data frame is consistent and usable for analysis.

Load
The last part of our entire anlaysis incorporated exporting the final transformed dataset. After completing all transformation and enrichment steps, we exported the final merged dataset to a CSV file to `team7_final_data.csv` which created a portable, analysis-ready version of the dataset that can be shared, archived, or used in downstream tools outside the database environment.
Note: To maintain data integrity and ensure recoverability, we also saved the final merged dataframe back into our SQLite database.

Description of Challenges/Decisions (made during ingestion)

During ingestion, one of the primary challenges was working with the nested JSON structure returned by the HockeyTech API, which required careful inspection and flattening before it could be converted into a usable dataframe. We also needed to standardize inconsistent data types, rename key fields (such as player_id to player_key) to align with our internal schema, and decide which summary columns were relevant to retain for merging. A key design decision was using a left join when integrating the external player summary data with our shot-level dataset to ensure that no shot events were lost during this process.

How the External Data Enhances the Original Dataset

The addition of season-level performance statistics significantly enhances the original dataset by providing broader context for individual game events, allowing us to compare in-game performance to season averages, analyze efficiency metrics, and support more advanced multi-level analysis that would not be possible using shot-level data alone.