Cricbuzz LiveStats: Real-Time Cricket Insights & SQL-Based Analytics – Project Report

---

Overview

This project focuses on building a Cricket Live Stats System using Streamlit, SQLite, and Python. The goal is to fetch and analyze cricket data from the Cricbuzz API and provide real-time statistics and insights on matches, players, venues, and series.

The objectives are:

- Allow users to browse and filter live matches, players, and venues.
- Provide SQL query execution for deeper insights.
- Enable CRUD operations on the cricket database.
- Store and analyze cricket stats like top runs, wickets, averages, and recent match results.

This system helps users track live scores, explore historical data, and analyze trends in cricket performance.

Data Source

The project integrates with the Cricbuzz API (via RapidAPI) and also uses JSON datasets exported from the API. These datasets are stored in SQLite (cricket.db).

Tables Created

- teams → Team information (team_id, team_name, short_name).
- players → Player details (id, name, batting style, bowling style, country, team_id).
- matches → Match details (id, series, format, teams, venue, status).
- scores → Score details (runs, wickets, overs, innings).
- venues → Venue details (name, location, capacity, profile).
- series → Series information (series_id, name, type, start_date, end_date).
- player_stats → Player performance statistics (matches, innings, runs, average, type).

Methodology

1. **Data Collection**
   - Live match data fetched from Cricbuzz API (/matches/v1/live).
   - Player stats from TopStats API (/stats/v1/topstats).
   - Venues from (/venues/v1).
   - Series from (/series/v1).

- Stored exported JSON files locally (all_team_players.json, all_venues.json, recent_matches.json, player_stats.json).

2. **Database Setup**
   - Created relational tables in SQLite.
   - Linked players → teams, matches → series, matches → venues.
   - Inserted JSON/API data into respective tables using utility functions in utils.py.

3. **Application Development**

   Built using:

   - Streamlit → For interactive dashboards and query execution.
   - SQLite → To store and query structured cricket data.
   - Pandas → For query results and transformations.
   - JSON → To store raw player details, matches, and statistical data fetched from the API.
   - Cricbuzz API → For retrieving live matches, player stats, venues, and other cricket-related data.

Dashboard Visualization

1. **KPI Overview**

   - Total Matches, Teams, Players, Venues, Series.

   Purpose: Snapshot of database scale.

2. **Browse & Filter Section**

   - **Filters**: Format, series, venue, team.
   - **Table View**: Shows matches with team names, scores, and status.

   Purpose: Helps users quickly find matches and results.

3. **SQL Query Insights**

   Some sample queries implemented:

   - Players from India.
   - Matches won by each team.
   - Top run scorers from player_stats.
   - Venues with highest capacity.
   - Series hosted by a given country.

Purpose: Give stakeholders analytical power to explore cricket stats.

4. **CRUD Operations**

    - **Create**: Insert new players, teams, venues, and matches.
    - **Update**: Modify player/team details.
    - **Delete**: Remove records.

    Purpose: Manage the cricket database directly by users.

Insights

- Most runs and wickets dominated by top international players.
- High-capacity venues are primarily in India & Australia.
- Teams with stronger batting lineups show higher win counts.
- Domestic series also contribute significantly to player stats.

Conclusion

The Cricbuzz Live Stats System demonstrates how cricket data can be collected, stored, and analyzed effectively. It enables:

- Real-time tracking of live cricket matches.
- Data-driven insights into players, teams, venues, and series.
- Interactive SQL exploration for custom queries.
- Integration of multiple APIs & JSON data into a unified platform.