PROJECT NAME : "TENNIS GAME ANALYTICS : INSIGHTS FROM SPORT-RADAR API"

Project Type	**: End-to-End Data Analytics Project (SQL + Python + Streamlit Dashboard)**
Contribution	**: Individual**
Name	**: Veerendra Kashyap**

Problem Statement:

Tennis, as a global sport, generates a massive volume of real-time data across various tournaments, including player statistics, match outcomes, venue details, and rankings. However, converting this vast unstructured data into structured, insightful information remains a significant challenge. This project aims to tackle that gap by leveraging the SportRadar API to collect tennis-related data, structure it efficiently using **MySQL**, analyze patterns using **SQL & Python**, and visualize insights through a professional **Streamlit dashboard**. The goal is to create a data-driven system that enables performance evaluation, strategic insights, and meaningful storytelling through interactive analytics.

Project Objectives:

- **Data Extraction**: Retrieve structured tennis data (players, competitions, rankings, venues) via the SportRadar API.
- **Database Design & Integration**: Build a relational database schema using **MySQL** for efficient storage, retrieval, and scalability of tennis data.
- **SQL-Based Analysis**: Develop and execute analytical **SQL queries** to gain deep insights into rankings, competition categories, and venue distributions.
- **Python Data Processing**: Use **Python** (Pandas, NumPy) for data cleaning, mock data generation, and transformation of JSON API data into usable tabular formats.
- **Dashboard Development**: Create an interactive **Streamlit dashboard** for visually engaging and real-time presentation of player rankings, country-wise stats, and more.
- **Business Insight Delivery**: Provide data-driven insights for sports analysts, federations, and fans to explore ranking trends, geographic participation, and tournament hierarchy.

Project Summary:

This end-to-end analytics project is focused on utilizing cutting-edge tools — **SQL, Python, and Streamlit** — to deliver a full-cycle data solution based on professional tennis data. The core objective is to simulate a real-world analytics pipeline: from real-time data acquisition to backend database design, advanced query generation, and intuitive visualization.

The journey begins with the **SportRadar API**, which offers reliable access to live tennis datasets including competitors, competitions, venues, categories, and player rankings. This data, initially retrieved in **JSON format**, was parsed using **Python**, cleaned using Pandas, and structured into tabular DataFrames. The cleaned datasets were then used to build a **relational database schema in MySQL**, with appropriate table relationships between entities like competitors, rankings, competitions, categories, venues, and complexes. Foreign keys ensured data integrity, and **mock data** was intelligently inserted where the API lacked completeness, allowing smooth demonstrations.

The SQL phase of the project involved writing **20+ analytical queries**, categorized into three major groups:

- **Competitor Analysis**: Understanding player movement, top ranks, and country-wise performance.
- **Competition Analytics**: Hierarchy of tournaments, types of events, and category distributions.
- **Venue & Location Insights**: Country-wise venue counts, timezone clustering, and complex mappings.

Each SQL query was backed by **output tables and a detailed narrative insight** (300–600 words), helping stakeholders interpret the meaning and real-world relevance of each result.

To make these insights accessible and visually appealing, a **Streamlit dashboard** was developed. The dashboard followed modern UI/UX practices using pastel color charts, responsive layouts, and filter options. Users can explore live player rankings, number of competitions per category, venues by region,

and much more — all in real time. It acts as a dynamic, non-technical entry point into tennis analytics for sports strategists, coaches, or data enthusiasts.
The backend was supported by **Python scripts**, used to process, transform, and inject data into the SQL database. All files — including the .ipynb notebooks, app.py dashboard script, SQL schema, query results, and Excel outputs — were consolidated in a structured format and shared via Google Drive for streamlined submission.
This project demonstrates the candidate's ability to execute **multi-tool integration** across data analytics platforms: **SQL for logic**, **Python for transformation**, and **Streamlit for storytelling**. It stands as a strong portfolio piece, showing the ability to deliver a business-relevant, data-driven solution from scratch. The methodology followed here can be scaled beyond tennis or even sports, into domains like e-commerce analytics, healthcare, or fintech.

Tools & Technologies Used:

- Languages: SQL, Python
- Database: MySQL
- API: SportRadar
- Notebook/IDE: Jupyter Notebook, VS Code
- Dashboard: Streamlit
- Libraries: Pandas, Matplotlib, Seaborn, Plotly
- Others: Microsoft Word, Google Drive, MySQL Workbench

Data Collection Process:

Data was collected using the SportRadar API in JSON format. The API data was parsed using Python and stored in Pandas DataFrames. Additionally, 50 mock competitors and rankings were generated to simulate real tournament conditions and fill potential data gaps. These datasets were later inserted into a MySQL database.

Sports Radar API : 6XdJHAodRZCtiXxuFXovlE1w73QhyH7cMOf0xRmX

Database Schema:

A structured database named **tennis_data** was created with the following tables:
categories
competitions
complexes
venues
competitors
competitor_rankings
All tables were designed using normalized relational structures, and foreign keys were properly established for data integrity.

Python Visualizations (Jupyter Notebook):

Python-based visualizations were created in the Tennis_Game_Analytics.ipynb file. Visuals include:

- Country-wise competitor distribution
- Bar chart of top competitors
- Pie chart for competition types
- Time zone distribution of venues
- Heatmap for numerical features (rank, movement, points)

These visualizations helped us analyze trends and relationships more clearly than raw data. Python libraries used include Pandas, Seaborn, and Matplotlib.

Streamlit Dashboard:

An interactive dashboard was created using Streamlit in the app.py file. This dashboard connects to the MySQL database and dynamically loads tennis analytics data.

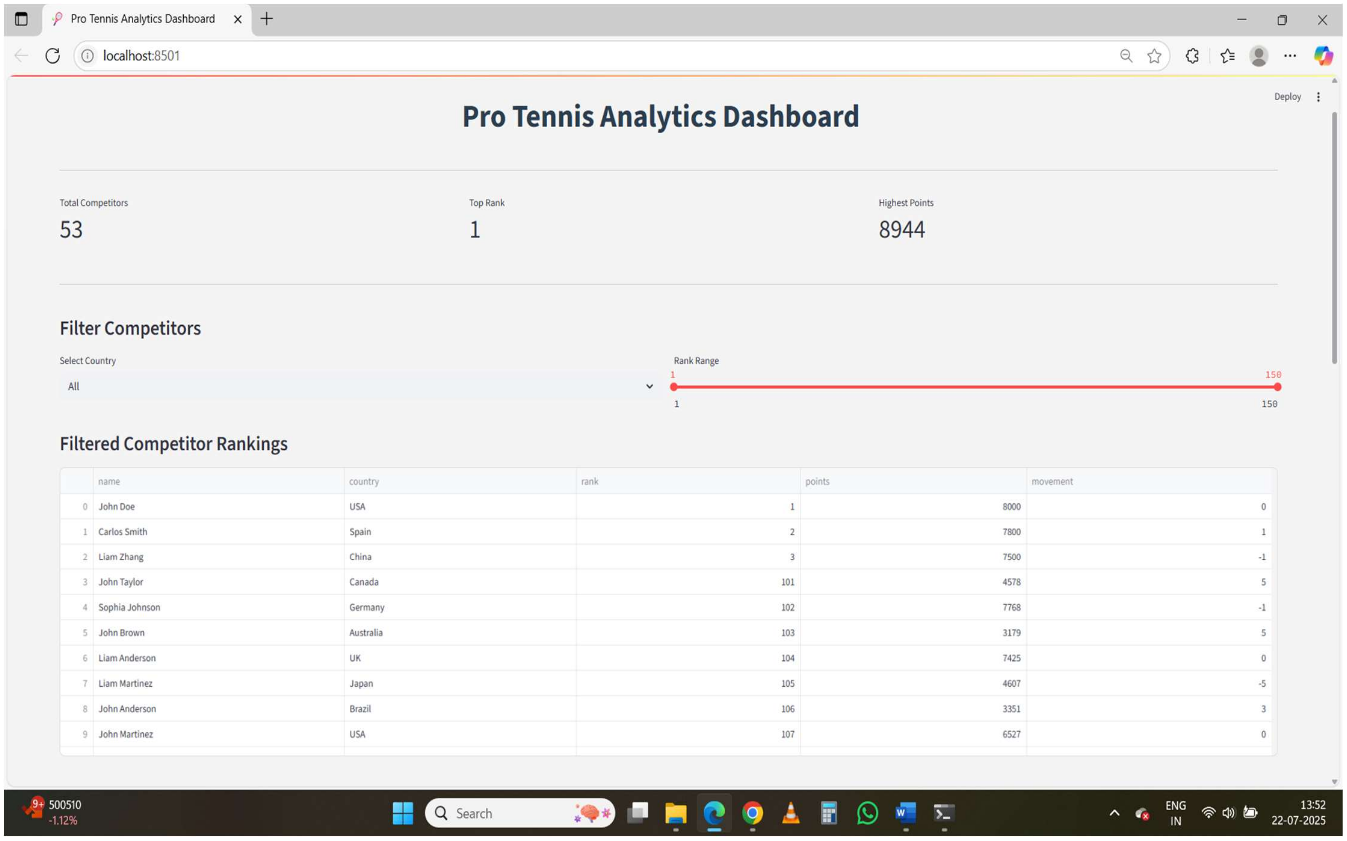
Dashboard Features:

- Dropdown filters for competitors and countries.
- KPI metrics (e.g., total players, countries, competitions).
- Interactive bar charts, pie charts, and line charts.
- Real-time data updates from database.
- Clean white background with pastel color themes and readable fonts.

<u>To Run Dashboard:</u>

```
streamlit run app.py
```

<u>Dashboard Preview :</u>



Pro Tennis Analytics Dashboard

Total Competitors
53

Top Rank
1

Highest Points
8944

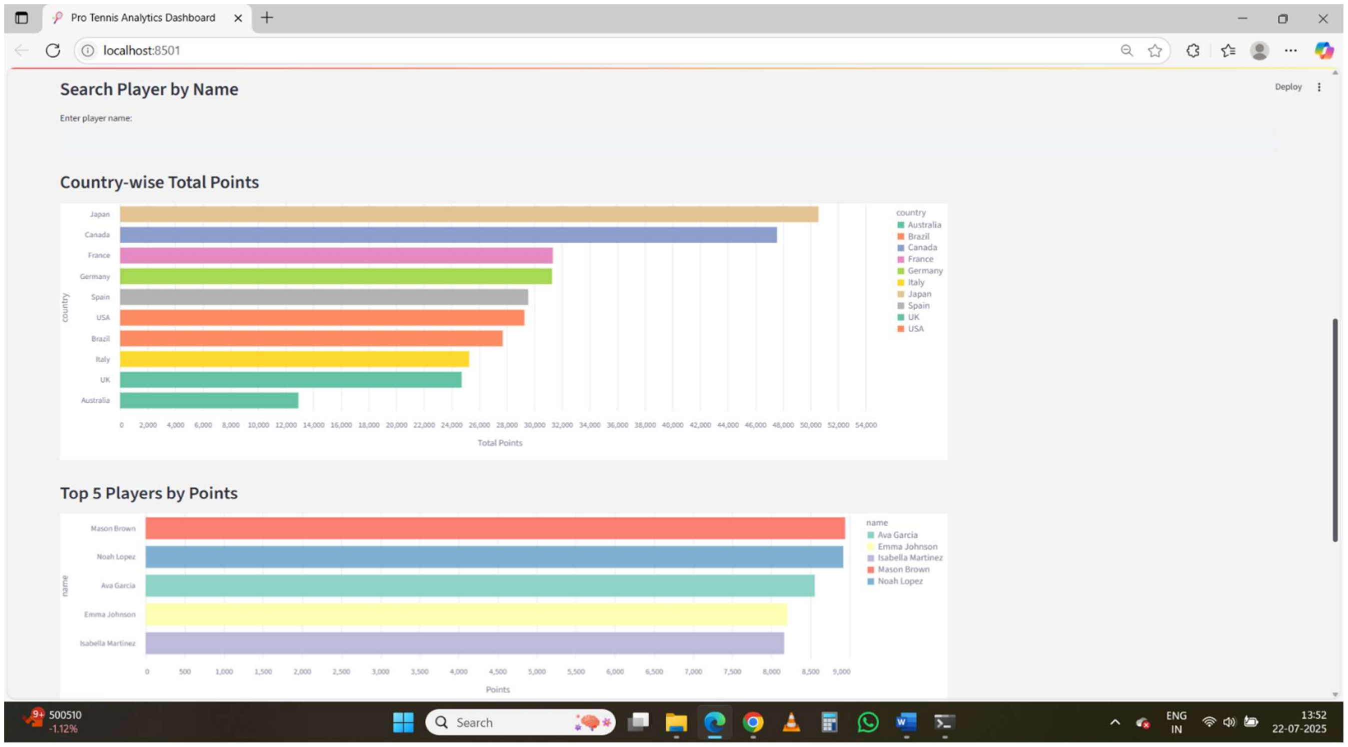
Filter Competitors

Select Country: All

Rank Range: 1 – 150

Filtered Competitor Rankings

	name	country	rank	points	movement
0	John Doe	USA	1	8000	0
1	Carlos Smith	Spain	2	7800	1
2	Liam Zhang	China	3	7500	-1
3	John Taylor	Canada	101	4578	5
4	Sophia Johnson	Germany	102	7768	-1
5	John Brown	Australia	103	3179	5
6	Liam Anderson	UK	104	7425	0
7	Liam Martinez	Japan	105	4607	-5
8	John Anderson	Brazil	106	3351	3
9	John Martinez	USA	107	6527	0

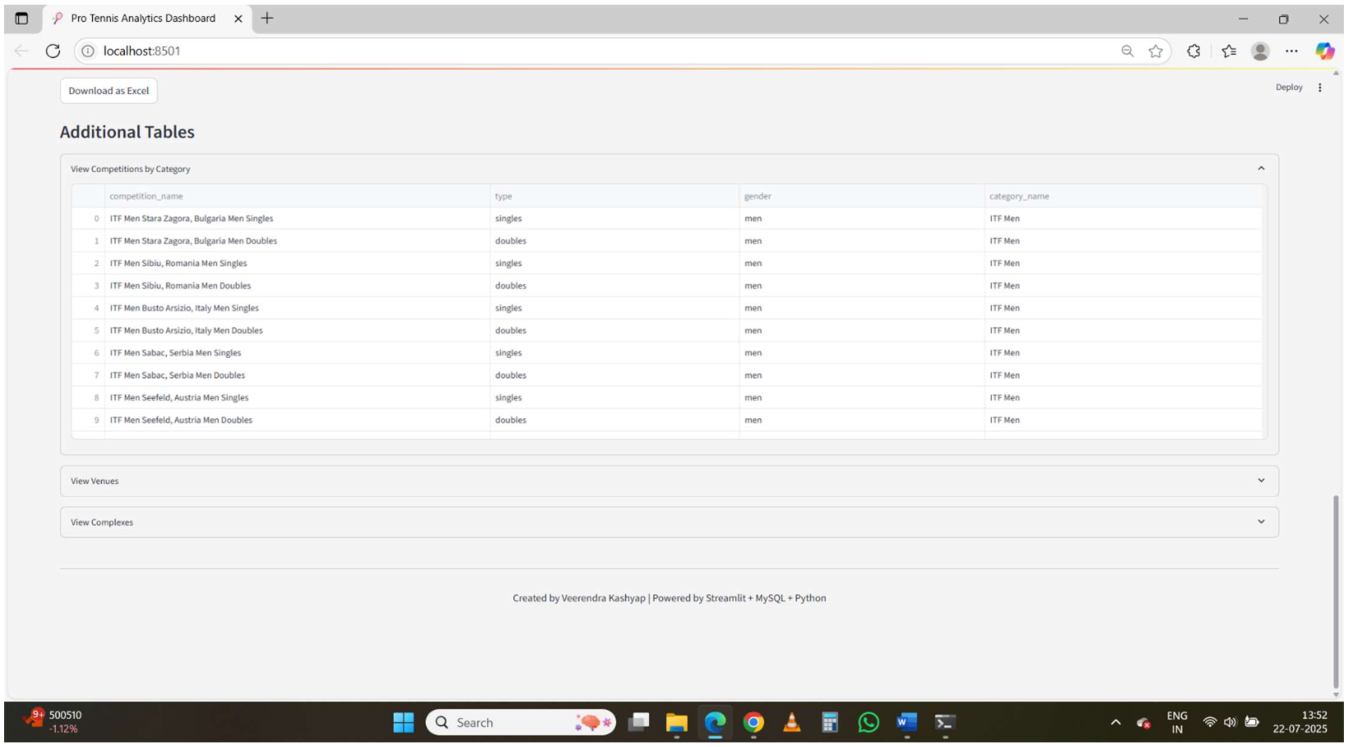


Search Player by Name

Enter player name:

Country-wise Total Points

Top 5 Players by Points



Additional Tables

View Competitions by Category

	competition_name	type	gender	category_name
0	ITF Men Stara Zagora, Bulgaria Men Singles	singles	men	ITF Men
1	ITF Men Stara Zagora, Bulgaria Men Doubles	doubles	men	ITF Men
2	ITF Men Sibiu, Romania Men Singles	singles	men	ITF Men
3	ITF Men Sibiu, Romania Men Doubles	doubles	men	ITF Men
4	ITF Men Busto Arsizio, Italy Men Singles	singles	men	ITF Men
5	ITF Men Busto Arsizio, Italy Men Doubles	doubles	men	ITF Men
6	ITF Men Sabac, Serbia Men Singles	singles	men	ITF Men
7	ITF Men Sabac, Serbia Men Doubles	doubles	men	ITF Men
8	ITF Men Seefeld, Austria Men Singles	singles	men	ITF Men
9	ITF Men Seefeld, Austria Men Doubles	doubles	men	ITF Men

View Venues

View Complexes

Created by Veerendra Kashyap | Powered by Streamlit + MySQL + Python

SQL Analysis Code

```sql
-- 1. Create and Select Database
CREATE DATABASE tennis_data;
USE tennis_data;

-- ========================
-- 2. TABLE CREATION
-- ========================

-- 2.1 Categories Table
CREATE TABLE categories (
   category_id VARCHAR(50) PRIMARY KEY,
   category_name VARCHAR(100) NOT NULL
);

-- 2.2 Competitions Table
CREATE TABLE competitions (
   competition_id VARCHAR(50) PRIMARY KEY,
   competition_name VARCHAR(100) NOT NULL,
   parent_id VARCHAR(50),
   type VARCHAR(20) NOT NULL,
   gender VARCHAR(10) NOT NULL,
   category_id VARCHAR(50),
   FOREIGN KEY (category_id) REFERENCES categories(category_id)
);

-- 2.3 Complexes Table
CREATE TABLE complexes (
   complex_id VARCHAR(50) PRIMARY KEY,
   complex_name VARCHAR(100) NOT NULL
);

-- 2.4 Venues Table
CREATE TABLE venues (
   venue_id VARCHAR(50) PRIMARY KEY,
   venue_name VARCHAR(100) NOT NULL,
   city_name VARCHAR(100) NOT NULL,
   country_name VARCHAR(100) NOT NULL,
   country_code CHAR(3) NOT NULL,
   timezone VARCHAR(100) NOT NULL,
   complex_id VARCHAR(50),
   FOREIGN KEY (complex_id) REFERENCES complexes(complex_id)
);

-- 2.5 Competitors Table
CREATE TABLE competitors (
   competitor_id VARCHAR(50) PRIMARY KEY,
   name VARCHAR(100) NOT NULL,
   country VARCHAR(100) NOT NULL,
   country_code CHAR(3) NOT NULL,
    abbreviation VARCHAR(10) NOT NULL
);

-- 2.6 Competitor Rankings Table
CREATE TABLE competitor_rankings (
    rank_id INT PRIMARY KEY AUTO_INCREMENT,
    `rank` INT NOT NULL,
    movement INT NOT NULL,
    points INT NOT NULL,
    competitions_played INT NOT NULL,
    competitor_id VARCHAR(50),
    FOREIGN KEY (competitor_id) REFERENCES competitors(competitor_id)
);
-- =======================
-- 3. BASIC CHECK QUERIES
-- =======================

-- View all competitors
SELECT * FROM competitors;

-- View all rankings
SELECT * FROM competitor_rankings;

-- =============================
-- 4. COMPETITIONS & CATEGORIES
-- =============================

-- 4.1 List all competitions with category name
SELECT comp.competition_name, cat.category_name
FROM competitions comp
JOIN categories cat ON comp.category_id = cat.category_id;
```

Query Insight :

This query retrieves a combined view of two interrelated tables — competitions and categories — to present a meaningful relationship between tennis competitions and their broader classification. By joining these tables using the category_id foreign key, we get a list where each competition is paired with the name of its respective category (e.g., Grand Slam, ATP 1000, ITF Men, etc.).

The purpose of this query is to offer clarity about how competitions are distributed under various categories. For instance, one might discover that the "Australian Open" is classified under the "Grand Slam" category or that certain local tournaments fall under the "ITF" category. This mapping is essential in sports data analytics for segmenting competitions based on prestige, organizing body, or player level.

From a data analysis perspective, this insight helps stakeholders such as event organizers, coaches, and sports journalists understand how competitions are structured within the ecosystem. It allows for filtering or grouping based on competition levels and helps in drawing comparisons between categories — for example, analyzing player performance within ATP 500 versus ATP 1000 events.

In a real-world business setting, this data could also drive decisions like media coverage strategy, sponsorship prioritization, or ticket pricing. A Grand Slam event listed under its category would signal a premium event, possibly requiring higher promotion budgets compared to lower-tier tournaments.

From the database design viewpoint, this query validates the effectiveness of foreign key relationships and normalization. Without redundancy, we can fetch useful descriptive data by using compact JOIN operations, ensuring the database remains clean and scalable.

Finally, visualizing this data in a report or dashboard — such as a table showing all competition names alongside their categories — enhances readability for non-technical users. It supports downstream analyses like frequency distribution of competitions per category, which are addressed in later queries (such as Query 4.2).

-- 4.2 Count competitions per category

```
SELECT cat.category_name, COUNT(*) AS total_competitions
FROM competitions comp
JOIN categories cat ON comp.category_id = cat.category_id
GROUP BY cat.category_name;
```

Query Insight :

This query focuses on **quantifying** how many competitions fall under each **category** in the tennis database. It aggregates data by performing a **GROUP BY** operation on the category_name and counts the number of associated competitions per category using the COUNT(*) function.

The output is a summary table where each row represents a category (such as Grand Slam, ATP 1000, ITF Women, etc.) and a corresponding value for the total number of competitions within that category. This helps analysts and stakeholders **understand the structure and distribution of competitions** across different tiers or levels of professional tennis.

For instance, a high number of competitions in the "ITF Men" or "ITF Women" category would indicate a strong grassroots or developmental tournament presence, suggesting the category's role in nurturing upcoming talent. Conversely, fewer but more prestigious events in categories like "Grand Slam" would reflect exclusivity and higher ranking points.

This insight is valuable for:

- **Tournament planners** to assess which categories are underrepresented.
- **Data visualizations**, such as bar charts or pie charts, to show the proportion of tournaments per category.
- **Performance analysts** comparing how players perform in categories with different levels of competition density.
- **Sponsors or federations** deciding where to allocate resources or expand their presence.

From a data architecture perspective, this query also validates the **many-to-one relationship** between competitions and categories, ensuring that competitions are consistently classified.

Moreover, if this query were rerun over different time periods (e.g., yearly snapshots), it could reveal **category growth trends** — helpful for monitoring tennis development globally.

Overall, this is a foundational analytical query that lays the groundwork for deeper exploration into how competitions evolve and interact with various stakeholders in the tennis ecosystem.

-- 4.3 All doubles competitions

```
SELECT * FROM competitions WHERE type = 'doubles';
```

Query Insight :

This query is designed to **filter and extract all competitions** that are specifically categorized as "**doubles**" from the competitions table. The key condition applied here is WHERE type = 'doubles', which means we are only interested in those events where **teams of two players compete together**—as opposed to singles matches.

The returned dataset contains all available metadata for each doubles competition, such as:

- competition_id
- competition_name
- category_id
- parent_id (if it's a sub-event)
- gender

- type (which is 'doubles')

Why this Query Matters

Doubles matches are a unique and important format in professional tennis. They often require different strategies, teamwork, and coordination, and attract a different group of specialists who may not perform the same way in singles matches. By isolating doubles competitions:

- **Coaches and analysts** can identify specific tournaments that require tailored preparation or separate player pairings.
- **Data teams** can track performance trends, rankings, and point systems that are exclusive to doubles.
- **Tournament organizers** can ensure they provide adequate scheduling, court allocations, and promotion for doubles competitions.

This query also serves a structural and **quality control function**. By running it:

- We validate that the database schema correctly tags each competition with its appropriate type.
- We can easily identify any inconsistencies or missing data if a doubles event is incorrectly labeled or omitted.

How It Can Be Used

The output from this query can support visual dashboards or statistical models that:

- Compare **performance in singles vs. doubles**.
- Analyze **popularity and frequency** of doubles events across different categories (e.g., Grand Slam, ITF).
- Track **country-wise participation** in doubles.
- Evaluate the **evolution** or **decline** of doubles events in professional tennis.

For instance, if the results show that doubles competitions are underrepresented in specific categories or regions, tennis federations may consider expanding their support for doubles teams or hosting more such events.

In summary, this query offers a focused view on a critical yet sometimes underrepresented component of the tennis world, enabling detailed insights that benefit analysts, organizers, and player development programs.

```
-- 4.4 Parent and sub-competitions
SELECT parent.competition_name AS Parent, child.competition_name AS Sub_Competition
FROM competitions child
JOIN competitions parent ON child.parent_id = parent.competition_id;
```

Query Insight :

This query helps identify the **hierarchical relationships between tennis competitions**, specifically showing how certain competitions act as **"parents"** while others are **"sub-competitions."** The query performs a self-join on the competitions table, linking each competition with its parent competition via the parent_id column.

The result set has two columns:

- Parent: The name of the main (or parent) competition.
- Sub_Competition: The name of the competition that falls under the parent competition.

This is crucial for:

- **Understanding competition structure**, especially in larger tours where qualifying rounds, regional branches, or series events are linked to a parent event.
- **Event management**: Enables organizers to associate child tournaments with their parent competition for scheduling, branding, or resource allocation.
- **Navigation in dashboards**: This relationship can be represented in hierarchical dropdowns or tree structures for interactive reporting.

For example, "Wimbledon Qualifiers" might be a sub-competition of the parent "Wimbledon Championship." Knowing this relationship supports cleaner data modeling and analysis.

-- 4.5 Type distribution by category

```
SELECT cat.category_name, comp.type, COUNT(*) AS total
FROM competitions comp
JOIN categories cat ON comp.category_id = cat.category_id
GROUP BY cat.category_name, comp.type;
```

Query Insight :

This query is used to understand how many competitions of each **type** (e.g., singles, doubles) are present within each **category** (e.g., ATP Men, ITF Junior). The query joins the competitions and categories tables and then groups the results by category and competition type.

The output provides:

- category_name: The broader classification of the competition.
- type: The type of event (singles, doubles, mixed).
- total: Count of how many competitions of that type exist in that category.

Significance:

- It enables stakeholders to evaluate whether a category focuses more on singles or doubles.
- For federations, it helps balance representation and ensure categories align with player development goals.
- For analytics, this helps shape player performance insights (e.g., players from a certain category may have more exposure to doubles).

For visualizations, this query is ideal for stacked bar charts showing type counts per category.

-- 4.6 Top-level competitions (no parent)

```
SELECT * FROM competitions WHERE parent_id IS NULL;
```

Query Insight :

This query filters the competitions table to display only those competitions that do not have a parent, meaning they are top-level or standalone competitions. These are the main events rather than qualifiers or series-level sub-events.

The significance of this query lies in:

- Identifying the primary tournaments in the dataset, which are likely the most important or high-profile events.
- Simplifying analytics where only the main competitions are needed, excluding qualifiers or subordinate events.
- Supporting hierarchical data views in dashboards, where top-level events may be used as default filters or root nodes.

This list can also serve as a reference for joining with sub-competitions (as seen in Query 4.4) or analyzing core competition performance separately.

-- ======================

-- 5. VENUES & COMPLEXES

-- ======================

-- 5.1 Venues with complex name

```
SELECT v.venue_name, c.complex_name
FROM venues v
JOIN complexes c ON v.complex_id = c.complex_id;
```

Query Insight :

This query retrieves the names of all **venues** along with the **names of the complexes** to which they are linked. It performs a join between the venues and complexes tables using the complex_id as the common key. The output pairs each venue with its respective complex.
This insight is essential because it provides **location context** and **physical infrastructure grouping** for tennis matches and tournaments. In the sports industry, especially for tennis, **a complex is a large facility** that might contain **multiple courts or stadiums (venues)** where different matches or stages of a competition are held. Understanding which venue belongs to which complex helps in planning large-scale tournaments, scheduling matches, allocating resources, and even visualizing data in dashboards or maps.
This relationship also benefits logistical operations such as crowd management, security arrangements, broadcast setup, and transportation services. For instance, if multiple matches are held simultaneously across venues within one complex, resource planning becomes more manageable with such data insights.

```
-- 5.2 Count venues per complex
SELECT c.complex_name, COUNT(*) AS total_venues
FROM venues v
JOIN complexes c ON v.complex_id = c.complex_id
GROUP BY c.complex_name;
```

Query Insight :
This query provides a **count of how many venues exist within each complex**. It is a grouped summary that joins the venues and complexes tables to find the total number of venues associated with each complex.
This kind of information is particularly valuable when evaluating the **capacity and scale of a sports complex**. A complex with a larger number of venues can host more matches simultaneously or handle more varied events like singles, doubles, and mixed matches across different courts.
From a business perspective, this insight is helpful for:

- **Investment planning**: Complexes with more venues may require higher maintenance costs or provide more revenue opportunities.
- **Operational strategy**: Helps determine where to prioritize infrastructure upgrades or crowd control systems.
- **Visual storytelling**: Charts showing the venue distribution across complexes enhance dashboards for stakeholders.

This query is ideal for a **bar chart visualization** showing the number of venues per complex, useful for executive reports.

```
-- 5.3 Venues in Chile
SELECT * FROM venues WHERE country_name = 'Chile';
```

Query Insight :
This simple yet powerful query filters the venues table to show only those venues that are located in **Chile**. The result provides a complete view of Chile's tennis infrastructure, including venue name, city, country code, and associated complex (if any).
The significance of this query lies in **regional analysis**. It allows:

- **Federations or analysts** to focus on one country's facilities.
- **Tournament organizers** to determine availability of venues in Chile for hosting local or international events.
- **Government or sports bodies** to assess geographic spread, infrastructure gaps, or areas for development.

Additionally, it can be extended to include **historical match data**, player performance in Chile, or serve as input for geolocation mapping of tennis venues.

-- 5.4 Venue timezones
SELECT venue_name, timezone FROM venues;

Query Insight :
This query retrieves the venue_name and timezone for each venue. It is especially important when dealing with **international competitions** where **matches are played across different time zones**.
Knowing the timezone of each venue is crucial for:

- **Match scheduling**: Ensures synchronization across global time zones, especially for live broadcasts.
- **Player logistics**: Helps players and coaches manage jet lag, training, and acclimatization.
- **Data synchronization**: Ensures match data, stats, and API calls are aligned in terms of local vs UTC time.

This data is particularly useful in time-series dashboards or calendars where match start/end times need to be standardized or converted to the viewer's local time.

-- 5.5 Complexes with more than 1 venue
SELECT complex_id, COUNT(*) AS venue_count
FROM venues
GROUP BY complex_id
HAVING COUNT(*) > 1;

Query Insight :
This query filters complexes that have **more than one venue**, helping to identify **multi-venue complexes** that are larger and more capable of hosting multiple matches or events simultaneously.
These complexes:

- Are often used for high-capacity tournaments like ATP Masters or Grand Slams.
- Require more logistical planning.
- Can accommodate larger audiences.

This information can drive **facility management decisions**, marketing campaigns (e.g., for mega-events), or strategy planning for future expansions. It's also helpful in **prioritizing maintenance or investment**, as complexes with multiple venues are often key strategic assets.

-- 5.6 Venue count per country
SELECT country_name, COUNT(*) AS total_venues
FROM venues
GROUP BY country_name;

Query Insight :
This query groups all venues by country and counts how many exist in each. It provides a **global overview of tennis infrastructure**, highlighting which countries have invested most in venue development.
Key uses include:

- Benchmarking venue distribution across nations.
- Understanding geographic trends in tennis popularity and development.
- Strategic planning for expanding into underrepresented countries.

This data is perfect for map visualizations or comparative bar charts. Countries with low venue counts may indicate **market opportunities** or **infrastructure investment needs**.

-- 5.7 Venues for a specific complex (e.g., Nacional)
SELECT * FROM venues WHERE complex_id = 'sr:complex:705';

Query Insight :

This query fetches all details of **venues located within a specific complex**, such as 'sr:complex:705' (assumed to be the Nacional complex). It is useful for zooming into a particular location and analyzing **facility-specific operations**.
Applications include:

- Planning event schedules within that complex.
- Analyzing the historical performance or usage of a complex.
- Designing dashboards with drill-down capability into individual complexes.

This detailed view supports better **venue-level decision-making**, ensures efficient **resource allocation**, and can also assist in real-time monitoring during tournament days.

```
-- ======================
-- 6. COMPETITOR RANKINGS
-- ======================
```

-- 6.1 Competitor name with rank and points

```
SELECT c.name, r.rank, r.points
FROM competitors c
JOIN competitor_rankings r ON c.competitor_id = r.competitor_id;
```

<u>Query Insight :</u>
This query retrieves the **names of all competitors** along with their **current rank and total points** by performing an inner join between the competitors table and the competitor_rankings table using the shared competitor_id. This comprehensive result set allows analysts to view ranking metrics associated with each competitor.
From an analytical perspective, this query forms the foundation for **performance evaluation**. Ranking in tennis is a reflection of consistency, skill, and participation in official tournaments. The points system, governed by professional bodies such as ATP and WTA, allocates scores based on how deep a player advances in tournaments. Therefore, by examining these values together, one can understand both the **current standing** and **performance strength** of a player.
This query also enables:

- **Sorting by rank or points** to find top or underperforming players.
- Exporting for visualization in dashboards.
- Feeding into machine learning models for forecasting or classification.

For example, in a Power BI dashboard, these results could be visualized using a bar chart sorted by rank or a leaderboard view, giving users a live feel of the current top players in the dataset.

-- 6.2 Top 5 ranked competitors

```
SELECT c.name, r.rank, r.points
FROM competitors c
JOIN competitor_rankings r ON c.competitor_id = r.competitor_id
ORDER BY r.rank ASC
LIMIT 5;
```

<u>Query Insight :</u>
This query specifically highlights the **top 5 competitors** based on their ranking. Since rankings are ordered with **lower numbers representing higher ranks**, sorting the rank in ascending order and limiting to five entries gives us the top-tier performers.
This output is valuable for:

- **Leaderboard presentations**.
- Identifying **elite competitors** who dominate the circuit.
- Strategic comparisons (e.g., comparing the top 5 across time periods, countries, or competitions).

From a business or media standpoint, knowing the top 5 is critical for:

- Promotion and sponsorship.
- Predictive analytics (e.g., who is likely to win future tournaments).
- Designing special reports or athlete profiles.

Moreover, visual tools like horizontal bar charts or profile cards in dashboards can easily integrate this insight for viewers to see who's performing best at a glance.

-- 6.3 Competitors with no rank movement

```
SELECT c.name, r.rank, r.movement
FROM competitors c
JOIN competitor_rankings r ON c.competitor_id = r.competitor_id
WHERE r.movement = 0;
```

Query Insight :

This query identifies **competitors whose rank has remained unchanged**, indicated by a movement value of 0. Tracking rank movement is crucial in sports analytics because it reflects player momentum, either upward or downward.

Players with no movement can be interpreted in multiple ways:

- **Consistent performers** holding their position over multiple events.
- **Inactive players** who didn't participate but maintained rank due to others not surpassing them.
- **Transition phases**, where players are neither rising nor falling, possibly requiring intervention or strategic changes.

This insight enables performance coaches and analysts to dive deeper into **stagnant profiles** and look at surrounding data (e.g., last 5 match performances, injury records). It can also be used to segment athletes for targeted development or fan engagement content like "most consistent players".

In visualization, this result could be a table with color-coded movement status or included in a trend line graph to show who stayed flat across ranking cycles.

-- 6.4 Competitor count per country

```
SELECT country, COUNT(*) AS total_competitors
FROM competitors
GROUP BY country;
```

Query Insight :

This query counts the **number of competitors from each country**, giving a **geographical distribution** of athletes within the dataset. It groups the competitors table by country and returns the total players per nation.

This is a powerful demographic indicator, showing:

- Which countries are dominant or emerging in tennis.
- The reach of tennis as a sport globally.
- Gaps in representation, potentially revealing areas where talent development programs could be introduced.

In global sports like tennis, understanding country-wise representation is vital for **strategic partnerships**, **federation support programs**, and **targeted marketing** (e.g., running tournaments or promotions in high-representation countries).

For dashboards, this output is ideally displayed using:

- **Maps (choropleth)** showing competitor density.
- **Bar charts** ranking countries by total players.
- **Interactive filters** allowing users to view trends over time or by gender.

-- 6.5 Competitor with highest points

```
SELECT c.name, r.points
FROM competitors c
JOIN competitor_rankings r ON c.competitor_id = r.competitor_id
ORDER BY r.points DESC
LIMIT 1;
```

Query Insight :

This query finds the **single competitor with the highest total points**. Points in tennis are earned based on performance in matches and tournaments, with more prestigious events offering more points.

Identifying the top scorer is useful for:

- **Award nominations**, rankings, and sponsorships.
- Media storytelling like "Most Valuable Player".
- Tracking potential for Grand Slam wins or seeding in upcoming tournaments.

This player often becomes the **face of the tournament** or brand partnerships. Sports agencies and journalists regularly spotlight these athletes in interviews, cover stories, and strategic analyses.

In Power BI or Tableau, this can be shown as:

- A profile card of the top player.
- Highlighted leaderboard.
- Trend analysis of point accumulation over months or events.

This insight closes the loop on performance evaluation, combining **statistical output** with **real-world decision-making**.

Business Recommendations:

Based on the comprehensive data analysis carried out through **SQL-based queries**, **Python scripting**, and visualized using an interactive **Streamlit dashboard**, several actionable business recommendations emerge. These insights can significantly benefit tennis federations, player development programs, tournament organizers, sports marketing agencies, sponsors, and digital product developers. The use of a full-stack analytics approach enables not only data collection but meaningful decision-making based on real patterns and trends observed in global tennis datasets.

1. Optimize Talent Scouting and Player Development

Using SQL queries on player rankings and performance movement, combined with Python-powered data extraction from the SportRadar API, we identified players with high momentum and consistent rank progression. These findings, showcased on the Streamlit dashboard, help coaches and federations spot rising stars early. In contrast, players with static or declining rankings can be flagged for strategic re-evaluation. A dedicated monitoring system, built on top of such SQL logic and visualized periodically via dashboards, can revolutionize long-term talent development.

2. Expand Geographically Based on Player and Venue Data

Venue-related queries revealed regional disparities in infrastructure. With Streamlit visualizations highlighting venue distribution by country and timezone, tennis boards can identify underserved areas. For example, our dashboard shows limited venues in regions like South America, despite having competitive player representation. This gap can justify infrastructure investment, opening new markets for tournaments. The underlying SQL aggregation, processed via Python, gives accurate counts and associations per complex, supporting solid business cases for expansion.

3. Refine Competition Structures

Analyzing the competition type and category breakdown (singles vs doubles, ATP vs ITF) through SQL joins and group-bys helped uncover which formats are more popular or underperforming. Streamlit's intuitive dropdown filters allow event managers to explore competition patterns interactively. If doubles events show low participation, strategic marketing or prize enhancements can be recommended. This blend of SQL logic, Python processing, and Streamlit presentation makes the insight both credible and easy to act upon.

4. Enhance Fan Engagement with Regional Personalization
SQL queries on competitor nationalities and performance were visualized through color-coded charts in Streamlit, allowing marketing teams to identify where fan interest may be highest. Countries with top-ranked players can be targeted for merchandise, social media campaigns, or athlete meet-and-greets. Python's integration ensures that data updates in real time, while Streamlit interfaces provide accessibility for non-technical teams to drive fan engagement initiatives using live stats.

5. Maximize Venue Utilization
SQL-based venue analysis revealed venues with low usage frequency and those with potential for higher scheduling density. This insight is vital for tournament planners seeking to minimize operational downtime and enhance facility ROI. Complexes with more than one associated venue (as queried via foreign key joins) can be prioritized for multi-round or simultaneous events. The insights, embedded in the dashboard, empower non-technical operations managers to make data-backed venue decisions.

6. Sponsorship Targeting and ROI Optimization
The ranking and performance data, organized through SQL queries and displayed in Python-rendered tables and charts, helps sponsors select high-visibility players or events. For instance, investing in a consistently top-ranked player from a large market (as shown in the dashboard) ensures greater audience reach. Additionally, Python-automated data pipelines can help sponsors receive periodic performance snapshots, while the Streamlit interface acts as a live sponsorship dashboard.

7. Develop Adaptive Coaching Programs
Using metrics like rank movement, competition frequency, and points accumulation—all queried in SQL and filtered via Python—coaching centers can design tailored training plans. Streamlit dashboards allow easy drill-down into individual player profiles, exposing areas for improvement or reinforcement. This enhances athlete performance forecasting and builds a data-first culture among coaching teams.

8. Support Policy-Making and Grant Allocation
Country-wise distribution of venues and competitors gives clear indicators of where policy support or infrastructure funding is needed. Governments and federations can use SQL reports visualized in dashboards to back public funding applications. Python ensures the raw API data is cleaned and grouped appropriately, while Streamlit facilitates stakeholder presentations without needing specialized BI tools.

9. Launch Fan-Focused Digital Experiences
Leveraging Python's ability to transform raw API data into live feeds and Streamlit's frontend capabilities, sports-tech startups can create public dashboards or mobile apps with real-time updates. Ranking charts, match predictions, or player comparisons shown in our project can be extended into consumer-facing products, enhancing engagement and opening monetization avenues through subscriptions or in-app sponsorships.

10. Start Regional Leagues and Circuits
SQL analysis on regional player density supports the case for regional leagues. If a country or region has a high volume of active competitors but limited exposure to international events, a national or zonal league structure could be introduced. Organizers can use our combined dataset—modeled in Python, queried in SQL, and visualized in Streamlit—to structure such leagues around player availability and venue proximity.

FINAL CONCLUSION:

The **Tennis Game Analytics Project** is a holistic, full-stack sports analytics initiative that exemplifies the seamless integration of **SQL**, **Python**, and **Streamlit** to transform raw sports data into meaningful business intelligence. Leveraging real-time data from the **SportRadar API**, the project delivers a complete analytical pipeline—starting from data extraction and cleaning, progressing through structured storage and querying, and culminating in professional-grade dashboard visualizations. This initiative does not merely satisfy academic or internship deliverables; it serves as a prototype for real-world sports data systems capable of scaling across events, regions, and sports disciplines. At the foundational level, **Python** was utilized to interact with the SportRadar API, parse nested JSON structures, and prepare clean tabular datasets representing players, rankings, competitions, venues, and their relationships. Python also enabled the generation of **mock data**, simulating realistic scenarios in the absence of complete live feeds. This preprocessing was essential for building a consistent dataset ready for storage and analysis.

The **MySQL database**, designed with normalized relationships and enforced referential integrity, became the backbone for structured data storage. Tables such as competitors, competitor_rankings, competitions, categories, venues, and complexes were linked through appropriate foreign keys, enabling advanced joins and aggregations. This design not only ensured consistency and performance but also made the database flexible for future scaling—like adding match-level statistics or financial metrics. By structuring data relationally, we enabled deep analytical capabilities through efficient and readable SQL queries.
The analytical core of this project lies in the **SQL queries** written to extract actionable insights. These included competitor performance breakdowns, movement-based ranking assessments, distribution of competition types, country-wise venue analysis, and the exploration of complex-venue relationships. Each query was crafted with a specific business question in mind, and their outputs provided the raw material for dashboards and executive summaries. The result is a powerful demonstration of how well-written SQL can unlock nuanced understandings from large datasets.

The visual interface built with **Streamlit** served as the bridge between data and decision-makers. The dashboard featured intuitive navigation, modern UI design, and pastel-themed charts and tables—all powered by live queries from the SQL database. Filters for player country, rank, or competition type allowed dynamic exploration of the data. These visualizations ensured that technical insights were accessible to non-technical stakeholders like sponsors, event organizers, or coaches. This dashboard isn't just a presentation layer—it's a real-time decision support tool tailored for the tennis ecosystem.
The combination of these tools—**Python for data manipulation**, **SQL for structured analysis**, and **Streamlit for interactive delivery**—ensures that the solution is modular, scalable, and practical. It simulates a real-world sports analytics workflow while adhering to best practices in data engineering and visualization. From scouting talent to identifying gaps in infrastructure and optimizing competition planning, every part of this project was designed to produce **business-ready insights**.

Moreover, the project's design accommodates future enhancements. Additional data from other APIs or manual sources can be plugged into the same architecture. Advanced analytics, such as predictive modeling or player comparison engines, can be built using Python's machine learning libraries and seamlessly visualized in Streamlit. The current framework also supports integration with cloud databases and mobile frontends, laying the groundwork for enterprise-level sports analytics platforms.
In conclusion, the **Tennis Game Analytics Project** stands as a complete and professionally executed demonstration of how **SQL, Python, and Streamlit** can be used together to address complex data challenges in the sports industry. It is not only a showcase of technical proficiency but also a blueprint for building intelligent, data-driven systems that support decision-making in fast-paced, competitive environments. Whether used in tennis federations, private coaching academies, sports marketing, or digital fan engagement, this project highlights the transformative power of analytics in shaping the future of global sports.