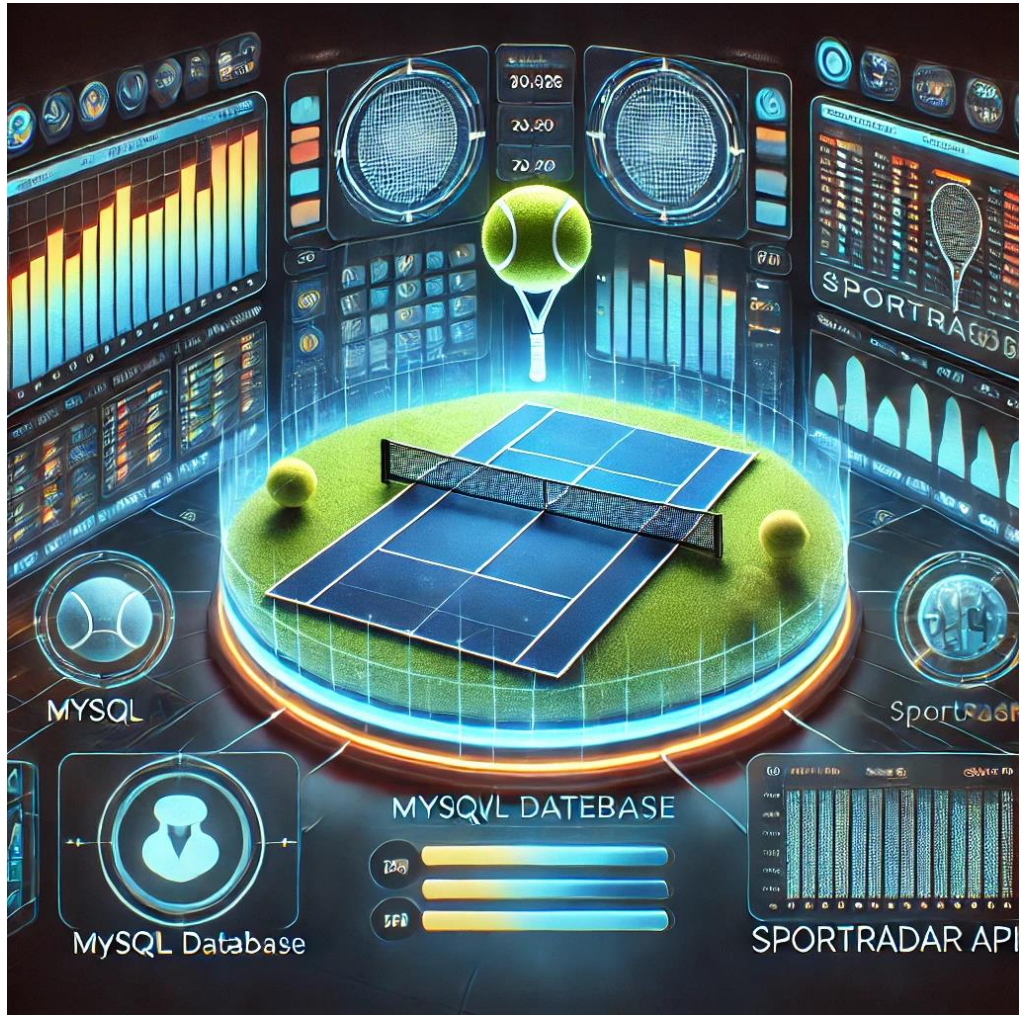


Tennis Data Analytics: Sport Radar API & Streamlit Dashboard

Name : Sham Prasath Kannaiyan

Batch : DS-C-WD-E-B34

Batch Type : WD

Problem Statement

- ◇ **Comprehensive Sports Data Solution** – Manage, visualize, and analyse **sports competition data** from the **Sport Radar API**.
- ◇ **Automated Data Processing** – Parse **JSON data**, structure it in a **relational database**, and ensure efficient storage.
- ◇ **Tournament & Event Insights** – Provide **hierarchical competition structures**, event breakdowns, and key analytics.
- ◇ **Real-Time Analytics & Trends** – Assist **analysts, organizations, and enthusiasts** in exploring **competition trends** dynamically.
- ◇ **Interactive Visualization** – Enable **intuitive dashboards** for deeper insights into tournaments and rankings.
- ◇ **Enhanced Decision-Making** – Deliver **detailed, structured data** to facilitate better sports analysis and engagement.

System Architecture Diagram

Sport radar data to JSON data

Data insertion and retrieval

Implemented SQL queries

Data Visualization Dashboard

Data Collection (API Integration)

Competitions Data

- Stores details of various tennis competitions and their classification.
- Competition ID, Name, Type, Gender, Parent ID, Category ID (FK).

Complexes & Venues

- Manages information about sports complexes and associated venues.
- Complex ID, Name, Venue ID, City, Country, Country Code, Time zone.

Doubles Competitor Rankings

- Tracks player rankings, movement, and performance in doubles competitions
- Rank ID, Rank, Movement, Points, Competitions Played, Competitor ID (FK)

Data Processing & Transformation

API Integration

Used Python requests to fetch real-time data from Sport Radar API

Data Parsing

Extracted JSON response and converted it into structured pandas Data Frames

Data Cleaning

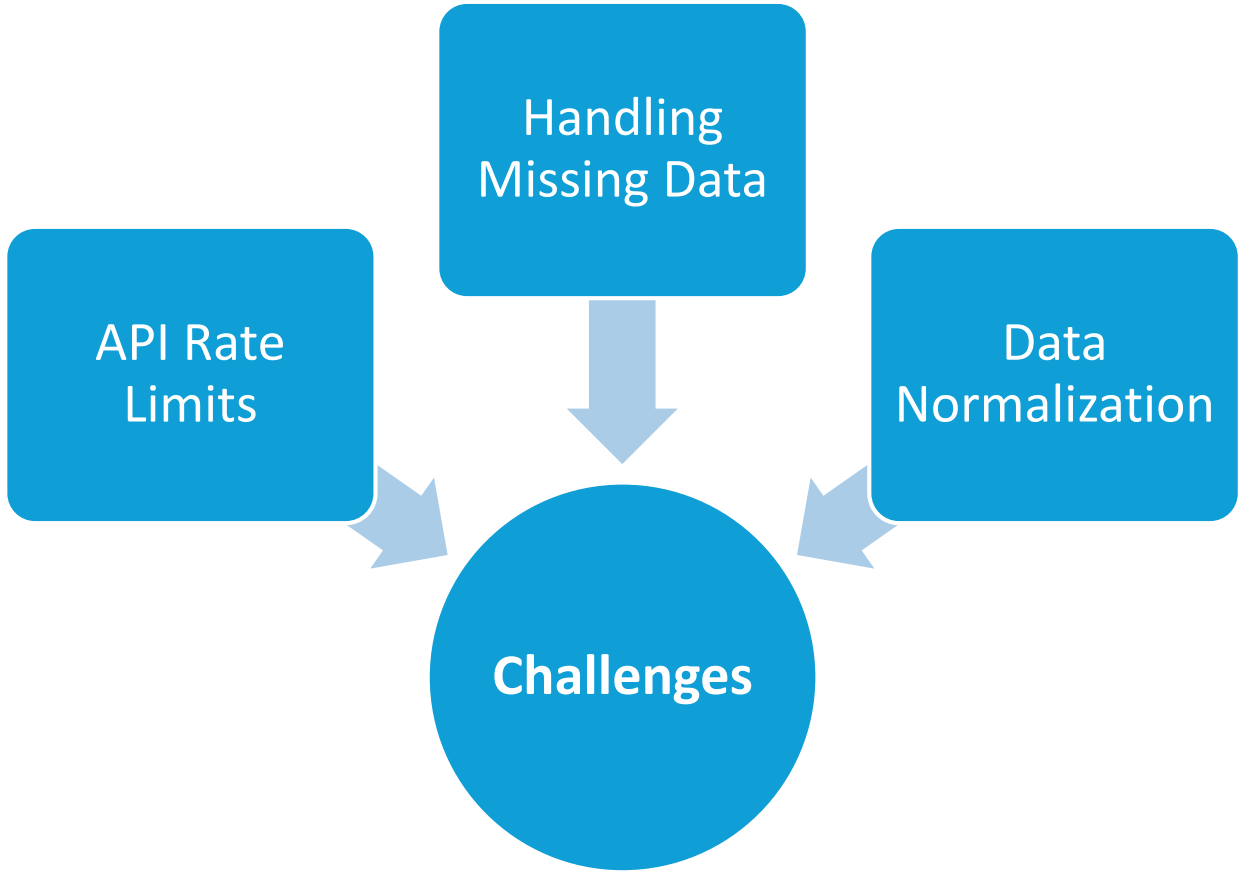
Handled missing values, normalized formats, and ensured consistency

Database Storage

Inserted cleaned data into MySQL tables using MySQL. Connector

Visualization

Connected data to Streamlit for interactive dashboards.



Python code & SQL Data

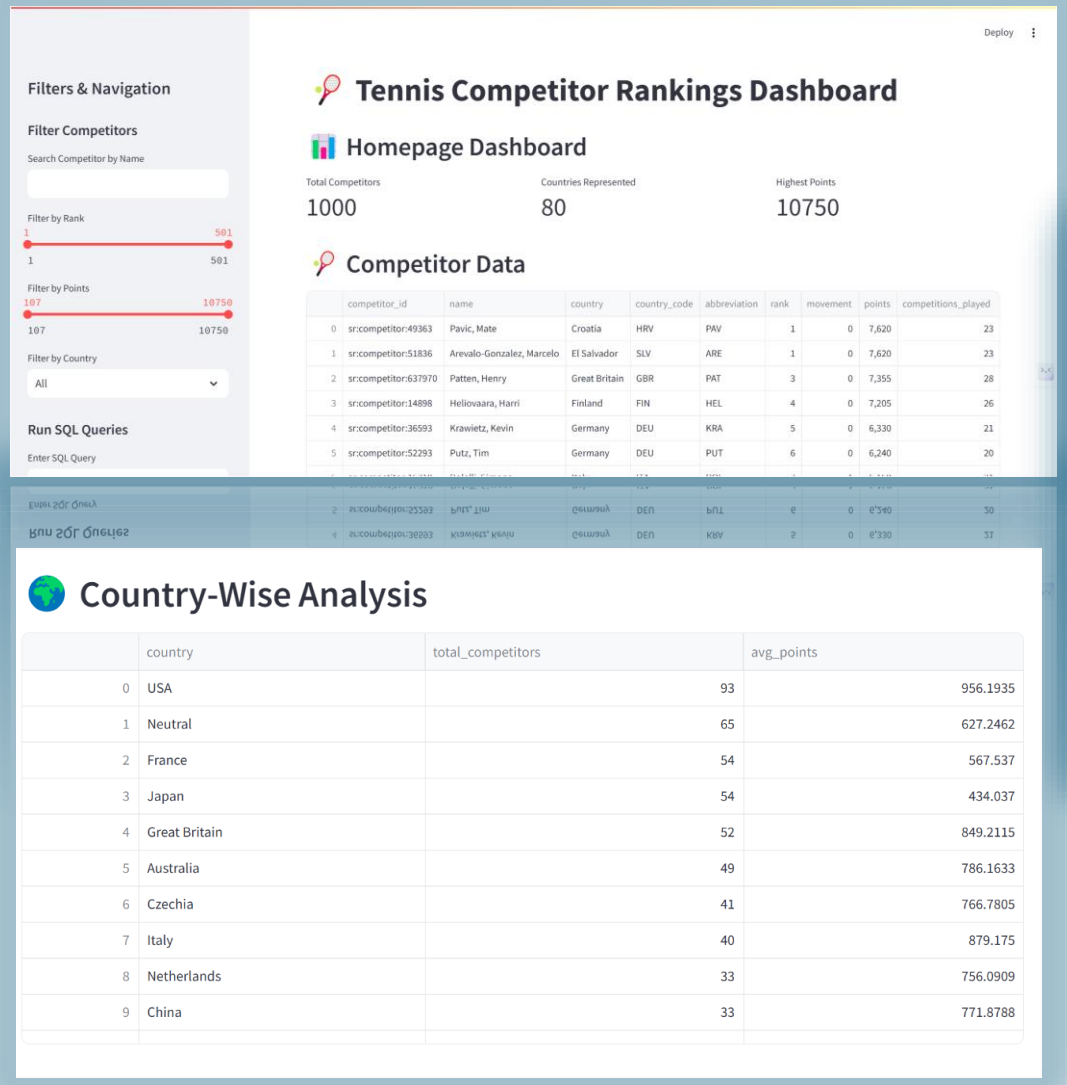
```python
import requests
import json

# API Configuration
API_KEY = "vPAgOe8ZLcCjUWQdDrIlxcQ6OPqH81i4DqzrjAoX"
BASE_URL = "https://api.sportradar.com/tennis/trial/v3/en"


def fetch_data(endpoint):  1 usage
    """Fetch data from SportRadar API and return JSON."""
    url = BASE_URL + "/" + endpoint + ".json?api_key=" + API_KEY
    response = requests.get(url)

    if response.status_code == 200:
        return response.json()

    print("Error:", response.status_code, response.text)
    return None


# Fetch & Display Competitions
competitions = fetch_data("competitions")
if competitions:
    print(json.dumps(competitions, indent=2))
else:
    print("No data received.")
```

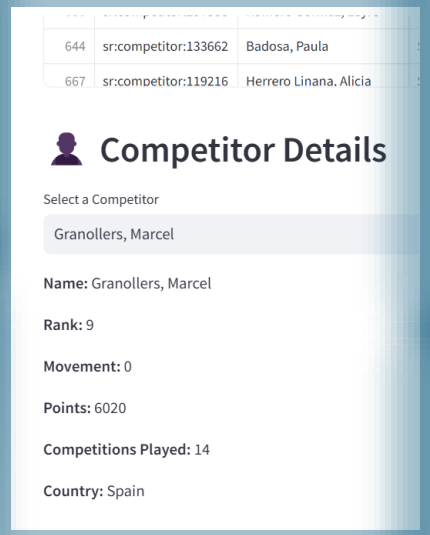
```python
import mysql.connector
from mysql.connector import Error
try:
    # Provide your correct credentials here
    connection = mysql.connector.connect(
        host="localhost",
        user="root",
        password="Sham@1234"
    )
    if connection.is_connected():
        print("Connection to MySQL server was successful!")

        # Create a cursor object to execute SQL queries
        cursor = connection.cursor()

        # SQL query to create a new database
        create_database_query = "CREATE DATABASE IF NOT EXISTS tennis_db"
        cursor.execute(create_database_query)
        print("Database 'tennis_db' created successfully!")
except Error as e:
    print(f"Error: {e}")
finally:
    # Close the connection
    if 'connection' in locals() and connection.is_connected():
        connection.close()
        print("MySQL connection is closed.")
```

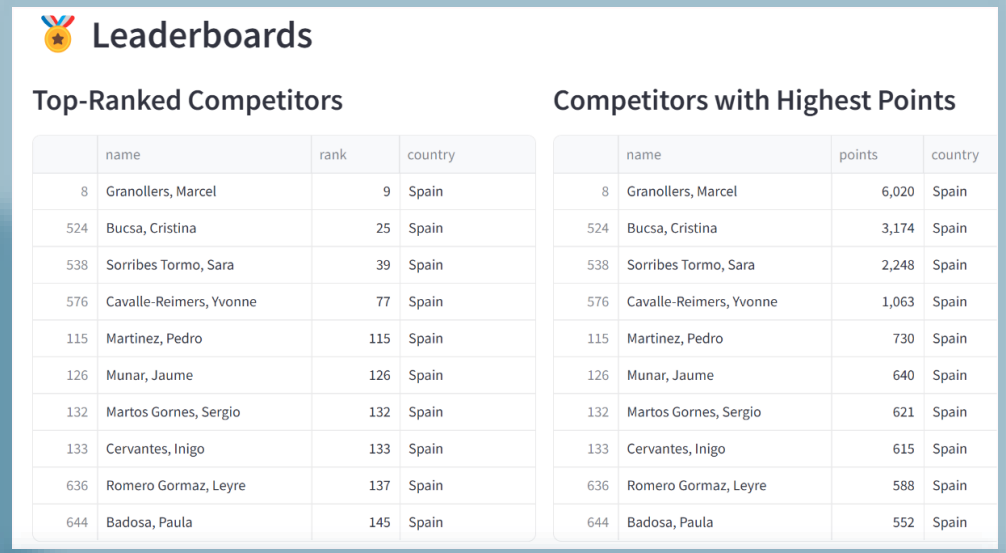
```python
import mysql.connector
import requests

# ✅ API Configuration
API_KEY = "vPAgOe8ZLcCjUWQdDrIlxcQ6OPqH81i4DqzrjAoX"
BASE_URL = "https://api.sportradar.com/tennis/trial/v3/en"

# ✅ Connect to MySQL
def get_db_connection():  3 usages
    return mysql.connector.connect(
        host="localhost",
        user="root",
        password="Sham@1234",
        database="tennis_db"
    )

# ✅ Create Tables
def create_competitions_tables():  1 usage
    connection = get_db_connection()
    cursor = connection.cursor()

    # Create Categories Table
    cursor.execute("""
        CREATE TABLE IF NOT EXISTS categories (
            category_id VARCHAR(50) PRIMARY KEY,
            category_name VARCHAR(100) NOT NULL
```

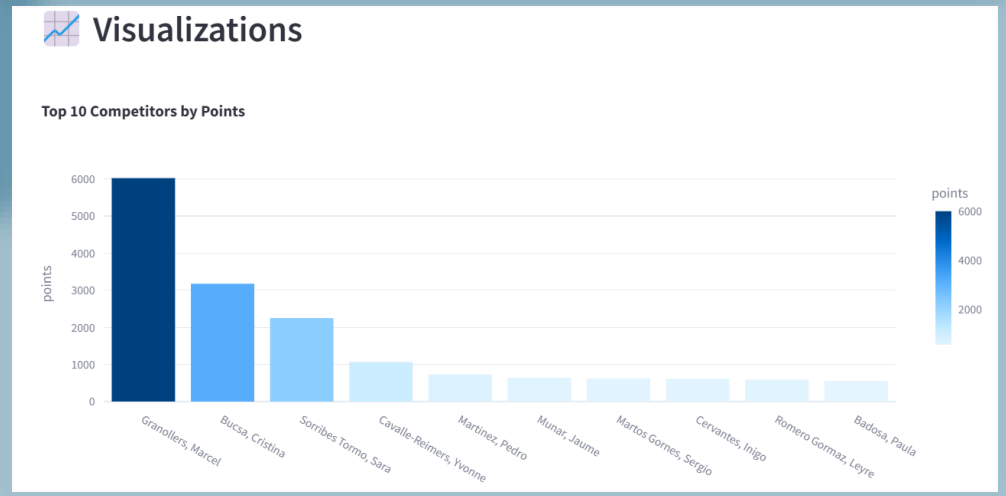
competition_id	competition_name	parent_id	type	gender	category_id
sr:competition:10025	ITF Men Stara Zagora, Bulgaria Men Singles	sr:competition:10023	singles	men	sr:category:785
sr:competition:10027	ITF Men Stara Zagora, Bulgaria Men Doubles	sr:competition:10023	doubles	men	sr:category:785
sr:competition:10031	ITF Men Sibiu, Romania Men Singles	sr:competition:10029	singles	men	sr:category:785
sr:competition:10033	ITF Men Sibiu, Romania Men Doubles	sr:competition:10029	doubles	men	sr:category:785
sr:competition:10037	ITF Men Busto Arsizio, Italy Men Singles	sr:competition:10035	singles	men	sr:category:785
sr:competition:10039	ITF Men Busto Arsizio, Italy Men Doubles	sr:competition:10035	doubles	men	sr:category:785
sr:competition:10043	ITF Men Sabac, Serbia Men Singles	sr:competition:10041	singles	men	sr:category:785

Streamlit Dashboard Features









Future Scope

Advanced Optimization: Ai-Driven Predictive analytics for player performance trends and match outcomes.

UI Enhancements: Real-time data visualization updates and interactive filtering for better user experience

Data Integration: Seamless AI expansion to include historical data and advanced event insights.

Automated Ranking System Generation: Machine learning-based ranking adjustments and competitor performance tracking.

Standards Compliance: Ensuring compatibility with global sports data standards and federations.

Cloud-Based Collaboration: Real-time database aces and multi-user dash board for enchased analytics.

Thank you!

Data is the new game-changer—unlocking insights, elevating strategy, and redefining tennis analytics