# BIG DATA MANAGEMENT

# Assignment 2

# MongoDB assignment

Shreyas Rajapur Sanjay - 1002221283

## Uploading Soccer World Cup Data to MongoDB

The goal of this task is to transform relational data from CSV files into a document-oriented NoSQL structure and store it in MongoDB. The data represents information about countries, players, stadiums, and World Cup history.

I designed two document schemas for MongoDB:

1. COUNTRY Document
    - Contains details of the country, including its capital, population, manager, a list of players, and historical World Cup wins.
2. STADIUM Document
    - Contains details about the stadiums, their location, and the matches they have hosted.

The entire process was implemented using Python and MongoDB.

I loaded the following CSV files using Pandas:

- Country.csv – Contains country names, population, and coach details.
- Match_results.csv – Contains match details such as teams, scores, stadium, and date.
- Players.csv – Contains player details including name, birthdate, and position.
- Player_Assists_Goals.csv – Contains player performance data (goals and assists).
- Player_Cards.csv – Contains player discipline data (yellow/red cards).
- Worldcup_History.csv – Contains historical World Cup winners and host details.

**Data Transformation**

Since MongoDB follows a document-based format, we needed to restructure the data. Instead of storing each record as a separate entry (as in a relational database), we grouped related information into nested documents.

Each COUNTRY document consists of:

- Cname: Country Name
- Capital: Country Capital

- Population: Country's population
- Manager: National Team Coach
- Players: A list containing:
    - First Name, Last Name, Height, Date of Birth
    - Whether the player is a captain
    - Position, Yellow/Red Cards, Goals, Assists
- WorldCupWins: A list containing:
    - Year of the win
    - Host country

To achieve this:

- I merged Players.csv with Player_Assists_Goals.csv and Player_Cards.csv using PID (Player ID) to consolidate player details.
- I filtered Worldcup_History.csv to get the list of countries that won the World Cup.
- Finally, I grouped players by their country and nested them into a single document per country.

Each STADIUM document consists of:

- Stadium: Name of the stadium
- City: City where the stadium is located
- Matches: A list containing:
    - Team 1, Team 2, Scores, and Date of the match

To achieve this:

- I grouped matches by stadium and nested match details under the Matches field.

**Data Loading into MongoDB**

After processing the data:

- We connected to a local MongoDB instance using pymongo.
- We created two collections: COUNTRY and STADIUM.
- Inserted the structured JSON-like documents into MongoDB using insert_many().

## Design and Implementation of the Program

The entire implementation was written in Python, utilizing:

- Pandas for data manipulation.
- pymongo for MongoDB integration.

STEPS

1. **Connect to MongoDB** – Establish a connection to the MongoDB instance.
2. **Load CSV Data** – Read relational data from CSV files using Pandas.
3. **Transform Data into Document Structure**:
    - **COUNTRY Collection**:
        - Includes country details, a nested list of players, and world cup history.
    - **STADIUM Collection**:
        - Includes stadium details and a nested list of matches hosted.
4. **Insert Data into MongoDB** – Store the transformed documents into MongoDB collections.
5. **Print Confirmation Messages** – Indicate successful data insertion.

## Updated Pseudocode

```
1. Connect to MongoDB
  - Initialize MongoClient
  - Create a database "Soccer_Database"

2. Read data from CSV files into Pandas DataFrames:
  - Country.csv → countries_df
  - Match_results.csv → matches_df
  - Players.csv → players_df
  - Player_Assists_Goals.csv → player_goals_df
  - Player_Cards.csv → player_cards_df
  - Worldcup_History.csv → worldcup_history_df

3. Merge Player Data:
  - Merge player_goals_df and player_cards_df with players_df using PID (Player ID)

4. Process COUNTRY data:
  - Create a MongoDB collection "COUNTRY"
  - For each unique country in countries_df:
    - Extract general country details (Cname, Capital, Population, Manager)
    - Retrieve all players associated with the country:
      - Extract player details (Fname, Lname, Height, DOB, Position, etc.)
    - Retrieve World Cup win history
    - Construct a document and append it to the list
  - Insert all country documents into MongoDB

5. Process STADIUM data:
  - Create a MongoDB collection "STADIUM"
  - For each unique stadium in matches_df:
    - Extract stadium details (Stadium Name, City)
    - Retrieve all matches played in that stadium:
      - Extract match details (Team1, Team2, Scores, Date)
    - Construct a document and append it to the list
  - Insert all stadium documents into MongoDB

6. Print success messages after inserting data.
```

## Data Structures Used and its details

1. countries_df (Pandas DataFrame)
   - Stores country-related data from Country.csv.
   - Columns: CountryName, population, no_of_world_cups_won, coach, capital.
2. matches_df (Pandas DataFrame)
   - Stores match results from Match_results.csv.
   - Columns: GID, match_date, match_start_time, team1, team2, score1, score2, stadium, city.
3. players_df (Pandas DataFrame)
   - Stores player data from Players.csv merged with Player_Assists_Goals.csv and Player_Cards.csv.
   - Columns: PID, FullName, Fname, Lname, BirthDate, Country, Height, Club, Position, gamesPlayed, isCaptain, no_of_yellow_cards, no_of_red_cards, goals, assists.
4. worldcup_history_df (Pandas DataFrame)
   - Stores World Cup history from Worldcup_History.csv.
   - Columns: Year, Host, Winner.
5. countries_data (List of Dictionaries)
   - Stores the transformed COUNTRY documents.

JSON STRUCTURE OF DOCUMENT COUNTRY

```
{
 "Cname": "<string>",        // Country Name
 "Capital": "<string>",      // Capital City
 "Population": <integer>,    // Population of the country
 "Manager": "<string>",      // Name of the national team coach
 "Players": [                // List of players in the national team
  {
   "Lname": "<string>",       // Last Name of the player
   "Fname": "<string>",       // First Name of the player
   "Height": <integer>,       // Height of the player in cm
   "DOB": "<string>",         // Date of Birth (YYYY-MM-DD)
   "is_Captain": <boolean>,   // Whether the player is the captain
   "Position": "<string>",    // Playing position (e.g., Forward, Midfielder)
   "no_Yellow_cards": <integer>,  // Number of yellow cards received
   "no_Red_cards": <integer>,     // Number of red cards received
   "no_Goals": <integer>,         // Number of goals scored
   "no_Assists": <integer>        // Number of assists made
  }
 ],
 "WorldCupWins": [           // List of World Cup victories
  {
   "Year": <integer>,       // Year of the World Cup win
   "Host": "<string>"       // Host country of that World Cup
  }
 ]
}
```

6. stadiums_data (List of Dictionaries)

- Stores the transformed STADIUM documents.

  JSON STRUCTURE OF DOCUMENT STADIUM

```
{
 "Stadium": "<string>",      // Stadium Name
 "City": "<string>",         // City where the stadium is located
 "Matches": [                // List of matches played at this stadium
   {
    "Team1": "<string>",     // Name of the first team
    "Team2": "<string>",     // Name of the second team
    "Team1Score": <integer>, // Score of Team1
    "Team2Score": <integer>, // Score of Team2
    "Date": "<string>"       // Match date (YYYY-MM-DD)
   }
 ]
}
```

## QUERIES AND THE RESULTS OBTAINED

### Query 1

Retrieve the list of country names that have won a world cup.

```
db["COUNTRY"].find( { "WorldCupWins": { $ne: [] } },
  { "Cname": 1, "_id": 0 });
```

**QUERY 1 RESULTS**

```
> use Soccer_Database
< switched to db Soccer_Database
> db["COUNTRY"].find( { "WorldCupWins": { $ne: [] } },
      { "Cname": 1, "_id": 0 });
< {
    Cname: 'Argentina'
  }
  {
    Cname: 'Brazil'
  }
  {
    Cname: 'England'
  }
  {
    Cname: 'France'
  }
  {
    Cname: 'Germany'
  }
  {
    Cname: 'Italy'
  }
  {
    Cname: 'Spain'
  }
  {
    Cname: 'Uruguay'
  }
```

## Query 2

Retrieve the list of country names that have won a world cup and the number of world cup each has won in descending order.

```
db["COUNTRY"].aggregate([
  {
    $match: { "WorldCupWins": { $ne: [] } } // Only include countries with at least one World Cup win
  },
  {
    $project: {
      "Cname": 1,
      "TotalWins": { $size: "$WorldCupWins" },
      "_id": 0
    }
  },
  { $sort: { "TotalWins": -1 } } // Sort by the number of World Cup wins in descending order
]);
```

**QUERY 2 RESULTS**

```
> db["COUNTRY"].aggregate([
    {
        $match: { "WorldCupWins": { $ne: [] } } // Only include countries with at least one World Cup win
    },
    {
        $project: {
            "Cname": 1,
            "TotalWins": { $size: "$WorldCupWins" },
            "_id": 0
        }
    },
    { $sort: { "TotalWins": -1 } } // Sort by the number of World Cup wins in descending order
]);

< {
    Cname: 'Brazil',
    TotalWins: 5
  }
  {
    Cname: 'Germany',
    TotalWins: 4
  }
  {
    Cname: 'Italy',
    TotalWins: 4
  }
  {
    Cname: 'Argentina',
    TotalWins: 2
  }
  {
    Cname: 'Uruguay',
    TotalWins: 2
  }
  {
    Cname: 'England',
    TotalWins: 1
  }
  {
    Cname: 'France',
    TotalWins: 1
  }
```

## Query 3

List the Capital of the countries in increasing order of country population for countries that have population more than 100 million.

```
db["COUNTRY"].find(
{ "Population": { $gt: 100 } },
{ "Capital": 1, "Population": 1, "_id": 0 }).sort({ "Population": 1 })
```

**QUERY 3 RESULTS**

```
> use Soccer_Database
< switched to db Soccer_Database
> db["COUNTRY"].find(
  { "Population": { $gt: 100 } },
  { "Capital": 1, "Population": 1, "_id": 0 }).sort({ "Population": 1 })
< {
    Capital: 'Mexico City',
    Population: 122.3
  }
  {
    Capital: 'Tokyo',
    Population: 127.06
  }
  {
    Capital: 'Moscow',
    Population: 142.46
  }
  {
    Capital: 'Abuja',
    Population: 173.6
  }
  {
    Capital: 'Brasilia',
    Population: 202.4
  }
  {
    Capital: 'Washington D.C.',
    Population: 318.9
  }
Soccer_Database>
```

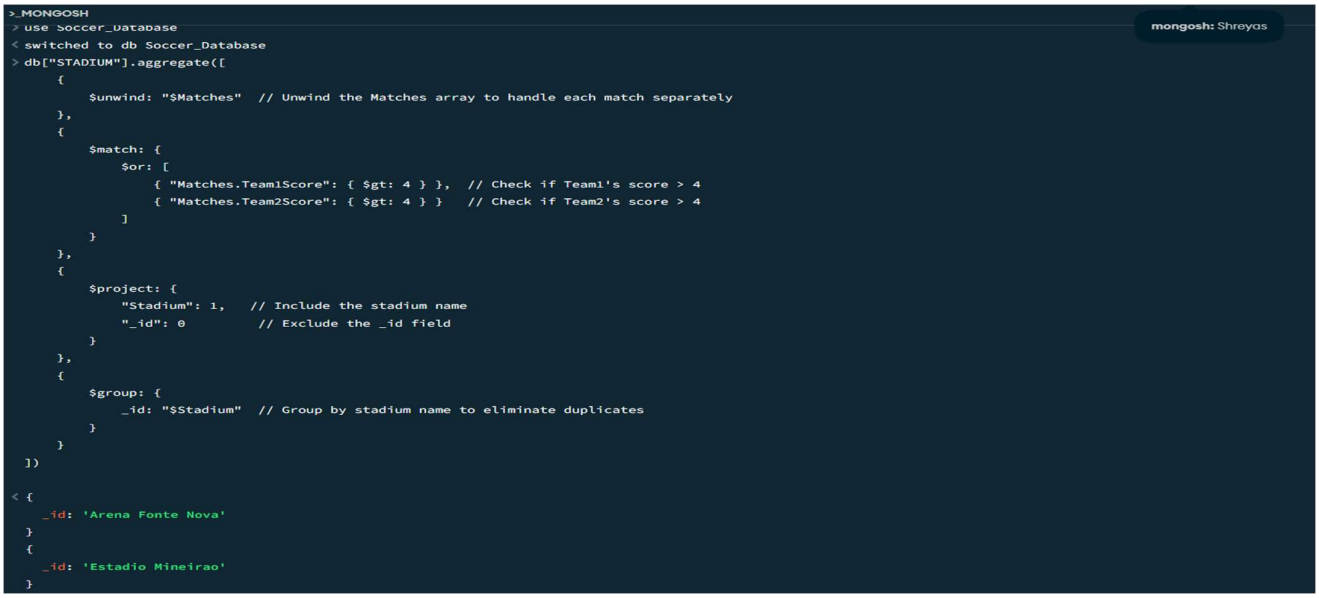
## Query 4

List the Name of the stadium which has hosted a match where the number of goals scored by a single team was greater than 4.

```
db["STADIUM"].aggregate([
  {
    $unwind: "$Matches"  // Unwind the Matches array to handle each match separately
  },
  {
    $match: {
      $or: [
        { "Matches.Team1Score": { $gt: 4 } },  // Check if Team1's score > 4
        { "Matches.Team2Score": { $gt: 4 } }   // Check if Team2's score > 4
      ]
    }
  },
  {
    $project: {
      "Stadium": 1,   // Include the stadium name
      "_id": 0        // Exclude the _id field
```

```
        }
      },
      {
        $group: {
          _id: "$Stadium"  // Group by stadium name to eliminate duplicates
        }
      }
    ])
```

**QUERY 4 RESULTS**

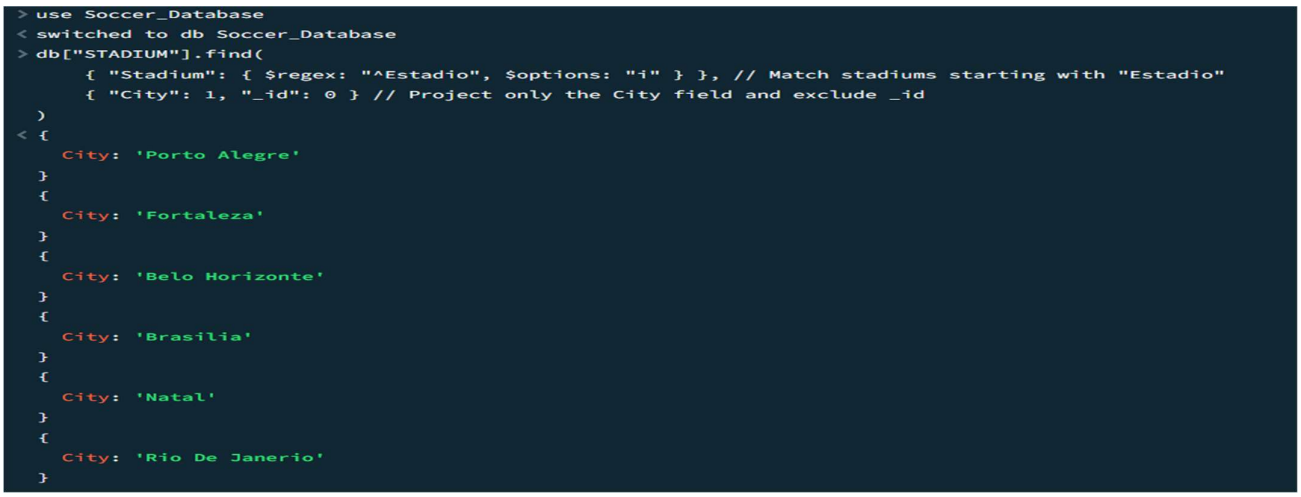



## Query 5

List the names of all the cities which have the name of the Stadium starting with “Estadio”.

```
db["STADIUM"].find(
  { "Stadium": { $regex: "^Estadio", $options: "i" } }, // Match stadiums starting with "Estadio"
  { "City": 1, "_id": 0 } // Project only the City field and exclude _id
)
```

**QUERY 5 RESULTS**

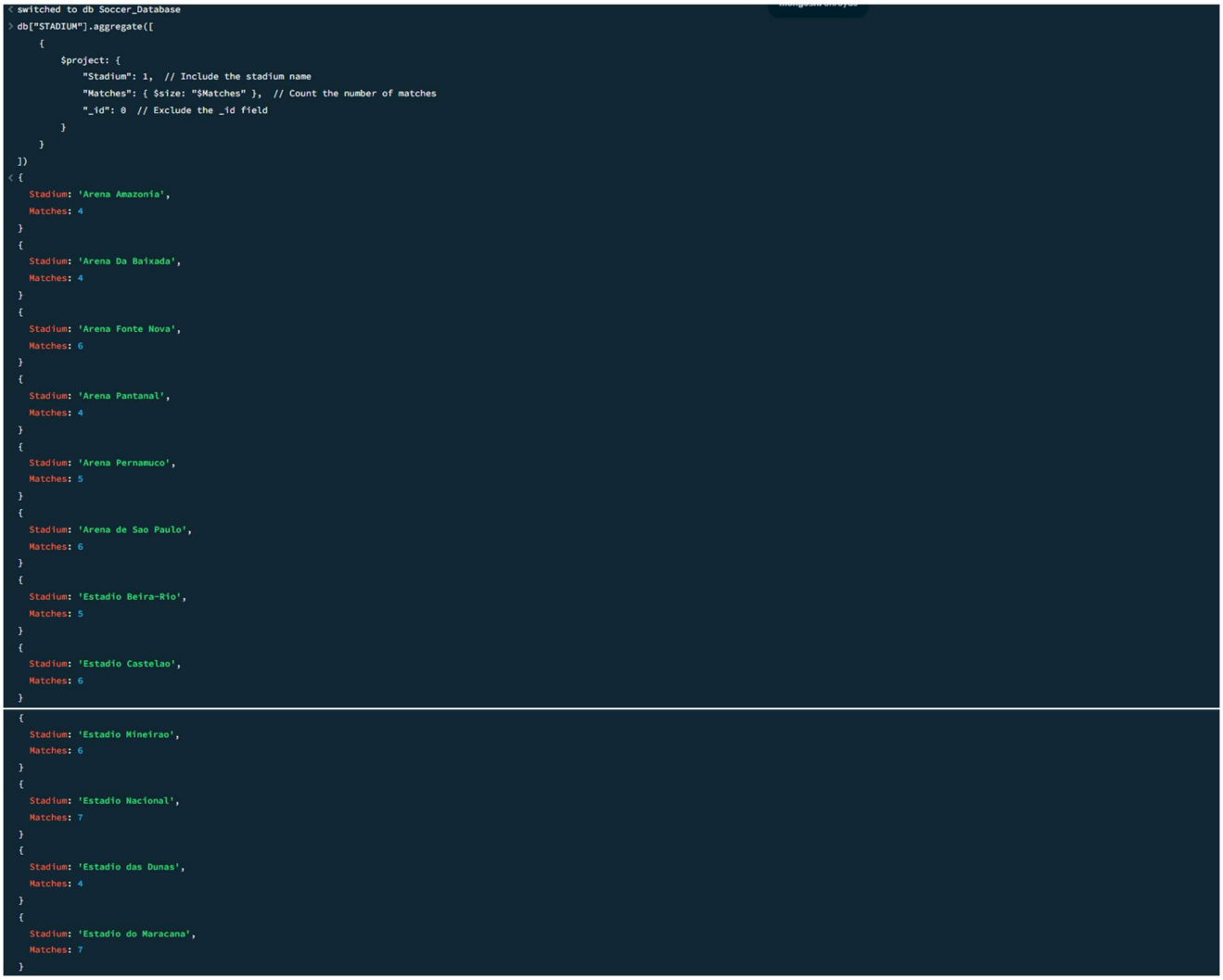


## Query 6

List all stadiums and the number of matches hosted by each stadium.

```
db["STADIUM"].aggregate([
  {
    $project: {
      "Stadium": 1,  // Include the stadium name
      "Matches": { $size: "$Matches" },  // Count the number of matches
      "_id": 0  // Exclude the _id field
    }
  }
])
```

**QUERY 6 RESULTS**



```
< switched to db Soccer_Database
> db["STADIUM"].aggregate([
    {
      $project: {
        "Stadium": 1,  // Include the stadium name
        "Matches": { $size: "$Matches" },  // Count the number of matches
        "_id": 0  // Exclude the _id field
      }
    }
])
< {
  Stadium: 'Arena Amazonia',
  Matches: 4
}
{
  Stadium: 'Arena Da Baixada',
  Matches: 4
}
{
  Stadium: 'Arena Fonte Nova',
  Matches: 6
}
{
  Stadium: 'Arena Pantanal',
  Matches: 4
}
{
  Stadium: 'Arena Pernamuco',
  Matches: 5
}
{
  Stadium: 'Arena de Sao Paulo',
  Matches: 6
}
{
  Stadium: 'Estadio Beira-Rio',
  Matches: 5
}
{
  Stadium: 'Estadio Castelao',
  Matches: 6
}
{
  Stadium: 'Estadio Mineirao',
  Matches: 6
}
{
  Stadium: 'Estadio Nacional',
  Matches: 7
}
{
  Stadium: 'Estadio das Dunas',
  Matches: 4
}
{
  Stadium: 'Estadio do Maracana',
  Matches: 7
}
```

**Query 7**

List the First Name, Last Name and Date of Birth of Players whose heights are greater than 198 cms.

```
db.COUNTRY.aggregate([
   { $unwind: "$Players" },
   { $match: { "Players.Height": { $gt: 198 } } },
   { $project: { "Fname": "$Players.Fname", "Lname": "$Players.Lname", "DOB": "$Players.DOB", "_id": 0 } }
])
```

**QUERY 7 RESULTS**

```
> use Soccer_Database
< switched to db Soccer_Database
> db.COUNTRY.aggregate([
    { $unwind: "$Players" },
    { $match: { "Players.Height": { $gt: 198 } } },
    { $project: { "Fname": "$Players.Fname", "Lname": "$Players.Lname", "DOB": "$Players.DOB", "_id": 0 } }
  ])
< {
    Fname: 'FRASER',
    Lname: 'FORSTER',
    DOB: '1988-03-17'
  }
  {
    Fname: 'LEE',
    Lname: 'BUMYOUNG',
    DOB: '1989-04-02'
  }
```

## Query 8

List the Fname, Lname, Position and No of Goals scored by the Captain of a team who has more than 2 Yellow cards or 1 Red card.

```
db.COUNTRY.aggregate([
   { $unwind: "$Players" },
   { $match: { "Players.is_Captain": true,
            $or: [
              { "Players.no_Yellow_cards": { $gt: 2 } },
              { "Players.no_Red_cards": { $gte: 1 } }
            ]
   }},
   { $project: { "Fname": "$Players.Fname", "Lname": "$Players.Lname", "Position": "$Players.Position",
"Goals": "$Players.no_Goals", "_id": 0 } }])
```

**QUERY 8 RESULTS**

```
> use Soccer_Database
< switched to db Soccer_Database
> db.COUNTRY.aggregate([
     { $unwind: "$Players" },
     { $match: { "Players.is_Captain": true,
                 $or: [
                     { "Players.no_Yellow_cards": { $gt: 2 } },
                     { "Players.no_Red_cards": { $gte: 1 } }
                 ]
     }},
     { $project: { "Fname": "$Players.Fname", "Lname": "$Players.Lname", "Position": "$Players.Position", "Goals": "$Players.no_Goals", "_id": 0 } }
 ])
< {
    Fname: 'THIAGO',
    Lname: 'SILVA',
    Position: 'Defender',
    Goals: 1
  }
  {
    Fname: 'ANTONIO',
    Lname: 'VALENCIA',
    Position: 'Midfielder',
    Goals: NaN
  }
```

THANK YOU