

Olympics Data Management

Project Report

Vamsi Krishna Jilla
Akshat Jain

(857) 763-7256
(857) 334-6863

jilla.v@northeastern.edu

jain.akshat2@northeastern.edu

Percentage of Effort Contributed by Vamsi: 50%

Percentage of Effort Contributed by Akshat: 50%

Signature of Student 1: Vamsi Krishna Jilla

Signature of Student 2: Akshat Jain

Submission Date: 12-10-2023

Business Problem

A robust DBMS is required for organizing a humongous event like olympics where multiple parties are involved. It streamlines vast amounts of data available regarding athletes, events, staff and sponsors. It majorly solves three problems as follows

- **Data Organization and Retrieval:** Managing a large amount of information about athletes, events, and results during the Olympics is challenging without a structured system
- **Accuracy and Integrity:** Ensuring the accuracy and integrity of Olympic data, including athlete details and event results, is prone to errors and inconsistencies without a robust DBMS
- **Decision-Making Support:** Making informed decisions, such as scheduling events, awarding medals and assigning staff is very hard for the organizing committee without a reliable database system providing real-time and accurate information during the Olympics data

Introduction

- The designed DBMS for Olympic organizers is capable of both storing and retrieving data related to athletes, events, sports, sponsors, and staff representing both the country and the organizers
- Furthermore, the DBMS also keeps track of medal counts (gold, silver, and bronze) as well as the total medal count for each country
- It helps event organizers efficiently handle their staff by easily assigning them specific roles for all events taking place in various stadiums based on the schedule.
- The DBMS supports the efficient management of sponsors by storing detailed information, tracking sponsorships for both events and countries, managing financial aspects, enabling organizers to build successful partnerships

Theory behind our project:

Our project's primary objective is to develop a Database Management System (DBMS) to assist the Olympic organizers in efficiently planning and executing the various events associated with the Games.

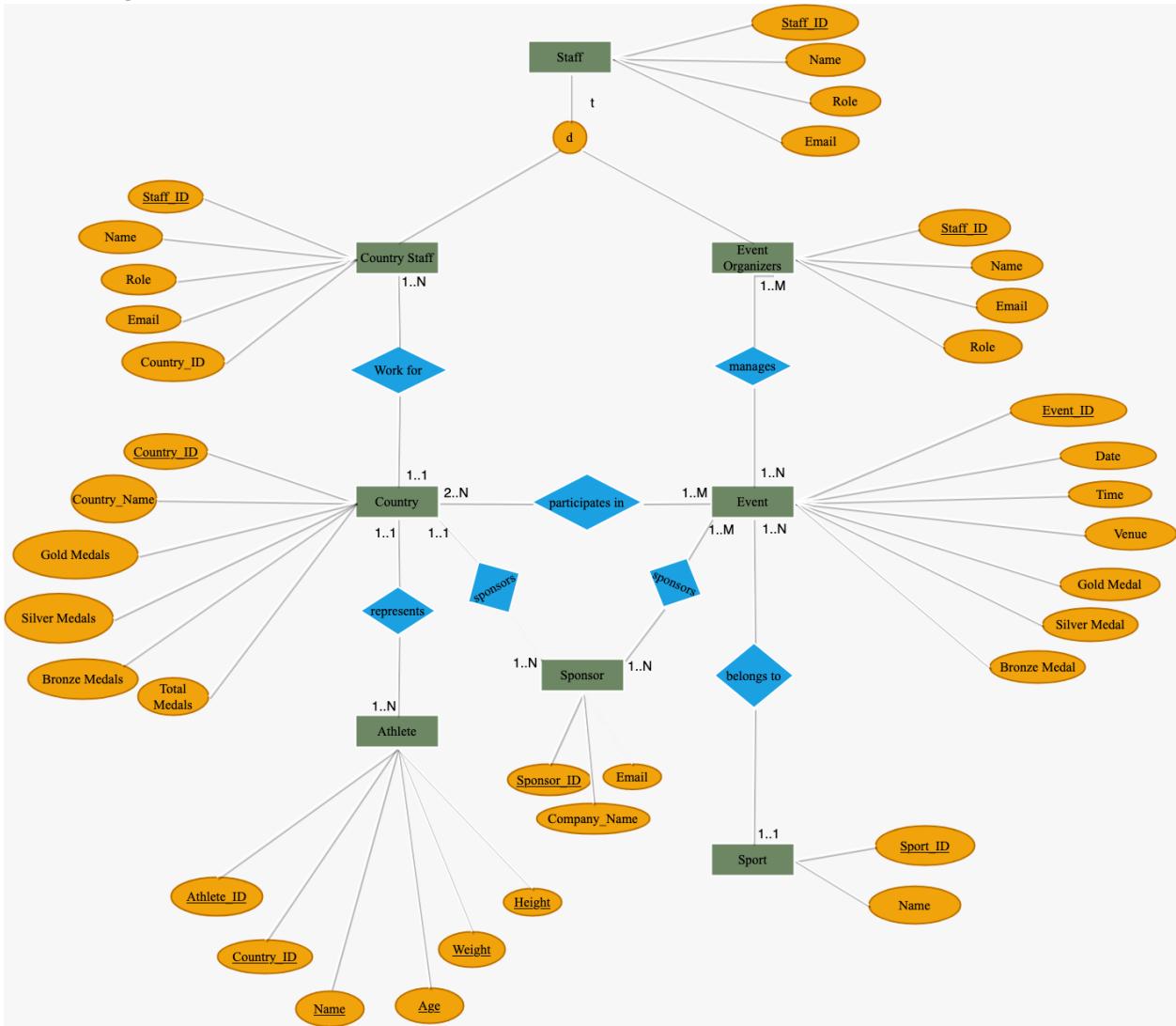
The key entities within our project framework are as follows:

- Country: Countries will participate in various Olympic events, with each country required to engage in at least one event. These countries can possess attributes such as a unique country ID and points accrued during the competition.
- Athlete: Athletes, representing their respective countries, will participate in Olympic events. Each athlete will have a distinct ID and will exclusively represent one country. In team sports, athletes may also be part of larger teams.
- Sport: The Olympics will feature a diverse range of sports, each with multiple events in both male and female categories. These sports will be identified by unique sport IDs and will contain at least one associated event.
- Event: Each Olympic sport will encompass specific events, each characterized by a unique event ID, a sport ID specifying its parent sport, gender categorization (male or female), and other relevant details.
- Sponsor: The project will also accommodate sponsors, both for specific events and for countries/athletes. Sponsors may choose to support particular events or affiliate themselves with specific countries."

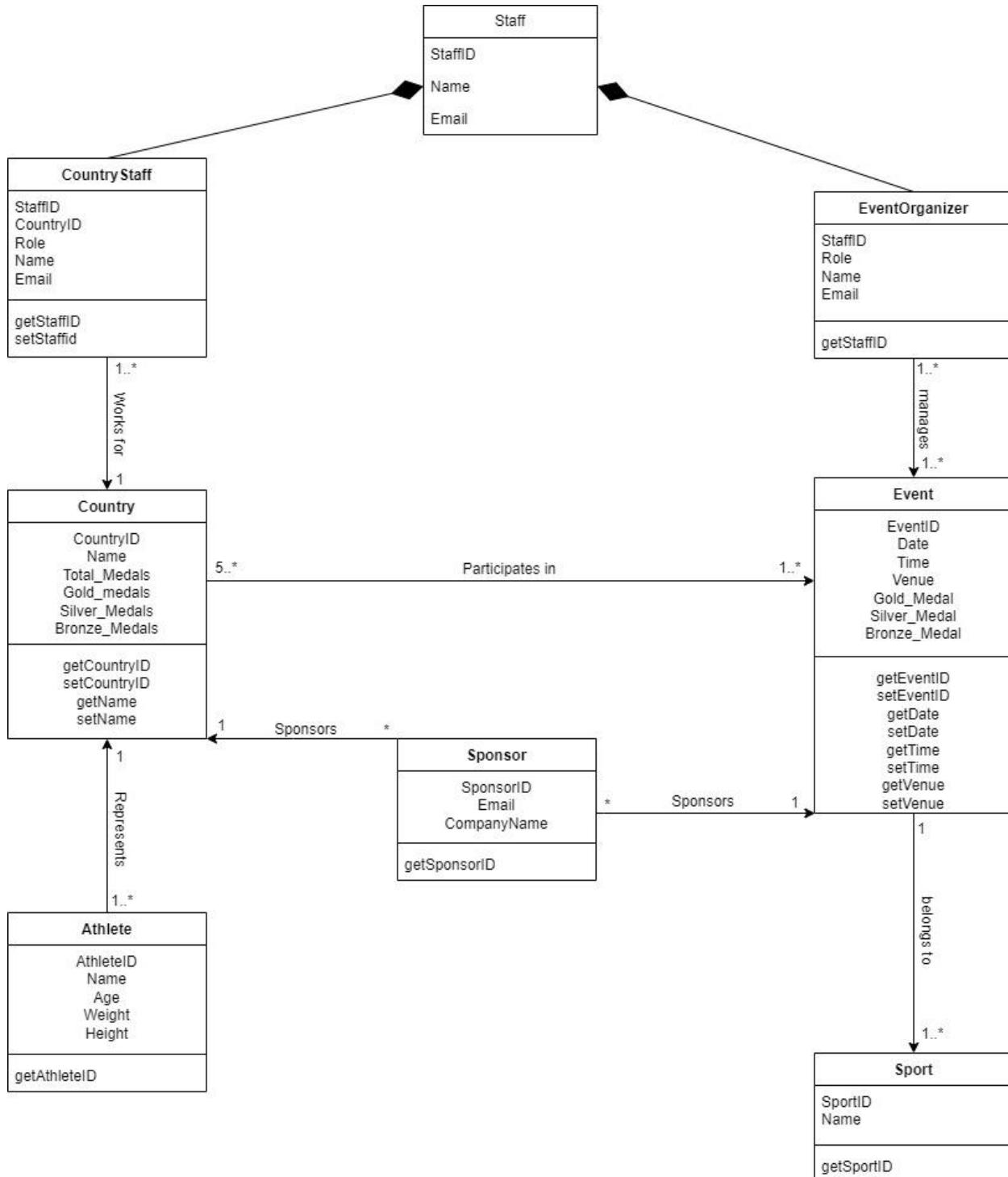
Requirements:

- Each country must have a minimum of one athlete.
- Every country is required to participate in at least one event.
- Sponsorship is optional for countries.
- Each athlete can only represent a single country.
- Each athlete must participate in at least one event.
- Each sport must host at least one event.
- Every event must be associated with a sport.
- Each event must feature a minimum of five participating countries/athletes.
- Every event must secure at least one sponsor.
- Each sponsor must support at least one event or one country.

ER Diagram



UML DIAGRAM



Relational Model:

i. Country (CountryID, Name, GoldMedals, SilverMedals, BronzeMedals, TotalMedals)

ii. Athlete (AthleteID, Name, Age, Weight, Height, CountryID)

- The relationship type represents is mapped by adding a foreign key CountryID referencing Country(CountryID)

iii. Event (EventID, Date, Time, Venue, GoldMedal, SilverMedal, BronzeMedal, SportID)

- The relationship type belongs to is mapped by adding a foreign key SportID referencing Sport(SportID)
- GoldMedal, SilverMedal, BronzeMedal are foreign keys referencing Country(CountryID)

iv. Sponsor (SponsorID, Name, Email, CountryID)

- The relationship type sponsors is mapped by adding a foreign key CountryID referencing Country(CountryID)

v. Sport (SportID, Name)

vi. Staff (StaffID, Name, Role, Email)

- The specialization is mapped by creating 3 relations, one for the superclass and one for each subclass

vii. Country Staff (StaffID, CountryID) - inherits all the attribute types from the entity type Staff

- The relationship type Work for is mapped by adding foreign key CountryID referencing Country(CountryID)

- The specialization is mapped by adding a foreign key StaffID referencing Staff(StaffID)

viii. Event Organizer (StaffID) - inherits all the attribute types from the entity type Staff

- The specialization is mapped by adding a foreign key StaffID referencing Staff(StaffID)

ix. Participates (CountryID, EventID)

- The relationship type participates is mapped by creating a new relation Participates and further adding two foreign keys CountryID, EventID referencing Country(CountryID) and Event(EventID) respectively which also acts as primary keys for the relation

x. Sponsors (SponsorID, EventID)

- The relationship type sponsors is mapped by creating a new relation Sponsors and further adding two foreign keys SponsorID, EventID referencing Sponsor(SponsorID) and Event(EventID) respectively which also acts as primary keys for the relation

xi. Manages (StaffID, EventID)

- The relationship type manages is mapped by creating a new relation Manages and further adding two foreign keys StaffID, EventID referencing Staff(StaffID) and Event(EventID) respectively which also acts as primary keys for the relation

MySQL Implementation:

Query 1: #Listing details of all athletes from India

The screenshot shows the MySQL Workbench interface with the query results for Query 1. The query is:

```
1 #Listing details of all athletes from India
2 • SELECT a.name,c.name as country,a.Age,a.H,a.W
3 FROM athlete a , country c
4 WHERE a.countryID = c.countryID and c.name="India";
5
```

The result grid shows two rows of data:

name	country	Age	H	W
Benjamin Simon	India	34	124.00	105.00
Action Tran	India	29	184.00	170.00

The output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	13:21:41	SELECT a.name,c.name as country,a.Age,a.H,a.W FROM athlete a , country c WHERE a.countryID = c.countryID and c.name="India";	2 row(s) returned	0.015 sec / 0.000 sec

Query 2: #Retrieving the names of athletes with gold medals for events held in "stadium4,"

The screenshot shows the MySQL Workbench interface with the query results for Query 2. The query is:

```
6 #Retrieving the names of athletes with gold medals for events held in "stadium4,"
7 • SELECT athlete.Name AS AthleteName
8 FROM athlete
9 JOIN participates ON athlete.CountryID = participates.CountryID
10 JOIN event ON participates.EventID = event.EventID
11 WHERE event.Venue = 'stadium4' AND event.Gold Medal = athlete.CountryID;
```

The result grid shows two rows of data:

AthleteName
Colorado Henry
Zephania Pugh

The output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	13:21:41	SELECT a.name,c.name as country,a.Age,a.H,a.W FROM athlete a , country c WHERE a.countryID = c.countryID and c.name="India";	2 row(s) returned	0.015 sec / 0.000 sec
2	13:23:04	SELECT athlete.Name AS AthleteName FROM athlete JOIN participates ON athlete...	2 row(s) returned	0.015 sec / 0.000 sec

Query 3: #Finding the top 3 countries with the highest total number of medals.

MySQL Workbench

Local instance MySQL80

Schemas: project

```

12
13  #Finding the top 3 countries with the highest total number of medals.
14 • SELECT CountryID, Name, total_medal
15 FROM country c1
16 WHERE (SELECT COUNT(*) FROM country c2 WHERE c2.total_medal > c1.total_medal) < 3
17 ORDER BY total_medal DESC;

```

Result Grid:

CountryID	Name	total_medal
117	Bangladesh	29
209	Jordan	27
309	French Southern Territories	27

Information: Schema: project

country 3

Action Output:

#	Time	Action	Message	Duration / Fetch
2	13:23:04	SELECT athlete.Name AS AthleteName FROM athlete JOIN participates ON athlete...	2 row(s) returned	0.015 sec / 0.000 sec
3	13:24:11	SELECT CountryID, Name, total_medal FROM country c1 WHERE (SELECT COU...	3 row(s) returned	0.031 sec / 0.000 sec

Query 4: #Finding countries where the total number of medals is greater than the average total number of medals.

MySQL Workbench

Local instance MySQL80

Schemas: project

```

19  #finding countries where the total number of medals is greater than the average total number of medals.
20 • SELECT CountryID, Name, total_medal
21 FROM country
22 HAVING total_medal > (SELECT AVG(total_medal) FROM country)
23 ORDER BY total_medal DESC;

```

Result Grid:

CountryID	Name	total_medal
117	Bangladesh	29
209	Jordan	27
309	French Southern Territories	27
134	Canada	26
189	Guyana	26
204	Iran	26
247	Montserrat	25
144	China	24
299	San Marino	24
114	Azerbaijan	23
122	Burundi	23
157	Algeria	23
162	Eritrea	23
214	Kiribati	23
303	SÃ£o TomÃ© and PrÃ©-Cristo	23
211	Vietnam	22

Information: Schema: project

country 4

Action Output:

#	Time	Action	Message	Duration / Fetch
3	13:24:11	SELECT CountryID, Name, total_medal FROM country c1 WHERE (SELECT COU...	3 row(s) returned	0.031 sec / 0.000 sec
4	13:24:46	SELECT CountryID, Name, total_medal FROM country HAVING total_medal > (SEL...	116 row(s) returned	0.016 sec / 0.000 sec

Query 5: #Finding countries where all athletes are older than 25.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: DMA_Project_Questions

SCHEMAS: project

```

25 #Finding countries where all athletes are older than 25.
26 • SELECT CountryID, Name
27 FROM country c
28 WHERE 25 < ALL (SELECT Age FROM athlete WHERE CountryID = c.CountryID);
29

```

Result Grid: Filter Rows: Export: Wrap Cell Content:

CountryID	Name
100	United Arab Emirates
103	Anguilla
104	Albania
105	Armenia
106	Netherlands Antilles
107	Angola
108	Antarctica
110	American Samoa
112	Australia
113	Aruba
115	Bosnia and Herzego...
116	Barbados
117	Bangladesh
119	Burkina Faso
120	Bulgaria
124	Maldives

country 5

Output:

#	Time	Action	Message	Duration / Fetch
4	13:24:46	SELECT CountryID, Name, total_medal FROM country HAVING total_medal > (SEL...	116 row(s) returned	0.016 sec / 0.000 sec
5	13:25:20	SELECT CountryID, Name FROM country c WHERE 25 < ALL (SELECT Age FRO...	129 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Query 6: # Combining the list of countries with gold medals and those with athletes older than 30.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: DMA_Project_Questions

SCHEMAS: project

```

30 # Combining the list of countries with gold medals and those with athletes older than 30.
31 • SELECT CountryID, Name
32 FROM country
33 WHERE gold_medal > 0
34 UNION
35 SELECT CountryID, Name
36 FROM country c
37 WHERE EXISTS (SELECT * FROM athlete WHERE CountryID = c.CountryID AND Age > 30);

```

Result Grid: Filter Rows: Export: Wrap Cell Content:

CountryID	Name
100	United Arab Emirates
101	Afghanistan
102	Antigua and Barbuda
103	Anguilla
104	Albania
105	Armenia
106	Netherlands Antilles
107	Angola
108	Antarctica
109	Argentina
110	American Samoa
111	Austria

Result 6

Output:

#	Time	Action	Message	Duration / Fetch
5	13:25:20	SELECT CountryID, Name FROM country c WHERE 25 < ALL (SELECT Age FRO...	129 row(s) returned	0.000 sec / 0.000 sec
6	13:25:46	SELECT CountryID, Name FROM country WHERE gold_medal > 0 UNION SELEC...	235 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Query 7: # Listing staff members and their roles for events managed in "Stadium2"

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: DMA_Project_Questions

SCHEMAS

project

Tables Views Stored Procedures Functions

sakila sys world

DMA_Project_Questions

```

39 # Listing staff members and their roles for events managed in "Stadium2"
40 • SELECT staff.Name AS StaffName, staff.Role, manages.EventID, event.Date, event.Time, event.Venue
41 FROM staff
42 JOIN manages ON staff.StaffID = manages.StaffID
43 JOIN event ON manages.EventID = event.EventID
44 WHERE event.Venue = 'Stadium2';

```

Result Grid

Staffname	Role	EventID	Date	Time	Venue
Nigel Cochran	referee	11	2023-12-28	14:20:00	stadium2
Brett Anderson	security	18	2023-05-27	16:14:00	stadium2
Lilith Bradley	cleaner	22	2023-02-22	17:36:00	stadium2
Chloe David	referee	22	2023-02-22	17:36:00	stadium2
Arsenio McDonald	security	27	2023-06-13	14:14:00	stadium2
Stewart Henderson	commentator	27	2023-06-13	14:14:00	stadium2
Hannah Franklin	cleaner	41	2023-10-12	17:55:00	stadium2
Judah Noel	referee	41	2023-10-12	17:55:00	stadium2
Chloe David	referee	41	2023-10-12	17:55:00	stadium2
Lucas Morse	cleaner	41	2023-10-12	17:55:00	stadium2
Chloe David	referee	45	2023-10-04	04:49:00	stadium2
Arden Hunter	referee	45	2023-10-04	04:49:00	stadium2
Lev Williams	security	45	2023-10-04	04:49:00	stadium2
Hannah Franklin	cleaner	48	2023-09-28	16:18:00	stadium2

Result 7

Action Output

#	Time	Action	Message	Duration / Fetch
6	13:25:46	SELECT CountryID, Name FROM country WHERE gold_medal > 0 UNION SELE...	235 row(s) returned	0.000 sec / 0.000 sec
7	13:26:17	SELECT staff.Name AS StaffName, staff.Role, manages.EventID, event.Date, eve...	18 row(s) returned	0.016 sec / 0.015 sec

Object Info Session

Query 8: # Finding staff members and their managed events for country ‘United States’

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: DMA_Project_Questions

SCHEMAS

project

Tables Views Stored Procedures Functions

sakila sys world

DMA_Project_Questions

```

46 # Finding staff members and their managed events for country 'United States'
47 • SELECT staff.Name AS StaffName, staff.Role, manages.EventID, event.Date, event.Time, event.Venue
48 FROM staff
49 JOIN manages ON staff.StaffID = manages.StaffID
50 JOIN event ON manages.EventID = event.EventID
51 JOIN participates ON event.EventID = participates.EventID
52 JOIN country ON participates.CountryID = country.CountryID
53 WHERE country.name like "%United States%";

```

Result Grid

Staffname	Role	EventID	Date	Time	Venue
Quentin Flowers	referee	13	2023-02-03	23:44:00	stadium3
Price Morton	cleaner	40	2022-12-02	00:26:00	stadium4
Hannah Franklin	cleaner	41	2023-10-12	17:55:00	stadium2
Judah Noel	referee	41	2023-10-12	17:55:00	stadium2
Chloe David	referee	41	2023-10-12	17:55:00	stadium2
Lucas Morse	cleaner	41	2023-10-12	17:55:00	stadium2
Hannah Franklin	cleaner	54	2024-01-16	14:19:00	stadium5
Felicia Benson	referee	54	2024-01-16	14:19:00	stadium5
Lucas Morse	cleaner	54	2024-01-16	14:19:00	stadium5

Result 8

Action Output

#	Time	Action	Message	Duration / Fetch
7	13:26:17	SELECT staff.Name AS StaffName, staff.Role, manages.EventID, event.Date, eve...	18 row(s) returned	0.016 sec / 0.015 sec
8	13:26:51	SELECT staff.Name AS StaffName, staff.Role, manages.EventID, event.Date, eve...	9 row(s) returned	0.016 sec / 0.000 sec

Object Info Session

Query 9: # Countries where the difference between maximum and minimum age of athletes of the country is more than 18

The screenshot shows the MySQL Workbench interface. In the top-left corner, the title bar reads "MySQL Workbench" and "Local instance MySQL80". The menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. Below the menu is a toolbar with various icons. The left sidebar, titled "Navigator", shows the schema tree under "SCHEMAS". The "project" schema is expanded, showing Tables, Views, Stored Procedures, and Functions. Other schemas like "classicmodels", "sakila", "sys", and "world" are also listed. The main area contains a query editor window titled "DMA_Project_Qualities*". It displays the following SQL code:

```
54
55  # Countries where the difference between maximum and minimum age of athletes of the country is more than 18
56  SELECT m.CountryID,m.countryname, m.min_age, m.max_age
57  FROM ( select a.countryID,c.name as countryname, min(a.age) as min_age, max(a.age) as max_age from athlete a join country c on a.CountryID=
58  WHERE m.max_age-m.min_age>18)
59
60
61
```

Below the code is a "Result Grid" table with columns: CountryID, countryname, min_age, and max_age. The data is as follows:

CountryID	countryname	min_age	max_age
314	Timor-Leste	19	39
157	Algeria	20	39
340	Mayotte	21	40
318	Turkey	20	39
155	Dominica	19	40

At the bottom of the interface, there is a "Result 9" window showing the "Output" of the session. It lists two actions with their times, messages, and durations:

#	Time	Action	Message	Duration / Fetch
8	13:26:51	SELECT staff.Name AS StaffName, staff.Role, manages.EventID, event.Date, eve...	9 row(s) returned	0.016 sec / 0.000 sec
9	13:27:10	SELECT m.CountryID,m.countryname, m.min_age, m.max_age FROM (select a.co...	5 row(s) returned	0.000 sec / 0.000 sec

NoSQL Implementation

For NoSQL Implementation, we have implemented the database in MongoDB using the Atlas cloud service and used the MongoDB compass to host it on the local device

Database Deployments | Cloud: +

cloud.mongodb.com/v2/656b98e17196d57f7238bb58#/clusters

Atlas Akshat's Org... Access Manager Billing All Clusters Get Help Akshat

Project 0 Data Services App Services Charts

Overview AKSHAT'S ORG - 2023-12-02 > PROJECT 0 Database Deployments

Find a database deployment... Edit Config Create

Cluster Connect View Monitoring Browse Collections ...

FREE S

Visualize Your Data

R 0.3 W 0 Lost 3 hours 0.4/s Connections 17.0 Last 3 hours 26.0 In 137.2 B/s Out 925.2 B/s Lost 3 hours 2.5 KB/s Data Size 331.8 KB / 512.0 MB Last 2 hours 512.0 MB

VERSION REGION CLUSTER TIER TYPE BACKUPS LINKED APP SERVICES ATLAS SQL ATLAS SEARCH

6.0.12 AWS / N. Virginia (us-east-1) M0 Sandbox (General) Replica Set - 3 nodes Inactive None Linked Connect Create

This screenshot shows the MongoDB Atlas Database Deployments dashboard. It features a sidebar with navigation links for Project 0, Data Services, App Services, and Charts. The main area displays a cluster named 'Cluster0' with monitoring tabs for Connect, View Monitoring, and Browse Collections. A 'Visualize Your Data' section includes four charts: R/W operations, connections, network traffic, and data size. Below this, detailed cluster information is listed, including version (6.0.12), region (AWS / N. Virginia), cluster tier (M0 Sandbox), type (Replica Set - 3 nodes), and backup status (Inactive). A 'Create' button is visible at the top right.

MongoDB Compass - cluster0.vybvp1b.mongodb.net/project

Connect Edit View Help

cluster0.vybvp1b ...

My Queries Databases

Search

project

athlete

Storage size: 45.06 kB Documents: 500 Avg. document size: 110.00 B Indexes: 1 Total index size: 32.77 kB

country

Storage size: 28.67 kB Documents: 244 Avg. document size: 126.00 B Indexes: 1 Total index size: 24.58 kB

countrystaff

Storage size: 20.48 kB Documents: 25 Avg. document size: 50.00 B Indexes: 1 Total index size: 20.48 kB

event

Storage size: 20.48 kB Documents: 50 Avg. document size: 151.00 B Indexes: 1 Total index size: 20.48 kB

eventorganizer

>_MONGOSH

This screenshot shows the MongoDB Compass interface connected to the 'cluster0.vybvp1b' database. The left sidebar lists databases (admin, local, project), collections (athlete, country, countrystaff, event, eventorganizer, manage, participates, sponsor, sport, staff), and queries. The main pane displays collection statistics for 'athlete', 'country', 'countrystaff', 'event', and 'eventorganizer'. Each collection card provides storage size, document count, average document size, index count, and total index size. At the bottom, a terminal window labeled '>_MONGOSH' is visible.

We have run 3 queries on the database after adding all the appropriate data:

1) Query to find all events with the venue as stadium4

The screenshot shows the MongoDB Compass interface for the 'project.event' collection. The sidebar on the left lists databases and collections, with 'event' selected. The main pane displays a query for events where the venue is 'stadium4'. The results show two documents:

```
_id: ObjectId('656ce52c7da8dcfe0d8c7400')
EventID: 10
Data: 2023-12-23T00:00:00.000+00:00
Time: "6:02:00"
Venue: "stadium4"
SportID: 3

_id: ObjectId('656ce52c7da8dcfe0d8c7402')
EventID: 12
Data: 2024-10-01T00:00:00.000+00:00
Time: "1:05:00"
Venue: "stadium4"
SportID: 1
```

2) Query to find all countries with total no of medals less than 15

The screenshot shows the MongoDB Compass interface for the 'project.country' collection. The sidebar on the left lists databases and collections, with 'country' selected. The main pane displays a query for countries with a total medal count less than 15. The results show four documents:

```
_id: ObjectId('656ce4767da8dcfe0d8c70fd')
name: "Antigua and Barbuda"

_id: ObjectId('656ce4767da8dcfe0d8c70fe')
name: "Anguilla"

_id: ObjectId('656ce4767da8dcfe0d8c70ff')
name: "Albania"

_id: ObjectId('656ce4767da8dcfe0d8c7102')
name: "Angola"
```

3) Query to find all athletes with age greater than 30 and weight less than 200

The screenshot shows the MongoDB Compass interface. The left sidebar lists databases and collections, with 'project' selected and 'athlete' highlighted. The main area displays the 'project.athlete' collection, showing 500 documents and 1 index. A query is set up to find documents where Age is greater than 30 and Weight is less than 200. The results show four documents, each with an ObjectId and a Name:

- _id: ObjectId('656ce4967da8dcfe0d8c71f0')
Name: "Aquila Lynch"
- _id: ObjectId('656ce4967da8dcfe0d8c71f1')
Name: "Kane George"
- _id: ObjectId('656ce4967da8dcfe0d8c71f2')
Name: "Jason Cantu"
- _id: ObjectId('656ce4967da8dcfe0d8c71f4')
Name: "Bruno Rieus"

Python Implementation:

For the Python Application, we have opened the connection to the MySQL Server in Jupyter Notebook-(For easy Implementation)

We have run 3 queries to retrieve data and have plotted the data into Bar Graph and Pie chart

```
In [1]: import mysql.connector  
import os  
import matplotlib.pyplot as plt
```

```
In [2]: mydb=mysql.connector.connect(host='localhost',user='root',password='Mysqlkapas')
```

```
In [3]: mycursor=mydb.cursor()
```

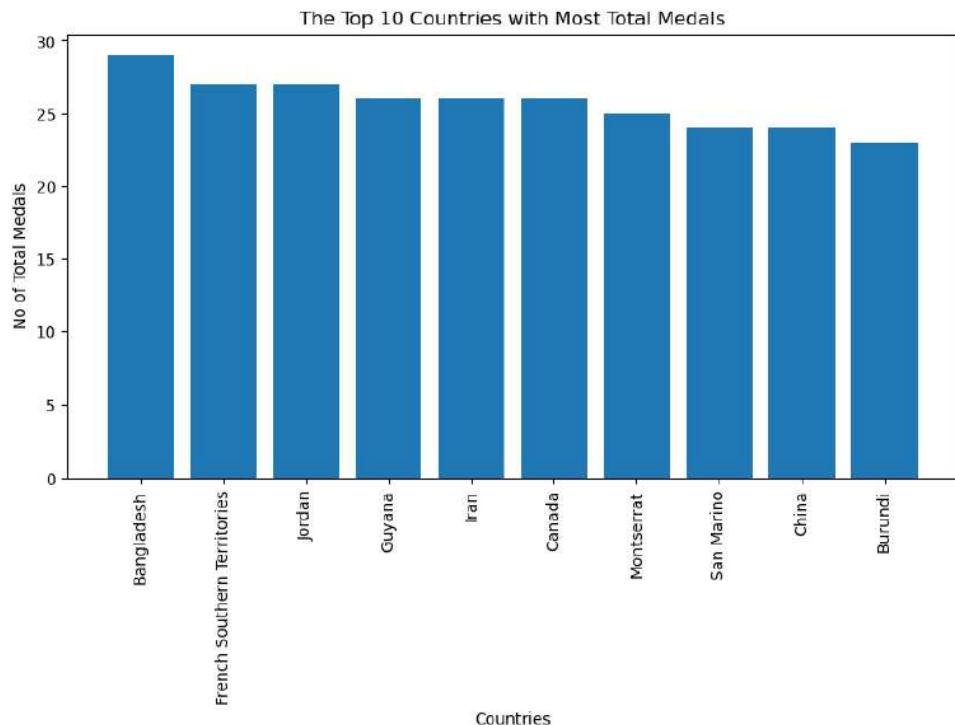
Query retrieves the top ten countries that have the highest no of total medals

```
In [4]: mycursor.execute('SELECT CountryID, Name, total_medal FROM country ORDER BY total_medal DESC LIMIT 10')
```

```
In [5]: r=mycursor.fetchall()
```

```
In [6]: country=[]  
total_medals=[]  
for x in r:  
    country.append(x[1])  
    total_medals.append(x[2])
```

```
In [7]: fig=plt.figure(figsize=(10,5))
plt.bar(country,total_medals)
plt.xlabel("Countries")
plt.ylabel("No of Total Medals")
plt.title("The Top 10 Countries with Most Total Medals")
plt.xticks(rotation=90)
plt.show()
```



```
In [8]: mycursor.close()
```

```
Out[8]: True
```

```
In [9]: mycursor=mydb.cursor()
```

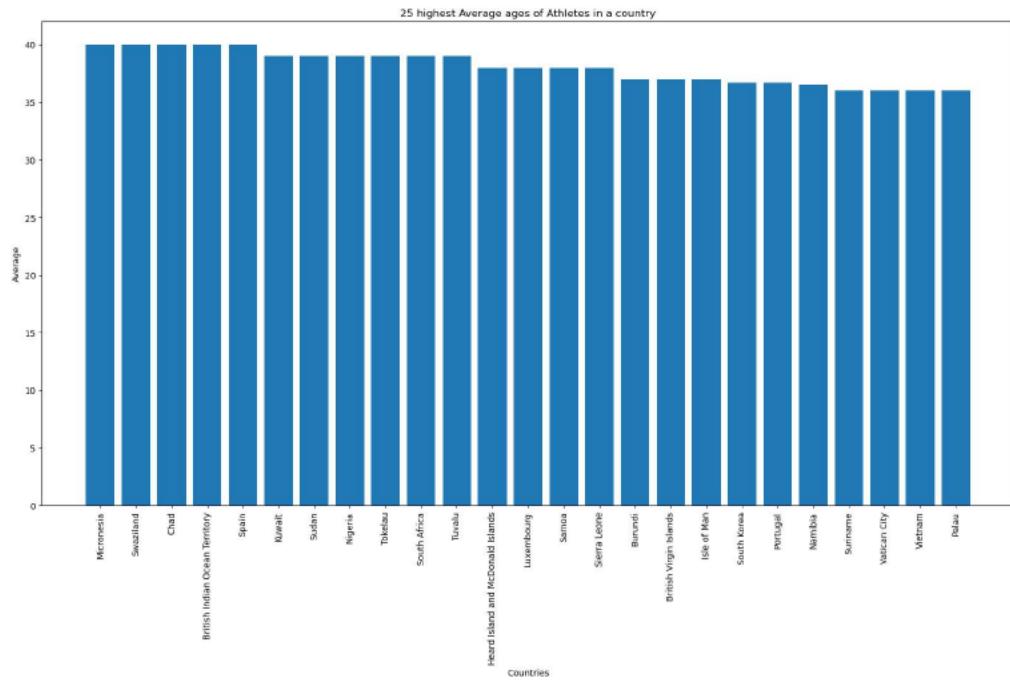
Query retries the Average age of the athletes of the countries with the 25 highest average ages

```
In [10]: mycursor.execute('SELECT c.Name AS CountryName, AVG(a.Age) FROM athlete a,cou
```

```
In [11]: r=mycursor.fetchall()
```

```
In [12]: country=[]
avg_age=[]
for x in r:
    country.append(x[0])
    avg_age.append(x[1])
```

```
In [13]: fig=plt.figure(figsize=(20,10))
plt.bar(country,avg_age)
plt.xlabel("Countries")
plt.ylabel("Average")
plt.title("25 highest Average ages of Athletes in a country")
plt.xticks(rotation=90)
plt.show()
```



```
In [14]: mycursor.close()
```

```
Out[14]: True
```

```
In [15]: mycursor=mydb.cursor()
```

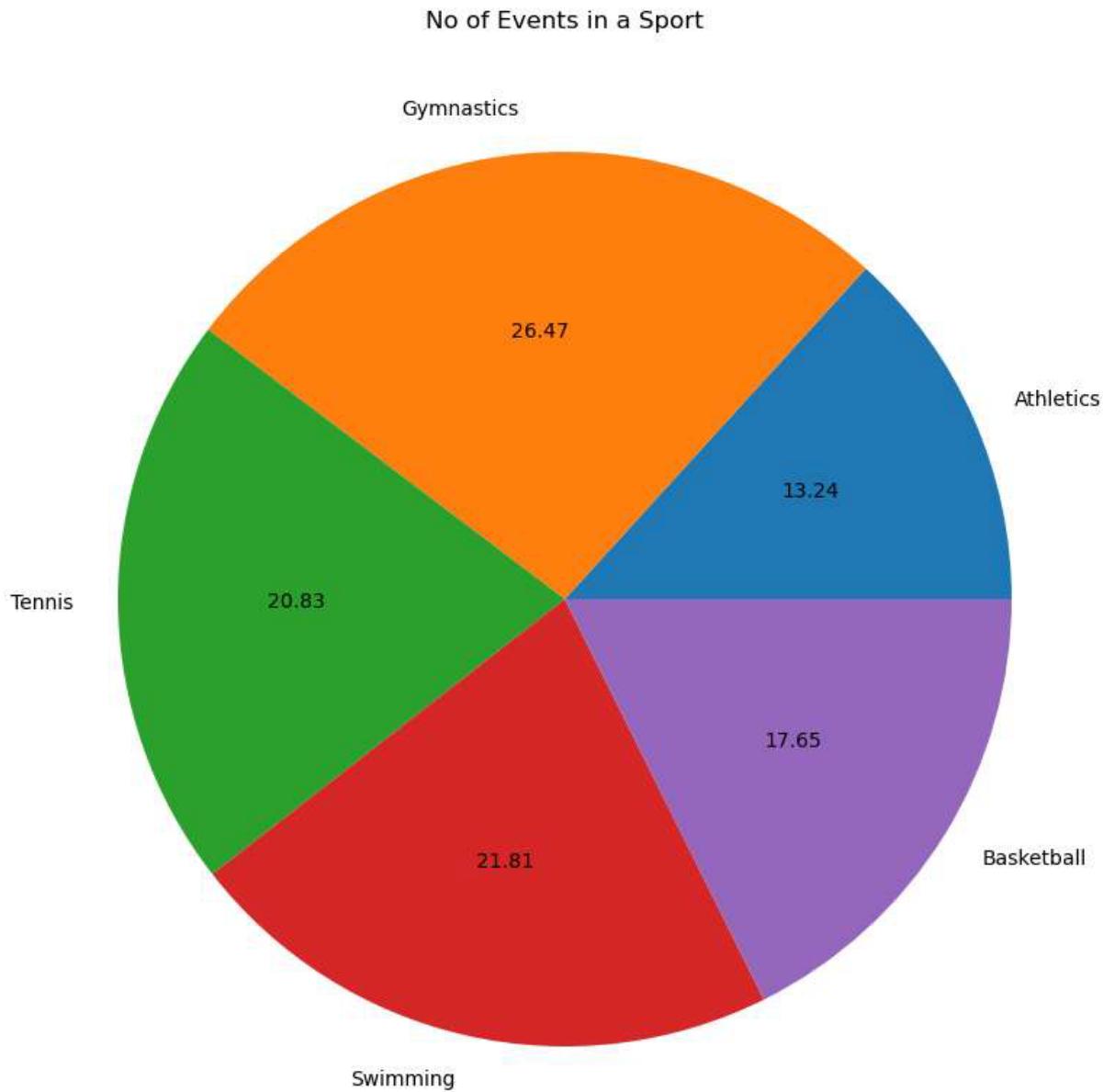
Query retrieves the Percentage of Events that belong to a particular sport

```
In [24]: , Participates WHERE Sport.SportID = Event.SportID AND Event.EventID = Participates.EventID GROUP BY Sport.SportID, Sport.Name;')
```

```
In [25]: r=mycursor.fetchall()
```

```
In [26]: sport=[]
no_event=[]
for x in r:
    sport.append(x[0])
    no_event.append(x[1])
```

```
In [27]: fig=plt.figure(figsize=(20,10))
plt.pie(no_event,labels=sport,autopct=".2f")
plt.title("No of Events in a Sport")
plt.show()
```



```
In [28]: mycursor.close()
```

```
Out[28]: True
```

```
In [29]: mydb.close()
```

As the queries are not clear in the Screenshots, we have included them below

- 1) SELECT CountryID, Name, total_medal FROM country ORDER BY total_medal DESC LIMIT 10;
- 2) SELECT c.Name AS CountryName, AVG(a.Age) FROM athlete a,country c WHERE a.CountryID = c.CountryID GROUP BY c.CountryID, c.Name order by avg(a.age) desc limit 25;
- 3) SELECT Sport.Name AS SportName, COUNT(DISTINCT Participates.CountryID) AS NumberOfCountries FROM Sport, Event, Participates WHERE Sport.SportID = Event.SportID AND Event.EventID = Participates.EventID GROUP BY Sport.SportID, Sport.Name;