# MySQL Exam Summer Olympics

*The Summer Olympic Games, often called the "Olympics," are a major international multi-sport event held every four years. Organised by the International Olympic Committee (IOC), they feature a wide range of sports and disciplines, drawing athletes from around the world to compete at the highest level.*

*The modern Summer Olympics were first held in 1896 in Athens, Greece. They were inspired by the ancient Olympic Games that took place in Olympia, Greece, from around 776 BC to 393 AD. Pierre de Coubertin revived the modern Games.*

*This task only affects individual sports with mixed participants (male/female). All athletes are fictional and have no relation to actual athletes.*

## Section 0: Database Overview

You have been given an Entity / Relationship Diagram of the Database:

A diagram of a sports program

Description automatically generated with medium confidence

The **Summer Olympics** **Database** needs to hold information about **countries**, **sports, disciplines**, **athletes**, **and medals**.

Your task is to create a database called **summer\_olympics**. Then you will have to create several **tables**.

* countries – contains information about the **countries**.
* sports – contains information about the sports.
* disciplines – contains information about the **disciplines**.
  + Each **discipline** has a **sport**.
* athletes – contains information about the **athletes**.
* Each **athlete** has a **country**.
* medals – contains information about the **medals**.
* disciplines\_athletes\_medals- a **many-to-many** **mapping** table between the **disciplines, athletes** and **medals**.
  + Each **athlete** has **disciplines** and **medals**. An athlete **can win a different** medal in **different** **disciplines**, but **not** multiple medals in the **same** discipline.
  + **No two athletes** can win the **same type of medal** *(e.g., gold, silver, bronze)* in the **same discipline**. **Only** **one** medal **of each type** can be awarded **per discipline**!

## Section 1: Data Definition Language (DDL) – 40 pts

Make sure you implement the whole database correctly on your local machine so that you can work with it.

The instructions you'll be given will be the minimum needed to implement the database.

### Table Design

You have been tasked to create the tables in the database by the following models:

#### countries

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| name | A **string** containing a maximum of **40 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**. UNIQUE** values. |

#### sports

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| name | A **string** containing a maximum of **20 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**. UNIQUE** values. |

#### disciplines

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| name | A **string** containing a maximum of **40 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**. UNIQUE** values. |
| sport\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table sports.  **NULL** is **NOT** permitted |

#### athletes

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| first\_name | A **string** containing a maximum of **40 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted. |
| last\_name | A **string** containing a maximum of **40 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted**.** |
| age | **Integer,** from **1** to **2,147,483,647.** | **NULL** is **NOT** permitted**.** |
| country\_id | **Integer,** from **1** to **2,147,483,647.** | Relationship with table countries.  **NULL** is **NOT** permitted |

#### medals

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| id | **Integer,** from **1** to **2,147,483,647.** | **Primary Key AUTO\_INCREMENT** |
| type | A **string** containing a maximum of **10 characters**. Unicode is **NOT** needed. | **NULL** is **NOT** permitted.  **UNIQUE** values. |

#### disciplines\_athletes\_medals

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| discipline\_id | **Integer**, from **1** to **2,147,483,647.** | Relationship with table disciplines.  **NULL** is **NOT** permitted. |
| athlete\_id | **Integer**, from **1** to **2,147,483,647.** | Relationship with table athletes.  **NULL** is **NOT** permitted |
| medal\_id | **Integer**, from **1** to **2,147,483,647.** | Relationship with table medals.  **NULL** is **NOT** permitted |

**Hint**: The combination of discipline and medal must be unique across all athletes.

Submit your solutions to the **Judge** for the first task. Submit **only** SQL **table** creation statements. **Do not** submit **CREATE SCHEMA** and **USE** statements.

You will also be given a data.sql file. It will contain a **dataset** with random data which you will need to **store** in your **local database**. This data will be given to you so you will not have to think of data and lose essential time in the process. The data is in the form of **INSERT** statement queries.

## Section 2: Data Manipulation Language (DML) – 30 pts

Here we need to do several manipulations in the database, like changing data, adding data, etc.

### Insert

You will have to **insert** records of data into the **athletes** table.

For all **athletes,** which comefrom a country whose name starts with the letter **"A"**, **insert data** in the **athletes** tablewith the **following values**:

• **first\_name** – set it to the first name of the athlete but **uppercase**.

• **last\_name** – set it to the athlete's last name followed by a white space and then **"comes from"** followed by one more white space and the name of the athlete's country.

e.g.(**last\_name**+" **comes from** "+*(country)***name**)

* **age** – set it to the sum of the **age** of the **athlete** and his **country\_id**.
* **country\_id** – keep the same value.

### Update

**Remove** the word **"weight"** from any **discipline** containing it.

### Delete

**Delete** all **athletes**, older than **35** years.

## Section 3: Querying – 50 pts

And now we need to do some data extraction. **Note** that the **example results** from **this section** use a **fresh database**. It is **highly recommended** that you **clear** the **database** that has been **manipulated** by the **previous problems** from the **DML** **section** and **insert again** the **dataset** you've been given, to ensure **maximum consistency** with the **examples** given in this section.

### Countries without athletes

Write a query that returns the **id** and **name** from table **countries**. **Filter** **only** the countries that **do not have** athletes at this Olympics. Show only the first **15 (fifteen)** results.

**Order** the results **descending** by country **name**.

#### Required Columns

* id (country)
* name (country)

#### Example

|  |  |
| --- | --- |
| **id** | **name** |
| 190 | Yemen |
| 189 | Virgin Islands, US |
| 188 | Virgin Islands, British |
| 187 | Vietnam |
| 186 | Venezuela |
| 185 | Vanuatu |
| 184 | Uzbekistan |
| 183 | Uruguay |
| 181 | United Republic of Tanzania |
| 180 | United Arab Emirates |
| 179 | Ukraine |
| 178 | Uganda |
| 177 | Tuvalu |
| 176 | Turkmenistan |
| 174 | Tunisia |

### Youngest medalists

Extract from the **summer\_olympics** database **full\_name** and **age** of the **two** athletes with the minimum age **who have won** at least one medal.

**Order** the results by **the athlete's id** in **ascending** order.

#### Required Columns

* full\_name (first\_name + " " + last\_name)
* age

#### Example

|  |  |
| --- | --- |
| **full\_name** | **age** |
| Melissa Olson | 17 |
| Nathan Cox | 17 |

### Athletes without medals

Not all athletes manage to reach the dream ladder of honour. Write a query that extracts (id, first\_name, last\_name) from the database **all** **athletes** who have **not won any medals**.

**Order** by **id** in **ascending** order.

#### Required Columns

* id (athlete)
* first\_name (athlete)
* last\_name (athlete)

#### Example

|  |  |  |
| --- | --- | --- |
| **id** | **first\_name** | **last\_name** |
| 18 | Courtney | Henry |
| 22 | Danielle | Dominguez |
| 24 | Darren | Foley |
| . . . | . . . | . . . |
| 54 | Michael | Clark |
| 58 | Regina | Ross |

### Athletes with medals divided by sports

Extract from the **summer\_olympics** database the **id, first\_name , last\_name, medals\_count** and the **sport** *(in which medals were won)*, **of the top ten athletes** who have won medals**.**

**Order** the results **descending** by **medals\_count**, then by **first\_name** in ascending order.

#### Required Columns

* id (athlete)
* first\_name (athlete)
* last\_name (athlete)
* medals\_count (total medals won by a single athlete)
* sport

#### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id** | **first\_name** | **last\_name** | **medals \_count** | **sport** |
| 20 | Daniel | King | 4 | Cycling |
| 41 | Jerry | Little | 3 | Fencing |
| 6 | Abigail | Foster | 2 | Karate |
| 2 | Adam | Flores | 2 | Athletics |
| 7 | Aidan | Hunt | 2 | Karate |
| 10 | Alice | Lopez | 2 | Swimming |
| 35 | Jenna | Cooper | 2 | Canoe |
| 48 | Kara | Fisher | 2 | Equestrian |
| 50 | Krystal | Garcia | 2 | Gymnastics |
| 59 | Regina | Stewart | 2 | Diving |

### Age groups of the athletes

From the database extract the **full\_name** and **age\_group** of the **athletes**. If the athlete is **18 years of age or younger** user must see **"Teenager",** if it **is between 18 (exclusive) and 25 years (inclusive) of age** it should display "**Young adult**". If it is **equal to or more than 26 of age** **->**"**Adult**".

**Order** the results **descending** by **athlete's age,** then by **first\_name** in ascending order.

#### Required Columns

* full\_name (first\_name + " " + last\_name)
* age\_group

#### Example

|  |  |
| --- | --- |
| **full\_name** | **age\_group** |
| James Soto | Adult |
| Joseph Cook | Adult |
| Alfred Gordon | Adult |
| Justin Howard | Adult |
| . . . | . . . |
| Melissa Olson | Teenager |
| Nathan Cox | Teenager |

## Section 4: Programmability – 30 pts

The time has come for you to prove that you can be a little more dynamic on the database. So, you will have to write several procedures.

### Find the total count of medals by country

Create a **user-defined function** with the name **udf\_total\_medals\_count\_by\_country (name VARCHAR(40))** that receives a **country\_name** and returns the **total number of medals** won by **all** athletes competing for that country.

#### Required Columns

* country\_name
* count\_of\_medals

#### Example

|  |  |
| --- | --- |
| **Query** | |
| SELECT c.name, udf\_total\_medals\_count\_by\_country ('Bahamas') as count\_of\_medals  FROM countries c  WHERE c.name = 'Bahamas' | |
| country\_name | count\_of\_medals |
| Bahamas | 6 |

### Update athlete's information

Create a **stored procedure** udp\_first\_name\_to\_upper\_case which accepts the following parameters:

* letter (CHAR(1))

Update the information in column **first\_name** from the athletes table by **capitalising** all the letters in it. **Do it only** for the athletes whose **first name** ends with the **given letter**. *(The given letter will always be lowercase)*

#### Result

|  |
| --- |
| **Query** |
| CALL **udp\_first\_name\_to\_upper\_case** ('s'); |
| This execution will update all the athletes whose first name ends with the letter 's'. |

#### Result

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id** | **first\_name ->** before | **last\_name** | **->** | **first\_name ->** after |
| 12 | Carlos | Alexander | -> | CARLOS |
| 26 | Dennis | Dickson | -> | DENNIS |
| 29 | James | Soto | -> | JAMES |
| 32 | James | Bryant | -> | JAMES |
| 65 | Travis | Buck | -> | TRAVIS |