

For creating columns in a table with different Data Types, we have specified datatype with its name.

**ID int:** ID will be name of column with ‘int’ data type, which will store integer values for each row.

**Name varchar (500):** Name column with varchar (500) will store variable character string with up to 500 characters.

**Sex varchar (10):** Sex column with 10 characters will store gender of athlete ‘M’ or ‘F’.

**Age int:** Integral column to store age of athletes.

**Height int:** An integer column to store height of athletes.

**Weight int:** An integer column to store weight of athletes.

**Team varchar (350):** Team column with character up to 350 to store Team name (i.e. Country Name)

**NOC varchar (100):** NOC (National Olympic Committees) with up to 100 characters, to store three letter country code.

**Games varchar (500):** Games column with characters up to 500 to store name of games played in Olympics.

**Year int:** Integer column to store Year when game took place.

* Create database sqlprjt1;

This command creates new database in MySQL Server with name sqlprjt1.

* Use sqlPrjt1;

Using this command a database(sqlPrjt1) is used and set to active database. Which stores table, views, procedure and other database objects.

This command creates table in database.

Which can be seen with command **“Show tables”**.



**show variables like "secure\_file\_priv";**

This command will return the directory path where MySQL is allowed to read and write files. It ensures only files this directory can be accessed for security.

**Load Data**: Use the LOAD DATA INFILE statement to load the data into your table.

"C:/ProgramData/MySQL/MySQL Server 9.0/Uploads/Athletes\_Transformed.csv" is the path of directory with file containing data (Athletes\_Transformed.csv).

* FIELDS TERMINATED BY ',': Specifies that fields in the file are separated by commas.
* ENCLOSED BY '"': Specifies that fields are enclosed by double quotes.
* LINES TERMINATED BY '\r\n': Specifies that lines are terminated by carriage return and newline characters.
* Ignore 1 rows: to ignore first row which (often) includes header roe in csv.

After this step our data will be loaded in our table which we created earlier with column specified with data present in our file. We also have Table Data Import Wizard which is handy to use for small datasets but for large data like this we have to use this method.

**Season varchar (400):** Season column up to 400 characters to stores season of games (e.g. summer, winter)

**City varchar (300):** City column up to 300 characters to store City name.

**Sport varchar (500):** Sport column up to 500 character to store Names of the sports.

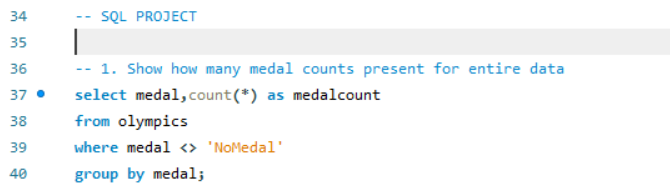
**Event varchar (400):** Event column up to 400 characters to store name of different event in Olympics of different sports.

**Medal varchar (200):**  Medal column to store medals won by athletes (Gold, Silver, Bronze) or NoMedal for if no medal is won.



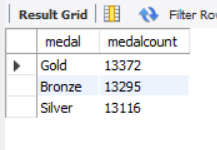
We can check table with ‘select \* from Olympics’ which will show Olympics table.

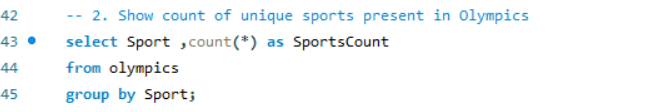
I have used **limit** to show only first 1000 rows.



This query is designed to count the number of medals (excluding ‘NoMedal’) for each type of medal in the olympics table.

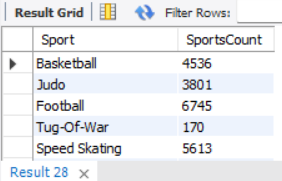
* 1. **Selects the Medal Type**: select medal
  2. **Counts the Number of Each Medal Type**: count (\*) as medalcount
  3. **From the Olympics Table**: from olympics
  4. **Excludes Entries with ‘NoMedal’**: where medal <> 'NoMedal'
  5. **Groups the Results by Medal Type**: group by medal

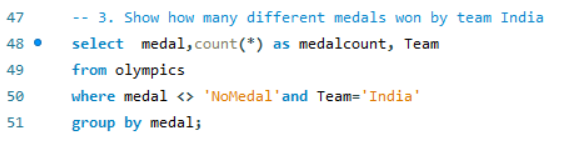




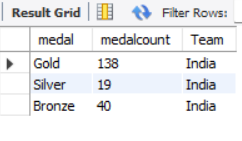
In this SQL query is we count number of occurrences for each sport in the olympics table. Here’s a breakdown of the query does:

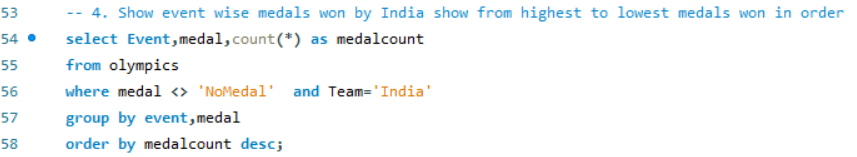
* **SELECT Sport, COUNT(\*) as SportsCount**: This selects the **Sport** column and counts the number of rows for each group, aliasing the count as **SportsCount**.
* **FROM olympics**: This specifies the table to query, which is **olympics**.
* **GROUP BY Sport**: This groups the results by the **Sport** column, allowing you to count the number of rows for each sport.



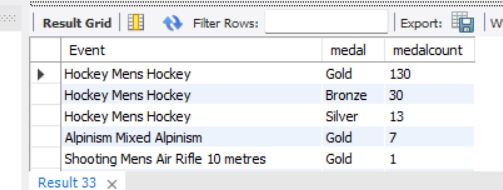


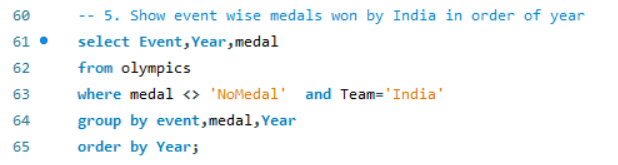
* **SELECT medal, COUNT (\*) as medalcount, Team**: This selects the **medal** and **Team** columns, and counts the number of rows for each group, aliasing the count as **medalcount**.
* **FROM olympics**: This specifies the table to query, which is **olympics**.
* **WHERE medal <> 'NoMedal' AND Team = 'India'**: This filters the results to only include rows where the **medal** is not 'NoMedal' and the **Team** is 'India'.
* **GROUP BY medal**: This groups the results by the **medal** column, allowing you to count the number of medals for each type of medal.





* **SELECT Event, medal, COUNT (\*) as medalcount**: This selects the **Event** and **medal** columns, and counts the number of rows for each group, aliasing the count as **medalcount**.
* **FROM olympics**: This specifies the table to query, which is **olympics**.
* **WHERE medal <> 'NoMedal' AND Team = 'India'**: This filters the results to only include rows where the **medal** is not 'NoMedal' and the **Team** is 'India'.
* **GROUP BY Event, medal**: This groups the results by both the **Event** and **medal** columns, allowing you to count the number of medals for each event and medal type.
* **ORDER BY medalcount DESC**: This sorts the results in descending order based on the **medalcount** column, so the events with the most medals will be shown first.



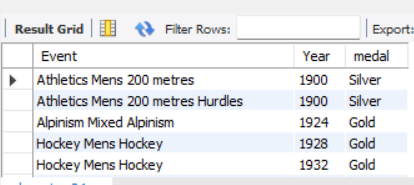


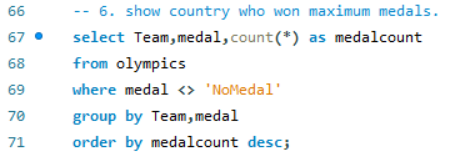
**Select specific columns**: The query selects three columns from the **olympics** table: **Event**, **Year**, and **medal**. This means that only these three columns will be included in the result set.

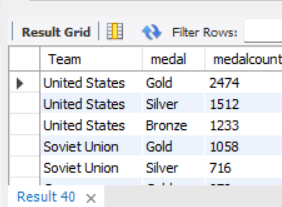
• **Filter out unwanted rows**: The query uses the **WHERE** clause to filter out rows that don't meet certain conditions. Specifically, it excludes rows where the **medal** column is equal to **'NoMedal'** and includes only rows where the **Team** column is equal to **'India'**.

• **Sort the result set**: The query uses the **ORDER BY** clause to sort the result set in ascending order by the **Year** column. This means that the earliest year will come first, and the latest year will come last.

• **Retrieve the result set**: The query retrieves the result set from the **olympics** table, which includes only the rows that meet the conditions specified in the **WHERE** clause and are sorted in the order specified by the **ORDER BY** clause.

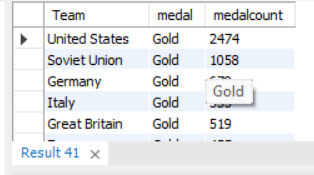


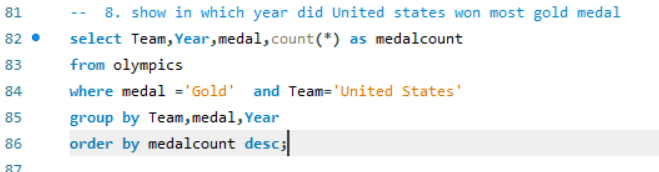


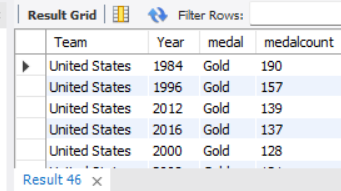
* **Selects the Team and Medal Type**: SELECT Team, medal
* **Counts the Number of Each Medal Type for Each Team**: COUNT (\*) AS medalcount
* **From the Olympics Table**: FROM olympics
* **Excludes Entries with ‘NoMedal’**: WHERE medal <> 'NoMedal'
* **Groups the Results by Team and Medal Type**: GROUP BY Team, medal
* **Orders the Results by Medal Count in Descending Order**: ORDER BY medalcount DESC

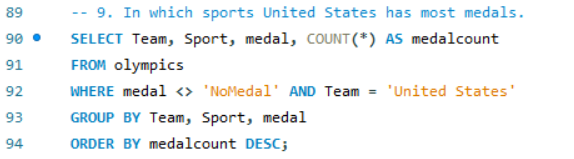


* Select specific columns: The query selects the Team, medal, and the count of gold medals (medalcount).
* Filter out unwanted rows: The WHERE clause includes only rows where the medal is 'Gold'.
* Group the results: The GROUP BY clause groups the results by Team and medal.
* Sort the result set: The ORDER BY clause sorts the result set in descending order by medalcount.
* Limit the result set: The LIMIT clause limits the result set to the top 10 rows.

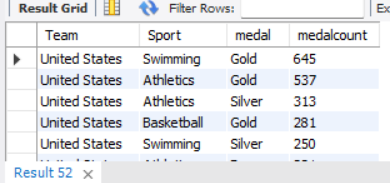


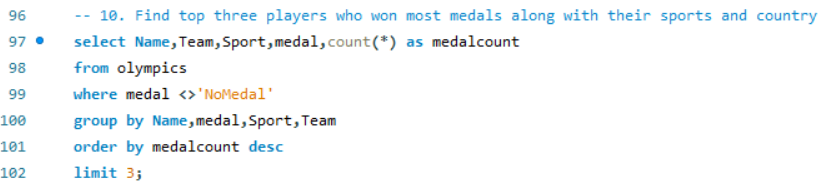


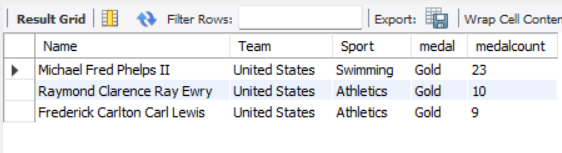
* **SELECT Team, Year, COUNT (\*) AS medalcount**: Selects the team, year, and counts the number of gold medals for each year.
* **FROM olympics**: Specifies the table to query from.
* **WHERE medal = 'Gold' AND Team = 'United States'**: Filters the records to include only gold medals won by the United States.
* **GROUP BY Team, Year**: Groups the results by team and year so that COUNT (\*) can aggregate the number of medals per year.
* **ORDER BY Year**: Orders the results by year, which is usually more intuitive to see the progression over time.

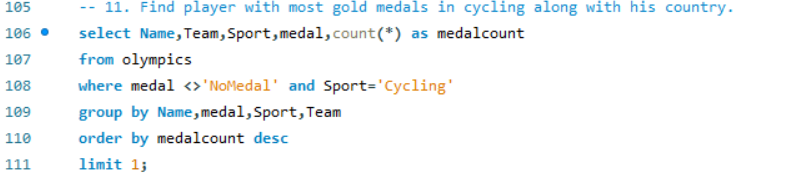


* **SELECT Team, Sport, medal, COUNT (\*) AS medalcount**: Selects the team, sport, type of medal, and counts the number of medals for each combination.
* **FROM olympics**: Specifies the table to query from.
* **WHERE medal <> 'NoMedal' AND Team = 'United States'**: Filters the records to include only those where the medal is not 'NoMedal' and the team is 'United States'.
* **GROUP BY Team, Sport, medal**: Groups the results by team, sport, and medal type so that COUNT (\*) aggregates the number of medals for each combination.
* **ORDER BY medalcount DESC**: Orders the results by the count of medals in descending order.



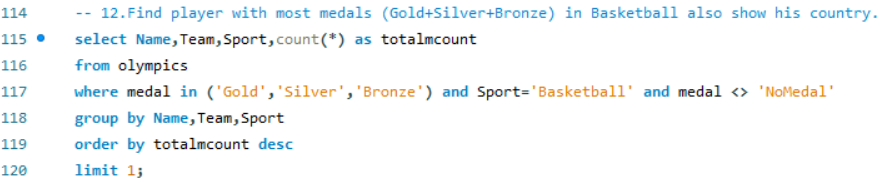


* **SELECT Name, Team, Sport, medal, COUNT (\*) AS medalcount**: Selects the athlete's name, team, sport, type of medal, and counts the number of medals won.
* **FROM olympics**: Specifies the table to query from.
* **WHERE medal <> 'NoMedal'**: Filters out records where the medal is 'NoMedal', focusing only on actual medal wins.
* **GROUP BY Name, medal, Sport, Team**: Groups the results by athlete name, medal type, sport, and team so that COUNT (\*) can aggregate the number of medals per grouping.
* **ORDER BY medalcount DESC**: Orders the results by the number of medals won, in descending order.
* **LIMIT 3**: Limits the results to the top 3 rows based on the highest medal counts.

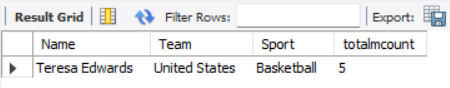


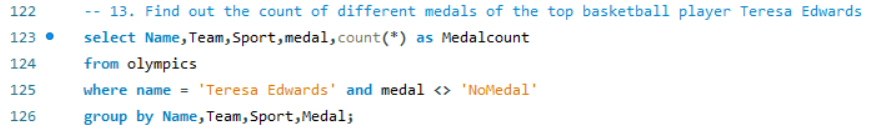
* **SELECT Name, Team, Sport, medal, COUNT (\*) AS medalcount**: Retrieves the athlete's name, team, sport, type of medal, and counts the number of medals won.
* **FROM olympics**: Specifies the table to query from.
* **WHERE medal <> 'NoMedal' AND Sport = 'Cycling'**: Filters records to include only those where the medal is not 'NoMedal' and the sport is 'Cycling'.
* **GROUP BY Name, medal, Sport, Team**: Groups the results by athlete name, medal type, sport, and team to count the number of medals for each grouping.
* **ORDER BY medalcount DESC**: Orders the results by the count of medals in descending order.
* **LIMIT 1**: Limits the results to the top 1 row, giving the athlete with the highest number of medals in the specified sport.



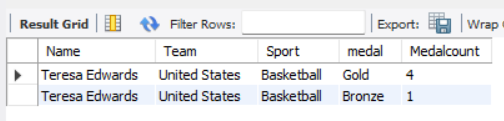


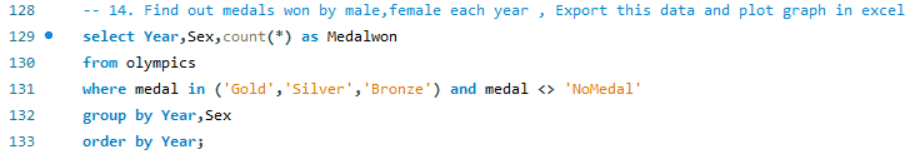
* **SELECT Name, Team, Sport, COUNT (\*) AS totalmcount**: Selects the athlete's name, team, sport, and counts the number of medals.
* **FROM olympics**: Specifies the table to query from.
* **WHERE medal IN ('Gold', 'Silver', 'Bronze')**: Filters the records to include only those where the medal is either 'Gold', 'Silver', or 'Bronze'.
* **AND Sport = 'Basketball'**: Further filters the results to include only records where the sport is 'Basketball'.
* **AND medal <> 'NoMedal'**: Excludes records where the medal is 'NoMedal'.
* **GROUP BY Name, Team, Sport**: Groups the results by athlete name, team, and sport to aggregate the number of medals.
* **ORDER BY totalmcount DESC**: Orders the results by the count of medals in descending order.
* **LIMIT 1**: Limits the results to the top 1 row, which will be the athlete with the highest number of medals.





* **SELECT Name, Team, Sport, medal, COUNT (\*) AS Medalcount**: Selects the name of the athlete, their team, sport, medal type, and counts the number of each type of medal.
* **FROM olympics**: Specifies the table to query from.
* **WHERE Name = 'Teresa Edwards' AND medal <> 'NoMedal'**: Filters the records to include only those where the athlete's name is 'Teresa Edwards' and the medal is not 'NoMedal'.
* **GROUP BY Name, Team, Sport, medal**: Groups the results by athlete name, team, sport, and medal type to aggregate the number of medals.





* **SELECT Year, Sex, COUNT (\*) AS Medalwon**: Selects the year, sex, and counts the number of medals.
* **FROM olympics**: Specifies the table to query from.
* **WHERE medal IN ('Gold', 'Silver', 'Bronze')**: Filters records to include only those with medals of 'Gold', 'Silver', or 'Bronze'.
* **AND medal <> 'NoMedal'**: Ensures 'NoMedal' entries are excluded.
* **GROUP BY Year, Sex**: Groups the results by year and sex to count the number of medals per combination.
* **ORDER BY Year**: Orders the results by year to see the progression over time.

