*SQL PROJECT*

*ON*

*OLYMPIC HISTORY DATABASE*

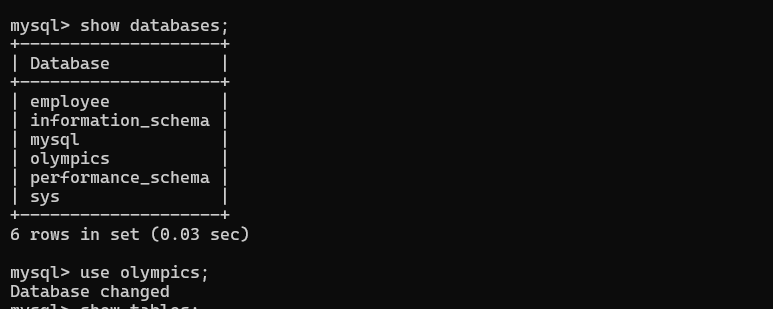
*BY: DIVYA SRI. B*

1. SHOW DATABASES AND USE DATABASES

QUERY:

1. Show Databases;
2. Use Databases;

OUTPUT:



1. SHOW TABLES:

QUERY:

Show tables;

OUTPUT:



1. CREATE TABLES

CREATE TABLE OLYMPIC\_HISTORY:

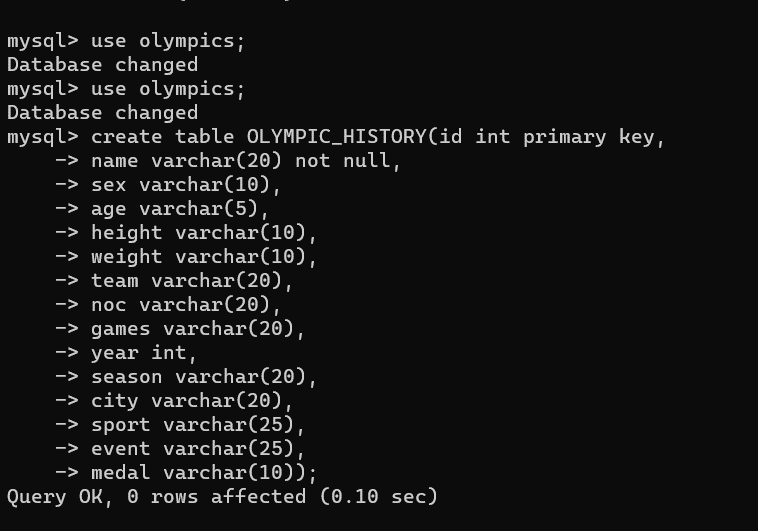
QUERY:

Create table Olympic\_history(id int primary key,

name varchar (20) not null, sex varchar (10), age varchar (5),

height varchar (10), weight varchar (10), team varchar (20),

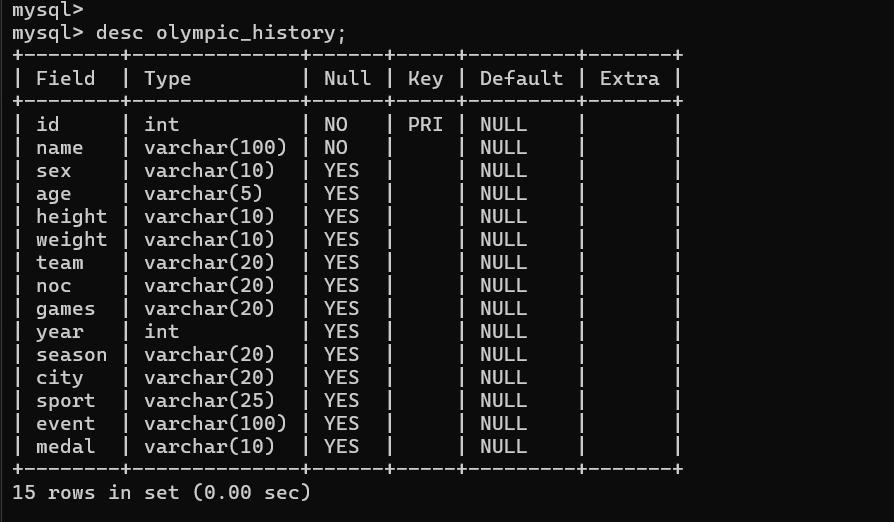
noc varchar (20), games varchar (20), year int, season varchar (20), city varchar (20), sports varchar (25), event varchar (25), medal varchar (10));

OUTPUT:

TO VIEW TABLE:

desc Olympic\_ history;

OUTPUT:



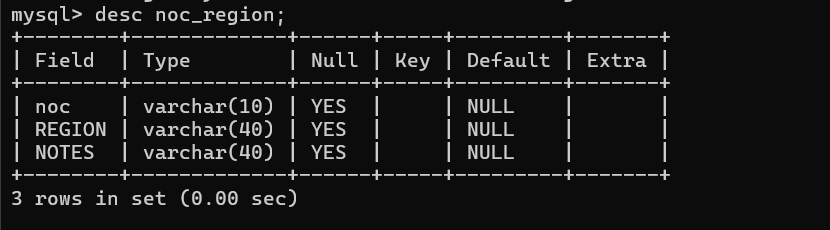
CREATE TABLE NOC\_REGION:

QUERY:

Create table noc\_region (noc varchar (10), region varchar (40), notes varchar (40));

To view table:

desc noc\_region;



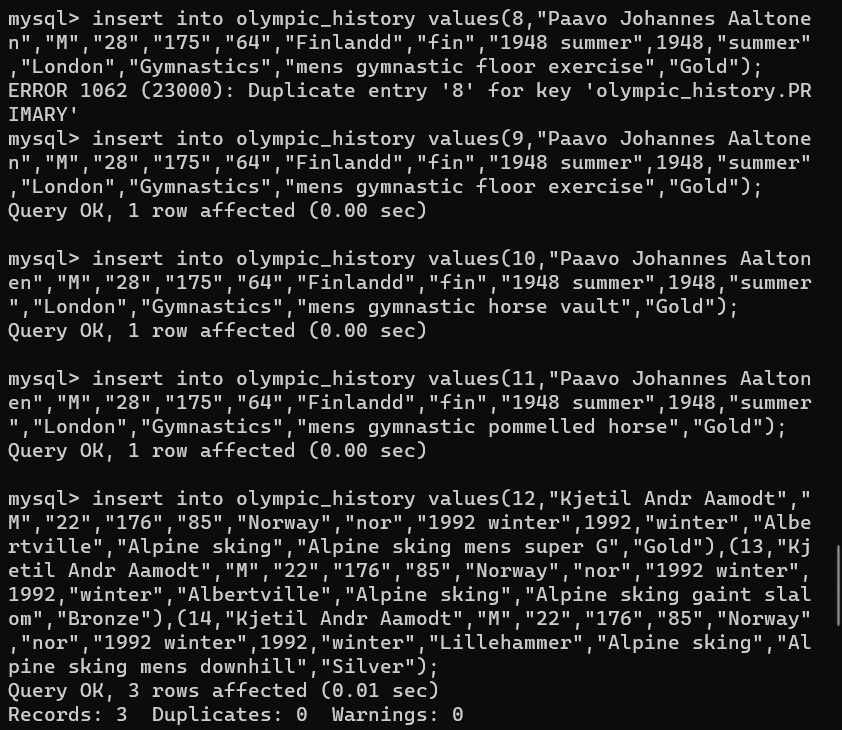
4.INSERT THE VALUES OR DATAS

QUERY FOR TABLE(OLYMPIC\_HISTORY):

Syntax:

Insert into table\_ name values (column1 datatype1, column2 datatype2, column3 datatype3….);

OUTPUT:



QUERY FOR TABLE(NOC\_REGION):

Insert into noc\_region values (column1 datatype1, column2 datatype2, column3 datatype3….);

OUTPUT:



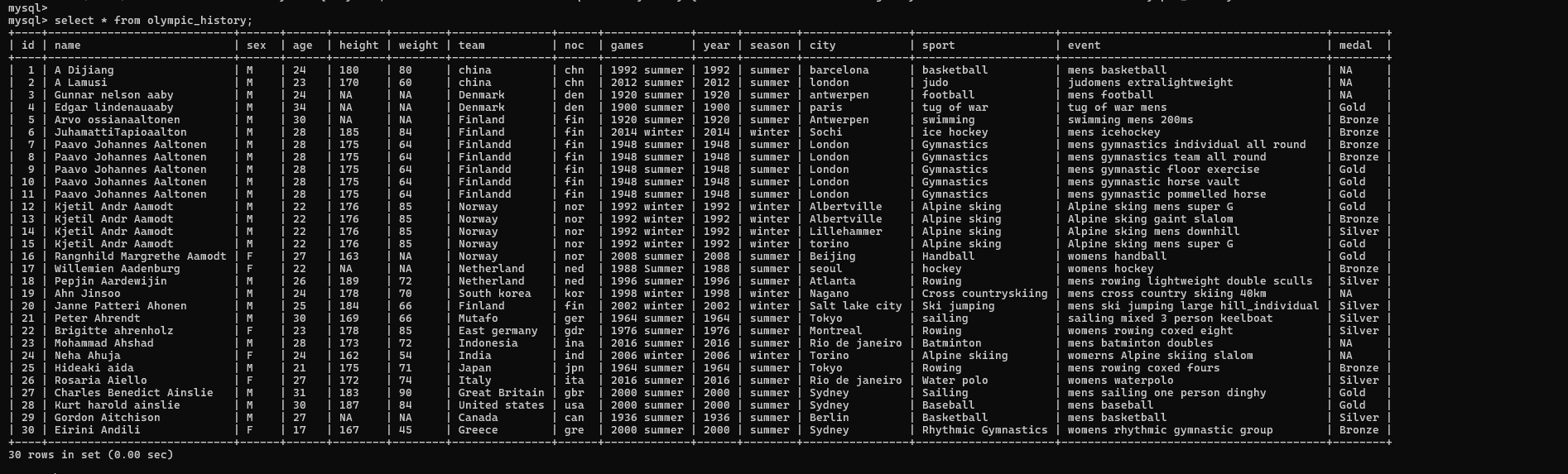
5.TO VIEW THE INSERTED TABLE WITH VALUES

QUERY:

Here we use Select query.

Select \* from Olympic\_history;

Output:

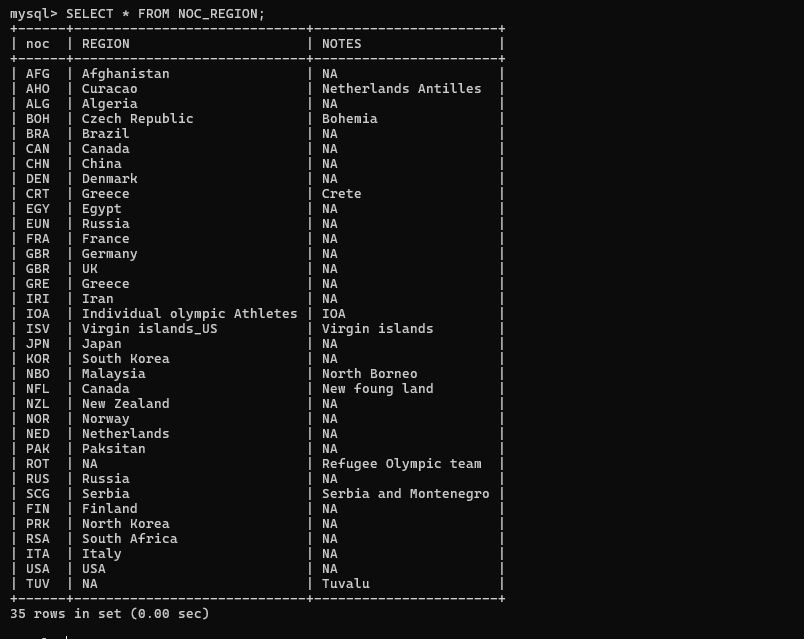


To view noc\_region table:

QUERY:

Select \* from noc\_region;

OUTPUT:



1. DDL & DML COMMANDS:

DDL (Data Definition Language):

1. Create
2. Drop
3. Truncate
4. Alter

DML (Data Manipulating Language):

1. Update
2. Insert
3. Delete

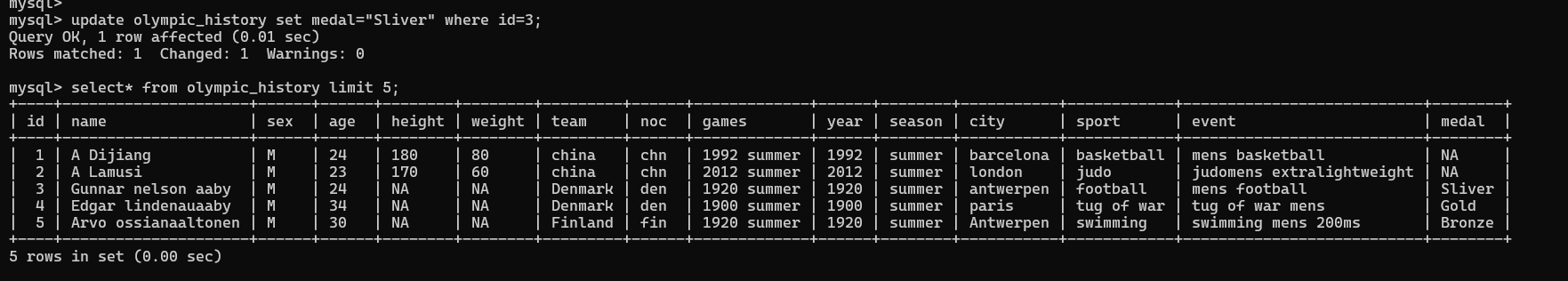
DQL (Data Query Language):

1. Select

Update command:

* The Update statement allows you to modify existing records in the table based on specified conditions.

QUERY:

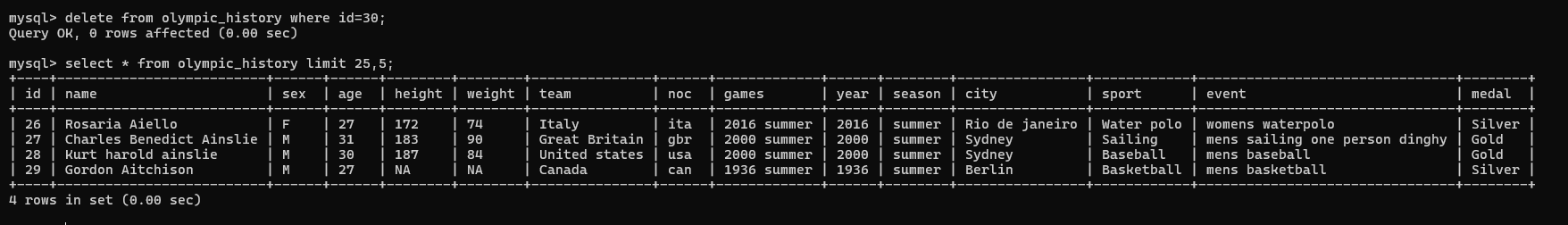
 Update Olympic \_history set medal=” Sliver” where id=3;

Delete command:

* The Delete command in SQL is used to remove rows from a table based on specified conditions.

QUERY:

Delete from olympic\_history where id=30;

OUTPUT:

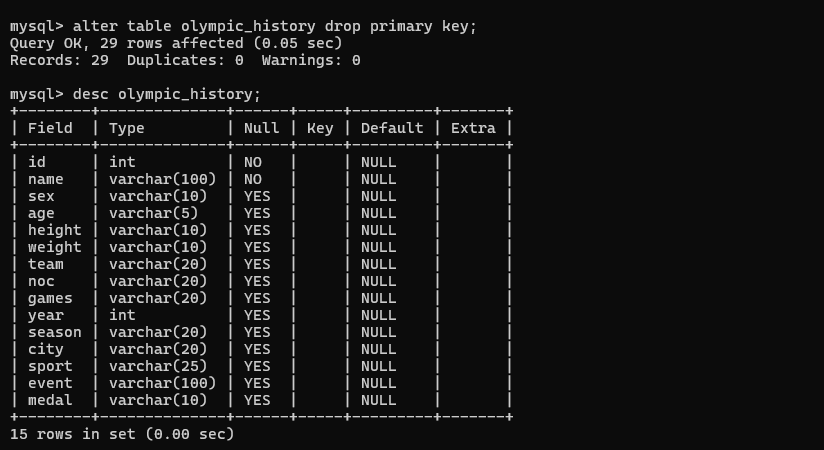
Alter command/ Alter Constraints:

* The Alter command in SQL is used to modify an existing database object, such as a table, to add, modify, or remove columns, constraints or indexes.

QUERY: To drop a constraint (primary key)

Alter table Olympic\_history drop primary key;

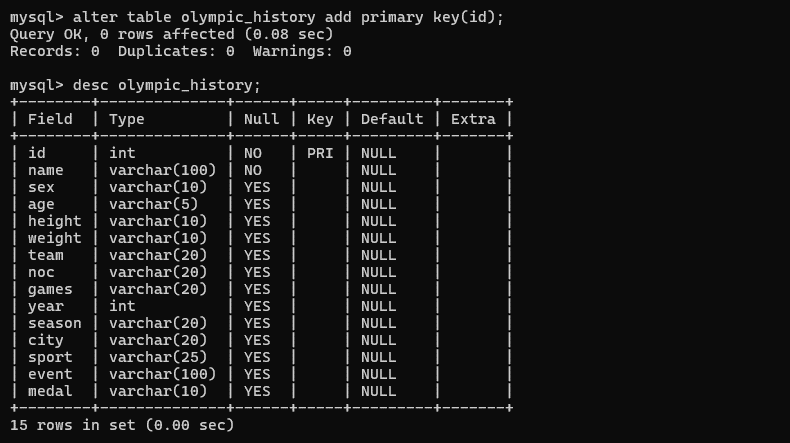
Output:



QUERY: To add a constraint (primary key)

Alter table Olympic\_history add primary key(id);

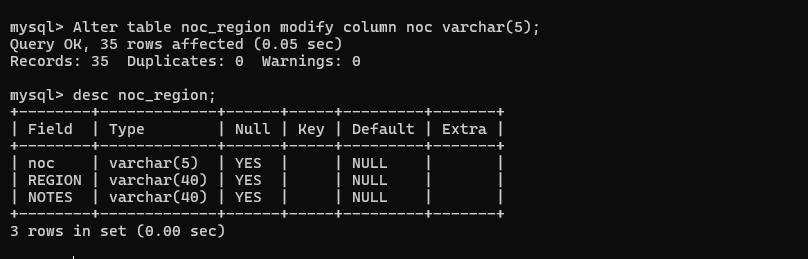
OUTPUT:



QUERY: To modify column using Alter

Alter table noc\_region modify column noc varchar(5);

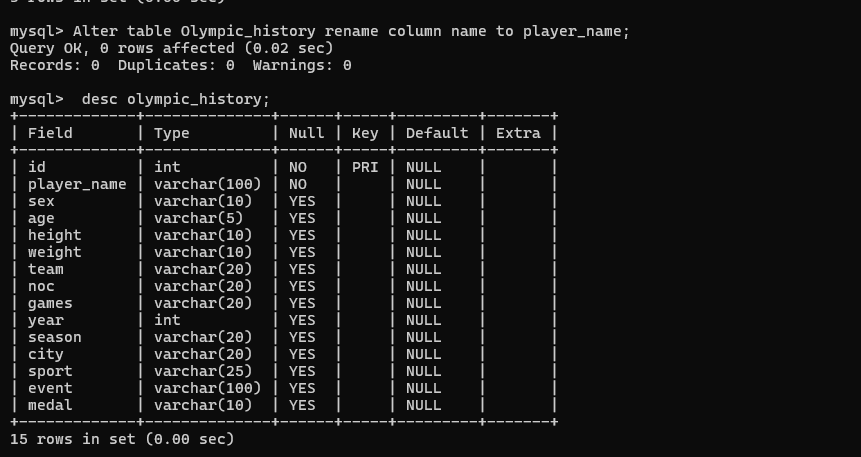
OUPUT:



Query: To rename column using alter

Alter table Olympic\_history rename column name to player\_name;

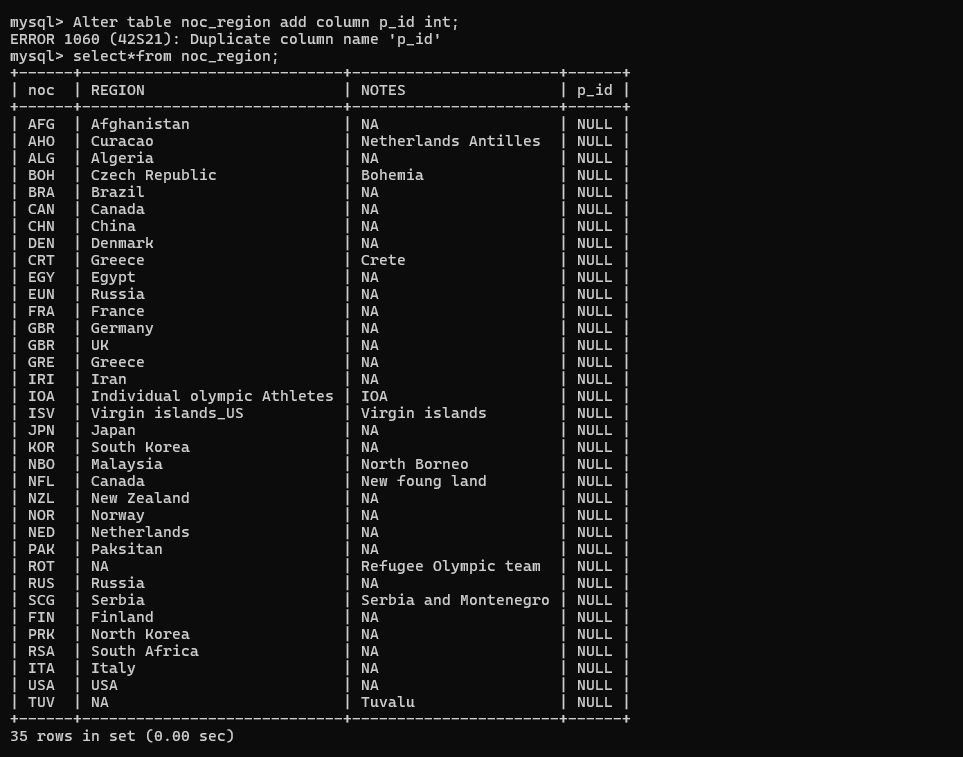
OUTPUT:



QUERY: To Add column using Alter

Alter table noc\_region add column p\_id;

OUTPUT:



Query: To drop a column using Alter

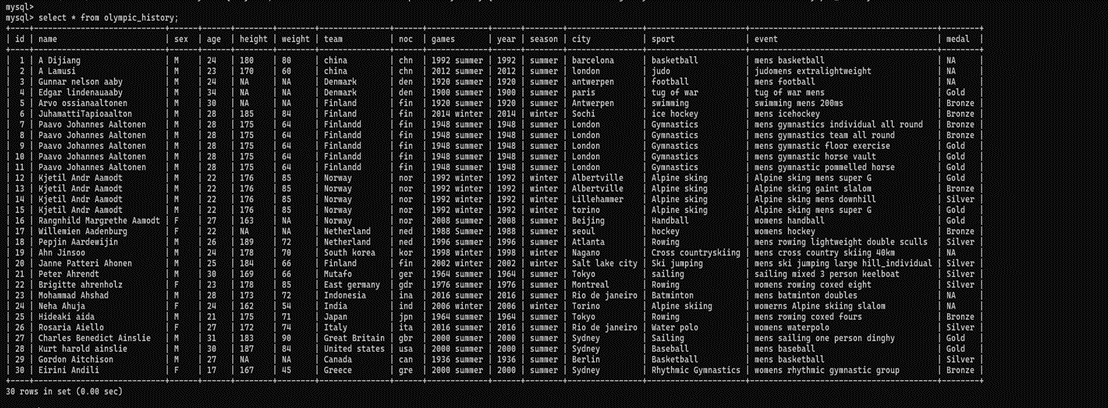
Alter table noc\_region drop column p\_id;

OUTPUT:



7.SELECT COMMAND

1. SELECT ALL COLUMNS FROM TABLE

QUERY: Select \* from Olympic\_ history;

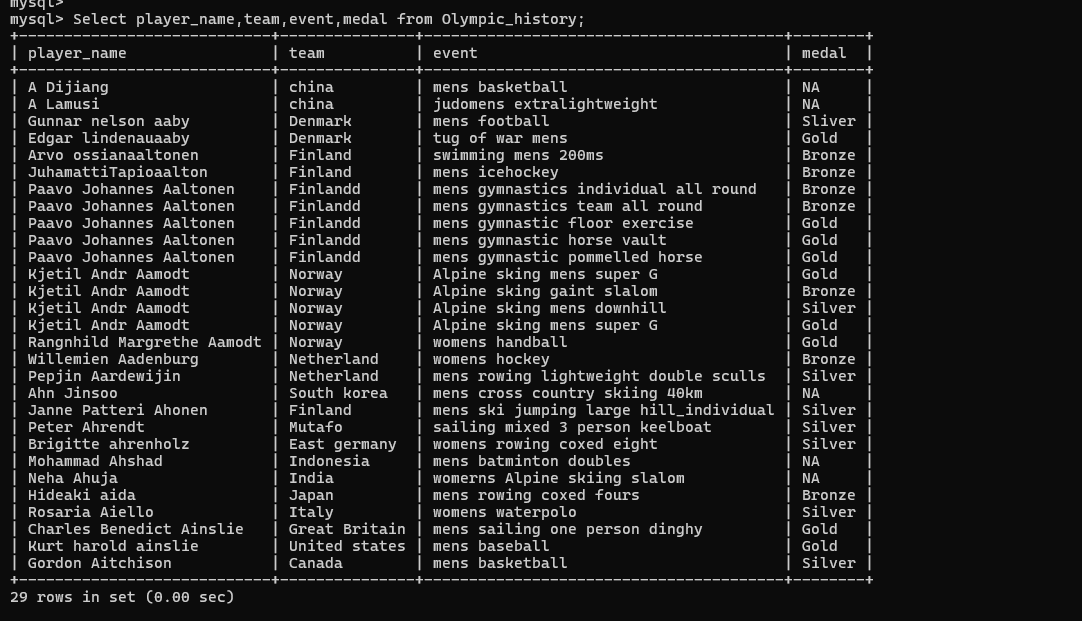
OUTPUT:

1. SELECT PARTICULAR COLUMN FROM TABLE

QUERY:

Select player\_ name, team, event, medal from Olympic\_ history;

OUTPUT:

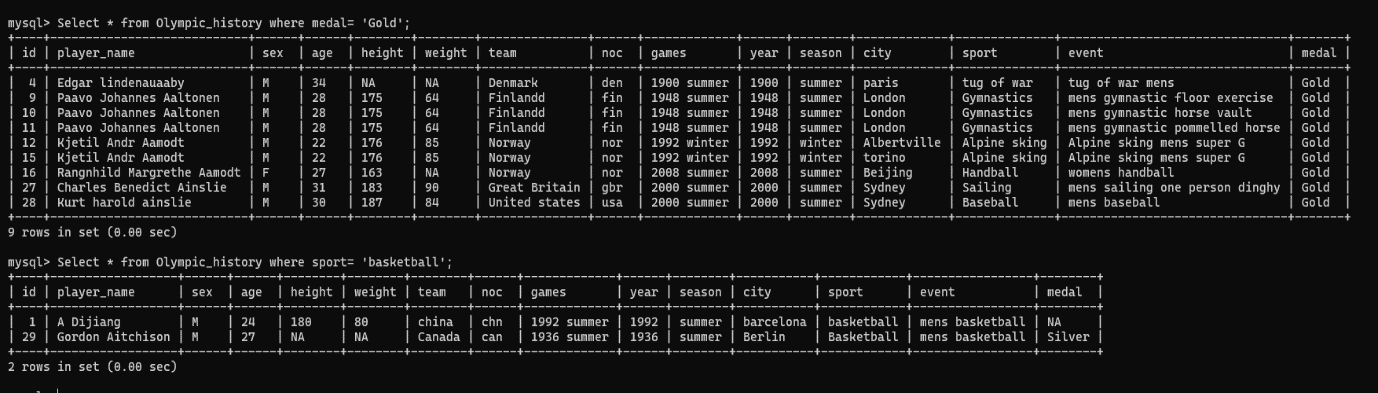


1. SELECT WITH CONDITION:

QUERY:

Select \* from Olympic\_ history where medal= ’Gold’;

Select\* from Olympic\_ history where sport= ’basketball’;

OUTPUT:

QUERY:

Select \*from Olympic\_ history where age< ‘25’;

OUTPUT:

1. SELECT USING AND PERATOR

QUERY:

Select \* from Olympic\_history where sex =’M’ AND age>’28’;

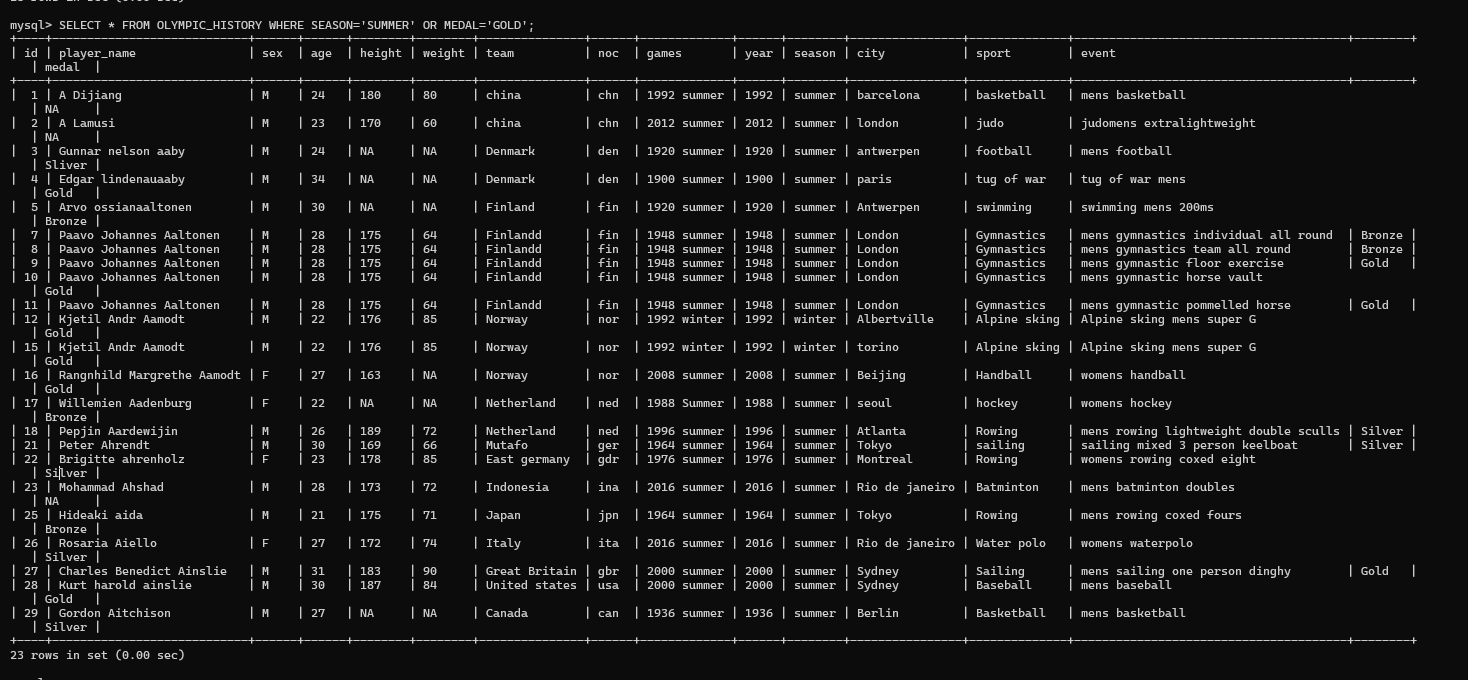
OUTPUT:

1. SELECT SING OR OPERTOR

QUERY:

Select \* from Olympic\_history where season=’summer’ OR medal=’Gold’;

OUTPUT:

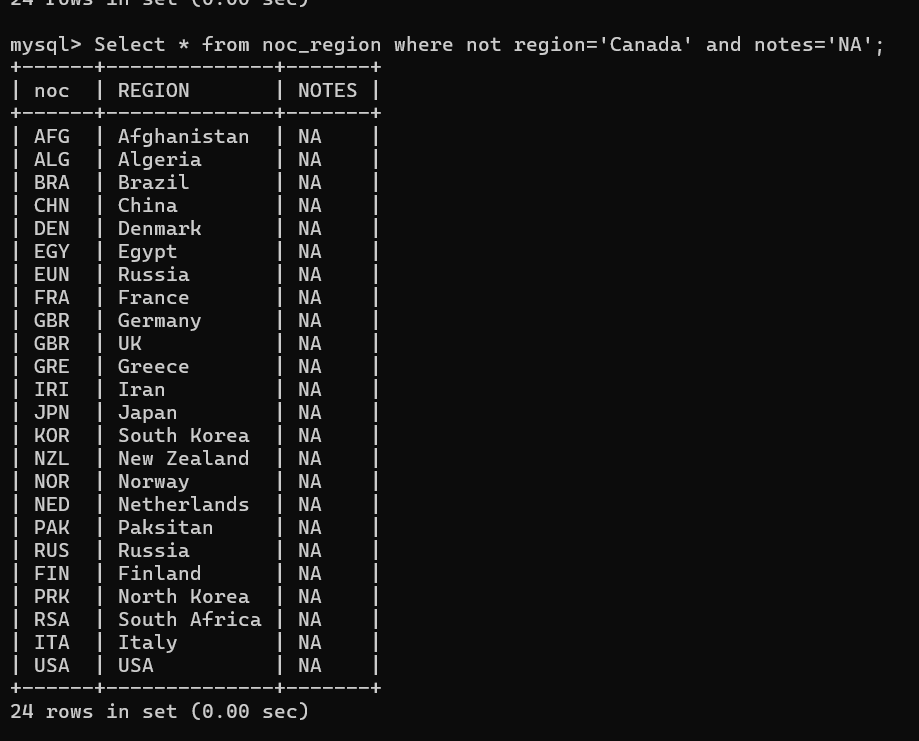


1. SELECT USING NOT OPERATORS

QUERY:

SELECT \* FROM NOC\_REGION WHERE NOT REGION=’CANADA’ AND NOTES=’NA’;

OUTPUT:



1. SELECT USING RANGE OPPERATOR(BETWEEN)

QUERY:

Select \* from Olympic\_history where age between ‘25’ and ‘35’ limit 5;

OUTPUT:



1. SELECT USING LIST OPERATOR

QUERY:

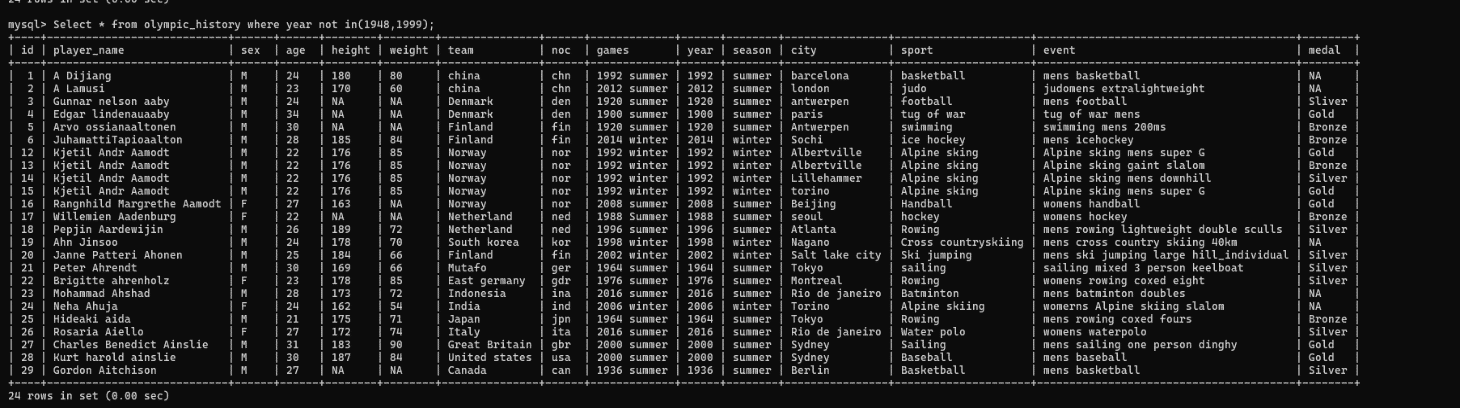
Select \* from olympic\_history where year in(1948,1999);

OUTPUT:



QUERY:

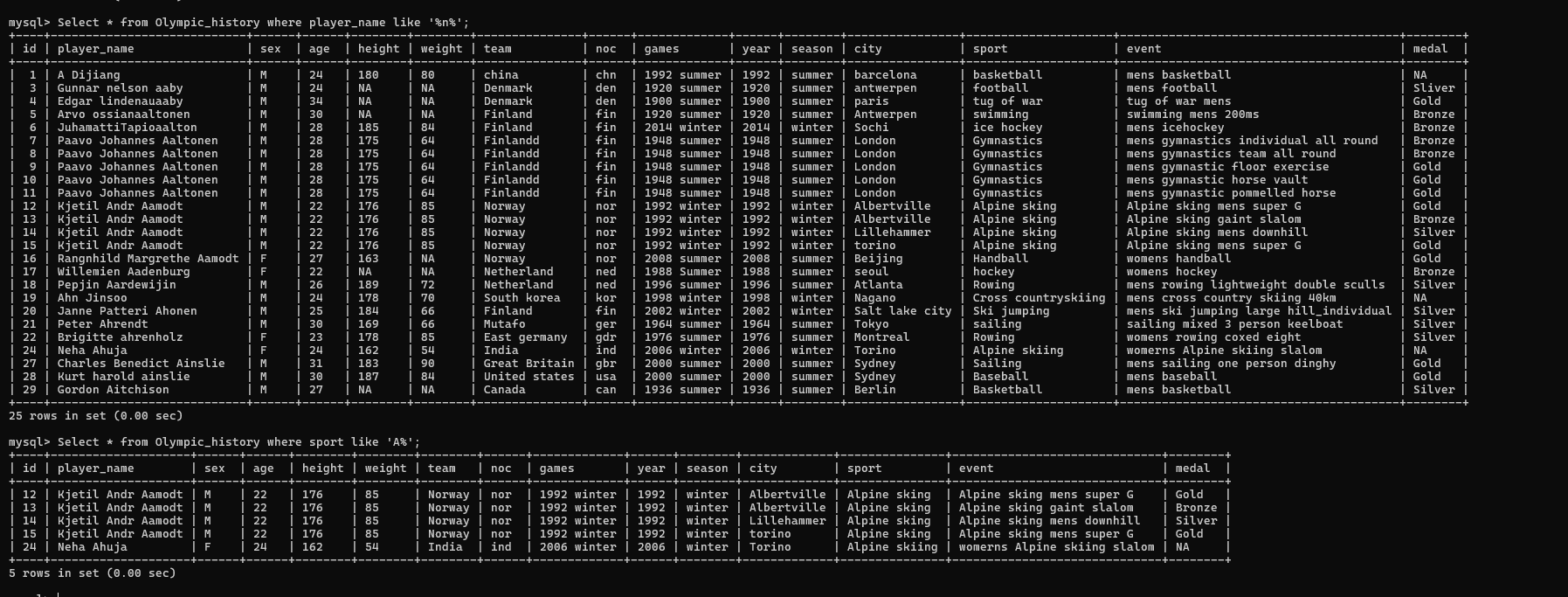
Select \* from Olympic\_history where year not in (1948,1999);

OUTPUT:

1. SELECT USING LIKE OPERATOR

QUERY:

Select \* from Olympic\_history where player\_name like “%a”;

OUTPUT:

1. SELECT COMMAND USING DISTINCT

QUERY:

Select distinct notes from noc\_ region;

Select distinct sport from Olympic\_ history;

OUTPUT:

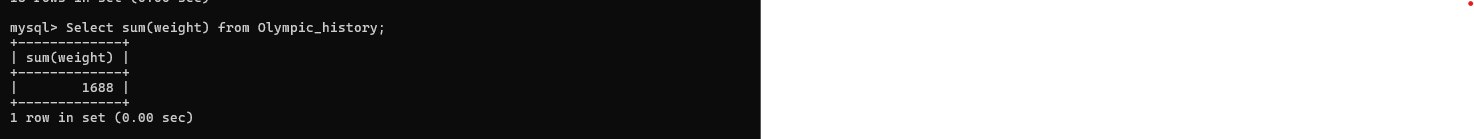


1. AGGREGATE FUNCTIONS
2. SUM
3. AVERAGE
4. COUNT
5. MIN
6. MAX
7. SUM

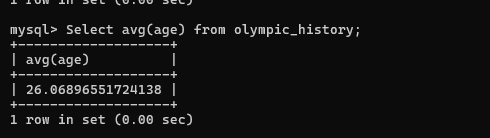
QUERY:

Select sum(weight) from Olympic\_ history;

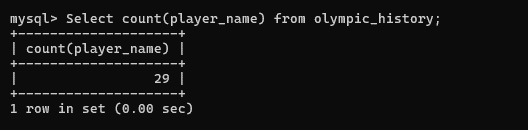
OUTPUT:



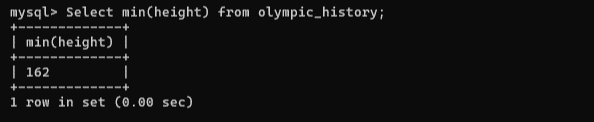
1. AVERAGE



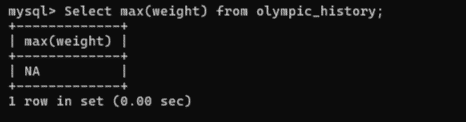
1. COUNT



1. MIN



1. MAX



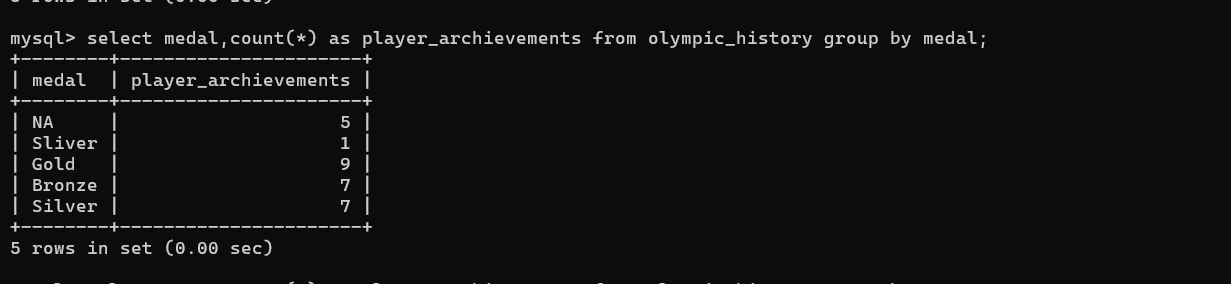
1. GROUP BY

* The GROUP BY operations is used in SQL to group rows from a table based on the values in one or more columns.

QUERY:

Select medal, count(\*) as player\_ achievements from Olympic\_ history group by medal;

OUTPUT:



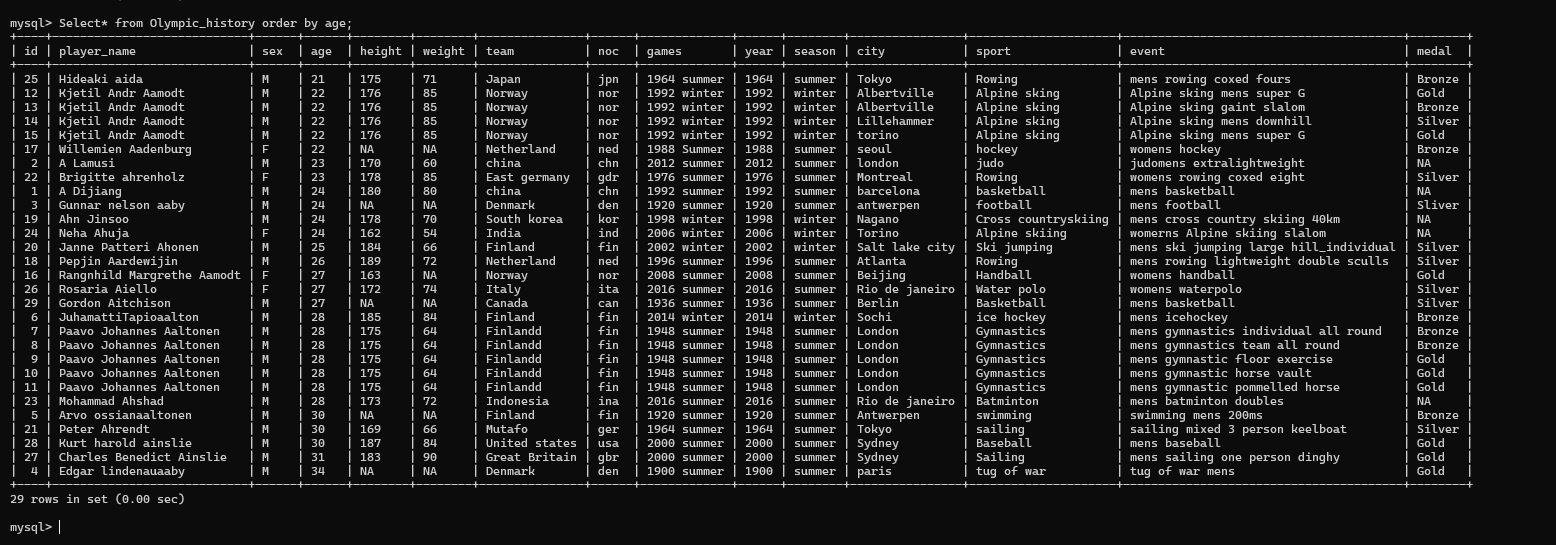
1. ORDER BY

* The ORDER BY Clause is used to sort the result set based on one or more columns in either ascending or descending order.

QUERY:

Select\* from Olympic\_ history order by age;

OUTPUT:



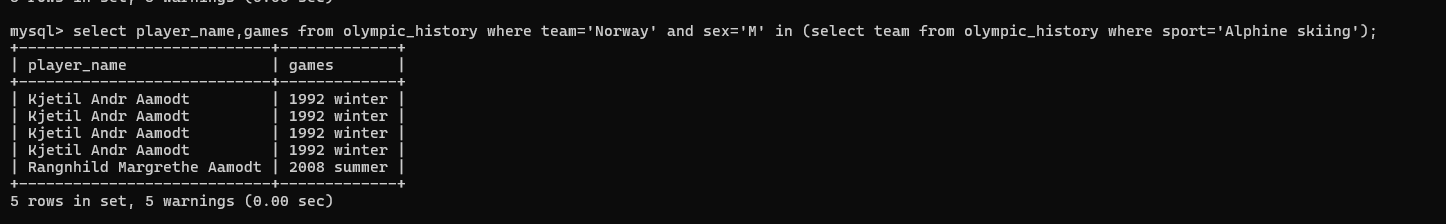
1. SUB QUERIES

* Subqueries can be used to perform complex queries and retrieve data based on the results of inner queries.

Query:

Select player\_ name, games from Olympic\_ history where team='Norway' and sex='M' in (select team from Olympic\_ history where sport='Alphine skiing');

OUTPUT:

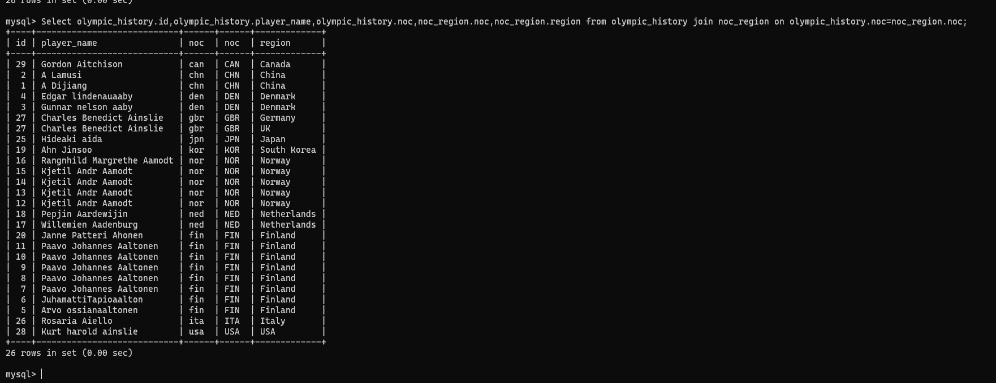


1. Joins

* Joins are used to combine data from multiple tables based on related columns.
* Joins helps to retrieve information that is distributed across the different tables in a more meaningful and consolidated manner.

QUERY:

Select Olympic\_history. id, olympic\_history. player\_ name, Olympic\_ history. noc, noc\_region. noc, noc\_region. region from Olympic\_ history join noc\_region on Olympic\_ history. noc = noc\_ region.noc;

OUTPUT:

THANK YOU