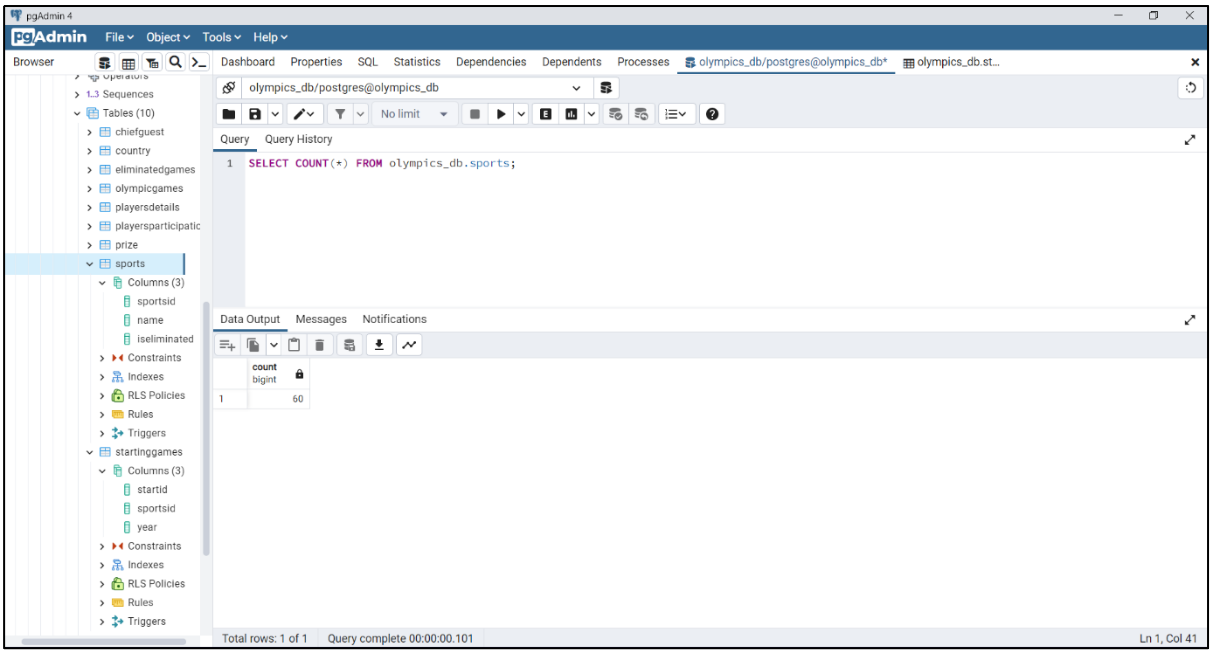
1. Select the total number of sports:

→ SELECT COUNT(*) FROM olympics_db.sports;
→ π_COUNT(*) (σ_sports(olympics_db))



```
SELECT COUNT(*) FROM olympics_db.sports;
```

```
count
bigint
1    60
```

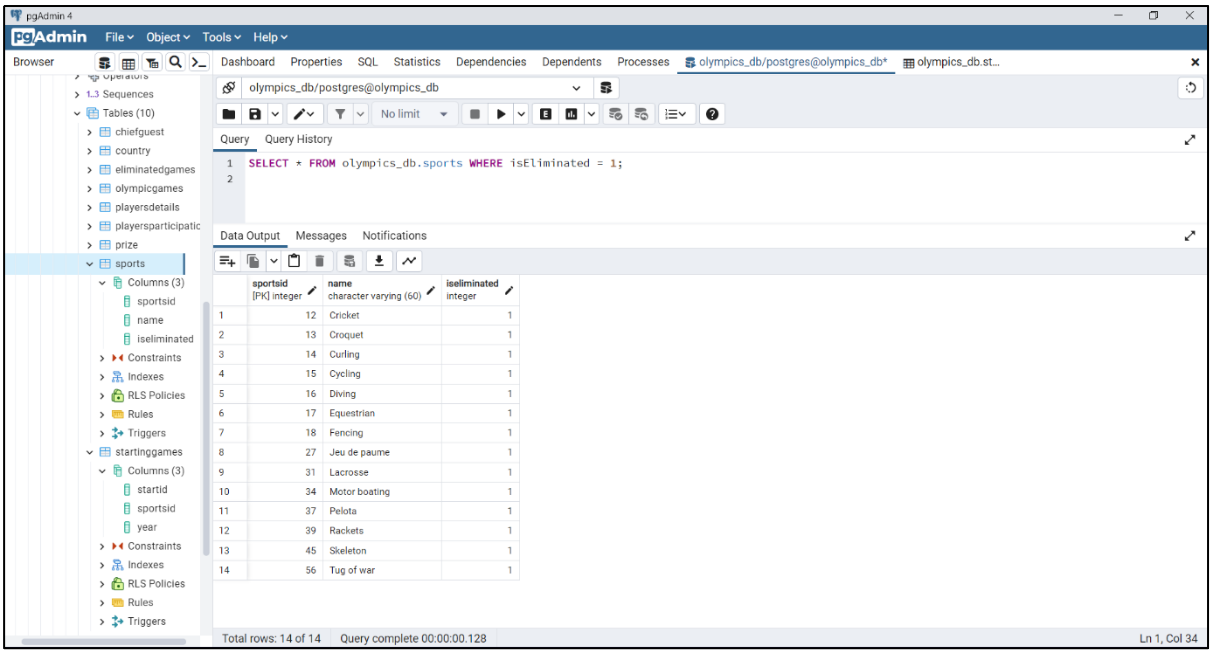
Total rows: 1 of 1   Query complete 00:00:00.101

→ Number of tuples-1

2. Select the sports that are eliminated:

→ SELECT * FROM olympics_db.sports WHERE isEliminated = 1;

→ Πsport_id,name,iseliminated(σ_isEliminated=1(sports))



```
SELECT * FROM olympics_db.sports WHERE isEliminated = 1;
```

	sportsid [PK] integer	name character varying (60)	iseliminated integer
1	12	Cricket	1
2	13	Croquet	1
3	14	Curling	1
4	15	Cycling	1
5	16	Diving	1
6	17	Equestrian	1
7	18	Fencing	1
8	27	Jeu de paume	1
9	31	Lacrosse	1
10	34	Motor boating	1
11	37	Pelota	1
12	39	Rackets	1
13	45	Skeleton	1
14	56	Tug of war	1

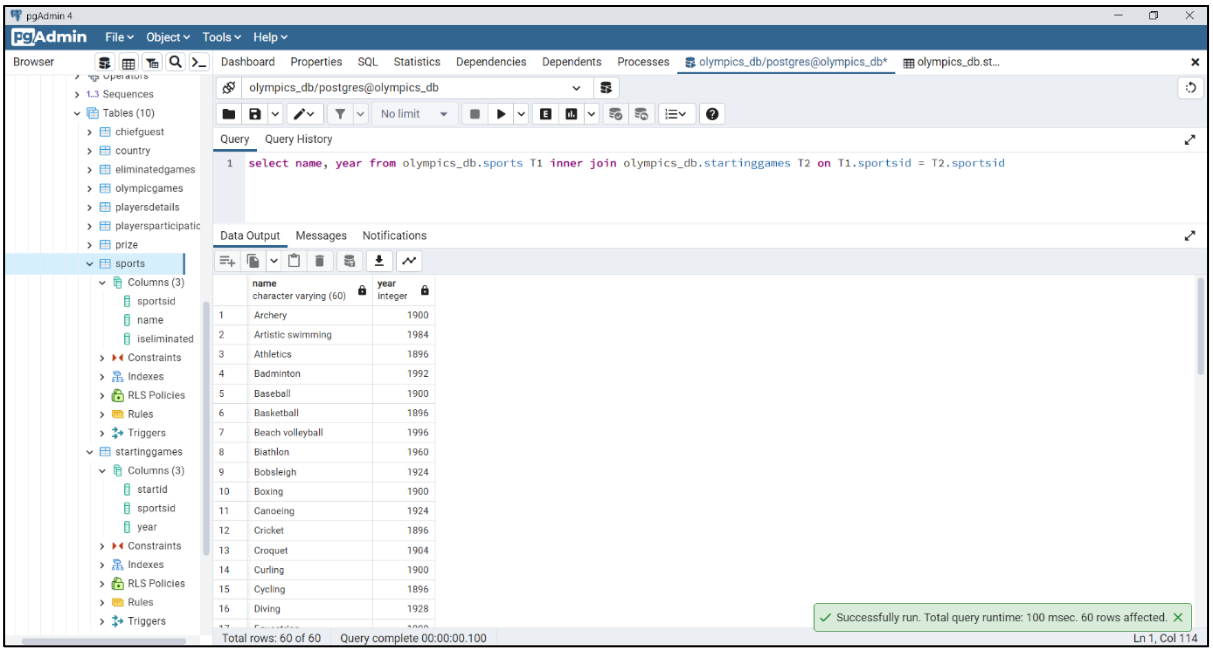
Total rows: 14 of 14 Query complete 00:00:00.128 Ln 1, Col 34

→ Number of tuples-14

3. Find the starting games for a specific sport:

→ select name, year from olympics_db.sports T1 inner join olympics_db.startinggames T2 on T1.sportsid = T2.sportsid

→ π_name,year(σ_sportsid(T1 ⋈ T2))



```
select name, year from olympics_db.sports T1 inner join olympics_db.startinggames T2 on T1.sportsid = T2.sportsid
```

	name character varying (60)	year integer
1	Archery	1900
2	Artistic swimming	1984
3	Athletics	1896
4	Badminton	1992
5	Baseball	1900
6	Basketball	1896
7	Beach volleyball	1996
8	Biathlon	1960
9	Bobsleigh	1924
10	Boxing	1900
11	Canoeing	1924
12	Cricket	1896
13	Croquet	1904
14	Curling	1900
15	Cycling	1896
16	Diving	1928

Successfully run. Total query runtime: 100 msec. 60 rows affected.

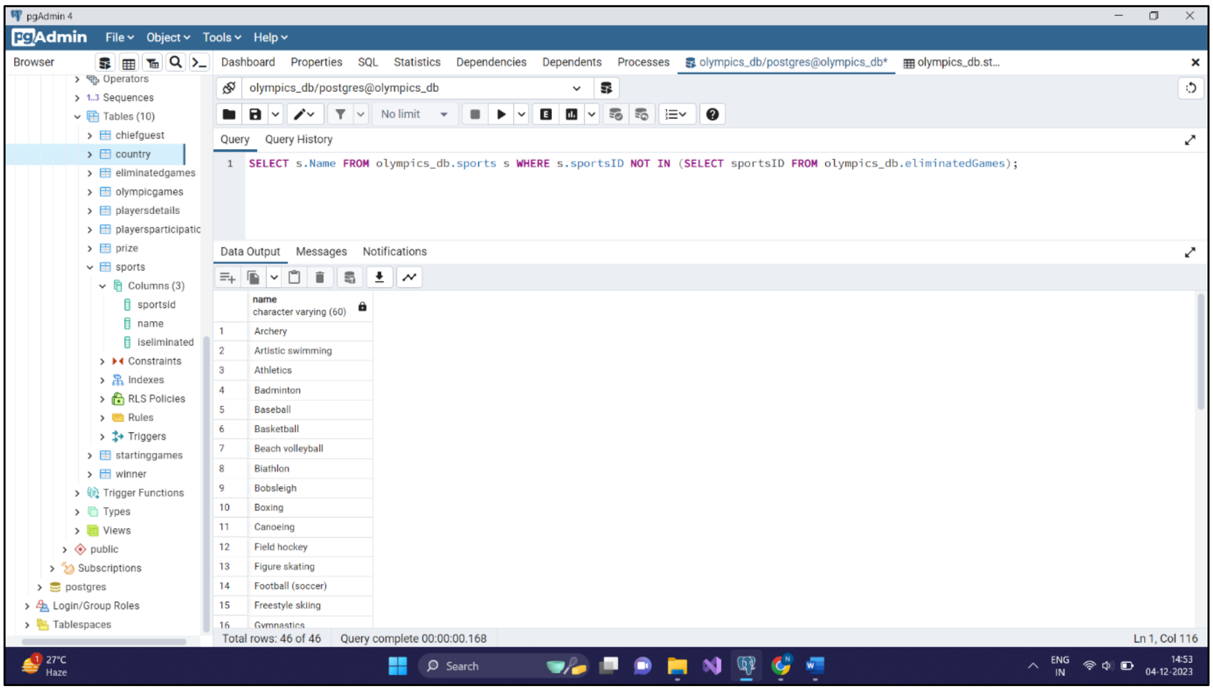
Total rows: 60 of 60 Query complete 00:00:00.100 Ln 1, Col 114

→ Number of tuples-60

4. Select the sports that have never been eliminated:

→ SELECT s.Name FROM olympics_db.sports s WHERE s.sportsID NOT IN (SELECT sportsID FROM olympics_db.eliminatedGames);

→ π_Name(σ_sportsID∉(eliminatedGames.sportsID)(sports))



```
SELECT s.Name FROM olympics_db.sports s WHERE s.sportsID NOT IN (SELECT sportsID FROM olympics_db.eliminatedGames);
```

	name character varying (60)
1	Archery
2	Artistic swimming
3	Athletics
4	Badminton
5	Baseball
6	Basketball
7	Beach volleyball
8	Biathlon
9	Bobsleigh
10	Boxing
11	Canoeing
12	Field hockey
13	Figure skating
14	Football (soccer)
15	Freestyle skiing
16	Gymnastics

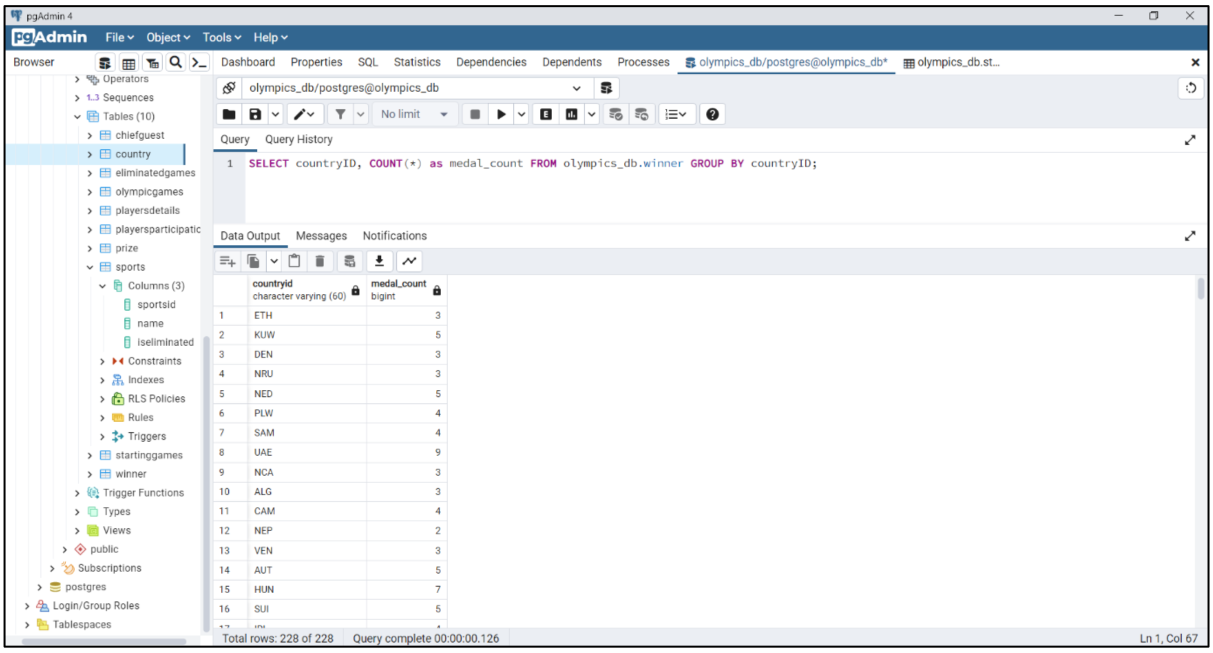
Total rows: 46 of 46 Query complete 00:00:00.168

→ Number of tuples-46

5. Get the total number of medals won by each country:

→ SELECT countryID, COUNT(*) as medal_count FROM olympics_db.winner GROUP BY countryID;

→ π_countryID,COUNT(*)(σ_winner(olympics_db))



```
SELECT countryID, COUNT(*) as medal_count FROM olympics_db.winner GROUP BY countryID;
```

	countryid character varying (60)	medal_count bigint
1	ETH	3
2	KUW	5
3	DEN	3
4	NRU	3
5	NED	5
6	PLW	4
7	SAM	4
8	UAE	9
9	NCA	3
10	ALG	3
11	CAM	4
12	NEP	2
13	VEN	3
14	AUT	5
15	HUN	7
16	SUI	5

Total rows: 228 of 228 Query complete 00:00:00.126 Ln 1, Col 67

→ Number of tuples-228

6.Select the names of players who won a gold medal:

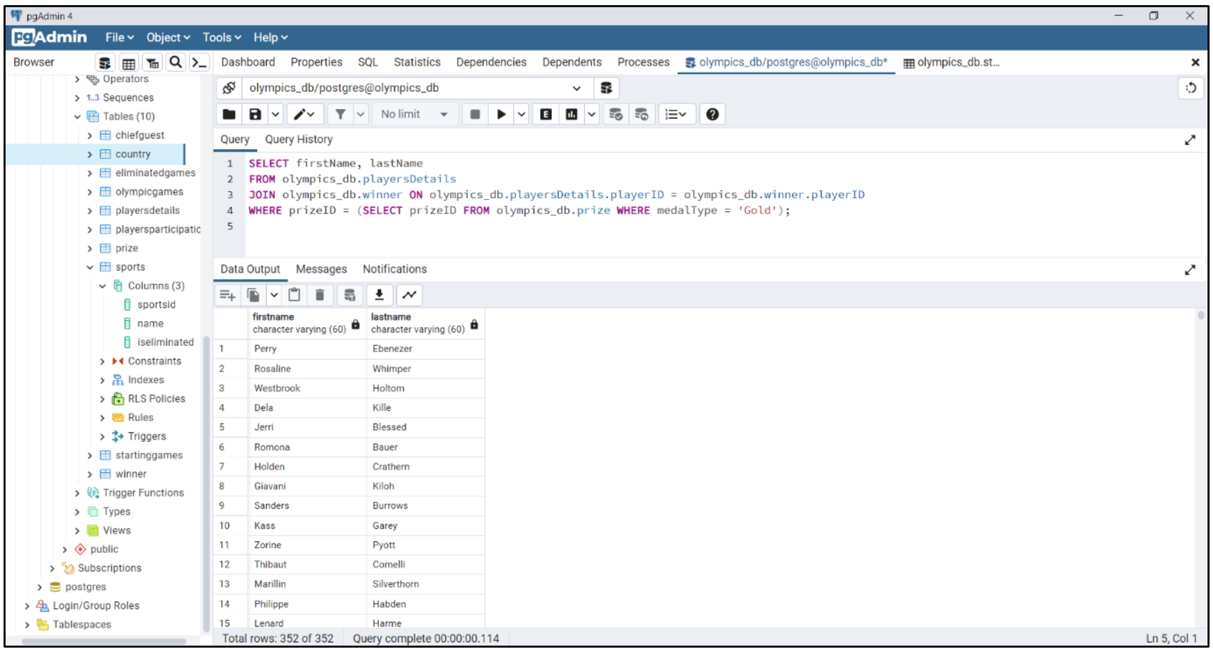
→ SELECT firstName, lastName

FROM olympics_db.playersDetails

JOIN olympics_db.winner ON olympics_db.playersDetails.playerID = olympics_db.winner.playerID

WHERE prizeID = (SELECT prizeID FROM olympics_db.prize WHERE medalType = 'Gold');

→ π firstName, lastName (σ prizeID = (π prizeID (σ medalType = 'Gold' (prize))) (playersDetails ⋈ winner ON playersDetails.playerID = winner.playerID))



```
SELECT firstName, lastName
FROM olympics_db.playersDetails
JOIN olympics_db.winner ON olympics_db.playersDetails.playerID = olympics_db.winner.playerID
WHERE prizeID = (SELECT prizeID FROM olympics_db.prize WHERE medalType = 'Gold');
```

	firstname character varying (60)	lastname character varying (60)
1	Perry	Ebenezer
2	Rosaline	Whimper
3	Westbrook	Holtom
4	Dela	Kille
5	Jerri	Blessed
6	Romona	Bauer
7	Holden	Crathern
8	Giavani	Kiloh
9	Sanders	Burrows
10	Kass	Garey
11	Zorine	Pyott
12	Thibaut	Comelli
13	Marillin	Silverthorn
14	Philippe	Habden
15	Lenard	Harme

Total rows: 352 of 352 Query complete 00:00:00.114 Ln 5, Col 1

→ Number of tuples-352

7. Calculate the total prize amount won by players between 1 to 100:

→ SELECT pd.firstName, pd.lastName, SUM(pr.amount) AS total_amount

FROM olympics_db.playersDetails pd

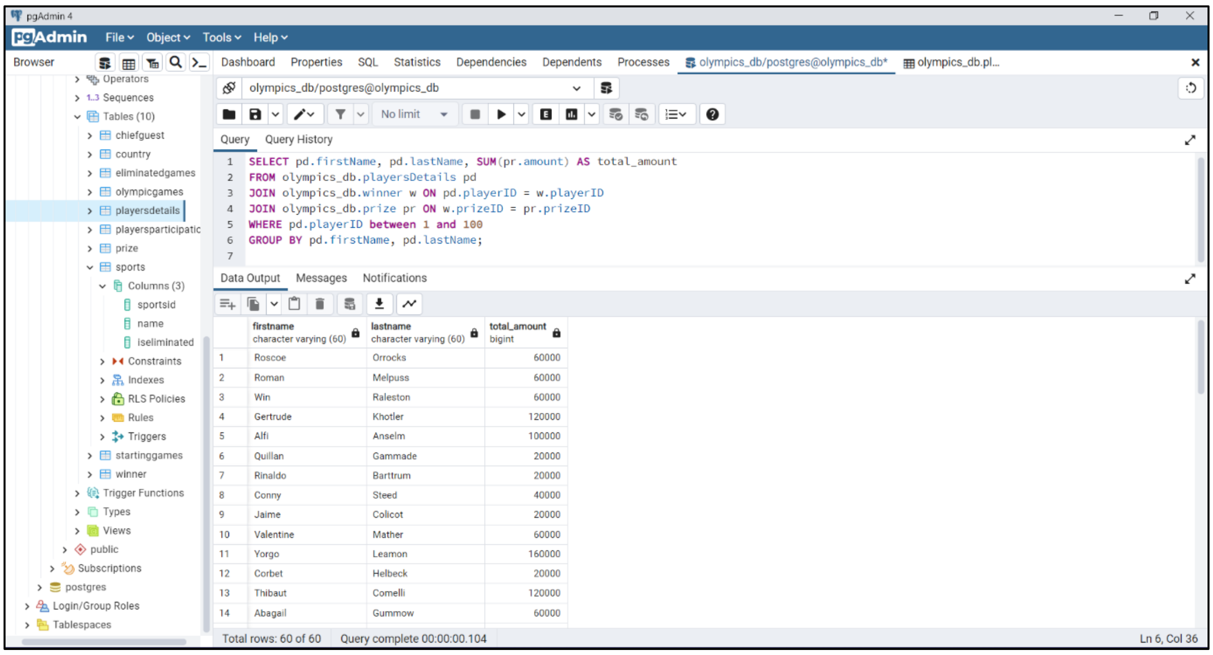
JOIN olympics_db.winner w ON pd.playerID = w.playerID

JOIN olympics_db.prize pr ON w.prizeID = pr.prizeID

WHERE pd.playerID between 1 and 100

GROUP BY pd.firstName, pd.lastName;

→ πfirstName, lastName, total_amount
(γpd.firstName, pd.lastName, SUM(pr.amount) AS total_amount
(σpd.playerID≥1∧pd.playerID≤100(ρplayerID→w.playerID
(playersDetails⋈pd.playerID=w.playerID winner)⋈prize)))



```
SELECT pd.firstName, pd.lastName, SUM(pr.amount) AS total_amount
FROM olympics_db.playersDetails pd
JOIN olympics_db.winner w ON pd.playerID = w.playerID
JOIN olympics_db.prize pr ON w.prizeID = pr.prizeID
WHERE pd.playerID between 1 and 100
GROUP BY pd.firstName, pd.lastName;
```

	firstname character varying (60)	lastname character varying (60)	total_amount bigint
1	Roscoe	Orrocks	60000
2	Roman	Melpuss	60000
3	Win	Raleston	60000
4	Gertrude	Khotler	120000
5	Alfi	Anselm	100000
6	Quillan	Gammade	20000
7	Rinaldo	Barttrum	20000
8	Conny	Steed	40000
9	Jaime	Colicot	20000
10	Valentine	Mather	60000
11	Yorgo	Leamon	160000
12	Corbet	Helbeck	20000
13	Thibaut	Comelli	120000
14	Abagail	Gummow	60000

Total rows: 60 of 60 Query complete 00:00:00.104

→ Number of tuples-60

8. Select the players who won a medal in every Olympic game they participated in:

→ SELECT pd.firstName, pd.lastName

FROM olympics_db.playersDetails pd

WHERE pd.playerID NOT IN (

SELECT pd.playerID

FROM olympics_db.playersDetails pd

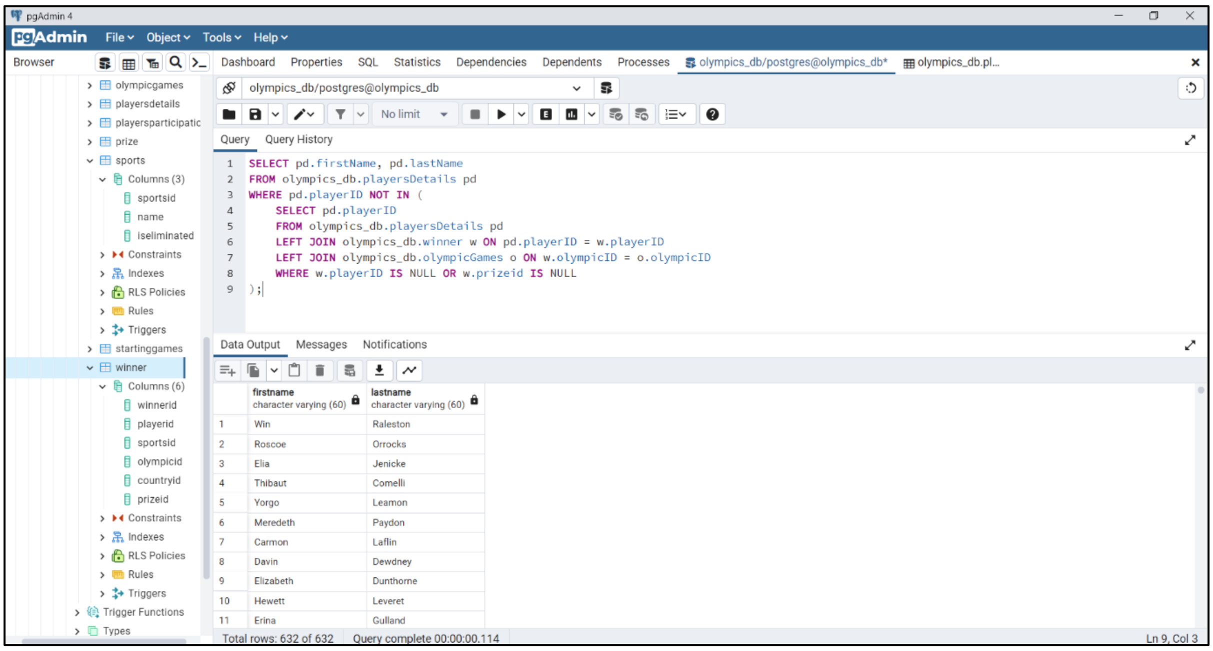
LEFT JOIN olympics_db.winner w ON pd.playerID = w.playerID

LEFT JOIN olympics_db.olympicGames o ON w.olympicID = o.olympicID

WHERE w.playerID IS NULL OR w.prizeid IS NULL

);

→ πfirstName, lastName(playersDetails−πpd.playerID (σw.playerID IS NULL OR w.prizeid IS NULL(ρplayerID→w.playerID (playersDetails⋈pd.playerID=w.playerIDwinner⋈olympicGames))))



```
SELECT pd.firstName, pd.lastName
FROM olympics_db.playersDetails pd
WHERE pd.playerID NOT IN (
    SELECT pd.playerID
    FROM olympics_db.playersDetails pd
    LEFT JOIN olympics_db.winner w ON pd.playerID = w.playerID
    LEFT JOIN olympics_db.olympicGames o ON w.olympicID = o.olympicID
    WHERE w.playerID IS NULL OR w.prizeid IS NULL
);
```

	firstname character varying (60)	lastname character varying (60)
1	Win	Raleston
2	Roscoe	Orrocks
3	Elia	Jenicke
4	Thibaut	Comelli
5	Yorgo	Leamon
6	Meredeth	Paydon
7	Carmon	Laflin
8	Davin	Dewdney
9	Elizabeth	Dunthorne
10	Hewett	Leveret
11	Erina	Gulland

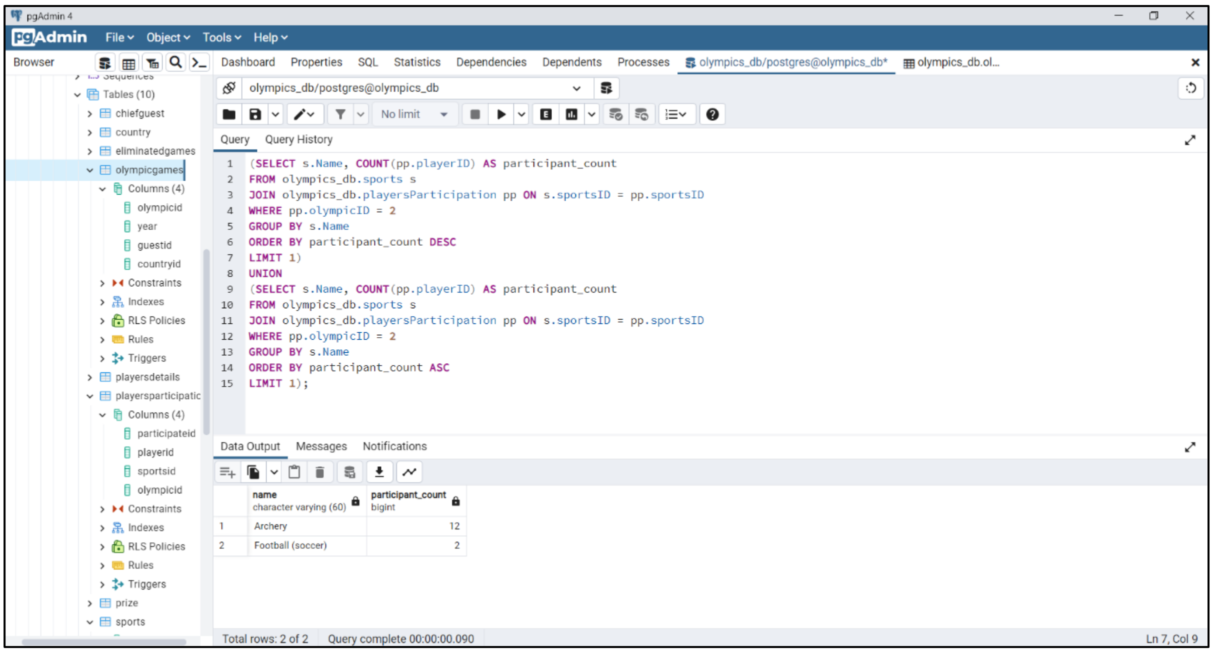
Total rows: 632 of 632 Query complete 00:00:00.114 Ln 9, Col 3

→ Number of tuples-632

9. Select the sports with the highest and lowest number of participants in a specific Olympic game:

→
```
(SELECT s.Name, COUNT(pp.playerID) AS participant_count
FROM olympics_db.sports s
JOIN olympics_db.playersParticipation pp ON s.sportsID = pp.sportsID
WHERE pp.olympicID = 2
GROUP BY s.Name
ORDER BY participant_count DESC
LIMIT 1)
UNION
(SELECT s.Name, COUNT(pp.playerID) AS participant_count
FROM olympics_db.sports s
JOIN olympics_db.playersParticipation pp ON s.sportsID = pp.sportsID
WHERE pp.olympicID = 2
GROUP BY s.Name
ORDER BY participant_count ASC
LIMIT 1);
```

→ πName, participant_count(σpp.olympicID=2(sports⋈s.sportsID=pp.sportsID playersParticipation)⋈s1.participant_count = s2.participant_count (ρs.Name, COUNT(pp.playerID) AS participant_count (γs.Name, COUNT(pp.playerID) AS participant_count(sports⋈s.sportsID=pp.sportsID playersParticipation))))



```
(SELECT s.Name, COUNT(pp.playerID) AS participant_count
FROM olympics_db.sports s
JOIN olympics_db.playersParticipation pp ON s.sportsID = pp.sportsID
WHERE pp.olympicID = 2
GROUP BY s.Name
ORDER BY participant_count DESC
LIMIT 1)
UNION
(SELECT s.Name, COUNT(pp.playerID) AS participant_count
FROM olympics_db.sports s
JOIN olympics_db.playersParticipation pp ON s.sportsID = pp.sportsID
WHERE pp.olympicID = 2
GROUP BY s.Name
ORDER BY participant_count ASC
LIMIT 1);
```

	name character varying (60)	participant_count bigint
1	Archery	12
2	Football (soccer)	2

Total rows: 2 of 2 Query complete 00:00:00.090

→ Number of tuples-2

10. Select the countries where the average age of players is above the global average:

→ SELECT co.countryName

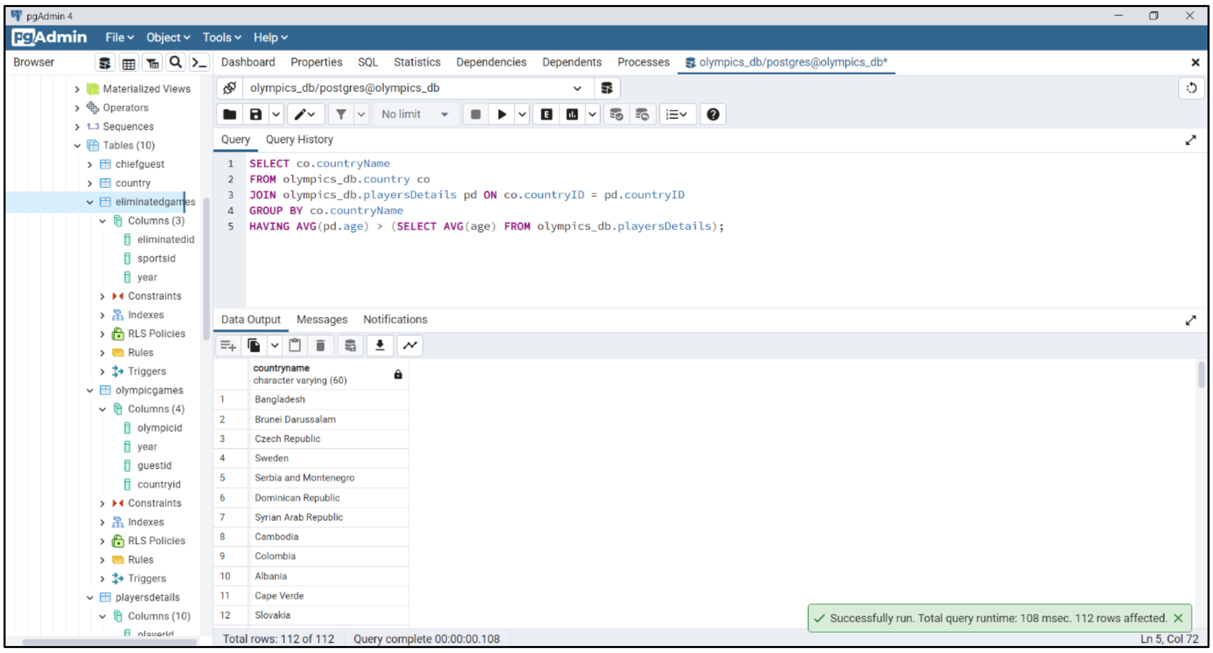
FROM olympics_db.country co

JOIN olympics_db.playersDetails pd ON co.countryID = pd.countryID

GROUP BY co.countryName

HAVING AVG(pd.age) > (SELECT AVG(age) FROM olympics_db.playersDetails);

→ πcountryName(σAVG(pd.age) > AVG(age)(γco.countryName (country⋈co.countryID=pd.countryIDplayersDetails)⋈γAVG(age)(playersDetails)))



```
SELECT co.countryName
FROM olympics_db.country co
JOIN olympics_db.playersDetails pd ON co.countryID = pd.countryID
GROUP BY co.countryName
HAVING AVG(pd.age) > (SELECT AVG(age) FROM olympics_db.playersDetails);
```

	countryname character varying (60)
1	Bangladesh
2	Brunei Darussalam
3	Czech Republic
4	Sweden
5	Serbia and Montenegro
6	Dominican Republic
7	Syrian Arab Republic
8	Cambodia
9	Colombia
10	Albania
11	Cape Verde
12	Slovakia

Successfully run. Total query runtime: 108 msec. 112 rows affected.

Total rows: 112 of 112 Query complete 00:00:00.108 Ln 5, Col 72

→ Number of tuples-112