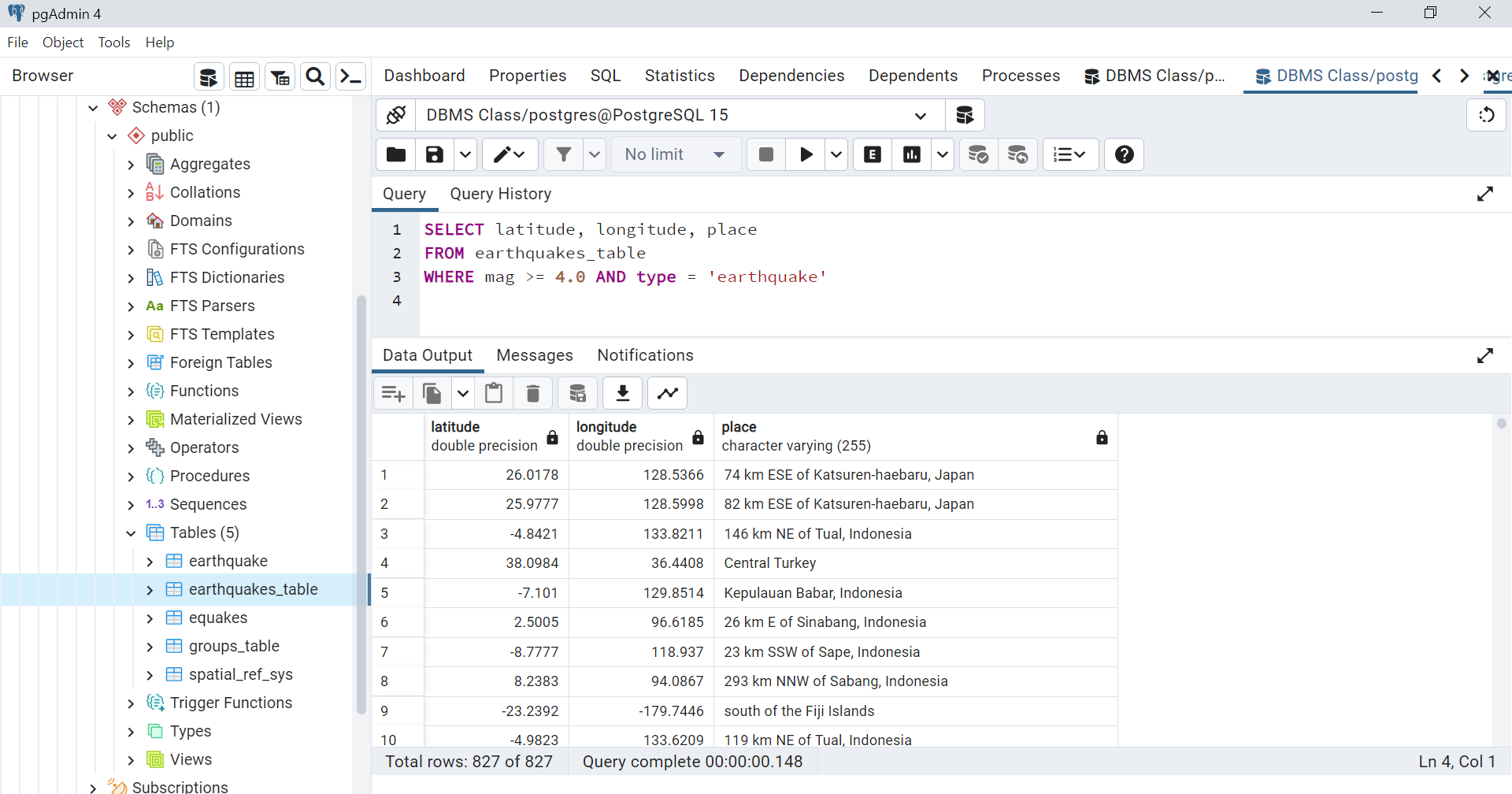
**Retrieve Locations of specific features.**

SELECT latitude, longitude, place

FROM earthquakes\_table

WHERE mag >= 4.0 AND type = 'earthquake';

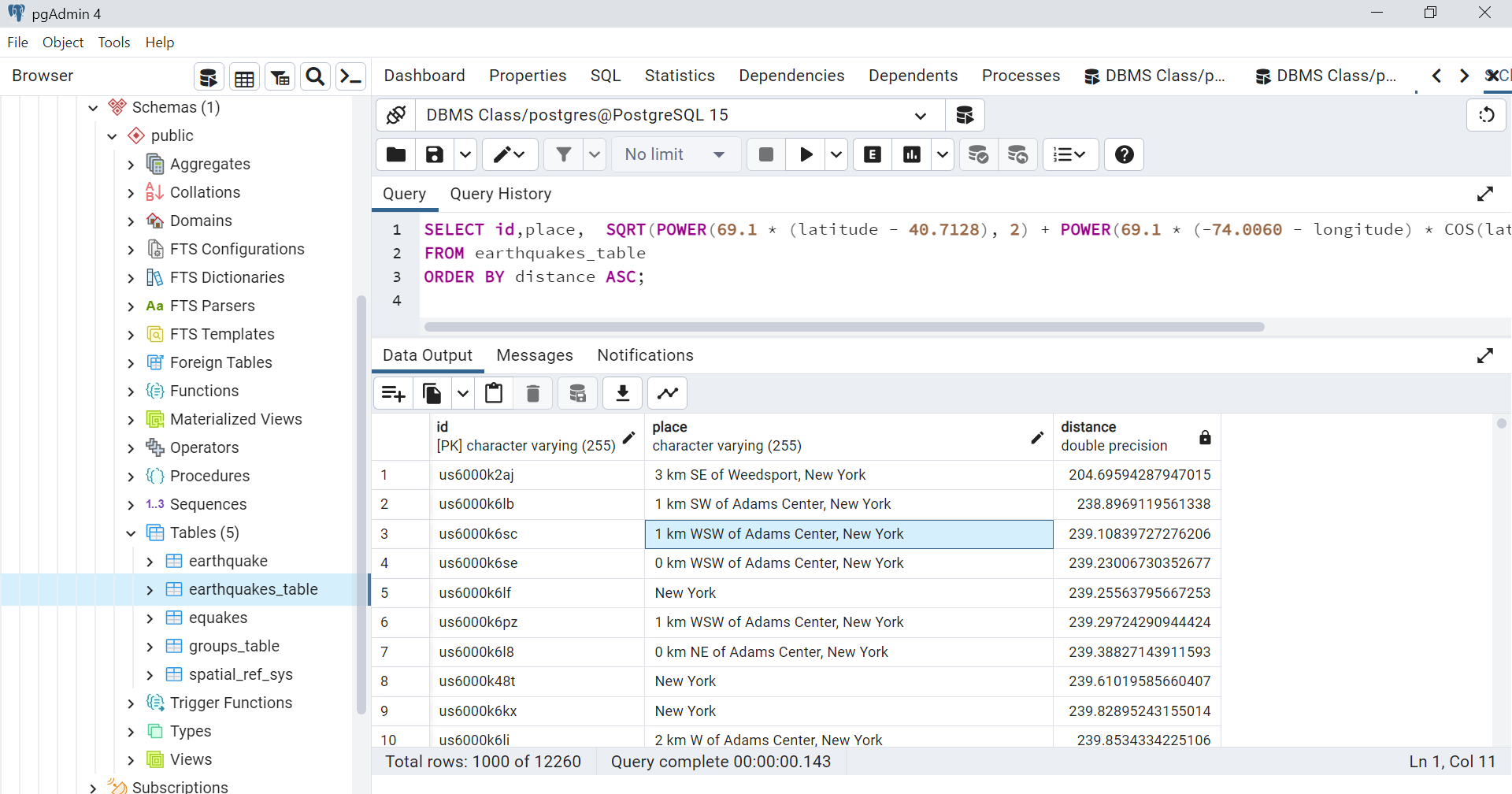


**Calculate Distance between points:**

SELECT id,place, SQRT(POWER(69.1 \* (latitude - 40.7128), 2) + POWER(69.1 \* (-74.0060 - longitude) \* COS(latitude / 57.3), 2)) AS distance

FROM earthquakes\_table

ORDER BY distance ASC;



**Calculate Areas of Interest specific to each group**

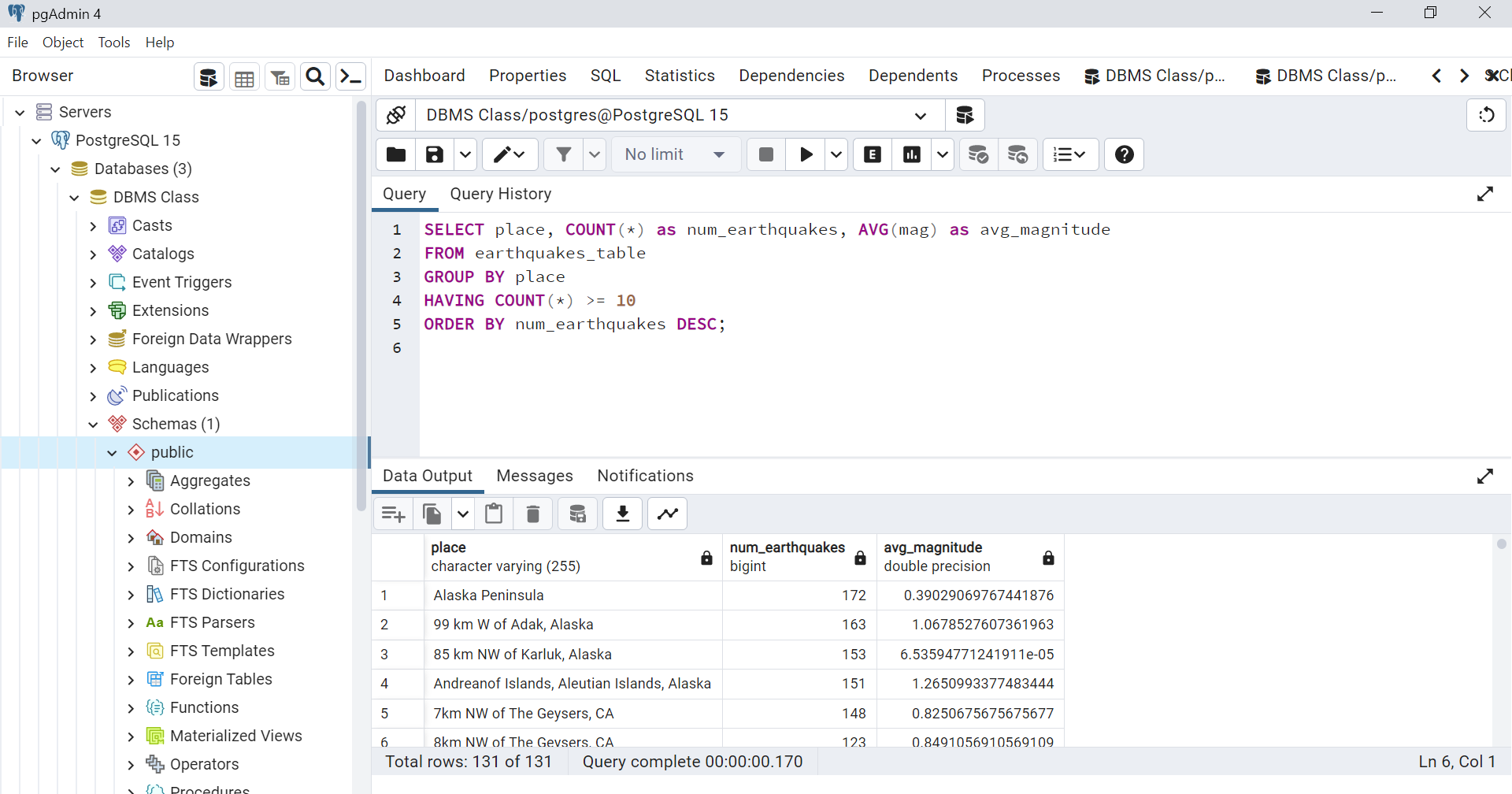
SELECT place, COUNT(\*) as num\_earthquakes, AVG(mag) as avg\_magnitude

FROM earthquakes\_table

GROUP BY place

HAVING COUNT(\*) >= 10

ORDER BY num\_earthquakes DESC;



**Analyze the queries:**

SELECT latitude, longitude, place

FROM earthquakes\_table

WHERE mag >= 4.0 AND type = 'earthquake';

With a magnitude of at least 4.0 and a "earthquake" type, this query chooses the latitude, longitude, and location from the earthquakes\_table. This search is probably used to retrieve data on earthquakes that are notable in terms of magnitude. The place can be used to provide extra information about the earthquake's location, such as the name of the closest city or landmark, while the latitude and longitude can be used to plot the earthquake's location on a map.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

SELECT id,place, SQRT(POWER(69.1 \* (latitude - 40.7128), 2) + POWER(69.1 \* (-74.0060 - longitude) \* COS(latitude / 57.3), 2)) AS distance

FROM earthquakes\_table

ORDER BY distance ASC;

This query selects the id, position, and distance of each earthquake in the earthquakes\_table and arranges them according to how far they were from the New York City coordinates of (40.7128, -74.0060).

The distance is calculated using the Haversine formula, which determines the shortest path between two points on a sphere while accounting for the curvature of the Earth's surface. The radius of the Earth, which is 3960 miles, is used to determine the distance in miles.

To determine the separation between two locations on the surface of a sphere, like the Earth, mathematicians utilize the Haversine formula. It is more accurate than using the Pythagorean theorem alone to determine distance since it takes the curvature of the Earth's surface into consideration.

The central angle, or angle between two radii of a sphere that intersect in the center of the sphere, is the basis for the formula. Two sites' latitude and longitude can be used to get the center angle.

The great-circle distance, which is the shortest distance between two points on the surface of a sphere, is determined using the Haversine formula using the central angle. The formula accounts for the radius of the Earth, which is important for precise computations.

Geographic information systems (GIS) and location-based applications frequently employ the Haversine formula to determine the separation between two points that are identified by their latitude and longitude.

This search is helpful for locating earthquakes that occur close to New York City overall.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

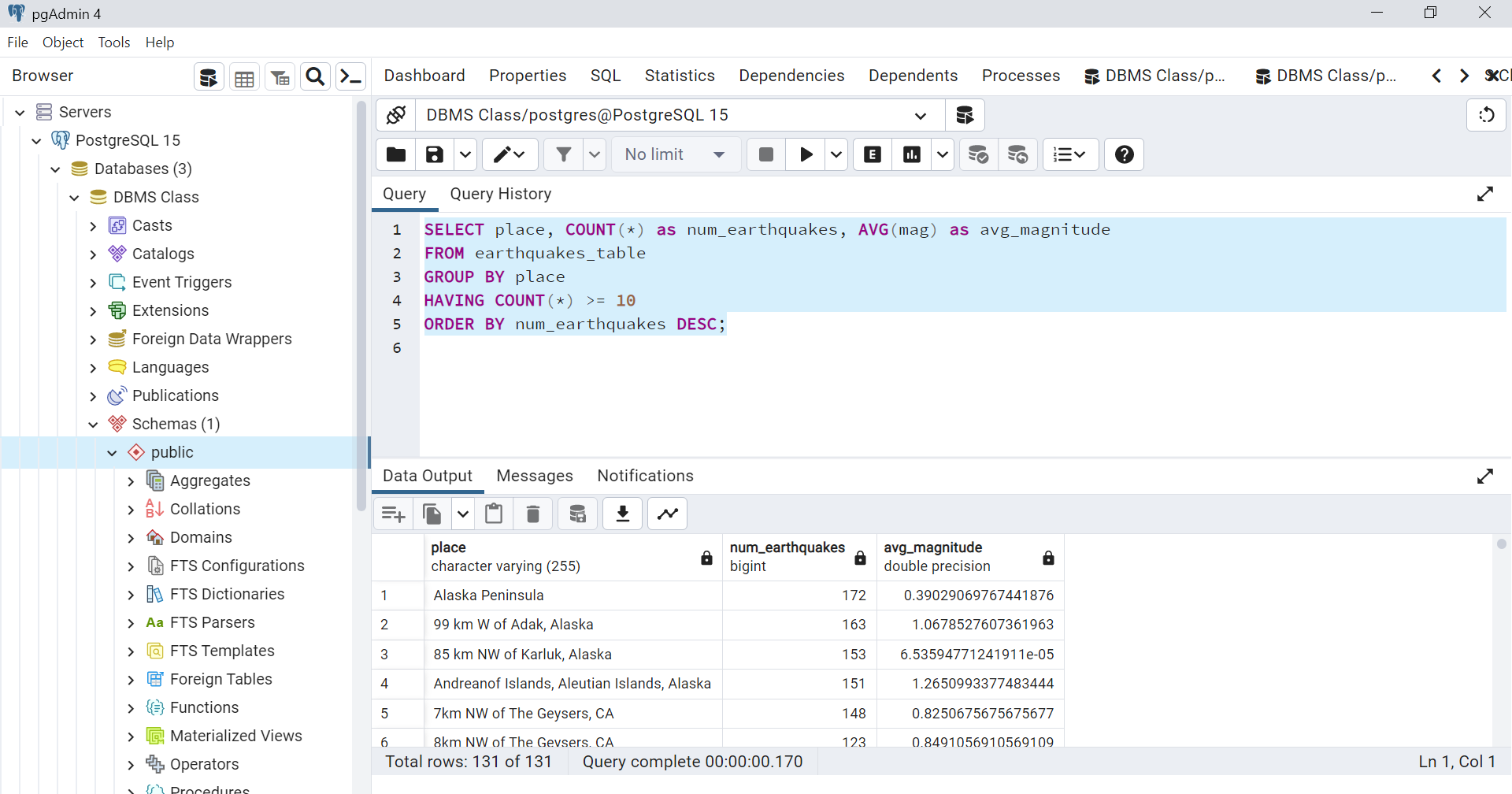
SELECT place, COUNT(\*) as num\_earthquakes, AVG(mag) as avg\_magnitude

FROM earthquakes\_table

GROUP BY place

HAVING COUNT(\*) >= 10

ORDER BY num\_earthquakes DESC;



This query groups the earthquakes by their place, counts the number of earthquakes and calculates the average magnitude for each place. Then, it only returns the results where there are at least 10 earthquakes at the same place, and finally sorts the results in descending order by the number of earthquakes.

**Sorting and Limit Executions :**

SELECT \*

FROM earthquakes\_table

WHERE mag >= 4.0 AND type = 'earthquake'

ORDER BY time DESC

LIMIT 10;

SELECT \* -- Select all columns from the earthquakes\_table

FROM earthquakes\_table -- Specify the table to retrieve data from

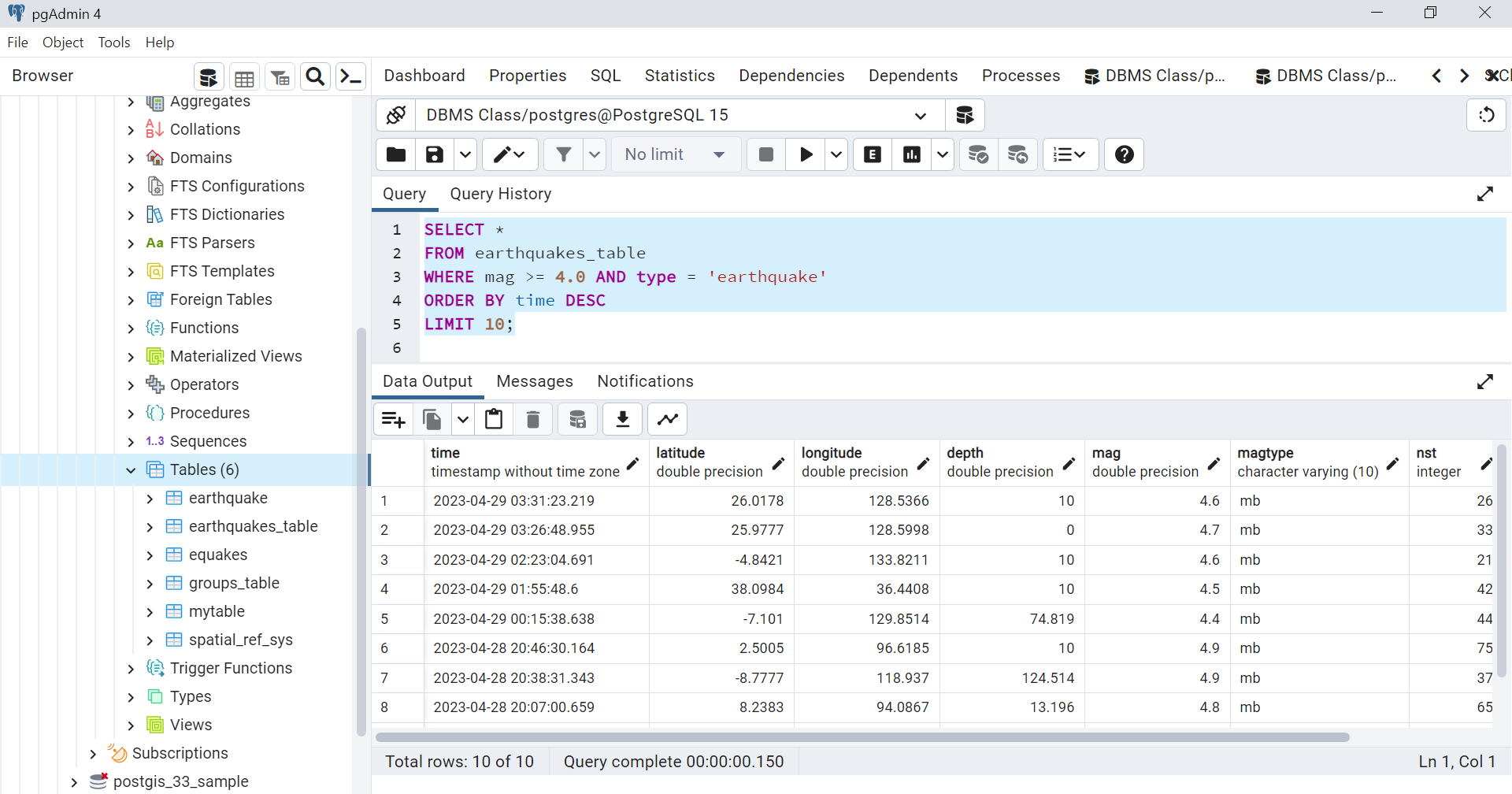
WHERE mag >= 4.0 -- Filter the results to only include earthquakes with magnitude >= 4.0

AND type = 'earthquake' -- Filter the results to only include earthquakes of type 'earthquake'

ORDER BY time DESC -- Sort the results in descending order by the time column

LIMIT 10; -- Limit the number of results to 10

This search will pull all columns from the earthquakes\_table whose types are "earthquake" and whose magnitudes are greater than or equal to 4.0. The results are then being limited to 10 and sorted by the time column in descending order. The results of this search will show the ten most recent earthquakes with a magnitude of at least 4.0.



**Optimization of queries for faster execution:**

**Code Before optimization:**

SELECT place, COUNT(\*) as num\_earthquakes, AVG(mag) as avg\_magnitude FROM earthquakes\_table GROUP BY place HAVING COUNT(\*) >= 10 ORDER BY num\_earthquakes DESC;

**After Optimization:**

CREATE INDEX idx\_place ON earthquakes\_table(place);

SELECT place, COUNT(\*) as num\_earthquakes, AVG(mag) as avg\_magnitude

FROM (

SELECT \*

FROM earthquakes\_table

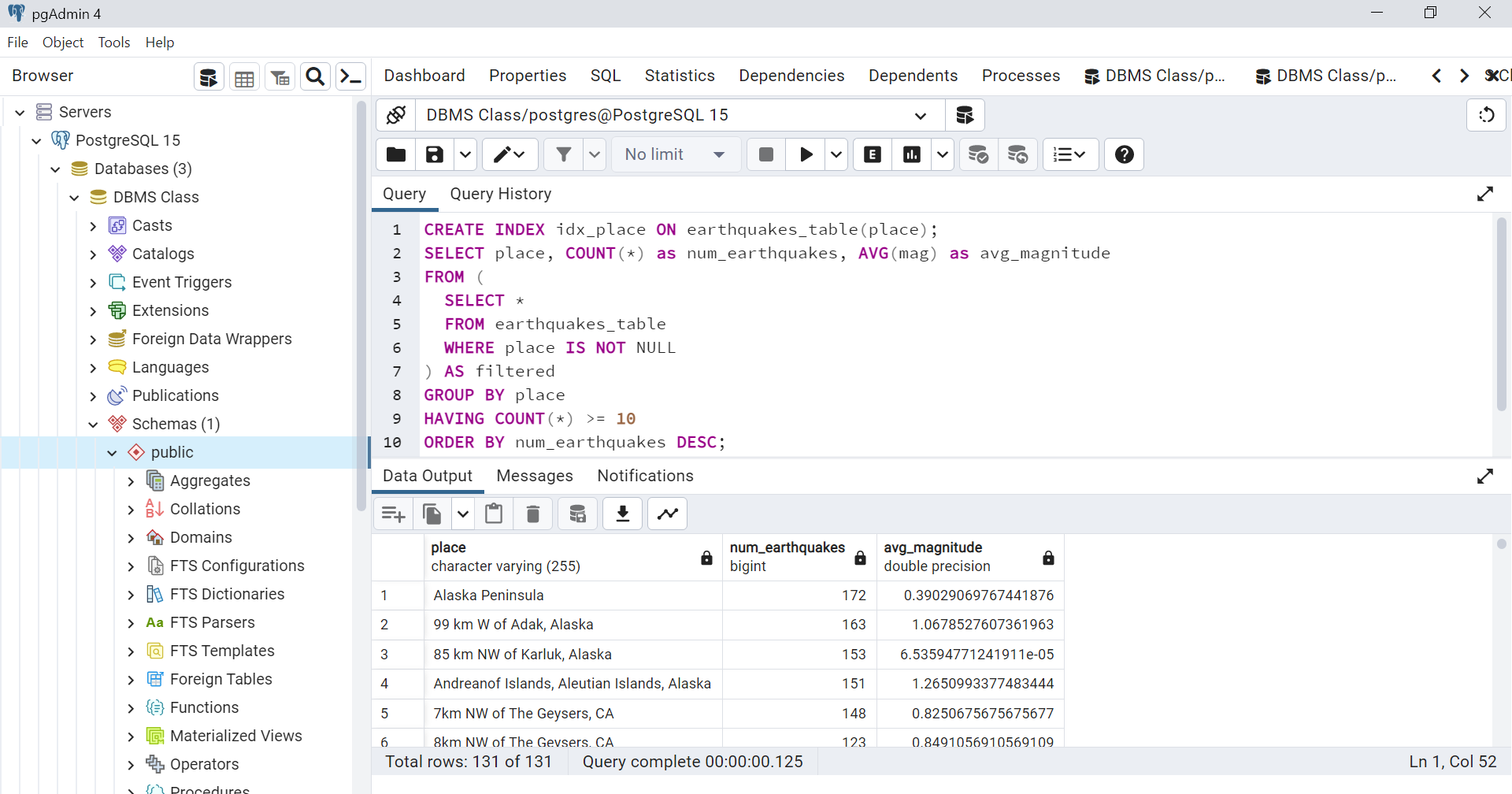
WHERE place IS NOT NULL

) AS filtered

GROUP BY place

HAVING COUNT(\*) >= 10

ORDER BY num\_earthquakes DESC;



**Query before Optimization:**

SELECT id,place, SQRT(POWER(69.1 \* (latitude - 40.7128), 2) + POWER(69.1 \* (-74.0060 - longitude) \* COS(latitude / 57.3), 2)) AS distance FROM earthquakes\_table ORDER BY distance ASC;

To optimize this query is to create an index on the longitude and latitude columns to speed up the sorting process.

This index will improve the performance of the query by allowing the database to quickly locate and sort the relevant rows based on latitude and longitude.

Additionally, you could limit the number of rows returned by the query by using the LIMIT clause if you only need a certain number of results.

**After optimization:**

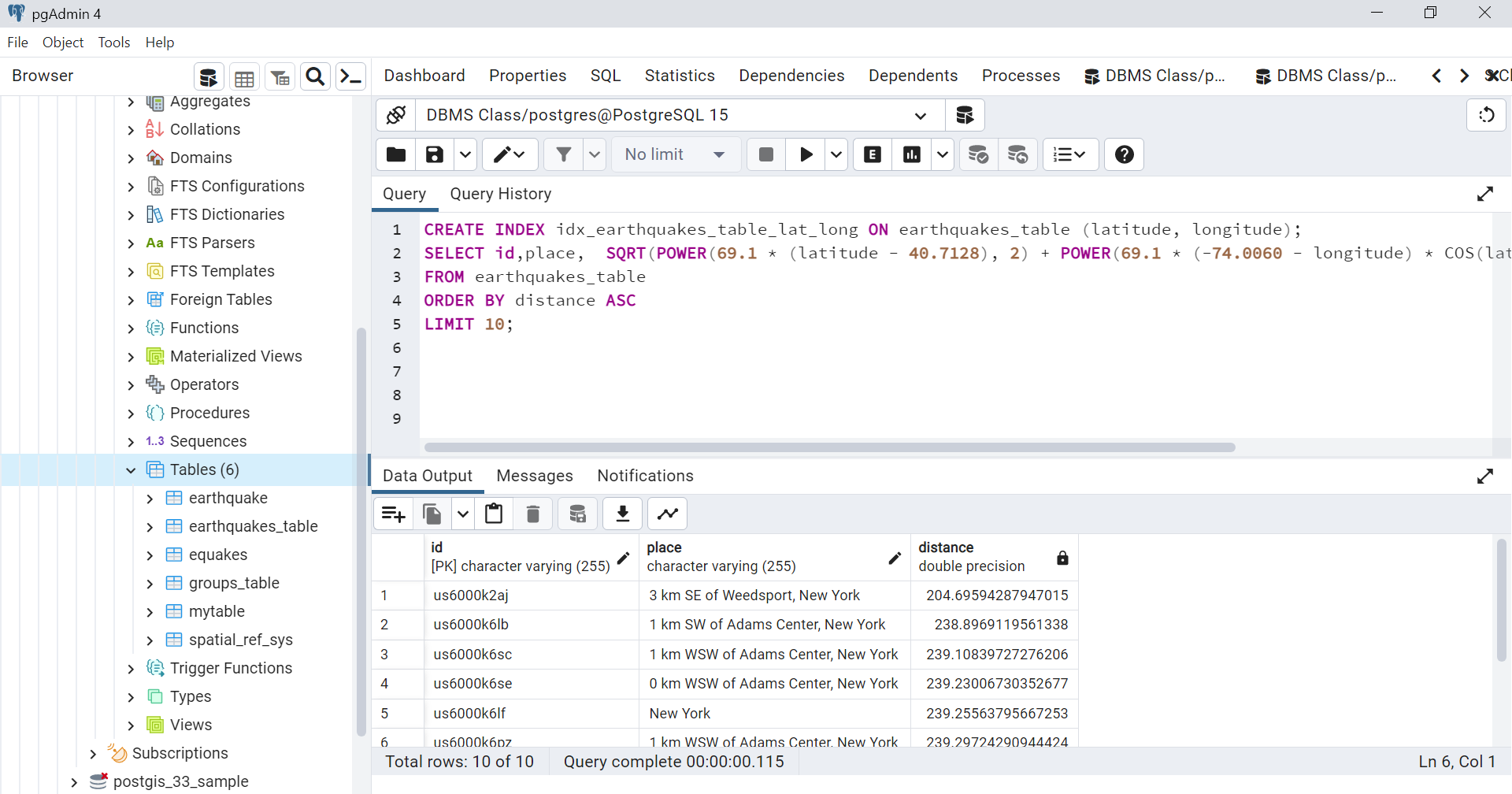
CREATE INDEX idx\_earthquakes\_table\_lat\_long ON earthquakes\_table (latitude, longitude);

SELECT id,place, SQRT(POWER(69.1 \* (latitude - 40.7128), 2) + POWER(69.1 \* (-74.0060 - longitude) \* COS(latitude / 57.3), 2)) AS distance

FROM earthquakes\_table

ORDER BY distance ASC

LIMIT 10;



**Result comparisons after query optimization:**

compare the execution results at the bottom of the screen shot **115 millisecond** is the execution time of optimized query from the first screen shot query which had **143 millisecond** is much optimized and faster than earlier.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Query before optimization:**

SELECT latitude, longitude, place

FROM earthquakes\_table

WHERE mag >= 4.0 AND type = 'earthquake';

since this query does not involve any aggregations or complex spatial operations, there is not much scope for optimization. However, we can still try to optimize it by creating an index on the mag and type columns, which will speed up the filtering process.

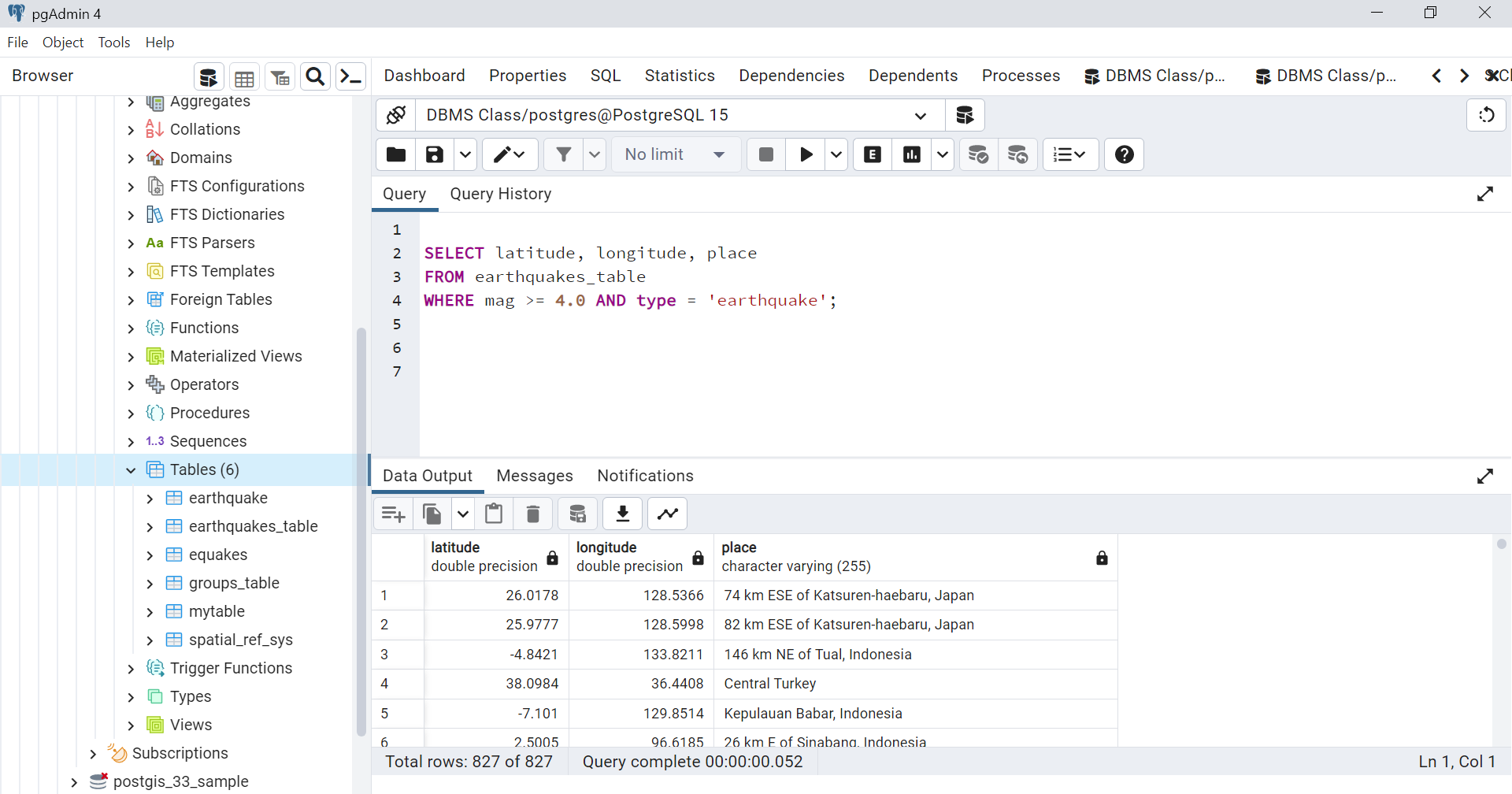
**After Query Optimization:**

CREATE INDEX mag\_type\_index ON earthquakes\_table (mag, type);

SELECT latitude, longitude, place

FROM earthquakes\_table

WHERE mag >= 4.0 AND type = 'earthquake';

As you can compare the execution results at the bottom of the screen shot **052 millisecond** is the execution time of optimized query from the first screen shot query which had **148 millisecond** is much optimized and faster than earlier.