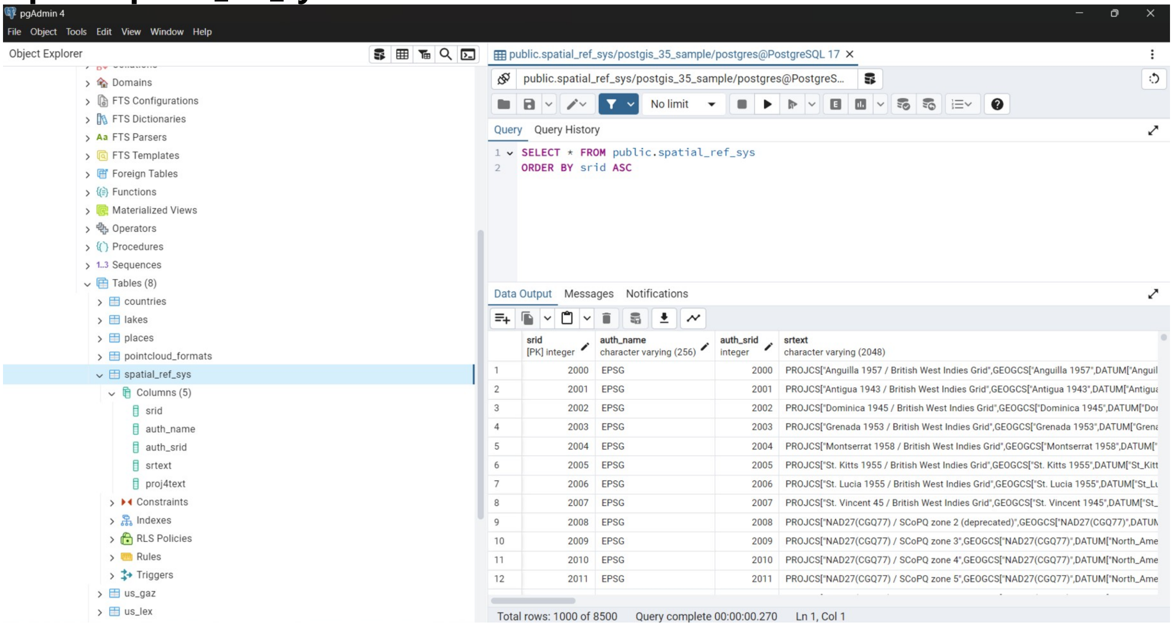
Submitted By:
Ashwin E
SC24M136

Ex. No. 9 Scientific Computing

Spatial DBMS

1. Spatial Queries

(a) Explore spatial_ref_sys table

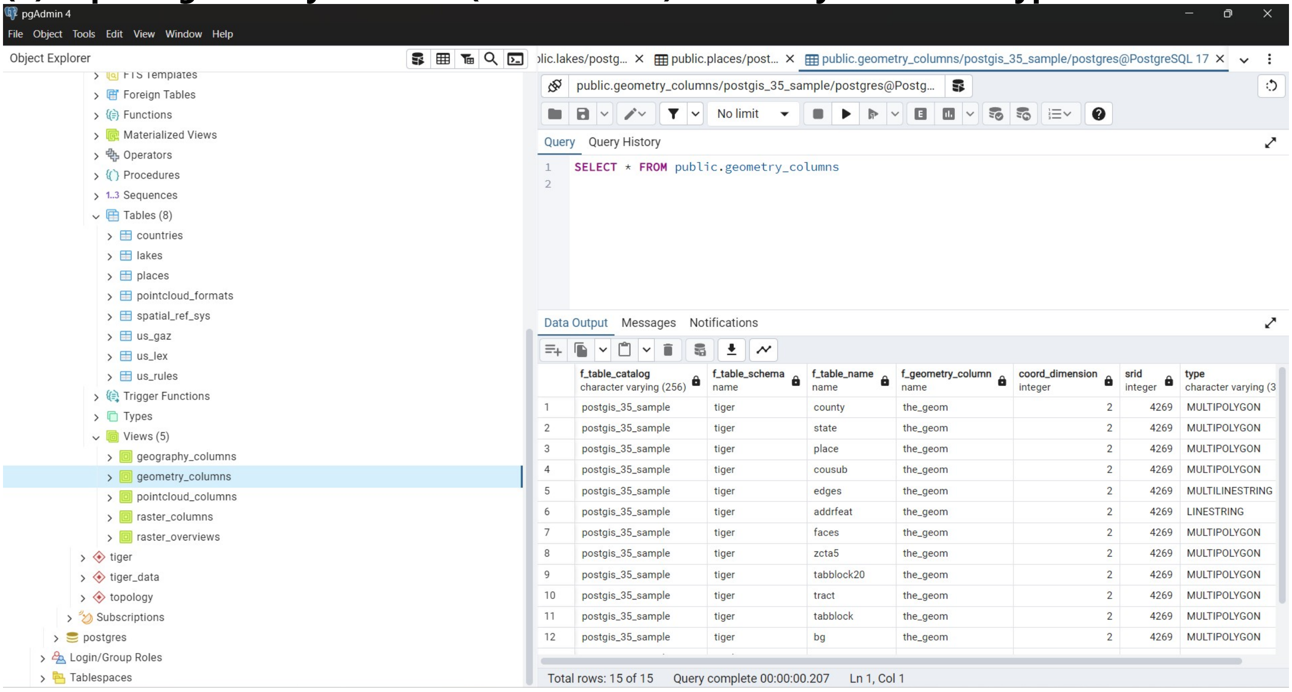


```
SELECT * FROM public.spatial_ref_sys
ORDER BY srid ASC
```

	srid [PK] integer	auth_name character varying (256)	auth_srid integer	srtext character varying (2048)
1	2000	EPSG	2000	PROJCS["Anguilla 1957 / British West Indies Grid",GEOGCS["Anguilla 1957",DATUM["Anguil
2	2001	EPSG	2001	PROJCS["Antigua 1943 / British West Indies Grid",GEOGCS["Antigua 1943",DATUM["Antigua
3	2002	EPSG	2002	PROJCS["Dominica 1945 / British West Indies Grid",GEOGCS["Dominica 1945",DATUM["Dor
4	2003	EPSG	2003	PROJCS["Grenada 1953 / British West Indies Grid",GEOGCS["Grenada 1953",DATUM["Grena
5	2004	EPSG	2004	PROJCS["Montserrat 1958 / British West Indies Grid",GEOGCS["Montserrat 1958",DATUM["
6	2005	EPSG	2005	PROJCS["St. Kitts 1955 / British West Indies Grid",GEOGCS["St. Kitts 1955",DATUM["St_Kitt
7	2006	EPSG	2006	PROJCS["St. Lucia 1955 / British West Indies Grid",GEOGCS["St. Lucia 1955",DATUM["St_Lu
8	2007	EPSG	2007	PROJCS["St. Vincent 45 / British West Indies Grid",GEOGCS["St. Vincent 1945",DATUM["St_
9	2008	EPSG	2008	PROJCS["NAD27(CGQ77) / SCoPQ zone 2 (deprecated)",GEOGCS["NAD27(CGQ77)",DATUM
10	2009	EPSG	2009	PROJCS["NAD27(CGQ77) / SCoPQ zone 3",GEOGCS["NAD27(CGQ77)",DATUM["North_Ame
11	2010	EPSG	2010	PROJCS["NAD27(CGQ77) / SCoPQ zone 4",GEOGCS["NAD27(CGQ77)",DATUM["North_Ame
12	2011	EPSG	2011	PROJCS["NAD27(CGQ77) / SCoPQ zone 5",GEOGCS["NAD27(CGQ77)",DATUM["North_Ame

Total rows: 1000 of 8500 Query complete 00:00:00.270 Ln 1, Col 1

(b) Explore geometry columns (under Views) Geometry of all data type



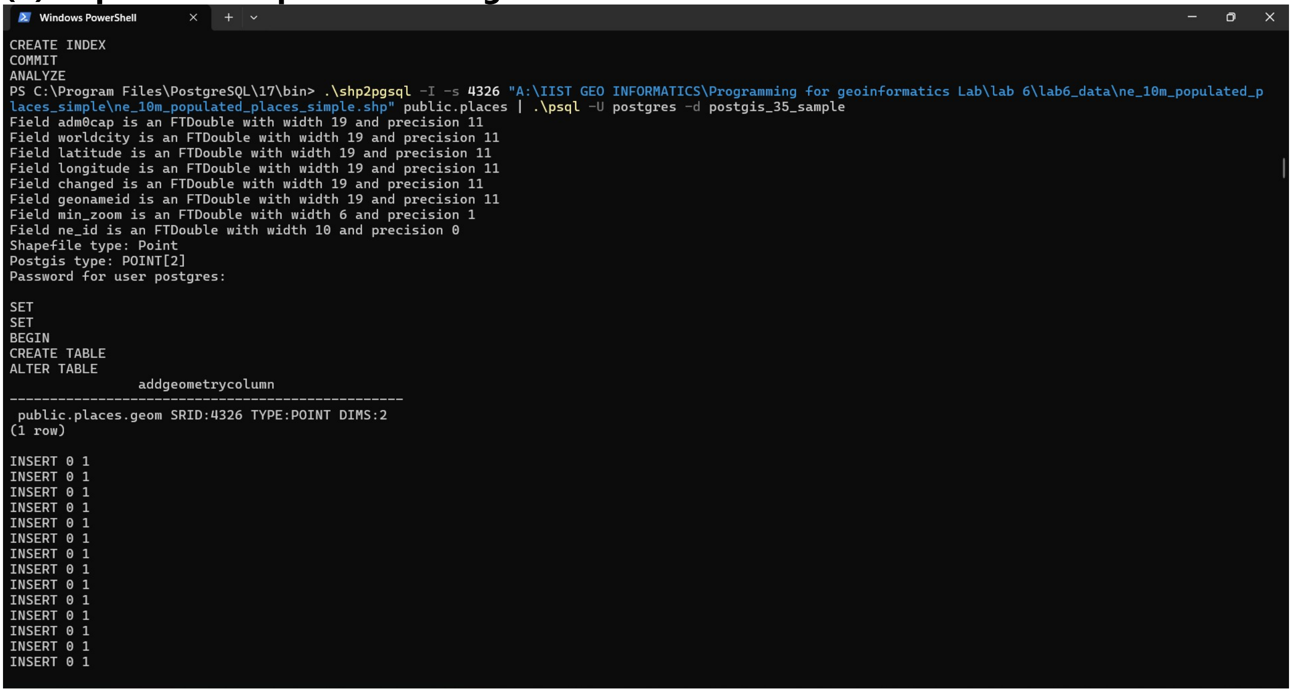
```
SELECT * FROM public.geometry_columns
```

	f_table_catalog character varying (256)	f_table_schema name	f_table_name name	f_geometry_column name	coord_dimension integer	srid integer	type character varying (3
1	postgis_35_sample	tiger	county	the_geom	2	4269	MULTIPOLYGON
2	postgis_35_sample	tiger	state	the_geom	2	4269	MULTIPOLYGON
3	postgis_35_sample	tiger	place	the_geom	2	4269	MULTIPOLYGON
4	postgis_35_sample	tiger	cousub	the_geom	2	4269	MULTIPOLYGON
5	postgis_35_sample	tiger	edges	the_geom	2	4269	MULTILINESTRING
6	postgis_35_sample	tiger	addrfeat	the_geom	2	4269	LINESTRING
7	postgis_35_sample	tiger	faces	the_geom	2	4269	MULTIPOLYGON
8	postgis_35_sample	tiger	zcta5	the_geom	2	4269	MULTIPOLYGON
9	postgis_35_sample	tiger	tabblock20	the_geom	2	4269	MULTIPOLYGON
10	postgis_35_sample	tiger	tract	the_geom	2	4269	MULTIPOLYGON
11	postgis_35_sample	tiger	tabblock	the_geom	2	4269	MULTIPOLYGON
12	postgis_35_sample	tiger	bg	the_geom	2	4269	MULTIPOLYGON

Total rows: 15 of 15 Query complete 00:00:00.207 Ln 1, Col 1

(c) LAB Use the shapefiles given (countries, river, and populated places) for the following query

(d) Import the shape files into Pgadmin

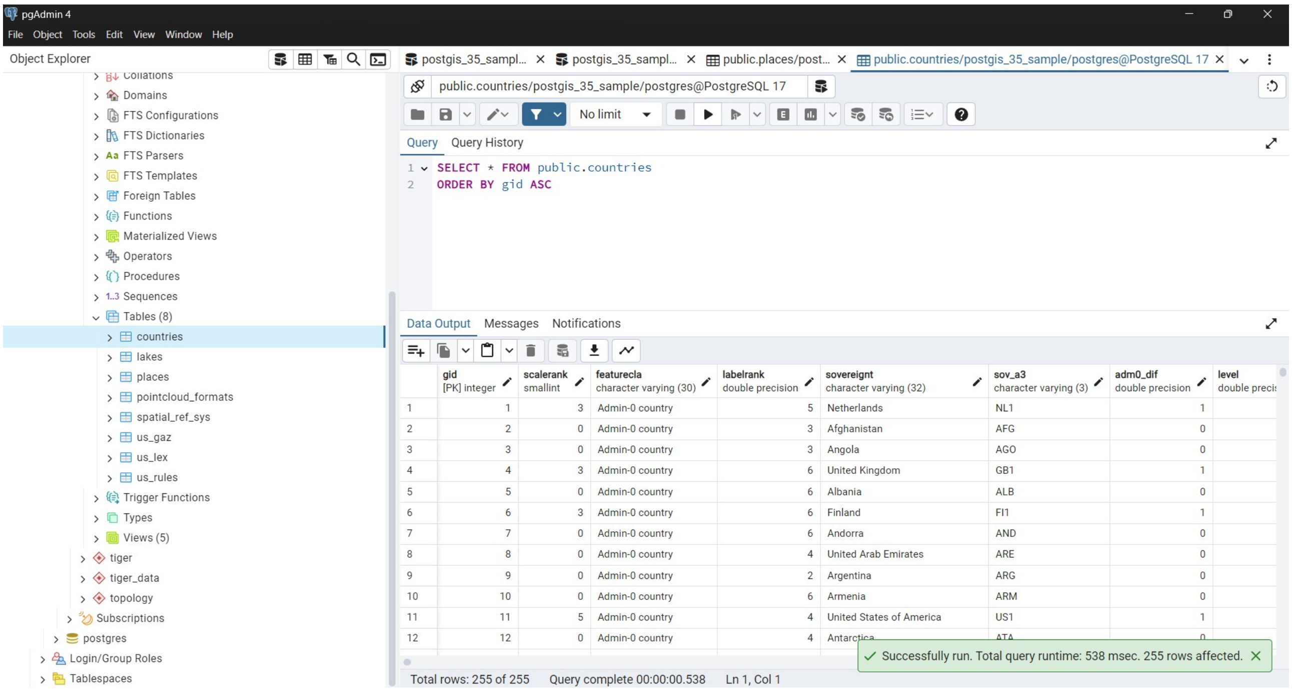


```
CREATE INDEX
COMMIT
ANALYZE
PS C:\Program Files\PostgreSQL\17\bin> .\shp2pgsql -I -s 4326 "A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab 6\lab6_data\ne_10m_populated_places_simple\ne_10m_populated_places_simple.shp" public.places | .\psql -U postgres -d postgis_35_sample
Field adm0cap is an FTDouble with width 19 and precision 11
Field worldcity is an FTDouble with width 19 and precision 11
Field latitude is an FTDouble with width 19 and precision 11
Field longitude is an FTDouble with width 19 and precision 11
Field changed is an FTDouble with width 19 and precision 11
Field geonameid is an FTDouble with width 19 and precision 11
Field min_zoom is an FTDouble with width 6 and precision 1
Field ne_id is an FTDouble with width 10 and precision 0
Shapefile type: Point
Postgis type: POINT[2]
Password for user postgres:

SET
SET
BEGIN
CREATE TABLE
ALTER TABLE
                addgeometrycolumn
--------------------------------------------------
 public.places.geom SRID:4326 TYPE:POINT DIMS:2
(1 row)

INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
INSERT 0 1
```

2. Explore the attribute



public.countries/postgis_35_sample/postgres@PostgreSQL 17

```
SELECT * FROM public.countries
ORDER BY gid ASC
```

	gid [PK] integer	scalerank smallint	featurecla character varying (30)	labelrank double precision	sovereignt character varying (32)	sov_a3 character varying (3)	adm0_dif double precision	level double preci
1	1	3	Admin-0 country	5	Netherlands	NL1	1	
2	2	0	Admin-0 country	3	Afghanistan	AFG	0	
3	3	0	Admin-0 country	3	Angola	AGO	0	
4	4	3	Admin-0 country	6	United Kingdom	GB1	1	
5	5	0	Admin-0 country	6	Albania	ALB	0	
6	6	3	Admin-0 country	6	Finland	FI1	1	
7	7	0	Admin-0 country	6	Andorra	AND	0	
8	8	0	Admin-0 country	4	United Arab Emirates	ARE	0	
9	9	0	Admin-0 country	2	Argentina	ARG	0	
10	10	0	Admin-0 country	6	Armenia	ARM	0	
11	11	5	Admin-0 country	4	United States of America	US1	1	
12	12	0	Admin-0 country	4	Antarctica	ATA	0	

✓ Successfully run. Total query runtime: 538 msec. 255 rows affected.

Total rows: 255 of 255 Query complete 00:00:00.538 Ln 1, Col 1

3. Write SQL queries

a) Find the total number of countries and order it alphabetically Later, display the names in such a way that countries get grouped alphabetically.

```sql
SELECT DISTINCT(sovereignt)
FROM countries
ORDER BY sovereignt ASC
```

```sql
SELECT LEFT(sovereignt,1) AS startingletter, array_agg(distinct(sovereignt)) AS country_count
FROM countries
GROUP BY LEFT(sovereignt,1) ORDER BY LEFT (sovereignt,1)
```

b) Find the number of populated cities within your choice of country(excluding India) listed in the given data

```sql
SELECT name,pop_max FROM places
WHERE pop_max = (SELECT MAX(pop_max) FROM places
WHERE sov0name = 'United States' GROUP BY sov0name)
```

c) Which is the most populous city in India, China and USA

```sql
SELECT name,pop_max FROM places
WHERE pop_max = (SELECT MAX(pop_max) FROM places
WHERE sov0name = 'India' GROUP BY sov0name)
```

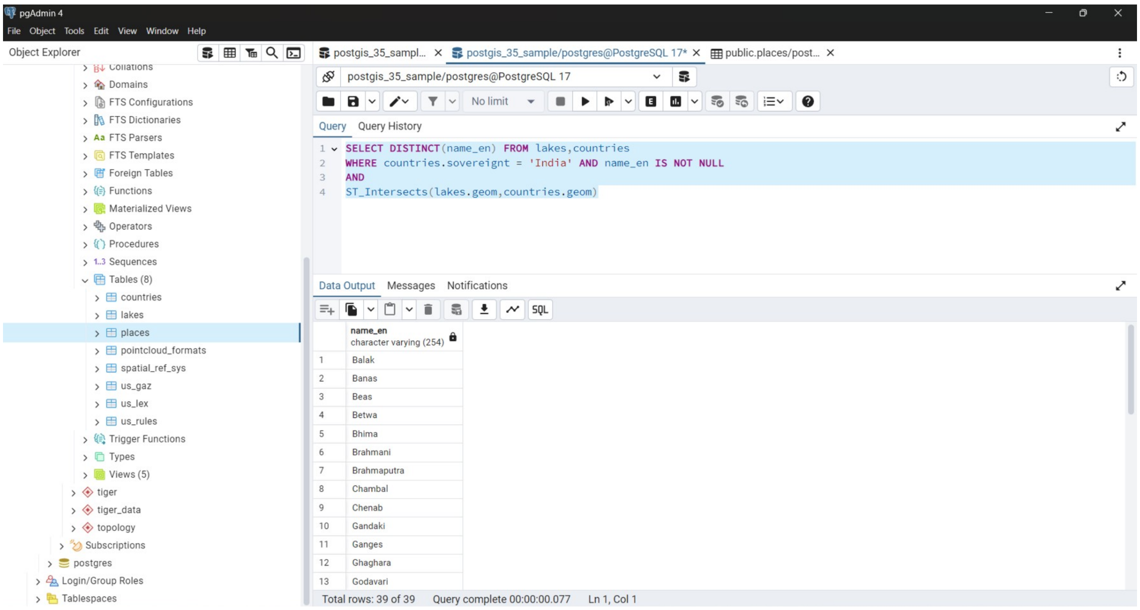
```sql
SELECT name,pop_max FROM places
WHERE pop_max = (SELECT MAX(pop_max) FROM places
WHERE sov0name = 'China' GROUP BY sov0name)
```

```sql
SELECT name,pop_max FROM places
WHERE pop_max = (SELECT MAX(pop_max) FROM places
WHERE sov0name = 'United States' GROUP BY sov0name)
```

d) Find the rivers which flow through India

```sql
SELECT DISTINCT(name_en) FROM lakes,countries
WHERE countries.sovereignt = 'India' AND name_en IS NOT NULL
AND
ST_Intersects(lakes.geom,countries.geom)
```

OUTPUT



```sql
SELECT DISTINCT(name_en) FROM lakes,countries
WHERE countries.sovereignt = 'India' AND name_en IS NOT NULL
AND
ST_Intersects(lakes.geom,countries.geom)
```

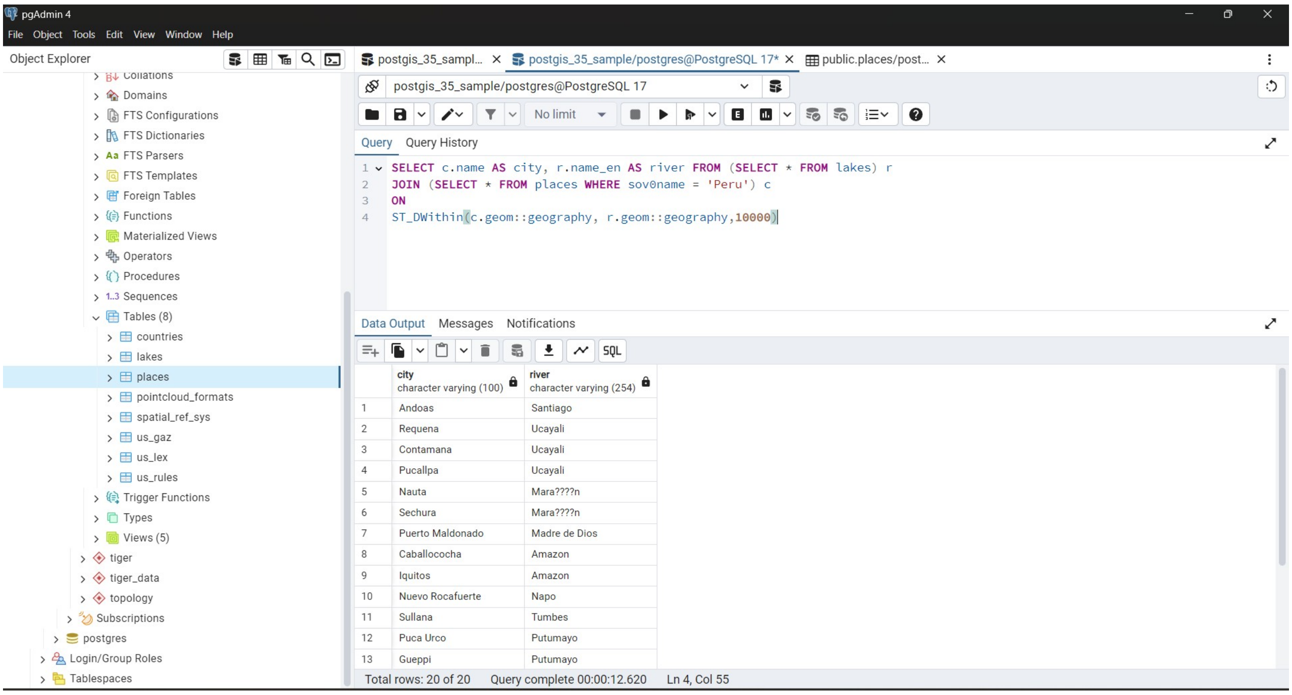
	name_en character varying (254)
1	Balak
2	Banas
3	Beas
4	Betwa
5	Bhima
6	Brahmani
7	Brahmaputra
8	Chambal
9	Chenab
10	Gandaki
11	Ganges
12	Ghaghara
13	Godavari

Total rows: 39 of 39 Query complete 00:00:00.077 Ln 1, Col 1

e) Find all cities that are within 10 kms from a river.

```sql
SELECT c.name AS city, r.name_en AS river FROM (SELECT * FROM lakes) r
JOIN (SELECT * FROM places WHERE sov0name = 'Peru') c
ON
ST_DWithin(c.geom::geography, r.geom::geography,10000)
```

OUTPUT



```sql
SELECT c.name AS city, r.name_en AS river FROM (SELECT * FROM lakes) r
JOIN (SELECT * FROM places WHERE sov0name = 'Peru') c
ON
ST_DWithin(c.geom::geography, r.geom::geography,10000)
```

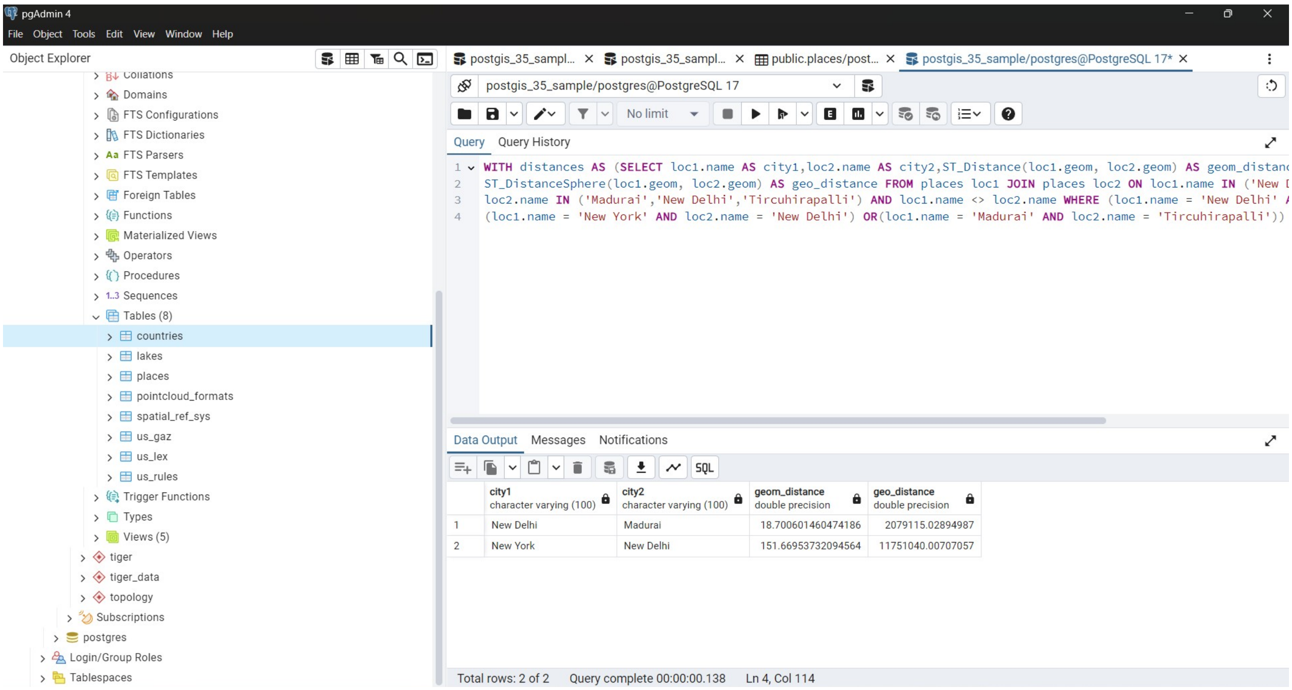
	city character varying (100)	river character varying (254)
1	Andoas	Santiago
2	Requena	Ucayali
3	Contamana	Ucayali
4	Pucallpa	Ucayali
5	Nauta	Mara????n
6	Sechura	Mara????n
7	Puerto Maldonado	Madre de Dios
8	Caballococha	Amazon
9	Iquitos	Amazon
10	Nuevo Rocafuerte	Napo
11	Sullana	Tumbes
12	Puca Urco	Putumayo
13	Gueppi	Putumayo

Total rows: 20 of 20 Query complete 00:00:12.620 Ln 4, Col 55

f) Find the distance between a) New Delhi and Madurai b) New York and New Delhi c) Madurai and Trichy (report in terms of geography and Geometry

```
WITH distances AS (
SELECT loc1.name AS city1,
loc2.name AS city2,
ST_Distance(loc1.geom, loc2.geom) AS geom_distance,
ST_DistanceSphere(loc1.geom, loc2.geom) AS geo_distance
FROM
places loc1
JOIN
places loc2
ON
loc1.name IN ('New Delhi','New York','Madurai') AND
loc2.name IN ('Madurai','New Delhi','Tircuhirapalli') AND
loc1.name <> loc2.name
WHERE
(loc1.name = 'New Delhi' AND loc2.name = 'Madurai') OR
(loc1.name = 'New York' AND loc2.name = 'New Delhi') OR
(loc1.name = 'Madurai' AND loc2.name = 'Tircuhirapalli'))
SELECT * FROM distances
```

OUTPUT



```
WITH distances AS (SELECT loc1.name AS city1,loc2.name AS city2,ST_Distance(loc1.geom, loc2.geom) AS geom_distanc
ST_DistanceSphere(loc1.geom, loc2.geom) AS geo_distance FROM places loc1 JOIN places loc2 ON loc1.name IN ('New D
loc2.name IN ('Madurai','New Delhi','Tircuhirapalli') AND loc1.name <> loc2.name WHERE (loc1.name = 'New Delhi' A
(loc1.name = 'New York' AND loc2.name = 'New Delhi') OR(loc1.name = 'Madurai' AND loc2.name = 'Tircuhirapalli'))
```

	city1 character varying (100)	city2 character varying (100)	geom_distance double precision	geo_distance double precision
1	New Delhi	Madurai	18.700601460474186	2079115.02894987
2	New York	New Delhi	151.66953732094564	11751040.00707057

Total rows: 2 of 2 Query complete 00:00:00.138 Ln 4, Col 114

4. Write your inference

Implemented PostGIS in PGSQL ,imported shape files and executed spatial queries in the same.