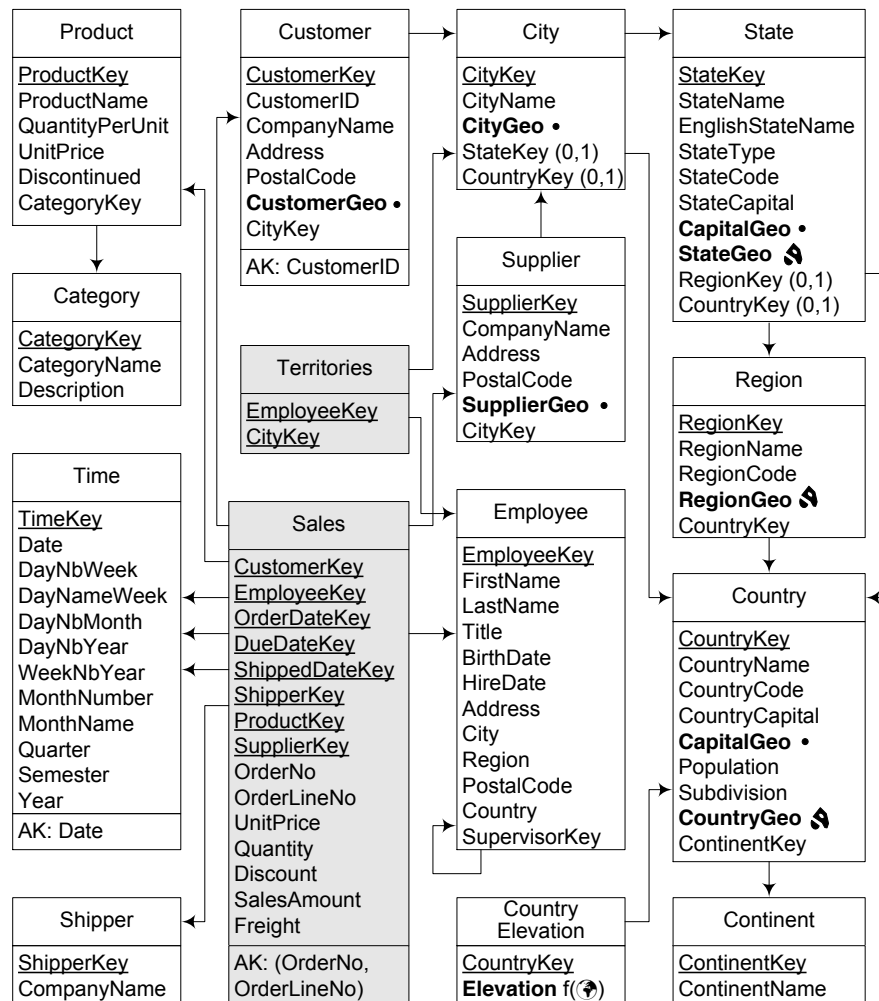


## Session 12 - Spatial Databases (3/3)

For this session, we consider the following spatial relational diagram of the GeoNorthwind database.



You can load this database using the following commands:

```
$ createdb NW
$ pg_restore -d NW -x -O geonorthwind.backup
```

The restoration process could lead to some unimportant errors. Please check that you have a database loaded by exploring it with psql.

For this exercise session, we provided a “spatial reference guide” which describe the different functions that you may need.

**Exercise 1.** Give the total sales in 1997 to customers located in cities that are within an area whose extent is a polygon drawn by the user. For the purpose of the exercise, we will consider the following coordinates for the polygon's opposite corners: (5, 40) and (100, 90).

You can create the rectangle with

```
| 'SRID=4326;POLYGON((5.0 40.0, 100 40.0, 100.0 90.0, 5.0 90.0, 5.0 40.0))'::geometry
```

For instance:

```
| SELECT ST_contains('SRID=4326;POLYGON((5 40, 100 40, 100 90, 5 9, 5 40))'::geometry,  
| 'SRID=4326;POLYGON((0 0, 200 0, 200 200, 0 200, 0 0))'::geometry)
```

Will return true

**Exercise 2.** What are total sales to customers located in a state that contains the capital city of the country?

**Exercise 3.** Give the spatial union of the states in the USA where at least one customer placed an order in 1997.

**Exercise 4.** What is the distance between the customers' locations and the capital of the state in which they are located?

**Exercise 5.** For each customer, give the total sales amount to its closest supplier.

**Exercise 6.** Give the total sales amount for customers that have orders delivered by suppliers such that their locations are less than 200 km from each other.

**Exercise 7.** What is the distance between the customer and supplier for customers that have orders delivered by suppliers of the same country.

**Exercise 8.** Give the number of customers for European countries with an area larger than 50,000 km<sup>2</sup>.

*Note: the database provided during the lab does referer to the table "Continent" (on the schema) as "Area"*

**Exercise 9.** For each supplier, give the number of customers located at more than 100 km from the supplier.

**Exercise 10.** For each supplier, give the distance between its location and the centroid of the locations of all its customers.

## Solutions for Session 12 - Spatial Databases (3/3)

---

### ➤ Solution to Exercise 1

```

SELECT C.CompanyName AS Customer,
       Y.CityName AS City,
       SUM(S.SalesAmount) AS TotalSales
FROM Sales S,
     Customer C,
     City Y,
     Time T
WHERE S.CustomerKey = C.CustomerKey
     AND C.CityKey = Y.CityKey
     AND S.OrderDateKey = T.TimeKey
     AND T.Year = '1997'
     AND ST_Within(Y.CityGeo, 'SRID=4326;POLYGON((5.0 40.0, 100 40.0,
                                                    100.0 90.0, 5.0 90.0, 5.0 40.0))'::geometry)
GROUP BY C.CompanyName, Y.CityName

```

Beware that you have to specify the SRID of the polygon and to “cast” the Geography-typed result to a Geometry type (with the `::geometry` statement) because the functions `ST_Within` and `ST_Contains` only accept geometry.

To check your results, you could add the following column to the query: `ST_AsText(Y.CityGeo) AS City`. Note that you could also use the `ST_GeographyFromText` function to interpret the polygon string into a geography, but you still would have to cast it into a geometry type. Practically, you could replace

```
'SRID=4326;POLYGON((5.0 40.0, 100 40.0, 100.0 90.0, 5.0 90.0, 5.0 40.0))'::geometry
```

by

```
ST_GeomFromText('SRID=4326;POLYGON((5.0 40.0, 100 40.0,
                                     100.0 90.0, 5.0 90.0, 5.0 40.0))')
```

### ➤ Solution to Exercise 2

```

SELECT C.CompanyName AS Customer,
       SUM(S.SalesAmount) AS TotalSales
FROM Sales S,
     Customer C,
     City Y,
     State A,
     Country O
WHERE S.CustomerKey = C.CustomerKey
     AND C.CityKey = Y.CityKey
     AND Y.StateKey = A.StateKey
     AND A.CountryKey = O.CountryKey
     AND ST_Contains(A.StateGeo, O.CapitalGeo)
GROUP BY C.CompanyName

```

Note that, like for the previous exercise, you could also use the `ST_Within` function (and swap the parameters).

## ➤ Solution to Exercise 3

```
SELECT ST_AsText(ST_Union(DISTINCT A.StateGeo)) AS States
FROM Sales S,
     Customer C,
     City Y,
     State A,
     Country O,
     Time T
WHERE S.CustomerKey = C.CustomerKey
     AND C.CityKey = Y.CityKey
     AND Y.StateKey = A.StateKey
     AND A.CountryKey = O.CountryKey
     AND O.CountryName = 'United States'
     AND S.OrderDateKey = T.TimeKey
     AND T.Year = '1997'
```

## ➤ Solution to Exercise 4

```
SELECT C.CompanyName AS Customer,
       ST_Distance(C.CustomerGeo,A.CapitalGeo) AS Distance
FROM Customer C,
     City Y,
     State A
WHERE C.CityKey = Y.CityKey
     AND Y.StateKey = A.StateKey
ORDER BY C.CompanyName
```

## ➤ Solution to Exercise 5

```
SELECT C.CompanyName AS Customer,
       SUM(S.SalesAmount) AS TotalSales
FROM Sales S,
     Customer C,
     Supplier U
WHERE S.CustomerKey = C.CustomerKey
     AND S.SupplierKey = U.SupplierKey
     AND ST_Distance(C.CustomerGeo,U.SupplierGeo)
         <= (SELECT MIN(ST_Distance(C.CustomerGeo,U1.SupplierGeo))
            FROM Sales S1,
                 Supplier U1
            WHERE S1.CustomerKey = C.CustomerKey
                AND S1.SupplierKey = U1.SupplierKey)
GROUP BY C.CompanyName
```

Note that this query is assuming that we cannot have two closest suppliers for any customer. Should we want to take this case into account, then the query could be extended as follows:

```
SELECT C.CompanyName AS Customer,
       U.CompanyName AS Supplier,
       SUM(S.SalesAmount) AS TotalSales
```

```

FROM Sales S,
     Customer C,
     Supplier U
WHERE S.CustomerKey = C.CustomerKey
     AND S.SupplierKey = U.SupplierKey
     AND ST_Distance(C.CustomerGeo,U.SupplierGeo)
         <= (SELECT MIN(ST_Distance(C.CustomerGeo,U1.SupplierGeo))
              FROM Sales S1,
              Supplier U1
              WHERE S1.CustomerKey = C.CustomerKey
                 AND S1.SupplierKey = U1.SupplierKey)
GROUP BY C.CompanyName, U.CompanyName

```

## ➤ Solution to Exercise 6

```

SELECT C.CompanyName AS Customer,
       SUM(S.SalesAmount) AS TotalSales
FROM Sales S,
     Customer C,
     Supplier U
WHERE S.CustomerKey = C.CustomerKey
     AND S.SupplierKey = U.SupplierKey
     AND ST_Distance(C.CustomerGeo,U.SupplierGeo) < 200
GROUP BY C.CompanyName

```

## ➤ Solution to Exercise 7

```

SELECT DISTINCT C.CompanyName AS Customer,
               U.CompanyName AS Supplier,
               ST_Distance(C.CustomerGeo,U.SupplierGeo) AS Distance
FROM Sales S,
     Customer C,
     City Y,
     State A,
     Supplier U,
     City AS SY,
     State AS SA
WHERE S.CustomerKey = C.CustomerKey
     AND C.CityKey = Y.CityKey
     AND Y.StateKey = A.StateKey
     AND S.SupplierKey = U.SupplierKey
     AND U.CityKey = SY.CityKey
     AND SY.StateKey = SA.StateKey
     AND SA.CountryKey = A.CountryKey
ORDER BY C.CompanyName, U.CompanyName

```

## ➤ Solution to Exercise 8

```
SELECT O.CountryName AS Country,
       COUNT(DISTINCT S.CustomerKey) AS NbCustomers
FROM Sales S,
       Customer C,
       City Y,
       State A,
       Country O,
       Area R
WHERE S.CustomerKey = C.CustomerKey
      AND C.CityKey = Y.CityKey
      AND Y.StateKey = A.StateKey
      AND A.CountryKey = O.CountryKey
      AND ST_Area(O.CountryGeo) > 5
      AND O.AreaKey = R.AreaKey
      AND TRIM(R.AreaName) = 'Europe'
GROUP BY O.CountryName
```

The table “Area” can be referred as “Conitinent” if the exercise is done on paper.

## ➤ Solution to Exercise 9

```
SELECT U.CompanyName AS Supplier,
       COUNT(DISTINCT C.CustomerKey) AS CustomerCount
FROM Sales S,
       Supplier U,
       Customer C
WHERE S.SupplierKey = U.SupplierKey
      AND S.CustomerKey = C.CustomerKey
      AND ST_Distance(U.SupplierGeo, C.CustomerGeo) > 100
GROUP BY U.CompanyName
```

## ➤ Solution to Exercise 10

```
SELECT U.CompanyName AS Supplier,
       ST_Distance(U.SupplierGeo, ST_Centroid(ST_Union(DISTINCT C.CustomerGeo))) AS Dist
FROM Sales S,
       Supplier U,
       Customer C
WHERE S.SupplierKey = U.SupplierKey
      AND S.CustomerKey = C.CustomerKey
GROUP BY U.CompanyName,
         U.SupplierGeo
```