Assignment 3

Tianle Yuan

04/14/2023

-- All scenarios are based on Database NORTHWND.

```
USE Northwind
GO
```

1. **List all cities that have both Employees and Customers.**

```
SELECT DISTINCT E.City
FROM Customers C INNER JOIN Employees E ON C.City = E.City
ORDER BY E.City
```

2. **List all cities that have Customers but no Employee.**

   (1) Use sub-query

```
SELECT DISTINCT C.City
FROM Customers C LEFT JOIN Employees E ON C.City = E.City
WHERE E.City IS NULL
ORDER BY C.City
```

   (2) Do not use sub-query

```
SELECT DISTINCT C.City
FROM Customers C
WHERE C.City NOT IN(
    SELECT E.City
    FROM Employees E
)
ORDER BY C.City
```

3. **List all products and their total order quantities throughout all orders.**

```
SELECT P.ProductName, SUM(OD.OrderID) AS TOQuantities
FROM Products P INNER JOIN [Order Details] OD ON P.ProductID = OD.ProductID
GROUP BY P.ProductName
ORDER BY TOQuantities DESC;
```

4. **List all Customer Cities and total products ordered by that city.**

```
SELECT C.City, SUM(OD.OrderID) AS TOQuantities
FROM Products P JOIN [Order Details] OD ON P.ProductID = OD.ProductID
JOIN Orders O ON OD.OrderID = O.OrderID
JOIN Customers C ON C.CustomerID = O.CustomerID
GROUP BY C.City
ORDER BY TOQuantities DESC;
```

5. List all Customer Cities that have at least two customers.

(1) Use union

```
SELECT city FROM customers
EXCEPT
SELECT city FROM customers
GROUP BY city
HAVING COUNT(*)=1
UNION
SELECT city FROM customers
GROUP BY city
HAVING COUNT(*)=0
```

(2) Use sub-query and no union

```
select city from customers group by city having COUNT(*)>=2
```

6. List all Customer Cities that have ordered at least two different kinds of products.

```
SELECT DISTINCT City
FROM Customers
WHERE CustomerID IN (
    SELECT CustomerID
    FROM Orders
    WHERE OrderID IN (
        SELECT OrderID
        FROM [Order Details]
        GROUP BY OrderID
        HAVING COUNT(DISTINCT ProductID) >= 2
    )
)
ORDER BY City;
```

7. List all Customers who have ordered products, but have the 'ship city' on the order different from their own customer cities.

```
SELECT DISTINCT C.CustomerID, C.CompanyName
FROM Customers C
JOIN Orders O ON C.CustomerID = O.CustomerID
WHERE O.OrderID IN (
    SELECT OrderID
    FROM [Order Details]
    WHERE ShipCity <> C.City
)
ORDER BY C.CustomerID;
```

8. List 5 most popular products, their average price, and the customer city that ordered most quantity of it.

```
SELECT TOP 5 P.ProductName, AVG(OD1.UnitPrice) AS AvgPrice,
    (
        SELECT TOP 1 C.City
        FROM Customers C
        JOIN Orders O ON C.CustomerID = O.CustomerID
        JOIN [Order Details] OD2 ON O.OrderID = OD2.OrderID
        WHERE OD2.ProductID = OD1.ProductID
        GROUP BY C.City
        ORDER BY SUM(OD2.Quantity) DESC
    ) AS City
FROM [Order Details] OD1
JOIN Products P ON OD1.ProductID = P.ProductID
GROUP BY P.ProductName, OD1.ProductID
ORDER BY SUM(OD1.Quantity) DESC;
```

9. List all cities that have never ordered something but we have employees there.

(1) Use sub-query

```
SELECT DISTINCT City
FROM Employees
WHERE City NOT IN(
    SELECT O.ShipCity
    FROM Orders O
    WHERE O.ShipCity IS NOT NULL
)
```

(2) Do not use sub-query

```
SELECT DISTINCT City
FROM Employees E LEFT JOIN Orders O ON E.City=O.ShipCity
WHERE O.ShipCity IS NULL
ORDER BY E.City
```

10. List one city, if exists, that is the city from where the employee sold most orders (not the product quantity) is, and also the city of most total quantity of products ordered from. (tip: join sub-query)

```
SELECT
(
    SELECT TOP 1 City
    FROM Orders O JOIN [Order Details] OD ON O.OrderID=OD.OrderID
    JOIN Employees E ON e.EmployeeID = O.EmployeeID
    GROUP BY E.EmployeeID, E.City
    ORDER BY COUNT(*) DESC
) AS MostOrderedCity,
(
    SELECT TOP 1 City
    FROM Orders O JOIN [Order Details] OD ON O.OrderID=OD.OrderID
    JOIN Employees E ON E.EmployeeID = O.EmployeeID
    GROUP BY E.EmployeeID,E.City
    ORDER BY sum(Quantity) DESC
) AS MostQunatitySoldCity
```

11. How do you remove the duplicates record of a table?

```
SELECT DISTINCT *
INTO NewTable
FROM OldTable

DROP TABLE OldTable

EXEC sp_rename 'NewTable', 'OriginalTable'
```