Assignment Day3 –SQL: Comprehensive practice

# Answer following questions

1. In SQL Server, assuming you can find the result by using both joins and subqueries, which one would you prefer to use and why?  
   - Joins is preferred because perforemce is high.
2. What is CTE and when to use it?  
   - A CTE is temperory view. We can use CTE for recursive queries. We can also use CTE if the scope of a table is very limited.
3. What are Table Variables? What is their scope and where are they created in SQL Server?  
   - A table variable is a data type that can be used within a Transact-SQL batch, stored procedure, or function—and is created and defined similarly to a table, only with a strictly defined lifetime scope. The lifetime of the table variable only lasts for the duration of the batch, function, or stored procedure Table variables will be created in temp db.
4. What is the difference between DELETE and TRUNCATE? Which one will have better performance and why?  
   - Delete will keep all deleted rows in log while truncate permenently delets records. Truncate will have more performence.
5. What is Identity column? How does DELETE and TRUNCATE affect it?  
   - Identity column is autogenerated column in a table. Identity column will will not get reset to initial value incase of delete but truncate will reset value to its initial value.
6. What is difference between “delete from table\_name” and “truncate table table\_name”?  
   - The DELETE command is used to remove rows from a table. A WHERE clause can be used to only remove some rows. TRUNCATE removes all rows from a table. The operation cannot be rolled back and no triggers will be fired.

# Write queries for following scenarios

All scenarios are based on Database NORTHWND.

1. List all cities that have both Employees and Customers.  
   - select distinct city from Customers where city in (select city from Employees)
2. List all cities that have Customers but no Employee.
   1. Use sub-query

- select distinct city from Customers   
where City not in (select distinct city from employees where city is not null)

* 1. where City not in (select distinct city from employees where city is not null)
* select distinct city from Customers  
  except  
  select distinct city from Employees

1. List all products and their total order quantities throughout all orders.  
   - select ProductID,SUM(Quantity) as QunatityOrdered from [order details]  
   group by ProductID
2. List all Customer Cities and total products ordered by that city.  
   - select city,sum(Quantity) as TotalQty from orders o join [order details] od on o.orderid=od.orderid join customers c on c.customerid=o.CustomerID  
   group by city
3. 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

* 1. Use sub-query and no union  
     - select distinct city from customers where city in (select City from customers group by city having COUNT(\*)>=2)

1. List all Customer Cities that have ordered at least two different kinds of products.  
   - select distinct city from orders o join [order details] od on o.orderid=od.orderid join customers c on c.customerid=o.CustomerID  
   group by city,ProductID  
   having COUNT(\*)>=2
2. 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 from orders o join [order details] od on o.orderid=od.orderid join customers c on c.customerid=o.CustomerID  
   where City <> ShipCity
3. List 5 most popular products, their average price, and the customer city that ordered most quantity of it.  
   - select top 5 ProductID,AVG(UnitPrice) as AvgPrice,(select top 1 City from Customers c join Orders o on o.CustomerID=c.CustomerID join [Order Details] od2 on od2.OrderID=o.OrderID where od2.ProductID=od1.ProductID group by city order by SUM(Quantity) desc) as City from [Order Details] od1  
   group by ProductID  
   order by sum(Quantity) desc
4. 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 ShipCity from Orders where ShipCity is not null)
   2. Do not use sub-query  
      - select distinct City from Employees where City is not null except (select ShipCity from Orders where ShipCity is not null)
5. 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
6. How do you remove the duplicates record of a table?  
   - use group by and count(\*), if count(\*)>1 then delete the rows using sub query

12. Sample table to be used for solutions below- Employee ( empid integer, mgrid integer, deptid integer, salary integer) Dept (deptid integer, deptname text)

Find employees who do not manage anybody.  
- select empid from Employee except select mgrid from Employee

13. Find departments that have maximum number of employees. (solution should consider scenario having more than 1 departments that have maximum number of employees). Result should only have - deptname, count of employees sorted by deptname.  
  
- select deptid from employee group by deptid having count(\*) = (select top 1 count(\*) from employee group by deptid order by count(\*) desc)

14. Find top 3 employees (salary based) in every department. Result should have deptname, empid, salary sorted by deptname and then employee with high to low salary.

- select top 3 deptname,empid,salary from employee e join dep d on e.deptid=d.deptid order by salary,deptname,empid desc

GOOD LUCK.