1. If both joins and subqueries are works. I prefer to use joins. Cause the inner query result will use as the input of the outer query, so the performance is not as good as joins.

2. CTE stands for Common Table Expressions. It is means the result set of a query which exists temporarily and for use only within the context of next query. When a view is not required, we can use CTE for substitution. And CTE is also used when to create a recursive query.

3. Table variable is created and defined similarly to a table, and it is also stored in tempdb database. But table variable only with a strictly defined lifetime scope. Basically, when the data set is not very large, we will consider to use the table variable instead of temporary table.

4. DELETE is used to delete specified rows, and TRUNCATE is used to delete all the rows from a table. Since DELETE removes rows one at a time and it will record the deleted row in the transaction log. TRUNCATE will only record the deleted data page in the transaction log. So the performance of TRUNCATE is better.

5. Identity column is used to automatically generate key values. The syntax of Identity column includes a provided seed (starting point) and an increment. And Identity column retains the identity after using DELETE statement on table. However, Identity column will reset to its seed value after using TRUNCATE statement.

6. When we want to delete certain row or rows, we can use“delete from table\_name” and followed with a WHERE statement to specify the condition. However, TRUNCATE is used to delete all the rows, so “truncate table table\_name” may not be followed with a WHERE statement.

11. Today we learned ROW\_NUMBER, by combine with “PARTITION BY” we can also retrieve rows which have more than 1 ROW count. As Q17 shows how to use it to delete duplicates.

12.

SELECT e.empid

FROM Employee e

WHERE e.empid NOT IN (

SELECT m.mgrid

FROM Employee m

)

13.

SELECT MaxDept.depname, MaxDept.totalCount

FROM (

SELECT d.depname, e.depid, COUNT(e.empid) AS totalCount, DENSE\_RANK() OVER(ORDER BY COUNT(e.empid) DESC) AS rnk

FROM Employee e

JOIN Dept d

ON e.depid = d.depid

GROUP BY d.depname, e.depid

) MaxDept

WHERE MaxDept.rnk <= 1

14.

SELECT MaxSalary.depname, MaxSalary.empid, MaxSalary.salary

FROM (

SELECT d.depname, e.empid, e.depid, e.salary, DENSE\_RANK() OVER(PARTITION BY e.depid ORDER BY e.salary DESC) AS rnk

FROM Employee e

INNER JOIN Dept d

ON e.depid = d.depid

) AS MaxSalary

WHERE MaxSalary.rnk <= 3

15.

;WITH cte

AS(

SELECT \*, ROW\_NUMBER() OVER(ORDER BY City DESC) AS RowNum

FROM CityDistance

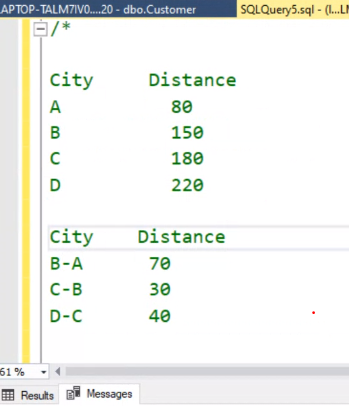
)

SELECT b.City + ‘ ’ + a.City AS City, (b.Distance – a.Distance) AS Distance

FROM cte a

LEFT JOIN cte b

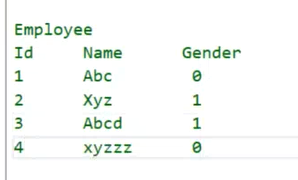
ON a.RowNum = b.RowNum – 1



16.

UPDATE Employee

SET Gender = Gender ^ 1



17.

;WITH cte

AS (

SELECT Name, Gender, ROW\_NUMBER() OVER (PARTITION BY Name, Gender ORDER BY Name, Gender) AS RowNum

FROM Customer

)

DELETE FROM cte WHERE RowNum > 1

