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?

A: In most of the cases, join is preferred because it save runtime.

1. What is CTE and when to use it?

A:

CTE stands for: common table expression. A Common Table Expression (CTE) is the result set of a query which exists temporarily and for use only within the context of a larger query. Much like a derived table, the result of a CTE is not stored and exists only for the duration of the query.

CTEs, like database views and derived tables, enable users to more easily write and maintain complex queries via increased readability and simplification. This reduction in complexity is achieved by deconstructing ordinarily complex queries into simple blocks to be used, and reused if necessary, in rewriting the query. Example use cases include:

1) Needing to reference a derived table multiple times in a single query

2) An alternative to creating a view in the database

3) Performing the same calculation multiple times over across multiple query components

1. What are Table Variables? What is their scope and where are they created in SQL Server?

A:

There are global table variables and local table variables. They are stored in tempdb, but one can only be accessed by location and the other can be accessed by other sections. Batch is declared. For rows less than 100 recommend using table variables.

1. What is the difference between DELETE and TRUNCATE? Which one will have better performance and why?

A:

1. Truncate reseeds identity values, whereas delete doesn't.
2. Truncate removes all records and doesn't fire triggers.
3. Truncate is faster compared to delete as it makes less use of the transaction log.
4. Truncate is not possible when a table is referenced by a Foreign Key or tables are used in replication or with indexed views.

(Seed: Starting value of a column. Default value is 1.)

1. What is Identity column? How does DELETE and TRUNCATE affect it?

Identity column of a table is a column whose value increases automatically. The value in an identity column is created by the server. A user generally cannot insert a value into an identity column. Identity column can be used to uniquely identify the rows in the table. Truncate reseeds identity values, whereas delete doesn't.

1. What is difference between “delete from table\_name” and “truncate table table\_name”?

Delete is more use to delete rows while truncate is for delete the whole table. However, delete without where clause works similarly as truncate table. Delete can aslo be retrieved and returns the number of rows that are deleted. Also, delete table deletes row one by one and may fires triggers. Truncate cannot be retrieved and delete the whole table at once and recreate a new one. Truncate table runs faster than delete table(I assume).

TRUNCATE TABLE differs from DELETE in the following ways :

1. TRUNCATE drops the table and re-create it. It is much faster than deleting rows one by one.
2. Once you use TRUNCATE command then u cannot retrieve the data again from the table.
3. TRUNCATE is not transaction-safe; an error occurs when attempting one in the course of an active transaction or active table lock.
4. TRUNCATE removes all the rows from the Table.
5. TRUNCATE does not return the number of deleted rows.
6. TRUNCATE functionality is an Oracle SQL extension adopted in MySQL.
7. TRUNCATE is probably better thought of as a shortcut for DROP TABLE/CREATE TABLE rather than a quirky kind of DELETE.

# Write queries for following scenarios

All scenarios are based on Database NORTHWND.

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

SELECT distinct C.City

FROM Employees E JOIN Customers C ON E.City = C.City

1. List all cities that have Customers but no Employee.
   1. Use sub-query

SELECT DISTINCT C.CITY

FROM Customers C

WHERE C.CITY NOT IN

(SELECT DISTINCT E.City

FROM Employees E)

* 1. Do not use sub-query

SELECT DISTINCT C.City

FROM Customers C LEFT JOIN Employees E ON C.City = E.City

WHERE E.City IS NULL

1. List all products and their total order quantities throughout all orders.

SELECT P.ProductName, SUM(OD.Quantity) AS [Total Quantity]

FROM [Order Details] OD JOIN Products P

ON OD.ProductID = P.ProductID

GROUP BY P.ProductName

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

SELECT C.City, COUNT(P.ProductID) AS [Total Products]

FROM Customers C JOIN Orders O ON

C.CustomerID = O.CustomerID JOIN

[Order Details]OD ON O.OrderID =

OD.OrderID JOIN Products P ON

OD.ProductID = P.ProductID

GROUP BY C.City

1. List all Customer Cities that have at least two customers.
   1. Use union

SELECT City

FROM Customers a

GROUP BY City

HAVING COUNT(a.CustomerId) = 2

UNION

SELECT City

FROM Customers b

GROUP BY City

HAVING COUNT(b.CustomerId) > 2

* 1. Use sub-query and no union

SELECT B.CITY

FROM

(SELECT A.CITY, COUNT (CustomerID) AS PEOPLE

FROM Customers A

GROUP BY A.CITY) AS B

WHERE B.PEOPLE >= 2

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

SELECT C.City

FROM Customers C JOIN Orders O ON

C.CustomerID = O.CustomerID JOIN

[Order Details]OD ON O.OrderID =

OD.OrderID JOIN Products P ON

OD.ProductID = P.ProductID

GROUP BY C.City, P.ProductName

HAVING COUNT(P.ProductID) >= 2

1. List all Customers who have ordered products but have the ‘ship city’ on the order different from their own customer cities.

SELECT DISTINCT C.ContactName

FROM Orders O JOIN Customers C ON O.CustomerID = C.CustomerID

WHERE O.ShipCity != C.City

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

WITH MostPopProducts

AS(

SELECT TOP 5 P.ProductName AS [Most Popular Products], P.ProductID

FROM [Order Details] OD JOIN Products P ON P.ProductID = OD.ProductID

GROUP BY P.ProductName, p.ProductID

ORDER BY SUM(OD.Quantity) DESC

)

SELECT M.ProductID, OD2.UnitPrice, City

FROM Orders O JOIN [Order Details] OD2 ON O.OrderID = OD2.OrderID

JOIN MostPopProducts M ON OD2.ProductID = M.ProductID

JOIN Customers C ON C.CustomerID = O.CustomerID

WHERE M.ProductID = OD2.ProductID

GROUP BY M.ProductID, City, UnitPrice

ORDER BY SUM(OD2.QUANTITY)

1. List all cities that have never ordered something but we have employees there.
   1. Use sub-query

SELECT E.CITY

FROM Employees E

WHERE E.City NOT IN(

SELECT CITY

FROM Customers)

* 1. Do not use sub-query

SELECT E.City

FROM Employees E LEFT JOIN Customers C ON E.City = C.City

WHERE C.City IS NULL

1. 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 \*

FROM

(select top 1 e.City

from Employees E JOIN Orders O ON E.EmployeeID= O.EmployeeID

group by e.city

order by count(o.OrderID) desc

) AS t1

JOIN (

select top 1 e1.City

from EmployeeS e1

join Orders o1 on e1.EmployeeID = o1.EmployeeID

join [Order Details] od on od.OrderID = o1.OrderID

group by od.OrderID, o1.EmployeeID, e1.City

order by sum(od.Quantity) desc

) as t2

ON t1.City = t2.City

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

A: We could use CTE to help us achieve this goal.

1. first create CTE and select all the attributes in the table.

2. add a new attribute to sorted the number of duplicate rows and fill it in with the

value of PARTITION BY function

3. Run DELETE function on the rows which has rank greater than 1.

A picture containing graphical user interface

Description automatically generated

12. Sample table to be used for solutions below-

Employee (empid integer, mgrid integer, deptid integer, salary money)

Dept (deptid integer, deptname varchar(20))

Find employees who do not manage anybody.

A:

SELECCT empid

FROM Employee E1

WHERE empid NOT IN

(select distinct mgrid from Employee E2)

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.

A:

SELECT deptname, COUNT(E.empid) AS CountEmployees

FROM Employee E RIGHT JOIN Dept D ON E.deptid = D.deptid

GROUP BY deptname

HAVING COUNT(E.empid) = (

SELECT MAX( COUNT(E.empid) ) AS maxEmp

FROM Employee E RIGHT JOIN Dept D ON E.deptid = D.deptid

GROUP BY deptname)

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.

Employee (empid integer, mgrid integer, deptid integer, salary money)

Dept (deptid integer, deptname varchar(20))

A:

SELECT TOP 3 deptname, empid, salary

FROM Employee E LEFT JOIN Dept D ON E.deptid = D.deptid

GROUP BY deptname, empid, salary DESC