--1

I would use join rather than subquery because it will have better performance.

--2

CTE is common table expressions, which is a temporary result set that you can use within another select query.

You can use it when you want to build more modular query block and complex query.

--3

The **table variable** is a special type of the local **variable** that helps to store data temporarily, similar to the temp **table** in **SQL Server**

**the table variable scope is within the stored procedure and function.**

**--4**

**The DELETE statement removes rows one at a time and records an entry in the transaction log for each deleted row. TRUNCATE TABLE removes the data by deallocating the data pages used to store the table data and records only the page deallocations in the transaction log.**

**TRUNCATE removes all rows from a table. The operation cannot be rolled back and no triggers will be fired. As such, TRUNCATE is faster and doesn't use as much undo space as a DELETE.**

--5

**Identity column** of a table is a **column** whose value increases automatically.

Delete retains the identity and does not reset it to the seed value, truncate reset the identity to its seed value.

--6

**Delete** command is useful to **delete** all or specific rows from a table specified using a Where clause

while truncate command removes all rows of a table but can’t use a where clause.

1.

select distinct c.City from Customers c

join Employees e on e.City = c.City

2.

1. use subquery

select distinct c.City from Customers c where c.City not in (

select e.City from Employees e

)

1. do not use subquery

select c.City from Customers c

left join Employees e on c.City = e.City where e.City is null

3.

select o.ProductID, p.ProductName, sum(o.Quantity) as "TotalQuantity" from [Order Details] o

join Products p on p.ProductID = o.ProductID

group by o.ProductID, p.ProductName order by ProductID

4.

select o.ShipCity, count(o.OrderID) as "TotalOrders" from Orders o

GROUP BY o.ShipCity

5.

use union

SELECT c.City from Customers c

group by c.City

having count(c.City) >= 2

union

SELECT o.ShipCity from Orders o

group by o.ShipCity

having count(o.ShipCity) >= 2

use subquery

select c.City from Customers c

where c.City in (

select o.ShipCity from Orders o

GROUP BY o.ShipCity

HAVING count(o.ShipCity) >=2

)

6.

select c.City from Customers c

join Orders o on c.CustomerID = o.CustomerID

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

GROUP BY c.City

having count(od.ProductID) >= 2

7.

Select distinct o.CustomerId from Orders o

join Customers c on c.CustomerID = o.CustomerID

where c.City != o.ShipCity

8.

9.

subquery

select e.City, e.EmployeeID from Employees e where

e.City not in

(Select distinct ShipCity from Orders)

and e.EmployeeID is not null

no subquery

select e.City,e.EmployeeID

from Employees e left join Orders o

on e.EmployeeID = o.EmployeeID

where o.OrderID is null

10.

select top 1 o.EmployeeID, count(o.OrderID) as "TotalOrder", e.City

from Orders o join Employees e

on o.EmployeeID = e.EmployeeID

GROUP BY o.EmployeeID, e.City

select top 1 o.ShipCity, dense\_rank() over(order by sum(Quantity) desc) rnk from [Order Details] od

join Orders o on o.OrderID = od.OrderID

GROUP by o.ShipCity

11.

Use delete statement to remove the duplicate rows

find duplicate rows use group by

12.

select empid from Employee where empid not in

(select mgrid from Employee)

13.

select d.deptname, count(e.empid) from Dept d join Employee e

on e.deptid = d.deptid

order by d.deptname

14.

select top 3 d.deptname, e.empid,dense\_rank() over(order by e.salary desc) rnk from Employee e

join Dept d on d.deptid = e.deptid order by d.deptname