```
--This query filters orders that were placed by customer(custid) 71, groups those orders
--by employee(empid) and order year, and filters only groups of employees and years that
--have more than one order. For the remaining groups, the query presents the employee ID,
-- order year, and count of orders, sorted by the employee ID and order year.
-- Sales.Orders Table
-----------------------------------------------------------------------------------------

SELECT empid, YEAR(orderdate) AS orderyear, COUNT(*) AS numoforders
FROM Sales.Orders
WHERE custid = 71
GROUP BY empid, YEAR(orderdate)
HAVING COUNT(*) >1
ORDER BY empid, orderyear;

-----------------------------------------------------------------------------------------
--The following query returns the total freight and number of orders per employee
--and order year orders and those placed by customer(custid) 71.
--groups those orders by employee(empid) and order year.
-- sorted by the employee ID and order year. Sales.Orders table

SELECT empid, YEAR(orderdate) as orderyear, SUM(freight) AS TotalFreight,
COUNT(*) as numoforders
FROM Sales.Orders
WHERE custid = 71
GROUP BY empid, YEAR(orderdate)
ORDER BY empid, YEAR(orderdate);

-----------------------------------------------------------------------------------------
--The query returns orders that were either “placed by customer 1 and handled by
--employees 1, 3, or 5” or “placed by customer 85 and handled by employees 2, 4, or 6.”
--orderid, custid, empi, orderdate columns. Sales.Orders Table

SELECT orderid, custid, empid, orderdate
FROM Sales.Orders
WHERE custid = 1 AND empid IN(1,3,5)
OR custid = 85 and empid IN(2,4,6);

-----------------------------------------------------------------------------------------
--String concatenation (plus-sign [+] operator and CONCAT function)
For example, the following query against the Employees table produces the fullname result
column by concatenating  firstname, a space, and lastname:

SELECT empid, firstname + '  ' + lastname AS fullname
FROM HR.Employees;

-----------------------------------------------------------------------------------------
--Write a query against the Sales.Orders table that returns orders placed in
--Between 2015-01-01 and 2016-01-01.
-- Tables involved: TSQLV4 database and the Sales.Orders table

SELECT orderid,orderdate,custid,empid
FROM Sales.Orders
WHERE orderdate BETWEEN '20150101' AND  '20160101'


-----------------------------------------------------------------------------------------
--Write a query against the Sales.Orders table that returns orders placed on the
--last day of the month:Tables involved: TSQLV4 database and the Sales.Orders table
```

```sql
SELECT orderid, orderdate, custid, empid
FROM Sales.Orders
WHERE orderdate = EOMONTH(orderdate);

-----------------------------------------------------------------------------------
--Write a query against the Sales.OrderDetails table that returns orders with a
--total value(quantity * unitprice) greater than 10,000, sorted by total value:
--Tables involved: TSQLV4 database and the Sales.OrderDetails table

SELECT orderid, SUM(qty*unitprice) AS totalvalue
FROM Sales.OrderDetails
GROUP BY orderid
HAVING SUM(qty*unitprice) > 10000
ORDER BY totalvalue DESC;

-----------------------------------------------------------------------------------
--The OFFSET-FETCH filter : goes with ORDER BY
--OFFSET(50) skip the first 50 rows -FETCH(25) fetch the next 25 rows
--Sales.Orders Table order by orderdate

SELECT orderid, orderdate, custid, empid
FROM Sales.Orders
ORDER BY orderdate, orderid
OFFSET 50 ROWS FETCH NEXT 25 ROWS ONLY

-----------------------------------------------------------------------------------
--Write a query against the Sales.Orders table that returns the three shipped-to
-- countries with the highest average freight in 2015:
--Tables involved: TSQLV4 database and the Sales.Orders table

SELECT shipcountry, AVG(freight) AS avgfreight
FROM Sales.Orders
WHERE orderdate BETWEEN '20150101' AND '20160101'
GROUP BY shipcountry
ORDER By avgfreight DESC
OFFSET 0 ROWS FETCH NEXT 3 ROWS ONLY

-----------------------------------------------------------------------------------
--T-SQL provides a predicate called LIKE that you can use to check whether a character
--string matches a specified pattern.--The percent sign represents a string of any size,
-----------------------------------------------------------------------------------
-- Return last name that starts with D from HR.Employees Table.
-- Returns employees where the second character in the last name is e:
-- Returns characters in range A-E

SELECT empid, firstname, lastname
FROM HR.Employees
WHERE lastname LIKE 'D%'

SELECT empid, firstname, lastname
FROM HR.Employees
WHERE lastname LIKE '_e%';

SELECT empid, firstname, lastname
FROM HR.Employees
WHERE lastname LIKE '[A-E]%'
```