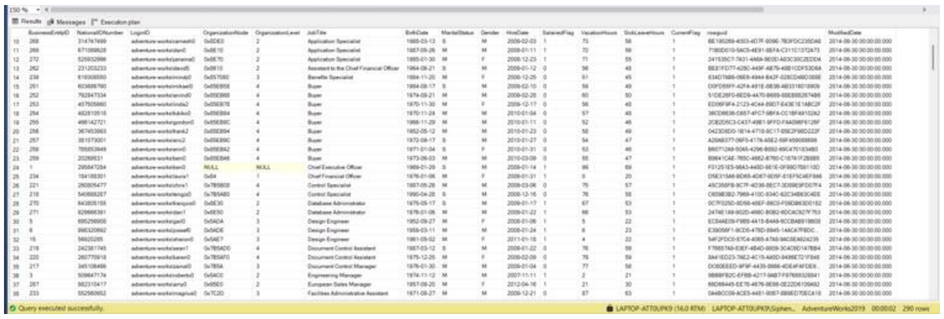
Assessment-1

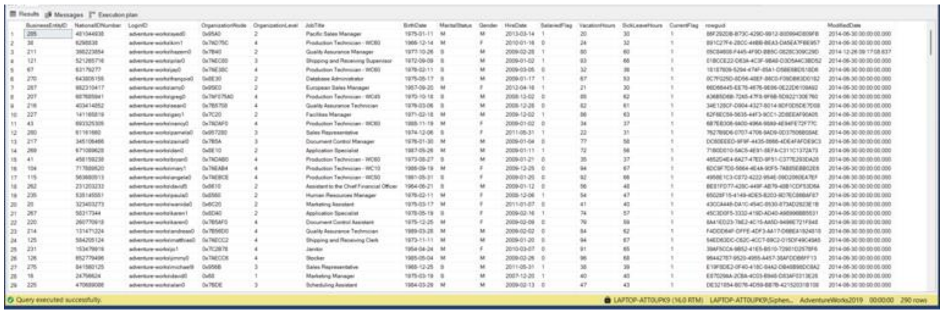
1. Write a query in SQL to retrieve all rows and columns from the employee table in the Adventureworks database. Sort the result set in ascending order on jobtitle.

```
SELECT *

FROM [HumanResources].[Employee]

ORDER BY jobtitle ASC;
```



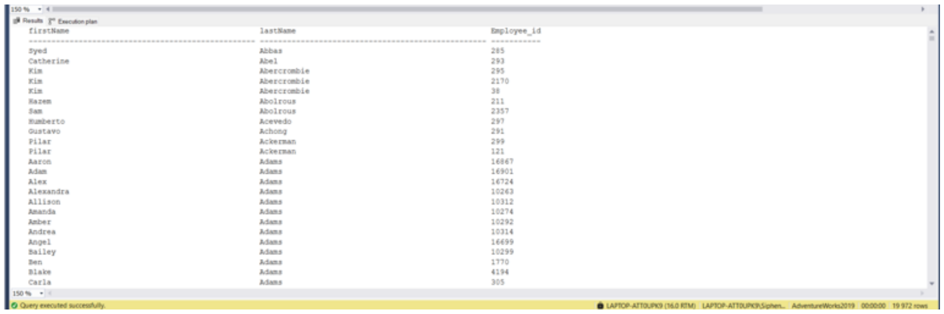
2. Write a query in SQL to retrieve all rows and columns from the employee table using table aliasing in the Adventureworks database. Sort the output in ascending order on lastname.

```
SELECT e.*

FROM HumanResources.Employee AS e

JOIN Person.Person AS p ON e.BusinessEntityID = p.BusinessEntityID

ORDER BY p.LastName ASC;
```



3. Write a query in SQL to return all rows and a subset of the columns (FirstName, LastName, businessentityid) from the person table in the AdventureWorks database. The third column heading is renamed to Employee_id. Arranged the output in ascending order by lastname.

```sql
SELECT firstName, lastName, businessentityid AS Employee_id

FROM [Person].[Person]

Order By lastName ASC;
```



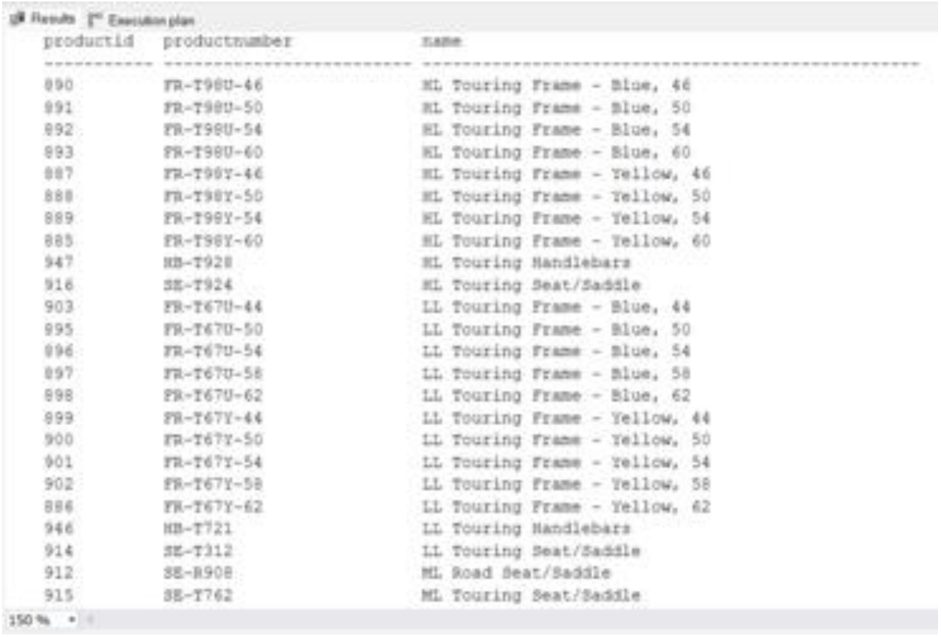
```
firstName                                      lastName                                       Employee_id
---------------------------------------------- ---------------------------------------------- -----------
Syed                                           Abbas                                          285
Catherine                                      Abel                                           293
Kim                                            Abercrombie                                    295
Kim                                            Abercrombie                                    2170
Kim                                            Abercrombie                                    38
Hazem                                          Abolrous                                       211
Sam                                            Abolrous                                       2357
Humberto                                       Acevedo                                        297
Gustavo                                        Achong                                         291
Pilar                                          Ackerman                                       299
Pilar                                          Ackerman                                       121
Aaron                                          Adams                                          16867
Adam                                           Adams                                          16901
Alex                                           Adams                                          16724
Alexandra                                      Adams                                          10263
Allison                                        Adams                                          10312
Amanda                                         Adams                                          10274
Amber                                          Adams                                          10292
Andrea                                         Adams                                          10314
Angel                                          Adams                                          16699
Bailey                                         Adams                                          10299
Ben                                            Adams                                          1770
Blake                                          Adams                                          4194
Carla                                          Adams                                          305
```

Query executed successfully.

LAPTOP-ATT0UPK9 (16.0 RTM) LAPTOP-ATT0UPK9\Siphen... AdventureWorks2019 00:00:00 19 972 rows

4. Write a query in SQL to return only the rows for product that have a sellstartdate that is not NULL and a productline of 'T'. Return productid, productnumber, and name. Arranged the output in ascending order on name.

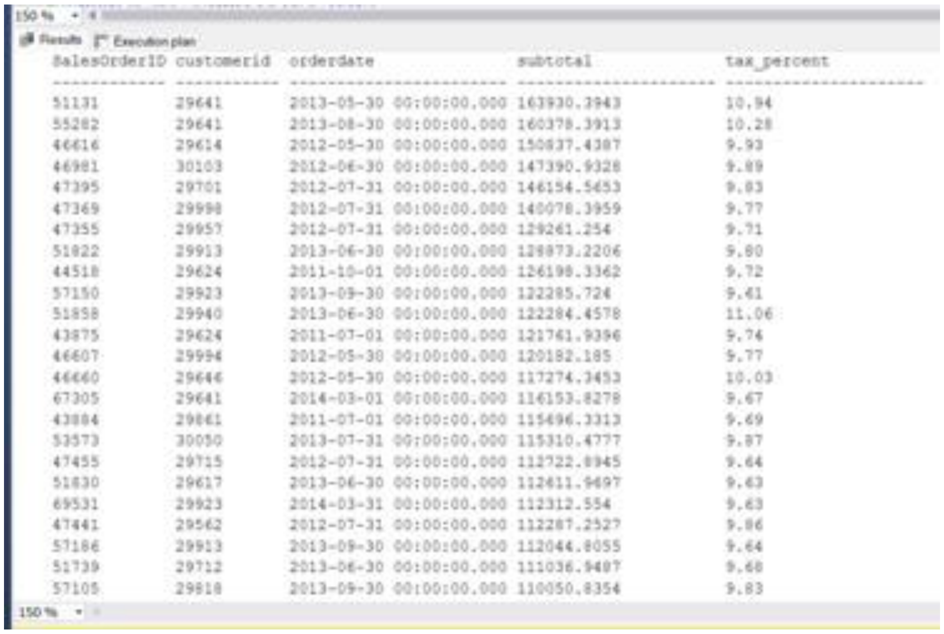
```sql
SELECT  productid, productnumber, name

FROM [Production].[Product]

WHERE SellStartDate IS NOT NULL AND ProductLine = 'T'

ORDER BY name ASC;
```



```
productid   productnumber             name
----------- ------------------------- --------------------------------------------------
890         FR-T98U-46                HL Touring Frame - Blue, 46
891         FR-T98U-50                HL Touring Frame - Blue, 50
892         FR-T98U-54                HL Touring Frame - Blue, 54
893         FR-T98U-60                HL Touring Frame - Blue, 60
887         FR-T98Y-46                HL Touring Frame - Yellow, 46
888         FR-T98Y-50                HL Touring Frame - Yellow, 50
889         FR-T98Y-54                HL Touring Frame - Yellow, 54
885         FR-T98Y-60                HL Touring Frame - Yellow, 60
947         HB-T928                   HL Touring Handlebars
916         SE-T924                   HL Touring Seat/Saddle
903         FR-T67U-44                LL Touring Frame - Blue, 44
895         FR-T67U-50                LL Touring Frame - Blue, 50
896         FR-T67U-54                LL Touring Frame - Blue, 54
897         FR-T67U-58                LL Touring Frame - Blue, 58
898         FR-T67U-62                LL Touring Frame - Blue, 62
899         FR-T67Y-44                LL Touring Frame - Yellow, 44
900         FR-T67Y-50                LL Touring Frame - Yellow, 50
901         FR-T67Y-54                LL Touring Frame - Yellow, 54
902         FR-T67Y-58                LL Touring Frame - Yellow, 58
886         FR-T67Y-62                LL Touring Frame - Yellow, 62
946         HB-T721                   LL Touring Handlebars
914         SE-T312                   LL Touring Seat/Saddle
912         SE-R908                   ML Road Seat/Saddle
915         SE-T762                   ML Touring Seat/Saddle
```

5. Write a query in SQL to return all rows from the salesorderheader table in Adventureworks database and calculate the percentage of tax on the subtotal have decided. Return salesorderid, customerid, orderdate, subtotal, percentage of tax column. Arranged the result set in ascending order on subtotal.

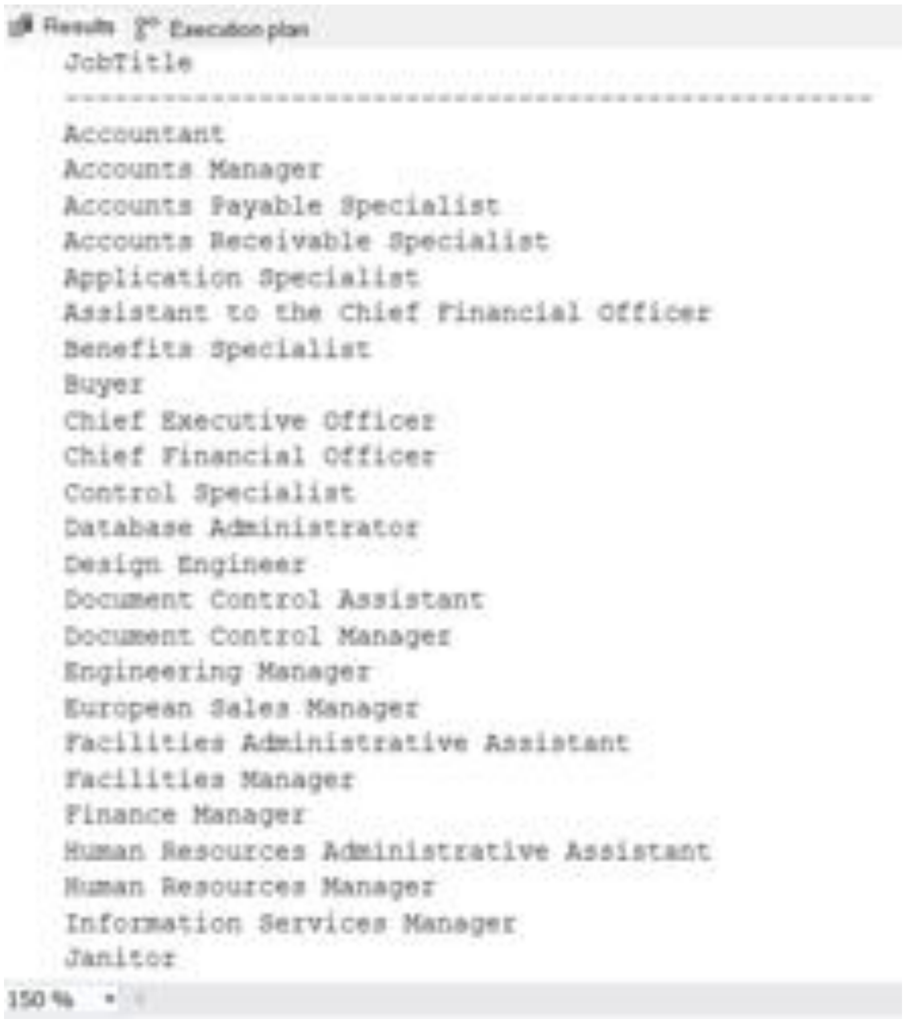
```
SELECT  SalesOrderID, customerid, orderdate, subtotal, (TaxAmt / SubTotal) *
100 AS tax_percent

FROM [Sales].[SalesOrderHeader]ORDER BY subtotal DESC
```



```
SalesOrderID customerid  orderdate               subtotal              tax_percent
------------ ----------- ----------------------- --------------------- ---------------------
51131        29641       2013-05-30 00:00:00.000 163930.3943           10.94
55282        29641       2013-08-30 00:00:00.000 160378.3913           10.28
46616        29614       2012-05-30 00:00:00.000 150837.4387           9.93
46981        30103       2012-06-30 00:00:00.000 147390.9328           9.89
47395        29701       2012-07-31 00:00:00.000 146154.5653           9.83
47369        29998       2012-07-31 00:00:00.000 140078.3959           9.77
47355        29957       2012-07-31 00:00:00.000 129261.254            9.71
51822        29913       2013-06-30 00:00:00.000 128873.2206           9.80
44518        29624       2011-10-01 00:00:00.000 126198.3362           9.72
57150        29923       2013-09-30 00:00:00.000 122285.724            9.61
51858        29940       2013-06-30 00:00:00.000 122284.4578           11.06
43875        29624       2011-07-01 00:00:00.000 121761.9396           9.74
46607        29994       2012-05-30 00:00:00.000 120182.185            9.77
46660        29646       2012-05-30 00:00:00.000 117274.3453           10.03
67305        29641       2014-03-01 00:00:00.000 116153.8278           9.67
43884        29861       2011-07-01 00:00:00.000 115696.3313           9.69
53573        30050       2013-07-31 00:00:00.000 115310.4777           9.87
47455        29715       2012-07-31 00:00:00.000 112722.8945           9.64
51830        29617       2013-06-30 00:00:00.000 112611.9697           9.63
69531        29923       2014-03-31 00:00:00.000 112312.554            9.63
47441        29562       2012-07-31 00:00:00.000 112287.2527           9.86
57186        29913       2013-09-30 00:00:00.000 112044.8055           9.64
51739        29712       2013-06-30 00:00:00.000 111036.9487           9.68
57105        29818       2013-09-30 00:00:00.000 110050.8354           9.83
```

6. Write a query in SQL to create a list of unique jobtitles in the employee table in Adventureworks database. Return jobtitle column and arranged the resultset in ascending order.

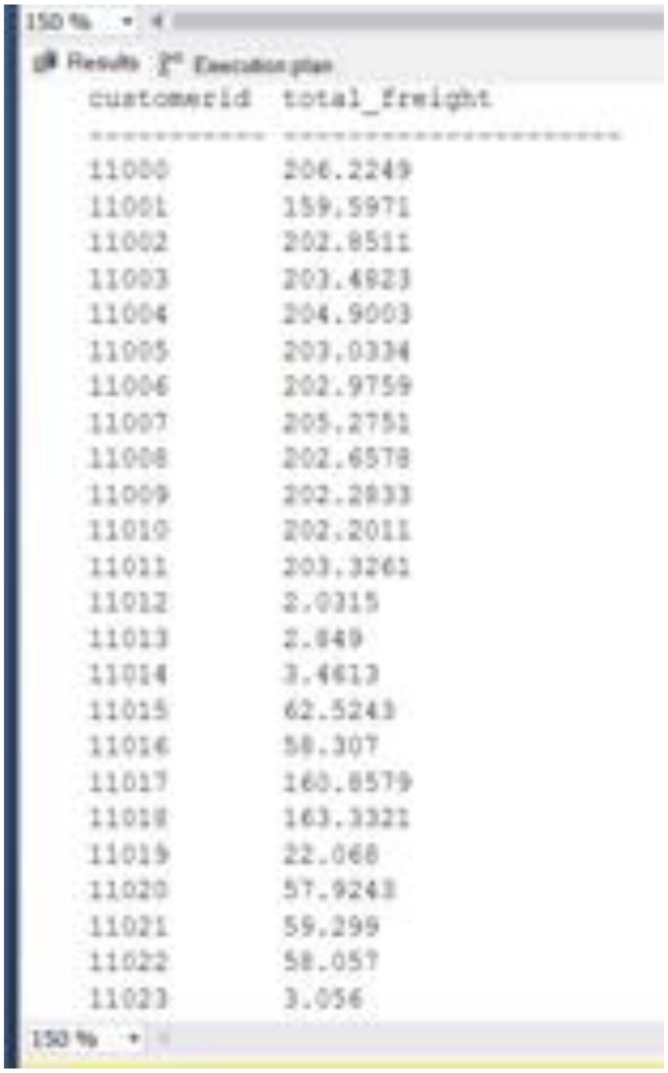
```
SELECT Distinct JobTitle

FROM [HumanResources].[Employee]

ORDER BY [JobTitle] ASC;
```



```
JobTitle
--------------------------------------------------
Accountant
Accounts Manager
Accounts Payable Specialist
Accounts Receivable Specialist
Application Specialist
Assistant to the Chief Financial Officer
Benefits Specialist
Buyer
Chief Executive Officer
Chief Financial Officer
Control Specialist
Database Administrator
Design Engineer
Document Control Assistant
Document Control Manager
Engineering Manager
European Sales Manager
Facilities Administrative Assistant
Facilities Manager
Finance Manager
Human Resources Administrative Assistant
Human Resources Manager
Information Services Manager
Janitor
```

7. Write a query in SQL to calculate the total freight paid by each customer. Return customerid and total freight. Sort the output in ascending order on customerid.

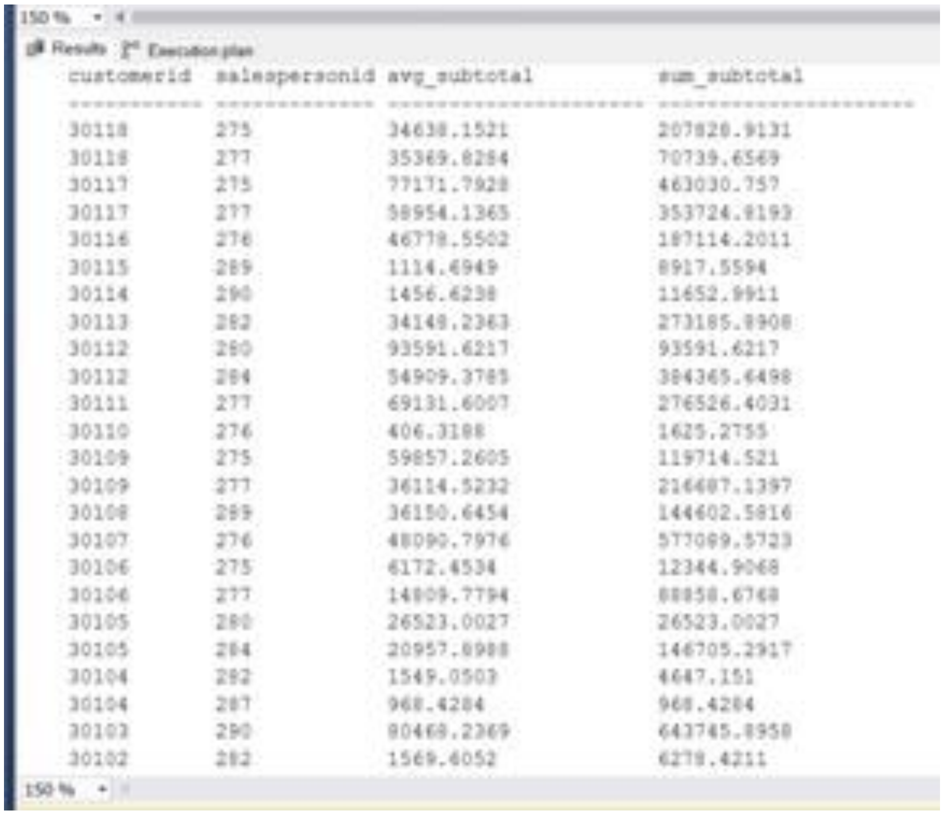
```
SELECT  customerid, SUM(Freight) AS total_freight

FROM [Sales].[SalesOrderHeader]

GROUP BY customerid

ORDER BY customerid ASC;
```



```
customerid  total_freight
----------- ---------------------
11000       206.2249
11001       159.5971
11002       202.8511
11003       203.4823
11004       204.9003
11005       203.0334
11006       202.9759
11007       205.2751
11008       202.6578
11009       202.2833
11010       202.2011
11011       203.3261
11012       2.0315
11013       2.849
11014       3.4613
11015       62.5243
11016       58.307
11017       160.8579
11018       163.3321
11019       22.068
11020       57.9243
11021       59.299
11022       58.057
11023       3.056
```

8. Write a query in SQL to find the average and the sum of the subtotal for every customer. Return customerid, average and sum of the subtotal. Grouped the result on customerid and salespersonid. Sort the result on customerid column in descending order.

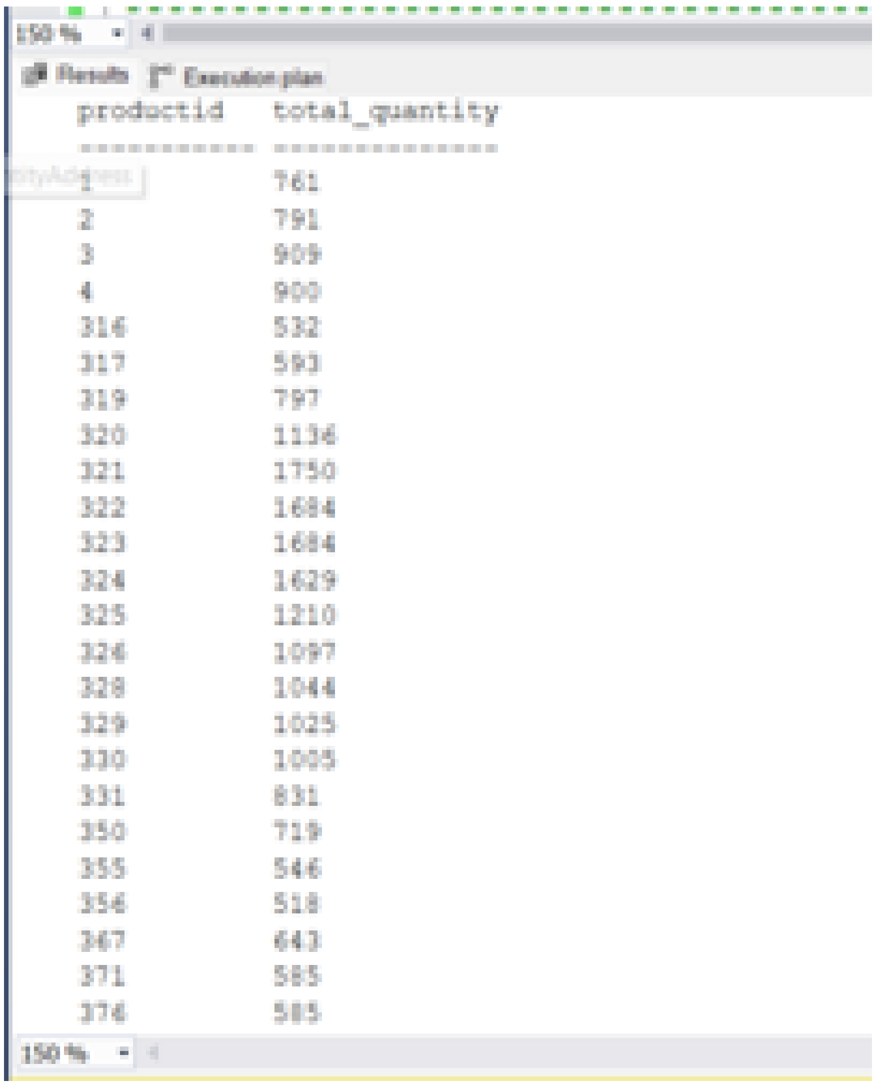
```
SELECT  customerid, salespersonid, AVG(SubTotal) AS avg_subtotal,
SUM(SubTotal) as sum_subtotal

FROM [Sales].[SalesOrderHeader]

GROUP By customerid, salespersonid

ORDER BY customerid DESC;
```



```
customerid  salespersonid avg_subtotal          sum_subtotal
----------- ------------- --------------------- ---------------------
30118       275           34638.1521            207828.9131
30118       277           35369.8284            70739.6569
30117       275           77171.7928            463030.757
30117       277           58954.1365            353724.8193
30116       276           46778.5502            187114.2011
30115       289           1114.6949             8917.5594
30114       290           1456.6238             11652.9911
30113       282           34148.2363            273185.8908
30112       280           93591.6217            93591.6217
30112       284           54909.3785            384365.6498
30111       277           69131.6007            276526.4031
30110       276           406.3188              1625.2755
30109       275           59857.2605            119714.521
30109       277           36114.5232            216687.1397
30108       289           36150.6454            144602.5816
30107       276           48090.7976            577089.5723
30106       275           6172.4534             12344.9068
30106       277           14809.7794            88858.6768
30105       280           26523.0027            26523.0027
30105       284           20957.8988            146705.2917
30104       282           1549.0503             4647.151
30104       287           968.4284              968.4284
30103       290           80468.2369            643745.8958
30102       282           1569.6052             6278.4211
```

9. Write a query in SQL to retrieve total quantity of each productid which are in shelf of 'A' or 'C' or 'H'. Filter the results for sum quantity is more than 500. Return productid and sum of the quantity. Sort the results according to the productid in ascending order.

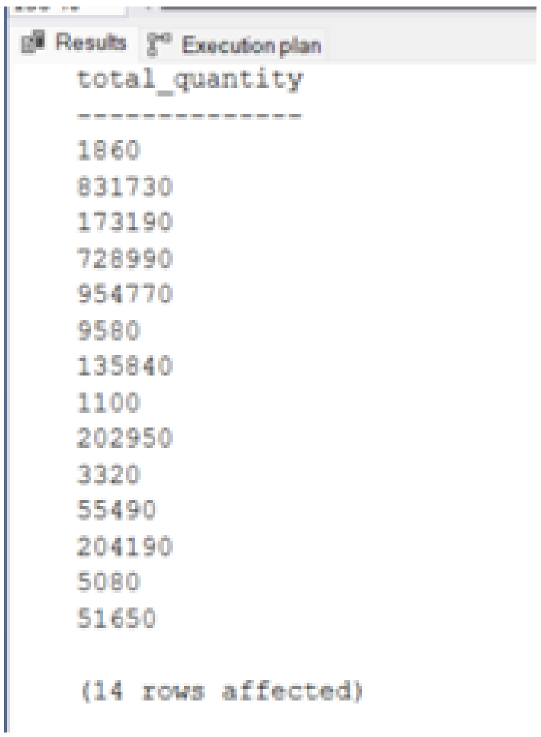
```
SELECT productid, SUM(quantity) AS total_quantity

FROM [Production].[ProductInventory]

WHERE shelf = 'A' OR shelf = 'C' OR shelf = 'H'

GROUP BY productid

HAVING SUM(quantity) > 500

ORDER BY productid ASC;
```



```
productid   total_quantity
----------- --------------
1           761
2           791
3           909
4           900
316         532
317         593
319         797
320         1136
321         1750
322         1684
323         1684
324         1629
325         1210
326         1097
328         1044
329         1025
330         1005
331         831
350         719
355         546
356         518
367         643
371         585
376         585
```

10. Write a query in SQL to find the total quentity for a group of locationid multiplied by 10

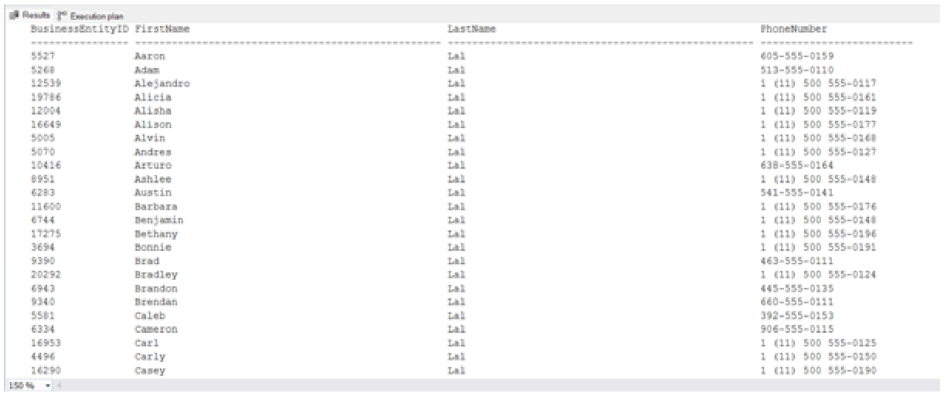
```
SELECT SUM(quantity) * 10 AS total_quantity

FROM [Production].[ProductInventory]

GROUP BY locationID
```



```
total_quantity
--------------
1860
831730
173190
728990
954770
9580
135840
1100
202950
3320
55490
204190
5080
51650

(14 rows affected)
```

11. Write a query in SQL to find the persons whose last name starts with letter 'L'. Return BusinessEntityID, FirstName, LastName, and PhoneNumber. Sort the result on lastname and firstname.

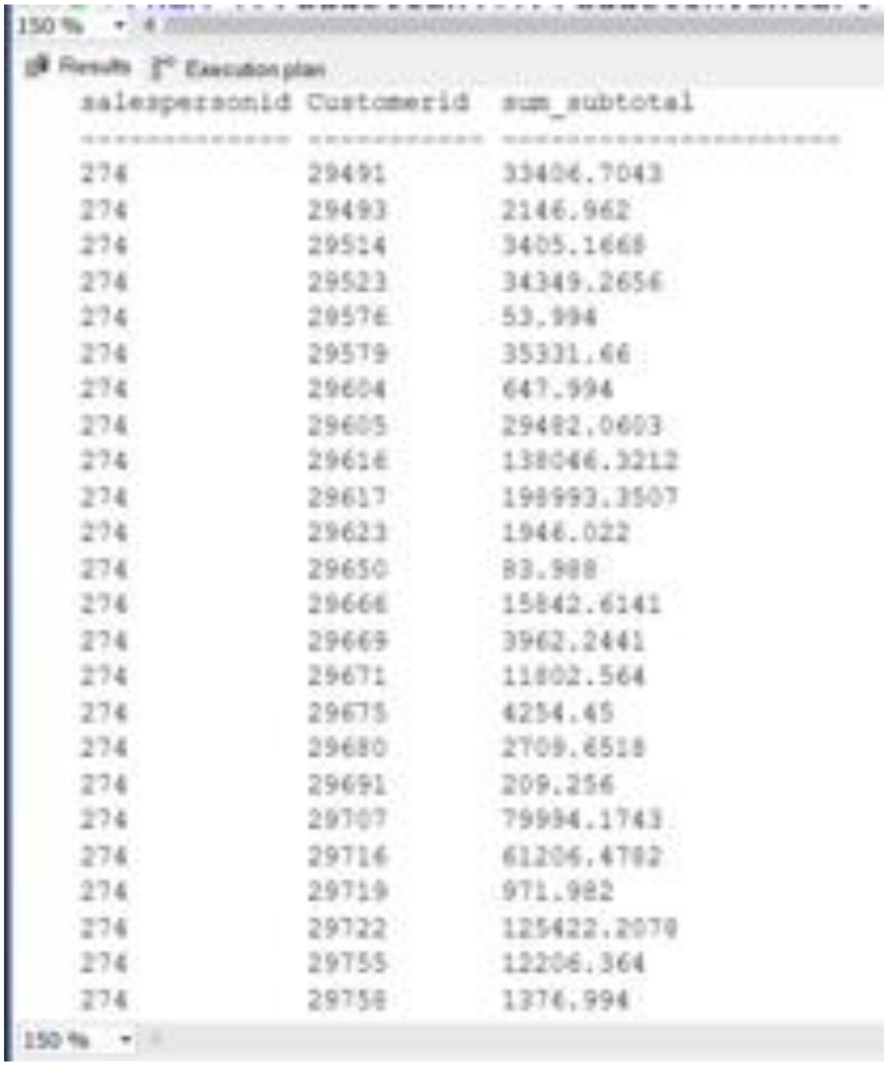
```
SELECT Person.BusinessEntityID, Person.FirstName, Person.LastName,
PersonPhone.PhoneNumber
FROM [Person].[Person]
JOIN [Person].[PersonPhone] ON Person.BusinessEntityID =
PersonPhone.BusinessEntityID
WHERE Person.LastName LIKE 'L%'
ORDER BY Person.LastName, Person.FirstName;
```



```
BusinessEntityID FirstName                                          LastName                                           PhoneNumber
---------------- -------------------------------------------------- -------------------------------------------------- -------------------------
5527             Aaron                                              Lal                                                605-555-0159
5268             Adam                                               Lal                                                513-555-0110
12539            Alejandro                                          Lal                                                1 (11) 500 555-0117
19786            Alicia                                             Lal                                                1 (11) 500 555-0161
12004            Alisha                                             Lal                                                1 (11) 500 555-0119
16649            Alison                                             Lal                                                1 (11) 500 555-0177
5005             Alvin                                              Lal                                                1 (11) 500 555-0168
5070             Andres                                             Lal                                                1 (11) 500 555-0127
10416            Arturo                                             Lal                                                638-555-0164
8951             Ashlee                                             Lal                                                1 (11) 500 555-0148
6283             Austin                                             Lal                                                541-555-0141
11600            Barbara                                            Lal                                                1 (11) 500 555-0176
6744             Benjamin                                           Lal                                                1 (11) 500 555-0148
17275            Bethany                                            Lal                                                1 (11) 500 555-0196
3694             Bonnie                                             Lal                                                1 (11) 500 555-0191
9390             Brad                                               Lal                                                463-555-0111
20292            Bradley                                            Lal                                                1 (11) 500 555-0124
6943             Brandon                                            Lal                                                445-555-0135
9340             Brendan                                            Lal                                                660-555-0111
5581             Caleb                                              Lal                                                392-555-0153
6334             Cameron                                            Lal                                                906-555-0115
16953            Carl                                               Lal                                                1 (11) 500 555-0125
4496             Carly                                              Lal                                                1 (11) 500 555-0150
16290            Casey                                              Lal                                                1 (11) 500 555-0190
```

12. Write a query in SQL to find the sum of subtotal column. Group the sum on distinct salespersonid and customerid. Rolls up the results into subtotal and running total. Return salespersonid, customerid and sum of subtotal column i.e. sum_subtotal.

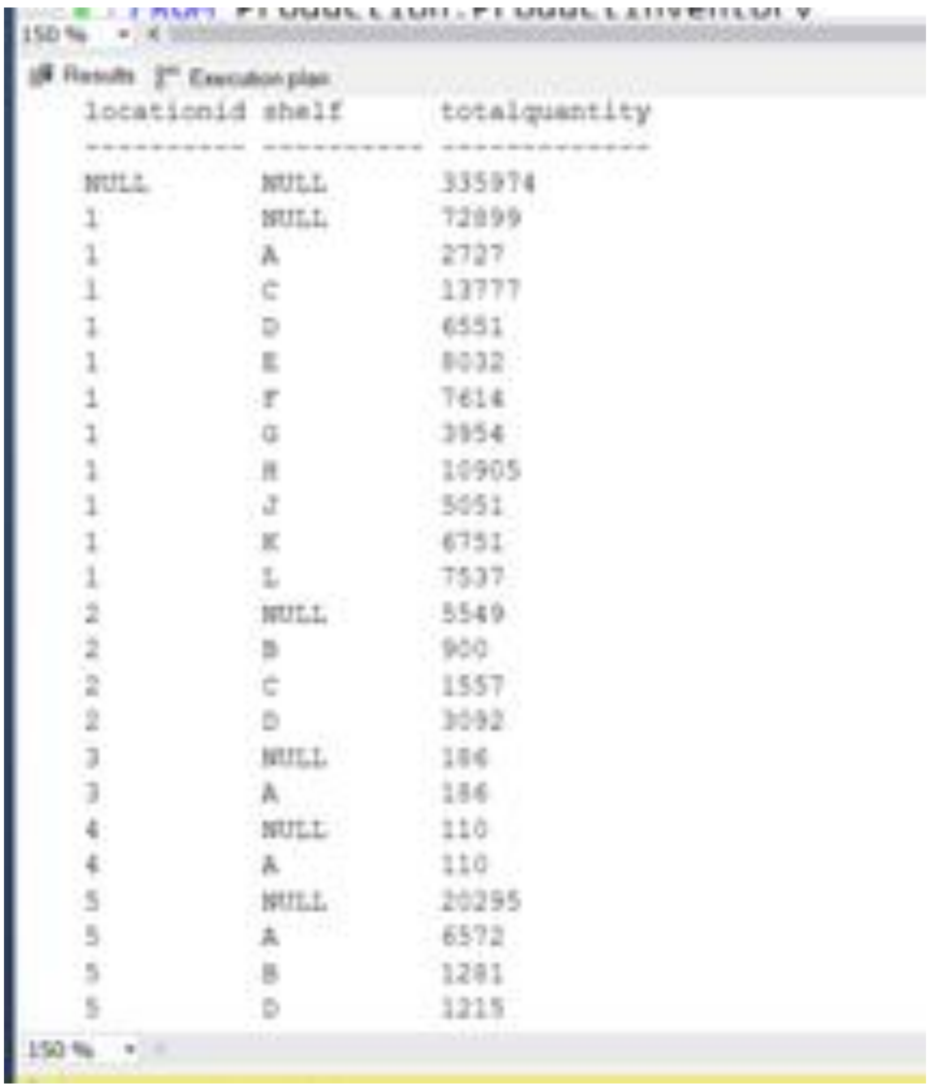
```
SELECT salespersonid, Customerid, sum(subtotal) AS sum_subtotal
FROM [Sales].[SalesOrderHeader]
WHERE salespersonid IS NOT NULL
GROUP BY ROLLUP (salespersonid, Customerid);
```



```
salespersonid Customerid  sum_subtotal
------------- ----------  ---------------------
274           29491       33406.7043
274           29493       2146.962
274           29514       3405.1668
274           29523       34349.2656
274           29576       53.994
274           29579       35331.66
274           29604       647.994
274           29605       29482.0603
274           29616       138046.3212
274           29617       198993.3507
274           29623       1946.022
274           29650       83.988
274           29666       15842.6141
274           29669       3962.2441
274           29671       11802.564
274           29675       4254.45
274           29680       2709.6518
274           29691       209.256
274           29707       79994.1743
274           29716       61206.4782
274           29719       971.982
274           29722       125422.2078
274           29755       12206.364
274           29758       1376.994
```

13. Write a query in SQL to find the sum of the quantity of all combination of group of distinct locationid and shelf column. Return locationid, shelf and sum of quantity as TotalQuantity

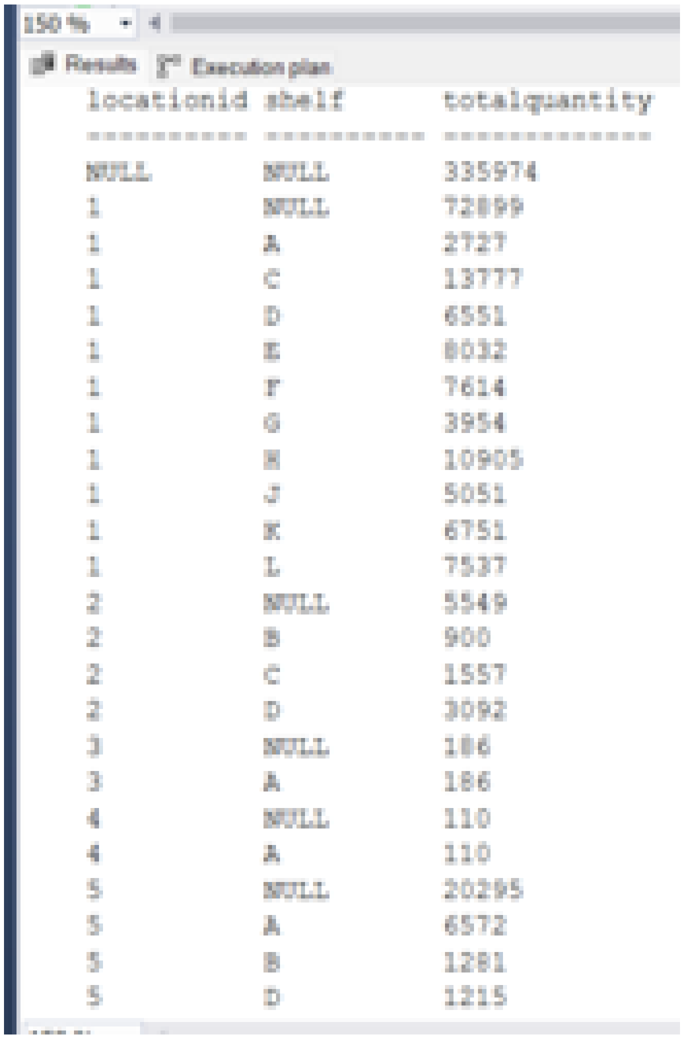
```
 SELECT locationid, shelf, sum(quantity) AS totalquantity

FROM [Production].[ProductInventory]

GROUP BY ROLLUP (locationid, shelf)

ORDER BY locationid, shelf
```



```
locationid shelf      totalquantity
---------- ---------- -------------
NULL       NULL       335974
1          NULL       72899
1          A          2727
1          C          13777
1          D          6551
1          E          8032
1          F          7614
1          G          3954
1          H          10905
1          J          5051
1          K          6751
1          L          7537
2          NULL       5549
2          B          900
2          C          1557
2          D          3092
3          NULL       186
3          A          186
4          NULL       110
4          A          110
5          NULL       20295
5          A          6572
5          B          1281
5          D          1215
```

14. Write a query in SQL to find the sum of the quantity with subtotal for each locationid. Group the results for all combination of distinct locationid and shelf column. Rolls up the results into subtotal and running total. Return locationid, shelf and sum of quantity as TotalQuantity.

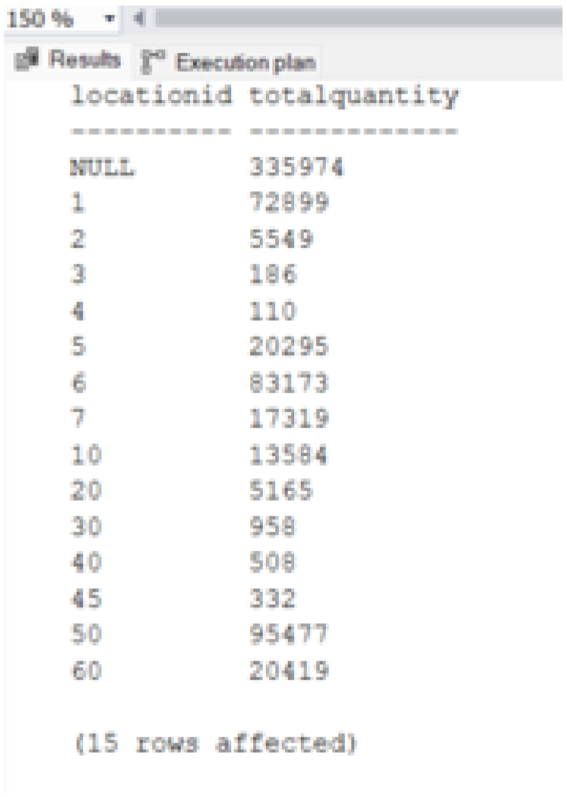
```
SELECT locationid, shelf, SUM(quantity) AS totalquantity
FROM Production.ProductInventory
GROUP BY ROLLUP (locationid,shelf)
ORDER BY locationid,shelf;
```



```
locationid shelf      totalquantity
---------- ---------- -------------
NULL       NULL       335974
1          NULL       72899
1          A          2727
1          C          13777
1          D          6551
1          E          8032
1          F          7614
1          G          3954
1          H          10905
1          J          5051
1          K          6751
1          L          7537
2          NULL       5549
2          B          900
2          C          1557
2          D          3092
3          NULL       186
3          A          186
4          NULL       110
4          A          110
5          NULL       20295
5          A          6572
5          B          1281
5          D          1215
```

15. Write a query in SQL to find the total quantity for each locationid and calculate the grand total for all locations. Return locationid and total quantity. Group the results on locationid

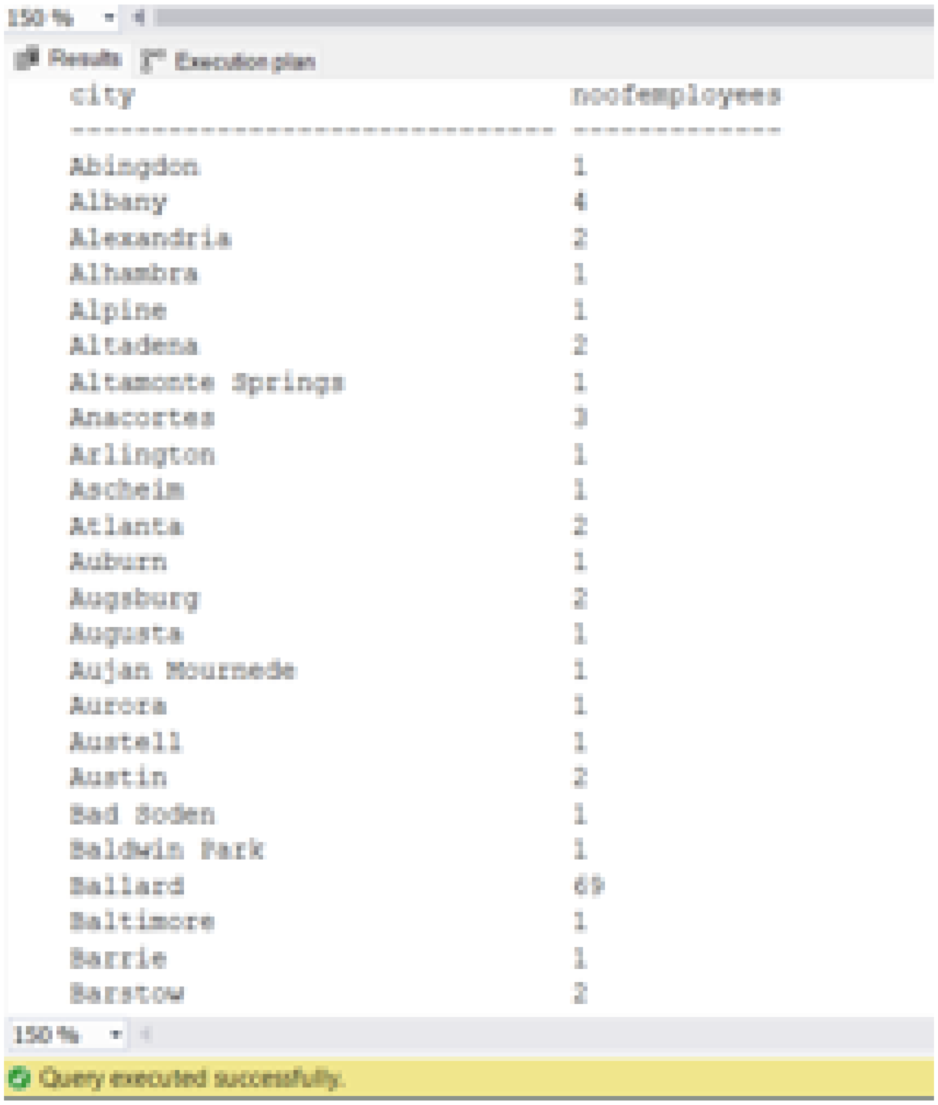
```
SELECT locationid, SUM(quantity) AS totalquantity
FROM [Production].[ProductInventory]
GROUP BY ROLLUP (locationid)
ORDER BY locationid;
```



```
locationid totalquantity
---------- -------------
NULL       335974
1          72899
2          5549
3          186
4          110
5          20295
6          83173
7          17319
10         13584
20         5165
30         958
40         508
45         332
50         95477
60         20419

(15 rows affected)
```

16. Write a query in SQL to retrieve the number of employees for each City. Return city and number of employees. Sort the result in ascending order on city.

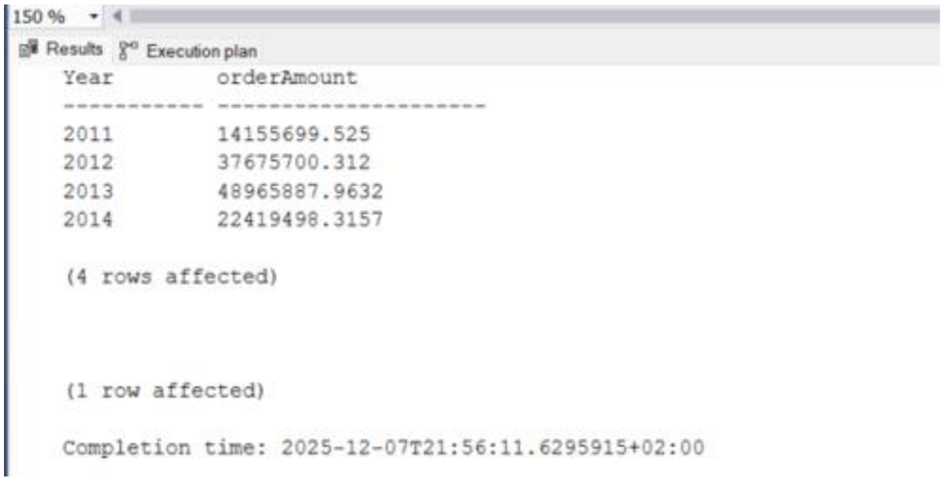
```
SELECT city,COUNT(*) AS noofemployees
FROM [Person].[Address]
GROUP BY city
ORDER BY city Asc
```



```
city                           noofemployees
------------------------------ -------------
Abingdon                       1
Albany                         4
Alexandria                     2
Alhambra                       1
Alpine                         1
Altadena                       2
Altamonte Springs              1
Anacortes                      3
Arlington                      1
Ascheim                        1
Atlanta                        2
Auburn                         1
Augsburg                       2
Augusta                        1
Aujan Mournede                 1
Aurora                         1
Austell                        1
Austin                         2
Bad Soden                      1
Baldwin Park                   1
Ballard                        69
Baltimore                      1
Barrie                         1
Barstow                        2
```

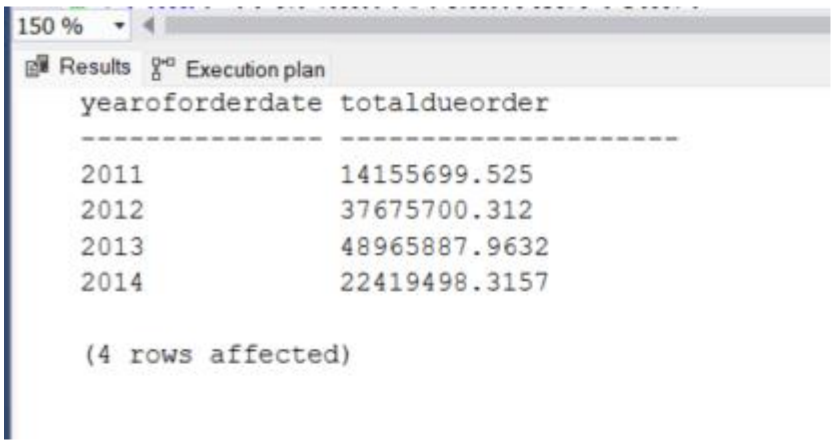
17. Write a query in SQL to retrieve the total sales for each year. Return the year part of order date and total due amount. Sort the result in ascending order on year part of order date

```
SELECT YEAR(orderdate) AS Year, sum(totaldue) AS orderAmount
FROM [Sales].[SalesOrderHeader]
GROUP BY YEAR(orderdate)
ORDER BY YEAR(orderdate);
```


```
Year        orderAmount
----------- ---------------------
2011        14155699.525
2012        37675700.312
2013        48965887.9632
2014        22419498.3157

(4 rows affected)


(1 row affected)

Completion time: 2025-12-07T21:56:11.6295915+02:00
```

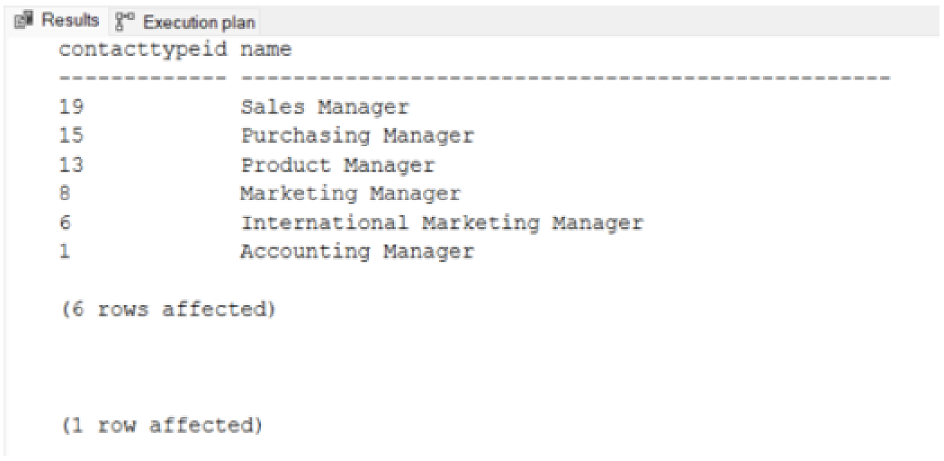
18. Write a query in SQL to retrieve the total sales for each year. Filter the result set for those orders where order year is on or before 2016. Return the year part of orderdate and total due amount. Sort the result in ascending order on year part of order date

```
SELECT YEAR(orderdate) AS yearoforderdate, SUM(totaldue) AS totaldueorder
FROM [Sales].[SalesOrderHeader]
WHERE YEAR(orderdate) <= 2016
GROUP BY YEAR(orderdate)
ORDER BY YEAR(orderdate)
```


```
yearoforderdate totaldueorder
--------------- ---------------------
2011            14155699.525
2012            37675700.312
2013            48965887.9632
2014            22419498.3157

(4 rows affected)
```

19. Write a query in SQL to find the contacts who are designated as a manager in various departments. Returns ContactTypeID, name. Sort the result set in descending order

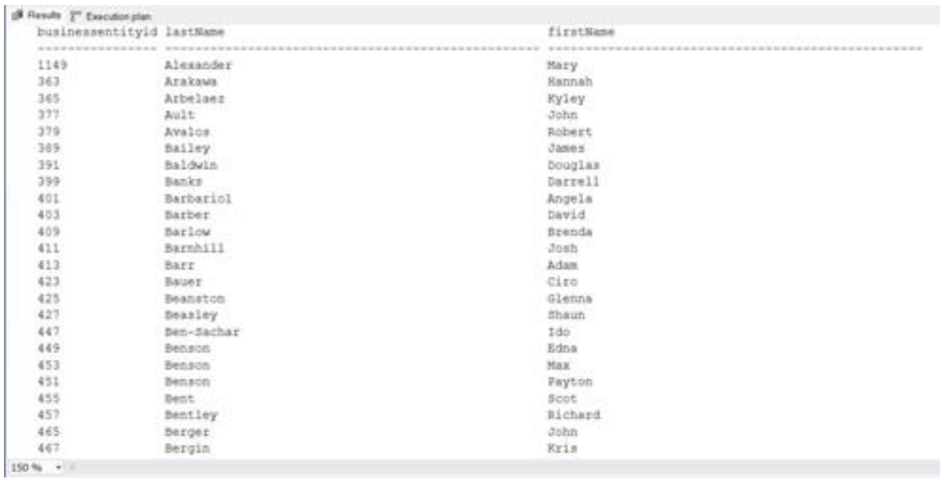
```
SELECT contacttypeid, name
FROM [Person].[ContactType]
WHERE name LIKE '%Manager%'
--GROUP BY contacttypeid
ORDER BY contacttypeid DESC;
```



```
contacttypeid name
------------- --------------------------------------------------
19            Sales Manager
15            Purchasing Manager
13            Product Manager
8             Marketing Manager
6             International Marketing Manager
1             Accounting Manager

(6 rows affected)



(1 row affected)
```

20. From the following tables write a query in SQL to make a list of contacts who are designated as 'Purchasing Manager'. Return BusinessEntityID, LastName, and FirstName columns. Sort the result set in ascending order of LastName, and FirstName

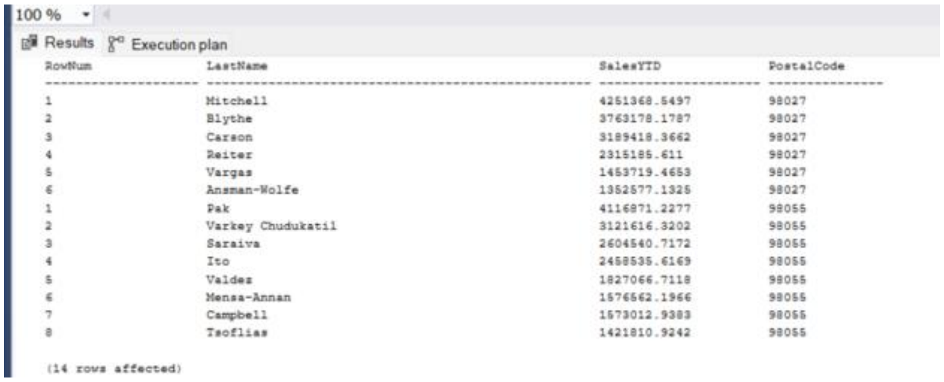
```
SELECT p.businessentityid, p.lastName, p.firstName
FROM [Person].[Person] as p
INNER JOIN [Person].[BusinessEntityContact] as pb ON p.[BusinessEntityID] =
pb.[PersonID]
INNER JOIN [Person].[ContactType] as pc ON pb.[ContactTypeID] =
pc.[ContactTypeID]
WHERE pc.Name = 'Purchasing Manager'
```



```
businessentityid lastName                                           firstName
---------------- -------------------------------------------------- --------------------------------------------------
1149             Alexander                                          Mary
363              Arakawa                                            Hannah
365              Arbelaez                                           Kyley
377              Ault                                               John
379              Avalos                                             Robert
389              Bailey                                             James
391              Baldwin                                            Douglas
399              Banks                                              Darrell
401              Barbariol                                          Angela
403              Barber                                             David
409              Barlow                                             Brenda
411              Barnhill                                           Josh
413              Barr                                               Adam
423              Bauer                                              Ciro
425              Beanston                                           Glenna
427              Beasley                                            Shaun
447              Ben-Sachar                                         Ido
449              Benson                                             Edna
453              Benson                                             Max
451              Benson                                             Payton
455              Bent                                               Scot
457              Bentley                                            Richard
465              Berger                                             John
467              Bergin                                             Kris
```

21. Write a query in SQL to retrieve the salesperson for each PostalCode who belongs to a territory and SalesYTD is not zero. Return row numbers of each group of PostalCode, last name, salesytd, postalcode column. Sort the salesytd of each postalcode group in descending order. Shorts the postalcode in ascending order

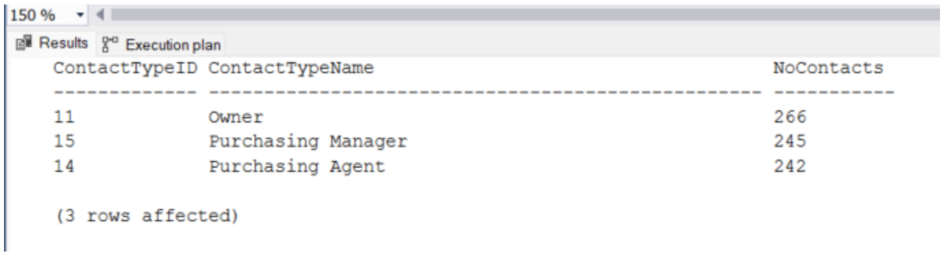
```
SELECT ROW_NUMBER() OVER (
PARTITION BY a.PostalCode
ORDER BY sp.SalesYTD DESC
) AS RowNum,
p.LastName,
sp.SalesYTD,
a.PostalCode
FROM Sales.SalesPerson sp
INNER JOIN Person.Person p
ON sp.BusinessEntityID = p.BusinessEntityID
INNER JOIN Person.Address a
ON p.BusinessEntityID = a.AddressID
INNER JOIN Sales.SalesTerritory st
ON sp.TerritoryID = st.TerritoryID
WHERE
sp.SalesYTD <> 0
AND a.PostalCode IS NOT NULL
ORDER BY
a.PostalCode ASC,
sp.SalesYTD DESC;
```



```
RowNum               LastName                                           SalesYTD              PostalCode
-------------------- -------------------------------------------------- --------------------- ---------------
1                    Mitchell                                           4251368.5497          98027
2                    Blythe                                             3763178.1787          98027
3                    Carson                                             3189418.3662          98027
4                    Reiter                                             2315185.611           98027
5                    Vargas                                             1453719.4653          98027
6                    Ansman-Wolfe                                       1352577.1325          98027
1                    Pak                                                4116871.2277          98055
2                    Varkey Chudukatil                                  3121616.3202          98055
3                    Saraiva                                            2604540.7172          98055
4                    Ito                                                2458535.6169          98055
5                    Valdez                                             1827066.7118          98055
6                    Mensa-Annan                                        1576562.1966          98055
7                    Campbell                                           1573012.9383          98055
8                    Tsoflias                                           1421810.9242          98055

(14 rows affected)
```

22. Write a query in SQL to count the number of contacts for combination of each type and name. Filter the output for those who have 100 or more contacts. Return ContactTypeID and ContactTypeName and BusinessEntityContact. Sort the result set in descending order on number of contacts

```
SELECT ct.ContactTypeID, ct.Name AS ContactTypeName, COUNT() AS NoContacts
FROM Person.ContactType AS ct
JOIN Person.BusinessEntityContact AS bec ON ct.ContactTypeID =
bec.ContactTypeID
GROUP BY ct.ContactTypeID, ct.Name HAVING COUNT() >= 100
ORDER BY NoContacts DESC
```

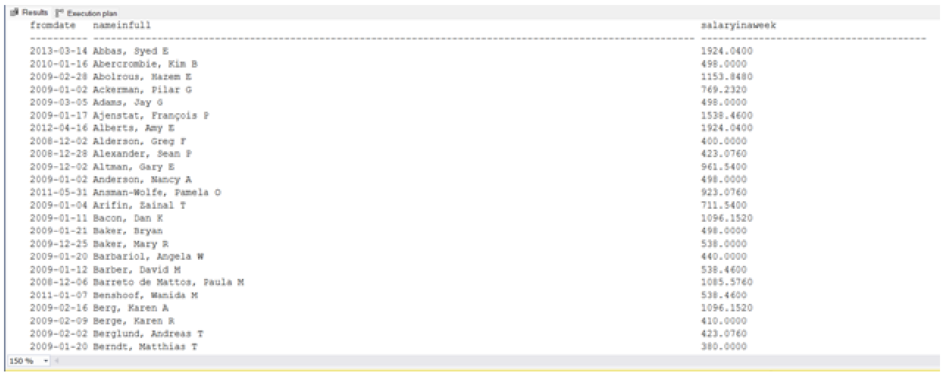


```
ContactTypeID ContactTypeName                                    NoContacts
------------- -------------------------------------------------- -----------
11            Owner                                              266
15            Purchasing Manager                                 245
14            Purchasing Agent                                   242

(3 rows affected)
```

23. Write a query in SQL to retrieve the RateChangeDate, full name (first name, middle name and last name) and weekly salary (40 hours in a week) of employees. In the output

the RateChangeDate should appears in date format. Sort the output in ascending order on NameInFull

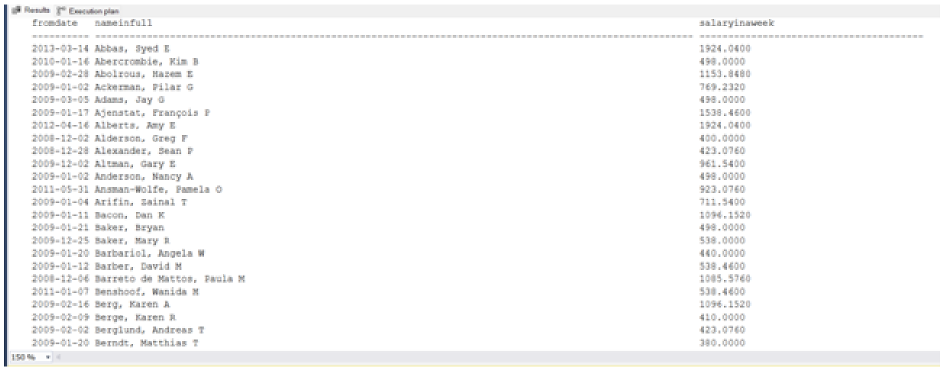
```
SELECT CONVERT(date, eph.RateChangeDate) AS fromdate, p.LastName + ', ' +
p.FirstName + COALESCE(' ' + LEFT(p.MiddleName, 1), '') AS nameinfull,
CAST(eph.Rate * 40 AS decimal(18,4)) AS salaryinaweek
FROM HumanResources.EmployeePayHistory AS eph
INNER JOIN HumanResources.Employee AS e ON e.BusinessEntityID =
eph.BusinessEntityID
INNER JOIN Person.Person AS p ON p.BusinessEntityID = e.BusinessEntityID
ORDER BY nameinfull ASC
```



```
fromdate   nameinfull                          salaryinaweek
---------- ----------------------------------- --------------------------------------
2013-03-14 Abbas, Syed E                       1924.0400
2010-01-16 Abercrombie, Kim B                  498.0000
2009-02-28 Abolrous, Hazem E                   1153.8480
2009-01-02 Ackerman, Pilar G                   769.2320
2009-03-05 Adams, Jay G                        498.0000
2009-01-17 Ajenstat, François P                1538.4600
2012-04-16 Alberts, Amy E                      1924.0400
2008-12-02 Alderson, Greg F                    400.0000
2008-12-28 Alexander, Sean P                   423.0760
2009-12-02 Altman, Gary E                      961.5400
2009-01-02 Anderson, Nancy A                   498.0000
2011-05-31 Ansman-Wolfe, Pamela O              923.0760
2009-01-04 Arifin, Zainal T                    711.5400
2009-01-11 Bacon, Dan K                        1096.1520
2009-01-21 Baker, Bryan                        498.0000
2009-12-25 Baker, Mary R                       538.0000
2009-01-20 Barbariol, Angela W                 440.0000
2009-01-12 Barber, David M                     538.4600
2008-12-06 Barreto de Mattos, Paula M          1085.5760
2011-01-07 Benshoof, Wanida M                  538.4600
2009-02-16 Berg, Karen A                       1096.1520
2009-02-09 Berge, Karen R                      410.0000
2009-02-02 Berglund, Andreas T                 423.0760
2009-01-20 Berndt, Matthias T                  380.0000
```

24. Write a query in SQL to calculate and display the latest weekly salary of each employee. Return RateChangeDate, full name (first name, middle name and last name) and weekly salary (40 hours in a week) of employees Sort the output in ascending order on NameInFull

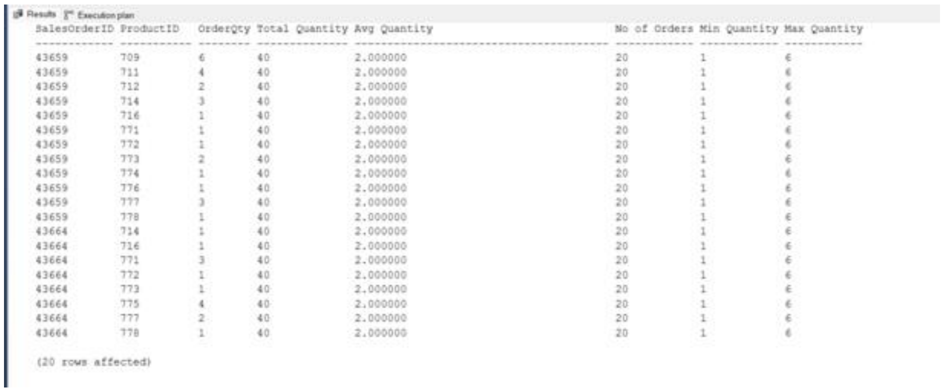
```
SELECT CONVERT(date, eph.RateChangeDate) AS fromdate, p.LastName + ', ' +
p.FirstName + COALESCE(' ' + LEFT(p.MiddleName, 1), '') AS nameinfull,
CAST(eph.Rate * 40 AS decimal(18,4)) AS salaryinaweek
FROM HumanResources.EmployeePayHistory AS eph
INNER JOIN HumanResources.Employee AS e ON e.BusinessEntityID =
eph.BusinessEntityID
INNER JOIN Person.Person AS p ON p.BusinessEntityID = e.BusinessEntityID
INNER JOIN (SELECT BusinessEntityID, MAX(RateChangeDate) AS LatestChange
FROM HumanResources.EmployeePayHistory
GROUP BY BusinessEntityID
) AS latest ON latest.BusinessEntityID = eph.BusinessEntityID AND
latest.LatestChange = eph.RateChangeDate
ORDER BY nameinfull ASC
```



```
fromdate   nameinfull                                  salaryinaweek
---------- ------------------------------------------- -------------------
2013-03-14 Abbas, Syed E                               1924.0400
2010-01-16 Abercrombie, Kim B                          498.0000
2009-02-28 Abolrous, Hazem E                           1153.8480
2009-01-02 Ackerman, Pilar G                           769.2320
2009-03-05 Adams, Jay G                                498.0000
2009-01-17 Ajenstat, François P                        1538.4600
2012-04-16 Alberts, Amy E                              1924.0400
2008-12-02 Alderson, Greg F                            400.0000
2008-12-28 Alexander, Sean P                           423.0760
2009-12-02 Altman, Gary E                              961.5400
2009-01-02 Anderson, Nancy A                           498.0000
2011-05-31 Ansman-Wolfe, Pamela O                      923.0760
2009-01-04 Arifin, Zainal T                            711.5400
2009-01-11 Bacon, Dan K                                1096.1520
2009-01-21 Baker, Bryan                                498.0000
2009-12-25 Baker, Mary R                               538.0000
2009-01-20 Barbariol, Angela W                         440.0000
2009-01-12 Barber, David M                             538.4600
2008-12-06 Barreto de Mattos, Paula M                  1085.5760
2011-01-07 Benshoof, Wanida M                          538.4600
2009-02-16 Berg, Karen A                               1096.1520
2009-02-09 Berge, Karen R                              410.0000
2009-02-02 Berglund, Andreas T                         423.0760
2009-01-20 Berndt, Matthias T                          380.0000
```

25. Write a query in SQL to find the sum, average, count, minimum, and maximum order quentity for those orders whose id are 43659 and 43664. Return SalesOrderID, ProductID, OrderQty, sum, average, count, max, and min order quantity.

```
SELECT d.SalesOrderID, d.ProductID, d.OrderQty, agg.TotalQty AS [Total
Quantity], agg.AvgQty AS [Avg Quantity], agg.OrderCount AS [No of Orders],
agg.MinQty AS [Min Quantity], agg.MaxQty AS [Max Quantity]
FROM Sales.SalesOrderDetail AS d
CROSS JOIN (SELECT SUM(d2.OrderQty) AS TotalQty, AVG(CAST(d2.OrderQty AS
decimal(18,6))) AS AvgQty, COUNT(*) AS OrderCount, MIN(d2.OrderQty) AS
MinQty,
MAX(d2.OrderQty) AS MaxQty
FROM Sales.SalesOrderDetail AS d2
WHERE d2.SalesOrderID IN (43659, 43664)) AS agg
WHERE d.SalesOrderID IN (43659, 43664)
ORDER BY d.SalesOrderID, d.ProductID
```



```
SalesOrderID ProductID   OrderQty Total Quantity Avg Quantity                            No of Orders Min Quantity Max Quantity
------------ ----------- -------- -------------- --------------------------------------- ------------ ------------ ------------
43659        709         6        40             2.000000                                20           1            6
43659        711         4        40             2.000000                                20           1            6
43659        712         2        40             2.000000                                20           1            6
43659        714         3        40             2.000000                                20           1            6
43659        716         1        40             2.000000                                20           1            6
43659        771         1        40             2.000000                                20           1            6
43659        772         1        40             2.000000                                20           1            6
43659        773         2        40             2.000000                                20           1            6
43659        774         1        40             2.000000                                20           1            6
43659        776         1        40             2.000000                                20           1            6
43659        777         3        40             2.000000                                20           1            6
43659        778         1        40             2.000000                                20           1            6
43664        714         1        40             2.000000                                20           1            6
43664        716         1        40             2.000000                                20           1            6
43664        771         3        40             2.000000                                20           1            6
43664        772         1        40             2.000000                                20           1            6
43664        773         1        40             2.000000                                20           1            6
43664        775         4        40             2.000000                                20           1            6
43664        777         2        40             2.000000                                20           1            6
43664        778         1        40             2.000000                                20           1            6

(20 rows affected)
```