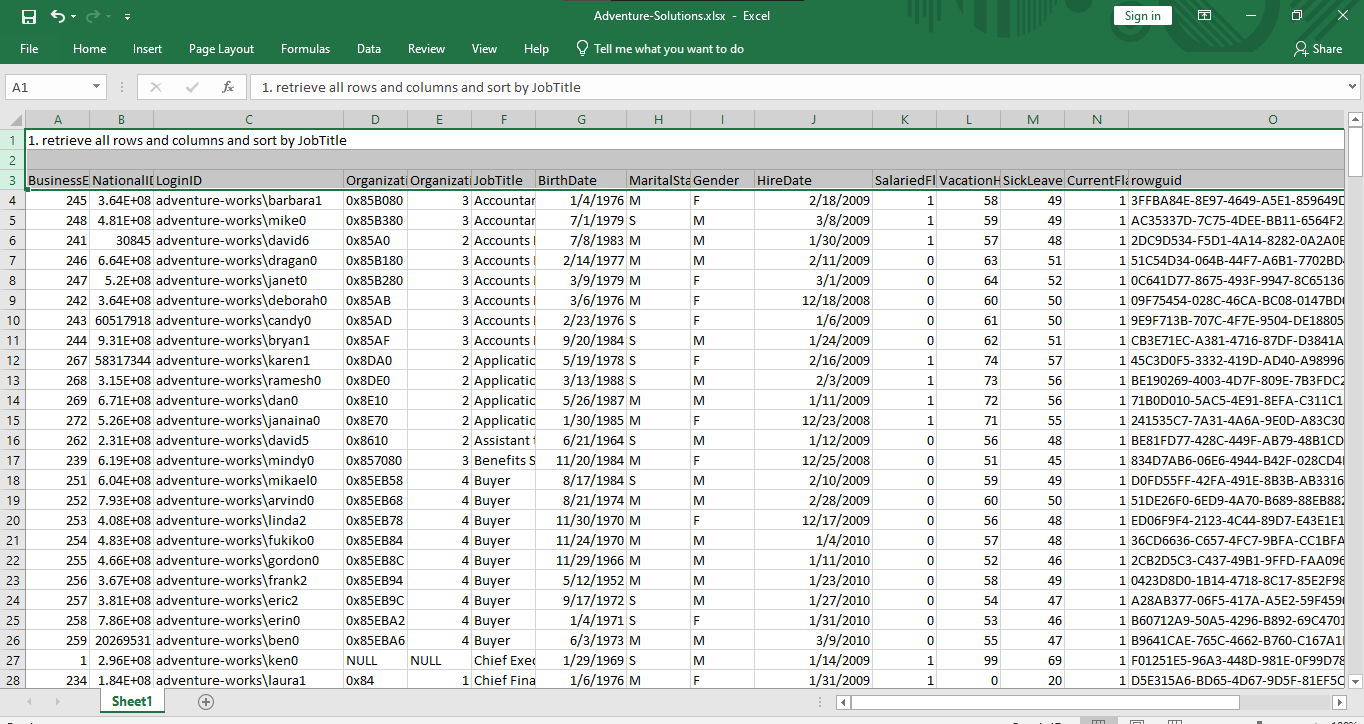
1. From the following table write a query in SQL to retrieve all rows and columns from the employee table in the Adventureworks database. Sort the result set in ascending order on jobtitle.

select \* from HumanResources.Employee order by JobTitle



=====================================================================================

2. From the following table 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.BusinessEntityID,

ISNULL(P.PersonType,'') as PersonType,

ISNULL(nullif(P.NameStyle,0),'false') as NameStyle,

isnull(P.Title,'') as Title,

ISNULL(P.FirstName,'')as FirstName,

ISNULL(P.MiddleName,'') as MiddleName,

ISNULL(P.LastName,'') as LastName,

ISNULL(P.Suffix,'') as Suffix,

ISNULL(P.EmailPromotion,'') as EmailPromotion,

ISNULL(P.AdditionalContactInfo,'') as AdditionalContactInfo,

ISNULL(P.Demographics,'') as Demographics,

ISNULL(P.rowguid,'') as rowguid ,

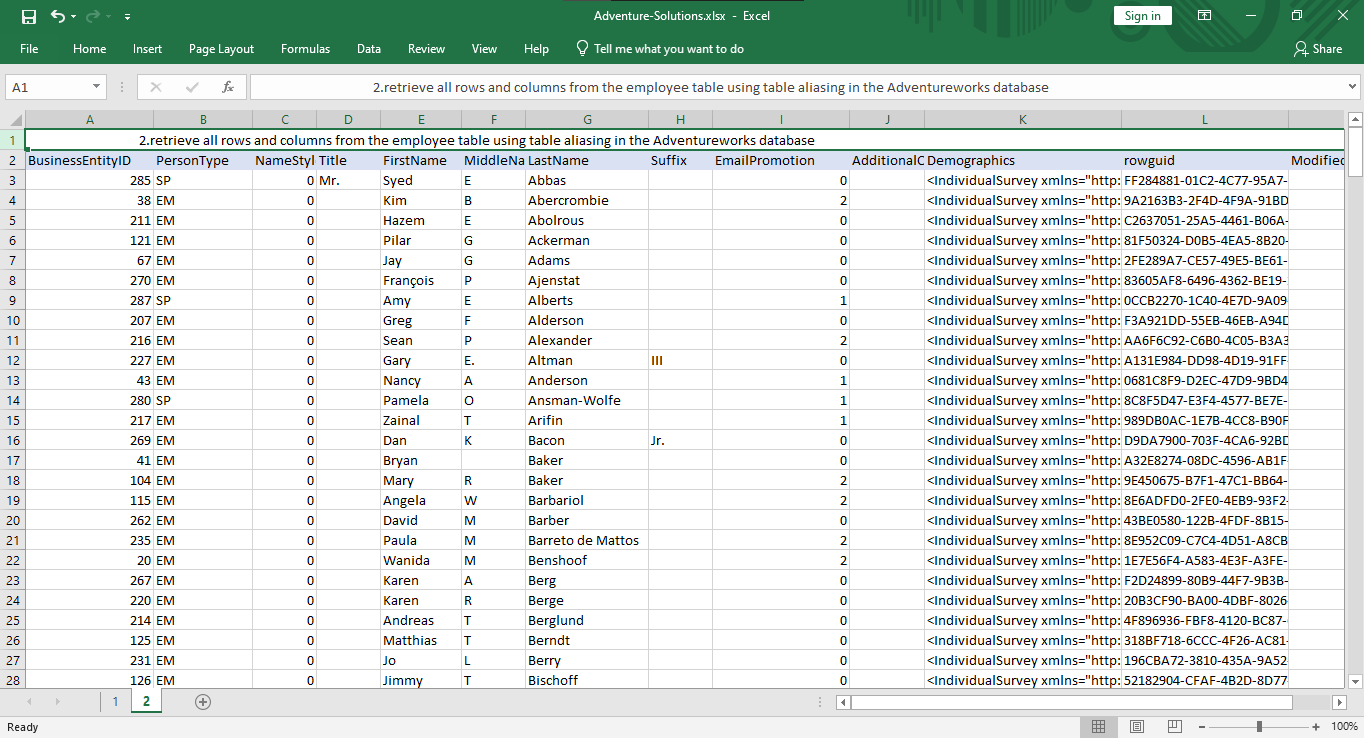
ISNULL(P.ModifiedDate,'')as ModifiedDate

from HumanResources.Employee E

join Person.Person P

on E.BusinessEntityID= P.BusinessEntityID

order by P.LastName asc



=======================================================================================

3. From the following table 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.

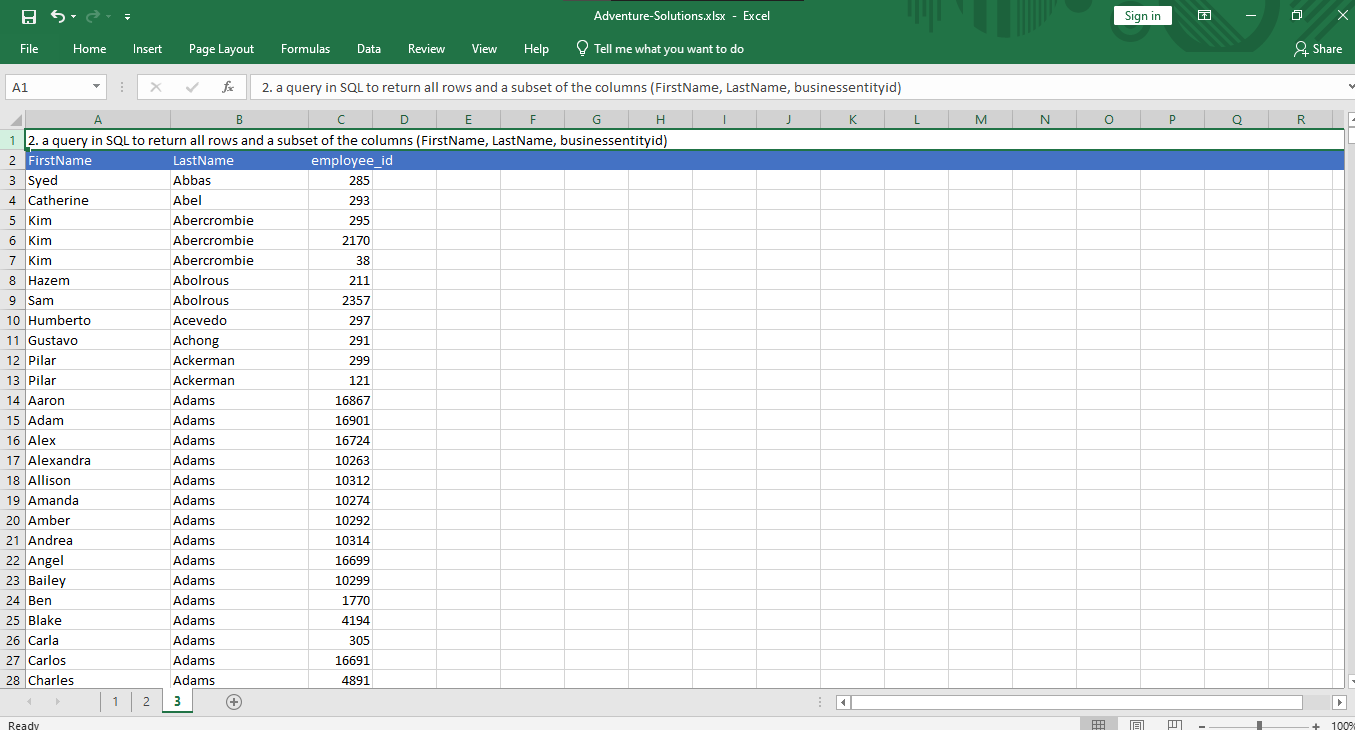
select P.FirstName,

P.LastName,

P.BusinessEntityID as employee\_id

from Person.Person P

order by P.LastName asc



=======================================================================================

4. From the following table 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.

select P.ProductID,

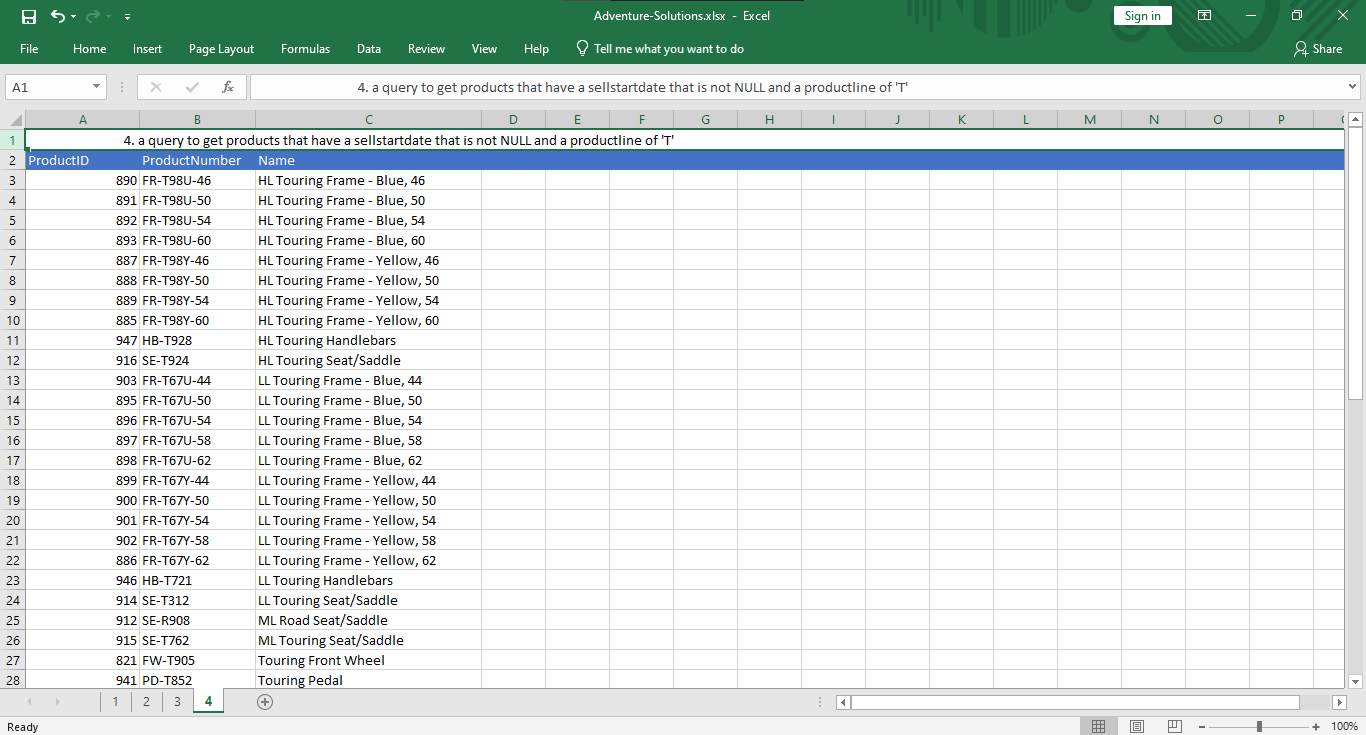
P.ProductNumber,

P.Name,

from production.Product P

where P.SellStartDate is not null and P.ProductLine='T'

order by P.Name asc



=====================================================================================

5. From the following table 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,

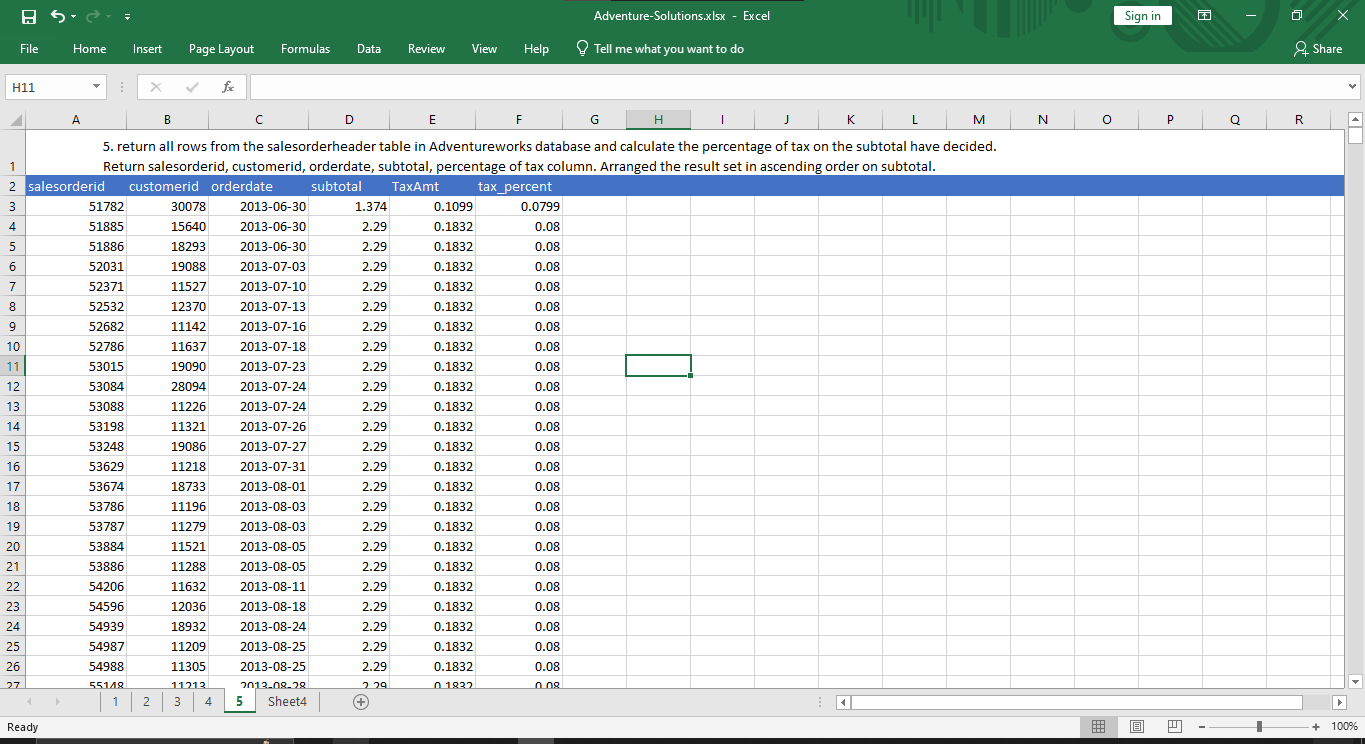
subtotal,

TaxAmt,

(TaxAmt/subtotal) as tax\_percent

from sales.salesorderheader S

order by subtotal asc



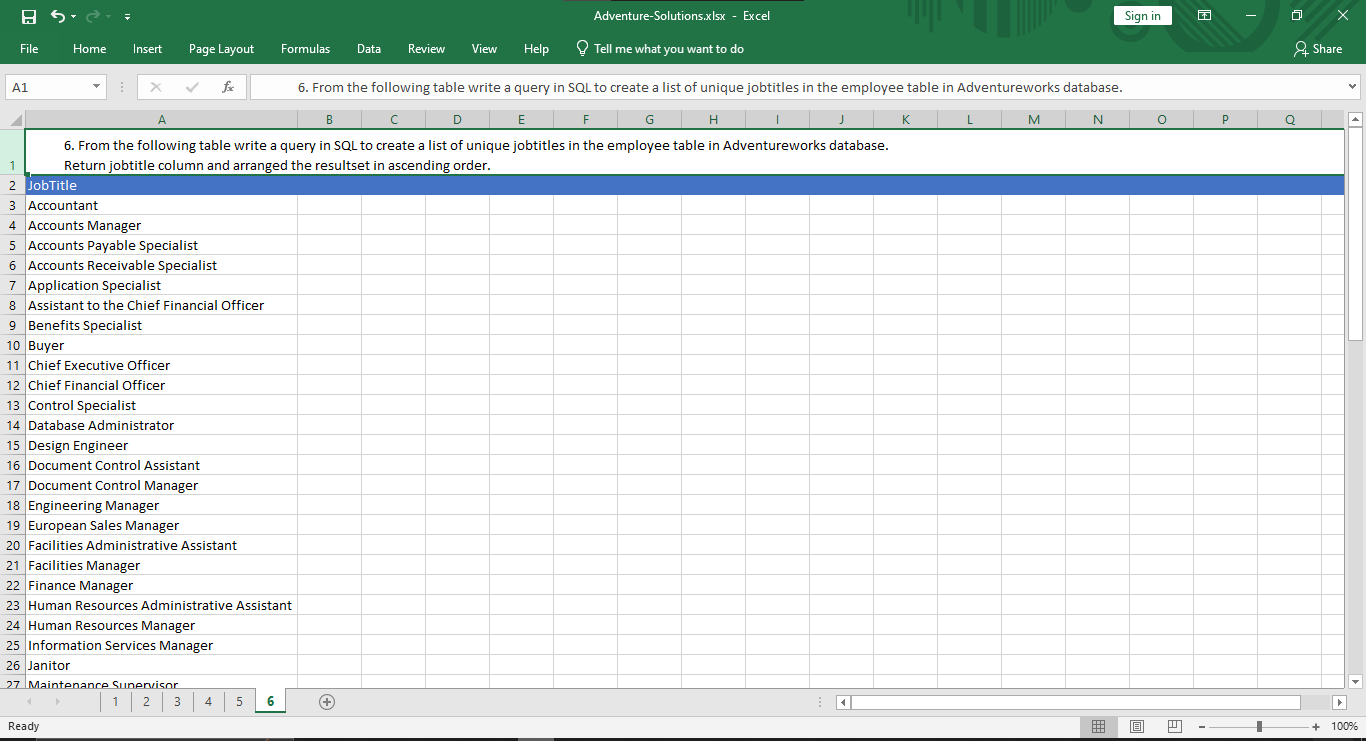
===========================================================================

6. From the following table 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



========================================================================================

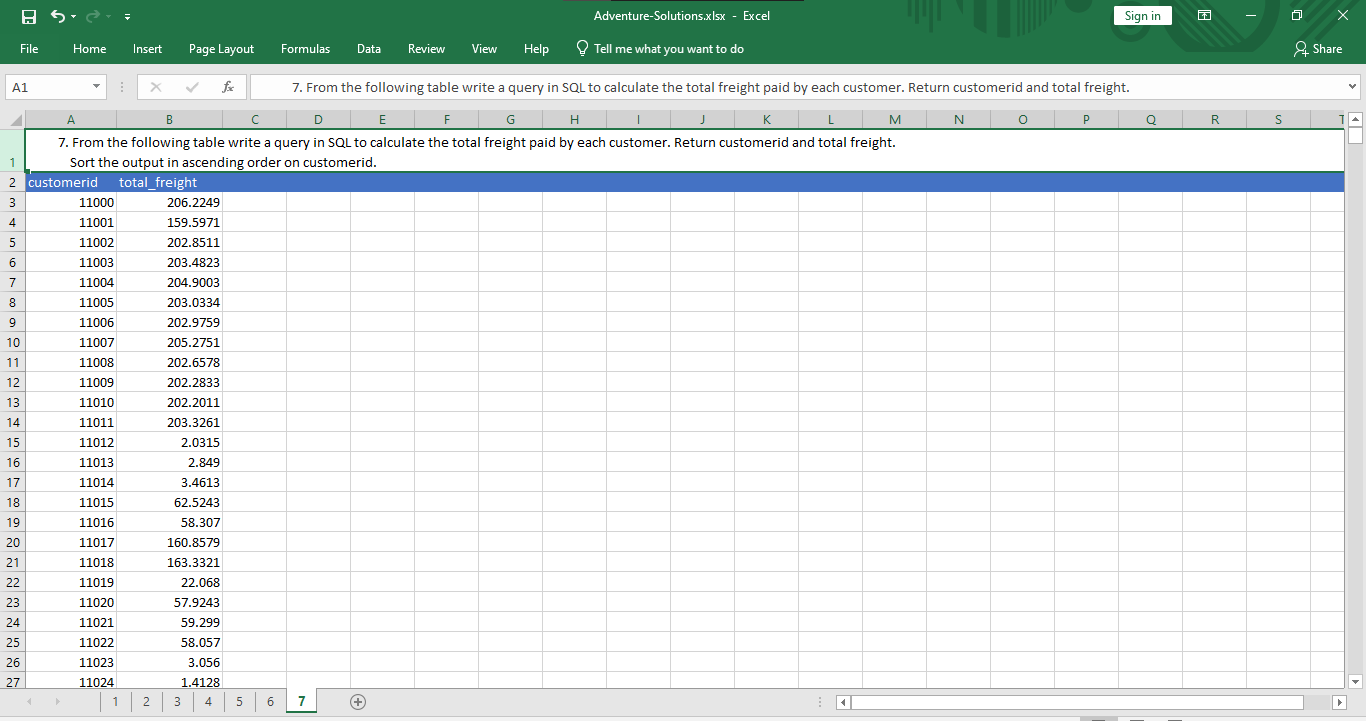
7. From the following table write a query in SQL to calculate the total freight paid by each customer. Return customerid and total freight. Sort the output in ascending order on customerid.

select customerid,sum(Freight) as total\_freight

from sales.salesorderheader

group by CustomerID

order by customerid asc



=========================================================================

8. From the following table 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,

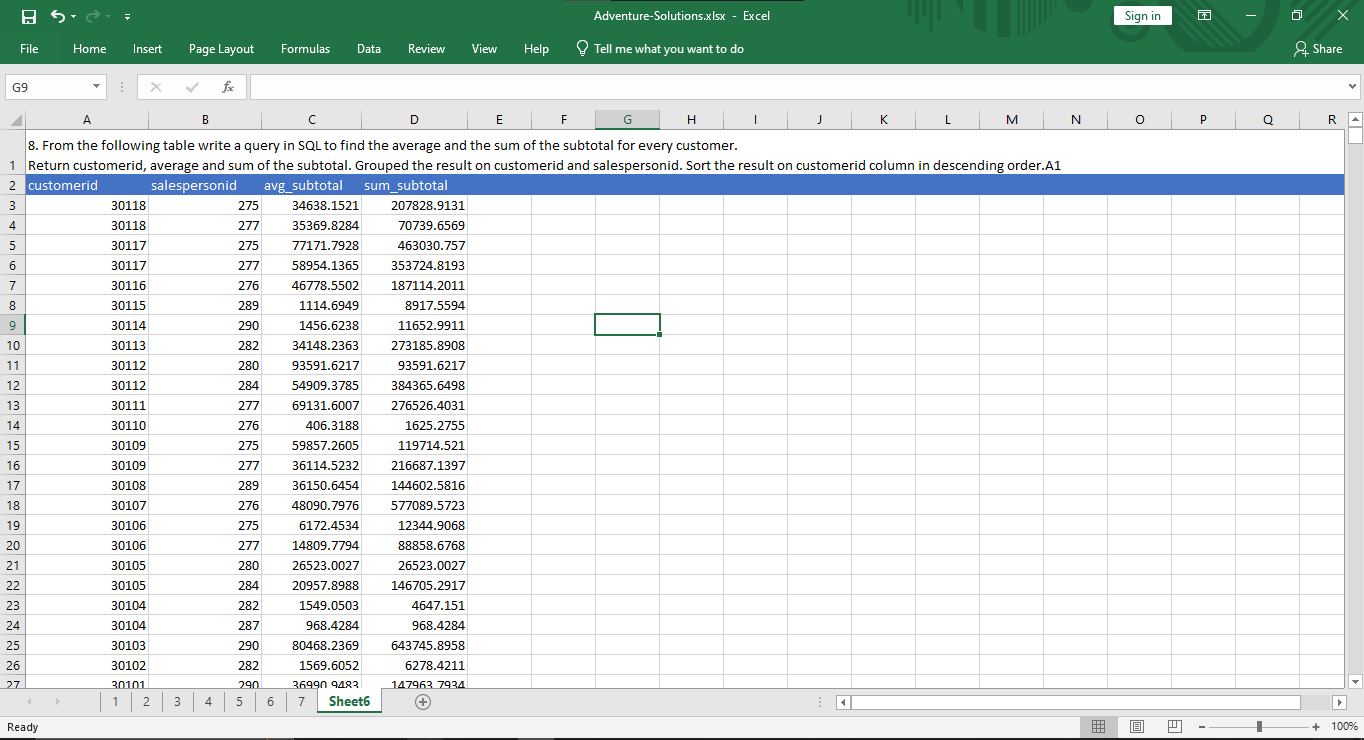
(sum(subtotal)/count(\*)) as avg\_subtotal,

sum(subtotal) as sum\_subtotal

from sales.salesorderheader

group by customerid, salespersonid

order by customerid desc



==========================================================================

9. From the following table 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,

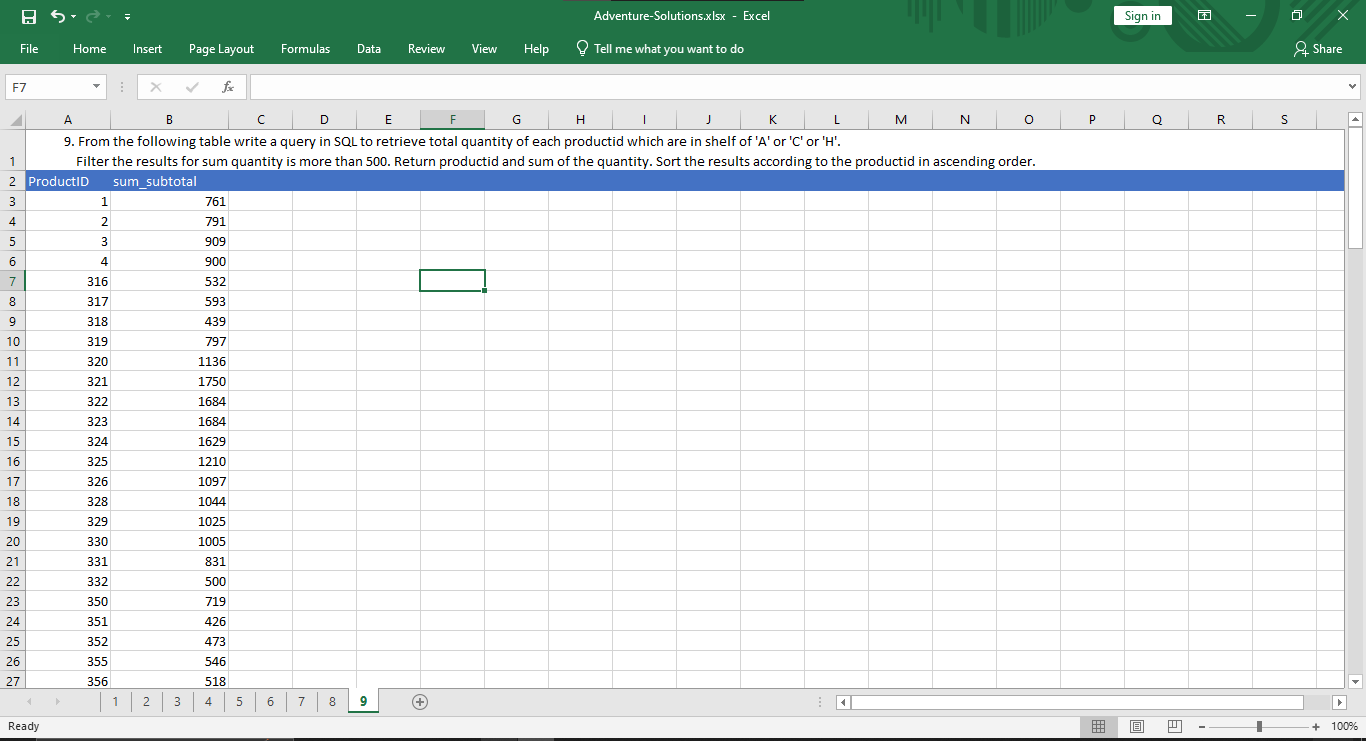
sum(Quantity) as sum\_subtotal

from production.productinventory

where shelf in ('A','C','H')

group by ProductID

order by productid asc



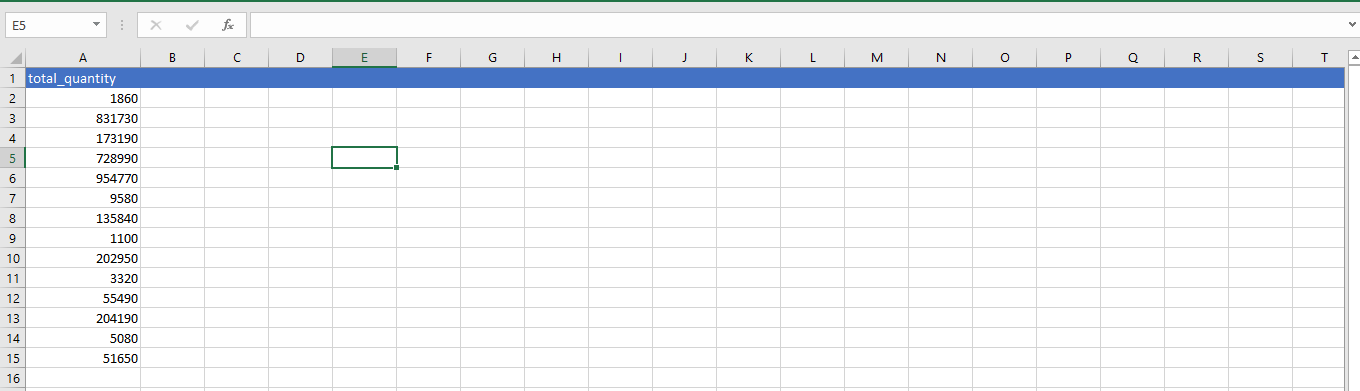
=========================================================================

10. From the following table write a query in SQL to find the total quentity for a group of locationid multiplied by 10.

select sum (Quantity) as sum\_subtotal

from production.productinventory

group by LocationID



===========================================================================

11. From the following tables 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 P.BusinessEntityID,

FirstName,

LastName,

F.PhoneNumber as person\_phone

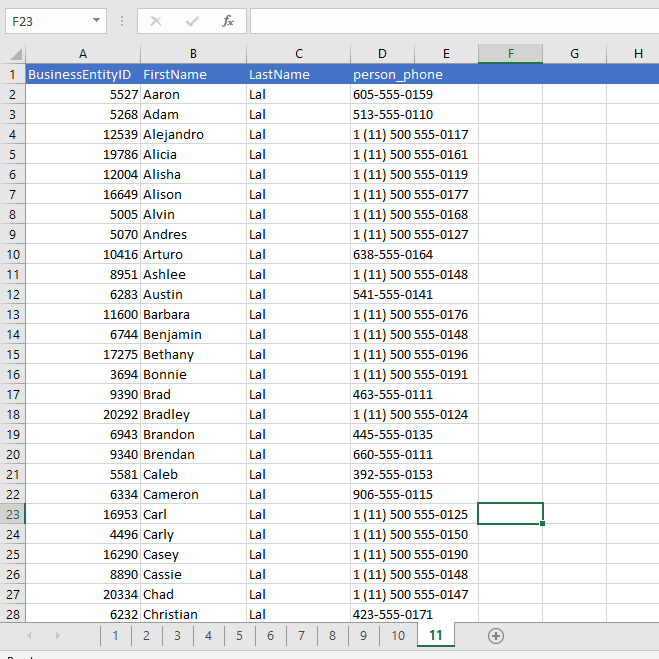
from Person.Person P

join Person.PersonPhone F

on P.BusinessEntityID=F.BusinessEntityID

where LastName like 'L%'

order by LastName,FirstName



==========================================================================

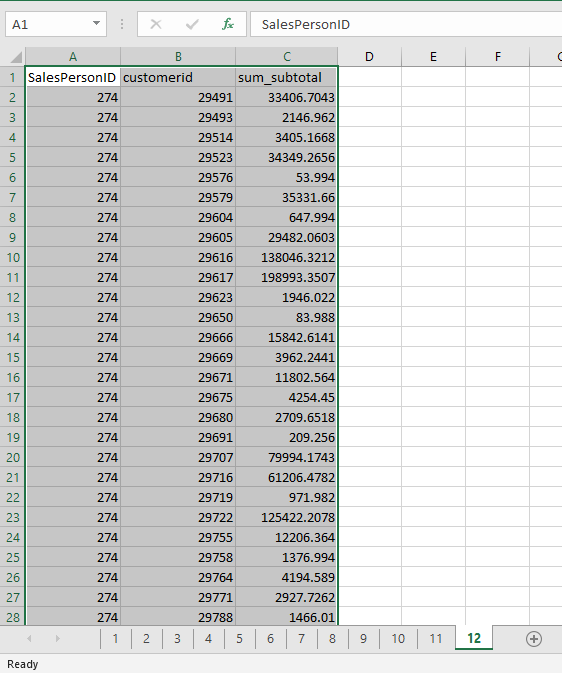
12. From the following table write a query in SQL to find the sum of subtotal column. Group the sum on distinct salespersonid and customerid. Rolls up the results into subtotal and running total. Return salespersonid, customerid and sum of subtotal column i.e. sum\_subtotal.

SELECT salespersonid,customerid,sum(subtotal) AS sum\_subtotal

FROM sales.salesorderheader s

where SalesPersonID is not null

GROUP BY ROLLUP (salespersonid, customerid);



===========================================================================

13. From the following table 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

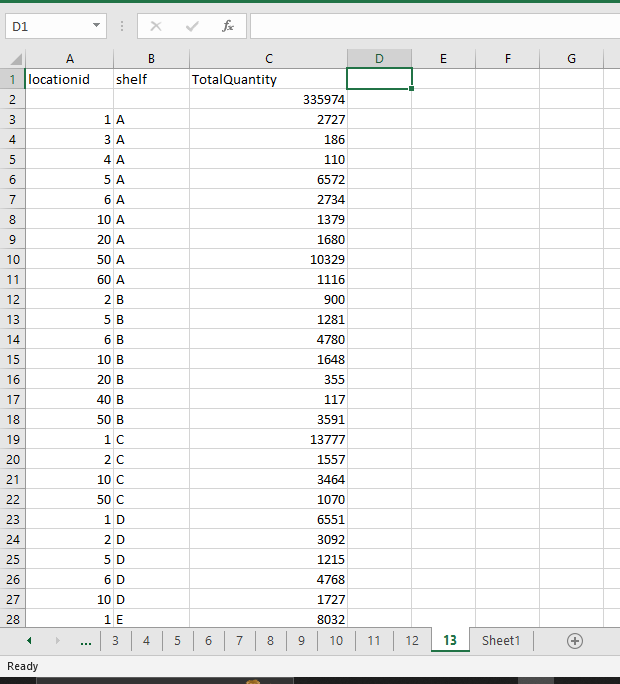
GROUP BY GROUPING SETS

(

(locationid, shelf),

()

)



===========================================================================

14. From the following table 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

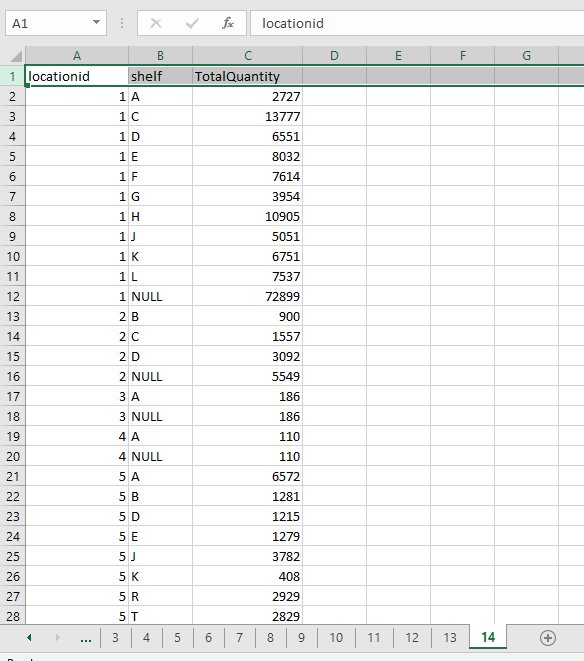
GROUP BY CUBE (locationid, shelf);

-------------------------------------OR----------------------------------------

SELECT locationid, shelf, SUM (quantity) AS TotalQuantity

FROM production.productinventory

GROUP BY ROLLUP (locationid, Shelf)



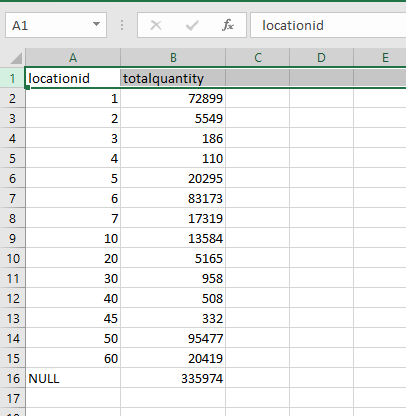
===========================================================================

15. From the following table 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) totalquantity

from production.productinventory

group by grouping sets (locationid,( ))



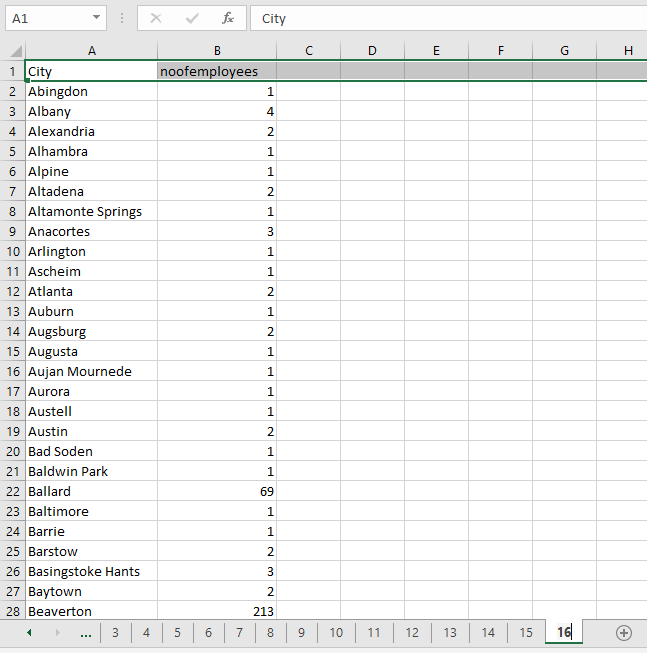
==========================================================================

16. From the following table 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(City)

from Person.Address

group by City order by City



==========================================================================

17. From the following table 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.

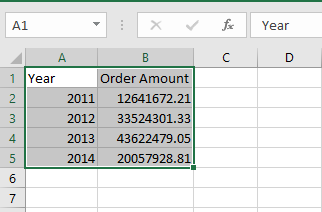
select YEAR(OrderDate) as [Year],

SUM(SubTotal) as [Order Amount]

from Sales.SalesOrderHeader

group by YEAR(OrderDate)

order by YEAR(OrderDate)



==========================================================================

18. From the following table 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 [Year],

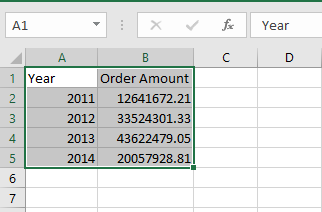
SUM(SubTotal) as [Order Amount]

from Sales.SalesOrderHeader

where YEAR(OrderDate) < '2016'

group by YEAR(OrderDate)

order by YEAR(OrderDate)



==========================================================================

19. From the following table 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'

order by ContactTypeID desc

