Class 2 & 3 Problem Set:

This problem set uses the WideWorldImporters database and covers material used in Modules 4, 5, & 6. This set covers two weeks.

1. Write a query that returns the StockItemID and StockItemName from the Warehouse.StockItems table. Create a third column that combines these two columns with a dash. Hint: you will need to use a function to avoid a conversion error.

   Here is an example of how the resulting column should look:
   3 – Office cube periscope (Black)

2. Write a query from the Warehouse.StockItems table. Include StockItemID, SupplierID, and ColorID. Include a column that adds up these three. Make sure that no NULLs end up in the calculated values.
3. Write a query from the Warehouse.StockItems table. Include StockItemID and the first 10 characters of the StockItemName.
4. Write a query from the Sales.Orders table. Include the OrderID, CustomerID, OrderDate, and ExpectedDeliveryDate. Include a column that returns the number of days between these two dates.
5. Modify the query from Question 4 so that only rows with more than one day between the OrderDate and ExpectedDeliveryDate are returned.
6. Write a query that joins the Sales.Orders and Sales.OrderLines tables on the OrderID column. Include OrderID, CustomerID, OrderDate, StockItemID, Quantity, and UnitPrice. Include a column that calculates the extended price for each line item.
7. Take the query from Question 6 and join to the Warehouse.StockItems table on the StockItemID column. Add the StockItemName to the returned columns.
8. Are there any rows in Warehouse.StockItems that have never been ordered? Write a query joining Warehouse.StockItems to Sales.OrderLines to find out. Hint: this will not be an INNER JOIN.
9. Write a query that returns a list of cities along with the state or province and country. See if you can figure out the three tables to use and how they are joined together. Return the CityName, StateProvinceName, and CountryName.
10. Filter the query written in Question 9 so that it returns only rows from outside the US.
11. Write a query that finds the average transaction amount from the Purchasing.SuppplierTransaction table.
12. Change the query from Question 11 so you get the average transaction amount for each SupplierID.
13. Write a query that joins the Purchasing. Suppliers to Purchasing.SupplierTransactions on SupplierID. Return a list of SupplierName and the number of transactions for each one.
14. Rewrite the query in Question 13 so that only rows with over 1000 transactions are returned.