Exercise 1

Create a query with the following columns:

- “PurchaseOrderID” from the Purchasing.PurchaseOrderHeader table
- “OrderDate” from the Purchasing.PurchaseOrderHeader table
- “TotalDue” from the Purchasing.PurchaseOrderHeader table
- “Name” from the Purchasing.Vendor table, which can be aliased as “VendorName”*****

**Join Purchasing.Vendor to Purchasing.PurchaseOrderHeader on BusinessEntityID = VendorID*

Apply the following criteria to the query:

- Order must have taken place on or after 2013
- TotalDue must be greater than $500

Exercise 2

Modify your query from Exercise 1 by adding a derived column called

"PrevOrderFromVendorAmt", that returns the “previous” TotalDue value (relative to the current row) *within the group of all orders with the same vendor ID.* We are defining “previous” based on order date.

Exercise 3

Modify your query from Exercise 2 by adding a derived column called

"NextOrderByEmployeeVendor", that returns the “next” vendor name (the “name” field from Purchasing.Vendor) *within the group of all orders that have the same EmployeeID value in* Purchasing.PurchaseOrderHeader. Similar to the last exercise, we are defining “next” based on order date.

Exercise 4

Modify your query from Exercise 3 by adding a derived column called "Next2OrderByEmployeeVendor" that returns, within the group of all orders that have the same EmployeeID*, the vendor name offset TWO orders into the “future” relative to the order in the current row.* The code should be very similar to Exercise 3, but with an extra argument passed to the Window Function used.