LINQ Querries

Vendor Info – POCO Can be used to find the parts, display vendor info, find the purchase Order

var vendorInfo = from vendor in Vendors

select new

{

vendorid = vendor.VendorID,

vendorname = vendor. VendorName,

vendorlocation = vendor.City,

vendorparts = from part in vendor.Parts

select new

{

partid = part.PartID

},

vendorpurchaseorders = from order in vendor.PurchaseOrders

where order.OrderDate == null

select order.PurchaseOrderID

};

vendorInfo.Dump();

//current order

var result = from order in PurchaseOrderDetails

where order.PurchaseOrder.VendorID == 1

&& order.PurchaseOrder.OrderDate == null

select new

{

PurchaseOrderDetailID = order.PurchaseOrderDetailID,

PartID = order.PartID,

Description = order.Part.Description,

QOH = order.Part.QuantityOnHand,

QOO = order.Part.QuantityOnOrder,

ROL = order.Part.ReorderLevel,

Qty = order.Quantity,

Price = order.PurchasePrice

};

result.Dump();

//suggested order

var result = from part in Parts

where part.Vendor.VendorID == 2 &&

(part.ReorderLevel - (part.QuantityOnHand + part.QuantityOnOrder))>0

select new

{

PartID = part.PartID,

Description = part.Description,

QOH = part.QuantityOnHand,

QOO = part.QuantityOnOrder,

ROL = part.ReorderLevel,

Qty = part.ReorderLevel - part.QuantityOnHand,

Price = part.PurchasePrice

};

result.Dump();

suggested order with someone current inventory

var result1 = from order in PurchaseOrderDetails

where order.PurchaseOrder.VendorID == 1

&& order.PurchaseOrder.OrderDate == null

select new

{

PartID = order.PartID

};

result1.Dump();

var result2 = from part in Parts

where part.VendorID==1

select new{

PartID = part.PartID,

Description = part.Description,

QOH = part.QuantityOnHand,

QOO = part.QuantityOnOrder,

ROL = part.ReorderLevel,

Buffer = part.QuantityOnHand - part.ReorderLevel,

Price = part.PurchasePrice

};

result2.Dump();

var result3 = result2.Where(p => !result1.Any(p2 => p2.PartID== p.PartID));

result3.Dump();

//vendor inventory

var result1 = from order in context.PurchaseOrderDetails

where order.PurchaseOrder.VendorID == 1

&& order.PurchaseOrder.OrderDate == null

select new CurrentOrder

{

PurchaseOrderID = order.PurchaseOrder.PurchaseOrderID,

PurchaseOrderDetailID = order.PurchaseOrderDetailID,

PartID = order.PartID,

Description = order.Part.Description,

QOH = order.Part.QuantityOnHand,

QOO = order.Part.QuantityOnOrder,

ROL = order.Part.ReorderLevel,

Buffer = (order.Part.QuantityOnHand + order.Part.QuantityOnHand) - order.Part.ReorderLevel,

Price = order.Part.PurchasePrice

};

var result2 = from part in context.Parts

where part.VendorID == 1

select new CurrentOrder

{

PurchaseOrderID = (from data in result1

select data.PurchaseOrderID).First(),

PartID = part.PartID,

Description = part.Description,

QOH = part.QuantityOnHand,

QOO = part.QuantityOnOrder,

ROL = part.ReorderLevel,

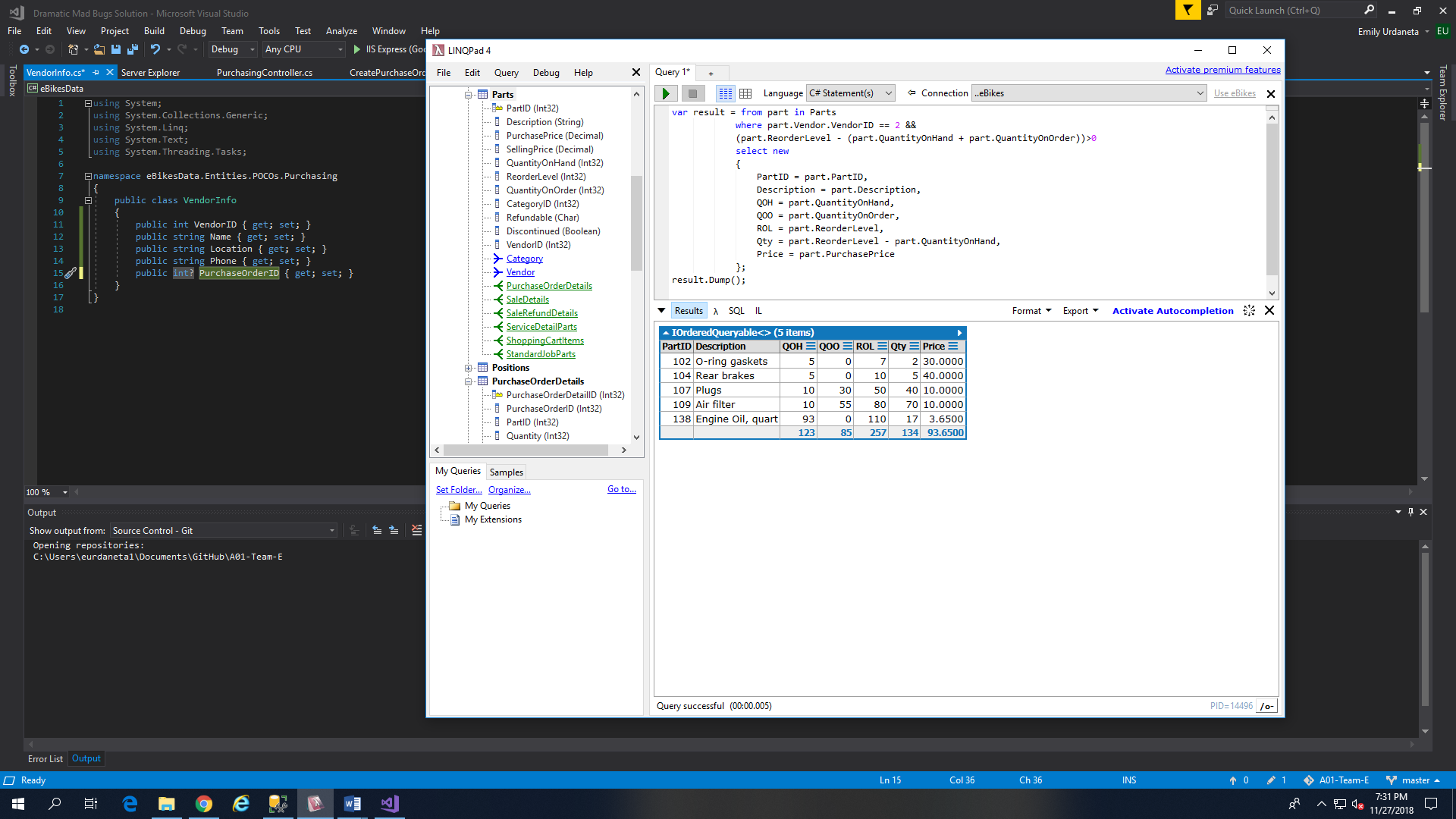
Buffer = part.QuantityOnHand - part.ReorderLevel,

Price = part.PurchasePrice

};

var result3 = result2.Where(p => !result1.Any(p2 => p2.PartID == p.PartID));

result3.Dump();



var result = from part in Parts

where part.Vendor.VendorID == 2 &&

(part.ReorderLevel - (part.QuantityOnHand + part.QuantityOnOrder))<0

select new

{

PartID = part.PartID,

Description = part.Description,

QOH = part.QuantityOnHand,

QOO = part.QuantityOnOrder,

ROL = part.ReorderLevel,

Buffer = part.QuantityOnHand - part.ReorderLevel,

Price = part.PurchasePrice

};

result.Dump();