**Question 1:**

SELECT DISTINCT e.EmpID, (e.emplastname + ', ' + e.empfirstname) as EmployeeName

FROM tblemployee e

INNER JOIN tblpurchaseorder po

ON e.empid = po.buyerempid

ORDER BY e.Empid;

Graphical user interface, text, application

Description automatically generated

**Question 2:**

SELECT e.EmpID, (e.emplastname + ', ' + e.empfirstname) as EmployeeName,

isnull((m.emplastname + ', ' + m.empfirstname), 'No Manager') as ManagerName, po.PONumber, po.PODatePlaced, po.VendorID,

v.VendorName, UPPER(v.VendorCity + ', ' +v.VendorState) as VendorLocation

FROM tblemployee e

INNER JOIN tblpurchaseorder po

ON e.empid = po.buyerempid

LEFT OUTER JOIN tblemployee m

ON e.empmgrid = m.empid

INNER JOIN tblVendor v

ON v.vendorID = po.VendorID

ORDER BY e.Empid, po.PONumber;

Table

Description automatically generated

**Question 3:**

CREATE VIEW TotalQuantity as

SELECT polineid, SUM(qtyreceived) as TotalQtyReceived

from tblreceiver

GROUP BY polineid

SELECT pol.PONumber, CONVERT(VARCHAR, po.PODatePlaced, 107) as PODatePlaced, v.VendorName,

isnull((e.emplastname + ', ' + UPPER(LEFT(empfirstname, 1)) + '.'), 'No Buyer on File') as EmployeeBuyer,

p.ProductID, p.ProductDescription,

CONVERT(VARCHAR, pol.dateNeeded, 107) as DateNeeded, pol.Price, pol.QtyOrdered,

isnull(TotalQuantity.TotalQtyReceived, 0) as TotalQtyReceived,

(pol.QtyOrdered - isnull(TotalQuantity.TotalQtyReceived, 0)) as QtyRemaining,

CASE

WHEN pol.qtyordered - TotalQtyReceived = 0 then 'Complete'

WHEN pol.qtyordered - TotalQtyReceived < 0 then 'Over Shipped'

WHEN pol.qtyordered - TotalQtyReceived > 0 then 'Partially Received'

WHEN isnull(TotalQtyReceived,0) = 0 then 'Not Received'

END 'ReceivingStatus'

FROM tblpurchaseorderline pol

LEFT OUTER JOIN tblpurchaseorder po

ON pol.PONumber = po.PONumber

INNER JOIN tblVendor v

ON po.VendorID = v.VendorID

LEFT OUTER JOIN tblEmployee e

on po.BuyerEmpID = e.EmpID

LEFT OUTER JOIN tblProduct p

ON pol.ProductID = p.ProductID

LEFT OUTER JOIN TotalQuantity

ON pol.polineid = TotalQuantity.polineid

ORDER BY pol.PONumber, p.ProductID;

Table

Description automatically generated

**Question 4:**

CREATE VIEW OrderStatus as

SELECT DISTINCT pol.PONumber, 'Open' as Status

FROM tblpurchaseOrderline pol

LEFT OUTER JOIN TotalQuantity

ON pol.polineid = TotalQuantity.polineid

WHERE pol.qtyordered - TotalQtyReceived > 0 or isnull(TotalQtyReceived,0) = 0

SELECT DISTINCT pol.PONumber, CONVERT(VARCHAR, po.PODatePlaced, 107) as PODatePlaced,

CONVERT(VARCHAR, po.PODateNeeded,107) as PODateNeeded,

V.VendorName

FROM tblpurchaseorderline pol

LEFT OUTER JOIN tblpurchaseorder po

ON pol.PONumber = po.PONumber

LEFT OUTER JOIN tblVendor v

ON po.VendorID = v.VendorID

WHERE pol.PONumber NOT IN (SELECT PONumber FROM OrderStatus);

Table

Description automatically generated

**Question 5:**

SELECT OS.PONumber, CONVERT(VARCHAR, po.PODatePlaced, 107) as PODatePlaced,

CONVERT(VARCHAR, po.PODateNeeded,107) as PODateNeeded, V.VendorName

FROM OrderStatus os

LEFT OUTER JOIN tblpurchaseorder po

ON os.PONumber = po.PONUmber

LEFT OUTER JOIN tblVendor v

ON po.VendorID = v.VendorID;

Table

Description automatically generated

**Question 6:**

CREATE VIEW UpdatedDate as

SELECT MAX(DatePurchased) as Recent, ProductID

FROM tblpurchasehistory

GROUP BY productID

CREATE VIEW HistoricPrice as

SELECT p.ProductID, p.ProductDescription,

isnull(CONVERT(VARCHAR, ph.datepurchased,107), 'Not in Purchase History') as MostRecentPurchaseDate,

ph.price as PurchasePrice, V.VendorName

FROM tblproduct p

FULL JOIN tblpurchasehistory ph

ON ph.productid = p.productid

FULL JOIN UpdatedDate UD

ON UD.ProductID = p.ProductID

LEFT OUTER JOIN tblVendor v

ON v.vendorid = ph.Vendorid

WHERE ph.Datepurchased = UD.Recent or UD.Recent is Null

SELECT hp.ProductID, hp.ProductDescription, hp.PurchasePrice as RecentHistoricPrice, hp.vendorName as VendorNamePast,

pol.price as CurrentPrice, pol.PONumber, v.VendorName as VendorNameCurrent

FROM Historicprice hp

INNER JOIN tblPurchaseorderline pol

ON pol.Productid = hp.ProductID

INNER JOIN tblpurchaseorder po

ON po.POnumber = pol.PoNumber

INNER JOIN tblVendor v

ON po.Vendorid = v.vendorid

WHERE (hp.PurchasePrice \* 1.2) < pol.price

ORDER BY hp.Productid, v.VendorName;

Graphical user interface, application

Description automatically generated

**Question 7:**

SELECT po.VendorID, COUNT(pol.polineid) as POLineCount

FROM tblpurchaseorderline pol

INNER JOIN tblpurchaseorder po

ON pol.PONumber = po.PONumber

INNER JOIN TotalQuantity tq

ON tq.polineid= pol.polineid

WHERE pol.qtyordered - TotalQtyReceived < 0

GROUP BY po.vendorid;

Table

Description automatically generated

**Question 8:**

CREATE VIEW CountofPOLineOrders as

SELECT po.VendorID, COUNT(pol.polineid) as POLineCount

FROM tblpurchaseorderline pol

INNER JOIN tblpurchaseorder po

ON pol.PONumber = po.PONumber

INNER JOIN TotalQuantity tq

ON tq.polineid= pol.polineid

WHERE pol.qtyordered - TotalQtyReceived < 0

GROUP BY po.vendorid;

SELECT cpo.VendorID, v.VendorName, V.VendorEmail, cpo.POLineCount

FROM CountofPOLineOrders cpo

INNER JOIN tblVendor v

ON cpo.vendorid = v.vendorid

WHERE cpo.POLineCount = (SELECT MAX(POLineCount) FROM CountofPOLineOrders);

Graphical user interface, application, Word

Description automatically generated

**Question 9:**

CREATE VIEW EmpCounts as

SELECT receiveempid, SUM(qtyreceived) as TotalDamagedItemsReceived

FROM tblreceiver r

INNER JOIN tblCondition c

ON r.ConditionID = c.ConditionID

WHERE c.ConditionDescription LIKE '%Damage%'

GROUP BY receiveempid

SELECT DISTINCT r.receiveempid as EmployeeID, e.Emplastname + ', ' + UPPER(LEFT(e.empfirstname, 1)) as EmployeeName,

e.EmpEmail, e.empmgrid as ManagerEmpID,

m.Emplastname + ', ' + UPPER(LEFT(m.empfirstname, 1)) as ManagerName, m.empemail as ManagerEmail,

ec.TotalDamagedItemsReceived

FROM tblReceiver r

INNER JOIN tblEmployee e

ON r.receiveempid = e.empid

INNER JOIN tblemployee m

ON e.empmgrid = m.empid

INNER JOIN EmpCounts ec

ON ec.receiveempid = r.receiveempid

WHERE TotalDamagedItemsReceived = (SELECT MAX(TotalDamagedItemsReceived) FROM EmpCounts);

Graphical user interface, application, Word

Description automatically generated

**Question 10:**

CREATE VIEW CurrentDate as

SELECT p.ProductID, p.ProductDescription,

isnull(SUM(pol.qtyordered), 0) as CurrentQtyonOrder, isnull(MAX(pol.price), 0) as CurrentMaxPrice,

isnull(MIN(pol.price), 0) as CurrentMinPrice, isnull(AVG(pol.price), 0) as CurrentAvgPrice

FROM tblPurchaseOrderLine pol

RIGHT OUTER JOIN tblProduct p

ON pol.ProductID = p.productID

GROUP BY p.ProductID, p.ProductDescription

CREATE VIEW PastDate as

SELECT p.productid, isnull(max(price),0) as PastMaxPrice, isnull(min(price),0) as PastMinPrice,

isnull(avg(price),0) as PastAvgPrice

FROM tblpurchasehistory ph

RIGHT OUTER JOIN tblProduct p

ON ph.ProductID = p.productID

GROUP BY p.productid

SELECT cd.\*, pd.PastMaxPrice,pd.PastMinPrice, pd.PastAvgPrice,

isnull(CONVERT(VARCHAR,ud.recent,107), 'No Previous Purchase') as MostRecentPastPurchaseDate,

isnull(ph.price,0) as MostRecentPastPurchasePrice

FROM CurrentDate cd

FULL JOIN PastDate pd

ON cd.productid = pd.productid

FULL JOIN UpdatedDate ud

ON ud.productid = cd.productID

FULL JOIN tblpurchasehistory ph

on ph.productid = ud.productid

WHERE ph.Datepurchased = UD.Recent or UD.Recent is Null;

Table

Description automatically generated