**IST 278 Unit 2 Lab**

Name: Ramon Rodriguez Date: 5/27/2024

# **Lab Instructions**

1. Type your name and the date in the spaces provided.
2. Use the SQL Server Management Studio and the AP database and the IST278EagleCorp database as specified below.
3. Complete all the 5 exercises in this lab (exercises 1, 2a, 2b, 3, and 4)
4. Upload and submit a completed copy of this lab sheet before the due date.

**Scripting Lab**

# **1. Unit 2 Lab Exercise 1**

Use the IST278EagleCorp database and write a script that declares and sets a variable to be equal to the total inventory stock level.

If this variables value is greater than 4600, the script is to return a result set consisting of PartNumber, PartDescription, and StockLevel sorted by StockLevel in DESC order for each row from the InventoryPart table that has a StockLevel > 55.

If the value in the variable is less or equal to 4600, the script is to return a message that displays the value in the variable and appropriate verbiage. This message is to be formatted as follows:

“Inventory Balance is XXXX and that is not too high. No need to be concerned at this time.”

Hints:

1. Work Page 455 exercise #1 using the AP database before attempting Lab exercise 1. The two exercises are similar.
2. Use the convert function and concatenation (+) to build the Inventory Balance is XXXX and that is not too high message. You can find examples of building the print message with concatenation (+) and the convert function on pages 423, 431, and 439.
   1. If you do not understand what the concatenation (+) is doing refer to page 94 to read about it.
   2. If you do not understand what the convert function is doing refer to page 98 and 255 to read about the convert function.

**---** **Paste below this line the Script you wrote for this exercise --**

Paste here

DECLARE @totalInventory INT;

SELECT @totalInventory = SUM(StockLevel) FROM InventoryPart;

if @totalInventory > 4600

BEGIN

SELECT PartNumber, PartDescription, StockLevel FROM InventoryPart

WHERE StockLevel > 55 ORDER BY StockLevel DESC;

END

ELSE if @totalInventory <= 4600

print 'Inventory balance is' + CAST(@totalInventory AS VARCHAR(50)) + 'and that is not too high';

**---** **Paste below this line the run results from executing the script you wrote--**

Paste here

A screenshot of a computer

Description automatically generated

# **2a. Unit 2 Lab Exercise 2a**

Use the AP database and write a script that uses a derived table to return the date and invoice total of the last (most recent) invoice issued by vendor26, 29, 34, 72, 83, 95, and 123 (these numbers are their VendorID values – only return data associated with these VendorIDs).

Hint: Study the sample script shown with page 455 exercise #2 before attempting Unit 2 Lab exercise 2. The script you are to write for this lab exercise will have a lot in common with that sample.

**--- Paste below this line the Script you wrote for exercise 2a --**

Paste here

SELECT VendorID, InvoiceDate,InvoiceTotal

FROM

(SELECT VendorID, InvoiceDate, InvoiceTotal, ROW\_NUMBER() OVER (PARTITION BY VendorID ORDER BY InvoiceDate DESC) AS RowNum

FROM Invoices WHERE VendorID IN (26, 29, 34, 72, 83, 95, 123)

) AS RecentInvoices

WHERE

RowNum = 1;

**--- Paste below this line the run results from executing the script you wrote--**

Paste here

A screenshot of a computer

Description automatically generated

# **2b. Unit 2 Lab Exercise 2b**

Write a script that generates the same result set as exercise 2a but uses a temporary table in place of the derived table. Make sure your script tests for the existence of any objects it creates.

Hint: Work Page 455 exercise #2 using the AP database before attempting Lab exercise 2b. The two exercises are similar.

**--- Paste below this line the Script you wrote for exercise 2b--**

Paste here

IF OBJECT\_ID('tempdb..#RecentInvoices') IS NOT NULL

DROP TABLE #RecentInvoices

CREATE TABLE #RecentInvoices (

VendorID INT,

InvoiceDate DATE,

InvoiceTotal DECIMAL(10, 2),

RowNum INT

);

INSERT INTO #RecentInvoices (VendorID, InvoiceDate, InvoiceTotal, RowNum)

SELECT VendorID, InvoiceDate, InvoiceTotal, ROW\_NUMBER() OVER (PARTITION BY VendorID ORDER BY InvoiceDate DESC) AS RowNum

FROM Invoices WHERE VendorID IN (26, 29, 34, 72, 83, 95, 123);

SELECT VendorID, InvoiceDate, InvoiceTotal

FROM #RecentInvoices

WHERE RowNum = 1;

**---** **Paste below this line the run results from executing the script you wrote --**

Paste here

A screenshot of a computer

Description automatically generated

# **3. Unit 2 Lab Exercise 3**

Hint: Work Page 455 exercise #3 using the AP database before attempting Lab exercise 3. The two exercises are similar.

For this exercise you are to produce the same result you did for exercise 2, but this time you are to make use of a view. Proceed as follows:

1. Use the AP database and write a query that Selects the VendorID, MAX(InvoiceDate) from the Invoices table. Give the second returned column a name of LastInvoice.
2. After you have the step a query working, write a script that uses this query to create a view name LastInvoice\_V\_XX (where the two XX’s are your initials). Make sure your script tests for the existence of any objects it creates.

**--- Paste below this line the Script you wrote for creating the view --**

Paste here

IF OBJECT\_ID('LastInvoice\_V\_RR', 'V') IS NOT NULL

DROP VIEW LastInvoice\_V\_RR

GO

CREATE VIEW LastInvoice\_V\_RR AS

SELECT VendorID, MAX(InvoiceDate) as LastInvoice FROM Invoices GROUP BY VendorID;

GO

1. Write a script that uses the view you created in step b to generate the same result set as the code you wrote for exercise 2.
2. Execute your script that uses the view (debug and correct as needed).

**--- Paste below this line the Script you wrote that uses the view ---**

Paste here

USE AP;

GO

SELECT VendorID, InvoiceDate, InvoiceTotal

FROM Invoices

WHERE (VendorID IN (26, 29, 34, 72, 83, 95, 123)) AND (InvoiceDate IN (SELECT LastInvoice FROM LastInvoice\_V\_RR ));

**--- Paste below this line the run results from executing the script that uses the view --**

Paste here

A screenshot of a computer screen

Description automatically generated

# **4. Unit 2 Lab Exercises 4**

Write a script that uses dynamic SQL to return a single column that represents the number of rows in the last table in the current database. The script should automatically choose the table that appears last alphabetically, and it should exclude tables named dtproperties and sysdiagrams. Name the column XXCountOfRowsOnTable, where XX is your initials and Table is the chosen table name.

Hints:

1. Use the sys.tables catalog view.
2. Work exercise #4 on page 455 before attempting this exercise. This exercise is similar.

**--- Paste below this line the Script you wrote for exercise 4--**

Paste here

DECLARE @TableName NVARCHAR(200)

DECLARE @SQL NVARCHAR(MAX)

SELECT TOP 1 @TableName = name FROM sys.tables WHERE name NOT IN ('dtproperties', 'sysdiagrams') ORDER BY name DESC

SET @SQL = 'SELECT COUNT(\*) AS RRCountOfRowsOnTable FROM ' + QUOTENAME(@TableName)

EXEC sp\_executesql @SQL

**--- Paste below this line the run results from executing the script you wrote for exercise 4--**

Paste here

A screenshot of a computer

Description automatically generated