**Gateway Technical College**

DATABASES

152-080

Unit 3 Assignment

# Introduction

This assignment will have you go back through how to create statements that JOIN data from multiple tables from our book.  The video lecture for this chapter demonstrated examples of the syntax of the different clauses.   You will now apply what you saw when you create your statements for this lab.

INNER JOIN

OUTER JOIN

FULL JOIN

CROSS JOIN

You will also practice Select statement to verify the above statements worked correctly

In order to do this lab you will need to copy and paste the commands in the SQL file (***create\_ap.sql***) to create a new database called AP that will have tables created and populated.

This lab describes 15 commands you need to create and execute on your local SQL Server in the AP database.  Please make sure to paste the commands you create and execute and paste them into this word document. MAKE SURE they work - before you past them into the document.

Once completed, make sure to attach your completed word document to this assignment for grading.   Each command will be worth 5 points.

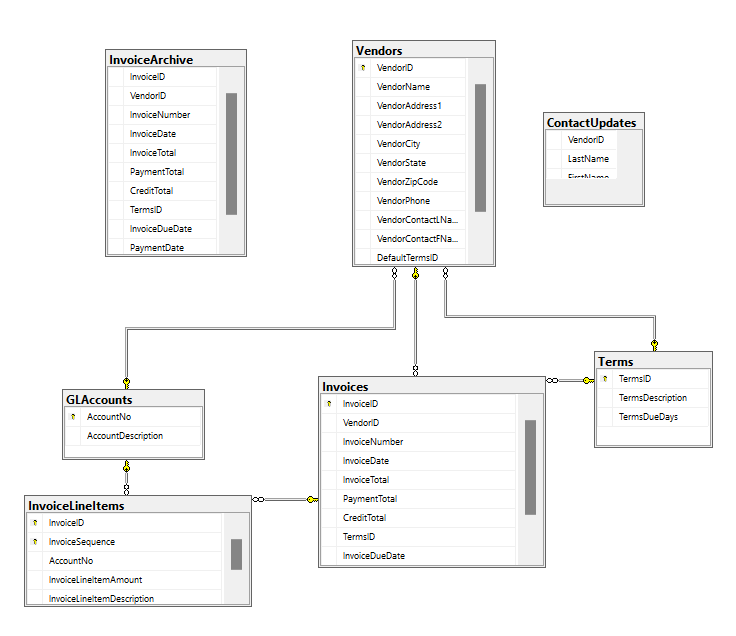
Use the **Discussion Forum** if you have any questions regarding the how to approach this assignment. You can also email your instructor directly for assistance if you have any questions.

Save your submission as ***lastnameFirstname\_assign3.docx*** and submit it in the unit *Apply* section of the course.

# Instructions

In order to do this assignment, you will need to run statements to create another database which has additional tables that we can practice our joins on. **COPY in your working commands into this document after each of the steps – this must be submitted for grading.**

1. Open the attached SQL file and copy and paste the code into a new query window. Another way to do it is to run the file but for now it is easier for you to copy and paste the statements. Make sure to take all of the statements. – Click execute
2. You now should have an entirely new database named AP - for the rest of this assignment – you will use the database instead of AdventureWorks2016
3. Create your database Diagram of your new database - AP and use all tables – it should look similar to the one below. (NOTE: you will have to right click on the database, properties, files and change the owner of the files – as done in the relationship assignment). **REPLACE the following diagram with your diagram below!**



1. To verify your tables have been populated correctly, count the records in each of the following tables:

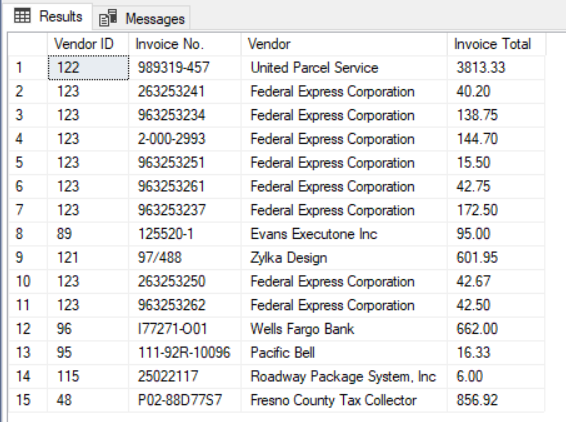
|  |  |
| --- | --- |
| Table | Records |
| ContactUpdates | 8 |
| GLAccounts | 75 |
| InvoiceArchive | 0 |
| InvoiceLineItems | 118 |
| Invoices | 114 |
| Terms | 5 |
| Vendors | 122 |

1. Write an INNER JOIN statement. Join the Invoices table to the Vendors table. Output the following fields (make sure to use column aliases as shown below).

**Note**: You don’t have to show all records but just the first 15 records should suffice. You can screenshot the top 15 records as shown below. Alternatively, you can use the “TOP(15)” command to limit the result. The following example will return up to 15 records only.  
  
SELECT TOP(15) \* FROM Vendors

**YOUR COMMAND WAS:**

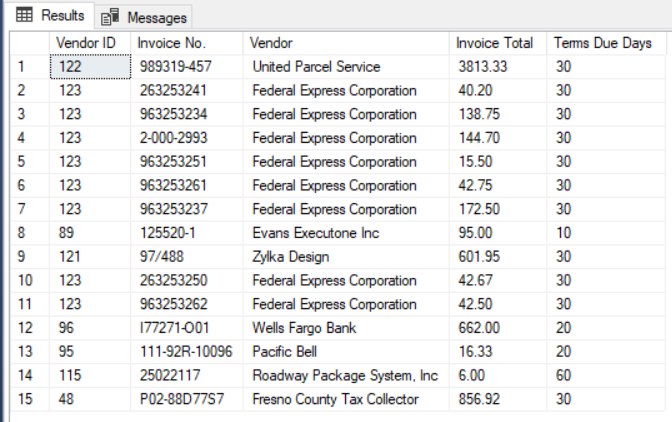
|  |
| --- |
| SELECT i.VendorID, i.InvoiceNumber as 'Invoice N0.' , v.VendorName as 'Vendor', i.InvoiceTotal as 'Invoice Total'  FROM Invoices as i  INNER JOIN Vendors as v  ON i.VendorID = v.VendorID |



1. Modify the above query to also INNER JOIN the Terms and the Invoices table to display one additional field Terms Due Days:

**YOUR COMMAND WAS:**

|  |
| --- |
| SELECT i.VendorID, i.InvoiceNumber as 'Invoice N0.' , v.VendorName as 'Vendor', i.InvoiceTotal as 'Invoice Total', t.TermsDueDays as 'Terms Due Days'  FROM Invoices as i  INNER JOIN Vendors as v  ON i.VendorID = v.VendorID  INNER JOIN Terms as t  ON i.TermsID = t.TermsID |



1. Write an INNER JOIN statement. Join the GLAccounts table to the Vendors table. Output the following fields (make sure to use column aliases as shown below).

**YOUR COMMAND WAS:**

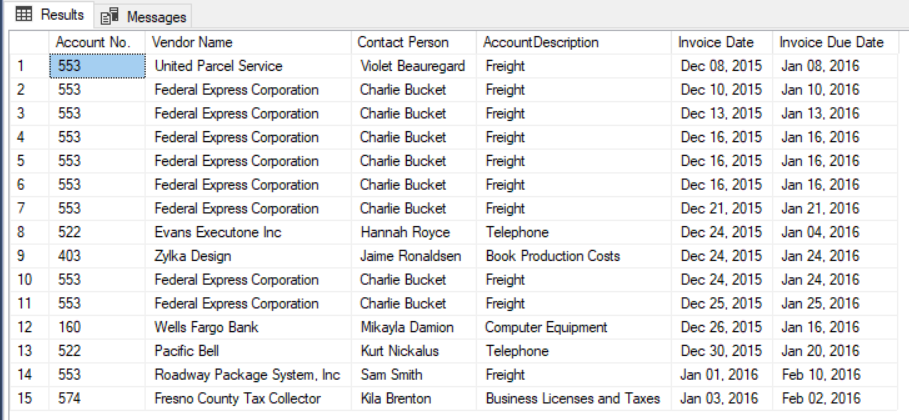
|  |
| --- |
| SELECT g.AccountNo, v.VendorName, v.VendorContactFName + ' ' + v.VendorContactLName as 'Contact Person', g.AccountDescription  FROM Vendors as v  INNER JOIN GLAccounts as g  ON v.DefaultAccountNo = g.AccountNo |



1. Modify the above query to also INNER JOIN the Vendors table to the Invoices table. Modify the select clause to output the following columns. Use the following command to convert the dates:  
     
   convert(varchar(12), InvoiceDueDate,107)  
     
   will convert: 2016-01-08 00:00:00 🡪 Jan 08, 2016

**YOUR COMMAND WAS:**

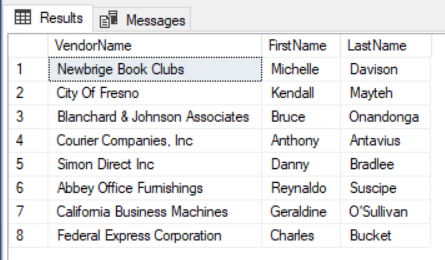
|  |
| --- |
| SELECT g.AccountNo, v.VendorName, v.VendorContactFName + ' ' + v.VendorContactLName as 'Contact Person', g.AccountDescription,  convert(varchar(12), InvoiceDueDate,107) as 'Invoice Date', convert(varchar(12), PaymentDate,107) as 'Invoice Due Date'  FROM Vendors as v  INNER JOIN GLAccounts as g  ON v.DefaultAccountNo = g.AccountNo  INNER JOIN Invoices as i  ON v.VendorID = i.VendorID |



1. Write an INNER JOIN statement. Join the Vendors table to the ContactUpdates table. Output the following fields (make sure to use column aliases as shown below).

**YOUR COMMAND WAS:**

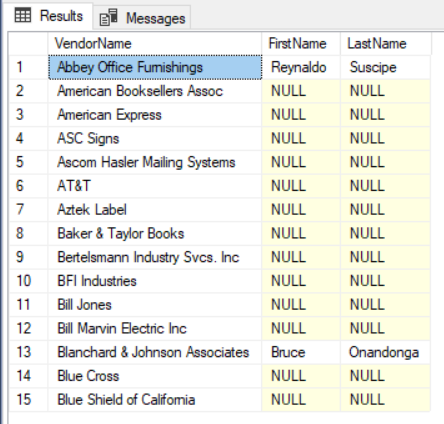
|  |
| --- |
| SELECT v.VendorName, c.FirstName, c.LastName  FROM Vendors as v  INNER JOIN ContactUpdates as c  ON v.VendorID = c.VendorID |



1. Write a LEFT OUTER JOIN statement. Join the Vendors table to the ContactUpdates table. Output the following fields (make sure to use column aliases as shown below). This shows also Vendors who do not have any contact updates.

**YOUR COMMAND WAS:**

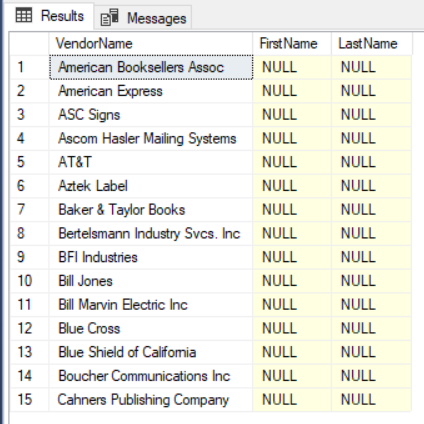
|  |
| --- |
| SELECT v.VendorName, c.FirstName, c.LastName  FROM Vendors as v  LEFT OUTER JOIN ContactUpdates as c  ON v.VendorID = c.VendorID |



1. Modify the above statement to only show the Vendors who do not have any contact updates in the ContactUpdates table

**YOUR COMMAND WAS:**

|  |
| --- |
| SELECT v.VendorName, c.FirstName, c.LastName  FROM Vendors as v  LEFT OUTER JOIN ContactUpdates as c  ON v.VendorID = c.VendorID  WHERE FirstName IS NULL |



1. Add the following contact person into the ContactUpdates table

Last Name: JOHNSON  
First Name: SHAYNA

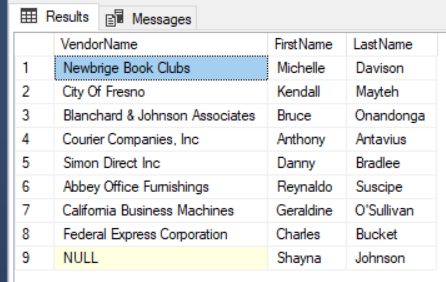
**YOUR COMMAND WAS:**

|  |
| --- |
| SELECT \*  FROM ContactUpdates  INSERT INTO ContactUpdates (LastName, FirstName)  VALUES  ('Johnson', 'Shayna') |

1. Write a RIGHT OUTER JOIN statement. Join the ContactUpdates table to the Vendors table. Output the following fields (make sure to use column aliases as shown below). This should include any contact person who are not assigned to any vendors.

**YOUR COMMAND WAS:**

|  |
| --- |
| SELECT v.VendorName, c.FirstName, c.LastName  FROM Vendors as v  RIGHT OUTER JOIN ContactUpdates as c  ON v.VendorID = c.VendorID |



1. Create a Cartesian product between the rows in the Vendors and Invoices tables. How many rows did you end up with and why?  
     
   **YOUR COMMAND WAS:**

|  |
| --- |
| SELECT \*  FROM Vendors  CROSS JOIN Invoices |

1. Insert the following data into the InvoiceArchive table. Just copy, paste and execute.  
     
   INSERT INTO InvoiceArchive  
   VALUES(1,122,'989319-457','2015-12-08 00:00:00',3813.33,3813.33,0.00,3,'2016-01-08 00:00:00','2016-01-07 00:00:00'),  
   (11,123,'963253262','2015-12-25 00:00:00',42.50,42.50,0.00,3,'2016-01-25 00:00:00','2016-01-20 00:00:00')  
     
     
   Create a FULL JOIN between the ContactUpdates table and the InvoiceArchive table (as shown below)

**YOUR COMMAND WAS:**

|  |
| --- |
| SELECT \*  FROM InvoiceArchive as i  FULL JOIN ContactUpdates as c on i.VendorID = c.VendorID |

