# SE 133 – Database Management

## Lab 1 – Create SQL statements to select very specific records from the AP database

Description:

**Provide SQL Statements for each of the following problems along with screen shots of the results.**

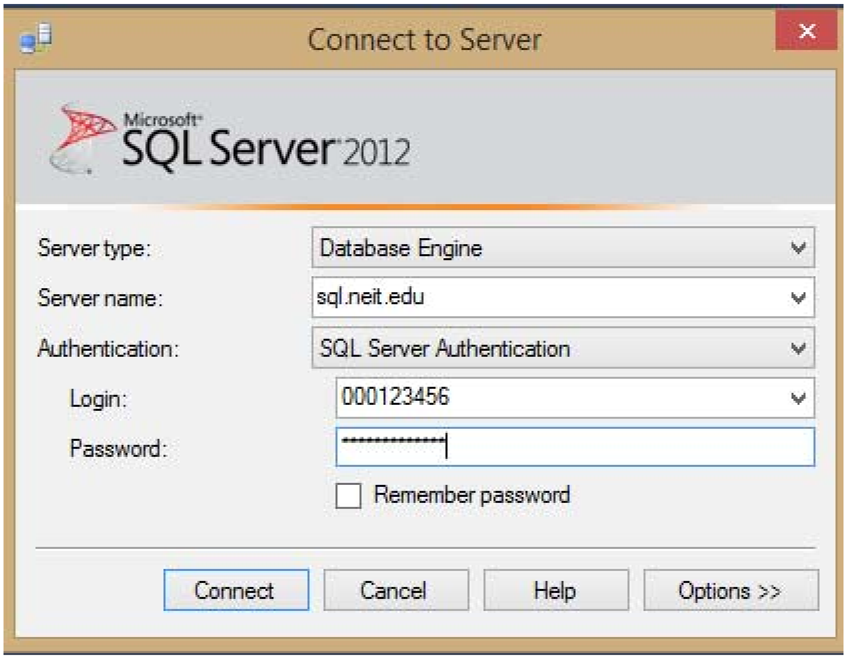
**Then create a web page lab1.html** on ict.neit.edu and submit the link to this page. Or, you can create a PDF file and link to it.

## Lab 1.1

**Go to Start > Programs > Microsoft SQL Server 2012 > SQL Server Management Studio**

**Remember that your credentials to sql.neit.edu are the same as your canvas credentials.** Here is an example of the login screen:

**NOTE: Use Windows Authentication to login.**



Connect to Server

Server type: Database Engine

Server name: sql.neit.edu

Authentication: SQL Server Authentication

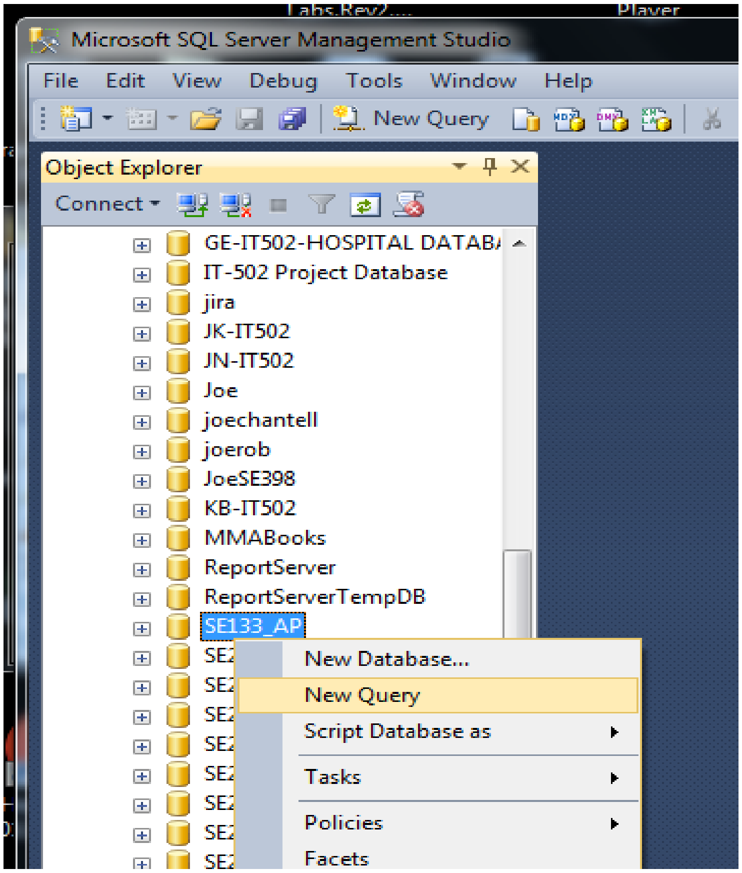
Login: 000123456

Password: **************

□ Remember password

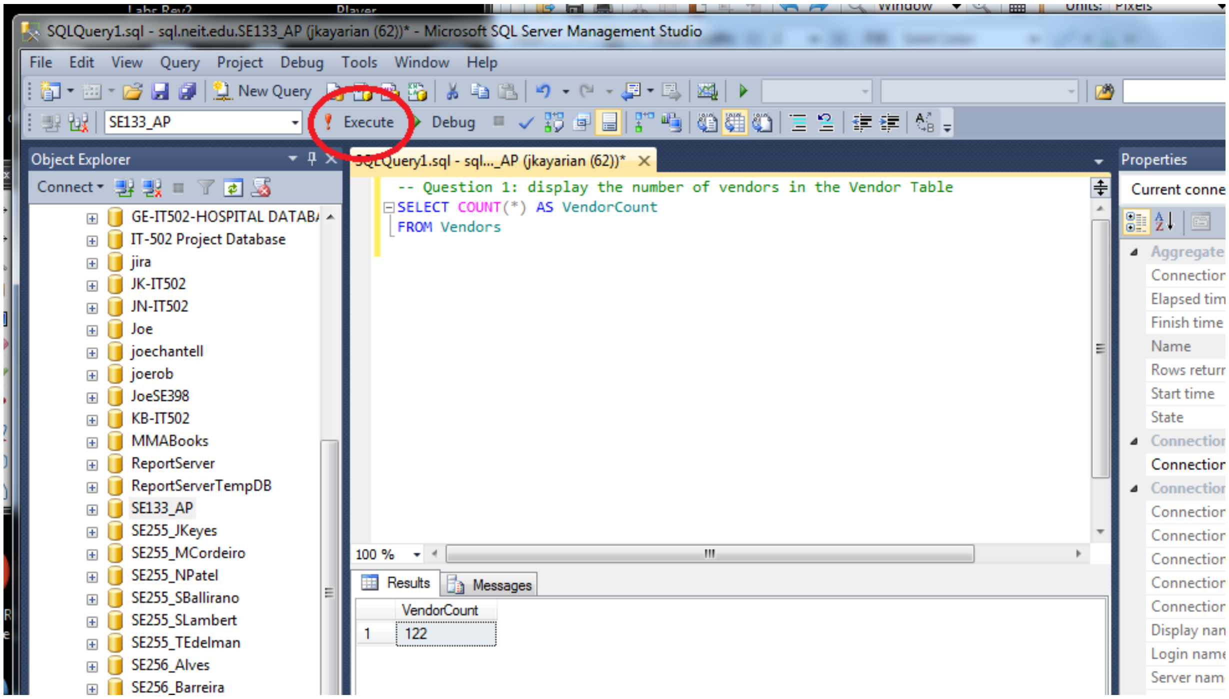
You will see a bunch of databases on your left. Don’t worry about any of them except **SE133_AP**.

**Right-Click the SE133_AP database folder and select New Query**. See the screen shot below.



**Selecting New Query will produce the Query Editor** in the middle frame of the SQL Management Studio (Shown below). Type this SQL code into the window and the **press the 'Execute' button or press F5 to run the SQL code.**

```
-- Question 1: display the number of vendors in the Vendor Table
SELECT COUNT(*) AS VendorCount
FROM Vendors
```



```
-- Question 1: display the number of vendors in the Vendor Table
SELECT COUNT(*) AS VendorCount
FROM Vendors
```

```
   VendorCount
1  122
```

Execute All of your lab queries within that query window. So, if the first question were to display the number of vendors in the Vendor Table, you would provide the text of the question in a comment and then the actual SQL Query as shown above.

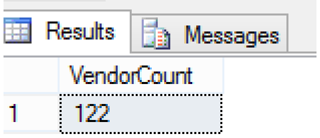
**Also provide the SQL code and result screen shot as shown in the example below.**

**THIS IS WHAT YOUR LAB RESULTS SHOULD LOOK LIKE**

**SQL Code:**

```
-- Question 1: display the number of vendors in the Vendor Table
SELECT COUNT(*) AS VendorCount
FROM Vendors
```

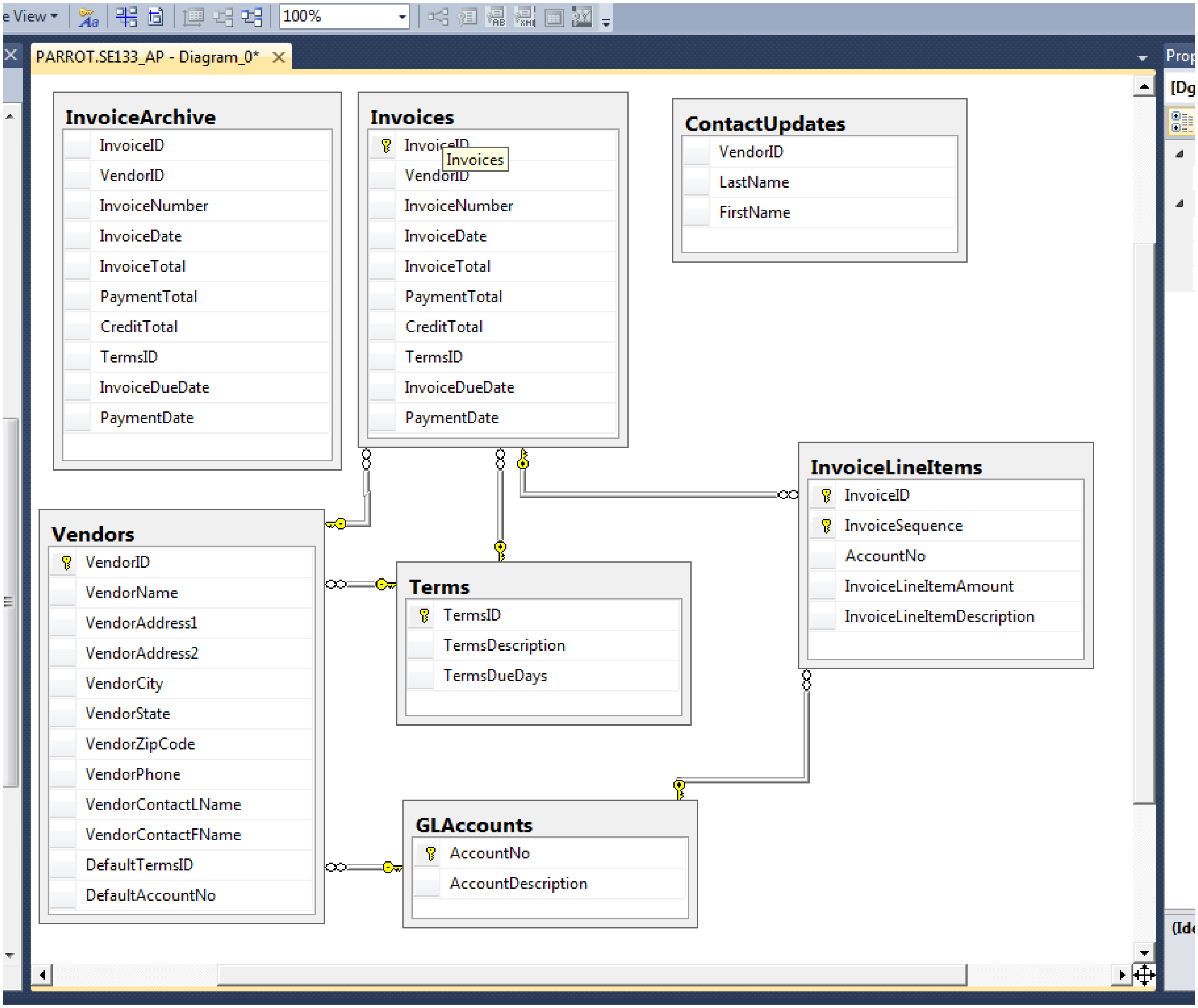
**Screenshot:**



```
   VendorCount
1  122
```

The ‘Print Screen’ or ‘Snipping Tool’ will become your friend real soon if it isn’t your friend already.

**Below is the Database Diagram of the SE133_AP database for your reference of Table names, fields, and Indexes.**



PARROT.SE133_AP - Diagram_0*

**InvoiceArchive**
- InvoiceID
- VendorID
- InvoiceNumber
- InvoiceDate
- InvoiceTotal
- PaymentTotal
- CreditTotal
- TermsID
- InvoiceDueDate
- PaymentDate

**Invoices**
- InvoiceID
- VendorID
- InvoiceNumber
- InvoiceDate
- InvoiceTotal
- PaymentTotal
- CreditTotal
- TermsID
- InvoiceDueDate
- PaymentDate

**ContactUpdates**
- VendorID
- LastName
- FirstName

**InvoiceLineItems**
- InvoiceID
- InvoiceSequence
- AccountNo
- InvoiceLineItemAmount
- InvoiceLineItemDescription

**Vendors**
- VendorID
- VendorName
- VendorAddress1
- VendorAddress2
- VendorCity
- VendorState
- VendorZipCode
- VendorPhone
- VendorContactLName
- VendorContactFName
- DefaultTermsID
- DefaultAccountNo

**Terms**
- TermsID
- TermsDescription
- TermsDueDays

**GLAccounts**
- AccountNo
- AccountDescription

# Lab 1.2

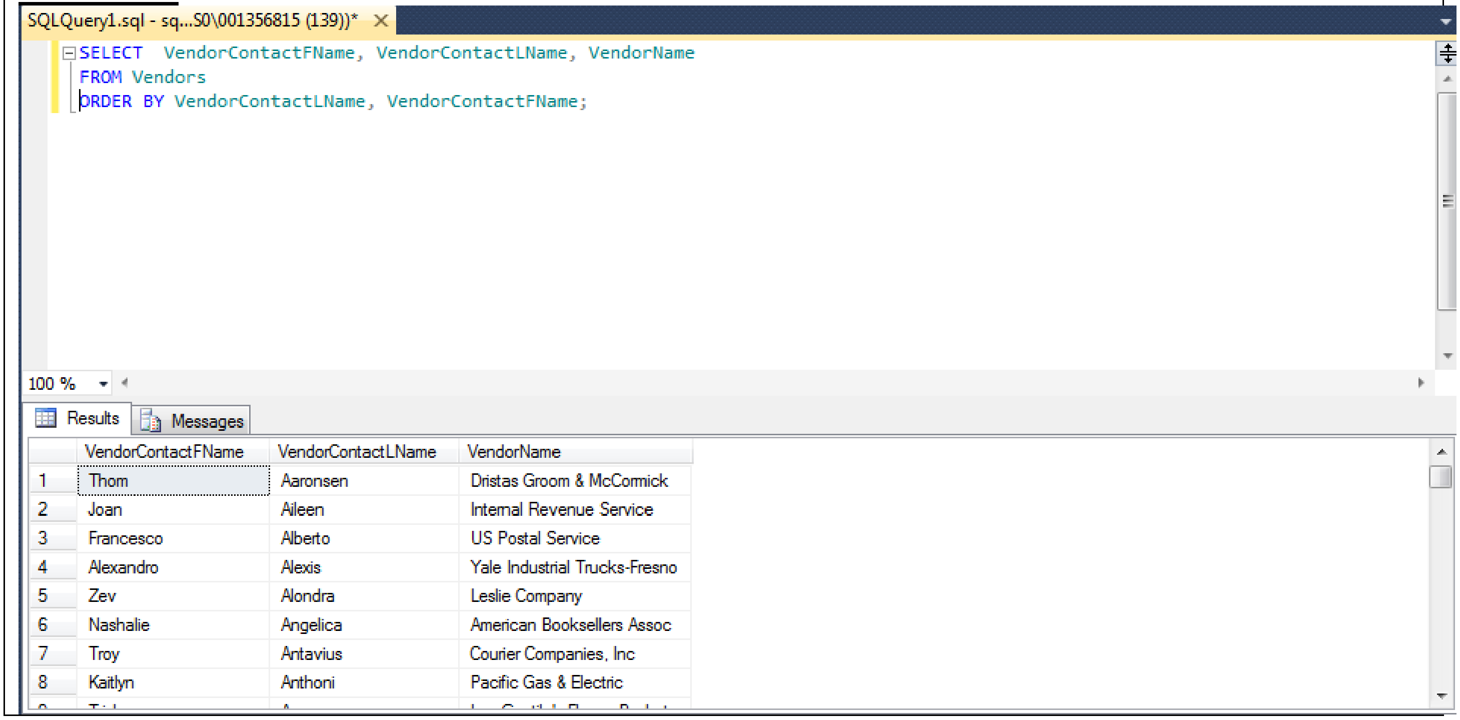
So without further ado, here are the questions: Questions 1 through 7 are the ones from page 123 in the text.

1. Write a SELECT statement that returns three columns from the Vendors table: VendorContactFName, VendorContactLName, and VendorName. Sort the result set by last name, then by first name. (Text:88-89, Ch3_Slide:6)

**SQL Code:**

```
SELECT  VendorContactFName, VendorContactLName, VendorName
FROM Vendors
ORDER BY VendorContactLName, VendorContactFName;
```

**Screenshot:**



```
SELECT  VendorContactFName, VendorContactLName, VendorName
FROM Vendors
ORDER BY VendorContactLName, VendorContactFName;
```

| | VendorContactFName | VendorContactLName | VendorName |
|---|---|---|---|
| 1 | Thom | Aaronsen | Dristas Groom & McCormick |
| 2 | Joan | Aileen | Internal Revenue Service |
| 3 | Francesco | Alberto | US Postal Service |
| 4 | Alexandro | Alexis | Yale Industrial Trucks-Fresno |
| 5 | Zev | Alondra | Leslie Company |
| 6 | Nashalie | Angelica | American Booksellers Assoc |
| 7 | Troy | Antavius | Courier Companies, Inc |
| 8 | Kaitlyn | Anthoni | Pacific Gas & Electric |

2. Write a SELECT statement that returns four columns from the Invoices table, named Number, Total, Credits, and Balance: (Text:89,93, Ch3_Slide:12-13)

Number Column alias for the InvoiceNumber column

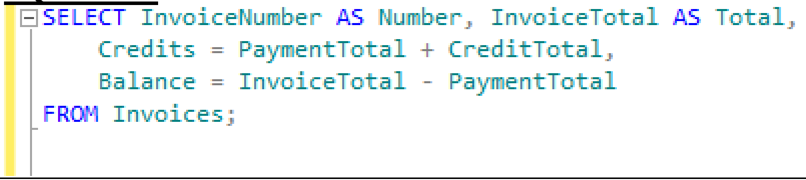
Total Column alias for the InvoiceTotal column

Credits Sum of the PaymentTotal and CreditTotal columns

Balance InvoiceTotal minus the sum of PaymentTotal and CreditTotal

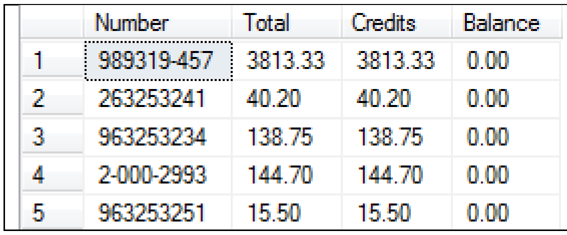
Use the AS keyword to assign column aliases.

**SQL Code:**



```
SELECT InvoiceNumber AS Number, InvoiceTotal AS Total,
       Credits = PaymentTotal + CreditTotal,
       Balance = InvoiceTotal - PaymentTotal
FROM Invoices;
```

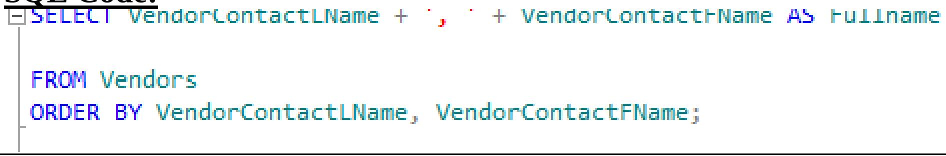
**Screenshot**



|  | Number | Total | Credits | Balance |
|---|---|---|---|---|
| 1 | 989319-457 | 3813.33 | 3813.33 | 0.00 |
| 2 | 263253241 | 40.20 | 40.20 | 0.00 |
| 3 | 963253234 | 138.75 | 138.75 | 0.00 |
| 4 | 2-000-2993 | 144.70 | 144.70 | 0.00 |
| 5 | 963253251 | 15.50 | 15.50 | 0.00 |

3. Write a SELECT statement that returns **one column** from the Vendors table named Full Name. Create this column from the VendorContactFName and VendorContactLName columns. Format it as follows: last name, comma, first name (for example, “Doe, John”). Sort the result set by last name, then by first name. (Text:91, Ch3_Slide12)

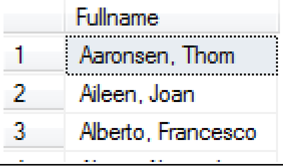
**SQL Code:**



```
SELECT VendorContactLName + ', ' + VendorContactFName AS Fullname

FROM Vendors
ORDER BY VendorContactLName, VendorContactFName;
```

**Screenshot**



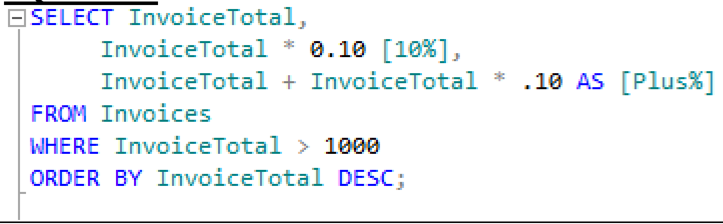
|  | Fullname |
|---|---|
| 1 | Aaronsen, Thom |
| 2 | Aileen, Joan |
| 3 | Alberto, Francesco |

4. Write a SELECT statement that returns three columns:

| | |
|---|---|
| **InvoiceTotal** | From the Invoices table |
| **10%** | 10% of the value of InvoiceTotal |
| **Plus 10%** | The value of InvoiceTotal plus 10% |

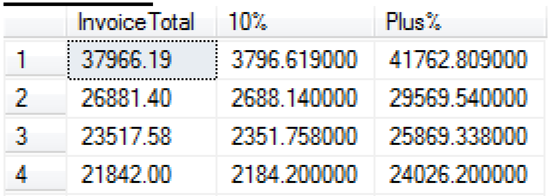
(For example, if InvoiceTotal is 100.0000, 10% is 10.0000, and Plus 10% is 110.0000.) Only return those rows with a balance due greater than 1000. Sort the result set by InvoiceTotal, with the largest invoice first. (Text 97, Slide18-21)

**SQL Code:**



```
SELECT InvoiceTotal,
     InvoiceTotal * 0.10 [10%],
     InvoiceTotal + InvoiceTotal * .10 AS [Plus%]
FROM Invoices
WHERE InvoiceTotal > 1000
ORDER BY InvoiceTotal DESC;
```

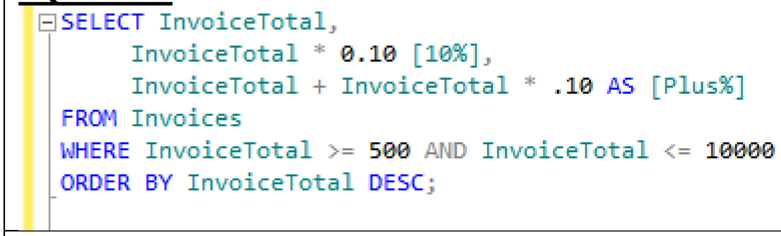
**Screenshot**



|  | InvoiceTotal | 10% | Plus% |
|---|---|---|---|
| 1 | 37966.19 | 3796.619000 | 41762.809000 |
| 2 | 26881.40 | 2688.140000 | 29569.540000 |
| 3 | 23517.58 | 2351.758000 | 25869.338000 |
| 4 | 21842.00 | 2184.200000 | 24026.200000 |

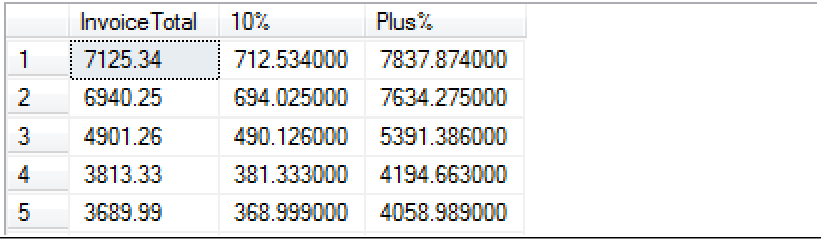
5. Modify the solution to exercise 2 to filter for invoices with an InvoiceTotal that's greater than or equal to $500 but less than or equal to $10,000. (Text:97,105,107,111, Ch3_Slides 31-37)

**SQL Code:**



```
SELECT InvoiceTotal,
       InvoiceTotal * 0.10 [10%],
       InvoiceTotal + InvoiceTotal * .10 AS [Plus%]
FROM Invoices
WHERE InvoiceTotal >= 500 AND InvoiceTotal <= 10000
ORDER BY InvoiceTotal DESC;
```

**Screenshot**



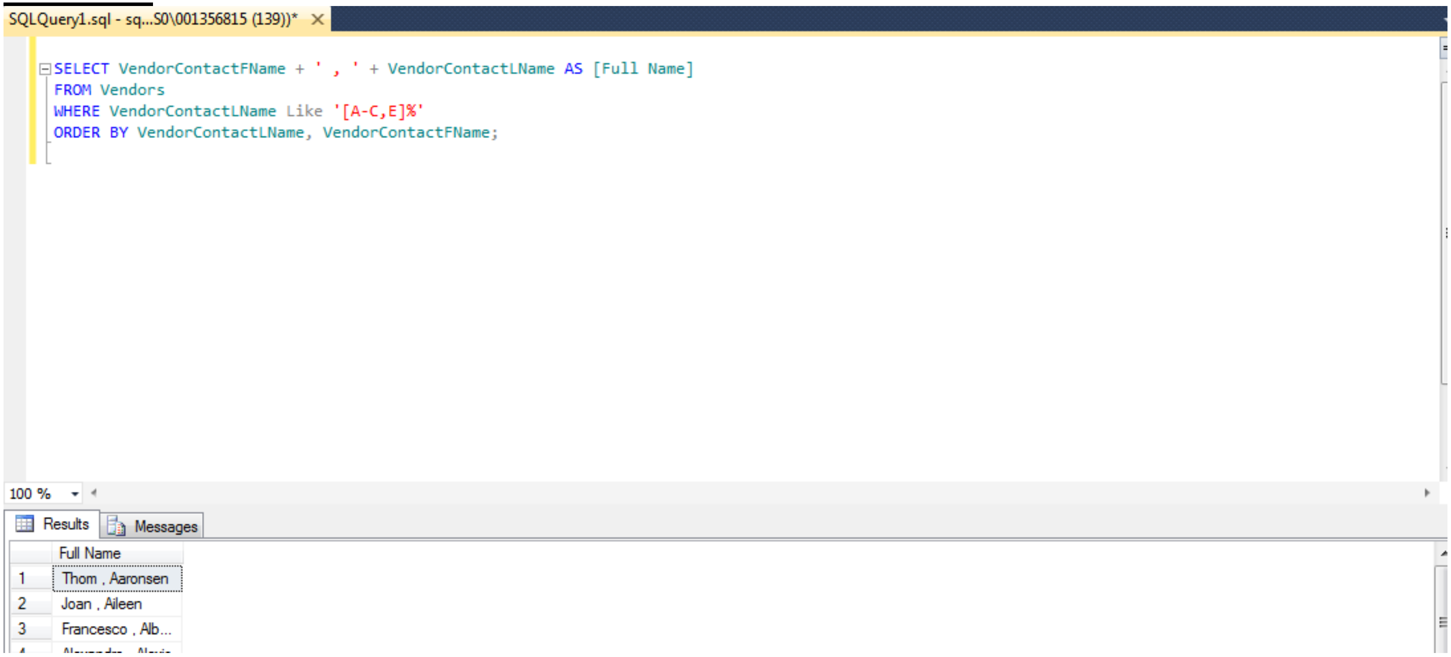
| | InvoiceTotal | 10% | Plus% |
|---|---|---|---|
| 1 | 7125.34 | 712.534000 | 7837.874000 |
| 2 | 6940.25 | 694.025000 | 7634.275000 |
| 3 | 4901.26 | 490.126000 | 5391.386000 |
| 4 | 3813.33 | 381.333000 | 4194.663000 |
| 5 | 3689.99 | 368.999000 | 4058.989000 |

6. Modify the solution to exercise 3 to filter for contacts whose last name begins with the letter A, B, C, or E. (Text 113, Ch3_Slides 39-41)

**SQL Code:**

SELECT VendorContactFName + ' , ' + VendorContactLName AS [Full Name]
FROM Vendors
Where VendorContactLName Like '[A-C,E]%'
ORDER BY VendorContactName, VendorContactFName;

**Screenshot**

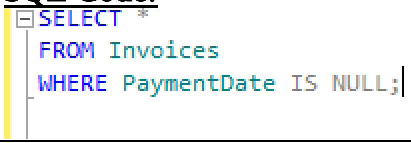


```
SELECT VendorContactFName + ' , ' + VendorContactLName AS [Full Name]
FROM Vendors
WHERE VendorContactLName Like '[A-C,E]%'
ORDER BY VendorContactLName, VendorContactFName;
```

| | Full Name |
|---|---|
| 1 | Thom , Aaronsen |
| 2 | Joan , Aileen |
| 3 | Francesco , Alb... |
| 4 | Alexandra , Alexis |

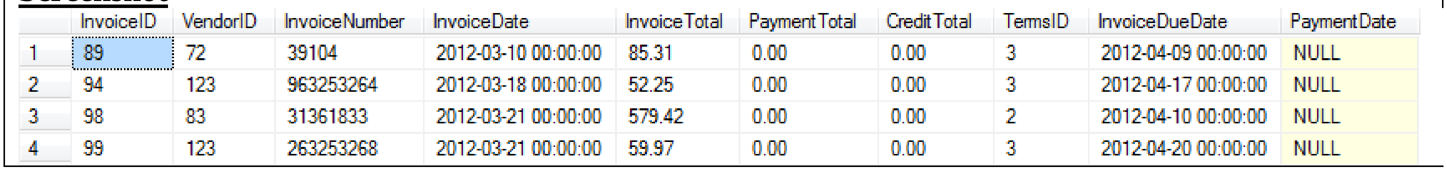
7. Write a SELECT statement that determines whether the PaymentDate column of the Invoices table has any invalid values. To be valid, PaymentDate must be a null value if there's a balance due and a non-null value if there's no balance due. Code a compound condition in the WHERE clause that tests for these conditions. (Text115, Ch3_Slides 42-44)

**SQL Code:**



```
SELECT *
FROM Invoices
WHERE PaymentDate IS NULL;
```

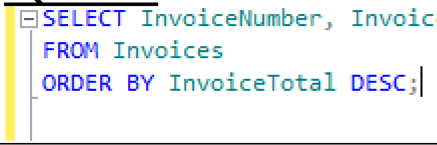
**Screenshot**



| | InvoiceID | VendorID | InvoiceNumber | InvoiceDate | InvoiceTotal | PaymentTotal | CreditTotal | TermsID | InvoiceDueDate | PaymentDate |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 89 | 72 | 39104 | 2012-03-10 00:00:00 | 85.31 | 0.00 | 0.00 | 3 | 2012-04-09 00:00:00 | NULL |
| 2 | 94 | 123 | 963253264 | 2012-03-18 00:00:00 | 52.25 | 0.00 | 0.00 | 3 | 2012-04-17 00:00:00 | NULL |
| 3 | 98 | 83 | 31361833 | 2012-03-21 00:00:00 | 579.42 | 0.00 | 0.00 | 2 | 2012-04-10 00:00:00 | NULL |
| 4 | 99 | 123 | 263253268 | 2012-03-21 00:00:00 | 59.97 | 0.00 | 0.00 | 3 | 2012-04-20 00:00:00 | NULL |

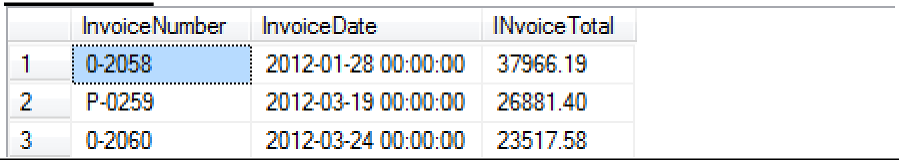
8. Create a SELECT statement that returns the InvoiceNumber, InvoiceDate and InvoiceTotal from the Invoices table. Display the invoices with the highest amounts first. (Text:88-89,117, Ch3_Slide:6,47)

**SQL Code:**



```
SELECT InvoiceNumber, InvoiceDate, INvoiceTotal
FROM Invoices
ORDER BY InvoiceTotal DESC;
```

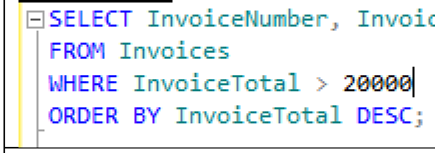
**Screenshot**



| | InvoiceNumber | InvoiceDate | INvoiceTotal |
|---|---|---|---|
| 1 | 0-2058 | 2012-01-28 00:00:00 | 37966.19 |
| 2 | P-0259 | 2012-03-19 00:00:00 | 26881.40 |
| 3 | 0-2060 | 2012-03-24 00:00:00 | 23517.58 |

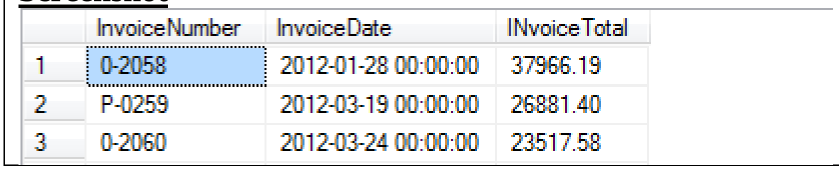
9. Same as (8) but only display invoices that are more than $20,000 (Text:88-89,105,117, Ch3_Slide:6,31,47)

**SQL Code:**



```
SELECT InvoiceNumber, InvoiceDate, INvoiceTotal
FROM Invoices
WHERE InvoiceTotal > 20000
ORDER BY InvoiceTotal DESC;
```

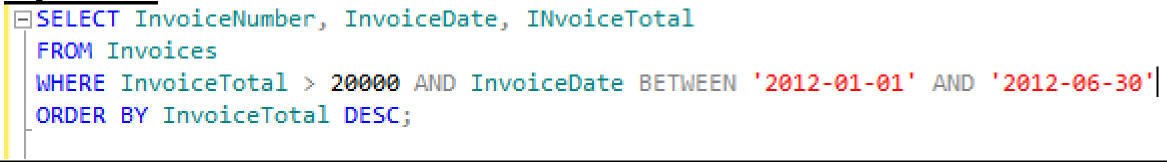
**Screenshot**



| | InvoiceNumber | InvoiceDate | INvoiceTotal |
|---|---|---|---|
| 1 | 0-2058 | 2012-01-28 00:00:00 | 37966.19 |
| 2 | P-0259 | 2012-03-19 00:00:00 | 26881.40 |
| 3 | 0-2060 | 2012-03-24 00:00:00 | 23517.58 |

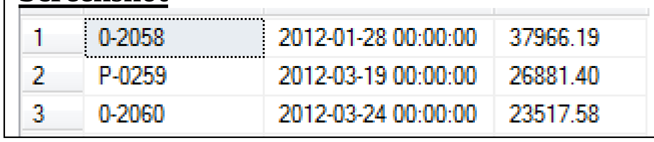
10. Same as (8) but only display invoices in the first six months of 2012 (Text:105,117, Ch3_Slide:6,31,47)

**SQL Code:**



```
SELECT InvoiceNumber, InvoiceDate, INvoiceTotal
FROM Invoices
WHERE InvoiceTotal > 20000 AND InvoiceDate BETWEEN '2012-01-01' AND '2012-06-30'
ORDER BY InvoiceTotal DESC;
```

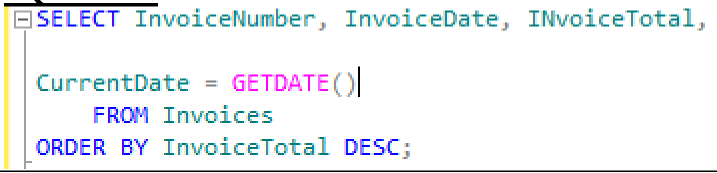
**Screenshot**



| | | | |
|---|---|---|---|
| 1 | 0-2058 | 2012-01-28 00:00:00 | 37966.19 |
| 2 | P-0259 | 2012-03-19 00:00:00 | 26881.40 |
| 3 | 0-2060 | 2012-03-24 00:00:00 | 23517.58 |

11. Same as (10) but include the Current Date as a separate column. (Text:110-111, Ch3 Slides 37)

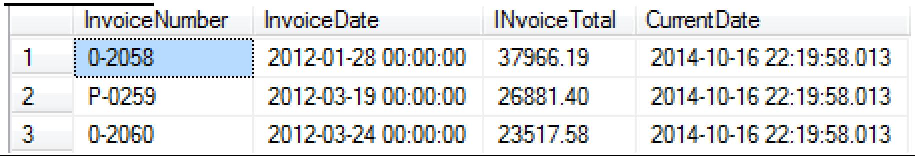
**SQL Code:**



```
SELECT InvoiceNumber, InvoiceDate, INvoiceTotal,

CurrentDate = GETDATE()
    FROM Invoices
ORDER BY InvoiceTotal DESC;
```

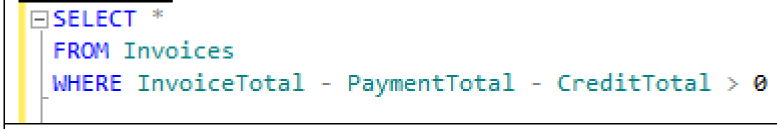
**Screenshot**



| | InvoiceNumber | InvoiceDate | INvoiceTotal | CurrentDate |
|---|---|---|---|---|
| 1 | 0-2058 | 2012-01-28 00:00:00 | 37966.19 | 2014-10-16 22:19:58.013 |
| 2 | P-0259 | 2012-03-19 00:00:00 | 26881.40 | 2014-10-16 22:19:58.013 |
| 3 | 0-2060 | 2012-03-24 00:00:00 | 23517.58 | 2014-10-16 22:19:58.013 |

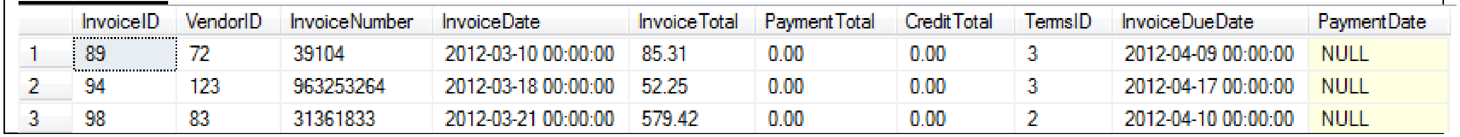
12. List all invoices that haven't been completely paid (Text 97, 105 Ch3_Slides 18, 31)

**SQL Code:**



```
SELECT *
FROM Invoices
WHERE InvoiceTotal - PaymentTotal - CreditTotal > 0
```

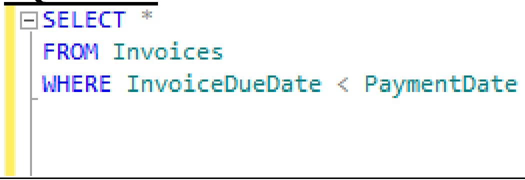
**Screenshot**



| | InvoiceID | VendorID | InvoiceNumber | InvoiceDate | InvoiceTotal | PaymentTotal | CreditTotal | TermsID | InvoiceDueDate | PaymentDate |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 89 | 72 | 39104 | 2012-03-10 00:00:00 | 85.31 | 0.00 | 0.00 | 3 | 2012-04-09 00:00:00 | NULL |
| 2 | 94 | 123 | 963253264 | 2012-03-18 00:00:00 | 52.25 | 0.00 | 0.00 | 3 | 2012-04-17 00:00:00 | NULL |
| 3 | 98 | 83 | 31361833 | 2012-03-21 00:00:00 | 579.42 | 0.00 | 0.00 | 2 | 2012-04-10 00:00:00 | NULL |

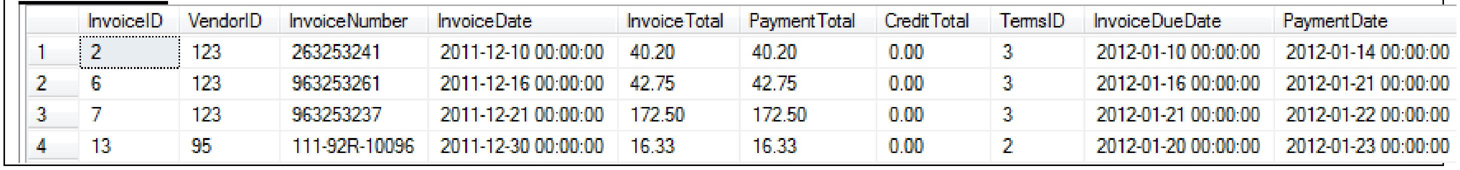
13. List all Invoices that were paid after the due date. List the worst offenders first (Text 98)

**SQL Code:**



```
SELECT *
FROM Invoices
WHERE InvoiceDueDate < PaymentDate
```

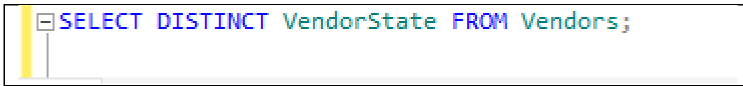
**Screenshot**



| | InvoiceID | VendorID | InvoiceNumber | InvoiceDate | InvoiceTotal | PaymentTotal | CreditTotal | TermsID | InvoiceDueDate | PaymentDate |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 123 | 263253241 | 2011-12-10 00:00:00 | 40.20 | 40.20 | 0.00 | 3 | 2012-01-10 00:00:00 | 2012-01-14 00:00:00 |
| 2 | 6 | 123 | 963253261 | 2011-12-16 00:00:00 | 42.75 | 42.75 | 0.00 | 3 | 2012-01-16 00:00:00 | 2012-01-21 00:00:00 |
| 3 | 7 | 123 | 963253237 | 2011-12-21 00:00:00 | 172.50 | 172.50 | 0.00 | 3 | 2012-01-21 00:00:00 | 2012-01-22 00:00:00 |
| 4 | 13 | 95 | 111-92R-10096 | 2011-12-30 00:00:00 | 16.33 | 16.33 | 0.00 | 2 | 2012-01-20 00:00:00 | 2012-01-23 00:00:00 |

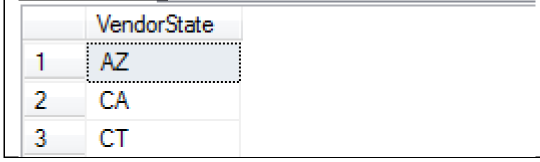
14. Select all the different states without from the Vendors table. Be sure to eliminate duplicates and sort by state (Text:101, Ch3_Slides:27)

**SQL Code:**



```sql
SELECT DISTINCT VendorState FROM Vendors;
```

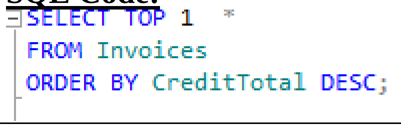
**Screenshot**



| | VendorState |
|---|---|
| 1 | AZ |
| 2 | CA |
| 3 | CT |

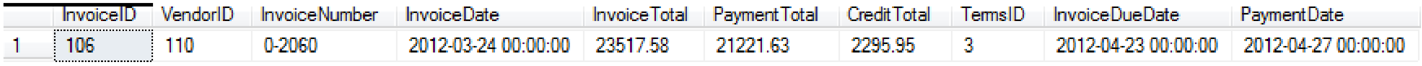
15. Display the invoice with the highest credit total (Text 102)

**SQL Code:**



```sql
SELECT TOP 1 *
FROM Invoices
ORDER BY CreditTotal DESC;
```

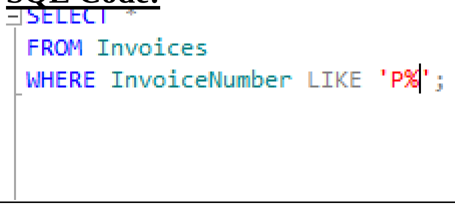
**Screenshot**



| | InvoiceID | VendorID | InvoiceNumber | InvoiceDate | InvoiceTotal | PaymentTotal | CreditTotal | TermsID | InvoiceDueDate | PaymentDate |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 106 | 110 | 0-2060 | 2012-03-24 00:00:00 | 23517.58 | 21221.63 | 2295.95 | 3 | 2012-04-23 00:00:00 | 2012-04-27 00:00:00 |

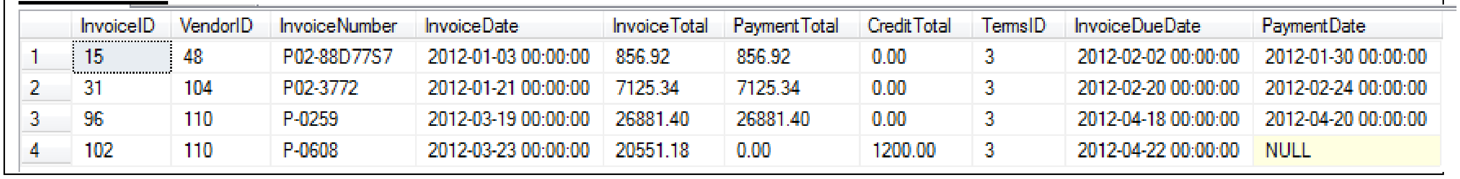
16. Display all Invoices with numbers starting with a P (Text113, Ch3_Slides39-41)

**SQL Code:**



```sql
SELECT *
FROM Invoices
WHERE InvoiceNumber LIKE 'P%';
```

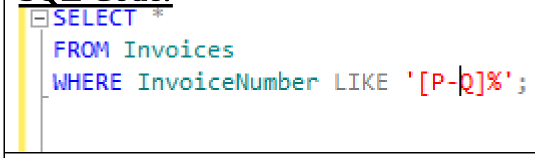
**Screenshot**



| | InvoiceID | VendorID | InvoiceNumber | InvoiceDate | InvoiceTotal | PaymentTotal | CreditTotal | TermsID | InvoiceDueDate | PaymentDate |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 15 | 48 | P02-88D77S7 | 2012-01-03 00:00:00 | 856.92 | 856.92 | 0.00 | 3 | 2012-02-02 00:00:00 | 2012-01-30 00:00:00 |
| 2 | 31 | 104 | P02-3772 | 2012-01-21 00:00:00 | 7125.34 | 7125.34 | 0.00 | 3 | 2012-02-20 00:00:00 | 2012-02-24 00:00:00 |
| 3 | 96 | 110 | P-0259 | 2012-03-19 00:00:00 | 26881.40 | 26881.40 | 0.00 | 3 | 2012-04-18 00:00:00 | 2012-04-20 00:00:00 |
| 4 | 102 | 110 | P-0608 | 2012-03-23 00:00:00 | 20551.18 | 0.00 | 1200.00 | 3 | 2012-04-22 00:00:00 | NULL |

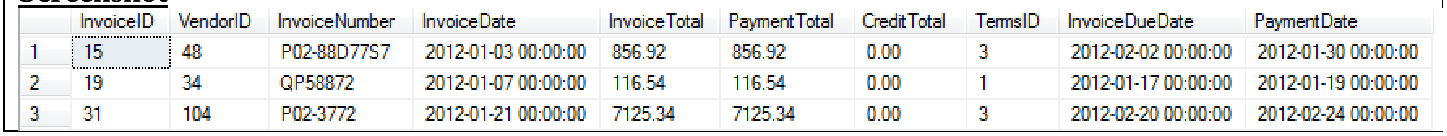
17. Display all invoices with invoice numbers starting with a P or a Q (Text113, Ch3_Slides39-41)

**SQL Code:**



```sql
SELECT *
FROM Invoices
WHERE InvoiceNumber LIKE '[P-Q]%';
```

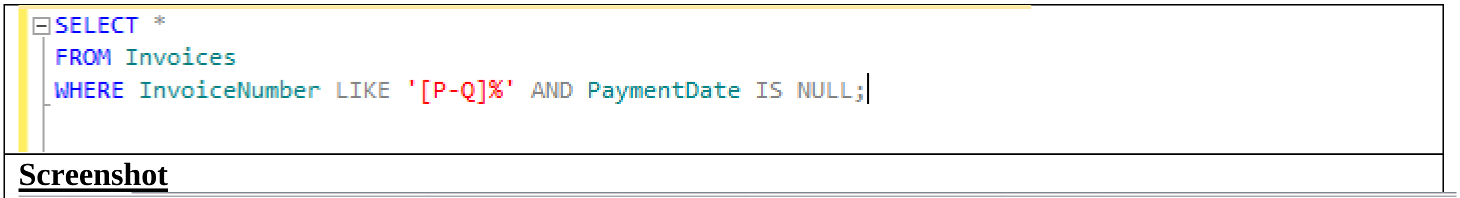
**Screenshot**



| | InvoiceID | VendorID | InvoiceNumber | InvoiceDate | InvoiceTotal | PaymentTotal | CreditTotal | TermsID | InvoiceDueDate | PaymentDate |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 15 | 48 | P02-88D77S7 | 2012-01-03 00:00:00 | 856.92 | 856.92 | 0.00 | 3 | 2012-02-02 00:00:00 | 2012-01-30 00:00:00 |
| 2 | 19 | 34 | QP58872 | 2012-01-07 00:00:00 | 116.54 | 116.54 | 0.00 | 1 | 2012-01-17 00:00:00 | 2012-01-19 00:00:00 |
| 3 | 31 | 104 | P02-3772 | 2012-01-21 00:00:00 | 7125.34 | 7125.34 | 0.00 | 3 | 2012-02-20 00:00:00 | 2012-02-24 00:00:00 |

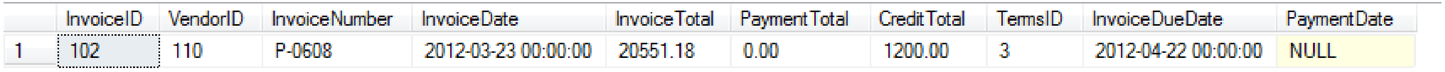
18. Same as (17) but only display those invoices that are not paid (PaymentDate is NULL) (Text113-115 Ch3_Slides39-44)

**SQL Code:**



```
SELECT *
FROM Invoices
WHERE InvoiceNumber LIKE '[P-Q]%' AND PaymentDate IS NULL;
```

**Screenshot**



|   | InvoiceID | VendorID | InvoiceNumber | InvoiceDate | InvoiceTotal | PaymentTotal | CreditTotal | TermsID | InvoiceDueDate | PaymentDate |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 102 | 110 | P-0608 | 2012-03-23 00:00:00 | 20551.18 | 0.00 | 1200.00 | 3 | 2012-04-22 00:00:00 | NULL |

**SUBMIT THIS DOCUMENT TO CANVAS: Course Documents -> Week1 -> Lab1**