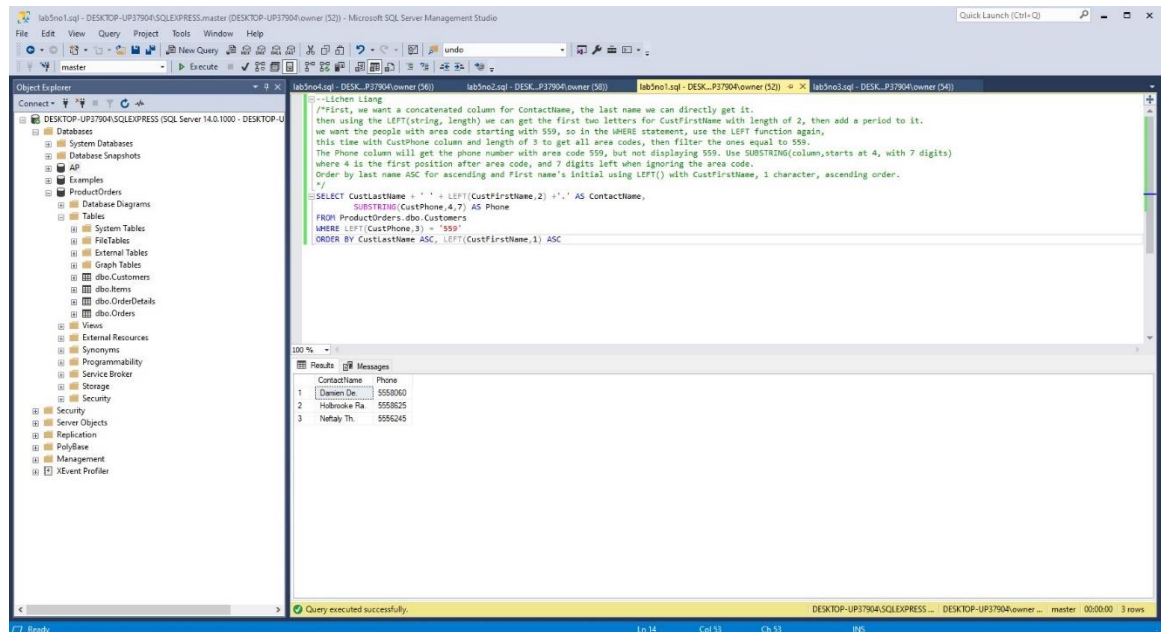


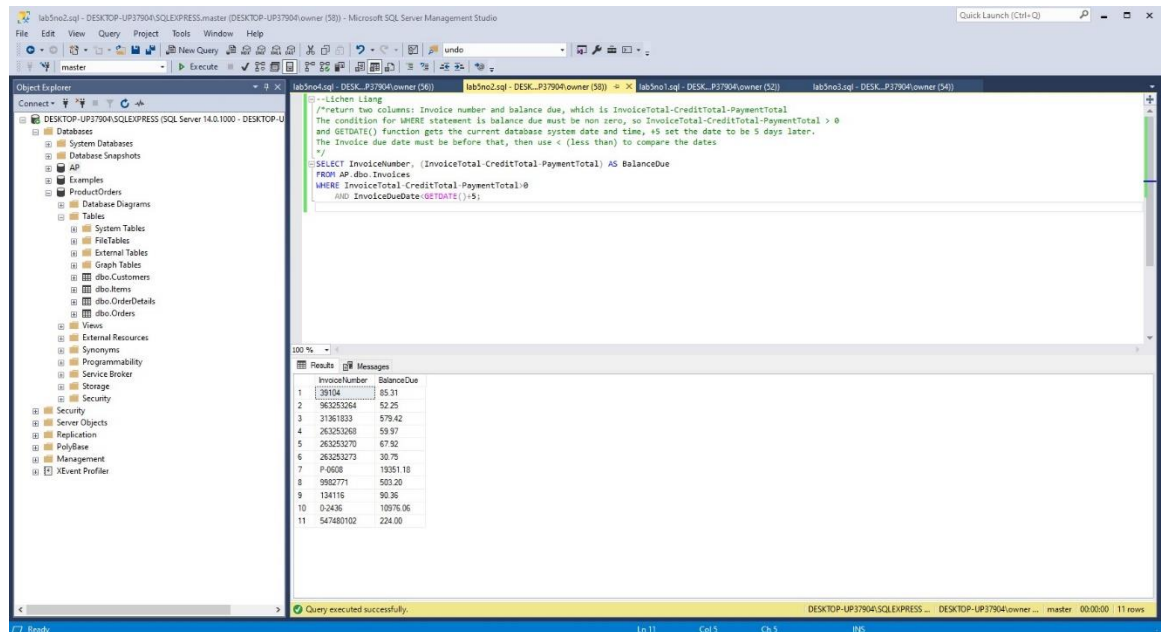
1.

```
SELECT CustLastName + ' ' + LEFT(CustFirstName,2) + '.' AS ContactName,  
SUBSTRING(CustPhone,4,7) AS Phone  
FROM ProductOrders.dbo.Customers  
WHERE LEFT(CustPhone,3) = '559'  
ORDER BY CustLastName ASC, LEFT(CustFirstName,1) ASC
```



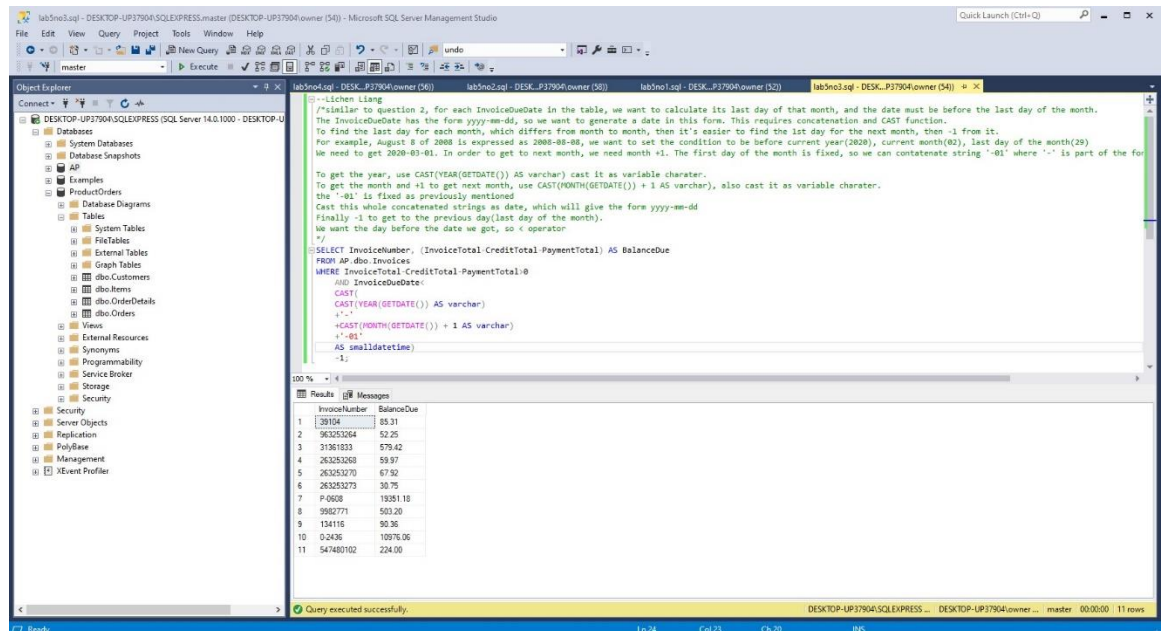
2.

```
SELECT InvoiceNumber, (InvoiceTotal-CreditTotal-PaymentTotal) AS  
BalanceDue  
FROM AP.dbo.Invoices  
WHERE InvoiceTotal-CreditTotal-PaymentTotal>0  
AND InvoiceDueDate<GETDATE()+5;
```



3.

```
SELECT InvoiceNumber, (InvoiceTotal-CreditTotal-PaymentTotal) AS
BalanceDue
FROM AP.dbo.Invoices
WHERE InvoiceTotal-CreditTotal-PaymentTotal>0
AND InvoiceDueDate<
CAST(
CAST(YEAR(GETDATE()) AS varchar)
+'-'
+CAST(MONTH(GETDATE()) + 1 AS varchar)
+'-01'
AS smalldatetime)
-1;
```



4.

```
SELECT InvoiceNumber,
(InvoiceTotal-CreditTotal-PaymentTotal) AS BalanceDue,
RANK() OVER (ORDER BY InvoiceTotal-CreditTotal-PaymentTotal
DESC) AS BalanceRank
FROM AP.dbo.Invoices
WHERE InvoiceTotal-CreditTotal-PaymentTotal>0
AND InvoiceDueDate<GETDATE()+5;
```

The screenshot displays the Microsoft SQL Server Enterprise Manager interface. The left pane shows the 'Object Explorer' with the 'master' database selected. The right pane shows a query window with the following SQL code:

```
--Lichen Liang
/*We want to get new column BalanceRank, using RANK() function, where balance is InvoiceTotal-CreditTotal-PaymentTotal.
using the RANK() OVER we will get a rank, use ORDER BY to get order of balance due in descending(DESC), and the column name BalanceRank
*/
SELECT InvoiceNumber,
       (InvoiceTotal-CreditTotal-PaymentTotal) AS BalanceDue,
       RANK() OVER (ORDER BY InvoiceTotal-CreditTotal-PaymentTotal DESC) AS BalanceRank
FROM AP.dbo.Invoices
WHERE InvoiceTotal-CreditTotal-PaymentTotal > 0
AND InvoiceDueDate < GETDATE();
```

The bottom pane shows the 'Results' tab with the following data:

InvoiceNumber	BalanceDue	BalanceRank
P-0608	1351.10	1
O-2436	10976.06	2
31361033	\$79.42	3
9982771	503.20	4
547480102	224.00	5
134116	90.36	6
39104	85.31	7
263253270	67.92	8
263253268	59.97	9
963253264	52.25	10
263253273	30.75	11

The status bar at the bottom indicates 'Query executed successfully.' and '11 rows'.

Remarks

In this lab we further practiced using functions, such as date/time, string, ranking, etc. Since there are so many functions there is no way to practice them all in one lab, I think question 3 is a very good practice in general. This lab is very efficient practice for lecture. The challenge would be more complicated and specific requirements that need more functions.