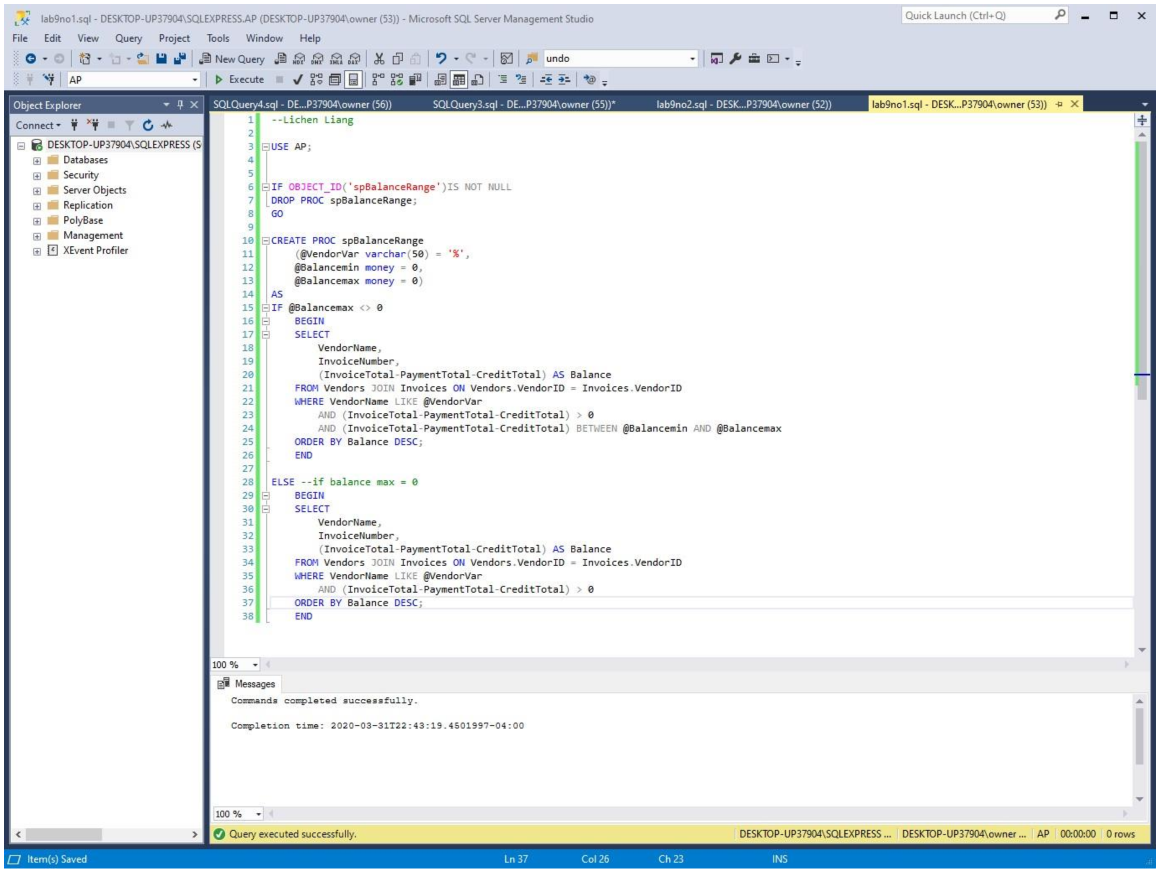
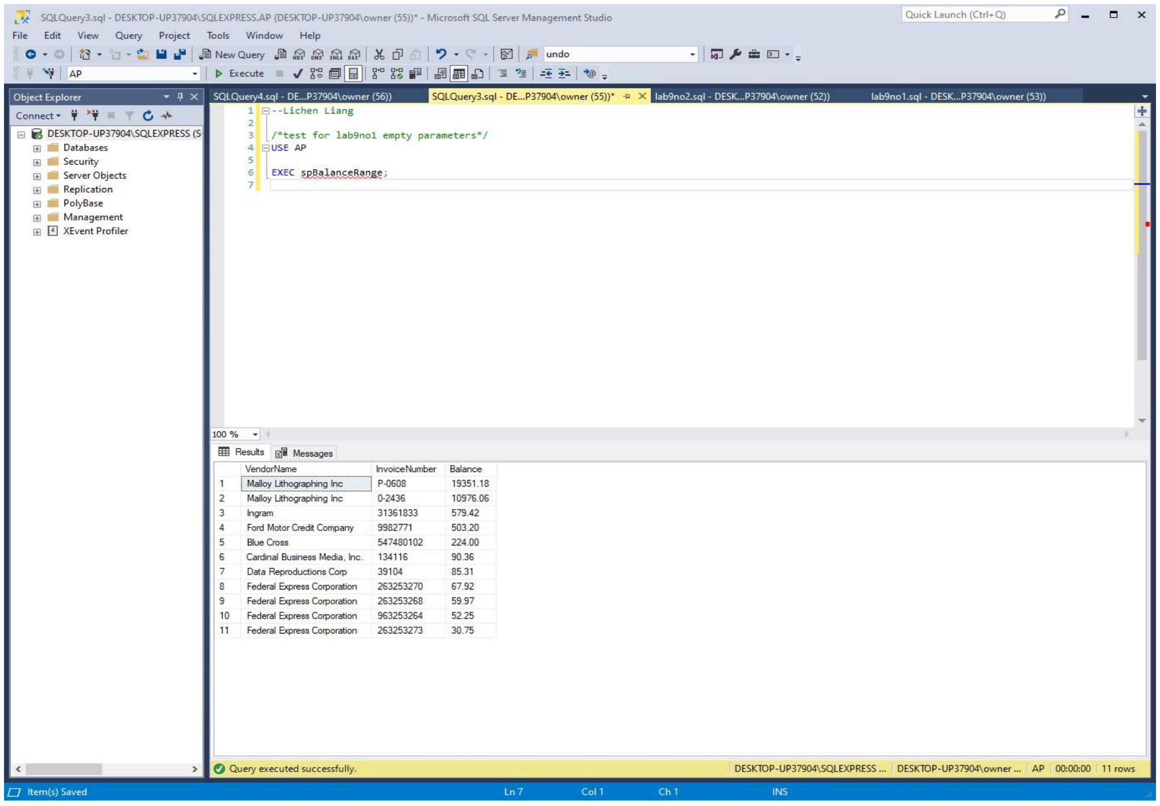
1. USE AP;

```
IF OBJECT_ID('spBalanceRange')IS NOT NULL
DROP PROC spBalanceRange;
GO

CREATE PROC spBalanceRange
        (@VendorVar varchar(50) = '%',
        @Balancemin money = 0,
        @Balancemax money = 0)
AS
IF @Balancemax <> 0
        BEGIN
        SELECT
                VendorName,
                InvoiceNumber,
                (InvoiceTotal-PaymentTotal-CreditTotal) AS Balance
        FROM Vendors JOIN Invoices ON Vendors.VendorID = Invoices.VendorID
        WHERE VendorName LIKE @VendorVar
                AND (InvoiceTotal-PaymentTotal-CreditTotal) > 0
                AND (InvoiceTotal-PaymentTotal-CreditTotal) BETWEEN
@Balancemin AND @Balancemax
        ORDER BY Balance DESC;
        END

ELSE --if balance max = 0
        BEGIN
        SELECT
                VendorName,
                InvoiceNumber,
                (InvoiceTotal-PaymentTotal-CreditTotal) AS Balance
        FROM Vendors JOIN Invoices ON Vendors.VendorID = Invoices.VendorID
        WHERE VendorName LIKE @VendorVar
                AND (InvoiceTotal-PaymentTotal-CreditTotal) > 0
        ORDER BY Balance DESC;
        END
```



```sql
--Lichen Liang

USE AP;


IF OBJECT_ID('spBalanceRange')IS NOT NULL
DROP PROC spBalanceRange;
GO

CREATE PROC spBalanceRange
    (@VendorVar varchar(50) = '%',
    @Balancemin money = 0,
    @Balancemax money = 0)
AS
IF @Balancemax <> 0
    BEGIN
    SELECT
        VendorName,
        InvoiceNumber,
        (InvoiceTotal-PaymentTotal-CreditTotal) AS Balance
    FROM Vendors JOIN Invoices ON Vendors.VendorID = Invoices.VendorID
    WHERE VendorName LIKE @VendorVar
        AND (InvoiceTotal-PaymentTotal-CreditTotal) > 0
        AND (InvoiceTotal-PaymentTotal-CreditTotal) BETWEEN @Balancemin AND @Balancemax
    ORDER BY Balance DESC;
    END

ELSE --if balance max = 0
    BEGIN
    SELECT
        VendorName,
        InvoiceNumber,
        (InvoiceTotal-PaymentTotal-CreditTotal) AS Balance
    FROM Vendors JOIN Invoices ON Vendors.VendorID = Invoices.VendorID
    WHERE VendorName LIKE @VendorVar
        AND (InvoiceTotal-PaymentTotal-CreditTotal) > 0
    ORDER BY Balance DESC;
    END
```

```
Commands completed successfully.

Completion time: 2020-03-31T22:43:19.4501997-04:00
```



```sql
--Lichen Liang

/*test for lab9no1 empty parameters*/
USE AP

EXEC spBalanceRange;
```

	VendorName	InvoiceNumber	Balance
1	Malloy Lithographing Inc	P-0608	19351.18
2	Malloy Lithographing Inc	0-2436	10976.06
3	Ingram	31361833	579.42
4	Ford Motor Credit Company	9982771	503.20
5	Blue Cross	547480102	224.00
6	Cardinal Business Media, Inc.	134116	90.36
7	Data Reproductions Corp	39104	85.31
8	Federal Express Corporation	263253270	67.92
9	Federal Express Corporation	263253268	59.97
10	Federal Express Corporation	963253264	52.25
11	Federal Express Corporation	263253273	30.75

Query executed successfully.

Creating a procedure using CREATE PROC…AS with 3 input parameters, their corresponding datatype and their initial value. Then use the IF..ELSE to check our conditions. If the Balancemax entered is non-zero, then the query returns 3 columns from two tables Vendors and Invoices where their vendor ID has to match. The VendorVar is a mask to the VendorName using the LIKE, the there should be a balance due, and the balance due should be between our inputs Balancemin and Balancemax using BETWEEN…AND... Finally ORDER BY the Balance in descending using DESC to sort the largest balance due first.

If Balancemax is entered to be 0 or any of the parameters are not entered, then using the same query to return all invoices with a balance due. The query is similar to above explained, but not including the part in the WHERE clause that Balance is between Balancemin and Balancemax.

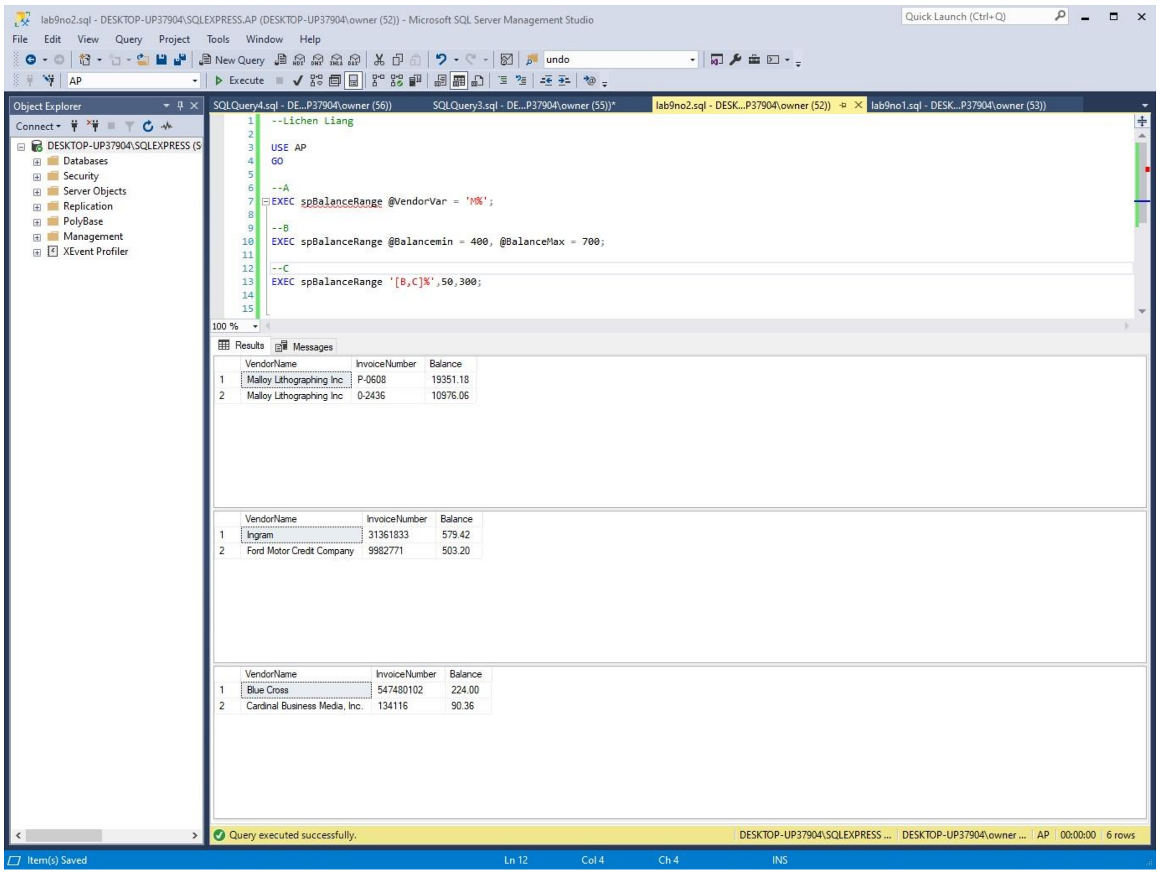
The second screenshot executes the procedure without any parameters, which returned all invoiced with a balance due.

2. USE AP
   GO

   --A
   EXEC spBalanceRange @VendorVar = 'M%';

   --B
   EXEC spBalanceRange @Balancemin = 400, @BalanceMax = 700;

   --C
   EXEC spBalanceRange '[B,C]%',50,300;



```
--Lichen Liang

USE AP
GO

--A
EXEC spBalanceRange @VendorVar = 'M%';

--B
EXEC spBalanceRange @Balancemin = 400, @BalanceMax = 700;

--C
EXEC spBalanceRange '[B,C]%',50,300;
```

	VendorName	InvoiceNumber	Balance
1	Malloy Lithographing Inc	P-0608	19351.18
2	Malloy Lithographing Inc	0-2436	10976.06

	VendorName	InvoiceNumber	Balance
1	Ingram	31361833	579.42
2	Ford Motor Credit Company	9982771	503.20

	VendorName	InvoiceNumber	Balance
1	Blue Cross	547480102	224.00
2	Cardinal Business Media, Inc.	134116	90.36

Query executed successfully.

This part we call the procedure three times with different parameters using EXEC.

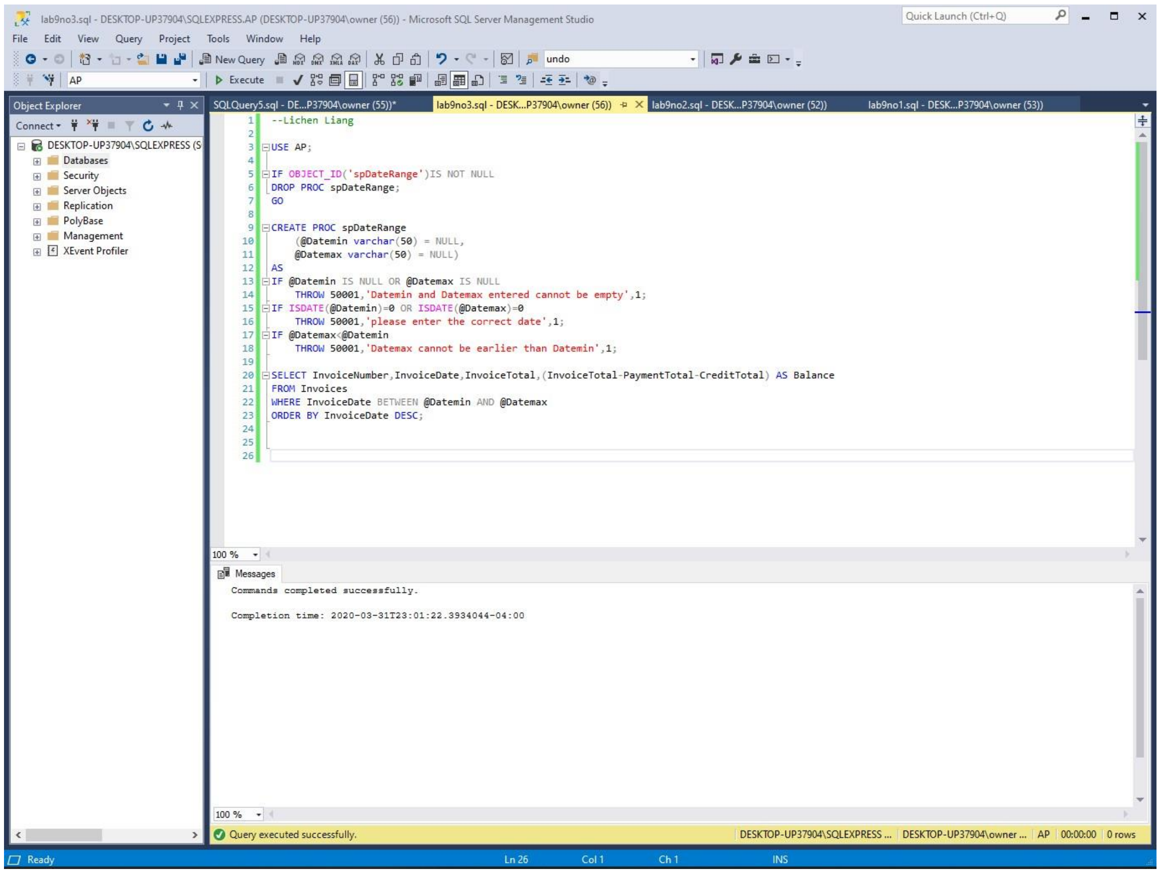
The first returns all invoices such that VendorName starts with M, and % denotes anything that comes after it.

The second returns all invoiced between the 2 balance we entered.

The third with all parameters entered returns VendorName starts with B or C and in the balance range.

3. USE AP;

```
IF OBJECT_ID('spDateRange')IS NOT NULL
DROP PROC spDateRange;
GO

CREATE PROC spDateRange
            (@Datemin varchar(50) = NULL,
            @Datemax varchar(50) = NULL)
AS
IF @Datemin IS NULL OR @Datemax IS NULL
            THROW 50001,'Datemin and Datemax entered cannot be empty',1;
IF ISDATE(@Datemin)=0 OR ISDATE(@Datemax)=0
            THROW 50001,'please enter the correct date',1;
IF @Datemax<@Datemin
            THROW 50001,'Datemax cannot be earlier than Datemin',1;

SELECT InvoiceNumber,InvoiceDate,InvoiceTotal,(InvoiceTotal-PaymentTotal-
CreditTotal) AS Balance
FROM Invoices
WHERE InvoiceDate BETWEEN @Datemin AND @Datemax
ORDER BY InvoiceDate DESC;
```


lab9no3.sql - DESKTOP-UP37904\SQLEXPRESS.AP (DESKTOP-UP37904\owner (56)) - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Query Project Tools Window Help
New Query
undo
AP
Execute
Object Explorer
Connect
DESKTOP-UP37904\SQLEXPRESS (S
Databases
Security
Server Objects
Replication
PolyBase
Management
XEvent Profiler
SQLQuery5.sql - DE...P37904\owner (55))*
lab9no3.sql - DESK...P37904\owner (56))
lab9no2.sql - DESK...P37904\owner (52))
lab9no1.sql - DESK...P37904\owner (53))
100 %
Messages
Ready
Ln 26
Col 1
Ch 1
INS

```
--Lichen Liang

USE AP;

IF OBJECT_ID('spDateRange')IS NOT NULL
DROP PROC spDateRange;
GO

CREATE PROC spDateRange
    (@Datemin varchar(50) = NULL,
    @Datemax varchar(50) = NULL)
AS
IF @Datemin IS NULL OR @Datemax IS NULL
    THROW 50001,'Datemin and Datemax entered cannot be empty',1;
IF ISDATE(@Datemin)=0 OR ISDATE(@Datemax)=0
    THROW 50001,'please enter the correct date',1;
IF @Datemax<@Datemin
    THROW 50001,'Datemax cannot be earlier than Datemin',1;

SELECT InvoiceNumber,InvoiceDate,InvoiceTotal,(InvoiceTotal-PaymentTotal-CreditTotal) AS Balance
FROM Invoices
WHERE InvoiceDate BETWEEN @Datemin AND @Datemax
ORDER BY InvoiceDate DESC;
```

```
Commands completed successfully.

Completion time: 2020-03-31T23:01:22.3934044-04:00
```

Query executed successfully. | DESKTOP-UP37904\SQLEXPRESS ... | DESKTOP-UP37904\owner ... | AP | 00:00:00 | 0 rows

Creating another procedure using CREATE PROC…AS with 2 parameters dates and their initial value set to NULL.

Using the IF to check constrains and THROW to raise errors.

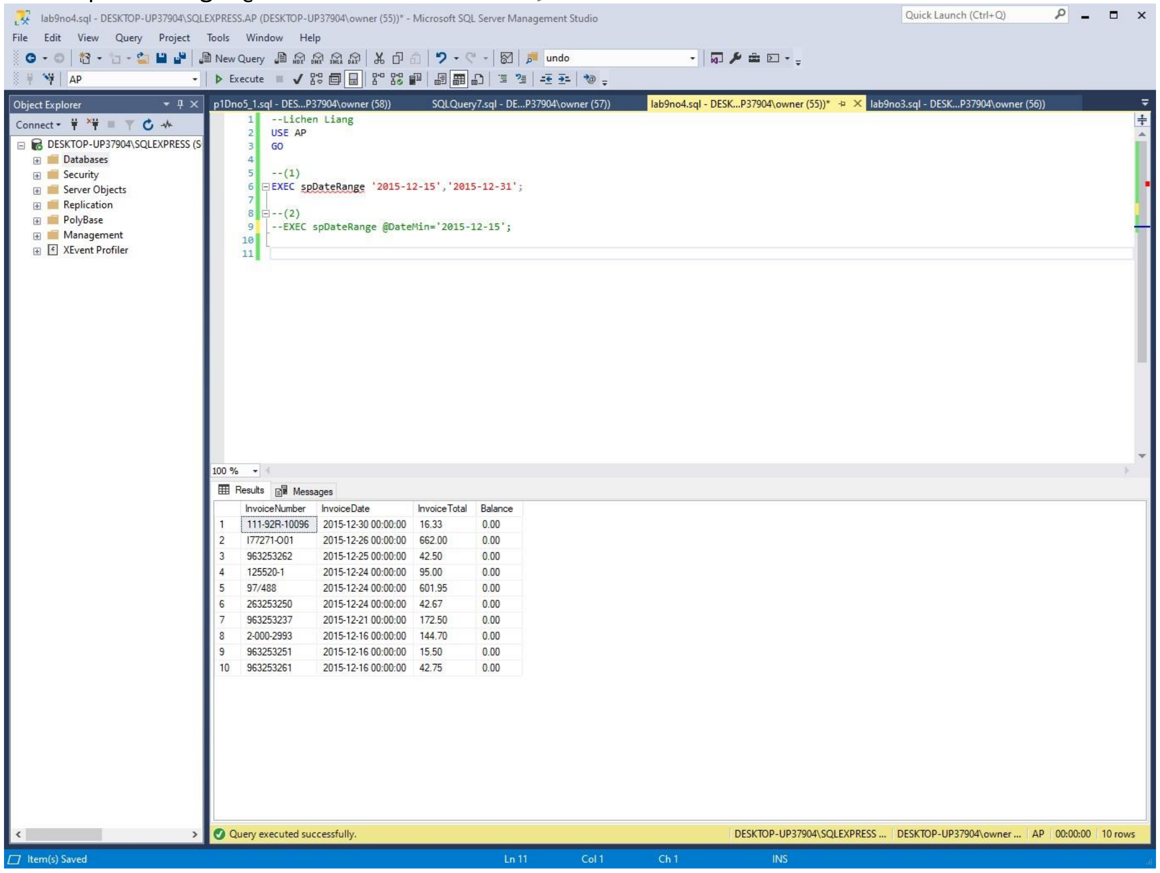
If any of the two are NULL or not entered, then raise error.

If any of the two dates are entered in wrong format which is checked using ISDATE() is false(=0), then raise error.

If Datemax is larger than Datemin (Datemax before Datemin), raise error.

Finally, select 4 columns from Invoiced such that the date is between the two dates we entered using BETWEEN..AND.., and sort by latest invoice first, using ORDER BY the invoice date in DESC.

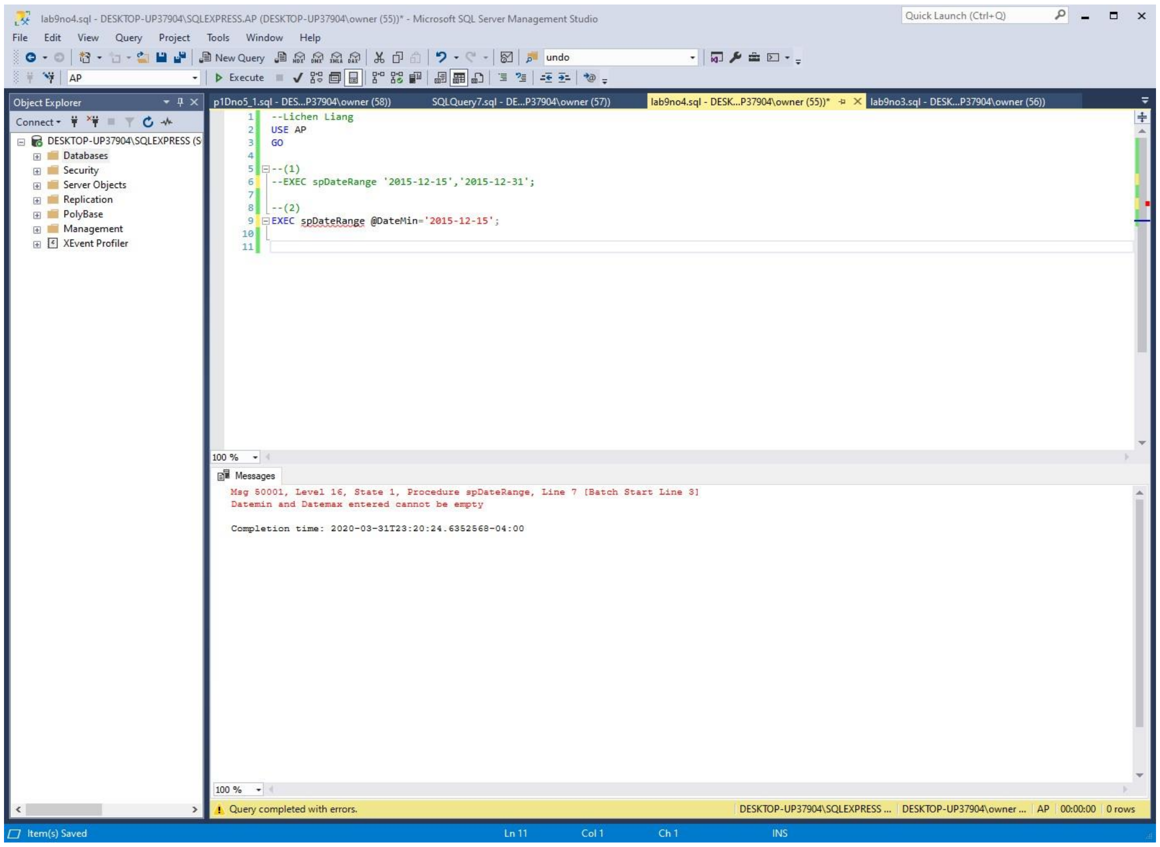
```
4. USE AP
GO

--(1)
EXEC spDateRange '2015-12-15','2015-12-31';

--(2)
EXEC spDateRange @DateMin='2015-12-15';
```



```
--Lichen Liang
USE AP
GO

--(1)
EXEC spDateRange '2015-12-15','2015-12-31';

--(2)
--EXEC spDateRange @DateMin='2015-12-15';
```

	InvoiceNumber	InvoiceDate	InvoiceTotal	Balance
1	111-92R-10096	2015-12-30 00:00:00	16.33	0.00
2	I77271-O01	2015-12-26 00:00:00	662.00	0.00
3	963253262	2015-12-25 00:00:00	42.50	0.00
4	125520-1	2015-12-24 00:00:00	95.00	0.00
5	97/488	2015-12-24 00:00:00	601.95	0.00
6	263253250	2015-12-24 00:00:00	42.67	0.00
7	963253237	2015-12-21 00:00:00	172.50	0.00
8	2-000-2993	2015-12-16 00:00:00	144.70	0.00
9	963253251	2015-12-16 00:00:00	15.50	0.00
10	963253261	2015-12-16 00:00:00	42.75	0.00

Query executed successfully. DESKTOP-UP37904\SQLEXPRESS ... DESKTOP-UP37904\owner ... AP 00:00:00 10 rows

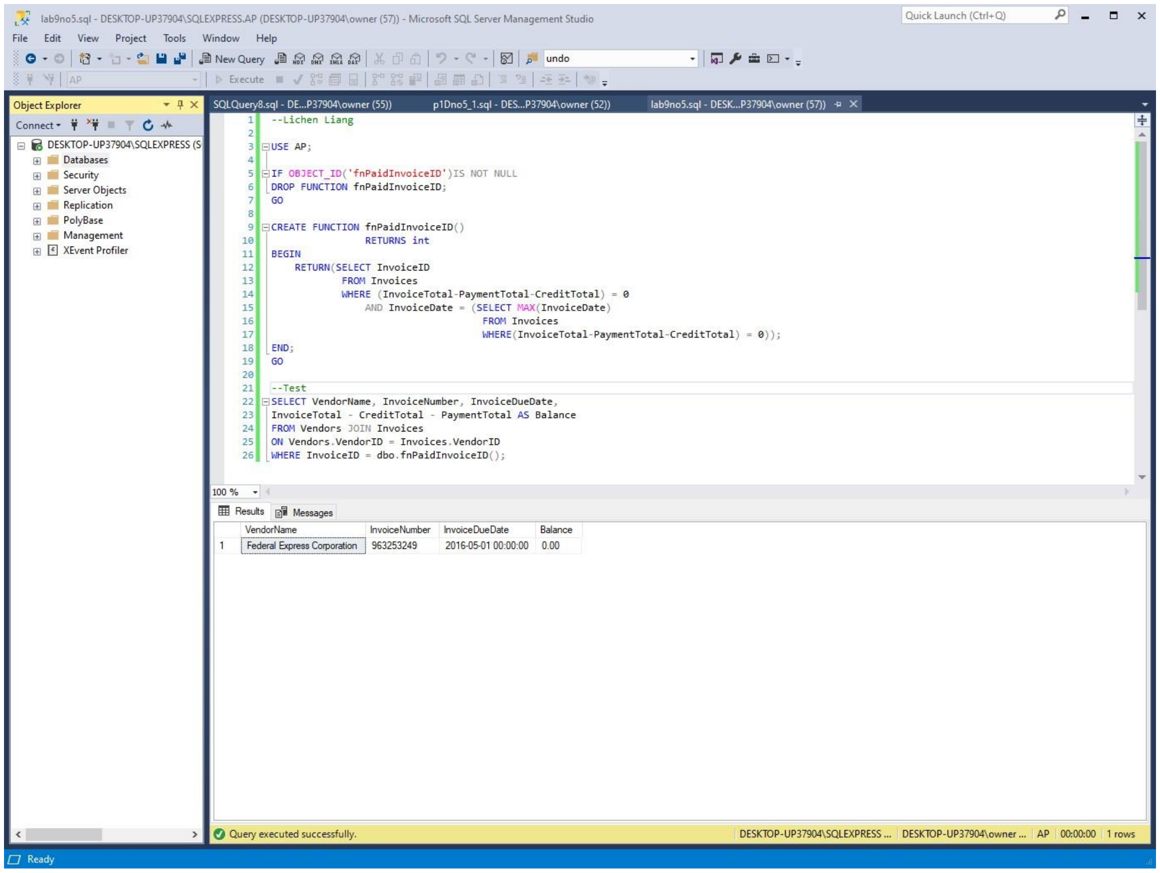

```sql
--Lichen Liang
USE AP
GO

--(1)
--EXEC spDateRange '2015-12-15','2015-12-31';

--(2)
EXEC spDateRange @DateMin='2015-12-15';
```

```
Msg 50001, Level 16, State 1, Procedure spDateRange, Line 7 [Batch Start Line 3]
Datemin and Datemax entered cannot be empty

Completion time: 2020-03-31T23:20:24.6352568-04:00
```

Query completed with errors.

Testing the procedure from last question using EXEC.
The first one returns the invoices between the two dates we entered.
The second one raised an error because we only entered one date (one parameter) instead of two, so one of the parameter is still NULL.

5.
```
USE AP;

IF OBJECT_ID('fnPaidInvoiceID')IS NOT NULL
DROP FUNCTION fnPaidInvoiceID;
GO

CREATE FUNCTION fnPaidInvoiceID()
                            RETURNS int
BEGIN
       RETURN(SELECT InvoiceID
                     FROM Invoices
                     WHERE (InvoiceTotal-PaymentTotal-CreditTotal) = 0
                            AND InvoiceDate = (SELECT MAX(InvoiceDate)
                                                      FROM Invoices

       WHERE(InvoiceTotal-PaymentTotal-CreditTotal) = 0));
END;
GO

--Test
SELECT VendorName, InvoiceNumber, InvoiceDueDate,
InvoiceTotal - CreditTotal - PaymentTotal AS Balance
FROM Vendors JOIN Invoices
ON Vendors.VendorID = Invoices.VendorID
   WHERE InvoiceID = dbo.fnPaidInvoiceID();
```


lab9no5.sql - DESKTOP-UP37904\SQLEXPRESS.AP (DESKTOP-UP37904\owner (57)) - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Project Tools Window Help
New Query
undo
AP
Execute
Object Explorer
Connect
DESKTOP-UP37904\SQLEXPRESS (S
Databases
Security
Server Objects
Replication
PolyBase
Management
XEvent Profiler
SQLQuery8.sql - DE...P37904\owner (55))
p1Dno5_1.sql - DES...P37904\owner (52))
lab9no5.sql - DESK...P37904\owner (57))
100 %
Results
Messages
Ready

```
--Lichen Liang

USE AP;

IF OBJECT_ID('fnPaidInvoiceID')IS NOT NULL
DROP FUNCTION fnPaidInvoiceID;
GO

CREATE FUNCTION fnPaidInvoiceID()
                RETURNS int
BEGIN
    RETURN(SELECT InvoiceID
            FROM Invoices
            WHERE (InvoiceTotal-PaymentTotal-CreditTotal) = 0
                AND InvoiceDate = (SELECT MAX(InvoiceDate)
                                   FROM Invoices
                                   WHERE(InvoiceTotal-PaymentTotal-CreditTotal) = 0));
END;
GO

--Test
SELECT VendorName, InvoiceNumber, InvoiceDueDate,
InvoiceTotal - CreditTotal - PaymentTotal AS Balance
FROM Vendors JOIN Invoices
ON Vendors.VendorID = Invoices.VendorID
WHERE InvoiceID = dbo.fnPaidInvoiceID();
```

	VendorName	InvoiceNumber	InvoiceDueDate	Balance
1	Federal Express Corporation	963253249	2016-05-01 00:00:00	0.00

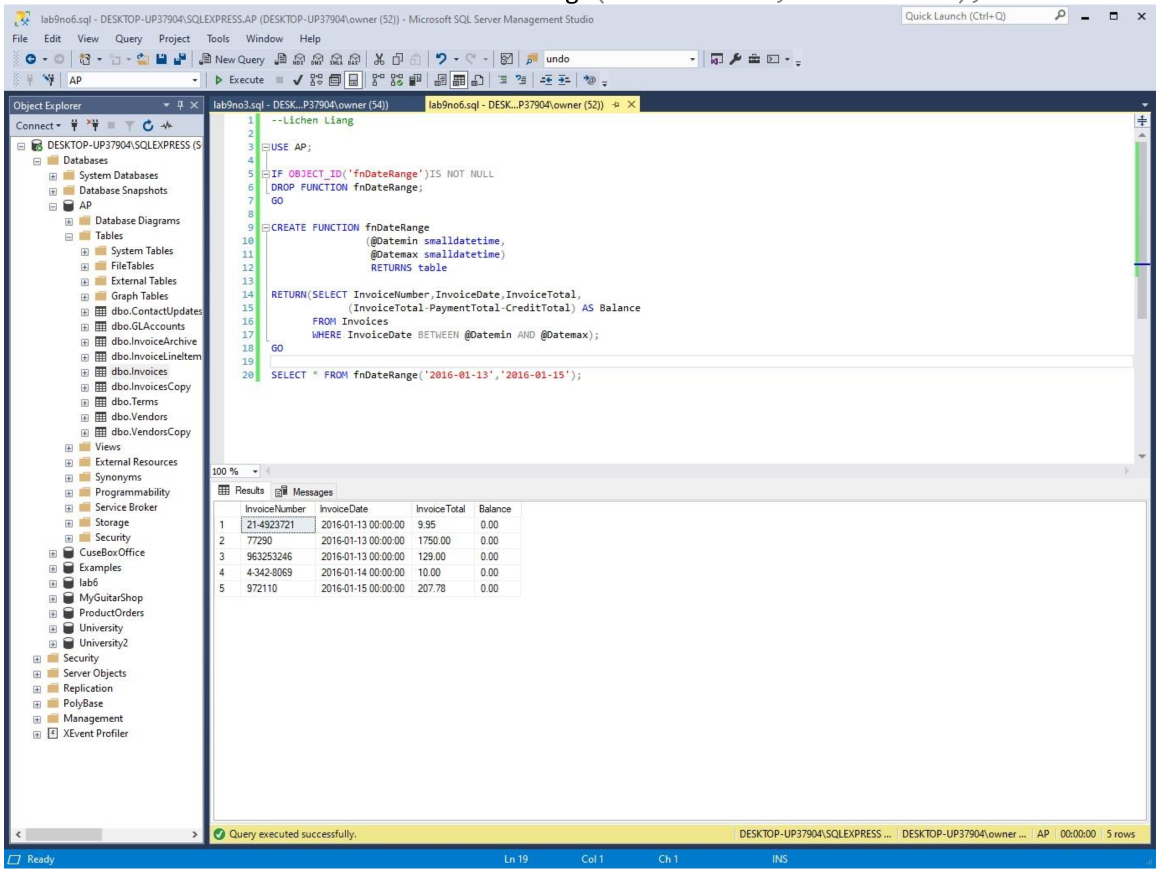
Query executed successfully. | DESKTOP-UP37904\SQLEXPRESS ... | DESKTOP-UP37904\owner ... | AP | 00:00:00 | 1 rows

Creating a function using CREATE FUNCTION which returns an integer. The query returns the invoice ID such that Balance due is 0 and date must match the date from the subquery. The subquery selects the latest invoice date by using MAX() and the balance should also be 0.

Then we use the query provided in the document that calls our function which the InvoiceID from this query is matched to the result we got from the function.

To test the function, we can select invoice ID from invoices table with balance is 0, then order by the invoice date descending. The first entry is the latest paid invoice. If we add a column InvoiceID to the test query provided, then this ID matches with the first entry we got.

6.
```
USE AP;

IF OBJECT_ID('fnDateRange')IS NOT NULL
DROP FUNCTION fnDateRange;
GO

CREATE FUNCTION fnDateRange
                                    (@Datemin smalldatetime,
                                     @Datemax smalldatetime)
                                     RETURNS table

RETURN(SELECT InvoiceNumber,InvoiceDate,InvoiceTotal,
                    (InvoiceTotal-PaymentTotal-CreditTotal) AS Balance
          FROM Invoices
          WHERE InvoiceDate BETWEEN @Datemin AND @Datemax);
GO

             SELECT * FROM fnDateRange('2016-01-13','2016-01-15');
```


lab9no6.sql - DESKTOP-UP37904\SQLEXPRESS.AP (DESKTOP-UP37904\owner (52)) - Microsoft SQL Server Management Studio

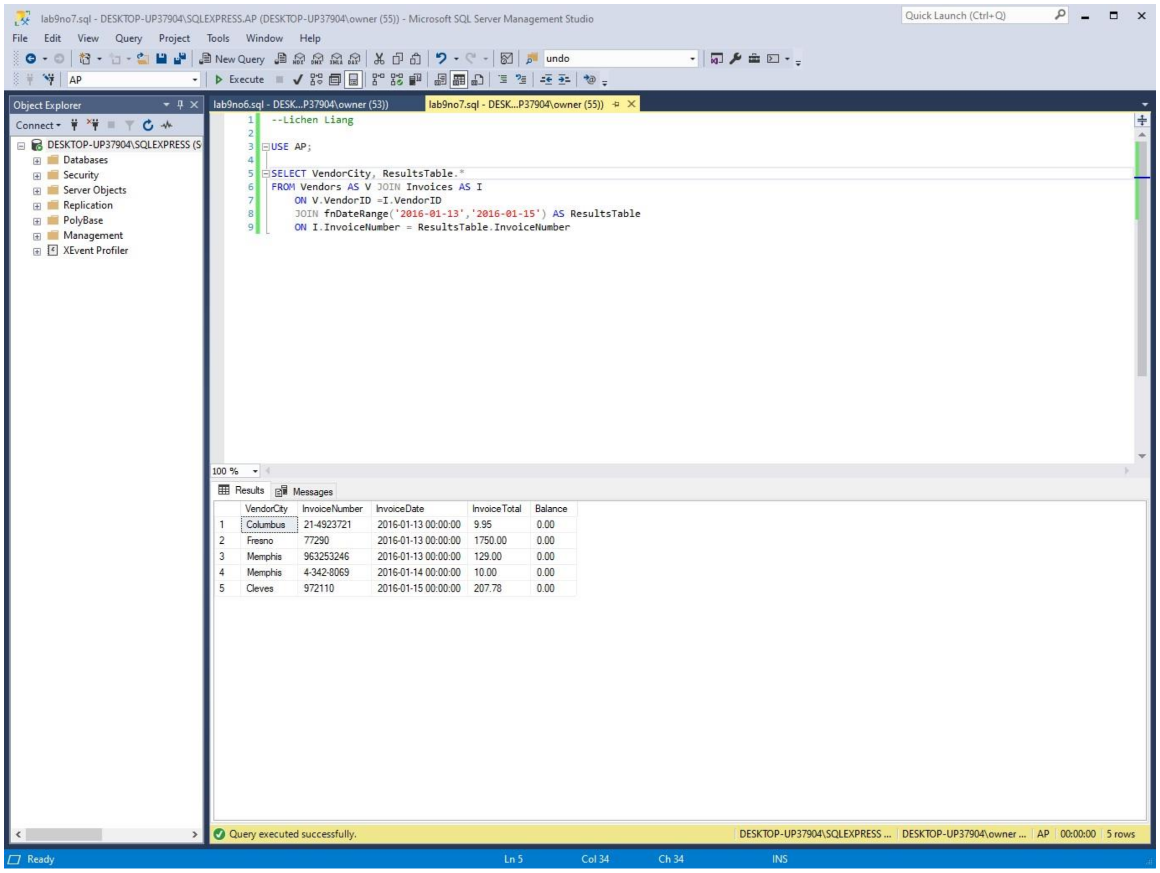
```
--Lichen Liang

USE AP;

IF OBJECT_ID('fnDateRange')IS NOT NULL
DROP FUNCTION fnDateRange;
GO

CREATE FUNCTION fnDateRange
              (@Datemin smalldatetime,
               @Datemax smalldatetime)
               RETURNS table

RETURN(SELECT InvoiceNumber,InvoiceDate,InvoiceTotal,
              (InvoiceTotal-PaymentTotal-CreditTotal) AS Balance
       FROM Invoices
       WHERE InvoiceDate BETWEEN @Datemin AND @Datemax);
GO

SELECT * FROM fnDateRange('2016-01-13','2016-01-15');
```

	InvoiceNumber	InvoiceDate	InvoiceTotal	Balance
1	21-4923721	2016-01-13 00:00:00	9.95	0.00
2	77290	2016-01-13 00:00:00	1750.00	0.00
3	963253246	2016-01-13 00:00:00	129.00	0.00
4	4-342-8069	2016-01-14 00:00:00	10.00	0.00
5	972110	2016-01-15 00:00:00	207.78	0.00

Query executed successfully. DESKTOP-UP37904\SQLEXPRESS ... DESKTOP-UP37904\owner ... AP 00:00:00 5 rows

This part we are creating the function using CREATE FUNCTION, with 2 parameters and smalldatetime datatype. The function returns a table.
The query selects the invoices that are between the 2 dates from the 2 parameters, which is similar to question 3 but without the validation.
Then using a select statement to return all columns by calling the function with 2 dates.

```
7. USE AP;

   SELECT VendorCity, ResultsTable.*
   FROM Vendors AS V JOIN Invoices AS I
          ON V.VendorID =I.VendorID
          JOIN fnDateRange('2016-01-13','2016-01-15') AS ResultsTable
          ON I.InvoiceNumber = ResultsTable.InvoiceNumber
```



lab9no7.sql - DESKTOP-UP37904\SQLEXPRESS.AP (DESKTOP-UP37904\owner (55)) - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Query Project Tools Window Help
New Query
undo
AP
Execute
Object Explorer
Connect
DESKTOP-UP37904\SQLEXPRESS (S
Databases
Security
Server Objects
Replication
PolyBase
Management
XEvent Profiler
lab9no6.sql - DESK...P37904\owner (53))
lab9no7.sql - DESK...P37904\owner (55))
100 %
Results
Messages

```
--Lichen Liang

USE AP;

SELECT VendorCity, ResultsTable.*
FROM Vendors AS V JOIN Invoices AS I
    ON V.VendorID =I.VendorID
    JOIN fnDateRange('2016-01-13','2016-01-15') AS ResultsTable
    ON I.InvoiceNumber = ResultsTable.InvoiceNumber
```

	VendorCity	InvoiceNumber	InvoiceDate	InvoiceTotal	Balance
1	Columbus	21-4923721	2016-01-13 00:00:00	9.95	0.00
2	Fresno	77290	2016-01-13 00:00:00	1750.00	0.00
3	Memphis	963253246	2016-01-13 00:00:00	129.00	0.00
4	Memphis	4-342-8069	2016-01-14 00:00:00	10.00	0.00
5	Cleves	972110	2016-01-15 00:00:00	207.78	0.00

Query executed successfully. DESKTOP-UP37904\SQLEXPRESS ... DESKTOP-UP37904\owner ... AP 00:00:00 5 rows

Ready Ln 5 Col 34 Ch 34 INS

This part we are selecting 1 column and all the columns from the function we just wrote. The vendor IDs should match between the Vendors and Invoices table. Also joins the function we called and name it as ResultTable, on the invoice numbers has to match. Note we used '.*' after the ResultsTable in the SELECT so that it shows all the columns that we retrieved from this function.

Remarks
This lab we practiced creating procedure and functions. I think it’s a very good practice for the lecture as the materials are becoming more difficult. However, this material is an overlap with the project, so it is not that difficult once I learned the concept. A more challenging lab would be adding more requirements such as complicating the query or adding more validation steps or conditions.