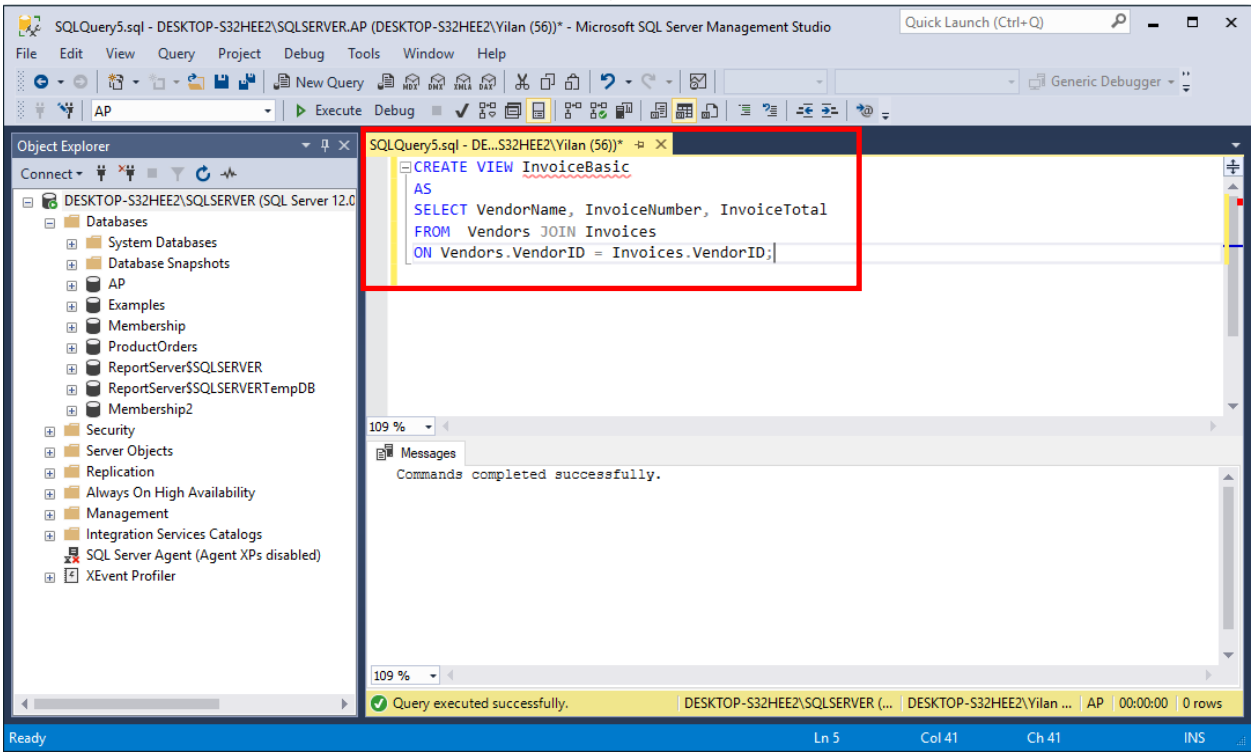
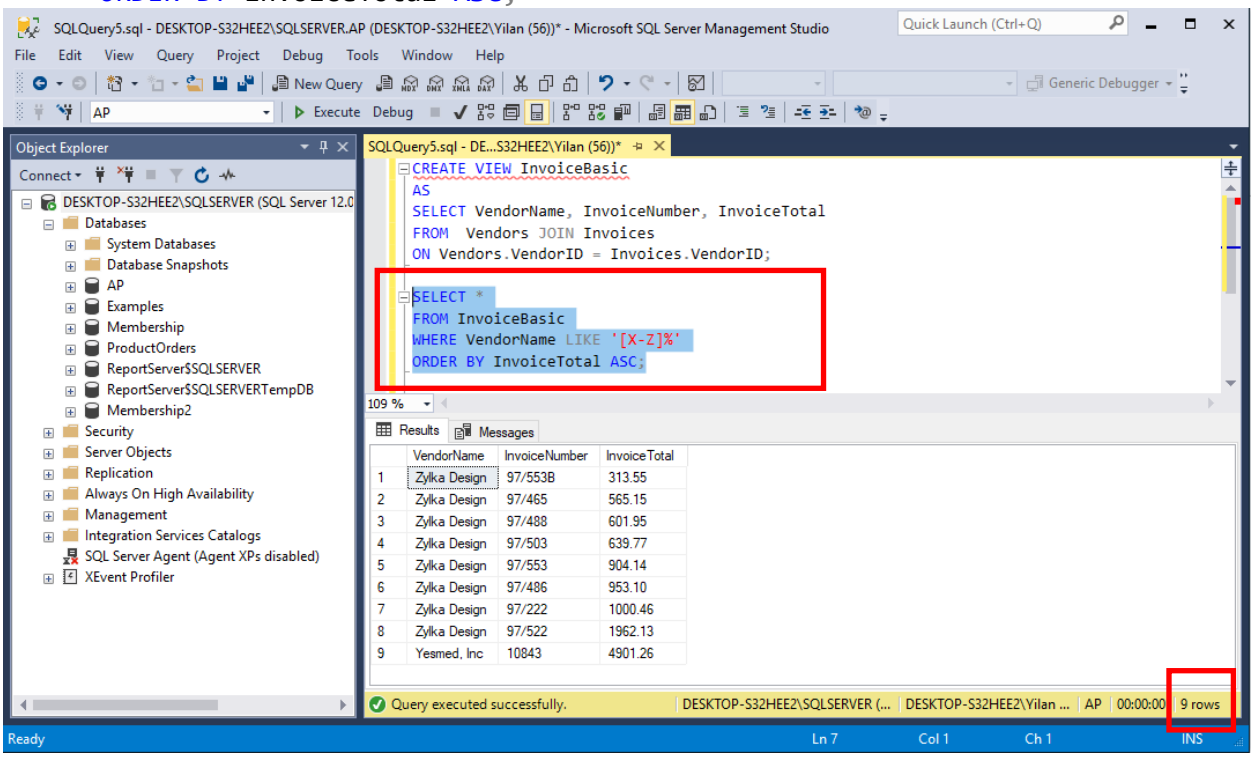
Lab 8: Views, Scripts Solution

1. Write a view named InvoiceBasic that returns three columns: VendorName, InvoiceNumber, and InvoiceTotal. Then, write a SELECT statement that returns all the columns in the view, sorted by InvoiceTotal from smallest to largest, where the first letter of the vendor name is X, Y or Z.

```
CREATE VIEW InvoiceBasic
AS
SELECT VendorName, InvoiceNumber, InvoiceTotal
FROM  Vendors JOIN Invoices
ON Vendors.VendorID = Invoices.VendorID;
```

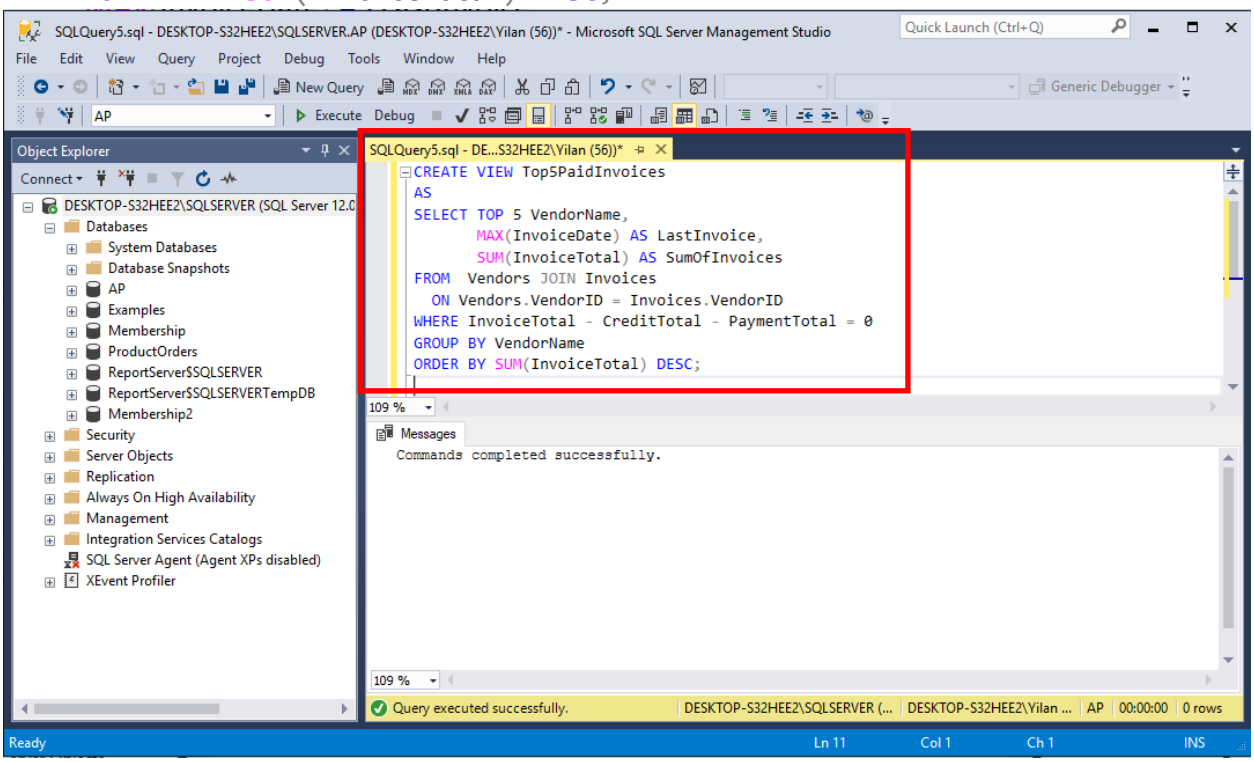


```sql
SELECT *
FROM InvoiceBasic
WHERE VendorName LIKE '[X-Z]%'
ORDER BY InvoiceTotal ASC;
```

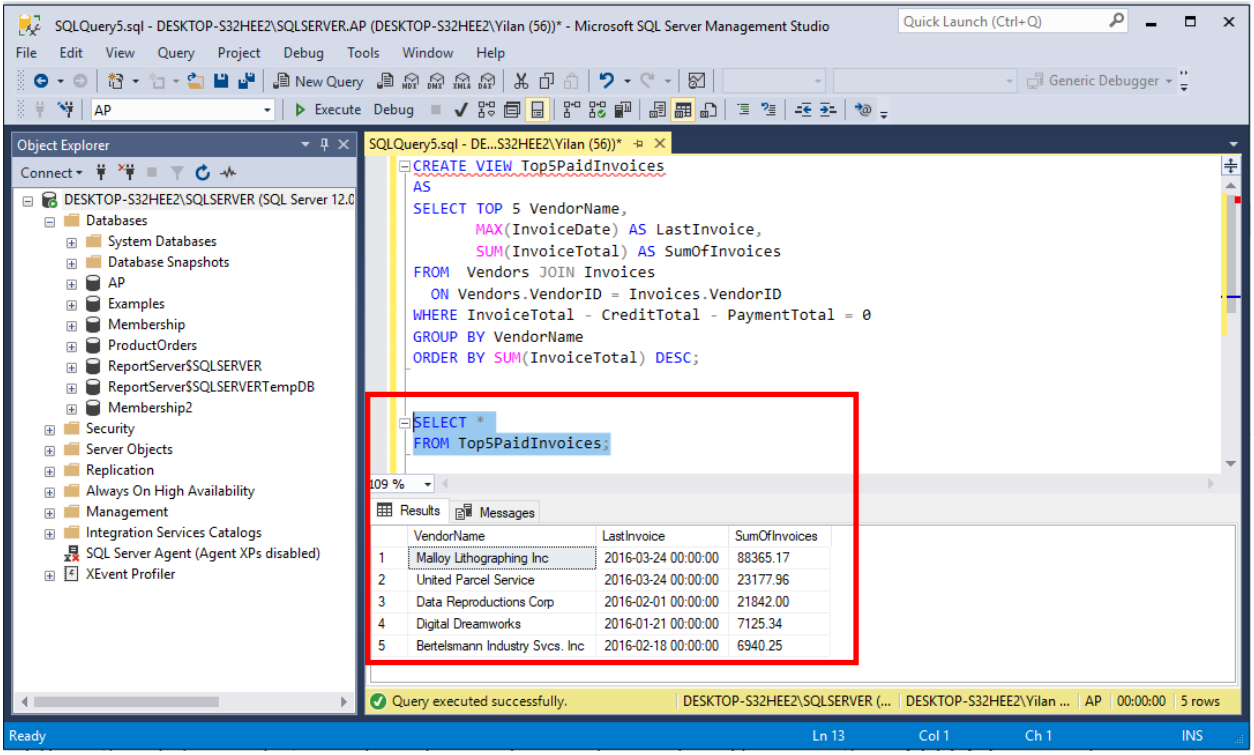


2. Create a view named Top5PaidInvoices that returns three columns for each vendor: VendorName, LastInvoiceDate (the most recent invoice date), and SumOfInvoices (the sum of the InvoiceTotal column). Return only the 5 vendors with the largest SumOfInvoices and include only paid invoices (i.e. InvoiceTotal – CredeitTotal – PaymentTotal = 0). Then write a SELECT statement to show results of the view.

```
CREATE VIEW Top5PaidInvoices
AS
SELECT TOP 5 VendorName,
       MAX(InvoiceDate) AS LastInvoice,
       SUM(InvoiceTotal) AS SumOfInvoices
FROM  Vendors JOIN Invoices
  ON Vendors.VendorID = Invoices.VendorID
WHERE InvoiceTotal - CreditTotal - PaymentTotal = 0
GROUP BY VendorName
ORDER BY SUM(InvoiceTotal) DESC;
```

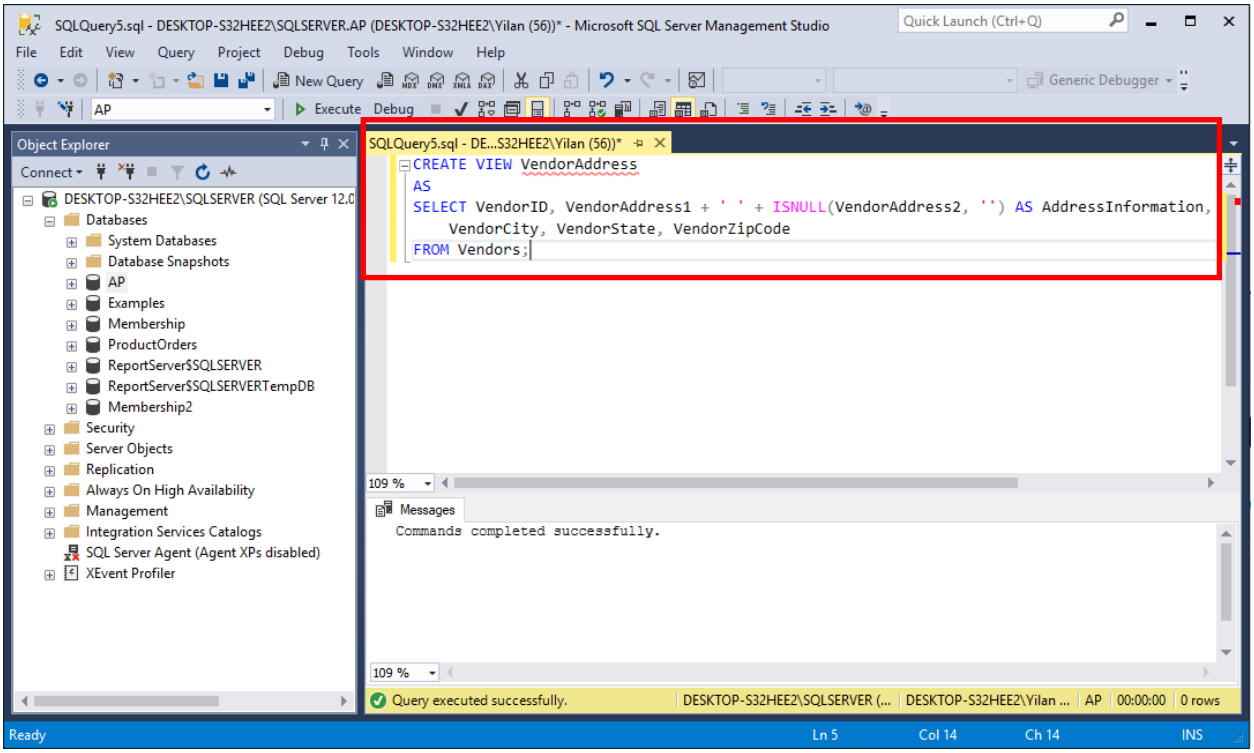


```
SELECT *
FROM Top5PaidInvoices;
```

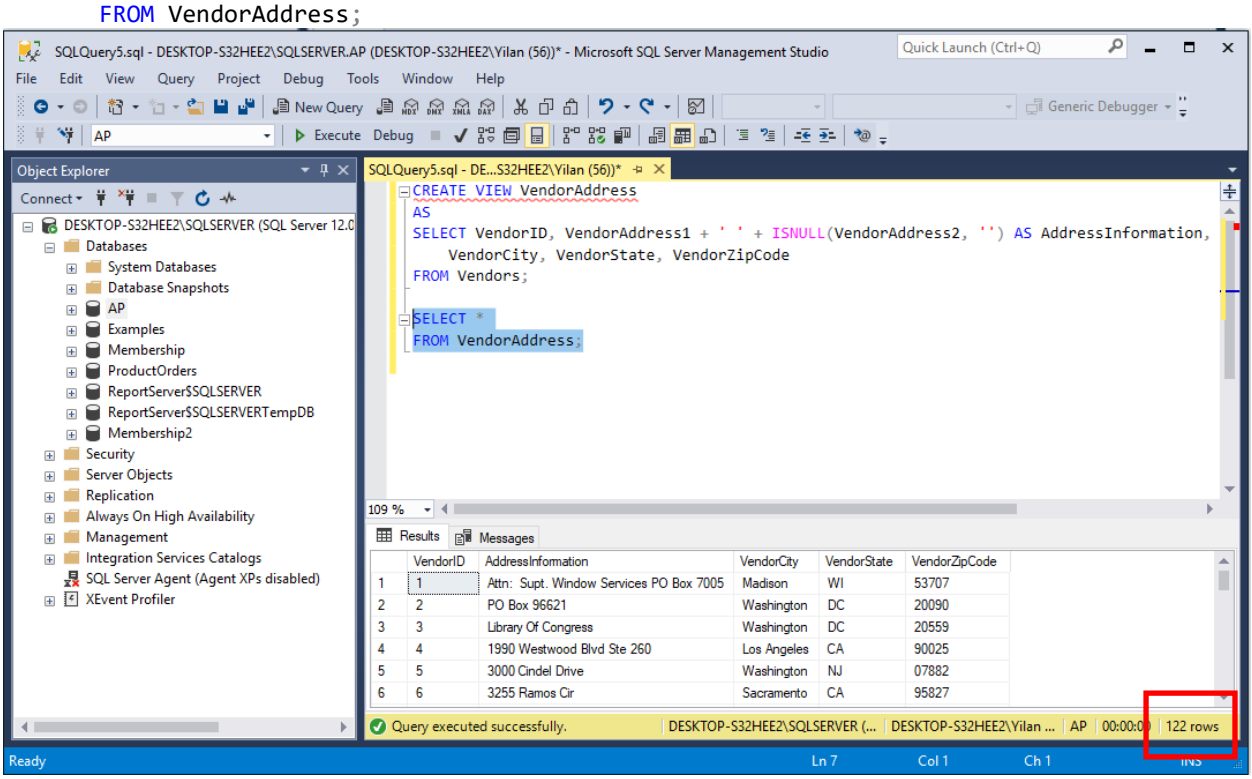


3. Create an updatable views named VendorAddress that returns the VendorID, Address (i.e. VendorAddress1 + ' ' + VendorAddress2), and the city, state, and zip code columns for each vendor. Then write a SELECT query to examine the result set where VendorID=6. Write a SELECT statement to verify the result.

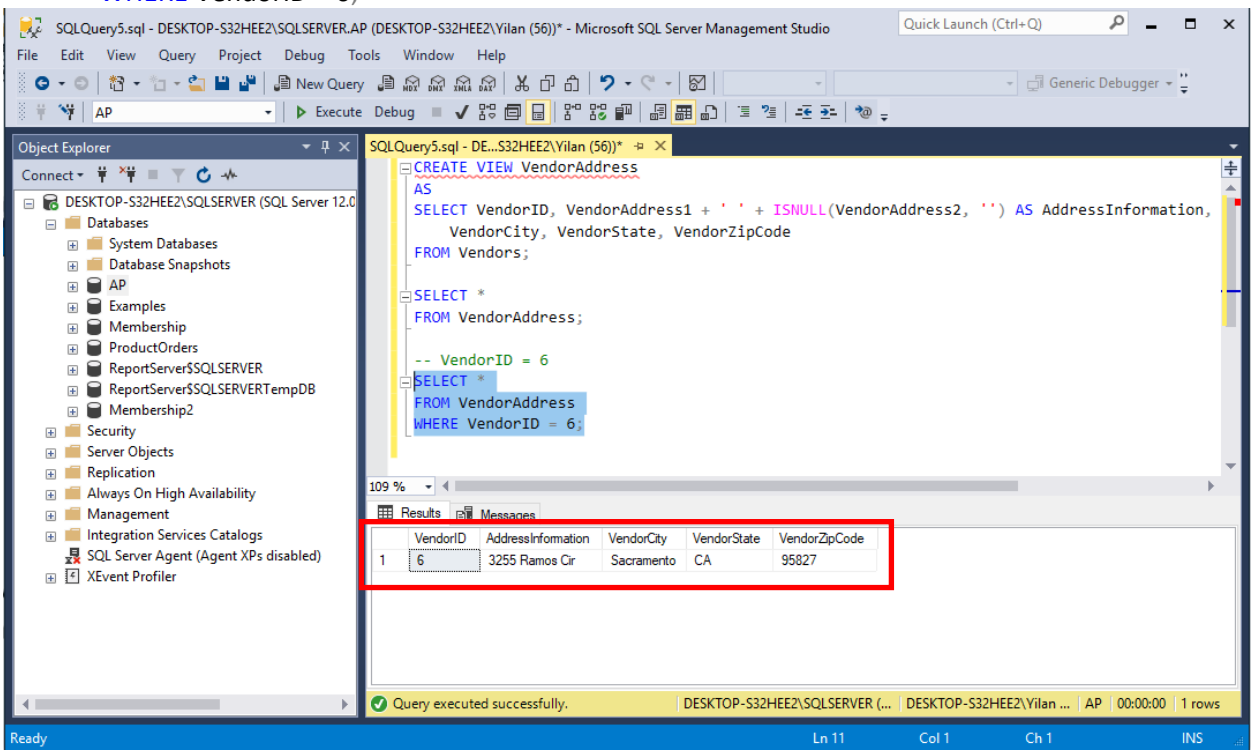
```
CREATE VIEW VendorAddress
AS
SELECT VendorID, VendorAddress1 + ' ' + ISNULL(VendorAddress2, '') AS
AddressInformation, VendorCity, VendorState, VendorZipCode
FROM Vendors;
```



```
SELECT *
FROM VendorAddress;
```

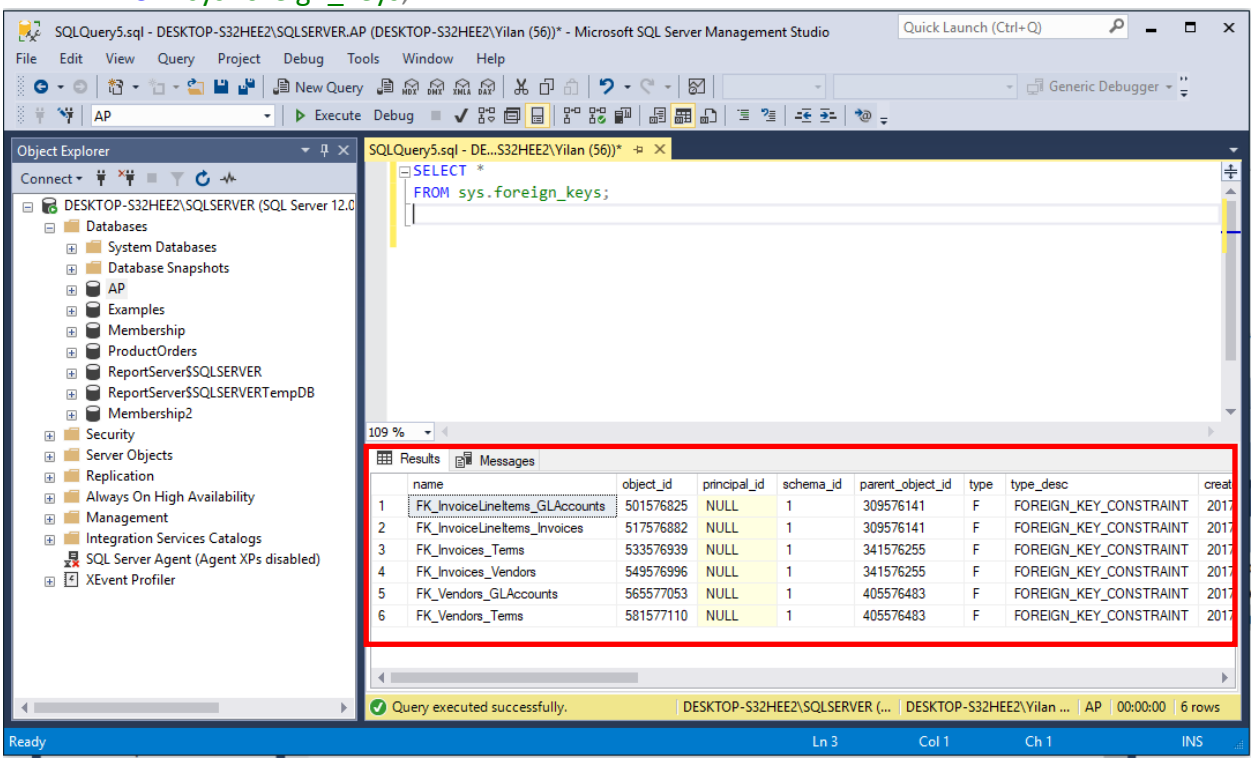



```
SELECT *
FROM VendorAddress
WHERE VendorID = 6;
```



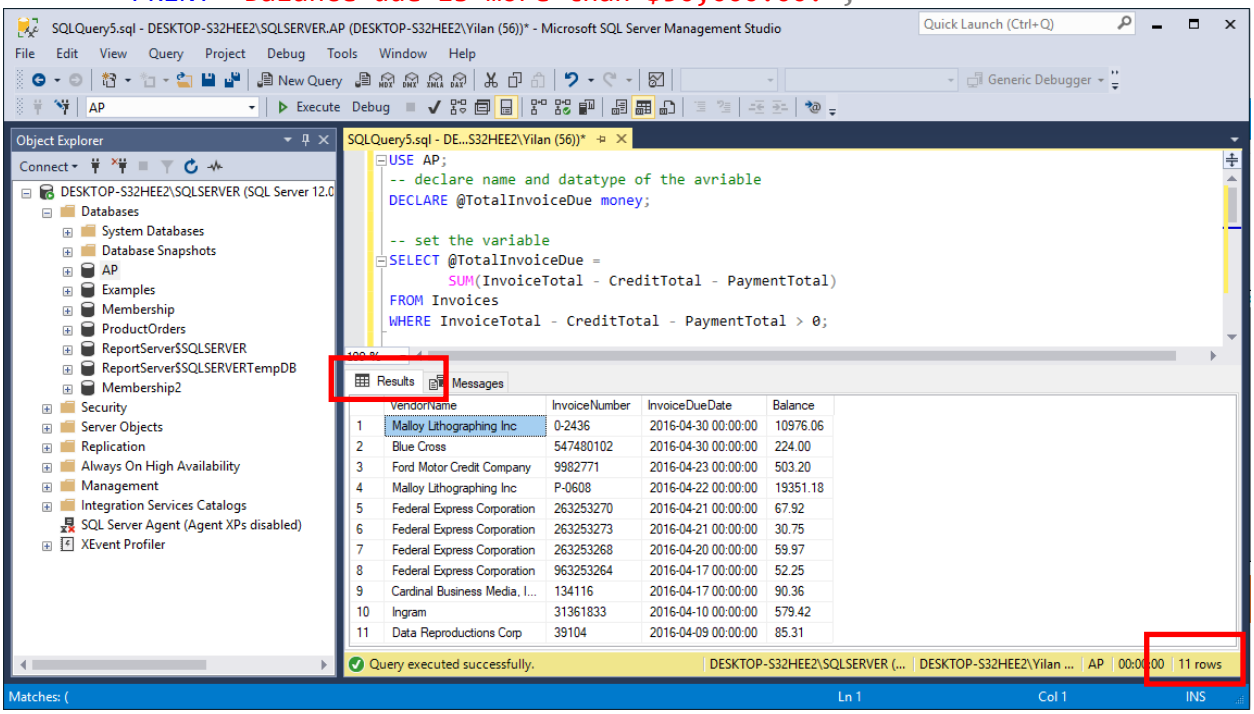
4. Write a SELECT statement that selects all the columns for the catalog view that returns information about foreign keys. How many foreign keys are defined in the AP database and what are they?

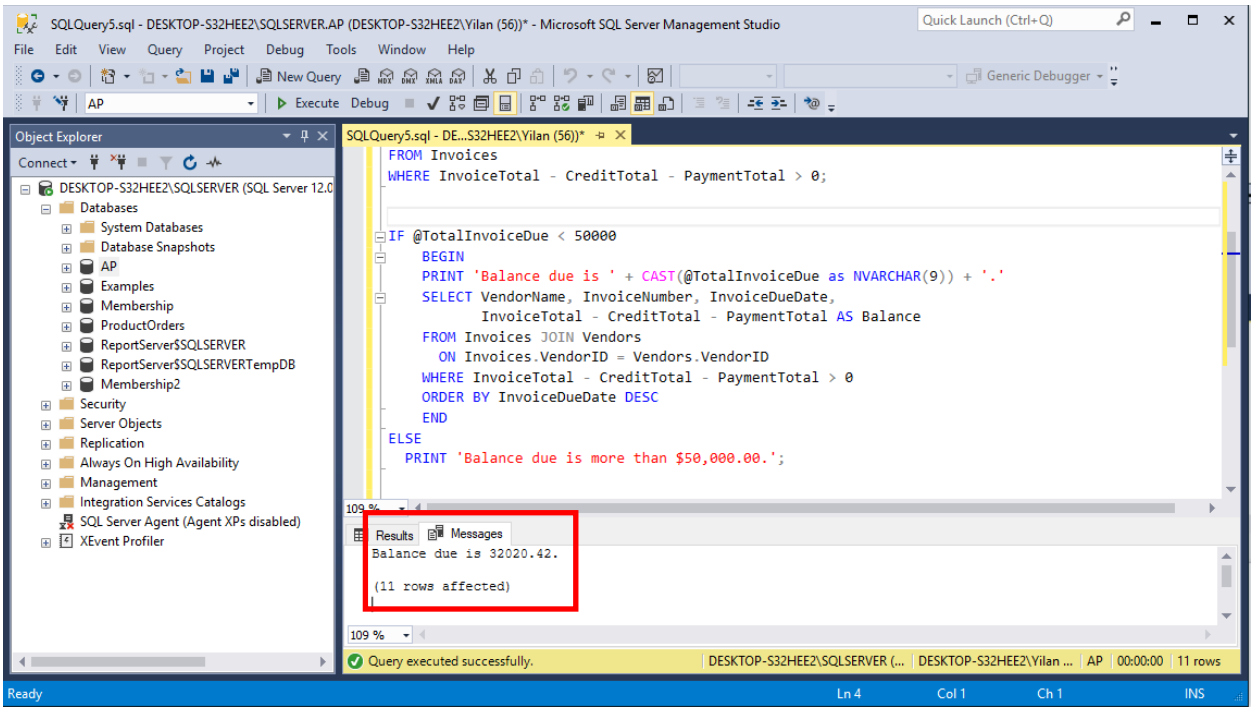
```
SELECT *
FROM sys.foreign_keys;
```



5. Write a script that declares and sets a variable named @TotalBalanceDue, which is equal to the total outstanding balance due. What is the datatype of the variable @TotalBalanceDue? If that balance due is less than $50,000.00, the script should return a result set consisting of VendorName, InvoiceNumber, InvoiceDueDate, and Balance for each invoice with a balance due, sorted with the newest due date first. Then also return the value of @TotalBalanceDue in the format of "Balance due is …". If the total outstanding balance due is more than $50,000.00, the script should return the message "Balance due is more than $50,000.00".

```
USE AP;
-- declare name and datatype of the avriable
DECLARE @TotalInvoiceDue money;
-- set the variable
SELECT @TotalInvoiceDue =
       SUM(InvoiceTotal - CreditTotal - PaymentTotal)
FROM Invoices
WHERE InvoiceTotal - CreditTotal - PaymentTotal > 0;

IF @TotalInvoiceDue < 50000
  BEGIN
  PRINT 'Balance due is ' + CAST(@TotalInvoiceDue as NVARCHAR(9)) + '.'
  SELECT VendorName, InvoiceNumber, InvoiceDueDate,
           InvoiceTotal - CreditTotal - PaymentTotal AS Balance
  FROM Invoices JOIN Vendors
  ON Invoices.VendorID = Vendors.VendorID
  WHERE InvoiceTotal - CreditTotal - PaymentTotal > 0
  ORDER BY InvoiceDueDate DESC
  END
ELSE
  PRINT 'Balance due is more than $50,000.00.';
```

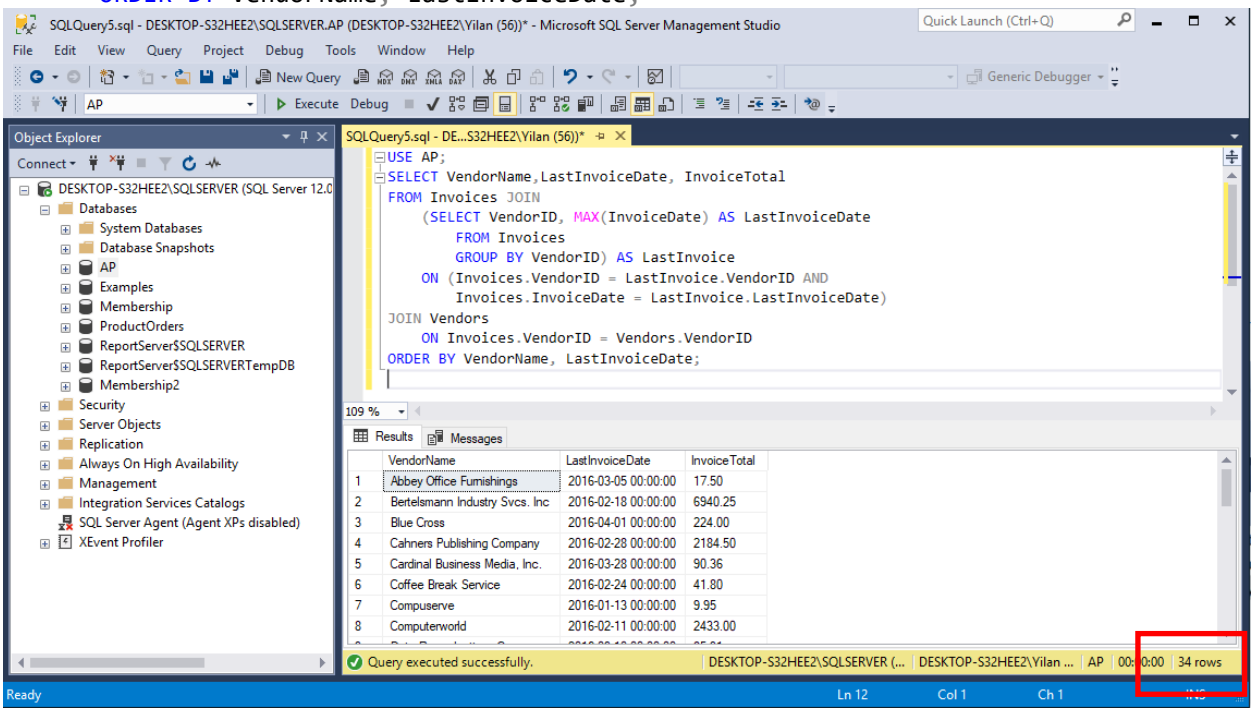



SQLQuery5.sql - DESKTOP-S32HEE2\SQLSERVER.AP (DESKTOP-S32HEE2\Yilan (56))* - Microsoft SQL Server Management Studio
File Edit View Query Project Debug Tools Window Help
New Query
Execute Debug
Object Explorer
Connect
DESKTOP-S32HEE2\SQLSERVER (SQL Server 12.0
Databases
System Databases
Database Snapshots
AP
Examples
Membership
ProductOrders
ReportServer$SQLSERVER
ReportServer$SQLSERVERTempDB
Membership2
Security
Server Objects
Replication
Always On High Availability
Management
Integration Services Catalogs
SQL Server Agent (Agent XPs disabled)
XEvent Profiler
SQLQuery5.sql - DE...S32HEE2\Yilan (56))*
FROM Invoices
WHERE InvoiceTotal - CreditTotal - PaymentTotal > 0;
IF @TotalInvoiceDue < 50000
BEGIN
PRINT 'Balance due is ' + CAST(@TotalInvoiceDue as NVARCHAR(9)) + '.'
SELECT VendorName, InvoiceNumber, InvoiceDueDate,
InvoiceTotal - CreditTotal - PaymentTotal AS Balance
FROM Invoices JOIN Vendors
ON Invoices.VendorID = Vendors.VendorID
WHERE InvoiceTotal - CreditTotal - PaymentTotal > 0
ORDER BY InvoiceDueDate DESC
END
ELSE
PRINT 'Balance due is more than $50,000.00.';
Results
Messages
Balance due is 32020.42.
(11 rows affected)
Query executed successfully.
DESKTOP-S32HEE2\SQLSERVER (...
DESKTOP-S32HEE2\Yilan ...
AP
00:00:00
11 rows
Ready
Ln 4
Col 1
Ch 1
INS

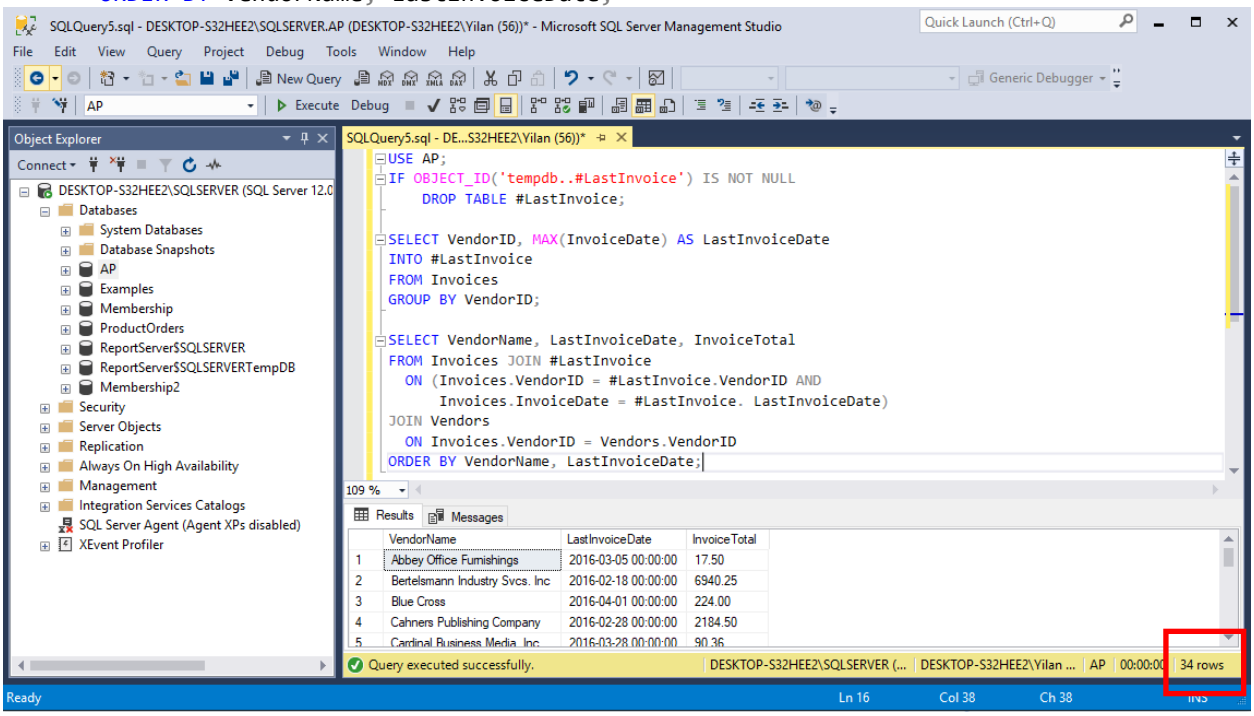
6. Explain the execution result generated by the following script. Then Write a script that generates the same result set but uses a temporary table in place of the derived table. Make sure your script tests for the existence of any objects it creates.

(1) The following script uses a derived table to return the date and invoice total of the latest invoice issued by each vendor.

```
USE AP;
SELECT VendorName,LastInvoiceDate, InvoiceTotal
FROM Invoices JOIN
      (SELECT VendorID, MAX(InvoiceDate) AS LastInvoiceDate
            FROM Invoices
            GROUP BY VendorID) AS LastInvoice
      ON (Invoices.VendorID = LastInvoice.VendorID AND
            Invoices.InvoiceDate = LastInvoice.LastInvoiceDate)
JOIN Vendors
      ON Invoices.VendorID = Vendors.VendorID
ORDER BY VendorName, LastInvoiceDate;
```

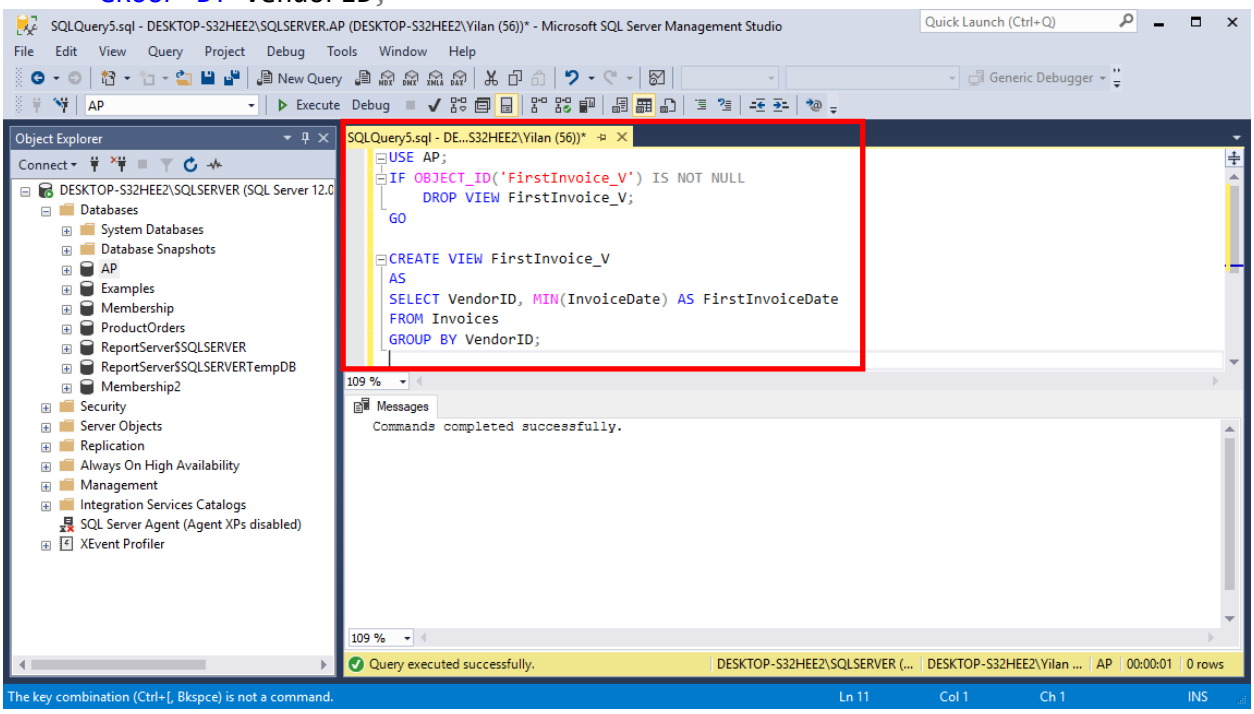


```
USE AP;
IF OBJECT_ID('tempdb..#LastInvoice') IS NOT NULL
    DROP TABLE #LastInvoice;

SELECT VendorID, MAX(InvoiceDate) AS LastInvoiceDate
INTO #LastInvoice
FROM Invoices
GROUP BY VendorID;

SELECT VendorName, LastInvoiceDate, InvoiceTotal
FROM Invoices JOIN #LastInvoice
  ON (Invoices.VendorID = #LastInvoice.VendorID AND
      Invoices.InvoiceDate = #LastInvoice. LastInvoiceDate)
JOIN Vendors
  ON Invoices.VendorID = Vendors.VendorID
ORDER BY VendorName, LastInvoiceDate;
```

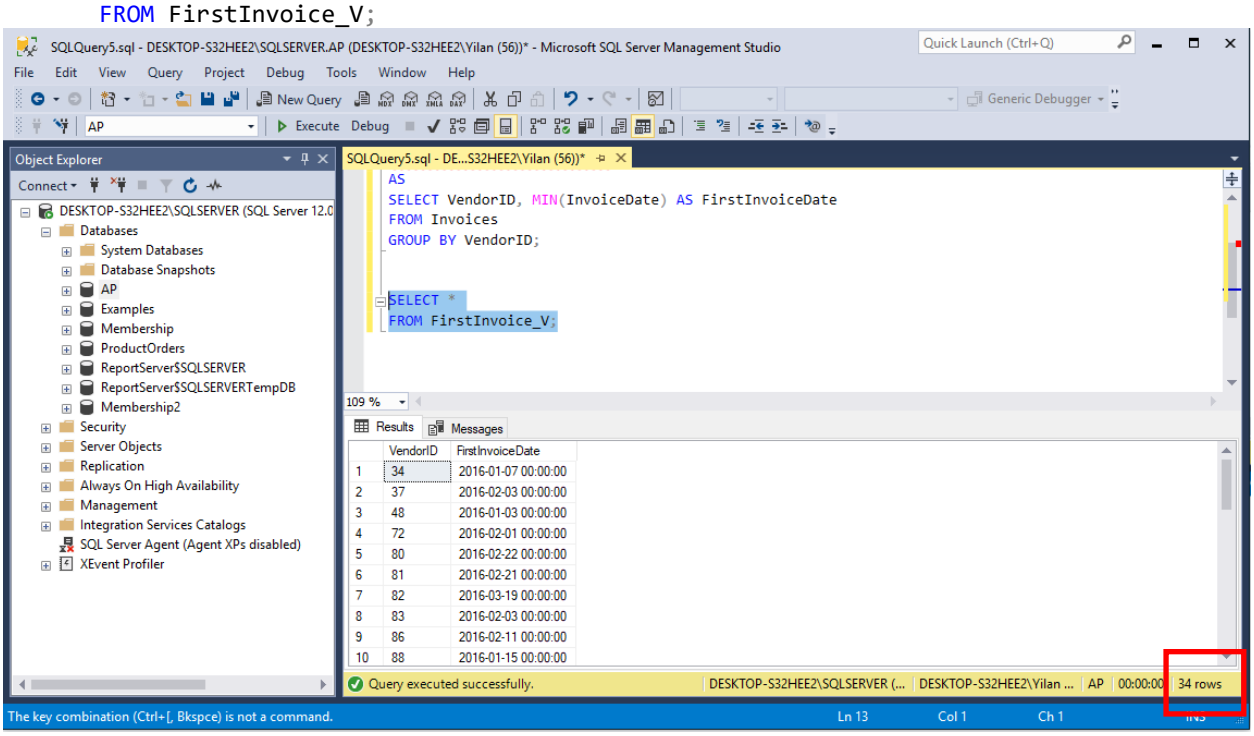


7. Write a script that generates the date and invoice total of the earliest invoice issued by each vendor, using a view instead of a derived table. Also write the script that creates the view, then use SELECT statement to show result of the view. Make sure that your script tests for the existence of the view. The view doesn't need to be redefined each time the script is executed.

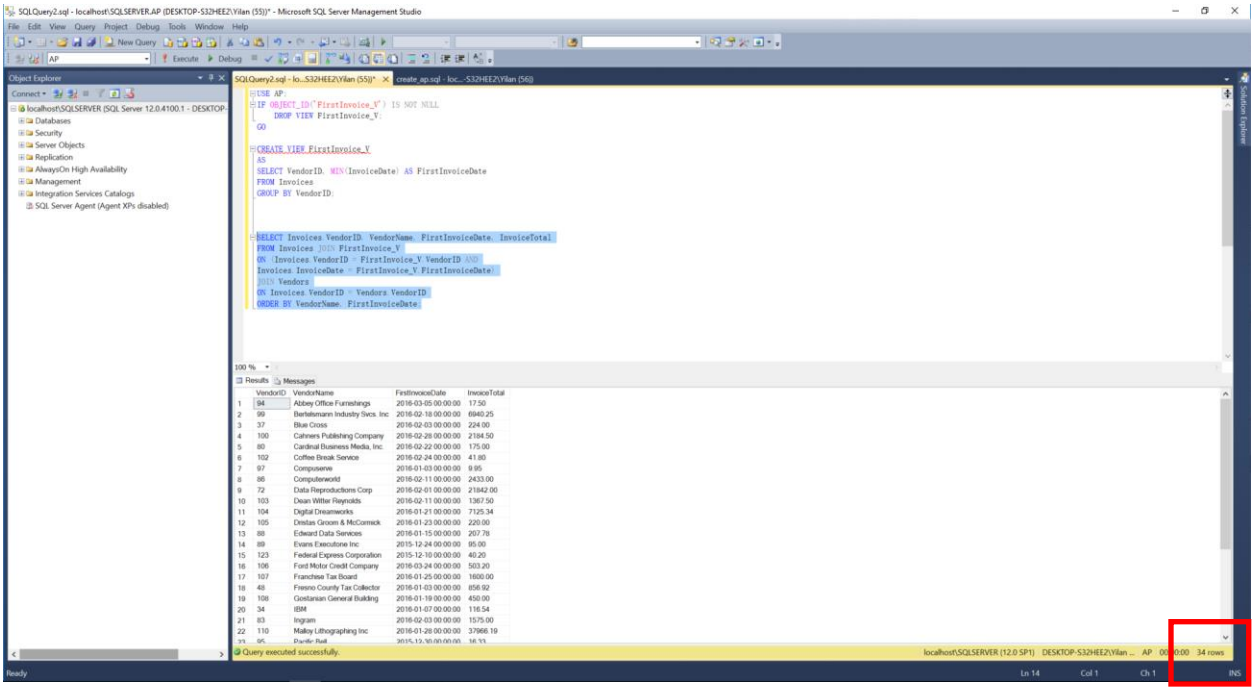
```
USE AP;
IF OBJECT_ID('FirstInvoice_V') IS NOT NULL
    DROP VIEW FirstInvoice_V;
GO

CREATE VIEW FirstInvoice_V
AS
SELECT VendorID, MIN(InvoiceDate) AS FirstInvoiceDate
FROM Invoices
GROUP BY VendorID;
```



```
SELECT *
FROM FirstInvoice_V;
```




```
SELECT Invoices.VendorID, VendorName, FirstInvoiceDate, InvoiceTotal
FROM Invoices JOIN FirstInvoice_V
ON (Invoices.VendorID = FirstInvoice_V.VendorID AND
Invoices.InvoiceDate = FirstInvoice_V.FirstInvoiceDate)
JOIN Vendors
ON Invoices.VendorID = Vendors.VendorID
ORDER BY VendorName, FirstInvoiceDate;
```



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

```
DECLARE @TableName varchar(128);

SELECT @TableName = MIN(name)
FROM sys.tables
WHERE name <> 'dtproperties' AND name <> 'sysdiagrams';

EXEC ('SELECT COUNT(*) AS CountOf' + @TableName +
' FROM ' + @TableName + ';');
```