

1. CREATE VIEW InvoiceBasic AS

```
SELECT VendorName, InvoiceTotal  
FROM AP.dbo.Vendors JOIN AP.dbo.Invoices  
ON Vendors.VendorID = Invoices.VendorID
```

```
GO
```

```
SELECT *  
FROM InvoiceBasic  
WHERE VendorName LIKE '[M-O]%'  
ORDER BY InvoiceTotal DESC;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including the master database and various tables like AP, Examples, lab6, ProductOrders, University, and University2. The central pane displays the script for creating the view:

```
--create a view named InvoiceBasic using CREATE VIEW...AS statement,  
--then select the two columns from the two tables where their vendorIDs match.  
--  
CREATE VIEW InvoiceBasic AS  
    SELECT VendorName, InvoiceTotal  
    FROM AP.dbo.Vendors JOIN AP.dbo.Invoices  
    ON Vendors.VendorID = Invoices.VendorID  
GO  
  
--select all columns using SELECT *,  
--from the view we just created InvoiceBasic,  
--starting with the letters M to O inclusive (M,N,O) and any letters after that using LIKE statement  
--then order in the descending order using DESC.  
--  
SELECT *  
FROM InvoiceBasic  
WHERE VendorName LIKE '[M-O]%'  
ORDER BY InvoiceTotal DESC;
```

The results pane on the right shows the output of the query:

| VendorName | InvoiceTotal |
|--------------------------|--------------|
| Maloy Lithographing Inc. | 37966.19 |
| Maloy Lithographing Inc. | 26881.40 |
| Maloy Lithographing Inc. | 23517.58 |
| Maloy Lithographing Inc. | 20551.18 |
| Maloy Lithographing Inc. | 10976.06 |

At the bottom, a message bar indicates "Query executed successfully." and shows the session details: DESKTOP-UP37904\SQLEXPRESS ... | DESKTOP-UP37904\owner ... | master | 00:00:00 | 5 rows.

2. CREATE VIEW Top10PaidInvoices AS

```

SELECT TOP 10
    VendorName,
    MIN(InvoiceDate) AS FirstInvoice,
    SUM(InvoiceTotal) AS SumOfInvoices
FROM AP.dbo.Vendors JOIN AP.dbo.Invoices
ON Vendors.VendorID = Invoices.VendorID
WHERE InvoiceTotal - CreditTotal - PaymentTotal <> 0
GROUP BY VendorName
ORDER BY SUM(InvoiceTotal) DESC

```

GO

SELECT * FROM Top10PaidInvoices;

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including the AP schema which contains tables like Vendors and Invoices. The central pane displays the SQL code for creating the view and executing it. The results pane shows a table with 7 rows of vendor information.

```

/*create a view named Top10PaidInvoices using CREATE VIEW...AS statement,
select 10 using SELECT TOP 10,
the least invoice date using MIN(InvoiceDate) and sum of invoices using SUM(InvoiceTotal).
Again, the VendorIDs has to match from the two tables.
The unpaid invoices determined by the total due being non-zero, condition set in WHERE clause,
and finally GROUP BY the vendor names and ORDER BY the top 10 of SumOfInvoices from highest to lowest using DESC
*/
CREATE VIEW Top10PaidInvoices AS
SELECT TOP 10
    VendorName,
    MIN(InvoiceDate) AS FirstInvoice,
    SUM(InvoiceTotal) AS SumOfInvoices
FROM AP.dbo.Vendors JOIN AP.dbo.Invoices
ON Vendors.VendorID = Invoices.VendorID
WHERE InvoiceTotal - CreditTotal - PaymentTotal <> 0
GROUP BY VendorName
ORDER BY SUM(InvoiceTotal) DESC
GO

--Select everthing from the view we just created.
SELECT * FROM Top10PaidInvoices;

```

| VendorName | FirstInvoice | SumOfInvoices |
|-------------------------------|---------------------|---------------|
| Malloy Lithographing Inc | 2016-03-23 00:00:00 | 31527.24 |
| Ingram | 2016-03-21 00:00:00 | 579.42 |
| Ford Motor Credit Company | 2016-03-24 00:00:00 | 503.20 |
| Blue Cross | 2016-04-01 00:00:00 | 224.00 |
| Federal Express Corporation | 2016-03-18 00:00:00 | 210.89 |
| Cardinal Business Media, Inc. | 2016-03-28 00:00:00 | 90.36 |
| Data Reproductions Corp | 2016-03-10 00:00:00 | 85.31 |

There are only 7 such vendors.

3. CREATE VIEW VendorAddress AS

```

SELECT VendorID,
       (ISNULL(VendorAddress1,'')+' '+ISNULL(VendorAddress2,'')+', '+VendorCity+
       '+VendorState+', '+VendorZipCode) AS Address
  FROM AP.dbo.Vendors

```

GO

```

SELECT *
  FROM VendorAddress
 WHERE VendorID = 10;

```

```

SELECT *
  FROM VendorAddress;

```

```

/*
create a view named VendorAddress using CREATE VIEW...AS statement,
then select the VendorID and its address.
The address is concatenated with VendorAddress1, VendorAddress2, VendorCity, VendorState, VendorZipCode
If the vendor address 1 or 2 is null, it's replaced by a space because the data in these two columns can be null.
*/
CREATE VIEW VendorAddress AS
    SELECT VendorID,
           (ISNULL(VendorAddress1,'')+' '+ISNULL(VendorAddress2,'')+', '+VendorCity+
           '+VendorState+', '+VendorZipCode) AS Address
    FROM AP.dbo.Vendors
GO

--get the result where vendor ID is 10
SELECT *
  FROM VendorAddress
 WHERE VendorID = 10;

--get everything from the view we just created.
SELECT *
  FROM VendorAddress;

```

| VendorID | Address |
|----------|--|
| 1 | 4669 N Fresno , Fresno CA, 93726 |
| 2 | Attn. Supt. Window Services PO Box 7005, Madison WI, 53707 |
| 3 | PO Box 95621 , Washington DC, 20090 |
| 4 | Library Of Congress , Washington DC, 20559 |
| 4 | 1990 Westwood Blvd Ste 260 , Los Angeles CA, 90025 |
| 5 | 3000 Cindel Drive , Washington NJ, 07828 |
| 6 | 3255 Ramos Ct Sacramento CA, 95827 |
| 7 | Kevin Minder 3441 W Macarthur Blvd. Santa Ana CA, 92704 |
| 8 | PO Box 9369 , Fresno CA, 93792 |
| 9 | Box 52001 , San Francisco CA, 94152 |
| 10 | 4669 N Fresno , Fresno CA, 93726 |
| 11 | 4581 E Home , Fresno CA, 93703 |
| 12 | PO Box 2069 , Fresno CA, 93718 |
| 13 | PO Box 85826 , San Diego CA, 92186 |
| 14 | 4420 N. First Street, Suite 108 , Fresno CA, 93726 |

Query executed successfully.

In this case, (before you announced the changes to the question) there is a vendor who has both address 1 and 2 are NULL. If you right click Vendors table and select Design, you can see that both Address 1 and 2 allow NULL values, so whenever address 2 is NULL replace with space, I also did the same with address 1. Therefore, I did not rewrite the code. Hope you can understand my point here.

4. USE Examples;

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

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including the EXAMPLES database and its tables like AP, Customers, Departments, Employees, etc. The Results tab on the right displays the output of the query:

| name | object_id | principal_id | schema_id | parent_object_id | type | type_desc | create_date | modify_date | is_ms_shipped | is_published | is_trusted |
|--------------------------|------------|--------------|-----------|------------------|------|------------------------|-------------------------|-------------------------|---------------|--------------|------------|
| FK_SalesTotals_SalesReps | 1445580188 | NULL | 1 | 1221579390 | F | FOREIGN_KEY_CONSTRAINT | 2020-01-19 20:11:52.880 | 2020-01-19 20:11:52.880 | 0 | 0 | 0 |

Query executed successfully.

One foreign key defined called RepID that relates SalesReps table(PK) -> SalesTotals table(FK).

5. USE AP;

```
DECLARE @TotalBalanceDue money;

SET @TotalBalanceDue =
    (SELECT SUM(InvoiceTotal - CreditTotal - PaymentTotal)
     FROM Invoices
     WHERE (InvoiceTotal - CreditTotal - PaymentTotal) > 0);

IF @TotalBalanceDue > 40000
    PRINT 'Balance due is more than $40,000.00'
ELSE
    BEGIN
        PRINT 'Balance due is $'+ CAST(@TotalBalanceDue AS varchar(10))
        SELECT      VendorName,
                    InvoiceNumber,
                    InvoiceDueDate,
                    (InvoiceTotal - CreditTotal - PaymentTotal) AS Balance
        FROM Vendors JOIN Invoices
            ON Vendors.VendorID = Invoices.VendorID
        WHERE (InvoiceTotal - CreditTotal - PaymentTotal) > 0
        ORDER BY InvoiceDueDate ASC
    END;
```

lab8no5.sql - DESKTOP-UP37904\SQLEXPRESS.AP [DESKTOP-UP37904\owner (56)] - Microsoft SQL Server Management Studio

```

File Edit View Query Project Tools Window Help
Connect AP New Query undo
Execute ✓ Run Stop Refresh Stop All Stop All
Object Explorer
Connect AP
DESKTOP-UP37904\SQLEXPRESS (SQL Server 14.0.10)
Databases System Databases Database Snapshots AP
Database Diagrams Tables System Tables FileTables External Tables Graph Tables
dbo.ContactUpdates dbo.GLAccounts dbo.invoiceArchive dbo.invoiceLineItems dbo.invoices
dbo.invoices
    Columns
        InvoiceID (PK, int, not null)
        VendorID (FK, int, not null)
        InvoiceNumber (varchar(50), r
        InvoiceDate (smalldatetime, n
        InvoiceTotal (money, not null)
        PaymentTotal (money, not nu
        CreditTotal (money, not null)
        TermID (FK, int, not null)
        InvoiceDueDate (smalldatetim
        PaymentDate (smalldatetim
    Keys
    Constraints
    Triggers
    Indexes
    Statistics
    dbo.invoicesCopy
    dbo.Terms
    dbo.Vendors
    dbo.VendorsCopy
Views External Resources Synonyms Programmability Service Broker Storage Security Examples lab6 ProductOrders

```

lab8no5.sql - DESK...P37904\owner (56) lab8no4.sql - DESK...P37904\owner (55) lab8no3.sql - DESK...P37904\owner (54)

```

1 --Lichen Liang
2
3 USE AP;
4
5 --set datatype of TotalBalanceDue to money, using DECLARE
6 DECLARE @TotalBalanceDue money;
7
8 --set the TotalBalanceDue to be the sum of balance due, using SET.
9 --then use subquery to find the sum of balance due where balance due is greater than 0
10 SET @TotalBalanceDue =
11     (SELECT SUMInvoiceTotal - CreditTotal - PaymentTotal)
12     FROM Invoices
13     WHERE (InvoiceTotal - CreditTotal - PaymentTotal) > 0;
14
15 /*if total balance due is more than 40000, print the message using PRINT.
16 Otherwise(total balance due below 40000), use BEGIN clause to start
17 print the message concatenated with total balance due, which is casted into varchar datatype.
18 Also return the four columns VendorName, InvoiceNumber, InvoiceDueDate, and Balance,
19 where balance should be greater than 0 and the vendor IDs should match from the two tables.
20 Order by the oldest due date first(in ASC).
21 and END clause to stop
22 */
23 IF @TotalBalanceDue > 40000
24     PRINT 'Balance due is more than $40,000.00'
25 ELSE
26     BEGIN
27         PRINT 'Balance due is '+ CAST(@TotalBalanceDue AS varchar(10))
28         SELECT VendorName,
29             InvoiceNumber,
30             InvoiceDueDate,
31             (InvoiceTotal - CreditTotal - PaymentTotal) AS Balance
32         FROM Vendors JOIN Invoices
33             ON Vendors.VendorID = Invoices.VendorID
34             WHERE (InvoiceTotal - CreditTotal - PaymentTotal) > 0
35             ORDER BY InvoiceDueDate ASC
36     END;
37

```

Results Messages

| VendorName | InvoiceNumber | InvoiceDueDate | Balance |
|-------------------------------|---------------|---------------------|----------|
| Data Reproductions Corp | 39104 | 2016-04-09 00:00:00 | 85.31 |
| Ingram | 31516133 | 2016-04-10 00:00:00 | 579.42 |
| Federal Express Corporation | 96325264 | 2016-04-17 00:00:00 | 52.25 |
| Cardinal Business Media, Inc. | 134116 | 2016-04-17 00:00:00 | 90.36 |
| Federal Express Corporation | 263252368 | 2016-04-20 00:00:00 | 59.97 |
| Federal Express Corporation | 263252370 | 2016-04-21 00:00:00 | 67.92 |
| Federal Express Corporation | 263252373 | 2016-04-21 00:00:00 | 30.75 |
| Malloy Lithographing Inc | P-0608 | 2016-04-22 00:00:00 | 19351.18 |
| Ford Motor Credit Company | 9982771 | 2016-04-23 00:00:00 | 503.20 |
| Malloy Lithographing Inc | O-2436 | 2016-04-30 00:00:00 | 10976.06 |

Query executed successfully. DESKTOP-UP37904\SQLEXPRESS... DESKTOP-UP37904\owner... AP 00:00:00 | 11 rows

Ready

lab8no5.sql - DESKTOP-UP37904\SQLEXPRESS.AP [DESKTOP-UP37904\owner (56)] - Microsoft SQL Server Management Studio

```

File Edit View Query Project Tools Window Help
Connect AP New Query undo
Execute ✓ Run Stop Refresh Stop All Stop All
Object Explorer
Connect AP
DESKTOP-UP37904\SQLEXPRESS (SQL Server 14.0.10)
Databases System Databases Database Snapshots AP
Database Diagrams Tables System Tables FileTables External Tables Graph Tables
dbo.ContactUpdates dbo.GLAccounts dbo.invoiceArchive dbo.invoiceLineItems dbo.invoices
dbo.invoices
    Columns
        InvoiceID (PK, int, not null)
        VendorID (FK, int, not null)
        InvoiceNumber (varchar(50), r
        InvoiceDate (smalldatetime, n
        InvoiceTotal (money, not null)
        PaymentTotal (money, not nu
        CreditTotal (money, not null)
        TermID (FK, int, not null)
        InvoiceDueDate (smalldatetim
        PaymentDate (smalldatetim
    Keys
    Constraints
    Triggers
    Indexes
    Statistics
    dbo.invoicesCopy
    dbo.Terms
    dbo.Vendors
    dbo.VendorsCopy
Views External Resources Synonyms Programmability Service Broker Storage Security Examples lab6 ProductOrders

```

lab8no5.sql - DESK...P37904\owner (56) lab8no4.sql - DESK...P37904\owner (55) lab8no3.sql - DESK...P37904\owner (54)

```

1 --Lichen Liang
2
3 USE AP;
4
5 --set datatype of TotalBalanceDue to money, using DECLARE
6 DECLARE @TotalBalanceDue money;
7
8 --set the TotalBalanceDue to be the sum of balance due, using SET.
9 --then use subquery to find the sum of balance due where balance due is greater than 0
10 SET @TotalBalanceDue =
11     (SELECT SUMInvoiceTotal - CreditTotal - PaymentTotal)
12     FROM Invoices
13     WHERE (InvoiceTotal - CreditTotal - PaymentTotal) > 0;
14
15 /*if total balance due is more than 40000, print the message using PRINT.
16 Otherwise(total balance due below 40000), use BEGIN clause to start
17 print the message concatenated with total balance due, which is casted into varchar datatype.
18 Also return the four columns VendorName, InvoiceNumber, InvoiceDueDate, and Balance,
19 where balance should be greater than 0 and the vendor IDs should match from the two tables.
20 Order by the oldest due date first(in ASC).
21 and END clause to stop
22 */
23 IF @TotalBalanceDue > 40000
24     PRINT 'Balance due is more than $40,000.00'
25 ELSE
26     BEGIN
27         PRINT 'Balance due is '+ CAST(@TotalBalanceDue AS varchar(10))
28         SELECT VendorName,
29             InvoiceNumber,
30             InvoiceDueDate,
31             (InvoiceTotal - CreditTotal - PaymentTotal) AS Balance
32         FROM Vendors JOIN Invoices
33             ON Vendors.VendorID = Invoices.VendorID
34             WHERE (InvoiceTotal - CreditTotal - PaymentTotal) > 0
35             ORDER BY InvoiceDueDate ASC
36     END;
37

```

Results Messages

Balance due is \$32020.42
(11 rows affected)

Completion time: 2020-03-16T03:29:33.2399570-04:00

Query executed successfully. DESKTOP-UP37904\SQLEXPRESS... DESKTOP-UP37904\owner... AP 00:00:00 | 11 rows

Item(s) Saved

Datatype for @TotalBalanceDue is money

6. USE AP;

```
IF OBJECT_ID('tempdb..#FirstInvoice')IS NOT NULL
DROP TABLE #FirstInvoice;
```

```
SELECT VendorID, MIN(InvoiceDate) AS FirstInvoiceDate
INTO #FirstInvoice
FROM Invoices
GROUP BY VendorID;
```

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

```

-->Lichen Liang
1
2
3 /*In AP database,
4 recreates the Invoices, FirstInvoice, and Vendors table from three tables:
5 Invoices, Vendors, and a temporary table from the subject FirstInvoice.
6 The FirstInvoice table contains VendorID and its oldest invoice date called FirstInvoiceDate.
7 VendorID and InvoiceDate from Invoices table should match VendorID and FirstInvoiceDate from FirstInvoice table.
8 VendorID between Invoices and Vendors should also match.
9 Finally, order by the vendor name and their oldest invoice date.
*/
10
11 USE AP;
12
13 --checks if the temporary table exists, if so, delete it so it can be recreated again
14 IF OBJECT_ID('tempdb..#FirstInvoice')IS NOT NULL
15   DROP TABLE #FirstInvoice;
16
17 --use SELECT...INTO to create the temporary table FirstInvoice. # denotes a temp table.
18 Then select the three columns we want to return,
19 Join the three tables, including the temporary table, and the rest are the same.
20
21
22 SELECT VendorID, MIN(InvoiceDate) AS FirstInvoiceDate
23 INTO #FirstInvoice
24 FROM Invoices
25 GROUP BY VendorID;
26
27 SELECT VendorName, FirstInvoiceDate, InvoiceTotal
28 FROM Invoices JOIN #FirstInvoice
29 ON ((Invoices.VendorID = #FirstInvoice.VendorID) AND (Invoices.InvoiceDate = #FirstInvoice.FirstInvoiceDate))
30 JOIN Vendors
31 ON Invoices.VendorID = Vendors.VendorID
32 ORDER BY VendorName, FirstInvoiceDate;
```

Results

| VendorName | FirstInvoiceDate | InvoiceTotal |
|-------------------------------|---------------------|--------------|
| Abbey Office Furnishings | 2016-03-05 00:00:00 | 17.50 |
| Betelmann Industry Svcs. Inc | 2016-02-18 00:00:00 | 6940.25 |
| Blue Cross | 2016-02-03 00:00:00 | 224.00 |
| Cahners Publishing Company | 2016-02-28 00:00:00 | 2184.50 |
| Cardinal Business Media, Inc. | 2016-02-22 00:00:00 | 175.00 |
| Coffee Break Service | 2016-02-24 00:00:00 | 41.80 |
| Compuserve | 2016-02-03 00:00:00 | 9.95 |
| Computerworld | 2016-02-11 00:00:00 | 2433.00 |
| Data Reproductions Corp | 2016-02-01 00:00:00 | 21842.00 |
| Dean Witter Reynolds | 2016-02-11 00:00:00 | 1367.50 |
| Digital Dreamworks | 2016-01-21 00:00:00 | 7125.34 |
| Dixies Groton & McCormick | 2016-01-23 00:00:00 | 220.00 |
| Edward Data Services | 2016-01-15 00:00:00 | 207.78 |

Query executed successfully.

Explanation see comment in screenshot.

7. USE AP;

IF OBJECT_ID('LatestInvoice')IS NOT NULL

```
DROP VIEW LatestInvoice;
```

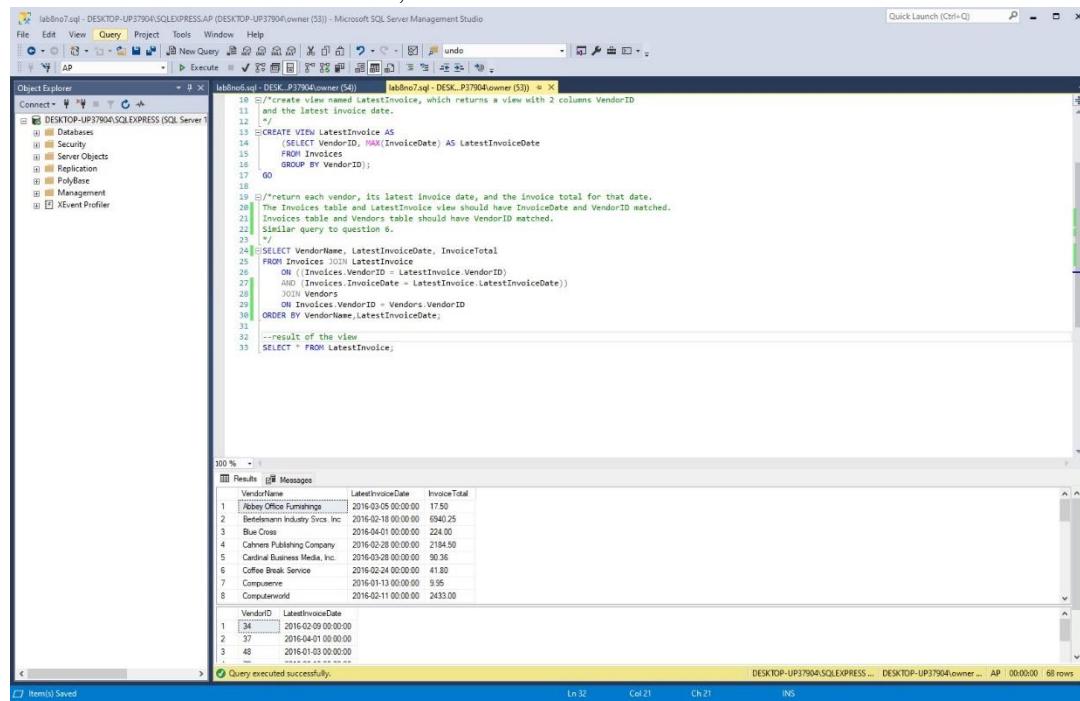
GO

```
CREATE VIEW LatestInvoice AS  
    (SELECT VendorID, MAX(InvoiceDate) AS LatestInvoiceDate  
     FROM Invoices  
     GROUP BY VendorID);
```

GO

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

```
SELECT * FROM LatestInvoice;
```



8. USE ProductOrders;

```
DECLARE @TableName varchar(100);
```

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

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

The screenshot shows the Microsoft SQL Server Management Studio interface. On the left, the Object Explorer tree view shows the database structure for 'DESKTOP-UP37904\SQLEXPRESS'. In the center, a query window displays the following T-SQL script:

```
--Lichen Liang  
1 USE ProductOrders;  
2  
3 --set the datatype of TableName to varchar using DECLARE  
4 DECLARE @TableName varchar(100);  
5  
6 /*set the TableName to be the first table in the alphabetical order in the ProductOrders database using MIN(name)  
7 and the name cannot be sysdiagrams or dtproperties  
8 */  
9 SET @TableName =  
10 (SELECT MIN(name)  
11 FROM ProductOrders.sys.tables  
12 WHERE name <> 'sysdiagrams' AND name <> 'dtproperties');  
13  
14 /*use the EXEC to execute the query,  
15 then returns the number of rows in the first table in the database,  
16 where @TableName is dynamic and can be changed from above SET statement.  
17 */  
18  
19 EXEC('SELECT COUNT(*) AS CountOf' + @TableName + ' FROM ' + @TableName + '');
```

The 'Results' tab shows the output of the query:

| CountOfCustomers |
|------------------|
| 25 |

At the bottom of the screen, the status bar indicates: 'Query executed successfully.' and 'Ln 21 Col 1 Ch 1 INS'.

Remarks

In this lab we practiced how to use views and write scripts. Also practiced using DDL, catalog views, statements for controlling the flow of execution, etc. I think this is a very good practice for the lecture. A more complicated challenge would be increasing the requirement in the question, such as more tables, columns, conditions, and so on.