

1. Select CustCity, CustState, CustZip, CustFax
FROM ProductOrders.dbo.Customers;

```
SQLQuery1.sql - DESKTOP-UP37904\SQLEXPRESS\ProductOrders (DESKTOP-UP37904\owner (53)) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
Object Explorer
Connect - Lichen Liang
DESKTOP-UP37904\SQLEXPRESS (SQL Server 14.0.1000 - DESKTO...
    Database
        System Databases
        Database Snapshots
        AP
        Examples
        ProductOrders
            Database Diagrams
            Tables
                System Tables
                FileTables
                External Tables
                Graph Tables
            dbo.Customers
                Columns
                    CustID (PK, int, not null)
                    CustFirstName (nvarchar(50), null)
                    CustLastName (nvarchar(50), not null)
                    CustAddress (nvarchar(255), not null)
                    CustCity (nvarchar(50), not null)
                    CustState (nvarchar(20), not null)
                    CustZip (nvarchar(20), not null)
                    CustPhone (nvarchar(30), not null)
                    CustFax (nvarchar(30), null)
                Keys
                Constraints
                Triggers
                Indexes
                Statistics
            dbo.Items
            dbo.OrderDetails
            dbo.Orders
        Views
        External Resources
        Synonyms
        Programmability
        Service Broker
        Storage
        Security
    Security
    System Objects
    Ready
SQLQuery1.sql - DESKTOP-UP37904\owner (53)* - X
--Lichen Liang
Select CustCity, CustState, CustZip, CustFax
FROM ProductOrders.dbo.Customers;
Results Messages
CustCity CustState CustZip CustFax
1 Columbus OH 43221 6145553928
2 Madison WI 53707 2095552262
3 New York NY 10010 NULL
4 Washington DC 20006 NULL
5 Cleves OH 45002 NULL
6 Cincinnati OH 45225 8005552826
7 Fairfield IA 52556 NULL
8 Fresno CA 93728 NULL
9 Los Angeles CA 90004 NULL
10 Valencia CA 91355 8055556689
11 Sacramento CA 95887 2095551302
12 Takoma Park MD 24512 NULL
13 Fresno CA 93711 NULL
14 Farfield NJ 07004 NULL
15 Orange CA 92807 NULL
16 Los Angeles CA 90008 NULL
17 Duluth GA 30136 NULL
Query executed successfully.
DESKTOP-UP37904\SQLEXPRESS ... DESKTOP-UP37904\owner ... ProductOrders 00:00:00 25 rows
Ln 4 Col 1 Ch 1 INS
```

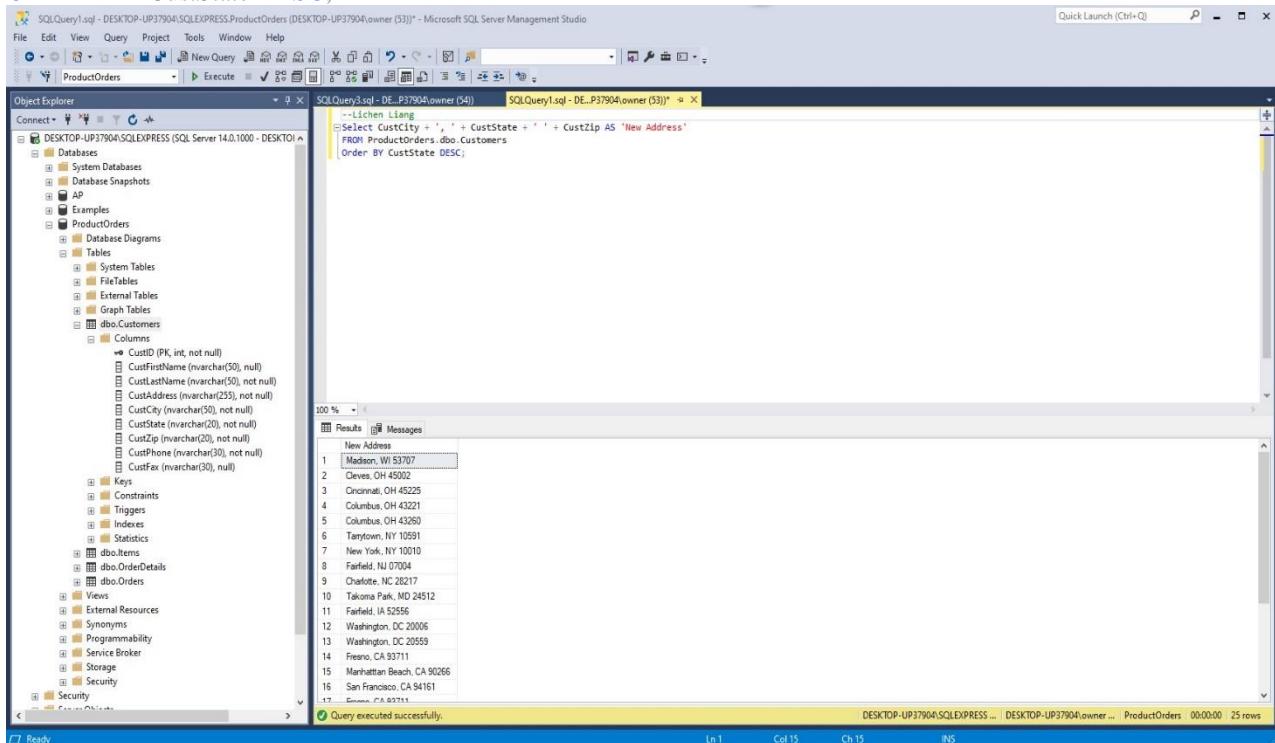
Select four columns from the table.

2. Select CustFirstName + '' + CustLastName AS Name, CustState AS State
FROM ProductOrders.dbo.Customers
WHERE CustState LIKE '[^A-C]%' ;

```
SQLQuery2.sql - DESKTOP-UP37904\SQLEXPRESS\ProductOrders (DESKTOP-UP37904\owner (54)) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
Object Explorer
Connect - Lichen Liang
DESKTOP-UP37904\SQLEXPRESS (SQL Server 14.0.1000 - DESKTO...
    Database
        System Databases
        Database Snapshots
        AP
        Examples
        ProductOrders
            Database Diagrams
            Tables
                System Tables
                FileTables
                External Tables
                Graph Tables
            dbo.Customers
                Columns
                    CustID (PK, int, not null)
                    CustFirstName (nvarchar(50), null)
                    CustLastName (nvarchar(50), not null)
                    CustAddress (nvarchar(255), not null)
                    CustCity (nvarchar(50), not null)
                    CustState (nvarchar(20), not null)
                    CustZip (nvarchar(20), not null)
                    CustPhone (nvarchar(30), not null)
                    CustFax (nvarchar(30), null)
                Keys
                Constraints
                Triggers
                Indexes
                Statistics
            dbo.Items
            dbo.OrderDetails
            dbo.Orders
        Views
        External Resources
        Synonyms
        Programmability
        Service Broker
        Storage
        Security
    Security
    System Objects
    Ready
SQLQuery2.sql - DESKTOP-UP37904\owner (54)* - X
--Lichen Liang
Select CustFirstName + '' + CustLastName AS Name, CustState AS State
FROM ProductOrders.dbo.Customers
WHERE CustState LIKE '[^A-C]%' ;
Results Messages
Name State
1 Korin Blanca OH
2 Yash Randal WI
3 Johnathan Milleron NY
4 Mikayla Danion DC
5 Kendall Mayne OH
6 Kathlin Hootley OH
7 Derek Chadwick IA
8 Anden Rohansen MD
9 Gonzalo Keeton NJ
10 Justin Javer NY
11 Erick Kaleigh NC
12 Marvin Quintin OH
13 Kristen Story DC
Query executed successfully.
DESKTOP-UP37904\SQLEXPRESS ... DESKTOP-UP37904\owner ... ProductOrders 00:00:00 13 rows
Ln 4 Col 28 Ch 28 INS
```

Concatenating two columns and giving them new column alias with specific requirement on the State column

3. Select CustCity + ',' + CustState + ' ' + CustZip AS 'New Address'
 FROM ProductOrders.dbo.Customers
 ORDER BY CustState DESC;



```
--Lichen Liang
--Select CustCity + ',' + CustState + ' ' + CustZip AS 'New Address'
FROM ProductOrders.dbo.Customers
ORDER BY CustState DESC;
```

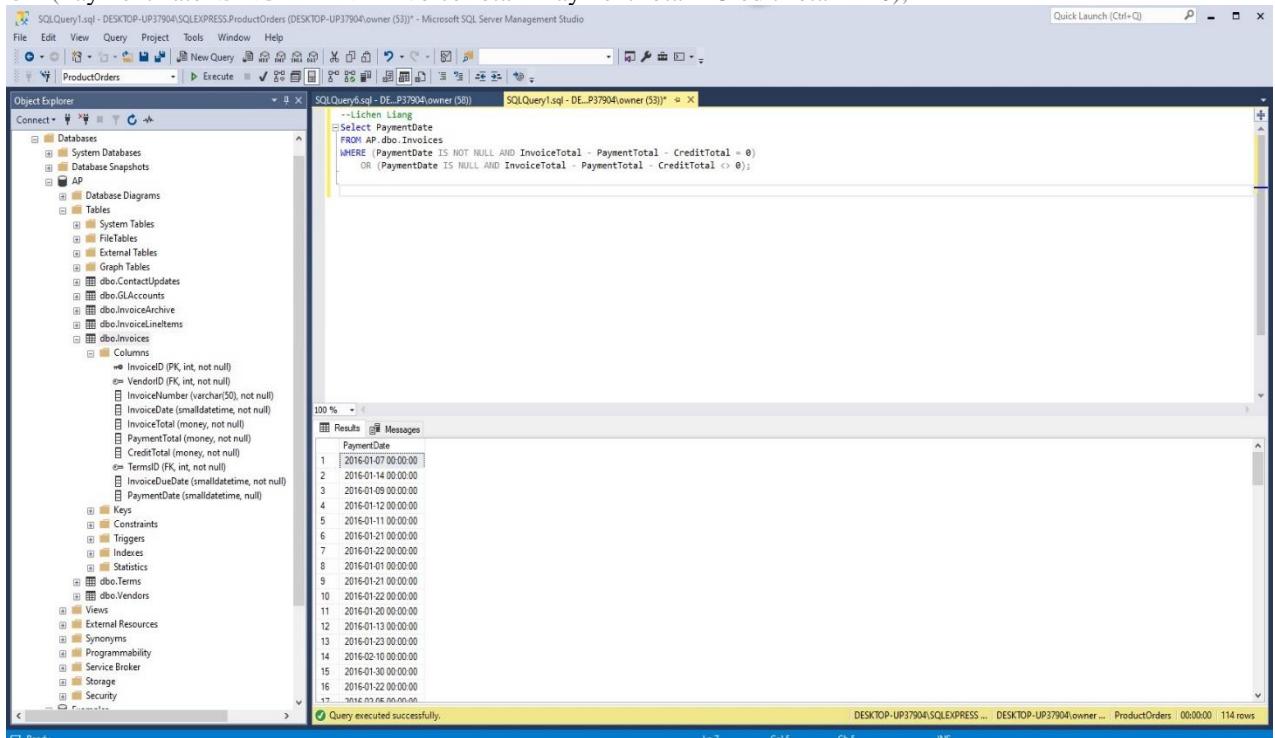
The screenshot shows the SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including the ProductOrders database and its tables like Customers. The Results pane on the right displays the output of the query, which concatenates the CustCity, CustState, and CustZip columns into a single 'New Address' column and orders the results by CustState. The results are as follows:

	New Address
1	Madison, WI 53707
2	Cleveland, OH 45002
3	Cincinnati, OH 45225
4	Columbus, OH 43221
5	Columbus, OH 43260
6	Tamptown, NY 10591
7	New York, NY 10010
8	Fairfield, NJ 07040
9	Charlotte, NC 28217
10	Takoma Park, MD 20912
11	Farfield, IA 52556
12	Washington, DC 20006
13	Washington, DC 20559
14	Fresno, CA 93711
15	Manhattan Beach, CA 90266
16	San Francisco, CA 94161
17	Brentwood, CA 94511

Query executed successfully.

Concatenating three columns, aliasing, and ordering.

4. Select PaymentDate
 FROM AP.dbo.Invoices
 WHERE (PaymentDate IS NOT NULL AND InvoiceTotal - PaymentTotal - CreditTotal = 0)
 OR (PaymentDate IS NULL AND InvoiceTotal - PaymentTotal - CreditTotal <> 0);



```
--Lichen Liang
--Select PaymentDate
FROM AP.dbo.Invoices
WHERE (PaymentDate IS NOT NULL AND InvoiceTotal - PaymentTotal - CreditTotal = 0)
  OR (PaymentDate IS NULL AND InvoiceTotal - PaymentTotal - CreditTotal <> 0);
```

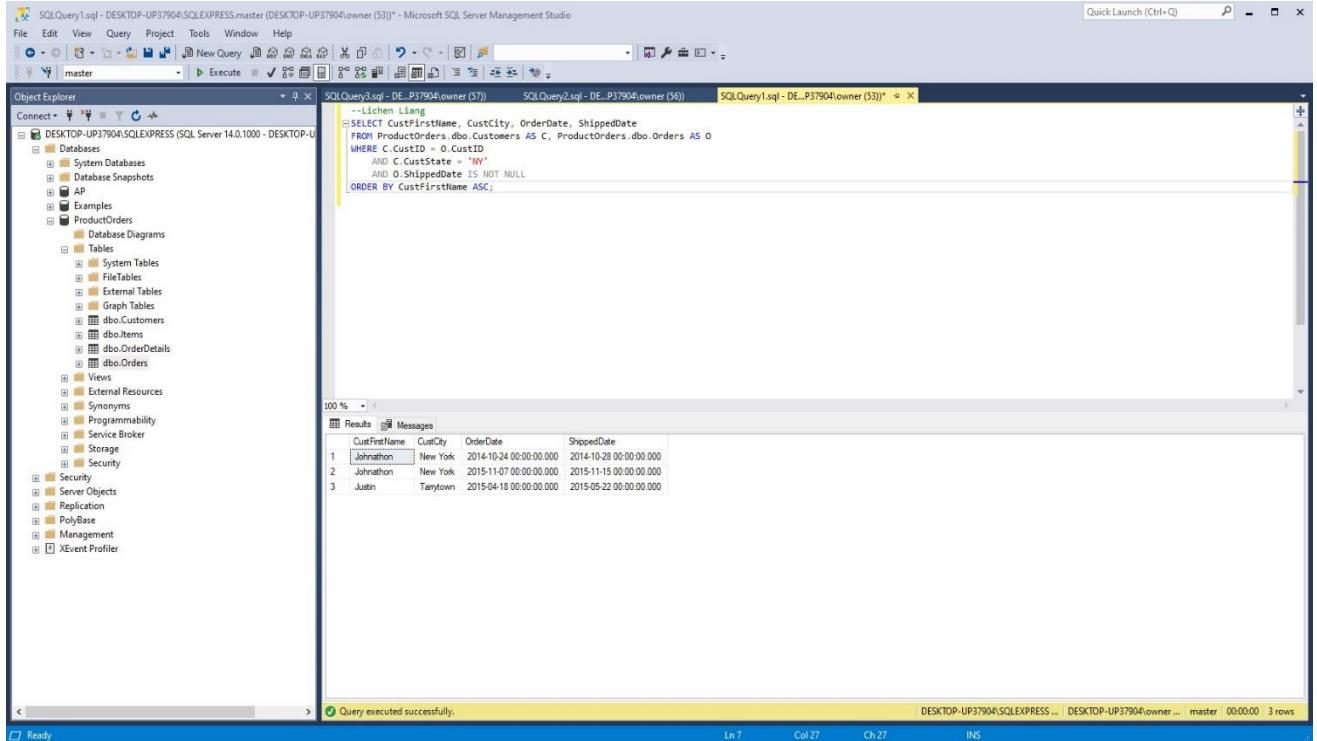
The screenshot shows the SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including the AP database and its tables like invoices. The Results pane on the right displays the output of the query, which filters payment dates based on specific conditions related to invoice total, payment total, and credit total. The results are as follows:

	PaymentDate
1	2016-01-07 00:00:00
2	2016-01-14 00:00:00
3	2016-01-09 00:00:00
4	2016-01-12 00:00:00
5	2016-01-11 00:00:00
6	2016-01-21 00:00:00
7	2016-01-22 00:00:00
8	2016-01-01 00:00:00
9	2016-01-21 00:00:00
10	2016-01-22 00:00:00
11	2016-01-20 00:00:00
12	2016-01-13 00:00:00
13	2016-01-23 00:00:00
14	2016-02-10 00:00:00
15	2016-01-30 00:00:00
16	2016-01-22 00:00:00
17	2016-01-08 00:00:00

Query executed successfully.

Looking for valid entries given specific condition
The output shows the valid entries.

5. **SELECT CustFirstName, CustCity, OrderDate, ShippedDate**
FROM ProductOrders.dbo.Customers AS C, ProductOrders.dbo.Orders AS O
WHERE C.CustID = O.CustID
AND C.CustState = 'NY'
AND O.ShippedDate IS NOT NULL
ORDER BY CustFirstName ASC;

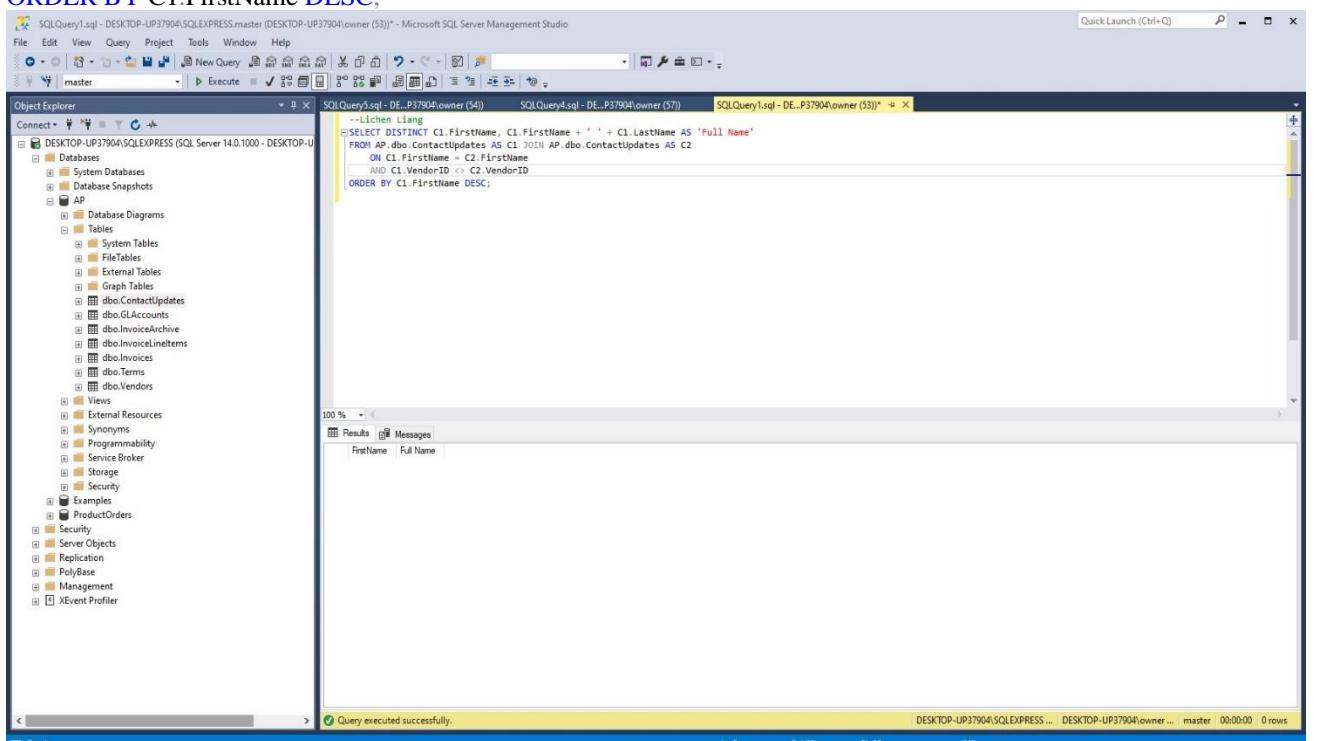


```
--Lichen Liang
--SELECT CustFirstName, CustCity, OrderDate, ShippedDate
--FROM ProductOrders.dbo.Customers AS C, ProductOrders.dbo.Orders AS O
--WHERE C.CustID = O.CustID
--AND C.CustState = 'NY'
--AND O.ShippedDate IS NOT NULL
--ORDER BY CustFirstName ASC;
```

CustFirstName	CustCity	OrderDate	ShippedDate
Johnathon	New York	2014-10-24 00:00:00.000	2014-10-28 00:00:00.000
Johnathon	New York	2015-11-07 00:00:00.000	2015-11-15 00:00:00.000
Justin	Tanytown	2015-04-18 00:00:00.000	2015-05-22 00:00:00.000

Matching the columns from two different tables, given conditions, order.

6. **SELECT DISTINCT C1.FirstName, C1.FirstName + ' ' + C1.LastName AS 'Full Name'**
FROM AP.dbo.ContactUpdates AS C1 JOIN AP.dbo.ContactUpdates AS C2
ON C1.FirstName = C2.FirstName
AND C1.VendorID <> C2.VendorID
ORDER BY C1.FirstName DESC;



```
--Lichen Liang
--SELECT DISTINCT C1.FirstName, C1.FirstName + ' ' + C1.LastName AS 'Full Name'
--FROM AP.dbo.ContactUpdates AS C1 JOIN AP.dbo.ContactUpdates AS C2
--ON C1.FirstName = C2.FirstName
--AND C1.VendorID <> C2.VendorID
--ORDER BY C1.FirstName DESC;
```

FirstName	Full Name
-----------	-----------

Looking for people with same first name and different last name from the same table.

There are no people who have the same first name.

7. `SELECT CustLastName, 'Columbus' AS CustCity
From ProductOrders.dbo.Customers
WHERE CustCity = 'Columbus'
UNION
SELECT CustLastName, 'Not in Co' AS CustCity
From ProductOrders.dbo.Customers
WHERE CustCity <> 'Columbus'
ORDER BY CustLastName ASC;`

```
--Lichen Liang
--SELECT CustLastName, 'Columbus' AS CustCity
From ProductOrders.dbo.Customers
WHERE CustCity = 'Columbus'

UNION

SELECT CustLastName, 'Not in Co' AS CustCity
From ProductOrders.dbo.Customers
WHERE CustCity <> 'Columbus'

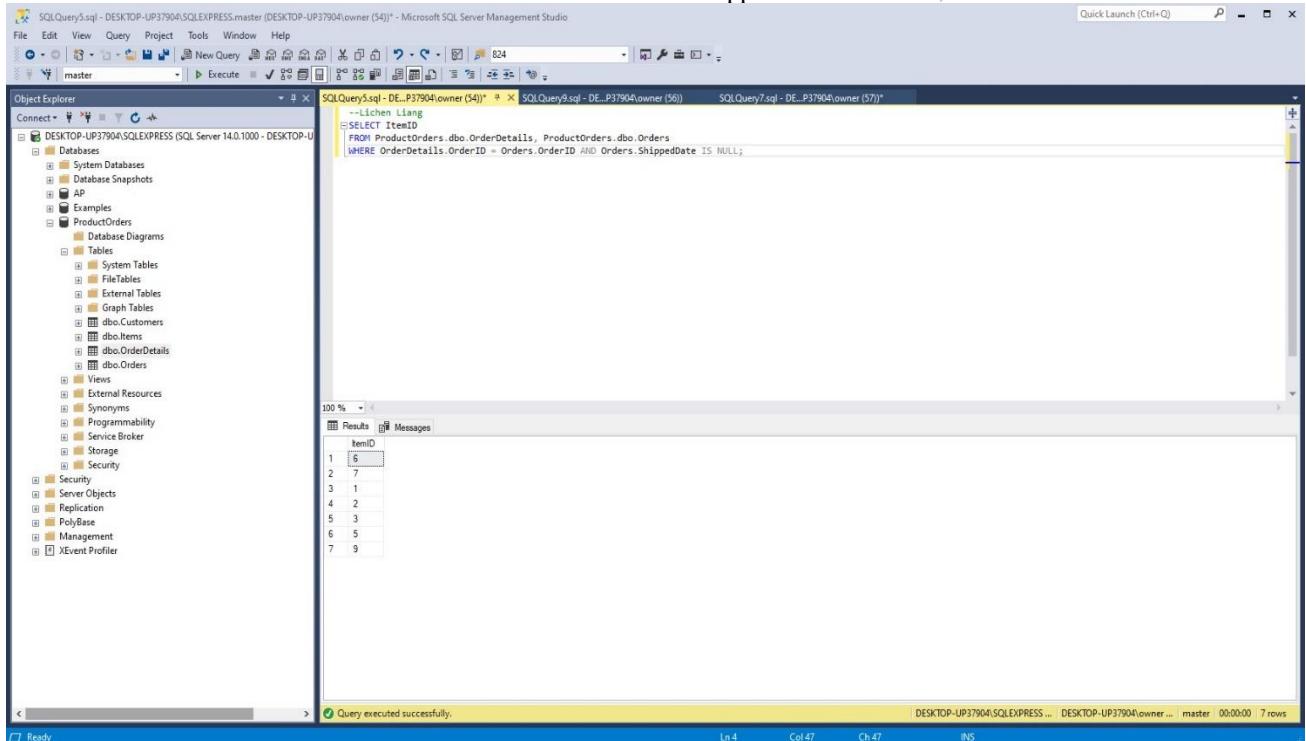
ORDER BY CustLastName ASC;
```

CustLastName	CustCity
Anum	Not in Co
Baylee	Not in Co
Blanca	Columbus
Caron	Not in Co
Chaddick	Not in Co
Damien	Not in Co
Damon	Not in Co
Eudalia	Not in Co
Holbroke	Not in Co
Hostery	Not in Co
Inri	Not in Co
Jacobseen	Not in Co
Javen	Not in Co
Kaleigh	Not in Co
Keeton	Not in Co
Lacy	Not in Co
Meredith	Not in Co
Natalia	Not in Co
Reed	Not in Co
Riley	Not in Co
Sherman	Not in Co
Tanner	Not in Co
Whitney	Not in Co

Listing people who live and don't live in Columbus.

8. **SELECT** ItemID

FROM ProductOrders.dbo.OrderDetails, ProductOrders.dbo.Orders
WHERE OrderDetails.OrderID = Orders.OrderID AND Orders.ShippedDate IS NULL;



The screenshot shows the Microsoft SQL Server Management Studio interface. In the Object Explorer on the left, the 'ProductOrders' database is selected, showing its tables: 'Customers', 'Items', 'OrderDetails', and 'Orders'. In the center pane, a query window displays the following T-SQL code:

```
--Lichen Liang
SELECT ItemID
FROM ProductOrders.dbo.OrderDetails, ProductOrders.dbo.Orders
WHERE OrderDetails.OrderID = Orders.OrderID AND Orders.ShippedDate IS NULL;
```

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

itemID
1
2
3
4
5
6
7

Below the results, a message bar indicates: 'Query executed successfully.' and 'DESKTOP-UP37904\SQLEXPRESS ... DESKTOP-UP37904\owner ... master 00:00:00 7 rows'.

Query data that has underlying conditions of which those are not displayed.

Remarks

In this lab we practiced query using different databases, tables, columns, etc. In the query, we learned how to use joins, unions, where, and order by, for setting the conditions for a query that meets the lab requirement. I think this lab is very efficient practice for getting used to SQL language. The challenge would be more specific requirements.