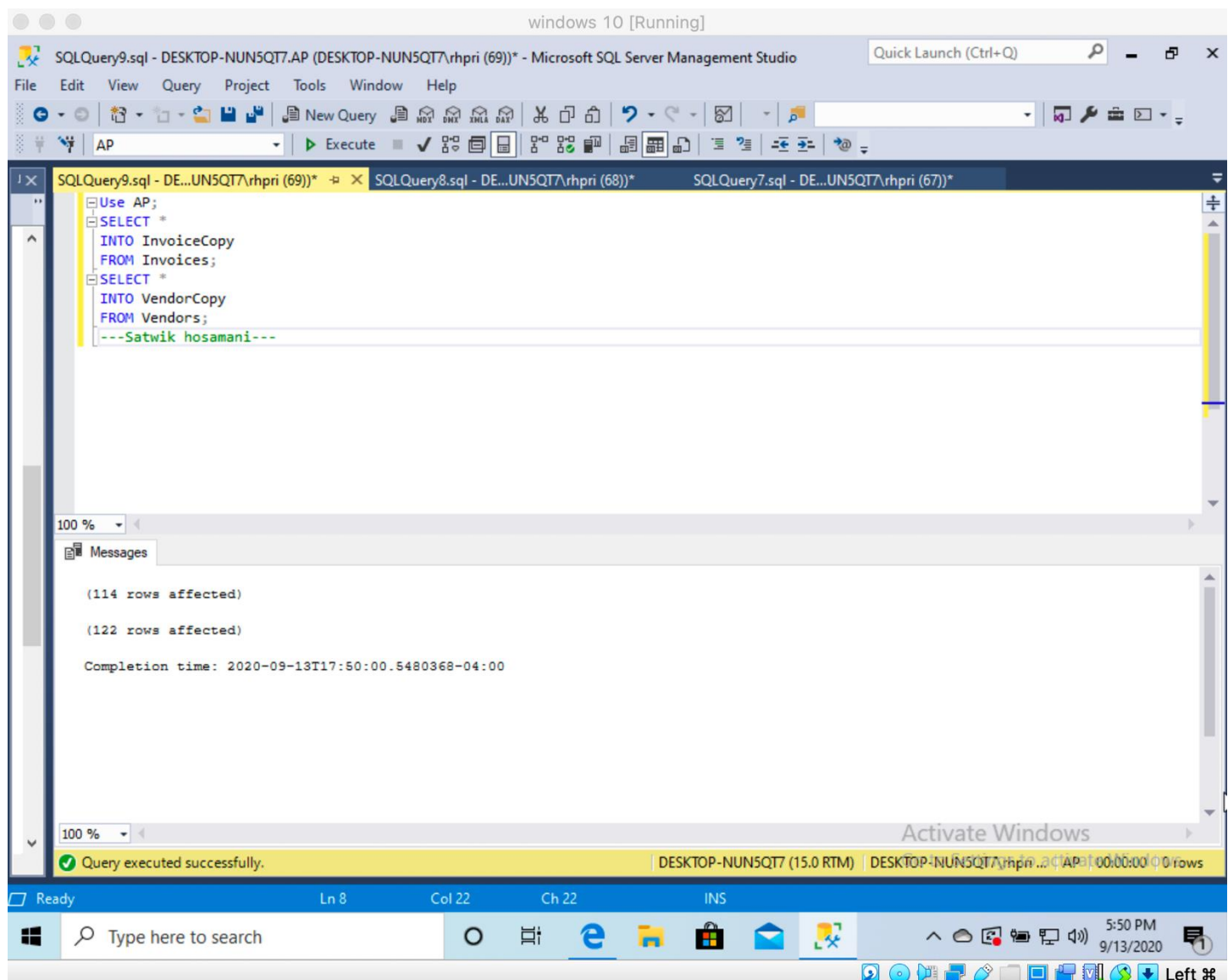


Q1. Create VendorCopy table and InvoiceCopy table

Comments: In this query we had to copy the table rows to another table so I have used the "select into" statement which copies all the rows from the parent table to the new table. The new table is created same as the parent one with all the column names same. We don't have to explicitly create a new table.

Sol:

```
Use AP;
SELECT
INTO InvoiceCopy
FROM Invoices;
SELECT
INTO VendorCopy
FROM Vendors;
----Satwik Hosamani----
```



Rows in InvoiceCopy :114
Rows in VendorCopy:122

2. Write an INSERT statement that adds a row to the InvoiceCopy table with the following values (USE SELECT statement to verify data changes in the table before and after the modification):

VendorID: 2 InvoiceTotal: \$401.40
 TermsID: 3 InvoiceNumber: CM-007-700
 PaymentTotal: \$2.99 InvoiceDueDate: 09/01/15
 InvoiceDate: 08/24/15 CreditTotal: \$5.99
 Do we explicitly need to insert InvoiceID?

Comments: We don't need to explicitly insert into InvoiceID because select into statement copies the properties of the column too not just the name, InvoiceID has identity(1,1) meaning the first 1 means the starting value of ID and the second 1 means the increment value of ID

Before Insertion:

Select * from AP.dbo.InvoiceCopy;

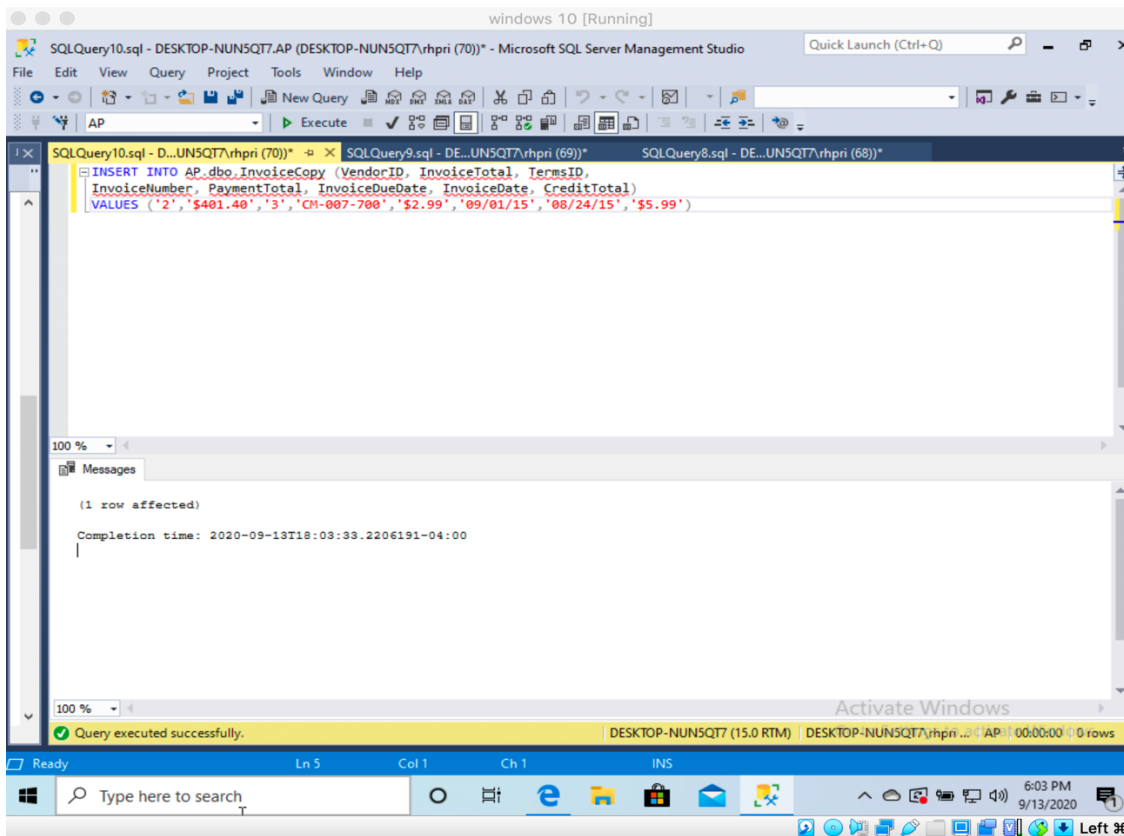
Rows: 114

The screenshot displays the Microsoft SQL Server Management Studio interface. The query editor shows the command: `Select * from AP.dbo.InvoiceCopy;`. The Results pane below shows a table with 114 rows. The columns are: InvoiceID, VendorID, InvoiceNumber, InvoiceDate, InvoiceTotal, PaymentTotal, CreditTotal, TermsID, InvoiceDueDate, and PaymentDate. The status bar at the bottom indicates 'Query executed successfully.' and '114 rows'.

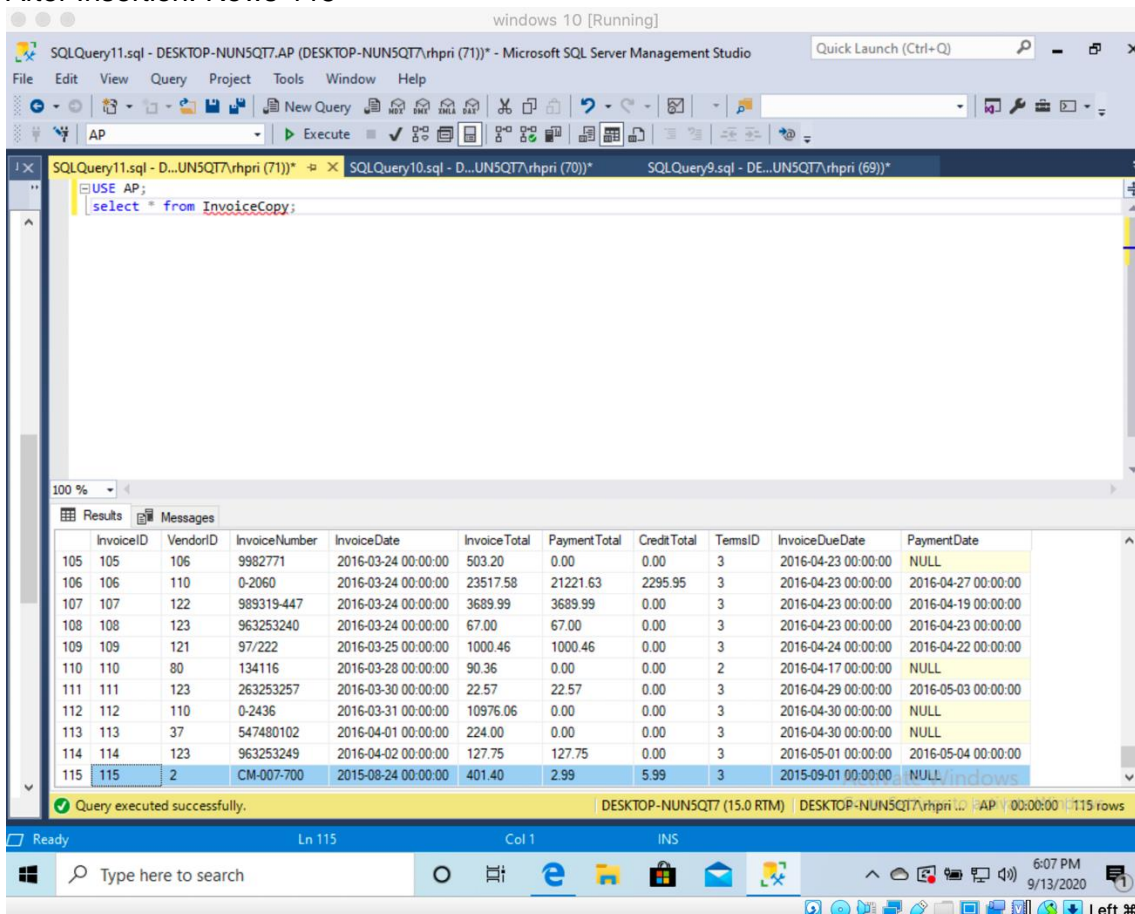
InvoiceID	VendorID	InvoiceNumber	InvoiceDate	InvoiceTotal	PaymentTotal	CreditTotal	TermsID	InvoiceDueDate	PaymentDate	
100	100	123	263253270	2016-03-22 00:00:00	67.92	0.00	0.00	3	2016-04-21 00:00:00	NULL
101	101	123	263253273	2016-03-22 00:00:00	30.75	0.00	0.00	3	2016-04-21 00:00:00	NULL
102	102	110	P-0608	2016-03-23 00:00:00	20551.18	0.00	1200.00	3	2016-04-22 00:00:00	NULL
103	103	122	989319-417	2016-03-23 00:00:00	2051.59	2051.59	0.00	3	2016-04-22 00:00:00	2016-04-24 00:00:00
104	104	123	263253243	2016-03-23 00:00:00	44.44	44.44	0.00	3	2016-04-22 00:00:00	2016-04-24 00:00:00
105	105	106	9982771	2016-03-24 00:00:00	503.20	0.00	0.00	3	2016-04-23 00:00:00	NULL
106	106	110	0-2060	2016-03-24 00:00:00	23517.58	21221.63	2295.95	3	2016-04-23 00:00:00	2016-04-27 00:00:00
107	107	122	989319-447	2016-03-24 00:00:00	3689.99	3689.99	0.00	3	2016-04-23 00:00:00	2016-04-19 00:00:00
108	108	123	963253240	2016-03-24 00:00:00	67.00	67.00	0.00	3	2016-04-23 00:00:00	2016-04-23 00:00:00
109	109	121	97/222	2016-03-25 00:00:00	1000.46	1000.46	0.00	3	2016-04-24 00:00:00	2016-04-22 00:00:00
110	110	80	134116	2016-03-28 00:00:00	90.36	0.00	0.00	2	2016-04-17 00:00:00	NULL

SOL:

```
INSERT INTO AP.dbo.InvoiceCopy (VendorID, InvoiceTotal, TermsID,
InvoiceNumber, PaymentTotal, InvoiceDueDate, InvoiceDate, CreditTotal)
VALUES ('2', '$401.40', '3', 'CM-007-700', '$2.99', '09/01/15', '08/24/15', '$5.99')
```



After Insertion: Rows-115



3. Write an UPDATE statement that modifies the VendorCopy table. Change the default account number to 542 for each vendor that has a default account number of 572. (USE SELECT statement to verify data changes in the table before and after the modification).

Comments: Update statement is used to modify the values of the rows which satisfy the condition. here the condition is that the account number of vendor should be 572 changed to 542. The condition is mentioned in the where clause

ROWS:14

Before:

```
USE AP;
SELECT * FROM VendorCopy where DefaultAccountNo = 572;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The query window displays the following SQL statement:

```
USE AP;
SELECT * FROM VendorCopy where DefaultAccountNo = 572;
```

The Results pane shows 14 rows of data from the VendorCopy table. The columns are: VendorAddress2, VendorCity, VendorState, VendorZipCode, VendorPhone, VendorContactLName, VendorContactFName, DefaultTermsID, and DefaultAccountNo. All rows have a DefaultAccountNo of 572.

	VendorAddress2	VendorCity	VendorState	VendorZipCode	VendorPhone	VendorContactLName	VendorContactFName	DefaultTermsID	DefaultAccountNo
1	NULL	Los Angeles	CA	90025	(800) 555-8725	Quinn	Kenzie	3	572
2	NULL	Sacramento	CA	95827	(916) 555-6670	Mauro	Anton	3	572
3	NULL	Oberlin	OH	44074	NULL	Bernard	Lucy	3	572
4	NULL	Los Angeles	CA	90038	(213) 555-4322	Paris	Gideon	3	572
5	NULL	Jacksonville	FL	32231	(800) 555-6041	Gerald	Kristofer	3	572
6	2709 Water Ridge Parkway	Charlotte	NC	28217	(704) 555-3500	Bernardo	Brittnee	3	572
7	NULL	Chicago	IL	60680	(614) 555-3663	Holbrooke	Rashad	3	572
8	NULL	Marion	OH	43305	(800) 555-1669	Carrollton	Priscilla	3	572
9	NULL	Manhattan Beach	CA	90266	(310) 555-2732	Walker	Jovon	3	572
10	NULL	Fresno	CA	93786	(559) 555-4442	Colton	Leah	2	572
11	NULL	Washington	DC	20006	(202) 555-5561	Rodolfo	Carlee	2	572
12	PO Box 61000	San Francisco	CA	94161	(617) 555-0700	Lloyd	Angel	1	572
13	NULL	Columbus	OH	43260	(614) 555-8600	Armando	Jan	2	572
14	NULL	Fresno	CA	93721	NULL	Kraggin	Laura	1	572

The status bar at the bottom indicates: Query executed successfully. DESKTOP-NUN5QT7 (15.0 RTM) DESKTOP-NUN5QT7\rhpri... AP 00:00:00 14 rows

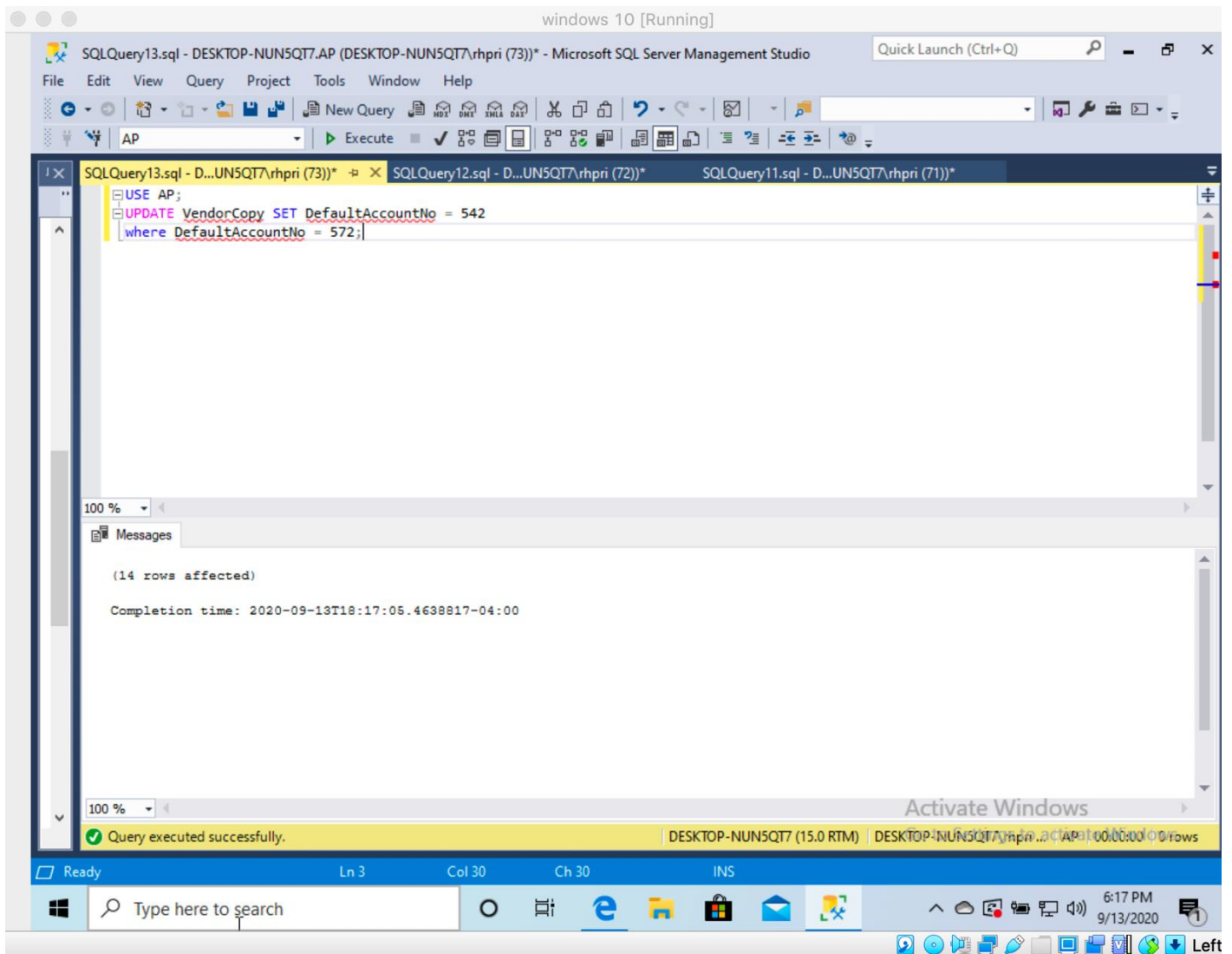
TO UPDATE:

```
USE AP;
UPDATE VendorCopy SET DefaultAccountNo= 542
WHERE DefaultAccountNo = 572;
```

SHOSAMAN

378000248

ROWS Affected: 14



CHECKING THE UPDATE:

```
USE AP;  
Select *VendorCopy wherer DefaultAccountNo = 542;
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The query window displays the following SQL statement:

```
USE AP;
SELECT * from VendorCopy where DefaultAccountNo=542;
```

The Results pane shows the following data:

	/endorAddress2	VendorCity	VendorState	VendorZipCode	VendorPhone	VendorContactLName	VendorContactFName	DefaultTermsID	DefaultAccountNo
1	NULL	Los Angeles	CA	90025	(800) 555-8725	Quinn	Kenzie	3	542
2	NULL	Sacramento	CA	95827	(916) 555-6670	Mauro	Anton	3	542
3	NULL	Oberlin	OH	44074	NULL	Bernard	Lucy	3	542
4	NULL	Los Angeles	CA	90038	(213) 555-4322	Paris	Gideon	3	542
5	NULL	Jacksonville	FL	32231	(800) 555-6041	Gerald	Kristofer	3	542
6	2709 Water Ridge Parkway	Charlotte	NC	28217	(704) 555-3500	Bernardo	Brittnee	3	542
7	NULL	Chicago	IL	60680	(614) 555-3663	Holbrooke	Rashad	3	542
8	NULL	Marion	OH	43305	(800) 555-1669	Carrollton	Priscilla	3	542
9	NULL	Manhattan Beach	CA	90266	(310) 555-2732	Walker	Jovon	3	542
10	NULL	Fresno	CA	93786	(559) 555-4442	Colton	Leah	2	542
11	NULL	Washington	DC	20006	(202) 555-5561	Rodolfo	Carlee	2	542
12	PO Box 61000	San Francisco	CA	94161	(617) 555-0700	Lloyd	Angel	1	542
13	NULL	Columbus	OH	43260	(614) 555-8600	Armando	Jan	2	542
14	NULL	Fresno	CA	93721	NULL	Kraggin	Laura	1	542

The status bar at the bottom indicates: Query executed successfully. DESKTOP-NUN5QT7 (15.0 RTM) DESKTOP-NUN5QT7\rhpri... AP 00:00:00 14 rows

4). Write an UPDATE statement that modifies the InvoiceCopy table. Change the TermsID to 5 for each invoice that's from a vendor with a defaultTermsID of 2. Use a subquery. (USE SELECT statement to verify data changes in the table before and after the modification).

Comments: Update statement is used to modify the values of the rows which satisfy the condition. here the condition is that the TermsID should be 2 which should be changed to 5. The condition is mentioned in the where clause for the update statement

Before Updating:

```
USE AP;
SELECT VendorCopy.VendorID, VendorCopy.DefaultTermsID, InvoiceCopy.TermsID
FROM VendorCopy JOIN InvoiceCopy
ON VendorCopy.VendorID = InvoiceCopy.VendorID
and VendorCopy.DefaultTermsID = '2'
```

The screenshot shows the Microsoft SQL Server Management Studio interface. The query window displays the following SQL code:

```
USE AP;
SELECT VendorCopy.VendorID, VendorCopy.DefaultTermsID, InvoiceCopy.TermsID
FROM VendorCopy JOIN InvoiceCopy
ON VendorCopy.VendorID = InvoiceCopy.VendorID
and VendorCopy.DefaultTermsID = '2'
```

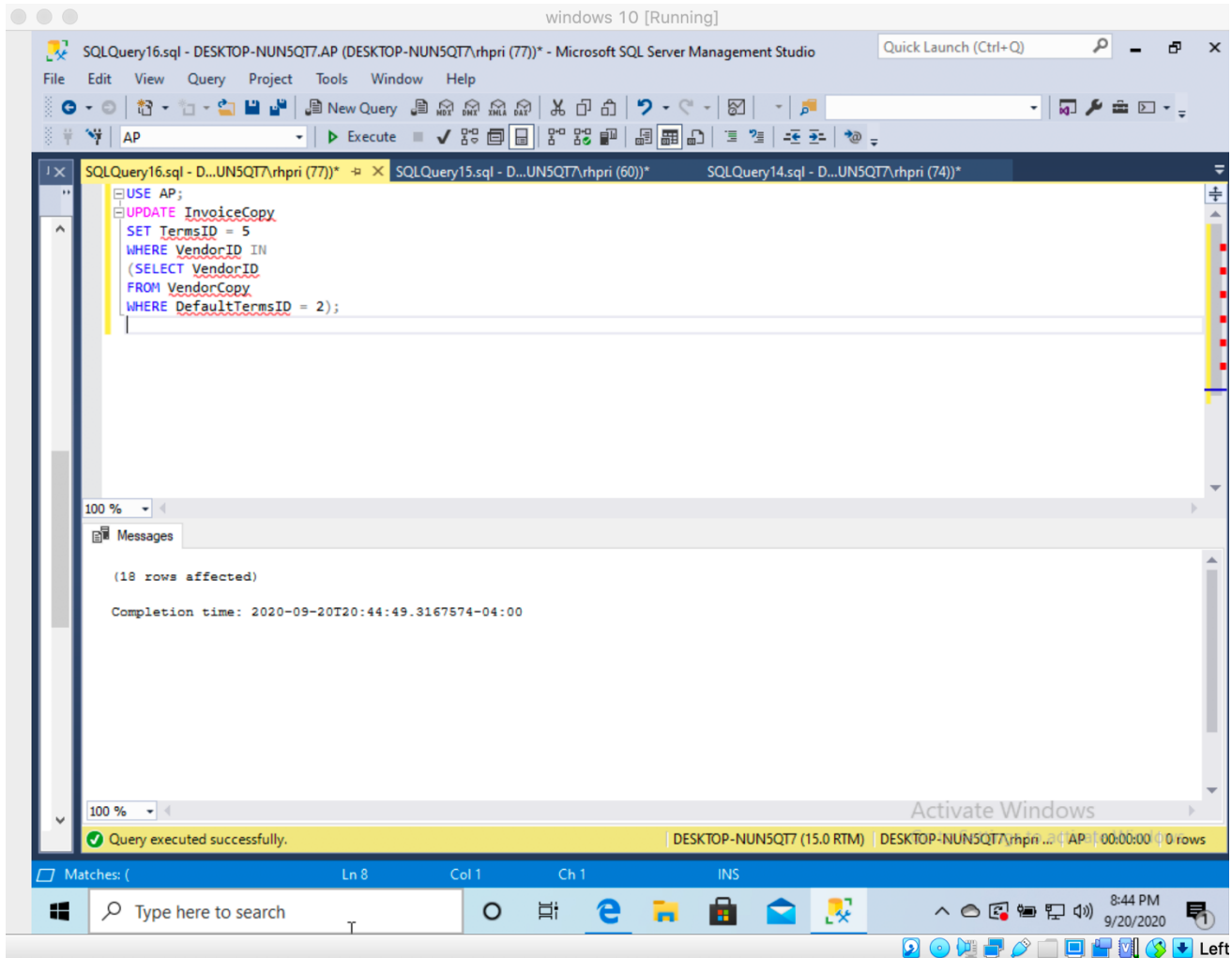
The Results pane shows the following data:

	VendorID	DefaultTermsID	TermsID
1	96	2	2
2	95	2	2
3	97	2	2
4	119	2	2
5	97	2	2
6	95	2	2
7	83	2	2
8	95	2	2
9	95	2	2
10	81	2	2
11	80	2	2
12	94	2	2
13	95	2	2
14	95	2	2
15	82	2	2
16	90	2	2
17	83	2	2
18	80	2	2

The status bar indicates "Query executed successfully." and "18 rows".

UPDATE:

```
USE AP;
UPDATE InvoiceCopy
SET TermsID = 5
WHERE VendorID IN
(SELECT VendorID
FROM VendorCopy
WHERE DefaultTermsID = 2);
```



After Updating:

```
USE AP;
SELECT VendorCopy.VendorID, VendorCopy.DefaultTermsID, InvoiceCopy.TermsID
FROM VendorCopy JOIN InvoiceCopy
ON VendorCopy.VendorID = InvoiceCopy.VendorID
and VendorCopy.DefaultTermsID = '2'
```


The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor contains the following SQL code:

```
USE AP;
SELECT VendorCopy.VendorID, VendorCopy.DefaultTermsID, InvoiceCopy.TermsID
FROM VendorCopy JOIN InvoiceCopy
ON VendorCopy.VendorID = InvoiceCopy.VendorID
and VendorCopy.DefaultTermsID = '2'
```

The Results pane displays the following data:

	VendorID	DefaultTermsID	TermsID
1	96	2	5
2	95	2	5
3	97	2	5
4	119	2	5
5	97	2	5
6	95	2	5
7	83	2	5
8	95	2	5
9	95	2	5
10	81	2	5
11	80	2	5
12	94	2	5
13	95	2	5
14	95	2	5
15	82	2	5
16	90	2	5
17	83	2	5
18	80	2	5

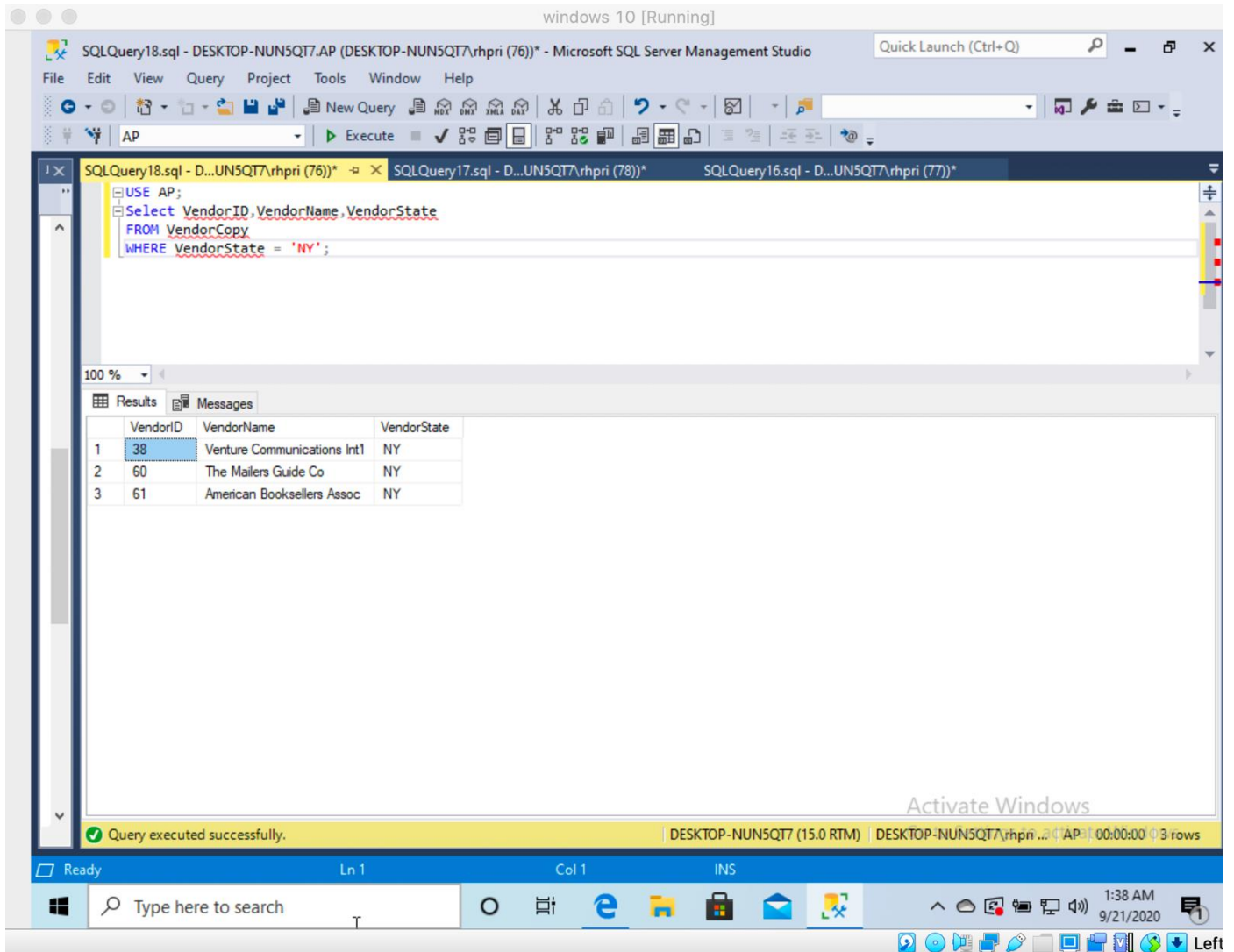
The status bar at the bottom indicates: "Query executed successfully. DESKTOP-NUN5QT7 (15.0 RTM) DESKTOP-NUN5QT7\rhpri... AP 00:00:00 18 rows".

5. Write a DELETE statement that deletes all vendors in the state of 'New York' from the VendorCopy table. (USE SELECT statement to verify data changes in the table before and after the modification).

Comments: Delete statement is used to delete the existing rows in the table. Depending on the condition a single or multiple row is deleted, the condition is specified in the where clause. Here we are told to delete the vendors with New York state.

Sol:

```
USE AP;
SELECT VendorID, VendorName, VendorState
FROM VendorCopy
WHERE VendorState='NY';
```



The screenshot displays the Microsoft SQL Server Management Studio interface. The query editor shows the following SQL script:

```
USE AP;  
SELECT VendorID, VendorName, VendorState  
FROM VendorCopy  
WHERE VendorState = 'NY';
```

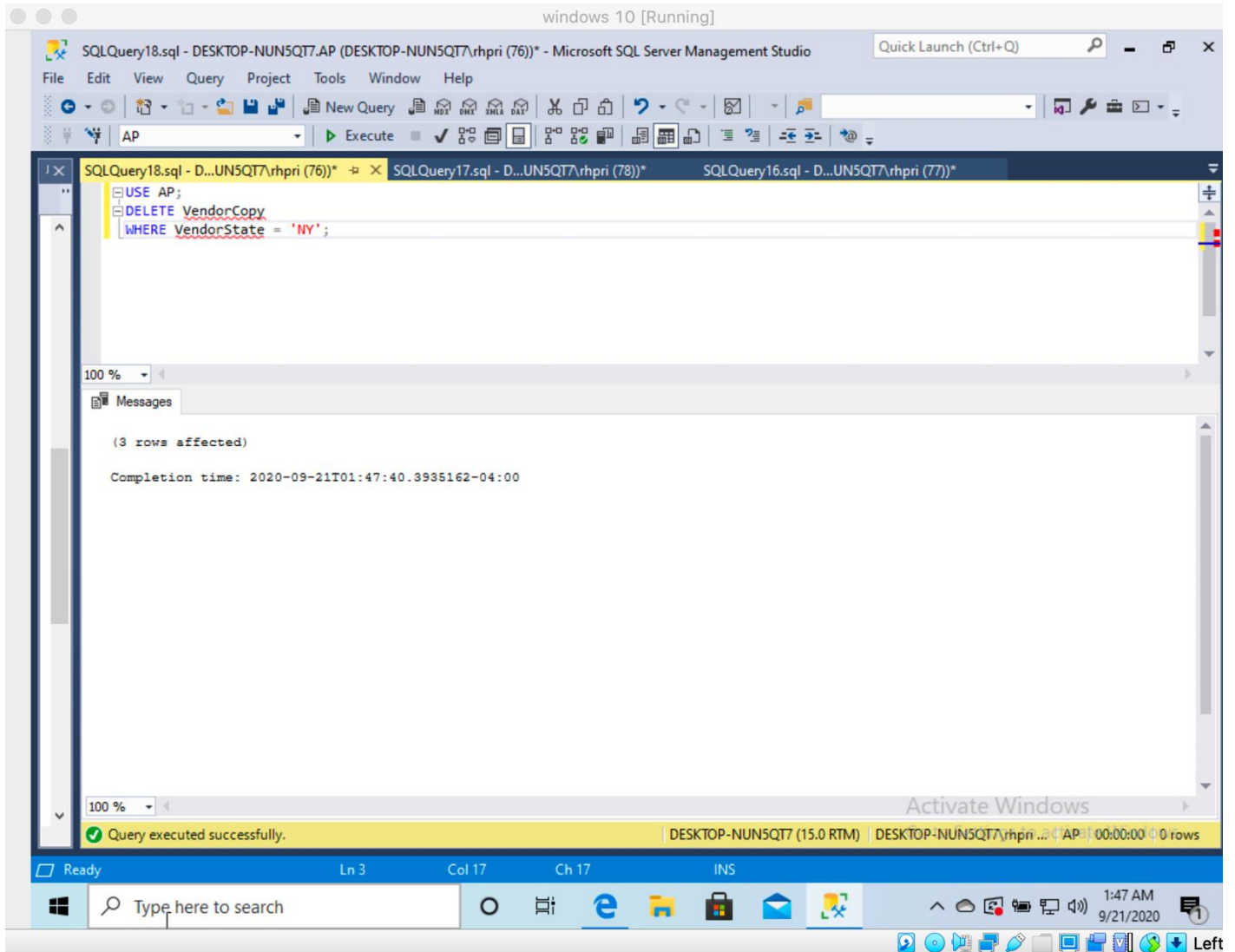
The Results pane shows the output of the query, which is a table with 3 rows and 3 columns: VendorID, VendorName, and VendorState.

	VendorID	VendorName	VendorState
1	38	Venture Communications Int1	NY
2	60	The Mailers Guide Co	NY
3	61	American Booksellers Assoc	NY

The status bar at the bottom indicates "Query executed successfully." and "DESKTOP-NUN5QT7 (15.0 RTM) | DESKTOP-NUN5QT7\rhpri... | AP | 00:00:00 | 3 rows".

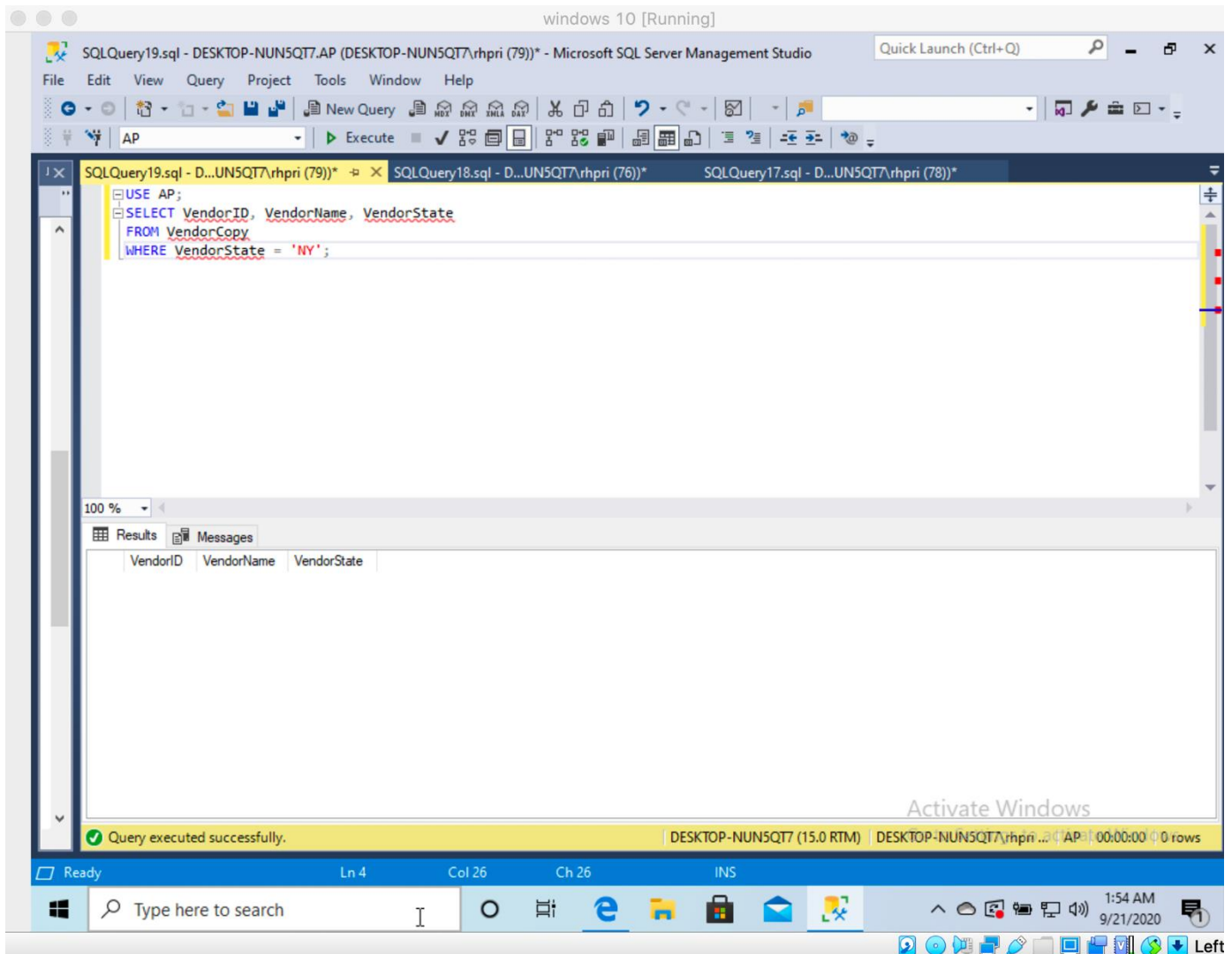
Sol:

```
USE AP;  
DELETE VendorCopy  
WHERE VendorState='NY';
```



After Deleting:

```
USE AP;  
SELECT VendorID, VendorName, VendorState  
FROM VendorCopy  
WHERE VendorState='NY';
```



6. Write a DELETE statement for the VendorCopy table. Delete the vendors that are located in cities from which no vendor has ever sent an invoice. (USE SELECT statement to verify data changes in the table before and after the modification)

Comments: Here the question is delete the rows which have no vendor invoice to do that I have joined vendorCopy and Invoice copy table and then not selecting those vendors which are there in the invoice table
BEFORE DELETION :119
AFTER DELETION:86

```

USE AP;
SELECT VendorCity
FROM VendorCopy
WHERE VendorCity NOT IN
(SELECT DISTINCT VendorCity
From VendorCopy JOIN InvoiceCopy
ON VendorCopy.VendorID = InvoiceCopy.VendorID);
SELECT COUNT (*)
FROM VendorCopy;
-----DELETING UNIQUE CITIES-----
DELETE VendorCopy
WHERE VendorCity NOT IN
    (SELECT DISTINCT VendorCity
    FROM VendorCopy JOIN InvoiceCopy
    ON VendorCopy.VendorID = InvoiceCopy.VendorID);

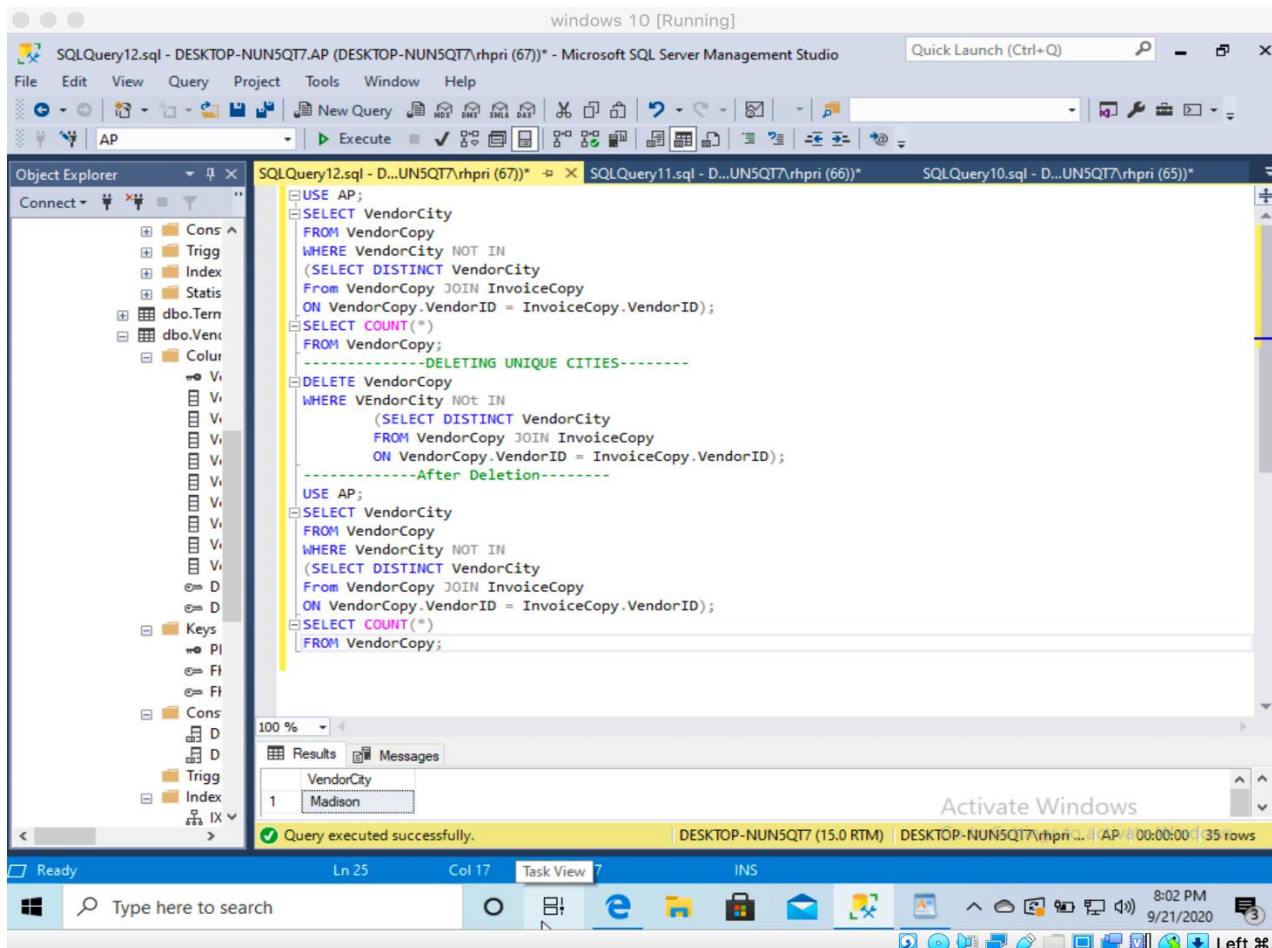
```

-----After Deletion-----

```

USE AP;
SELECT VendorCity
FROM VendorCopy
WHERE VendorCity NOT IN
(SELECT DISTINCT VendorCity
From VendorCopy JOIN InvoiceCopy
ON VendorCopy.VendorID = InvoiceCopy.VendorID);
SELECT COUNT(*)
FROM VendorCopy;

```



The screenshot shows the Microsoft SQL Server Management Studio interface. The query window displays the following SQL code:

```

-----After Deletion-----
USE AP;
SELECT VendorCity
FROM VendorCopy
WHERE VendorCity NOT IN
(SELECT DISTINCT VendorCity
FROM VendorCopy JOIN InvoiceCopy
ON VendorCopy.VendorID = InvoiceCopy.VendorID);
SELECT COUNT(*)
FROM VendorCopy;

```

The Results pane shows the output of the query:

VendorCity
1
2
3
4
5
6
7
8

(No column name)
1

VendorCity
1

(No column name)
1

The status bar at the bottom indicates: "Query executed successfully. DESKTOP-NUN5QT7 (15.0 RTM) DESKTOP-NUN5QT7\rhpri... AP 00:00:00 35 rows".

7. Write a SELECT statement that returns four columns based on the InvoiceTotal column of the Invoices table:
- Use CAST function to return the first column as data type decimal with 4 digits to the right of the decimal point.
 - Use CAST to return the second column as a VARCHAR.
 - Use CONVERT function to return third column as the same type as the first column.
 - Use CONVERT to return the fourth column as a VARCHAR, using style 10.

Comments: CAST function converts an expression from one datatype to another datatype. If the conversion fails, the function will return an error. CAST function converts an expression from one datatype to another datatype. If the conversion fails, the function will return an error.

SOL:

SHOSAMAN

378000248

```

USE AP;
SELECT CAST (InvoiceTotal AS DECIMAL(17,4)) AS FIRST_COLUMN,
CAST (InvoiceTotal AS VARCHAR) AS SECOND_COLUMN,
CONVERT(DECIMAL (17,4),InvoiceTotal) AS THIRD_COLUMN,
CONVERT (VARCHAR, InvoiceTotal,2) AS FOURTH_COLUMN
FROM Invoices;

```

The screenshot shows the Microsoft SQL Server Management Studio interface. The query window displays the following SQL code:

```

USE AP;
SELECT CAST (InvoiceTotal AS DECIMAL(17,4)) AS FIRST_COLUMN,
CAST (InvoiceTotal AS VARCHAR) AS SECOND_COLUMN,
CONVERT(DECIMAL (17,4), InvoiceTotal) AS THIRD_COLUMN,
CONVERT (VARCHAR, InvoiceTotal,10) AS FOURTH_COLUMN
FROM Invoices;

```

The Results pane shows the output of the query, which consists of 19 rows. The columns are labeled FIRST_COLUMN, SECOND_COLUMN, THIRD_COLUMN, and FOURTH_COLUMN. The data is as follows:

	FIRST_COLUMN	SECOND_COLUMN	THIRD_COLUMN	FOURTH_COLUMN
1	3813.3300	3813.33	3813.3300	3.813.33
2	40.2000	40.20	40.2000	40.20
3	138.7500	138.75	138.7500	138.75
4	144.7000	144.70	144.7000	144.70
5	15.5000	15.50	15.5000	15.50
6	42.7500	42.75	42.7500	42.75
7	172.5000	172.50	172.5000	172.50
8	95.0000	95.00	95.0000	95.00
9	601.9500	601.95	601.9500	601.95
10	42.6700	42.67	42.6700	42.67
11	42.5000	42.50	42.5000	42.50
12	662.0000	662.00	662.0000	662.00
13	16.3300	16.33	16.3300	16.33
14	6.0000	6.00	6.0000	6.00
15	856.9200	856.92	856.9200	856.92
16	9.9500	9.95	9.9500	9.95
17	10.0000	10.00	10.0000	10.00
18	104.0000	104.00	104.0000	104.00
19	116.5400	116.54	116.5400	116.54

The status bar at the bottom indicates "Query executed successfully." and "DESKTOP-NUN5QT7 (15.0 RTM) | DESKTOP-NUN5QT7\rhpri ... | AP | 00:00:00 | 114 rows".

8. Write a SELECT statement that returns four columns based on the InvoiceDate column of the Invoices table:
- Use the CAST function to return the first column as data type VARCHAR.
 - Use the CONVERT function to return the second and third columns as a VARCHAR, using style 5 and style 9, respectively.
 - Use the CAST function to return the fourth column as a data type real.

Sol:

```

USE AP;
SELECT CAST(InvoiceDate AS VARCHAR) AS FIRST_COLUMN,
CONVERT(VARCHAR,InvoiceDate,2) AS SECOND_COLUMN,
CONVERT(VARCHAR,InvoiceDate,10) AS THIRD_COLUMN,
CAST(InvoiceDate AS REAL) AS FOURTH_COLUMN
FROM Invoices;

```

Comments: Here we have casted first column as Varchar directly ,CAST (expression AS [data type]) this is the format of the cast statement ,style is an integer expression which determines how will the function traslate.

The screenshot shows the Microsoft SQL Server Management Studio interface. The query window displays the following SQL code:

```

USE AP;
SELECT CAST(InvoiceDate AS VARCHAR) AS FIRST_COLUMN,
CONVERT (VARCHAR,InvoiceDate,5) AS SECOND_COLUMN,
CONVERT(VARCHAR,InvoiceDate,9) AS THIRD_COLUMN,
CAST(InvoiceDate AS REAL) AS FOURTH_COLUMN
FROM Invoices;

```

The query has been executed successfully, and the results are displayed in a table with 4 columns: FIRST_COLUMN, SECOND_COLUMN, THIRD_COLUMN, and FOURTH_COLUMN. The table contains 19 rows of data.

	FIRST_COLUMN	SECOND_COLUMN	THIRD_COLUMN	FOURTH_COLUMN
1	Apr 2 2016 12:00AM	02-04-16	Apr 2 2016 12:00:00:000AM	42460
2	Apr 1 2016 12:00AM	01-04-16	Apr 1 2016 12:00:00:000AM	42459
3	Mar 31 2016 12:00AM	31-03-16	Mar 31 2016 12:00:00:000AM	42458
4	Mar 30 2016 12:00AM	30-03-16	Mar 30 2016 12:00:00:000AM	42457
5	Mar 28 2016 12:00AM	28-03-16	Mar 28 2016 12:00:00:000AM	42455
6	Mar 25 2016 12:00AM	25-03-16	Mar 25 2016 12:00:00:000AM	42452
7	Mar 24 2016 12:00AM	24-03-16	Mar 24 2016 12:00:00:000AM	42451
8	Mar 24 2016 12:00AM	24-03-16	Mar 24 2016 12:00:00:000AM	42451
9	Mar 24 2016 12:00AM	24-03-16	Mar 24 2016 12:00:00:000AM	42451
10	Mar 24 2016 12:00AM	24-03-16	Mar 24 2016 12:00:00:000AM	42451
11	Mar 23 2016 12:00AM	23-03-16	Mar 23 2016 12:00:00:000AM	42450
12	Mar 23 2016 12:00AM	23-03-16	Mar 23 2016 12:00:00:000AM	42450
13	Mar 23 2016 12:00AM	23-03-16	Mar 23 2016 12:00:00:000AM	42450
14	Mar 22 2016 12:00AM	22-03-16	Mar 22 2016 12:00:00:000AM	42449
15	Mar 22 2016 12:00AM	22-03-16	Mar 22 2016 12:00:00:000AM	42449
16	Mar 21 2016 12:00AM	21-03-16	Mar 21 2016 12:00:00:000AM	42448
17	Mar 21 2016 12:00AM	21-03-16	Mar 21 2016 12:00:00:000AM	42448
18	Mar 20 2016 12:00AM	20-03-16	Mar 20 2016 12:00:00:000AM	42447
19	Mar 19 2016 12:00AM	19-03-16	Mar 19 2016 12:00:00:000AM	42446

The status bar at the bottom indicates: Query executed successfully. DESKTOP-NUN5QT7 (15.0 RTM) | DESKTOP-NUN5QT7\rhpri ... | AP | 00:00:00 | 114 rows