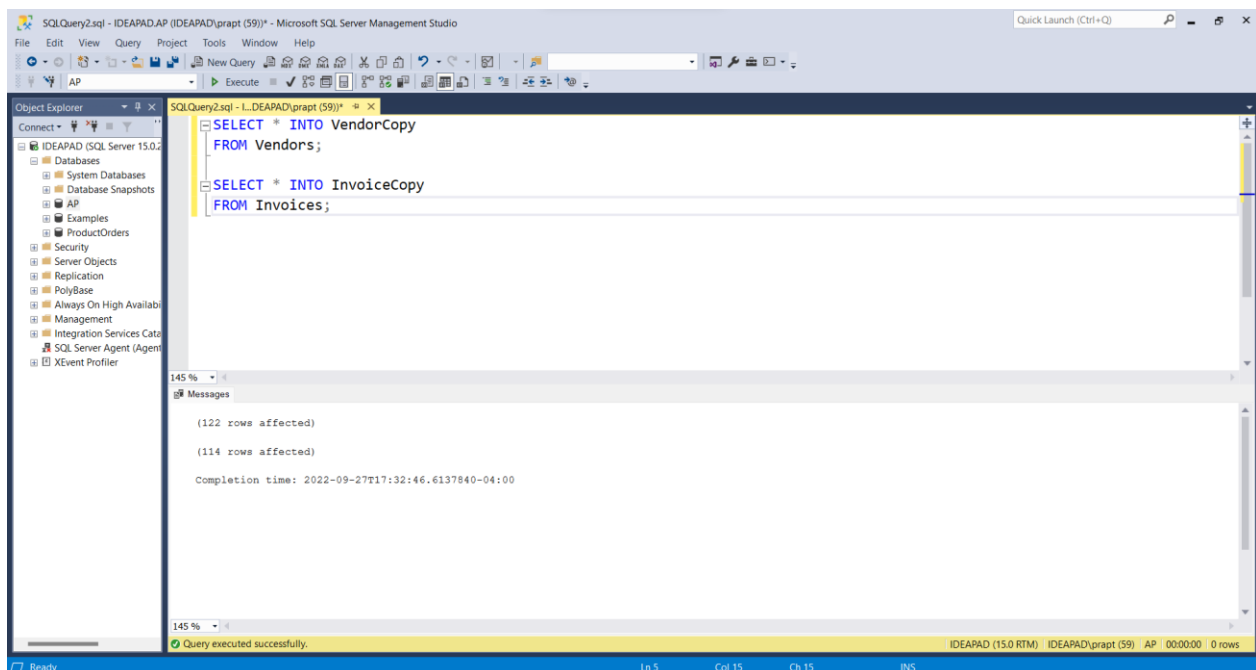


CSE 581 : INTRODUCTION TO DATABASE MANAGEMENT SYSTEMS**LAB : 4****DATE : 9/28/2022****Q1)** Create VendorCopy table and InvoiceCopy table.**Ans:** SELECT * INTO VendorCopy

FROM Vendors;

SELECT * INTO InvoiceCopy

FROM Invoices;



Comment: Here, SELECT statement is used to select the rows from Vendors and Invoices tables respectively for the two different queries. In the first query, records are selected from the Vendors table and are duplicated/copied into another table named VendorCopy using the SELECT INTO statement. Similarly, in the second query, records are selected from the Invoices table and are duplicated/copied into another table named Invoice Copy using SELECT INTO statement.

Remark: Here, SELECT INTO statement is used to copy the data and the table definition from the specified table.

Q2) 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:	4	InvoiceTotal:	\$750.48
TermsID:	2	InvoiceNumber:	UN-004-400
PaymentTotal:	\$100.00	InvoiceDueDate:	12/01/22
InvoiceDate:	10/01/22	CreditTotal:	\$7.50

Do we explicitly need to have an InvoiceID to insert?

Ans: INSERT INTO InvoiceCopy

VALUES (4, 'UN-004-400', '2022-01-10', 750.48, 100.00, 7.50, 2, '2022-01-12', NULL)

Before adding the new row:

SELECT * FROM InvoiceCopy

SQLQuery2.sql - IDEAPAD.AP (IDEAPAD\prapti (59)) - Microsoft SQL Server Enterprise Manager

Object Explorer: IDEAPAD (SQL Server 15.0.2) > Databases > AP > InvoiceCopy

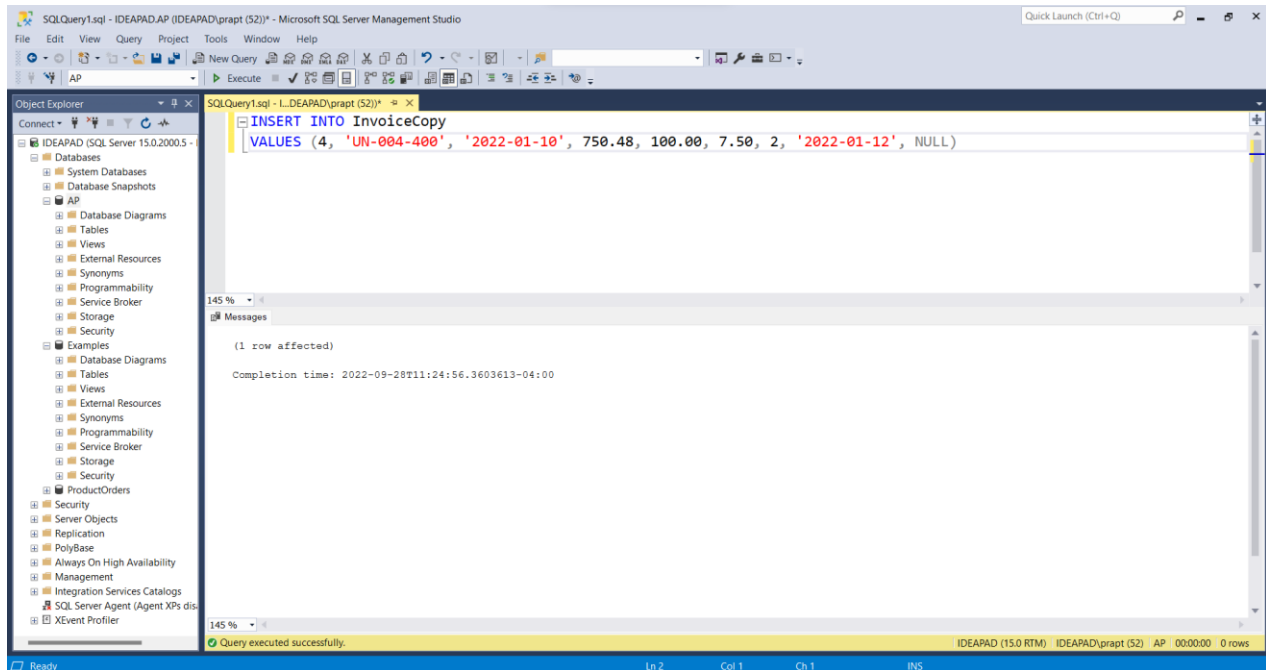
SQLQuery2.sql - L:\IDEAPAD\prapti (59) > SELECT * FROM InvoiceCopy;

Results: 145 %

InvoiceID	VendorID	InvoiceNumber	InvoiceDate	InvoiceTotal	PaymentTotal	CreditTotal	TermsID	InvoiceDueDate	PaymentDate
1	122	989319-457	2020-12-08 00:00:00	3813.33	3813.33	0.00	3	2021-01-08 00:00:00	2021-01-07 00:00:00
2	123	263253241	2020-12-10 00:00:00	40.20	40.20	0.00	3	2021-01-10 00:00:00	2021-01-14 00:00:00
3	123	963253234	2020-12-13 00:00:00	138.75	138.75	0.00	3	2021-01-13 00:00:00	2021-01-09 00:00:00
4	123	2-000-2993	2020-12-16 00:00:00	144.70	144.70	0.00	3	2021-01-16 00:00:00	2021-01-12 00:00:00
5	123	963253251	2020-12-16 00:00:00	15.50	15.50	0.00	3	2021-01-16 00:00:00	2021-01-11 00:00:00
6	123	963253261	2020-12-16 00:00:00	42.75	42.75	0.00	3	2021-01-16 00:00:00	2021-01-21 00:00:00
7	123	963253237	2020-12-21 00:00:00	172.50	172.50	0.00	3	2021-01-21 00:00:00	2021-01-22 00:00:00
8	89	125520-1	2020-12-24 00:00:00	95.00	95.00	0.00	1	2021-01-04 00:00:00	2021-01-01 00:00:00
9	121	97488	2020-12-24 00:00:00	601.95	601.95	0.00	3	2021-01-24 00:00:00	2021-01-21 00:00:00
10	10	263253250	2020-12-24 00:00:00	42.67	42.67	0.00	3	2021-01-24 00:00:00	2021-01-22 00:00:00
11	123	963253262	2020-12-25 00:00:00	42.50	42.50	0.00	3	2021-01-25 00:00:00	2021-01-20 00:00:00
12	96	177271-001	2020-12-26 00:00:00	662.00	662.00	0.00	2	2021-01-16 00:00:00	2021-01-13 00:00:00
13	95	111-829-10096	2020-12-30 00:00:00	16.33	16.33	0.00	2	2021-01-20 00:00:00	2021-01-23 00:00:00
14	115	25022117	2021-01-01 00:00:00	6.00	6.00	0.00	4	2021-02-10 00:00:00	2021-02-10 00:00:00
15	48	P02-8807787	2021-01-03 00:00:00	856.92	856.92	0.00	3	2021-02-02 00:00:00	2021-01-30 00:00:00
16	97	21-4748363	2021-01-03 00:00:00	9.95	9.95	0.00	2	2021-01-23 00:00:00	2021-01-22 00:00:00
17	123	4-321-2596	2021-01-05 00:00:00	10.00	10.00	0.00	3	2021-02-04 00:00:00	2021-02-05 00:00:00
18	123	963253242	2021-01-06 00:00:00	104.00	104.00	0.00	3	2021-02-05 00:00:00	2021-02-05 00:00:00
19	34	QP58872	2021-01-07 00:00:00	116.54	116.54	0.00	1	2021-01-17 00:00:00	2021-01-19 00:00:00
20	115	24863706	2021-01-10 00:00:00	6.00	6.00	0.00	4	2021-02-19 00:00:00	2021-02-15 00:00:00
21	119	10843	2021-01-11 00:00:00	4901.26	4901.26	0.00	2	2021-01-31 00:00:00	2021-01-29 00:00:00
22	123	963253235	2021-01-11 00:00:00	108.25	108.25	0.00	3	2021-02-10 00:00:00	2021-02-09 00:00:00
23	97	21-4323721	2021-01-13 00:00:00	9.95	9.95	0.00	2	2021-02-02 00:00:00	2021-01-28 00:00:00
24	113	77290	2021-01-13 00:00:00	1750.00	1750.00	0.00	5	2021-03-02 00:00:00	2021-03-05 00:00:00
25	123	963253246	2021-01-13 00:00:00	129.00	129.00	0.00	3	2021-02-12 00:00:00	2021-02-09 00:00:00
26	123	4-342-8069	2021-01-14 00:00:00	10.00	10.00	0.00	3	2021-02-13 00:00:00	2021-02-13 00:00:00
27	88	972110	2021-01-15 00:00:00	207.78	207.78	0.00	1	2021-01-25 00:00:00	2021-01-27 00:00:00
28	123	963253263	2021-01-16 00:00:00	109.50	109.50	0.00	3	2021-02-15 00:00:00	2021-02-15 00:00:00
29	108	121897	2021-01-19 00:00:00	450.00	450.00	0.00	4	2021-02-28 00:00:00	2021-03-03 00:00:00

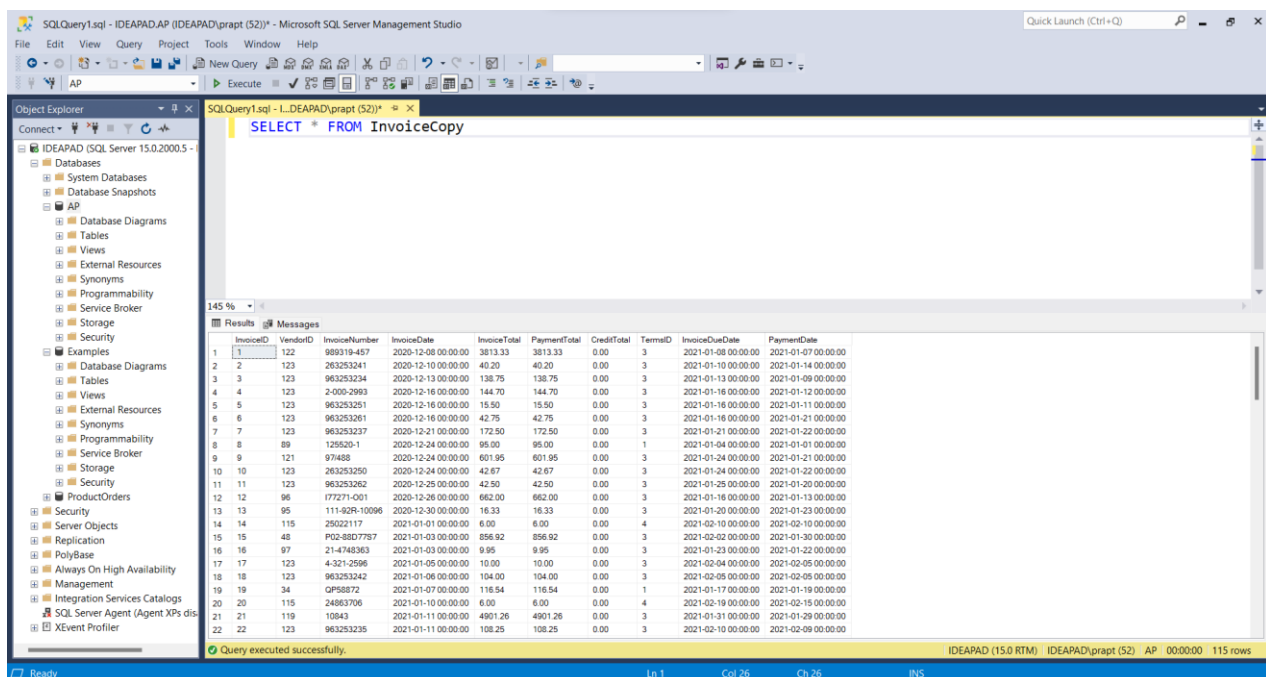
Query executed successfully. IDEAPAD (15.0 RTM) IDEAPAD\prapti (59) AP 00:00:00 114 rows

Inserting the new row:



After adding the new row:

SELECT * FROM InvoiceCopy



Comment: Here, INSERT INTO statement is used to insert the given row with the given values into the InvoiceCopy table. Here as we are adding values for all the columns of the table, we do not need to write the column names of the table. The values that are to be

added in the respective columns are written using VALUES keyword and are written in the same order as the columns of the table.

Remark: INSERT INTO is used to insert rows in the table by: 1) specifying the specific column names where you want to enter values, if you do not want to add values to all the columns 2) if you are adding values to all columns then without specifying the column names and writing values in the same order as the columns.

VALUES keyword is used to specify the values that are to be added in a single specific cell.

Q3) Write an UPDATE statement that modifies the VendorCopy table. Change the default account number to 970 for each vendor that has a default account number of 170. (USE SELECT statement to verify data changes in the table before and after the modification).

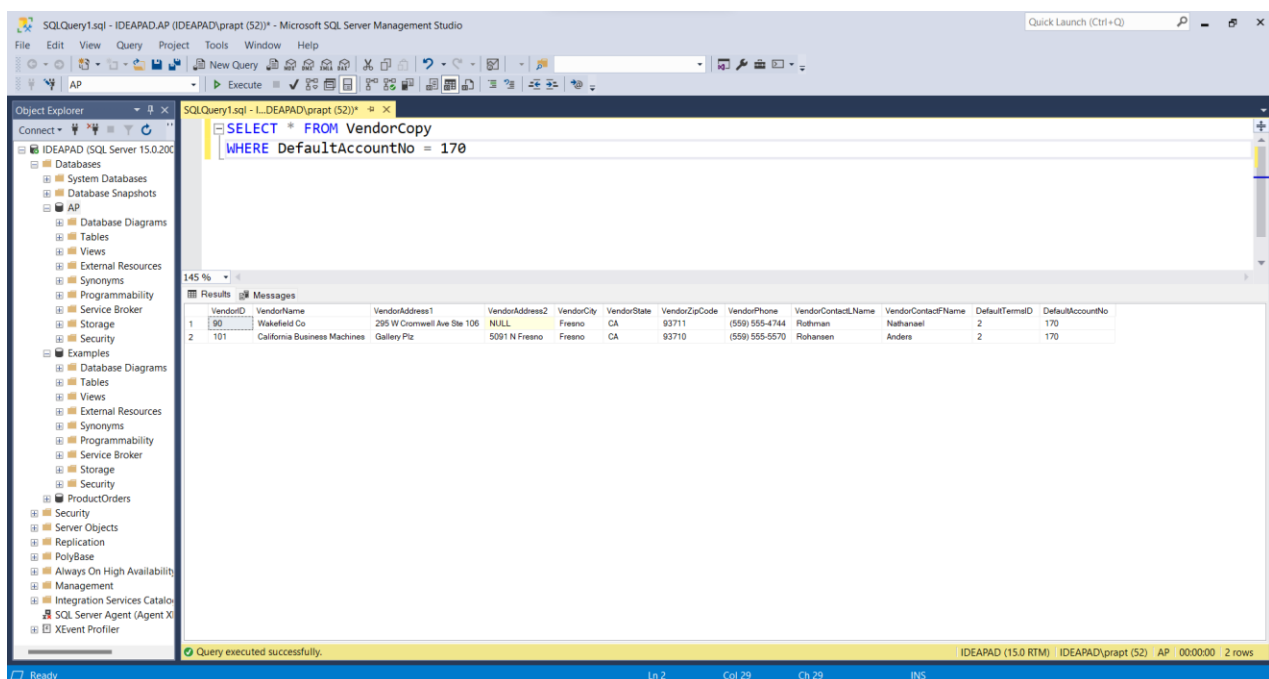
Ans: UPDATE VendorCopy SET DefaultAccountNo = 970

WHERE DefaultAccountNo = 170

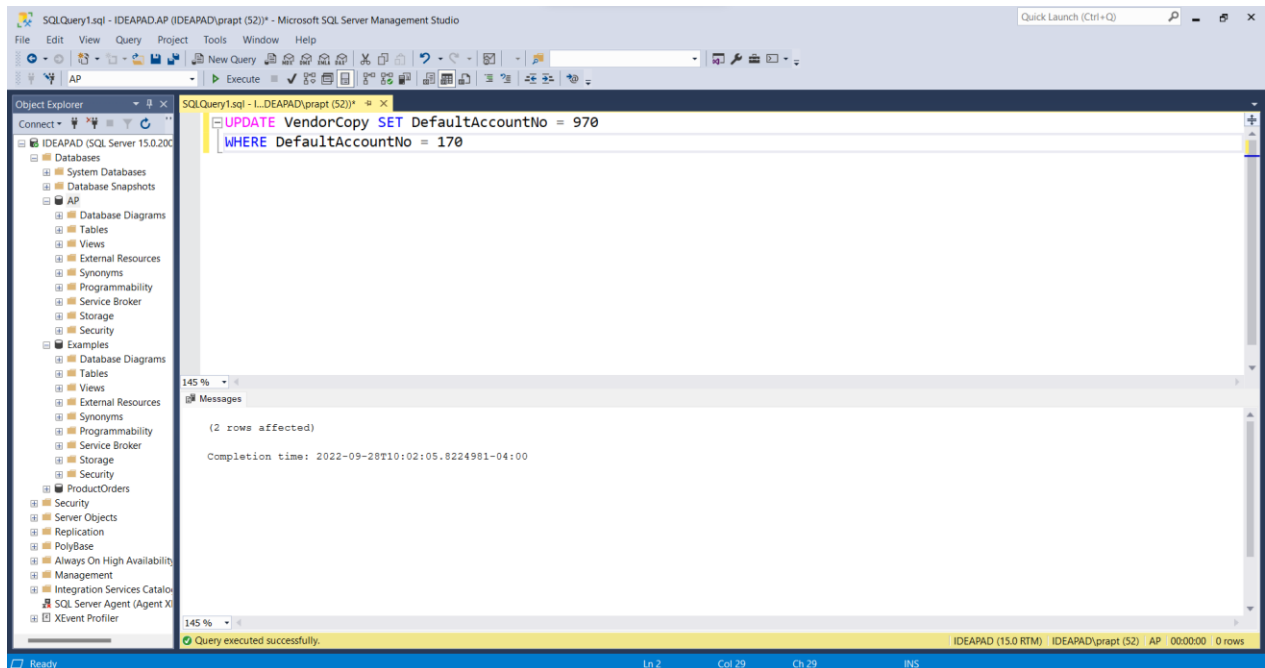
Before updating the records:

SELECT * FROM VendorCopy

WHERE DefaultAccountNo = 170



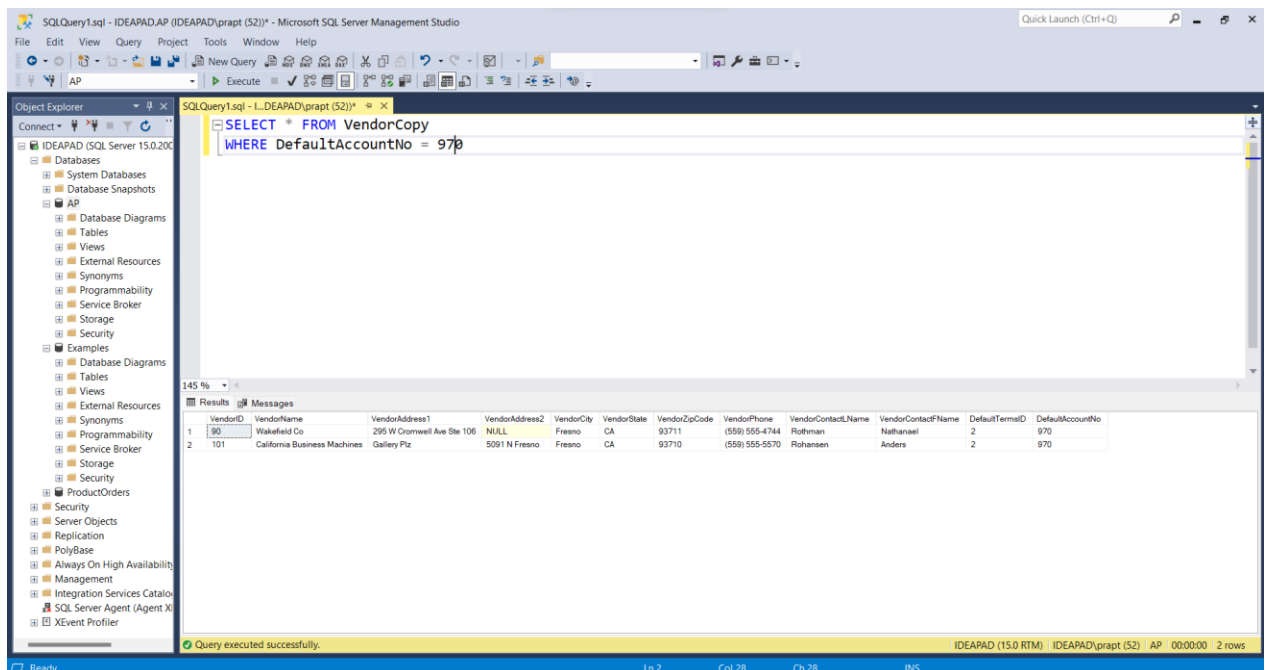
Updating the records:



After updating the records:

SELECT * FROM VendorCopy

WHERE DefaultAccountNo = 170



Comment: Here, UPDATE statement is used for updating/modification of the account number. Using UPDATE statement, the records with DefaultAccountNo 170 is changed to 970 using SET keyword. The condition is put using WHERE keyword.

Remark: To change/modify the existing data of the table, UPDATE statement is used.

SET is used to specify the value which is to be set instead of the previous value.

Q4) Write an UPDATE statement that modifies the InvoiceCopy table. Change the TermsID to 3 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).

Ans: UPDATE InvoiceCopy SET TermsID = '3'

```
WHERE VendorID IN(SELECT VendorID
                    FROM VendorCopy
                    WHERE DefaultTermsID = '2')
```

Before Updating the records:

```
SELECT VendorCopy.VendorID, InvoiceCopy.TermsID, VendorCopy.DefaultTermsID
FROM VendorCopy JOIN InvoiceCopy
ON VendorCopy.VendorID = InvoiceCopy.VendorID
WHERE DefaultTermsID = '2'
```

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure for 'IDEAPAD (SQL Server 15.0.2008)'. The central query window shows the following SQL query:

```
SELECT VendorCopy.VendorID, InvoiceCopy.TermsID, VendorCopy.DefaultTermsID
FROM VendorCopy JOIN InvoiceCopy
ON VendorCopy.VendorID = InvoiceCopy.VendorID
WHERE DefaultTermsID = '2'
```

The Results pane displays the following data:

VendorID	TermsID	DefaultTermsID
96	2	2
95	2	2
97	2	2
119	2	2
97	2	2
95	2	2
83	2	2
95	2	2
95	2	2
81	2	2
80	2	2
94	2	2
95	2	2
95	2	2
82	2	2
90	2	2
83	2	2
80	2	2

The status bar at the bottom indicates 'Query executed successfully.' and '18 rows'.

Updating the records:

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure for 'IDEAPAD (SQL Server 15.0.2008)'. The central query window shows the following SQL query:

```
UPDATE InvoiceCopy SET TermsID = '3'
WHERE VendorID IN (SELECT VendorID
FROM VendorCopy
WHERE DefaultTermsID = '2')
```

The Messages pane displays the following information:

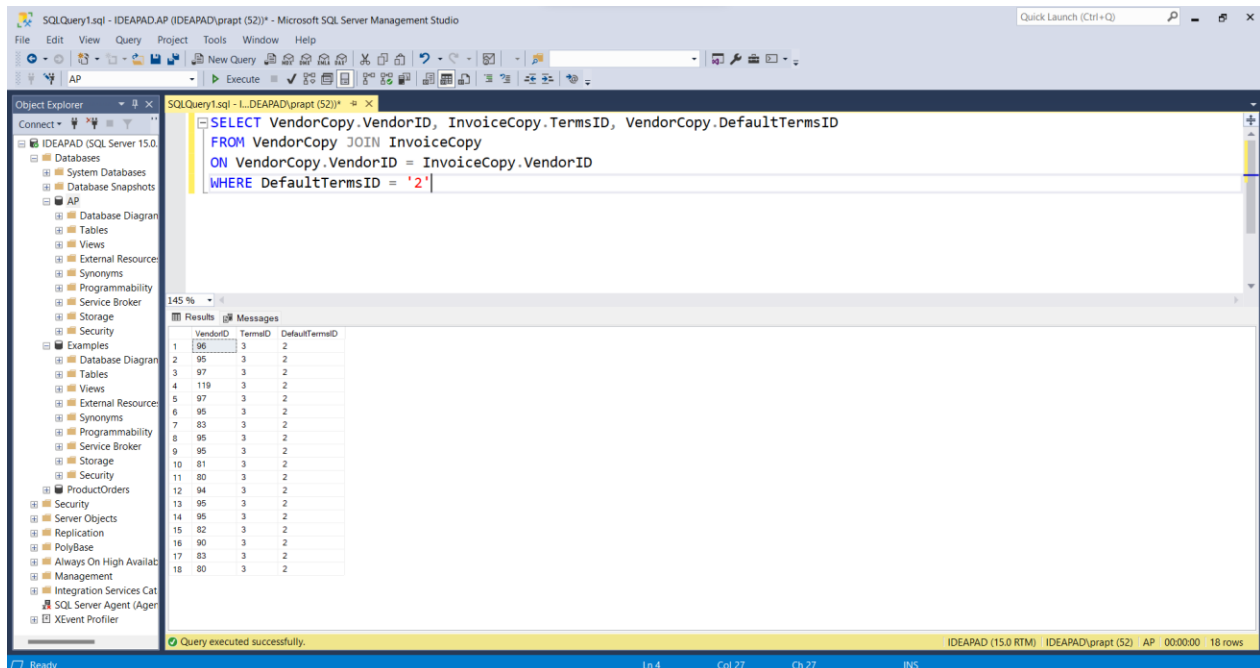
```
(18 rows affected)
Completion time: 2022-09-28T10:41:34.5969905-04:00
```

The status bar at the bottom indicates 'Query executed successfully.' and '0 rows'.

After updating the records:

```
SELECT VendorCopy.VendorID, InvoiceCopy.TermsID, VendorCopy.DefaultTermsID
FROM VendorCopy JOIN InvoiceCopy
ON VendorCopy.VendorID = InvoiceCopy.VendorID
```

WHERE DefaultTermsID = '2'



Comment: Here, an UPDATE statement is used to update/modify the records of the InvoiceCopy table. The records in the VendorCopy table with DefaultTermsID as 2 are selected and the VendorID of the records of the InvoiceCopy table that are from the pool of the subquery's result set are to be changed. This is done using the IN keyword on the subquery. The TermsID of these records are then set to 3 using SET keyword.

Remark: To change/modify the existing data of the table, an UPDATE statement is used. Subqueries can also be used to select certain records that are to be changed.

Q5) Write a DELETE statement that deletes all vendors in the state of Illinois from the VendorCopy table. (USE SELECT statement to verify data changes in the table before and after the modification).

Ans: DELETE FROM VendorCopy

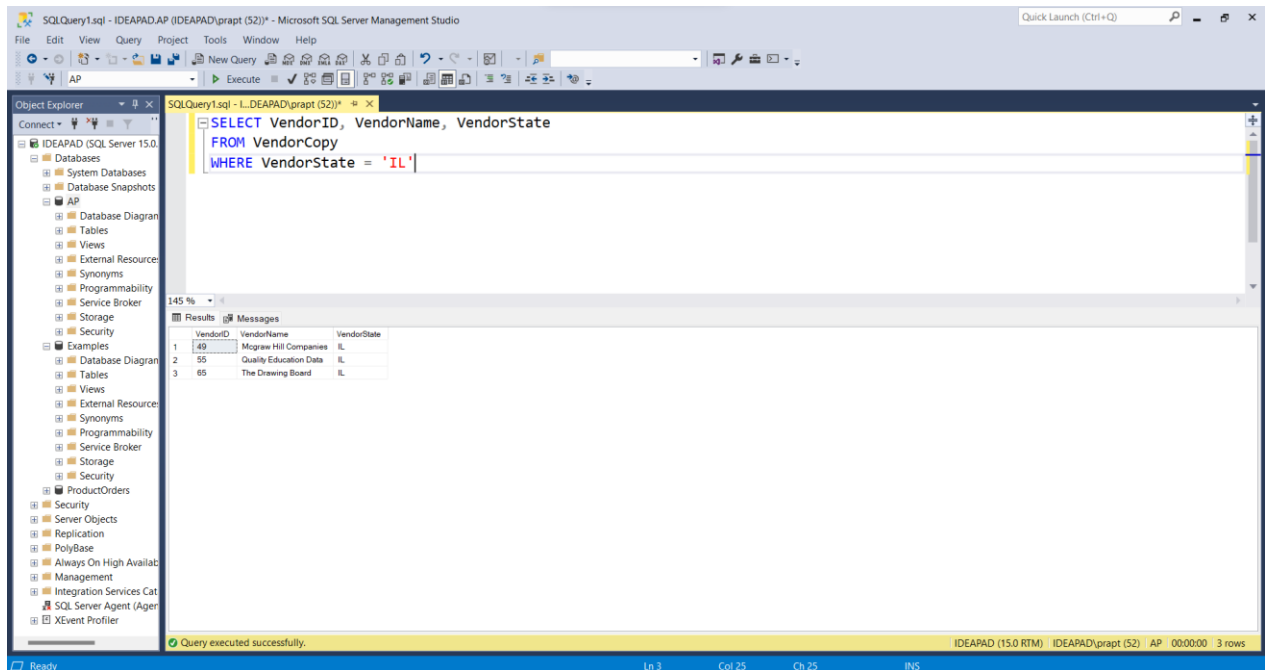
WHERE VendorState = 'IL'

Before deleting records:

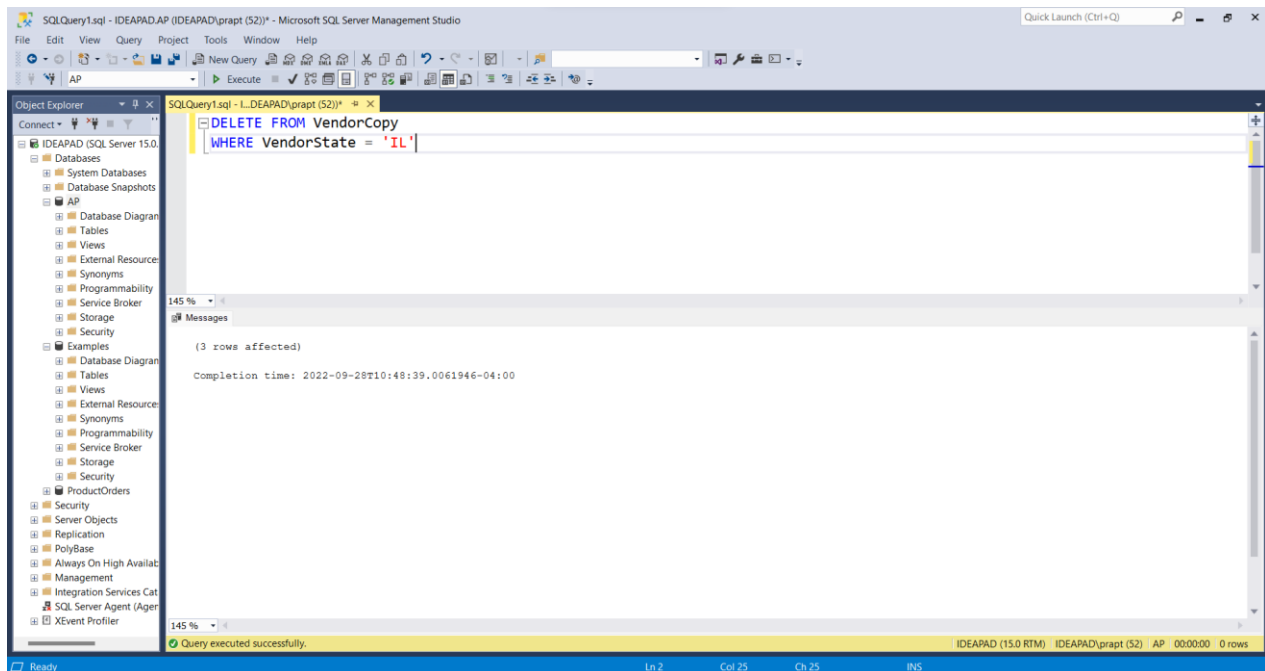
SELECT VendorID, VendorName, VendorState

FROM VendorCopy

WHERE VendorState = 'IL'



Deleting the records:

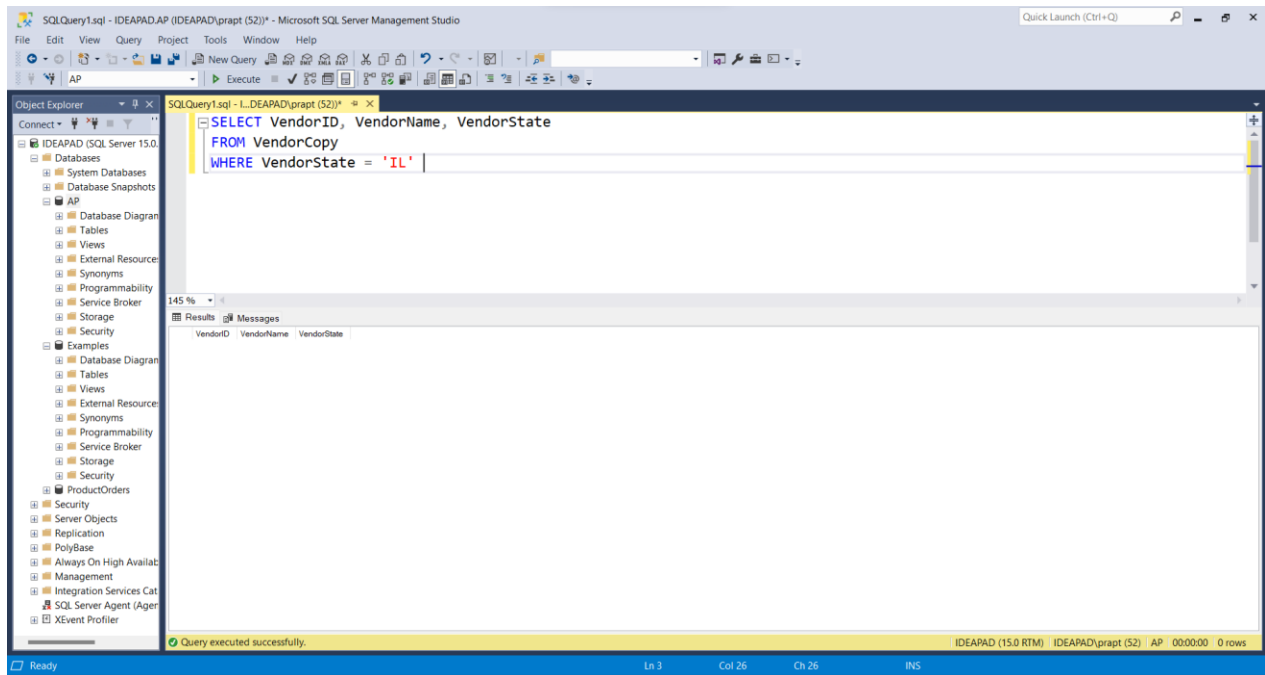


After deleting the records:

SELECT VendorID, VendorName, VendorState

FROM VendorCopy

WHERE VendorState = 'IL'



Comment: Here, DELETE statement is used to delete the records from the VendorCopy table. WHERE clause is used to specify the condition of the records that are to be deleted. So, the records with VendorState as 'IL' (Illinois) are deleted.

Remark: DELETE statement is used to delete all or certain records from a table.

Q6) Write a DELETE statement for the VendorCopy table. Delete the vendors that are located in states from which no vendor has ever sent an invoice. (USE SELECT statement to verify data changes in the table before and after the modification)

Ans: DELETE FROM VendorCopy

WHERE VendorState NOT IN (SELECT VendorState

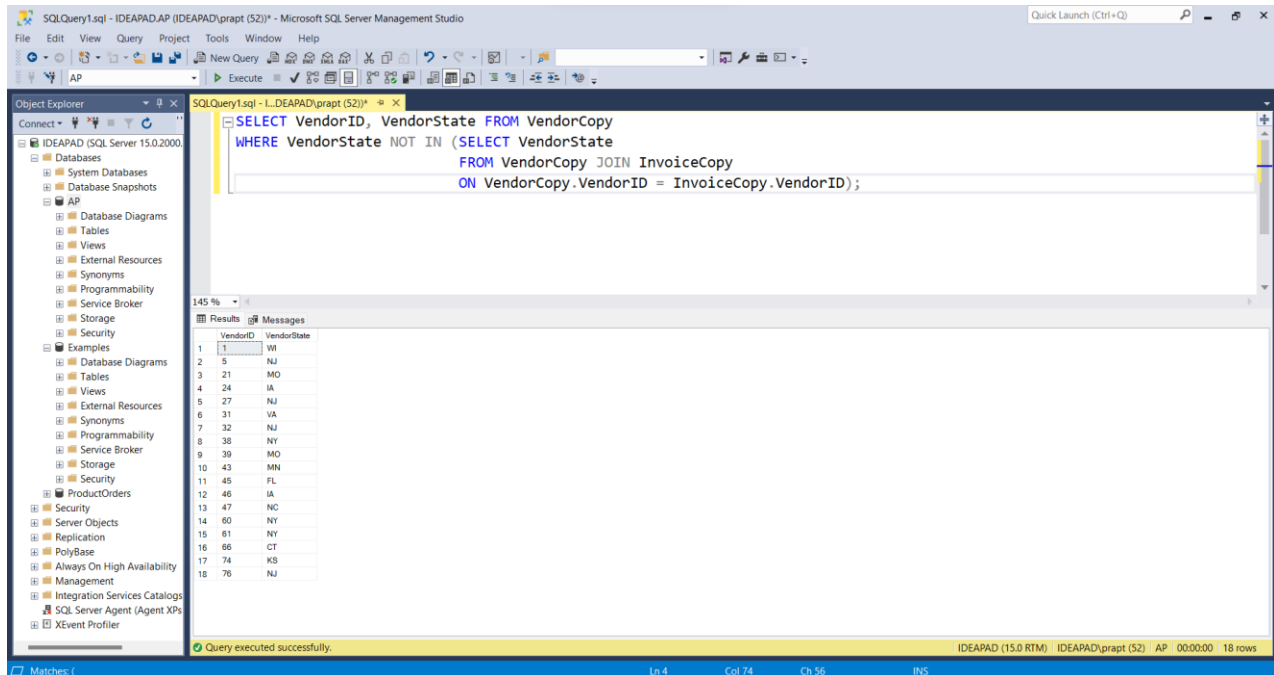
FROM VendorCopy JOIN InvoiceCopy

ON VendorCopy.VendorID = InvoiceCopy.VendorID);

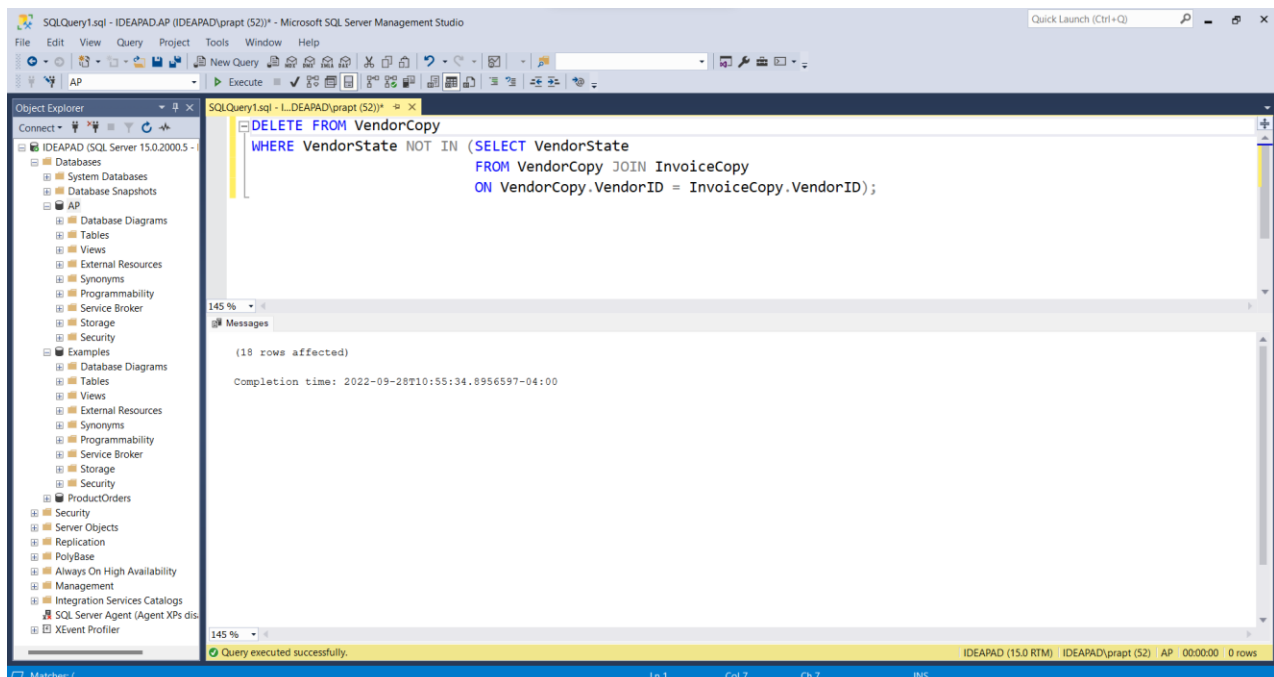
Before deleting records:

SELECT VendorID, VendorState FROM VendorCopy

WHERE VendorState NOT IN (SELECT VendorState
 FROM VendorCopy JOIN InvoiceCopy
 ON VendorCopy.VendorID = InvoiceCopy.VendorID);

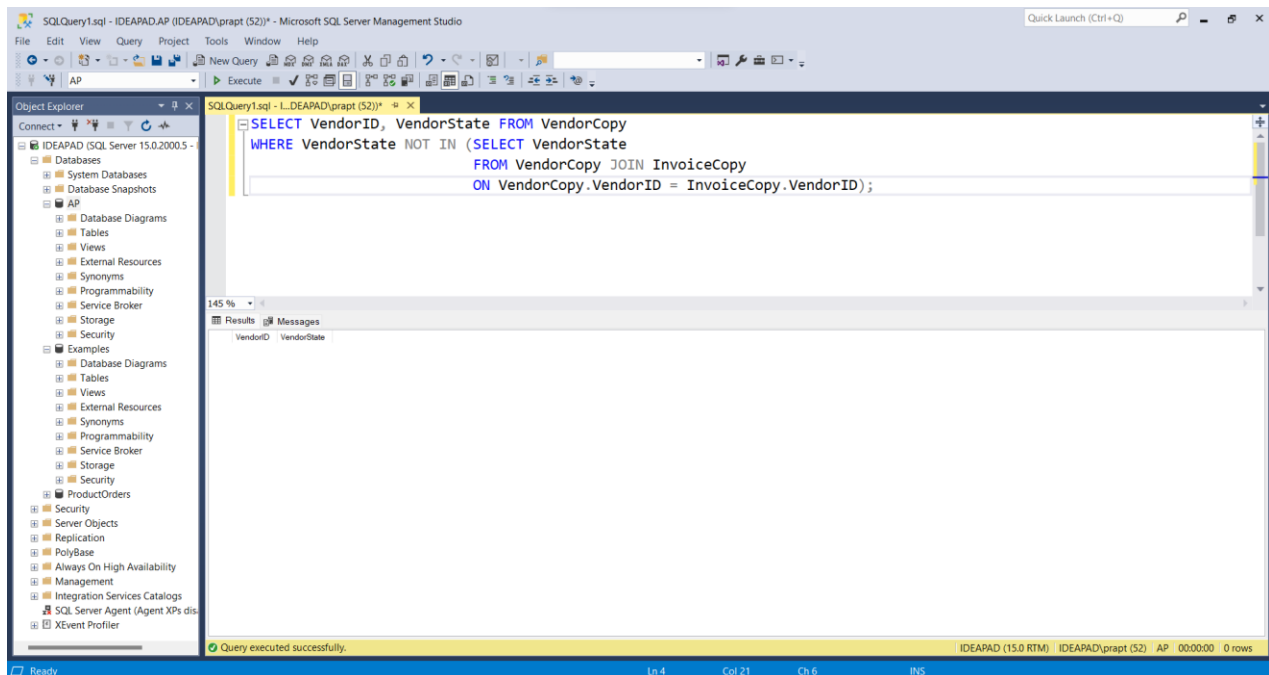


Deleting the records:



After deleting the records:

```
SELECT VendorID, VendorState FROM VendorCopy
WHERE VendorState NOT IN (SELECT VendorState
                           FROM VendorCopy JOIN InvoiceCopy
                           ON VendorCopy.VendorID = InvoiceCopy.VendorID);
```



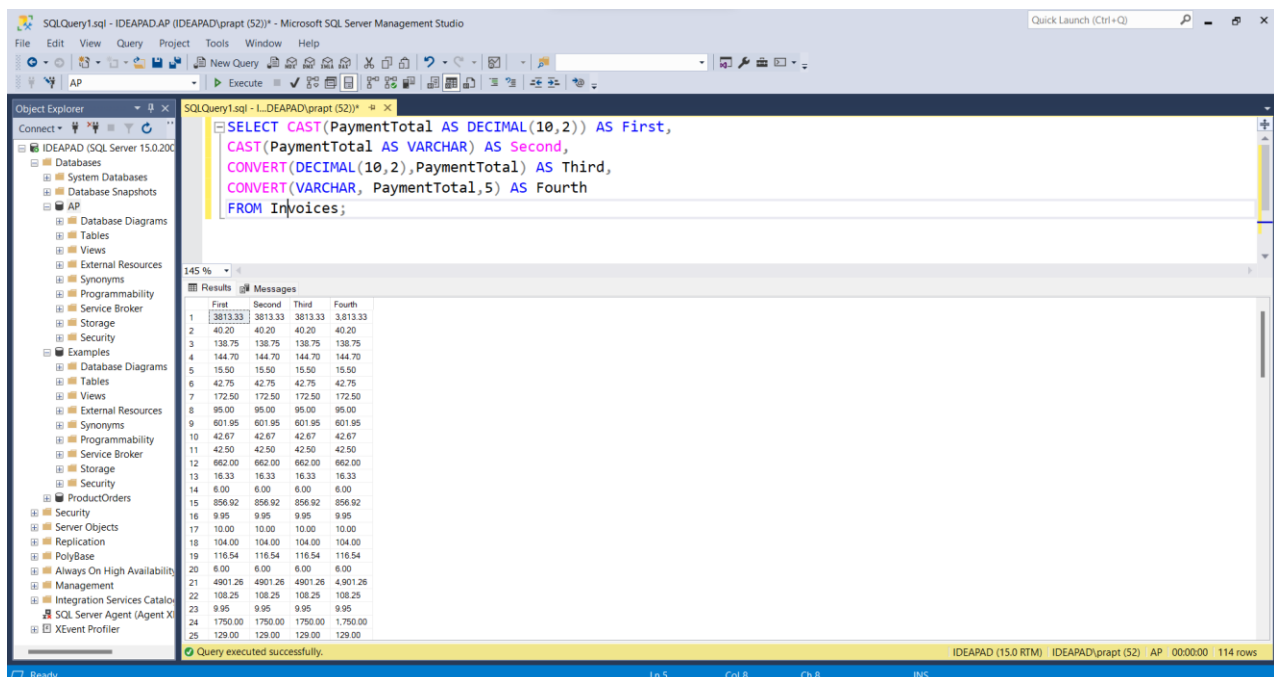
Comment: Here, DELETE statement is used to delete the records from the VendorCopy whose vendors are in such a list of states of vendors who have never sent an invoice. So, a subquery is written which selects the records of such vendors. SELECT statement is used in the subquery to select the VendorState from the JOIN of the VendorCopy and InvoiceCopy table on VendorID i.e., those records that have sent an invoice. NOT IN is used to delete those records that are not in the result set of the subquery.

Remark: NOT IN operator is used to select the values that are not in the specified result set.

Q7) Write a SELECT statement that returns four columns based on the PaymentTotal column of the Invoices table:

1. Use CAST function to return the first column as data type decimal with 2 digits to the right of the decimal point.
2. Use CAST to return the second column as a VARCHAR.
3. Use CONVERT function to return third column as the same type as the first column.
4. Use CONVERT to return the fourth column as a VARCHAR, using style 5.

Ans: SELECT CAST(PaymentTotal AS DECIMAL(10,2)) AS First,
 CAST(PaymentTotal AS VARCHAR) AS Second,
 CONVERT(DECIMAL(10,2),PaymentTotal) AS Third,
 CONVERT(VARCHAR, PaymentTotal,5) AS Fourth
 FROM Invoices;



Comment: Here, a SELECT statement is used to display different types of records by applying different functions to the PaymentTotal column of the Invoices table. For the First column, CAST function is used on the PaymentTotal column to return the data of the length of the data type DECIMAL with 2 digits to the right of the decimal point. For the Second column, CAST function is used on the PaymentTotal column to return the data in VARCHAR datatype. For the Third column, CONVERT function is used on the PaymentTotal column to convert the datatype to DECIMAL with 2 digits to the right of

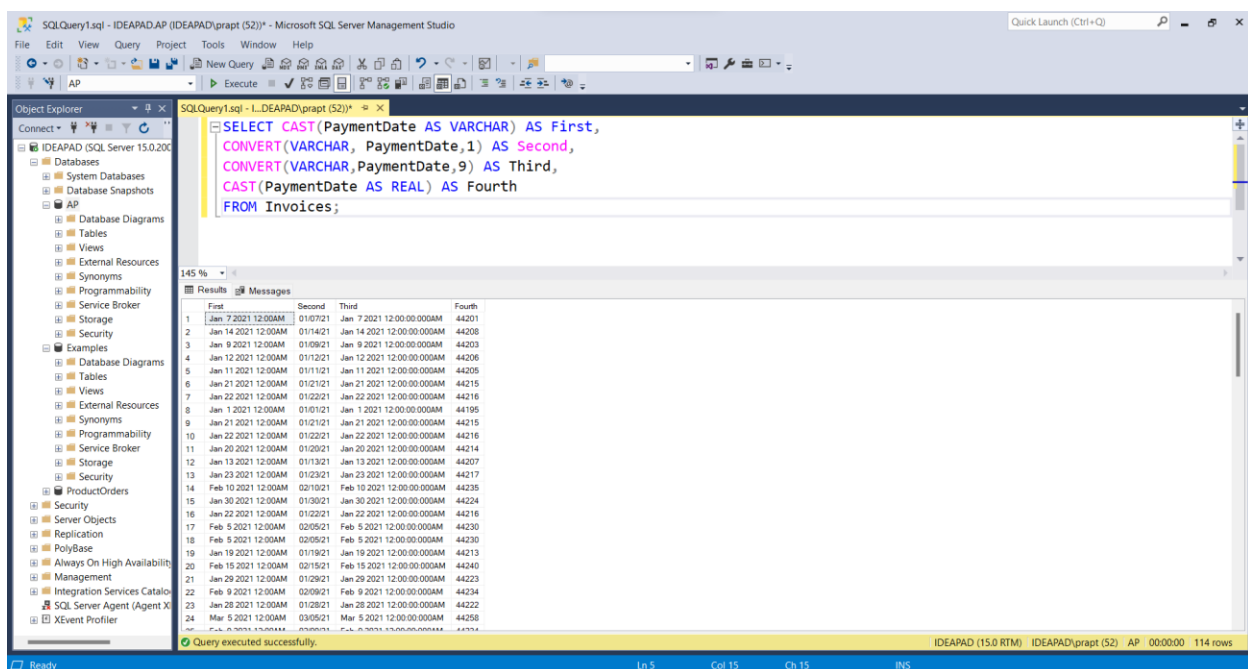
the decimal point. For the Fourth column, CONVERT function is used on the PaymentTotal column to convert the datatype to VARCHAR and format using style 5.

Remark: CAST function is used to convert data. CONVERT function is used to convert and format the data.

Q8) Write a SELECT statement that returns four columns based on the PaymentDate column of the Invoices table:

1. Use the CAST function to return the first column as data type VARCHAR.
2. Use the CONVERT function to return the second and third columns as a VARCHAR, using style 1 and style 9, respectively.
3. Use the CAST function to return the fourth column as a data type real.

Ans: SELECT CAST(PaymentDate AS VARCHAR) AS First,
 CONVERT(VARCHAR, PaymentDate,1) AS Second,
 CONVERT(VARCHAR,PaymentDate,9) AS Third,
 CAST(PaymentDate AS REAL) AS Fourth
 FROM Invoices;



Comment: Here, a SELECT statement is used to display different types of records by applying different functions to the PaymentDate column of the Invoices table. For the First column, CAST function is used on the PaymentDate column to return the data of the data type VARCHAR. For the Second column, CONVERT function is used on the PaymentDate column to return the data in VARCHAR datatype and format using style 1. For the Third column, CONVERT function is used on the PaymentDate column to convert the datatype to VARCHAR and format with style 9. For the Fourth column, CONVERT function is used on the PaymentDate column to convert the datatype to REAL.

Remark for the lab: Concepts related to data manipulation and data types are used including the basic keywords, clauses and statements in this lab.