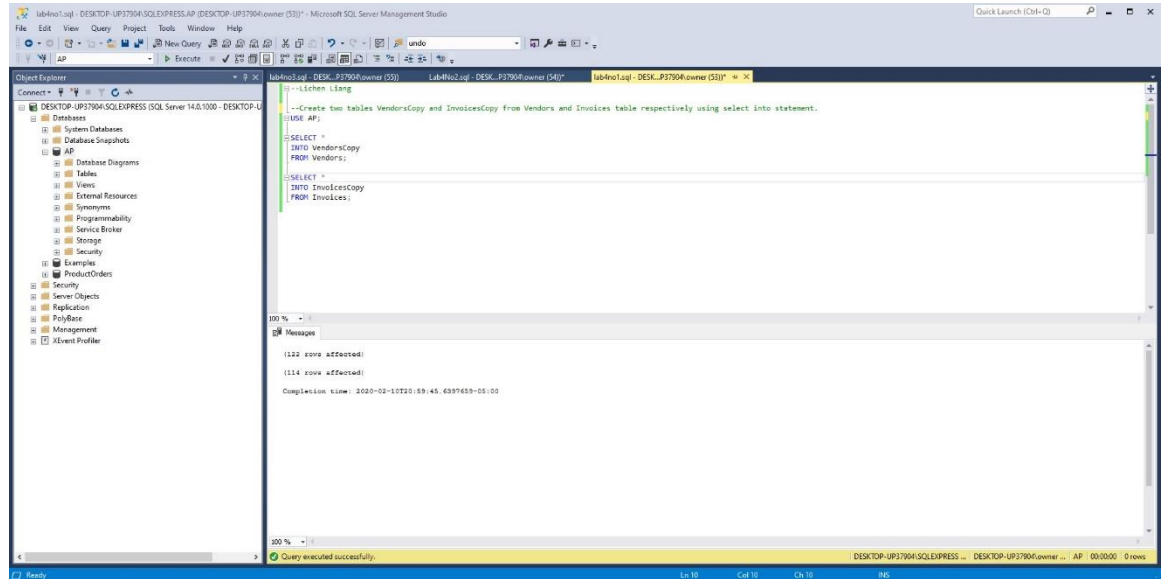


ADDITIONAL COMMENTS ARE UNDER THE SCREENSHOT

1. USE AP;

```
SELECT *  
INTO VendorsCopy  
FROM Vendors;
```

```
SELECT *  
INTO InvoicesCopy  
FROM Invoices;
```



This creates a copy of the table. Only the columns and data are copied. Not the attributes such as primary key, foreign key, indexes, etc.

2. USE AP;

```
SELECT * FROM InvoicesCopy;
```

```
INSERT INTO InvoicesCopy
```

```
(VendorID, TermsID, PaymentTotal, InvoiceDate, InvoiceTotal, InvoiceNumber, InvoiceDueDate,  
CreditTotal)
```

```
VALUES(1, 2, 0.00, '2015-08-24', 334.21, 'ZU_009-900', '2015-09-01', 0.00);
```

```
SELECT * FROM InvoicesCopy;
```

The top screenshot shows the Microsoft SQL Server Enterprise Manager interface. The query window displays the following SQL code:

```
--Lichen Liang  
--Insert a row into InvoicesCopy table, with columns and its values specified, using insert into statement.  
--USE AP;  
  
SELECT * FROM InvoicesCopy;  
  
INSERT INTO InvoicesCopy  
(VendorID, TermsID, PaymentTotal, InvoiceDate, InvoiceTotal, InvoiceNumber, InvoiceDueDate, CreditTotal)  
VALUES(1, 2, 0.00, '2015-08-24', 334.21, 'ZU_009-900', '2015-09-01', 0.00);  
  
SELECT * FROM InvoicesCopy;
```

The Results tab shows a table with the following data:

InvoiceID	VendorID	InvoiceNumber	InvoiceDate	InvoiceTotal	PaymentTotal	CreditTotal	TermsID	InvoiceDueDate	PaymentDate
107	107	122	2016-03-24 00:00:00	3609.99	3609.99	0.00	3	2016-04-23 00:00:00	2016-04-23 00:00:00
108	108	123	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	2016-03-25 00:00:00	1000.45	1000.45	0.00	3	2016-04-24 00:00:00	2016-04-24 00:00:00
110	110	80	2016-03-28 00:00:00	90.36	0.00	0.00	2	2016-04-17 00:00:00	NULL
111	111	123	2016-03-30 00:00:00	22.57	22.57	0.00	3	2016-04-29 00:00:00	2016-05-03 00:00:00
112	112	110	2016-03-31 00:00:00	10076.06	0.00	0.00	3	2016-04-30 00:00:00	NULL
113	113	37	2016-04-01 00:00:00	224.00	0.00	0.00	3	2016-04-30 00:00:00	NULL
114	114	123	2016-04-02 00:00:00	127.75	127.75	0.00	3	2016-05-01 00:00:00	2016-05-04 00:00:00
115	115	1	2016-08-24 00:00:00	334.21	0.00	0.00	2	2016-09-01 00:00:00	NULL

The bottom screenshot shows the same query window with the Messages tab selected. It displays the following messages:

```
(114 rows affected)  
(1 row affected)  
(115 rows affected)  
  
Completion time: 2020-02-11T21:51:51.4102901-08:00
```

The values are inserted into the row in corresponding to the column names, so the order matters. We don't need to insert InvoiceID. It's the primary key.

3. USE AP;

```
SELECT * FROM VendorsCopy;
```

```
UPDATE VendorsCopy  
SET DefaultAccountNo = 530  
WHERE DefaultAccountNo = 521;
```

```
SELECT * FROM VendorsCopy;
```

The first screenshot shows the SQL Server Enterprise Manager interface with a query window open. The query is as follows:

```
--change the default account number from 521 to 530 using update statement.  
--USE AP;  
  
SELECT * FROM VendorsCopy;  
  
UPDATE VendorsCopy  
SET DefaultAccountNo = 530  
WHERE DefaultAccountNo = 521;  
  
SELECT * FROM VendorsCopy;
```

The query results are displayed in a table with the following columns: VendorID, VendorName, VendorAddress1, VendorAddress2, VendorCity, VendorState, VendorZipCode, VendorPhone, VendorContactName, VendorContactPhone, DefaultTermID, and DefaultAccountNo. The results show 12 rows of data, with the last row (VendorID 12) having a DefaultAccountNo of 521.

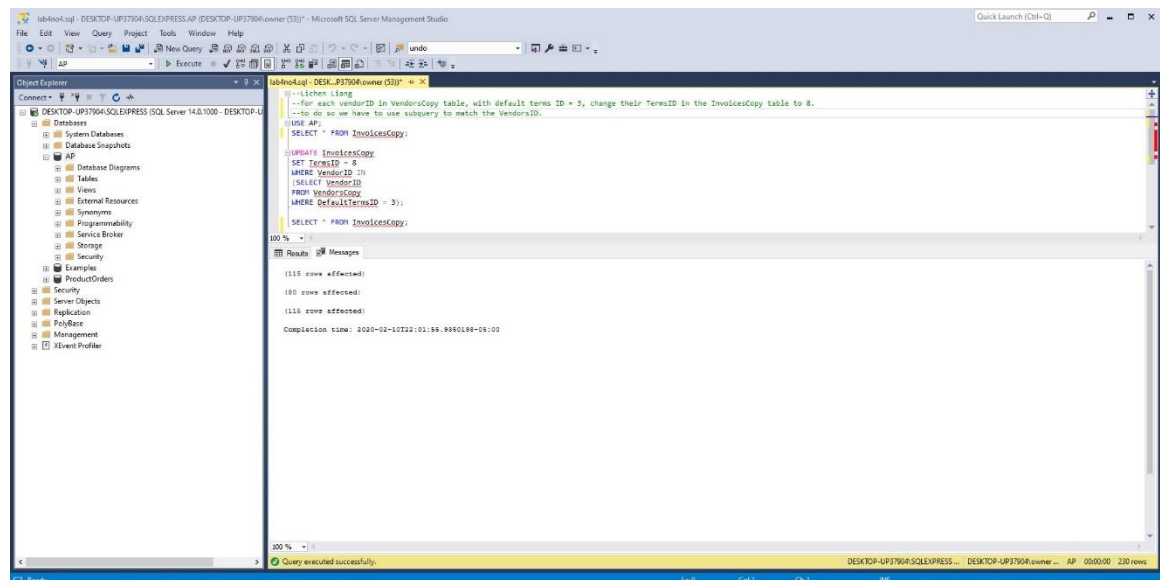
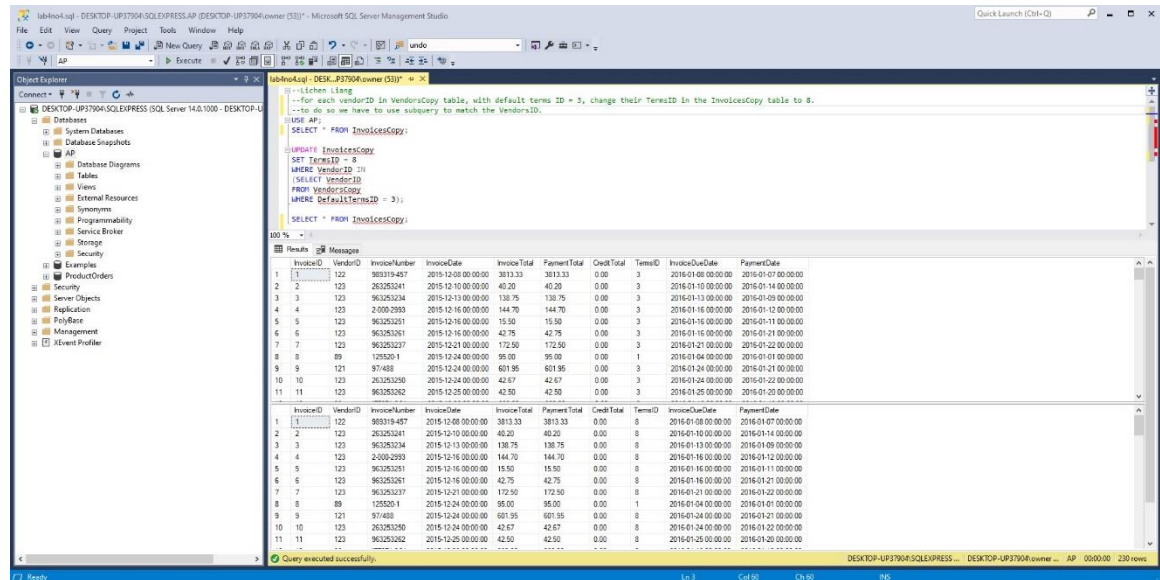
The second screenshot shows the same query window after execution. The status bar at the bottom indicates that the query was executed successfully, affecting 122 rows. The completion time is 2020-12-10T21:54:35.8910284-05:00.

First, we find the rows with default account number = 521 using WHERE statement, then using SET to update the entry.

4. USE AP;
SELECT * FROM InvoicesCopy;

UPDATE InvoicesCopy
SET TermsID = 8
WHERE VendorID IN
(SELECT VendorID
FROM VendorsCopy
WHERE DefaultTermsID = 3);

SELECT * FROM InvoicesCopy;



The subquery returns all vendorIDs with default terms ID =3, then using this VendorID, SET all the TermsID to 8 in another table.

5. USE AP;
SELECT * FROM VendorsCopy;

DELETE VendorsCopy
WHERE VendorCity = 'Washington';

SELECT * FROM VendorsCopy;

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The left pane displays the Object Explorer with the 'AP' database selected. The right pane shows a query window with the following SQL code:

```
-- In the VendorsCopy table, delete all rows with vendors whose VendorCity is Washington, using delete statement.  
USE AP;  
SELECT * FROM VendorsCopy;  
DELETE VendorsCopy  
WHERE VendorCity = 'Washington';  
SELECT * FROM VendorsCopy;
```

The 'Results' tab shows the output of the queries. The first query returns 17 rows of vendor data. The second query (DELETE) returns 0 rows affected. The third query (SELECT) returns 16 rows of vendor data, indicating that one row was successfully deleted.

VendorID	VendorName	VendorAddress1	VendorAddress2	VendorCity	VendorState	VendorZipCode	VendorPhone	VendorContactName	VendorContactPhone	DefaultTermsID	DefaultAccountNo
1	US Postal Service	Attn: Sup. Window Services	PO Box 7005	Madison	WI	53707	(800) 555-1205	Alberto	Francisco	1	582
2	National Information Data Ctr	PO Box 9621		Washington	DC	20090	(202) 555-6950	Irene	Alex	3	540
3	Register of Copyrights	Library of Congress		Washington	DC	20540	NULL	Laura	Lukas	3	453
4	Jabarak	1990 Westwood Blvd Ste 260		Los Angeles	CA	90025	(800) 555-8725	Quinn	Kenzie	3	572
5	Newbridge Book Clubs	3000 Orinda Drive		Washington	DC	07882	(800) 555-9980	Marka	Michelle	4	334
6	California Chamber Of Commerce	3255 Ralston Cr		Sacramento	CA	95827	(916) 555-6670	Mauro	Arton	3	572
7	Tower-Sunderland's Mailing Svcs	Kevin Winder	3441 W MacArthur Blvd	Santa Ana	CA	92704	NULL	Maureen	Ted	3	540
8	BFI Industries	PO Box 3369		Fresno	CA	93792	(559) 555-1551	Kathryn	Erin	3	530
9	Pacific Gas & Electric	Box 52001		San Francisco	CA	94152	(800) 555-6081	Anthony	Kathryn	3	530
10	Robbins Mobile Lock And Key	4659 N Fresno		Fresno	CA	93726	(559) 555-9175	Leigh	Bill	2	523
11	Bill Marvin Electric Inc	4653 E Home		Fresno	CA	93703	(559) 555-5106	Hudley	Kathryn	2	523
12	City Of Fresno	PO Box 2069		Fresno	CA	93710	(559) 555-9999	Mayle	Kendall	3	574
13	Golden Eagle Insurance Co	PO Box 59526		San Diego	CA	92186	NULL	Blanca	Korah	3	590
14	Espeleta Inc	4420 N. First Street, Suite 108		Fresno	CA	93726	(559) 555-9636	Quentin	Heaven	3	569
15	ASC Signs	1520 N. Sierra Vista		Fresno	CA	93703	NULL	Owen	Elizabeth	1	546
16	Internal Revenue Service			Fresno	CA	93888	NULL	Aileen	Joan	1	235
17	Blaumhert & Johnson Associates	77771 Vintennes	NI 1	Blaumhert	CA	57661	(714) 555-3647	Klinton	Connolly	3	540

The screenshot shows the Microsoft SQL Server Enterprise Manager interface after the DELETE statement has been executed. The 'Results' tab displays the following messages:

```
(17 rows affected)  
(0 rows affected)  
(16 rows affected)  
Completion time: 2020-02-18T22:07:21.4280157-08:00
```

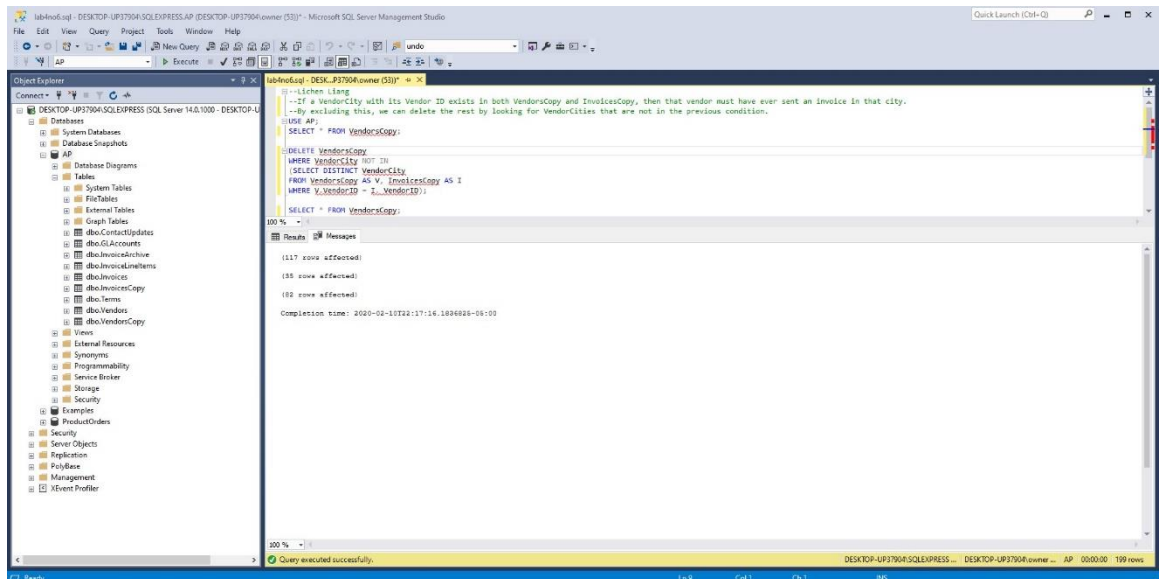
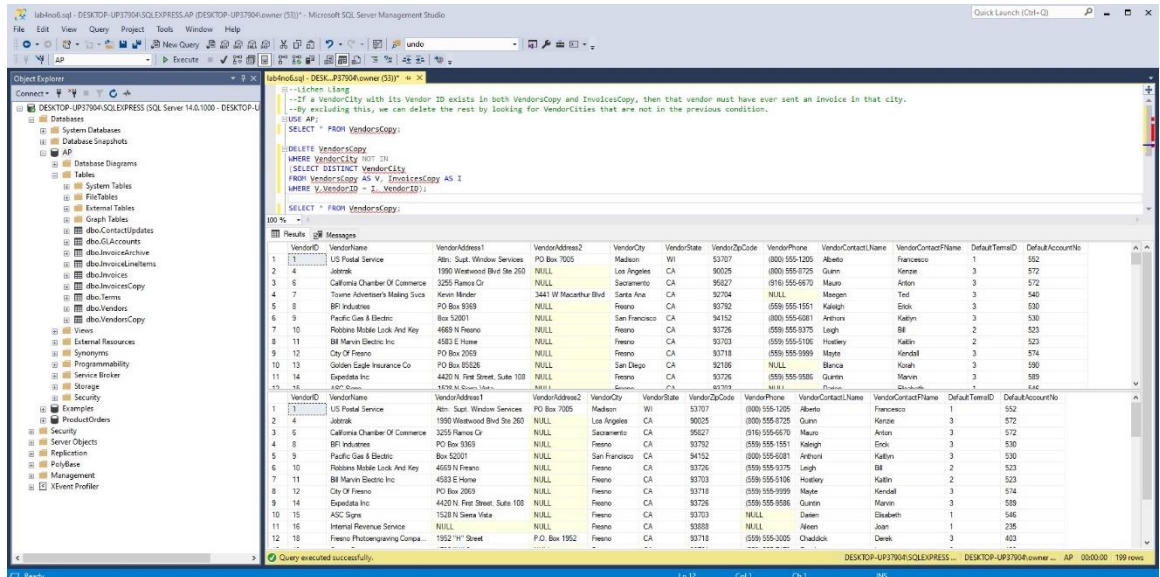
The status bar at the bottom indicates that the query was executed successfully and that 16 rows were affected.

First find all entries using WHERE statement such that VendorCity is Washington, then delete this entry using DELETE statement.

6. USE AP;
SELECT * FROM VendorsCopy;

```
DELETE VendorsCopy
WHERE VendorCity NOT IN
(SELECT DISTINCT VendorCity
FROM VendorsCopy AS V, InvoicesCopy AS I
WHERE V.VendorID = I.VendorID);
```

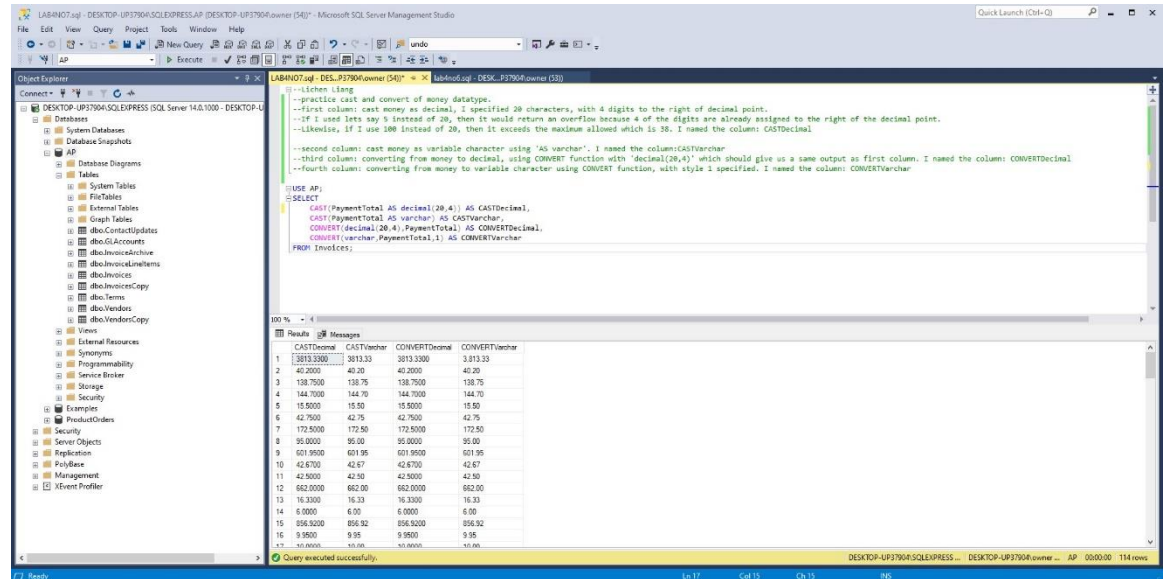
SELECT * FROM VendorsCopy;



We need to use DISTINCT in the subquery to get non-duplicated value for VendorCity. The subquery gets the ones that have sent an invoice, then simply using NOT IN in the WHERE statement to filter the ones who have not sent an invoice.

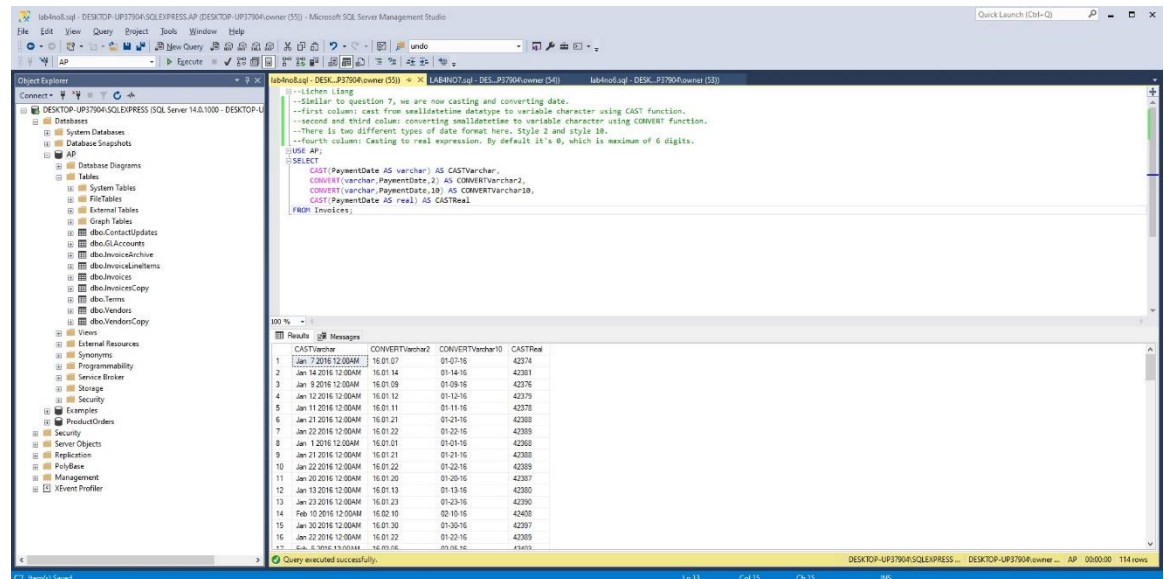
7. USE AP;

```
SELECT
CAST(PaymentTotal AS decimal(20,4)) AS CASTDecimal,
CAST(PaymentTotal AS varchar) AS CASTVarchar,
CONVERT(decimal(20,4),PaymentTotal) AS CONVERTDecimal,
CONVERT(varchar,PaymentTotal,1) AS CONVERTVarchar
FROM Invoices;
```



8. USE AP;

```
SELECT
CAST(PaymentDate AS varchar) AS CASTVarchar,
CONVERT(varchar,PaymentDate,2) AS CONVERTVarchar2,
CONVERT(varchar,PaymentDate,10) AS CONVERTVarchar10,
CAST(PaymentDate AS real) AS CASTReal
FROM Invoices;
```



Remarks

In this lab we practiced data manipulation using UPDATE, SET, INSERT, DELETE, etc. Using the knowledge that we learned from class, we set up conditions, search requirements, and knowledge from previous labs to meet this lab's requirement. By comparing before and after using query, we can see the changes that have been done to a table.

We also practiced CAST and CONVERT functions to experience with different data types and know what are the limits, such as what will work and what will not work.

I think this lab is very efficient practice for lecture. The challenge would be more complicated and specific requirements.