Chapter 7

How to insert, update, and delete data

The syntax of the SELECT INTO statement

```
SELECT select_list
 INTO table_name
 FROM table_source
 [WHERE search_condition]
 [GROUP BY group_by_list]
 [HAVING search_condition]
 [ORDER BY order_by_list] ;
```

Create a complete copy of the Invoices table

```
SELECT *
INTO InvoiceCopy
FROM Invoices;
```

```
(114 row(s) affected)
```

Create a partial copy of the Invoices table

```
SELECT *
INTO OldInvoices
FROM Invoices
WHERE InvoiceTotal - PaymentTotal - CreditTotal = 0;
```

```
(103 row(s) affected)
```

Create a table with summary rows

```
SELECT VendorID, SUM(InvoiceTotal) AS SumOfInvoices
INTO VendorBalances
FROM Invoices
WHERE InvoiceTotal - PaymentTotal - CreditTotal <> 0
GROUP BY VendorID;
```

```
(7 row(s) affected)
```

Delete a table

```
DROP TABLE InvoiceCopy;
```

Warnings

- When you use the SELECT INTO statement to create a table, only the column definitions and data are copied.
- Definitions of primary keys, foreign keys, indexes, default values, and so on are not included in the new table.

The syntax of the INSERT statement

```
INSERT [INTO] table_name [(column_list)]
[DEFAULT] VALUES (expression_1 [, expression_2]...)
[, (expression_1 [, expression_2]...)...]
```

The values for a new row in the Invoices table

Column	Value
InvoiceID	(Next available unique ID)
VendorID	97
InvoiceNumber	456789
InvoiceDate	4/01/2016
InvoiceTotal	8,344.50
PaymentTotal	0
CreditTotal	0
TermsID	1
InvoiceDueDate	4/31/2016
PaymentDate	null

Insert the row without using a column list

```
INSERT INTO InvoiceCopy
VALUES (97, '456789', '2016-04-01', 8344.50, 0, 0, 1,
        '2016-04-30', NULL);
```

Insert the row using a column list

```
INSERT INTO InvoiceCopy
    (VendorID, InvoiceNumber, InvoiceTotal,
    PaymentTotal, CreditTotal, TermsID, InvoiceDate,
    InvoiceDueDate)
VALUES
    (97, '456789', 8344.50, 0, 0, 1, '2016-04-01',
    '2016-04-30');
```

The response from the system

```
(1 row(s) affected)
```

Insert three rows

```
INSERT INTO InvoiceCopy
VALUES
    (95, '111-10098', '2016-04-01', 219.50, 0, 0, 1,
    '2016-04-30', NULL),
    (102, '109596', '2016-04-01', 22.97, 0, 0, 1,
    '2016-04-30', NULL),
    (72, '40319', '2016-04-01', 173.38, 0, 0, 1,
    '2016-04-30', NULL);
```

The response from the system

```
(3 row(s) affected)
```

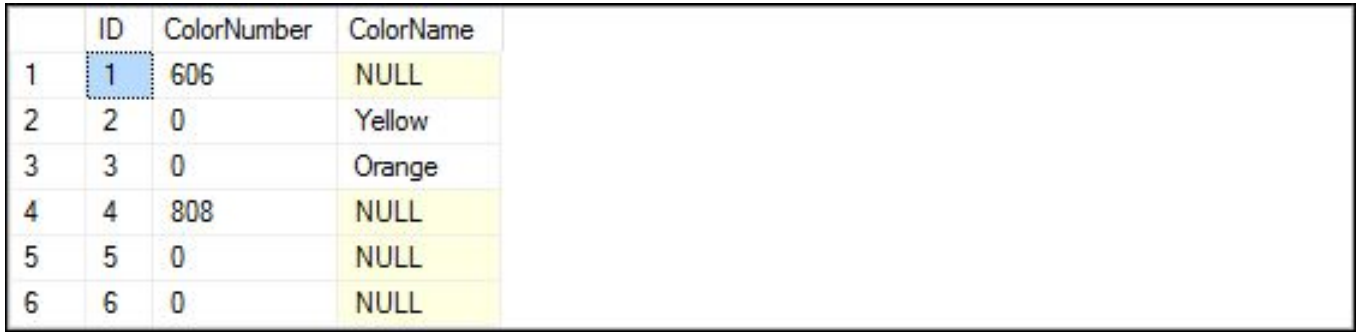
The definition of the ColorSample table

Column Name	Data Type	Length	Identity	Allow Nulls	Default Value
ID	Int	4	Yes	No	No
ColorNumber	Int	4	No	No	0
ColorName	VarChar	10	No	Yes	No

Six INSERT statements for the ColorSample table

```
INSERT INTO ColorSample (ColorNumber)
VALUES (606);

INSERT INTO ColorSample (ColorName)
VALUES ('Yellow');

INSERT INTO ColorSample
VALUES (DEFAULT, 'Orange');

INSERT INTO ColorSample
VALUES (808, NULL);

INSERT INTO ColorSample
VALUES (DEFAULT, NULL);

INSERT INTO ColorSample
DEFAULT VALUES;
```

The ColorSample table after the rows are inserted



	ID	ColorNumber	ColorName
1	1	606	NULL
2	2	0	Yellow
3	3	0	Orange
4	4	808	NULL
5	5	0	NULL
6	6	0	NULL

The syntax of the INSERT statement for inserting rows selected from another table

```
INSERT [INTO] table_name [(column_list)]
SELECT column_list
FROM table_source
[WHERE search_condition]
```

Insert paid invoices into the InvoiceArchive table

```
INSERT INTO InvoiceArchive
SELECT *
FROM InvoiceCopy
WHERE InvoiceTotal - PaymentTotal - CreditTotal = 0;
```

```
(103 row(s) affected)
```

The same INSERT statement with a column list

```
INSERT INTO InvoiceArchive
    (InvoiceID, VendorID, InvoiceNumber, InvoiceTotal,
    CreditTotal, PaymentTotal, TermsID, InvoiceDate,
    InvoiceDueDate)
SELECT
    InvoiceID, VendorID, InvoiceNumber, InvoiceTotal,
    CreditTotal, PaymentTotal, TermsID, InvoiceDate,
    InvoiceDueDate
FROM InvoiceCopy
WHERE InvoiceTotal - PaymentTotal - CreditTotal = 0;
```

```
(103 row(s) affected)
```

The syntax of the UPDATE statement

```
UPDATE table_name
SET column_name_1 = expression_1 [, column_name_2 =
    expression_2]...
[FROM table_source [[AS] table_alias]
[WHERE search_condition]
```

Update two columns of a single row

```
UPDATE InvoiceCopy
SET PaymentDate = '2016-05-21',
    PaymentTotal = 19351.18
WHERE InvoiceNumber = '97/522';
```

```
(1 row(s) affected)
```

Update one column of multiple rows

```
UPDATE InvoiceCopy
SET TermsID = 1
WHERE VendorID = 95;
```

```
(6 row(s) affected)
```

Update a column using an arithmetic expression

```
UPDATE InvoiceCopy
SET CreditTotal = CreditTotal + 100
WHERE InvoiceNumber = '97/522';
```

```
(1 row(s) affected)
```

Warning

- If you omit the WHERE clause, all the rows in the table will be updated.

The syntax of the DELETE statement

```
DELETE [FROM] table_name
[FROM table_source]
[WHERE search_condition]
```

Delete a single row from the InvoiceCopy table

```
DELETE InvoiceCopy
WHERE InvoiceID = 115;
```

```
(1 row(s) affected)
```

Delete all the invoices for a vendor

```
DELETE InvoiceCopy
WHERE VendorID = 37;
```

```
(3 row(s) affected)
```

Delete all paid invoices

```
DELETE InvoiceCopy
WHERE InvoiceTotal - PaymentTotal - CreditTotal = 0;
```

```
(103 row(s) affected)
```

Delete all the rows

```
DELETE InvoiceCopy;
```

```
(114 row(s) affected)
```

Warning

- If you omit the WHERE clause from a DELETE statement, all the rows in the table will be deleted.