Data Manipulation Language (DML),

Author: TrungDVQ (Fsoft-Academy)

Lesson Objectives

01

INSERT Statement

- ✓ Syntax
- ✓ INSERT INTO SELECT
- ✓ Best practice

02

UPDATE Statement

- ✓ Syntax

03

DELETE Statement

- ✓ Syntax

DML

INSERT	UPDATE	DELETE
Add Data	**Modify Data**	**Remove Data**

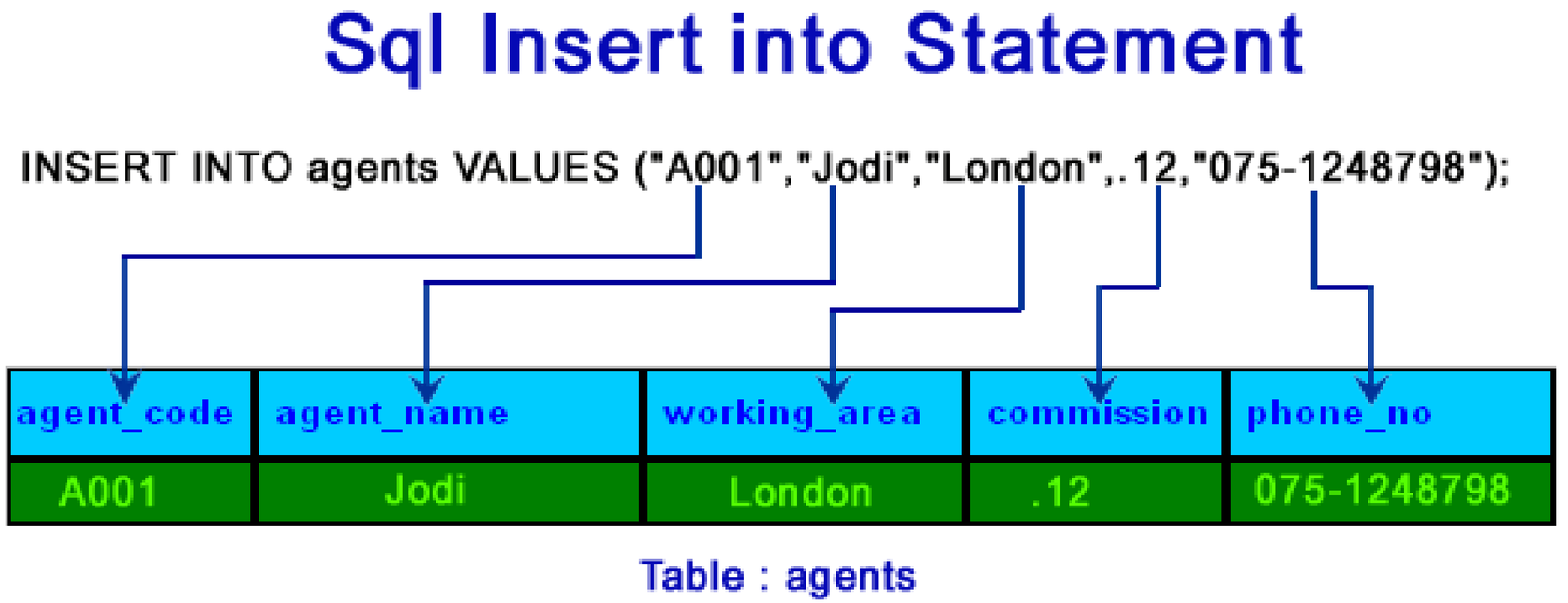
FPT Software | FRESHER ACADEMY

1

INSERT STATEMENT

INSERT

The INSERT INTO statement is used to adds one or more rows to a table or a view



INSERT syntax

```
INSERT INTO table_name
( column1,
  column2,
  column3 ,
  ... ,
  columnN
 )
VALUES
( value1 ,
  value2,
  value3,
  ... ,
  valueN
);
```

- Only one table allowed
- List of columns in table
- *Dismiss– insert data to all columns*
- Values and columns must be equal
- List value use to insert
- Repeat it to insert multi row

INSERT

	Column Name	Data Type	Allow Nulls
🔑	person_id	int	☐
	first_name	nvarchar(100)	☐
	last_name	nvarchar(100)	☐
	dob	date	☑
	full_name		☐

INSERT

```
INSERT INTO [dbo].[persons]
  VALUES
    ('Trung','Dinh','02/09/1985')
GO
```

```
INSERT INTO [dbo].[persons]
    ([first_name]
    ,[last_name]
    ,[dob])
VALUES
    ('Trung','Dinh', '02/09/1985'  ),
    (‘Nhung',‘Duong', '14/12/1985' )
GO
```

```
INSERT INTO [dbo].[persons]
    ([first_name]
    ,[last_name]
    ,[dob])
VALUES
    ('Trung','Dinh','02/09/1985')
GO
```

```
INSERT INTO [dbo].[persons]
    ([first_name]
    ,[last_name]
    ,[dob]
    ,[full_name])
VALUES
    (‘Nhung',‘Duong',
     '02/09/1985', ‘Nhung Duong' )
```

INSERT

SQL Server automatically uses the ***following value*** for the column that is ***available in the table*** but ***does not appear in the column list*** of the INSERT statement:

- The next incremental value if the column has an IDENTITY property.
- The default value if the column has a default value specified.
- The current timestamp value if the data type of the column is a timestamp data type.
- The NULL if the column is nullable.
- The calculated value if the column is a computed column.

INSERT INTO SELECT

The INSERT INTO SELECT statement copies data from one table and inserts it into another table.

- INSERT INTO SELECT requires that **data types** in **source** and **target** tables **match**
- The existing records in the target table are unaffected

```
INSERT INTO table2
SELECT *
FROM table1
WHERE condition;
```

```
INSERT INTO table2 (column1, column2, column3, ...)
SELECT column1, column2, column3, ...
FROM table1
WHERE condition;
```

INSERT INTO SELECT

The INSERT INTO SELECT statement copies data from one table and inserts it into another table.

- INSERT INTO SELECT requires that **data types** in **source** and **target** tables **match**
- The existing records in the target table are unaffected

```
INSERT INTO table2
SELECT *
FROM table1
WHERE condition;
```

```
INSERT INTO table2 (column1, column2, column3, ...)
SELECT column1, column2, column3, ...
FROM table1
WHERE condition;
```

INSERT INTO SELECT

```
CREATE TABLE Customer
(
    PersonID int PRIMARY KEY IDENTITY,,
    LastName nvarchar(100),
    FirstName nvarchar(100),
    PostalCode nvarchar(100),
    Country nvarchar(100)
);
```

Column Name	Data Type	Allow Nulls
PersonID	int	☐
LastName	nvarchar(100)	☑
FirstName	nvarchar(100)	☑
PostalCode	nvarchar(100)	☑
Country	nvarchar(100)	☑

```
CREATE TABLE Person
(
    person_id int PRIMARY KEY IDENTITY,
    first_name  nvarchar(100) NOT NULL,
    last_name  nvarchar(100) NOT NULL,
    full_name AS (first_name + ' ' + last_name),
    dob date
);
```

persons

Column Name	Data Type	Allow N...
person_id	int	☐
first_name	nvarchar(100)	☐
last_name	nvarchar(100)	☐
dob	date	☑
full_name		☐

INSERT INTO SELECT

persons

	Column Name	Data Type	Allow N...
🔑	person_id	int	☐
	first_name	nvarchar(100)	☐
	last_name	nvarchar(100)	☐
	dob	date	☑
	full_name		☐
			☐

Copy data from table **person** to table **customer**

customer *

	Column Name	Data Type	Allow Nulls
🔑	PersonID	int	☐
	LastName	nvarchar(100)	☑
	FirstName	nvarchar(100)	☑
	Address	nvarchar(255)	☑
	City	nvarchar(255)	☑
			☐

```
INSERT INTO [dbo].[customer]
           ([LastName],[FirstName])

SELECT [last_name],[first_name]

FROM [dbo].[persons]

SELECT @@rowcount as 'ROWCOUNT'
```

Tip 1: INSERT and return inserted values

```
INSERT INTO [dbo].[persons]
    ([first_name]
    ,[last_name]
    ,[dob]
    )

OUTPUT
inserted.[first_name],
inserted.[last_name],
inserted.[dob]

VALUES
   ('Trung','Dinh', '02/09/1985'  ),
   (‘Nhung',‘Duong', '12/14/1985' )

select @@ROWCOUNT AS ROW_COUNT
```

Results | Messages

	first_name	last_name	dob
1	Trung	Dinh	1985-02-09
2	Nhung	Duong	1983-12-14

	ROW_COUNT
1	2

Tip 1: INSERT and convert date

```
INSERT INTO [dbo].[persons]
    ([first_name]
    ,[last_name]
    ,[dob]
    )

OUTPUT
inserted.[first_name],
inserted.[last_name],
inserted.[dob]

VALUES
   ('Trung','Dinh', '02/09/1985'   ),
   (‘Nhung',‘Duong', '12/14/1985'  )

select @@ROWCOUNT AS ROW_COUNT
```

many case, It will be error

Tip 2: INSERT and convert date

```
INSERT INTO [dbo].[persons]
    ([first_name]
    ,[last_name]
    ,[dob]
    )

VALUES

('Trung','Dinh', '02/09/1985'  ),
(‘Nhung',‘Duong', '12/14/1985' )
```

```
INSERT INTO [dbo].[persons]
    ([first_name]
    ,[last_name]
    ,[dob]
    )

VALUES

('Trung','Dinh',
            CONVERT (date, '02/09/1985', 103 )),
( 'Nhung','Duong',
            CONVERT (date, '12/14/1985', 101 ))
```

Best Practice

- Use **CONVERT()** function to convert string to date type with format

CONVERT() syntax

CONVERT (data_type(length), expression, style)

Value	Description
data_type	Required. The datatype to convert expression to.
(length)	Optional. The length of the resulting data type (for char, varchar, nchar, nvarchar, binary and varbinary)
expression	Required. The value to convert to another data type
style	Optional. The format used to convert between data types, such as a date or string format.

https://www.w3schools.com/sql/func_sqlserver_convert.asp

CONVERT() syntax

Without century	With century	Input/Output	Standard
0	100	mon dd yyyy hh:miAM/PM	Default
1	101	mm/dd/yyyy	US
2	102	yyyy.mm.dd	ANSI
3	103	dd/mm/yyyy	British/French
4	104	dd.mm.yyyy	German
5	105	dd-mm-yyyy	Italian
10	110	mm-dd-yyyy	USA
11	111	yyyy/mm/dd	Japan

Tip 3: Set DATEFORMAT

SET DATEFORMAT { format | @format_var }

Valid parameters are **mdy**, **dmy**, **ymd**, **ydm**, **myd**, and **dym**

```
-- Set date format to day/month/year.
SET DATEFORMAT dmy;
GO
DECLARE @datevar DATETIME2 = '31/12/2021 09:01:01.1234567';
SELECT @datevar;
```

Result: 2008-12-31 09:01:01.123

Note: The DATEFORMAT **ydm** isn't supported for **date**, **datetime2**, and **datetimeoffset** data types.

2

UPDATE STATMENT

UPDATE statement

The UPDATE statement is used to changes existing data in a table or view

SQLQuery1.sql -

```
1  SELECT TOP 5 * FROM Sales.CurrencyRate
2  GO
3
4  UPDATE Sales.CurrencyRate
5  SET AverageRate = AverageRate + 0.01,
6      EndOfDayRate = EndOfDayRate + 0.01
7  GO
8
9  SELECT TOP 5 * FROM Sales.CurrencyRate
10
```

Results | Messages

Before Update

	CurrencyRateID	CurrencyRateDate	FromCurrencyCode	ToCurrencyCode	AverageRate	EndOfDayRate
1	1	2001-07-01 00:00:00.000	USD	ARS	1.00	1.0002
2	2	2001-07-01 00:00:00.000	USD	AUD	1.5491	1.55
3	3	2001-07-01 00:00:00.000	USD	BRL	1.9379	1.9419
4	4	2001-07-01 00:00:00.000	USD	CAD	1.4641	1.4683
5	5	2001-07-01 00:00:00.000	USD	CNY	8.2781	8.2784

After Update

	CurrencyRateID	CurrencyRateDate	FromCurrencyCode	ToCurrencyCode	AverageRate	EndOfDayRate
1	1	2001-07-01 00:00:00.000	USD	ARS	1.01	1.0102
2	2	2001-07-01 00:00:00.000	USD	AUD	1.5591	1.56
3	3	2001-07-01 00:00:00.000	USD	BRL	1.9479	1.9519
4	4	2001-07-01 00:00:00.000	USD	CAD	1.4741	1.4783
5	5	2001-07-01 00:00:00.000	USD	CNY	8.2881	8.2884

UPDATE Syntax

Syntax:

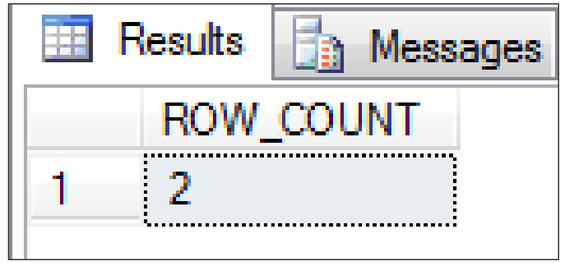
```
UPDATE table_name
SET    column1=value1,
       column2=value2,...
WHERE condition;
```

- **Notice the WHERE clause in the SQL UPDATE statement!**

The WHERE clause specifies which record or records that should be updated. If you **omit** the WHERE clause, **all records will be updated**!

Ex:

```
USE Fsoft_Training
UPDATE dbo.Customer
SET PostalCode = '4006'
WHERE Country = 'Norway'
SELECT @@ROWCOUNT AS ROW_COUNT
```



3

DELETE SATEMENT

DELETE Statement

Removes one or more rows from a table or view

CustomerId	CustomerName	ContactName
1	Alfreds Futterkiste	Maria Anders
2	Around the Horn	Thomas Hardy
3	Berglunds snabbköp	Christina Berglund
4	Antonio Moreno	Antonio Moreno
5	Ana Trujillo	Ana Trujillo

Best Practice:

To delete all the rows in a table, use **TRUNCATE TABLE**. **TRUNCATE TABLE** is faster than **DELETE** and uses fewer system and transaction log resources.

TRUNCATE TABLE drops and recreates the table instead of deleting rows one by one.

DELETE Syntax

- **Syntax:**

```
DELETE FROM table_name
WHERE some_column=some_value;
```

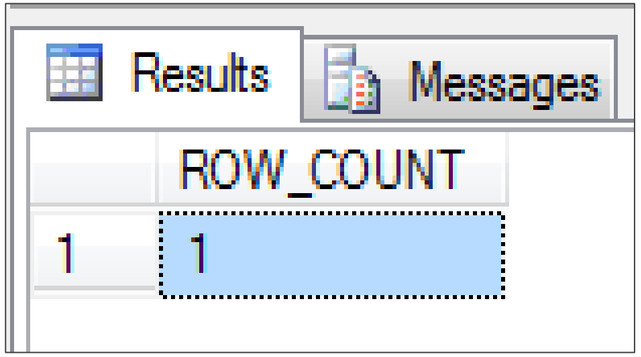
- **Notice the WHERE clause in the SQL DELETE statement!**

The WHERE clause specifies which record or records that should be deleted. If you omit the WHERE clause, all records will be deleted!

Please note that the DELETE FROM command cannot delete any rows of data that would violate FOREIGN KEY or other constraints.

- **Ex:**

```
USE Fsoft_Training
DELETE dbo.Customer
WHERE Country = 'Germany'
SELECT @@ROWCOUNT AS ROW_COUNT
```



Thank you!

Any questions?