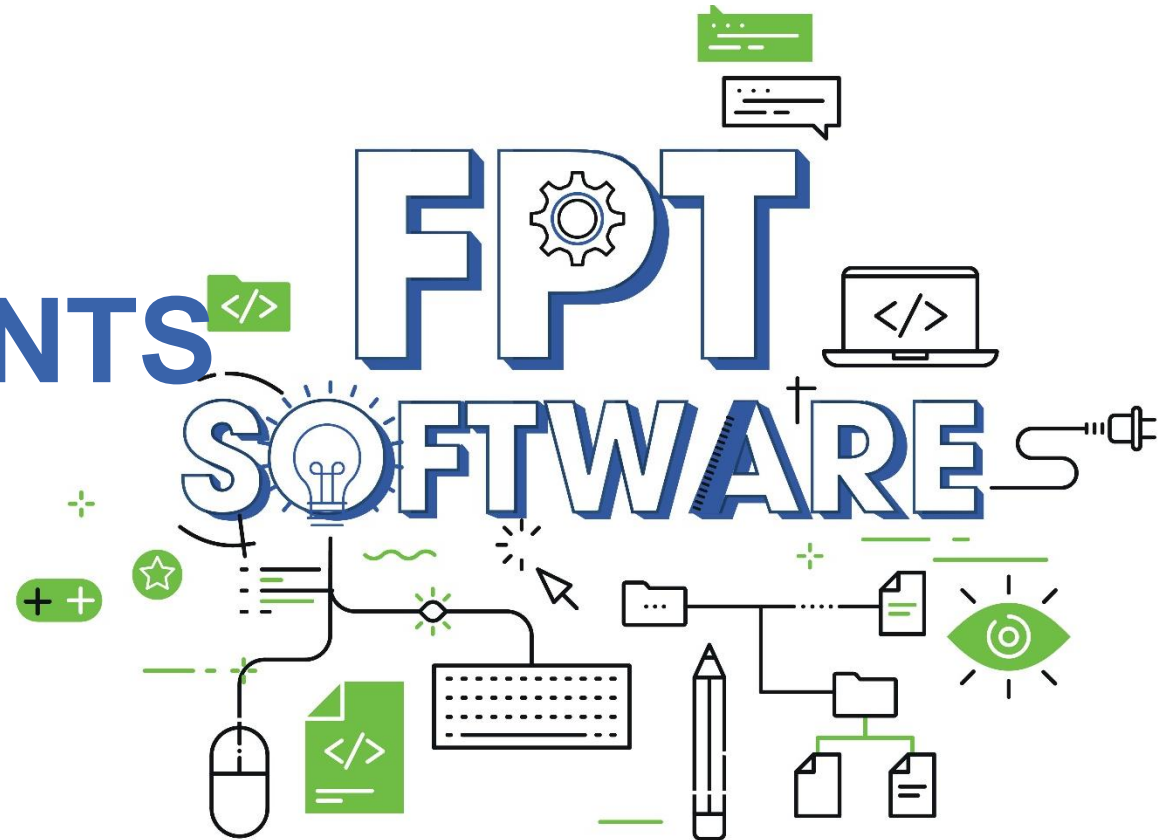


# BASIC DML STATEMENTS

*Fsoft Academy*



# Lesson Objectives

01

Describe each data manipulation language (DML) statement

02

Insert rows into a table

03

Update rows in a table

04

Delete rows from a table

05

Able to use basic SQL Operators



# Agenda

## 1. DML Statements

## 2. SQL Operators



## Section 1

# DML Statements

# DML Statement

- ✓ Insert Statement
- ✓ Update Statement
- ✓ Delete Statement
- ✓ Select Statement

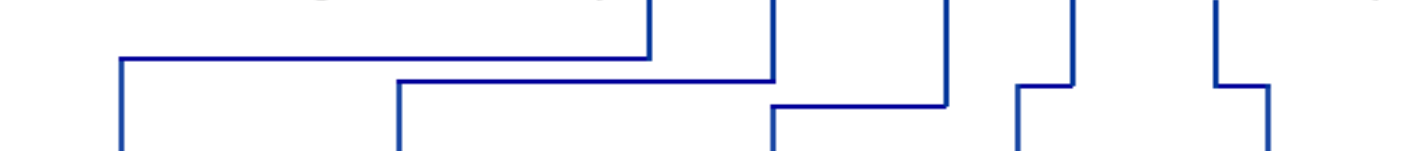


# INSERT Statements

- The **INSERT INTO** statement is used to add one or more rows to a table or a view

## Sql Insert into Statement

```
INSERT INTO agents VALUES ("A001","Jodi","London",.12,"075-1248798");
```



agent_code	agent_name	working_area	commission	phone_no
A001	Jodi	London	.12	075-1248798

Table : agents

# INSERT Statements

## ▪ Syntax:

### (1) Inserting data to all columns

```
INSERT INTO table_name  
VALUES (value1,value2,value3,...);
```

#### ✓ Ex1:

```
USE Fsoft_Training -- Database name
```

```
INSERT INTO dbo.Persons  
VALUES ( 1,'Tom', 'B. Erichsen','Skagen 21','Stavanger')
```

### (2) Inserting data to selected columns

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

#### ✓ Ex2:

```
USE Fsoft_Training  
INSERT INTO dbo.Customer (CustomerName, City, Country)  
VALUES ('Cardinal', 'Stavanger', 'Norway');
```

# INSERT Statement (3/3)

## ■ Demo

- ✓ Inserting data to selected columns
- ✓ Inserting data to all columns with identity column
- ✓ Insert many rows at one time





# UPDATE Statement

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

The screenshot shows a SQL query window with the following code:

```
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
```

Below the query window, the 'Results' tab displays two tables. The first table, titled 'Before Update', shows the state of the Sales.CurrencyRate table before the update. The second table, titled 'After Update', shows the state after the update. The 'AverageRate' and 'EndOfDayRate' columns are highlighted with red boxes in both tables, showing an increase of 0.01 for each row.

	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

	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

- **Best Practice**

- ✓ Use the **@@ROWCOUNT** function to return the number of inserted rows to the client application.

# UPDATE Statement

## ■ Syntax:

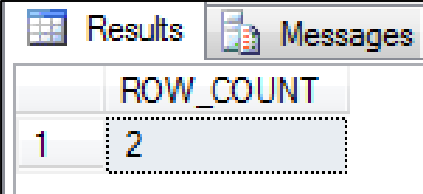
```
UPDATE table_name  
SET column1=value1,column2=value2,...  
WHERE some_column=some_value;
```

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!

## ■ Code:

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



Results	
	ROW_COUNT
1	2

# DELETE Statement

- Removes one or more rows from a table or view:

CustomerId	CustomerName	ContactName
<del>1</del>	<del>Alfreds Futterkiste</del>	<del>Maria Anders</del>
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 has restrictions, for example, the table cannot participate in replication

nếu 1 table mà có bản sao thì k thể truncate dc table đó

# DELETE Statement

- **Syntax:**

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

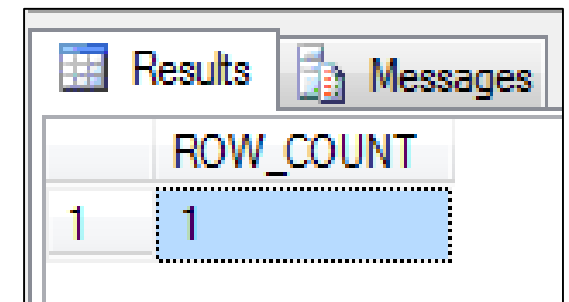
- **Notice the WHERE clause in the SQL DELETE statement!**

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
```



The screenshot shows a SQL Server interface with two tabs: 'Results' and 'Messages'. The 'Results' tab is active and displays a table with one row. The first column is labeled 'ROW\_COUNT' and contains the value '1'.

ROW_COUNT
1

# SELECT Statement

- Retrieves rows from the database and enables the selection of one or many rows or columns from one or many tables



# SELECT Statement

## ■ Syntax:

```
SELECT [ALL/DISTINCT/TOP [ WITH TIES ]] <Column name1>, <Column name2>,...  
FROM <Table name>  
    [WHERE <Search condition>]  
    [GROUP BY grouping columns]  
    [HAVING search condition]  
    [ORDER BY sort specification]
```

- ✓ **Ex1:** USE AdventureWorks  
GO  
SELECT ProductID, Name FROM Production.Product  
ORDER BY Name ASC;  
(504 rows)
- ✓ **Ex2:**  
SELECT DISTINCT E.Title FROM HumanResources.Employee E  
ORDER BY E.Title;  
(67 rows)

Results		Messages	
ProductID	Name		
1	Adjuster		
2	879		
3	712		
4	3		
5	2		
6	877		
7	316		
8	843		
9	952		
10	324		
11	322		
12	320		
13	321		
14	866		
15	865		
16	864		
17	505		


Results		Messages	
Title			
1	Accountant		
2	Accounts Manager		
3	Accounts Payable Specialist		
4	Accounts Receivable Specialist		
5	Application Specialist		
6	Assistant to the Chief Financial Officer		
7	Benefits Specialist		
8	Buyer		
9	Chief Executive Officer		
10	Chief Financial Officer		
11	Control Specialist		
12	Database Administrator		
13	Design Engineer		
14	Document Control Assistant		
15	Document Control Manager		
16	Engineering Manager		
17	European Sales Manager		

✓ Query executed successfully. ✓ Query executed successfully.

# SELECT Statement

- The **SELECT INTO** statement selects data from one table and inserts it into a different table.
- **Syntax:**

```
SELECT *  
INTO new_table_name  
FROM old_tablename
```



**Tips:** The SELECT INTO statement can also be used to **create a new**, empty table using the schema of another. **Just add a WHERE clause that causes the query to return no data:**

```
SELECT *  
INTO newtable  
FROM table1 WHERE 1=0;
```

# SELECT Statement

- **SQL Alias syntax:**

- ✓ *For table*

- ```
SELECT column_name(s)
FROM table_name AS alias_name
```

- ✓ *For Column(s)*

- ```
SELECT column_name AS alias_name
FROM table_name
```

- **Ex:**

```
USE AdventureWorks
GO
SELECT c.CustomerID, s.Name
FROM Sales.Customer AS c JOIN Sales.Store AS s
ON c.CustomerID = s.SalesPersonID
```



## Section 2

# SQL Operators

# What is an Operator in SQL?

- An **operator** is a **symbol specifying** an *action* that is *performed* on one or more expressions.
- Operators are used to specify conditions in an SQL statement and to serve as conjunctions for multiple conditions in a statement.
- In this section, we learn some types of following most operators:
  - 1 Arithmetic operators
  - 2 Comparison operators
  - 3 Logical operators.
  - 4 Compound Operators (Transact-SQL)
  - 5 String operators (Transact-SQL)

<https://learn.microsoft.com/en-us/sql/t-sql/language-elements/operators-transact-sql?view=sql-server-ver16>

# SQL Arithmetic Operators

- Here is a list of the Arithmetic operators available in SQL

Operator	Description	Example
+	Addition	$a + b \rightarrow 30$
-	Subtraction	$a - b \rightarrow -10$
*	Multiplication	$a * b \rightarrow 200$
/	Division	$b / a \rightarrow 2$
%	Modulus	$b \% a \rightarrow 0$

( Assume variable **a** holds **10** and variable **b** holds **20**)

# SQL Comparison Operators

- Here is a list of all the Comparison operators available in SQL

Operator	Description	Operator	Description
=	equal to	>=	greater than or equal to
!=, <>	not equal to	<=	less than or equal to
<	less than	!<	not less than
>	greater than	!>	not greater than

## Example

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

CUSTOMERS TABLE

**SQL:** *SELECT \* FROM CUSTOMERS WHERE SALARY > 5000;*



ID	NAME	AGE	ADDRESS	SALARY
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
7	Muffy	24	Indore	10000.00

# SQL Logical Operators

Operator	Description
ALL	✓ Used to compare a value to all values in another value set.
AND	✓ Used when both conditions are included
ANY	✓ Used to compare a value to any applicable value in the list according to the condition
BETWEEN	✓ Used to limit the values in a range e.g.
EXISTS	✓ Used to search for the presence of a row in a specified table that meets certain criteria
IN	✓ Included in the list e.g.
LIKE	✓ Equal to some character (use quotes)
NOT	✓ Opposite of the logical value
OR	✓ Used when either of the condition is true
IS NULL	✓ This checks if the field has a null
UNIQUE	✓ Searches every row of a specified table for uniqueness

# Compound Operators (Transact-SQL)

- Compound operators execute some operation and set an original value to the result of the operation.
- Transact-SQL provides the following compound operators:**

Operator	Action
+=	Adds some amount to the original value and sets the original value to the result.
-=	Subtracts some amount from the original value and sets the original value to the result.
*=	Multiplies by an amount and sets the original value to the result.
/=	Divides by an amount and sets the original value to the result.
%=	Divides by an amount and sets the original value to the modulo.
&=	Performs a bitwise AND and sets the original value to the result.
^=	Performs a bitwise exclusive OR and sets the original value to the result.
=	Performs a bitwise OR and sets the original value to the result.

# String Operators (Transact-SQL)

- **Wildcard string operators** can match one or more characters in a string comparison operation. LIKE and PATINDEX are examples of two of these operations.
- SQL Server provides the following string operators.
  - **=** (String comparison or assignment)
  - **+** (String concatenation)
  - **+=** (String concatenation assignment)
  - **%** (Wildcard - character(s) to match)
  - **[ ]** (Wildcard - character(s) to match)
  - **[^]** (Wildcard - character(s) not to match)
  - **\_** (Wildcard - match one character)

# String Operators (Transact-SQL)

## ■ Examples:

- Returns the **first and last names of people** in the Person.Person table of AdventureWorks2022, where the first name starts with Dan.

```
SELECT FirstName, LastName  
FROM Person.Person WHERE FirstName LIKE 'Dan%';
```

- Find the **top 5 people** in the Contact table who have a first name that starts with Al and has a third letter that is not the letter a.

```
SELECT TOP 5 FirstName, LastName  
FROM Person.Person WHERE FirstName LIKE 'Al[^a]%';
```



- **DML Statements:** INSERT, UPDATE, DELETE, SELECT
- **SQL Operators:**
  - ✓ Arithmetic operators
  - ✓ Comparison operators
  - ✓ Compound operators
  - ✓ String Concatenation operator
  - ✓ Logical operators

# THANK YOU!

