Contents

[1. Insert demo 1](#_Toc372746825)

[Insert data to selected columns 1](#_Toc372746826)

[Inserting data to all columns with identity column 2](#_Toc372746827)

[Insert many rows at one time 2](#_Toc372746828)

[Using INSERT ... SELECT 2](#_Toc372746829)

[Using SELECT... INTO 2](#_Toc372746830)

[2. Update demo 3](#_Toc372746831)

[This demo use data which is created from previous demo (*Insert many rows at one time*) 3](#_Toc372746832)

[3. Delete demo 3](#_Toc372746833)

[This demo use data which is created from previous demo (*Insert many rows at one time*) 3](#_Toc372746834)

[4. Select demo 3](#_Toc372746835)

[TOP 3](#_Toc372746836)

[DISTINCT 3](#_Toc372746837)

[Aggregate function 3](#_Toc372746838)

[SUM 3](#_Toc372746839)

[MAX 3](#_Toc372746840)

[MIN 4](#_Toc372746841)

[AVG 4](#_Toc372746842)

[COUNT 4](#_Toc372746843)

[5. SQL Operators 4](#_Toc372746844)

[String concatenation + 4](#_Toc372746845)

[Logical 4](#_Toc372746846)

# Insert demo

## Insert data to selected columns

We can specific columns that we would like to insert as below

INSERT dbo.Employee (FirstName, LastName, NationalIDNumber, ManagerID, Title, BirthDate, MaritalStatus, Gender)

VALUES ('Bill', 'Gates', '123456', NULL, 'CEO', '1959-01-01', 'M' , 'M')

## Inserting data to all columns with identity column

Sometimes, we have to insert a specific value (for instance 100) for identity column (for example, EmployeeID), for the case we use IDENTITY\_INSERT as below:

SET IDENTITY\_INSERT dbo.Employee ON

INSERT dbo.Employee (EmployeeID, FirstName, LastName, NationalIDNumber, ManagerID, Title, BirthDate, MaritalStatus, Gender)

VALUES (100, 'Steve', 'Jobs', '123456', NULL, 'CEO', '1963-01-01', 'M' , 'M')

SET IDENTITY\_INSERT dbo.Employee OFF

## Insert many rows at one time

### Using INSERT ... SELECT

In order to insert multiple rows into a table we can use INSERT...SELECT as below.

ALTER TABLE dbo.Employee

ADD Salary DECIMAL(19, 4)

INSERT dbo.Employee (FirstName, LastName, NationalIDNumber, ManagerID, Title, BirthDate, MaritalStatus, Gender, Salary)

SELECT 'Bill' + CAST(EmployeeID AS VARCHAR(10)) AS FirstName, 'Gates' + CAST(EmployeeID AS VARCHAR(10)) AS LastName,

NationalIDNumber, ManagerID, Title, BirthDate, ISNULL(MaritalStatus, 'S'), Gender, 1000 + EmployeeID \*10 Salary

FROM AdventureWorks.HumanResources.Employee

### Using SELECT... INTO

We can insert multiple rows into a table by using SELECT ... INTO. This statement will automatically create a new table in INTO phrase. For instance, below statement will create a new table Employee2.

SELECT

EmployeeID, 'Bill' + CAST(EmployeeID AS VARCHAR(10)) AS FirstName, 'Gates' + CAST(EmployeeID AS VARCHAR(10)) AS LastName,

NationalIDNumber, ManagerID, Title, BirthDate, MaritalStatus, Gender, 1000 + EmployeeID \* 10 Salary, 0 IsDeletedFlag

INTO dbo.Employee2

FROM AdventureWorks.HumanResources.Employee

# Update demo

## This demo use data which is created from previous demo (*Insert many rows at one time*)

For example, we are trying to raise for employees who has salary less than 1500.

UPDATE dbo.Employee

SET Salary = 1500

WHERE Salary < 1500

# Delete demo

## This demo use data which is created from previous demo (*Insert many rows at one time*)

For example, we will fire employees who has salary more than 3500 USD

DELETE dbo.Employee WHERE Salary > 3500

# Select demo

### TOP

Suppose we would like to select the 100 oldest employees.

SELECT TOP 100 \*

FROM dbo.Employee

ORDER BY BirthDate

### DISTINCT

For example, we want to see Title of employees. Due to some employees have the same Title we should use DISTINCT

SELECT DISTINCT Title

FROM dbo.Employee

## Aggregate function

### SUM

Suppose, we want to know how much the company need to pay for the employees, we will use SUM

SELECT SUM(Salary)

FROM dbo.Employee

### MAX

SELECT MAX(Salary)

FROM dbo.Employee

### MIN

SELECT MIN(Salary)

FROM dbo.Employee

### AVG

SELECT AVG(Salary)

FROM dbo.Employee

### COUNT

We want to know how many employees for each Title, we can use COUNT

SELECT Title, COUNT(\*)

FROM dbo.Employee

GROUP BY Title

or, we would like to know Title which has more than 10 employees. We use COUNT and HAVING

SELECT Title, COUNT(\*)

FROM dbo.Employee

GROUP BY Title

HAVING COUNT(\*) > 10

# SQL Operators

## String concatenation +

This example concate FirstName and LastName into FullName  
SELECT EmployeeID, FirstName + ' ' + LastName FullName

FROM dbo.Employee

## Logical

Below statement list all employees who has Title as 'Production Technician' (use LIKE)

SELECT \* FROM dbo.Employee

WHERE Title LIKE 'Production Technician%'

Below statement get all employees who are still single and female (use AND)

SELECT \* FROM dbo.Employee

WHERE MaritalStatus = 'S' AND Gender = 'F'

Below statement list employees who has salary in 1500 and 2000 USD

SELECT \* FROM dbo.Employee

WHERE Salary BETWEEN 1500 AND 2000