SQL for Citizen Data Scientists

SQL UPDATE Statement - Learn By Example

SETScholars & WACAMLDS

The UPDATE statement allows you to update existing rows with new values.

With it, you can change the value of one or more columns in each row that meets the search condition of the WHERE clause.

Syntax

The UPDATE statement is made up of three parts:

1. The table you want to update
2. The column names with their new values
3. The condition that determines which row to update

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

Sample Table

To help you better understand the examples, and enable you to follow along with the tutorial, we are going to use the following sample table.

This table is part of an 'Employee Management System' that contains basic information about employees.

ID	Name	Age	City	Job	Salary
1	Bob	28	New York	Manager	60000
2	Kim	25	Chicago	Manager	55000
3	Eve	24	New York	Developer	32000
4	Joe	23	Chicago	Developer	30000
5	Max	26	New York	Janitor	9000
6	Sam	27	Chicago	Janitor	10000

UPDATE Single Column

The simplest use of the UPDATE statement is to update a single column in a table.

Let's take a simple example. Suppose the fourth employee (ID = 4) moved from 'New York' to 'Newark', therefore that row needs updating. The following query will perform this update:

```
UPDATE Employees
SET City = 'Newark'
WHERE ID = 4;
```

ID	Name	Age	City	Job	Salary
1	Bob	28	New York	Manager	60000
2	Eve	24	New York	Developer	32000
3	Max	26	New York	Janitor	9000
4	Kim	25	Newark	Manager	55000
5	Joe	23	Chicago	Developer	30000
6	Sam	27	Chicago	Janitor	10000

The UPDATE statement always starts with the name of the table to be updated. In our case, it's the 'Employees' table. The SET clause is then used to assign the new value to the specified column. As the SET clause here, sets the 'City' column to the new value 'Newark'. The UPDATE statement ends with a WHERE clause that tells the DBMS which row to update. Here it's the fourth row.

Special care must be taken when using UPDATE, because if you accidentally omit the WHERE clause you will end up updating every row in the table.

```
UPDATE Employees
SET City = 'Newark';
```

ID	Name	Age	City	Job	Salary
1	Bob	28	Newark	Manager	60000
2	Eve	24	Newark	Developer	32000
3	Max	26	Newark	Janitor	9000
4	Kim	25	Newark	Manager	55000
5	Joe	23	Newark	Developer	30000
6	Sam	27	Newark	Janitor	10000

UPDATE Multiple Columns

You can also update multiple columns at once. When updating multiple columns, only a single SET keyword is used, and each column=value pair is separated by a comma.

In this example, columns 'City', 'Age' and 'Salary' will be updated for the fourth employee.

```
UPDATE Employees
SET City = 'Newark',
   Age = 26,
   Salary = 58000
WHERE ID = 4;
```

ID	Name	Age	City	Job	Salary
1	Bob	28	New York	Manager	60000
2	Eve	24	New York	Developer	32000
3	Max	26	New York	Janitor	9000
4	Kim	26	Newark	Manager	58000
5	Joe	23	Chicago	Developer	30000
6	Sam	27	Chicago	Janitor	10000

UPDATE Multiple Rows

With the UPDATE statement, you can update multiple rows at once. To update several rows, use a WHERE clause that selects only the rows you want to update.

For example, you might want to increase the salary of everyone over the age of 25 by 10%.

```
UPDATE Employees
SET Salary = Salary * 1.1
WHERE Age > 25;
```

ID	Name	Age	City	Job	Salary
1	Bob	28	New York	Manager	66000
2	Eve	24	New York	Developer	32000
3	Max	26	New York	Janitor	9900
4	Kim	25	Chicago	Manager	55000
5	Joe	23	Chicago	Developer	30000
6	Sam	27	Chicago	Janitor	11000

Note that the expression Salary * 1.1 in this example returns the salary increased by 10%.

UPDATE When Corresponding Rows Exist

Sometimes you want to update rows in one table when corresponding rows exist in another. For example, if an employee appears in table 'Employee_Bonus', you want to increase that employee's salary (in table Employees) by 10%.

Suppose the 'Employee_Bonus' table looks like this:

There are three ways to achieve this.

Method 1: Using IN operator

You can use a subquery in the WHERE clause of your UPDATE statement to find employees in the 'Employees' table that are also in the 'Employee_Bonus' table. Your UPDATE will then work only on those rows, so that you will be able to increase their salary by 10 percent.

```
UPDATE Employees
SET Salary = Salary * 1.1
WHERE ID IN (SELECT ID
        FROM Employee_Bonus);
```

ID	Name	Age	City	Job	Salary
1	Bob	28	New York	Manager	66000
2	Eve	24	New York	Developer	32000
3	Max	26	New York	Janitor	9000
4	Kim	25	Chicago	Manager	60500
5	Joe	23	Chicago	Developer	33000
6	Sam	27	Chicago	Janitor	10000

The results from the subquery represent the rows that will be updated in table 'Employees'. The IN operator tests the IDs from the 'Employees' table to see if they are in the list of IDs returned by the subquery. When they are, the corresponding 'Salary' values are updated.

Method 2: Using EXISTS clause

Alternatively, you can use EXISTS instead of IN:

```
UPDATE Employees
SET Salary = Salary * 1.1
WHERE EXISTS (SELECT NULL
        FROM Employee_Bonus
        WHERE Employees.ID = Employee_Bonus.ID );
```

ID	Name	Age	City	Job	Salary
1	Bob	28	New York	Manager	66000
2	Eve	24	New York	Developer	32000
3	Max	26	New York	Janitor	9000
4	Kim	25	Chicago	Manager	60500
5	Joe	23	Chicago	Developer	33000
6	Sam	27	Chicago	Janitor	10000

Method 3: Using FROM clause

Above example can also be solved using the FROM clause of the UPDATE statement. The FROM clause contains the names of tables that are involved in the UPDATE statement. All these tables must be subsequently joined.

```
UPDATE Employees
SET Salary = Salary * 1.1
FROM Employees, Employee_Bonus
WHERE Employees.ID = Employee_Bonus.ID;
```

ID	Name	Age	City	Job	Salary
1	Bob	28	New York	Manager	66000
2	Eve	24	New York	Developer	32000
3	Max	26	New York	Janitor	9000
4	Kim	25	Chicago	Manager	60500
5	Joe	23	Chicago	Developer	33000
6	Sam	27	Chicago	Janitor	10000

UPDATE with Values from Another Table

You wish to update rows in one table using values from another. For example, you have a table called 'New_Salary', which holds the new salaries for certain employees. The contents of the table 'New_Salary' are:

ID	Salary
1	66000
4	60500
5	33000

Now you want to update the salaries of certain employees in the 'Employees' table using values from the 'New_Salary' table, if there is a match. It is quite common for updates such as this one to be performed via Inner Join.

```
UPDATE Employees
SET Employees.Salary = New_Salary.Salary
FROM Employees
INNER JOIN New_Salary
ON Employees.ID = New_Salary.ID;
```

ID	Name	Age	City	Job	Salary
1	Bob	28	New York	Manager	66000
2	Eve	24	New York	Developer	32000
3	Max	26	New York	Janitor	9000
4	Kim	25	Chicago	Manager	60500
5	Joe	23	Chicago	Developer	33000
6	Sam	27	Chicago	Janitor	10000

Conditional UPDATE with CASE

Sometimes you need to update the column conditionally, at that time you can use a CASE expression.

For example, you want to increase each employee's salary by (20, 10, or 5) percent based on their previous salary. Those who earn less than $ 20000 will get a 20% increase, those who have a salary of more than $ 20000 but less than $ 50000 will get a 10% increase and others will get a 5% increase.

```
UPDATE Employees
SET Salary = CASE
        WHEN Salary > 0 AND Salary < 20000 THEN Salary * 1.2
        WHEN Salary >= 20000 AND Salary < 50000 THEN Salary * 1.1
        ELSE Salary * 1.05
        END
```

ID	Name	Age	City	Job	Salary
1	Bob	28	New York	Manager	63000
2	Eve	24	New York	Developer	35200
3	Max	26	New York	Janitor	10800
4	Kim	25	Chicago	Manager	57750
5	Joe	23	Chicago	Developer	33000
6	Sam	27	Chicago	Janitor	12000