UPDATE - SQL

This chapter covers the following topics:

- Function
- Syntax 1 - Searched UPDATE
- Syntax 2 - Positioned UPDATE
- Examples

For an explanation of the symbols used in the syntax diagram, see *Syntax Symbols*.

Belongs to Function Group: *Database Access and Update*

Function

The SQL `UPDATE` statement is used to perform an `UPDATE` operation on either rows in a table without using a cursor ("searched" `UPDATE`) or columns in a row to which a cursor is positioned ("positioned" `UPDATE`).

Two different structures are possible.

Syntax 1 - Searched UPDATE

The "Searched" `UPDATE` statement is a stand-alone statement not related to any `SELECT` statement. With a single statement you can update zero, one, multiple or all rows of a table. The rows to be updated are determined by a *`search-condition`* that is applied to the table. Optionally, view and table names can be assigned a *`correlation-name`*.

Note:
The number of rows that have actually been updated with a "searched" `UPDATE` can be ascertained by using the system variable `*ROWCOUNT`.

```
UPDATE                    { view-name  [correlation-name] SET *                      }
                          { table-name  [correlation-name] SET assignment-list       }
  [WHERE search-condition] [             WITH { RR   }  ]   [QUERY NO  integer]
                           [                  { RS   }  ]
                           [                  { CS   }  ]
```

For an explanation of the symbols used in the syntax diagram, see *Syntax Symbols*.

Syntax Element Description - Syntax 1:

Syntax Element	Description	
view-name	**View Name:** Refers to the name of a Natural view as defined in the DEFINE DATA statement. For further information, see *view-name* (in the section *Basic Syntactical Items*).	
correlation-name	**Correlation Name:** The item *correlation-name* represents an alias name for a *table-name*. For further information, see *correlation-name* (in the section *Basic Syntactical Items*).	
SET	**SET Clause:** If a view has been specified for updating, an asterisk (*) has to be specified in the SET clause, because all columns of the view must be updated. If a table has been specified for updating, the SET clause must contain either an *assignment-list* or the name of the view which contains the columns to be updated.	
assignment-list	**Assignment List:** See *Assignment List* below.	
WHERE *search-condition*	**WHERE Clause:** This clause is used to specify the selection criteria for the rows to be updated. If no WHERE clause is specified, the entire table is updated.	
WITH	**WITH Clause:** This clause allows the explicit specification of the isolation level used when locating the row to be updated. For detailed information, see *WITH Clause - Isolation Level* in the description of the SELECT statement. It is only valid against DB2 databases. When used against other databases, it will cause runtime errors.	
	CS	Cursor Stability
	RR	Repeatable Read
	RS	Read Stability
QUERYNO *integer*	**QUERYNO Clause:** This clause is not currently supported and will be ignored.	

Assignment List

```
{ column-name = { scalar-expression } } , ...
{               { NULL              } }
```

In an *assignment-list*, you can assign values to one or more columns. A value can be either a *scalar-expression* or NULL. For further information, see *Scalar Expressions*.

If the value NULL has been assigned, it means that the addressed field is to contain no value (not even the value "0" or "blank").

Syntax Element Description:

Syntax Element	Description
column-name	**Column Name:** Specifies the name of a column of the result table of the MERGE statement that is not the same name as another include column or a column in the target table.
NULL	**NULL Option:** Specifies the null value as the new value of the column. If the value NULL has been assigned, it means that the addressed field is to contain no value (not even the value 0 or "blank").

Syntax 2 - Positioned UPDATE

The "positioned" UPDATE statement always refers to a cursor within a database loop. Thus, the table or view referenced by a positioned UPDATE statement must be the same as the one referenced by the corresponding SELECT statement; otherwise an error message is returned. A positioned UPDATE cannot be used with a non-cursor selection.

Common Set Syntax:

```
UPDATE { view-name  SET *               } [WHERE CURRENT OF CURSOR (r)]
       { view-name  SET assignment-list }
```

Extended Set Syntax:

```
UPDATE { view-name  SET *               } [WHERE CURRENT OF CURSOR (r) [ FOR ROW { [:]host-variable } OF ROWSET ]
       { view-name  SET assignment-list }                               [         { integer         }           ]
```

Syntax Element Description - Syntax 2:

Syntax Element	Description
`view-name`	**Natural View:** Refers to the name of a Natural view as defined in the `DEFINE DATA` statement; see also `view-name` (in the section *Basic Syntactical Items*).

`SET *`
`SET assignment-list`
	SET Clause:
	If a Natural view has been specified for updating, an asterisk (*) has to be specified in the `SET` clause, because all columns of the view must be updated.
	If a table has been specified for updating, the `SET` clause must contain either an `assignment-list` or the name of the view which contains the columns to be updated.

`WHERE CURRENT OF CURSOR (r)`	**Statement Reference:** The `(r)` notation is used to reference the statement which was used to select the row to be updated. If no statement reference is specified, the `UPDATE` statement is related to the innermost active processing loop in which a database record was selected.
`FOR ROW … OF ROWSET`	**FOR ROW … OF ROWSET Clause:** This clause belongs to the SQL Extended Set. The optional `FOR ROW … OF ROWSET` clause for positioned SQL `UPDATE` statements specifies which row of the current rowset has to be updated. It should only be specified if the `UPDATE` statement is related to a `SELECT` statement which uses rowset positioning and which has column arrays in the *INTO Clause*. If this clause is omitted, all rows of the current rowset are updated by the values in the `assignment-list`. This clause cannot be specified if `view-name` `SET *` is specified.

Examples

- Example 1 - Searched UPDATE
- Example 2 - Searched UPDATE with assignment-list
- Example 3 - Positioned UPDATE

- Example 4 - Positioned UPDATE with assignment-list

Example 1 - Searched UPDATE

```
DEFINE DATA LOCAL
1 PERS VIEW OF SQL-PERSONNEL
2 NAME
2 AGE
...
END-DEFINE
...
ASSIGN AGE = 45
ASSIGN NAME = ’SCHMIDT’
UPDATE PERS SET * WHERE NAME = ’SCHMIDT’
...
```

Example 2 - Searched UPDATE with *assignment-list*

```
DEFINE DATA LOCAL
1 PERS VIEW OF SQL-PERSONNEL
2 NAME
2 AGE
...
END-DEFINE
...
UPDATE SQL-PERSONNEL SET AGE = AGE + 1 WHERE NAME = ’SCHMIDT’
...
```

Example 3 - Positioned UPDATE

```
DEFINE DATA LOCAL
1 PERS VIEW OF SQL-PERSONNEL
2 NAME
2 AGE
...
END-DEFINE
...
SELECT * INTO PERS FROM SQL_PERSONNEL WHERE NAME = ’SCHMIDT’
COMPUTE AGE = AGE + 1
UPDATE PERS SET * WHERE CURRENT OF CURSOR
END-SELECT
...
```

Example 4 - Positioned UPDATE with *assignment-list*

```
DEFINE DATA LOCAL
1 PERS VIEW OF SQL-PERSONNEL
2 NAME
2 AGE
...
END-DEFINE
...
SELECT * INTO PERS FROM SQL-PERSONNEL WHERE NAME = ’SCHMIDT’
UPDATE SQL-PERSONNEL SET AGE = AGE + 1 WHERE CURRENT OF CURSOR
END-SELECT
...
```