MULTIPLY

Dieses Kapitel behandelt folgende Themen:

- Funktion
- Syntax-Beschreibung
- Beispiel

Verwandte Statements: ADD | COMPRESS | COMPUTE | DIVIDE | EXAMINE | MOVE | MOVE ALL | RESET | SEPARATE | SUBTRACT

Gehört zur Funktionsgruppe: *Arithmetische Funktionen und Datenzuweisungen*

Funktion

Das Statement `MULTIPLY` dient dazu, zwei Operanden miteinander zu multiplizieren. Das Ergebnis der Multiplikation kann entweder in *operand1* oder in *operand3* ausgegeben werden.

Wird ein Datenbankfeld als Ergebnisfeld verwendet, so ändert sich durch die Multiplikation nur der programmintern benutzte Wert des Feldes. Der in der Datenbank gespeicherte Feldwert wird davon nicht beeinflusst.

Zu Multiplikationen mit Arrays siehe auch den Abschnitt *Arithmetische Operationen mit Arrays* im *Leitfaden zur Programmierung*.

Syntax-Beschreibung

Bei diesem Statement sind unterschiedliche Strukturen möglich.

- Syntax 1 — MULTIPLY ohne GIVING-Klausel
- Syntax 2 — MULTIPLY mit GIVING-Klausel

Syntax 1 — MULTIPLY ohne GIVING-Klausel

Wenn Syntax 1 benutzt wird, kann das Ergebnis der Multiplikation in *operand1* gespeichert werden.

`MULTIPLY` [**`ROUNDED`**] *operand1* **`BY`** *operand2*

Eine Erläuterung der in dem Syntax-Diagramm verwendeten Symbole entnehmen Sie dem Abschnitt *Syntax-Symbole*.

Operanden-Definitionstabelle (Syntax 1):

Operand	Mögliche Struktur					Mögliche Formate											Referenzierung erlaubt	Dynam. Definition
operand1		S	A		M		N	P	I	F							ja	nein
operand2	C	S	A		N		N	P	I	F							ja	nein

Syntax-Element-Beschreibung (Syntax 1):

operand1* BY *operand2	*operand1* ist der Multiplikand, *operand2* ist der Multiplikator. Da keine `GIVING`-Klausel benutzt wird, wird das Ergebnis in *operand1* gespeichert, folglich ist das Statement äquivalent zu: `<oper1> := <oper1> * <oper2>`
ROUNDED	Wenn Sie das Schlüsselwort `ROUNDED` angeben, wird der Wert auf- oder abgerundet, bevor er dann *operand1* zugewiesen wird. Informationen zum Runden, siehe *Regeln für arithmetische Operationen*, *Abschneiden und Runden von Feldwerten* im *Leitfaden zur Programmierung*.

Syntax 2 — MULTIPLY mit GIVING-Klausel

Wenn Syntax 2 benutzt wird, kann das Ergebnis der Multiplikation in *operand3* gespeichert werden.

`MULTIPLY` [**`ROUNDED`**] *operand1* **`BY`** *operand2* **`GIVING`** *operand3*

Eine Erläuterung der in dem Syntax-Diagramm verwendeten Symbole entnehmen Sie dem Abschnitt *Syntax-Symbole*.

Operanden-Definitionstabelle (Syntax 2):

Operand	Mögliche Struktur					Mögliche Formate													Referenzierung erlaubt	Dynam. Definition
operand1	C	S	A		M			N	P	I	F								ja	nein
operand2	C	S	A		N			N	P	I	F								ja	nein
operand3		S	A		M	A	U	N	P	I	F	B*		T					ja	ja

* Format B von *operand3* kann nur mit einer Länge von kleiner gleich 4 verwendet werden.

Syntax-Element-Beschreibung (Syntax 2):

operand1* BY *operand2* GIVING *operand3	*operand1* ist der Multiplikand, *operand2* ist der Multiplikator. Da die GIVING-Klausel benutzt wird, wird *operand1* nicht geändert, und das Ergebnis wird in *operand3* gespeichert, folglich ist das Statement äquivalent zu: `<oper3> := <oper1> * <oper2>` Ist *operand1* eine numerische Konstante, ist die GIVING-Klausel erforderlich.
ROUNDED	Wenn Sie das Schlüsselwort ROUNDED angeben, wird der Wert auf- oder abgerundet, bevor er *operand1* zugewiesen wird. Informationen zum Runden siehe Abschnitt *Regeln für arithmetische Operationen*, *Abschneiden und Runden von Feldwerten* im *Leitfaden zur Programmierung*.

Beispiel

```
** Example 'MULEX1': MULTIPLY
************************************************************************
DEFINE DATA LOCAL
1 #A      (N3) INIT <20>
1 #B      (N5)
1 #C      (N3.1)
1 #D      (N2)
1 #ARRAY1 (N5/1:4,1:4) INIT (2,*) <5>
1 #ARRAY2 (N5/1:4,1:4) INIT (4,*) <10>
END-DEFINE
*
MULTIPLY #A BY 3
WRITE NOTITLE 'MULTIPLY #A BY 3'            25X '=' #A
*
MULTIPLY #A BY 3 GIVING #B
WRITE 'MULTIPLY #A BY 3 GIVING #B'          15X '=' #B
*
MULTIPLY ROUNDED 3 BY 3.5 GIVING #C
WRITE 'MULTIPLY ROUNDED 3 BY 3.5 GIVING #C'  6X '=' #C
*
MULTIPLY 3 BY -4 GIVING #D
WRITE 'MULTIPLY  3 BY -4 GIVING #D'         14X '=' #D
*
MULTIPLY -3 BY -4 GIVING #D
WRITE 'MULTIPLY -3 BY -4 GIVING #D'         14X '=' #D
*
MULTIPLY  3 BY  0 GIVING #D
WRITE 'MULTIPLY  3 BY  0 GIVING #D'         14X '=' #D
*
WRITE / '=' #ARRAY1 (2,*) '=' #ARRAY2 (4,*)
MULTIPLY #ARRAY1 (2,*) BY #ARRAY2 (4,*)
WRITE / 'MULTIPLY #ARRAY1 (2,*) BY #ARRAY2 (4,*)'
      / '=' #ARRAY1 (2,*) '=' #ARRAY2 (4,*)
*
END
```

Ausgabe des Programms MULEX1:

```
MULTIPLY #A BY 3                        #A:   60
MULTIPLY #A BY 3 GIVING #B              #B:    180
MULTIPLY ROUNDED 3 BY 3.5 GIVING #C     #C:   10.5
MULTIPLY  3 BY -4 GIVING #D             #D: -12
MULTIPLY -3 BY -4 GIVING #D             #D:  12
MULTIPLY  3 BY  0 GIVING #D             #D:   0

#ARRAY1:      5      5      5      5 #ARRAY2:     10     10     10     10

MULTIPLY #ARRAY1 (2,*) BY #ARRAY2 (4,*)
#ARRAY1:     50     50     50     50 #ARRAY2:     10     10     10     10
```