RUN

RUN [**REPEAT**] *operand1* [*operand2* [(*parameter*)]] **...** 40

Dieses Kapitel behandelt folgende Themen:

- Funktion
- Syntax-Beschreibung
- Dynamische Sourcecode-Generierung und -Ausführung
- Beispiel

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

Gehört zur Funktionsgruppe: *Aufrufen von Programmen und Unterprogrammen*

Funktion

Das RUN-Statement dient dazu, ein Natural-Source-Programm aus der Natural-Systemdatei zu lesen und es dann auszuführen.

Für Natural Remote Procedure Call (RPC): Siehe *Notes on Natural Statements on the Server* (in der *Natural Remote Procedure Call (RPC)*-Dokumentation).

Syntax-Beschreibung

Operanden-Definitionstabelle:

Operand	Mögliche Struktur					Mögliche Formate													Referenzierung erlaubt	Dynam. Definition
operand1	C	S				A													ja	nein
operand2	C	S	A	G		A	U	N	P	I	F	B	D	T	L		G		ja	nein

Syntax-Element-Beschreibung:

REPEAT	Mit RUN REPEAT wird ein Programm vollständig ausgeführt, ohne dass der Benutzer zwischendurch auf etwaigen (durch INPUT-Statements ausgegebenen) Ausgabeschirmen durch eine Eingabe reagieren muss. Diese Option kann eingesetzt werden, wenn ein Programm mehrere Schirme mit Informationen ausgibt, bei denen es nicht erforderlich ist, dass der Benutzer auf jeden ausgegebenen Schirm reagiert.
operand1	**Programmname:** Der Name des auszuführenden Programms (*operand1*) kann als alphanumerische Konstante oder als Inhalt einer alphanumerischen Variablen angegeben werden. Wird eine Variable verwendet, so muss sie 8 Stellen lang sein. Das Programm kann entweder in der aktuellen Library oder in einer Steplib (Standard-Steplib ist SYSTEM) gespeichert sein. Wird es dort nicht gefunden, gibt Natural eine Fehlermeldung aus. Das ausgeführte Programm wird in den Arbeitsbereich des Programm-Editors gelesen und überschreibt dabei den vorherigen Inhalt des Arbeitsbereichs.
operand2	**Parameter:** Mit dem RUN-Statement können Parameter an das Programm, das ausgeführt werden soll, übergeben werden. Die Parameter können mit beliebigem Format definiert werden; sie werden automatisch in Formate umgesetzt, die zu den entsprechenden INPUT-Feldern passen. Alle angegebenen Parameter werden oben auf dem Natural-Stack abgelegt. Parameter können von einem INPUT-Statement gelesen werden. Das erste INPUT-Statement fügt alle Parameter in die im INPUT-Statement angegebenen Felder ein. Bei numerischen Feldern muss der Vorzeichen-Parameter SG auf ON gesetzt werden. Werden mehr Parameter übergeben als INPUT-Felder vorhanden sind, so werden überschüssige Parameter ignoriert. Die Anzahl der Parameter kann über die Systemvariable *DATA ermittelt werden. **Anmerkung:** Wenn *operand2* eine Zeitvariable (Format T) ist, wird nur die Zeitkomponente des Variableninhalts übergeben, aber nicht die Datumskomponente.
parameter	Wenn *operand2* eine Datumsvariable ist, können Sie den Session- Parameter DF als *parameter* für diese Variable angeben.

Dynamische Sourcecode-Generierung und -Ausführung

Das RUN-Statement kann dazu verwendet werden, ein Programm dynamisch zu kompilieren und auszuführen, dessen Source ganz oder teilweise dynamisch erstellt wird.

Dynamische Sourcecode-Generierung erfolgt, indem man Sourcetext in globalen Variablen unterbringt, und diese Variablen dadurch referenziert, dass man im Sourcecode als erstes Zeichen im Variablennamen ein Pluszeichen (+) jeweils durch ein Und-Zeichen (&) ersetzt.

Wird das Programm mit RUN aufgerufen, so wird der Inhalt der globalen Variablen als Sourcecode interpretiert. Eine globale Variable mit Index darf nicht in einem mit RUN ausgeführten Programm verwendet werden.

Eine globale Variable darf keinen Kommentar und kein INCLUDE-Statement enthalten.

Beispiel

Programm mit RUN-Statement:

```
** Example 'RUNEX1': RUN (with dynamic source program creation)
************************************************************************
DEFINE DATA
GLOBAL
  USING RUNEXGDA
LOCAL
1 #NAME (A20)
1 #CITY (A20)
END-DEFINE
*
INPUT 'Please specify the search values:' //
      'Name:' #NAME /
      'City:' #CITY
*
RESET +CRITERIA     /* defined in GDA 'RUNEXGDA'
*
IF #NAME = ' ' AND #CITY = ' '
  REINPUT 'Enter at least 1 value'
END-IF
*
IF #NAME NE ' '
  COMPRESS 'NAME' ' =''' #NAME '''' INTO +CRITERIA LEAVING NO
END-IF
IF #CITY NE ' '
  IF +CRITERIA NE ' '
    COMPRESS +CRITERIA 'AND' INTO +CRITERIA
  END-IF
  COMPRESS +CRITERIA ' CITY =''' #CITY '''' INTO +CRITERIA LEAVING NO
END-IF
*
RUN 'RUNEXFND'
*
END
```

Programm RUNEXFND, das per RUN-Statement ausgeführt wird:

```
** Example 'RUNEXFND': RUN (program executed with RUN in RUNEX1)
************************************************************************
DEFINE DATA
GLOBAL
  USING RUNEXGDA
LOCAL
1 EMPLOY-VIEW VIEW OF EMPLOYEES
  2 NAME
  2 CITY
END-DEFINE
*
* &CRITERIA filled with  "NAME = 'xxxxx' AND CITY = 'xxxx'"
*
FIND NUMBER EMPLOY-VIEW WITH &CRITERIA
```

```
     RETAIN AS 'EMP-SET'
DISPLAY *NUMBER
*
END
```

Global Data Area RUNEXGDA:

```
Global    RUNEXGDA  Library SYSEXSYN                     DBID    10 FNR    32
Command                                                                > +
I T L  Name                            F Length     Miscellaneous
All -- --------------------------------- - ---------- ------------------------>
     1 +CRITERIA                       A         80
```