WRITE WORK FILE

```
WRITE WORK [FILE] work-file-number [VARIABLE] operand1 ...
```

Dieses Kapitel behandelt folgende Themen:

- Funktion
- Syntax-Beschreibung
- Externe Darstellung der Felder
- Verarbeitung großer und dynamischer Variablen
- Beispiel

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

Verwandte Statements: `DEFINE WORK FILE` | `READ WORK FILE` | `CLOSE WORK FILE`

Gehört zur Funktionsgruppe: *Verarbeitung von Arbeitsdateien/PC-Dateien*

Funktion

Das Statement `WRITE WORK FILE` dient dazu, Datensätze auf eine physisch-sequentielle Arbeitsdatei (Work File) zu schreiben.

Dieses Statement kann nur im Batch-Betrieb verwendet werden..

Es ist möglich, in einem Programm oder einer Verarbeitungsschleife eine Arbeitsdatei zu erstellen und diese dann in einem anderen Programm oder einer anderen eigenständigen Verarbeitungsschleife mit einem `READ WORK FILE`-Statement zu lesen.

Anmerkung:
Bezüglich Unicode-Support siehe *Work Files and Print Files on Windows, UNIX and OpenVMS Platforms* in der *Unicode and Code Page Support*-Dokumentation.

Syntax-Beschreibung

Operanden-Definitionstabelle:

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

Anmerkung:
Bei den Arbeitsdateitypen ENTIRECONNECTION oder TRANSFER darf *operand1* nicht im Format C oder G sein.

Syntax-Element-Beschreibung:

work-file-number	**Arbeitsdateinummer:** Die für Natural definierte Nummer der Arbeitsdatei, die verwendet werden soll.
VARIABLE	**Variablen-Eintrag:** Es ist möglich, mittels verschiedener `WRITE WORK FILE`-Statements Datensätze mit verschiedenen Feldern auf dieselbe Arbeitsdatei zu schreiben. In diesem Fall müssen alle betreffenden `WRITE WORK FILE`-Statements das Schlüsselwort `VARIABLE` enthalten; die Datensätze werden dann mit variablem Format auf die externe Datei geschrieben. Wenn die Operanden-Liste eine dynamische Variable enthält (die je nach Ausführungsart des `WRITE WORK FILE`-Statements eine andere Größe annimmt), muss der `VARIABLE`-Eintrag in allen `WRITE WORK FILE`-Statements angegeben werden. **Variabler Indexbereich:** Wenn Sie ein Array auf eine Arbeitsdatei schreiben, können Sie für das Array einen variablen Indexbereich angeben. Zum Beispiel: `WRITE WORK FILE work-file-number VARIABLE #ARRAY (I:J)`
operand1	**Felder:** Als *operand1* geben Sie die Felder an, die auf die Arbeitsdatei geschrieben werden sollen. Dies können entweder Datenbankfelder, Benutzervariablen, Systemvariablen und/oder Felder sein, die mit einem `READ WORK FILE`-Statement von einer anderen Arbeitsdatei gelesen wurden. Bei Datenbank-Arrays kann durch eine einen Bereich umfassende Indexierung angegeben werden, welche Ausprägungen auf die Arbeitsdatei geschrieben werden sollen. Feldgruppen können durch Angabe des Gruppennamens referenziert werden; alle Felder einer Gruppe werden einzeln auf die Arbeitsdatei geschrieben.

Externe Darstellung der Felder

Mit einem `WRITE WORK FILE`-Statement auf eine Arbeitsdatei geschriebene Felder werden auf der externen Datei entsprechend ihrer internen Definition dargestellt. Die Feldwerte werden nicht verändert.

Bei Feldern der Formate A oder B entspricht die Anzahl der Bytes auf der externen Datei der programminternen Längendefinition. Die Feldwerte werden nicht verändert; ein Komma (Dezimalpunkt) wird nicht wiedergegeben.

Bei Feldern des Formats N ergibt sich die Anzahl der Bytes auf der externen Datei aus der Summe der Stellen vor und nach dem Komma. Das Komma (Dezimalpunkt) wird auf der externen Datei nicht wiedergegeben.

Bei Feldern des Formats P ergibt sich die Anzahl der Bytes auf der externen Datei aus der Summe der Stellen vor und nach dem Komma plus einer Stelle für das Vorzeichen, geteilt durch 2, wobei auf ganze Bytes aufgerundet wird.

Anmerkung:
Beim Schreiben von Feldern auf eine Arbeitsdatei erfolgt keine Umsetzung von Feldformaten.

Beispiele für Felddarstellung:

Felddefinition	**Ausgabelänge**
#FIELD1 (A10)	10 Bytes
#FIELD2 (B15)	15 Bytes
#FIELD3 (N1.3)	4 Bytes
#FIELD4 (N0.7)	7 Bytes
#FIELD5 (P1.2)	2 Bytes
#FIELD6 (P6.0)	4 Bytes

Anmerkung:
Wenn die Systemfunktionen AVER, NAVER, SUM oder TOTAL für numerische Felder (Format N oder P) auf eine Arbeitsdatei geschrieben werden, vergrößert sich intern die Länge dieser Felder um eine Stelle (z.B.: SUM eines Feldes vom Format P3 wird auf P4 verlängert). Dies ist beim Lesen der Arbeitsdatei zu berücksichtigen.

Verarbeitung großer und dynamischer Variablen

Arbeitsdateityp	Verarbeitung
ASCII ASCII-COMPRESSED SAG (binär)	Die Arbeitsdateitypen ASCII, ASCII-COMPRESSED und SAG (binär) können keine dynamischen Variablen verarbeiten und rufen einen Fehler hervor. Sie können jedoch große Variablen mit einer maximalen Feld-/Datensatzlänge von 32766 Bytes verarbeiten.
ENTIRECONNECTION	Der Arbeitsdateityp ENTIRECONNECTION kann keine dynamische Variablen verarbeiten. Er kann jedoch große Variablen mit einer maximalen Feld-/Datensatzlänge von 1073741824 Bytes verarbeiten.
PORTABLE UNFORMATTED	Große und dynamische Variablen können mit den beiden Arbeitsdateitypen PORTABLE und UNFORMATTED in Arbeitsdateien geschrieben oder aus Arbeitsdateien gelesen werden. Bei diesen Typen gibt es keine Größenbeschränkung für dynamische Variablen. Große Variablen dürfen jedoch eine maximale Feld-/Datensatzlänge von 32766 Bytes nicht überschreiten. Beim Arbeitsdateityp PORTABLE wird die Feldinformation in der Arbeitsdatei gespeichert. Während des `READ` wird die Größe einer dynamischen Variablen geändert, wenn die Feldgröße im Datensatz von der aktuellen Größe abweicht. Mit dem `WRITE WORK FILE`-Statement werden Felder mit ihrer Byte-Länge in die angegebene Datei geschrieben. Alle Datentypen (DYNAMIC oder nicht) werden gleich behandelt. Es werden keine strukturellen Informationen eingefügt. Natural verwendet einen Puffermechanismus. Daher sind die Daten erst nach einem `CLOSE WORK` vollständig geschrieben. Das ist dann besonders wichtig, wenn die Datei mit einem anderen Utility verarbeitet werden soll während Natural aktiv ist. Mit dem `READ WORK FILE`-Statement werden Felder mit fester Länge in ihrer ganzen Länge gelesen. Wenn das Ende der Datei erreicht wird, wird der Rest des aktuellen Feldes mit Leerzeichen aufgefüllt. Die nachfolgenden Felder werden nicht verändert. Bei DYNAMIC-Datentypen wird der ganze Rest der Datei gelesen, außer wenn sie länger als 1073741824 Bytes ist. Wenn das Ende der Datei erreicht wird, bleiben die restlichen Felder (Variablen) unverändert (normales Natural-Verhalten).
CSV	Die maximale Feld-/Datensatzlänge für dynamische und große Variablen ist 32766 Bytes. Dynamische Variables werden unterstützt. X-Arrays sind nicht erlaubt und resultieren in einer Fehlermeldung.

Beispiel

```
** Example 'WWFEX1': WRITE WORK FILE
************************************************************************
DEFINE DATA LOCAL
1 EMPLOY-VIEW VIEW OF EMPLOYEES
  2 PERSONNEL-ID
  2 NAME
END-DEFINE
*
FIND EMPLOY-VIEW WITH CITY = 'LONDON'
```

```
  WRITE WORK FILE 1
        PERSONNEL-ID NAME
END-FIND
*
END
```