Eine Kurzübersicht über

OpenVMS und DCL

von

Dipl. Ing. (BA) Matthias Schmitt
www.tecmumas.de

Version 1.1
2017-04-18

1 Inhalt

1 Inhalt 2
2 Vorbemerkungen 3
3 Grundlagen 4
3.1 Zurückrufen von Befehlen 4
3.2 Kontrollsequenzen 4
3.3 Unterprozeß erzeugen 4
3.4 Anzeigen von Informationen 5
4 Arbeiten mit Dateien 6
4.1 Allgemeine Operationen 6
4.2 Arbeiten mit Textdateien 6
4.2.1 Allgemeines 6
4.2.2 Arbeiten mit dem TPU-Editor 7
5 Arbeiten mit Verzeichnissen 8
5.1 Operationen 8
5.2 Backup eines Verzeichnisses 8
6 Schützen von Dateien und Verzeichnissen 9
6.1 Schutzkonzept 9
6.2 Befehle 9
7 Verwendung von logischen Namen und Symbolen 10
7.1 Unterschiede zwischen logischen Namen und Symbolen 10
7.2 Logische Namen 10
7.3 Symbole 10
8 Kommandoprozeduren 11
8.1 Aufrufen einer Kommandoprozedur 11
8.2 Grundlagen 11
8.3 spezielle Befehle in einer Befehlsprozedur 11
8.4 lexikalische Funktionen 12
9 Lizenzverwaltung 13
10 Benutzerverwaltung 14
11 Tips & Tricks 15
11.1 Fragmentierung einer Datei feststellen 15
11.2 Gerätenamen unter VMS 15
12 Stichwortverzeichnis 17

2 Vorbemerkungen

Schreibweisen in diesem Dokument:

- Text in der Schriftart Courier steht für Eingaben:
  `$ directory /size`
- Klein und kursiv geschriebene Begriffe stehen für Variablen:

name	Name der Datei oder des Verzeichnisses
ext	Extension
ver	Version
**datei*	vollständige Bezeichnung der Datei, evtl. mit Platte, Pfad etc.

Fehler gefunden? Anregungen?

Bitte schreiben Sie an:
Matthias Schmitt
Kimbacher Straße 79
D-64732 Bad König
eMail: ms@tecmumas.de

3 Grundlagen

- Eine DCL Kommandozeile besteht aus folgenden Teilen (Teile in eckigen Klammern sind optional):
  `[$] command [/qualifier[=value]...] [parameter [/qualifier...]]`
  `$ directory /since=yesterday *.log`
  - `$` → Prompt
  - *command* → Befehl, z.B. „directory“
  - *qualifier* → Qualifizierer, schränkt den Befehl ein; z.B. „since“
  - *value* → Wert, z.B. „yesterday“
  - *parameter* → beschreibt, worauf der Befehl angewendet wird; z.B. „*.log“
- Ein vollständiger Dateiname sieht wie folgt aus:
  device:[*directory*]*name.extension;version*
  Beispiel: VAX1$DKA100:[HOME.TEST]LOGIN.COM;2
- Bei der Angabe von Unterverzeichnissen muß dem Namen des Unterverzeichnisses stets ein Punkt vorangestellt werden, sonst interpretiert OpenVMS die Eingabe so, als währe das Unterverzeichnis direkt unter der obersten Ebene.
- Bei der Eingabe von DCL-Befehlen und Qualifizierern genügt stets die Eingabe so vieler Buchstaben, dass die Angabe eindeutig ist, z.B. DIR statt DIRECTORY.
- Das Betriebssystem OpenVMS unterstützt folgende Platzhalter:
  * variable Anzahl von Zeichen z.B. HA.*
  % ein Zeichen z.B. H%A.TXT
- Viele Qualifizierer kann man durch NO in der Bedeutung umkehren (z.B. /LOG <-> /NOLOG)

3.1 Zurückrufen von Befehlen

- OpenVMS merkt sich die letzten 20 Befehle
- Der letzte Befehl wird mit der Taste <Pfeil-nach-oben> aufgerufen.
- Die Liste der letzten 20 Befehle erhält man mit
  `$ RECALL /ALL`
- Den *n*-ten Befehl erhält man mit
  `$ RECALL n`
- Den Befehl, der mit *zeichenfolge* anfängt erhält man mit
  `$ RECALL zeichenfolge`

3.2 Kontrollsequenzen

<Ctrl>-C vom Programm definiert
<Ctrl>-O Aus-/Einschalten der Ausgabe am Terminal, ohne das Programm zu unterbrechen
<Ctrl>-T gibt Kurzinformation zu laufendem Prozess aus
<Ctrl>-W Bildschirm neu aufbauen
<Ctrl>-Y unterbricht das laufende Programm
<Ctrl>-Z geregeltes Verlassen des laufenden Programms

3.3 Unterprozeß erzeugen

`$ SPAWN befehlskette`

wichtige Qualifizierer

`/INPUT=datei`	die Kommandoprozedur *datei* wird ausgeführt
`/OUTPUT= datei`	die Ausgaben des Unterprozesses werden in die Datei *datei* geschrieben
`/PROCESS=name`	der Unterprozeß erhält den Namen *name*

`/WAIT`	der Benutzer wartet das Ende des Unterprozesses ab, bevor er weitere Befehle aufruft (default)
`/NOWAIT`	der Benutzer arbeitet im Mutterprozeß weiter

3.4 Anzeigen von Informationen

Mit dem DCL-Befehl `SHOW` *thema* kann man sich Informationen zu einem bestimmten Thema *thema* anzeigen lassen.

Mögliche Themen sind:

`BROADCAST`	Informationen über aktive Sperren
`CLUSTER`	Informationen über Aktivitäten im Cluster
`DEFAULT`	zeigt aktuelle Platte und Verzeichnis
`ENTRY`	Informationen über eigene Einträge in Warteschlangen z.B. Druckaufträge
`KEY /ALL`	zeigt die Belegung der Funktionstasten
`LOGICAL`	zeigt alle logischen Name an
`NETWORK`	Informationen über das Netzwerk
`PROCESS` *`prozname`*	Informationen über den Prozeß *prozname*
`QUEUE` *`qname`*	Informationen über Warteschlange *qname*
`SYMBOL /ALL`	zeigt alle Symbole an
`SYSTEM`	Informationen über das System und alle laufenden Prozesse
`TERMINAL`	Informationen über eigenes Terminal
`TIME`	zeigt aktuelle Systemzeit mit Datum an

4 Arbeiten mit Dateien

4.1 Allgemeine Operationen

Folgende Qualifizierer gelten für alle aufgeführten Dateioperationen:

/BEFORE=*time*	nur die vor *time* erstellten Dateien werden bearbeitet
/CONFIRM	jeder Vorgang muß bestätigt werden
/EXCLUDE=*datei*	*datei* wird nicht bearbeitet
/LOG	der Vorgang wird protokolliert
/SINCE=*time*	nur nach *time* erstellten Dateien werden bearbeitet

Kopieren	$ COPY *quelldatei zieldatei*	
Löschen	$ DELETE *name.ext;ver*	
alte Versionen löschen	$ PURGE	/KEEP=*anz* maximal *anz* Versionen werden aufgehoben
Umbenennen	$ RENAME *altname neuname*	
Verschieben	$ RENAME *name.ext;ver verzeichnis*	

4.2 Arbeiten mit Textdateien

4.2.1 Allgemeines

Erstellen	$ CREATE *name.ext*	Text eingeben, Ende mit <Ctrl>-Z /LOG nach dem Erstellen wird eine Meldung ausgegeben /PROTECTION=code legt Schutz für die Textdatei fest
Anzeigen	$ TYPE *datei*	/BEFORE=*time* nur vor *time* erstellten Dateien werden ausgegeben /EXCLUDE=*datei1 datei1* wird nicht ausgegeben /PAGE die Datei *datei* wird seitenweise ausgegeben
Nach Zeichenketten durchsuchen	$ SEARCH *datei zeichenkette*	für *datei* kann stehen: *name.ext;ver* (auch Platzhalter) [...] Durchsuchen der gesamten Unterverzeichnisse, z.B. wenn der Dateiname unbekannt ist *zeichenkette* muß in Anführungszeichen stehen, wenn der Suchbegriff aus mehreren Worten besteht. Mehrere Suchbegriffe werden durch Komma getrennt.
Drucken	$ PRINT *datei*	/AFTER=*time* der Druckauftrag wird angehalten bis a) die Zeit *time* verstrichen ist (bei Angabe einer Zeitspanne) oder b) der Zeitpunkt *time* erreicht ist (bei Angabe eines Zeitpunktes) /CONFIRM jeder Ausdruck muß bestätigt werden /COPIES=*n* die Dateien *n*-mal drucken /DELETE Dateien nach Druck löschen /EXCLUDE=datei Datei *datei* wird nicht gedruckt /FLAG=keyword Deckblatt drucken

a) vor jedem Druckauftrag: für *keyword* steht ALL
b) nur vor dem ersten Auftrag: für *keyword* steht ONE

`/NAME=jobname` Druckauftrag benennen
`/NOTE=text` ordnet dem Druckauftrag einen Text für Deckblatt zu (Länge maximal 255 Zeichen)
`/NOTIFY` es wird eine Nachricht angezeigt (bei Druckanfang /-ende)
`/QUEUE=qname` legt die Warteschlange fest, Standard ist SYS$PRINT

4.2.2 Arbeiten mit dem TPU-Editor

4.2.2.1 Einführung

- Der Editor wird mit `EDIT /TPU name.ext` aufgerufen.
- Die Eingabe von Befehlen erfolgt durch Drücken der Taste <Ausführen> oder <Do>.
- Einige Funktionstasten
  <F10> Verlassen des Editors (mit Speichern)
  <F11> Umschalten der Schreibrichtung
  <F14> Umschalten 'Überschreiben' und 'Einfügen'
- Der numerische Tastenblock wird durch den Befehl SET KEYPAD EDT aktiviert.
  <PF1>-<Pfeiltaste> Sprung an Textanfang/-ende bzw. zum Zeilenanfang/-ende
  <Suchen> Suchen eines Ausdrucks
  <PF4> Zeile löschen
  <PF1> <PF4> gelöschte Zeile einfügen

4.2.2.2 die wichtigsten Befehle

`BUFFER name`	der Textpuffer *name* wird aufgerufen
`DEFINE KEY=keyn`	die Taste *keyn* wird mit einem Befehl belegt
`ENLARGE n`	aktuelles Textfenster wird um *n* Zeilen vergrößert
`GET FILE datei`	neuer Textpuffer wird erstellt und die Datei *datei* wird geladen
`GO TO name`	springt zu der Textmarke *name*
`QUIT`	Editor verlassen ohne zu sichern
`LEARN`	zur Belegung von Tasten mit mehreren Befehlen / Text
`LINE nr`	springt in die Zeile *nr*
`MARK name`	setzt Textmarke *name*
`REPLACE`	Textteile ersetzen
`SHOW BUFFERS`	zeigt die vorhandenen Textpuffer an
`TWO WINDOWS`	der Bildschirm wird geteilt
`ONE WINDOW`	nur noch ein Textfenster
`OTHER WINDOW`	Umschalten zwischen den Textfenstern
`WHAT LINE`	zeigt die aktuelle Zeilennummer an
`WRITE FILE`	aktueller Textpuffer wird gespeichert, ohne den Editor zu verlassen
`TPU SET (MOUSE,OFF)`	Einfügen durch Cut-&-Paste mit der Maus ermöglichen

5 Arbeiten mit Verzeichnissen

5.1 Operationen

Erstellen	`$ CREATE /DIRECTORY [.name]`	`/LOG` die Umbenennung wird protokolliert `/PROTECTION=code` legt Schutz für das Verzeichnis fest `/VERSION_LIMIT=n` legt die maximale Anzahl Versionen fest
Löschen	`$ SET PROTECTION=(O:RWED)` *`name`*`.DIR;*` `$ DELETE` *`name`*`.DIR;*`	
Inhalt ausgeben	`$ DIRECTORY`	`/BEFORE=time` nur vor *time* erstellten Dateien ausgeben `/EXCLUDE=datei datei` wird nicht ausgegeben `/FULL` es werden Name, Typ, Versionsnummer, Größe, Besitzer, Schutzangaben usw. ausgegeben `/OWNER` der Besitzer wird mit ausgegeben `/SINCE=time` nur nach *time* erstellten Dateien werden angezeigt
Wechseln	`$ SET DEFAULT [.`*`name`*`]`	in das Verzeichnis *name* unterhalb es aktuellen Verzeichnisses wechseln
	`$ SET DEFAULT [-]`	eine Ebene nach oben
Aktuelles ausgeben	`$ SHOW DEFAULT`	

5.2 Backup eines Verzeichnisses

1. Ist das Band neu, so muß es initialisiert werden:
   `$ init` *`tape`*`:` *`name`*
2. Kopieren aller Dateien eines Verzeichnisses
   `$ BACKUP [base_`*`dir`*`...]*.*;*` *`tape:name`*`.bck /log /rewind /label=`*`name`*
   Wichtig: /rewind nur bei der ersten Sicherung auf dem Band verwenden!
3. Band dismounten:
   `$ DISMOUNT` *`tape`*`: /nounload`
4. Band wieder mounten zum Lesen:
   `$ MOUNT /FOREIGN` *`tape`*`:` *`name`*
5. Inhalt der Sicherung anzeigen:
   `$ BACKUP /list` *`tape:name`*`.bck`
6. Band dismounten zum Entnehmen:
   `$ DISMOUNT` *`tape`*`: /unload`
7. Band entnehmen

6 Schützen von Dateien und Verzeichnissen

6.1 Schutzkonzept

In OpenVMS gibt es vier verschiedene Benutzergruppen:

WORLD	alle Benutzer des Rechners / Netzes
GROUP	alle Mitglieder einer definierten Gruppe (z.B. Arbeitsgruppe, Abteilung)
OWNER	der Eigentümer (meist Erzeuger)
SYSTEM	Systembetreuer

Es gibt vier verschiedene Zugriffsarten auf Dateien (D) und Verzeichnisse (V):

READ (Lesen)	D: Erzeugen einer Kopie der Datei V: Liste ausgeben (DIRECTORY)
WRITE (Schreiben)	D: Änderung der Datei (APPEND) V: Dateien hinzufügen oder löschen
EXECUTE (Ausführen)	D: Ausführen der Datei (@ oder RUN) V: auf Dateien zugreifen
DELETE (Löschen)	D: Löschen der Datei (DELETE, PURGE) V: Löschen des Verzeichnisses

6.2 Befehle

Anzeigen des Dateischutzes	`$ DIRECTORY /SECURITY`	Der Dateischutz wird in der Reihenfolge SYSTEM, OWNER, GROUP, WORLD angezeigt.
Ändern des Dateischutzes	`$ SET PROTECTION=`*`neuschutz datei`	*neuschutz* = (S:*code*;O: *code*;G: *code*;W: *code*)
Standard-Dateischutz anzeigen	`$ SHOW PROTECTION`	
Standard-Dateischutz ändern	`$ SET PROTECTION=`*`neuschutz`*` /DEFAULT`	

7 Verwendung von logischen Namen und Symbolen

7.1 Unterschiede zwischen logischen Namen und Symbolen

- Symbole stehen für DCL-Befehle. Sie sind nur prozeßweit definierbar und können nur in DCL verwendet werden.
- Logische Namen stehen für Geräte, Verzeichnisse und Dateien. Sie sind prozeß- und systemweit definierbar und können auch in Programmen und Dienstprogrammen verwendet werden.

7.2 Logische Namen

anzeigen	`$ SHOW LOGICAL` *`log_name`*	Auch Platzhalter möglich
definieren	`$ ASSIGN` *`bez log_name`*	Beide Befehle sind identisch.
	`$ DEFINE` *`log`*`_name` *`bez`*	`/GROUP` logischer Name gilt für die gesamte Gruppe (Privileg GRPNAM oder SYSPRV nötig) `/JOB` logischer Name gilt für den laufenden Job, d.h. für den Prozeß und alle Unterprozesse `/PROCESS` logischer Name gilt nur für den laufenden Prozeß (default)
löschen	`$ DEASSIGN` *`log_name`*	Qualifizierer siehe 'definieren'

7.3 Symbole

anzeigen	`$ SHOW SYMBOL` *`symbolname`*	`/ALL` alle Symbole anzeigen `/GLOBAL` nur aus der globalen Symboltabelle anzeigen `/LOCAL` nur aus der lokalen Symboltabelle anzeigen
definieren	*`symbolname = text`*	lokales Symbol für Komandoprozeduren
	`symbolname == text`	globales Symbol für Prozeß
	`symbolname :== text`	in Text stehen logische Namen, die vor der Zuweisung an das Symbol übersetzt werden

8 Kommandoprozeduren

8.1 Aufrufen einer Kommandoprozedur

`$ SUBMIT` *dateiname* — Die Prozedur *dateiname* wird als Stapeljob abgearbeitet
`/AFTER=`*zeit* Job wird erst nach *zeit* gestartet
`/NOAFTER` Job wird sofort gestartet (default)
`/NAME=`*name* Job erhält den Namen name
`/PARAMETERS=`*pa* der Prozedur können bis zu 8 Parameter, durch Kommas getrennt, übergeben werden; sie stehen in den Variablen p1 bis p8
`/PRINTER=`*name* Kommentare über den Ablauf des Jobs werden auf dem Drucker *name* gedruckt (default ist SYS$PRINT)
`/NOPRINTER` keine Kommentare werden gedruckt (default)

`$ @`*dateiname* — *dateiname* wird ausgeführt

8.2 Grundlagen

- Jede Zeile in einer Kommandoprozedur muß mit einem Dollarzeichen beginnen.
- Jede Kommandoprozedur muß als letzte Befehlszeile „`$ EXIT`“ enthalten.
- Bemerkungszeilen beginnen mit einem Ausrufungszeichen (nach dem Dollarzeichen).
- Es können alle DCL-Befehle verwendet werden.
- Es können Symbole verwendet werden
- Werden Symbole als Parameter für DCL-Befehle benutzt, müssen sie in Apostrophe eingeschlossen werden.

8.3 spezielle Befehle in einer Befehlsprozedur

auf Bildschirm schreiben
```
$ WRITE SYS$OUTPUT
```

Benutzereingaben
```
$ INQUIRE /NOPUNCTUATION symbol "text"
$ READ /PROMPT="text" SYS$COMMAND symbol
```

Steuerkonstrukte
```
$ IF ausdruck
$    THEN
       befehle
$    ELSE
       befehle
$ ENDIF
```

in *ausdruck* sind folgende Entscheidungen möglich:

`.EQS.`	gleich (Zeichenkette)
`.NES.`	ungleich (Zeichenkette)
`.GTS.`	größer als (Zeichenkette)
`.GES.`	größer als oder gleich (Zeichenkette)
`.LTS.`	kleiner als (Zeichenkette)
`.LES.`	kleiner oder gleich (Zeichenkette)

Für Operationen mit ganzen Zahle wird der Buchstabe S weggelassen.

Sprung
```
$ GOTO label
$ label:
```

Sprung in Unterprogramm
```
$ GOSUB label
$ label:
$ RETURN
```

Rechnen

- Werden Zuweisungen an Variablen (Symbole) in Anführungszeichen gesetzt, so werden sie als Textvariablen behandelt. Mögliche Operationen sind Addition und Subtraktion.
- Werden die Zuweisungen ohne Anführungszeichen geschrieben und handelst es sich bei den zugewiesenen Zeichen um Zahlen, so werden die Variablen als ganzzalige Variablen behandelt. Mögliche Operationen sind die vier Grundrechenarten, für die Division wird der Schrägstrich '/' verwendet.

Dateizugriffe

`$ OPEN /WRITE logischer_name datei`	öffnet zum Schreiben
`$ OPEN /READ logischer_name datei`	öffnet zum Lesen
`$ OPEN /APPEND logischer_name datei`	öffnet zum Anhängen
`$ WRITE logischer_name text`	schreibt text
`$ READ logischer_name symbol`	liest eine Zeile ein `/END_OF_FILE=marke`, bei Dateiende wird zu *marke* gesprungen.
`$ CLOSE logischer_name`	schließt Datei

Fehlerbehandlung `$ ON ERROR befehl`

8.4 lexikalische Funktionen

`F$DIRECTORY()`	Name des aktuellen Verzeichnisses
`F$EXTRACT(start,länge,zkette)`	entnimmt der Zeichenkette *zkette* eine Zeichenkette der Länge *länge*, die bei *start* beginnt
`F$INTEGER(ausdruck)`	wandelt *ausdruck* in eine ganze Zahl um
`F$LENGTH(zkette)`	gibt die Länge von *zkette* an
`F$LOCATE(zkette1,zkette2)`	sucht *zkette1* in *zkette2* und gibt (wenn gefunden) die Startposition zurück (sonst die Länge) Achtung: *zkette1* wird mit 0 beginnend gezählt!
`F$MODE()`	gibt INTERACTIVE, BATCH, NETWORK oder OTHER zurück
`F$PROCESS()`	gibt den Namen des aktuellen Prozesses zurück
`F$PID()`	gibt die ID des aktuellen Prozesses zurück
`F$STRING(ausdruck)`	wandelt *ausdruck* in eine Zeichenkette um
`F$TIME()`	gibt die aktuelle Systemzeit im Format *dd-mmm-yyyy hh:mm:ss.cc* zurück

9 Lizenzverwaltung

```
$ license befehl
```

Befehl	Bedeutung
`list` *produkt* `/full`	anzeigen von *produkt*
`disable` *produkt* `/authorisation=`*auth*	*produkt* deaktivieren (mit Authorisation Number *auth*)
`unload` *produkt* `/authorisation=`*auth*	*produkt* aus Lizenzcache nehmen
`load` *produkt*	*produkt* in Lizenzcache nehmen
`delete` *produkt* `/authorisation=`*auth*	Lizenz löschen

Außerdem werden die aktuellen Lizenzen über das Kommando
`$ show license`
angezeigt.

10 Benutzerverwaltung

```
$ SET DEFAULT SYS$SYSTEM
$ MC AUTHORIZE
```

Identifier neu anlegen:
`UAF> ADD /IDENTIFIER name`

Identifier zu Account hinzufügen:
`UAF> GRANT /IDENTIFIER name account`

11 Tips & Tricks

11.1 Fragmentierung einer Datei feststellen

```
$ DUMP /HEADER /BLOCK=COUNT=0 datei
```

Bei `/BLOCK=COUNT=0` wird der Dateiinhalt nicht mit ausgegeben.

11.2 Gerätenamen unter VMS

cn	DECnet über CI
cs	Console Terminal an Q-Bus-Workstations
dad	CD-ROM an Infoserver
dk	SCSI-Platte
du	MSCP-Platte
es	Ethernet-Controller an buslosen Maschinen (DESVA)
et	Ethernet-Controller an BI-Bus (DEBNA, DEBNI)
ex	Ethernet-Controller an XMI-Bus
ft	Pseudo-Terminal
fy	SET HOST /HSC
ga	DECwindows Color Graphic Controller (GPX)
gc	DECwindows Monochrome Graphic Controller
gf	DECwindows Color Graphic Controller (SPX)
ik	DECwindows Keyboard Input Processing
im	DECwindows Mouse Input Processing
in	DECwindows General Input Processing
lac	Pathworks PCDISK
lad	Pathworks Disk Service
lans	LANsess
last	Transport für Pathworks Disk Service bzw. Infoserver
lt	DECserver Terminal (LAT)
mb	Mailbox
mk	SCSI Tape
mu	TMSCP Tape
nl	NULL Device
nd	MOP
net	DECnet Mailbox
nv	PSI Terminal Emulator
nw	PSI Mailbox
op	Operator Console
pa	CI Adapter
pb	Tape Controller für BI-Bus (TBK50, TBK70)
pe	VAXcluster über Ethernet
pk	SCSI Disk Controller
pt	TMSCP Tape Controller am Q-Bus (TQK50, TQK70)
pu	MSCP Disk Controller
py	DECwindows Terminal Controller
rt	DECnet Remote Terminal
sj	DSV-11 (synchrone Schnittstelle)
tk	Tektronix Terminal Emulator unter VMS
tt	direkte Schnittstelle (z.B. buslose Workstation)
tw	DECwindows Terminal Emulator
tx	direkte Schnittstelle (DHV11 usw.)
va	VWS Color Graphic Controller

vc	VWS Monochorme Graphic Controller
ws	DECwindows Display
wt	VMS Terminal Emulator
xq	Ethernet Controller am Q-Bus (DEQNA)
xt	XX Windows Terminal über LAT
zs	DST-32 (synchrone Schnittstelle)

12 Stichwortverzeichnis

@ 11
ASSIGN 10
BACKUP 8
CLOSE 12
COPY 6
CREATE 6
CREATE /DIRECTORY 8
Dateiname 4
DEASSIGN 10
DEFINE 10
DELETE 6, 8
DIRECTORY 8
DIRECTORY /SECURITY 9
DUMP 15
EDIT 7
F$DIRECTORY() 12
F$EXTRACT() 12
F$INTEGER() 12
F$LENGTH() 12
F$LOCATE() 12
F$MODE() 12
F$PID() 12
F$PROCESS() 12
F$STRING() 12
F$TIME() 12
GOSUB 11
GOTO 11
Identifier 14
IF 11
INQUIRE /NOPUNCTUATION 11
license 13
Logische Namen 10
MC AUTHORIZE 14
ON ERROR 12
OPEN 12
Platzhalter 4
PRINT 6
PURGE 6
READ 12
READ /PROMPT 11
RECALL 4
RENAME 6
SEARCH 6
SET DEFAULT 8
SET PROTECTION 8, 9
SHOW 5
SHOW DEFAULT 8
show license 13
SHOW LOGICAL 10
SHOW PROTECTION 9
SHOW SYMBOL 10
SPAWN 4
SUBMIT 11
Symbole 10
TYPE 6
WRITE 12
WRITE SYS$OUTPUT 11