allegro news Nr.57

Zur Auslieferung der V20

Ausgabe 2000/1, 22.5.2000
Diese Ausgabe hat 16 Seiten

Universitätsbibliothek der Technischen Universität Braunschweig, Pockelsstr. 13, D-38106 Braunschweig. Tel. (0531) 391-5011, FAX -5836

www.allegro-c.de/allegro/

Wie finden Sie Goethe ...

... in einem "Brockhaus" oder "Meyers" von vor 1970? Vielleicht gar nicht - versuchen Sie es mal. Es sei denn, Ihnen fällt auf, dass da zwischen "Goethals" und "Goffiné" ein seltsamer Blackout vorliegt. Da steht weder "Goethe" noch "Götterspeise" noch "Götzendienst"! Ähnliches widerfährt dem, der "Caesar, Gaius Julius" finden will: nichts zwischen "Caernarvon" und "Caëtani". Jetzt werden Sie aber doch stutzig: es *kann* nicht sein (im *Brockhaus*!), dass Dichter und Diktator schlicht vergessen wurden. Das wurden sie auch nicht, sie stehen nur woanders, und zwar hinter "Gotha" bzw. hinter "Casanova". Wer Katalogisieren gelernt hat und noch Zettel sortieren musste, dem wurde dabei bewusst, dass das alphabetische Ordnen keineswegs eine triviale Sache ist. Neben vielen anderen Details machen dabei die deutschen Umlaute immer Ärger. Wohl weil sie sprachgeschichtlich von den Grundbuchstaben abstammen, hat man sie bis ins zwanzigste Jahrhundert hinein beim Ordnen gleichgeschaltet, d.h. die Pünktchen ignoriert. Die Schreibungen mit "oe" statt "ö" sind jedoch gleichwertig (ein Phänomen, das es nur im Deutschen gibt), und da liegt dann das Problem, wenn man zu Goethe kommt: man hielt es für sinnvoll, die Ordnung an der Sprachgeschichte festzumachen, also unter anderem die Kombination "oe" so zu ordnen, als sei das "e" nicht da. (In alten Büchern sieht man auch oft, dass über dem 'o' keine Pünktchen schweben, sondern ein kleines 'e'.) Nun, wenn man das einmal weiß, ist das kein Problem - wären da nicht die Fremdwörter und ausländischen Namen, in denen die Kombination "oe" keineswegs unserem "ö" gleichwertig ist, siehe das Beispiel "Goethals" (ein amerikanischer Name). Die alten Nachschlagewerke mussten deshalb Ausnahmen machen, sie konnten nicht mechanisch "oe" in *jedem* Fall wie "o" ordnen. Erst recht nicht "ae" immer wie "a" oder gar "ue" wie "u". Stellen Sie sich vor, was mit "Mensaessen" passieren würde, mit "Bauer" oder mit "Duell". Aber für denkende, der deutschen Sprache mächtige Menschen ist es nicht schwierig, diese Ausnahmen zu erkennen und zu berücksichtigen. Alle alten Nachschlagewerke verfuhren so, und auch die großen alten Bibliographien im deutschen Sprachgebiet, Heinsius und Kayser, und in deren Nachfolge auch das Deutsche Bücherverzeichnis, bis in die 50er Jahre. Warum wurde hier eine Jahrhunderte alte Gewohnheit aufgegeben? Weil keiner mehr Goethe finden konnte? Wohl auch deswegen, weil zunehmend keine denkenden Menschen mehr das Ordnen besorgten, sondern Maschinen, die wirklich nur mechanisch sortieren und die Ausnahmen nicht erkennen und berücksichtigen konnten.

Die "Preußischen Instruktionen" und der Gesamtkatalog führten aber schon 1899 die aufgelöste Ordnung der Umlaute ein. In den wissenschaftlichen Bibliotheken war wohl der Anteil ausländischer Namen, und damit die Zahl der notwendigen Ausnahmen, schon unangenehm groß geworden. Die RAK-Eltern haben die Auflösung dann als bewährte Praxis übernommen.

Erst in jüngerer Zeit macht sich bemerkbar, dass das Suchen in deutschen Katalogen deswegen für Ausländer ein Problem darstellt (und für uns das Suchen in ausländischen Katalogen!): überall sonst in der Welt wird nämlich "Müller" wie "muller" geordnet, also nach altdeutscher Manier. Jedoch "Goethe" keineswegs wie "gothe", da liegt der Unterschied. Nun könnten wir zwar leicht unsere Computer anweisen, es ebenso zu machen und uns dann eben wieder mal umgewöhnen, wären da nicht diejenigen Personen, die sich wirklich "Mueller" schreiben und nicht "Müller". Und wären da nicht große Mengen von Altdaten aus der Steinzeit der EDV, wo man noch keine Umlautzeichen hatte und jeden Müller als Mueller eingab. Daher hätten wir dann für jeden Umlaut-Namen potenziell zwei Stellen im Katalog, und man müsste oft an beiden Stellen nachsehen. In ausländischen Katalogen ist das wirklich so, nur spielen dort die deutschen Namen quantitativ eine viel geringere Rolle.

Wörterbücher und Lexika führten übrigens nicht die Umlaut-Auflösung ein, sie ordnen weiterhin nach der Gleichung ö=o, nur die Gleichung oe=o wurde abgeschafft. Wörterbücher können so verfahren, weil sie überwiegend Wörter auflisten, nur zum kleinen Teil Personennamen. Telefonbücher dagegen halten es wie Bibliothekskataloge, aus demselben Grund.

Wer sich in das Thema weiter vertiefen möchte, kann ein Gutachten nachlesen, das zur Frage der Umlautordnung voriges Jahr erstellt wurde (http://www.allegro-c.de/allegro/formate/umlaut.htm). Als Konsequenz, und deshalb sprechen wir das Thema hier an, wurden die Standard-Indexparameter CAT.API geändert, einer Empfehlung des Gutachtens folgend. Personennamen werden doppelt indexiert: "Müller" wird als "mueller" *und* als "muller" ins Register 1 eingetragen.

Für Titelwörter wird dieses Verfahren nicht angewendet. Dort scheint es weniger sinnvoll. Etwas anderes aber wurde im Register 3 verbessert: bei Bindestrich-Kombinationen werden nun beide Bestandteile eingetragen, die Kombination als solche aber auch. Also: aus "deutsch-polnisch" wird "deutsch", "polnisch" und "deutsch-polnisch", bisher fehlte "deutsch". Der Bindestrich bleibt aber erhalten, sonst stünde "deutschpolnisch" im Register (bei Pica ist das so).

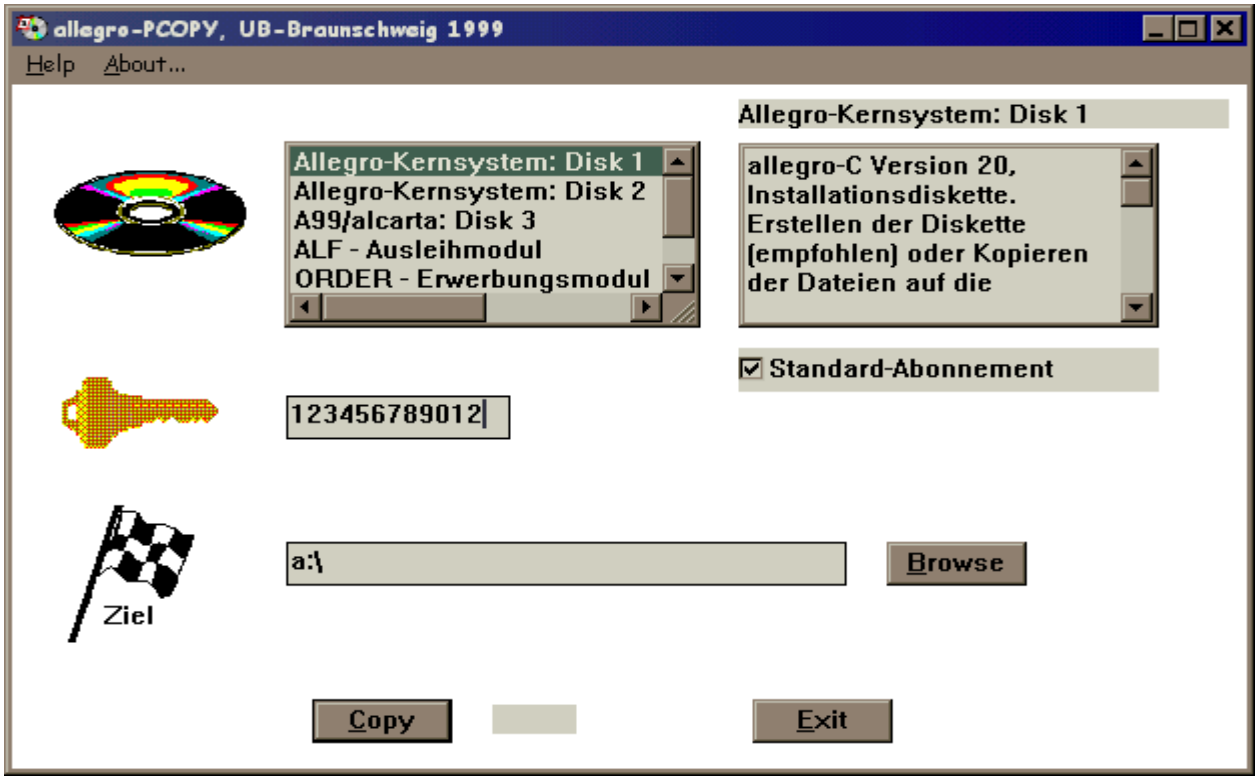
Um diese Dinge zu realisieren, hätte man allerdings keine neue Version gebraucht, das konnte man auch mit der alten machen! Die Entwicklung hatte sich in den letzten Monaten mit ganz anderen und viel schwierigeren Problemen herumzuschlagen.

Nach einer überaus hektischen Testphase wurde am 8. Mai 2000 um 8.20 Uhr die Version 20 festgeschrieben. Abrupt kehrte in der E-Mail-Liste Stille ein, nachdem in scheinbar endloser Kette ein Wunsch den nächsten nach sich gezogen hatte, jedes gelöste Problem mindestens ein neues hervorbrachte und die Innovationen sich überstürzten. Warum **V20**? Das steht im Vorwort des neuen Handbuchs. Mit dem Befehl `h handbuch` im Schreibfeld von *a99* ruft man die Online-Ausgabe auf.

Festschreibung : beinahe wie ein Damoklesschwert hing dieses Wort über der Szene, doch als sie dann kam, war die Hektik wie ein Spuk verflogen. Vorerst nehmen wir mal an, dass es tatsächlich keine akuten Probleme mehr gibt, und wenden uns in Ruhe der Frage zu, wie man die verbliebenen DOS-Funktionen auch noch in die Windows-Umgebung überführen kann. Um mit Goethe zu sprechen: In der Kunst ist nur das Beste gut genug (*Italienische Reise*). Nur in der Kunst? Und ist Windows das Beste? Wird es, wenn V21 kommt, wieder heißen wie beim Literarischen Quartett: Vorhang zu – und alle Fragen offen?

V20 installieren

Sollte das Startprogramm der CD nicht von selber anlaufen, müssen Sie es über den Explorer oder mit der Hand vom DOS-Fenster aus starten. Es heißt **`autorun32.exe`**. Unter Windows 3.*x* muß man über den Dateimanager das Programm **`start16.exe`** starten. Dann kommt ein Programmfenster mit Bild und drei Knöpfen auf der rechten Seite. Mit den ersten beiden können Sie auf die Datenbanken zugreifen, mit dem dritten Knopf werden die Programme installiert. Der Weg zu den weiteren Inhalten auf der CD-ROM führt über das Menü *Information*. Von hier aus finden Sie detaillierte Hinweise zur Installation der Programme, den Zugang zu den WWW-Seiten und allen bisher erschienenen ***allegro news*** bis Nr.56.
Nun also zur Installation. Nach Druck auf den besagten Button startet das Programm **PCOPY** (das man auch mit der Hand vom Verzeichnis PROGRAMM aus starten kann). Sie sehen dieses Bild:



Jetzt wird die so genannte **Schlüsselzahl** gebraucht, die wir den Abonnenten im Begleitschreiben mitteilen. Zuerst wählt man oben in der Auswahlliste, welche Diskette erstellt werden soll, dann trägt man die 13stellige Zahl ein. Ist sie falsch, kommt eine Fehlermeldung, ist sie richtig, kann man im Feld "Ziel" noch eintragen, wohin die entschlüsselten Dateien kopiert werden sollen. Entweder stellt man sich tatsächlich Disketten her (3 leere Disketten bereithalten!), dann übernimmt man die Vorgabe **a:**\, oder man trägt als Ziel den Namen eines Verzeichnisses ein, z.B. C:\TEMP (für alle drei Disketten des Kernsystems dann dasselbe Verzeichnis). Es dauert nur Sekunden, bis die Diskette bzw. die Kopie dann fertig ist.

Alternativ können Sie über den mitgeteilten FTP-Zugang auch die Inhalte der drei Disketten über das Netz beziehen. Sie liegen im Verzeichnis AC20 des FTP-Servers. Dort ist auch ein Verzeichnis UPDATE mit den aktuellen Versionen derjenigen Programme, die inzwischen wieder verbessert worden sind! (Bei Gelegenheit mal reinschauen)

Wenn Sie weder Internet-Anschluss haben noch einen Rechner mit Windows, besuchen Sie jemand, der einen solchen hat und stellen Sie sich dort die Disketten her.

Nun kann die eigentliche Installation beginnen. Es handelt sich um ein DOS-Programm. Das ist deswegen notwendig, weil es noch immer viele Anwender gibt, die nicht über Win'95 oder NT verfügen und wir nicht den Aufwand treiben können, noch ein zweites Installationsprogramm zu machen.

A. Installation von Diskette

a) **Neu-Installation**: Diskette 1 (Kernsystem) einlegen, INSTALL.BAT starten (anklicken oder im DOS-Fenster den Befehl `a:install` eingeben). Danach den Anweisungen folgen, die das Programm gibt.

b) **Update-Installation**: Diskette 2 einlegen. Vom ***CockPit*** aus den Menüpunkt "Neue Version installieren | Update-Installation" starten (dann wird INST.BAT gestartet)

B. Installation von der Platte

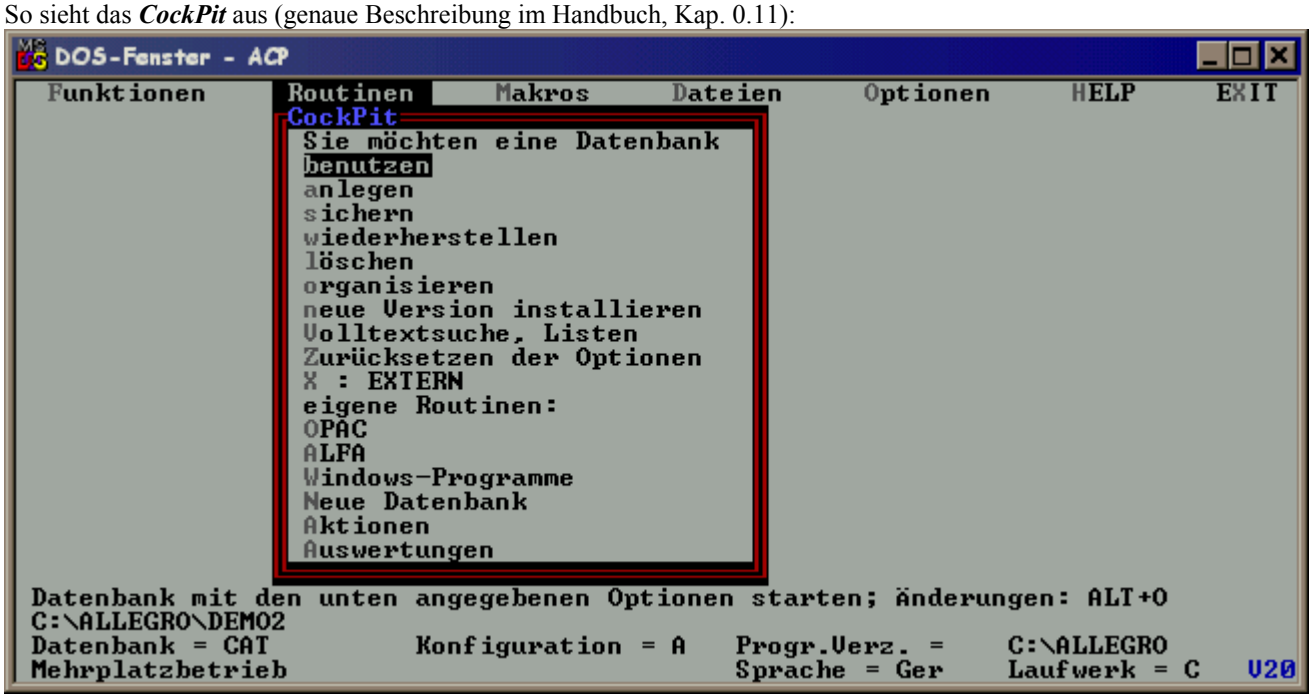
Hat man die Dateien auf ein Verzeichnis kopiert, z.B. C:\TEMP, dann startet man so:
In einem DOS-Fenster gibt man diesen Befehl ein:

`c:\temp\install c:\temp c` für die Neu-Installation
bzw. `c:\temp\inst c:\temp c` für die Update-Installation

Statt des **c** am Ende kann man einen anderen Laufwerksbuchstaben setzen, z.B. **k**, wenn auf eine andere Platte als C: installiert werden soll. Auf der Zielplatte entsteht dann das Verzeichnis **\ALLEGRO** mit der fertigen Installation und es startet sogleich das ***CockPit***. Für Einsteiger: das ist das DOS-Programm, mit dem alle Vorgänge gestartet werden.

Probleme bei der Installation? → Kap. 0.10 im Handbuch.

So sieht das ***CockPit*** aus (genaue Beschreibung im Handbuch, Kap. 0.11):



Dieses DOS-Hauptprogramm starten Sie dann immer mit dem Befehl **cp** auf dem ***allegro***-Verzeichnis.
Der nächste Druck auf Enter startet das DOS-Hauptprogramm PRESTO (Taste F1 für Hilfe, und Handbuch Kap. 1).
Der Menüpunkt "Windows-Programme" ermöglicht einen Start der beiden Programme ***a99*** und ***alcarta***. Besser ist es auf die Dauer, wenn Sie sich für Ihre eigene(n) Datenbank(en) jeweils eine Datei xyz.INI anfertigen (nehmen Sie eine Kopie von **A99.INI** als Grundlage und tragen Sie Ihre eigenen Werte darin ein), dann können Sie eine Desktop-Verknüpfung anlegen mit dem Programmaufruf `a99 xyz` bzw. `alcarta xyz`. Verwenden Sie dafür das mitgelieferte Icon `allegro.ico`.

Erste Schritte

- Ganz dringend ist zu empfehlen, im Handbuch die Seite hinter dem Vorwort zu lesen: **Leitfaden für die Einarbeitung**. Mit "trial and error" brauchen Sie garantiert mehr Zeit - um weniger weit zu kommen.
- Generell sollten Sie die ersten Erfahrungen mit der DEMO-Datenbank machen: Diese erscheint, wenn Sie nach dem ***CockPit***-Start [Enter] drücken. Taste **F1** bringt dann Hilfe.
- Die erste eigene Datenbank können Sie anlegen mit dem Menüpunkt "**Neue Datenbank**". Sie erhalten dann die Auswahl
  - NEU (eigene Datenbank mit demselben Schema wie die DEMO-Datenbank, ohne Ausleihe/Erwerbung)
  - Erweitert (dasselbe, aber mit den erweiterten Parametern, die auch Ausleihe und Erwerbung ermöglichen)
  - Manuell (startet das PRONTO-Programm zum Anlegen einer eigenen Datenbank mit eigener Struktur)

Bevor Sie aber eine Datenbank "manuell" erstellen, also mit neuer, selbst entworfener Struktur, sollten Sie einige Erfahrungen gesammelt haben. Wenn Sie mit dem "Leitfaden" durch sind, werden Sie verstehen, warum.

Die Windows-Programme haben ein umfangreiches Hilfesystem, das man "online" abrufen kann (mehr dazu ab Seite 7). Man drückt auf den Button mit dem großen Fragezeichen, dann geht's los. Oder: im Schreibfeld den Befehl **h doku** eingeben.
Tip: Mit "Option | Display Size" kann man die Anzeige zwischen normal und breit hin und zurück schalten. Ansonsten kann man das ganze Programmfenster in der Größe beliebig einstellen. (Empfohlene Auflösung: mindestens 800x600)

Neue FLEX-Befehle

In der vorigen Nummer war die FLEX-Sprache komplett dokumentiert worden. Danach jedoch wurden bis zur Festschreibung der V20 noch etliche Befehle neu eingebaut oder verbessert. Hier folgt die Übersicht der Neuerungen, gedacht für diejenigen Leser, die sich gezielt darüber informieren wollen. Denn ansonsten erhält man die vollständige Dokumentation als Online-Hilfe, wenn man im Schreibfeld **`h flex`** eingibt. Printer-Button drücken, und der Text wird ausgedruckt!
Doch zuerst ein
Tip: Wie bringt man **Kategorien in den Hintergrundspeicher**? Es gibt dafür zwar keinen Befehl, doch es geht:

`... #nnn\transfer #nnn\insert #nnn ...`
zuerst #nnn löschen (dabei wird es in den Hintergrund kopiert), dann in die iV holen und in #nnn zurück

`... new\erase ...`
Neuen Satz anlegen (dabei wird der aktuelle Satz komplett in den Hintergrund kopiert!), dann den leeren Neusatz sofort wieder beseitigen (dann kommt der kopierte Satz wieder als aktueller Satz zurück)

Neu ist ein weiterer AutoFLEX: der Vor-FLEX. Er wird ausgeführt unmittelbar VOR der Erstellung einer Anzeige, also vor Ausführung der Anzeigeparameter. Man muss ihn speichern in der Variablen `#uX=` , das genügt. Das kann z.B. auch im START.FLX geschehen, damit gleich bei der ersten Anzeige der Vor-FLEX schon da ist!
Der Vor-FLEX kann natürlich auf die Elemente des aktuellen Satzes zugreifen, denn dieser ist es ja, der gerade für die Anzeige aufbereitet werden soll und deshalb im Arbeitsspeicher vorliegt.

Neu ist, neben der "internen Variablen" (einer Zeichenkette) die "**interne Zahl**" (kurz **iZ**). Auch diese hat keinen Namen, sondern die Rechenbefehle beziehen sich direkt auf sie, ohne dass sie genannt wird. Verwenden kann man sie dann in den Befehlen **`variable`** und **`write`** als "besondere Variable" Z, siehe unten.
Folgende Zeichen bilden jeweils einen Rechenbefehl, d.h. die FLEX-Zeile besteht dann nur aus diesem einen Zeichen:

`=` **iV** in Zahl umwandeln und in **iZ** speichern
`+` **iV** in Zahl umwandeln und zu **iZ** addieren, bzw. **iZ-iV** bilden, bzw. **iZ/iV,** bzw. **iZ*iV**
`-` Das Ergebnis steht jeweils wieder in der **iZ**, bis es erneut überschrieben wird.
`/`
`X`

`close` Die vorher im selben FLEX (!) mit **`open`** geöffnete Datei wird geschlossen.
(Eine in einem FLEX geöffnete Datei wird jedoch automatisch bei Beendigung des FLEXes geschlossen.)

`Download` **NEU**
`download`
Aktuellen Satz exportieren (wie über Menü Export).
Wenn **D** statt **d** gesetzt ist: Export über PrintParameter statt ExportParameter, aber in dieselbe Datei.
Tip: Soll nur ein bestimmter Abschnitt in den Exportparametern ausgeführt werden: vorher eine Variable besetzen, z.B. `#uFL` (mit Befehl `var "xxx"\ins #uFL`), und in den Parametern am Anfang einen Sprung einbauen:
`#uFL +A e0`
Die Parameter und die Ausgabedatei kann man mit dem Befehl **`xport ...`** ändern.

`Download set` **NEU**
`download set`
Die gesamte aktuelle Ergebnismenge exportieren (wie über Menü Export)

`fcopy` `name1 name2`
Datei `name1` wird kopiert auf `name2`. Fehlen `name name2,` wird der Inhalt der iV genommen, d.h. es müssen darin zwei Namen hintereinander stehen. Beide Namen dürfen vollständige Pfadnamen sein; Vorsicht: innerhalb des FLEX muss jeder `\` dann verdoppelt werden, in der iV aber nicht!

`fetch` `number`
Aus der mit **`open`** geöffneten Datei werden `number` Bytes in die interne Variable gelesen. Es wird ... angehängt, wenn die Datei noch nicht zu Ende war. Mit **`if no ...`** kann man prüfen, ob überhaupt gelesen werden konnte.
Normalerweise wird man aber mit **`get`** ganze Zeilen aus der Datei lesen.

`fnam` `bezeichnung|typ`
Eine Dateiauswahl-Box wird präsentiert. *bezeichnung* steht dann in der Zeile unter dem Eingabefeld der Box, *typ* ist eine Angabe wie z.B. `P*.?PR` oder `*.*`. Dadurch wird gesteuert, was für Dateinamen überhaupt gezeigt werden. Fehlt dies, werden alle gezeigt. Mit **`if no`** kann man checken, ob nichts ausgewählt wurde. *Wenn* ein Name ausgewählt wurde, steht er anschließend in der iV, und zwar als kompletter Pfadname. Anschließend kann man z.B. sofort mit **`open`** diese Datei öffnen lassen.

fsize *dateiname*
Die Größe der Datei in Bytes wird in die iV übertragen. Wenn *dateiname* fehlt, wird der Inhalt der iV genommen (anschließend ist dieser dann überschrieben!).
Mit **if no ...** kann man checken, ob es die Datei gar nicht gibt. Mit **if g**N **...** und **if l**N **...** kann man hinterher prüfen, ob sie größer oder kleiner als ein bestimmter Wert *N* ist.

get #xyz [Änderung! War vorher **read**]
get [iV]
get +[iV]
Aus der mit **open ...** geöffneten Datei wird eine Zeile gelesen und in die Kategorie #xyz bzw. in die iV kopiert.
#xyz kann eine normale Kategorienummer, aber auch eine #u-Variable sein.
Mit **ascii** bzw. **ansi** kann man die iV umwandeln. Wenn also die Datei ANSI-Daten hat, man braucht aber ASCII: **get iV\ascii\ins #uxy** statt nur **get #uxy**.
Mit **if cancel ...** kann man checken, ob die Datei zu Ende war (d.h. es konnte nichts mehr gelesen werden), mit **if no ...** ob sie gar nicht geöffnet war, mit **if yes ...** ob es geklappt hat. Wenn '+' gesetzt ist, wird an die iV angehängt.

if deleted command
Ist der aktuelle Satz ein gelöschter Satz? Dann command ausführen.

if Locked command
Ist der aktuelle Satz gerade gesperrt? ...

if >N command
if <N command
Wenn die aktuelle Ergebnismenge größer bzw. kleiner als N ist, wird command ausgeführt.

if gN command
if lN command
Wenn die **iV** als Zahl größer bzw. kleiner als N ist, wird command ausgeführt. (Mit der iZ passiert dabei nichts.)

if =#nnn command (vgl. oben den Befehl **if #nnn ...** : nicht dasselbe!)
Wenn die iV mit dem Anfang des Inhalts von **#nnn** übereinstimmt, wird command ausgeführt.
#nnn kann eine Kategorie oder eine #u-Variable sein.

if %abc% command
Wenn die iV die Zeichenkette **abc** enthält, wird command ausgeführt.

if _abc_ command
Wenn der aktuelle Datensatz die Zeichenkette **abc** enthält, wird command ausgeführt.

if *#nnn_abc_ command
Wenn Kategorie **#nnn** im aktuellen Datensatz die Zeichenkette **abc** enthält, wird command ausgeführt.

read #xyz geändert, jetzt **get ...**
read [iV]

rename *oldname newname*
Datei mit dem Namen *oldname* umbenennen in *newname*. Mit **if no** *command* kann man checken, ob es gelungen ist.
Fehlen *oldname newname*, wird der Inhalt der iV genommen.

set b Bookmark : Lesezeichen auf den aktuellen Satz setzen. (Anzeigen: **show b**)

set c0/c1
Eingabedaten sind ASCII / ANSI (e.adt und **insert**-Befehle) Default ist 0.

set d/dn/db/d+/d-
Das Anzeigefeld wird auf **n**ormale Größe bzw. auf Übergröße (**b**ig) eingestellt. Fehlt 'n' oder 'b', dann wird der aktuelle Zustand in den anderen umgeschaltet.
Mit + und - wird die Schriftgröße im Anzeigefenster erhöht bzw. gesenkt. Das funktioniert nur, wenn im RTF-Text der Anzeige keine zusätzlichen Befehle \fs*nn* vorkommen, denn diese haben dann Vorrang! Die Grundgröße wird in den Dateien DISPHEAD.RTF, HELPHEAD.RTF etc. gesetzt. Bei RTF-Hilfedateien funktioniert die Sache nicht, denn in diesen kommen normalerweise viele \fs-Befehle vor.

set e0/e1
Die Editor-Fehlermeldungen werden aus- bzw. eingeschaltet. (z.B. Komma im Namen, Artikel am Anfang, ...)

show *xxx*

Steuerung des Auswahlfeldes (Listenfenster links). Für *xxx* kann stehen:

<	Back-Button (links oben) : vorigen Satz bzw. Hilfetexte
>	Forward-Button
reserve	Hintergrundspeicher
record	Datensatz-Kategorien
bookmarks	die Lesezeichen als Ergebnismenge zeigen
sets	Liste der Ergebnismengen (auch: `show ergeb`)
cfg	Konfiguration
list	Kurzliste der aktuellen Erg.Menge anzeigen, als Fenster
List	dasselbe, aber im Anzeigefeld
keys	Registereinträge zum aktuellen Satz anzeigen
iv	Inhalt der internen Variablen im Schreibfeld auswerfen (zum Editieren)
IV	Inhalt der internen Variablen im Anzeigefeld anzeigen (anhängen, wenn +IV), das Zeichen ¶ (Code 20) wirkt als Zeilentrennung
B	Zeige den Inhalt der iV binär an (auch hier ist +B möglich)

Jeweils 3 Buchstaben des Befehls genügen, z.B. sho rec.

variable / write

Neu hinzugekommen: die Manipulationsbefehle t und T. Wenn man schreibt `... #nnn(t"xyz")` bzw. `... nnn(T"xyz")`, dann ergibt das den Bestandteil von #nnn, der vor bzw. hinter dem letzten darin vorkommenden "xyz" steht. Wenn "xyz" nicht vorkommt, ist das Ergebnis bei T leer, bei t die unveränderte #nnn.

xport p *name*
xport t *name*

Parameterdatei *name.c*PR bzw. *name.c*PT für den Export laden. Im zweiten Fall wird die Tabellendatei zur schon vorhandenen Export-Parameterdatei hinzugeladen. Das entspricht dem Befehl **t***name* in der Exportsprache.
Wenn *name* fehlt, wird der Inhalt der iV genommen. Steht **?** statt *name*, wird eine Dateiauswahl angeboten.

xport f *filename*
xport f +*filename*

Nachfolgende Exporte (per **download** oder **write**) sollen in die Datei *filename* gehen.
Das '+' bewirkt, dass an die existierende Datei angehängt wird, sonst wird sie überschrieben.
Fehlt der *filename,* wird der Inhalt der iV als Dateiname genommen.

Ferner wurde die Liste der "besonderen Variablen" noch erweitert, die man in den Befehlen **variable** und **write** einsetzen kann. Hier die aktuelle Liste: (die ersten 5 entsprechen den bekannten Optionen der DOS-Programme)

B	Name der Datenbank (vorher N)
D	Datenbank-Pfadname (hat einen \ am Ende)
K	Name der Konfigurationsdatei
K1	nur der erste Buchstabe davon
P	Name des Programmverzeichnisses (mit \ am Ende)
F	File : hinter F folgt direkt ein Dateiname; diese Datei wird in die Ausgabe eingefügt folgt kein Dateiname, wird Inhalt der iV als Dateiname genommen
I	Indexliste : die Liste der symbolischen Registernamen (I-Befehle der Indexparameter)
R	Restriktionen : Liste der Namen der Restriktionen (R-Befehle) Trennzeichen bei I und R ist der Code 20 (¶)
S	Short-title Überschrift (aus Zeile |<="..." der Indexparameter)
T	Titel der Datenbank (aus Zeile |a="..." der Indexparameter)
Z	Wert der internen Zahlenvariablen **iZ**
f	find Befehl (Name) der letzten Erg.Menge
h	Überschrift des Registers, dessen Nummer in der iZ steht (vorher mit = hineinschreiben)
i	interne Nummer des Datensatzes
j	Nummer der .cLD-Datei
l	Größe ("length") der Ergebnismenge
n	neue Zeile
p	Primärschlüssel des aktuellen Satzes
r	relative Nummer des Satzes in der Ergebnismenge
s	Kurzzeile des Satzes (aus der .STL-Datei)

Das Hilfesystem der Windows-Programme

Es wird *nicht* das Microsoft-Hilfesystem bemüht, das die meisten Windows-Programme verwenden,. Zwar ist dieses sehr benutzerfreundlich, es ist aber für den Anwender eine BlackBox - man kann die Texte nicht verändern und keine eigenen einbringen. Das wäre gegenüber dem DOS-System ein Rückschritt gewesen, schon deshalb wurde ein eigenes Hilfesystem geschaffen. So kann nun jeder, der will, eigene Hilfetexte machen, und das auch noch in mehreren Sprachen. Ferner können in die Hilfetexte Flips eingebaut werden (also auch FLEX-Aufrufe), auch das ginge beim Microsoft-System nicht. Ein großes Angebot von allgemeinen Hilfetexten wird bereits mitgeliefert, einige auch in Englisch, bis hin zu ganzen Kapiteln aus dem Handbuch. (Kap. 0 und 10, Anh. A und B (Format). Geben Sie mal im Schreibfeld ein: **`h handbuch`**)
Zum Anzeigen der Hilfetexte dient das Anzeigefenster, wo normalerweise die Datensätze erscheinen. Drücken Sie auf F1 oder auf den Button mit dem Fragezeichen, dann sehen Sie schon, wie die Sache läuft.
Ist ein Hilfetext in der Anzeige, wirken links oben die beiden Knöpfe [←] / [→] anders: es werden die vorher aufgerufenen Hilfetexte durchgeblättert, ganz ähnlich wie bei bei einem Web-Browser.

Den nachfolgenden Text können Sie auch online abrufen: geben Sie im Schreibfeld ein **`h help`**
Zu den Hilfetexten gehört die Kopfdatei HELPHEAD.RTF. Diese muß unbedingt da sein, sonst erscheinen ASCII-Hilfetexte nicht (sondern nur RTF-Texte). Gesucht wird sie zuerst auf dem Daten-, dann auf dem Programmverzeichnis. D.h. es kann datenspezifische Versionen geben, die dann vorgezogen werden. Darin kann man die Schriftart festlegen, in der die ASCII-Dateien erscheinen sollen.
Tip: Jede Textdatei kann auch von Hand direkt ins Anzeigefenster geladen werden: man gibt im Schreibfeld ein:

```
h dateiname , z.B.  h doku  oder  h handbuch  oder  h a99.ini
```

und schon erscheint der Text. Man kann ihn bearbeiten und über das Menü "Datei | Anzeige speichern als..." wieder abspeichern. Wenn die Datei vom Typ .RTF ist, oder wenn das erste Zeichen ein \ ist (!), wird sie als RTF-Datei angezeigt, sonst wird sie als ASCII-Datei gezeigt. Beim Abspeichern kommt es ebenfalls auf den Typ an: wählt man etwas anderes als .RTF, wird nur der Text gespeichert, nicht die RTF-Befehle. Das kann man nutzen, wenn man einmal nur den Textinhalt extrahieren will. (Zur Bearbeitung von RTF-Hilfetexten mit eingebetteten Flips beachte man die Bemerkungen unter E.)
Aber: Wenn man RTF-Hilfetexte im Fenster **bearbeiten** will, unbedingt mit komplettem Namen aufrufen, also incl. .RTF, sonst verschwinden die Flips. Und man braucht Berechtigungsstufe 4 (`access=4` in der INI-Datei). Wenn hinterher ein Flip nicht funktioniert: 1. Prüfen ob Code 160 vor und hinter dem Text, 2. Umlaute müssen in der Flip-Zeile ASCII-Code sein, mit X-Editor nachbearbeiten (die Angaben mit "`\par ?...`" müssen außerdem am Zeilenanfang beginnen, s. unter C.).

`Standort der Hilfedateien: Lokal, DbDir, ProgDir\help, ProgDir`
In dieser Reihenfolge wird nach den Dateien gesucht, d.h. nutzerspezifische Versionen haben Vorrang.

Empfehlung: Nutzerspezifisch: `Startverzeichnis` (Lokal), Datenspezifisch: `Datenbankverzeichnis`,
Allgemein: `Programmverzeichnis\help`

A. Unterfenster-Hilfetexte
Sonderfall! Die Namen der zu den Unterfenstern gehörigen Hilfetexte beginnen alle mit HE bzw. HA und enden mit GER oder ENG (deutsche bzw. engl. Version). Dazwischen steht der eigentliche Name der Hilfeseite, der immer aus 3 Zeichen besteht.
Wird eine Datei HExxxGER nicht gefunden, sucht das Programm nach HExxx, also ohne Sprachkennung.
(HA steht für *alcarta*, HE für *a99*)
Wird sie immer noch nicht gefunden, wird noch nach HExxxGER.RTF gesucht, dies also erst zuletzt.

Folgende Texte gibt es für die Unterfenster: (ASCII-Text ohne RTF. Der 3buchstabige Name muß dann mit '_' beginnen.)
Diese Texte erscheinen jeweils dann, wenn F1 gedrückt wird, während das Unterfenster aktiv ist:

für beide Programme: (mit HA (für alcarta) bzw. HE (für a99))

```
..IX   Index
..BR   Kurzanzeige (Brief Display)
..CH   Suchbefehls-Menü
```

nur ***a99*** (d.h. nur HE...)

```
HE_AD   Formular-Editor
HE_GR   Globale Ersetzungen
```

Diese Texte erscheinen dann in einem eigenen Fenster, das erst mit [Enter] oder Klick auf [OK] beseitigt werden muß, bevor man weitermachen kann.

B. Hilfetexte für den EDITOR
Die Hilfetexte für das Editieren können von DOS übernommen werden (z.B. h20ger oder h2ger oder h20 oder h2). Sie erscheinen entweder im Anzeigefenster (wenn Cursor im Schreibfeld) oder im Help-Fenster (wenn gerade ein Formular benutzt wird). Ist kein geeigneter H-Text vorhanden für die aktuelle Kategorie, erscheint WRITEGER.RTF.
Diese Texte dürfen keine RTF-Texte sein.

C. Hilfetexte für die Hauptfunktionen
Festgelegt sind folgende Dateinamen, alle vom Typ .RTF (Angehängt wird jeweils die Sprachkennung. Es liegen die Versionen GER und ENG vor, es gibt also `START`**`GER`**`.RTF` und `START`**`ENG`**`.RTF` usw.)

START	:	Hilfeseite (gleich nach dem Start)
GENRL	:	Ersatz, wenn eine Hilfeseite zur Datenbank nicht existiert (Wichtigste Funktionen)
DOKU	:	Leitseite zur Dokumentation
BUTTN	:	Hilfe zu den Buttons und sonstigen Bedienelementen
EXPRT	:	Export-Menü
FIND	:	Zum Feld "Suchbefehl erste " (F1 auf der Eingabezeile)
FLEX	:	Zur FLEX-Makrosprache
KONFG	:	wenn die CFG-Kategorienliste angezeigt wird
MENU	:	Hilfe zu den Menüs
PRMPT	:	wenn die Abfrageliste angezeigt wird
RECRD	:	wenn die Kategorien des Datensatzes im Listenfeld stehen
RESRV	:	wenn der Hintergrundspeicher in der Anzeige steht
RESLT	:	Hilfe zu Ergebnismengen (wenn diese im Listenfeld stehen)
WRITE	:	Eingabefeld zum Eingeben/Korrigieren von Kategorien (nur a99)

Innerhalb eines Hilfetextes können Flips eingebettet sein. Wie sieht ein solcher Flip aus?
Zum Beispiel in `GENRLGER.RTF`, da steht drin:

```
Der Besen ist zum Löschen aller Ergebnismengen, Bookmarks etc.
...
```

(und ganz unten steht dann auf eigener Zeile:)

```
?Besen=h besen
```

oder aber:

```
\par ?Besen=h besen
```

Die zweite Form entsteht, wenn man den RTF-Text mit WORD oder WordPad speichert. Beides funktioniert. Wenn ein Flip nicht funktioniert, prüfe man die Datei nicht mit Word, sondern z.B. mit X-Editor. Die Zeichenfolge

`\par ?` muss am Zeilenanfang stehen! Nichts anderes davor oder dazwischen.
Hinter dem '?' muss exakt dieselbe Zeichenkette stehen wie zwischen den Begrenzungszeichen im RTF-Text (die Leerzeichen hinter \ul und \ul0 gehören zum RTF-Steuercode, nicht zum Text - so ist das bei RTF), also zwischen zwei Codes 160 oder zwischen [ ]. Umlaute und ß müssen ASCII-Codes sein.

Das Wort "Besen" im Text erscheint blau und unterstrichen. Zu diesem Wort gehört der Flip "h besen", der dann die Seite "besenger.rtf" aufruft, s.u. Solche Flips werden auf die #uY/#uZ-Variablen mit den Codes ab 129 gelegt.
Hinter dem '=' kann einer der Flip-Befehle stehen oder ein FLEX-Befehl, nicht nur h ...
z.B. `...=j Xabc`
das heißt: führe Sprungmarke #-X in den Anzeigeparametern aus (in #u1 steht dann "abc").
Meistens wird man aber mit x bzw. X einen FLEX angeben. Vgl. ORDA.RTF als Beispiel (Demo-Datenbank).
`Nochmal:` Wenn im blau unterstrichenen Teil ein Umlaut vorkommt, muss man nach dem Bearbeiten der Datei per Word oder WordPad nochmal dieselbe Datei per X-Editor aufrufen und unten die FLEX-Zeile ändern: dort steht z.B. `\'e4` statt ä. Man ersetze es durch ein ASCII-ä (daher wird der X-Editor empfohlen, nicht Notepad.)

D. Registerspezifische Hilfe
für jedes Register kann man eine Hilfeseite HA_InGER für *alcarta* anlegen. Diese sind vom Typ A, also nicht RTF, sondern simple ASCII-Texte. (n=1,2,3...) für *a99* lauten die Namen HE_InGER

E. Benutzergestaltete, freie RTF-Hilfeseiten
Die mächtigste der Möglichkeiten.
Dateiname ist frei bis zu 5 Buchstaben, dann GER oder ENG, Dateityp .RTF, z.B. flexGER.RTF.
Wenn man nicht mehrsprachig arbeiten will, kann man die Sprachkennung weglassen. Das Programm zieht die sprachspezifische Variante vor, d.h. es schaut zuerst nach flexGER.RTF, dann nach flex.RTF. (Damit können auch die eigenen Hilfeseiten mehrsprachig angelegt werden)
Schreiben kann man diese Dateien mit WinWord, speichern mit Dateityp .RTF. (Daher haben diese Dateien ANSI-Codierung!). Grafik und Tabellen kann man nicht einsetzen.

Schreiben und bearbeiten kann man die RTF-Dateien auch mit RTFEDIT.EXE, das man sich per FTP besorgen kann. Oder mit WordPad, wenn man dieses besitzt. Diese Programm erzeugen viel kompaktere Dateien als Word!
Wenn man FLEXe einbettet, muss man meistens die entstandene RTF-Datei noch mit der Hand nacharbeiten, z.B. mit dem X-Editor. Und zwar müssen die FLEX-Zeilen (die mit "`\par ?`" beginnen) die Sonderzeichen als ASCII- (nicht ANSI-) Zeichen enthalten, und RTF verdoppelt jeden \, macht also \\ draus - das muss man wieder ändern. Achtung: Pfadnamen innerhalb von FLEXen müssen einen doppelten Backslash haben, d.h. in der RTF-Datei sind es dann vier und man muss mit der Hand wieder zwei draus machen.
Wichtigstes Beispiel: *dbn*`GER.RTF` bzw. *dbn*`ENG.RTF` wobei *dbn* der Datenbankname ist, als allgemeine Hilfeseite für Ihre Datenbank. Standort für diese Dateien deshalb: `DbDir`
Vorlage: siehe Demo-Datenbank CAT: Die Datei `CATGER.RTF` erscheint bei dem Fragezeichen-Button (unten links)
Empfehlung: eine solche Datei für die eigene Datenbank machen. Andere Dateien können per Flip daran angeknüpft werden.
Bis auf Grafik und Tabellen kann man das meiste ausnutzen, was WinWord kann. Flips bindet man genauso ein wie oben, d.h. der Flip besteht im Text aus einer markierten Zeichenkette, der eigentliche Befehl steht auf eigener Zeile mit '?' am Anfang. Diese Zeilen werden natürlich nicht mit angezeigt. Wo diese Zeilen stehen, spielt daher keine Rolle.

Will man einen Hilfeseiten-Aufruf in die SATZanzeige einbauen, muss der Flip z.B. so aussehen:

```
#xyz p"Information zu diesem Datensatz!" e"!" =Y~
#xyz p"h name!" e"!" =Z~
#t{ t97 t74 160 "Information zu diesem Datensatz" 160 t76 t96 }
```

Der Text erscheint blau und unterstrichen, 160 = unsichtbares Zeichen). Der "name" muss ohne Sprachkennung und ohne .RTF angegeben werden, z.B
`... p"h hilfe!" e"!" ...` für die Datei `hilfeGER.RTF` bzw. `hilfeENG.RTF`

Zum Markieren kann man entweder [ ] oder den unsichtbaren Code 160 einsetzen. In WinWord macht man diesen Code mit Shift+Strg+Leertaste, sonst mit Alt+0160. In der RTF-Datei steht dann an der Stelle \~ . Der Fliptext muß also genau zwischen zwei solchen Markierungen stehen.
Auch in einer FLEX-Befehlskette können Hilfetexte aufgerufen werden, dazu dient der Befehl **help**.

Einbetten von Variablen in Hilfetexte
Innerhalb eines Hilfetextes kann jederzeit ein Ausdruck **%i** stehen, wobei **i** eine Ziffer oder ein Buchstabe ist. Wenn es dann die Variable **#uVi** gibt, wird deren Inhalt anstelle von %i eingesetzt. Dadurch kann man Satzinhalte oder anderes variables Material in Hilfesätze einbetten.
Soll etwa innerhalb eines Hilfetextes xyz.rtf der Titel (#20) des aktuellen Satzes erscheinen, jedoch ohne Zusatz, dann macht man es so: (siehe FLEX-Dokumentation)

```
x var #20(e" : ")\ins #uVt\help xyz
```

und in xyz.rtf muss an der betreffenden Stelle nur %t stehen. Die Variablen #uVi können auch innerhalb der Anzeigeparameter besetzt werden, das spielt keine Rolle.

Tip zum bequemen Bearbeiten der RTF-Hilfetexte:
Man gibt im Suchbefehls-Feld ausdrücklich ein: **`h name.rtf`** (also den Namen **mit** .rtf)
Dann kommt die Datei name.rtf zur Anzeige. Dort kann man sie bearbeiten, unten sieht man auch die Flip-Zeilen (!), und über das Menü "Datei | Anzeige speichern" speichert man den Text zurück.
Wenn man nur h name gibt, also ohne .rtf, werden die Flip-Zeilen nicht sichtbar - wie es im Normalbetrieb sein soll.

Sonderfall: **Einbetten von Phrasen in Hilfetexte (für Strg-Tasten)**
Wenn man eine Zeile der Gestalt

```
\par ?zzz=^pabc
```

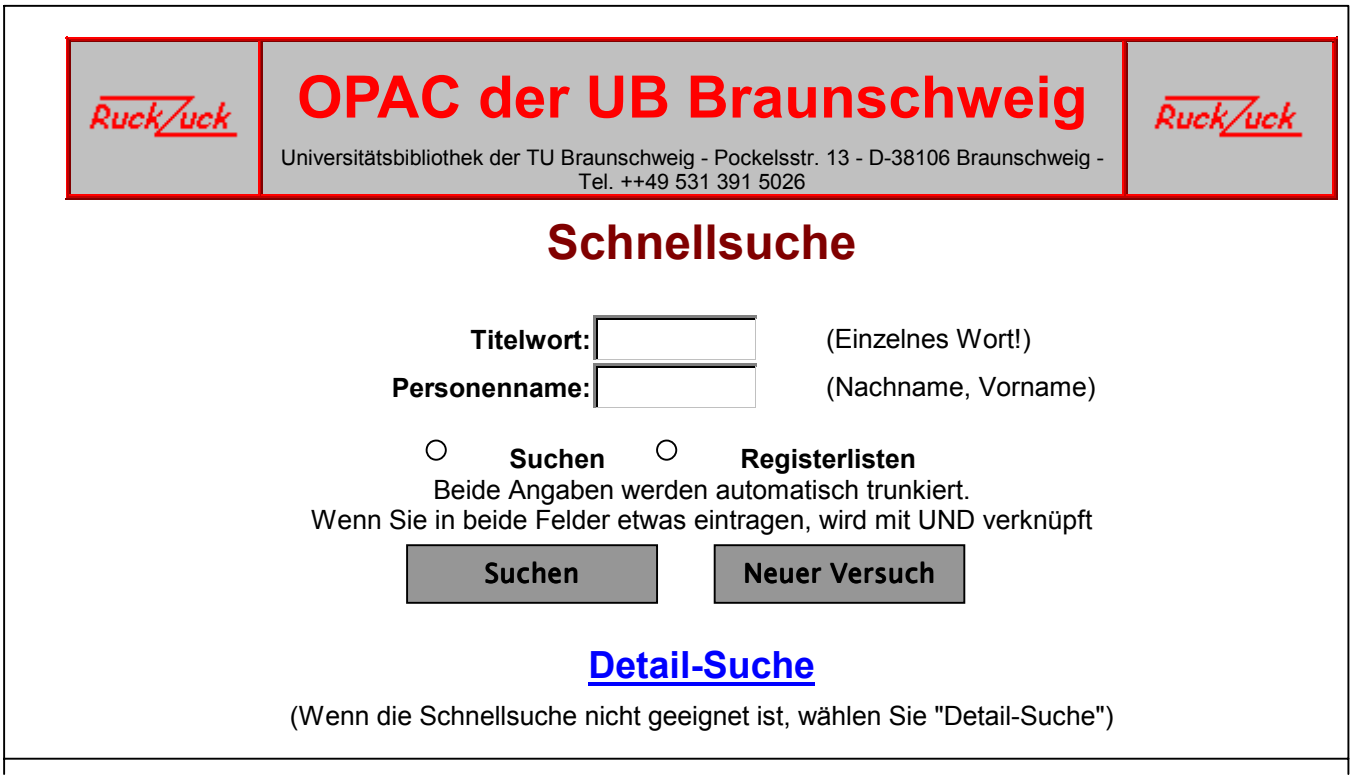
in einer RTF-Hilfedatei hat, wird beim Einlesen die Phrase Strg+p mit der Zeichenkette "abc" belegt. Wenn dabei der Text abc mit "x " oder "X " anfängt, hat man einen FLEX. Beispiele dafür findet man am Ende der Datei FLEXGER.RTF. (Die Zeichen `zzz` sind ohne Bedeutung!) Wenn innerhalb "abc" ein \ vorkommt, ist dieser hier ganz normal hinzuschreiben, nicht wie sonst in RTF doppelt. Denn die Zeilen mit "\par ?" am Anfang werden nicht als RTF-Zeilen ausgewertet.

F. Hilfetexte als Datensätze
Als weitere Möglichkeit kann man einen Datensatztyp schaffen, der Hilfetexte enthält. Das ist ausführlich im Beispiel der Datenbank "Theologischer Zeitschriften-Inhaltsdienst" (Tübingen, Dr. Weisweiler) durchgeführt worden. In ca. 70 solchen Hilfesätzen (jeder in deutscher und engl. Version) sind alle Hinweise zu den DOS- und Windows-Funktionen der Datenbank untergebracht. Geeignete Parametrierung ermöglicht natürlich auch hier, neben der Indexierung der Hilfesätze, dass man Flips einbaut, die den Nutzer dann zu anderen Hilfesätzen, zu Registerabschnitten oder zu bestimmten Datensätzen hinführen. Es ist nur eine Frage der Zeit und Phantasie, dieses Konzept beliebig auszubauen. Einmal parametriert, spart man sich dabei die Kleinarbeit, die in die RTF-Dateien manchmal zu investieren ist. (Die Datei HELPHEAD.RTF wird für diese Zwecke nicht gebraucht, sondern DISPHEAD.RTF, wie für andere Datensätze auch.)

RuckZuck

Web-Anbindung auf die Schnelle

Nicht selten fehlt die Zeit, das Geld, die Erfahrung - oder alles zusammen - um eine Web-Schnittstelle nach allen Regeln der Kunst für die eigene Datenbank zu machen. Nicht immer ist der Aufwand notwendig, den man mit ***avanti*** und mit ACWWW25 oder "populo" treiben muss. Deshalb gibt es eine Schnittstelle für Schnellschüsse: ***RuckZuck***. Diese Alternative kommt sogar ohne Skriptsprache aus (Perl, Python, Javascript), nur ganz normales HTML ist nötig. Es wird hiermit der Versuch gemacht, ein möglichst vielseitiges Repertoire von Funktionen (einschl. Schreibzugriff) vorgefertigt bereitzustellen, wobei die Anforderungen an das System und die Notwendigkeit von Anpassungen auf ein Minimum reduziert sind. Lokalen Anpassungen sind keine Grenzen gesetzt, vor allem mit JavaScript, an der Oberfläche ist nicht viel ***allegro***-Kenntnis notwendig. Wenn keine zusätzlichen Anforderungen auftreten, kann man hiermit eine Datenbank in einer Stunde am Netz haben.
So kann ein Startbild aussehen, wenn man eine mit *RuckZuck* angebundene Datenbank anwählt:



Was ist *RuckZuck*?

Sie haben eine *allegro*-Datenbank (oder eine andere, aus der man eine machen könnte), **die ins Inter- oder Intranet soll?**
Nein: → **Ende**
Ja: **Welche dieser Aussagen gelten für Sie:**

- Sie haben zu wenig Geld
- Sie haben zu wenig Zeit
- Sie haben zu wenig Ahnung von der WWW-Programmierung, insbes. von Perl, Python etc.
- Ihre Datenbank soll nur für begrenzte Zeit oder einen begrenzten Kreis verfügbar gemacht werden, vielleicht gar nur für eine einzelne Person am anderen Ende des Internet : d.h. hoher Aufwand lohnt nicht.

Nichts davon trifft zu: → Ende.
Denn dann verwenden Sie "avanti" auf einer UNIX- oder Linux-Anlage und Sie richten mit Hilfsmitteln wie z.B. Thomas Bergers "populo"-Paket einen maßgeschneiderten Qualitätsdienst ein, mit hoher Betriebssicherheit, der monatelang ohne Aufsicht und ohne Eingriff laufen kann. Damit verglichen ist ***RuckZuck*** vielleicht eine 90%-Lösung - doch manchmal braucht man ja gar nicht mehr!
Wenn eine oder mehr dieser Aussagen für Sie zutreffen, ist die neue RuckZuck-Methode eine Überlegung wert. Oder auch dann, wenn schnell ein Provisorium gebraucht wird, damit man die Dauerlösung dann in Ruhe ausarbeiten kann.

Und was ist *RuckZuck* nun wirklich?
Es ist ein **Fertigpaket** für das Bereitstellen einer ***allegro***-Datenbank im Inter- oder Intranet. Es kommt **ohne Skriptsprache** (wie Perl, Python oder Javascript) aus. Programmiert wird mit den Mitteln der Exportsprache und der FLEX-Sprache und mit ganz gewöhnlichem HTML. Das frei verfügbare Paket KANN durch weitere Module ergänzt und natürlich auch lokal angepasst und ausgebaut werden (vor allem mit Javascript); nur an ganz wenigen Punkten MUSS man aber wirklich eingreifen.
Es sollte mit jedem Kategorienschema, also mit jeder ***allegro***-Datenbank funktionieren. Es enthält nur die wichtigsten Funktionen, die jeder braucht, aber sorgfältig durchdacht und offen für Erweiterungen. Die Standarddateien werden ohne grafische Elemente geliefert, die Einbindung eigener Grafiken, insbes. Logos oder GIF-Dateien für Links wie "Nächste Seite", "Neue Suche" usw., ist für HTML-Kenner sehr einfach. Neudeutsch: man kann leicht seine eigene "Corporate Identity" integrieren.
RuckZuck besteht aus ganz wenigen Dateien und bleibt damit sehr übersichtlich. Alle Dateien sind voll kommentiert, die Stellen für notwendige Eingriffe sind markiert, daher braucht man nicht alles zu verstehen, was da abläuft.
Was man braucht:

1. Rechner mit Win'NT (Workstation reicht) oder sogar nur Win'95/98 . Er muss am Internet hängen, braucht aber keinen eigenen Namen, nur eine IP-Nummer. Eine solche hat jeder PC, der z.B. in einem Hochschulnetz steht.
2. Das Programm ***a99***
3. Das neue Hilfsprogramm `cgi.exe`
4. Die Datei opac.htm und das Paket ruckzuck.lzh, dieses enthält Dateien, die man für jede Datenbank auf ein eigenes Verzeichnis kopieren muss. Alternativ: das Paket cool.lzh mit etwas weiter ausgebauten Funktionen.
5. Ein Webserver, z.B. den kostenlosen Xitami (dieser lässt sich in 5 Minuten installieren *und* in Betrieb setzen).

Alles muss auf demselben PC installiert werden! Webserver und a99 können nicht über CGI.EXE kommunizieren, wenn das nicht der Fall ist. Die Datenbank kann aber im LAN auf einem anderen Fileserver (z.B. auch Novell) liegen. Bereitgestellt wurde ab 30.3.2000 mit **`ruckzuck.exe`** ein Paket, das außer Xitami alles enthält, was gebraucht wird, und das nach der Installation (per "InstallShield", wie unter Windows üblich) sofort lauffähig ist. Damit kann eine Datenbank von bis zu 80.000 Sätzen aufgebaut und betrieben werden. (Mit dem Vollprogramm sehr viel mehr.) Die darin enthaltene Demo-Datenbank ist ein Auszug aus dem CoOL-System. Dieses ist sozusagen die Link-Sammlung der UB Braunschweig.
Für UNIX oder Linux existiert RuckZuck nicht, dafür empfiehlt sich der ***avanti***-Server.

So startet man *RuckZuck*

1. Webserver starten. Auf NT als Konsolprogramm, nicht als Systemdienst starten.
   Standard: auf C:\XITAMI den Befehl **`xiwin32`** geben

2. ***a99*** starten
   Standard: auf c:\cool den Befehl **`a99 cool`** geben

Über die präparierte HTML-Eingangsseite (OPAC.HTM mit lokaler Anpassung) kann jetzt der Zugriff losgehen.
An der Datenbank kann von anderen Fenstern aus normal gearbeitet werden, entweder mit den DOS-Programmen oder *a99*.
Das Fenster von *a99* kann man u.a. benutzen, um sich die abgerufenen Ergebnismengen anzuschauen (Alt+e, dann sieht man die Liste).
Es entsteht ein Protokoll RZ.LOG auf dem CGI-Verzeichnis des Web-Servers. Dieses kann man jederzeit gefahrlos löschen.

Wie läuft es?

Funktionsweise der RuckZuck-Schnittstelle

Als Datenbankserver dient a99.exe, das Windows-Hauptprogramm des ***allegro***-Systems, die Anfragen werden ihm als externe FLEXe übergeben. (Ein FLEX ist eine Art Makro, das man sowohl interaktiv in *a99* ausführen, aber auch von außen dem Programm übergeben kann, gewissermaßen zur Fernsteuerung.)

So wird installiert, wenn man eine eigene Datenbank zugänglich machen will:

1. In den eigenen Indexparametern (Standard: CAT.API) muss man symbolische Namen für die Register und die Restriktionen haben (z.B. PER für das Personenregister etc.) siehe Handbuch S. 176.

2. Die Datei OPAC.HTM kopiert man auf das Dokumentenverzeichnis des Webservers. Bei Xitami ist das normalerweise c:\xitami\webpages (anderer Name ist möglich). In OPAC.HTM muss man evtl. an wenigen Stellen eingreifen. Zweimal muss der Verzeichnisname eingetragen werden, wo *a99* läuft (siehe 3., Default ist C:\OPAC.)
   Die URL dieser Datei teilt man dann den betroffenen Personen mit. Diese Datei ist der Zugang und zugleich so etwas wie die INI-Datei für Ihren Web-Katalog. Daher enthält sie den Pfad, wo *a99* läuft.

3. Das Programm CGI.EXE muss auf das CGI-Verzeichnis des Web-Servers. Bei einer Standard-Installation von Xitami ist das ...\xitami\cgi-bin. CGI.EXE wird bei Bedarf vom Webserver aufgerufen, wie jedes normale CGI-Programm.

4. Man legt ein Verzeichnis für die Schnittstelle an, z.B. C:\OPAC (günstig ist ein nicht zu langer Name, denn er muss bei jeder Transaktion mit durchgereicht werden). Der Name dieses Verzeichnisses muss in OPAC.HTM stehen (siehe 2.). Wenn Sie es einmal verlagern: nur OPAC.HTM entsprechend ändern. Auf C:\OPAC entpackt man ruckzuck.lzh: die FLEXe, Parameter und HTM-Dateistücke, aus denen die Schnittstelle besteht. Hier muss auch eine Datei xyz.ini angelegt werden. Darin stehen Name und Pfad der Datenbank etc. Mit **`a99 xyz`** startet man *a99* dann auf diesem Verzeichnis. Man stellt es dann so ein, dass irgendein Titelsatz in der Anzeige ist. Nun wartet es auf Arbeit...
   In den .FLX-Dateien und HTM-Teilen sind Stellen für Anpassungen mit XXX markiert.

5. Für weitere Datenbanken legt man weitere Verzeichnisse an, die jeweils eine induviduelle .INI für den Start von *a99* brauchen. D.h. es muss für jede Datenbank das Programm *a99* auf dem zugehörigen Schnittstellenverzeichnis gestartet werden. Im Hinblick auf Systemressourcen ist das unkritisch: jedes laufende *a99* belegt höchstens 2.5MB Arbeitsspeicher und braucht nur dann Rechenleistung, wenn gerade eine Anfrage bearbeitet wird. Ändernde Eingriffe sind voraussichtlich nur nötig in der Datei FIND-S.FLX (kommentiert), die aus OPAC.HTM aufgerufen wird. Außerdem in GETEDIT.FLX und WRITE.FLX sowie D-EDIT.APR, wenn man Schreibzugriffe erlauben will. Für die Einstellung der Sortierstelle auch FIND-D.FLX.

6. In der INI-Datei muß stehen: `PrintParameter=D-H0`. Wenn *a99* nicht auf C:\ALLEGRO installiert ist, muß man die Eintragung `ProgDir` entsprechend ändern.

7. Wer nicht mit dem Standardschema A.CFG arbeitet: Für die eigene Datenbank muss eine Datei `D-HTML.cPR` angelegt werden, abgeleitet von D-1.cPR (wobei c der eigene Konfigurationsbuchstabe ist, z.B. **P** für Pica), sowie eine `D-EDIT.cPR`, wenn man Schreibzugriff anbieten will. Die Datei `D-H0.APR` kann unverändert auf `D-H0.cPR` kopiert werden, sie wird für einige Hilfsfunktionen gebraucht.

   Schnelle Lösung:
   Man kopiert hilfsweise `D-H0.APR` zusätzlich auf `D-HTML.cPR`. Dann bekommt man nur eine Anzeige im Internformat. Außerdem: P-HTM.APT auf P-HTM.cPT kopieren.
   Aufwendigere Lösung:
   Man erstellt `D-HTML.cPR` aus der in der Regel schon vorhandenen `D-1.cPR`, indem man diese Einstellungen darin vornimmt:

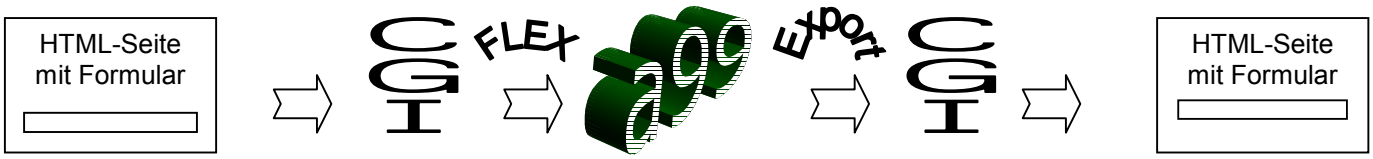
```
zl=80
ze=10
zm=0
zb=" "
ab=""
as=""
tasciansi          diese Zeile am besten ans Ende der Datei! (Codetabelle ASCII - ANSI laden)
```

   Wenn man keine ASCIANSI.cPT hat, kopiert man ASCIANSI.APT auf diesen Namen. Im allgemeinen wird es dann schon klappen. Das Erscheinungsbild der Datensätze lässt sich durch Bearbeitung der `D-HTML.cPR` noch beliebig verbessern. In die Hilfsparameter `D-H0.APR` wird man normalerweise nicht eingreifen müssen.

Die Funktionsweise

- Für die Übergabe von Nutzerdaten werden ganz normale HTML-Formulare verwendet. Diese werden an ein kleines CGI-Programm übergeben, CGI.EXE. Es ist in C geschrieben und daher viel schneller als z.B. Perl.
- CGI.EXE wandelt die Formulardaten um, verbindet sie mit einem vorgefertigten FLEX und übergibt diesen an *a99*.
- Der FLEX veranlasst *a99*, eine HTML-Daten zu exportieren, die durch CGI unmittelbar an den Nutzer übergeben wird. Auch hier können vorgefertigte Teile zum Einsatz kommen, z.B. HTML-Kopfdateien. Die exportierte HTML-Datei enthält immer ein Formular oder Links, und damit geht dann der Vorgang von vorn los.

So sieht das Ganze grafisch aus:



Es gibt eine Anzahl von vorgefertigten FLEXen mit wichtigen Grundfunktionen, die auf diesem Wege aktiviert werden können. Diese sind jeweils kommentiert. Insbesondere ist markiert, wo man Veränderungen vornehmen kann oder muss. Man braucht nur eine einzige feste HTML-Seite: die für den Einstieg, z.B. COOL.HTM. Alle anderen Texte werden immer ad-hoc von den FLEXen erstellt. Ein FLEX kann unterschiedliche Formulare erzeugen, z.B. **FIND-D.FLX**

Übersicht:
Was sieht/macht der Nutzer → was wird dann ausgelöst

START: z.B.
COOL.HTM → FIND-S.FLX Ausführung einer Suche
Ausgabe: Ergebnisliste oder Registerausschnitt
oder → DETAIL.FLX produziert Formular für kombinierte Suche
Ausgabe: Das **Eingabeformular**

Eingabeformular für kombinierte Suche
→ FIND-D.FLX Neue Suche ausführen
Ausgabe: **Ergebnisliste** oder **Registerausschnitt**

Ergebnisliste
→ GETREC.FLX holt einen einzelnen Satz
Ausgabe: **Satzanzeige**
→ FIND-D.FLX veränderte Suche ausführen - andere **Ergebnisliste**
→ DETAIL.FLX **Eingabeformular** zum Start einer neuen Suche

Registerausschnitt
→ FIND-D.FLX Ausgabe: **Ergebnisliste** oder **Satzanzeige**
→ PAGE.FLX eine Seite vor oder zurück
Ausgabe: neuer **Registerausschnitt**
→ DETAIL.FLX **Eingabeformular** zum Start einer Neuen Suche

Satzanzeige
→ DETAIL.FLX **Eingabeformular** zum Start einer Neuen Suche
→ GETEDIT.FLX Satz im **Bearbeitungsformular** bereitstellen

Bearbeitungsformular
→ WRITE.FLX Satz speichern, entw. als Korrektur oder Neusatz, **Satzanzeige**

Eingreifen muss man bei eigenen (nicht dem Standard entsprechenden) Datenbanken in die unterstrichenen Dateien, denn darin gibt es datenbankabhängige Teile. Man findet die Stellen leicht, denn es steht immer XXX und ein Kommentar darüber.

Liste der FLEXe

Diese FLEXe werden als Grundpaket bereitgestellt. Sie erzeugen jeweils einen HTML-Text als Ausgabe. Sie benutzen jeweils eine (meistens) gleichnamige .HTM-Datei als Kopf. Hier kann man lokale Design-Elemente einbringen, z.B. ein eigenes Logo statt `ruckzuck.gif` (man kopiert das eigene auf diesen Namen, dann hat man weiter nichts zu tun). und Links zu eigenen Seiten. In diesen Kopfdateien kann man aber auch JavaScript-Funktionen unterbringen. In die FLEXe eingebunden sind die Seiten mit dem Befehl **write F**`filename.` Die HTM-Köpfe enthalten kein "<head>...</head>", denn dieser Abschnitt wird schon von CGI.EXE erstellt und enthält nur einen <title> mit der internen Nummer des Jobs.

In der nachfolgenden Liste der FLEXe mit zugehörigen htm-Kopfdateien werden noch einige Angaben zu den Funktionen dieser Dateien gemacht:

FIND-S.FLX wird von der Startseite `OPAC.HTM` gestartet
`find.htm`

DETAIL.FLX kann von `OPAC.HTM` (Link "Detail-Suche") aufgerufen werden, und durch den Link
`detail.htm` `"Neue Suche"` von anderen Stellen. Holt sich die Registernamen für die Auswahl automatisch

FIND-D.FLX Start aus `DETAIL.FLX` und aus `PAGE.FLX`
`find.htm` zeigt Erg.Menge (wenn mehr als 1 Satz), sonst sofort den Satz, ODER einen Registerabschnitt

PAGE.FLX Ruft sich selber auf mit `[Seite höher] / [Seite weiter]`
`page.htm` Zeigt einen Registerabschnitt, ruft FIND-D.FLX auf, wenn eine Zeile angeklickt wird.

GETREC.FLX Aufruf aus `FIND-D.FLX,` holt und zeigt einen einzelnen Datensatz
`getrec.htm` Startet `GETEDIT.FLX,` wenn gewünscht (Link Bearbeiten)

GETEDIT.FLX Start aus `GETREC.FLX` (Link "Bearbeiten")
`getedit.htm` Legt einen Satz zum Bearbeiten vor. Mit **JavaScript-Beispiel** zur Eingabekontrolle

WRITE.FLX `Start aus GETEDIT.FLX` (Button "Speichern")
`getrec.htm` Schreibt den Datensatz zurück, wahlweise als neuen oder korrigierten Satz

Weitere Hinweise für CGI-kundige Leser

Das Programm CGI.EXE kann sowohl POST wie auch GET-Aufträge entgegennehmen, d.h. man kann Aufträge über Formulare oder über Links erzeugen.
In einem Auftrag müssen bestimmte Dinge enthalten sein:

VuwD=Name des Verz., wo *a99* läuft ("working Directory"; da liegt nicht die Datenbank, nur das RuckZuck-Paket!)

flex=Name eines FLEXes (der auf dem "working Directory" liegen muss)

Vuxy=... beliebig viele #uxy-Variablen; werden angelegt, bevor der FLEX anläuft

Optional können folgende Variablen auftreten (nur bei Schreibaufträgen sinnvoll, s. WRITE.FLX):

Vnnn=... beliebig viele Kategorien #nnn , kommen direkt in den Aufnahmespeicher

Rnum=... interne Nummer des zu bearbeitenden Satzes. Dann wird dieser Satz zuerst geladen, bevor der FLEX anläuft und bevor die V-Befehle ausgeführt werden, d.h. die in den anderen Variablen mitgelieferten Kategorien werden in den Satz eingeordnet, bevor der eigentliche FLEX beginnt.

Text=... Inhalt eines TEXTAREA-Feldes in einem Eingabeformular; Verwendung in GETEDIT.FLX, siehe dort.

Die Variablen Vuxy werden also dem zu startenden FLEX überreicht, d.h. CGI.EXE fabriziert einen neuen FLEX (und löscht ihn hinterher sofort wieder) Dieser temporäre FLEX enthält am Anfang die Setzung der #uxy-Variablen etc. (s.o.) und am Ende deren Löschung. Man verwende im FLEX daher nicht den Befehl "end", sondern statt dessen "jump exit", denn die Sprungmarke `:exit` wird ebenfalls erzeugt. D.h. CGI erstellt einen "Sandwich"-FLEX aus mehreren Teilen:

```
Laden des Satzes (wenn Rnum gegeben)
Setzungen der #uxy-Variablen
Inhalt von Text (d.h. Einfügen der darin enth. Kategorien in den Datensatz)
der FLEX aus der CGI-Variablen flex=...
:exit
Löschungen der #uxy-Variablen
```

Der Name des temporären FLEX enthält u.a. die Uhrzeit in Sekunden, daher ist keine Kollision zweier Aufträge zu befürchten.
CGI sendet diesen Namen an *a99* und wartet, bis die Ausgabedatei fertig ist (d.h. von *a99* das Signal zurückkommt, es sei alles erledigt). Keines der Programme muss also ständig die Platte überwachen oder dergleichen, sondern die Kommunikation geschieht über Windows-Systemmeldungen.
Als Exportparameter wird per default D-HTML.cPR benutzt. Ansonsten ist der Output des FLEXes genau das, was CGI.EXE dann dem Web-Server zurückreicht, d.h. man muss in seinem FLEX den kompletten HTML-Ausgabetext fabrizieren. Das kann abwechselnd durch "write"- und "download"-Befehle passieren. Bei "write" hat man ja die Möglichkeit, ganze Fertigteile, d.h. Dateien "name.htm", mit **`wri Fname.htm`** auszugeben.
Übrigens kann auf dem Rechner, auf dem der Webserver, *a99* und CGI laufen, durchaus ganz normal gearbeitet werden. Z.B. kann man darauf auch Netscape starten und den Katalog lokal selber benutzen, indem man die Adresse
`http://127.0.0.1/allegro/opac.htm` aufruft.

Tips
Sollen einige Nutzer Schreibrechte haben, andere aber nicht: zwei verschiedene Eingangsseiten und zwei Startverzeichnisse anlegen, *a99* zweimal starten (auf jedem der Verzeichnisse). In der INI-Datei für die Nutzer ohne Schreibrecht kann man zur Sicherheit eintragen: access=0. Aus den Dateien GETREC.FLX und FIND-D.FLX (auf dem Verzeichnis ohne Schreibrecht) nimmt man die kommentierten Zeilen heraus, die den Link "Bearbeiten" erzeugen.
Den Zugang zu der Eingangsseite mit Schreibrecht kann man zusätzlich durch Passwort absichern.
Auf einem weiteren Verzeichnis kann man ein drittes *a99* starten, um damit ohne Störung durch die Web-Zugriffe selber an der Datenbank arbeiten zu können. Hier sollte dann `access=4` in der INI-Datei stehen, damit man alles machen kann, und `exflex=0` oder `exflex=2`, damit keine störenden Meldungen kommen.

Laden eigener Daten
Sollen eigene Daten in die Datenbank eingemischt werden, muss man eine Datei EXTERN.DAT in folgender Form bereitstellen (und zwar auf dem Verzeichnis, wo man *a99* gestartet hat):

```
#00  IdNr  <optional>
#0c  Rubrik (z.B. 015 für E-Journal; s. Register 7 unter F)
#20  Titel
#8e  URL
#30  Sachgruppe (z.B. ma für Mathematik; s. Register 7)
#40  Verfassername (falls einer existiert)
#61  Körperschaftsname
#88  ISSN
#98  Kommentar
```

(Das sind die relevanten Kategorien für CoOL-Daten.)
Zeichencodierung: DOS-ASCII.
Weitere Kategorien: siehe "allegro"-Formatdokumentation (http://www.allegro-c.de/allegro/format/).
EXTERN.DAT kann beliebig viele Datensätze enthalten, zwischen zwei Sätzen muss immer eine Leerzeile sein
Das Einlesen geht dann so: In *a99* wählt man den Menüpunkt "Datei | Externe Ergebnismenge laden", dann wird die Datei EXTERN.DAT zunächst in einer Kurzübersicht angezeigt. Wenn alles richtig aussieht: "Datei | Offline-Datei -> Datenbank". Lange Datenfelder, z.B. #98, können umgebrochen sein, dann muss aber jede Fortsetzungszeile mit einem Leerzeichen anfangen.

Hinweis für *allegro*-kundige Leser

Die Parameterdatei `D-H0.APR` besorgt einige Standardaufgaben wie das Produzieren von Registerauszügen und SELECT-Listen für die Registerauswahl. Sie wird über den neuen Befehl `Download` (mit großem D) benutzt. Dieser verwendet zum Exportieren nicht die `ExportParameter`, sondern die `PrintParameter` - die man als solche für RuckZuck nicht braucht. In der INI-Datei muss stehen **`PrintParameter=d-h0`**. So wird vermieden, dass man die Anzeigeparameter D-HTML.APR auch noch mit diesen Sonderaufgaben befrachten bzw. dass man in vorhandene Anzeigeparameter, die man für "avanti" schon gemacht hat, die etwas komplizierten Hilfsprozeduren auch noch einbauen muss. Das spart u.U. viel Zeit. Für eine andere CFG kann man D-H0.APR einfach kopieren, sie ist CFG-unabhängig.
Ausnutzen kann man in eigenen FLEXen evtl. zwei weitere Nutzervariablen: `#urH` und `#urA`. Diese enthalten jeweils den Namen des Host bzw. die Adresse, von der aus der FLEX gekommen ist.
allegro-Anwender wissen: fast nichts ist unverrückbar festgeschraubt, fast alles lässt sich irgendwie variieren oder die Entwicklungsabteilung macht, wenn begründete Vorschläge kommen, eine Programmverbesserung. Das gilt auch für RuckZuck. Schreiben Sie uns (allegro@tu-bs.de)!

alcarta im Netz oder auf CD

Die Windows-Programme brauchen ein Startverzeichnis, auf dem sie Schreibberechtigung haben, denn sie legen Hilfsdateien an, z.B. für die Ergebnismengen. Daher kann man ***alcarta*** nicht unmittelbar auf der CD starten oder auf einem Netzlaufwerk, wo man kein Schreibrecht hat. Es gibt zwei Möglichkeiten, diese Situation zu meistern:

1. Man startet auf der lokalen Platte, z.B. auf `C:\TEMP`
2. Man schreibt in die INI-Datei einen Pfadnamen, der für die Hilfsdateien zu nutzen ist. Das geht mit dem Befehl
   `DbAux`=*verzeichnis* z.B. `DbAux=C:\TEMP`

Das Programm alcarta.exe auf dem Verzeichnis \ALLEGRO der CD-ROM wählt automatisch das TEMP-Verzeichnis des PC. Das werden die zukünftigen Versionen von a99 und alcarta generell so machen.

allegro classico

"Abbohrung"
In der vorigen Nummer wurde dargestellt, wie man die früher gültige Dateigröße von 16 MB (.cLD-Dateien) überschreiten kann. Eine Schlüsselrolle spielt dabei der zweite Byte der .TBL-Datei. Dort steht im Normalfall der Wert 3. Im Falle einer "Aufbohrung" steht in den Indexparametern z.B. ein Befehl `ii=2`. Die Dateigröße ist dann maximal 2x16 = 32 MB. Das besagte .TBL-Byte muss dann den Wert 4 haben. Bei vorhandenen Datenbanken, also ohne ii-Wert in den Indexparametern, muss für die neuen Programme unbedingt der Wert 3 in der .TBL stehen. Dort kann, von früher her, höchstens dann etwas anderes stehen, wenn man mit Sniffer die Datenbank "behandelt" hat oder wenn man, untypisch, mit UPDATE die Datenbank einmal neu aufgebaut hatte. Dann steht in der TBL fälschlich der Wert 0 oder einer höher als 3. Es kommt dann zu der Fehlermeldung **`wrong recn ...`** und es wird kein Datensatz oder der falsche angezeigt. Zum Glück gibt es Abhilfe:
Entweder (so geht's am schnellsten) mit einem HEX-Editor wie HE.EXE den Wert auf 3 stellen, oder aber (auch das geht schnell) per ***CockPit*** die .TBL-Datei erneuern über das Menü "Routinen | Organisieren | Satztabelle erneuern". (Die .TBL vorher löschen, um ganz sicher zu gehen.) Dann kommt die Sicherheits-Frage "Wieviele Sätze enthält die Datenbank?". Geben Sie dann eine Zahl ein, die etwas größer ist, wenn Sie die wirkliche Zahl nicht wissen. Also etwa 10000, wenn es vielleicht um die 9000 oder ein paar mehr sind, aber sicher nicht mehr als 10000. Es macht auch nichts, wenn Sie dann 200000 angeben, die Zahl sollte nur größer als die wirkliche, aber nicht abenteuerlich groß.

INDEX - QRIX

`fatal error 233`
Dieser Fehler passierte bisher immer wieder mal, wenn am Ende nur noch genau 10 Zwischendateien (`ii1 - ii10`) übrig waren. Zwar kann man die Situation dann noch retten, indem man den QRIX-Befehl so startet, wie er am Ende der Datei PROTOK zu sehen ist. Der Fehler ist also gar nicht fatal. Jedoch kommt man darauf nicht sofort.
Da sich das Problem leider nicht ganz eliminieren ließ, wurde der Vorgang jetzt so eingerichtet, dass der QRIX-Befehl am Ende automatisch nochmal gegeben wird, wenn noch eine Datei `ii1` vorhanden ist. Dazu wurden INDEX.EXE und ACP.EXE verbessert. Die Situation tritt jetzt nicht mehr auf.

Puffergröße
INDEX arbeitet intern jetzt mit einem Sicherheitspuffer von 8000 Byte. Ist diese Grenze erreicht, werden die angesammelten Schlüssel vorsortiert und in die nächste ii-Datei geschrieben. Meistens sind 8000 viel zu viel, d.h. die ii-Pakete werden kleiner als sie sein dürften, und folglich wird ihre Anzahl größer, als sie sein müsste. Bei großen Datenmengen kann das für die Gesamtlaufzeit denn doch was ausmachen.
Daher kann man jetzt INDEX mit der neuen Option **`-B`** eine neue Größe für den Sicherheitspuffer zuweisen, z.B. `-B4000`, wenn man sicher ist, dass kein Satz so groß ist, dass er mehr als 4000 Byte Schlüssel erzeugt. Am Ende von PROTOK kann man sehen, was beim letzten Durchlauf die maximale Anzahl Schlüssel eines Satzes (und welches Satzes) war. Sind das z.B. 250, und ist die durchschnittliche Schlüssellänge ca.12 Byte, kann man `-B3500` setzen, das dürfte genügend sicher sein.
CockPit setzt -B nicht, d.h. man muss sich eine eigene Batchdatei für die Indexierung machen, wenn man dies nutzen will. *Tip:* Man starte das ***CockPit*** mit acp statt cp und löse "Organisieren | Index erneuern" aus. Die dann entstehende CCC.BAT braucht man nur entsprechend zu ergänzen, dann hat man die Batchdatei.

UPDATE

Nachladen
Das Nachladen von Sätzen sollte in UPDATE, und zwar in den Index-Parametern, genauso funktionieren wie in PRESTO. Das war nicht der Fall, es funktionierte meistens nicht. Das Problem wurde behoben.
Doppeleinträge
Es konnte bei sehr starker Belastung (zwei oder mehr UPDATEs gleichzeitig) dazu kommen, dass Identnummern doppelt vergeben wurden. Die Vorgänge wurden intern so abgesichert, dass dies nun nicht mehr passieren kann.

E-Mail-Diskussionsliste

Man schließt sich den ca. 280 Lesern der *allegro*-Liste an, indem man an die Adresse **`maiser@buch.biblio.etc.tu-bs.de`** eine Botschaft mit nur einer Zeile sendet: **`subscribe allegro`** . Man wird dann sofort eingetragen und erhält eine Mitteilung mit weiteren Informationen, insbesondere, wie man sich wieder abmeldet. Es gibt auch ein Archiv der Liste.

Frühere Ausgaben, aktuelle Programme etc.: `ftp://134.169.20.1` oder `http://www.allegro-c.de/allegro`