CBASE

ein von dBase II/III inspiriertes Datenbankprogramm

für

NKC 68000 (FPGA TangNano)

Stand: 01/2026

Inhaltsverzeichnis

Einführung

Was ist CBASE?

CBASE ist ein freies, eigenständig entwickeltes Datenbanksystem für den NKC 68000, inspiriert von den klassischen dBase-Systemen der 1980er Jahre (dBase II / III).

Ziel des Projekts ist es, die Funktionsweise früher datenbankbasierter Anwendungen auf Retro-Hardware nachvollziehbar, erlernbar und erweiterbar zu machen ? nicht die exakte Nachbildung oder Emulation eines kommerziellen Produkts.

Abgrenzung

CBASE ist keine Portierung, Emulation oder Rekonstruktion von dBase II, dBase III oder verwandter kommerzieller Software.

- Es wurde kein Original-Quellcode verwendet
- Es wurden keine Binärformate oder Routinen übernommen

Die Entwicklung erfolgte ausschließlich auf Basis:
- öffentlich dokumentierter Dateiformate (DBF)
- historischer Literatur
- eigener Analyse
- praktischer Erfahrung und Erinnerung an frühere Systeme

CBASE ist eine eigenständige Neuentwicklung, die sich konzeptionell an der Arbeitsweise historischer Datenbanksysteme orientiert.

Lernsystem & Community-Projekt

CBASE wurde als offenes Lern- und Experimentiersystem entwickelt.
Der vollständige Quelltext wird gemeinsam mit dem ausführbaren Programm bereitgestellt, um:
- den Aufbau klassischer Datenbanksysteme zu verstehen
- eigene Erweiterungen zu entwickeln
- Funktionen zu verändern oder wiederzuverwenden
- Retro-Softwareentwicklung auf dem NKC 68000 zu fördern

Das Projekt richtet sich ausdrücklich an Entwickler, Bastler und Retro-Computer-Interessierte.

Weiterverwendung des Codes

Die Nutzung, Modifikation und Weiterverwendung des Quelltexts im Rahmen nicht-kommerzieller Projekte ist ausdrücklich erwünscht.

Ziel ist es, Wissen zu teilen und den Austausch innerhalb der Retro-Community zu fördern.
insbesondere im Umfeld des NKC 68000 (mit JADOS 3.5) und hier, da als Plattform genutzt, der FPGA NKC mit TangNano 20k aus dieser Community.

Historischer Kontext

Datenbanksysteme wie dBase II und dBase III prägten in den 1980er Jahren die Anwendungsentwicklung auf Mikrocomputern maßgeblich.

CBASE greift diese Ideen auf und überträgt sie auf ein modernes Retro-System, den NKC 68000 auf FPGA Basis mit dem TangNano 20k, einer Weiterentwicklung, ohne den Anspruch auf Vollständigkeit oder historische Exaktheit im Detail.

CBASE entstand aus der Faszination für frühe Datenbanksysteme und dem Wunsch, deren Prinzipien auf moderner Retro-Hardware wieder lebendig zu machen. Das Programm entstand zusammen mit der freien Version von ChatGPT 4.0/5.0. Ziel war und ist das Erlernen von C für den 68000 unter JADOS 3.5. Dabei wurde ChatGPT als geduldiger Lehrer kennengelernt.

Verwendung von Namen

Der Begriff dbase II oder dbase III in diesem Handbuch bezieht sich nicht auf Softwareprodukte sondern lediglich auf das Datenbankformat der Datenbanksysteme dbase II / III aus den 1980er Jahren.

Beschreibung

Entwicklungssoftware (Stand 01/2026)

Für die Entwicklung wurde der Crosscompiler m68k-gcc unter Linux DEBIAN 12 (Bookworm) verwendet. In diesem Fall mit einem gcc-6.3.0 und newlib-2.5.0 und dem Environment nkc_common von Andreas Voggeneder's NKC TangNano. Die Toolchains können von der Projektwebseite (Stand 01/2026) heruntergeladen werden:

Für die Entwicklung auf Windows empfiehlt sich die fertige Toolchain SYSGCC für Windows https://github.com/avg67/nkc/tree/main/SW in Verbindung mit MSYS2 als Unix-Shell.

Der Crosscompiler lässt sich auch auf Linux installieren. Hier gibt es Informationen im NKC-Forum bei

https://www.nkcforum.de/forumdrc/index.php

in verschiedenen Rubriken.

Hardware (Stand 01/2026)

Geschrieben wurde die Software für den NKC 68000 auf Basis eines FPGA mit TangNano, der alle historischen Komponenten des NKC (NDR-Klein-Computer) aus den 1980er Jahren zur Verfügung stellt. Informationen /home/name/working/m68k-gcc/Examples/cbasehierzu bei

https://www.nkcforum.de/forumdrc/index.php

https://test.nkc-wiki.de/index.php?title=TANG-NANO

https://github.com/avg67/nkc/tree/main/tang_nano_20k

Eine Lauffähigkeit auf den dedizierten NKC Systemen mit 68000 oder 68008 wurde nicht geprüft.

Quellcode, Hilfsprogramme

Im bereitgestellten ZIP-File findet sich der gesamte Quellcode inklusive der Makefile und Shell Skripten zur Übertragung auf den NKC 68000 (TangNano):

cbase.c	Startprogramm mit main()
cbase.h	Alle Randbedingungen
create.c	CREATE Datenbank
create.h	
db.c	Datenbankgenerierung und Dateiverwaltung
db.h	
export.c	Exportschnittstelle für CSV Dateien
export.h	
help.c	Hilfe zur Laufzeit mit Übersicht und Einzelbefehlen
help.h	
locate.c	Funktionen zur Suche in der Datenbank mit Ergebnismenge
locate.h	

messages.h	Meldungen
record.c	Funktionen für die Datensatzverwaltung
record.h	
show.c	Funktionen für die Anzeige der Datensätze
show.h	
util.c	Hilfsfunktionen
util.h	
Makefile	Makefile für den Crosscompiler m68k
trans.sh	Script zur Übertragung des cbase.68k an den NKC
dtrans.sh	Script zur Übertragung der Demodatenbank an den 68k

Verzeichnisstruktur

Die Verzeichnisorganisation auf dem Entwicklungssystem orientiert sich an den Vorgaben aus der fertigen Toolchain. In diesem Fall für Linux (DEBIAN 12 Bookworm)

/home/name/working

Darunter der Crosscompiler

/home/name/working/m68k-gcc

und die Beispielverzeichnisse mit dem cbase-Verzeichnis

/home/name/working/m68k-gcc/Examples/cbase
/home/name/working/m68k-gcc/Examples/cbase/obj
/home/name/working/m68k-gcc/Examples/cbase/_out

Datenübertragung zum NKC (TangNano)

In _out befindet sich noch das dl.py Programm zur Übertragung mit PYTHON3 und pyserial über den USB-Anschluss des Rechners. Üblicherweise wird als Übertragungskanal

/dev/ttyUSB1

verwendet. Am TangNano 20k wird der DEBUG-USB-Port verwendet, über den auch die Stromversorgung des TangNano vom Entwicklungsrechner aus erfolgt.

Software am NKC (TangNano)

Auf der NKC (TangNano) Seite ist das Grundprogramm GP 7.1 R5 installiert und darüber JADOS 3.5, das auf der SD-Karte im TangNano gestartet wird.

Übertragungssoftware auf dem NKC (TangNano)

Auf JADOS 3.5 wird das Übertragngsprogramm RS232D benötigt, dass den UP- und Download von Dateien vom Entwicklungsrechner über das dl.py realisiert.

Informationen zur Übertragung findet sich auch wieder in den oben genannten Foren und auf Github.

Erste Schritte in CBASE

CBASE verarbeitet das dbase III kompatible Datenbankformat. Es verzichtet auf Indexdateien. Die geöffnete Datenbank wird komplett im Arbeitsspeicher gehalten und verwaltet. Zur Speicherung auf Datenträger stehen Befehle zur Verfügung. CBASE kann auch neue, dem dbase III Datenformat entprechende Datenbanken erzeugen.

Einschränkungen

CBASE ist nicht geeignet, große Datenbanken mit sehr vielen Datensätzen (>= 1000) und sehr vielen Datenfeldern (>= 23) zu verarbeiten. Das war auch bei der Inspiration zur Entwicklungszeit nicht vorgesehen.

Die Anzahl der Datenfelder ist durch die Anzahl der 24 Bildschirmzeilen im NKC begrenzt. Ein Seitenübergreifendes Scrolling in APPEND, EDIT und INSERT ist nicht implementiert.

Mehr als 999 Datensätze für in der Anzeige von SHOW zu Problemen in der ersten Spalte, in der die Recordnummer angezeigt wird.

Es kann nur eine Datenbank zur Laufzeit bearbeitet werden.

CBASE ist ein Retro-Projekt und kein kommerzielles System. Es dient dem Lernen der Programmiersprache C auf dem NKC 68000.

Grundsätzliche Bedienung

Programmstart

CBASE wird in JADOS 3.5 durch die Eingabe von

cbase

gestartet. Als Parameter kann der Name einer im selben Laufwerk befindlichen Datenbank (<name>.dbf> angegeben werden.

cbase <name>

CBASE wird dann die Datenbank sofort für die Bearbeitung öffnen, wenn sie vorhanden und korrekt ist.

Hilfesystem in CBASE

Bereits nach dem Start wird der Hinweis auf eine direkte Hilfe im System angeboten.

HELP for help…

Über HELP, gefolgt von <RETURN> gelangt man in eine Übersicht aller Befehle.

Über HELP <Befehl> erhält man eine Beschreibung zum Befehl. Die Befehlsliste kann über HELP ohne Parameter für die erste Information abgeholt werden.

Eingabe und Historie

Befehle werden in CBASE am Prompt „.“ eingegeben und mit <RETURN> bestätigt.

Mit <ESC> können Befehle vorzeitig abgebrochen werden. CBASE kehrt dann zum Prompt zurück und wartet auf das nächste Kommando.

CBASE verfügt über eine Befehlshistorie von insgesamt 10 Befehlen. Früher erfasste Befehle können über <PFEIL AUF> oder <PFEIL AB> erreicht werden, danach folgt <RETURN> für die Bestätigung. Die Befehlshistorie wird nicht gespeichert und zum Programmstart leer.

Alle Befehle können in Kleinbuchstaben eingegeben werden. Die Konvertierung erfolgt im Programm automatisch, wo erforderlich.

Aktiver Datensatz

Über Navigtionsbefehle wird ein Datensatz aktiv gesetzt. Im SHOW Befehl wird dies mit einem „>“ nach der Datensatznummer angezeigt. Der aktive Datensatz wird dann z.B. beim EDIT Befehl direkt ohne Angabe der Datensatznummer angesprochen.

Programm beenden

CBASE wird am Prompt „.“ mit

QUIT gefolgt von <RETURN> beendet.

Wenn noch nicht gespeicherte Daten im Arbeitsspeicher vorliegen, erfolgt ein Warnmeldung, die mit <J> (Verlust der Daten) oder <N> (Zurück zum Programm) beantwortet wird.

Befehl von CBASE

Nachfolgend eine kurze Darstellung der verfügbaren Befehle zur Datenbankverwaltung und -bearbeitung mit CBASE, gegliedert nach Funktionsblöcken.

Datenbank

In dieser Gruppe finden sich alle Befehle, die zur Verwaltung einer Datenbak in CBASE benötigt werden.

CREATE

CREATE <Datei> Neue Datenbank <Datei>.DBF anlegen\n"

Syntax:
CREATE <Datei>

Beschreibung:
Eine neue Datenbank wird angelegt mit dem Namen <Datei>.DBF.
Wenn der Dateiname gueltig ist, wird der Datenbankheader angelegt und die Struktur kann Zeilenweise für jedes Feld erfolgen.

Hinweis: Die folgenden Eingaben erfolgen in Großbuchstaben, es erfolgt keine interne Umwandlung. Fehleingaben werden mit „Invalid field definition“ abgelehnt und die Eingabe kann wiederholt werden.

Feldtyp C (Character)
Es wird angegeben: NAME,C, LÄNGE (max. 254 Zeichen)

Feldtyp L (logisch)
Es wird angegeben NAME, L

Feldtyp D (Datum)
Es wird angegeben: NAME,D

Feldtyp N (Numerisch)
Es wird angegeben:
Ganzzahlen: NAME,N,Feldlänge gesamt, 0 für Ganzzahlen
Dezimalzahlen: NAME,N,Feldlänge gesamt inkl. Komma, Anzahl der Nachkommastellen

Mit RETURN in der letzten, leeren Eingabezeile wird die Erfassung abgeschlossen.
Mit ESC erfolgt Abbruch ohne Speichern

Allgemeine Regeln:
Feldname maximal 10 Zeichen, keine Sonderzeichen, kein Blank, Grossbuchstaben
Typ ist C (Zeichen), N (numerisch), L (logisch), D (Datum)
Typ C: Feldlaenge maximal 254 Zeichen
Typ L: Feldlaenge immer 1
Typ D: Feldlaenge immer 8 (jjjjmmtt)
Typ N: Feldlaenge inkl. Dezimalpunkt und Nachkommastellen
Dezimalstellen muessen in der gesamten Feldlaenge eingerechnet sein.

Hinweis: Wenn die Summe aller Felder in der Länge größer als die Anzeigebreite am Monitor mit 80 Zeichen, so werden die Daten über den rechten Rand hianusgeschrieben und sind im SHOW Befehl nicht mehr sichtbar. Es gibt aktuell keinen Befehl der die Darstellung filtert nach Feldern.

USE

USE <Name> Datenbank <Name> öffnen

Syntax:

USE <Name> Datenbank mit Namen <Nem> öffnen
USE DIR Verzeichnis mit Datenbanken (*.DBF) anzeigen

Beschreibung:

Der USE Befehl öffnet eine gültige DBF-Datenbank.
Sollten noch nicht gespeicherte Daten einer noch offenen Datenbank vorliegen, erfolgt eine Warnung.

SAVE

SAVE Offene Datenbank speichern

Syntax:

SAVE Speichert offene Datenbank unter ihrem Namen ab.

Beschreibung:
Aktuell geöffnete Datenbank wird unter ihrem Namen zurück gespeichert. Als gelöscht markierte Datensätze bleiben erhalten.

PACK

PACK gelöscht markierte Datensätze werden entfernt

Syntax:

PACK

Beschreibung:

Mit PACK werden als gelöscht markierte Datensätze entfernt. PACK speichert die Datenbank sofort unter ihrem Namen auf dem Datenträger. Ein vorheriges Backup kann mit COPY erzeugt werden.

COPY

COPY <Datei> COPY speichert datenbank unter anderem Namen.

Syntax:

COPY <Datei>
COPY LOC <Datei>
COPY DIR

Beschreibung:

Mit COPY kann die aktuell geöffnete Datenbank unter einem neuen Namen abgespeichert werden. Exisitiert der Dateiname bereits, wird eine Warnmeldung ausgegeben.

Mit COPY LOC werden nur Datensätze, die der mit LOCATE ermittelten Trefferliste entsprechen, in die neue Datei gespeichert.

COPY DIR zeigt alle Datenbankdateien im aktuellen Verzeichnis.

EXPORT

EXPORT Exportiert Daten im CSV Textformat

Syntax:

EXPORT <Datei>
EXPORT ALL <Datei>
EXPORT LOC <Datei>
EXPORT DIR

Beschreibung:

EXPORT exportiert den aktuell markierten Datensatz „>“ in eine Textdatei mit der Endung CSV. Die Endung CSV wird automatisch vergeben.

EXPORT ALL exportiert alle Datensätze mit Ausnahme der als gelöscht markierten Datensätze.

EXPORT LOC exportiert alle Datensätze, die mit LOCATE ermittelt wurden.

EXPORT DIR zeigt alle CSV Dateien im aktuellen Verzeichnis.

INFO

INFO Zeigt Informationen zur geöffneten Datenbank

Syntax:

INFO

Beschreibung:

INFO zeigt Informationen zur geöffneten Datenbank. Es werden der

- Name
- Anzahl der Datensätze
- Länge eines Datensatzes
- Größe des Headers
- Anzahl der Felder
- das DIRTY-Flag (liegen Änderungen vor)
- Der Speicherverbrauch in ganzen Kilobytes
- Anzahl der Sektoren auf dem Datenträger
- und eine Liste der Felder

angezeigt.

CLOSE

CLOSE	Die geöffnete Datenbank wird geschlossen.

Syntax:

CLOSE

Beschreibung:

CLOSE schießt die aktuell geöffnete Datenbank. Es werden keine Daten gespeichert. Sollten noch ungespeicherte Daten vorliegen, erfolgt eine Sicherheitswarnung.

Datensatzbearbeitung

In dieser Rubrik werden Befehle für die Datensatzbearbeitung besprochen.

APPEND

APPEND Ein neuer Datensatz wird angelegt.

Syntax:

APPEND

Beschreibung:

Mit APPEND wird ein neuer Datensatz angelegt. Zunächst wird die neue Datensatznummer angezeigt. Danach erfoglt die Eingabe in einer einfachen Maske.

Fehleingaben werden durch eine Meldung gezeigt und können korrigiert werden.

Es wird Groß-/Kleinschreibung in Charakterfeldern (C) unterschieden. Die Eingabelänge ist durch die Feldlänge begrenzt.

In Logikfeldern (L) wird F (false), T (true), J (Ja), Y (Yes) und N (Nein, No) zugelassen.

In Datumsfeldern erfolgt die Eingabe im Format TT.MM.JJJJ. Der Punkt muss nicht angegeben werden. Intern wird im dbase-Format mit JJJJMMTT gespeichert, was für die Sortierung nützlich ist.

Numerische Felder werden als Ganzzhal oder Dezimalzahl erfasst. Es wird der Dezimalpunkt als Trennung verwendet. Ganzhaleingaben in Feldern mit Nachkommastellen werden automatisch mit dezimalpunkt und der Anzahl der Dezimalstellen mit „0“ aufgefüllt.

Falsche Eingaben werden mit einer Meldung angemahnt und können geändert werden.

Mit <PFEIL AUF> und <PFEIL AB> kann durch die Felder navigiert werden.

Mit ESC wird die Eingabe abgebrochen.

Wird das letzte Feld nach unten verlassen, wird gefragt, ob der Datensatz gespeichert werden soll. Wenn mit <J> bestätigt wird, kann mit einem weiteren <J> ein nächster Datensatz angelegt werden. Mit <N> wird APPEND verlassen.

INSERT

INSERT Neuer Datensatz wird an bestimmter Position

Syntax:

INSERT [BLANK] <nr>
INSERT AFTER [BLANK] <nr>

INSERT BEFORE [BLANK] <nr>

Beschreibung:

Mit INSERT wird ein neuer Datensatz an bestimmter Position <nr> vor (BEFORE), oder nach (AFTER = Default) eingefügt und wie bei APPEND angelegt (siehe dort). Mit der Option BLANK wird ein leerer Datensatz eingefügt.

Der neue Datensatz wird zum aktuellen Datensatz und kann z.B. mit EDIT bearbeitet werden.

EDIT

EDIT Einen bestehenden Datensatz ändern

Syntax:

EDIT <nr>
EDIT

Beschreibung:

Mit EDIT wird ein bestehender Datensatz geändert. Wird <nr> angegeben, wird der Datensatz mit der Datensatznummer <nr> in der Eingabemaske aufgerufen. EDIT ohne Datensatznummer gibt den aktuellen Datensatz zur Änderung.

Die Eingabe erfolgt wie bei APPEND beschrieben. Alle Datensatzinhalte werden dargestellt und können geändert werden. Wird ein bestehendes Feld mit <RETURN> oder <PFEIL AUF>, <PFEIL AB> übersprungen, bleiben die vorhandenen Daten unverändert erhalten.

Eine Sicherheitsabfrage schließt den BEFEHl EDIT ab.

DEL

DEL Markiert Datensatz zum Löschen

Syntax:

DEL <nr>

Beschreibung:

DEL markiert den Datensatz mit der Datensatznummer <nr>als gelöscht. Gelöscht markierte Datensätze werden erst mit PACK aus der Datensatz entfernt.

UNDEL

UNDEL Gelöscht markierten Datensatz „entlöschen“

Syntax:

UNDEL <nr>

Beschreibung:

Mit UNDEL <nr> wird die Löschmarkierung des datensatzes mit der Datensatznummer <nr> wieder entfernt.

Navigation in der Datenbank

In diesem Kapitel wird die Naviagtion in der geöffneten Datenbank beschrieben.

GOTO

GOTO Gehe zu Datensatz

Syntax:

GOTO <nr>
GOTO TOP
GOTO BOTTOM
GOTO PREV
GOTO NEXT

Beschreibung:

Mit GOTO <nr> wird der mit der Datnsatznummer <nr> angegebene Datensatz zum aktiven Datensatz. Ist die angegebene Datensatznummer ungültig (höher als die Anzahl der Datensätze in der Datenbank), wird zum letzten Datensatz gesprungen.

GOTO TOP aktiviert den ersten, GOTO BOTTOM den letzten Datensatz in der Datenbank.
GOTO PREV aktiviert den vorherigen (in Bezug auf den aktuellen aktivierten Datensatz) und GOTO NEXT den darauffolgenden Datensatz.

Der Datensatzzeiger „>“ makriert den aktuellen, aktiven Datensatz.

SKIP

SKIP Datensatzzeiger bewegen

Syntax:

SKIP
SKIP <nr>
SKIP <-nr>

Beschreibung:

Mit SKIP wird der Datensatzzeiger (aktueller, aktiver Datensatz) beeinflusst.

Mit SKIP ohne Argument wird der Datensatzzeiger inkrementiert und zeigt auf den nächsten Datensatz in der Datenbank.

Mit SKIP <nr> wird der Zeiger um die Anzahl <nr> vorwärts verschoben.

Mit SKIP ← nr> wird er Zeiger rückwärts verschoben.

Der Datensatzzeiger wird z.B. auch in SHOW verwendet und markiert den Beginn der angezeigten Liste.

Datensätze suchen und sortieren

Hier geht es jetzt um Befehle in CBASE zum Suchen in einer Datenbank und Sortieren.

LOCATE

LOCATE Suchen von Datensätzen

Syntax:

LOCATE <feld> <text>
LOCATE <feld> <op> <zahl>

Beschreibung:

Mit LOCATE werden Daten in der Datenbank gesucht.

Mit LOCATE <feld> <text> wird in Charakterfeldern (Typ C) gesucht. Wenn es einen Treffer gibt, wird die erste Datensatznummernummer ausgegeben. Im C-Feld wir eine Instring-Suche durchgeführt.

Mit LOCATE <feld> <op> <zahl> wird in numerischen Feldern gesucht.
Als <op> stehen < <= = => > zur Verfügung.

Der Datensatz mit dem ersten Treffer wird zum aktuellen, aktiven Datensatz. Der nächste Treffer kann mit CONT erreicht werden. Mit SHOW LOC werden alle Treffer angezeigt.

CONT

CONT Markiert den nächsten Treffer von LOCATE

Syntax:

CONT

Beschreibung:

Mit CONT wird nach einer erfolgreichen Suche mit LOCATE der Nächste, als Treffer identifizierte Datensatz in der der Ergebnismenge als aktueller Datensatz markiert.

SORT

SORT Sortieren von Datensätzen

Syntax:

SORT ON <feld> [DESC][, <feld> [DESC]] … TO <datei>

Beschreibung:

Mit SORT über ein Datenbankfeld <feld> [und mehrere weitere Felder] wird die aktuelle Datenbank sortiert in eine neue Datenbank <datei> in der Standardsortierfolge ASC gespeichert.

ASC bedeutet aufsteigend, DESC absteigende Sortierung.

Es sind maximal 8 Sortierfelder erlaubt.

Der Datenbankname wird automatisch mit DBF erweitert.

Datensätze anzeigen

SHOW

SHOW Anzeige von Datensätzen

Syntax:

SHOW
SHOW ALL
SHOW LOC
SHOW <anzahl>
SHOW <start> <anzahl>
SHOW DEL
SHOW DEL <start> <anzahl>

Beschreibung:

SHOW zeigt Datensätze seitenweise an. SHOW ohne Parameter zeigt den aktuell markierten Datensatz an. Der aktuelle Datensatz ist immer der erste Datensatz in einer Liste, außer es werden andere Optionen gewählt.

SHOW ALL zeigt alle Datensätze an. Bei größeren Listen ist die Ausgabe seitenweise organisiert. Die Anzeige kann mit <RETURN> fortgesetzt oder mit <ESC> abgebrochen werden

SHOW LOC zeigt die Trefferliste von LOCATE

SHOW <anzahl> zeigt die <anzahl> Datensätze nach dem aktuellen Datensatz an.

SHOW <start> <anzahl> definiert einen Startdatensatz (Datensatznummer> und die <anzahl> Datensätze, die gelistet werden. Ist die Anzahl größer als eine Bildschrimseite, wir die Ausgabe unterbrochen, kann mit <RETURN> fortgesetzt oder mit <ESC> abgebrochen werden.

SHOW DEL zeigt als gelöscht markierte Datensätze.

SHOW DEL <start> <anzahl> ist identisch zum Befehl wie oben, gilt jedoch nur für gelöscht markierte Datensätze.

Systembefehle

Zum Ende werden noch die Systembefehle und Beschreibungen gelistet.

CLS

CLS Löscht den Bildschirm

Syntax:

CLS

Beschreibung:

CLS löscht den Bildschirmzeilen

DIR

DIR Ziegt Inhaltsverzeichnis des Datenträgers

Syntax:

DIR

Beschreibung:

DIR Zeigt den Inhalt des aktuellen Datenträgers (*,*) an.

HELP

HELP Zeigt Hilfe über Programm oder Befehl

Syntax:

HELP
HELP <befehl>

Beschreibung:

HELP zeigt eine Übersicht zu den Befehlen im Programm CBASE

HELP <befehl> zeigt Hilfe zu einem Befehl an.

HELP CBASE zeigt Informationen zur Bedienung

HELP PROJEKT zeigt Informationen zum CBASE Projekt.