# THE PICK PC SYSTEM

# TUTORIAL

# THE PICK SYSTEM

# TUTORIAL

## PROPRIETARY INFORMATION

## 1.1 INTRODUCTION

This tutorial will guide the new user of PICK through the basic operations of the system.

This tutorial will allow the new user to:

Learn how to log on to the system
Get data from the files produced into meaninful reports
Create data files
Define data file content

The PICK System uses some terms that are a little different than those used by most systems in data processing. Basic comparisons could be:

| PICK SYSTEM | OTHER SYSTEMS |
|---|---|
| Item | Record |
| Item-id | Key |
| Attribute | Field |

## 1.2 LOGON

When the system is booted up, or turned on, it verifies the operating system and has the user enter the time and date. Time is entered as 24-hour military time, therefore, when it is 2:00 PM, the time is entered as 14:00. The date is entered as MM-DD-YY.

The system will then display a message that it is verifying the system. This means that it is checking each frame of system code to make sure that everything is the way it should be.

When this is complete, the system will display:

```
Logon
```

Logging on is the means by which an account is entered. You must be logged on to an account in order to do any data processing. Every account has a Master Dictionary which contains all file names for that account and verbs that will cause the system to perform specific actions.

The name of the account that will be used to learn the system is called TUTOR.

To enter the PC-XT Tutorial Account, key in:

```
TUTOR <CR>
```

NOTE: <CR> means to press the Carriage Return key.

The screen will display:

```
PASSWORD:
```

The password for the TUTOR account is LEARN <CR>. When this is keyed in the system will not display it. If keyed in incorrectly, the system will return with the message:

```
USER ID?
```

Key in TUTOR again, and carefully key in LEARN for the password.

The system will enter the account, and display the TCL prompt (>).

## 1.3 TCL or TERMINAL CONTROL LANGUAGE

The TUTOR account is set up to display the TCL prompt. Unless the system is set up to automatically enter a program or another process the user will be at TCL.

TCL is shorthand for Terminal Control Lanuage. The user can tell if they are at TCL because a ">" prompt displays at the left-hand side of the screen. TCL is the basic way that the user communicates with the PICK Operating System.

Verbs can be entered at TCL to access files. Verbs are commands such as LIST, SORT, COPY, etc. TCL commands are not executed until a carriage return (noted as <CR>) is entered.

Although the tutotrial is primarily concerned with ACCESS commands, it should be pointed out that there are a variety of other commands which are also entered at the TCL prompt. All of them may be found in the PICK USER'S MANUAL. A few useful ones are noted here. The user may enter these commands any time the system is displaying the TCL prompt ( > ).

```
>TIME
```

Outputs system time and date.

```
>LISTU
```

List all ports (users) currently logged on the system.

```
>POVF
```

Display the available space (frames) on the disk.

```
>LISTFILES
```

List the files on this account.

The user should consult the PICK USER'S MANUAL for additional options and commands which are available at TCL.

## 1.4 ACCESS

ACCESS is a powerful, yet easy-to-use database retrieval language. ACCESS commands are entered at TCL. Let's enter a simple ACCESS command to see how the system works:

```
>LIST CUST <CR>
```

Where:

```
        LIST    is the ACCESS verb to list a file
        CUST    is the name of the file
```

The screen will display:

```
PAGE   1                                             17:53:38  21 JAN 1985

CUST : 1006
COMPANY   TRACK AUTOMOTIVE
CONTACT   JACK NORTON
ADDRESS   7812 MAIN STREET
CITY   NEWARK
STATE   NJ
ZIP   07182
TELEPHONE   (206) 555-8347
INV#   17254   23846   48776   49003   55241
AMT   $11.27   $392.72   $371.82   $984.84   $93.89
DATE1   15 DEC 1984   31 DEC 1984   15 JAN 1985   25 JAN 1985

CUST : 1000
COMPANY   ACME HARDWARE COMPANY
CONTACT   JOHN THOMPSON
ADDRESS   1134 BRISTOL PKWAY
CITY   IRVINE
STATE   CA
ZIP   92714
TELEPHONE   (714) 555-9384
```

The system will display 22 lines of data on the screen. Of course, most reports are more than a screenful, so to see the remainder, key a <CR>. If one screenful is enough information, then press the <CTRL> and <X> keys simultaneously. The listing will terminate and the system will return to TCL.

The definitions on the attributes that were displayed are:

```
CUST#     - The item-id of this item in the CUST file
COMPANY   - The first attribute, defined as both "1" and "COMPANY"
CONTACT   - The second attribute, defined as both "2" and "CONTACT'
ADDRESS   - The third attribute, defined as both "3" and "ADDRESS"
CITY      - The fourth attribute, defined as both "4" and CITY
STATE     - The fifth attribute, defined as both "5" and STATE
ZIP       - The sixth attribute, defined as both "6" and ZIP
TELEPHONE - The seventh attribute, defined as both "7" and TELEPHONE
INV#      - The eighth attribute, defined as both "8" and INV#
AMT       - The ninth attribute, defined as both "9" and AMT
DATE1     - The tenth attribute, defined as both "10" and DATE1
```

The attributes are defined in a section of the file called the dictionary. Each file has a dictionary section and a data section. When a LIST command is given the system will look at the file dictionary and display all sequential fields defined as 1, 2, 3, etc. If these have not been defined, then the system will display only the item-id for each item in the file.

The attributes INV#, AMT and DATE1 are called multi-valued attributes. In other words, there is more than one value in the field. Attributes can be broken down into values and these values further broken down into sub-values to define data.

This CUST file has synonym attributes defined. This means that whether the field is entered as "1" or "COMPANY" the result will be the same, because both are defined exactly the same way in the dictionary of CUST.

The user may define any number of synonym definitions, which are helpful because the user doesn't have to remember which attribute is number four, but can call up the data by specifying a meaninful synonym (e.g., "CITY" or "SHIP.TO.CITY", etc.).

If a listing was to be sorted by the Customer Number, the command would be:

```
>SORT CUST <CR>
```

The screen will display:

```
PAGE    1                                         17:53:11  21 JAN 1985

CUST : 1000
COMPANY   ACME HARDWARE COMPANY
CONTACT   JOHN THOMPSON
ADDRESS   1134 BRISTOL PKWAY
CITY   IRVINE
STATE   CA
ZIP   92714
TELEPHONE   (714) 555-9384
INV#   48372   49182   50192   51327   82712
AMT   $512.13   $439.98   $283.47   $283.74   $182.73
DATE1   28 OCT 1984   18 NOV 1984   15 DEC 1984   31 DEC 1984

CUST : 1001
COMPANY   NEWTON DEVELOPMENT
CONTACT   THOMAS NEWTON
ADDRESS   1970 SKYLARK STREET
CITY   HUNTINGTON BEACH
STATE   CA
ZIP   92785
TELEPHONE   (714) 555-9283
```

Notice that this time, the difference is that the CUST items are sorted in order.

It should be evident that these two commands are very similar. The real difference being that one lists the items in the order they are stored on the disk, and the other sorts the items by the item-id.

## 1.5 ACCESS SENTENCE SYNTAX

ACCESS is an English-like inquiry language. Each ACCESS "sentence" must consist of a verb followed by a file name.

A verb is an action-oriented word which will invoke a specific ACCESS processor. The LIST CUST statement above is an example of the simplest form of an ACCESS command.

However, ACCESS commands may be made even more useful, using the verb and file name and then adding selection criteria, sort keys, output specifications and print limiters, to get custom reports.

To list the file and only see certain attributes, key in:

```
>LIST CUST COMPANY CITY STATE <CR>
```

The screen will display:

```
PAGE   1                                          17:54:06  21 JAN 1985

CUST...... COMPANY.................. CITY................ STATE

1006       TRACK AUTOMOTIVE          NEWARK               NJ
1000       ACME HARDWARE COMPANY     IRVINE               CA
1007       MESA TRAVEL AGENCY        HUNTINGTON BEACH     CA
1001       NEWTON DEVELOPMENT        HUNTINGTON BEACH     CA
1008       WORD ALBEGRA              CHICAGO              IL
1002       UPTOWN PRINTERS           LOS ANGELES          CA
1009       MY TIMES MAGAZINE         NEWARK               NJ
1003       RITE-WAY DRUGS            CHICAGO              IL
1010       PICK SYSTEMS              IRVINE               CA
1004       LIKE-NU UPHOLSTERY        CHICAGO              IL
1005       A-1 APPLIANCES            NEWARK               NJ

11 ITEMS LISTED.
```

NOTE: >LIST CUST 1 4 5 <CR>
Would have produced an identical listing.

(See 6.26 LIST verb)

Notice that this time the data displayed across the screen in a horizontal fashion rather than down the page as in the first two listings. This is because a screen can only display 79 characters. ACCESS will check to see if the generated report will be wider than 79 characters. If it is, then the listing is done vertically. If the listing fits into 79 characters, the system will list the data horizontally as shown above.

## 1.6 SORTING WITH ACCESS

To produce a report that lists only certain fields, is sorted by zip code and lists only the company name, city and zip code, enter:

```
>SORT CUST BY ZIP COMPANY CITY ZIP <CR>
```

Where:

| | |
|---|---|
| SORT | is the ACCESS verb |
| CUST | is the file name |
| BY ZIP | is the attribute to sort by |
| COMPANY | is the first attribute to display. |
| CITY | is the second attribute to display. |
| ZIP | is the third attribute to display. |

The screen will display:

```
PAGE   1                                            18:02:00  21 JAN 1985

CUST...... COMPANY................... CITY................ ZIP..

1005       A-1 APPLIANCES             NEWARK              07152
1006       TRACK AUTOMOTIVE           NEWARK              07182
1009       MY TIMES MAGAZINE          NEWARK              07273
1008       WORD ALBEGRA               CHICAGO             60611
1003       RITE-WAY DRUGS             CHICAGO             60623
1004       LIKE-NU UPHOLSTERY         CHICAGO             60681
1002       UPTOWN PRINTERS            LOS ANGELES         90099
1007       MESA TRAVEL AGENCY         HUNTINGTON BEACH    92647
1000       ACME HARDWARE COMPANY      IRVINE              92714
1010       PICK SYSTEMS               IRVINE              92714
1001       NEWTON DEVELOPMENT         HUNTINGTON BEACH    92785

11 ITEMS LISTED.
```

NOTE: >SORT CUST BY 6 1 4 6 <CR>
Would have produced an identical listing.

(See 6.27 SORT verb)

## 1.6.1 DESCENDING SORTS WITH ACCESS

If the report was to be sorted with zip codes in the order 99999 to 00001, instead of the usual 00001 to 99999, the ACCESS command would be:

```
          >SORT CUST BY-DSND ZIP COMPANY CITY ZIP <CR>

PAGE   1                                        17:55:05  21 JAN 1985

CUST...... COMPANY.................. CITY................ ZIP..

1001       NEWTON DEVELOPMENT        HUNTINGTON BEACH     92785
1000       ACME HARDWARE COMPANY     IRVINE               92714
1010       PICK SYSTEMS              IRVINE               92714
1007       MESA TRAVEL AGENCY        HUNTINGTON BEACH     92647
1002       UPTOWN PRINTERS           LOS ANGELES          90099
1004       LIKE-NU UPHOLSTERY        CHICAGO              60681
1003       RITE-WAY DRUGS            CHICAGO              60623
1008       WORD ALBEGRA              CHICAGO              60611
1009       MY TIMES MAGAZINE         NEWARK               07273
1006       TRACK AUTOMOTIVE          NEWARK               07182
1005       A-1 APPLIANCES            NEWARK               07152

11 ITEMS LISTED.
```

NOTE: `>SORT CUST BY-DSND 6 1 4 6 <CR>`
Would have produced an identical listing.

BY-DSCD tells ACCESS to sort this attribute in descending order, 9-0 for numbers and Z-A for alphabetic characters.

(See 6.27.1 BY-DSND modifier)

## ..7 CONTROL BREAKS WITH ACCESS

If the list is to be separated into categories, for instance city, then the command would be:

```
          >SORT CUST BY CITY COMPANY BREAK-ON CITY <CR>

PAGE   1                                          19:43:33  21 JAN 1985

CUST...... COMPANY.................. CITY................

1003       RITE-WAY DRUGS            CHICAGO
1004       LIKE-NU UPHOLSTERY        CHICAGO
1008       WORD ALBEGRA              CHICAGO

                                     ***

1001       NEWTON DEVELOPMENT        HUNTINGTON BEACH
1007       MESA TRAVEL AGENCY        HUNTINGTON BEACH

                                     ***

1000       ACME HARDWARE COMPANY     IRVINE
1010       PICK SYSTEMS              IRVINE

                                     ***

1002       UPTOWN PRINTERS           LOS ANGELES

                                     ***

1005       A-1 APPLIANCES            NEWARK
1006       TRACK AUTOMOTIVE          NEWARK
1009       MY TIMES MAGAZINE         NEWARK

                                     ***

***

11 ITEMS LISTED.
```

NOTE: >SORT CUST BY 4 1 BREAK-ON 4 <CR>
Would have produced an identical listing.

(See 6.25 CONTROL BREAKS)

## 1.8 HEADINGS & FOOTINGS WITH ACCESS

Reports may have either a heading or a footing so that the person reading it can readily ascertain what report they are looking at.

To this end, ACCESS has HEADING and FOOTING directives. Key in the following ACCESS sentence:

```
>SORT CUST COMPANY CITY STATE HEADING "MY FIRST PICK
 REPORT 'CL' TODAYS DATE IS 'DCL' PAGE 'PCL'" <CR>
```

The result should look like:

```
                         MY FIRST PICK REPORT
                      TODAYS DATE IS 23 JAN 1985
                               PAGE    1

CUST...... COMPANY.................. CITY................ STATE

1000       ACME HARDWARE COMPANY     IRVINE              CA
1001       NEWTON DEVELOPMENT        HUNTINGTON BEACH    CA
1002       UPTOWN PRINTERS           LOS ANGELES         CA
1003       RITE-WAY DRUGS            CHICAGO             IL
1004       LIKE-NU UPHOLSTERY        CHICAGO             IL
1005       A-1 APPLIANCES            NEWARK              NJ
1006       TRACK AUTOMOTIVE          NEWARK              NJ
1007       MESA TRAVEL AGENCY        HUNTINGTON BEACH    CA
1008       WORD ALBEGRA              CHICAGO             IL
1009       MY TIMES MAGAZINE         NEWARK              NJ
1010       PICK SYSTEMS              IRVINE              CA

11 ITEMS LISTED.
```

NOTE: >SORT CUST 1 4 5 HEADING... <CR>
Would have produced an identical listing.

(See 6.20 HEADINGS & FOOTINGS)

### 1.8.1 HEADING & FOOTING OPTIONS

The date on the report will be the current date that the report is run. The heading text must be enclosed in double quotes (") after the word heading. The mnemonics enclosed in single quotes (') must be within the HEADING double quotes and is telling the system:

C - Center the line
L - perform a Line feed
D - todays Date
P - incrementing Page number

There are other parameters that can be used in a heading or footing. The only difference between a HEADING and a FOOTING directive is that a heading prints at the top of each page and a footing at the bottom. Otherwise, the way they are used is exactly the same. ACCESS can generate both a heading and a footing on the same page.

(See 6.20 HEADINGS & FOOTINGS)

### 1.8.2 TOTAL MODIFIER

To total specific attribute values, consider the following :

```
>SORT CUST BREAK-ON COMPANY TOTAL AMT INV# DATE1 <CR>
```

WHERE:

SORT is the ACCESS verb
CUST is the file name
BREAK-ON causes a break-on output when value changes
COMPANY is the attribute to BREAK-ON
TOTAL totals all values of following attribute upon break
AMT is the attribute containing the values to be totaled
INV# is an attribute to display
DATE1 is an attribute to display

The result should be:

```
PAGE   1                                            20:04:18  21 JAN 198

CUST...... COMPANY.................. AMT... INV#...... DATE1...

1000       ACME HARDWARE COMPANY      $512.13 48372   28 OCT 1984
                                      $439.98 49182   18 NOV 1984
                                      $283.47 50192   15 DEC 1984
                                      $283.74 51327   31 DEC 1984
                                      $182.73 82712   15 JAN 1985

           ***                      $1,702.05

1001       NEWTON DEVELOPMENT         $489.38 18473   28 SEP 1984
                                      $384.98 28374   27 OCT 1984
                                      $184.89 39475   15 NOV 1984
                                      $852.43 48567   20 DEC 1984
                                      $348.78 50572   25 JAN 1985

           ***                      $2,260.46

  .          .                        .       .          .
  .          .                        .       .          .
  .          .                        .       .          .
  .          .                        .       .          .

1010       PICK SYSTEMS                $38.18 19573   14 SEP 1984
                                       $28.19 22014   28 OCT 1984
                                      $349.53 34001   15 DEC 1984
                                      $493.61 48900   31 DEC 1984
                                       $10.53 52261   27 OCT 1984

           ***                        $920.04

***                                $22,306.70

11 ITEMS LISTED.
```

(See 6.21 TOTAL MODIFIER)

## 1.9 SELECTION-CRITERIA: "WITH"

To make a selection from one of the attributes of the file, the ACCESS command line could be:

```
        >SORT CUST BY CITY COMPANY CITY ZIP WITH CITY = "CHICAGO" <CR>
```

The result should be:

```
PAGE   1                                                 09:35:28  24 JAN 1985

CUST...... COMPANY.................. CITY................

1003       RITE-WAY DRUGS            CHICAGO
1004       LIKE-NU UPHOLSTERY        CHICAGO
1008       WORD ALBEGRA              CHICAGO

3 ITEMS LISTED.
```

(See 6.10 SELECTION-CRITERIA)

Experiment with the CUST file and ACCESS commands. More detailed explanations and other commands may be found in the ACCESS chapter of the PICK USER REFERENCE MANUAL.

## 1.10 CREATING A FILE

All data on the PICK System is in files. Data files have two portions to them, the DICT portion and the DATA portion.

The DICT of a file has all of the attributes of the data file defined in it. The DICT controls how many characters to allocate an attribute upon output, whether it is left- or right-justified on a report, the column heading to print for an attribute on a report and other paramemters.

The DATA portion of a file contains the data. All of the ACCESS commands in the examples have been run against the data portion of the file CUST.

The PICK System stores data on the disk in "frames". A frame is 512 bytes (or characters). If one frame is full, then the system will automatically attach another frame to it so that a file can "grow" naturally.

When a file is created, the user must specify how many frames should be initially allocated for the DICT and DATA portions of the file. This is generally figured by how many characters there are going to be in an item (record) and how many items will be in the file.

Let's create a file called NAMES and then define what the fields will be and enter data. To do this, key in:

```
>CREATE-FILE NAMES 3 7 <CR>
```

WHERE:

```
CREATE-FILE   is the TCL command to create a file
NAMES         is the file name
3             is the number of frames to allocate to the DICT
7             is the number of frames to allocate to the DATA
              portion of the file
```

The numbers 3 and 7 indicate the number of frames to be reserved for the DICT and DATA portions of the file respectively. This is referred to as the MODULO of the file.

After the CREATE-FILE command is keyed in, the system will respond with:

```
[417] FILE 'NAMES' CREATED; BASE = XXXX, MODULO = 3, SEPAR = 1.
[417] FILE 'NAMES' CREATED; BASE = XXXX, MODULO = 7, SEPAR = 1.
```

The two lines that are returned by the system refer to the DICT and DATA portions respectively. BASE is the starting frame address, MODULO is how many frames were specified and SEPAR (separation) is always 1.

## 1.11 DEFINING DICTIONARY ATTRIBUTES

The DICT portion of the file has items that define what the data will be in the DATA portion of the file.

A PICK/BASIC program is on the TUTOR account that will allow you to define the DICT section of your NAMES file.

### 1.11.1 BUILD.DICT PROGRAM

To enter the Dictionary Definition program, key in:

```
>BUILD.DICT <CR>
```

The screen will prompt to enter the file name:

```
ENTER FILE NAME:  NAMES <CR>
```

If the file NAMES has not been created, then the system will return an error message that the file cannot be found. Return to the section on creating a file and create the NAMES file.

The entry screen will display:

```
---------------------------------------------------------------------------
:        A T T R I B U T E    D E F I N I T I O N    E N T R Y           :
:                                                                         :
:                          File DICT is:  NAMES                           :
                                                                          :
                           This ITEM-ID is:  1                            :
:                                                                         :
:    1.  Enter attribute NAME/DESCRIPTION:                                :
:                                                                         :
:    2.  Enter attribute JUSTIFICATION:                                   :
:                                                                         :
:    3.  Enter attribute LENGTH:                                          :
---------------------------------------------------------------------------
```

The BUILD.DICT program will allow the definition of up to ten (10) attribute definitions. The Item-id for these definition items start with "1" and increments for each new definition entered (up to 10). In a name and address file the definitions would probably be:

| DESCRIPTION | JUSTIFICATION | LENGTH |
|---|---|---|
| NAME | L | 20 |
| ADDRESS | L | 20 |
| CITY | L | 20 |
| STATE | L | 2 |
| ZIP | R | 5 |

The BUILD.DICT program will prompt for the description, justification and output length for each attribute defining item.

Whatever is defined as the DESCRIPTION will be the column heading on a report that is produced through ACCESS.

JUSTIFICATION refers to whether the data should line up at the left or right of the field. Alpha/numeric data is generally specified as being left justified and numeric data is generally specified as being right justified. The difference for numeric fields is:

```
LEFT JUSTIFY                   RIGHT JUSTIFY

  1001                               1001
  101                                 101
  10001                             10001
```

The LENGTH refers to the column width of an attribute upon output. If data is entered in an attribute that is longer than the defined length there is no error. However, that data will "wrap" on a horizontally listed report if it has more characters in the attribute than was defined in the length. This should not be a problem if fields are realistically defined for the length of the data.

While using the dictionary attribute definition program, if there is a question on an input field, press "?" as the first character and a help screen will display for that attribute.

After the description, justification and length have been defined for an attribute definition item, the system will prompt:

ALL FIELDS CORRECT? (Y/N)

If the data entered suits you, then key an upper case "Y" and press the carriage return. If there needs to be a change, press "N", a <CR> and the system will prompt for the line number to change. Enter the number (1 - 3) and press <CR>. The cursor will return to that field to correct input.

If all data is correct and "Y" is entered, the system will prompt;

ENTER ANOTHER ATTRIBUTE DEFINITION (Y/N):

Enter a "Y" and press carriage return until all the fields desired have been defined.

The program will automatically increment the Item-ID by 1. Again, only ten (10) attributes may be defined using this program. When all attributes have been defined answer the last prompt with an "N", a <CR> and the system will return to TCL.

## 1.12 INPUT.DATA PROGRAM

There is a program on the TUTOR account called INPUT.DATA. This program will allow you to enter information into the DATA portion of the NAMES file.

To use this program, key in:

```
>INPUT.DATA <CR>
```

The screen will prompt:

```
ENTER FILE NAME:  NAMES <CR>
```

Key in NAMES and press carriage return. If the NAMES file has not been created the system will return an error message. Return to the section on creating a file and create the NAMES file.

The screen will display:

```
---------------------------------------------------------------------------
:                    D A T A    E N T R Y    S C R E E N                  :
:                                                                         :
:    File name is:  NAMES                                                 :
:                                                                         :
:    Enter unique ID:                                                     :
---------------------------------------------------------------------------
```

Below this will be the descriptions that were defined for the DICT NAMES using the BUILD.DICT program. If there are no prompts for inputting data fields, then return to the BUILD.DICT program to create them.

Each item (or record) MUST HAVE A UNIQUE ID. This can be almost anything the user wants. For the purposes here, we suggest you use 101, 102, 103, etc.

After the Item-ID is entered, press carriage return to go from line to line and enter the appropriate data. When finished, the system will display:

```
DO YOU WANT TO ENTER ANOTHER ITEM? (Y/N)
```

Press "Y" to enter another item into the NAMES file. Press "N" to signal that data entry is complete and the system will return to TCL. Both must be followed with <CR>.

Once the data has been entered, return to the section on ACCESS and try some of the commands on your data. Since the file dictionary contains sequential item-ids ( 1,2...10), the command "LIST NAMES" will default to show as many attributes as you have consecutively defined.