	represented as an asterisk (*) indicating that all of the values (or subarrays) along that dimension are the same.
<majority>	The variable majority of the CDF. Variable majorities are described in Section 2.3.11.
<num-attributes>	The number of attributes in the CDF.
<num-dimensions>	The number of dimensions for the rVariables or a zVariable.
<num-g-entries>	The number of gEntries for the gAttribute.
<num-global-scope>	The number of gAttributes in the CDF.
<num-r-entries>	The number of rEntries for the vAttribute.
<num-r-records>	Of all the rVariables the maximum number of records written. (zVariables are not considered.)
<num-r-variables>	The number of rVariables in the CDF.
<num-records>	The number of records actually written for a variable (physical, not virtual).
<num-variable-scope>	The number of vAttributes in the CDF.
<num-z-entries>	The number of zEntries for the vAttribute.
<num-z-variables>	The number of zVariables in the CDF.
<rec-variance>	The variable record variance. A T if values do vary from record to record (VARY) or an F if they do not (NOVARY). Record variances are described in Section 2.3.9.
<record-num>	The record number. If the variable's record variance is FALSE (NOVARY), this will be an asterisk (*) indicating that the values are the same for all records.
<release>	The release number of the CDF library that created/modified the CDF.
<var-data-type>	The variable data type. Data types are described in Section 2.2.9.
<var-name>	The variable name.
<var-num>	The variable number.
<var-num-elems>	The number of elements of the data type. The number of elements for a variable is explained in Section 2.3.8.
<var-value>	The variable value.
<version>	The version number of the CDF library that created/modified the CDF.

If a display mask specifying less than "ihg+ar+z+v" is used, the information not requested will be omitted from the output.

3.8 SkeletonTable

3.8.1 Introduction

The `SkeletonTable` program is used to create an ASCII text file called a skeleton table containing information about a given CDF. It reads a CDF and writes into the skeleton table the following information.

1. Data encoding, majority, and format.
2. Number of dimensions and dimension sizes for the rVariables.
3. gAttribute definitions (and gEntry values).
4. rVariable and zVariable definitions and vAttribute definitions (and rEntry/zEntry values).
5. Non-record-variant (NRV) data for all of the CDF's NRV variables (if requested). NRV variables are described in Section 2.3.9

The above information is written in a format that can be "understood" by the `CDFskeleton` program (see Section 3.9). `CDFskeleton` reads a skeleton table and creates a new CDF.

3.8.2 Executing the SkeletonTable Program

The command line syntax for `SkeletonTable` is as follows:

Usage:

VMS:

```
$ SKELETONTABLE [/SKELETON=<skeleton-path>] [/[NO]LOG] [/ZMODE <mode>]
                [/NONRV | /NRVTABLE] [-[NO]NEG2POSFP0] [/[NO]FORMAT]
                [/REPORT=(<types>)] <cdf-path>
```

UNIX:

```
% skeletontable [-skeleton <skeleton-path>] [-[no]log] [-zmode <mode>]
                [-nonrv | -nrvtable] [-[no]neg2posfp0] [-[no]format]
                [-report "<types>"] <cdf-path>
```

MS-DOS:[4]

```
> skeletontable [-skeleton <skeleton-path>] [-[no]log] [-zmode <mode>]
                [-nonrv | -nrvtable] [-[no]neg2posfp0] [-[no]format]
                [-report "<types>"] <cdf-path>
```

[4] On MS-DOS systems the executable is named `CDF2SKT.EXE`.

Parameter(s):

`<cdf-path>` The pathname of the CDF to read. (Do not enter an extension.) It is from this CDF that the skeleton table will be created.

Qualifier(s):

`/SKELETON=<skeleton-path>`, `-skeleton <skeleton-path>`

`<skeleton-path>` is the pathname of the skeleton table to be created. Any valid pathname may be specified. (Do not enter an extension; `.skt` is appended automatically.) If this qualifier is not specified, the default skeleton table name is `<cdf-name>.skt` in the default/current directory (where `<cdf-name>` is the name portion of `<cdf-path>`).

`/NONRV`, `-nonrv` or `/NRVTABLE`, `-nrvtable`

Only one of these qualifiers may be specified. The meaning of each is as follows:

`/NONRV`, `-nonrv`

Ignore NRV data. (The values are not placed in the skeleton table.)

`/NRVTABLE`, `-nrvtable`

Put NRV data values in the skeleton table.

`/[NO]LOG`, `-[no]log` Specifies whether or not messages are displayed as the program executes.

`/ZMODE=<mode>`, `-zmode <mode>`

Specifies the zMode that should be used with the CDF. `<mode>` may be one of the following:

`0` Indicates that zMode should be disabled.

`1` Indicates that zMode/1 should be used. (The dimension variances of rVariables will be preserved.)

`2` Indicates that zMode/2 should be used. (The dimensions of rVariables having a variance of `NOVARY` [false] are removed.)

`/[NO]FORMAT`, `-[no]format`

Specifies whether or not the `FORMAT` attribute is used when writing NRV variable values (if the `FORMAT` attribute exists and an entry exists for the variable).

`/[NO]NEG2POSFP0`, `-[no]neg2posfp0`

Specifies whether or not `-0.0` is converted to `0.0` by the CDF library when read from a CDF. (The CDF is not modified.) `-0.0` is an illegal floating point value on VAXes and DEC Alphas running OpenVMS.

`/REPORT=(<types>)`, `-report "<types>"`

Specifies the types of return status codes from the CDF library that should be reported/displayed. `<types>` is a comma-separated list of zero or more of the following symbols: `errors`, `warnings`, or `informationals`.

Example(s):

VMS:

```
$ SKELETONTABLE/NOLOG/REPORT=(ERRORS) FGGE3B
$ SKELETONTABLE/SKELETON=FGGE3B/NONRV FGGE3B
```

UNIX:

```
% skeletontable -nolog -report "errors" fgge3b
% skeletontable -skeleton fgge3b -nonrv ../cdfs/fgge3b
```

MS-DOS:

```
> skeletontable -nolog -report "errors" fgge3b
> skeletontable -skeleton fgge3b -nonrv c:\temp\fgge3b
```

Help is displayed when `SkeletonTable` is executed without any arguments.

3.8.3 Output from the SkeletonTable Program

The format of the skeleton table is described in Appendix A.

3.9 CDFskeleton

3.9.1 Introduction

The `CDFskeleton` program is used to make a fully structured CDF by reading an ASCII text file called a skeleton table. The skeleton table contains the information necessary to create a CDF that is complete in all respects except for record-variant (RV) variable values. (RV variables vary from record to record.) RV values are then written to the CDF by execution of an application program. The `CDFskeleton` program allows a CDF to be created with the following.

1. The necessary header information — the number of dimensions and dimension sizes for the rVariables, format, data encoding, and variable majority.
2. The gAttribute definitions and any number of gEntries for each.
3. The rVariable and zVariable definitions.

4. The vAttribute definitions and the entries corresponding to each variable.
5. The data values for those variables that are non-record-variant (NRV). NRV variables do not vary from record to record.

The created CDF is often referred to as a skeleton CDF.

3.9.2 Executing the CDFskeleton Program

The command line syntax for the `CDFskeleton` program is as follows:

Usage:

VMS:

```
$ CDFSKELETON [/CDF=<cdf-path>] [/[NO]LOG] [/[NO]DELETE] [/[NO]FILLVAL]
              [/REPORT=(<types>)] <skeleton-path>
```

UNIX:

```
% cdfskeleton [-cdf <cdf-path>] [-[no]log] [-[no]delete] [-[no]fillval]
              [-report "<types>"] <skeleton-path>
```

MS-DOS:[5]

```
> cdfskeleton [-cdf <cdf-path>] [-[no]log] [-[no]delete] [-[no]fillval]
              [-report "<types>"] <skeleton-path>
```

Parameter(s):

`<skeleton-path>`	The pathname of the skeleton table. (Do not specify an extension.) This skeleton table will be read and its information used to create a new CDF.

Qualifier(s):

`/CDF=<cdf-path>`, `-cdf <cdf-path>`

`<cdf-path>` is the pathname of the CDF that will be created (overriding the pathname in the skeleton table). If this qualifier is not specified, the CDF pathname in the skeleton table is used. Do not specify an extension in the pathname.

`/[NO]LOG`, `-[no]log` Specifies whether or not messages are displayed as the program executes.

[5] On MS-DOS systems the executable is named `SKT2CDF.EXE`.

`/[NO]DELETE`, `-[no]delete`

Specifies whether or not the CDF will be deleted first if it already exists (essentially overwriting it).

`/[NO]FILLVAL`, `-[no]fillval`

Specifies whether or not entries of the `FILLVAL` vAttribute are used to set the pad values for the corresponding variables. If this qualifier is specified, the `FILLVAL` vAttribute must exist and only those variables with an entry for the `FILLVAL` vAttribute will be affected.

`/REPORT=(<types>)`, `-report "<types>"`

Specifies the types of return status codes from the CDF library that should be reported/displayed. `<types>` is a comma-separated list of zero or more of the following symbols: `errors`, `warnings`, or `informationals`.

Example(s):

VMS:

```
$ CDFSKELETON FGGE3B
$ CDFSKELETON/NOLOG/CDF=[-.TEMP]FGGE3B_X/REPORT=(ERRORS) FGGE3B
```

UNIX:

```
% cdfskeleton fgge3b
% cdfskeleton -nolog -cdf ../fgge3b_x -report "errors" fgge3b
```

MS-DOS:

```
> cdfskeleton fgge3b
> cdfskeleton -nolog -cdf ..\fgge3b_x -report "errors" a:\fgge3b
```

Help is displayed when `CDFskeleton` is executed without any arguments.

3.9.3 Creating the Skeleton Table

A skeleton table is an ASCII text file having `.skt` as a file extension. The normal method of creating and using a skeleton table would be to use `SkeletonTable` on an existing CDF that is similar to the CDF you want to create. Then edit the created skeleton table to meet your needs, and use `CDFskeleton` to create the new CDF. The skeleton table could also be created from scratch with any text editor.

The format of the skeleton table is described in Appendix A.

3.10 CDFbrowse

`CDFbrowse` has been replaced by `CDFedit` (see Section 3.13).[6]

3.11 CDFlist

3.11.1 Introduction

The `CDFlist` program allows you to selectively list all (or a subset of) the rVariable data values from a CDF. The values can be displayed at the screen, output to a text file, or used to create a new CDF. (The new CDF will most likely be a subset of the original CDF.)

3.11.2 Executing the CDFlist Program

The command line syntax for `CDFlist` is as follows:

Usage:

VMS:

```
$ CDFLIST [/INITIAL=(<defaults>)] [/SPECIFICATION=<cdf-spec>] [<cdf-spec>]
```

UNIX:

```
% cdflist [-initial "<defaults>"] [-specification <cdf-spec>] [<cdf-spec>]
```

MS-DOS:

```
> cdflist [-initial "<defaults>"] [-specification <cdf-spec>] [<cdf-spec>]
```

Parameter(s):

`<cdf-spec>`	The initial specification of the CDF(s) to list. (Do not specify an extension.) This may be either a single CDF pathname or a directory/wildcard path. Wildcards are allowed in the CDF name but not in the directory path. This parameter is optional.

Qualifier(s):

[6] The alias/symbol `CDFbrowse` still exists in the "definitions" file on UNIX/VMS systems but now executes `CDFedit` in a browse-only mode.

`/INITIAL=(<defaults>)`,`-initial "<defaults>"`

The default settings that initially appear on the `Input Options`, `Output Options`, and `Listing Options` windows. Whether or not to automatically select zMode may also be specified. `<defaults>` is a comma separated list of settings consisting of one or more of the following:

`zmode`	When specified, zMode is automatically selected when a CDF is opened.
`[no]erase`	Specifies whether or not to erase the `Variable Selection` window when a new CDF is selected.
`[no]filter`	Specifies whether or not to use the variable filters when generating output.
`[no]fill`	Specifies whether or not to use the `FILLVAL` attribute when filtering.
`[no]format`	Specifies whether or not to use the `FORMAT` when encoding values to be displayed.
`[no]neg2posfp0`	Specifies whether or not to convert `-0.0` to `0.0` when detected.
`single | multi`	Specifies the format of created CDFs.
`host | network`	Specifies the encoding of created CDFs.
`vertical | horizontal`	Specifies the type of text output desired while in zMode.
`epoch` or `epoch1`	Specifies the style in which EPOCH values should be displayed.

These settings are only the settings initially in effect. The user may change any of them at any time. More detailed descriptions of each option may be found in the appropriate sections that follow.

`/SPECIFICATION=<cdf-spec>`,`-specification <cdf-spec>`

A default CDF specification (`<cdf-spec>`) for the CDF name field. The user may choose this specification or enter another.

Example(s):

VMS:

```
$ CDFLIST
$ CDFLIST GISS_WETL
$ CDFLIST NCDS$DATA:FGGE3B
$ CDFLIST/INITIAL=(ZMODE,HORIZONTAL)/SPECIFICATION=NCDS$DATA:
```

UNIX:

```
% cdflist
% cdflist ../cac_sst_blended
% cdflist ~/CDFs
% cdflist -initial "zmode,horizontal" -specification ~/CDFs
```

MS-DOS:

```
> cdflist
> cdflist ..\cac_sst
> cdflist a:\cdfs
> cdflist -initial "zmode,horizontal" -specification a:\cdfs
```

3.11.3 Interaction with CDFlist

The user interface to `CDFlist` is contained on a single screen divided into several fixed windows and pop-up windows. The `Action Menu` controls the execution of the program. This menu allows you to move to a different window, start the data output, browse a help file, or exit the program. You must return to this menu each time you want to select a new variable, output option, listing type, or input CDF. This menu may be displayed at any time by entering the `Action` key. Use the `Up` and `Down` keys to highlight a choice in the `Action Menu` and then enter the `Select` key to execute that function. The `Action Menu` appears in Figure 3.1.

```
Generate Output
Select Input CDF name
Select Output Device
Select Variables
Select Monotonic Variables
Save Variables to file
Restore Variables from file
Select r/z/zMODE Variables
Select CDF Input Options
Select CDF Output Options
Select CDF Listing Options
Help
Exit
```

Figure 3.1: `Action Menu`, `CDFlist`

Two very important windows are the `Keypad Definition` and `Message` windows. They contain information on keyboard functionality (i.e., which keys to use to perform the operations described below) and program status/error messages.

When `CDFlist` is executed (and a CDF and/or directory is not entered on the command line), you must first specify the CDF you wish to list. If you enter a disk/directory/wildcard specification (without a CDF name), all of the CDFs meeting the specification will appear in a separate pop-up window. Use the `Up` and `Down` keys to choose a CDF and then enter the `Select` key to select that CDF.

The `Output` window will then be displayed with the valid choices. Use the `Up` and `Down` keys to move up and down within the window and then enter the `Select` key to select the desired output option. If output to an ASCII text file or CDF is selected, you will be prompted for a file/CDF name. You may enter any file/CDF name valid for the operating system being used or accept the default file/CDF name that appears. Do not enter a file extension. (It is automatically appended.) The file extension for a generated ASCII text file is `.lis`.

The second window (that remains constant on the screen) is the `Variable Selection` window. This window allows the user to select the variables to be listed or put into a CDF along with any desired filters on the variables. You will automatically be placed in this window after you have entered an output option. You can also enter this window by requesting the `Action Menu` and then selecting the `Select Variables` option.

There are four fields on each line in the `Variable Selection` window: variable name, minimum filter value, maximum filter value, and output toggle. You can move forward to the next field by entering the `Next Field` key or backwards to the previous field with the `Prev Field` key. You may also use the `Up` and `Down` keys to move vertically through the fields. When the cursor is positioned in a variable name field, entering the `Pop-up` key will display the variables in the CDF in a pop-up window. Use the arrow keys to highlight a variable and then enter the `Select` key to select it. You can quit the pop-up window without selecting a variable by entering the `Quit` key. You may also use the `Page Up` and `Page Down` keys to traverse through the pop-up window. The first line in this pop-up menu contains `**SELECT ALL**`. Selecting this item causes all of the variables to be selected. The second line in this pop-up window contains `**RECORD NUM**`. This simply gives the user the opportunity to display/filter on the record numbers in the CDF.

After selecting a variable, the cursor is positioned in the minimum filter field. A filter is simply the range of data that you would like to see from the selected variable. If the CDF contains a `VALIDMIN` and `VALIDMAX` attribute entry for the selected variable, these values will be displayed as default minimum/maximum filter values. If just specifying a minimum filter value is sufficient, a maximum filter value need not be entered. You do not have to enter a filter for each selected variable. For a line of output to be written for a given record/indices, each filter must be passed. There are two ways to enter minimum and maximum filters. The first is by simply entering the desired values in the appropriate fields. The second method can only be used for variables that consist of a discrete set of values. This method involves entering the `Pop-up` key (when in the desired filter field), which displays a pop-up window that contains the discrete values for the variable. The desired filter value may then be selected. (The `Page Up` and `Page Down` keys may also be used to traverse through the window.)

The minimum/maximum fields (filters) will initially be highlighted. This indicates that the corresponding filter will be used for a variable (if filter values are present). A filter for a particular variable may also be turned off (without erasing the filter values) by entering the `Filter On/Off` key when on the variable name field. This will toggle off the highlighting of the filter values and indicates that no filtering is to be performed on that variable. Entering the `Filter On/Off` key again toggles filtering back on for that variable. Note that erasing the filter values is another way to turn off filtering for a variable.

The output toggle field simply gives the user the option not to view a variable they have selected. A user may only wish to filter on a variable and not view it. When positioned on this field, the user simply enters the `Out on` key to display the variable and `Out Off` key to not display the variable. Note that the user may also enter `Out On/Off` keys from the variable name field.

When all of the desired variables and filters have been selected, enter the `Action` key to display the `Action Menu`. Selecting `Generate Output` will cause the listing or new CDF to be generated. Note that the amount of time needed to generate the output will in most cases be significantly reduced if the `MONOTON` attribute and corresponding entries are present in the CDF (and the variables being filtered are monotonic). See the

section explaining `Specifying Monotonic Variables` for more information on monotonic variables. When `CDFlist` completes the generation of the output, the `Action Menu` is redisplayed. It may then be used to select a new CDF to list, respecify the output option, select new variables/filters, get on-line help, select special options, or exit the program.

NOTE: On MS-DOS systems a listing may be terminated early by entering the `Quit` key.

When on the `Action Menu` options are available to save/restore the current selections in the `Variable Selection` window. A user may wish to exit the program and upon returning restore the current selections quickly to the window. When the `Save Variables To File` option is selected, the user is prompted for a file name. (Do not enter an extension.) All of the selections in the `Variable Selection` window are stored in that file. A default file name of `cdflist` is available by just entering the `Select` key at the prompt. When the `Restore Variables From File` option is selected, the user is again prompted for a file name. (`cdflist` is again the default.) The selections are restored to the `Variable Selection` window from that file. `CDFlist` will check the current CDF for the existence of each variable (displaying only those variables that do exist).

Another option on the `Action Menu` is `Select r/z/zMODE Variables`. `CDFlist` groups variables in three separate manners: rVariables, zVariables, and zMode. All of the rVariables are contained in one group; zVariables are grouped together by dimensionality; and zMode consists of all of the variables in the CDF with the rVariables being made to appear as zVariables. Variables from only one of these groups at a time may be selected for filtering/output. See Section 2.3 for more information on r/zVariables and Section 2.1.2 for more information on zMode.

On the `Action Menu` three windows of special options are available. Each option brings up a window containing a set of small option selection lists (an example of which is shown in Figure 3.2). The user makes selections by moving the `*` (the current choice) with the `Up` and `Down` keys and moving to the next list by entering the `Select` key. When entering an option selection list, if the current choice is the one desired, just enter the `Select` key to move to the next list. To return back to the `Action Menu`, enter the `Quit` key at any time.

```
(*) Single
( ) Multi
```

Figure 3.2: Option Selection List, `CDFlist`

Selecting `Select CDF Input Options` causes a window to be displayed with the following option selection lists:

1. Whether or not to erase the variables in the `Variable Selection` window each time a new input CDF is selected. If users select a CDF that contains duplicate variables, they may wish to reuse the current selections. If users choose not to erase selected variables, `CDFlist` will check to make sure that the current variables actually exist when a new CDF is selected.

Selecting `Select CDF Output Options` causes a window to be displayed with the following option selection lists:

1. When selecting output to a CDF, whether the file format of the new CDF is single-file or multi-file.
2. When selecting output to a CDF, whether the encoding of the new CDF is network or host.

Selecting `Select CDF Listing Options` causes a window to be displayed with the following option selection lists:

1. The format used when displaying EPOCH values may be selected. (The `FORMAT` attribute is always ignored when displaying EPOCH values.)
2. Whether or not to use the `FORMAT` attribute when displaying values may be selected. (The `FORMAT` attribute and corresponding entries must exist in the CDF.)
3. Whether or not to use the `FILLVAL` attribute when filtering (assuming the `FILLVAL` attribute exists). If the `FILLVAL` attribute is to be used, fill values in the data are ignored when determining whether or not to filter out a set (line) of values (i.e., fill values outside the filter range will appear in the output).
4. Whether or not to ignore all of the filters when generating the output. This is a more convenient way to turn off filtering than to do so manually for each variable on the `Variable Selection` window.
5. Whether or not to convert `-0.0` to `0.0` when detected. Some machines will not accept `-0.0` as a valid value. (`CDFlist` may abort with a system error.)
6. Whether to display values vertically or horizontally while in zMode.

On the `Action Menu` the option `Specify Monotonic Variables` gives the user the opportunity to specify that a variable is monotonic. This is useful when a CDF does not contain the attribute `MONOTON` or has the entries incorrectly specified. When this option is executed, a pop-up window is displayed containing all of the variables in the CDF and their monotonicities. If a `MONOTON` attribute entry exists for a variable, it is set accordingly. An `I` is displayed if the values increase, `D` if the values decrease, and `F` if the variable is not monotonic (or if an entry does not exist). If a variable varies with more than one "dimension", `N/A` is displayed — these variables can never be specified monotonic. Once inside the pop-up window, the user traverses through the window as in any other pop-up window but with one exception — the cursor is positioned to the monotonic value of the variable. The user then may change that value by entering the `INCREASE` key for increasing values, the `DECREASE` key for decreasing values, or the `FALSE` key for not monotonic.

3.12 CDFwalk

3.12.1 Introduction

The `CDFwalk` program allows direct access to all variable data in a CDF. The data values of up to 27 variables may be displayed at once. (The CDF can have more variables but only 27 may be viewed at any one time.) The data value for each variable at a given record/indices location within the CDF conceptual view is displayed. The user can "walk" from one record/indices location to another sequentially (in any "direction") or a particular record/indices location can be "jumped" to directly. Variable values may also be used to directly "jump" to a record/indices location.

3.12.2 Executing the CDFwalk Program

The command line syntax for `CDFwalk` is as follows:

Usage:

VMS:

```
$ CDFWALK [/INITIAL=(<defaults>)] [/SPECIFICATION=<cdf-spec>] [<cdf-spec>]
```

UNIX:

```
% cdfwalk [-initial "<defaults>"] [-specification <cdf-spec>] [<cdf-spec>]
```

MS-DOS:

```
> cdfwalk [-initial "<defaults>"] [-specification <cdf-spec>] [<cdf-spec>]
```

Parameter(s):

`<cdf-spec>` The initial specification of the CDF(s) at which to begin "walking." (Do not specify an extension.) This may be either a single CDF pathname or a directory/wildcard path. Wildcards are allowed in the CDF name but not in the directory path. This parameter is optional.

Qualifier(s):

`/INITIAL=(<defaults>)`,`-initial "<defaults>"`

The default settings that initially appear on the `Special Options` window. `<defaults>` is a comma-separated list of settings consisting of one or more of the following:

`[no]format`	Specifies whether or not to use the `FORMAT` when encoding values to be displayed.
`[no]neg2posfp0`	Specifies whether or not to convert `-0.0` to `0.0` when detected.

These setting are only the settings initially in effect. The user may change any of them at any time. More detailed descriptions of each option may be found in the appropriate sections that follow.

`/SPECIFICATION=<cdf-spec>`,`-specification <cdf-spec>`

A default CDF specification (`<cdf-spec>`) for the CDF name field. The user may choose this specification or enter another.

Example(s):

VMS:

```
$ CDFWALK
$ CDFWALK [-.TEMP]NOAA7
$ CDFWALK/INITIAL=(NOFORMAT,NEG2POSFP0)/SPECIFICATION=NCDS$DATA:
```

UNIX:

```
% cdfwalk
% cdfwalk ../cac_sst
% cdfwalk /disk3/CDFs/ssclimate
% cdfwalk -initial "noformat,neg2posfp0" -specification ~/CDFs
```

MS-DOS:

```
> cdfwalk
> cdfwalk ..\cac_sst
> cdfwalk c:\cdfs
> cdfwalk -initial "noformat,neg2posfp0" -specification c:\cdfs
```

3.12.3 Interaction with CDFwalk

The user interface to `CDFwalk` is contained on a single screen divided into several fixed windows and pop-up windows. The `Action Menu` controls the execution of the program. This menu allows you to "walk" through a CDF, select a new CDF through which to "walk", select r/zVariables, select `CDFwalk` special options, browse on-line help, or exit the program. This menu may be displayed at any time by entering the `Action` key. Use the `Up` and `Down` keys to highlight a choice in the `Action Menu`, and then enter the `Select` key to execute that function. The `Action Menu` appears in Figure 3.3.

```
Walk Thru CDF data
Select Input CDF name
Select Special Options
Select r/z Variables
Help
Exit
```

Figure 3.3: `Action Menu`, `CDFwalk`

Two very important windows are the `Keypad Definition` and `Message` windows. They contain information on keyboard functionality (i.e., which keys to use to perform the operations described below) and program status/error messages.

When `CDFwalk` is executed (and a CDF and/or directory/wildcard specification was not entered on the command line), you must first specify the CDF through which you wish to "walk." If you enter a disk/directory/wildcard specification, all of the CDFs meeting that specification will appear in a separate pop-up window. Use the `Up` and `Down` keys to choose a CDF and then enter the `Select` key to "walk" through that CDF.

After the input CDF is selected, the `Record/Indices` window appears. The size of this window depends on the dimensionality of the variables being displayed (initially the rVariables if any exist). Vertically,

the window will contain one row for the record number and one row for each dimension of the variables. Horizontally, each line of the window contains three fields. The first field contains a variable name; the second field contains a data value for the variable; and the third field contains a record number or dimension index.

The variable selected for the record number line will be one that only varies from record to record (not varying along any of the dimensions). If more than one variable of that type exists, a pop-up window will be displayed containing the list of variables. Select one of the variables, keeping in mind that you will have an opportunity to use it for positioning by value to a desired record. If a variable meeting the requirements mentioned above does not exist, only the record number field will be used.

The variable selected for each dimension line will be one that only varies along the corresponding dimension (and does not vary from record to record). Again, if more than one variable of that type exists in the CDF, a pop-up window will be displayed to allow you to select the variable to use for that dimension. Keep in mind that you will be able to use it to position by value to a desired index along that dimension. If for a particular dimension a variable meeting the requirements mentioned above does not exist, only the dimension index field will be used.

After the `Record/Indices` window has been specified (which will usually occur automatically), the `Variable Values` window will be displayed. It contains up to 27 variable names and the associated data values for the record/indices shown in the `Record/Indices` window.

Initially, the cursor will be positioned at the data value field of the record number line in the `Record/Indices` window (assuming that a variable was found meeting the above requirements; if not, the cursor will start at the record number field). You may now begin to "walk" through the CDF by using the right arrow to increment to and the left arrow to decrement to a new record. For each new record the data for each variable is updated in both the `Record/Indices` and `Variable Values` windows. If the CDF contains many records, you may want to advance quickly to a desired record. You may do this by entering the `Input` key. `CDFwalk` enters an input mode and prompts for the value to which to advance. If the exact value is not in the CDF for the variable, the record number is changed to correspond to the closest possible value. Another method of advancing to a desired value is by entering the `Pop-up` key. This will cause a pop-up window containing all of the values for the variable to be displayed. You may select the value desired as you would with any other pop-up window and `CDFwalk` will automatically position the record number to the corresponding value.

After positioning to a desired record (and if the CDF is multidimensional) you may now wish to "walk" through the dimension indices. This is done by entering the up and down arrow keys to position the cursor to the desired dimension line. "Walking" through dimensions is done in the same way as "walking" through the record numbers.

You may also wish to "walk" through the CDF by using the actual record number or dimension indices rather than data values. (This would also be necessary if variables meeting the above requirements were not found.) This is done by entering the `Next Field` key, which causes the cursor to advance to the next field (i.e., from the data value field to the record number/indices field). Entering the `Back Field` key causes the cursor to move back to the previous field. "Walking" through the CDF using the record number/indices fields is the same as using the data value fields but with a few exceptions. First, no `Pop-up` option is available with these fields. Also, in input mode (after entering the `Input` key) you may input either an absolute record number/index to which to move or a relative offset from the current record number/index. Prefixing the value with a `+/-` will increment/decrement the record number/index by that amount.

If more than 27 variables exist and a particular variable is not currently displayed in the `Variable Values` window, an option is provided to scroll new variables into that window. This is done by entering the `Scroll` key. A small pop-up window is displayed in which you may use the `Next/Prev Variable` keys to scroll

in or scroll out variables. When the desired set of variables is displayed, enter the `Select` key to continue "walking" through the CDF.

When you are finished "walking" through the CDF, enter the `Action` key to display the `Action Menu`. You may input a new CDF, resume "walking" through the current CDF, select special options, read on-line help, or exit the program.

On the `Action Menu` selecting `Special Options` causes a window to be displayed with the following option selection lists:

1. Whether or not to use the `FORMAT` attribute when displaying values may be selected. (The `FORMAT` attribute and corresponding entries must exist in the CDF.)
2. Whether or not to convert `-0.0` to `0.0` when detected. Some machines will not accept `-0.0` as a valid value. (`CDFwalk` may abort with a system error.)

The user makes a selection by moving the `*` (the current choice) with the `Up` and `Down` keys and moving to the next options selection list (if another exists) by entering the `Select` key. An example option selection list is shown in Figure 3.4. To return to the `Action Menu`, enter the `Quit` key at any time.

```
(*) CDF FORMAT attribute
( ) CDFWALK defaults
```

Figure 3.4: Option Selection List, `CDFwalk`

Another option on the `Action Menu` is `Select r/z Variables`. `CDFwalk` groups variables in two separate manners: rVariables and zVariables. All of the rVariables are in one group and the zVariables are grouped together by dimensionality. Only one of these groups may be "walked" through at a time. For more information on the r/zVariables see Section 2.3.

3.13 CDFedit

3.13.1 Introduction

The `CDFedit` program allows the display and/or modification of practically all of the contents of a CDF by way of a full-screen interface. It is also possible to run `CDFedit` in a browse-only mode in order to prevent accidental modifications.[7]

3.13.2 Executing the CDFedit Program

The command line syntax for `CDFedit` is as follows:

Usage:

VMS:

[7] Running `CDFedit` in a browse-only mode provides the same functionality as `CDFbrowse` once did.

```
$ CDFEDIT [/[NO]BROWSE] [/ZMODE=<mode>] [/[NO]FORMAT] [/[NO]PROMPT]
          [/[NO]NEG2POSFP0] [/REPORT=(<types>)] <cdf-spec>
```

UNIX:

```
% cdfedit [-[no]browse] [-zmode <mode>] [-[no]format] [-[no]prompt]
          [-[no]neg2posfp0] [-report "<types>"] <cdf-spec>
```

MS-DOS:

```
> cdfedit [-[no]browse] [-zmode <mode>] [-[no]format] [-[no]prompt]
          [-[no]neg2posfp0] [-report "<types>"] <cdf-spec>
```

Parameter(s):

`<cdf-spec>` The specification of the CDF(s) to edit. (Do not specify an extension.) This may be either a single CDF pathname or a directory/wildcard path. Wildcards are allowed in the CDF name but not in the directory path. If the `/PROMPT`,`-prompt` qualifier is used, ¡cdf-spec¿ will appear as the initial specification at the prompt. If this parameter is omitted, the `/PROMPT`,`-prompt` qualifier must be specified (and the initial specification at the prompt will be the current directory).

Qualifier(s):

`/[NO]BROWSE`, `-[no]browse`

Specifies whether or not a browsing mode is desired. In browsing mode the creation, modification, or deletion of a CDF is not allowed.

`/ZMODE=<mode>`, `-zmode <mode>`

Specifies which zMode should be used. `<mode>` may be one of the following:

- `0` Indicates that zMode should be disabled.
- `1` Indicates that zMode/1 should be used. (The dimension variances of rVariables will be preserved.)
- `2` Indicates that zMode/2 should be used. (The dimensions of rVariables having a variance of `NOVARY` [false] are removed.)

`/[NO]FORMAT`, `-[no]format`

Specifies whether or not the `FORMAT` attribute is used when displaying variable values (if the `FORMAT` attribute exists and an entry exists for the variable).

`/[NO]PROMPT`, `-[no]prompt`

Specifies whether or not a prompt is issued for the CDF(s) specification. When enabled the prompt will be issued both at program startup and after editing the current CDF(s) specification (at which point a new CDF[s] specification may be specified). If a CDF(s) specification is entered at the command line along with this qualifier, the CDF(s) specification will appear at the prompt. (Otherwise, the current directory will appear at the prompt.)

`/[NO]NEG2POSFP0`, `-[no]neg2posfp0`

Specifies whether or not `-0.0` is converted to `0.0` by the CDF library when read from a CDF. (The CDF is not modified.) `-0.0` is an illegal floating point value on VAXes and DEC Alphas running OpenVMS.

`/REPORT=(<types>)`, `-report "<types>"`

Specifies the types of return status codes from the CDF library that should be reported/displayed. `<types>` is a comma-separated list of zero or more of the following symbols: `errors`, `warnings`, or `informationals`.

Example(s):

VMS:

```
$ CDFEDIT [.SAMPLES]
$ CDFEDIT/ZMODE=2/NOFORMAT GISS_WETLX
$ CDFEDIT/BROWSE/PROMPT/REPORT=(ERRORS)
```

UNIX:

```
% cdfedit samples
% cdfedit -zmode 2 -noformat giss_wetl
% cdfedit -browse -prompt -report "errors"
```

MS-DOS:

```
> cdfedit samples
> cdfedit -zmode 2 -noformat giss_wetl
> cdfedit -browse -prompt -report "errors"
```

3.13.3 Interaction with CDFedit

Interaction with `CDFedit` is through a series of menus and windows. Extensive on-line help is provided and will not be repeated here.[8] The on-line help does refer to the sections of a window by name. Figure 3.5 illustrates the various sections of the possible types of windows.

[8] It is our intention that the use of `CDFedit` be as intuitive as possible. You may not even need the on-line help. We're sure you'll let us know.

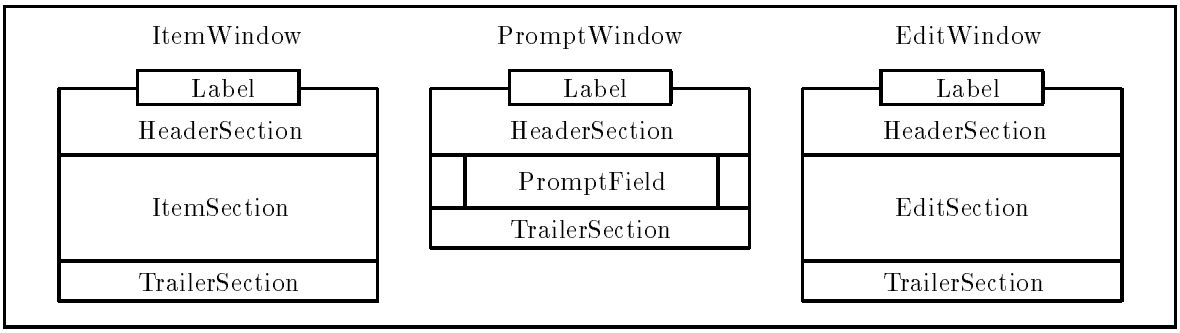

Figure 3.5: Window Sections, `CDFedit`

ItemWindows are used when a choice is to be made from a list of one or more items (e.g., functions to perform, CDFs to edit, variable names, etc.). In some cases the entire list of items may not fit on the screen at once. When this occurs, the ItemSection may be scrolled to display hidden items. Some ItemWindows have a percentage indicator at the bottom right portion of the ItemSection. The percentage indicator shows which part of the ItemSection is being displayed.

PromptWindows are used when a textual response is required (e.g., a CDF specification, a new attribute name, a variable value, etc.). If the text is too long to fit into the PromptField, the "more" indicators ("<" and ">") at the left and right ends of the PromptField will display where hidden characters exist.

EditWindows are used to display/edit a text file or group of lines. EditWindows are currently only used to display on-line help.