`<cdf-path>`	The pathname of the CDF for which to display a directory listing (do not specify an extension).

Example(s):

VMS:

```
$ CDFDIR NCDS$DATA:GISS_WETL_CLIMATOLOGY
$ CDFDIR [-.TEMP]FGGE3B
```

UNIX:

```
% cdfdir ../cac_sst_blended
% cdfdir ~/CDFs/giss_wetl_climatology
```

MS-DOS:

```
> cdfdir ..\cac_sst
> cdfdir c:\cdfs\gisswetl
```

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

3.2.3 Output from the CDFdir Program

The format of the output from `CDFdir` is that of a directory listing on the operating system being used.

3.3 CDFcompare

3.3.1 Introduction

The CDFcompare program displays the differences between two CDFs. More than one pair of CDFs can be compared. This program would be used to verify changes made to a CDF (comparing it with the saved original) or to verify the conversions performed by `CDFconvert` (see Section 3.4).

3.3.2 Executing the CDFcompare Program

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

Usage:

VMS:

```
$ CDFCOMPARE [/[NO]LOG] [/[NO]ATTR] [/[NO]VAR] [/[NO]NUMBER] [/[NO]PERCENT]
             [/[NO]ETC] [/[NO]NEG2POSFP0] [/ZMODES=(<mode1>,<mode2>)]
             [/[NO]LOCATION] [/REPORT=(<types>)] <cdf-spec-1> <cdf-spec-2>
```

UNIX:

```
% cdfcompare [-[no]log] [-[no]attr] [-[no]var] [-[no]number] [-[no]percent]
             [-[no]etc] [-[no]neg2posfp0] [-zmodes "<mode1>,<mode2>"]
             [-[no]location] [-report "<types>"] <cdf-spec-1> <cdf-spec-2>
```

MS-DOS:[1]

```
> cdfcompare [-[no]log] [-[no]attr] [-[no]var] [-[no]number] [-[no]percent]
             [-[no]etc] [-[no]neg2posfp0] [-zmodes "<mode1>,<mode2>"]
             [-[no]location] [-report "<types>"] <cdf-spec-1> <cdf-spec-2>
```

Parameter(s):

`<cdf-spec-1> <cdf-spec-2>`

The specifications of the CDFs to be compared. (Do not enter extensions.) These can be either a pathname specifying a single CDF or a directory/wildcard path specifying more than one CDF. Wildcards are allowed in the CDF name but not in the directory path.

If two directory/wildcard paths are specified, all of the CDFs with matching names will be compared. If a CDF pathname and a directory/wildcard path are specified, the CDF specified will be compared with the CDF in the directory/wildcard path having the same name. If two CDF pathnames are specified, the CDFs are compared. (This is the only way to compare two CDFs having different names.)

Qualifier(s):

`/[NO]LOG`, `-[no]log` Specifies whether or not messages about the progress of each comparison are displayed.

`/[NO]PERCENT`, `-[no]percent`

Specifies whether or not the percentage of a variable's values compared is displayed during the comparison of that variable. Message logging must also be enabled.

`/[NO]ATTR`, `-[no]attr` Specifies whether or not attributes (and their entries) are to be compared.

`/[NO]VAR`, `-[no]var` Specifies whether or not variables are to be compared. Note that an rVariable will never be compared with a zVariable.

[1] On MS-DOS systems the executable is named `CDFCMP.EXE`.

/[NO]NUMBER, -[no]number

Specifies whether or not numbering differences between attributes with the same names and between variables with the same names are to be displayed.

/[NO]ETC, -[no]etc

Specifies whether or not differences transparent to an application will be displayed. These would consist of the version/release/increment of the creating CDF library, format, encoding, etc.

/ZMODES=(<mode1>,<mode2>), -zmodes "<mode1>,<mode2>"

Specifies the zModes that should be used with the CDF(s) being compared. <mode1> applies to the first CDF specification and <mode2> applies to the second CDF specification. The zModes may each 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]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.

/[NO]LOCATION, -[no]location

Specifies whether or not the locations of variable value differences are displayed. The locations are displayed in the form
<record-number>:[<index1>,<index2>,...,<indexN>]

/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:

```
$ CDFCOMPARE GISS_WETL GISS_WETL1
$ CDFCOMPARE/LOG/NOATTR/NUMBER/REPORT=(ERRORS) GISS_WETL CDF$SMPL:GISS_WETL
$ CDFCOMPARE/NOVAR/NOETC/ZMODES=(1,2) NCDS$SMPL: NCDS$DATA:
```

UNIX:

```
% cdfcompare giss_wetl giss_wetl1
```

```
% cdfcompare -log -noattr -number -report "errors" giss_wetl ../giss_wetlx
% cdfcompare -novar -noetc -zmodes "1,2" /user5/CDFs /user6/CDFs
```

MS-DOS:

```
> cdfcompare gisswetl c:\gisswetl
> cdfcompare -log -noattr -number -report "errors" gisswetl ..\..\giswetlx
> cdfcompare -novar -noetc -zmodes "1,2" a:\cdfs c:\cdf\cdfs
```

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

3.3.3 Output from the CDFcompare Program

The output from `CDFcompare` consists of messages indicating the differences found. If message logging is enabled, the progress of each comparison is also displayed.

3.4 CDFconvert

3.4.1 Introduction

The `CDFconvert` program is used to convert various properties of a CDF. In all cases new CDFs are created. (Existing CDFs are not modified.) Any combination of the following properties may be changed when converting a CDF.

1. A CDF created with an earlier release of the CDF library may be converted to the current library release.
2. The format of the CDF may be changed (see Section 2.2.6).
3. The data encoding of the CDF may be changed (see Section 2.2.7).
4. The variable majority of the CDF may be changed (see Section 2.3.11).

3.4.2 Executing the CDFconvert Program

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

Usage:

VMS:

```
$ CDFCONVERT [/SKELETON=<cdf-path>] [/[NO]LOG]
             [/[NO]PERCENT] [/REPORT=(<types>)]
```

```
                <src-cdf-spec> [/ZMODE=<mode>] [/[NO]NEG2POSFP0]
                <dst-cdf-spec> [/SINGLE | /MULTI] [/ROW | /COLUMN]
                               [/HOST | /NETWORK] [/[NO]DELETE]
```

UNIX:

```
% cdfconvert [-skeleton <cdf-path>] [-[no]log]
            [-[no]percent] [-report "<types>"]
            <src-cdf-spec> [-zmode <mode>] [-[no]neg2posfp0]
            <dst-cdf-spec> [-single | -multi] [-row | -column]
                           [-host | -network] [-[no]delete]
```

MS-DOS:[2]

```
> cdfconvert [-skeleton <cdf-path>] [-[no]log]
             [-[no]percent] [-report "<types>"]
             <src-cdf-spec> [-zmode <mode>] [-[no]neg2posfp0]
             <dst-cdf-spec> [-single | -multi] [-row | -column]
                            [-host | -network] [-[no]delete]
```

Parameter(s):

`<src-cdf-spec>` The source CDF(s). This can be either a single CDF pathname or a directory/wildcard path in which case all CDFs that match the specification will be converted. Wildcards are allowed in the CDF name but not in the directory path. In either case do not specify an extension.

`<dst-cdf-spec>` The destination of the converted CDF(s). This may be a single CDF pathname only if `<src-cdf-spec>` is a single CDF pathname. If the directory paths are the same, then a different CDF name must be specified. If `<src-cdf-spec>` is a directory/wildcard path, then this must be a directory path (other than the source directory path). This may also be a directory path if only a single CDF is being converted. In any case do not specify an extension.

Qualifier(s):

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

`<cdf-path>` is the pathname of a skeleton CDF to be used during the conversions. (Do not enter an extension.) The skeleton CDF is used in the following cases:

1. If an assumed attribute scope is found in a source CDF, then the scope of that attribute in the skeleton CDF (if the attribute exists) will be used in the destination CDF.

[2] On MS-DOS systems the executable is named `CDFCVT.EXE`.

2. If a format for the destination CDF was not specified, then the format of the skeleton CDF will be used.
3. If a majority for the destination CDF was not specified, then the majority of the skeleton CDF will be used.
4. If an encoding for the destination CDF was not specified, then the encoding of the skeleton CDF will be used.

Specifying a skeleton CDF is optional.

`/[NO]LOG`, `-[no]log` Specifies whether or not messages about the progress of each conversion are displayed.

`/[NO]PERCENT`, `-[no]percent`

Specifies whether or not the percentage of a variable's values converted is displayed during the conversion of that variable. Message logging must also be enabled.

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

Specifies whether or not a destination CDF is deleted if it already exists.

`/SINGLE`, `-single` or `/MULTI`, `-multi`

The format of the destination CDF(s). This overrides the format of the skeleton CDF (if specified). If neither this qualifier nor a skeleton CDF is specified, then the format of a destination CDF will be the same as that of the source CDF.

`/ROW`, `-row` or `/COLUMN`, `-column`

The majority of the destination CDF(s). This overrides the majority of the skeleton CDF (if specified). If neither this qualifier nor a skeleton CDF is specified, then the majority of a destination CDF will be the same as that of the source CDF.

`/HOST`, `-host` or `/NETWORK`, `-network`

The encoding of the destination CDF(s). This overrides the encoding of the skeleton CDF (if specified). If neither this qualifier nor a skeleton CDF is specified, then the encoding of a destination CDF will be the same as that of the source CDF.

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

Specifies the zMode that should be used with the source CDF(s). `<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.)

Note that using zMode/1 or zMode/2 on a source CDF that contains

rVariables will produce a destination CDF containing only zVariables. The zMode "view" provided for the source CDF is written to the destination CDF during the conversion.

`/[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:

```
$ CDFCONVERT CDF$SMPL:TEMPLATE0 TEMPLATE0X
$ CDFCONVERT/LOG/REPORT=(ERRORS) CDF$SMPL: USER_DISK:[USER.CDF]
$ CDFCONVERT CAC_SST_BLENDED CAC_SST_BLENDEDX/SINGLE/NETWORK
$ CDFCONVERT/SKELETON=CDF$SMPL:TEMPLATE3 CAC_SST_BLENDED* [USER.CDF]
```

UNIX:

```
% cdfconvert ../samples/template0 template0x
% cdfconvert -log -report "errors" ../samples /disk4/user/cdf
% cdfconvert cac_sst_blended cac_sst_1 -single -network
% cdfconvert -skeleton template3 '../cdf/cac_sst*' ~user/cdf
```

MS-DOS:

```
> cdfconvert ..\samples\tplate0 tplate0x
> cdfconvert -log -report "errors" ..\samples c:\dir4\user\cdf
> cdfconvert cac_sst cac_sst1 -single -network
> cdfconvert -skeleton tplate3 a:\cdf\cac_sst dir5\cdf
```

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

3.4.3 Output from the CDFconvert Program

As `CDFconvert` executes, the name of each CDF being converted is displayed. If message logging is enabled, the progress of each conversion is also displayed.

3.5 CDFscope

3.5.1 Introduction

The `CDFscope` program is used to correct assumed attribute scopes found in a CDF. Section 2.4.3 explains attribute scopes and how assumed scopes can arise.

NOTE: This program is not a permanent one — it may be deleted from a future release of the CDF distribution. Version 1 CDFs should be converted to Version 2 CDFs as soon as possible. Any assumed attribute scopes should then be corrected using this program.

3.5.2 Executing the CDFscope Program

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

Usage:

VMS:

```
$ CDFSCOPE [/[NO]CONFIRM] [/[NO]LOG] [/REPORT=(<types>)] <cdf-path>
```

UNIX:

```
% cdfscope [-[no]confirm] [-[no]log] [-report "<types>"] <cdf-path>
```

MS-DOS:

```
> cdfscope [-[no]confirm] [-[no]log] [-report "<types>"] <cdf-path>
```

Parameter(s):

`<cdf-path>` The pathname of the CDF to be corrected. (Do not specify an extension.)

Qualifier(s):

`/[NO]CONFIRM, -[no]confirm`

Specifies whether or not `CDFscope` prompts for confirmation before correcting a scope. The valid responses to the confirmation prompts are as follows:

`Y/y` Yes, correct the scope (assumed to definite).

`N/n` No, don't correct the scope.

`G/g` Correct the scope to global.

`V/v` Correct the scope to variable.

`/[NO]LOG, -[no]log` Specifies whether or not message logging is enabled.

`/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:

```
$ CDFSCOPE DUA5:[CDF.TEST]TOGA_ECMWF
$ CDFSCOPE/CONFIRM/NOLOG/REPORT=(ERRORS) TOGA_ECMWF
```

UNIX:

```
% cdfscope /disk5/cdf/toga_ecmwf
% cdfscope -confirm -nolog -report "errors" toga_ecmwf
```

MS-DOS:

```
> cdfscope a:\dir5\cdf\toga
> cdfscope -confirm -nolog -report "errors" toga
```

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

3.5.3 Output from the CDFscope Program

If message logging is enabled, the number of scopes corrected will be displayed.

3.6 CDFstats

3.6.1 Introduction

The `CDFstats` program produces a statistical report on a CDF's variable data. Both rVariables and zVariables are analyzed. For each variable it determines the actual minimum and maximum values (in all of the variable records), the minimum and maximum values within the valid range (if the `VALIDMIN` and `VALIDMAX` vAttributes and corresponding entries are present in the CDF), and the monotonicity. An option exists to allow fill values (specified by the `FILLVAL` vAttribute) to be ignored when collecting statistics.

Monotonicity refers to whether or not variable data values increase or decrease from record to record or along a dimension. This property is checked only if the variable varies along just one "dimension" (considering records to be another "dimension"). For example, consider a CDF with the 2-dimensional rVariables shown in Table 3.1.

rVariable	Record Variance	Dimension Variances
EPOCH	VARY	NOVARY,NOVARY
LATITUDE	NOVARY	VARY,NOVARY
LONGITUDE	NOVARY	NOVARY,VARY
ELEVATION	NOVARY	VARY,VARY
TEMPERATURE	VARY	VARY,VARY

Table 3.1: Example rVariables, CDFstats Monotonicity Checking

The EPOCH, LATITUDE, and LONGITUDE rVariables would be checked for monotonicity but the ELEVATION and TEMPERATURE rVariables would not be checked.

An additional option exists that allows the VALIDMIN, VALIDMAX, SCALEMIN, and SCALEMAX attribute entries to be updated based on the actual minimum and maximum values found for a variable. The MONOTON attribute entries may also be updated based on the monotonicity found for a variable.

3.6.2 Executing the CDFstats Program

The command line syntax for CDFstats is as follows:

Usage:

VMS:

```
$ CDFSTATS [/[NO]RANGE] [/[NO]FILL] [/OUTPUT=<file-path>] [/[NO]FORMAT]
           [/[NO]PAGE] [/[NO]UPDATE_VALIDS] [/[NO]UPDATE_SCALES]
           [/[NO]UPDATE_MONOTONIC] [/ZMODE=<mode>] [/[NO]NEG2POSFP0]
           [/REPORT=(<types>)] <cdf-path>
```

UNIX:

```
% cdfstats [-[no]range] [-[no]fill] [-output <file-name>] [-[no]format]
           [-[no]page] [-[no]update_valids] [-[no]update_scales]
           [-[no]update_monotonic] [-zmode <mode>] [-[no]neg2posfp0]
           [-report "<types>"] <cdf-path>
```

MS-DOS:

```
> cdfstats [-[no]range] [-[no]fill] [-output <file-name>] [-[no]format]
           [-[no]page] [-[no]update_valids] [-[no]update_scales]
           [-[no]update_monotonic] [-zmode <mode>] [-[no]neg2posfp0]
           [-report "<types>"] <cdf-path>
```

Parameter(s):

`<cdf-path>`	The pathname of the CDF to check. (Do not specify an extension.)

Qualifier(s):

`/[NO]RANGE`, `-[no]range` Specifies whether or not range checking will be performed. To perform range checking, the CDF must contain `VALIDMIN` and `VALIDMAX` attributes. A variable must also have an entry for each of these attributes in order for range checking to be performed on that variable.

`/[NO]FILL`, `-[no]fill` Specifies whether or not fill values are ignored when collecting statistics (using the `FILLVAL` attribute if it exists).

`/OUTPUT=<file-path>`, `-output <file-path>`

Redirects the output to a file. The file created will be named `<file-path>`. (If `<file-path>` does not have an extension, `.sts` is appended automatically.) If this qualifier is not specified, the output is displayed at the terminal.

`/[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]PAGE`, `-[no]page` Specifies whether or not the output is displayed a page (screen) at a time. If the output is directed to a file, paging is not allowed.

`/[NO]UPDATE_VALIDS`, `-[no]update_valids`

Specifies whether or not the `VALIDMIN` and `VALIDMAX` attribute entry values are updated for each variable based on the actual minimum and maximum values found (with fill values being ignored if requested). If the `VALIDMIN` and `VALIDMAX` attributes do not exist, they are created.

`/[NO]UPDATE_SCALES`, `-[no]update_scales`

Specifies whether or not the `SCALEMIN` and `SCALEMAX` attribute entry values are updated for each variable based on the actual minimum and maximum values found (with fill values being ignored if requested). If the `SCALEMIN` and `SCALEMAX` attributes do not exist, they are created.

`/[NO]UPDATE_MONOTONIC`, `-[no]update_monotonic`

Specifies whether or not the `MONOTONIC` attribute entry values are updated for each variable based on the monotonicity found (with fill values being ignored if requested). If the `MONOTONIC` attribute does not exist, it is created.

`/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]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:

```
$ CDFSTATS TEST1
$ CDFSTATS/REPORT=(ERRORS) GISS_SOIL
$ CDFSTATS/NOFILL/OUTPUT=TEMPLATE3/NORANGE CDF$SMPL:TEMPLATE3
```

UNIX:

```
% cdfstats giss_soil
% cdfstats -range -fill -report "errors" $CDF_SMPL/giss_soil
% cdfstats -norange -output template3 ../../samples/template3
```

MS-DOS:

```
> cdfstats gisssoil
> cdfstats -range -nofill -report "errors" a:\cdfs\gisssoil
> cdfstats -norange -output tplate3 ..\..\samples\tplate3
```

Help is displayed when CDFstats is executed without any arguments.

3.6.3 Output from the CDFstats Program

The format of the output from CDFstats is as follows:

For each variable (rVariables and zVariables),

```
<number>.  <name>  <n-dims>:[<dim-sizes>]  <rec-vary>/<dim-varys>
```

```
                                                    (<data-type>, <n-elems>)

          min: <min-value>
min in range: <min-value-in-range>
   valid min: <valid-min>, <low-values> low value(s)

          max: <max-value>
max in range: <max-value-in-range>
   valid max: <valid-max>, <high-values> high value(s)

  fill value: <fill-value>, <fill-values> fill value(s)

   monotonic: <monotonicity>
```

If range checking and/or fill value filtering is disabled, the corresponding fields will not be displayed. The fields are defined as follows:

`<number>`	The variable number.
`<name>`	The variable name.
`<rec-vary>`	The record variance of the variable — either a `T` or `F`.
`<dim-varys>`	The dimension variances of the variable — for each dimension either a `T` or `F`. This field is not present if there are zero (0) dimensions.
`<data-type>`	The data type of the variable (e.g., `CDF_REAL4`).
`<n-elems>`	The number of elements of the variable's data type.
`<n-dims>`	The number of dimensions of a zVariable. This field is not present for an rVariable.
`<dim-sizes>`	The dimension sizes of a zVariable. This field is not present for an rVariable or if the zVariable has zero (0) dimensions.
`<min-value>`	The minimum value found (regardless of any range checking performed).
`<min-value-in-range>`	The minimum value found within the valid range.
`<valid-min>`	The minimum valid value (`VALIDMIN` attribute entry value).
`<low-values>`	The number of values found that are less than the valid minimum.
`<max-value>`	The maximum value found (regardless of any range checking performed).
`<max-value-in-range>`	The maximum value found with the valid range.
`<valid-max>`	The maximum valid value (`VALIDMAX` attribute entry value).
`<high-values>`	The number of values found that are greater than the valid maximum.
`<fill-value>`	The fill value (`FILLVAL` attribute entry value).

`<fill-values>`	The number of fill values found.
`<monotonicity>`	The monotonicity of the variable.

The `<monotonicity>` field may take on one of the following values.

`Steady (one value)`	The variable has only one value in the CDF.
`Steady (all values the same)`	All values of the variable are the same.
`Increase`	Values strictly increase (with increasing record number/dimension index).
`Decrease`	Values strictly decrease (with increasing record number/dimension index).
`noDecrease (some values the same)`	Consecutive values either increase or are the same (with increasing record number/dimension index).
`noIncrease (some values the same)`	Consecutive values either decrease or are the same (with increasing record number/dimension index).
`False`	Consecutive values both increase and decrease.
`n/a`	The variable was not checked for monotonicity because it varies along more than one "dimension" (if records are considered another "dimension").

3.7 CDFinquire

3.7.1 Introduction

The `CDFinquire` program produces a report on a CDF's contents. It may be used to inquire the following.

1. The version, release, and increment numbers of the creating CDF library.
2. The copyright notice of the creating CDF library.
3. Header information — the CDF's data encoding, variable majority, number of records, number of gAttributes and vAttributes, number of variables (rVariables and zVariables), and number of dimensions and dimension sizes for the rVariables.
4. Global scope attribute definitions — the name, number, and number of entries for each gAttribute.
5. Global scope attribute entries — for each gAttribute entry, the data type, number of elements, and value(s).
6. Variable scope attribute definitions — the name, number, and number of entries for each vAttribute.
7. Variable definitions — the name, number, data type, number of elements, record variance, and dimension variances for each variable. For zVariables the number of dimensions and dimension sizes are also inquired.

8. Variable scope attribute entries — for each vAttribute entry, the data type, number of elements, and value(s).
9. Variable values.

3.7.2 Executing the CDFinquire Program

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

Usage:

VMS:

```
$ CDFINQUIRE [/DISPLAY=<mask>] [/OUTPUT=<file-path>] [/[NO]PAGE] [/ID]
             [/VARIABLES=<set>] [/ZMODE=<mode>] [/[NO]NEG2POSFP0]
             [/[NO]FORMAT] [/REPORT=(<types>)] <cdf-path>
```

UNIX:

```
% cdfinquire [-display <mask>] [-output <file-path>] [-[no]page] [-id]
             [-variables <set>] [-zmode <mode>] [-[no]neg2posfp0]
             [-[on]format] [-report "<types>"] <cdf-path>
```

MS-DOS:[3]

```
> cdfinquire [-display <mask>] [-output <file-path>] [-[no]page] [-id]
             [-variables <set>] [-zmode <mode>] [-[no]neg2posfp0]
             [-[on]format] [-report "<types>"] <cdf-path>
```

Parameter(s):

`<cdf-path>` The pathname of the CDF to inquire. (Do not specify an extension.)

Qualifier(s):

`/DISPLAY=<mask>`, `-display <mask>`

The display mask indicating the type of inquiry to be done. `<mask>` consists of one or more of the following:

- `i` CDF copyright and version/release/increment.
- `h` Header information.
- `g` gAttribute definitions.

[3] On MS-DOS systems the executable is named `CDFINQ.EXE`.

`g+` gAttribute definitions and entries.

`a` vAttribute definitions.

`r` rVariable definitions.

`r+` rVariable definitions and vAttribute rEntries.

`z` zVariable definitions.

`z+` zVariable definitions and vAttribute zEntries.

`v` rVariable/zVariable values. The `/VARIABLES`,`-variables` qualifier may be used to select the variables for which values are to be displayed.

Do not specify both `g` and `g+`, both `r` and `r+`, or both `z` and `z+`.

`/OUTPUT=<file-path>`, `-output <file-path>`

Redirects the output to a file. The file created will be named `<file-path>`. (If `<file-path>` does not have an extension, `.inq` is appended automatically.) If this qualifier is not specified, the output is displayed at the terminal.

`/[NO]PAGE`, `-[no]page` Specifies whether or not the output is displayed a page (screen) at a time. If the output is directed to a file, paging is not allowed.

`/VARIABLES=<set>`, `-variables <set>`

Specifies the set of variables (rVariables and/or zVariables) for which values should be displayed. `<set>` may be one of the following:

`all` All of the variables.

`nrv` Only those variables that are non-record variant (NRV, a record variance of `NOVARY`).

`novary+` Those variables for which one or more record/dimension variances are NOVARY.

`(<name1>,<name2>,...,<nameN>)`

A named set of variables. Each name must be delimited with a unique character. The names are separated by commas and enclosed in parentheses. **NOTE:** On VMS and OpenVMS systems the entire set must be enclosed in double quote marks, and double quote marks should not be used as delimiters [e.g., `"('LATITUDE','SST')"`]. **NOTE:** On UNIX systems the entire set must be enclosed in single quote marks and single quote marks should not be used as delimiters [e.g., `'("LATITUDE","SST")'`]. **NOTE:** On MS-DOS systems do not use double quote marks as delimiters.

Note that "`v`" must be present in the display mask for this qualifier to have an effect.

`/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 displaying 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`.

`/ID,-id` Causes the version of your CDF distribution and the default toolkit qualifiers to be displayed. All other parameters/qualifiers are ignored.

Example(s):

VMS:

```
$ CDFINQUIRE GISS_VEG
$ CDFINQUIRE/DISPLAY=ihzv/OUTPUT=GISS_VEG/REPORT=(ERRORS) GISS_VEG
$ CDFINQUIRE/ID
```

UNIX:

```
% cdfinquire giss_veg
% cdfinquire -display ihzv -output giss_veg -report "errors" giss_veg
% cdfinquire -id
```

MS-DOS:

```
> cdfinquire giss_veg
> cdfinquire c:\cdfs\giss\giss_veg
> cdfinquire -display ihzv -output ..\giss_veg -report "errors" giss_veg
```

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

3.7.3 Output from the CDFinquire Program

The format of the output from `CDFinquire` is as follows (assuming a display mask of `ihg+ar+z+v`):

```
INFORMATION BLOCK
-----------------

<copyright>

Created by CDF library version: <version>.<release>.<increment>


HEADER BLOCK
------------

  CDF name.............. <cdf-name>
  Encoding.............. <encoding>
  Majority.............. <majority>
  Format................ <format>
  rRecords.............. <num-r-records>
  Attributes............ <num-attributes> (<num-global-scope> global scope,
                                          <num-variable-scope> variable scope)
  rVariables............ <num-r-variables>
  rDimensions........... <num-dimensions>, sizes: [<dimension-sizes>]
  zVariables............ <num-z-variables>


GLOBAL SCOPE ATTRIBUTES
-----------------------

  <attr-num>.  Name...... <attr-name>
               Entries... <num-g-entries>

                  Number...... <entry-num>
                  Data type... <entry-data-type> (<entry-num-elems> elements)
                  Value(s).... <entry-value(s)>

                     .
                     .
                     .

      .
      .
      .
```

```
VARIABLE SCOPE ATTRIBUTES
-------------------------

  <attr-num>.  Name.............. <attr-name>
               rEntries.......... <num-r-entries>
               zEntries.......... <num-z-entries>
       .
       .
       .


rVARIABLES
----------

  <var-num>.  Name........ <var-name>
              Data type... <var-data-type> (<var-num-elems> elements)
              Variances... <rec-variance>/<dim-variances>

                 Attribute name... <attr-name>
                 Data type........ <entry-data-type> (<entry-num-elems> elements)
                 Entry value(s)... <entry-value(s)>

                    Values... (<num-records> records)

                              <record-num>:[<indices>] = <var-value>
                              <record-num>:[<indices>] = <var-value>
                                  .
                                  .
                                  .
                              <record-num>:[<indices>] = <var-value>

                       .
                       .
                       .

    .
    .
    .

zVARIABLES
----------

  <var-num>.  Name......... <var-name>
              Data type.... <var-data-type> (<var-num-elems> elements)
              Dimensions... <num-dimensions>, sizes: <dimension-sizes>
              Variances.... <rec-variance>/<dim-variances>

                 Attribute name... <attr-name>
                 Data type........ <entry-data-type> (<entry-num-elems> elements)
                 Entry value(s)... <entry-value(s)>
```

```
                    Values... (<num-records> records)

                              <record-num>:[<indices>] = <var-value>
                              <record-num>:[<indices>] = <var-value>
                                   .
                                   .
                                   .
                              <record-num>:[<indices>] = <var-value>

                    .
                    .
                    .

     .
     .
     .
```

Each field is defined as follows:

`<attr-name>`	The attribute name.
`<attr-num>`	The attribute number.
`<cdf-name>`	The name of the CDF.
`<copyright>`	The CDF copyright notice.
`<dim-variances>`	The variable dimension variances. For each dimension a `T` if values vary along that dimension (`VARY`) or an `F` if they do not (`NOVARY`). Dimension variances are described in Section 2.3.10. If there are zero (0) dimensions, this field is omitted.
`<dimension-sizes>`	The dimension sizes for the rVariables or a zVariable. If there are zero (0) dimensions, this field is omitted.
`<encoding>`	The data encoding of the CDF. Data encodings are described in Section 2.2.7.
`<entry-data-type>`	The attribute entry data type. Data types are described in Section 2.2.9.
`<entry-num>`	The attribute entry number.
`<entry-num-elems>`	The number of elements of the data type. The number of elements for an attribute entry is explained in Section 2.4.4.
`<entry-value(s)>`	The attribute entry value(s).
`<format>`	The format of the CDF. CDF formats are described in Section 2.2.6.
`<increment>`	The increment number of the CDF library that created/modified the CDF.
`<indices>`	The dimension indices separated by commas. An index for which the corresponding dimension variance is FALSE (`NOVARY`) will be