Table 1.3 Sample Data Objects in an HDF File

Tag	Ref	Data
FID	1	file identifier: user-assigned title for file
FD	1	file descriptor: user-assigned block of text describing overall file contents
IP8	1	image palette (768 bytes)
ID8	1	x and y dimensions of the 2D arrays that contain the raster images (4 bytes)
RI8	1	first 2D array of raster image pixel data (x*y bytes)
RI8	2	second 2D array of pixel data (also x*y bytes)

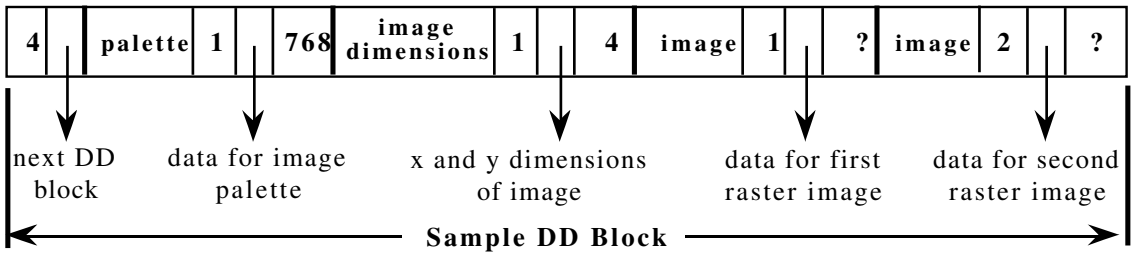
Assuming, for example, that the size of a DD block is 10 DDs, the physical organization of the contents of the file might be described as shown in Figure 1.5.

Figure 1.5 Physical Representation of Data Objects

Offset	Contents
0	FH
4	DDH (5 0L)
10	DD (FID 1 130 4)
22	DD (FD 1 134 41)
34	DD (IP8 1 175 768)
46	DD (ID8 1 943 4)
58	DD (RI8 1 947 240000)
70	DD (RI8 2 240947 240000)
82	DD (empty)
94	DD (empty)
106	DD (empty)
118	DD (empty)
130	"sw3"
134	"solar wind simulation: third try. 8/8/88"
175	<data for the image palette>
943	<data for the image dimensions>: 400, 600
947	<data for the first raster image>
240947	<data for the second raster image>

In this instance, the file contains two raster images. The two images have the same dimensions and are to be used with the same palette. So, the same data objects for the palette (IP8) and dimension record (ID8) can be used with both images.

Figure 1.4 Sample Data Descriptor Block



Physical Organization of HDF Files

Physically, the file header, DD blocks, and data elements are organized as follows. The file header is followed by the first DD block, which is followed by data elements and, if necessary, more DD blocks. These relationships are summarized in Table 1.2.

There are no rules governing the distribution of DD blocks and data elements within a file, except that the first DD block must follow immediately after the header. The pointers in the DD headers connect the DD blocks in a linked list, and the offsets in the individual DDs connect the DDs to the data elements. Beyond this basic structure there is no necessary ordering among the objects in an HDF file, although there are guidelines that you are encouraged to follow. More information regarding these guidelines is presented in Chapter 4, “HDF Conventions.”

Table 1.2 Summary of the Relationships among Parts of an HDF File

Part	Constituents
HDF-file	FH, DD-block, data, DD-block, data, DD-block, data...
FH	^N ^C ^S ^A [32 bits]
DD-block	DDH, DD, DD, DD...
DDH	number-of-DDs [16 bits], offset-to-next-DD block [32 bits]
DD	tag [16 bits], ref [16 bits], offset [32 bits], length [32 bits]

Sample HDF File

Consider an HDF file that contains two 400-by-600 8-bit raster images. Typically, such a file might contain the objects described in Table 1.3.

Data Offset and Length

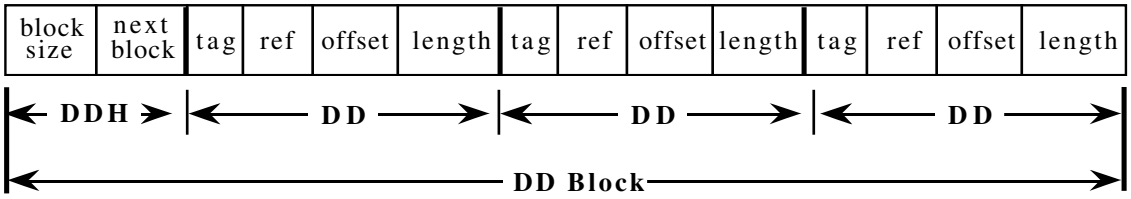
The data *offset* reflects the byte position of the corresponding data element from the start of the file. The *length* gives the number of bytes occupied by the data element. Offset and length are both 32-bit unsigned integers.

DD Blocks

Data descriptors are stored physically in a linked list of blocks called *data descriptor blocks*, or DD blocks. The individual components of a data descriptor block are depicted in Figure 1.3. All of the DDs in a DD block are assumed to contain significant data unless they have a tag that is equal to NULL (no data).

In addition to its DDs, each data descriptor block has a *data descriptor header* (DDH). The DDH has two fields—a *block size* field and a *next block* field. The block size field is a 16-bit unsigned integer that indicates the number of DDs in the following DD block. The next block field is 32-bit unsigned integer giving the offset of the next DD block, if there is one. The last DDH in the list contains a 0 in its next block field.

Figure 1.3 Model of a Data Descriptor Block



Data Element

A *data element* is the raw data part of a data object. Its basic data type is determined by its tag, but other interpretive information may be required before it can be processed properly.

Each data element is stored as a set of contiguous bytes starting at the offset given in the corresponding DD (see Figure 1.4).

Figure 1.2 and are briefly described in Table 1.1. Explanations of each part appear in the paragraphs following Table 1.1.

Figure 1.2 A Data Descriptor (DD)

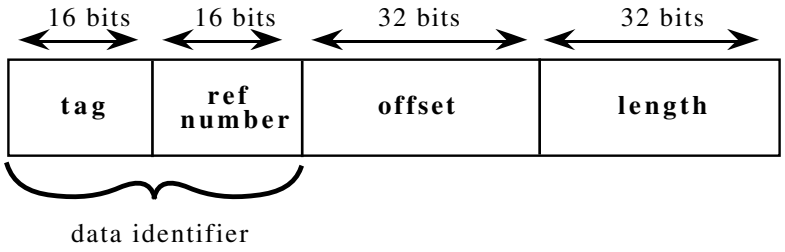


Table 1.1 Parts of a Data Descriptor

Part	Description
tag	designates the type of data in a data element
reference number	uniquely distinguishes corresponding data element from others with the same tag
data identifier	tag/ref; uniquely identifies data element
offset	byte offset of corresponding data element
length	length of data element

Tag

A *tag* is the part of a data descriptor that tells what kind of data is contained in the corresponding data element. A tag is actually a 16-bit unsigned integer between 1 and 65535, but every tag is also usually given a name that programs can refer to instead of the number. If a DD has no corresponding data element, the value of its tag is *no data* (NULL). A tag may never be zero.

The extensibility of HDF results from the fact that new tags can be assigned when it becomes necessary to store new types of data elements. Tags are assigned by NCSA as part of the specification of HDF. Appendix A contains full specifications for all currently supported NCSA HDF tags.

As NCSA HDF grows, the number of tags will grow. In addition to the tags that are defined in this document, some tags are reserved for experimentation and some are delegated to other individuals or institutions in "round" intervals of 100's, 1000's, or 10,000's. All numbers that are not already designated are reserved for future definition by NCSA. Appendix A, "NCSA HDF Tags," contains the current number assignments.

Reference Number

For each occurrence of a tag in an HDF file, a unique *reference number* is stored with the tag in the data descriptor. Reference numbers are 16-bit unsigned integers.

Data Identifier

The combination of a tag and its reference number uniquely identifies the corresponding data object in the file. For this reason, the tag/ref combination is sometimes referred to as a *data identifier.*

Chapter Overview

This chapter introduces and describes the components and organization of Hierarchical Data Format files.

File Header

The first component of an HDF file is the *file header (FH)*, which takes up the first four bytes in an HDF file. The file header is a signature that indicates that the file is an HDF file. Specifically, it is the 32-bit magic number formed by the four characters ^N, ^C, ^S, and ^A (hexadecimal value 0e031301).

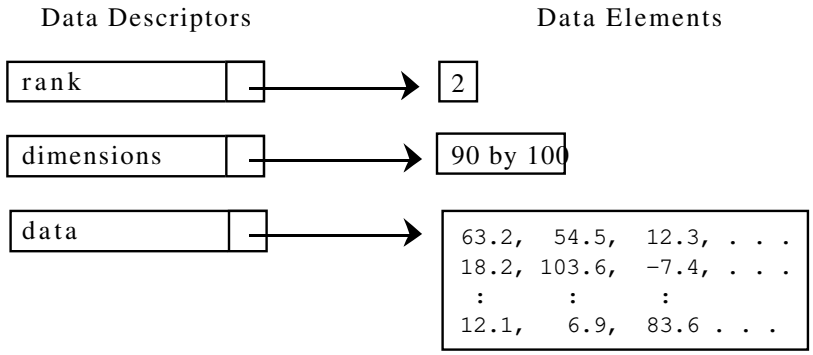
NOTE: On some machines the order of bytes in the file header might be swapped when the header is written to an HDF file, causing these characters to be written in the wrong order. To maintain machine portability when developing software for such machines, you should counteract this byte-swapping by making sure the characters are read and written in the exact order shown.

Data Object

The basic building block in an HDF file is the *data object*, which contains both data and information about the data. A data object has two parts: a 12-byte *data descriptor* (DD) and a *data element*. Figure 1.1 shows three examples of data objects.

As the names imply, the data descriptor gives information about the data, and the data element is the data itself. In other words, all data in an HDF file has attached to it information about itself. For this reason, HDF files are examples of *self-describing files*.

Figure 1.1
Three Data Objects



Data Descriptor (DD)

A DD has four fields: a 16-bit tag, a 16-bit reference number, a 32-bit data offset, and a 32-bit data length. These parts of a DD are depicted in

Chapter 1 The Basic Structure of HDF Files

Chapter Overview

File Header

Data Object

- Data Descriptor
- DD Blocks
- Data Element

Physical Organization of HDF Files

Sample HDF File