CHAPTER

13 Informats

Informats Under UNIX

This chapter describes SAS informats that have behavior or syntax this is specific to UNIX environments. Each informat description includes a brief “UNIX specifics” section that tells which aspect of the informat is specific to UNIX. All of these informats are described in both this documentation and in *SAS Language Reference: Dictionary*.

HEX*w.* informat

Converts hexadecimal positive binary values to fixed- or floating-point binary

Numeric

Width range: 1 to 16

Default width: 8

UNIX specifics: floating-point representation

Details

The HEX*w*. informat converts the hexadecimal representation of positive binary numbers to real floating-point binary values. The width value of the HEXw. informat determines whether the input represents an integer (fixed-point) or real (floating-point) binary number. When you specify a width of 1 through 15, the informat interprets the input hexadecimal as an integer binary number. When your specify 16 for the with value, the informat interprets the input hexadecimal as a floating-point value.

For more details, see “Reading and Writing Binary Data” on page 159 .

$HEX*w.* informat

Converts hexadecimal data to character data

Character

Width range: 1 to 32767

Default width: 2

UNIX specifics: values are interpreted as ASCII values

Details

The $HEX*w.* informat converts every two digits of hexadecimal data into one byte of character data. Use the $HEX*w.* informat to encode hexadecimal values into a character variable when your input data is limited to printable characters. The SAS System under UNIX interprets values that are read with this informat as ASCII values.

IB*w.d* informat

Reads integer binary (fixed-point) values

Numeric

Width range: 1 to 8

Default width: 4

Decimal range: 0 to 10

UNIX specifics: byte values

Details

The IB*w.d* informat reads fixed-point binary values. For integer binary data, the high-order bit is the value's sign: 0 for positive values, 1 for negative. Negative values are represented in two's-complement notation. If the informat includes a *d* value, the data value is divided by 10^d.

For more details, see "Reading and Writing Binary Data" on page 159 .

PD*w.d* informat

Reads packed decimal data

Numeric

Width range: 1 to 16

Default width: 1

Decimal range: 0 to 31

UNIX specifics: data representation

Details

The PD*w.d* informat reads packed decimal data. Although it is usually impossible to type in packed decimal data directly from a console, many programs write packed decimal data.

Each byte contains two digits in packed decimal data. The value's sign is the first byte. Because the entire first byte is used for the sign, you should specify at least a width of 2.

For more details, see "Reading and Writing Binary Data" on page 159 .

PIB*w.d* informat

Reads positive integer binary (fixed-point) values

Numeric

Width range: 1 to 8

Default width: 1

Decimal range: 0 to 10

UNIX specifics: byte order

Details

The PIB*w.d* informat reads integer binary (fixed-point) values. Positive integer binary values are the same as integer binary (see "IB*w.d* informat" on page 208), except that all values are treated as positive. Thus, the high-order bit is part of the value rather than the value's sign. If the informat includes a *d* value, the data value is divided by 10^d.

For more details, see "Reading and Writing Binary Data" on page 159 .

RB*w.d* informat

Reads real binary (floating-point) data

Numeric

Width range: 2 to 8

Default width: 4

Decimal range: 0 to 10

UNIX specifics: floating-point representation; supports single-precision numbers only for those applications that truncate numeric data

Details

The RB*w.d* informat reads numeric data that are stored in real binary (floating-point) notation. The SAS System stores all numeric values in floating-point.

It is usually impossible to type in floating-point binary data directly from a console, but many programs write floating-point binary data. Use caution if you are using the RB*w.d* informat to read floating-point data created by programs other than the SAS System because the RB*w.d* informat is designed to read only double-precision data.

All UNIX systems that are currently supported by the SAS System use the IEEE standard for floating-point representation. This representation supports both single-precision and double-precision floating-point numbers. Double-precision representation has more bytes of precision, and the data within the representation is interpreted differently. For example, for single-precision, the value of 1 in hexadecimal representation is **3F800000**. For double-precision, the hexadecimal representation of 1 is **3FF0000000000000**.

The RB*w.d* informat is designed to read only double-precision data. It supports widths less than 8 only for applications that truncate numeric data for space-saving purposes. RB4. does *not* expect a single-precision floating-point number; it expects a double-precision number truncated to four bytes. Using the example of 1 above, RB4. expects **3FF00000** to be the hexadecimal representation of the four bytes of data to be interpreted as 1. If given **3F800000**, the single-precision value of 1, a different number results.

External programs such as those written in C and FORTRAN can only produce single- or double-precision floating-point numbers. No length other than four or eight bytes is allowed. RB*w.d* allows a length of 3 through 8, depending on the storage you need to save.

The FLOAT4. informat has been created to read a single-precision floating-point number. If you read 3F800000 with FLOAT4., the result is a value of 1.

To read data created by a C or FORTRAN program, you need to decide on the proper informat to use. If the floating-point numbers require an eight-byte width, you should use the RB8. informat. If the floating point numbers require a four-byte width, you should use FLOAT4.

Consider this C example:

```
#include <stdio.h>

main() {

FILE *fp;
float x[3];

fp = fopen("test.dat","wb");
x[0] = 1; x[1] = 2; x[2] = 3;

fwrite((char *)x,sizeof(float),3,fp);
fclose(fp);
}
```

The file **test.dat** contains **3f8000004000000040400000** in hexadecimal representation.

The following statements read **test.dat** correctly:

```
data _null_;
   infile 'test.dat';
   input (x y z) (float4.);
run;
```

Also available is the IEEE*w.d* informat, which reads IEEE floating-point data. On UNIX systems, IEEE8. is equivalent to RB8., and IEEE4. is equivalent to FLOAT4. IEEE*w.d* can be used on any platform, as long as the original IEEE binary data originated on a platform that uses the IEEE representation.

For more details, see “Reading and Writing Binary Data” on page 159 .

ZD*w.d* informat

Reads zoned decimal data

Numeric

Width range: 1 to 32

Default width: 1

UNIX specifics: last byte includes the sign; data representation

Details

The ZD*w.d* informat reads zoned decimal data; it is also known as overprint trailing numeric format. Under UNIX, the last byte of the field includes the sign along with the last digit. The conversion table for the last byte is as follows:

Digit	ASCII Character	Digit	ASCII Character
0	{	-0	}
1	A	-1	J
2	B	-2	K
3	C	-3	L
4	D	-4	M
5	E	-5	N
6	F	-6	O
7	G	-7	P
8	H	-8	Q
9	I	-9	R

For more details, see “ZD*w.d* format” on page 196 and “Reading and Writing Binary Data” on page 159 .

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS® Companion for UNIX Environments, Version 8*, Cary, NC: SAS Institute Inc., 1999.

SAS® Companion for UNIX Environments, Version 8

ISBN 1–58025–502–7

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, October 1999