PO05

Hexadecimals in SAS®: Have You Been Hexed?

Imelda C. Go, South Carolina Department of Education, Columbia, SC

ABSTRACT

For one reason or another, nonalphabetic characters, such as a tab, may unexpectedly appear in a raw data file where only alphabetic characters are expected. This paper discusses how to determine which nonalphabetic characters have contaminated the data file and how to eliminate them during data processing.

The numbers we use for daily life are in the decimal system with a base 10. Numbers in the decimal system are sums of powers of 10. For example, $821 = 8(10^2) + 2(10^1) + 1(10^0) = 800 + 20 + 1$. There are 10 numerals used in the decimal system: 0 through 9.

Binary numbers play a major role in the digital age because they can be easily represented electrically through charges (pulses of high and low voltage). This is much simpler than representing ten numerals as those used in the decimal system. Consequently, the design of computer circuitry and digital data transmission are simpler. There are 2 symbols used in the binary system: 0 and 1. Binary numbers are equivalent to sums of powers of 2 in the decimal system. For example, 821 in the decimal system is represented, in the binary system, by 1100110101 or is $2^9 + 2^8 + 2^5 + 2^4 + 2^2 + 2^0 = 512 + 256 + 32 + 16 + 4 + 1 = 821$. The disadvantage of binary numbers is that they can become much longer than decimal numbers as shown in the above examples for 821.

Hexadecimal numbers are equivalent to sums of powers of 16 in the decimal system. There are 16 symbols in the hexadecimal system: 0 through 9, A, B, C, D, E, and F where A = 10, B = 11, C = 12, D = 13, E = 14, and F = 15.
For example, 821 in the decimal system is represented, in the hexadecimal system, by 335 or is $3(16^2) + 3(16^1) + 5(16^0) = 768 + 48 + 5 = 821$.

Hexadecimal numbers can be used to represent binary quantities since a base of 16 is 2^4 (4^{th} power of 2). Anyone working seriously with computers has to be very familiar with hexadecimal notation because the industry uses hexadecimals in its programs, numbering, etc. This is because hexadecimals use fewer symbols than binary numbers to represent the same decimal number and the hexadecimals are easily represented in computer hardware/software once hexadecimals are converted into binary numbers.

The following table summarizes the differences mentioned above.

	Binary	Decimal	Hexadecimal
Base	2	10	16
Number of Symbols Used	2	10	16
Symbols Used	0, 1	0, 1, 2, 3, 4, 5, 6, 7, 8, 9	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

The following table provides examples of numbers in the three types of notations.

Number	Binary Notation	Decimal Notation	Hexadecimal Notation
zero	0	0	0
one	1	1	1
two	10	2	2
three	11	3	3
four	100	4	4
five	101	5	5
six	110	6	6
seven	111	7	7
eight	1000	8	8
nine	1001	9	9
ten	1010	10	A
eleven	1011	11	B
twelve	1100	12	C
thirteen	1101	13	D
fourteen	1110	14	E
fifteen	1111	15	F
sixteen	10000	16	10
seventeen	10001	17	11
eighteen	10010	18	12
one hundred	1100100	100	64
eight hundred twenty one	1100110101	821	335

I first encountered the problem in the late 1990's using PC SAS. I was running frequency distributions for a race variable where the only known race values were AI, AP, B, H, and W. To my great surprise a frequency distribution, similar to the following, appeared.

The FREQ Procedure

race	Frequency	Percent	Cumulative Frequency	Cumulative Percent
AI	1	16.67	1	16.67
AP	1	16.67	2	33.33
B	1	16.67	3	50.00
H	1	16.67	4	66.67
H	1	16.67	5	83.33
W	1	16.67	6	100.00

Why would there be two entries for H? A call to SAS technical support revealed that the apparently duplicate entry for H appeared because one of the H entries is really an H with an "invisible" tab.

I tried to replicate this problem after more than 10 years later using PC SAS 9.1.3 and was pleased to see that the tabs are no longer invisible.

The FREQ Procedure

race	Frequency	Percent	Cumulative Frequency	Cumulative Percent
AI	1	16.67	1	16.67
AP	1	16.67	2	33.33
B	1	16.67	3	50.00
H□	1	16.67	4	66.67
H	1	16.67	5	83.33
W	1	16.67	6	100.00

I am unaware of how an embedded tab would appear should the problem occur in other operating systems, but whatever the case might be, the following simple steps will solve the problem.

1. Determine which offending characters are embedded in the character data.
2. Remove the offending characters using the COMPRESS function.

This type of problem only emphasizes the need to validate and know the quality of the data. For example prior to using the data, it is useful to check the range of values for a variable. In the examples above, if I just continued to assume that only AI, AP, B, H, and W per se are the valid values that *appear* on the data set, then obviously my assumption would be wrong. Hence, results that are based on that assumption could be unreliable depending on the project's tolerance for error.

Three examples of SAS program code and output are presented below. These examples illustrate one way of solving the problem described above.

EXAMPLE 1

```
data one;
input race $2.;
raceinhex=race;
cards;
AI
AP
B
H
W
;
proc print;
var race raceinhex;
format raceinhex $hex4.;
```

Obs	race	raceinhex
1	AI	4149
2	AP	4150
3	B	4220
4	H	4820
5	W	5720

- The `race` variable is a 2-character string.
- The `raceinhex` variable contains the hexadecimal equivalent of the race variable's value. From the `raceinhex` values above, we can make the following determinations.

Character	Hexadecimal Equivalent	Hexadecimal Notation in SAS
A	41	'41'x
I	49	'49'x
P	50	'50'x
B	42	'42'x
H	48	'48'x
W	57	'57'x
blank	20	'20'x

- Based on the previous bullet's determinations, we can also infer that the hexadecimal equivalent of a string is the concatenation of the 2-character hexadecimal equivalents for each part of the string. For example, AI is 41 (for A) and 49 (for I) joined together as 4149.
- To indicate a number is a hexadecimal one, the SAS convention is to enclose the hexadecimal portion in quotes and to put an x after the quoted value.
- If a character value is a certain length, you will need twice that length in formatting width for the $HEX*w* format, which converts each character in the value into two hexadecimal digits each and where *w* specifies the width of the output field. In the example above, the race variable is a 2-character one. The `raceinhex` values are 4 symbols in length. To make sure that you see the full hexadecimal equivalent of the values, make *w* twice the length of the character value/field. Otherwise, there will be no error message and you will see a truncated and incomplete value.

EXAMPLE 2

```
data two;
input race $2.;
if _n_=4 then race='4809'x;
raceinhex=race;
cards;
AI
AP
B
H
H
W
;

proc print;
var race raceinhex;
format raceinhex $hex4.;

proc freq;
tables race;
```

Obs	race	raceinhex
1	AI	4149
2	AP	4150
3	B	4220
4	H□	4809
5	H	4820
6	W	5720

The FREQ Procedure

race	Frequency	Percent	Cumulative Frequency	Cumulative Percent
AI	1	16.67	1	16.67
AP	1	16.67	2	33.33
B	1	16.67	3	50.00
H□	1	16.67	4	66.67
H	1	16.67	5	83.33
W	1	16.67	6	100.00

- We know beforehand that a tab is '09'x.
- The fourth record is deliberately contaminated with a tab with `if _n_=4 then race='4809'x;`
- The tab follows the H in the contaminated fourth record.

EXAMPLE 3

```
data three;
input race $2.;
if _n_=4 then race='4809'x;
race=compress(race,'09'x);
raceinhex=race;
cards;
AI
AP
B
H
H
W
;

proc print;
var race raceinhex;
format raceinhex $hex4.;

proc freq; tables race;
```

Obs	race	raceinhex
1	AI	4149
2	AP	4150
3	B	4220
4	H	4820
5	H	4820
6	W	5720

The FREQ Procedure

race	Frequency	Percent	Cumulative Frequency	Cumulative Percent
AI	1	16.67	1	16.67
AP	1	16.67	2	33.33
B	1	16.67	3	50.00
H	2	33.33	5	83.33
W	1	16.67	6	100.00

- This example uses the same data set in the second example and eliminates the mischievous tab using the COMPRESS function.
- In the raw data, the fourth record starts as H. It then becomes tab-contaminated with `if _n_=4 then race='4809'x;`
- The tab is removed with `race=compress(race,'09'x);`
- The tab is '09'x in SAS hexadecimal notation.

The examples above show one type of problem that needs to be solved prior to being able to use the data confidently. In the overall scheme of things, data validation is an indispensable step towards ensuring the integrity of data analytical results.

REFERENCES

SAS Institute Inc. 2002. *SAS® 9 Language Reference: Dictionary, Volumes 1 and 2*. Cary, NC: SAS Institute Inc.

TRADEMARK NOTICE