10

Para saber mais

Vimos na aula como usar comandos nas chamadas das bases para personalizar como elas serão usadas na entrada dos meus procedimentos e manipulações, como por exemplo no caso abaixo:

```
DATA teste1;
set alura.cadastro_cliente (obs=15 keep=CPF CEP);

Estado = put(input(substr(cep,1,2),best.),estados_.);

RUN;
```

Mas nós podemos usar esses comandos nos comandos que *escrevem*, que *criam* as bases também, não apenas nos comandos que as leem. Por exemplo, nos *data steps* eu declaro qual base quero criar após o comando `DATA`, e eu poderia mudar como ela será criada usando estes mesmo comandos, por exemplo:

```
DATA teste1  (obs=15 keep=CPF CEP Estado);
set alura.cadastro_cliente;

Estado = put(input(substr(cep,1,2),best.),estados_.);

RUN;
```

A diferença é que aqui a base que eu lerei permanece inalterada, com todas as linhas e variáveis, apenas a base que eu irei escrever será afetada pelos comandos `obs` e `keep`. Observe que como a variável `Estado` existe na minha base de saída, eu preciso acrescentar ele ao `keep` também, caso contrário ele não estará na base.

O comando `rename` também pode ser usado nas chamadas de leitura e escrita de bases, da mesma forma que o `where`. Apenas preciso lembrar de colocar seus parâmetros entre parênteses também. Por exemplo, o código abaixo que vimos anteriormente:

```
DATA teste1;
set alura.cadastro_cliente (obs=max);

precep = substr(cep,1,2);
precep2 = input(precep,best.);

drop precep;
rename precep2 = precep;

RUN;
```

Poderia ser reescrito da seguinte forma:

```
DATA teste1
    (drop=precep
    rename=(precep2=precep));
set alura.cadastro_cliente;
```

```
precep = substr(cep,1,2);
precep2 = input(precep,best.);

RUN;
```

E sobre o uso de formatos e do `PROC FORMAT`, eles interagem com os *missings* de uma forma interessante. Lembrando: *missings* não são um valor em si, mas se eu precisar comparar ele com um número, o SAS sempre considera que o *missing* é menor que qualquer outro valor.

Agora, por eu não estar comparando os *missings* diretamente com um número, e sim procurando um valor em uma lista de valores, se meu formato personalizado não incluir especificamente uma classe onde o *missing* se encaixe, minha variável resultante também será *missing*. Por exemplo, no formato abaixo (usado para classificar a data de lançamento dos jogos):

```
PROC FORMAT;
    value lancamento_
        low    - 201312 = "Antigo"
        201401 - 201606 = "Outro"
        201607 - high   = "Lançamento";
RUN;
```

Se eu usá-lo para classificar minha variável de data na base `cadastro_produto` as observações em que minha data é *missing* continuarão sendo *missing* na minha *flag* de lançamento. Agora, como no formato abaixo eu tenho uma categoria de exceção, e o *missing* é uma observação que cairia justamente nesta classe também e consequentemente seria classificado dentro da categoria `Outro`.

```
PROC FORMAT;
    value lancamento_
        low    - 201312 = "Antigo"
        201607 - high   = "Lançamento"
        other           = "Outro";
RUN;
```