



SÉRIE DE NOTAS TÉCNICAS EM ECONOMIA DA UFG

NT N. 015

NOTA TÉCNICA EM ECONOMIA

n. 015

Publicação cujo objetivo é auxiliar na elaboração de aulas e de pesquisas do Curso de Ciências Econômicas da UFG, divulgando demonstrações técnicas e metodológicas ou rotinas computacionais voltadas para a economia.

As opiniões contidas nesta publicação não representam o ponto de vista do Curso ou da FACE/UFG, sendo de inteira responsabilidade de seu(s) autor(es).

Reprodução permitida, desde que citada a fonte.

FACE – Faculdade de Administração, Ciências Contábeis e Ciências Econômicas

Curso de Ciências Econômicas

Direção FACE

Profª. Andrea Freire de Lucena

Vice-Direção FACE

Profª. Daiana Paula Pimenta

Coordenação do Curso de Ciências Econômicas

Prof. Everton Sotto Tibiriçá Rosa

Coordenação do Programa de Pós- Graduação em Economia

Prof. Cleyzer Adrian da Cunha

Uso dos comandos gen, egen by e bys em Stata

Sandro Eduardo Monsueto

LAM/FACE/UFG

Alicia Araújo Amaral de Oliveira

LAM/FACE/UFG

FACE/UFG

Goiânia – Janeiro de 2025

Versão 1.0

Endereço

Campus Samambaia, Prédio da FACE – Rodovia
Goiânia/Nova Veneza, km. 0, CEP 74690-900,
Goiânia – GO. Tel. (62) 3521 – 1390

URL

<http://www.eco.face.ufg.br>

**CIÊNCIAS
ECONÔMICAS**

FACE
FACULDADE DE
ADMINISTRAÇÃO, CIÊNCIAS CONTÁBEIS E
CIÊNCIAS ECONÔMICAS



Dados Internacionais de Catalogação na Publicação (CIP)
GPT/BC/UFG

Monsueto, Sandro Eduardo. Uso dos comandos gen, egen by e bys em Stata / Alicia Araújo Amaral de Oliveira. - 2025.

13 f.: (Série de Notas Técnicas em Economia, 015)

Universidade Federal de Goiás, Goiânia, 2025. Bibliografia.

Anexos. 1. Software Stata. 2. Comandos. 3. Dados. I. Título.

Uso dos comandos gen, egen by e bys em Stata

Sandro Eduardo Monsueto¹

Alicia Araújo de Amaral de Oliveira²

LAM/FACE/UFG

Resumo: Esta nota técnica tem por objetivo auxiliar no uso de alguns comandos especiais presentes no Stata. Os comandos `gen` e `egen` permitem criar novas variáveis a partir de outras já presentes na base de dados, enquanto os comandos `by` e `bys` facilitam a geração de informações para grupos de observações. São usados alguns exemplos com dados disponíveis no próprio Stata e com dados da Pnad Contínua.

Palavras chave: Stata, comandos, variáveis.

***Abstract:** This technical note aims to assist in the use of some special commands available in Stata. The commands `gen` and `egen` allow for the creation of new variables based on those already present in the dataset, while the commands `by` and `bys` facilitate the generation of information for groups of observations. Some examples use data available within Stata and data from the PNAD Contínua.*

***Key words:** Stata, commands, variables.*

1. Introdução

Nesta Nota Técnica apresentamos de forma sintética o uso de três comandos do Stata que costumam gerar alguma confusão entre os usuários no momento de criar variáveis e manipular o banco de dados. São eles os dois comandos básicos para criar variáveis (*generate* e *egen*) e os prefixos *by* e *bys*. Iremos explorar apenas algumas das opções mais simples, mas que já permitem gerar variáveis com certo nível de complexidade. Os manuais e *help* dos comandos apresentam com mais detalhes todas as opções de uso, além de outros exemplos aplicados dos comandos.

Para tanto, serão usadas como exemplos duas bases de dados. Para os exemplos iniciais, será utilizada a base nativa **nlsw88.dta**, que já vem disponível com a instalação do Stata. Para utilizá-la, entre com a linha de comando abaixo:

```
sysuse nlsw88, clear
```

¹ monsueto@ufg.br.

² aliciaaraujo@egresso.ufg.br.

A base contém informações sobre 2.246 trabalhadores, tais como salário (*wage*), idade (*age*), setor de trabalho (*industry*) e ocupação (*occupation*). É essencial entender como a base de dados está organizada e quais os tipos de variáveis disponíveis. Portanto, invista um tempo em conhecer sua base de dados. Dois comandos que podem ser úteis para essa função são o *describe* e o *browse*, que mostram, respectivamente, a descrição das variáveis e a “planilha” de dados. Também é recomendável o uso destes comandos após a criação de uma nova variável para entender como ficou seu resultado. Além desta base inicial, montaremos um exemplo final utilizando dados da Pnad Contínua, com informações do quarto trimestre de 2020.

2. O comando *gen* (*generate*)

O primeiro comando a ser usado é o *generate*, ou apenas *gen*, responsável por criar variáveis mais simples a partir daquelas já existentes na base de dados. Ele tem a seguinte sintaxe básica:

```
gen nome_da_nova_variável = fórmula ou expressão
```

Onde **fórmula** ou **expressão** corresponde às funções aplicadas às variáveis originais para criar novas variáveis, tais como logaritmos e expressões algébricas de soma, multiplicação etc. Seguem alguns exemplos abaixo usando a base **nlsw88.dta**:

- Gerar a idade ao quadrado:
gen age2 = age^2
- Gerar o logaritmo do salário:
gen lwage = ln(wage)
- Criar a diferença entre a idade e o tempo de experiência no trabalho:
gen idade_inicial = age - tenure
- Criar uma variável multiplicativa entre status civil e idade:
gen casado_idade = married*age

A execução e resultado destes comandos, linha por linha, pode ser seguida no Quadro 1 abaixo. Note que a criação da variável *idade_inicial* segue acompanhada pelo aviso de que 15 missings (informações ausentes) foram criados. Isso ocorre pelo fato de nem todos os trabalhadores da base de dados possuírem a informação de experiência (*tenure*) disponível. O comando *sum* (*sumarize*) é usado para mostrar algumas estatísticas descritivas das novas variáveis.

Quadro 1: Exemplos do comando *gen*

```
. gen age2 = age^2
. gen lwage = ln(wage)
. gen idade_inicial = age - tenure
(15 missing values generated)
. gen casado_idade = married*age
. sum age age2 lwage idade_inicial casado_idade
```

Variable	Obs	Mean	Std. dev.	Min	Max
age	2,246	39.15316	3.060002	34	46
age2	2,246	1542.329	241.6547	1156	2116
lwage	2,246	1.868645	.57458	.0049396	3.707372
idade_inic~l	2,231	33.17365	6.103198	16	45
casado_idade	2,246	25.11398	18.91692	0	45

O comando também pode ser usado para criar variáveis de texto (*string*), além de admitir o uso de operadores lógicos. No exemplo do Quadro 2 abaixo, o uso do comando *gen* é combinado com o comando *replace* e operadores *if* para criar uma variável categórica para a jornada semanal de trabalho.

Quadro 2: Exemplos de uma variável de texto

```
. gen jornada = "parcial" if hours<=20
(1,998 missing values generated)
. replace jornada = "semi_integral" if hours>20 & hours<=40
variable jornada was str7 now str13
(1,604 real changes made)
. replace jornada = "integral" if hours>40
(394 real changes made)
. tab jornada
```

jornada	Freq.	Percent	Cum.
integral	394	17.54	17.54
parcial	248	11.04	28.58
semi_integral	1,604	71.42	100.00
Total	2,246	100.00	

Antes de prosseguir, pode ser útil saber reconhecer alguns dos erros mais comuns no uso deste comando. No Quadro 3, o primeiro comando resulta na mensagem de erro “*variable age2 already defined*”, que simplesmente indica que já existe na sua base de dados uma variável chamada *age2*. No segundo comando, a mensagem “*type mismatch*” indica que tentamos fazer uma operação que não é possível entre os dois tipos de variáveis. Não é possível fazer uma divisão de uma variável numérica (*wage*) com uma de texto (*jornada*). Para evitar esse tipo de erro, fique atento à sequência de execução dos comandos e tenha sempre em mente as características das variáveis em sua base de dados.

Quadro 3: Alguns erros frequentes

```
. gen age2 = age^2
variable age2 already defined
r(110);

. gen salario_hora = wage/jornada
type mismatch
r(109);
```

Também podemos usar o comando *gen* para criar variáveis por meio das denominadas variáveis de sistema, que guardam, por exemplo, a quantidade de observações do banco de dados ou o resultado de outros comandos. As linhas abaixo criam uma variável contadora de observações, iniciando de 1 até a última observação da base de dados, e uma constante com o total de observações, usando os operadores *_n* e *_N* respectivamente.

- Criar uma variável que conta sequencialmente as observações:
gen contador = **_n**
- Criar uma variável constante que guarda a quantidade de observações da base de dados:
gen Total_OBS = **_N**

Para gerar variáveis que usam informações fornecidas nos resultados de outros comandos, verifique no *help* de cada comando, na seção ***Saved results***, o que é guardado dentro das opções **r()**. As linhas abaixo mostram como criar algumas variáveis armazenando os resultados do comando *summarize*:

- Criar variáveis que guardam os resultados do comando *summarize*:
sum wage
gen mediawage = **r(mean)** // uma constante com a média dos salários
gen minwage = **r(min)** // uma constante com o valor mínimo
gen maxwage = **r(max)** // uma constante com o valor máximo

A Figura 1 mostra um recorte da base de dados onde aparecem as colunas com as variáveis criadas até o momento. Note a diferença entre as variáveis **contator** e **Total_OBS**, criadas com os operadores *_n* e *_N* respectivamente. A primeira é uma sequência numérica a partir de 1, enquanto a segunda é uma constante cujo valor é o tamanho da base de dados. Da mesma forma, as últimas três variáveis são constantes que guardam os valores respectivamente da média, mínimo e máximo do salário.

	age	age2	lwage	idade_ini...	casado_i...	jornada	contador	Total_OBS	mediawage	minwage	maxwage
1	37	1369	2.462927	31.66667	0	integral	1	2246	7.766949	1.004952	40.74659
2	37	1369	1.856449	31.75	0	semi_inte...	2	2246	7.766949	1.004952	40.74659
3	42	1764	1.612777	40.75	0	semi_inte...	3	2246	7.766949	1.004952	40.74659
4	43	1849	2.200974	41.25	43	integral	4	2246	7.766949	1.004952	40.74659
5	42	1764	2.089854	24.25	42	integral	5	2246	7.766949	1.004952	40.74659
6	39	1521	1.532477	36.75	39	semi_inte...	6	2246	7.766949	1.004952	40.74659
7	37	1369	2.350531	18	0	semi_inte...	7	2246	7.766949	1.004952	40.74659
8	40	1600	2.845265	25.83333	40	integral	8	2246	7.766949	1.004952	40.74659
9	40	1600	2.57137	34.5	40	parcial	9	2246	7.766949	1.004952	40.74659
10	40	1600	2.047121	37.75	40	integral	10	2246	7.766949	1.004952	40.74659
11	39	1521	2.82111	30.58333	39	parcial	11	2246	7.766949	1.004952	40.74659
12	40	1600	2.739748	26.16667	40	semi_inte...	12	2246	7.766949	1.004952	40.74659
13	40	1600	1.655079	32.25	40	semi_inte...	13	2246	7.766949	1.004952	40.74659
14	40	1600	2.318559	34.41667	0	semi_inte...	14	2246	7.766949	1.004952	40.74659
15	39	1521	2.268184	34.33333	39	parcial	15	2246	7.766949	1.004952	40.74659
16	41	1681	2.203645	38	41	semi_inte...	16	2246	7.766949	1.004952	40.74659
17	42	1764	2.085862	33.91667	42	integral	17	2246	7.766949	1.004952	40.74659

Figura 1: variáveis criadas com o comando *gen*.

3. O comando *egen*

O comando *egen* pode ser considerado uma espécie de “primo rico” do comando *gen*, permitindo criar variáveis mais complexas com o auxílio de funções estatísticas ou lógicas, tanto para variáveis numéricas como aquelas que contêm informação de texto. Seguem abaixo alguns exemplos mais simples. Outras possibilidades podem ser vistas nos exemplos do *help* do comando.

- Criar uma variável que guarda o valor do desvio padrão da idade:
`egen dp_age = sd(age)`
- Criar uma variável que guarda o valor da idade média da base:
`egen media_age = mean(age)`
- Criar uma variável que guarda o valor da mediana:
`egen mediana_age = median(age)`
- Criar uma variável que guarda o valor da soma de todas as idades:
`egen soma_age = total(age)`
- Criar uma variável padronizada pela média e pelo desvio padrão:
`egen vp_age = std(age)`
- Criar uma variável que armazena a moda:
`egen moda_age = mode(age)`

Este último exemplo em particular admite uma série de opções internas para calibrar a determinação da moda quando houver mais de um valor que seja o mais frequente. Ela também pode ser usada com variáveis de texto (*string*), como por exemplo:

```
gen cor = ""
replace cor = "Branco" if race == 1
replace cor = "Negro" if race == 2
replace cor = "Outros" if race == 3
```

```

tab cor
egen cor_mais_frequente = mode(cor)
tab cor_mais_frequente

```

A Figura 2 apresenta um recorte da base de dados com as novas variáveis criadas nesta seção.

	age	dp_age	media_age	soma_age	vp_age	moda_age	cor	cor_mais_frequente
1	37	3.060002	39.15316	87938	-.703647	35	Negro	Branco
2	37	3.060002	39.15316	87938	-.703647	35	Negro	Branco
3	42	3.060002	39.15316	87938	.9303388	35	Negro	Branco
4	43	3.060002	39.15316	87938	1.257136	35	Branco	Branco
5	42	3.060002	39.15316	87938	.9303388	35	Branco	Branco
6	39	3.060002	39.15316	87938	-.0500526	35	Branco	Branco
7	37	3.060002	39.15316	87938	-.703647	35	Branco	Branco
8	40	3.060002	39.15316	87938	.2767445	35	Branco	Branco
9	40	3.060002	39.15316	87938	.2767445	35	Branco	Branco
10	40	3.060002	39.15316	87938	.2767445	35	Branco	Branco
11	39	3.060002	39.15316	87938	-.0500526	35	Branco	Branco
12	40	3.060002	39.15316	87938	.2767445	35	Branco	Branco
13	40	3.060002	39.15316	87938	.2767445	35	Branco	Branco
14	40	3.060002	39.15316	87938	.2767445	35	Branco	Branco
15	39	3.060002	39.15316	87938	-.0500526	35	Branco	Branco
16	41	3.060002	39.15316	87938	.6035417	35	Branco	Branco
17	42	3.060002	39.15316	87938	.9303388	35	Branco	Branco
18	41	3.060002	39.15316	87938	.6035417	35	Branco	Branco
19	42	3.060002	39.15316	87938	.9303388	35	Branco	Branco

Figura 2: variáveis criadas com o comando *egen*.

4. Os pré-comandos *by* e *bysort*

A maior parte das linhas de comandos aplicadas nas seções anteriores, assim como diversos outros comandos do Stata, permitem o uso de alguns prefixos, ou pré-comandos. Entre esses, se destacam os prefixos *by* e *bysort* (ou simplesmente *bys* para este último), que fazem com que o comando seja executado para cada grupo de observações ou categorias. Por exemplo, se desejamos criar uma variável que armazena a média de idade dos trabalhadores de cada ocupação na base de dados, podemos entrar com a linha de comando:

```

bys occupation: egen mage = mean(age)

```

Os prefixos *by* e *bysort* funcionam basicamente da mesma forma, com a diferença de que o uso do primeiro exige que a base esteja pré-ordenada pela variável responsável pelo

agrupamento. Então, para executar a mesma tarefa acima usando *by* ao invés de *bys*, teríamos que fazer:

```
sort occupation
by occupation: egen mage = mean(age)
```

Ou seja, parece mais fácil usar diretamente o prefixo *bys*. Outros exemplos de operações são listados abaixo:

- Contagem segundo cada tipo de ocupação
bys occupation: gen c = _n
- Quantidade total de observações de cada tipo de ocupação
bys occupation: gen C = _N
- Criar a idade média de cada ocupação dentro de cada setor
bys industry occupation: egen m_io_age = mean(age)

A Figura 3 tem um recorte da base para mostrar as novas variáveis criadas com o uso do prefixo *bys*. Note que a variável *c* funciona como um contador de observações para cada ocupação, enquanto a variável *C* armazena o total de observações presente em cada tipo de ocupação. Por exemplo, a base contém 317 observações de trabalhadores da ocupação *Professional/Technical* e 264 observações em *Managers/Admin*.

	occupation	mage	c	C	m_io_age
310	Professional/Technical	38.99054	310	317	39.53333
311	Professional/Technical	38.99054	311	317	38.92788
312	Professional/Technical	38.99054	312	317	38.92788
313	Professional/Technical	38.99054	313	317	38.92788
314	Professional/Technical	38.99054	314	317	37
315	Professional/Technical	38.99054	315	317	38.92788
316	Professional/Technical	38.99054	316	317	38.5
317	Professional/Technical	38.99054	317	317	38.92788
318	Managers/Admin	39.01515	1	264	39.05357
319	Managers/Admin	39.01515	2	264	39.05357
320	Managers/Admin	39.01515	3	264	39.08108
321	Managers/Admin	39.01515	4	264	39.2
322	Managers/Admin	39.01515	5	264	37
323	Managers/Admin	39.01515	6	264	39.08108
324	Managers/Admin	39.01515	7	264	39.10769
325	Managers/Admin	39.01515	8	264	39.08108
326	Managers/Admin	39.01515	9	264	39.10769
327	Managers/Admin	39.01515	10	264	39.05357

Figura 3: variáveis criadas com o prefixo *bys*.

5. Um exemplo com a PNAD Contínua

Veremos agora algumas aplicações usando uma base de dados brasileira, a Pnad Contínua, organizada e disponibilizada livremente pelo IBGE. Aqui, utilizamos os microdados referentes à pesquisa do primeiro trimestre de 2023 para criar um conjunto de novas variáveis que indicam a presença de menores de 10 anos nos domicílios. Serão usadas as variáveis com os códigos originais, disponíveis no dicionário de dados fornecido pelo IBGE.

O primeiro passo é retirar da base de dados as informações de pensionistas, empregado(a) doméstico(o) e seus respectivos parentes, deixando apenas os dados das pessoas que efetivamente residem no domicílio. Para tanto, usamos a variável *vd2002* com a seguinte linha de comando:

```
drop if vd2002 >= 15
```

Para o que vamos fazer, pode ser útil criar uma variável que identifica cada domicílio da base de dados. Com isso, todos os moradores de um mesmo domicílio irão receber o mesmo código identificador. A seguinte sequência de comandos configura corretamente esse identificador para os dados da Pnad Contínua. As duas primeiras linhas alteram a configuração original da variável *upa* para que seus números sejam corretamente reconhecidos, enquanto a última linha utiliza o comando *egen* para juntar as variáveis *uf*, *upa*, *v1008* e *v1014* em uma única, chamada *domicilio*, que contém o identificador:

```
format %15.0g upa // altera o formato da variável
tostring upa, replace // transformando em string
egen domicilio = concat(uf upa v1008 v1014) // cria o identificador
```

Como mostra a Figura 4, todos os moradores de um domicílio receberão o mesmo código identificador:

	uf	upa	v1008	v1014	domicilio
13	11	110000016	6	10	11110000016610
14	11	110000016	7	10	11110000016710
15	11	110000016	7	10	11110000016710
16	11	110000016	8	10	11110000016810
17	11	110000016	10	10	111100000161010
18	11	110000016	10	10	111100000161010
19	11	110000016	11	10	111100000161110
20	11	110000016	12	10	111100000161210
21	11	110000016	14	10	111100000161410
22	11	110000016	14	10	111100000161410
23	11	110000016	14	10	111100000161410
24	11	110000016	14	10	111100000161410
25	11	110000016	14	10	111100000161410
26	11	110000032	1	10	11110000032110
27	11	110000032	1	10	11110000032110
28	11	110000032	2	10	11110000032210
29	11	110000032	2	10	11110000032210

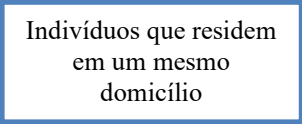


Figura 4: variáveis criadas com o prefixo *bys*.

Feito esses ajustes, vamos agora criar uma variável dummy de valor 1 para identificar todos os menores de 10 anos de idade presentes na base de dados. Chamando essa variável de **dmenor**, podemos gerá-la com os comandos *gen* e *recode* da seguinte forma:

```
gen dmenor = 1 if v2009 < 10
recode dmenor .=0
tab dmenor
```

```
. gen dmenor = 1 if v2009 < 10
(415,394 missing values generated)

. recode dmenor .=0
(415,394 changes made to dmenor)

. tab dmenor
```

dmenor	Freq.	Percent	Cum.
0	415,394	87.76	87.76
1	57,941	12.24	100.00
Total	473,335	100.00	

Figura 5

Como mostra a Figura 5, temos ao todo 57.941 menores de 10 anos na base de dados. Agora vamos identificar com outra dummy todas as pessoas que vivem nos mesmos domicílios que estes menores de idade. Ou seja, será gerada a variável **domicilio_com_menores** que recebe valor 1 se o indivíduo vive em um domicílio que possui crianças menores de 10 anos de idade e de valor zero em caso contrário. Para tanto, usaremos o comando *egen* junto com o prefixo *bys* e combinaremos as informações dos moradores de um mesmo domicílio. Dentro de um mesmo domicílio, a variável **dmenor** terá seu valor máximo igual a 1 se um dos moradores for menor de 10 anos. Caso contrário, seu valor máximo será zero. Então, podemos criar nossa nova dummy com comando:

```
bys domicilio: egen domicilio_com_menores = max(dmenor)
tab domicilio_com_menores
```

```
. bys domicilio: egen domicilio_com_menores = max(dmenor)

. tab domicilio_com_menores
```

domicilio_c om_menores	Freq.	Percent	Cum.
0	293,686	62.07	62.07
1	179,492	37.93	100.00
Total	473,178	100.00	

Figura 6

Como mostra a Figura 6, 37,93% da base de dados vive em um domicílio com ao menos um menor de 10 anos de idade (inclusive os próprios menores). Podemos também criar uma variável que mostra a quantidade total de menores de 10 anos de idade em cada domicílio. Usamos o comando abaixo para criar a variável **qmenores** que é uma soma da variável **dmenor** para cada domicílio:

```
bys domicilio: egen qmenores = total(dmenor)
tab qmenores
```

```
. bys domicilio: egen qmenores = total(dmenor)

. tab qmenores
```

qmenores	Freq.	Percent	Cum.
0	293,686	62.07	62.07
1	116,468	24.61	86.68
2	47,080	9.95	96.63
3	12,183	2.57	99.21
4	2,742	0.58	99.78
5	729	0.15	99.94
6	170	0.04	99.97
7	64	0.01	99.99
8	38	0.01	100.00
9	18	0.00	100.00
Total	473,178	100.00	

Figura 7

Por último, vamos criar a renda domiciliar per capita que, neste caso, irá usar apenas o rendimento proveniente do trabalho. Para isso, vamos criar uma variável que é o total de moradores em cada domicílio e outra que é a soma da renda do trabalho de todos os moradores. Finalmente, geramos a renda domiciliar, que é a simples divisão de uma pela outra:

```
bys domicilio: gen moradores = _N
bys domicilio: egen renda = total(vd4019)
gen rendacapita = renda/moradores
sum rendacapita
```

```
. bys domicilio: gen moradores = _N

. bys domicilio: egen renda = total(vd4019)

. gen rendacapita = renda/moradores

. sum rendacapita
```

Variable	Obs	Mean	Std. dev.	Min	Max
rendacapita	473,178	1090.284	2031.16	0	180000

Figura 8

A presença de menores de idade nos domicílios, por exemplo, pode ser usada como variável explicativa em modelos para a probabilidade de inserção no mercado de trabalho e a renda domiciliar per capita empregada em análises de pobreza e desigualdade. Os comandos

usados podem ser combinados e aplicados em outras situações de interesse do pesquisador. Mais uma vez, recomendamos a leitura atenta das respectivas páginas de *help* de cada comando para entender seus usos mais complexos. O software também disponibiliza uma documentação em formato PDF para seus principais comandos e funções. A documentação do comando *gen*, por exemplo, pode ser consultada em StarCorp (2023).

Referências bibliográficas

StataCorp. 2023. Stata 18 Data Management Reference Manual. College Station, TX: Stata Press.