

## Stata versão10 – Noções básicas

---

Tópicos que serão vistos neste material

1. Informações sobre o *Stata*
2. Iniciando o *Stata*
3. Leitura e salvamento de banco de dados
4. Criando banco de dados com entrada de dados diretamente no *Stata*, pelo teclado
5. Variáveis
6. Sintaxe dos comandos
7. Manipulação de dados
  - 7.1 Expressões
  - 7.2 Gerando e recodificando variáveis
8. Descrevendo dados
  - 8.1 Gráficos
  - 8.2 Tabelas e resumo de dados
9. Inferência – tomando decisões
10. Modo de programa (extensão do ?)

### 1. Informações sobre o *Stata*

*Stata* é um pacote comercial estatístico (não de domínio público) desenvolvido por *Stata Corporation*. Este curso é referente ao *Intercooled Stata*, **versão 10** para Sistema *Windows*.

O *Stata* é descrito em cinco manuais e em *Hamilton* (1998). Cada comando está associado a um arquivo-*help* que pode ser acessado durante a utilização do programa.

Informações sobre o *Stata*, bem como atualizações, realização de cursos via *Internet* e lista das dúvidas mais freqüentes podem ser obtidas no site: <http://www.stata.com>. O *Stata* possui suporte técnico e informações que podem ser obtidas pelo **Help** disponível no menu principal.

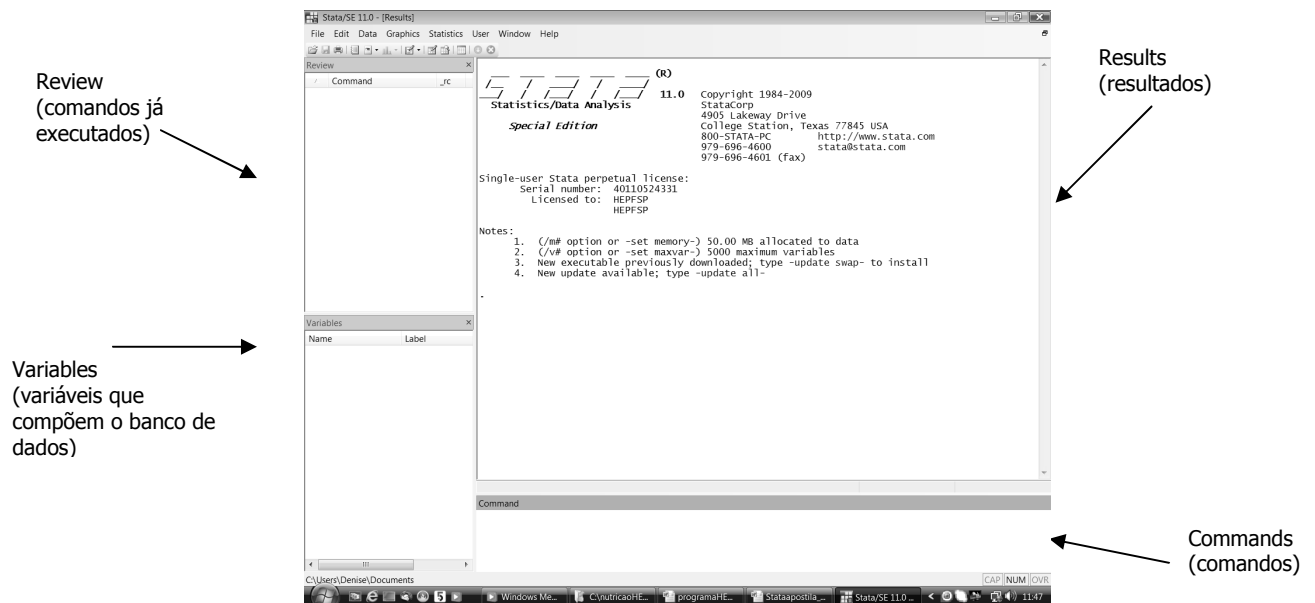
<b>O <i>Stata</i> diferencia entre letra maiúscula e minúscula.</b>
---

### 2. Iniciando o *Stata*

O programa pode ser aberto diretamente pelo ícone (clcando duas vezes sobre ele) na tela de abertura do *Windows* ou seguindo o caminho **Iniciar, Programas ...**

Quando o programa é aberto, abre-se uma tela contendo janelas menores, com cabeçalhos. A finalidade de cada janela é apresentada a seguir:

Cabeçalho	Finalidade
<i>Review</i>	Armazenamento dos comandos já utilizados
<i>Variables</i>	Apresentação das variáveis contidas no banco de dados
<i>Stata Results</i>	Apresentação dos resultados obtidos com a execução dos comandos
<i>Stata Command</i>	Digitação dos comandos a serem executados



Quando o *prompt* (sinal de barra | intermitente) aparece na janela *Stata Command*, significa que o comando pode ser digitado.

A execução de um comando é feita digitando-se o comando segundo uma sintaxe definida e **pressionando a tecla *Enter***.

O comando, uma vez executado, irá aparecer na janela *Review* e pode ser reutilizado e corrigido, se necessário, posicionando o cursor sobre ele e pressionando *Enter* (para retornar na linha de comando para correção) e mais um *Enter* para ser executado; ou utilizando-se as teclas *PgUp* (*page up*) e *PgDown* (*page down*).

Após a execução do comando, o resultado aparecerá na janela *Stata Results*. Se o espaço necessário para a apresentação dos resultados não for suficiente, aparecerá a palavra **---more---**. Neste caso deve-se pressionar *Enter* até que todo o resultado seja apresentado. Para desligar esta função, deve-se digitar o comando **set more off** <E>. A janela de resultados pode ser rolada para baixo e para cima utilizando-se a barra de ferramenta posicionada à direita da janela.

No *Stata* **somente um arquivo de dados pode ser aberto e utilizado de cada vez**. Os dados são apresentados em formato de matriz onde as colunas representam as variáveis e as linhas as observações para cada registro. Quando um banco de dados (*dataset*) é aberto, as variáveis aparecem listadas na janela *Variables*.

**O *Stata* é um programa de comandos, que na versão 10 utiliza a estrutura de menus para facilidade do usuário.**

Os comandos têm uma forma geral do tipo **comando lista de variáveis (*command varlist*)**. Por exemplo, se o banco de dados contém as variáveis **x** e **y**, então, o comando para listá-las é: **list x y**.

Outros componentes podem ser adicionados, por exemplo, **if x>y** fará com que sejam listados somente os registros onde os valores de **x** são maiores que os de **y**. As opções são incluídas no comando após uma vírgula. A estrutura completa de um comando será descrita oportunamente.


A utilização do *Help* é fortemente recomendada; clicando-se em *Help* no menu principal, uma segunda janela é apresentada, onde é possível pesquisar qualquer comando utilizando-se a opção **Contents**, **Search** ou **Stata command**.

O *Stata* trabalha com **basicamente 4 tipos de arquivos**:

tipo de arquivo	Extensão
arquivo que contém os dados	<b>.dta</b>
arquivo que contém os comandos e resultados obtidos durante a sessão de trabalho	<b>.log</b>
arquivo que contém comandos	<b>.do</b>
arquivo que contém sub-rotinas	<b>.ado</b>

### Arquivo log

Logo que for iniciado o trabalho no *Stata*, é aconselhável abrir um arquivo **log** que armazenará todos os comandos e resultados da execução destes. O arquivo **log** é de tipo **somente texto** e **não permite alteração em outro programa a não ser que seja convertido em arquivo texto, do Word, por exemplo**. Caso isto seja de interesse, pode-se abrir um arquivo \*.log, no *Word for Windows* e, após salvamento como um documento do *Word* (com extensão **.doc**), ser manipulado segundo a necessidade.

É possível abrir um arquivo **.log** clicando no ícone  referente a "*Begin log*". **Deve-se escolher tipo de arquivo \*.log**, dar um nome tal como analise1.log.

Observar a sintaxe do comando para abrir um arquivo **.log**

**Ex: log using <C:\HEP139\_2011\analise1.log>, text replace**

Para fechar um arquivo log deve-se utilizar o comando **log close**.

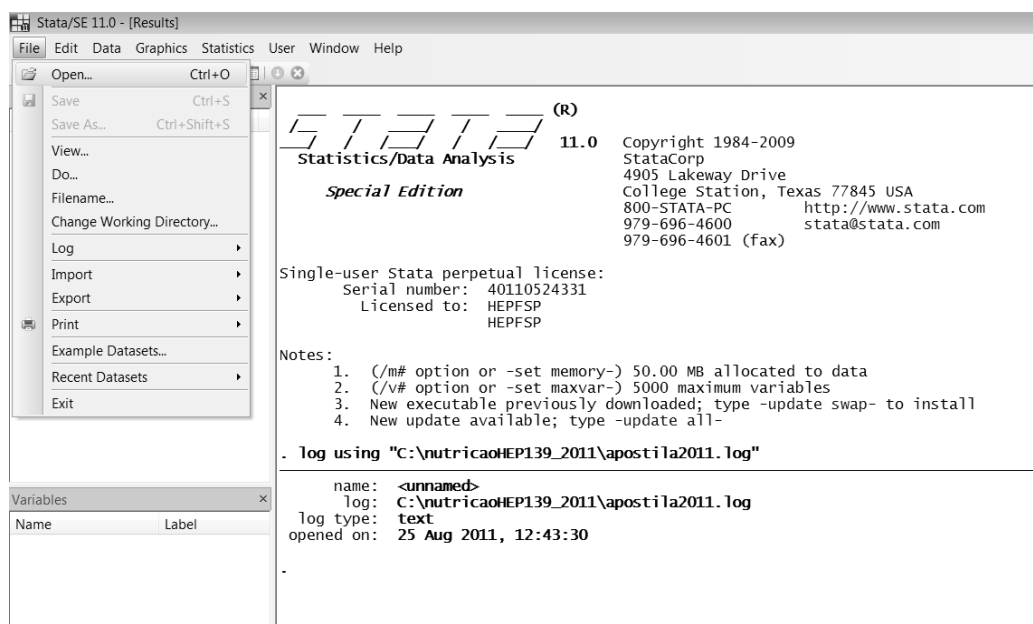
### 3. Leitura e salvamento de banco de dados

O *Stata* possui seu próprio formato de banco de dados com extensão **.dta**.

Para ler um arquivo no *Stata*, é necessário o comando **use**. Antes de utilizar este comando deve-se limpar os dados utilizados anteriormente porque podem estar ainda na memória de execução do programa. Deve-se, portanto, rodar o comando **clear** antes do comando **use**.

### Dois modos para abrir um banco de dados:

**Pelo menu:** clicar sobre **File** e escolher **Open**



Também pode-se digitar o comando  
**use <caminho\oswego.dta>**

#### Para salvar um arquivo de dados (extensão .dta)

Para salvar o banco com o mesmo nome, é necessário acrescentar a opção **replace** ao comando **save**:

**save c:\diurno\oswego, replace**

OU

**save, replace**

Neste caso, o *Stata* salvará o banco atual com as eventuais modificações introduzidas no banco durante a sessão de trabalho; o banco inicial será substituído.


Banco de dados em um formato diferente de .dta: Os dados que não estiverem no formato *Stata* devem ser convertidos, utilizando-se, por exemplo, o **Stat/Transfer** ou outro pacote que realize conversão de bancos de dados (Anexo I).

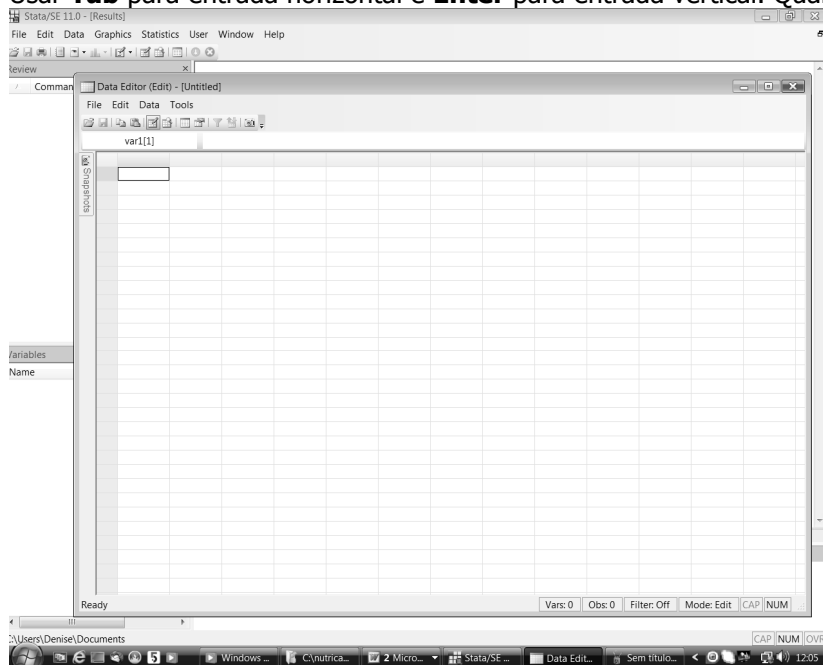
#### Leitura e salvamento de banco de dados via caixa de diálogo (menu principal)

O salvamento do banco pode ser realizado selecionando-se **Save** ou **Save As** na opção **File**

#### 4. Criando banco de dados

Abrir o modo de edição clicando sobre o ícone  **Data editor** e digitar os dados dos registros.

Usar **Tab** para entrada horizontal e **Enter** para entrada vertical. Quando terminar, escolha **Close** 



Ex: Criar um banco de dados com nome **banco1** que contenha as variáveis: identificação (id), nome (nome), tratamento (tratamen), peso inicial (pesoinic) e sexo (sexo); para 5 pacientes, com dados apresentados a seguir.

id	nome	tratamen	pesoinic	sexo	marital
1	"A S"	0	98.4	1	1
2	"G S"	1	75.5	2	1
3	"V G"	1	93.6	2	2
4	"M C"	0	80.2	1	3
5	"A C"	0	70.0	2	4

O arquivo deve ser salvo utilizando a caixa de diálogo, na sequência: **File, Save As ...**

## Abrir o banco de dados oswego.dta

### 5. Variáveis

Existem essencialmente dois tipos de variáveis no *Stata*: *string* e *numérica*. Estas variáveis são armazenadas de formas diferentes que requerem tamanhos diferentes: *byte*, *int*, *long* e *float* para variáveis numéricas e *str1* até *str80* para variáveis *string* de tamanhos diferentes. Além disto, associado à cada variável existe um nome, um rótulo (*label*) e um formato.

Utilizar o comando **describe** para investigar as variáveis que compõem o banco de dados.

Utilizar o comando **codebook** para descrever as variáveis

O nome de uma variável pode ser alterado. Por exemplo, a variável **ill** pode ser mudada para **doente** usando o comando:

**rename ill doente**

(ou **ren ill doente**, utilizando as 3 primeiras letras do comando)

O rótulo da variável pode ser definido com o comando

**label variable age "idade"**

ou

**label var onsetdate "data de inicio"**

A definição de **rótulos para categorias** de variáveis pode ser feita com os comandos:

**label define ill 1 "doente" 0 "não doente"**

**label values doente ill**

**tab doente**  
**tab doente,nolabel** ou **tab doente,nol**

Dando rótulos para a variável **sex**

A variável **sex** é string

Primeiro é necessário construir uma variável em numérica utilizando-se o comando **gen**:

**gen sexo = 1 if sex=="M"**

**replace sexo=0 if sex=="F"**

**tab sexo**

Agora é possível associar um rótulo aos valores 0 e 1:

**label define s 1 "masculino" 0 "feminino"**  
**label val sexo s**  
**tab sexo**

**OBS** Quando o *label* for igual para várias variáveis, é possível direcionar o *label* utilizando do comando:  
**label val <var1> <var2> ....<varn> <nome do label>**

Para fechar o banco contendo as modificações mudando-se o nome do banco de dados utilize o comando **save as <nome novo>**. Para salvar por cima do banco original utilizar o comando **save, replace**

## 6. Sintaxe dos comandos

Os comandos seguem a forma

<b>[by varlist:] command [varlist] [weight] [if exp] [in range] [using filename] [,options]</b>
---

onde,

**[by varlist:]** instrui *Stata* para repetir o comando para cada combinação de valores nas variáveis listadas em *varlist*;

**command** é o nome do comando, ex: **list**

**[varlist]** é a lista de variáveis para as quais o comando é executado

**[weight]** permite que pesos sejam associados às observações

**[if exp]** restringe o comando a um subconjunto de observações que satisfazem a expressão lógica definida em *exp*

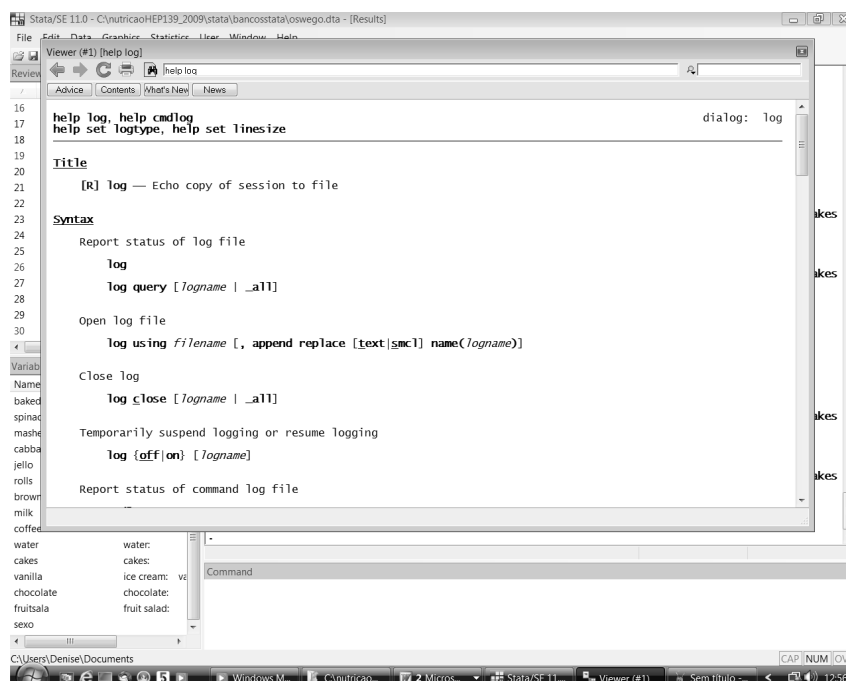
**[in range]** restringe o comando àquelas observações cujos índices pertencem a um determinado subconjunto

**[using filename]** especifica o arquivo que deve ser utilizado

**[,options]** são específicas de cada comando.

**Ex:**

**help log** fornece a informação:



**log:** comando

**using filename:** arquivo onde será armazenada a sessão de trabalho (extensão **.log**)

**append:** esta opção instrui programa a salvar a sessão atual, como continuação de uma sessão anterior.

**replace:** esta opção instrui programa a salvar a sessão atual sobre um texto já existente.

**text|smcl:** tipo de arquivo

#### Options for use with log

**text** and **smcl** specify the format in which the log is to be recorded. The default is complicated to describe but is what you would expect:

If you specify the file as *filename.smcl*, the default is to write the log in SMCL format (regardless of the value of **set logtype**).

If you specify the file as *filename.log*, the default is to write the log in text format (regardless of the value of the **set logtype**).

If you type *filename* without an extension and specify neither the **smcl** option nor the **text** option, the default is to write the file according to the value of **set logtype**. If you have not **set logtype**, then the default is SMCL. Also the *filename* you specified will be fixed to read *filename.smcl* if a SMCL log is being created or *filename.log* if a text log is being created.

If you specify either of the options **text** or **smcl**, then what you specify determines how the log is written. If *filename* was specified without an extension, the appropriate extension is added for you.

If you open multiple log files, you may choose a different format for each file.

## 7. Manipulação de dados

### 7.1 - Expressões

Existem expressões lógicas, *string* e algébricas, no *Stata*.

Expressões lógicas atribuem 1 (verdadeiro) ou 0 (falso) e utiliza os operadores:

Operador	Significado	Operador	Significado
<	menor que	= =	igual a
<=	menor ou igual a	~=	diferente de
>	maior que	~	não
>=	maior ou igual a	&	E

**if (y~=2 & z>x) | x= =1**

Significa: se **y** for diferente de 2 e **z** maior do que **x** ou **x** for igual a 1

Expressões algébricas utilizam os operadores:

Operador	Significado	Operador	Significado
+ -	soma, subtração	Exp( )	função exponencial
* /	multiplicação, divisão	log( )	função logarítmica (base 10)
^	elevado à potência	ln( )	função logarítmica (base e) - logaritmo natural
Sqrt( )	função raiz quadrada		

### Situação 1:

Uma enzima (*ornithine carbonyl transferase*) do fígado de pessoas que sofrem de dois tipos de hepatite (viral aguda e crônica) é quantificada.

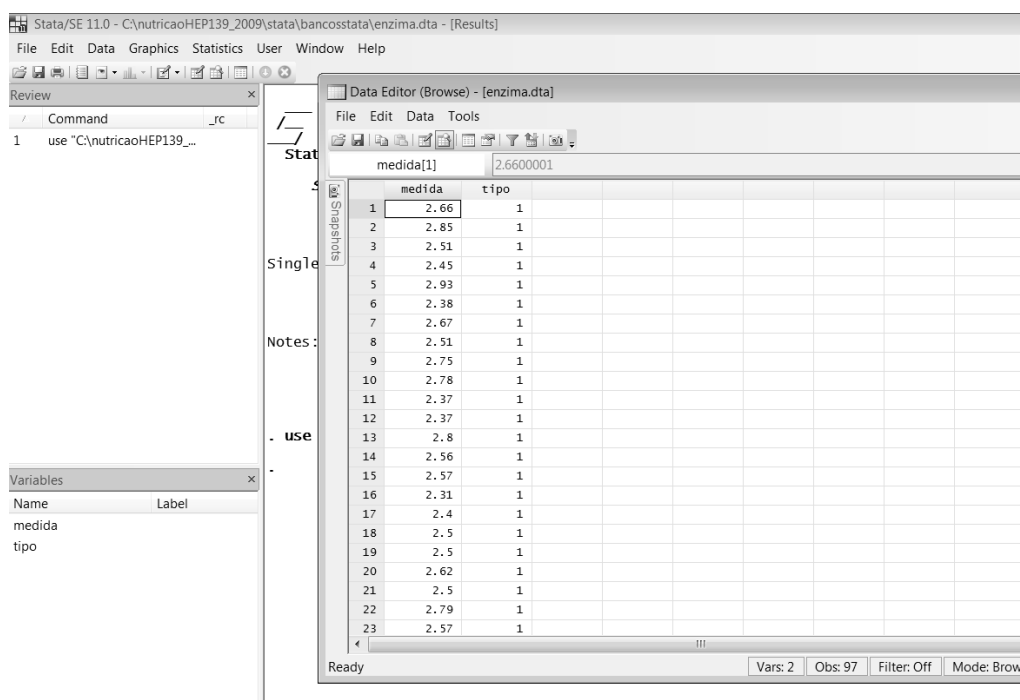
#### Medida da enzima entre pacientes com hepatite viral aguda (log das medidas)

2,66	2,38	2,37	2,31	2,50	1,96	2,85	2,68	1,76	2,36	2,56	2,09
2,85	2,67	2,37	2,40	2,79	1,82	3,00	2,50	2,36	2,48	2,60	2,42
2,51	2,51	2,80	2,50	2,57	2,54	2,53	2,78	2,07	2,35	2,98	2,31
2,45	2,75	2,56	2,50	3,00	2,94	2,46	2,83	3,61	2,99	2,78	3,02
2,93	2,78	2,57	2,62	2,71	2,18	3,21	2,86	2,51			

#### Medida da enzima entre pacientes com hepatite crônica (log das medidas)

3,01	2,99	2,60	2,47	3,04	1,92	2,17	2,33	2,07	2,30	2,56	2,11
3,32	2,21	1,71	2,60	2,79	2,71	2,64	2,52	2,21	2,58	2,40	2,45
3,18	2,84	2,84	2,31	2,71	2,47	2,72	3,71	2,73	3,69	3,40	2,77
2,28	2,84	2,80	3,02								

### Banco de dados: **enzima.dta**



Para repetir comandos para variáveis ou categorias de variáveis, utilizar **by varlist**; os dados precisam estar ordenados antes disto, o que é feito utilizando o comando **sort**.

Fornecendo rótulos à variável tipo

**label define tipo 1 "hepatite aguda" 2 "hepatite crônica"**

**label val tipo tipo**

**tab tipo**

Listando os dados segundo tipo

**by tipo: list medida**

OBS: o comando **by** pede que os dados sejam ordenados (utilizar o comando **sort**)

**sort tipo**

**by tipo: list medida**

Resumindo os dados segundo a variável **tipo** utilizando o comando **summarize** ou **sum**

**by tipo: sum medida**



## 7.2 Gerando e recodificando variáveis

Uma variável existente pode ser retirada do banco de dados com o comando **drop <var>**

### Banco de dados: colesterolhomens.dta

Investigando a existência de valores ignorados; normalmente codificados com 99 ou -99.

Recodificando valores ignorados -99 para valores missing, utilizando o comando **recode**:

```
recode colesterol -99=.
```

Utilizar o comando **tab** com opção **mis** ou o comando **codebook** para investigar a existência de valores missing

```
tab colesterol, mis ou codebook colesterol
```

O comando **generate** iguala uma nova variável a uma expressão que é construída para cada observação

Gerando uma variável **colesterol2** construída a partir da variável **colesterol**. A nova variável será categórica com duas categorias: abaixo de 200 e igual ou maior que 200.

```
generate colesterol2= 1 if colesterol>=200 & colesterol ~=. .
```

Gerar uma nova variável (**colesterol2**) que contém valores 1 (se **colesterol**  $\geq 200$  e diferente de *missing*) e *missing* (se **colesterol**  $< 200$ ). Para substituir os valores *missing* por um valor numérico, utilizar o comando **replace** que permite alterar uma variável já existente.

```
replace colesterol2= 0 if colesterol<200  
list colesterol colesterol2  
tab colesterol2  
tab colesterol2,mis  
label define colesterol2 0 "<200" 1 ">=200"  
label val colesterol2 colesterol2  
tab colesterol2
```

Ex2: Dividir a variável em 4 partes iguais, utilizando os valores dos percentis como pontos de corte.

```
sum colesterol,d  
gen colesterolg =colesterol  
recode colesterolg 100/187=1 188/215 =2 216/234=3 235/382=4  
tab colesterolg  
label define colesterolg 1 "100 – 187" 2 "188 – 215" 3 "216 – 234" 4 "235 – 382"  
label val colesterolg colesterolg  
tab colesterolg
```

Ex3: A variável **colesterol** pode ser recodificada em duas ou mais categoria por meio do comando **recode**. Criando uma nova variável **colesterol** codificada em duas categorias (abaixo e acima do valor da mediana)

```
gen colesterolg3 = colesterolg  
recode colesterolg3 1 2 =0 3 4 =1 (ou recode colesterolg3 1/2=0 3/4=1)  
tab colesterolg3  
label define colesterolg3 0 "100 – 215" 1 "216-382"  
label val colesterolg3 colesterolg3  
tab colesterolg3
```

Fazendo tabelas cruzadas

Para cruzar duas variáveis qualitativas deve-se utilizar o comando **tab <va1> <var2>**.

Ex: Cruzando as variáveis **colesterol2** e **dic**  
**tab dic colesterol2, col**

Fornecendo rótulos para a variável **dic**:

```
label define dic 0 "DIC ausente" 1 "DIC presente"  
label val dic dic  
tab dic colesterol2, col
```

Ex: Cruzando as variáveis **colesterolg** e **dic**  
**tab colesterolg dic, row**

Ex: Cruzando as variáveis **colesterolg3** e **dic**  
**tab colesterolg3 dic,col**

Banco de dados: **enzima.dta**

Supor que se deseja transformar os códigos da variável **tipo (1 e 2)** de 1 para 0 e de 2 para 1, ou seja hepatite aguda era 1 e passa a ser 0 e hepatite crônica era 2 e passa a ser 1:

```
gen tipo2=tipo  
recode tipo2 1=0 2=1  
label define tipo2 0 "hepatite aguda" 1 "hepatite crônica"  
label val tipo2 tipo2  
tab tipo2 tipo                (para checar se deu certo)
```

## 8. Descrição de dados

### 8.1 - Gráficos

**Abrir o arquivo oswego.dta**

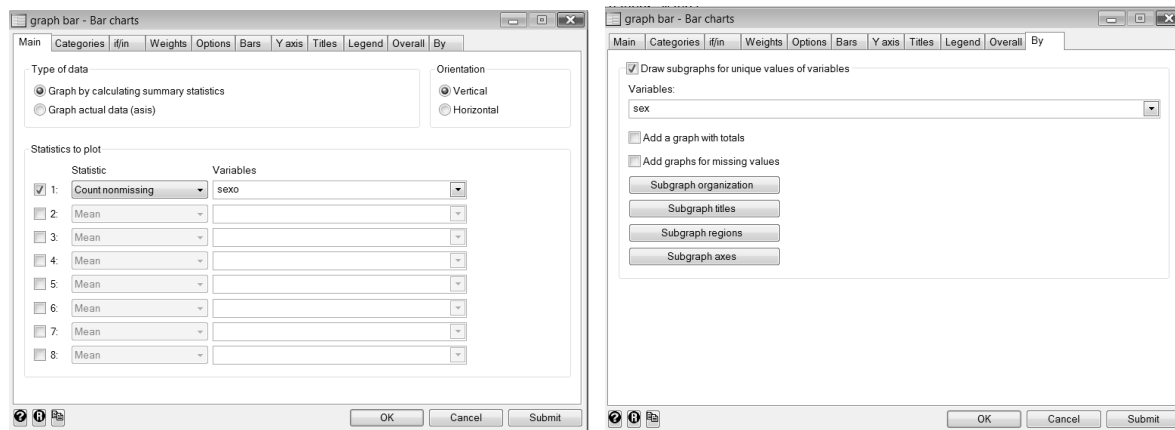
#### Gráfico de barras

Antes de fazer o gráfico da idade (age) média segundo sexo, é necessário transformar a variável **sexo** de string em numérica. Para isto, utilizar a sequência de comandos:

```
desc  
gen sexo=1 if sex=="F"  
replace sexo=2 if sex=="M"  
tab sexo  
label define sexo 1 "Feminino" 2 "Masculino"  
label val sexo sexo
```

Gráfico da variável **sexo**

Utilizando-se os menus: *Graphics, Bar chart, Main [Statistic:counting nonmissing; variables:sexo]; By[sex]*



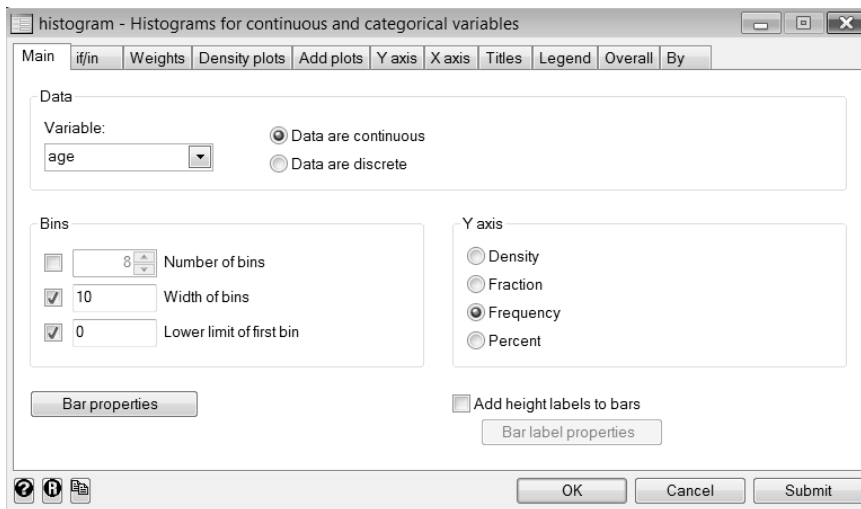
OBS: veja a sintaxe do comando

**graph bar (count) sexo, by(sex)**

Fazer o gráfico da variável **ill** (doente)  
**graph bar (count) ill, by(sex)**

## Histograma

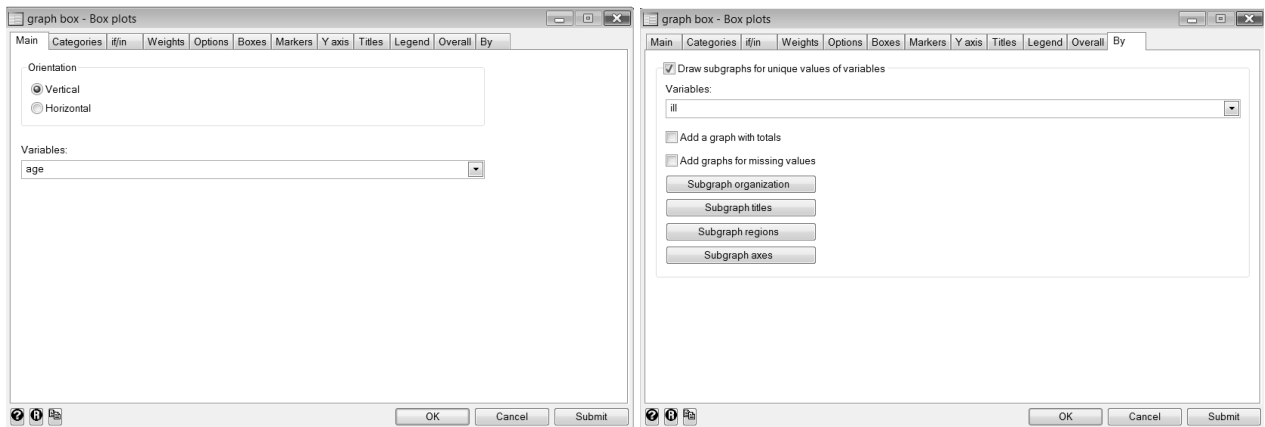
Para desenhar um histograma utilizar o menu *Graphics; Histogram* seguida da seleção da variável, tipo de variável, a amplitude dos intervalos (*bins*) bem como o valor de início do primeiro intervalo e no eixo Y a frequência de ocorrência.



OBS: veja a sintaxe do comando  
**histogram age, width(10) start(0) frequency**

## Boxplot

Construindo-se o box plot para a variável idade segundo a variável **ill**



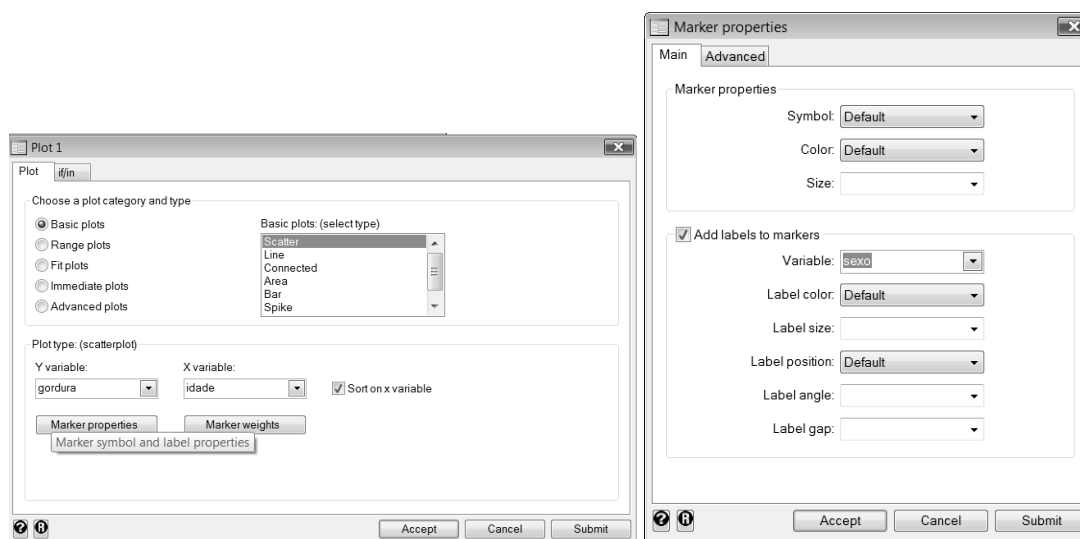
Utilizando-se o menu: *Graphics; Box plot*; e opções apresentadas nos quadros acima.

OBS: veja a sintaxe do comando  
**graph box age, by(ill)**

Utilizar o banco de dados: **gordura.dta**

## Diagrama de dispersão

Utilizando-se o menu: Graphics; Twoway; Create ...;(scatterplot, line, etc.); Scatter; Y variable: gordura; X variable: idade; Marker properties [variable: sexo]



OBS: perceba a sintaxe do comando

**twoway (scatter gordura idade, sort mlabel(sexo))**

Para construir dois gráficos de dispersão, um para cada sexo

**twoway (scatter gordura idade, sort mlabel(sexo)), by(sexo)**

## 8.2 – Tabelas e resumo dos dados

Os dados que serão utilizados nesta sessão constituem uma amostra de 118 pacientes psiquiátricas, do sexo feminino e estão disponíveis em Hand DJ, et al. ***A Handbook of Small Data Sets***. Chapman & Hall, London, 1994. Algumas das variáveis estudadas foram:

- **age**: idade em anos
- **iq**: escore de inteligência (-99 = ignorado)
- **anxiety**: ansiedade (1= nenhuma, 2= leve, 3= moderada, 4=grave, -99=ignorado)
- **depress**: depressão (1=nenhuma, 2= leve, 3= moderada, 4=grave, -99=ignorado)
- **sleep**: você pode dormir normalmente? (1=sim, 2=não, -99=ignorado)
- **weight**: mudança no peso durante os últimos 6 meses (em libras)

### Banco de dados: fem.dta

As variáveis medidas em escala de razões (contínuas) são **age**, **iq** e **weight**. As variáveis qualitativas ordinais são **anxiety** e **depress** e a variável **sleep** é qualitativa nominal

A seguir, apresenta-se uma lista de comandos utilizados para a construção de tabelas de frequência uni e bi-dimensionais, para resumo dos dados em medidas de tendência central e de dispersão

### Comando **describe** ou **desc**

Permite verificar as variáveis que compõem o banco de dados

### Comando **tabulate** ou **tab** ou **tab1**

construir uma tabela de frequências simples de cada variável

Possibilidades: tab1 \_all

tab1 age anxiety iq depress sleep weight

tab age

```
tab anxiety
tab iq
tab depress
tab sleep
tab weight
```

#### Comando **recode**

Utilizado para recodificar valores

Recodificar os valores faltantes das variáveis, codificadas como -99

```
recode sleep -99=.
recode depress -99=.
```

#### Comandos **label define** e **label values**

Permitem fornecer rótulos para as variáveis

Para a variável **anxiety**

```
label define grau 1 "nenhum" 2 "leve" 3 "moderada" 4 "grave"
label val anxiety grau
label val depress grau
tab anxiety
tab depress
```

Para a variável **sleep**

```
label define sn 1 "sim" 2 "não"
label val sleep sn
```

#### Comando **summarize** ou **sum**

Permite resumir uma variável quantitativa contínua (cálculo de medidas de tendência central e de dispersão)

Resumindo as variáveis **iq**, **age** e **weight**

```
sum iq
sum iq,d
sum weight,d
sum age,d
```

Resumir a variável **weight** segundo a variável **sleep**

```
sort sleep
by sleep: sum weight,d ou table sleep ,c(mean weight sd weight)
```

#### Comando **generate** ou **gen**

Permite gerar novas variáveis

Gerando nova variável **peso** construída a partir da variável **weight**. A variável **peso** deve ser igual a 1 se a pessoa perdeu peso e 0 se permaneceu no mesmo peso ou se ganhou peso.

```
gen peso= 1 if weight<0
replace peso =0 if weight >=0
tab peso
label define peso 0 "ganhou peso" 1 "perdeu peso"
label val peso peso
tab peso
```

Criar uma variável **ageg** contendo a variável **age** em intervalos de classes de 5 anos

```
gen ageg=age
recode ageg 25/29=1 30/34=2 35/39=3 40/44=4 45/49=5
label define idade 1 "25-29" 2 "30-34" 3 "35-39" 4 "40-44" 5 "45-49"
label val ageg idade
tab ageg
```

Criar a variável **agedic** contendo a variável **age** em duas categorias (<= média e acima da média)

```
sum age
gen agedic =age
recode agedic min/37.5=1 37.6/max=2
```

label define agedic 1 "<=37.5" 2 ">37.6 e mais"  
label val agedic agedic  
tab agedic

## 9. Testes de hipóteses

Com base nos dados do banco **fem.dta**, apresenta-se a seguir a comparação de algumas medidas estatísticas mediante a realização de testes de hipóteses.

Para comparar as variáveis quantitativas pode-se utilizar o teste *t de "Student"* que assume que as observações nos dois grupos são independentes; as amostras foram retiradas de populações com distribuição normal, com mesma variância.

Também é possível calcular correlações entre variáveis contínuas. Se for de interesse testar se o coeficiente de correlação de *Pearson* é estatisticamente diferente de zero, o Stata apresenta um teste que pressupõe que as variáveis são normais bivariadas.

Para as variáveis qualitativas nominais pode-se utilizar o teste Qui-quadrado, de *Pearson*.

### Teste de hipóteses para uma, duas e mais de duas médias; associação pelo qui-quadrado de Pearson e intervalos de confiança para proporção e média populacional

Utilize o banco de dados: **enzima.dta**

Teste de igualdade de variâncias

Testar a diferença entre as variâncias da variável **medida** segundo **tipo**

```
sort tipo  
sdtest medida,by(tipo)
```

Teste de igualdade de duas médias

Testar se existe diferença entre a mudança média de níveis da enzima nos dois tipos de hepatite

```
sort tipo  
ttest medida,by(tipo)
```

Intervalo de confiança para a média populacional

Apresentar o intervalo de confiança para as médias de **medida** segundo **tipo**

```
sort tipo  
ci medida,by(tipo)
```

Intervalo de confiança para uma média populacional

Construir o intervalo de confiança de 95% para uma amostra de 100 pessoas, média observada igual a 2 e desvio padrão populacional igual a 1,0

```
cii 100 2 1.0
```

Utilizando o banco fem.dta

Teste de uma média populacional

Testar a hipótese de que a média observada da variável **weight** ( $\bar{x}_{obs} = 1,585$ ) é igual à média populacional ( $\mu = 2$ ) de mulheres sem doença mental

```
ttest weight=2
```

Intervalo de confiança para proporção populacional

Construir um intervalo de confiança (exato) para a proporção de pacientes com problemas para dormir

```
tab sleep  
cii 112 0.125
```

Teste de uma proporção populacional

Testar a hipótese de que a proporção de pacientes com problemas para dormir é igual à proporção populacional de pessoas sem doença mental ( $p=0.2$ )

```
recode sleep -99=.
```

```
recode sleep 2=1 1=0
```

```
bitest sleep=0.20
```

ou

```
bitesti 117 14 0.20
```

Teste de associação pelo Qui-quadrado

Verificar a existência de associação entre as variáveis **peso** e **agedic** e entre **peso** e **sleep**

**OBS: se as variáveis peso e agedic já estiverem no banco, considerar apenas os dois últimos comandos. Caso contrário realizar todos os comando abaixo que geram novas variáveis categóricas para peso e idade**

```
gen peso= 1 if weight<0
```

```
replace peso =0 if weight >=0
```

```
tab peso
```

```
label define peso 0 "ganhou peso" 1 "perdeu peso"
```

```
label val peso peso
```

```
tab peso
```

```
sum age
```

```
gen agedic =age
```

```
recode agedic min/37.5=1 37.6/max=2
```

```
label define agedic 1 "<=37.5" 2 "37.6 e mais"
```

```
label val agedic agedic
```

```
tab agedic
```

```
tab peso agedic,col chi
```

```
tab peso sleep,row chi
```

Calcular a correlação entre as variáveis **weight**, **iq** e **age**

```
corr weight iq age
```

ou

```
pwcorr weight iq age,obs sig
```

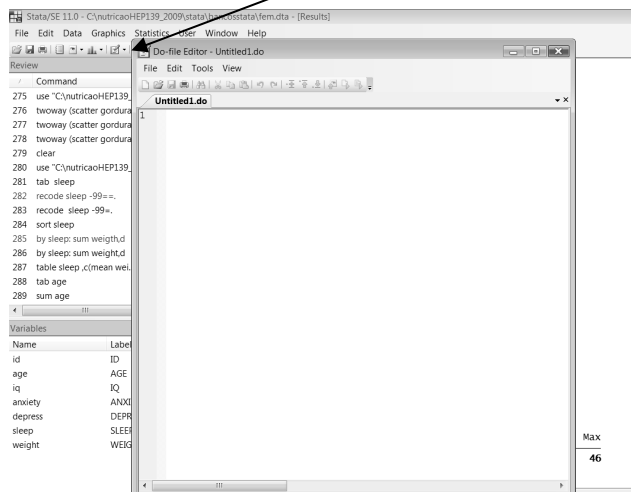
## 10. Introdução à programação

Às vezes é necessário realizar uma análise igual para conjuntos de dados diferentes. Isto é possível realizar armazenando-se os comandos em um arquivo com extensão **.do**, por exemplo **analise1.do**, que pode ser executado com o comando:

```
do <C:\ ..... \analise1.do>
```

ou

utilizando o editor Do-file



A seguir é apresentada uma estrutura básica de um arquivo **do**:

```
clear
set memory 32M
*comentário descrevendo o que o arquivo programa faz*
version
capture log close
log using c:\not2002\analise1, replace text
set more off
use c:\not2002\oswego.dta
desc
tab ill
ci ill
sum age,d
sort ill
ttest age, by(ill)
tab sex ill,row chi
tab vanilla ill,row chi
log close
exit
clear
```

Rodar o programa diretamente pelo ícone **Do-file Editor** ou digitando **do <c:\ ....\analise1.do** na janela de comandos

### Significado de cada linha:

1. Se houver algum banco de dados aberto, o programa começa limpando da memória, os dados utilizados anteriormente.
2. Se for necessária mais memória para a realização do programa.
3. Os asteriscos fazem com que seja ignorado o que está entre eles; são usados para comentários.
4. Especifica a versão do Stata em que o programa foi escrito; útil para futura atualização do programa.
5. O comando **capture** faz com que o *Stata* continue rodando mesmo que ocorra um erro na execução de um comando. O comando **capture log close** fecha o arquivo **log** em uso se for aberto outro ou envia mensagem de erro.
6. O comando **log using filename,replace** abre um arquivo **log** substituindo o já existente.
7. O comando **set more off** faz com que a saída seja apresentada na tela automaticamente sem ter que manualmente instruir o *Stata* para mostrar o que está faltando.
8. O comando abre o arquivo de dados que será utilizado inicialmente.
9. Depois que a análise é feita, o arquivo **.log** é fechado com o comando **log close**.



10. A última linha do programa contendo o comando **exit** não é necessária. Ela é útil para fazer o programa parar de ser rodado.

### Utilizando o Stata/transfer

- Clicar em Iniciar, Programas, Transfer, Statatransfer ou clicar sobre o ícone na área de trabalho;
- Clicar sobre a seta da caixa de diálogo Input File Type e escolher o programa onde o banco de dados foi criado;
- Clicar em Browse para escolher o arquivo (File Specification);
- Automaticamente o transfer oferece o nome e caminho do arquivo em formato dta;
- Clicar sobre a caixa Transfer.
- Notar o nome e caminho onde o arquivo foi armazenado.

