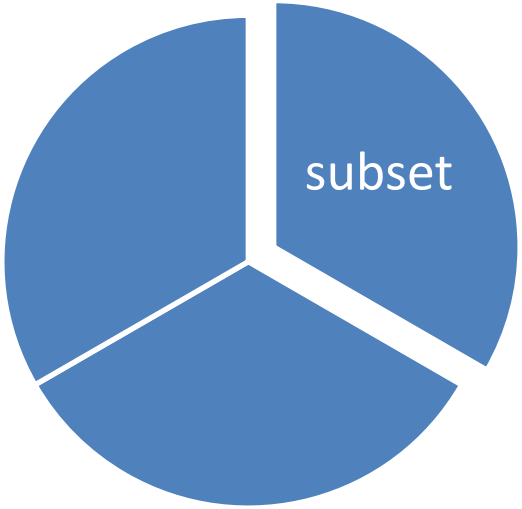
SAS 6 - Subsets in SAS



Types of Subsets

- Retain selected:
  - Variables
  - Observations
    - Conditional
    - Random
    - Range

Two Methods

- Create a new data set $B \subset A$
  - In a DATA step
    - Subsetting IF
    - Data set options

- Use part of an existing data set
  - In a PROC step
    - Data set options

Documentation

- Base SAS
  - SAS 9.2 Language Reference: Dictionary
  - Dictionary of Language Elements
  - Statements
- Base SAS
  - Base SAS 9.2 Procedures Guide
  - Concepts
  - Statements with the Same function in Multiple Procedures

Selecting variables

- Keep (or drop)
  - Within a DATA step

```
libname y "y:\sas\data";
data subset1;
   set y.employees;
   keep salary salbegin;
   run;

proc means; run;
```

“Keep” option on input

- Keep (or drop)
  - As a data set option

```
data subset2;
   set y.employees(keep=salary salbegin);
   run;

proc means; run;
```

“Keep” option on output

- Keep (or drop)
  - As a data set option

```
data subset3(keep=salary salbegin);
   set y.employees;
    salaryincrease = salary - salbegin;
   run;

proc means; run;
```

“Keep” option in a PROC

- Keep (or drop)
  - As a data set option

```
proc means data=y.employees(keep=salary salbegin);
   run;
```

“Keep” details

- In DATA steps
  - As an input data set option, some data is never read
  - As an output data set option, all data is read but some is not output
  - A keep *statement* is the same as an *output* data set option

“Keep” details

- Compare the different output:

```
data subset4; /*input option */
    set y.employees(keep=salary salbegin);
    if gender = "f";
    run;
proc means; run;

data subset5(keep=salary salbegin); /*output option */
    set y.employees;
    if gender = "f";
    run;
proc means; run;

data subset6; /*keep statement */
    set y.employees;
    keep salary salbegin;
    if gender = "f";
    run;
proc means; run;
```

Selecting observations

- Conditional on a data value
  - e.g. keep observations if gender = “m”
- Random
  - e.g. keep a random 10% of observations
- Range of observations
  - e.g. keep the first 100 observations

Conditional selection

- Subsetting IF

```
data subset7;
   set y.employees;
   if gender = "f";
   run;

proc means; var salary; run;
```

Subsetting IF

- Can also be written:

```
data subset7;
   set y.employees;
   if gender = "f" then output;
   run;

proc means; var salary; run;
```

Multiple subsets

- Can be extended:

```
data women men;
   set y.employees;
   if gender = "f" then output women;
   if gender = "m" then output men;
   run;

proc means; var salary; run;
/*Question:  which data set does proc means use? */
```

Conditional selection

- WHERE statement

```
data subset7;
   set y.employees;
   where gender eq "f";
   run;

proc means; var salary; run;
```

Where

- WHERE data set option

```
data subset7;
   set y.employees(where=(gender eq "m"));
   run;

proc means; var salary; run;
```

WHERE vs. IF

- WHERE uses existing data values; IF can calculate new values
- WHERE can be a statement or an option; IF can only be a statement
- WHERE can be used with PROCS (both as a statement and a data set option); IF can only be used in a DATA step

More WHERE and IF examples

```
data subset10; /* Not possible with WHERE */
   set y.employees;
   if salary - salbegin > 15000;
   run;

proc means data=y.employees(where=(gender eq "m"));
var salary;
run;

proc means data=y.employees;
where gender eq "m"; /* as a statement */
var salary;
run;
```

Random Samples

- Simple random sampling
  - Proportion (approximate)
  - Number of cases (exact)

Procedure

- Generate random values from a uniform distribution
- Keep a proportion, or keep a specified n

Random sampling by proportion

```
data random;
   set y.employees;
   x = ranuni(-1); /* seed with clock time */
   if x lt .1;
   drop x;
   run;
proc means; var salary; run;
```

Random sampling for N

```
data temp;
   set y.employees;
   x = ranuni(-1); /* seed with clock time */
   run;

proc sort;
   by x;
   run;

data random1;
   set temp(drop=x obs=40);
   run;
proc means; var salary; run;

* There are other, more efficient, algorithms for this;
```

Ranges

- Sampling by observation range
  - Data set options `firstobs=` and `obs=`
  - Automatic variable `_N_`