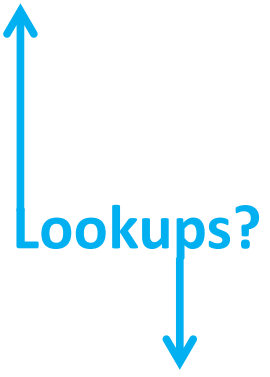
What's Up with Table Lookups?



Christopher Bost

Challenge

Look up **median height** by SEX and AGE for each student in SASHELP.CLASS

Data set HEIGHTS is the "lookup table"

	sex	age11	age12	age13	age14	age15	age16
1	F	56.8	59.6	61.9	63.2	63.7	64.0
2	M	56.6	58.8	61.6	64.6	67.0	68.4

Note: Heights are in inches. Source: www.cdc.gov/growthcharts

Goal

	Name	Sex	Age	Height	Weight	medht
1	Alfred	M	14	69	112.5	64.6
2	Alice	F	13	56.5	84	61.9
3	Barbara	F	13	65.3	98	61.9
4	Carol	F	14	62.8	102.5	63.2
5	Henry	M	14	63.5	102.5	64.6
6	James	M	12	57.3	83	58.8
7	Jane	F	12	59.8	84.5	59.6
8	Janet	F	15	62.5	112.5	63.7
9	Jeffrey	M	13	62.5	84	61.6
10	John	M	12	59	99.5	58.8
11	Joyce	F	11	51.3	50.5	56.8
12	Judy	F	14	64.3	90	63.2
13	Louise	F	12	56.3	77	59.6
14	Mary	F	15	66.5	112	63.7
15	Philip	M	16	72	150	68.4
16	Robert	M	12	64.8	128	58.8
17	Ronald	M	15	67	133	67
18	Thomas	M	11	57.5	85	56.6
19	William	M	15	66.5	112	67

Methods

1. IF-THEN-ELSE
2. Restructure and MERGE
3. Restructure and hash
4. Informat and INPUT
5. Two-dimensional array

1. IF-THEN-ELSE

1. IF-THEN-ELSE

```
data class1;
set sashelp.class;
      if sex='F' and age=11 then medht=56.8;
else if sex='F' and age=12 then medht=59.6;
else if sex='F' and age=13 then medht=61.9;
else if sex='F' and age=14 then medht=63.2;
else if sex='F' and age=15 then medht=63.7;
else if sex='F' and age=16 then medht=64.0;
else if sex='M' and age=11 then medht=56.6;
else if sex='M' and age=12 then medht=58.8;
else if sex='M' and age=13 then medht=61.6;
else if sex='M' and age=14 then medht=64.6;
else if sex='M' and age=15 then medht=67.0;
else if sex='M' and age=16 then medht=68.4;
run;
```

1. IF-THEN-ELSE

```
data class1a;
set sashelp.class;
if sex='F' then do;
     if age=11 then medht=56.8;
else if age=12 then medht=59.6;
else if age=13 then medht=61.9;
else if age=14 then medht=63.2;
else if age=15 then medht=63.7;
else if age=16 then medht=64.0;
end;
else if sex='M' then do;
     if age=11 then medht=56.6;
else if age=12 then medht=58.8;
else if age=13 then medht=61.6;
else if age=14 then medht=64.6;
else if age=15 then medht=67.0;
else if age=16 then medht=68.4;
end;
run;
```

	Name	Sex	Age	Height	Weight	medht
1	Alfred	M	14	69	112.5	64.6
2	Alice	F	13	56.5	84	61.9
3	Barbara	F	13	65.3	98	61.9
4	Carol	F	14	62.8	102.5	63.2
5	Henry	M	14	63.5	102.5	64.6
6	James	M	12	57.3	83	58.8
7	Jane	F	12	59.8	84.5	59.6
8	Janet	F	15	62.5	112.5	63.7
9	Jeffrey	M	13	62.5	84	61.6
10	John	M	12	59	99.5	58.8
11	Joyce	F	11	51.3	50.5	56.8
12	Judy	F	14	64.3	90	63.2
13	Louise	F	12	56.3	77	59.6
14	Mary	F	15	66.5	112	63.7
15	Philip	M	16	72	150	68.4
16	Robert	M	12	64.8	128	58.8
17	Ronald	M	15	67	133	67
18	Thomas	M	11	57.5	85	56.6
19	William	M	15	66.5	112	67

Pros | Cons

- It works
- Impractical with many values

2. Restructure and MERGE

2. Restructure and MERGE

```
data multiple;
set heights;
array hts{11:16} age11-age16;
do age=11 to 16;
  medht=hts{age};
  output;
end;
keep sex age medht;
run;
```

- Specify lower and upper bound of array `{11:16}`
- Create AGE as array index

	sex	age	medht
1	F	11	56.8
2	F	12	59.6
3	F	13	61.9
4	F	14	63.2
5	F	15	63.7
6	F	16	64
7	M	11	56.6
8	M	12	58.8
9	M	13	61.6
10	M	14	64.6
11	M	15	67
12	M	16	68.4

2. Restructure and MERGE

```
proc sort data=sashelp.class
          out=class_sort;
by sex age;
run;

proc sort data=multiple
          out=multiple_sort;
by sex age;
run;

data class2;
merge class_sort(in=inclass)
      multiple_sort;
by sex age;
if inclass;
run;
```

	Name	Sex	Age	Height	Weight	medht
1	Joyce	F	11	51.3	50.5	56.8
2	Jane	F	12	59.8	84.5	59.6
3	Louise	F	12	56.3	77	59.6
4	Alice	F	13	56.5	84	61.9
5	Barbara	F	13	65.3	98	61.9
6	Carol	F	14	62.8	102.5	63.2
7	Judy	F	14	64.3	90	63.2
8	Janet	F	15	62.5	112.5	63.7
9	Mary	F	15	66.5	112	63.7
10	Thomas	M	11	57.5	85	56.6
11	James	M	12	57.3	83	58.8
12	John	M	12	59	99.5	58.8
13	Robert	M	12	64.8	128	58.8
14	Jeffrey	M	13	62.5	84	61.6
15	Alfred	M	14	69	112.5	64.6
16	Henry	M	14	63.5	102.5	64.6
17	Ronald	M	15	67	133	67
18	William	M	15	66.5	112	67
19	Philip	M	16	72	150	68.4

sorted by
sex, **age**

Pros | Cons

- Simple restructuring
- Ordinary match-merge
- Must sort source data

2. ~~Restructure and~~ MERGE

```
proc sort data=sashelp.class
          out=class_sort;
by sex;
run;

proc sort data=heights
          out=heights_sort;
by sex;
run;

data class2a;
merge class_sort(in=inclass)
      heights_sort;
by sex;
if inclass;
array hts{11:16} age11-age16;
medht=hts{age};
drop age11-age16;
run;
```

	Name	Sex	Age	Height	Weight	medht
1	Alice	F	13	56.5	84	61.9
2	Barbara	F	13	65.3	98	61.9
3	Carol	F	14	62.8	102.5	63.2
4	Jane	F	12	59.8	84.5	59.6
5	Janet	F	15	62.5	112.5	63.7
6	Joyce	F	11	51.3	50.5	56.8
7	Judy	F	14	64.3	90	63.2
8	Louise	F	12	56.3	77	59.6
9	Mary	F	15	66.5	112	63.7
10	Alfred	M	14	69	112.5	64.6
11	Henry	M	14	63.5	102.5	64.6
12	James	M	12	57.3	83	58.8
13	Jeffrey	M	13	62.5	84	61.6
14	John	M	12	59	99.5	58.8
15	Philip	M	16	72	150	68.4
16	Robert	M	12	64.8	128	58.8
17	Ronald	M	15	67	133	67
18	Thomas	M	11	57.5	85	56.6
19	William	M	15	66.5	112	67

sorted by **sex**

3. Restructure and hash

3. Restructure and hash

```
data multiple;
set heights;
array hts{11:16} age11-age16;
do age=11 to 16;
  medht=hts{age};
  output;
end;
keep sex age medht;
run;
```

	sex	age	medht
1	F	11	56.8
2	F	12	59.6
3	F	13	61.9
4	F	14	63.2
5	F	15	63.7
6	F	16	64
7	M	11	56.6
8	M	12	58.8
9	M	13	61.6
10	M	14	64.6
11	M	15	67
12	M	16	68.4

- Specify lower and upper bound of array `{11:16}`
- Create AGE as array index

[same restructuring as on Slide 10]

3. Restructure and hash

```
data class3;

if 0 then set sashelp.class;

if _N_ = 1 then do;
  declare hash
  lookup(dataset:'multiple');
  lookup.defineKey('sex','age');
  lookup.defineData('medht');
  lookup.defineDone();
  call missing(medht);
end;

set sashelp.class;

rc=lookup.find();

run;
```

- never executes but preserves variable order during compile phase [optional]
- does something once at the beginning
- creates hash table named **lookup**
- loads data from data set **multiple**
- defines key ["lookup" variable(s)]
- defines additional variable(s)
- ends definition of hash table
- eliminates *uninitialized* note

- loads observation from data set

- looks up values of **sex** and **age** from current obs in hash table
- if finds key values, adds the respective value of **medht** to obs

3. Restructure and hash

	Name	Sex	Age	Height	Weight	medht	rc
1	Alfred	M	14	69	112.5	64.6	0
2	Alice	F	13	56.5	84	61.9	0
3	Barbara	F	13	65.3	98	61.9	0
4	Carol	F	14	62.8	102.5	63.2	0
5	Henry	M	14	63.5	102.5	64.6	0
6	James	M	12	57.3	83	58.8	0
7	Jane	F	12	59.8	84.5	59.6	0
8	Janet	F	15	62.5	112.5	63.7	0
9	Jeffrey	M	13	62.5	84	61.6	0
10	John	M	12	59	99.5	58.8	0
11	Joyce	F	11	51.3	50.5	56.8	0
12	Judy	F	14	64.3	90	63.2	0
13	Louise	F	12	56.3	77	59.6	0
14	Mary	F	15	66.5	112	63.7	0
15	Philip	M	16	72	150	68.4	0
16	Robert	M	12	64.8	128	58.8	0
17	Ronald	M	15	67	133	67	0
18	Thomas	M	11	57.5	85	56.6	0
19	William	M	15	66.5	112	67	0

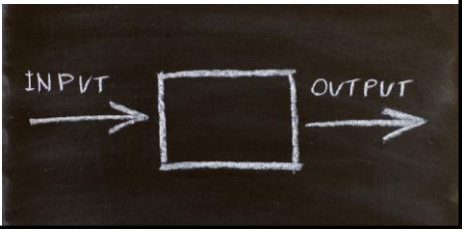
original order

0 = match

Pros | Cons

- Simple restructuring
- No sorting required
- Learn “hash syntax”

4. Informat and INPUT



4. Informat and INPUT

```
data myinformat;
retain fmtname 'AGEMEDIAN' type 'I';
length start $ 3 label $ 4;
set heights;
array hts{11:16} age11-age16;
do age=11 to 16;
  start=cats(sex,age);
  label=put(hts{age},4.1);
  output;
end;
keep fmtname type start label;
run;
```

	fmtname	type	start	label
1	AGEMEDIAN	I	F11	56.8
2	AGEMEDIAN	I	F12	59.6
3	AGEMEDIAN	I	F13	61.9
4	AGEMEDIAN	I	F14	63.2
5	AGEMEDIAN	I	F15	63.7
6	AGEMEDIAN	I	F16	64.0
7	AGEMEDIAN	I	M11	56.6
8	AGEMEDIAN	I	M12	58.8
9	AGEMEDIAN	I	M13	61.6
10	AGEMEDIAN	I	M14	64.6
11	AGEMEDIAN	I	M15	67.0
12	AGEMEDIAN	I	M16	68.4

- Builds **input control data set** to create an informat
- FMTNAME, START, and LABEL are required variables
- TYPE value of 'I' is required to create an informat
- CATS strips leading/trailing blanks and concatenates

4. Informat and INPUT

```
proc format cntlin=myinformat fmtlib;
run;
```

```
|                    INFORMAT NAME: @AGEMEDIAN LENGTH: 3                     |
|   MIN LENGTH:   1  MAX LENGTH:  40  DEFAULT LENGTH:   3  FUZZ:        0   |
|----------------------------------------------------------------------------|
|START           |END             |INVALUE(VER. 9.4     19MAR2015:22:43:10)|
|----------------+----------------+---------------------------------------|
|F11             |F11             |                                   56.8|
|F12             |F12             |                                   59.6|
|F13             |F13             |                                   61.9|
|F14             |F14             |                                   63.2|
|F15             |F15             |                                   63.7|
|F16             |F16             |                                     64|
|M11             |M11             |                                   56.6|
|M12             |M12             |                                   58.8|
|M13             |M13             |                                   61.6|
|M14             |M14             |                                   64.6|
|M15             |M15             |                                     67|
|M16             |M16             |                                   68.4|
```


4. Informat and INPUT

```
data class4;
set sashelp.class;
medht=input(cats(sex,age),agemedian.);
run;
```

	Name	Sex	Age	Height	Weight	medht
1	Alfred	M	14	69	112.5	64.6
2	Alice	F	13	56.5	84	61.9
3	Barbara	F	13	65.3	98	61.9
4	Carol	F	14	62.8	102.5	63.2
5	Henry	M	14	63.5	102.5	64.6
6	James	M	12	57.3	83	58.8
7	Jane	F	12	59.8	84.5	59.6
8	Janet	F	15	62.5	112.5	63.7
9	Jeffrey	M	13	62.5	84	61.6
10	John	M	12	59	99.5	58.8
11	Joyce	F	11	51.3	50.5	56.8
12	Judy	F	14	64.3	90	63.2
13	Louise	F	12	56.3	77	59.6
14	Mary	F	15	66.5	112	63.7
15	Philip	M	16	72	150	68.4
16	Robert	M	12	64.8	128	58.8
17	Ronald	M	15	67	133	67
18	Thomas	M	11	57.5	85	56.6
19	William	M	15	66.5	112	67

Pros | Cons

- Creates data-driven informat
- Converts values with INPUT
- No sorting is required
- Need input control data set
- Not obvious or intuitive

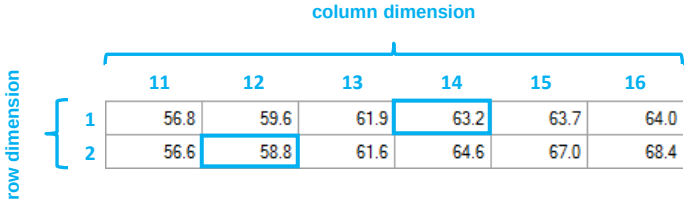
5. Two-dimensional array

5. Two-dimensional array

```
data class5a;
array hts{1:2,11:16} _temporary_
  (56.8 59.6 61.9 63.2 63.7 64.0
   56.6 58.8 61.6 64.6 67.0 68.4);
set sashelp.class;
if sex='F' then sexN=1;
else if sex='M' then sexN=2;
medht=hts{sexN,age};
drop sexN;
run;
```

- Creates temporary array HTS
- Defines 2 rows and 6 columns
- Hard codes all lookup values
- SexN = row , Age = column

5. Two-dimensional array



row , col

```
medht=hts{sexN,age};
```

Examples

SEXN=1 and AGE=14 looks up Row 1 and Column 14 = 63.2

SEXN=2 and AGE=12 looks up Row 2 and Column 12 = 58.8

5. Two-dimensional array

```
data class5;
array hts{1:2,11:16} _temporary_;
if _n_=1 then do s=1 to 2;
  set heights;
  array ages{11:16} age11-age16;
  do a=11 to 16;
    hts{s,a}=ages{a};
  end;
end;
set sashelp.class;
if sex='F' then sexN=1;
else if sex='M' then sexN=2;
medht=hts{sexN,age};
drop s age11-age16 a sexN;
run;
```

	sex	Name	Age	Height	Weight	medht
1	M	Alfred	14	69	112.5	64.6
2	F	Alice	13	56.5	84	61.9
3	F	Barbara	13	65.3	98	61.9
4	F	Carol	14	62.8	102.5	63.2
5	M	Henry	14	63.5	102.5	64.6
6	M	James	12	57.3	83	58.8
7	F	Jane	12	59.8	84.5	59.6
8	F	Janet	15	62.5	112.5	63.7
9	M	Jeffrey	13	62.5	84	61.6
10	M	John	12	59	99.5	58.8
11	F	Joyce	11	51.3	50.5	56.8
12	F	Judy	14	64.3	90	63.2
13	F	Louise	12	56.3	77	59.6
14	F	Mary	15	66.5	112	63.7
15	M	Philip	16	72	150	68.4
16	M	Robert	12	64.8	128	58.8
17	M	Ronald	15	67	133	67
18	M	Thomas	11	57.5	85	56.6
19	M	William	15	66.5	112	67

Pros | Cons

- Creates array in memory
- Values loaded only once
- Uses a single DATA step
- No sorting is required
- Conceptually challenging
- Requires sequential index

Conclusions

- There are many ways to do table lookups
- Additional methods are not covered here
- Methods vary in efficiency and complexity
- The best approach will depend on your
  - Lookup table
  - File sizes
  - Infrastructure
- Keep these techniques in your **SAS toolkit**

Contact info

mdrc
BUILDING KNOWLEDGE TO IMPROVE SOCIAL POLICY

CHRISTOPHER J. BOST
Director, Research Technology Unit

16 East 34th Street
New York, NY 10016-4326
www.mdrc.org

Tel: 212 340 8613
Fax: 212 684 0832
christopher.bost@mdrc.org