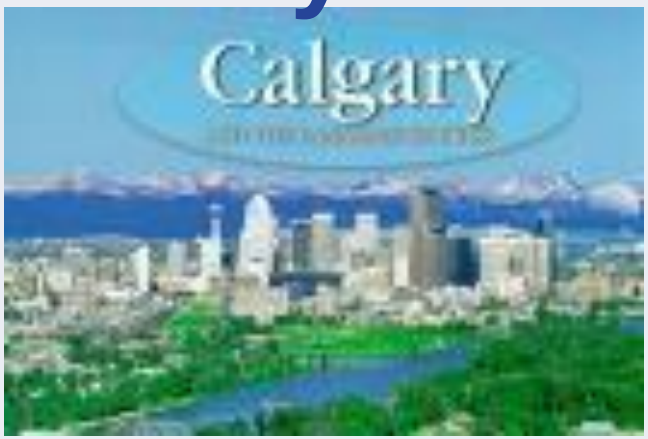
The SAS® Hash Object: It's Time To .find() Your Way Around



Peter Eberhardt
Fernwood Consulting Group Inc.

Agenda

Introduction

Table Lookups in SAS

Declaring your HASH Object

Basic Usage of the HASH Object

Review

Introduction

- Practical Examples
  - DECLARING a HASH Object
  - Populating a HASH Object
  - Using a HASH Object

Introduction

What is a HASH Object?

- Memory resident data structure
  - Key
  - Data
  - Methods
  - Attributes

Introduction

What is a HASH Object?

- DATA step only
- Transient
- Run time

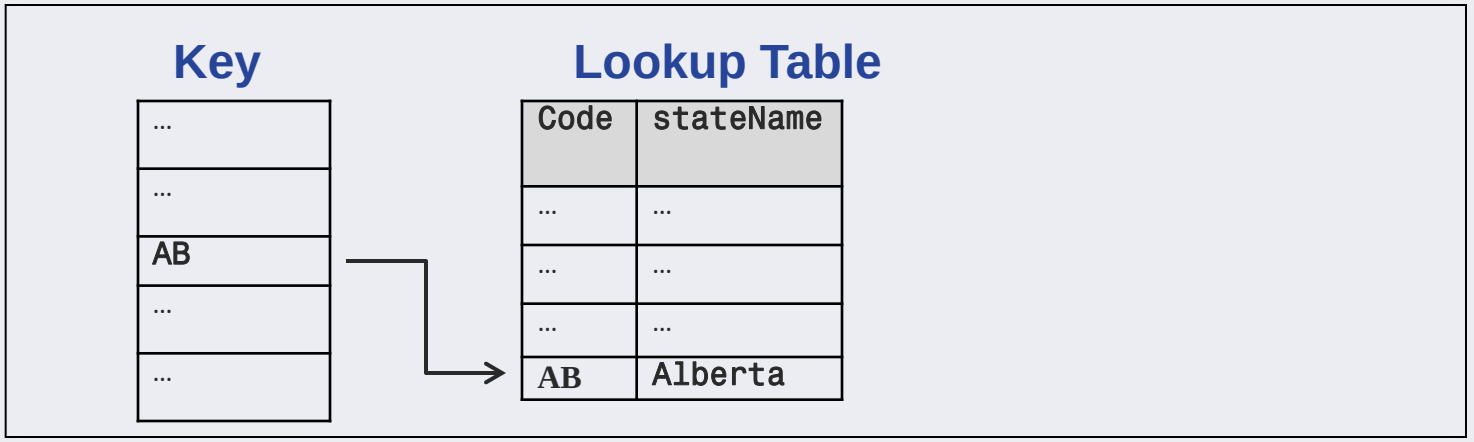
Agenda

- ☑ *Introduction*
- Table Lookups in SAS
- Declaring your HASH Object
- Basic Usage of the HASH Object
- Review

Lookups In SAS

What is a Table Lookup

- A method of transforming one value (KEY) into another

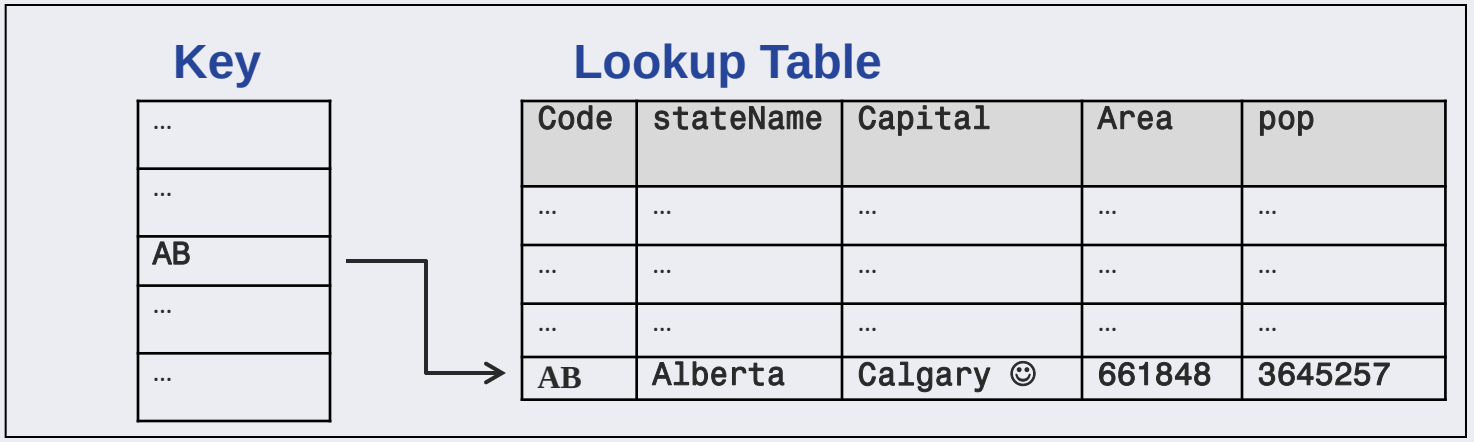


Peter Eberhardt, Fernwood Consulting Group Inc.

Lookups In SAS

What is a Table Lookup

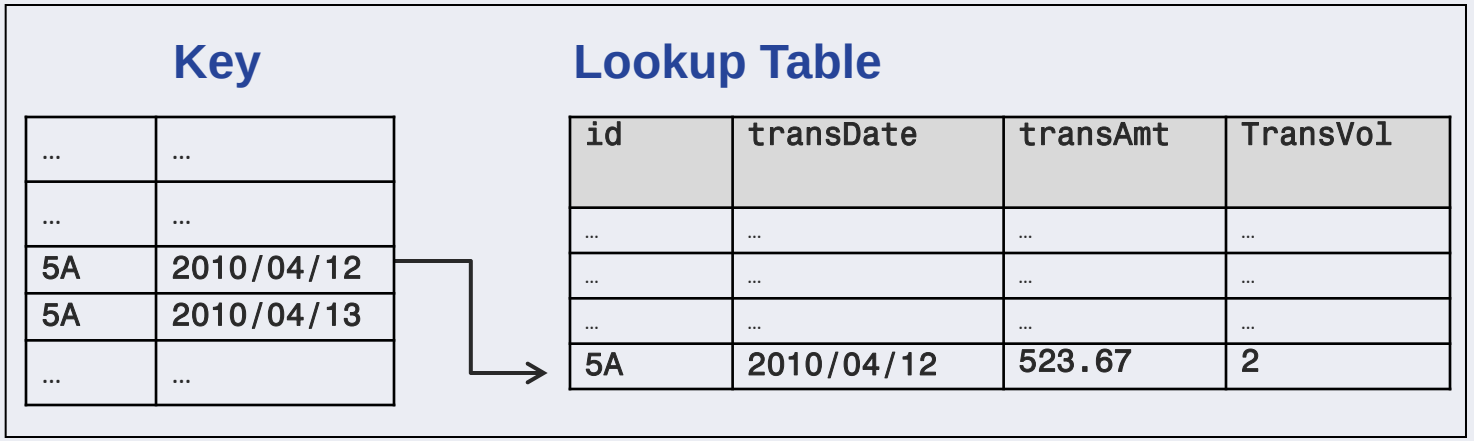
- A method of transforming one value (KEY) into another
- The lookup can be a compound value



Lookups In SAS

What is a Table Lookup

- A method of transforming one value (KEY) into another
- The KEY can be a compound value



Peter Eberhardt, Fernwood Consulting Group Inc.

Lookups In SAS

In SAS DATA step:

1. KEY
2. LOOKUP TABLE

```
PROC sort data= custDates; by ID transDate; run;
PROC sort data= custTrans; by ID transDate; run;
DATA custValues;
    merge          ❶ custDates
                   ❷ custTrans;
    by ID transDate;
    keep ID transDate transAmt transVol;
run;
```

Lookups In SAS

In SAS PROC SQL:

1. KEY
2. LOOKUP TABLE

```
PROC SQL;
        create table custValues as
        select ct.*
        from    ❶ custDates as cd inner join
                ❷ custTrans as ct
        on  cd.ID         = ct.ID
        and ct.transDate = ct.transDate
        order by ct.ID, ct.transDAte;
QUIT;
```

Lookups In SAS

Popular alternative for simple keys

- Format lookup
  - capital = put('WI', capitals.);
    - Key cannot be compound
    - Result cannot be compound

HASH Object

Lookups In SAS

Data we will be using:

- Fee Codes
- Doctors
- Patients
- Transactions

Lookups In SAS

FEECODES Table:

Variables in Creation Order						
#	Variable	Type	Len	Format	Informat	Label
1	section	Char	1			
2	subSection	Num	8	11.	11.	Sub Section
3	**feecode**	Char	4			
4	feeAmount	Num	8	COMMA8.2		Fee Amount

Lookups In SAS

DOCTORS Table:

Variables in Creation Order						
#	Variable	Type	Len	Format	Informat	Label
1	DID	Num	8			
2	postcode	Char	6	$6.	$6.	postcode
3	dob	Num	8	YYMMDD10.		
4	sex	Char	1	$1.	$1.	sex

Lookups In SAS

PATIENTS Table:

Variables in Creation Order						
#	Variable	Type	Len	Format	Informat	Label
1	**PID**	Num	8			
2	studyID	Char	10			
3	postcode	Char	10	$10.	$10.	postcode
4	dob	Num	8	YYMMDD10.	YYMMDD10.	dob
5	sex	Char	1	$1.	$1.	sex

Lookups In SAS

TRANSACTIONS Table:

Variables in Creation Order						
#	Variable	Type	Len	Format	Informat	Label
1	PID	Num	8			
2	DID	Num	8			
3	visitdate	Num	8	YYMMDD10.		
4	feecode	Char	4	$4.	$4.	feecode

Agenda

- ☑ *Introduction*
- ☑ Table *Lookups in SAS*
- Declaring your HASH Object
- Basic Usage of the HASH Object
- Review

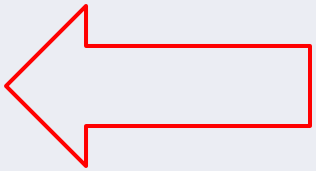
DECLARING Your HASH Object

Five basic steps:

1. DECLARE the object:
2. DEFINE the hash key.
3. DEFINE the data variables.
4. Complete the definitions
5. Load data into a hash object

DECLARING Your HASH Object

```
DATA exercise01;
     length feecode    $4.
            section    $1.
            subSection 8.
            feeAmount  8.
            ;
```



All the variables used in the HASH Object must be defined BEFORE you DECLARE the HASH Object

1. `DECLARE hash feecodes() ;`
2. `rc=feecodes.defineKey("feecode");`
3. `rc=feecodes.defineData("section","subsection","feeAmount");`
4. `rc=feecodes.defineDone();`

DECLARING Your HASH Object

```
      do while (not done);
         set data.feecodes end=done;
5        rc = feecodes.add();
         if rc NE 0
         then
           do;
              put "Problem with .add()."
                  feecode= section= subsection= feeAmount= rc=;
           end;
      end;
      STOP;
RUN ;
```

DECLARING Your HASH Object

1. DECLARING the HASH Object
   - ***DECLARE hash feecodes();***

     - DECLARE is telling SAS you want an object
       - Can be shortened to DCL

     - hash is the type of object

     - feecodes is the name of the HASH object
       - I tend to give the HASH object the same name as the underlying dataset it is using

DECLARING Your HASH Object

1. DECLARING the HASH Object
   - ***DECLARE hash feecodes();***

     - The parenthesis tell SAS we want to also instantiate (create) the object

     - There are a number of options that can be used in the DECLARE statement.
       - See the online help for a complete list

DECLARING Your HASH Object

2. Define the HASH key
   - ***rc = feecodes.defineKey("feecode");***
     - feecodes is the name of the HASH object
     - .defineKey() is the method
     - "feecode" is the string with the name of the key variable
       - NOTE: this is a character string, not the actual variable.

DECLARING Your HASH Object

2. Define the HASH key
   - ***rc = feecodes.defineKey("feecode");***

     - You can have more than one variable as the key. The key variables can be character, numeric, or a combination of character and numeric.

DECLARING Your HASH Object

3. Define the data variables
   - ***rc = feecodes.defineData("section", "subSection", "feeAmount");***
     - feecodes is the name of the HASH object
     - .defineData() is the method
     - "section", "subSection", and "feeAmount" are the strings with the names of the data variable
       - NOTE: these are character strings, not the actual variables.

DECLARING Your HASH Object

4. Complete the definintions
   - ***rc = feecodes.defineDone();***
     - feecodes is the name of the HASH object.
     - defineDone() is the method.

DECLARING Your HASH Object

5. Adding items to the HASH Object
   - ***rc = feecodes.add();***
     - feecodes is the name of the HASH object
     - .add() is the method
     - All data elements from .defineKey() and .defineData() that are currently in the PDV are added to the HASH object.

DECLARING Your HASH Object

5. Adding items to the HASH Object
   - ***rc = feecodes.add();***

     - What if the KEY value does not exist in the HASH?
       - Item added
       - rc = 0

     - What if the KEY value already exists?
       - Item NOT added
       - rc ~= 0

     - With .add() **FIRST** in "wins"

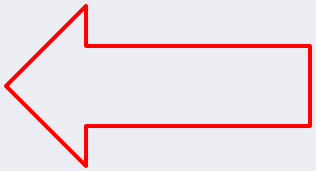
DECLARING Your HASH Object

Five basic steps:

1. DECLARE the object: ***DECLARE hash***
2. DEFINE the hash key: ***.defineKey()***
3. DEFINE the data variables: ***.defineData()***
4. Complete the definitions: ***.defineDone()***
5. Load data into a hash object ***.add()***

DECLARING Your HASH Object

```
DATA exercise01;
     length feecode    $4.
            section    $1.
            subSection 8.
            feeAmount  8.
            ;
```



All the variables used in the HASH Object must be defined BEFORE you DECLARE the HASH Object

1. `DECLARE hash feecodes() ;`
2. `rc=feecodes.defineKey("feecode");`
3. `rc=feecodes.defineData("section","subsection","feeAmount");`
4. `rc=feecodes.defineDone();`

DECLARING Your HASH Object

```
      do while (not done);
         set data.feecodes end=done;
5        rc = feecodes.add();
         if rc NE 0
         then
           do;
              put "Problem with .add()."
                  feecode= section= subsection= feeAmount= rc=;
           end;
      end;
      STOP;
RUN ;
```

DECLARING Your HASH Object

Five basic steps:

1. DECLARE the object: ***DECLARE hash***
2. DEFINE the hash key: ***.defineKey()***
3. DEFINE the data variables: ***.defineData()***
4. Complete the definitions: ***.defineDone()***
5. Load data into a hash object ***.add()***

DECLARING Your HASH Object

Five basic steps:

1. DECLARE the object: ***DECLARE hash***
2. DEFINE the hash key: ***.defineKey()***
3. DEFINE the data variables: ***.defineData()***
4. Complete the definitions: ***.defineDone()***
5. ***Load data into a hash object .add()***

DECLARING Your HASH Object

```
      do while (not done);
         set data.feecodes end=done;
5        rc = feecodes.add();
         if rc NE 0
         then
           do;
              put "Problem with .add()."
                  feecode= section= subsection= feeAmount= rc=;
           end;
      end;
      STOP;
RUN ;
```

DECLARING Your HASH Object

5. Adding items to the HASH Object
   - ***rc = feecodes.add();***

     - What if the KEY value does not exist in the HASH?
       - Item added
       - rc = 0

     - What if the KEY value already exists?
       - Item NOT added
       - rc ~= 0

     - With .add() **FIRST** in "wins"

DECLARING Your HASH Object

```
    do while (not done);
       set data.feecodes end=done;
       rc = feecodes.replace();
       if rc NE 0
       then
         do;
            put "Problem with .replace()."
                feecode= section= subsection= feeAmount= rc=;
         end;
    end;
    STOP;
RUN ;
```

DECLARING Your HASH Object

5. Adding items to the HASH Object
   - ***rc = feecodes.replace();***
     - feecodes is the name of the HASH object
     - .replace() is the method
     - All data elements from .defineKey() and .defineData() that are currently in the PDV are added to the HASH object.

DECLARING Your HASH Object

5. Adding items to the HASH Object
   - ***rc = feecodes.replace();***
     - What if the KEY value does not exist?
       - Item added
       - rc = 0
     - What if the KEY value already exists?
       - Item replaced
       - rc = 0
     - With .replace() **LAST** in "wins"

DECLARING Your HASH Object

Five basic steps:

1. DECLARE the object: ***DECLARE hash***
5. ***Load data into a hash object DATASET:***
2. DEFINE the hash key: ***.defineKey()***
3. DEFINE the data variables: ***.defineData()***
4. Complete the definitions: ***.defineDone()***

DECLARING Your HASH Object

```
DATA exercise03;
     length feecode    $4.
            section    $1.
            subSection 8.
            feeAmount  8.
            ;
     DECLARE hash  feecodes(DATASET:'data.feecodes') ;
       rc=feecodes.defineKey("feecode");
       rc=feecodes.defineData("section", "subsection", "feeAmount");
       rc=feecodes.defineDone();
     STOP;
RUN ;
```

DECLARING Your HASH Object

5. Adding items to the HASH Object
   - ***DECLARE hash feecodes(DATASET:"data.feecodes");***
     - "data.feecodes" is the string with the name of the dataset to load into the HASH.
     - Acts like .add() - **FIRST** in "wins"

DECLARING Your HASH Object

```
DATA exercise03;
     length feecode    $4.
            section    $1.
            subSection 8.
            feeAmount  8.
            ;
     DECLARE hash  feecodes(DATASET:'data.feecodes', ,
                            DUPLICATE:'replace') ;
        rc=feecodes.defineKey("feecode");
        rc=feecodes.defineData("section","subsection","feeAmount");
        rc=feecodes.defineDone();
     STOP;
RUN ;
```

DECLARING Your HASH Object

5. Adding items to the HASH Object
   - ***DECLARE hash feecodes(DATASET:"data.feecodes",***
     DUPLICATE:"replace");
     - Acts like .replace() - **LAST** in "wins"
     - Can use: DUPLICATE:'Y'

DECLARING Your HASH Object

What if there are changes in the length or type of the Key or Data items?

```
length feecode   $4.
       section   $1.
       subSection 8.
       feeAmount  8.
       ;
```

DECLARING Your HASH Object

What if there are changes in the length or type of the Key or Data items?

```
length feecode    $6.
       section    $1.
       subSection 8.
       feeAmount  8.
       ;
```

Program maintenance issue

- All DATA steps using a HASH with feecode need to be changed.

DECLARING Your HASH Object

```
DATA exercise03;
    if _n_ = 0 then set data.feecodes;
    DECLARE hash  feecodes(DATASET:'data.feecodes') ;
      rc=feecodes.defineKey("feecode");
      rc=feecodes.defineData("section","subsection","feeAmount");
      rc=feecodes.defineDone();
    STOP;
RUN ;
```

DECLARING Your HASH Object

What if there are changes in the length or type of the Key or Data items?

- ***if _n_ = 0 then set data.feecodes;***
  - DATA step compiler opens the table and reads the metadata, bringing the column names, type, and length into the Program Data Vector (PDV)
  - Condition _n_ = 0 is never met so no records are read from the table

Program maintenance issue

- **Disappears**

DECLARING Your HASH Object

Five basic steps:

1. DECLARE the object: ***DECLARE hash***
2. DEFINE the hash key: ***.defineKey()***
3. DEFINE the data variables: ***.defineData()***
4. Complete the definitions: ***.defineDone()***
5. Load data into a hash object ***.add()***
   .replace()
   DATASET:

Agenda

- ☑ *Introduction*
- ☑ Table *Lookups in SAS*
- ☑ *Declaring your HASH Object*

Basic Usage of the HASH Object

Review

Basic Usage of the HASH Object

Lookups Revisited

- Transactions table has codes for
  - feecode
  - Doctor ID (DID)
  - Patient ID (PID)

Basic Usage of the HASH Object

Lookup using the HASH object

```
* now we need to read each record in the transactions table;
     do while (not done);
        set data.transactions end=done;
        rc = feecodes.find();
         output;
     end;
```


Peter Eberhardt, Fernwood Consulting Group Inc.

Basic Usage of the HASH Object

Lookup using the HASH object

- ***rc = feecodes.find();***
  - Uses the current value of *feecode* from *data.transactions*
  - If the feecode is in the HASH object
    - rc = 0
    - All the items in ***defineData()*** are brought to the DATA step
    - **Any previous values of these data are overwritten**

Basic Usage of the HASH Object

Lookup using the HASH object

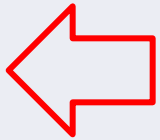
- ***rc = feecodes.find();***
  - If the feecode is NOT in the HASH object
    - rc ~= 0
    - No existing data are overwritten

If you do not check the return code you have a potential data integrity problem

Basic Usage of the HASH Object

Lookup using the HASH object

```
* now we need to read each record in the transactions table;
     do while (not done);
        set data.transactions end=done;
        rc = feecodes.find();
         output;
     end;
```



If the feecode was not found then the values of *section*, *subSection* and *feeAmount* from the last found feecode would be incorrectly saved with the current feecode

Basic Usage of the HASH Object

Lookup using the HASH object

```
* now we need to read each record in the transactions table;
     do while (not done);
        set data.transactions end=done;
        rc = feecodes.find();
        if rc = 0
           then output;
            else /* do something to handle the missing code */
   end;
```

Basic Usage of the HASH Object

What if you only want to know if the value exists in the HASH Object?

```
* now we need to read each record in the transactions table;
     do while (not done);
        set data.transactions end=done;
        rc = patients.check();
        if rc = 0
           then /* do something when patient is in table */;
           else /* do something when patient is NOT in table  */
         output;
     end;
```

Basic Usage of the HASH Object

What if you only want to know if the value exists in the HASH?

- ***rc = patients.check();***
  - If the patient ID is in the HASH object
    - rc = 0
  - If the patient ID is NOT in the HASH object
    - rc ~= 0

No variables are brought into the DATA step.

Basic Usage of the HASH Object

.find();

- Variables are brought into the DATA step

.check();

- No variables are brought into the DATA step

Basic Usage of the HASH Object

Multiple HASH Objects.

```
DATA exercise05
     exercise05_missingFC (keep=feecode)
     exercise05_missingDOC(keep=did);;
   if _n_ = 0
     then
       do;
         set data.feecodes;
         set data.doctors;
       end;
```


Peter Eberhardt, Fernwood Consulting Group Inc.

Basic Usage of the HASH Object

```
DECLARE hash  feecodes(DATASET:'data.feecodes') ;
        rc=feecodes.defineKey('feecode');
        rc=feecodes.defineData('section','subSection','feeAmount');
        rc=feecodes.defineDone();
DECLARE hash  doctors(DATASET:'data.doctors') ;
        rc=doctors.defineKey('did');
        rc=doctors.defineData('dob');
        rc=doctors.defineDone();
```

Basic Usage of the HASH Object

```
* now we need to read each record in the transactions table;
   do while (not done);
        set data.transactions end=done;
        rcFC = feecodes.find();
        rcDOC = doctors.find();
        doctorDOB = dob;

        output exercise05;
```

Basic Usage of the HASH Object

```
        if rcFC  NE 0 then output exercise05_missingFC;
        if rcDOC NE 0 then output exercise05_missingDOC;
end;
     * we are doing nothing else, so stop;
     STOP;
RUN ;
```

Basic Usage of the HASH Object

Multiple HASH Objects.

- `DECLARE hash feecodes(DATASET:'data.feecodes');`
- `DECLARE hash doctors(DATASET:'data.doctors');`
  - **Separate DECLARE for each hash object**

- `rc=feecodes.defineKey('feecode');`
- `rc=doctors.defineKey('did');`
  - **Same method .defineKey() for each hash object**

- `rcFC = feecodes.find();`
- `rcDOC = doctors.find();`
  - **.find() will use the appropriate key**

Basic Usage of the HASH Object

SAS/SQL Lookups Revisited

```
proc sql;
create table SQL_lookup as
select visitDate,
       f.section,  f.subsection, t.feecode, f.feeAmount,
       t.pid, p.sex as patientSex, p.dob as patientDOB,
       t.did, d.sex as doctorSex, d.dob as doctorDOB
from data.transactions as t
     left join data.patients as p on t.pid = p.pid
     left join data.doctors  as d on t.did = d.did
     left join data.feecodes as f on t.feecode = f.feecode
order by visitdate, t.pid, t.feecode, t.did
;
Quit;
```

Basic Usage of the HASH Object

Multiple HASH Objects – using dataset options.

```
DATA exercise06(drop=rcFC rcDOC) exercise06_missingFC(keep=feecode)
    exercise06_missingDOC(keep=did) exercise06_missingPAT(keep=pid)
   ;
   if _n_ = 0
     then
       do;
         set data.feecodes;
         set data.doctors  (keep=did dob rename=(dob=doctorDOB));
         set data.patients (keep=pid dob rename=(dob=patientDOB));
       end;
```

Basic Usage of the HASH Object

```
DECLARE hash  feecodes(DATASET:'data.feecodes') ;
      rc=feecodes.defineKey('feecode');
      rc=feecodes.defineData('section','subSection','feeAmount');
      rc=feecodes.defineDone();
DECLARE hash  doctors(DATASET:'data.doctors (keep=did dob
                              rename=(dob=doctorDOB))') ;
      rc=doctors.defineKey('did');
      rc=doctors.defineData('doctorDOB');
      rc=doctors.defineDone();
DECLARE hash  patients(DATASET:'data.patients (keep=pid dob
                              rename=(dob=patientDOB))') ;
      rc=patients.defineKey('pid');
      rc=patients.defineData('patientDOB');
      rc=patients.defineDone();
```

Basic Usage of the HASH Object

```
* now we need to read each record in the transactions table;
   do while (not done);
        set data.transactions end=done;
        call missing(section, subSection, feeAmount,
                     doctorDOB, patientDOB);
        rcFC = feecodes.find();
        rcDOC = doctors.find();
        rcPAT = patients.find();

        output exercise06;
```

Basic Usage of the HASH Object

```
        if rcFC  NE 0 then output exercise06_missingFC;
        if rcDOC NE 0 then output exercise06_missingDOC;
        if rcPAT NE 0 then output exercise06_missingPAT;
end;
     * we are doing nothing else, so stop;
     STOP;
RUN ;
```

Basic Usage of the HASH Object

Multiple HASH Objects

- Multiple sets of
  - ***DECLARE***,
  - ***.defineKey()***,
  - ***.defineData()***
  - ***.defineDone()***
- ***.find()*** for each hash object is invoked

Basic Usage of the HASH Object

Multiple HASH Objects

- Starting in 9.2 dataset options can be used with the DATASET: option
  - ```
    DECLARE hash  doctors(DATASET:'data.doctors
                          (keep=did dob
                          rename=(dob=doctorDOB))') ;
    ```
    ```

Basic Usage of the HASH Object

What if you want to save items in the HASH object?

- Transient
- Run time

- ***.output()*** method
  - Save the data items ***(.defineData())*** of the hash object to a dataset

Basic Usage of the HASH Object

What if you want to save items in the HASH object?

```
DATA _null_;
   if _n_ = 0 then set data.feecodes;
   DECLARE hash feecodes(DATASET:'data.feecodes', ORDERED:'D');
        rc=feecodes.defineKey('feecode');
        rc=feecodes.defineData(ALL:'yes');
        rc=feecodes.defineDone();
        rc = feecodes.output(dataset:"feecodesHASH");
   STOP;
RUN ;
```

Basic Usage of the HASH Object

What if you want to save items in the HASH object?

- ***rc = feecodes.output(dataset:"feecodesHASH");***
  - feecodes is the HASH object
  - .output () is the method
  - Dataset:"feecodesHASH" specifies the name of the dataset.
    Note: in the example we were using a DATA _NULL_;
    - The dataset name can be data driven

Agenda

- ☑ *Introduction*
- ☑ Table *Lookups in SAS*
- ☑ *Declaring your HASH Object*
- ☑ *Basic Usage of the HASH Object*
- Review

The SAS® Hash Object: It's Time To .find() Your Way Around

For Sample Code and Data email

hash@fernwood.ca

Peter Eberhardt
Fernwood Consulting Group Inc.

peter@fernwood.ca

Basic Usage of the HASH Object

What is we want to load duplicates into the hash object?

- SAS 9.1
  - Had to create a 'dummy' secondary key
- SAS 9.2
  - MULTIDATA option

```
DECLARE hash  feecodes(DATASET:'feecodes',
                       ORDERED:'A',
                       MULTIDATA:"Y");
```

Basic Usage of the HASH Object

What is we locate duplicates into the hash object?

- ***.find()***
  - Will find only one occurrence
- ***.has_next()/.find_next()***
  - ***.has_next()*** to determine is there is another value
  - ***.find_next()*** to retrieve the next duplicate

Basic Usage of the HASH Object

How we locate duplicates into the hash object?

```
feecode = 'A007';
rc = feecodes.find();
anotherCode = .;
rc = feecodes.has_next(RESULT: anotherCode);
do while (anotherCode NE 0);
    rc = feecodes.find_next();
    rc = feecodes.has_next(RESULT: anotherCode);
end;
```

Basic Usage of the HASH Object

EXERCISE 8

- Open EXERCISE08.SAS
  - **Run the initial datastep to create a duplicate record**
  - **DECLARE hash feecodes(DATASET:'feecodes',**
    ORDERED:'A', MULTIDATA:"Y");
  - **feecodes.defineKey('feecode')**
  - **feecodes.defineData('feecode', 'feeAmount')**
  - **feecodes.add()**
  - **feecodes.output(DATASET:"feecodesHASHDuplicates");**

Basic Usage of the HASH Object

What if you want to remove items from the hash object?

- ***rc = feecodes.remove();***
  - Removes the current item from the hash object

Basic Usage of the HASH Object

EXERCISE 9

- Open EXERCISE09.SAS
  - **feecodes.defineKey('feecode')**
  - **feecodes.defineData('feecode', 'feeAmount')**
  - **feecodes.defineDone()**
  - **feecodes.check()**
  - **feecodes.remove()**
  - **feecodes.output(DATASET:"feecodesHASHRemove");**

  - **Watch for errors being generated!!**

Basic Usage of the HASH Object

What if you want to 'step through' items in the HASH object?

- HASH Iterator object

Basic Usage of the HASH Object

What if you want to 'step through' items in the HASH object?

```
DATA _null_;
   if _n_ = 0 then set data.feecodes (keep=feecode feeAmount);
   DECLARE hash  feecodes(DATASET:"data.feecodes
                              (where=(feecode like 'H1%'))
                                keep= feecode feeAmount");
        rc=feecodes.defineKey('feecode');
        rc=feecodes.defineData( 'feecode', 'feeAmount');
   DECLARE hiter hi_feecodes('feecodes');
        rc=feecodes.defineDone();
```

Basic Usage of the HASH Object

What if you want to 'step through' items in the HASH object?

```
* start the the beginning;
   put / "----- Traversing using the iterator first time";
   rc = hi_feecodes.first();
   do while (rc = 0) ;
        put feecode= feeAmount= ;
        rc = hi_feecodes.next() ;
   end ;
```

Basic Usage of the HASH Object

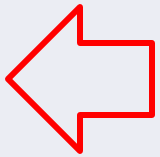
What if you want to 'step through' items in the HASH object?

```
* start the the end;
   put / "----- Traversing using the iterator second time";
   rc = hi_feecodes.last();
   do while (rc = 0) ;
        put feecode= feeAmount= ;
        rc = hi_feecodes.prev() ;
   end ;
     STOP;
RUN ;
```

Basic Usage of the HASH Object

What if you want to 'step through' items in the HASH object?

```
----- Traversing using the iterator first time
row=1 feecode=H131 feeAmount=18.70
row=2 feecode=H132 feeAmount=46.30
row=3 feecode=H133 feeAmount=40.10
...
row=9 feecode=H101 feeAmount=15.00
row=10 feecode=H102 feeAmount=37.20
row=11 feecode=H103 feeAmount=32.25
```



NOTE: the feecodes are not sorted

Basic Usage of the HASH Object

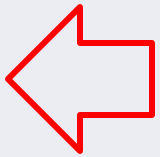
What if you want to 'step through' items in the HASH object?

```
DATA _null_;
   if _n_ = 0 then set data.feecodes (keep=feecode feeAmount);
   DECLARE hash  feecodes(DATASET:"data.feecodes
                              (where=(feecode like 'H1%'))
                                keep= feecode feeAmount",
                              ORDERED:'A' );
        rc=feecodes.defineKey('feecode');
        rc=feecodes.defineData( 'feecode', 'feeAmount');
   DECLARE hiter hi_feecodes('feecodes');
        rc=feecodes.defineDone();
```

Basic Usage of the HASH Object

What if you want to 'step through' items in the HASH object?

----- Traversing using the iterator first time
row=1 feecode=H101 feeAmount=15.00
row=2 feecode=H102 feeAmount=37.20
row=3 feecode=H103 feeAmount=32.25
…
row=9 feecode=H122 feeAmount=73.90
row=10 feecode=H123 feeAmount=62.30
row=11 feecode=H124 feeAmount=29.80



NOTE: the feecodes are sorted

Basic Usage of the HASH Object

What if you want to 'step through' items in the HASH object?

- DECLARE hiter hi_feecodes('feecodes');
  - DECLARE is telling SAS you want an object
  - hiter is the type of object you want
  - hi_feecodes is the name HITER object
  - 'feecodes' is the string with the name of the HASH object

Basic Usage of the HASH Object

What if you want to 'step through' items in the HASH object?

- Before an Iterator object can be created, the HASH object upon which it is based must be DECLARED and instantiated.
- When using an iterator you probably want to order the HASH object with the ORDERED: option
  - DECLARE hash feecodes(DATASET:"data.feecodes "
    ORDERED:'A');

Peter Eberhardt, Fernwood Consulting Group Inc.

Basic Usage of the HASH Object

What if you want to 'step through' items in the HASH object?

- How does ***.find()/.check()*** work with the iterator?

```
rc = feecodes.find(key:'H102');
do while (rc = 0) ;
    put feecode= feeAmount= ;
    rc = hi_feecodes.next() ;
end ;
```


Peter Eberhardt, Fernwood Consulting Group Inc.

Basic Usage of the HASH Object

What if you want to 'step through' items in the HASH object?

- How does ***.find()/.check()*** work with the iterator?
- Iterator ***.setCur()*** method

```
rc = feecodes.find(key:'H102');
rc = hi_feecodes.setcur();
do while (rc = 0) ;
    put feecode= feeAmount= ;
    rc = hi_feecodes.next() ;
end;
```

Basic Usage of the HASH Object

EXERCISE 10

- Open EXERCISE09.SAS
  - **DECLARE hiter hi_feecodes('feecodes');**
  - **feecodes.defineDone()**
  - **rc = hi_feecodes.first()**
  - **rc = hi_feecodes.next();**
  - **rc = hi_feecodes.last()**
  - **rc = hi_feecodes.prev();**
  - **rc = hi_feecodes.setcur()**

Basic Usage of the HASH Object

What if you want to save items in the HASH object – Part 2?

- ***.output(DATASET:'datasetname');***

Basic Usage of the HASH Object

What if you want to save multiple datasets?

- ***.output(DATASET:'datasetname')***
  - ***'datasetname' is a string***
    - ***Can be set at run time***

Basic Usage of the HASH Object

What if you want to save items in the HASH object?

- One dataset for each section

```
do until(done);
      do rec = 1 by 1 until ( last.section );
         set data.feecodes end=done;
         by section;
         feecodes.add() ;
      end ;
      feecodes.output (dataset: 'section' || section) ;
      feecodes.clear();
   end;
run ;
```

Basic Usage of the HASH Object

What if you want to save items in the HASH object?

```
NOTE: The data set WORK.SECTIONA has 711 observations and 4
   variables.

NOTE: The data set WORK.SECTIONB has 164 observations and 4
   variables.

NOTE: The data set WORK.SECTIONC has 21 observations and 4
   variables.

NOTE: The data set WORK.SECTIOND has 338 observations and 4
   variables.

NOTE: The data set WORK.SECTIONE has 65 observations and 4
   variables.

NOTE: The data set WORK.SECTIONF has 25 observations and 4
   variables.
```

Basic Usage of the HASH Object

EXERCISE 11

- Open EXERCISE11.SAS
  - **feecodes.definekey ('rec')**
  - **feecodes.defineData('section', 'subsection', 'feecode', 'feeAmount')**
    - **NOTE: could have used feecodes.defineData(ALL:"Y")**
  - **rc = feecodes.defineDone()**
  - **rc = feecodes.add()**
  - **rc = feecodes.output(dataset: 'section' || section)**
  - **rc = feecodes.clear();**

Agenda

- ☑ *Introduction*
- ☑ Table *Lookups in SAS*
- ☑ *Declaring your HASH Object*
- ☑ *Basic Usage of the HASH Object*
- Review

For Sample Code and Data email

mwsug2010@fernwood.ca

The SAS® Hash Object: It's Time To .find() Your Way Around

Peter Eberhardt
Fernwood Consulting Group Inc.

peter@fernwood.ca

??Questions??

The SAS® Hash Object:
It's Time To .find() Your Way Around

peter@fernwood.ca

Peter Eberhardt
Fernwood Consulting Group Inc.