Hash Object – Methods

```plaintext
declare hash obj();
declare hash obj(dataset: 'dataset_name',
   duplicate: 'replace' | 'error', hashexp: n,
   ordered: 'a' | 'd' | 'no', suminc: 'count_var');

Creates a hash object with the properties:
dataset: loads the hash object from a data set.
duplicate: controls how duplicate keys are handled when loading from a data set.
hashexp: n declares 2^n slots for the hash object.
ordered: specifies a key sort order when using a hash iterator or the output method.
suminc: count_var contains the increment value for a key summary that is retrieved by the sum method.*

rc = obj.defineKey('key_val1', ... , 'key_valN');
rc = obj.defineKey(all: 'yes');
Defines a set of hash object keys given by key_val1...key_valN.

rc = obj.defineData('data_var1', ..., 'data_varN');
r = obj.defineData(all: 'yes');
Defines data, given by data_var1...data_varN, to be stored in the hash object.

rc = obj.defineDone();
Indicates that key and data definitions are complete.

rc = obj.add();
r = obj.add(key: 'key_val1', ..., key: 'key_valN',
   data: 'data_val1', ..., data: 'data_valN');
Adds the specified data associated with the given key to the hash object.

rc = obj.find();
r = obj.find(key: 'key_val1', ..., key: 'key_valN');
Determines whether the given key has been stored in the hash object. If it has, the data variables are updated and the return code is set to zero. If the key is not found, the return code is non-zero.

rc = obj.remove();
r = obj.remove(key: 'key_val1', ..., key: 'key_valN');
Removes all entries from a hash object without deleting the hash object.

rc = obj.clear();
Removes all entries from a hash object without deleting the hash object.

rc = obj.equals(hash: 'hash_obj', result: res_var);
Determines if two hash objects are equal. If they are equal, res_var is set to 1, otherwise it is set to zero.

rc = obj.replace();
r = obj.replace(key: 'key_val1', ..., key: 'key_valN',
   data: 'data_val1', ..., data: 'data_valN');
Replaces the data associated with the given key with new data as specified in data_val1...data_valN.

rc = obj.check();
r = obj.check(key: 'key_val1', ..., key: 'key_valN');
Checks whether the given key has been stored in the hash object. The data variables are not updated. Return codes are the same as for find.

rc = obj.output(dataset: 'dataset_name');
Creates dataset dataset_name which will contain the data in the hash object.

rc = obj.sum(sum: sum_var);
r = obj.sum(key: 'key_val1', ..., key: 'key_valN',
   sum: sum_var);
Gets the key summary for the given key and stores it in the DATA Step variable sum_var. Key summaries are incremented when a key is accessed.*

rc = obj.ref();
r = obj.ref(key: 'key_val1', ..., key: 'key_valN');
Performs a find operation for the current key. If the key is not in the hash object, it will be added.*

rc = obj.replace();
r = obj.replace(key: 'key_val1', ..., key: 'key_valN',
   data: 'data_val1', ..., data: 'data_valN');
Replaces the data associated with the given key with new data as specified in data_val1...data_valN.

rc = obj.check();
r = obj.check(key: 'key_val1', ..., key: 'key_valN');
Checks whether the given key has been stored in the hash object. The data variables are not updated. Return codes are the same as for find.

rc = obj.remove();
r = obj.remove(key: 'key_val1', ..., key: 'key_valN');
Removes the data associated with the given key.

rc = obj.clear();
Removes all entries from a hash object without deleting the hash object.

rc = obj.equals(hash: 'hash_obj', result: res_var);
Determines if two hash objects are equal. If they are equal, res_var is set to 1, otherwise it is set to zero.

All methods return zero for success
```

Feature available in SAS 9.2 and later.

Hash Object – Attributes

```plaintext
i = obj.num_items;
Retrieves the number of elements in the hash object.
sz = obj.item_size;
Obtains the item size, in bytes, for an item in the hash object.*
```

Hash Iterator – Methods

```plaintext
declare hiter iterobj('hash_obj');

Creates a hash iterator to retrieve items from the hash object named hash_obj.

rc = iterobj.first();
Copies the data for the first item in the hash object into the data variables for the hash object.

rc = iterobj.last();
Copies the data for the last item in the hash object into the data variables for the hash object.

rc = iterobj.next();
Copies the data for the next item in the hash object into the data variables for the hash object. A non-zero value is returned if the next item cannot be retrieved.

Use iteratively to traverse the hash object and return the data items in key order. If first has not been called, next begins with the first item.

rc = iterobj.prev();
Copies the data for the previous item in the hash object into the data variables for the hash object. A non-zero value is returned if the next item cannot be retrieved.

Use iteratively to traverse the hash object and return the data items in reverse key order. If last has not been called, prev begins with the last item.
```
Example – Load and Find

/* Create Input Data Set */
data names;
  length first last title $ 16 born died 8;
  input first last born died title & $16.;
  datalines;
  William Blake 1757 1827 Spring
  John Keats 1795 1821 To Autumn
  Mary Shelley 1797 1851 Frankenstein;
/* Load and Find */
data _null_;   length first last title $ 16;
  length born died 8;
  declare hash ht(dataset:"names");
  ht.defineKey("first", "last");
  ht.defineData("born", "died", "title");
  ht.defineDone();
/* Find John Keats */
  first = "John";
  last = "Keats";
  rc = ht.find();
  if rc = 0 then
    put "Found " first last title $QUOTE.;;
  else
    put "Not Found " first last;
run;
Output:
  Found John Keats "To Autumn"

Example – Sorted Output

/* Add to hash and then output */
data _null_;   length patient_id $ 16 discharge 8;
  if _N_ = 1 then do;
    declare hash ht(ordered:"a");
    ht.defineKey("patient_id");
    ht.defineData("patient_id", "discharge");
    ht.defineDone();
  end;
infile datalines eof=output;
  input patient_id discharge:DATE9.;
  ht.add();
  output:
    ht.output(dataset:"sorted_ids");
  datalines;
  Smith-4123 15MAR2004
  Hagen-2834 23APR2004
  Smith-2437 15JAN2004
  Flinn-2940 12FEB2004;
run;
Output:
  Flinn-2940 12FEB2004
  Hagen-2834 23APR2004
  Smith-2437 15JAN2004
  Smith-4123 15MAR2004

Example – Hash Iterator

/* Create Input Data Set */
data patients;
  length patient_id $ 16 discharge 8;
  input patient_id discharge:DATE9.;
  datalines;
  Smith-4123 15MAR2004
  Hagen-2834 23APR2004
  Smith-2437 15JAN2004
  Flinn-2940 12FEB2004;
/* Load and iterate over hash */
data _null_;   length patient_id $ 16 discharge 8;
  declare hash ht(dataset:"patients", ordered:"ascending");
  ht.defineKey("patient_id");
  ht.defineData("patient_id", "discharge");
  ht.defineDone();
  declare hiter iter("ht");
  rc = iter.first();
do while (rc=0);
    put patient_id discharge:DATE9.;
    rc = iter.next();
end;
run;
Output:
  Flinn-2940 12FEB2004
  Hagen-2834 23APR2004
  Smith-2437 15JAN2004
  Smith-4123 15MAR2004

For complete information refer to the Base SAS documentation at http://support.sas.com/base