```
%global dsn;
%let dsn=PriceData;
%global metric;
%let metric=price;
%global prim;
%let prim=brand;
%global sec;
%let sec=store;
%global nid;
%let nid=date;
%global copy;
%let copy=selected category state;
%global ndsn;
%let ndsn=Price;

%global PriceDataName;
%let PriceDataName=TOBACCO_MART_AVG_PRICE;
data &dsn;
set &PriceDataName;
run;
proc print data=&dsn;
run;



%macro stackData(dsn, metric, prim, sec, nid, copy, ndsn);

/*Create two lookup datasets: prim_lookup with primary varaible values
and record numbers as variable prim_id and sec_lookup with secondary
variable values and record numbers as variable sec_id. The prim_id and
sec_id variable values will then be incorporated into the main dataset
because when the dataset is transformed, these values will be SAS
compliant variable names.*/

/*Sort the data by primary variable.*/
proc sort data=&dsn;
by &prim;
run;
/*Create a lookup dataset for primary using the record number as the
id.*/
data prim_lookup;
 set &dsn;
 run;
proc sort data=prim_lookup(keep=&prim) nodupkey;
 by &prim;
 run;
data prim_lookup;
 set prim_lookup;
 length &prim._id 5 &prim._code $8;
 &prim._id = _n_;
 &prim._code = cats('var_',_n_);

/*Sort the data by secondary variable.*/
proc sort data=&dsn;
by &sec;
run;
/*Create a lookup dataset for secondary using the record number as the
id.*/
data sec_lookup;
 set &dsn;
 run;
proc sort data=sec_lookup(keep=&sec) nodupkey;
 by &sec;
 run;
data sec_lookup;
 set sec_lookup;
 length &sec._id 5 &sec._code $5;
 &sec._id = _n_;
 &sec._code = cats('_',_n_);
 run;

 /*Incorporate the newly created ids into the working dataset.*/
data &dsn;
set &dsn;
 length &prim._id 5 &prim._code $8 &sec._id 5 &sec._code $5;
 &prim._id = .;
 &prim._code = '';
 &sec._id = .;
 &sec._code = '';
proc sql;
 update &dsn as d set &prim._id=(select &prim._id from prim_lookup as
p where d.&prim = p.&prim), &prim._code=(select &prim._code from
prim_lookup as p where d.&prim = p.&prim) , &sec._id=(select &sec._id
from sec_lookup as s where d.&sec = s.&sec), &sec._code=(select
&sec._code from sec_lookup as s where d.&sec = s.&sec);
 quit;


/*Create a lookup table to store the variables chosen to be copied in
the &copy macro variable, the primary and secondary variables,
 and the prim_id and sec_id variables so no information is lost. This
information will later be incorporated back into the stacked
dataset.*/
data lookup;
 set &dsn(keep=&prim &sec &prim._id &sec._id &copy);
Proc sort data=lookup (keep=&prim &sec &prim._id &sec._id &copy)
nodupkey;
 by &prim &sec;
 run;
```

```
/*Transpose the data naming the _name_ var as the new key &nid.*/
proc transpose data=&dsn(drop=&prim &sec &prim._id &sec._id &copy)
out=&dsn._t(rename=(_name_=&nid));
id &prim._code &sec._code;
run;

/*Macro getvars obtains a list of variables from a dataset.*/
%macro getvars(dsn);
%global vlist;
%global nvar;
%let nvar=%sysfunc(attrn(%sysfunc(open(work.&dsn,i)),nvars));
proc sql;
select name into :vlist separated by ' '
from dictionary.columns
where memname=upcase("&dsn");
quit;
%mend;
%getvars(&dsn._t);
%put &vlist;
%put &nvar;

/*Create a new list from vlist to hold the values of the temporary
datasets that will be named tmp_&prim._&sec.*/
%macro createNewList(str, len);
%let n=2;
%let delim=' ';
%let modif=mo;
%global nlist;
%let nlist='';
%do %while (&n <= &len);
        %let var = %SYSFUNC(cats(tmp_,%scan(&str,&n,&delim,&modif)));
        %let nlist = %SYSFUNC(catx(%str( ),&nlist,&var));
        /*The following blocked out code was supposed to eliminate the
initialization of nlist with single quotes
        and the need to compress nlist to eliminate the initial single
quotes later. It does not work though.*/
        /*
        %if &n = 2 %then %let nlist = &var
        %else %let nlist = %SYSFUNC(catx(%str( ),&nlist,&var));
        */
%let n=%eval(&n+1);
%end;
%mend;
%createNewList(&vlist,&nvar);
%let nlist=%SYSFUNC(COMPRESS(&nlist.,%str(%')));
%put &nlist;

%macro stackDatasets(str,len);
/*start at 2 bc first var is &nid and must be present with each of the
other variables.*/
```

```
%let n=2;
%let delim=' ';
%let modif=mo;
%do %while (&n <= &len);
        %let var = %scan(&str,&n,&delim,&modif);
        %let dset = %sysfunc(cats(tmp_,&var));
                data &dset;
        set &dsn._t(keep=&nid &var rename=(&var=&metric));
        delim='_';
        modif='o';
        length &prim._id 5 &sec._id 5;
        &prim._id = scan("&var",2,delim,modif);
        &sec._id = scan("&var",3,delim,modif);
        run;

        %let n=%eval(&n+1);
%end;
%mend;
%stackDatasets(&vlist,&nvar);

data stacked;
set &nlist;

proc sql;
  create table &ndsn as
  select *
  from stacked, lookup
  where stacked.&prim._id=lookup.&prim._id AND
stacked.&sec._id=lookup.&sec._id;
quit;

data &ndsn;
set &ndsn(drop=&prim._id &sec._id  delim modif);
run;

/*END OF STACKDATA MACRO*/
%mend;

/*Call macro stackData and send dataset to be stacked, the dataset
metric, primary transpose variable, secondary transpose variable,
name of new id variable for after transpose, list of variables to be
copied to stacked dataset (variables in this list will be lost
in transpose so must list them to keep them) ,and a name for the final
stacked dataset.*/
%stackData(&dsn, &metric, &prim, &sec, &nid, &copy, &ndsn);

proc print data=&ndsn;
run;
```