SAS® Programming Tips, Tricks, and Techniques

A presentation by Kirk Paul Lafler
Presentation Objectives - Explore

- Useful SAS System Options
- Interesting PROC SQL Options
- A User-defined Function using PROC FCMP
- Integrity Constraints for Tables
- A User-defined Dictionary Table and SASHELP View Tool
## Example Datasets / Tables

### Movies

<table>
<thead>
<tr>
<th>Title</th>
<th>Length</th>
<th>Category</th>
<th>Year</th>
<th>Studio</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brave Heart</td>
<td>177</td>
<td>Action Adventure</td>
<td>1995</td>
<td>Paramount Pictures</td>
<td>R</td>
</tr>
<tr>
<td>Casablanca</td>
<td>103</td>
<td>Drama</td>
<td>1942</td>
<td>MGM / UA</td>
<td>PG</td>
</tr>
<tr>
<td>Christmas Vacation</td>
<td>97</td>
<td>Comedy</td>
<td>1988</td>
<td>Warner Brothers</td>
<td>PG.13</td>
</tr>
<tr>
<td>Coming to America</td>
<td>116</td>
<td>Comedy</td>
<td>1988</td>
<td>Paramount Pictures</td>
<td>R</td>
</tr>
<tr>
<td>Dracula</td>
<td>130</td>
<td>Horror</td>
<td>1993</td>
<td>Columbia TriStar</td>
<td>R</td>
</tr>
<tr>
<td>Dressed to Kill</td>
<td>105</td>
<td>Drama Mysteries</td>
<td>1960</td>
<td>Filmways Pictures</td>
<td>R</td>
</tr>
<tr>
<td>Forrest Gump</td>
<td>142</td>
<td>Drama</td>
<td>1994</td>
<td>Paramount Pictures</td>
<td>PG.13</td>
</tr>
<tr>
<td>Ghost</td>
<td>127</td>
<td>Drama Romance</td>
<td>1990</td>
<td>Paramount Pictures</td>
<td>PG.13</td>
</tr>
<tr>
<td>Jaws</td>
<td>125</td>
<td>Action Adventure</td>
<td>1975</td>
<td>Universal Studios</td>
<td>PG</td>
</tr>
<tr>
<td>Jurassic Park</td>
<td>127</td>
<td>Action</td>
<td>1993</td>
<td>Universal Pictures</td>
<td>PG.13</td>
</tr>
<tr>
<td>Lethal Weapon</td>
<td>110</td>
<td>Action Cops &amp; Robber</td>
<td>1987</td>
<td>Warner Brothers</td>
<td>R</td>
</tr>
<tr>
<td>Michael</td>
<td>106</td>
<td>Drama</td>
<td>1997</td>
<td>Warner Brothers</td>
<td>PG.13</td>
</tr>
<tr>
<td>National Lampoon’s Vacation</td>
<td>98</td>
<td>Comedy</td>
<td>1993</td>
<td>Warner Brothers</td>
<td>PG.13</td>
</tr>
<tr>
<td>Poltergeist</td>
<td>115</td>
<td>Horror</td>
<td>1982</td>
<td>MGM / UA</td>
<td>PG</td>
</tr>
<tr>
<td>Rocky</td>
<td>120</td>
<td>Action Adventure</td>
<td>1976</td>
<td>MGM / UA</td>
<td>PG</td>
</tr>
<tr>
<td>Scarface</td>
<td>170</td>
<td>Action Cops &amp; Robber</td>
<td>1983</td>
<td>Universal Studios</td>
<td>R</td>
</tr>
<tr>
<td>Silence of the Lambs</td>
<td>118</td>
<td>Drama Suspense</td>
<td>1991</td>
<td>Orion</td>
<td>R</td>
</tr>
<tr>
<td>Star Wars</td>
<td>124</td>
<td>Action Sci-Fi</td>
<td>1977</td>
<td>Lucas Films Ltd</td>
<td>PG</td>
</tr>
<tr>
<td>The Hunt for Red October</td>
<td>135</td>
<td>Action Adventure</td>
<td>1989</td>
<td>Paramount Pictures</td>
<td>PG</td>
</tr>
<tr>
<td>The Wizard of Oz</td>
<td>101</td>
<td>Adventure</td>
<td>1939</td>
<td>MGM / UA</td>
<td>G</td>
</tr>
<tr>
<td>Titanic</td>
<td>134</td>
<td>Drama Romance</td>
<td>1997</td>
<td>Paramount Pictures</td>
<td>PG.13</td>
</tr>
</tbody>
</table>

### Actors

<table>
<thead>
<tr>
<th>Title</th>
<th>Actor_Leading</th>
<th>Actor_Supporting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brave Heart</td>
<td>Mel Gibson</td>
<td>Sophie Marceau</td>
</tr>
<tr>
<td>Christmas Vacation</td>
<td>Chevy Chase</td>
<td>Beverly D’Angelo</td>
</tr>
<tr>
<td>Coming to America</td>
<td>Eddie Murphy</td>
<td>Arsenio Hall</td>
</tr>
<tr>
<td>Forrest Gump</td>
<td>Tom Hanks</td>
<td>Sally Field</td>
</tr>
<tr>
<td>Ghost</td>
<td>Patrick Swayze</td>
<td>Demi Moore</td>
</tr>
<tr>
<td>Lethal Weapon</td>
<td>Mel Gibson</td>
<td>Danny Glover</td>
</tr>
<tr>
<td>Michael</td>
<td>John Travolta</td>
<td>Andre MacDowell</td>
</tr>
<tr>
<td>National Lampoon’s Vacation</td>
<td>Chevy Chase</td>
<td>Beverly D’Angelo</td>
</tr>
<tr>
<td>Rocky</td>
<td>Sylvester Stallone</td>
<td>Taia Shire</td>
</tr>
<tr>
<td>Silence of the Lambs</td>
<td>Anthony Hopkins</td>
<td>Jodie Foster</td>
</tr>
<tr>
<td>The Hunt for Red October</td>
<td>Sean Connery</td>
<td>Alec Baldwin</td>
</tr>
<tr>
<td>The Terminator</td>
<td>Arnold Schwarzenegge</td>
<td>Michael Eisner</td>
</tr>
<tr>
<td>Titanic</td>
<td>Leonardo DiCaprio</td>
<td>Kate Winslet</td>
</tr>
</tbody>
</table>
Exploring
SAS® System
Options
Primary Source Statements:

OPTIONS SOURCE2;
%INCLUDE ‘c:\Workshops\LogControlOptions.sas’;
PROC PRINT DATA=MOVIES NOOBS;
   VAR TITLE RATING CATEGORY;
RUN;

Secondary Source Statements (Included Code):

OPTIONS MSGLEVEL=I ;
/* DISPLAY SORT, MERGE PROCESSING, AND INDEX USAGE */;
SAS Log Results:

OPTIONS SOURCE2;
%INCLUDE 'c:\Workshops\LogControlOptions.sas';

NOTE: %INCLUDE (level 1) file
c:\Workshops\LogControlOptions.sas is file
+OPTIONS MSGLEVEL=I
+/* DISPLAY SORT, MERGE PROCESSING, AND INDEX USAGE */
+
NOTE: %INCLUDE (level 1) ending.

PROC PRINT DATA=MOVIES NOOBS;
  VAR TITLE RATING CATEGORY;
RUN;
Exploring PROC SQL Options
SQL Join Algorithms

- Nested Loop (aka “Brute” force) join algorithm
- Sort-merge join algorithm
- Index join algorithm
- Hash join algorithm
Influencing the SQL Optimizer

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAGIC=101</td>
<td>Influences the SQL optimizer to select the Nested Loop join algorithm.</td>
</tr>
<tr>
<td>MAGIC=102</td>
<td>Influences the SQL optimizer to select the Sort-Merge join algorithm.</td>
</tr>
<tr>
<td>MAGIC=103</td>
<td>Influences the SQL optimizer to select the Hash join algorithm.</td>
</tr>
</tbody>
</table>
Specifying MAGIC=101

PROC SQL MAGIC=101;
SELECT *
FROM MOVIES, ACTORS
WHERE MOVIES.TITLE = ACTORS.TITLE;
QUIT;

SAS Log Results
PROC SQL MAGIC=101;
SELECT *
FROM MOVIES, ACTORS
WHERE MOVIES.TITLE = ACTORS.TITLE;

NOTE: PROC SQL planner chooses sequential loop join.
QUIT;
Specifying MAGIC=102

PROC SQL MAGIC=102;

SELECT *
FROM MOVIES, ACTORS
WHERE MOVIES.TITLE = ACTORS.TITLE;
QUIT;

SAS Log Results
PROC SQL MAGIC=102;
SELECT *
FROM MOVIES, ACTORS
WHERE MOVIES.TITLE = ACTORS.TITLE;

NOTE: PROC SQL planner chooses merge join.
QUIT;
Specifying MAGIC=103

PROC SQL MAGIC=103;
SELECT *
FROM MOVIES, ACTORS
WHERE MOVIES.TITLE = ACTORS.TITLE;
QUIT;

SAS Log Results
PROC SQL MAGIC=103;
SELECT *
FROM MOVIES, ACTORS
WHERE MOVIES.TITLE = ACTORS.TITLE;

NOTE: PROC SQL planner chooses merge join.
NOTE: A merge join has been transformed to a hash join.
QUIT;
Specifying IDXWHERE=Yes

The IDXWHERE= dataset option can be specified to influence the SQL optimizer to use the most efficient index available (if one exists) to execute a query.
Specifying IDXWHERE=Yes

OPTIONS MSGLEVEL=I;

PROC SQL;
SELECT *
FROM MOVIES(IDXWHERE=Yes), ACTORS
WHERE MOVIES.TITLE = ACTORS.TITLE;
QUIT;

SAS Log Results
PROC SQL;
SELECT *
FROM MOVIES(IDXWHERE=Yes), ACTORS
WHERE MOVIES.TITLE = ACTORS.TITLE;
INFO: Index Rating selected for WHERE clause optimization.
QUIT;
Exploring a User-defined Function using PROC FCMP
PROC FCMP Advantages

User-Defined functions created with the FCMP procedure provide specific advantages:

✓ Code can be easier to read, write and modify
✓ Code is modular and callable
✓ Code is independent and not affected by its implementation
✓ Code is reusable by any program that has access to the dataset where the function is stored
/* Constructing a User Defined Function in FCMP */
proc fcmp outlib=sasuser.myfunctions.examples;
  function Age_of_Movie_function(year);
    if year NE . then do;
      Age_of_Movie = year(today()) – year;
    end;
    return(Age_of_Movie);
  endsub;
quit;
options cmplib=sasuser.myfunctions;

/* Call the User Defined Function created in FCMP */
data Age_of_Movie;
  set mydata.movies;
  Age_of_Movie = Age_of_Movie_function(year);
  put Year= Age_of_Movie=;
run;
User-defined Function (Log Results)

/* Constructing a User Defines Function (UDF) in FCMP */
proc fcmp outlib=sasuser.myfunctions.examples;
  function Age_of_Movie_function(year);
    if year NE . then do;
      Age_of_Movie = year(today()) - year;
    end;
    return(Age_of_Movie);
  endsub;
quit;

NOTE: Function Age_of_Movie_function saved to sasuser.myfunctions.examples.
NOTE: PROCEDURE FCMP used (Total process time):  
  real time 0.15 seconds
  cpu time 0.04 seconds

options cmplib=sasuser.myfunctions;
/* Call the user defined Function (UDF) created in FCMP */
data Age_of_Movie;
  set mydata.movies;
  Age_of_Movie = Age_of_Movie_function(year);
  put Year= Age_of_Movie=;
run;

NOTE: The data set WORK.AGE_OF_MOVIE has 22 observations and 7 variables.
NOTE: DATA statement used (Total process time):  
  real time 0.07 seconds
  cpu time 0.06 seconds
Exploring Integrity Constraints for Tables
Methods of Building Integrity

Data integrity problems such as missing information, duplicate values, and invalid data values can affect user confidence in a database environment. The objective is to establish rules in the database table(s) to safeguard and protect data.

- Application programs – Older / less reliable
- Database table environment – Newer / more reliable
  - PROC DATASETS
  - PROC SQL
Data integrity is maintained by assigning column and table constraints. Modifications made through update and delete operations can have referential integrity constraints built into the database environment.

Column and Table Constraints

- **NOT NULL**
- **UNIQUE**
- **CHECK**
To prevent null values from appearing in any row of a table for a specified column, a NOT NULL constraint can be coded.

```
PROC SQL;
CREATE TABLE work.RENTAL_INFO
  (TITLE CHAR(30) NOT NULL,
   RENTAL_AMT NUM FORMAT=DOLLAR6.2);
QUIT;
```
The UNIQUE constraint prevents rows containing duplicate values for a specified column from being added to a table.

PROC SQL;
CREATE TABLE work.RENTAL_INFO
   (TITLE CHAR(30) UNIQUE,
    RENTAL_AMT NUM FORMAT=DOLLAR6.2);
QUIT;
CHECK Constraint

A CHECK constraint can be specified to assign specific rules that a column must adhere to.

PROC SQL;

ALTER TABLE MOVIES
ADD CONSTRAINT CHECK_RATING
CHECK (RATING IN ('G', 'PG', 'PG-13', 'R'));

QUIT;
Exploring a User-defined Tool using Dictionary Tables
Dictionary Tables / SASHELP Views

• SAS collects information about a session
• Session information is captured as read-only content
• Tables are accessible using PROC SQL
  ✓ Specify table in FROM clause of a SELECT
  ✓ DICTIONARY libref is automatically assigned

• SASHELP Views are accessed in a DATA step or with any of your favorite PROCs
Viewing Dictionary Tables / Views

• # of DICTIONARY Tables/Views:
  ✓ 22 in SAS 9.1
  ✓ 29 in SAS 9.2
  ✓ 30 in SAS 9.3
PROC PRINT DATA=SASHELP.VCOLUMN NOOBS;
VAR LIBNAME MEMNAME NAME TYPE LENGTH;
WHERE UPCASE(LIBNAME)=UPCASE("SASUSER") AND
UPCASE(NAME)=UPCASE("TITLE");
RUN;

Cross-reference listing for the column TITLE

<table>
<thead>
<tr>
<th>Library Name</th>
<th>Member Name</th>
<th>Column Name</th>
<th>Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>SASUSER</td>
<td>ACTORS</td>
<td>Title</td>
<td>char</td>
<td>30</td>
</tr>
<tr>
<td>SASUSER</td>
<td>MOVIES</td>
<td>Title</td>
<td>char</td>
<td>30</td>
</tr>
</tbody>
</table>
%MACRO CROSSREF(LIB, COLNAME);
  PROC PRINT DATA=SASHELP.VCOLUMN NOOBS;
   VAR LIBNAME MEMNAME NAME TYPE LENGTH;
   WHERE UPCASE(LIBNAME)=UPCASE("&LIB") AND UPCASE(NAME)=UPCASE("&COLNAME");
  RUN;
%MEND CROSSREF;
%CROSSREF(SASUSER,TITLE);
Cross-reference Listing – PROC PRINT

PROC PRINT DATA=SASHELP.VCOLUMN NOOBS;
  VAR LIBNAME MEMNAME NAME TYPE LENGTH;
  WHERE UPCASE(LIBNAME)="SASUSER" AND
       UPCASE(NAME)="TITLE";
RUN;

Cross-reference listing for the column TITLE

<table>
<thead>
<tr>
<th>Library Name</th>
<th>Member Name</th>
<th>Column Name</th>
<th>Type</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>SASUSER</td>
<td>ACTORS</td>
<td>Title</td>
<td>char</td>
<td>30</td>
</tr>
<tr>
<td>SASUSER</td>
<td>MOVIES</td>
<td>Title</td>
<td>char</td>
<td>30</td>
</tr>
</tbody>
</table>
Conclusion

User-defined Tool using Dictionary Tables

Useful SAS System Options

PROC SQL Options

Integrity Constraints for Tables

User-defined Function using PROC FCMP
Thank You for Attending!

Questions?

A presentation by

Kirk Paul Lafler
KirkLafler@cs.com
@sasNerd