SAS 9.4 Advanced Programming – Performance Based Exam

Accessing Data Using SQL (35%)

Generate detail reports by working with a single table, joining tables, or using set operators in SQL

- Use PROC SQL to perform SQL queries.
- Select columns in a table with a SELECT statement and FROM clause.
- Create a table from a query result set.
- Create new calculated columns.
- Assign an alias with the AS keyword.
- Use case logic to select values for a column.
- Retrieve rows that satisfy a condition with a WHERE clause.
- Subset data by calculated columns.
- Join tables - inner joins, full joins (coalesce function), right joins, left joins.
- Combine tables using set operators - union, outer union, except, intersect.
- Sort data with an ORDER BY clause.
- Assign labels and formats to columns.

Generate summary reports by working with a single table, joining tables, or using set operators in the SQL.

- Summarize data across and down columns using summary functions (AVG, COUNT, MAX, MIN, SUM).
- Group data using GROUP BY clause.
- Filter grouped data using HAVING clause.
- Eliminate duplicate values with the DISTINCT keyword.

Construct sub-queries and in-line views within an SQL procedure step.

- Subset data by using non-correlated subqueries.
- Reference an in-line view with other views or tables (multiple tables).

Use SAS SQL procedure enhancements.

- Use SAS data set options with PROC SQL (KEEP=, DROP=, RENAME=, OBS=).
- Use PROC SQL invocation options (INOBS=, OUTOBS=. NOPRINT, NUMBER)
- Use SAS functions (SCAN, SUBSTR, LENGTH).
- Access SAS system information by using DICTIONARY tables (members, tables, columns)
- Use the CALCULATED keyword.

Macro Processing (35%)

Create and use user-defined and automatic macro variables within the SAS Macro Language.

- Define and use macro variables.
- Use macro variable name delimiter. (.)
- Use INTO clause of the SELECT statement in SQL to create a single variable or a list of variables.
- Use the SYMPUTX routine in a DATA Step to create a single variable or a list of variables.
- Control variable scope with:
    - %GLOBAL statement
    - %LOCAL statement
    - SYMPUTX scope parameter

Automate programs by defining and calling macros using the SAS Macro Language.

- Define a macro using the %MACRO and %MEND statements.
- Calling a macro with and without parameters.
- Document macro functionality with comments
- Generate SAS Code conditionally by using the %IF-%THEN-%ELSE macro statements or iterative %DO statements.
- Use the SAS AUTOCALL facility to permanently store and call macros.

Use macro functions.

- Use macro functions. (%SCAN, %SUBSTR, %UPCASE)
- Use macro quoting functions. (%NRSTR, %STR)
- Use macro evaluation functions. (%SYSEVALF)
- Use %SYSFUNC to execute DATA step functions within the SAS Macro Language.

Debug macros.

- Trace the flow of execution with the MLOGIC option.
- Examine the generated SAS statements with the MPRINT option.
- Examine macro variable resolution with the SYMBOLGEN option.
- Use the %PUT statement to print information to the log.

Create data-driven programs using SAS Macro Language.

- Create a series of macro variables.
- Use indirect reference to macro variables. (&&, etc.)
- Incorporate DICTONARY tables in data driven macros.
- Generate repetitive macro calls.

Advanced Techniques (30%)

Process data using 1 and 2 dimensional arrays.

- Define and use character arrays.
- Define and use numeric arrays.
- Create variables with arrays.
- Reference arrays within a DO loop.
- Specify the array dimension with the DIM function.
- Define arrays as temporary arrays.
- Load initial values for an array from a SAS data set.

Process data using hash objects.

- Declare hash and hash iterator objects
  - Dataset argument
  - Ordered argument
  - Multidata argument
- Use hash object methods
  - definekey()
  - definedata()
  - definedone()
  - find()
  - add()
  - output()
- Use hash iterator object methods
  - first()
  - next()
  - last()
  - prev()
- Use hash objects as lookup tables.
- Use hash objects to create sorted data sets.
- Use hash iterator objects to access data in forward or reverse key order.

Use SAS utility procedures.

- Specify a template using the PICTURE statement within the FORMAT Procedure*
  - Specify templates for date, time, and datetime values using directives.
  - Specify templates for numeric values using digit selectors.
  - PICTURE statement options: round, default, datatype, multiplier, prefix
- Create custom functions with the FCMP procedure
  - Create character and numeric custom functions with single or multiple arguments.
  - Create custom functions based on conditional processing.
  - Use custom functions with the global option CMPLIB=.

Use advanced functions.

- Finding strings or words with the FINDC/FINDW functions.
- Counting strings or words with the COUNT/COUNTC/COUNTW functions.
- Retrieve previous values with the LAG function.
- Regular expression pattern matching with PRX functions*
    - Metacharacters: ()[]{}*+?.|^$\d\D\s\S\w\W
    - Functions and call routines: PRXMATCH, PRXPARSE, PRXCHANGE

Note: All 13 main objectives will be tested on every exam. The additional details provide for additional explanation and define the entire domain that could be tested.

** For these topics, a reference aid is provided during the exam to assist with directives/metacharacters.*