SAS 9.4 Advanced Programming – Performance Based Exam

Accessing Data Using SQL

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 with the CALCULATED keyword.
- Join tables - inner joins, full joins (coalesce function), right joins, left joins, cross joins.
- Combine tables using set operators - union, outer join, 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 special features of the SQL procedure.

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

Macro Processing

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.
- Use the SYMPUTX routine in a DATA Step.
- 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.
- Insert comments into macros.
- Pass Information into a macro using parameters.
- Generate SAS Code conditionally by using the %IF-%THEN-%ELSE macro statements or iterative %DO statements.

Use macro functions.

- Use macro character functions. (%SCAN, %SUBSTR, %INDEX, %UPCASE)
- Use macro quoting functions. (%NRSTR, %STR)
- Use macro evaluation functions. (%EVAL)
- 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.
- Create a macro variable containing a delimited list of values using PROC SQL.
- Use indirect reference to macro variables. (&&, etc)
- Generate repetitive macro calls using:
    - the %DO loop,
    - SQL query with SELECT INTO
    - DATA Step with DOSUBL or the EXECUTE routine function.

Advanced Techniques

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.
- 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=.

Note: All 12 main objectives will be tested on every exam. The 62 expanded objectives are provided for additional explanation and define the entire domain that could be tested.