

SAS Advanced Programming for SAS 9 Exam

Accessing Data Using SQL

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

- Use the SELECT statement.
- Select columns in a table.
- Create new columns.
- Sort data.
- Retrieve rows that satisfy a condition.
- Validate a query.
- Join tables - inner joins, full joins, right joins, left joins.
- Combine tables using set operators - union, outer join, except, intersect.

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

- Summarize data.
- Group data.
- Filter grouped data.

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

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

Compare solving a problem using the SQL procedure versus using traditional SAS programming techniques.

- Use SAS data set options with PROC SQL.
- Use PROC SQL with the SAS Macro Facility.
- Create SAS Data sets (tables).
- Insert rows into tables.
- Update data values in a table.
- Delete rows.
- Alter columns attributes.
- Create an index.
- Delete a table.

Access Dictionary Tables using the SQL procedure.

- Access SAS system information by using DICTIONARY tables.
- Use the DESCRIBE TABLE statement.

Macro Processing

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

- Define Macro variables.
- Use %GLOBAL statement.
- Use %INPUT statement.
- Use INTO clause of the SELECT statement in SQL.
- Use %LOCAL statement.
- Use the SYMPUT and SYMPUTX routine in the DATA Step.
- Use the SYMGET function to return the value of a macro variable to the DATA step during DATA step execution.

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

- Define a macro.
- Use the %MACRO statement.
- 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.

Understand the use of macro functions.

- Use macro character functions.
- Use macro quoting functions.
- Use macro evaluation functions.

Use various system options that are available for macro debugging and displaying values of user-defined and automatic macro variables in the SAS log.

- Use system options to track problems.
- Trace the flow of execution with MLOGIC.
- Examine the generated SAS statements with MPRINT.
- Examine macro variable resolution with SYMBOLGEN.
- Use the %PUT statement to track problems.

Create data-driven programs using SAS Macro Language.

- Create macro variables with a list of values.
- Use indirect reference to macro variables.
- Generate repetitive macro calls using the %DO loop, macro variable, and the EXECUTE routine.

Advanced Programming Techniques

Demonstrate the use of advanced data look-up techniques such as array processing, hash objects, formats, and merging.

- Combine data using multiple set statements with KEY= option.
- Combine data conditionally using multiple set statements.
- Combine multiple data sets using FILEVAR= option.
- Compare DATA step match-merge and PROC SQL joins.
- Use formats to create data via lookups.
- Use hash objects as lookup tables.
- Manage custom formats with FMTSEARCH= system option.
- Create custom formats with the PICTURE statement.
- Process data with multi-dimensional arrays.

Reduce computing resource requirements by controlling the space required to store SAS data sets.

- Use compression techniques, RLE (Run-Length Encoding) and RDC (Ross Data Compression).
- Reduce length of numeric variables.
- Eliminate variables and observations.
- Use SAS views.

Use the FCMP procedure to create a user-defined function.

- Define a SAS function. Example:

```
PROC FCMP OUTLIB=libref.data-set.package;
  FUNTION function-name(argument-1 <$>,...,argument-m<$>) <$>;
    Programing statements
  RETURN(expression);
  ENDSUB;
QUIT;
```

Perform effective benchmarking.

- Understand resources related to efficiency.
- Use SAS System options to track resources.

- Interpret the resulting resource utilization statistics for the Z/OS environment and for directory based OS.

Use SAS indexes.

- Identify appropriate applications for using indexes.
- Create and delete indexes using the DATA step, the DATASETS procedure, or the SQL procedure.

Compare techniques to eliminate duplicate data.

- Use the DATA step.
- Use the SORT procedure.
- Use the SQL procedure.

Note: All 16 main objectives will be tested on every exam. The 68 expanded objectives are provided for additional explanation and define the entire domain that could be tested.