**Clinical SAS Trainer:** NAIDU
**E-mail:** contact@greatonlinetraining.com
**India:** +91 9966956770
**USA:** +1 (551) 226-606
**Whatsapp:** +91 9966956770
**www.greatonlinetraining.com**

# SAS 9.4 Advanced Programming – Performance Based Exam

**SAS9.4 Programming fundamentals eaxm:**

- This exam is administered by SAS and Pearson VUE.
- 60-65 multiple choice and short-answer questions.
- 120 minutes to complete exam.
- Must achieve a score of 68% to pass.
- This exam is based on SAS 9.4 M5.

**Use this exam ID to register:**
**A00-215**

## Pricing %

$120 Exam fee in the US and most other countries.

## 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()
  - ast()
  - 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.*

## Sample Questions

*The following sample questions are not inclusive and do not necessarily represent all of the types of questions that comprise the exams. The questions are not designed to assess an individual's readiness to take a certification exam.*

## SAS 9.4 Advanced Programming Performance-Based Exam

### Performance-Based Programming Questions

*Note: The programming projects are assessed with a scoring macro that is stored on the lab computer. At the end of your project, you will invoke the scoring macro and it will investigate the results of your project. It will look at parameters and content of output data set as well as values of macro variables stored in the symbol tables. This macro will also investigate the code that you wrote to check that the problem was solved as requested. These are broad checks, so there is still a significant amount of freedom in your chosen coding solution. For example, in the SQL topic, we want to ensure that an SQL procedure was used to create the output data set rather than a DATA Step. The scoring macro will return a 3-digit value to the SAS log. You will record this 3-digit value as your answer to the project to determine your score for the project.*

#### Question 1

Open a new programming window to create **ACT01.sas** in **c:\cert\programs**.

Write a SAS program that will:

- Create output data set **work.ACT0**1 using **sashelp.pricedata** as input.
- Use an array to increase the values of the **price1** through **price17** variables by 10%.

Run your program and troubleshoot as necessary. When you are finished with the project:

1. Ensure that you have saved your program as **ACT01.sas** in **c:\cert\programs**.
2. From **the score.sas** program, call the **scoreit** macro using **ACT01** as the parameter: **%scoreit(ACT01).**

What is the value for Response in the SAS log? ___

*Correct Solution: All price values for all price1-through price17 will be increased by 10%. For example, price2 in observation 5 will now be 126.50. Arrays and do loops would be used in the program.*

#### Question 2

Open a new programming window to create **MAC01**.sas in **c:\cert\programs**.

Write a DATA step that reads only the first observation of the **sashelp.cars** data set and stores the value of the **Make** variable in a macro variable named **CarMaker**.  The macro variable must be defined from within the DATA Step.

Run your program and troubleshoot as necessary. When you are finished with the project:

1. Ensure that you have saved your program as MAC01.sas in c:\cert\programs.
2. From the **score.sas pro**gram, call the **scoreit** macro using MAC01 as the parameter: **%scoreit(MAC01)**.

What is the value for Response in the SAS log?  __

*Correct Solution: The CarMaker macro variable will have a value of Acura. The program will include a symputx routine.*

## Question 3

Open a new programming window to create **SQL01**.sas in **c:\cert\programs**.

Write an SQL query that will:

- Create output data set **work.SQL01** using **sashelp.cars** as input.
- Compute the average **MPG_City** for each group of **Make**. Name the calculated variable **AvgCityMPG**.
- The output data should have 2 columns, **Make** and **AvgCityMPG**.

Run your program and troubleshoot as necessary. When you are finished with the project:

1. Ensure that you have saved your program as **SQL01.sas** in c:\cert\programs.
2. From the **score.sas** program, call the **scoreit** macro using SQL01 as the parameter: **%scoreit(SQL01)**.

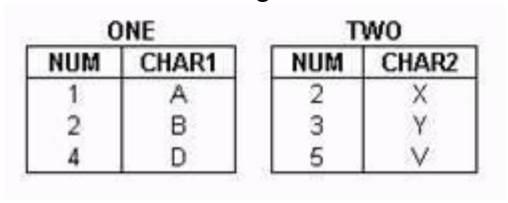
What is the value for Response in the SAS log?  __

*Correct Solution: An SQL query with a group by clause will be written. The AvgCityMPG for MAKE=MINI will be 26.5.*

## Standard Questions

### Question 4

Given the following SAS data sets ONE and TWO:



ONE

| NUM | CHAR1 |
|---|---|
| 1 | A |
| 2 | B |
| 4 | D |

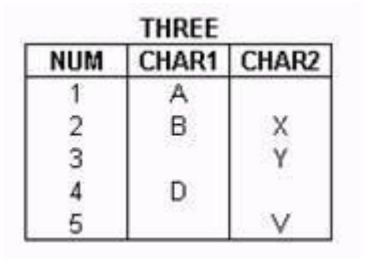
TWO

| NUM | CHAR2 |
|---|---|
| 2 | X |
| 3 | Y |
| 5 | V |

The following SAS program is submitted creating the output table THREE:

```
data three;
merge one (in = in1) two (in = in2);
by num;

run;
```



THREE

| NUM | CHAR1 | CHAR2 |
|---|---|---|
| 1 | A | |
| 2 | B | X |
| 3 | | Y |
| 4 | D | |
| 5 | | V |

Which one of the following SQL programs creates an equivalent SAS data set THREE?

A.
```
proc sql; create table
three as    select *
       from one full join two
 where one.num = two.num; quit;
```

B.
```
proc sql; create table
three as    select
coalesce(one.num,
two.num)       as NUM,
char1, char2       from
one full join two
where one.num =
two.num; quit;
```

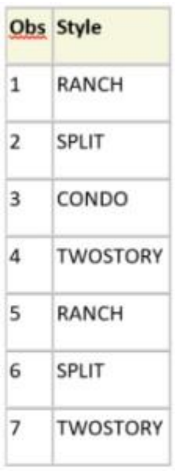
C.
```
C. proc sql; create
table three as
select one.num, char1,
char2 from one full
join two on one.num =
two.num;
quit;
```

D. D.
```
proc sql; create
table three as
select
coalesce(one.num,two.n)
as NUM, char1, char2
from one full join two
on one.num = two.num;
quit;
```

**correct answer = "D"**

Question 5

Given the following SAS data set Sasuser.Houses:



| Obs | Style |
|---|---|
| 1 | RANCH |
| 2 | SPLIT |
| 3 | CONDO |
| 4 | TWOSTORY |
| 5 | RANCH |
| 6 | SPLIT |
| 7 | TWOSTORY |

You submit the following SAS program:

```
proc sql noprint;
select distinct
style
into :styles
separated by ' '
from sasuser.houses
order by style;
quit;
```

What is the value of the resulting macro variable?

A. CONDO
B. TWOSTORY
C. CONDO RANCH SPLIT TWOSTORY
D. RANCH SPLIT CONDO TWOSTORY

**correct answer = "C"**

Question 6

Given the following SQL procedure output:

| Table | Physical Obs | % Deleted |
|---|---|---|
| EMPLOYEE_ADDRESSES | 424 | 5.0% |
| EMPLOYEE_PAYROLL | 424 | 5.0% |

Which SQL query will produce a report for tables in the ORION library which have had at least 5% of their physical rows deleted, as shown above?

A.
```
select MEMNAME 'Table', NOBS 'Physical Obs'
DELOBS/NOBS   '%   Deleted'   format=percent6.1
from dictionary.tables
where LIBNAME='ORION' AND DELOBS/NOBS >= .05;
```

B.
```
select Table_Name, Num_Rows 'Physical Obs'
,Deleted_Rows/Num_Rows'%Deleted'
format=percent6.1
from dictionary.DBA_TABLES where
TABLESPACE_NAME='ORION'   AND
Deleted_Rows/Num_Rows>= .05;
```

C.
```
select MEMNAME 'Table', NLOBS 'Physical Obs'
, DELOBS/NLOBS LABEL='% Deleted'
format=percent6.from dictionary.tables
where LIBNAME='ORION' AND
DELOBS/NLOBS >=.05;
```

D.
```
select MEMNAME 'Table', NOBS 'Physical  Obs',
DELOBS/NOBSLABEL='%Deleted'
format=percent6.1
from dictionary.members
where LIBNAME='ORION' AND DELOBS/NOBS >= .05;
```

**correct answer = "A"**

## Question 7

The following SAS program is submitted:

```
options                        ;
%abc(work.look,Hello,There);
```

In the text box above, complete the options statement that will produce the following log messages:

```
M*****(ABC):   title1 "Hello" ;
M*****(ABC):   title2 "There" ;
M*****(ABC):   proc print data=work.look ;
M*****(ABC):   run ;
```

**Correct answer = "mprint"**

## Question 8

The following SAS program is submitted:

```
%macro mysum(n);
  %if &n > 1 %then %eval (%n + %mysum
(%eval (&n-1))) ;
%else &n;
 %mend;
%put %mysum(4);
```

Which output is written to the log?

A. 10

B. 4+3+2+1

C. 7

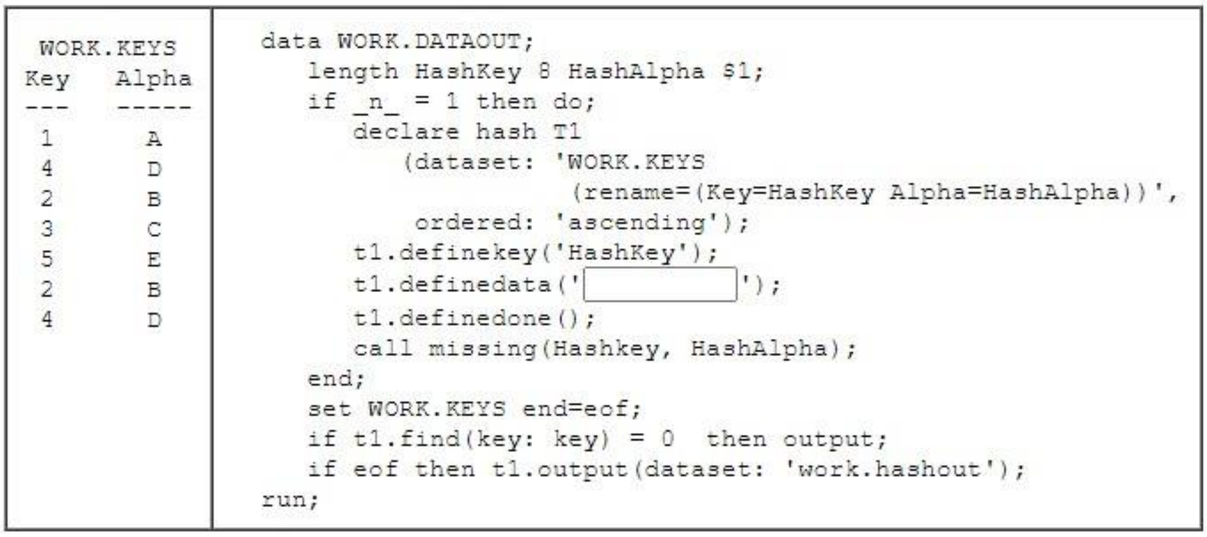
D. A character operand was found in the %EVAL function or %IF condition where a numeric operand is required.

**correct answer = "A"**

Question 9

This question will ask you to provide a segment of missing code.

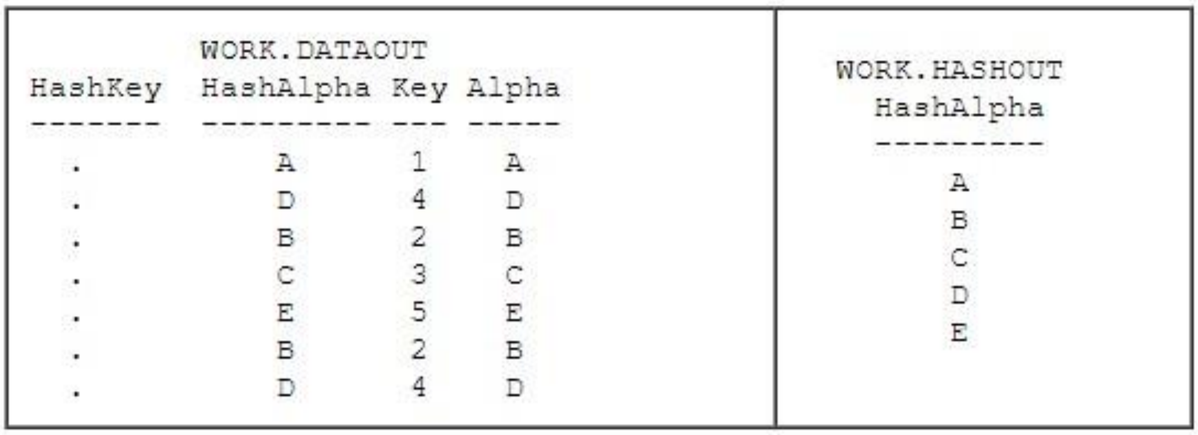
The WORK.KEYS data set is shown on the left. The SAS program shown on the right is submitted.



```
WORK.KEYS
Key   Alpha
---   -----
 1      A
 4      D
 2      B
 3      C
 5      E
 2      B
 4      D
```

```
data WORK.DATAOUT;
   length HashKey 8 HashAlpha $1;
   if _n_ = 1 then do;
      declare hash T1
         (dataset: 'WORK.KEYS
                  (rename=(Key=HashKey Alpha=HashAlpha))',
          ordered: 'ascending');
      t1.definekey('HashKey');
      t1.definedata('          ');
      t1.definedone();
      call missing(Hashkey, HashAlpha);
   end;
   set WORK.KEYS end=eof;
   if t1.find(key: key) = 0  then output;
   if eof then t1.output(dataset: 'work.hashout');
run;
```

In the text box above, enter the code to complete the program so that it will produce the output shown below:



```
            WORK.DATAOUT
HashKey   HashAlpha  Key  Alpha
-------   ---------  ---  -----
   .          A       1     A
   .          D       4     D
   .          B       2     B
   .          C       3     C
   .          E       5     E
   .          B       2     B
   .          D       4     D
```

```
WORK.HASHOUT
 HashAlpha
 ---------
     A
     B
     C
     D
     E
```

Case is ignored and standard SAS syntax rules apply.

correct answer = "HashAlpha"