SAS Advanced Programming Certification Practice Exam (Sample)

Study Guide

BY EXAMZIFY

Everything you need from our exam experts!

Questions

1. **How do numeric and character variables differ in SAS?**
   - **A. Numeric variables can store text, while character variables cannot**
   - **B. Character variables hold numerical values for calculations**
   - **C. Numeric variables store numerical values and character variables store text**
   - **D. There is no difference between them**

2. **Which declare statement has correct syntax?**
   - **A. declare hash class (dsn:pg3.class_birthdate);**
   - **B. declare hash class(data=pg3.class_birthdate);**
   - **C. declare hash class(data='pg3.class_birthdate);**
   - **D. declare hash class(dataset:'pg3.class_birthdate');**

3. **In SAS, what does the DIM function do when used in conjunction with an array?**
   - **A. It returns the number of dimensions of the array**
   - **B. It returns the number of elements in the array**
   - **C. It initializes the array with default values**
   - **D. It creates a permanent array in the PDV**

4. **Which query finds the ID, Job_Title, and Salary values of all employees in the Engineering Department?**
   - **A. select ID, JOb_TItle, Salary from staff where ID in (select ID from staff where Department = "Engineering")**
   - **B. select ID, Job_Title, Salary from staff where ID = (select ID from staff where Department = "Engineering")**
   - **C. select ID, Job_Title, Salary from staff where Department = "Engineering"**
   - **D. select all from staff where Job_Title = (select Job_Title from staff where Department = "Engineering")**

5. **What is the use of the WHERE statement in PROC SQL?**
   - **A. To define new variables based on conditions**
   - **B. To filter rows based on specific conditions**
   - **C. To format the output results of a query**
   - **D. To create indexes for a dataset**

6. **What information does PROC CONTENTS provide about a dataset?**
   - **A. Only the first few records of the dataset**
   - **B. Metadata information including variable types and dataset attributes**
   - **C. Graphs and charts representing the data**
   - **D. Summary statistics of the dataset**

7. **True or False: The RETURN statement can have the custom function return multiple values.**
   - **A. True**
   - **B. False**
   - **C. Depends on the function**
   - **D. None of the above**

8. **How does the SAS system handle character data types?**
   - **A. As integers with defined lengths**
   - **B. As floating-point numbers**
   - **C. As strings with defined lengths**
   - **D. As binary objects**

9. **What is a common use case for PROC FEDSQL in data management?**
   - **A. Data entry and validation**
   - **B. Data cleaning and transformation**
   - **C. Data warehousing**
   - **D. Data retrieval from a database**

10. **Which program correctly displays a list of qualified column names in the SAS log?**
   - **A. proc sql; describe table acme.staff; quit;**
   - **B. proc sql noexec; select * from acme.staff; quit;**
   - **C. proc sql list; select * from acme.staff; quit;**
   - **D. proc sql feedback; select * from acme.staff; quit;**

Answers

1. **C**
2. **D**
3. **B**
4. **A**
5. **B**
6. **B**
7. **B**
8. **C**
9. **D**
10. **B**

Explanations

1. How do numeric and character variables differ in SAS?

A. Numeric variables can store text, while character variables cannot

B. Character variables hold numerical values for calculations

<u>C. Numeric variables store numerical values and character variables store text</u>

D. There is no difference between them

Numeric and character variables in SAS serve distinct purposes and are fundamental to data management within the system. Numeric variables are designed specifically to hold numerical values, such as integers or decimals, which can be used for various mathematical operations, including addition, subtraction, multiplication, and division. This capability is crucial for performing calculations, statistical analyses, and mathematical modeling. On the other hand, character variables are used to store text data. This can include letters, symbols, and numeric values that are treated as text rather than something to be calculated. For instance, a character variable could store the zip code "12345" or a string of characters like "John Doe," where the numerical aspect holds no arithmetic relevance. The distinction between numeric and character types is essential for data integrity and ensuring that analyses are performed correctly. The functions and operations available in SAS differ based on the type of variable, making understanding this difference vital when programming in SAS. Choosing the appropriate variable type based on the data you are working with directly impacts the accuracy and efficiency of data processing tasks.

2. Which declare statement has correct syntax?

A. declare hash class (dsn:pg3.class_birthdate);

B. declare hash class(data=pg3.class_birthdate);

C. declare hash class(data='pg3.class_birthdate);

<u>D. declare hash class(dataset:'pg3.class_birthdate');</u>

The correct syntax for declaring a hash object in SAS requires the use of the `data=` option, which specifies the data source that the hash object will be working with. The correct choice uses the proper syntax for declaring a hash object with the `dataset:` keyword, which indicates that it is referring to a dataset in SAS. The option includes the necessary structure for pointing to the dataset by using proper syntax, allowing the hash object to function correctly within the Data Step. The declaration command formats the object as a hash table that will hold the values from the specified dataset. In contrast, options that use `dsn:` or lack proper quotes may not adhere to the required SAS language standards, leading to syntax errors. Correct syntax is crucial for SAS programs to execute properly, so adhering to the established conventions is important for successful implementation.

3. In SAS, what does the DIM function do when used in conjunction with an array?

A. It returns the number of dimensions of the array

B. It returns the number of elements in the array

C. It initializes the array with default values

D. It creates a permanent array in the PDV

The DIM function in SAS is designed to provide the number of elements in an array. When you declare an array in SAS, the DIM function can be utilized to get the size of that array, which refers to how many variables or elements are included within it. This capability is crucial when iterating through the elements of the array or when performing operations that require an understanding of how many elements are being referenced. Understanding the distinction between arrays and their dimensions is important, as a one-dimensional array would just have a length defined by its number of elements. While two-dimensional arrays do exist in SAS, typically the DIM function is used to address a specific array of single dimension to determine its length directly. This role is fundamental in the manipulation of data, allowing programmers to dynamically handle datasets based on their varying sizes. The other possible answers do not accurately reflect the functionality of the DIM function within the context of an array. For instance, while one might consider dimensions in relation to arrays, the DIM function does not return the number of dimensions but rather focuses on counting elements. Initialization of arrays with default values or creating permanent arrays are functions that would involve different statements and procedures in SAS, thus further clarifying the specificity of the DIM function's role.

4. Which query finds the ID, Job_Title, and Salary values of all employees in the Engineering Department?

A. select ID, JOb_TItle, Salary from staff where ID in (select ID from staff where Department = "Engineering")

B. select ID, Job_Title, Salary from staff where ID = (select ID from staff where Department = "Engineering")

C. select ID, Job_Title, Salary from staff where Department = "Engineering"

D. select all from staff where Job_Title = (select Job_Title from staff where Department = "Engineering")

The correct approach to finding the ID, Job_Title, and Salary values of all employees in the Engineering Department is to use a query that retrieves rows based on their department classification. The third option provides a direct and effective solution to this requirement. The query that correctly retrieves the desired information specifies the condition to only include employees whose Department matches "Engineering." This means that it directly filters the data from the staff table based on a specific criterion, ensuring that only the relevant employees are selected. By selecting ID, Job_Title, and Salary in this way, the query returns results tailored specifically to the Engineering Department. In contrast, the other options introduce complexities or incorrect conditions. The first option, while it might seem plausible, utilizes a subquery that checks for IDs in a nested way, which is unnecessarily complicated for this task. Similarly, the second option suggests an approach that only works for a single ID, which would limit the results to one employee only. Lastly, the fourth option attempts to select all rows based on a Job_Title derived from a subquery, which doesn't filter for department-specific criteria but instead relies on job titles that may not be unique or applicable to all Engineering employees. Therefore, the most efficient and accurate method to achieve the objective is to

5. What is the use of the WHERE statement in PROC SQL?

A. To define new variables based on conditions

B. To filter rows based on specific conditions

C. To format the output results of a query

D. To create indexes for a dataset

The WHERE statement in PROC SQL is primarily utilized to filter rows based on specific conditions. This functionality allows the user to specify criteria that determine which records are included in the result set of the SQL query. By using the WHERE clause, you can narrow down the data to only those rows that meet your specified conditions. For example, if you're interested in retrieving records of employees with a salary greater than a specific amount, the WHERE clause would enable you to extract only those relevant entries from the dataset. In contrast, the other provided options serve different purposes. Defining new variables based on conditions is typically handled through the CASE statement or other data manipulation techniques, rather than the WHERE statement. Formatting output results is generally achieved using the FORMAT statement or attributes in SQL, rather than filtering directly. Creating indexes for a dataset involves methods that are focused on improving query performance but is not related to the WHERE statement's functionality.

6. What information does PROC CONTENTS provide about a dataset?

A. Only the first few records of the dataset

B. Metadata information including variable types and dataset attributes

C. Graphs and charts representing the data

D. Summary statistics of the dataset

PROC CONTENTS in SAS is designed to provide detailed metadata information about a dataset. This includes essential characteristics such as the names of the variables, their respective types (e.g., numeric or character), the length of each variable, and any associated attributes like labels and formats. Additionally, it provides the number of observations and variables in the dataset, offering an overview of the dataset's structure. This specific focus on metadata distinguishes PROC CONTENTS from other procedures. For example, it does not display actual data records or graphical representations, nor does it calculate summary statistics. Instead, it presents a structured overview that is crucial for understanding the dataset in preparation for analysis or manipulation. This makes option B the correct answer, as it accurately describes the comprehensive informational output that PROC CONTENTS provides about datasets.

7. True or False: The RETURN statement can have the custom function return multiple values.

A. True

B. False

C. Depends on the function

D. None of the above

The RETURN statement in SAS functions is designed to return a single value and is fundamentally limited in that respect. When a function is called, it executes code and computes a result; the RETURN statement conveys that result back to the caller. Unlike some programming languages that allow returning multiple values, SAS maintains a one-value-return policy with its RETURN statement. In practice, if there is a need to return multiple pieces of data from a function, they would typically be returned as part of a data structure—such as an observation in a dataset, using a macro variable, or through global variables—rather than through a RETURN statement. This is a key aspect of effective coding in SAS, as it encourages proper structuring of data into datasets or using arrays when multiple values need to be managed together. Considering the context, some might mistakenly think it can return multiple values either due to misunderstanding how RETURN works or because they may confuse it with features available in other programming languages. However, in SAS, the premise holds that a function defined with a RETURN statement can only yield a single value upon execution.

8. How does the SAS system handle character data types?

A. As integers with defined lengths

B. As floating-point numbers

C. As strings with defined lengths

D. As binary objects

The SAS system manages character data types by treating them as strings with defined lengths. When a character variable is created in SAS, you specify a length for that variable, which determines how many characters can be stored in it. This length is important because it influences memory allocation and how data is processed. SAS allows for the manipulation of these character strings through various functions and can handle them effectively in different programming scenarios. Character data types in SAS are designed to store text data, such as names, addresses, or any other non-numeric information. The defined length ensures that the data fits within the allocated space, as exceeding this length would result in truncation. Consequently, the way SAS organizes and operates on strings is integral to how it processes character data. Thus, option C accurately describes the handling of character data types in the SAS system.   Other options fall short because integers and floating-point numbers pertain to numeric data types, and binary objects refer to data structures that do not apply to standard character handling in SAS.

9. What is a common use case for PROC FEDSQL in data management?

A. Data entry and validation

B. Data cleaning and transformation

C. Data warehousing

D. Data retrieval from a database

PROC FEDSQL is specifically designed to facilitate efficient data retrieval from a variety of database systems. This procedure provides users with the ability to run SQL queries to access data stored in both SAS datasets and databases such as relational databases, making it a powerful tool for extracting valuable insights from large datasets. In particular, PROC FEDSQL supports the SQL syntax which can work with data stored in different formats, allowing users to create complex queries, join multiple tables, and filter results based on specific criteria. This capability makes PROC FEDSQL especially useful in situations where users need to gather data from various sources into a coherent output that can be further analyzed or reported on. It is worth noting that while PROC FEDSQL may have functionalities that overlap with data cleaning or transformation indirectly during the retrieval or preparation of data for analysis, its primary and most common use case is focused on efficiently retrieving data from databases. This differentiates it from the functions more directly associated with other specific data manipulation tasks such as data entry, validation, or warehousing practices.

10. Which program correctly displays a list of qualified column names in the SAS log?

A. proc sql; describe table acme.staff; quit;

B. proc sql noexec; select * from acme.staff; quit;

C. proc sql list; select * from acme.staff; quit;

D. proc sql feedback; select * from acme.staff; quit;

The correct answer showcases the use of the NOEXEC option in PROC SQL. When you include the NOEXEC option, SAS prepares the SQL statement for execution but does not actually execute it. Instead, it will generate a report in the SAS log that includes information about the query, including the qualified column names that are in the table acme.staff. This is a powerful feature for examining the structure of your data without running the full query, which can be especially useful when working with large datasets or when you're only interested in the metadata associated with the data you are querying. The other choices do not provide the same functionality. For example, using the DESCRIBE TABLE statement allows you to get details about the table structure, but it requires the table to be in the active library and formatted properly, thus it’s not the best choice for simply listing qualified column names. The LIST option does not exist as a valid option in PROC SQL syntax, and the FEEDBACK option generates feedback in the log about the processing but does not specifically list qualified column names.