Vendor:SAP

Exam Code:C_ABAPD_2309

Exam Name:SAP Certified Associate - Back-End Developer - ABAP Cloud

Version: Demo

QUESTION 1

When processing an internal table with the statement LOOP AT itab... ENDLOOP, what system variable contains the current row number?

A. sy-index

B. sy-subrc

C. sy-linno

D. sy-tabix

Correct Answer: D

When processing an internal table with the statement LOOP AT itab... ENDLOOP, the system variable that contains the current row number is sy-tabix. The sy-tabix variable is a predefined field of the system structure sy that holds the index or the row number of the current line in an internal table loop. The sy-tabix variable is initialized with the value 1 for the first loop pass and is incremented by 1 for eachsubsequent loop pass. The sy-tabix variable can be used to access or modify the current line of the internal table using the index access12. References: 1: LOOP AT itab - ABAP Keyword Documentation - SAP Online Help 2: System Fields - ABAP Keyword Documentation - SAP Online Help

---

QUESTION 2

What are some features of a unique secondary key? Note: There are 2 correct answers to this question.

A. It is created when a table is filled.

B. It is updated when the modified table is read again.

C. It is created with the first read access of a table.

D. It is updated when the table is modified.

Correct Answer: CD

A unique secondary key is a type of secondary key that ensures that the key combination of all the rows in a table is unique. A unique secondary key has two purposes: firstly, to speed up access to the table, and secondly, to enforce data

integrity1. It is created with the first read access of a table: This is true. A unique secondary key is created when an internal table is filled for the first time using the statement READ TABLE or a similar statement. The system assigns a name

and an index to each row of the table based on the key fields23.

It is updated when the modified table is read again: This is false. A unique secondary key does not need to be updated when the internal table content changes, because it already ensures data uniqueness. The system uses a lazy update

strategy for non-unique secondary keys, which means that it delays updating them until they are actually accessed23.

You cannot do any of the following:

It is created when a table is filled: This is false. As explained above, a unique secondary key is created only with the first read access of a table23. It is updated when the modified table is read again: This is false. As explained above, a unique

secondary key does not need to be updated when the internal table content changes23.

References: 1: Improving Internal Table Performance Using Secondary Keys - SAP Learning 2: [Secondary Key - ABAP Keyword Documentation - SAP Online Help] 3:

[Secondary Table Key - ABAP Keyword Documentation - SAP Online Help]

QUESTION 3

Class super has subclass sub. Which rules are valid for the sub constructor? Note: There are 2 correct answers to this question.

A. The method signature can be changed.

B. Import parameters can only be evaluated after calling the constructor of super.

C. The constructor of super must be called before using any components of your own instance.

D. Events of your own instance cannot be raised before the registration of a handler in super.

Correct Answer: AC

The sub constructor is the instance constructor of the subclass sub that inherits from the superclass super. The sub constructor has some rules that it must follow when it is defined and implemented12. Some of the valid rules are:

The method signature can be changed: This is true. The sub constructor can have a different method signature than the super constructor, which means that it can have different input parameters, output parameters, or exceptions. However,

the sub constructor must still call the super constructor with appropriate actual parameters that match its interface12.

The constructor of super must be called before using any components of your own instance: This is true. The sub constructor must ensure that the super constructor is called explicitly using super->constructor before accessing any instance

components of its own class, such as attributes or methods. This is because the super constructor initializes the inherited components of the subclass and sets the self-reference me-> to the current instance12.

You cannot do any of the following:

Import parameters can only be evaluated after calling the constructor of super:

This is false. The sub constructor can evaluate its own import parameters before calling the constructor of super, as long as it does not access any instance components of its own class. For example, the sub constructor can use its import

parameters to calculate some values or check some conditions that are needed for calling the super constructor12.

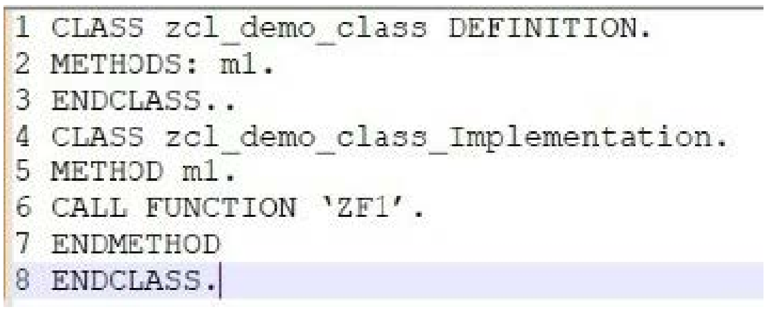
Events of your own instance cannot be raised before the registration of a handler in super: This is false. The sub constructor can raise events of its own instance before calling the constructor of super, as long as it does not access any instance components of its own class. For example, the sub constructor can raise an event to notify the consumers of the subclass about some status or error that occurred during the initialization of the subclass12.

References: 1: Inheritance and Constructors - ABAP Keyword Documentation - SAP Online Help 2: Using Static and Instance constructor methods | SAP Blogs

---

QUESTION 4

Given the following code in an SAP S/4HANA Cloud private edition tenant:



```
1 CLASS zcl_demo_class DEFINITION.
2 METHODS: m1.
3 ENDCLASS..
4 CLASS zcl_demo_class_Implementation.
5 METHOD m1.
6 CALL FUNCTION 'ZF1'.
7 ENDMETHOD
8 ENDCLASS.
```

The class zcl_demo_class is in a software component with the language version set to "ABAP Cloud". The function module ZF1\\' is in a different software component with the language version set to "Standard ABAP". Both the class and function module are customer created.

Regarding line #6, which of the following are valid statements? Note: There are 2 correct answers to this question.

A. ZF1\\' can be called only if it is released for cloud development.

B. \\'ZF1\\' can be called if a wrapper is created for it and the wrapper itself is released for cloud development.

C. "ZF1" can be called whether it is released or not for cloud development

D. ZF1" can be called if a wrapper is created for it but the wrapper itself is not released for cloud development.

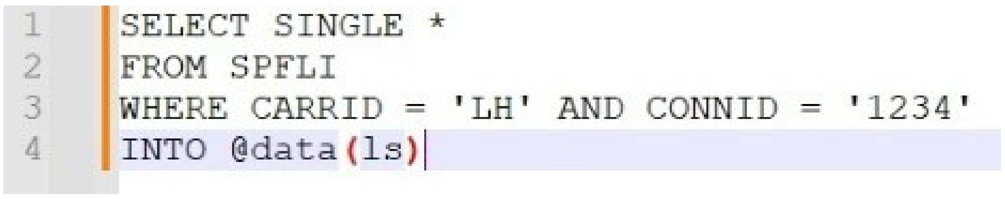
Correct Answer: AB

The ABAP Cloud Development Model requires that only public SAP APIs and extension points are used to access SAP functionality and data. These APIs and extension points are released by SAP and documented in the SAP API BusinessHub1. Customer-created function modules are not part of the public SAP APIs and are not released for cloud development. Therefore, calling a function module directly from an ABAP Cloud class is not allowed and will result in a syntax error. However, there are two possible ways to call a function module indirectly from an ABAP Cloud class: Create a wrapper class or interface for the function module and release it for cloud development. A wrapper is a class or interface that encapsulates the function module and exposes its functionality through public methods or attributes. The wrapper must be created in a software component with the language version set to "Standard ABAP" and must be marked as released for cloud development using the annotation @EndUserText.label. The wrapper can then be called from an ABAP Cloud class using the public methods or attributes2. Use the ABAP Cloud Connector to call the function module as a remote function call (RFC) from an ABAP Cloud class. The ABAP Cloud Connector is a service that enables the secure and reliable communication between SAP BTP, ABAP environment and on-premise systems. The function module must be exposed as an RFC-enabled function module in the on-premise system and must be registered in the ABAP Cloud Connector. The ABAP Cloud class can then use the class cl_rfc_destination_service to get the destination name and the class cl_abap_system to create a proxy object for the function module. The proxy object can then be used to call the function module3. References: 1: SAP API Business Hub 2: Creating an ABAP Cloud Project | SAP Help Portal 3: Calling Remote Function Modules | SAP Help Portal

---

QUESTION 5

You are given the following information:



```
SELECT SINGLE *
FROM SPFLI
WHERE CARRID = 'LH' AND CONNID = '1234'
INTO @data(ls)
```

1.

The data source "spfli" on line #2 is an SAP HANA database table

2.

"spfli" will be a large table with over one million rows.

3.

This program is the only one in the system that accesses the table.

4.

This program will run rarely.

Based on this information, which of the following general settings should you set for the spfli database table? Note: There are 2 correct answers to this question.

A. "Storage Type" to "Column Store"

B. "Load Unit to "Column Loadable"

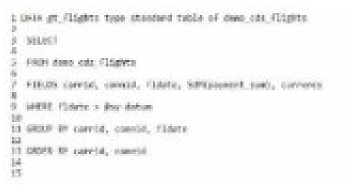
C. "Storage Type" to "Row Store"

D. "Load Unit\\' to \\'Page Loadable"

Correct Answer: CD

Based on the given information, the spfli database table should have the following general settings: "Storage Type" to "Row Store": This setting determines how the data is stored in the SAP HANA database. Row store is suitable for tables that are accessed by primary key or by a small number of columns. Column store is suitable for tables that are accessed by a large number of columns or by complex analytical queries. Since the spfli table is a large table with over one million rows, and this program is the only one in the system that accesses the table, it is likely that the program will use primary key access or simple queries to access the table. Therefore, row store is a better choice than column store for this table12. "Load Unit" to "Page Loadable": This setting determines how the data is loaded into the memory when the table is accessed. Page loadable means that the data is loaded in pages of 16 KB each, and only the pages that are needed are loaded. Column loadable means that the data is loaded in columns, and only the columns that are needed are loaded. Since the spfli table is a row store table, and this program will run rarely, it is more efficient to use page loadable than column loadable for this table. Page loadable will reduce the memory consumption and the loading time of the table13. References: 1: Table Types in SAP HANA | SAP Help Portal 2: [Row Store vs Column Store in SAP HANA | SAP Blogs] 3: [Load Unit | SAP Help Portal]

QUESTION 6



```
1 DATA gt_flights type standard table of demo_cds_flights
2
3  SELECT
4
5  FROM demo_cds_flights
6
7  FIELDS carrid, connid, fldate, SUM(payment_sum), currency
8
9  WHERE fldate > @sy-datum
10
11 GROUP BY carrid, connid, fldate
12
13 ORDER BY carrid, connid
14
15
```

(Sorry, we do not have a more clear image. If we have a better resource for the image, we will update this one immediately.)

To adhere to the most recent ABAP SQL syntax conventions from SAP, on which line must you insert the "INTO TABLE @gt flights" clause to complete the SQL statement?

A. #15

B. #4

C. #6

D. #8

Correct Answer: B

To adhere to the most recent ABAP SQL syntax conventions from SAP, you must insert the "INTO TABLE @gt flights" clause on line #4 to complete the SQL statement. This is because the INTO or APPENDING clause should be specified immediately after the SELECT clause, according to the ABAP SQL syntax conventions1. The INTO or APPENDING clause defines the data object to which the results set of the SELECT statement is assigned. The data object can be an internal table, a work area, or an inline declaration. In this case, the data object is an internal table named gt_flights, which is created using the inline declaration operator @DATA. The inline declaration operator allows you to declare and create a data object in the same statement where it is used, without the need for a separate DATA statement2. The other lines are not suitable for inserting the "INTO TABLE @gt flights" clause, as they would violate the ABAP SQL syntax conventions or cause syntax errors. These lines are: #6: This line is not suitable for inserting the "INTO TABLE @gt flights" clause, as it would cause a syntax error. This is because the FROM clause must be specified before the INTO or APPENDING clause, according to the ABAP SQL syntax conventions1. The FROM clause defines the data sources from which the data is read, such as database tables, CDS view entities, or CDS DDIC-based views. In this case, the data source is the database table flights. #8: This line is not suitable for inserting the "INTO TABLE @gt flights" clause, as it would cause a syntax error. This is because the ORDER BY clause must be specified after the INTO or APPENDING clause, according to the ABAP SQL syntax conventions1. The ORDER BY clause defines the sort order of the results set of the SELECT statement. In this case, the results set is sorted by the fields carrid, connid, and fltime. #15: This line is not suitable for inserting the "INTO TABLE @gt flights" clause, as it would violate the ABAP SQL syntax conventions. This is because the INTO or APPENDING clause should be specified as close as possible to the SELECT clause, according to the ABAP SQL syntax conventions1. The INTO or APPENDING clause should not be separated from the SELECT clause by other clauses, such as the WHERE clause, the GROUP BY clause, the HAVING clause, the UNION clause, or the ORDER BY clause. This is to improve the readability and maintainability of the ABAP SQL statement. References: SELECT - ABAP Keyword Documentation, Inline Declarations - ABAP Keyword Documentation

QUESTION 7

Which of the following string functions are predicate functions? Note: There are 2 correct answers to this question.

A. find_any_not_of()

B. contains_any_of()

C. count_any_of()

D. matchesQ

Correct Answer: BD

String functions are expressions that can be used to manipulate character-like data in ABAP. String functions can be either predicate functions or non-predicate functions. Predicate functions are string functions that return a truth value (true or false) for a condition of the argument text. Non-predicate functions are string functions that return a character-like result for an operation on the argument text1. The following string functions are predicate functions:

B. contains_any_of(): This function returns true if the argument text contains at least one of the characters specified in the character set. For example, the following expression returns true, because the text `ABAP\\' contains at least one of the

characters `A\\', `B\\', or `C\\':

contains_any_of(val = `ABAP\\' set = `ABC\\').

D. matches(): This function returns true if the argument text matches the pattern specified in the regular expression. For example, the following expression returns true, because the text `ABAP\\' matches the pattern that consists of four

uppercase letters:

matches(val = `ABAP\\' regex = `[A-Z]{4}\\').

The following string functions are not predicate functions, because they return a character- like result, not a truth value:

A. find_any_not_of(): This function returns the position of the first character in the argument text that is not contained in the character set. If no such character is found, the function returns 0. For example, the following expression returns 3,

because the third character of the text `ABAP\\' is not contained in the character set `ABC\\':

find_any_not_of(val = `ABAP\\' set = `ABC\\').

C. count_any_of(): This function returns the number of characters in the argument text that are contained in the character set. For example, the following expression returns 2, because there are two characters in the text `ABAP\\' that are

contained in the character set `ABC\\':

count_any_of(val = `ABAP\\' set = `ABC\\').

References: 1: String Functions - ABAP Keyword Documentation

QUESTION 8

Which of the following are valid sort operations for internal tables? Note: There are 3 correct answers to this question.

A. Sort a standard table using SORT itab ASCENDING. Sort a sorted table using

B. SORT itab BY fieldl ASCENDING field2 DESCENDING. Sort a standard table using

C. SORT itab BY field1 field2. Sort a standard table using

D. SORT itab. Sort a sorted table using

E. SORT itab DESCENDING.

Correct Answer: ACD

QUESTION 9

Which internal table type allows unique and non-unique keys?

A. Sorted

B. Hashed

C. Standard

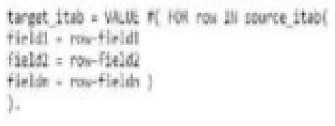
Correct Answer: C

The internal table type that allows both unique and non-unique keys is the standard table. A standard table has an internal linear index that can be used to access the table entries. The key of a standard table is always non-unique, which means that the table can contain duplicate entries. However, the system does not check the uniqueness of the key when inserting new entries, so the programmer can ensure that the key is unique by using appropriate logic. A standard table can be accessed either by using the table index or the key, but the response time for key access is proportional to the table size. The other two internal table types, sorted and hashed, do not allow non-unique keys. A sorted table is filled in sorted order according to the defined table key, which must be unique. A sorted table can be accessed either by using the table index or the key, but the response time for key access is logarithmically proportional to the table size. A hashed table can only be accessed by using a unique key, which must be specified when declaring the table. A hashed table has no index, and the response time for key access is constant, regardless of the table size. References: Internal Tables - ABAP Keyword Documentation, SAP ABAP: Types Of Internal Table Declaration - dan852.com

---

QUESTION 10

Exhibit:



```
target_itab = VALUE #( FOR row IN source_itab(
field1 = row-field1
field2 = row-field2
fieldn = row-fieldn )
).
```

(Sorry, we do not have a more clear image. If we have a better resource for the image, we will update this one immediately.)

Which of the following statements are correct? Note: There are 2 correct answers to this question.

A. FOR defines a loop that runs over the content of source_itab

B. source_itab is only visible within the loop.

C. row is a predefined name and cannot be chosen arbitrarily.

D. row is only visible within the loop.

Correct Answer: AD

The code snippet in the image is an example of using the FOR statement to create an internal table with a constructor expression. The FOR statement introduces an iteration expression that runs over the content of source_itab and assigns

each row to the variable row. The variable row is then used to populate the fields of target_itab12. Some of the correct statements about the code snippet are:

FOR defines a loop that runs over the content of source_itab: This is true. The FOR statement iterates over the rows of source_itab and assigns each row to the variable row. The iteration expression can also specify a range or a condition for

the loop12.

row is only visible within the loop: This is true. The variable row is a local variable that is only visible within the scope of the iteration expression. It cannot be accessed outside the loop12.

You cannot do any of the following:

source_itab is only visible within the loop: This is false. The variable source_itab is not a local variable that is defined by the FOR statement. It is an existing internal table that is used as the data source for the iteration expression. It can be

accessed outside the loop12.

row is a predefined name and cannot be chosen arbitrarily: This is false. The variable row is not a predefined name that is reserved by the FOR statement. It is a user-defined name that can be chosen arbitrarily. However, it must not conflict

with any existing names in the program12.

References: 1: FOR - Iteration Expressions - ABAP Keyword Documentation - SAP Online Help 2: ABAP 7.4 Syntax - FOR Loop iteration | SAP Community

---

QUESTION 11

What is the purpose of a foreign key relationship between two tables in the ABAP Dictionary?

A. To document the relationship between the two tables

B. To ensure the integrity of data in the corresponding database tables

C. To create a corresponding foreign key relationship in the database

Correct Answer: B

The purpose of a foreign key relationship between two tables in the ABAP Dictionary is to ensure the integrity of data in the corresponding database tables. A foreign key relationship defines a logical link between a foreign key table and a check table, where the foreign key fields of the former are assigned to the primary key fields of the latter. This means that the values entered in the foreign key fields must exist in the check table, otherwise the system will reject the entry. This way, the foreign key relationship prevents the insertion of invalid or inconsistent data in the database tables. A foreign key relationship also serves to document the relationship between the two tables in the ABAP Dictionary, but this is not its primary purpose. A foreign key relationship does not necessarily create a corresponding foreign key relationship in thedatabase, as this depends on the database system and the settings of the ABAP Dictionary. Some database systems do not support foreign keys at all, while others require additional steps to activate them. Therefore, the foreign key relationship in the ABAP Dictionary is mainly a logical concept that is enforced by the ABAP runtime environment. References: Foreign Keys (SAP Library - ABAP Dictionary), Foreign Keys (SAP Library - BC - ABAP Dictionary) https://help.sap.com/doc/saphelp_snc70/7.0/enUS/cf/21ea77446011d189700000e8322d00/content.htm

---

QUESTION 12

What are advantages of using a field symbol for internal table row access? Note: There are 2 answers to this question.

A. The field symbol can be reused for other programs.

B. A MODIFY statement to write changed contents back to the table is not required.

C. The row content is copied to the field symbol instead to a work area

D. Using a field symbol is faster than using a work area.

Correct Answer: BD

A field symbol is a pointer that allows direct access to a row of an internal table without copying it to a work area. Using a field symbol for internal table row access has some advantages over using a work area, such as12:

A MODIFY statement to write changed contents back to the table is not required:

This is true. When you use a work area, you have to copy the row content from the internal table to the work area, modify it, and then copy it back to the internal table using the MODIFY statement. This can be costly in terms of performance

and memory consumption. When you use a field symbol, you can modify the row content directly in the internal table without any copying. Therefore, you do not need the MODIFY statement12.

Using a field symbol is faster than using a work area: This is true. As explained above, using a field symbol avoids the overhead of copying data between the internal table and the work area. This can improve the performance of the loop

considerably, especially for large internal tables. According to some benchmarks, using a field symbol can save 25?0% of the runtime compared to using a work area12.

You cannot do any of the following:

The field symbol can be reused for other programs: This is false. A field symbol is a local variable that is only visible within the scope of its declaration. It cannot be reused for other programs unless it is declared globally or passed as a

parameter. Moreover, a field symbol must have the same type as the line type of the internal table that it accesses. Therefore, it cannot be used for any internal table with a different line type12.

The row content is copied to the field symbol instead to a work area: This is false. As explained above, using a field symbol does not copy the row content to the field symbol. Instead, the field symbol points to the memory address ofthe row in

the internal table and allows direct access to it. Therefore, there is no copying involved when using a field symbol12.

References: 1: Using Field Symbols to Process Internal Tables - SAP Learning 2: Access to Internal Tables - ABAP Keyword Documentation - SAP Online Help