Free Questions for C_ABAPD_2309 by certsdeals

Shared by Patton on 29-01-2024

Question 1

Question Type: **MultipleChoice**

What RESTful Application Programming feature is used to ensure the uniqueness of a semantic key?

Options:

A- Validation

B- Action

C- Determination

Answer:

C

Explanation:

The RESTful Application Programming feature that is used to ensure the uniqueness of a semantic key is determination. A determination is a type of behavior implementation that defines a logic that is executed automatically when certain events occur, such as create, update, delete, or activate. A determination can be used to calculate or derive values for certain fields, such as semantic keys, based on other fields or external sources. A determination can also be used to check the uniqueness of a semantic key by comparing it with the existing values in the database or the transaction buffer. A determination can use the ABAP SQL or the EML syntax to access and manipulate data. A determination can be defined using the DETERMINE action clause in the behavior definition of a CDS view entity or a projection view.A determination can also be annotated with the @ObjectModel.determination annotation to specify the event, the timing, and the scope of the determination12

The other RESTful Application Programming features are not used to ensure the uniqueness of a semantic key, but have different purposes and effects. These features are:

Validation: A validation is a type of behavior implementation that defines a logic that is executed automatically when certain events occur, such as create, update, delete, or activate. A validation can be used to check the consistency and correctness of the data, such as mandatory fields, data types, value ranges, or business rules. A validation can use the ABAP SQL or the EML syntax to access and manipulate data. A validation can be defined using the VALIDATE action clause in the behavior definition of a CDS view entity or a projection view.A validation can also be annotated with the @ObjectModel.validation annotation to specify the event, the timing, and the scope of the validation12

Action: An action is a type of behavior implementation that defines a logic that is executed explicitly by the user or the application. An action can be used to perform a specific business operation, such as creating, updating, deleting, or activating an entity instance, or triggering a workflow or a notification. An action can use the ABAP SQL or the EML syntax to access and manipulate data. An action can be defined using the ACTION clause in the behavior definition of a CDS view entity or a projection view.An action can also be annotated with the @ObjectModel.action annotation to specify the name, the description, the parameters, and the visibility of the action12

Question 2

Question Type: MultipleChoice

In the assignment, data (gv_result) = 1/8. what will be the data type of gv_result?

Options:

A- OTYPE I

B- TYPE DEFLOAT 16

C- TYPE P DECIMALS 3

D- TYPE P DECIMALS 2

Answer:

B

Explanation:

The data type of gv_result in the assignment data (gv_result) = 1/8 will be TYPE DECFLOAT 16.This is because the assignment operator (=) in ABAP performs an implicit type conversion from the source type to the target type, according to the following rules12:

If the target type is specified explicitly, the source value is converted to the target type.

If the target type is not specified explicitly, the source type is used as the target type, unless the source type is a literal or an expression, in which case the target type is determined by the following priority order: DECFLOAT34, DECFLOAT16, P, F, I, C, N, X, STRING, XSTRING.

In this case, the target type is not specified explicitly, and the source type is an expression (1/8).Therefore, the target type is determined by the priority order, and the first matching type is DECFLOAT16, which is a decimal floating point type with 16 digits of precision12.

Question 3

Question Type: MultipleChoice

You want to define the following CDS view entity with an input parameter:

Define view entity Z_CONVERT With parameters currency : ???

Which of the following can you use to replace "???? Note: There are 2 correct answers to this

question.

Options:

A- built-in ABAP type

B- A built-in ABAP Dictionary type

C- A data element

D- A component of an ABAP Dictionary structure

Answer:

A, C

Explanation:

The possible replacements for "???" in the CDS view entity definition with an input parameter are A. built-in ABAP type and C. A data element. These are the valid types that can be used to specify the data type of an input parameter in a CDS view entity. A built-in ABAP type is a predefined elementary type in the ABAP language, such as abap.char, abap.numc, abap.dec, etc. A data element is a reusable semantic element in the ABAP Dictionary that defines the technical attributes and the meaning of a field12. For example:

The following code snippet defines a CDS view entity with an input parameter currency of type abap.cuky, which is a built-in ABAP type for currency key:

Define view entity Z_CONVERT With parameters currency : abap.cuky as select from ... { ... }

The following code snippet defines a CDS view entity with an input parameter currency of type waers, which is a data element for currency key:

Define view entity Z_CONVERT With parameters currency : waers as select from ... { ... }

You cannot do any of the following:

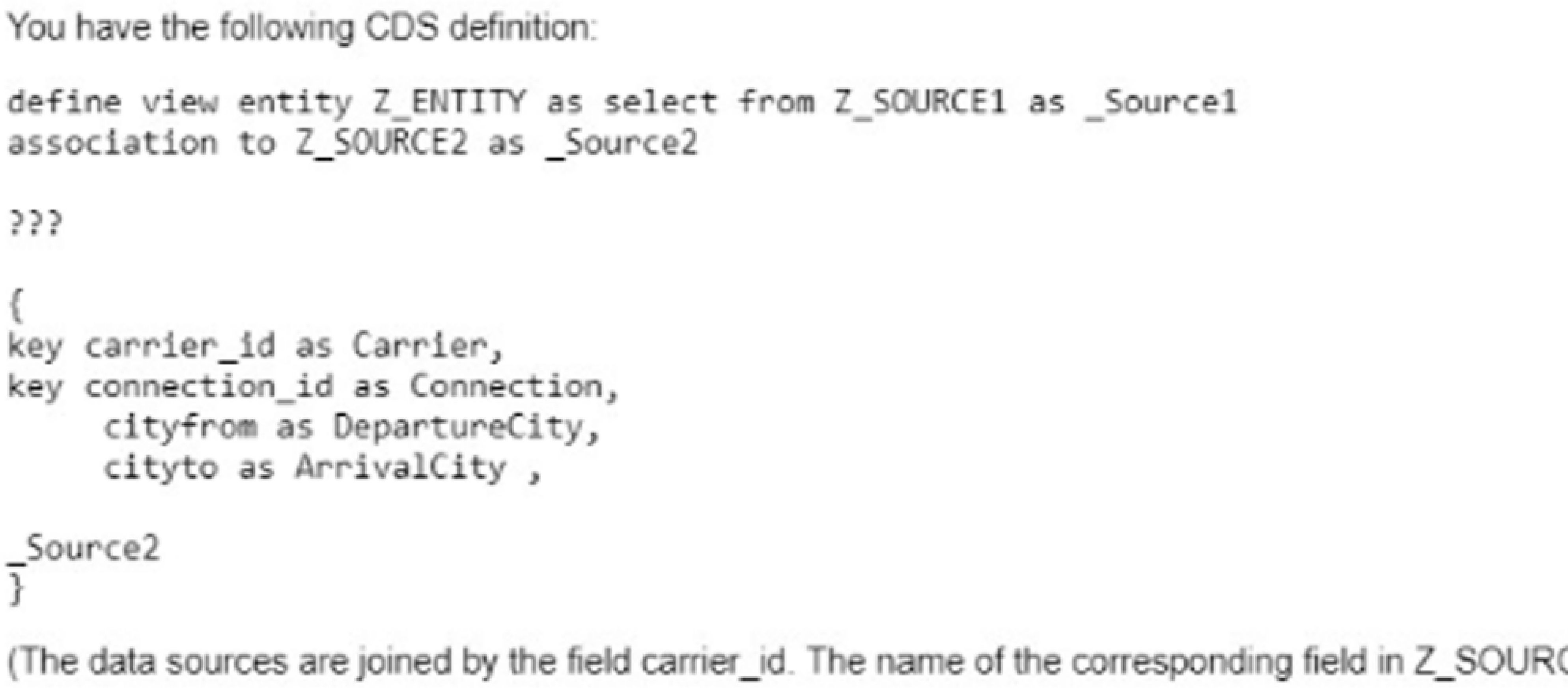
B) A built-in ABAP Dictionary type: This is not a valid type for an input parameter in a CDS view entity. A built-in ABAP Dictionary type is a predefined elementary type in the ABAP Dictionary, such as CHAR, NUMC, DEC, etc. However, these types cannot be used directly in a CDS view entity definition. Instead, they have to be prefixed with abap. to form a built-in ABAP type, as explained above12.

D) A component of an ABAP Dictionary structure: This is not a valid type for an input parameter in a CDS view entity. A component of an ABAP Dictionary structure is a field that belongs to a structure type, which is a complex type that consists of multiple fields. However, an input parameter in a CDS view entity can only be typed with an elementary type, which is a simple type that has no internal structure12.

Question 4

Question Type: **MultipleChoice**

Refer to the Exhibit.



You have the following CDS definition:

```
define view entity Z_ENTITY as select from Z_SOURCE1 as _Source1
association to Z_SOURCE2 as _Source2

???

{
key carrier_id as Carrier,
key connection_id as Connection,
     cityfrom as DepartureCity,
     cityto as ArrivalCity ,

_Source2
}
```

(The data sources are joined by the field carrier_id. The name of the corresponding field in Z_SOUR

Which of the following ON conditions must you insert in place of "???"?

Options:

A- ON Z_Sourcel.camer_id = 7_Source2 carrier_id

B- ON Sprojection Camer=Source2 carrier_id

C- ON Sprojection. Carrier Source2.carrier

D- ON Sprojection.carrier_id=Z_Source2.carrier_id

Answer:

D

Explanation:

The correct ON condition that must be inserted in place of "???" is:

ON Sprojection.carrier_id=Z_Source2.carrier_id

This ON condition specifies the join condition between the CDS view Sprojection and the database table Z_Source2. The join condition is based on the field carrier_id, which is the primary key of both the CDS view and the database table.The ON condition ensures that only the records that have the same value for the carrier_id field are joined together1.

The other options are not valid ON conditions, because:

A) ON Z_Sourcel.camer_id = 7_Source2 carrier_id is not valid because Z_Sourcel and 7_Source2 are not valid data sources in the given code. There is no CDS view or database table named Z_Sourcel or 7_Source2. The correct names are Z_Source1 and Z_Source2. Moreover, the field camer_id is not a valid field in the given code. There is no field named camer_id in any of the data sources. The correct name is carrier_id.

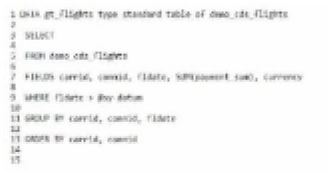
B) ON Sprojection Camer=Source2 carrier_id is not valid because Sprojection and Source2 are not valid data sources in the given code. There is no CDS view or database table named Sprojection or Source2. The correct names are Sprojection and Z_Source2. Moreover, the field Camer is not a valid field in the given code. There is no field named Camer in any of the data sources. The correct name is carrier_id.Furthermore, the ON condition is missing the dot (.) operator between the data source name and the field name, which is required to access the fields of the data source1.

C) ON Sprojection. Carrier Source2.carrier is not valid because Carrier and carrier are not valid fields in the given code. There is no field named Carrier or carrier in any of the data sources. The correct name is carrier_id.Moreover, the ON condition is missing the dot (.) operator between the data source name and the field name, which is required to access the fields of the data source1.

Question 5

Question Type: MultipleChoice

Refer to the Exhibit.



```
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
```

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?

Options:

A- #15

B- #4

C- #6

D- #8

Answer:

B

Explanation:

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.

Question 6

Question Type: MultipleChoice

For the assignment, gv_target = gv_source.

which of the following data declarations will always work without truncation or rounding? Note: There

are 2 correct answers to this question.

Options:

A- DATA gv_source TYPE string, to DATA gv_target TYPE c.

B- DATA gv_source TYPE c. to DATA gv_target TYPE string.

C- DATA gv_source TYPE d. to DATA gv_target TYPE string.

D- DATA gv_source TYPE p LENGTH 8 DECIMALS 3. to DATA gv_target TYPE p LENGTH 16 DECIMALS 2.

Answer:

B, C

Explanation:

The data declarations that will always work without truncation or rounding for the assignment gv_target = gv_source are B and C. This is because the target data type string is a variable-length character type that can hold any character string, including those of data types c (fixed-length character) and d (date).The assignment of a character or date value to a string variable will not cause any loss of information or precision, as the string variable will adjust its length to match the source value12.

You cannot do any of the following:

A) DATA gv_source TYPE string, to DATA gv_target TYPE c.: This data declaration may cause truncation for the assignment gv_target = gv_source. This is because the target data type c is a fixed-length character type that has a predefined length.If the source value of type string is longer than the target length of type c, the source value will be truncated on the right to fit the target length12.

D) DATA gv_source TYPE p LENGTH 8 DECIMALS 3. to DATA gv_target TYPE p LENGTH 16 DECIMALS 2.: This data declaration may cause rounding for the assignment gv_target = gv_source. This is because the target data type p is a packed decimal type that has a predefined length and number of decimal places.If the source value of type p has more decimal places than the target type p, the source value will be rounded to the target number of decimal places12.

Question 7

Question Type: **MultipleChoice**

In RESTful Application Programming, which EML statement retrieves an object?

Options:

A- Find entity

B- Select entity

C- Get entity

D- Read entity

Answer:

C

Explanation:

In RESTful Application Programming, the EML statement that retrieves an object is GET entity. The GET entity statement is used to read data of an entity instance from the database or the transaction buffer. The GET entity statement can specify the entity name, the entity key, and the entity elements to be retrieved. The GET entity statement can also use the IN LOCAL MODE addition to bypass the access control, authorization control, and feature control checks. The GET entity statement returns a single entity instance or raises an exception if no instance is found or multiple instances match the key.

The other EML statements are not used to retrieve an object, but have different purposes and effects. These statements are:

FIND entity: This statement is used to search for entity instances that match a given condition. The FIND entity statement can specify the entity name, the entity elements to be returned, and the condition to be applied. The FIND entity statement can also use the IN LOCAL MODE addition to bypass the access control, authorization control, and feature control checks. The FIND entity statement returns a table of entity instances or an empty table if no instances match the condition.

SELECT entity: This statement is used to query data of entity instances from the database or the transaction buffer. The SELECT entity statement can specify the entity name, the entity elements to be returned, and the filter, order, and aggregation options to be applied. The SELECT entity statement can also use the IN LOCAL MODE addition to bypass the access control, authorization control, and feature control checks. The SELECT entity statement returns a table of entity instances or an empty table if no instances match the query.

READ entity: This statement is not a valid EML statement, but an ABAP statement. The READ statement is used to access a single row of an internal table using the table index or the table key. The READ statement can also use the TRANSPORTING addition to specify which fields should be returned, and the INTO addition to specify the target variable. The READ statement returns a single row of the internal table or raises an exception if no row is found or multiple rows match the key.

Question 8

Question Type: MultipleChoice

Which of the following is a generic internal table type?

Options:

A- SORTED TABLE

B- INDEX TABLE

C- STANDARD TABLE

D- HASHED TABLE

Answer:

B

Explanation:

A generic internal table type is a table type that does not define all the attributes of an internal table in the ABAP Dictionary; it leaves some of these attributes undefined.A table type is generic in the following cases1:

You have selected Index Table or Not Specified as the access type.

You have not specified a table key or specified an incomplete table key.

You have specified a generic secondary table key.

A generic table type can be used only for typing formal parameters or field symbols.A generic table type cannot be used for defining data objects or constants2.

Therefore, the correct answer is B. INDEX TABLE, which is a generic table type that does not specify the access type or the table key. The other options are not generic table types, because:

A)SORTED TABLE is a table type that specifies the access type as sorted and the table key as a unique or non-unique primary key3.

C)STANDARD TABLE is a table type that specifies the access type as standard and the table key as a non-unique standard key that consists of all the fields of the table row in the order in which they are defined4.

D)HASHED TABLE is a table type that specifies the access type as hashed and the table key as a unique primary key5.