SAP.C_ABAPD_2309.v2025-08-14.q36

Exam Code:	C_ABAPD_2309
Exam Name:	SAP Certified Associate - Back-End Developer - ABAP Cloud
Certification Provider:	SAP
Free Question Number:	36
Version:	v2025-08-14
# of views:	112
# of Questions views:	360
https://www.freepdfdumps.com/SAP.C_ABAPD_2309.v2025-08-14.q36.html	

NEW QUESTION: 1
Exhibit:
DEFINE TABLE demo_table {
KEY field1 : REFERENCE TO abap.cint(3);
KEY field2 : abap.char(133);
@Semantics.quantity.unitOfMeasure : 'demo_table.field4'
field3 : abap.quan(2);
field4 : abap.unit(2);
}
Which field is defined incorrectly?
A. field2
B. field4
C. field1
D. field3
Answer: C (LEAVE A REPLY)
* Field 1 (field1)
* The syntax REFERENCE TO abap.cint(3) is incorrect for defining a database table field. Database table fields must be based on valid HANA-compatible data types or ABAP dictionary data types.
* REFERENCE TO is used for object references in ABAP classes and is not valid in this context.
* Therefore, field1 is defined incorrectly, making option C correct.
* Field 2 (field2)
* The data type abap.char(133) is valid for defining a character-based field in a table.
* This field is correctly defined.
* Field 3 (field3)
* The data type abap.quan(2) is valid for defining a quantity field. The @Semantics.quantity. unitOfMeasure annotation correctly links this field to field4, which is a unit of measure field.

* This field is correctly defined.
* Field 4 (field4)
* The data type abap.unit(2) is valid for defining a unit of measure field.
* This field is correctly defined.
References:
* SAP ABAP Documentation: ABAP Data Types for SAP HANA Tables
* SAP Training for Back-End Developer - ABAP Cloud

NEW QUESTION: 2
In what order are objects created to generate a RESTful Application Programming application?
A. Database table 1
B. Service binding Projection view 4
C. Service definition 3
D. Data model view 2
E. D A B C
F. B D C A
G. A D C B
H. C B A B
Answer: C (LEAVE A REPLY)
The order in which objects are created to generate a RESTful Application Programming application is A, D, C, B. This means that the following steps are followed:
* First, a database table is created to store the data for the application. A database table is a CDS DDIC- based view that defines a join or union of database tables. A database table has an SQL view attached and can be accessed by Open SQL or native SQL.
* Second, a data model view is created to define a data model based on the database table or other CDS view entities. A data model view is a CDS view entity that can have associations, aggregations, filters, parameters, and annotations. A data model view can also define the behavior definition and implementation for the business object.
* Third, a service definition is created to define the service interface for the application. A service definition is a CDS view entity that defines a projection on a data model view or another service definition. A service definition can also define service metadata, such as service name, version, description, and annotations.
* Fourth, a service binding is created to define the service binding for the application. A service binding is a CDS view entity that defines a projection on a service definition. A service binding can also define the service protocol, such as OData V2, OData V4, or REST, and the service URL.
References: CDS Data Model Views - ABAP Keyword Documentation, CDS Service Definitions - ABAP Keyword Documentation, CDS Service Bindings - ABAP Keyword Documentation, CDS Projection Views - ABAP Keyword Documentation

NEW QUESTION: 3

Exhibit:

The class zcl_demo_class is in a software component with the language version set to "Standard ABAP". The function module "ZF11 is in a software component with the language version set to "ABAP Cloud". Both the class and function module are customer created. Regarding line #6, which of the following is a valid statement?

A. 'ZF1' can be called whether it has been released or not for cloud development.

B. 'ZF1' can be called via a wrapper that itself has been released for cloud development.

C. 'ZF1' can be called via a wrapper that itself has not been released for cloud development.

D. 'ZF1' must be released for cloud development to be called.

Answer: (SHOW ANSWER)

The function module ZF1 is in a software component with the language version set to "ABAP Cloud". This means that it follows the ABAP Cloud Development Model, which requires the usage of public SAP APIs and extension points to access SAP functionality and data. These APIs and extension points are released by SAP and documented in the SAP API Business Hub1.

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 a class with the language version set to "Standard ABAP" is not allowed and will result in a syntax error. However, there is a possible way to call a function module indirectly from a class with the language version set to "Standard ABAP":

* 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 "ABAP Cloud" and must be marked as released for cloud development using the annotation @EndUserText.label. The wrapper can then be called from a class with the language version set to "Standard ABAP" using the public methods or attributes2.

For example, the following code snippet shows how to create a wrapper class for the function module ZF1 and call it from the class zcl_demo_class:

@EndUserText.label: 'Wrapper for ZF1' CLASS zcl_wrapper_zf1 DEFINITION PUBLIC FINAL CREATE PUBLIC. PUBLIC SECTION. CLASS-METHODS: call_zf1 IMPORTING iv_a TYPE i iv_b TYPE i EXPORTING ev_result TYPE i. ENDCLASS.

CLASS zcl_wrapper_zf1 IMPLEMENTATION. METHOD call_zf1. CALL FUNCTION 'ZF1' EXPORTING a = iv_a b = iv_b IMPORTING result = ev_result. ENDMETHOD. ENDCLASS.
CLASS zcl_demo_class DEFINITION. METHODS: m1. ENDCLASS.
CLASS zcl_demo_class IMPLEMENTATION. METHOD m1. DATA(lv_result) = zcl_wrapper_zf1=>call_zf1(iv_a = 2 iv_b = 3). WRITE: / lv_result. ENDMETHOD. ENDCLASS.
The output of this code is:
5
References: 1: SAP API Business Hub 2: Creating an ABAP Cloud Project | SAP Help Portal

NEW QUESTION: 4
You are designing the following select statement in ABAP Open SQL:

```
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?
A. #6
B. #15
C. #4
D. #8
Answer: (SHOW ANSWER)

NEW QUESTION: 5
Which of the following integration frameworks have been released for ABAP cloud development?
Note:
There are 3 correct answers to this question.
A. SOAP consumption
B. CDS Views
C. Business Add-ins (BAdIs)

D. Business Events
E. OData services
Answer: A,D,E (LEAVE A REPLY)
The following are the integration frameworks that have been released for ABAP cloud development:
* SOAP consumption: This framework allows you to consume SOAP web services from ABAP cloud applications. You can use the ABAP Development Tools in Eclipse to create a service consumption model based on a WSDL file or URL. The service consumption model generates the required ABAP artifacts, such as proxy classes, data types, and constants, to access the web service. You can then use the proxy classes to call the web service operations from your ABAP code1
* Business Events: This framework allows you to publish and subscribe to business events from ABAP cloud applications. Business events are messages that represent a change in the state of a business object or process. You can use the ABAP Development Tools in Eclipse to create a business event definition based on a CDS view entity or a projection view. The business event definition specifies the event key, the event payload, and the event metadata. You can then use the ABAP Messaging Channel (AMC) framework to publish and subscribe to business events using the AMC API2
* OData services: This framework allows you to expose and consume OData services from ABAP cloud applications. OData is a standardized protocol for creating and consuming RESTful APIs. You can use the ABAP RESTful Application Programming Model (RAP) to create OData services based on CDS view entities or projection views. The RAP framework generates the required OData metadata and runtime artifacts, such as service definitions, service bindings, and service implementations. You can then use the SAP Gateway framework to register and activate your OData services. You can also use the ABAP Development Tools in Eclipse to consume OData services from other sources using the service consumption model3 The other integration frameworks are not released for ABAP cloud development, as they are either not supported or not recommended for cloud scenarios. These frameworks are:
* CDS Views: CDS views are not an integration framework, but a data modeling framework. CDS views are used to define data models based on database tables or other CDS view entities. CDS views can have associations, aggregations, filters, parameters, and annotations. CDS views can also be used as the basis for other integration frameworks, such as OData services or business events4
* Business Add-ins (BAdIs): BAdIs are not supported for ABAP cloud development, as they are part of the classic ABAP enhancement framework. BAdIs are used to implement custom logic in predefined enhancement spots in the standard SAP code. BAdIs are not compatible with the cloud strategy and the clean core paradigm, as they modify the SAP code and can cause upgrade and maintenance issues. For ABAP cloud development, SAP recommends using the key user extensibility tools or the side-by-side extensibility approach instead of BAdIs.
References: Consuming SOAP Services - ABAP Keyword Documentation, Business Events - ABAP Keyword Documentation, OData Services - ABAP Keyword Documentation, CDS Data

Model Views - ABAP Keyword Documentation, [Business Add-Ins (BAdIs) - ABAP Keyword Documentation]

NEW QUESTION: 6
What RESTful Application Programming object contains only the fields required for a particular app?
A. Database view
B. Metadata extension
C. Projection View
D. Data model view
Answer: C (LEAVE A REPLY)
A projection view is a RESTful Application Programming object that contains only the fields required for a particular app. A projection view is a CDS view entity that defines a projection on an existing CDS view entity or CDS DDIC-based view. A projection view exposes a subset of the elements of the projected entity, which are relevant for a specific business service. A projection view can also define aliases, virtual elements, and annotations for the projected elements. A projection view is the top-most layer of a CDS data model and prepares data for a particular use case. A projection view can have different provider contracts depending on the type of service it supports, such as transactional query, analytical query, or transactional interface.
A database view is a CDS DDIC-based view that defines a join or union of database tables. A database view has an SQL view attached and can be accessed by Open SQL or native SQL. A database view can be used as a projected entity for a projection view, but it does not contain only the fields required for a particular app.
A metadata extension is a RESTful Application Programming object that defines additional annotations for a CDS view entity or a projection view. A metadata extension can be used to enhance the metadata of a CDS data model without changing the original definition. A metadata extension does not contain any fields, but only annotations.
A data model view is a CDS view entity that defines a data model based on database tables or other CDS view entities. A data model view can have associations, aggregations, filters, parameters, and annotations. A data model view can be used as a projected entity for a projection view, but it does not contain only the fields required for a particular app.
References: CDS Projection Views - ABAP Keyword Documentation, CDS Projection Views in ABAP CDS:
What's Your Flavor, Business Object Projection - ABAP Keyword Documentation

NEW QUESTION: 7
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

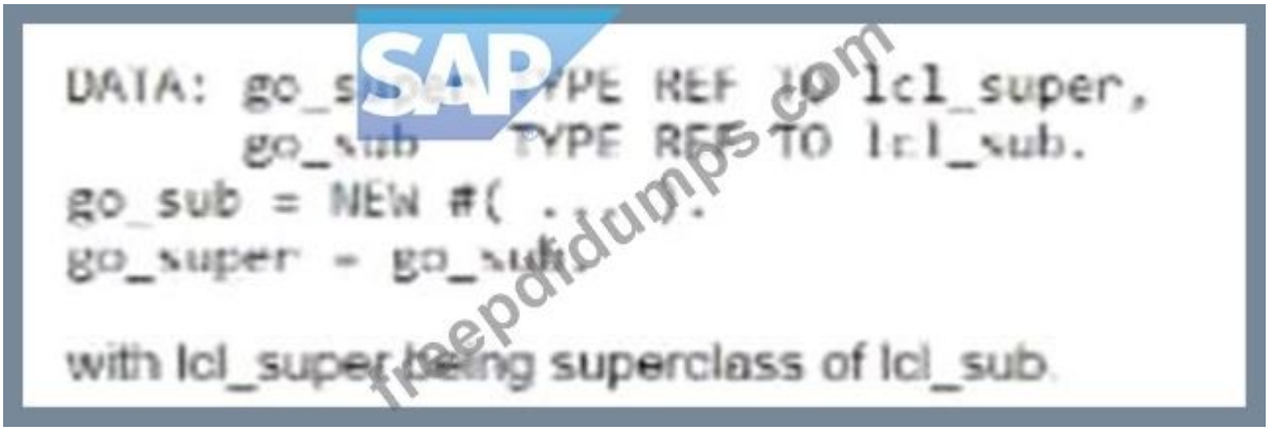
D. sy-tabix

Answer: (SHOW ANSWER)

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 each subsequent 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

NEW QUESTION: 8



When accessing the subclass instance through go_super, what can you do? Note: There are 2 correct answers to this question.

A. Access the inherited private components.

B. Access the inherited public components.

C. Call a subclass specific public method

D. Call inherited public redefined methods.

Answer: A,B (LEAVE A REPLY)

When accessing the subclass instance through go_super, you can do both of the following:

* Access the inherited private components: A subclass inherits all the private attributes and methods of its superclass, unless they are explicitly overridden by the subclass. Therefore, you can access the inherited private components of the superclass through go_super, as long as they are not hidden by other attributes or methods in the subclass12.
* Access the inherited public components: A subclass inherits all the public attributes and methods of its superclass, unless they are explicitly overridden by the subclass. Therefore, you can access the inherited public components of the superclass through go_super, as long as they are not hidden by other attributes or methods in the subclass12.

You cannot do any of the following:

* Call a subclass specific public method: A subclass does not have any public methods that are not inherited from its superclass. Therefore, you cannot call a subclass specific public method through go_super12.

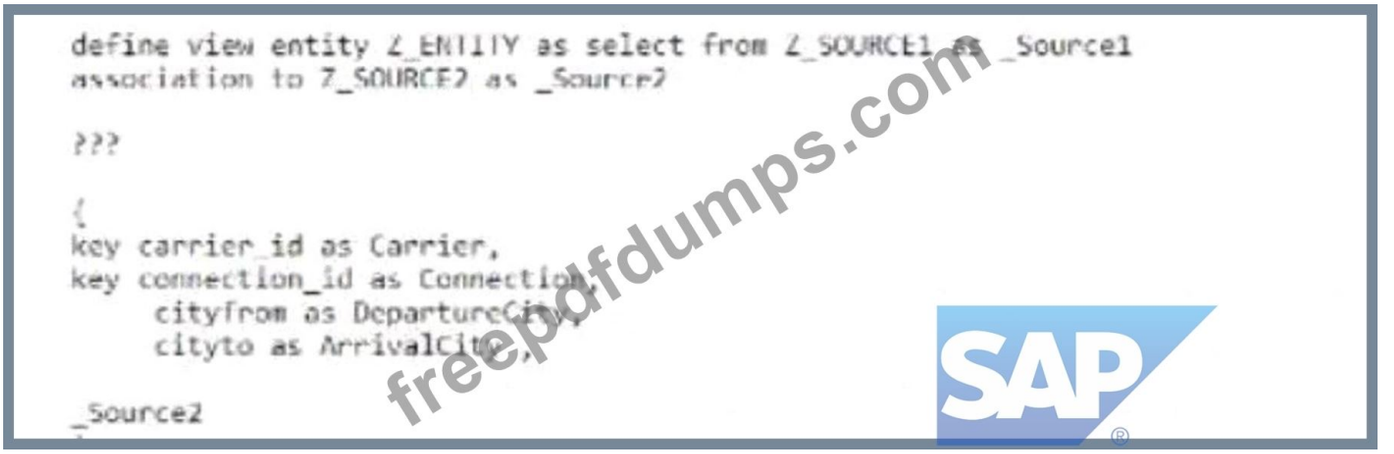
* Call inherited public redefined methods: A subclass does not have any public methods that are redefined from its superclass. Therefore, you cannot call inherited public redefined methods through go_super12.
References: 1: Object Oriented - ABAP Development - Support Wiki 2: Inheritance and Instantiation - ABAP Keyword Documentation

NEW QUESTION: 9
In an Access Control Object, which clauses are used? Note: There are 3 correct answers to this question.
A. Where (to specify the access conditions)
B. Crant (to identify the data source)
C. Return code (to assign the return code of the authority check)
D. Define role (to specify the role name)
E. Revoke (to remove access to the data source)
Answer: A,D,E (LEAVE A REPLY)
An Access Control Object (ACO) is a CDS annotation that defines the access control rules for a CDS view entity. An ACO consists of one or more clauses that specify the role name, the data source, the access conditions, and the return code of the authority check12. Some of the clauses that are used in an ACO are:
* Where (to specify the access conditions): This clause is used to define the logical expression that determines whether a user has access to the data source or not. The expression can use the fields of the data source, the parameters of the CDS view entity, or the predefined variables $user and $session. The expression can also use the functions check_authorization and check_role to perform additional authority checks12.
* Define role (to specify the role name): This clause is used to assign a name to the role that is defined by the ACO. The role name must be unique within the namespace of the CDS view entity and must not contain any special characters. The role name can be used to reference the ACO in other annotations, such as @AccessControl.authorizationCheck or @AccessControl.grant12.
* Revoke (to remove access to the data source): This clause is used to explicitly deny access to the data source for a user who meets the conditions of the where clause. The revoke clause overrides any grant clause that might grant access to the same user. The revoke clause can be used to implement the principle of least privilege or to enforce data segregation12.
You cannot do any of the following:
* Grant (to identify the data source): This is not a valid clause in an ACO. The grant clause is a separate annotation that is used to grant access to a CDS view entity or a data source for a user who has a specific role. The grant clause can reference an ACO by its role name to apply the access conditions defined by the ACO12.
* Return code (to assign the return code of the authority check): This is not a valid clause in an ACO. The return code of the authority check is a predefined variable that is set by the system after performing the access control check. The return code can be used in the where clause of the ACO to specify different access conditions based on the outcome of the check12.

References: 1: Access Control Objects - ABAP Keyword Documentation - SAP Online Help 2: Access Control in Core Data Services (CDS) | SAP Help Portal

NEW QUESTION: 10
In the assignment, data (gv_result) = 1/8. what will be the data type of gv_result?
A. OTYPE I
B. TYPE DEFLOAT 16
C. TYPE P DECIMALS 3
D. TYPE P DECIMALS 2
Answer: (SHOW ANSWER)
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.
References: 1: ABAP Assignment Rules - ABAP Keyword Documentation - SAP Online Help 2: ABAP Data Types - ABAP Keyword Documentation - SAP Online Help

NEW QUESTION: 11
You have the following CDS definition:



Which of the following ON conditions must you insert in place of "???"?
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 (LEAVE A REPLY)

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.
References: 1: ON Condition - ABAP Keyword Documentation

NEW QUESTION: 12
In RESTful Application Programming, a business object contains which parts? Note: There are 2 correct answers to this question.
A. CDS view
B. Behavior definition
C. Authentication rules
D. Process definition
Answer: A,B (LEAVE A REPLY)
In RESTful Application Programming, a business object contains two main parts: a CDS view and a behavior definition1.
* A. CDS view: A CDS view is a data definition that defines the structure and the data source of a business object. A CDS view can consist of one or more entities that are linked by associations or compositions. An entity is a CDS view element that represents a node or a projection of a business object. An entity can have various annotations that define the metadata and the semantics of the business object2.

* B. Behavior definition: A behavior definition is a source code artifact that defines the behavior and the validation rules of a business object. A behavior definition can specify the standard CRUD (create, read, update, delete) operations, the draft handling, the authorization checks, and the side effects for a business object. A behavior definition can also define custom actions, validations, and determinations that implement the business logic of a business object3.
The following are not parts of a business object in RESTful Application Programming, because:
* C. Authentication rules: Authentication rules are not part of a business object, but part of a service binding. A service binding is a configuration artifact that defines how a business object is exposed as an OData service. A service binding can specify the authentication method, the authorization scope, the protocol version, and the service options for the OData service4.
* D. Process definition: Process definition is not part of a business object, but part of a workflow. A workflow is a business process that orchestrates the tasks and the events of a business object. A workflow can be defined using the Workflow Editor in the SAP Business Application Studio or the SAP Web IDE. A workflow can use the business object's APIs to trigger or consume events, execute actions, or read or update data5.
References: 1: Business Object | SAP Help Portal 2: CDS View Entities | SAP Help Portal 3: Behavior Definition | SAP Help Portal 4: Service Binding | SAP Help Portal 5: Workflow | SAP Help Portal

NEW QUESTION: 13
In a program you find this source code
AUTHORITY-CHECK OBJECT '/DWO/TRVL (ID 'CNTRY' FIELD 'DE*
ID ACTVT FIELD '03".
Which of the following apply? Note: There are 2 correct answers to this question.
A. If the user is authorized for 'CNTRY = 'DE' AND for 'ACTVT = '03 then the return code is 0.
B. AUTHORITY CHECK verifies whether a user is authorized for/DMO/TRVL" with the listed field values.
C. If the user is NOT authorized for 'CNTRY' = 'DE' OR for 'ACTVT' = '03 then the program will terminate.
D. If the user is authorized for 'CNTRY = 'DE' then the return code is always 0.
Answer: A,B (LEAVE A REPLY)

NEW QUESTION: 14
Which statement can you use to change the contents of a row of data in an internal table?
A. Append table
B. Modify table
C. Insert table
D. Update table
Answer: B (LEAVE A REPLY)
The statement that can be used to change the contents of a row of data in an internal table is MODIFY table.

The MODIFY table statement can be used to change the contents of one or more rows of an internal table, either by specifying the table index, the table key, or a condition. The MODIFY table statement can also be used to change the contents of a database table, by specifying the table name and a work area or an internal table. The MODIFY table statement can use the TRANSPORTING addition to specify which fields should be changed, and the WHERE addition to specify which rows should be changed.
The other statements are not suitable for changing the contents of a row of data in an internal table, as they have different purposes and effects. These statements are:
* APPEND table: This statement can be used to add a new row of data to the end of an internal table, either by specifying a work area or an inline declaration. The APPEND table statement does not change the existing rows of the internal table, but only increases the number of rows by one.
* INSERT table: This statement can be used to insert a new row of data into an internal table, either by specifying the table index, the table key, or a sorted position. The INSERT table statement does not change the existing rows of the internal table, but only shifts them to make room for the new row. The INSERT table statement can also be used to insert a new row of data into a database table, by specifying the table name and a work area or an inline declaration.
* UPDATE table: This statement can be used to update the contents of a database table, by specifying the table name and a work area or an internal table. The UPDATE table statement can use the SET addition to specify which fields should be updated, and the WHERE addition to specify which rows should be updated. The UPDATE table statement does not affect the internal table, but only the corresponding database table.
References: MODIFY table - ABAP Keyword Documentation, APPEND table - ABAP Keyword Documentation, INSERT table - ABAP Keyword Documentation, UPDATE table - ABAP Keyword Documentation

NEW QUESTION: 15
Why would you use Access Controls with CDS Views? Note: There are 2 correct answers to this question.
A. Only the data corresponding to the user's authorization is transferred from the database to the application layer.
B. The system field sy-subrc is set, giving you the result of the authorization check
C. You do not have to remember to implement AUTHORITY CHECK statements.
D. All of the data from the data sources is loaded into your application automatically and filtered there according to the user's authorization.
Answer: A,C (LEAVE A REPLY)
You would use Access Controls with CDS Views for the following reasons:
* A. Only the data corresponding to the user's authorization is transferred from the database to the application layer. This is true because Access Controls allow you to define CDS roles that specify the authorization conditions for accessing a CDS view. The CDS roles are evaluated for

every user at runtime and the system automatically adds the restrictions to the selection conditions of the CDS view.
This ensures that only the data that the user is authorized to see is read from the database and transferred to the application layer. This improves the security and the performance of the data access1.
* C. You do not have to remember to implement AUTHORITY CHECK statements. This is true because Access Controls provide a declarative and centralized way of defining the authorization logic for a CDS view. You do not have to write any procedural code or use the AUTHORITY CHECK statement to check the user's authorization for each data source or field. The system handles the authorization check automatically and transparently for you2.
The following reasons are not valid for using Access Controls with CDS Views:
* B. The system field sy-subrc is set, giving you the result of the authorization check. This is false because the system field sy-subrc is not used by Access Controls. The sy-subrc field is used by the AUTHORITY CHECK statement to indicate the result of the authorization check, but Access Controls do not use this statement. Instead, Access Controls use CDS roles to filter the data according to the user' s authorization2.
* D. All of the data from the data sources is loaded into your application automatically and filtered there according to the user's authorization. This is false because Access Controls do not load all the data from the data sources into the application layer. Access Controls filter the data at the database layer, where the data resides, and only transfer the data that the user is authorized to see to the application layer. This reduces the data transfer and the memory consumption of the application layer1.
References: 1: Access Controls | SAP Help Portal 2: ABAP CDS - Access Control - ABAP Keyword Documentation

NEW QUESTION: 16
In ABAP SQL, which of the following retneves the association field _Airline-Name of a CDS view?
A. @_Airline-Name
B. \ Airnline-Name
C. *_Airline-Name
D. /_Anine-Name
Answer: A (LEAVE A REPLY)

NEW QUESTION: 17
For what kind of applications would you consider using on-stack developer extensions? Note: There are 2 correct answers to this question.
A. Applications that provide APIs for side by side SAP BTP apps
B. Applications that access SAP S/4HANA data using complex SQL
C. Applications that integrate data from several different systems
D. Applications that run separate from SAP S/4HANA
Answer: A,B (LEAVE A REPLY)
On-stack developer extensibility is a type of extensibility that allows you to create development projects directly on the SAP S/4HANA Cloud technology stack. It gives you the opportunity to develop cloud-ready and upgrade-stable custom ABAP applications and services inside the SAP S/4HANA Cloud, public edition system. You can use the ABAP Development Tools in Eclipse to create and deploy your on-stack extensions.
On-stack developer extensibility is suitable for the following kinds of applications:
* Applications that provide APIs for side by side SAP BTP apps. On-stack developer extensibility allows you to create OData services or RESTful APIs based on CDS view entities or projection views. These services or APIs can expose SAP S/4HANA data and logic to other applications that run on the SAP Business Technology Platform (SAP BTP) or other platforms. This way, you can create a loosely coupled integration between your SAP S/4HANA system and your side by side SAP BTP apps.
* Applications that access SAP S/4HANA data using complex SQL. On-stack developer extensibility allows you to use ABAP SQL to access SAP S/4HANA data using complex queries, such as joins, aggregations, filters, parameters, and code pushdown techniques. You can also use ABAP SQL to perform data manipulation operations, such as insert, update, delete, and upsert. This way, you can create applications that require advanced data processing and analysis on SAP S/4HANA data.
The other kinds of applications are not suitable for on-stack developer extensibility, as they have different requirements and challenges. These kinds of applications are:
* Applications that integrate data from several different systems. On-stack developer extensibility is not meant for creating applications that integrate data from multiple sources, such as other SAP systems, third-party systems, or cloud services. This is because on-stack developer extensibility does not support remote access or data replication, and it may cause performance or security issues. For this kind of applications, you should use side by side extensibility, which allows you to create applications that run on the SAP BTP and communicate with the SAP S/4HANA system via public APIs or events.
* Applications that run separate from SAP S/4HANA. On-stack developer extensibility is not meant for creating applications that run independently from the SAP S/4HANA system, such as standalone apps, microservices, or web apps. This is because on-stack developer extensibility

requires a tight coupling with the SAP S/4HANA system, and it may limit the scalability, flexibility, and portability of the applications. For this kind of applications, you should use side by side extensibility, which allows you to create applications that run on the SAP BTP and leverage the cloud-native features and services of the platform.
References: Developer Extensibility in SAP S/4HANA Cloud ABAP Environment, SAP S/4HANA Extensibility - Simplified Guide for Beginners

NEW QUESTION: 18
Given the following code,
DATA gv_text1 TYPE string. "#EC_NEEDED
DATA gv_text2 TYPE string ##NEEDED.
What are valid statements? Note: There are 2 correct answers to this question.
A. ##NEEDED is checked by the syntax checker.
B. The pragma is not checked by the syntax checker.
C. #EC_NEEDED is not checked by the syntax checker.
D. The pseudo-comment is checked by the syntax checker
Answer: A,B (LEAVE A REPLY)
Both statements are valid in ABAP, but they have different effects on the program.
* ##NEEDED is a pragma that can be used to hide warnings from the ABAP compiler syntax check. It tells the check tools that a variable or a parameter is needed for further processing, even if it is not used in the current statement. For example, if you declare a variable without assigning any value to it, you can use ##NEEDED to suppress the warning about unused variables12.
* The pragma is not checked by the syntax checker means that you can use any pragma to hide any warning from the ABAP compiler syntax check, regardless of its effect on the program logic or performance. For example, if you use ##SHADOW to hide a warning about an obscured function, you can also use it to hide a warning about an invalid character in a string12.
You cannot do any of the following:
* #EC_NEEDED is not checked by the syntax checker: This is not a valid statement in ABAP. There is no pseudo-comment with #EC_NEEDED in ABAP3.
* The pseudo-comment is checked by the syntax checker: This is false. Pseudo-comments are obsolete and should no longer be used in ABAP. They were replaced by pragmas since SAP NW 7.0 EhP2 (Enhancement Package)4.
References: 1: Pragmas - ABAP Keyword Documentation - SAP Online Help 2: [What are pragmas and pseudo comments in ABAP? | SAP Blogs - SAP Community] 3: ABAP Keyword Documentation - SAP Online Help 4: What are PRAGMAS and Pseudo comments in SAP ABAP

NEW QUESTION: 19
Which patterns raise an exception? Note: There are 3 correct answers to this question.
A. DATA: gv_target TYPE p DECIMALS 2. CONSTANTS: go intl TYPE i VALUE 3. gv_target -U EXACT (2 gcojntl).
B. DATA: gv_target TYPE string. # CONSTANTS: gco_string TYPE LENGTH 16 VALUE

0123456789ABCDEF*. gv_target = EXACT # gco_string+5 (5)).
C. DATA: gv_target TYPE c LENGTH 5. V # CONSTANTS: ECO string TYPE string VALUE 0123456789ABCDEF". gv_target - EXACT (gco_string + 5 (6)).
D. DATA: Ev target TYPE p DECIMALS 3. CONSTANTS: gcojntl TYPE i VALUE 2. Ev_target -U EXACT #2 / gcojntl).
E. DATA: gv_target TYPE d. s/ # CONSTANTS: gco_date TYPE d VALUE '20331233*. gv_target EXACT (geo_date).
Answer: A,C,E (LEAVE A REPLY)
The patterns that raise an exception are those that use the constructor operator EXACT to perform a lossless assignment or calculation, but the result cannot be converted to the target data type without data loss. The following are the explanations for each pattern:
* A: This pattern raises the exception CX_SY_CONVERSION_LOST because the result of the calculation 2 * 3 is 6, which cannot be assigned to a packed number with two decimal places without losing the integer part. The operator -U is used to perform a lossless calculation with the calculation type decfloat34.
* B: This pattern does not raise an exception because the result of the substring expression gco_string+5 (5) is '6789A', which can be assigned to a string without data loss. The operator EXACT # is used to perform a lossless assignment with the data type of the argument.
* C: This pattern raises the exception CX_SY_CONVERSION_LOST because the result of the substring expression gco_string+5(6) is '6789AB', which cannot be assigned to a character field with length 5 without losing the last character. The operator EXACT is used to perform a lossless assignment with the data type of the target field.
* D: This pattern does not raise an exception because the result of the calculation 2 / 2 is 1, which can be assigned to a packed number with three decimal places without data loss. The operator -U is used to perform a lossless calculation with the calculation type decfloat34.
* E: This pattern raises the exception CX_SY_CONVERSION_ERROR because the constant gco_date contains an invalid value '20331233' for a date data type, which cannot be converted to a valid date.
The operator EXACT is used to perform a lossless assignment with the data type of the target field.
References: EXACT - Lossless Operator - ABAP Keyword Documentation, Lossless Assignments - ABAP Keyword Documentation

NEW QUESTION: 20
/DMO/I_Connection is a CDS view.
What variable type is connection full based on the following code? DATA connection full TYPE /DMD/I_Connection.
A. Simple variable
B. Structure
C. Internal Table
Answer: (SHOW ANSWER)

Based on the following code, the variable type of connection_full is a structure. A structure is a complex data type that consists of a group of related data objects, called components, that have their own data types and names. A structure can be defined using the TYPES statement or based on an existing structure type, such as a CDS view entity or a CDS DDIC-based view. In this case, the variable connection_full is declared using the TYPE addition, which means that it has the same structure type as the CDS view entity /DMO/I_Connection.
The CDS view entity /DMO/I_Connection is a data model view that defines a data model based on the database table /DMO/Connection. The CDS view entity /DMO/I_Connection has the following components:
carrid, connid, airpfrom, airpto, distance, and fltime. Therefore, the variable connection_full has the same components as the CDS view entity /DMO/I_Connection, and each component has the same data type and length as the corresponding field in the database table /DMO/Connection.
References: CDS Data Model Views - ABAP Keyword Documentation, DATA - ABAP Keyword Documentation, Structure Types - ABAP Keyword Documentation

NEW QUESTION: 21
Using ABAP SQL, which select statement selects the mat field on line #17?
A. SELECT mat FROM Material...
B. SELECT mat FROM demo_sales_cds_so_i_ve...
C. SELECT mat FROM demo_sales_so_i...
D. SELECT mat FROM demo sales cds material ve...
Answer: (SHOW ANSWER)
Using ABAP SQL, the select statement that selects the mat field on line #17 is:
SELECT mat FROM demo_sales_cds_so_i_ve...
This statement selects the mat field from the CDS view demo_sales_cds_so_i_ve, which is defined on line #1.
The CDS view demo_sales_cds_so_i_ve is a projection view that projects the fields of the CDS view demo_sales_cds_so_i, which is defined on line #2. The CDS view demo_sales_cds_so_i is a join view that joins the fields of the database table demo_sales_so_i, which is defined on line #3, and the CDS view demo_sales_cds_material_ve, which is defined on line #4. The CDS view demo_sales_cds_material_ve is a value help view that provides value help for the material field of the database table demo_sales_so_i. The mat field is an alias for the material field of the database table demo_sales_so_i, which is defined on line #91.
The other options are not valid because:
* A. SELECT mat FROM Material... is not valid because Material is not a valid data source in the given code. There is no CDS view or database table named Material.
* C. SELECT mat FROM demo_sales_so_i... is not valid because demo_sales_so_i is not a valid data source in the given code. There is no CDS view named demo_sales_so_i, only a database table. To access a database table, the keyword TABLE must be used, such as SELECT mat FROM TABLE demo_sales_so_i...

* D. SELECT mat FROM demo sales cds material ve... is not valid because demo sales cds material ve is not a valid data source in the given code. There is no CDS view or database table named demo sales cds material ve. The correct name of the CDS view is demo_sales_cds_material_ve, with underscores instead of spaces.
References: 1: Projection Views - ABAP Keyword Documentation

NEW QUESTION: 22

Given the following Core Data Services View Entity Data Definition,

```
1 @AccessControl.authorizationCheck: #NOT_REQUIRED
2 DEFINE VIEW ENTITY demo_cds_data_source_matrix
3  AS SELECT FROM
4  <source>
5  {
6    KEY field_1,
7    field_2,
8    field_3
9  }
```


Which of the following types are permitted to be used for <source> on line #4? Note: There are 2 correct answers to this question.

A. A database table from the ABAP Dictionary

B. A CDS DDIC-based view

C. An external view from the ABAP Dictionary

D. A database view from the ABAP Dictionary

Answer: A,B (LEAVE A REPLY)

The <source> clause in the CDS View Entity Data Definition can be used to specify the data source for the view entity. The <source> clause can accept different types of data sources, depending on the type of the view entity1.

* A database table from the ABAP Dictionary: This is a valid type of data source for a CDS View Entity Data Definition. A database table from the ABAP Dictionary is a table that is defined in the ABAP Dictionary using the keyword TABLE or TABLE OF. The name of the database table must be unique within its namespace and must not contain any special characters2.

* A CDS DDIC-based view: This is also a valid type of data source for a CDS View Entity Data Definition. A CDS DDIC-based view is a view that is defined in the Core Data Services using the keyword DEFINE VIEW ENTITY. The name of the CDS DDIC-based view must be unique within its namespace and must not contain any special characters3.

You cannot do any of the following:

* An external view from the ABAP Dictionary: This is not a valid type of data source for a CDS View Entity Data Definition. An external view from the ABAP Dictionary is a view that is defined in an external application using any language supported by SAP, such as SQL, PL/SQL, or Java.

The name of the external view must be unique within its namespace and must not contain any special characters4.
* A database view from the ABAP Dictionary: This is not a valid type of data source for a CDS View Entity Data Definition. A database view from the ABAP Dictionary is a view that is defined in an external application using any language supported by SAP, such as SQL, PL/SQL, or Java. The name of the database view must be unique within its namespace and must not contain any special characters4.
References: 1: CDS DDL - DEFINE VIEW ENTITY - ABAP Keyword Documentation - SAP Online Help 2:
ABAP Dictionary Tables - SAP Online Help 3: CDS DDL - DEFINE VIEW ENTITY - ABAP Keyword Documentation - SAP Online Help 4: ABAP Dictionary Views - SAP Online Help

NEW QUESTION: 23
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
Answer: B,D (LEAVE A REPLY)
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

NEW QUESTION: 24
Which of the following are valid sort operations for internal tables? Note: There are 3 correct answers to this question.
A. SORT itab BY fieldl ASCENDING field2 DESCENDING.
Sort a standard table using
B. Sort a standard table using
SORT itab ASCENDING.
Sort a sorted table using
C. SORT itab DESCENDING.
D. SORT itab BY field1 field2.
Sort a standard table using
E. SORT itab.
Sort a sorted table using
Answer: B,D,E (LEAVE A REPLY)

NEW QUESTION: 25
Which internal table type allows unique and non-unique keys?
A. Sorted
B. Hashed
C. Standard
Answer: (SHOW ANSWER)
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

NEW QUESTION: 26
What are some of the reasons that Core Data Services are preferable to the classical approach to data modeling? Note: There are 2 correct answers to this question.
A. They implement code pushdown.
B. They avoid data transfer completely.
C. They transfer computational results to the application server.
D. They compute results on the application server.
Answer: (SHOW ANSWER)
Core Data Services (CDS) are preferable to the classical approach to data modeling for several reasons, but two of them are:
* They implement code pushdown. Code pushdown is the principle of moving data-intensive logic from the application server to the database server, where the data resides. This reduces the data transfer between the application server and the database server, which improves the performance and scalability of the application. CDS enable code pushdown by allowing the definition of semantic data models and business logic in the database layer, using SQL and SQL-based expressions1.
* They transfer computational results to the application server. CDS allow the application server to access the data and the logic defined in the database layer by using Open SQL statements. Open SQL is a standardized and simplified subset of SQL that can be used across different database platforms. Open SQL statements are translated into native SQL statements by the ABAP runtime environment and executed on the database server. The results of the computation are then transferred to the application server, where they can be further processed or displayed2.
References: 1: ABAP - Core Data Services (ABAP CDS) - ABAP Keyword Documentation 2: Open SQL - ABAP Keyword Documentation

NEW QUESTION: 27
Which of the following are features of Core Data Services? Note: There are 3 correct answers to this question.
A. Inheritance
B. Associations
C. Annotations
D. Delegation
E. Structured Query Language (SQL)
Answer: B,C,E (LEAVE A REPLY)
Core Data Services (CDS) is a framework for defining and consuming semantically rich data models in SAP HANA. CDS supports various features that enhance the capabilities of SQL and enable developers to create data models that are optimized for performance, readability, and extensibility12. Some of the features of CDS are:

* Associations: Associations are a way of defining relationships between CDS entities, such as tables or views. Associations enable navigation and path expressions in CDS queries, which allow accessing data from related entities without explicit joins. Associations also support cardinality, referential constraints, and cascading options34.
* Annotations: Annotations are a way of adding metadata to CDS entities or their elements, such as fields or parameters. Annotations provide additional information or instructions for the CDS compiler, the database, or the consumers of the CDS views. Annotations can be used for various purposes, such as defining access control, UI rendering, OData exposure, or search capabilities5 .
* Structured Query Language (SQL): SQL is the standard language for querying and manipulating data in relational databases. CDS is based on SQL and extends it with additional features and syntax. CDS supports SQL features such as joins, aggregations, filters, expressions, functions, and subqueries. CDS also supports SQL Script, which is a scripting language for stored procedures and functions in SAP HANA .
You cannot do any of the following:
* Inheritance: Inheritance is not a feature of CDS. Inheritance is a concept in object-oriented programming that allows a class to inherit the properties and methods of another class. CDS does not support object-oriented programming or classes.
* Delegation: Delegation is not a feature of CDS. Delegation is a concept in object-oriented programming that allows an object to delegate some of its responsibilities to another object. CDS does not support object-oriented programming or objects.
References: 1: Core Data Services (CDS) | CAPire 2: Core Data Services [CDS] in SAP S/4 HANA | SAP Blogs 3: Associations in Core Data Services (CDS) | SAP Help Portal 4: [CDS DDL - Association - ABAP Keyword Documentation - SAP Online Help] 5: [Annotations in Core Data Services (CDS) | SAP Help Portal] : [CDS DDL - Annotation - ABAP Keyword Documentation - SAP Online Help] : [Structured Query Language (SQL) | SAP Help Portal] : [CDS DDL - SQL Features - ABAP Keyword Documentation - SAP Online Help] : [Object-Oriented Programming in ABAP | SAP Help Portal]

NEW QUESTION: 28
Which of the following are parts of the definition of a new database table?Note: There are 2 correct answers to this question.
A. Semantic table attributes
B. Partitioning attribues
C. Extension
D. Field list
Answer: A,D (LEAVE A REPLY)

NEW QUESTION: 29
Which of the following is a generic internal table type?
A. SORTED TABLE

B. INDEX TABLE
C. STANDARD TABLE
D. HASHED TABLE
Answer: B (LEAVE A REPLY)
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.
References: 1: Generic Table Types - ABAP Dictionary - SAP Online Help 2: Generic ABAP Types - ABAP Keyword Documentation - SAP Online Help 3: Sorted Tables - ABAP Keyword Documentation - SAP Online Help 4: Standard Tables - ABAP Keyword Documentation - SAP Online Help 5: Hashed Tables - ABAP Keyword Documentation - SAP Online Help

NEW QUESTION: 30

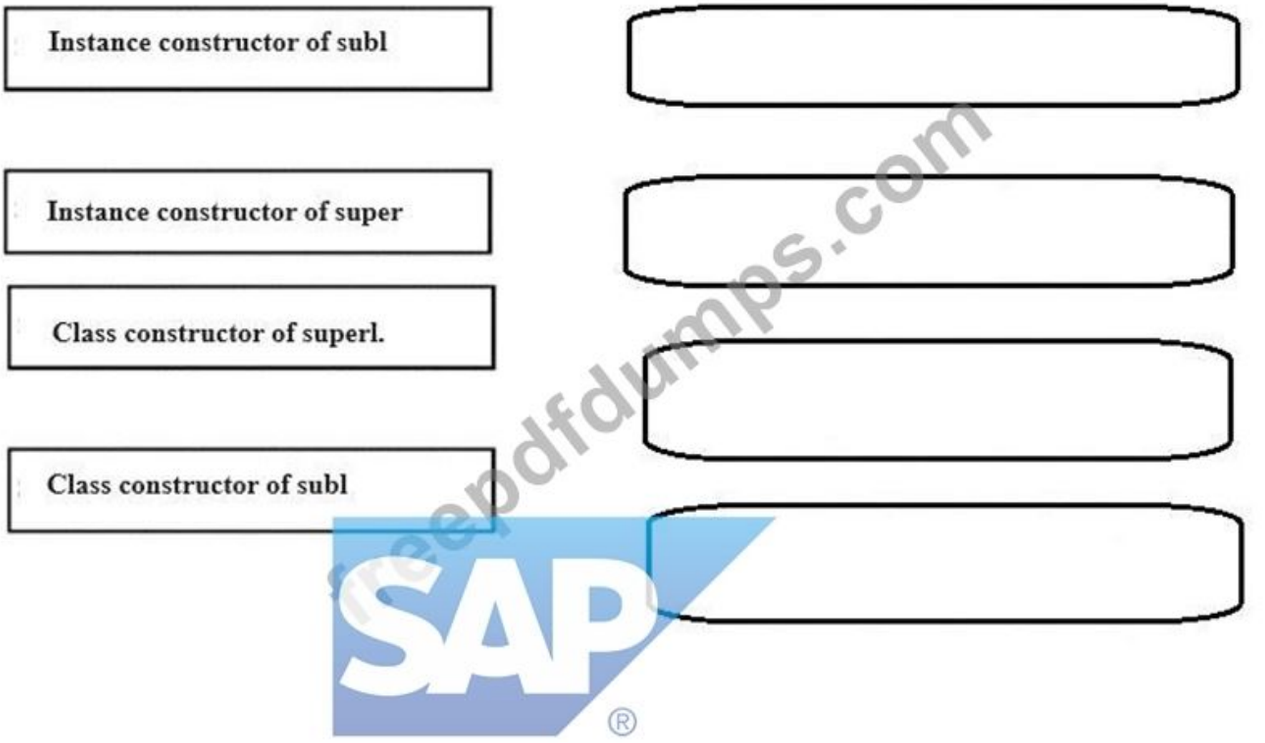
Given the following Core Data Services View Entity Data Definition:

```
1 @AccessControl.authorizationCheck: #NOT_REQUIRED
2 DEFINE VIEW ENTITY demo_cds_assoc_element
3   AS SELECT FROM scarr
4   ASSOCIATION OF ONE TO MANY demo_cds_assoc_spfli AS _spfli
5   ON scarr.carrid = _spfli.carrid
6   {
7     KEY carrid,
8 ?
9     carrname
10  }
```

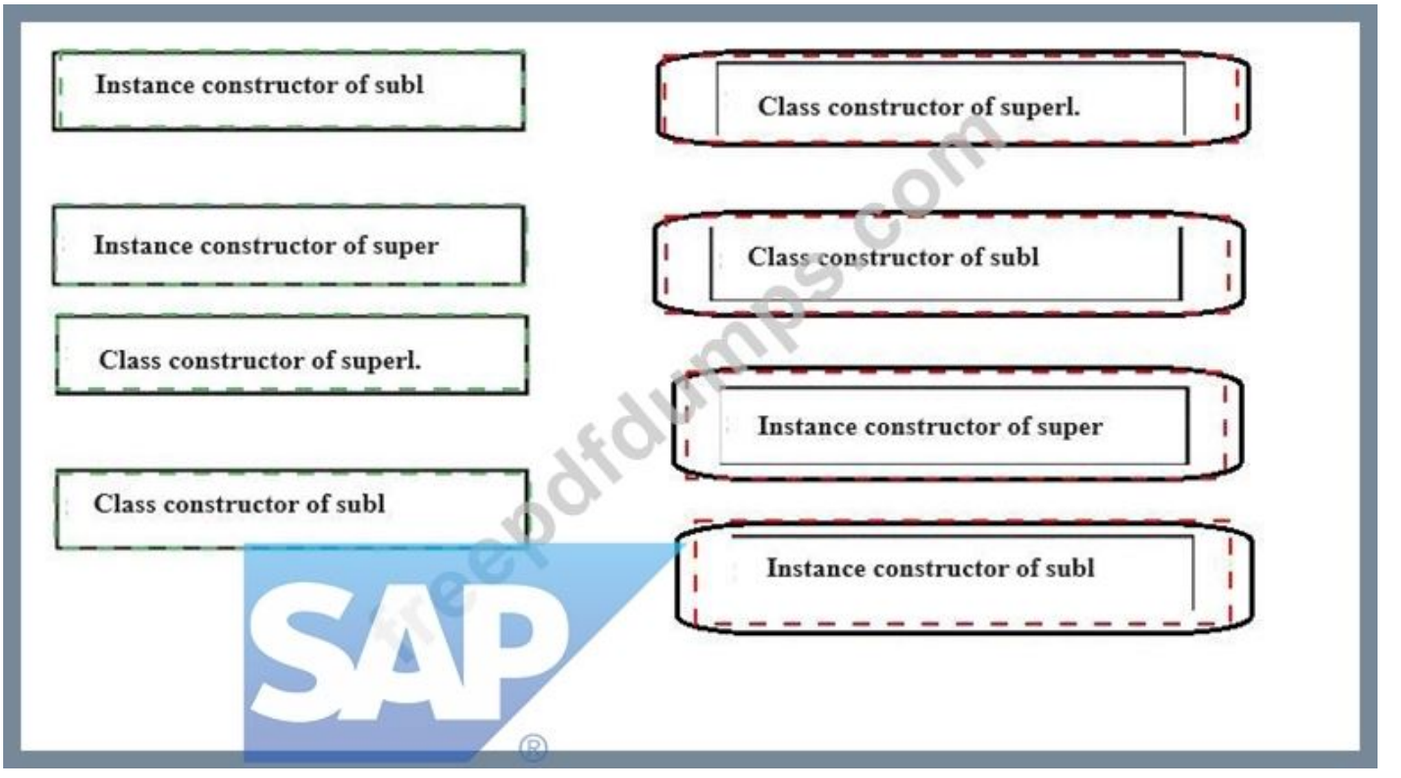

The "demo_ods_assoc_spfi data source referenced in line #4 contains a field "connid" which you would like to expose in the element list.
Which of the following statements would do this if inserted on line #8?
A. demo_ods_assoc_spfli.connid,
B. demo_ods_assoc_spfli-connid/

C. spfli-connid,
D. _spfli.connid/
Answer: (SHOW ANSWER)
The statement that can be used to expose the field "connid" of the data source "demo_ods_assoc_spfli" in the element list is A. demo_ods_assoc_spfli.connid,. This statement uses the dot notation to access the field
"connid" of the data source "demo_ods_assoc_spfli", which is an association defined on line #4. The association "demo_ods_assoc_spfli" links the data source "demo_ods" with the table "spfli" using the field
"carrid". The statement also ends with a comma to separate it from the next element in the list12. You cannot do any of the following:
* B. demo_ods_assoc_spfli-connid/: This statement uses the wrong syntax to access the field "connid" of the data source "demo_ods_assoc_spfli". The dash notation is used to access the components of a structure or a table, not the fields of a data source. The statement also ends with a slash, which is not a valid separator for the element list12.
* C. spfli-connid,: This statement uses the wrong data source name to access the field "connid". The data source name should be "demo_ods_assoc_spfli", not "spfli". The statement also uses the wrong syntax to access the field "connid", as explained above12.
* D. _spfli.connid/: This statement uses the wrong data source name and the wrong separator to access the field "connid". The data source name should be "demo_ods_assoc_spfli", not "_spfli". The statement also ends with a slash, which is not a valid separator for the element list12.
References: 1: ABAP CDS - SELECT, select_list - ABAP Keyword Documentation - SAP Online Help 2: ABAP CDS - SELECT, from - ABAP Keyword Documentation - SAP Online Help

NEW QUESTION: 31
You have a superclass superl and a subclass subl of superl. Each class has an instance constructor and a static constructor. The first statement of your program creates an instance of subl. In which sequence will the constructors be executed?



Answer:



Explanation:

The sequence in which the constructors will be executed is as follows:

* Class constructor of superl. This is because the class constructor is a static method that is executed automatically before the class is accessed for the first time. The class constructor is used to initialize the static attributes and components of the class. The class constructor of the superclass is executed before the class constructor of the subclass, as the subclass inherits the static components of the superclass12

* Class constructor of subl. This is because the class constructor is a static method that is executed automatically before the class is accessed for the first time. The class constructor is used to initialize the static attributes and components of the class. The class constructor of the subclass is executed after the class constructor of the superclass, as the subclass inherits the static components of the superclass12
* Instance constructor of superl. This is because the instance constructor is an instance method that is executed automatically when an instance of the class is created using the statement CREATE OBJECT.
The instance constructor is used to initialize the instance attributes and components of the class. The instance constructor of the superclass is executed before the instance constructor of the subclass, as the subclass inherits the instance components of the superclass. The instance constructor of the subclass must call the instance constructor of the superclass explicitly using super->constructor, unless the superclass is the root node object12
* Instance constructor of subl. This is because the instance constructor is an instance method that is executed automatically when an instance of the class is created using the statement CREATE OBJECT.
The instance constructor is used to initialize the instance attributes and components of the class. The instance constructor of the subclass is executed after the instance constructor of the superclass, as the subclass inherits the instance components of the superclass. The instance constructor of the subclass must call the instance constructor of the superclass explicitly using super->constructor, unless the superclass is the root node object12 References: Constructors of Classes - ABAP Keyword Documentation, METHODS - constructor - ABAP Keyword Documentation

NEW QUESTION: 32
Which of the following are ABAP Cloud Development Model rules?
Note: There are 2 correct answers to this question.
A. Use public SAP APIs and SAP extension points.
B. Build ABAP RESTful application programming model-based services.
C. Reverse modifications when a suitable public SAP API becomes available.
D. Build ABAP reports with either ABAP List Viewer (ALV) or SAP Fiori.
Answer: A,C (LEAVE A REPLY)

* Use public SAP APIs and SAP extension points. This rule ensures that the ABAP Cloud code is stable, reliable, and compatible with the SAP solutions and the cloud operations. Public SAP APIs and SAP extension points are the only allowed interfaces and objects to access the SAP platform and the SAP applications. They are documented, tested, and supported by SAP. They also guarantee the lifecycle stability and the upgradeability of the ABAP Cloud code1.
* Build ABAP RESTful application programming model-based services. This rule ensures that the ABAP Cloud code follows the state-of-the-art development paradigm for building cloud-ready business services. The ABAP RESTful application programming model (RAP) is a framework that provides a consistent end-to-end programming model for creating, reading, updating, and deleting (CRUD) business data. RAP also supports draft handling, authorization checks, side effects, validations, and custom actions. RAP exposes the business services as OData services that can be consumed by SAP Fiori apps or other clients2.

NEW QUESTION: 33
What are advantages of using a field symbol for internal table row access? Note: There are 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.
Answer: B,D (LEAVE A REPLY)
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 25-40% 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 of the 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

NEW QUESTION: 34
What are some characteristics of secondary keys for internal tables? Note: There are 3 correct answers to this question.
A. Secondary keys must be chosen explicitly when you actually read from an internal table.
B. Multiple secondary keys are allowed for any kind of internal table.
C. Hashed secondary keys do NOT have to be unique.
D. Sorted secondary keys do NOT have to be unique.
E. Secondary keys can only be created for standard tables.
Answer: A,B,D (LEAVE A REPLY)
Secondary keys are additional keys that can be defined for internal tables to optimize the access to the table using fields that are not part of the primary key. Secondary keys can be either sorted or hashed, depending on the table type and the uniqueness of the key. Secondary keys have the following characteristics1:
* A. Secondary keys must be chosen explicitly when you actually read from an internal table. This means that when you use a READ TABLE or a LOOP AT statement to access an internal table, you have to specify the secondary key that you want to use with the USING KEY addition. For example, the following statement reads an internal table itab using a secondary key sec_key:
READ TABLE itab USING KEY sec_key INTO DATA(wa).
If you do not specify the secondary key, the system will use the primary key by default2.
* B. Multiple secondary keys are allowed for any kind of internal table. This means that you can define more than one secondary key for an internal table, regardless of the table type. For example, the following statement defines an internal table itab with two secondary keys sec_key_1 and sec_key_2:
DATA itab TYPE SORTED TABLE OF ty_itab WITH NON-UNIQUE KEY sec_key_1 COMPONENTS field1 field2 sec_key_2 COMPONENTS field3 field4.
You can then choose which secondary key to use when you access the internal table1.
* D. Sorted secondary keys do NOT have to be unique. This means that you can define a sorted secondary key for an internal table that allows duplicate values for the key fields. A sorted secondary key maintains a predefined sorting order for the internal table, which is defined by the key fields in the order in which they are specified. For example, the following statement defines a sorted secondary key sec_key for an internal table itab that sorts the table by field1 in ascending order and field2 in descending order:
DATA itab TYPE STANDARD TABLE OF ty_itab WITH NON-UNIQUE SORTED KEY sec_key COMPONENTS field1 ASCENDING field2 DESCENDING.

You can then access the internal table using the sorted secondary key with a binary search algorithm, which is faster than a linear search3.
The following are not characteristics of secondary keys for internal tables, because:
* C. Hashed secondary keys do NOT have to be unique. This is false because hashed secondary keys must be unique. This means that you can only define a hashed secondary key for an internal table that does not allow duplicate values for the key fields. A hashed secondary key does not have a predefined sorting order for the internal table, but uses a hash algorithm to store and access the table rows. For example, the following statement defines a hashed secondary key sec_key for an internal table itab that hashes the table by field1 and field2:
DATA itab TYPE STANDARD TABLE OF ty_itab WITH UNIQUE HASHED KEY sec_key COMPONENTS field1 field2.
You can then access the internal table using the hashed secondary key with a direct access algorithm, which is very fast.
* E. Secondary keys can only be created for standard tables. This is false because secondary keys can be created for any kind of internal table, such as standard tables, sorted tables, and hashed tables.
However, the type of the secondary key depends on the type of the internal table. For example, a standard table can have sorted or hashed secondary keys, a sorted table can have sorted secondary keys, and a hashed table can have hashed secondary keys1.
References: 1: Secondary Table Key - ABAP Keyword Documentation 2: READ TABLE - ABAP Keyword Documentation 3: Sorted Tables - ABAP Keyword Documentation : Hashed Tables - ABAP Keyword Documentation

NEW QUESTION: 35
What is the sequence priority when evaluating a logical expression?
A. NOT 1
B. OR 3
C. AND 2
D. A B C
E. CAB
F. A C B
G. B A C
Answer: C (LEAVE A REPLY)
The sequence priority when evaluating a logical expression is C. A C B, which means NOT, AND, OR. This is the order of precedence of the Boolean operators in ABAP, which determines how the system implicitly parenthesizes all logical expressions that are not closed by explicit parentheses. The operator with the highest priority is evaluated first, and the operator with the lowest priority is evaluated last. The order of precedence of the Boolean operators in ABAP is as follows12:
* NOT: The NOT operator is a unary operator that negates the logical expression that follows it. It has the highest priority and is evaluated before any other operator. For example, in the

expression NOT a AND b, the NOT operator is applied to a first, and then the AND operator is applied to the result and b.
* AND: The AND operator is a binary operator that returns true if both logical expressions on its left and right are true, and false otherwise. It has the second highest priority and is evaluated before the OR and EQUIV operators. For example, in the expression a AND b OR c, the AND operator is applied to a and b first, and then the OR operator is applied to the result and c.
* OR: The OR operator is a binary operator that returns true if either or both logical expressions on its left and right are true, and false otherwise. It has the third highest priority and is evaluated after the NOT and AND operators, but before the EQUIV operator. For example, in the expression a OR b EQUIV c, the OR operator is applied to a and b first, and then the EQUIV operator is applied to the result and c.
* EQUIV: The EQUIV operator is a binary operator that returns true if both logical expressions on its left and right have the same truth value, and false otherwise. It has the lowest priority and is evaluated after all other operators. For example, in the expression a AND b EQUIV c OR d, the EQUIV operator is applied to a AND b and c last, after the AND and OR operators are applied.
References: 1: log_exp - Boolean Operators and Parentheses - ABAP Keyword Documentation - SAP Online Help 2: Logical Expressions (log_exp) - ABAP Keyword Documentation - SAP Online Help

NEW QUESTION: 36
You want to provide a short description of the data definition for developers that will be attached to the database view

```
1 @AccessControl.authorizationCheck: #NOT_REQUIRED
2 ?
3 DEFINE VIEW ENTITY demo_sales_cds_so_ve_simple
4   AS SELECT FROM demo_sales_order AS SalesOrder
5   {
6     KEY so_key,
7         buyer_id     AS BuyerID,
8         currency_sum AS currencySum
9   }
```

You want to provide a short description of the data definition for developers that will be attached to the database view

Which of the following annotations would do this if you inserted it on line #27
A. @UI headerinto description label
B. @UI.badge.title.label
C. @EndUserText.quickInfo
D. @EndUserText label
Answer: (SHOW ANSWER)
The annotation that can be used to provide a short description of the data definition for developers that will be attached to the database view is the @EndUserText.label annotation. This annotation is used to specify a text label for the data definition that can be displayed in the development tools or in the documentation. The annotation can be inserted on line #27 in the code snippet provided in the question12. For example:

* The following code snippet uses the @EndUserText.label annotation to provide a short description of the data definition for the CDS view ZCDS_VIEW:
@AbapCatalog.sqlViewName: 'ZCDS_VIEW' @AbapCatalog.compiler.compareFilter: true @AbapCatalog.
preserveKey: true @AccessControl.authorizationCheck: #CHECK @EndUserText.label: 'CDS view for flight data' "short description for developers define view ZCDS_VIEW as select from sflight { key carrid, key connid, key fldate, seatsmax, seatsocc } You cannot do any of the following:
* @UI.headerInfo.description.label: This annotation is used to specify a text label for the description field of the header information of a UI element. This annotation is not relevant for the data definition of a database view12.
* @UI.badge.title.label: This annotation is used to specify a text label for the title field of a badge UI element. This annotation is not relevant for the data definition of a database view12.
* @EndUserText.quickInfo: This annotation is used to specify a quick information text for the data definition that can be displayed as a tooltip in the development tools or in the documentation. This annotation is not the same as a short description or a label for the data definition12.
References: 1: ABAP CDS - SAP Annotations - ABAP Keyword Documentation - SAP Online Help 2: ABAP CDS - Data Definitions - ABAP Keyword Documentation - SAP Online Help