DUMPSBOSS.

# Oracle Hyperion Data Relationship Management Essentials

## Oracle 1z0-588

## Version Demo

**Total Demo Questions: 11**

**Total Premium Questions: 112**

**Buy Premium PDF**

https://dumpsboss.co

support@dumpsboss.co

support@dumpsboss.co
dumpsboss.co

# Topic Break Down

| Topic | No. of Questions |
|---|---|
| Topic 1, Data Relationship Management Overview | 6 |
| Topic 2, Data Relationship Management Architecture | 3 |
| Topic 3, Version Management | 21 |
| Topic 4, Hierarchy Management | 10 |
| Topic 5, Node Management | 5 |
| Topic 6, Property Management | 20 |
| Topic 7, Importing and Exporting | 25 |
| Topic 8, Validations | 9 |
| Topic 9, Action Scripts | 5 |
| Topic 10, Security | 8 |
| Total | 112 |

QUESTION NO: 1

Identify the true statements about the DRM Change Tracking properties.

**A.** The "Last Changed On" property captures the date and time of the node's most recent change.

**B.** "Last Changed By" provides the user name of the person making the change.

**C.** The "Node Changed" property is false after a node has been added and is changed to True after it is modified.

**D.** The DRM Change-Tracking Properties are found under the System property category.

**E.** The data is managed by the system and updated as users modify or update nodes within a hierarchy.

**F.** Change Tracking properties can be overwritten by Admin Users.

**ANSWER: B**

**Explanation:**

"Last Changed By" provides the user name of the person making the change. In Oracle Data Relationship Management, change-tracking information is maintained for nodes so that administrators and data stewards can identify the user responsible for the latest update. The value recorded for this property is the DRM user associated with the change operation, creating an audit trail that supports governance, review, and investigation of hierarchy maintenance activity. This type of system-maintained metadata is particularly useful when several users are working in the same version or hierarchy, because it identifies accountability for the node's current change state without requiring users to enter the information manually. Oracle's DRM documentation describes change tracking as part of the product's node and property-management capabilities, and Oracle's enterprise performance-management documentation also emphasizes auditability and controlled data maintenance. See the Oracle Data Relationship Management documentation library and the Oracle Enterprise Performance Management master data management overview for related product documentation and governance context.

**QUESTION NO: 2**

You decide to implement node type? for the Account dimension. What are the required steps that must be completed to implement a node type?

1. Under Administer, create node types with valid assigned DRM elements.

2. Create a hierarchy property that contains a list of values that matches the node types defined.

3. Create a local node property called "HierarchyNodeType".

4. For version, set the HierarchyNodeType to "Dimension".

5. For the hierarchy, set the Dimension property to the desired Node type value (for example. Account dimension type for the Account hierarchy).

6. Upload a glyph for each node type.

**A.** 1, 2, 3, 5 only

**B.** 1, 3, 4, 5 only

**C.** 1, 2, 3, 4, 5 only

**D.** 1, 2, 3, 5, 6 only

**E.** 1, 2, 3, 4, 5, 6

**ANSWER: C**

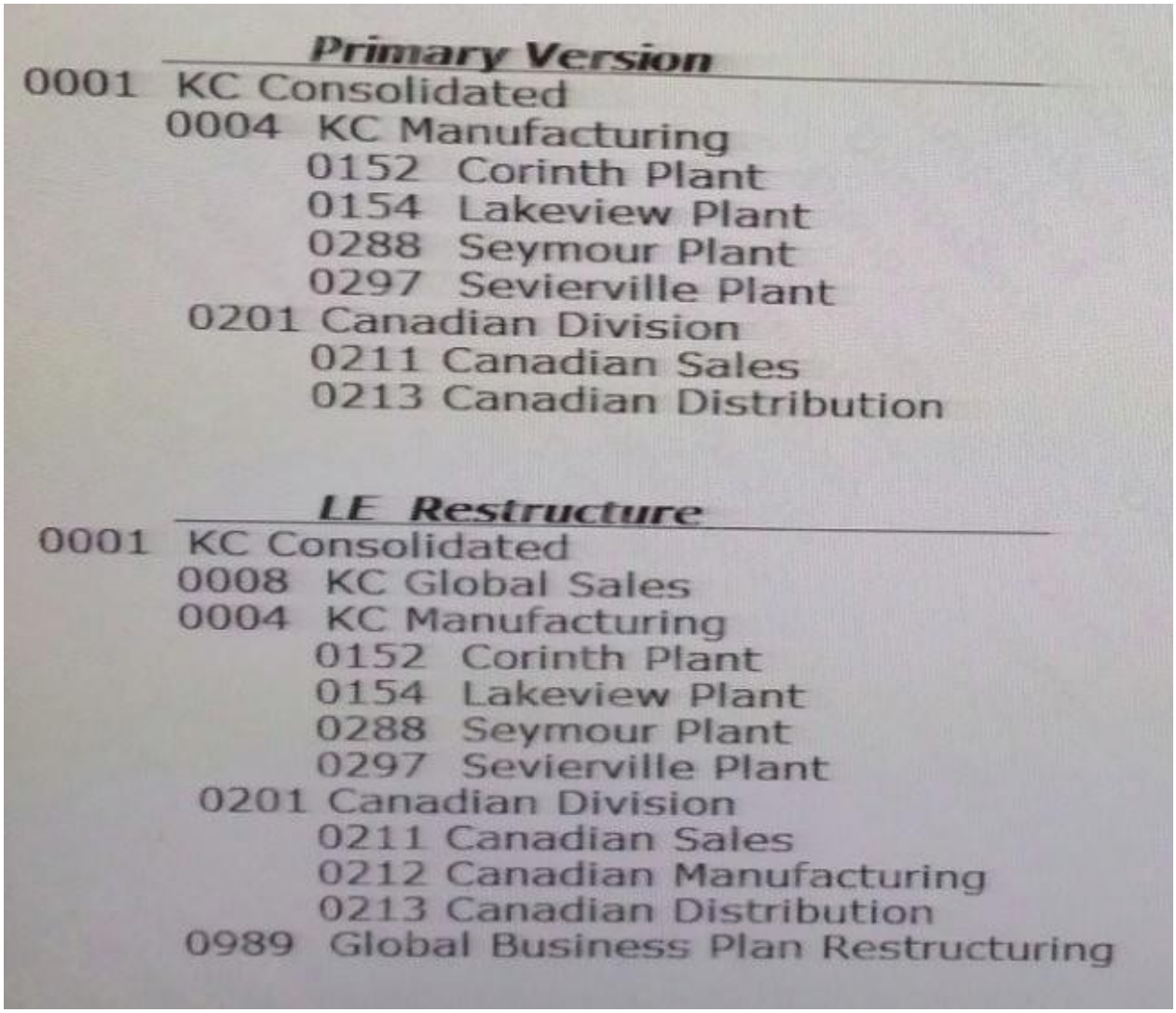
**Explanation:**

The required implementation sequence is **1, 2, 3, 4, 5 only**. In Data Relationship Management, node types must first be defined in the administration area and associated with valid DRM elements. A hierarchy property is then needed to hold the allowable node-type values, so that each hierarchy can be classified using one of the defined types. The local node property named `HierarchyNodeType` establishes the property that DRM uses for node-type determination. At the version level, this local property must be mapped to the hierarchy property—in this case, `Dimension`. Finally, each hierarchy receives the appropriate value in its `Dimension` property, such as the Account node type for an Account hierarchy. Together, these settings define the node-type metadata, make the values available to the version, and apply the selected type to each hierarchy. A glyph is a visual representation and is not required for the node-type configuration itself. Oracle's DRM documentation describes administration of node types and hierarchy/node properties in the Oracle Data Relationship Management User's Guide. Related DRM product documentation is available from the Oracle Enterprise Performance Management DRM documentation library.

## QUESTION NO: 3

"Primary" and "LE_Restructure" are both versions in your DRM application and both have a hierarchy named "Legal Entity".

The nodes and structure of each hierarchy are as follows:




Select the nodes that would be included in a Structure Compare for Differences.

**A.** 0008 KC Global Sales

**B.** 0989 Global Business Plan Restructuring

**C.** 0201 Canadian Division

**D.** 000l KC Consolidated

**E.** 0154 Lakeview Plant

**F.** 0212 Canadian Manufacturing

**ANSWER: B**

**Explanation:**

**0989 Global Business Plan Restructuring** is included because a Structure Compare evaluates the relationships that form a hierarchy's structure, rather than merely comparing whether node records exist in both versions. In DRM, a node can occur in both compared versions with the same identifying information, yet still be a structural difference when its placement or relationship in the selected hierarchy has changed. The displayed Legal Entity structures show that this node is involved in the restructuring relationship that differs between the Primary and LE_Restructure versions. Therefore, it is returned when the comparison is run with the Differences result set.

Structure comparisons are useful for validating reorganizations because they identify hierarchy changes such as a node being moved to another parent, added to or removed from a branch, or otherwise having a different occurrence relationship. This lets administrators focus on the specific organizational change represented by the restructure instead of treating unchanged nodes as differences. Oracle's DRM documentation describes comparison functionality as a way to compare versions and hierarchies and review differences in their structural relationships. See the Oracle Data Relationship Management Administrator's Guide and Oracle's Data Relationship Management product information.

**QUESTION NO: 4**

Per the example:

What are the three outcomes after running the action script?

**A.** Nothing changes in DRM, because the action script will fail to import due to formatting issues.

**B.** A hierarchy called "Product" will be created in the Reorganization version in DRM.

**C.** The top node in the hierarchy is All Products.

**D.** Products 100, 200, 300, and 400 are first created under the incorrect parent but are subsequently moved in the action script.

**E.** The two lines with the "Move" command will fail, because a value is missing from the last column for those lines.

**F.** Products 100-10, 100-20, 100-30, 200-10, 200-20, 200-40, 400-10, and 400-30 are some of the leaf nodes in the hierarchy.

**ANSWER: B C F**

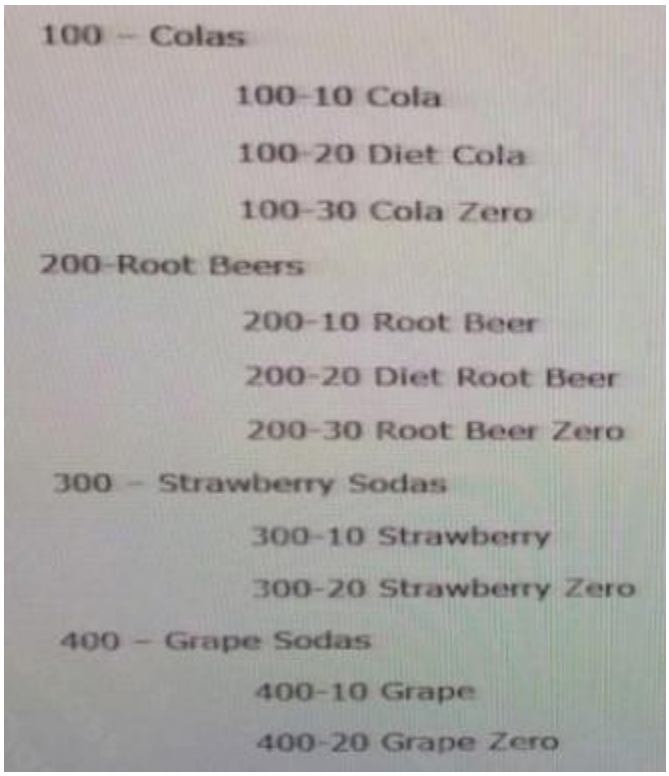
**Explanation:**

The action script creates the **Product** hierarchy in the **Reorganization** version and establishes **All Products** as its root, or top, node. In Data Relationship Management, an action-script import applies the specified version, hierarchy, node, and relationship actions in the file, allowing a hierarchy and its member relationships to be built in a controlled sequence. Once the root is established, the product members in the script are added beneath the appropriate product branches. Therefore, product codes such as 100-10, 100-20, 100-30, 200-10, 200-20, 200-40, 400-10, and 400-30 occur at the bottom of their respective branches and are leaf nodes: they do not act as parents for further product members in the illustrated hierarchy. The result is a usable Product hierarchy whose structure begins with All Products and contains the listed detailed product members. Oracle's DRM documentation describes importing and maintaining hierarchy structures through data and action-oriented processes; see the Oracle Data Relationship Management 11.2 documentation library and Oracle's Enterprise Performance Management documentation resources.

**QUESTION NO: 5**

Per the following example:

All Products




A user is assigned to two node access group, G_Colas and G_Diet Cola. G_Colas is assigned Add permission for the Colas node in the product hierarchy and the G_Diet Cola group is assigned I dit permission for the Diet Cola node.

He is assigned the Interactive User role. Identify the three true statements.

**A.** The user can add new nodes into the Product category under the "Colas" node.

**B.** The user can edit the Diet Cola node only.

**C.** The user can delete nodes under the "Colas'* node.

**D.** The user only sees the "Diet Cola" node in the Product hierarchy.

**E.** The user cannot delete nodes under the "Colas" node.

**F.** The user may inactivate nodes under the "Colas" node.

**ANSWER: A B E**

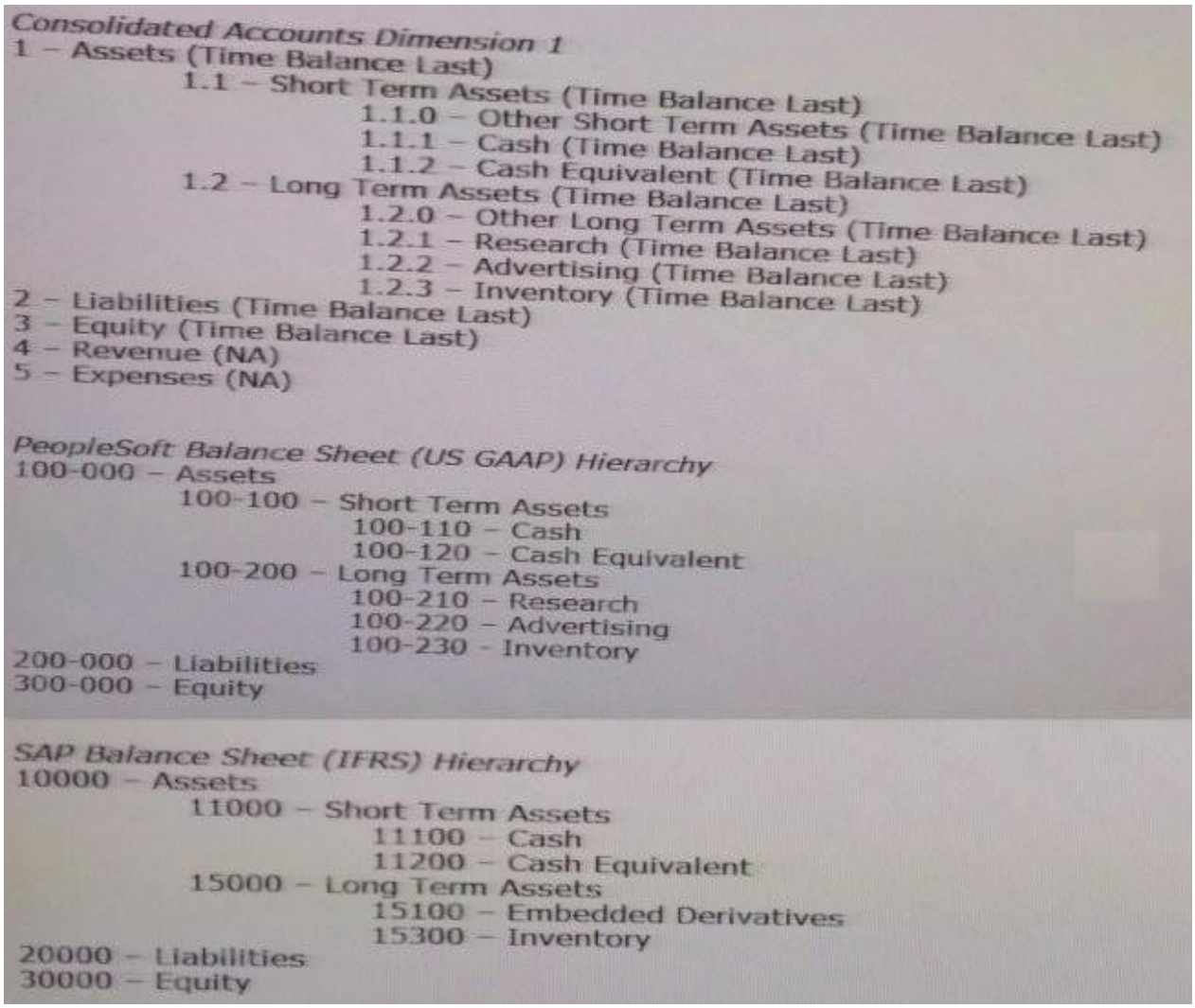
**Explanation:**

Node access-group permissions in Oracle Data Relationship Management are additive: the user receives the permissions granted through each assigned node access group, while each permission remains specific to the operation it authorizes. The Add permission assigned through G_Colas authorizes the user to create child nodes under the Colas node, so the user can add new nodes in the Product category beneath Colas. The Edit permission assigned through G_Diet Cola authorizes changes to the Diet Cola node; therefore, the user can edit that explicitly permitted node. These permissions are not interchangeable: an Add assignment grants the capability to add below the assigned node, whereas Edit applies to modifying the assigned node.

Deleting nodes requires the separate Delete permission. Since no Delete permission is assigned for the Colas node, the user cannot delete nodes below Colas. The Interactive User role permits the user to work interactively in DRM, but node access groups still control which hierarchy operations the user may perform. Oracle’s DRM documentation describes node access groups as the mechanism for assigning secured, node-level access in hierarchies and for controlling permitted maintenance actions. See the Oracle Data Relationship Management documentation library and the Oracle DRM 11.2 documentation.

## QUESTION NO: 6

Per the example:




Your organization has two ERPs: PeopleSoft (PS) and SAP. Each ERP has its own chart of accounts. Each chart of accounts is mapped to a consolidated chart of accounts. The Finance department maintains the consolidated account hierarchy.

Identify the steps in the correct order to manage the mapping of the local chart of accounts in the consolidated chart of accounts structure using DRM hierarchies.

1. Build the consolidated chart of accounts hierarchy.

2. Build the PeopleSoft and SAP chart of accounts hierarchies.

3. Create a property category called CoA Map with two property definitions: PS CoA Mapping and SAP CoA Mapping.

4. Create an alternate Market hierarchy with limb nodes "large", "Medium", and "Small". Insert cities as leaf nodes under the appropriate Market Size nodes.

5. Use the Blend function to merge the PeopleSoft and sap chart of accounts to the correct consolidated account node In the consolidated hierarchy.

6. Drag and drop the PeopleSoft and SAP chart of accounts to the correct consolidated account in the consolidated hierarchy.

**A.** 1, 2, 3, 4, 5, 6

**B.** 3, 1, 2, 4, 6

**C.** 3, 1, 2, 5, 6

**D.** 1, 2, 4, 6

**E.** 4, 3, 1, 2, 5, 6

**ANSWER: C**

**Explanation:**

The correct sequence is **3, 1, 2, 5, 6**. First, the CoA Map property category and its PeopleSoft and SAP mapping properties establish the metadata fields that will hold and identify the relationship between local accounts and consolidated accounts. Creating these definitions before performing hierarchy work ensures that the mapping structure is available consistently as the local and consolidated account content is managed.

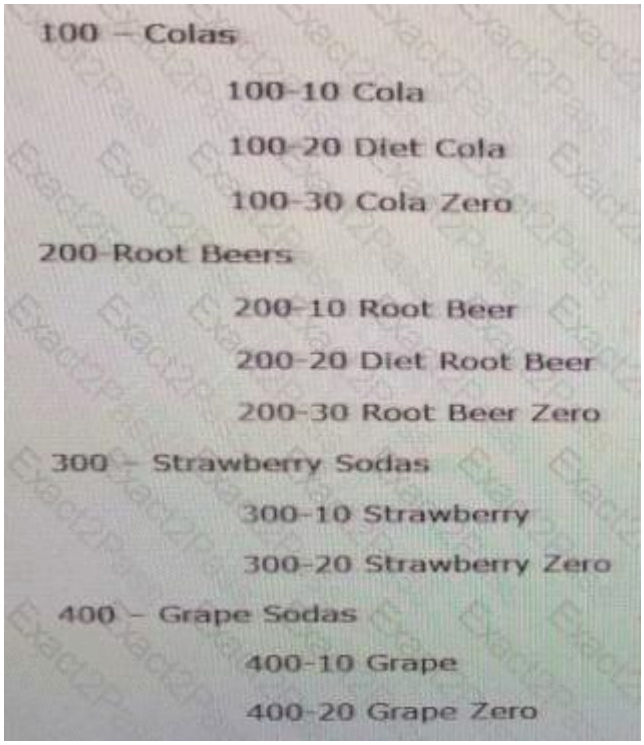
Next, Finance creates the consolidated chart of accounts hierarchy, which provides the governed target structure for reporting and consolidation. The PeopleSoft and SAP chart-of-accounts hierarchies are then built as the local source structures. With all three hierarchies available, the Blend operation is used to merge the local account structures into the appropriate locations in the consolidated hierarchy while preserving the intended hierarchy-management relationship. The final drag-and-drop activity places the PeopleSoft and SAP account branches under their appropriate consolidated account nodes, completing the maintained local-to-consolidated arrangement.

This workflow uses DRM's hierarchy and node-management capabilities to centralize stewardship of the consolidated structure while retaining the distinct local ERP account structures. Oracle DRM documentation is available through the Oracle Data Relationship Management documentation library and Oracle's Data Relationship Management product page.

**QUESTION NO: 7**

Per the following example:

All Products




A user is assigned to two node access group, G_Colas and G_Diet Cola. G_Colas is assigned Add permission for the Colas node in the product hierarchy and the G_Diet Cola group is assigned I dit permission for the Diet Cola node.

He is assigned the Interactive User role. Identify the three true statements.

**A.** The user can add new nodes into the Product category under the "Colas" node.

**B.** The user can edit the Diet Cola node only.

**C.** The user can delete nodes under the "Colas'" node.

**D.** The user only sees the "Diet Cola" node in the Product hierarchy.

**E.** The user cannot delete nodes under the "Colas" node.

**F.** The user may inactivate nodes under the "Colas" node.

**ANSWER: A B E**

**Explanation:**

Node access groups in Oracle Data Relationship Management grant the assigned access privilege within the defined portion of a hierarchy. The Add permission on the *Colas* node authorizes the user to create new child nodes below that node. Therefore, the user can add nodes into the Product category under *Colas*. This permission is particularly useful when responsibility is delegated for maintaining a product branch without granting broad administrative control of the entire hierarchy.

The Edit permission granted through the G_Diet Cola node access group authorizes maintenance of the *Diet Cola* node. With the hierarchy scope shown in the example, this supports editing that node. The Interactive User role enables the user to work interactively in DRM, while the node access-group permissions determine the specific hierarchy areas and maintenance actions available to that user.

Deletion is a distinct node-access privilege. Since the user has not been granted Delete access for the *Colas* branch, deletion under that node is unavailable. Oracle describes node access groups as the mechanism for granting scoped hierarchy access and separately documents the available DRM security and node-management capabilities. See the Oracle Data Relationship Management 11.1.2 documentation library and the Oracle DRM documentation page.

**QUESTION NO: 8**

Identify the three benefits of using an External Connection for several DRM exports that write to the same database environment.

**A.** Multiple exports can use the same target definition.

**B.** A changed target location can be updated in one connection definition.

**C.** Allowed tables can be specified for an external database connection.

**D.** All DRM repository tables become available to export users.

**E.** Export object security is automatically bypassed.

**F.** A baseline version is automatically created before each export.

**ANSWER: A B C**

**Explanation:**

An External Connection provides a reusable definition of a target location, such as a database or network location. Multiple exports can reference the same connection, which avoids repeating connection details in each export definition. If the target endpoint changes, an administrator can update the External Connection rather than editing every dependent export. For external database connections, permitted target tables can be explicitly configured to control where exports are written.

An External Connection does not provide unrestricted access to all repository tables and does not replace DRM export security. It is an integration endpoint definition, not a version-management or workflow object. Oracle describes External Connections in the Oracle Data Relationship Management documentation library.

**QUESTION NO: 9**

Identify the three true statements.

**A.** A version may contain multiple hierarchies.

**B.** A version may contain a single hierarchy with shared nodes.

**C.** A version may contain version-specific node types.

**D.** A version may be assigned validations.

**E.** A version may be assigned exports.

**ANSWER: A B C**

**Explanation:**

In Oracle Data Relationship Management, a version is a controlled working copy of master data that provides the context in which hierarchies, nodes, and their relationships are maintained. A version may contain multiple hierarchies, allowing separate but related organizational, legal-entity, product, or other structures to be managed in the same version. DRM also supports shared nodes within a hierarchy. Consequently, a version may contain a single hierarchy in which a node is shared under more than one parent, enabling alternate rollups while retaining one underlying node and its properties.

Node types are also associated with a version's data model. Version-specific node types allow the version to distinguish and govern categories of nodes, including the properties and behavior appropriate to those nodes in that version. Together, these capabilities let a DRM version represent a complete, governed set of master-data structures and classifications before it is finalized or used downstream. Oracle's DRM documentation describes versions as the containers for hierarchy and node information and covers hierarchy/node-type administration in the DRM user documentation. See the Oracle Data Relationship Management 11.2 documentation and the Oracle DRM User's Guide.

## QUESTION NO: 10

Two normal versions contain independently maintained changes to the same master-data application. The approved changes from one version must be incorporated into the other version without manually recreating each change.

Which DRM feature should be used?

**A.** Blend

**B.** Inactivate

**C.** Generation Export

**D.** Hierarchy Group

**E.** Property Query

**ANSWER: A**

**Explanation:**

**Blend** is correct. DRM Blend is used to combine data changes from one version or hierarchy with another version or hierarchy. It supports controlled incorporation of independently maintained content and is useful when change sets have been developed separately before they are brought together for review or further maintenance.

The blend results should be reviewed and validated so that hierarchy, relationship, and property conflicts can be resolved before the target version is used downstream. See the Oracle Data Relationship Management documentation for version-management operations.

## QUESTION NO: 11

Your budget office is preparing the Entity dimension for the upcoming year with organisation changes. They are working in a version called '"Budget". This version has a Working status until the Entity hierarchy is finalized. The budget office user has

received new information from management that requires him to back out the numerous changes he has made over the last five days. The AllowAsOf system preference is set to True. What is the next step?

**A.** To manually back out the changes in the Budget version.

**B.** To recover the DRM application from a backup.

**C.** To use the copy feature to copy the version and enter the "As Of" date to create the version as of the certain date.

**D.** To use the Create As Of Version feature to create a copy of the version as of a specified date.

E) To use the Copy feature to copy the baseline version to a new version and re-apply changes "As Of" the desired date.

**ANSWER: C**

**Explanation:**

To use the copy feature to copy the version and enter the "As Of" date to create the version as of the certain date is correct because the enabled *AllowAsOf* system preference permits Data Relationship Management to construct a new version from the selected source version's historical state at a specified point in time. The user can select a date and time immediately before the unwanted five days of edits, producing a separate version that reflects the Entity hierarchy as it existed at that moment. This approach preserves the current Budget version and its audit history while giving the budget office a recoverable, usable version from which to continue planning. Since the Budget version remains in Working status, the restored point-in-time version can be reviewed and adjusted before it is finalized or made available for downstream use. The As Of capability relies on DRM's version history and transaction audit information rather than requiring a database or application-level recovery. Oracle describes version management and historical version operations in its Data Relationship Management documentation; see the Oracle Data Relationship Management 11.2 documentation library and the DRM User's Guide.