Workday-Pro-Integrations Clearer Explanation | Workday Workday-Pro-Integrations Valid Test Voucher: Workday Pro Integrations Certification Exam Latest Released

P.S. Free & New Workday-Pro-Integrations dumps are available on Google Drive shared by TrainingQuiz: https://drive.google.com/open?id=1lHPPbDY02NgZb8qyJ_bqzcRhDk8dGsNr

TrainingQuiz provides you with a free demo of Workday Workday-Pro-Integrations Questions so you do not have any doubts about the quality of our exam prep material. Similarly, We also provide free updates up to 365 days after purchasing Workday Pro Integrations Certification Exam dumps questions, so that you always get the latest Workday dumps.

Workday Workday-Pro-Integrations Exam Syllabus Topics:

Topic	Details
Topic 1	• Enterprise Interface Builders: This section of the exam measures the skills of Integration Developers and covers the use of Workday's Enterprise Interface Builder (EIB) to design, deploy, and maintain inbound and outbound integrations. It evaluates the candidate's ability to create templates, configure transformation rules, schedule integrations, and troubleshoot EIB workflows efficiently.
Topic 2	• Integrations: This section of the exam measures the skills of Integration Specialists and covers the full spectrum of integration techniques in Workday. It includes an understanding of core integration architecture, APIs, Workday Studio, and integration system user setup. The focus is on building scalable, maintainable, and secure integrations that ensure seamless system interoperability.
Topic 3	• Calculated Fields: This section of the exam measures the skills of Workday Integration Analysts and covers the creation, configuration, and management of calculated fields used to transform, manipulate, and format data in Workday integrations. It evaluates understanding of field types, dependencies, and logical operations that enable dynamic data customization within integration workflows.

Topic 4	• Reporting: This section of the exam measures the skills of Reporting Analysts and focuses on building, modifying, and managing Workday reports that support integrations. It includes working with report writer tools, custom report types, calculated fields within reports, and optimizing report performance to support automated data exchange.

>> Workday-Pro-Integrations Clearer Explanation <<

Workday - Workday-Pro-Integrations - Workday Pro Integrations Certification Exam –The Best Clearer Explanation

Without doubt, our Workday-Pro-Integrations practice dumps keep up with the latest information and contain the most valued key points that will show up in the real Workday-Pro-Integrations exam. Meanwhile, we can give you accurate and instant suggestion for our customer services know every detail of our Workday-Pro-Integrations Exam Questions. And they are pleased to give guide for 24 hours online. You can get assistant by them as long as you made your inquire.

Workday Pro Integrations Certification Exam Sample Questions (Q38-Q43):

NEW QUESTION # 38
Refer to the following XML to answer the question below.

```
<wd:Get_Job_Profiles_Response xmlns:wd="urn:com.workday/bsvc" wd:version="v43.0">
    <wd:Response_Data>
        <wd:Job_Profile>
            <wd:Job_Profile_Reference>
                <wd:ID wd:type="WID">174c31eca2f24ed9b6174ca7d2aeb88c</wd:ID>
                <wd:ID wd:type="Job_Profile_ID">Senior_Benefits_Analyst</wd:ID>
            </wd:Job_Profile_Reference>
            <wd:Job_Profile_Data>
                <wd:Job_Code>Senior Benefits Analyst</wd:Job_Code>
                <wd:Effective_Date>2024-05-15</wd:Effective_Date>
                <wd:Education_Qualification_Replacement_Data>
                    <wd:Degree_Reference>
                        <wd:ID wd:type="WID">61303c9b1d094d4a73166ad39caebce</wd:ID>
                        <wd:ID wd:type="Degree_ID">MBA</wd:ID>
                    </wd:Degree_Reference>
                    <wd:Field_Of_Study_Reference>
                        <wd:ID wd:type="WID">62e42dfd4b8c49b5842114f67369a96f</wd:ID>
                        <wd:ID wd:type="Field_Of_Study_ID">Economics</wd:ID>
                    </wd:Field_Of_Study_Reference>
                    <wd:Required>0</wd:Required>
                </wd:Education_Qualification_Replacement_Data>
                <wd:Education_Qualification_Replacement_Data>
                    <wd:Degree_Reference>
                        <wd:ID wd:type="WID">8db9b8e5f53c4cbdb7f7a984c6afde28</wd:ID>
                        <wd:ID wd:type="Degree_ID">B_S</wd:ID>
                    </wd:Degree_Reference>
                    <wd:Required>1</wd:Required>
                </wd:Education_Qualification_Replacement_Data>
            </wd:Job_Profile_Data>
        </wd:Job_Profile>
    </wd:Response_Data>
</wd:Get_Job_Profiles_Response>
```


You are an integration developer and need to write XSLT to transform the output of an EIB which is making a request to the Get Job Profiles web service operation. The root template of your XSLT matches on the <wd:
Get_Job_Profiles_Response> element. This root template then applies a template against <wd:Job_Profile>.
What XPath syntax would be used to select the value of the wd:Job_Code element when the <xsl:value-of> element is placed within the template which matches on <wd:Job_Profile>?

- A. wd:Job_Profile_Reference/wd:ID[@wd:type='Job_Profile_ID']
- B. wd:Job_Profile_Data[@wd:Job_Code]
- C. wd:Job_Profile/wd:Job_Profile_Data/wd:Job_Code
- D. wd:Job_Profile_Data/wd:Job_Code

Answer: D

Explanation:
As an integration developer working with Workday, you are tasked with transforming the output of an Enterprise Interface Builder

(EIB) that calls the Get_Job_Profiles web service operation. The provided XML shows the response from this operation, and you need to write XSLT to select the value of the <wd:
Job_Code> element. The root template of your XSLT matches on <wd:Get_Job_Profiles_Response> and applies a template to <wd:Job_Profile>. Within this template, you use the <xsl:value-of> element to extract the <wd:Job_Code> value. Let's analyze the XML structure, the requirement, and each option to determine the correct XPath syntax.
Understanding the XML and Requirement
The XML snippet provided is a SOAP response from the Get_Job_Profiles web service operation in Workday, using the namespace xmlns:wd="urn:com.workday/bsvc" and version wd:version="v43.0". Key elements relevant to the question include:
* The root element is <wd:Get_Job_Profiles_Response>.
* It contains <wd:Response_Data>, which includes <wd:Job_Profile> elements.
* Within <wd:Job_Profile>, there are:
* <wd:Job_Profile_Reference>, which contains <wd:ID> elements (e.g., a Job_Profile_ID).
* <wd:Job_Profile_Data>, which contains <wd:Job_Code> with the value
Senior_Benefits_Analyst.
The task is to select the value of <wd:Job_Code> (e.g., "Senior_Benefits_Analyst") using XPath within an XSLT template that matches <wd:Job_Profile>. The <xsl:value-of> element outputs the value of the selected node, so you need the correct XPath path from the <wd:Job_Profile> context to <wd:Job_Code>.
Analysis of Options
Let's evaluate each option based on the XML structure and XPath syntax rules:
* Option A: wd:Job_Profile/wd:Job_Profile_Data/wd:Job_Code
* This XPath starts from wd:Job_Profile and navigates to wd:Job_Profile_Data/wd:Job_Code.
However, in the XML, <wd:Job_Profile> is the parent element, and <wd:Job_Profile_Data> is a direct child containing <wd:Job_Code>. The path wd:Job_Profile/wd:Job_Profile_Data/wd:
Job_Code is technically correct in terms of structure, as it follows the hierarchy:
* <wd:Job_Profile> # <wd:Job_Profile_Data> # <wd:Job_Code>.
* However, since the template matches <wd:Job_Profile>, the context node is already <wd:
Job_Profile>. You don't need to include wd:Job_Profile/ at the beginning of the XPath unless navigating from a higher level. Starting directly with wd:Job_Profile_Data/wd:Job_Code (Option C) is more concise and appropriate for the context. This option is technically valid but redundant and less efficient, making it less preferred compared to Option C.
* Option B: wd:Job_Profile_Data[@wd:Job_Code]
* This XPath uses an attribute selector ([@wd:Job_Code]) to filter <wd:Job_Profile_Data> based on an attribute named wd:Job_Code. However, examining the XML, <wd:Job_Profile_Data> does not have a wd:Job_Code attribute-it has a child element <wd:Job_Code> with the value
"Senior_Benefits_Analyst." The [@attribute] syntax is used for attributes, not child elements, so this XPath is incorrect. It would not select the <wd:Job_Code> value and would likely return no results or an error. This option is invalid.
* Option C: wd:Job_Profile_Data/wd:Job_Code
* This XPath starts from wd:Job_Profile_Data (a direct child of <wd:Job_Profile>) and navigates to wd:Job_Code. Since the template matches <wd:Job_Profile>, the contextnode is <wd:
Job_Profile>, and wd:Job_Profile_Data/wd:Job_Code correctly points to the <wd:Job_Code> element within <wd:Job_Profile_Data>. This path is:
* Concise and appropriate for the context.
* Directly selects the value "Senior_Benefits_Analyst" when used with <xsl:value-of>.
* Matches the XML structure, as <wd:Job_Profile_Data> contains <wd:Job_Code> as a child.
* This is the most straightforward and correct option for selecting the <wd:Job_Code> value within the <wd:Job_Profile> template.
* Option D: wd:Job_Profile_Reference/wd:ID[@wd:type='Job_Profile_ID']
* This XPath navigates to <wd:Job_Profile_Reference> (a child of <wd:Job_Profile>) and then to <wd:ID> with an attribute wd:type="Job_Profile_ID". In the XML, <wd:Job_Profile_Reference> contains:
* <wd:ID wd:type="WID">1740d3eca2f2ed9b6174ca7d2ae88c8c</wd:ID>
* <wd:ID wd:type="Job_Profile_ID">Senior_Benefits_Analyst</wd:ID>
* The XPath wd:Job_Profile_Reference/wd:ID[@wd:type='Job_Profile_ID'] selects the <wd:ID> element with wd:type="Job_Profile_ID", which has the value "Senior_Benefits_Analyst." However, this is not the <wd:Job_Code> value-the <wd:Job_Code> is a separate element under
<wd:Job_Profile_Data>, not <wd:Job_Profile_Reference>. The question specifically asks for the <wd:Job_Code> value, so this option is incorrect, as it selects a different piece of data (the job profile ID, not the job code).
Why Option C is Correct
Option C, wd:Job_Profile_Data/wd:Job_Code, is the correct XPath syntax because:
* It starts from the context node <wd:Job_Profile> (as the template matches this element) and navigates to <wd:Job_Profile_Data/wd:Job_Code>, which directly selects the <wd:Job_Code> element's value ("Senior_Benefits_Analyst").
* It is concise and aligns with standard XPath navigation in XSLT, avoiding unnecessary redundancy (unlike Option A) or incorrect attribute selectors (unlike Option B).
* It matches the XML structure, where <wd:Job_Profile_Data> is a child of <wd:Job_Profile> and contains <wd:Job_Code> as a

child.
* When used with <xsl:value-of select="wd:Job_Profile_Data/wd:Job_Code"/> in the template, it outputs the job code value, fulfilling the requirement.
Practical Example in XSLT
Here's how this might look in your XSLT:
xml
WrapCopy
<xsl:template match="wd:Job_Profile">
<xsl:value-of select="wd:Job_Profile_Data/wd:Job_Code"/>
</xsl:template>
This would output "Senior_Benefits_Analyst" for the <wd:Job_Code> element in the XML.
Verification with Workday Documentation
The Workday Pro Integrations Study Guide and SOAP API Reference (available via Workday Community) detail the structure of the Get_Job_Profiles response and how to use XPath in XSLT for transformations. The XML structure shows <wd:Job_Profile_Data> as the container for job profile details, including <wd:
Job_Code>. The guide emphasizes using relative XPath paths within templates to navigate from the matched element (e.g., <wd:Job_Profile>) to child elements like <wd:Job_Profile_Data/wd:Job_Code>.
Workday Pro Integrations Study Guide References
* Section: XSLT Transformations in EIBs- Describes using XSLT to transform web service responses, including selecting elements with XPath.
* Section: Workday Web Services- Details the Get_Job_Profiles operation and its XML output structure, including <wd:Job_Profile_Data> and <wd:Job_Code>.
* Section: XPath Syntax- Explains how to navigate XML hierarchies in Workday XSLT, using relative paths like wd:Job_Profile_Data/wd:Job_Code from a <wd:Job_Profile> context.
* Workday Community SOAP API Reference - Provides examples of XPath navigation for Workday web service responses.
Option C is the verified answer, as it correctly selects the <wd:Job_Code> value using the appropriate XPath syntax within the <wd:Job_Profile> template context.

NEW QUESTION # 39
What attribute(s) can go into the xsl:stylesheet element?

- A. Namespaces & Encoding
- B. XSLT Version & Namespaces
- C. XML Version & Namespaces
- D. XSLT Version & Encoding

Answer: B

Explanation:
The <xsl:stylesheet> element is the root element in an XSLT document. It must include:
XSLT Version - This defines the XSLT specification version being used (e.g., version="1.0" or version="2.0").
Namespaces - XSLT operates within an XML namespace (xmlns:xsl="http://www.w3.org/1999/XSL/Transform"), which is required to define the transformation rules.
Breakdown of Answer Choices:
A . XSLT Version & Namespaces □ (Correct)
The <xsl:stylesheet> element requires both the XSLT version and the namespace declaration for proper execution.
Example:
xml
CopyEdit
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"> B . XSLT Version & Encoding □ (Incorrect)
Encoding (encoding="UTF-8") is a property of the XML declaration (<?xml version="1.0" encoding="UTF-8"?>), not an attribute of <xsl:stylesheet>.
C . XML Version & Namespaces □ (Incorrect)
XML version (<?xml version="1.0"?>) is part of the XML prolog, not an attribute of <xsl:stylesheet>.
D . Namespaces & Encoding □ (Incorrect)
Encoding is not an attribute of <xsl:stylesheet>.
Final Correct Syntax:
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"> This ensures that the XSLT file is processed correctly.
Workday Pro Integrations Study Guide Reference:

ReportWriterTraining.pdf - Chapter 9: Working With XML and XSLT covers XSLT basics, including the required attributes for <xsl:stylesheet> .
Workday_Advanced_Business_Process_part_2.pdf - Chapter 5: Web Services and Integrations details how Workday uses XSLT for transformations .

NEW QUESTION # 40
A calculated field used as a field override in a Connector is not appearing in the output. Assuming the field has a value, what could cause this to occur?

- A. Access not provided to all instances of calculated field.
- B. Access not provided to all fields in the calculated field.
- C. Access not provided to calculated field data source.
- D. Access not provided to Connector calculated field web service.

Answer: B

NEW QUESTION # 41
Refer to the following XML to answer the question below.

```
<wd:Get_Job_Profiles_Response xmlns:wd="urn:com.workday/bsvc" wd:version="v43.0">
    <wd:Response_Data>
        <wd:Job_Profile>
            <wd:Job_Profile_Reference>
                <wd:ID wd:type="WID">174c31eca2f24ed9b6174ca7d2aeb88c</wd:ID>
                <wd:ID wd:type="Job_Profile_ID">Senior_Benefits_Analyst</wd:ID>
            </wd:Job_Profile_Reference>
            <wd:Job_Profile_Data>
                <wd:Job_Code>Senior Benefits Analyst</wd:Job_Code>
                <wd:Effective_Date>2024-05-15</wd:Effective_Date>
                <wd:Education_Qualification_Replacement_Data>
                    <wd:Degree_Reference>
                        <wd:ID wd:type="WID">61393c9b1d094d44a73166ad39caebce</wd:ID>
                        <wd:ID wd:type="Degree_ID">MBA</wd:ID>
                    </wd:Degree_Reference>
                    <wd:Field_Of_Study_Reference>
                        <wd:ID wd:type="WID">62e42dfd4b8c49b5842114f67369a96f</wd:ID>
                        <wd:ID wd:type="Field_Of_Study_ID">Economics</wd:ID>
                    </wd:Field_Of_Study_Reference>
                    <wd:Required>0</wd:Required>
                </wd:Education_Qualification_Replacement_Data>
                <wd:Education_Qualification_Replacement_Data>
                    <wd:Degree_Reference>
                        <wd:ID wd:type="WID">8db9b8e5f53c4cbdb7f7a984c6afde28</wd:ID>
                        <wd:ID wd:type="Degree_ID">B_S</wd:ID>
                    </wd:Degree_Reference>
                    <wd:Required>1</wd:Required>
                </wd:Education_Qualification_Replacement_Data>
            </wd:Job_Profile_Data>
        </wd:Job_Profile>
    </wd:Response_Data>
</wd:Get_Job_Profiles_Response>
```


You are an integration developer and need to write XSLT to transform the output of an EIB which is making a request to the Get Job Profiles web service operation. The root template of your XSLT matches on the <wd:Get_Job_Profiles_Response> element. This root template then applies templates against <wd:Job_Profile>. What XPath syntax would be used to select the value of the ID element which has a wd:type attribute named Job_Profile_ID when the <xsl:value-of> element is placed within the template which matches on <wd:Job_Profile>?

- A. wd:Job_Profile_Reference/wd:ID/[@wd:type='Job_Profile_ID']

- B. wd:Job_Profile_Reference/wd:ID/wd:type='Job_Profile_ID'
- C. wd:Job_Profile_Reference/wd:ID[@wd:type='Job_Profile_ID']
- D. wd:Job_Profile_Reference/wd:ID/@wd:type='Job_Profile_ID'

Answer: C

Explanation:
As an integration developer working with Workday, you are tasked with transforming the output of an Enterprise Interface Builder (EIB) that calls the Get_Job_Profiles web service operation. The provided XML shows the response from this operation, and you need to write XSLT to select the value of the <wd:ID> element where the wd:type attribute equals "Job_Profile_ID." The root template of your XSLT matches on <wd:Get_Job_Profiles_Response> and applies templates to <wd:Job_Profile>. Within this template, you use the <xsl:value-of> element to extract the value. Let's analyze the XML structure, the requirement, and each option to determine the correct XPath syntax.
Understanding the XML and Requirement
The XML snippet provided is a SOAP response from the Get_Job_Profiles web service operation in Workday, using the namespace xmlns:wd="urn:com.workday/bsvc" and version wd:version="v43.0". Key elements relevant to the question include:
The root element is <wd:Get_Job_Profiles_Response>.
It contains <wd:Response_Data>, which includes <wd:Job_Profile> elements.
Within <wd:Job_Profile>, there is <wd:Job_Profile_Reference>, which contains multiple <wd:ID> elements, each with a wd:type attribute:
<wd:ID wd:type="WID">1740d3eca2f2ed9b6174ca7d2ae88c8c</wd:ID>
<wd:ID wd:type="Job_Profile_ID">Senior_Benefits_Analyst</wd:ID>
The task is to select the value of the <wd:ID> element where wd:type="Job_Profile_ID" (e.g., "Senior_Benefits_Analyst") using XPath within an XSLT template that matches <wd:Job_Profile>. The <xsl:value-of> element outputs the value of the selected node, so you need the correct XPath path from the <wd:Job_Profile> context to the specific <wd:ID> element with the wd:type attribute value "Job_Profile_ID." Analysis of Options Let's evaluate each option based on the XML structure and XPath syntax rules:
Option A: wd:Job_Profile_Reference/wd:ID/wd:type='Job_Profile_ID'
This XPath attempts to navigate from wd:Job_Profile_Reference to wd:ID, then to wd:type='Job_Profile_ID'. However, there are several issues:
wd:type='Job_Profile_ID' is not valid XPath syntax. In XPath, to filter based on an attribute value, you use the attribute selector [@attribute='value'], not a direct comparison like wd:type='Job_Profile_ID'.
wd:type is an attribute of <wd:ID>, not a child element or node. This syntax would not select the <wd:ID> element itself but would be interpreted as trying to match a nonexistent child node or property, resulting in an error or no match.
This option is incorrect because it misuses XPath syntax for attribute filtering.
Option B: wd:Job_Profile_Reference/wd:ID/@wd:type='Job_Profile_ID'
This XPath navigates to wd:Job_Profile_Reference/wd:ID and then selects the @wd:type attribute, comparing it to "Job_Profile_ID" with =@wd:type='Job_Profile_ID'. However:
The =@wd:type='Job_Profile_ID' syntax is invalid in XPath. To filter based on an attribute value, you use [@wd:type='Job_Profile_ID'] as a predicate, not an equality comparison in this form.
This XPath would select the wd:type attribute itself (e.g., the string "Job_Profile_ID"), not the value of the <wd:ID> element. Since <xsl:value-of> expects a node or element value, selecting an attribute directly would not yield the desired "Senior_Benefits_Analyst" value.
This option is incorrect due to the invalid syntax and inappropriate selection of the attribute instead of the element value.
Option C: wd:Job_Profile_Reference/wd:ID[@wd:type='Job_Profile_ID']
This XPath navigates from wd:Job_Profile_Reference to wd:ID and uses the predicate [@wd:type='Job_Profile_ID'] to filter for <wd:ID> elements where the wd:type attribute equals "Job_Profile_ID." In the XML, <wd:Job_Profile_Reference> contains:
<wd:ID wd:type="WID">1740d3eca2f2ed9b6174ca7d2ae88c8c</wd:ID>
<wd:ID wd:type="Job_Profile_ID">Senior_Benefits_Analyst</wd:ID>
The predicate [@wd:type='Job_Profile_ID'] selects the second <wd:ID> element, whose value is "Senior_Benefits_Analyst." Since the template matches <wd:Job_Profile>, and <wd:Job_Profile_Reference> is a direct child of <wd:Job_Profile>, this path is correct: <wd:Job_Profile> → <wd:Job_Profile_Reference> → <wd:ID[@wd:type='Job_Profile_ID']>.
When used with <xsl:value-of select="wd:Job_Profile_Reference/wd:ID[@wd:type='Job_Profile_ID']"/>, it outputs "Senior_Benefits_Analyst," fulfilling the requirement.
This option is correct because it uses proper XPath syntax for attribute-based filtering and selects the desired <wd:ID> value.
Option D: wd:Job_Profile_Reference/wd:ID/[@wd:type='Job_Profile_ID']
This XPath is similar to Option C but includes an extra forward slash before the predicate: wd:ID/[@wd:type='Job_Profile_ID']. In XPath, predicates like [@attribute='value'] are used directly after the node name (e.g., wd:ID[@wd:type='Job_Profile_ID']), not separated by a slash. The extra slash is syntactically incorrect and would result in an error or no match, as it implies navigating to a child node that doesn't exist.
This option is incorrect due to the invalid syntax.
Why Option C is Correct

Option C, wd:Job_Profile_Reference/wd:ID[@wd:type='Job_Profile_ID'], is the correct XPath syntax because:
It starts from the context node <wd:Job_Profile> (as the template matches this element) and navigates to <wd:Job_Profile_Reference/wd:ID>, using the predicate [@wd:type='Job_Profile_ID'] to filter for the <wd:ID> element with wd:type="Job_Profile_ID".
It correctly selects the value "Senior_Benefits_Analyst," which is the content of the <wd:ID> element where wd:type="Job_Profile_ID".
It uses standard XPath syntax for attribute-based filtering, aligning with Workday's XSLT implementation for web service responses.
When used with <xsl:value-of>, it outputs the required value, fulfilling the question's requirement.
Practical Example in XSLT
Here's how this might look in your XSLT:
<xsl:template match="wd:Job_Profile">
<xsl:value-of select="wd:Job_Profile_Reference/wd:ID[@wd:type='Job_Profile_ID']"/>
</xsl:template>
This would output "Senior_Benefits_Analyst" for the <wd:ID> element with wd:type="Job_Profile_ID" in the XML.
Verification with Workday Documentation
The Workday Pro Integrations Study Guide and SOAP API Reference (available via Workday Community) detail the structure of the Get_Job_Profiles response and how to use XPath in XSLT for transformations. The XML structure shows <wd:Job_Profile_Reference> containing <wd:ID> elements with wd:type attributes, and the guide emphasizes using predicates like [@wd:type='value'] to filter based on attributes. This is a standard practice for navigating Workday web service responses.
Workday Pro Integrations Study Guide Reference
Section: XSLT Transformations in EIBs - Describes using XSLT to transform web service responses, including selecting elements with XPath and attribute predicates.
Section: Workday Web Services - Details the Get_Job_Profiles operation and its XML output structure, including <wd:Job_Profile_Reference> and <wd:ID> with wd:type attributes.
Section: XPath Syntax - Explains how to use predicates like [@wd:type='Job_Profile_ID'] for attribute-based filtering in Workday XSLT.
Workday Community SOAP API Reference - Provides examples of XPath navigation for Workday web service responses, including attribute selection.
Option C is the verified answer, as it correctly selects the <wd:ID> value with wd:type="Job_Profile_ID" using the appropriate XPath syntax within the <wd:Job_Profile> template context.

NEW QUESTION # 42
This is the XML file generated from a Core Connector; Positions integration.

When performing an XSLT Transformation on the Core Connector: Positions XML output file, you want to show a hyperlink of positions that are not available for hiring as an entry in the Message tab.
What are all the needed ETV items to meet the above requirements?

- A. 

- B. 
- C. 
- D. 

Answer: B

Explanation:
In Workday integrations, the Extension for Transformation and Validation (ETV) framework is used within XSLT transformations to apply validation and formatting rules to XML data, such as the output from a Core Connector (e.g., Positions integration). In this scenario, you need to perform an XSLT transformation on the Core Connector: Positions XML output file to display a hyperlink for positions that are not available for hiring as an entry in the Message tab. This requires configuring ETV attributes to ensure the data is present and correctly targeted for the hyperlink.
Here's why option B is correct:
Requirement Analysis: The requirement specifies showing a hyperlink for positions "not available for hiring." In the provided XML, the ps:Available_For_Hire field under ps:Position_Data indicates whether a position is available for hire (e.g., <ps:Available_For_Hire>true</ps:Available_For_Hire>). For positions where this is false, you need to create a message (hyperlink) in the Message tab, which typically requires linking to a Workday ID (WID) or other identifier.
ETV Attributes:
etv:required="true": This ensures that the ps:WID value under ps:Additional_Information is mandatory for the transformation. If the WID is missing, the transformation will fail or generate an error, ensuring that the hyperlink can be created only for valid positions with an associated WID.
etv:target="[ps:Additional_Information/ps:WID]": This specifies that the target of the transformation (e.g., the hyperlink) should be the WID value found at ps:Additional_Information/ps:WID in the XML. This WID can be used to construct a hyperlink to the position in Workday, meeting the requirement to show a hyperlink for positions not available for hiring.
Context in XML: The XML shows ps:Additional_Information containing ps:WID (e.g., <ps:WID>73bd4d8562e04b1820f55818467905b</ps:WID>), which is a unique identifier for the position. By targeting this WID with etv:target, you ensure the hyperlink points to the correct position record in Workday when ps:Available_For_Hire is false.
Why not the other options?
A .
etv:minLength="0"
etv:targetWID="[ps:Additional_Information/ps:WID]"
etv:minLength="0" allows the WID to be empty or have zero length, which contradicts the need for a valid WID to create a hyperlink. It does not ensure the data is present, making it unsuitable. Additionally, etv:targetWID is not a standard ETV attribute; the correct attribute is etv:target, making this option incorrect.
C .
etv:minLength="0"
etv:target="[ps:Additional_Information/ps:WID]"
Similar to option A, etv:minLength="0" allows the WID to be empty, which does not meet the requirement for a mandatory WID to create a hyperlink. This makes it incorrect, as the hyperlink would fail if the WID is missing.
D .
etv:required="true"
etv:targetWID="[ps:Additional_Information/ps:WID]"
While etv:required="true" ensures the WID is present, etv:targetWID is not a standard ETV attribute. The correct attribute is etv:target, making this option syntactically incorrect and unsuitable for the transformation.
To implement this in XSLT for a Workday integration:
Use the ETV attributes from option B (etv:required="true" and etv:target="[ps:Additional_Information/ps:WID]") within your XSLT template to validate and target the ps:WID for positions where ps:Available_For_Hire is false. This ensures the transformation generates a valid hyperlink in the Message tab, linking to the position's WID in Workday.
:
Workday Pro Integrations Study Guide: Section on "ETV in XSLT Transformations" - Details the use of ETV attributes like required and target for validating and targeting data in Workday XML, including handling identifiers like WID for hyperlinks.

Workday Core Connector and EIB Guide: Chapter on "XML Transformations" - Explains how to use ETV attributes in XSLT to process position data, including creating messages or hyperlinks based on conditions like Available_For_Hire.
Workday Integration System Fundamentals: Section on "ETV for Message Generation" - Covers applying ETV attributes to generate hyperlinks in the Message tab, ensuring data integrity and correct targeting of Workday identifiers like WID.

NEW QUESTION # 43
.....

You always need actual and updated Workday-Pro-Integrations exam questions to prepare the test successfully in less time. If you don't study with real Workday Pro Integrations Certification Exam (Workday-Pro-Integrations) questions, you will ultimately fail and waste your money and time. To save yourself from this loss, you just need to prepare with updated Workday Pro Integrations Certification Exam (Workday-Pro-Integrations) exam questions of TrainingQuiz.