Workday-Pro-Integrations Practice Questions

Shared by Snow on 24-08-2026

For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

You are creating an outbound connector using the Core Connector: Job Postings template. The vendor has provided the following specification for worker subtype values:

Internal	External
Seasonal (Fixed)	S
Regular	R
Contractor	C
Consultant	C
Any Other Value should be assigned a "U"	

The vendor has also requested that any output file have the following format "CC_Job_Postings_dd-mm-yy_#.xml". Where the dd is the current day at runtime, mm is the current month at runtime, yy is the last two digits of the current year at runtime, and # is the current value of the sequencer at runtime. What configuration step(s) must you complete to meet the vender requirements?

Options:

A- * Enable the Sequence Generator Field Attribute * Configure the Sequence Generator * Configure the Worker Sub Type Integration Mapping leaving the default value blank
B- * Enable the Integration Mapping Field Attribute * Configure the Worker Sub Type Integration Mapping leaving the default value blank * Configure the Sequence Generator
C- * Enable the Integration Mapping Integration Service * Configure the Worker Sub Type Integration Mapping and include a default value of 'U' * Configure the Sequence Generator
D- * Enable the Sequence Generator Integration Service * Configure the Sequence Generator * Configure the Worker Sub Type Integration Mapping and include a default value of 'U'

Answer:

D

Explanation:

This question involves configuring an outbound connector using the Core Connector: Job Postings template in Workday Pro Integrations. We need to meet two specific vendor requirements:

Map worker subtype values according to the provided table (e.g., Seasonal (Fixed) = 'S',

Regular = 'R', Contractor = 'C', Consultant = 'C', and any other value = 'U').

Format the output file name as 'CC_Job_Postings_dd-mm-yy_#.xml', where:

'dd' is the current day at runtime,

'mm' is the current month at runtime,

'yy' is the last two digits of the current year at runtime,

'#' is the current value of the sequencer at runtime.

Let's break down the requirements and evaluate each option to determine the correct configuration steps.

Understanding the Requirements

1. Worker Subtype Mapping

The vendor provides a table for worker subtype values:

Internal Seasonal (Fixed) maps to 'S'

Internal Regular maps to 'R'

Internal Contractor maps to 'C'

Internal Consultant maps to 'C'

Any other value should be assigned 'U'

In Workday, worker subtypes are typically part of the worker data, and for integrations, we use integration mappings to transform these values into the format required by the vendor. The integration mapping allows us to define how internal Workday values (e.g., worker subtypes) map to external values (e.g., 'S', 'R', 'C', 'U'). If no specific mapping exists for a value, we need to set a default value of 'U' for any unmatched subtypes, as specified.

This mapping is configured in the integration system's 'Integration Mapping' or 'Field Mapping' settings, depending on the template. For the Core Connector: Job Postings, we typically use the 'Integration Mapping' feature to handle data transformations, including setting default values for unmapped data.

2. Output File Name Format

The vendor requires the output file to be named 'CC_Job_Postings_dd-mm-yy_#.xml', where:

'CC_Job_Postings' is a static prefix,

'dd-mm-yy' represents the current date at runtime (day, month, last two digits of the year),

'#' is the current value from a sequence generator (sequencer) at runtime.

In Workday, file names for integrations are configured in the 'File Utility' or 'File Output' settings of the integration. To achieve this format:

The date portion ('dd-mm-yy') can be dynamically generated using Workday's date functions or runtime variables, often configured in the File Utility's 'Filename' field with a 'Determine Value at Runtime' setting.

The sequence number ('#') requires a sequence generator, which is enabled and configured to provide a unique incrementing number for each file. Workday uses the 'Sequence Generator' feature for this purpose, typically accessed via the 'Create ID Definition / Sequence Generator' task.

The Core Connector: Job Postings template supports these configurations, allowing us to set filename patterns in the integration's setup.

Evaluating Each Option

Let's analyze each option step by step, ensuring alignment with Workday Pro Integrations best practices and the vendor's requirements.

Option A:

* Enable the Sequence Generator Field Attribute

* Configure the Sequence Generator

* Configure the Worker Sub Type Integration Mapping leaving the default value blank

Analysis:

Sequence Generator Configuration: Enabling the 'Sequence Generator Field Attribute' and configuring the sequence generator is partially correct for the file name's '#' (sequencer) requirement. However, 'Sequence Generator Field Attribute' is not a standard term in Workday; it might refer to enabling a sequence generator in a field mapping, but this is unclear and likely incorrect. Sequence generators are typically enabled as an 'Integration Service' or configured in the File Utility, not as a field attribute.

Worker Subtype Mapping: Configuring the worker subtype integration mapping but leaving the default value blank is problematic. The vendor requires any unmapped value to be 'U,' so leaving it blank would result in missing or null values, failing to meet the requirement.

Date in Filename: This option doesn't mention configuring the date ('dd-mm-yy') in the filename, which is critical for the 'CC_Job_Postings_dd-mm-yy_#.xml' format.

Conclusion: This option is incomplete and incorrect because it doesn't address the default 'U' for unmapped subtypes and lacks date configuration for the filename.

Option B:

* Enable the Integration Mapping Field Attribute

* Configure the Worker Sub Type Integration Mapping leaving the default value blank

* Configure the Sequence Generator

Analysis:

Sequence Generator Configuration: Configuring the sequence generator addresses the '#' (sequencer) in the filename, which is correct for the file name requirement.

Worker Subtype Mapping: Similar to Option A, leaving the default value blank for the worker subtype mapping fails to meet the vendor's requirement for 'U' as the default for unmapped values. This would result in errors or null outputs, which is unacceptable.

Date in Filename: Like Option A, there's no mention of configuring the date ('dd-mm-yy') in the filename, making this incomplete for the full file name format.

Integration Mapping Field Attribute: This term is ambiguous. Workday uses 'Integration Mapping' or 'Field Mapping' for data transformations, but 'Field Attribute' isn't standard for enabling mappings. This suggests a misunderstanding of Workday's configuration.

Conclusion: This option is incomplete and incorrect due to the missing default 'U' for worker subtypes and lack of date configuration for the filename.

Option C:

* Enable the Integration Mapping Integration Service

* Configure the Worker Sub Type Integration Mapping and include a default value of 'U'

* Configure the Sequence Generator

Analysis:

Sequence Generator Configuration: Configuring the sequence generator is correct for the '#' (sequencer) in the filename, addressing part of the file name requirement.

Worker Subtype Mapping: Including a default value of 'U' for the worker subtype mapping aligns perfectly with the vendor's requirement for any unmapped value to be 'U.' This is a strong point.

Date in Filename: This option doesn't mention configuring the date ('dd-mm-yy') in the filename, which is essential for the 'CC_Job_Postings_dd-mm-yy_#.xml' format. Without this, the file name requirement isn't fully met.

Integration Mapping Integration Service: Enabling the 'Integration Mapping Integration Service' is vague. Workday doesn't use this exact term; instead, integration mappings are part of the integration setup, not a separate service. This phrasing suggests confusion or misalignment with Workday terminology.

Conclusion: This option is partially correct (worker subtype mapping) but incomplete due to the missing date configuration for the filename and unclear terminology.

Option D:

* Enable the Sequence Generator Integration Service

* Configure the Sequence Generator

* Configure the Worker Sub Type Integration Mapping and include a default value of 'U'

Analysis:

Sequence Generator Configuration: Enabling the 'Sequence Generator Integration Service' and configuring the sequence generator addresses the '#' (sequencer) in the filename. While 'Sequence Generator Integration Service' isn't a standard term, it likely refers to enabling and configuring the sequence generator functionality, which is correct. In Workday, this is done via the 'Create ID Definition / Sequence Generator' task and linked in the File Utility.

Worker Subtype Mapping: Configuring the worker subtype integration mapping with a default value of 'U' meets the vendor's requirement for any unmapped value, ensuring 'S,' 'R,' 'C,' or 'U' is output as specified in the table. This is accurate and aligns with Workday's integration mapping capabilities.

Date in Filename: Although not explicitly mentioned in the steps, Workday's Core Connector: Job Postings template and File Utility allow configuring the filename pattern, including dynamic date values ('dd-mm-yy'). The filename 'CC_Job_Postings_dd-mm-yy_#.xml' can be set in the File Utility's 'Filename' field with 'Determine Value at Runtime,' using date functions and the sequence generator. This is a standard practice and implied in the configuration, making this option complete.

Conclusion: This option fully addresses both requirements: worker subtype mapping with 'U' as the default and the file name format using the sequence generator and date. The terminology ('Sequence Generator Integration Service') is slightly non-standard but interpretable as enabling/configuring the sequence generator, which is correct in context.

Final Verification

To confirm, let's summarize the steps for Option D and ensure alignment with Workday Pro Integrations:

Enable the Sequence Generator Integration Service: This likely means enabling and configuring the sequence generator via the 'Create ID Definition / Sequence Generator' task, then linking it to the File Utility for the '#' in the filename.

Configure the Sequence Generator: Set up the sequence generator to provide incremental numbers, ensuring each file has a unique '#' value.

Configure the Worker Sub Type Integration Mapping with a default value of 'U': Use the integration mapping to map Internal Seasonal (Fixed) to 'S,' Regular to 'R,' Contractor to 'C,' Consultant to 'C,' and set 'U' as the default for any other value. This is done in the integration's mapping configuration.

Filename Configuration (Implied): In the File Utility, set the filename to 'CC_Job_Postings_dd-mm-yy_#.xml,' where 'dd-mm-yy' uses Workday's date functions (e.g., %d-%m-%y) and '#' links to the sequence generator.

This matches Workday's documentation and practices for the Core Connector: Job Postings template, ensuring both requirements are met.

Why Not the Other Options?

Options A and B fail because they leave the default worker subtype value blank, not meeting the 'U' requirement.

Option C fails due to missing date configuration for the filename and unclear terminology ('Integration Mapping Integration Service').

Option D is the only one that fully addresses both the worker subtype mapping (with 'U' default) and implies the filename configuration, even if the date setup isn't explicitly listed (it's standard in Workday).

Supporting Documentation

The reasoning is based on Workday Pro Integrations best practices, including:

Workday Tutorial: Activity Creating Unique Filenames from EIB-Out Integrations -- Details on using sequence generators for filenames.

Workday Tutorial: EIB Features -- Explains integration mappings and default values.

Get_Sequence_Generators Operation Details -- Workday API documentation on sequence generators.

Workday Advanced Studio Tutorial -- Covers Core Connector templates and file name configurations.

r/workday Reddit Post: How to Create a New Sequence Generator for Filename for EIB -- Community insights on sequence generators.

Question 2

Question Type: MultipleChoice

Refer to the scenario. You are configuring a Core Connector: Worker integration with the Data Initialization Service (DIS) enabled. The integration must extract worker contact details and job information, including a calculated field override that determines phone allowance eligibility.

When testing, you run the Test Security Related Action from the Configure Integration Field Override step. Several field overrides display ''No'' in the Available by User column.

To ensure the ISSG has access to these field overrides and that ''Yes'' is displayed in the Test Security step, what configuration should you review?

Options:

A- Provide the ISSG View permissions to the domain security policies securing each overridden field.
B- Assign the ISSG to the domain security policies that govern the web service operations with Get access.
C- Grant View permissions to the ISSG for the domain security policies that secure the web

service operations.
D- Identify the domain security policies securing the field overrides and grant Modify permissions.

Answer:

A

Explanation:

The Test Security Related Action shows Available by User = No when the security group running the integration lacks View permissions to the fields used in the override logic.

From Workday documentation:

Field Overrides require the ISSG to have View access to the domain policies securing each field referenced in the override, otherwise Workday blocks the field from execution.

Therefore, the appropriate fix is to:

* Identify the domains that secure the calculated fields and overridden fields

* Grant the ISSG View access in those domain security policies

* Activate pending changes

Options B and C incorrectly focus only on web service operations.

Option D incorrectly suggests Modify access --- but View is the required minimum.

Question 3

Question Type: MultipleChoice

You are configuring an EIB that uses a custom report as its data source. When attempting to transfer ownership of the report to the Integration System User (ISU), the ISU does not appear as an option for new report owners. You confirm that the ISU already has the necessary access to the report data source and related fields.

Within the Custom Report Creation domain, which security configuration should you update to allow the ISU to appear as a valid report owner?

Options:

A- Assign the ISSG to a row within the Report/Task Permissions table that has Modify access enabled.
B- Assign the ISSG to a row within the Integration Permissions table that has Get access enabled.
C- Assign the ISSG to a row within the Report/Task Permissions table that has View access enabled.
D- Assign the ISSG to a row within the Integration Permissions table that has Put access enabled.

Answer:

A

Explanation:

In Workday, for an Integration System User (ISU) to be selectable as a Custom Report Owner, the security group the ISU belongs to must have Modify access to custom reports.

From Workday's security configuration principle:

An ISU does not appear as a valid report owner unless its security group has Modify permission in the Report/Task Permissions section of the Custom Report Creation domain security policy.

This is because report ownership requires write-level access over custom report objects.

Therefore, you must update the Report/Task Permissions table to include the ISSG with Modify access.

Options B, C, and D are incorrect because View or Get/Put do not provide report ownership capabilities.

Question 4

Question Type: MultipleChoice

Your manager has asked for a value on their dashboard for how many days away the birthdays are of their direct reports. The format of the output should be [Worker's Name]'s birthday is in [X] days, where you must calculate the number of days until a Worker's next birthday. An example output is "Logan McNeil's birthday is in 103 days."

Which calculated field functions do you need to accomplish this?

Options:

A- Format Date, Increment or Decrement Date, Extract Single Instance, Format Text
B- Build Date, Format Date, Extract Single Instance, Format Text
C- Date Difference, Format Number, Text Constant, Concatenate Text
D- Increment or Decrement Date, Format Number, Text Constant, Concatenate Text: Incorrect. Increment or Decrement Date can't directly calculate days to a future birthday without additional complexity; Date Difference is more appropriate.
Implementation:
Use Date Difference to calculate days from today to the next birthday (adjusting the year dynamically with additional logic if needed).
Apply Format Number to ensure the result is a clean integer.
Use Text Constant for static text (''s birthday is in ' and ' days').
Use Concatenate Text to combine Worker Name, static text, and the formatted number.
Reference from Workday Pro Integrations Study Guide:
Workday Calculated Fields: Section on 'Date Functions' explains Date Difference for calculating time spans.
Report Writer Fundamentals: Covers Concatenate Text and Text Constant for string building in reports.
D- Increment or Decrement Date, Format Number, Text Constant, Concatenate Text

Answer:

C

Explanation:

The requirement is to create a calculated field for a dashboard that displays a worker's name and the number of days until their next birthday in the format '[Worker's Name]'s birthday is in [X] days' (e.g., 'Logan McNeil's birthday is in 103 days'). This involves calculating the difference between today's date and the worker's next birthday, then formatting the output as a text string. Let's break down the necessary functions:

Date Difference:To calculate the number of days until the worker's next birthday, you need to determine the difference between the current date and the worker's birthdate in the current or next year (whichever is upcoming). The Date Difference function calculates the number of days between two dates. In this case:

Use the worker's 'Date of Birth' field (from the Worker business object).

Adjust the year of the birthdate to the current year or next year (if the birthday has already passed this year) using additional logic.

Calculate the difference from today's date to this adjusted birthday date. For example, if today is February 21, 2025, and Logan's birthday is June 4 (adjusted to June 4, 2025), Date Difference returns 103 days.

Format Number:The result of Date Difference is a numeric value (e.g., 103). To ensure it displays cleanly in the output string (without decimals or unnecessary formatting), Format Number can be used to convert it to a simple integer string (e.g., '103').

Text Constant:To build the output string, static text like ''s birthday is in ' and ' days' is needed.

The Text Constant function provides fixed text values to include in the final concatenated result.

Concatenate Text:The final step is to combine the worker's name (e.g., 'Logan McNeil'), the static text, and the calculated days into one string. Concatenate Text merges multiple text values into a single output, such as 'Logan McNeil' + ''s birthday is in ' + '103' + ' days'.

Option Analysis:

A . Format Date, Increment or Decrement Date, Extract Single Instance, Format Text: Incorrect. Format Date converts dates to strings but doesn't calculate differences. Increment or Decrement Date adjusts dates but isn't suited for finding days until a future event. Extract Single Instance is for multi-instance fields, not relevant here. Format Text adjusts text appearance, not numeric calculations.

B . Build Date, Format Date, Extract Single Instance, Format Text: Incorrect. Build Date creates a date from components, useful for setting the next birthday, but lacks the difference calculation. Format Date and Extract Single Instance don't apply to the core need.

C . Date Difference, Format Number, Text Constant, Concatenate Text: Correct. These functions cover calculating the days, formatting the number, adding static text, and building the final string.

Question 5

Question Type: MultipleChoice

Refer to the following scenario to answer the question below. Your integration has the following runs in the integration events report (Date format of MM/DD/YYYY):

Run #1

* Core Connector: Worker Integration System was launched on May 15, 2024 at 3:00:00 AM.

* As of Entry Moment: 05/15/2024 3:00:00 AM

* Effective Date: 05/15/2024

* Last Successful As of Entry Moment: 05/01/2024 3:00:00 AM

* Last Successful Effective Date: 05/01/2024

Run #2

* Core Connector: Worker Integration System was launched on May 31, 2024 at 3:00:00 AM.

* As of Entry Moment: 05/31/2024 3:00:00 AM

* Effective Date: 05/31/2024

* Last Successful As of Entry Moment: 05/15/2024 3:00:00 AM

* Last Successful Effective Date: 05/15/2024 On May 13, 2024 Brian Hill receives a salary increase. The new salary amount is set to $90,000.00 with an effective date of April 30,2024. Which of these runs will include Brian Hill's compensation change?

Options:

A- Brian Hill will be included in both integration runs.
B- Brian Hill will only be included in the second integration run.
C- Brian Hill will only be included in the first integration run.
D- Brian Hill will be excluded from both integration runs.

Answer:

D

Explanation:

The scenario involves a Core Connector: Worker integration with two runs detailed in the integration events report. The goal is to determine whether Brian Hill's compensation change, effective April 30, 2024, and entered on May 13, 2024, will be included in either of the runs based on their date launch parameters. Let's analyze each run against the change details to identify the correct answer.

In Workday, the Core Connector: Worker integration in incremental mode (as indicated by the presence of 'Last Successful' parameters) processes changes based on the Transaction Log, filtering them by the Entry Moment (when the change was entered) and Effective Date (when the change takes effect). The integration captures changes where:

The Entry Moment falls between the Last Successful As of Entry Moment and the As of Entry Moment, and

The Effective Date falls between the Last Successful Effective Date and the Effective Date.

Brian Hill's compensation change has:

Entry Moment: 05/13/2024 (time not specified, so we assume it occurs at some point during the day, before or up to 11:59:59 PM).

Effective Date: 04/30/2024.

Analysis of Run #1

Launch Date: 05/15/2024 at 3:00:00 AM

As of Entry Moment: 05/15/2024 3:00:00 AM -- The latest point for when changes were entered.

Effective Date: 05/15/2024 -- The latest effective date for changes.

Last Successful As of Entry Moment: 05/01/2024 3:00:00 AM -- The starting point for entry moments.

Last Successful Effective Date: 05/01/2024 -- The starting point for effective dates.

For Run #1 to include Brian's change:

The Entry Moment (05/13/2024) must be between 05/01/2024 3:00:00 AM and 05/15/2024 3:00:00 AM. Since 05/13/2024 falls within this range (assuming the change was entered before 3:00:00 AM on 05/15/2024, which is reasonable unless specified otherwise), this condition is met.

The Effective Date (04/30/2024) must be between 05/01/2024 (Last Successful Effective Date) and 05/15/2024 (Effective Date). However, 04/30/2024 is before 05/01/2024, so this condition is not met.

Since the effective date of Brian's change (04/30/2024) precedes the Last Successful Effective Date (05/01/2024), Run #1 will not include this change. In incremental mode, Workday excludes changes with effective dates prior to the last successful effective date, as those are assumed to have been processed in a prior run (before Run #1's baseline of 05/01/2024).

Analysis of Run #2

Launch Date: 05/31/2024 at 3:00:00 AM

As of Entry Moment: 05/31/2024 3:00:00 AM -- The latest point for when changes were entered.

Effective Date: 05/31/2024 -- The latest effective date for changes.

Last Successful As of Entry Moment: 05/15/2024 3:00:00 AM -- The starting point for entry moments.

Last Successful Effective Date: 05/15/2024 -- The starting point for effective dates.

For Run #2 to include Brian's change:

The Entry Moment (05/13/2024) must be between 05/15/2024 3:00:00 AM and 05/31/2024 3:00:00 AM. However, 05/13/2024 is before 05/15/2024 3:00:00 AM, so this condition is not met.

The Effective Date (04/30/2024) must be between 05/15/2024 (Last Successful Effective Date) and 05/31/2024 (Effective Date). Since 04/30/2024 is before 05/15/2024, this condition is also not met.

In Run #2, the Entry Moment (05/13/2024) precedes the Last Successful As of Entry Moment (05/15/2024 3:00:00 AM), meaning the change was entered before the starting point of this run's detection window. Additionally, the Effective Date (04/30/2024) is well before the Last Successful Effective Date (05/15/2024). Both filters exclude Brian's change from Run #2.

Conclusion

Run #1: Excluded because the effective date (04/30/2024) is before the Last Successful Effective Date (05/01/2024).

Run #2: Excluded because the entry moment (05/13/2024) is before the Last Successful As of

Entry Moment (05/15/2024 3:00:00 AM) and the effective date (04/30/2024) is before the Last Successful Effective Date (05/15/2024).

Brian Hill's change would have been processed in an earlier run (prior to May 1, 2024) if the integration was running incrementally before Run #1, as its effective date (04/30/2024) predates both runs' baselines. Given the parameters provided, neither Run #1 nor Run #2 captures this change, making D. Brian Hill will be excluded from both integration runs the correct answer.

Workday Pro Integrations Study Guide Reference

Workday Integrations Study Guide: Core Connector: Worker -- Section on 'Incremental Processing' explains how changes are filtered based on entry moments and effective dates relative to the last successful run.

Workday Integrations Study Guide: Launch Parameters -- Details how 'Last Successful As of Entry Moment' and 'Last Successful Effective Date' define the starting point for detecting new changes, excluding prior transactions.

Workday Integrations Study Guide: Change Detection -- Notes that changes with effective dates before the last successful effective date are assumed processed in earlier runs and are skipped in incremental mode.

Question 6

Question Type: MultipleChoice

When creating an XSLT file to transform the XML output of an EIB, you must have the XSL namespace. What other namespace(s) do you need to process any part of the source XML file?

Options:

A- The most commonly used namespace of the source XML document.
B- All namespaces that are a part of the source XML document.
C- Either the ETV or XTT namespace based on the type of output file desired.
D- No namespaces from the source XML document are needed.

Answer:

B

Explanation:

When writing XSLT to transform an XML document, you must declare and reference all XML namespaces used in the source XML.

''To accurately access and transform nodes using XPath, every namespace in the source document must be declared in the XSLT stylesheet.''

This ensures that XPath expressions correctly match the fully qualified elements, especially when multiple namespaces are in use.

Why the others are incorrect:

A (most commonly used) would be incomplete.

C (ETV/XTT) are specific Workday terminologies but don't replace namespace declarations.

D is incorrect; namespaces are required to avoid XPath resolution failures.

Question 7

Question Type: MultipleChoice

What XSL component is required to execute valid transformation instructions in the XSLT code?

Options:

A- xsl:template
B- xsl:apply-template
C- xsl:call-template
D- xsl:output

Answer:

A

Explanation:

The <xsl:template> is the core component in XSLT. It defines the transformation rules that will be applied to nodes in the XML document.

''Without at least one <xsl:template> element, an XSLT file cannot perform any transformation. This is the execution block where processing logic begins.''

Why the others are incorrect:

B . <xsl:apply-templates> applies templates but is not valid without the actual template definitions.

C . <xsl:call-template> calls named templates --- which must first exist.

D . <xsl:output> defines format but does not perform transformation logic.

Question 8

Question Type: MultipleChoice

How does an XSLT processor identify the specific nodes in an XML document to which a particular transformation rule should be applied?

Options:

A- The processor matches nodes using XPath expressions within templates.
B- The stylesheet element directs the processor to specific XML sections.
C- Named templates explicitly call processing for designated elements.
D- The processor targets nodes based on declared namespace prefixes.

Answer:

A

Explanation:

In XSLT, the processor applies transformation rules by matching nodes using XPath expressions inside <xsl:template match=''> statements.

''Templates define the rule, and XPath expressions determine which nodes they apply to.''

This is the foundational mechanism by which XSLT processes XML data.

Why the others are incorrect:

B . The <xsl:stylesheet> element defines scope, not node matching.

C . <xsl:call-template> invokes a named template but does not itself match nodes.

D . Namespace prefixes are used within XPath, but node matching is based on XPath.

Question 9

Question Type: MultipleChoice

How do you initially upload the XSLT file to a Document Transformation integration system?

Options:

A- From the Related Action on the Document Transformation, select Configure Integration Attachment Service.
B- From the Related Action on the Document Transformation, select Configure Integration Attributes.
C- In the Global Workday Search bar, run the Edit Integration Attachment Service task.
D- In the Global Workday Search bar, run the Edit Integration Service Attachment task.

Answer:

A

Explanation:

To upload an XSLT file to a Document Transformation integration system, you use the Configure Integration Attachment Service.

As per Workday documentation:

''The Configure Integration Attachment Service option on the Related Actions menu allows you to attach and manage XSLT files or other transformation documents used in Document Transformation integrations.''

This is the initial and correct method to upload the XSLT used for transforming incoming or outgoing XML.

Why the others are incorrect:

B . Configure Integration Attributes configures integration behavior, not attachments.

C and D reference invalid or misnamed tasks; they are not valid Workday tasks for XSLT upload.