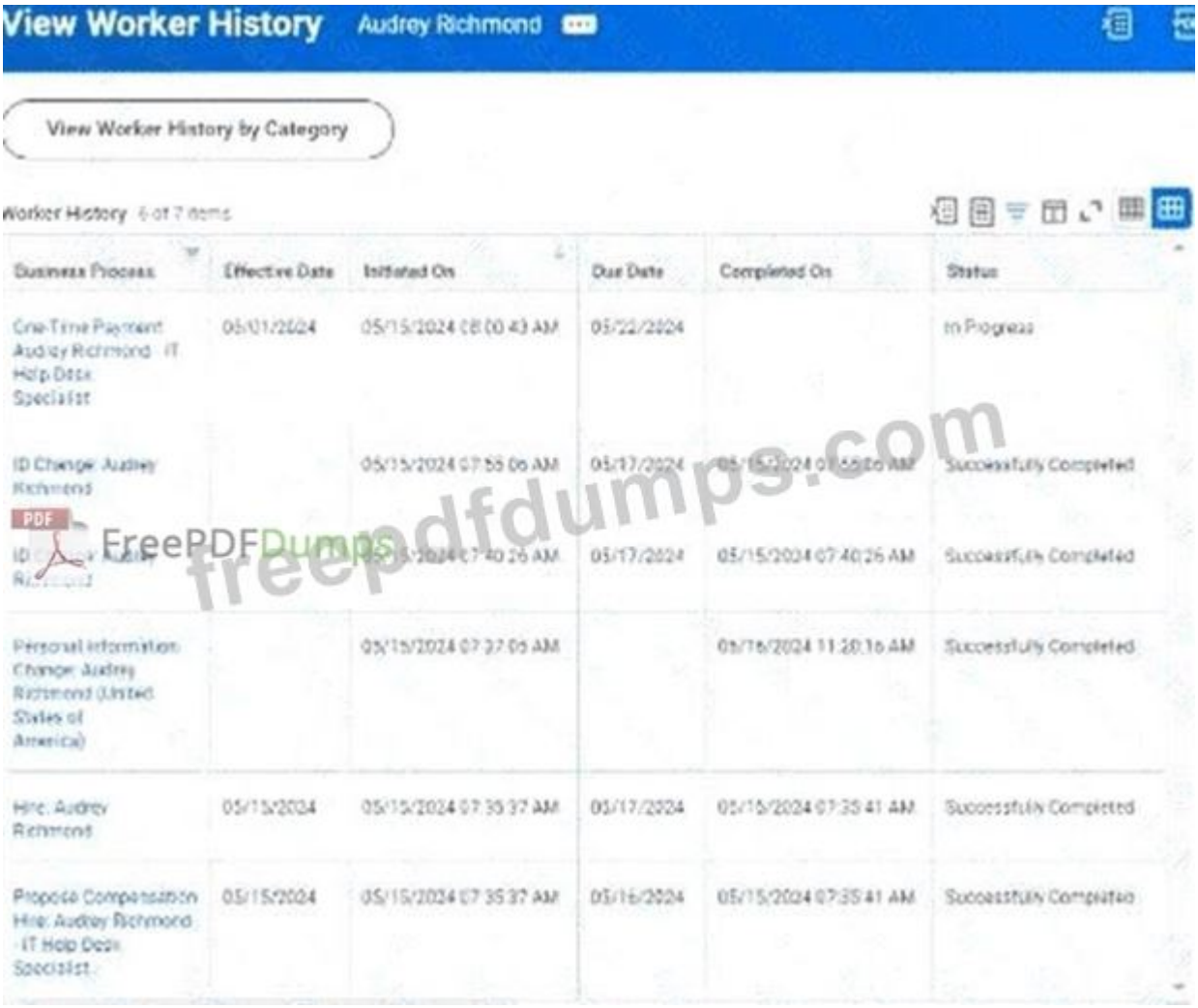
Workday.Workday-Pro-Integrations.v2026-09-16.q48

Exam Code:	Workday-Pro-Integrations
Exam Name:	Workday Pro Integrations Certification Exam
Certification Provider:	Workday
Free Question Number:	48
Version:	v2026-09-16
# of views:	121
# of Questions views:	480
https://www.freepdfdumps.com/Workday.Workday-Pro-Integrations.v2026-09-16.q48.html	

NEW QUESTION: 1
Refer to the following scenario to answer the question below.
A connector is configured to detect changes to government IDs, personal information, and compensation data.
The worker history report below shows recent transactions for a new hire.



Worker History: Audrey Richmond
The worker history includes a One Time Payment transaction for Audrey Richmond with an effective date of
05/01/2024, initiated on 05/15/2024, due date of 05/17/2024, and status of In Progress. Other transactions shown include ID Change, Personal Information Change, Hire, and Propose Compensation records.
You launch the connector integration to process changes with the following parameters:
As Of Entry Moment: 05/17/2024 12:00:00 AM
Effective Date: 05/17/2024
Last Successful As Of Entry Moment: 05/15/2024 12:00:00 AM
Last Successful Effective Date: 05/15/2024
Why will the integration output file exclude the one-time payment change?
A. Due date is outside the launch timeline.
B. An HR administrator rescinded the change.
C. Effective date is outside the launch timeline.
D. Status of the change is in progress.
Answer: (SHOW ANSWER)
Comprehensive and Detailed 100 to 150 words of Explanation From Workday Pro Integration topics:
The one-time payment change is excluded because its business process status is still In Progress. Core connector change detection processes completed and eligible transactions based on the configured transaction subscriptions and launch parameters. Even though the transaction appears in worker history and its dates may fall within the relevant launch window, Workday should not send incomplete business process results to an external system. An in-progress compensation event has not reached a final approved or successfully completed state, so the connector does not treat it as ready for outbound delivery. The due date is not the controlling factor, and there is no evidence that the change was rescinded. The effective date is also not the main issue here. The exclusion is caused by the transaction's incomplete status.

NEW QUESTION: 2
Refer to the following scenario to answer the question below.
You need to configure a Core Connector: Candidate Outbound integration for your vendor. The connector requires the data initialization service (DIS).
The vendor requests additional formatting of the candidate Country field. For example, if a candidate ' s country is the United States of America, the output should show USA.
What steps do you follow to meet this request?
A. Use an Evaluated Expression calculation and add it to the integration ' s report data source.
B. Use the integration related action Configure Integration Population Eligibility.
C. Use the integration services to only output shortened country codes.

D. Use the integration related action Configure Integration Maps.

Answer: D (LEAVE A REPLY)

The scenario involves a Core Connector: Candidate Outbound integration with the Data Initialization Service (DIS), where the vendor requires the " Country " field to be formatted differently (e.g., " United States of America " to " USA "). This is a data transformation requirement, and Core Connectors provide specific tools to handle such formatting. Let's evaluate the solution:

* Requirement:The vendor needs a shortened country code (e.g., " USA " instead of " United States of America ") in the output file. This involves transforming the delivered " Country " field value from the Candidate business object into a vendor-specific format.
* Integration Maps:In Workday Core Connectors, integration maps are used to transform or map field values from Workday's format to a vendor's required format. For example, you can create a map that replaces " United States of America " with " USA, " " Canada " with " CAN, " etc. This is configured via the " Configure Integration Maps " related action on the integration system, allowing you to define a lookup table or rule-based transformation for the Country field.
* Option Analysis:
* A. Use an Evaluated Expression calculation and add it to the integration's report data source:

Incorrect. While an Evaluate Expression calculated field could transform the value (e.g., if-then logic), Core Connectors don't directly use report data sources for output formatting. Calculated fields are better suited for custom reports or EIBs, not Core Connector field mapping.

* B. Use the integration related action Configure Integration Population Eligibility: Incorrect. This action filters the population of candidates included (e.g., based on eligibility criteria), not the formatting of individual fields like Country.
* C. Use the integration services to only output shortened country codes: Incorrect. Integration services define the dataset or events triggering the integration, not field-level formatting or transformations.
* D. Use the integration related action Configure Integration Maps: Correct. Integration maps are the standard Core Connector tool for transforming field values (e.g., mapping " United States of America " to " USA ") to meet vendor requirements.
* Implementation:
* Navigate to the Core Connector: Candidate Outbound integration system.
* Use the related action Configure Integration Maps.
* Create a new map for the " Country " field (e.g., Source Value: " United States of America, " Target Value: " USA ").
* Apply the map to the Country field in the integration output.
* Test the output file to ensure the transformed value (e.g., " USA ") appears correctly.

References from Workday Pro Integrations Study Guide:

* Core Connectors & Document Transformation: Section on " Configuring Integration Maps " details how to transform field values for vendor-specific formatting.
* Integration System Fundamentals: Explains how Core Connectors handle data transformation through maps rather than calculated fields or services for field-level changes.

NEW QUESTION: 3
How many integration systems can an ISU be assigned to concurrently?
A. One
B. Three
C. Five
D. Unlimited
Answer: (SHOW ANSWER)
The Integration System User (ISU) in Workday is a specialized user account designed for automation and system-level integrations. It can be assigned to any number of integration systems concurrently - there is no limit.
From Workday documentation and Pro training:
"A single ISU can be assigned to multiple integration systems across tenants and environments, provided it has the correct permissions and security group assignments. Workday does not impose a hard limit on the number of systems an ISU can be linked to."
This design provides scalability for environments with multiple integrations (e.g., EIBs, Core Connectors, Studio integrations) without needing to create redundant users.
Incorrect Options Explained:
* A, B, C: These options imply arbitrary limits (one, three, five), which do not exist in Workday ' s ISU architecture.
References:
Workday Pro: Integrations - Integration System Security User Management Workday Community: How ISUs Function in Multi-Integration Environments

NEW QUESTION: 4
You are configuring a Core Connector integration that will send data to an external vendor. The vendor requires that the integration output file includes a unique Batch ID in its filename.
This ID must follow the format VNDR_YEARMONTH_seq.xml, where YEAR is the four-digit year, MONTH is the two-digit month, and seq is a five-digit sequence number that must reset to 00001 at the beginning of each new month.
Which settings must you configure on the integration Filename Sequence Generator to produce this Batch ID filename?
A. Sequence ID Format = VNDR_[yyyy][MM] _[seq].xml
Padding = 5
Restart Frequency = Month

B. Sequence ID Format = VNDR_yyyyMM_seq.xml
Padding = 00005
Restart Frequency = Year
C. Sequence ID Format = VNDR_[year][month] _[seq].xml
Padding = 00000
Restart Frequency = Day
D. Sequence ID Format = VNDR_[yearmonth]_[sequence] .xml
Padding = 0
Restart Frequency = Month
Answer: A (LEAVE A REPLY)
The required filename needs a static prefix, runtime date tokens, and a monthly resetting sequence. The correct Workday-style pattern uses bracketed date tokens: [yyyy] for the four-digit year and [MM] for the two- digit month. The sequence token [seq] supplies the next sequence number. Because the sequence must be five digits, Padding must be set to 5 so values appear as 00001, 00002, and so on. Restart Frequency must be Month because the sequence resets at the beginning of each new month. Option B incorrectly treats the tokens as literal text and resets yearly. Option C uses invalid token names and resets daily. Option D uses nonstandard token names and lacks the required five-digit padding.

NEW QUESTION: 5
What is the workflow to chain a Document Transformation system to a Connector integration for the purpose of transforming the output?
A. Add a Service step of Fire Integration to the Document Transformation (DT) Business Process (BP)
B. Add an Integration step to the Connector Business Process (BP)
C. Add an Integration step to the Document Transformation (DT) Business Process (BP)
D. Add a Service step of Fire Integration to the Connector Business Process (BP)
Answer: (SHOW ANSWER)
To chain a Document Transformation system to a Connector Integration, you must configure the Connector Integration System's Business Process (BP) to include a "Service step of Fire Integration", which triggers the Document Transformation after the connector completes.
From Workday documentation:
"To execute a Document Transformation after a connector integration, use the Fire Integration service step in the connector's business process to trigger the Document Transformation integration." This allows Workday to chain multiple integrations, such as taking the output of a Core Connector and sending it through a transformation step (e.g., XSLT) before delivering to an endpoint.
Why other options are incorrect:
* A. Fire Integration in the DT BP is not used to call itself.

* B. "Integration step" in BP is not a valid step type.
* C. Same issue - DT's own BP doesn't call itself or other integrations.
Reference:Workday Pro: Document Transformation - Integration Chaining via Fire Integration StepWorkday Integration Certification Guide - Document Transformation and BP Chaining

NEW QUESTION: 6
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 calculated field data source.
B. Access not provided to all fields in the calculated field.
C. Access not provided to Connector calculated field web service.
D. Access not provided to all instances of calculated field.
Answer: (SHOW ANSWER)
This question addresses a troubleshooting scenario in Workday Pro Integrations, where a calculated field used as a field override in a Connector does not appear in the output, despite having a value. Let's analyze the potential causes and evaluate each option.
Understanding Calculated Fields and Connectors in Workday
* Calculated Fields: In Workday, calculated fields are custom fields created using Workday's expression language to derive values based on other fields, conditions, or functions. They are often used in reports, integrations, and business processes to transform or aggregate data. Calculated fields can reference other fields (data sources) and require appropriate security permissions to access those underlying fields.
* Field Override in Connectors: In a Core Connector or other integration system, a field override allows you to replace or supplement a default field with a custom value, such as a calculated field. This is configured in the integration's mapping or transformation steps, ensuring the output includes the desired data. However, for the calculated field to appear in the output, it must be accessible, have a valid value, and be properly configured in the integration.
* Issue: Calculated Field Not Appearing in Output: If the calculated field has a value but doesn't appear in the Connector's output, the issue likely relates to security, configuration, or access restrictions. The question assumes the field has a value, so we focus on permissions or setup errors rather than data issues.
Evaluating Each Option
Let's assess each option based on Workday's integration and security model:
Option A: Access not provided to calculated field data source.
* Analysis: This is partially related but incorrect as the primary cause. Calculated fields often rely on underlying data sources (e.g., worker data, organization data) to compute their values. If access to the data source is restricted, the calculated field might not compute correctly or appear in the output.

However, the question specifies the field has a value, implying the data source is accessible. The more specific issue is likely access to the individual fields within the calculated field's expression, not just the broader data source.
* Why It Doesn't Fit: While data source access is important, it's too general here. The calculated field's value exists, suggesting the data source is accessible, but the problem lies in finer-grained permissions for the fields used in the calculation.
Option B: Access not provided to all fields in the calculated field.
* Analysis: This is correct. Calculated fields in Workday are expressions that reference one or more fields (e.g., Worker_ID + Position_Title). For the calculated field to be used in a Connector's output, the ISU (via its ISSG) must have access to all fields referenced in the calculation. If any field lacks " Get " or " View " permission in the relevant domain (e.g., Worker Data), the calculated field won't appear in the output, even if it has a value. This is a common security issue in integrations, as ISSGs must be configured with domain access for every field involved.
* Why It Fits: Workday's security model requires granular permissions. For example, if a calculated field combines Worker_Name and Hire_Date, the ISU needs access to both fields' domains. If Hire_Date is restricted, the calculated field fails to output, even with a value. This aligns with the scenario and is a frequent troubleshooting point in Workday Pro Integrations.
Option C: Access not provided to Connector calculated field web service.
* Analysis: This is incorrect. There isn't a specific " Connector calculated field web service " in Workday. Calculated fields are part of the integration's configuration, not a separate web service. The web service operation used by the Connector (e.g., Get_Workers) must have permissions, but this relates to the overall integration, not the calculated field specifically. The issue here is field-level access, not a web service restriction.
* Why It Doesn't Fit: This option misinterprets Workday's architecture. Calculated fields are configured within the integration, not as standalone web services, making this irrelevant to the problem.
Option D: Access not provided to all instances of calculated field.
* Analysis: This is incorrect. The concept of " instances " typically applies to data records (e.g., all worker records), not calculated fields themselves. Calculated fields are expressions, not data instances, so there's no need for " instance-level " access. The issue is about field-level permissions within the calculated field's expression, not instances of the field. This option misunderstands Workday's security model for calculated fields.
* Why It Doesn't Fit: Calculated fields don't have " instances " requiring separate access; they depend on the fields they reference, making this option inaccurate.
Final Verification
The correct answer is Option B, as the calculated field's absence in the output is likely due to the ISU lacking access to all fields referenced in the calculated field's expression. For example, if the calculated field in a Core Connector: Worker Data combines Worker_ID and Department_Name, the ISSG must have " Get " access to both the Worker Data and

Organization Data domains. If Department_Name is restricted, the calculated field won't output, even with a value. This is a common security configuration issue in Workday integrations, addressed by reviewing and adjusting ISSG domain permissions.
This aligns with Workday's security model, where granular permissions are required for all data elements, as seen in Questions 26 and 28. The assumption that the field has a value rules out data or configuration errors, focusing on security as the cause.
Supporting Documentation
The reasoning is based on:
* Workday Community documentation on calculated fields, security domains, and integration mappings.
* Tutorials on configuring Connectors and troubleshooting, such as Workday Advanced Studio Tutorial, highlighting field access issues.
* Integration security guides from partners (e.g., NetIQ, Microsoft Learn, Reco.ai) detailing ISSG permissions for fields in calculated expressions.
* Community discussions on Reddit and Workday forums on calculated field troubleshooting (r/workday on Reddit).

NEW QUESTION: 7
You are creating a connector based integration where all fields are provided by the template. However, the vendor would also like the following configurations as well:
* A file name output to have the current date and integration run number
* Have internal values for a particular field transferred to their external values What workflow would you follow to create this integration?
A. * Enable Needed Integration Services * Configure Integration Field Attributes * Configure Integration Maps * Configure Sequence Generator
B. * Enable Needed Integration Attributes * Configure Integration Maps * Configure Integration Services * Configure Sequence Generator
C. * Enable Needed Integration Maps * Configure Integration Services * Configure Integration Field Attributes * Configure Sequence Generator
D. * Enable Needed Integration Services * Configure Integration Attributes * Configure Integration Maps * Configure Sequence Generator
Answer: A (LEAVE A REPLY)
To create a connector-based integration with additional custom configurations such as dynamic file naming and internal-to-external value mapping, the following steps must be followed:
* Enable Needed Integration Services:
* This step involves activating the required integration services to ensure that the necessary API calls, security, and processing capabilities are available within Workday.
* Configure Integration Field Attributes:

* Integration Field Attributes allow customization of fields within the integration, enabling changes to formats, mappings, and transformations, such as including a dynamically generated file name with the current date and integration run number.
* Configure Integration Maps:
* Integration Maps are used to transform internal values into external values as per the vendor's requirements. This ensures that data fields in Workday align correctly with external system specifications.
* Configure Sequence Generator:
* The Sequence Generator is used to append unique identifiers to output files, ensuring each integration run produces a uniquely named file (e.g., including the current date and run number).
This workflow ensures that the integration is set up efficiently while meeting the vendor's additional configuration needs.
References: Workday Advanced Business Process documentation

NEW QUESTION: 8
Refer to the following scenario to answer the question below.
An external system needs a file containing data for recent worker job changes. They would like to receive a file routinely at 5 PM eastern standard time, every 48 hours. The file should show job changes since the last integration run.
What is the run frequency of the integration schedule?
A. Hourly Recurrence
B. Custom Recurrence
C. Daily Recurrence
D. Minute Recurrence
Answer: (SHOW ANSWER)
The schedule requirement is every 48 hours, which is handled as a daily recurrence pattern with an interval of two days. Workday scheduling separates the general run frequency from the more specific recurrence interval.
The run frequency is therefore Daily Recurrence, while the recurrence detail would define the integration to recur every two days. Hourly Recurrence would be used when the integration runs at hourly intervals, not every two calendar days. Minute Recurrence is far too granular for this requirement. Custom Recurrence is unnecessary because the stated pattern can be handled by the standard daily recurrence configuration. Since the file must include changes since the last run, this schedule supports incremental processing while keeping the cadence predictable.

NEW QUESTION: 9
What is the relationship between the Integration System User (ISU), Integration System Security Group (ISSG), and domain security policies?
A. Assign domain security policies to the ISSG, and then assign the ISSG to the ISU.

B. Assign domain security policies to the ISU, and then assign the ISU to the ISSG.
C. Assign the ISU to the ISSG, and then assign the ISSG to domain security policies.
D. Assign the ISSG to the ISU, and then assign the ISU to domain security policies.
Answer: C (LEAVE A REPLY)
This question is about the correct order of Workday security assignment for integrations.
Workday clearly specifies the security structure:
"You assign the ISU to the Integration System Security Group (ISSG).
Then you assign the ISSG to the domain security policies."
This is because domain security policies apply to security groups, not directly to ISUs.
Correct Relationship Order:
* Create ISU
* Create/assign ISU to ISSG
* Assign ISSG to the domain security policies (Get/Put/View)
That aligns exactly to option C.
References:Admin#Guide#Authentication#and#Security.pdf - Concept: Integration Security in Workday (Security Groups # Domain Policies hierarchy)

NEW QUESTION: 10
What is the purpose of granting an ISU modify access to the Integration Event domain via an ISSG?
A. To have the ISU own the integration schedule.
B. To let the ISU configure integration attributes and maps.
C. To log into the user interface as the ISU and launch the integration.
D. To build the integration system as the ISU.
Answer: (SHOW ANSWER)
Understanding ISUs and Integration Systems in Workday
* Integration System User (ISU): An ISU is a specialized user account in Workday designed for integrations, functioning as a service account to authenticate and execute integration processes. ISUs are created using the " Create Integration System User " task and are typically configured with settings like disabling UI sessions and setting long session timeouts (e.g., 0 minutes) to prevent expiration during automated processes. ISUs are not human users but are instead programmatic accounts used for API calls, EIBs, Core Connectors, or other integration mechanisms.
* Integration Systems: In Workday, an " integration system " refers to the configuration or setup of an integration, such as an External Integration Business (EIB), Core Connector, or custom integration via web services. Integration systems are defined to handle data exchange between Workday and external systems, and they require authentication, often via an ISU, to execute tasks like data retrieval, transformation, or posting.
* Assigning ISUs to Integration Systems: ISUs are used to authenticate and authorize integration systems to interact with Workday. When configuring an integration system, you assign an ISU to provide the credentials needed for the integration to run. This assignment

ensures that the integration can access Workday data and functionalities based on the security permissions granted to the ISU via its associated Integration System Security Group (ISSG).
* Limitation on Assignment: Workday's security model imposes restrictions to maintain control and auditability. Specifically, an ISU is designed to be tied to a single integration system to ensure clear accountability, prevent conflicts, and simplify security management. This limitation prevents an ISU from being reused across multiple unrelated integration systems, reducing the risk of unintended access or data leakage.
Evaluating Each Option
Let's assess each option based on Workday's integration and security practices:
Option A: An ISU can be assigned to five integration systems.
* Analysis: This is incorrect. Workday does not impose a specific numerical limit like " five " for ISU assignments to integration systems. Instead, the limitation is more restrictive: an ISU is typically assigned to only one integration system to ensure focused security and accountability. Allowing an ISU to serve multiple systems could lead to confusion, overlapping permissions, or security risks, which Workday's design avoids.
* Why It Doesn't Fit: There's no documentation or standard practice in Workday Pro Integrations suggesting a limit of five integration systems per ISU. This option is arbitrary and inconsistent with Workday's security model.
Option B: An ISU can be assigned to an unlimited number of integration systems.
* Analysis: This is incorrect. Workday's security best practices do not allow an ISU to be assigned to an unlimited number of integration systems. Allowing this would create security vulnerabilities, as an ISU' s permissions (via its ISSG) could be applied across multiple unrelated systems, potentially leading to unauthorized access or data conflicts. Workday enforces a one-to-one or tightly controlled relationship to maintain auditability and security.
* Why It Doesn't Fit: The principle of least privilege and clear accountability in Workday integrations requires limiting an ISU's scope, not allowing unlimited assignments.
Option C: An ISU can be assigned to only one integration system.
* Analysis: This is correct. In Workday, an ISU is typically assigned to a single integration system to ensure that its credentials and permissions are tightly scoped. This aligns with Workday's security model, where ISUs are created for specific integration purposes (e.g., an EIB, Core Connector, or web service integration). When configuring an integration system, you specify the ISU in the integration setup (e.g., under " Integration System Attributes " or " Authentication " settings), and it is not reused across multiple systems to prevent conflicts or unintended access. This limitation ensures traceability and security, as the ISU's actions can be audited within the context of that single integration.
* Why It Fits: Workday documentation and best practices, including training materials and community forums, emphasize that ISUs are dedicated to specific integrations. For example, when creating an EIB or Core Connector, you assign an ISU, and it is not shared across other integrations unless explicitly reconfigured, which is rare and discouraged for security reasons.

Option D: An ISU can only be assigned to an ISSG and not an integration system.
* Analysis: This is incorrect. While ISUs are indeed assigned to ISSGs to inherit security permissions (as established in Question 26), they are also assigned to integration systems to provide authentication and authorization for executing integration tasks. The ISU's role includes both: it belongs to an ISSG for permissions and is linked to an integration system for execution. Saying it can only be assigned to an ISSG and not an integration system misrepresents Workday's design, as ISUs are explicitly configured in integration systems (e.g., EIB, Core Connector) to run processes.
* Why It Doesn't Fit: ISUs are integral to integration systems, providing credentials for API calls or data exchange. Excluding assignment to integration systems contradicts Workday's integration framework.
Final Verification
The correct answer is Option C, as Workday limits an ISU to a single integration system to ensure security, accountability, and clarity in integration operations. This aligns with the principle of least privilege, where ISUs are scoped narrowly to avoid overexposure. For example, when setting up a Core Connector: Job Postings (as in Question 25), you assign an ISU specifically for that integration, not multiple ones, unless reconfiguring for a different purpose, which is atypical.
Supporting Documentation
The reasoning is based on Workday Pro Integrations security practices, including:
* Workday Community documentation on creating and managing ISUs and integration systems.
* Tutorials on configuring EIBs, Core Connectors, and web services, which show assigning ISUs to specific integrations (e.g., Workday Advanced Studio Tutorial).
* Integration security overviews from implementation partners (e.g., NetIQ, Microsoft Learn, Reco.ai) emphasizing one ISU per integration for security.
* Community discussions on Reddit and Workday forums reinforcing that ISUs are tied to single integrations for auditability (r/workday on Reddit).
This question focuses on the purpose of granting an Integration System User (ISU) modify access to the Integration Event domain via an Integration System Security Group (ISSG) in Workday Pro Integrations. Let' s analyze the role of the ISU, the Integration Event domain, and evaluate each option to determine the correct answer.
Understanding ISUs, ISSGs, and the Integration Event Domain
* Integration System User (ISU): As described in previous questions, an ISU is a service account for integrations, used to authenticate and execute integration processes in Workday. ISUs are assigned to ISSGs to inherit security permissions and are linked to specific integration systems (e.g., EIBs, Core Connectors) for execution.
* Integration System Security Group (ISSG): An ISSG is a security group that defines the permissions for ISUs, controlling what data and functionalities they can access or modify. ISSGs can be unconstrained (access all instances) or constrained (access specific

instances based on context). Permissions are granted via domain security policies, such as " Get, " " Put, " " View, " or " Modify, " applied to Workday domains.
* Integration Event Domain: In Workday, the Integration Event domain (or Integration Events security domain) governs access to integration-related activities, such as managing integration events, schedules, attributes, mappings, and logs. This domain is critical for integrations, as it controls the ability to create, modify, or view integration configurations and runtime events.
* " Modify " access to the Integration Event domain allows the ISU to make changes to integration configurations, such as attributes (e.g., file names, endpoints), mappings (e.g., data transformations), and event settings (e.g., schedules or triggers).
* This domain does not typically grant UI access or ownership of schedules but focuses on configuration and runtime control.
* Purpose of Granting Modify Access: Granting an ISU modify access to the Integration Event domain via an ISSG enables the ISU to perform configuration tasks for integrations, ensuring the integration system can adapt or update its settings programmatically. This is essential for automated integrations that need to adjust mappings, attributes, or event triggers without manual intervention. However, ISUs are not designed for UI interaction or administrative ownership, as they are service accounts.
Evaluating Each Option
Let's assess each option based on Workday's security and integration model:
Option A: To have the ISU own the integration schedule.
* Analysis: This is incorrect. ISUs do not " own " integration schedules or any other integration components. Ownership is not a concept applicable to ISUs, which are service accounts for execution, not administrative entities. Integration schedules are configured within the integration system (e.g., EIB or Core Connector) and managed by administrators or users with appropriate security roles, not by ISUs. Modify access to the Integration Event domain allows changes to schedules, but it doesn't imply ownership.
* Why It Doesn't Fit: ISUs lack administrative control or ownership; they execute based on permissions, not manage schedules as owners. This misinterprets the ISU's role.
Option B: To let the ISU configure integration attributes and maps.
* Analysis: This is correct. Granting modify access to the Integration Event domain allows the ISU to alter integration configurations, including attributes (e.g., file names, endpoints, timeouts) and mappings (e.g., data transformations like worker subtype mappings from Question 25). The Integration Event domain governs these configuration elements, and " Modify " permission enables the ISU to update them programmatically during integration execution. This is a standard use case for ISUs in automated integrations, ensuring flexibility without manual intervention.
* Why It Fits: Workday's documentation and training materials indicate that the Integration Event domain controls integration configuration tasks. For example, in an EIB or Core Connector, an ISU with modify access can adjust mappings or attributes, as seen in

tutorials on integration setup (Workday Advanced Studio Tutorial). This aligns with the ISU's role as a service account for dynamic configuration.
Option C: To log into the user interface as the ISU and launch the integration.
* Analysis: This is incorrect. ISUs are not intended for UI interaction. When creating an ISU, a best practice is to disable UI sessions (e.g., set " Allow UI Sessions " to " No ") and configure a session timeout of 0 minutes to prevent expiration during automation. ISUs operate programmatically via APIs or integration systems, not through the Workday UI. Modify access to the Integration Event domain enables configuration changes, not UI login or manual launching.
* Why It Doesn't Fit: Logging into the UI contradicts ISU design, as they are service accounts, not user accounts. This option misrepresents their purpose.
Option D: To build the integration system as the ISU.
* Analysis: This is incorrect. ISUs do not " build " integration systems; they execute or configure existing integrations based on permissions. Building an integration system (e.g., creating EIBs, Core Connectors, or web services) is an administrative task performed by users with appropriate security roles (e.g., Integration Build domain access), not ISUs. Modify access to the Integration Event domain allows configuration changes, not the creation or design of integration systems.
* Why It Doesn't Fit: ISUs lack the authority or capability to build integrations; they are for runtime execution and configuration, not development or design.
Final Verification
The correct answer is Option B, as granting an ISU modify access to the Integration Event domain via an ISSG enables it to configure integration attributes (e.g., file names, endpoints) and maps (e.g., data transformations), which are critical for dynamic integration operations. This aligns with Workday's security model, where ISUs handle automated tasks within defined permissions, not UI interaction, ownership, or system building.
For example, in the Core Connector: Job Postings from Question 25, an ISU with modify access to Integration Event could update the filename pattern or worker subtype mappings, ensuring the integration adapts to vendor requirements without manual intervention. This is consistent with Workday's design for integration automation.
Supporting Documentation
The reasoning is based on Workday Pro Integrations security practices, including:
* Workday Community documentation on ISUs, ISSGs, and domain security (e.g., Integration Event domain permissions).
* Tutorials on configuring EIBs and Core Connectors, showing ISUs modifying attributes and mappings (Workday Advanced Studio Tutorial).
* Integration security overviews from implementation partners (e.g., NetIQ, Microsoft Learn, Reco.ai) detailing domain access for ISUs.
* Community discussions on Reddit and Workday forums reinforcing ISU roles for configuration, not UI or ownership (r/workday on Reddit).

NEW QUESTION: 11

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">61383c9b1d094d4da73166ad39caebce</wd:ID>
                        <wd:ID wd:type="Degree_ID">Mast</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 (LEAVE A REPLY)

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 References
* 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: 12
You need to create a report that includes data from multiple business objects. For a supervisory organization specified at run time, the report must output one row per worker, their active benefit plans, and the names and ages of all related dependents. The Worker business object contains the Employee, Benefit Plans, and Dependents fields. The Dependent business object contains the employee's dependent's Name and Age fields. How would you select the primary business object (PBO) and related business objects (RBO) for the report?

A. PBO: Dependent, RBO: Worker
B. PBO: Worker, RBO: Dependent
C. PBO: Dependent, no RBOs
D. PBO: Worker; no RBOs
Answer: B (LEAVE A REPLY)
In Workday reporting, selecting the appropriate Primary Business Object (PBO) and Related Business Objects (RBOs) is critical to ensure that the report retrieves and organizes data correctly based on the requirements.
The requirement here is to create a report that outputs one row per worker for a specified supervisory organization, including their active benefit plans and the names and ages of all related dependents. The Worker business object contains fields like Employee, Benefit Plans, and Dependents, while the Dependent business object provides the Name and Age fields for dependents.
* Why Worker as the PBO?The report needs to output "one row per worker," making the Worker business object the natural choice for the PBO. In Workday, the PBO defines the primary dataset and determines the granularity of the report (i.e., one row per instance of the PBO). Since the report revolves around workers and their associated data (benefit plans and dependents), Worker is the starting point. Additionally, the requirement specifies a supervisory organization at runtime, which is a filter applied to the Worker business object to limit the population.
* Why Dependent as an RBO?The Worker business object includes a "Dependents" field, which is a multi-instance field linking to the Dependent business object. To access detailed dependent data (Name and Age), the Dependent business object must be added as an RBO. This allows the report to pull in the related dependent information for each worker. Without the Dependent RBO, the report could only reference the existence of dependents, not their specific attributes like Name and Age.
* Analysis of Benefit Plans:The Worker business object already contains the "Benefit Plans" field, which provides access to active benefit plan data. Since this is a field directly available on the PBO (Worker), no additional RBO is needed to retrieve benefit plan information.
* Option Analysis:
* A. PBO: Dependent, RBO: Worker: Incorrect. If Dependent were the PBO, the report would output one row per dependent, not one row per worker, which contradicts the requirement.
Additionally, Worker as an RBO would unnecessarily complicate accessing worker-level data.
* B. PBO: Worker, RBO: Dependent: Correct. This aligns with the requirement: Worker as the PBO ensures one row per worker, and Dependent as the RBO provides access to dependent details (Name and Age). Benefit Plans are already accessible via the Worker PBO.

* C. PBO: Dependent, no RBOs: Incorrect. This would result in one row per dependent and would not allow easy access to worker or benefit plan data, failing to meet the "one row per worker" requirement.
* D. PBO: Worker, no RBOs: Incorrect. While Worker as the PBO is appropriate, omitting the Dependent RBO prevents the report from retrieving dependent Name and Age fields, which are stored in the Dependent business object, not directly on Worker.
* Implementation:
* Create a custom report with Worker as the PBO.
* Add a filter for the supervisory organization (specified at runtime) on the Worker PBO.
* Add Dependent as an RBO to access Name and Age fields.
* Include columns from Worker (e.g., Employee, Benefit Plans) and Dependent (e.g., Name, Age).
References from Workday Pro Integrations Study Guide:
* Workday Report Writer Fundamentals: Section on "Selecting Primary and Related Business Objects" explains how the PBO determines the report's row structure and RBOs extend data access to related objects.
* Integration System Fundamentals: Discusses how multi-instance fields (e.g., Dependents on Worker) require RBOs to retrieve detailed attributes.

NEW QUESTION: 13
What is the purpose of the < xsl:template > element?
A. Determine the output file type.
B. Grant access to the XSLT language.
C. Provide rules to apply to a specified node.
D. Generate an output file name.
Answer: C (LEAVE A REPLY)
The < xsl:template > element is a fundamental component of XSLT (Extensible Stylesheet Language Transformations), which is widely used in Workday integrations, particularly within document transformation systems such as those configured via the Enterprise Interface Builder (EIB) or Document Transformation Connectors. Its primary purpose is to define rules or instructions that dictate how specific nodes in an XML source document should be processed and transformed into the desired output format.
Here's a detailed explanation of why this is the correct answer:
* In XSLT, the < xsl:template > element is used to create reusable transformation rules. It typically includes a match attribute, which specifies the XML node or pattern (e.g., an element, attribute, or root node) to which the template applies. For example, < xsl:template match= " Employee " > would target all < Employee > elements in the source XML.
* Inside the < xsl:template > element, you define the logic-such as extracting data, restructuring it, or applying conditions-that determines how the matched node is transformed into the output. This makes it a core mechanism for controlling the transformation process in Workday integrations.

* In the context of Workday, where XSLT is often used to reformat XML data into formats like CSV, JSON, or custom XML for external systems, < xsl:template > provides the structure for specifying how data from Workday's XML output (e.g., payroll or HR data) is mapped and transformed.
Let's evaluate why the other options are incorrect:
* A. Determine the output file type: The < xsl:template > element does not control the output file type (e.
g., XML, text, HTML). This is determined by the < xsl:output > element in the XSLT stylesheet, which defines the format of the resulting file independently of individual templates.
* B. Grant access to the XSLT language: This option is nonsensical in the context of XSLT. The < xsl:
template > element is part of the XSLT language itself and does not " grant access " to it; rather, it is a functional building block used within an XSLT stylesheet.
* D. Generate an output file name: The < xsl:template > element has no role in naming the output file. In Workday, the output file name is typically configured within the integration system settings (e.g., via the EIB or connector configuration) and is not influenced by the XSLT transformation logic.
An example of < xsl:template > in action might look like this in a Workday transformation:
< xsl:template match= " wd:Worker " >
< Employee >
< Name > < xsl:value-of select= " wd:Worker_Name " / > < /Name >
< /Employee >
< /xsl:template >
Here, the template matches the Worker node in Workday's XML schema and transforms it into a simpler < Employee > structure with a Name element, demonstrating its role in providing rules for node transformation.
Workday Pro Integrations Study Guide: " Configure Integration System - TRANSFORMATION " section, which explains XSLT usage in Workday and highlights < xsl:template > as the mechanism for defining transformation rules.
Workday Documentation: " XSLT Transformations in Workday " under the Document Transformation Connector, noting < xsl:template > as critical for node-specific processing.
W3C XSLT 1.0 Specification (adopted by Workday): Section 5.3, " Defining Template Rules, " which confirms that < xsl:template > provides rules for applying transformations to specified nodes.
Workday Community: Examples of XSLT in integration scenarios, consistently using < xsl:template > for transformation logic.

NEW QUESTION: 14
The following XML code was generated using Core Connector: Location.

You need to validate that both the locc:Location_Name and locc:Municipality elements are not empty, and provide custom error messages with a severity level for each.
Which XSLT attributes and values should you use when producing a pipe-delimited file?
A. locc:Location_Name would need: xtt:required= " true " , xtt:severity= " error " , xtt:name= " Location Name is missing " locc:Municipality would need: xtt:required= " true " , xtt:severity= " error " , xtt:name= " Municipality is missing "
B. locc:Location_Name would need: etv:required= " true " , etv:severity= " error " , etv:name= " Location Name is missing " locc:Municipality would need: etv:required= " true " , etv:severity= " error " , etv:name= " Municipality is missing "
C. locc:Location_Name would need: xtt:required= " true " , xtt:target= " error " , xtt:name= " Location Name is missing " locc:Municipality would need: xtt:required= " true " , xtt:target= " error " , xtt:name= " Municipality is missing "
D. locc:Location_Name would need: etv:required= " true " , etv:target= " error " , etv:name= " Location Name is missing " locc:Municipality would need: etv:required= " true " , etv:target= " error " , etv:name= " Municipality is missing "
Answer: B (LEAVE A REPLY)
This is a validation requirement, not a fixed-width formatting requirement. The goal is to confirm that required XML elements are not empty and to raise custom error messages with a defined severity. Workday's ETV attributes are used for this type of validation behavior. etv:required= " true " marks the value as mandatory, etv:severity= " error " defines the severity level, and etv:name provides the custom validation message name shown when the rule fails. XTT attributes are primarily used for text transformation and formatting, such as fixed length, alignment, padding, and truncation handling. The target attribute is not the correct severity control in this context. Therefore, the ETV required, severity, and name attributes are the correct configuration.

NEW QUESTION: 15
What is a key function and primary benefit of using a Document Transformation Connector within the integration capabilities of Workday?
A. It provides functionality for defining a business process to manage both the connector integrations and document transformations output files.
B. It enables the application of intricate calculations on Workday data before it is extracted by other integration tools for external transmission.
C. It plays a crucial role in converting the XML outputs generated by connector integrations into diverse formats and allows for data formatting and validation.
D. It serves as the principal tool for establishing and maintaining secure connections of connector integrations with various external systems.
Answer: C (LEAVE A REPLY)
The Document Transformation Connector is used in Workday to process and reformat XML outputs - often from Core Connector or EIB integrations - into custom formats like CSV, JSON, or flattened XML.

"The primary role of the Document Transformation Connector is to apply XSLT-based formatting, data reorganization, and validation to the output of Workday integrations before delivery to downstream systems." This is especially useful when third-party vendors require a specific format not natively supported by the integration system.
Why the other options are incorrect:
* A. Managing business processes is not a DT Connector's function.
* B. Calculations are not the main purpose - that's more for Calculated Fields or Studio.
* D. While security is essential, secure connections are managed through Workday's integration system and transport configuration, not the DT connector.
Reference:Workday Pro: Document Transformation Overview - Use Cases for Document Transformation ConnectorWorkday Integration Community: Best Practices for XSLT and Output Formatting

NEW QUESTION: 16
Refer to the following scenario to answer the question below.
You have configured a Core Connector: Worker integration, which utilizes the following basic configuration:
* Integration field attributes are configured to output the Position Title and Business Title fields from the Position Data section.
* Integration Population Eligibility uses the field Is Manager which returns true if the worker holds a manager role.
* Transaction Log service has been configured to Subscribe to specific Transaction Types: Position Edit Event.
You launch your integration with the following date launch parameters (Date format of MM/DD/YYYY):
* As of Entry Moment: 05/25/2024 12:00:00 AM * Effective Date: 05/25/2024
* Last Successful As of Entry Moment: 05/23/2024 12:00:00 AM
* Last Successful Effective Date: 05/23/2024
To test your integration, you made a change to a worker named Jared Ellis who is assigned to the manager role for the IT Help Desk department. You use the Change Business Title related action on Jared and update the Business Title of the position to a new value. Jared Ellis' worker history shows the Title Change Event as being successfully completed with an effective date of 05/24/2024 and an Entry Moment of 05/24/2024 07:58:53 AM however Jared Ellis does not show up in your output. What configuration element would have to be modified for the integration to include Jared Ellis in the output?
A. Transaction log subscription
B. Date launch parameters
C. Integration Field Attributes
D. Integration Population Eligibility
Answer: (SHOW ANSWER)

The scenario involves a Core Connector: Worker integration configured to output Position Title and Business Title fields for workers who meet the Integration Population Eligibility criteria (Is Manager = true), with the Transaction Log service subscribed to the "Position Edit Event." The integration is launched with specific date parameters, and a test is performed by updating Jared Ellis' Business Title using the "Change Business Title" related action. Jared is a manager, and the change is logged with an effective date of 05/24/2024 and an entry moment of 05/24/2024 07:58:53 AM. Despite this, Jared does not appear in the output. Let's determine why and identify the configuration element that needs modification.

In Workday, the Core Connector: Worker integration uses the Transaction Log service to detect changes based on subscribed transaction types. The subscribed transaction type in this case is "Position Edit Event," which is triggered when a position is edited via the "Edit Position" business process. However, the test scenario involves a "Change Business Title" related action, which is a distinct business process in Workday.

This action updates the Business Title field but does not necessarily trigger a "Position Edit Event." Instead, it generates a different event type, such as a "Title Change Event" (as noted in Jared's worker history), depending on how the system logs the action.

The date launch parameters provided are:

* As of Entry Moment:05/25/2024 12:00:00 AM - The latest point for entry moments.
* Effective Date:05/25/2024 - The latest effective date for changes.
* Last Successful As of Entry Moment:05/23/2024 12:00:00 AM - The starting point for entry moments from the last run.
* Last Successful Effective Date:05/23/2024 - The starting point for effective dates from the last run.

Jared's change has:

* Entry Moment:05/24/2024 07:58:53 AM - Falls between 05/23/2024 12:00:00 AM and 05/25/2024 12:
00:00 AM.
* Effective Date:05/24/2024 - Falls between 05/23/2024 and 05/25/2024.

The date parameters correctly cover the time window of Jared's change, meaning the issue is not with the date range but with the event detection logic. The Transaction Log subscription determines which events are processed by the integration. Since the subscription is set to "Position Edit Event" and the change was made via "Change Business Title" (logged as a "Title Change Event"), the integration does not recognize this event because it is not subscribed to the appropriate transaction type.

To include Jared Ellis in the output, theTransaction Log subscriptionmust be modified to include the event type associated with the "Change Business Title" action, such as "Title Change Event" or a broader category like "Position Related Event" that encompasses both position edits and title changes. This ensures the integration captures the specific update made to Jared's Business Title.

Let's evaluate the other options:

* B. Date launch parameters:The parameters already include Jared's entry moment and effective date within the specified ranges (05/23/2024 to 05/25/2024). Adjusting these would not address the mismatch between the subscribed event type and the actual event triggered.
* C. Integration Field Attributes:These are set to output Position Title and Business Title, and the change to Business Title is within scope. The field configuration is correct and does not need modification.
* D. Integration Population Eligibility:This is set to "Is Manager = true," and Jared is a manager. This filter is functioning as intended and is not the issue.
The root cause is the Transaction Log subscription not aligning with the event type generated by the "Change Business Title" action, makingA. Transaction log subscriptionthe correct answer.
Workday Pro Integrations Study Guide References
* Workday Integrations Study Guide: Core Connector: Worker- Section on "Transaction Log Configuration" explains how subscribing to specific transaction types filters the events processed by the integration.
* Workday Integrations Study Guide: Change Detection- Details how different business processes (e.g., Edit Position vs. Change Business Title) generate distinct event types in the Transaction Log.
* Workday Integrations Study Guide: Event Subscription- Notes the importance of aligning subscription types with the specific business actions being tested or monitored.

NEW QUESTION: 17
You have been asked to create a report that will be used by the EIB to output only workers with Child Dependents.
How do you configure the custom report to meet these requirements?
A. Add a Subfilter on the Dependents Business Object with the Field Relationship that is an exact match specified by the filter of Child and a Filter on Dependents with the operator is not empty.
B. Add a Filter on the Worker Business Object with the Field Relationship that prompts the user for the value of Child and a Subfilter on Dependents with the operator is not empty.

C. Add a Subfilter on the Dependents Business Object with the Field Relationship that prompts the user for the value of Child and a Filter on Dependents with the operator is not empty.
D. Add a Filter on the Worker Business Object with the Field Relationship that is an exact match specified by the filter of Child and a Subfilter on Dependents with the operator is not empty.
Answer: A (LEAVE A REPLY)
For an EIB that uses a custom report, the report must return only the correct worker population before the integration extracts the data. Because "Dependents" is a related multi-instance object, the correct approach is to use a subfilter on the Dependents business object to restrict the related dependent records to the Child relationship. Then the report-level filter ensures that the Dependents field is not empty after the subfilter is applied. This combination prevents workers without child dependents from being included. A filter alone on the Worker business object would not correctly evaluate the dependent relationship detail, and prompting the user for "Child" is unnecessary because the requirement is fixed. This is a reporting configuration issue used to control integration output.

NEW QUESTION: 18
A vendor needs an EIB that uses a custom report to output a list of new hires and the date they are eligible for benefits. You have been asked to create a calculated field that adds each worker ' s hire date + 85 days and displays the result in YYYY-MM-DD format.
Which calculated field functions do you need to accomplish this?
A. Date Constant, Arithmetic Calculation, Format Date
B. Numeric Constant, Date Difference, Format Date
C. Date Constant, Increment or Decrement Date, Format Date
D. Numeric Constant, Increment or Decrement Date, Format Date
Answer: (SHOW ANSWER)
You are asked to create a calculated field that:
* Takes the Hire Date
* Adds 85 days
* Formats it as YYYY-MM-DD
To accomplish this in Workday, you need the following calculated field functions:
* Numeric Constant # define 85
* Increment or Decrement Date # add 85 days to the Hire Date
* Format Date # convert the resulting date to YYYY-MM-DD
Why other options are incorrect:
* A. Date Constant would define a fixed date, not a dynamic calculation.
* B. Date Difference is for subtraction between two dates.
* C. Date Constant is still incorrect for offsetting a variable date.

Reference:Workday Calculated Fields Training - Increment or Decrement Date, Format Date FunctionsWorkday Pro: HCM Calculated Fields - Best Practices for Date Arithmetic

NEW QUESTION: 19
You have a population of workers who have put multiple names in their Legal Name - First Name Workday delivered field. Your third-party vendor only accepts one-word first names. For workers that have included a middle name, the first and middle names are separated by a single space. You have been asked to implement the following logic:
* Extract the value before the single space from the Legal Name - First Name Workday delivered field.
* Count the number of characters in the extracted value.
* Identify if the number of characters is greater than.
* If the count of characters is greater than 0, use the extracted value. Otherwise, use the Legal Name - First Name Workday delivered field.
What functions are needed to achieve the end goal?
A. Extract Single Instance, Text Length, Numeric Constant, True/False Condition
B. Text Constant, Substring Text, Arithmetic Calculation, Evaluate Expression
C. Format Text, Convert Text to Number, True/False Condition, Evaluate Expression
D. Substring Text, Text Length, True/False Condition, Evaluate Expression
Answer: D (LEAVE A REPLY)
The task involves processing the "Legal Name - First Name" field in Workday to meet a third-party vendor's requirement of accepting only one-word first names. For workers with multiple names (e.g., "John Paul"), separated by a single space, the logic must:
* Extract the value before the space (e.g., "John" from "John Paul").
* Count the characters in the extracted value.
* Check if the character count is greater than 0.
* Use the extracted value if the count is greater than 0; otherwise, use the original "Legal Name - First Name" field.
This logic is typically implemented in Workday using calculated fields within a custom report or integration (e.g., EIB or Studio). Let's break down the required functions:
* Substring Text:This function is needed to extract the portion of the "Legal Name - First Name" field before the space. In Workday, the Substring Text function allows you to specify a starting position (e.
g., 1) and extract text up to a delimiter (e.g., a space). For example, Substring Text("John Paul", 1, Index of " ") would return "John."
* Text Length:After extracting the substring (e.g., "John"), the logic requires counting its characters to ensure it's valid. The Text Length function returns the number of characters in a text string (e.g., Text Length("John") = 4). This is critical for the condition check.
* True/False Condition:The logic involves a conditional check: "Is the number of characters greater than

0?" The True/False Condition function evaluates this (e.g., Text Length(extracted value) > 0), returning True if the extracted value exists and False if it's empty (e.g., if no space exists or extraction fails).
* Evaluate Expression:This function implements the if-then-else logic: if the character count is greater than 0, use the extracted value (e.g., "John"); otherwise, use the original "Legal Name - First Name" field (e.g., "John Paul"). Evaluate Expression combines the True/False Condition with the output values.
* Option Analysis:
* A. Extract Single Instance, Text Length, Numeric Constant, True/False Condition: Incorrect.
Extract Single Instance is used for multi-instance fields (e.g., selecting one dependent), not text parsing. Numeric Constant isn't needed here, as no fixed number is involved.
* B. Text Constant, Substring Text, Arithmetic Calculation, Evaluate Expression: Incorrect.
Text Constant provides a fixed string (e.g., "abc"), not dynamic extraction. Arithmetic Calculation isn' t required, as this is a text length check, not a numeric operation beyond comparison.
* C. Format Text, Convert Text to Number, True/False Condition, Evaluate Expression: Incorrect.
Format Text adjusts text appearance (e.g., capitalization), not extraction. Convert Text to Number isn't needed, as Text Length already returns a number.
* D. Substring Text, Text Length, True/False Condition, Evaluate Expression: Correct.
These functions align perfectly with the requirements: extract the first name, count its length, check the condition, and choose the output.
* Implementation:
* Create a calculated field using Substring Text to extract text before the space.
* Use Text Length to count characters in the extracted value.
* Use True/False Condition to check if the length > 0.
* Use Evaluate Expression to return the extracted value or the original field based on the condition.
References from Workday Pro Integrations Study Guide:
* Workday Calculated Fields: Section on "Text Functions" details Substring Text and Text Length usage.
* Integration System Fundamentals: Explains how calculated fields with conditions (True/False, Evaluate Expression) transform data for third-party systems.
* Core Connectors & Document Transformation: Highlights text manipulation for outbound integration requirements.

NEW QUESTION: 20
Which three features must all XSLT files contain to be considered valid?
A. A root element, namespace, and at least one transformation
B. A root element, namespace, and at least one template

C. A header, a footer, and a namespace
D. A template, a prefix, and a header
Answer: B (LEAVE A REPLY)
For an XSLT (Extensible Stylesheet Language Transformations) file to be considered valid in the context of Workday integrations (and per general XSLT standards), it must adhere to specific structural and functional requirements. The correct answer is that an XSLT file must contain a root element, a namespace, and at least one template. Below is a detailed explanation of why this is the case, grounded in Workday's integration practices and XSLT specifications:
* Root Element:
* Every valid XSLT file must have a single root element, which serves as the top-level container for the stylesheet. In XSLT, this is typically the <xsl:stylesheet> or <xsl:transform> element (both are interchangeable, though <xsl:stylesheet> is more common).
* The root element defines the structure of the XSLT document and encapsulates all other elements, such as templates and namespaces. Without a root element, the file would not conform to XML well-formedness rules, which are a prerequisite for XSLT validity.
* Example:
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
</xsl:stylesheet>
* Namespace:
* An
XSLT file must declare the XSLT namespace, typically http://www.w3.org/1999/XSL /Transform, to identify it as an XSLT stylesheet and enable the processor to recognize XSLT- specific elements (e.g.,
<xsl:template>, <xsl:value-of>). This is declared within the root element using the xmlns:xsl attribute.
* The namespace ensures that the elements used in the stylesheet are interpreted as XSLT instructions rather than arbitrary XML. Without this namespace, the file would not function as an XSLT stylesheet, as the processor would not know how to process its contents.
* In Workday's Document Transformation integrations, additional namespaces (e.g., for Workday- specific schemas) may also be included, but the XSLT namespace is mandatory for validity.
* At Least One Template:
* An XSLT file must contain at least one <xsl:template> element to define the transformation logic. Templates are the core mechanism by which XSLT processes input XML and produces output. They specify rules for matching nodes in the source XML (via the match attribute) and generating the transformed result.
* Without at least one template, the stylesheet would lack any transformation capability, rendering it functionally invalid for its intended purpose. Even a minimal XSLT file requires

a template to produce meaningful output, though built-in default templates exist, they are insufficient for custom transformations like those used in Workday.
* Example:
<xsl:template match="/">
<result>Hello, Workday!</result>
</xsl:template>
Complete Minimal Valid XSLT Example:
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<output>Transformed Data</output>
</xsl:template>
</xsl:stylesheet>
Why Other Options Are Incorrect:
* A. A root element, namespace, and at least one transformation: While this is close, "transformation" is not a precise term in XSLT. The correct requirement is a "template," which defines the transformation logic. "Transformation" might imply the overall process, but the specific feature required in the file is a template.
* C. A header, a footer, and a namespace: XSLT files do not require a "header" or "footer." These terms are not part of XSLT or XML standards. The structure is defined by the root element and templates, not headers or footers, making this option invalid.
* D. A template, a prefix, and a header: While a template is required, "prefix" (likely referring to the namespace prefix like xsl:) is not a standalone feature-it's part of the namespace declaration within the root element. "Header" is not a required component, making this option incorrect.
Workday Context:
* In Workday's Document Transformation systems (e.g., Core Connectors or custom integrations), XSLT files are uploaded as attachment transformations. Workday enforces these requirements to ensure the stylesheets can process XML data (e.g., from Workday reports or connectors) into formats suitable for external systems. The Workday platform validates these components when an XSLT file is uploaded, rejecting files that lack a root element, namespace, or functional templates.
Workday Pro Integrations Study Guide References:
* Workday Integration System Fundamentals: Describes the structure of XSLT files, emphasizing the need for a root element (<xsl:stylesheet>), the XSLT namespace, and templates as the building blocks of transformation logic.
* Document Transformation Module: Details the requirements for uploading valid XSLT files in Workday, including examples that consistently feature a root element, namespace declaration, and at least one template (e.g., "XSLT Basics for Document Transformation").
* Core Connectors and Document Transformation Course Manual: Provides sample XSLT files used in labs, all of which include these three components to ensure functionality within Workday integrations.

* Workday Community Documentation: Reinforces that XSLT files must be well-formed XML with an XSLT namespace and at least one template to be processed correctly by Workday's integration engine.

NEW QUESTION: 21
A benefits provider requests a file containing updated compensation data from your Workday tenant. They want to receive this file at exactly 5:00 PM Eastern Time every weekday, but not on weekends. The file should include only compensation changes that occurred since the last integration run.
How should you configure the run frequency of the integration schedule to meet these requirements?
A. Configure to recur every weekday
B. Configure to recur every 1 day
C. Configure to recur every 5 days
D. Configure to recur every 7 days
Answer: A (LEAVE A REPLY)
The schedule must run every weekday and exclude weekends. "Configure to recur every weekday" is the only option that directly matches Monday through Friday processing. Configuring the schedule to recur every one day would run seven days a week unless additional restrictions were added, so it does not meet the "not on weekends" requirement. Recurring every five days is also incorrect because it creates a rolling five-day interval, not a Monday-to-Friday pattern. Recurring every seven days would run weekly, not every weekday.
The phrase "since the last integration run" indicates incremental change processing, but the question is specifically asking for run frequency. Therefore, the weekday recurrence is the correct scheduling configuration.

NEW QUESTION: 22
Refer to the scenario. You are implementing a Core Connector: Worker integration to send employee data to a third-party active employee directory. The external vendor requires the following:
* The Employee's Active Directory User Principal Name.
* A mapping from Worker Type values to external worker type codes.
* A specific filename format that includes a timestamp and sequence number.
You also need to ensure the document transformation occurs before the file is delivered to the endpoint. You must include an Employee's Active Directory User Principal Name (generated by a Calculated Field).
How do you ensure this field is pulled into the output?
A. Configure an integration map.
B. Configure an integration field override.
C. Configure an integration field attribute.

D. Configure an integration attribute.

Answer: B (LEAVE A REPLY)

To surface a Calculated Field in a Core Connector: Worker (CCW) outbound, you use an Integration Field Override to substitute the connector's default source with your calculated value. An integration map (Option A) is intended to translate or normalize code values (for example, mapping internal Worker Type codes to the vendor's codes), not to replace the source of a field. Integration attributes (Option D) and integration field attributes (Option C) manage connector behavior and attributes, but they do not replace a field's data source with a calculated field. Therefore, the correct method to "pull" a calculated field into the CCW output is an Integration Field Override (Option B).

Why the other elements in the scenario matter (and how they're handled) - with exact extracts from your materials:

* Mapping Worker Type to external codes # Integration Maps (supports, but not the asked action):Your deployment guides call out maintaining and using Integration System Maps for code translations. This is exactly where you'd map "Worker Type" to the external system's codes, but it is not how you inject a calculated field into the payload.

"Maintenance of Integration System Maps"

"WORKDAY SETUP - NON STATIC MAPS" and "WORKDAY SETUP - STATIC MAPS" (table of contents for configuration of maps)

* Filename requires timestamp/sequence number # Sequence Generator (supports the scenario):Your Time Tracking/PECI deployment guide explicitly includes a Sequence Generator configuration that's used with certified connectors to build compliant, unique file names (often with timestamps and/or sequence numbers) before delivery.

"3.6 Sequence Generator" (configuration item for certified integrations used in file naming)

* Transformation before delivery # Standard integration flow (transform then deliver):The same deployment materials describe document/file delivery mechanics (for example, SFTP), which occur after the integration produces/transforms the document. This supports the scenario requirement that transformation happens prior to transmission.

"4. FILE DELIVERY SERVICE ... 4.4 SFTP Configuration" (document delivery occurs after the integration generates/transforms the output)

* Security posture for integrations (context):For outbound/system users and secure delivery, the Workday Authentication & Security guide documents integration-appropriate authentication (e.g., X.509) and general integration security steps - relevant background for productionizing CCW but not directly affecting how to bring a calculated field into the payload.

"X509 Recommended for web services users and integrations that use an integration system user account." Putting it all together for the scenario:

* Use Integration Field Override to point the CCW field to your Calculated Field for UPN # (Correct answer: B).

* Use Integration Maps to translate Worker Type to the vendor's codes (supports the mapping requirement).

* Configure filename rules via Sequence Generator to include timestamp and sequence in the produced file name (supports the file-naming requirement).
* Ensure the document transformation runs as part of the integration generation step and then deliver via SFTP (file delivery service).
References (Workday Pro: Integrations-aligned materials):
* GPC_PECI_TimeTracking_DeploymentGuide_CloudPay.pdf - Sections "3.6 Sequence Generator" and "4. File Delivery Service" (delivery occurs after file generation/transform).
* GPC_PECI_DeploymentGuide_CloudPay_2.9.pdf - Map configuration sections ("WORKDAY SETUP - NON STATIC MAPS", "WORKDAY SETUP - STATIC MAPS").
* GPC_PECI_UserGuide_CloudPay_2.1.1.pdf - "Maintenance of Integration System Maps."
* Admin-Guide-Authentication-and-Security.pdf - Integration security notes, including X.509 recommendation for integrations.

NEW QUESTION: 23
Refer to the following scenario to answer the question below.
You are configuring a filename sequence generator for a connector. Below are common pattern tokens for timestamps ranging from the year to the millisecond.
Define the sequence format using a combination of string constants and pattern tokens to create a unique identifier. Note the example tokens include the square brackets.
For the next sequence number: [Seq] or [seq]
Assume date of September 21, 2022, 12:35:59:123 PM. Some common tokens:
Year: [yyyy] = 2022, [yy] = 22
Month: [MMM] = Sep, [MM] = 09, [M] = 9
Day: [d] = 21, [E] = Wed, [D] = 265
Hours: [k] = 13, [h] = 1
Minutes: [m] = 35
Seconds: [s] = 59
Milliseconds: [S] = 123
What pattern will generate the hour minute second timestamp format of "4:35:15"?
A. [h:M:s]
B. [h:m:s]
C. [h]-[m] -[s]
D. [h]:[M] :[s]
Answer: B (LEAVE A REPLY)
The required timestamp format is hour, minute, and second separated by colons. In the pattern token list, [h] represents the hour value, [m] represents minutes, and [s] represents seconds. Therefore, [h:m:s] is the correct pattern because it uses the correct time tokens and the correct colon separators. Option A and option D incorrectly use uppercase [M], which represents month, not minutes. Option C uses the correct hour, minute, and second tokens, but the separators are hyphens instead of colons, so it would not generate the

requested timestamp format. Filename and sequence generator patterns are strict, so both token case and literal separators must match the required output exactly.

NEW QUESTION: 24
Refer to the following scenario to answer the question below.
You are configuring a Core Connector: Worker integration to send data to a new external compliance and certification tracking vendor. You have begun to configure the connector with the Data Initialization Service (DIS) enabled. Your goal is to extract worker qualification data, but the vendor has three specific requirements:
The file must only include Active workers who are in the "Clinical Staff" Job Family.
The vendor has specified that for each worker's Education data, they only want to receive the Institution Name, Institution Type and Degree.
The vendor requires a custom "License ID" that must combine the Certification Name and Issuing State, for example, "RN-CA". A Calculated Field that provides this custom "License ID" already exists in the tenant.
During testing, using a Full File run, the output file is missing all education-related information, even though you believe you have initially configured field attributes such as Institution Name, Institution Type and Degree in the Configure Integration Field Attributes step. Additionally, you confirmed that the Worker Qualification Data Section is marked as Include in Output.
What should you do to resolve this issue?
A. Enable the Education subfolder in Configure Integration Field Attributes and then reselect the Institution Name, Institution Type, and Degree fields.
B. Within the Configure Integration Services task, select the Enable All Services checkbox.
C. Enable the Worker Qualification Data Section Fields Service within the Configure Integration Services step.
D. Mark each education field in the Education subfolder as Required in Configure Integration Field Attributes.
Answer: C (LEAVE A REPLY)
For Core Connector: Worker, selecting fields in Integration Field Attributes is not always enough if the related integration service is not enabled. Worker qualification information, including education-related data, is controlled by the Worker Qualification Data Section Fields Service. If that service is not enabled, the connector does not extract the section's data, even when individual field attributes appear to be selected or the section is marked for output. Enabling every service is excessive and not the controlled configuration choice. Marking fields as required affects validation or required output behavior, but it does not activate the underlying qualification service. The correct fix is to enable the specific service that supplies the worker qualification data to the connector output.

NEW QUESTION: 25

A vendor needs to create a Date Difference calculated field. However, the two dates needed for that calculation are on two separate business objects.
What additional calculated field do you need to create that Date Difference calculated field?
A. Lookup Related Value
B. Build Date
C. Lookup Date Rollup
D. Lookup Value as of Date
Answer: A (LEAVE A REPLY)
When creating a Date Difference calculated field in Workday, both dates must exist on the same business object. If they are on different business objects, you need to first bring the second date onto the primary object. To do that, you use a:
Lookup Related Value calculated field - this allows you to retrieve a field (like a date) from a related business object, so it can then be used in further calculations.
Example scenario:
* You want to subtract Hire Date (on the Worker object) from Dependent's Birth Date (on the Dependent object).
* These are on different objects # use Lookup Related Value to pull the second date into the current object context.
* Then, create the Date Difference using both dates on the same object.
Why other options are incorrect:
* B. Build Date creates a synthetic date, not for bridging objects.
* C. Lookup Date Rollup rolls up values across multiple related objects, not typically used for 1-to-1 value bridging.
* D. Lookup Value as of Date is used for time-sensitive lookups (e.g., point-in-time values), not structural bridging.
Reference:Workday Pro: Calculated Fields - Working Across Business Objects with Lookup Related ValueWorkday Community: Bringing Dates Across Objects to Support Date Difference Calculations

NEW QUESTION: 26
How does an XSLT processor identify the specific nodes in an XML document to which a particular transformation rule should be applied?
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 (LEAVE A REPLY)
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.
Reference:W3C XSLT 1.0 Specification - xsl:template and XPath MatchingWorkday Integration Training -
"How XSLT Applies Rules to XML Output"

NEW QUESTION: 27
You are developing a Core Connector: Worker integration with DIS enabled. You configure the Population Eligibility to include all US workers and also define the Eligibility Criterion field to return only those workers assigned to the Sales organization.
When you run the integration, the output includes all US workers instead of just those in Sales.
What should you do to ensure only workers in the Sales organization appear in the output?
A. Remove the Eligibility Criteria field and apply the filtering logic in the Population Eligibility.
B. Update the Eligibility Criteria to a valid field and rerun the integration.
C. Remove the Population Eligibility step and use only Eligibility Criteria for filtering.
D. Keep both the Population Eligibility and Eligibility Criteria as is and run with Full File enabled.
Answer: A (LEAVE A REPLY)
Population Eligibility controls which workers are included in the Core Connector output population. In this scenario, the Population Eligibility is configured to include all US workers, so the connector correctly outputs that entire eligible population. The Eligibility Criterion field does not replace Population Eligibility as the population filter. It can support eligibility-related logic, but it is not the correct place to enforce the final worker population restriction when the actual extract should include only Sales workers. To fix the output, the Sales organization condition must be moved into the Population Eligibility rule itself. Updating the criterion field or running a full file would not solve the problem because the population filter would still include all US workers. The integration population must be filtered at the source.

NEW QUESTION: 28
Refer to the following scenario to answer the question below. You have configured a Core Connector: Worker integration, which utilizes the following basic configuration:
* Integration field attributes are configured to output the Position Title and Business Title fields from the Position Data section.

* Integration Population Eligibility uses the field Is Manager which returns true if the worker holds a manager role.
* Transaction Log service has been configured to Subscribe to specific Transaction Types: Position Edit Event. You launch your integration with the following date launch parameters (Date format of MM/DD
/YYYY):
* As of Entry Moment: 05/25/2024 12:00:00 AM
* Effective Date: 05/25/2024
* Last Successful As of Entry Moment: 05/23/2024 12:00:00 AM
* Last Successful Effective Date: 05/23/2024
To test your integration, you made a change to a worker named Jared Ellis who is assigned to the manager role for the IT Help Desk department. You perform an Edit Position on Jared and update their business title to a new value. Jared Ellis ' worker history shows the Edit Position Event as being successfully completed with an effective date of 05/27/2024 and an Entry Moment of 05/24/2024 07:58:53 AM however Jared Ellis does not show up in your output. What configuration element would have to be modified for the integration to include Jared Ellis in the output?
A. Integration Population Eligibility
B. Date launch parameters
C. Integration Field Attributes
D. Transaction log subscription
Answer: B (LEAVE A REPLY)
The scenario describes a Core Connector: Worker integration configured to output Position Title and Business Title fields for workers who meet the Integration Population Eligibility criteria (Is Manager = true), with the Transaction Log service subscribed to the " Position Edit Event. " The integration is launched with specific date parameters, and a test is performed by updating Jared Ellis' Business Title via an " Edit Position " action.
Jared is a manager, and the change is logged with an effective date of 05/27/2024 and an entry moment of 05
/24/2024 07:58:53 AM. Despite this, Jared does not appear in the output. Let's analyze why and determine the configuration element that needs modification.
In Workday, the Core Connector: Worker integration relies on the Transaction Log service to detect changes based on subscribed transaction types and processes them according to the date launch parameters. The integration is configured as an incremental run (since " Last Successful " parameters are provided), meaning it captures changes that occurred since the last successful run, within the specified date ranges. The date launch parameters are:
* As of Entry Moment: 05/25/2024 12:00:00 AM - The latest point for when changes were entered into the system.
* Effective Date: 05/25/2024 - The latest effective date for changes to be considered.

* Last Successful As of Entry Moment: 05/23/2024 12:00:00 AM - The starting point for entry moments from the last run.
* Last Successful Effective Date: 05/23/2024 - The starting point for effective dates from the last run.
For an incremental run, Workday processes changes where:
* The Entry Moment falls between the Last Successful As of Entry Moment (05/23/2024 12:00:00 AM) and the As of Entry Moment (05/25/2024 12:00:00 AM), and
* The Effective Date falls between the Last Successful Effective Date (05/23/2024) and the Effective Date (05/25/2024).
Now, let's evaluate Jared Ellis' change:
* Entry Moment: 05/24/2024 07:58:53 AM - This falls within the range of 05/23/2024 12:00:00 AM to 05
/25/2024 12:00:00 AM, so the entry timing is captured correctly.
* Effective Date: 05/27/2024 - This is after the Effective Date of 05/25/2024 specified in the launch parameters.
The issue arises with the Effective Date. The integration only processes changes with an effective date between 05/23/2024 (Last Successful Effective Date) and 05/25/2024 (Effective Date). Jared's change, with an effective date of 05/27/2024, falls outside this range. In Workday, the effective date determines when a change takes effect, and incremental integrations rely on this date to filter relevant transactions. Even though the entry moment (when the change was entered) is within the specified window, the effective date being in the future (relative to the integration's Effective Date of 05/25/2024) excludes Jared from the output.
To include Jared Ellis in the output, the Date launch parameters must be modified. Specifically, the Effective Date needs to be adjusted to a date that includes 05/27/2024 (e.g., 05/27/2024 or later). This ensures the integration captures changes effective up to or beyond Jared's edit. Alternatively, if the intent is to process future-dated changes entered within the current window, the integration could be adjusted to consider the entry moment as the primary filter, though this would typically require a different configuration approach (e.
g., full file mode or a custom report, not standard incremental behavior).
Let's evaluate the other options:
* A. Integration Population Eligibility: Set to " Is Manager = true, " and Jared is a manager. This filter is correct and does not need modification.
* C. Integration Field Attributes: Configured to output Position Title and Business Title, and the change to Business Title is within scope. The field configuration is appropriate.
* D. Transaction log subscription: Subscribed to " Position Edit Event, " which matches the " Edit Position " action performed on Jared. The subscription type is correct.
The mismatch between the integration's Effective Date (05/25/2024) and Jared's change effective date (05/27
/2024) is the reason for exclusion, making B. Date launch parameters the correct answer.

Workday Pro Integrations Study Guide References
* Workday Integrations Study Guide: Core Connector: Worker - Section on " Change Detection " explains how effective dates and entry moments govern incremental processing.
* Workday Integrations Study Guide: Launch Parameters - Details the roles of " Effective Date " and " As of Entry Moment " in filtering changes, emphasizing that incremental runs focus on the effective date range.
* Workday Integrations Study Guide: Incremental Processing - Describes how future-dated changes (effective dates beyond the launch parameter) are excluded unless the parameters are adjusted accordingly.

NEW QUESTION: 29
What XSL component is required to execute valid transformation instructions in the XSLT code?
A. xsl:template
B. xsl:apply-template
C. xsl:call-template
D. xsl:output
Answer: (SHOW ANSWER)
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.
Reference:W3C XSLT Specification - Section: xsl:template Required for ExecutionWorkday XSLT Examples - "Template-Based Transformations in Workday"

NEW QUESTION: 30
What attribute(s) can go into the xsl:stylesheet element?
A. XSLT Version & Namespaces
B. XSLT Version & Encoding
C. Namespaces & Encoding
D. XML Version & Namespaces
Answer: A (LEAVE A REPLY)

NEW QUESTION: 31
The following XML code was generated through a RaaS that will be used in an EIB.

Within a template that matches on wd:Report_Entry, what XPath expression do you use to select the value of the Relationship_ID element?

A. wd:Dependents_Group/wd:Relationship/wd:ID/wd:type= ' Relationship_ID '

B. wd:Dependents_Group/wd:Relationship/wd:ID/wd:type= ' Relationship_ID '

C. wd:Dependents_Group/wd:Relationship/wd:ID

D. ./wd:Dependents_Group/wd:Relationship/wd:ID

Answer: D (LEAVE A REPLY)

The XML fragment shown follows the ReportasaService (RaaS) structure typical for Workday custom report output:

< wd:Report_Entry >

< wd:Dependents_Group >

< wd:Name > Megan McNeil < /wd:Name >

< wd:Age > 25 < /wd:Age >

< wd:Relationship wd:Descriptor= " Child " >

< wd:ID wd:type= " WID " > 7507df6a99664ce7bc0cb902cf370543 < /wd:ID >

< wd:ID wd:type= " Relationship_ID " > Child < /wd:ID >

< /wd:Relationship >

< /wd:Dependents_Group >

< /wd:Report_Entry >

Inside each < wd:Report_Entry > , the target value Relationship_ID resides under:

wd:Dependents_Group

wd:Relationship

wd:ID (wd:type= " Relationship_ID ")

When writing the template:

< xsl:template match= " wd:Report_Entry " >

XSLT uses a relative XPath (starting with ./) to navigate from the matched node.

Therefore, the correct XPath should be:
/wd:Dependents_Group/wd:Relationship/wd:ID
That expression selects the wd:ID element so you can then test/extract where wd:type= " Relationship_ID " .
Why the other options are incorrect:
Option
Why Incorrect
A & B
These use an equality test incorrectly inside the XPath expression - they would not return the node value and are syntactically invalid for value extraction.
C
Missing ./ - would still work in many cases, but Workday XSLT best practice is to use relative paths when inside a match.
Workday Pro Integration guidance for RaaS/XSLT stresses:
Always scope node selection relative to the current context tree using prefixqualified XPath expressions.
Reference:Workday Pro: Integrations - XSLT for Workday XML (Namespaces, Context Node, Relationship ID Extraction)W3C XSLT Specification - Relative XPath from context node

NEW QUESTION: 32
You are creating a report that needs a data source of All Active and Terminated Workers. However, when searching for that data source it is not showing in the prompt field.
Why is All Active and Terminated Workers not showing as an option?
A. Optimized for Performance is checked.
B. The incorrect report type is selected.
C. The report is enabled as a temporary report.
D. Enabled as a Web Service is checked.
Answer: (SHOW ANSWER)
In Workday Report Writer, the available data sources depend heavily on the selected report type and the reporting context. If the required data source, such as All Active and

Terminated Workers, does not appear in the prompt, the most likely cause is that the wrong report type has been selected during report creation.
Workday filters the available data sources based on compatibility with the selected report type. Enabling a report as a web service does not remove data sources from the prompt, and temporary report settings do not determine whether this worker data source appears. Optimized for Performance affects report execution and performance behavior, not whether the core data source is selectable. Correcting the report type allows the proper worker data source to appear.

NEW QUESTION: 33
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?
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 (LEAVE A REPLY)
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 writelevel 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.
References:Workday Pro: Integrations - Integration Security and Report Ownership
RulesAdminGuideAuthenticationandSecurity.pdf - Security Policies & Required Permissions Model

NEW QUESTION: 34

After you transfer ownership of an Integration System to an ISU, what other component, if it exists, must you transfer ownership of to ensure the integration continues to run in an automated fashion?
A. Integration Schedule
B. Integration Maps
C. Integration Attributes
D. Integration Field Attributes
Answer: A (LEAVE A REPLY)
When an integration is moved to an Integration System User, ownership must support unattended execution.
The integration system itself can be owned by the ISU, but if an integration schedule exists, that schedule also needs to be owned by the ISU or transferred appropriately. Otherwise, the automated run can fail or continue to depend on the original human owner's security context. Integration Maps, Integration Attributes, and Integration Field Attributes are configuration components inside the integration; they do not independently control scheduled execution ownership. The schedule is the object responsible for recurring automated launches, so it must align with the service account that owns and runs the integration. This is a security and operational control for stable Workday integration execution.

NEW QUESTION: 35
As of May 1, 2024 Brian Hill's annual salary is $60,000.00. On May 13, 2024 Brian Hill received a salary increase and data was entered into Workday at 2:00 PM the same day. The new salary amount is set to
$90,000.00 with an effective date of May 10, 2024.
Run #1
* Core Connector: Worker Integration System was launched as an ad-hoc manual run on May 13, 2024.
* As of Entry Moment: 05/11/2024 2:00:00 PM
* Effective Date: 05/11/2024
* Last Successful As of Entry Moment: 05/09/2024 2:00:00 PM
* Last Successful Effective Date: 05/09/2024
What will be the expected output in the Run #1 of the Core Connector: Worker Integration System?
A. Brian Hill will be excluded in the output file due to the Effective Date of his salary.
B. Brian Hill will be included in the output file. The salary amount will be $60,000.00.
C. Brian Hill will be excluded in the output file due to the Entry Moment of his salary.
D. Brian Hill will be included in the output file. The salary amount will be $90,000.00.
Answer: D (LEAVE A REPLY)
Let's break this down:
* Effective Date of salary change: May 10, 2024

* Entry Moment (data entry timestamp): May 13, 2024, 2:00 PM
* Integration Run As of Entry Moment: May 11, 2024, 2:00 PM
* Salary data was entered AFTER this moment (May 13 vs May 11)
So based on Workday's Change Detection logic:
A worker is included in the integration output only if the transaction was entered into Workday after the last successful entry moment, and the effective date is on or after the "Last Successful Effective Date".
In this case:
* Entry was made after the last As-of Entry Moment (May 13 > May 11)
* Effective date (May 10) is after the last successful effective date (May 9) Both conditions are met, so Brian Hill will be included, and the new salary of $90,000.00 will be reflected in the output.
Why other options are incorrect:
* A. The effective date is valid.
* B. $60,000 would be outdated.
* C. Entry moment is after the As-of date, so not excluded.
Reference:Workday Pro: Core Connector Change Detection - How Entry Moment and Effective Date Determine Output

NEW QUESTION: 36
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.
While testing, the output contains no records, and the Messages tab shows exception logs stating you don't have access to the Exempt field. You note this is the same field being used for Population Eligibility in the integration.
What must you configure to resolve this security issue?
A. Assign the ISSG to a row with Modify access in the domain security policy securing the Web Service.
B. Assign the ISSG to a row with View access in the domain security policy securing the Web Service.
C. Assign the ISSG to a row with Modify access in the domain security policy securing the Population Eligibility field.
D. Assign the ISSG to a row with View access in the domain security policy securing the Population Eligibility field.
Answer: (SHOW ANSWER)
The Exempt field is being used in Population Eligibility, and eligibility fields must be readable by the ISSG.
If the domain security policy for a field denies View access, Workday cannot evaluate the eligibility and returns no data.

From Workday security governance:
"For integrations using Population Eligibility, the ISSG must have View permission on all fields referenced in eligibility rules." If View is missing, the eligibility rule cannot execute # No workers are considered eligible # Output contains zero records # Error logged for denied field access.
Therefore, the solution is:
* Grant the ISSG View access to the domain that secures the Population Eligibility field
Modify access (A/C) is not needed - eligibility only needs read-access.
References:Workday Pro: Integrations - Population Eligibility Security RequirementsAdmin#Guide#Authentication#and#Security.pdf - View permission required to access report
/integration data fields

NEW QUESTION: 37
Which components make up the three primary parts of an outbound Enterprise Interface Builder (EIB)?
A. One data source
One transformation
One delivery endpoint
B. Multiple data sources
One transformation
One delivery endpoint
C. One web service call
One sequence generator
One delivery endpoint
D. One data source
One transformation
Multiple delivery endpoints
Answer: A (LEAVE A REPLY)
An outbound EIB is organized around three main configuration components: Get Data, Transform, and Deliver. The Get Data section uses one data source, commonly a custom report or another supported source.
The Transform section applies one transformation, such as an XSLT transformation, when formatting is required. The Deliver section sends the output to one delivery endpoint, such as SFTP, email, or another supported transport. Multiple data sources or multiple delivery endpoints are not the standard three-part model for a basic outbound EIB. A web service call and sequence generator may be used in some integration scenarios, but they are not the three primary EIB components. Therefore, option A matches the EIB architecture.

NEW QUESTION: 38

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

A. xsl:template

B. xsl:apply-template

C. xsl:call-template

D. xsl:output

Answer: A (LEAVE A REPLY)

The required XSLT component is xsl:template. XSLT transformations are driven by templates, which define the transformation rules that apply to matched XML nodes. A template can match a source element or be called by name, and it contains the instructions that produce the transformed output. xsl:apply-template is not the correct element name; the correct XSLT instruction is xsl:apply-templates, and it is used to invoke template processing on selected nodes. xsl:call-template calls a named template but does not by itself define the rule being executed. xsl:output controls serialization settings such as output method or encoding. For valid transformation logic, the core executable rule container is xsl:template.

NEW QUESTION: 39

Refer to the following scenario to answer the question below.

An external system needs a file containing data for recent worker job changes. They would like to receive a file routinely at 5:00 PM Eastern Standard Time, every 48 hours. The file should show job changes since the last integration run.

What is the recurrence type of the integration schedule?

A. Recurs every 2 day(s)

B. Custom Recurrence

C. Recurs every 48 hours

D. Dependent Recurrence

Answer: A (LEAVE A REPLY)

The requirement is to run the integration every 48 hours at a fixed time. In schedule configuration terms, 48 hours is equivalent to every two days. Therefore, the correct recurrence type is "Recurs every 2 day(s)."

"Recurs every 48 hours" sounds mathematically equivalent, but it is not the schedule recurrence type shown for this configuration. Custom Recurrence would only be needed if the schedule pattern could not be represented by the standard recurrence options. Dependent Recurrence is used when an integration should run based on another process or dependency, which is not stated here. Since the file must include changes since the last run, the two-day recurrence controls how often incremental job changes are extracted.

NEW QUESTION: 40

You have been asked to create an integration using the Core Connector: Worker with DIS template. The vendor has requested that you only include employees who are based in the San Francisco area that are on leave.
How do you configure your integration so that only workers who meet the requirements are included in the output file?
A. Configure a Boolean field for San Francisco workers on leave in the field overrides.
B. Configure a Boolean field for Population Eligibility for San Francisco workers on leave.
C. Configure the integration attributes to include workers in San Francisco on leave.
D. Configure a Boolean field for San Francisco workers on leave under the field attributes.
Answer: (SHOW ANSWER)
When using Core Connector: Worker with DIS, to restrict the population to employees who:
* Are on leave, and
* Are located in San Francisco
You must configure Population Eligibility, which is the only place to filter the worker population included in the connector output.
From Workday Pro documentation:
"The Population Eligibility section defines which workers are eligible for extraction in the integration based on location, status, organization, and other conditions. Boolean calculated fields can be used here to define complex eligibility criteria." In this case:
* Create a Boolean calculated field that returns true for "On Leave AND Location = San Francisco"
* Use that field in Population Eligibility
Why the others are incorrect:
* A, D. Field Overrides and Field Attributes only modify what data is extracted-not who is included.
* C. Integration Attributes don't control population filtering.
Reference:Workday Pro: Core Connector Worker - Population Eligibility and Filtering LogicWorkday Community - Using Boolean Fields in Population Eligibility Rules

NEW QUESTION: 41
The following XML code was generated through a RaaS that will be used in an EIB.
You want to use predicated templates that will process USA workers one way, GBR workers another way, and all other countries a standard way.
What XML code will create these templates?
A. < xsl:template match= " wd:Report_Entry[wd:Country_Code = ' USA '] " >
...
< /xsl:template >
< xsl:template match= " wd:Report_Entry[wd:Country_Code = ' GBR '] " >
...
< /xsl:template >

< xsl:template match= " wd:Report_Entry[not(wd:Country_Code = ' USA ' or wd:Country_Code = ' GBR ')] " >
...
< /xsl:template >
B. < xsl:template match= " wd:Report_Entry/wd:Country_Code[' USA '] " >
...
< /xsl:template >
< xsl:template match= " wd:Report_Entry/wd:Country_Code[' GBR '] " >
...
< /xsl:template >
< xsl:template match= " wd:Report_Entry/wd:Country_Code[not(' USA ' or ' GBR ')] " >
...
< /xsl:template >
C. < xsl:template match= " wd:Report_Entry[wd:Country_Code = ' USA ' or wd:Country_Code = ' GBR '
] " >
...
< /xsl:template >
< xsl:template match= " wd:Report_Entry[wd:Country_Code = ' USA '] " >
...
< /xsl:template >
< xsl:template match= " wd:Report_Entry[wd:Country_Code = ' GBR '] " >
...
< /xsl:template >
D. < xsl:template match= " wd:Report_Entry/wd:Country_Code = ' USA ' " >
...
< /xsl:template >
< xsl:template match= " wd:Report_Entry/wd:Country_Code = ' GBR ' " >
...
< /xsl:template >
< xsl:template match= " wd:Report_Entry/not(wd:Country_Code = ' USA ' or wd:Country_Code = ' GBR ') " >
...
< /xsl:template >
Answer: (SHOW ANSWER)
Predicated templates use XPath predicates inside square brackets to match only specific source nodes. The template must match the wd:Report_Entry node and then apply a predicate against the child value wd:
Country_Code. Option A correctly creates three separate template matches: one for USA records, one for GBR records, and one fallback template for records where the country code is neither USA nor GBR. This is the cleanest XSLT pattern because each template

processes a specific population of report entries. Options B and D incorrectly place the predicate or comparison after navigating directly to wd:Country_Code, which does not properly match the full report entry node. Option C combines USA and GBR in the first template and would not separate processing correctly.

NEW QUESTION: 42
A vendor needs an EIB that uses a custom report to output data for payroll results. You have been asked to create a calculated field that will be used to output the highest deduction from a worker's payroll results.
Which calculated field functions do you use to accomplish this?
A. True/False Condition(s), Extract Single Instance
B. True/False Condition(s), Evaluate Expression
C. Text Constant, True/False Condition(s), Evaluate Expression
D. Text Constant, True/False Condition(s), Extract Single Instance
Answer: A (LEAVE A REPLY)
Payroll results usually contain multiple related rows, so the report must identify the correct payroll result instance before outputting a single value. A True/False Condition is used to define which deduction rows qualify for evaluation, such as identifying valid deduction results. Extract Single Instance is then used to return one instance from that qualified set, typically based on a sort or selection rule that identifies the highest deduction. Evaluate Expression is unnecessary because the requirement is not an if/then output selection; it is selecting one related instance from a multi-instance group. Text Constant is also irrelevant because no fixed text value is being added to the output. This is a data transformation problem inside a report-driven EIB.

NEW QUESTION: 43
The following XML code was generated using Core Connector: Location.
You need to format the locc:Entry_Date element for both US (month/day/year) and EU (day.month.year) date styles while also ensuring that the locc:Entry_Date field contains a value when producing a CSV file.
Which combination of attributes and values should you use to create these classes?
A. < xtt:class xtt:name= " UsDateFormat " xtt:dateFormat= " MM/dd/yyyy " xtt:required= " true " / >
< xtt:class xtt:name= " EuDateFormat " xtt:dateFormat= " dd.MM.yyyy " xtt:required= " true " / >
B. < etv:class etv:name= " UsDateFormat " etv:dateFormat= " MM/dd/yyyy " etv:required= " true " / >
< etv:class etv:name= " EuDateFormat " etv:dateFormat= " dd.MM.yyyy " etv:required= " true " / >
C. < xtt:class xtt:name= " UsDateFormat " xtt:dateFormat= " [M01]/[D01] /[Y0001] " xtt:severity= " error

" / >
< xtt:class xtt:name= " EuDateFormat " xtt:dateFormat= " [D01].[M01] .[Y0001] "
xtt:severity= " error
" / >
D. < etv:class etv:name= " UsDateFormat " etv:dateFormat= " [M01]/[D01] /[Y0001] "
etv:severity= " error " / >
< etv:class etv:name= " EuDateFormat " etv:dateFormat= " [D01].[M01] .[Y0001] "
etv:severity= " error " / >
Answer: A (LEAVE A REPLY)
The requirement combines CSV output formatting with a required-value check for date fields. For Workday transformation formatting classes, XTT is the correct attribute family. xtt:class defines reusable transformation formatting rules, xtt:name names the class, and xtt:dateFormat controls the rendered date pattern. The required US format is MM/dd/yyyy, and the EU format is dd.MM.yyyy. xtt:required= " true " ensures the Entry Date has a value when the output is produced. ETV attributes are used for validation rules rather than transformation formatting classes in this context. The bracketed date tokens in options C and D do not match the required dateFormat style shown for these XTT classes. Therefore, option A is the correct configuration.

NEW QUESTION: 44
You have been asked to refine a report which outputs one row per worker and is being used in an integration that sends worker data to one of your third-party systems. The integration should only send workers who have been hired in the last 30 days. Where in the custom report definition can you specify a condition that would include only workers who have been hired in the last 30 days?
A. Subfilter
B. Output
C. Columns
D. Filter
Answer: (SHOW ANSWER)
In Workday, when refining a custom report to include specific conditions such as limiting the output to workers hired in the last 30 days, the appropriate place to specify this condition is within the Filter tab of the custom report definition. The Filter tab allows you to define criteria that determine which instances of the primary business object (in this case, " Worker ") are included in the report output. This is critical for integrations, as the filtered data ensures that only relevant records are sent to the third-party system.
The requirement here is to restrict the report to workers hired within the last 30 days. In Workday reporting, this can be achieved by adding a filter condition on the " Hire Date " field of the Worker business object.
Specifically, you would configure the filter to compare the " Hire Date " against a dynamic date range, such as " Current Date minus 30 days " to " Current Date. " This ensures the

report dynamically adjusts to include only workers hired in the last 30 days each time it runs, which aligns with the needs of an integration sending real-time data to a third-party system.
Here's why the other options are incorrect:
* A. Subfilter: Subfilters in Workday are used to further refine data within a related business object or a subset of data already filtered by the primary filter. They are not the primary mechanism for applying a condition to the main dataset (e.g., all workers). For this scenario, a subfilter would be unnecessary since the condition applies directly to the Worker business object, not a related object.
* B. Output: The Output section of a custom report definition controls how the report is displayed or delivered (e.g., file format, scheduling), not the data selection criteria. It does not allow for specifying conditions like hire date ranges.
* C. Columns: The Columns tab defines which fields are displayed in the report output (e.g., Worker ID, Name, Hire Date). While you can add the " Hire Date " field here for visibility, it does not control which workers are included in the report-that is the role of the Filter tab.
To implement this in practice:
* In the custom report definition, go to the Filter tab.
* Add a new filter condition.
* Select the " Hire Date " field from the Worker business object.
* Set the operator to " in the range " and define the range as " Current Date - 30 days " to " Current Date " (using dynamic date functions available in Workday).
* Save and test the report to ensure it returns only workers hired within the last 30 days.
This filtered report can then be enabled as a web service (via the Advanced tab) or used in an Enterprise Interface Builder (EIB) or Workday Studio integration to send the data to the third-party system, meeting the integration requirement.
References from Workday Pro Integrations Study Guide:
* Workday Report Writer Fundamentals: Section on " Creating and Managing Filters " explains how filters are used to limit report data based on specific conditions, such as date ranges.
* Integration System Fundamentals: Discusses how custom reports serve as data sources for integrations and the importance of filters in defining the dataset.
* Core Connectors & Document Transformation: Highlights the use of filtered custom reports in outbound integrations to third-party systems.

NEW QUESTION: 45
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?
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 (LEAVE A REPLY)
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 References
* 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.

NEW QUESTION: 46
You are creating an outbound connector using the Core Connector: Job Postings template. The vendor has provided the following specification for worker subtype values:

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?
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 (LEAVE A REPLY)
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.

NEW QUESTION: 47
Refer to the following scenario to answer the question below.
You are implementing a Core Connector: Worker integration to send employee data to a third-party active employee directory. The external vendor requires the following:
* The Employee ' s Active Directory User Principal Name.
* A mapping from Worker Type values to external worker type codes.
* A specific filename format that includes a timestamp and sequence number.
You also need to ensure the document transformation occurs before the file is delivered to the endpoint. The connector ' s output must be transformed before the file is delivered to the vendor.
What step must be taken to ensure this occurs correctly?
A. Schedule the Document Transformation connector to run after the Core Connector: Worker completes.

B. Configure the business process to run the Document Delivery Service before the Document Transformation step.
C. Configure the business process to run the Document Transformation step before the Document Delivery Service step.
D. Schedule the Document Transformation connector in a separate integration system from the business process delivery step.
Answer: C (LEAVE A REPLY)
The requirement states that the connector output must be transformed before the file is delivered to the endpoint. This means the Document Transformation step must run first, followed by the Document Delivery step.
In Workday, this is managed through the Business Process (BP) attached to the integration system.
From Workday documentation:
"To transform an integration file before delivery, configure the Business Process to run the Document Transformation step before the Document Delivery Service step." This ensures that:
* The file is converted (via XSLT) to the correct format (e.g., CSV or flat XML)
* Only the final, transformed file is sent to the endpoint
Why the others are incorrect:
* A. Scheduling separately does not ensure correct sequence.
* B. Delivery before transformation would send the wrong file.
* D. A separate integration system is unnecessary and not best practice for chained transformations.
Reference:Workday Pro: Document Transformation - Chaining Transformation and Delivery Steps in Integration BPWorkday Integration Certification Guide - Document Transformation Use Cases

NEW QUESTION: 48
A vendor needs an EIB that uses a custom report to output a list of new hires and their child dependent(s).
You have been asked to create a calculated field that will be used to add only child dependent(s).
Which calculated field functions do you need to accomplish this?
A. Text Constant, True/False Condition, Evaluate Expression
B. True/False Condition, Evaluate Expression
C. Text Constant, True/False Condition, Extract Multi-Instance
D. True/False Condition, Extract Multi-Instance
Answer: D (LEAVE A REPLY)
In this case, you're asked to create a calculated field that:
* Filters dependent records
* Includes only child relationships

This means:
* The worker has multiple dependents (a multi-instance field).
* You need to extract only those dependent(s) where the relationship is "Child".
To achieve this in Workday, use:
* True/False Condition # check if the relationship descriptor = "Child"
* Extract Multi-Instance # filters the multi-instance field (Dependents) using the above condition to return only matching records This two-step logic filters multi-instance relationships correctly.
Why the other options are incorrect:
* A and B are missing Extract Multi-Instance, which is required to filter multi-values.
* C includes Text Constant unnecessarily - only True/False Condition and Extract Multi-Instance are required.
Reference:Workday Pro: Calculated Fields - Filtering Multi-Instance Fields (Dependents, Emergency Contacts)Workday Community: Best Practice for Extracting Specific Relationships in EIB Reports