PassQueen
a.k.a dumpsqueen.com

Workday Pro Integrations Certification Exam

Workday Workday-Pro-Integrations

Version Demo

Total Demo Questions: 12

Total Premium Questions: 120

Buy Premium PDF

https://passqueen.com

support@passqueen.com

QUESTION NO: 1

Which components make up the three primary parts of an outbound Enterprise Interface Builder (EIB)?

A. One data sourceOne transformationOne delivery endpoint

B. Multiple data sourcesOne transformationOne delivery endpoint

C. One web service callOne sequence generatorOne delivery endpoint

D. One data sourceOne transformationMultiple delivery endpoints

ANSWER: A

Explanation:

One data source
One transformation
One delivery endpoint correctly represents the standard outbound EIB design model. An outbound EIB extracts data from a single configured source, most commonly a Workday custom report, although supported alternatives may be available depending on the use case. The extracted data can then pass through a single transformation configuration to format the output for the receiving system; this is commonly used for XML-based formatting and XSLT processing. Finally, the EIB sends the resulting file or payload through one configured delivery mechanism, such as SFTP, email, or a supported integration transport. This structure aligns with the outbound EIB workflow of obtaining Workday data, optionally formatting it, and delivering the completed output to its destination. It provides a streamlined, configuration-driven approach for file-based integrations without requiring custom code for typical reporting and delivery needs. Workday positions EIB as part of its broader integration capabilities for securely connecting and exchanging data with external systems. See Workday Integration and the Workday Community for Workday documentation and tenant-specific EIB configuration guidance.

QUESTION NO: 2

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

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

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

A. Provide the ISSG View permissions to the domain security policies securing each overridden field.

B. Assign the ISSG to the domain security policies that govern the web service operations with Get access.

C. Grant View permissions to the ISSG for the domain security policies that secure the web service operations.

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

ANSWER: A

Explanation:

Provide the ISSG View permissions to the domain security policies securing each overridden field. In a Core Connector: Worker integration, the integration system security group must be authorized to view every underlying business object field used by an integration field override, including fields referenced by any calculated-field logic. The Test Security related action evaluates whether the effective security of the user or ISSG can access those fields. A value of “No” in Available by User therefore indicates that the required field-level domain security access is not available to the integration’s security context.

Review the domain security policies associated with the overridden fields and with every field the calculated field evaluates. Add the ISSG to the relevant domain policies with View permission, then activate the pending security-policy changes. View access is the appropriate permission because the integration needs to read the values in order to evaluate the override and include the resulting data in its output. Retesting after activation should show the fields as available, assuming all referenced fields are similarly secured. Workday's platform security model uses domain security policies to control access to business-process and data elements; see Workday Security and Workday Documentation.

QUESTION NO: 3

Refer to the following XML to answer the question below.

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 element. This root template then applies templates against . 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 element is placed within the template which matches on ?

A. wd:Job_Profile_Reference/wd:ID/wd:type='Job_Profile_ID'

B. wd:Job_Profile_Reference/wd:ID/@wd:type='Job_Profile_ID'

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

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

ANSWER: C

Explanation:

`wd:Job_Profile_Reference/wd:ID[@wd:type='Job_Profile_ID']` is the appropriate relative XPath expression when the current context node is a Job Profile. It first navigates from that context node to its Job Profile Reference child and then to the contained ID elements. The predicate `[@wd:type='Job_Profile_ID']` restricts that node selection to the ID whose namespaced `type` attribute has the required value. Because the selected node is the ID element rather than its attribute, `xsl:value-of` returns the element's string value, which is the Job Profile identifier required by the transformation. XPath predicates are evaluated against each candidate node in the node sequence, making this the standard syntax for selecting an element based on an attribute condition. The namespace prefix must be declared in the stylesheet and bound to the same Workday business-services namespace as the source document; XPath uses the prefix binding in the stylesheet to resolve element and attribute names. The XPath specification describes predicates and attribute axes at W3C XPath 3.1, and the XSLT specification defines how `xsl:value-of` obtains the string value of its selected item at W3C XSLT 3.0.

QUESTION NO: 4

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

When testing, you run the Test Security Related Action from the Configure Integration Field Override step. Several field overrides display â€œNoâ€□ in the Available by User column.

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

A. Provide the ISSG View permissions to the domain security policies securing each overridden field.

B. Assign the ISSG to the domain security policies that govern the web service operations with Get access.

C. Grant View permissions to the ISSG for the domain security policies that secure the web service operations.

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

ANSWER: A

Explanation:

Provide the ISSG View permissions to the domain security policies securing each overridden field. The Test Security related action evaluates whether the Integration System Security Group (ISSG), acting as the integration's execution identity, can read every field required by an integration field override. This includes fields directly returned by the override as well as fields referenced by the calculated-field logic used to derive a value, such as the phone allowance eligibility calculation in this scenario.

An "Available by User" result of "Yes" requires the ISSG to have effective *View* access through the relevant domain security policies. Review the security domains for the overridden business-object fields and for all dependent calculated-field inputs, add the ISSG to the applicable View policy permissions, and activate the resulting pending security policy changes. After activation, rerun Test Security to validate the ISSG's effective access.

Workday's integration security model relies on domain security policy access for integration-system users, and calculated fields remain subject to the security of their underlying referenced data. See Workday's security overview and integration system security guidance.

QUESTION NO: 5

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

Explanation:

Configure the business process to run the Document Transformation step before the Document Delivery Service step. This establishes an explicit, managed processing sequence within the integration's business process: Core Connector: Worker produces the output, the Document Transformation step processes that output, and Document Delivery Service sends the resulting transformed document to the vendor endpoint. The delivered attachment is therefore the final document rather than the connector's untransformed output.

Document Transformation is the appropriate place to apply an XSLT-based layout or format conversion when the downstream system requires a particular structure, such as a delimited file or vendor-specific XML. It can also support a delivery-ready result whose name and content meet the vendor's defined conventions. Configuring both transformation and delivery as ordered business-process steps provides traceability in the integration event and ensures that delivery waits for the transformation output. This is particularly important for worker-directory feeds, where the endpoint must receive the mapped and formatted file consistently on every run. Workday describes its integration capabilities, including secure connectivity and data transformation, at Workday Integration. Tenant-specific configuration guidance and delivered integration documentation are available to authorized customers through the Workday Community.

QUESTION NO: 6

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 configuration would generate a filename matching the day month year format for “File-14-1-2024.xml”?

A. File-[d]-[MM] -[yyyy].xml

B. File-[d]-[mmm] -[yyyy].xml

C. File-[d]-[M] -[yyyy].xml

D. File-[d]-[m] -[yyyy].xml

E. File-[d]-[M]-[yyyy].xml

ANSWER: E

Explanation:

File-[d]-[M]-[yyyy].xml is the configuration that exactly produces the required filename, **File-14-1-2024.xml**, for a date of January 14, 2024. The **[d]** token supplies the day of the month without requiring a leading zero, yielding **14**. The capitalized **[M]** token supplies the numeric month without a leading zero, yielding **1** for January. The **[yyyy]** token supplies the complete four-digit year, yielding **2024**. The literal text **File-**, the hyphens, and the **.xml** extension are retained exactly as written, so the assembled value has no unwanted characters or spaces. This is important for connector filenames because literal whitespace is part of the generated name and can affect downstream file pickup, naming conventions, and matching rules. Workday integration practitioners should validate filename patterns using representative dates, particularly months and days that reveal whether zero-padding is present. Workday Pro learning and integration resources are available through Workday Community, while the broader Workday Pro program is described at Workday Pro.

QUESTION NO: 7

Refer to the following XML and example transformed output to answer the question below.

```
<wd:Report_Data xmlns:wd="urn:com.workday.report/Int_Report">
    <wd:Report_Entry>
        <wd:Worker>Logan McNeil</wd:Worker>
        <wd:Education_Group>
            <wd:Education>California University</wd:Education>
            <wd:Degree>MBA</wd:Degree>
        </wd:Education_Group>
        <wd:Education_Group>
            <wd:Education>Georgetown University</wd:Education>
            <wd:Degree>B.S.</wd:Degree>
        </wd:Education_Group>
    </wd:Report_Entry>
    <wd:Report_Entry>
        <wd:Worker>Steve Morgan</wd:Worker>
        <wd:Education_Group>
            <wd:Education>Iowa State University</wd:Education>
            <wd:Degree>B.A.</wd:Degree>
        </wd:Education_Group>
        <wd:Education Group>
            <wd:Education>Northwestern University</wd:Education>
            <wd:Degree>MBA</wd:Degree>
        </wd:Education_Group>
    </wd:Report_Entry>
</wd:Report_Data>
```

Example transformed wd:Report_Entry output;

```
<Transformed_Record>
    <Worker>Logan McNeil</Worker>
    <Degrees>
        <Degree>California University MBA</Degree>
        <Degree>Georgetown University B.S.</Degree>
    </Degrees>
</Transformed_Record>
```

What is the XSLT syntax tor a template that matches on wd: Educationj3roup to produce the degree data in the above Transformed_Record example?

A.

B.

C.

D.

ANSWER: B

Explanation:

`<xsl:template match="wd:Education_Group"><Degree><xsl:value-of select="*"/></Degree></xsl:template>` is the correct template structure because it matches each `wd:Education_Group` node and explicitly creates the required target-side `Degree` element. Within that new result element, `xsl:value-of` writes the string value of the selected source node into the output. This is the appropriate operation when the transformed record requires degree data as text rather than a copied Workday XML element tree. The literal `Degree` element is part of the result document, while the XPath selection is evaluated against the education-group node currently being processed. In an XSLT transformation used by a Workday integration, this distinction allows a report's namespaced source XML to be reshaped into the element names and text-oriented structure required by the receiving system. The XSLT specification defines `xsl:value-of` as the instruction that creates a text node from the selected value; its behavior is documented by the W3C XSLT 1.0 specification. The matching behavior used to select `wd:Education_Group` is also defined in the W3C template-rule documentation.

QUESTION NO: 8

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: A

Explanation:

The template set using `wd:Report_Entry[wd:Country_Code = 'USA']`, `wd:Report_Entry[wd:Country_Code = 'GBR']`, and `wd:Report_Entry[not(wd:Country_Code = 'USA' or wd:Country_Code = 'GBR')]` is correct. In an XSLT transformation used with Workday RaaS output, each `wd:Report_Entry` is a source record. A predicate in square brackets filters that record node according to the value of its child `wd:Country_Code` element. The first template therefore matches only report entries for USA workers, while the second matches only entries for GBR workers. The final predicate uses `not()` around the two-country condition, producing a mutually exclusive fallback population for every entry whose country code is neither USA nor GBR.

This structure is appropriate for EIB transformation logic because the templates are selected at the report-entry level, preserving access to all fields belonging to the worker record within each template body. The expressions use standard XPath comparison and Boolean operators, so they can be evaluated directly as part of XSLT template matching. See the W3C references for XPath predicates and XSLT template rules.

QUESTION NO: 9

You have an existing EIB that you configured with a filename sequence generator that is associated with the EIB in the Deliver section. However, after testing the EIB, you notice that the filename is not using the configured sequence generator.

What is causing the sequence generator to not be used?

A. The Determine Value at Runtime for the File Name launch parameter to choose the field Next Sequence for Integration File Utility was not used.

B. The Dynamic Filename checkbox within the Enterprise Interface Builder was not enabled.

C. The Specify Value for the File Name launch parameter and typing in the name of the sequence generator object was not completed.

D. The File Name value in the Deliver section of the EIB's graphical guided interface was not deleted.

ANSWER: A

Explanation:

The correct configuration is to use *Determine Value at Runtime* for the *File Name* launch parameter and select *Next Sequence for Integration File Utility*. Associating a filename sequence generator with an EIB's delivery configuration establishes the sequence generator that is available to the integration, but it does not automatically make the output file name consume the next generated value. The EIB must obtain its filename at launch time from the Integration File Utility's next-sequence value. That runtime selection is what invokes the configured generator, advances the sequence as appropriate, and supplies the resulting value for the delivered file name. This design supports repeatable outbound processing, avoids manually maintained file names, and enables downstream systems to receive names with the expected incremental sequence. Workday integrations use configurable delivery and runtime processing patterns so that file-generation behavior can be controlled separately from the integration's data transformation logic. For broader context on Workday's integration platform and its supported integration capabilities, see Workday Integrations and Workday Platform and Product Extensions.

QUESTION NO: 10

What is the relationship between an ISU (Integration System User) and an ISSG (Integration System Security Group)?

A. The ISU is a member of the ISSG.

B. The ISU owns the ISSG.

C. The ISU grants security policies to the ISSG.

D. The ISU controls what accounts are in the ISSG.

ANSWER: A

Explanation:

The ISU is a member of the ISSG. An Integration System User is the non-interactive account under which an integration authenticates and runs in the Workday tenant. The Integration System Security Group is the security principal to which domain and business-process security permissions are assigned. Adding the Integration System User to that group gives the integration account the permissions configured for the group, such as access to required web-service operations and integration-related domains. This separation is important because it lets administrators manage permissions centrally at the group level while maintaining a distinct identity, authentication configuration, and audit trail for each integration account. It also supports least-privilege design: the group can be granted only the domain permissions needed by the integration, and an Integration System User can be associated with the appropriate group or groups based on its purpose. After security configuration changes are made, tenant administrators must activate the pending security policy changes before the revised access takes effect. Workday describes its platform security model and role-based access approach at Workday Security. Workday's integration platform overview, including secure connectivity for integrations, is available at Workday Integration.

QUESTION NO: 11

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

Explanation:

Lookup Related Value is the required calculated field because a Date Difference calculation evaluates date values in the context of one business object. When one required date resides on a related business object, a Lookup Related Value calculated field follows the applicable relationship from the primary object to that related object and returns the needed date value onto the primary object's calculated-field context. The resulting date can then be selected together with the local date when configuring the Date Difference calculation.

This approach preserves the business-object relationship and produces a reusable derived field that can be referenced by reports, integrations, or subsequent calculated fields. The lookup must be configured with the appropriate related business object and the date field to return; its data type should resolve to Date so that it is available to the Date Difference function. This is a standard calculated-field design pattern: first retrieve a single related value, then apply the arithmetic or date operation using values available from the same object context.

For Workday learning resources and tenant-specific calculated-field guidance, consult Workday Community and Workday's Education and Training resources.

QUESTION NO: 12

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">61383c9b1d094d44a73166ad39caebce</wd:ID>
                        <wd:ID wd:type="Degree_ID">MBA</wd:ID>
                    </wd:Degree_Reference>
                    <wd:Field_Of_Study_Reference>
                        <wd:ID wd:type="WID">62e42dfd4b8c49b5842114f67369a96f</wd:ID>
                        <wd:ID wd:type="Field_Of_Study_ID">Economics</wd:ID>
                    </wd:Field_Of_Study_Reference>
                    <wd:Required>0</wd:Required>
                </wd:Education_Qualification_Replacement_Data>
                <wd:Education_Qualification_Replacement_Data>
                    <wd:Degree_Reference>
                        <wd:ID wd:type="WID">8db9b8e5f53c4cbdb7f7a904c6afde28</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 using a web service enabled report to output worker data along with their dependents. You currently have a template which matches on wd:Report_Data/wd:Report_Entry for creating a record from each report entry.

Within the template which matches on wd:Report_Entry you would like to conditionally process the wd:Dependents_Group elements by using an element.

What XPath syntax would be used as the select for the apply templates so as to iterate over only the wd:Dependents_Group elements where the dependent relationship is Child?

A. wd:Dependents_Group[@wd:Relationship='Child']

B. wd:Dependents_Group[wd:Relationship='Child']

C. wd:Dependents_Group/wd:Relationship='Child'

D. wd:Dependents_Group/@wd:Relationship='Child'

ANSWER: B

Explanation:

`wd:Dependents_Group[wd:Relationship='Child']` is the appropriate XPath selection because it starts from the current `wd:Report_Entry` context and selects each child `wd:Dependents_Group` element. The bracketed portion is an XPath predicate: it evaluates the `wd:Relationship` child element for each candidate dependent group and retains that group only when its string value is `Child`. Therefore, an `xsl:apply-templates` instruction using this select expression invokes the applicable dependent-group template once for every dependent whose relationship is Child. This is especially useful in report-based EIB transformations, where a worker report entry can contain a repeating group of dependents and the XSLT must generate output selectively from that group. XPath predicates are evaluated relative to each node selected

by the preceding location step, which is why the relationship test is written inside square brackets after `wd:Dependents_Group`. The namespace prefix must be declared in the stylesheet and bound to the same Workday namespace used by the source report XML. The XPath predicate syntax is defined in the W3C XPath specification, and the behavior of `xsl:apply-templates` with a select expression is described in the W3C XSLT specification.