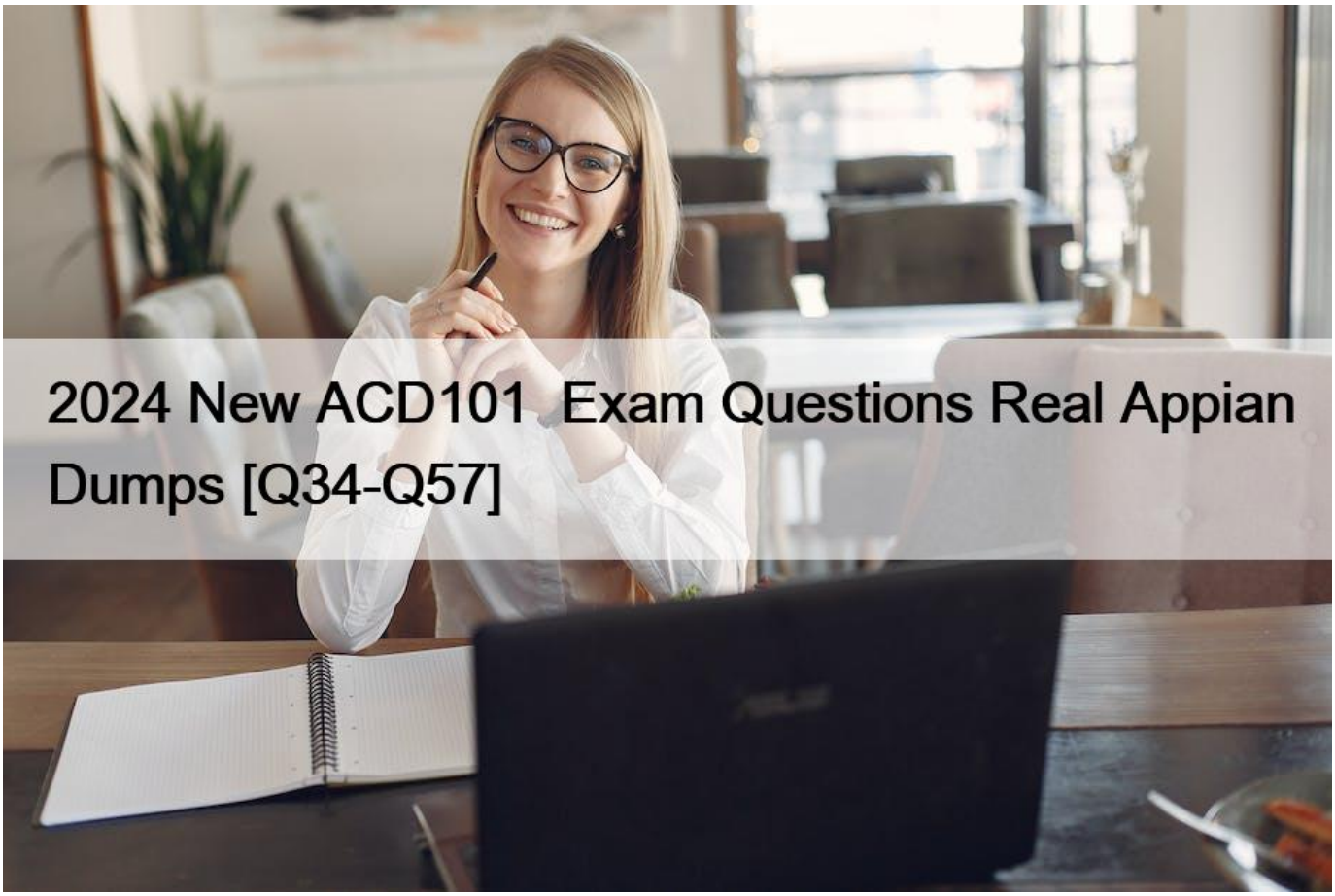
2024 New ACD101 Exam Questions Real Appian Dumps [Q34-Q57



2024 New ACD101 Exam Questions Real Appian Dumps
Course 2024 ACD101 Test Prep Training Practice Exam Download

QUESTION 34

Which code snippet calls the interface APP_RecordDashboard while following best practices for passing in values for “recordId” and “firstName”?

*
```
rule!APP_RecordDashboard(
    1,
    "Kyle"
)
```

*
```
rule!APP_RecordDashboard(
    recordId: 1,
    firstName: "Kyle"
)
```

*
```
rule!APP_RecordDashboard(
    recordId >> 1,
    firstName >> "Kyle"
)
```

The best practice in Appian for passing values into an interface is to use named parameters, which is demonstrated by Option B. Named parameters make the code more readable and maintainable by clearly specifying which parameter each value is being passed to. In this case, the recordId and firstName parameters are clearly being assigned the values 1 and “Kyle” respectively.

Reference:

Appian Documentation: Passing Parameters to Interfaces

QUESTION 35

You are developing an expression rule. You need to find information on employing an Appian function that you have not used before.

For more information on the Appian function, what should you do first?
* Look up the function in the Appian Documentation.
* Search the Appian Knowledge Base Articles.
* Post a question on Appian Community.

When you need information on using a specific Appian function that you have not used before, the first step should be to consult the Appian Documentation. The documentation provides comprehensive details on each function, including syntax, parameters, usage examples, and best practices, which is essential for understanding how to correctly employ the function in an expression.

Reference: Appian Documentation – Functions

QUESTION 36

Review the following code snippet:

index({“a”, “b”, “c”}, 1, “x”)

Which value is returned?
* a
* b
* c
* x

The index() function in Appian is used to replace an element at a specified index in a list. In the code snippet index({“a”, “b”, “c”}, 1, “x”), the function is called to replace the element at index 1 of the list {“a”, “b”, “c”} with the string “x”. Since Appian lists are 1-indexed, this means that the first element “a” will be replaced by “x”. However, since the function is used to modify a list and not to return a value, if this function is used as is without being part of a larger expression that outputs the list, it would not return any of the listed values.

Reference: Appian Expression Language Documentation – Functions

QUESTION 37

Where can an Appian Developer connect with and share their expertise with other Appian Developers?
* Appian Learning Paths via Appian Academy
* Appian Knowledge Base

* Appian Community discussions

Appian Community discussions provide a platform for Appian Developers to connect with, share expertise, and learn from each other. The community is a vibrant space where developers can ask questions, share solutions, and discuss best practices related to Appian development. While Appian Learning Paths via Appian Academy and Appian Knowledge Base are valuable resources for learning and troubleshooting, the Community discussions specifically facilitate peer-to-peer interaction and knowledge sharing among developers.

Reference: Appian Community Website

QUESTION 38

You are working on a process model “VIM Update Vehicle.”

You want to call another process “VIM Get Service Date” that accepts pv!vehicleId as a process parameter and sets a value for pv!serviceDate. The next node in VIM Update Vehicle depends on the value of pv!serviceDate.

Which node should you use to execute “VIM Get Service Date” from VIM Update Vehicle?

* Start Process smart service
* Asynchronous subprocess with activity chaining
* Synchronous subprocess with input and output variables configured

When a process model depends on the value of a variable being set by another process, a synchronous subprocess should be used. This ensures that the calling process (VIM Update Vehicle) will wait for the subprocess (VIM Get Service Date) to complete and set the necessary pv!serviceDate value before continuing. The subprocess node should be configured with input and output variables to pass the pv!vehicleId and receive the pv!serviceDate.

Reference:

Appian Documentation: Subprocess Node

QUESTION 39

Which Appian feature can help the implementation team analyze the event log data of an existing process?

* RPA
* Process Mining
* Workflow

vent logs. It uses the data from process models to give insights into process performance, bottlenecks, and compliance with the designed process flow. This feature is valuable for understanding the actual performance of business processes and for identifying areas for improvement.

Reference: Appian Documentation – Process Mining

QUESTION 40

You need to start a process using an email trigger.

How should you configure this process model using the Process Model Properties dialog?

* Go to File > Properties > Alerts. Configure the Custom Alert settings.
* Go to File > Properties Set the proper Process Display Name
* Go to File > Properties. Select the Public Events checkbox to allow anyone to fire triggers.

To start a process using an email trigger, you need to configure the process model to listen for an email event. This is done by going

to File > Properties > Alerts in the process model properties dialog and configuring the Custom Alert settings. Here you can specify the email address that will trigger the process when an email is sent to it.

Reference: Appian Documentation – Configuring Alerts in Process Models

QUESTION 41

Which set of out-of-the-box features is only available when data sync is enabled on a record type?
* Generate record actions

Define record type object security

Add custom record fields
* Define record type relationships

Add custom record fields

Configure record-level security
* Define record type relationships

Add hidden record fields

Configure record-level security
Data sync enables additional features for record types in Appian. With data sync enabled, you can define relationships between different record types, add fields to a record type that do not appear in the source database (hidden fields), and configure record-level security to control access to individual records based on user roles or other criteria. These features are part of the enhanced functionality provided by data sync to ensure efficient data management and security within Appian applications.

Reference: Appian Documentation – Record Type Features and Data Sync

QUESTION 42

Which two scenarios are ideal for using Appian Portals? (Choose two.)
* A manager wants to obtain a view of their team’s performance.
* A retail customer wants to conduct a public survey for their recently launched product.
* An employee who does not have an account wants to register for their company’s vehicle fleet a management system.
* A user needs to submit support requests when they are using their mobile device in areas with bad network coverage.

Appian Portals are designed for scenarios where users who do not have an Appian account need to interact with Appian applications. This makes them ideal for public-facing applications such as surveys (Option B) or for allowing external users to initiate processes like registrations (Option C). They are not intended for internal use by employees (Option A) or for scenarios requiring offline capabilities (Option D).

Reference:

Appian Documentation: Appian Portals

QUESTION 43

What is an Appian best practice for calling interface rules on your interface?
* Call the interface rule on a rule input.

* Use keyword syntax.
* Always use consistent ordering of rule parameters.

An Appian best practice for calling interface rules within your interfaces is to always use a consistent ordering of rule parameters. This practice enhances the readability and maintainability of your interfaces, ensuring that other developers can understand and modify the interface more easily.

Reference:

Appian Documentation: Designing Interfaces

QUESTION 44

You have two Custom Data Types (CDT): ACME_invoice and ACME_invoiceItem that have a flat relationship.

The invoice item table has a field that is a foreign key to the invoice table. You are leveraging the database to automatically generate their primary keys.

How should you structure the process model to add a new invoice and the new invoice items to the system?

* 1. Write to Multiple Data Store Entities smart service (Writing to the ACME_invoiceItem table and ACME_invoice table).
* 1. Write to Data Store Entity smart service (Writing to the ACME_invoiceItem table).

2. Script Task to update foreign keys.

3. Write to Data Store Entity smart service (Writing to the ACME_invoice table).

* 1. Write to Data Store Entity smart service (Writing to the ACME_invoice table).

2. Script Task to update foreign keys.

3. Write to Data Store Entity smart service (Writing to the ACME_invoiceItem table).

When dealing with related data types where one has a foreign key to another, you must first create the record in the primary table (ACME_invoice) and then use the generated primary key to create related records in the secondary table (ACME_invoiceItem). This is why you first write to the ACME_invoice table, then update the foreign keys in a Script Task, and finally write to the ACME_invoiceItem table.

Reference:

Appian Documentation: Relational Databases

QUESTION 45

Which two groups can be set within Application Properties? (Choose two.)

* Developers Groups
* Designers Groups
* Administrators Groups
* Users Groups

Within Application Properties in Appian, you can set two groups: Administrators Groups and Users Groups. The Administrators Group is responsible for managing and configuring the application, while the Users Group is designated for end-users who interact with the application’s functionalities.

Reference:

Appian Documentation: Application Properties

QUESTION 46

You need to be able to define record type relationships.

What is a required prerequisite in Appian?
* The record types must have data sync enabled.
* The record types must be on a virtualized data source.
* The record types must be stored in the local Appian business database.

To define record type relationships in Appian, it’s required that the record types have data sync enabled. This is necessary to ensure that the data is readily available in Appian for the relationships to be established and maintained.

Reference:

Appian Documentation: Record Type Relationships

QUESTION 47

You want to add a script task in between two User Input Tasks assigned to the same user.

What needs to be configured so that the user sees the second form immediately after submitting the first?
* Set all process variables as parameters.
* Enable activity chaining.
* Run the script task as “synchronous.”

To ensure that the user sees the second form immediately after submitting the first, when adding a script task between two User Input Tasks assigned to the same user, it is essential to enable activity chaining. Activity chaining in Appian allows for the seamless transition between user tasks within a process, eliminating unnecessary delays and enhancing the user experience by immediately presenting the subsequent task.

Reference:

Appian Documentation: Activity Chaining in Processes

QUESTION 48

You are configuring an employee onboarding User Input Task that will be assigned to the human resources group.

Based on the default behavior for task assignments, which statement is valid?
* Multiple users can accept the task at the same time up until the point that the first user completes it.
* For each user in the group, a task is generated and assigned to them to complete.
* One user in the group can accept the task for themselves and complete it.

Based on the default behavior for task assignments in Appian, when a User Input Task is assigned to a group, any one member of the group can accept the task. Once accepted, the task becomes locked to that user, and they are responsible for completing it. This prevents multiple users from working on the same task simultaneously and ensures clear accountability.

Reference: Appian Documentation – Task Assignments and User Input Tasks

QUESTION 49

ACME Automobile uses Appian to manage their vehicle fleet. Vehicle records can have a status of either “active” or “inactive”.

Users are primarily concerned with active vehicles and want to see only those records by default when viewing the Vehicle records list. However, it is important for users to be able to see the unfiltered list of Vehicle records on demand to address occasional auditing requests from managers.

Which configuration supports the desired Vehicle record list behavior?
* Visibility on the Status column in the Vehicle record list set with conditional logic.
* A source filter set to exclude vehicles with status “inactive”.
* A user filter for the status field with a default option corresponding to “active”.

To achieve the behavior where users see only “active” vehicle records by default but can also view all records when needed, you should configure a user filter for the status field on the Vehicle record list. This user filter should have a default value set to “active”, which will filter the list to only show active records initially. However, users will still have the option to adjust the filter to see all records, thus accommodating occasional auditing requests.

Reference: Appian Documentation – Record List Filters and User Filters

QUESTION 50

You are using a local variable in an expression rule to describe the height of an applicant.

Which statement correctly describes the application of Appian best practices for naming your local variable?
* local!hoaa – This employs the naming convention of abbreviating “Height of an applicant” to minimize both the typing required by developers and the length of code Appian is required to parse.
* local!applicantHeight – This employs the naming convention of specifically describing the value contained by the variable.
* local!x – This employs the naming convention of using algebraic variables for a value that may either change over time or be used by future developers for other purposes.

The best practice for naming variables in Appian is to use clear and descriptive names that convey the purpose or content of the variable. Therefore, local!applicantHeight is the best option as it precisely describes the value contained by the variable, which is the height of an applicant. This naming convention aids in readability and maintainability of the code, making it easier for developers to understand and modify the code in the future.

Reference: Appian Best Practices – Expression Writing and Naming Conventions

QUESTION 51

You need to pass data into a process from other parts of your Appian application.

Which configuration is required in your process model?
* Toggle the Parameter field to ‘True” on the configuration of a process variable.
* Create process variables on the Data Management tab of Process Model Properties.
* Add an interface as a Process Start Form.

To pass data into a process from other parts of an Appian application, you need to configure process variables. This is done on the Data Management tab within the Process Model Properties. Here, you can define process variables that can receive data from external sources, such as interfaces, other processes, or direct user input, when the process is started. These variables serve as placeholders for the data that will be used throughout the execution of the process.

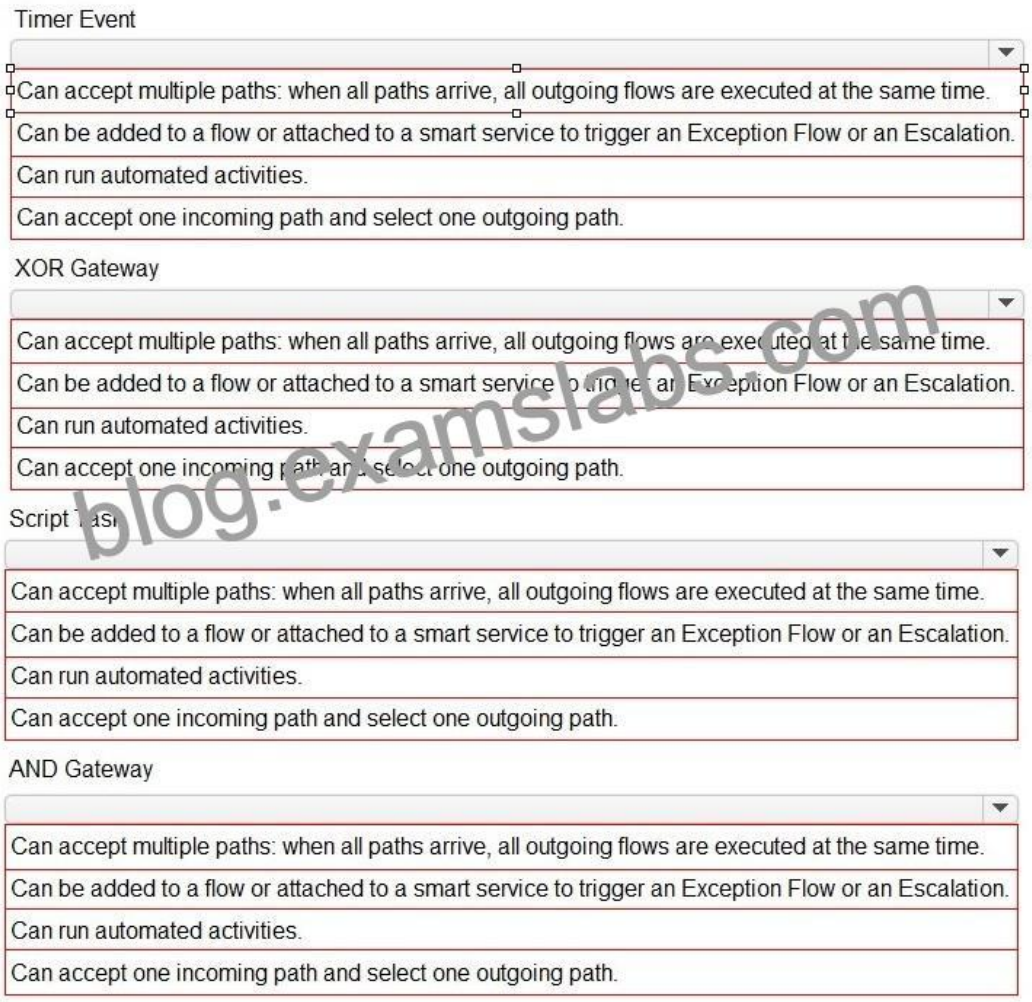
Reference: Appian Documentation – Process Model Properties

QUESTION 52

Match each node to the correct description for the node.

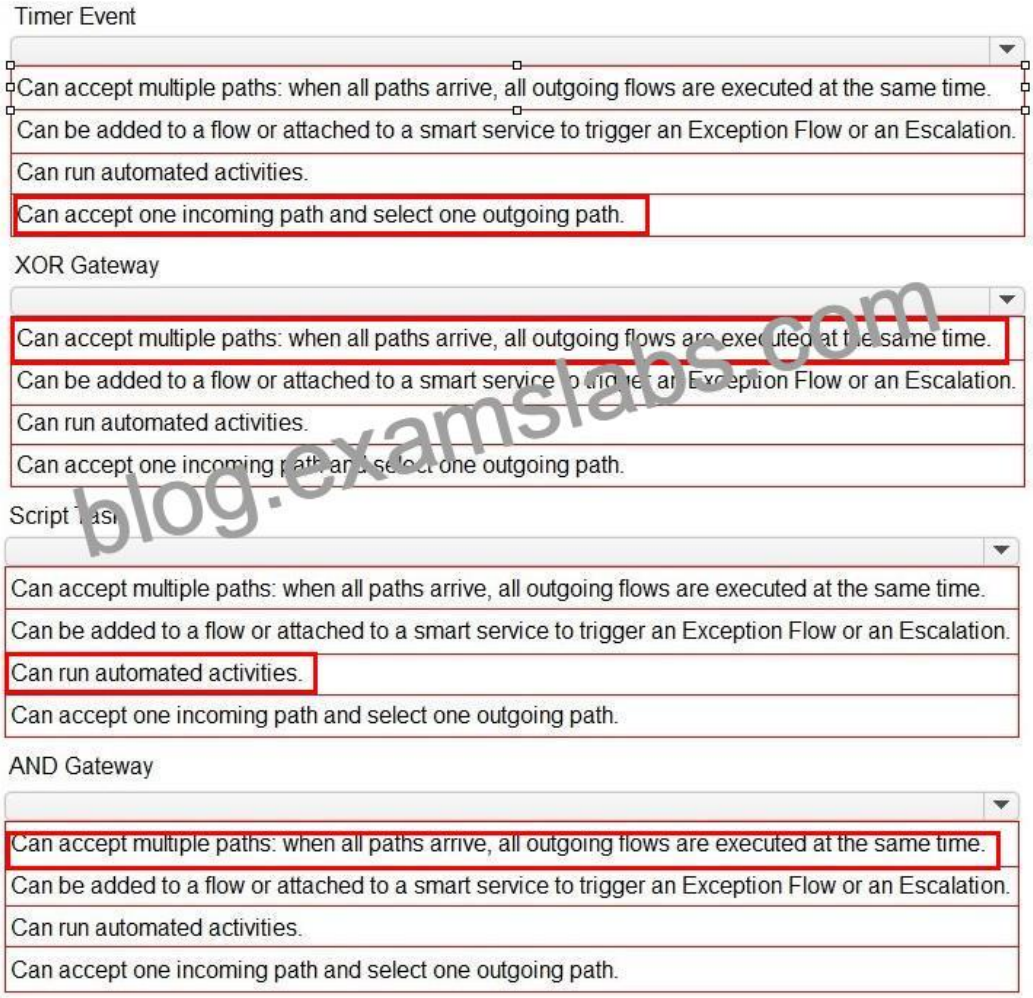
Note: Each description will be used once. To change your responses, you may deselect your response by clicking the blank space at the top of the selection list.

Answer Area



blog.examslabs.com

Answer Area



blog.examslabs.com

Reference:

Appian Documentation: Process Modeler Objects

QUESTION 53

You are creating a form used to order a pizz

a. You use a radio button component for the selection.

The pizza selection labels include a list of toppings. You do not want the selection labels to be truncated.

Which layout should you choose?
* Compact
* Grid
* Stacked

For a pizza ordering form where you do not want the radio button selection labels to be truncated, the Stacked layout is the most appropriate. This layout will list the options vertically, giving each one adequate space and preventing truncation, which is

particularly useful when the labels include longer text, such as a list of toppings.

Reference: Appian Documentation – Interface Components

QUESTION 54

Which step can be critical in passing information from a form back to a process model?
* Configure the Data Management tab.
* Configure the activity class parameters of a Write to Data Store Entity node, a
* Configure inputs on the Data tab of a User Input Task.

The critical step in passing information from a form back to a process model is to configure inputs on the Data tab of a User Input Task. When you create a User Input Task, it includes a form for users to interact with. The data entered into this form can be mapped to process variables via the Data tab configuration. This ensures that the information collected in the form is available to the process for further use.

Reference: Appian Documentation – User Input Tasks

QUESTION 55

You receive a bug ticket that states “After selecting a value for the drop-down field, the value disappears.” You investigate and notice that when you select the drop-down, the proper choice labels display. When you select an option, the value updates properly in the corresponding rule input.

What is the issue and how can you fix this bug?
* The value parameter is improperly configured on the drop-down component. You need to map the value to the proper rule input or variable.
* The user group for the lookup table is incorrect. You need to add the user to the proper group.
* The choice labels parameter of the drop-down field is not configured as a list. You need to wrap the value with curly brackets.

The described bug typically occurs when the value parameter of the drop-down component is not correctly mapped to the corresponding rule input or variable that is supposed to hold the selected value. To fix the issue, you should ensure that the drop-down’s value parameter is correctly mapped so that the selected option is retained and displayed properly.

Reference: Appian Documentation – Dropdown Field Component

QUESTION 56

Which statement about local variables is valid?
* The data type of a local variable is determined by its value.
* Local variables can only store primitive data types, such as numbers and strings.
* Local variables must have an initial value set when defining them.

In Appian, the data type of a local variable is inferred from the value it is set to. Unlike some other programming languages where the data type must be explicitly declared, Appian determines the data type automatically based on the initial value assigned to the local variable. Local variables in Appian are quite flexible and can store various types of data, including complex data types, not just primitive ones.

Reference: Appian Documentation – Local Variables

QUESTION 57

An interface references an expression rule.

What are the relationships between these objects?

* Dependents and Reliants
* Dependents and Precedents
* Inheritance and Association

In Appian, when an interface references an expression rule, the interface is considered a dependent because it depends on the expression rule to function correctly. Conversely, the expression rule is a precedent because the interface relies on it.

Reference:

Appian Documentation: Managing Application Contents

ACD101 Exam Info and Free Practice Test Professional Quiz Study Materials:
https://www.examslabs.com/Appian/Associate-Developer/best-ACD101-exam-dumps.html]