ACD200 Q&As

Appian Certified Senior Developer

Pass Appian ACD200 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

https://www.pass4itsure.com/acd200.html

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Appian Official Exam Center

QUESTION 1

You are creating an ERD that models the data for a college and includes a Many-to-Many relationship, Student-to-Class, where a student can be enrolled in multiple classes, and a class can enroll multiple students.

How can you handle this relationship so that it can be supported in Appian and remain in at least First Normal Form? (Choose the best answer.)

A. A joining table can be used to hold instances of Student/Class relationships.

B. The Student table should have a Class field to hold an array of Class IDs.

C. The Class table should have a Student field to hold an array of Student IDs.

D. It cannot be done, because Appian CDTs cannot handle Many-to Many relationships.

Correct Answer: A

A joining table can be used to hold instances of Student/Class relationships, so that it can be supported in Appian and remain in at least First Normal Form. A joining table is a table that contains only foreign keys that reference the primary keys of other tables. A joining table can be used to model a Many-to-Many relationship between two entities, such as Student and Class. For example, a joining table called Enrollment could have two columns: Student_ID and Class_ID, which reference the primary keys of the Student and Class tables respectively. Each row in the Enrollment table represents an instance of a student enrolled in a class. References: [Joining Tables], [Many-to-Many Relationships], [First Normal Form]

QUESTION 2

You are tasked with configuring a process model to store the result of an expression rule for every item in a list.

Which process model design has the lowest memory footprint?

A. Run MNI over a script task for each item in the list.

B. Call a sub-process for each item in the list.

C. Configure a script task to use a!foreach to iterate over each item in the list.

D. Create a loop of smart service nodes in the process model and execute it for each item in the list.

Correct Answer: C

The question is about designing a process model to store the result of an expression rule for every item in a list with the lowest memory footprint. The best design for this purpose is to configure a script task to use a!foreach to iterate over each item in the list, as it allows you to execute an expression rule for each element of an array without creating multiple nodes or subprocesses in the process model. A script task is a node that executes an expression without user interaction, and a!foreach is a function that applies an expression to each element of an array and returns an array of results. The other options are not optimal for this purpose, as they either create more nodes or subprocesses in the process model, which consume more memory and resources. References: Script Task a!foreach()

QUESTION 3

When looking at the process model metrics for your application, you see that one of your process models has a low completion rate of 10%.

What are two potential causes of this? (Choose two.)

A. The process instances are long-lived compared to the configured days until archival or deletion.

B. A large number of instances are encountering process errors, and they are not being addressed by the production support team.

C. A large value is configured for days until archival or deletion compared to other process models in your application.

D. A large number of smart service nodes are configured in the process model.

Correct Answer: AB

The question is about the potential causes of a low completion rate of 10% for a process model. The following are two possible causes of this:

The process instances are long-lived compared to the configured days until archival or deletion. This means that the process instances take a long time to complete, and they are archived or deleted before they reach the end event. This

reduces the completion rate, as only the instances that reach the end event are counted as completed.

A large number of instances are encountering process errors, and they are not being addressed by the production support team. This means that the process instances are stuck in an error state, and they cannot proceed to the next step or

the end event. This reduces the completion rate, as only the instances that reach the end event are counted as completed.

The following are not likely causes of a low completion rate:

A large value is configured for days until archival or deletion compared to other process models in your application. This means that the process instances have more time to complete before they are archived or deleted. This should increase

the completion rate, as more instances can reach the end event before they are removed from the system.

A large number of smart service nodes are configured in the process model. This means that the process model has a complex logic or functionality that requires multiple smart services. This does not directly affect the completion rate, as

long as the smart services execute successfully and do not cause errors or delays.

References:

Process Model Metrics

Archiving and Deleting Process Instances

QUESTION 4

Your organization is considering the adoption of Behavior-Driven Development (BDD) and automated testing as part of application development.

Which three testing tools have packages preconfigured to work with Appian applications and are available via the AppMarket? (Choose three.)

A. Cucumber

B. Jenkins

C. Selenium API

D. Git

E. FitNesse

Correct Answer: ACE

The question is about the testing tools that have packages preconfigured to work with Appian applications and are available via the AppMarket. The following are three testing tools that meet these criteria:

Cucumber: This is a tool that supports Behavior-Driven Development (BDD) and automated testing by allowing users to write test scenarios in plain language and execute them using step definitions. Cucumber has a package called Appian

Cucumber Test Framework that provides a set of predefined step definitions and helper methods for testing Appian applications.

Selenium API: This is a tool that allows users to automate web browser interactions and test web applications. Selenium API has a package called Appian Selenium Framework that provides a Java library and a set of helper methods for

testing Appian applications.

FitNesse: This is a tool that allows users to create and run acceptance tests using wiki pages and tables. FitNesse has a package called Appian FitNesse Testing Framework that provides a set of fixtures and helper methods for testing

Appian applications.

The following are not testing tools that have packages preconfigured to work with Appian applications and are available via the AppMarket:

Jenkins: This is a tool that allows users to automate software development tasks such as building, testing, and deploying applications. Jenkins does not have a package specifically for Appian applications, but it can be integrated with other

testing tools such as Cucumber or Selenium API.

Git: This is a tool that allows users to manage version control and collaboration for software projects. Git does not have a package specifically for Appian applications, but it can be used to store and share test scripts or scenarios created by other testing tools. References: AppMarket Cucumber Selenium API FitNesse

QUESTION 5

You have a requirement that can only be achieved by using a plug-in, where more than one plug-in does a similar job.

What are three primary considerations in choosing the right plugin? (Choose three.)

A. What are the compatible Appian versions?

B. Is the plug-in Cloud-approved?

C. What are the limitations of the plug-in?

D. What is the size of the plug-in?

E. Who is the plug-in author?

Correct Answer: ABC

The question is about choosing the right plug-in for a requirement that can only be achieved by using a plug-in. The following are three primary considerations in choosing the right plug-in: What are the compatible Appian versions? This is important because some plug- ins may not work with older or newer versions of Appian due to changes in APIs or features. You should check the plug-in documentation or release notes to see which Appian versions are supported by the plug-in. Is the plug-in Cloud-approved? This is important because some plug-ins may not be allowed in Appian Cloud environments due to security or performance reasons. You should check the Appian Cloud Approved Plug-ins list to see which plug-ins are approved for use in Appian Cloud. What are the limitations of the plug-in? This is important because some plug-ins may have known issues or limitations that could affect your requirement or use case. You should check the plug-in documentation or forums to see if there are any reported bugs or limitations for the plug-in. The following are not primary considerations in choosing the right plug-in: What is the size of the plug-in? This is not a major factor, as plug-ins are usually small in size and do not affect the performance or storage of Appian significantly. However, you should still follow the best practices for managing plug-ins and avoid installing unnecessary or unused plug-ins. Who is the plug-in author? This is not a decisive factor, as plug-ins can be authored by Appian, third-party vendors, or community members. However, you should still consider the reputation and reliability of the plug-in author and check their support and update policies. References: Plug-ins Appian Cloud Approved Plug-ins

QUESTION 6

You are troubleshooting slow response times on a SAIL interface.

What are two potential causes of the performance issues? (Choose two.)

A. Multiple rule inputs

B. Multiple a!save operations

C. Configuration of refresh variables

D. Stacked button layout

Correct Answer: BC

The question is about troubleshooting slow response times on a SAIL interface. The following are two potential causes of the performance issues:

Multiple a!save operations: This means that the interface has multiple expressions that use a!save to store data in variables or update components. This can cause performance issues, as each a!save operation triggers a server round-trip

and refreshes all dependent components on the interface. You should minimize the use of a!save operations and use local variables or component parameters instead.

Configuration of refresh variables: This means that the interface has components that use refresh variables to control when they should be refreshed based on changes in other components or variables. This can cause performance issues,

as some components may be refreshed unnecessarily or too frequently due to incorrect or excessive refresh variables. You should optimize the refresh variables and use them only when needed.

The following are not likely causes of slow response times on a SAIL interface:

Multiple rule inputs: This means that the interface has expressions that use rule inputs to pass data or parameters to expression rules. This does not affect the performance of the interface, as rule inputs are evaluated on the server and do not

cause additional server round-trips or refreshes. Stacked button layout: This means that the interface has a layout component that arranges buttons in a vertical stack. This does not affect the performance of the interface, as it is a simple

layout component that does not require any server interaction or refresh logic.

References:

a!save()

Refresh Variables

Rule Inputs

Stacked Button Layout

QUESTION 7

You need to show joined data from 5 tables. Each table contains a large number of rows and could generate a large result set after executing the Joins.

The business is not expecting live data, and a 2-hour refresh is acceptable. Performance is a top priority.

What should you use? (Choose the best answer.)

A. Table

B. View

C. Stored procedure

D. Materialized view

Correct Answer: D

A materialized view is the best option to show joined data from 5 tables that contain a large number of rows and could generate a large result set after executing the joins. A materialized view is a physical table that holds the results of the SQL that a view would normally be constructed from and can be generated periodically. A materialized view can improve performance by reducing the execution time of complex queries that involve multiple joins, aggregations, or calculations. A materialized view can also reduce the load on the database server by storing the query results in advance. A materialized view can be refreshed at regular intervals or on demand to reflect the changes in the underlying tables. References: [Materialized Views], [View Performance]

QUESTION 8

You are reviewing a recent Health Check report and notice that a process model has high memory consumption.

What are three possible reasons for this? (Choose three.)

A. Too many process variables

B. Misconfigured error alerts

C. Nested CDTs with large numbers of fields

D. Too many nodes

E. Gateway nodes with multiple incoming flows

Correct Answer: ACD

Three possible reasons for high memory consumption of a process model are: Too many process variables. Process variables are used to store data that is needed throughout the process execution. However, having too many process variables can increase the memory usage of the process engine, especially if the variables store large or complex data types, such as documents or CDTs. It is recommended to use local variables whenever possible, and to delete or nullify process variables that are no longer needed. Nested CDTs with large numbers of fields. CDTs are custom data types that define the structure and validation of business data in Appian. CDTs can be nested within other CDTs to create complex data models. However, nesting CDTs with large numbers of fields can result in high memory consumption and performance degradation, as each field requires additional memory allocation and processing. It is recommended to limit the number of fields and nesting levels of CDTs, and to use references instead of embedding whenever possible. Too many nodes. Nodes are the graphical elements that represent the activities and events in a process model. Having too many nodes in a process model can increase the memory usage and complexity of the process execution. It can also make the process model harder to read and maintain. It is recommended to simplify the process model by using subprocesses, smart services, or expression rules to encapsulate common or reusable logic. References: Process Variables, Custom Data Types, Process Model Best Practices

QUESTION 9

You are required to display information for pending tasks for each individual in an application. There will be tasks for a single specific application.

Which context type should you choose for the Task Report Type?

A. Tasks by process model

B. Tasks attributed to user

C. Tasks by process

D. Tasks assigned to a group

Correct Answer: B

The context type that should be chosen for the Task Report Type is Tasks attributed to user. This is because this context type allows you to display information for pending tasks for each individual in an application. You can filter the tasks by application name, process model name, or task name. This context type also supports drilldown functionality, which enables users to view more details about a specific task or complete it directly from the report. References: Task Report Types

QUESTION 10

You are creating the group structure of a new application.

Which three best practices apply? (Choose three.)

A. Avoid creating custom group types unless there is a strong need/requirement.

B. Only create the groups necessary for task assignment or security.

C. Flat group structures should be avoided.

D. Group names should not include the application prefix.

E. Keep group names unique.

Correct Answer: ABE

QUESTION 11

You are creating an expression rule that will be reused throughout your environment.

What are two benefits of including meaningful test cases when creating a new expression rule? (Choose two.)

A. Speed up unit, regression, and exploratory testing.

B. Improve the appearance of the code.

C. Improve performance.

D. Increase code quality.

Correct Answer: AD

QUESTION 12

More than one user is editing the same record in database.

With XSD, how do you avoid the collision of data from Application? (Choose the best answer.)

A. @Version

B. @OrderBy

C. @inheritance

D. @AssociationOverrides

Correct Answer: A

The @Version annotation is used to avoid the collision of data from an application when more than one user is editing the same record in a database. The @Version annotation enables optimistic locking, which is a mechanism to detect and prevent concurrent updates on the same record. Optimistic locking assumes that multiple transactions can

complete without affecting each other, and that therefore transactions do not need to lock data resources until they are ready to commit. However, if a transaction tries to commit changes to a record that has been modified by another transaction since it was last read, an OptimisticLockException is thrown and the transaction is rolled back. The @Version annotation can be applied to a field or property of a CDT that stores a numeric value. This value is used as a version number that is automatically incremented by Appian every time the record is updated. References: Does Appian support @Version JPA Annotation?, JPA - Optimistic Locking and Use of @Version annotation

QUESTION 13

What is the lowest permission required on the data store in order to query, write, and delete data?

A. Manager

B. Administrator

C. Viewer

D. Editor

Correct Answer: C

The lowest permission required on the data store in order to query, write, and delete data is Viewer. This is because the data store permissions only control the ability to view and modify the data store object in Appian Designer, not the actual data in the database. To query, write, and delete data, the user needs to have the appropriate SQL permissions on the database tables or views that are mapped to the data store entities. References: Data Store Permissions

QUESTION 14

HOTSPOT

Match each of the business data concepts to an Appian data type. Each data type may be used once, more than once, or not at all.

Hot Area:

Transaction Timestamp

- Date and Time
- Number (Integer)
- Number (Decimal)
- Boolean
- Document

Amount (Value)

- Date and Time
- Number (Integer)
- Number (Decimal)
- Boolean
- Document

List Index

- Date and Time
- Number (Integer)
- Number (Decimal)
- Boolean
- Document

Has Paid?

- Date and Time
- Number (Integer)
- Number (Decimal)
- Boolean
- Document

Correct Answer:

Transaction Timestamp

- Date and Time
- Number (Integer)
- Number (Decimal)
- Boolean
- Document

Amount (Value)

- Date and Time
- Number (Integer)
- Number (Decimal)
- Boolean
- Document

List Index

- Date and Time
- Number (Integer)
- Number (Decimal)
- Boolean
- Document

Has Paid?

- Date and Time
- Number (Integer)
- Number (Decimal)
- Boolean
- Document

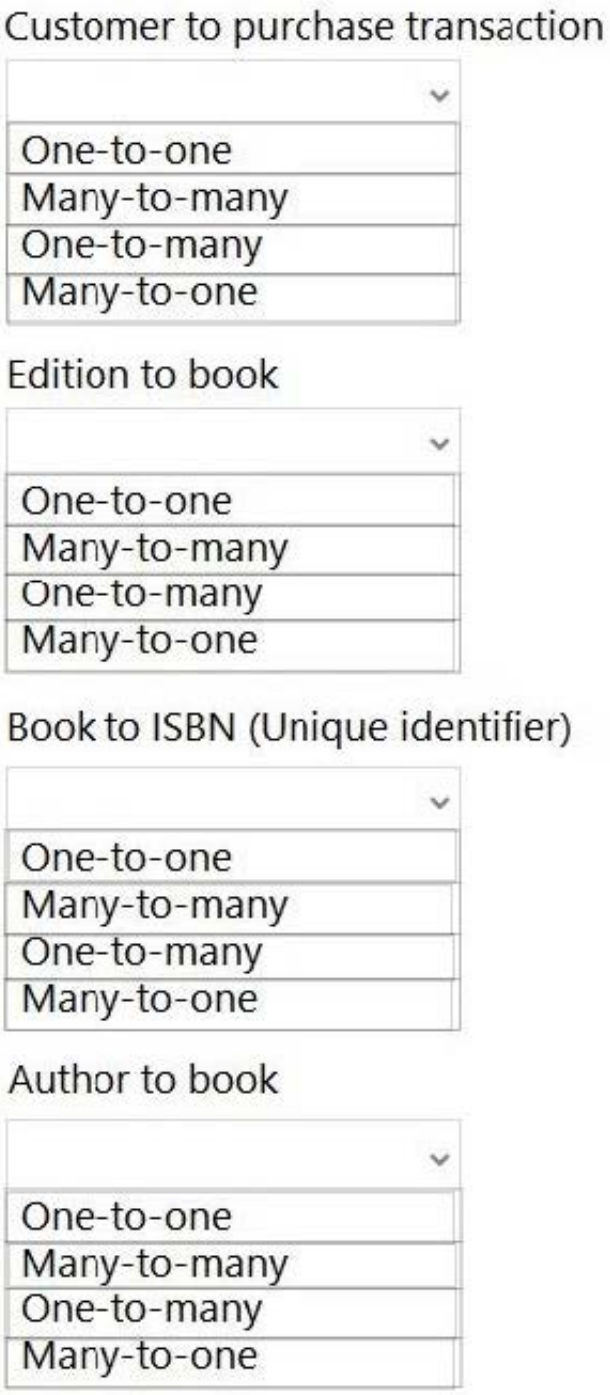
QUESTION 15

HOTSPOT

You need to create and design an ERD that represents the client\\'s bookstore inventory.

Match the entity descriptions to the most appropriate relationship. Each relationship type will be used once.

Hot Area:



Correct Answer:

Customer to purchase transaction

One-to-one
Many-to-many
One-to-many
Many-to-one

Edition to book

One-to-one
Many-to-many
One-to-many
Many-to-one

Book to ISBN (Unique identifier)

One-to-one
Many-to-many
One-to-many
Many-to-one

Author to book

One-to-one
Many-to-many
One-to-many
Many-to-one