ACD300 Q&As

Appian Certified Lead Developer

Pass Appian ACD300 Exam with 100% Guarantee

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

https://www.geekcert.com/acd300.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 presented with the following application requirement:

Users must be able to navigate throughout the application while maintaining complete visibility in the

application structure, and easily navigate to previous locations.\\'\\'

Which Appian Interface Pattern would you recommend?

A. Use Bullous as Cards pattern on the home page lo prominently display application choices.

B. Implement an Activity History pattern to track an organizations activity measures.

C. implement a drilldown report pattern to show detailed information about report data.

D. Include a breadcrumbs pattern on applicable inert aces to show the organizational hierarchy

Correct Answer: C

To meet the application requirement of allowing users to view summary and detailed information about report data, you should implement a drilldown report pattern to show detailed information about report data. A drilldown report pattern is a user interface component that displays data in a hierarchical structure, and allows users to expand or collapse different levels of data. For example, if the user is viewing a sales report by region, the drilldown report pattern could show something like "North America > USA > California > Los Angeles". The user can click on any level of data to see more or less details. This way, the user can see both summary and detailed information about report data, and explore different aspects of the data. The other options are not as effective. Option A, using Tiles as Cards pattern on the home page to prominently display application choices, would provide a way for users to access different parts of the application from the home page, but it would not show summary or detailed information about report data. Option B, implementing an Activity History pattern to track an organization\\'s activity measures, would provide a way for users to see the recent actions performed by themselves or others in the application, but it would not show summary or detailed information about report data. Option D, including a breadcrumbs pattern on applicable interfaces to show the organizational hierarchy, would provide a way for users to see where they are in the application, and easily go back to any previous level by clicking on the corresponding link, but it would not show summary or detailed information about report data.

QUESTION 2

You are taskedto build a large scale acquisition application for a prominent customer. The acquisition process tracks the time it takes is fulfill a purchase request with an award.

The customer has structured the contract so that there are multiple application dev teams.

How should you design for multiple processes and forms, while minimizing repeated code?

A. Create a Center of Excellence (CoE)

B. Create a common objects application.

C. Create a Scrum of Scrums sprint meeting for the team leads

D. Create duplicate processes and forms as needed

Correct Answer: B

To build a large scale acquisition application for a prominent customer, you should design for multiple processes and forms, while minimizing repeated code. One way to do this is to create a common objects application, which is a shared application that contains reusable components, such as rules, constants, interfaces, integrations, or data types, that can be used by multiple applications. This way, you can avoid duplication and inconsistency of code, and make it easier to maintain and update your applications. You can also use the common objects application to define common standards and best practices for your application development teams, such as naming conventions, coding styles, or documentation guidelines. Verified References: [Appian Best Practices], [Appian Design Guidance]

QUESTION 3

You need to design a complex Appian integration to call a RESTful API. The RESTful API will be used to update a case in a customer\\'s legacy system.

What are three prerequisites for designing the integration?

A. Define the HTTP method that the integration will use.

B. Understand the content of the expected body. Deluding each field type and their limits

C. Understand whether this integration will be used in an interface or in a process model

D. Understand the different error codes managed by the API and the process of error handing m Appall

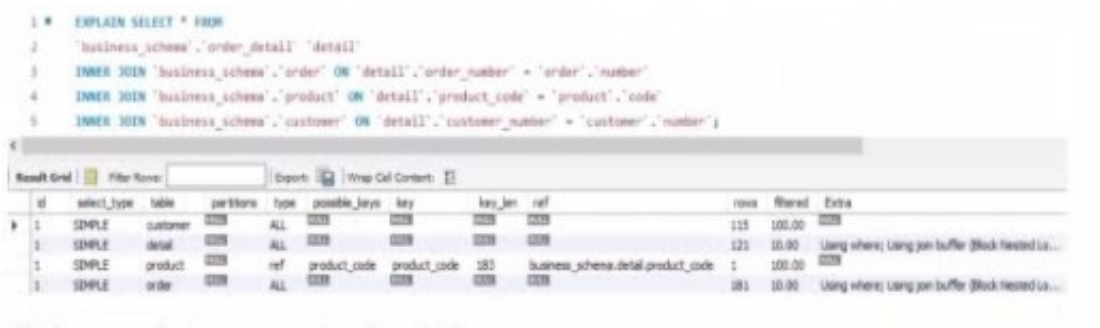
E. Understand the business rules to be applied to ensure the business logic of the data

Correct Answer: ABD

To design a complex Appian integration to call a RESTful API, you need to have some prerequisites, such as: Define the HTTP method that the integration will use. The HTTP method is the action that the integration will perform on the API, such as GET, POST, PUT, PATCH, or DELETE. The HTTP method determines how the data will be sent and received by the API, and what kind of response will be expected. Understand the content of the expected body, including each field type and their limits. The body is the data that the integration will send to the API, or receive from the API, depending on the HTTP method. The body can be in different formats, such as JSON, XML, or form data. You need to understand how to structure the body according to the API specification, and what kind of data types and values are allowed for each field. Understand the different error codes managed by the API and the process of error handling in Appian. The error codes are the status codes that indicate whether the API request was successful or not, and what kind of problem occurred if not. The error codes can range from 200 (OK) to 500 (Internal Server Error), and each code has a different meaning and implication. You need to understand how to handle different error codes in Appian, and how to display meaningful messages to the user or log them for debugging purposes. The other two options are not prerequisites for designing the integration, but rather considerations for implementing it. Understand whether this integration will be used in an interface or in a process model. This is not a prerequisite, but rather a decision that you need to make based on your application requirements and design. You can use an integration either in an interface or in a process model, depending on where you need to call the API and how you want to handle the response. For example, if you need to update a case in real-time based on user input, you may want to use an integration in an interface. If you need to update a case periodically based on a schedule or an event, you may want to use an integration in a process model. Understand the business rules to be applied to ensure the business logic of the data. This is not a prerequisite, but rather a part of your application logic that you need to implement after designing the integration. You need to apply business rules to validate, transform, or enrich the data that you send or receive from the API, according to your business requirements and logic. For example, you may need to check if the case status is valid before updating it in the legacy system, or you may need to add some additional information to the case data before displaying it in Appian.

QUESTION 4

Review the following resultof an explain statement: Which two conclusions can you draw from this?



A. The request is good enough to support a high volume of data. but could demonstrate some limitations if the developer queries information related to the product

B. The worst join isthe one between the table order_detail and order.

C. The join between the tables order_detail, order and customerneeds to be tine-tuned due to indices.

D. The join between the tables 0rder_detail and productneeds to be fine-tuned due to Indices

E. The worst join is the one between the table order_detail and customer

Correct Answer: DE

D. The join between the tables order_detail and product needs to be fine-tuned due to Indices. This is correct because the result of the explain statement showsthat the join between these two tables has a high cost of 0.99, which indicates that it is inefficient and needs to be fine-tuned. One possible reason for the high cost is that there are no indices on the columns that are used for joining these two tables, which leads to a full table scan. Therefore, creating indices on these columns could improve the performance of this join. E. The worst join is the one between the table order_detail and customer. This is correct because the result of the explain statement shows that the join between these two tables has a very high cost of 1.00, which indicates that it is the worst join in terms of efficiency and needs to be fine-tuned. One possible reason for the high cost is that there are no indices on the columns that are used for joining these two tables, which leads to a full table scan. Therefore, creating indices on these columns could improve the performance of this join. The other options are incorrect for the following reasons:

A. The request is good enough to support a high volume of data, but could demonstrate some limitations if the developer queries information related to the product. This is incorrect because the request is not good enough to support a high volume of data, as it has two joins with very high costs that need to be fine- tuned. Moreover, querying information related to the product would not necessarily cause any limitations, as long as the join between order_detail and product is optimized.

B. The worst join is the one between the table order_detail and order. This is incorrect because the result of the explain statement shows that the join between these two tables has a low cost of 0.01, which indicates that it is efficient and does not need to be fine-tuned.

C. The join between the tables order_detail, order and customer needs to be fine- tuned due to indices. This is incorrect because there is no such join between three tables in the result of the explain statement. There are only two joins: one between order_detail and order, and another between order_detail and customer. Each of these joins needs to be fine-tuned separately due to indices.

QUESTION 5

You are reviewing the Engine Performance Logs in Production for a single application thathas been live for six months. This application experiences concurrent user activity and has a fairly sustained load during business hours. The client has reported performance issues with the application during business hours.

During your investigation, you notice a high Work Queue - Java Work Queue Size value in the logs You also notice unattended process activities, including timer events and sending notifications emails, are taking far longer to execute than normal.

The client Increased the number of CPU cores prior to the application going live

What is the next recommendation?

A. Add more engine replicas.

B. Optimize slow-performing user interfaces.

C. Add more application servers.

D. Add execution and analytics shards

Correct Answer: A

Adding more engine replicas will increase the number of threads available to execute unattended process activities, such as timer events and sending notification emails. This will reduce the Java Work Queue Size and improve the performance of the application. Verified References: Appian Engine Performance Logs, Appian Engine Configuration