

100% Free ACD-301–100% Free Valid Dumps Questions | High Hit-Rate Appian Certified Lead Developer Latest Exam Materials

ACC 202 EXAM STUDY SHEET 2026 FULL **SOLUTION GUARANTEED TO PASS**

➤ Perpetual terms on journal entry. Answer: Inventory instead of purchases

Inventory instead of freight in

➤ Periodic terms in JE. Answer: Purchases instead of inventory

Freight in instead of inventory

Equipment is equipment

➤ periodic inventory system. Answer: An inventory system in which a company does not maintain detailed records of goods on hand throughout the period and determines the cost of goods sold only at the end of an accounting period.

➤ FOB. Answer: free on board

➤ FOB destination. Answer: Freight terms indicating that ownership of goods remains with the seller until the goods reach the buyer.

Once you accept the guidance of our ACD-301 training engine, you will soon master all knowledge about the real exam. Because there are all the keypoints of the subject in our ACD-301 training guide. All in all, you will save a lot of preparation troubles of the ACD-301 Exam with the help of our study materials. We will go on struggling and developing new versions of the ACD-301 study materials. Please pay close attention to our products!

Do you still worry about that you can't find an ideal job and earn low wage? Do you still complaint that your working abilities can't be recognized and you have not been promoted for a long time? You can try to obtain the ACD-301 certification and if you pass the exam you will have a high possibility to find a good job with a high income. If you buy our ACD-301 questions torrent you will pass the exam easily and successfully. Our ACD-301 Study Materials are compiled by experts and approved by professionals with experiences for many years. We provide 3 versions for the client to choose and free update. Different version boosts different advantage and please read the introduction of each version carefully before your purchase.

>> ACD-301 Valid Dumps Questions <<

Pass Guaranteed 2026 Reliable Appian ACD-301 Valid Dumps Questions

If you choose the help of BraindumpsPrep, we will spare no effort to help you pass the exam. Moreover, we also provide you with a

year of free after-sales service to update the exam practice questions and answers. Do not hesitate! Please select BraindumpsPrep, it will be the best guarantee for you to pass ACD-301 Certification Exam. Now please add BraindumpsPrep to your shopping cart.

Appian Certified Lead Developer Sample Questions (Q35-Q40):

NEW QUESTION # 35

While working on an application, you have identified oddities and breaks in some of your components. How can you guarantee that this mistake does not happen again in the future?

- A. Provide Appian developers with the "Designer" permissions role within Appian. Ensure that they have only basic user rights and assign them the permissions to administer their application.
- B. Design and communicate a best practice that dictates designers only work within the confines of their own application.
- C. **Create a best practice that enforces a peer review of the deletion of any components within the application.**
- D. Ensure that the application administrator group only has designers from that application's team.

Answer: C

Explanation:

Comprehensive and Detailed In-Depth Explanation:

As an Appian Lead Developer, preventing recurring "oddities and breaks" in application components requires addressing root causes-likely tied to human error, lack of oversight, or uncontrolled changes-while leveraging Appian's governance and collaboration features. The question implies a past mistake (e.g., accidental deletions or modifications) and seeks a proactive, sustainable solution. Let's evaluate each option based on Appian's official documentation and best practices:

A . Design and communicate a best practice that dictates designers only work within the confines of their own application:

This suggests restricting designers to their assigned applications via a policy. While Appian supports application-level security (e.g., Designer role scoped to specific applications), this approach relies on voluntary compliance rather than enforcement. It doesn't directly address "oddities and breaks"-e.g., a designer could still mistakenly alter components within their own application. Appian's documentation emphasizes technical controls and process rigor over broad guidelines, making this insufficient as a guarantee.

B . Ensure that the application administrator group only has designers from that application's team

This involves configuring security so only team-specific designers have Administrator rights to the application (via Appian's Security settings). While this limits external interference, it doesn't prevent internal mistakes (e.g., a team designer deleting a critical component). Appian's security model already restricts access by default, and the issue isn't about unauthorized access but rather component integrity. This step is a hygiene factor, not a direct solution to the problem, and fails to "guarantee" prevention.

C . Create a best practice that enforces a peer review of the deletion of any components within the application:

This is the best choice. A peer review process for deletions (e.g., process models, interfaces, or records) introduces a checkpoint to catch errors before they impact the application. In Appian, deletions are permanent and can cascade (e.g., breaking dependencies), aligning with the "oddities and breaks" described. While Appian doesn't natively enforce peer reviews, this can be implemented via team workflows-e.g., using Appian's collaboration tools (like Comments or Tasks) or integrating with version control practices during deployment. Appian Lead Developer training emphasizes change management and peer validation to maintain application stability, making this a robust, preventive measure that directly addresses the root cause.

D . Provide Appian developers with the "Designer" permissions role within Appian. Ensure that they have only basic user rights and assign them the permissions to administer their application:

This option is confusingly worded but seems to suggest granting Designer system role permissions (a high-level privilege) while limiting developers to Viewer rights system-wide, with Administrator rights only for their application. In Appian, the "Designer" system role grants broad platform access (e.g., creating applications), which contradicts "basic user rights" (Viewer role).

Regardless, adjusting permissions doesn't prevent mistakes-it only controls who can make them. The issue isn't about access but about error prevention, so this option misses the mark and is impractical due to its contradictory setup.

Conclusion: Creating a best practice that enforces a peer review of the deletion of any components (C) is the strongest solution. It directly mitigates the risk of "oddities and breaks" by adding oversight to destructive actions, leveraging team collaboration, and aligning with Appian's recommended governance practices. Implementation could involve documenting the process, training the team, and using Appian's monitoring tools (e.g., Application Properties history) to track changes-ensuring mistakes are caught before deployment. This provides the closest guarantee to preventing recurrence.

Appian Documentation: "Application Security and Governance" (Change Management Best Practices).

Appian Lead Developer Certification: Application Design Module (Preventing Errors through Process).

Appian Best Practices: "Team Collaboration in Appian Development" (Peer Review Recommendations).

NEW QUESTION # 36

Your client's customer management application is finally released to Production. After a few weeks of small enhancements and patches, the client is ready to build their next application. The new application will leverage customer information from the first

application to allow the client to launch targeted campaigns for select customers in order to increase sales. As part of the first application, your team had built a section to display key customer information such as their name, address, phone number, how long they have been a customer, etc. A similar section will be needed on the campaign record you are building. One of your developers shows you the new object they are working on for the new application and asks you to review it as they are running into a few issues. What feedback should you give?

- A. Provide guidance to the developer on how to address the issues so that they can proceed with their work.
- **B. Ask the developer to convert the original customer section into a shared object so it can be used by the new application.**
- C. Create a duplicate version of that section designed for the campaign record.
- D. Point the developer to the relevant areas in the documentation or Appian Community where they can find more information on the issues they are running into.

Answer: B

Explanation:

Comprehensive and Detailed In-Depth Explanation:

The scenario involves reusing a customer information section from an existing application in a new application for campaign management, with the developer encountering issues. Appian's best practices emphasize reusability, efficiency, and maintainability, especially when leveraging existing components across applications.

Option B (Ask the developer to convert the original customer section into a shared object so it can be used by the new application): This is the recommended approach. Converting the original section into a shared object (e.g., a reusable interface component) allows it to be accessed across applications without duplication. Appian's Design Guide highlights the use of shared components to promote consistency, reduce redundancy, and simplify maintenance. Since the new application requires similar customer data (name, address, etc.), reusing the existing section-after ensuring it is modular and adaptable-addresses the developer's issues while aligning with the client's goal of leveraging prior work. The developer can then adjust the shared object (e.g., via parameters) to fit the campaign context, resolving their issues collaboratively.

Option A (Provide guidance to the developer on how to address the issues so that they can proceed with their work):

While providing guidance is valuable, it doesn't address the root opportunity to reuse existing code. This option focuses on fixing the new object in isolation, potentially leading to duplicated effort if the original section could be reused instead.

Option C (Point the developer to the relevant areas in the documentation or Appian Community where they can find more information on the issues they are running into):

This is a passive approach and delays resolution. As a Lead Developer, offering direct support or a strategic solution (like reusing components) is more effective than redirecting the developer to external resources without context.

Option D (Create a duplicate version of that section designed for the campaign record):

Duplication violates Appian's principle of DRY (Don't Repeat Yourself) and increases maintenance overhead. Any future updates to customer data display logic would need to be applied to multiple objects, risking inconsistencies.

Given the need to leverage existing customer information and the developer's issues, converting the section to a shared object is the most efficient and scalable solution.

NEW QUESTION # 37

You are reviewing the Engine Performance Logs in Production for a single application that has 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 notification 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 execution and analytics shards
- **B. Add more engine replicas.**
- C. Add more application servers.
- D. Optimize slow-performing user interfaces.

Answer: B

Explanation:

As an Appian Lead Developer, analyzing Engine Performance Logs to address performance issues in a Production application requires understanding Appian's architecture and the specific metrics described. The scenario indicates a high "Work Queue - Java Work Queue Size," which reflects a backlog of tasks in the Java Work Queue (managed by Appian engines), and delays in unattended process activities (e.g., timer events, email notifications). These symptoms suggest the Appian engines are overloaded, despite the client increasing CPU cores. Let's evaluate each option:

A . Add more engine replicas: This is the correct recommendation. In Appian, engine replicas (part of the Appian Engine cluster) handle process execution, including unattended tasks like timers and notifications. A high Java Work Queue Size indicates the engines are overwhelmed by concurrent activity during business hours, causing delays. Adding more engine replicas distributes the workload, reducing queue size and improving performance for both user-driven and unattended tasks. Appian's documentation recommends scaling engine replicas to handle sustained loads, especially in Production with high concurrency. Since CPU cores were already increased (likely on application servers), the bottleneck is likely the engine capacity, not the servers.

B . Optimize slow-performing user interfaces: While optimizing user interfaces (e.g., SAIL forms, reports) can improve user experience, the scenario highlights delays in unattended activities (timers, emails), not UI performance. The Java Work Queue Size issue points to engine-level processing, not UI rendering, so this doesn't address the root cause. Appian's performance tuning guidelines prioritize engine scaling for queue-related issues, making this a secondary concern.

C . Add more application servers: Application servers handle web traffic (e.g., SAIL interfaces, API calls), not process execution or unattended tasks managed by engines. Increasing application servers would help with UI concurrency but wouldn't reduce the Java Work Queue Size or speed up timer/email processing, as these are engine responsibilities. Since the client already increased CPU cores (likely on application servers), this is redundant and unrelated to the issue.

D . Add execution and analytics shards: Execution shards (for process data) and analytics shards (for reporting) are part of Appian's data fabric for scalability, but they don't directly address engine workload or Java Work Queue Size. Shards optimize data storage and query performance, not real-time process execution. The logs indicate an engine bottleneck, not a data storage issue, so this isn't relevant. Appian's documentation confirms shards are for long-term scaling, not immediate performance fixes.

Conclusion: Adding more engine replicas (A) is the next recommendation. It directly resolves the high Java Work Queue Size and delays in unattended tasks, aligning with Appian's architecture for handling concurrent loads in Production. This requires collaboration with system administrators to configure additional replicas in the Appian cluster.

Appian Documentation: "Engine Performance Monitoring" (Java Work Queue and Scaling Replicas).

Appian Lead Developer Certification: Performance Optimization Module (Engine Scaling Strategies).

Appian Best Practices: "Managing Production Performance" (Work Queue Analysis).

NEW QUESTION # 38

The business database for a large, complex Appian application is to undergo a migration between database technologies, as well as interface and process changes. The project manager asks you to recommend a test strategy. Given the changes, which two items should be included in the test strategy?

- A. Tests for each of the interfaces and process changes
- B. Tests that ensure users can still successfully log into the platform
- C. Penetration testing of the Appian platform
- D. Internationalization testing of the Appian platform
- E. A regression test of all existing system functionality

Answer: A,E

Explanation:

Comprehensive and Detailed In-Depth Explanation:

As an Appian Lead Developer, recommending a test strategy for a large, complex application undergoing a database migration (e.g., from Oracle to PostgreSQL) and interface/process changes requires focusing on ensuring system stability, functionality, and the specific updates. The strategy must address risks tied to the scope-database technology shift, interface modifications, and process updates-while aligning with Appian's testing best practices. Let's evaluate each option:

A . Internationalization testing of the Appian platform

Internationalization testing verifies that the application supports multiple languages, locales, and formats (e.g., date formats). While valuable for global applications, the scenario doesn't indicate a change in localization requirements tied to the database migration, interfaces, or processes. Appian's platform handles internationalization natively (e.g., via locale settings), and this isn't impacted by database technology or UI/process changes unless explicitly stated. This is out of scope for the given context and not a priority.

B . A regression test of all existing system functionality:

This is a critical inclusion. A database migration between technologies can affect data integrity, queries (e.g., a!queryEntity), and performance due to differences in SQL dialects, indexing, or drivers. Regression testing ensures that all existing functionality-records, reports, processes, and integrations-works as expected post-migration. Appian Lead Developer documentation mandates regression testing for significant infrastructure changes like this, as unmapped edge cases (e.g., datatype mismatches) could break the application. Given the "large, complex" nature, full-system validation is essential to catch unintended impacts.

C . Penetration testing of the Appian platform

Penetration testing assesses security vulnerabilities (e.g., injection attacks). While security is important, the changes described-database migration, interface, and process updates-don't inherently alter Appian's security model (e.g., authentication, encryption), which is managed at the platform level. Appian's cloud or on-premise security isn't directly tied to database technology unless new vulnerabilities are introduced (not indicated here). This is a periodic concern, not specific to this migration, making it less relevant

than functional validation.

D . Tests for each of the interfaces and process changes:

This is also essential. The project includes explicit "interface and process changes" alongside the migration. Interface updates (e.g., SAIL forms) might rely on new data structures or queries, while process changes (e.g., modified process models) could involve updated nodes or logic. Testing each change ensures these components function correctly with the new database and meet business requirements. Appian's testing guidelines emphasize targeted validation of modified components to confirm they integrate with the migrated data layer, making this a primary focus of the strategy.

E . Tests that ensure users can still successfully log into the platform

Login testing verifies authentication (e.g., SSO, LDAP), typically managed by Appian's security layer, not the business database. A database migration affects application data, not user authentication, unless the database stores user credentials (uncommon in Appian, which uses separate identity management). While a quick sanity check, it's narrow and subsumed by broader regression testing (B), making it redundant as a standalone item.

Conclusion: The two key items are B (regression test of all existing system functionality) and D (tests for each of the interfaces and process changes). Regression testing (B) ensures the database migration doesn't disrupt the entire application, while targeted testing (D) validates the specific interface and process updates. Together, they cover the full scope-existing stability and new functionality-aligning with Appian's recommended approach for complex migrations and modifications.

Appian Documentation: "Testing Best Practices" (Regression and Component Testing).

Appian Lead Developer Certification: Application Maintenance Module (Database Migration Strategies).

Appian Best Practices: "Managing Large-Scale Changes in Appian" (Test Planning).

NEW QUESTION # 39

You need to export data using an out-of-the-box Appian smart service. Which two formats are available (or data generation)?

- A. JSON
- B. Excel
- C. CSV
- D. XML

Answer: B,C

Explanation:

The two formats that are available for data generation using an out-of-the-box Appian smart service are:

A . CSV. This is a comma-separated values format that can be used to export data in a tabular form, such as records, reports, or grids. CSV files can be easily opened and manipulated by spreadsheet applications such as Excel or Google Sheets.

C . Excel. This is a format that can be used to export data in a spreadsheet form, with multiple worksheets, formatting, formulas, charts, and other features. Excel files can be opened by Excel or other compatible applications.

The other options are incorrect for the following reasons:

B . XML. This is a format that can be used to export data in a hierarchical form, using tags and attributes to define the structure and content of the data. XML files can be opened by text editors or XML parsers, but they are not supported by the out-of-the-box Appian smart service for data generation.

D . JSON. This is a format that can be used to export data in a structured form, using objects and arrays to represent the data. JSON files can be opened by text editors or JSON parsers, but they are not supported by the out-of-the-box Appian smart service for data generation. Verified Appian Documentation, section "Write to Data Store Entity" and "Write to Multiple Data Store Entities".

NEW QUESTION # 40

.....

For candidates who will attend an exam, some practice for it is necessary. ACD-301 Exam Dumps of us will give you the practice you need. ACD-301 exam dumps of us contain the knowledge point of the exam. Skilled professionals will verify the questions and answers, which will guarantee the correctness. Besides, we also offer you free update for one year after purchasing, and the update version will send to your email address automatically.

ACD-301 Latest Exam Materials: <https://www.briandumpsprep.com/ACD-301-prep-exam-braindumps.html>

For example, if you are a college student, you can study and use online resources through the student column of our ACD-301 learning guide, and you can choose to study our ACD-301 exam questions in your spare time, Appian ACD-301 Valid Dumps Questions Sure, we have discounts for promotion in some special festival, Maybe, you have been confused by various website and ACD-301 Latest Exam Materials - Appian Certified Lead Developer prep study material.

He has been researching in the areas of strategic sourcing and Certification ACD-301 Cost procurement, service operations management, and supply chain management. Focusing on a combination of digital and analog circuit theory, this comprehensive volume will help engineers ACD-301 who work with digital systems, shorten their product development cycles, and fix their latest high-speed design problems.

First-Grade ACD-301 Valid Dumps Questions & Leader in Qualification Exams & Useful ACD-301: Appian Certified Lead Developer

For example, if you are a college student, you can study and use online resources through the student column of our ACD-301 learning guide, and you can choose to study our ACD-301 exam questions in your spare time.

Sure, we have discounts for promotion in some ACD-301 Valid Dumps Questions special festival, Maybe, you have been confused by various website and Appian Certified Lead Developer prep study material, You can pass at first time by using our ACD-301 sure prep torrent and get a high score in the actual test.

We have researched an intelligent system to help testing errors of the ACD-301 exam materials.