

PDII-JPN Updated Demo | PDII-JPN 100% Free Test Certification Cost



P.S. Free 2026 Salesforce PDII-JPN dumps are available on Google Drive shared by ExamDiscuss: <https://drive.google.com/open?id=1fFtlNIR6lzVMRpg2P6Pp9qWUKEcjErBm>

Knowledge is defined as intangible asset that can offer valuable reward in future, so never give up on it and our PDII-JPN exam preparation can offer enough knowledge to cope with the exam effectively. To satisfy the needs of exam candidates, our experts wrote our PDII-JPN practice materials with perfect arrangement and scientific compilation of messages, so you do not need to study other PDII-JPN training questions to find the perfect one anymore.

The client can try out and download our PDII-JPN training materials freely before their purchase so as to have an understanding of our PDII-JPN exam questions and then decide whether to buy them or not. The website pages of our product provide the details of our PDII-JPN learning questions. You can see the demos of our PDII-JPN Study Guide, which are part of the all titles selected from the test bank and the forms of the questions and answers and know the form of our software on the website pages of our PDII-JPN study materials.

>> PDII-JPN Updated Demo <<

PDII-JPN Test Certification Cost | PDII-JPN Guide

Perhaps you worry about the quality of our PDII-JPN exam questions. We can make solemn commitment that our PDII-JPN study materials have no mistakes. All contents are passing rigid inspection. You will never find small mistakes such as spelling mistakes and typographical errors in our PDII-JPN learning guide. No one is willing to buy a defective product. And our PDII-JPN practice braindumps are easy to understand for all the candidates.

Salesforce Sample Questions (Q65-Q70):

NEW QUESTION # 65

開発者は、フィールド履歴追跡に依存する `AccountHistoryManager` という名前のクラスを作成しました。このクラスには `getAccountHistory` という静的メソッドがあり、このメソッドは `Account` にパラメーターとして含まれ、関連付けられた `AccountHistory` オブジェクトレコードのリストを返します。

次のテストは失敗します。

このテストに合格するには何をすべきでしょうか?

- A. `@isTest (SeeAllData=true)` を使用して組織の履歴データを表示し、`AccountHistory` レコードをクエリします。
- B. テスト セットアップで `AccountHistory` レコードを手動で作成し、それらを取得するクエリを作成します。
- C. テストを使用します。 `getAccountHistory()` の `isRunningTest []` を使用して、条件付きで偽の `AccountHistory` レコードを返します。
- D. このコードはテストできないため、タクトメソッドは延期されるべきです

Answer: A

Explanation:

To make the test pass, @isTest(SeeAllData=true) should be used to allow the test to access historical data from the org. This is required because field history tracking data is not copied to a test context, so you need to allow the test to access live data to assert against it. References: Apex Developer Guide - IsTest Annotation

NEW QUESTION # 66

Salesforce 管理者と Cloud Kicks は、米国内のすべての郵便番号とその郵便番号が属する Cloud Kicks 販売地域を保存するために、Region_c というカスタム オブジェクトを作成しました。

Cloud Kicks は、リードの郵便番号に基づいてリージョンを設定するためのリード上のトリガーを必要としています。

この要求を満たす最も効率的な方法はどのコード セグメントですか？

- A.
- B.
- C.
- D.

Answer: A

NEW QUESTION # 67

どのコード スニペットが最もメモリ効率の高い方法でレコードを処理し、ガバナ制限などを回避するか「Apex ヒープ サイズが大きすぎます」？

- A.
- B.
- C.
- D.

Answer: B

Explanation:

Option D, which uses the Database.query() method in combination with a for loop, is the most memory-efficient way to process records. This is known as the Query Locator pattern, which allows you to work with a large number of records by querying them in batches. This approach helps to avoid hitting the governor limit for "Apex heap size too large" as it processes one record at a time from the database without storing the entire result set in memory.

References:

Working with Very Large SOQL Queries

Apex Governor Limits

NEW QUESTION # 68

開発者は、取引先の登録に使用される Lightning Web コンポーネントからの入力に基づいて取引先を更新する Apex クラスを作成しました。アカウントの更新は、まだ登録されていない場合にのみ行う必要があります。

ユーザーが同時に更新を行った場合に、同じアカウントに対する互いの更新を上書きしないようにするには、開発者は何をすべきでしょうか？

- A.
- B.
- C.
- D.

Answer: B

Explanation:

When multiple users are updating the same record at the same time, there is a risk of overwriting each other's changes. Salesforce provides a mechanism called 'record locking' to prevent this from happening.

Option D is correct because including FOR UPDATE in the SOQL query locks the retrieved records for the duration of the transaction. This prevents other transactions from updating the record until the current transaction is complete, which is essential for avoiding race conditions where two users might overwrite each other's updates.

Option A is incorrect because while adding a try/catch block around the update operation is good practice for handling exceptions, it does not prevent overwrites from concurrent updates.

Option B is incorrect because using upsert instead of update does not address the problem of concurrent updates. The upsert operation is used to either insert a new record or update an existing one based on whether a record with a matching ID or external ID already exists.

Option C is incorrect because FOR UPDATE should be used in the SOQL query to lock the records, not in the SELECT statement itself.

References:

Salesforce Developer Documentation on Locking Statements: Locking Statements
Salesforce Developer Blog on Handling Concurrency in Apex: Handling Concurrency in Apex

NEW QUESTION # 69

以下のマークアップを参照してください。

HTML

```
<template>
<lightning-record-form
record-id={recordId}
object-api-name="Account"
layout-type="Full">
</lightning-record-form>
</template>
```

Lightning Webコンポーネントは、取引先名と、オブジェクトに存在する275個のカスタム項目のうち2つを表示します。カスタム項目は正しく宣言され、値も入力されています。しかし、開発者はコンポーネントのパフォーマンスが遅いという苦情を受けています。開発者はパフォーマンスを改善するために何をすればよいでしょうか？

- A. コンポーネントに `density="compact"` を追加します。
- B. `layout-type="Full"` を `layout-type="Partial"` に置き換えます。
- C. `layout-type="Full"` を `fields={fields}` に置き換えます。
- D. コンポーネントに `cache="true"` を追加します。

Answer: C

Explanation:

The `lightning-record-form` is a powerful, high-level component that simplifies data entry and display.

However, its performance is heavily influenced by the `layout-type` attribute. When `layout-type="Full"` is used, the component fetches and renders every single field defined on the object's "Full" page layout in the Salesforce metadata. In this case, the Account object has 275 fields. Fetching and rendering such a large volume of metadata and data causes a significant performance lag, even if the user only cares about three specific fields.

To improve performance, the developer should switch from a layout-based approach to a field-based approach. By removing the `layout-type` attribute and adding the `fields` attribute (Option A), the developer can pass an array of only the specific field API names required (e.g., `['Name', 'CustomField1__c', 'CustomField2__c']`). This drastically reduces the amount of data requested from the Lightning Data Service (LDS) and minimizes the DOM elements the browser needs to render. Option C is incorrect because "Partial" is not a valid value for `layout-type`; the only supported values are "Full" and "Compact". Option B only affects the visual spacing of the fields and does not reduce the data load. Option A is the direct and most effective way to optimize component responsiveness by limiting the data scope.

NEW QUESTION # 70

.....

Our company really took a lot of thought in order to provide customers with better PDII-JPN learning materials. First of all, in the setting of product content, we have hired the most professional team who analyzed a large amount of information and compiled the most reasonable PDII-JPN Exam Questions. And you can find the most accurate on our PDII-JPN study braindumps. Secondly, our services are 24/7 available to help our customers solve all kinds of questions.

PDII-JPN Test Certification Cost: <https://www.examdiscuss.com/Salesforce/exam/PDII-JPN/>

