# [Q86-Q103 2024 Updated PDII Tests Engine pdf - All Free Dumps Guaranteed!

2024 Updated PDII Tests Engine pdf - All Free Dumps Guaranteed!
Latest Salesforce Developers PDII Actual Free Exam Questions

Salesforce PDII (Salesforce Certified Platform Developer II) certification exam is designed for experienced Salesforce developers who have a solid understanding of Apex programming language, Visualforce and Lightning Components. Salesforce Certified Platform Developer II (PDII) certification validates a developer's skills in building complex business logic and interfaces on the Salesforce platform. The PDII certification exam is the second level of Salesforce developer certification after the Salesforce Certified Platform Developer I (PD1) exam. It is a challenging exam that requires extensive hands-on experience and knowledge of Salesforce development best practices.

Salesforce PDII (Salesforce Certified Platform Developer II) Exam is a certification that is designed for developers who want to showcase their expertise in Salesforce development. Salesforce Certified Platform Developer II (PDII) certification exam is a step above the Salesforce Certified Platform Developer I certification and is intended for developers who have advanced knowledge of Salesforce development. PDII certification validates a developer's ability to design and implement advanced business logic and interfaces using Apex code, Visualforce, and Lightning components.

**Q86.** What is a potential design issue with the following code?

trigger accountTrigger on Account (before update){ Boolean processOpportunity = false; List<opportunity> opptysClosedLost = new List<opportunity>() List<opportunity> IstAllOpp = [select StageName from Opportunity where accountId IN :Trigger.newMap.keySet()]; if(!IstAllOpp.isEmpty()) processOpportunity = true; while(processOpportunity){ for(opportunity o : IstAllOpp) if(o.StageName

&#8216;Closed &#8211; Lost&#8217;) opptysClosedLost.add(o); processOpportunity = false; if (!opptysClosedLost.isEmpty()) delete opptysClosedLost;
* SOQL could be avoided by creating a formula field for StageName in Account from the related Opportunity
* The code will result in a System.LimitException : Too many script statements error
* The code will result in a System.DmlException:Entity_is_Deleted error
* The code will result in a System.LimitException: Apex CPU time limit exceeded error

**Q87.** What is the optimal technique a developer should use to programmatically retrieve Global Picklist options in a Test Method?
* Perform a callout to the Metadata API.
* Use the Schema namespace.
* Perform a SOQL Query.
* Use a static resource.

**Q88.** A developer is creating unit tests for code that makes SOAP web service callouts. The developer needs to insert some test data as a part of the unit tests setup. What are three actions to enable this functionality?
* Surround the callout with Test.startTest(), Test.stopTest().
* Surround the data insertion with Test.startTest(), Test.stopTest().
* Implement the WebServiceMock interface.
* Update code to call Test.setMock().
* Implement the HttpCalloutMock interface.

**Q89.** A company has an Apex process that makes multiple extensive database operations and web service callouts. The database processes and web services can take a long time to run and must be run sequentially.

How should the developer write this Apex code without running into running into governor limits and system limitations?
* Use multiple @future methods for each process and callout
* Use Apex Scheduler to schedule each process
* Use Queueable Apex to chain the jobs to run sequentially
* Use Limits class to stop entire process once governor limits are reached

**Q90.** An org has a requirement that addresses on Contacts and Accounts should be normalized to a company standard by Apex code any time that they are saved.

What is the optimal way to implement this?
* Apex trigger on Contact that calls the Account trigger to normalize the address
* Apex triggers on Contact and Account that normalize the address
* Apex trigger on Account that calls the Contact trigger to normalize the address
* Apex triggers on Contact and Account that call a helper class to normalize the address

**Q91.** A developer implemented a custom data table in a Lightning web component with filter functionality.

However, users are submitting support tickets about long load times when the filters are changed. The component uses an Apex method that is called to query for records based on the selected filters.

What should the developer do to improve performance of the component?
* Return all records into a list when the component is created and filter the array In JavaScript.
* Use a selective SOQL query with a custom Index.
* Use SOSL to query the records on filter change.
* Use setstoraclel() in the Apex method to store the response In the client-side cache.

When faced with performance issues in a Lightning Web Component (LWC) due to SOQL query load times, the optimal approach is often to improve the query&#8217;s selectivity. This can be achieved by using a selective SOQL query with a custom index. Salesforce can create custom indexes to improve the performance of queries that cannot be optimized through standard indexing. When a query is selective, it can efficiently retrieve records from the database using the index, thus reducing the query execution time and speeding up the component&#8217;s performance when filters are changed. The other options (returning all records, using SOSL, or client-side caching) do not directly address the root cause of the performance issue, which is the need for a more efficient database operation.

References:

Make SOQL Query Selective

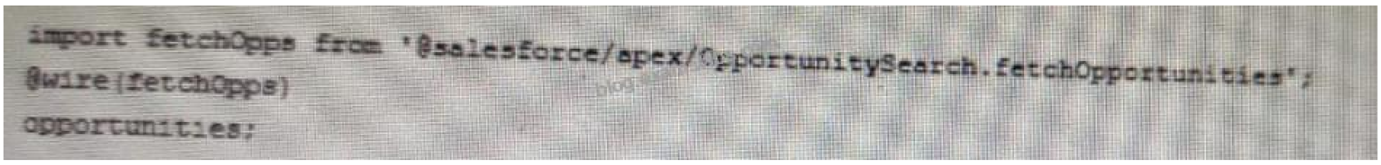
Use of Indexes in SOQL Queries

**Q92.** How can the DISTANCE and GEOLOCATION functions be used i|n SOQL queries? (Choose two.)
* To filter results based on distance from a latitude and longitude
* To get the distance results from a latitude and longitude
* To order results by distance from a latitude or longitude
* To group results in distance ranges from a latitude and longitude

**Q93.** A developer migrated functionality from JavaScript demoting to a Lightning web component and wants to use the existing getOpportunities() method to provide data.

Which modification to the method is necessary?
* A The method must be decorated with AuraEnabled.
* The method must return a JSON Object.
* The method must be decorated with (cacheable=true).
* The method must return a String of a serialized JSON Array.

**Q94.** Refer to the code snippet below:



```
import fetchOpps from '@salesforce/apex/OpportunitySearch.fetchOpportunities';
@wire(fetchOpps)
opportunities;
```

When a Lightning web component is rendered, a list of opportunity that match certain criteria should be retrieved from the database and displayed to the end-user.

Which three considerations must the developer implement to make the fetchOpps method available within the Lightning web component?
* The fetchOpps method must be annotated with the @InvocableMethod annotation.
* The fetchOpps method must specify the (continustion-true) attribute
* The fetchOpps method cannot mutate the result set retrieved from the database.
* The fetchOpps method must specify the (cacheable =true) attribute
* The fecthOpps method must be annotated with the @ AuraEnabled annotation.

**Q95.** A company has reference data stored in multiple custom metadata records that represent default information and delete behavior for certain geographic regions.

When a contact is inserted, the default information should be set on the contact from the custom metadata records based on the contact&#8217;s address information.

Additionally, if a user attempts to delete a contact that belongs to a flagged region, the user must get an error message.

Depending on company personnel resources, what are two ways to automate this?

Choose 2 answers
* Remote action
* Flow Builder
* Apex trigger
* Apex invocable method
To automate the process based on the contact&#8217;s address information and prevent deletion from flagged regions:

Flow Builder: Can be used to set default information on records when they are created.

Apex Trigger: Allows for complex logic to be executed before or after data manipulation operations, including insert and delete. It can be used to enforce rules such as preventing deletion.

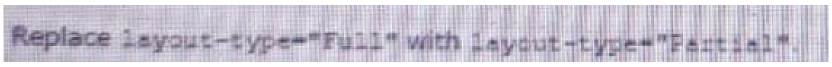
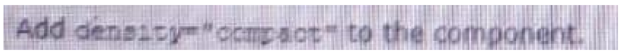
References:

Flow Builder: Salesforce Help Article

Apex Triggers: Apex Developer Guide

**Q96.** Refer to the markup below:

A Lightning web component displays the Account name and two custom fields out of 275 that exist on the object. The developer receives complaints that the component performs slowly.

What can the developer do to improve the performance?
* 
Replace layout-type="Full" with layout-type="Partial".
* 
Add density="compact" to the component.
*

**Q97.** In an organization that has multi-currency enabled, a developer Is tasked with building a Lighting component that displays the top ten Opportunities most recently accessed by the logged in user. The developer must ensure the Amount and LastModifiedDate field values are displayed according to the user&#8217;s locale.

What is the most effective approach to ensure values displayed respect the user&#8217;s locale settings?
* Use REGEX expressions to format the values retrieved via SOQL.
* Use a wrapper class to format the values retrieved via SOQL.
* Use the FOR VIEW clause in the SOQL query.
* Use the FORMAT () function in the SOQL query.

To ensure the values displayed respect the user&#8217;s locale settings, the most effective approach is to use the FORMAT() function in the SOQL query. The FORMAT() function converts the field values in the query result to the user&#8217;s locale, if the field is a date, date/time, currency, number, or percent field. The FORMAT() function also respects the multi-currency settings, and converts the currency values to the user&#8217;s currency. The developer can use the FORMAT() function for the Amount and LastModifiedDate fields in the SOQL query, and display the formatted values in the Lightning component. Using REGEX expressions to format the values retrieved via SOQL would not be effective, as it would require complex and error-prone string manipulation, and it would not respect the multi-currency settings. Using a wrapper class to format the values retrieved via SOQL would not be effective, as it would require additional Apex code and memory allocation, and it would not respect the multi-currency settings. Using the FOR VIEW clause in the SOQL query would not be effective, as it would only mark the records as viewed, but it would not format the field values according to the user&#8217;s locale. Reference: [SOQL Functions], [Working with Currencies]

**Q98.** An Apex class does not achieve expected code coverage. The testSetup method explicitly calls a method in the Apex class. How can the developer generate the code coverage?
* Add @testVisible to the method in the class the developer is testing.
* Use system.assert() in testSetup to verify the values are being returned.
* Call the Apex class method from a testMethod instead of the testSetup method.
* Verify the user has permissions passing a user into System.runAs().

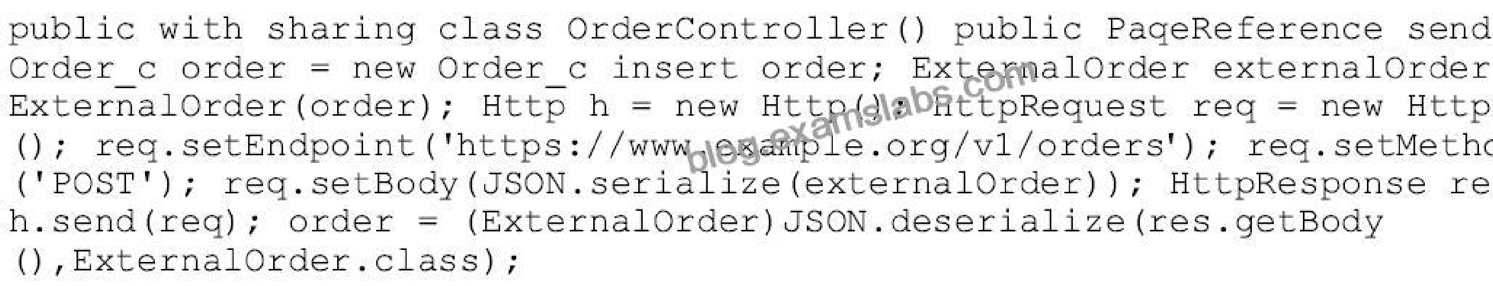
**Q99.** A developer is writing a Jest for a Lightning web component that conditionally displays child components based on a user&#8217;s checkbox selections.

What should the developer do to property test that the correct components display and hide for each scenario?
* Reset the DOM after each test with the after Each method.
* Add a teardown block to reset the DOM after each test.
* Create a new describe block for each test
* Create a new jsdom instance for each test

**Q100.** Recently a Salesforce org&#8217;s integration failed because it exceeded the number of allowed API calls in a

24-hour period. The integration handles a near real-time, complex insertion of data into Salesforce. The flow of data is as follows: The integration looks up Contact records with a given email address and, if found, the integration adds a Task to the first matching Contact it finds. If a match is not found, the integration looks up Lead records with a given email address and, if found, the integration adds a Task to the first matching Lead it finds. If a match is not found, the integration will create a Lead and a Task for that newly created Lead. What is one way in which the integration can stay near real-time, but not exceed the number of allowed API calls in a 24-hour period?
* Use the REST API as well as the SOAP API to effectively double the API calls allowed in a 24-hour period.
* write a custom Apex web service that, given an email address, does all of the logic the integration code was doing.
* Create several Apex InboundEmailHandlers to accept calls from the third-party system, thus bypassing the API limits.
* Create an Inbound Message that, using Flow, can do all of the logic the integration code was doing.

**Q101.** A developer writes the following code:



```
public with sharing class OrderController() public PaqeReference send
Order_c order = new Order_c insert order; ExternalOrder externalOrder
ExternalOrder(order); Http h = new Http(); HttpRequest req = new Http
(); req.setEndpoint('https://www.example.org/v1/orders'); req.setMetho
('POST'); req.setBody(JSON.serialize(externalOrder)); HttpResponse re
h.send(req); order = (ExternalOrder)JSON.deserialize(res.getBody
(),ExternalOrder.class);
```

While testing the code, the developer receives the following error message: System.CalloutException : You have uncommitted work pending What should the developer do? (Choose two.)
* Use the asyncSend() method of the HTTP class to send the request in async context
* Ensure all callouts are completed prior to executing DML statements
* Move the web service callout into an future method
* Use Database.insert (order, true) to immediately commit any database changes

**Q102.** Universal Containers wants to use a Customer Community with Customer Community Plus licenses to allow their customers access to track how many containers they have rented and when they are due back, Universal Containers uses a Private sharing model for External users.

Many of their customers are multi-national corporations with complex Account hierarchies. Each account on the hierarchy represents a department within the same business.

One of the requirements is to allow certain community users within the same Account hierarchy to see several departments&#8217; containers, based on a custom junction object thatrelated the Contact to the various Account records that represent the departments.

Which solution solves these requirements?
* An Apex trigger that creates Apex managed sharing records based on the junction object&#8217;s relationships
* A custom list view on the junction object with filters that will show the proper records based on owner
* A Lightning web component on the Community Home Page that uses Lightning Data Services.
* A Visualforce page that uses a custom controller that specifies without sharing to expose the records

**Q103.** How should a developer verify that a specific Account record is being tested in a test class for a visualforce controller?
* Insert the Account in the test class, instantiate the page reference in the test class, then use System.currentPageReference().getParameters{}.put() to set the Account ID.
* Instantiate the page reference in the test class, insert the Account in the test class, then use seeAHData-true to view the Account.
* Insert the Account into Salesforce, instantiate the page reference in the test class, then use System.setParentRecordId().get() to set the Account ID.
* Instantiate the page reference in the test class, insert the Account in the test class, then use System.setParentRecordld().get() to set the Account ID.