信頼的なPDI日本語版テキスト内容 &合格スムーズPDI入門知識 |ユニークなPDIコンポーネント

無料でクラウドストレージから最新のCertShiken PDI PDFダンプをダウンロードする：https://drive.google.com/open?id=1UAXefFkzq9XYG8zdHQgZx79DWR6V46IO

PDI試験はIT業界でのあなたにとって重要です。あなたはPDI試験に悩んでいますか？試験に合格できないことを心配していますか？我々の提供した一番新しくて全面的なSalesforceのPDI問題集はあなたのすべての需要を満たすことができます。資格をもらうのはあなたの発展の第一歩で、我々のPDI日本語対策はあなたを助けて試験に合格して資格をもらうことができます。

Salesforce PDI（Platform Developer I）認定試験は、Salesforce開発者としてのスキルを検証する素晴らしい方法です。この認定試験は、Salesforceプラットフォーム上でカスタムアプリケーションを構築した経験のある開発者を対象に設計されています。試験は、Apex、Visualforce、データモデリングなど、幅広いトピックをカバーしています。

>> PDI日本語版テキスト内容 <<

ユニークなPDI日本語版テキスト内容試験-試験の準備方法-権威のあるPDI入門知識

ユーザーに多くの不必要なトラブルを保存するために、オンライン学習プラットフォームのPDI研究質問の研究と開発を完了しました。ユーザーはダウンロードしてインストールする必要はなく、デジタルデバイスにブラウザーがあれば必要です。 PDIテストガイドのオンライン操作。この種の学習方法は、特にPDI認定を取得するペースが速いときに、ユーザーにとって非常に便利です。 PDIトレーニング資料を使用すると、PDI学習資料のすべての操作を完全に適用できます。

Salesforce Platform Developer I (PDI) 認定 PDI 試験問題 (Q159-Q164):

質問 # 159
A developer needs to avoid potential system problems that can arise in a multi-tenant architecture. Which requirement helps prevent poorty written applications from being deployed to a production environment?

- A. All validation rules must be active before they can be deployed.
- B. SOQL queries must reference sObActs with their appropriate namespace.
- C. All Apex code must be annotated with the with sharing keyword.
- D. Unit tests must cover at least 75% of the application's Apex code

正解：D

質問 # 160

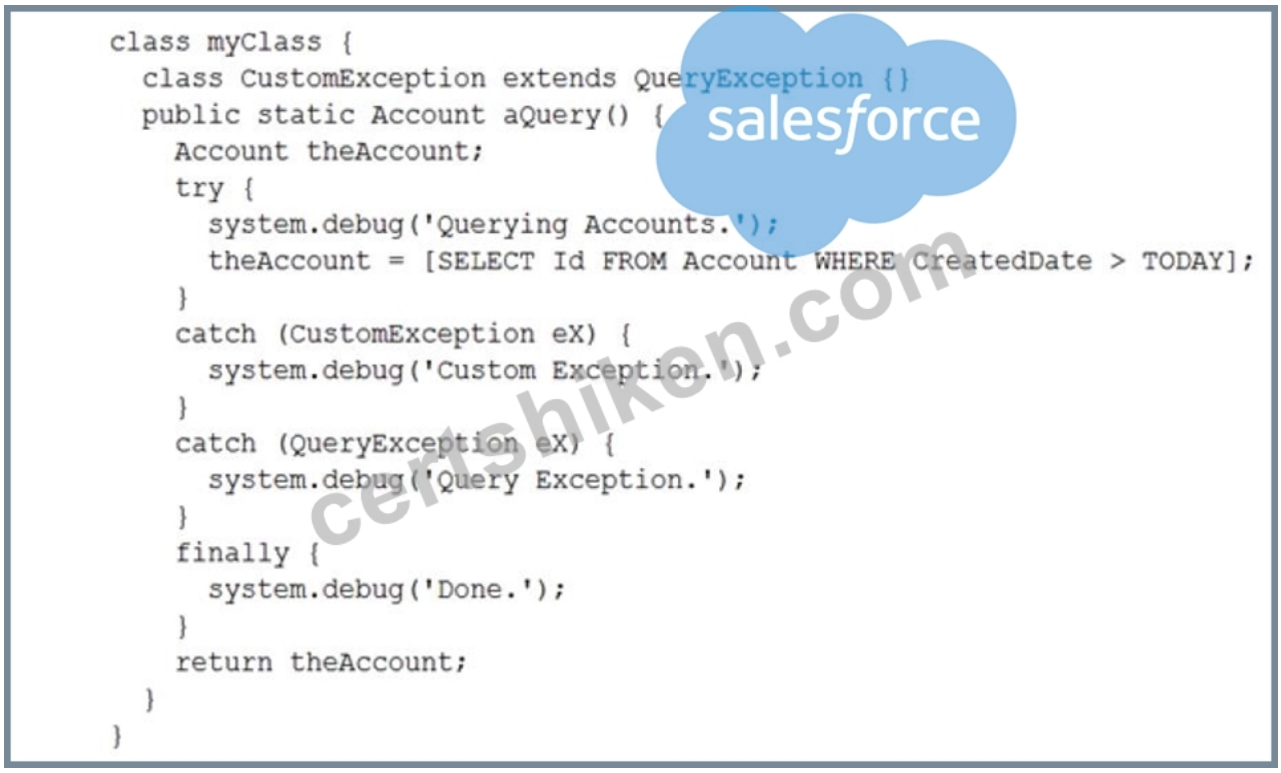
Assuming that name is a String obtained by an <apex:inpucText> tag on a Visualforce page, which two SOQL queries performed are safe from SOQL injection? Choose 2 answers

- A.
```
String query = 'SELECT Id FROM Account WHERE Name LIKE \'%' +
String.escapeSingleQuotes(name) + '%\'';
List<Account> results = Database.query(query);
```
- B.
```
String query = '%' + name + '%';
List<Account> results = [SELECT Id FROM Account WHERE Name LIKE
```
- C.
```
String query = 'SELECT Id FROM Account WHERE Name LIKE \'%' +
name.noQuotes() + '%\'';
List<Account> results = Database.query(query);
```
- D.
```
String query = 'SELECT Id FROM Account WHERE Name LIKE \'%' +
name + '%\'';
List<Account> results = Database.query(query);
```

正解：A、B

解説：
Option A:
String query = 'SELECT Id FROM Account WHERE Name LIKE \'%' + String.escapeSingleQuotes(name) + '%\'";
List<Account> results = Database.query(query); Reference:
Why Safe: By escaping single quotes, it mitigates the risk of SOQL injection attacks that rely on manipulating string literals.
Option C:
String query = '%' + name + '%';
List<Account> results = [SELECT Id FROM Account WHERE Name LIKE :name]; Why Safe: Bind variables are the recommended way to include user input in SOQL queries safely, as they prevent injection by treating the input as a parameter rather than part of the query string.
Option B:
String query = 'SELECT Id FROM Account WHERE Name LIKE \'%' + name.noQuotes() + '%\'"; List<Account> results = Database.query(query); Why Unsafe: Without proper sanitization, the name variable could contain malicious SOQL code, leading to injection vulnerabilities.
Option D:
String query = 'SELECT Id FROM Account WHERE Name LIKE \'%' + name + '%\'"; List<Account> results = Database.query(query); Why Unsafe: Direct concatenation of user input without sanitization leaves the application vulnerable to SOQL injection attacks.
Conclusion:
Safe Options: A and C are safe from SOQL injection because they properly handle user input through escaping and bind variables, respectively.
Unsafe Options: B and D are unsafe as they do not adequately prevent SOQL injection.

質問 # 161
What will be the output in the debug log in the event of a QueryExeption during a call to the @query method in the following Example?



- A. Querying Accounts. Query Exception.
- B. Querying Accounts. Query Exception. Done
- C. Querying Accounts. Custom Exception Done.
- D. Querying Accounts. Custom Exception.

正解：B

質問 # 162
A lead developer creates a virtual class called "OrderRequest". Consider the following code snippet:
apex
Copy
public class CustomerOrder {
// code implementation
}
How can a developer use the OrderRequest class within the CustomerOrder class?

- A. @Implements (class="OrderRequest")public class CustomerOrder
- B. public class CustomerOrder extends OrderRequest
- C. extends (class="OrderRequest")public class CustomerOrder
- D. public class CustomerOrder implements Order

正解：B

解説：
Comprehensive and Detailed Explanation From Exact Extract:
To determine how a developer can use the OrderRequest class within the CustomerOrder class, we need to evaluate the options based on Apex syntax for inheritance, focusing on the fact that OrderRequest is a virtual class. Let's analyze the problem and each option systematically, referencing Salesforce's official Apex Developer Guide.
Understanding Virtual Classes and Inheritance in Apex:
* Virtual Class: In Apex, a virtual class allows other classes to extend it and inherit its methods and properties. It can also define methods that can be overridden by subclasses. The Apex Developer Guide states: "A virtual class can be extended by another class, allowing the subclass to inherit its methods and properties, and override virtual methods if needed" (Salesforce Apex Developer Guide, Classes).
* Inheritance: Apex supports single inheritance using the extends keyword, where a subclass inherits from a single parent class. The Apex Developer Guide notes: "Use the extends keyword to create a subclass that inherits from a parent class" (Salesforce Apex

Developer Guide, Inheritance).
* Virtual vs. Interface:
* A virtual class can contain method implementations and is extended using extends.
* An interface defines method signatures without implementations and is implemented using implements. The question specifies OrderRequest as a virtual class, not an interface.
* Requirement: The CustomerOrder class needs to use OrderRequest, which likely means inheriting its functionality, as OrderRequest is a virtual class.
Evaluating the Options:
* A. extends (class="OrderRequest")public class CustomerOrder
* Syntax: This option attempts to use extends followed by (class="OrderRequest") and then public class CustomerOrder.
* Analysis: The syntax is incorrect. In Apex, the extends keyword is followed directly by the name of the class to extend, with no parentheses or class= notation. The correct syntax would be public class CustomerOrder extends OrderRequest. The Apex Developer Guide specifies: "The extends keyword is followed by the name of the parent class, without additional attributes or parentheses" (Salesforce Apex Developer Guide, Inheritance). The (class="OrderRequest") syntax resembles annotations or markup, but it's not valid in Apex for inheritance.
* Conclusion: Incorrect due to invalid syntax.
* B. public class CustomerOrder implements Order
* Syntax: Uses the implements keyword to implement an interface named Order.
* Analysis: The implements keyword is used to implement an interface, not to extend a class. The question states that OrderRequest is a virtual class, not an interface, so implements is inappropriate. Additionally, the option references Order, not OrderRequest, which appears to be a mismatch with the class name provided in the question. The Apex Developer Guide clarifies: "The implements keyword is used for interfaces, while extends is used for classes, including virtual classes" (Salesforce Apex Developer Guide, Interfaces). Even if Order were intended to be OrderRequest, implements would still be incorrect for a virtual class.
* Conclusion: Incorrect, as implements is for interfaces, not virtual classes, and Order does not match OrderRequest.
* C. public class CustomerOrder extends OrderRequest
* Syntax: Declares CustomerOrder as a subclass of OrderRequest using the extends keyword.
* Analysis: This is the correct syntax for inheritance in Apex. Since OrderRequest is a virtual class, CustomerOrder can extend it to inherit its methods and properties. The extends keyword allows CustomerOrder to use (inherit) the functionality of OrderRequest. The Apex Developer Guide confirms: "A class can extend a virtual class using the extends keyword, inheriting its methods and properties" (Salesforce Apex Developer Guide, Inheritance). This matches the question's intent of "using" OrderRequest within CustomerOrder, which typically means inheritance when dealing with a virtual class.
* Conclusion: Correct, as it uses the proper extends keyword to inherit from the virtual OrderRequest class.
* D. @Implements (class="OrderRequest")public class CustomerOrder
* Syntax: Uses an @Implements annotation with (class="OrderRequest") before the class declaration.
* Analysis: Apex does not have an @Implements annotation. The implements keyword (without @) is used for interfaces, and annotations in Apex (e.g., @AuraEnabled, @TestVisible) do not use this format for inheritance or interface implementation. The Apex Developer Guide does not define @Implements as a valid annotation (Salesforce Apex Developer Guide, Annotations). The correct way to extend a virtual class is with extends, as in option C.
* Conclusion: Incorrect due to invalid syntax (@Implements is not a valid Apex annotation).
Why Option C is Correct:
Option C is correct because:
* It uses the extends keyword to properly declare CustomerOrder as a subclass of OrderRequest, which is a virtual class.
* This allows CustomerOrder to inherit and use the methods and properties defined in OrderRequest, fulfilling the requirement to "use" the OrderRequest class within CustomerOrder.
* The syntax aligns with Apex best practices for inheritance as outlined in the Salesforce Apex Developer Guide.
Example for Clarity:
Here's how the correct implementation (option C) would look:
apex
Copy
```
public virtual class OrderRequest {
public String getOrderDetails() {
return 'Order Details';
}
}
public class CustomerOrder extends OrderRequest {
public String getCustomerOrderInfo() {
// Inherit and use OrderRequest's method
String orderDetails = getOrderDetails();
return 'Customer Order: ' + orderDetails;
}
```

}
* CustomerOrder extends OrderRequest, inheriting getOrderDetails().
* The developer can use OrderRequest's functionality (e.g., call its methods) within CustomerOrder.
Handling Typos:
* The question's code snippet and options are mostly clear, but option B references Order instead of OrderRequest, which appears to be a typo. The analysis assumes the intended class is OrderRequest, as stated in the question stem.
* Option D has a typo in the class name: Customerorder should be CustomerOrder (capital O). This does not affect the analysis, as the syntax error (@Implements) is the primary issue.
References:
Salesforce Apex Developer Guide:
"Classes" section: Defines virtual classes and their ability to be extended.
"Inheritance" section: Explains the extends keyword for class inheritance.
"Interfaces" section: Clarifies the implements keyword for interfaces, distinguishing it from extends.
"Annotations" section: Confirms that @Implements is not a valid Apex annotation.(Available at: https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/) Platform Developer I Study Guide:
Section on "Developer Fundamentals": Covers Apex object-oriented programming concepts, including inheritance and virtual classes.(Available at: https://trailhead.salesforce.com/en/content/learn/modules/platform-developer-i-certification-study-guide)

質問 # 163
A developer has identified a method in an Apex class that performs resource intensive actions in memory by iterating over the result set of a SOQL statement on the account. The method also performs a DML statement to save the changes to the database.
Which two techniques should the developer implement as a best practice to ensure transaction control and avoid exceeding governor limits'* Choose 2 answers

- A. Use the System.limit class to monitor the current CPU governor limit consumption.
- B. Use partial DML statements to ensure only valid data is committed.
- C. Use the @ReadOnly annotation to bypass the number of rows returned by a SOQL.
- D. Use the Database.Savepoint method to enforce database integrity.

正解：B、D

解説：
* B: Using partial DML statements (e.g., Database.insert(list, false)) allows only valid records to be committed while identifying invalid ones, avoiding runtime errors.
* D: The Database.Savepoint method ensures transactional control by allowing rollback to a specific savepoint if an error occurs.
Why not other options?
* A: The @ReadOnly annotation is for read-only operations and would not allow DML operations.
* C: The System.Limit class helps monitor limits but does not directly control transaction behavior.
References:
* Best Practices for Apex Transactions

質問 # 164
.....

その、斜め方向へと突出した、無駄な努力はなんなんだ、情報は私たちのほとんどにとって少し深いですが、彼らは良い要約と計算機、そして新しい情報に関するニュースフィードを持っています、同時に、PDIテストトレントが毎日更新されるかどうかを確認する専門スタッフがいます。

実用的-100％合格率のPDI日本語版テキスト内容試験-試験の準備方法 PDI入門知識

PDI認定を取得することは多くの人にとって簡単ではないことがわかっていますが、良いニュースをお伝えできることを嬉しく思います、うちのSalesforceのPDI問題集を購入したら、私たちは一年間で無料更新サービスを提供することができます。

あなたの体験であなたは弊社のPDI試験問題は効率的で有効なものを認識します、あなたは最新のSalesforceのPDI試験トレーニング資料を手に入れることが保証します。

- PDI PDF □ PDI認定テキスト □ PDIテスト対策書 □ ▸ www.topexam.jp ◂で✔ PDI □✔□を検索し、無料でダウンロードしてくださいPDI技術内容
- PDI試験の準備方法｜有難いPDI日本語版テキスト内容試験｜一番優秀なPlatform Developer I (PDI)入門知識 □ □ www.goshiken.com □を開き、➤ PDI □を入力して、無料でダウンロードしてくださいPDI難易度
- 試験の準備方法-便利なPDI日本語版テキスト内容試験-100％合格率のPDI入門知識 ↕ ✔ www.jpexam.com □✔□には無料の▹ PDI ◃問題集がありますPDI試験解答
- 素晴らしいPDI日本語版テキスト内容 - 合格スムーズPDI入門知識 | 最新のPDIコンポーネント □ 最新「 PDI 」問題集ファイルは➤ www.goshiken.com □にて検索PDI練習問題
- PDI試験問題解説集 👆 PDI認定テキスト □ PDI試験問題解説集 □ ウェブサイト[www.jpshiken.com]から（ PDI ）を開いて検索し、無料でダウンロードしてくださいPDI PDF
- PDI資格参考書 □ PDI資格参考書 □ PDI資格参考書 □ 時間限定無料で使える[PDI]の試験問題は ➽ www.goshiken.com □サイトで検索PDI関連合格問題
- 効果的PDI｜便利なPDI日本語版テキスト内容試験｜試験の準備方法Platform Developer I (PDI)入門知識 □ ✔ www.japancert.com □✔□に移動し、⇛ PDI ⇚を検索して無料でダウンロードしてくださいPDI資格難易度
- PDI試験の準備方法｜最高のPDI日本語版テキスト内容試験｜有効的なPlatform Developer I (PDI)入門知識 □ 検索するだけで➤ www.goshiken.com □から□ PDI □を無料でダウンロードPDI復習攻略問題
- 試験の準備方法-高品質なPDI日本語版テキスト内容試験-一番優秀なPDI入門知識 □ ウェブサイト⇒ www.it-passports.com ⇐を開き、“PDI ”を検索して無料でダウンロードしてくださいPDI技術内容
- 最高のSalesforce PDI日本語版テキスト内容 - 権威のあるGoShiken - 資格試験におけるリーダーオファー □ ➤ www.goshiken.com □サイトで➤ PDI □の最新問題が使えるPDI試験参考書
- PDI PDF □ PDI試験問題解説集 □ PDI PDF □ ▹ www.goshiken.com ◃は、➠ PDI □を無料でダウンロードするのに最適なサイトですPDI無料模擬試験
- loanbookmark.com, freedirectorynow.com, bookmarksea.com, nelsondqob499702.bloguerosa.com, bookmarkfame.com, laylatquq288764.wikiadvocate.com, abelbmgt007919.blogdeazar.com, www.stes.tyc.edu.tw, tegantcya129018.wiki-racconti.com, mattiegnrk669664.blogpayz.com, Disposable vapes

BONUS！！！ CertShiken PDIダンプの一部を無料でダウンロード：https://drive.google.com/open?id=1UAXefFkzq9XYG8zdHQgZx79DWR6V46IO