Adobe AD0-E722 Practice Questions

Shared by Whitfield on 19-08-2026

For More Free Questions and Preparation Resources

Check the Links on Last Page

Question 1

Question Type: MultipleChoice

A custom cron job has been added to an Adobe Commerce system to collect data for several reports. Its crontab. xml configuration is as follows:

```
<config>
    <group id="default">
        <job name="gather_reporting_data" instance="Vendor\Reports\Cron\DataGatherer" method="execute">
            <schedule>0 0 * * *</schedule>
        </job>
    </group>
</config>
```

The job is data intensive and runs for between 20 and 30 minutes each night.

Within a few days of deployment, it is noticed that the sites sitemap. xml file has not been updated since the new job was added.

What should be done to fix this issue?

Options:

A- Change the schedule of the siten.aP_generate cron job to 30 0 * * *so that it runs after the gather_reporting_data job has completed.
B- Create a new cron group for the reporting job, specifying <use_separate_process>i</use_separate_process>
C- Break the data gathering job into a number of smaller jobs, so that each individual job runs for a maximum of 5 minutes

Answer:

B

Explanation:

The issue here is that the gather_reporting_data job is running for a long time and blocking the sitemap_generate job from running. The solution is to create a new cron group for the reporting job and specify <use_separate_process>i</use_separate_process> so that the reporting job runs in a separate process and does not block the sitemap_generate job.Reference: https://experienceleague.adobe.com/docs/commerce-cloud-service/user-guide/architecture/starter-architecture.html?lang=en#cron-groups-and-processes

Question 2

Question Type: MultipleChoice

A merchant asks for a new category attribute to allow uploading an additional mobile image against categories. The merchant utilizes the content staging and preview feature in Adobe Commerce and wants to schedule and review changes to this new mobile image field.

A developer creates the attribute via a data patch and adds it to view/adminhtml/ui_component/category_f orm. xml. The attribute appears against the category in the main form, but does not appear in the additional form when scheduled updates are made.

To change this attribute when scheduling new category updates, which additional action should the Architect ask the developer to take?

Options:

A- The attribute must have its apply_to field set to 'staging' in the data patch file.
B- The attribute must have <item- name=''allow_staging' xsi:type=''boolean''>true</item> set in the cjt.gopy_for-.xni file under the attributes config' section.
C- The attribute must also be added to view/adminhtml/ui_co-component/catalogstaging_category_update_form.xml.

Answer:

C

Explanation:

This action is necessary to make the attribute available for content staging and preview. According to the Adobe Commerce documentation, the catalogstaging_category_update_form.xml file defines the fields that are displayed in the Scheduled Changes section of the category form. The file extends the category_form.xml file and adds additional fields that are specific to content staging, such as start and end dates, campaign name, description, etc. To include a custom category attribute in the Scheduled Changes section, the attribute must also be declared in the catalogstaging_category_update_form.xml file with the same configuration as in the category_form.xml file.

Content staging | Adobe Commerce Developer Guide

Create a category attribute | Adobe Commerce Developer Guide

Question 3

Question Type: MultipleChoice

An Architect needs to review a custom product feed export module that a developer created for a merchant. During final testing before the solution is deployed, the product feed output is verified as correct. All unit and integration tests for code pass.

However, once the solution is deployed to production, the product price values in the feed are incorrect for several products. The products with incorrect data are all currently part of a content staging campaign where their prices have been reduced.

What did the developer do incorrectly that caused the feed output to be incorrect for products in the content staging campaign?

Options:

A- The developer retrieved product data directly from the database using the entity_id column rather than a collection or repository.
B- The developer forgot to use the getContentStagingValue method to retrieve the active campaign value of the product data.
C- The developer did not check for an active content staging campaign and emulates the campaign state when retrieving product data.

Answer:

C

Explanation:

Option C is correct because the developer did not check for an active content staging campaign and emulate the campaign state when retrieving product data. Content staging campaigns can modify the product data such as price, name, description, and so on, based on a schedule.To get the correct product data for a specific date and time, the developer needs to use the Magento\Staging\Model\VersionManager class to set the current version ID and emulate the campaign state1. Otherwise, the product data will be retrieved from the default store view without applying the campaign changes.

Option A is incorrect because retrieving product data directly from the database using the entity_id column is not necessarily wrong. It may not be the best practice, but it does not cause the feed output to be incorrect for products in the content staging campaign.The content staging campaigns are stored in separate tables with a version ID that links to the main product table2. The developer can still join these tables and query the product data based on the version ID and date.

Option B is incorrect because there is no such method as getContentStagingValue in Magento 2.

The developer cannot use this method to retrieve the active campaign value of the product data. The correct way to get the product data for a specific campaign is to use the Magento\Staging\Model\VersionManager class as mentioned above.

1: Content Staging | Adobe Commerce Developer Guide

2: Content Staging | Adobe Commerce Developer Guide

Question 4

Question Type: MultipleChoice

An Adobe Commerce Architect notices that the product price index takes too long to execute. The store is configured with multiple websites and dozens of customer groups.

Which two ways can the Architect shorten the full price index execution time? (Choose two.)

Options:

A- Set mage_ihdexer_threads_COUNT environment variable to enable parallel mode
B- Move catalog_Price_index indexer to another custom indexer group
C- Enable price index customer group merging for products without tier prices
D- Set Customer Share Customer Accounts Option to Global
E- Edit customer groups to exclude websites that they are not using

Answer:

A, C

Explanation:

The product price index can be optimized by using parallel mode and customer group merging. Parallel mode allows the indexer to run multiple threads simultaneously, which can speed up the indexing process. Customer group merging reduces the number of rows in the price index table by merging customer groups that have the same product prices. This can improve the performance of the price index queries and reduce the index size.Reference:Indexing optimization,Price index customer group merging

Question 5

Question Type: MultipleChoice

A developer needs to uninstall two custom modules as well as the database data and schemas. The developer uses the following command: bin/magento module:uninstall Vendor_SampleMinimal Vendor_SampleModifyContent

When the command is run from CLI, the developer fails to remove the database schema and data defined in the module Uninstall class. Which three requirements should the Architect recommend be checked to troubleshoot this issue? (Choose three.)

Options:

A- invoked uninstall() and uninstallschema are defined in the Uninstall class
B- invoked unlnstalK) method is implemented in the Uninstall class
C- bin/magento maintenance:enable command should be run in CLI before
D- --remove-data option is specified as an argument for the CLI command
E- --remove-schema and --remove-data options are specified as arguments for the CLI command
F- composer.json file is present and defines the module as a composer package

Answer:

B, D, F

Question 6

Question Type: MultipleChoice

An Adobe Commerce Architect designs a data flow that contains a new product type with its own custom pricing logic to meet a merchant requirement. Which three steps are required when adding a product type with custom pricing? (Select three.)

Options:

A- Content of the etc/product_types.xml file
B- Data patch to register the new product type
C- Hydrator for attributes belonging to the new product type
D- New price model extending \Magento\Catalog\Model\Product\Type\Price
E- Custom type model extended from the abstract Product Type model
F- A new class with custom pricing logic, extending the abstract Product model class

Answer:

A, D, E

Explanation:

To add a product type with custom pricing, the Architect needs to do the following steps:

Create a content of the etc/product_types.xml file that defines the new product type, its label, model, index priority, and price model.This file is used to register the new product type and its associated classes in Magento1.

Create a new price model that extends \Magento\Catalog\Model\Product\Type\Price and implements the custom pricing logic for the new product type.The price model is responsible for calculating the final price of the product based on various factors, such as special price, tier price, catalog price rules, etc2.

Create a custom type model that extends from the abstract Product Type model (\Magento\Catalog\Model\Product\Type\AbstractType) and overrides the methods related to the product type behavior, such as prepareForCart, getAssociatedProducts, etc.The type model defines how the product type interacts with other components, such as quote, order, cart, etc3.Reference:

How to add a new product type in Magento 2? (MageStackDay mystery question 1) - Magento Stack Exchange

Magento 2: How to create custom product types - BelVG Blog

Magento 2: How to create custom product types - BelVG Blog

Question 7

Question Type: MultipleChoice

An Adobe Commerce store owner sets up a custom customer attribute "my.attribute".

An Architect needs to display additional content on the home page, which should display only to Customers with "my.attribute" of a certain value and be the same content for all of them. The website is running Full Page Cache.

With simplicity in mind, which two steps should the Architect take to implement these requirements? (Choose two.)

Options:

A- Add a new context value of 'my_attribute' to Magento\Framework\App\Http\Context
B- Create a Customer Segment and use 'my.attribute' in the conditions
C- Add a custom block and a pHTML template with the content to the cmsjndexjndex.xml layout
D- Add a dynamic block with the content to the Home Page
E- Use customer-data JS library to retrieve 'my.attribute' value

Answer:

A, D

Explanation:

To display additional content on the home page based on a custom customer attribute, the Architect needs to do the following steps:

Add a new context value of ''my_attribute'' to Magento\Framework\App\Http\Context. This will allow the Full Page Cache to generate different versions of the page for customers with different values of ''my.attribute''. The context value can be set using a plugin on the Magento\Customer\Model\Context class.

Add a dynamic block with the content to the Home Page. A dynamic block is a type of content block that can be configured to display only to specific customer segments or conditions. The Architect can use the 'my.attribute' in the conditions of the dynamic block and assign it to the Home Page in the Content > Blocks section of the Admin Panel.Reference:

Private content | Magento 2 Developer Documentation

Dynamic Blocks | Adobe Commerce 2.3 User Guide - Magento

Question 8

Question Type: MultipleChoice

An Architect working on a headless Adobe Commerce project creates a new customer attribute named my_attribute. Based on the attribute value of the customer, the results of GraphQI queries are modified using a plugin. The frontend application is communicating with Adobe Commerce through Varnish by Fastly. which is already caching the queries that will be modified. The Adobe Commerce Fastly extension is installed, and no other modifications are made to the application.

Which steps should the Architect take to make sure the vcl_hash function of Varnish also considers the newly created attribute?

Options:

A- Create a new ClaSS inheriting from Magento\GraphQlCache\Model\CacheId\CacheIdFactorProvidftrInterface and returning the Value of my_attribute from the getFactorValue function and my_attribute from the getFactorName function. Then add this class through Dl to the idFactorProviders array of Magento\GraphQlCache\Model\CacheId\CacheIdCalculator.
B- Create a new class inheriting from Magento\Framework\GraphQi\Query\Resolver\identityinterfaca and returning the value of my_attribute from the getidentities function.
Then specify a ecache(cacheidentity: Path\\To\\identityclass) directive for each GraphQL query to include the newly created IdentityClass to each query that adds the cache tags for each customer.
C- Create a new class inheriting from Magento\customer\customerData\stctionSourceinterface and returning the value of my_attribute from the getSectionData function. Then add this ClaSS through the sectionSourceMap array Of Magento\Customer\CustomerData\SectionPoolInterface.

Answer:

A

Explanation:

To make sure the vcl_hash function of Varnish considers the newly created attribute, the Architect needs to do the following steps:

Create a new class that implements the Magento\GraphQlCache\Model\CacheId\CacheIdFactorProviderInterface interface. This interface defines two methods: getFactorName and getFactorValue. The getFactorName method should return the name of the attribute, in this case, my_attribute.The getFactorValue method should return the value of the attribute for the current customer, which can be obtained from the customer session or customer repository1.

Add this class to the idFactorProviders array of Magento\GraphQlCache\Model\CacheId\CacheIdCalculator through dependency injection. The CacheIdCalculator is responsible for generating a cache ID for each GraphQL request based on the factors provided by the idFactorProviders.By adding the new class to this array, the Architect ensures that the cache ID will include the value of my_attribute1.

The cache ID is then used by Varnish to hash and lookup the cached response for each request.By including my_attribute in the cache ID, the Architect ensures that Varnish will serve different responses based on the attribute value of the customer2.Reference:

Magento_GraphQlCache module | Magento 2 Developer Documentation

Varnish caching | Adobe Commerce 2.4 User Guide - Magento

Question 9

Question Type: MultipleChoice

A client has multiple warehouses where orders can be fulfilled. The cost of shipping goods from each warehouse varies by day, due to the number of workers available. The Architect needs to make sure that when an order is shipped, it is shipped from the lowest cost warehouse that is open.

How should this functionality be implemented?

Options:

A- Create a new class as a preference for Magento\inventoryShipping\piugin\Sales\shipment\AssignSourceCodeToShipmentPlugin to set the lowest-cost warehouse on a shipment.
B- Create a new class implementing Magento\invtntorysourceSelectionApi\Modei\sourceSelectioninterfacece. which returns open warehouses sorted by cost.
C- Create an after plugin On Hagento\InventoryDistanceBasedSourceSelection\Hodel\Algorithms\DistanceBasedAlgorithto sort to Warehouse sources by cost

Answer:

B

Explanation:

According to the Adobe Commerce documentation, the Source Selection Interface is the main interface for implementing custom source selection algorithms. The interface defines a method called execute(), which takes a list of items to be shipped and a stock ID as parameters, and returns a SourceSelectionResultInterface object, which contains the recommended sources and quantities for each item. The Architect can create a new class that implements this interface and provides the logic for finding the lowest-cost warehouse that is open for each item. The Architect can then register the new class as an option for the source selection algorithm in the di.xml file of the custom module.

Source Selection Algorithm | Adobe Commerce Developer Guide

Source Selection Interface | Adobe Commerce Developer Guide

Question 10

Question Type: MultipleChoice

While reviewing a newly developed pull request that refactors multiple custom payment methods, the Architect notices multiple classes that depend on \Magento\Framework\Encryption\EncryptorInterface to decrypt credentials for sensitive dat

a. The code that is commonly repeated is as follows:

```
namespace Vendor\PaymentModule\Gateway\Config;

class Config extends \Magento\Payment\Gateway\Config\Config
{
    ...
    public function __construct(
        ...
        ScopeConfigInterface $scopeConfig,
        EncryptorInterface $encryptor,
        ...
    ) {
        parent::__construct($scopeConfig, $methodCode, $pathPattern);
        $this->scopeConfig = $scopeConfig;
        $this->encryptor = $encryptor;
    }

    public function getUserSecret(): string
    {
        return $this->encryptor->decrypt(
            $this->scopeConfig->getValue('payment/method_code/user_secret')
        );
    }
```

The Architect needs to recommend an optimal solution to avoid redundant dependency and duplicate code among the methods. Which solution should the Architect recommend?

Options:

A- Create a common config service class vndor\Pay-ient\Gatway\conf ig\conf ig under Vendor.Payment and use it as a parent class for all of the
B- Replace all Vendor\PaymentModule\Gateway\Config\Config ClaSSeS With virtualType Of Magento\Payiaent\Gateway\Conf ig\Conf ig and Set <user_secret backend_Model='Magento\Config\Model\Config\Backend\Encrypted' /> Under config.xml
C- Add a plugin after the getvalue method of $scopeConfig, remove the $encryptor from dependency and use it in the plugin to decrypt the value if the config name is user.secret'

Answer:

B

Explanation:

The Architect should recommend replacing all Vendor\PaymentModule\Gateway\Config\Config Classes with virtualType of Magento\Payment\Gateway\Config\Config and setting <user_secret backend_Model=''Magento\Config\Model\Config\Backend\Encrypted'' /> under config.xml. This will avoid redundant dependency and duplicate code among the methods. The virtualType of Magento\Payment\Gateway\Config\Config will inherit the functionality of the base class and allow the customization of the constructor arguments, such as the pathPattern and valueHandlerPool.The backend_Model attribute of the user_secret field will specify that the value of this field should be encrypted and decrypted by the Magento\Config\Model\Config\Backend\Encrypted class, which implements the \Magento\Framework\App\Config\ValueInterface interface and uses the \Magento\Framework\Encryption\EncryptorInterface internally12.This way, the payment modules do not need to depend on the \Magento\Framework\Encryption\EncryptorInterface or the \Magento\Framework\App\Config\ScopeConfigInterface directly, and can use the getValue method of the Magento\Payment\Gateway\Config\Config class to get the decrypted value of the user_secret field3.Reference:

How to encrypt system configuration fields in Magento 2 - Mageplaza

Magento 2: How to Encrypt/Decrypt System Configuration Fields - Webkul Blog

Magento 2: How to create custom payment method - BelVG Blog