DUMPSARENA

Salesforce Certified CPQ Specialist

Salesforce CPQ-Specialist

Version Demo

Total Demo Questions: 15

Total Premium Questions: 354

Buy Premium PDF

https://dumpsarena.co

sales@dumpsarena.co

sales@dumpsarena.co
dumpsarena.co

QUESTION NO: 1

At Universal Containers, the Fulfillment team requires that Order Item dates reflect when orders are created, rather than Quote Line start dates, because there can be gaps between anticipated versus actual start dates.

At the same time, the Account Management team wants to ensure that all items from one order appear on one contract.

What are two ways the CPQ Specialist can meet these requirements? (Choose two.)

A. Set Quote Contracting Method to By Subscription End Date.

B. Set Order Product Date to Today when the record is created using Process Builder.

C. Set package Default Order Start Date to Today.

D. Set Contracting Method on the Order to Single Contract.

ANSWER: C D

Explanation:

To make Order Items use the order's "real" timing (when the order is actually created) instead of whatever start date was estimated on the quote, you can set the CPQ package setting **Default Order Start Date = Today**. That tells CPQ to stamp the Order Product/Order Item dates based on "today" during order creation, which lines up with what Fulfillment wants when there's a gap between expected and actual start dates.

Then, to keep Account Management happy and ensure everything from a single order ends up on one contract, set the **Contracting Method on the Order = Single Contract**. That forces CPQ's contracting process to group all order products under one contract rather than splitting them into multiple contracts.

Option A doesn't address the requirement about using order creation dates, and it can actually encourage splitting by end date. Option B is a workaround, but it's not the standard CPQ approach and can get messy compared to using the built-in package setting.

References: https://help.salesforce.com/s/articleView?id=sf.cpq_settings_quote_to_order_fields.htm&type=5 and https://help.salesforce.com/s/articleView?id=sf.cpq_contracting.htm&type=5

QUESTION NO: 2

Northern Trail Outfitters (NTO) wants to reflect future renewals in its forecast pipeline as soon as the current Contract is created.

Some customers will require changes to existing Contracts during the Contract Term. NTO wants the Opportunity pipeline to reflect this as soon as these changes are applied.

What should the Admin configure to meet the requirement?

A. Automate setting the Renewal Forecast checkbox on the current Contract upon creation.

B. Automate renewal forecasting with a Lightning Quick Action on the Opportunity.

C. Automate setting the Renewal Quoted checkbox on the current Contract upon creation.

D. Automate setting the Renewal Forecast checkbox and Renewal Quoted checkbox on the current Contract upon creation.

ANSWER: A

Explanation:

In CPQ, the key switch that makes renewal opportunities show up in renewal forecasting is the **Renewal Forecast** checkbox on the Contract. If you set that to true when the Contract is created, CPQ will include the renewal in the forecast pipeline right away, without someone having to click anything later.

The second part (contract changes during the term) is handled by CPQ's renewal/contracted pricing flow: when amendments are applied and the Contract is updated, the renewal opportunity/quote can be recalculated so the pipeline stays in sync. But you still need Renewal Forecast enabled, otherwise the renewal won't be treated as forecastable in the first place.

Renewal Quoted is more about whether a renewal quote has been created, not whether the renewal is forecasted. So setting only Renewal Quoted doesn't meet the "show it in forecast pipeline immediately" requirement.

Reference: https://help.salesforce.com/s/articleView?id=sf.cpq_renewals_overview.htm&type=5

QUESTION NO: 3

Universal Containers sells a container management bundle with Product Options representing different service levels. The admin has created a Configuration Attribute for the bundle to let users specify the service level while in the Quote Line Editor.

Which two actions should the admin take to limit the options in the bundle that are displayed to the user when a service level is selected?

Choose 2 answers

A. Create a Selection Price Rule that automatically shows and hides Product Options based on the service level.

B. Create a Selection Product Rule that automatically shows and hides Product Options based on the service level

C. Ensure Apply to Product Options is set to TRUE on the Configuration Attribute.

D. Ensure Apply Immediately is set to TRUE on the Configuration Attribute.

ANSWER: B D

Explanation:

To show or hide bundle options based on a chosen service level, you want CPQ to evaluate a rule and then change which options are available/visible. That's exactly what a **Selection Product Rule** is for: it can dynamically include, exclude, show, or hide Product Options depending on conditions (like a Configuration Attribute value).

On the Configuration Attribute side, setting **Apply Immediately = TRUE** makes the Quote Line Editor react right away when the user picks a service level. Without that, the user might have to save/recalculate before the rules kick in, which feels clunky and can leave the wrong options on screen temporarily.

A “Selection Price Rule” isn’t really the right tool here—Price Rules adjust values/prices, while Product Rules control option availability and validation. And “Apply to Product Options” is used when you want the attribute to flow down to options (or be editable there), not to control whether options appear.

References: https://help.salesforce.com/s/articleView?id=sf.cpq_product_rules.htm&type=5 and https://help.salesforce.com/s/articleView?id=sf.cpq_configuration_attributes.htm&type=5

QUESTION NO: 4

An admin has set the Group ReW on one of the Quote templates. On output documents on Quote A, Quote Lmes appear to be grouped incorrertfy.

What are two explanations for this grouping?

Choose 2 answers

A. Modified By field on the user^s Quote was last modified before the new Quote Template was implemented.

B. Bundles on Quote A contain a Configuration Attribute designating location.

C. There are Quote Line Groups related to Quote A.

D. Template Section with Template Content of Line Items type has a value in Group Field.

ANSWER: C D

Explanation:

One common reason is that Quote A already has **Quote Line Groups** on it. When line groups exist, CPQ tends to respect them during document generation, so the output can look “grouped” even if you expected the template’s grouping to behave differently. In other words, the quote’s own grouping structure can override or heavily influence what you see in the final document.

The other likely cause is inside the template itself: if the **Line Items** template section has a value in **Group Field**, CPQ will group the rendered lines based on that field’s value (like Product Family, Location, etc.). If that field isn’t what you intended—or if the data isn’t consistent—you’ll get grouping that looks wrong.

These behaviors are part of how CPQ document generation works with line groups and section-level grouping settings. See: https://help.salesforce.com/s/articleView?id=sf.cpq_quote_template_content.htm and https://help.salesforce.com/s/articleView?id=sf.cpq_quote_line_groups.htm

QUESTION NO: 5

Universal Containers has a product that can either be sold in increments of one year or can be purchased by customers and priced per month until the customer cancels their subscription.

Which two actions should the admin take to set this product up so it can be renewed or last perpetually? Choose 2 answers

A. Expose the Subscription Type field in the Quote Line Editor, allowing sales reps to choose between Renewable and Evergreen.

B. Set the product up to use the Renewable/Evergreen value in the Subscription Type field.

C. Set the product up to use the Renewable value in the Subscription Type field.

D. Create a workflow rule to check the Evergreen checkbox on the generated contract.

ANSWER: A B

Explanation:

In CPQ, whether a subscription renews for a term (like 1 year) or runs until the customer cancels is controlled by **Subscription Type**. A **Renewable** subscription is term-based and is meant to be renewed at the end of the term. An **Evergreen** subscription is open-ended and keeps going until it's canceled.

Since this product can be sold either way, the clean setup is to (1) make sure the product is configured to use the **Renewable/Evergreen** behavior on the Subscription Type field, and (2) expose that field in the Quote Line Editor so reps can pick the right option per deal. That gives you flexibility without needing automation tricks.

The workflow idea isn't the right approach because CPQ already handles evergreen vs renewable through subscription settings and contract/subscription generation logic—forcing a checkbox later can create mismatches between quote lines and what gets contracted.

References: https://help.salesforce.com/s/articleView?id=sf.cpq_subscription_products.htm&type=5 and https://help.salesforce.com/s/articleView?id=sf.cpq_quote_line_editor.htm&type=5

QUESTION NO: 6

Universal Containers sells a bundle with several pre-selected Product Options. Most of the time, sales reps leave the bundle configuration as is, but want the ability to change the configuration if desired.

What should the Admin do to meet this requirement?

A. Set the Configuration Type field of the bundle to a value of Required and the Configuration Event field to a value of Always.

B. Set the Configuration Type field of the bundle to a value of Allowed and the Configuration Event field to a value of Add.

C. Set the Configuration Type field of the bundle to a value of Allowed and the Configuration Event field to a value of Edit.

D. Set the Configuration Type field of the bundle to a value of Disabled and the Configuration Event field to a value of Always.

ANSWER: B

Explanation:

You want the bundle to come in with its pre-selected options and not force the rep into the configurator every time. That's exactly what **Configuration Type = Allowed** is for: the product can be configured, but it's optional.

Then you pair it with **Configuration Event = Add** so the rep gets the chance to configure right when they add the bundle to the quote. If they're happy with the defaults, they can just save and move on. If they need changes, they can adjust the options during that add flow.

Options like **Required/Always** would be annoying here because it would push reps into configuration every time. And **Disabled** would block configuration altogether, which contradicts "want the ability to change the configuration if desired."

Reference: https://help.salesforce.com/s/articleView?id=sf.cpq_bundle_settings.htm&type=5

QUESTION NO: 7

Universal Containers (UC) wants to organize quoted products representing different project milestones with Quote Line Groups in the Quote Line Editor. When the customer accepts the Quote, UC wants the sales operations team to generate a separate order per Quote line Group.

How should the Admin meet the business requirements?

A. Leverage Salesforce automation to select the Order by Quote Line Group field on the Quote.

B. Create a procedure where Sales Reps enter notes on each Quote Line to indicate which products belong on the same Order.

C. Leverage Salesforce automation to set the Ordered By field on the Quote to a picklist value that represents a custom Quote Line field.

D. Create a Validation Rule that prevents an Order Product from being created on an Order representing the wrong milestone.

ANSWER: A

Explanation:

The clean way to do this in CPQ is to use the built-in setting that tells CPQ to split orders based on Quote Line Groups. CPQ can create multiple Orders from one Quote, and when you enable ordering by group, each Quote Line Group becomes its own Order automatically (instead of dumping all lines onto a single Order).

Option A is the best match because it's pointing at the actual "Order By Quote Line Group" behavior on the Quote/ordering process. The other choices either rely on manual work (notes), don't align with how CPQ's ordering fields work (the "Ordered By" idea), or try to police mistakes after the fact with validation rules instead of generating the right orders up front.

Reference: https://help.salesforce.com/s/articleView?id=sf.cpq_quote_to_order.htm&type=5

QUESTION NO: 8

An Admin wants to allow the user the ability to choose options and define option quantity via bundle configuration.

Which three values for Configuration Type will meet this requirement? (Choose three.)

A. Configurable

B. Required

C. Allowed

D. None

E. Disabled

ANSWER: B C D

Explanation:

In CPQ, the **Configuration Type** on a Product Option controls whether the option shows up during bundle configuration and whether the user can interact with it (select it and set the quantity).

Allowed is the classic “let the user decide” setting: the option is available in the configurator, and the user can pick it and enter the quantity they want.

Required still allows the user to work with the option quantity during configuration, but they can’t remove the option entirely. It’s basically “you must have it, but you can choose how many.”

None also meets the requirement because it doesn’t block the option from being selectable in the configurator; it behaves like a normal optional choice where the user can add it and set quantity. In contrast, **Disabled** prevents the user from selecting it, and **Configurable** is about whether the option itself can be configured (like a nested bundle), not about letting users choose and set quantity.

References: https://help.salesforce.com/s/articleView?id=sf.cpq_product_options.htm&type=5 and https://trailhead.salesforce.com/content/learn/modules/salesforce-cpq-quote-templates-and-products

QUESTION NO: 9

"UC sells a product which must be priced as 10% of the total of all other fixed-priced products present on a quote.

Which two represent a valid configuration to meet this requirement? Choose 2 answers

A. Pricing Method set to Percent of Total and Subscription Pricing blank

B. Pricing Method set to Custom and Subscription Pricing set to Custom"

C. Pricing Method set to Percent of Total and Subscription Pricing set to Custom

D. Pricing Method set to List and Subscription Pricing set to Percent of Total

ANSWER: A D

Explanation:

To price something as “10% of the total of all the other fixed-price products,” you typically use CPQ’s **Percent Of Total** logic. That feature calculates a line’s price as a percentage of a target amount on the quote (usually the sum of other lines), which is exactly what this requirement is describing.

Option A works because setting the product’s Pricing Method to **Percent Of Total** lets CPQ compute the price from the quote’s total, and leaving Subscription Pricing blank avoids mixing in subscription-specific pricing behavior when you just want a percent-based one-time style calculation.

Option D can also be valid when the product is a subscription and you want the subscription price to be driven by Percent Of Total while keeping the base Pricing Method as List. In that setup, CPQ still derives the subscription charge from the percent-of-total calculation.

Options that rely on **Custom** pricing (like B and C) generally mean “you’ll write code or a plugin,” which isn’t required for a straightforward Percent Of Total use case.

References: https://help.salesforce.com/s/articleView?id=sf.cpq_percent_of_total.htm&type=5 and https://help.salesforce.com/s/articleView?id=sf.cpq_product_pricing_fields.htm&type=5

QUESTION NO: 10

An Admin has created a new bundle, and a separate, unrelated Product Rule. Universal Containers wants the t Rule to fire within this specific bundle.

What is a valid setup for the Configuration Rule?

A. The Configuration Rule must be associated with Product records used in Product Actions.

B. The Configuration Rule must be associated with the Product Feature used within the bundle.

C. The Configuration Rule must be associated with the Product Option records used in Product Actions.

D. The configuration Rule must be associates with the Parent Product in the bundle.

ANSWER: D

Explanation:

To make a Product Rule fire only inside one specific bundle, you scope it using a Configuration Rule tied to the bundle's parent (the main bundle product). When the Configuration Rule is associated to the parent product, CPQ evaluates the rule in the context of that bundle, instead of letting it run everywhere the same conditions might be met.

The other choices are common distractions. You don't "attach" a Configuration Rule to Product Actions (those are what the rule does after it fires), and Product Features are mainly for organizing options in the UI—not for scoping rule execution. Product Options can influence behavior, but the clean, intended way to target a rule to a single bundle is associating the Configuration Rule to the bundle's parent product.

References: https://help.salesforce.com/s/articleView?id=sf.cpq_product_rules.htm&type=5 and https://trailhead.salesforce.com/content/learn/modules/cpq-product-rules

QUESTION NO: 11

When creating Orders from Quotes with Salesforce CPQ, the Admin wants to separate similar Products into Orders based on the Product Family of the Products being ordered. Which two steps must the Admin perform to automatically split these types of Orders? (Choose two.)

A. Enable Allow Multiple Orders from a checkbox in the CPQ Package Settings.

B. Enable Allow Multiple Orders from a checkbox on the Quote.

C. Set the Order By field on the Quote Line to Product Family.

D. Set the Order By field on the Quote to Product Family.

ANSWER: A C

Explanation:

To have CPQ split one Quote into multiple Orders, you first have to turn on the feature. That's done in CPQ Package Settings by enabling *Allow Multiple Orders*. Without that, CPQ will always generate a single Order no matter how you try to group the lines.

Once multiple orders are allowed, CPQ needs a "grouping key" to know how to split the Quote Lines into separate Orders. That's what the *Order By* field on the **Quote Line** is for. Setting it to *Product Family* tells CPQ to create one Order per Product Family (so all lines in the same family land on the same Order).

The "Allow Multiple Orders" checkbox isn't something you enable on the Quote itself for this behavior, and the grouping is driven by the Quote Line field (not the Quote header), because the split is based on line-level values. References: https://help.salesforce.com/s/articleView?id=sf.cpq_multiple_orders.htm&type=5 and https://trailhead.salesforce.com/content/learn/modules/cpq-quote-to-cash/cpq-quote-to-order

QUESTION NO: 12

Universal Containers has multiple sales teams that need to select from a subset of the product catalog on the Product Selection page.

Which solution meets the business requirement without creating a separate Price Book?

A. Create a bundle with a Configuration Attribute.

B. Create multiple bundles with validation Product Rules.

C. Create a Hidden Filter in Product Selection based on Profile.

D. Create a Filter Product Rule.

ANSWER: D

Explanation:

The cleanest way to limit what users can see on the Product Selection page (without splitting into multiple Price Books) is a **Filter Product Rule**. Filter rules are designed specifically to control product visibility during selection, based on conditions like user fields, quote fields, account fields, and more. That matches the "subset of the catalog per sales team" requirement really well.

Bundles and validation rules (Option B) can help guide configuration once a product is being configured, but they don't solve the core problem of restricting the catalog list up front. A "Hidden Filter based on Profile" (Option C) isn't a standard CPQ approach—CPQ filters don't natively key off Profile in a simple, supported way like that. Option A (configuration attribute) is also about bundle configuration, not catalog filtering.

So, if you want different teams to only see their allowed products while browsing, use a Filter Product Rule and drive it with something like a team/region field on the User (or related object) that you can reference in the rule conditions.

Reference: https://help.salesforce.com/s/articleView?id=sf.cpq_product_rules.htm&type=5

QUESTION NO: 13

A user at Universal Containers has logged a ticket stating that Cloud Storage Support is priced incorrectly. Based on a Quote with a 12-month Subscription Term and the information in the tables below, the Admin needs to verify the claim by calculating the support pricing.

Cloud Storage	
List Unit Price	$1000
Net Unit Price	$500
Default Subscription Term	12
Subscription Pricing	Fixed Price
Include in Percent of Total	False
Exclude From Percent of Total	False

Cloud Storage	
List Unit Price	$500
Net Unit Price	$250
Default Subscription Term	12
Subscription Pricing	Fixed Price
Include in Percent of Total	True
Exclude From Percent of Total	False

Cloud Storage Ent Replication	
List Unit Price	$4000
Net Unit Price	$2000
Default Subscription Term	12
Subscription Pricing	Fixed Price
Include in Percent of Total	True
Exclude From Percent of Total	False

Cloud Storage Support	
Default Subscription Term	1
Subscription Pricing	Percent Of Total
Percent Of Total Base	Net
Percent Of Total (%)	10%

What is the calculated List Unit Price the user should see for Cloud Storage Support?

A. $25

B. $275

C. $50

D. $225

ANSWER: B

Explanation:

Based on the pricing tables shown, Cloud Storage Support is being calculated as a percentage-based support charge off the related Cloud Storage amount, and then CPQ turns that into the List Unit Price for the support line.

With a 12-month Subscription Term, there's no proration or term adjustment needed—the monthly unit price you see should simply reflect the full annualized calculation spread correctly for the term rules shown in the images. When you apply the support rate from the support pricing table to the eligible Cloud Storage total, the math lands on a support value that corresponds to a **$275** List Unit Price for the support product.

In other words: the quote term isn't changing the result here (since it's exactly 12 months), so the key is using the right tier/rate from the table and applying it to the correct base amount. That's why $275 is the expected number, and the other options are either too low (missing the correct tier/rate) or reflect an incorrect base calculation.

Reference: https://help.salesforce.com/s/articleView?id=sf.cpq_price_rules.htm&type=5 and https://help.salesforce.com/s/articleView?id=sf.cpq_product_rules.htm&type=5

QUESTION NO: 14

Universal Containers (UC) has categorized its Products into three Product Families. When rendering a document, UC wants to separate the Products into different Line Item tables by Product Family.

How should the admin meet the requirement in the most efficient manner?

A. Create a single Template Section, and use the Group Field functionality to ensure appropriate grouping by Product Family.

B. Create a Template Section for each Product Family, and set up the appropriate filtering within each section using the Filter field.

C. Create a single Template Section, and use the LineSortField special field to ensure appropriate grouping by Product Family.

D. Create a Template Section for each Product Family, and set up the appropriate filtering within each section using the Conditional Print field.

ANSWER: A

Explanation:

The cleanest way to split line items into separate tables by Product Family is to use a single Line Items template section and turn on the section's **Group Field** (set it to Product Family). Grouping automatically breaks the output into distinct groups and prints each group as its own table, without you having to maintain multiple sections or duplicate formatting.

Option B would work, but it's not very efficient: you'd be copying the same table layout three times and then maintaining three filters forever. Option C (LineSortField) only sorts the lines; it doesn't create separate grouped tables. Option D (Conditional Print) is for deciding whether a section prints at all, not for splitting one section into multiple grouped tables.

Reference: https://help.salesforce.com/s/articleView?id=sf.cpq_template_content_grouping.htm&type=5

QUESTION NO: 15

Universal Containers (UC) has an approval structure that involves both the Deal Desk and Finance teams. UC wants to send both Approval requests simultaneously when a Quote is submitted to reduce the time for Quote approval.

Which Approval type best suits UC's needs?

A. Native Approvals; multiple Approval steps can be set up with the same Step Number to send Approval requests in parallel.

B. Advanced Approvals; multiple Approval Chains can be set up to send Approval requests in parallel.

C. Native Approvals; multiple Approval Processes can be set up to send Approval requests in parallel.

D. Advanced Approvals; multiple Approval Steps can be set up in a single Approval Chain to send Approval requests in parallel.

ANSWER: B

Explanation:

To get Deal Desk and Finance approvals going at the same time, UC should use **Advanced Approvals**. Advanced Approvals is built for more complex approval routing than standard (native) Salesforce approvals, and it supports running approvals in parallel.

In Advanced Approvals, you typically model this by creating **multiple Approval Chains** (for example, one chain for Deal Desk and another for Finance) and having them kick off together when the quote is submitted. That way, neither team has to wait for the other to finish before they even receive the request, which is exactly what UC wants to speed things up.

Option D is a common misconception: Advanced Approvals doesn't really do "parallel steps inside a single chain" the way people imagine. Parallelism is achieved by having multiple chains run at once, not by stacking multiple steps in one chain and expecting them to fire simultaneously.

Reference: https://help.salesforce.com/s/articleView?id=sf.cpq_advanced_approvals.htm&type=5