DUMPSBOSS.

# Microsoft Power Platform Developer

## Microsoft PL-400

Version Demo

Total Demo Questions: 43

Total Premium Questions: 435

Buy Premium PDF

https://dumpsboss.co

support@dumpsboss.co

support@dumpsboss.co
dumpsboss.co

# Topic Break Down

| Topic | No. of Questions |
|---|---|
| Topic 1, Perform technical design | 8 |
| Topic 2, Configure Common Data Service | 61 |
| Topic 3, Create and configure Power Apps | 48 |
| Topic 4, Extend the user experience | 72 |
| Topic 5, Extend the platform | 52 |
| Topic 6, Develop integrations | 134 |
| Topic 7, Develop Application Lifecycle Management | 32 |
| Topic 8, Case Study 1 | 3 |
| Topic 9, Case Study 2 | 4 |
| Topic 10, Case Study 3 | 3 |
| Topic 11, Case Study 4 | 2 |
| Topic 12, Case Study 5 | 4 |
| Topic 13, Case Study 6 | 2 |
| Topic 14, Case Study 7 | 2 |
| Topic 15, Case Study 8 | 3 |
| Topic 16, Case Study 9 | 3 |
| Topic 17, Case Study 10 | 2 |
| Total | 435 |

QUESTION NO: 1

You need to modify the Power Automate flow to resolve CustomerC's issue.

What should you do?

**A.** Add a configure run that is set to is successful.

**B.** Add a data operation that specifies the false conditions.

**C.** Add a condition containing approval hierarchy.

**D.** Add a timeout setting to the approval flow.

**ANSWER: D**

**Explanation:**

Add a timeout setting to the approval flow. The reported issue is that the request has remained unanswered for one week, so the flow needs a defined limit on how long it waits for the approval or response. Power Automate approval actions can be configured with a timeout using an ISO 8601 duration, such as `P7D` for seven days. When that period expires without a response, the flow can proceed through its timeout path, where it can notify the parts department, escalate the request, or otherwise handle the overdue item according to the organization's process. This prevents an approval instance from waiting indefinitely and provides a controlled way to address unanswered customer requests. A timeout is particularly appropriate when the business requirement is based on elapsed time rather than on a specific approval-routing structure. Microsoft documents that action settings include a configurable timeout and describes the ISO 8601 duration format used for this setting. See Power Automate action limits and configuration and Modern approvals in Power Automate.

**QUESTION NO: 2**

You need to create the eligibility assessment app. Which command should you run?

**A.** pac application install

**B.** pac solution init

**C.** pac plugin init

**D.** pac pcf init

**ANSWER: B**

**Explanation:**

`pac solution init` is the appropriate command because it initializes a Power Platform solution project locally. A solution provides the container used to organize, version, and move the app's Dataverse components and customizations through development environments. The command creates the basic solution-project structure and configuration files, including the solution definition, so developers can add components and use source control and application lifecycle management practices. This is the normal starting point when developing a business application such as an eligibility assessment app that is delivered as a managed or unmanaged Dataverse solution. After initialization, the solution project can be populated with the required app components and then synchronized with a target environment through the Power Platform CLI. Microsoft documents `pac solution init` as the command for creating a new solution project, with parameters for specifying the solution's publisher and name. This establishes a reusable, deployable package boundary for the application and its related customizations. See the pac solution init command reference and Microsoft's guidance on solutions and application lifecycle management.

**QUESTION NO: 3 - (SIMULATION)**

You are researching integrations with several external systems.

Each integration has different requirements.

You need to determine which data sources to use to meet each requirement.

What should you use? To answer, drag the appropriate data sources to the correct requirements. Each data source may be used once, more than one, or not at all. You may need to drag the split bar between panes or scroll to view content.

NOTE: Each correct selection is worth one point.

**ANSWER: See the explanation for the answer**

**Explanation:**

Drag the data sources as follows:

**Requirement A: Virtual Entity (virtual table)**
**Requirement B: Custom Connector**
**Requirement C: Virtual Entity (virtual table)**

A virtual table is appropriate when external data must be surfaced in Dataverse/model-driven app clients while retaining the external system's record identity. Use a custom connector for the requirement that needs direct interaction with the external system rather than the virtual-table scenario.

Reference: https://learn.microsoft.com/en-us/power-apps/maker/data-platform/virtual-entities

## QUESTION NO: 4

A company has two development instances, two test instances, two staging instances, and one production instance.

The test team reports connection issues with the test and staging instances.

You need to identify which of the instances the testing team currently has access.

Which two URLs can you use? Each correct answer presents a complete solution.

NOTE: Each correct selection is worth one point.

**A.** https://globaldisco.crm.dynamics.com/api/discovery/v9.1/instances

**B.** https://myorg.api.crm.dynamics.com/api/data/v9.1/

**C.** https://dev.crm.dynamics.com/api/discovery/v9.1/instances

**D.** https://disco.crm.dynamics.com/api/discovery/v9.1/

**E.** https://dev.crm.dynamics.com/api/discovery/v9.1/instances(UniqueName='myorg')

**ANSWER: C E**

**Explanation:**

The Discovery Web API exposes organization (instance) information through the `Instances` entity set. `https://dev.crm.dynamics.com/api/discovery/v9.1/instances` queries that entity set without an instance key, so the response contains the instances to which the authenticated test-team user has access. This lets the administrator determine whether the relevant test and staging instances appear in that user's accessible-instance list.

`https://dev.crm.dynamics.com/api/discovery/v9.1/instances(UniqueName='myorg')` uses the alternate key, `UniqueName`, to retrieve information for one known instance. It is therefore a complete solution when validating whether the user can discover a particular test or staging organization; the returned instance metadata confirms that the organization is available to the authenticated identity. Instance discovery is performed using the Discovery service rather than the environment's normal Web API data endpoint, because the latter is used to access Dataverse table data after connecting to a specific organization.

Microsoft documents querying the Discovery Web API `Instances` entity set, including retrieving all available instances and retrieving a specific instance by unique name: Discover the URL for an organization by using the Web API. See also Use the Microsoft Dataverse Web API.

## QUESTION NO: 5

You are developing a Power Apps app to manage records in the Account table in Microsoft Dataverse. You must configure a Web API request to retrieve changes from the table. You need to configure the preference header for the API request. What should you include in the request header?

**A.** odata.nextLink

**B.** odata-context

**C.** odata.deltaLink

**D.** odata.track-changes

## ANSWER: D

**Explanation:**

`odata.track-changes` is the required preference for a Dataverse Web API request that must begin tracking changes to an entity set. Send it in the HTTP `Prefer` request header, for example: `Prefer: odata.track-changes`. Dataverse then returns the initial set of matching records and provides an `@odata.deltaLink` annotation in the response when applicable. The application stores that delta link and subsequently issues a GET request to it to retrieve only records that have changed since the earlier query, rather than retrieving the entire Account table again. This enables efficient synchronization and incremental refresh scenarios. Change tracking must also be enabled for the relevant Dataverse table before this capability can be used. Microsoft documents that the `odata.track-changes` preference requests change tracking for the query, while the delta link is a response value used later to request changes. See Retrieve data about changes to entity data and Dataverse Web API Prefer headers.

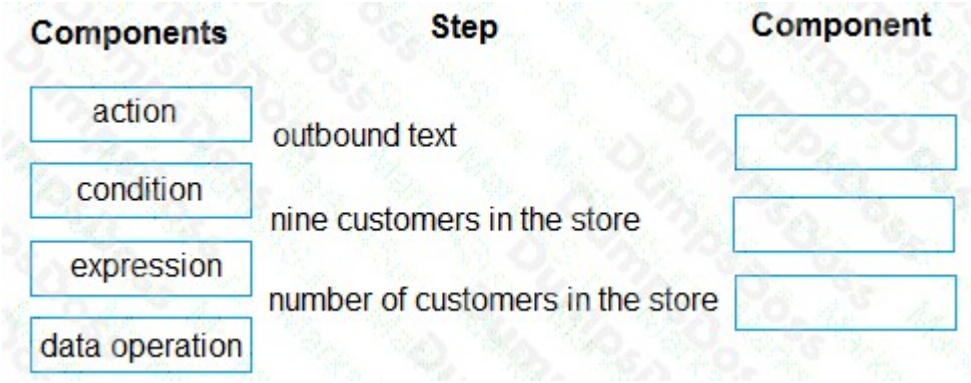
## QUESTION NO: 6 - (SIMULATION)

You need to identify why employees are not receiving notification that nine customers are checked in and waiting in the repair area.

Which components should you test for each step? To answer, drag the appropriate components to the correct steps. Each component may be used once, more than once, or not at all.

You may need to drag the split bar between panes or scroll to view content.

**NOTE:** Each correct selection is worth one point.




## ANSWER: See the explanation for the answer

**Explanation:**

Test the components in this order:

**outbound text → action**
**nine customers in the store → condition**
**number of customers in the store → data operation**

The data operation obtains or calculates the repair-area customer count, the condition evaluates whether it exceeds eight (nine in this case), and the action sends the outbound text notification.

## QUESTION NO: 7

You have a Power Automate flow. The flow uses an HTTP action to connect to a web sen/ice and update a Microsoft Dataverse record with the result. You must modify the flow to send an email if the web service connection fails. You need to update the flow.

Which two actions should you perform? Each correct answer presents part of the solution. Choose two. NOTE: Each correct selection is worth one point.

**A.** Configure the send email step to run when the previous step fails.

**B.** Add a send email step after the update Dataverse record step.

**C.** Add a send email step after the HTTP step.

**D.** Update the HTTP step settings and set the retry policy to: None.

## ANSWER: A D

**Explanation:**

Configure the send email step to run when the previous step fails. Power Automate actions normally run only after the preceding action succeeds. Using *Configure run after* on the email action and selecting the failed state makes the notification path execute when the HTTP action has a failure outcome. This provides an explicit failure-handling branch without requiring a separate flow.

Update the HTTP step settings and set the retry policy to: None. HTTP actions can retry transient failures automatically under their default retry behavior. Setting the policy to *None* disables those automatic retry attempts, so a failed web-service connection is surfaced immediately as a failure and the configured email notification can run. This is appropriate when the requirement is to notify on a connection failure rather than wait for retries to be exhausted.

Power Automate documents that run-after configuration controls whether a subsequent action runs after a prior action succeeds, fails, is skipped, or times out. Its retry policy settings also allow an action's automatic retry behavior to be disabled. See Configure run after in Power Automate and Retry policies in Power Automate.

## QUESTION NO: 8

A company has a model-driven app that uses Microsoft Dataverse.

"he company requires a web application that retrieves information from the model-driven app. The requirements for the web application include:

• Must be a single-page web application that uses the Web API.

• Must display the correct company information.

• Must authenticate using OAuth without additional verification.

You need to configure the web application. Which two methods should you use? Each correct answer presents a complete solution. NOTE: Each correct selection is worth one point.

**A.** NTLM authentication

**B.** multifactor authentication

**C.** Kerberos Authentication

**D.** Microsoft Authentication Library (MSAL)

**E.** Microsoft Azure Active Directory Authentication Libraries (ADAL)

**ANSWER: D E**

**Explanation:**

Microsoft Authentication Library (MSAL) is an appropriate client library for a single-page application that calls the Dataverse Web API. MSAL obtains Microsoft Entra ID OAuth access tokens for the signed-in user and supports the authorization code flow with PKCE, which is the recommended approach for browser-based public clients. The application then sends the acquired bearer token in requests to the Dataverse Web API, allowing Dataverse to authorize access according to that user's security roles and privileges.

Microsoft Azure Active Directory Authentication Libraries (ADAL) also provided OAuth token acquisition for Azure AD-protected services and was the legacy client library supported by older Dataverse web-application implementations. Therefore, it meets the intended historical exam scenario as another complete OAuth authentication method. For new development, Microsoft recommends MSAL rather than ADAL because ADAL is retired; however, ADAL remains the second intended answer for this question's library-based alternatives. See Authenticate to Dataverse using OAuth and MSAL.js documentation.

**QUESTION NO: 9**

You are developing reusable business logic for a Dataverse solution.

The logic must be invoked from client-side JavaScript and from a cloud flow. Only users who have a specific security privilege must be able to invoke the logic.

You need to configure the business logic.

Which two actions should you perform? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

**A.** Create a Custom API.

**B.** Set the Execute Privilege Name property for the Custom API.

**C.** Create a business process flow for the operation.

**D.** Enable field-level security for all Custom API request parameters.

**E.** Create a Dataverse duplicate detection rule.

**ANSWER: A B**

**Explanation:**

Create a **Custom API** to expose the reusable operation through the Dataverse Web API. A custom API can be bound or unbound and can define request parameters and response properties. It can be invoked from JavaScript by using `Xrm.WebApi.online.execute` and from Power Automate by using the Microsoft Dataverse connector.

Set the Custom API **Execute Privilege Name** property to the required privilege. Dataverse checks this privilege when the custom API is invoked. Users who do not have the privilege are denied access to the operation, allowing the API to be secured independently of the tables that the implementation uses.

For more information, see Create and use custom APIs and Secure a custom API with a privilege.

## QUESTION NO: 10

You are implementing JavaScript for a model-driven app form.

The form must not be saved when a validation condition is not met. The validation is performed in an OnSave event handler.

You need to cancel the save operation.

What should you use?

**A.** executionContext.getEventArgs().preventDefault()

**B.** formContext.data.refresh()

**C.** formContext.ui.close()

**D.** formContext.data.save()

## ANSWER: A

**Explanation:**

`executionContext.getEventArgs().preventDefault()` is correct because it cancels the save operation from an OnSave event handler. The event handler receives the execution context, and the save event arguments expose the `preventDefault` method for preventing the save from continuing.

`formContext.data.refresh()` refreshes form data and does not cancel a save. `formContext.ui.close()` closes the form and does not provide validation control over the save pipeline. `formContext.data.save()` initiates a save rather than preventing one.

For more information, see preventDefault (Client API reference).

## QUESTION NO: 11

You need to reduce response time for the information email on the website.

What should you create?

**A.** a flow that creates a SharePoint item for each email response

**B.** a flow that creates a notification in Microsoft Teams

**C.** a Power Apps app that displays the number of email received in a dashboard

**D.** a logic app that moves all emails received to Azure Blob storage

## ANSWER: B

**Explanation:**

A flow that creates a notification in Microsoft Teams is the appropriate solution because it can alert the responsible team as soon as a message arrives at the information email address. Rather than relying on users to periodically check the mailbox, an automated cloud flow can use an email-arrival trigger and then post a notification or message to a Teams channel or chat. This makes the new inquiry immediately visible in the collaboration tool the organization already uses, helping staff acknowledge and assign the request sooner and thereby reducing the overall response time.

Power Automate includes Outlook email triggers, such as receiving a new email, that can start the process automatically. The Teams connector can then post a message in a channel or chat and can include relevant details from the email, such as the sender, subject, and message content. The flow can also target a designated support channel so that the correct group receives the alert consistently. This is a low-code Power Platform solution that directly addresses the delay between email receipt and staff awareness. See Use email triggers in Power Automate and Microsoft Teams connector reference.

## QUESTION NO: 12

A company employs four Microsoft Power Platform developers. Each developer has an individual development environment. Each development environment is refreshed regularly from source control as a managed layer.

The developers must implement their changes in isolation within these environments. Development work must be merged into a base solution within a shared sandbox environment.

You need to ensure work carried out by each developer is deployed into the main solution in the sandbox environment.

What should you do?

**A.** Export each developer ' s solution as managed.

**B.** Perform all customization* in the shared environment.

**C.** Copy each development to the shared environment each day.

**D.** Export each developer ' s solution as unmanaged.

## ANSWER: D

**Explanation:**

Export each developer ' s solution as unmanaged. is correct because unmanaged solutions are the appropriate format for moving and consolidating active development work between development environments. Each developer can work independently in an unmanaged solution layer in their own environment, then export that solution and import it into the shared sandbox environment. Importing the developers' unmanaged solutions into the sandbox enables their component changes to be incorporated into the shared base solution and allows the team to continue making modifications there.

Microsoft's solution-layering model supports this approach: unmanaged customizations sit above managed layers, so the managed solution restored from source control remains the foundation while developer changes are maintained as a separate, editable layer. This separation is especially useful for isolated developer environments that are routinely refreshed, because the developer's work can be packaged and transferred before the environment is reset. The shared sandbox can then serve as the integration environment where changes from multiple developers are assembled and validated before a managed release artifact is produced for downstream environments. Microsoft recommends unmanaged solutions for development and managed solutions for distribution to environments where direct customization should be restricted. See Solution concepts for application lifecycle management and Solution layers.

## QUESTION NO: 13

You need to determine the cause for the issue reported by the interviewers. What is the root cause of the issue?

**A.** The plug-in used to synchronize the Person of Interest field from Contact to Interview was not triggered.

**B.** There was an error in the event pipeline and the entire transaction was rolled back.

**C.** There is no plug-in registered to run when an interview record is created.

## ANSWER: C

**Explanation:**

There is no plug-in registered to run when an interview record is created. A Dataverse plug-in does not execute merely because its assembly exists or because similar logic is registered for another table or message. It must have an active plug-in step registered for the specific table, message, and pipeline stage that correspond to the operation being performed. For logic that populates or synchronizes values at the time an Interview row is created, the required step must target the Interview table and the Create message. Without that registration, Dataverse has no event handler to invoke during creation, so the synchronization logic never runs and the expected field value is not populated. Plug-in registrations are made through a plug-in step, which defines the event message, primary table, execution stage, and other runtime behavior. Microsoft describes plug-in steps and their event registrations in the Dataverse plug-in registration documentation. The Create

message is one of the Dataverse event messages available for plug-in processing, as documented in the Dataverse event framework.

## QUESTION NO: 14

An organization uses Dynamics 365 Sales. You plan to add a custom button to the app ribbon.

You need to ensure that the button displays only when conditions specified by business rules are met.

Which two code segments can you use? Each correct answer presents a complete solution.

NOTE: Each correct selection is worth one point.

**A.** gridContext.refresh();

**B.** formContext.ui.refreshRibbon(refreshAll);

**C.** formContext.data.refresh(save).then(successCallback, errorCallback);

**D.** formContext.ui.refreshRibbon();

**E.** formContext.getControl(arg).refresh();

## ANSWER: B D

**Explanation:**

`formContext.ui.refreshRibbon(refreshAll);` and `formContext.ui.refreshRibbon();` are valid ways to cause a model-driven app command bar to reevaluate its ribbon rules. Ribbon commands commonly use enable rules or display rules that depend on values held in the current form. When client-side logic, including logic associated with business-rule-driven field changes, affects a value used by one of those rules, refreshing the ribbon applies the updated condition and can show or hide the custom button accordingly.

The `refreshRibbon` method accepts an optional Boolean parameter. Supplying `refreshAll` permits the caller to control whether every ribbon command bar on the page is refreshed; when the parameter is omitted, the method refreshes the page-level ribbon command bar. Thus, either invocation can be used as the complete ribbon-refresh action after the relevant business condition has changed. In a real implementation, `refreshAll` must resolve to a Boolean value, such as `true` or `false`, or be a Boolean variable in scope.

Microsoft documents this method specifically for scenarios in which a ribbon rule depends on form data that has been changed programmatically. See formContext.ui.refreshRibbon and Define ribbon enable rules.

## QUESTION NO: 15

A company has a Power Pages site for authenticated partners.

Partners must use Power Pages Web API requests to read and create records in a custom Request table. Partners must not be granted access to records that belong to other partners.

You need to configure access to the custom table.

Which three actions should you perform? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

**A.** Create a table permission that grants the required privileges for the Request table.

**B.** Associate the table permission with the partner web role.

**C.** Configure site settings to enable Web API access and expose the required Request table columns.

**D.** Create a page permission for the Request table.

**E.** Add the Request table to a basic form and grant anonymous access to the page.

**ANSWER: A B C**

**Explanation:**

Create a table permission for the Request table and grant only the required privileges, such as Read and Create. Table permissions control Power Pages access to Dataverse table data and can be configured with an appropriate access type to limit records to the relevant contact, account, or parent relationship.

Associate the table permission with the web role assigned to the partner users. A table permission does not grant access until it is associated with a web role that the authenticated users receive.

You must also enable Power Pages Web API access for the table by configuring the required site settings. This includes enabling the table for Web API access and defining the columns that can be exposed. The Web API honors the table permissions and web-role access that are configured for the site.

For more information, see Power Pages Web API overview and Configure table permissions.

## QUESTION NO: 16

You need to ensure that Adventure Works Cycle can track information from visitors to bike fairs.

What should you create?

**A.** A workflow in Dynamics 365 Sales Engagement for capabilities leads

**B.** A flow to capture customer data from the bike fair Power Apps in SharePoint and create a lead in Microsoft Teams.

**C.** A flow that connects with the bike fair Power Apps to create a lead in Dynamics 365 Sales

**D.** A Microsoft flow that generates a new customer record in SharePoint

**ANSWER: C**

**Explanation:**

A flow that connects with the bike fair Power Apps to create a lead in Dynamics 365 Sales is the appropriate solution because a lead is the Dynamics 365 Sales record type used to capture and qualify a potential customer who has not yet been established as an account or contact. Information collected from a visitor at a bike fair—such as name, email address, product interests, and consent preferences—can be submitted through the Power Apps app and passed to a Power Automate cloud flow. The flow can then use the Microsoft Dataverse connector to create a Lead row in the Dynamics 365 Sales environment. This provides a consistent, centrally managed sales record that sales staff can qualify, enrich, assign, and eventually convert into the relevant customer and opportunity records. Power Automate supports automated business processes across Power Apps and Dataverse, enabling the event-capture app to reliably trigger the lead-creation process without requiring users to manually re-enter data. Microsoft describes leads as potential customers that can be qualified into opportunities, accounts, and contacts; this makes leads the suitable starting point for tracking prospective bike-fair visitors. See Manage leads in Dynamics 365 Sales and Use Dataverse triggers in Power Automate.

## QUESTION NO: 17

A client requires that the system send an email from a button on their customer contact form.

You need to call the action from JavaScript.

Which two functions achieve this result? Each correct presents a complete solution.

NOTE: Each correct selection is worth one point.

**A.** Xrm.WebApi.online.createRecord()

**B.** Xrm.WebApi.online.updateRecord()

**C.** Xrm.WebApi.online.execute()

**D.** Xrm.WebApi.online.executeMultiple()

**ANSWER: C D**

**Explanation:**

`Xrm.WebApi.online.execute()` is appropriate because it invokes one Dataverse Web API operation from client-side JavaScript, including an action. A button command can build the action request object, define its metadata and bound or unbound parameters as applicable, and submit that request through `execute`. The resulting promise enables the script to handle successful completion or an error response.

`Xrm.WebApi.online.executeMultiple()` also achieves the requirement because it accepts a collection of Web API operations, which can include action requests. A collection containing the single email-sending action is a complete solution; it is also useful when the button workflow needs to submit the action together with other action, function, create, update, or delete requests. The method supports configurable batch behavior, including whether requests should be independent or processed as a changeset where supported. Both APIs are online Web API methods, so the form command must run while the user has connectivity to the Dataverse service.

Microsoft documents single-operation execution at Xrm.WebApi.online.execute and collection/batch execution at Xrm.WebApi.online.executeMultiple.

## QUESTION NO: 18

You need to resolve the address validation API error.

Which method should you use to connect?

**A.** an Azure function triggered by a webhook

**B.** JavaScript code

**C.** a custom connector used in a cloud flow

**D.** a plug-in attached to a custom action called from JavaScript

**ANSWER: C**

**Explanation:**

**a custom connector used in a cloud flow** is the appropriate method for connecting Power Platform to an address-validation API. A custom connector provides a reusable, managed definition of an external service's REST API, including its host, endpoints, request and response schemas, authentication method, and required headers or parameters. After the connector is configured and tested, its operations are available as actions in a cloud flow. The flow can submit address data to the validation service, consume the returned validation result, and then use that result to update Dataverse data or continue the business process.

This approach keeps the integration configuration centralized and avoids embedding endpoint or authentication details directly in client-side code. It also allows the connection to use an appropriate authentication configuration and connection reference, which supports deployment across Power Platform environments. For an address-validation scenario, the flow can be triggered by the relevant business event and call the custom connector as a discrete integration step, providing a maintainable way to handle API requests and outcomes. Microsoft describes custom connectors as wrappers around REST or SOAP APIs that make those APIs available to Power Apps, Power Automate, and Logic Apps. See Custom connectors overview and Use custom connectors in Power Automate.

## QUESTION NO: 19 - (DRAG DROP)

You are testing the deployment of a model-driven app. Microsoft Entra ID groups are used exclusively to manage environment- and team-level permissions.

Users encounter several access-related issues that prevent them from testing the app.

You need to select the component to use to resolve each issue by using the principle of least privilege.

What should you assign to resolve each issue? To answer, move the appropriate resolutions to the correct issues. You may use each resolution once, more than once, or not at all. You may need to move the split bar between panes or scroll to view content.

NOTE: Each correct selection is worth one point.

**ANSWER:**

**Explanation:**

The correct assignments are Microsoft Entra ID group for being unable to access a single environment, Security role for being unable to update Account rows, Team for being unable to access rows owned by the human resources team, and Microsoft Power Platform for being unable to create a sandbox environment.

An environment can be associated with a Microsoft Entra security group. This makes the group the narrowest appropriate mechanism for controlling who can enter that specific environment, while keeping group-based access management centralized in Microsoft Entra ID. Microsoft documents this environment access pattern in Control user access to environments: security groups and licenses.

After a user can enter an environment, access to Dataverse tables and operations is governed by security roles. The permission to change an existing Account record is a table-level privilege, so assigning a security role that includes the needed Account write privilege grants exactly the required application capability. This matches the Dataverse security model described in Security roles and privileges.

Records owned by a team are accessed through that team's membership and privileges. Assigning the user to the human resources team is therefore the appropriate way to grant access to the team-owned rows without extending ownership or access beyond that collaboration boundary. For group-managed membership, a Microsoft Entra group can be associated with a team, as described in Manage group teams.

Creating a sandbox is an environment administration task, so it requires the relevant Microsoft Power Platform administrative capability rather than record-level or team-level permission. The Power Platform administration guidance for creating environments is available at Create an environment.

## QUESTION NO: 20

You are developing an integration for a custom Asset table in Microsoft Dataverse.

When an Asset record is deleted, the integration must send selected values from the deleted record to an external system. The external call must not delay the user who deletes the record.

You need to register a plug-in step.

Which two actions should you perform? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

**A.** Register the step asynchronously in the PostOperation stage.

**B.** Add a pre-entity image that includes the required Asset columns.

**C.** Add a post-entity image that includes the required Asset columns.

**D.** Register the step synchronously in the PreValidation stage.

**E.** Use an Update message step instead of a Delete message step.

**ANSWER: A B**

**Explanation:**

Registering the Delete step for **asynchronous PostOperation** processing prevents the external integration logic from running in the user-facing delete transaction. The deletion completes first, and the asynchronous service later executes the plug-in step.

Add a **pre-entity image** that contains only the Asset columns required by the external system. After a record is deleted, the plug-in cannot retrieve the deleted row. The pre-image captures values that existed before the Delete operation and makes them available to the plug-in execution context. A post-image is not available for a Delete operation because the row no longer exists after the operation.

For more information, see Add an entity image and Dataverse event framework.

## QUESTION NO: 21

You are configuring a Microsoft Power Virtual Agents chatbot to use the authenticate option for Microsoft Teams only.

You need to select the variables that will return information about the logged in user.

Which two variables should you use?

Each correct answer presents a complete solution.

NOTE: Each correct selection is worth one point.

**A.** AuthToken

**B.** UserDisplayName

**C.** UserID

**D.** IsLoggedIn

**ANSWER: B C**

**Explanation:**

When a copilot is configured for Microsoft Teams authentication, it can access Teams user-context details for the signed-in person. **UserDisplayName** returns the display name associated with the user's Microsoft Teams identity. This is useful for personalizing messages, such as greeting the user by name, without requiring the user to enter identifying information in the conversation.

**UserID** represents the unique identifier for the signed-in user. A stable user identifier is appropriate when the copilot must associate actions, records, preferences, or data lookups with the current Teams user. Together, a display name and user identifier provide both a human-readable identity value and a unique programmatic identity value for the logged-in user.

Microsoft documents that Teams provides user context to copilots and bots running in Teams, and that user identity data can be used to tailor the experience for the signed-in user. For current guidance on authentication and user identity in Microsoft Copilot Studio, see Configure user authentication in Copilot Studio and Use variables in Copilot Studio.

## QUESTION NO: 22

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than once correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You are designing a one-way integration from the Common Data Service to another system.

You must use an Azure Function to update the other system. The integration must send only newly created records to the other system. The solution must support scenarios where a component of the integration is unavailable for more than a few seconds to avoid data loss.

You need to design the integration solution.

Solution: Register a service endpoint in the Common Data Service that connects to an Azure Service Bus queue.

Register a step at the endpoint which runs asynchronously on the record's Create message and in the portoperation stage.

Configure the Azure Function to process records as they are added to the queue.

Does the solution meet the goal?

**A.** Yes

**B.** No

**ANSWER: A**

**Explanation:**

Yes. A Dataverse (formerly Common Data Service) service endpoint can send event data to an Azure Service Bus queue when the registered plug-in step executes. Registering the step for the *Create* message limits the integration trigger to newly created records, satisfying the requirement not to send updates or other operations. Using an asynchronous step decouples the record creation transaction from downstream processing.

An Azure Service Bus queue provides durable, brokered messaging between Dataverse and the Azure Function. If the Azure Function or the target system is temporarily unavailable, the message remains in the queue until a receiver successfully processes and completes it, subject to the queue's configured message retention and retry behavior. The Azure Function can use a Service Bus trigger to process queued messages as it becomes available, rather than requiring the receiving component to be online at the instant Dataverse publishes the event. This architecture therefore supports resilient one-way delivery while allowing the function to update the external system independently.

For implementation details, see Azure integration with Dataverse and Azure Service Bus trigger for Azure Functions.

**QUESTION NO: 23**

A company is creating a one-way integration from the Common Data Service to an external system. Data will be sent from a webhook to an Azure Function.

You need to configure the Azure Function to handle data from the webhook.

Which class and data type must the Azure Function handle?

**A.** RemoteExecutionContext in .NET binary format

**B.** RemoteExecutionContext in JSON format

**C.** RemoteExecutionContext in XML format

**D.** IPluginExecutionContext in JSON format

**E.** IPluginExecutionContext in XML format

**ANSWER: B**

**Explanation:**

**RemoteExecutionContext in JSON format** is correct. When Microsoft Dataverse (previously Common Data Service) sends an event to a registered webhook, it sends an HTTP POST request whose body is a string representing the JSON serialization of a `RemoteExecutionContext` instance. The Azure Function should therefore accept the webhook request body and deserialize its JSON payload into a `RemoteExecutionContext` object. This context supplies the execution information needed by the receiving service, including event details, the target entity data, input and output parameters, shared variables, user information, correlation details, and organization context.

The function endpoint must be reachable through HTTPS and should process the posted JSON according to the webhook contract. In practice, an Azure Function can read the request body as a string and use an appropriate serializer to reconstruct the execution-context data required by the integration. This payload format enables Dataverse to communicate event data to an external HTTP endpoint without requiring the endpoint to run inside the Dataverse plug-in execution pipeline. Microsoft documents that the webhook request body contains a JSON value that represents a `RemoteExecutionContext`. See Use webhooks with Dataverse and RemoteExecutionContext class.

**QUESTION NO: 24**

A company implementsDynamics 365 Customer Service. The company deploys synchronous plug-ins for the PreOperation and PostOperation stages on create and for the PostOperation stage on update for processing different case type.

Users experience errors when updating cases. The plug-in trace log files show that the PostOperation plug-in update of case times out after two minutes.

You perform basic testing and discover that this plug-in is triggered on every update of a case. You examine the code and discover that the plug-in retrieves all columns for the updated case record performing its work.

You need to reduce the number of errors. You need to achieve this goal with the test amount of changes.

Solution:

* In the Plug-in Registration tool, add a post Image to the plug-in step and include the

Fields that the plug-in needs.

* Remove the retrieves statement from the plug-in code and reference the post image.

Does the solution meet the goal?

**A.** Yes

**B.** No

**ANSWER: A**

**Explanation:**

Yes. A post-operation entity image supplies a snapshot of the case row after the update has been applied. Configuring the image to include only the fields required by the plug-in lets the plug-in read those values directly from the execution context, rather than issuing a separate Retrieve request for the case on every update. This is a focused, low-impact change because the plug-in's existing processing logic can generally continue to use entity attribute values; only the source entity changes from a retrieved record to the registered post image.

Eliminating an unnecessary retrieve operation is particularly valuable for a synchronous plug-in, because synchronous processing runs within the user's request and contributes directly to the elapsed time before the user receives a response. Limiting the image to required columns also follows Power Platform guidance to avoid retrieving all columns, which can increase processing time and return unnecessary data. Reducing the database operation and data volume helps the update plug-in complete within the applicable execution time limit, thereby reducing timeout-related update errors. Microsoft documents entity images and their availability by pipeline stage in Add an entity image, and recommends selecting only the columns needed for retrieval in Select columns using QueryExpression.

## QUESTION NO: 25

You need to modify Microsoft flow to resolve CustornerCs issue. What should you do?

**A.** Add a data operation that specifies the false conditions.

**B.** Add a configure run that is set to Is successful.

**C.** Add a timeout setting to the approval flow.

**D.** Add b condition containing approval hierarchy.

## ANSWER: C

**Explanation:**

Add a timeout setting to the approval flow. The scenario requires the flow to handle an approval or requested response that remains unanswered for a defined period, in this case one week. Power Automate actions support a timeout configuration through their settings, using an ISO 8601 duration such as `P7D` for seven days. When the configured time elapses without completion, the approval action is considered timed out. The flow can then use run-after behavior to continue with an appropriate follow-up action, such as assigning or sending an approval request to another person or group, notifying the parts department, or escalating the request. This approach directly implements a time-bound response requirement rather than relying on a manually evaluated condition. The timeout belongs on the relevant approval action so that Power Automate can enforce the deadline as part of the flow execution. Microsoft documents timeout settings for actions and the use of configured run-after conditions to control subsequent processing after outcomes such as a timeout. See Microsoft Learn: duration format and Microsoft Learn: configure run after.

## QUESTION NO: 26

You need to replace the bicycle inspection forms.

Which two solutions should you use? Each answer presents part of the solution.

NOTE: Each correct selection is worth one point.

**A.** a flow that maps inspection data to Dynamics 365 Field Service

**B.** a logic app that guides the technician through the inspection

**C.** a canvas app that guides the technician through the inspection

**D.** a model-driven app based on customer service entities

## ANSWER: A C

**Explanation:**

A canvas app that guides the technician through the inspection is appropriate for replacing a paper inspection form because canvas apps support task-focused, customized mobile experiences. The app maker can create an inspection workflow with controls such as check boxes, dropdown lists, photos, signatures, and required fields, allowing technicians to consistently capture the inspection and any recommended work. Canvas apps can use Dataverse as a data source and can be designed for phone or tablet use, which is well suited to technicians completing inspections while working on bicycles.

A flow that maps inspection data to Dynamics 365 Field Service completes the solution by automating the handoff of submitted inspection information into the Field Service process. A cloud flow can be invoked from the canvas app or triggered when inspection data is created, then create or update the relevant Field Service records, such as work-order-related data or service tasks. This keeps the guided data-entry experience separate from downstream Field Service processing while ensuring that inspection results are available for scheduling, follow-up work, and service history. Microsoft describes canvas apps as highly customizable applications that connect to data sources, and Power Automate flows can automate actions across Power Platform services. See Microsoft Learn: Canvas apps and Microsoft Learn: Power Automate.

## QUESTION NO: 27

You are creating a Power Pages site for suppliers.

Authenticated suppliers must be able to view only their own Purchase Request records. The Purchase Request table has a lookup column to the Contact table that identifies the supplier who submitted each request.

You need to configure table permissions.

Which two actions should you perform? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

**A.** Create a table permission for Purchase Request that uses Contact access through the supplier lookup relationship.

**B.** Assign the table permission to the supplier web role.

**C.** Assign a Dataverse security role directly to each supplier contact.

**D.** Enable auditing for the Purchase Request table.

**E.** Create an organization-level Read privilege for Purchase Request.

## ANSWER: A B

**Explanation:**

Create a **table permission** for the Purchase Request table and configure Contact access with the relationship that links a Purchase Request to its submitting Contact. Contact access scopes permissions to records associated with the signed-in contact, which prevents suppliers from viewing records associated with other contacts.

Assign the table permission to the appropriate **web role**. Power Pages applies table permissions to authenticated users through web roles. A supplier who is assigned the web role receives only the privileges and record scope configured by the table permission.

Adding a table permission without associating it to a web role does not grant access to site users. A Dataverse security role is not used to grant Power Pages visitors access to table records.

For more information, see Configure table permissions and Manage web roles.

## QUESTION NO: 28

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than once correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You are designing a one-way integration from the Common Data Service to another system.

You must use an Azure Function to update the other system. The integration must send only newly created records to the other system. The solution must support scenarios where a component of the integration is unavailable for more than a few seconds to avoid data loss.

You need to design the integration solution.

Solution: Register a webhook in the Common Data Service that connects to the Azure Function. Register a step on the webhook which runs synchronously on the record's Create message and in the post-operation stage.

Does the solution meet the goal?

**A.** Yes

**B.** No

**ANSWER: B**

**Explanation:**

No is correct because a synchronous post-operation webhook directly couples record creation to the availability and response time of the Azure Function endpoint. Dataverse waits for the webhook request to complete as part of synchronous processing. Consequently, a temporary outage or delay in the Function is not handled through a durable messaging buffer that can retain the event and deliver it after the dependent component recovers. A resilient integration for newly created records needs asynchronous processing with durable delivery, such as an Azure Service Bus queue or topic, so that the create event can be retained independently of the Function's immediate availability. The Azure Function can then process queued messages when it is available, while message retry and dead-letter capabilities support recovery from prolonged failures. Dataverse supports registering service endpoints and steps for asynchronous execution, and Azure Service Bus is intended to decouple communicating components and reliably broker messages. This design both limits processing to the Create message and avoids depending on a live Function invocation at the time the Dataverse record is created. See Use webhooks with Dataverse and Azure Service Bus messaging overview.

**QUESTION NO: 29**

You create a Power Virtual Agents chatbot in an environment named Environment1. A colleague creates a Power Automate flow in the default solution in the default environment.

The chatbot in Environment1 does not recognize the flow in the default environment.

You need to ensure the chatbot can access the flow.

Which two actions should you perform? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

**A.** Add the Power Automate flow to a solution in Environment1.

**B.** Send a copy of the Power Automate flow from the default environment.

**C.** Add the Power Automate flow to a custom solution in the default environment.

**D.** Export the solution containing the flow from the default environment and import the solution into Environment1.

**E.** Share the Power Automate flow from the default environment.

**ANSWER: C D**

**Explanation:**

A chatbot can use a cloud flow only when the flow is available in the chatbot's environment. Because the flow was created in the default environment, it must be packaged and deployed to Environment1 rather than merely being made available in its original environment. First, add the Power Automate flow to a custom solution in the default environment. Solution-aware flows can be transported between environments together with the solution metadata required for application lifecycle management. A flow that exists only in the default solution should be added to an exportable, unmanaged solution before it is moved.

Next, export that solution and import it into Environment1. Importing creates the solution component in Environment1, allowing the chatbot author to select and call the flow from that same environment. During import, connections and connection references must be configured as appropriate for the target environment. This deployment pattern keeps the bot and its automation in the same environment and supports repeatable movement between development, test, and production environments. Microsoft documents solution-aware cloud flows and solution export/import in Create a cloud flow in a solution and Export solutions.

**QUESTION NO: 30**

You need to create the model-driven app for referral.

Which function should you add?

**A.** Flow

**B.** Workflow

**C.** Business rule

**D.** Chart

**E.** Subgrid

**ANSWER: C**

**Explanation:**

**Business rule** is the appropriate function because it can apply conditional form logic in a model-driven app without requiring custom code. Configure the rule to evaluate the referral customer field: when it is selected, use the business-rule action to show the additional referral fields; when it is not selected, use an *Else* branch to hide those fields. This directly implements the required interactive behavior as users create or edit an account record. Business rules run on the form and provide actions such as showing or hiding fields, setting field requirement levels, locking or unlocking fields, setting values, and displaying error messages. The rule can be scoped to the account table's main forms so that the behavior is consistently available wherever the relevant account form is used. Because the requirement concerns immediate, conditional visibility of form fields, a business rule is the declarative Power Apps feature designed for this purpose. For configuration details, see Create business rules to apply logic in model-driven apps and Business rules overview.

**QUESTION NO: 31 - (HOTSPOT)**

You need to implement ribbon display rules to control availability for the scoring command button. Which rule types should you use? To answer, select the appropriate options in the answer area. NOTE: Each correct selection is worth one point.

**Answer Area**

**ANSWER:**

Answer Area

**Explanation:**

Use **CustomRule** to configure visibility for recruiters. Command bar display rules include built-in checks for items such as entity privileges and form state, but they do not provide a rule that directly tests whether the current user belongs to a particular security role. A custom rule invokes a JavaScript function, allowing the command to evaluate the current user's roles or another organization-specific definition of a recruiter. The function returns true when the scoring command should be displayed and false when it should be hidden. This makes it the appropriate mechanism for role-specific availability. Microsoft documents the custom-rule behavior in the CustomRule display rule documentation.

Use **FormStateRule** to configure visibility based on the form mode. This rule evaluates the state of the current form, so the scoring command can be shown only when the record is in the required state—for example, on an existing record rather than while it is being created, or only when the form is editable. That directly corresponds to the requirement to make visibility depend on the form's mode. Form-state evaluation is specifically supported as a command definition display rule; see Microsoft's FormStateRule reference. Together, the custom role check and the form-state check ensure that the scoring command is available only to the intended users and only in the appropriate form context.

## QUESTION NO: 32 - (DRAG DROP)

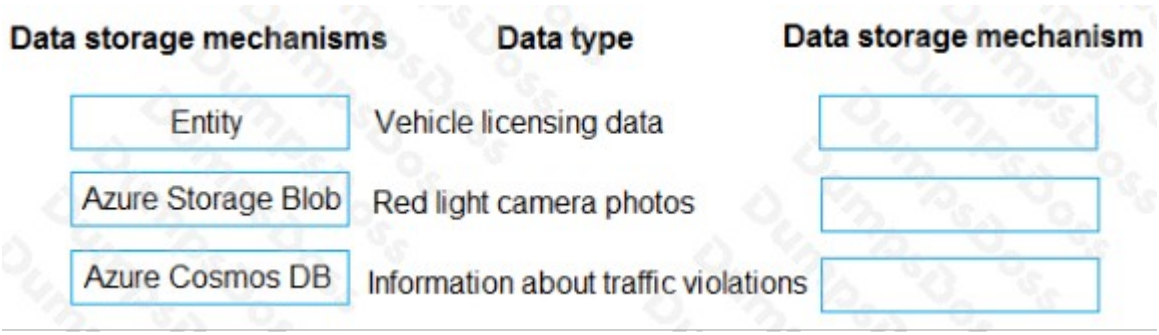
An international organization has a series of client-server applications that manage red light cameras and traffic violations across a wide geographic region. The daily volume of traffic violations is very high and growing.

You plan to use Microsoft Power Platform apps to manage the following types of data:

You need to select data storage mechanisms for the new apps.

Which data storage mechanisms should you use? To answer, drag the appropriate data storage mechanisms to the correct data types. Each storage mechanism may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.

NOTE: Each correct selection is worth one point.




**ANSWER:**

**Explanation:**

Vehicle licensing data should be stored in an **Entity** in Microsoft Dataverse. Licensing details are structured business records, such as a vehicle identifier, license information, registered owner, status, and dates. Dataverse entities provide the relational data model, validation, security roles, auditing, and Power Platform integration that model-driven and canvas apps use for business data. This makes an entity the appropriate operational store for data that the application must create, retrieve, update, secure, and relate to other business records. Microsoft documents Dataverse tables as the way to store business data for Power Platform applications: Microsoft Dataverse overview.

Red light camera photos should be stored in **Azure Storage Blob**. A photograph is unstructured binary content rather than a normal relational record. Blob Storage is designed to store large volumes of unstructured objects efficiently, including images and other media, while allowing the app to retain a reference or metadata record associated with the violation. Azure documents Blob Storage specifically as object storage for massive quantities of unstructured data: Introduction to Azure Blob Storage.

Information about traffic violations should be stored in **Azure Cosmos DB**. The scenario explicitly identifies a high and increasing daily volume across a wide geographic area. Cosmos DB is a distributed NoSQL database built for elastic scale, high throughput, and low-latency access, making it suitable for the continuously growing stream of violation information. Its global-distribution and partitioning capabilities support this high-scale workload while Power Platform can access the data through the appropriate integration approach. See Azure Cosmos DB introduction for its scalable, globally distributed database capabilities.

## QUESTION NO: 33

A financial services company uses the Common Data Service (CDS) to develop solutions. The company uses development and production instances.

You need to move solutions from the development instance to the production instance.

What are two possible ways to achieve this goal? Each correct answer presents a complete solution.

NOTE: Each correct selection is worth one point.

**A.** In the development instance, make changes to the solutions that are deployed in the production instance, export the solutions as managed solutions, and import the managed solutions into the production instance.

**B.** In the development instance, highlight the solution you want to make changes to, select Clone a Patch, make changes, export the solution, and import the solution into the production instance.

**C.** Export all managed solutions from the development instance and import the solutions into the production instance.

**D.** In the production instance, import solutions with the same version number or higher when updating solutions.

## ANSWER: A B

**Explanation:**

Making changes in the development instance, exporting the solution as managed, and importing it into the production instance is a supported solution lifecycle approach. Development should occur in an unmanaged solution in the development environment, while a managed export provides the deployable artifact for production. Importing the managed solution applies its components and updates to the target environment in a controlled manner.

Creating a patch by using *Clone a Patch*, making the required incremental changes, exporting that patch, and importing it into production is also supported. A patch is intended for a small, focused update to an existing solution and is associated with a specific parent solution version. It enables the organization to deploy only the changes included in that patch rather than redeploying the entire solution. Patches can later be rolled up into a new version of the parent solution when appropriate. This supports incremental release management while preserving the solution's upgrade path and dependency structure.

For current Microsoft guidance on solution exports, imports, and managed deployment, see Solution concepts for application lifecycle management and Create patches to simplify solution updates.

## QUESTION NO: 34

A travel company plans to track the address of places their clients visit in an entity named Destination. Client information is captured as contact records. Client records include links to the places that clients visit.

The company must be able to link multiple rating records to the new address record.

You find a custom Rating entity that is incomplete.

You need to expand the Rating entity to include contact, address, and rating information in one place.

Which three actions should you perform? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

**A.** Create a 1:N relationship between the Contact system entity and the Address system entity named Destination.

**B.** Create a mapping for the Contact – Rating relationship.

**C.** Create a 1:N relationship between the Address system entity and the Rating entity.

**D.** Create a 1:N relationship between the Contact system entity and the Rating entity.

**E.** Create a mapping for the Destination – Rating relationship.

**F.** Create a 1:N relationship between the Destination entity and the Rating entity.

## ANSWER: A C E

**Explanation:**

Creating a 1:N relationship between the Contact system entity and the Address system entity named Destination establishes the required client-to-destination structure. A one-to-many relationship lets one contact be associated with multiple destination address records, while each destination address is associated with the appropriate contact. This provides the client context needed for destinations.

Creating a 1:N relationship between the Address system entity and the Rating entity enables the required rating model: one destination address can have multiple related Rating records. The Rating table therefore becomes the child in that relationship and can retain the lookup to the destination address for every rating captured.

Creating a mapping for the Destination – Rating relationship completes the data-entry experience by automatically carrying mapped values from the destination address into a newly created related Rating record. Dataverse relationship mappings apply when a related child record is created through the relationship, reducing repeated entry and ensuring the Rating record includes relevant destination information alongside its rating data. Microsoft documents one-to-many relationships and their lookups at Create and edit table relationships, and describes field mappings for related records at Map table fields.

## QUESTION NO: 35

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

A Common Data Service (CDS) environment has two custom entities named Building code and Work item. Building code has a code date custom field and Work item has an elapsed time custom field. Construction workers use a consolidated custom form with data from both entities to fill.in their daily work items.

A JavaScript library is used with these custom entities and fields to apply complex logic.

You need to ensure that the JavaScript library continues to function as originally designed if other developers expand the environment.

Solution: In the JavaScript library, add Building code with Code date and Work item with Elapsed time in the dependencies tab.

Does the solution meet the goal?

**A.** Yes

**B.** No

**ANSWER: A**

**Explanation:**

Yes. Adding the Building code entity and its Code date field, plus the Work item entity and its Elapsed time field, as dependencies of the JavaScript web resource explicitly records the components that the script relies on. This is particularly important when a library uses entity or attribute logical names in client-side code: those references are not necessarily discoverable automatically by the platform.

Declared dependencies protect the integrity of the customizations as the environment evolves. For example, if a developer attempts to remove or modify a referenced component through solution management, Dataverse can identify that the web resource has a dependency on it. The dependency metadata also travels with the solution, helping ensure that solution deployment respects the required entity and field components. This preserves the script's assumptions about the schema and reduces the risk that later changes leave the consolidated form calling code against unavailable metadata.

Microsoft documents that web resource dependencies can be configured for components, including entity metadata and fields referenced by a web resource. See Web resource dependencies and Dependency tracking for solution components.

**QUESTION NO: 36**

You are developing a Dataverse plug-in that occasionally fails in a production environment.

You need to collect diagnostic information from both successful and failed executions without attaching a debugger to the production service.

Which two actions should you perform? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

**A.** Set the Plug-in trace log setting to All.

**B.** Use ITracingService.Trace in the plug-in code.

**C.** Attach a debugger to the production Dataverse worker process.

**D.** Enable auditing for every table used by the plug-in.

**E.** Add a synchronous workflow after the plug-in step.

**ANSWER: A B**

**Explanation:**

Configure the environment Plug-in trace log setting to record All executions. This setting causes Dataverse to retain Plug-in Trace Log records for successful and failed plug-in executions, subject to the environment's trace-log retention configuration.

The plug-in must also obtain `ITracingService` from the service provider and call its `Trace` method to write useful diagnostic information. Trace messages can include execution-path details, input values that are appropriate to log, timing information, and exception context. These messages are stored in the Plug-in Trace Log record when tracing is enabled.

Solution checker can identify some design concerns, but it does not capture runtime execution details. For more information, see Debug plug-ins with tracing and Logging and tracing in Dataverse.

## QUESTION NO: 37

You need to configure the app to meet the address verification requirements.

Which two actions should you perform? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

**A.** Enable the command bar button only when the address shows as not verified.

**B.** Make the address fields read-only after they have been verified.

**C.** Open a confirmation dialog when the command bar button is selected.

**D.** Clear the field indicating the address is valid when an address field changes.

## ANSWER: A D

**Explanation:**

“Enable the command bar button only when the address shows as not verified.” is correct because the verification action should be available only when it is relevant. Command bar rules in a model-driven app can evaluate record data and control whether a command is available. Restricting the command to unverified addresses helps guide users through the intended process and prevents unnecessary repeat verification of an unchanged, already validated address.

“Clear the field indicating the address is valid when an address field changes.” is also correct because verification status is dependent on the values that were submitted for verification. If a user edits any component of the address after it has been marked valid, the prior result can no longer be treated as confirmation of the current address. Clearing the validity indicator returns the record to an unverified state, allowing the command to become available again and ensuring that the revised address is verified before it is relied upon. This behavior can be implemented with client-side form logic that responds to column changes and updates the verification-status column. See addOnChange (Client API reference) and Define ribbon enable rules.

## QUESTION NO: 38

You need to choose a technology to access the Web API. Which technology should you select?

**A.** Canvas app that uses the custom connector

**B.** Plug-in that contacts the Web API

**C.** Power Automate flow that uses the custom connector

**D.** Webhook that contacts the Web API

## ANSWER: A

**Explanation:**

Canvas app that uses the custom connector is the appropriate choice because a custom connector provides a reusable, strongly defined way for a canvas app to call a REST-based Web API. The connector defines the API host, security

configuration, operations, request parameters, and response schemas. Once configured and shared in the appropriate Power Platform environment, app makers can add it as a data source and invoke its operations from Power Fx formulas. This approach supports interactive application experiences: users can initiate calls from screens or controls, and the returned data can be displayed, filtered, or incorporated into the app's business process. Custom connectors can use common authentication approaches, including OAuth 2.0, API keys, and basic authentication where supported by the target service. They also centralize connection and connector configuration rather than requiring API-call implementation details to be duplicated throughout the canvas app. Microsoft documents custom connectors specifically as the mechanism for connecting Power Apps to REST APIs that are not available through a standard connector. See Custom connectors overview and Register and use custom APIs in canvas apps.

## QUESTION NO: 39

You are creating a canvas app that realtors use to identify neighbors for properties that are for sale. The OnStart property includes the following code:

The app is running slower than expected.

You need to resolve the issue.

What should you do?

A)

B)

C)

**A.** Option A

**B.** Option B

**C.** Option C

**D.** Use the Concurrent function to run independent OnStart data-loading operations in parallel.

## ANSWER: D

**Explanation:**

Use the `Concurrent` function to execute independent data-loading formulas in parallel during app startup. Canvas app formulas separated by semicolons normally run sequentially, so if the startup logic performs multiple unrelated connector or data-source calls, each request waits for the preceding request to finish. This can significantly increase the time before the app is ready for a realtor to use.

Wrap the independent `ClearCollect`, `Collect`, or other retrieval operations in `Concurrent(...)`. For example, separate collections used to identify a property, its neighborhood, and related neighbor information can be populated at the same time, provided none of those operations depends on a result produced by another operation. Power Apps waits for all formulas within `Concurrent` to complete before it continues with formulas that follow it, preserving a reliable startup sequence while reducing total wait time.

This is a recommended Power Apps performance technique for reducing latency caused by multiple independent network calls. See Concurrent function in Power Fx and Fast app page load.

## QUESTION NO: 40

You have two tables, account and contact, with a parent-child relationship. Records in the account table can be created in the UI or from an external system integrated with Dataverse by using an Azure Function.

When a user creates a record in the account table, a default record in the contact table must be created automatically.

You need to implement a process to create the contact records.

Solution: Create a custom API bound to the account table that uses a plug-in. Use the Organization service to create the contact record.

Does the solution meet the goal?

**A.** Yes

**B.** No

**ANSWER: B**

**Explanation:**

**No** is correct because binding a custom API to the account table does not cause that API to run automatically when an account row is created. A custom API defines a named Dataverse operation that must be explicitly invoked by a client, such as application code, the Dataverse Web API, or an SDK request. Therefore, account records created through the model-driven app UI or directly by the Azure Function would not consistently trigger the custom API unless both creation paths were deliberately changed to call it.

The stated requirement is event-driven: creation of an account must automatically result in creation of its default contact. Dataverse plug-ins support this through registration on the `Create` message for the account table, typically in the post-operation stage so that the account identifier is available and the plug-in can use the Organization service to create the related contact. Such a plug-in executes regardless of whether the account is created through the UI, SDK, Web API, or another supported Dataverse entry point, subject to the plug-in step's registration settings. Microsoft describes custom APIs as operations invoked by callers, while plug-in steps are registered to execute in response to Dataverse messages. See Microsoft documentation on custom APIs and Microsoft documentation on Dataverse plug-ins.

## QUESTION NO: 41

An organization implements Dynamics 36S Supply Chain Management.

You need to create a Microsoft Flow that runs daily.

What are two possible ways to achieve this goal? Each correct answer presents a complete solution.

NOTE: Each correct selection is worth one point.

**A.** Create the flow and set the flow frequency to daily and the interval to 1.

**B.** Create the flow and set the flow frequency to hourly and the interval to 24.

**C.** Create the flow and set the flow frequency to hourly and the interval to 1.

**D.** Create the flow and set the flow frequency to daily and the interval to 24.

**ANSWER: A B**

**Explanation:**

A scheduled cloud flow uses the Recurrence trigger to run on a defined cadence. The trigger combines a frequency unit with an interval value: the interval specifies how many units of the selected frequency occur between runs. Therefore, setting the flow frequency to daily with an interval of 1 schedules one run every day. This is the direct daily schedule configuration and is appropriate when the flow should recur once per calendar day.

Setting the flow frequency to hourly with an interval of 24 also schedules one run per day because 24 hours elapse between each recurrence. This is an equivalent duration-based configuration for a daily execution cadence. In either case, the flow designer can further configure the trigger's time zone and advanced scheduling settings, such as a specific start time, when the business process must run at a particular time of day.

Microsoft documents that scheduled cloud flows use the Recurrence trigger and that its interval and frequency determine when the flow runs. See Run scheduled tasks with Power Automate and Types of flows in Power Automate.

## QUESTION NO: 42

You plan to create a canvas app to manage large sets of records. Users will filter and sort the data.

You must implement delegation in the canvas app to mitigate potential performance issues.

You need to recommend data sources for the app.

Which two data sources should you recommend? Each correct answer presents a complete solution.

NOTE: Each correct selection is worth one point.

**A.** SQL Server

**B.** Common Data Service

**C.** Azure Data Factory

**D.** Azure Table Storage

## ANSWER: A B

**Explanation:**

**SQL Server** and **Common Data Service** are appropriate choices because both support delegation in canvas apps. Common Data Service is the former name for Microsoft Dataverse, which is the current Power Platform data platform. With a delegable query, Power Apps sends supported filtering and sorting operations to the underlying service instead of retrieving only a limited initial set of records and processing it locally. This is essential for an app that must manage large record sets, because users can filter and sort across the full dataset rather than potentially receiving incomplete results after reaching the nondelegation record limit.

SQL Server supports delegation for commonly required canvas-app operations, including supported `Filter`, `Sort`, and `SortByColumns` expressions. Dataverse likewise supports delegation for a broad set of query operations and is designed to work natively with Power Apps. The exact formula must use functions, operators, and column types that the selected connector can delegate; app makers should heed delegation warnings and validate queries against the connector's supported-operation list. Microsoft's delegation overview and the Dataverse delegation reference provide the relevant guidance: Delegation overview and Filter and delegation in Power Fx.

## QUESTION NO: 43 - (HOTSPOT)

HOTSPOT

You have the following JavaScript function: (Line numbers are included for reference only.)

```
01 function displayIconTooltip(rowData, userLCID)
02 {
03 var imgName = "";
04 var tooltip = "Relationship Health";
05 var str = JSON.parse(rowData);
06 var prevrev = str.new_previousyearannualrevenue_Value;
07 var rev = str.revenue_Value;
08 var health = parseFloat(rev) - parseFloat(prevrev);
09 if (health > 0)
10    imgName = "new_good";
11 else if (health == 0)
12    imgName = "new_warm";
13 else
14    imgName = "new_bad";
15 var resultarray = [imgName, tooltip];
16 return resultarray;
17 }
```

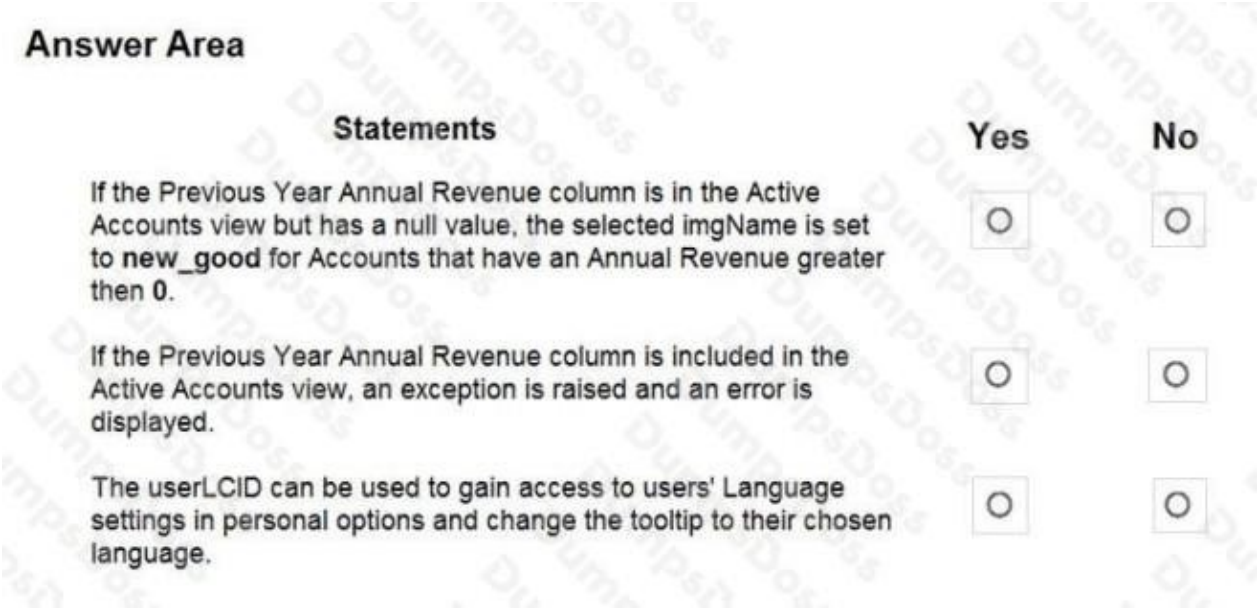
The Annual Revenue view column is configured to call the function as shown in the Column Properties exhibit. (Click the Change Column Properties tab.)

Users report that the icons that appear in the Active Accounts view are incorrect, as shown in the Active Accounts View exhibit. (Click the Active Accounts View tab.)

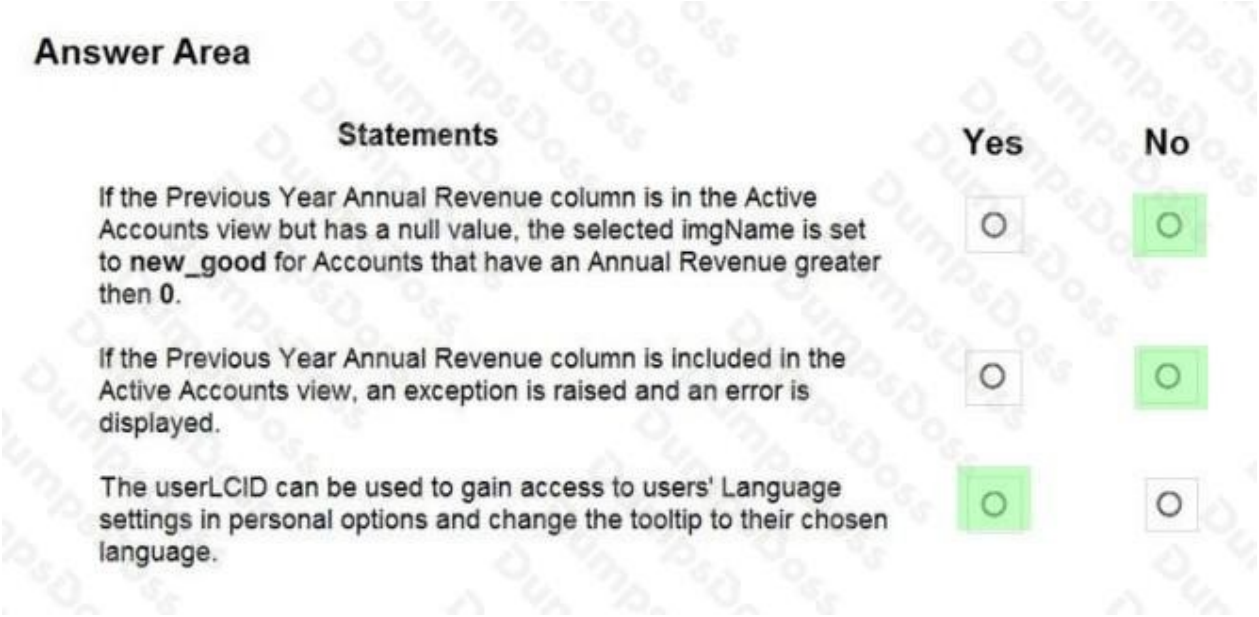
You need to determine why the incorrect icons are being displayed.

For each of the following statements, select Yes if the statement is true. Otherwise, select No.

**Hot Area:**




**ANSWER:**




**Explanation:**

The correct selections are No, No, and Yes. A custom image-provider function evaluates the values supplied in the view row data by using normal JavaScript semantics. In particular, `parseFloat` does not turn an unavailable or nonnumeric value into zero. It returns `NaN`. As documented by MDN's parseFloat reference, the result is `NaN` when the input cannot be converted to a number. Numeric comparisons that involve `NaN` do not produce a successful greater-than result. Consequently, an empty Previous Year Annual Revenue does not by itself cause the good-revenue image to be selected, even if the current Annual Revenue contains a positive value.

Similarly, data that is not supplied for a view column is not automatically an exception condition for this type of JavaScript calculation. The function can receive an unavailable property as `undefined`; parsing that value yields `NaN`, allowing the conditional image logic to continue without displaying an error. This is why the icon behavior can be incorrect without the view failing to render.

The user LCID is appropriate for localization logic. In model-driven apps, the user context exposes the user's language identifier, which enables client-side code to select localized resources or text, such as a tooltip, matching the language configured for that user. Microsoft documents the user-language API through userSettings in getGlobalContext. Using that locale identifier lets the function present the tooltip in the user's chosen personal-options language.