DUMPSARENA

Certified Application Developer - ServiceNow

ServiceNow CAD

Version Demo

Total Demo Questions: 32

Total Premium Questions: 328

Buy Premium PDF

https://dumpsarena.co

sales@dumpsarena.co

sales@dumpsarena.co
dumpsarena.co

Topic Break Down

Topic	No. of Questions
Topic 1, Designing and Creating Applications	35
Topic 2, Application User Interface	63
Topic 3, Application Security	55
Topic 4, Application Data Model	44
Topic 5, Application Development	110
Topic 6, Application Collaboration	21
Total	328

QUESTION NO: 1

Which scenario will a Scheduled Script Execution (Scheduled Job) be implemented?

A. The application executes a client-side script daily at the same time

B. The application queries the database daily to check unassigned records

C. The application displays a custom welcome message when a user logs in

D. The application validates form input fields before a record is submitted

ANSWER: B

Explanation:

The application queries the database daily to check unassigned records is an appropriate use case for a Scheduled Script Execution. A Scheduled Script Execution runs server-side JavaScript automatically according to a defined schedule, without requiring a user to open a form, load a page, or take any action. This makes it suitable for recurring background processing such as finding records that meet defined criteria, evaluating their state, and performing follow-up work such as updating records, creating tasks, sending notifications, or assigning work. For example, a daily job can query a task table for records whose assignment group or assignee is empty and then apply an organization's escalation or routing logic. Scheduled Script Executions are configured under System Definition > Scheduled Jobs and can be set to run once or repeatedly at a selected interval. The script executes on the instance, so it can use server-side APIs such as GlideRecord to retrieve and process database records. This recurring database-check pattern is exactly the type of unattended automation Scheduled Script Executions are designed to support. See the ServiceNow developer documentation on GlideRecord server-side API and the ServiceNow product documentation search for Scheduled Jobs.

QUESTION NO: 2

What field type would you select if you want to query records from another table on a form?

A. Use the Date field type.

B. Use the Phone Number field type.

C. Use the String field type.

D. Use the Reference field type.

ANSWER: D

Explanation:

The correct choice is **Use the Reference field type.** In ServiceNow, a reference field establishes a relationship from a field on the current table to a record in a specified target table. The field stores the target record's unique sys_id while presenting a readable display value to users, such as a person's name or a configuration item's name. On a form, users can search or use the lookup functionality to query the referenced table and select an existing record. This supports normalized data, because the related record is maintained in its own table rather than duplicating its information as plain text on every record.

For example, the Caller field on an incident record references the User [sys_user] table. A user completing the incident can search the user records and select the appropriate caller. Scripts, reporting, and configuration can then use that stored relationship to retrieve fields and related records from the referenced table. When configuring a reference field, the target table is selected through the field's Reference specification. ServiceNow's developer documentation describes reference fields and their use in table relationships at Reference fields. Additional guidance on creating and configuring table fields is available in the Create a new field documentation.

QUESTION NO: 3

What tool is used to import data from various data sources, and map that data into ServiceNow tables?

A. Transform Set

B. Import Set

C. Data Pack

D. Update Set

ANSWER: B

Explanation:

Import Set is correct because it is ServiceNow's data-import framework for bringing external data into the platform from sources such as spreadsheets, CSV files, XML files, JDBC data sources, and integrations. An import set first loads the incoming records into a staging table, which allows the source data to be reviewed and processed without immediately writing it to the destination production table. The import process then uses a transform map to define how fields in that staging table correspond to fields in a target ServiceNow table. Transform maps can also apply field mappings, coalesce logic to identify existing records, scripts, and other processing rules while records are created or updated. In practical use, an import set and its associated transform map work together: the import set receives and stages the data, while the transform map performs the mapping into the selected ServiceNow table. This makes Import Sets the appropriate ServiceNow tool for the end-to-end task described in the question. ServiceNow documentation describes the process and its staging-table model in the Import Sets documentation and explains field and record processing in the Transform Maps documentation.

QUESTION NO: 4

What is the ServiceNow App Repository?

A. A Request table

B. Another name for update sets

C. A database containing ServiceNow apps

D. A collection of files in a Git database

ANSWER: C

Explanation:

A database containing ServiceNow apps is the best description of the ServiceNow App Repository. More specifically, the App Repository is ServiceNow's centralized, instance-hosted repository for scoped applications. When an application developer publishes an application to the repository, ServiceNow stores the application and its version information so that authorized target instances can discover, install, and update it. This supports controlled distribution of custom scoped applications between ServiceNow instances without requiring the application to be manually recreated on each instance.

The repository also tracks application versions and application entitlements. Entitlements identify which customer instances are authorized to install a published application, while versioning lets developers publish updates and lets consumers obtain the appropriate released version. In practical application-development work, this makes the App Repository a distribution and lifecycle-management mechanism for ServiceNow applications. It is the platform location where published applications are stored for installation and upgrade, which is accurately captured by the description of a database containing ServiceNow apps. ServiceNow's application-development documentation describes publishing applications to the Application Repository and installing applications from it: Application Repository documentation. Additional developer learning material is available at Publish an application to the Application Repository.

QUESTION NO: 5

Which of the following methods is NOT part of the ServiceNow REST API?

A. COPY

B. GET

C. DELETE

D. POST

ANSWER: A

Explanation:

COPY is correct because it is not an HTTP method exposed by ServiceNow REST APIs, including the Table API. ServiceNow REST endpoints use standard HTTP request methods to express an operation on a resource. For example, a client retrieves records or a specific record through a read request, creates a record by submitting a request body, updates a record using an update request, and removes a record with a delete request. A request to duplicate information is not represented by a standalone **COPY** verb in the ServiceNow REST API contract.

When an integration needs to create a duplicate of an existing record, it normally first retrieves the source record, then sends a new creation request with the desired field values. This keeps the action consistent with ServiceNow's resource-oriented REST design and allows business rules, data policies, access controls, and other server-side processing to apply to the newly created record. The ServiceNow Table API documentation lists the supported operations and their corresponding standard HTTP methods; **COPY** is not among them. See the ServiceNow Table API reference and the HTTP request methods reference.

QUESTION NO: 6

Which client-side scripts apply to Record Producers? (Choose 2 answers)

A. Fix Scripts

B. Catalog Client Scripts

C. Catalog UI Policies

D. Record Producer Policies

E. UI Scripts

ANSWER: B C

Explanation:

Catalog Client Scripts apply to record producers because a record producer is a Service Catalog item type. They run in the user's browser while the producer form is being used and can respond to events such as loading the form, changing a variable, or submitting the producer. This enables developers to validate variable values, set values dynamically, and improve the requester experience before the record producer creates its target-table record.

Catalog UI Policies also apply to record producers. A catalog UI policy provides declarative, client-side control over catalog variables, including making a variable mandatory, read-only, or hidden based on conditions. Its associated UI policy actions are evaluated on the record producer form, allowing the producer to present only the variables that are relevant for the requester's selections. Together, these mechanisms provide both scripted and configuration-based client-side behavior for record producers without requiring customization of the target record's standard form.

ServiceNow's catalog development guidance describes the client-side behavior available for catalog items and record producers, including catalog client scripts and catalog UI policies: ServiceNow Catalog Client Scripts documentation and ServiceNow Catalog UI Policies documentation.

QUESTION NO: 7

Which framework can automatically populate values for the Priority and Category fields based on the Short description field value?

A. Action

B. Assignment Rule

C. Predictive Intelligence

D. CSDM

E. UI Policy

ANSWER: C

Explanation:

Predictive Intelligence is the ServiceNow machine-learning framework designed to derive field values from patterns in historical records. A classification solution can be trained with an input field such as Short description and target fields such as Category and Priority. After the model is trained, evaluated, and deployed, ServiceNow can use the text entered in a new record's Short description to generate predictions and automatically apply the predicted values to those target fields. This supports more consistent, faster record categorization and prioritization while allowing the model to improve through retraining as additional labeled records become available. The solution depends on sufficiently representative historical data, because the model learns the relationship between the wording in prior short descriptions and the values that were ultimately used for Category and Priority. ServiceNow describes Predictive Intelligence as its platform capability for using machine learning to make predictions and classifications from instance data. See the ServiceNow Predictive Intelligence product overview and the ServiceNow Predictive Intelligence documentation.

QUESTION NO: 8

Which one of the following objects CANNOT be used in a Script Action script?

A. previous

B. GlideRecord

C. event

D. current

ANSWER: A

Explanation:

`previous` cannot be used in a Script Action script. A Script Action runs asynchronously in response to an event that has been placed on the ServiceNow event queue. At that point, the script has access to the event context and to the record represented by `current`, but it does not execute as part of the original database transaction that updated the record. The `previous` object is specifically a before-update snapshot used by synchronous server-side logic, such as Business Rules, where ServiceNow is processing both the record's old and new values in the same transaction. Because an event is handled later, that prior in-transaction record state is not available to the Script Action. If a script needs state from when the event was queued, the appropriate design is to pass the required value in one of the event parameters when calling `gs.eventQueue()`, then retrieve it from the event object in the Script Action. ServiceNow's Script Action documentation describes the event-driven execution context and available objects, while the Business Rule documentation explains the use of `previous` during record operations. See ServiceNow Script Actions documentation and ServiceNow Business Rules documentation.

QUESTION NO: 9

Why create Applications in ServiceNow?

A) To replace outdated inadequate custom business applications and processes

B) To extend service delivery and management to all enterprise departments

C) To allow users full access to all ServiceNow tables, records and fields

D) To extend the value of ServiceNow

A. a b and c

B. a b c and d

C. b c and d

D. a b and d

ANSWER: D

Explanation:

“A, B, and D” is correct because ServiceNow applications are created to address business needs on the Now Platform and to broaden the platform’s usefulness across an organization. A custom application can modernize or replace an inadequate legacy business process by providing structured data, workflows, experiences, automation, and integrations in a governed platform. It can also bring service delivery and management practices to departments beyond traditional IT, enabling teams such as HR, facilities, legal, finance, or workplace services to build and manage purpose-built workflows. In doing so, applications extend the value of an organization’s ServiceNow investment: they reuse shared platform capabilities, including security controls, data services, reporting, automation, and user experiences, rather than requiring separate disconnected tools. ServiceNow App Engine is specifically intended to support the development of these enterprise workflow applications, from low-code solutions to more advanced custom development. Application access is designed around roles, permissions, and scoped security, so the purpose of creating an application is not unrestricted access to platform data. See ServiceNow’s App Engine overview and the ServiceNow Developer learning plans for the platform approach to building business applications.

QUESTION NO: 10

Which methods can be used to install an application on a ServiceNow instance?

Choose 3 answers

A. Import an application from an XML file

B. Use the ' Install ' button on the application record

C. Install from the Google Play Store

D. Download from Stack Overflow

E. Install an application from the Application Repository

F. Download and install a third-party application from the ServiceNow Store

ANSWER: A E F

Explanation:

ServiceNow supports application installation through application-package import, the Application Repository, and the ServiceNow Store. “Import an application from an XML file” is a supported approach for bringing an exported application package into an instance. An administrator can use the application import capability to upload the XML package, review its contents, and install the application and its associated records. This is useful when an application is distributed directly rather than retrieved from an online catalog.

“Install an application from the Application Repository” is also correct because the repository is ServiceNow’s mechanism for publishing application versions from a development instance and installing those published versions on other authorized instances. It supports governed promotion and version management for scoped applications. “Download and install a third-party application from the ServiceNow Store” is correct because the Store provides ServiceNow and partner-published applications that customers can obtain and install, subject to entitlement, role, and instance requirements. These methods are documented installation channels for ServiceNow applications. See the ServiceNow Application Repository documentation and the ServiceNow Developer learning material on installing applications.

QUESTION NO: 11

For your implementation the following tables are extended from each other:

• Incident table is extended from Task table

• Super Incident table is extended from Incident table

In this situation, which table(s) are Parent. Child and Base tables?

Choose 5 answers

A. Incident table is a Child table

B. Incident table is a Parent table

C. Task table is a Child table

D. Super Incident table is a Child table

E. Super Incident table is a Parent table

F. Task table is a Base table

G. Task table is a Parent table

ANSWER: A B D F G

Explanation:

Table extension creates an inheritance hierarchy. A table that extends another table is its child, while the table being extended is its parent. Therefore, the Incident table is a Child table because it extends the Task table. The Incident table is also a Parent table because the Super Incident table extends it. This illustrates that an intermediate table in a hierarchy can simultaneously be a child of one table and a parent of another.

The Super Incident table is a Child table because it inherits from Incident. It inherits the fields, behavior, and configuration available through both Incident and its ancestor, Task. The Task table is a Parent table because Incident directly extends it. It is also the Base table for the hierarchy described because it is the top-level table from which Incident and Super Incident inherit in this scenario. ServiceNow's table-extension model supports shared fields and logic across related record types while allowing child tables to add their own specialized fields and behavior. See the ServiceNow documentation on extending tables and data modeling.

QUESTION NO: 12

Many actions are included with flow designer what are some frequently used core actions?

Choose 4 answers

A. Look Up Record

B. Ask for Approval

C. Create Record

D. Look for Update

E. Wait for Condition

F. Wart for Match

ANSWER: A B C E

Explanation:

Flow Designer core actions provide reusable, no-code building blocks for common automation steps. **Look Up Record** is a frequently used data action because a flow commonly needs to retrieve a single record from a selected table before evaluating its fields or using its values later in the flow. **Create Record** is another core action used to insert a new record into a chosen table, such as creating a task, request-related record, or other business record as part of an automated process.

Ask for Approval supports an approval stage in a flow by sending an approval request and allowing the flow to continue based on the approval outcome. This is a core workflow need for processes involving governance, purchasing, access, or change-related decisions. **Wait for Condition** pauses flow execution until a record meets configured criteria, enabling the automation to wait for a meaningful state or field change without requiring custom polling logic. Together, these actions cover common record retrieval, record creation, approval, and conditional-wait requirements in Flow Designer. ServiceNow documents Flow Designer as its process-automation capability and describes the available action types and flow logic in its product documentation: ServiceNow Flow Designer documentation and ServiceNow Developer learning plans.

QUESTION NO: 13

When creating application tables, a user role is automatically added to the table record. Which other role does an application typically have?

A. Application Manager

B. Application Fulfiller

C. Application Super User

D. Application Admin

ANSWER: D

Explanation:

Application Admin is correct because a scoped application commonly uses an administrative role in addition to the application user role associated with its tables. The application admin role is intended for developers or designated application administrators who must maintain the application and manage its configuration. It provides the level of access needed to administer application resources, including the application's tables and related artifacts, while access to individual records and operations can still be governed through table-level access controls and roles.

In ServiceNow application development, roles are a core mechanism for controlling who can use an application and who can administer its underlying components. An application's administrative role conventionally follows the application scope naming pattern and is used to grant administrative capabilities within that application. This design supports separation between ordinary application users and personnel responsible for configuring, maintaining, and extending the app. ServiceNow's application-development guidance describes how application roles are used to secure application resources and how access controls work with those roles. See the ServiceNow documentation on application roles and the ServiceNow Developer site for application-development learning resources.

QUESTION NO: 14

Which one of the following is NOT part of the Form Designer?

A. Form layout

B. Page header

C. Schema map

D. Field navigator

ANSWER: C

Explanation:

Schema map is not part of the Form Designer. Form Designer is the ServiceNow interface used to configure the presentation and arrangement of a table's form, including the form layout, sections, fields, and header-related form configuration. Its purpose is to help an administrator or developer control what users see when opening a record and how those elements are organized on the form.

A schema map is a separate platform visualization tool. It displays table relationships, such as parent/child table extension and reference relationships, to help developers understand the data model and navigate related tables. Because it is focused on the database schema rather than the composition of an individual record form, it is accessed independently and is not a Form Designer component. This distinction is important in application development: use Form Designer when configuring the user-facing record experience, and use Schema Map when investigating table structure and relationships. ServiceNow's form administration documentation describes form configuration capabilities, while its schema-map documentation identifies the tool's role in visualizing table relationships. See ServiceNow Form Designer documentation and ServiceNow Schema Map documentation.

QUESTION NO: 15

In a privately-scoped application, which methods are used for logging messages in server-side scripts?

Choose 2 answers

A. gs.debug()

B. gs.logError()

C. gs.error()

D. gs.warn()

E. gs.log()

ANSWER: A C D

Explanation:

For server-side scripts in a privately scoped ServiceNow application, `gs.debug()`, `gs.error()`, and `gs.warn()` are supported GlideSystem logging methods. `gs.debug()` records diagnostic details useful while developing or troubleshooting an application. `gs.error()` records an error condition that requires attention and is appropriate for exceptions, failed operations, or other significant processing failures. `gs.warn()` records a warning when processing can continue but an unusual or potentially problematic condition should be visible to administrators and developers.

These methods write messages using ServiceNow logging levels, allowing entries to be reviewed in the system log and filtered according to severity. The question's instruction to choose two is incomplete because all three listed methods are valid for logging from a scoped server-side script. In practice, select the method whose severity accurately represents the event rather than using errors for routine diagnostic output. ServiceNow documents the scoped GlideSystem API, including the logging methods and their availability, in the GlideSystem API reference. Additional guidance on application scope and cross-scope behavior is available in the Application scope documentation.

QUESTION NO: 16

Which of the following is true about g_scratchpad?

Choose 2 answers

A. Has default properties passed by client-side scripts

B. Does not exist on the mobile platform

C. Used to push information from the server to the client

D. Has constructors and methods

ANSWER: B C

Explanation:

Used to push information from the server to the client is correct because `g_scratchpad` is a global client-side object populated by a Display Business Rule that runs on the server. A developer assigns values to `g_scratchpad` in the Display Business Rule, and those values are then available when client-side scripts execute as the form loads. This provides a controlled way to make server-derived values available to Client Scripts without an additional asynchronous server call. ServiceNow documents Display Business Rules as a mechanism for making server-side data available to client-side scripts through the scratchpad object. See the ServiceNow Display Business Rules documentation.

Does not exist on the mobile platform is also correct in the context of the traditional ServiceNow mobile client environment covered by this exam topic. The scratchpad is associated with the standard browser form-loading lifecycle and its client scripting model; mobile experiences do not expose this form global in the same way. Developers implementing mobile behavior should use the APIs, data resources, and client logic supported by the applicable mobile framework rather than relying on `g_scratchpad`. For the server-to-client use pattern and the client scripting context in which the scratchpad is available, refer to the ServiceNow Developer site's Business Rules learning material.

QUESTION NO: 17

How does ServiceNow match inbound email to an existing record? (Choose 2 answers)

A. Sys_id

B. Record link

C. Watermark

D. Subject line with record number

ANSWER: C D

Explanation:

ServiceNow uses a watermark to reliably associate a reply or forwarded email with the record that generated the original outbound notification. A watermark is a unique identifier inserted into outbound email, typically in the message body, and inbound email processing reads it to locate the associated task or record. This provides a dependable correlation method even when the subject text has been changed by the sender or their email client.

ServiceNow can also match an inbound message through a record number in the subject line, such as an incident number. When an inbound email subject contains a recognized record number, ServiceNow can use that number to identify the target record and process the message in its context. These matching mechanisms support inbound actions that update an existing record, add journal entries, or perform other configured processing. See the ServiceNow documentation on email watermarks and inbound email actions.

QUESTION NO: 18

When building an extended table from a base table, which fields do you need to create?

Choose 2 answers

A. The reference fields for the base table.

B. The fields that are not in the base table

C. The mandatory fields for the base table

D. The fields that are specific to the extended table

ANSWER: B D

Explanation:

When a table extends another table, it inherits the base table's existing fields, including its standard fields, reference fields, and any mandatory fields defined on the parent. The child table therefore needs only its additional data model: fields that are not in the base table and fields that are specific to the extended table. These describe the attributes required for records of the specialized child-table type but not for every record represented by the parent table. For example, a child table can add fields that capture details unique to its particular business purpose while retaining access to the inherited parent fields. This inheritance model promotes a consistent schema, avoids duplicating common fields, and lets applications use the child table wherever the parent-table structure is appropriate. ServiceNow describes table extension as a way for a child table to inherit fields and behavior from its parent while allowing child-specific additions. See the ServiceNow documentation on extending tables and the ServiceNow data-modeling documentation.

QUESTION NO: 19

Which GlideForm method removes a message displayed beneath a specific form field?

A. g_form.hideFieldMsg()

B. g_form.clearMessages()

C. g_form.removeInfoMessage()

D. g_form.clearFieldValue()

ANSWER: A

Explanation:

`g_form.hideFieldMsg()` removes a message that was displayed beneath a specified field. It is commonly used after a Client Script has corrected a validation condition or when an onChange Client Script needs to clear feedback previously shown with `g_form.showFieldMsg()`. Field messages are associated with an individual field, unlike form-level messages created with methods such as `g_form.addInfoMessage()`. See the ServiceNow GlideForm API reference.

QUESTION NO: 20

Which of the following objects does a Before Business Rule have access to in a script?

A. current

B. previous

C. GlideRecord

D. All of the above

ANSWER: D

Explanation:

All of the above is correct. A Before Business Rule runs on the server before the database operation is completed, so its script receives the `current` GlideRecord representing the record being inserted, updated, deleted, or queried. The script can inspect field values and, for insert or update operations, modify values on `current` before ServiceNow writes the record. For an update operation, the `previous` object provides the values that existed before the update, enabling the rule to compare the prior and proposed record states. Business Rule scripts are also server-side JavaScript, where the GlideRecord API is available. A developer can therefore instantiate and use GlideRecord objects to query, read, update, insert, or otherwise work with records in other tables when that is appropriate for the business logic. These capabilities make Before Business Rules particularly useful for validation, normalization, and setting field values before a record is saved. ServiceNow documents the `current` and `previous` Business Rule objects and their use in server-side rule scripts in its Business Rules documentation: ServiceNow Business Rules. GlideRecord server-side usage is documented at ServiceNow GlideRecord API.

QUESTION NO: 21

Which one of the following is the correct Link Type to select when creating a module that opens the Record Producer UI for a user rather than the ServiceNow form UI?

A. URL (from Arguments:)

B. HTML (from Arguments:)

C. Script (from Arguments:)

D. Content Page

ANSWER: A

Explanation:

URL (from Arguments:) is the appropriate link type because a navigation module can use its Arguments field to supply the URL that launches the intended record producer experience. A record producer is a catalog-style interface that collects user input and then creates a record in a target table, so it should be launched through its producer or catalog URL rather than by navigating directly to a standard table form. This lets the module present the producer's configured variables, layout, client behavior, and submission processing to the user. The module record remains part of the application navigator, while the URL and any required parameters are maintained in the module arguments. This approach is particularly useful when an application needs a direct, user-friendly entry point for creating records through a producer, without exposing the underlying platform form as the initial interface. ServiceNow describes modules as navigation items that can open URLs and other content, and its record-producer guidance explains that producers provide a catalog interface for creating records. See ServiceNow application modules documentation and ServiceNow record producers documentation.

QUESTION NO: 22

What happens if a Record Producer script aborts the record generation process in ServiceNow?

A. Record is not generated in the Item Produced Record table

B. Record creation is redirected to a confirmation page

C. Record creation displays an error message to the user

D. Record creation terminates without notifying the user

ANSWER: A

Explanation:

Record is not generated in the Item Produced Record table is correct. A Record Producer collects catalog input and uses it to generate a record on its configured target table. Its server-side script can stop that database operation, commonly by calling `current.setAbortAction(true)` on the record being produced. When the action is aborted, the generated-record workflow does not complete: the target record is not inserted, and ServiceNow does not create the corresponding produced-record tracking relationship in the Item Produced Record `[sc_item_produced_record]` table. That table is used to associate a Record Producer submission with the record it successfully produced, so an aborted generation has no created record to track.

Aborting the action controls whether ServiceNow performs the insert; it is not itself a user-interface response. If an implementation needs to inform the requester about the condition that caused the abort, the producer script should explicitly add an appropriate message before stopping the action. This behavior follows the server-side record API's abort-action capability and the Record Producer model, in which catalog input is mapped to a record created in a target table. See the ServiceNow GlideRecord API reference and Record Producer documentation.

QUESTION NO: 23

Which of the following are valid uses for a Display Business Rule? (Choose two.)

A. Populate values in g_scratchpad for a Client Script

B. Query server-side data before a form is rendered

C. Prevent a record from being saved by the user

D. Set a field to mandatory in the browser

ANSWER: A B

Explanation:

A Display Business Rule runs on the server before a form is sent to the browser. It can place server-calculated values in `g_scratchpad` for use by client-side scripts when the form loads. It can also use server-side APIs, such as GlideRecord, to determine data needed for the initial client experience. A Display Business Rule is not used to update a record as part of a user save operation, and it does not execute in the browser. See the ServiceNow Business Rules documentation.

QUESTION NO: 24

Which actions can be configured in a UI Policy Action? (Choose three.)

A. Make a field mandatory

B. Make a field read-only

C. Make a field visible

D. Set a field default value

ANSWER: A B C

Explanation:

UI Policy Actions can set a form field to be **mandatory**, **read-only**, or **visible**. A UI Policy evaluates conditions in the browser and then applies its actions to guide users through a form without requiring custom client-side scripting for common presentation requirements. For example, a field can become mandatory and visible only when a selected choice indicates that additional information is required. UI Policy Actions do not define the field's default value; that behavior can be configured through dictionary defaults, Client Scripts, or other applicable logic. See the ServiceNow UI Policy Actions documentation.

QUESTION NO: 25

When creating an application through the Guided Application Creator, what are the options for creating a table? (Choose 3 answers)

A. Link to external tables

B. Create a table from scratch

C. Create a table from a template

D. Run import jobs

E. Upload a spreadsheet

F. Use API calls

G. Extend a table

ANSWER: B E G

Explanation:

Guided Application Creator supports three practical paths for defining an application table: creating a table from scratch, extending a table, and uploading a spreadsheet. “Create a table from scratch” is used when the application requires a wholly new data structure. The developer supplies the table’s label and configuration, then defines the fields needed to store the application’s records.

“Extend a table” is appropriate when the new records should inherit fields, behavior, and relationships from an existing platform table. Table extension is a central ServiceNow data-modeling capability and lets an application reuse common record attributes while adding fields that are specific to the new business purpose.

“Upload a spreadsheet” is also available during guided creation as a fast way to begin with a data model based on existing tabular data. The spreadsheet’s column headings can be used to establish fields for the new table, reducing manual setup when an organization already has a structured list of the information it needs to manage. ServiceNow’s developer learning material describes using Guided Application Creator to create an app and its data model, while the platform documentation explains table creation and extension concepts. See ServiceNow Developer: Build My First Application and ServiceNow Documentation: Data modeling.

QUESTION NO: 26

Identify characteristic(s) of a Record Producer. (Choose three.)

A. All records created using this strategy are inserted into the Requested Item [sc_req_item] table.

B. Each field prompts the user with a question rather than a field label.

C. They must be scripted.

D. You can script behaviors of fields in the user interface.

E. Graphics can be included on the user interface.

ANSWER: B D E

Explanation:

A record producer is a Service Catalog item type that presents a catalog-style form to collect information and then creates a record in a target table. Its variables are displayed to requesters as questions or prompts, rather than as the standard field labels found on a direct table form. This enables an application developer to provide a requester-friendly experience while mapping submitted variable values to fields on the record being created.

You can script behaviors of fields in the user interface. Record producers support catalog client scripts and catalog UI policies, allowing developers to dynamically control variable visibility, mandatory state, read-only state, and values while the requester completes the form. Server-side record-producer scripting is also available when custom processing is needed, but the form itself can use configuration and client-side behavior.

Graphics can be included on the user interface. Record producers use the Service Catalog presentation framework, which supports rich catalog content and variable types that can display informational content, including images, to improve the requester experience. These characteristics make record producers suitable for creating records through a guided, branded, and user-friendly catalog interface. See the ServiceNow documentation on record producers and catalog client scripts.

QUESTION NO: 27

What data types of Flow Variables are supported to store record data?

Choose 3 answers

A. Array

B. Choice

C. Integer

D. String

E. Date

ANSWER: A C D

Explanation:

The supported Flow Variable data types for storing record-related values in this context are **Array**, **Integer**, and **String**. A String variable holds textual values, such as a record number, short description, sys_id represented as text, or another field value returned by an action. An Integer variable holds whole-number values, such as a count, sequence value, or numeric field value used later in flow logic. An Array variable stores an ordered collection of values, making it useful when an action or lookup returns multiple records or when the flow must retain a list for subsequent processing, such as iterating through returned items.

Flow variables allow a flow to preserve values during execution so they can be reused as data pills in later actions, conditions, and flow logic. Selecting the appropriate storage type helps ensure the value can be passed reliably between flow steps and processed in the expected form. ServiceNow's Flow Designer documentation describes flow construction and the use of data pills and flow logic: ServiceNow Product Documentation. For hands-on Flow Designer development concepts, see the official ServiceNow Developer Site.

QUESTION NO: 28

Which role(s) are required to impersonate a user?

Choose 2 answers

A. security_admin

B. sys_admin

C. admin

D. sys_user

E. impersonator

ANSWER: C E

Explanation:

The `admin` role permits an administrator to use the platform's impersonation capability. This is commonly used for administrative testing and troubleshooting because it lets the administrator experience the instance with the target user's roles, access controls, and user-specific interface settings. ServiceNow records impersonation activity, and an impersonating session is clearly indicated in the user interface so that administrative actions can be performed in the appropriate context.

The `impersonator` role is the dedicated role for users who need permission to impersonate without being granted the broad administrative permissions included with `admin`. Assigning this role supports least-privilege administration: a designated support or testing user can validate another user's experience while retaining only the access independently granted to their own account outside of impersonation. In both cases, the ability to select a particular target user can also be constrained by platform security settings and impersonation policies. ServiceNow's product documentation describes impersonation as an administrator or authorized impersonator function, and its security guidance emphasizes controlling elevated access and auditing administrative activity. See ServiceNow: Impersonate a user and ServiceNow: User impersonation.

QUESTION NO: 29

On a form, which type of field has this icon which can be clicked, to see a preview of the associated record?

A. Reference

B. Lookup

C. Preview

D. Quickview

E. Drilldown

F. Snapshot

ANSWER: A

Explanation:

Reference is correct. A reference field stores a relationship to a record in another table, rather than merely storing free-form text. On a form, ServiceNow displays the referenced record's display value and provides reference controls that let a user search for, select, open, or inspect the related record. One of these controls is the preview (reference preview) icon; selecting it opens a compact preview of the record associated with the current reference value. This supports quick review of related information without navigating away from the form or loading the full target record in the primary workspace. For example, an Incident's Caller field references a User record, so its preview control can show selected details from that Caller record. The precise fields shown in the preview can be configured through the platform's reference-preview behavior and dictionary/form configuration. ServiceNow defines reference fields as fields that contain a reference to a record on another table, which is the relationship required for the platform to retrieve and preview the associated record. See ServiceNow's Reference fields documentation and reference field user-interface documentation.

QUESTION NO: 30

What are the ways to designate data tables when Guided Application Creator (GAC)?

Choose 3 answers

A. Upload an existing PDF

B. Create a new table on the platform

C. Use an existing table on the platform

D. Upload an existing spreadsheet

E. Upload an existing word processing document.

F. Use a freeform database

ANSWER: B C D

Explanation:

Guided Application Creator supports three practical ways to establish the data model for a scoped application. **Create a new table on the platform** is appropriate when the application requires a purpose-built record type and fields that do not already exist. The creator can define the table as part of the application-building flow, keeping the data structure within the application scope. **Use an existing table on the platform** supports reuse of an established table when it already represents the records the application needs to manage. This enables the application to work with the platform's existing data model rather than duplicating records in a new table. **Upload an existing spreadsheet** lets a builder start with spreadsheet data and use it as the basis for application data; the imported columns can be used to form the table structure and populate records. These choices align with the platform's table-based application model and its spreadsheet import capabilities. See ServiceNow's application development documentation and its Import Sets documentation for the underlying table and import concepts.

QUESTION NO: 31

Which of the following statements must evaluate to true for a user to pass an Access Control?

Choose 3 answers

A. Other matching Access Controls for the records evaluate to true.

B. Conditions configured in the Access Control must evaluate to true.

C. The user must be granted access through a business rule.

D. The user has one of the roles specified in the Required roles related list.

E. Scripts configured in the Access Control must evaluate to true.

ANSWER: B D E

Explanation:

Conditions configured in the Access Control must evaluate to true because the condition builder defines record- or context-specific criteria that must be satisfied before the rule grants the requested operation. This lets an administrator restrict access based on field values, user attributes, or other available data.

The user has one of the roles specified in the Required roles related list because roles are an ACL requirement when roles have been configured. ServiceNow evaluates the role requirement as satisfied when the user holds at least one role listed for that access control rule; users can receive those roles directly or through role inheritance.

Scripts configured in the Access Control must evaluate to true because an ACL script is an additional programmatic authorization test. The script must set the access-control answer to a true result for the rule's script requirement to pass. Together, configured conditions, role requirements, and scripts form the ACL rule's authorization criteria; each configured criterion must pass for that ACL to grant access. ServiceNow documents ACLs as security rules that control access to data and operations, with conditions, required roles, and scripts available to define the rule. See the ServiceNow documentation on access control rules and the ACL rule evaluation documentation.

QUESTION NO: 32

You are developing the MyApp application that has a table, Table A. When the MyApp application is installed on another instance, you want Table A ' s records to be installed as part of the application. Table A ' s records will be installed when:

A. Table A is active and extends the Task table.

B. Table A has an automatic number counter for new records and the table property " Include data " is set to true.

C. Table A ' s records are added to the application record using the " Create Application Files " feature.

D. Table A is not included in the System Clone > Exclude Tables list.

ANSWER: C

Explanation:

Table A ' s records are added to the application record using the " Create Application Files " feature. In a scoped ServiceNow application, an application is distributed as a set of application files (records in the application's update payload). Creating an application file for a particular Table A record explicitly makes that record part of the application artifact. Consequently, when the application is installed or upgraded on a target instance, ServiceNow processes that application file and delivers the included record along with the table definition, scripts, UI components, and other application resources.

This approach is commonly used for application seed data: records an application needs in order to operate consistently after installation, such as predefined configuration, reference values, or sample records. The developer selects the desired records and uses the Create Application Files capability so the records are tracked by the application rather than existing only as local instance data. This also permits the records to be versioned and transported through normal application publishing and installation workflows. ServiceNow describes application files as the records that comprise an application and are captured for deployment; see ServiceNow Application Files documentation and the ServiceNow Developer learning module on creating application files.