**Servicenow System Administrator Training**

**Platform Overview and Architecture**

* **Servicenow:** American company providing a cloud-based application platform (APaaS) for automating business processes.
* **Usage:** IT, HR, finance, security, and more.
* **Architecture:**
  + **Multi-Instance vs. Multi-Tenant:** Multi-instance means isolated data and applications per organization; multi-tenant means shared data among customers.
  + **Backup:** Four weekly full backups and six daily differential backups.
  + **Security:** Single sign-on (SAML 2.0), role-based access.

**Platform Interface**

* **User Interface (Native UI):** Main way to interact; includes real-time updates, activity streams, and application navigator.
* **Mobile Apps:**
  + Servicenow Agent
  + Now Mobile
  + Servicenow Onboarding
* **Service Portal:** Advanced UI for self-service; customizable widgets; accessible via /sp in URL.

**Supported Authentication**

* **Local Database:** Username and password stored in Servicenow.
* **Single Sign-On (SSO):** Uses identity providers like Okta.
* **LDAP:** Lightweight Directory Access Protocol.
* **OAuth 2.0:** Open authentication with client ID and secret.
* **Digest Token:** Encrypted username and password.
* **Multi-Factor Authentication (MFA):** OTP via Google Authenticator.

**Role-Based Access**

* **Roles:** Collection of permissions assigned to users or groups.
* **Users:** Individuals granted access to Servicenow.
* **Groups:** Sets of users with similar responsibilities.
* **Role Assignment:** Ensures users access only information they are permitted to.

**Tables and Fields**

* **User Table:** Stores user data.
* **Group Table:** Stores group data; both tables are part of out-of-the-box features.

**Group Record Details**

* **Group Name:** Application Development
* **Details:**
  + **Manager:** Displayed
  + **Roles:** Assign roles to the group to control access.
  + **Group Membership:** Lists members in the group.
  + **Child Groups:** Any groups under this group.
  + **Skills and Locations:** Assign as needed.

**Roles**

* **Overview:** Out-of-the-box table storing roles for access control.
* **Navigation:** System Security > Roles module.

**Assigning Roles**

* **Best Practice:** Assign roles to groups rather than individual users for easier maintenance.
* **Example:** If Group X has roles Role X and Role Y, members of Group X get all permissions from both roles.

**Base System Roles**

* **Examples:**
  + **Admin:** Full platform access.
  + **Approver\_User:** Task approval access.
  + **ITIL:** Access to ITSM applications.
  + **Catalog\_Admin:** Service catalog management.

**Self-Service Users**

* **Access:** Limited to basic functionalities like home page, incident/request raising, and service catalog.
* **Impersonation:** Admins can impersonate users to see their access level.

**Impersonation Example**

* **Creating a User:**
  + **User ID:** Demo User
  + **No Roles or Groups Assigned.**
* **Impersonation Steps:**
  + **Admin View:** Full access to applications.
  + **Self-Service User View:** Limited access, basic dashboard.
  + **ITIL User View:** Broader access, more modules.

**User Interface (UI)**

* **Login Screen:** Username and password required.
* **Native UI:** Main interface after login.
* **UI Versions:**
  + **UI 16** (default)
  + **UI 15** (has additional Edge element)
  + **Switching:** Settings > Switch UI Version

**UI Elements**

* **Banner Frame:** Contains logo, user menu, global search.
* **Application Navigator:**
  + **All Applications:** Complete list of accessible applications.
  + **Favorites:** Starred applications and modules.
  + **History:** Recently visited items.
* **Content Frame:** Displays selected application or module information.
* **Edge (UI 15 only):** Additional UI element in UI 15.

**Additional UI Features**

* **User Menu:** Profile, Impersonate User, Elevate Roles, Log Out.
* **Global Search:** Allows searching across the platform.
* **Role Elevation:** Temporary access to higher roles.

**Global Search**

* **Function:** Searches all tables in the Servicenow system based on the keyword entered.
* **Admin vs. End User:** Admins see more results due to broader access. End users see results based on their table access permissions.

**Chat Option**

* **Usage:** Allows communication with other users within the organization who are on Servicenow.
* **Location:** Available in the main interface.

**Help and Settings**

* **Help Icon:** Provides assistance and information.
* **Settings:**
  + **General Settings:** Includes theme customization.
  + **Accessibility Options:** Various accessibility configurations.
  + **List and Form Configurations:** Adjustments for list views and form notifications.
  + **Developer Settings:** For admin access, includes development-related configurations.

**Application Navigator**

* **Filter Navigator:** Type in to search for applications or modules.
* **Favorites Tab:**
  + **Tagging Favorites:** Mark entire applications or specific modules as favorites for quick access.
  + **Managing Favorites:** Add or remove favorites as needed.
* **History Tab:**
  + **Function:** Tracks recent views and activities.
  + **Usage:** Helpful for reviewing recent actions without searching again.

**Creating and Using Modules**

* **Creating Records:** Use modules like "Create New" to start new records.
* **Viewing Records:** Access and manage existing records through various modules.

**Branding Overview**

* **Purpose:** Customize the instance to align with company branding.
* **Configuration:**
  + **UI 16 Configuration:** Change instance name, banner image, and other settings.
  + **Welcome Page Settings:** Customize welcome page content.

**Lists and Filters**

* **List Components:**
  + **Main List:** Displays records from a table.
  + **Title Bar, Filters, Breadcrumbs, Columns, List Title Menu:** Various elements to manage and navigate lists.
  + **List Context Menu:** Options for sorting, exporting, and manipulating list data.
* **Filters and Personalization:**
  + **Applying Filters:** Set conditions to view specific records.
  + **Creating and Saving Filters:** Save frequently used filters for convenience.
  + **Changing Page Size:** Adjust the number of records displayed per page.

**Views**

* **Purpose:** Customize how data is displayed in lists.
* **Examples:** Major Incidents View, Mobile View.
* **Usage:** Modify views to display specific fields or data based on requirements.

**Saving and Applying Filters:**

1. **Save a Filter:**
   * Apply your desired filter conditions.
   * Click the save button.
   * Name your filter and choose visibility options (e.g., visible to me, everyone, or specific groups).
   * Click "Save."
2. **Apply a Saved Filter:**
   * Click on filters.
   * Select your saved filter from the bottom of the list.
   * The filter conditions will be applied automatically.

**Grouping Data:**

1. **Group By:**
   * You can group data by fields (e.g., State) using options available in the list view or by right-clicking on a column header and selecting "Group By."

**Refreshing and Managing Lists:**

1. **Refresh List:**
   * Use the refresh option to update the list view based on any changes made to the form or data.
2. **Favorites:**
   * Save a favorite view of your list with options to name it, add a color, and an icon.
   * Access saved favorites from the Application Navigator.

**List Context Menu:**

1. **Sort and Group:**
   * Sort data (e.g., alphabetically by short description).
   * Group data by specific fields or ungroup.
2. **Charts and Configurations:**
   * Create bar or pie charts directly from the list.
   * Configure list settings and layout (e.g., remove or add columns).

**Import and Export:**

1. **Import Data:**
   * Import data into the list from various formats (e.g., XML).
2. **Export Data:**
   * Export list data in formats like Excel, CSV, XML, or PDF.

**Updating Records:**

1. **Update Selected Records:**
   * Select records and use the "Update Selected" option to make changes to multiple records simultaneously.
2. **Update All:**
   * Update all records currently displayed in the list.

**Searching and Personalizing Lists:**

1. **Search:**
   * Use the search option to find specific records.
   * Searches can be part of filter conditions.
2. **Personalize List Columns:**
   * Add or remove fields from your list view via the "Personalize List Columns" option.

**Forms Overview:**

1. **Form Structure:**
   * Includes content frame, form title, form buttons, and menu options.
2. **Creating and Editing Records:**
   * Use "Create New" to add records.
   * Difference between "Save" (keeps the form open) and "Submit" (closes the form).
3. **Mandatory and Read-Only Fields:**
   * Mandatory fields require input to save or submit a record.
   * Read-only fields cannot be edited by users.
4. **Field Types:**
   * **Choice Fields:** Provide dropdown options.
   * **Reference Fields:** Pull data from other tables (e.g., Caller, Assignment Group).

**Overview of how to work with lists and forms in ServiceNow, including:**

* **Saving and Applying Filters:** How to create and apply saved filters, group data, and use favorites.
* **List Context Menu:** Options for sorting, grouping, charting, and exporting data from lists.
* **Form Functionality:** Creating, updating, and saving records, handling mandatory and read-only fields, and utilizing reference and choice fields.
* **Form Layout and Design:** Customizing forms with form layout and form design, adding and arranging fields, and configuring related lists.
* **Templates:** Using and scheduling templates to streamline record creation.

1. **Saving Filters:**
   * Save filters to quickly access predefined data views.
   * Share saved filters with others or groups.
   * Apply saved filters via the filters menu.
2. **Grouping and Sorting Data:**
   * Use "Group by" options to categorize records by fields.
   * Sort data alphabetically or numerically.
3. **Form Elements:**
   * **Mandatory Fields:** Required to be filled to save or submit a record.
   * **Read-Only Fields:** Non-editable by users.
   * **Reference Fields:** Show data from other tables (e.g., users).
4. **Form Actions:**
   * **Submit vs. Save:** "Submit" creates and exits the form, while "Save" keeps the form open.
   * **Work Notes and Comments:** Capture activity and changes related to the record.
5. **Form Configuration:**
   * **Form Layout:** Adjust the layout and fields on the form.
   * **Form Design:** Advanced UI for drag-and-drop form design.
   * **Related Lists:** Configure related lists to show associated records.
6. **Templates:**
   * Create and apply templates to pre-fill fields with common values.
   * Schedule templates for automatic record creation.
7. **Knowledge Checks:**
   * Questions are designed to test understanding of the material, such as which applications are visible to users with no roles and interface versions.

**Task Management in ServiceNow**

**Task Management Overview:**

* **Definition:** A task is any record assigned or completed by a user. Tasks track actions and progress for work assigned to individuals or teams. Examples include incidents, problems, and requests.

**Task Workflow Example:**

1. **Incident Creation:** A user reports an issue.
2. **Assignment:** The incident is assigned to a group or individual.
3. **Resolution:** The assignee works on and resolves the issue.
4. **Completion:** The incident state is updated to resolved, and the user is notified.

**Task Table:**

* **Core Table:** The task table is a foundational table in ServiceNow, and many other task-related tables extend from it (e.g., incident, problem, request).

**Functionalities Associated with Tasks:**

* **Approvals:** Can be manual or automated based on rules.
* **Assignments:** Can be manual or automatic via assignment rules.
* **SLA (Service Level Agreement):** Tracks whether tasks meet agreed-upon completion times.

**Task Table and Its Extended Tables:**

* The task table is the parent table, and other tables (e.g., incident, problem, request) extend from it, inheriting fields and adding their own specific fields.

**Access and Permissions:**

* Users need proper access to the records assigned to them. Permissions are critical for task completion.

**Using the Task Table:**

* **Viewing and Grouping:** Records from extended tables are visible in the task table list. Tasks can be grouped by type (e.g., incidents, problems).

**SLA Tracking:**

* **SLA Records:** Associated with tasks to track compliance with agreed resolution times.

**Service Level Agreements (SLAs)**

* **Configuration:** SLAs are configured in the Service Level Management application. You can set SLAs for different tables and tasks, specifying conditions and goals for resolution.
* **Approvals:** Approvals can be attached automatically or manually. In requests, approvers are notified and can approve or reject requests via email or the ServiceNow interface. Approvals can also be integrated into workflows for automated handling.

**Task Assignment**

* **Manual Assignment:** Users can manually assign tasks to groups or individuals. For example, in an incident record, you can select an assignment group and an individual user.
* **Assignment Rules:** ServiceNow provides pre-built assignment rules that automatically assign tasks based on predefined conditions. This is configured under System Policy > Rules > Assignment Rules.
* **Predictive Intelligence:** Uses machine learning to predict values for fields like assignment group and assign to. This requires enabling a plugin and might require a license. It analyzes historical data to improve task assignments.
* **Custom Rules/Scripts:** Administrators and developers can create custom scripts or rules to automate task assignments based on specific business requirements.

**Service Desk Application**

* **Purpose:** Provides a central location for users to access all their tasks (incidents, requests, changes, problems) regardless of the task type.
* **Modules:**
  + **Callers:** View all users in the system.
  + **Incidents:** Access all incidents assigned to the user.
  + **Knowledge:** Access knowledge records.
  + **My Work:** View all tasks assigned directly to the user.
  + **My Group's Work:** View tasks assigned to groups the user is a member of.
  + **My Approvals:** View tasks awaiting the user’s approval.
  + **SLA:** Track SLAs related to tasks assigned to the user or their group.

**Effective Task Management**

* **Work Notes and Comments:** Used to document the progress and actions taken on a task. This is crucial for tracking work and communication.
* **Activity Stream:** Displays all activities related to a task, such as updates, comments, and emails, providing a timeline of actions.
* **Email Functionality:** Allows users to send custom emails related to a task, including attachments if needed.

**Demonstration and Impersonation**

* **Impersonation:** Allows administrators to view and interact with the platform as different users. This is useful for demonstrating functionality and troubleshooting from the perspective of a regular user.

**Incident Management**

**1. Viewing Incidents:**

* **Impersonation:** You can impersonate a user to view their incidents. For instance, to see all incidents in a user’s queue, go to the Service Desk application and select "My Work."
* **Incident Details:** The incident record contains various fields such as state, impact, urgency, assignment group, and assigned to. These fields help in tracking and managing the incident.

**2. Working on Incidents:**

* **Work Notes vs. Additional Comments:**
  + **Work Notes:** Used for internal documentation of actions and updates. They are visible to other agents and admins, but not to the end user.
  + **Additional Comments:** Visible to the end user. Used for communicating updates or requests to the user.
* **Activity Stream:** Displays a timeline of all activities related to the incident, including work notes, additional comments, and email communications.

**3. Email Functionality:**

* **Composing Emails:** You can send custom emails directly from the incident record. This functionality allows you to communicate with the incident’s caller and track the email in the activity stream.
* **Notification Settings:** Notifications for emails might be disabled by default. Ensure they are enabled for users to receive and track notifications effectively.

**Notifications Management**

**1. Types of Notifications:**

* **Outbound Notifications:** Sent by ServiceNow to users based on certain events (e.g., incident assignment). Users receive these notifications and can respond or act based on them.
* **Inbound Actions:** Allows ServiceNow to process emails sent to the instance, such as creating or updating records based on email content.

**2. Notification Management:**

* **System Notification Module:** Accessible via Application Navigator under “System Notification.” It includes:
  + **Email Notifications:** Manage email notifications and templates. You can configure intervals, create new notifications, and manage email scripts.
  + **Push Notifications:** For mobile apps.
  + **Provider Notifications:** For agent workspace and virtual agent configurations.

**3. Notification Creation:**

* **Digest Intervals:** Define how often digest emails are sent to consolidate notifications.
* **Notification Types:** Create and manage notifications for various tables (e.g., incident, problem) and categories.
* **Configuration Fields:**
  + **When to Send:** Configure when notifications should be triggered (on record insert, update, or event).
  + **Who Will Receive:** Define recipients (users, groups) and whether the notification is subscribable.
  + **What It Will Contain:** Configure content type (HTML, plain text), attachments, watermark settings, importance, and sender/reply-to email addresses.

**4. Managing Notifications:**

* **Notification Filters:** Apply filters to customize which notifications users receive based on their preferences.
* **Email Access Restrictions:** Restrict access to specific types of email notifications, such as approval or password reset emails.

**5. Email Logs:**

* **System Logs:** Access logs to view all sent or received emails. This helps in tracking and troubleshooting email notifications.

**6. Practical Example:**

* **Impersonation:** As an admin, impersonate a user, check their incidents, and demonstrate functionalities such as work notes, additional comments, and email sending to showcase how notifications and updates are handled.

This summary should provide a clear overview of how to effectively manage incidents and notifications within ServiceNow, ensuring all relevant features and processes are utilized to optimize task management and communication.

**Notifications**

1. **Types and Creation**:
   * **Email Notifications**: These are messages sent via email based on certain conditions or events. You can create and manage these under the System Notification application.
   * **Push Notifications**: These are messages sent to mobile devices.
   * **Provider Notifications**: These are for agent workspace and virtual agents.
2. **Fields and Sections**:
   * **When to Send**: Define the conditions under which the notification should be sent (e.g., record inserted or updated, specific event).
   * **Who Will Receive**: Specify the recipients, including users, groups, and fields (e.g., caller, assignment group).
   * **What It Will Contain**: Configure the content of the notification, including subject, body, and email templates.
3. **Testing**:
   * Test notifications by changing relevant records (e.g., changing an incident priority) to ensure that notifications are sent as expected.
4. **Advanced Features**:
   * **Email Templates**: Reusable templates for email content.
   * **Watermark**: Unique alphanumeric code added to each email to help identify responses.
   * **Unsubscribe and Notification Preferences**: Allow users to opt-out or manage their notification settings.

**Inbound Actions**

1. **Purpose**:
   * Inbound actions allow ServiceNow to create or update records based on emails received from external sources.
2. **Configuration**:
   * **Name**: Give your inbound action a name.
   * **Target Table**: Specify the table (e.g., incident, change).
   * **Action Type**: Define what should happen when an email is received (e.g., create, update, or respond).
   * **Conditions**: Set conditions for when the inbound action should be triggered.
   * **Script**: Write a script to define the action taken when conditions are met.
3. **Types**:
   * **New Email**: Actions triggered by new emails.
   * **Reply Email**: Actions triggered by replies to existing emails.
   * **Forward Email**: Actions triggered by forwarded emails.
4. **Testing and Debugging**:
   * Test inbound actions by sending test emails to ensure they are processed correctly. Check the **System Logs > Emails** for logs and debugging.

**Knowledge Management**

1. **Knowledge Articles**:
   * Records in a knowledge base that provide information, such as troubleshooting tips or policy documentation.
2. **Benefits**:
   * Centralized information repository for quick answers and self-help.
   * Reduces the workload on support staff by allowing users to find solutions themselves.
   * Enhances efficiency and consistency in information delivery.
3. **Processes**:
   * **Creation**: Define and create knowledge articles with relevant content.
   * **Feedback**: Collect feedback to improve article quality.
   * **Import**: Import knowledge articles from external sources if needed.

**ServiceNow Knowledge Management Overview**

1. **Purpose and Functionality**:
   * **Centralized Knowledge Base**: Stores and categorizes articles based on business units or processes.
   * **Knowledge Articles**: Referred to as KB articles, these are managed within the Knowledge Management application.
2. **Access and Navigation**:
   * **Roles and Access**: Users with the appropriate roles can access, create, and maintain knowledge articles. This is managed through the "Knowledge" application in the Application Navigator.
   * **Home Page**: Provides a search feature for articles and displays different knowledge bases based on user access.
3. **Creating and Managing Articles**:
   * **Article Creation**: Users with the right roles can create new articles, import articles, or manage existing ones.
   * **Article States**: Articles can be in various states such as unpublished, published, or retired.
   * **Submissions**: Allows users without article creation access to request new articles.
4. **Feedback and Ratings**:
   * **User Feedback**: Users can rate articles, flag them, or provide comments to help improve content.
   * **Feedback Management**: Admins can track and manage feedback, including ratings and flagged articles.
5. **Administration**:
   * **Guided Setup**: Helps with the initial setup and configuration of Knowledge Management.
   * **Knowledge Bases**: Allows creation and management of knowledge bases, defining access rights, and setting workflows.
   * **User Criteria**: Controls who can view or contribute to knowledge articles based on various criteria like roles, departments, etc.
   * **Diagnostics**: Helps admins check user access and article visibility.
6. **Knowledge Portal**:
   * **Advanced UI**: Provides an enhanced, user-friendly interface for accessing knowledge articles.
   * **Customization**: Developers can customize the portal based on customer requirements.
7. **Workflow**:
   * **Article Workflow**: Articles move through stages like draft, review, and publication before becoming available to end users.
   * **Approval Process**: Customizable workflows ensure that articles follow the appropriate review and approval process before publishing or retiring.
8. **Roles**:
   * **Knowledge Roles**: Includes roles for creating, editing, and managing articles (e.g., Knowledge Admin, Knowledge Manager).

**Workflow and Customization**

* **Workflow Editor**: Allows the customization of workflows to fit specific organizational processes.
* **Out-of-the-box Workflows**: Includes predefined workflows for managing articles, which can be tailored as needed.

**Knowledge Management in ServiceNow**

1. **Workflow Overview**:
   * **Draft Stage**: Initial creation.
   * **Approval**: Articles must be approved before publishing.
   * **Event Creation**: Associated with the article.
   * **Publishing**: Making the article available to users.
   * **End**: Final stage.
2. **Custom Workflows**:
   * You can create and modify workflows to fit custom requirements.
3. **Importing Articles**:
   * **From Word Documents**: ServiceNow allows importing articles from .docx files.
   * **Steps**:
     1. Upload the document.
     2. Import the document into the knowledge base.
     3. Review and publish the article.
   * **Attachment Handling**: You can attach files to articles, which will be downloadable from the article but will not display content directly in the article view unless attachments are unchecked.

**Service Catalog**

1. **Definition**:
   * A request ordering system for services and products within an organization.
2. **Benefits**:
   * Centralized access to various services.
   * Categorized offerings make finding services easier.
   * Multiple catalogs can be implemented.
   * Track requests and their statuses.
3. **Catalog Modules**:
   * **Catalogs**: Define different catalogs.
   * **Open Records**: View active requests and tasks.
   * **Catalog Definitions**: Define and manage catalog items and services.
   * **Categories**: Organize items into logical groups.
   * **Items**: The individual services or products users can request.
4. **Managing Categories**:
   * Create, edit, and maintain categories.
   * Support for parent-child relationships between categories.
5. **Catalog Items**:
   * Represent services/products available for request.
   * Generate tasks for fulfillment.

**Key Components of ServiceNow Service Catalog:**

1. **Catalog Item**:
   * **Definition**: A catalog item can be a service or product that users can request, such as software, hardware, or services like creating a new email account.
   * **Details**: For each catalog item, you can specify the price, delivery time, vendor, and other details.
   * **Roles**:
     + **Admin**: Full access to create and manage catalog items.
     + **Catalog Admin**: Can manage the service catalog application but not scripting.
     + **Catalog Manager**: Can edit and update catalogs, categories, and items; assign roles.
     + **Catalog Editor**: Can edit and update items but cannot change catalog managers.
2. **Order Guide**:
   * **Definition**: A bundle of multiple catalog items grouped together for a specific request, like onboarding a new employee.
   * **Usage**: Helps users order multiple items in a single request. Each item can be routed to different departments for fulfillment.
3. **Record Producer**:
   * **Definition**: Creates records in existing tables such as incidents or change requests based on user input.
   * **Usage**: Useful for creating forms that generate records in tables, facilitating tasks like server reboots or incident reporting.

**Key Components of Order Form:**

1. **Variables**:
   * **Definition**: Fields or questions on a form that users fill out. Examples include text fields, checkboxes, and date pickers.
   * **Usage**: Customizable to collect necessary information from users.
2. **Variable Sets**:
   * **Definition**: Groups of variables that can be reused across multiple forms.
   * **Usage**: Useful for common fields that appear on several forms, like "Requested for" or "On behalf of".

**Ordering Process:**

1. **Workflow**:
   * **Definition**: A sequence of tasks and approvals that are processed in the background when a request is submitted.
   * **Usage**: Can include custom logic and scripting.
2. **Flows**:
   * **Definition**: A more recent addition that allows users to create automated processes with a drag-and-drop interface, reducing the need for scripting.
   * **Usage**: Ideal for less complex automation needs.

**Viewing and Managing Components in ServiceNow:**

1. **Catalog Item Management**:
   * You can view and edit catalog items, set details such as pricing and visibility, and add related articles or catalog items.
   * **End User View**: Users can see and submit the form via the "Try it" button.
2. **Order Guide Management**:
   * **Setup**: You can configure what items are included in an order guide and under what conditions they appear.
   * **User Interaction**: Users complete a single form that dynamically adjusts based on their selections.
3. **Record Producer Management**:
   * **Setup**: Configure forms that create records in tables based on user input.
   * **Usage**: Simplifies the process of generating records in ServiceNow.

**Overview of Service Catalog in ServiceNow**

**Key Components**

1. **Service Catalog**: A centralized platform for users to request services or products.
2. **Catalog Items**: Specific services or products available for request.
3. **Variables**: Fields on catalog items that gather specific information from users.
4. **Variable Sets**: Groupings of related variables that can be reused across multiple catalog items.
5. **Record Producers**: Forms that create records in various tables based on user input.
6. **Order Guides**: Sequences of catalog items that users can request as a bundle.
7. **Workflows and Flow Designer**: Tools for automating processes and approvals related to catalog items.

**Service Catalog Record Flow**

1. **Request Record Creation**: When a user submits a request, a record is created in the sc\_request table.
2. **Request Items**: Each catalog item within a request creates a record in the sc\_req\_item table.
3. **Catalog Tasks**: For each request item, catalog tasks are created in the sc\_task table for processing and approvals.

**1.Creating a Catalog**

* + Navigate to **Service Catalog > Maintain Catalogs**.
  + Click **New** and enter details (e.g., Catalog Name, Description).
  + Save the catalog.

1. **Creating a Category**
   * Go to **Service Catalog > Maintain Categories**.
   * Click **New** and enter category details.
   * Save the category.
2. **Creating a Catalog Item**
   * Go to **Service Catalog > Maintain Items**.
   * Click **New** and provide item details (e.g., Name, Description).
   * Select the category created earlier.
   * Save the catalog item.
3. **Adding Variables**
   * Within the catalog item, go to the **Variables** tab.
   * Click **New** to create a variable (e.g., Single Line Text).
   * Define the variable question and attributes.
   * Save the variable.
4. **Adding Variable Sets**
   * Go to **Service Catalog > Variable Sets**.
   * Create a new variable set or use an existing one.
   * Add variables to the variable set.
   * Attach the variable set to the catalog item.
5. **Designing Workflow**
   * Navigate to **Workflow > Workflow Editor**.
   * Create or modify a workflow to define the process for handling catalog requests.
   * Attach the workflow to a catalog item.
6. **Using Flow Designer**
   * Go to **Flow Designer**.
   * Create a new flow or modify an existing one.
   * Define the steps and actions for handling catalog requests.
   * Attach the flow to a catalog item.

**Request Example**

* **Placing an Order**:
  + Go to **Service Catalog** and select an item.
  + Fill in the form and submit the request.
  + A request record is created in the sc\_request table.
  + Request items are created in the sc\_req\_item table.
  + Associated catalog tasks are created in the sc\_task table.
* **Viewing Stages**:
  + After submission, you can view the stages of the request (e.g., waiting for approval, configuration).
  + Stages can be customized as per business needs

**1. Overview of Tables in ServiceNow**

* **Tables**: In ServiceNow, tables are fundamental components of the database, used to store data in a structured format. Each table consists of records (rows) and fields (columns).
* **Records**: Each row in a table represents a specific entry or record, and it contains data that is organized in fields.
* **Fields**: Columns in a table are known as fields, which hold specific types of data for each record.

**2. Navigating and Managing Tables**

* **System Definition**: This section in the Application Navigator includes:
  + **Tables**: Lists all tables available in your instance, including both out-of-the-box and custom tables.
  + **Tables and Columns**: Provides an advanced view of tables and their respective columns.
  + **Dictionary**: Contains definitions for each table and field, allowing for detailed configuration.

**3. Table Details**

* **Viewing Table Details**: By navigating to a specific table (e.g., User Table), you can see the configuration and columns associated with it.
* **Indexing**: Indexes are used to optimize query performance. They help speed up searches and improve overall database efficiency.

**4. Field Configuration**

* **Field Attributes**: Each field has attributes such as:
  + **Field Label**: The name displayed to users.
  + **Field Name**: The backend name used in scripts.
  + **Field Value**: The data stored in the field.
* **Configuring Fields**: Fields can be made read-only or mandatory, and their dictionary configurations can be adjusted.

**5. Table Relationships**

* **One-to-Many Relationship**: For example, the User table (one) has many incidents (many) associated with it.
* **Many-to-Many Relationship**: For example, the Group and Roles tables are related through a many-to-many relationship via the Group Roles table.
* **Extended Relationship**: This is where one table (child) extends another (parent). For instance, Incident and Problem tables extend from the Task table.

**6. Types of Tables**

* **Base Tables**: These are not extended from other tables. They serve as the root tables. Example: Task Table.
* **Extended Tables**: These tables inherit fields from a parent table. Example: Incident Table extends from Task Table.
* **Core Tables**: Provided by ServiceNow out-of-the-box for standard applications. Examples include Incident, Change, Problem tables.
* **Custom Tables**: Created by administrators or developers to meet specific business needs. Example: A custom table for tracking specific business metrics.

**7. Creating and Managing Custom Tables**

To create a custom table in ServiceNow:

1. **Navigate to the Table Creation Form**: Use the "New" button in the Tables module.
2. **Fill Out the Form**: Provide a label for the table and a name (which is auto-generated based on the label).
3. **Add Fields**: Define the fields you need for your custom table. ServiceNow will automatically create some default fields such as sys\_id, created, and updated.

**Example Workflow for Custom Table Creation:**

1. **Create a Custom Table**:
   * Label: Demo Table
   * Table Name: u\_demo\_table
2. **Define Fields**:
   * Field 1: Name (Type: String)
   * Field 2: Description (Type: String)
   * Field 3: Status (Type: Choice, with options like "New", "In Progress", "Closed")
3. **Save and Use**: Once created, the table can be used to store and manage custom data.

**Practical Example:**

When creating a new table:

* Go to **System Definition > Tables**.
* Click **New**.
* Enter details (e.g., Demo Table).
* Save the table and start adding fields as needed.

**Tables and Fields in ServiceNow**

1. **Tables**:
   * **Definition**: Tables store records in a structured format. Each table consists of rows (records) and columns (fields).
   * **Types**:
     + **Base Tables**: Not extended from any other table (e.g., task table).
     + **Extended Tables**: Inherit fields from a parent table (e.g., incident and change tables are extended from the task table).
     + **Core Tables**: Provided out-of-the-box by ServiceNow (e.g., incident, change).
     + **Custom Tables**: Created by administrators or developers to meet specific business needs.
2. **Fields**:
   * **Definition**: Columns in a table that store data. Each field has attributes like field label, field name, and field value.
   * **Types**: Various field types are available, such as string, choice, reference, etc.
3. **Creating and Configuring Tables**:
   * **Creating a Table**: You can create a custom table through the Tables module in the System Definition application.
   * **Field Inheritance**: When creating a table that extends from another (e.g., extending from task), it inherits all fields from the parent table.
   * **Adding Fields**: Fields can be added directly in the table’s configuration. Default fields like created\_by, created\_at, sys\_id, etc., are automatically included.
4. **Schema Map**:
   * **Purpose**: Provides a graphical representation of table relationships and schema configurations. Useful for understanding how tables are interconnected.

**Access Control (ACL)**

1. **Overview**:
   * **Definition**: Security rules that control access to tables and records. ACLs manage which users or roles can create, read, update, or delete records.
2. **Types of ACLs**:
   * **Table-Level**: Controls access to entire tables.
   * **Record-Level**: Controls access to specific records (rows).
   * **Field-Level**: Controls access to individual fields (columns).
3. **Operations Restricted by ACLs**:
   * **Create**: Restricts users from creating new records.
   * **Read**: Restricts users from viewing records or fields.
   * **Update**: Restricts users from modifying records or fields.
   * **Delete**: Restricts users from deleting records.
4. **Special Operations**:
   * **Execute**: Prevents execution of scripts on records.
   * **Add CI Relations**: Restricts adding relationships in the CMDB.
   * **Save as Template**: Restricts the ability to create templates.
   * **Report On**: Restricts the creation of reports.
   * **Personalize Choice**: Restricts the ability to configure choice fields.
5. **Access Control Form**:
   * **Fields**:
     + **Type**: Specifies the type of ACL (e.g., record, rest endpoint).
     + **Operation**: Specifies the operation being restricted (e.g., read, update).
     + **Admin Override**: Allows admins to bypass ACL rules.
     + **Table/Object**: Specifies the table and object the ACL applies to.
     + **Required Roles**: Defines roles required to bypass ACL rules.
     + **Condition**: Custom conditions for access control.
     + **Custom Script**: Allows for custom scripting if other conditions are insufficient.

**Practical Application**

1. **Creating Custom Tables**:
   * When creating a custom table, you can decide whether it should extend an existing table or be standalone.
   * **Example**: Creating a custom table demo\_youtube\_task that extends the task table inherits all fields from task.
2. **Managing Fields**:
   * Fields can be added or removed as required. The field configuration affects the table's form layout and list views.
3. **Defining Access Control**:
   * ACLs are configured based on the specific needs for access control. You need to set conditions, roles, and operations as per your organizational security requirements.

**Creating Tables:**

1. **Naming Conventions:**
   * Custom tables are prefixed with u\_, which helps identify them as user-created tables.
   * Tables can be global or private in scope. Global tables are accessible across the platform, while private tables are restricted.
2. **Extending Tables:**
   * When creating a new table, you can extend from an existing table (e.g., the task table). This means your new table will inherit fields from the parent table, which can save time and ensure consistency.
3. **Fields and Columns:**
   * You can define various field types (e.g., string, integer) and set properties like maximum length and default values.
   * Additional fields like created\_by, created\_on, sys\_id, etc., are automatically included in new tables to track metadata.
4. **Module Creation:**
   * When creating a table, you can choose to create a module for it in the application navigator, or you can opt out if you don't want to add it to any application.
5. **Schema Map:**
   * The Schema Map provides a graphical representation of the relationships between tables. It helps visualize how tables are related and which fields they reference.

**Access Control Lists (ACLs):**

1. **Types of ACLs:**
   * **Table.none:** Applies to the entire table without any field-specific restrictions. If users don’t match the conditions, they can't access the table.
   * **Table.star (\*):** Applies to all fields of the table. This is useful for restricting access to all fields within the table, even if users have table access.
   * **Table.field:** Applies to specific fields within a table. Users may access the table but may not see or interact with particular fields.
2. **Operations Controlled by ACLs:**
   * **Create:** Restricts the ability to create new records.
   * **Read:** Restricts the ability to view records.
   * **Update:** Restricts the ability to modify records.
   * **Delete:** Restricts the ability to delete records.
   * **Execute:** Restricts the ability to run scripts or certain actions.
   * **Add CI Relations, Save as Template, Report on, Personalize Choice:** Additional specific actions that can be restricted.
3. **How ACLs Work:**
   * When a user attempts to access a record, ServiceNow evaluates applicable ACL rules.
   * If no matching ACL is found, access is granted.
   * If matching ACLs are found, they are evaluated based on conditions and roles. Access is granted if conditions are met; otherwise, it is denied.
4. **Creating and Managing ACLs:**
   * To create or edit ACLs, you need the security\_admin role.
   * ACLs can be managed under System Security → Access Control (ACL).
   * You can use ACLs to control access at the table, record, or field level.

**Examples:**

1. **Automatic ACL Creation:**
   * When creating a table with the option to "Create Access Control," default ACLs are automatically created for CRUD operations (Create, Read, Update, Delete).
2. **Custom ACL Creation:**
   * If you create a table without checking the access control box, you’ll need to manually create and configure ACLs to control access.
3. **Impersonation Testing:**
   * To test access controls, you can impersonate users and verify whether they can access records or fields based on the ACLs configured.

**Handling ACLs and Data Import in ServiceNow**

**Access Control Lists (ACLs)**

1. **Hiding Field Data:**
   * To restrict access to a specific field in a table, you create a new ACL rule:
     + **Navigate to:** System Security → Access Control (ACL)
     + **Create New Rule:** Choose to add a rule for the table and select the specific field you want to control.
     + **Specify Conditions:** Set conditions to ensure only the intended users (e.g., admins) can view or interact with that field.
     + **Test:** Use impersonation to verify if the field data is correctly hidden from non-authorized users.

Example:

* + **Table:** ACL Demo
  + **Field to Hide:** test\_two
  + **Condition:** Only users with the admin role can see test\_two.

1. **Impersonation:**
   * Use impersonation to test how different users see the data:
     + **Impersonate:** Go to User Administration → Users, select a user, and click Impersonate.
     + **Verify Access:** Check the affected fields or records to ensure ACLs are applied as expected.

**Data Import into ServiceNow**

1. **Need for Data Import:**
   * **Bulk Uploads:** Import data to avoid manual entry.
   * **Sources:** Examples include Active Directory for user data, HR systems for employee records, or external sources for knowledge articles and asset management.
2. **Ways to Import Data:**
   * **Import XML:**
     + Use for importing data from XML files into ServiceNow.
     + **Procedure:**
       - Navigate to the list view of the table.
       - **List Context Menu:** Click Import XML and upload the XML file.
     + **Use Case:** Commonly used for moving data between ServiceNow instances.
   * **Import Data (Excel):**
     + **Procedure:**
       - Go to System Import Sets → Load Data.
       - Choose Import Data to upload Excel files.
       - **Templates:** Download templates if needed, fill in the data, and upload it.
     + **Use Case:** Ideal for importing structured data like spreadsheets.
   * **Import Sets:**
     + **Core Functionality:** Manages data imports using various sources.
     + **Components:**
       - **Data Source:** Defines where the data comes from (e.g., file, LDAP, JDBC).
       - **Import Set Table:** Staging area for data before it's transformed and inserted into the target table.
       - **Transform Map:** Maps source data fields to target table fields.
       - **Mapping Assist:** Helps in visual field mapping; can be done manually or automatically.
       - **Coalescing Fields:** Ensures existing records are updated if matches are found; new records are inserted otherwise.
       - **Target Table:** Final destination where the data is inserted after processing.
3. **Example of Data Import:**
   * **Import XML Example:**
     + **Scenario:** Export an XML of incident records with empty short descriptions.
     + **Steps:**
       - **Export Data:** Select the XML format for export.
       - **Upload Data:** Use the Import XML feature to import this XML back into ServiceNow.

**plaintext**

<incident>

<short\_description></short\_description>

</incident>

**Import Process:**

**->Load Data:** Navigate to System Import Sets and select Load Data.

**->Upload XML:** Upload the XML file and ensure it is processed correctly.

**Importing Data into ServiceNow: Practical Example**

**1. XML Import**

* **Exporting Data**:
  + Export the data from a ServiceNow table (e.g., Incident) in XML format.
  + Edit the XML file to update specific fields (e.g., short\_description).
* **Editing XML**:
  + Open the XML file in a text editor.
  + Locate the field you want to update (e.g., short\_description).
  + Modify the field value (e.g., This is a demo for importing data).
  + Save the XML file.
* **Importing XML**:
  + In ServiceNow, navigate to the list view of the table (e.g., Problem).
  + Click on the list context menu and select **Import XML**.
  + Upload the edited XML file.
  + ServiceNow will automatically recognize the table and import/update the data.

**2. Import Set and Transform Map**

* **Setting Up**:
  + Navigate to **System Import Sets** in the Application Navigator.
  + Create a new import set to load data from an Excel file.
* **Loading Data**:
  + Click **Load Data**.
  + Select the file (e.g., an Excel file you created with data).
  + Define the target table (e.g., Demo ACL).
  + Submit to create a staging table.
* **Creating Transform Map**:
  + Click **Create Transform Map**.
  + Define the transform map, selecting the target table (e.g., Demo ACL).
  + Click **Mapping Assist** to map fields from the staging table to the target table fields.
  + Map fields from the Excel file to the ServiceNow fields.
  + Save the transform map.
* **Running Transform**:
  + Click **Run Transform** to process the data from the staging table to the target table.

**3. Working with Import Sets**

* **Data Sources**:
  + Define different data sources (e.g., FTP servers, LDAP, CSV files).
  + Set up the connection and specify the source data.
* **Importing Data**:
  + Use the **Load Data** feature to bring data into ServiceNow.
  + Create a staging table to temporarily hold the data.
* **Transforming Data**:
  + Use the **Create Transform Map** to map and transform data.
  + Run transformations to move data from staging to the target table.
* **Monitoring**:
  + Check the **Import Sets** module for progress and errors.
  + Review the **Transform History** and **Transform Errors** for troubleshooting.
* **Advanced Options**:
  + Schedule imports for regular data updates.
  + Clean up import logs to manage system performance.

**Data Import Process:**

1. **Create Transform Map:**
   * Define how data from the import set maps to your target table.
   * Ensure the transform map is set up correctly for the fields you want to map.
2. **Transform Data:**
   * Click on "Transform" to process the import set using the transform map.
   * Check the import set log to verify the records were inserted or updated.
3. **Handle Coalesce Fields:**
   * Coalesce fields determine whether records should be updated or inserted.
   * For example, if a number field is set to coalesce, it will update existing records with that number instead of creating new ones.
4. **Update Transform Map:**
   * If you need to change how the data is handled (e.g., by using coalesce fields), update the transform map to include the necessary mappings.
5. **Test Data Import with Policies:**
   * Create and apply data policies to ensure mandatory fields are filled.
   * Verify the system’s behavior with missing mandatory fields and adjust policies as needed.
6. **Review and Debug:**
   * If data isn't imported or updated as expected, review error messages and settings.
   * Adjust coalesce settings or data policies if needed to correct the issues.

**Creating and Managing CMDB:**

1. **Understand CMDB:**
   * The Configuration Management Database (CMDB) stores information about Configuration Items (CIs) and their relationships.
   * Examples include computers, servers, applications, and databases.
2. **Utilize CMDB Modules:**
   * Use modules such as CI Class Manager, CMDB Groups, and CMDB Quality Builder to manage and maintain CMDB data.
3. **Configuration Items and Relationships:**
   * Manage CIs by editing or configuring items in your CMDB.
   * View and understand relationships between different CIs using CI Dependency View.

**Tips for Effective Data Management:**

* **Always Check Coalesce Settings:** Make sure that the coalesce fields are correctly set to avoid unintended data duplication.
* **Apply Data Policies Appropriately:** Ensure data policies are configured to prevent the import of invalid or incomplete data.
* **Test Thoroughly:** Before performing large-scale data imports, test with a subset to ensure that all configurations and policies work as expected.

Configuration Management Database (CMDB) and its components in ServiceNow is comprehensive. Here’s a condensed summary of the key points you covered:

**Configuration Item (CI) Fields**

1. **Name**: The name of the CI.
2. **Asset Tag**: A unique identifier for the CI.
3. **Manufacturer**: The company that manufactured the CI.
4. **Asset**: Linked to the asset management table for tracking.
5. **Class**: Defines the category or type of the CI.
6. **Company**: The customer or owner of the CI.
7. **Serial Number**: The unique serial number of the CI.
8. **Model ID**: The model identifier of the CI.
9. **Assigned To**: The primary person responsible for the CI.
10. **Comments**: Any additional notes about the CI.

**Configuration Details**

* Fields like Hostname, OS, Domain, IP Address, etc., describe the CI’s configuration.

**Related Items**

* Shows CIs related to the current CI and their connections.

**CI Dashboard View**

* Provides a health overview, related CI details, and metrics like completeness, compliance, and correctness.

**Key CMDB Tables**

1. **Base Configuration Item**: Main parent table.
2. **Configuration Item**: Extends from Base Configuration Item.
3. **CI Relationship**: Stores relationships between CIs.

**CMDB Table Hierarchy**

* **Base Configuration Item** extends to **Configuration Item**.
* Specific tables like **Hardware** or **Application** extend further into specific classes such as **Computer** or **Application Server**.

**Usage of CMDB**

* **Incident Management**: Link incidents to CIs for effective resolution.
* **Problem Management**: Identify and resolve issues related to CIs.
* **Change Management**: Track changes to CIs.

**Dependency View**

* Displays a graphical view of a CI and its related CIs, showing impact and connections, useful for troubleshooting.

**CI Class Manager**

* Manages CI classes in a hierarchical structure, allowing modification and extension of classes.

**Creating and Managing Relationships**

* **Manual Relationships**: Add relationships using the Relationship Editor.
* **Suggested Relationships**: Use predefined relationships to maintain data integrity and relevance.

**Dependency View Example**

* Visualizes connections and dependencies among CIs to identify potential impacts and issues.

**Dependency View in ServiceNow**

The Dependency View provides a graphical representation of how different Configuration Items (CIs) are related. Here’s a quick guide on how to use and interpret it:

1. **Displaying Relationships**:
   * When you open the Dependency View for a CI, it shows how various CIs are connected to the selected CI. For example, if you view a server, it might show which computers or other servers are using it.
2. **Interpreting the View**:
   * **Graphical Representation**: Displays CIs in a visual format, showing connections like "used by" or "depends on."
   * **Indicators**: Icons might indicate if there are incidents, problems, or maintenance tasks associated with any of the CIs.
3. **Use Case Example**:
   * **Issue Diagnosis**: If an application is down, you can use Dependency View to identify related CIs (like database instances or servers). This helps in pinpointing which CIs might be causing the problem or are affected by it.

**Integration and Update Sets**

**Integrations in ServiceNow**:

* **Common Integrations**: CMDB, Incident Management, Problem Management, Change Management, User Administration, and Single Sign-On (SSO).
* **Integration Methods**: Web Services, LDAP, Excel, Email.
* **Integration Hub**: Allows integration with third-party applications using Flow Designer. Packages are available in Standard, Professional, and Enterprise tiers.

**Update Sets**:

* **Definition**: A group of configuration changes that can be moved from one instance to another. They allow administrators to package changes and move them as a unit.
* **Example**: Moving form changes, UI actions, or scripts from a development instance to a production instance.
* **Application**: Managed under the System Update Sets application, including modules for updating sources, retrieving update sets, and managing update set information.

**Update Sets in ServiceNow**

1. **Understanding Update Sets**

* **Purpose**: Capture changes (development, configuration) made in one instance and move them to another.
* **Components**:
  + **Update Set**: A collection of changes.
  + **Local Update Sets**: Changes captured within the current instance.
  + **Merge Update Sets**: Combine multiple update sets into one.
  + **Completed Update Sets**: Update sets that are finalized and ready for deployment.
  + **Update Sets to Commit**: Update sets pending installation in the target instance.

2. **When to Use Update Sets**

* **Consistency**: Use update sets to ensure changes are consistent across instances (e.g., dev, test, production).
* **Testing**: Perform changes in lower environments before moving them to production.
* **Impact**: Avoid making direct changes in production; use update sets to mitigate risk.

3. **Items Captured in Update Sets**

* **Captured**: Form configurations, business rules, client scripts, UI policies, notifications, script includes, etc.
* **Not Captured**: Task records (e.g., incidents), user data, groups, scheduled jobs, CMDB records, system properties.

4. **Planning and Promotion Process**

* **Version Consistency**: Ensure instances are on the same ServiceNow version.
* **Correct Update Set**: Select the appropriate update set before making changes.
* **Instance Cloning**: Clone production instances to lower environments to maintain consistency.
* **Movement Path**: Define a clear path for moving update sets (e.g., dev → UAT → staging → production).
* **Naming Conventions**: Use clear, understandable names for update sets.
* **Preview and Commit**: Review update sets for errors or warnings before committing.

5. **Promotion Steps**

* **Development**: Capture changes in the development instance.
* **Testing**: Move to UAT, test, then to staging for further validation.
* **Production**: Finally, move validated update sets to the production instance.

6. **System Update Set Tables**

* **Update Set Table**: sys\_update\_set – Stores update set data.
* **Customer Update Table**: Stores individual changes within an update set.

**Events in ServiceNow**

1. **Generating Events**

* **Business Rules**: Define triggers (e.g., record insert/update).
* **Scripting**: Use gs.eventQueue() to generate events from server-side scripts.
* **Flow and Workflow**: Define triggers and actions based on events.

2. **Event Registry**

* **Event Names**: Define event names in the event registry (e.g., incident.inserted, problem.updated).

3. **Event Actions**

* **Notifications**: Send notifications based on events.
* **Business Rules and Flows**: Define actions in business rules and flows.
* **Script Actions**: Execute scripts as a result of events.

4. **Event Log**

* **Monitoring**: View generated events, check for troubleshooting issues.

**Platform Stats**

* **Module**: System Diagnostic → Stats.
* **Information**: Includes version, cluster info, build name, instance name, IP address, memory usage, transaction statistics, etc.
* **Use**: Troubleshooting and system performance monitoring.