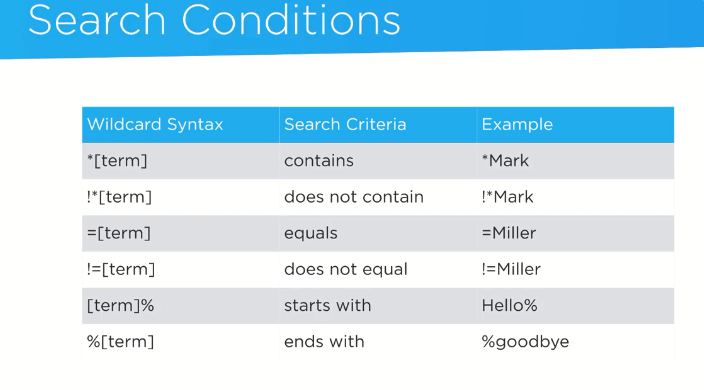
**Filters and search within ServiceNow**

* Many search bars
* Similar search functions
* Search Wildcards

**Search wildcards**



The **condition builder** and **bread** **crumbs**

**Condition builder**

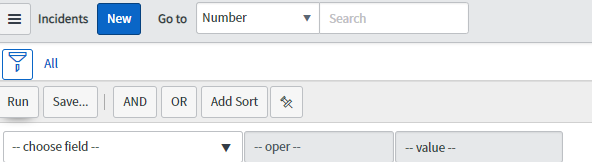
* Very powerful tool that allows user to define search criteria without needing to actually write SQL statements
* We can access related fields such as the caller and user fields
* We can save the condition, copy condition
* The format is field, operator and finally the value that we can provide

**BreadCrumbs**

* Hand in hand with condition builder
* Top of any list view in service now
* They are dynamic and change when there is any change in condition builder
* We can use the shortcut to remove a certain filter by avoid going into the condition builder and using “>” sign just before the condition
* We can right click on the breadcrumb and copy the URL or the query

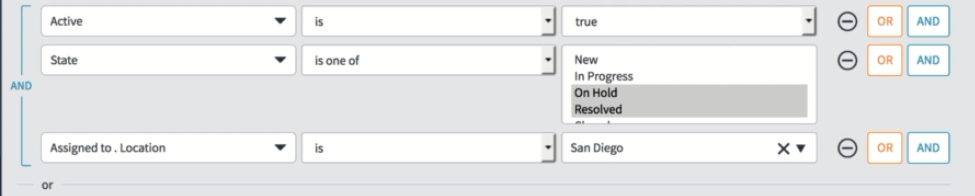


Condition builder in actual development instance



In this module, I explored the incident application and under that I selected all the incidents and applied filter on that. I then saved the filter. Used the > sign to remove the condition and then see the filtered results without the specific conditions. Then, again loaded the filter with the saved filter.

Also we accessed the reference fields that points to the



**Copy url form bread crumbs**

<https://dev56350.service-now.com/incident_list.do?sysparm_query=caller_id.nameSTARTSWITHjoe%5Ecaller_id%3D681ccaf9c0a8016400b98a06818d57c7>

**Query** caller\_id.nameSTARTSWITHjoe^caller\_id=681ccaf9c0a8016400b98a06818d57c7

For more information - Check the condition builder service now docs page

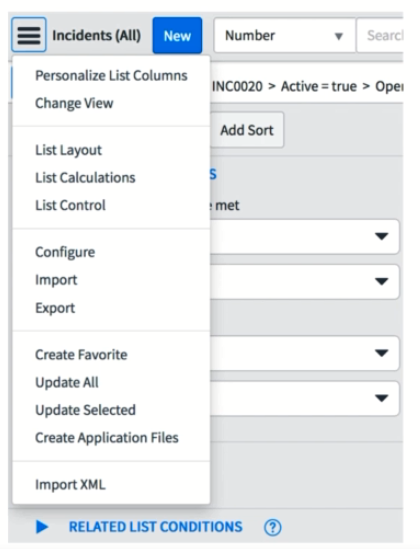
**Context Menus within ServiceNow**

Appears in both list and forms views of the platform

Provide many options depending upon the application

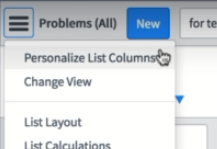
They are accessed by the context menu icons which is generally shown as a hamburger icon in the service now

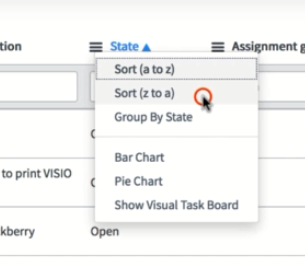
Provide application look and feel



**There is a difference between the context menus in the developer instance because the trainer is using the list v3 and the ServiceNow has removed the support for the list v3**

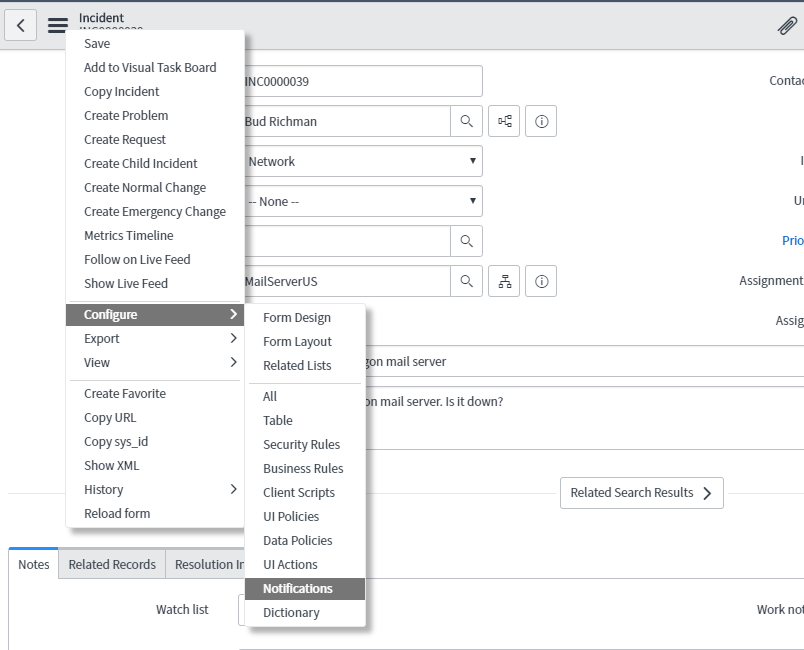
There are several context menus in the ServiceNow

* **Main list context menu** - top left of the main content frame 
* **Column context menu**



* **Cell context menu**

**Main form context menu** – located on same position as of the main list context menu, this form is opened when we click on a particular incident



**Modifying list and forms**

Personalize vs configure

Personalizing list is user specific, user has the ability to add or remove the existing columns. It is specific to the user

Configuring lists is done by system admin and needs specific permissions and it is applied system wide.

We can change lists and add several fields and columns and similarly we can change the form fields

**Section 4**

**Customizations within service now**

* **Client side vs server side**
* **Customizing service now**
* **UI policies**
* **UI actions**
* **Business Rules**
* **Client Scripts**
* **Data Policies**
* **Script Includes**
* **Update sets**
* **Plugins**

**Client side vs server side**

As contrary to desktop applications where the data is stored in the desktop and processed by computer itself, for the web applications, the data is processed in the data centre.

In this course, the **client side** is the **browser** I used for accessing the service now ex desktop, laptop

The **server side** refers to the **datacentre** where the application server and database servers and instance data are located and stored

**Request and response flow**

Anytime I want to access the data in ServiceNow, the client or the browser makes a request to the service now datacenters.

The application server then receives the request, and contact with the service now datacenters. Once the application server receives the servicenow data, it makes a package then sends the response to the client which is the browser and it renders the data on the screen

Example- when I want to access a single incident request, the client does not have much information to display so, the browser make the request to the server side instance for that particular incident, after that the servicenow datacenters provide the response data for that incident request. It all happens in real time and the time taken by the request and response is called the **round trip time (RTT)**. Moreover, the servicenow servers are very powerful but require the network activity and also no matter how good the internet connectivity is, there can still be some lag/delay due rtt.

**Main points about servicenow**

* It is very flexible we can make a lot of customizations
* We can create custom applications, tables and pages for business needs that can interact with existing instance data
* Customizations can be added to a number of different places within service now

We will be learning a number of customizations as following



**UI Policies**

We need UI policies to define certain conditions that must be done or some rules that should be followed when performing an action.

These are used to set form fields to mandatory, read only, show and hide the field. Also, policies reside in the client side where UI is implemented.

For example,

**If I want to resolve and incident and set its state to resolved, then it is mandatory to set the close code, close notes or else there will be an error.**

Another example – Set an incident’s short description field to read-only if the incident state I closed

* Hide the incident’s resolution notes field if the state is open

**UI actions**

Used to add buttons, links, and items to context menus

**Server side and client side**

They run on the server side and can optionally run JavaScript on the client side. This gives them a unique ability of one of the few customizations that may run either client side or server side. This **leverages JavaScript** so user can do some advanced operations with them.

Example – copy change button that will copy a change into new records

Another – propose a sample change request into the context menu of the change request

**Scenario uses of the actions**

1. **Trigger Salesforce integration, creating an associated salesforce ticket**

Let’s say we have an UI action on the incident record, that when clicked will send data on the incident record and create a salesforce ticket via a REST API

1. **Reject an approval record**

Suppose we have a UI action labelled as Reject and when pressed it will change the approval to reject

We can make the buttons and define the actions performed by them using JavaScript, make those buttons active or inactive.

**Business Rules Customization**



Let’s discuss the example for updating a record request

1. User sends request to the server for specific incident or query
2. Application server receives the request and contact with the database server for the record
3. Database server check the database and then gives the application server with the record
4. Application server checks for display business rules and then sends the data to the user
5. User modifies the incident record via form and sends the updated request to the application server
6. The application server then receives the update and check for before business rule and sends the data to the database server
7. The database server then updates the record
8. The application server then checks after business rules and sends the response back to the user

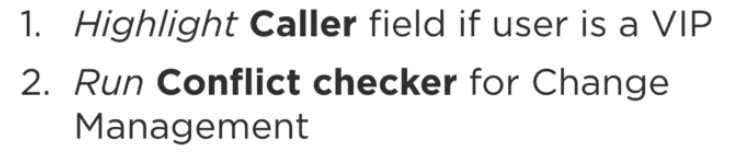
Business rules use cases

1. Create an associated CI when new asset is created

Ci record is created when a new asset record is created on the asset table, we create a new business rule which will create a new CI, run on insert and assign to the asset table

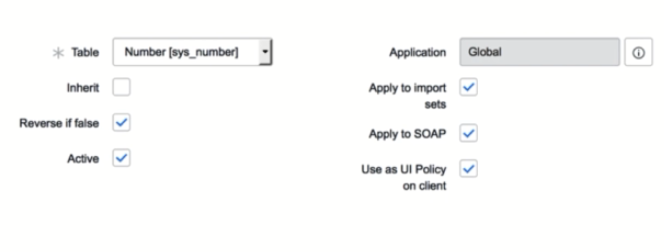
1. When an incident is reopened, incident the reopen count

**Client scripts**

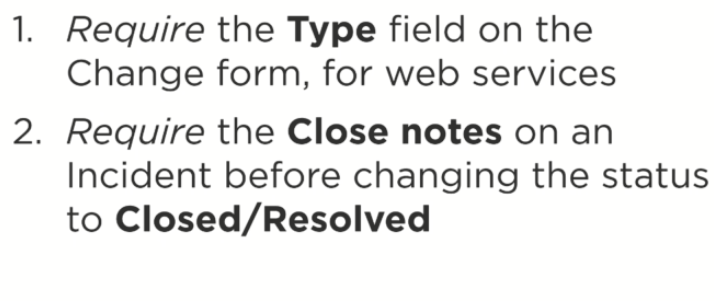


**Data policies**

Form view of a data policy

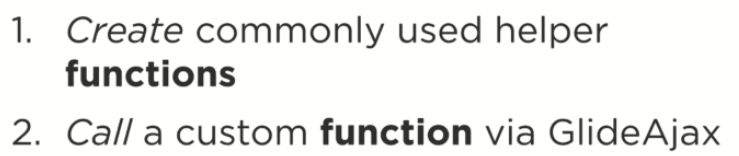


Two use cases

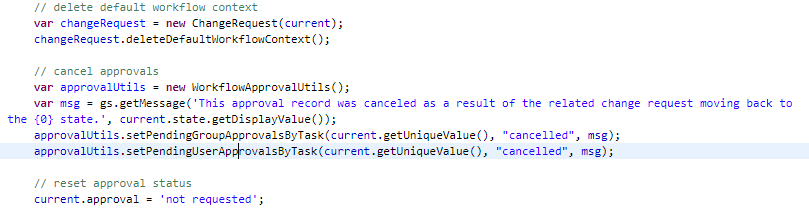


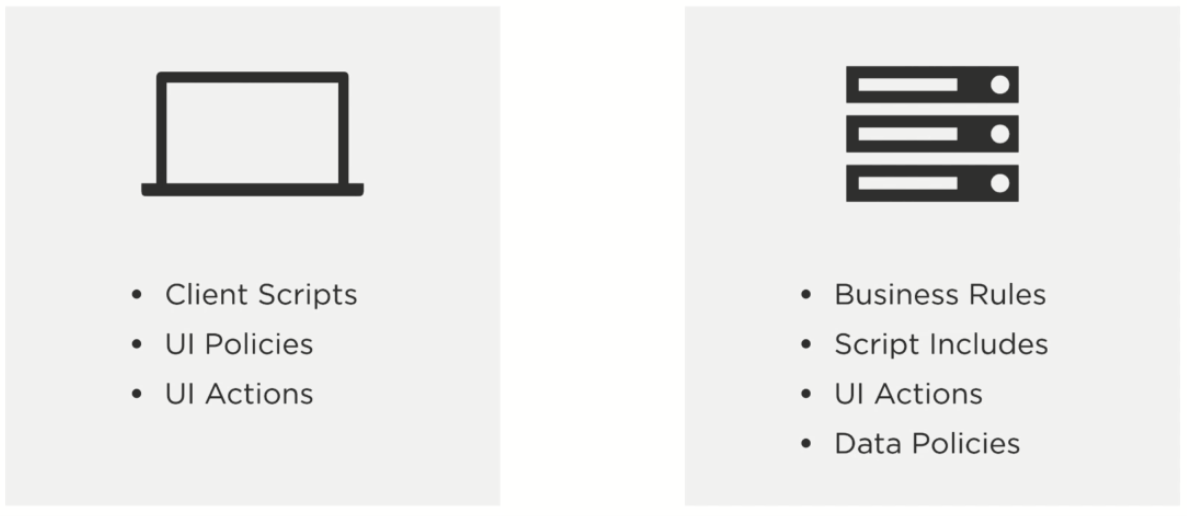
**Script includes – server side**

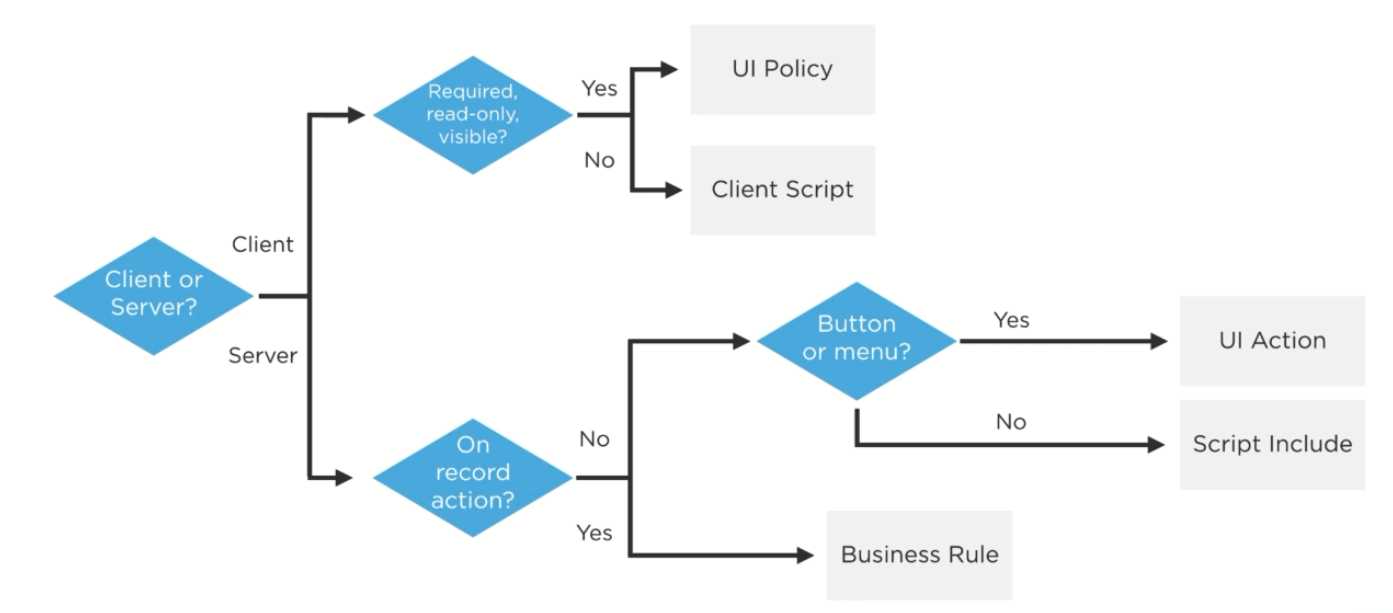
Use cases

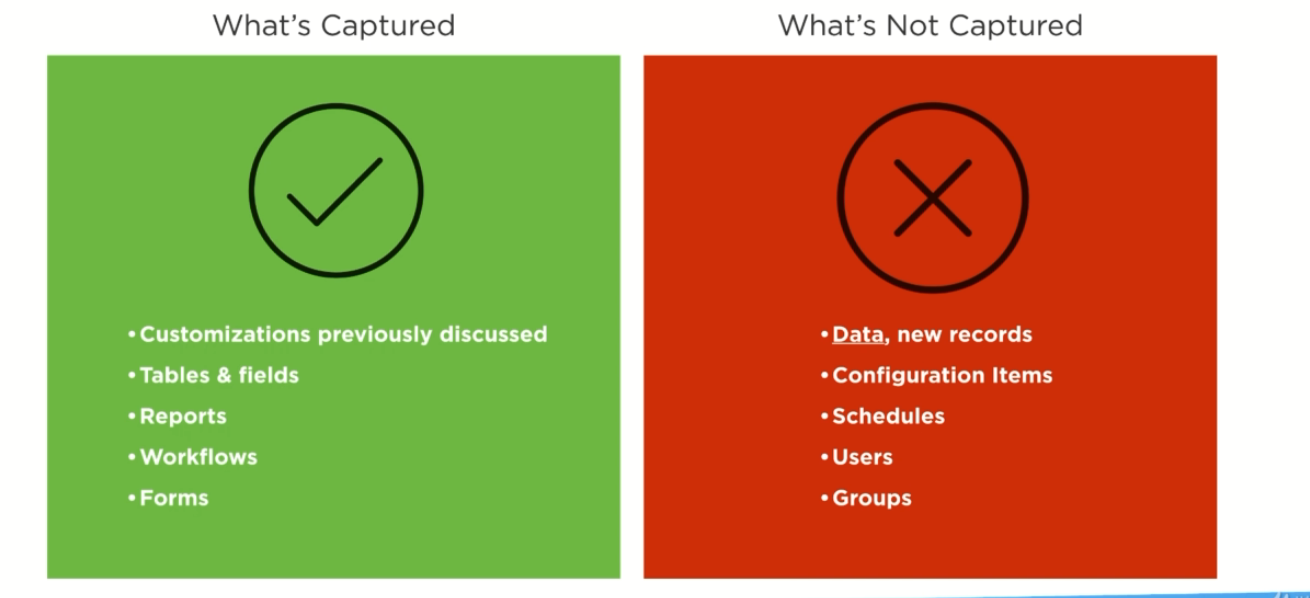


Calling the functions in the script include via the business rules



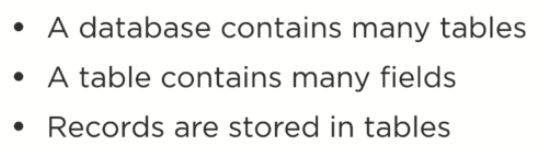
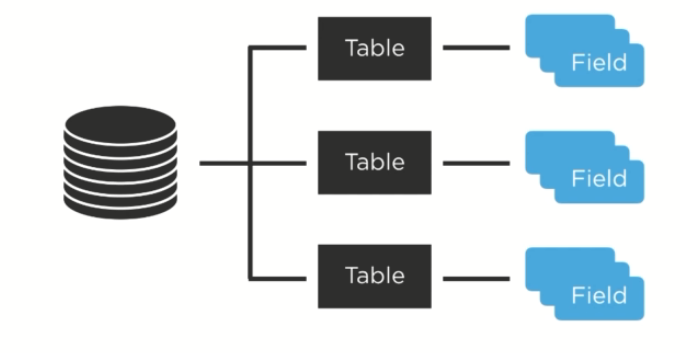






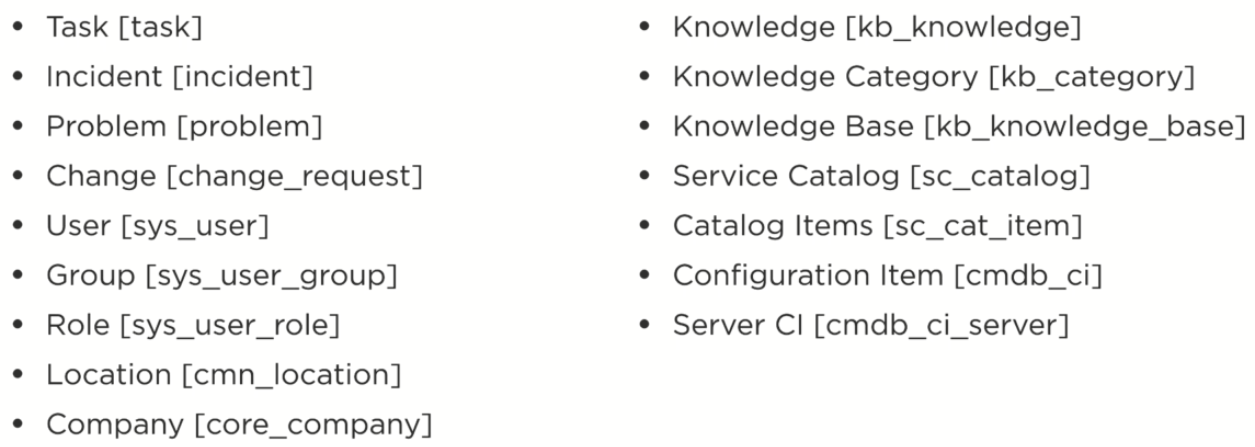
**Plugins**

**Tables**



Column in a **field** and row is a **record**

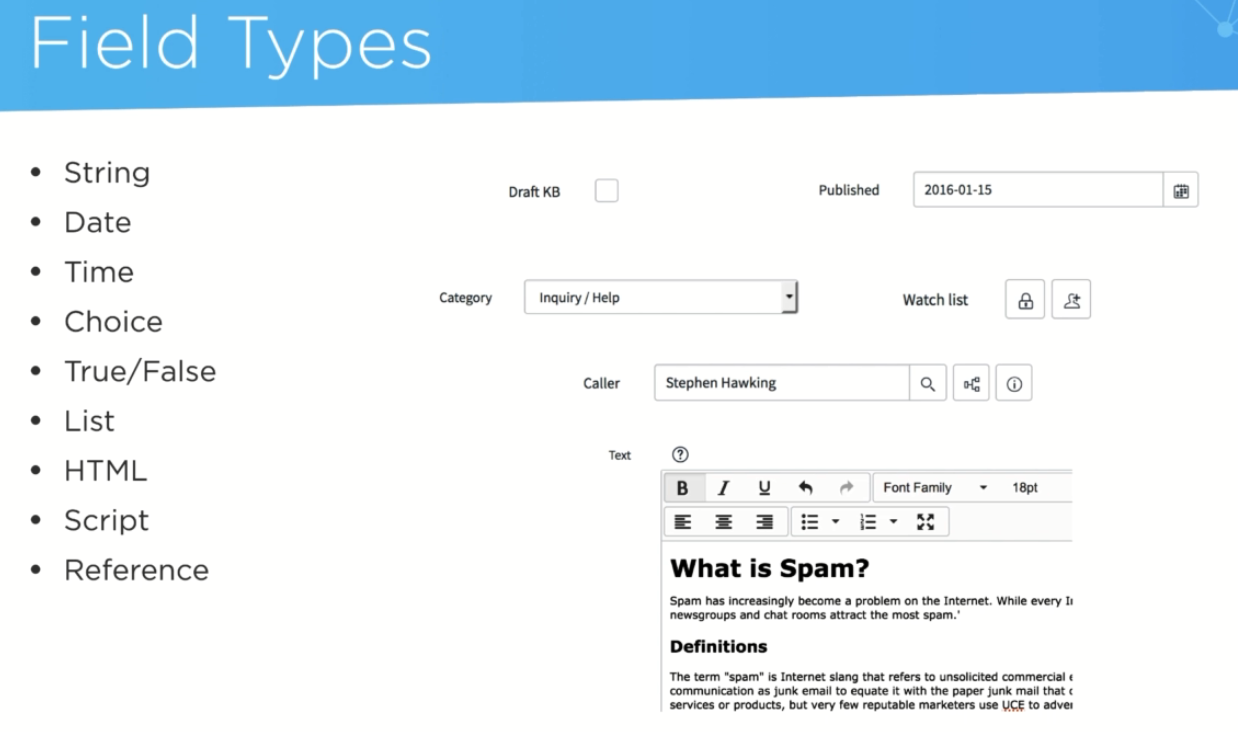
**Major tables**



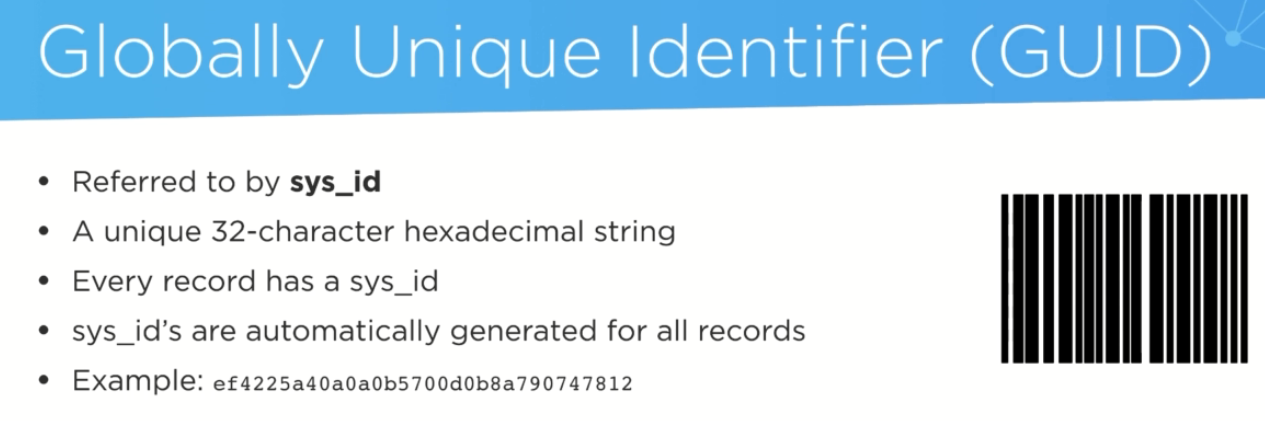
Data dictionary tables

Have Meta data about the tables

**Fields**



Guids



SECTION 6

User

Groups

Roles

Access controls

Access operations

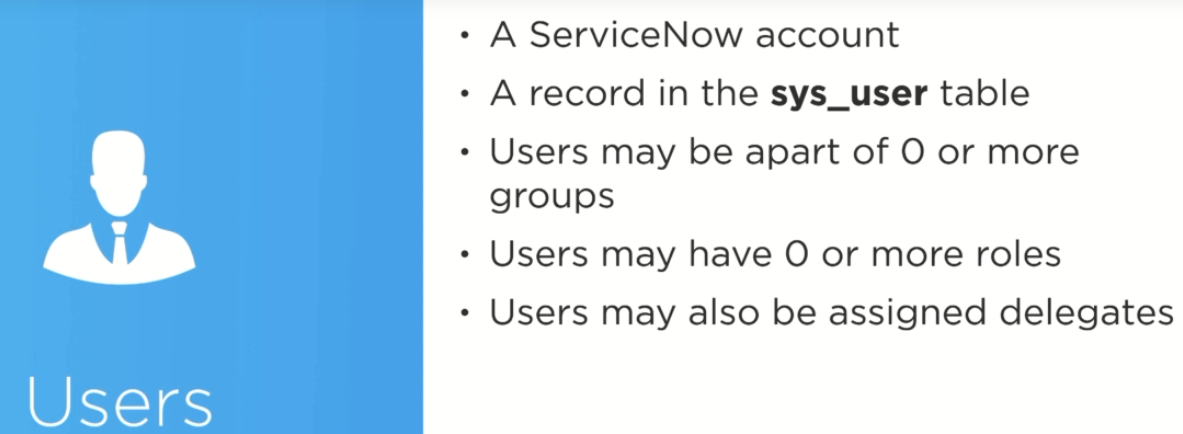
Access Control execution

LDAP

SSO

Impersonation

**User**



**Groups**

