



ServiceNow to ServiceNow Integration: The Step-by-Step Guide to Setting up a Two-Way Sync

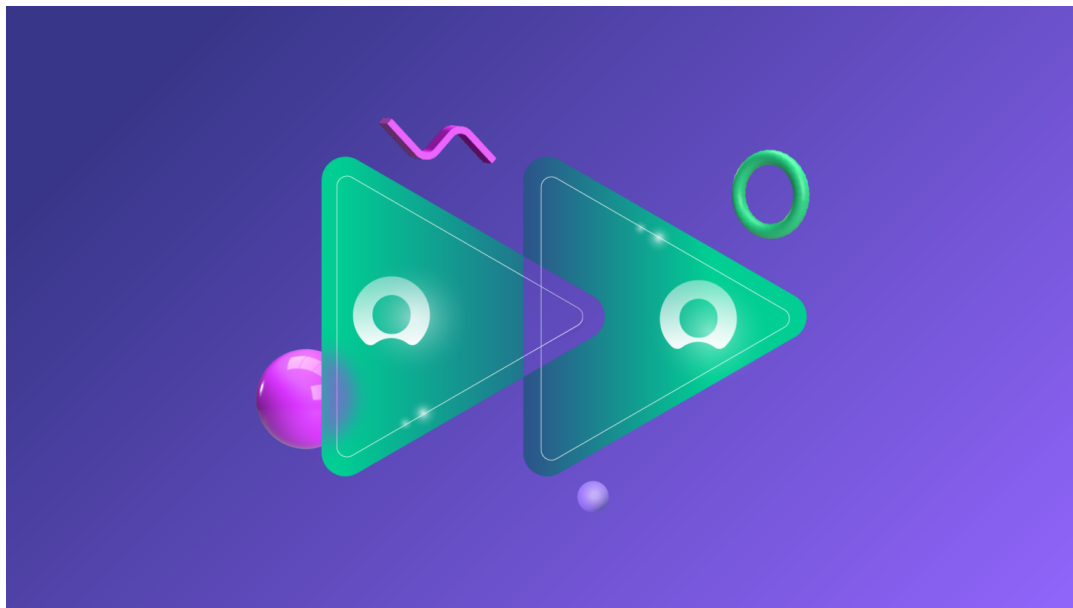


Table of contents

Common Use Cases for a ServiceNow to ServiceNow Integration

- Why Integrate Multiple ServiceNow Instances
- Use Cases

How to Set up a ServiceNow to ServiceNow Integration in 5 Steps

- Step 1 - Install Exalate
- Step 2 - Connect Your ServiceNow Instances
- Step 3 - Configure Your Connection to Share the Right Data
- Step 4 - Create Automated Synchronization Triggers
- Step 5 - Start Task Synchronization

Common Pitfalls to Keep in Mind When Integrating Multiple ServiceNow Instances

- Notification Overload
- Role Clarification

Conclusion



ServiceNow is one of the popular work management systems that allows you to store all kinds of useful customer data. If different teams (working in ServiceNow) have their own instances, then they can easily manage that data independently while making it available to the other teams through a ServiceNow to ServiceNow integration.

Such integration can exchange data between ServiceNow instances automatically. You can control what you send, and set the conditions for data transfer. That opens up all kinds of possibilities for teams wanting to help each other work more effectively, and can also make existing data transfer tasks faster, cheaper, and more reliable.

Here's a preview of what is covered in this blog post:

- [Common Use Cases for a ServiceNow to ServiceNow Integration](#)
- [How to Set up a ServiceNow to ServiceNow Integration in 5 Steps](#)
- [Common Pitfalls to Keep in Mind When Integrating Multiple ServiceNow Instances](#)

Common Use Cases for a ServiceNow to ServiceNow Integration

Why Integrate Multiple ServiceNow Instances

Data is critical in many business areas and there are all kinds of scenarios where you can benefit from sharing it through [B2B integration](#) or [eBonding](#).

By making data available to different teams, you prevent work from being repeated but allow specialists to apply their own expertise to common tasks.

servicenow  **servicenow**

Teams may have some data they want to keep private and some they want to share, so simply copying over the data wouldn't work. You need to filter it effectively so it only shares what you want it to. This way you maintain your customers' privacy and stay compliant with data sharing regulations.

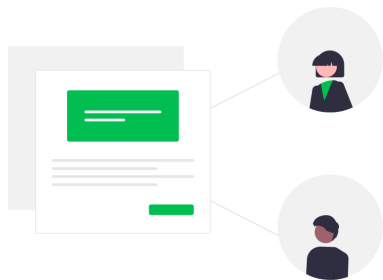
By implementing a ServiceNow to ServiceNow integration, teams can take advantage of each other's work without stepping on each other's toes.

Use Cases

Let's have a look at a couple of specific examples of situations where you can benefit from this integration.

Quality Control and Customer Support

Your customer support team deals with incidents and problems reported by clients. They file these in ServiceNow and keep track of related customer communications. Quality Control tracks product issues themselves, in their own system.



Both of these teams are interested in some, but not all of the other team's data. With a ServiceNow to ServiceNow integration, you can exchange the appropriate items, along with the fields that are pertinent to each team's work.

Consolidating Regional Databases

If you have multiple sales teams running their own customer databases, then you have a wealth of data that can be combined and studied. To your data analysts, this information is a treasure trove of insights waiting to be uncovered.



Using integrations, you can move data from regional servers to a common data store for your analysts to use to figure out which customers are providing the most value, and which marketing actions are most effective. They can then share that data with your regional teams.

How to Set up a ServiceNow to ServiceNow Integration in 5 Steps

Now you'll learn how to set up an integration between multiple ServiceNow instances.

The tool I'll use to do this is called [Exalate](#). Exalate is designed to meet the challenges you encounter when setting up a potentially complex data integration.

Firstly, it is **reliable**, meaning you don't have to worry about fixing it when something goes wrong. If one end of your connection experiences an outage, it can handle it and will start working again automatically when the connection is restored.

Secondly, it supports **decentralized integration**. You can manage your own end of the connection autonomously, controlling what is allowed out, and how what comes in is mapped to items on your system.

Thirdly, it is **flexible**, giving you easy ways to decide what is exchanged, how it is mapped between instances, and when data exchange takes place.

So let's see how a ServiceNow to ServiceNow integration is implemented, step by step.

Step 1 - Install Exalate

The first step is to install Exalate on your ServiceNow instances. If connecting two instances, you need to repeat this step for each of them.

You can also set up an integration on a single server if you need to exchange information between projects. That can be useful if your departments share a server. In that case, you'll get the option to create the connection locally in step two.

The easiest way to get started is to request an instance from Exalate. Alternatively, you can [install Exalate yourself via docker](#), though that can be complicated. I'll discuss the first way.

For more details on either method, [consult the documentation](#).

To request an Exalate instance, go [here](#) and select ServiceNow from the options available. If you have teams using other platforms, you can use Exalate to integrate them with ServiceNow too, so take a look at what else is available. Exalate works just as well with Jira, Azure DevOps, GitHub, Salesforce, Zendesk, and HP QC/ALM.

Reserve an Exalate for ServiceNow evaluation instance ✕

The Exalate for ServiceNow instance will allow you to setup a fully functional integration between your ServiceNow environment and any other supported tracking platform.

Enter Organisation*

Enter First name

Enter Last name*

Enter Phone number*

Enter Email address*

SUBMIT

We respect your privacy - more details [here](#)

Click ServiceNow on the [integrations page](#) and a form will appear. Fill in the details and click “Submit”. You’ll get an email containing the URL of your new node, along with an evaluation key.

Back in your ServiceNow account, you need to create a proxy user with the correct permissions, which you can [read about here](#). You’ll also need the Role Management V2 REST API plugin available and active. It’s included from the ‘New York’ version of ServiceNow onwards.

Now access the Exalate node, using the URL you got in the email. You’ll need to accept the EULA. After that, provide the node with the details it needs to connect to your ServiceNow instance. Enter the URL, along with your proxy account details. You’ll also need to provide your evaluation key, which was also in the email you received.

Now you’re ready to move to the next steps. You’ll need a ServiceNow account with administrative access to proceed.

Step 2 – Connect Your ServiceNow Instances

To connect your ServiceNow instances, you need to generate an invitation code on one end and paste it into the other. That creates a link between the two, that you can then configure to exchange the data you want.

Connections

Connection defines synchronization behavior, including communication details, sync rules, and scope.

Initiate connection

Accept invitation

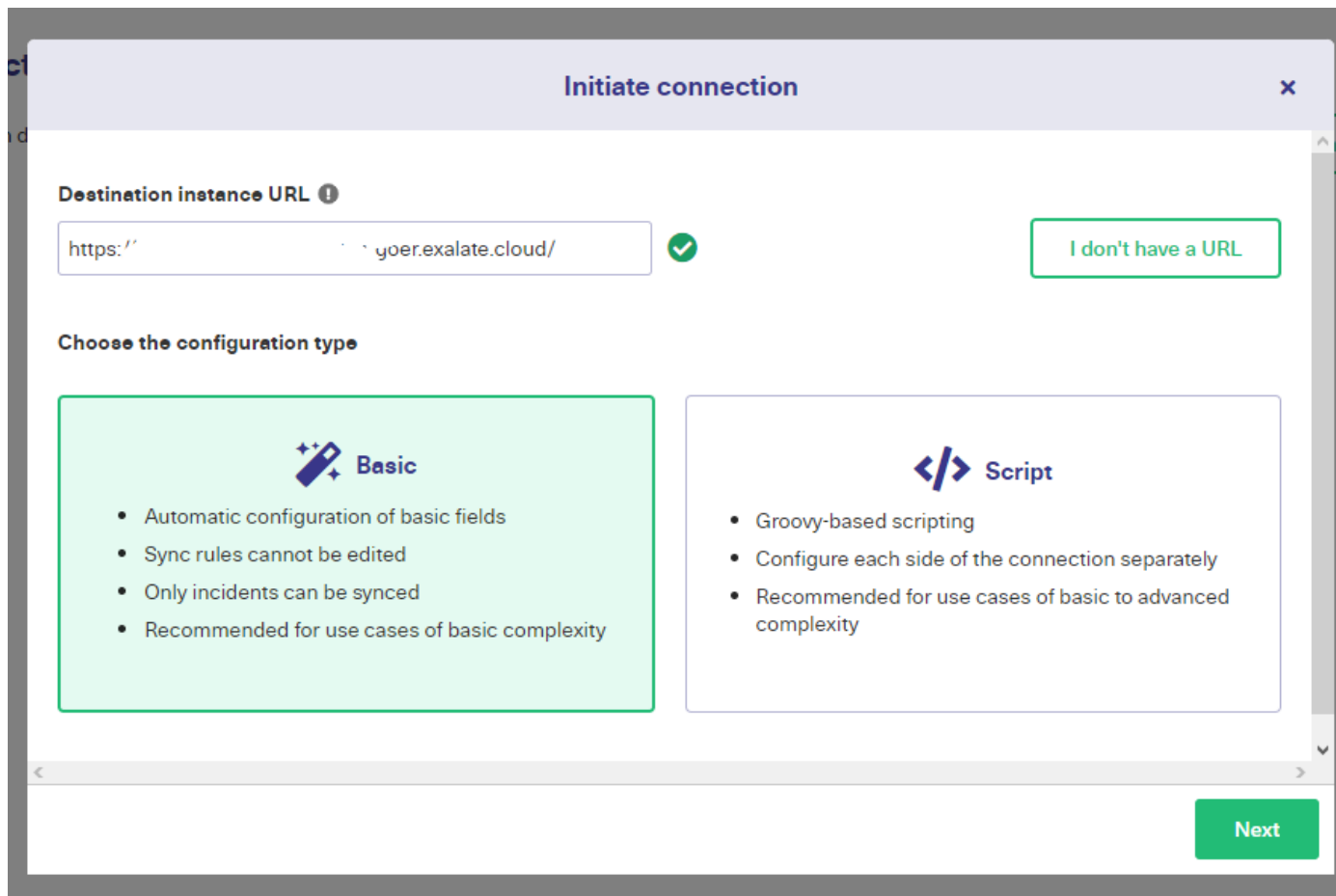


Connection	Entities under sync	Last sync	Status
dev67732_to_ven03945	1	incident INCO0... 9 minutes ago	Active
Sales_node_to_Engineering_node This is a sample SNOW-to-SNOW connection	0		Active



Open your Exalate node in your browser and navigate to the Connections screen via the left-hand menu. Click the “Initiate connection”.

Enter the URL of the instance you want to connect to. Exalate will look for it, and check Exalate is also installed there. When it finds it, you’ll be given the option to proceed using the **Basic mode**, or **Script mode**.



The screenshot shows a dialog box titled "Initiate connection" with a close button (x) in the top right corner. Below the title bar, there is a section for "Destination instance URL" with an information icon (i). A text input field contains the URL "https://goer.exalate.cloud/" and has a green checkmark to its right. To the right of the input field is a button labeled "I don't have a URL". Below this is a section titled "Choose the configuration type" with two options:

- Basic** (highlighted with a green border):
 - Automatic configuration of basic fields
 - Sync rules cannot be edited
 - Only incidents can be synced
 - Recommended for use cases of basic complexity
- Script**:
 - Groovy-based scripting
 - Configure each side of the connection separately
 - Recommended for use cases of basic to advanced complexity

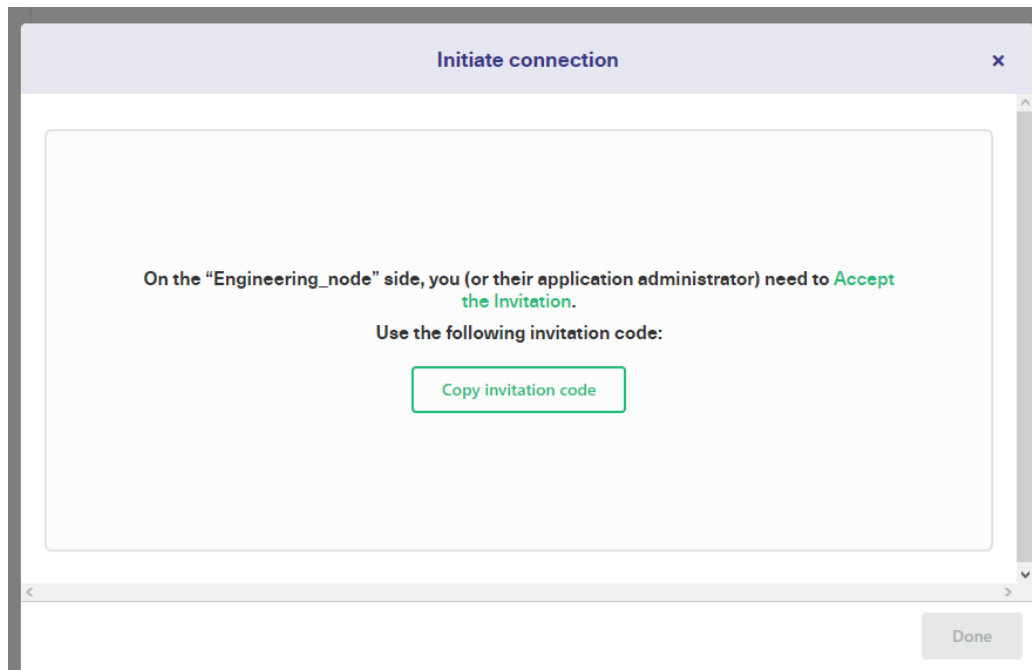
At the bottom right of the dialog is a green "Next" button.

The Basic mode is free and configures everything for you automatically. It's great when you don't want to worry about the details, and also lets you test everything out easily.

The Script mode is more advanced and lets you decide the details of how things are shared. You can specify what fields are shared, how they are mapped and set the conditions for sharing.

It's more involved than basic mode, and it helps if you're familiar with simple scripting or programming. There's a 30-day trial if you want to test it out.

Let's start with the Basic mode and return to Script mode later. Click "Basic" and click "Next".



You will then be asked to choose whether you have admin access to the destination ServiceNow instance. Click "Yes, I have admin access" if you have access. Else click "No, I don't have admin access."

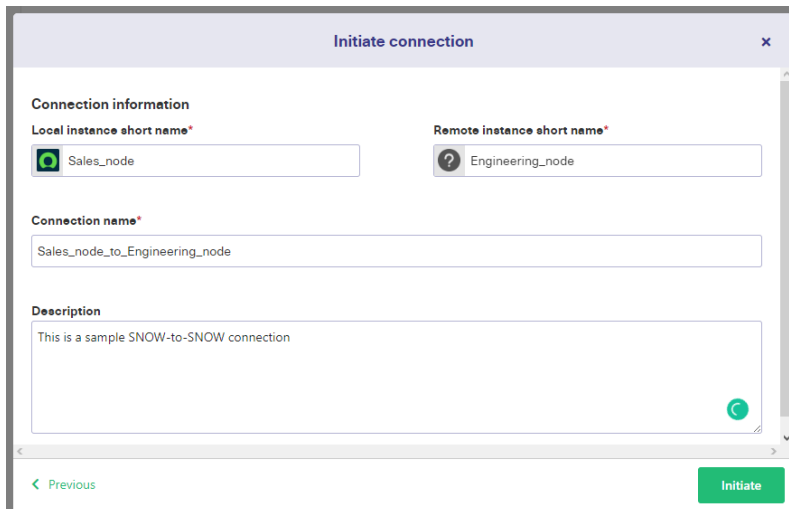
In case you don't have access, you can generate a code to paste into your other instance. Click "Copy invitation code" to copy the code to your clipboard as shown in the image above. Paste it somewhere safe, you'll need it later.

Click "Go to remote" to access your other instance or navigate there directly. On the connections screen, click "Accept invitation". You'll see a text field where you can paste the code you just generated. Do that, and click "Next".

After a brief wait, Exalate will establish the connection. To give you a taste of how it works, you can enter an incident number to test the synchronization.

Now let's try setting up a connection in Script mode.

Initiate a connection using the green button as before, but on the mode selection screen, choose the Script mode before clicking next.



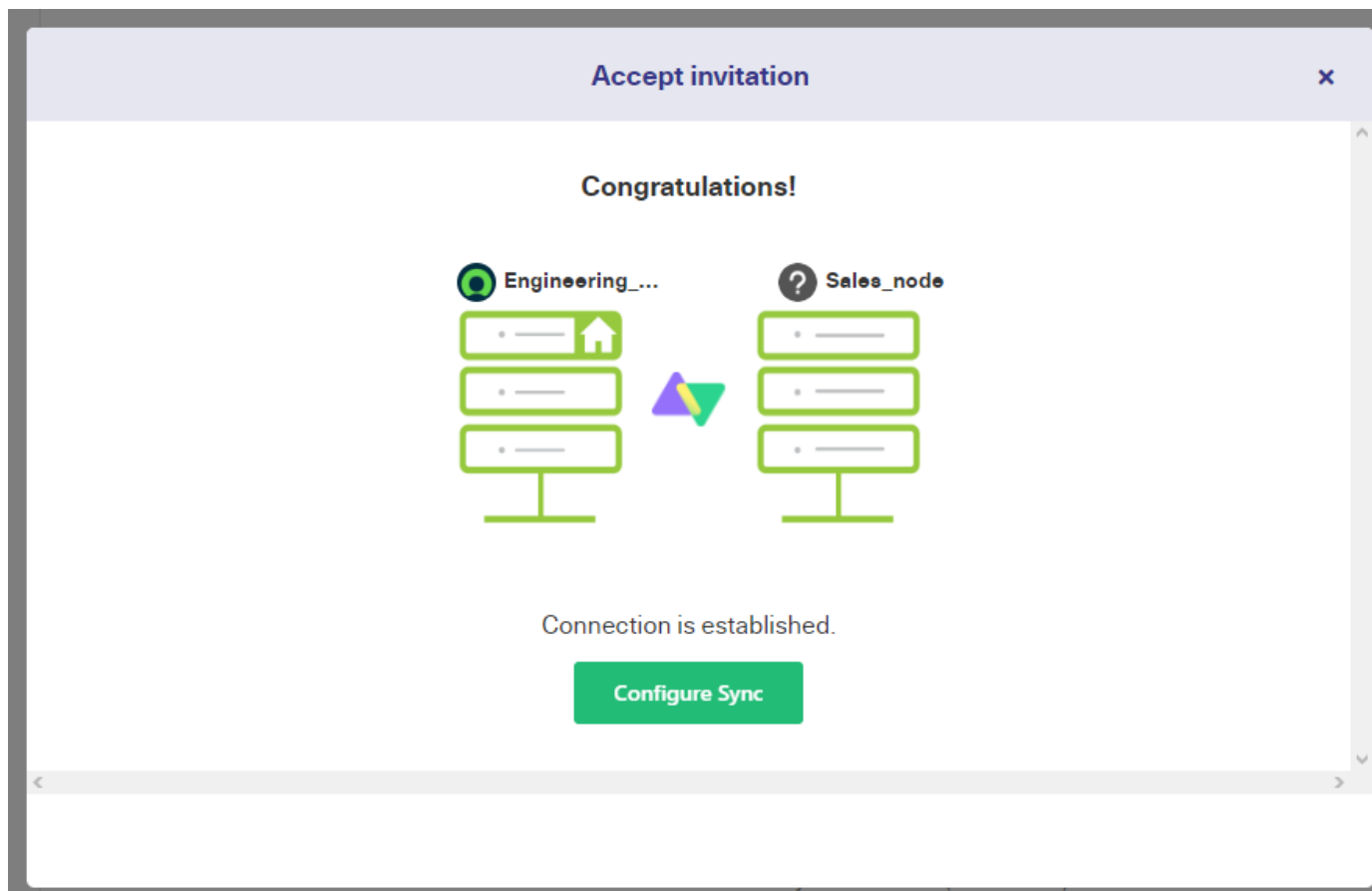
The screenshot shows a dialog box titled "Initiate connection" with a close button (X) in the top right corner. The dialog is divided into several sections:

- Connection information:** This section contains two dropdown menus. The first is labeled "Local instance short name*" and has "Sales_node" selected. The second is labeled "Remote instance short name*" and has "Engineering_node" selected.
- Connection name*:** A text input field containing the text "Sales_node_to_Engineering_node".
- Description:** A text area containing the text "This is a sample SNOW-to-SNOW connection".

At the bottom of the dialog, there is a "< Previous" button on the left and a green "Initiate" button on the right.

This time, you'll get more advanced options to choose from. Firstly, decide what you want to name each side of your connection. The names you choose will be combined automatically to create a connection name, but you can overwrite that if you prefer.

You can also add a description which is highly recommended. If you have multiple connections, or several people use the same connection, the description can help identify it. That's very helpful when returning later to modify it.



Click “Initiate” when you’re ready. You’re then given a code to copy as before, so follow the same steps. When the connection is established, Exalate will let you know. Click “Configure Sync” to choose some of the details that control how your connection behaves.

» Sales_node_to_Engineering_node

● Active

[← Back to Connections](#)

[Publish](#)

Rules

Triggers

Statistics

Info

▼ Outgoing sync ⓘ

```
1 if(entity.tableName == "incident") {
2   replica.key           = entity.key
3   replica.summary      = entity.short_description
4   replica.description   = entity.description
5   replica.attachments  = entity.attachments
6   replica.comments     = entity.comments
7   replica.state        = entity.state
8
9   /*
10  Use a field's internal name to send its value
11  Example: Resolution Notes -> resolution_notes
12  This works for all other entity types as well
13
14  replica.resolution_notes = entity.resolution_notes
15  */
16 }
17 //any other entity can be synced using the table name and the entity variable
18 if(entity.tableName == "cmdb_ci_business_app") {
19   replica.key           = entity.key
20   replica.summary      = entity.short_description
21   replica.description   = entity.description
22   replica.name         = entity.name
23 }
24
```

[Copy outgoing sync processor to clipboard](#)

When items are synced, they are copied from one side of your integration to the other. Items contain multiple fields.

You can choose which ones to sync, and can also set specific values if you prefer. For example, you might want to mark a field as 'synced from the Sales team'. You can also use code to create advanced rules or to conditionally determine what is shared.

The rules use the [Groovy](#) scripting language.

The outgoing sync rules define how items on the instance you are looking at are sent over to the other instance. The incoming rules define how the data your instance receives are mapped onto the synced items on your side.

Each line corresponds to a field. For example, in the outgoing sync, you can see the line `replica.description = entity.description`.

That means the description will be copied from items on this node to the corresponding items on the other node. If you don't want that to happen, just delete the line.

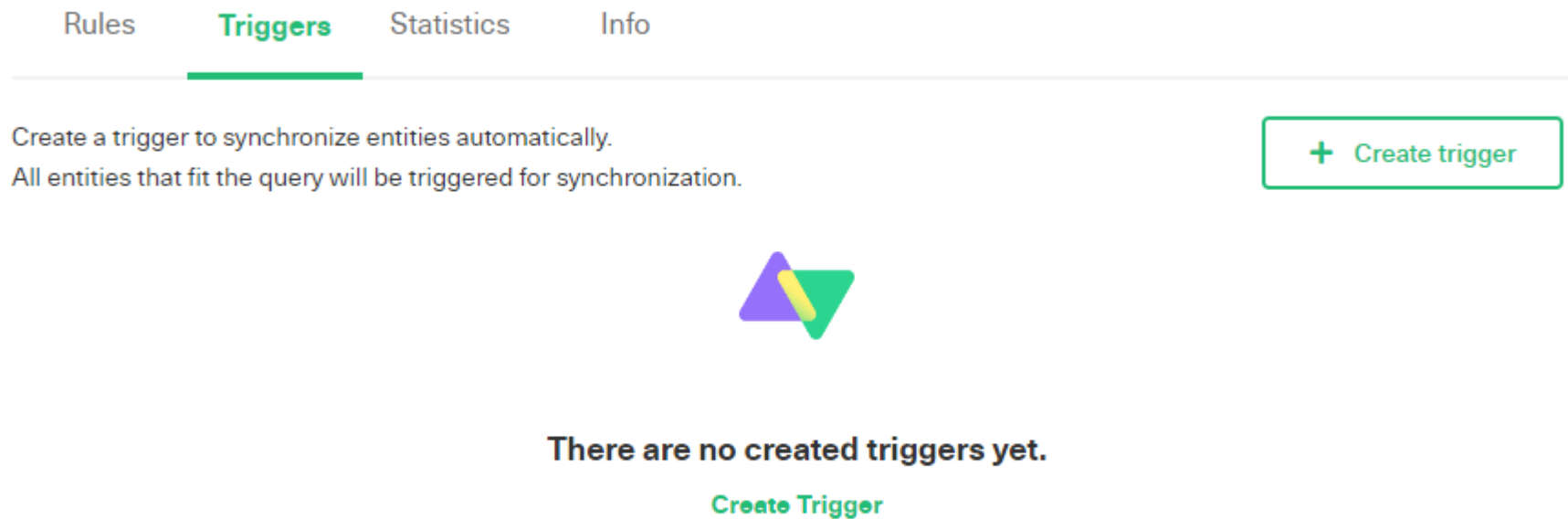
If you want synced items to have a fixed description, you could change it to `replica.description = 'synced from sales team'`

You could even change it to add that text at the beginning of the existing description with `replica.description = 'synced from sales team' + entity.description`

There are also helper functions you can use to manage items like comments and attachments. For more on those and other things you can do with sync rules, take a look at [Exalate's documentation](#).

Step 4 - Create Automated Synchronization Triggers

Synchronization triggers determine when synchronization takes place. To create one, you define a condition using [ServiceNow search syntax](#). Items that meet the condition are synced.



Rules **Triggers** Statistics Info

Create a trigger to synchronize entities automatically.
All entities that fit the query will be triggered for synchronization.

[+ Create trigger](#)

There are no created triggers yet.
[Create Trigger](#)

To get started, go to the “Triggers” tab and click “Create trigger”. On the dialog box, you can choose the type of entity the trigger applies to, using the dropdown box.

Add trigger ✕

Specify a ServiceNow search query to synchronize entities automatically. All entities that fit the query will be triggered for synchronization. [Find more details.](#)

Trigger will apply to selected entity type* ⓘ

incident | ▾

If* ⓘ

urgency=1 G

Notes

This trigger sets the urgency of the sync G

Active?

Add

There's also a field to enter your condition. Entities that meet this condition will be synchronized.

The suggested example is `urgency = 1`, so any items that have their urgency attribute set to one will be exchanged with the other side. You can use any field to decide whether to sync.

The system is highly flexible. You could sync tickets that are assigned to a specific person, sync tickets with comments, or have specific text in their description.

There's also a field to make notes. Again, it's a good idea to do so. It will help you keep track of things if you add more triggers, connections, and users later.

Finally, there's a switch to activate the trigger. Naturally, you'll want to activate this if you want anything to happen. You can switch triggers on and off, which is useful if you have one you want to apply temporarily, but regularly. Perhaps you want to share work with another department when things get busy, for example.

When you're done, click "Add". The trigger will be created, and you'll see it in the list.

For each entry in the list, you can edit or delete it using the icons on the right. You can also click the dots on the right and select "Bulk Exalate" which will tell you if any items match the query and synchronize them for you. That's a great way to test if your triggers are working as you expect.

Step 5 - Start Task Synchronization

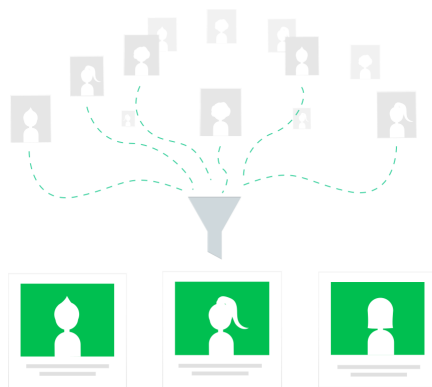
Now your instances are connected and will automatically exchange information at regular intervals. It isn't instantaneous, as that would create performance issues, so grab a coffee, come back, and see if items are exchanged. If it doesn't happen within an hour or so, have a look, and check that there are items matching the conditions you have set.

Once it works, you can sit back and let Exalate do the hard work of synchronizing your data. Enjoy!

Common Pitfalls to Keep in Mind When Integrating Multiple ServiceNow Instances

Notification Overload

When setting up a brand-new integration, it can be tempting to share as much information as possible. Be cautious. Filtering what people see is important. If people keep getting alerts when synchronizations happen, they might start ignoring them.



If people are only alerted to the items they need to deal with, they are more likely to check their notifications and spend their time on things that are relevant to them.

Role Clarification

When integrating systems there is a risk that you can go too far and end up with teams performing the same tasks. It is important to clarify who is responsible for what, and ensure work isn't repeated.

If there's a risk of overlap, make sure your integration labels items appropriately, so everyone knows who is ultimately responsible for each one. Also, ensure progress-related data in areas of overlap is shared and not stored exclusively in different systems, so people on different teams can see what has already been done.



Note: *Exalate lets you filter items using the username field, and you can use the configuration to have separate behavior for teams or individuals, so you can control synchronization on a very granular level if you need to.*

Conclusion

Setting up a ServiceNow to ServiceNow integration is easy, provided you use the right tools, and the rewards are more than worth it. By connecting your teams, you can make sure everyone in your organization benefits from sharing knowledge.

With Exalate, you can be connected in minutes, and have complete control over what you share and when you share it. It's fast, efficient, and reliable. You can leave it to work autonomously, and easily update it when you want to make changes.

Recommended Reads:

- [ServiceNow Integrations: Integrate ServiceNow and Other Systems Bidirectionally](#)
- [How to Set Up a Salesforce ServiceNow Integration](#)
- [Jira ServiceNow Integration: How to Set up an Integration in 6 Steps](#)
- [Zendesk ServiceNow Integration](#)
- [How to Set up a ServiceNow GitHub Integration](#)
- [How to Set Up an Azure DevOps ServiceNow Integration](#)