



## Salesforce to Salesforce Integration: Sync Multiple Salesforce Instances Bidirectionally



## Table of contents

### Why Integrate Multiple Salesforce Instances?

- Salesforce to Salesforce Integration: Common Use Cases
- Sales and Marketing Teams
- Sales and Customer Service Teams
- Grow and Diversify your Partners and Business

### 2 Ways to Integrate Multiple Salesforce Instances

- Salesforce to Salesforce: The Native Salesforce Functionality
- Exalate App for a Salesforce to Salesforce Integration

### How to Set up a Salesforce to Salesforce Integration in 5 Steps

- Step 1: Install Exalate on Salesforce
- Step 2: Connect your Salesforce Instances
- Step 3: Configure the Connection to Share Data
- Step 4: Create Automated Sync Triggers
- Step 5: Start the Synchronization

### Things to Consider when Using Exalate

- Defined User Roles and Responsibilities



- Access to the Exalate app
- Notification Handling

Conclusion



Over the years, CRM and Salesforce have become companions, inseparable and tightly bonded. The number of capabilities Salesforce offers towards CRM is huge and proof enough to leverage it for enhanced functionalities, like a Salesforce to Salesforce integration between partners or other businesses using the platform.

Such integration would allow sharing data (for instance: Leads and Opportunities) and its updates between different Salesforce instances. This will help achieve close collaboration with different partners or businesses using Salesforce, in turn, helping you grow your customer experience manifold.

So in this article, we aim to target the benefits of this setup by having a look at a few use cases. And then, we will see how we can achieve a decentralized Salesforce-to-Salesforce integration using a tool called [Exalate](#).

Jump to:

- [Why Integrate Multiple Salesforce Instances?](#)
  - [Common Use Cases](#)
- [2 Ways to Integrate Multiple Salesforce Instances](#)
- [How to Set up a Salesforce to Salesforce Integration in 5 Steps](#)
  - [Continue with the Basic Mode](#)
  - [Continue with the Script Mode](#)
- [Things to Consider when Using Exalate](#)

## Why Integrate Multiple Salesforce Instances?

In today's digital world, data is gold, and if mined properly it can help different teams share expertise and grow together through common business-critical information.

But the fundamental problem lies when this information is passed manually between them. It leads to discrepancies and errors. Plus, you need to filter the exchanged data, as you don't want to send across all the information. There is certain information you would rather keep within your Salesforce instance; to comply with your security or data regulatory requirements. So much filtering if done manually can become cumbersome and lead to friction among team members.

To avoid that, a Salesforce to-Salesforce integration seems like the way out.



It will deal with all the above problems like a cakewalk, providing along the way a few extra toppings like secure, automatic, filtered, reliable, and real-time bi-directional information exchange. All this, so the other party has the most accurate representation of information it wants to see and doesn't receive any unauthorized information.

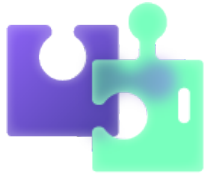
To elaborate a little on how exactly it will be beneficial, let's have a look at a few scenarios where exchanging information between multiple Salesforce instances can help companies collaborate and communicate effectively.

## Salesforce to Salesforce Integration: Common Use Cases

### Sales and Marketing Teams

Sales and marketing are probably the teams in your organization that use Salesforce day in and day out. But what these teams probably lack is collaboration on common customer enhancement goals with each other; as they might be stuck up in their respective

Salesforce instances.



An integration in such a case would help to share customer information in the form of Accounts, Opportunities, Leads, etc, and design campaigns or other marketing initiatives around them. The treasure house of information within individual Salesforce instances can be used by both these teams and prove beneficial to open up new avenues for enriching customer experience.

### **Sales and Customer Service Teams**

Customers are and have always been the focus of businesses worldwide. To enrich and nurture your relationship further with your customers, an integration between the Salesforce instances used by the sales and customer service teams can definitely work in a positive manner.

Cases, Tasks, and Opportunities always have valuable insights into customer behavior, feedback, problems, and queries. So getting early access to such information and proactively dealing with it can only leave you with happier customers.

### **Grow and Diversify your Partners and Business**

Allowing multiple Salesforce organizations (Salesforce instances) to integrate would mean expanding your partnerships with newer business interests; in turn, targeting a wider customer base and increasing the services to offer.

This can also allow vendors to diversify their business operations, wherein such different partners could bring in their expertise to provide a package of services to their customers.

This brings us to the point of reminiscing about ways to achieve such a Salesforce to Salesforce integration.

## 2 Ways to Integrate Multiple Salesforce Instances

### Salesforce to Salesforce: The Native Salesforce Functionality

Well, Salesforce is well-known for its customizability to suit almost any business case and model. As such it has already envisioned the future of integration and has brought to you “Salesforce to Salesforce”; a service that offers to connect 2 Salesforce organizations (or instances).

But to use this feature there are certain points to consider like: the ability to share a maximum of 100 tasks per related Salesforce record. There are also other guidelines you can refer to [here](#).



This is a great option if you have simple out-of-the-box use cases. But if you have custom or advanced integration needs then you can achieve that through REST API callouts using [Apex](#).

Now here is where the real challenge lies.

Apex is a powerful medium that allows you to extend the native capabilities of Salesforce to include custom and advanced business logic, but it boils down to implementing an integration, which honestly is too much work.

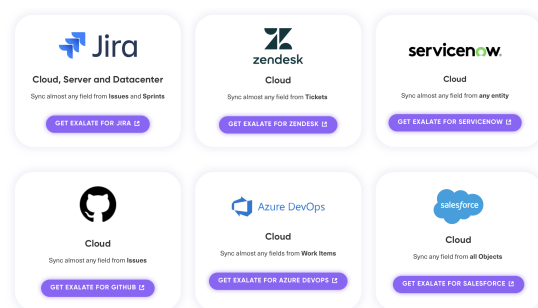
This is one of the things you might want to consider if your teams do not have the necessary technical resources to carry it forward. Also, Apex isn't really optimized nor designed for the complex integration processing that is expected from typical integration tools; not to mention that it is subject to [governor limits](#) defined by Salesforce.

In lieu of this, [AppExchange](#) from Salesforce has ready-made apps that help remove the sting of implementing an integration and allow you to integrate multiple Salesforce instances in a graceful manner. So here is a tool that can checkmark almost all your integration needs.

## Exalate App for a Salesforce to Salesforce Integration

I will use an integration solution called [Exalate](#) for the purpose of this article. It was designed specifically with **decentralized integration** in mind and allows you to bidirectionally sync information in real-time between 2 (or more) Salesforce instances or between Salesforce and other applications like Jira, GitHub, ServiceNow, Zendesk, HP ALM, etc.

You can have a look at Exalate's [integrations](#) page for more info.





So let's see why Exalate would be the right choice to handle a complex Salesforce to Salesforce integration:

- Perhaps, we will get straight to the point; what is **decentralized integration**?  
This means that each side will independently control what it sends to the other side and how it wants to receive information from the other side. An effect of this arrangement is that it ensures unauthorized Salesforce users don't get access to shared information, because you simply choose what to send (or receive) over and what not to.  
Since it has **distributed architecture**, the systems are **loosely coupled**, making your integration more maintainable and scalable.
- It provides **secure** information exchange through role-based access controls, encrypted information exchange, use of HTTPS, etc. You can learn more about its security features in this [whitepaper](#).
- It has an intuitive scripting engine that allows you to integrate with even the most advanced or complex integration use cases. So in this way, it is **flexible**.
- You don't have to worry about downtimes or outages with Exalate in place; it ensures **reliability**. It will gracefully resume operations in case of breakdowns and ensure the information keeps flowing from the point of interruption without the need for manual intervention.

But the question still remains, how do you achieve such a Salesforce to Salesforce integration using Exalate?

So let's see how!

## How to Set up a Salesforce to Salesforce Integration in 5 Steps

### Step 1: Install Exalate on Salesforce

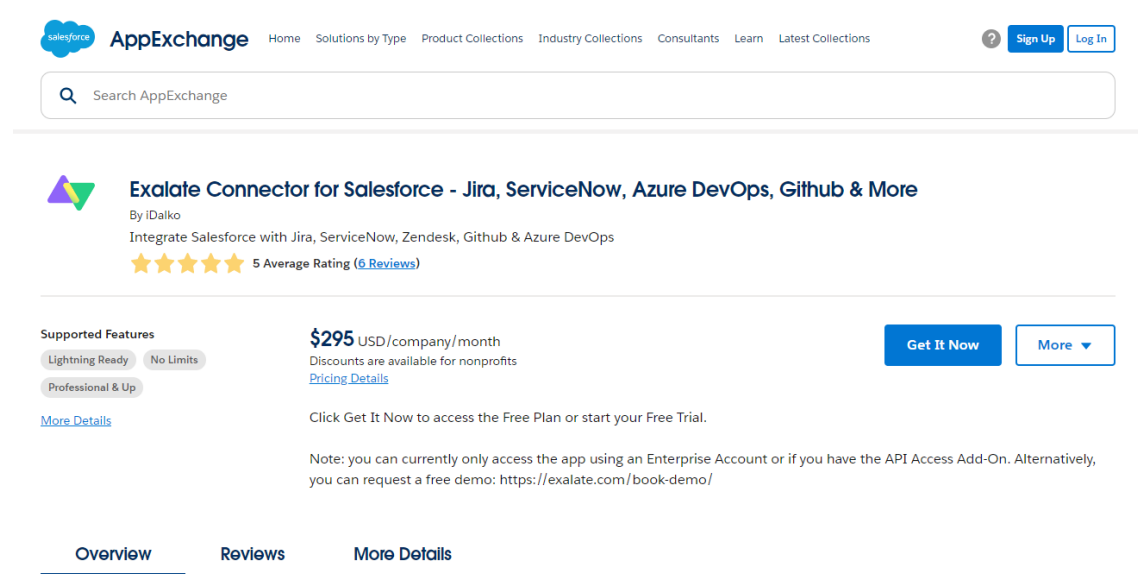
The first step is to install Exalate on Salesforce. Since we are integrating two Salesforce instances, you would need to install it on the other instance as well.

You can find the installation steps [here](#), but I will also discuss them briefly in this section.

**Note:** Exalate accesses Salesforce through APIs. Salesforce has got its own [guidelines](#) for API Access add-ons. For example, API access is provided by default in Enterprise accounts, but it is not the case with other accounts like Professional. Learn more about the different Salesforce editions Exalate supports on this documentation [page](#).

Head over to [AppExchange](#) by Salesforce. In the global search bar, search for Exalate. Choose Exalate from the drop-down list.

Click “Get It Now”.



The screenshot shows the Salesforce AppExchange page for the 'Exalate Connector for Salesforce - Jira, ServiceNow, Azure DevOps, Github & More' app. The page includes a search bar, navigation links, and a 'Get It Now' button. The app is by iDalko and has a 5-star average rating from 6 reviews. The price is listed as \$295 USD/company/month. The page also features a 'Supported Features' section with tags like 'Lightning Ready', 'No Limits', and 'Professional & Up'. A note at the bottom states that the app can only be accessed using an Enterprise Account or API Access Add-On, and provides a link to request a free demo.

AppExchange Home Solutions by Type Product Collections Industry Collections Consultants Learn Latest Collections Sign Up Log In

Search AppExchange

**Exalate Connector for Salesforce - Jira, ServiceNow, Azure DevOps, Github & More**  
By iDalko  
Integrate Salesforce with Jira, ServiceNow, Zendesk, Github & Azure DevOps  
★★★★★ 5 Average Rating (6 Reviews)

Supported Features: Lightning Ready, No Limits, Professional & Up

**\$295** USD/company/month  
Discounts are available for nonprofits  
[Pricing Details](#)

Get It Now More

Click Get It Now to access the Free Plan or start your Free Trial.

Note: you can currently only access the app using an Enterprise Account or if you have the API Access Add-On. Alternatively, you can request a free demo: <https://exalate.com/book-demo/>

Overview Reviews More Details

Then click “Install in Production” to install the package in the production environment.

Where do you want to install this package?

---

**Install in a Production Environment**

Install this package in the org where you or your users work, including Developer Edition orgs.

\* Connected Salesforce Accounts ⓘ

teja.bhutada@idalko.com ↕ ↻

Don't see your account? [More Info](#)

**Install in Production**

**Install in a Sandbox**


Test this package in a copy of a production org.

**Install in Sandbox**

**Cancel**


After this, "Confirm and Install".


Next, you need to choose the users' profiles that will have permission to access the Exalate app. Click on the permissions according to their roles. Don't worry if you can't decide immediately, you can [change the permissions](#) anytime later.




## Install ExalateBridgeApp

By Exalate

  
**Install for Admins Only**

  
**Install for All Users**


  
**Install for Specific Profiles...**

App Name	Publisher	Version Name	Version Number
ExalateBridgeApp	Exalate	1.1.0	1.1

**Additional Details** [View Components](#)

I have chosen “Install for Admins Only”. Once done, click “Install”.


Then approve access to 3rd-party websites and click “Continue”. You will be redirected to your Salesforce instance by clicking “Done”.



## Install ExalateBridgeApp

By Exalate

---

**Installation Complete!**

**Done**


---

App Name	Publisher	Version Name	Version Number
ExalateBridgeApp	Exalate	1.1.0	1.1

Now in Salesforce, access the Exalate app by searching for it in the “Apps” section. Then “Request Node” to request an Exalate node.

Then "Allow" the app for necessary permissions.

Now to activate an evaluation license and to verify your Exalate for Salesforce instance, fill in a simple form with your details. Click on "Agree and submit" after filling out the information.



company.tracker.com

Registration

### Registration

**Email \*** ⓘ

**Contact name \***

**Organization \***

**Phone \***

By clicking Agree and submit below, you agree to our end user license agreement - available [here](#)

[Documentation](#) [EULA](#) [Support](#) [Report a bug](#)

[Agree And Submit](#)

Head over to your inbox. Open the email you have received from the Exalate License manager and click “Verify Exalate instance”.



Hi,

Thank you for trying out Exalate

Click here to verify your Exalate node:

Verify Exalate instance

After verifying your instance, you will get an admin account with a 30-day evaluation license.

Use this link to access the Exalate admin console: <https://exalate.tracker.net>

Check out this short guide how to verify your Exalate instance: [How to verify your Exalate instance](#)

If you have any questions - please [raise a ticket on our service portal](#).

Enjoy,  
The Exalate team

If you received this message by accident, you can ignore this message, or [contact our support](#)

This will finish the installation.

**Note:** To log in to your Salesforce Exalate node, follow this [process](#).

Now proceed to the next step.

## Step 2: Connect your Salesforce Instances

To begin with, you need to connect both your Salesforce instances. This will help form a link between them. Once the link is established, you can proceed to configure the connection and share the required information.

For this, initiate the connection from one Salesforce instance and accept it on the other.











Go to the "Connections" tab in the left-hand menu of the Exalate console on any one of the Salesforce instances. It doesn't matter which instance you start from; Exalate has a uniform interface on all applications.

Then click on "Initiate Connection".

**Connections**

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

[Initiate connection](#) [Accept invitation](#) [Refresh](#)

Connection	Entities under sync	Last sync	Status
 Salesforce_to_Salesforce Please do not delete	5	Account Connect... 10 months ago	● Deactivated  
 SF_to_Jira desc	1	Case Connect... 10 months ago	● Deactivated  
 Salesforce_to_ADO	0		● Pending   



Enter the destination instance URL. This will be the URL of the other Salesforce instance. Copy it from the address bar and paste it into the text box shown in the image below.

After Exalate performs a check of whether it has been installed on the destination instance or not, it will give you further options to proceed. These options would include choosing the configuration mode.


**Connections**

**Initiate connection** ×


**Destination instance URL** ⓘ

ⓘ I don't have a URL

**Choose the configuration type**

 **Basic** ⓘ

- Automatic configuration of basic fields
- Sync rules cannot be edited
- Only Cases 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

Next

It supports 2 configuration modes: **Basic** and **Script**.

The Basic mode is free and is configured automatically for you. It works well for basic sync use cases and is quite suitable for testing the connection you have just established.

You can sync only “Cases” with the Basic mode. It also comes with a [Free Plan](#) that allows you up to 1000 free syncs per month.

The Script mode is what makes Exalate compatible with almost any scenario. It is more advanced and lets you decide what you want to share with the other side and how you want to map the information coming from the other side.

It comes in Groovy scripting language which is not very difficult to understand by non-technical users. There is also elaborate [documentation](#) on sample scripts that can come in handy when you get stuck.

We will cover this mode in a while. Let’s continue with the Basic mode for now.

### Continue with the Basic Mode

Once you select “Basic”, click “Next”.

You then need to verify if you have admin access to the other Salesforce instance. Don’t worry if you don’t. An invitation code will be generated. You can copy and paste it on the other side in this case.

I have access, so I will click on “Yes, I have admin access”. Then I will be redirected to the other Salesforce instance to verify the access and establish the connection.

On the next screen, you can test the connection you have just created by entering the Case number in the text box given and then clicking “Exalate”.

**Accept invitation** ✕

**Connection established successfully** ✓

Sync your first case to see how it works.

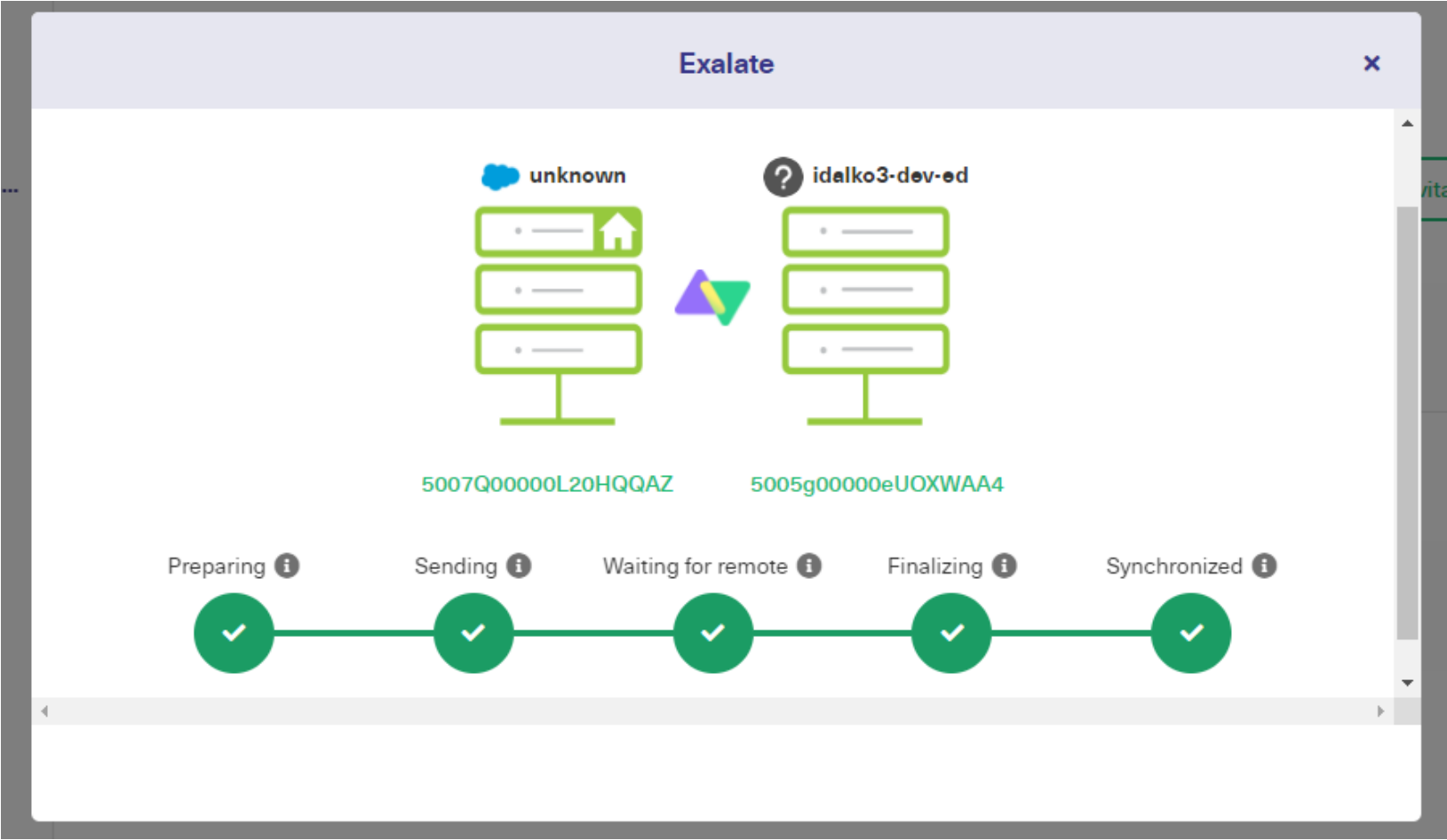
Please enter a case key to proceed

**Exalate**



**BOOK DEMO**

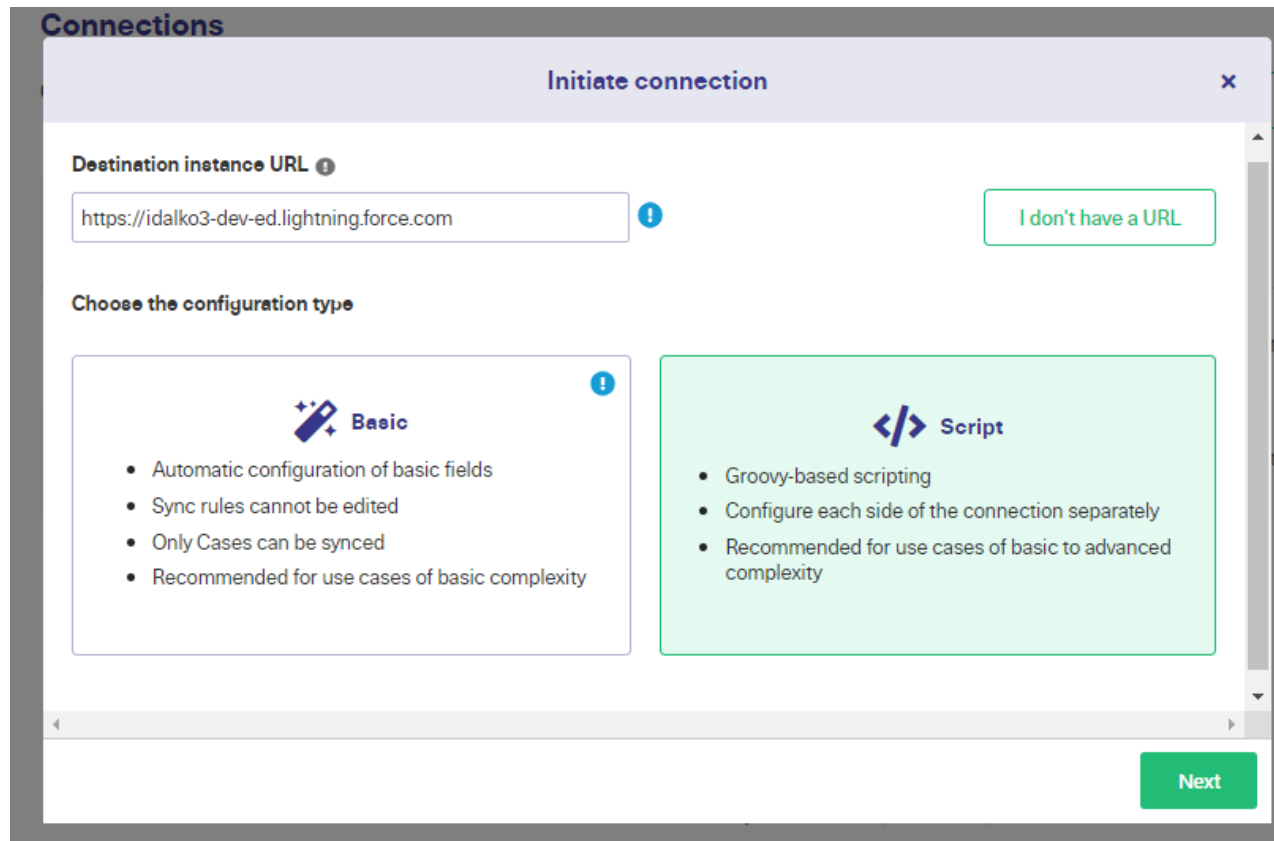
After a brief pause, you can see the Case getting synced to the other side. As shown in the image below, you can even click on the respective links and head over directly to the particular Case.



Now let's move ahead with the Script mode.

### Continue with the Script Mode

Initiate the connection in the same manner as you just did. But now instead of choosing "Basic", choose "Script". And then click "Next".



Now you need to enter a name for your local Salesforce instance and one for the remote instance. A connection name by combining both of these names will be auto-generated, but you can change it if you want.

### Initiate connection

**Connection information**

**Local instance short name\***

**Remote instance short name\***

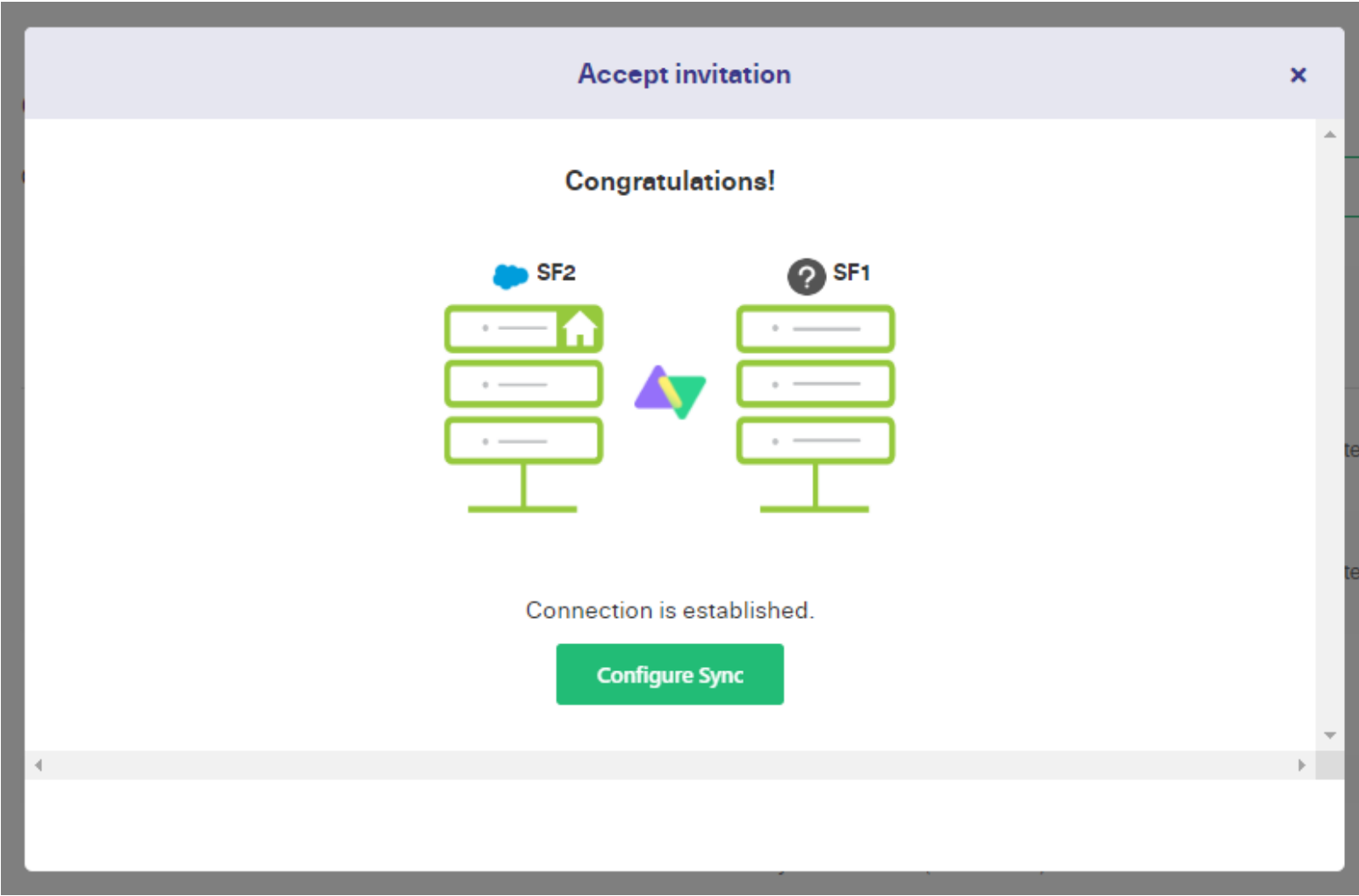
**Connection name\***

**Description**

[Previous](#) [Initiate](#)



In a few seconds, the connection is successfully created.





You can now configure it to decide what and how you want to send and receive information.

### Step 3: Configure the Connection to Share Data

You can continue to configure the sync by clicking the “Configure Sync” button on the screen above.

If you don’t choose to do it immediately, you can go back to your “Connections” anytime and click on the edit connection button in front of your connection name.

Both approaches will redirect you to a page that looks like shown below.

As seen, there are 4 tabs:

- Rules
- Triggers
- Statistics and
- Info

» SF1\_to\_SF2 Active [Back to Connections](#) [Publish](#)

Rules Triggers Statistics Info

▼ Outgoing sync

```
1 = if(entity.entityType == "Case") {
2   replica.key = entity.Id
3   replica.summary = entity.Subject
4   replica.description = entity.Description
5   replica.comments = entity.comments
6   replica.attachments = entity.attachments
7   replica.status = entity.status
8 }
```

Copy outgoing sync processor to clipboard

▼ Incoming sync

```
1 = if(firstSync){
2   entity.entityType = "Case"
3 }
4 = if(entity.entityType == "Case"){
5   entity.Subject = replica.summary
6   entity.Description = replica.description
7   entity.Origin = "Web"
8   entity.Status = "New"
9   entity.comments = commentHelper.mergeComments(entity, replica)
```

Statistics show how many items you are sharing and the details of when you have last synced them.

The Info tab gives information about the type of connection. You can find the destination instance URL here. You can edit the description of the connection here, too.

Moving to the “Rules” tab.

There are 2 sections under it. The “Outgoing Sync” decides what information should be sent and the “Incoming Sync” decides how you want to map the information coming from the other side.

Each line corresponds to a field you want to sync. For instance, `replica.description = entity.Description` in the Outgoing sync would mean that the description of the case is sent over to the other side. Replica acts like an empty placeholder to dump information you want to send to the other side. Likewise in the incoming sync you take contents from the replica and place it under the appropriate fields, i.e: `entity.Description = replica.description`, meaning that the replica’s description is copied to the description of the Salesforce entity.

Note that these rules exist at either end, so each side can decide independently of what it needs to send and receive. The syncs are written in [Groovy](#) scripting language.

To stop sending or receiving certain information, you can simply choose to comment or delete that particular line. So if you delete `replica.description = entity.Description` in the outgoing sync section, then it means the description won’t be synced to the other side.

You can even choose to have a fixed field description, for instance: `replica.description = 'synced from the sales team'`.

You can also add some text at the beginning of the description, for instance:

```
replica.description = 'synced from the sales team' + entity.Description
```

To send additional information, you can add scripts in the existing sync rules section.

For instance, if you want to send Tasks in addition to Cases from one Salesforce instance to the other, you can add the following script in the “Outgoing sync”:

```
if(entity.entityType == "Case") {
  replica.key          = entity.Id
  replica.summary      = entity.Subject
  replica.description  = entity.Description
  replica.comments     = entity.comments
  replica.attachments  = entity.attachments
  replica.Status       = entity.Status
}

if(entity.entityType == "Task") {
  replica.key          = entity.Id
  replica.summary      = entity.Subject
  replica.description  = entity.Description
  replica.Status       = entity.Status
}
```

Don't forget to map the corresponding Task details in the “Incoming sync” of the other Salesforce instance here.

This way, you can even set up the most advanced configuration between the instances.

There is no need to get bogged down by so much code. There are a lot of [script helpers](#) that can help you with advanced processing.

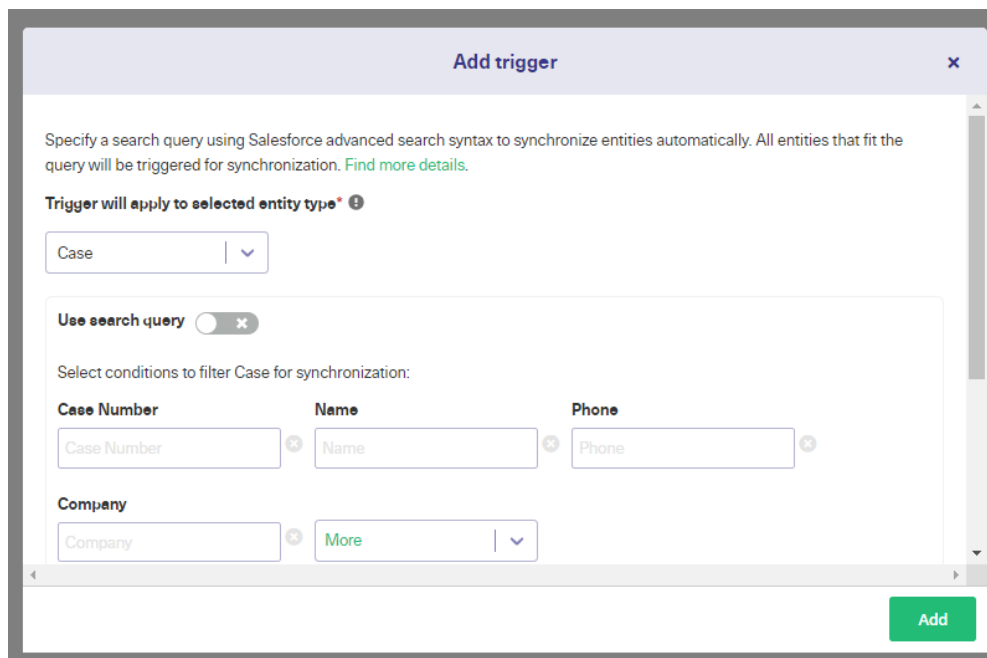
## Step 4: Create Automated Sync Triggers

Sync rules as seen above decide what information should be sent and received, whereas triggers decide when to send that information. They are conditions that we set if satisfied will send information according to the Sync rules.

To create triggers, click the “Triggers” tab as shown in the above section.

If this is your first time, the screen will be empty. So start creating a new trigger by clicking “Create trigger”.

The “Add trigger” screen allows you to create your trigger.



The screenshot shows the 'Add trigger' configuration interface. At the top, there is a title bar with 'Add trigger' and a close button. Below the title bar, there is a text instruction: 'Specify a search query using Salesforce advanced search syntax to synchronize entities automatically. All entities that fit the query will be triggered for synchronization. [Find more details.](#)'

Underneath, there is a section titled 'Trigger will apply to selected entity type' with a help icon. A dropdown menu is set to 'Case'. Below this is a 'Use search query' toggle switch, which is currently turned off.

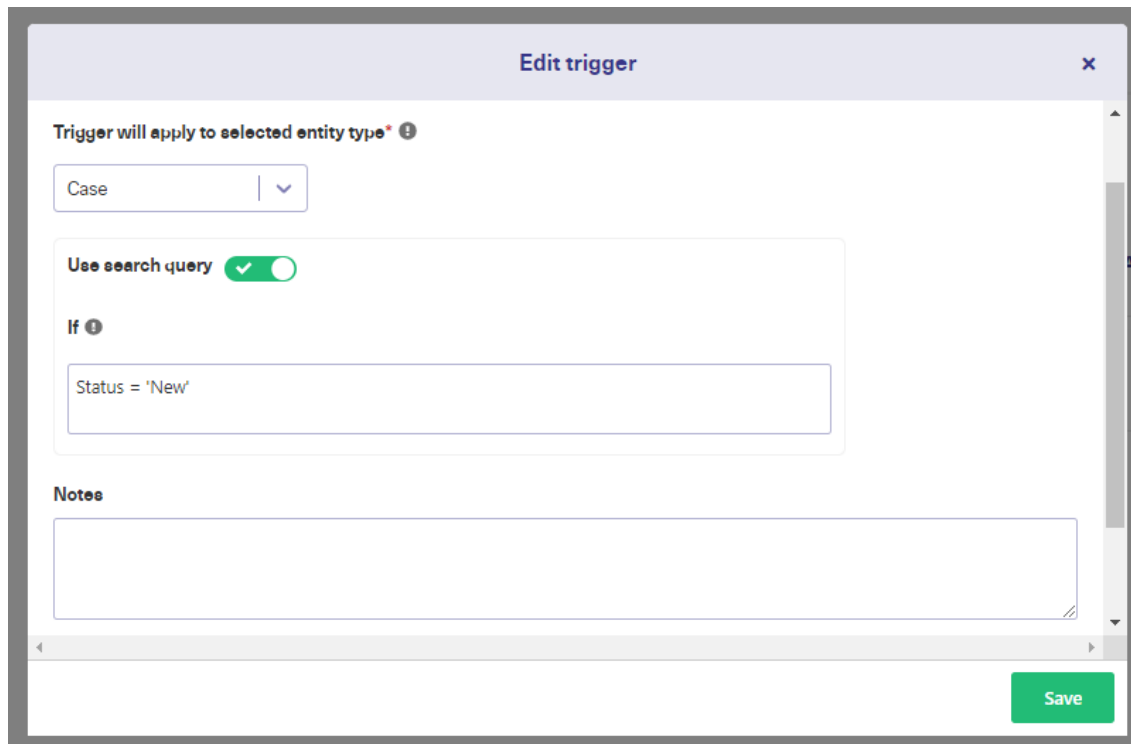
The next section is 'Select conditions to filter Case for synchronization:'. It contains two rows of conditions. The first row has three conditions: 'Case Number' (with a search box containing 'Case Number'), 'Name' (with a search box containing 'Name'), and 'Phone' (with a search box containing 'Phone'). The second row has two conditions: 'Company' (with a search box containing 'Company') and 'More' (with a dropdown arrow).

At the bottom right of the form, there is a green 'Add' button.

Select the Salesforce entities to which the trigger will apply, for instance, Case, Account, Product, Opportunity, etc.

Then in the next section, you can enter the details of the Case in the fields provided to create a trigger. There are a lot of options in the “More” section as well. For instance, you can choose to sync Cases belonging to a particular company, in which case you need to write the name of the company in the text box provided.

Or you can even create it by toggling the search query option, which will allow you to write a trigger in the Salesforce Object Query Language (SOQL).



**Edit trigger** x

Trigger will apply to selected entity type\* ⓘ

Case | v

Use search query

If ⓘ

Status = 'New'

Notes

Save

I have written a trigger to sync every case with the status 'New' to the other side.

Once you write your query, scroll down and fill in some notes for the trigger. This will help you identify the purpose of your trigger in case you have too many.

And then click "Active" to activate the trigger.

» **SF1\_to\_SF2**  
● Active

< Back to Connections

Publish

Rules




**Triggers**

Statistics

Info

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

+ Create trigger

When	If	Status	Action
Case Events: create/update	Status = 'New'	<input checked="" type="checkbox"/>	  

< 1 >

On the previous screen, you will be able to see your trigger listed. You can choose to edit or delete it from this screen. You can even choose to sync existing Salesforce objects fulfilling the condition of the trigger by clicking the three dots next to its name and selecting [“Bulk Exalate”](#).

And last but not least, the most important step is to “Publish” the changes you have made so they can be reflected in your sync.

## Step 5: Start the Synchronization

There are various ways you can start your synchronization with Exalate.

- You can use the Basic mode to sync as shown in step 2.
- You can create triggers as shown in step 5.
- You can “Bulk Exalate” Salesforce entities satisfying the trigger conditions.
- Or you can manually sync a particular entity by clicking on the “Exalate” [sync panel](#) in the Salesforce object page. You just need to select the connection name through which you want to sync and click “Exalate”. This panel will also show you the status of your synchronization.
- Another way to sync is using the [“Bulk Connect”](#) option that allows you to sync existing Salesforce records present in both instances.

This is all you need to know about how Exalate can help you achieve your integration.

## Things to Consider when Using Exalate

### Defined User Roles and Responsibilities

It is important to define user roles and responsibilities, especially in an integration setup. Such a filter is necessary to avoid friction between team members. As an appropriate measure you can label the items to be synced correctly and clearly, so everyone on the team stays on the same page and is aware of what needs to be done in case of a discrepancy or error.



With Exalate it is possible to filter items to even the most granular level. For instance, you can filter them according to the username field so things can be handled differently for teams or individuals.

## Access to the Exalate app

We need proper access control mechanisms to ensure authorized people have access to the Exalate app. This requirement is important in an integration setup to ensure your information does not end up in the wrong hands, or any data regulatory requirements are not violated. Such permissions are already set up while the app is being installed, but they can even be [changed](#) later if needed.

## Notification Handling

When dealing with newer integration apps like this, it is tempting to overload users with every little sync. However, we need to avoid this, as sometimes people can tend to overlook important notifications if they are overloaded with a lot of unnecessary ones.





A plan must be in place so that only the required and minimum number of users are notified in case something goes wrong, or to have updates about necessary syncs. This will ensure the app is used correctly and no one is missing important notifications.

## Conclusion

Integration is a tricky subject and can seem gigantic. Honestly, it is! But if handled carefully and with proper planning, it can make your life a lot easier.

We saw how this can be true for a Salesforce to Salesforce integration where you can expand your business to include newer partners and your business teams to work towards common customer goals.

We covered native ways to handle this integration. And then moved on to see how to get your integration to work the way you want it to, using a cross-company integration solution called Exalate.

And lastly, we saw how Exalate can be used to integrate multiple Salesforce instances in 5 easy steps and pointed out a few things that can make your integration path a smoother one.

### ***Recommended Reads:***

- [How to set up a Jira Salesforce Integration](#)
- [Salesforce Zendesk Integration: The Complete Guide](#)

- [How to Set Up a Salesforce ServiceNow Integration](#)
- [GitHub Salesforce Integration: How to Set up a Sync in 6 Steps](#)
- [How to Set up an Azure DevOps Salesforce Integration](#)