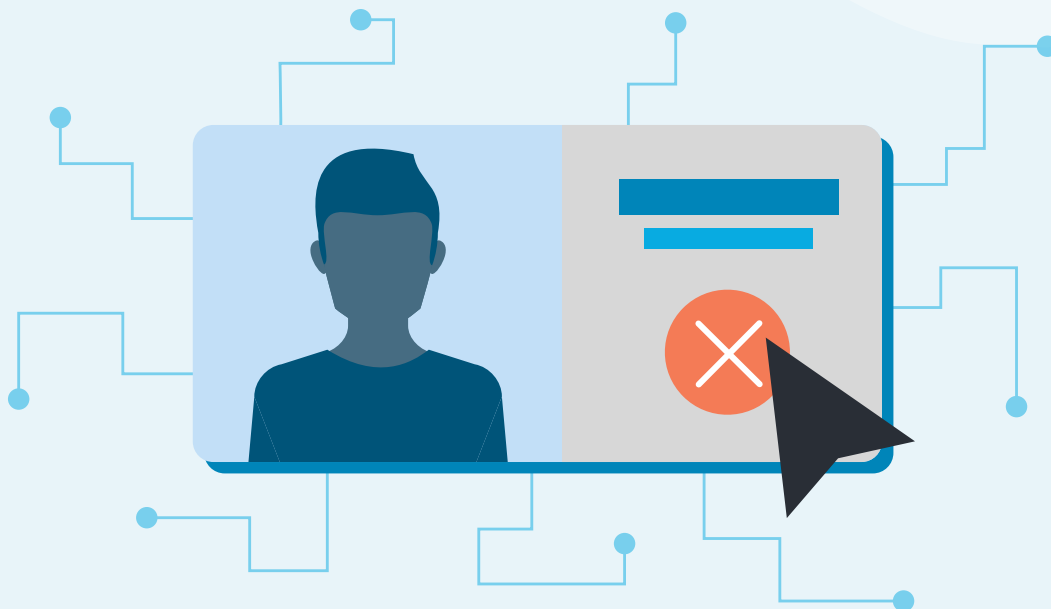# How to Safely Deactivate a User in Salesforce

CloudKettle

Deactivating a User in Salesforce may sound simple, but it can cause problems if not done correctly. Before deactivating a User in Salesforce, we recommend reviewing this cheat sheet.

Before we dive in, this cheat sheet uses the term **Dedicated Integration User**. If you're unfamiliar with the concept of a Dedicated Integration User, check out this blog post. (bit.ly/sfintegrationuser)
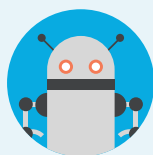
A Dedicated Integration User is a User License not used by a specific employee but for integrations to keep your Salesforce Org secure, auditable, and easier to manage.

Note:

The steps taken when deactivating a User change based on the customization that exists within your Salesforce Org. While this document covers the bulk of considerations, there may be other considerations not listed.
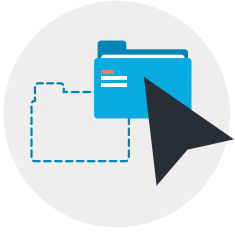
1. Records

Improperly deactivated Users can still own Records within Salesforce. This is a problem because any Leads, Opportunities, or Accounts attached to that User will be sitting in void (not assigned to any active User).

As an example, if a salesperson leaves a company and their User License isn't correctly deactivated, all alerts set up to notify the Account owner of accounts up for renewal won't deliver to any active User. To avoid this and other issues, follow the steps below.

How to Move Records from a Deactivated User to Another User

1. Run a report on each Object in Salesforce that is important for your business, to create a list of all Records the User being deactivated owns. This is done by filtering a list of individual Object reports to only show Records owned by the User that is being deactivated. We recommend filtering by *OwnerId* and *UserId* instead of filtering by the User's name. If you filter by the User's name, the results are more likely to include other User Records.

   For Sales Cloud Users, we recommend reviewing the core objects (Accounts, Leads, Opportunities, Cases and Contacts) and any other custom objects that are important to your business.

2. After you create these reports you can easily reuse them when deactivating Users in the future by updating the *OwnerId*.

3. Once you have a full report of the records you want to update, export them as CSVs.

4. In the CSVs, change the current *OwnerId* to the *OwnerId* of the user you'd like to transfer the records to.

5. Use a data import tool you're familiar with (Data Import Wizard, Data Loader, DataLoader.io, etc.) to update these Records in Salesforce with their new owner.

> ### Note:
> If you are not sure which User the records should be reassigned to, move them over to either the User's manager or a Dedicated Integration User (bit.ly/sfintegrationuser). Or for Leads, Cases, and Custom Objects you can assign them to a Queue. This creates a central place to store unparented records so they don't get lost when deactivating more Users in the future.

2. Dashboards

Dashboards have to run from the point of view of a User. When a User is deactivated, if they were the running User of a Dashboard or report, it will cause the components on that Dashboard to break. Check if there are any Dashboards in Salesforce created by or run as the User being deactivated.

How to Get a List of Dashboards Running as the User being Deactivated

1. Go to Developer Console.

2. Go to Query Editor at the bottom and type in the following ***Select title, ID, RunningUserId from dashboard where RunningUserId = UserId***
   (Replace the text in orange with the *UserId* of the User you want to deactivate.)

3. If this is a small list, you can tackle Dashboards individually by going to each one and changing the Running User.

4. If this is a larger list, an easier approach may be to use the same query in Data Loader. Extract the dashboards, change all the *RunningUserIds* in a CSV file, then use this new file to update the dashboards.
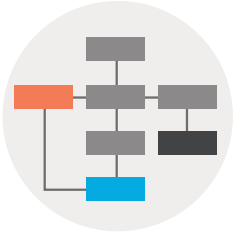
3. Removing Workflows

Salesforce won't allow a user to be deactivated if there are any active workflows attached to the User. Before a User is deactivated, validate they aren't being used in any Workflows.

Salesforce also won't let you deactivate the User, if that User is the default Lead, default Case, or default Workflow User. Here's how to deal with Web-to-Lead, Web-to-Case, and default Workflows when deactivating a User.

Web-to-Lead

1. Go to Setup.

2. Type in **Web-to-Lead** and click **edit**.

3. Change the default **Lead Creator** to another User (ideally your Dedicated Integration User).
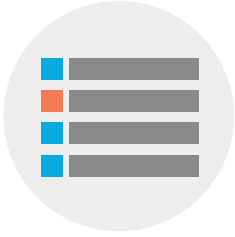
Web-to-Case

1. Go to Setup.

2. Type in **Support Settings** and click **edit**.

3. Change the default **Case Owner** to another User (ideally your Dedicated Integration User).

Default Workflow

1. Go to Setup.

2. Type in **Process Automation Settings**.

3. Where it reads **Default Workflow User**, assign this to another User (ideally your Integration User).

4. Queues

Check if the User you're deactivating belongs to any Queues and/or Public Groups. Also check if this User is the sole member of a Queue.

How to Check if a User is the Sole Member of a Queue

1. Go to Developer Console and press **cmd-e** (**ctrl-e** for Windows). This should bring up the Execute Anonymous window.

2. Once open, paste in the following code: ***DeactivatedUserId = UserId*** (Replace the text in orange with the *UserId* of the User you want to deactivate.)

```
Id DeactivatedUserId = '00563000002giHK';

Map<Id,Group> groupsUserIsIn = new Map<Id,Group>([Select Id,Name,Type
from Group where id in (Select groupid from groupMember where
userOrGroupId =:DeactivatedUserId) ]);


AggregateResult[] groupMemberCount = [Select count(id) memcount, Groupid
gid from groupMember where groupid in: groupsUserIsIn.keyset() group by
groupid];

Set<Id> groupsWithOnlyOneUser = new Set<Id>();


For(AggregateResult result : groupMemberCount){
     Integer count = (Integer)result.get('memcount');
   if(count <= 1){
     // user is only group or already deactivate and was only member
             groupsWithOnlyOneUser.add((Id)result.get('gid'));
   }
}
String subject = 'Groups in danger of being empty after deactivating User';

String body = 'These are the groups that will be empty after deactivating user:'
+ DeactivatedUserId + '\n';

for(Id groupid: groupsWithOnlyOneUser){
```

An email will be sent to the running User's email (ensure deliverability is on) that includes a breakdown of all items mentioned above that you may need to reassign.

5. Scheduled Jobs

Next, take stock of what Scheduled Jobs are attached to the User being deactivated.

1. Go to Setup and type in **Scheduled Jobs**.

2. Create a new List View that filters the **Submitted By** by the User's name you want to deactivate.

This will produce a view of what Scheduled Jobs need to be updated. There are a number of types of Scheduled Jobs that could exist within your instance. We'll cover what to do about three common ones:

Data Export

If the User being deactivated is in charge of weekly exports, this task will have to be reassigned internally to another employee(s).

1. Delete the current data export **Schedule Jobs**.

2. Create them again with a new User. Here's a simple resource (bit.ly/schedulejobs) for setting up a new Scheduled Job.

Scheduled Report

1. Write down the name of the Dashboard (as seen in Job Name Column).

2. Open a new tab, then go to this Dashboard and view its scheduled settings. It *looks* like you can just change the User who sends the Scheduled Report, but saving a new User doesn't actually change the job.

3. Write down the scheduled settings and then delete the job from the Scheduled Jobs tab.

4. Reschedule the Report as another User, ideally the Dedicated Integration User, to the same frequency.

Scheduled Apex

1. Open the job by clicking **Manage** next to the job.

2. Write down these frequency settings, including the Job Name and Apex Class.

3. Delete this job and go to Setup.

4. Type in **Apex Classes**.

5. Click the **Schedule Apex** button at the top of the ListView. This will allow you to fill in the information that you had previously recorded.

Note:
Whichever User you are logged in as, will be the running User of this newly scheduled job. As always, this is a great use case to leverage and be logged in as your Dedicated Integration User.

6. Metadata

Now for the tough one; hard coded references to the User or User's email in Salesforce Metadata. A Salesforce Developer will have to examine the Org's Metadata and search for the User's ID and email address throughout all files. If you don't have a Salesforce Developer on staff, at least ensure that the User's email isn't hard coded into any email alerts. Otherwise emails will continue to go to this address.

How to Search for Hard Coded References

1. Go to Setup and type in *Email Alerts*.

2. Create a new List View that will still show all alerts; include the from email address in the List View as well.

3. If you see the User's email address in this column, update it so that it sends to a different User.

4. The main item we are searching for here is hardcoded email addresses. Unfortunately, this process is extremely manual. Sort through each email alert in your instance and check the *Recipients Field* in every alert.

   For some, this may not be a daunting task, as you may only have 10-50 alerts in your instance. However, for others this may take awhile.

Wrap Up

Is this full process necessary every time a User is deactivated in Salesforce? The short answer is no, especially steps five and six. If the same person(s) in your organization is always responsible for deactivating Users, after they complete those steps for one User, they'll have a pretty good idea of what Users in the instance are hard coded or key for running Scheduled Jobs. However, we still recommend using these six steps as a guide each time you deactivate a User in Salesforce.

If you enjoyed this read, you may like our Security Guide:

How to Protect and Monitor Salesforce

bit.ly/sfsecurityguide

Have questions about Salesforce strategic planning or how to monitor your Salesforce instance 24 hours a day?

Reach out today. We love helping companies with Salesforce strategic planning.