

Common Flow Problems

A binder of common flow problems and how to work around them

- [Avoiding the 5 recipient limit on email action](#)

Avoiding the 5 recipient limit on email action

A Warning in Advance

Skipping limits is dumb, you probably shouldn't do it, but history is filled with people doing dumb stuff and getting away with it. What's the worst that could happen here? Maybe an email doesn't get sent, maybe your release manager vomits at the sight of your commit?

Problem Statement:

I want to send an email for notification from a flow, but I want to [send it to more than 5 people](#) from the standard email component.


In this example, I'm going to set up a scheduled flow to check for expiring package licenses and notify the admin team, via the [apex exception queue](#). (FYI - the apex exception email recipients list can be set as the default notification for flow debug emails, which is quite useful in smaller orgs)

Gather the email addresses into a collection:

Our first action is to collect the email addresses from the Apex Email Notification table:

Edit Get Records

Find Salesforce records and store their field values in flow variables.


Get ApexExceptionEmails (Get_ApexExceptionEmails) 

Get Records of This Object

• Object

Apex Email Notification

Filter Apex Email Notification Records


 With no conditions, the flow retrieves **all** Apex Email Notification records.

Condition Requirements

None—Get All Apex Email Notificati... ▼

Sort Apex Email Notification Records

Sort Order

Not Sorted ▼  If you store only the first record, filter by a unique field, such as ID.

How Many Records to Store

☐ Only the first record

☒ All records

How to Store Record Data

☒ Automatically store all fields


☐ Choose fields and let Salesforce do the rest

☐ Choose fields and assign variables (advanced)

Cancel Done

And add the records to a text collection variable:

EmailRecipientsCollection Edit Variable

EmailRecipientsCollection 

• Data Type ⓘ

Text ▼ ☒ Allow multiple values (collection) ⓘ

Availability Outside the Flow

☐ Available for input

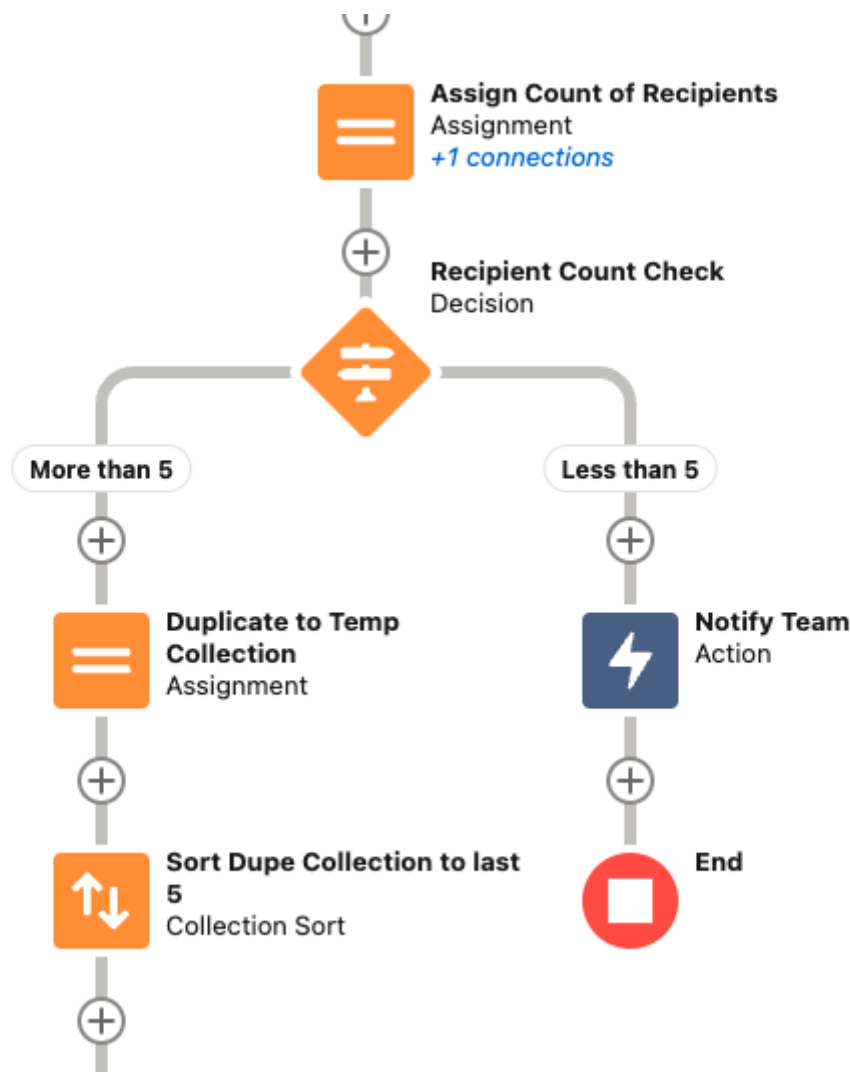
☐ Available for output

Cancel Done

In this case, there are two possible locations for the email, as the notification emails can either hold an email address, or a lookup to a user, both can be added to the string collection.

Iterating our collection:

Our first action, once we have built our recipient collection, is to assign the count of records to a number variable. Then we can have a decision element to determine if we need to select only 5 recipients from the collection, or have fewer than five and can simply send the email.



you can see the decision above, with the looping process handled by the more than 5 branch.

How to iterate:

The process for iterating here is simple step-wise, but not overly obvious when starting from a blank canvas. If we were doing this in the natural way (how you might describe it without platform limitations, beyond the 5 recipient limit)

1. Take the first 5 out of the recipient collection
2. Send them the email
3. Back to the collection count element and update

Step 1, unfortunately, is not available in salesforce, we instead will have to take a few more steps:

1. Duplicate your collection into a temporary collection

2. Sort your collection and limit to 5 using the [Collection Sort](#) element*
3. Your temp collection now only has 5 recipients in it, so send them the email
4. Remove those 5 recipients from your *Original Collection*
5. Re-count the original collection and proceed to the decision

*Unlike the filter element, collection sort DOES modify the supplied collection, and DOES NOT accept resources as the limit variable. The modification is why we need to create a temporary collection for the 5 recipients in the loop.

Step 1:

Duplicate your collection into a temporary collection:

Edit Assignment

Duplicate to Temp Collection (Duplicate_to_Temp_Collection)

Set Variable Values

Each variable is modified by the operator and value combination.

Variable	Operator	Value	
Last5Recipients	Equals	EmailRecipientsCollection	

Add Assignment

Cancel

Done

Last5Recipients is an identically configured variable text collection that we will use to hold our current batch. Using the equals means that any current values in the collection should be overwritten each time.

Step 2:

Sort your collection and limit to 5 using the [Collection Sort](#) element:

Edit Collection Sort

Sort Dupe Collection to last 5 (Sort_Dupe_Collection_to_last_5)

Sort Options

• Collection Variable

Aa EmailRecipientsCollection

×

Sort Order

Descending

▼

☐ Put empty string and null values first

Control the Collection Size

• How Many Items to Keep After Sorting

☐ Keep all items

☒ Set the maximum number of items

After sorting, the collection keeps the first items, up to the specified maximum, and removes the rest.

5

Cancel

Done

This selects us the last 5 recipients in the collection, these will be the target for the next email action

Step 3:

Your temp collection now only has 5 recipients in it, so send them the email

Step 4:

Remove those 5 recipients from your *Original Collection*:

Edit Assignment

Remove Temp 5 from collection (Remove_Temp_5_from_collection)

Set Variable Values

Each variable is modified by the operator and value combination.

Variable

Aa EmailRecipientsCollection

×

Operator

Remove All

▼

Value

Aa Last5Recipients

×

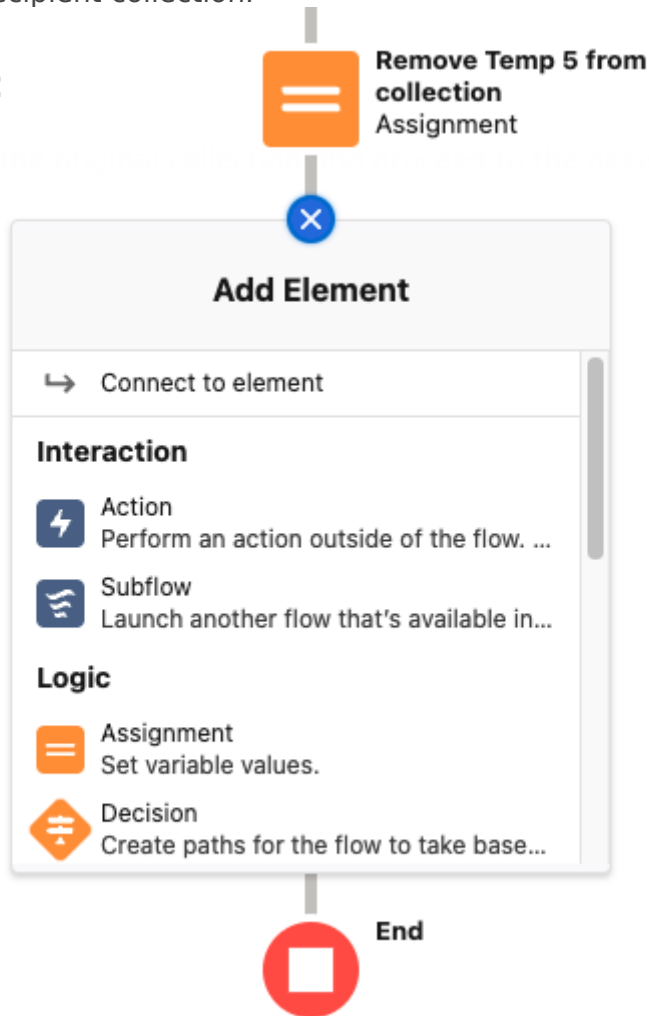
+ Add Assignment

Cancel

Done

The [remove all](#) operator for the assignment element removes all occurrences of the *Value* from the *Variable* in our case, we are then removing the 5 recipients that we have just emailed, from the original recipient collection.

Step 5:



The connect to element option

allows you to connect to other parts of your flow, simply select your *Assign Count of Recipients* element ([see here](#)) and you've created your loop.

One final warning: I have NO IDEA how far you could push this, I certainly wouldn't recommend building a huge notification list into this flow, there are better tools for mass emailing, and you're almost certain to have issues if you push this really far.

If you were feeling particularly cunning, you could package this into an autolaunched flow, receiving a collection of recipients, a subject, and body, then call it asynchronously from across your org, saving you from having to rewrite it every time.