

## Snapshot Best Practices: Org Splits



Snapshot has powerful metadata deployment and data migration tools that enable administrators to split complex Salesforce orgs. The need to perform an org split is usually a special situation and not something that is done on a regular basis. These projects are often mission critical and may involve contractual commitments or other external deadlines. Because of this, Metazoa also offers professional services to perform this work on behalf of customers. Either way, an org split is a complex undertaking that requires careful planning, business analysis, and project management to ensure successful completion.

This whitepaper discusses some of the best practices when using the Snapshot toolset to perform an org split. We will cover project management, synchronizing settings, removing technical debt, metadata deployments, data migration, and final steps. First off, let's define some of the most common org transformations, and when they are likely to be encountered:

**Org Clones** • The source org is used to create a destination org that has the same structure and customizations. This situation can happen when a company wants to copy an existing org for a new division, or when an org is rebooted because of technical debt or other issues.

**Org Splits** • An org split is like an org clone, except that a subset of users and data from the source org are moved to the destination. This situation can happen when a company sells a division to another company. The Salesforce org is used to transfer the assets.

**Org Merges** • Two or more source Salesforce orgs are merged into a single destination org. This procedure requires careful planning and business analysis to decide how the new destination org should function. This situation can happen when a company wants to consolidate two divisions.

For the purposes of this whitepaper, we are going to focus on org splits. An org clone is essentially the same thing as an org split except that all or none of the source users are migrated to the destination. An org merge is quite different than an org split or an org clone. The topic of org merges is covered in a separate whitepaper available on the Metazoa website.

## Project Management

A high degree of coordination is required between the team performing the org split and the existing administrative staff. The simplest scenario is when the migration team has unfettered access to the source org and is able to create a new destination org that they have complete control of. A more likely scenario is that the source org will remain in active use during the project. This can complicate the situation, especially if metadata and data assets in the source org are being modified during the migration process. The best practice here is to agree on outage windows where the migration team can have exclusive access to the source org. Even still, this usually requires a two-phase migration: phase one moves all the metadata and data, and phase two checks for new differences before final delivery.

Any org split will need an administrative champion who can answer questions about the corporate requirements for a successful transition. For example, the migration team will need to know what group of users should be moved to the destination org. What data records will this group of users require? Should this information also be deleted from the source? How should inactive users be handled? The administrative champion should also coordinate outage windows, org validation, user acceptance testing, user training, and final cutover to the new destination org.

## Inevitable Differences

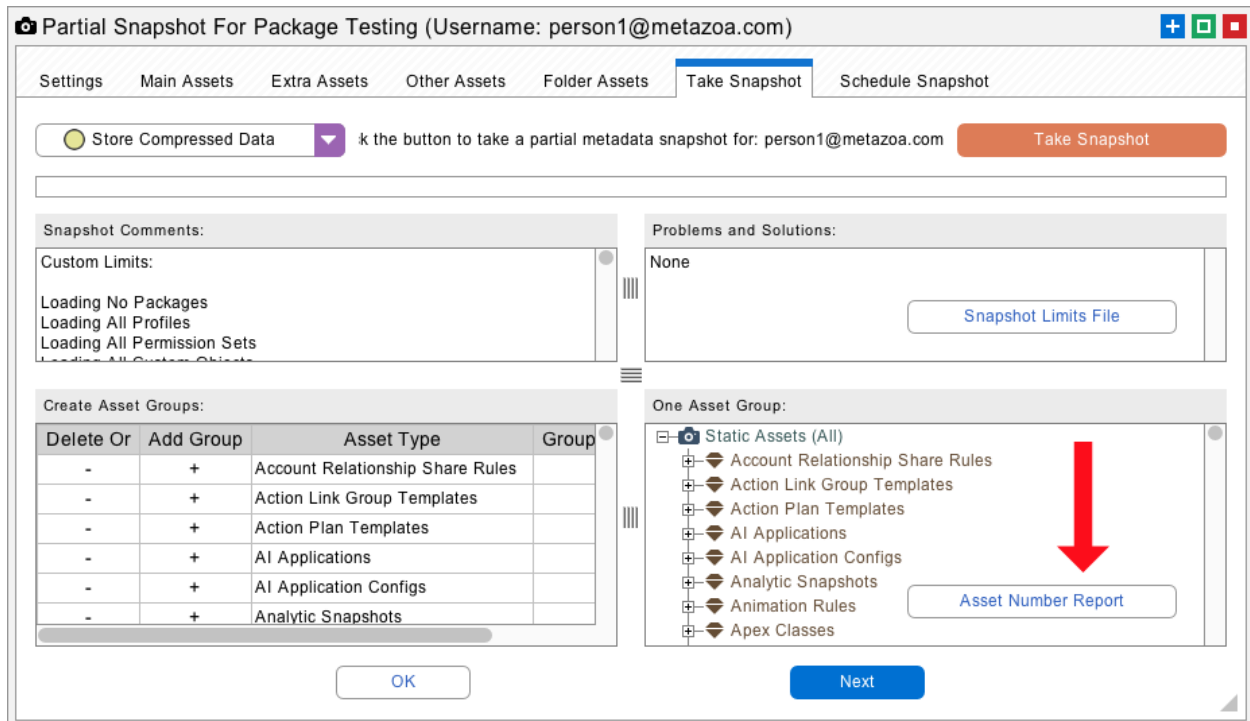
At some point, the project will be finished, and the new destination org will be ready for cutover or phased transition. Unfortunately, there usually some inevitable differences between the original source and the new destination org. For example, older Salesforce orgs can start up in Classic mode, but all new orgs will automatically start up with the Lightning user experience. SControls can no longer be created, and they will need to be replaced with Visualforce pages. Workflows are being deprecated, and they should probably be replaced with Flows. Portals can no longer be migrated, so they will need to be replaced with Communities.

Because of these differences, user training and adoption are an important part of any org split. The creation of a new destination org is an opportunity to increase user adoption with additional training and to modernize the Salesforce implementation.

## Sanity Check

Let's look at the big picture. First, the migration team will need to take snapshots of the metadata from the source org and then deploy these customizations into the destination org. Then they will need to create datasets of various records in the source org and migrate this information to the destination as well. Before work begins, the migration team needs to conduct a sanity check of the required team size, processing power and hard disk storage required to get the job done.

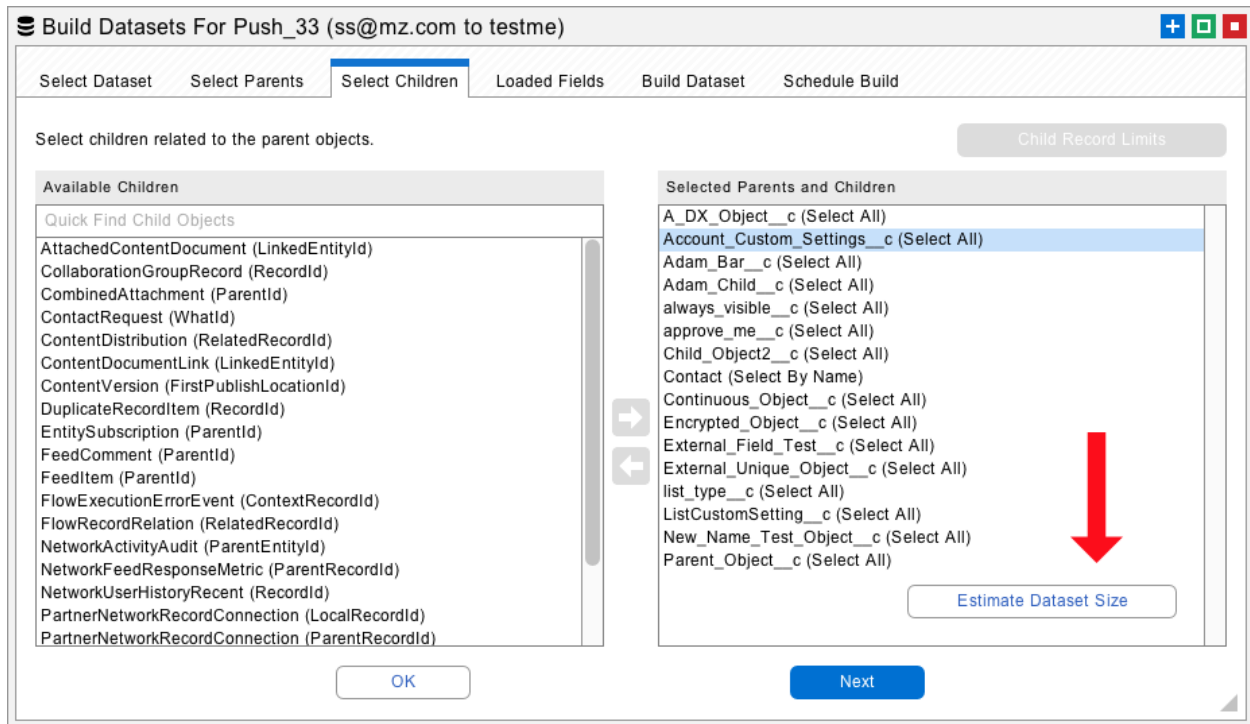
The number of metadata assets in the source org can be estimated with the Partial Snapshot interface. Enter your org credentials and navigate to the Take Snapshot tab, and you will see the Asset Number Report button at below right. This report will count all the metadata assets in the source org, including reports, dashboards, and packages.



There is a limit of 10,000 metadata assets in a single download. If the source org falls inside this limit, then you can use the Full Snapshot interface to capture all the metadata. If the source org has a gigantic number of assets, then the Partial Snapshot interface can be used to take multiple snapshots and stitch them together into a complete picture. You can also grab individual pieces of the source metadata and deploy each one separately. The controls in the lower left of the Partial Snapshot interface define the groups of metadata to download, see the help page on this interface for more information.

Next, you need to get an estimate of the number of data records that must be transferred. The Storage Usage in Salesforce Setup can give you some of this information, but if you are not migrating all objects and fields, then what you really need are statistics on the records actually being moved. An easy way to get this information is from the Build Datasets interface.

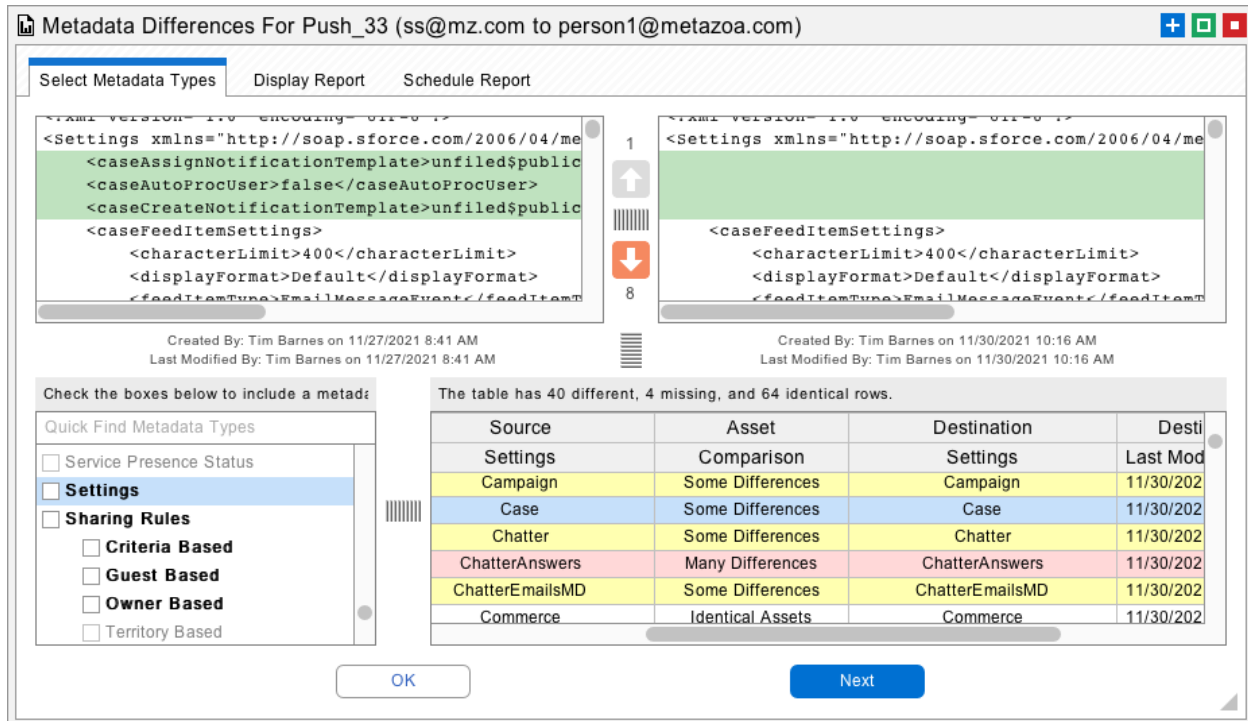
Select all the parent and child objects that must be migrated and navigate to the Select Children tab. The Estimate Dataset Size button at the bottom right will create a report that gives you an accurate count of the number of records that must be moved and an estimate of the total size of the dataset. You probably don't want to build a single dataset with all the records that need to be moved. You can move the data in sections. But this report will give you a bird's eye view of the size of the job.



Now it's time for the sanity check. Orgs with less than 10,000 metadata assets and 1 million records can usually be migrated by a small team. They will need modern laptop computers, and a fast Internet connection. Since millions of files are involved, a fast solid-state hard drive with many GB of storage is helpful. But there are also situations where org migrations have millions of metadata assets and hundreds of millions of data records. In this situation, we recommend using Windows or Macintosh cloud servers. They can be outfitted with huge amounts of processing power, hard disk storage, and Internet connectivity. They can be controlled remotely from a laptop functioning as a terminal. You can run multiple instances of the Metazoa Player on each cloud server. Obviously, the team members will have to coordinate their efforts and migrate the metadata and data records. We discuss some of the best practices to organize massive migrations into logical sections, below.

## Synchronize Environments

Before trying to deploy any metadata or migrate any data, you should do everything possible to make sure that the source and destination org environments are as similar as possible. If the environment is the same on both orgs, many metadata deployments and data migrations that would otherwise be impossible will work just fine. The first order of business is to make sure that the destination org has been outfitted with the same Salesforce features and products as the source org. Your account representative can help provision the destination org.



Metadata Differences For Push\_33 (ss@mz.com to person1@metazoa.com)

Select Metadata Types | Display Report | Schedule Report

XML Snippets:

```
<Settings xmlns="http://soap.sforce.com/2006/04/me
<caseAssignNotificationTemplate>unfiled$public
<caseAutoProcUser>false</caseAutoProcUser>
<caseCreateNotificationTemplate>unfiled$public
<caseFeedItemSettings>
  <characterLimit>400</characterLimit>
  <displayFormat>Default</displayFormat>
  <feedItemType>EmailMessageEvent</feedItemT
```

Created By: Tim Barnes on 11/27/2021 8:41 AM  
Last Modified By: Tim Barnes on 11/27/2021 8:41 AM

Created By: Tim Barnes on 11/30/2021 10:16 AM  
Last Modified By: Tim Barnes on 11/30/2021 10:16 AM

Check the boxes below to include a metadata type:

Quick Find Metadata Types

- Service Presence Status
- Settings**
- Sharing Rules
  - Criteria Based
  - Guest Based
  - Owner Based
  - Territory Based

The table has 40 different, 4 missing, and 64 identical rows.

Source	Asset	Destination	Destination Last Modified
Settings	Comparison	Settings	Last Modified
Campaign	Some Differences	Campaign	11/30/2021
Case	Some Differences	Case	11/30/2021
Chatter	Some Differences	Chatter	11/30/2021
ChatterAnswers	Many Differences	ChatterAnswers	11/30/2021
ChatterEmailsMD	Some Differences	ChatterEmailsMD	11/30/2021
Commerce	Identical Assets	Commerce	11/30/2021

Buttons: OK, Next

Next, take a snapshot of the Settings metadata type on the source and destination orgs. The Settings control many aspects of the org environment. For example, you can choose to automate Flows on deployment, or declare the org to be multi-currency. Then use the Metadata Differences Report to drill down into the differences. You can deploy the Settings metadata from the source org into the destination org. There may be deployment errors, which will point out other differences between the orgs. You might need to purchase some additional Salesforce product, or you might need to make manual changes with the Setup Menu. We estimate that 80% of the environmental differences between orgs can be rectified by deploying the metadata Settings to the destination org.

All of the **custom** Objects, Tabs, Applications, and other assets that were created on the source org can be transferred to the destination org. But the **standard** Objects, Tabs, Applications, and User Permissions that are on the destination org cannot be changed. Look at any differences between these asset types. The Compare Profiles interface is an easy place to do this. This information can be used to discover any remaining environmental differences. Also be aware that Snapshots metadata deployment tools can Remove Bad References to **standard** objects during the deployment process if necessary.

After that, you need to make sure that the managed packages are the same in each org. If the source org does not have the latest version of a managed package installed, you will have to track down the historical package version. Your friendly neighborhood AppExchange partner should be able to help with that. In other cases, you can install the most recent version of the managed package. This upgrades the destination org to the latest version but be aware that this will also probably lead to some environmental differences. The deployment of unmanaged packages and package customizations can wait for later in the migration process.

## Technical Debt

It doesn't make any sense to transfer a bunch of technical debt from the source org into the destination org. A complete technical debt analysis is a great first step to discover the assets that should and should not be migrated. Snapshot has amazing reports that help with technical debt analysis, including the ability to identify and merge similar profiles, remove connections to inactive users, optimize license expense, delete forgotten metadata, and retire unused fields and picklists. Depending on the situation, you might want to remove technical debt from the source org or use this knowledge to control what is migrated to the destination.

When it comes to data migration, make sure that Contacts, Leads, and Accounts in the source org have been deduped if possible. This makes polymorphic references in Tasks and Events more likely to match by name. Converted Leads often do not need to be migrated, because that information has already been captured by Opportunities and Contacts. Attachments are being deprecated in favor of ContentDocuments, so you might want to convert them before migration to the destination org.

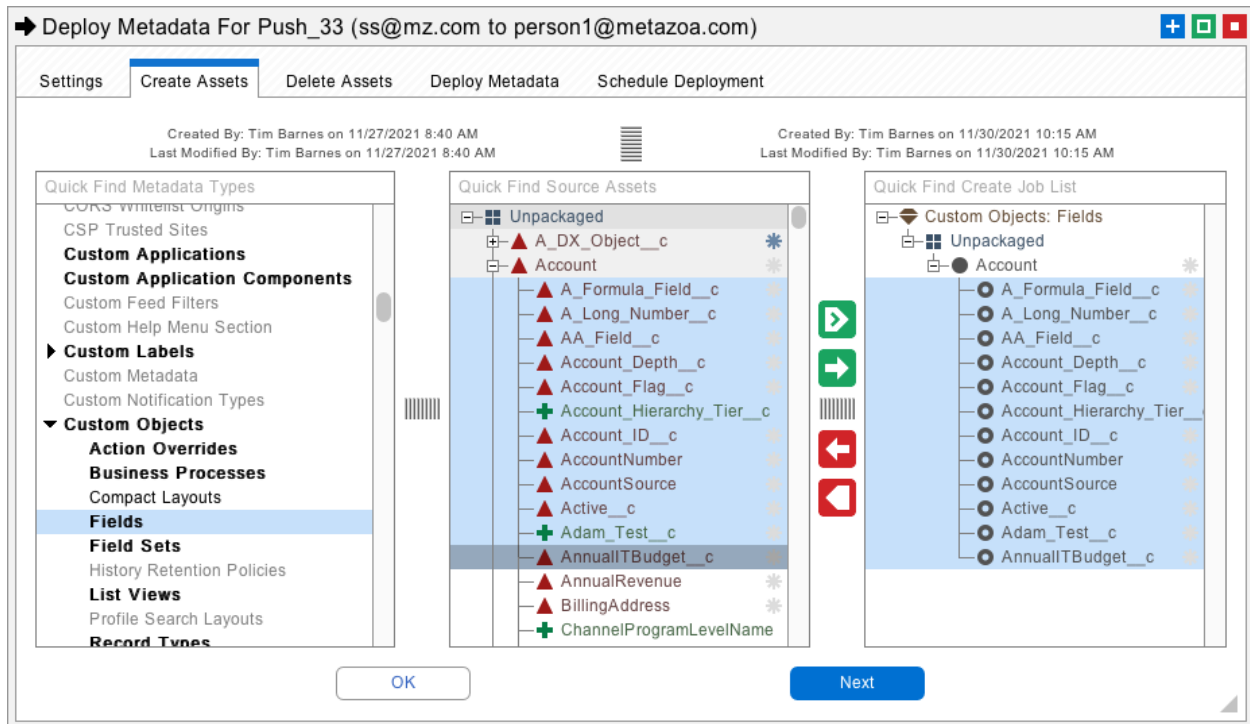
Another huge issue is how inactive users are handled. Snapshot has powerful tools to delete all the connections to inactive users in the source org. Once this is done, only the active users should be migrated to the destination. Sometimes administrators don't want to clean up the inactive users if the source org is in active use. That might cause unnecessary disruptions. The best practice is to clean up the inactive users in a sandbox org and deploy the metadata from there. Both the metadata deployment tools and data migration tools in Snapshot can transform usernames as necessary during migration.

## Metadata Deployment

There are over 250 different metadata types in Salesforce that define most of the customizations in an org. There are also 1200 different kinds of dependencies between these metadata types. Each dependency is a reference that connects two assets. When an asset is migrated, there can be validation problems or missing dependencies. Snapshot has an excellent Impact Analysis Report for understanding dependencies, and our deployment engine has features that simplify the deployment process. But deploying all the assets from one org into another org can be difficult because of complex circular dependencies and environmental differences.

We recommend getting your destination org started off with assets that don't have many dependencies. First, you should migrate Groups, Roles, and Value Sets. They don't have any dependencies. Next, try moving only Custom Fields. If the Custom Object does not exist on the destination, then Snapshot will create the shell of the object for you. Since most of the dependencies are in the Custom Object itself (not in the Custom Fields) this deployment doesn't involve many dependencies. After Custom Fields are moved, data can be migrated.





The use of dummy Profiles is another clever trick to dodge dependencies. From the Setup Menu in the destination org, create a Minimum Access Profile. Duplicate this profile many times, and give each copy the same name as the corresponding profile in the source org. This clears the way for users to be migrated as data, and Queues to be deployed as metadata. The dummy Profiles will be updated later with real permissions when all the other assets they depend on are available on the destination. This is a best practice for getting your destination org off the ground.

Now you want to build up the other parts of the Custom Object one at a time. List Views (which depend on Queues) can now be migrated, followed by Validation Rules and Page Layouts. Apex components and Lightning Record Pages may complicate the process with additional dependencies. Take Snapshots of the source and destination org, then look at the metadata comparison, and keep moving whatever metadata assets that you can until the orgs are the same. The last metadata assets to deploy are probably Profiles and Permission Sets because they depend on many other asset types.

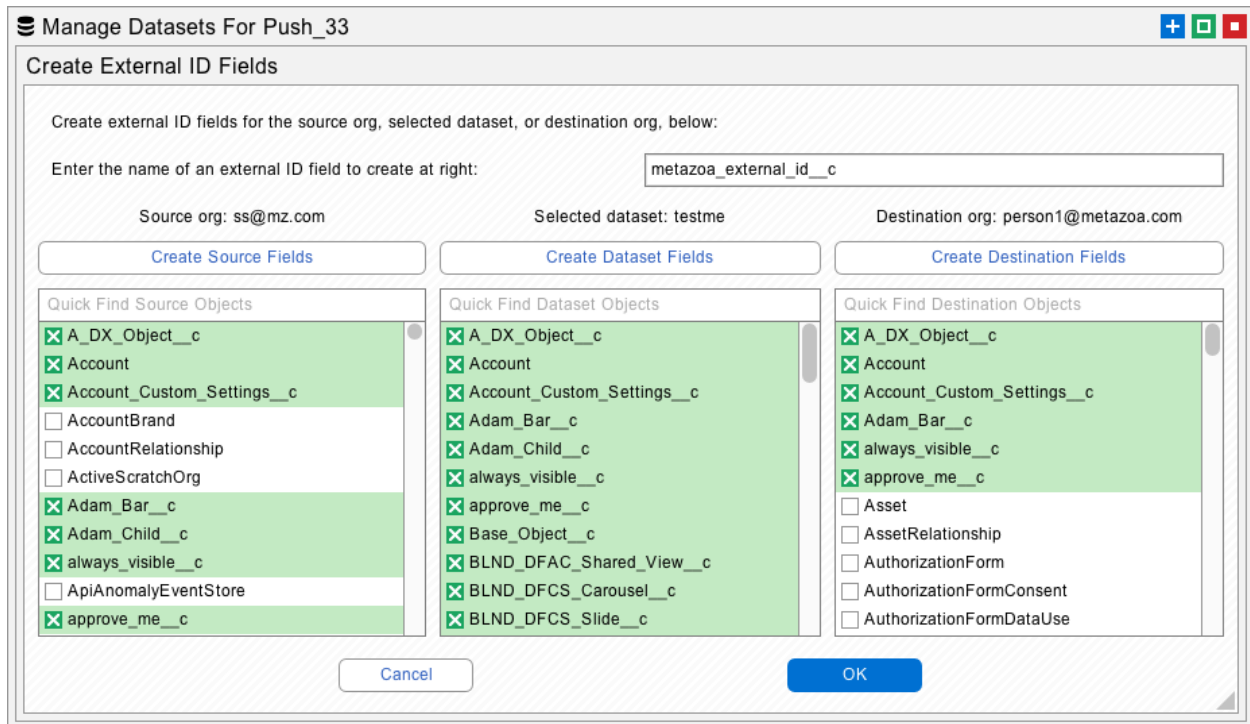
The difference between metadata and data is rather blurry. Many important junction objects are actually saved as data, not metadata. Examples of this include Group Members, Campaign Members, Permission Set Assignments, and Account, Opportunity, and Case Team Members. Once your metadata has been deployed, it's time to focus on data migration.

## Data Migration

The data migration process has some similarities to metadata deployment. There are likely to be multiple migrations, and various errors along the way that need to be addressed. Because of this, the best practice is to use external IDs to match destination data records with source records. Here is how this works. The external ID is a custom field on the source object that holds a unique external value. Often this value is just the source record ID itself. When these records are migrated to the destination, each destination record is permanently tagged with this value. Now when you match the records by external ID, the same record will be matched every time, even after repeated migrations.

Snapshot has introduced a killer feature that vastly simplifies the process of creating external IDs. The Manage Dataset interface has a new option to create external IDs on the source org, the dataset itself, or the destination org. When you create an external ID on the source org, Snapshot creates a custom **formula** field that sets the field value equal to the object ID. When you create an external ID on the destination org, then Snapshot creates a custom **text** field that receives the source ID value. You can give the new custom field any name that you like. Field Level Security is automatically set for the System Administrator Profile and the field is hidden from everyone else.

But sometimes administrators don't want to create a bunch of external IDs in the source org. That might be disruptive if the org is in active use. In this case, Snapshot offers the capability to create an external ID on the dataset itself, as if the external ID was actually from the source org. This works exactly the same way when the destination records are tagged or matched with the source ID. The beauty of this is that you can fully exploit the power of migrating data with external IDs without actually having any external ID fields in the source org.



Snapshot has very powerful tools to select source records for migration. You can select parent objects with a filter interface, match them by name, or manually enter a SOQL filter. Then you can select any number of child objects that are related to the parent. This user interface makes it easy to build up datasets that move the desired records.

However, when you get into the realm of moving millions of records, there are other strategies that might work better. For example, if you tag Accounts, Contacts, and Campaigns with an external ID then you could move them one at a time. First all Accounts, then all Contacts, then all Campaigns. Dividing the job up like this reduces the total number of objects and can be conducted on multiple machines.

Next, you could move all CampaignMembers. Each CampaignMember must have the correct reference to an Account, Contact, and Campaign before it can be created. The use of external IDs guarantees that each CampaignMember will be connected correctly to the parent object. The only trick here is moving the objects in the right order with the junction objects like CampaignMember in last place.

## Final Steps

As mentioned previously, some of the last metadata assets to deploy will be Profiles and Permission Sets, because they reference many other metadata types. Move the Sharing Rules as well. We save them for the end because they will trigger a sharing recalculation that slows everything down. Now you should also be able to migrate the entire Custom Object with all dependencies. This will ensure that Org Wide Defaults and other options are set on the destination org. If the source org was in active use during the project, then a final validation of data and metadata values needs to occur.

But the real action starts when the new destination org is tested for validation. The testing might be conducted by multiple departments or different administrators who are familiar with the source org. If there are substantial changes in the destination org, say because there was a switch to Lightning, then there will be user acceptance testing as well. Ultimately users must be trained to adopt the new org, just like any other Salesforce onboarding experience.

## Conclusion

The Snapshot toolset is purpose built to solve difficult problems like org splits. The elimination of technical debt is a bonus along the way. The other option is to hire a large team of developers or consultants to do the work by hand. This approach is guaranteed to be slow, expensive, and error prone. If you need help with a mission critical org transformation, please reach out and let us know how to provide support. We offer consulting on a project basis, or you can have our Professional Services Team handle everything for you.

Bill Appleton  
CTO Metazoa  
bill@metazoa.com  
@bill\_appleton