Digital Preservation the Hard Way

“Just back up the assetstore & DB,” they said.

“I may have deleted the Electronic Theses community. Is there any way to un-delete it?”

Seven missing communities, 11 missing collections, 315 missing items (containing 878 bitstreams). Only 1.4 GB of data. Born-digital data.
This is a story of survival. Of that data, metadata, and everyone responsible for it.

“Don’t panic! Your bitstreams are probably fine!”

DSpace does not actually delete bitstreams right away; it merely marks the bitstream metadata for deletion. A cleanup script needs to be run monthly to actually free up disk space, and ensure bitstreams marked for deletion are removed. Which means, as long as you can disable that scheduled script (look in *crontab* if you’re hosting DSpace on Linux/UNIX, *Task Scheduler* if you’re running Windows), your data will be fine. This part is easy, but don't forget it: *disable this script*:

```
# MONTHLY TASKS
0 1 1 * * $DSPACE/bin/dspace cleanup > /dev/null
```

“Now, all we have to do is restore the missing metadata.”

The good news is, someone has done the hard work of figuring out the SQL queries[1]. But before we get too far, you need to secure your database backup, and arrange for a snapshot of your database to be restored to a new database. If you have a DBA, *great*, ask them to take care of that for you. If you don't, you are practically a DBA already, just restore that data somewhere you can query it. We are lucky, we have DBAs. For convenience, I will assume you also have a DBA.

1. Find the ID of the deleted object (in our case it was a community)
2. Obtain a copy of the DSpace Data Schema diagram[2]
3. Using your prefered SQL client, craft queries[3] for all child objects of your object. For our example, this would be
   a. epersongroup for approved submitters for each collection
   b. sub-communities
   c. community2community relationships
   d. collections
   e. community2collection relationships
   f. items
   g. bundles
   h. bitstreams
   i. handles for items
   j. handles for collections
   k. handles for communities
   l. metadatavalues
   m. item2bundle relationships
   n. bunde2bitstream relationships
   o. collection2item relationships
   p. resourcepolicies

Don't panic! The SQL for all of this is available online.[3]

4. for each of the SQL queries above, export the results as SQL INSERT statements.
5. for the handle table, it's possible some handle rows remain, to be careful, you should also write UPDATE queries[3] in addition to INSERTs
6. concatenate all INSERTs and UPDATES in the following order: epersongroup, community, community2community, collection, community2collection, handle_community, update_handle_community, handle_collection, update_handle_collection, item, metadatavalue, item2bundle, bundle, bundle2bitstream, collection2item resourcepolicy, handle_item, update_handle_item (many thanks to Sam Ottenhoff from Longsight for determining this order and making it available in his *move-structure.php* script[1].
7. schedule a maintenance window, coordinate with your DBA (even if it's you) to be available to provide back up
8. during the maintenance window
   a. shut down the DSpace servlet container
   b. have your DBA take a snapshot of your live/production database, in case of further disaster
   c. have your DBA disable foreign key constraints on your live/production database
   d. have your DBA execute all the insert and update SQL statements you produced in steps 3-6 above
   e. using your SQL client, review the changes you have made
   f. if it all looks right, re-index your repository[4] (may require you to bring your servlet container back up)
   g. if your servlet container is still down, bring it back up
   h. verify your work

“Disaster Recovery is not Digital Preservation.”

If you are a developer reading this, thinking about ways your institution could improve its backup stategy, I've got bad news for you. *A backup strategy is not digital preservation*.

If you are the only person at your institution thinking of ways to improve your backup strategy, your first job, before you even start to change your backup strategy, is to find other people to work with you on digital preservation. Repository development is a difficult enough job, you do not need to also assume responsibility for ensuring the data you are storing is valid, usable and backed up.

Digital preservation is a full time job. If you are not giving it your full attention, if you are just backing up your assetstore and database, you have already accepted the responsibility of digital preservation, you are just not doing the job.

The right tools for the job

If your institution is OK with storing your assets in “the cloud”, check out...

DuraCloud[5]
A managed service from DuraSpace, providing the tools and the storage space to ensure your data and metadata are backed up, valid, and usable.

If your institution would prefer to retain control of where (geographically) your digital assets are stored, check out...

Replication Task Suite[6]
A DSpace Add-on providing a set of curation system tasks to assist in performing replication (backup/restore/audit) of DSpace contents to other locations, including local storage.

Links

1. https://github.com/ottenhoff/dspace-utils/blob/master/move-structure.php
2. https://wiki.duraspace.org/display/DSDOC5x/Storage+Layer
3. https://github.com/hardyoyo/or2015poster-preservation-the-hard-way/blob/master/⏎
   helpful_SQL.md
4. https://wiki.duraspace.org/display/DSDOC5x/Command+Line+Operations
5. http://duracloud.com
6. https://wiki.duraspace.org/display/DSPACE/ReplicationTaskSuite