SOLR Statistics Maintenance

DSpace Log Converter

The use of Solr for statistics in DSpace makes it possible to have a database of statistics. With this in mind, there is the issue of the older log files and how a site can use them. The following command process is able to convert the existing log files and then import them for Solr use. The user will need to perform this conversion only once.

The Log Converter program converts log files from dspace.log into an intermediate format that can be inserted into Solr.

Command used:	`[dspace]/bin/dspace stats-log-converter`
Java class:	org.dspace.statistics.util.ClassicDSpaceLogConverter
Arguments short and long forms):	Description
`-i` or `--in`	Input file
`-o` or `--out`	Output file
`-m` or `--multiple`	Adds a wildcard at the end of input and output, so it would mean if `-i dspace.log -m` was specified, dspace.log* would be converted. (i.e. all of the following: *dspace.log, dspace.log.1, dspace.log.2, dspace.log.3,* etc.)
`-n` or `--newformat`	If the log files have been created with DSpace 1.6 or newer
`-v` or `--verbose`	Display verbose output (helpful for debugging)
`-h` or `--help`	Help

The command loads the intermediate log files that have been created by the aforementioned script into Solr.

Command used:	`[dspace]/bin/dspace stats-log-importer`
Java class:	org.dspace.statistics.util.StatisticsImporter
Arguments (short and long forms):	Description
`-i` or `--in`	input file
`-m` or `--multiple`	Adds a wildcard at the end of the input, so it would mean dspace.log* would be imported
`-s` or `--skipdns`	To skip the reverse DNS lookups that work out where a user is from. (The DNS lookup finds the information about the host from its IP address, such as geographical location, etc. This can be slow, and wouldn't work on a server not connected to the internet.)
`-v` or `--verbose`	Display verbose ouput (helpful for debugging)
`-l` or `--local`	For developers: allows you to import a log file from another system, so because the handles won't exist, it looks up random items in your local system to add hits to instead.
`-h` or `--help`	Help

Although the DSpace Log Convertor applies basic spider filtering (googlebot, yahoo slurp, msnbot), it is far from complete. Please refer to Filtering and Pruning Spiders for spider removal operations, after converting your old logs.

Filtering and Pruning Spiders

Command used:	`[dspace]/bin/dspace stats-util`
Java class:	org.dspace.statistics.util.StatisticsClient
Arguments (short and long forms):	Description
`-b` or `--reindex-bitstreams`	Reindex the bitstreams to ensure we have the bundle name
`-r` or `--remove-deleted-bitstreams`	While indexing the bundle names remove the statistics about deleted bitstreams
`-u` or `--update-spider-files`	Update Spider IP Files from internet into `[dspace]/config/spiders`. Downloads Spider files identified in `dspace.cfg` under property `solr.spiderips.urls`. See Configuration settings for Statistics
`-f` or `--delete-spiders-by-flag`	Delete Spiders in Solr By isBot Flag. Will prune out all records that have `isBot:true`
`-i` or `--delete-spiders-by-ip`	Delete Spiders in Solr By IP Address, DNS name, or Agent name. Will prune out all records that match spider identification patterns.
`-m` or `--mark-spiders`	Update isBot Flag in Solr. Marks any records currently stored in statistics that have IP addresses matched in spiders files
`-h` or `--help`	Calls up this brief help table at command line.

Notes:

The usage of these options is open for the user to choose. If you want to keep spider entries in your repository, you can just mark them using "`-m`" and they will be excluded from statistics queries when "`solr.statistics.query.filter.isBot = true`" in the `dspace.cfg`. If you want to keep the spiders out of the solr repository, just use the "`-i`" option and they will be removed immediately.

Spider IPs are specified in files containing one pattern per line. A line may be a comment (starting with "#" in column 1), empty, or a single IP address or DNS name. If a name is given, it will be resolved to an address. Unresolvable names are discarded and will be noted in the log.

There are guards in place to control what can be defined as an IP range for a bot. In `[dspace]/config/spiders`, spider IP address ranges have to be at least 3 subnet sections in length 123.123.123 and IP Ranges can only be on the smallest subnet [123.123.123.0 - 123.123.123.255]. If not, loading that row will cause exceptions in the dspace logs and exclude that IP entry.

Spiders may also be excluded by DNS name or Agent header value. Place one or more files of patterns in the directories `[dspace]/config/spiders/domains` and/or `[dspace]/config/spiders/agents`. Each line in a pattern file should be either empty, a comment starting with "#" in column 1, or a regular expression which matches some names to be recognized as spiders.

Export SOLR records to intermediate format for import into another tool/instance

Command used:	`[dspace]/bin/dspace stats-util`
Java class:	org.dspace.statistics.util.StatisticsClient
Arguments (short and long forms):	Description
`-e` or `--export`	Export SOLR view statistics data to usage statistics intermediate format

This exports the records to `[dspace]/temp/usagestats_0.csv`. This will chunk the files at 10,000 records to new files. This can be imported with `stats-log-importer` to SOLR Statistics

Export SOLR statistics, for backup and moving to another server

Command used:	`[dspace]/bin/dspace solr-export-statistics`
Java class:	org.dspace.util.SolrImportExport
Arguments (short and long forms):	Description
`-i` or `--index-name`	optional, the name of the index to process. "statistics" is the default. "authority" can also be exported.
`-l or --last` *`integer`*	optionally export only *integer* many days worth of statistics
`-d or --directory`	optional, directory to use for storing the exported files. By default, `[dspace]/solr-export` is used. If that is not appropriate (due to storage concerns), we recommend you use this option to specify a more appropriate location.

-f or --force-overwrite	optional, overwrite export file if it exists (DSpace 6.1 and later)

Import SOLR statistics, for restoring lost data or moving to another server

Command used:	`[dspace]/bin/dspace solr-import-statistics`
Java class:	org.dspace.util.SolrImportExport
Arguments (short and long forms):	Description
-i or --index-name	optional, the name of the index to process. "statistics" is the default. "authority" can also be imported.
`-c or --clear`	optional, clears the contents of the existing stats core before importing
`-d or --directory`	optional, directory which contains the files for importing. By default, `[dspace]/solr-export` is used. If that is not appropriate (due to storage concerns), we recommend you use this option to specify a more appropriate location.

Reindex SOLR statistics, for upgrades or whenever the Solr schema for statistics is changed

Command used:	`[dspace]/bin/dspace solr-reindex-statistics`
Java class:	org.dspace.util.SolrImportExport
Arguments (short and long forms):	Description
-i or --index-name	optional, the name of the index to process. "statistics" is the default
`-k or --keep`	optional, tells the script to keep the intermediate export files for possible later use (by default all exported files are removed at the end of the reindex process).
`-d or --directory`	optional, directory to use for storing the exported files (temporarily, unless you also specify `--keep`, see above). By default, `[dspace]/solr-export` is used. If that is not appropriate (due to storage concerns), we recommend you use this option to specify a more appropriate location. Not sure about your space requirements? You can estimate the space required by looking at the current size of `[dspace]/solr/statistics`
-f or --force-overwrite	optional, overwrite export file if it exists (DSpace 6.1 and later)

NOTE: `solr-reindex-statistics` is safe to run on a live site. The script stores incoming usage data in a temporary SOLR core, and then merges that new data into the reindexed data when the reindex process completes.

Upgrade Legacy DSpace Object Identifiers (pre-6x statistics) to DSpace 6x UUID Identifiers

This command was introduced in **DSpace 7.0** and will be included in the **DSpace 6.4** release as well.

It is recommended that all DSpace instances with legacy identifiers perform this one-time upgrade of legacy statistics records.

This action is safe to run on a live site. As a precaution, it is recommended that you backup you statistics shards before performing this action.

Note: a link to this section of the documentation should be added to the DSpace 6.4 Release Notes. (It is already noted in the DSpace 7.0 Upgrading DSpace page, step 11d)

The DSpace 6x code base changed the primary key for all DSpace objects from an integer id to UUID identifiers. Statistics records that were created before upgrading to DSpace 6x contain the legacy identifiers.

While the DSpace user interfaces make some attempt to correlate legacy identifiers with uuid identifiers, it is recommended that users perform this one time upgrade of legacy statistics records.

If you have sharded your statistics repository, this action must be performed on each shard.

Command used:	`[dspace]/bin/dspace solr-upgrade-statistics-6x`
Java class:	org.dspace.util.SolrUpgradePre6xStatistics

Arguments (short and long forms):	Description
- i or - -index-name	Optional, the name of the index to process. "statistics" is the default
`-n or --num_rec`	Optional. Total number of records to update (defaut=100,000). To process all records, set -n to 10000000 or to 100000000 (10M or 100M) If possible, please allocate 2GB of memory to this process (e.g. -Xmx2000m)
`-b or --batch_size`	Number of records to batch update to SOLR at one time (default=10,000).

NOTE: This process will rewrite most solr statistics records and may temporarily double the size of your statistics repositories.

If a UUID value cannot be found for a legacy id, the legacy id will be converted to the form "xxxx-unmigrated" where xxxx is the legacy id.

Solr Sharding By Year

The DSpace tool described below for managing Solr data through yearly sharding no longer functions in DSpace 7.x (see also https://github.com/DSpace/DSpace/issues/8478). Using these tools to manage Solr shards is no longer recommended. Alternative approaches are being explored and this page will be updated to reflect those findings.

Command used:	`[dspace]/bin/dspace stats-util`
Java class:	org.dspace.statistics.util.StatisticsClient
Arguments (short and long forms):	Description
`-s` or `--shard-solr-index`	Splits the data in the main Solr core up into a separate core for each year. This will upgrade the performance of Solr.

Notes:

Yearly Solr sharding is a routine that can drastically improve the performance of your DSpace SOLR statistics. It was introduced in DSpace 3.0 and is not backwards compatible. The routine decreases the load created by the logging of new usage events by reducing the size of the SOLR Core in which new usage data are being logged. By running the script, you effectively split your current SOLR core, containing all of your usage events, into different SOLR cores that each contain the data for one year. In case your DSpace has been logging usage events for less than one year, you will see no notable performance improvements until you run the script after the start of a new year. Both writing new usage events as well as read operations should be more performant over several smaller SOLR Shards instead of one monolithic one.

It is highly recommended that you execute this script once at the start of every year. To ensure this is not forgotten, you can include it in your crontab or other system scheduling software. Here's an example cron entry (just replace [dspace] with the full path of your DSpace installation):

```
# At 12:00AM on January 1, "shard" the DSpace Statistics Solr index.  Ensures each year has its own Solr index
- this improves performance.
0 0 1 1 * [dspace]/bin/dspace stats-util -s
```

You MUST restart Tomcat after sharding

After running the statistics shard process, the "View Usage Statistics" page(s) in DSpace will **not** automatically recognize the new shard.

Restart tomcat to ensure that the new shard is recognized & included in usage statistics queries.

Repair of Shards Created Before DSpace 5.7 or DSpace 6.1

If you ran the shard process before upgrading to DSpace 5.7 or DSpace 6.1, the multi-value fields such as owningComm and onwningColl are likely be corrupted. Previous versions of the shard process lost the multi-valued nature of these fields. Without the multi-valued nature of these fields, it is difficult to query for statistics records by community / collection / bundle.

You can verify this problem in the solr admin console by looking at the owningComm field on existing records and looking for the presence of "\\," within that field.

The following process may be used to repair these records.

1. Backup your solr statistics-xxxx directories while tomcat is down.
2. Backup and delete the contents of the dspace-install/solr-export directory
3. For each "statistics-xxxx" shard that exists, export the repository

```
dspace solr-export-statistics -i statistics-xxxx -f
```

4. Run the following to repair records in the dspace-install/solr-export directory

```
for file in *
do
sed -E -e "s/[\\]+,/,/g" -i $file
done
```

5. For each shard that was exported, run the following import

```
dspace solr-import-statistics -i statistics-xxxx -f
```

If you repeat the query that was run previously, the fields containing "\\," should now contain an array of owning community ids.

Shard Naming

Prior to the release of DSpace 6.1, the shard names created were off by one year in timezones with a positive offset from GMT.

Shar

See Unable to locate Jira server for this macro. It may be due to Application Link configuration.

Technical implementation details

After sharding, the Solr data cores are located in the [dspace.dir]/solr directory. There is no need to define the location of each individual core in solr.xml because they are automatically retrieved at runtime. This retrieval happens in the *static* method located in the *org.dspace.statistics.SolrLogger* class. These cores are stored in the *statisticYearCores* list. Each time a query is made to Solr, these cores are added as shards by the *addAdditionalSolrYearCores* method. The cores share a common configuration copied from your original *statistics* core. Therefore, no issues should be resulting from subsequent `an t updates`.

The actual sharding of the of the original Solr core into individual cores by year is done in the *shardSolrIndex* method in the *org.dspace.statistics.SolrLogger* class. The sharding is done by first running a facet on the time to get the facets split by year. Once we have our years from our logs we query the main Solr data server for all information on each year & download these as CSVs. When we have all data for one year, we upload it to the newly created core of that year by using the update csv handler. Once all data of one year have been uploaded, those data are removed from the main Solr (by doing it this way if our Solr crashes we do not need to start from scratch).

Multiple Shard Fix (DSpace 6.1)

A bug exists in the DSpace 6.0 release that prevents tomcat from starting when multiple shards are present.

To address this issue, the initialization of SOLR shards is deferred until the first SOLR related requests are processed.

See Unable to locate Jira server for this macro. It may be due to Application Link configuration.

Testing Solr Shards

Testing Solr Shards