Optimizing the Sync and Resync Processes in Metadb

Basics

The Metadb Test instance is currently set up to synchronize data with the FOLIO Test instance. The goal is to optimize, document, and understand this process to prepare to synchronize data between Metadb Test and FOLIO Production.

Debezium Connector, Synch, and Resynch

Our hosting team is preparing for synchronizing our Metadb Test instance with our FOLIO Production instance. The Debezium connector reads the FOLIO database and populates Kafka with topic messages, which takes about 31 hours for Cornell's data, so that is okay. The part that is really slow is where Metadb is consuming the Kafka topics and populating the instance with either new data or changes to data. The most recent synchronization test took 8 days. We are trying to figure out why this is happening so we can get the sync down to a more reasonable timeframe, like 2 days. Our hosting team is running Metadb version 1.2 on an instance that meets the specs per the metadb documentation. Getting the synching and resynching down to shorter timeframes is our top priority right now.

John Malconian from Index Data indicates that given that it takes 31 hours for Debezium to complete the initial snapshot, 6-8 days to complete the Metadb portion of the snapshot is about normal based on his experience. This also depends on the data set. He says Nassib is better equipped to comment on Metadb performance specifically, but, generally speaking, it’s going to take longer for Metadb to consume the snapshot than Debezium. Metadb is doing data transformation and database operations that just take longer than reading a logical replication slot and pushing messages into Kafka.   From an Ops perspective, ensure that the Metadb database is adequately sized to handle the processing (CPU) and that it is not hindered by I/O limitations.   If using RDS for the Metadb database, provisioned IOPS or gp3 storage is best (John prefers gp3).

Carole Godfrey from the hosting team says they have monitored cpu usage (it did not go above ~26% during the initial sync process) and we are using gp3 storage. One thing that they are trying to test is the resync process. The MetaDB database has been populated from the initial sync and now they need to adjust and get it in sync with an updated RDS instance (in this case - FOLIO has been upgrade from Orchid --> Poppy). Carole wonders if John thinks a resync should take the same or more/less time to complete as the initial sync. John says It’s going to take about the same time whether it’s an initial sync or re-sync.   It’s essentially the same process either way.

Carole says they did get 1 resync running and trying to understand if it completed successfully -- but wasnt sure how to confirm its success. We had gotten messages when running an endsync
I continued on from this step - but wasnt sure on queries or ? to confirm all is in syncTable sizes did not match up (when compared to FOLIO) -- but filtering __current = true

SELECT count(*) FROM folio_inventory.instance__ where __current = true;

SELECT count(*) FROM folio_inventory.instance__t__ where __current = true;

!!

34% of current records have not been confirmed by the new snapshot.

The unconfirmed records will be marked as deleted.

!!

(From Nassib) This warning is not always a problem. It means that a lot of the current data in Metadb have not (so far) been confirmed by the new snapshot.This could happen if many records were deleted in the source (FOLIO) during the time when the databases were not synchronized.But typically it indicates that a lot of snapshot data have not yet been read from Kafka. If this is the reason for the warning, then running endsync would remove those records in Metadb until they are eventually streamed. So the records would temporarily be unavailable in the current tables in Metadb.

(Carole) In the case of a resync -- should counts from these 2 queries alignIn MetaDB:
`SELECT count(*) FROM folio_inventory.instance__ where __current = true;`In FOLIO
`SELECT count(*) FROM tenant_mod_inventory_storage.instance;`

`(Nassib) Yes`

(Carole) Also with each resync - is it expected that the count of records in Metadb tables will increase by the number of records in the corresponding FOLIO table?

(Nassib) The counts should be the same when the tables are synchronized.

(Carole) Ok -- thanksAfter the initial sync process - observed these 2 queries aligned
(1)SELECT count(*) FROM folio_inventory.instance__
(2)SELECT count(*) FROM tenant_mod_inventory_storage.instance

After a resync - observed results from (1) was ~2X the size of (2) and I needed to filter by `_current` to get counts that more aligned

Safe to assume the filtering by `_current` is needed?

(Nassib) Yes, non-current records are not retained by FOLIO.

(Carole) OK, thanks

(Nassib) In other words, FOLIO discards the non-current records but Metadb retains them. So we can only compare counts of the current records.

More on Resync

(Carole) When performing a resync of a data source in MetaDB https://d1f3dtrg62pav.cloudfront.net/doc/#_resynchronizing_a_data_sourceShould the associated Debezum Connector `snapshot.mode` be set to something other than `initial`?For example - Is an ad-hoc snapshot or signalling tables (described below) required for the MetaDB resync process? https://debezium.io/documentation/reference/stable/connectors/postgresql.html#postgresql-ad-hoc-snapshotsOr can the same process as followed for the initial sync be assumed?

(John) Hi Carole - I only use the default which is `initial` . Essentially, take the snapshot and then go into streaming update mode.

My Debezium configs typically look like this (v1.8 or v1.9):

```
{
  "name": "<connector_name>",
  "config": {
    "connector.class": "io.debezium.connector.postgresql.PostgresConnector",
    "database.dbname": "<db name>",
    "database.hostname": "<db host>",
    "database.password": "<db user password>",
    "database.port": "5432",
    "database.server.name": "<prepend to kafka topic>",
    "schema.exclude.list": ".*_mod_login,.*_mod_pubsub,public,.*pubsub_config,supertenant_mod_.*,.*_mod
_kb_ebsco_java,.*__system",
    "table.exclude.list": ".*_mod_agreements\\.entitlement_tag,.*_mod_agreements\\.erm_resource_tag,.*_
mod_agreements\\.string_template,.*_mod_agreements\\.string_template_scopes,.*_mod_agreements\\.templat
ed_url,.*_mod_data_export_worker\\.batch_job_execution_params,.*_mod_oai_pmh\\.instances,.*_mod_remote_
storage\\.original_locations,.*_mod_remote_storage\\.item_notes,.*_mod_service_interaction\\.dashboard_
access,.*app_setting,.*alternate_name,.*databasechangelog,.*databasechangeloglock,.*directory_entry_tag
,.*license_document_attachment,.*license_supp_doc,.*license_tag,.*log_entry_additional_info,.*subscript
ion_agreement_supp_doc,.*subscription_agreement_document_attachment,.*subscription_agreement_ext_lic_do
c,.*subscription_agreement_tag,.*tenant_changelog,.*tenant_changelog_lock,.*marc_indexers.*,.*rmb_inter
nal.*,.*rmb_job.*",
    "database.user": "<db user>",
    "heartbeat.interval.ms": "30000",
    "heartbeat.action.query": "UPDATE id_dbz.heartbeat_query set last_heartbeat_ts = now();",
    "plugin.name": "pgoutput",
    "publication.autocreate.mode": "filtered",
    "tasks.max": "1",
    "truncate.handling.mode": "include"
  }
}
```

(Carole) We are running version 2.3.4 of Debezium (which was the latest at the time first setting things up) -- assuming its ok to run a version 2.X?

(John) I don't see why not as long as it's compatible with your version of Kafka and Postgresql. Personally, though, I haven't tried 2.x quite yet. We are still using Kafka 2.8 and so 1.8 and 1.9 have suits our needs just fine. I have it on my list to test with 2.x at some point soon.