Files Didn't Finish Processing Because ElasticSearch DB Node was dropped

We had an incident close to when we migrated our ElasticSearch DB to Elastic Cloud, where we did not have replicas of our elastic search shards, and during regular maintenance (seriously, they said it was regular maintenance) they decided to remove a node from our cluster with a new one without copying over the data from the original node. This led to about 10% of are data being deleted and 100 or so unassigned shards that we could not read or write to until we restored our elastic search db from a backup.

This caused lots of files being processed to fail to process at the very end of processing because their extracted text could not be saved to elastic search. Also, because we were restoring the Elastic Search DB from a backup, the files that did process successfully would need to be reindexed in the elastic search db because all data that had been added since the incident was lost from the db (because we restored from a backup).

To remediate this situation, we added some functions to file processing, that given a specific time period, would reprocess all items, so that extracted text could be sent to elastic search. We add to modify the file processing process a little bit, and make it so that when reprocessing, a new file version wouldn't be created again if that had already happened. Note that this process ended up being quite finicky, and required quite a bit of manual effort so that the cosmos db was not overloaded.

We have since added replicas to our elastic search db, and don't expect an incident like this to happen again.

We have also made it so if a file is completely processed except for saving extracted text to elastic search, it will put the identifying file information in special queue with an infinite expiration, so that when elastic search is back up again, we can just process this queue and only send their extracted text to elastic search, instead of completely sending the item through file processing again.

How to do this?

It's been to long since we did this for me to document in great detail how you do this (likely you won't need to do this in the future again either because of the remediation steps we've taken) but I will do my best to write a guide, just in case.

You'll want to have a dev build out a cosmos db query that you can run manually (sorry I don't know where it got put) from the query that is run the RequeueProcessingItemsFunction in the fileProcessingManager function app that gets all the items to be requeued, so you can use it to make sure everything gets queued.

You will want to disable other file processing while you queue things for processing so you don't overload cosmos db.

When you run the RequeueProcessingItemsFunction, you will want to disable the `ProcessRequeuedItems` function in the `revver-fileProcessingManager` function app, and make sure that expected number of items get placed in the `processrequeueditemsqueue` queue in the `revvertextextractprocus` storage account. (use the query generated previously to get expected number of items to queue).

Once you've validated the expected number of items were queued (you can clear the queue and re queue with a smaller time window if needed, but avoid if possible because the same item can potentially appear in different time windows just becuase of the way items are timestamped during file processing), enable the `ProcessRequeuedItems` function, but only do this for about a minute, after a minute disable and let things process for a few minutes. If you don't do this, Cosmos Db will be overwhelmed and you will start getting unexpected errors.

Once you've done the previous step, and the queue is empty, you should disable the ProcessRequeuedItems function, and re-enable the normal file processing functions that you disabled, and file processing should work per usual (you will just have quite a few items in the initiate ocr and text extraction queues that will need to be processed)

Revision #5
Created 22 August 2024 23:03:46 by Quinn Godfrey
Updated 22 August 2024 23:32:15 by Quinn Godfrey