Merge July 25 changes :

1. Merge
   1. Changes in GetSearchIndex to pass curL1SizeInMB , maxSizeInMB to stored procedure.

File :

CloudArchive\Source\LiveOffice.CloudArchive.Search.Merge.Data.Db\Repositories\Data Mappers\SearchSearchIndexDataMapper.cs

public virtual List<SearchIndex> GetSearchIndex(int groupId, int totalDocCount, int maxDocCount, decimal curL1SizeInMB, decimal maxSizeInMB, eIndexTypes indexType, eAppTypes appType, int maxShards, Guid batchGuid, out bool issuccess)

* 1. Change in **Get L1 Step**

File : ***CloudArchive\Source\LiveOffice.CloudArchive.Search.Merge.BL\Core\Steps\Step1\GetL1Step.cs***

* + 1. We accumulate DocCount in totalDocCount , size in totalSize
    2. The count of L1 that reach the Max DocCount or MaxIndexSize is the batch will be merged
    3. The L1 that is over our MaxDcoCount and MaxSize will be released.
  1. Change in **GetL3** Step it consider L1Size & L1DocCount , it pass the currentDocCount and the Current Size to the stored procedure to select best candidates.

File : ***CloudArchive\Source\LiveOffice.CloudArchive.Search.Merge.BL\Core\Steps\Step2\GetL3Step.cs***

* + 1. It keep selecting shards as long as the totalDocCount and totalSize is less than the max
    2. The L3 unused will be unlocked
  1. Change in **Merge To New L3 Step**

File: **CloudArchive\Source\LiveOffice.CloudArchive.Search.Merge.BL\Core\Steps\Step3\MergeToNewL3Step.cs**

* + 1. We are validating L3 index after download and checking for DocCount

If index is valid and DocCount>0 we added to the shards that will be merged.

If not it will be excluded from merge and unlocked before we start merging

* 1. Change in Upload and Update Indices step

File : **CloudArchive\Source\LiveOffice.CloudArchive.Search.Merge.BL\Core\Steps\Step5\UploadAndUpdateIndexciesStep.cs**

* + 1. We are Validating the Index locally before upload if the index is valid and the DocCount >0 we continue to upload otherwise rollback.
    2. After uploading We are Validating the Index Remotely before Activating the index if the index is valid and the DocCount >0 we continue to Activate/DeActivate otherwise rollback.
  1. Change in FileProcessor

File : **CloudArchive\Source\LiveOffice.CloudArchive.Search.Merge.BL\Core\Processors\FileProcessor.cs**

Unify and fix all the retry Count Issues

* 1. Change in

1. DB
   1. AppType=0 change

CREATE PROC [dbo].[stp\_SearchIndex\_TurnonSearchIndexes]

@NewIndexId INT,

@NewDocCount INT,

@NewDocSize DECIMAL(10,4),

@Indexes\_Deactivate Tp\_SearchIndexEntityType READONLY,

@GroupID INT,

@BatchGuid UNIQUEIDENTIFIER,

@AppType INT

AS

BEGIN

SET XACT\_ABORT ON

 BEGIN TRANSACTION

   --activate index

  UPDATE SearchIndex

     SET active=1,

                  islocked=0,

                  DocCount=@NewDocCount,

                  SizeInMb=@NewDocSize,

                  lastUpdate=GetDate(),

                  BatchGUID=NULL,

                 AppType=0

     WHERE  indexid=@NewIndexId AND

                  BatchGUID=@BatchGuid AND

                  AppType=@AppType AND

                  groupnum=@GroupID

   --deactivate indexes

  UPDATE SearchIndex

       SET active=0,

                  DocCount=0,

                  SizeInMB=0,

                  lastUpdate=GetDate(),

                  AppType=0

       WHERE      indexid IN (SELECT IndexId FROM @Indexes\_Deactivate) AND

                  BatchGUID=@BatchGuid AND

                  AppType=@AppType AND

                  groupnum=@GroupID

   --update reference in IndexLockRequestQueueItem table

   UPDATE IndexLockRequestQueueItem SET NewSearchIndexID=@NewIndexId WHERE SearchIndexID in (SELECT IndexId FROM @Indexes\_Deactivate) AND IsComplete=0

COMMIT TRANSACTION

END

GO

CREATE PROC [dbo].[stp\_SearchIndex\_UnlockSearchIndexes]

      @SearchIndexes\_Ids Tp\_SearchIndexEntityType READONLY,

      @GroupId INT,

      @AppType TINYINT,

      @BatchGuid UNIQUEIDENTIFIER=NULL

AS

BEGIN

  UPDATE SearchIndex

      SET IsLocked=0,

            lastUpdate=GetDate(),

            BatchGUID=NULL,

            AppType=0

  WHERE indexid IN (select IndexId from @SearchIndexes\_Ids)

     AND IsLocked=1

       AND groupnum=@GroupId

       AND AppType=@AppType

       AND

      (

        BatchGUID=ISNULL(@BatchGuid,BatchGUID) OR BatchGUID IS NULL

     )

END

GO

CREATE PROC [dbo].[Stp\_SearchIndex\_UnlockExpired]

(

@ExpireHours int=32

)

AS

BEGIN

      UPDATE searchindex

            SET IsLocked=0 ,

                BatchGuid=NULL,

            AppType=0,

            active = CASE

                                    WHEN DocCount > 0 THEN 1

                                    ELSE 0

                     END

      WHERE indexid IN

            (

                  SELECT indexid

                  FROM searchindex (NOLOCK)

                  WHERE Islocked=1 AND

                          AppType<> 5 AND

                              (DATEADD(HOUR,@ExpireHours,DateLeased)<GETDATE() OR DateLeased IS NULL))

END

* 1. Get L3 Candidates

ALTER PROCEDURE [dbo].[Stp\_SearchIndex\_GetLevel3IndexBestCandidateAndLock]

      @TotalDocCount INT,

      @MaxIndexDocCount INT,

      @GroupId INT,

      @Type TINYINT,

      @AppType TINYINT,

      @MaxShards INT=4,

      @BatchGuid UNIQUEIDENTIFIER,

      @CurSizeInMB DECIMAL(10,4),

      @MaxSizeInMB DECIMAL(10,4)

AS

BEGIN

      DECLARE @Tbl TABLE(IndexId INT)

      IF @AppType = 2 -- Remerge Service

      BEGIN

      SET @Type= 3 -- index type distinguish indexes created by remerge service.

      END

     DECLARE @Tbl\_IndexesToExclude TABLE(IndexId INT)

       -- Lock this section so that no other agents can grab the same mails to index

       DECLARE @LockResult int, @LockName varchar(50)

       SELECT @LockName = 'SearchIndex\_GetLevel3IndexBestCandidateAndLock'

       SET XACT\_ABORT ON

      BEGIN TRANSACTION

     EXEC  @LockResult = sp\_getapplock @lockname, 'Exclusive', 'Transaction', 300

       IF @LockResult IN (0,1)

       BEGIN

             INSERT INTO @Tbl\_IndexesToExclude

            SELECT DISTINCT ISNULL(NewSearchIndexID,SearchIndexID)

             FROM IndexLockRequestQueueItem (NOLOCK) A INNER JOIN

             IndexLockRequestQueue (NOLOCK) B ON A.LockQueueID=B.LockQueueID AND A.IsComplete=0

            WHERE A.IsComplete=0 AND B.Step < 2 AND

             GETDATE()<B.DateExpired

             SELECT TOP (@MaxShards) indexid INTO #TEMP

            FROM searchindex (NOLOCK)

             WHERE groupnum = @GroupId and TYPE=@Type and active=1 and IsLocked=0 and DocCount<=@MaxIndexDocCount-@TotalDocCount

             AND (SizeinMB<@MaxSizeInMB-@CurSizeInMB OR SizeInMB IS NULL)

            ORDER BY DocCount ASC

            UPDATE searchindex SET IsLocked=1,BatchGuid=@BatchGuid,DateLeased=GETDATE(),AppType=@AppType

             OUTPUT inserted.indexid INTO @Tbl

            WHERE indexid IN

                   (

                    SELECT IndexID FROM #TEMP

                  )

                  AND NOT EXISTS(SELECT IndexId FROM @Tbl\_IndexesToExclude B WHERE B.IndexId=searchindex.IndexID)

                  AND IsLocked=0

              -- Remove the lock

                    EXEC sp\_releaseapplock @LockName, 'Transaction'

       END

       ELSE

            PRINT 'Applocked'

      COMMIT TRANSACTION

        SELECT indexid,indexpath,groupnum,[Type],LeaseSeconds,active,DocCount,IsLocked,lastUpdate,SizeInMB,BatchGUID,DateLeased,AppType

        FROM searchindex (NOLOCK)

        WHERE indexid IN

        (

         SELECT indexid FROM @Tbl

        )

END

GO

* 1. setsearchindexID batch , removing setting batch guid for the history

ALTER PROCEDURE [dbo].[Stp\_MailsToIndex\_SetSearchIndexIDBatch]

(

@oldindexIds Tp\_SearchIndexEntityType READONLY,

@search\_index\_id int,

@GUID uniqueidentifier = null

)

AS

begin

SET NOCOUNT ON;

declare @uid bigint = 0

select uid into #uid

from MailsToIndex (NOLOCK)

WHERE IndexID in (select IndexID from @oldindexIds)

create clustered index ix\_u on #uid(uid)

while (1=1)

begin

select top 100 uid into #u

from #uid

where uid>@uid

order by uid

if @@ROWCOUNT = 0

break

UPDATE MailsToIndex

SET DateFinished=GETDATE(),

IndexID=@search\_index\_id,

~~BatchGUID=ISNULL(@GUID, BatchGUID)~~

WHERE UID IN

(SELECT uid FROM #u)

select @uid=MAX(uid) from #u

drop table #u

end

end

1. Config changes

<add key="MaxIndexSize" value="3221225472" />  (3GB)

<add key="MaxIndexShardsToMerge" value="20" />   (need to disscuss)

(will be handled in code, just need more candidates in case if sizeinmb is null

<add key="MaxDocCount" value="40000" /> (Red will provide more data)

<add key="SANRetryCount" value="5" /> (we fix the SANRrtry issue)