CREATE PROC [dbo].[Stp\_IndexLockQueueItem\_GetIndexesToProcess]

(

@LockQueueID INT,

@GroupID INT,

@IndexID INT=NULL,

@AppType TINYINT,

@MaxRows TINYINT=1

 )

AS

BEGIN

--if queue is expired return nothing

IF (EXISTS(SELECT 1 FROM IndexLockRequestQueue (NOLOCK)

  WHERE LockQueueID=@LockQueueID AND (DateExpired<GETDATE() OR Step=2)))

   RETURN

--if no indexid passing it returns any availble index ready to process

DECLARE @Tbl TABLE(IndexId INT)

UPDATE TOP (@MaxRows) searchindex SET IsLocked=1,DateLeased=GETDATE(),AppType=@AppType,BatchGUID=NEWID()

 OUTPUT inserted.indexid INTO @Tbl

WHERE

groupnum=@GroupID AND

indexid=ISNULL(@IndexID,indexid)

AND active=1 AND BatchGUID IS NULL

AND (

(IsLocked=1 AND AppType=@AppType)

OR

(IsLocked=0 AND

EXISTS

    (

      SELECT 1 FROM IndexLockRequestQueueItem (NOLOCK)

  WHERE LockQueueID=@LockQueueID

  AND IsComplete=0

  AND ISNULL(NewSearchIndexID,SearchIndexID)=indexid

    )

)

)

    IF (EXISTS(SELECT 1 FROM @Tbl))

    BEGIN

       DECLARE @DateExpired DATETIME

       SELECT @DateExpired = DateExpired

   FROM IndexLockRequestQueue (NOLOCK)

   WHERE LockQueueID=@LockQueueID

       --prolong dateexpired in IndexLockRequestQueue table for two hours, if it's less than 3 hours, because you are going to work on it. Hopefully you will finish for that time

       IF (DATEDIFF(hour,GETDATE(),@DateExpired)<=3)

       BEGIN

         UPDATE IndexLockRequestQueue

         SET DateExpired=DATEADD(hour,2,DateExpired)

         WHERE LockQueueID=@LockQueueID

       END

      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

END

GO

CREATE PROC [dbo].[Stp\_IndexLockQueueItem\_GetIndexToProcessByApp](@AppType TINYINT)

AS

BEGIN

    DECLARE @TblQueue TABLE(LockQueueID INT,GroupID INT, IndexID INT,OldIndexID INT, DateExpired DATETIME,IsComplete BIT)

    INSERT INTO @TblQueue(LockQueueID, GroupID, IndexID, OldIndexID, DateExpired,IsComplete)

    SELECT A.LockQueueID,

           A.GroupID,

        ISNULL(B.NewSearchIndexID,B.SearchIndexID),

     B.SearchIndexID,

     A.DateExpired,

     ISNULL(B.IsComplete,0)

 FROM IndexLockRequestQueue A (NOLOCK)

    LEFT JOIN IndexLockRequestQueueItem B (NOLOCK) ON A.LockQueueID=B.LockQueueID

    WHERE A.AppType=@Apptype

    AND A.Step>0

    AND A.DateExpired>GETDATE()

    IF (NOT EXISTS(SELECT 1 FROM @TblQueue)) RETURN

    DECLARE @Tbl TABLE(IndexId INT, GroupID INT)

    UPDATE TOP (1) searchindex SET IsLocked=1,DateLeased=GETDATE(),AppType=@AppType, BatchGUID=NEWID()

  OUTPUT inserted.indexid,inserted.groupnum INTO @Tbl

    WHERE BatchGUID IS NULL

AND active=1

AND

    (

    (

    EXISTS(

       SELECT 1 FROM @TblQueue A

   WHERE A.IndexId=searchindex.indexid

   AND A.IsComplete=0

    )

    AND IsLocked=0

    )

    OR

    (

      EXISTS(

       SELECT 1 FROM @TblQueue A

   WHERE A.GroupID=searchindex.groupnum

      )

      AND IsLocked=1 AND Apptype=@Apptype

    )

    )

    IF (EXISTS(SELECT 1 FROM @Tbl))

    BEGIN

       DECLARE @LockQueueID INT

       DECLARE @IndexID INT

       DECLARE @OldIndexID INT

       DECLARE @GroupID INT

       SELECT TOP 1 @IndexID=IndexId,@GroupID=GroupID FROM @Tbl

       SELECT TOP 1 @LockQueueID=A.LockQueueID,@OldIndexID=A.OldIndexID

       FROM @TblQueue A

       WHERE A.IsComplete=0

   AND A.GroupID=@GroupID

       ORDER BY A.DateExpired

       --prolong dateexpired in IndexLockRequestQueue table for two hours, because you are going to work on it. Hopefully you will finish for that time

       UPDATE IndexLockRequestQueue

       SET DateExpired=DATEADD(hour,2,DateExpired)

       WHERE LockQueueID=@LockQueueID AND DateExpired>GETDATE() AND DateExpired<DATEADD(hour,2,GETDATE())

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

       FROM searchindex (NOLOCK)

    WHERE indexid=@IndexID

    END

END

GO

CREATE PROC [dbo].[Stp\_IndexLockRequestQueue\_Add](@GroupID INT, @AppType TINYINT,@DateRequested DATETIME,

@DateExpired DATETIME,@Indexes Tp\_SearchIndexEntityType READONLY)

AS

BEGIN

DECLARE @LockQueueID INT

DECLARE @Tbl TABLE(IndexId INT)

DECLARE @TblIndexes TABLE(IndexId INT)

     --lock all available indexes

     IF (NOT EXISTS(SELECT 1 FROM @Indexes))

     BEGIN

        INSERT INTO @TblIndexes

        SELECT indexid FROM searchindex (NOLOCK) WHERE groupnum=@GroupID

     END

     ELSE

     BEGIN

        INSERT INTO @TblIndexes

        SELECT IndexId FROM @Indexes

     END

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

       OUTPUT inserted.indexid INTO @Tbl

      WHERE EXISTS(select 1 from @TblIndexes A WHERE A.IndexID=searchindex.indexid AND groupnum = @GroupId AND IsLocked=0 AND active=1)

    --create parent record

     INSERT INTO IndexLockRequestQueue(GroupID,AppType,DateRequested,DateExpired,Step,DateFinished)

     VALUES (@GroupID,@AppType,@DateRequested,@DateExpired,0,NULL)

     SET @LockQueueID = SCOPE\_IDENTITY()

     --insert locked indexes by other apps into child table

     INSERT INTO IndexLockRequestQueueItem(LockQueueID,SearchIndexID,DateRequested,IsComplete)

     SELECT @LockQueueID,indexid,@DateRequested,0

       FROM searchindex (NOLOCK) A WHERE groupnum = @GroupId

      AND EXISTS(SELECT 1 FROM @TblIndexes T WHERE T.IndexId=A.indexid)

      AND

       (

      (

       NOT EXISTS(SELECT 1 FROM @Tbl B WHERE B.IndexId=A.indexid)

       AND A.active=1 and A.IsLocked=1

      )

        OR (@Apptype=4 AND active=0 AND IsLocked=1 AND AppType=3)

     )

SELECT LockQueueID,GroupID,AppType,DateRequested,DateExpired,Step,DateFinished

FROM IndexLockRequestQueue (NOLOCK)

WHERE LockQueueID=@LockQueueID

END

GO

CREATE PROC [dbo].[Stp\_IndexLockRequestQueue\_UpdateStep](@LockQueueID INT, @Step TINYINT)

AS

BEGIN

  UPDATE IndexLockRequestQueue

         SET Step=@Step

         WHERE LockQueueID=@LockQueueID

END

GO

/\*it should return 0, if there are still children

it should return 1 and update parent step to 2 (complete), when no children left and unlock unused indexes

\*/

CREATE PROC [dbo].[Stp\_IndexLockRequestQueueItem\_AllDone](@LockQueueID INT,@AppType TINYINT,@GroupID INT,@IsExpired bit OUT)

AS

BEGIN

IF (

    EXISTS(

       SELECT 1 FROM IndexLockRequestQueue (NOLOCK)

   WHERE LockQueueID=@LockQueueID

   AND DateExpired<GETDATE()

  )

    OR NOT EXISTS(

       SELECT 1 FROM IndexLockRequestQueueItem (NOLOCK) WHERE LockQueueID=@LockQueueID

   AND IsComplete=0)

         )

  BEGIN

       UPDATE IndexLockRequestQueue

   SET Step=2, DateFinished=GETDATE()

   WHERE LockQueueID=@LockQueueID

       SET @IsExpired = (

                     SELECT 1 FROM IndexLockRequestQueueItem (NOLOCK)

 WHERE LockQueueID=@LockQueueID

 AND IsComplete=0

 )

       SELECT 1

  END

SELECT 0

END

GO

CREATE PROC [dbo].[Stp\_IndexLockRequestQueueItem\_Get](@LockQueueID INT)

AS

BEGIN

 SELECT LockQueueIDItem, LockQueueID,SearchIndexID,NewSearchIndexID,IsComplete

 FROM IndexLockRequestQueueItem

 WHERE LockQueueID=@LockQueueID

END

GO

CREATE PROC [dbo].[Stp\_IndexLockRequestQueueItem\_SetCompleteIndex](@LockQueueID INT,@Indexes Tp\_SearchIndexEntityType READONLY,@AppType TINYINT)

AS

BEGIN

  UPDATE IndexLockRequestQueueItem

  SET IsComplete=1,DateFinished=GETDATE()

  WHERE LockQueueID=@LockQueueID

  AND

  EXISTS(

         SELECT 1 FROM @Indexes A

 WHERE A.IndexID=ISNULL(NewSearchIndexID,SearchIndexID)

)

END

GO

CREATE PROC [dbo].[Stp\_SearchIndex\_ResetBatchGUID](@indexid INT, @apptype TINYINT)

AS

BEGIN

UPDATE searchindex SET BatchGUID=NULL

WHERE indexid=@indexid AND AppType=@apptype

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\_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

     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()

     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

**RetentionPolicy DB:**

Create PROC [dbo].[Stp\_RetentionBatchMail\_GetExpiredMailIDs]

(

@GroupID INT,

@LastExpiredMailID BIGINT,

@MaxRows INT

)

AS

BEGIN

SELECT TOP (@MaxRows) MailID

FROM RetentionBatchMail rbm (NOLOCK)

WHERE

MailID > @LastExpiredMailID

AND EXISTS

(

SELECT \* FROM RetentionBatch rb (NOLOCK)

WHERE rb.BatchID = rbm.BatchID

and rb.GroupID=@GroupID

)

END

**SearchIndexing / SearchIndexingRebuilding:**

USE [SearchIndexing]

GO

Create PROC [dbo].[Stp\_MailsToIndex\_GetLastMailIDForIndexes]

(

@Indexes Tp\_SearchIndexEntityType READONLY,

@GroupID INT

)

AS

BEGIN

SELECT MAX(EndMailID)

FROM MailsToIndex (NOLOCK)

WHERE GroupID=@GroupID

AND IndexID in

(SELECT IndexID FROM @Indexes)

END

GO

CREATE PROC [dbo].[Stp\_MailsToIndex\_DeleteRemergedRanges]

(

@LatMailIdRemerged BIGINT=NULL,

@GroupID INT

)

AS

BEGIN

DECLARE @i INT=1

WHILE @i>0

BEGIN

BEGIN TRANSACTION

SELECT TOP(1000)\* INTO #temp

FROM MailsToIndex

WHERE GroupID= @GroupID

AND EndMailID <=ISNULL(@LatMailIdRemerged,EndMailID)

DELETE m FROM MailsToIndex m

JOIN #temp t on m.uid=t.uid

SET @i = @@ROWCOUNT;

DROP TABLE #temp

COMMIT TRANSACTION

END

END

GO

Create PROCEDURE [dbo].[stp\_SearchIndex\_GetNewIndexPath]

@Group\_id INT,

@IndexType TINYINT,

@AppType TINYINT,

@LeaseSec INT,

@BatchGuid UNIQUEIDENTIFIER,

@IndexPathEmpty varchar(260)

AS

BEGIN

DECLARE @LockResult INT

DECLARE @MaxLevelNum INT

DECLARE @IndexId INT

DECLARE @IndexPath VARCHAR(260)

IF @AppType = 2 -- ReMerge service

BEGIN

SET @IndexType = 3 -- the index type that belong to ReMerge service

END

SELECT TOP 1 @IndexPath=indexpath,@IndexId=indexid FROM searchindex (NOLOCK)

WHERE groupnum = @Group\_id and active=0 and IsLocked=0 and DocCount=0

--if no indexpath found select any

IF (ISNULL(@IndexPath,'')='')

BEGIN

SET @IndexId=NULL

SELECT TOP 1 @MaxLevelNum=CONVERT(INT,REVERSE(SUBSTRING(REVERSE(indexpath),0,CHARINDEX('\', REVERSE(indexpath))))),

@IndexPath=@IndexPathEmpty

FROM searchindex (NOLOCK)

WHERE groupnum = @group\_id

ORDER BY CONVERT(INT,REVERSE(SUBSTRING(REVERSE(indexpath),0,CHARINDEX('\', REVERSE(indexpath))))) DESC

END

IF (ISNULL(@IndexPath,'')='')

BEGIN

SET @IndexPath=@IndexPathEmpty

SET @MaxLevelNum=0

END

IF (ISNULL(@IndexId,-1)=-1)

BEGIN

SET @MaxLevelNum=@MaxLevelNum+1

--because we have indexpath unique index, i don't use locking for some it will fail if both will receive same indexpath

IF @MaxLevelNum<10

SET @IndexPath=@IndexPath+'0'+CONVERT(VARCHAR(200),@MaxLevelNum)

ELSE

SET @IndexPath=@IndexPath+CONVERT(VARCHAR(200),@MaxLevelNum)

INSERT INTO searchindex(indexpath,groupnum,Type,LeaseSeconds,active,DocCount,IsLocked,lastUpdate,SizeInMB,BatchGUID,DateLeased,AppType)

VALUES (@IndexPath,@group\_id,@IndexType,@LeaseSec,0,0,1,GETDATE(),0,@BatchGuid,GETDATE(),@AppType)

SET @IndexId = SCOPE\_IDENTITY()

IF (@IndexId<=0)

SET @IndexPath=''

END

ELSE

BEGIN

--lock found levelnum record

UPDATE searchindex SET IsLocked=1,DateLeased=GETDATE(),BatchGUID=@BatchGuid,AppType=@AppType,LeaseSeconds=@LeaseSec,[Type]=@IndexType WHERE indexid=@IndexId and IsLocked=0

IF @@ROWCOUNT = 0

SET @IndexPath=''

END

IF ISNULL(@IndexPath,'')<>''

BEGIN

SELECT \* FROM searchindex(NOLOCK) WHERE indexid=@IndexId

END

END

CREATE PROCEDURE [dbo].[stp\_SearchIndex\_GetLevel3IndexBestCandidateAndLock]

@TotalDocCount INT,

@MaxIndexDocCount INT,

@GroupId INT,

@Type TINYINT,

@AppType TINYINT,

@MaxShards INT=4,

@BatchGuid UNIQUEIDENTIFIER

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)

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

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

OUTPUT inserted.indexid INTO @Tbl

WHERE indexid IN

(

SELECT TOP (@MaxShards) indexid FROM searchindex (NOLOCK)

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

ORDER BY DocCount DESC

)

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

AND IsLocked=0

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