**This is a recursive query view using a windows function.**

create view [dbo].[AgencyAncestry]

as

with agencies as

(

select

a.AgencyId

,a.Active

,aa.ParentId

from Atlas.dbo.Agency as a

left join Atlas.dbo.AgencyAssoc as aa

on aa.ChildId = a.AgencyID

and aa.RemovedDate is null

)

,agencyAncestryRecursive as

(

select

a.AgencyID

,a.AgencyID as AncestorAgencyId

,a.ParentId

,1 as AscendantId

from agencies as a

union all

select

aar.AgencyID

,a.AgencyId

,a.ParentId

,aar.AscendantId + 1

from agencyAncestryRecursive as aar

join agencies as a

on a.AgencyID = aar.ParentId

and a.Active = 1 -- ensure that any parent relationship refers to an active agency...

)

select

aar.AgencyID

,aar.AncestorAgencyId

,aar.AscendantId

,DescendantId = row\_number() over

(

partition by aar.AgencyId

order by aar.AscendantId desc

)

from agencyAncestryRecursive as aar;

**Used for entering partial text into a search criteria parameter SSRS report**

ALTER procedure [dbo].[reportGetMarketingGroup]

@SearchCriteria varchar(100) = null,

@ActiveOnly bit = 1

as

begin

set nocount on

if @SearchCriteria is not null

begin

set @SearchCriteria = ltrim(rtrim(@SearchCriteria))

while charindex(' ', @SearchCriteria) > 0 set @SearchCriteria = replace(@SearchCriteria, ' ', ' ')

if object\_id('tempdb.dbo.#Results') is not null drop table #Results

create table #Results (

SortOrder integer,

Value integer,

Label varchar(100))

insert into #Results

select 0 as SortOrder,

-re.RepID as Value,

re.FirstName + ' ' + re.LastName as Label

from Marketing.dbo.Representatives as re

where re.Active = isnull(@ActiveOnly, 1)

and ( re.FirstName + ' ' + re.LastName like '%' + @SearchCriteria + '%'

or re.LastName like '%' + @SearchCriteria + '%'

or re.FirstName like '%' + @SearchCriteria + '%')

union

select 1 as SortOrder,

-(at.TierID + 500) as Value,

at.Tier + ' Tier' as Label

from Atlas.dbo.AgencyTier as at

where at.Tier + ' Tier' like '%' + @SearchCriteria + '%'

or at.Tier like '%' + @SearchCriteria + '%'

union

select top 500

2 as SortOrder,

ag.AgencyID as Value,

'[' + ag.AgencyCode + '] - ' + ag.AgencyName as Label

from Atlas.dbo.Agency as ag

where ag.Active = isnull(@ActiveOnly, 1)

and ( ag.AgencyCode like '%' + @SearchCriteria + '%'

or ag.AgencyName like '%' + @SearchCriteria + '%')

if @@rowcount = 0

begin

if object\_id('tempdb.dbo.#Words') is not null drop table #Words

create table #Words (Word varchar(100))

create clustered index idx\_Word on #Words (Word)

insert into #Words

select '%' + sp.StringValue + '%'

from Common.dbo.Split(@SearchCriteria, ' ') as sp

where sp.StringValue != ''

insert into #Results

select 0 as SortOrder,

-re.RepID as Value,

re.FirstName + ' ' + re.LastName as Label

from Marketing.dbo.Representatives as re

inner join #Words as wo

on re.FirstName like wo.Word

or re.LastName like wo.Word

where re.Active = isnull(@ActiveOnly, 1)

union

select top 500

1 as SortOrder,

ag.AgencyID as Value,

'[' + ag.AgencyCode + '] - ' + ag.AgencyName as Label

from Atlas.dbo.Agency as ag

inner join #Words as wo

on ag.AgencyCode like wo.Word

or ag.AgencyName like wo.Word

where ag.Active = isnull(@ActiveOnly, 1)

end

select r.Value,

r.Label

from #Results as r

order by r.SortOrder,

r.Label

if object\_id('tempdb.dbo.#Results') is not null drop table #Results

if object\_id('tempdb.dbo.#Words') is not null drop table #Words

end

else

begin

begin with

WithSortOrder as (

select 0 as SortOrder,

-re.RepID as Value,

re.FirstName + ' ' + re.LastName as Label

from Marketing.dbo.Representatives as re

where re.Active = 1

union

select 1 as SortOrder,

-(at.TierID + 500) as Value,

at.Tier + ' Tier' as Label

from Atlas.dbo.AgencyTier as at

union

select 1 as SortOrder,

-500 as Value,

'No Tier' as Label

-- union

--select 2 as SortOrder,

-- ag.AgencyID as Value,

-- '[' + ag.AgencyCode + '] - ' + ag.AgencyName as Label

--from Atlas.dbo.Agency as ag

--where ag.Active = 1

)

select wso.Value,

wso.Label

from WithSortOrder as wso

order by wso.SortOrder,

wso.Label

end

end

end

**UNIVERSAL SPLIT FUNCTION**

ALTER FUNCTION [dbo].[Split]

(

@String varchar(max)

,@Delimiter varchar

)

RETURNS @Results table

(

Ordinal int

,StringValue varchar(max)

)

as

begin

set @String = isnull(@String,'')

set @Delimiter = isnull(@Delimiter,'')

declare

@TempString varchar(max) = @String

,@Ordinal int = 0

,@CharIndex int = 0

set @CharIndex = charindex(@Delimiter, @TempString)

while @CharIndex != 0 begin

set @Ordinal += 1

insert @Results values

(

@Ordinal

,substring(@TempString, 0, @CharIndex)

)

set @TempString = substring(@TempString, @CharIndex + 1, len(@TempString) - @CharIndex)

set @CharIndex = charindex(@Delimiter, @TempString)

end

if @TempString != '' begin

set @Ordinal += 1

insert @Results values

(

@Ordinal

,@TempString

)

end

return

end

**MOM Written Premium %Change report**

ALTER proc [dbo].[GetCRMnewWrittenPremium%Change]

as begin

set nocount on;

declare @StartingDate as datetime = dbo.SpecialDate(dateadd(month, -12, dateadd(day, -1, cast(getdate() as date))), 'FOM'),

@EndingDate as datetime = dateadd(millisecond, -3, cast(cast(getdate() as date) as datetime))

begin with

MonthlyPremium as(

select

dbo.SpecialDate(p.BoundDate, 'FOM') as BoundDate,

sum(pva.TotalPremium) as Premium

from dbo.Policy as p

inner join ( select pv.PolicyID,

min(pv.PolicyVersionID) as PolicyVersionID

from dbo.PolicyVersion as pv

group by pv.PolicyID) as pvm

inner join dbo.PolicyVersionAttributes as pva

on pva.PolicyVersionID = pvm.PolicyVersionID

on pvm.PolicyID = p.PolicyID

where p.BoundDate

between @StartingDate

and @EndingDate

group by dbo.SpecialDate(p.BoundDate, 'FOM')

)

select mpc.BoundDate, mpc.Premium, mpl.Premium as LastMthPremium, ((mpc.Premium - mpl.Premium) / mpl.Premium) as '%Change'

from MonthlyPremium as mpc

left join MonthlyPremium mpl

on mpl.BoundDate = dateadd(month, -1, mpc.BoundDate)

order by 1

**30 day Aging report**

ALTER procedure [dbo].[reportPolicy30DayAging] 30

@Age int

as

begin

;with Balance

as

(

select at.PolicyID,

sum(at.Amount) as Amount

from AccountingTransaction as at

group by at.PolicyID

having sum(at.Amount) > 0

)

select distinct(p.PolicyID),

p.PolicyNumber,

dbo.GetPolicyInsureds(p.policyID) as Applicant,

pv.EffectiveDate,

a.AgencyCode as Agency,

b.Amount as Balance

from Policy as p

join Balance as b

on b.PolicyID = p.PolicyID

join Agency as a

on p.AgencyID = a.AgencyID

join PolicyVersion as pv

on pv.PolicyVersionID=P.PolicyVersionID

left join AccountingTransaction as at

on at.PolicyID = p.PolicyID

and at.AccountingTransactionTypeID IN (6, 15)

where pv.EffectiveDate < dateadd(day, -1 \* (@Age + 1), getdate())

and at.PolicyID is null

and p.Inforce = 1

end

**THIS IS GREAT OPTIMIZED CODE TO STUDY**

USE [Atlas]

GO

/\*\*\*\*\*\* Object: StoredProcedure [Workflow].[spGetQueueList] Script Date: 2/16/2017 10:45:03 AM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER procedure [Workflow].[spGetQueueList]

@UserID integer,

@TypeID integer,

@StatusID integer = null,

@QueueID integer = null,

@PolicyNumber varchar(25) = null,

@ClaimNumber varchar(20) = null,

@MyItemsOnly bit = 0,

@ConcernCodeID int = 0,

@FilterID int = 0,

@IncludeMyRoles bit = 0,

@SubjectList varchar(max) = null

as

begin

set nocount on

set @TypeID = iif(@TypeID > 0, @TypeID, null)

set @StatusID = iif(@StatusID > 0, @StatusID, null)

set @QueueID = iif(@QueueID > 0, @QueueID, null)

set @ConcernCodeID = iif(@ConcernCodeID > 0, @ConcernCodeID, null)

set @FilterID = iif(@FilterID = 41, @FilterID, null)

declare @PolicyID as integer = (select PolicyID from dbo.Policy where PolicyNumber = @PolicyNumber)

declare @ClaimID as integer = (select ClaimID from dbo.Claim where ClaimNumber = @ClaimNumber)

if object\_id('tempdb.dbo.#Queue') is not null drop table #Queue

create table #Queue (

QueueID integer,

TypeID integer,

StatusID integer,

FollowUpDate date,

FollowUpReminder bit,

CreatedBy integer,

CreatedDate datetime,

[Priority] integer,

AssignedTo integer,

AssignedToRoleID integer,

Age integer)

create clustered index idx\_QueueID on #Queue (QueueID)

create nonclustered index idx\_TypeID on #Queue (TypeID)

if object\_id('tempdb.dbo.#ConcernCodes') is not null drop table #ConcernCodes

create table #ConcernCodes (ConcernCodeID integer)

create clustered index idx\_ConcernCodeID on #ConcernCodes (ConcernCodeID)

if @ConcernCodeID is null and isnull(@SubjectList, '') = ''

begin

insert into #ConcernCodes

select ConcernCodeID

from Workflow.ConcernCodes

end

else

begin

insert into #ConcernCodes

select StringValue

from Common.dbo.Split(@ConcernCodeID, ',')

union

select StringValue

from Common.dbo.Split(@SubjectList, ',')

end

if @TypeID is null or @TypeID = 2 -- Alerts

begin

if isnull(@ClaimID, @PolicyID) is null

begin

insert into #Queue

select q.QueueID,

q.TypeID,

q.StatusID,

q.FollowUpDate,

q.FollowUpReminder,

q.CreatedBy,

q.CreatedDate,

q.[Priority],

n.ReceiverID as AssignedTo,

null as AssignedToRoleID,

datediff(day, q.CreatedDate, getdate()) as Age

from [Workflow].[Queue] as q with (readpast)

inner join [Workflow].[Notifications] as n with (readpast)

inner join [Workflow].[Queue] as qa with (readpast)

on qa.QueueID = n.AssociatedQueueID

and qa.StatusID = 1

on n.QueueID = q.QueueID

where n.ReceiverID = @UserID

and n.Deleted = 0

and q.TypeID = 2

and q.FollowUpDate > dateadd(day, -30, getdate())

end

else

begin

insert into #Queue

select q.QueueID,

q.TypeID,

q.StatusID,

q.FollowUpDate,

q.FollowUpReminder,

q.CreatedBy,

q.CreatedDate,

q.[Priority],

n.ReceiverID as AssignedTo,

null as AssignedToRoleID,

datediff(day, q.CreatedDate, getdate()) as Age

from [Workflow].[Queue] as q with (readpast)

inner join [Workflow].[Notifications] as n with (readpast)

inner join [Workflow].[Referrals] as r with (readpast)

on r.QueueID = n.AssociatedQueueID

on n.QueueID = q.QueueID

where n.ReceiverID = @UserID

and n.Deleted = 0

and q.TypeID = 2

and ( r.PolicyID = @PolicyID

or r.ClaimID = @ClaimID)

end

end

if @QueueID is not null

begin

insert into #Queue

select q.QueueID,

q.TypeID,

q.StatusID,

q.FollowUpDate,

q.FollowUpReminder,

q.CreatedBy,

q.CreatedDate,

q.[Priority],

n.ReceiverID as AssignedTo,

null as AssignedToRoleID,

datediff(day, q.CreatedDate, getdate()) as Age

from [Workflow].[Queue] as q with (readpast)

inner join [Workflow].[Notifications] as n with (readpast)

on n.QueueID = q.QueueID

where q.QueueID = @QueueID

and q.TypeID = 2

and nullif(@TypeID, q.TypeID) is null

and nullif(@StatusID, q.StatusID) is null

and n.ReceiverID = @UserID

and n.Deleted = 0

union

select q.QueueID,

q.TypeID,

q.StatusID,

q.FollowUpDate,

q.FollowUpReminder,

q.CreatedBy,

q.CreatedDate,

q.[Priority],

a.AssignedTo,

a.AssignedToRoleID,

datediff(day, q.CreatedDate, getdate()) as Age

from [Workflow].[Queue] as q with (readpast)

inner join [Workflow].[Referrals] as r with (readpast)

on r.QueueID = q.QueueID

inner join [Workflow].[Assignments] as a with (readpast)

on a.QueueID = q.QueueID

and a.Active = 1

inner join #ConcernCodes as cc

on cc.ConcernCodeID = r.ConcernCodeID

where q.QueueID = @QueueID

and nullif(@TypeID, q.TypeID) is null

and nullif(@StatusID, q.StatusID) is null

and nullif(@PolicyID, r.PolicyID) is null

and nullif(@ClaimID, r.ClaimID) is null

--and nullif(@ConcernCodeID, r.ConcernCodeID) is null

and ( @FilterID is null

or ( r.XMLMapping is not null

and convert(xml, r.XMLMapping).value('(//Object[@Name="ReviewStatus"]/@Value)[1]','varchar(128)') is null))

union

select q.QueueID,

q.TypeID,

q.StatusID,

q.FollowUpDate,

q.FollowUpReminder,

q.CreatedBy,

q.CreatedDate,

q.[Priority],

a.AssignedTo,

a.AssignedToRoleID,

datediff(day, q.CreatedDate, getdate()) as Age

from [Workflow].[Queue] as q with (readpast)

inner join [Workflow].[Approval] as ap with (readpast)

on ap.QueueID = q.QueueID

inner join [Workflow].[Assignments] as a with (readpast)

on a.QueueID = q.QueueID

and a.Active = 1

inner join #ConcernCodes as cc

on cc.ConcernCodeID = ap.ConcernCodeID

where q.QueueID = @QueueID

and q.TypeID = 5

and nullif(@TypeID, q.TypeID) is null

and nullif(@StatusID, q.StatusID) is null

--and nullif(@ConcernCodeID, ap.ConcernCodeID) is null

end

else

begin

if object\_id('tempdb.dbo.#Assignments') is not null drop table #Assignments

create table #Assignments (

QueueID integer,

AssignedTo integer,

AssignedToRoleID integer)

if isnull(@MyItemsOnly, 0) = 0

begin

insert into #Assignments

select a.QueueID,

a.AssignedTo,

a.AssignedToRoleID

from [Workflow].[Assignments] as a

inner join [Workflow].[Queue] as q

on q.QueueID = a.QueueID

where a.Active = 1

and nullif(@TypeID, q.TypeID) is null

and nullif(@StatusID, q.StatusID) is null

end

else

begin

insert into #Assignments

select a.QueueID,

a.AssignedTo,

a.AssignedToRoleID

from [Workflow].[Assignments] as a with (readpast)

inner join [Workflow].[Queue] as q with (readpast)

on q.QueueID = a.QueueID

where a.Active = 1

and a.AssignedTo = @UserID

and nullif(@TypeID, q.TypeID) is null

and nullif(@StatusID, q.StatusID) is null

/\* Anticiapte a request to... \*/

union

select a.QueueID,

a.AssignedTo,

a.AssignedToRoleID

from [Workflow].[Assignments] as a with (readpast)

inner join [Workflow].[Queue] as q with (readpast)

on q.QueueID = a.QueueID

inner join dbo.Roles as r

inner join dbo.UserRoles as ur

inner join dbo.SystemUsers as su

on su.SystemUserID = ur.SystemUserID

on ur.RoleID = r.RoleID

on r.RoleID = a.AssignedToRoleID

where ur.SystemUserID = @UserID

and @IncludeMyRoles = 1

and a.Active = 1

and su.Active = 1

and ur.Active = 1

and nullif(@TypeID, q.TypeID) is null

and nullif(@StatusID, q.StatusID) is null

/\* ...comment the section out. \*/

end

create clustered index idx\_QueueID on #Assignments (QueueID)

if @FilterID = 41

begin

insert into #Queue

select top 2000

q.QueueID,

q.TypeID,

q.StatusID,

q.FollowUpDate,

q.FollowUpReminder,

q.CreatedBy,

q.CreatedDate,

q.[Priority],

a.AssignedTo,

a.AssignedToRoleID,

datediff(day, q.CreatedDate, getdate()) as Age

from [Workflow].[Queue] as q with (readpast)

inner join [Workflow].[Referrals] as r with (readpast)

on r.QueueID = q.QueueID

inner join #Assignments as a

on a.QueueID = q.QueueID

inner join #ConcernCodes as cc

on cc.ConcernCodeID = r.ConcernCodeID

where nullif(@ClaimID, r.ClaimID) is null

--and nullif(@ConcernCodeID, r.ConcernCodeID) is null

and ( r.XMLMapping is not null

and convert(xml, r.XMLMapping).value('(//Object[@Name="ReviewStatus"]/@Value)[1]','varchar(128)') is null)

end

else

begin

if @TypeID is null or @TypeID <> 5 -- Referrals

begin

insert into #Queue

select top 2000

q.QueueID,

q.TypeID,

q.StatusID,

q.FollowUpDate,

q.FollowUpReminder,

q.CreatedBy,

q.CreatedDate,

q.[Priority],

a.AssignedTo,

a.AssignedToRoleID,

datediff(day, q.CreatedDate, getdate()) as Age

from [Workflow].[Queue] as q with (readpast)

inner join [Workflow].[Referrals] as r with (readpast)

on r.QueueID = q.QueueID

inner join #Assignments as a

on a.QueueID = q.QueueID

inner join #ConcernCodes as cc

on cc.ConcernCodeID = r.ConcernCodeID

where nullif(@PolicyID, r.PolicyID) is null

and nullif(@ClaimID, r.ClaimID) is null

--and nullif(@ConcernCodeID, r.ConcernCodeID) is null

end

if @TypeID is null or @TypeID = 5 -- Approvals

begin

insert into #Queue

select top 2000

q.QueueID,

q.TypeID,

q.StatusID,

q.FollowUpDate,

q.FollowUpReminder,

q.CreatedBy,

q.CreatedDate,

q.[Priority],

a.AssignedTo,

a.AssignedToRoleID,

datediff(day, q.CreatedDate, getdate()) as Age

from [Workflow].[Queue] as q with (readpast)

inner join [Workflow].[Approval] as ap with (readpast)

on ap.QueueID = q.QueueID

inner join #Assignments as a

on a.QueueID = q.QueueID

inner join #ConcernCodes as cc

on cc.ConcernCodeID = ap.ConcernCodeID

where q.TypeID = 5

--and nullif(@ConcernCodeID, ap.ConcernCodeID) is null

and isnull(@PolicyID, @ClaimID) is null

end

end

end

begin with

WFAlerts as (

select q.QueueID,

q.TypeID,

t.[Description] as WorkFlowTypeDescription,

q.StatusID,

s.[Description] as WorkFlowStatusDescription,

q.FollowUpDate,

q.FollowUpReminder,

q.CreatedBy,

isnull(suc.FirstName + ' ' + suc.LastName, lower(suc.UserName)) as CreatedByDesc,

q.CreatedDate,

n.[Subject],

cast('' as varchar(25)) as PolicyNumber,

cast('' as varchar(20)) as ClaimNumber,

cast('' as varchar(100)) as Catastrophe,

q.AssignedTo,

isnull(sur.FirstName + ' ' + sur.LastName, lower(sur.UserName)) as AssignedToDesc,

q.[Priority],

cast(null as varchar(max)) as ConcernMemo,

cast(null as varchar(max)) as Resolution,

cast(null as varchar(max)) as ResolutionMemo,

cast(0 as int) as ConcernCodeID,

q.Age,

cast(0 as int) as ExaminerID,

cast('' as varchar(100)) as ExaminerDesc,

cast(0 as int) as AdjusterID,

cast('' as varchar(100)) as AdjusterDesc,

cast('' as varchar(100)) as BusinessName,

lt.[Description] as LossType,

c.ReportedDate,

c.LossDate,

datediff(day, c.ReportedDate, getdate()) as ClaimAge

from #Queue as q

left join [Workflow].[Referrals] as r with (readpast)

inner join dbo.Claim as c with (readpast)

inner join dbo.LossType as lt with (readpast)

on c.LossTypeID = lt.LossTypeID

on c.ClaimID = r.ClaimID

on r.QueueID = q.QueueID

inner join [Workflow].[Notifications] as n with (readpast)

on n.QueueID = q.QueueID

inner join dbo.SystemUsers as sur with (readpast)

on sur.SystemUserID = q.AssignedTo

inner join [Workflow].[Type] as t with (readpast)

on q.TypeID = t.TypeID

inner join [Workflow].[Status] as s with (readpast)

on q.StatusID = s.StatusID

inner join dbo.SystemUsers as suc with (readpast)

on suc.SystemUserID = q.CreatedBy

where q.TypeID = 2

),

WFReferrals as (

select q.QueueID,

q.TypeID,

t.[Description] as WorkFlowTypeDescription,

q.StatusID,

s.[Description] as WorkFlowStatusDescription,

q.FollowUpDate,

q.FollowUpReminder,

q.CreatedBy,

isnull(suc.FirstName + ' ' + suc.LastName, lower(suc.UserName)) as CreatedByDesc,

q.CreatedDate,

cc.[Description] + char(13) + char(10) + 'Insured(s): ' + dbo.GetPolicyInsureds(r.policyID) as [Subject],

p.PolicyNumber,

c.ClaimNumber,

isnull(ca.[Description], '') as Catastrophe,

isnull(q.AssignedTo, 0) as AssignedTo,

case when q.AssignedTo is null then ro.[Description] else isnull(sua.FirstName + ' ' + sua.LastName, lower(sua.UserName)) end as AssignedToDesc,

q.[Priority],

r.ConcernMemo,

rc.[Description] as Resolution,

r.ResolutionMemo,

r.ConcernCodeID,

q.Age,

cast(0 as int) as ExaminerID,

cast('' as varchar(100)) as ExaminerDesc,

cast(0 as int) as AdjusterID,

cast('' as varchar(100)) as AdjusterDesc,

cast('' as varchar(100)) as BusinessName,

lt.[Description] as LossType,

c.ReportedDate,

c.LossDate,

datediff(day, c.ReportedDate, getdate()) as ClaimAge

from #Queue as q

inner join [Workflow].[Referrals] as r with (readpast)

inner join dbo.[Policy] as p with (readpast)

on p.PolicyID = r.PolicyID

left join dbo.Claim as c with (readpast)

inner join LossType as lt with (readpast)

on c.LossTypeID = lt.LossTypeID

left join dbo.Catastrophe as ca with (readpast)

on ca.CatastropheID = c.CatastropheID

on c.ClaimID = r.ClaimID

on r.QueueID = q.QueueID

inner join [Workflow].[ConcernCodes] as cc with (readpast)

on cc.ConcernCodeID = r.ConcernCodeID

left join [Workflow].[ResolutionCodes] as rc with (readpast)

on rc.ResolutionCodeID = r.ResolutionCodeID

inner join [Workflow].[Type] as t

on q.TypeID = t.TypeID

inner join [Workflow].[Status] as s

on q.StatusID = s.StatusID

inner join dbo.SystemUsers as suc

on suc.SystemUserID = q.CreatedBy

left join dbo.SystemUsers as sua

on sua.SystemUserID = q.AssignedTo

left join dbo.Roles as ro

on ro.Roleid = q.AssignedToRoleID

where q.TypeID not in (2, 5)

),

WFApprovals as (

select q.QueueID,

q.TypeID,

t.[Description] as WorkFlowTypeDescription,

q.StatusID,

s.[Description] as WorkFlowStatusDescription,

q.FollowUpDate,

q.FollowUpReminder,

q.CreatedBy,

isnull(suc.FirstName + ' ' + suc.LastName, lower(suc.UserName)) as CreatedByDesc,

q.CreatedDate,

cc.[Description] as [Subject],

isnull(p.POlicyNumber, '') as PolicyNumber,

cast('' as varchar(20)) as ClaimNumber,

cast('' as varchar(100)) as Catastrophe,

isnull(q.AssignedTo, 0) as AssignedTo,

case when q.AssignedTo is null then ro.[Description] else isnull(sua.FirstName + ' ' + sua.LastName, lower(sua.UserName)) end as AssignedToDesc,

q.[Priority],

wa.ConcernMemo,

rc.[Description] as Resolution,

wa.ResolutionMemo,

wa.ConcernCodeID,

q.Age,

cast(0 as int) as ExaminerID,

cast('' as varchar(100)) as ExaminerDesc,

cast(0 as int) as AdjusterID,

cast('' as varchar(100)) as AdjusterDesc,

cast('' as varchar(100)) as BusinessName,

cast('' as varchar(50)) as LossType,

cast(null as datetime) as ReportedDate,

cast(null as datetime) as LossDate,

cast(null as integer) as ClaimAge

from #Queue as q

inner join [Workflow].[Approval] as wa with (readpast)

left join dbo.ContestEntry as ce with (readpast)

inner join dbo.PolicyVersion as pv with (readpast)

inner join dbo.[Policy] as p with (readpast)

on p.PolicyID = pv.PolicyID

on pv.PolicyVersionID = ce.PolicyVersionID

on ce.ContestEntryID = wa.ContextID

and wa.ContextTypeID = 4

inner join [Workflow].[ConcernCodes] as cc with (readpast)

on cc.ConcernCodeID = wa.ConcernCodeID

left join [Workflow].[ResolutionCodes] as rc with (readpast)

on rc.ResolutionCodeID = wa.ResolutionCodeID

on wa.QueueID = q.QueueID

inner join [Workflow].[Type] as t with (readpast)

on q.TypeID = t.TypeID

inner join [Workflow].[Status] as s with (readpast)

on q.StatusID = s.StatusID

inner join dbo.SystemUsers as suc with (readpast)

on suc.SystemUserID = q.CreatedBy

left join dbo.SystemUsers as sua with (readpast)

on sua.SystemUserID = q.AssignedTo

left join dbo.Roles as ro with (readpast)

on ro.Roleid = q.AssignedToRoleID

where q.TypeID = 5

),

WFAttachments as (

select wfa.QueueID

from WFAlerts as wfa

inner join [Workflow].[Attachments] as a with (readpast)

on a.QueueID = wfa.QueueID

union

select wfr.QueueID

from WFReferrals as wfr

inner join [Workflow].[Attachments] as a with (readpast)

on a.QueueID = wfr.QueueID

union

select wfa.QueueID

from WFApprovals as wfa

inner join [Workflow].[Attachments] as a with (readpast)

on a.QueueID = wfa.QueueID

),

Assignments as (

select q.QueueID,

--cd.SystemUserID as ExaminerID,

--isnull(sux.FirstName + ' ' + sux.LastName,'') as ExaminerDesc,

--cd.AdjusterSystemUserID as AdjusterID,

sux.SystemUserID as ExaminerID,

isnull(sux.FirstName + ' ' + sux.LastName,'') as ExaminerDesc,

sua.SystemUserID as AdjusterID,

coalesce(sua.FirstName + ' ' + sua.LastName, co.FirstName + ' ' + co.LastName, '') as AdjusterDesc,

co.BusinessName,

--row\_number() over (partition by q.QueueID, cd.ClaimID order by cd.CreatedDate desc) as RowNumber

1 as RowNumber

from #Queue as q

inner join Workflow.Referrals as re with (readpast)

left join dbo.SystemUsers as sux with (readpast)

on sux.SystemUserID = dbo.GetExaminerByClaimID(re.ClaimID)

left join dbo.SystemUsers as sua with (readpast)

on sua.SystemUserID = dbo.GetAdjusterByClaimID(re.ClaimID)

left join dbo.Contact as co with (readpast)

on co.ContactID = dbo.GetAdjustingCompanyByClaimID(re.ClaimID)

--inner join dbo.ClaimDetailsAndAssignments as cd with (readpast)

-- left join dbo.SystemUsers as sux with (readpast)

-- on sux.SystemUserID = cd.SystemUserID

-- left join dbo.SystemUsers as sua with (readpast)

-- on sua.SystemUserID = cd.AdjusterSystemUserID

-- left join dbo.Contact as co with (readpast)

-- on co.ContactID = cd.AdjustingCompanyContactID

-- on cd.ClaimID = re.ClaimID

on re.QueueID = q.QueueID

where q.TypeID not in (2, 5)

--and cd.ClaimDetailTypeID in (1, 2)

--and cd.Active = 1

)

select wf.QueueID,

wf.TypeID,

wf.WorkFlowTypeDescription,

wf.StatusID,

wf.WorkFlowStatusDescription,

wf.FollowUpDate,

wf.FollowUpReminder,

wf.CreatedBy,

wf.CreatedByDesc,

wf.CreatedDate,

wf.[Subject],

wf.PolicyNumber,

wf.ClaimNumber,

wf.Catastrophe,

wf.AssignedTo,

wf.AssignedToDesc,

wf.[Priority],

cast(iif([at].QueueID is null, 0, 1) as bit) Attachment,

wf.ConcernMemo,

wf.Resolution,

wf.ResolutionMemo,

wf.ConcernCodeID,

wf.Age,

wf.ExaminerID,

wf.ExaminerDesc,

wf.AdjusterID,

wf.AdjusterDesc,

wf.BusinessName as UAC,

wf.LossType,

wf.ReportedDate,

wf.LossDate,

wf.ClaimAge

from WFAlerts as wf

left join WFAttachments as [at] with (readpast)

on [at].QueueID = wf.QueueID

union all

select wf.QueueID,

wf.TypeID,

wf.WorkFlowTypeDescription,

wf.StatusID,

wf.WorkFlowStatusDescription,

wf.FollowUpDate,

wf.FollowUpReminder,

wf.CreatedBy,

wf.CreatedByDesc,

wf.CreatedDate,

wf.[Subject],

wf.PolicyNumber,

wf.ClaimNumber,

wf.Catastrophe,

wf.AssignedTo,

wf.AssignedToDesc,

wf.[Priority],

cast(iif([at].QueueID is null, 0, 1) as bit) Attachment,

wf.ConcernMemo,

wf.Resolution,

wf.ResolutionMemo,

wf.ConcernCodeID,

wf.Age,

isnull(a.ExaminerID, wf.ExaminerID) as ExaminerID,

isnull(a.ExaminerDesc, wf.ExaminerDesc) as ExaminerDesc,

isnull(a.AdjusterID, wf.AdjusterID) as AdjusterID,

isnull(a.AdjusterDesc, wf.AdjusterDesc) as AdjusterDesc,

a.BusinessName as UAC,

wf.LossType,

wf.ReportedDate,

wf.LossDate,

wf.ClaimAge

from WFReferrals as wf with (readpast)

left join WFAttachments as [at] with (readpast)

on [at].QueueID = wf.QueueID

left join Assignments as a

on a.QueueID = wf.QueueID

where isnull(a.RowNumber, 1) = 1

union all

select wf.QueueID,

wf.TypeID,

wf.WorkFlowTypeDescription,

wf.StatusID,

wf.WorkFlowStatusDescription,

wf.FollowUpDate,

wf.FollowUpReminder,

wf.CreatedBy,

wf.CreatedByDesc,

wf.CreatedDate,

wf.[Subject],

wf.PolicyNumber,

wf.ClaimNumber,

wf.Catastrophe,

wf.AssignedTo,

wf.AssignedToDesc,

wf.[Priority],

cast(iif([at].QueueID is null, 0, 1) as bit) Attachment,

wf.ConcernMemo,

wf.Resolution,

wf.ResolutionMemo,

wf.ConcernCodeID,

wf.Age,

wf.ExaminerID,

wf.ExaminerDesc,

wf.AdjusterID,

wf.AdjusterDesc,

wf.BusinessName as UAC,

wf.LossType,

wf.ReportedDate,

wf.LossDate,

wf.ClaimAge

from WFApprovals as wf

left join WFAttachments as [at] with (readpast)

on [at].QueueID = wf.QueueID

order by FollowUpDate,

QueueID

end

end

ALTER proc [dbo].[GetMonthEnds]

(

@max int = null

)

as

begin

set nocount on;

set @max = isnull(@max, 60)

declare

@start date = getdate()

,@tempDate date

,@count int = 0

declare @dates table

(

Description varchar(20)

,Value date

)

set @start = dateadd(month,1,@start)

set @start = dateadd(day,-(datepart(day,@start)-1),@start)

while @count < @max begin

select

@start = dateadd(month,-1,@start)

,@count = @count + 1

,@tempDate = dateadd(day,-1,@start)

insert @dates values(datename(month,@tempDate) + ' ' + datename(year, @tempDate), @tempDate)

end

select

\*

from @dates

order by Value desc

end

/\* This SP ranks the top 25 Agencies/Reps by 4 different ranking types (categories)\*/

**THIS IS GREAT RANK OVER CODE TO STUDY**

ALTER proc [dbo].[ReportGetAgencyYoYGrowthTop25\_DOMO\_NEW]

as begin

set nocount on

declare @ThisYear datetime = getdate()

declare @LastYear datetime = dateadd(year, -1, @ThisYear)

if object\_id('tempdb.dbo.#LastYear') is not null drop table #LastYear

select pah.AgencyID as AgencyID,

count(\*) as Policies,

sum(pva.TotalPremium) as Premium,

month(p.CreatedDate) as [Month]

into #LastYear

from dbo.[Policy] as p

inner join ( select pv.PolicyID,

min(pv.PolicyVersionID) as PolicyVersionID

from dbo.PolicyVersion as pv

group by pv.PolicyID) as pvm

inner join dbo.PolicyVersionAttributes as pva

on pva.PolicyVersionID = pvm.PolicyVersionID

on pvm.PolicyID = p.PolicyID

inner join ( select pa.PolicyID,

min(pa.PolicyAgencyHistoryID) as PolicyAgencyHistoryID

from dbo.PolicyAgencyHistory as pa

group by pa.PolicyID) as pahm

inner join dbo.PolicyAgencyHistory as pah

on pah.PolicyAgencyHistoryID = pahm.PolicyAgencyHistoryID

on pahm.PolicyID = p.PolicyID

where p.CreatedDate

between dateadd(day, -datepart(dayofyear, @LastYear) + 1, cast(@LastYear as date))

and @LastYear

and p.ProductID != 2

group by pah.AgencyID, month(p.CreatedDate)

create clustered index idx\_LastYear\_AgencyID\_Month on #LastYear (AgencyID, [Month])

--select \* from #LastYear

if object\_id('tempdb.dbo.#ThisYear') is not null drop table #ThisYear

select pah.AgencyID as AgencyID,

count(\*) as Policies,

sum(pva.TotalPremium) as Premium,

month(p.CreatedDate) as [Month]

into #ThisYear

from dbo.[Policy] as p

inner join ( select pv.PolicyID,

min(pv.PolicyVersionID) as PolicyVersionID

from dbo.PolicyVersion as pv

group by pv.PolicyID) as pvm

inner join dbo.PolicyVersionAttributes as pva

on pva.PolicyVersionID = pvm.PolicyVersionID

on pvm.PolicyID = p.PolicyID

inner join ( select pa.PolicyID,

min(pa.PolicyAgencyHistoryID) as PolicyAgencyHistoryID

from dbo.PolicyAgencyHistory as pa

group by pa.PolicyID) as pahm

inner join dbo.PolicyAgencyHistory as pah

on pah.PolicyAgencyHistoryID = pahm.PolicyAgencyHistoryID

on pahm.PolicyID = p.PolicyID

where p.CreatedDate

between dateadd(day, -datepart(dayofyear, @ThisYear) + 1, cast(@ThisYear as date))

and @ThisYear

and p.ProductID != 2

group by pah.AgencyID, month(p.CreatedDate)

create clustered index idx\_ThisYear\_AgencyID\_Month on #ThisYear (AgencyID, [Month])

begin with

ThisYear as (

select AgencyID,

sum(Policies) as Policies,

sum(Premium) as Premium

from #ThisYear

group by AgencyID

),

LastYear as (

select AgencyID,

sum(Policies) as Policies,

sum(Premium) as Premium

from #LastYear

group by AgencyID

),

Ranked as (

select 'Growth By Submitted Premium YTD' as RankType,

0 as [Month],

isnull(ly.Premium, 0.0) as PriorYearPremium,

isnull(ty.Premium, 0.0) as CurrentYearPremium,

isnull(ty.Premium, 0) - isnull(ly.Premium, 0) as PremiumGrowth,

isnull(ly.Policies, 0) as PriorYearPolicies,

isnull(ty.Policies, 0) as CurrentYearPolicies,

isnull(ty.Policies, 0) - isnull(ly.Policies, 0) as PolicyGrowth,

rank() over (order by isnull(ty.Premium, 0) - isnull(ly.Premium, 0) desc) as Ranking,

isnull(ty.AgencyID, ly.AgencyID) as AgencyID

from ThisYear as ty

full

outer join LastYear as ly

on ly.AgencyID = ty.AgencyID

union

select 'Growth By Submitted Policies YTD' as RankType,

0 as [Month],

isnull(ly.Premium, 0.0) as PriorYearPremium,

isnull(ty.Premium, 0.0) as CurrentYearPremium,

isnull(ty.Premium, 0) - isnull(ly.Premium, 0) as PremiumGrowth,

isnull(ly.Policies, 0) as PriorYearPolicies,

isnull(ty.Policies, 0) as CurrentYearPolicies,

isnull(ty.Policies, 0) - isnull(ly.Policies, 0) as PolicyGrowth,

rank() over (order by isnull(ty.Policies, 0) - isnull(ly.Policies, 0) desc) as Ranking,

isnull(ty.AgencyID, ly.AgencyID) as AgencyID

from ThisYear as ty

full

outer join LastYear as ly

on ly.AgencyID = ty.AgencyID

union

select 'Growth By Submitted Premium MTD' as RankType,

isnull(ty.[Month], ly.[Month]) as [Month],

isnull(ly.Premium, 0.0) as PriorYearPremium,

isnull(ty.Premium, 0.0) as CurrentYearPremium,

isnull(ty.Premium, 0) - isnull(ly.Premium, 0) as PremiumGrowth,

isnull(ly.Policies, 0) as PriorYearPolicies,

isnull(ty.Policies, 0) as CurrentYearPolicies,

isnull(ty.Policies, 0) - isnull(ly.Policies, 0) as PolicyGrowth,

rank() over (partition by isnull(ty.[Month], ly.[Month]) order by isnull(ty.Premium, 0) - isnull(ly.Premium, 0) desc) as Ranking,

isnull(ty.AgencyID, ly.AgencyID) as AgencyID

from #ThisYear as ty

full

outer join #LastYear as ly

on ly.AgencyID = ty.AgencyID

and ly.[Month] = ty.[Month]

union

select 'Growth By Submitted Policies MTD' as RankType,

isnull(ty.[Month], ly.[Month]) as [Month],

isnull(ly.Premium, 0.0) as PriorYearPremium,

isnull(ty.Premium, 0.0) as CurrentYearPremium,

isnull(ty.Premium, 0) - isnull(ly.Premium, 0) as PremiumGrowth,

isnull(ly.Policies, 0) as PriorYearPolicies,

isnull(ty.Policies, 0) as CurrentYearPolicies,

isnull(ty.Policies, 0) - isnull(ly.Policies, 0) as PolicyGrowth,

rank() over (partition by isnull(ty.[Month], ly.[Month]) order by isnull(ty.Policies, 0) - isnull(ly.Policies, 0) desc) as Ranking,

isnull(ty.AgencyID, ly.AgencyID) as AgencyID

from #ThisYear as ty

full

outer join #LastYear as ly

on ly.AgencyID = ty.AgencyID

and ly.[Month] = ty.[Month]

)

select ra.RankType,

year(@ThisYear) as [Year],

case ra.[Month]

when 1 then 'Jan'

when 2 then 'Feb'

when 3 then 'Mar'

when 4 then 'Apr'

when 5 then 'May'

when 6 then 'Jun'

when 7 then 'Jul'

when 8 then 'Aug'

when 9 then 'Sep'

when 10 then 'Oct'

when 11 then 'Nov'

when 12 then 'Dec'

end as [Month],

--ra.[Month],

concat(r.FirstName, ' ', r.LastName) as RepName,

ra.PriorYearPremium,

ra.CurrentYearPremium,

ra.PremiumGrowth,

ra.PriorYearPolicies,

ra.CurrentYearPolicies,

ra.PolicyGrowth,

ra.Ranking,

concat(ag.AgencyName , ' - ' , ag.AgencyCode) as AgencyName,

ag.LicenseState as [State]

from Ranked as ra

inner join dbo.Agency as ag

inner join [Marketing].[dbo].[AgencyReps] as ar

inner join [Marketing].[dbo].[Representatives] as r

on ar.RepID = r.RepID

on ar.AgencyID = ag.AgencyID

on ag.AgencyID = ra.AgencyID

where Ranking <= 25

order by RankType,

[Month],

Ranking

end

end

/\* THIS SP DAISY CHAINS THE CLAIMS DATA IN THE FIRST WITH BLOCK THEN DAISY CHAINS THE INFORCE DATA IN THE SECOND WITH BLOCK AND THEN JOINS THE TWO TOGETHER \*/

USE [Atlas]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[reportYearOverClaimsTrend] Script Date: 8/14/2015 9:05:45 AM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

--[dbo].[reportYearOverClaimsTrend] '20150101', '20150804', '20150804', 1, 2,3

CREATE procedure [dbo].[reportYearOverClaimsTrend]

@StartingDate as datetime = null,

@EndingDate as datetime = null,

@AsOfDate as datetime = null,

@CompanyID as integer = null,

@LookBackPeriod as integer = null,

@ReportTypeID as integer = null

as

begin

set nocount on

if @EndingDate is null set @EndingDate = cast(dateadd(day, -day(getdate()), getdate()) as date)

if @StartingDate is null or @StartingDate > @EndingDate set @StartingDate = dbo.SpecialDate(@EndingDate, 'FOY')

if @AsOfDate is null set @AsOfDate = cast(dateadd(day, -1, getdate()) as date)

if @CompanyID not in (1, 2) set @CompanyID = null

if @ReportTypeID is null set @ReportTypeID = 1 -- 'Reported during the Period'

declare @OccurredOnly as bit = iif(@ReportTypeID = 2, 1, 0) -- 'Reported and Occurring during the Period'

if object\_id('tempdb.dbo.#Query') is not null drop table #Query

**/\* This is an example of a recursive CTE. Google “query hints” to view how to set the MAX recursion \*/**

begin with

DateRange as (

select 1 as DateRangeID,

@StartingDate as StartingDate,

dateadd(millisecond, -3, dateadd(day, 1, @EndingDate)) as EndingDate,

dateadd(millisecond, -3, dateadd(day, 1, @AsOfDate)) as AsOfDate

**/\* I need to verify with Don why he uses Millisecond \*/**

union all

select dr.DateRangeID + 1 as DateRangeID,

dateadd(year, -1, dr.StartingDate) as StartingDate,

dateadd(year, -1, dr.EndingDate) as EndingDate,

dateadd(year, -1, dr.AsOfDate) as AsOfDate

from DateRange as dr

where dr.DateRangeID <= @LookBackPeriod

),

Summary as (

select ct.ClaimID,

lt.[Description] as LossType,

st.StateCode as [State],

dr.DateRangeID,

year(c.LossDate) as AccidentYear,

cast(c.ReportedDate as date) as DateReported,

sum(case when ct.ClaimAccountingTransactionTypeID in (5, 6) and ct.DraftNumber <> 'QUEUED' then ct.Total else 0.00 end) as Losses,

sum(case when ct.ClaimAccountingTransactionTypeID < 5 or ct.DraftNumber = 'QUEUED' then ct.Total else 0.00 end) as Outstanding,

sum(case when ct.ClaimAccountingTransactionTypeID in (7, 8) then Amount else 0.00 end) as LAE,

cast(max(case when ct.ClaimAccountingTransactionTypeID < 5 or ct.DraftNumber = 'QUEUED' then ct.CreatedDate else null end) as date) as Closed

from Atlas.dbo.ClaimTransactions as ct

inner join Atlas.dbo.Claim as c

inner join dbo.LossType as lt

on lt.LossTypeID = c.LossTypeID

inner join DateRange as dr

on iif(@ReportTypeID = 3, c.LossDate, c.ReportedDate)

/\* This is called an Immediate IF. New in 2012 \*/

between dr.StartingDate

and dr.EndingDate

inner join Atlas.dbo.PolicyVersion as pv

inner join Atlas.dbo.Policy as p

inner join Atlas.dbo.States as st

on st.StateID = p.StateID

on p.PolicyID = pv.PolicyID

and p.CompanyID = isnull(@CompanyID, p.CompanyID)

on pv.PolicyVersionID = c.PolicyVersionID

left join dbo.Catastrophe as ca

inner join dbo.GetBODCatastrophe() as bc

on bc.CatastropheID = ca.CatastropheID

on ca.CatastropheID = c.CatastropheID

on c.ClaimID = ct.ClaimID

where ct.CreatedDate <= dr.AsOfDate

and ct.BatchNum in (-41, -777, -778)

and ct.ClaimAccountingTransactionTypeID between 1 and 8

and ( @OccurredOnly = 0

or c.LossDate >= dr.StartingDate)

and bc.CatastropheID is null

group by ct.ClaimID,

lt.[Description],

st.StateCode,

dr.DateRangeID,

cast(c.ReportedDate as date),

year(c.LossDate)

),

ReserveByDate as (

select sm.ClaimID,

sm.DateRangeID,

cast(ct.CreatedDate as date) as ReserveDate,

sum(case when ct.ClaimAccountingTransactionTypeID < 5 or ct.DraftNumber = 'QUEUED' then ct.Total else 0.00 end) as Outstanding

from Summary as sm

inner join DateRange as dr

on dr.DateRangeID = sm.DateRangeID

inner join dbo.ClaimTransactions as ct

on ct.ClaimID = sm.ClaimID

where sm.Outstanding != 0

and ct.CreatedDate <= dr.AsOfDate

and ct.BatchNum in (-41, -777, -778)

and ct.ClaimAccountingTransactionTypeID between 1 and 8

group by sm.ClaimID,

sm.DateRangeID,

cast(ct.CreatedDate as date)

having sum(case when ct.ClaimAccountingTransactionTypeID < 5 or ct.DraftNumber = 'QUEUED' then ct.Total else 0.00 end) != 0

),

Reopened as (

select distinct

rbdl.ClaimID,

rbdl.DateRangeID

from ReserveByDate as rbdl

inner join ReserveByDate as rbdr

on rbdr.ClaimID = rbdl.ClaimID

and rbdr.ReserveDate <= rbdl.ReserveDate

group by rbdl.ClaimID,

rbdl.DateRangeID,

rbdl.ReserveDate

having sum(rbdr.Outstanding) = 0

),

ProofOfLoss as (

select distinct

sm.ClaimID,

sm.DateRangeID

from Summary as sm

inner join DateRange as dr

on dr.DateRangeID = sm.DateRangeID

where sm.Outstanding != 0

and exists ( select \*

from dbo.PrintedDocuments as pd

where pd.ClaimID = sm.ClaimID

and pd.ReportID = 1029

and pd.CreatedDate <= dr.AsOfDate)

),

RepresentedClaim as (

select c.ClaimID,

--'Represented' as Category

cast('With Rep.' as varchar(50)) as Category

from dbo.Claim as c

inner join dbo.PrintedDocuments as pd

on pd.ClaimID = c.ClaimID

and pd.ReportID in (1072, 1073)

where cast(pd.CreatedDate as date) <= @AsOfDate

union

select c.ClaimID,

--'Represented' as Category

cast('With Rep.' as varchar(50)) as Category

from dbo.Claim as c

inner join dbo.ClaimContactAssoc as cta

on cta.ClaimID = c.ClaimID

inner join dbo.Contact as co

on co.ContactID = cta.ContactID

and co.ContactTypeID in(15,16)

where cast(cta.AssociatedDate as date) <= @AsOfDate

--and ( cta.RemovedDate is null

-- or cast(cta.RemovedDate as date) > @AsOfDate)

),

FastTrack as (

select distinct

ClaimID,

'FastTrack' as Category

--'FastTrack-W/O Rep' as Category

from Claim.FastTrack as ft

where cast(ft.FastTrackedOn as date) <= @AsOfDate

--and ( ft.RemovedOn is null

-- or cast(ft.RemovedOn as date) > @AsOfDate)

)

select sm.ClaimID,

dr.DateRangeID,

cast(dr.EndingDate as date) as EndingDate,

convert(varchar(10), dr.StartingDate, 101) + ' - ' +

convert(varchar(10), dr.EndingDate, 101) + ' evaluated as of ' +

convert(varchar(10), dr.AsOfDate, 101) + '.' as Period,

sm.AccidentYear,

sm.[State],

sm.LossType,

-- coalesce(rc.Category, ft.Category, 'Regular') as Category,

--isnull(ft.Category, 'Regular') + isnull(rc.Category, '') as Category,

isnull(ft.Category, 'Regular') as Category,

--CASE

--WHEN sm.LossType IS NULL THEN 'Inforce'

--ELSE isnull(ft.Category, 'Regular') END as Category,

isnull(rc.Category, 'without Rep.') as Represented,

sm.Losses,

sm.Outstanding,

sm.LAE,

sm.DateReported,

sm.Closed,

iif(re.ClaimID is null, 0, 1) as Reopened,

iif(pol.ClaimID is null, 0, 1) as ProofOfLoss

into #Query

from Summary as sm

inner join DateRange as dr

on dr.DateRangeID = sm.DateRangeID

left join Reopened as re

on re.ClaimID = sm.ClaimID

and re.DateRangeID = sm.DateRangeID

left join ProofOfLoss as pol

on pol.ClaimID = sm.ClaimID

and pol.DateRangeID = sm.DateRangeID

left join RepresentedClaim as rc

on rc.ClaimID = sm.ClaimId

left join FastTrack as ft

on ft.ClaimID = sm.ClaimId

option (optimize for unknown)

end

if object\_id('tempdb.dbo.#InforceByDate') is not null drop table #InforceByDate

select dt.Period,

year(dt.AsOfDate) as AccidentYear,

st.StateCode as [State],

cast('' as varchar(50)) as LossType,

cast('' as varchar(50)) as Category,

cast('' as varchar(50)) as Represented,

count(\*) as [Count],

sum(pva.SubTotal) as Premium

into #InforceByDate

from ( select distinct

Period,

EndingDate

from #Query) as dt(Period, AsOfDate)

cross apply dbo.GetInforcePolicies(dt.AsOfDate) as ip

inner join dbo.Policy as p

inner join dbo.States as st

on st.StateID = p.StateID

on p.PolicyID = ip.PolicyID

and p.CompanyID = isnull(@CompanyID, p.CompanyID)

inner join dbo.PolicyVersionAttributes as pva

on pva.PolicyVersionID = ip.PolicyVersionID

group by dt.Period,

st.StateCode,

year(dt.AsOfDate)

begin with

ClosedByState as (

select q.Period,

q.[State],

q.AccidentYear,

q.LossType,

q.Category,

q.Represented,

count(\*) as [Count],

round(avg(cast(datediff(day, q.DateReported, q.Closed) as float)), 2) as AverageDaysToClose

from #Query as q

where q.Outstanding = 0

group by q.Period,

q.[State],

q.LossType,

q.Category,

q.Represented,

q.AccidentYear

),

IncurredByState as (

select q.Period,

q.[State],

q.LossType,

q.Category,

q.Represented,

q.AccidentYear,

count(\*) as [Count],

round(avg(q.Losses + q.Outstanding), 2) as AverageIncurred,

round(sum(q.Losses + q.Outstanding), 2) as TotalIncurred,

sum(q.Reopened) as Reopened,

sum(q.ProofOfLoss) as ProofOfLoss

from #Query as q

group by q.Period,

q.[State],

q.LossType,

q.Category,

q.Represented,

q.AccidentYear

),

PeriodState as (

select p.Period,

s.[State],

p.LossType,

p.Category,

p.Represented,

p.AccidentYear

from ( select Period,

AccidentYear,

LossType,

Category,

Represented

from IncurredByState

union

select Period,

AccidentYear,

LossType,

Category,

Represented

from #InforceByDate ) as p(Period, AccidentYear, LossType, Category, Represented),

( select [State]

from IncurredByState

union

select [State]

from #InforceByDate

) as s([State])

)

select ps.Period,

ps.[State],

ps.AccidentYear,

ps.LossType,

ps.Category,

ps.Represented,

isnull(ins.[Count], 0) as InforceCount,

isnull(ins.Premium, 0) as InforcePremium,

isnull(ibs.[Count], 0) as Reported,

isnull(ibs.AverageIncurred, 0) as AverageIncurred,

isnull(ibs.TotalIncurred, 0) as TotalIncurred,

isnull(cbs.[Count], 0) as Closed,

isnull(ibs.ProofOfLoss, 0) as ProofOfLoss,

isnull(cbs.AverageDaysToClose, 0) as AverageDaysToClose,

isnull(ibs.Reopened, 0) as Reopened

from PeriodState as ps

left join #InforceByDate as ins

on ins.Period = ps.Period

and ins.[State] = ps.[State]

and ins.AccidentYear = ps.AccidentYear

and ins.LossType = ps.LossType

and ins.Category = ps.Category

and ins.Represented = ps.Represented

left join IncurredByState as ibs

on ibs.Period = ps.Period

and ibs.[State] = ps.[State]

and ibs.AccidentYear = ps.AccidentYear

and ibs.LossType = ps.LossType

and ibs.Category = ps.Category

and ibs.Represented = ps.Represented

left join ClosedByState as cbs

on cbs.Period = ps.Period

and cbs.[State] = ps.[State]

and cbs.AccidentYear = ps.AccidentYear

and cbs.LossType = ps.LossType

and cbs.Category = ps.Category

and cbs.Represented = ps.Represented

where coalesce(ins.AccidentYear, ibs.AccidentYear, cbs.AccidentYear) is not null

--AND isnull(ibs.[Count], 0) >0

order by ps.[State],

ps.AccidentYear,

ps.LossType,

ps.Category,

ps.Represented

end

if object\_id('tempdb.dbo.#InforceByDate') is not null drop table #InforceByDate

if object\_id('tempdb.dbo.#Query') is not null drop table #Query

end

GO

ALTER procedure [dbo].[reportGetFieldDraftsAndClaimDetail]

--@AsOfDate as datetime = null

as

begin

set nocount on

if object\_id('tempdb.dbo.#Claimdetail') is not null drop table #Claimdetail

create table

#Claimdetail (

claimid integer,

PolicyVersionID integer,

policyid integer,

ClaimNumber varchar(100),

LossDate Date,

ReportedDate Date,

LossType varchar(100),

CatastropheCode varchar(100),

Catastrophe varchar(100),

CatDate Date,

Category varchar(100),

FastTrack varchar(100),

AdjustingCompany varchar(100)

)

INSERT INTO #Claimdetail

select c.claimid,

c.PolicyVersionID,

pv.policyid,

c.ClaimNumber,

CAST(c.LossDate AS DATE) AS LossDate,

CAST(c.ReportedDate AS DATE) AS ReportedDate,

lt.[Description] as LossType,

iif(cat.CatastropheCode is null, 'No', 'Yes') as CatastropheCode,

isnull(cat.Description,'Non-Cat') as Catastrophe,

CAST(cat.DateOccurred AS DATE) as CatDate,

Category = iif(exists ( select \*

from dbo.Claim as cl

inner join dbo.ClaimContactAssoc as cta

on cta.ClaimID = cl.ClaimID

and cta.RemovedBy is null

inner join dbo.Contact as co

on co.ContactID = cta.ContactID

and co.ContactTypeID in(15,16)

where cl.ClaimID = c.ClaimID), 'With Rep.', 'Without Rep.'),

FastTrack = iif(exists ( select \*

from Claim.FastTrack as ft

where ft.ClaimID = c.ClaimID), 'FastTrack', 'Regular'),

isnull(co.BusinessName, 'Unknown') as AdjustingCompany

from Claim as c

left join dbo.Contact as co

on co.ContactID = dbo.GetAdjustingCompanyByClaimID(c.ClaimID)

inner join PolicyVersion as pv

on pv.PolicyVersionID = c.PolicyVersionID

left join GetBODCatastrophe() as bod

inner join Catastrophe as cat

on cat.CatastropheID = bod.CatastropheID

on bod.CatastropheID = c.CatastropheID

inner join LossType as lt

on lt.LossTypeID = c.LossTypeID

where cast(c.createddate as date) < cast(getdate() as date)

if object\_id('tempdb.dbo.#FieldDraft') is not null drop table #FieldDraft

create table

#FieldDraft (

claimNumber varchar(100),

DraftNumber varchar(100),

Payee1 varchar(500),

Amount integer,

DraftStatus varchar(100),

CreatedDate Date,

CreatedBy varchar(100),

PrintedDate Date,

PrintedBy varchar(100),

ClaimsDraftPrintLocation varchar(100)

)

create clustered index idx\_FieldDraft\_claimNumber on #FieldDraft (claimNumber)

insert into #FieldDraft

select

c.claimnumber,

cd.DraftNumber,

cd.Payee1,

cd.Amount,

cds.Description as [DraftStatus],

cd.CreatedDate,

su.UserName as [CreatedBy],

cd.PrintedDate,

(select top 1 su3.UserName from ClaimDraftActivity cda

join systemusers su3 on su3.SystemUserID = cda.CreatedBy

where cda.ClaimDraftID = cd.ClaimDraftId and cda.ClaimDraftStatusId = 3

order by cda.ClaimDraftActivityID desc) as [PrintedBy],

case when (select top 1 su3.UserName from ClaimDraftActivity cda

join systemusers su3 on su3.SystemUserID = cda.CreatedBy

where cda.ClaimDraftID = cd.ClaimDraftId and cda.ClaimDraftStatusId = 3 and cda.CreatedBy = cd.CreatedBy

order by cda.ClaimDraftActivityID desc) is null then 'InternalDraft' else 'FieldDraft' end as ClaimsDraftPrintLocation

--into #results

from ClaimDrafts cd

join PropertyStatements ps on ps.PropertyStatementId = cd.PropertyStatementId

join claim c on c.claimid = ps.claimid

left join claim.FastTrack ft on ft.ClaimID = c.ClaimID and ft.RemovedOn is null

join ClaimDraftStatus cds on cds.ClaimDraftStatusID = cd.ClaimDraftStatusId

join systemusers su on su.SystemUserID = cd.CreatedBy

join UserRoles ur on ur.SystemUserID = su.SystemUserID

where ur.RoleID = 135

and cd.CreatedDate >= '1/1/2014' and datediff(year, cast(cd.PrintedDate as date), getdate()) = 0

AND cast(cd.PrintedDate as date) <= getdate()

and cd.DraftNumber <> ''

SELECT cd.claimid,

cd.PolicyVersionID,

cd.policyid,

cd.ClaimNumber,

cd.LossDate,

cd.ReportedDate,

cd.LossType,

cd.CatastropheCode,

cd.Catastrophe,

cd.CatDate,

cd.Category,

cd.FastTrack,

cd.AdjustingCompany,

fd.ClaimsDraftPrintLocation,

--COUNT(fd.DraftNumber) AS YTDFieldDrafts

CASE WHEN fd.ClaimsDraftPrintLocation = 'FieldDraft' THEN COUNT(fd.DraftNumber) ELSE 0 end AS YTDClaimsWFieldDrafts,

CASE WHEN fd.ClaimsDraftPrintLocation = 'InternalDraft' THEN COUNT(fd.DraftNumber) ELSE 0 end AS YTDClaimsWInternalDrafts

FROM #Claimdetail cd

left JOIN #FieldDraft fd

ON cd.ClaimNumber = fd.ClaimNumber

GROUP BY cd.claimid,

cd.PolicyVersionID,

cd.policyid,

cd.ClaimNumber,

cd.LossDate,

cd.ReportedDate,

cd.LossType,

cd.CatastropheCode,

cd.Catastrophe,

cd.CatDate,

cd.Category,

cd.FastTrack,

cd.AdjustingCompany,

fd.ClaimsDraftPrintLocation

end

--[dbo].[reportYearOverClaimsTrendByPerils] '20150101', '20150804', '20150804', 1, 2,3

ALTER procedure [dbo].[reportYearOverClaimsTrendByPerils]

@StartingDate as datetime = null,

@EndingDate as datetime = null,

@AsOfDate as datetime = null,

@CompanyID as integer = null,

@LookBackPeriod as integer = null,

@ReportTypeID as integer = null

as

begin

set nocount on

if @EndingDate is null set @EndingDate = cast(dateadd(day, -day(getdate()), getdate()) as date)

if @StartingDate is null or @StartingDate > @EndingDate set @StartingDate = dbo.SpecialDate(@EndingDate, 'FOY')

if @AsOfDate is null set @AsOfDate = cast(dateadd(day, -1, getdate()) as date)

if @CompanyID not in (1, 2) set @CompanyID = null

if @ReportTypeID is null set @ReportTypeID = 1 -- 'Reported during the Period'

declare @OccurredOnly as bit = iif(@ReportTypeID = 2, 1, 0) -- 'Reported and Occurring during the Period'

if object\_id('tempdb.dbo.#Query') is not null drop table #Query

begin with

DateRange as (

select 1 as DateRangeID,

@StartingDate as StartingDate,

dateadd(millisecond, -3, dateadd(day, 1, @EndingDate)) as EndingDate,

dateadd(millisecond, -3, dateadd(day, 1, @AsOfDate)) as AsOfDate

union all

select dr.DateRangeID + 1 as DateRangeID,

dateadd(year, -1, dr.StartingDate) as StartingDate,

dateadd(year, -1, dr.EndingDate) as EndingDate,

dateadd(year, -1, dr.AsOfDate) as AsOfDate

from DateRange as dr

where dr.DateRangeID <= @LookBackPeriod

),

Summary as (

select ct.ClaimID,

lt.[Description] as LossType,

st.StateCode as [State],

dr.DateRangeID,

year(c.LossDate) as AccidentYear,

cast(c.ReportedDate as date) as DateReported,

sum(case when ct.ClaimAccountingTransactionTypeID in (5, 6) and ct.DraftNumber <> 'QUEUED' then ct.Total else 0.00 end) as Losses,

sum(case when ct.ClaimAccountingTransactionTypeID < 5 or ct.DraftNumber = 'QUEUED' then ct.Total else 0.00 end) as Outstanding,

sum(case when ct.ClaimAccountingTransactionTypeID in (7, 8) then Amount else 0.00 end) as LAE,

cast(max(case when ct.ClaimAccountingTransactionTypeID < 5 or ct.DraftNumber = 'QUEUED' then ct.CreatedDate else null end) as date) as Closed

from Atlas.dbo.ClaimTransactions as ct

inner join Atlas.dbo.Claim as c

inner join dbo.LossType as lt

on lt.LossTypeID = c.LossTypeID

inner join DateRange as dr

on iif(@ReportTypeID = 3, c.LossDate, c.ReportedDate)

between dr.StartingDate

and dr.EndingDate

inner join Atlas.dbo.PolicyVersion as pv

inner join Atlas.dbo.Policy as p

inner join Atlas.dbo.States as st

on st.StateID = p.StateID

on p.PolicyID = pv.PolicyID

and p.CompanyID = isnull(@CompanyID, p.CompanyID)

on pv.PolicyVersionID = c.PolicyVersionID

left join dbo.Catastrophe as ca

inner join dbo.GetBODCatastrophe() as bc

on bc.CatastropheID = ca.CatastropheID

on ca.CatastropheID = c.CatastropheID

on c.ClaimID = ct.ClaimID

where ct.CreatedDate <= dr.AsOfDate

and ct.BatchNum in (-41, -777, -778)

and ct.ClaimAccountingTransactionTypeID between 1 and 8

and ( @OccurredOnly = 0

or c.LossDate >= dr.StartingDate)

and bc.CatastropheID is null

--AND lt.Description IS NOT NULL

group by ct.ClaimID,

lt.[Description],

st.StateCode,

dr.DateRangeID,

cast(c.ReportedDate as date),

year(c.LossDate)

),

ReserveByDate as (

select sm.ClaimID,

sm.DateRangeID,

cast(ct.CreatedDate as date) as ReserveDate,

sum(case when ct.ClaimAccountingTransactionTypeID < 5 or ct.DraftNumber = 'QUEUED' then ct.Total else 0.00 end) as Outstanding

from Summary as sm

inner join DateRange as dr

on dr.DateRangeID = sm.DateRangeID

inner join dbo.ClaimTransactions as ct

on ct.ClaimID = sm.ClaimID

where sm.Outstanding != 0

and ct.CreatedDate <= dr.AsOfDate

and ct.BatchNum in (-41, -777, -778)

and ct.ClaimAccountingTransactionTypeID between 1 and 8

group by sm.ClaimID,

sm.DateRangeID,

cast(ct.CreatedDate as date)

having sum(case when ct.ClaimAccountingTransactionTypeID < 5 or ct.DraftNumber = 'QUEUED' then ct.Total else 0.00 end) != 0

),

Reopened as (

select distinct

rbdl.ClaimID,

rbdl.DateRangeID

from ReserveByDate as rbdl

inner join ReserveByDate as rbdr

on rbdr.ClaimID = rbdl.ClaimID

and rbdr.ReserveDate <= rbdl.ReserveDate

group by rbdl.ClaimID,

rbdl.DateRangeID,

rbdl.ReserveDate

having sum(rbdr.Outstanding) = 0

),

ProofOfLoss as (

select distinct

sm.ClaimID,

sm.DateRangeID

from Summary as sm

inner join DateRange as dr

on dr.DateRangeID = sm.DateRangeID

where sm.Outstanding != 0

and exists ( select \*

from dbo.PrintedDocuments as pd

where pd.ClaimID = sm.ClaimID

and pd.ReportID = 1029

and pd.CreatedDate <= dr.AsOfDate)

),

RepresentedClaim as (

select c.ClaimID,

--'Represented' as Category

cast('With Rep.' as varchar(50)) as Category

from dbo.Claim as c

inner join dbo.PrintedDocuments as pd

on pd.ClaimID = c.ClaimID

and pd.ReportID in (1072, 1073)

where cast(pd.CreatedDate as date) <= @AsOfDate

union

select c.ClaimID,

--'Represented' as Category

cast('With Rep.' as varchar(50)) as Category

from dbo.Claim as c

inner join dbo.ClaimContactAssoc as cta

on cta.ClaimID = c.ClaimID

inner join dbo.Contact as co

on co.ContactID = cta.ContactID

and co.ContactTypeID in(15,16)

where cast(cta.AssociatedDate as date) <= @AsOfDate

--and ( cta.RemovedDate is null

-- or cast(cta.RemovedDate as date) > @AsOfDate)

),

FastTrack as (

select distinct

ClaimID,

'FastTrack' as Category

--'FastTrack-W/O Rep' as Category

from Claim.FastTrack as ft

where cast(ft.FastTrackedOn as date) <= @AsOfDate

--and ( ft.RemovedOn is null

-- or cast(ft.RemovedOn as date) > @AsOfDate)

)

select sm.ClaimID,

dr.DateRangeID,

cast(dr.EndingDate as date) as EndingDate,

convert(varchar(10), dr.StartingDate, 101) + ' - ' +

convert(varchar(10), dr.EndingDate, 101) + ' evaluated as of ' +

convert(varchar(10), dr.AsOfDate, 101) + '.' as Period,

sm.AccidentYear,

sm.[State],

sm.LossType,

-- coalesce(rc.Category, ft.Category, 'Regular') as Category,

--isnull(ft.Category, 'Regular') + isnull(rc.Category, '') as Category,

isnull(ft.Category, 'Regular') as Category,

--CASE

--WHEN sm.LossType IS NULL THEN 'Inforce'

--ELSE isnull(ft.Category, 'Regular') END as Category,

isnull(rc.Category, 'without Rep.') as Represented,

sm.Losses,

sm.Outstanding,

sm.LAE,

sm.DateReported,

sm.Closed,

iif(re.ClaimID is null, 0, 1) as Reopened,

iif(pol.ClaimID is null, 0, 1) as ProofOfLoss

into #Query

from Summary as sm

inner join DateRange as dr

on dr.DateRangeID = sm.DateRangeID

left join Reopened as re

on re.ClaimID = sm.ClaimID

and re.DateRangeID = sm.DateRangeID

left join ProofOfLoss as pol

on pol.ClaimID = sm.ClaimID

and pol.DateRangeID = sm.DateRangeID

left join RepresentedClaim as rc

on rc.ClaimID = sm.ClaimId

left join FastTrack as ft

on ft.ClaimID = sm.ClaimId

option (optimize for unknown)

end

if object\_id('tempdb.dbo.#InforceByDate') is not null drop table #InforceByDate

select dt.Period,

year(dt.AsOfDate) as AccidentYear,

st.StateCode as [State],

cast('' as varchar(50)) as LossType,

cast('' as varchar(50)) as Category,

cast('' as varchar(50)) as Represented,

count(\*) as [Count],

sum(pva.SubTotal) as Premium

into #InforceByDate

from ( select distinct

Period,

EndingDate

from #Query) as dt(Period, AsOfDate)

cross apply dbo.GetInforcePolicies(dt.AsOfDate) as ip

inner join dbo.Policy as p

inner join dbo.States as st

on st.StateID = p.StateID

on p.PolicyID = ip.PolicyID

and p.CompanyID = isnull(@CompanyID, p.CompanyID)

inner join dbo.PolicyVersionAttributes as pva

on pva.PolicyVersionID = ip.PolicyVersionID

group by dt.Period,

st.StateCode,

year(dt.AsOfDate)

begin with

ClosedByState as (

select q.Period,

q.[State],

q.AccidentYear,

q.LossType,

q.Category,

q.Represented,

count(\*) as [Count],

round(avg(cast(datediff(day, q.DateReported, q.Closed) as float)), 2) as AverageDaysToClose

from #Query as q

where q.Outstanding = 0

group by q.Period,

q.[State],

q.LossType,

q.Category,

q.Represented,

q.AccidentYear

),

IncurredByState as (

select q.Period,

q.[State],

q.LossType,

q.Category,

q.Represented,

q.AccidentYear,

count(\*) as [Count],

round(avg(q.Losses + q.Outstanding), 2) as AverageIncurred,

round(sum(q.Losses + q.Outstanding), 2) as TotalIncurred,

sum(q.Reopened) as Reopened,

sum(q.ProofOfLoss) as ProofOfLoss

from #Query as q

group by q.Period,

q.[State],

q.LossType,

q.Category,

q.Represented,

q.AccidentYear

),

PeriodState as (

select p.Period,

s.[State],

p.LossType,

p.Category,

p.Represented,

p.AccidentYear

from ( select Period,

AccidentYear,

LossType,

Category,

Represented

from IncurredByState

union

select Period,

AccidentYear,

LossType,

Category,

Represented

from #InforceByDate ) as p(Period, AccidentYear, LossType, Category, Represented),

( select [State]

from IncurredByState

union

select [State]

from #InforceByDate

) as s([State])

)

select ps.Period,

ps.[State],

ps.AccidentYear,

ps.LossType,

ps.Category,

ps.Represented,

--isnull(ins.[Count], 0) as InforceCount,

--isnull(ins.Premium, 0) as InforcePremium,

isnull(ibs.[Count], 0) as Reported,

isnull(ibs.AverageIncurred, 0) as AverageIncurred,

isnull(ibs.TotalIncurred, 0) as TotalIncurred,

isnull(cbs.[Count], 0) as Closed,

isnull(ibs.ProofOfLoss, 0) as ProofOfLoss,

isnull(cbs.AverageDaysToClose, 0) as AverageDaysToClose,

isnull(ibs.Reopened, 0) as Reopened

from PeriodState as ps

left join #InforceByDate as ins

on ins.Period = ps.Period

and ins.[State] = ps.[State]

and ins.AccidentYear = ps.AccidentYear

and ins.LossType = ps.LossType

and ins.Category = ps.Category

and ins.Represented = ps.Represented

left join IncurredByState as ibs

on ibs.Period = ps.Period

and ibs.[State] = ps.[State]

and ibs.AccidentYear = ps.AccidentYear

and ibs.LossType = ps.LossType

and ibs.Category = ps.Category

and ibs.Represented = ps.Represented

left join ClosedByState as cbs

on cbs.Period = ps.Period

and cbs.[State] = ps.[State]

and cbs.AccidentYear = ps.AccidentYear

and cbs.LossType = ps.LossType

and cbs.Category = ps.Category

and cbs.Represented = ps.Represented

where coalesce(ins.AccidentYear, ibs.AccidentYear, cbs.AccidentYear) is not null

AND isnull(ibs.[Count], 0) >0

order by ps.[State],

ps.AccidentYear,

ps.LossType,

ps.Category,

ps.Represented

end

if object\_id('tempdb.dbo.#InforceByDate') is not null drop table #InforceByDate

if object\_id('tempdb.dbo.#Query') is not null drop table #Query

end

USE [Atlas]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[reportGetClaimsByAdjusterYTD] Script Date: 2/17/2016 5:09:03 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER procedure [dbo].[reportGetClaimsByAdjusterYTD]

@AsOfDate as datetime = null

as

begin

set nocount on

declare @FirstOfYear as datetime,

@FirstOfMonth as datetime,

@FirstOfWeek as datetime

if @AsOfDate is null or @AsOfDate >= cast(getdate() as date)

set @AsofDate = dateadd(day, -1, cast(getdate() as date))

else

set @AsOfDate = cast(@AsOfDate as date)

select @FirstOfYear = dateadd(day, -datepart(dayofyear, @AsOfDate) + 1, @AsOfDate),

@FirstOfMonth = dateadd(day, -datepart(day, @AsOfDate) + 1, @AsOfDate),

@FirstOfWeek = dateadd(day, -datepart(dw, @AsOfDate) + 1, @AsOfDate),

@AsOfDate = dateadd(millisecond, -3, dateadd(day, 1, @AsOfDate))

if object\_id('tempdb.dbo.#AdjusterDetail') is not null drop table #AdjusterDetail

create table

#AdjusterDetail (

ClaimID integer,

AdjusterSystemUserID integer,

MTD integer,

WTD integer,

FastTrack integer,

FastTrackMTD integer,

FastTrackWTD integer,

[Open] integer,

PaidLoss money)

create clustered index idx\_AdjusterDetail\_AdjusterSystemUserID on #AdjusterDetail (AdjusterSystemUserID)

create index idx\_AdjusterDetail on #AdjusterDetail (ClaimID)

begin with

Assignments as (

select c.ClaimID,

ca.AdjusterSystemUserID,

ft.FastTrackID,

ft.FastTrackedOn,

ca.CreatedDate,

row\_number() over (partition by c.ClaimID, ca.AdjusterSystemUserID order by c.ClaimID) as DuplicateCount

from dbo.ClaimDetailsAndAssignments as ca

inner join dbo.FieldAdjuster as fa

on fa.AdjusterSystemUserID = ca.AdjusterSystemUserID

and fa.AdjustingCompanyContactID = ca.AdjustingCompanyContactID

inner join dbo.Claim as c

left join Claim.FastTrack as ft

on ft.ClaimID = c.ClaimID

and ft.FastTrackedOn <= @AsOfDate

on c.ClaimID = ca.ClaimID

where ca.CreatedDate

between @FirstOfYear

and @AsOfDate

and ca.ClaimDetailTypeID = 1

and ca.Active = 1

),

[Open] as (

select ass.ClaimID

from Assignments as ass

inner join dbo.ClaimStatusHistory as csh

inner join ( select max(ClaimStatusHistoryID) as ClaimStatusHistoryID

from dbo.ClaimStatusHistory

where CreatedDate <= @AsOfDate

group by ClaimID) as hid

on hid.ClaimStatusHistoryID = csh.ClaimStatusHistoryID

on ass.ClaimID = csh.ClaimID

where csh.ClaimStatusID = 1

),

PaidLosses as (

select ass.ClaimID,

ass.AdjusterSystemUserID,

sum(ct.Amount) as PaidLoss

from Assignments as ass

inner join dbo.ClaimTransactions as ct

on ct.ClaimId = ass.ClaimID

where ct.CreatedDate <= @AsOfDate

and ct.BatchNum in (-41,-777,-778)

and ct.ClaimAccountingTransactionTypeID < 7

and ass.DuplicateCount = 1

group by ass.ClaimID,

ass.AdjusterSystemUserID

having sum(ct.Amount) >= 0

)

insert into #AdjusterDetail

select ass.ClaimID,

ass.AdjusterSystemUserID,

iif(ass.CreatedDate >= @FirstOfMonth, 1, 0) as MTD,

iif(ass.CreatedDate >= @FirstOfWeek, 1, 0) as WTD,

iif(ass.FastTrackID is not null, 1, 0) as FastTrack,

iif(ass.FastTrackedOn >= @FirstOfMonth, 1, 0) as FastTrackMTD,

iif(ass.FastTrackedOn >= @FirstOfWeek, 1, 0) as FastTrackWTD,

iif(o.ClaimID is not null, 1, 0) as [Open],

isnull(pl.PaidLoss, 0) as PaidLoss

from Assignments as ass

left join PaidLosses as pl

on pl.ClaimID = ass.ClaimID

and pl.AdjusterSystemUserID = ass.AdjusterSystemUserID

left join [Open] as o

on o.ClaimID = ass.ClaimID

where ass.DuplicateCount = 1

end

if object\_id('tempdb.dbo.#CampusDetail') is not null drop table #CampusDetail

create table

#CampusDetail (

ClaimID integer,

AdjustingCompanyContactID integer,

AdjusterTitle varchar(100),

Location varchar(100),

MTD integer,

WTD integer,

FastTrack integer,

FastTrackMTD integer,

FastTrackWTD integer,

[Open] integer,

PaidLoss money)

create clustered index idx\_CampusDetail\_AdjustingCompanyContactID on #CampusDetail (AdjustingCompanyContactID)

begin with

Assignments as (

select c.ClaimID,

ca.AdjustingCompanyContactID,

fa.AdjusterTitle,

fa.Location,

ft.FastTrackID,

ft.FastTrackedOn,

ca.CreatedDate,

row\_number() over (partition by c.ClaimID, ca.AdjustingCompanyContactID, fa.Location order by c.ClaimID) as DuplicateCount

from dbo.ClaimDetailsAndAssignments as ca

inner join dbo.FieldAdjuster as fa

on fa.AdjustingCompanyContactID = ca.AdjustingCompanyContactID

and fa.AdjusterSystemUserID is null

inner join dbo.Claim as c

inner join dbo.PolicyVersion as pv

inner join dbo.Policy as p

on p.PolicyID = pv.PolicyID

on pv.PolicyVersionID = c.PolicyVersionID

left join Claim.FastTrack as ft

on ft.ClaimID = c.ClaimID

and ft.FastTrackedOn <= @AsOfDate

on c.ClaimID = ca.ClaimID

where ca.CreatedDate

between @FirstOfYear

and @AsOfDate

and ca.ClaimDetailTypeID = 1

and ca.Active = 1

and not exists ( select \*

from #AdjusterDetail as ad

where ad.ClaimID = ca.ClaimID)

),

[Open] as (

select ass.ClaimID

from Assignments as ass

inner join dbo.ClaimStatusHistory as csh

inner join ( select max(ClaimStatusHistoryID) as ClaimStatusHistoryID

from dbo.ClaimStatusHistory

where CreatedDate <= @AsOfDate

group by ClaimID) as hid

on hid.ClaimStatusHistoryID = csh.ClaimStatusHistoryID

on ass.ClaimID = csh.ClaimID

where csh.ClaimStatusID = 1

),

PaidLosses as (

select ass.ClaimID,

ass.AdjustingCompanyContactID,

ass.Location,

sum(ct.Amount) as PaidLoss

from Assignments as ass

inner join dbo.ClaimTransactions as ct

on ct.ClaimId = ass.ClaimID

where ct.CreatedDate <= @AsOfDate

and ct.BatchNum in (-41,-777,-778)

and ct.ClaimAccountingTransactionTypeID < 7

and ass.DuplicateCount = 1

group by ass.ClaimID,

ass.AdjustingCompanyContactID,

ass.Location

having sum(ct.Amount) >= 0

)

insert into #CampusDetail

select ass.ClaimID,

ass.AdjustingCompanyContactID,

ass.AdjusterTitle,

ass.Location,

iif(ass.CreatedDate >= @FirstOfMonth, 1, 0) as MTD,

iif(ass.CreatedDate >= @FirstOfWeek, 1, 0) as WTD,

iif(ass.FastTrackID is not null, 1, 0) as FastTrack,

iif(ass.FastTrackedOn >= @FirstOfMonth, 1, 0) as FastTrackMTD,

iif(ass.FastTrackedOn >= @FirstOfWeek, 1, 0) as FastTrackWTD,

iif(o.ClaimID is not null, 1, 0) as [Open],

isnull(pl.PaidLoss, 0) as PaidLoss

from Assignments as ass

left join PaidLosses as pl

on pl.ClaimID = ass.ClaimID

and pl.AdjustingCompanyContactID = ass.AdjustingCompanyContactID

and pl.Location = ass.Location

left join [Open] as o

on o.ClaimID = ass.ClaimID

where ass.DuplicateCount = 1

and not exists ( select \*

from #AdjusterDetail as ad

where ad.ClaimID = ass.ClaimID)

end

create index idx\_CampusDetail\_ClaimID on #CampusDetail (ClaimID, AdjusterTitle) include (Location)

delete from #CampusDetail

from #CampusDetail as cd

inner join dbo.Claim as c

inner join dbo.PolicyVersion as pv

inner join dbo.Policy as p

on p.PolicyID = pv.PolicyID

on pv.PolicyVersionID = c.PolicyVersionID

on c.ClaimID = cd.ClaimID

where cd.AdjusterTitle = 'In-house'

and (( cd.Location = 'FastTrack Campus' and p.StateID != 15)

or ( cd.Location = 'Other States Campus' and p.StateID = 15))

if object\_id('tempdb.dbo.#FieldDraft') is not null drop table #FieldDraft

create table

#FieldDraft (

ClaimID integer,

AdjusterSystemUserID integer,

YTD integer,

MTD integer,

WTD integer)

create clustered index idx\_FieldDraft\_ClaimID on #FieldDraft (ClaimID)

insert into #FieldDraft

select caid.ClaimID,

cda.CreatedBy as AdjusterSystemUserID,

sum(1) as YTD,

sum(iif(cda.CreatedDate >= @FirstOfMonth, 1, 0)) as MTD,

sum(iif(cda.CreatedDate >= @FirstOfWeek, 1, 0)) as WTD

from dbo.ClaimDraftActivity as cda

inner join ( select ad.ClaimID,

min(ClaimDraftActivityID) as ClaimDraftActivityID

from #AdjusterDetail as ad

inner join dbo.PropertyStatements as ps

inner join dbo.ClaimDrafts as cd

inner join dbo.ClaimDraftActivity as cda

on cda.ClaimDraftID = cd.ClaimDraftID

on cd.PropertyStatementID = ps.PropertyStatementID

on ps.ClaimID = ad.ClaimID

and cda.CreatedBy = ad.AdjusterSystemUserID

where cda.CreatedDate

between @FirstOfYear

and @AsOfDate

and cda.ClaimDraftStatusID = 3

group by ad.ClaimID,

cda.ClaimDraftID) as caid

on caid.ClaimDraftActivityID = cda.ClaimDraftActivityID

group by caid.ClaimID,

cda.CreatedBy

select cast(@AsOfDate as date) as AsOfDate,

case Location

when 'FastTrack South' then '01'

when 'FastTrack North' then '02'

when 'FastTrack Campus' then '03'

when 'Florida Large Loss' then '04'

when 'Other States' then '05'

else '06' end +

case AdjusterTitle

when 'Supervisor' then '01'

when 'Adjuster' then '02'

when 'In-house' then '03'

else '04' end as SortOrder,

fa.AdjusterName,

fa.AdjusterTitle,

fa.Location,

isnull(fa.Zone, 0) as Zone,

isnull(count(\*), 0) as TotalYTD,

isnull(sum(ad.MTD), 0) as TotalMTD,

isnull(sum(ad.WTD), 0) as TotalWTD,

isnull(sum(ad.[Open]), 0) as TotalOpen,

isnull(sum(ad.FastTrack), 0) as FastTrackYTD,

--isnull(sum(ad.MTD / nullif(ad.FastTrack, 0)), 0) as FastTrackMTD,

--isnull(sum(ad.WTD / nullif(ad.FastTrack, 0)), 0) as FastTrackWTD,

isnull(sum(ad.FastTrackMTD), 0) as FastTrackMTD,

isnull(sum(ad.FastTrackWTD), 0) as FastTrackWTD,

isnull(sum(fd.YTD), 0) as PrintedYTD,

isnull(sum(fd.MTD), 0) as PrintedMTD,

isnull(sum(fd.WTD), 0) as PrintedWTD,

isnull(avg(ad.PaidLoss), 0) as AverageYTD,

isnull(sum(ad.PaidLoss \* ad.MTD) / nullif(sum(ad.MTD), 0), 0) as AverageMTD,

isnull(sum(ad.PaidLoss \* ad.WTD) / nullif(sum(ad.WTD), 0), 0) as AverageWTD,

--isnull(sum(ad.PaidLoss \* (1 - ad.FastTrack)) / nullif(sum(1 - ad.FastTrack), 0), 0) as AverageYTD,

--isnull(sum(ad.PaidLoss \* (1 - ad.FastTrack) \* ad.MTD) / nullif(sum((1 - ad.FastTrack) \* ad.MTD), 0), 0) as AverageMTD,

--isnull(sum(ad.PaidLoss \* (1 - ad.FastTrack) \* ad.WTD) / nullif(sum((1 - ad.FastTrack) \* ad.WTD), 0), 0) as AverageWTD,

isnull(sum(ad.PaidLoss \* ad.FastTrack) / nullif(sum(ad.FastTrack), 0), 0) as FastTrackAverageYTD,

--isnull(sum(ad.PaidLoss \* ad.FastTrack \* ad.MTD) / nullif(sum(ad.FastTrack \* ad.MTD), 0), 0) as FastTrackAverageMTD,

--isnull(sum(ad.PaidLoss \* ad.FastTrack \* ad.WTD) / nullif(sum(ad.FastTrack \* ad.WTD), 0), 0) as FastTrackAverageWTD,

isnull(sum(ad.PaidLoss \* ad.FastTrackMTD) / nullif(sum(ad.FastTrackMTD), 0), 0) as FastTrackAverageMTD,

isnull(sum(ad.PaidLoss \* ad.FastTrackWTD) / nullif(sum(ad.FastTrackWTD), 0), 0) as FastTrackAverageWTD

from dbo.FieldAdjuster as fa

left join #AdjusterDetail as ad

left join #FieldDraft as fd

on fd.AdjusterSystemUserID = ad.AdjusterSystemUserID

and fd.ClaimID = ad.ClaimID

on ad.AdjusterSystemUserID = fa.AdjusterSystemUserID

where fa.AdjusterSystemUserID is not null

group by fa.AdjusterName,

fa.AdjusterTitle,

fa.Location,

isnull(fa.Zone, 0),

case Location

when 'FastTrack South' then '01'

when 'FastTrack North' then '02'

when 'FastTrack Campus' then '03'

when 'Florida Large Loss' then '04'

when 'Other States' then '05'

else '06' end +

case AdjusterTitle

when 'Supervisor' then '01'

when 'Adjuster' then '02'

when 'In-house' then '03'

else '04' end

union

select cast(@AsOfDate as date) as AsOfDate,

case fa.Location

when 'FastTrack South' then '01'

when 'FastTrack North' then '02'

when 'FastTrack Campus' then '03'

when 'Florida Large Loss' then '04'

when 'Other States' then '05'

else '06' end +

case fa.AdjusterTitle

when 'Supervisor' then '01'

when 'Adjuster' then '02'

when 'In-house' then '03'

else '04' end as SortOrder,

fa.AdjusterName,

fa.AdjusterTitle,

fa.Location,

isnull(fa.Zone, 0) as Zone,

isnull(count(\*), 0) as TotalYTD,

isnull(sum(cd.MTD), 0) as TotalMTD,

isnull(sum(cd.WTD), 0) as TotalWTD,

isnull(sum(cd.[Open]), 0) as TotalOpen,

isnull(sum(cd.FastTrack), 0) as FastTrackYTD,

--isnull(sum(cd.MTD / nullif(cd.FastTrack, 0)), 0) as FastTrackMTD,

--isnull(sum(cd.WTD / nullif(cd.FastTrack, 0)), 0) as FastTrackWTD,

isnull(sum(cd.FastTrackMTD), 0) as FastTrackMTD,

isnull(sum(cd.FastTrackWTD), 0) as FastTrackWTD,

isnull(sum(fd.YTD), 0) as PrintedYTD,

isnull(sum(fd.MTD), 0) as PrintedMTD,

isnull(sum(fd.WTD), 0) as PrintedWTD,

isnull(avg(cd.PaidLoss), 0) as AverageYTD,

isnull(sum(cd.PaidLoss \* cd.MTD) / nullif(sum(cd.MTD), 0), 0) as AverageMTD,

isnull(sum(cd.PaidLoss \* cd.WTD) / nullif(sum(cd.WTD), 0), 0) as AverageWTD,

--isnull(sum(cd.PaidLoss \* (1 - cd.FastTrack)) / nullif(sum(1 - cd.FastTrack), 0), 0) as AverageYTD,

--isnull(sum(cd.PaidLoss \* (1 - cd.FastTrack) \* cd.MTD) / nullif(sum((1 - cd.FastTrack) \* cd.MTD), 0), 0) as AverageMTD,

--isnull(sum(cd.PaidLoss \* (1 - cd.FastTrack) \* cd.WTD) / nullif(sum((1 - cd.FastTrack) \* cd.WTD), 0), 0) as AverageWTD,

isnull(sum(cd.PaidLoss \* cd.FastTrack) / nullif(sum(cd.FastTrack), 0), 0) as FastTrackAverageYTD,

--isnull(sum(cd.PaidLoss \* cd.FastTrack \* cd.MTD) / nullif(sum(cd.FastTrack \* cd.MTD), 0), 0) as FastTrackAverageMTD,

--isnull(sum(cd.PaidLoss \* cd.FastTrack \* cd.WTD) / nullif(sum(cd.FastTrack \* cd.WTD), 0), 0) as FastTrackAverageWTD

isnull(sum(cd.PaidLoss \* cd.FastTrackMTD) / nullif(sum(cd.FastTrackMTD), 0), 0) as FastTrackAverageMTD,

isnull(sum(cd.PaidLoss \* cd.FastTrackWTD) / nullif(sum(cd.FastTrackWTD), 0), 0) as FastTrackAverageWTD

from dbo.FieldAdjuster as fa

left join #CampusDetail as cd

left join #FieldDraft as fd

on fd.AdjusterSystemUserID = cd.AdjustingCompanyContactID

and fd.ClaimID = cd.ClaimID

on cd.AdjustingCompanyContactID = fa.AdjustingCompanyContactID

and cd.Location = fa.Location

where fa.AdjusterSystemUserID is null

group by fa.AdjusterName,

fa.AdjusterTitle,

fa.Location,

isnull(fa.Zone, 0),

case fa.Location

when 'FastTrack South' then '01'

when 'FastTrack North' then '02'

when 'FastTrack Campus' then '03'

when 'Florida Large Loss' then '04'

when 'Other States' then '05'

else '06' end +

case fa.AdjusterTitle

when 'Supervisor' then '01'

when 'Adjuster' then '02'

when 'In-house' then '03'

else '04' end

order by SortOrder,

AdjusterName

end

USE [Atlas]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[reportYearOverReserveAnalysis] Script Date: 2/18/2016 2:22:16 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

--[dbo].[reportYearOverReserveAnalysis] '20150101', '20151012', '20151012', 2, 2, 15, 43, 'With Rep.'

ALTER procedure [dbo].[reportYearOverReserveAnalysis]

@StartingDate as datetime = null,

@EndingDate as datetime = null,

@AsOfDate as datetime = null,

@LookBackPeriod as integer = null,

@CompanyID as integer = null,

@State as varchar(MAX) = null,

@County as varchar(MAX) = null,

@RepType as varchar(1024) = null

as

begin

set nocount on

if @EndingDate is null set @EndingDate = cast(dateadd(day, -day(getdate()), getdate()) as date)

if @StartingDate is null or @StartingDate > @EndingDate set @StartingDate = dbo.SpecialDate(@EndingDate, 'FOY')

if @AsOfDate is null set @AsOfDate = cast(dateadd(day, -1, getdate()) as date)

if @CompanyID not in (1, 2) set @CompanyID = null

if object\_id('tempdb.dbo.#Query') is not null drop table #Query

begin with

DateRange as (

select 1 as DateRangeID,

@StartingDate as StartingDate,

dateadd(millisecond, -3, dateadd(day, 1, @EndingDate)) as EndingDate,

dateadd(millisecond, -3, dateadd(day, 1, @AsOfDate)) as AsOfDate

union all

select dr.DateRangeID + 1 as DateRangeID,

dateadd(year, -1, dr.StartingDate) as StartingDate,

dateadd(year, -1, dr.EndingDate) as EndingDate,

dateadd(year, -1, dr.AsOfDate) as AsOfDate

from DateRange as dr

where dr.DateRangeID <= @LookBackPeriod

),

Summary as (

select

ct.ClaimID,

lt.[Description] as LossType,

st.StateID AS StateID,

st.StateCode as [State],

co.CountyID,

co.CountyDescription as County,

dr.DateRangeID,

year(c.LossDate) as AccidentYear,

cast(c.ReportedDate as date) as DateReported,

sum(case when ct.ClaimAccountingTransactionTypeID in (5, 6) and ct.DraftNumber <> 'QUEUED' then ct.Total else 0.00 end) as Losses,

sum(case when ct.ClaimAccountingTransactionTypeID < 5 or ct.DraftNumber = 'QUEUED' then ct.Total else 0.00 end) as Outstanding,

sum(case when ct.ClaimAccountingTransactionTypeID in (7, 8) then Amount else 0.00 end) as LAE,

cast(max(case when ct.ClaimAccountingTransactionTypeID < 5 or ct.DraftNumber = 'QUEUED' then ct.CreatedDate else null end) as date) as Closed,

pva.FormType

from Atlas.dbo.ClaimTransactions as ct

inner join Atlas.dbo.Claim as c

inner join dbo.LossType as lt

on lt.LossTypeID = c.LossTypeID

inner join DateRange as dr

on c.ReportedDate between dr.StartingDate

and dr.EndingDate

inner join Atlas.dbo.PolicyVersion as pv

INNER join [dbo].[PolicyVersionAttributes] pva

ON pva.PolicyVersionID = pv.PolicyVersionId

inner join Atlas.dbo.Policy as p

INNER JOIN Atlas.dbo.InsuredEntity as ie

on p.PolicyID = ie.PolicyID and p.CompanyID = isnull(@CompanyID, p.CompanyID)

INNER JOIN Atlas.dbo.Addresses as a

ON ie.PropertyAddressID = a.AddressId

inner join Atlas.dbo.States as st

on st.StateID = p.StateID

on p.PolicyID = pv.PolicyID

on pv.PolicyVersionID = c.PolicyVersionID

INNER JOIN Atlas.dbo.County co

ON a.CountyID = co.CountyID

left join dbo.Catastrophe as ca

inner join dbo.GetBODCatastrophe() as bc

on bc.CatastropheID = ca.CatastropheID

on ca.CatastropheID = c.CatastropheID

on c.ClaimID = ct.ClaimId

where ct.CreatedDate <= dr.AsOfDate

and ct.BatchNum in (-41, -777, -778)

and ct.ClaimAccountingTransactionTypeID between 1 and 8

and c.LossDate >= dr.StartingDate

and bc.CatastropheID is null

group by ct.ClaimID,

lt.[Description],

st.StateID,

st.StateCode,

co.CountyID,

co.CountyDescription,

dr.DateRangeID,

cast(c.ReportedDate as date),

year(c.LossDate),

pva.FormType

),

RepresentedClaim as (

select c.ClaimID,

--'Represented' as Category

cast('With Rep.' as varchar(50)) as Category

from dbo.Claim as c

inner join dbo.PrintedDocuments as pd

on pd.ClaimID = c.ClaimID

and pd.ReportID in (1072, 1073)

where cast(pd.CreatedDate as date) <= @AsOfDate

union

select c.ClaimID,

--'Represented' as Category

cast('With Rep.' as varchar(50)) as Category

from dbo.Claim as c

inner join dbo.ClaimContactAssoc as cta

on cta.ClaimID = c.ClaimID

inner join dbo.Contact as co

on co.ContactID = cta.ContactID

and co.ContactTypeID in(15,16)

where cast(cta.AssociatedDate as date) <= @AsOfDate

)

--FastTrack as (

--select distinct

-- ClaimID,

-- 'FastTrack' as Category

-- --'FastTrack-W/O Rep' as Category

--from Claim.FastTrack as ft

--where cast(ft.FastTrackedOn as date) <= @AsOfDate

-- --and ( ft.RemovedOn is null

-- -- or cast(ft.RemovedOn as date) > @AsOfDate)

--)

select

sm.ClaimID,

dr.DateRangeID,

cast(dr.EndingDate as date) as EndingDate,

convert(varchar(10), dr.StartingDate, 101) + ' - ' +

convert(varchar(10), dr.EndingDate, 101) + ' evaluated as of ' +

convert(varchar(10), dr.AsOfDate, 101) + '.' as Period,

sm.AccidentYear,

sm.StateID,

sm.[State],

sm.CountyID,

sm.County,

sm.LossType,

isnull(rc.Category, 'without Rep.') as Represented,

sm.Losses,

sm.Outstanding,

sm.LAE,

sm.DateReported,

sm.Closed,

sm.FormType

into #Query

from Summary as sm

inner join DateRange as dr

on dr.DateRangeID = sm.DateRangeID

left join RepresentedClaim as rc

on rc.ClaimID = sm.ClaimId

--left join FastTrack as ft

-- on ft.ClaimID = sm.ClaimId

option (optimize for unknown)

end

if object\_id('tempdb.dbo.#InforceByDate') is not null drop table #InforceByDate

select dt.Period,

year(dt.AsOfDate) as AccidentYear,

p.StateID,

st.StateCode as [State],

ad.CountyID,

c.CountyDescription as County,

pr.Name as FormType,

cast('' as varchar(50)) as LossType,

cast('' as varchar(50)) as Category,

cast('' as varchar(50)) as Represented,

count(\*) as [Count],

sum(pva.SubTotal) as Premium

into #InforceByDate

from ( select distinct

Period,

EndingDate

from #Query) as dt(Period, AsOfDate)

cross apply dbo.GetInforcePolicies(dt.AsOfDate) as ip

inner join dbo.Policy as p

inner join dbo.Product as pr

on pr.ProductID = p.ProductID

inner join dbo.InsuredEntity as ie

inner join dbo.Addresses as ad

inner join dbo.County c

on c.CountyID = ad.CountyID

on ad.AddressId = ie.PropertyAddressID

on ie.PolicyID = p.PolicyID

inner join dbo.States as st

on st.StateID = p.StateID

on p.PolicyID = ip.PolicyID

and p.CompanyID = isnull(@CompanyID, p.CompanyID)

inner join dbo.PolicyVersionAttributes as pva

on pva.PolicyVersionID = ip.PolicyVersionID

group by dt.Period,

year(dt.AsOfDate),

p.StateID,

st.StateCode,

ad.CountyID,

c.CountyDescription,

pr.Name

begin with

ClosedByState as (

select

q.Period,

q.StateID,

q.[State],

q.CountyID,

q.County,

q.AccidentYear,

q.LossType,

q.FormType,

q.Represented,

count(\*) as ClosedClaimCount,

ROUND(AVG(q.Losses),2) AS ClosedClaimsAVG,

ROUND(SUM(q.Losses),2) AS TotalClosedClaimsIncurred,

round(avg(cast(datediff(day, q.DateReported, q.Closed) as float)), 2) as AverageDaysToClose

from #Query as q

where q.Outstanding = 0

group by

q.Period,

q.StateID,

q.[State],

q.CountyID,

q.County,

q.LossType,

q.FormType,

q.Represented,

q.AccidentYear

),

IncurredByState as (

select

q.Period,

q.StateID,

q.[State],

q.CountyID,

q.County,

q.LossType,

q.FormType,

q.Represented,

q.AccidentYear,

--q.Losses AS Losses,

count(\*) as TotalClaimCount,

round(avg(q.Losses + q.Outstanding), 2) as AverageIncurred,

sum(q.Losses) as TotalPaid,

sum(q.Outstanding) as TotalOpenReserve

from #Query as q

group by

q.Period,

q.StateID,

q.[State],

q.CountyID,

q.County,

q.LossType,

q.FormType,

q.Represented,

q.AccidentYear--,

--q.Losses

),

PeriodState as (

select Period,

StateID,

[State],

CountyID,

County,

LossType,

FormType,

Represented,

AccidentYear

from ClosedByState

union

select Period,

StateID,

[State],

CountyID,

County,

LossType,

FormType,

Represented,

AccidentYear

from IncurredByState

union

select Period,

StateID,

[State],

CountyID,

County,

LossType,

FormType,

Represented,

AccidentYear

from #InforceByDate

)

select

ps.Period,

ps.StateID,

ps.[State],

ps.CountyID,

ps.County,

ps.AccidentYear,

ps.LossType,

ps.FormType,

ps.Represented,

isnull(ins.[Count], 0) as InforceCount,

isnull(ins.Premium, 0) as InforcePremium,

isnull(ibs.TotalClaimCount, 0) as TotalClaimCount,

--isnull(ibs.TotalIncurred, 0) as TotalPaid,

isnull(cbs.TotalClosedClaimsIncurred, 0) as TotalPaid,

(isnull(ibs.TotalClaimCount, 0) - isnull(cbs.ClosedClaimCount, 0)) AS OpenReserveCount,

--isnull(ibs.TotalOpenReserve + ibs.TotalPaid, 0) - isnull(cbs.TotalClosedClaimsIncurred, 0) as OpenReserve,

--isnull(ibs.TotalOpenReserve, 0) - isnull(cbs.TotalClosedClaimsIncurred, 0) as OpenReserve,

isnull(ibs.TotalOpenReserve,0) as OpenReserve,

isnull((isnull(ibs.TotalOpenReserve + ibs.TotalPaid, 0) - isnull(cbs.TotalClosedClaimsIncurred, 0)) / nullif((isnull(ibs.TotalClaimCount, 0) - isnull(cbs.ClosedClaimCount, 0)), 0), 0) AS OpenReserveAVG,

isnull(ibs.AverageIncurred, 0) as AveragePaid,

isnull(ibs.TotalPaid, 0) - isnull(cbs.TotalClosedClaimsIncurred, 0) as OpenTotalPaid,

isnull(cbs.ClosedClaimCount, 0) as ClosedClaimCount,

--ISNULL(cbs.ClosedClaimsAVG,0) AS ClosedClaimsAVG,

ISNULL(cbs.TotalClosedClaimsIncurred, 0) / nullif(isnull(cbs.ClosedClaimCount,0), 0) AS ClosedClaimsAVG,

isnull(cbs.AverageDaysToClose, 0) as AverageDaysToClose--,

from PeriodState as ps

left join #InforceByDate as ins

on ins.Period = ps.Period

and ins.[StateID] = ps.[StateID]

AND ins.CountyID = ps.CountyID

and ins.AccidentYear = ps.AccidentYear

and ins.LossType = ps.LossType

and ins.Represented = ps.Represented

and ins.FormType = ps.FormType

left join IncurredByState as ibs

on ibs.Period = ps.Period

and ibs.[StateID] = ps.[StateID]

AND ibs.CountyID = ps.CountyID

and ibs.AccidentYear = ps.AccidentYear

and ibs.LossType = ps.LossType

and ibs.Represented = ps.Represented

and ibs.FormType = ps.FormType

left join ClosedByState as cbs

on cbs.Period = ps.Period

and cbs.[StateID] = ps.[StateID]

AND cbs.CountyID = ps.CountyID

and cbs.AccidentYear = ps.AccidentYear

and cbs.LossType = ps.LossType

and cbs.Represented = ps.Represented

and cbs.FormType = ps.FormType

where coalesce(ibs.AccidentYear, cbs.AccidentYear, ins.AccidentYear) is not null

AND ps.[StateID] IN (SELECT StringValue FROM dbo.Split(@State,','))

AND ps.CountyID IN (SELECT StringValue FROM dbo.Split(@County,','))

AND (ps.Represented IN (SELECT StringValue FROM dbo.Split(@RepType,',')) or ps.Represented = '')

--AND isnull(ibs.[Count], 0) >0

order by ps.[State],

ps.AccidentYear,

ps.LossType,

ps.Represented

end

end

/\*

DATASET FOR DEFAULT PARAMETER VALUES

select dbo.SpecialDate(getdate(), 'FOY') as StartingDate,

cast(dateadd(day, -1, getdate()) as date) as EndingDate,

cast(dateadd(day, -1, getdate()) as date) as AsOfDate,

cast(2 as integer) as LookBackPeriod

\*/

/\*

STATES FILTER

SELECT DISTINCT S.[StateID],S.StateCode, S.[StateDescription]

FROM States S

WHERE S.[StateID] IN (15,25,30,13,32,42,67,36,35,17,21)

\*/

/\*

DECLARE @StateID varchar(50)

SET @StateID = ('15,25,30,13,32,42,67,36,35,17,21')

SELECT S.StateID, S.StateDescription AS [State]

from dbo.States S

WHERE S.[StateID] IN (SELECT StringValue FROM dbo.Split(@StateID,','))

Order by S.StateCode

\*/

/\*

SELECT DISTINCT CountyID,[StateID], CountyDescription

FROM [dbo].[County]

WHERE [StateID] IN (15,25,30,13,32,42,67,36,35,17,21)

\*/

/\*

SELECT c.CountyID,c.StateID, s.StateCode + ' - ' + c.CountyDescription as CountyDescription

from dbo.County c

JOIN dbo.Split(@StateID, ',') as sd

ON sd.StringValue = c.StateID

JOIN dbo.States s on s.StateID = c.StateID

Order by s.StateCode,c.CountyDescription

\*/

/\*

SELECT DISTINCT C.CountyID, S.[StateID], S.[StateDescription], C.CountyDescription

FROM [dbo].[County] C

INNER JOIN [dbo].[States] S

ON C.[StateID] = S.[StateID]

WHERE S.[StateID] IN (15,25,30,13,32,42,67,36,35,17,21)

\*/

USE [Atlas]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[reportMonthlyProductionByProducer] Script Date: 3/2/2016 3:17:04 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER procedure [dbo].[reportMonthlyProductionByProducer]

@AsOfDate as date = null,

@AgencyCode as varchar(20) = null

as

begin

set nocount on

if object\_id('tempdb.dbo.#DateRange') is not null drop table #DateRange

create table #DateRange (

AgencyID integer,

SortOrder integer,

CreatedDate date,

[Month] varchar(20),

StartingDate date,

EndingDate date)

begin with

DateRange as (

select ag.AgencyID,

1 as SortOrder,

cast(ag.CreatedDate as date) as CreatedDate,

datename(month, @AsOfDate) as [Month],

dateadd(day, -day(@AsOfDate) + 1, @AsOfDate) as StartingDate,

@AsOfDate as EndingDate

from dbo.Agency as ag

where ag.AgencyCode = @AgencyCode

union all

select dr.AgencyID,

dr.SortOrder + 1 as SortOrder,

dr.CreatedDate,

datename(month, dateadd(day, -1, dr.StartingDate)) as [Month],

dateadd(month, -1, dr.StartingDate) as StartingDate,

dateadd(day, -1, dr.StartingDate) as EndingDate

from DateRange as dr

where dr.EndingDate > eomonth(dr.CreatedDate)

and dr.StartingDate > dateadd(month, -11, dateadd(day, -day(@AsOfDate) + 1, @AsOfDate))

)

insert into #DateRange

select dr.AgencyID,

dr.SortOrder,

dr.CreatedDate,

dr.[Month],

dr.StartingDate,

dr.EndingDate

from DateRange as dr

order by dr.StartingDate

end

create clustered index idx\_DateRange\_AgencyID\_StartingDate on #DateRange (AgencyID, StartingDate)

if object\_id('tempdb.dbo.#Detail') is not null drop table #Detail

create table #Detail (

UserID integer,

SortOrder integer,

DataType varchar(100),

PolicyForm varchar(10),

[Count] integer)

insert into #Detail

select p.UserID,

dr.SortOrder,

'Quoted' as DataType,

pa.PolicyFormTypeTextChoice as PolicyForm,

count(pa.PolID) as [Count]

from #DateRange as dr

inner join AtlasBridge.dbo.Producers as p

inner join AtlasBridge.dbo.PolicyRelationships as pr

inner join AtlasBridge.dbo.PolicyAttributes as pa

on pa.PolID = pr.PolID

and pa.BillingTotalEstPremiumAmount > 0

on pr.ProducerID = p.UserID

inner join AtlasBridge.dbo.SystemUsers as su

inner join AtlasBridge.dbo.CRMCustomer as c

on c.UserID = su.UserID

on su.UserID = p.UserID

on p.AgencyID = dr.AgencyID

where cast(pa.PolicyCreationDate as date)

between dr.StartingDate

and dr.EndingDate

group by p.UserID,

dr.SortOrder,

pa.PolicyFormTypeTextChoice

insert into #Detail

select p.UserID,

dr.SortOrder,

'Bound' as DataType,

pa.PolicyFormTypeTextChoice as PolicyForm,

count(pa.PolID) as [Count]

from #DateRange as dr

inner join AtlasBridge.dbo.Producers as p

inner join AtlasBridge.dbo.PolicyRelationships as pr

inner join AtlasBridge.dbo.PolicyAttributes as pa

on pa.PolID = pr.PolID

and pa.QueueStage = 'Submitted'

on pr.LicenseNumber = p.LicenseNumber

inner join AtlasBridge.dbo.SystemUsers as su

inner join AtlasBridge.dbo.CRMCustomer as c

on c.UserID = su.UserID

on su.UserID = p.UserID

on p.AgencyID = dr.AgencyID

where pa.PolicyCreationDate >= dr.CreatedDate

and pa.AppSubmitDate

between dr.StartingDate

and dr.EndingDate

group by p.UserID,

dr.SortOrder,

pa.PolicyFormTypeTextChoice

option (force order)

select ag.AgencyCode,

ag.AgencyName,

cmb.UserID,

crm.FirstName + ' ' + crm.LastName as Producer,

cmb.DataType,

cmb.PolicyForm,

cmb.SortOrder,

cmb.[Month],

isnull(d.[Count], 0) as [Count]

from ( select d.UserID,

'Quoted' as DataType,

d.PolicyForm,

dr.AgencyID,

dr.SortOrder,

concat(left(dr.[Month], 3), ' ''', year(dr.StartingDate) % 100) as [Month]

from #DateRange as dr

cross apply #Detail as d

union

select d.UserID,

'Bound' as DataType,

d.PolicyForm,

dr.AgencyID,

dr.SortOrder,

concat(left(dr.[Month], 3), ' ''', year(dr.StartingDate) % 100) as [Month]

from #DateRange as dr

cross apply #Detail as d) as cmb

inner join AtlasBridge.dbo.Producers as pr

inner join AtlasBridge.dbo.SystemUsers as su

inner join AtlasBridge.dbo.CRMCustomer as crm

on crm.UserID = su.UserID

on su.UserID = pr.UserID

on pr.UserID = cmb.UserID

inner join dbo.Agency as ag

on ag.AgencyID = cmb.AgencyID

left join #Detail as d

on d.UserID = cmb.UserID

and d.DataType = cmb.DataType

and d.PolicyForm = cmb.PolicyForm

and d.SortOrder = cmb.SortOrder

order by Producer,

DataType,

PolicyForm,

SortOrder

end

USE [Atlas]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[ReportGetAgencyTop10\_DOMO] Script Date: 3/16/2016 11:01:56 AM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER proc [dbo].[ReportGetAgencyTop10\_DOMO]

as begin

set nocount on

begin with

RankPremByWeek as (

select

'Premium By Week' As RankType

,datepart(year, pa.AppSubmitDate) as [Year]

,NULL as [Month]

,datepart(week, pa.AppSubmitDate) as [Week]

,eomonth(dateadd(month, 0, pa.AppSubmitDate)) AS MonthEnd

,DATEADD(dd, 7-(DATEPART(dw, pa.AppSubmitDate)), CAST(pa.AppSubmitDate AS date)) AS WkEnd

,sum(pa.BillingTotalEstPremiumAmount) as Premium

,count(\*) as Policies

,rank() over (partition by datepart(year, pa.AppSubmitDate), datepart(week, pa.AppSubmitDate) order by sum(pa.BillingTotalEstPremiumAmount) desc) as Ranking

,CONCAT(a.AgencyName , ' - ' , a.AgencyCode) as AgencyName

,a.LicenseState AS State

from Agency as a

join AtlasBridge.dbo.Producers as p

join AtlasBridge.dbo.PolicyRelationships as pr

join AtlasBridge.dbo.PolicyAttributes as pa

on pa.PolID = pr.PolID

and pa.QueueStage = 'Submitted'

and pa.VIPSPolicyNumber is not null

and pa.VIPSPolicyNumber != ''

on pr.ProducerID = p.UserID

on p.AgencyID = a.AgencyID

--where a.AgencyCode = @agencyCode

WHERE datepart(year, pa.AppSubmitDate) > 2012

AND pa.PolicyFormTypeTextChoice <> 'HO4'

group by

a.AgencyID

,datepart(year, pa.AppSubmitDate)

,datepart(week, pa.AppSubmitDate)

,a.AgencyName

,a.AgencyCode

,a.LicenseState

,eomonth(dateadd(month, 0, pa.AppSubmitDate))

,DATEADD(dd, 7-(DATEPART(dw, pa.AppSubmitDate)), CAST(pa.AppSubmitDate AS date))

)

,

RankPremByMonth as (

select

'Premium By Month' As RankType

,datepart(year, pa.AppSubmitDate) as [Year]

,datename(month, pa.AppSubmitDate) as [Month]

,NULL as [Week]

,eomonth(dateadd(month, 0, pa.AppSubmitDate)) AS MonthEnd

--,DATEADD(dd, 7-(DATEPART(dw, pa.AppSubmitDate)), CAST(pa.AppSubmitDate AS date)) AS WkEnd

,NULL AS WkEnd

,sum(pa.BillingTotalEstPremiumAmount) as Premium

,count(\*) as Policies

,rank() over (partition by datepart(year, pa.AppSubmitDate), datename(month, pa.AppSubmitDate) order by sum(pa.BillingTotalEstPremiumAmount) desc) as Ranking

,CONCAT(a.AgencyName , ' - ' , a.AgencyCode) as AgencyName

,a.LicenseState AS State

from Agency as a

join AtlasBridge.dbo.Producers as p

join AtlasBridge.dbo.PolicyRelationships as pr

join AtlasBridge.dbo.PolicyAttributes as pa

on pa.PolID = pr.PolID

and pa.QueueStage = 'Submitted'

and pa.VIPSPolicyNumber is not null

and pa.VIPSPolicyNumber != ''

on pr.ProducerID = p.UserID

on p.AgencyID = a.AgencyID

--where a.AgencyCode = @agencyCode

WHERE datepart(year, pa.AppSubmitDate) > 2012

AND pa.PolicyFormTypeTextChoice <> 'HO4'

group by

a.AgencyID

,datepart(year, pa.AppSubmitDate)

,datename(month, pa.AppSubmitDate)

--,datepart(week, pa.AppSubmitDate)

,a.AgencyName

,a.AgencyCode

,a.LicenseState

,eomonth(dateadd(month, 0, pa.AppSubmitDate))

--,DATEADD(dd, 7-(DATEPART(dw, pa.AppSubmitDate)), CAST(pa.AppSubmitDate AS date))

)

,

RankPolicyByWeek as (

select

'Policys By Week' As RankType

,datepart(year, pa.AppSubmitDate) as [Year]

,NULL as [Month]

,datepart(week, pa.AppSubmitDate) as [Week]

,eomonth(dateadd(month, 0, pa.AppSubmitDate)) AS MonthEnd

,DATEADD(dd, 7-(DATEPART(dw, pa.AppSubmitDate)), CAST(pa.AppSubmitDate AS date)) AS WkEnd

,sum(pa.BillingTotalEstPremiumAmount) as Premium

,count(\*) as Policies

,rank() over (partition by datepart(year, pa.AppSubmitDate), datepart(week, pa.AppSubmitDate) order by count(pa.PolID) desc, sum(pa.BillingTotalEstPremiumAmount)desc) as Ranking

,CONCAT(a.AgencyName , ' - ' , a.AgencyCode) as AgencyName

,a.LicenseState AS State

from Agency as a

join AtlasBridge.dbo.Producers as p

join AtlasBridge.dbo.PolicyRelationships as pr

join AtlasBridge.dbo.PolicyAttributes as pa

on pa.PolID = pr.PolID

and pa.QueueStage = 'Submitted'

and pa.VIPSPolicyNumber is not null

and pa.VIPSPolicyNumber != ''

on pr.ProducerID = p.UserID

on p.AgencyID = a.AgencyID

--where a.AgencyCode = @agencyCode

WHERE datepart(year, pa.AppSubmitDate) > 2012

AND pa.PolicyFormTypeTextChoice <> 'HO4'

group by

a.AgencyID

,datepart(year, pa.AppSubmitDate)

,datepart(week, pa.AppSubmitDate)

,a.AgencyName

,a.AgencyCode

,a.LicenseState

,eomonth(dateadd(month, 0, pa.AppSubmitDate))

,DATEADD(dd, 7-(DATEPART(dw, pa.AppSubmitDate)), CAST(pa.AppSubmitDate AS date))

)

,

RankPolicyByMonth as (

select

'Policys By Month' As RankType

,datepart(year, pa.AppSubmitDate) as [Year]

,datename(month, pa.AppSubmitDate) as [Month]

,NULL as [Week]

,eomonth(dateadd(month, 0, pa.AppSubmitDate)) AS MonthEnd

--,DATEADD(dd, 7-(DATEPART(dw, pa.AppSubmitDate)), CAST(pa.AppSubmitDate AS date)) AS WkEnd

,NULL AS WkEnd

,sum(pa.BillingTotalEstPremiumAmount) as Premium

,count(\*) as Policies

,rank() over (partition by datepart(year, pa.AppSubmitDate), datename(month, pa.AppSubmitDate) order by count(pa.PolID) desc , sum(pa.BillingTotalEstPremiumAmount)desc) as Ranking

,CONCAT(a.AgencyName , ' - ' , a.AgencyCode) as AgencyName

,a.LicenseState AS State

from Agency as a

join AtlasBridge.dbo.Producers as p

join AtlasBridge.dbo.PolicyRelationships as pr

join AtlasBridge.dbo.PolicyAttributes as pa

on pa.PolID = pr.PolID

and pa.QueueStage = 'Submitted'

and pa.VIPSPolicyNumber is not null

and pa.VIPSPolicyNumber != ''

on pr.ProducerID = p.UserID

on p.AgencyID = a.AgencyID

--where a.AgencyCode = @agencyCode

WHERE datepart(year, pa.AppSubmitDate) > 2012

AND pa.PolicyFormTypeTextChoice <> 'HO4'

group by

a.AgencyID

,datepart(year, pa.AppSubmitDate)

,datename(month, pa.AppSubmitDate)

--,datepart(week, pa.AppSubmitDate)

,a.AgencyName

,a.AgencyCode

,a.LicenseState

,eomonth(dateadd(month, 0, pa.AppSubmitDate))

--,DATEADD(dd, 7-(DATEPART(dw, pa.AppSubmitDate)), CAST(pa.AppSubmitDate AS date))

)

select

RankType

,[Year]

,[Month]

,[Week]

,MonthEnd

,WkEnd

,Premium

,Policies

,Ranking

,AgencyName

,[State]

from RankPremByWeek

where Ranking <= 10

--ORDER BY [Year], [Week], Premium DESC

union

select

RankType

,[Year]

,[Month]

,[Week]

,MonthEnd

,WkEnd

,Premium

,Policies

,Ranking

,AgencyName

,[State]

from RankPremByMonth

where Ranking <= 10

--ORDER BY [Year], [Month], Premium DESC

union

select

RankType

,[Year]

,[Month]

,[Week]

,MonthEnd

,WkEnd

,Premium

,Policies

,Ranking

,AgencyName

,[State]

from RankPolicyByWeek

where Ranking <= 10

--ORDER BY [Year], [Week], Policies DESC

union

select

RankType

,[Year]

,[Month]

,[Week]

,MonthEnd

,WkEnd

--,SUM(Premium) as Premium

,Premium

,Policies

--,SUM(Policies) as Policies

,Ranking

,AgencyName

,[State]

from RankPolicyByMonth

where Ranking <= 10

--ORDER BY [Year], [Month], Policies DESC

--GROUP BY

--RankType

--,[Year]

--,[Month]

--,[Week]

--,MonthEnd

--,WkEnd

----,Policies

--,Ranking

--,AgencyName

--,[State]

end

end