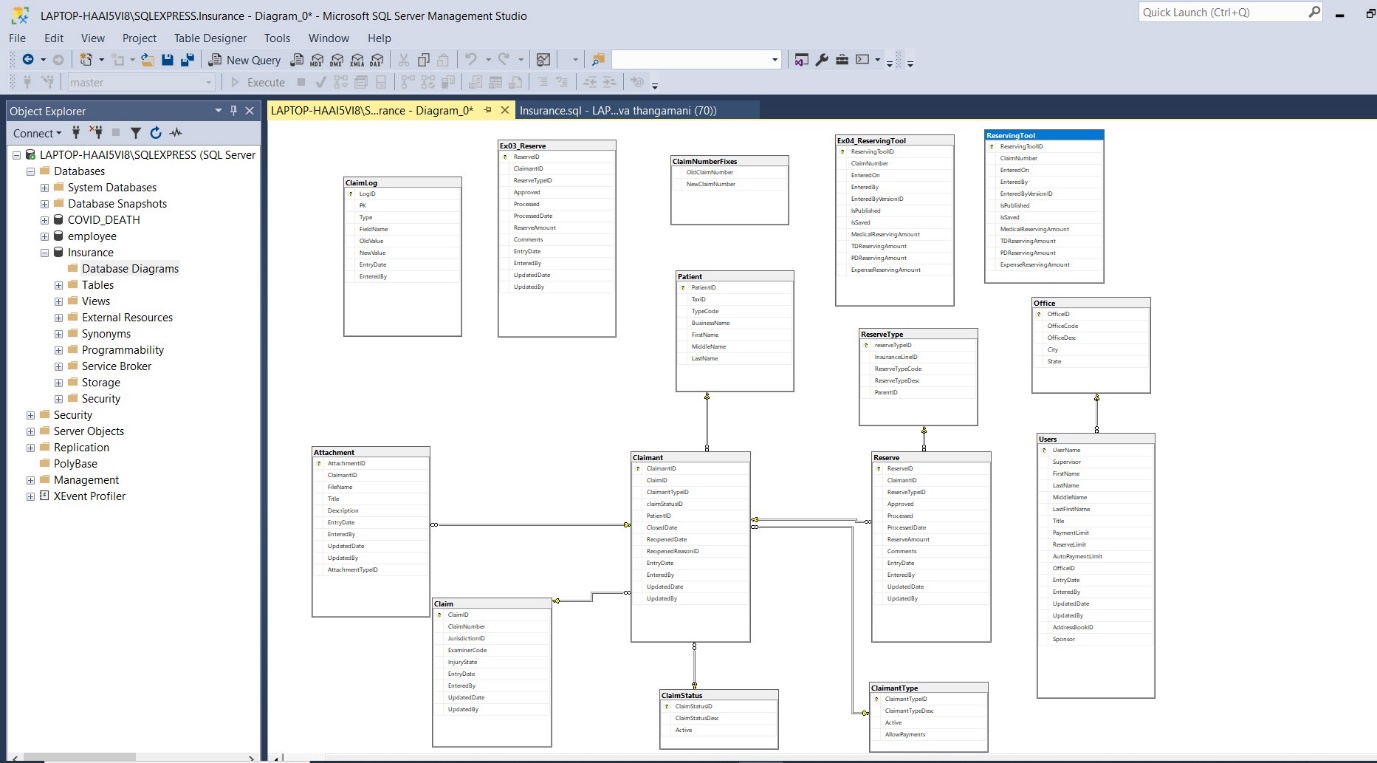
INSURANCE COMPANY DATABASE SCHEMA AND ANALYSIS



PROJECT1:

# Objective:

Develop a process that identifies, calculates, and displays critical information related to claims requiring periodic publishes in the Reserving Tool.

# Key Steps:

## 1. Identifying Claims that Require Publishes

Eligible Claims:  
- Status: Open or re-opened (if not reopened for bill payment).  
- Office: Located in Sacramento, San Francisco, or San Diego.

Specific Scenarios:  
- First Aid or Medical-Only claimants.  
- San Diego claims with examiners titled 'analyst' and reserve sums exceeding their reserve limit.  
- Sacramento or San Francisco claims meeting at least one of the following conditions:  
 - Medical reserve bucket total > $1000.  
 - Expense reserve bucket total > $100.  
 - Sum of TD, PD, and Rehab reserve buckets > $0.

## 2. Calculating Days Left/Overdue for a Publish

Rules for Publish Completion:  
- If no publishes exist, the examiner has 14 days from claim assignment to complete a publish.  
- If at least one publish exists, the examiner has the later of 90 days since the last publish or 14 days from assignment.

Calculation:  
- Days To Complete: Shows remaining days if publish is not yet overdue; otherwise 0.  
- Days Overdue: Shows overdue days if publish is past due; otherwise 0.

## 3. Displaying Key Information

Required fields to display:  
- Claim Number  
- Examiner Code, Name, and Title  
- Supervisor and Manager Details  
- Office, Claim Status, Claimant Name, and Claimant Type  
- Assigned Date, Reopened Date, and Adjusted Assigned Date  
- Last Published Date and related date calculations.

## 4. Accepting Parameters

Optional parameters to filter results:  
- Days To Complete and Days Overdue  
- Office, Manager, Supervisor, or Examiner Code  
- Team title containing specific keywords (e.g., 'analyst')  
- Claims without a publish if specified.

SOLUTION

CREATE PROCEDURE SPGetOutstandingRTPublish (

@DaysToComplete AS INT = NULL,

@DaysOverdue AS INT = NULL,

@Office AS VARCHAR(32) = NULL,

@ManagerCode AS VARCHAR(32) = NULL,

@SupervisorCode AS VARCHAR(32) = NULL,

@ExaminerCode AS VARCHAR(32) = NULL,

@Team AS VARCHAR(32) = NULL,

@ClaimsWithoutRTPublish AS BIT = 0

)

AS

BEGIN

DECLARE @DateAsOf date

SET @DateAsOf = '1/1/2019'

declare @ReservingToolPbl table (ClaimNumber varchar(max), LastPublishedDate datetime)

declare @AssignedDateLog table (PK int, ExaminerAssignedDate datetime)

INSERT INTO @ReservingToolPbl

SELECT ClaimNumber, max(EnteredOn) as LastPublishedDate

FROM Insurance.dbo.ReservingTool

where IsPublished = 1

group by ClaimNumber

insert into @AssignedDateLog (PK, ExaminerAssignedDate)

SELECT PK, max(EntryDate) as ExaminerAssignedDate

FROM Insurance.dbo.ClaimLog

where FieldName = 'examinercode'

group by PK

SELECT \*

FROM

(

SELECT ClaimNumber

, ManagerCode

, SupervisorCode

, ExaminerCode

, ManagerTitle

, SupervisorTitle

, ExaminerTitle

, ManagerName

, SupervisorName

, ExaminerName

, Office

, ClaimStatusDesc

, ClaimantName

, ClaimantTypeDesc

, ExaminerAssignedDate

, ReopenedDate

, AdjustedAssignedDate

, LastPublishedDate

, DaysSinceLastPublishedDate

, DaysSinceAdjustedAssignedDate

, CASE WHEN DaysSinceAdjustedAssignedDate >= 15 AND (DaysSinceLastPublishedDate >= 91 OR DaysSinceLastPublishedDate IS NULL) THEN 0

WHEN 91 - DaysSinceLastPublishedDate >= 15 - DaysSinceAdjustedAssignedDate AND DaysSinceLastPublishedDate IS NOT NULL THEN 91 - DaysSinceLastPublishedDate

ELSE 15 - DaysSinceAdjustedAssignedDate

END AS DaysToComplete

, CASE WHEN 14 >= DaysSinceAdjustedAssignedDate OR (90 >= DaysSinceLastPublishedDate AND DaysSinceLastPublishedDate IS NOT NULL) THEN 0

WHEN DaysSinceLastPublishedDate - 90 <= DaysSinceAdjustedAssignedDate - 14 AND DaysSinceLastPublishedDate IS NOT NULL THEN DaysSinceLastPublishedDate - 90

ELSE DaysSinceAdjustedAssignedDate - 14

END AS DaysOverdue

/\* ALTERNATE WAY TO DO DaysToComplete AND DaysOverdue \*/

/\*

, (SELECT MAX(DaysLeft1)

FROM (VALUES (91 - DaysSinceLastPublishedDate),(15 - DaysSinceAdjustedAssignedDate),(0)) AS DaysUnder(DaysLeft1)) AS DaysToComplete

, (SELECT MIN(DaysLeft2)

FROM (VALUES (DaysSinceLastPublishedDate - 90),(DaysSinceAdjustedAssignedDate - 14)) AS DaysOver(DaysLeft2)) AS DaysOverdue

\*/

FROM

(

SELECT

C.ClaimNumber

, R.ReserveAmount

, Office.OfficeDesc as Office

, U.UserName as ExaminerCode

, users2.UserName as SupervisorCode

, users3.UserName as ManagerCode

, U.Title as ExaminerTitle

, users2.Title as SupervisorTitle

, users3.Title as ManagerTitle

, CS.ClaimStatusDesc

, P.LastName + ', ' + TRIM(P.FirstName + ' ' + P.MiddleName) AS ClaimantName

, U.LastFirstName as ExaminerName

, users2.lastfirstname as SupervisorName

, users3.lastfirstname as ManagerName

, CL.ReopenedDate

, ADL.ExaminerAssignedDate

, CASE WHEN CL.ClaimStatusID = 2 AND CL.ReopenedDate > ADL.ExaminerAssignedDate

THEN CL.ReopenedDate

ELSE ADL.ExaminerAssignedDate

END as AdjustedAssignedDate

, CT.ClaimantTypeDesc

, Office.State

, U.ReserveLimit

, (CASE

WHEN RT.parentid in (1,2,3,4,5,10) THEN RT.parentid

ELSE RT.ReserveTypeID END

) AS ReserveCostID

, RTP.LastPublishedDate

, datediff(day, RTP.LastPublishedDate, @DateAsOf) as DaysSinceLastPublishedDate

, CASE WHEN CL.ClaimStatusID = 2 AND CL.ReopenedDate > ADL.ExaminerAssignedDate

THEN datediff(day, CL.ReopenedDate, @DateAsOf)

ELSE datediff(day, ADL.ExaminerAssignedDate, @DateAsOf)

END as DaysSinceAdjustedAssignedDate

from Claimant CL

INNER JOIN Claim C ON C.ClaimID=CL.ClaimID

INNER JOIN [Users] U on U.Username = C.ExaminerCode

INNER JOIN [Users] users2 on U.Supervisor = users2.UserName

INNER JOIN [Users] users3 on users2.Supervisor = users3.UserName

INNER JOIN [Office] on U.OfficeID = Office.OfficeID

INNER JOIN ClaimantType CT ON CT.ClaimantTypeID=CL.ClaimantTypeID

INNER JOIN Reserve R ON CL.ClaimantID=R.ClaimantID

LEFT JOIN ClaimStatus CS ON CS.ClaimStatusID=CL.ClaimStatusID

LEFT JOIN ReserveType RT ON R.ReserveTypeID=RT.ReserveTypeID

LEFT JOIN Patient P ON P.PatientID=CL.PatientID

LEFT JOIN @AssignedDateLog ADL ON ADL.PK = C.ClaimID

LEFT JOIN @ReservingToolPbl RTP ON RTP.Claimnumber = C.Claimnumber

WHERE

(RT.parentid in (1,2,3,4,5,10) or RT.ReserveTypeID in (1,2,3,4,5,10))

AND (CL.ClaimStatusID = 1 OR (CL.ClaimStatusID = 2 AND CL.ReopenedReasonID IN (1,2,4,5,6)))

and office.OfficeDesc in ('Sacramento', 'San Francisco', 'San Diego')

) as Basedata

PIVOT

(SUM(ReserveAmount)

FOR ReserveCostID IN ([1],[2],[3],[4],[5],[10])

) as PivTbl

WHERE

ClaimantTypeDesc IN ('First Aid', 'Medical-Only')

OR

(Office = 'San Diego'

AND isnull([1],0) + isnull([2],0) + isnull([3],0) + isnull([4],0) + isnull([5],0) >= ReserveLimit

AND ExaminerTitle LIKE '%analyst%'

)

OR

(Office in ('Sacramento', 'San Francisco')

AND (isnull([1],0) > 1000

or isnull([5],0) > 100

or (isnull([2],0) + isnull([3],0) + isnull([4],0) + isnull([10],0)) > 0

)

)

) MainQuery

WHERE (@DaysToComplete IS NULL OR DaysToComplete <= @DaysToComplete)

AND (@DaysOverdue IS NULL OR DaysOverdue >= @DaysOverdue)

AND (@Office IS NULL OR Office = @Office)

AND (@ManagerCode IS NULL OR ManagerCode = @ManagerCode)

AND (@SupervisorCode IS NULL OR SupervisorCode = @SupervisorCode)

AND (@ExaminerCode IS NULL OR ExaminerCode = @ExaminerCode)

AND (@Team IS NULL OR ExaminerTitle like '%' + @Team + '%'

OR SupervisorTitle like '%' + @Team + '%'

OR ManagerTitle like '%' + @Team + '%')

AND (@ClaimsWithoutRTPublish = 0 OR LastPublishedDate IS NULL)

ORDER BY

ManagerCode

, SupervisorCode

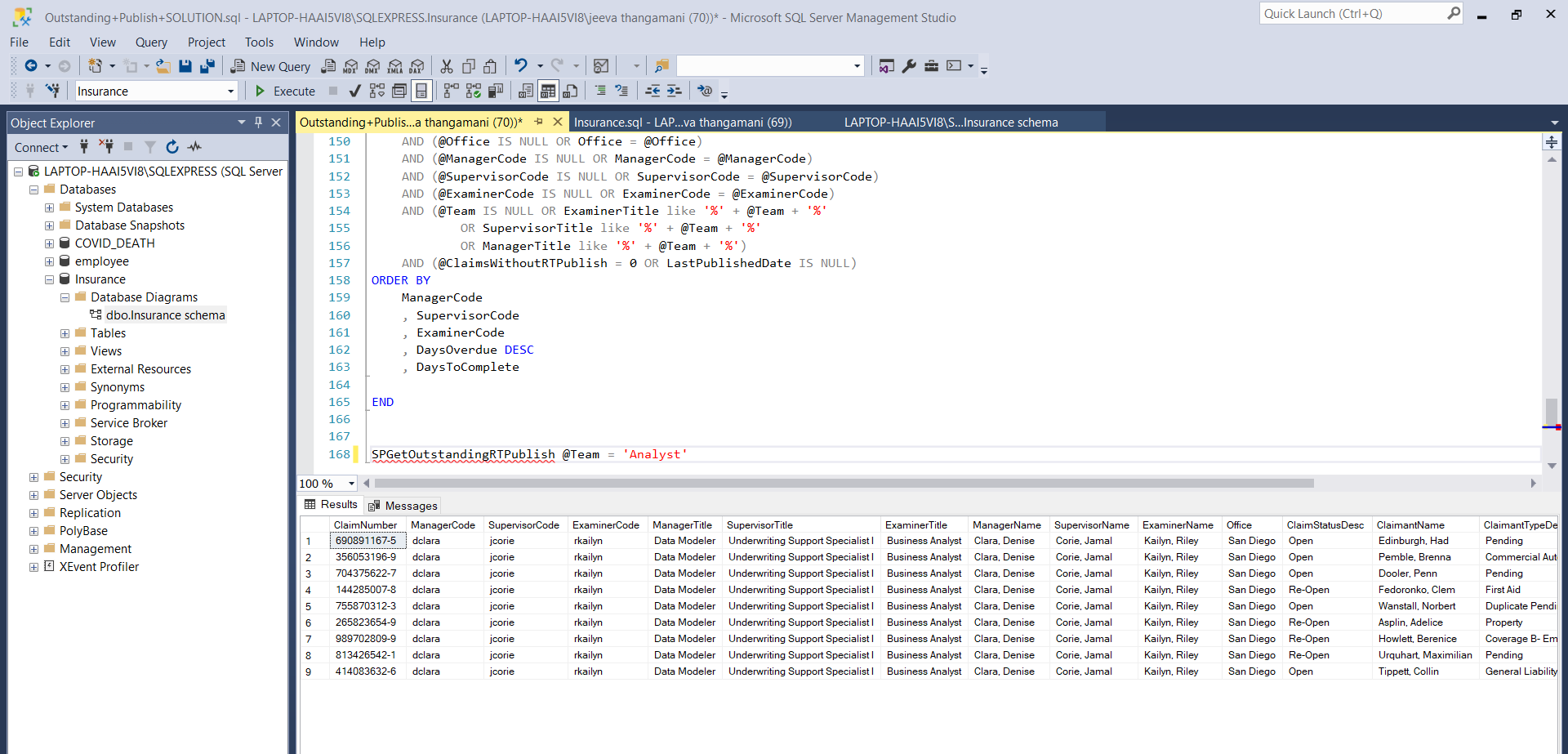
, ExaminerCode

, DaysOverdue DESC

, DaysToComplete

END

SPGetOutstandingRTPublish @Team = 'Analyst'



PROJECT 2:

Timeliness Project Description

Throughout the course, we worked on a project to identify how long until claims required another publish in the reserving tool. This complementary project, referred to as the 'Timeliness Project', focuses on analyzing already completed publishes to determine if they were completed in a timely manner.

# Project Requirements:

## 1. Grouping a Publish by Its Characteristics

The publishes will be grouped based on:  
- The month and year of the publish  
- Whether it was the first or subsequent publish on a claim  
- For the first publish: Whether the examiner handled the claim from the start or took over from another examiner

## 2. Determining the Timeliness Tier Based on Group

The timeliness of a publish will be categorized into different tiers based on the number of days to complete it.

### Timeliness Tiers:

1. \*\*First Publish - Examiner Handled Claim from Start\*\*  
- Tier 1: 10 or less days  
- Tier 2: 11-14 days  
- Tier 3: 15-20 days  
- Tier 4: 21-30 days  
- Tier 5: 31 or more days

2. \*\*First Publish - Examiner Transferred onto the Claim\*\*  
- Tier 1: 14 or less days  
- Tier 2: 15-45 days  
- Tier 3: 46-90 days  
- Tier 4: 91-120 days  
- Tier 5: 121 or more days

3. \*\*Subsequent Publishes\*\*  
- Tier 1: 30 or less days  
- Tier 2: 31-60 days  
- Tier 3: 61-90 days  
- Tier 4: 91-180 days  
- Tier 5: 181 or more days

## Determining Examiner Assignment:

The 'EnteredBy' field in the Reserving Tool is unreliable since anyone can enter records. Instead, the examiner assigned at the time of the publish is determined using the 'ClaimLog' table.

## Determining Examiner Transfer Status:

To check if the examiner was the first on the claim, compare the claim assignment date from the 'ClaimLog' table with the first reserve date. The first reserve date refers to the earliest non-zero reserve change posted by someone other than 'DBA'.

# Assumptions:

- Only open claims or claims reopened for reasons other than ReopenReasonID = 3 are considered.  
- If multiple publishes occur on the same day, only the last one is counted.

SOLUTION

USE Insurance

GO

/\* ---- Part 1 - Create a temporary table for first query ---- \*/

IF OBJECT\_ID('tempdb..#Temp\_Timeliness') IS NOT NULL

/\*Then it exists\*/

DROP TABLE #Temp\_Timeliness

CREATE TABLE #Temp\_Timeliness (

ReservingToolID INT PRIMARY KEY

, ClaimNumber VARCHAR(50) NOT NULL

, ClaimantID INT NOT NULL

, ExaminerCode VARCHAR(50)

, SupervisorCode VARCHAR(50)

, ManagerCode VARCHAR(50)

, ExaminerTitle VARCHAR(50)

, SupervisorTitle VARCHAR(50)

, ManagerTitle VARCHAR(50)

, Office VARCHAR(50)

, PublishedDate DATETIME

, PublishOrder INT

, HandledFromStartFlag BIT

, DaysSinceLastPublish INT

, DaysToFirstPublish INT

, PublishYear INT

, PublishMonth INT

)

/\* ------------------------------------------------------------- \*/

/\* ---- Part 2 - Insert the first query into the temp table ---- \*/

/\* ------------------------------------------------------------- \*/

INSERT INTO #Temp\_Timeliness

SELECT ReservingToolID, ClaimNumber, ClaimantID

, ExaminerCode, SupervisorCode, ManagerCode

, ExaminerTitle, SupervisorTitle, ManagerTitle

, Office

, PublishedDate

, PublishOrder

, HandledFromStartFlag

, DaysSinceLastPublish

, DaysToFirstPublish

, PublishYear

, PublishMonth

FROM (

SELECT sub2.\*

, DATEDIFF(day, PreviousPublishedDate, PublishedDate) as DaysSinceLastPublish

, CASE WHEN PublishOrder = 1 THEN DATEDIFF(day, AssignedDate, PublishedDate)

END AS DaysToFirstPublish

, YEAR(PublishedDate) as PublishYear

, MONTH(PublishedDate) as PublishMonth

FROM (

--------- add reserves to the list of claims by reserve type and process date

SELECT sub.ClaimNumber, sub.ClaimantID, sub.ReservingToolID

, sub.ExaminerCode, sub.SupervisorCode, sub.ManagerCode

, sub.ExaminerTitle, sub.SupervisorTitle, sub.ManagerTitle

, sub.Office, sub.AssignedDate, sub.PublishedDate

, row\_Number() OVER (partition by ClaimantID order by PublishedDate asc)

as PublishOrder

, CASE WHEN try\_convert(date, AssignedDate) <= try\_convert(date,

initialprocessdate) THEN 1 ELSE 0 END AS HandledFromStartFlag

, LAG(PublishedDate,1) OVER (PARTITION BY ClaimNumber ORDER BY

PublishedDate) as PreviousPublishedDate

FROM

(

--------- get dates, resolution type, and the last CP on each claim

SELECT C.ClaimNumber, cl.ClaimantID, RT.ReservingToolID

, x.newvalue as ExaminerCode, U2.Username as SupervisorCode,

U3.Username as ManagerCode

, U.Title as ExaminerTitle, U2.Title as SupervisorTitle, U3.Title as

ManagerTitle

, o.OfficeDesc as Office

, y.AssignedDate

, RT.EnteredOn as PublishedDate

, cl.ClosedDate

, cl.ClaimStatusID

, min(r.processeddate) as InitialProcessDate

FROM

(

--------- get list of claims with the last time the examiner changed

select pk, max(entrydate) as AssignedDate

from [ClaimLog]

where FieldName = 'examinercode'

group by pk

--order by a.pk

) y

INNER JOIN [ClaimLog] x ON x.PK = y.PK and x.EntryDate = y.AssignedDate

and x.FieldName = 'examinercode'

INNER JOIN [Claim] C ON C.ClaimID = y.PK

INNER JOIN [Claimant] Cl ON Cl.ClaimID = C.ClaimID

INNER JOIN [ReservingTool] RT ON C.ClaimNumber = RT.ClaimNumber and

RT.IsPublished = 1

LEFT JOIN [Reserve] R ON Cl.ClaimantID = R.ClaimantID

LEFT JOIN [Users] U ON x.newvalue = U.Username

LEFT JOIN [Users] U2 ON U.Supervisor = U2.Username

LEFT JOIN [Users] U3 ON U2.Supervisor = U3.Username

LEFT JOIN [Office] O ON U.OfficeID = O.OfficeID

WHERE

((cl.closeddate is null)

OR (cl.ReopenedDate > cl.ClosedDate and cl.reopenedreasonid <> 3)

)

and r.EnteredBy not like 'DBA'

GROUP BY C.ClaimNumber, cl.ClaimantID

, RT.ReservingToolID, y.AssignedDate

, x.newvalue, U2.Username, U3.Username

, U.Title, U2.Title, U3.Title

, o.officedesc

, cl.ClosedDate

, RT.EnteredOn

, cl.ClaimStatusID

) sub

WHERE

PublishedDate >= AssignedDate

) sub2

) sub3

WHERE DaysSinceLastPublish IS NULL OR DaysSinceLastPublish > 0

/\* ---- Part 3 - Use the temp table in a query that groups and aggregates the data for

the final results ---- \*/

SELECT PublishOrder

, HandledFromStartFlag

, PublishYear

, PublishMonth

, Tier1

, Tier2

, Tier3

, Tier4

, Tier5

FROM (

SELECT ReservingToolID

, CASE WHEN PublishOrder = 1 THEN 'First' ELSE 'Subsequent' END AS

PublishOrder

, CASE WHEN PublishOrder = 1 THEN

CASE WHEN HandledFromStartFlag = 1 THEN 'HandledFromStart'

ELSE 'Transferred' END

ELSE

'N/A'

END AS HandledFromStartFlag

, PublishYear

, PublishMonth

, CASE WHEN PublishOrder = 1 THEN

CASE WHEN HandledFromStartFlag = 1 THEN

CASE WHEN DaysToFirstPublish IS NULL THEN NULL

WHEN DaysToFirstPublish <= 10 THEN 'Tier1'

WHEN DaysToFirstPublish <= 14 THEN 'Tier2'

WHEN DaysToFirstPublish <= 20 THEN 'Tier3'

WHEN DaysToFirstPublish <= 30 THEN 'Tier4'

WHEN DaysToFirstPublish >= 31 THEN 'Tier5'

ELSE 'Other' END

ELSE

CASE WHEN DaysToFirstPublish IS NULL THEN NULL

WHEN DaysToFirstPublish <= 14 THEN 'Tier1'

WHEN DaysToFirstPublish <= 45 THEN 'Tier2'

WHEN DaysToFirstPublish <= 90 THEN 'Tier3'

WHEN DaysToFirstPublish <= 120 THEN 'Tier4'

WHEN DaysToFirstPublish >= 121 THEN 'Tier5'

ELSE 'Other' END

END

ELSE

CASE WHEN DaysSinceLastPublish IS NULL THEN NULL

WHEN DaysSinceLastPublish <= 30 THEN 'Tier1'

WHEN DaysSinceLastPublish <= 60 THEN 'Tier2'

WHEN DaysSinceLastPublish <= 90 THEN 'Tier3'

WHEN DaysSinceLastPublish <= 180 THEN 'Tier4'

WHEN DaysSinceLastPublish >= 181 THEN 'Tier5'

ELSE 'Other' END

END AS DayTiers

FROM #Temp\_Timeliness TT

GROUP BY ReservingToolID

, CASE WHEN PublishOrder = 1 THEN 'First' ELSE 'Subsequent' END

, CASE WHEN PublishOrder = 1 THEN

CASE WHEN HandledFromStartFlag = 1 THEN 'HandledFromStart'

ELSE 'Transferred' END

ELSE

'N/A'

END

, PublishYear

, PublishMonth

, CASE WHEN PublishOrder = 1 THEN

CASE WHEN HandledFromStartFlag = 1 THEN

CASE WHEN DaysToFirstPublish IS NULL THEN NULL

WHEN DaysToFirstPublish <= 10 THEN 'Tier1'

WHEN DaysToFirstPublish <= 14 THEN 'Tier2'

WHEN DaysToFirstPublish <= 20 THEN 'Tier3'

WHEN DaysToFirstPublish <= 30 THEN 'Tier4'

WHEN DaysToFirstPublish >= 31 THEN 'Tier5'

ELSE 'Other' END

ELSE

CASE WHEN DaysToFirstPublish IS NULL THEN NULL

WHEN DaysToFirstPublish <= 14 THEN 'Tier1'

WHEN DaysToFirstPublish <= 45 THEN 'Tier2'

WHEN DaysToFirstPublish <= 90 THEN 'Tier3'

WHEN DaysToFirstPublish <= 120 THEN 'Tier4'

WHEN DaysToFirstPublish >= 121 THEN 'Tier5'

ELSE 'Other' END

END

ELSE

CASE WHEN DaysSinceLastPublish IS NULL THEN NULL

WHEN DaysSinceLastPublish <= 30 THEN 'Tier1'

WHEN DaysSinceLastPublish <= 60 THEN 'Tier2'

WHEN DaysSinceLastPublish <= 90 THEN 'Tier3'

WHEN DaysSinceLastPublish <= 180 THEN 'Tier4'

WHEN DaysSinceLastPublish >= 181 THEN 'Tier5'

ELSE 'Other' END

END

) BaseData

PIVOT

(count(ReservingToolID)

FOR DayTiers IN ([Tier1]

, [Tier2]

, [Tier3]

, [Tier4]

, [Tier5])

) as PivtTbl

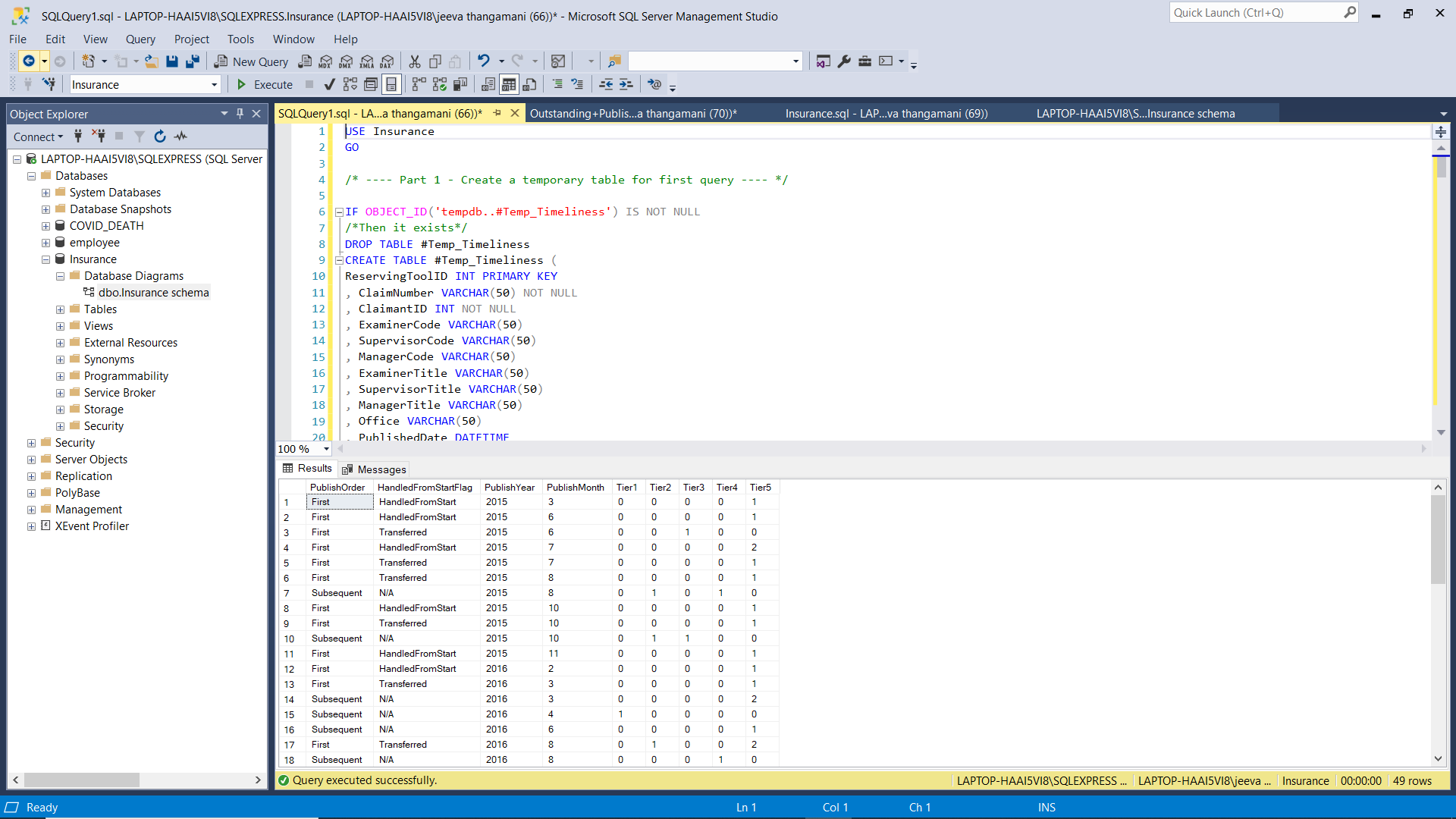
ORDER BY PublishYear, PublishMonth, PublishOrder, HandledFromStartFlag

/\*---- Drop temp table when query is done ---- \*/

IF OBJECT\_ID('tempdb..#Temp\_Timeliness') IS NOT NULL

/\*Then it exists\*/

DROP TABLE #Temp\_Timeliness



PROJECT 3

Time To Publish - Project Description

This project focuses on determining how much time claims professionals are spending to publish claims in the reserving tool. A query is needed to provide insights into the total time spent on each publish.

**Required Fields:**

- Claim Number  
- Office  
- Claims Professional Name (aka Examiner)  
- New or Updated publish status  
- Total time spent to publish  
- Publish date

**Determining Who Worked on the Claim:**

The 'EnteredBy' field in the Reserving Tool is unreliable as anyone can create records. Instead, the ClaimLog table provides the examiner assignment at the time of publish. If the examiner is no longer in the Users table, the publish should still be included.

If a claim is reassigned to another examiner mid-publish, the total time still counts towards the final publish.

**Calculating Time Spent on a Claim:**

To calculate the total time spent, sum the periods between when work starts and stops. For simple cases with a single retrieval and publish date, subtract the retrieval date from the publish date. For more complex cases, exclude 'down time' when the examiner was not actively working on the claim.

**When Work Starts:**

A new record with IsSaved = 0 and IsPublished = 0 indicates the start of work. If this record is missing, use the first save of the day (as long as it’s not the only record or the only publish of the day).

**When Work Stops:**

Work is considered complete when IsSaved = 1 or IsPublished = 1. If multiple saves occur in a row, assume continuous work. Ignore saves if they follow the last publish of the day.

**Determining New or Updated Publish:**

The first publish for a claim is marked as 'New'. Any subsequent publishes are labeled as 'Updated'.

**Assumptions:**

- Examiners do not usually work at night.  
- If no save follows a new record on the same day, assume no time was spent.  
- Ignore the save if it follows the last publish of the day.  
- For multiple publishes on the same day, only the last one counts.

SOLUTION

USE Insurance

GO

/\*

-------------------------------------------------------------

------------------- Days to Publish Query -------------------

-------------------------------------------------------------

\*/

SELECT

Pub.ClaimNumber

, Pub.OfficeCode

, Pub.ExaminerCodeAtTimeOfPublish

, CASE WHEN SUM(CASE WHEN NewOrUpdated\_Save = 'New' THEN 1 ELSE 0 END) > 0 THEN

'New' ELSE 'Updated' END AS NewOrUpdated\_Publish

, SUM(HoursFinal) as HoursSpentToPublish

, Pub.DatePublished

FROM

(

------------------- Date Published with Examiner Query -------------------

SELECT ClaimNumber

, O.OfficeCode

, clmLg.NewValue as 'ExaminerCodeAtTimeOfPublish'

, DatePublished

, PriorDatePublished

FROM

(

--------- Get the Date Published query along with the last time the examiner

select pk

, pub1.ClaimNumber

, pub1.DatePublished

, pub1.PriorDatePublished

, max(z.entrydate) as AssignedDate

from [ClaimLog] z

INNER JOIN [Claim] C ON C.ClaimID = z.PK

INNER JOIN

(

/\*

------------------- Date Published along with Prior Date Published

Query

------------------- The Prior Date Published will be used to join

onto the larger Save and Retrieve Query\*/

SELECT ClaimNumber, EnteredOn as DatePublished

, LAG (EnteredOn, 1, NULL) OVER (PARTITION BY ClaimNumber

ORDER BY EnteredOn) AS PriorDatePublished

FROM [ReservingTool]

WHERE IsPublished = 1 AND ClaimNumber IS NOT NULL

--ORDER BY ClaimNumber, EnteredOn DESC

) Pub1

ON Pub1.ClaimNumber = C.ClaimNumber AND z.entrydate <

Pub1.DatePublished

where FieldName = 'examinercode'

group by pk

, pub1.ClaimNumber

, pub1.DatePublished

, pub1.PriorDatePublished

--order by pk

--order by ClaimNumber

) z

INNER JOIN [ClaimLog] clmLg ON clmLg.PK = z.PK and clmLg.EntryDate = z.AssignedDate and clmLg.FieldName = 'ExaminerCode'

LEFT JOIN [Users] U ON U.Username = clmLg.NewValue

LEFT JOIN [Office] O ON O.OfficeID = U.OfficeID

--order by claimnumber

) Pub

INNER JOIN

(

------------------- Query to find the final time between retrievals and saves

-------------------

SELECT DISTINCT

ClaimNumber

, MaxSavedDate as MaxSavedOrPublishedDate

, MinRetrievalDate

, TotalSubtractHours

, NewOrUpdated\_Save

, DATEDIFF(SECOND, MinRetrievalDate, MaxSavedDate)/(60.0 \* 60.0) +

TotalSubtractHours AS HoursFinal

FROM

(

/\*

------------ Here we find the Maximum Saved Date and the Minimum Retrieval

Date. The difference will be the Gross Time Spent.

------------ We will then calculate the exact number of hours worked by

subtracting out the down-time

------------ This query finds the down time by finding which retrieves

were not the first retrieve on

------------ the day (Flag\_FirstRetrieveOfDay = 0), and finding if that

retrieve came after a save

------------ earlier in the day (PriorSave\_SameDay).

------------ If it did come after a save earlier in the day, we find the

time between the retrieve and

------------ the prior save = the down-time.\*/

SELECT \*

, MAX(DateSavedOrPublished) OVER (PARTITION BY ClaimNumber, convert

(date, DateSavedOrPublished)) as MaxSavedDate

, MIN(RetrieveDate) OVER (PARTITION BY ClaimNumber, convert(date,

DateSavedOrPublished)) as MinRetrievalDate

, SUM(CASE WHEN Flag\_FirstRetrieveOfDay = 0 THEN DATEDIFF(SECOND,

RetrieveDate, PriorSave\_SameDay)/(60.0 \* 60.0) ELSE 0 END) OVER

(PARTITION BY ClaimNumber, convert(date, DateSavedOrPublished)) AS

TotalSubtractHours

, CASE WHEN SUM(NewFlag) OVER (PARTITION BY ClaimNumber, convert(date,

DateSavedOrPublished)) > 0 THEN 'New' ELSE 'Updated' END as

NewOrUpdated\_Save

FROM

(

SELECT Sav.ClaimNumber

, Sav.IsPublished

, Sav.DateSavedOrPublished

, Sav.PriorDateSavedOrPublished

, CASE WHEN Sav.PriorDateSavedOrPublished IS NULL THEN NULL

WHEN convert(date, Sav.DateSavedOrPublished) = convert

(date, Sav.PriorDateSavedOrPublished) THEN

Sav.PriorDateSavedOrPublished

ELSE NULL

END as PriorSave\_SameDay

, Ret.RetrieveDate

, Ret.PriorRetrieveDate

, CONVERT(DATETIME, MAX(Flag\_LastSaveOfTheDay \* convert(float,

DateSavedOrPublished)) OVER (PARTITION BY Sav.ClaimNumber,

convert(date, Sav.DateSavedOrPublished))) as

RemoveSavesAfterThisTimeInDay

, Flag\_FirstRetrieveOfDay

, ROW\_NUMBER() OVER (PARTITION BY Sav.ClaimNumber,

Sav.DateSavedOrPublished ORDER BY Ret.RetrieveDate) as

RetrieveOrder

, CASE WHEN Sav.PriorDateSavedOrPublished IS NULL THEN 1 ELSE 0

END AS NewFlag

FROM

(/\*

------------------- Find all the Save/Publish dates, the

------------------- prior saves/publishes on their claim,

and

------------------- flag if it is the last save/publish\*/

SELECT ClaimNumber

, IsPublished

, EnteredOn as DateSavedOrPublished

, LAG (EnteredOn, 1, NULL) OVER (PARTITION BY ClaimNumber

ORDER BY EnteredOn) AS PriorDateSavedOrPublished

, CASE WHEN SUM(ispublished + 0) OVER (PARTITION BY

ClaimNumber, convert(date, enteredon)) > 0

THEN CASE WHEN CONVERT(float, enteredon) = MAX

(CONVERT(float, enteredon) \* IsPublished) OVER (PARTITION BY

ClaimNumber, convert(date, enteredon)) THEN 1 ELSE 0 END

ELSE CASE WHEN enteredon = MAX(EnteredOn) OVER

(PARTITION BY ClaimNumber, convert(date, enteredon)) THEN 1 ELSE

0 END

END AS Flag\_LastSaveOfTheDay

FROM [ReservingTool]

WHERE IsSaved = 1 AND ClaimNumber IS NOT NULL

--ORDER BY ClaimNumber, EnteredOn DESC

) Sav

LEFT JOIN

(

------------------- Find all the Retrieve dates, the

------------------- prior retrieves on their claim, and

------------------- flag if it is the first retrieve

SELECT ClaimNumber, EnteredOn as RetrieveDate

, LAG (EnteredOn, 1, NULL) OVER (PARTITION BY ClaimNumber

ORDER BY EnteredOn) AS PriorRetrieveDate

, CASE WHEN enteredon = MIN(EnteredOn) OVER (PARTITION BY

ClaimNumber, convert(date, enteredon)) THEN 1 ELSE 0 END AS

Flag\_FirstRetrieveOfDay

FROM [ReservingTool]

WHERE IsPublished = 0 AND IsSaved = 0 AND ClaimNumber IS NOT

NULL

--ORDER BY ClaimNumber, EnteredOn DESC

) Ret

ON Sav.ClaimNumber = Ret.ClaimNumber

AND (CONVERT(date, Sav.DateSavedOrPublished) = TRY\_CONVERT(date,

Ret.RetrieveDate) OR Ret.RetrieveDate IS NULL)

AND Ret.RetrieveDate <= Sav.DateSavedOrPublished

AND (Ret.RetrieveDate > Sav.PriorDateSavedOrPublished OR

Sav.PriorDateSavedOrPublished IS NULL)

WHERE (Ret.RetrieveDate IS NULL

OR CONVERT(date, Sav.DateSavedOrPublished) = TRY\_CONVERT(date,

Ret.RetrieveDate))

) x

WHERE RetrieveOrder = 1 AND DateSavedOrPublished <=

RemoveSavesAfterThisTimeInDay

) y

WHERE MinRetrievalDate IS NOT NULL

--ORDER BY ClaimNumber, MaxSavedDate

) SavRet

ON Pub.ClaimNumber = SavRet.ClaimNumber

AND SavRet.MaxSavedOrPublishedDate <= Pub.DatePublished

AND (SavRet.MaxSavedOrPublishedDate > Pub.PriorDatePublished OR

Pub.PriorDatePublished IS NULL)

GROUP BY Pub.ClaimNumber

, Pub.OfficeCode

, Pub.ExaminerCodeAtTimeOfPublish

, Pub.DatePublished

ORDER BY Pub.ClaimNumber, DatePublished

