Sub Plot1()

Dim WS As Worksheet

Dim gatePlot As Worksheet

Dim FLIGHTTIME As String

Dim flightGate As Integer

Dim flightNumber As String

Dim aircraft As String ' Variable to store aircraft value

Dim dest As String ' Variable to store DEST value

Dim inbound As String ' Variable to store Inbound value

Dim apt As String ' Variable to store APT value

Dim flt2 As String ' Variable to store FLT2 value

Dim flightRow As Integer

Dim gateRow As Integer

Dim gateColumn As Integer

Dim minuteCell As Range

Dim greenCell As Range

Dim startCell As Range

Dim endCell As Range

Dim i As Integer

' Define the relevant worksheets

Set WS = ThisWorkbook.Sheets("1")

Set gatePlot = ThisWorkbook.Sheets("GATEPLOT")

' Dictionary to keep track of gate occurrences

Dim gateOccurrences As Object

Set gateOccurrences = CreateObject("Scripting.Dictionary")

' Loop through each cell in column G of "1" sheet

For flightRow = 3 To WS.Cells(WS.Rows.Count, "G").End(xlUp).row

' Check if the cell has a white fill color

If WS.Cells(flightRow, "G").Interior.Color = RGB(255, 255, 255) Then

' Extract flight time and gate from column G

FLIGHTTIME = Trim(WS.Cells(flightRow, "G").Text) ' Flight time as string

' Round the flight time to the nearest 5-minute interval

FLIGHTTIME = RoundToNearest5(FLIGHTTIME)

' Extract flight gate from column D

Dim flightGateValue As Variant

flightGateValue = WS.Cells(flightRow, "D").value

' Check if the flight gate value is numeric

If IsNumeric(flightGateValue) Then

' Convert the value to an Integer

flightGate = CInt(flightGateValue)

Else

' Handle the case where the value is not numeric

MsgBox "Invalid flight gate value in cell D" & flightRow, vbExclamation

' Skip to the next iteration of the loop

GoTo NextRow

End If

' Extract flight number from column B

flightNumber = WS.Cells(flightRow, "B").value ' Flight number as string

' Extract aircraft from column C

aircraft = WS.Cells(flightRow, "C").value ' Aircraft as string

' Extract DEST from column E

dest = WS.Cells(flightRow, "E").value ' DEST as string

' Check if there is a value one cell above in the same column as aircraft

If flightRow > 3 Then

If WS.Cells(flightRow - 1, "C").value = aircraft Then

' Extract the Inbound value from one cell above in the same column

inbound = WS.Cells(flightRow - 1, "C").value

' Extract APT value from column A of the same row

apt = WS.Cells(flightRow - 1, "A").value

' Extract FLT2 value from column B of the same row

flt2 = WS.Cells(flightRow - 1, "B").value

Else

' Clear the APT and FLT2 values if there is no matching aircraft value one row above

apt = ""

flt2 = ""

End If

End If

' Find the corresponding hour column in GATEPLOT sheet

gateColumn = FindHourColumn(gatePlot, Left(FLIGHTTIME, 2))

If gateColumn <> 0 Then

' Check if minute value is "00"

If Right(FLIGHTTIME, 2) = "00" Then

' If minute value is "00", directly assign minute cell to corresponding hour column

Set minuteCell = gatePlot.Cells(1, gateColumn)

Else

' Find the corresponding minute cell in GATEPLOT sheet

Set minuteCell = FindMinuteCell(gatePlot.Rows(1), Right(FLIGHTTIME, 2), gateColumn)

End If

If Not minuteCell Is Nothing Then

' Calculate the row to insert "Gate" based on flight gate

If gateOccurrences.Exists(flightGate) Then

' If gate already exists in dictionary, increment occurrence count

gateOccurrences(flightGate) = gateOccurrences(flightGate) + 1

Else

' If gate is encountered for the first time, set occurrence count to 1

gateOccurrences(flightGate) = 1

End If

' Determine the row to insert the gate value based on the gate number

Select Case flightGate

Case 10

gateRow = 20

Case 11

gateRow = 22

Case 12

gateRow = 24

Case 13

gateRow = 26

Case 14

gateRow = 28

Case Else

gateRow = (flightGate \* 2) + IIf(gateOccurrences(flightGate) Mod 2 = 0, -1, 0)

End Select

' Determine the column to insert the gate value (1 cell left)

Dim greenColumn As Integer

greenColumn = minuteCell.Column - 1

' Mark the gate by filling the cell with green color

gatePlot.Cells(gateRow + 1, greenColumn).Interior.Color = RGB(0, 255, 0) ' Green color

' Insert the flight number into the shaded green cell

gatePlot.Cells(gateRow + 1, greenColumn).value = flightNumber

' Insert the APT value 7 cells to the left of the shaded green cell if not exceeding column B

If greenColumn > 10 And Not IsEmpty(apt) Then

gatePlot.Cells(gateRow + 1, greenColumn).Offset(0, -7).value = apt

End If

' Insert the FLT2 value 6 cells to the left of the shaded green cell if not exceeding column B

If greenColumn > 8 And Not IsEmpty(flt2) Then

gatePlot.Cells(gateRow + 1, greenColumn).Offset(0, -6).value = flt2

End If

' Insert the DEST value 2 cells to the left of the shaded green cell if not exceeding column B

If greenColumn > 2 Then

gatePlot.Cells(gateRow + 1, greenColumn).Offset(0, -2).value = dest

End If

' Insert the aircraft value 5 cells to the left of the shaded green cell if not exceeding column B

If greenColumn > 5 Then

gatePlot.Cells(gateRow + 1, greenColumn).Offset(0, -5).value = aircraft

End If

End If

End If

End If

NextRow:

Next flightRow

End Sub

Function FindHourColumn(WS As Worksheet, hour As String) As Integer

Dim cell As Range

For Each cell In WS.Rows(1).Cells

If Left(cell.value, 2) = hour Then

FindHourColumn = cell.Column

Exit Function

End If

Next cell

FindHourColumn = 0 ' If hour not found, return 0

End Function

Function FindMinuteCell(row As Range, minuteValue As String, startColumn As Integer) As Range

Dim cell As Range

Dim cleanedMinuteValue As String

cleanedMinuteValue = Right("0" & Replace(minuteValue, ":", ""), 2) ' Remove colon and ensure 2 digits

Dim searchRange As Range

' Set the search range to start from the next column after the identified hour column

Set searchRange = row.Cells(1, startColumn + 1).Resize(1, 12)

' Round the minute value to the nearest 5-minute interval

Dim roundedMinute As Integer

roundedMinute = Round(val(minuteValue) / 5, 0) \* 5

If roundedMinute = 60 Then

roundedMinute = 0

End If

' Format the rounded minute value

cleanedMinuteValue = Format(roundedMinute, "00")

For Each cell In searchRange

If Right(Replace(cell.value, ":", ""), 2) = cleanedMinuteValue Then

Set FindMinuteCell = cell

Exit Function

End If

Next cell

Set FindMinuteCell = Nothing ' If minute not found, return Nothing

End Function

Function RoundToNearest5(timeString As String) As String

Dim hours As Integer

Dim minutes As Integer

Dim roundedMinutes As Integer

' Parse the time string into hours and minutes

hours = val(Left(timeString, 2))

minutes = val(Right(timeString, 2))

' Round the minutes to the nearest 5-minute interval

roundedMinutes = WorksheetFunction.Round(minutes / 5, 0) \* 5

' Handle the case where rounding to the next hour is required

If roundedMinutes >= 60 Then

hours = hours + 1

roundedMinutes = 0

End If

' Format the rounded time

RoundToNearest5 = Format(TimeSerial(hours, roundedMinutes, 0), "hh:mm")

End Function

Sub Plot2()

Dim WS As Worksheet

Dim gatePlot As Worksheet

Dim arrivalTime As String

Dim matchValue As String

Dim arrivalRow As Integer

Dim gateRow As Integer

Dim gateColumn As Integer

Dim minuteCell As Range

Dim greenCell As Range

Dim startCell As Range

Dim endCell As Range

Dim i As Integer

' Define the relevant worksheets

Set WS = ThisWorkbook.Sheets("1")

Set gatePlot = ThisWorkbook.Sheets("GATEPLOT")

' Loop through each cell in column H of "4" sheet

For arrivalRow = 3 To WS.Cells(WS.Rows.Count, "H").End(xlUp).row

' Check if the cell has a non-white fill color

If WS.Cells(arrivalRow, "H").Interior.Color <> RGB(255, 255, 255) Then

' Check if the corresponding cell in column M has "TURN" value

If WS.Cells(arrivalRow, "M").value = "TURN" Then

' Extract arrival time and match value from column H and B respectively

arrivalTime = Trim(WS.Cells(arrivalRow, "H").Text) ' Arrival time as string

matchValue = Trim(WS.Cells(arrivalRow, "B").Offset(1, 0).Text) ' Match value as string

' Check if the arrival time is before "0500"

If val(Left(arrivalTime, 2)) < 5 Then

arrivalTime = "05:00" ' Set arrival time to "05:00" if it's earlier than "0500"

Else

' Round the arrival time to the nearest 5-minute interval

arrivalTime = RoundToNearest5(arrivalTime)

End If

' Find the corresponding hour column in GATEPLOT sheet

gateColumn = FindHourColumn(gatePlot, Left(arrivalTime, 2))

If gateColumn <> 0 Then

' Check if minute value is "00"

If Right(arrivalTime, 2) = "00" Then

' If minute value is "00", directly assign minute cell to corresponding hour column

Set minuteCell = gatePlot.Cells(1, gateColumn)

Else

' Find the corresponding minute cell in GATEPLOT sheet

Set minuteCell = FindMinuteCell(gatePlot.Rows(1), Right(arrivalTime, 2), gateColumn)

End If

If Not minuteCell Is Nothing Then

' Find the green cell with match value

Set greenCell = gatePlot.Cells.Find(What:=matchValue, LookIn:=xlValues, LookAt:=xlWhole)

If Not greenCell Is Nothing Then

' Determine the row to insert the purple fill color

gateRow = greenCell.row

' Mark the cell with a purple fill color

If gateRow > 1 Then ' Exclude header row

Dim markedCell As Range

Set markedCell = gatePlot.Cells(gateRow, minuteCell.Column)

markedCell.Interior.Color = RGB(219, 59, 227) ' Custom color "#db3be3"

' Look to the right of the marked cell in the same row and identify the first cell with a 3-letter value

Dim nextCell As Range

Dim lastColumn As Long

lastColumn = gatePlot.Cells(gateRow, gatePlot.Columns.Count).End(xlToLeft).Column

For Each nextCell In gatePlot.Range(markedCell.Offset(0, 1), gatePlot.Cells(gateRow, lastColumn))

If Len(nextCell.value) = 3 Then

' Copy the 3-letter value and insert it into the marked cell

markedCell.value = nextCell.value

' Clear the contents of the cell that was copied

nextCell.ClearContents

' Look to the right of the marked cell in the same row and identify the first cell with a 3- to 4-numbered value

Dim numberedCell As Range

For Each numberedCell In gatePlot.Range(markedCell.Offset(0, 1), gatePlot.Cells(gateRow, lastColumn))

If IsNumeric(numberedCell.value) And Len(numberedCell.value) >= 3 And Len(numberedCell.value) <= 4 Then

' Check if the numbered value is within 3 cells of the marked cell

If Abs(numberedCell.Column - markedCell.Column) <= 3 Then

' Append the numbered value to the 3-letter value in the same cell (e.g., "MSP 346")

markedCell.value = markedCell.value & " " & numberedCell.value

Else

' Otherwise, insert the numbered value two cells to the right of the marked cell

markedCell.Offset(0, 2).value = numberedCell.value

End If

' Clear the contents of the numbered cell that was copied

numberedCell.ClearContents

' Exit the loop after copying the value

Exit For

End If

Next numberedCell

' Exit the loop after copying the 3-letter value

Exit For

End If

Next nextCell

End If

' Shade cells between purple-marked cells and next green cell with the same match value

ShadeCellsBetween gatePlot, gateRow, minuteCell.Column, matchValue

End If

End If

End If

End If

End If

Next arrivalRow

End Sub

Function RoundToNearest55(timeString As String) As String

Dim hours As Integer

Dim minutes As Integer

Dim roundedMinutes As Integer

' Parse the time string into hours and minutes

hours = val(Left(timeString, 2))

minutes = val(Right(timeString, 2))

' Round the minutes to the nearest 5-minute interval

roundedMinutes = WorksheetFunction.RoundDown(minutes / 5, 0) \* 5

' Handle the case where rounding to the next hour is required

If roundedMinutes = 60 Then

roundedMinutes = 0

hours = hours + 1

End If

' Format the rounded time

RoundToNearest5 = Format(TimeSerial(hours, roundedMinutes, 0), "hh:mm")

End Function

Sub ShadeCellsBetween(WS As Worksheet, row As Integer, startCol As Integer, matchValue As String)

Dim cell As Range

Dim endCol As Integer

Dim lastCol As Integer

Dim currentCol As Integer

' Determine the last column in the worksheet

lastCol = WS.Cells(row, WS.Columns.Count).End(xlToLeft).Column

' Initialize the current column to the next column after the starting column

currentCol = startCol

' Find the end column based on the next green cell with the same match value or the last column

Do While currentCol <= lastCol

Set cell = WS.Cells(row, currentCol)

If cell.value = matchValue And cell.Interior.Color = RGB(0, 255, 0) Then

endCol = currentCol

Exit Do

End If

currentCol = currentCol + 1

Loop

' If no matching green cell was found, set the end column to the last column of the worksheet

If currentCol > lastCol Then

endCol = lastCol

End If

' Shade cells between startCol and endCol (including startCol)

If endCol >= startCol Then ' Check if there are cells to shade

For Each cell In WS.Range(WS.Cells(row, startCol), WS.Cells(row, endCol))

If row > 1 Then ' Exclude header row

cell.Interior.Color = RGB(219, 59, 227) ' Custom color "#db3be3"

End If

Next cell

End If

End Sub

Sub Plot3()

Dim gatePlot As Worksheet

Dim greenColor As Long ' RGB value for the green color

Dim cell As Range

Dim adjacentCells As Range

Dim cellValue As String

On Error Resume Next ' Enable error handling

' Define the relevant worksheet

Set gatePlot = ThisWorkbook.Sheets("GATEPLOT")

' Check if the worksheet exists

If gatePlot Is Nothing Then

MsgBox "Worksheet 'GATEPLOT' not found!", vbExclamation

Exit Sub

End If

' Get the RGB value for the green color used in the existing script

greenColor = RGB(0, 255, 0) ' Green color used in the existing script

' Loop through each cell in the GATEPLOT sheet

For Each cell In gatePlot.UsedRange.Cells

' Check if the cell has a green fill color and a 3 to 4-letter value

If Not cell.Interior.ColorIndex = xlNone Then ' Check if cell has fill color

If cell.Interior.Color = greenColor And Len(cell.value) >= 3 And Len(cell.value) <= 4 Then

' Shade nine cells to the left with the same green color

If cell.Column > 9 Then ' Check if enough cells to the left

Set adjacentCells = gatePlot.Range(gatePlot.Cells(cell.row, cell.Column - 9), cell.Offset(0, -1))

Else

Set adjacentCells = gatePlot.Range(gatePlot.Cells(cell.row, 2), cell.Offset(0, -1))

End If

adjacentCells.Interior.Color = greenColor

End If

End If

Next cell

On Error GoTo 0 ' Disable error handling

If Not gatePlot Is Nothing Then Set gatePlot = Nothing ' Release object reference

End Sub

Sub Plot4()

Dim WS As Worksheet

Dim gatePlot As Worksheet

Dim arrivalTime As String

Dim gate As Integer

Dim arrivalRow As Integer

Dim gateRow As Integer

Dim gateColumn As Integer

Dim minuteCell As Range

Dim blueCell As Range

Dim tailCell As Range

' Define the relevant worksheets

Set WS = ThisWorkbook.Sheets("1")

Set gatePlot = ThisWorkbook.Sheets("GATEPLOT")

' Dictionary to keep track of gate occurrences

Dim gateOccurrences As Object

Set gateOccurrences = CreateObject("Scripting.Dictionary")

' Loop through each cell in column H of "1" sheet

For arrivalRow = 3 To WS.Cells(WS.Rows.Count, "H").End(xlUp).row

' Check if the cell has a non-white fill color

If WS.Cells(arrivalRow, "H").Interior.Color <> RGB(255, 255, 255) Then

' Look in column C of the same row to identify the "Tail" value

Set tailCell = WS.Cells(arrivalRow, "C")

' Check if the "Tail" value exists

If Not IsEmpty(tailCell.value) Then

Dim tailValue As String

tailValue = Trim(tailCell.Text)

' Look 1 row below the "Tail" value

If Not IsEmpty(tailCell.Offset(1, 0).value) Then

Dim nextRowTailValue As String

nextRowTailValue = Trim(tailCell.Offset(1, 0).Text)

' If the value in the cell 1 row below the identified "Tail" value does not match the tail value

If tailValue <> nextRowTailValue Then

' Extract the 4-digit number value from column H as "arrival time"

arrivalTime = Trim(WS.Cells(arrivalRow, "H").Text)

' Extract the value in column D as "gate"

gate = val(Trim(WS.Cells(arrivalRow, "D").Text))

' Round the arrival time to the nearest 5-minute interval

arrivalTime = RoundToNearest5(arrivalTime)

' Check if the arrival time is between "0500" and "0600"

If Not (arrivalTime >= "0500" And arrivalTime <= "0600") Then

' Find the corresponding hour column in GATEPLOT sheet

gateColumn = FindHourColumn(gatePlot, Left(arrivalTime, 2))

If gateColumn <> 0 Then

' Check if minute value is "00"

If Right(arrivalTime, 2) = "00" Then

' If minute value is "00", directly assign minute cell to corresponding hour column

Set minuteCell = gatePlot.Cells(1, gateColumn)

Else

' Find the corresponding minute cell in GATEPLOT sheet

Set minuteCell = FindMinuteCell(gatePlot.Rows(1), Right(arrivalTime, 2), gateColumn)

End If

If Not minuteCell Is Nothing Then

' Calculate the row to insert "Gate" based on gate occurrences

If gateOccurrences.Exists(gate) Then

' If gate already exists in dictionary, increment occurrence count

gateOccurrences(gate) = gateOccurrences(gate) + 1

Else

' If gate is encountered for the first time, set occurrence count to 1

gateOccurrences(gate) = 1

End If

' Determine the row to insert the light blue fill color based on gate occurrences

gateRow = (gate \* 2) + IIf(gateOccurrences(gate) Mod 2 = 0, -1, 0)

' Mark the cell with a light blue fill color one cell below the identified cell

If gateRow > 1 Then ' Exclude header row

Set blueCell = gatePlot.Cells(gateRow + 1, minuteCell.Column)

blueCell.Interior.Color = RGB(173, 216, 230) ' Light blue color

' Insert APT1 value in the blue cell

blueCell.value = WS.Cells(arrivalRow, "A").value

' Insert FLT3 value two cells to the right of the blue cell

blueCell.Offset(0, 2).value = WS.Cells(arrivalRow, "B").value

' Insert Tail value five cells to the right of the blue cell

blueCell.Offset(0, 5).value = tailValue

' Adjust column width to fit FLT3 value

gatePlot.Columns(minuteCell.Column + 2).AutoFit

' Shade 9 cells to the right of the blue cell with the same color, up until column IL

Dim shadeRange As Range

Set shadeRange = blueCell.Offset(0, 1).Resize(1, 9)

If shadeRange.Column + shadeRange.Columns.Count - 1 <= gatePlot.Columns("IL").Column Then

shadeRange.Interior.Color = RGB(173, 216, 230) ' Light blue color

Else

' If shading exceeds column IL, adjust the range accordingly

Set shadeRange = gatePlot.Range(blueCell.Offset(0, 1), gatePlot.Cells(blueCell.row, "IL"))

shadeRange.Interior.Color = RGB(173, 216, 230) ' Light blue color

End If

End If

End If

End If

End If

End If

End If

End If

End If

Next arrivalRow

End Sub

Function RoundToNearest52(timeString As String) As String

Dim hours As Integer

Dim minutes As Integer

Dim roundedMinutes As Integer

' Parse the time string into hours and minutes

hours = val(Left(timeString, 2))

minutes = val(Right(timeString, 2))

' Round the minutes to the nearest 5-minute interval

roundedMinutes = WorksheetFunction.RoundDown(minutes / 5, 0) \* 5

' Handle the case where rounding to the next hour is required

If roundedMinutes = 60 Then

roundedMinutes = 0

hours = hours + 1

End If

' Format the rounded time

RoundToNearest5 = Format(TimeSerial(hours, roundedMinutes, 0), "hh:mm")

End Function

Function FindHourColumn5(sheet As Worksheet, hour As String) As Integer

Dim cell As Range

Dim foundColumn As Integer

Set cell = sheet.Rows(1).Find(What:=hour & ":", LookIn:=xlValues, LookAt:=xlWhole)

If Not cell Is Nothing Then

foundColumn = cell.Column

Else

foundColumn = 0

End If

FindHourColumn = foundColumn

End Function

Function FindMinuteCell5(row As Range, minute As String, startCol As Integer) As Range

Dim cell As Range

Dim foundCell As Range

For Each cell In row.Cells(startCol, row.Columns.Count)

If Right(cell.value, 2) = ":" & minute Then

Set foundCell = cell

Exit For

End If

Next cell

Set FindMinuteCell = foundCell

End Function

Function GetGateRow(gate As Integer) As Integer

Dim gateRow As Integer

If gate >= 1 And gate <= 10 Then

gateRow = gate \* 2

Else

gateRow = 0

End If

GetGateRow = gateRow

End Function