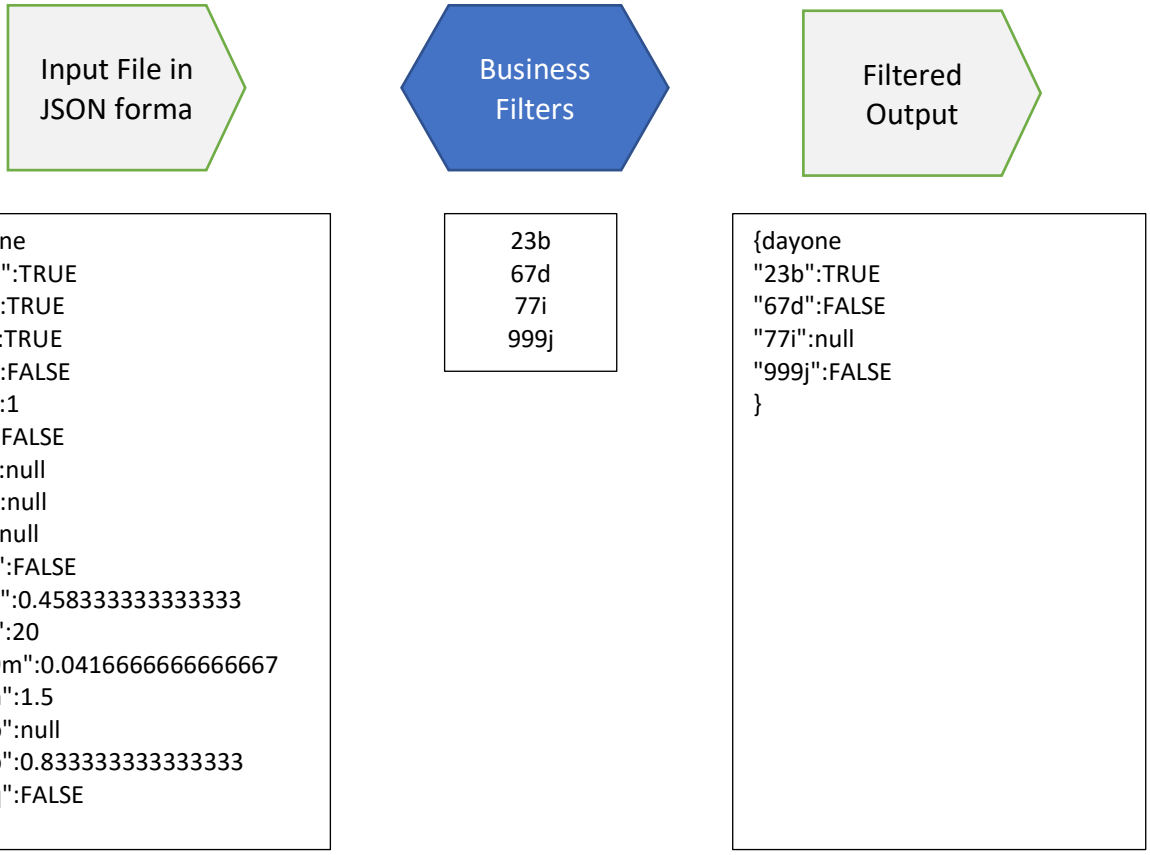
Situation Analysis

Business Context of the Problem

The raw data is related to payroll and contains details of an employee working for 7 days in week. A timesheet data dump is generated. The Timesheet gives the output in json format. I need to analyse the data whether for the time worked all the payments are correct. But the file generally has 16000 lines of data out of which 700 lines are relevant for my analysis. So I want to filter that data. Like for example I want to know what's the overtime, penalty, meal payment etc paid for each day, So these become my filters and these filters need to be applied for the entire week data and what I initially gave was just one day of data.



A dummy sample of expected input and the desired input is in the next page.
This will help in finding a pattern and formulating a code based solution within VBA environment.

Brief Note About JSON

A valid JSON document can be either of the following:

- An Object surrounded by curly braces, { and }
- An Array enclosed by brackets, [and]

Below are two samples corresponding to above :

```
{ "thisIs": "My first JSON document" }
```

```
[ "also",
  "a",
  "valid",
  "JSON",
  "doc"
]
```

Above Examples are from the book "JSON at Work" authored by Tom Marrs and Page no.4.

Sample of Input, Criteria and Expected Output

Input file		Business Rule		Output
{dayone		23b		{dayone
"120a":TRUE		67d		"23b":TRUE
"23b":TRUE		77i		"67d":FALSE
"45c":TRUE		999j		"77i":null
"67d":FALSE				"999j":FALSE
"89e":1				}
"90f":FALSE				{daytwo
"45g":null				"23b":TRUE
"66h":null				"67d":FALSE
"77i":null				"777i":null
"999j":FALSE				}
"678k":0.458333333333333				{daythree
"345l":20				"23b":TRUE
"6789m":0.0416666666666667				"67d":FALSE
"222n":1.5				"77i":null
"333o":null				"999j":FALSE
"609p":0.833333333333333				}
"909q":FALSE				
}				
{daytwo				
"120a":TRUE				
"23b":TRUE				
"45c":TRUE				
"67d":FALSE				
"89e":1				
"90f":FALSE				
"456g":null				
"666h":null				
"777i":null				
"9099j":FALSE				
}				
{daythree				
"120a":TRUE				
"23b":TRUE				
"45c":TRUE				
"67d":FALSE				
"89e":1				
"90f":FALSE				
"45g":null				
"66h":null				
"77i":null				
"999j":FALSE				
"678k":0.458333333333333				
"345l":20				
"6789m":0.0416666666666667				
"222n":1.5				
"333o":null				
"609p":0.833333333333333				
"909q":FALSE				
}				

Edge Case where no values match with criteria filters

Input file		Business Rule		Output
{dayfour		23b		{dayfour
"120a":TRUE		67d		}
"45c":TRUE		77i		
"89e":1		999j		
"89e":1				
"90f":FALSE				
"45g":null				
"333o":null				
"609p":0.833333333333333				
"909q":FALSE				
}				

Contextualising the problem in terms of VBA based code

The first major challenge is ensuring that the sequence of row values after filtering is same as in the original data. That means filtered values should correctly correspond to its parent day. We do not want a filtered string value to land up within a different day.

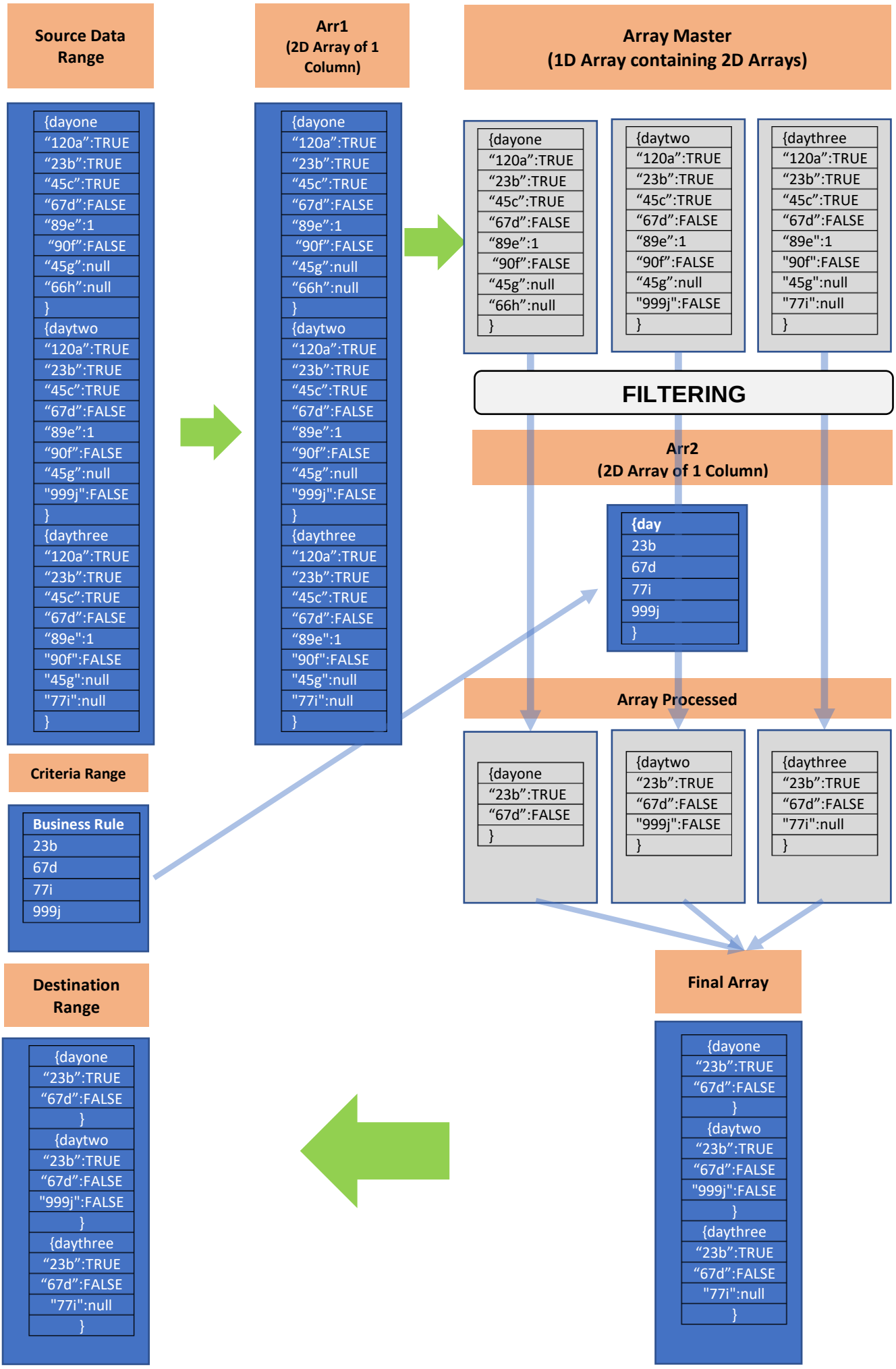
While there are multiple ways to do lookup and match in VBA environment, it does not necessarily ensure the sequence unless explicit coding is done.

To ensure the purity of the extracted required labels and no spill over at all stages, we ensure that day-wise data is transferred into a container of its own. This container corresponding to data pertaining to a single day will undergo filtering to return match values.

So, what happens is that if there are n days in the JSON data from which day-wise data will be transferred to n containers. Hence, there will be n containers created. Each container will undergo filtering resulting in n filtered containers.

The n filtered containers will then be aggregated to single column with the sequence maintained.

Visualising the VBA Solution



Going into details : Structure of the VBA code

Main Procedure that contains all processes from user input to transformations to final output	'M0_ReturnMatchValues()		
Taking User Input			
User to specify source Range	'M1_SetSourceRange(SrcRange)	User Input	SrcRange
User to specify Criteria Range	'M2_SetCriteriaRange(CritRange)	User Input	CritRange
User to specify Destination Range (first cell only)	'M3_SetDestinationRange(DesRange)	User Input	DesRange
Transforming the User Specified Ranges to Arrays for onward processing			
Converting Source Range into an Array named arr1	Code within main procedure - M0_ReturnMatchValues	SrcRange	arr1
Converting Criteria Range into an Array named arr2	'M4_CreateCriteriaArray(CritRange, arr2)	CritRange	arr2
Splitting the Arrays, Filtering and Aggregating back			
Extracting day-wise data (dictionary : Key Value pairings) and transferring into Master Array. Master Array is a 1D Array containing daywise 2D Arrays. ArrMaster(1) contains to values of dayone. ArrMaster(2) contains values of daytwo. ArrMaster(3(contains values of daythree.	'M5_CreateArrMaster(arr1, ArrMaster)	arr1	ArrMaster
Filtering the Master Array to create Array Processed Array Processed is thus a filtered version of Master Array and has same no. of dimensions. ArrProcessed (1) contains filtered values of ArrMaster(1). ArrProcessed(2) contains filtered values of ArrMaster(2). ArrProcessed(3) contains filtered values of ArrMaster(3).	'M6_1_ProcessArrMaster(arr2, ArrMaster, ArrProcessed)	arr2, ArrMaster	ArrProcessed
It receives a call from preceding procedure to apply filtering on each object within ArrMaster. This is where filtering at a granular level happens.	'M6_2_ProcessSingleDayArr(arrtemp, arr2)	arr2*	Arrtemp*
Transferring the Array Processed values into one column.	'M7_WidetoLong(FinArray, ArrProcessed)	ArrProcessed	FinArray
Copying the Final Array values into the Destination Range of worksheet			
The Final Array is outputted to Destination	>> Set DesRange = DesRange.Cells(1, 1).Resize(UBound(FinArray, 1), 1) >> DesRange.Value = Application.Transpose(FinArray)	DesRange	

* Check the coding to understand the mechanism.

Objects and Containers in the VBA code

The Range Objects used are as follows :

Variable	Descriptive Name	Dimensions	Detail
SrcRange	Source Range	User Input – Single column	
CritRange	Criteria Range	User Input – Single column	Should contain only filter labels without any header. Should not contain Dictionary Start string "{day" and Dictionary End String "}"
DesRange	Destination Range	User Input – Single Cell	

The Array containers used are as follows :

Variable	Descriptive Name	Dimensions	Detail
arr1	Arr1	2D Array	▪ Multiple Rows and One Column ▪ Contains SrcRange Values
arr2	Arr2	2D Array	▪ CritRange Row Count + 2 additional Rows ▪ One Column ▪ Contains CritRange Values ▪ Contains Strt String appended to beginning of CritRange value ▪ Contains End String appended to end of CritRange value
ArrMaster	Array Master	1D Array containing 2D Arrays	Master Array is a 1D Array containing day-wise 2D Arrays.
ArrProcessed	Array Processed	1D Array containing 2D Arrays	Array Processed is filtered version of Master Array.
FinArray	Final Array	1D Array containing all the filtered values in one column	Fin Array contains all the values aggregated from Array Processed sequentially.

Keeping track of the dimensions of array is important.
Below two thing depend on the Array dimensions :

- Redimming of Arrays
- number of loops for populating arrays

Important considerations for successfully running the code :

Raw Data Format

- Raw data consists of a single column in dictionary format where each dictionary starts with "{day***" and ends with "}".
- A new dictionary cannot start without the previous dictionary being ended.
- A new dictionary cannot end without a corresponding Start String
- The above implies Dictionary Start string should encounter Dictionary End String somewhere down the line and not Start String. Same implies for Dictionary End String.

Criteria Range

- The Criteria Range has to be selected without any header and should contain only the labels for filtering.
- An Initial Criteria Array is created from Criteria Range Values
- The code shall append Dictionary Start String "{day" to the beginning of the Initial Criteria Array.
- The code shall append Dictionary End String "}" to the end of the Initial Criteria Array.

VBA Code

1. Module Level Declarations

```
Option Explicit
Const Str2strt As String = "{day"
Const Str2end As String = "}"
```

2. M0_ReturnMatchValues() : Main Procedure that contains all processes from user input to transformations to final output

```
Sub M0_ReturnMatchValues()

'M0_ReturnMatchValues()
'M1_SetSourceRange(SrcRange)
'M2_SetCriteriaRange(CritRange)
'M3_SetDestinationRange(DesRange)
'M4_CreateCriteriaArray(CritRange, arr2)
'M5_CreateArrMaster(arr1, ArrMaster)
'M6_1_ProcessArrMaster(arr2, ArrMaster, ArrProcessed)
'M6_2_ProcessSingleDayArr(arrtemp, arr2)
'M7_WidetoLong(FinArray, ArrProcessed)

'Taking User inputs on Source Data Range, Criteria Range and Destination Range
Dim SrcRange As Range
Dim CritRange As Range
Dim DesRange As Range

'Selecting Source Range
Call M1_SetSourceRange(SrcRange)


'Selecting Criteria Range
Call M2_SetCriteriaRange(CritRange)

'Selecting Destination Range
Call M3_SetDestinationRange(DesRange)

On Error GoTo 0

'Creating Array arr1 containing Source Range Values
'2D Array of multiple Rows and one column
Dim arr1() As Variant
ReDim arr1(1 To SrcRange.Rows.Count, 1 To SrcRange.Columns.Count)
arr1 = SrcRange.Value


'Creating Array containing Criteria Range Values
'2D Array of multiple Rows and one column
Dim arr2() As Variant
Call M4_CreateCriteriaArray(CritRange, arr2)


'Creating Master Array containing 1D Arrays
'Each 1DArray inside Master Array corresponds to values of single day
'Creating ArrayContainer of 1D Arrays
Dim ArrMaster() As Variant      '1D Array
```

```
Call M5_CreateArrMaster(arr1, ArrMaster)

'Filtering each of the single day Arrays within Master Array
'Creating ArrayContainer of 1D Arrays
Dim ArrProcessed() As Variant
Call M6_1_ProcessArrMaster(arr2, ArrMaster, ArrProcessed)

'Creating 1D Array containing all the filtered values collated to one column
'Wide to Long format
Dim FinArray() As Variant
Call M7_WidetoLong(FinArray, ArrProcessed)

Set DesRange = DesRange.Cells(1, 1).Resize(UBound(FinArray, 1), 1)
DesRange.Value = Application.Transpose(FinArray)

MsgBox "Task completed."

End Sub
```

2. User to specify source Range

```
Private Sub M1_SetSourceRange(ByRef SrcRange As Range)
'Selecting Source Range

'Important line otherwise do while loop will not start
On Error Resume Next

Do While SrcRange.Columns.Count <> 1
   Set SrcRange = Nothing
   Set SrcRange = Application.InputBox(Title:="Select Source Data Range", _
   Left:=2880, Top:=100, _
   Prompt:="Select Range without headers:", Type:=8)

   If SrcRange Is Nothing Then
   'MsgBox Prompt:="You have chosen to cancel.", Title:="Cancelling the Text processing"
   Exit Sub
   ElseIf SrcRange.Columns.Count <> 1 Then
   MsgBox "Select range of one column only.", Title:="Select Range Properly"
   End If

Loop

End Sub
```

3. User to specify Criteria Range

```
Private Sub M2_SetCriteriaRange(ByRef CritRange As Range)
'Selecting Criteria Range

'Important line otherwise do while loop will not start
On Error Resume Next

Do While CritRange.Columns.Count <> 1
   Set CritRange = Nothing
   Set CritRange = Application.InputBox(Title:="Select Criteria Range", _
   Left:=2880, Top:=100, _
   Prompt:="Select Range without headers:", Type:=8)
```

```
   If CritRange Is Nothing Then
   'MsgBox Prompt:="You have chosen to cancel.", Title:="Cancelling the Text processing"
   Exit Sub
   ElseIf CritRange.Columns.Count <> 1 Then
   MsgBox "Select range of one columns only.", Title:="Select Range Properly"
   End If

Loop

End Sub
```

4. User to specify Destination Range (first cell only)

```
Private Sub M3_SetDestinationRange(ByRef DesRange As Range)
'Selecting Destination Range first cell
'Userinput of single cell required

'Important line otherwise do while loop will not start
On Error Resume Next

Do While DesRange.Columns.Count <> 1
   Set DesRange = Nothing
   Set DesRange = Application.InputBox(Title:="Select Destination Range:", _
   Prompt:="Select Output Range", Left:=2880, Top:=100, Type:=8)
   'Left:=2880, Top:=100, Tried adjusting position of Msgbox
   '1440 twip to an inch

   If DesRange Is Nothing Then
   'MsgBox Prompt:="You have chosen to cancel.", Title:="Cancelling the Text processing"
   Exit Sub
   ElseIf DesRange.Columns.Count <> 1 Then
   MsgBox "Select range of one columns only.", Title:="Select Range Properly"
   End If
Loop
End Sub
```

5. Converting Source Range into an Array named arr1

```
Private Sub M4_CreateCriteriaArray(ByRef CritRange As Range, ByRef arr2() As Variant)



ReDim arr2(1 To CritRange.Rows.Count, 1 To CritRange.Columns.Count)
arr2 = CritRange.Value

Dim arrsize As Long
arrsize = UBound(arr2) + 2



Dim arr2temp() As Variant
ReDim arr2temp(1 To arrsize, 1 To 1)
   arr2temp(1, 1) = Str2strt

   Dim i As Long
   For i = 2 To UBound(arr2) + 1
   arr2temp(i, 1) = arr2(i - 1, 1)
   Next i
   arr2temp(UBound(arr2) + 2, 1) = Str2end
```

```
   ReDim arr2(1 To UBound(arr2temp), 1 To 1)
arr2 = arr2temp

End Sub
```

6. Converting Criteria Range into an Array named arr2

```
Private Sub M5_CreateArrMaster(ByRef arr1() As Variant, ByRef ArrMaster() As Variant)

Dim Array_Day() As Variant
Dim RowDay As Long

Dim Str2strt As String
Dim Str2end As String

Str2strt = "{day"
Str2end = "}"

'Declaring and Initializing landmarks for traversal through arr1
Dim rowid As Long
Dim rowstrt As Long
Dim rowend As Long

rowstrt = 0
rowend = 0


Dim i As Long, j As Long 'For traversing through arr1
Dim m As Long, n As Long 'For traversing through arr1
Dim p As Long 'populating ArrMaster
p = 0

rowid = 1
For i = 1 To UBound(arr1, 1)

'If i = 127 Then MsgBox "Loop violated"

   If Not IsEmpty(arr1(i, 1)) And InStr(1, CStr(Trim(arr1(i, 1))), Str2strt, vbTextCompare) = 1 Then
      If rowend < rowstrt Then
         MsgBox "Error in source Data." & Chr(10) & _
         "Check row number :" & rowid + 1 & "."
         Exit Sub
         'GoTo ErrHandler  'Trapping Error
      End If
      rowstrt = rowid
   ElseIf Not IsEmpty(arr1(i, 1)) And InStr(1, CStr(Trim(arr1(i, 1))), Str2end, vbTextCompare) = Len(CStr(Trim(arr1(i, 1))))
Then

      If rowstrt < rowend Then
      MsgBox "Error in source Data." & Chr(10) & _
      "Check row number :" & rowid + 1 & "."
      'GoTo ErrHandler  'Trapping Error
      Exit Sub
      End If

'       If i = 126 Then
'       MsgBox "Err Trapped"
```

```
'       End If

        rowend = rowid
        p = p + 1

        'Creating temporary Array_Day to be transfered to ArrMaster
        RowDay = rowend - rowstrt + 1
        ReDim Array_Day(1 To RowDay)

        m = 1
        For j = rowstrt To rowend
        Array_Day(m) = arr1(j, 1)
        m = m + 1
        Next j

        ReDim Preserve ArrMaster(1 To p)
        ArrMaster(p) = Array_Day

    End If

rowid = rowid + 1
Next i


'ErrHandler:
'MsgBox "Error in source Data."

End Sub
```

7. Filtering the Master Array to create ArrProcessed

```
Private Sub M6_1_ProcessArrMaster(arr2() As Variant, ArrMaster() As Variant, ArrProcessed() As Variant)

Dim arrtemp() As Variant
ReDim arrtemp(1 To 1)
Dim p As Long, i As Long
p = 1

For i = 1 To UBound(ArrMaster)
    ReDim arrtemp(1 To UBound(ArrMaster(i), 1))
    arrtemp = ArrMaster(i)
    Call M6_2_ProcessSingleDayArr(arrtemp, arr2)
    ReDim Preserve ArrProcessed(1 To p)
    ArrProcessed(p) = arrtemp
    p = p + 1
Next i

End Sub
```

8. A sub-procedure that is called from Sub M6_1_ProcessArrMaster. It is the procedure that actually does the filtering.

```
Private Sub M6_2_ProcessSingleDayArr(ByRef arrday() As Variant, ByRef arr2() As Variant)

'arr2 is the criteria array
Dim Str1 As String, Str2 As String
Dim i As Long
Dim p As Long
Dim matchrow As Long
```

```
matchrow = 0

Dim arrtemp() As Variant


For i = 1 To UBound(arr2, 1)
   For p = 1 To UBound(arrday, 1)
      If IsEmpty(arr2(i, 1)) = False And IsEmpty(arrday(p)) = False Then
         If Trim(CStr(arr2(i, 1))) <> "" And Trim(CStr(arrday(p))) <> "" Then
            Str1 = Trim(CStr(arrday(p)))
            Str2 = Trim(CStr(arr2(i, 1)))
            If InStr(1, Str1, Str2) >= 1 Then
               matchrow = matchrow + 1
               ReDim Preserve arrtemp(1 To matchrow)
               arrtemp(matchrow) = Str1
            End If
         End If
      End If
   Next p
Next i

ReDim arrday(1 To UBound(arrtemp))
arrday = arrtemp

End Sub
```

9. Penultimate step where the filtered values residing within a numbered sequence of Arrays within Array Processed Container is aggregated within a 1D Array

```
Private Sub M7_WidetoLong(ByRef FinArray() As Variant, ByRef ArrProcessed() As Variant)

Dim i As Long, j As Long
Dim m As Long, n As Long
m = 1

Dim arrtemp() As Variant
ReDim arrtemp(1 To 1)

For i = 1 To UBound(ArrProcessed)
For j = 1 To UBound(ArrProcessed(i))
   ReDim Preserve arrtemp(1 To m)
   arrtemp(m) = ArrProcessed(i)(j)
   m = m + 1
Next j
Next i

ReDim FinArray(1 To UBound(arrtemp))
FinArray = arrtemp

End Sub
```

10. The final step is copying the FinArray Values by transposing into the Destination Range. This step is carried out within the main procedure.

```
Set DesRange = DesRange.Cells(1, 1).Resize(UBound(FinArray, 1), 1)
DesRange.Value = Application.Transpose(FinArray)

MsgBox "Task completed."
```

Two main pointers :

- A 1D Array by nature is aligned along a row. It has a horizontal orientation. Hence, while copying to worksheet, we will transpose it to copy it to single column.
- Further, the destination range object has to be resized to the same dimensions as in the FinArray prior to the copying operation.