**44-599: Introduction to Data Visualization**

**Worksheet for drawing Radial Stacked bar chart using Tableau**

In this worksheet we are going to draw radial stacked bar chart using tableau. In order to use tableau you can download it by clicking the following link

<https://www.tableau.com/products/desktop/download?signin=8a29da95a0d4bc38b8df82bbf64a08a4>

You can use it for a 15 day free trail provided by tableau or if you have a product key just enter it.(northwest has provided a product key for students who have enrolled in ITM course). After entering your details you are ready to use it.

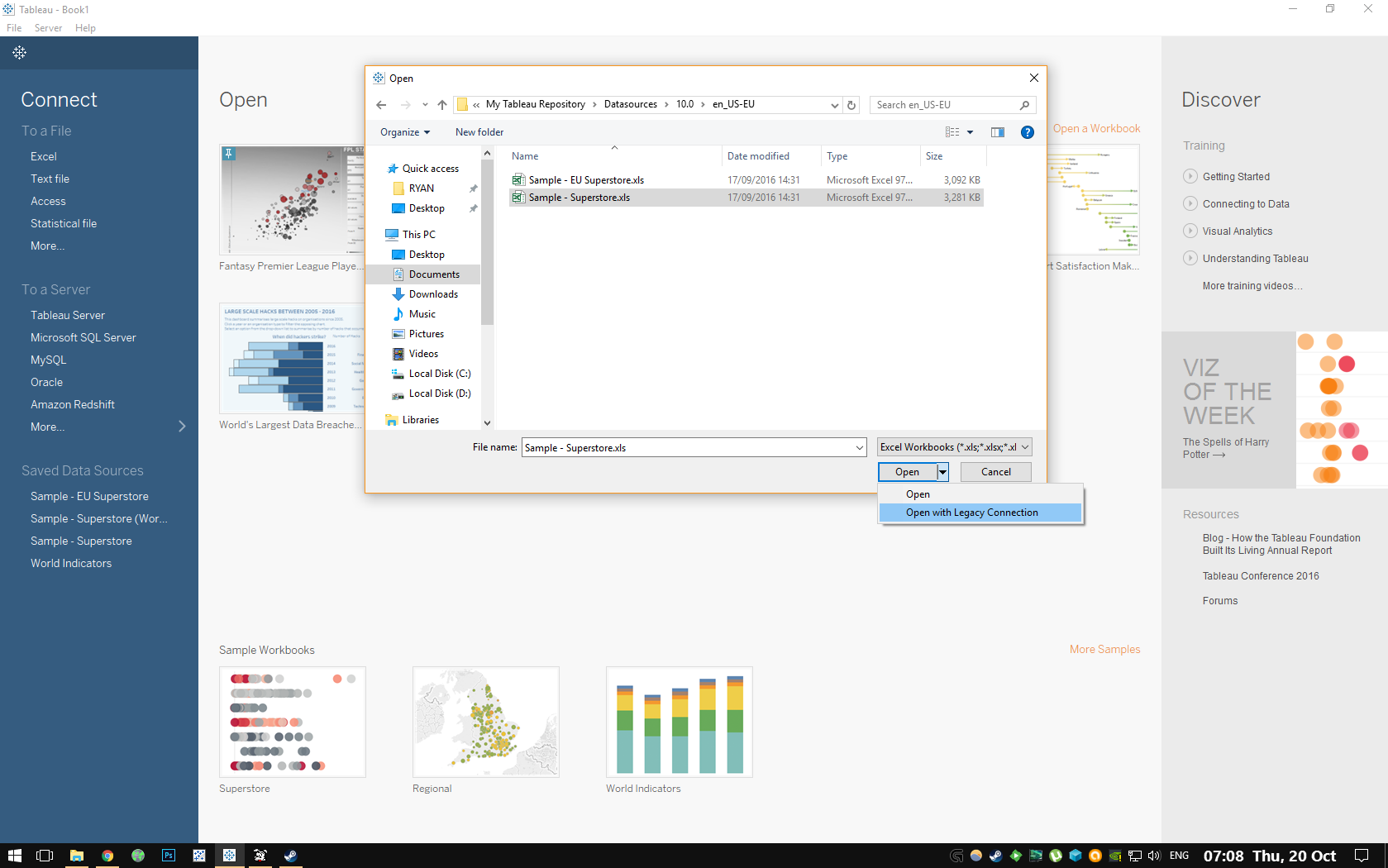
The following Excel sheet contains Sample Orders information about various products that were bought by customers in different states of U.S in the years from 2013 to 2016.The dataset contains 9995 rows



**Goal 1:** **To visualize a radial stacked bar chart showing the sales of different categories of products in different states of U.S.A.**

1. After downloading the data set just rename it to its original name (after downloading the file name may change)

2. Open tableau and to connect data, click on the excel tab (as we are using a excel file to load our data) and go the destination folder and open the sheet using the legacy connection as we are going to write some custom SQL queries.



3. After connecting to excel sheet, we are going to write a custom SQL query. To write SQL query click on new custom SQL query and copy the following SQL query in to the dialog box provided

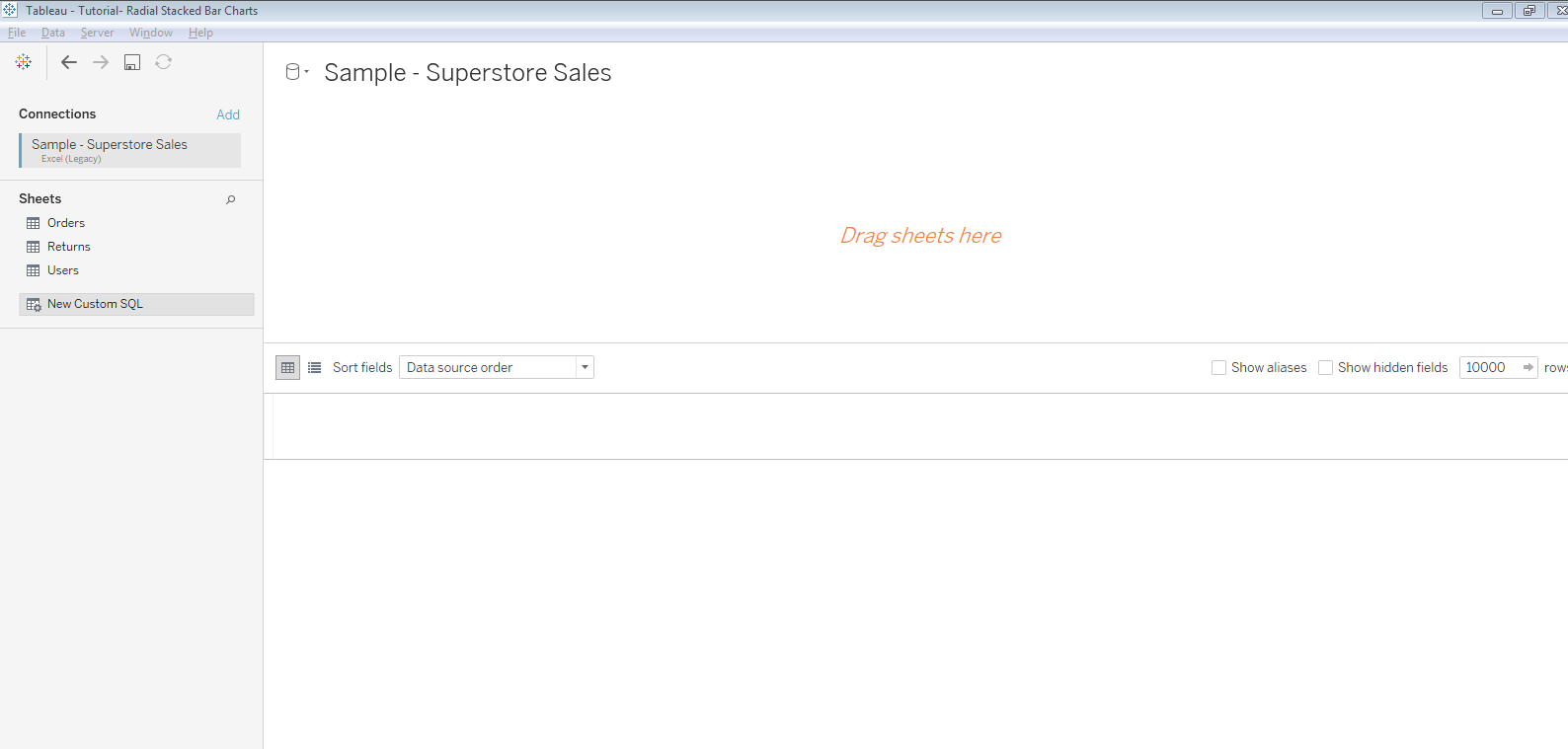
select 1 as [PathOrder], \*

from [Orders$]

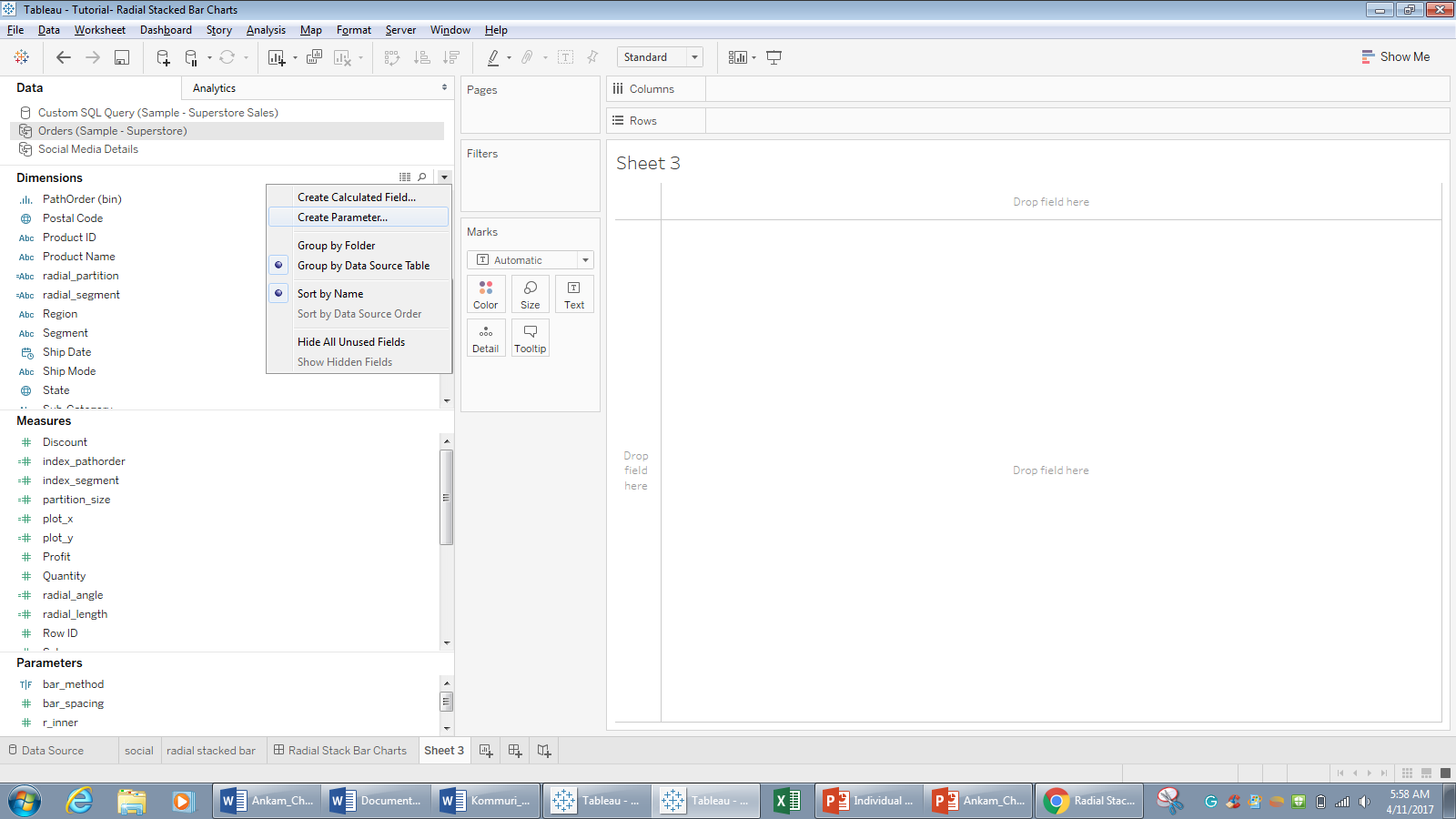
UNION ALL

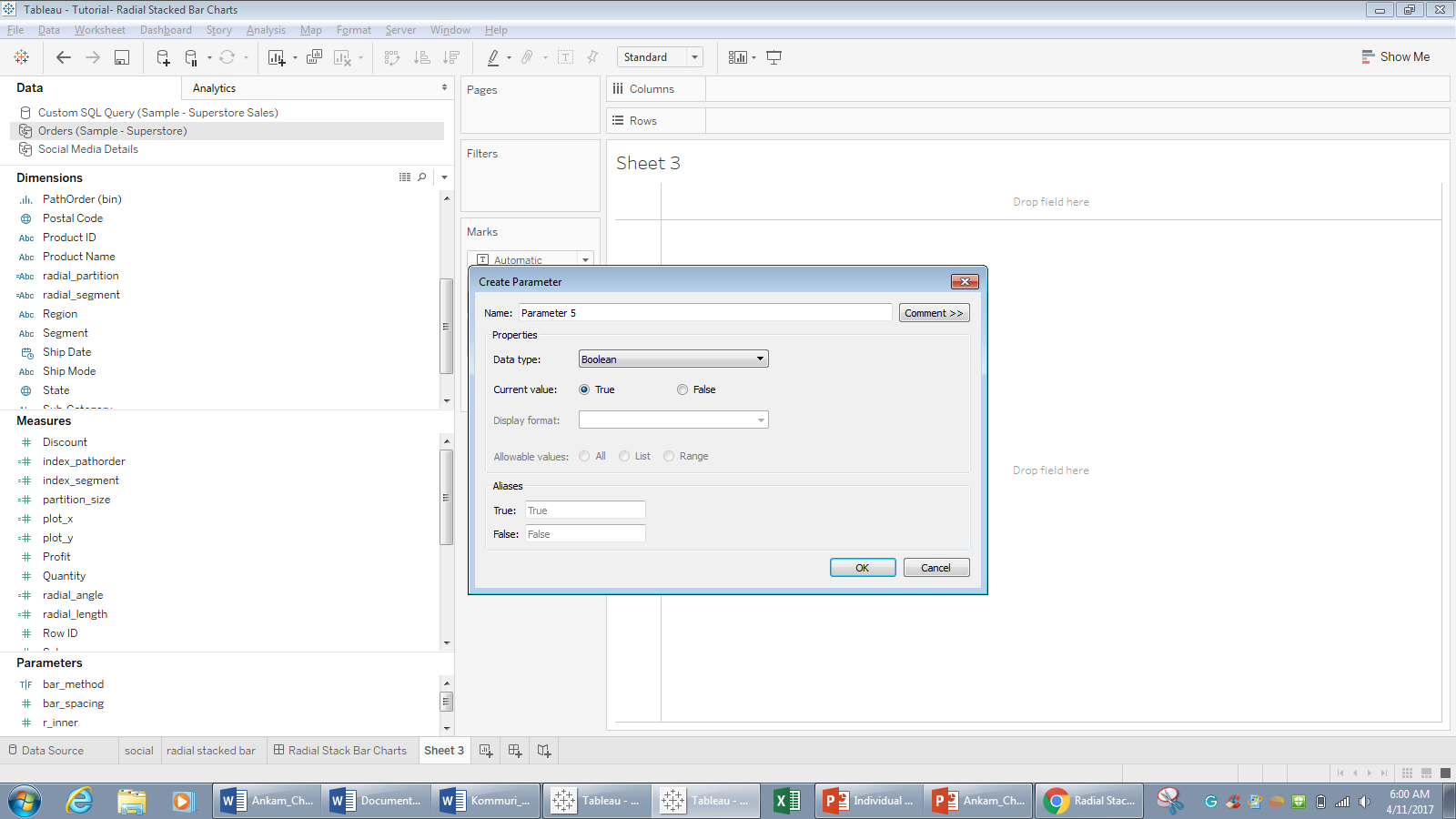
select 5 as [PathOrder], \*

from [Orders$]



4. After writing the query we will get the dataset and go to the sheet and here we are going to add some parameters, create the 4 following parameters. To add a parameter click on a dropdown menu just beside the Dimensions tab you will find a Create Parameter field, just click on it.





5. Create the following parameters in our data sheet

**5.1 bar\_method (boolean)** – This parameter will drive whether our chart displays the partitions as absolute values or as a % of the segment total. I set the alias on true to “Absolute Values” and the alias on false to “% of Segment Total”.

**5.2 bar\_spacing (float**) – This one will determine the amount of white space between each of our bars (our value = 0.04).

**5.3 r\_inner (float)** – Determines the inner radius of our circle – i.e. the amount of white space in the middle of the circle (our value = 3).

**5.4 r\_outer (float)** – Determines the outer radius of our circle – i.e. how long our longest bar will be in relation to r\_inner (our value = 5).

6. After creating these parameters we are going to add some calculated fields in our data set.To create the calculated fields just click the “Create calculated field” above the “create parameter” field and add the following calculated fields

**radial\_segment** (dimension) – This field will determine our bar separation, We will use [State], meaning we ’ll get a bar for each individual state in the data.

[State]

**radial\_partition** (dimension) – This field will determine how each bar is partitioned for stacking, we will use [Category], meaning we’ll get a bar stack for each category in the data.

[Category]

**index\_pathorder** (measure) – This field will be used to allow our table calcs to perform calculations on rows that don’t exist in our data (we’re going to mimic them later with a bin).

index()

**index\_segment** (measure) – This field will be used to determine how many bars we need to draw and from that determine the correct angles for each bar.

index()

**value\_field** (measure) – This field will determine which value gets summarised, we force it to equal 0 on rows where we set PathOrder to 1 (in the custom SQL) as we don’t want to double count – Change [Sales] to any other measure you’d like to base your bars on.

if [PathOrder] = 1 then 0 else [Sales] end

**value\_partition** (measure) – Simply sums up our value field at the deepest level required (state>category) and inserts the value on every row in the data (including fake rows in our bin).

window\_sum(sum([value\_field]))

**value\_segment** (measure) – As above, but sums up our value field at the segment (State) level rather than the partition (State>Category) level.

WINDOW\_SUM(sum([value\_field]))

**value\_segment\_max** (measure) – This field determines the maximum value\_segment in the view – which will be used for generating our Absolute Values view of the data.

WINDOW\_MAX([value\_segment])

**value\_limit** (measure) – This field will be used to set where each bar should end, either in relation to value\_segment for % of segment view or value\_segment\_max for the absolute values view – based on the selection in our bar\_method parameter.

case [bar\_method]

when true then [value\_segment\_max]

else [value\_segment]

end

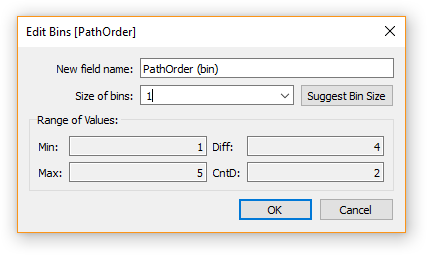
The PathOrder field we created is going to be our point order, but the problem is we only have 2 entries in our dataset, 1 and 5. We can create the missing points using a bin, so let’s do that!

Right-click the [PathOrder] field in the dimension window.

Goto Create > Bin.

Setup per image below and click ok.

pathorder-bin



**partition\_size** (measure) – Right here is, this field will determine the height of each partition in our chart

IF([index\_pathorder]=2 or [index\_pathorder]=3)

THEN RUNNING\_SUM([value\_partition])/[value\_limit]

else (RUNNING\_SUM([value\_partition])-[value\_partition]) / [value\_limit]

end

**radial\_length** (measure) – We have our partition size, but we need to generate the co-ordinates of each of our points, taking into consideration our r\_inner and r\_outer parameters. This field determines, for each point, the distance from the center of the circle.

[r\_inner] +

([partition\_size]\* ([r\_outer]-[r\_inner]))

**radial\_angle** (measure) – We also need to determine the angle at which our points should sit in relation to our circle we do this based on the index of the current segment in relation to the total number of segments we need to draw.

IF([index\_pathorder]=3 or [index\_pathorder]=4)

THEN

[index\_segment] +1 – [bar\_spacing]

else

[index\_segment] + [bar\_spacing]

end

\* (1/window\_max([index\_segment])) \* 2 \* 3.14159265359 + (3.14159265359/2)

Now we have our radial\_length and our radial\_angle, all that’s left to do is calculate our x and y co-ordinates using some basic trigonometry.

**plot\_x (measure)**

[radial\_length] \* COS([radial\_angle])

**plot\_y (measure)**

[radial\_length] \* SIN([radial\_angle])

7. Now to create our chart drag the plot\_x in to the columns field and plot\_y into the rows field and add the radial segment radial partition and the pathorder bins in to the the marks card.

Now begin to set the settings as follows

**partition\_size**

Compute Using: Specific Dimensions (radial\_segment & radial\_partition)

At the level: Deepest

Restarting every: radial\_segment

**index\_pathorder**

Compute Using: Specific Dimensions (PathOrder (bin))

**value\_partition**

Compute Using: Specific Dimensions (PathOrder (bin))

**value\_segment\_max**

Compute Using: Specific Dimensions (PathOrder (bin), radial\_partition & radial\_segment)

At the level: Deepest

Restarting every: None

**value\_segment**

Compute Using: Specific Dimensions (radial\_partition & PathOrder (bin))

At the level: Deepest

Restarting every: None

**radial\_angle**

Compute Using: Specific Dimensions (PathOrder (bin), radial\_partition & radial\_segment)

At the level: Deepest

Restarting every: None

**index\_segment**

Compute Using: Specific Dimensions (radial\_segment, radial\_partition & PathOrder (bin))

At the level: radial\_segment

Restarting every: None

8. Follow the same for the plot\_y field and the radial stacked bar chart is displayed in the sheet and the chart should look as follows

