Okay, the screenshot and your new JSON configuration provide crucial clues!

The image shows a **donut chart** (which is a variation of a pie chart, specifically variablepie in your code attempts something similar but not quite standard donut) where *each individual CTRL\_ID* (or perhaps each unique combination based on your previous setup) is being rendered as its own slice. This confirms that **QuickSight is still sending detailed, unaggregated data** to the Highcharts visual.

The label "Slice" appears because your Highcharts mapping (series.data.map) isn't correctly assigning the name property that Highcharts uses for labels and tooltips.

**Let's Fix This:**

**1. CRITICAL STEP: Fix the QuickSight Visual Setup**

You *must* change the fields in the QuickSight visual's field wells for this to work. Highcharts cannot perform the aggregation you want automatically with the mapping you're using.

* **Go to the Field Wells for the Highcharts Visual:**
  + **Group/Color by:** Make sure **ONLY** AGGREGATED\_STATUS is in this field well. Remove DATE, remove CTRL\_ID. AGGREGATED\_STATUS will now be **index 0**.
  + **Value:** Make sure **ONLY** the distinct count of control IDs is here.
    - Use the calculated field: distinct\_count(CTRL\_ID) (create this if you haven't)
    - OR drag CTRL\_ID to the Value well and set its aggregation to Count Distinct.
    - This distinct count measure will now be **index 1**.
* **Why this is essential:** This forces QuickSight to aggregate the data *before* sending it to Highcharts. It will send only 3 rows (or fewer if some statuses have zero counts):
  + ["green", <distinct count for green>]
  + ["yellow", <distinct count for yellow>]
  + ["red", <distinct count for red>]

**2. Update the Highcharts JSON Configuration**

Now, use a much simpler Highcharts JSON that expects this pre-aggregated data. Forget the complex map function for now; it's not needed if QuickSight sends the right structure. Use a standard pie chart type.

{

"chart": {

"type": "pie" // Use standard pie

},

"title": {

"text": "DISTRIBUTION BREAKDOWN: PERCENTAGE SPLIT BY COMPLIANCE STATUS"

},

"tooltip": {

// Show Status Name, the count (y value), and the percentage

"pointFormat": "{series.name}: <b>{point.y}</b> ({point.percentage:.1f}%)<br/>Status: <b>{point.name}</b>"

},

"plotOptions": {

"pie": {

"allowPointSelect": true,

"cursor": "pointer",

// Optional: If you want a donut chart, uncomment the next line

"innerSize": "50%",

"colors": [

// Define colors. Highcharts tries to match these to the point names

// if `colorByPoint` is false, or assigns them sequentially if true.

// For explicit matching, structure data as {name: 'Red', y: value, color: '#FF0000'} below.

// Let's rely on `colorByPoint` for simplicity first. Remove this `colors` array for now

// or ensure the order matches your data (e.g., Green, Yellow, Red if sorted that way).

],

"dataLabels": [

{

// Optional: Connector line label (shows category name by default)

"enabled": true,

"distance": 20,

// Filter based on percentage

"filter": {

"operator": ">",

"property": "percentage",

"value": 10

}

},

{

// Percentage label inside/outside slice

"enabled": true,

"distance": -40, // Adjust position as needed (-ve = inside)

"format": "{point.percentage:.1f}%",

"style": {

"fontSize": "1.2em",

"fontWeight": "bold",

"color": "white",

"textOutline": "none"

},

// Filter based on percentage

"filter": {

"operator": ">",

"property": "percentage",

"value": 10

}

}

],

"showInLegend": true

}

},

"series": [

{

"type": "pie", // Ensure series type matches chart type

"name": "Control Count", // Name for the series (appears in tooltip)

"colorByPoint": true, // Assign colors from default palette or `plotOptions.pie.colors` sequentially

"data": {

// Simple array mapping for pre-aggregated data

"type": "array",

"values": [

// Column 0 (AGGREGATED\_STATUS) -> Point Name

["getColumn", 0],

// Column 1 (Distinct Count) -> Point Value (y)

["getColumn", 1]

]

/\* --- Alternative Data Structure for Explicit Color Mapping ---

// If colorByPoint doesn't give the right colors consistently,

// you might need to structure data points explicitly.

// This usually requires processing \*within\* Highcharts if QuickSight

// doesn't let you add a 'color' column easily.

// Example (conceptual - harder to achieve with direct QuickSight mapping):

"values": [

{ "name": "Red", "y": ["getColumn", 1, {"where": {"field": ["getColumn",0], "eq": "Red"}}], "color": "#FF0000" },

{ "name": "Green", "y": ["getColumn", 1, {"where": {"field": ["getColumn",0], "eq": "Green"}}], "color": "#008000" },

{ "name": "Yellow", "y": ["getColumn", 1, {"where": {"field": ["getColumn",0], "eq": "Yellow"}}], "color": "#FFA500" }

]

// Stick to the simple ["getColumn", 0], ["getColumn", 1] for now.

\*/

}

}

]

}

content\_copydownload

Use code [with caution](https://support.google.com/legal/answer/13505487).Json

**Explanation of Changes and Why:**

1. **QuickSight Fields (Most Important):** Correcting the fields ensures QuickSight sends aggregated data (3 rows maximum), not detailed data (12 rows in your example).
2. **chart.type: "pie":** Changed back to a standard pie chart. Add innerSize: "50%" in plotOptions.pie if you want the donut appearance. variablepie is for varying slice radii, which isn't needed here.
3. **Simplified series.data:** The ["map", ...] structure is overly complex for this goal *if* QuickSight sends aggregated data. The simple {"type": "array", "values": [["getColumn", 0], ["getColumn", 1]]} tells Highcharts:
   * Take the value from the first column (index 0, which is AGGREGATED\_STATUS after fixing field wells) as the slice name.
   * Take the value from the second column (index 1, which is distinct\_count(CTRL\_ID)) as the slice value (y).
4. **colorByPoint: true:** This tells Highcharts to assign a unique color to each data point (slice) from its default color theme or the plotOptions.pie.colors array if provided. Since you should only have 3 points ("red", "yellow", "green"), this usually works well.
5. **Tooltip:** Updated to show the series name (Control Count), the actual distinct count (point.y), the percentage, and the status name (point.name).
6. **Data Labels:** Kept your desired percentage formatting and the filter (>10%). Adjusted positioning and styling slightly.

**Troubleshooting Colors:**

* If "red" slices aren't red, "green" aren't green, etc., using colorByPoint: true:
  + Make sure the actual data values in your AGGREGATED\_STATUS column are exactly "red", "yellow", "green" (case-sensitive). Highcharts might match these names to standard colors sometimes.
  + Provide the plotOptions.pie.colors array: ["#008000", "#FFA500", "#FF0000"]. The order matters here. Highcharts will assign these colors based on the order of the data points it receives from QuickSight (which might be alphabetical or based on value). You might need to experiment with the color order.
  + The most robust (but complex) way is explicit mapping (like the commented-out alternative series.data structure), but try the simpler methods first.

First, **fix the QuickSight field wells**. Then, use the updated, simpler JSON configuration. This should result in a standard pie (or donut) chart with 3 slices representing the aggregated counts for each status.