# Grouping Functions

**Alchemer Dashboard is currently waitlist only.** Visit this page **to learn more about Dashboard or join the waitlist!**

What if you want to aggregate a value by a specific attribute (for example, show revenue by product)? This is known as a grouped aggregation, but some people call it a pinned measure or level-based measure. You can do this for any aggregation using the grouping functions.
Each of the grouping functions accepts a measure and one or more optional attributes:

```
formula (measure, [attribute, attribute, ...])
```

Only the measure value is required. If you supply both a measure and an attribute, the function returns the aggregate of the measure grouped by the attribute(s). You should experiment with only a measure and then with an attribute to see which output best meets your use case.

## List of Group Functions

Group aggregation functions have names with formats like `group_<aggregation>`. The group aggregation functions are the following:

| Function | Description |
|---|---|
| `group_aggregate` | Takes a measure and optional attributes and filters. Used to aggregate measures with different granularities and filters than the columns used in the search. |
| `group_average` | Takes a measure and one or more attributes. Returns the average of the measure grouped by the attribute(s).<br>`group_average (revenue, customer region)` |
| `group_count` | Takes a measure and one or more attributes. Returns the count of the measure grouped by the attribute(s).<br>`group_count (revenue, customer region)` |
| `group_max` | Takes a measure and one or more attributes. Returns the maximum of the measure grouped by the attribute(s).<br>`group_max (revenue, customer region)` |

| Function | Description |
| --- | --- |
| `group_min` | Takes a measure and one or more attributes. Returns the minimum of the measure grouped by the attribute(s).<br>`group_min (revenue, customer region)` |
| `group_stddev` | Takes a measure and one or more attributes. Returns the standard deviation of the measure grouped by the attribute(s).<br>`group_stddev (revenue, customer region)` |
| `group_sum` | Takes a measure and one or more attributes. Returns the sum of the measure grouped by the attribute(s).<br>`group_sum (revenue, customer region)` |
| `group_unique_count` | Takes a column name and one or more attributes. Returns the number of unique values in a column, grouped by the attribute(s).<br>`group_unique_count ( product, supplier)` |
| `group_variance` | Takes a measure and one or more attributes. Returns the variance of the measure grouped by the attribute(s).<br>`group_variance (revenue, customer region)` |

# Flexible Aggregation

The `group_aggregate` function gives you more control over aggregation and filtering.

See Flexible aggregation to learn more about specifying `query_groups` with this formula.

# Limitations of Group Aggregation Functions

Group aggregation functions have the following limitations:

- You can't run AI Highlights analysis on a visualization that contains a group aggregation function.
- Alchemer Dashboard doesn't support aggregate table summaries for formulas that have group aggregates and are conditional.
- You can't run a `vs` query that also contains a group aggregation function.
- You can't run a group function on a group function. If you would like to create a nested group aggregation function, you can do so by first saving the Chart with the first level of the group function as a View, then using the View as the data source for a second Chart with the second

level of the group function.

- Any group aggregation function that returns a measure at the row level is implicitly reaggregated with a `sum` to match the level of detail defined in the Search bar. To circumvent this behavior, define the aggregation type within the formula, for example, `sum(revenue) + group_sum(tax)`. This behavior extends to `if... then..else` statements: `if(revenue = 0) then group_sum(revenue) else 0` would be reaggregated with a `sum`, while `if (sum(revenue) = 0) then group_sum(revenue) else 0` would not.

## Related Articles