Defining Metrics Reference Sheet

In GoodData, metrics are the numerical values displayed in reports. They represent aggregations of facts and attributes.

Most metrics are summaries of individual business records, or *fact values*. Such metrics are formed by aggregating facts with functions like SUM, MAX, MIN, and AVG.[1] Attributes can also be aggregated into metrics by taking a COUNT of the number of values they have.

SUM	Returns the sum of all fact values.
MAX	Returns the largest fact value.
MIN	Returns the smallest fact value.
AVG	Returns the average fact value.
COUNT	Returns the number of unique values belonging to some attribute.

Create basic metrics in the Report Editor's *What* pane by clicking **+ Add New Metric**. From there you can define new metrics on the fly in the Simple Metric Editor and add them directly to a report.

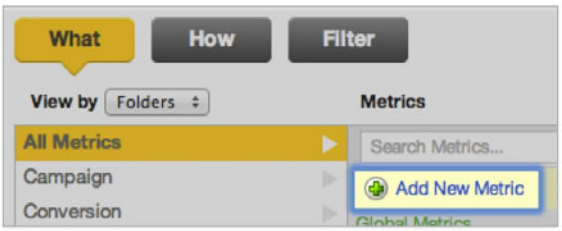

Figure 1 Add New Metric link in the Report Editor

Defining Custom Metrics

While all metrics are aggregations of some fact or attribute, that's not to say that aggregating is all that goes into defining a metric. With the Custom Metric Editor you can write advanced metrics using GoodData's data query language: MAQL (Multi-dimension Analytical Query Language).

Accessing the Custom Metric Editor

Access the advanced metric editors from the *Manage* page by selecting **Metrics** and then **Create Metric**. You can also access the advanced editors from the Report Editor's *What* pane by clicking **(Advanced)** next to *Add New Metric*.

When the Metric Editor is first opened, you are presented with four options. The first three are metric wizards that provide a structure for creating specific types of advanced metrics with MAQL. Clicking **Custom metric** allows you complete flexibility to define metrics from scratch using MAQL.

While MAQL commands can be entered directly into the main text field, particular project elements like facts, attributes, and variables must be added from the Project Element Sidebar.

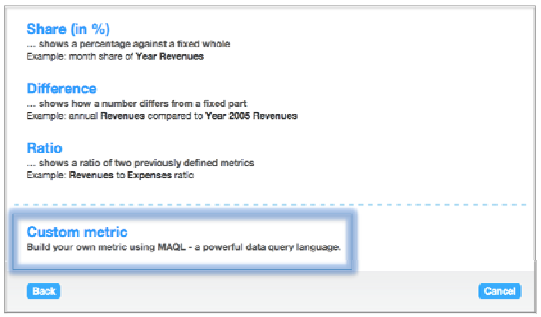

Figure 2 Navigating to The Custom Metric Editor

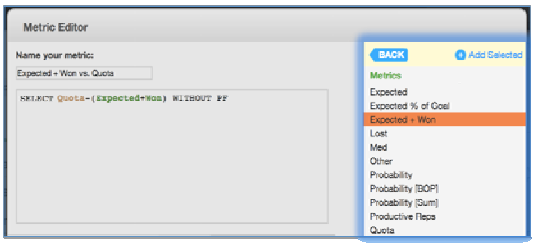

Figure 3 Add project elements to your MAQL metric definitions using the sidebar.

[1] Running total functions like RUNSUM, RUNMAX, RUNMIN, and RUNAVG can also be used to create cumulative metrics. **Note**, however, that running totals can only be broken across *time* (date) *attributes* in reports.

Custom Metric Editor Tips

- Use the following color codes to distinguish different project elements: **Facts**, **Metrics**, **Attributes**, **Attribute Values**, and **Variables**.
- The MAQL directory tabs toward the bottom of the editor provide a handy MAQL syntax reference. Double click code snippets to insert them into the MAQL field.
- Metrics created through the Manage section are global by default and can be used by other project users in other reports. Name, tag, and categorize global metrics carefully to allow teammates to find and reuse them!

Defining Custom Metrics with MAQL

In the following MAQL syntax summary, we'll begin by reviewing how to write the basic aggregation functions in MAQL, before going on to explore how MAQL can be used to approach more advanced use cases. For a comprehensive overview, see the **MAQL Reference Guide**.

Aggregations: SUM | MAX | MIN | AVG

All metric definitions begin with the `SELECT` command. Think of this command in terms of making a request to the database and selecting the data you'd like returned. Basic aggregation functions apply an operation to a fact in parentheses:

```
SELECT SUM(Payment)
```

Aggregations: COUNT

COUNT aggregations work in a similar way to those described above, but may also require you to specify the data set where the count should take place. This information appears as a second parameter in the parentheses.

```
SELECT COUNT(Employees, Facts_of_Payroll)
```

Arithmetic Operations: + - * /

Predefined metrics (like *Expenses*, *Salary*, and *Profit,* below) can also be nested within parent metrics and manipulated with arithmetic operations.

```
SELECT Expenses + Salary

SELECT Profit * 0.03
```

Numeric Variables

Numeric variables that have been assigned to particular users can be introduced into metrics to automatically tailor metric computations to each report viewer. This could be useful for using a single report to display total commissions to sales reps with different commission rates.

```
SELECT Amount_Won * Commission_Rate_Variable
```

Mathematical Functions

In addition to aggregation functions, mathematical functions like absolute value, signum, and square root can be applied to constants and other metrics.

```
SELECT ABS(-5)
```
returns 5

```
SELECT SIGN(-3)
```
returns -1

```
SELECT SQRT(33.6)
```
returns 5.796550698

Running Total Functions

Running totals represent the sum of all prior values and current value of a metric, broken down by time (date) attributes in a report. GoodData supports running sums (RUNSUM), averages (RUNAVG), minimums (RUNMIN), and maximums (RUNMAX).

```
SELECT RUNSUM(Sales)
```

Filtering

Use the WHERE keyword to filter a metric based on certain conditions. Conditions can contain relational operators (**=** **<>** **<** **<=** **>** **>=**) or relational keywords (**IN**, **NOT IN**, **BETWEEN**, **NOT BETWEEN**, as well as time keywords such as **THIS**, **NEXT**, and **PREVIOUS**. Multiple conditions can be combined with logical expressions like **NOT**, **OR**, and **AND**.

```
SELECT Revenues WHERE Year = 2006

SELECT Payment WHERE Date = THIS - 1

SELECT Revenues WHERE Year=2006 AND Month=5
```

You can also use filtered variables to define dynamic filter conditions that change depending on the user at hand.

```
SELECT #Leads WHERE Var_Industry
```

Conditional Statements

IF THEN ELSE conditional statements allow you to return one of two possible values or perform one of two possible computations, depending on whether some condition is met.

```
SELECT IF SUM(Amount) >= AVG(Amount) THEN 10 ELSE 0 END
```

Use CASE statements for complex conditionals that contain three or more conditions.

```
SELECT CASE WHEN SUM(Amount) > SUM(Lost) AND SUM(Amount) - SUM(Lost)
> 100000 THEN 2, WHEN SUM(Amount) > SUM(Lost) AND SUM(Amount) -
SUM(Lost) < 100000 THEN 1 ELSE 0 END
```

Use IFNULL to define a replacement value (second parameter) to be inserted in place of any null value returned by some metric expression (first parameter).

```
SELECT IFNULL(SUM(Amount), 0)
```

Time Transformations

Time transformations alter the time period to which a metric value relates, which is useful for time over time comparisons (e.g., month over month, or quarter over quarter). A second parameter indicates the number of time periods from the present that the transformation should span.

```
SELECT Payment FOR Next(Quarter,3)
```

Ranking Functions

Use the RANK function to sequentially rank all of a report's values or to rank within report sub-groups, specified by the WITHIN keyword. Ranking can be carried out in ascending or descending order.

```
SELECT RANK(Amount) ASC WITHIN(Year(Closed))
```

TOP(n) and BOTTOM(n) are ranking filters that rank the top or bottom n or n% of values and then exclude all other report values from being displayed.

```
SELECT Amount WHERE TOP(5%) OF (Amount)
```

Overriding Report Attributes and Filters

MAQL keywords can customize how report-level attribute and filter configurations affect metrics.

The BY keyword sets a minimum level of granularity ("aggregation floor") by which a metric can be broken down—even if a report attribute would otherwise serve to break down the metric further.

```
SELECT Payment BY Year
```

Several additional formulations include the clauses BY ALL, BY ALL IN ALL OTHER DIMENSIONS, and BY ALL IN ALL OTHER DIMENSIONS EXCEPT FOR.

```
SELECT Payment BY ALL IN ALL OTHER DIMENSIONS EXCEPT FOR Date
```

To override all report level filters defined in the active report, use the WITHOUT PARENT FILTER clause (which can also be written "WITHOUT PF").

```
SELECT Payment WITHOUT PARENT FILTER
```

Ready to learn more? **For articles, resources, & more:** help.gooddata.com