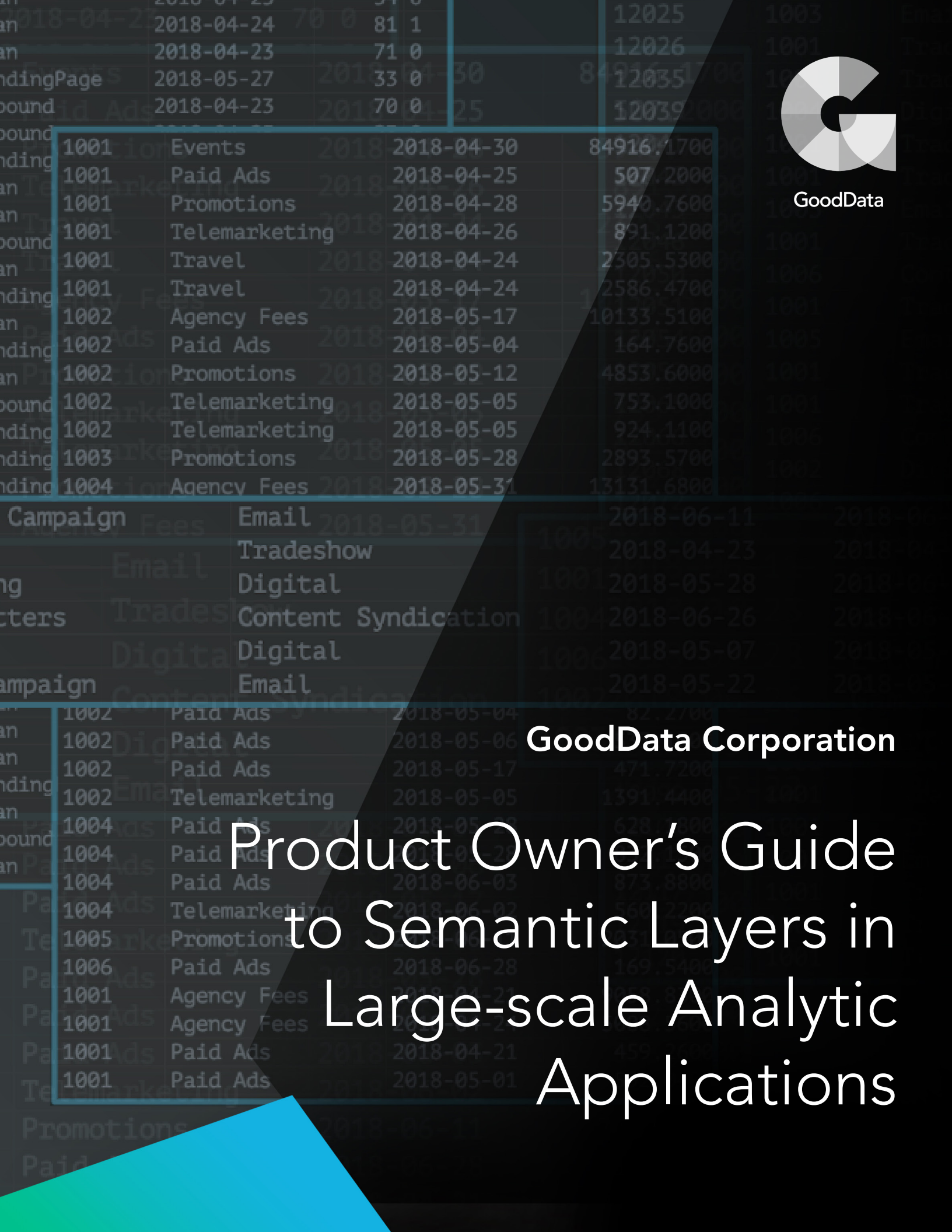


GoodData

GoodData Corporation

# Product Owner's Guide to Semantic Layers in Large-scale Analytic Applications

# Introduction

**If you're a product owner of a data-driven application, you're likely looking to integrate embedded analytics into your application in a way that makes analytical insights usable and available to every user. However, not all embedded analytics solutions are created equal in terms of maintainability, clarity, and power, and a very important component of the solution is the semantic layers.**

## How this eBook will help you

As analytics becomes more pervasive, your user base will expand—or may have already expanded—dramatically. Increasingly, when building analytic applications, you'll need to think about how you'll translate technical representations of data into business needs.

There are three main sections to this eBook. In the first section, we'll review the components of the semantic layer—logical data models, measures and metrics, and insights—and explore some concrete examples of each of these components. In the second section, we'll discuss best practices for creating each of these semantic layer components. In the third section, we'll explain how key features of the GoodData platform can help you successfully follow these best practices as you go forward. As a result, you'll be more knowledgeable about semantic layers and be more familiar with best practices for building semantic layers for large-scale analytics applications.

## Semantic layer overview

The purpose of the semantic layer is to convert complex data into clear, easy-to-understand insights that business people and others can use to make decisions in their day-to-day work. Semantic layers are essentially a shield between the user and the sheer volume of data that's been collected. They're designed to make things simple and intuitive for business users by abstracting away the complexity of the underlying data source, reducing the need for training, and allowing information to be easily understood.

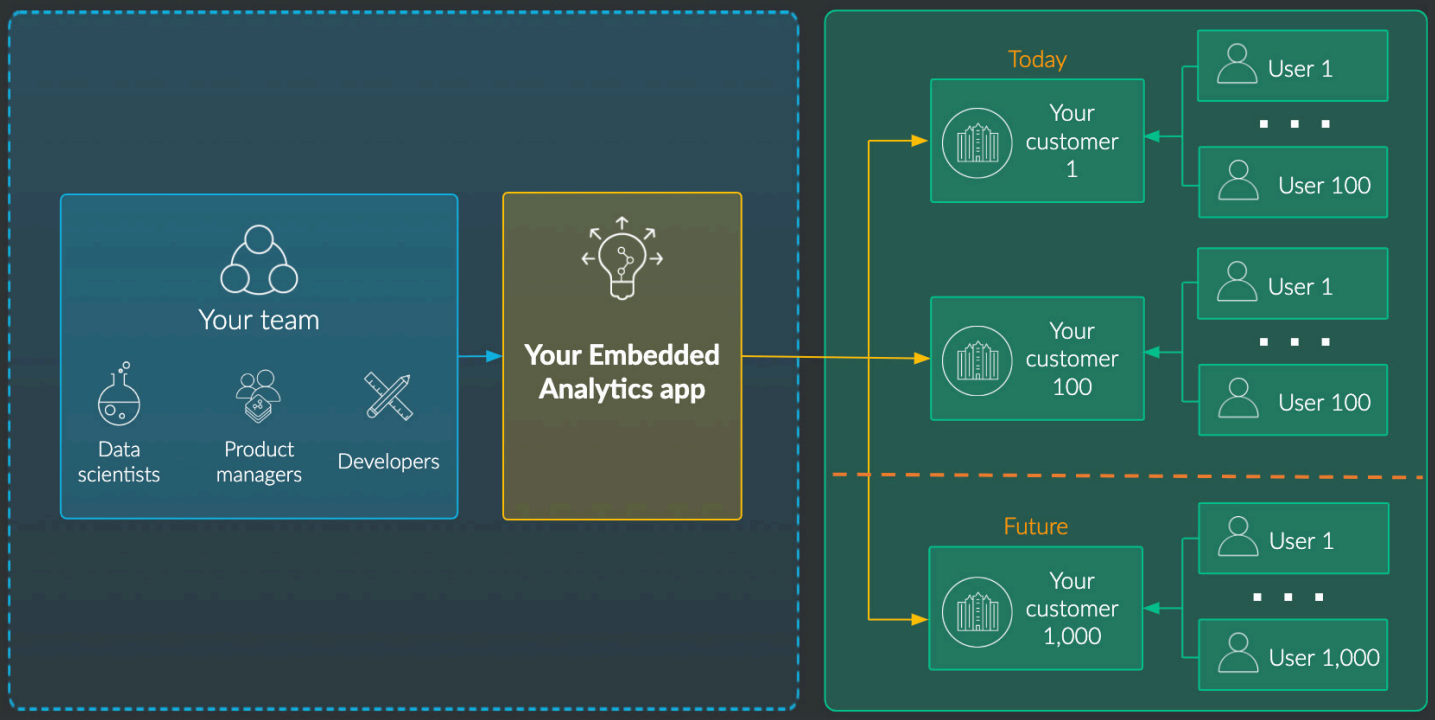
For a sales department, dozens of data points may be collected, most of which would make no sense to the user when seen individually. Even if that data has been cleaned, prepared, and put into a data warehouse, it can still be very difficult to understand when you take into account the different interpretations of the data, rules, and conventions that may be at play. With semantic layers, those dozens of concrete data points are synthesized into a familiar concept—like "Q1 sales performance"—that is easier for the user to understand.

For a small-scale departmental BI solution used internally, like for a sales and marketing team, internal BI with a few prepackaged reports and a data analyst to create additional reports on request is usually sufficient. However, if your goal is to deliver analytics at scale, then a robust semantic layers should be well considered.

Imagine that a hospitality enterprise resource planning (ERP) solution provider offers a web portal and a mobile application with embedded analytics for hundreds of small- to medium-sized hotels and chains. Each of these hotels has dozens of different employees who use the analytics capabilities of the mobile app for tracking inventory and pricing in different areas of the hotel, and there are also about 10 hotel managers who use analytics to optimize the hotel's running costs.

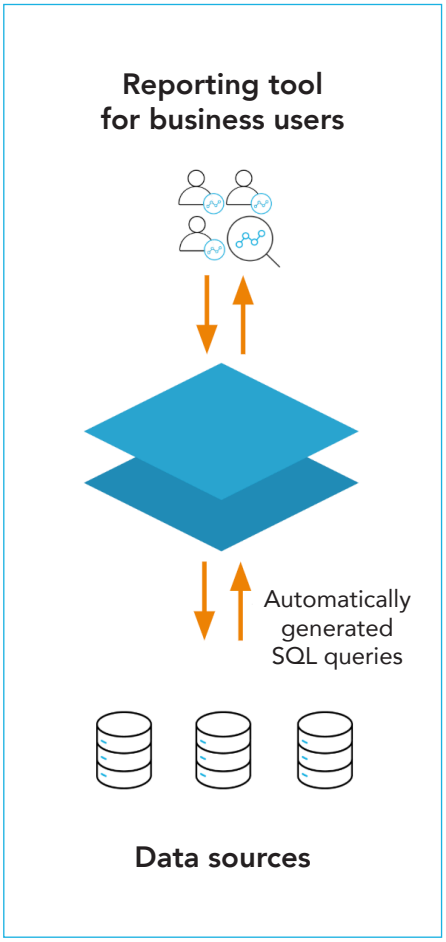
Before too long, the provider decides to expand to the enterprise market and adds enterprise hospitality companies with thousands of locations worldwide. As a result, the provider's average hotel customer has grown to one that has thousands of employees and hundreds of managers using the embedded analytics capabilities—and the game has changed as a result.

By creating a strong semantic layers as the basis forin the analytic application, the ERP solution provider will be able to help those thousands of users—most of whom will not have a technical background—make sense of the data that's being collected, ultimately helping the solution provider's customers improve business performance.



# Benefits of semantic layers

When the sheer number of end-users who need to understand and interpret insights grows, it's much easier to justify using semantic layers especially when you consider that most broad-based analytics products serve business users that don't know how to analyze or interpret data. As the use of embedded analytics becomes more widespread, semantic layers will ensure that non-technical line-of-business users don't need to understand the complexity of the schemas, tables, and columns in data sources. As a result, these users will be able to conduct ad-hoc data explorations in a way that makes sense to them, without having to dive into the underlying data structure or programming or data language.




By enabling users to answer questions themselves, semantic layers can help remove the bottlenecks that occur when users have to rely on other teams to respond to their queries. While this is great for the user, it has the added benefit of freeing up your IT resources, so excess time and energy aren't devoted to answering users' queries, which may vary in complexity and difficulty in answering.

Another benefit is that semantic layers can help encapsulate the author's knowledge and serve as a single source of truth. With data that is properly handled and insights that come from a single point of truth, users can always be sure that they're getting the correct answer from a reliable source, which leads to better data governance. And in the event that the original author leaves the organization, the semantic layer can hold onto their knowledge, storing details about where the data lives and how to query it correctly so things can continue to run smoothly.

From a security perspective, organizations may need to know who accessed which data and when or may need to allow only authorized users to see certain data. Semantic layers can track all of that information and confirm authorization, so organizations can be confident that they're in compliance with any requirements, legal or otherwise.

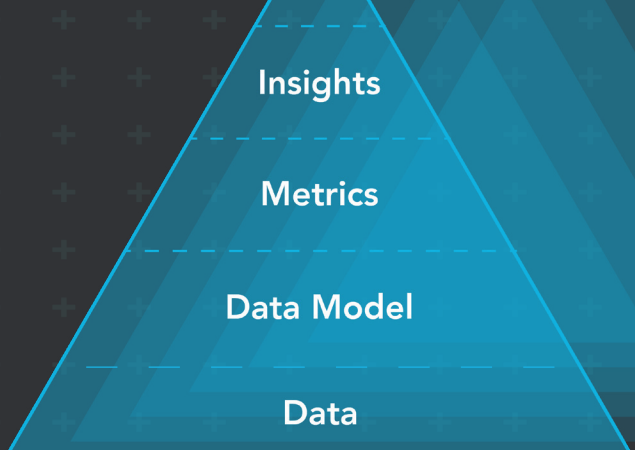
# Making the business case for semantic layers

On a basic level, having a large-scale application means that many developers, data scientists, and insight creators are working collaboratively. By using semantic layers, everyone who works on the product is referencing the same common set of definitions.

Semantic layers are also useful for the demands of agile products, which must constantly evolve to stay relevant as customer needs change and necessitate rapid product releases as analytics capabilities continue to improve. And, because semantic layers act as an intermediary, they ensure that any changes made are implemented to all of your customers in a governed manner so a common understanding of how to interpret data persists as the product changes.

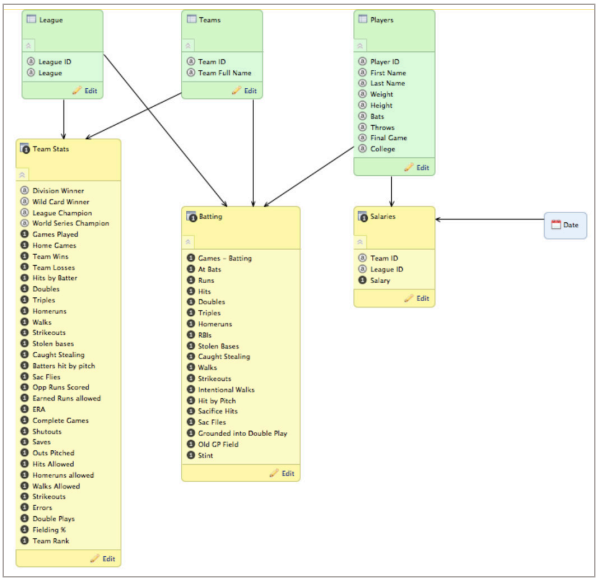
And finally, semantic layers benefit providers by protecting data quality and your brand, because your customers expect the analytics you provide to be bulletproof. One wrong calculation or an error means losing trust in data—and your product—and could ultimately lead to churn or low usage and adoption of analytics.

# Components of semantic layers

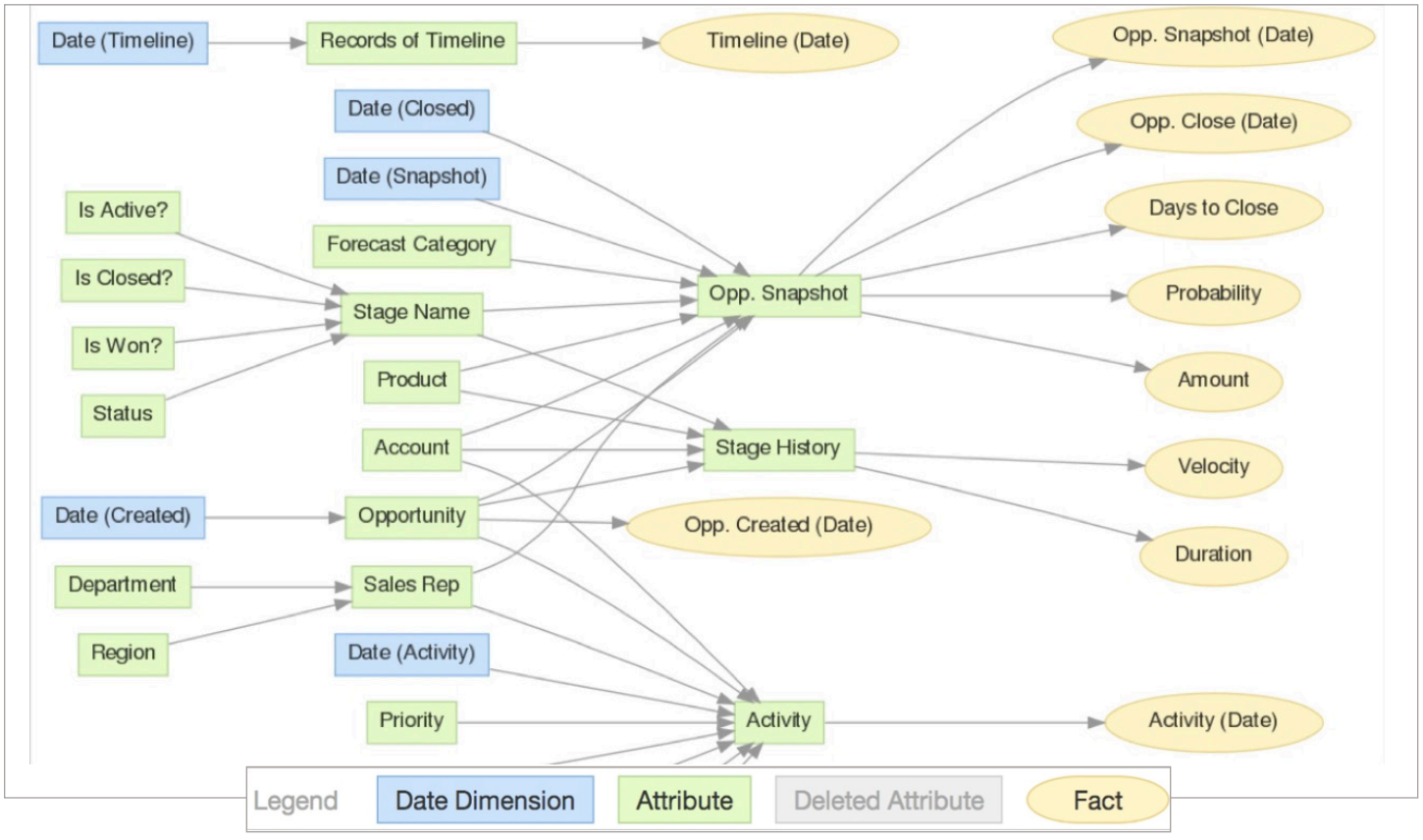



**Logical data models**

Put simply, logical data models provide an abstract—either visual (diagrammatic) or programmatic (code)—representation of a domain of information as it is understood at a point in time. If you reference the graphic above, you'll also see that they serve as a foundation and support structure for other semantic layer components such as measures and insights.



To get a bit more technical, logical data models describe facts, or numerical information around transactions, and dimensions, or aspects of the transaction. For example, a fact could be the price paid for an item, and a dimension might be the product category of the item sold. They're also optimized for answering questions in a multitude of ways, are independent of underlying data structures, and are a useful abstraction from database and data warehouse technologies.




**Visual representation of an LDM inside the GoodData platform**

**Measures and metrics**
Above the logical data model are metrics and measures, which hold the knowledge of how data is interpreted, combined, and used. The data model plus the metrics are what form the semantic layer and attach meaning to the data.

Measures are numeric aggregations of business transactions or events that have occurred over a period of time and for which a record is stored. A measure can be something as simple as the sum of all sales for last month. Other kinds of measures might be number of leads generated, outside temperature, miles driven, quantity sold, or total weight. Measures are typically segmented (sliced) or filtered by different dimensions.

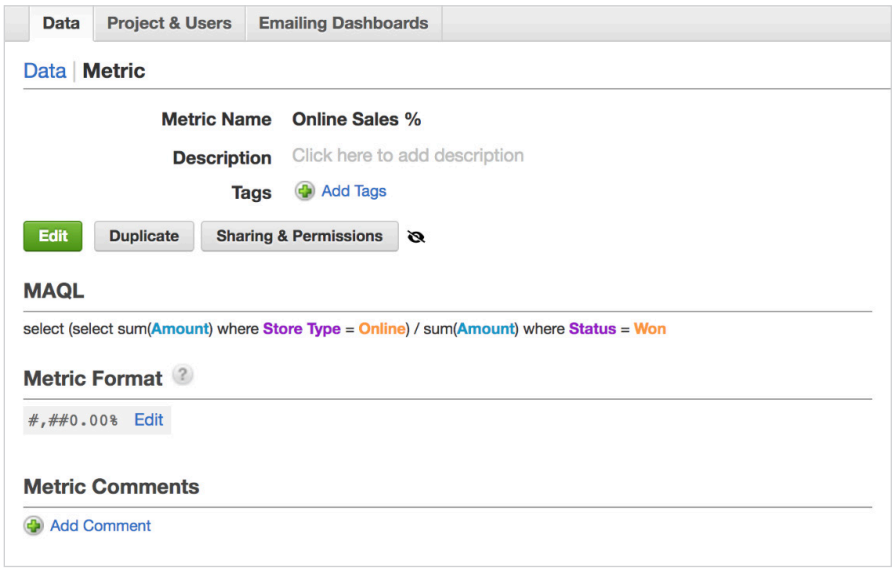
A key aspect of a measure is the grain of the underlying fact table in the logical data model. Did the data engineer store every sales transaction? Or only the sum of transactions each day? Or perhaps only the sum of sales each month broken out by product number and geographic region. The underlying grain will dictate how detailed the answers to our queries can be.

Another key aspect of measures is that they don't necessarily reveal whether a particular campaign or initiative is performing well because they lack context. The last marketing campaign might have generated 100 qualified leads, but is that good? Answering that question requires a metric, which comprises one or more measures that are typically evaluated against a goal or a standard. For example, a metric might be called "Qualified Leads per Month." This metric can show if the number of leads is increasing—or decreasing—and compare it to a standard, like 50 qualified leads per month.

In addition to simple sums and averages, metrics can also be calculated. Going back to the sales example, it's possible to track the percentage of online sales to retail sales with a metric definition like this:

$$\text{Online Sales \%} = \frac{\text{Sum of Sales where Store Type is Online}}{\text{Sum of Sales}}$$

In GoodData, the same metric is defined using GoodData's MAQL as follows:




**Example screenshot from GoodData metric editor using MAQL**

Metrics can also include other metrics. For example, a business may want to compute total costs by combining the cost of sales (one metric) with inventory and carrying costs (two more metrics). Read more on creating and using metrics with GoodData.

The semantic layer not only contains the definitions for measures and metrics, but it is responsible for abstracting all the lower-level details from the dashboards, reports, and analytical applications that use them. For example, a dashboard that the executive team uses to make decisions might contain a metric called "Year-to-Date Profit on Retail Sales."

The semantic layer descends through the layers of metrics and measures that comprise such things as inventory expenses, invoices, and overhead formulas, and will issue the detailed queries needed against the fact tables to render the final number.

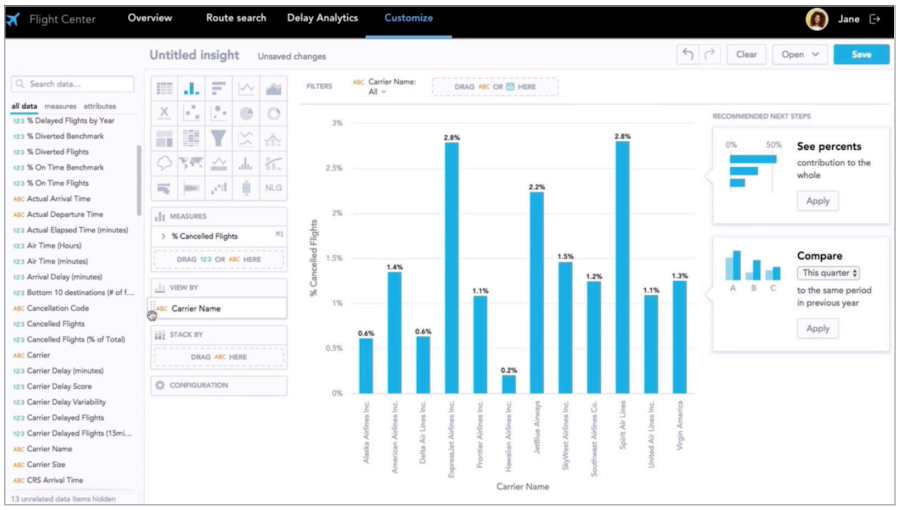
**Benefits of using measures and metrics**
Both data providers and data consumers benefit from using metrics. On the provider side, business or domain analysts can define a metric once and then use it many times over to provide a range of insights. Users of analytic applications can reuse metrics to look at information from different angles, or even deconstruct a metric and recombine the components in a different way to create a customized insight.

Another benefit is that metrics represent, in one place and for all to see, how the business uses this data to measures itself. A data engineer, an analyst, an application developer, a front-line worker, and a business manager are all able to see the same numbers. Metrics are maintained centrally, such that multiple applications can access them, and they can be replicated and tailored to the needs of individual departments, business units, clients, or customer groups. And, when changes need to be made, they can be rolled out quickly, thanks to clear dependencies and distinct layers of abstraction.

For the organization, metrics create a "single source of truth" for how the business computes its performance measures. In this way, everyone from a data analyst to a mobile application developer to a product manager can leverage the same metrics and obtain the same results.

# Insights

Insights are at the top of the pyramid, and these are the key performance indicators or charts or other visualizations that are used to uncover important things in the data or make clear something that wasn't clear before. From all of the work going on behind the scenes on the product side, the end result—from the user's perspective—is an actionable insight that helps them make better business decisions or suggests the next best action to take.




Example: In GoodData, insights are created using the [Analytical Designer]

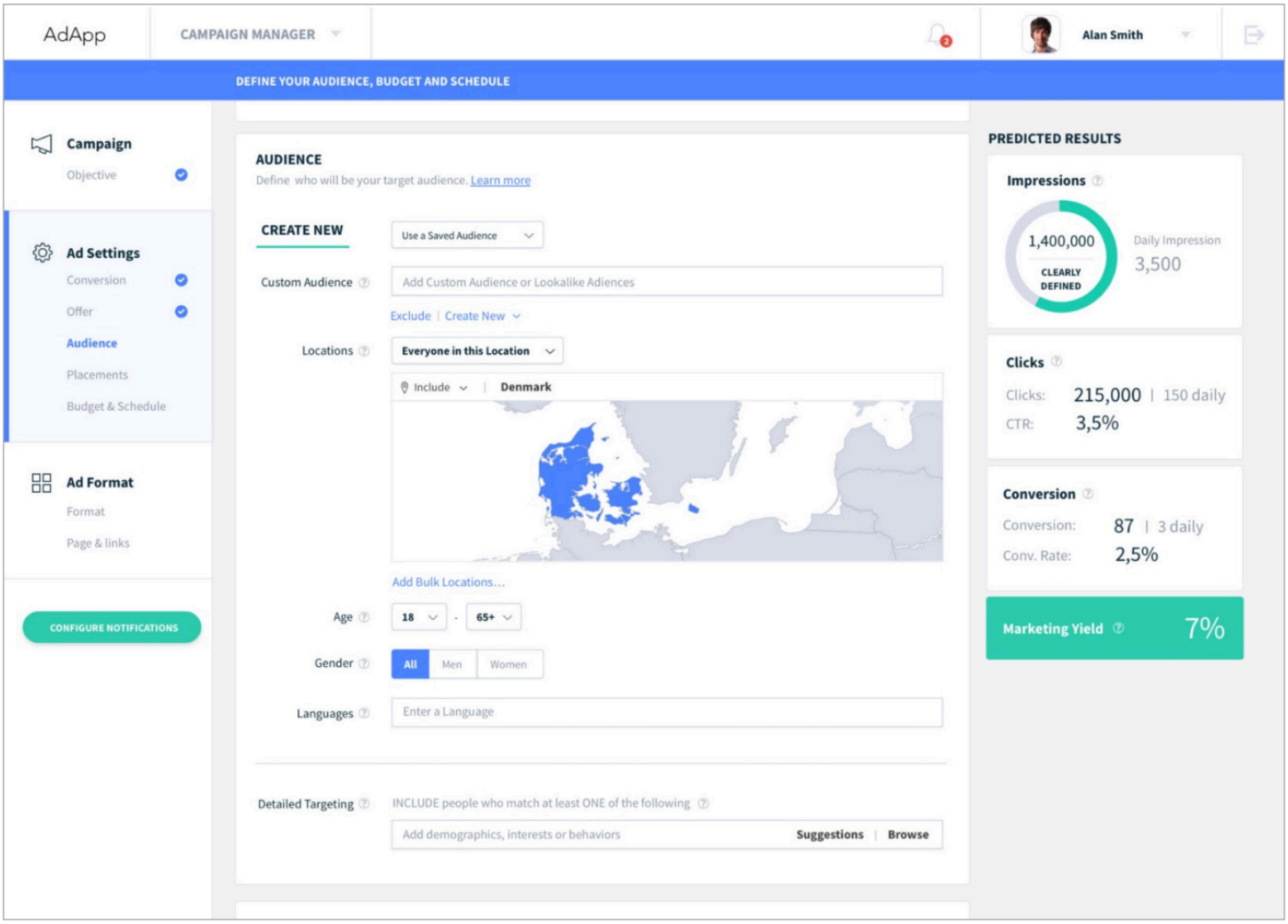
In addition to being the recommendations and possible actions supported by data in a given analytic application, insights are collections or visualizations of metrics that reveal patterns or previously unknown trends—for example showing change over time, or a combination of metrics that together show a correlation. By uncovering trends and relationships, and delivering recommended actions to the user, the application can inspire users to think of creative ways to solve a problem and make better decisions.

In slightly more technical terms, insights combine one or more metrics and slice or filter them by selected dimensions to create relevant and actionable information. They are often delivered in the form of data visualizations—things like charts and graphs—and they can incorporate information gleaned from machine learning and artificial intelligence, not just from raw analytical data.

**Benefits of using insights**

From a user's perspective, insights are key to enabling them to do their jobs faster and more easily than they could before, because insights deliver data-driven recommendations. They no longer have to rely on their own personal experience to make decisions, and conversely they are not overwhelmed by the volume of data available to them. The insight is delivered in a form that's actionable and easy to understand.

Because of their value, companies often strive to use insights to support their business decisions, preferably when those insights are embedded in a user's daily workflow.




**Insights embedded in a user's daily workflow (in a web applicationbuilt with the GoodData platform).**

## Best practices when working with semantic layers

For those looking to begin using semantic layers, a few best practices should be kept in mind. First, semantic layers should make interacting with data in your product simple. Start by understanding the end-user's journey and designing for their experience, then focus on building out your semantic layers to meet their needs. Sometimes this exercise can lead to a tendency to build your semantic layers with total flexibility in mind, so the end-user can run any report, slicing and dicing in any way they want. However, be aware that for the regular user, this only serves to confuse rather than empower.

Imagine if LinkedIn gave users this amount of power. Most users wouldn't touch it and would likely get overwhelmed and turn to another tool. Instead, LinkedIn provides just the curated information you need, and the definitions require zero explanation. Our advice: Don't give complete flexibility to end users. Build your app's semantic layers for today's needs, but allow for future extensibility for the next few releases.

**Validate new semantic layers**
Remember to validate the usability, quality, and accuracy of a new semantic layer or modifications to an existing semantic layer on different audiences early and often. Remember, tens of thousands of users with different needs expect the analytics you provide to be bulletproof. Reach out to real users to help evaluate prototypes or different versions of your product for different audiences. Then when you're ready to make changes to the semantic layers, remember that you cannot simply switch off your application, so you will need automated release management to ensure quality control. Bear in mind that the **80/20 rule** also applies in analytic applications, so design your semantic layers for the majority of your potential users.

**Measure semantic layers' performance and adoption**
Make sure that you plan for and measure performance, because you'll have thousands of users using your analytic application at the same time, and they won't necessarily understand that the underlying analytics infrastructure will need some time to crunch the numbers on millions of records. To ensure superb performance, look at the adoption of analytical features in your application, identify where users are not getting the value from the insights they expect and maybe are even giving up, and then actually talk to those users about their pain points. Then use your development, staging, and beta environments to test and fine-tune the performance of your semantic layers.

**Follow good governance and monitoring principles for semantic layers**
Finally, follow good governance, versioning, and monitoring principles for semantic layers. Good semantic layer governance results in a procedure, which specifies how to integrate measures and insights into the standard product to be used by other customers. Good semantic layers will not only record your analytics solution's dimensional model design, but will also serve as a "source of truth" for measure definitions as they are understood by your users. In addition, as part of your semantic layer governance, you should monitor how they are used by your users. If the definition of any of the semantic layer components changes, be sure to keep "snapshots" of how they've evolved over time.

## Logical data models: best practices for large-scale analytic applications

**Start with the business case**
Before you can start building a logical data model, you need to first understand your end-users including how many users you have and how many of them will interact with analytics as well as their analytics needs and business processes. Map out their needs and business processes, interview them about the decisions they need to make as they use the application. By having a better understanding of their workflow, you can design a logical data model that can answer their questions and support their decisions with accuracy and good performance.

Of course, you can't ignore the business case either. So take the time to interview key stakeholders and understand their needs and the kinds of decisions they've made in the past. How do they inform decisions today? And how can the analytic application help them make better decisions in the future?

**Build a logical data model one step at a time**
Don't try to build a perfectly comprehensive logical data model. Customers may ask you to give them access to all of their data, which is a big task that requires a lot of resources to successfully complete.

Not only that, but it doesn't really answer concrete questions or solve any particular problem just creates new problems for users who are overwhelmed by the amount of information they have access to.

Starting with a smaller initial model is also beneficial for you, not just the user. With a more manageable selection of data, you can get a better idea of what steps you'll need to take before launch. Will you need to prepare to teach every user, or is the change small enough that they'll adjust to it fairly easily? Even if you did wind up needing to teach your users, that time and resource commitment is less taxing if your users can start with a manageable set of data elements.

A less ambitious data model also enables user access to the data more quickly and gives you feedback about their experience fairly early on, which makes it easier for you to start working on the next iteration. So start with what you know your users need today, and then build on that knowledge as needs and decisions evolve over time. If you don't, there's less chance of your model scaling and performing as your customer expects.

Instead, focus on a logical data model that can evolve over time. Let's say you've designed a model with three different datasets for sales, product, and customer data. This model works well for the first few releases of your analytic application, but soon a new requirement is made known, and this data needs to be compared with competitor sales. In this case, you'd extract relevant data from the customer dataset—like city, region, and country information—into a separate dataset to be linked to the external benchmark data. Now the logical data model has evolved.

**Deal effectively with change**

Even though the model might not change every week, that doesn't mean that it never changes. As your model evolves, you'll need to ensure that it doesn't require you to put your analytic application on hold while you make changes—especially critical for a SaaS business where things change quickly.

To accommodate those changes, you should be following agile methodologies and avoiding downtime. Establishing a closed-loop feedback process with your users as you build your analytic application is critical to help inform what future changes should be.

You'll not only need tools for the creation and management of models, but you'll also need automated deployment and data loading tools. After all, you're building a large-scale analytics application for lots of external users, so you can't be sending emails to notify your users that there will be downtime caused by deployment or data loads.

Let's refer back to that example above. At this point, you've already made changes to the data model to add in the competitor sales information. That entailed modifying the underlying data model for a GoodData master workspace. Then a new insight is added to compare competitor sales to the workspace, and all of these changes are automatically delivered. This is all done without any downtime for customers.

## Measures and metrics: best practices for large-scale analytic applications

To create measures and metrics, data engineers and application designers should work together with those who have an in-depth understanding of the core business. This is the best way to ensure that metrics are meeting the business need.

One of the many values of measures and metrics is that they serve as a single source of truth. When creating measures and metrics, establish clear standards for metric definitions, composition, and layering.

One of the many values of measures and metrics is that they serve as a single source of truth. When creating measures and metrics, establish clear standards for metric definitions, composition, and layering. In addition, insist on a clear title and description for every metric, so that everyone understands what the metric is-and isn't. Achieve consistency by recording the definition in a semantic layer that is shared across the organization, and ensure that all developers are aware of the structure. Everyone should be referencing the same definition and structure.

Customization is another key component. Extend the usefulness of metrics by incorporating variables that automatically filter the data so that it is appropriate to the user viewing the results. Then ensure that you create sufficient documentation to account for future development or customization of your product by other teams. If metrics are widely shared across development teams, consider versioning the metrics so that testing and deployment can be done in a controlled and predictable manner.

Finally, you don't want to constantly be creating new metrics as business needs change. Be sure to encourage reuse of existing metrics and discourage the building of ad-hoc metrics as new demands crop up. A well-designed semantic layer will facilitate reuse.

## Insights: best practices for large-scale analytic applications

In order to be meaningful to the business, insights must be designed to support decisions that will further the organization's goals. This means validating key objectives and desired outcomes early in the process and identifying places in the workflow where additional information or a recommendation can make a positive impact. Insights should be tested under a variety of conditions and edge cases to ensure that the implied actions and recommendations are accurate.

Don't overwhelm your users with too many insights or too much data at once. Instead, walk through the workflow and identify key decision points. Provide insights that support the most important decision points within the business process you are trying to improve. Understand your user's needs and workflow, and think about revealing information only when it is needed, in a stepwise fashion or by allowing drill-downs.

Keep in mind that the majority of your users will not be data analysts. Utilize good user-centered design principles to provide users with a logical, consistent, and intuitive layout that keeps the focus on the most important metrics. If you do decide to allow for self-service exploration, make sure that you choose a user-friendly platform that not only lets users easily combine measures to build insights, but also gives automated recommendations to modify, customize, and break down an insight.

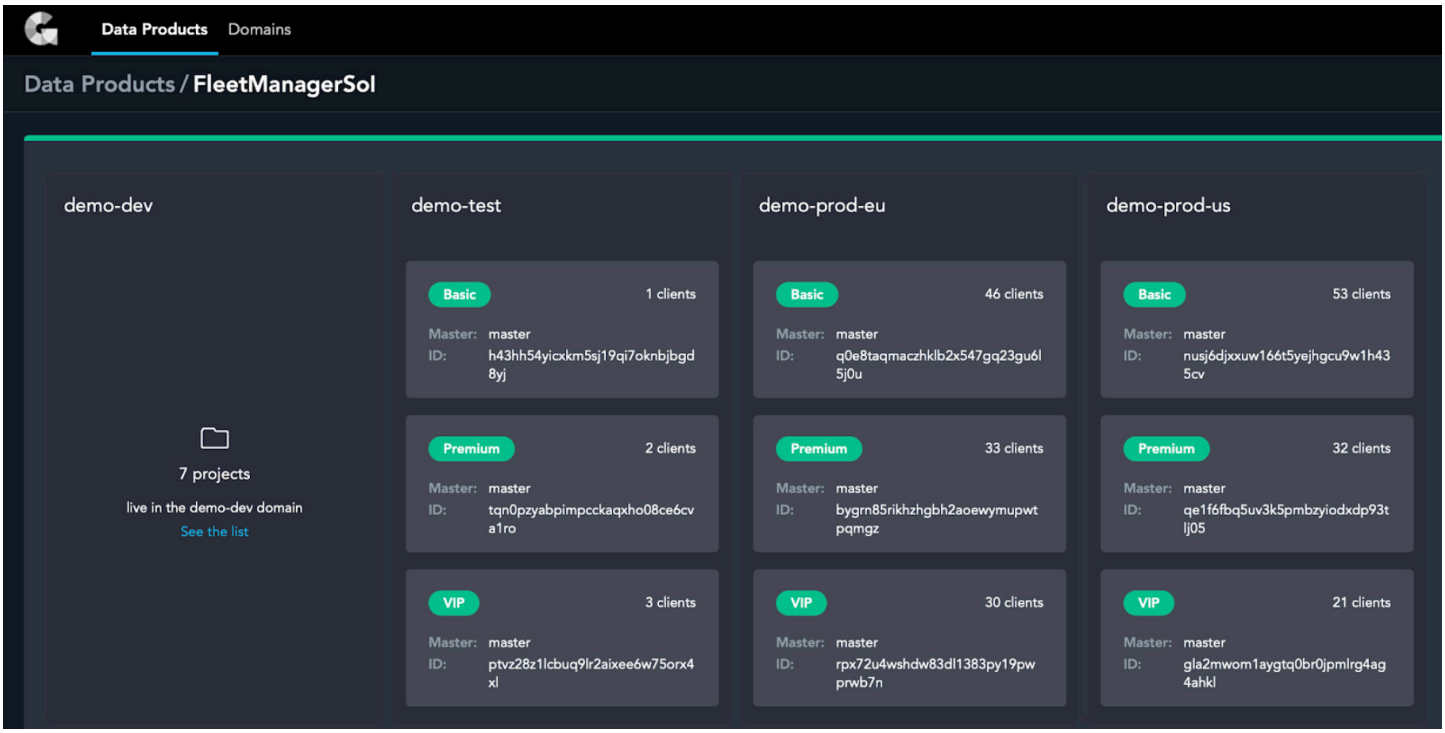
Finally, remember that insights that are integrated into day-to-day (especially front-line) decision making are on the critical path, so performance, accuracy, and reliability cannot be taken lightly.

## How GoodData helps you follow achieve semantic layer best practices

GoodData automatically organizes analytical data of your customers into customer-specific workspaces. Each workspace includes different semantic layer components (metrics and insights, queries and filters, and all visualizations including reports, dashboards, alerts, and annotations), analytic data, users, and groups. Workspaces enable per-customer customizability and access controls and enable management of releases using GoodData's out-of-the-box *Life Cycle Management* feature.

The Life Cycle Management (LCM) feature enables you to:

- Frequently release features and deliver changes across tens to thousands of customers. Staged or segmented rollouts across thousands of analytic applications are possible with GoodData's Automated Data Distribution (ADD) feature. With ADD, you can automatically upload data from a data warehouse (e.g., GoodData's ADS data store or Snowflake) and quickly distribute it to thousands of workspaces and tens of thousands of users, all of whom have different needs.
- Centrally manage delivery of analytic features to each product audience for beta releases or early access.
- Validate the usability, quality, and accuracy of a new semantic layer or modifications to an existing semantic layer on different audiences early and often.
- On-board and off-board customers automatically, such as after they sign up to use your product via a web form.
- Automatically "merge" changes made to a "master workspace" of each product segment into customer-specific workspaces.
- Issue or revoke thousands of workspaces based on a predefined template (such as a master workspace), and then manage users in these workspaces. LCM ensures a customer sign-up request triggers creation of a workspace according to a predefined "master" workspace template.




**LCM management console.**

GoodData's out-of-the-box GoodSuccess application helps you:

- Keep track of usage and engagement trends for each customers and different product segments, so you can identify areas of improvements and opportunity for future releases.
- Plan for and measure performance by examining the adoption of analytic features in your application, identifying where users are not getting the value they expect, and then talking to those users about their pain points. Use your development, staging, and beta environments to test and fine-tune the performance of your semantic layers.
- Track the performance of various semantic layer components such as dashboards, reports, and other dashboard components.

# Conclusion

The semantic layer of your application documents the way your business uses data to measure itself, and translates that data into business-friendly terms. It creates a single-source-of-truth for everyone from developers to business users. And it allows for agility and flexibility by providing an abstraction between your application and all of the original sources of data.

Remember to validate and support the business goals of your organization, and keep the focus on your users' critical decision points while you're designing. And as your business goals and procedures change, your semantic layer will need to change too, so build up your data model, metrics, and insights in a way that allows for future extensibility. By doing so, you're more likely to build an application that continues to deliver value to end users for years to come.

**Example on right: GoodSuccess**