



Building Governed Data Models With Sigma

A Practical Guide
For The Data Professional

Phil Ballai
Enterprise Architect
Sigma

Table of Contents

Table of Contents	1
1 Introduction	2
2 Sigma's Philosophy On Data Modeling & Governance	3
3 Key Functional Capabilities To Look For	4
4 Let's Make It Work	5
5 Using The Data Model	24
6 Best Practices	29
7 Reference Architecture	30
8 What's Next?	32

1 Introduction

The cloud data warehouse has become the de facto system of record for most modern enterprises—making governance, compliance, security, and a unified source of truth more critical than ever. But simply centralizing data in the warehouse doesn't guarantee consistent, trustworthy analytics. The real challenge is ensuring that every dashboard, report, and application draws from the same definitions and relationships, without creating bottlenecks for data teams.

That's where Sigma Data Models come in. Sigma's modeling layer is designed to be the governed, reusable semantic layer for your analytics environment—sitting directly on top of your warehouse, preserving its performance and security, and enabling trusted self-service for every business user. With Sigma Data Models, you can define joins, metrics, and permissions once, and reuse them everywhere—from dashboards and operational workflows to AI-driven analytics.

Sigma Data Models offer a path forward, enabling teams to:

- **Create and reuse** governed data assets that power dashboards, operational workflows, and AI-driven analytics.
- **Define and manage metrics, joins, and permissions** in one place, ensuring every analysis uses consistent business logic.
- **Adapt quickly to change** with versioning, lineage, and dependency awareness that prevent breakage as schemas evolve.

By putting governance, reusability, and agility in the same layer, Sigma makes it possible to deliver consistent insights without slowing innovation, allowing data teams to maintain control while enabling true self-service for business users.

Who is this eBook for?

This guide is for SQL-fluent, warehouse-first practitioners—data engineers, analytics engineers, BI leads, and architects—who are tasked with scaling governed analytics environments.

We'll explore how to model with confidence in Sigma, bring consistency to your data layer, and align your governance strategy with modern cloud architecture.

The goal is simple: **keep your logic in one place, run it in your warehouse, and make it accessible to everyone who needs it—all without losing control.**

2: Sigma's Philosophy On Data Modeling & Governance

Sigma Data Models are built on a simple truth: **governance and agility don't need to be at odds.**

The best modeling environments deliver trusted outputs without slowing innovation. That means giving data teams the tools to centrally define relationships, metrics, and security, while still enabling business users to explore, extend, and apply that logic in their own workflows.

Our approach blends governed modeling with real-time exploration, combining the flexibility of spreadsheets with the control of a semantic layer. Because models run directly on your cloud data warehouse, Sigma preserves your existing security posture, enforces role-based access, and scales seamlessly with your data.

Core Principles

Security First – All queries run in your warehouse. Sigma supports private deployments, works natively with warehouse row-level and column-level security (RLS/CLS) and OAuth authentication, and never stores data.

True Self-Service – Modeling features are opt-in. Data teams can define reusable calculations, metrics, and joins, while business teams leverage them without breaking governance.

Beyond Dashboards – Sigma Data Models power end-to-end workflows, from reconciliations and approvals to forecasting and write-back applications.

AI That Speaks SQL – Logic defined in your models becomes the knowledge base for "Ask Sigma," an AI assistant that improves query accuracy, search, and data labeling.

This modeling strategy aligns with the future of governed analytics: a modular, direct-to-warehouse layer that serves both technical and non-technical users without forcing a rigid process.

3. Key Functional Capabilities To Look For

Modeling Capabilities That Matter:

- **Simple Setup** – No separate modeling server or deployment step. All logic lives in Sigma Data Models and runs **directly on your warehouse**, preserving its performance, security, and cost controls.
- **Visual Relationships** – Define joins between datasets with an intuitive visual builder. Supports complex joins, one-to-many lookups, and fanout prevention to keep metrics consistent.
- **Security Controls** – Native support for your warehouse’s RLS/CLS policies, private cloud deployments, column-level permissions, and join pruning—all enforced in real time because queries run where your data lives.
- **Reusable Metrics** – Define KPIs like revenue, margin, or churn once and use them consistently across workbooks, dashboards, and applications without redefining them for each project.
- **Ask Sigma (AI)** – Models are queryable through natural language. Defined metrics and relationships improve context and accuracy for AI-generated queries, so answers match your governed definitions.
- **Limitless Exploration** – Users can interact with governed models flexibly and drill, filter, sort, and pivot without requiring new datasets for every question.
- **Writeback & Input Tables** – Users can enter data directly into governed workflows like reconciliations, forecasts, QA flags, and more, all while the logic still runs against live warehouse data.

Together, these features deliver on the promise of governed self-service—letting teams build confidently while maintaining consistency and trust, all in a **direct-to-warehouse** modeling environment.

4. Let's Make It Work

In this example, we'll build a Sigma Data Model ("model") for a fictional retail chain to answer core business questions about sales, profitability, and promotion effectiveness.

The model starts with point-of-sale transactions and enriches them with product, store, and calendar details.

From this foundation, we'll define governed metrics such as gross margin, units sold, and gross margin percentage that can be reused across dashboards, reports, and AI-driven queries.

We'll also apply column-level security (CLS) to protect sensitive cost and margin data, ensuring that only authorized roles can see it.

Finally, we'll demonstrate how Sigma's content validation prevents downstream breakage when model changes are published, and how governed metrics make it simple for anyone—from analysts to AI assistants—to explore sales performance confidently and consistently.

Key Business Questions This Model Will Answer

- Which product categories drive the highest revenue and margin?
- How does sales performance vary by store, region, or channel?
- What is our average order value (AOV) and units per transaction (UPT)?
- Which promotions delivered the largest sales lift?
- Are there seasonal patterns in revenue and margin across product lines?
- How does performance compare during promotional vs. non-promotional periods?

By answering these questions with a single, governed model, the retail team can trust that every metric—whether viewed in a dashboard, AI query, or ad hoc analysis—comes from the same consistent definitions.

NOTE

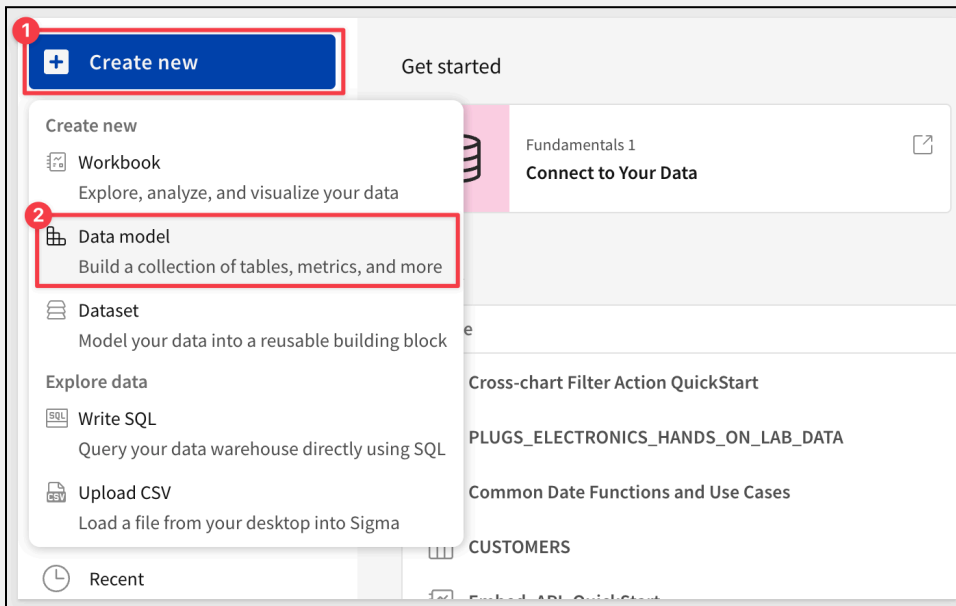
All demonstrations are performed using a trial environment.

Sigma recommends that you use non-production resources when completing this section.

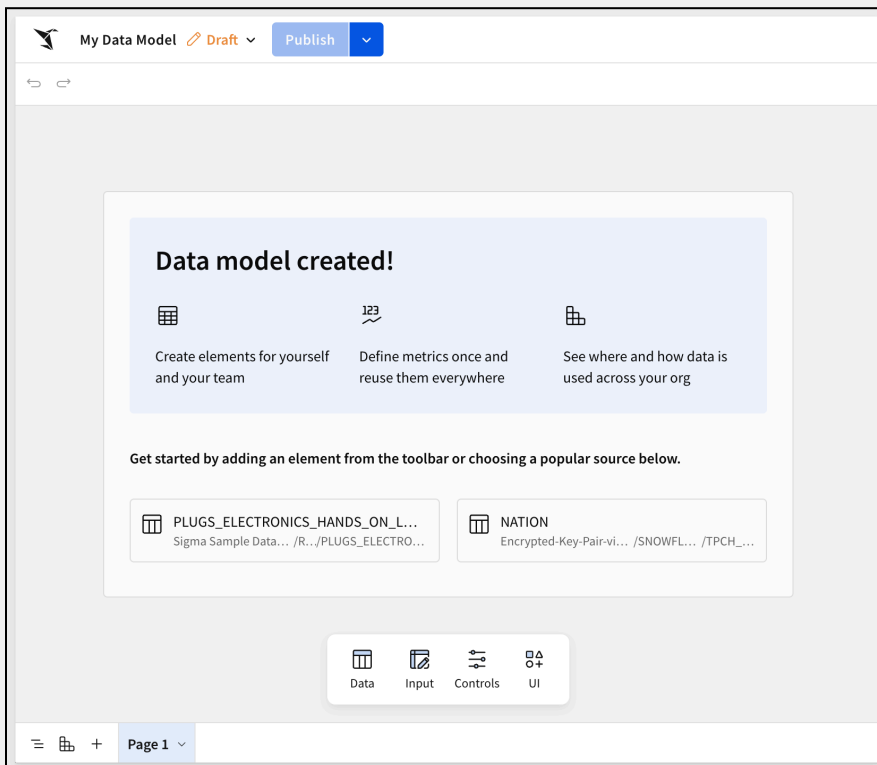
[Sigma Trial](#) 

1. Log into Sigma and create a new Data Model

We will use the Sigma provided sample database so that this can be recreated in any Sigma environment. Click the **Create new** button and select **Data model**:

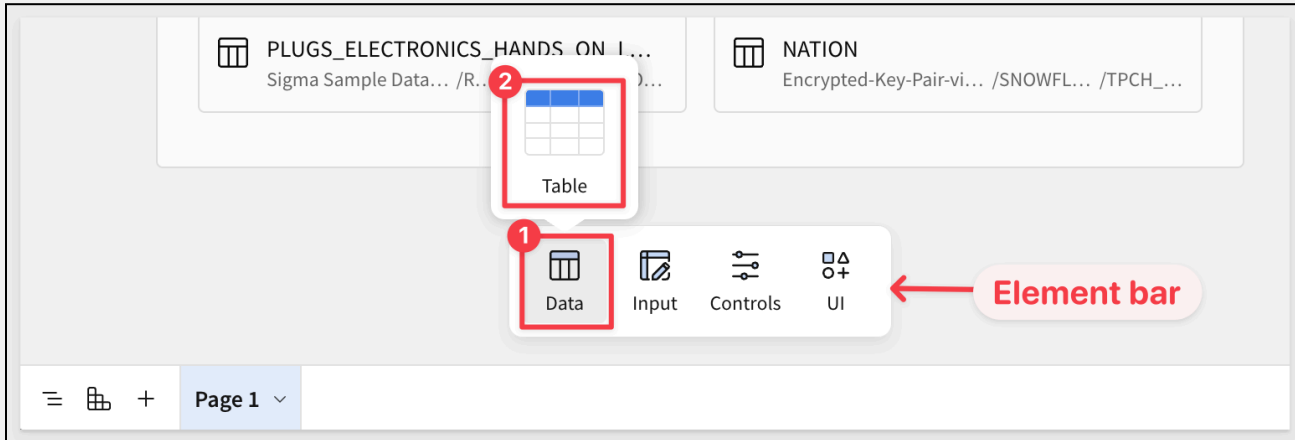


The modeling page looks similar to the workbook page so that model builders are working in a familiar interface:



2. Add the fact table

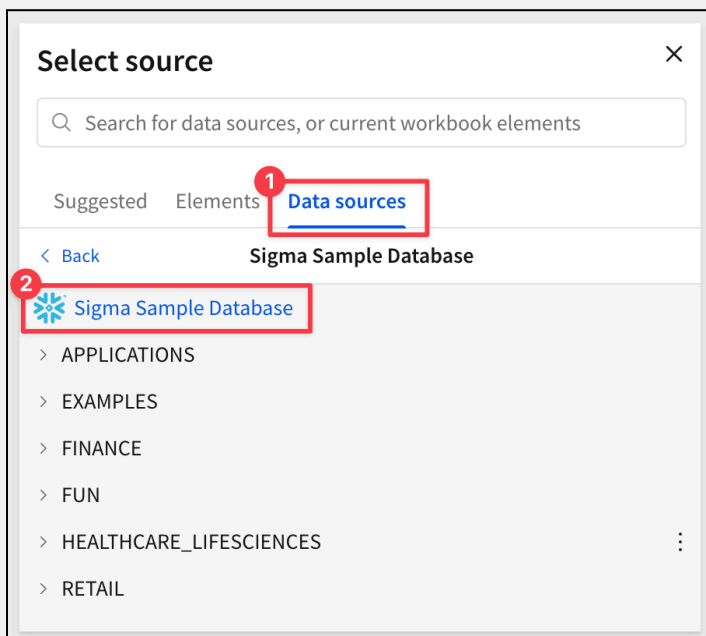
Using the **Element bar** click on **Data** and select **Table**:



Sigma presents us with three options on where to obtain our source data:

1. **Suggested:** a list of frequently used sources
2. **Elements:** elements that already exist in the data model we are working with
3. **Data Sources:** any connection that the user has access to (ie: Snowflake, Databricks, SQL)

Since we are using sample data, select the **Data sources** tab and the **Sigma Sample Database**:



Expand the `RETAIL` > `PLUGS_ELECTRONICS` schema and select the `F_SALES` table.

Once selected, it will load onto the workspace and we can access the `Element panel` which allows us to fully customize the table in a variety of ways:

The screenshot shows the Sigma Administrator interface. On the left, the `F_SALES` table is displayed with columns: Order Number, Cust Key, Store Key, Transaction Type, Date, and Purchase Method. A red arrow points from the `Element panel` label to the right-hand side of the interface. The `Element panel` is a sidebar with tabs for Modeling, Properties, and Format. The Properties tab is active, showing the table's data source, groupings, and columns. A red box highlights the `Columns` tab, which lists the table's columns: Order Number, Cust Key, Store Key, Transaction Type, Date, and Purchase Method.

Change the table name to `Sales Orders` and the description to `Order-level sales data, one row per order`

This fact table contains one row per product sold per order. We'll join this to the `Point of Sale`, `Store Locations` and `Customer List` dimensions, then define metrics directly in the model.

Open the data model's menu and select `Rename`. Use the name `Retail Sales Model`:

The screenshot shows the Sigma Administrator interface with the data model's menu open. A red box highlights the `Draft` button, and another red box highlights the `Rename` option in the menu. The menu also includes options like Move..., Save as a new data model..., Set badge..., Add shortcut..., Add to favorites, and Delete....

Open the same menu again and select `About this data model`. Here we can provide a useful description for users to better understand the model when it is first opened:

About this data model

Retail Sales Model ☆

1

Retail sales model combining orders, line items, and related dimensions for analysis of revenue, cost, and margin.

My documents

Owned by

SA Sigma Administrator

Last published 19 hours ago by you

Viewed 6 times in the last 60 days

5 elements

Click `Publish`.

3. Add dimension tables

Using the same workflow add the `F_POINT_OF_SALE` table to the model.

Rename it `Point of Sales` and set the description to `Line-item details for all point-of-sale transactions`:

Next we will relate the two tables. Click `Sales Orders` to select the table and then click the `+` as shown:

Retail Sales Model

Draft

Publish

1

Sigma Administrator

Sales Orders

Order-level sales data, one row per order

Order Number	Cust Key	Store Key	Transaction Type
227492	1807	318	Purchase
227507	1807	318	Purchase
227509	1807	318	Purchase
227514	1807	318	Purchase
227480	1807	318	Purchase
227557	1808	12	Purchase
227560	1808	12	Purchase

SUMMARY ^ 717,747 rows - 6 columns

Point of Sales

Line-item details for all point-of-sale transactions

Order Number	Product Key	Sales Quantity	Sales #
176581	264	1	
176814	264	1	
176816	264	1	
176919	264	1	
176969	264	2	
177143	363	5	
177204	351	1	

SUMMARY ^ 4,584,628 rows - 5 columns

Sales Orders

Modeling Properties Format

VISIBILITY

Visible as source ~ 0

2

RELATIONSHIPS → +

METRICS +

COLUMN SECURITY +

MATERIALIZATION +

Configure the relationship as shown. Most orders have more than one line item which is expected:

Add a relationship

Relationships allow you to pre-define a join between two tables. Relationships currently support Many-to-One or One-to-One logic.

Name

Description

Primary source

Sales Orders

Join keys

Order Number

Target source

Point of Sales (Page 1)

Join keys

Order Number

+ Add another mapping

Sales Orders key matching

2

No matches

0 (0%)

One match

10.34k (1%)

2+ matches

707.41k (99%)

Key matches Result preview

☐ Only show keys with no matches

Order Number
130
131
132
133
134
135
136
137

Order Number
130
131
132
133
134
135
136
137

Cancel

Save

We also want to know which products we purchased.

Repeat the workflow to add the `Products` table, but this time join it to the `Point of Sales` table:

Add a relationship

Relationships allow you to pre-define a join between two tables. Relationships currently support Many-to-One or One-to-One logic.

Name

Description

Primary source

Point of Sales

Join keys

Product Key

Target source

Products (Page 1)

Join keys

Product Key

+ Add another mapping

Point of Sales key matching

2

No matches

0 (0%)

One match

1.1k (100%)

2+ matches

0 (0%)

Key matches Result preview

☐ Only show keys with no matches

Product Key
1
2
3
4
5
6

Product Key
1
2
3
4
5
6

We have 100% match because every sale has to have an accompanying product.

Building Governed Data Models with Sigma

10

Add the `D_STORE` table and join it to the `Sales Order` table.

Every order is associated with a store location in our data:

Add a relationship

Relationships allow you to pre-define a join between two tables. Relationships currently support Many-to-One or One-to-One logic.

Name

D_STORE

Description

Describe what this source adds

Primary source

Sales Orders

Target source

D_STORE (Page 1)

Join keys

Store Key

Store Key

+ Add another mapping

Sales Orders key matching

No matches0 (0%)

One match200 (100%)

2+ matches0 (0%)

Key matchesResult preview

Only show keys with no matches

Store Key

6

8

12

Store Key

6

8

12

Rename the table `Stores Locations`.

Lastly, add the `D_CUSTOMER` table and join it to the `Sales Order` table.

Every order has a customer associated with it:

Add a relationship

Relationships allow you to pre-define a join between two tables. Relationships currently support Many-to-One or One-to-One logic.

Name

D_CUSTOMER

Description

Describe what this source adds

Primary source

Sales Orders

Target source

D_CUSTOMER (Page 1)

Join keys

Cust Key

Cust Key

+ Add another mapping

Sales Orders key matching

No matches0 (0%)

One match4.87k (100%)

2+ matches0 (0%)

Key matchesResult preview

Only show keys with no matches

Cust Key

1

2

3


Cust Key

1

2

3

Click `Publish`.



Building Governed Data Models with Sigma

11

Now that we have all our source tables, let's rename the workbook page to `Model Structure` to help future editors understand the model better:

Retail Sales Model Draft Publish

Sales Orders

Order Number	Cust Key	Store Key	Transaction Type	
227492	1807	318	Purchase	202
227507	1807	318	Purchase	202
227509	1807	318	Purchase	202
SUMMARY ^ 717,747 rows – 6 columns				

Point of Sales

Order Number	Product Key	Sales Quantity	Sales Amount
176581	264	1	
176814	264	1	
176816	264	1	
SUMMARY ^ 4,584,628 rows – 5 columns			

Products

Product Key	Product Name	Product Type	Product F
1	1-60" Class (59.5" Diag.) ...	Entertainment	High Defi
2	2-65" Class (64.5" Diag.) ...	Entertainment	High Defi
SUMMARY ^ 1,096 rows – 14 columns			

Stores Locations

Store Key	Store Name	Store Address	Sto
84	Baltic Store #84	84 Baltic Pass	Bet
122	Underwood Store #122	122 Underwood Route	Gre
200 rows – 19 columns			

Model Structure

All changes published

For more information on navigating the data model workbook page, see [Data model overview page](#)

4. Data Model landing page

Before we go further, let's take a moment to understand how users will access the model when they want to build content from its resources.

From the `Publish` menu, select `Go to published version`:

Retail Sales Model Draft Publish

Sales Orders

Order Number	Cust Key	Store Key	Tran
227492	1807	318	Purchase
2024			

Go to published version

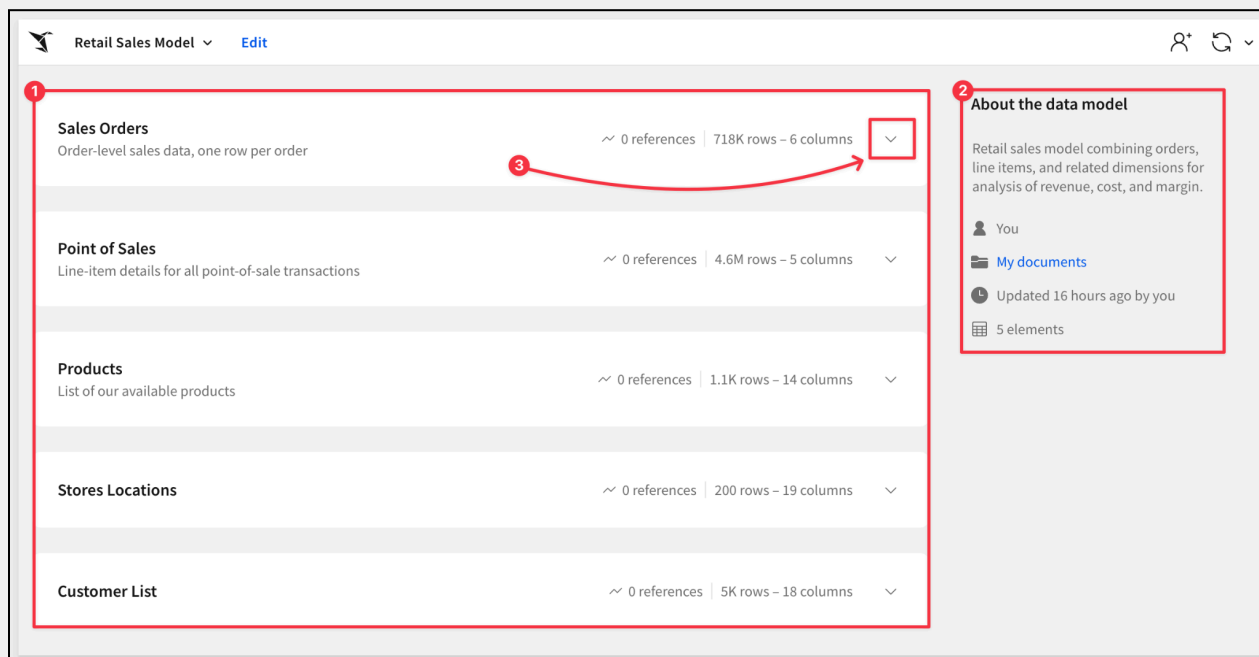
Discard draft

Switch to mobile view

Preview

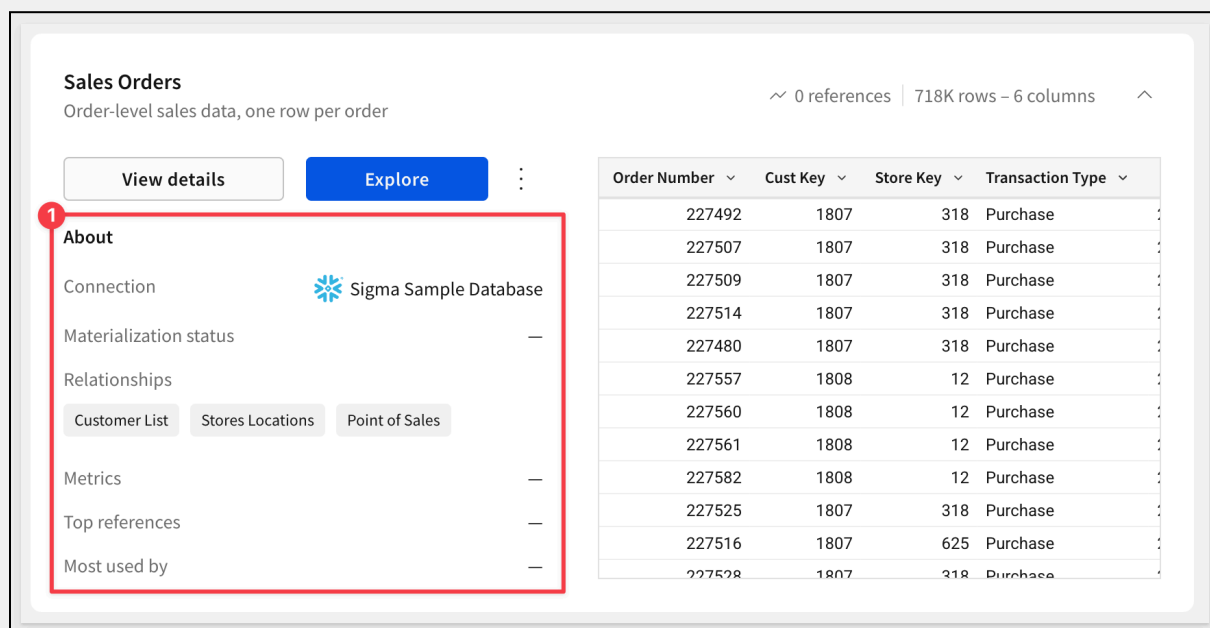
We can see all the tables in the model and some basic information about it like a description and when it was last updated.

We can get detailed information about each table, for example expanding the `Sales Orders` table:



Here we can see all the detailed information about the table, its relationships, metrics, and so on. We also can see the columns in the table and how many rows there are.

Users can click `Explore` and jump straight into a workbook with the table pre-loaded so that they can start building immediately:



This is just a preview to orient you while we build out the data model more.

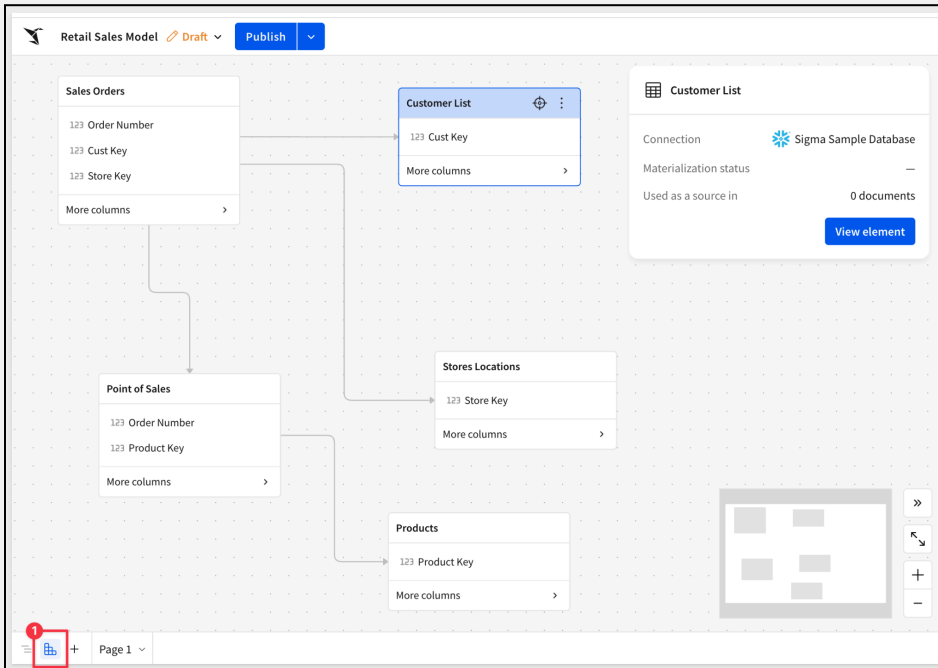
For more information on navigating the landing page, see [Data model overview page](#)

Click `Edit` so we can build more functionality into the model.

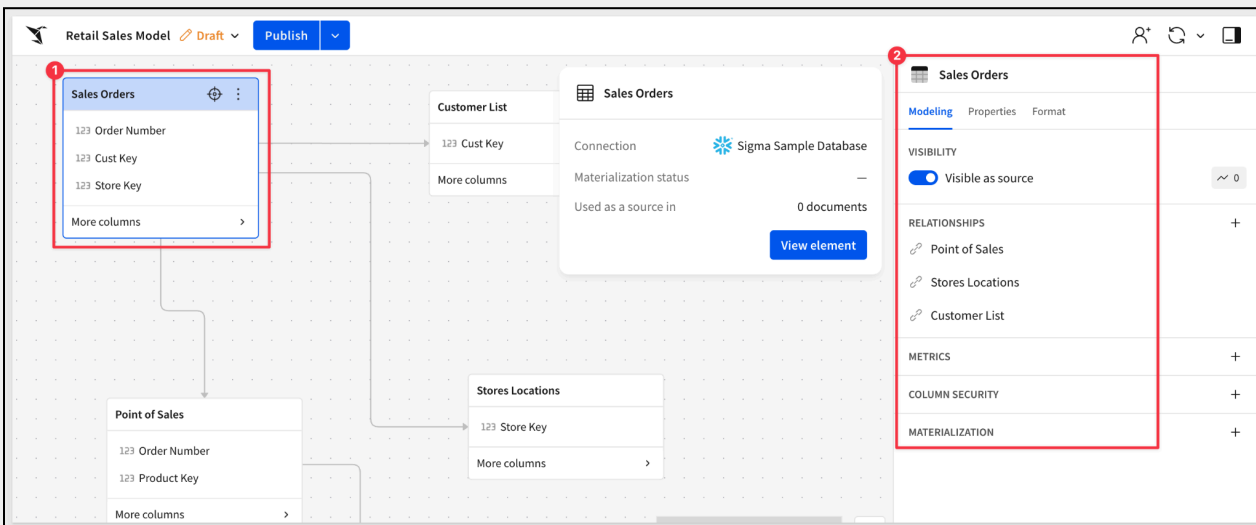
5. Data Model Entity Relationship Diagram (ERD)

By clicking on the “stack-boxes” icon in the lower left corner of the page, we access the entity relationship diagram (ERD) which provides a more visual layout that can be more familiar to some users.

All the same operations are available via the ERD as the workbook page. The tables can be dragged around to visualize the design to suit:



If we select the `Sales Orders` table, we have access to the `Element panel` and all its existing configuration:



For more information, see [Guidance for modeling relationships](#) [↗](#)

Click `Publish` and then click the `ERD` icon again to exit it.

6. Metrics

Metrics in Sigma are dynamic and reusable calculations that are specific to a data source. As such, metrics usually reference specific columns. You can create a metric to provide reliable and efficient aggregate calculations, abstracting complex formulas away from business users.

Let's add a few simple metrics by selecting the `Point of Sales` table and clicking the `+` for `Metrics`:

The screenshot shows the Sigma interface with two tables: 'Point of Sales' and 'Stores Locations'. The 'Point of Sales' table is highlighted with a red box and a red circle with the number 1. The 'Stores Locations' table is also visible. On the right side, the 'Point of Sales' table's properties are shown, including 'Modeling', 'Properties', and 'Format' tabs. The 'METRICS' button is highlighted with a red box and a red circle with the number 2.

Order Number	Product Key	Sales Quantity	Sales Amount
176581	264	1	
176814	264	1	
176816	264	1	
176818	264	1	

SUMMARY ^ 4,584,628 rows - 5 columns

Store Key	Store Name	Store Address
84	Baltic Store #84	84 Baltic Pass
122	Underwood Store #122	122 Underwood Route

SUMMARY ^ 200 rows - 19 columns

Point of Sales

Modeling Properties Format

VISIBILITY

Visible as source ~ 0

RELATIONSHIPS +

Products

METRICS +

COLUMN SECURITY +

MATERIALIZATION +

NOTE

The complexity of the calculation is not the focus right now. The point is that Sigma makes it easy to create anything from simple to wildly complex calculations to use as metrics. This way, users of the data model can use the result and not make mistakes by creating their own calculations for common data.

Next, configure the following metrics:

Gross Margin:

Add a metric

Name
Gross Margin

Description
Profit in dollars before operating expenses

Formula
fx Sum([Sales Amount]) - Sum([Cost Amount])

☐ Timeline

Point of Sales

Order Number	Product Key	Sales Quantity	Sales Amount	Cost Amount
176581	264	1	96.6	35.78971962616822
176814	264	1	96.6	35.78971962616822
176816	264	1	96.6	35.78971962616822
176919	264	1	96.6	35.78971962616822
176969	264	2	193.2	71.57943925233644
177143	363	5	341.98125	272.9018691588785
177204	351	1	68.172	33.98130841121495
177247	391	3	249.33149999999998	195.98130841121494
177247	380	3	155.24999999999997	111.86915887850466

Preview

\$ % .0 .00 123

Gross Margin
\$231,517,478.81

Cancel Save

Did you notice the **Timeline** option?

That's interesting, but notice we don't have a date column in **Point of Sales**. **No problem!** We can simply use a **Lookup** to pull it from the **Sales Orders** table so we can use the timeline feature.

Click **Save**.

Add a new column via lookup as shown:

Sigma Administrator

Point of Sales

Order Number	Product Key	Sales Quantity	Sales Amount
176581	264	1	96.6
176814	264	1	96.6
176816	264	1	96.6

Context Menu:

- Filter
- Add new column
- Add column via** (2)
 - Lookup...** (3)
 - Period over period comparison...
- Duplicate column
- Rename column
- Set description...

Modeling Proper

Visibility

☒ Visible as

Relationships

Products

Metrics

Gross Margin

Column Security

Configure the lookup:

Edit lookup

1. Which column would you like to add?

Select element

Model Structure - Sales Orders

Column(s) to add: Date

Aggregate: None

+ Add column

2. Map two elements

Select key columns that have matching values in both sources

Point of Sales: Order Number

Sales Orders: Order Number

+ Add another mapping

KEYS WITH MATCHES: 100%

KEYS WITH MULTIPLE MATCHES: 0%

Learn more Cancel Done

There is a 100% match as every order has to have occurred on a date.

Click `Done`.

Now we have the column we need to use the `Timeline` feature.

Click the pencil icon to edit the `Gross Margin` metric:

Sigma Administrator

Point of Sales

Order Number	Day of Date (Sales Order...)	Product Key	Sale
489707	2025-02-19	744	
320516	2023-05-06	715	
322319	2023-08-29	714	

SUMMARY 4,584,628 rows - 6 columns

Stores Locations

Store Key	Store Name	Store Address
84	Baltic Store #84	84 Baltic Pass

Modeling Properties Format

VISIBILITY

Visible as source

RELATIONSHIPS

Products

METRICS

123 Gross Margin

COLUMN SECURITY

MATERIALIZATION

We can configure the timeline using the date column and select from several pre-configured comparison periods. Once selected, the metric has a chart and displays data related to the comparison period:

Edit metric

Name

Gross Margin

Description

Profit in dollars before operating expenses

Formula

fx

Sum([Sales Amount]) - Sum([Cost Amount])

Timeline

Date

Day of Date (Sales Orders)

Truncation

Month

Comparison period

This month last year

Direction

Higher is better

Point of Sales

Order Number	Day of Date (Sales Orders)	Product Key	Sales Quantity	Sales Amount	Cost Amount
321217	2024-03-06	744	1	66.05775	78.03738317757009
320991	2024-12-16	726	2	46.512	38.523809523809526
327496	2023-11-07	704	2	40.698	39.271844660194176
328771	2023-02-18	704	3	61.047	56.6822429906542
328633	2024-12-21	715	5	95.401875	92.45238095238095
328633	2024-12-21	725	2	147.798	139.80952380952382

Preview

\$

%

0

.00

123

Gross Margin

\$6,598,992

14.2% Jun 2025 vs 2024

Cancel

Save

Click `Save`.

Repeat this process to add two more metrics:

Units Sold:

COUNT([Sales Quantity])

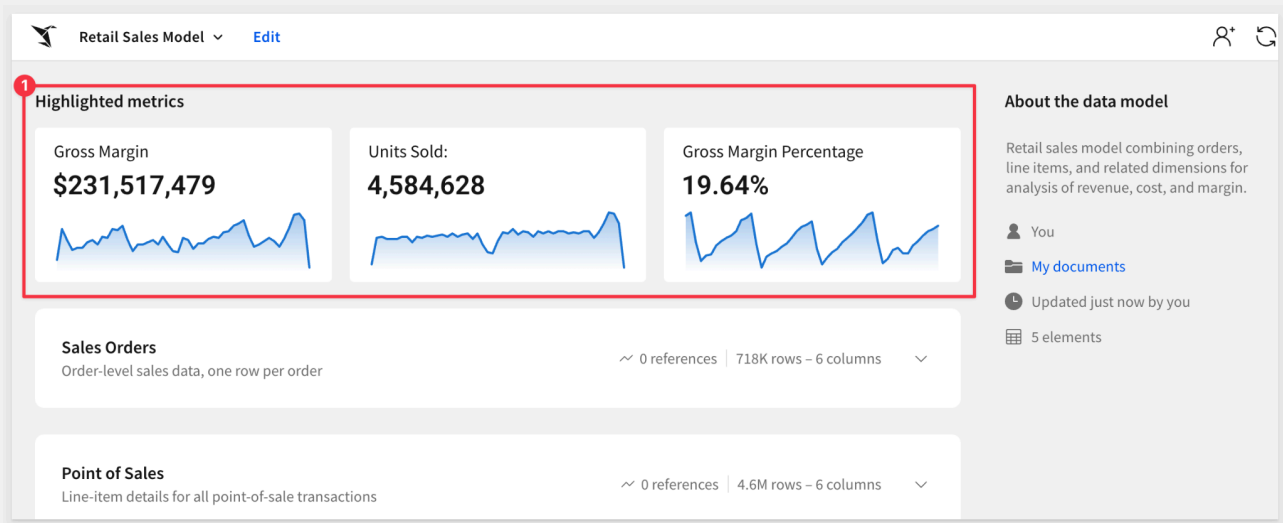
Total number of units sold in the period

Gross Margin Percentage:

(SUM([Sales Amount]) - SUM([Cost Amount])) / SUM([Sales Amount])

Profitability as a percentage of revenue

Once done, we can publish the data model and `Go to the published version` again to see how metrics are displayed:



7. Protecting sensitive data – Column-Level Security (CLS)

Often the same data model will be used by users who have different needs and permission levels. In some cases, showing all columns is permissible, but in others it isn't.

For example, if we look at the `Customer List` table, we see a column `Cust Age`; we need to protect that so only select users have access to it.

In this case, we only want our “Legal” team members to access that column.

Configuring that is simple, and we can also select specific users:

The screenshot shows the Sigma Data Model interface. On the left, a table preview for 'Customer List' is visible with columns: 'Number', 'Day of Date (Sales Order)', and 'Product Key'. The table contains 4,584,628 rows. On the right, the 'Modeling' tab is active, showing the 'Customer List' table. The 'COLUMN SECURITY' section is expanded, and the 'Cust Age' column is selected. A red box labeled '1' highlights the 'Customer List' table name. A red box labeled '2' highlights the '+' button next to 'COLUMN SECURITY'. A red box labeled '3' highlights the 'Update column security' dialog box. In this dialog, 'Restricted columns' is set to 'Cust Age', 'Criteria' is set to 'Specific users and teams', and the 'Legal (Team)' is selected. A 'Delete' button is at the bottom right of the dialog.

Now when users access the data model, if they are not a member of the Legal team, they will not see the column or know it even exists in the data.

There is also a column of JSON data that came through from the warehouse. We need to be sure that the data in that is permitted. Working with JSON data can be a pain but Sigma has handled the heavy-lifting for us.

First, let's see what's in the data:

Age Group ▾	Civil Status ▾	Loyalty Program ▾	Cust Json Field ▾	Updated At ▾
36-55	Single	1	{\"AGE_GROUP\": \"36-55\", \"CIVIL_STATUS\": \"Single\", \"LOYALTY_PROGRAM\": \"1\"}	null
36-55	Widowed	1	{\"AGE_GROUP\": \"36-55\", \"CIVIL_STATUS\": \"Widowed\", \"LOYALTY_PROGRAM\": \"1\"}	null
18-35	Divorced	1	{\"AGE_GROUP\": \"18-35\", \"CIVIL_STATUS\": \"Divorced\", \"LOYALTY_PROGRAM\": \"1\"}	null
18-35	Divorced	0	{\"AGE_GROUP\": \"18-35\", \"CIVIL_STATUS\": \"Divorced\", \"LOYALTY_PROGRAM\": \"0\"}	null
56+	Widowed	0	{\"AGE_GROUP\": \"56+\", \"CIVIL_STATUS\": \"Widowed\", \"LOYALTY_PROGRAM\": \"0\"}	null
18-35	Single	0	{\"AGE_GROUP\": \"18-35\", \"CIVIL_STATUS\": \"Single\", \"LOYALTY_PROGRAM\": \"0\"}	null

SUMMARY ^ 4,972 rows – 18 columns

This looks like data we already have in existing columns so we can safely delete the column since it does not provide additional data:

Cell value

☒ Tree view ☐ Raw text

```
AGE_GROUP: 36-55
CIVIL_STATUS: Single
CUST_ADDRESS: 418 Andrews Street
CUST_AGE: 40
CUST_CITY: Costa Mesa
CUST_COUNTY: Orange County, California
CUST_GENDER: Female
CUST_KEY: 1
CUST_NAME: Rosa Newton
CUST_REGION: West
CUST_SINCE: 2023-10-14
CUST_STATE: California
CUST_TYPE: Individual
CUST_ZIP_CODE: 92628
> LOYALTY_EXTRA
LOYALTY_PROGRAM: 1
```

NOTE

If the JSON data did have a value, Sigma can extract it directly in a few simple “wizard-based” steps—no additional tooling required!

For more information, see the QuickStart, [Parsing JSON Data in Seconds](#) ↗

8. Materialization

Materialization is built-in and easy to use. For those not familiar, materialization is a type of caching, where query results are written into a table in a data warehouse, and then refreshed at regular intervals (often daily).

This pre-calculated or pre-aggregated data can be accessed more efficiently and quickly than recomputing the results every time the query is executed.

In our use case, we don't have a very complex model, but when a model has a table that is very large, contains many complex calculations, or other complexities, it can be a good candidate for materialization.

For example, the `Point of Sales` table has 4.5M rows and three metrics, so let's use that to demonstrate the process.

Select the table and create a materialization:

The screenshot shows the Sigma Administrator interface. On the left, there are two tables: 'Point of Sales' and 'Stores Locations'. The 'Point of Sales' table is highlighted with a red box and a red circle with the number 1. The table has columns: Order Number, Day of Date (Sales Orde..., and Product Key. Below the table, it says 'SUMMARY ^ 4,584,628 rows - 6 columns'. The 'Stores Locations' table has columns: Store Key, Store Name, and Store Address. Below the table, it says 'SUMMARY ^ 200 rows - 19 columns'. On the right, there is a sidebar for the 'Point of Sales' table. It has tabs: Modeling, Properties, and Format. Under the 'Modeling' tab, there are sections: VISIBILITY (Visible as source), RELATIONSHIPS (Products), METRICS (Gross Margin, Units Sold, Gross Margin Percentage), and COLUMN SECURITY. At the bottom of the sidebar, there is a 'MATERIALIZATION' button with a red circle with the number 2 and a red arrow pointing to a '+' icon.

Sigma has drastically reduced the effort it takes to materialize data into the UI directly. We simply create a schedule, selecting the elements and frequency to suit our needs.

Once created, the status is `Pending` and will be run immediately, and then will run on the schedule. We can also execute it on-demand:

The screenshot shows the 'Materialization schedules' dialog. It has tabs: Schedules and History. Under the 'Schedules' tab, there is a table with columns: Summary, Status, and Next run. The table has one row: 'Point of Sales' with a status of 'Pending' and a next run of 'Aug 15, 2025 - 3:00:00 AM'. The 'Status' column is highlighted with a red box and a red circle with the number 1. Below the table, there is a '+ New schedule' button. On the right, there is a 'Materialize now' button with a red circle with the number 2 and a red arrow pointing to it. Below the 'Materialize now' button, there is a 'Delete schedule' button. A red circle with the number 3 is also present near the 'Materialize now' button.

We can also see that materialization is configured on the `Point of Sales` table and get the status directly:

The screenshot shows the Sigma Administrator interface. A table named 'Point of Sales' is displayed with columns 'Order Number' and 'Day of Date (Sales Orders)'. A popup menu is open over the table, showing the 'Point of Sales' materialization status. The status is 'Success' with a green checkmark, indicating it was updated 'a minute ago'. The next materialization is scheduled 'in 6 hours'. There are two red circles with numbers: '1' points to the table name 'Point of Sales' and '2' points to the information icon in the table's header.

Order Number	Day of Date (Sales Orders)
489707	2025-0
320516	2023-0
322319	2023-0
...	...
SUMMARY	4,584,628 rows – 6 columns

Point of Sales

MATERIALIZATION

Materialization status ✓ Success
a minute ago

Next materialization in 6 hours

[Materialize now](#) [View schedule](#)

NOTE

There are “data-freshness” implications when using materialized data that need to be carefully considered.

Materializing improves performance by storing a snapshot of query results, but those results will only update on the refresh schedule you define. This means users may not see the most up-to-date transactions until the next refresh.

The optimal approach balances performance gains with your organization’s tolerance for latency in reporting.

For more information, see the QuickStart, [Materialization with Sigma](#)

Click `Publish`

The final step is to share the data model with specific users or teams. This step is optional but useful to see the impact of column-level security.

We created a few test users in our Sigma trial ([using Gmail’s alias feature](#)) so that we could assign different roles for testing.

We will share the data model with one of them, permitting `View` permission:

The screenshot shows the Sigma Administrator interface with the 'Retail Sales Model' data model. A 'Share Data model' dialog is open, showing a search for members and teams. A table lists the person or team with access and the permission. The table shows 'QuickStarts Build (phil+build@sigmacomputing.com)' with a 'Can view' permission. There are three red circles with numbers: '1' points to the 'Retail Sales Model' tab, '2' points to the user icon in the top right, and '3' points to the 'QuickStarts Build' entry in the table.

Retail Sales Model Draft Publish

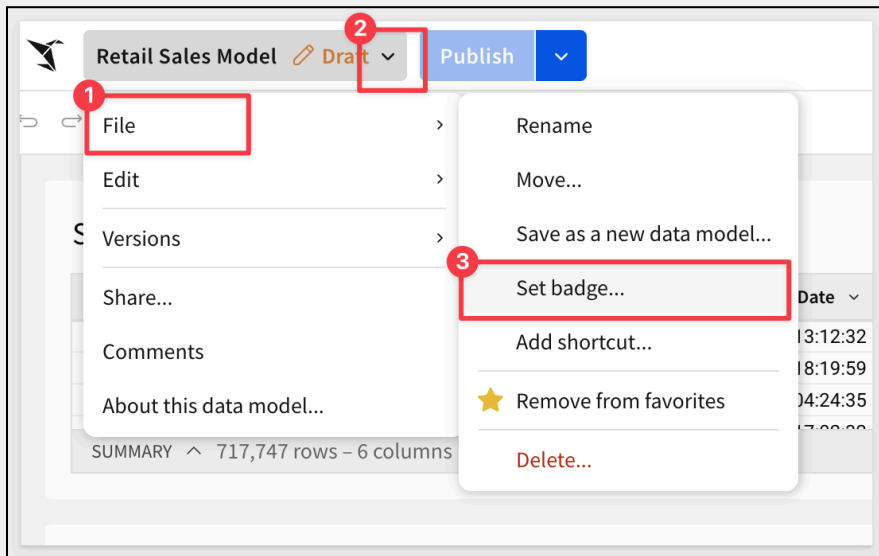
Share Data model

[Back](#)

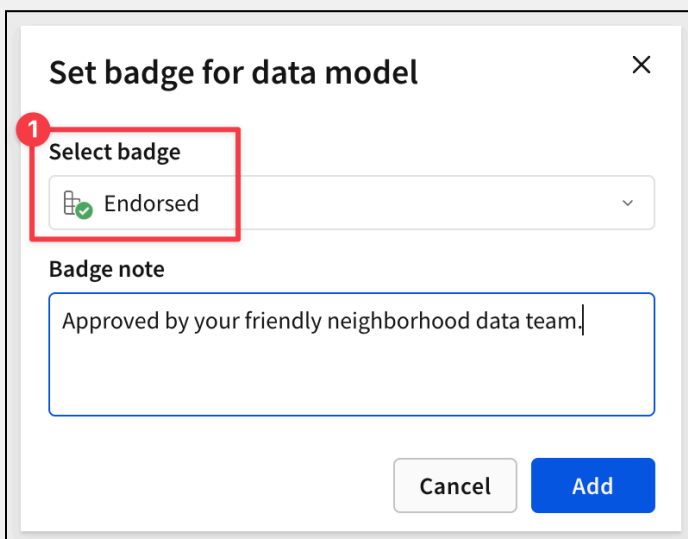
Search for members and teams

Person or team with access	Permission
QB QuickStarts Build (phil+build@sigmacomputing.com)	Can view

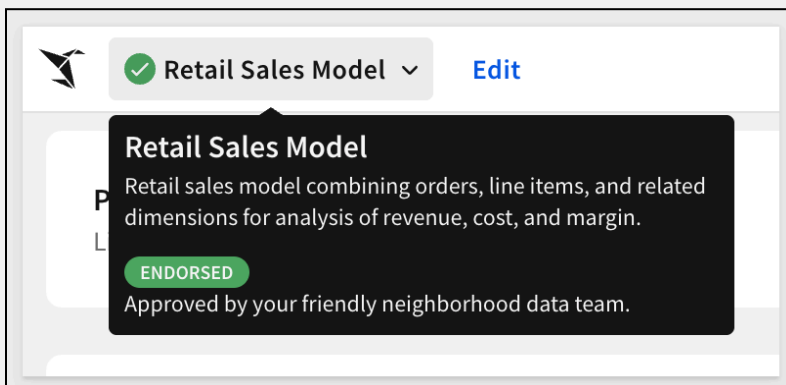
Lastly, we should `Badge` the model so that users know they are using the correct data:



Choose the `Endorsed` badge:

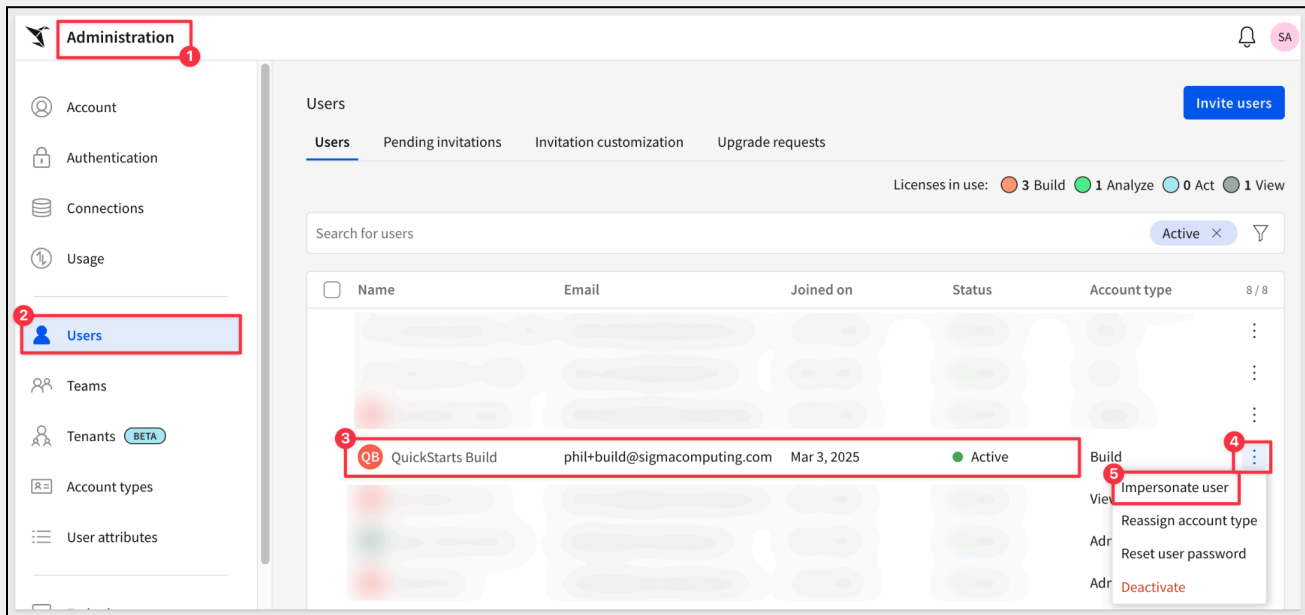


Now when users access the model, they can immediately tell it is the correct model for retail sales:

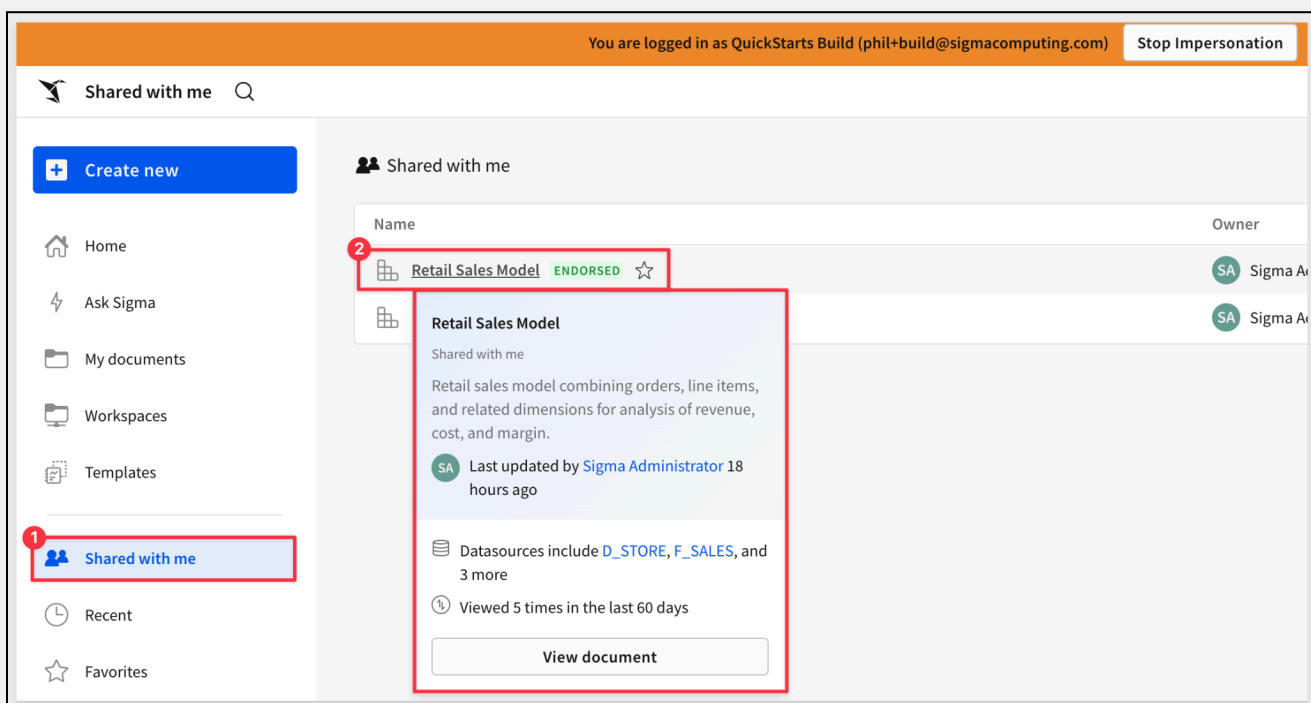


5: Using The Data Model

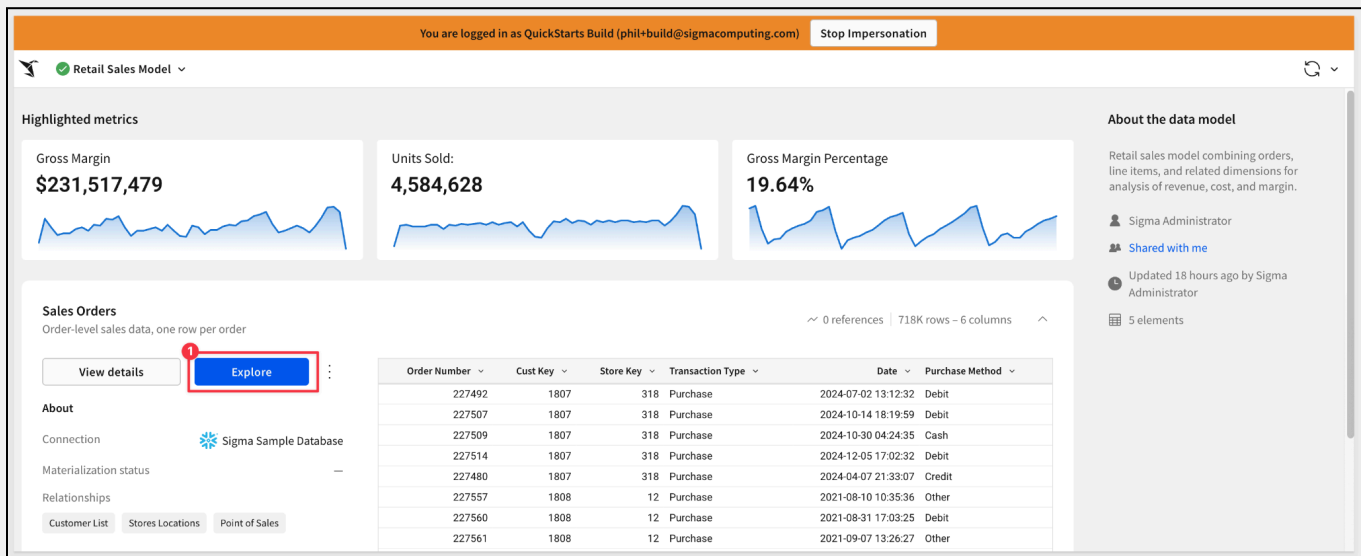
In Sigma, navigate to `Administration` > `Users` and select [impersonate](#) for the user that we shared the data model with:



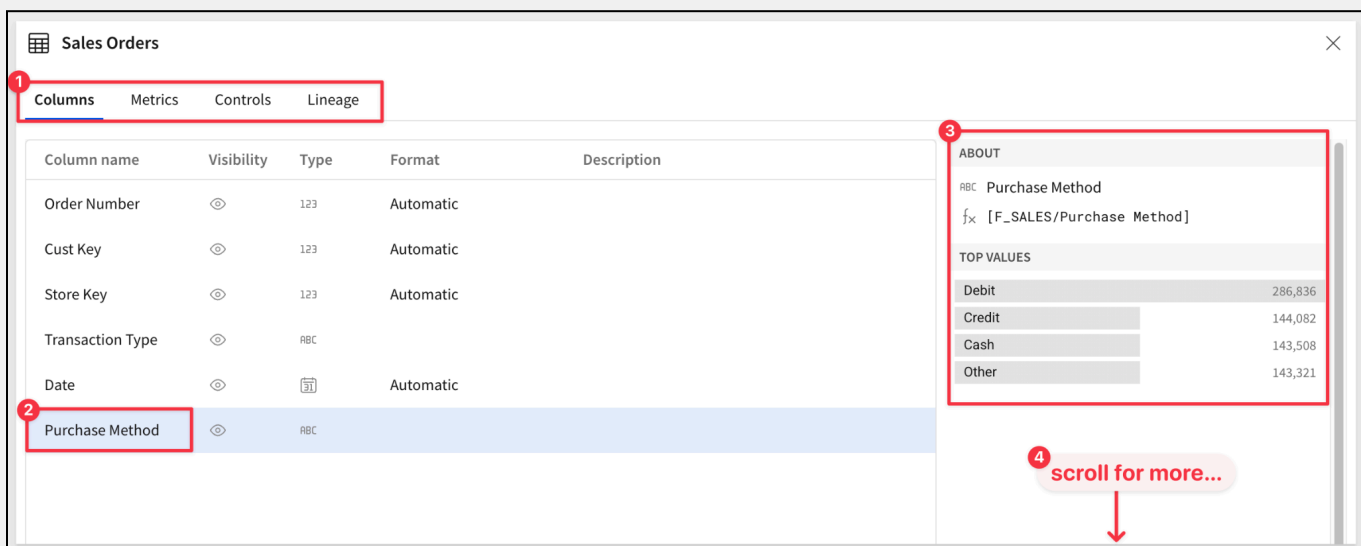
Now we are working in Sigma as if we are the selected user. On the `Shared with me` page we can see the `Retail Sales Model` listed; click on that:



Now we can see all the detail that was pre-configured for us and can select from any table to launch into an exploration:



The `View Details` button provides a way to see lower-level information on columns, metrics, controls, and lineage for each table:



Note: our metrics are on the `Point of Sales` table so they won't appear in the `Sales Orders` > `Metrics` tab.

Close the detail view and click the `Explore` button to land in a workbook.

A typical Sigma workbook appears with the table pre-loaded, but there is something special about the table:

The screenshot displays the Sigma Analytics interface. At the top, a status bar indicates the user is logged in as 'QuickStarts Build (phil+build@sigma.computing.com)' and provides a 'Stop Impersonation' button. Below this, the 'Exploration' tab is active, showing a 'Sales Orders' table. The table contains 17 rows of data with columns: Order Number, Cust Key, Store Key, Transaction Type, Date, and Purchase Method. A summary bar at the bottom of the table indicates 'SUMMARY 717,747 rows - 6 columns'. On the right, the 'Properties' panel for the 'Sales Orders' table is visible, showing a list of columns and metrics. The bottom of the interface features a toolbar with icons for Data, Input, Charts, Controls, UI, and Layout.

Order Number	Cust Key	Store Key	Transaction Type	Date	Purchase Method
227492	1807	318	Purchase	2024-07-01 13:12:32	Debit
227507	1807	318	Purchase	2024-10-13 18:19:59	Debit
227509	1807	318	Purchase	2024-10-29 04:24:35	Cash
227514	1807	318	Purchase	2024-12-04 17:02:32	Debit
227480	1807	318	Purchase	2024-04-06 21:33:07	Credit
227557	1808	12	Purchase	2021-08-09 10:35:36	Other
227560	1808	12	Purchase	2021-08-30 17:03:25	Debit
227561	1808	12	Purchase	2021-09-06 13:26:27	Other
227582	1808	12	Purchase	2022-02-01 19:30:34	Debit
227525	1807	318	Purchase	2025-02-22 02:47:05	Other
227516	1807	625	Purchase	2024-12-19 14:31:58	Credit

Because we configured relationships in the model, the user has access to all the other visible tables in the model and can select whichever columns they require for their analysis.

IMPORTANT

Sigma has automatically applied join pruning so that the auto-generated machine SQL used to load the data from the model—and subsequently the warehouse—only requests the data from the `Sales Orders` table.

Until the user selects columns from another table they will not be requested from the warehouse.

This makes the queries both efficient and cost effective.

For more information, see the QuickStart, [Sigma's Query Engine](#) [↗](#)

For example, we may want to add some store location related data that we can group on:

The screenshot shows the Sigma Explorer interface. At the top, there's a 'QuickStarts Build' button and a 'Save as' button. Below that, a formula bar shows '[Sales Orders/Stores Locations/Store Region]'. The main area displays a table with columns: Date, Purchase Method, Store Name (Stores Locations), Store State (Stores Locations), and Store Region (Stores Locations). The table contains several rows of data. On the right, a 'Source columns' dialog is open, showing a list of columns with checkboxes. The 'Store Name' and 'Store State' columns are selected. Red circles and boxes highlight the 'Sales Orders' data source and the selected columns.

While this is interesting, we may want to add one of the tables from the model directly. That's easy to do and follows Sigma's familiar workbook workflow.

Just click on `Data` in the `Element bar` and select `Table`.

Then select the `Data sources` tab, `Shared with me` and select the `Retail Sales Model` which will act just like any other source of data:

The screenshot shows the 'Select source' dialog in Sigma. The 'Data sources' tab is selected. Under the 'Shared with me' section, the 'Retail Sales Model' is highlighted. Red circles and boxes highlight the 'Data sources' tab, the 'Shared with me' section, and the 'Retail Sales Model' entry.

Let's select the `Customer List` table. Notice that the `Cust Age` column does not appear due to our CLS configuration:

QuickStarts Build

Customer List

Cust Key	Cust Name	Cust Address
1	Rosa Newton	418 Andrews St
3	Erick Mooney	512 Bacon Hill
5	Eduardo Rixon	364 Hamilton Li
7	Daria Griffiths	779 Fairview All
11	Daniel Lomax	803 Addison All
13	Natalie Mould	828 Cliff Street
15	Camila Dowson	211 Boldero Rur
17	Matt Leigh	988 Kinglake Ru
19	Tom Fall	507 Cheltenham
21	Ronald Rogan	297 Kimberley I
22	Renealie Fhrlen	188 Arlington H

SUMMARY ^ 4,972 rows - 17 columns

Columns

ADD COLUMN

- 123 Cust Key
- RBC Cust Name
- RBC Cust Address
- RBC Cust City
- RBC Cust State
- RBC Cust Zip Code
- RBC Cust County
- RBC Cust Region
- 123 Cust Since
- RBC Cust Type
- RBC Cust Gender
- RBC Age Group
- RBC Civil Status
- RBC Loyalty Program
- 123 Cust Json Field
- 123 Updated At
- RBC Updated By

Data Input Charts Controls UI Layout

We can also add KPI elements using our pre-built metrics:

QuickStarts Build

Point of Sales

Order Number	Day of Date (Sales Orders)	Product Key	Sales Quantity	Sales Price
221946	2022-05-27	937	3	282.48
221947	2022-06-04	956	3	
221947	2022-06-04	956	3	
221997	2025-05-19	942	3	439.3
221998	2023-05-27	924	2	
222049	2024-05-15	974	1	
222052	2024-06-04	920	3	618.6
222052	2024-06-04	126	1	
222162	2022-05-24	945	1	
222162	2022-05-24	895	3	260.88
222163	2022-05-30	857	2	

SUMMARY ^ 4,584,628 rows - 6 columns

GROUPINGS

Drag and drop columns from the Columns tab to create groupings

Columns Metrics

AVAILABLE METRICS

- 123 Gross Margin
- 123 Units Sold
- 123 Gross Margin Percentage

Create KPI element

Now that our test user has access to the data, they can build whatever they need and rest assured it is approved for use, accurate, and will produce analytics that can be trusted:

You are logged in as QuickStarts Build (phil@build@sigmacomputing.com) Stop Impersonation

Exploration Save as

Gross Margin \$231,517,479

Gross Margin Percentage 19.64%

Units Sold: 4,584,628

Sales Orders

Store Region (Stores Locations)	Purchase Method	CountDistinct of Order Number	Store Name (Stores Locations)	Store State (Stores Locations)	Cust Key	Store
- East	+ Cash	25554				
	+ Credit	25498				
	+ Debit	50938				
	+ Other	25797				
+ Midwest						
+ South						
+ Southwest						
+ West						

SUMMARY ^ 5 rows - 10 columns

Sales Orders

Properties Format Actions

DATA SOURCE

Sales Orders

GROUPINGS

GROUP BY

- RBC Store Region (Stores Locations)

CALCULATIONS

GROUP BY

- RBC Purchase Method

CALCULATIONS

- 123 CountDistinct of Order Number

Columns

ADD COLUMN

- RBC Store Name (Stores Locations)
- RBC Store State (Stores Locations)

6: Best Practices

As your Sigma deployment grows, governance and reusability become critical. A well-modeled Sigma data layer reduces rework, enforces consistency, and ensures that changes don't break downstream content.

Model reusability

- Build reusable models that serve as canonical sources of truth.
- Centralize joins and relationships **inside Sigma Data Models**, not in individual workbooks.
- Define metrics at the model level so they can be reused across dashboards, Input Tables, and Ask Sigma.
- Use links to prevent logic sprawl and enforce consistent joins across datasets.
- Add navigation controls inside the model so they can be selected for use easily in workbooks.

Versioning and change management

- Use Sigma's content validation to identify downstream dependencies before pushing changes.
- Tag major versions and maintain copies if needed to support parallel development.
- Adopt a branching workflow when working with shared data models or multiple contributors.

Testable, trustworthy logic

- Test your data logic—use row counts, joins, and group-by checks before publishing.
- Avoid unnecessary nesting in calculated columns in favor of model-level calculations or metrics.
- Enable lineage to trace how each field was derived and where it's used.
- Endorse data models and provide descriptions whenever possible.
- Minimize fanout and duplicate aggregates.
- Use relationships and link keys to control join paths.

A governed Sigma data model isn't just a modeling layer—it's a **direct-to-warehouse governance platform**. By defining relationships, metrics, and permissions in Sigma while executing queries in your warehouse, you avoid the sprawl and shadow logic that plague legacy BI stacks. The result: fewer surprises, faster iteration, and a modeling layer that earns trust at every level—from dashboards to AI assistants.

7: Reference Architecture

Modern analytics stacks are converging on a shared pattern: data is transformed and secured in the warehouse, while Sigma provides the governed, reusable modeling layer on top, executing queries directly on the warehouse for real-time, secure access.

In a typical Sigma deployment:

1. **Raw warehouse tables** (often landed via tools like Fivetran or Airbyte) are transformed into subject-specific data marts using [dbt](#) or similar tools.
2. These curated marts feed **Sigma Data Models**, where relationships, metrics, and permissions are defined once and reused across dashboards, Input Tables, embedded analytics, and AI assistants.
3. Sigma connects live to the warehouse for all queries, preserving full security controls (RLS/CLS), performance tuning, and cost visibility.

This architecture enables:

- A single governed layer for business logic, owned and maintained in Sigma.
- Governed and reusable across all personas—BI, operations, finance, and embedded app developers.
- Real-time data access without extra ETL layers, cubes, or data extracts.

Sigma sits at the center of your analytics workflow—integrating governed data models, enterprise-grade platform controls, AI-assisted analysis, and a full range of workbook experiences—while remaining directly connected to your warehouse. The result is trusted, consistent analytics for every user, from spreadsheets to AI queries.

In Figure 1 below, the top layer represents the many ways users work with governed data models inside Sigma, from spreadsheets and reports to AI-driven queries and embedded applications.

The middle layers show Sigma’s AI and enterprise platform capabilities, **which sit alongside data modeling to deliver secure, scalable analytics.**

At the bottom are the live data sources—cloud data warehouses, databases, and AI models—that Sigma connects to directly for every query.

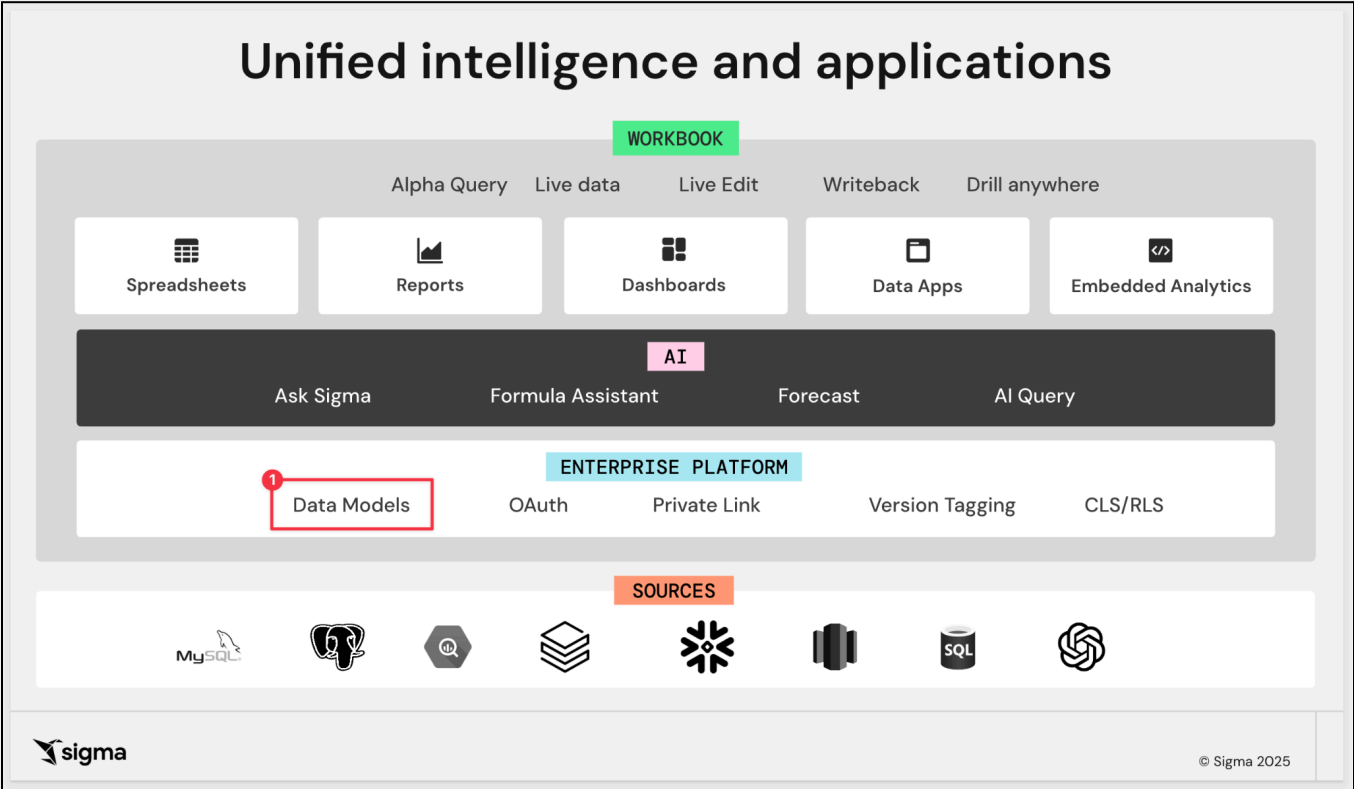


Figure 1

8: What's Next?

As data modeling matures, it's evolving beyond just defining structure—it's becoming the semantic foundation for every analytics experience. Sigma Data Models are already delivering that foundation today, providing a governed, reusable layer that **executes directly on your warehouse** and serves every type of user and workflow.

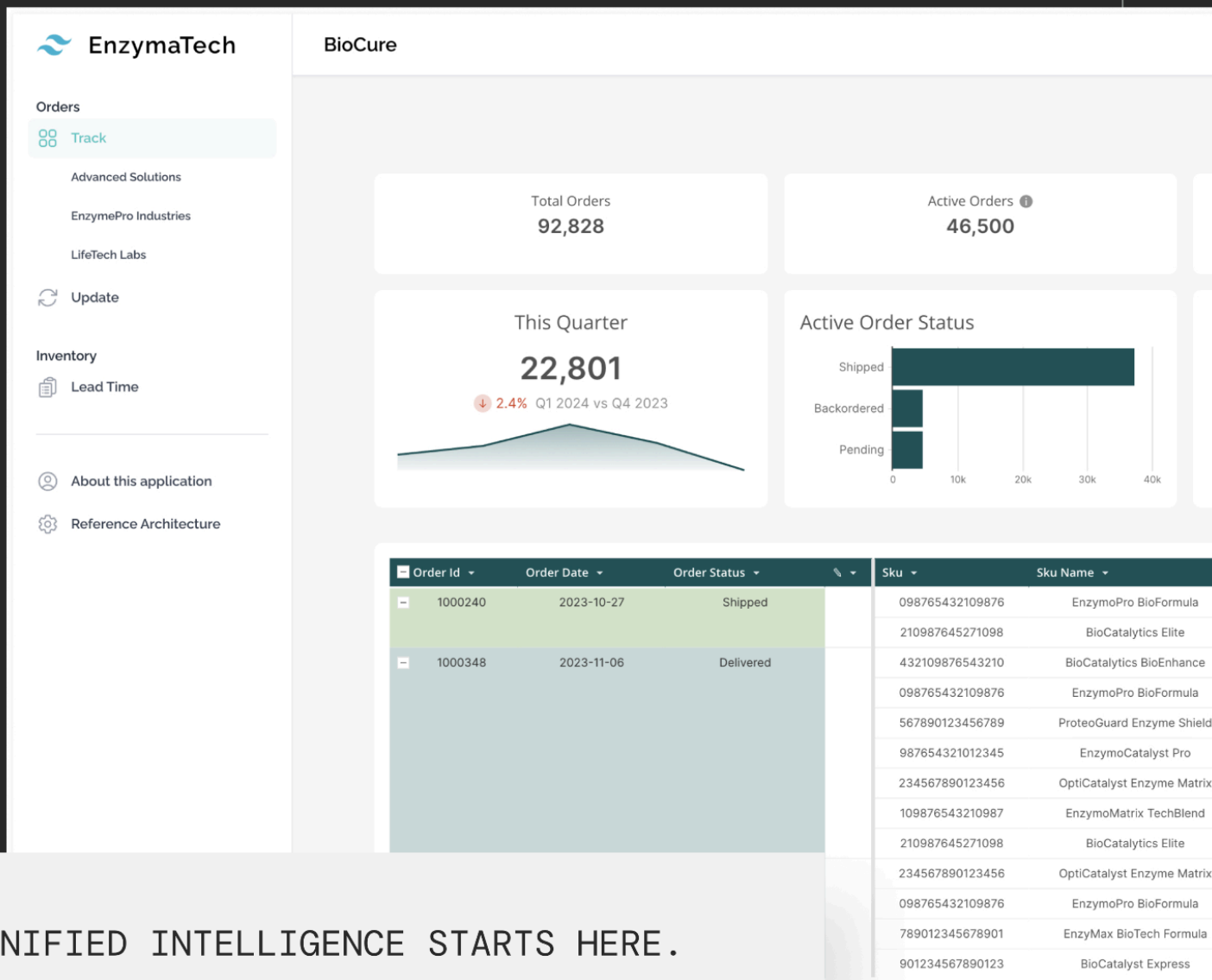
The broader analytics ecosystem is also moving toward more formalized semantics within the warehouse itself. Platforms like Snowflake, Databricks, and BigQuery are introducing capabilities to define metrics, relationships, and filters as first-class warehouse objects. Sigma is working closely with all leading cloud data warehouse providers to ensure that as these features emerge, they can integrate seamlessly with the governed models you've built in Sigma.

For now, the fastest path to delivering consistent, trusted, and business-friendly analytics is to model directly in Sigma. Doing so allows you to define and manage business logic in one place, reuse it everywhere, and keep it fully aligned with your warehouse's performance and security model. As warehouse-layer semantics mature, Sigma Data Models will be ready to extend and interoperate, ensuring your governance strategy remains future-proof.

BE SURE TO CHECK OUT ALL THE LATEST DEVELOPMENTS AT

[Sigma's First Friday Feature page!](#)

[Help Center Home](#) | [Sigma Community](#) | [Sigma Blog](#)



UNIFIED INTELLIGENCE STARTS HERE.

AI. Apps. Analytics.

Run granular analytics on billions of records without compromising speed or security—all from an intuitive interface anyone can use.