

Sigma Input Tables

Building data applications
with Write-back and Input
Tables

AUTHOR

Phil Ballai

Enterprise Architect

The Importance of Data Applications

Let's face it—traditional BI has run its course. It fails to deliver what users truly need for maximum efficiency and effectiveness in their roles. While BI vendors often tout “actionable analytics,” the reality is their tools fall short of enabling direct action.

In today's data-driven landscape, organizations spanning diverse industries increasingly rely on multiple data applications to drive decision-making and operational efficiency. These applications empower users to engage with data meaningfully, shifting from static reports to dynamic, real-time data interactions.

Industries such as finance, healthcare, retail, and manufacturing harness data applications to uncover insights, streamline processes, and elevate service delivery. Financial institutions use interactive dashboards to monitor market trends and manage portfolios; healthcare providers leverage data applications to track patient outcomes and refine treatment plans; and retailers employ these tools to analyze consumer behavior and optimize inventory management.

The Importance of Data Applications

The demand for interactive data applications stems from the need for:

Real-time insights: Today's fast-paced organizations thrive on current, actionable information. Our interactive data applications deliver real-time access and analysis, empowering swift, informed decision-making.

User engagement: By integrating interactive features like input fields, sliders, and intuitive menus, users effortlessly explore data, conduct hypothetical scenarios, and extract meaningful insights.

Customization and flexibility: Every business is unique, requiring adaptable solutions that seamlessly fit their workflows. Our interactive data applications offer customizable experiences, perfectly aligned with specific organizational needs and goals.

Enhanced collaboration: Foster teamwork and synergy with our data applications, which unify insights and discussions on a single platform. This collaborative approach ensures cohesive decision-making and shared understanding across teams.

Imagine accessing actionable insights and taking immediate action—all within a single application.

Welcome to Sigma and Write-back. With Sigma, you gain powerful tools to create workflows that turn the vision of actionable data applications into reality.

By bridging traditional BI with the data-driven needs of diverse business users, Sigma empowers organizations to unlock substantial benefits.

[Workato](#), an enterprise automation and integration platform, has streamlined complex workflows and achieved significant productivity gains since adopting Sigma. Learn more about Workato's success story [here](#).

Nick Bunick, Principal at [NewView Capital](#), shared his firsthand experience using Sigma during their due diligence process: "We had the privilege of leveraging Sigma during our due diligence. What impressed us most was how quickly we mastered the platform. Within minutes, we refined projections, visualized historical trends, and collaborated effectively with the Sigma team." Discover more about NewView Capital's journey with Sigma [here](#).

Addressing an unmet need

Workato, NewView Capital, and [numerous other innovators](#) have harnessed Sigma to pioneer new data applications centered around input tables.

Input tables stand as a cornerstone feature in crafting dynamic, interactive data applications—an exclusive hallmark of Sigma.

They serve as the crucial link between static data and user-driven interactivity, empowering users to input, adjust, and manipulate data seamlessly within the application. This capability revolutionizes conventional data analysis, fostering a more immersive and responsive user experience.

Critical attributes of input tables that underscore their indispensability in data applications include:

User input capabilities: At Sigma, we empower users to directly input data into our platform, driving real-time calculations, updating visualizations, and triggering workflows. This hands-on approach fosters interactive data analysis and enhances user engagement.

Data validation: Data accuracy is paramount. With built-in validation features, Sigma's input tables enforce custom rules and checks to uphold data quality, ensuring reliable insights.

Real-time updates: Updates made in Sigma's input tables propagate instantly across the platform, ensuring all stakeholders access the latest data. This dynamic capability boosts responsiveness and maximizes utility.

Integration with other features: Our input tables seamlessly integrate with Sigma's controls, visualizations, and external databases. This unified approach empowers users to build robust data applications tailored to diverse analytical needs.

Customization and flexibility: Customizable to meet specific requirements, Sigma's input tables offer flexibility in data entry, formatting, and interaction. This adaptability makes them ideal for a wide range of data-driven scenarios.

Minimize time to production: Sigma streamlines development cycles by enabling rapid creation and deployment of interactive data forms. This process minimizes reliance on extensive coding or IT support, while upholding stringent security and governance standards.

Across industries, the increasing demand for interactive data applications highlights the critical role of features such as input tables. Sigma's input tables empower user interactivity, ensure real-time updates, and seamlessly integrate, revolutionizing how organizations engage with and extract value from their data.

Getting started with Sigma Input Tables

How is data stored?

Before diving in, it's crucial to note: Sigma never stores customer data outside of your cloud data warehouse (CDW). Our Input Tables and Write-back functionality are essential features that empower Sigma data applications to securely send data back to your CDW for storage.

Here are the key points to understand:

1. Sigma never stores customer data outside the CDW.
2. When using Write-back, data is stored in the CDW only where you specify, as configured in the Sigma administrative interface, shown below.
3. Input table data is stored adjacent to other data, allowing it to be joined on key-value pairs if needed.
4. Sigma never overwrites data. Instead, it captures every save made by any user of an input table, maintaining a history of changes for tracking purposes.

Having grasped those points, let's dive into configuring Sigma and Snowflake to begin using input tables and discover their versatile applications.

For the most up-to-date details on supported warehouses, [see here](#).

Configuring write back

To showcase this, we'll utilize Sigma and Snowflake trial instances.

Our first step involves configuring resources within Snowflake to store data written back to the warehouse from Sigma.

START A FREE [SIGMA TRIAL](#) TODAY.

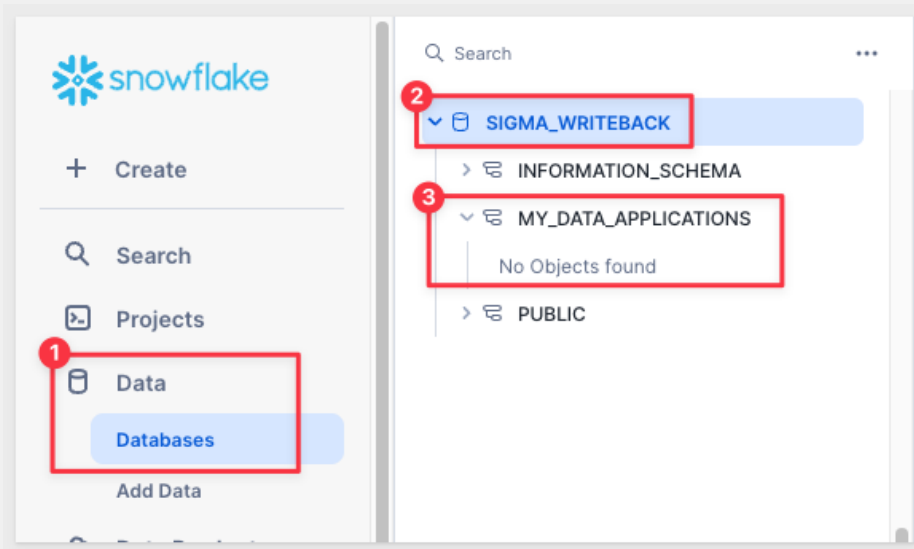
In Snowflake's interface, logged in as an "ACCOUNTADMIN", create a new worksheet and run the following script, which creates a new database and schema and grants the required usage to the role "ACCOUNTADMIN":

```
-- RUN STATEMENTS AS "ACCOUNTADMIN" AND USE THE "COMPUTE_WH"  
  
-- Create the database SIGMA_WRITEDB if it does not already exist  
CREATE DATABASE if not exists SIGMA_WRITEBACK;  
  
-- Switch to the newly created or existing database SIGMA_WRITEDB  
USE DATABASE SIGMA_WRITEBACK;  
  
-- Create a schema named QUICKSTART within the SIGMA_WRITEDB database:  
CREATE SCHEMA if not exists MY_DATA_APPLICATIONS;  
  
-- Grant the ACCOUNTADMIN role usage privileges on the database SIGMA_WRITEDB:  
GRANT USAGE ON DATABASE SIGMA_WRITEBACK TO ROLE ACCOUNTADMIN;  
  
-- Grant the ACCOUNTADMIN role usage, create table, create view, and create stage privileges on  
the MY_DATA_APPLICATIONS schema:  
  
GRANT USAGE, CREATE TABLE, CREATE VIEW, CREATE STAGE ON SCHEMA MY_DATA_APPLICATIONS TO ROLE  
ACCOUNTADMIN;
```

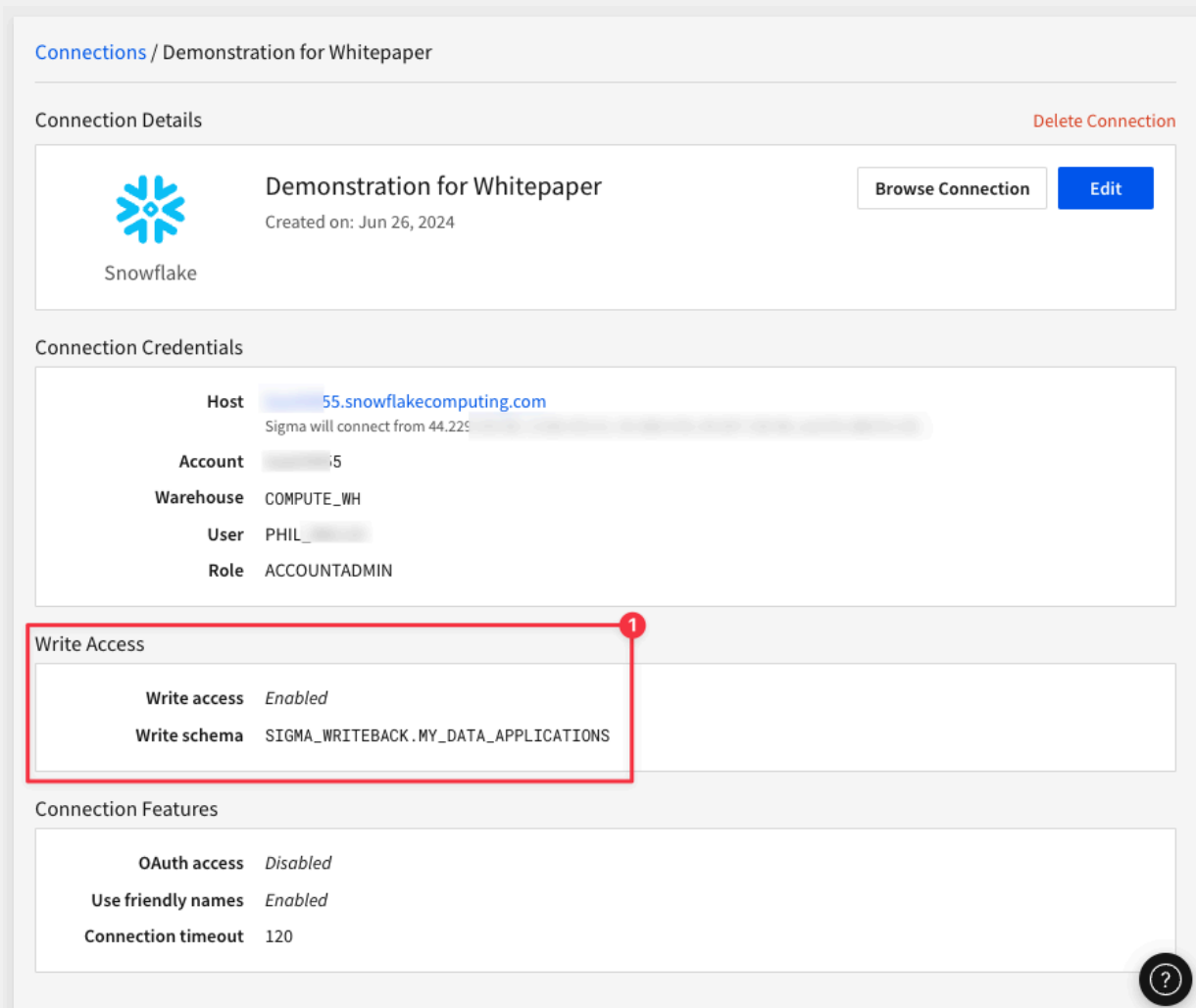
After running the script, you should see the result: "Statement executed successfully".

The screenshot displays the Snowflake web interface. At the top, a worksheet titled "Sigma Write-back Setup" is open. The SQL script from the previous block is pasted into the editor. A red box labeled '1' highlights the 'Run' button in the left sidebar. Another red box labeled '2' highlights the worksheet title. A third red box labeled '3' highlights the SQL script text. A fourth red box labeled '4' highlights the 'Share' button. A fifth red box labeled '5' highlights the 'Run All' button, which has a tooltip that says "Run All" and "% + shift + enter". Below the script, the 'Results' tab is active, showing a single row with the status "Statement executed successfully." A red box labeled '6' highlights this status message. On the right side, the 'Query Details' panel shows a query duration of 45ms.

This script results are a new database and schema to hold input table data in the warehouse:



We can now jump over to Sigma, log in as an **“Administrator,”** and configure Write-back using the database and schema we created above:



Once the connection is created, Sigma checks to ensure it is valid. That's it—we are ready to create our data application.

Note: Sigma also supports using [OAuth for write back enabled connections to Snowflake](#)

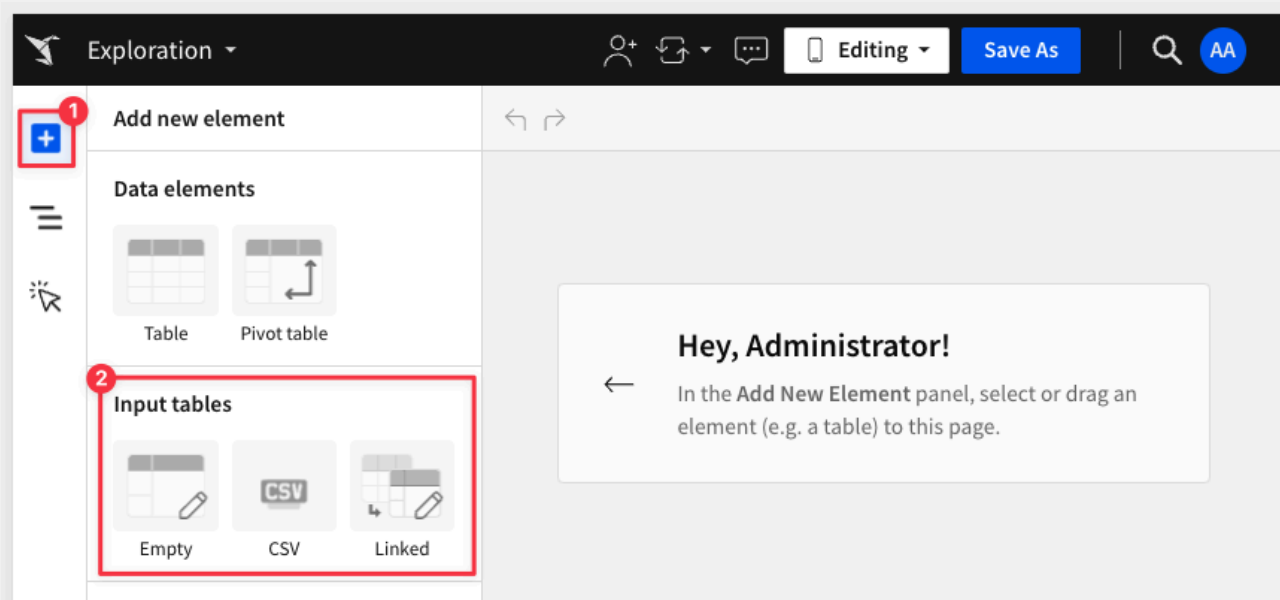
My first Sigma data application

Occasions arise when gathering even a modest amount of data from a Sigma user proves incredibly beneficial for the business. However, this does come with its own set of challenges:

- Will you have to build a new application to capture the data?
- When is the right time to capture the data?
- Will the users accept another tool?
- Once we have the data, how will it relate to existing warehouse data?
- Many other questions may arise!

Whether you're already leveraging Sigma or still relying on spreadsheets (*when you should be using Sigma*), input tables offer a powerful solution. Imagine creating a Sigma workbook enriched with the capability for users to input small data sets in real-time. Sigma takes care of the complex UI and data operations, seamlessly storing your data in the warehouse.

Within Sigma workbooks, users can choose from three types of input tables. The first two types function much like traditional spreadsheets, making them intuitive to us.



1. **Empty input tables** are blank tables that support data entry in standalone tables independent of existing data. You can edit data at the cell level and add editable rows and columns to construct your table as you see fit. You can also copy/paste values in up to 12,500 cells at once (500 rows and 25 columns).
2. **CSV input tables** support data entry in standalone tables; however, they allow you to pre-populate the table with uploaded CSV data (max 200 MB). You can then edit the uploaded data at the cell level and add editable rows and columns to construct your table as you see fit.
3. **Linked input tables** support data entry alongside existing data from other elements in the same workbook. Typically, these existing elements are other data from the warehouse that are related in some way. As a child element, a linked input table includes one or more linked columns that reference data in the parent element.

In all three types, data is written to the CDW as previously described.

Allowing users to add or supplement warehouse data opens a world of possibilities. We will demonstrate how this can be done using a very simple example.

Example use case

PROBLEM STATEMENT

Consider this scenario: Our accounting team requires real-time shipment status updates and needs to collect comments or notes from the shipping department. However, they're reluctant to grant shipping clerks access to our accounting system. Given our current budget constraints, we're searching for a solution that doesn't involve investing in new software or upgrading existing systems.

SIGMA INPUT TABLES ARE THE SOLUTION

Since Sigma is already in play, we can harness its power to swiftly and effortlessly tackle this issue, all without concerns about extra expenses, developers, or even security and governance.

Users know how to use Sigma, have permissions based on their role, and the data will be stored in the existing warehouse, so all that is handled already. Awesome!

We will use sample data provided to all Snowflake customers and join table data from the warehouse as our source of invoice data.

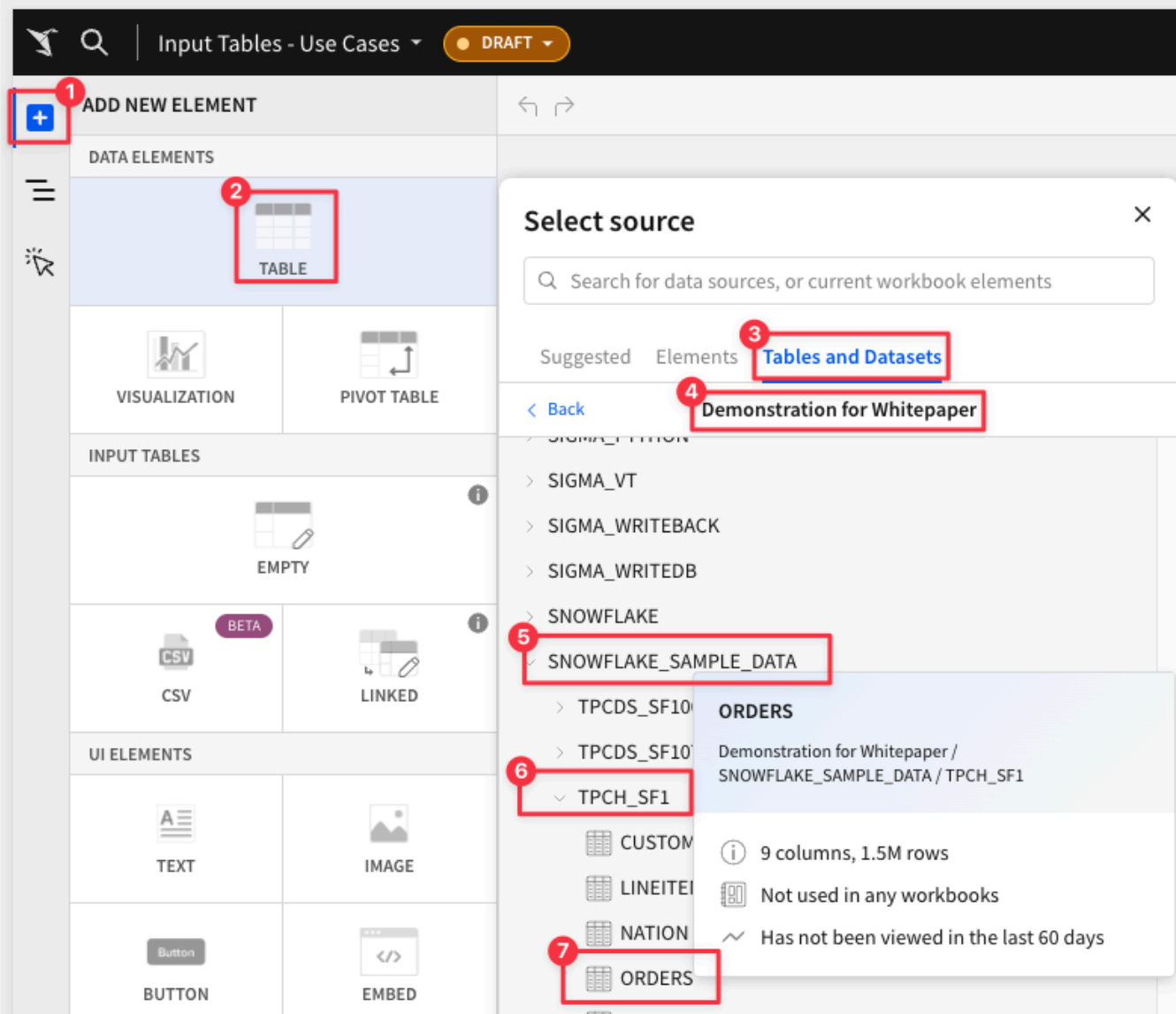
Note: Sigma provides all customers a sample dataset as well, but we will use the Snowflake sample so that we can look “under-the-hood” at the Write-back data later. Since the Sigma sample data is hosted by Sigma, we would not be able to do that.

Along the way, we will also expose you to just a few of the many amazing features that Sigma provides.

There are many ways to solve problems with Sigma, and while we don’t have space to show them all, we want you to see just how fast and flexible Sigma can be while solving this use case.

1: In Sigma, create a new workbook named “**Input Tables – Use Cases**”

2: Add a new table element to the page and select (or search for and select) the table in “**SNOWFLAKE_SAMPLE_DATA**” > “**TPCH_SF1**” > “**ORDERS**”:



Note: We could have also just asked Sigma to show all the tables named "Orders" across all connections to find it fast!

3: Rename the new table to "Warehouse Data - Read Only" and the page name to "Data."

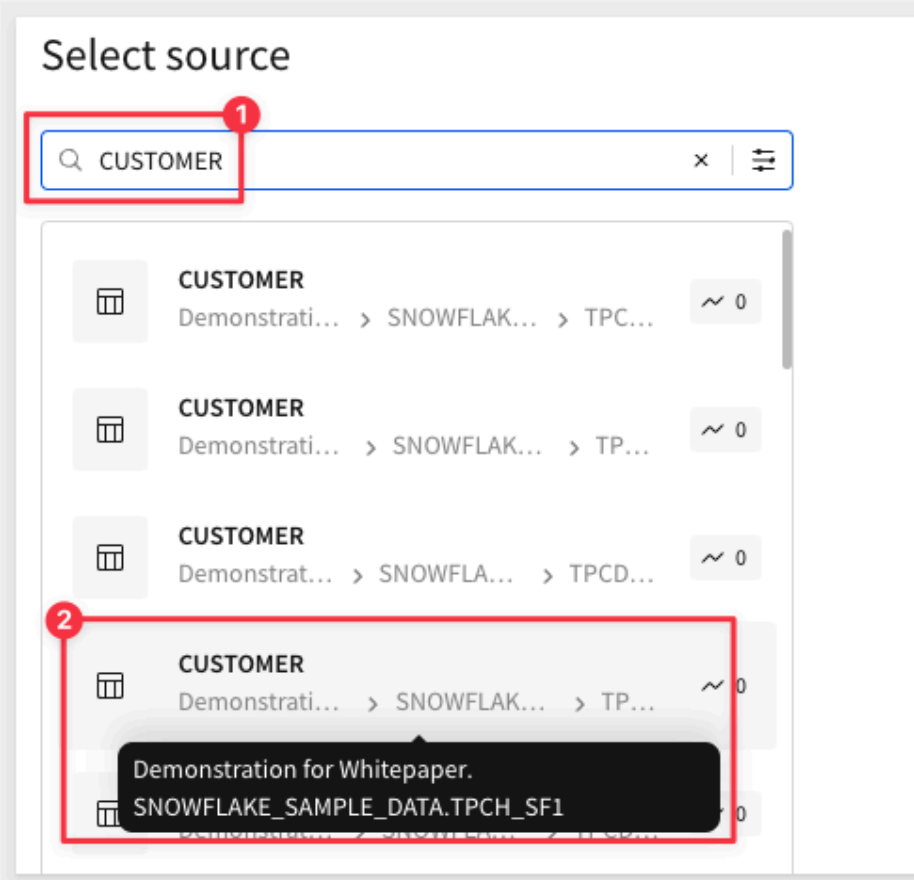
We want to enrich this data by joining it with the customer information for each order. Click the table to select it and join the additional table as shown below.

The screenshot shows the Sigma Administrator interface. On the left, there is a 'GROUPINGS' panel with instructions to drag and drop columns. Below it is a 'COLUMNS' and 'METRICS' section with an 'ADD COLUMN' button. A list of columns is shown, including 'O Orderkey', 'O Custkey', 'O Orderstatus', 'O Totalprice', 'O Orderdate', 'O Orderpriority', 'O Cler...', 'O Ship...', and 'O Com...'. A context menu is open over the 'O Orderkey' column, with options: 'View table', 'Replace table', 'Change source', 'TRANSFORM', 'Join', and 'Union'. The 'Join' option is highlighted with a red box and a red circle containing the number '2'. The 'Union' option is also highlighted with a red box and a red circle containing the number '1'. At the bottom left, a table named 'ORDERS' is selected, also highlighted with a red box and a red circle containing the number '1'. On the right, a table titled 'Warehouse Data - Read Only' is displayed with the following data:

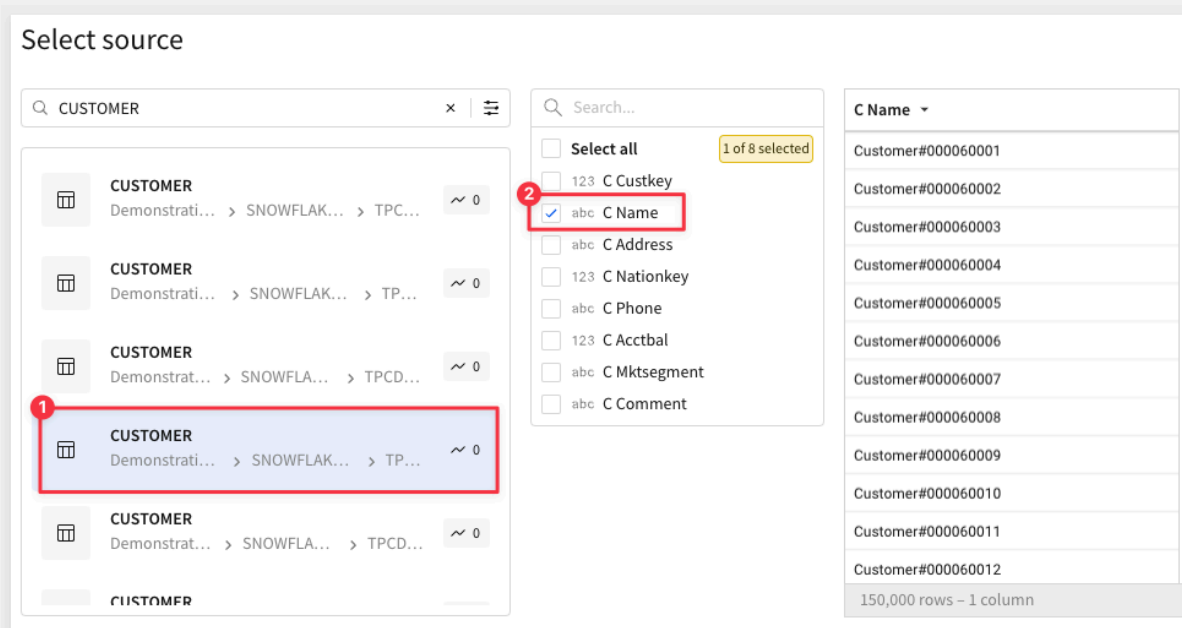
O Orderkey	O Custkey	O Orderstatus	O Totalpri
3000001	145618	F	301
3000002	1481	O	2979
3000003	127432	O	3454
3000004	47423	O	1359
3000005	84973	F	2099
3000006	135136	O	1401
3000007	78841	F	2986
3000032	124576	F	175
3000033	30247	F	46
3000034	5498	F	3483
3000035	1537	O	62
3000036	28564	F	460
3000037	46702	O	1912
3000038	106394	O	1086
3000039	4087	F	1490
3000064	121798	O	602
3000065	4583	O	119

Below the table, a summary bar indicates 'SUMMARY ^ 1,500,000 rows - 9 columns'.

Sigma's source selector lets us search for tables named "customer" and select the one we want.



After clicking on the correct “**CUSTOMERS**” table, we can also prune the list of columns we want to include. We will keep this bare minimum and select just the “**C_NAME**” column.



Next, we have the option to adjust the join type, join keys, and examine the matches. Make sure that the Join keys are matching on **“Customer Id”**.

In this case, there are 50,004 customers who have no orders. This is normal and expected; click **“Preview Output.”**

The screenshot shows the 'Create Join' configuration in Sigma. On the left, the 'SOURCES' panel lists 'ORDERS' and 'CUSTOMER'. The 'Join with' dropdown is set to 'CUSTOMER'. The 'Join type' is 'Left outer join'. The 'Join keys' are '123 O Custkey' and '123 C Custkey'. The 'Preview Output' button is highlighted. A table below shows the join output with columns: O Orderkey, O Custkey, O Orderstatus, O Totalprice, O Orderdate, O Orderpriority, O Clerk, O Shippriority, and O Comment.

O Orderkey	O Custkey	O Orderstatus	O Totalprice	O Orderdate	O Orderpriority	O Clerk	O Shippriority	O Comment
3000001	145618	F	30175.88	1992-12-17 00:00:00	4-NOT SPECIFIED	Clerk#000000141		0 l packages. furiously careful instructions gr
3000002	1481	O	297999.63	1995-07-28 00:00:00	1-URGENT	Clerk#000000547		0 carefully unusual dependency
3000003	127432	O	345438.38	1997-11-04 00:00:00	5-LOW	Clerk#000000488		0 n packages boost slyly bold deposits. depo
3000004	47423	O	135965.53	1996-06-13 00:00:00	4-NOT SPECIFIED	Clerk#000000004		0 nts wake carefullv final decovs. quicklv fina

Sigma shows us the lineage to make sure it looks correct (these can become quite complex) and prune the column list further. We don't need many columns for this use case, so we will limit to what is shown below.

The screenshot shows the 'Create Join' configuration in Sigma. The 'SOURCES' panel lists 'ORDERS' and 'CUSTOMER'. The 'Join with' dropdown is set to 'CUSTOMER'. The 'Join type' is 'Left outer join'. The 'Join keys' are '123 O Custkey' and '123 C Custkey'. The 'Preview Output' button is highlighted. A table below shows the join output with columns: O Orderkey, O Totalprice, O Orderdate, and C Name.

O Orderkey	O Totalprice	O Orderdate	C Name
3000001	30175.88	1992-12-17 00:00:00	Customer#000145618
3000002	297999.63	1995-07-28 00:00:00	Customer#000001481
3000003	345438.38	1997-11-04 00:00:00	Customer#000127432
3000004	135965.53	1996-06-13 00:00:00	Customer#000047423
3000005	209937.09	1992-09-12 00:00:00	Customer#000084973
3000006	140186.32	1996-09-26 00:00:00	Customer#000135136
3000007	298655.07	1992-04-13 00:00:00	Customer#000078841

Shipping has requested that we also show the part name so that the clerk can visually verify that while looking at the shipment being final packed.

That is really simple and is just repeating the workflow we just did, adding the two required tables.

We do this by clicking on the “+” icon in the image above.

We need to add the “LINEITEM” table and the “ORDERS” table, joining them on “Order Key”.

The screenshot shows the 'Create Join' interface. On the left, under 'SOURCES', the 'LINEITEM' table is highlighted with a red box and a '1' icon. The central panel is titled 'Create Join' and contains the following fields:

- Join with:** ORDERS
- Selected source:** LINEITEM
- Join type:** Left outer join
- Join keys:** 123 O Orderkey = 123 L Orderkey

On the right, there are two summary tables:

ORDERS	LINEITEM
Total key count 1,500,000	Total key count 1,500,000
Keys with no matches 0	Keys with no matches 0
Keys with 2+ matches 1,285,828	Keys with 2+ matches 1,285,828

Below these are two key selection tables:

O Orderkey	L Orderkey
1	1
2	2
3	3
4	4
5	5

To get the actual part name, we need to add the “PART” table and the “PART” table, joining them on “PartKey”.

The screenshot shows the 'Create Join' interface. On the left, under 'SOURCES', the 'PART' table is highlighted with a red box and a '1' icon. The central panel is titled 'Create Join' and contains the following fields:

- Join with:** LINEITEM
- Selected source:** PART
- Join type:** Left outer join
- Join keys:** 123 L Partkey = 123 P Partkey

On the right, there are two summary tables:

LINEITEM	PART
Total key count 200,000	Total key count 200,000
Keys with no matches 0	Keys with no matches 0
Keys with 2+ matches 200,000	Keys with 2+ matches 200,000

Below these are two key selection tables:

L Partkey	P Partkey
1	1
2	2
3	3
4	4
5	5

Click “Preview Output”.

We have a revised lineage and the six source columns we need for our data application.

Create Join Cancel Done

Search

ORDERS
CUSTOMER
LINEITEM
PART

Select all
123 P Partkey
 abc P Name
abc P Mfgr
abc P Brand
abc P Type

1 of 9 selected

O Orderkey	O Totalprice	O Orderdate	C Name	L Partkey	P Name
4200002	256838.41	1997-04-14 00:00:00	Customer#000129376	128310	ivory tomato spring cornflower lime
4200002	256838.41	1997-04-14 00:00:00	Customer#000129376	105053	dim saddle orchid purple magenta
4200002	256838.41	1997-04-14 00:00:00	Customer#000129376	22622	khaki metallic bisque burnished antique
4200002	256838.41	1997-04-14 00:00:00	Customer#000129376	6562	spring hot blanched chocolate orange
4200002	256838.41	1997-04-14 00:00:00	Customer#000129376	87247	firebrick royal dim smoke grey
4200002	256838.41	1997-04-14 00:00:00	Customer#000129376	157605	wheat goldenrod antique cyan tomato
4200002	256838.41	1997-04-14 00:00:00	Customer#000129376	54648	peru chocolate chiffon bisque misty
4200036	185232.08	1994-10-16 00:00:00	Customer#000002356	125803	mint forest honeydew light beige
4200036	185232.08	1994-10-16 00:00:00	Customer#000002356	162160	bisque powder rose papaya spring
4200036	185232.08	1994-10-16 00:00:00	Customer#000002356	17136	lavender medium cornsilk dodger puff
4200036	185232.08	1994-10-16 00:00:00	Customer#000002356	4374	lavender royal sandy cyan red
4200103	182718.54	1007-12-03 00:00:00	Customer#000106213	25046	blue medium dark cream hot

6,001,215 rows - 6 columns

This source data from accounting will operate as the base data for our input table to come. We did this in just a few minutes; the Sigma interface is always working to make your job easier. Click **“Done”**.

Warehouse Data - Read Only Administrator Account

Drag and drop columns from the Columns tab to create groupings

GROUPINGS

COLUMNS METRICS

ADD COLUMN

123 Invoice Id
123 Invoice Date
123 Quantity
123 Unit Price
abc Billing Country
abc Customer Id
abc Customer Id (CUSTOMERS)
abc Name
abc Company

Warehouse Data - Read Only

Invoice Id	Invoice Date	Quantity	Unit Price	Billing Country	Customer Id	Customer Id (CUSTOMERS)
1575036	2017-10-09 11:45:00	17	1.74	United Kingdom	15249	15249
1546771	2016-07-09 07:00:00	4	35.18	United Kingdom	17515	17515
1567372	2015-12-18 08:00:00	25	3.24	United Kingdom	14191	14191
1560698	2015-09-05 07:00:00	29	10.69	United Kingdom	16353	16353
1551893	2013-07-09 10:14:02	43	4.42	United Kingdom	17675	17675
1553505	2016-12-20 08:00:00	8	20.7	United Kingdom	12949	12949
1565218	2015-03-11 10:52:02	12	8.67	United Arab Emirates	12739	12739
1569520	2016-06-09 11:32:00	27	12.85	United Kingdom	17667	17667
1579135	2018-03-10 08:00:00	7	1.62	United Kingdom	18096	18096
1540940	2018-02-08 11:19:02	10	9.94	United Kingdom	18109	18109
1566033	2015-10-09 10:14:02	4	8.58	United Kingdom	16031	16031
1551539	2013-05-09 10:47:02	205	2.19	France	12598	12598
1539016	2016-04-10 07:00:00	18	8.62	United Kingdom	13418	13418
1545696	2011-08-09 10:28:02	32	3.64	United Kingdom	15696	15696
1569843	2016-08-09 11:29:00	2	1	United Kingdom	14449	14449
1571036	2016-12-31 08:00:00	239	5.09	United Kingdom	17567	17567
1545014	2017-09-15 07:00:00	147	1.74	United Kingdom	18223	18223

SUMMARY 7,586,908 rows - 9 columns

The next step is to add our input table. However, when we add a new linked input table, we will need to select a column to act as a unique identifier.

This will present a problem; most invoices contain more than one line item, so we need to have an identifier that will be unique per row.

Are we out of luck? Do we need to engage a developer? **Nope.**

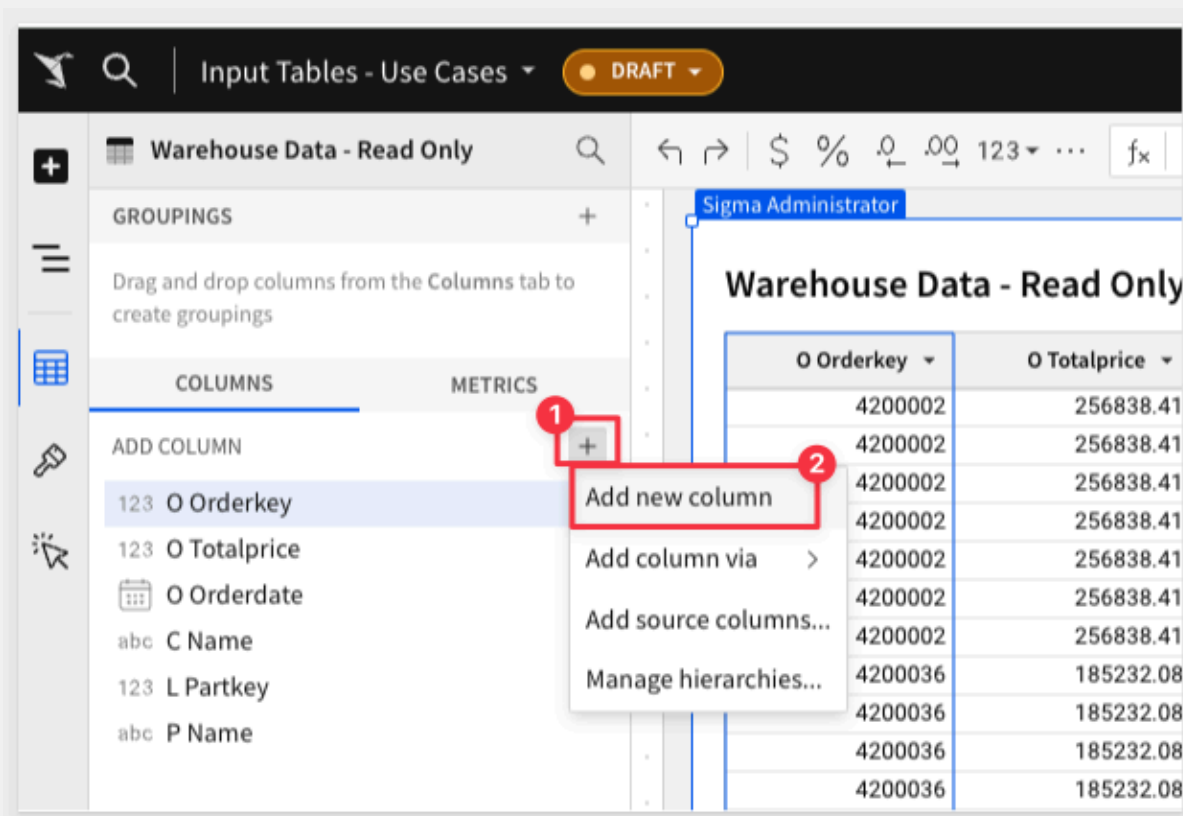
A cool feature of Sigma is its ability to enrich warehouse data on the fly, in a user's browser. This allows us to add a column to our source data that will provide the necessary uniqueness to rows, but not alter the warehouse data in any way.

To learn more about how Sigma does this and more to allow us to operate at massive scale without sacrificing performance and keeping costs in check, [read this QuickStart](#).

Let's not add the input table just yet.

4: On the Data page, click to add a new column, name the column "**Row Number**", and set its formula to:

```
=RowNumber()
```



Drag the column to the first position. You may want to sort the "**Row Number**" column, but it is not required. We can apply styling and formatting to columns as shown below.

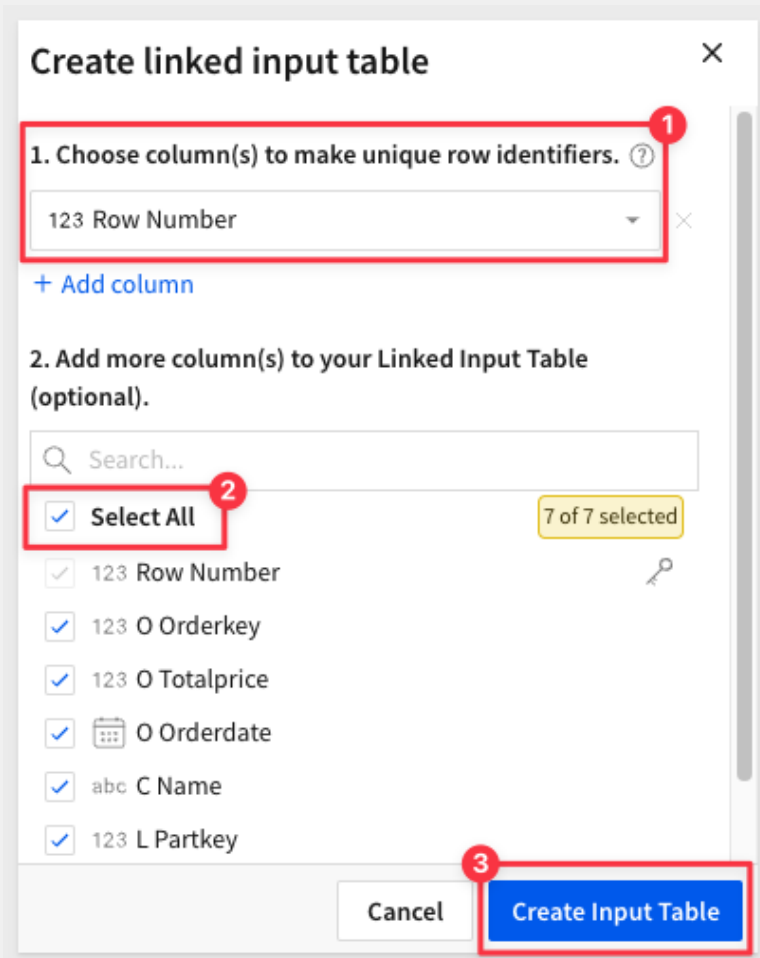
Row Number	O Orderkey	O Totalprice	O Orderdate	C
1	1736421	210955.63	1993-01-14 00:00:00	Ci
2	1736485	168183.29	1992-12-25 00:00:00	Ci
3	1736485	168183.29	1992-12-25 00:00:00	Ci
4	1736514	56354.94	1997-01-17 00:00:00	Ci
5	1736641	209573.69	1996-07-18 00:00:00	Ci
6	1736743	277317.66	1992-12-24 00:00:00	Ci
7	1736743	277317.66	1992-12-24 00:00:00	Ci
8	1736838	221631.92	1997-09-14 00:00:00	Ci
9	1736928	226949.34	1998-02-11 00:00:00	Ci
10	1736932	185358.67	1997-06-08 00:00:00	Ci
11	1736997	82251.08	1998-01-21 00:00:00	Ci

Now we are ready for our input table.

5: Add a new page to the workbook called **“Shipment Updates”** Add a **“Linked Input Table”** to the page and select the table on our **“Data”** page as shown below.

The screenshot shows the Sigma Computing interface. At the top, there's a header with 'Invoice Status' and a 'DRAFT' button. On the left, a sidebar contains navigation icons. The main area shows the 'Add new element' menu. A red box with a '1' highlights the '+' icon. Below it, there are sections for 'Data elements' (Table, Pivot table), 'Input tables' (Empty, CSV, Linked), and 'Code' (Python). The 'Input tables' section has 'Linked' selected, marked with a red box and '2'. A 'Select source' dialog is open, showing a search bar and a list of sources. 'Elements' is highlighted in blue with a red box and '3'. 'Data' is highlighted with a red box and '4'. 'Warehouse Data - Read Only' is highlighted with a red box and '5'. At the bottom, there are tabs for 'Shipment Updates' and 'Data'.

Select the "Row Number" column for the unique identifier and **select all other columns**. Click "Create Input Table"



We now have our first input table ready for the next step.

6: Let's refine the input table a little.

We can rename each column by double-clicking on the column header, to make the name more user friendly. Remember, these changes do not affect the source warehouse data in any way. We can also hide the "Row Number" and "L Partkey" columns by opening the column menu for each and selecting "Hide".

Columns can be centered, formatted as currency and more. For example, the "Order Date" column shows the order time and we really don't need that. We can easily truncate that column to just the day.

New Linked Input Table EDITABLE

	Order	Customer Number	Order Date		Total Value
1	2386439	Customer#000121384	1995-12-22 00:00		\$241,455.24
2	2386439	Customer#000121384	1995-12-22 00:00		\$241,455.24
3	2386530	Customer#000053437	1992-07-03 00:00		\$315,522.68
4	2386530	Customer#000053437	1992-07-03 00:00		\$315,522.68
5	2386530	Customer#000053437	1992-07-03 00:00		\$315,522.68
6	2386662	Customer#000092995	1994-05-31 00:00		\$182,705.25
7	2386722	Customer#000056492	1995-05-20 00:00		\$248,401.96
8	2386727	Customer#000029491	1997-11-27 00:00		\$241,254.84
9	2386754	Customer#000088033	1994-11-17 00:00		\$321,840.74
10	2386757	Customer#000103489	1996-07-19 00:00		\$207,060.27
11	2386820	Customer#000149207	1997-02-11 00:00		\$108,898.33
12	2386820	Customer#000149207	1997-02-11 00:00		\$108,898.33
13	2386918	Customer#000025720	1995-02-04 00:00		\$314,395.19
14	2386918	Customer#000025720	1995-02-04 00:00		\$314,395.19
15	2386976	Customer#000001903	1994-11-27 00:00		\$160,103.13
16	2386981	Customer#000023182	1997-12-11 00:00:00	drab almond snow steel li...	\$163,166.69
17	2387201	Customer#000109901	1993-06-27 00:00:00	nuff frosted lime linen c...	\$318,956.71

SUMMARY ^ 6,001,215 rows - 6 columns

Our input table now looks like this.

New Linked Input Table EDITABLE

	Order	Customer Number	Order Date	Part Name	Total Value	Text	
1	2386439	Customer#000121384	1995-12-22	drab slate olive moccasin ...	\$241,455.24		
2	2386439	Customer#000121384	1995-12-22	floral navy light spring me...	\$241,455.24		
3	2386530	Customer#000053437	1992-07-03	honeydew yellow midnight...	\$315,522.68		
4	2386530	Customer#000053437	1992-07-03	salmon drab wheat spring ...	\$315,522.68		
5	2386530	Customer#000053437	1992-07-03	sandy linen royal tomato c...	\$315,522.68		
6	2386662	Customer#000092995	1994-05-31	white cornflower linen fire...	\$182,705.25		
7	2386722	Customer#000056492	1995-05-20	magenta tan burnished sie...	\$248,401.96		
8	2386727	Customer#000029491	1997-11-27	frosted cornsilk metallic tu...	\$241,254.84		
9	2386754	Customer#000088033	1994-11-17	beige hot grey yellow pow...	\$321,840.74		
10	2386757	Customer#000103489	1996-07-19	forest dodger pink tan bla...	\$207,060.27		
11	2386820	Customer#000149207	1997-02-11	medium burlywood chiffo...	\$108,898.33		
12	2386820	Customer#000149207	1997-02-11	purple rosy cyan deep ghost	\$108,898.33		
13	2386918	Customer#000025720	1995-02-04	plum coral turquoise sprin...	\$314,395.19		
14	2386918	Customer#000025720	1995-02-04	grey chartreuse dim salmo...	\$314,395.19		
15	2386976	Customer#000001903	1994-11-27	drab forest lace midnight ...	\$160,103.13		
16	2386981	Customer#000023182	1997-12-11	drab almond snow steel li...	\$163,166.69		
17	2387201	Customer#000109901	1993-06-27	nuff frosted lime linen cho...	\$318,956.71		

SUMMARY ^ 6,001,215 rows - 6 columns

Shipping asked for an easy way to identify the higher value shipments. Since we don't have guidance on what constitutes a **"high value shipment"**, we can use column details on the **"Total Value"** column to see what the statistics show us.

Customer Number	Order Date	Part Name	Total Value	Text
Customer#000121384	1995-12-22	drab slate olive moccasin ...	\$241,455.24	
Customer#000121384	1995-12-22	floral navy light spring me...	\$241,455.24	

Column Details

ABOUT

123 Total Value

TOP VALUES

\$217,595.19	24
\$250,917.40	21
\$252,569.97	21
\$254,828.07	21
\$329,825.01	21
\$155,832.34	20
\$188,214.57	20
\$188,963.15	20
\$197,460.49	20
\$206,757.38	20
\$217,595.19	20

SUMMARY

Values	100.0%	6,001,215
Nulls	0.0%	0
Row count	100.0%	6,001,215
Distinct values	24.4%	1,464,556

STATISTICS

Minimum	\$857.71
25th percentile	\$128,265.19
Median	\$189,705.24
75th percentile	\$248,145.95
Maximum	\$555,285.16
Average	\$189,034.40
Standard deviation	\$83,390.59
Variance	\$6,953,990,385.29

Learn more about [Columns](#)

With this data, we can consider values that are significantly above the median and closer to the upper percentiles. Here are a few approaches we can consider.:

Use 75th Percentile as a Threshold:

The 75th percentile value is \$248,145.95. This indicates that 25% of the orders are above this amount. Using this value as a threshold can be a good indicator of “**high value**” orders.

Use Maximum Value:

The maximum order value is \$555,285.16. Orders closer to this maximum value can be considered exceptionally high value.

Use Standard Deviation Above the Mean:

The average (mean) order value is \$189,034.40, with a standard deviation of \$83,390.59. Orders that are one standard deviation above the mean ($\$189,034.40 + \$83,390.59 = \$272,425$) can be considered high value. This would mean orders above approximately \$272,425 are high value.

We could do this easily by adding a calculated column, and using the Sigma function for [standard deviation](#).

Let's keep it simple for now and use option one. The 75th percentile value is a commonly used threshold for high value, as it naturally divides the top 25% of orders from the rest.

We want to make the interface really simple for the users, so we will add a toggle control that shows only the high value orders, with a single click.

Add a new calculation column and set it's formula to:

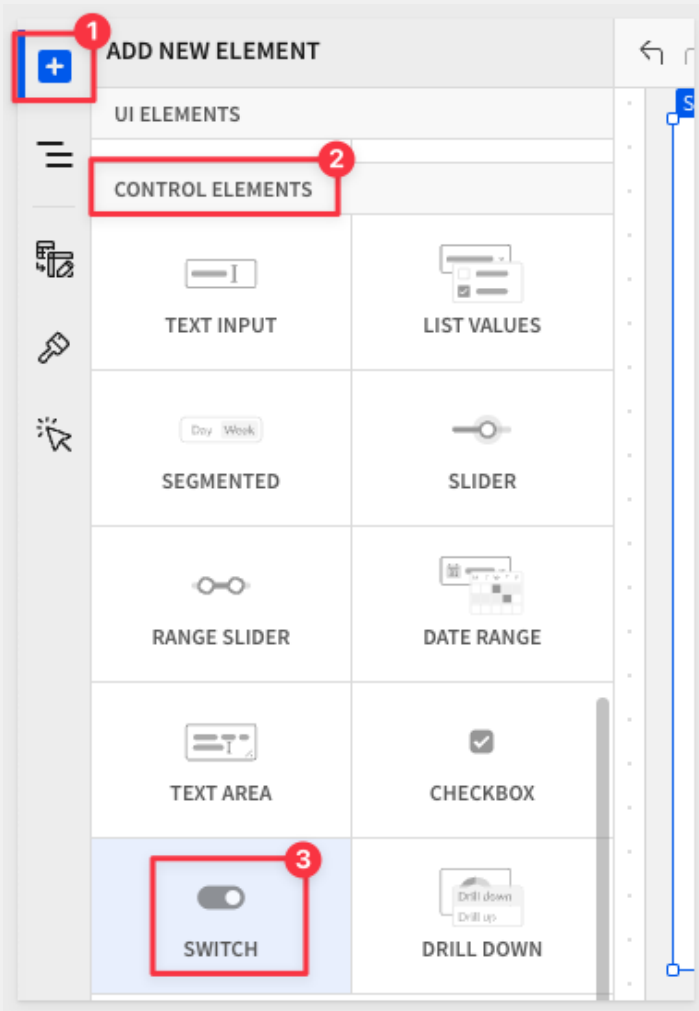
```
If([Total Value] > 248145.95, True, False)
```

This formula is pretty self-explanatory but if the value of **"Total Value"** in the adjacent cell is greater than the 75th percentile value (as determined by the column details earlier), the cell is set to **"True"**. Otherwise it is **"False"**.

The screenshot shows a Sigma table interface. At the top, a formula bar contains the formula `If([Total Value] > 248145.95, True, False)`. Below the formula bar is a table with the following columns: Order Date, Part Name, Total Value, and High Value. The 'High Value' column contains Boolean values (True or False) based on the 'Total Value' column. A red box highlights the formula bar, and another red box highlights the 'High Value' column header and its data cells. A red circle with the number '1' is placed above the 'High Value' column header, and a red circle with the number '2' is placed below the formula bar.

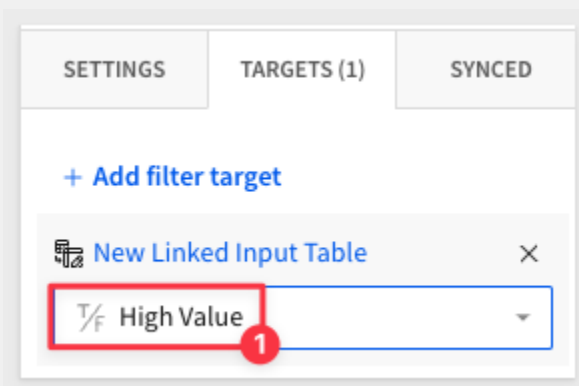
Order Date	Part Name	Total Value	High Value
1993-01-25	saddle navy orchid goldenr...	\$25,446.83	False
1995-10-23	aquamarine antique dodg...	\$233,591.11	False
1995-10-23	dim beige steel puff sandy	\$233,591.11	False
1993-10-09	magenta beige lace black l...	\$263,200.57	True
1993-08-02	peach papaya ivory chiffo...	\$285,424.38	True
1993-08-02	cyan indian light dim green	\$285,424.38	True
1993-08-17	linen olive drab yellow snow	\$154,333.25	False
1993-08-17	orange metallic orchid dar...	\$154,333.25	False
1993-12-17	tan pale metallic purple ne	\$154,681.01	False

Next we add a **"Switch control"** to toggle the input table, based on **"High Value"**.



We set the controls “**Target**” to the input table.

And specify the “**High Value**” column.



With the control set to “**True**” we only show the orders that are above the 75th percentile. Notice that the row count is reduced to 1.5M rows from the previous 6M.

We also can hide the "High Value" column since users don't need to see it to have it work with the switch control.

Show High Value Orders Only
 False True

New Linked Input Table EDITABLE

	Order	Customer Number	Order Date	Part Name	Total Value	+
1	2377825	Customer#000098993	1993-10-09	magenta beige lace black l...	\$263,200.57	
2	2377991	Customer#000104035	1993-08-02	peach papaya ivory chiffo...	\$285,424.38	
3	2377991	Customer#000104035	1993-08-02	cyan indian light dim green	\$285,424.38	
4	2378177	Customer#000032671	1994-04-12	aquamarine goldenrod pin...	\$280,596.36	
5	2378177	Customer#000032671	1994-04-12	mint chartreuse cornsilk n...	\$280,596.36	
6	2378212	Customer#000103771	1995-05-07	seashell azure smoke ivor...	\$268,132.45	
7	2378311	Customer#000121619	1993-03-27	plum smoke aquamarine o...	\$275,720.70	
8	2378912	Customer#000001492	1992-11-30	azure seashell metallic wh...	\$267,129.22	
9	2379683	Customer#000132212	1997-05-30	midnight steel ivory thistle...	\$314,202.74	
10	2379714	Customer#000076711	1997-01-14	blanched puff dark honeyd...	\$271,835.02	
11	2379714	Customer#000076711	1997-01-14	navajo magenta blanched ...	\$271,835.02	
12	2380130	Customer#000065300	1994-07-22	chartreuse navy firebrick h...	\$366,394.85	
13	2380167	Customer#000073861	1997-11-30	aquamarine green plum or...	\$290,769.00	
14	2380260	Customer#000117818	1995-08-02	tomato royal slate chocola...	\$261,274.68	
15	2380482	Customer#000122143	1995-10-26	forest almond papaya oliv...	\$308,080.99	
16	2380966	Customer#000070957	1996-10-06	drab seashell bisque sand...	\$280,097.20	
17	2381284	Customer#000069548	1996-06-09	ivory thistle metallic nowd...	\$275,674.99	

SUMMARY ^ 1,500,298 rows - 5 columns

By the way, did you notice that we have been working with around 6 million rows of data like it was nothing. That is the power of Sigma.

We are done with the basic cleanup operation.

7: Now we can add the columns we want shipping to use for capturing the line item status.

Click the "+" after the last column in the input table and add a new "Text" column.

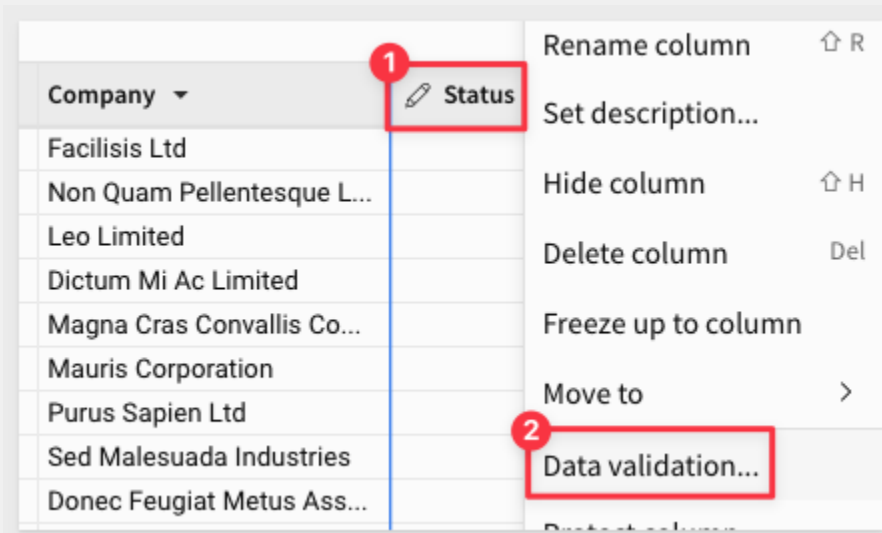
New Linked Input Table EDITABLE

	Order	Customer Number	Order Date	Part Name	Total Value	+
1	2377734	Customer#000078581	1993-01-25	saddle navy orchid golden...	\$25,446.83	
2	2377792	Customer#000133933	1995-10-23	aquamarine antique dodg...	\$233,591.11	
3	2377792	Customer#000133933	1995-10-23	dim beige steel puff sandy	\$233,591.11	
4	2377825	Customer#000098993	1993-10-09	magenta beige lace black l...	\$263,200.57	
5	2377991	Customer#000104035	1993-08-02	peach papaya ivory chiffo...	\$285,424.38	
6	2377991	Customer#000104035	1993-08-02	cyan indian light dim green	\$285,424.38	
7	2378023	Customer#000018992	1993-08-17	linen olive drab yellow snow	\$154,333.25	
8	2378023	Customer#000018992	1993-08-17	orange metallic orchid dar...	\$154,333.25	

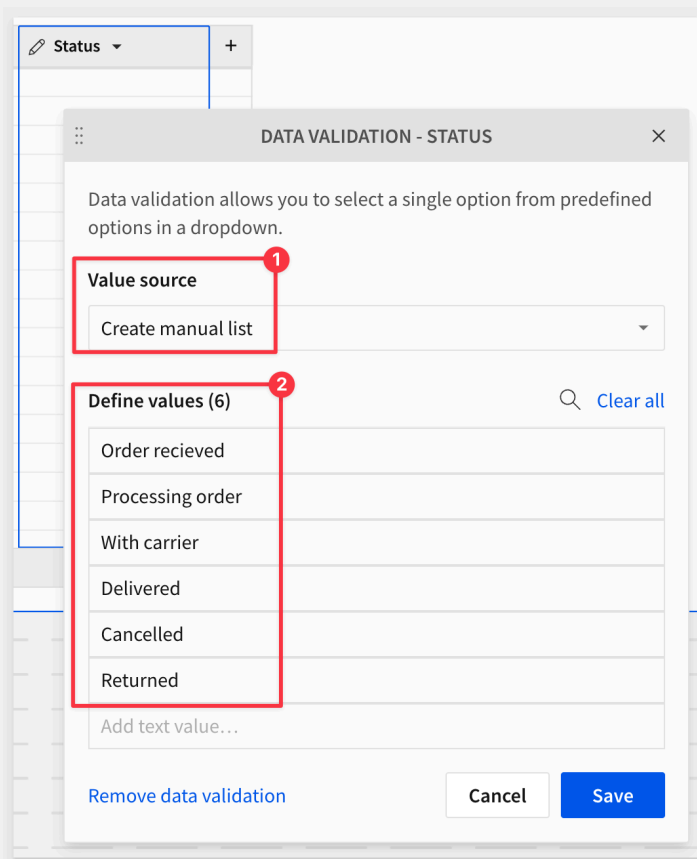
COLUMN TYPE
 Text
 Number
 Date
 Checkbox

Rename the "Text" column to "Status."

We don't want users free-form typing into the status cells, so we can add data validation. This will allow them to select from a predefined list.



We have the option to drive the validation list from another data source or create a manual list, which we did:



Now users are required to select from the list of valid choices or none.

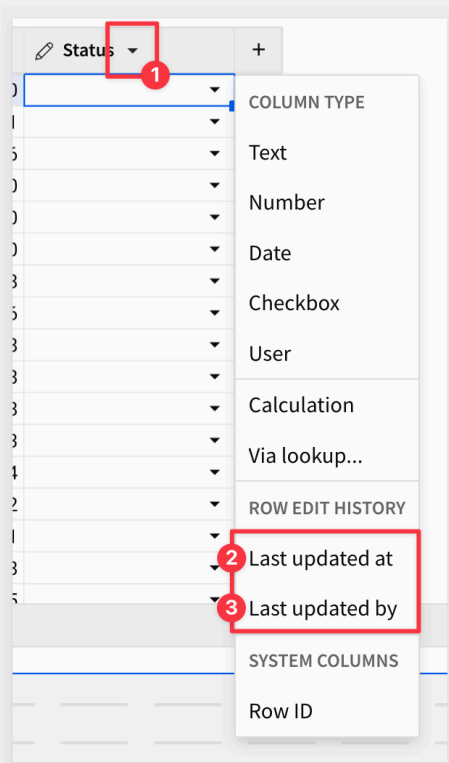
New Linked Input Table EDITABLE

	Order	Customer Number	Order Date	Part Name	Total Value	Status
1	2377734	Customer#000078581	1993-01-25	saddle navy orchid golden...	\$25,446.83	
2	2377792	Customer#000133933	1995-10-23	aquamarine antique dodg...	\$233,591.11	
3	2377792	Customer#000133933	1995-10-23	dim beige steel puff sandy	\$233,591.11	
4	2377825	Customer#000098993	1993-10-09	magenta beige lace black l...	\$263,200.57	
5	2377991	Customer#000104035	1993-08-02	peach papaya ivory chiffo...	\$285,424.38	
6	2377991	Customer#000104035	1993-08-02	cyan indian light dim green	\$285,424.38	
7	2378023	Customer#000018992	1993-08-17	linen olive drab yellow snow	\$154,333.25	
8	2378023	Customer#000018992	1993-08-17	orange metallic orchid dar...	\$154,333.25	
9	2378146	Customer#000043321	1993-12-17	tan pale metallic purple pe...	\$154,681.91	
10	2378177	Customer#000032671	1994-04-12	aquamarine goldenrod pin...	\$280,596.36	
11	2378177	Customer#000032671	1994-04-12	mint chartreuse cornsilk n...	\$280,596.36	

The Status dropdown menu is open, showing the following options: Order received, Processing order, Delivered, Cancelled, Returned, and empty. A red box highlights the dropdown menu, and a red circle with the number 1 is next to the 'Order received' option.

We also want to track who made the last change, and when. That is simple too.

By opening the menu for the "Status" column (since it is last in line, or just use the + icon as shown earlier), we can add the columns for "Last updated at" and "Last updated by".



After renaming the input table to "Shipping Status Updates," we have our solution ready for the shipping department to start using.

Invoice Status PUBLISHED Explore Edit AA

Shipping Status Updates

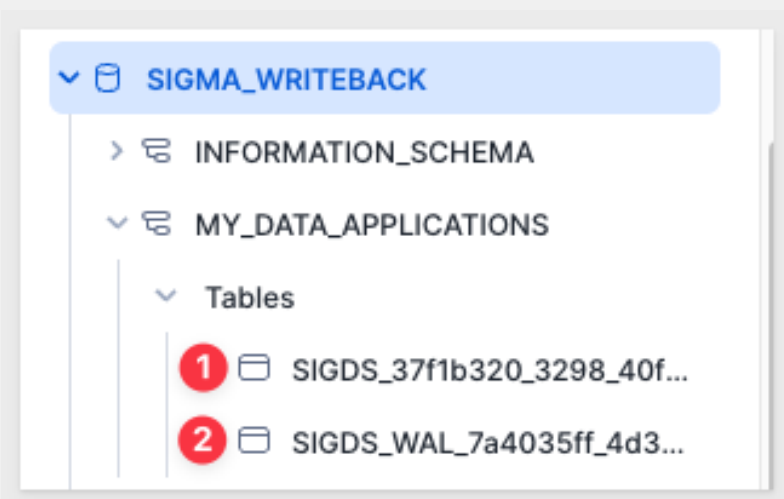
Row Nu...	Invoice Id	Name	Company	Day of Invoice Date	Quantity	Unit Price	Total Price	Status	Last updated at	Last updated by
1	1546837	Malik Porter	Pede Incorporated	2016-09-13	10	\$2.10	\$21.00	Order recieved	2024-06-27 14:58:54	phil@sigmacomputing.com
2	1556283	Gemma Wise	Sit Amet LLC	2014-12-09	14	\$1.70	\$23.80	With carrier	2024-06-27 14:59:21	phil@sigmacomputing.com
3	1579141	Tarik Bradley	Dui LLC	2018-03-16	17	\$3.03	\$51.51	Cancelled	2024-06-27 14:59:28	phil@sigmacomputing.com
4	1542370	Xantha Christensen	Parturient Montes Nascet...	2016-06-18	67	\$0.28	\$18.76	Returned	2024-06-27 14:59:30	phil@sigmacomputing.com
5	1549522	Josephine Johnson	Suspendisse Institute	2012-10-09	34	\$3.97	\$134.98			
6	1548154	Octavius Evans	Rutrum Lorem LLC	2017-04-23	23	\$2.87	\$66.01			
7	1564733	Barclay Mcdonald	Purus Gravida Limited	2017-09-24	3	\$2.08	\$6.24			
8	1556304	Marny Walton	Malesuada Id Erat Industries	2014-12-09	56	\$1.65	\$92.40			
9	1553385	Kirby West	Praesent Luctus Industries	2016-08-22	1	\$2.97	\$2.97			
10	1547721	Wesley Davidson	Luctus Industries	2016-02-15	134	\$0.39	\$52.26			
11	1544480	Brendan Campos	Habitant Morbi Tristique C...	2016-03-30	192	\$0.72	\$138.24			
12	1556415	Gemma Hull	Odio A Company	2014-12-09	140	\$5.04	\$705.60			
13	1572309	Renee Tucker	Fringilla Donec Feugiat LLC	2017-06-27	3	\$1.47	\$4.41			
14	1553220	Dustin Roberts	Et Arcu Consulting	2016-03-10	12	\$0.27	\$3.24			
15	1559905	Cathleen Patterson	In Nec Orci Corporation	2016-07-03	1	\$0.83	\$0.83			
16	1547871	Cleo Kim	Commodo Limited	2016-07-14	2	\$1.94	\$3.88			
17	1564856	Ernest Walle	Fauciat LLC	2018-01-25	9	\$3.78	\$34.02			

SUMMARY ^ 7,586,908 rows - 11 columns

Behind the scenes

We want to verify that Sigma Write-back is working.

In the Snowflake interface, after refreshing the “**Databases**” data, we can see that Sigma has indeed created two tables in the location configured in the Sigma connection we set up earlier.



The first table (#1 in the above image) hold the input table columns that are not part of the original source data (ie: the Status column) and the tracking columns, along with other reference columns.

We can use Snowflake to query table #1 and see the five status changes we made in Sigma earlier.

2024-06-28 1:13pm

No Database selected Settings

```
1 | select updated_at, updated_by, HAA4IGDM2C from
2 | SIGMA_WRITEBACK.MY_DATA_APPLICATIONS. "SIGDS_37f1b320_3298_40f6_913e_8d1a263b5d7c"
```

Results Chart

	UPDATED_AT	UPDATED_BY	HAA4IGDM2C
1	2024-06-28 10:02:23.346 -0700	phil@sigmacomputing.com	Order received
2	2024-06-28 10:02:25.367 -0700	phil@sigmacomputing.com	Processing order
3	2024-06-28 10:02:27.140 -0700	phil@sigmacomputing.com	Delivered
4	2024-06-28 10:02:29.368 -0700	phil@sigmacomputing.com	Cancelled
5	2024-06-28 10:02:31.670 -0700	phil@sigmacomputing.com	Returned

The second table contains metadata about every change made to the input table, structurally.

Since this data is stored in the customer's warehouse, it can be joined with other tables as needed, bringing back user-provided data for use in other ways too.

Inevitable scope creep

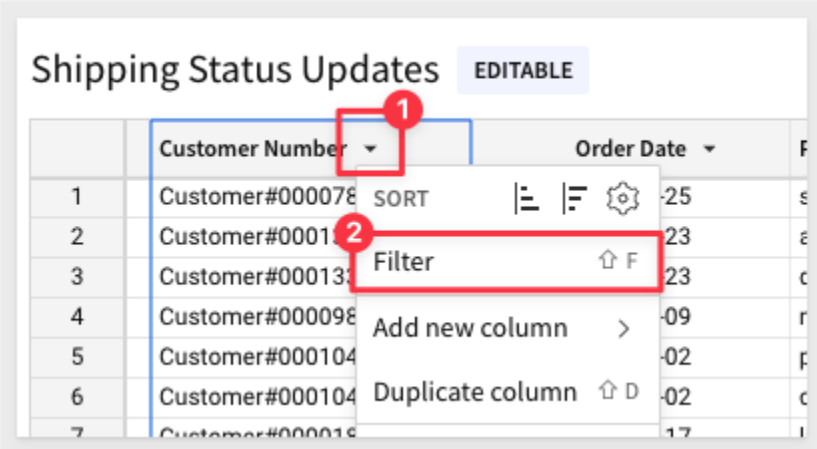
Like most software projects, the scope of requirements often changes during development or just after user acceptance testing is underway. Fortunately, with Sigma, that is not a problem at all.

The shipping department has asked for two things, and if they can get them, they will fully accept the solution.

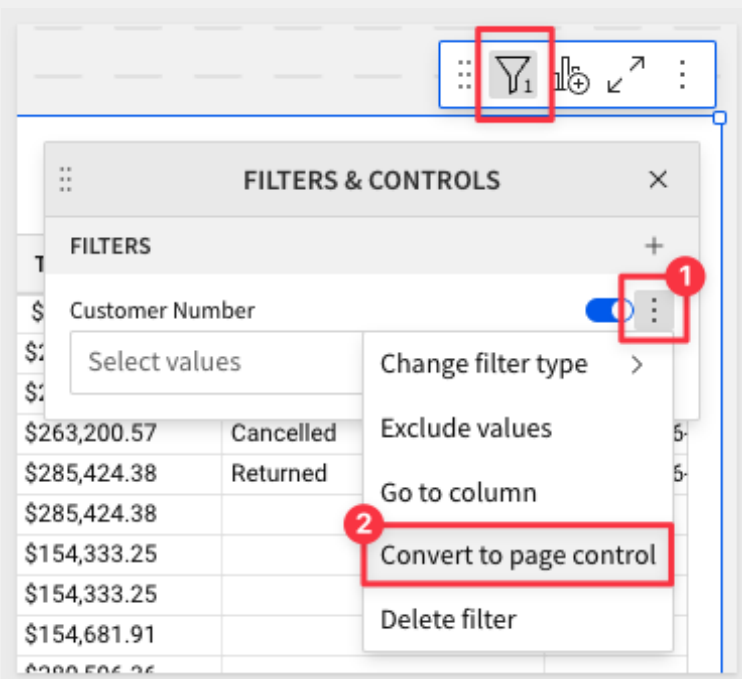
1. They need a way to easily find an order by customer name.
2. The department head wants to see how they are performing. They want to know the turnaround time from invoice date to when the line item is marked "**With carrier.**"

Filters and controls, Simplified

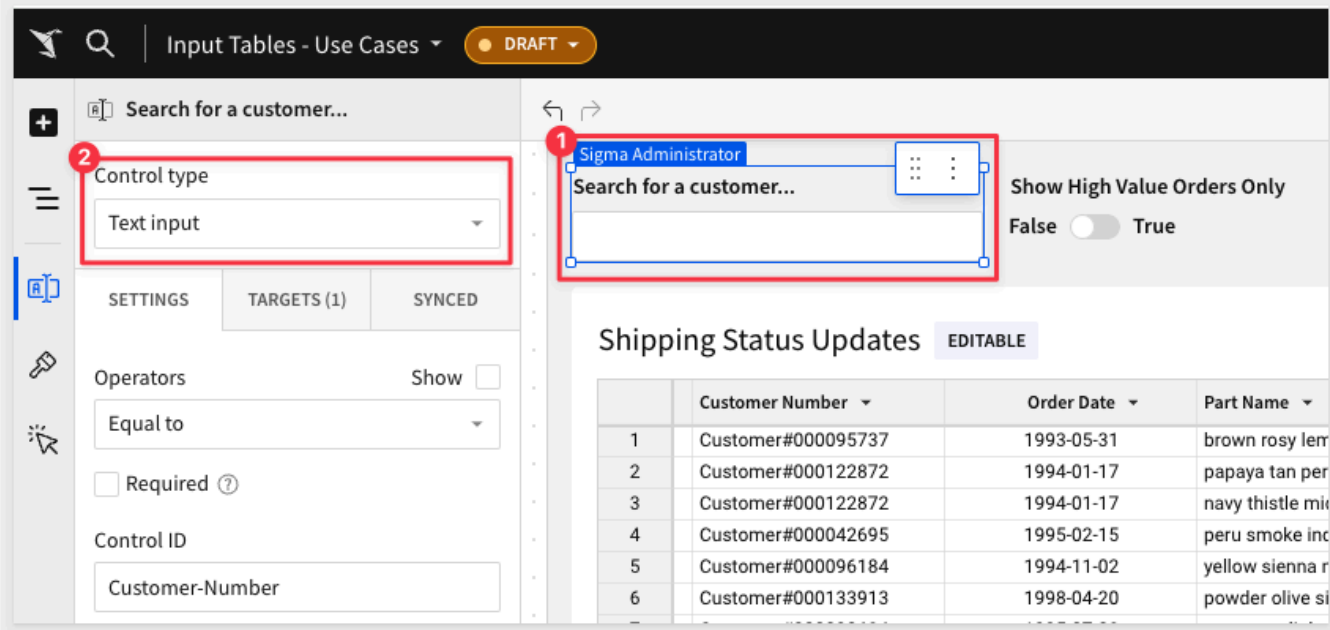
Let's tackle the first request, adding a filter control. From the "**Customer Number**" column, we select "**Filter.**"



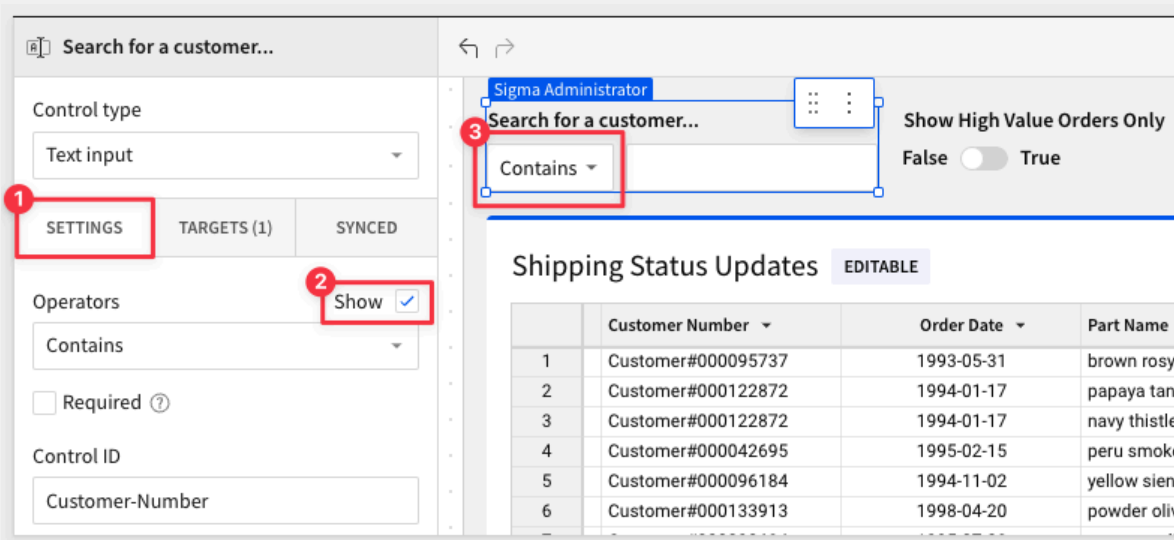
This opens the table's "FILTERS & CONTROLS" dialogue. Some users find that the filters icon on the table is not obvious enough and want something more noticeable, so we convert the new filter to a page control.



Now we have a page control, but quickly realize that a filter against 6M records is somewhat cumbersome to use, so we can change the control to a "Text Input" instead.



Let's give shipping some search power by enabling the user to select from all the available search operators.



Now, if we search for "88055," we get only 46 rows. We could refine this further with more filters, controls, and sortation, but you get just how easily Sigma handled this requirement change.

Sigma Administrator

Search for a customer... 1

Contains 88055

Show High Value Orders Only
False True

Shipping Status Updates EDITABLE

	Customer Number ▾	Order Date ▾	Part Name ▾	Total Value ▾	Status
1	Customer#000088055	1996-03-20	wheat ghost violet lime navy	\$71,876.41	
2	Customer#000088055	1996-11-27	blanched navajo seashell ...	\$234,953.70	
3	Customer#000088055	1996-02-01	wheat royal brown lace tur...	\$271,106.47	
4	Customer#000088055	1997-08-14	navy floral yellow cornsilk ...	\$204,344.55	
5	Customer#000088055	1996-03-20	rosy salmon ghost peach ...	\$71,876.41	
6	Customer#000088055	1996-11-27	olive coral forest smoke br...	\$234,953.70	
7	Customer#000088055	1993-06-09	sandy dim deep cornflowe...	\$292,539.86	
8	Customer#000088055	1995-10-03	indian magenta bisque bur...	\$77,532.94	
9	Customer#000088055	1996-02-01	pale sienna pink chiffon c...	\$271,106.47	
10	Customer#000088055	1996-02-01	hot peach magenta royal b...	\$271,106.47	
11	Customer#000088055	1993-12-12	rosy royal thistle turquoise...	\$101,075.04	
12	Customer#000088055	1996-11-27	white tan sienna deep azure	\$234,953.70	
13	Customer#000088055	1995-10-03	firebrick ivory slate cornflo...	\$77,532.94	
14	Customer#000088055	1996-02-01	midnight violet honeydew ...	\$271,106.47	
15	Customer#000088055	1997-08-14	smoke blue slate lemon p...	\$204,344.55	
16	Customer#000088055	1993-12-12	cyan coral sandy chartreu...	\$101,075.04	
17	Customer#000088055	1993-06-09	roval frosted blue pale dim	\$292,539.86	

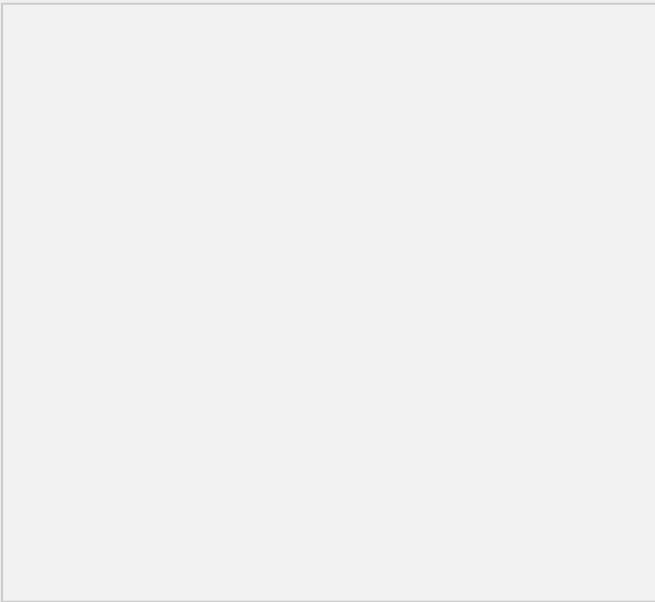
SUMMARY ^ 46 rows - 6 columns 2

However, shipping has reported that users had difficulty clearing the search control after a search.

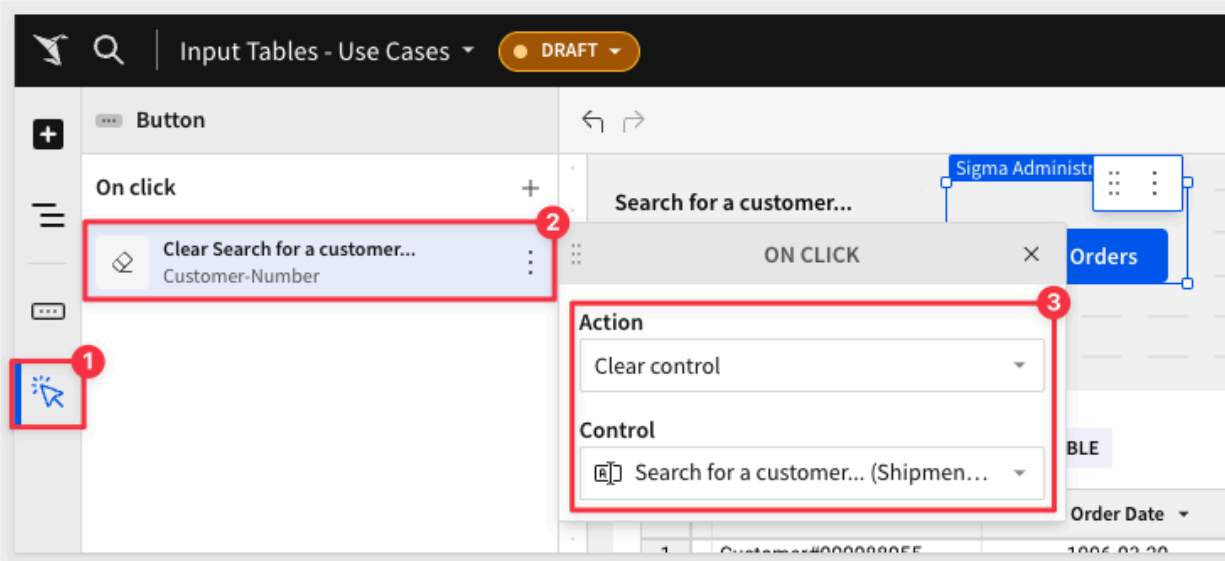
Actions

Sigma supports interactivity through a feature we call actions. In this case, we can simply add a button to the page and set its action to clear the filter control's value. Again, a simple example to make the point about how easy it is.

Add a **"UI element"** > **"Button"**.



Configure its action.



Now, when users search, they can reset the input table with a simple and obvious button click.

Complex calculations, simplified

Restating the second request from shipping: "We want to know the turnaround time from order date to when the line item is marked with carrier".

Note: It is important to understand that new columns are not written back on top of the original source data, but rather they are calculated on the fly, in the user's browser. Sigma alpha query handles this operation automatically.

However, input table columns that are not part of the source data are written back to the warehouse upon user save, via Sigma Write-back.

With that in mind, it opens up some interesting possibilities that are enabled by Write-back. Input table data that is written back to the warehouse could potentially be used by other applications too, since they also contain columns that can be joined on (e.g., Order ID, Customer Number, etc.) opening up a world of possibilities.

Now, you may have noticed that the **"Order Date"** column's dates are pretty old; **that is okay**. We will use that to make our point about complex calculations.

Add a new calculation column and name it **"Turnaround Time"**.

Set its formula to:

```
If([Status] = "Delivered", If(IsNull([Last updated at]), "",  
Concat(Text(DateDiff("year", [Order Date], [Last updated at])), " years, ",  
Text(DateDiff("month", [Order Date], [Last updated at]) % 12), " months, ",  
Text(DateDiff("day", [Order Date], [Last updated at]) % 30), " days")), "")
```

For those not used to these types of functions, which are similar to what spreadsheet users use all the time, here is an explanation.

Here's the step-by-step breakdown:

1: Check if the "Status" column is "Delivered":

- IF([Status] = "Delivered", ... , "")
- If true, it proceeds to the next condition.
- if false, it returns an empty string.

2: Check if "Last updated at" is null:

- IF(IsNull([Last updated at]), "", ...)
- If true, it returns an empty string.
- If false, it proceeds to calculate the date difference and concatenate the result.

3: Calculate the date differences and concatenate the results:

- Concat(Text(DateDiff("year", [Order Date], [Last updated at])), " years, ", Text(DateDiff("month", [Order Date], [Last updated at]) % 12), " months, ", Text(DateDiff("day", [Order Date], [Last updated at]) % 30), " days")

The results are now easily readable by users.

The screenshot shows a Sigma table interface. At the top, a formula editor displays the following formula:

```
fx If([Status] = "Delivered", If(IsNull([Last updated at]), "", Concat(Text(DateDiff("year", [Order Date], [Last updated at])), " years, ", Text(DateDiff("month", [Order Date], [Last updated at]) % 12), " months, ", Text(DateDiff("day", [Order Date], [Last updated at]) % 30), " days")), ""))
```

Below the formula editor, there are controls for 'All Orders', a 'False' toggle, and a 'True' toggle. A table of data is displayed below, with columns: Order Date, Part Name, Total Value, Status, Turnaround Time, Last updated at, and Last. A red box highlights the 'Delivered' status in the Status column and the '28 years, 5 months, 29 days' value in the Turnaround Time column for the row with Order Date 1996-01-28 and Part Name 'olive pale steel peru coral'.

Order Date	Part Name	Total Value	Status	Turnaround Time	Last updated at	Last
1993-11-07	purple beige lemon floral b...	\$166,213.29	Order received		2024-06-28 17:02:23	phil
1996-01-28	chiffon burlywood wheat b...	\$236,694.04	Processing order		2024-06-28 17:02:25	phil
1996-01-28	olive pale steel peru coral	\$236,694.04	Delivered	28 years, 5 months, 29 days	2024-06-28 17:02:27	phil
1998-03-27	white khaki mint goldenro...	\$265,058.12	Cancelled		2024-06-28 17:02:29	phil
1994-10-04	honeydew magenta blue c...	\$83,824.87	Returned		2024-06-28 17:02:31	phil
1992-03-04	coral forest bisque dodger...	\$104,691.35				
1992-11-13	chocolate drab cyan cornfl...	\$241,549.11				
1994-06-01	khaki snow gainsboro red ...	\$128,762.33				
1994-06-01	floral burlywood orchid dar...	\$128,762.33				

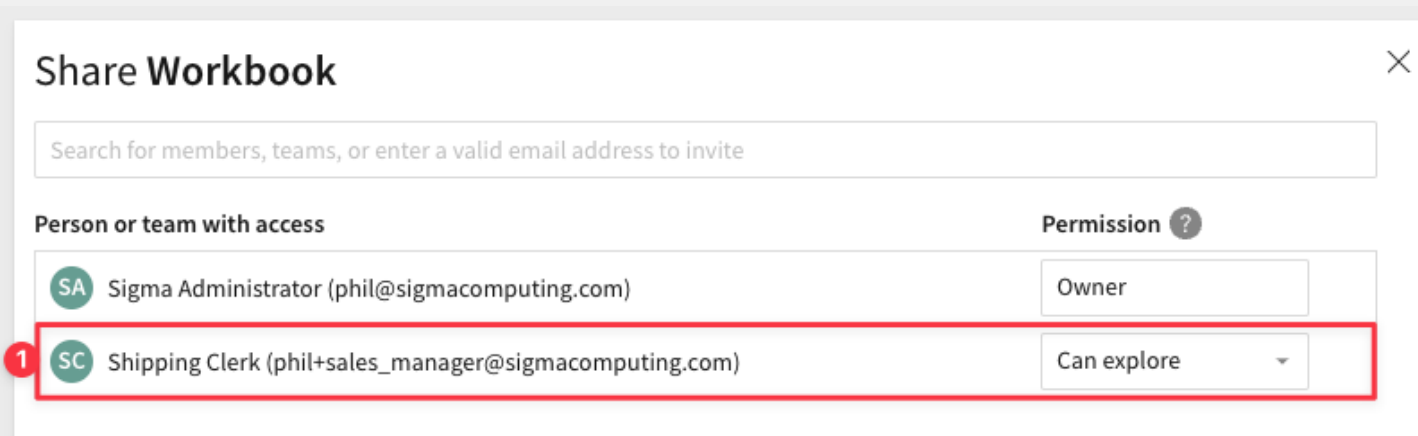
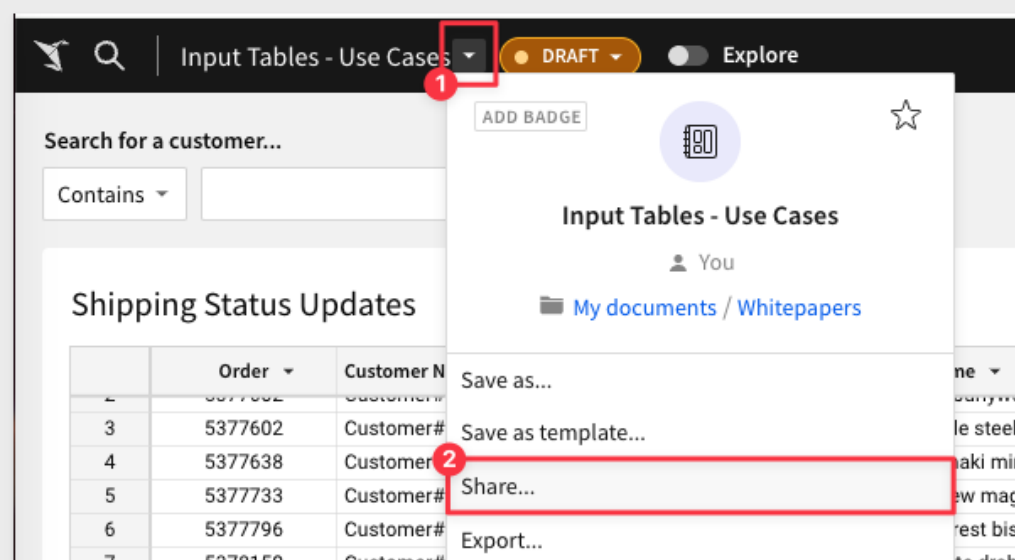
Sigma supports a wide range of functions. There is a [list of the popular](#) ones along with details on all the available functions as well.

With respect to corporate governance, Sigma also supports centralizing calculations into reusable objects that can be easily added to workbooks. This ensures that calculations are standardized and consistently applied. Sigma calls these metrics; [more reading here](#).

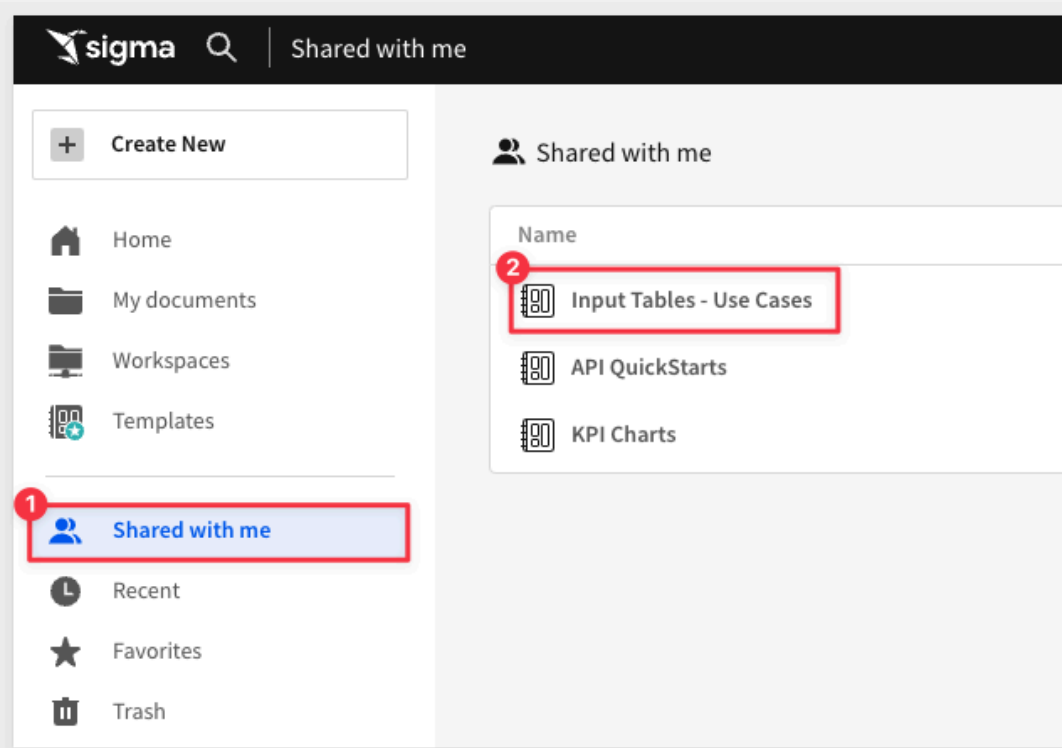
Clean up and sharing

To wrap up the new data application, we added a few more page controls, a title, dynamic text to show pending shipments and an image.

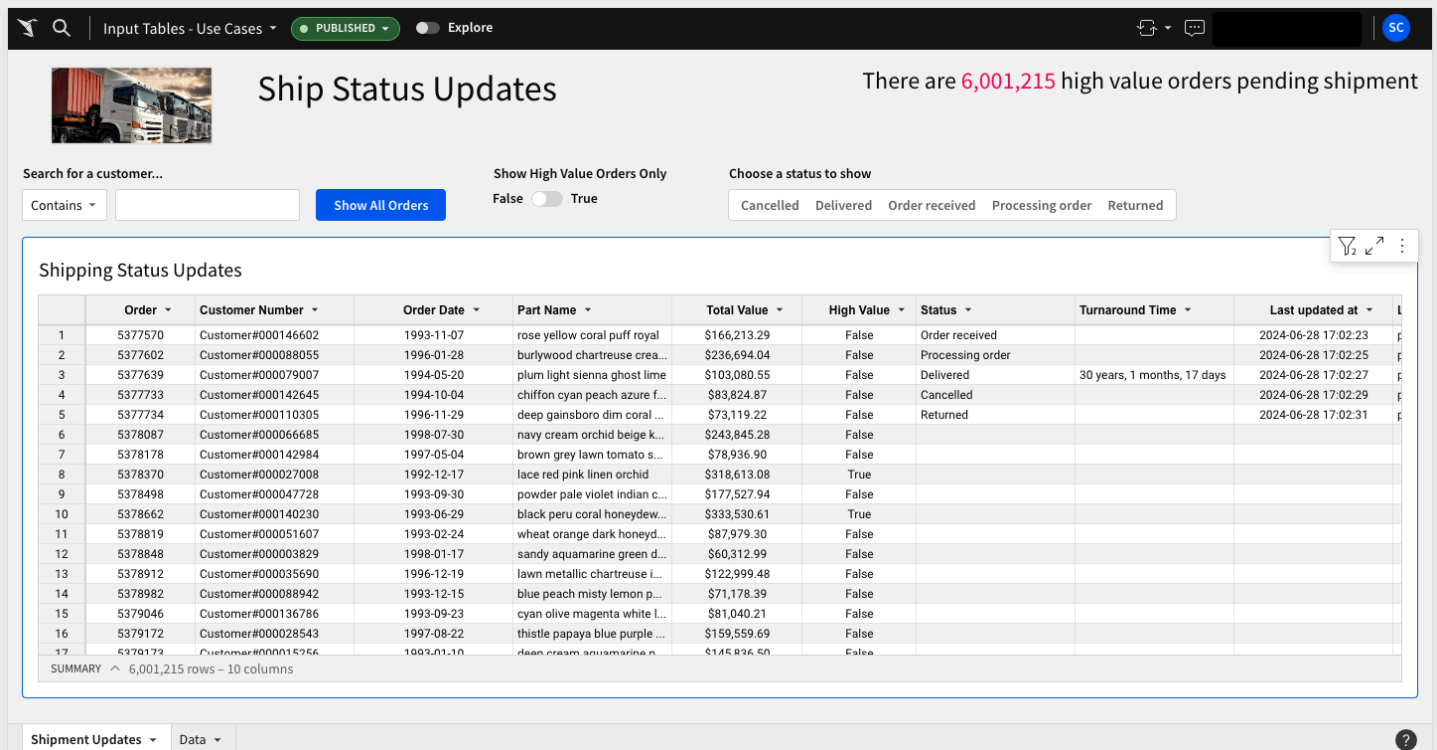
The last step is to share this with the shipping team.



After logging into Sigma as the shipping clerk, they are able to see the **"Input Tables - Use Cases"** workbook in **"Shared with Me"** and start working straight away.



Now they can see and update the final version.



Conclusion

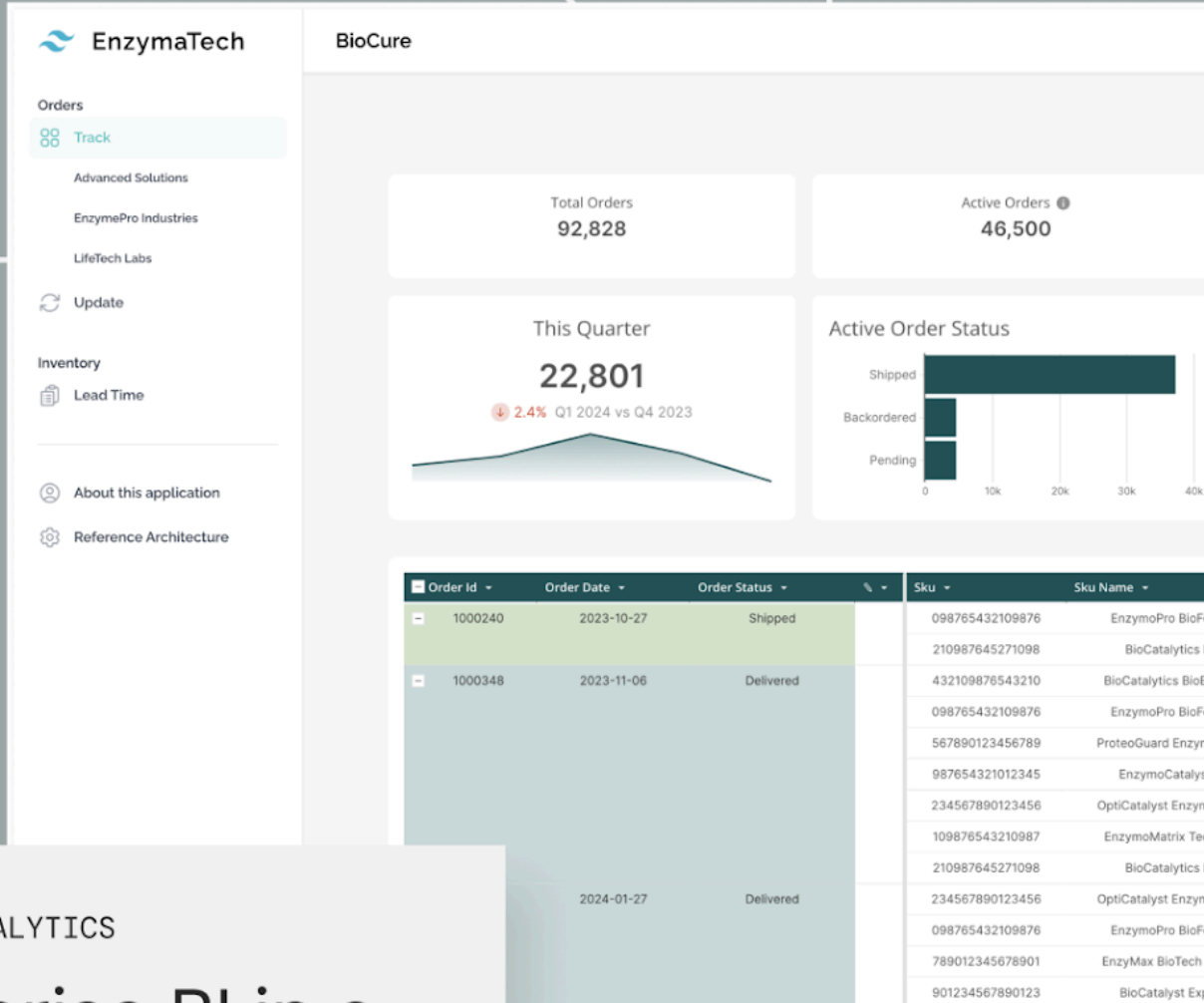
Unlock the potential of your data with Sigma. Traditional BI models offer static pages, but Sigma liberates cloud warehouse data. With Input Tables, capture insights previously locked in users' minds—no need for additional applications.

Imagine the possibilities!

Explore more Input Table use cases in our [QuickStart](#):

- Forecast Adjustment
- Rapid Data Prototyping
- Sales Territory Planning
- Embedding Input Tables
- Data Collection

TO LEARN MORE ABOUT SIGMA INPUT TABLES, [SEE HERE.](#)



NEXT GEN ANALYTICS

Enterprise BI in a Spreadsheet UI

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

LEARN MORE FROM SIGMACOMPUTING.COM