

Decode Documentation

Complete documentation for the Decode platform

Generated on: 12/2/2025

Table of Contents

[Welcome to Decode by BIMAnalytics.ai](#)
[Accessing Your Table Data](#)
[What are Metadata Tags in Decode?](#)
[The Data Page: Your Data Management Hub](#)
[Overview: From Spreadsheet to Structured Table](#)
[How Mapping Works: Connecting Spreadsheet to Table](#)
[What are Data Profiles?](#)
[Table Settings Overview](#)
[What is a Decode Table?](#)
[Understanding the Data Table View](#)
[Managing Metadata Categories](#)
[Correcting Mappings \(Simple & Formula\)](#)
[The Upload Workflow: Getting Your Files into Decode](#)
[Creating and Managing Data Profiles](#)
[Creating a New Decode Table](#)
[Ingestion Rules](#)
[Using Filters to Explore Data](#)
[Managing Values within Metadata Categories](#)
[Understanding File & Sheet Processing Status](#)
[Display Columns](#)
[Viewing Your Table Schema](#)
[Exporting Data](#)
[Export Settings](#)
[Managing Your Uploaded Files](#)
[Manually Fixing Structure Detection](#)
[Common Errors](#)
[Approving the Upload](#)
[Handling Sheets with Missing Row Identifiers](#)
[Mapping Header Data](#)
[Understanding the Preview Screen](#)
[Step 1: Adding a Data Source Node](#)
[What are Data Functions?](#)
[The Dashboard Page: Analyzing Your Data](#)
[What are Views?](#)
[Step 2: Adding a Logic Node](#)

Building a View

Step 3 (Optional): Adding a Predefined Node

Managing Data Functions

Interacting with Views

Step 4: Adding a Visualization Node

Running Data Functions

Managing Views

Step 5: Connecting Nodes

Welcome to Decode by BIMAnalytics.ai

Welcome! Decode is designed specifically for professionals in the insurance industry, particularly those dealing with the complexities of underwriting and wholesale data flows. We understand the challenges of managing vast amounts of information from bordereaux, often trapped in inconsistent spreadsheets from various wholesalers and brokers.

Decode transforms this complex, often manual process into a streamlined, powerful data management and analysis platform. Our goal is to empower you to understand your data better, make informed decisions faster, and ultimately improve your business outcomes, without needing deep technical expertise.

Who is Decode For?

Decode is built for:

- **Underwriting Firms (MGAs/MGUs):** Gain clear insights into portfolio performance, track premiums accurately, manage claims data effectively, and make data-driven underwriting decisions.
- **Wholesale Brokers:** Aggregate broker data efficiently, ensure data quality before forwarding to underwriters, and get a better handle on your book of business.
- **Insurance Startups:** Establish robust data practices from the beginning, avoiding the pitfalls of manual spreadsheet management as you scale.

If you spend significant time cleaning, verifying, and trying to make sense of insurance data arriving in spreadsheets, Decode is designed to help.

Core Concepts at a Glance

As you explore Decode, you'll encounter a few key components:

- **Tables:** Think of these as your central, organized databases. You define their structure once (including static, metadata, and dynamic columns), and Decode stores all your cleaned spreadsheet data here, adding new information over time (like monthly updates).

- **Metadata:** These are like smart labels or tags (e.g., 'Wholesaler Name', 'Contract ID') you define and can attach to your data uploads. They help you organize, filter, and add context beyond what's in the original spreadsheets.
- **Data Profiles:** Time-saving templates that pre-fill metadata tags for specific, recurring types of uploads (e.g., all uploads for a particular contract).
- **Data Functions:** Reusable analysis pipelines you build using a simple flowchart interface. They let you ask complex questions of your data (like calculating profitability per region) using AI assistance for logic creation and run the analysis repeatedly with different parameters.
- **Views:** Customizable dashboards where you can arrange and run your Data Functions side-by-side to gain comprehensive insights and compare results easily.

How to Use These Docs

This documentation is designed to help you get the most out of Decode. It's organized into logical sections accessible via the sidebar on the left:

- **The Data Page:** Details everything about setting up tables, managing metadata and data profiles, uploading and reviewing files, and viewing your clean data.
- **The Dashboard Page:** Explains how to build powerful, reusable Data Functions and arrange them into insightful Views.
- **Glossary (Coming Soon):** Defines key insurance and platform terms you might encounter.
- **Troubleshooting & FAQ (Coming Soon):** Provides answers to common questions and solutions for potential issues.

We recommend starting with the **Getting Started** guide if you're new. If you have a specific task in mind (like creating a table or building a function), navigate directly to the relevant section using the sidebar.

Navigate to these docs whenever you need help from the bottom left of the page in Decode, or look out for the help icons within the Decode platform – they link directly to the relevant documentation page!

Ready to take control of your insurance data? Let's dive in!

Accessing Your Table Data

Once you have successfully uploaded and processed data into your Decode Tables, the **Data Page** provides the primary interface for viewing and exploring this cleaned, structured information.

Selecting a Table to View

1. Navigate to the **Data Page**.
2. In the left sidebar, you will see a list under the **"Tables"** section. This list contains all the Tables you have created.
3. Click on the name of the **Table** you wish to inspect (e.g., "Premiums", "Claims").

What Happens:

- The main content area of the Data Page will update.
- The **Data Table View** for the selected Table will load, displaying the rows and columns of data currently stored in that Table.
- The **Filter Panel** (usually on the right side) will also update to show filter options relevant to the columns of the *selected* Table.

You can switch between viewing different Tables simply by clicking on their names in the sidebar list. The Data Table View and Filter Panel will refresh accordingly.

What are Metadata Tags in Decode?

In Decode, **Metadata Tags** refers to descriptive information *about* your data that helps you organize, filter, and enrich the information you upload. Think of it as adding smart labels or tags to your files and the data rows generated from them.

This metadata doesn't necessarily exist as columns within your original spreadsheets, but it's crucial context that Decode allows you to associate with your data during the upload process.

The Building Blocks: Categories, Values, and Tags

Decode's metadata system is built on three core concepts:

1. Metadata Tag Categories:

- **What they are:** Broad classifications or types of information you want to track. You define these first.
- **Examples:** `Wholesaler` , `Contract ID` , `Region` , `Underwriter` , `Data Source Type` .
- **Think of them as:** The *type* of label you want to apply.

2. Values:

- **What they are:** The specific options available within a Metadata Category. You define these *within* each Category.
- **Examples:** Within the `wholesaler` Category, Values might be `wholesaler A` , `wholesaler B` , `wholesaler C` . Within `Region` , Values might be `North America` , `Europe` , `Asia` .
- **Think of them as:** The *specific* label you can choose from.

3. Tags:

- **What they are:** The actual application of a Category and a specific Value to a data upload (and subsequently, to the rows created by that upload).
- **How they're used:** When you [upload files](#), you select a Category (e.g., `wholesaler`) and then choose a specific Value (e.g., `wholesaler A`). This creates a "Tag" (`wholesaler = wholesaler A`) that gets associated with that upload.

- **Think of them as:** The *applied* label, like `wholesaler: wholesaler A` or `Contract ID: C12345`.

Why Use Metadata?

Using metadata in Decode provides significant benefits:

- **Organization:** Easily group and separate data from different sources, contracts, regions, etc., even if that information isn't explicitly in the spreadsheets.
- **Filtering:** Powerful filtering capabilities on the Data Page and within Data Functions. You can instantly view data tagged with specific metadata (e.g., show all data for `Contract ID: C12345`).
- **Calculations:** Numeric metadata tag values can be used during upload to calculate column values. For example you may define a category `Brokerage Percentage`, then map a column like `Premium minus brokerage` as `Gross Premium x (1 - Brokerage Percentage)`. More on this later.
- **Enrichment:** Add crucial business context to your data that might be missing from the source files (e.g., tagging which underwriter is responsible for a set of policies).
- **Streamlining Uploads:** By defining [Data Profiles](#), you can automatically apply a set of default tags to uploads, saving time and ensuring consistency.
- **Semantic Clarity:** Makes your data more understandable by associating it with meaningful business labels.

Metadata Columns in Tables

When you define a [Table](#), you specify which Metadata Categories should be treated as **Metadata Columns**. When you upload a file and assign tags, the values of those tags are automatically inserted into the corresponding Metadata Columns for every row generated from that file.

This seamlessly integrates your organizational tags directly into your structured data, making filtering and analysis straightforward.

Next, learn how to set up your organizational structure by [Managing Metadata Categories](#).

The Data Page: Your Data Management Hub

The **Data Page** is the central location within Decode for managing all aspects of your insurance data. This is where you'll define how your data is structured, upload your raw spreadsheet files (like bordereaux), manage organizational tags (metadata), and ultimately view your cleaned, standardized data.

Think of the Data Page as the foundation upon which your analysis in the **Dashboard Page** is built. Getting your data setup correctly here is key to unlocking powerful insights later.

Key Functionality

The Data Page provides several core capabilities:

1. **Tables:** Define the structure (schema) for your different datasets. Tables are the containers that hold your cleaned and organized information, ready for analysis. You can create tables for premiums, claims, or any other relevant insurance data.
2. **Metadata Tags:** Create and manage custom labels (like 'Wholesaler Name', 'Region', 'Contract ID') that you can attach to your data uploads. This helps organize files and adds important context that might not be present in the original spreadsheets.
3. **Data Profiles:** Set up reusable templates with pre-defined metadata tags. This speeds up the upload process for recurring file types (e.g., monthly bordereaux from a specific wholesaler under a specific contract).
4. **File Upload & Processing:** Upload your raw spreadsheets (.xlsx, .xls). Decode's AI engine then cleans the data, intelligently maps spreadsheet columns to your defined table structure, and prepares it for review.
5. **File Management:** View the status of your uploaded files, see processing history, filter uploads by metadata tags, and manage files (including deleting them, which reverses their impact on your table data).

6. **Data Inspection:** View the final, cleaned data within your tables. Use powerful filters based on any column (including metadata and dynamic periods) to explore and understand your information.

The Typical Workflow

While you can use these features independently, a common workflow on the Data Page looks like this:

1. **Define Metadata:** Set up the categories and values you'll use for tagging (e.g., Wholesaler names, Contract IDs).
2. **Create a Table:** Define the structure for the data you want to store (e.g., a Premiums table).
3. **Create Data Profiles:** Set up profiles for common upload scenarios.
4. **Upload Files:** Upload your spreadsheets, selecting the target Table and a Data Profile (or assigning tags manually).
5. **Review & Approve:** Check Decode's automated mapping, make any corrections if needed, and approve the files for ingestion.
6. **Inspect Data:** View the cleaned data in your table using the data inspection tools.

Ready to dive deeper? Use the sidebar navigation to explore each function in detail.

Overview: From Spreadsheet to Structured Table

The core function of the **Data Page** is to get your raw insurance data, typically arriving in spreadsheets like bordereaux, into your well-defined Decode **Tables**. This section details the entire workflow, from initiating an upload to reviewing the processed data and getting it into its final, structured form.

The Goal: Automated Cleaning and Mapping

Manually cleaning spreadsheets and mapping their columns to a standard format is tedious and error-prone. Decode aims to automate as much of this as possible using AI, while still giving you the crucial final review step.

The High-Level Process

1. **Upload:** You select the target Table and provide the spreadsheet file(s), along with relevant [Metadata Tags](#) (often via [Data Profiles](#)).
2. **Extraction:** Decode reads the spreadsheet, identifies individual sheets within it, and analyzes the structure (schema) of each sheet, including header information and data columns.
3. **Automated Mapping:** Using Artificial Intelligence (AI) and a library of "possible values" it learns over time, Decode attempts to automatically match the columns found in your spreadsheet sheet to the columns defined in your target Table's schema. It also extracts header data if mappings are defined.
4. **Status Update:** Decode assigns a processing status to each sheet based on the initial mapping attempt (e.g., `Pending Review`, `Missing Row Identifier`).
5. **User Review (Crucial Step):** You are prompted to review Decode's work for each sheet. You'll see the extracted schema, the proposed mapping, and sample data. You can correct any incorrect mappings, define mappings for unmatched columns (teaching Decode for the future), and handle essential columns like the Row Identifier. You can also define formula-based mappings.
6. **Approval:** Once you are satisfied with the mapping and preview, you approve the sheet.

7. **Ingestion:** Upon approval, Decode processes the data according to the confirmed mapping and inserts or updates the relevant rows in your BigQuery Table, potentially adding new dynamic columns for new reporting periods.

Why the Review Step is Important

While Decode's AI is powerful, spreadsheet formats vary wildly. The review step ensures:

- **Accuracy:** You confirm that data is going into the correct columns in your Table.
- **Completeness:** You handle any columns the AI couldn't automatically map.
- **Control:** You have the final say before data enters your structured tables, maintaining data integrity.
- **Learning:** Your corrections help Decode improve its automatic mapping for future uploads.

This combination of automation and user oversight provides both efficiency and accuracy.

Let's start by looking at [The Upload Workflow](#) in detail.

How Mapping Works: Connecting Spreadsheet to Table

The core task during the [review process](#) is ensuring that the columns from your uploaded spreadsheet are correctly **mapped** to the columns defined in your target Decode Table. Decode uses a learning system called **Possible Values** to automate this process over time: the more you use it, the smarter it gets.

⋮info The Complete Ingestion Pipeline

When you upload a file, Decode processes it through several stages:

1. **Structure Extraction** — AI detects header rows, body rows, and metadata
2. **Schema Extraction** — Column names and hierarchy are identified
3. **Mapping** — Spreadsheet columns are mapped to Table columns using Possible Values
4. **Transformation** — Data is cleaned, formatted, and converted to match Table schema
5. **Ingestion Rules** — Your custom validation and transformation rules are applied
6. **Database Write** — Clean data is written to BigQuery

This page focuses on steps 1-4. For step 5, see [Ingestion Rules](#).

⋮

Step 0: AI Structure Extraction

Before Decode can map your spreadsheet columns to your Table, it first needs to understand the **structure** of your spreadsheet. Insurance spreadsheets (like bordereaux) often have complex layouts with:

- **Header metadata** at the top (report date, wholesaler name, contract info, etc.)
- **Column headers** that may span multiple rows (hierarchical or merged headers)
- **Data rows** containing the actual records

Decode uses **AI** to automatically detect:

1. **Where the column headers start and end** (which rows contain column names)
2. **Where the data body starts and end** (which rows contain actual data records)

3. Header metadata (key-value pairs extracted from the top section)

This AI-powered extraction works automatically for most spreadsheet formats. However, if your spreadsheet has an unusual layout that the AI misinterprets, you can [manually override the structure detection](#) to tell Decode exactly where your headers and data are located.

Once the structure is understood, Decode extracts the **schema** (the column names and their hierarchy) and proceeds to the mapping step described below.

The Goal of Mapping

Mapping tells Decode: "Take the data from *this column* in my spreadsheet (or *this piece of header metadata*) and put it into *that specific column* in my Decode Table."

For example, you might map:

- A spreadsheet column called "Gross Written Premium" → Table column `Gross Premium`
 - A header field like "Report Month: Jan 2024" → Table column `month`
-

What Are Possible Values?

Possible Values are the "memory" that allows Decode to automatically map your spreadsheets. For every column in your Table, Decode maintains a list of spreadsheet column names (and formulas) that have previously been mapped to it.

Think of it like teaching Decode your vocabulary:

- When you first create a Table, these lists are empty - Decode doesn't know what to expect.
- Each time you approve a mapping, Decode learns and adds that mapping to its memory.
- Future uploads with similar column names are then mapped automatically.

Example

Imagine you have a Table column called `Gross Premium`. Over time, different wholesalers send you spreadsheets with columns named:

- "Gross Premium"
- "Gross Written Prem"
- "GWP"
- "Premium > Gross"

After you map each of these once, all four names become **possible values** for `Gross Premium`. The next time Decode sees any of these column names, it automatically proposes the correct mapping.

Two Levels of Possible Values

Decode maintains possible values at two levels:

1. Global Possible Values (Table Level)

These are stored on the Table itself and apply to **all uploads** to that Table, regardless of which Data Profile is used. This is the master list of everything Decode has learned for each Table column.

Where to manage: Table Settings

Mapping Values

Configure possible values for automatic field mapping during file uploads

Row Identifier

Claim Number

- Claim No > Claim No
- Claim Reference / Number
- Claim Reference
- Claim Number
- CR0104 > Mandatory > Claim Reference / Number > nan
- Claim Reference/Number
- CR0104 > Mandatory > Claim Reference / Number
- Claim Reference / Number > nan
- CR0104 > Claim Reference / Number

Add mapping value

Static Columns

Policy Number

- Cert No > Cert No
- Certificate Reference
- Policy Reference
- Policy Number
- CR0029 > Mandatory > Certificate Reference > nan
- CR0029 > Mandatory > Certificate Reference
- Certificate Reference > nan
- CR0029 > Certificate Reference
- Claim No > Claim No

Add value

Insured Name

- Insured Name > Insured Name
- Insured Full Name or Company Name
- Insured Name
- CR0035 > Mandatory > Insured Full Name or Company Name > nan
- CR0035 > Mandatory > Insured Full Name or Company Name
- Insured Full Name or Company Name > nan
- CR0035 > Insured Full Name or Company Name
- Insured Name > Insured Name

Add value

Date of Loss

- Loss Date > Loss Date
- Date of Loss (From)
- Date of Loss
- CR0119 > Conditional > Date of Loss (From)
- Date of Loss (From)
- Date of Loss (From) > Either date of loss from or date of loss to
- CR0119 > Date of Loss (From)
- Loss Date > Loss Date

Add value

Loss Description

- Nature of Loss > Nature of Loss
- Loss Description
- empty
- Cause of Loss
- Description of Loss
- CR0118 > Conditional > Loss Description > Either cause of loss code or description
- CR0118 > Conditional > Loss Description
- Loss Description > Either cause of loss code or description
- CR0118 > Loss Description
- Type of Loss
- Nature of Loss > Nature of Loss

Add value

Class of Business

- Class of Business
- External > Class of Business
- Class of Business
- Class of Business as per CIP
- External > Class of Business
- External >> Class of Business
- CR0017 > Class of Business
- empty
- Mandatory > Class of Business: Section No, Risk code or description > CR0007 or CR0017 or CR0016
- Class of Business: Section No, Risk code or description

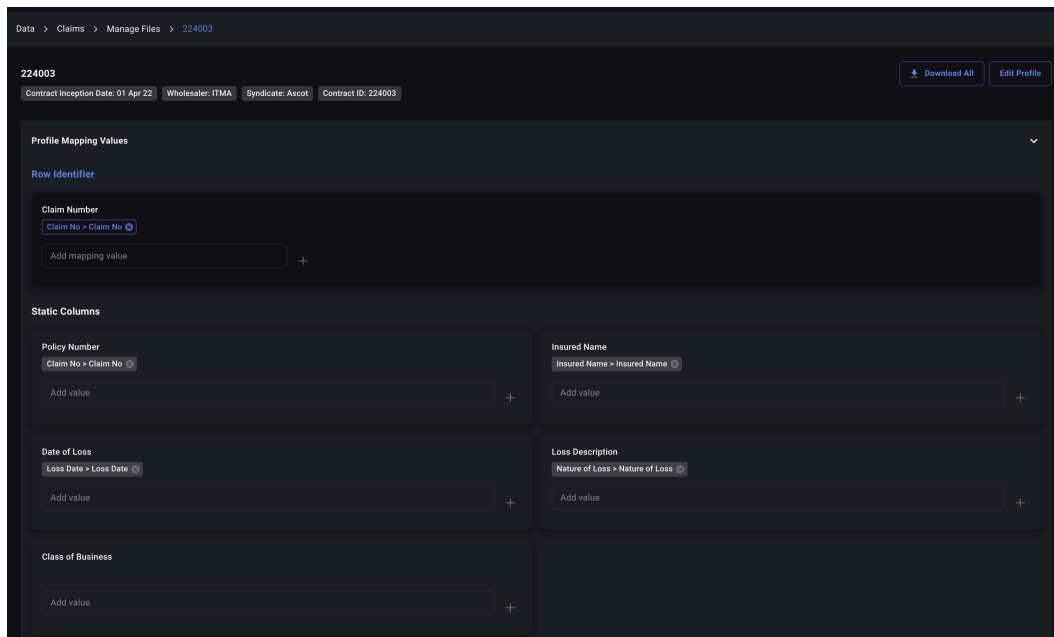
Add value

Global Possible Values

2. Data Profile Possible Values

These are stored on individual [Data Profiles](#) and are a **subset** of the global possible values. They allow you to set preferred mappings for specific data sources.

Where to manage: Data Profile Dashboard



The screenshot shows the 'Data Profile Dashboard' for contract 224003. The dashboard is divided into several sections. At the top, there is a breadcrumb trail: 'Data > Claims > Manage Files > 224003'. Below this, the contract details are displayed: 'Contract Inception Date: 01 Apr 22', 'Wholesaler: ITMA', 'Syndicate: Ascot', and 'Contract ID: 224003'. There are two buttons: 'Download All' and 'Edit Profile'. The main section is titled 'Profile Mapping Values' and contains a 'Row Identifier' section with a 'Claim Number' dropdown menu and an 'Add mapping value' button. Below this is the 'Static Columns' section, which contains five columns: 'Policy Number', 'Insured Name', 'Date of Loss', 'Loss Description', and 'Class of Business'. Each column has a dropdown menu and an 'Add value' button.

Data Profile Possible Values

Why Two Levels?

Different wholesalers or data sources often use different column naming conventions. Data Profile possible values let you tell Decode: "When uploading files through *this* Data Profile, prefer *this specific* mapping."

Example: Your Table column `Net Premium` could be mapped from:

- "Net Premium" (a direct column in some spreadsheets)
- A formula like `Gross Premium - Fees` (for spreadsheets that don't have a Net Premium column)

If Wholesaler A always provides "Net Premium" directly, you can add that to Wholesaler A's Data Profile possible values. Even though the formula exists in global possible values, the Data Profile's direct mapping will be preferred.

How Decode Proposes Mappings

When you upload a file, Decode follows this process for each Table column:

Step 1: Extract the Spreadsheet Schema

Decode analyzes your spreadsheet to identify:

- **Column headers** (including nested/hierarchical headers)
- **Header metadata** (key-value pairs from the top section of the spreadsheet, like "Report Month: Jan 2024")

Step 2: Search for Matches

For each Table column, Decode searches its possible values in this priority order:

Priority	Source	Type	Description
1st	Data Profile	Direct (1-to-1)	A direct column match from the Data Profile's possible values
2nd	Data Profile	Formula	A formula mapping from the Data Profile's possible values
3rd	Global	Direct (1-to-1)	A direct column match from the Table's global possible values
4th	Global	Formula	A formula mapping from the Table's global possible values

Decode stops as soon as it finds a match at any level.

Step 3: Fuzzy Matching

Decode uses **AI** when comparing possible values to your spreadsheet columns. This means small spelling differences or data entry errors won't break the mapping. For example, if "Gross Written Premium" is in your possible values, Decode will also match "Gross Writen Premium" (note the typo).

Step 4: Propose the Mapping

The results appear in the [Preview Screen](#):

- **Mapped columns:** Show the proposed spreadsheet column (or formula)
 - **Unmapped columns:** Are left blank for you to fill in manually
-

How Possible Values Are Updated

Possible values are updated automatically when you approve a mapping:

1. **New mappings** (column names Decode hasn't seen before) are added to:
 - The Table's global possible values
 - The Data Profile's possible values (if a Data Profile was selected)
2. **Existing global mappings** that you accept are also added to the Data Profile's possible values. This ensures the Data Profile "remembers" your preferred choice.

This means:

- Global possible values grow over time as you process more spreadsheets
- Data Profile possible values become tailored to the specific formats used by that data source

Mapping to Header Metadata

Sometimes the value you need isn't in a spreadsheet column—it's in the header section at the top of the spreadsheet. For example:

```
Report Month: January 2024
Class of Business: Property
Wholesaler: ABC Insurance

| Policy Number | Insured Name | Premium | ...
|-----|-----|-----|----
| POL-001      | Acme Corp   | 5000    | ...
```

In this case, "January 2024" and "Property" are **header metadata**, not column data.

When mapping, you can select these values using the **External >>** prefix:

- `External >> Report Month` → Maps to "January 2024"
- `External >> Class of Business` → Maps to "Property"

These External mappings can also be stored as possible values, so Decode learns to extract them automatically from future uploads.

Managing Possible Values

You can view and edit possible values directly:

- **Global Possible Values:** Navigate to the Table settings to view and manage the master list for each table column.
- **Data Profile Possible Values:** Open the Data Profile dashboard to manage the subset specific to that profile.

This is useful when you want to:

- Remove outdated or incorrect mappings
 - Add mappings proactively before uploading files
 - Review what Decode has learned
-

Summary

Concept	Description
Possible Values	Lists of column names/formulas that Decode has learned to map to each Table column
Global (Table) Level	Master list applying to all uploads to a Table
Data Profile Level	Subset of global, with preferred mappings for a specific data source
Priority	Data Profile direct → Data Profile formula → Global direct → Global formula
Learning	Every approved mapping is saved for future automation
External >>	Maps header metadata (top-section key-value pairs) to Table columns

The more files you process, the more Decode learns your column naming conventions, and the less manual mapping you'll need to do.

Now that you understand how Decode proposes mappings, let's look at how to [Correct Mappings and Handle Different Mapping Types.](#)

What are Data Profiles?

Data Profiles in Decode are essentially saved templates or shortcuts designed to make your [file upload process](#) faster and more consistent.

The Purpose: Streamlining Tagging

When you upload data, you often need to apply the same set of [Metadata Tags](#) repeatedly. For example, every monthly premium bordereau you receive from "Wholesaler A" for "Contract C12345" might need to be tagged with:

- Wholesaler = Wholesaler A
- Contract ID = C12345

Instead of manually selecting these three tags every single time you upload a file fitting this description, you can create a **Data Profile** (perhaps named "Wholesaler A - Contract C12345 Premiums") that contains these **default tags**.

How They Work

- **Table-Specific:** Data Profiles are defined *per Table*. A profile created for your "Premiums" table won't be available when uploading to your "Claims" table.
- **Pre-defined Tags:** Each profile stores a specific combination of Metadata Categories and their chosen Values.
- **Upload Selection:** When you upload a file, instead of selecting individual tags, you can simply choose the relevant Data Profile from a dropdown list.
- **Automatic Tagging:** Decode automatically applies the default tags defined in the selected profile to your uploaded file and the resulting data rows.
- **Possible Values:** Each data profile maintains a set of possible values specific to that data profile. This allows you to specify preferred mapping configurations per data profile.
- **Overrides Possible:** Even when using a profile, you usually have the option to add extra tags or *change* a default tag for that specific upload if needed (though using profiles promotes consistency).

Benefits of Using Data Profiles

- **Saves Time:** Drastically reduces the number of clicks needed during file upload.
- **Ensures Consistency:** Minimizes errors from manually selecting tags each time, ensuring data is categorized uniformly.
- **Simplifies Workflow:** Makes the upload process easier, especially for users who handle many similar files regularly.
- **Semantic Grouping:** Clearly associates uploads with predefined business contexts (like specific contracts or wholesalers).

By setting up Data Profiles for your common upload scenarios (e.g., one profile per active contract/wholesaler combination), you significantly improve the efficiency and reliability of getting data into Decode.

Next, learn how to [Create and Manage Data Profiles](#).

Table Settings Overview

Every Decode Table has configurable settings that control how data is processed during ingestion, how it's displayed when you view it, and how it can be exported. These settings are accessible through the **Table Settings** panel.

Accessing Table Settings

1. Navigate to the **Data** page
2. Select a Table from the sidebar
3. Click the **Settings** tab (gear icon)

Available Settings

Table Settings are organized into several sections:

Schema

View and edit your Table's column structure. This shows the Row Identifier, Static Columns, Metadata Columns, and Dynamic Columns that define your Table.

→ Learn more about schemas in [What is a Table?](#)

Mapping Values (Possible Values)

Configure the lists of column names and formulas that Decode uses to automatically map incoming spreadsheet columns to your Table columns.

→ Learn more in [How Cleaning Works](#)

Display Columns

Control which columns are visible when viewing data in the Data tab, and customize the order in which columns appear.

→ [Display Columns Settings](#)

Ingestion Rules

Define rules that automatically validate, transform, or filter data during the upload process. Ingestion rules give you precise control over data quality.

→ [Ingestion Rules](#)

Export Settings

Create export configurations that map your Table data to predefined Excel templates. Useful for generating standardized reports.

→ [Export Settings](#)

Saving Changes

After making any changes to Table Settings, click the **Save Settings** button at the bottom of the page to persist your changes. Settings are applied immediately to all future data operations.

:::caution

Changes to ingestion rules affect all future file uploads. Existing data in your Table is not retroactively modified by new rules.

:::

What is a Decode Table?

In Decode, a **Table** is the fundamental building block for storing your cleaned, organized, and standardized insurance data. Think of it as a smart, structured spreadsheet specifically designed for the kind of information you work with, like premium bordereaux or claims details.

Unlike a simple spreadsheet, a Decode Table has a defined structure (called a **schema**) that you create. This structure ensures consistency across all the data you upload, making analysis much easier and more reliable.

Key Characteristics of a Decode Table

- **Structured Storage:** Each Table corresponds to a dedicated table in the Decode database, providing robust and scalable storage.
- **Defined Schema:** You explicitly define the columns and their types when you create a Table. This brings order to potentially messy spreadsheet data.
- **Dynamic Growth:** Tables are designed to handle data that arrives periodically (like monthly reports). They can automatically grow by adding new columns for new reporting periods (e.g., adding a 'Feb 2024' column next to 'Jan 2024').
- **Data Hub:** Tables serve as the central repository from which you can inspect data directly on the Data Page or pull data into **Data Functions** for analysis on the Dashboard Page.

Anatomy of a Table Schema

When you define a Table, you specify four main types of columns:

1. Row Identifier:

- **Purpose:** This is the *single most crucial column*. It's the unique key that identifies each individual record (like a specific policy or claim). Decode uses this to know whether incoming spreadsheet data relates to a *new* record or is an *update* to an existing one.
- **Example:** Policy Number , Claim ID .

2. Static Columns:

- **Purpose:** These columns hold information about a record that typically *doesn't change* when new periodic data arrives.
- **Example:** For a policy, this might include `Policyholder State` , `Class of Business` , `Inception Date` .

3. Metadata Columns:

- **Purpose:** These are columns you define in the Table schema but which *don't necessarily exist* in your uploaded spreadsheets. You assign values to these columns during the upload process using **Metadata Tags** and **Data Profiles**. They allow you to add crucial organizing context.
- **Example:** `Wholesaler Name` , `Contract ID` , `Underwriter Initials` , `Region` .

4. Dynamic Columns:

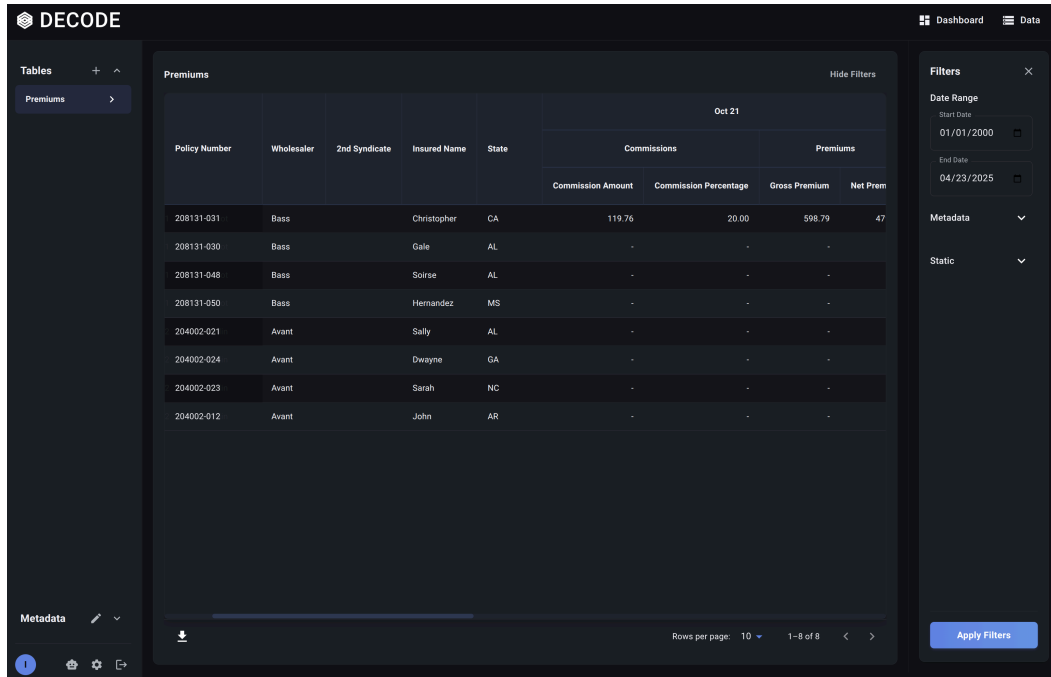
- **Purpose:** This special structure handles data that is reported periodically (e.g., monthly, quarterly). Decode automatically creates new columns over time as data for new periods is uploaded. Each dynamic column stores the relevant metrics for that specific period.
- **Example:** If you track monthly premiums, you might have dynamic columns like `2024-01` , `2024-02` , `2024-03` , etc. Each of these columns would contain details like `Gross Premium` , `Commission` , `Net Premium` *for that specific month*.

By understanding these components, you can effectively structure your Tables to accurately reflect and organize your specific insurance data needs.

Next, let's learn how to [Create a New Table](#).

Understanding the Data Table View

When you select a Table on the Data Page, the main area displays the **Data Table View**. This is a powerful grid showing your cleaned and structured data, incorporating all the column types you defined: Row Identifier, Metadata, Static, and Dynamic columns.



The screenshot shows the DECODE application interface. On the left, a sidebar contains a 'Tables' section with a 'Premiums' table selected. Below this is a 'Metadata' section. The main area displays the 'Premiums' table for the date 'Oct 21'. The table has columns for Policy Number, Wholesaler, 2nd Syndicate, Insured Name, State, Commissions (Commission Amount, Commission Percentage), and Premiums (Gross Premium, Net Prem). The right sidebar contains a 'Filters' section with a 'Date Range' filter (Start Date: 01/01/2000, End Date: 04/23/2025) and expandable sections for 'Metadata' and 'Static'. At the bottom right, there is an 'Apply Filters' button. The bottom of the table shows 'Rows per page: 10' and '1-8 of 8'.

Policy Number	Wholesaler	2nd Syndicate	Insured Name	State	Oct 21			
					Commissions		Premiums	
					Commission Amount	Commission Percentage	Gross Premium	Net Prem
208131-031	Bass		Christopher	CA	119.76	20.00	598.79	47
208131-030	Bass		Gale	AL	-	-	-	-
208131-048	Bass		Soirse	AL	-	-	-	-
208131-050	Bass		Hernandez	MS	-	-	-	-
204002-021	Avant		Sally	AL	-	-	-	-
204002-024	Avant		Dwayne	GA	-	-	-	-
204002-023	Avant		Sarah	NC	-	-	-	-
204002-012	Avant		John	AR	-	-	-	-


The data table view showing premium information

Key Features of the Data Table

- **Structured Grid:** Presents data in a familiar row-and-column format.
- **Column Headers:** Displays the names of your Table columns.
 - **Static & Metadata Columns:** Appear as standard headers.
 - **Dynamic Columns:** These are often displayed with nested headers to represent the periodic nature and the specific metrics within each period. For example, you might see a main header for 'Jan 2024' spanning sub-headers for 'Gross_Premium' and 'Commission'.

Oct 21				Jan 22			
Commissions		Premiums		Commissions		Premiums	
Commission Amount	Commission Percentage	Gross Premium	Net Premium	Commission Amount	Commission Percentage	Gross Premium	Net Premium
119.76	20.00	598.79	479.03	-	-	-	-
-	-	-	-	-	-	-	-

Hierarchically rendered multi-column headers

- **Row Identifier:** The unique key column you defined is typically shown prominently as the first column.
- **Metadata:** Values for Metadata Columns (assigned during upload via tags/profiles) are shown as columns in each row.
- **Dynamic Data:** Data from dynamic columns is laid out chronologically, showing how metrics change over time for each record.
- **Pagination:** For large tables, the data is displayed in pages (e.g., 10, 25, 50 rows per page) with controls to navigate between pages.
- **Horizontal Scrolling:** Due to the potential width of tables (especially with many dynamic periods), horizontal scrolling is often necessary to view all columns.
- **Export:** An option () is available to export the *currently filtered* view of the data to an Excel (.xlsx) file.

Interpreting Dynamic Columns in the View

Dynamic columns are represented uniquely:

- **Top-Level Period Header:** A header indicates the period (e.g., 'Jan 2024', '2024-Q1').
- **Nested Metric Headers:** Under the period header, you'll see the specific metrics defined in your dynamic column schema (e.g., 'Gross_Premium', 'Paid_Claims'). If you defined groups, you might see a middle-level header for the group name as well.
- **Cell Values:** The cell at the intersection of a specific row (e.g., Policy 123) and a specific nested dynamic column (e.g., 'Jan 2024' > 'Gross_Premium') shows the value of that metric *for that record during that period*.


This structure allows you to easily track the evolution of key metrics over time for each policy, claim, or other entity identified by the Row Identifier.

Now that you understand how the data is presented, learn how to drill down into specific subsets using the [Filter Panel](#).

Managing Metadata Categories

Metadata Categories are the high-level classifications you use to organize your data tags (e.g., `Wholesaler`, `Contract ID`, `Region`). You need to define these Categories *before* you can add specific values to them or use them for tagging uploads.

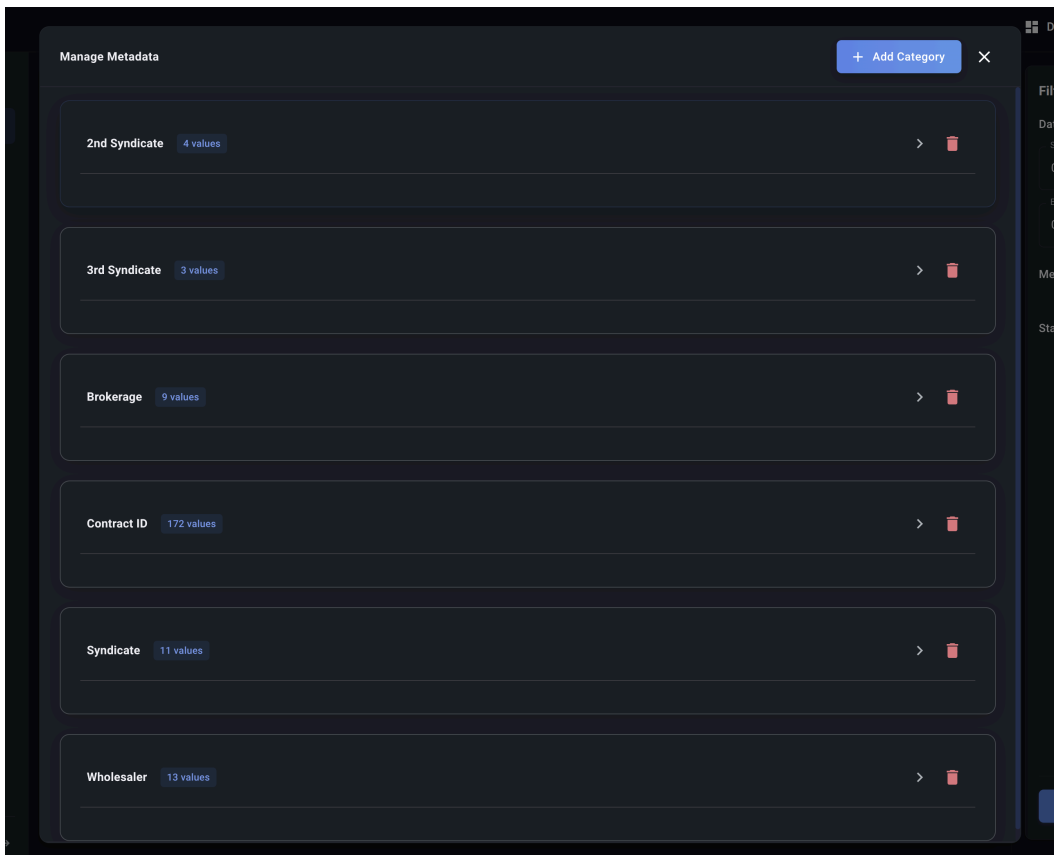
Accessing Metadata Management

1. Navigate to the **Data Page**.
2. Find the **"Metadata"** section in the left sidebar.
3. Click on the edit button indicated by an edit icon .

This will open a dedicated view showing your existing categories and allowing you to add new ones.

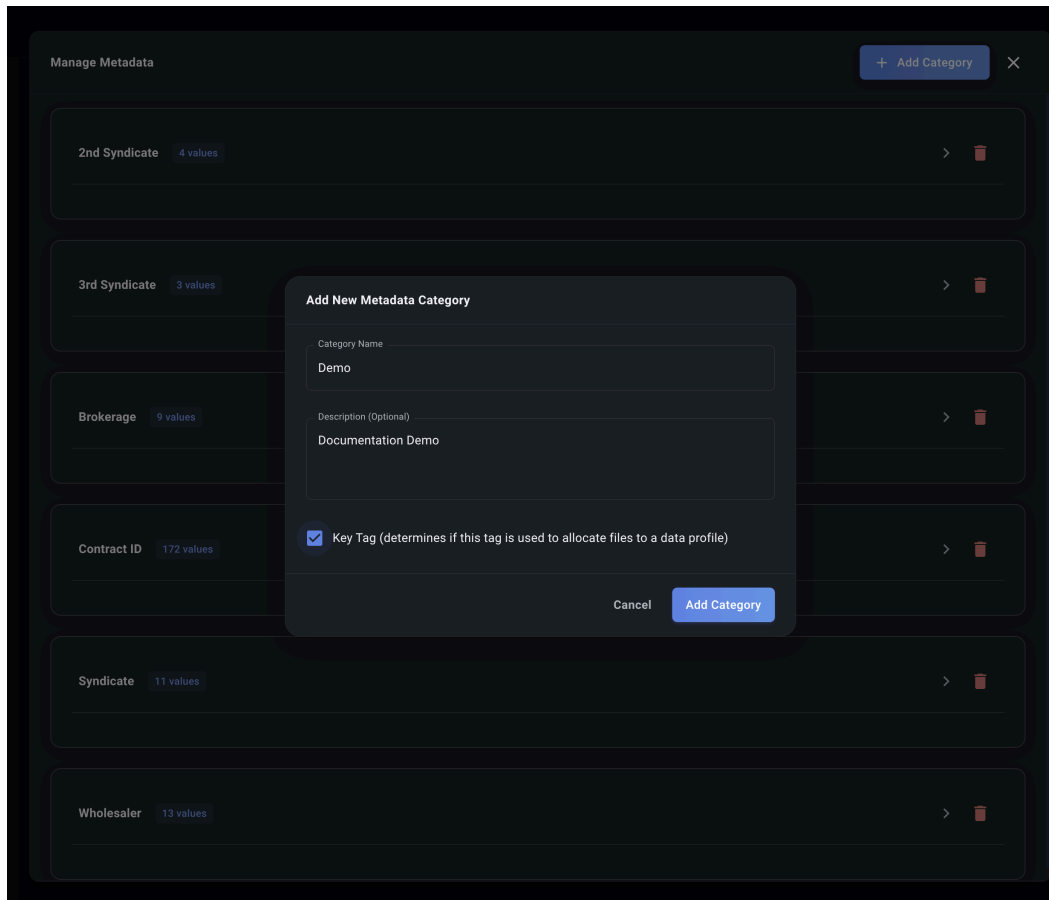
Adding a New Category

1. In the Metadata Management view, click the **"Add Category"** button.



Manage Metadata

2. A dialog box will appear. Fill in the following details:



Add Metadata Category


- * **Category Name:** Enter the name for your new category (e.g., `Syndicate`, `Line
- * **Note:** Decode will create a system ID based on this name (e.g., `Line_of_Bus
- * **Description (Optional):** Briefly explain what this category represents. This i
- * **Key Tag (Checkbox):** This is an important setting used for [Batch Uploads and

3. Click **"Add Category"**.

Your new category will now appear in the list.

Deleting a Category

Warning: Deleting a Metadata Category is a permanent action and will also remove all associated Values within it. It will *not* remove tags already applied to historical data uploads, but you won't be able to filter by that category anymore or add new tags using it.

1. Find the Category you want to delete in the Metadata Management view.
2. Click the **Delete icon** () next to the category name.
3. Confirm the deletion when prompted.

The category and all its values will be removed.

With your categories defined, the next step is to add specific options within them by [Managing Values within Categories](#).

Correcting Mappings (Simple & Formula)

The **Mapping Table** in the [Preview Screen](#) is where you ensure data from your spreadsheet flows correctly into your Decode Table columns. While Decode automates much of this, you'll sometimes need to review and make adjustments.

There are two primary ways to map a spreadsheet column or header value to a Table column: **Simple Mapping** and **Formula Mapping**.

Simple Mapping (Direct 1-to-1)

This is the most common type of mapping. You are telling Decode to take the data directly from one specific spreadsheet column (or header field) and put it into one specific Table column.

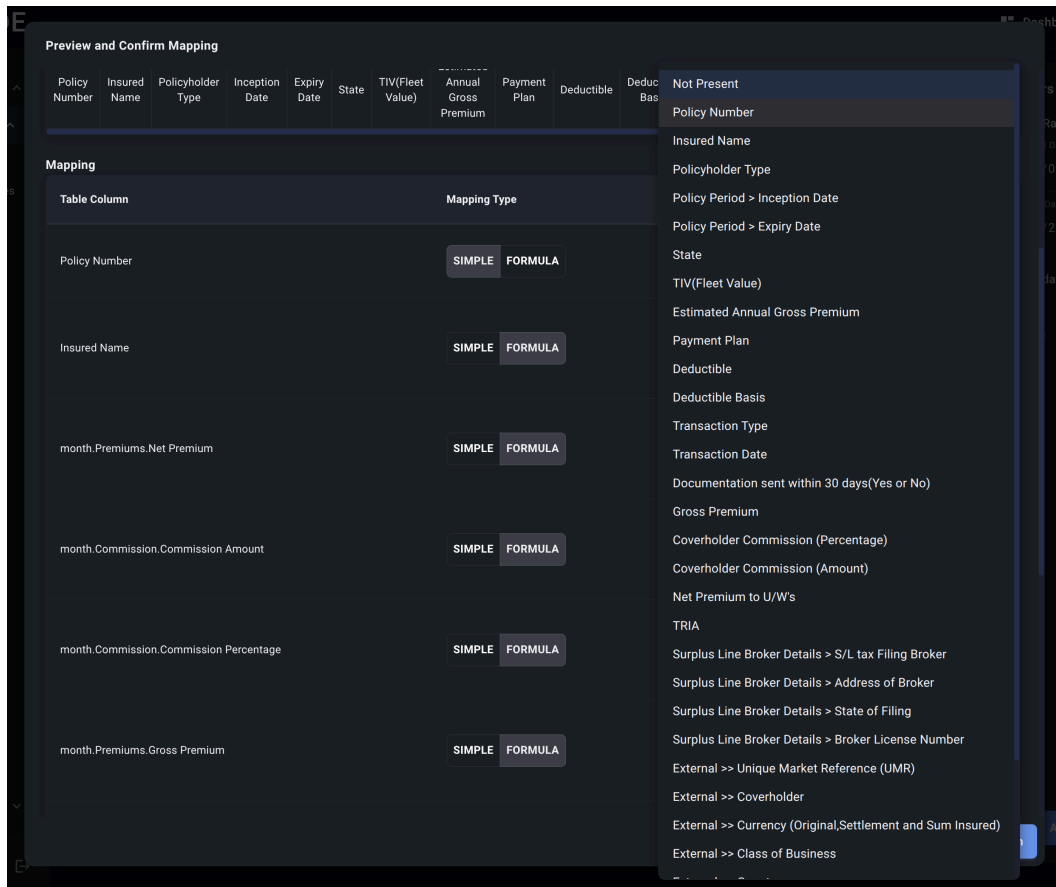
When to Use:

- When a Table column directly corresponds to a single column in your spreadsheet (e.g., Table `Policy Number` maps to spreadsheet `Policy #`).
- When correcting a mapping Decode missed or got wrong.
- When mapping header data (e.g., Table `Report Month` maps to spreadsheet header `External >> Report Month`).

How to Correct/Set a Simple Mapping:

1. Locate the **Table Column** row in the Mapping Table that needs adjustment.
2. Ensure the **Mapping Type** toggle is set to **"Simple"**.
3. Click on the dropdown list in the "Mapping" column for that row.
4. The dropdown will list:
 - All detected **column names** from your spreadsheet sheet.
 - Any detected **header fields** (prefixed with `External >>`).
 - The option `Not Present` .
5. Select the **spreadsheet column name** or **header field** that contains the correct data for this Table Column.
 - If the data for this Table Column genuinely doesn't exist in the spreadsheet, select `Not Present` . (Be careful doing this for Mandatory

columns!).

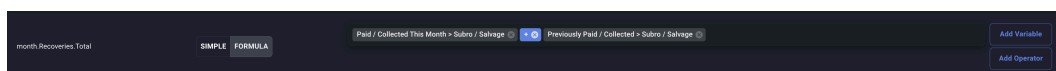


Mapping Interface

6. **Learning:** When you select a spreadsheet column and approve, Decode adds that spreadsheet column name to the library of "possible values" the ingestion AI has access to, improving future automatic mapping.

Formula Mapping (Calculated Values)

Sometimes, a column in your Decode Table doesn't exist directly in the spreadsheet but needs to be *calculated* from other spreadsheet columns or tag values. Formula Mapping allows you to define these calculations.



Formula Mapping

When to Use:

- Calculating derived metrics (e.g., $\text{Net Premium} = \text{Gross Premium} - \text{Commission}$).
- Applying simple transformations (e.g., $\text{Premium USD} = \text{Premium Local} * \text{Exchange Rate}$).

How to Set a Formula Mapping:

1. Locate the **Table Column** row in the Mapping Table where you want to define a calculation (e.g., **Net Premium**).
2. Change the **Mapping Type** toggle to **"Formula"**.
3. This will reveal the **Formula Builder** interface in the "Mapping" column.
4. **Build the Formula:** Use the Formula Builder buttons:
 - **Add Variable:** Click this, then select a **spreadsheet column name** or **header field** from the dropdown menu. This adds the selected column as a variable to your formula.
 - **Add Operator:** Click this, then select a mathematical operator (**+** , **-** , ***** , **/**) or parentheses (**(** , **)**) from the dropdown menu.
 - Build your formula step-by-step by adding variables and operators.
 - **Example:** To calculate $\text{Net Premium} = \text{Gross Premium} - \text{Commission}$:
 1. Click "Add Variable" -> Select **Gross Premium** .
 2. Click "Add Operator" -> Select **-** .
 3. Click "Add Variable" -> Select **Commission** .
 - **Delete Elements:** Click the 'x' on any chip in the formula display area to remove it.
5. **Validation:** The Formula Builder provides basic validation (e.g., checks for mismatched parentheses or consecutive operators). Address any errors shown below the builder.

Important Considerations for Formulas:

- **Data Types:** Ensure you are performing valid operations (e.g., arithmetic on numbers). The system may attempt type conversions, but incorrect formulas can lead to errors.
- **Column Availability:** You can only use spreadsheet columns or header fields that are present in the *current sheet* being reviewed.

By mastering both Simple and Formula Mapping, you gain fine-grained control over how your spreadsheet data is transformed and loaded into your structured Decode Tables.

How Correcting Mappings Updates Possible Values

Every mapping you correct or set (whether Simple or Formula) teaches Decode for the future. When you approve an upload, your mappings are saved to the [Possible Values](#) system:

1. **New mappings** (column names or formulas Decode hasn't seen before) are added to:
 - The **Table's global possible values** - so Decode can recognize them in *any* future upload to this Table
 - The **Data Profile's possible values** (if you selected a Data Profile) - so this specific mapping is preferred for future uploads using that profile
2. **Existing mappings** that you accept are also added to the Data Profile's possible values, ensuring your preferred choice is remembered for that data source.

Why this matters:

- The first time you upload from a new wholesaler, you may need to manually map several columns.
- The next time you upload from the same wholesaler (using the same Data Profile), Decode will automatically propose those same mappings.
- Over time, your Table learns the vocabulary of all your data sources, and manual mapping becomes rare.

:::tip

If a particular data source always uses different column names than your other sources, create a dedicated Data Profile for it. This keeps the mappings separate and ensures Decode proposes the right mappings automatically.

:::

Next, let's address how to handle the critical [Missing Row Identifiers](#).

The Upload Workflow: Getting Your Files into Decode

Uploading files is how you populate your Decode Tables with data from your spreadsheets (like premium or claims bordereaux). Decode provides a structured workflow to ensure data is correctly associated and tagged.

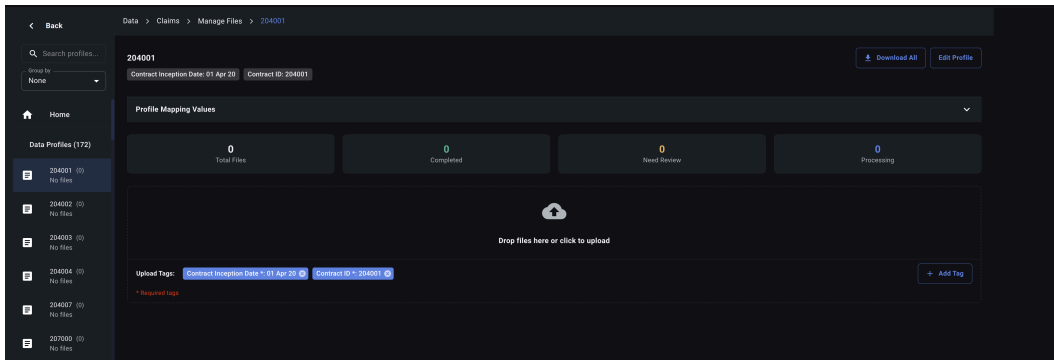
There are two main ways to upload files, accessed via the **"Upload"** action for a specific table:

1. **Direct Upload:** You select a Table, a Data Profile, and the file(s). The tags are applied immediately, and the file goes directly into processing for mapping and review. This is best for files where you know the context (e.g., which contract/wholesaler it belongs to) and all files belong to the same data profile.
2. **Batch Upload & Allocation:** You upload files without pre-assigning a Data Profile or tags. The Decode AI attempts to automatically determine the correct Data Profile based on "Key Tags" (like Contract ID) found in the file or sheet content. You then review these suggestions before the files are fully processed. This is useful for bulk uploads of sheets for different data profiles (or multiple data profiles in a single workbook) where manual tagging would be time-consuming.

Steps for Direct Upload

This is the most common method when you know the context of the file(s) you are uploading.

1. **Navigate & Select Table:** Go to the **Data Page** and select the specific **Table** you want to upload data into from the sidebar list.
2. **Select a Data Profile:** The left sidebar lists all the data profiles for the selected table. Select the data profile to which you wish to upload data.
3. **Initiate Upload:** Click the **"Upload"** button associated with the selected table. This will typically open a panel.
4. **Upload Panel:**



Direct Upload Tab

- * **Select Data Profile:** Choose a pre-defined [Data Profile](../using-data-profiles/).
- * **Select Files:** Click the file input area (or drag and drop) to select the spreadsheet files to upload.
- * **Review:** Briefly check that the correct profile/tags are selected and the desired upload options are chosen.

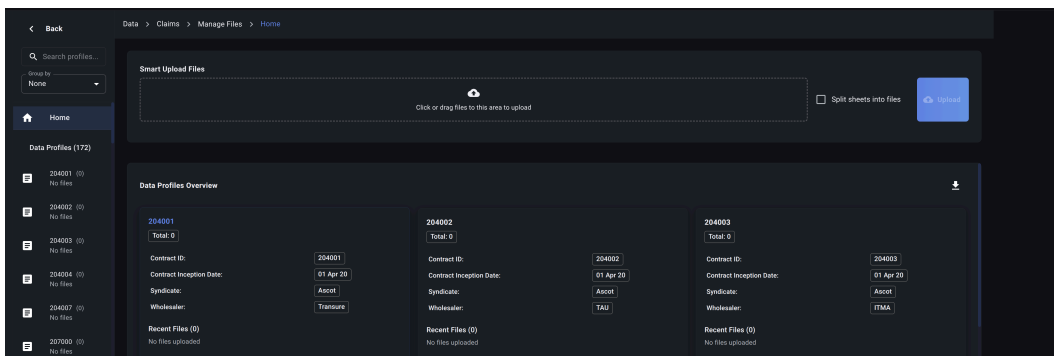
What Happens Next?

- The files are securely uploaded to Decode's storage.
- Decode begins the [processing and automated mapping](#) for each sheet within the files.
- The status of the uploaded files/sheets will update, typically moving to a state requiring your review (e.g., `PreviewReady` or `MissingRowIdentifier`). You can monitor this in the **Data Profile Dashboard** for the specific data profile you have selected.
- Proceed to [Reviewing and Approving Uploads](#) to complete the process.

Steps for Batch Upload & Allocation

Use this method when you have many files and want Decode to help assign them to the correct Data Profiles based on their content.

1. **Navigate & Select Table:** Go to the **Data Page** and select the target **Table**.
2. **Initiate Upload:** Drag and drop files into the **Smart Upload Files** section.

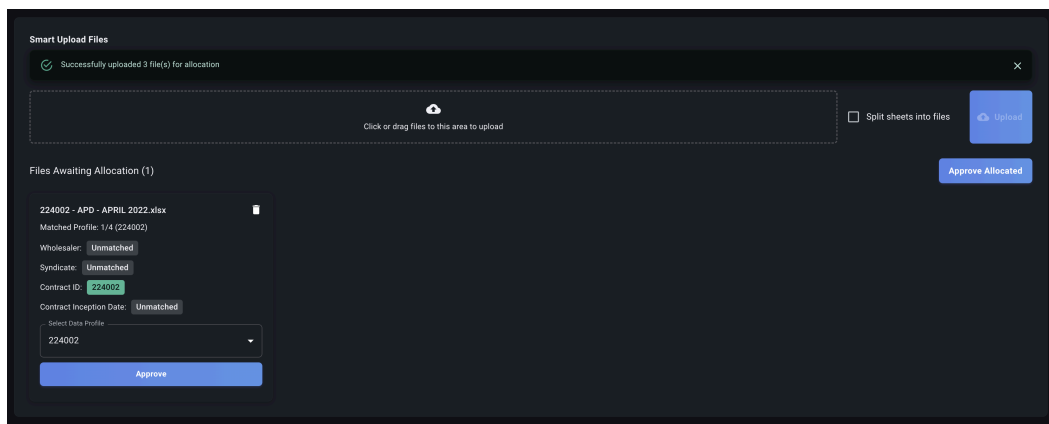


Smart Upload

3. Click **"Upload"**: Upload the files *without* assigning profiles or tags yet.

What Happens Next?

- Files are uploaded to a temporary "unallocated" area for the selected table.
- Decode uses AI analyze each file, looking for values that match **Key Tags** defined in your [Metadata Categories](#) (e.g., it might find a specific `Contract ID` in a file header or filename).
- Based on these matches, Decode **proposes** the most likely [Data Profile](#) for each file.
- You will then need to go to review the suggested data profiles for each file:
 - Review Decode's proposed profile for each file.
 - Confirm the suggestion or manually select the correct profile if needed.
 - Approve the allocation.
- Once allocated (either automatically confirmed or manually assigned), the files proceed to the standard [processing and mapping stages](#) and will require your review as usual.



Sheets Allocated to Data Profiles

Note: The Decode AI **cannot** allocate files to data profiles that do not exist - so ensure all necessary data profiles are set up before hand.

Choose the upload method that best suits your workflow and the information you have readily available when uploading your files. Next, let's understand the different [Processing Statuses](#) your files might go through.

Creating and Managing Data Profiles

Data Profiles allow you to save sets of default [Metadata Tags](#) for specific Tables, streamlining your upload process.

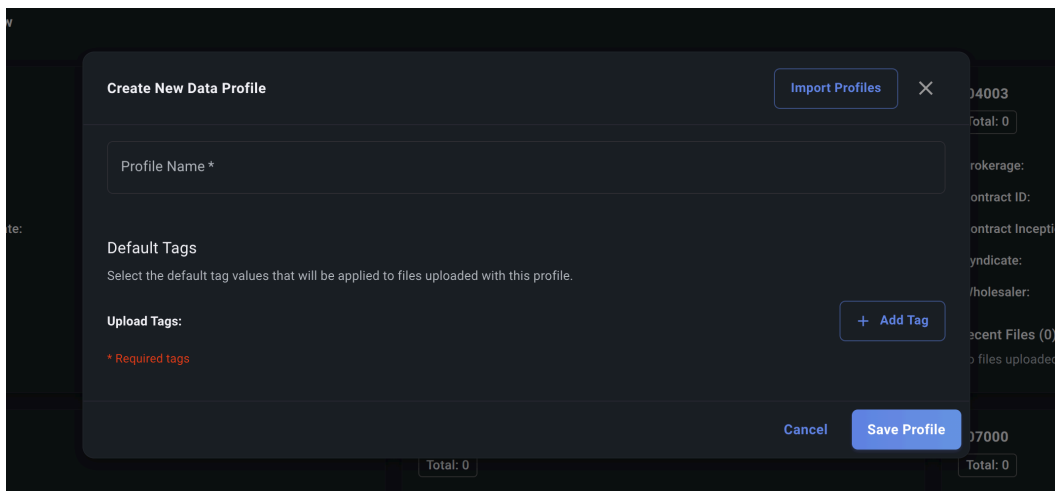
Accessing Data Profile Management

Data Profiles are managed **within the context of a specific Table**.

1. Navigate to the **Data Page**.
2. Select the **Table** you want to create or manage profiles for from the sidebar or list.
3. Click the **Data Profiles** button below the table declaration.
4. Clicking this option will open a panel showing existing profiles for that table and allowing you to create new ones.

Creating a New Data Profile

1. In the Data Profile management view for your chosen table, click the "**Create New Profile**" button.
2. A dialog box for the **Data Profile Editor** will appear.



Data Profile Panel

3. **Profile Name:**

- Enter a clear, descriptive name for this profile. This name should help you easily identify the scenario it represents (e.g., "Wholesaler A - Contract C12345 Premiums", "Broker B Claims Data", or even simply the Contract ID).
- This name is how you'll select the profile during upload.


4. Assigning Default Tags:

- This is the core of the Data Profile. You'll see an area to add tags.
- Click the **"Add Tag"** button.
- A dropdown will appear listing the available **Metadata Categories** *that have been defined as Metadata Columns for this specific Table*.
- Select the **Category** you want to include in this profile (e.g., `wholesaler`).
- Another dropdown or dialog will appear, listing the **Values** defined within that Category.
- Select the specific **Value** you want to set as the default for this profile (e.g., `wholesaler A`).
- The selected tag (e.g., `wholesaler: wholesaler A`) will be added to the profile.
- Repeat this process ("Add Tag" -> Select Category -> Select Value) for all the tags you want to include as defaults in this profile.
- Remember to include values for any **Mandatory** metadata columns defined for the table unless you intend to provide them manually during every upload using this profile.

5. **Save the Profile:** Once you've added all the desired default tags, click **"Save Profile"** or **"Create Profile"**.


Your new Data Profile is now saved and will be available for selection when uploading files to this specific Table.

Editing an Existing Data Profile

1. Access the Data Profile management view for the relevant Table.
2. Find the profile you wish to edit in the list.
3. Click the **Edit icon** () next to the profile name.
4. The Data Profile Editor dialog will open, pre-filled with the profile's current settings.

- You **cannot** change the Profile Name after creation.
 - You **can** add, remove, or change the selected Value for any default tag.
5. Make your desired changes to the default tags.
 6. Click "**Save Profile**".

Deleting a Data Profile

1. Access the Data Profile management view for the relevant Table.
2. Find the profile you wish to delete.
3. Click the **Delete icon** () next to the profile name.
4. Confirm the deletion when prompted.

Deleting a profile removes the template; it does *not* affect tags already applied to past uploads that used this profile.

With Data Profiles set up, your [file upload process](#) becomes significantly more efficient.

Creating a New Decode Table

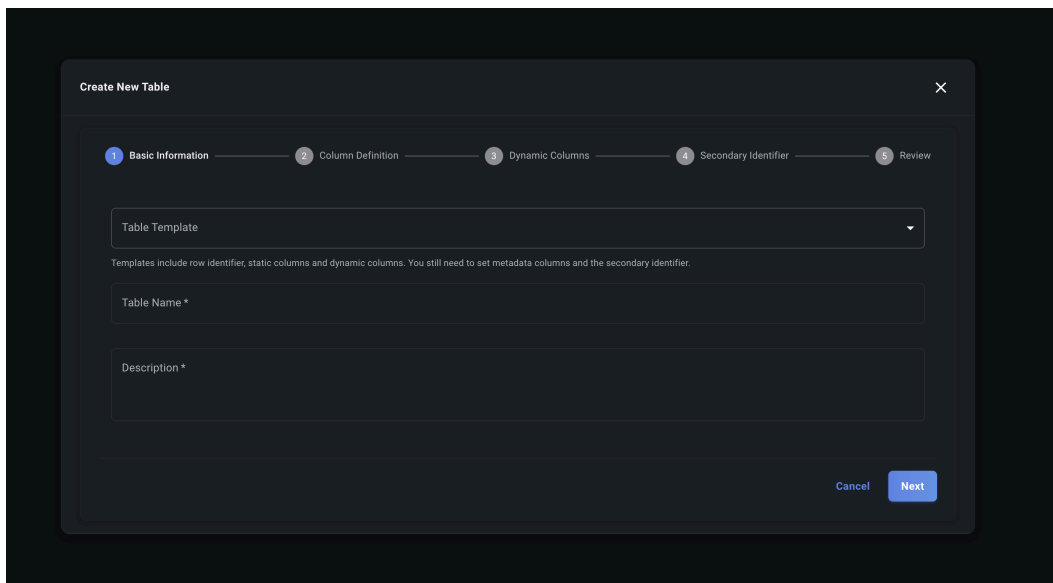
Creating a Table is the first step towards organizing and analyzing your insurance data within Decode. This process involves defining the structure (schema) that will hold your cleaned data. You'll use a multi-step form to specify the different types of columns your table will contain.

Accessing the Create Table Form

1. Navigate to the **Data Page**.
2. Locate and click the **"Create New Table"** button (displayed as a plus icon at the top of the left sidebar). This will open the multi-step table creation form.

Step 1: Basic Information

This step captures the fundamental details about your table.

The image shows a dark-themed modal window titled "Create New Table" with a close button (X) in the top right corner. At the top, there is a progress bar with five steps: 1. Basic Information (active), 2. Column Definition, 3. Dynamic Columns, 4. Secondary Identifier, and 5. Review. The main content area includes a "Table Template" dropdown menu, a text input field for "Table Name *" with a red asterisk indicating it is required, and a larger text input field for "Description *" also with a red asterisk. Below these fields is a small line of text: "Templates include row identifier, static columns and dynamic columns. You still need to set metadata columns and the secondary identifier." At the bottom right of the modal, there are two buttons: "Cancel" and "Next".

Basic Info

- **Table Name:** Enter a descriptive name for your table (e.g., "US Property Premiums", "Commercial Auto Claims"). This is the name you'll see in the Decode interface. Spaces are allowed here.
 - *Note:* Decode will automatically create a system name by converting spaces to underscores and removing special characters.

- **Description:** Provide an explanation of what data this table holds. This helps you, your team, and Decode understand its purpose later (e.g., "Monthly premium bordereaux data for US property policies underwritten via Wholesaler X"). Detail about the data and how it will be stored is important here as it provides more context for the Decode Agent and function building AI.
- **(Optional) Import Template:** This allows you to import a predefined template for your data. This template can be edited to match your needs, but provides a strong starting point.

Click **"Next"** to proceed.

Step 2: Column Definition

This is where you define the core structure: the Row Identifier, Metadata columns, and Static columns.

Create New Table

Progress: 1 Basic Information, 2 Column Definition, 3 Dynamic Columns, 4 Secondary Identifier, 5 Review

Row Identifier Column

Column Name *	Type
Policy Number	Text

Metadata Columns

Column Name *	Description	Mandatory	
Contract ID	Description	Mandatory <input checked="" type="checkbox"/>	
Wholesaler	Description	Mandatory <input checked="" type="checkbox"/>	
Syndicate	Description	Mandatory <input checked="" type="checkbox"/>	
2nd Syndicate	Description	Mandatory <input type="checkbox"/>	

[+ Add Metadata Column](#)

Static Columns

Column Name *	Description	Mandatory	
Insured Name	Description	Mandatory <input type="checkbox"/>	
State	Description	Mandatory <input type="checkbox"/>	

[+ Add Static Column](#)

Setup Columns


Row Identifier

This is the **most important column** you will define.

- **What it is:** The single column that uniquely identifies each row in your table (e.g., each distinct policy or claim).
- **Why it matters:** Decode uses this identifier to track records over time. When you upload new data, Decode checks this column to see if a row represents an *update* to an existing record or if it's a *brand new* record.
- **Fields:**
 - **Column Name:** Enter the exact name of the column in your typical spreadsheets that contains this unique identifier (e.g., `Policy Number` , `Claim ID`). *Crucially, this column MUST exist in the files you plan to upload.*

Metadata Columns

These columns allow you to add contextual information or tags during the upload process. The data for these columns *does not* need to be present in the uploaded spreadsheet itself.

- **Fields:**
 - **Column Name:** Select an existing metadata category from the dropdown.
 - **Description (Optional):** Explain what this tag represents.
 - **Mandatory:** Check this box if a value *must* be provided for this tag during every file upload to this table. Uploads will fail if a mandatory metadata tag is missing.
- **Adding/Deleting:** Use the **"Add Metadata Column"** button to add more rows and the **Delete icon** () to remove them.

Static Columns

These columns represent attributes of your records that generally don't change from one reporting period to the next.

- **Fields:**
 - **Column Name:** The name of the column as it often appears in your spreadsheets (e.g., `Insured Name` , `Policy State` , `Coverage Type`).
 - **Description (Optional):** Explain the column's content.
 - **Mandatory:** Check this box if this column *must* be present and have a value in every uploaded file destined for this table. If data is missing for a mandatory static column in any row of an uploaded file, the file processing will fail.

- **Adding/Deleting:** Use the **"Add Static Column"** button and the **Delete icon** (🗑️) as needed.

Click **"Next"** when you have defined these columns.

Step 3: Dynamic Columns

This section is for defining data that changes or is reported periodically (e.g., monthly premiums).

Dynamic Column Builder

- **Enable Dynamic Columns:** Use the toggle switch to activate this feature. If your data doesn't have a periodic component (e.g., it's just a static list), you can leave this disabled.
- **Structure Definition:** If enabled, you define the *template* for the data that will be stored for *each period*. Decode will automatically create new columns based on this template as data for new periods (like months) arrives.
 - **Adding Values/Groups:** You can define:
 - **Single Values:** Direct metrics for the period (e.g., `Gross Premium Reported`).

- **Groups:** Collections of related metrics (e.g., a `Financials` group containing `Gross Premium`, `Commission`, `Net Premium`). This helps organize complex periodic data.
- **Fields (for each value/nested value):**
 - **Name:** The name of the specific metric (e.g., `Paid Claims`).
 - **Update Behavior:** How should Decode handle this value if it already exists for a given record and period?
 - `Overwrite` : Replace the old value with the new one (default).
 - `Add` : Add the new value to the existing value (useful for cumulative numbers).
 - `Subtract` : Subtract the new value from the existing value.
 - **Mandatory:** Must this specific metric be present in the data for every period?
- **Adding/Deleting:** Use the "Add Single Value", "Add Group", and delete icons within the builder interface.

Click "**Next**" to review your schema.

Step 4: Secondary Identifier

This step is simple but important. For each table you must select a secondary identifier. This is used by Decode to prepend to the row identifier if the value for a given spreadsheet is too simple and may cause clashes with other data.

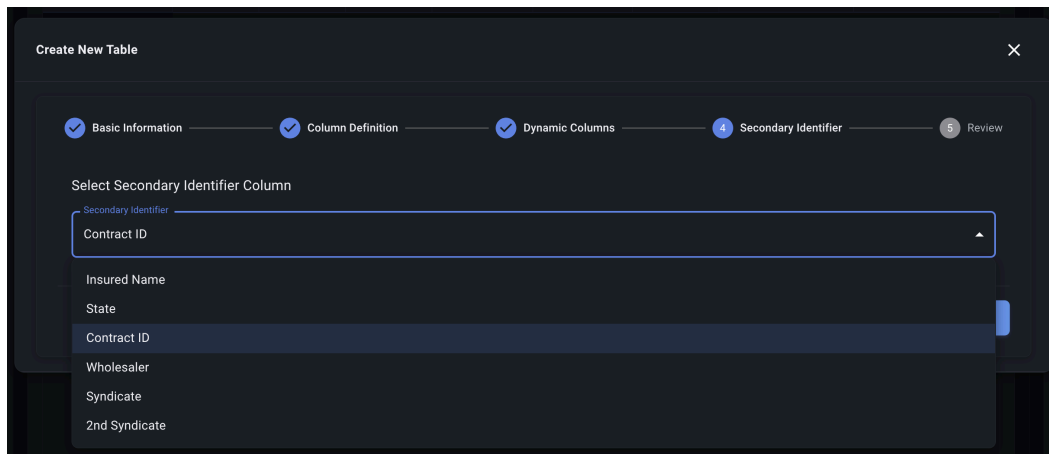
Example:

Policy Number	Contract ID	Other Data ...
01	20021
02	20021

With the secondary identifier set to **Contract ID**, Decode will ingest the data by setting the data like so:

Policy Number	Contract ID	Other Data ...
20021-01	20021
20021-02	20021

in the database. This is crucial for prevent clashes between rows that are not actually the same.



Create New Table

✓ Basic Information — ✓ Column Definition — ✓ Dynamic Columns — 4 Secondary Identifier — 5 Review

Select Secondary Identifier Column

Secondary Identifier

Contract ID

Insured Name

State

Contract ID

Wholesaler

Syndicate

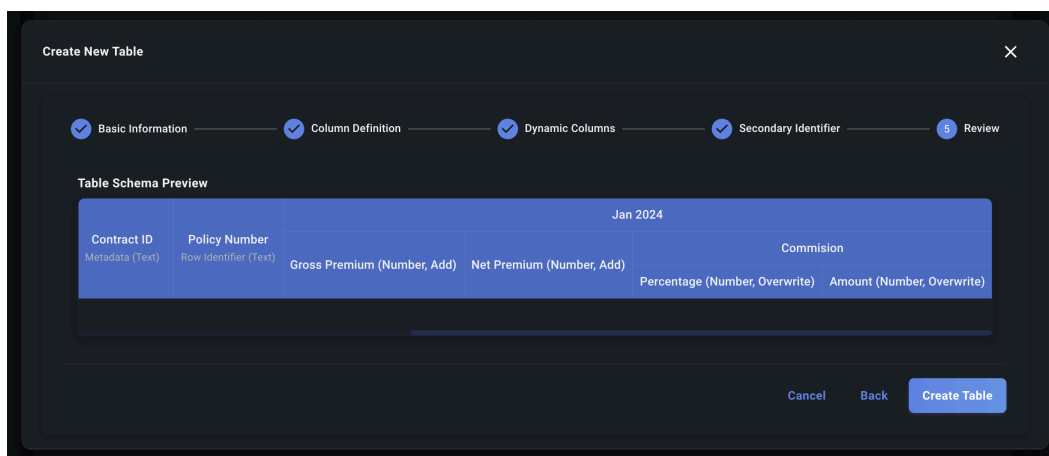
2nd Syndicate

Selecting a Secondary Identifier

Note: Any row identifier less than 5 characters long is considered too simple, and will have the secondary identifier prepended to ensure safety.

Step 5: Review

This final step displays a preview of the complete table schema you have just defined.



Create New Table

✓ Basic Information — ✓ Column Definition — ✓ Dynamic Columns — ✓ Secondary Identifier — 5 Review

Table Schema Preview

Jan 2024

Contract ID	Policy Number	Jan 2024			
Metadata (Text)	Row Identifier (Text)	Gross Premium (Number, Add)	Net Premium (Number, Add)	Commission	
				Percentage (Number, Overwrite)	Amount (Number, Overwrite)

Cancel Back Create Table

Table Preview

- **Carefully Review:** Double-check all column names, types, mandatory settings, and the dynamic column structure (if enabled). Ensure the Row Identifier is correct.
- **Go Back if Needed:** Use the "**Back**" button to navigate to previous steps and make corrections.

Creating the Table

Once you are satisfied with the schema preview:

1. Click the "**Create Table**" button.
2. Decode will now:
 - Create the corresponding table.
 - Save the table definition (schema, metadata) in our database.
 - Make the new table available on the Data Page sidebar.





Congratulations! Your table is now ready. You can proceed to [manage Metadata](#), [create Data Profiles](#), or start [uploading data](#) to it.

Ingestion Rules

Ingestion Rules allow you to automatically validate, transform, skip, or reject data during the file upload process. They give you fine-grained control over data quality without requiring manual intervention on every upload.

When Are Ingestion Rules Applied?

Ingestion rules are applied **after** the spreadsheet has been cleaned and mapped to your Table schema, but **before** the data is written to the database. This means:

1.  The spreadsheet structure has been detected (headers, body rows)
2.  Column mapping has been completed (your approvals applied)
3.  Data has been transformed to match your Table columns
4. → **Ingestion rules are applied here**
5.  Data has NOT yet been written to BigQuery

This timing is important: rules operate on the **cleaned, mapped data** using your Table's column names (like `Policy_Number` Or `month.Gross_Premium`), not the original spreadsheet column names.

Two Types of Rules

Decode supports two types of ingestion rules that address different use cases:

Row-Level Rules

Row-level rules evaluate conditions across one or more columns and then perform an action that typically affects the entire row.

Use cases:

- Skip rows where multiple columns are empty
- Set a default value when a condition is met across columns
- Validate that related fields are consistent

Cell-Level Rules

Cell-level rules target a specific column and evaluate each cell in that column individually.

Use cases:

- Clean up formatting in a specific column (remove spaces, trim whitespace)
- Transform values that meet certain criteria (prepend IDs, pad with zeros)
- Validate individual cell values

Creating Ingestion Rules

Accessing the Rules Builder

1. Navigate to **Data** → Select your Table → **Settings**
2. Expand the **Ingestion Rules** accordion
3. Click **Add Row Rule** or **Add Cell Rule**

Rule Components

Every rule has these common components:

Component	Description
Name	A descriptive name for the rule (e.g., "Skip empty premium rows")
Enabled	Toggle to activate/deactivate the rule without deleting it
Level	Row or Cell
Conditions	The criteria that must be met for the rule to trigger
Action	What happens when conditions are met

Rule Ordering

Rules are executed in the order they appear in the list. You can drag and drop rules to reorder them. This is important because:

- Earlier rules may modify data that later rules depend on
 - A "skip row" rule that runs first will prevent later rules from seeing that row
-

Row-Level Rules

Conditions

Row-level rules use conditions that evaluate column values. Multiple conditions can be combined with **AND** or **OR** logic.

Available Operators

Category	Operator	Description	Value Required
Existence	Is Empty	Cell is null or blank	No
	Is Not Empty	Cell has a value	No
Comparison	Equals	Exact match	Yes
	Not Equals	Does not match	Yes
Text	Contains	Text includes substring	Yes
	Not Contains	Text excludes substring	Yes
Numeric	Greater Than	Numeric comparison	Yes
	Less Than	Numeric comparison	Yes
	Greater or Equal	Numeric comparison	Yes
	Less or Equal	Numeric comparison	Yes
	Between	Value in range	Two values
Data Type	Is Valid Date	Can be parsed as date	No
	Is Number	Can be parsed as number	No
	Is Text	Has any value	No
	Is Boolean	True/false/yes/no/1/0	No
String	Length Equals	Character count matches	Yes

Category	Operator	Description	Value Required
	Length Greater Than	Character count exceeds	Yes
	Length Less Than	Character count below	Yes
	Starts With	Text begins with	Yes
	Ends With	Text ends with	Yes
	Matches Pattern	Regex match	Pattern

Negation

Any condition can be negated using the **NOT** checkbox. This inverts the condition's result.

Example: Combining Conditions

```
Condition 1: month.Gross_Premium IS EMPTY
Condition 2: month IS EMPTY
Logic: AND

Result: Rule triggers only when BOTH Gross_Premium and month are empty
```

Row-Level Actions

Action	Description	Options
Skip Row	Remove the row from the data being ingested	None
Fail Row	Stop processing and show an error	Custom error message
Set Value	Set a column to a specific value	Static value or copy from another column
Concatenate	Combine multiple columns into one	Source columns, separator, optional transform

Action	Description	Options
Transform	Modify a column's values	Uppercase, lowercase, trim, replace, format date, round number

Cell-Level Rules

Cell-level rules operate on a **target column** and can use either simple cell conditions or advanced conditions.

Target Column

Select which column this rule applies to. The rule will evaluate and potentially modify cells in this column.

Cell Conditions

Cell conditions evaluate individual cell values:

Condition	Description	Value Required
Always	Always triggers	No
Is Empty	Cell is null or blank	No
Content Length	Character count comparison	Operator + number
Content Type	Value type check	date/number/text
Contains	Text includes substring	Text to find
Is Contained In	Value is in allowed list	List of values
Numeric Comparison	Numeric evaluation	Operator + number

Content Length Example

Condition: Content Length < 5
Target: Policy_Number

Result: Triggers for policy numbers shorter than 5 characters

Advanced Conditions

Toggle **Use advanced conditions** to access the same powerful condition builder used in row-level rules. This allows you to trigger cell actions based on conditions in *other* columns.

Cell-Level Actions

Action	Description	Options
Skip Row	Remove the entire row	None
Fail Row	Stop with error	Custom message
Trim	Remove leading/trailing whitespace	None
Remove Spaces	Remove all spaces	None
Set Value	Replace the cell value	Static, prepend, append, arithmetic, conditional
Transform	Apply transformations	See transformation options

Transformation Options

Transform	Description	Parameters
Uppercase	Convert to uppercase	None
Lowercase	Convert to lowercase	None
Trim Whitespace	Remove leading/trailing spaces	None
Remove Spaces	Remove all spaces	None
Strip Leading Zeros	Remove leading 0s (keeps at least one)	None
Keep Digits Only	Remove all non-numeric characters	None

Transform	Description	Parameters
Remove Characters	Remove specific characters	Characters to remove
Left (first N)	Keep first N characters	Count
Right (last N)	Keep last N characters	Count
Substring	Extract portion of text	Start position, length
Find & Replace	Replace text	Find text, replace with
Pad Left	Add characters to start	Target length, pad character
Pad Right	Add characters to end	Target length, pad character
Drop Prefix	Remove text from start	Text to drop
Drop Suffix	Remove text from end	Text to drop
Cast to Number String	Convert number to clean text	None

Set Value Formulas

Cell-level "Set Value" actions support several formula types:

Prepend from Column

Adds another column's value before the current value.

```
Source Column: Contract_ID
Separator: -
Result: "ABC123-POL001" (Contract_ID + separator + Policy_Number)
```

Append from Column

Adds another column's value after the current value.

Arithmetic Operation

Perform math on numeric values:

- Multiply by
 - Divide by
 - Add
 - Subtract
-

Real-World Examples

Here are some common ingestion rule patterns:

Example 1: Remove Whitespace from Policy Numbers

Type: Cell-level

Target Column: Policy_Number

Condition: Contains (value: " " - a space)

Action: Remove Spaces

This cleans up policy numbers that accidentally contain spaces.

Example 2: Prepend Contract ID to Short Policy Numbers

Type: Cell-level

Target Column: Policy_Number

Condition: Content Length < 5

Action: Set Value → Formula → Prepend

Formula Config: Source Column = Contract_ID, Separator = "-"

When policy numbers are too short to be unique, this prepends the contract ID to create a unique identifier.

Example 3: Skip Rows with Empty Key Fields

Type: Row-level

Conditions:

- month.Gross_Premium IS EMPTY
- month IS EMPTY

Logic: AND

Action: Skip Row

This removes rows that have no premium data and no reporting month (likely blank rows in the source spreadsheet).

Example 4: Set Default Value When Field is Empty

Type: Row-level

Conditions:

- Coverholder_Commission_Percentage IS EMPTY
- OR Coverholder_Commission_Percentage EQUALS "0"
- OR Coverholder_Commission_Percentage MATCHES REGEX "^\\s0(.0+)?\\s%?\\s*\$"

Logic: OR

Action: Set Value

Target Column: month.Net_Premium_Less_Brokerage

Value Type: From Column

Source Column: month.Net_Premium

When commission is zero or empty, copy Net Premium to Net Premium Less Brokerage (since they're equivalent).



Tips and Best Practices

Order Matters

- Put "skip row" rules early to avoid processing rows you'll discard
- Put data cleaning rules (trim, remove spaces) before validation rules
- Put transformation rules in logical sequence (clean → validate → transform)

Use Descriptive Names

Name your rules clearly so you can understand their purpose at a glance:

-  "Skip rows with empty gross premium and month"
-  "Rule 1"

Test with Preview

After creating rules, upload a test file and review the preview carefully to ensure rules are working as expected before processing production data.

Disable vs Delete

If a rule isn't working correctly, disable it rather than deleting it. This lets you troubleshoot without losing your configuration.

Regular Expressions

For complex pattern matching, use the **Matches Pattern** operator with regex.

Common patterns:

- `^[A-Z]{2}\d{4}$` - Two letters followed by four digits
 - `^\s*$` - Empty or whitespace only
 - `^\d+\.\d*$` - Numeric values (with optional decimal)
-

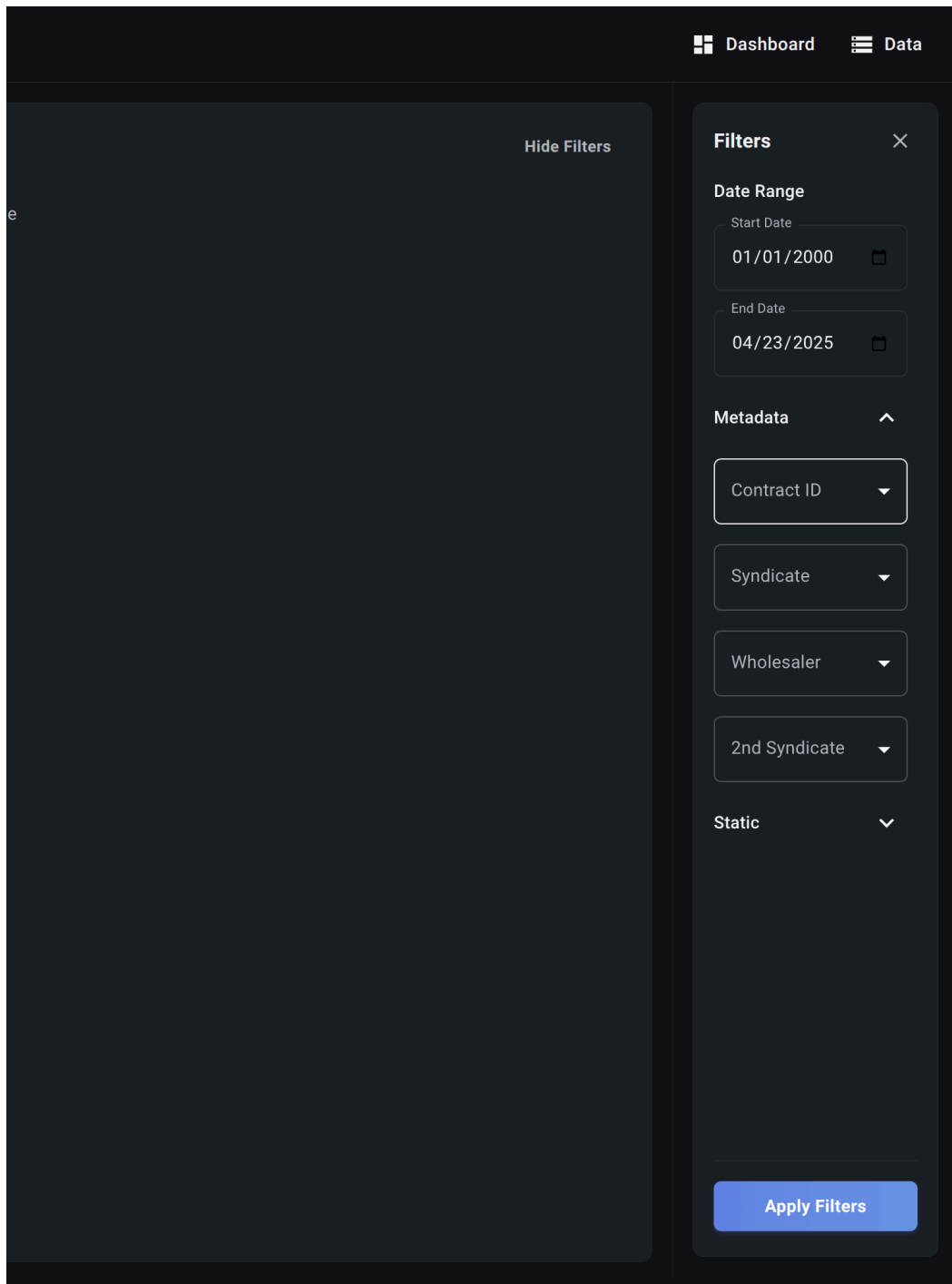
Using Filters to Explore Data

The **Data Table View** allows you to analyze the entire dataset composed of the spreadsheets you have uploaded. To analyze specific segments or answer targeted questions, you use the **Filter Panel**, typically located on the right side of the Data Page.

The Filter Panel

When you select a Table, the Filter Panel dynamically updates to show filtering options based on that Table's specific columns:

- **Date Range:**
 - **Start Date / End Date:** Allows you to filter the data based on a relevant date column.
- **Metadata Column Filters:**
 - For each **Metadata Column** defined in your Table schema, a filter control (multi-select dropdown) appears.
 - You can select one or more **Values** for a category (e.g., select 'Wholesaler A' and 'Wholesaler C' under the `wholesaler` filter).
 - The table will show rows matching *any* of the selected values within that category.
- **Static Column Filters:**
 - Filters are also provided for your **Static Columns**.
 - **Text Columns:** All static columns are converted to text, allowing you to enter text to search for matches (e.g., enter "CA" in the `state` filter).



Sidebar for applying filters.

Applying Filters

1. **Select Filters:** Choose your desired date range and select/enter values in the relevant Metadata and Static column filters.
2. **Click "Apply Filters":** Press the button at the bottom of the Filter Panel.

Result:


- The **Data Table View** will refresh, displaying only the rows that match *all* your specified filter criteria.
- If you selected multiple values within a single Metadata filter dropdown (e.g., two Wholesalers), it shows rows matching *either* of those values (OR logic within the category).
- Filters across *different* columns/categories are usually combined with AND logic (e.g., rows must match the selected Wholesaler *AND* the selected State *AND* fall within the Date Range).

The Filter Panel is a powerful tool for slicing and dicing your data directly within the Data Page, allowing you to perform quick explorations and verify subsets of your information before potentially building more complex analyses in [Data Functions](#).

Managing Values within Metadata Categories

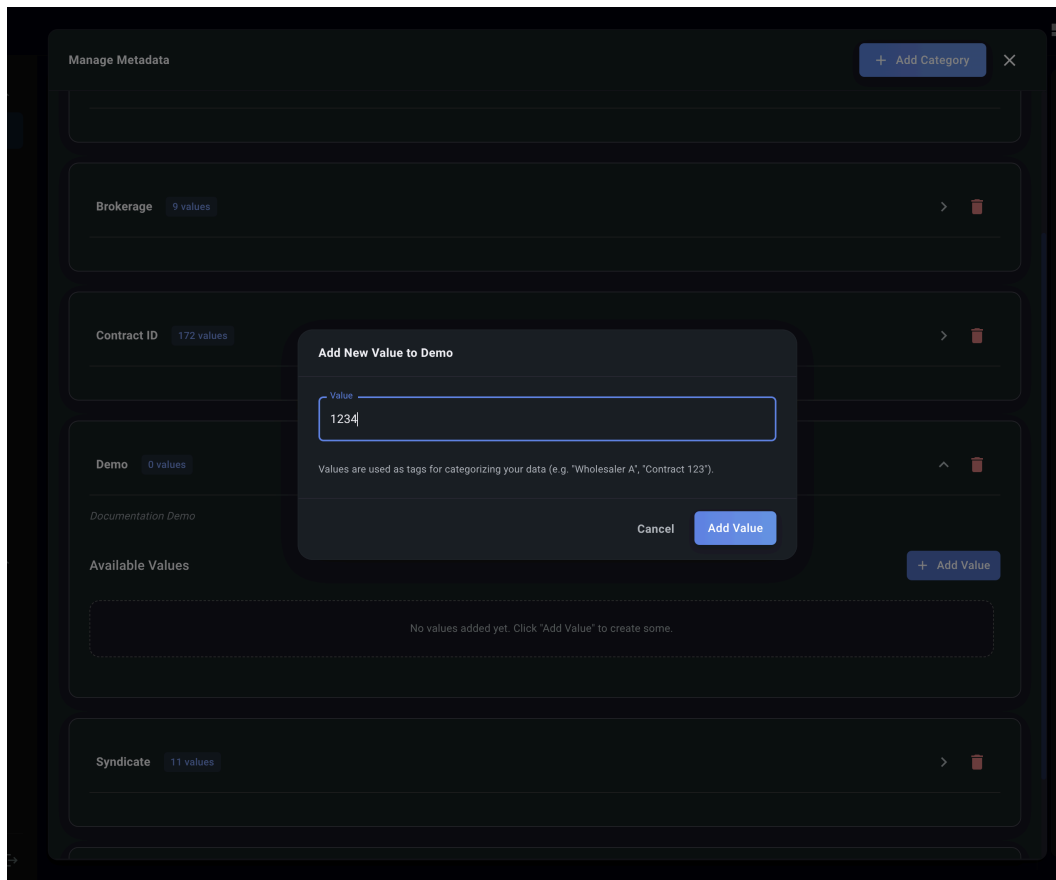
Once you have defined your **Metadata Categories** (like `wholesaler` or `Contract ID`), the next step is to add the specific **Values** that belong to each category. These values are the actual options you will select when tagging your data uploads.

Accessing Category Values

1. Navigate to the **Manage Metadata** view (by pressing the *edit* button next to the **Metadata** header in the left sidebar).
2. You will see a list of your defined Metadata Categories displayed as cards.
3. Find the Category for which you want to manage values (e.g., `wholesaler`).
4. Click the **Expand icon**  on the Category to reveal its details and value list.

Adding a New Value

1. Within the expanded Category card, click the **"Add Value"** button.
2. A dialog box will appear, prompting you to enter the new value.



Add Metadata Value

3. **Value:** Enter the specific value for this category (e.g., if the category is `wholesaler` , you might enter `wholesaler A` , `Lloyds Broker Inc` , etc.).
 - *Note:* Decode uses this value directly. Spaces and special characters are generally acceptable.
4. Click "**Add Value**".

The new value will now appear in the list of available values for that category. Repeat this process to add all necessary values (e.g., all your relevant wholesaler names, contract IDs, regions).

Deleting a Value

Warning: Deleting a value is permanent. While it won't remove tags using this value from historical uploads, you won't be able to select this specific value for future uploads or easily filter by it.

1. Expand the Category card containing the value you want to remove.
2. Find the specific value in the list.

3. Click the **Delete icon** () next to the value.
4. Confirm the deletion.

The value will be removed from the list of options for that category.

By carefully managing your Categories and their corresponding Values, you create a robust tagging system that makes organizing and filtering your data in Decode much more effective. Now you can use these to set up [Data Profiles](#) or apply them directly during [File Uploads](#).

Understanding File & Sheet Processing Status

After you upload a spreadsheet file to Decode, it goes through several processing steps. Each *sheet* within your file is processed individually, and its status reflects where it is in the journey from raw spreadsheet data to structured table data. You will see these statuses in the Data Profile Dashboard or the Smart Upload section.

Here are the common statuses you might encounter:

- **PreviewReady:**

- **Meaning:** Decode has successfully processed the sheet, extracted its schema, and generated an initial mapping proposal. The [Row Identifier](#) column was successfully found and mapped automatically.
- **Action Needed: Crucial.** You must now [Review and Approve the Upload](#). Check the proposed mapping, make any necessary corrections (for simple or formula mappings), review the sample data, and then approve the sheet for ingestion into the Table.

- **MissingRowIdentifier:**

- **Meaning:** Decode processed the sheet but **could not** automatically find or map the critical **Row Identifier** column defined in your Table schema. This is a blocking issue, as the Row Identifier is essential for tracking records.
- **Action Needed: Urgent.** You must go to the [Review and Approve Uploads](#) screen for this sheet and *manually map* the correct spreadsheet column to your Table's Row Identifier column. Once mapped and approved, processing can continue.

- **Ready:**

- **Meaning:** This status indicates that a sheet requiring review (`PreviewReady` or `MissingRowIdentifier`) has been successfully reviewed and **approved** by you, but the final data ingestion into the table might still be pending in the background queue.
- **Action Needed:** None, the system is finalizing the data insertion.

- **Pending / Processing:**

- **Meaning:** Decode has received the sheet and is actively working on it. This includes extracting the data, analyzing its structure (schema), and attempting the initial automated column mapping against your target Table schema.
- **Action Needed:** None. Wait for the process to complete.

- **Completed:**

- **Meaning:** The sheet has been fully processed, reviewed (if necessary), approved, and its data has been successfully ingested into the target BigQuery Table according to the final mapping.
- **Action Needed:** None. The data from this sheet is now part of your Table.

- **Failed:**

- **Meaning:** The processing or ingestion of the sheet encountered an error it could not recover from.
- **Common Causes:**
 - A **Mandatory** column (Static, Metadata, or Dynamic) was not mapped or was missing data in one or more rows of the spreadsheet.
 - Data type mismatch that couldn't be resolved (e.g., text in a number column after mapping).
 - Errors during formula calculation (if using Formula Mapping).
 - The AI incorrectly detected the spreadsheet structure (header/body row boundaries).
 - Internal system errors.
- **Action Needed:**
 - Check the error message associated with the sheet (usually visible in the File Management view).
 - If the schema looks wrong, try [manually fixing the header/body rows](#) and recleaning.
 - You may need to [review the mapping](#) again or potentially correct the source spreadsheet data and re-upload the file.
 - [See Common Upload Errors](#) for more troubleshooting tips.

Understanding these statuses helps you track your uploads and quickly identify sheets that require your attention for review or troubleshooting. The next critical step

is learning how to [Review and Approve Uploads](#).

Display Columns

The **Display Columns** settings control which columns are visible when viewing data in the Data tab, and in what order they appear. This helps you focus on the information that matters most for your workflow.

Why Configure Display Columns?

Tables often have many columns, but you may only need to see a subset when reviewing data:

- **Hide internal tracking columns** (like system-generated IDs)
- **Focus on key metrics** (like premium amounts and dates)
- **Organize columns logically** (group related fields together)
- **Simplify views** for specific analysis tasks

Display settings affect **only the view**—they don't delete or modify your underlying data. All columns remain available in Data Functions and exports.

Accessing Display Column Settings

1. Navigate to **Data** → Select your Table → **Settings**
 2. Expand the **Display Columns** accordion
-

Visibility Settings

Static Columns

Toggle checkboxes to show or hide each static column in your Table. Static columns contain information that doesn't change when new data is uploaded (like policyholder state or inception date).

Metadata Columns

Toggle checkboxes to show or hide metadata columns. These are columns you assign values to during upload using Data Profiles and Metadata Tags (like

Wholesaler Name or Contract ID).

Dynamic Columns

Toggle checkboxes to show or hide dynamic columns. Dynamic columns contain time-series data (like monthly premium amounts).

For nested dynamic schemas (e.g., `Premium > Gross`, `Premium > Net`), you can toggle individual leaf columns.

Additional Display Options

Propagate Most Recent Month Values

When enabled, cells that are empty for a given month will display the most recent non-empty value from earlier months.

Use case: If a policy's state is recorded in January but not re-stated in February's data, enabling this option will show the January value in the February column view.

⋮:note

This is a display-only feature. The underlying data is not modified.

⋮

Show Column Totals

When enabled, a totals row appears at the bottom of the data view showing the sum of numeric columns.

Use case: Quickly see total premiums, total claims, or other aggregate values without running a separate calculation.

Column Order

By default, columns appear in a standard order:

1. Row Identifier
2. Metadata Columns
3. Static Columns

4. Dynamic Columns (sorted alphabetically)

You can customize this order to match your workflow.

Reordering Columns

Use the **up/down arrow buttons** next to each column to move it within its category:





- **Metadata** section: Reorder metadata columns relative to each other
- **Static** section: Reorder static columns relative to each other
- **Dynamic** section: Reorder dynamic columns relative to each other (applies within each month's data)

Reset to Default

Click **Use default order** to restore the standard column ordering.

How Display Settings Are Applied

Display settings are saved to your Table and apply to:

-  The **Data** view when browsing table contents
 -  Quick data previews
 -  **NOT** Data Functions (which access all columns)
 -  **NOT** Exports (which use their own mapping configuration)
-

Saving Changes

After adjusting display settings, click **Save Settings** at the bottom of the Table Settings page. Changes take effect immediately for all views of this Table.

Viewing Your Table Schema

Once you have created a Table in Decode, you can easily view its defined structure, known as the **schema**. Understanding the schema is helpful for several reasons:

- **Verification:** Confirming the columns you expect (Row Identifier, Static, Metadata, Dynamic) are set up correctly.
- **Upload Preparation:** Knowing the exact column names and structure helps ensure your spreadsheets align with the table's expectations before uploading.
- **Troubleshooting:** If uploads fail or data looks incorrect, reviewing the schema can help identify potential mismatches.
- **Collaboration:** Sharing the schema with team members ensures everyone understands how the data is organized.

How to View the Schema

1. Navigate to the **Data Page**.
2. Find the Table you want to inspect in the list of available tables on the left sidebar.
3. Click on the **"Schema"** button below the table name.
4. A dialog or panel will open, displaying the schema details.

Understanding the Schema View

The schema view provides a clear breakdown of your table's structure:

Table Schema

✕

Row Identifier

Column Name	Type	Description
Policy Number	Text	-

Metadata Columns

Column Name	Type	Description
Contract ID	Text	-
Syndicate	Text	-
Wholesaler	Text	-
2nd Syndicate	Text	-

Static Columns

Column Name	Type	Description
Insured Name	Text	-
State	Text	-

Dynamic Columns

Example Month			
Premiums		Commissions	
Net Premium (Number, Add)	Gross Premium (Number, Add)	Commission Amount (Number, Overwrite)	Commission Percentage (Number, Overwrite)

Table Schema View

- **Row Identifier:** The single unique key column for the table is explicitly shown, along with its data type.
- **Metadata Columns:** Lists all defined metadata columns (tags) associated with this table, including their names and descriptions (if provided).
- **Static Columns:** Displays all static columns, showing their names, data types, and descriptions.
- **Dynamic Columns (if enabled):** If you configured dynamic columns, this section shows the template structure for your periodic data.
 - It will visualize the nesting (if any), displaying group names and individual value names.
 - For each dynamic value, it also shows the defined **Type** (Number, Text, etc.) and **Update Behavior** (Overwrite, Add, Subtract).

This view gives you a complete snapshot of how Decode expects data to be structured for this specific table, both for unchanging attributes and for data that evolves over time.

Export Settings

Export Settings allow you to create predefined mappings between your Table's data and standardized Excel templates. This is essential for generating reports in specific formats required by regulators, partners, or internal processes.

Why Use Export Settings?

Insurance reporting often requires data in specific formats:

- **Regulatory submissions** (Lloyd's bordereaux, state filings)
- **Partner reports** (wholesaler summaries, MGA reports)
- **Internal dashboards** (management reports, portfolio reviews)
- **Standardized templates** (industry-standard formats)

Rather than manually reformatting data for each export, you can configure mappings once and reuse them whenever you need to generate a report.

How Export Settings Work

Export Settings create a **mapping** between:

1. **Template columns** — The column structure of a predefined Excel template
2. **Table columns** — Your Decode Table's columns (row identifier, static, metadata, dynamic)

When you export data using a configured mapping, Decode takes your cleaned data and reorganizes it to match the template's structure.

Accessing Export Settings

1. Navigate to **Data** → Select your Table → **Settings**
 2. Expand the **Export Settings** accordion
-

Managing Export Mappings

Viewing Existing Mappings

The Export Settings section shows all mappings you've created for this Table:

- **Name** — Your descriptive name for the mapping
- **Template** — Which Excel template it maps to
- **Edit** — Modify the mapping
- **Delete** — Remove the mapping

Creating a New Mapping

1. Click **Add New Mapping**
2. Enter a **Name** (e.g., "Lloyd's V52 Premium Export")
3. Select a **Template** from the dropdown

Selecting a Template

Templates define the target format for your export. Decode provides a library of common insurance templates, including:

- Lloyd's bordereaux formats
- Regulatory reporting templates
- Industry-standard formats

Each template has a predefined schema (column structure) that your Table data will be mapped to.

Field Mapping

Once you select a template, the **Field Mapping** section shows all columns from the template. For each template column, you select which Table column should populate it.

Mapping Options

For each template field, you can map to:

Source Type	Description	Example
Row Identifier	Your Table's unique record key	<code>Policy_Number</code>
Metadata Column	Context added during upload	<code>Wholesaler_Name</code> , <code>Contract_ID</code>
Static Column	Fixed record attributes	<code>Policyholder_State</code> , <code>Class_of_Business</code>
Dynamic Column	Time-series data	<code>month.Gross_Premium</code> , <code>month.Net_Premium</code>
Month	The reporting period itself	<code>month</code>
(blank)	Leave the template column empty	—

Example Mapping

Template Column	Table Column
Policy Reference	<code>Policy_Number</code>
Insured Name	<code>Insured_Name</code>
Inception Date	<code>Inception_Date</code>
Premium Amount	<code>month.Gross_Premium</code>
Commission	<code>month.Commission</code>
Reporting Period	<code>month</code>

Per-Time-Period Templates

The current export system supports **per-time-period** templates. This means:

- The export generates output for each reporting period (month/quarter) in your data

- Dynamic columns are mapped to a single template column that repeats per period
- The `month` field identifies which period each row represents

This format is ideal for bordereaux-style reports where each row represents a policy-period combination.

Using Export Mappings

Once you've configured an export mapping, you can use it to generate reports from the Data view or through Data Functions.

From the Data View

1. Navigate to your Table's data
2. Apply any desired filters
3. Click **Export** and select your configured mapping
4. Download the generated Excel file

From Data Functions

Export mappings can be referenced in Data Functions for automated report generation.

Tips and Best Practices

Naming Conventions

Use clear, descriptive names that identify:

- The template format (e.g., "Lloyd's V52")
- The data type (e.g., "Premium" vs "Claims")
- Any specific configuration (e.g., "Monthly" vs "Quarterly")

Example: "Lloyd's V52 Premium - Monthly Submission"

Multiple Mappings

You can create multiple mappings for the same Table:

- Different templates for different recipients
- Variations for different time periods
- Customized formats for specific partners

Unmapped Fields

Template columns that you leave unmapped will be empty in the export. This is useful when:

- Your Table doesn't have data for certain template fields
- You'll fill in those fields manually after export
- The template includes optional fields you don't use

Validation

After creating a mapping, test it with a small export to verify:

- All required fields are mapped correctly
- Data types match expectations (dates formatted correctly, numbers not truncated)
- The output matches your recipient's requirements

Saving Changes

Click **Save Mapping** after configuring your field mappings. The mapping is saved to your Table and can be used immediately for exports.

Managing Your Uploaded Files

Once you've uploaded files to a Data Profile, Decode provides several tools to manage, inspect, and maintain them. All file management actions are accessible from the **Data Profile Dashboard** for the profile you're working with.

File Structure Overview

When you upload a spreadsheet to Decode:

- The original file is stored as a **Raw File**.
- Each worksheet within that file becomes a **Cleaned Sheet** after processing.
- Both raw files and individual sheets can be managed independently.

Files and sheets are displayed in the Profile Dashboard, grouped by their current status (Needs Attention, Processing, Completed, etc.). You can expand any raw file card to see its individual cleaned sheets.

Available Actions

Downloading Files

Decode allows you to download both your original raw files and the cleaned/processed versions.

Individual Downloads:

- **Raw File:** Click the download icon on a raw file card. If the file has cleaned sheets, Decode will bundle both the raw file and all its cleaned sheets into a ZIP file with separate `raw/` and `cleaned/` folders.
- **Cleaned Sheet:** Click the download icon on an individual cleaned sheet card to download just that processed sheet.

Bulk Downloads:

- **Download All:** Use the "Download All" button at the top of the Profile Dashboard to download every file (raw and cleaned) associated with the current Data Profile as a single ZIP archive.

- **Download Selected:** Select multiple files or sheets using the checkboxes, then click "Download" in the selection toolbar to download only the selected items.

Deleting Files

Deleting files removes them from Decode and **reverses any changes** they made to your database table (undoing the diffs).

Deleting a Cleaned Sheet:

1. Expand the raw file card to view its sheets.
2. Click the delete icon on the specific sheet you want to remove.
3. Confirm the deletion in the dialog.

The sheet's data contributions will be removed from the table.

Deleting a Raw File:

1. Click the delete icon on the raw file card.
2. Confirm the deletion in the dialog.

This will delete the raw file **and all of its cleaned sheets**, reversing all changes those sheets made to the database.

Bulk Deletion:

Select multiple files or sheets using the checkboxes, then click "Delete" in the selection toolbar to queue them all for deletion at once.

:::note

Files that are currently processing cannot be deleted. Wait for processing to complete before attempting deletion.

:::

Recleaning Files

If a sheet encountered an error during processing, or if you've updated the [Profile Mapping Values](#) to better accommodate your file's schema, you can **reclean** the sheet to re-run the extraction and mapping process from scratch.

To reclean a sheet:

1. Expand the raw file card to view its sheets.

2. Click the refresh icon on the sheet you want to reclean.

Decode will reprocess the sheet using the current mapping values and AI extraction logic. The sheet's status will change to "Processing" while this occurs.

Manual Structure Override:

If the AI is incorrectly detecting your spreadsheet's structure (wrong header rows or data boundaries), you can manually specify the correct row ranges. Click the **"Review"** button on the sheet, then use the **"Fix header/body rows"** button to [manually override the structure detection](#). This bypasses AI detection entirely and uses your exact specifications.

Viewing Diffs (Data Changes)

For sheets that have been successfully ingested (**Completed** status) or have failed, you can view the **diffs**—the numerical changes that sheet made (or attempted to make) to your database table.

To view diffs:

1. Expand the raw file card to find a completed or failed sheet.
2. Click the expand/details icon on the sheet card.

This opens a dialog showing the diff table: a record of rows added, updated, or modified when that sheet was ingested.

Mapping Preview:

For sheets with **PreviewReady** or **Completed** status, you can also click directly on the sheet card to open a visual preview of how the spreadsheet columns are mapped to your table columns.

Bulk Selection

The Profile Dashboard supports bulk operations for efficiency:

1. Use the checkboxes on raw file cards to select entire files (including all their sheets).
2. Use the checkboxes on individual sheet cards to select specific sheets.
3. A selection toolbar appears showing how many items are selected.

4. From the toolbar, you can:

- **Download** all selected items as a ZIP.
- **Delete** all selected items (queues them for deletion).
- **Clear** the selection.

Summary of Actions by Status

Status	Available Actions
Processing	Download only (deletion disabled)
PreviewReady	Download, Delete, Reclean, Review, View Mapping
MissingRowIdentifier	Download, Delete, Reclean, Review
Completed	Download, Delete, Reclean, View Diffs, View Mapping
Failed	Download, Delete, Reclean, View Diffs

For guidance on reviewing files that need attention, see [Reviewing and Approving Uploads](#).

Manually Fixing Structure Detection

Decode uses AI to automatically detect the structure of your spreadsheets—identifying where column headers begin and end, and where the data body is located. While this works reliably for most spreadsheet formats, occasionally the AI may misinterpret unusual layouts.

When this happens, you can **manually override** the AI's structure detection by telling Decode exactly which rows contain your headers and which rows contain your data.

When to Use Manual Override

Consider using manual override when:

- The AI incorrectly identified your header rows (e.g., included metadata rows as headers, or missed multi-row headers)
- The extracted schema doesn't match what you see in your spreadsheet
- The data preview shows header text instead of actual data values
- You see unexpected or garbled column names in the mapping table
- A sheet repeatedly fails processing due to structure detection issues

How It Works

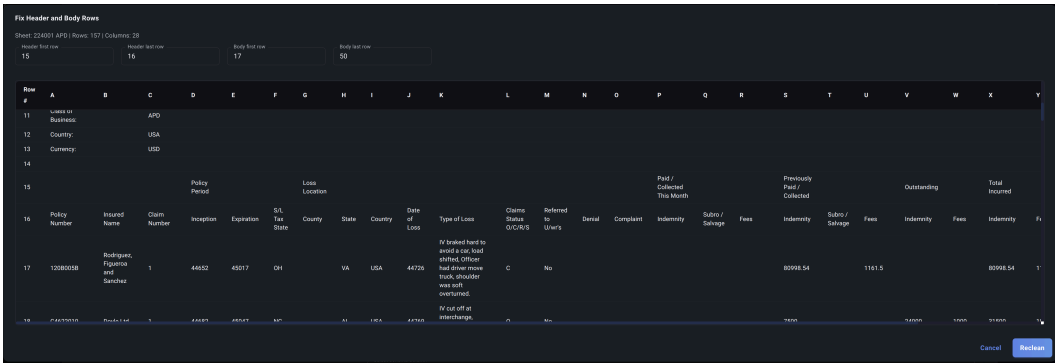
When you provide manual row bounds, Decode bypasses the AI structure detection entirely and uses your specified row numbers directly. This means:

1. **No AI guessing** – your exact row specifications are used
2. **Deterministic results** – the same input always produces the same output
3. **Full control** – you define precisely which rows form the header and which form the data body

Accessing the Structure Override Tool

1. Navigate to the sheet that needs correction (in `PreviewReady`, `MissingRowIdentifier`, or `Failed` status)
2. Click **"Review"** to open the Preview & Mapping Dialog
3. In the **Extracted Schema** section, click the **"Fix header/body rows"** button

This opens the **Header and Body Row Selector** dialog.



Header Extraction

Using the Header and Body Row Selector

The selector dialog displays a preview of your raw spreadsheet data and allows you to specify four values:

Field	Description
Header first row	The row number where your column headers begin (1-based)
Header last row	The row number where your column headers end
Body first row	The row number where your actual data begins (must be after header last row)
Body last row	The row number where your data ends

Example

Consider a spreadsheet with this structure:

Row 1:	Report Month: January 2024
Row 2:	Wholesaler: ABC Insurance
Row 3:	(empty)
Row 4:	Policy Number Insured Name Premium Type Amount
Row 5:	Gross Net
Row 6:	POL-001 Acme Corp 5000 4500
Row 7:	POL-002 Beta Ltd 3000 2700
...	
Row 150:	POL-145 Zeta Inc 2500 2250

In this case, you would specify:

- **Header first row:** 4 (where "Policy Number" starts)
- **Header last row:** 5 (the second header row with "Gross" and "Net")
- **Body first row:** 6 (where "POL-001" starts)
- **Body last row:** 150 (the last row of actual data)

Rows 1–3 contain metadata (which Decode extracts separately) and are not part of the column headers.

Validation Rules

The selector enforces these rules:

- **Maximum 4 header rows** – Column headers cannot span more than 4 rows
- **No overlap** – The header last row must be before the body first row
- **Valid ordering** – First rows must come before last rows
- **Positive numbers** – All values must be 1 or greater

If your inputs don't meet these rules, you'll see a validation error and the "Reclean" button will be disabled.

What Happens After You Submit

When you click "**Reclean**" with valid row bounds:

1. Decode re-processes the sheet using your exact row specifications
2. The column schema is extracted from the rows you specified as headers
3. The mapping is regenerated based on the new schema
4. The sheet status updates (typically to `PreviewReady` or `MissingRowIdentifier` depending on whether the Row Identifier was found)
5. The Preview Dialog refreshes to show the corrected extraction

You'll then need to review the updated mapping and approve the sheet as usual.

Tips for Success

Finding the Right Rows

- Use the scrollable table preview in the dialog to visually identify row numbers

- Look for the transition from descriptive text to repeated column patterns (that's your header start)
- Multi-row headers often have a parent row (like "Premium") with child rows below (like "Gross", "Net")
- The body starts where you see actual data values (policy numbers, names, amounts, etc.)

Handling Multi-Row Headers

Insurance bordereaux commonly use hierarchical headers spanning 2–3 rows. For example:

Row 5:	Premium		Commission		
Row 6:	Gross	Net	Amount	Rate	
Row 7:	5000	4500	500	10%	

Here, rows 5–6 are headers (hierarchical), and row 7 is where data begins.

AI Bounds as Starting Point

When you open the structure override dialog, the input fields are pre-populated with the AI's best guess (if available). These appear under `ai_header_bounds` and `ai_body_bounds` in the sheet data. Use these as a starting point and adjust as needed.

Preserving Your Override

Once you submit a manual override, Decode stores your specified bounds with the sheet. If you later need to reclean the sheet (e.g., after updating possible values), the system will use your manual bounds rather than re-running AI detection.

Troubleshooting

Issue	Solution
"Invalid bounds" error	Ensure header rows don't overlap with body rows and all values are positive
Still seeing wrong schema after reclean	Double-check your row numbers against the preview table; remember rows are 1-based

Issue	Solution
Reclean button disabled	Check that all four fields are filled and validation passes
Raw sheet preview unavailable	The original sheet file may not have been preserved; contact support

Manual structure override gives you full control when the AI needs a helping hand. For most uploads, the automatic detection works seamlessly, but this tool ensures you're never blocked by an unusual spreadsheet format.

Return to [How Mapping Works](#) or continue to [Correcting Mappings](#).

Common Errors

Approving the Upload

The final step in the review process for an uploaded spreadsheet sheet is **Approving** it. This action confirms that you are satisfied with the schema extraction, the column mappings (including any corrections or formulas you've applied), and the sample data preview.

Approval is the trigger that tells Decode to proceed with ingesting the data from that sheet into your target Decode Table.

Before You Approve - Final Checks

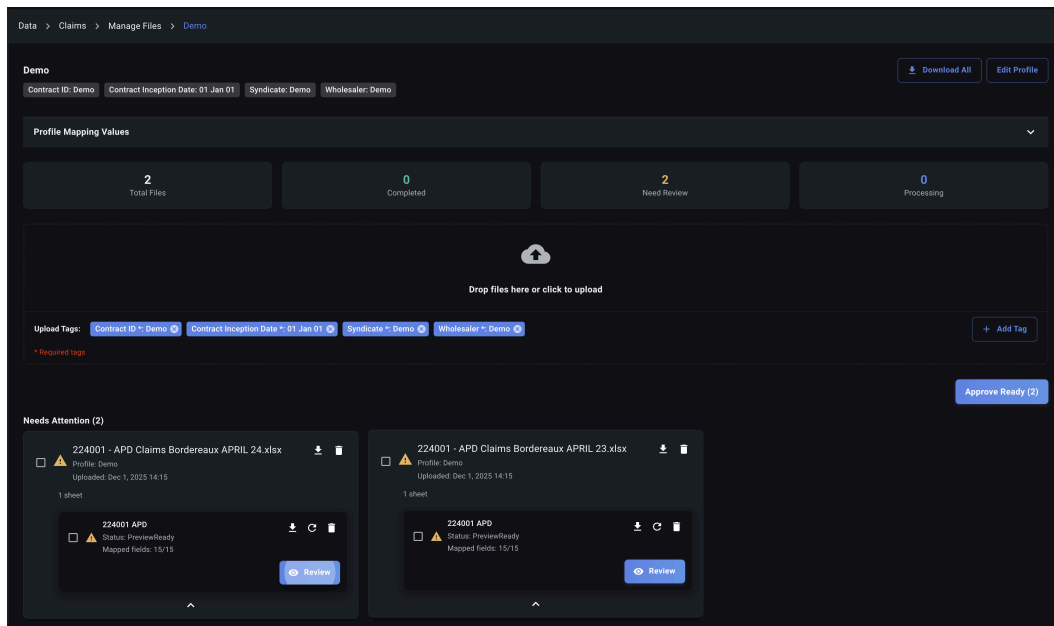
Before clicking the "Approve" button in the [Preview & Mapping Dialog](#), perform these quick final checks:

1. **Row Identifier: Crucially**, ensure the Table's Row Identifier column is correctly mapped to the unique identifier column in your spreadsheet. If the sheet had a `MissingRowIdentifier` status, this *must* be resolved first.
2. **Mandatory Columns**: Verify that all columns marked as **Mandatory** in your Table schema have a valid mapping (either a direct Simple mapping to a spreadsheet column/header or a Formula mapping). If any mandatory column is mapped to `Not Present` or left unmapped, the ingestion will fail.
3. **Formula Mappings (If Used)**: Briefly review the formulas you created and glance at the Sample Data Preview to ensure the calculated results look reasonable.
4. **Sample Data**: Does the data in the Sample Data Preview generally look correct and appear in the right columns based on your mappings?

Clicking "Approve"

Once you are confident with your review and final checks:

1. Click the **"Approve"** button at the bottom of the Preview & Mapping Dialog. Note you can also **"Approve All"** to batch approve ready files.



Approve Upload

What Happens Next?

- The dialog closes.
- Decode saves the final mapping configuration (including any corrections or formulas) associated with this specific spreadsheet sheet. This helps with potential reprocessing or auditing later.
- The system adds any *newly mapped* spreadsheet column names to the "possible values" list for the corresponding Table columns, improving future automation.
- The sheet's status typically changes to **Ready** or directly to **Processing** for ingestion.
- Decode's backend system queues the sheet for final ingestion into the target Table. This involves:
 - Applying the confirmed mappings to extract data row by row.
 - Adding the selected Metadata Tags to each row.
 - Calculating any formula-based columns.
 - Using the Row Identifier to determine if rows should be inserted as new records or used to update existing records (potentially adding new dynamic period data).
 - Dynamically adding new columns to the table if data for a new period (e.g., a new month) is encountered in a dynamic column structure.
- Once ingestion is complete, the sheet status should update to **Completed**. If errors occur during ingestion, the status will change to **Failed**, and an error message should be available.

You have successfully reviewed and approved the data! You can monitor the final status in the Data Profile Dashboard.

Now that you understand the review process, let's look at [Understanding Common Upload Errors](#). *(Link assumes the next file will be created directly under `uploading-data`)*

Handling Sheets with Missing Row Identifiers

When you upload a spreadsheet sheet, one of the first things Decode tries to do is find and map the **Row Identifier** column you defined when [creating the Table](#).

If Decode **cannot** automatically find a matching column in the spreadsheet for your designated Row Identifier, the sheet's status will become **MissingRowIdentifier**.

Why is this Status Critical?

The Row Identifier is essential for Decode to function correctly:

- **Tracking Records:** It's the unique key used to distinguish between different policies, claims, or other entities.
- **Updating Data:** When new data arrives (e.g., next month's premium), Decode uses the Row Identifier to find the existing record and update it (or add a new dynamic period).
- **Preventing Duplicates:** It ensures that the same policy or claim isn't accidentally added multiple times.

Without a correctly mapped Row Identifier, Decode cannot safely ingest the data into your Table. Therefore, processing is paused until you manually provide the correct mapping.

How to Resolve **MissingRowIdentifier** Status

1. **Locate the Sheet:** Find the sheet with the **MissingRowIdentifier** status in the recent uploads section.
2. **Open for Review:** Click the **"Review"** button for that sheet. This opens the [Preview & Mapping Dialog](#).
3. **Identify the Row ID Row:** In the **Mapping Table** section, find the row corresponding to your Table's designated Row Identifier column (it will likely be highlighted or marked as unmapped).
4. **Select the Correct Mapping (Crucial Step):**

- Ensure the Mapping Type toggle is set to **"Simple"**.
- Click the dropdown in the "Mapping" column for the Row Identifier row.
- Carefully select the **spreadsheet column name** from the list that *actually contains* the unique identifier (e.g., `Policy #` , `Claim Ref` , `Member ID`).

Double-check this selection.

5. **Review Other Mappings:** While you're here, review the rest of the mappings as described in [Correcting Mappings](#). **This is crucial** because you once you approve the file, it will be ingested.

6. **Approve:** Once you have correctly mapped the Row Identifier (and are satisfied with other mappings), click the **"Approve"** button.

What Happens Next?

- Decode saves your manual mapping for the Row Identifier.
- It adds the selected spreadsheet column name to the "possible values" for the Row Identifier Table column, helping it map automatically next time.
- The sheet's status will change (e.g., to `Ready` or `Processing`), and Decode will proceed with the final data ingestion into your Table.

Resolving the `MissingRowIdentifier` status promptly by providing the correct mapping is essential for keeping your data flowing into Decode.

Next, let's look at [Mapping Header Data](#).

Mapping Header Data

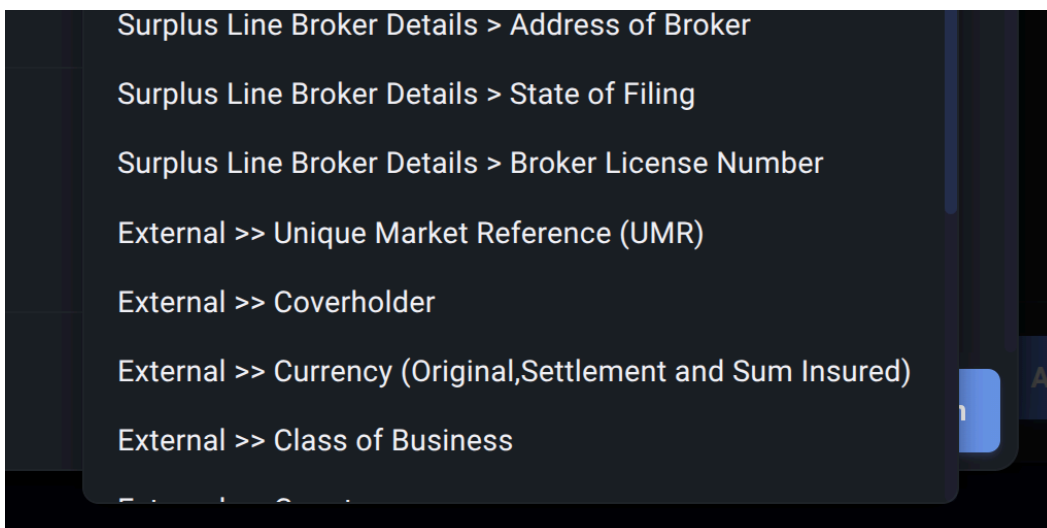
Often, important contextual information isn't located in the main data columns of your spreadsheet but resides in the **header rows** at the top of the sheet. Examples include:

- Report Month or Quarter
- Coverholder
- Contract ID
- Currency
- Class of Business

Decode allows you to extract this header information and map it to specific columns in your target Decode Table, just like regular data columns.

How Header Data is Handled

1. **Extraction:** During the initial processing phase, Decode attempts to identify key-value pairs in the top few rows of your spreadsheet sheet. It looks for common patterns like "Label: Value".
2. **Availability in Mapping:** Any successfully extracted header fields are made available in the mapping dropdown during the [review process](#). They are typically prefixed with `External >>` to distinguish them from regular spreadsheet columns (e.g., `External >> Report Month`, `External >> wholesaler`).



Mapping Header Data to Table Columns

The process is identical to performing a [Simple Mapping](#):

1. **Locate Target Table Column:** In the **Mapping Table** section of the Preview Dialog, find the row for the Table Column where you want to store the header data (e.g., a Static Column named `Report Period` or a Metadata Column named `Wholesaler Name`).
2. **Ensure "Simple" Mapping Type:** Make sure the toggle for that row is set to "Simple".
3. **Select Header Field:** Click the dropdown in the "Mapping" column. Scroll through the list and select the appropriate header field, identifiable by the `External >>` prefix (e.g., select `External >> Report Month` to map it to your `Report Period` Table column).
4. **Review Sample Data:** Check the Sample Data Preview at the bottom. While the header data itself won't appear as a *column* in the sample (since it applies to *all* rows from that sheet), ensure other mappings still look correct. The header value will be added to each row during final ingestion.
5. **Confirm/Approve:** Include this mapping when you [approve the sheet](#).

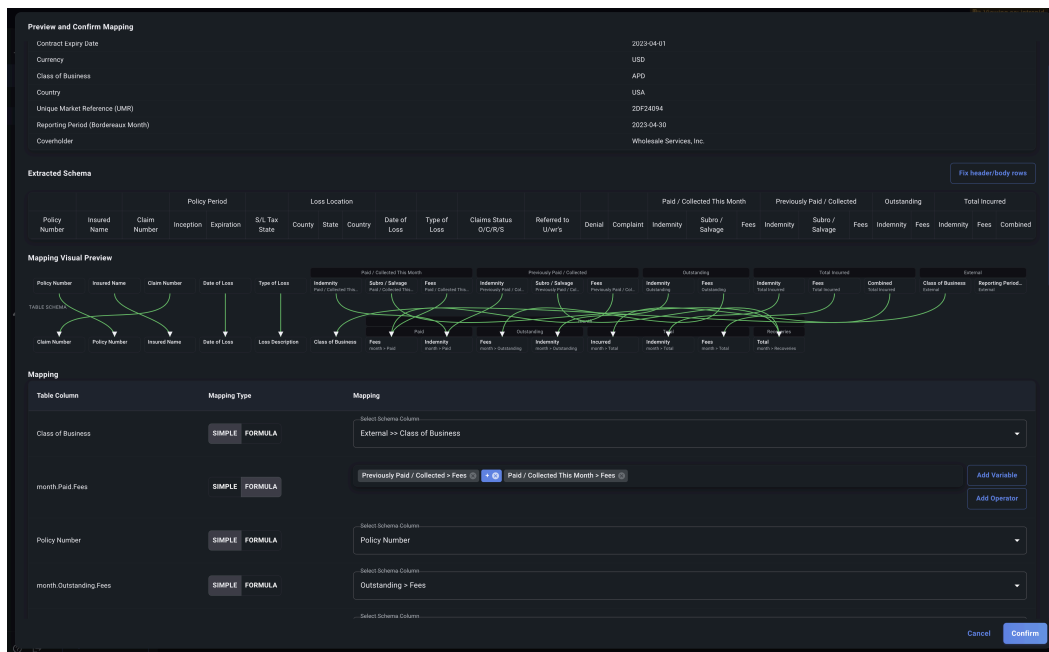
By mapping relevant header data, you enrich the data within your Decode Table, adding valuable context directly alongside your core metrics.

Finally, let's look at the last step: [Approving the Upload](#).

Understanding the Preview Screen

After Decode processes an uploaded spreadsheet sheet and its status becomes `PreviewReady` or `MissingRowIdentifier`, you need to review it before the data can be added to your Table. Clicking the "Review" button for that sheet in the **Recent Uploads** section will open the **Preview & Mapping Dialog**.

This dialog is your central hub for verifying Decode's automated work and ensuring data accuracy. It contains several key sections:



Preview Dialog

Key Sections of the Preview Dialog

1. Header Metadata (If Applicable):

- **What it shows:** Information Decode extracted from the header rows (top few rows) of your spreadsheet *if* you have mapped any header fields during the [Table Creation](#) or mapping correction process.
- **Purpose:** Allows you to verify that key contextual data (like 'Report Month', 'Wholesaler Name' if present in the header) was correctly identified.

- **What it shows:** If Decode detected that columns in your Table schema are missing from the *spreadsheet* or couldn't be mapped, they will be listed here.
- **Purpose:** Highlights critical omissions that would cause the upload to fail if not addressed.

5. Sample Data Preview:

- **What it shows:** The first few rows of data *as they would appear in your Table* based on the *current* mapping settings in the dialog.
- **Purpose:** Allows you to quickly sanity-check the results. Do the values look correct? Are they in the right columns? This helps catch mapping errors visually.

6. Action Buttons:

- **Cancel:** Closes the dialog without saving any changes you made to the mappings.
- **Approve:** Saves any mapping changes you made and signals to Decode that this sheet is ready for final ingestion into the Table. **You must click this for the data to be loaded.**

Familiarize yourself with these sections. Your primary task in this dialog is to carefully examine the **Mapping Table** and use the **Sample Data Preview** to ensure everything aligns correctly before clicking **Approve**.

:::tip

If the extracted schema doesn't match your spreadsheet's actual structure, use the **"Fix header/body rows"** button to [manually override the AI detection](#) before proceeding with mapping corrections.

:::

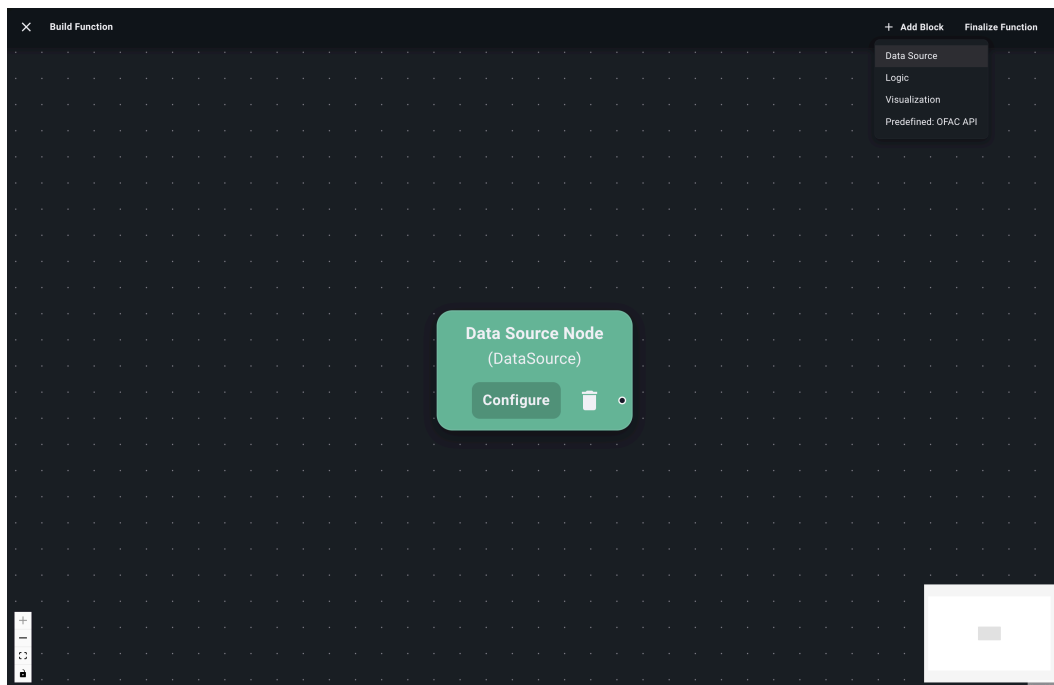
Next, let's understand [How Mapping Works](#) in more detail.

Step 1: Adding a Data Source Node

Every Data Function starts with retrieving data. The **Data Source Node** is the first building block you'll add to your function's flowchart. Its purpose is to specify *which* [Decode Table](#) contains the data you want to include in your analysis.

Adding the Node

1. **Create or Edit a Function:** Start by [creating a new function or editing an existing draft](#). This will open the **Function Builder** with a blank canvas or your existing flowchart.
2. **Open the "Add Block" Menu:** Click the **"Add Block"** button (found in the header or toolbar of the builder).
3. **Select "Data Source":** Choose **"Data Source"** from the dropdown menu.
4. A new Data Source node will appear on the canvas.

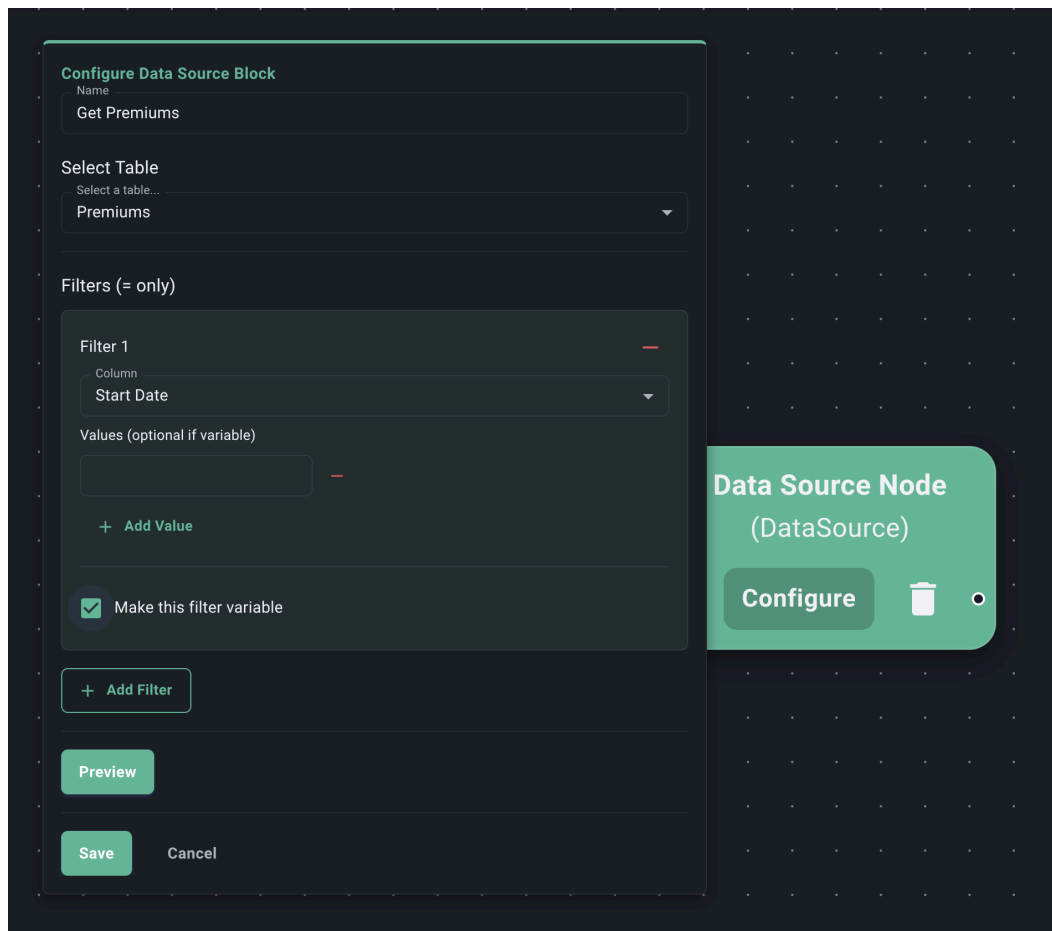


Data Source

Configuring the Data Source Node

Newly added nodes need configuration.

1. **Open Configuration:** Click the "**Configure**" button on the Data Source node you just added.
2. The **Data Source Block Configuration** panel or dialog will appear.



Configure Data Source Panel

3. **Name (Optional but Recommended):** Give this data source step a descriptive name (e.g., "Get Premium Data", "Select Claims FY2023").
4. **Select Table:** Choose the specific Decode Table you want to pull data from using the "**Select a table...**" dropdown. This list will contain all the Tables you have created on the Data Page.
5. **Define Filters:** This is where you specify criteria to narrow down the data retrieved from the table *before* it moves to the next step in your function.
 - Click "**Add Filter**".
 - **Column:** Select the Table column you want to filter on (e.g., `Region` , `Policy_Status` , `Inception_Date` , or special filters like `Start Date` , `End Date`).
 - **Value(s):** Enter the specific value(s) to filter by.

- For most columns, you can add multiple values (using the "+" button next to the value field) to filter for rows matching *any* of those values (e.g., filter **Region** for "North America" OR "Europe").
- For **Start Date** and **End Date**, provide a single date. These are often used to define the time period for your analysis.
- **Make Variable (Checkbox):** This is a key feature for reusability!
 - **If unchecked:** The filter value you enter here is *fixed*. The function will *always* use this value when running.
 - **If checked:** This filter becomes a **parameter**. When you [run the function later](#), this filter will appear in the sidebar, allowing you to easily change its value without editing the function itself. **Check this box for criteria you expect to change often**, like date ranges, specific wholesalers, regions, etc. You can provide optional *default* values here, which will be pre-filled when running the function.

Filter 1

Column: Start Date

Values (optional if variable):

+ Add Value

☒ Make this filter variable

Filter 2

Column: End Date

Values (optional if variable):

+ Add Value

☒ Make this filter variable

+ Add Filter

Preview

Save Cancel

Preview Data

Policy Number	Insured Name	State	Contract ID	Syndicate	Wholesal
208131-031	Christopher	CA	208131	Ascot	Bass
208131-030	Gale	AL	208131	Ascot	Bass
208131-048	Soirse	AL	208131	Ascot	Bass
208131-050	Hernandez	MS	208131	Ascot	Bass
204002-021	Sally	AL	204002	Amlin	Avant
204002-024	Dwayne	GA	204002	Amlin	Avant
204002-023	Sarah	NC	204002	Amlin	Avant
204002-012	John	AR	204002	Amlin	Avant

Rows per page: 10 1-8 of 8

Set Up Configuration

6. Preview:

- Click the **"Preview"** button.
- Decode runs a query based on your selected table and *non-variable* filters (using default values for variable ones if provided) and shows a sample of

the data that this node will output.

- **Verify:** Check the preview table. Does it contain the kind of data you expect based on your filters? This helps catch errors early. *You must run a preview after making changes before you can save.*

7. **Save:** Once you are satisfied with the table selection, filters, and preview, click **"Save"**.

The configuration panel will close, and your Data Source node is now set up. It serves as the starting point, providing filtered data to the next node you connect it to.

Next, you'll typically connect this to a [Logic Node](#) to perform calculations or transformations.

What are Data Functions?

Data Functions are the core analysis tool within Decode's Dashboard Page. Think of them as **reusable, visual pipelines** that allow you to:

1. **Select** specific data from one or more of your [Tables](#).
2. **Filter** that data based on criteria you define (some of which can be easily changed each time you run the function).
3. **Transform and Analyze** the data using custom logic (often built with AI assistance) or pre-built tools.
4. **Visualize** the results as tables or charts.

Why Use Data Functions?

Instead of performing one-off analyses in spreadsheets, Data Functions offer several advantages:

- **Reusability:** Build the analysis logic once, then run it again and again with different parameters (like changing dates or regions) without rebuilding the entire process.
- **Consistency:** Ensures the same logic is applied every time, reducing manual errors.
- **Complexity Handled:** Allows for multi-step analyses involving data retrieval, complex calculations, and visualization, all within a single workflow.
- **Accessibility:** Enables users without strong technical or coding backgrounds to perform sophisticated analysis by leveraging AI assistance for logic creation.
- **Parameterization:** Easily define inputs (like Start Date, End Date, Wholesaler Name) that can be changed when running the function, allowing you to explore different data slices quickly.

The Flowchart Concept

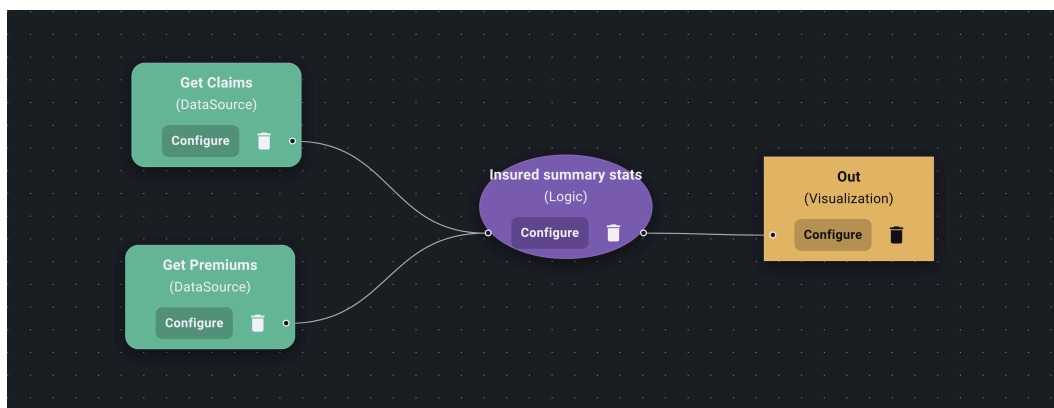
You build Data Functions visually using a flowchart-like interface. You connect different **Nodes**, each performing a specific task:

- **Data Source Nodes:** Specify which Table(s) to pull data from and define initial filters. You can also mark certain filters as **parameters** that can be changed

later when running the function.

- **Logic Nodes:** Apply custom transformations or calculations. You interact with an AI assistant to define this logic (e.g., "Calculate the loss ratio," "Group by state and sum premiums").
- **Predefined Nodes:** Integrate specialized, pre-built tools or external API calls into your workflow (e.g., running a list of names through a sanctions check).
- **Visualization Nodes:** Determine how the final results are displayed – as a data table, a bar chart, or a line chart. AI assistance helps configure chart mappings.

By connecting these nodes, you create a clear, visual representation of your analysis pipeline.



Flow Chart

Next, let's walk through [Building Your First Function](#). *(This link assumes the next sub-category)*

The Dashboard Page: Analyzing Your Data

Welcome to the **Dashboard Page**, the central hub in Decode for analyzing the data you've meticulously organized on the [Data Page](#). This is where you move beyond simple data viewing and start deriving meaningful insights through reusable analysis pipelines (**Data Functions**) and customizable dashboards (**Views**).

If the Data Page is where you prepare your ingredients (clean data), the Dashboard Page is where you cook your meals (perform analysis and create reports).

Key Capabilities

The Dashboard Page revolves around two primary features:

1. Data Functions:

- Build reusable workflows to query, transform, analyze, and visualize your data.
- Combine data from one or more Tables.
- Apply custom logic using an AI assistant – simply describe what you want to calculate or manipulate, and the AI helps generate the necessary steps.
- Integrate pre-built analytical tools (Predefined Nodes) like compliance checks.
- Define parameters (like date ranges or regions) that can be easily changed each time you run the function.
- Visualize results as tables, bar charts, or line charts.
- Learn more: [Working with Data Functions](#)

2. Views:

- Create custom dashboard layouts by arranging your saved Data Functions onto a grid.
- Run multiple functions side-by-side, potentially with different parameters, to compare results and gain a holistic understanding.
- Save these layouts for quick access to your key reports and analyses.
- Learn more: [Creating and Using Views](#) (Link will work once page exists)

The Goal: Insight and Efficiency

The Dashboard Page aims to:

- **Democratize Analysis:** Enable users, even those without coding skills, to perform sophisticated data analysis through AI assistance and a visual interface.
- **Promote Reusability:** Build an analysis once as a Data Function and run it repeatedly with different inputs, saving significant time and effort.
- **Facilitate Comparison:** Use Views to easily compare different data segments, time periods, or scenarios side-by-side.
- **Generate Insights:** Move beyond raw data to uncover trends, calculate key performance indicators (KPIs), and answer critical business questions.
- **Automate Processes:** Data functions and views allow you to construct standard workflows (e.g. checking for missing premiums, or calculating loss ratios) and re-use them at the push of a button.

Ready to start analyzing? Explore the sections on **Data Functions** and **Views** using the sidebar navigation.

What are Views?

While [Data Functions](#) allow you to build powerful, reusable analyses, **Views** let you take the next step by combining multiple finalized Data Functions onto a single, customizable dashboard grid.

Think of a View as your personalized reporting dashboard within Decode.

The Purpose: Comparison and Overview

Views are designed to help you:

- **Compare Results:** Place related Data Functions side-by-side to easily compare metrics across different segments, time periods, or scenarios (e.g., view profitability for Region A next to profitability for Region B).
- **Get a Holistic Overview:** Combine key functions representing different aspects of your business (e.g., premium summaries, claim trends, loss ratios) onto one screen for a comprehensive snapshot.
- **Save Common Layouts:** Create and save specific arrangements of functions that you frequently need to consult together.

How They Work

- **Composition:** Views are composed of one or more **Final** Data Functions. You cannot add Draft functions to a View.
- **Grid Layout:** You arrange the selected functions visually on a flexible grid, controlling their position and size.
- **Combined Parameters:** When viewing a saved View, the right sidebar conveniently aggregates the **variable parameters** from *all* the functions included in that View, allowing you to control their inputs from one place.
- **Independent Execution:** Each function within the View still runs its own analysis pipeline when triggered.

Views provide a simple yet powerful way to organize and interact with the outputs of your various Data Functions, turning individual analyses into a cohesive dashboard experience.

Next, learn how to create your own dashboard by [**Building a View**](#).

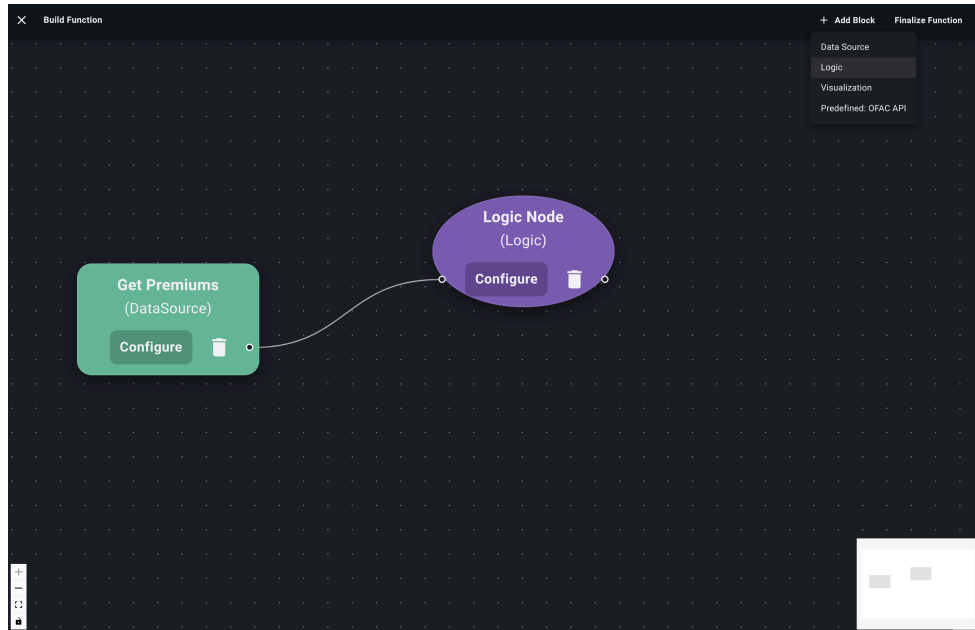
Step 2: Adding a Logic Node

After retrieving data with a [Data Source Node](#), the **Logic Node** is where you perform calculations, transformations, filtering, grouping, or any other custom manipulation of that data.

Decode makes this powerful by allowing you to **describe the logic you need in plain English using an AI analyst**. The AI then generates code to implement your logic and runs the logic behind the scenes.

Adding the Node

1. **Open the "Add Block" Menu:** In the Function Builder Dialog, click **"Add Block"**.
2. **Select "Logic":** Choose **"Logic"** from the menu.
3. A new Logic node will appear on the canvas.



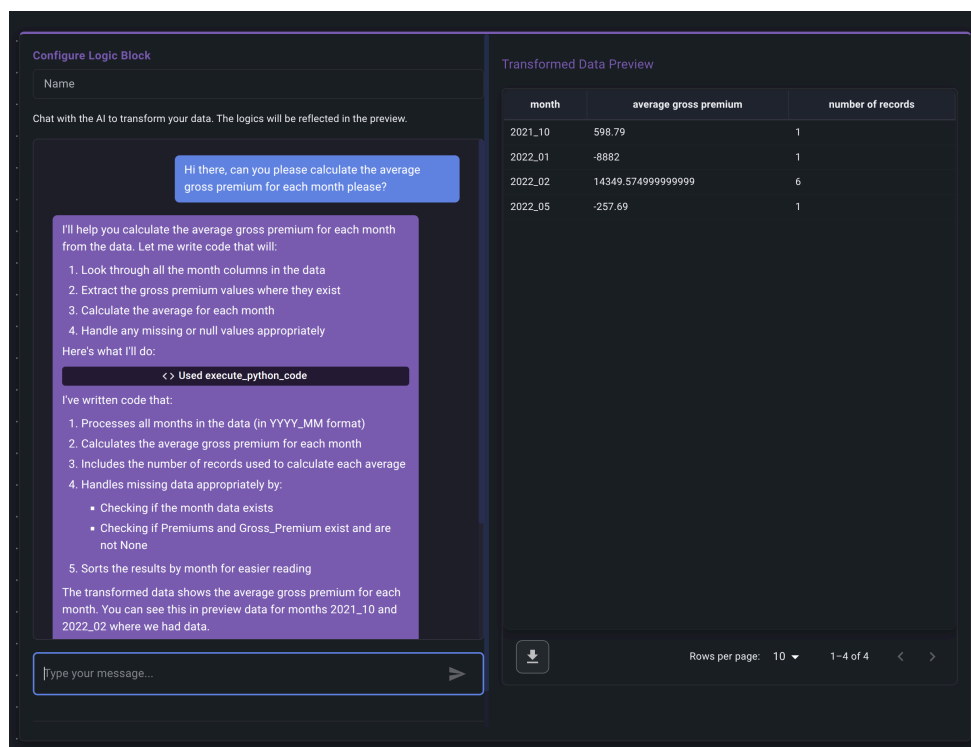
Add Logic Node

4. **Connect Nodes:** You need to connect the output handle (usually on the right) of your Data Source Node to the input handle (usually on the left) of the new Logic Node. Click and drag from one handle to the other.

- **Important:** You generally must connect an input (like a Data Source Node) *before* you can configure the Logic Node.

Configuring the Logic Node (Interacting with AI)

1. **Open Configuration:** Click the "**Configure**" button on the Logic Node.
2. The **Logic Block Configuration** panel or dialog will appear. This interface is primarily a chat window.



Configure Logic Node

3. **Name (Optional but Recommended):** Give this logic step a descriptive name (e.g., "Calculate Loss Ratio", "Filter California Policies", "Group by Wholesaler").
4. **Instruct the AI:** In the chat input box at the bottom, type instructions describing the transformation or calculation you want to perform on the data coming *from the connected input node(s)*. Be clear and specific.

Examples:

- "Calculate a new column called 'Net Premium' by subtracting the 'Commission Amount' column from the 'Gross Premium' column."

- "Filter the data to only include rows where the 'Policy State' column is equal to 'CA'."
- "Group the data by the 'Wholesaler Name' column and calculate the sum of 'Net Premium' for each wholesaler. Call the summed column 'Total Net Premium'."
- "Show me all the policies that missed a premium payment inside of their contract window."

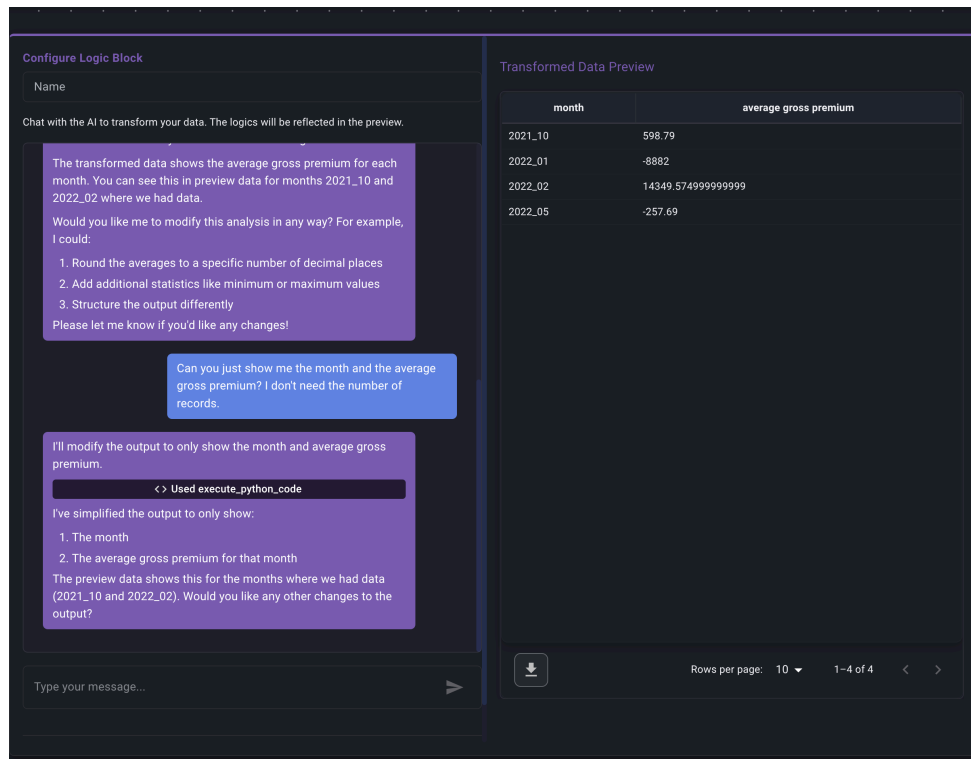
5. **Send Message:** Press Enter or click the Send button.

6. **AI Processing:** Decode's AI assistant will:

- Interpret your request.
- If the AI needs any clarity or further specification, it will ask follow up questions.
- Generate Python code to perform the requested operation.
- Run this code on a *sample* of the input data.
- Respond in the chat, confirming its understanding or showing the sample results.

7. **Review AI Response & Preview:**

- Read the AI's text response.
- Crucially, look for a **Preview Table** displayed within or alongside the chat. This shows the *result* of the AI's generated code applied to the sample data.
- **Verify the Preview:** Does the output table look correct?



Chat with the AI

8. Iterate if Necessary:

- If the preview isn't right, **chat further with the AI**. Provide clarifying instructions or corrections.
- **Example Corrections:** "Actually, I need the average premium, not the sum." or "The 'Net Premium' calculation seems wrong, please make sure you're using coverholder commision, not total commision."
- Continue chatting and reviewing previews until the sample output matches your requirements.

9. Save the Logic: Once you are satisfied with the preview generated by the AI's final correct instruction:

- Click the **"Save"** button. This saves the *last successful code snippet* generated by the AI as the logic for this node.
- The configuration panel will close.

Your Logic Node is now configured. It will take data from its input, apply the AI-generated transformation you approved, and pass the results to the next connected node.

Key Points about AI Interaction:

- **Be Specific:** The clearer your instructions, the better the AI can understand. Mention column names exactly as they appear in the input data (check the Data Source preview if unsure).
- **Focus on the Goal:** Tell the AI *what* you want to achieve, not necessarily *how* to code it.
- **Trust but Verify:** Always check the Preview Data carefully. The AI is a tool for turning your logic into reusable functions; you are responsible for ensuring the final logic is correct.

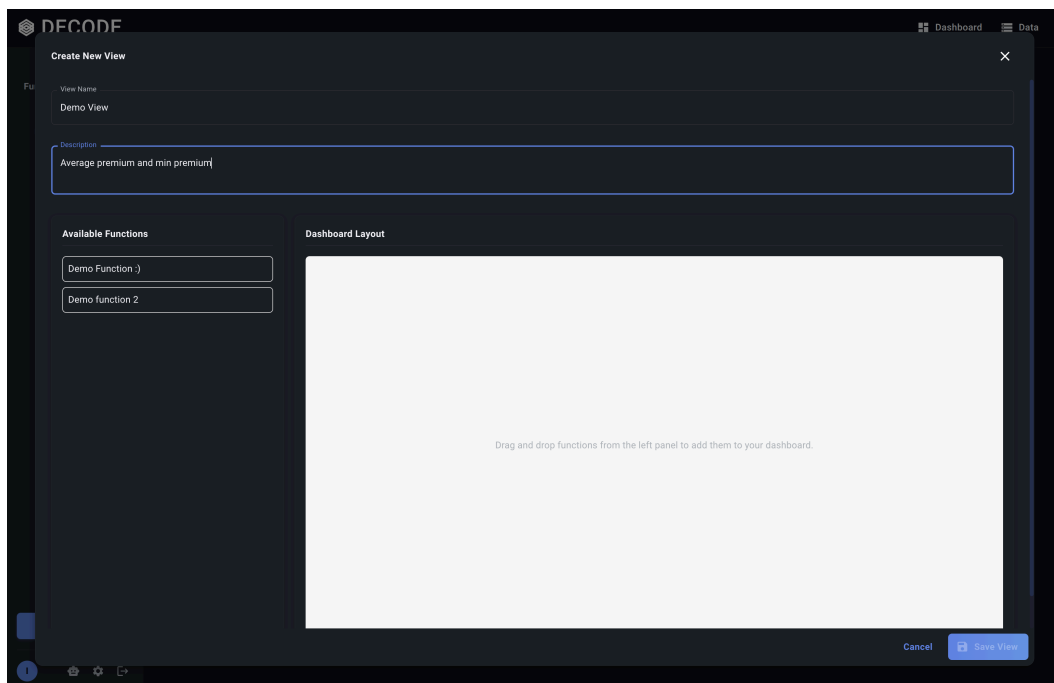
After defining your logic, you will connect this node to a [Visualization Node](#) to display the results.

Building a View

Creating a View allows you to design a custom dashboard layout featuring the outputs of your finalized Data Functions.

Starting the View Builder

1. Navigate to the **Dashboard Page**.
2. Ensure the **"Views"** tab is selected in the left sidebar.
3. Click the **"New View"** button.
4. This opens the **View Builder**.



Build a View

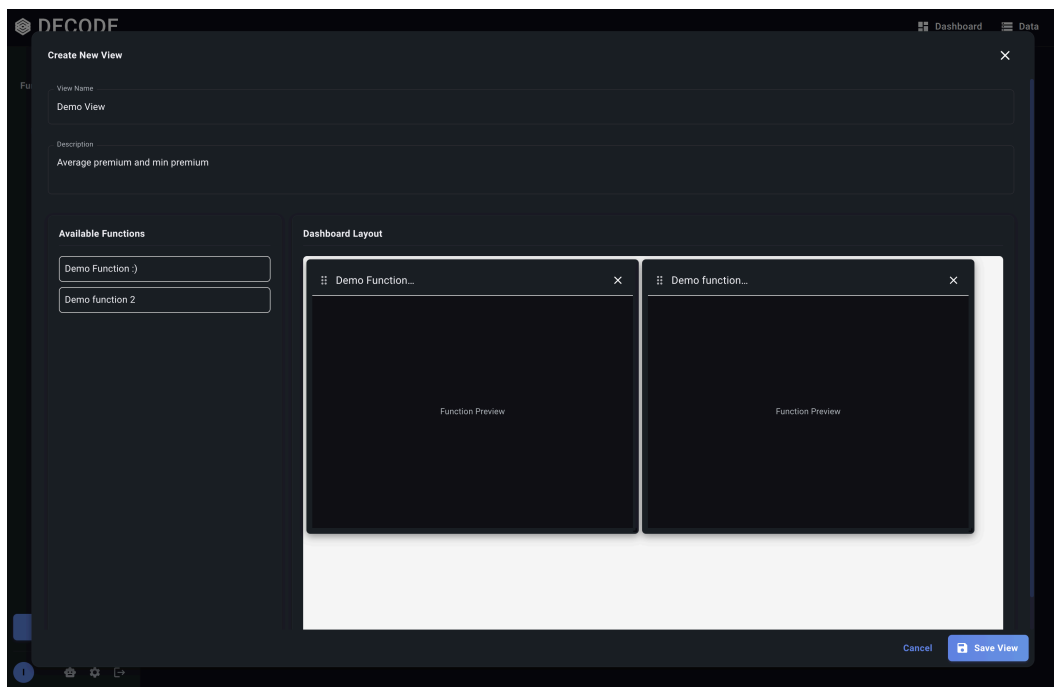
The View Builder Interface

The View Builder has two main sections:

- **Available Functions (Left Panel):** Lists all your **Final** Data Functions. You cannot add Draft functions.
- **Dashboard Layout (Right Panel):** A blank grid representing your dashboard canvas where you will place and arrange functions.

Adding Functions to the Grid

1. **Drag and Drop:** Find the Data Function you want to add in the "Available Functions" list on the left. Click and hold on the function, drag it over to the grid layout area on the right, and release the mouse button.
2. A placeholder card representing the function will appear on the grid.
 - *Note:* This card is just for layout purposes in the builder; it doesn't show the function's actual output here.



Configure a View

3. **Repeat:** Drag and drop other desired functions onto the grid. You can add multiple instances of the *same* function if needed (e.g., to run it with different parameters side-by-side later).

Arranging and Resizing Functions

Once functions are on the grid:

- **Move:** Click and drag the header area of a function card to move it to a different position on the grid.
- **Resize:** Click and drag the resize handle (usually a small icon or triangle in the bottom-right corner) of a function card to make it wider or taller, spanning more

grid cells.

- **Remove:** Click the **Close icon (X)** on a function card within the grid to remove that specific instance from the View layout.

Arrange and resize the function cards until you achieve your desired dashboard layout.

Naming and Saving the View

1. **Enter View Details:** At the top of the View Builder dialog, provide:
 - **View Name:** A required, descriptive name for your dashboard (e.g., "Quarterly Performance Review", "Wholesaler X Dashboard").
 - **Description (Optional):** Briefly explain the purpose or content of this View.
2. **Save:** Click the **"Save View"** button .

What Happens:

- Decode saves the layout configuration (which functions are included, their positions, and sizes) and the View's name and description.
- The View Builder dialog closes.
- Your newly created View will now appear in the "Views" list in the Dashboard Page sidebar.

Your dashboard View is now built! The next step is to [Interact with your View](#) by running the functions within it.

Step 3 (Optional): Adding a Predefined Node

While **Logic Nodes** allow for flexible, custom transformations using AI, **Predefined Nodes** offer a way to integrate specific, pre-built functionalities into your Data Function workflow.

These nodes encapsulate more complex or specialized operations that go beyond simple data manipulation within your tables, often interacting with services outside the Decode platform.

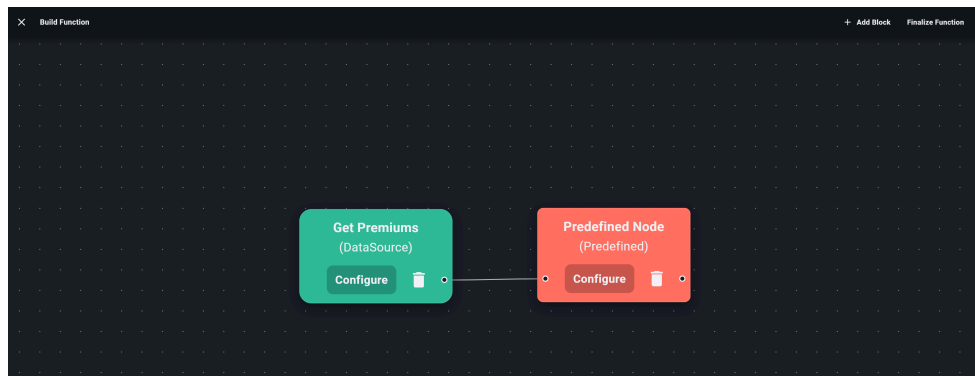
Examples of Predefined Nodes

The available Predefined Nodes depend on what has been integrated into the Decode platform. Right now:

- **OFAC Sanctions Check:** Takes a list of names/entities as input and returns potential matches against sanctions lists.

Adding the Node

1. **Open the "Add Block" Menu:** In the Function Builder Dialog, click **"Add Block"**.
2. **Select the Predefined Function:** The menu will list available Predefined Nodes, often prefixed with "Predefined:". Choose the specific one you need (e.g., "Predefined: OFAC Sanctions Check").
3. A new Predefined Node, labeled with the function's name, will appear on the canvas.
4. **Connect Input Node(s):** Connect the output handle(s) of the preceding node(s) (usually a Data Source or Logic Node) that provide the necessary input data to the input handle(s) of the Predefined Node. *Predefined Nodes require connected input before configuration.*

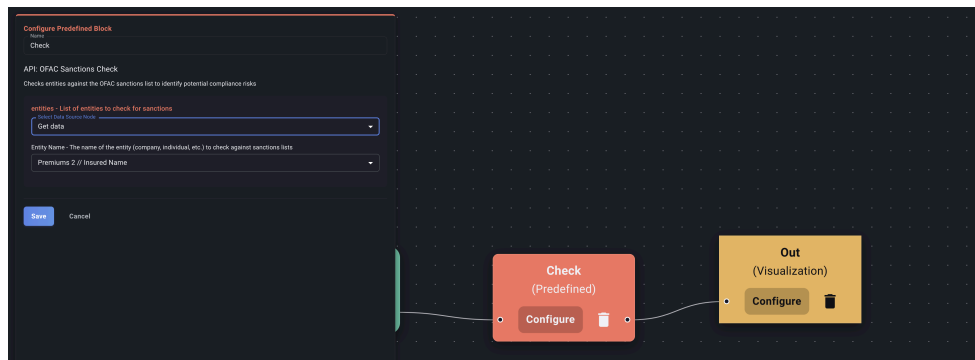


Add Predefined Node

Configuring the Predefined Node

Configuration involves mapping the data columns from your connected input node(s) to the specific input parameters expected by the predefined function/API.

1. **Open Configuration:** Click the **"Configure"** button on the Predefined Node.
2. The **Predefined Block Configuration** panel or dialog will appear.
3. **Name (Optional but Recommended):** Give this step a descriptive name (e.g., "Run Sanctions Check on Insureds").
4. **Map Input Parameters:** This is the main configuration step. For *each required parameter* listed for the API:
 - Use the dropdown list next to the parameter name.
 - This dropdown will contain column names available from the *connected input node(s)*. The columns might be prefixed with the source node's name or table name for clarity if multiple inputs are possible (e.g., `DataSource1 // Insured Name`).
 - Select the input column that contains the data corresponding to the required API parameter. For example, for an OFAC check API parameter named `entity name`, you would select the input column containing the company or individual names (e.g., `InputData // Insured Name`).
 - **Ensure all required parameters are mapped.**



Configure Predefined Node

5. **Preview (If Available):** Some Predefined Nodes might offer a limited preview based on sample input data. If available, click **"Preview"** to test the mapping and see sample output. *Note: Running previews for external APIs might have limitations or costs.*

6. **Save:** Once all required parameters are mapped, click **"Save"**.

The configuration panel will close. Your Predefined Node is now set up to receive data from the previous node, send the mapped parameters to the underlying function/API, and pass the results (e.g., sanctions matches, validated addresses) to the next node in your flowchart.

After a Predefined Node, you might connect its output to a [Logic Node](#) for further filtering/joining or directly to a [Visualization Node](#) to display the results.

Managing Data Functions

As you build more analyses, the Dashboard Page provides tools to manage your Data Functions, including handling drafts, finalizing functions, editing, and deleting them.

Function Status: Draft vs. Final

Functions exist in one of two states:

- **Draft:** A function that is still under construction or hasn't been explicitly finalized.
 - Can be edited using the [Function Builder](#).
 - Cannot be run directly from the main Dashboard Page list or added to [Views](#).
 - Appears in a separate "Drafts" section in the Dashboard sidebar.
- **Final:** A completed and saved function ready for execution.
 - Appears in the main "Functions" list in the Dashboard sidebar.
 - Can be run directly from the Dashboard Page using the Parameters Panel.
 - Can be added to Views.
 - Editing a Final function involves opening it in the Function Builder again.

Finding Your Functions

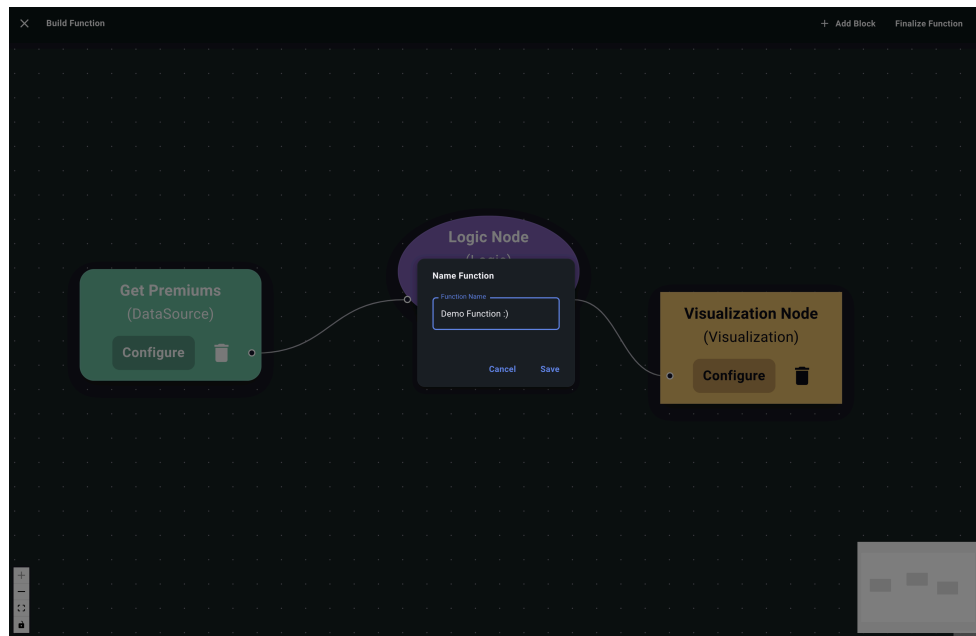
- **Final Functions:** Listed under the "Functions" tab in the left sidebar of the Dashboard Page.
- **Draft Functions:** Listed under the expandable "Drafts" section within the "Functions" tab of the left sidebar. Click the section header to expand/collapse it.

Finalizing a Draft Function

When you have finished building or editing a function in the Function Builder Dialog and are ready to run it or add it to views:

1. Click the **"Finalize Function"** button within the Function Builder Dialog header.

2. You may be prompted to **Name the Function** if you haven't already. Provide a clear, descriptive name.



Save Function

3. Click **"Save"**.
4. The Function Builder Dialog closes.
5. The function's status changes from 'Draft' to 'Final'.
6. It will now appear in the main "Functions" list in the sidebar, ready to be run.

Editing a Function

- **Editing a Draft:**

1. Find the function in the "Drafts" section of the sidebar.
2. Click the **Edit icon** (✎) next to its name.
3. The Function Builder Dialog opens, allowing you to modify the flowchart.
4. Click **"Finalize Function"** when done editing to save it as 'Final', or simply close the dialog to keep it as a draft (changes are usually saved automatically).


- **Editing a Final Function:**

1. Find the function in the main "Functions" list.
2. Click the **Edit icon** (✎). (Note: This icon might be disabled for predefined functions not created by the user).

3. The Function Builder Dialog opens.
4. Make your changes.
5. Click "**Finalize Function**" to save the updated version.

Deleting a Function

Warning: Deleting a function is permanent and cannot be undone.

1. Find the function you want to delete in either the "Functions" or "Drafts" list in the sidebar. *(Note: Deletion might be restricted for predefined functions).*
2. Click the **Delete icon** () next to its name.
3. A confirmation dialog will appear.
4. Confirm the deletion by clicking "**Delete**".

The function will be removed from the list and its associated configuration will be deleted. If the deleted function was part of any Views, it will likely disappear from those Views as well.

Interacting with Views

Once you have built and saved a View, you can use it as an interactive dashboard to run multiple analyses simultaneously and compare results.

Selecting a View

1. Navigate to the **Dashboard Page**.
2. Ensure the **"Views"** tab is selected in the left sidebar.
3. Click on the name of the **View** you want to open from the list.

What Happens:

- The main content area updates to display the grid layout you designed, with placeholders or previous results for each Data Function included in the View.
- The **Parameters Panel** appears on the right sidebar. Crucially, this panel now aggregates the **variable parameters** from *all* the functions present in the current View, typically grouped by function instance.

Using the Combined Parameters Panel

The Parameters Panel in a View allows you to control the inputs for all the functions displayed:

1. **Identify Function Parameters:** Parameters are grouped under the name of the function instance they belong to within the View.
2. **Modify Values:** Change the parameter values as needed for your current analysis session, just like when [running a single function](#). You can set different parameters for different instances of the *same* function if you added it multiple times to the View.

Running Functions within the View

You have two primary ways to execute the analyses:

1. **Run All Functions:**
 - Set all desired parameters in the right sidebar for all functions.

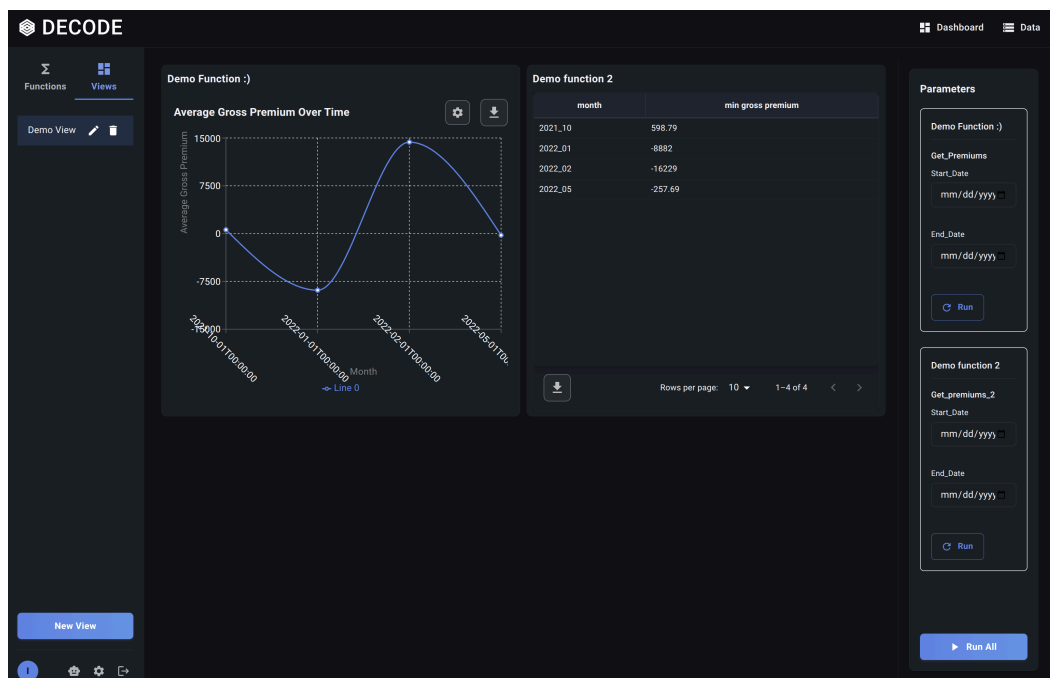
- Click the main **"Run All"** button (usually at the bottom of the Parameters Panel).
- Decode will execute *every* Data Function included in the View using the currently set parameters for each.
- Loading indicators will appear in each function's panel on the grid, and they will update with results as they complete. This is ideal for refreshing the entire dashboard.

2. Run a Single Function:

- Set the parameters specifically for the *one* function instance you want to run or refresh in the Parameters Panel.
- Find the corresponding function's section within the Parameters Panel and click its individual **"Run"** or **"Refresh"** button.
- Only that specific function's panel on the grid will show a loading indicator and update with new results. This is useful for iterating on one part of the dashboard without re-running everything.

Viewing Results

As functions complete, their respective panels on the grid will display the output (Table, Bar Chart, Line Chart) defined in their Visualization Node. You can now view and compare these results side-by-side within your custom layout.



Run a View

Views provide a dynamic way to interact with multiple related analyses, making it easier to spot trends, compare segments, and gain a comprehensive understanding from your data.

Finally, learn about [Managing Your Views](#).

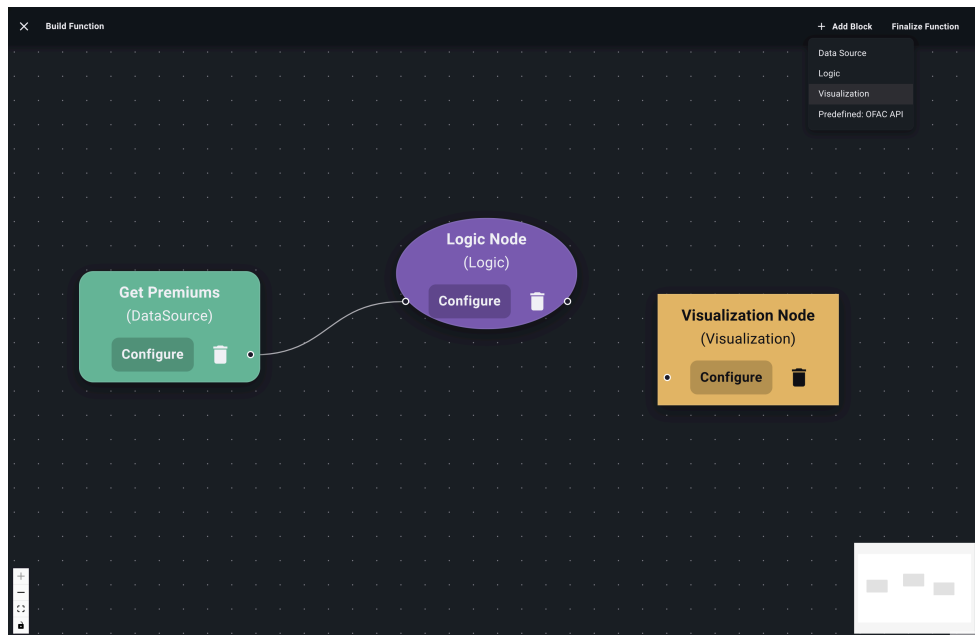
Step 4: Adding a Visualization Node

The **Visualization Node** is the final step in most Data Functions. Its purpose is to take the processed data from the preceding Logic or Predefined Node and display it in a chosen format: either a **Table**, a **Bar Chart**, or a **Line Chart**.

This node determines *how* the results of your function will be presented when you run it later.

Adding the Node

1. **Open the "Add Block" Menu:** In the Function Builder Dialog, click **"Add Block"**.
2. **Select "Visualization":** Choose **"Visualization"** from the menu.
3. A new Visualization node will appear on the canvas.



Add Visualization Node

4. **Connect Input Node:** Connect the output handle of the preceding Logic or Predefined Node to the input handle of the Visualization Node. *A Visualization Node must have an input connected before configuration.*

Configuring the Visualization Node

1. **Open Configuration:** Click the **"Configure"** button on the Visualization Node.
2. The **Visualization Block Configuration** panel or dialog will appear.
3. **Name (Optional but Recommended):** Give this output a title (e.g., "Loss Ratio Trend", "Premiums by State Table", "Claim Counts Over Time").
4. **Select Visualization Type:** Choose the desired output format from the dropdown:
 - **Table**
 - **Bar Chart**
 - **Line Chart**
 - **Pie Chart**
 - **Box and Whisker Chart**
 - **Scatter Chart**

Configuring a Table Visualization

- If you selected **"Table"**, no further configuration is needed. The node will simply display the data table it receives from the input node.
- Click **"Save"**.

Configuring Charts (AI Assistance)

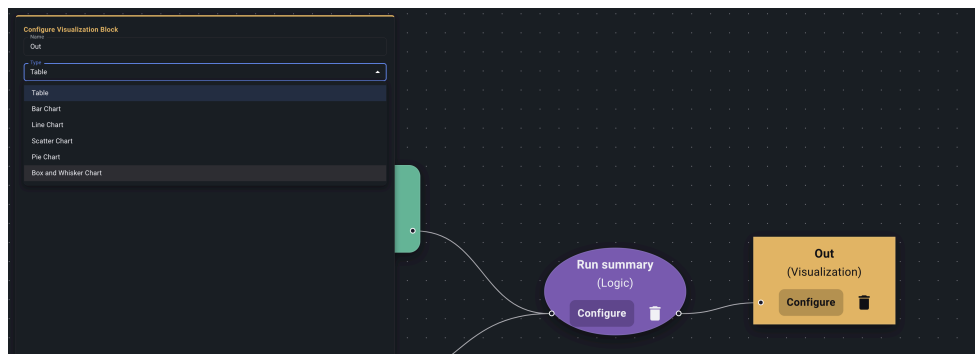
Creating effective charts often requires selecting the right columns for axes and values, which can sometimes be complex depending on the output structure of your Logic node. Decode uses AI assistance for this.

1. **Select Chart Type:** Choose one of the permitted chart types.
2. **Interact with AI:** Similar to the [Logic Node](#), a chat interface will appear.
 - **Initial Prompt:** Start by telling the AI what you want to plot. Be specific about which columns from the *input data* should be used for the X-axis, Y-axis, and potentially different series (for line charts).
 - **Examples:**
 - (For Bar Chart): "Create a bar chart showing the 'Total Net Premium' for each 'Wholesaler Name'."
 - (For Line Chart): "Plot the 'Loss Ratio' over time using the 'Report Month' column for the x-axis."

- (For Line Chart with multiple lines): "Plot 'Gross Premium' and 'Net Premium' as separate lines over time, using 'Month' for the x-axis."
- **Send Message:** Send your instructions to the AI.

3. AI Processing & Chart Preview:

- The AI interprets your request, generates the necessary configuration to map the input data columns to the chart structure, and potentially performs minor data restructuring if needed (like pivoting).
- A **Chart Preview** will be displayed within the configuration panel, showing how the chart will look based on the *sample data* from the input node.



Configure Visualization Node

4. Review and Iterate:

- Examine the Chart Preview. Is it displaying the data correctly? Are the axes labeled appropriately? Are the lines/bars representing the right things?
- If it's not right, **chat further with the AI**. Provide corrections or refinements.
- **Example Corrections:** "Use 'Region' on the x-axis instead." or "Label the Y-axis 'Total Premium (\$)'. " or "Make the bars blue."
- Continue iterating until the chart preview accurately reflects the visualization you need.

5. **Save:** When satisfied with the chart configuration achieved through the AI interaction, click **"Save"**.

Your Visualization Node is now configured to display the results of your Data Function in the chosen format. This is typically the final node in a function pipeline.

The last step in building the basic structure is [Connecting Nodes](#).

Running Data Functions

Once you have built and finalized a Data Function (status marked as 'Final'), you can run it anytime from the Dashboard Page to perform the analysis and view the results based on current data and specified parameters.

Selecting a Function to Run

1. Navigate to the **Dashboard Page**.
2. In the left sidebar, under the **"Functions"** tab, locate the list of your **Final** functions. (Draft functions cannot be run directly here; they must be finalized via the builder first).
3. Click on the name of the **Function** you want to run.

What Happens:

- The main content area updates to show the output area for the selected function. Initially, this will be empty or show previous results if any.
- The **Parameters Panel** appears on the right sidebar, displaying any variable filters you defined in the function's [Data Source Node\(s\)](#).

Using the Parameters Panel

The Parameters Panel is key to the reusability of your Data Functions. It allows you to change the inputs for filters you marked as **"variable"** during the function building process without needing to edit the function itself.

1. **Review Parameters:** The panel lists parameters grouped by the Data Source Node they belong to (if you named your nodes).
2. **Modify Values:**
 - For each parameter you want to change for this specific run, update its value in the corresponding input field.
 - Enter dates for date parameters, type text for text parameters, etc. If a parameter allows multiple values (like filtering by multiple regions), enter them separated by commas (e.g., `CA, NY, TX`).
 - Any default values you set when building the function will be pre-filled.

3. **Fixed Filters:** Filters that were *not* marked as variable during function creation will not appear in this panel; the function will always use the fixed value defined in the builder.

Running the Function

1. Once you have set the desired parameter values in the right sidebar, click the **"Run Function"** or **"Run All"** button.

2. **Processing:**

- Decode executes the function pipeline in the background.
- It retrieves data from the specified Tables, applying both the fixed filters defined in the function and the *current values* you just set in the Parameters Panel.
- It processes the data through any Logic or Predefined Nodes.
- A loading indicator will typically appear in the main output area.

3. **Viewing Results:**

- Once execution is complete, the main content area refreshes to display the output generated by the function's **Visualization Node(s)**.
- This could be a data table, a bar chart, a line chart, or potentially multiple outputs if your function includes them.
- If an error occurs during execution, an error message will be displayed instead.



Run Function

You can change the parameters in the sidebar and click "Run Function" again as many times as needed to explore different data slices or time periods using the same underlying analysis logic.

Next, learn about [Managing Your Functions](#) (Drafts, Finalizing, Editing, Deleting).

Managing Views


As you create more Views, Decode provides options to edit their layout or delete them when they are no longer needed.

Finding Your Views

Saved Views are listed under the **"Views"** tab in the left sidebar of the **Dashboard Page**.


Editing a View

If you need to change the layout, add/remove functions, or resize elements in a saved View:

1. Find the View you want to modify in the sidebar list.
2. Click the **Edit icon** () next to the View name.
3. The **View Builder Dialog** will open, pre-loaded with the selected View's current layout and function instances.
4. Make your desired changes:
 - Drag and drop functions from the "Available Functions" list to add them.
 - Rearrange existing function cards on the grid.
 - Resize function cards.
 - Remove function instances using the Close icon (X) on their card.
 - Update the View's Name or Description if needed.
5. Click **"Save View"** to save your changes.

Deleting a View

Warning: Deleting a View is permanent and cannot be undone. This only deletes the dashboard layout; the Data Functions included in it will *not* be deleted.

1. Find the View you want to delete in the sidebar list.
2. Click the **Delete icon** () next to the View name.
3. A confirmation dialog will appear.
4. Confirm the deletion by clicking **"Delete"**.

The View will be removed from the sidebar list.

Managing your Views ensures your Dashboard Page stays organized and relevant, allowing you to easily access the specific combinations of analyses you need most often.

Step 5: Connecting Nodes

Once you have added the necessary nodes ([Data Source](#), [Logic](#), [Predefined](#), [Visualization](#)) to your function's canvas, you need to connect them to define the flow of data.

Connections represent data moving from the output of one node to the input of the next.

How to Connect Nodes

1. **Identify Handles:** Hover your mouse over a node. You will see small circles or connection points, called **handles**.
 - **Source Handles:** Typically located on the **right** side of a node. Data flows *out* from these handles. Data Source, Logic, and Predefined nodes usually have source handles.
 - **Target Handles:** Typically located on the **left** side of a node. Data flows *into* these handles. Logic, Predefined, and Visualization nodes usually have target handles.
2. **Click and Drag:**
 - Click and hold your mouse button on a **Source Handle** of the node you want data to flow *from*.
 - Drag your mouse pointer towards the **Target Handle** of the node you want data to flow *to*.
 - An arrow (edge) will appear, following your cursor.
3. **Release:** Release the mouse button when your cursor is over the desired Target Handle.

Result: An edge (arrow) will be drawn between the two nodes, indicating the direction of data flow.

Connection Rules & Tips

- **Direction:** Data flows from a Source handle to a Target handle. You generally cannot connect Source-to-Source or Target-to-Target.
- **Input Requirements:** Logic, Predefined, and Visualization nodes usually *must* have at least one incoming connection before they can be configured or function correctly.
- **Multiple Inputs/Outputs (Less Common):** While most nodes have one input and one output, some advanced scenarios might involve nodes with multiple handles. The connection principle remains the same.
- **Deleting Connections:** To remove a connection you must delete one of the nodes and re-add it.

Connecting your nodes correctly defines the sequence of operations in your Data Function, ensuring data is retrieved, processed, and visualized in the order you intend.

With your nodes added and connected, you are ready for [Running Your Function](#).