data.all - User Guide

v2.7.0

None

None

Table of contents

1. Introduction

This section defines what is data.all, what is the challenge that it is trying to overcome and the value that it can bring to your teams.

1.1 What is data.all?

A modern data workspace that makes collaboration among diverse users (like business, analysts and engineers) easier, increasing efficiency and agility in data projects ✨

1.2 Why did we built data.all?

Data teams can be diverse: analysts, scientists, engineers, business users. Diverse people, with diverse tools and skillsets — diverse "DNAs". All leading to chaos and resulting in titanic efforts spent in **Collaboration Overhead**.

Using data.all, any line of business within an organization can create their own isolated data lake, produce, consume and share data within and across business units, worldwide. By simplifying data discovery, data access management while letting more builders use AWS vast portfolio of data and analytics services, data.all helps more data teams discover relevant data and let them use the power of the AWS cloud to create data driven applications faster.

1.3 How can data.all help data teams?

Teams can easily DISCOVER AND UNDERSTAND data 💫

data.all makes all your datasets easily discoverable! No more Slack messages saying "Where's that dataset?" or long email threads for approvals. With data.all, you can simply browse the data catalog.

Key Capabilities: Discovery and Search, Data Preview & Worksheets and Notebooks

Teams can easily SHARE AND COLLABORATE with data 🤝

Data practitioners spend 30-50% of their time finding and understanding data. data.all cuts that time by 95%. Your data team will be shipping 2-3 times more projects in no time.

Key Capabilities: Data Profiling & Data Sharing and Subscriptions

Teams don't have to worry about SECURING their data 🛡️

Don't lose sleep trying to figure out if your sensitive data is secure. Build ecosystems of trust, make your team happy, and let data.all manage governance and security behind the scenes.

Key Capabilities: Granular Access Control

2. Main components

This section introduces the main components of data.all which are divided in 3 groups. This is an overview, for more details please refer to their specific sections.

- **Administrate**: used by team and data lake administrators to organise and manage teams and users inside data.all
- **Discover**: used by all users to contribute with data, search for data and share data.
- **Play**: once data is in data.all, all users can use these tools to work with data.

2.1 Administrate

2.1.1 Organizations

Organizations are high level constructs where business units can collaborate across different AWS accounts at once. An organization includes environments (see below). Organizations are abstractions, they **don't** contain AWS resources, consequently there is no CloudFormation stack associated with them.

Organizations usually correspond to whole organizations, organization divisions or a separated geographical region within an organization.

2.1.2 Environments

An environment is a workplace where a team can bring, process, analyze data and build data driven applications. This workspace is mapped to an AWS account in one region. It is possible to have more than one environment in the same AWS Account, however we recommend to stick to one environment - one account.

An environment usually corresponds to a business unit or a department. Inside an environment we add teams and assign them different levels of permissions.

2.1.3 Teams

A team corresponds to an IdP group that has been onboarded to data.all. A special case for the administration of data.all is the **Tenant**, an IdP group with high level application (tenant) permissions. As with IdP groups, users can belong to multiple teams.

Teams corresponds to real teams.

> **but really, what are teams?**
>
> Data in data.all is isolated at team level, meaning that all members of a team can access all team's datasets. Thus, a team is any group of users that can access the team's datasets. We can have bigger teams with generic data and project-based teams owning data that requires more restrictive access to only members of the project.

2.2 Discover

2.2.1 Datasets

A dataset is a representation of multiple AWS resources that helps users store data. When data owners create a **S3 dataset** on data.all the following resources are created:

- Amazon S3 Bucket to store the data on AWS.
- AWS KMS key to encrypt the data on AWS.
- AWS IAM role that gives access to the data on Amazon S3.
- AWS Glue database that is the representation of the structured data on AWS.

Inside the dataset we can store structured data as tables or unstructured data in folders.

Alternatively, when data owners import a **Redshift dataset** on data.all a subset of the tables can be imported from a specific Redshift database schema.

2.2.2 Catalog

data.all centralized Catalog is an inventory of datasets, tables, folders and dashboards. It contains metadata for each of the mentioned data assets and thanks to its search capabilities, users can filter based on type of data, type of asset, tags, region and on glossary terms.

We use the Catalog to search and discover data

2.2.3 Glossaries

A Glossary is a list of terms, organized in a way to help users understand the context of their datasets. For example, terms like "cost", "revenue", etc, can be used to group and search all financial datasets.

Glossaries are used to add meaning to data assets metadata facilitating and enhancing Catalog searching

2.2.4 Shares

A Share is an access request to a data asset. Users search and discover data in the catalog and for those data assets that belong to other teams, users can create a Share on behalf of a team (remember, data access: at team level!!). Then, the owners of the asset can accept or reject the share.

We use Shares to collaborate and share data with other teams.

2.3 Play

2.3.1 Worksheets

Worksheets are AWS Athena sessions that allow us to query our datasets as if we were in the AWS Athena Query editor console.

2.3.2 Notebooks

Data practitioners can experiment machine learning algorithms spinning up Jupyter notebook with access to all your datasets. data.all leverages Amazon SageMaker instance to access Jupyter notebooks.

2.3.3 ML Studio

With ML Studio Notebooks we can add users to our SageMaker domain and open Amazon SageMaker Studio

2.3.4 Dashboards

In the Dashboard window we can start Quicksight sessions, create visual analysis and dashboards.

2.3.5 Omics

Provides the capability to view and instantiate HealthOmics Ready2Run workflows against data.all datasets and save omics data from workflow output.

3. Administrate

3.1 Tenant and Organizations

data.all manages teams' permissions at four levels:

1. Tenant team
2. Organization
3. Environment (next section)
4. Teams (next section)

3.1.1 Tenant

data.all has a super user's team which is a group from your IdP that has the right to manage high level application (tenant) permissions for all IdP groups integrated with data.all.

This super user's team maps to a group from your IdP that's by default named "**DAAdministrators**", any user member of this group will be able to:

- create organizations
- manage tenant permissions on onboarded teams (IdP groups) as shown below.

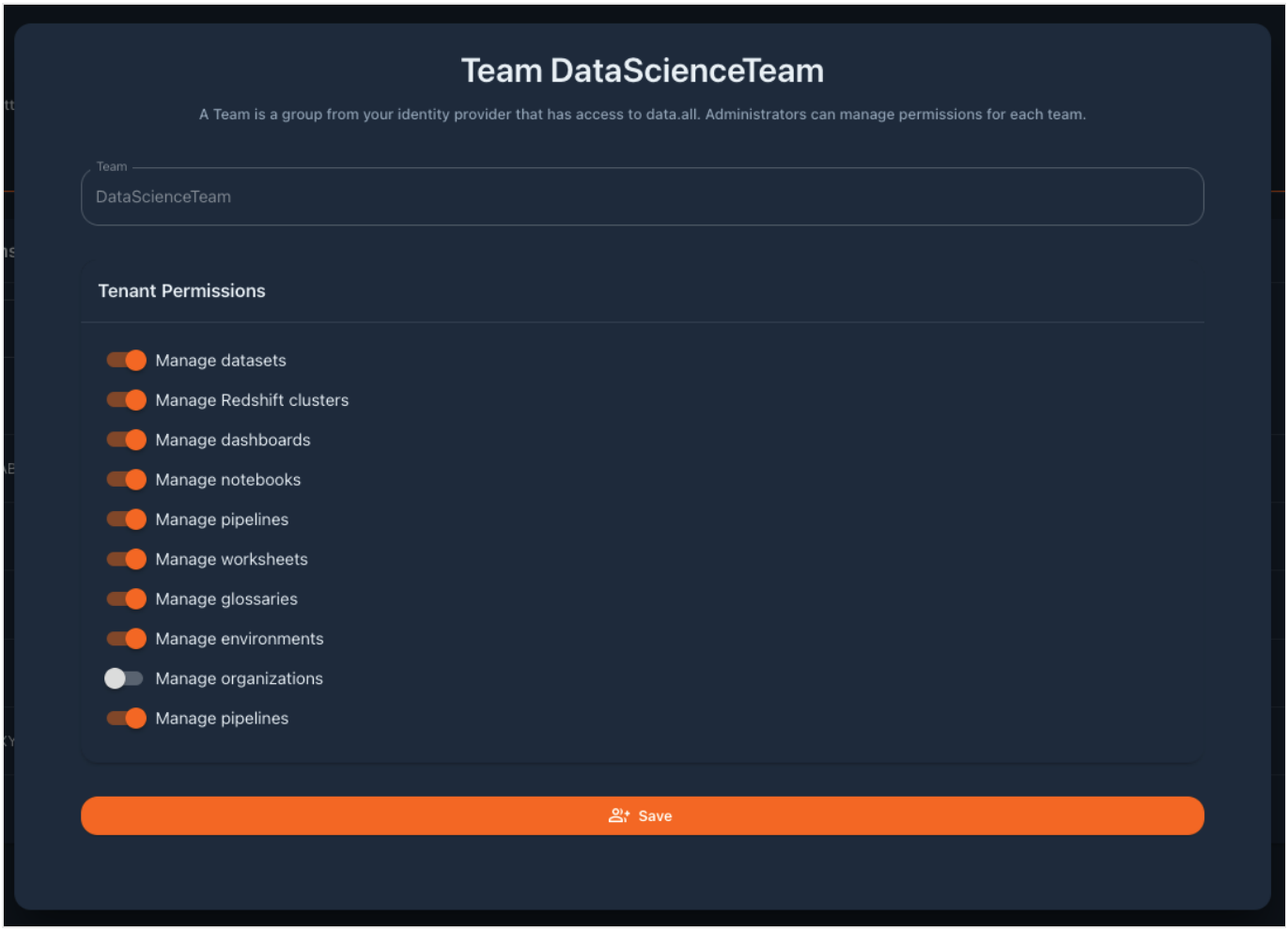
Manage tenant permissions

As a user part of "**DAAdministrators**" on your IdP you can access the settings menu from the profile icon.

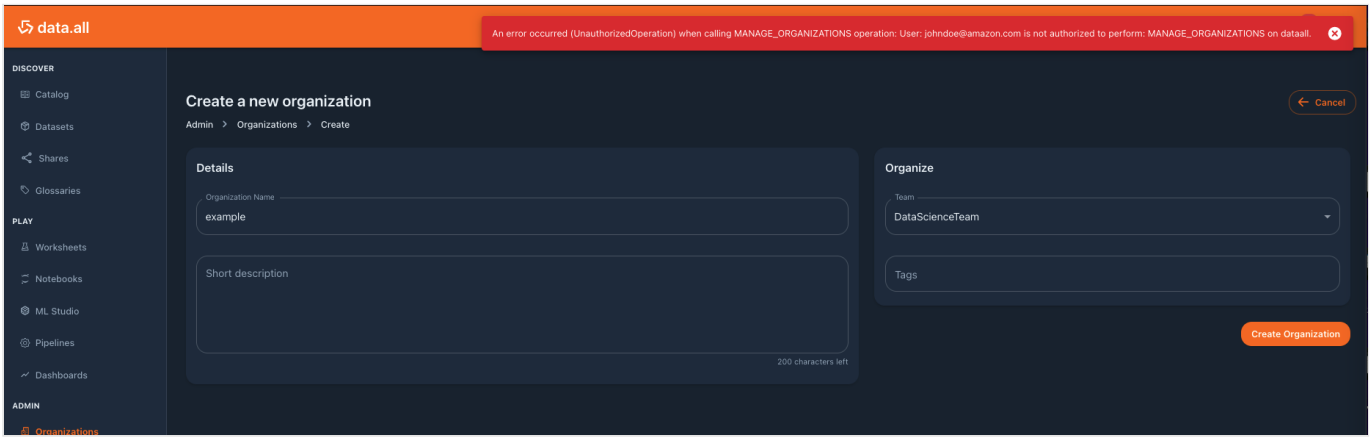
For example, Maria Garcia is not part of "**DAAdministrators**", therefore she sees nothing

On the other hand, Tenant user is part of this group and can navigate to **Admin settings**

In Admin Settings, the Tenant user can manage tenant permissions. In the following picture, the user is NOT granting the DataScienceTeam that John belongs to permissions to create an organization.



If the tenant revokes the permission of a team to manage an object, that team won't be able to perform any action on that particular object. For the given example, assuming that John only belongs to the DataScienceTeam, he is not able to create organizations:



3.1.2 Organizations

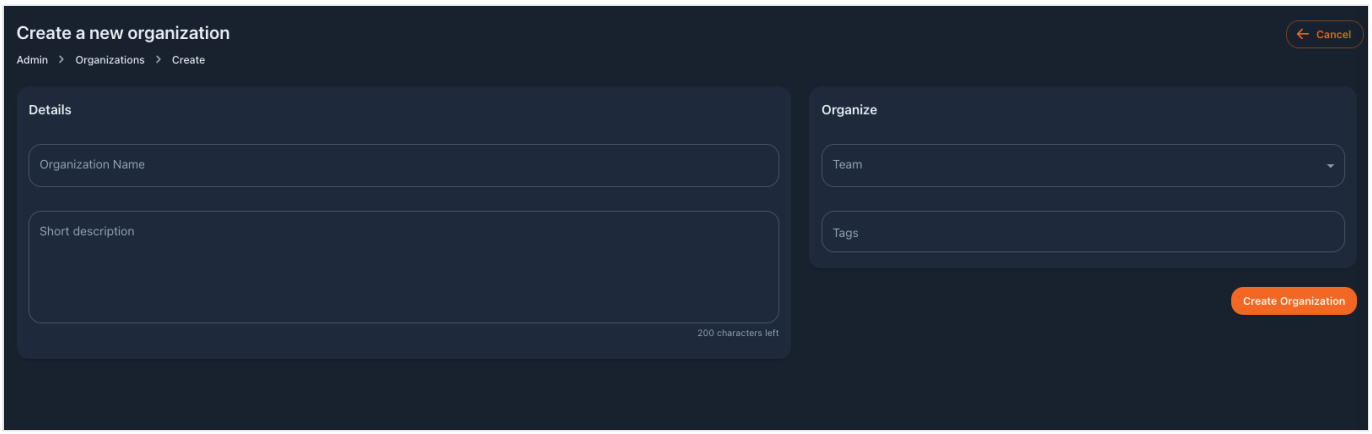
Organizations are high level constructs where business units can collaborate across many different AWS accounts at once. An organization includes environments and teams (see next section). Organizations are abstractions, they **don't** contain AWS resources, consequently there is no CloudFormation stack associated with them.

Organizations usually correspond to whole organizations, organization divisions or a separated geographical region within an organization.

Create an organization

> **Organization permissions**
>
> Any user can create an organization as long as he or she belongs to a group with tenant permission "Manage Organizations" (see previous chapter, "Manage tenant permissions").

To create an organization, on the left pane select **Organization**, click **Create** and complete the following form.



Field	Description	Required	Editable	Example
Organization name	Name of the organization	Yes	Yes	AnyCompany EMEA
Short description	Short description about the organization	No	Yes	AnyCompany EMEA region
Team	Name of the team managing the organization	Yes	No	EMEAAdmin
Tags	List of tags	No	Yes	fin,rnd,mark,sales

The next step to onboard your IdP groups is to link an environment and add teams, check Link an environment and Add a team to an environment

Edit and update an organization

On the organisation window we can check the organization metadata, as well as the environments and teams that belong to this organisation (we will come back to this in Environments and teams).

To edit the metadata of the organisation, click in **Edit** and update the information. Name, description and tags are editable, however the organisation team cannot be updated.

Delete an organization

> **Warning**
>
> Make sure that you delete the organisation environments before deleting the organisation. Otherwise, orphan environments might run into conflicts.

To archive an organisation, click on the **Archive** button next to the Edit button. A window with the previous warning will appear. If you want to go ahead and delete the organization, type permantly archive in the box and submit.

3.2 Environments and Teams

An environment is a **workplace** where a team can bring, process, analyze data and build data driven applications. Environments comprise AWS resources, thus when we create an environment, we deploy a CDK/CloudFormation stack to an AWS account and region. In other words, **an environment is mapped to an AWS account in one region, where users store data and work with data.**

> **One AWS account, One environment**
>
> To ensure correct data access and AWS resources isolation, onboard one environment in each AWS account. **We strongly discourage users to use the same AWS account for multiple environments**.

3.2.1 AWS account Pre-requisites

data.all does not create AWS accounts. You need to provide an AWS account and complete the following bootstraping steps. Only the first step, CDK bootstrap, is mandatory; the rest are needed depending on your deployment configuration or on the features enabled in the environment.

1. CDK Bootstrap

data.all uses AWS CDK to deploy and manage resources on your AWS account. AWS CDK requires some resources to exist on the AWS account, and provides a command called `bootstrap` to deploy these specific resources in a particular AWS region.

In this step we establish a trust relationship between the data.all infrastructure account and the accounts to be linked as environments. data.all codebase and CI/CD resources are in the data.all **tooling account**, and all the application resources used by the platform are located in a **infrastructure account**. From the infrastructure account we will deploy environments and other resources inside each of our business accounts. We are granting permissions to the infrastructure account by setting the `--trust` parameter in the cdk bootstrap command.

To boostrap the AWS account using AWS CDK, you need the following (which are already fulfilled if you open AWS CloudShell from the environment account).

1. to have AWS credentials configured in ~/.aws/credentials or as environment variables.
2. to install cdk: `npm install -g aws-cdk`

Then, you can copy/paste the following command from the UI and run from your local machine or CloudShell:

```
cdk bootstrap --trust DATA.ALL_AWS_ACCOUNT_NUMBER  -c @aws-cdk/core:newStyleStackSynthesis=true --cloudformation-execution-policies arn:aws:iam::aws:policy/AdministratorAccess aws://
YOUR_ENVIRONMENT_AWS_ACCOUNT_NUMBER/ENVIRONMENT_REGION
```

> **Which account should I put in the command?**
>
> Let's check with an example: the **tooling account** is 111111111111 and data.all was deployed to the **infrastructure account** = 222222222222. Now we want to onboard a **business account** = 333333333333 in region eu-west-1. Then the cdk bootstrap command will look like: `bash`
> `cdk bootstrap --trust 222222222222  -c @aws-cdk/core:newStyleStackSynthesis=true --cloudformation-execution-policies arn:aws:iam::aws:policy/AdministratorAccess aws://333333333333/eu-west-1`

> **After deleting an environment it is strongly recommended to untrust data.all infrastructure account. Read more here**

RESTRICTED CDK EXECUTION ROLE

In the above command we define the `--cloudformation-execution-policies` to use the AdministratorAccess policy `arn:aws:iam::aws:policy/AdministratorAccess`. This is the default policy that CDK uses to deploy resources, nevertheless it is possible to restrict it to any IAM policy created in the account.

A more restricted policy named `DataAllCustomCDKPolicyREGION` is provided and directly downloadable from the UI. This more restrictive policy can be optionally passed in to the parameter `--cloudformation-execution-policies` instead of `arn:aws:iam::aws:policy/AdministratorAccess` for the CDK Execution role.

```
aws cloudformation --region REGION create-stack --stack-name DataAllCustomCDKExecPolicyStack --template-body file://cdkExecPolicy.yaml --parameters
ParameterKey=EnvironmentResourcePrefix,ParameterValue=dataall --capabilities CAPABILITY_NAMED_IAM && aws cloudformation wait stack-create-complete --stack-name DataAllCustomCDKExecPolicyStack
--region REGION && cdk bootstrap --trust 225091619433 -c @aws-cdk/core:newStyleStackSynthesis=true --cloudformation-execution-policies arn:aws:iam::ACCOUNT_ID:policy/
DataAllCustomCDKPolicyREGION aws://ACCOUNT_ID/REGION
```

ENVIRONMENTS IN MULTIPLE REGIONS

v2.4.0 allows the creation of multiple environments in the same AWS account and in multiple regions. We need to bootstrap every region that will host an environment.

> **Regional CDK Execution Policy**
>
> Every CDK execution role requires its own `DataAllCustomCDKPolicyREGION` IAM policy. If you are using restricted CDK execution roles you need a different `DataAllCustomCDKExecPolicyStack` for each region used.

2. (For manual) Pivot role

data.all assumes a certain IAM role to be able to call AWS SDK APIs on your account. The Pivot Role is a super role in the environment account and thus, it is protected to be assumed only by the data.all central account using an external Id.

Since release V1.5.0, the Pivot Role can be created as part of the environment CDK stack, given that the trust between data.all and the environment account is already explicitly granted in the bootstraping of the account. To enable the creation of Pivot Roles as part of the environment stack, the `cdk.json` parameter `enable_pivot_role_auto_create` needs to be set to `true`. When an environment is linked to data.all a nested stack creates a role called **dataallPivotRole-cdk**.

For versions prior to V1.5.0 or if `enable_pivot_role_auto_create` is `false` the Pivot Role needs to be created manually. In this case, the AWS CloudFormation stack of the role can be downloaded from data.all environment creation form. (Navigate to an organization and click on link an environment to see this form). Fill the CloudFormation stack with the parameters available in data.all UI to create the role named **dataallPivotRole**.

> **Upgrading from manual to cdk-created Pivot Role**
>
> If you have existing environments that were linked to data.all using a manually created Pivot Role you can still benefit from V1.5.0 `enable_pivot_role_auto_create` feature. You just need to update that parameter in the `cdk.json` configuration of your deployment. Once the CICD pipeline has completed: new linked environments will contain the nested cdk-pivotRole stack (no actions needed) and existing environments can be updated by:
> a) manually, by clicking on "update stack" in the environment --> stack tab
> b) automatically, wait for the `stack-updater` ECS task that runs daily overnight
> c) automatically, set the added `enable_update_dataall_stacks_in_cicd_pipeline` parameter to `true` in the `cdk.json` config file. The `stack-updater` ECS task will be triggered from the CICD pipeline

3. (For Dashboards) Subscribe to Amazon Quicksight

This is an optional step. To link environments with **Dashboards enabled** , you will also need a running Amazon QuickSight subscription on the bootstraped account. If you have not subscribed to Quicksight before, go to your AWS account and choose the Enterprise option as show below:

Your AWS Account is not signed up for QuickSight. Would you like to sign up now?

AWS Account [redacted]

Sign up for QuickSight

To access QuickSight with a different account, log in again.

Create your QuickSight account

Enterprise | Standard

Edition	Enterprise	Enterprise + Q Learn more
Team trial for 30 days (4 authors)*	FREE	FREE
Author per month (yearly)**	$18	$28
Author per month (monthly)**	$24	$34
Readers (pay-per-Session)	$0.30 / session (max $5)****	$0.30 / session (max $10)****
Additional SPICE per month	$0.38 per GB	$0.38 per GB
QuickSight Q regional fee	N/A	$250 / mo / region
Personalized Q authoring workshop	N/A	Starting from $199
Natural language query with QuickSight Q	N/A	INCLUDED
Single Sign On with SAML or OpenID Connect	✓	✓
Connect to spreadsheets, databases & business apps	✓	✓
Access data in Private VPCs	✓	✓
Row-level security for dashboards	✓	✓
Secure data encryption at rest	✓	✓
Connect to your Active Directory	✓	✓
Use Active Directory groups***	✓	✓
Send email reports	✓	✓
Embed QuickSight	✓	✓
Capacity-based pricing	✓	✓
Supported regions	Learn more	Learn more

4. (For ML Studio) Specifying a VPC or using default

If ML Studio is enabled, data.all will create a new SageMaker Studio domain in your AWS Account and use the domain later on to create ML Studio profiles.

Prior to V1.5.0 data.all always used the default VPC to create a new SageMaker domain. The default VPC had then to be customized to fulfill the networking requirements specified in the Sagemaker documentation for VPCOnly domains.

In V1.5.0 we introduce the creation of a suitable VPC for SageMaker as part of the environment stack. However, it is not possible to edit the VPC used by a SageMaker Studio domain, it requires deletion and re-creation. To allow backwards compatibility and not delete the pre-existing domains, in V1.5.0 the default behavior is still to use the default VPC.

In V2.2.0, we introduced the ability to select your own VPC ID and Subnet IDs to deploy the VPC-Only Sagemaker Studio domain to.

Data.all will follow the following rules to establish which VPC to use for Sagemaker Studio domain creation:

- If MLStudio enabled with VPC and subnet IDs specified
- Use the specified VPC and subnet IDs
- If MLStudio enabled with no VPC/subnet IDs specified
- default VPC exists --> Uses default VPC and all subnets available
- default VPC does not exist --> Creates a new VPC and uses with private subnets

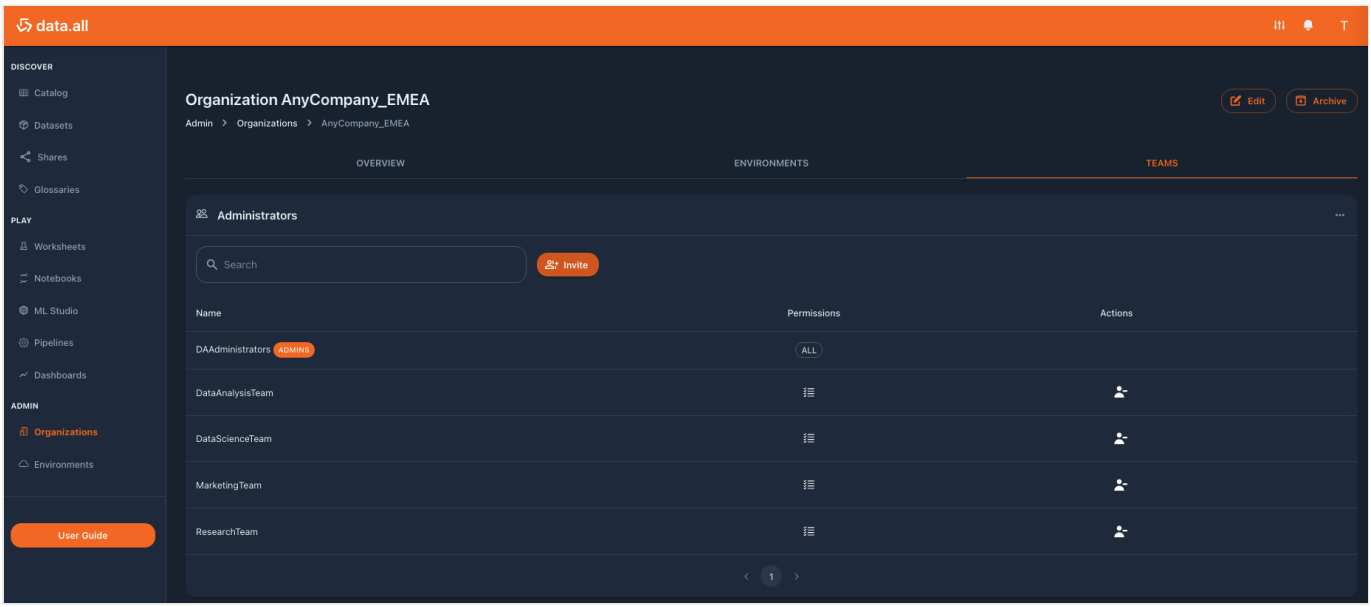
Pre-existing environments from older versions of data.all will have their Sagemaker Studio domain remain unchanged if already enabled. Users can get a better understanding of what VPC configuration is being used by navigating to the environment --> MLStudio Tab in the data.all UI once the environment stack is created.

3.2.2 NEW Link an environment

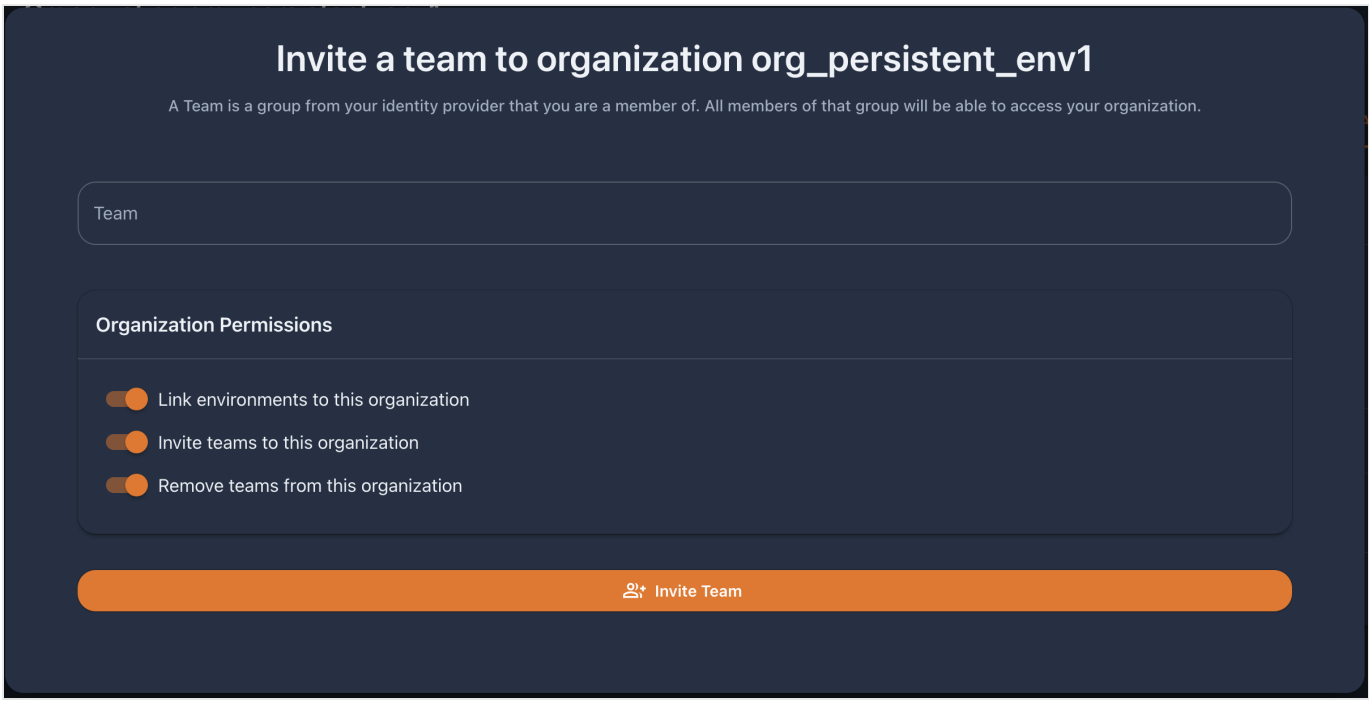
Necessary permissions

> **Environment permissions**
>
> Only organization Administrator teams can link environments to the Organization. The Organization creator team is the by default Organization Administrator team, but users of this group can now invite other teams and grant them permission to manage organization teams, and link environment to the organization.

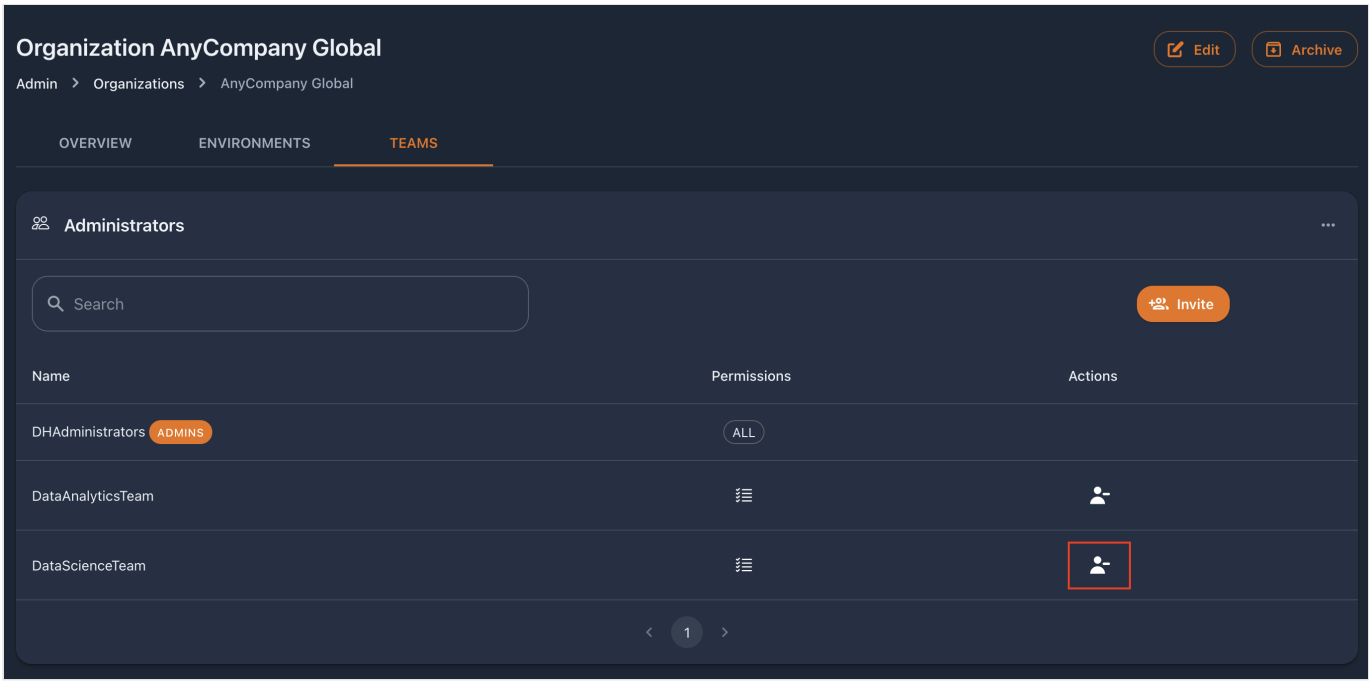
Managing organization teams can be done through the UI or APIs. From the UI, navigate to your organizations and click on the **Teams** tab.



Invite button opens a dialog that gives the organization creators the possibility to invite one of the IdP groups they belong to, which will appear in a dropdown when we click on **Teams**. They can also invite an IdP group that they don't belong to, as long as they type the exact group name (**case sensitive**):

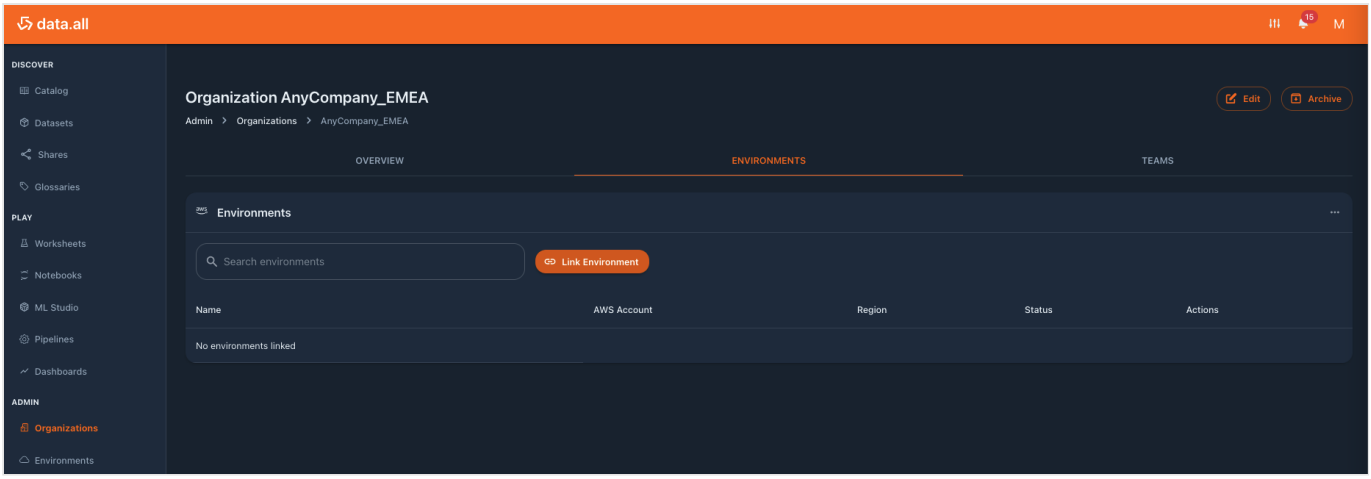


You can check the Organization administrators teams in the Organization's **Teams** tabs and remove a team if necessary on the icon in the Actions column.



Link environment

Once the AWS account/region is bootstraped and we have permission to link an environment to an organization, let's go! Navigate to your organization, click on the **Link Environment** button, and fill the environment creation form:



Field	Description	Required	Editable	Example
Environment name	Name of the environment	Yes	Yes	Finance
Short description	Short description about the environment	No	Yes	Finance department teams
Account number	AWS bootstraped account maped to the environment	Yes	No	111111111111
Region	AWS region	Yes	No	Europe (Ireland)
IAM Role ARN	Alternative name of the environment IAM role	No	No	anotherRoleName
Resources prefix	Prefix for all AWS resources created in this environment. Only (^[a-z-]*$)	Yes	Yes	fin
Team	Name of the group initially assigned to this environment	Yes	No	FinancesAdmin
Tags	Tags that can later be used in the Catalog	Yes	Yes	finance, test
ML Studio VPC ID	VPC to host the environment sagemaker studio domain (if mlstudio is enabled) instead than the default VPC or the VPC created by data.all	No	No	vpc-.....
ML Studio Subnet ID(s)	Subnet(s) to host the environment sagemaker studio domain (if mlstudio is enabled) instead than the default subnets or the subnets created by data.all	No	No	subnet-....

Features Management

An environment is defined as a workspace and in this workspace we can flexibly activate or deactivate different features, adapting the workspace to the teams' needs. If you want to use Dashboards, you need to complete the optional third step explained in the previous chapter "Bootstrap your AWS account".

This is not set in stone!

Don't worry if you change your mind, features are editable. You can always update the environment to enable or disable a feature.

Click on Save, the new Environment should be displayed in the Environments section of the left side pane.

3.2.3 Manage your Environment

Go to the environment you want to check. You can find your environment in the Environments list clicking on the left side pane or by navigating to the environment organization. There are several tabs just below the environment name:

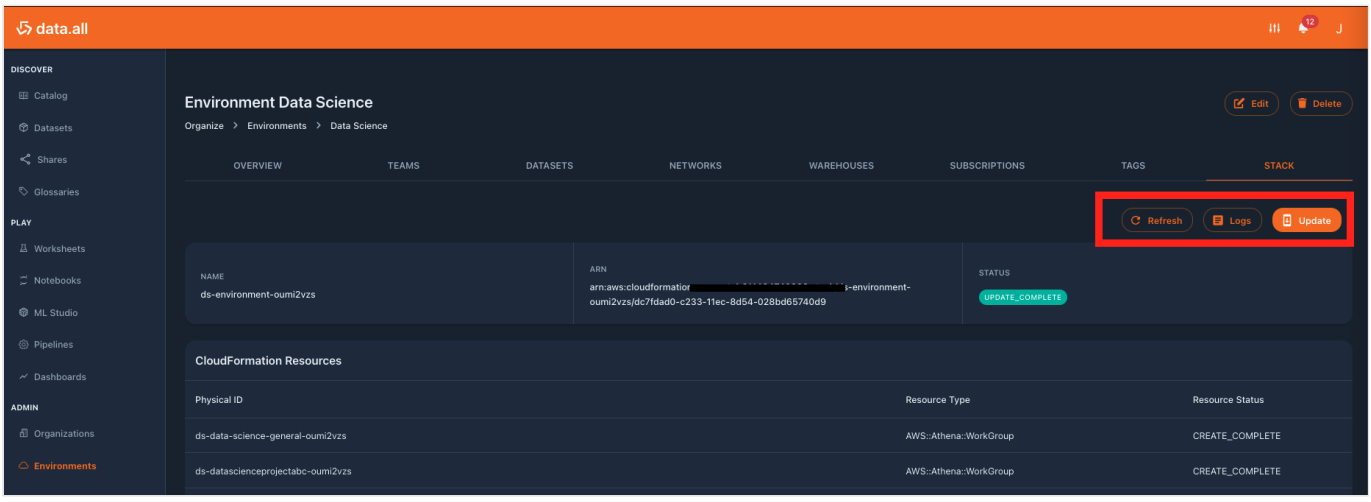
- Overview: summary of environment information and AWS console and credential access.
- Teams: list of all teams onboarded to this environment.
- Datasets: list of all datasets owned and shared with for this environment
- MLStudio: summary of Sagemaker Studio domain configuration (if enabled)
- Networks: VPCs created and owned by the environment
- Subscriptions: SNS topic subscriptions enabled or disabled in the environment
- Tags: editable key-value tags
- Stack: CloudFormation stack details and logs

Environment access

If **none** of the teams you belong to (IdP groups) has been onboarded to the environment, you won't be able to see the environment in the environments menu or in the organization environments list. **Check the "Manage teams" section**

Check CloudFormation stack

After linking an environment we can check the deployment of AWS resources in CloudFormation, click on the environment and then on the **Stack** tab. Right after linking an environment you should find something like the below picture.



After some minutes its status should go from "PENDING" to "CREATE_COMPLETE" and we will be able to look up the AWS resources created as part of the environment CloudFormation stack. Moreover, we can manually trigger the update in case of change sets of the CloudFormation stack with the **Update** button.

Pro Tip

If something in the creation or update of an environment fails, we can directly check the logs by clicking the logs button. No need to navigate to the AWS console to find your logs!

After being processed (not in `PENDING`), the status of the CloudFormation stack is directly read from CloudFormation.

Edit and update an environment

Find your environment in the Environments list or by navigating to the corresponding organization. Once in your selected environment, click on **Edit** in the top-right corner of the window and make all the changes you want.

Finally, click on **Save** at the bottom-right side of the page to update the environment.

Automatically updates the CloudFormation stack

Clicking on Save will update the environment metadata as well as the CloudFormation stack on the AWS account

Delete an environment

In the chosen environment, next to the Edit button, click on the **Delete** button.

orphan data.all resources

A message like this one: "Remove all environment related objects before proceeding with the deletion!" appears in the delete display. Don't ignore it! Before deleting an environment, clean it up: delete its datasets and other resources.

Untrust data.all infrastructure account

A message like this one: "After removal users must untrust the data.all account manually from env account CDKToolkit stack!" appears in the delete display. Don't ignore it! When you boostrapped the environment account you explicitly "trusted" (using the `--trust <account id>` flag) the infrastructure account to make deployments to your account.

- If you don't want to make CDK deployments (not necesserily related to data.all) to that account/region you can completely remove the CDKToolkit stack from CFN
- If you want to continue using the account/region for other CDK deployments you must untrust the data.all account by rerunning `cdk bootstrap --trust <TRUSTED_NON_DATAALL_ACC1> --trust <TRUSTED_NON_DATAALL_ACC2> ...`

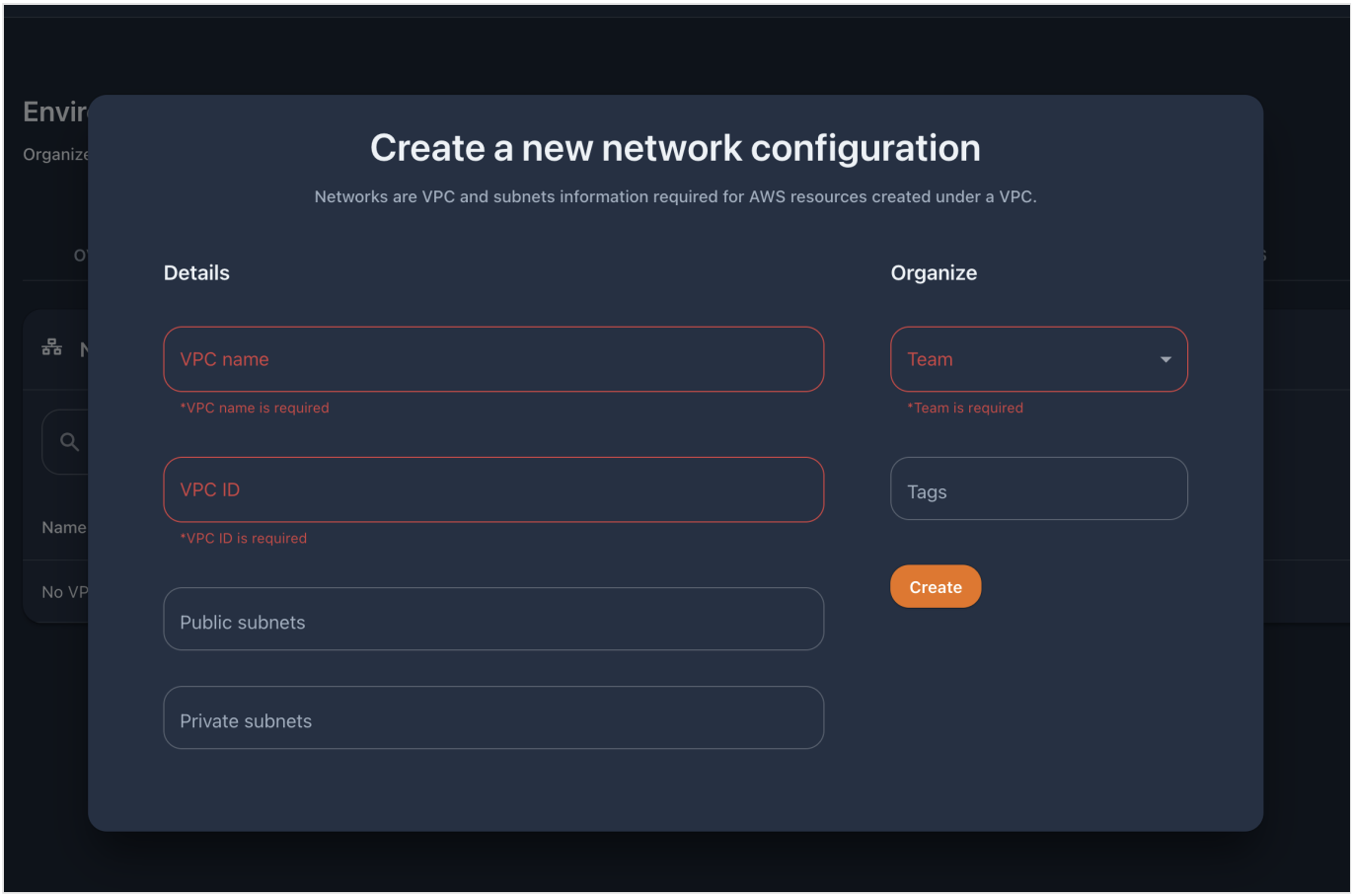
Note that we can keep the environment CloudFormation stack. What is this for? This is useful in case you want to keep using the environment resources (IAM roles, etc) created by data.all but outside of data.all

Create networks

Networks are pre-existing VPCs that are onboarded to data.all and belonging to an environment and team. To create a network, click in the **Networks** tab in the environment window, then click on **Add** and finally fill the following form.

Using Networks

After onboarding your network(s) in data.all, users can easily select the VPC and Subnet information of that network to seamlessly deploy new resources in data.all that require VPC configurations, such as data.all Notebooks. For example, if a User wants to create a notebook in their environment after onboarding a network, the VPC and Subnet ID fields in the create notebook form on data.all will auto-populate with the VPC and subnet information for the user to easily to select (rather than navigating to and from the AWS Console)!

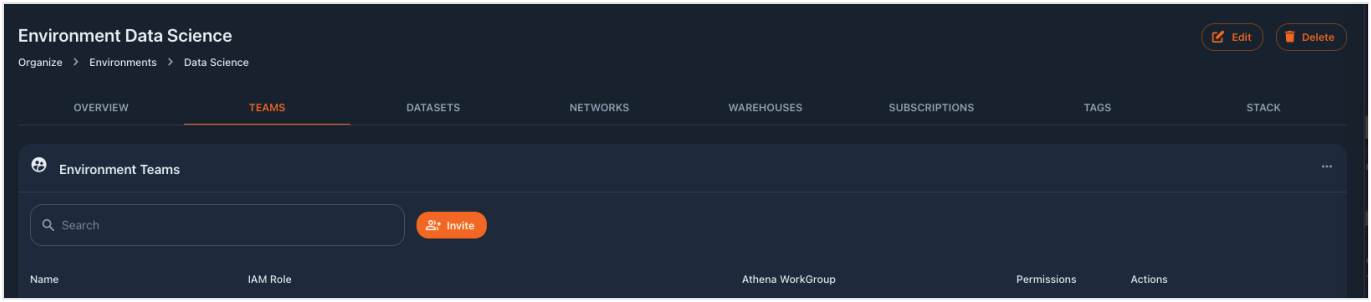


Create Key-value tags

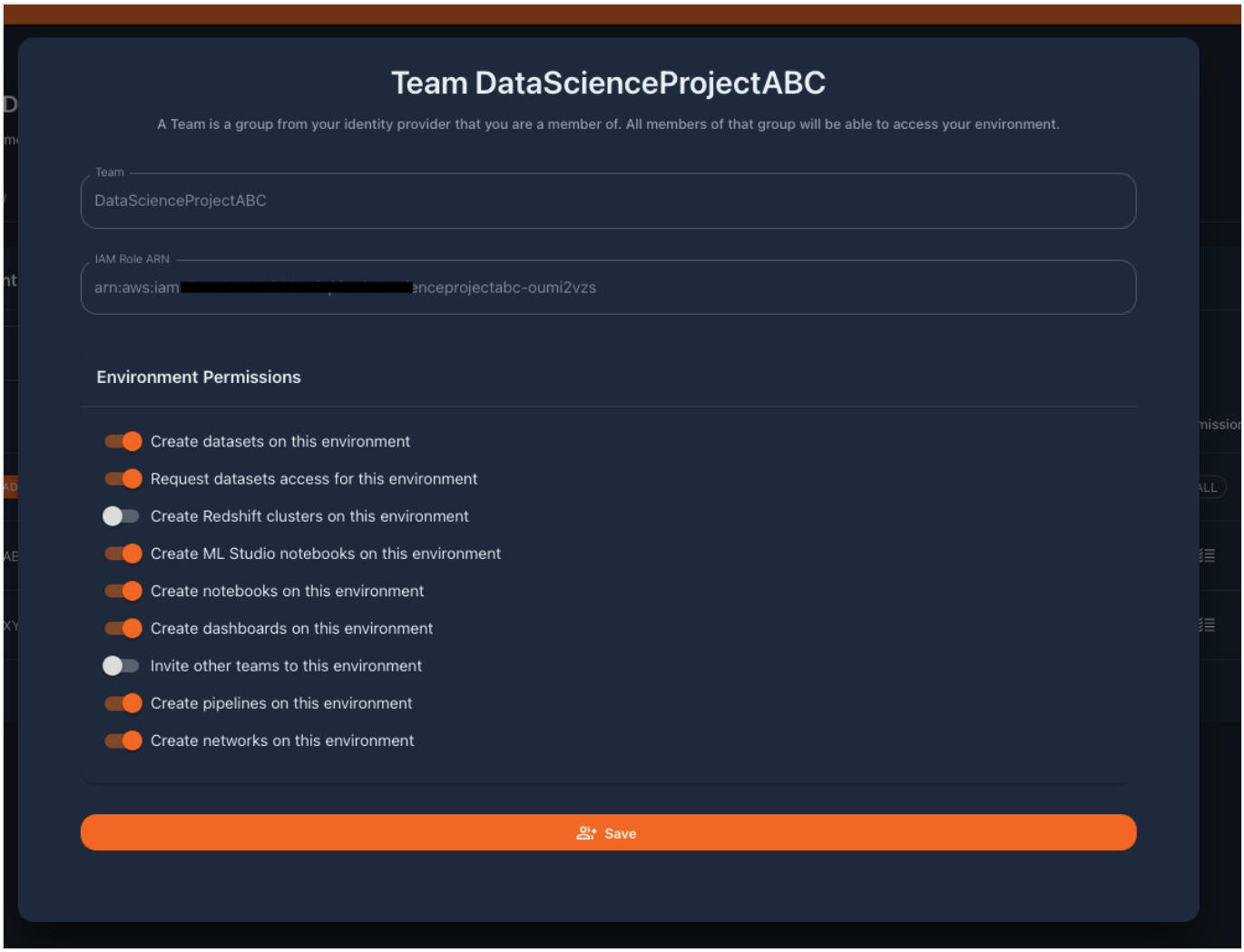
In the **Tags** tab of the environment window, we can create key-value tags. These tags are not data.all tags that are used to tag datasets and find them in the catalog. In this case we are creating AWS tags as part of the environment CloudFormation stack. There are multiple tagging strategies as explained in the documentation.

3.2.4 Manage Teams

Environment creators have all permissions on the environment, and can invite other teams to the onboarded environment. To add an IdP group to an environment, navigate to the **Teams** tab of the environment and click on the **Invite** button.



A display will allow you to customize the AWS permissions that the onboarded group will have, adapting to different types of users (data scientists, data engineers, data analysts, management). The customizable permissions can be enabled or disabled as appears in the following picture.

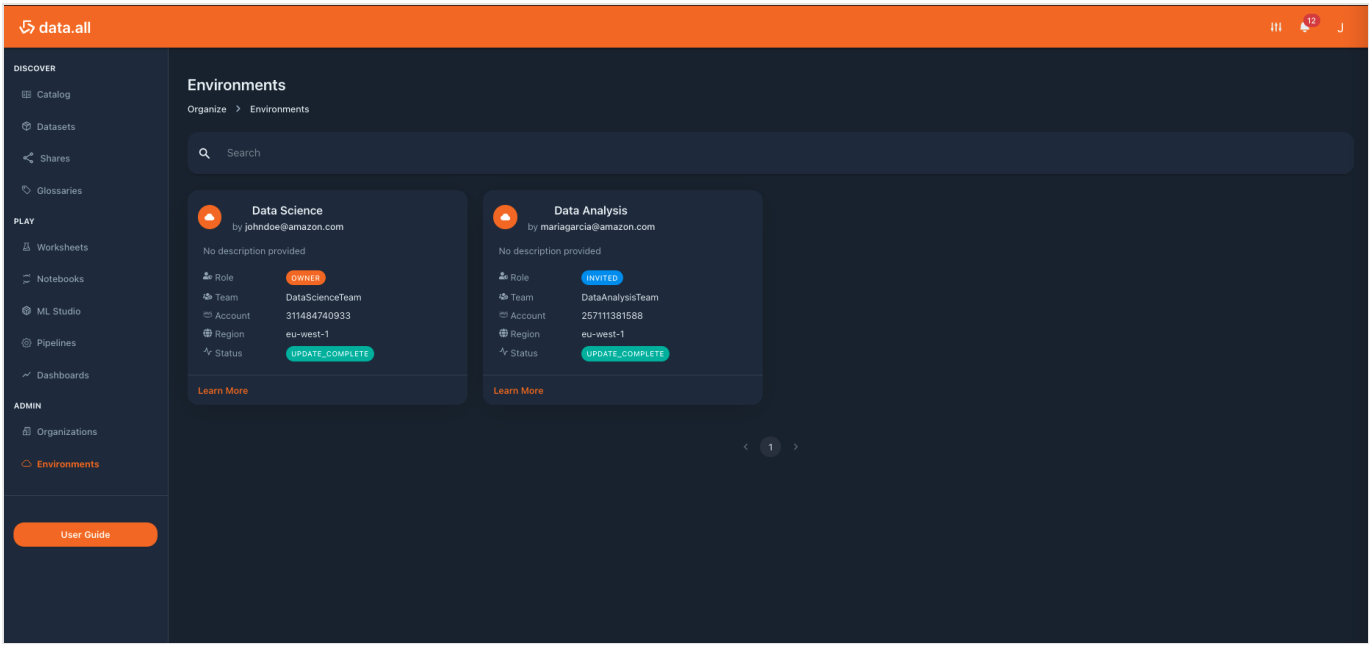


When the invitation is saved, the environment CloudFormation stack gets automatically updated and creates a new IAM role for the new team. The IAM role policies are mapped to the permissions and are granted to the invited team (e.g., a team invited without "Create ML Studio" permission will not have Sagemaker permissions on the associated IAM role).To remove a group, in the Actions column select the minus icon.

> **Automated permission assignment**
>
> Groups retrieved from the IdP are automatically granted all application high level permissions by default to accelerate the onboarding process.

Users will only be able to see the environments where a team that they belong to has been onboarded (either as creator of the environment or invited to the environment). In the following picture, John belongs to the DataScienceTeam that owns the Data Science environment, but on top of that he can access the Data Analysis environment because her team has been invited by Maria.

> **Pro tip!**
>
> You know whether you are `OWNER` or `INVITED` in an environment by checking **your Role** in that environment. This information appears in the picture in each environment box in the field "Role".

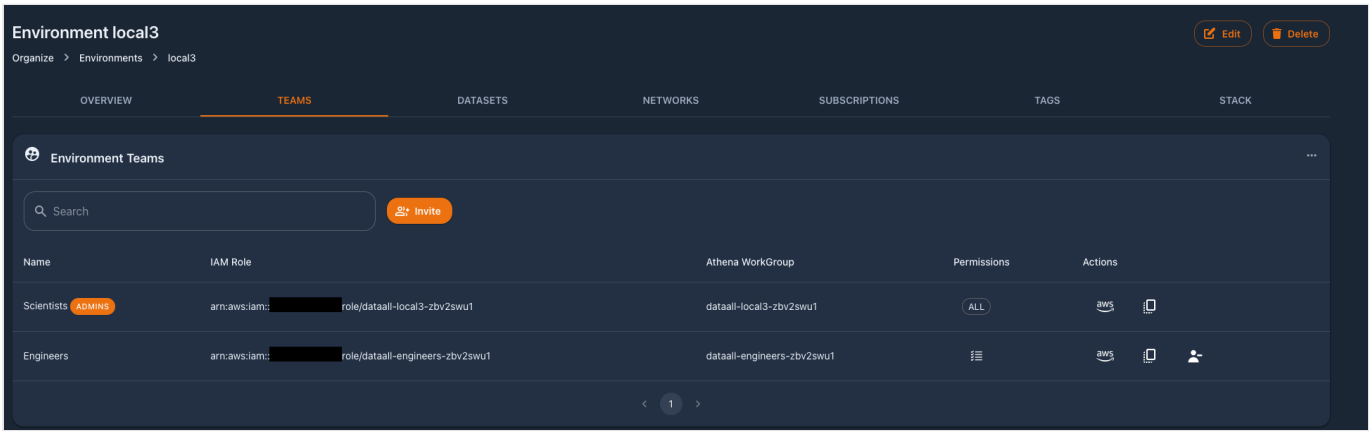


Difference between invited and owner

A team that has been invited to an environment has slight limitations, because well, it is not their environment! Invited teams cannot access the **Stack** tab of the environment because they should not be handling the resources of the environment. Same applies for **Tags** and **Subscriptions**. Other limitations come from the permissions that have been assigned to the team.

AWS access - Environment IAM roles

For the environment admin team and for each team invited to the environment data.all creates an IAM role. From the **Teams** tab of the environment we can assume our team's IAM role to get access to the AWS Console or copy the credentials to the clipboard. Both options are under the "Actions" column in the Teams table (these options are only available if `core.features.env_aws_actions` is set to `True` in the `config.json` used for deployment of data.all).



Usage

- Assumed by Team members from data.all UI to explore and work with data
- Credentials can be copied in data.all UI to explore and work with data
- Assumed by data.all Worksheets to query data using Athena

IAM Permissions

Default permissions

- read permissions to profiling/code folder in the Environment S3 Bucket
- Athena permissions to use the Team's workgroup
- CloudFormation permissions to resources tagged with Team tag and prefixed with environment `resource_prefix`
- SSM Parameter Store permissions to resources tagged with team tag and prefixed with environment `resource_prefix`
- Secrets Manager permissions to resources tagged with team tag and prefixed with environment `resource_prefix`
- read permissions on Logs and IAM
- PassRole permissions for itself to Glue, Lambda, SageMaker, StepFunctions and DataBrew

Data permissions

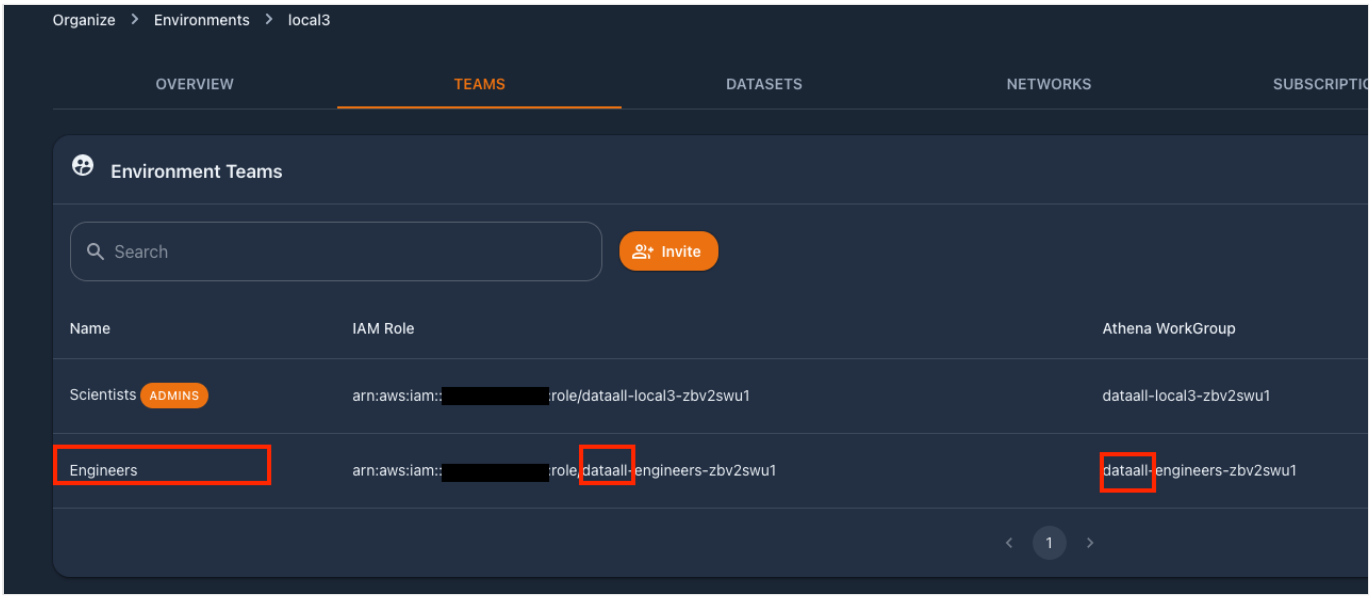
- read and write permissions to the Team-owned Dataset S3 Buckets
- encrypt/decrypt data with the Team-owned Dataset KMS keys
- read and write permissions Dataset Glue databases - governed with Lake Formation

Feature permissions

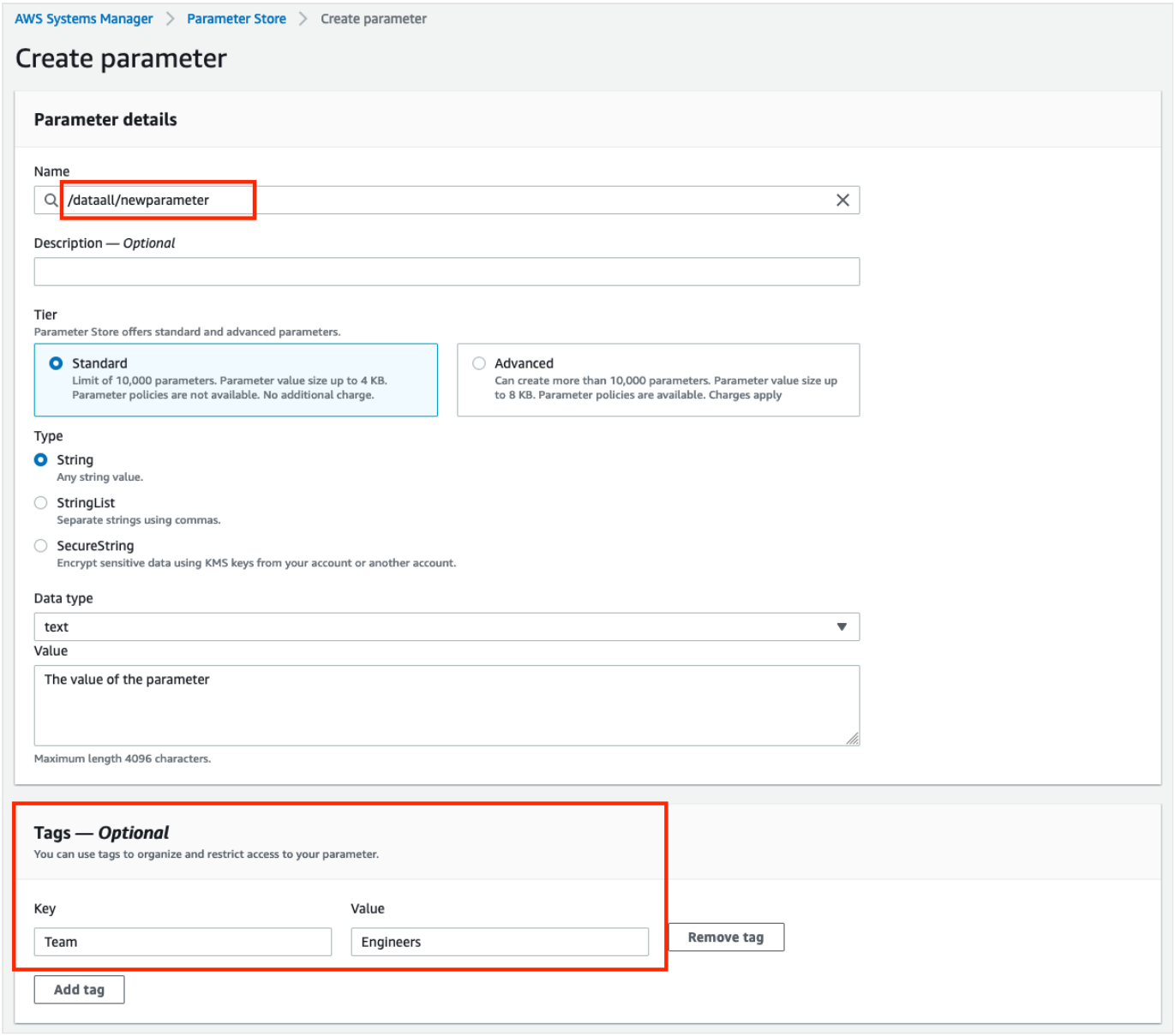
Depending on the features enabled in the environment and granted to the Team, additional AWS permissions are given to the role. Permissions for any AWS service need to be defined to allow access onlt to resources tagged with team tag and prefixed with environment `resource_prefix`

> **Access denied? You need to tag resources when you create them**
>
> Since permissions to AWS services are restricted to team-tagged resources, you need to tag any new resource that you create at creation time.

Let's say you are using the "Engineers" IAM role in an environment that prefixes all resources with the `resource_prefix` = "dataall" as in the following picture.



Assuming the IAM role you will be able to create parameters prefixed by "dataall" and tagged with a tag Team=Engineers, otherwise you will get AccessDenied errors.



All the resources created in the environment stack are tagged with the tag `Team=EnvAdminTeam`, which means that environment admins can access and manage the environment baseline AWS resources.

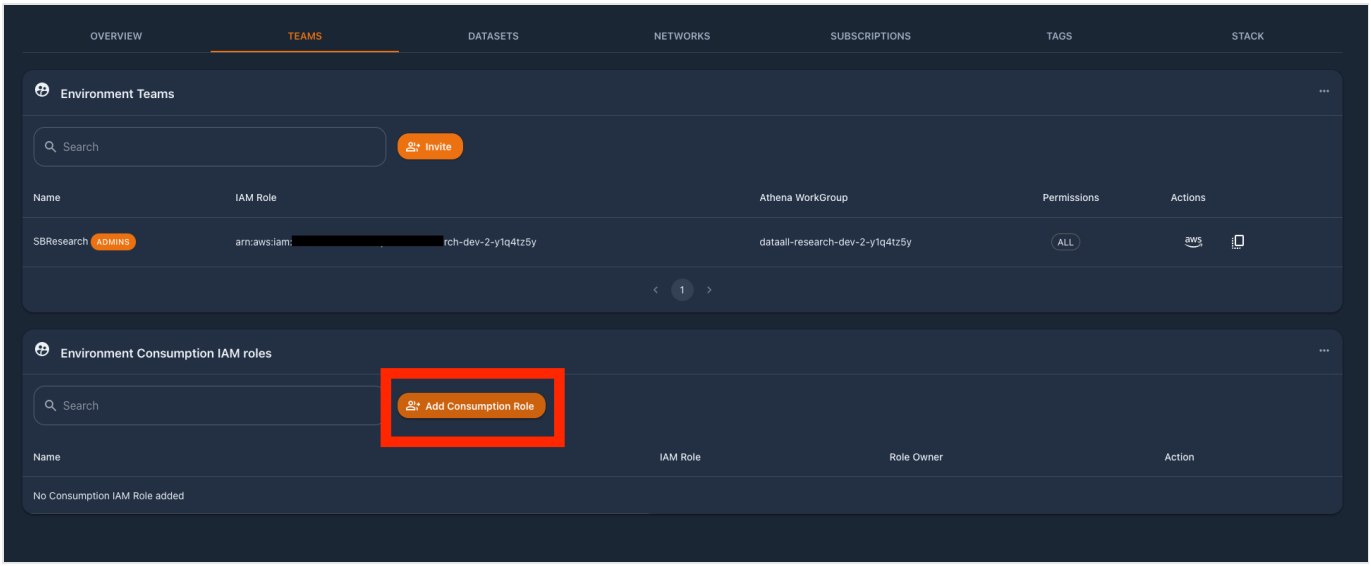
Data Governance with Lake Formation

We use AWS Lake Formation to govern Glue databases and tables. Using Lake Formation, we grant permissions to the Environment teams IAM roles to read and write the Glue databases and tables that the Team owns. In other words, each environment team IAM role can only access the Glue databases and tables of the Datasets that the team owns.

3.2.5 Manage Consumption Roles

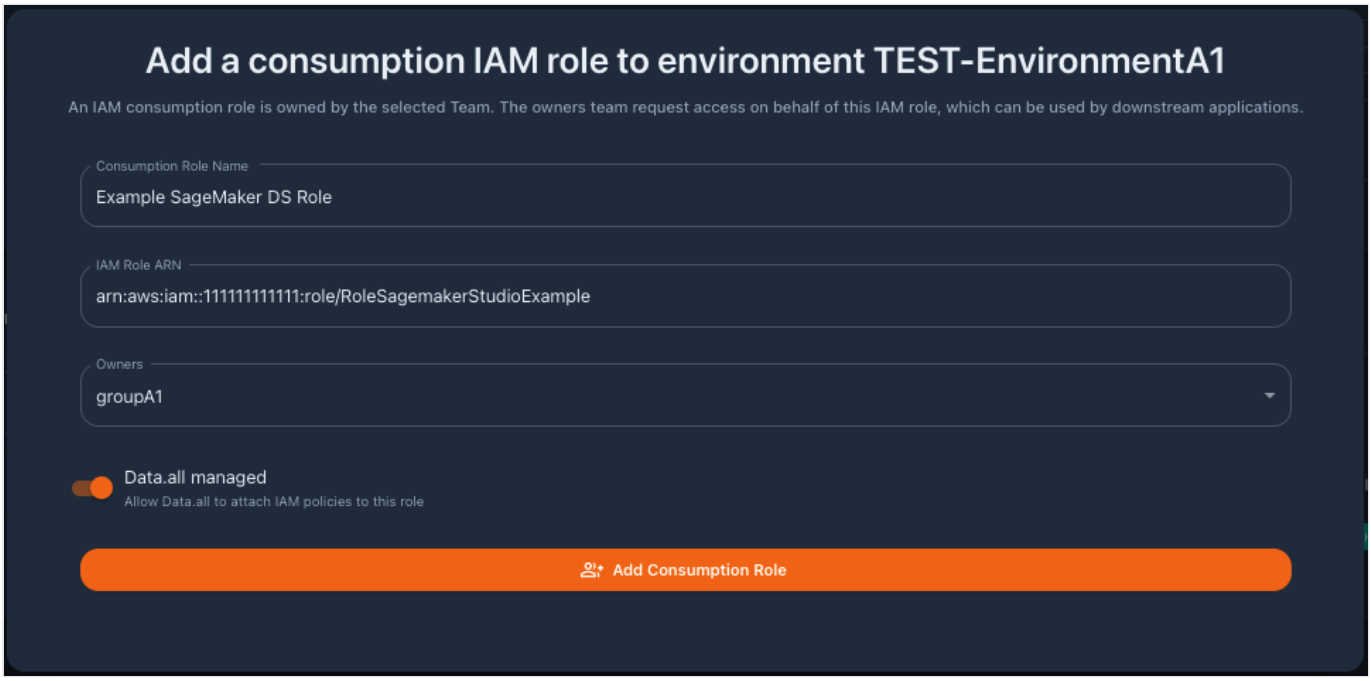
data.all creates or imports one IAM role per Cognito/IdP group that we invite to the environment. With these IAM roles data producers and consumers can ingest and consume data, but sometimes we want to consume data from an application that already has an execution role. To increase the flexibility in the data consumption patterns, data.all introduces Consumption Roles.

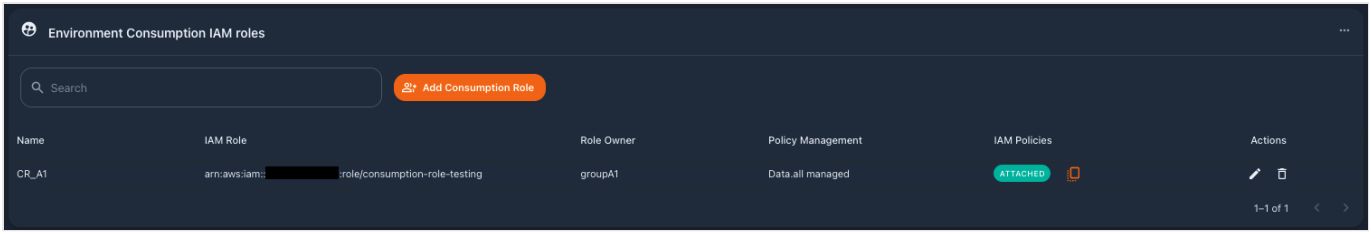
Any IAM role that exists in the Environment AWS Account can be added to data.all. In the **Teams** tab click on Add Consumption Role



A window like the following will appear for you to introduce a name for the consumption role in data.all, the arn of the IAM role, the Team that owns the consumption role and whether data.all should manage the consumption role. Enabling "data.all managed" on the consumption role allows data.all to attach IAM policies to the role used for data.all related activities, such as sharing data, rather than having a user manually add those policies to the role.

Only members of this team and tenants of data.all can edit or remove the consumption role.





Existing roles only

data.all checks whether that IAM role exists in the AWS account of the environment before adding it as a consumption role.

Data Access

- By default, a new consumption role does NOT have access to any data in data.all.
- The team that owns the consumption role needs to open a share request for the consumption role as discussed more in the Discover --> Shares section.

3.3 Maintenance Window

When deploying new releases, patch updates, etc there may arise a situation in which a user may be performing an action, and, at the same time some AWS resources might be getting updated. This can put data.all created components (Environments, Datasets, Dataset Shares) into broken state. Also, there might be a need to debug (or patch update few things in data.all) when the data.all administrators may want to restrict actions taken by users in data.all. In order to protect such a deployment and create a safe environment for deployment / patch updates, data.all can be put into maintenance mode.

In order to enable use of maintenance mode into your deployment of data.all, modify the config.json and add this to the modules section

```
"maintenance": {
    "active": true
}
```

Note: Only data.all administrators can start the maintenance mode. Maintenance window is available in the `Admin Settings` section. Data.all currently supports two maintenance modes.

Read-only : In this mode, a user can visit data.all and navigate through data.all but won't be able to update/modify any data.all related components. No-Access : In this mode, a user is shown a blank page after the user logs into data.all. In this mode, all user actions are blocked.

Note - During both the maintenance modes, data.all admins can perform all data.all actions (i.e. an admin can login and modify data.all related components where they have access)

The following happens when a maintenance mode / window is started in data.all

1. All Scheduled ECS tasks (such as Catalog-Indexer, Share Verifier, etc) are disabled
2. If there is any running ECS task at the time of starting maintenance window, the status of that ECS task is polled and only when all the ECS tasks have completed , the maintenance mode status is changed to ACTIVE - indicating that it is safe to deploy or carry out any maintenance activities.
3. GraphQL calls are blocked depending on the maintenance mode. If the maintenance mode is Read-Only, then only mutation graphql calls are blocked. In case of No-Access maintenance mode, both mutation and query graphql calls are blocked for the user.

3.3.1 **Enable / Disable Maintenance mode**

In order to enable maintenance mode, goto `Admin Settings` page - which is only accessible to data.all adminstrators - and navigate to the `Maintenance` tab. Once you are on the maintenance tab, select the mode and click on `Start Maintenance`.

Please wait for the maintenance window status to change from PENDING to ACTIVE before taking actions.

You can disable maintenance mode the same way it was enabled by clicking on `End Maintenance`.

4. Discover

4.1 S3 Datasets

In data.all, a S3/Glue Dataset is a representation of multiple AWS resources that helps users store data in a data lake and establish the basis to make this data discoverable and shareable with other teams.

When data owners create a S3/Glue dataset the following resources are deployed on the selected environment and its linked AWS account:

1. Amazon S3 Bucket to store the data on AWS.
2. AWS KMS key to encrypt the data on AWS.
3. AWS IAM role that gives access to the data on Amazon S3 (Dataset IAM role, see below)
4. AWS Glue database that is the representation of the structured data on AWS.

Dataset IAM role

Usage

- Assumed by Dataset owners from data.all UI to quickly ingest or access Dataset data
- Assumed by Dataset Glue crawler
- Assumed by the Dataset Glue profiling job

IAM Permissions

- read and write permissions to the Dataset S3 Bucket (ONLY this bucket)
- encrypt/decrypt data with the Dataset KMS key (ONLY this key)
- read and write permissions to the Dataset Glue database and tables (ONLY this database)
- read permissions to profiling/code folder in the Environment S3 Bucket (ONLY this folder)
- read and write permissions to profiling/results/datasetUri folder in the Environment S3 Bucket (ONLY this folder)
- put logs permissions to log crawler and profiling jobs results

Data Governance with Lake Formation

In addition to restricting the access via IAM policies, Dataset Glue database and tables are protected using AWS Lake Formation. With Lake Formation, the Dataset IAM role gets granted access to the Dataset Glue database only.

Glue Tables and S3 Folders

Inside a S3/Glue dataset we can store structured data in Glue tables and unstructured data in S3 folders.

- Tables are the representation of **AWS Glue Catalog** tables that are created on the dataset's Glue database on AWS.
- Folders are the representation of an **Amazon S3 prefix** where any type of file can be stored. Such as images, unstructured text formats...

Dataset ownership

Dataset ownership refers to the ability to access, modify or remove data from a dataset, but also to the responsibility of assigning these privileges to others.

- **Owners**: When you create a dataset and associate it with a team, the dataset business ownership belongs to the associated team.
- **Stewards**: You can delegate the stewardship of a dataset to a team of stewards. You can type a name of an IdP group or choose one of the teams of your environment to be the dataset stewards.

> **Note**
>
> Dataset owners team is a required, non-editable field, while stewards are optional and can be added post the dataset has been created. If no other stewards team is designated, the dataset owner team will be the only responsible in managing access to the dataset.

Dataset access

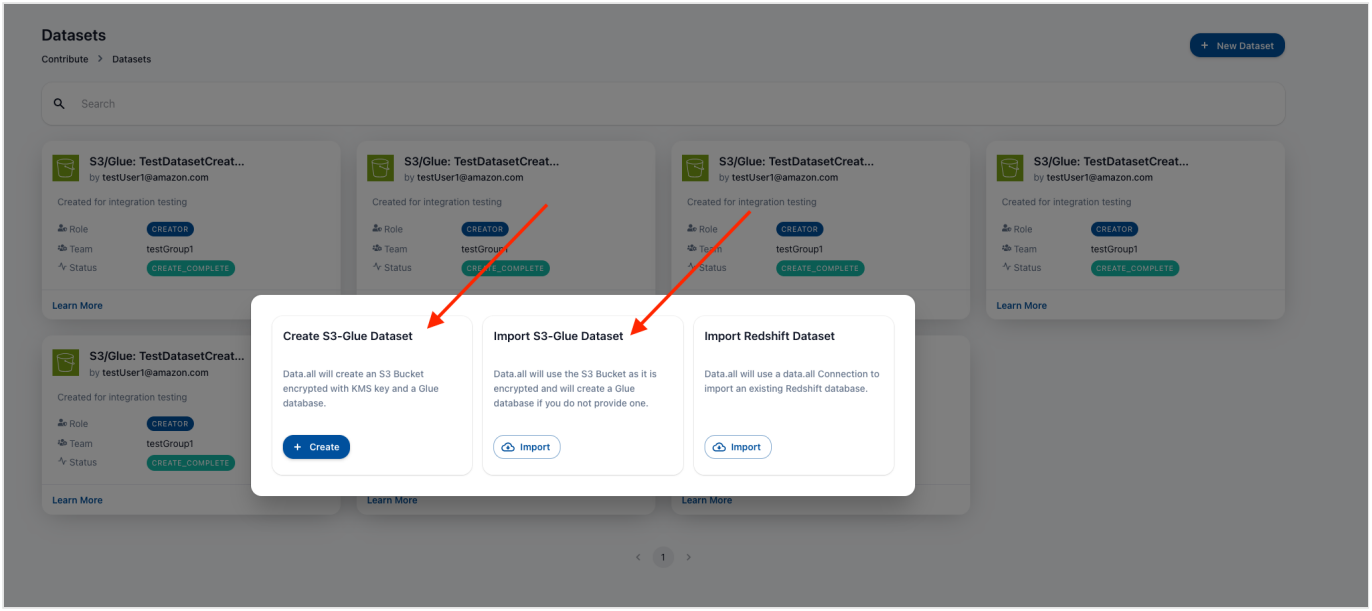
In this case we are referring to the ability to access, modify or remove data from a dataset. Who can access the dataset content? users belonging to...

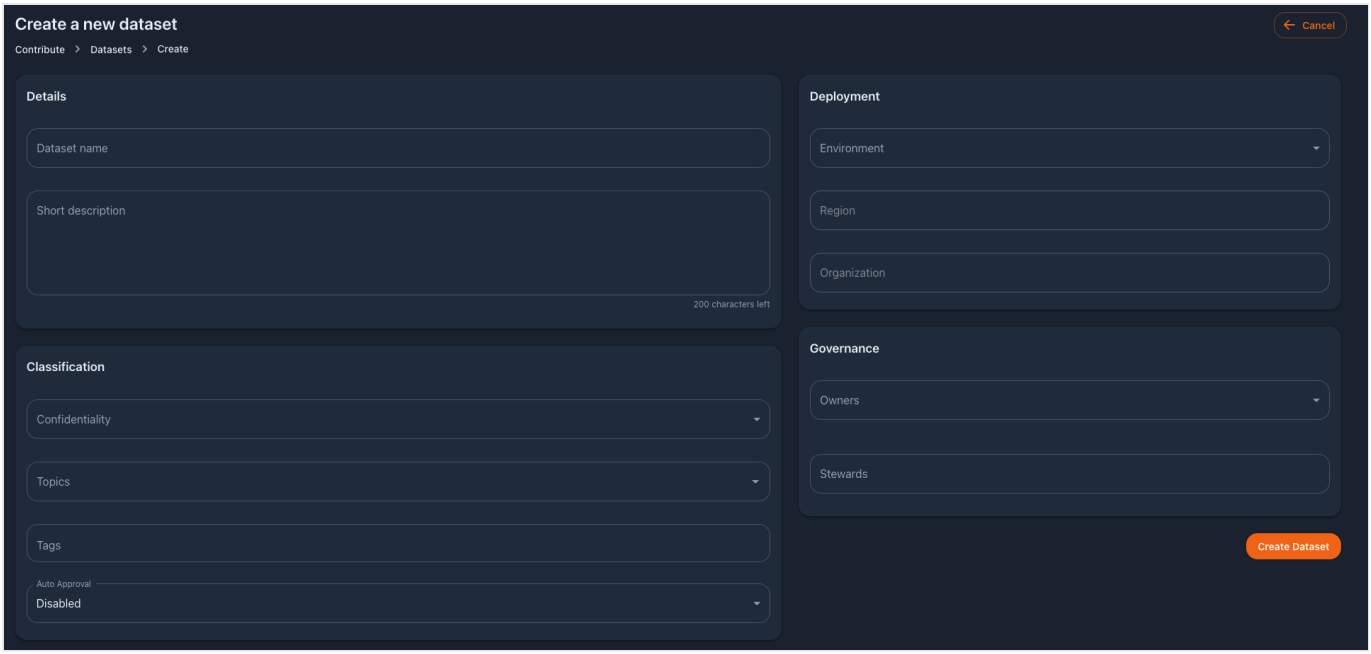
- the dataset owner team
- a dataset steward team
- teams with a share request approved to dataset content

> **Note**
>
> Dataset metadata is available for all users in the centralized data catalog.

4.1.1 NEW Create a dataset

To create a new dataset, navigate to the Datasets view and click on **New Dataset**. A window like the one in the picture will allow you to select the type of Dataset you want to create or import. In this case you need to select the Create S3/Glue Dataset option.





Field	Description	Required	Editable	Example
Dataset name	Name of the dataset	Yes	Yes	AnyDataset
Short description	Short description about the dataset	No	Yes	For AnyProject predictive model
Environment	Environment (mapped to an AWS account)	Yes	No	DataScience
Region (auto-filled)	AWS region of the environment	Yes	No	Europe (Ireland)
Organization (auto-filled)	Organization of the environment	Yes	No	AnyCompany EMEA
Owners	Team that owns the dataset	Yes	No	DataScienceTeam
Stewards	Team that can manage share requests on behalf of owners	No	Yes	FinanceBITeam, FinanceMgmtTeam
Confidentiality	Level of confidentiality: Unclassified, Oficial or Secret	Yes	Yes	Secret
Topics	Topics that can later be used in the Catalog	Yes, at least 1	Yes	Finance
Tags	Tags that can later be used in the Catalog	Yes, at least 1	Yes	deleteme, ds
Auto Approval	Whether shares for this dataset need approval from dataset owners/ stewards	Yes (default `Disabled`)	Yes	Disabled, Enabled

4.1.2 Import a dataset

If you already have data stored on Amazon S3 buckets in your data.all environment, data.all has got you covered with the import feature. In addition to the fields of a newly created dataset you have to specify the S3 bucket and optionally a Glue database and a KMS key Alias. If the Glue database is left empty, data.all will create a Glue database pointing at the S3 Bucket. As for the KMS key Alias, data.all assumes that if nothing is specified the S3 Bucket is encrypted with SSE-S3 encryption. Data.all performs a validation check to ensure the KMS Key Alias provided (if any) is the one that encrypts the S3 Bucket specified.

Imported KMS key and S3 Bucket policies requirements

Data.all pivot role will handle data sharing on the imported Bucket and KMS key (if imported). Make sure that the resource policies allow the pivot role to manage them. For the KMS key policy, explicit permissions are needed. See an example below.

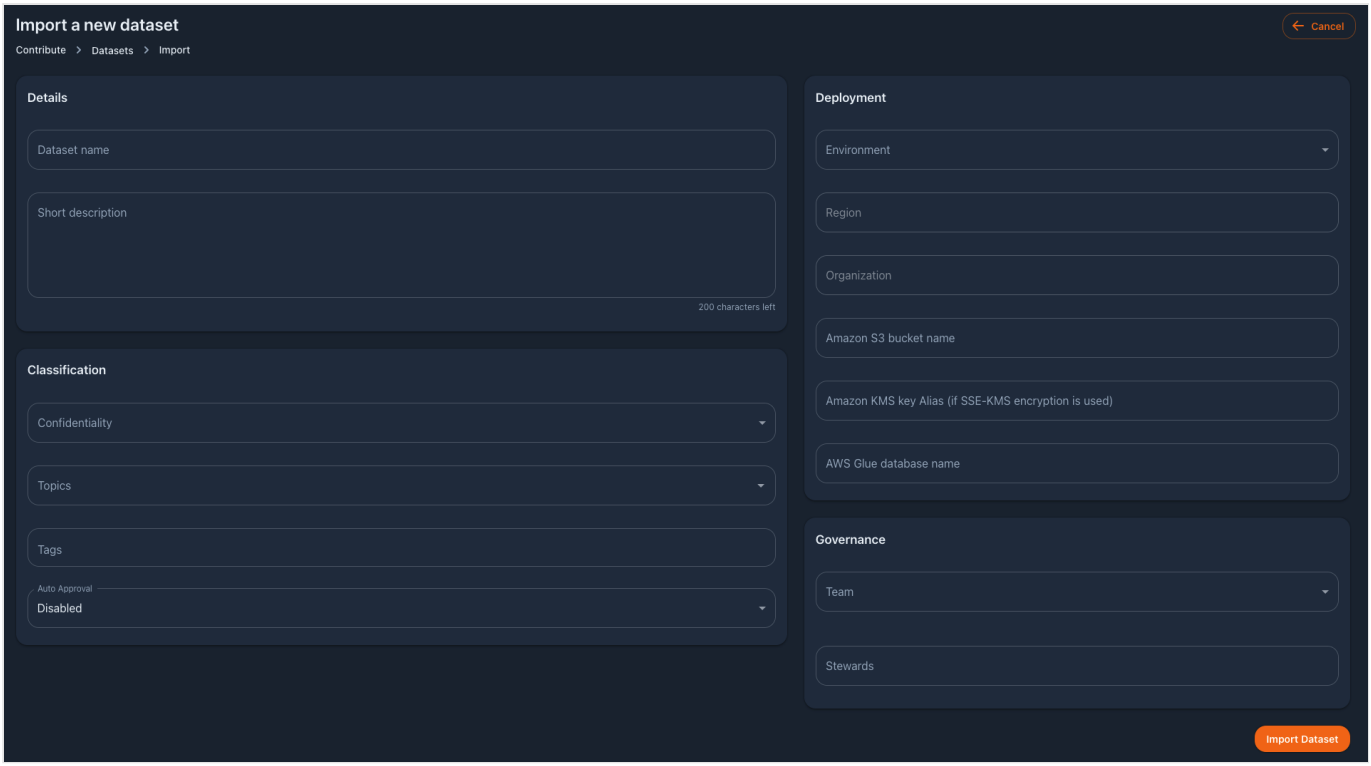
KMS key policy

In the KMS key policy we need to grant explicit permission to the pivot role. At a minimum the following permissions are needed for the pivotRole:

```
{
  "Sid": "Enable Pivot Role Permissions",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111122223333:role/dataallPivotRole-cdk"
   },
  "Action": [
    "kms:Decrypt",
    "kms:Encrypt",
    "kms:GenerateDataKey*",
    "kms:PutKeyPolicy",
    "kms:GetKeyPolicy",
    "kms:ReEncrypt*",
    "kms:TagResource",
    "kms:UntagResource"
    'kms:DescribeKey'
   ],
  "Resource": "*"
}
```

Update imported Datasets

Imported keys is an addition of V1.6.0 release. Any previously imported bucket will have a KMS Key Alias set to `Undefined`. If that is the case and you want to update the Dataset and import a KMS key Alias, data.all let's you edit the Dataset on the **Edit** window.



Field	Description	Required	Editable	Example
Amazon S3 bucket name	Name of the S3 bucket you want to import	Yes	No	DOC-EXAMPLE-BUCKET
Amazon KMS key Alias	Alias of the KMS key used to encrypt the S3 Bucket (do not include alias/, just)	No	No	somealias
AWS Glue database name	Name of the Glue database tht you want to import	No	No	anyDatabase

(Going Further) Support for Datasets with Externally-Managed Glue Catalog

If the dataset you are trying to import relates to Glue Database that is managed in a separate account, data.all's import dataset feature can also handle importing and sharing these type of datasets in data.all. Assuming the following pre-requisites are complete:

- There exists an AWS Account (i.e. the Catalog Account) which is:
- Onboarded as a data.all environment (e.g. Env A)
- Contains the Glue Database with Location URI (as S3 Path from Dataset Producer Account) AND Tables
- Glue Database has a resource tag `owner_account_id=<PRODUCER_ACCOUNT_ID>`
- Data Lake Location registered in LakeFormation with the role used to register having permissions to the S3 Bucket from Dataset Producer Account
- Resource Link created on the Glue Database to grant permission for the Dataset Producer Account on the Database and Tables
- There exists another AWS Account (i.e. the Dataset Producer Account) which is:
- Onboarded as a data.all environment (e.g. Env B)
- Contains the S3 Bucket that contains the data (used as S3 Path in Catalog Account)

The data.all producer, a member of EnvB Team(s), would import the dataset specifying the S3 bucket as the bucket name that exists in the Dataset Producer Account and specifying the Glue database name as the Glue DB resource link name in the Dataset Producer Account.

This dataset will then be properly imported and can be discovered and shared the same way as any other dataset in data.all.

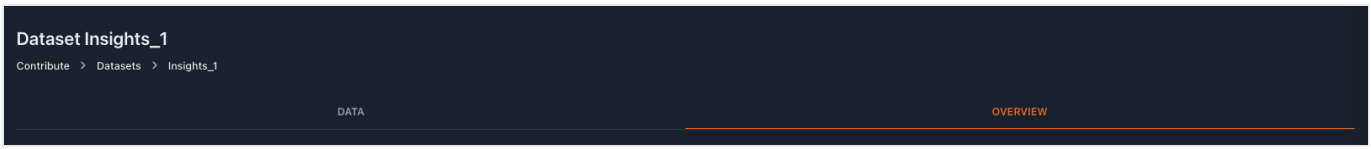
4.1.3 Navigate dataset tabs

When we belong to the dataset owner team

After creating or importing a dataset it will appear in the datasets list (click on Datasets on the left side pane). In this window, it will only ve visible for those users belonging to the dataset owner team. If we select one of our datasets we will see the following dataset window:

When we DON'T belong to the dataset owner team

How do we access a dataset if we don't have access to it? IN THE CATALOG! on the left pane click on Catalog, find the dataset you are interested in, click on it and if you don't have access to it, you should see only some of the tabs in comparison with the previous pic, something like:



4.1.4 Edit and update a dataset

Data owners can edit the dataset by clicking on the **edit** button, editing the editable fields and saving the changes.

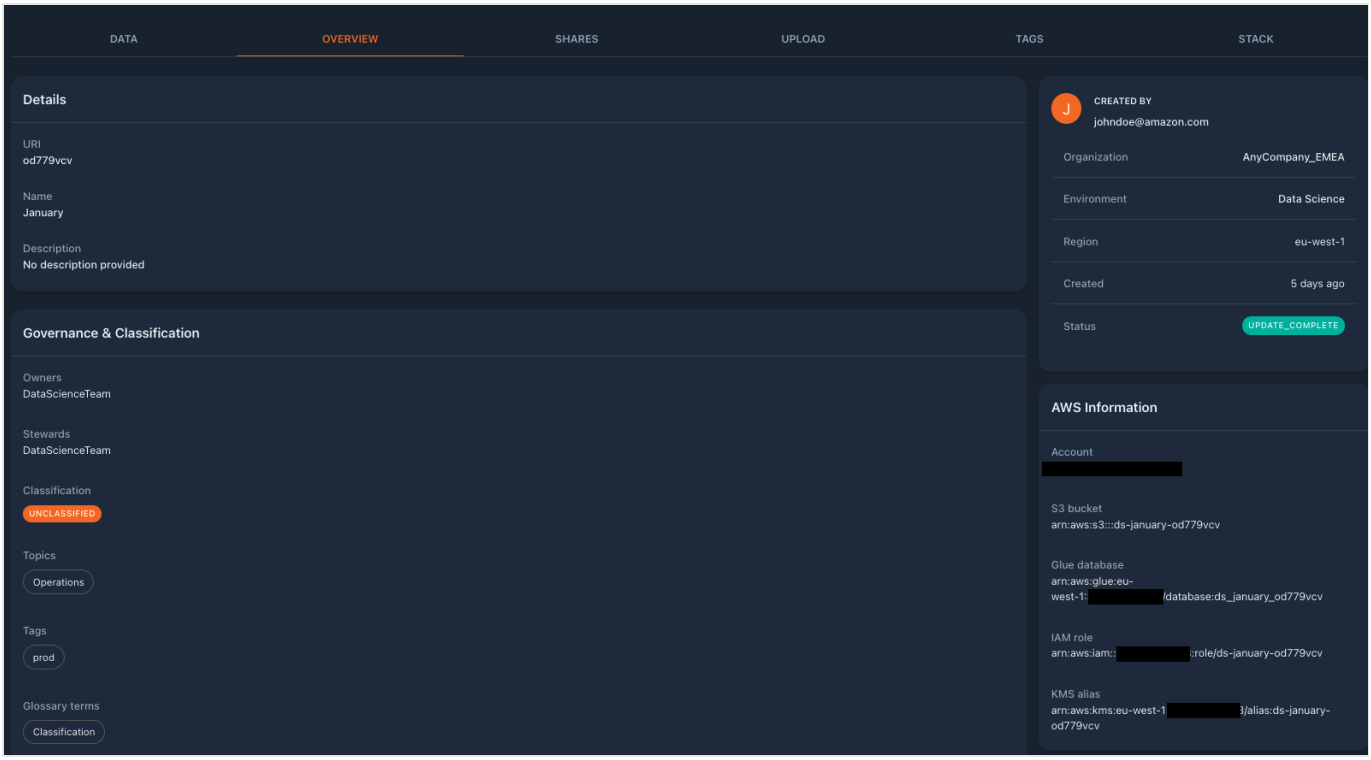
4.1.5 Delete a dataset

To delete a dataset, in the selected dataset window click on the **delete** button in the top-right corner. As with environments, it is possible to keep the AWS CloudFormation stack to keep working with the data and resources created but outside of data.all.

4.1.6 Check dataset info and access AWS

The **Overview** tab of the dataset window contains dataset metadata, including governance and creation details. Moreover, AWS information related to the resources created by the dataset CloudFormation stack can be consulted here: AWS Account, Dataset S3 bucket, Glue database, IAM role and KMS Alias.

You can also assume this IAM role to access the S3 bucket in the AWS console by clicking on the **S3 bucket** button. Alternatively, click on **AWS Credentials** to obtain programmatic access to the S3 bucket (only available if `modules.dataset.features.aws_actions` is set to `True` in the `config.json` used for deployment of data.all).



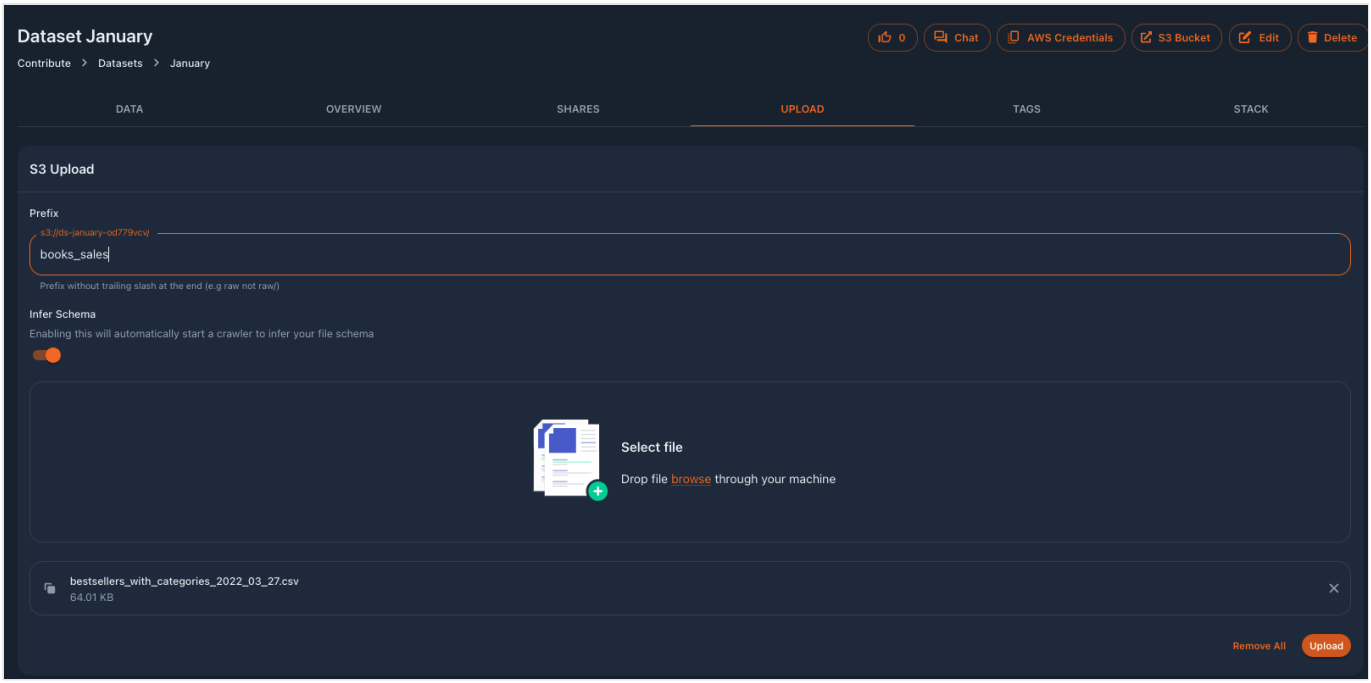
4.1.7 Fill the dataset with data

Tables

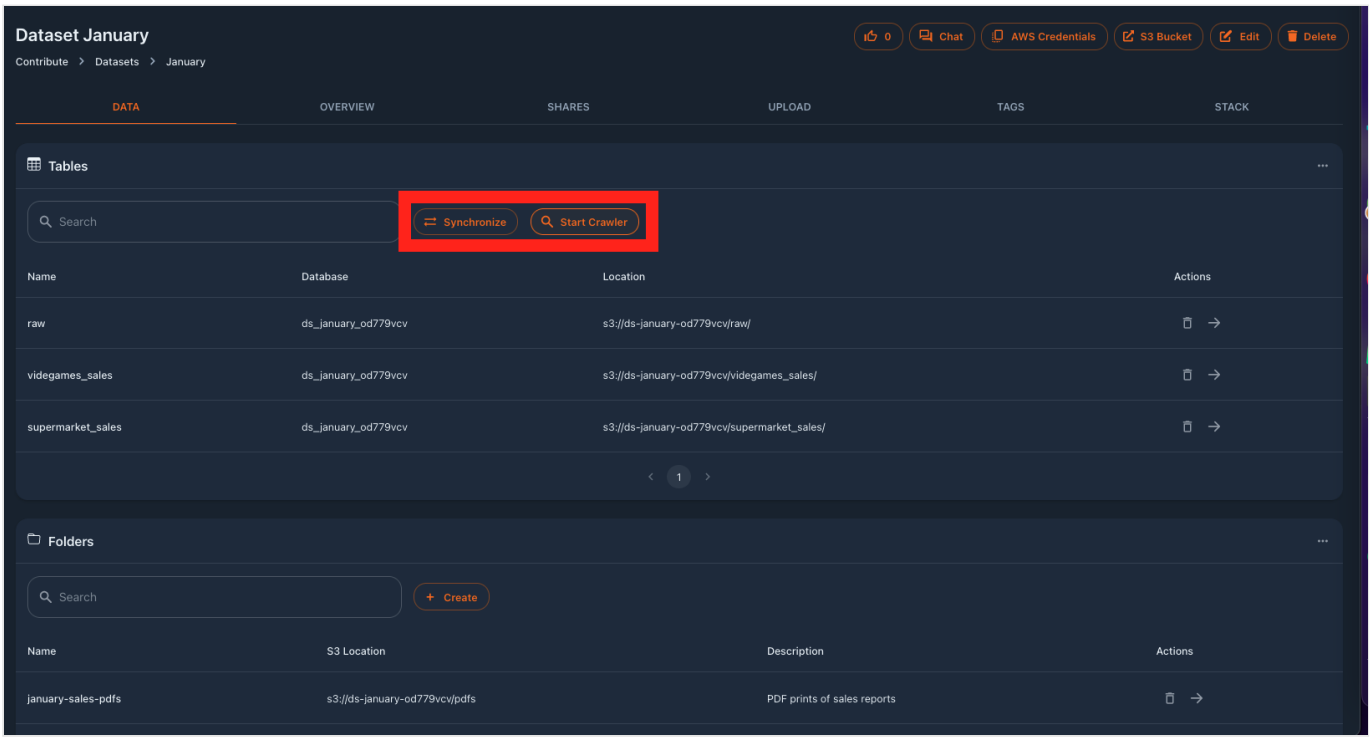
Quickly upload a file for data exploration

Users may want to experiment with a small set of data (e.g. a csv file). To create tables from a file, we first upload the file, then run the crawler to infer its schema, and finally, we read the schema by synchronizing the table. Upload & Crawl & Sync

1. Upload data: Go to the **Upload** tab of the dataset and browse or drop your sample file. It will be uploaded to the dataset S3 bucket in the prefix specified. By default, a Glue crawler will be triggered by the upload of a file, however this feature can be disabled as appears in the picture.



1. Crawl data: the file has been uploaded but the table and its schema have not been registered in the dataset Glue Catalog database. If you have disabled the crawler in the upload, click on the **Start Crawler** button in the Data tab. If you just want to crawl one prefix, you can specify it in the Start Crawler feature.



1. Synchronize tables: Once crawled and registered in the Glue database, you can synchronize tables from your dataset's AWS Glue database by using **Synchronize** tables feature in the Data tab. In any case, data.all will synchronize automatically the tables for you at a frequency of **15 minutes**.

You can preview your small set of data right away from data.all, check Tables.

Ingest data

If you need to ingest larger quantities of data, manage bigger files, or simply you cannot work with local files that can be uploaded; this is your section!

There are multiple ways of filling our datasets with data and actually, the steps don't differ much from the upload-crawl-sync example.

- Crawl & Sync option: we can drop the data from the source to our dataset S3 bucket. Then, we will crawl and synchronize data as we did in the previous steps 2 and 3.
- Register & Sync option: we drop the data from the source to our dataset S3 bucket. However, if we want to have more control over our tables and its schema, instead of starting the crawler we can **register the tables** in the Glue Catalog and then click on Synchronize as we did in step 3.

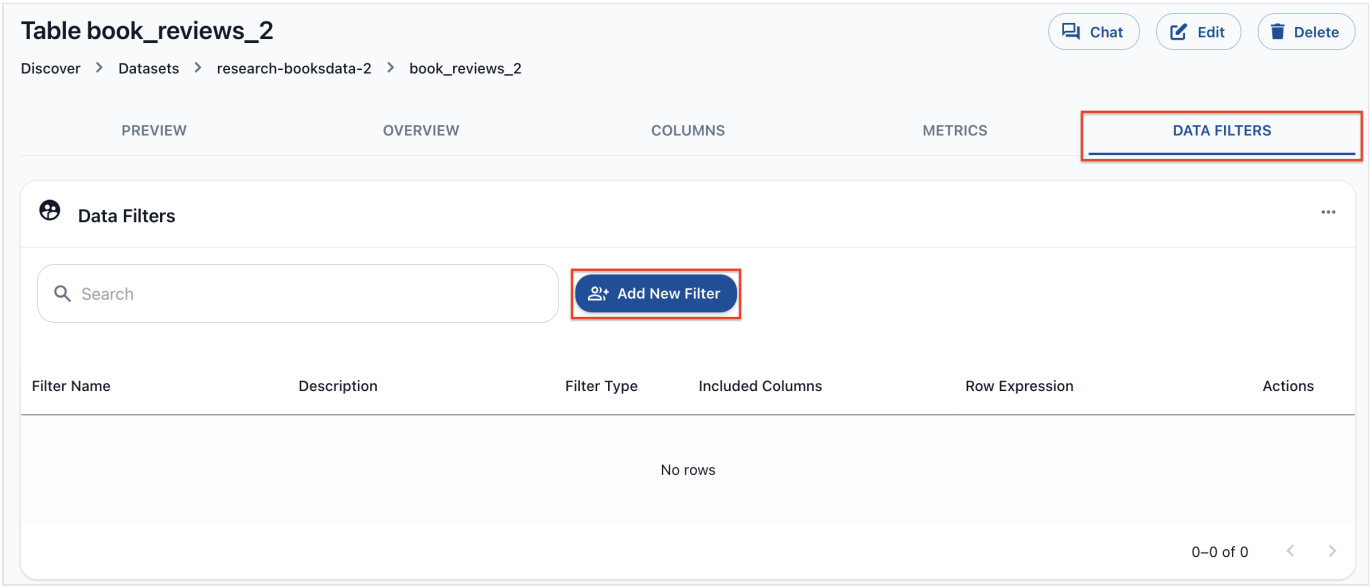
How do we register Glue tables? There are numerous ways:

- manually from the AWS Glue console in your environment account
- Using AWS Glue API, `CreateTable`.
- In a Glue Job leveraging Glue PySpark DynamicFrame class
- With boto3
- Or with AWS Data Wrangler, Pandas on AWS.
- Also, you can deploy Glue resources using CloudFormation
- Or directly, migrating from Hive Metastore.
- there are more for sure :)

(GOING FURTHER) CREATING FILTERS ON TABLES

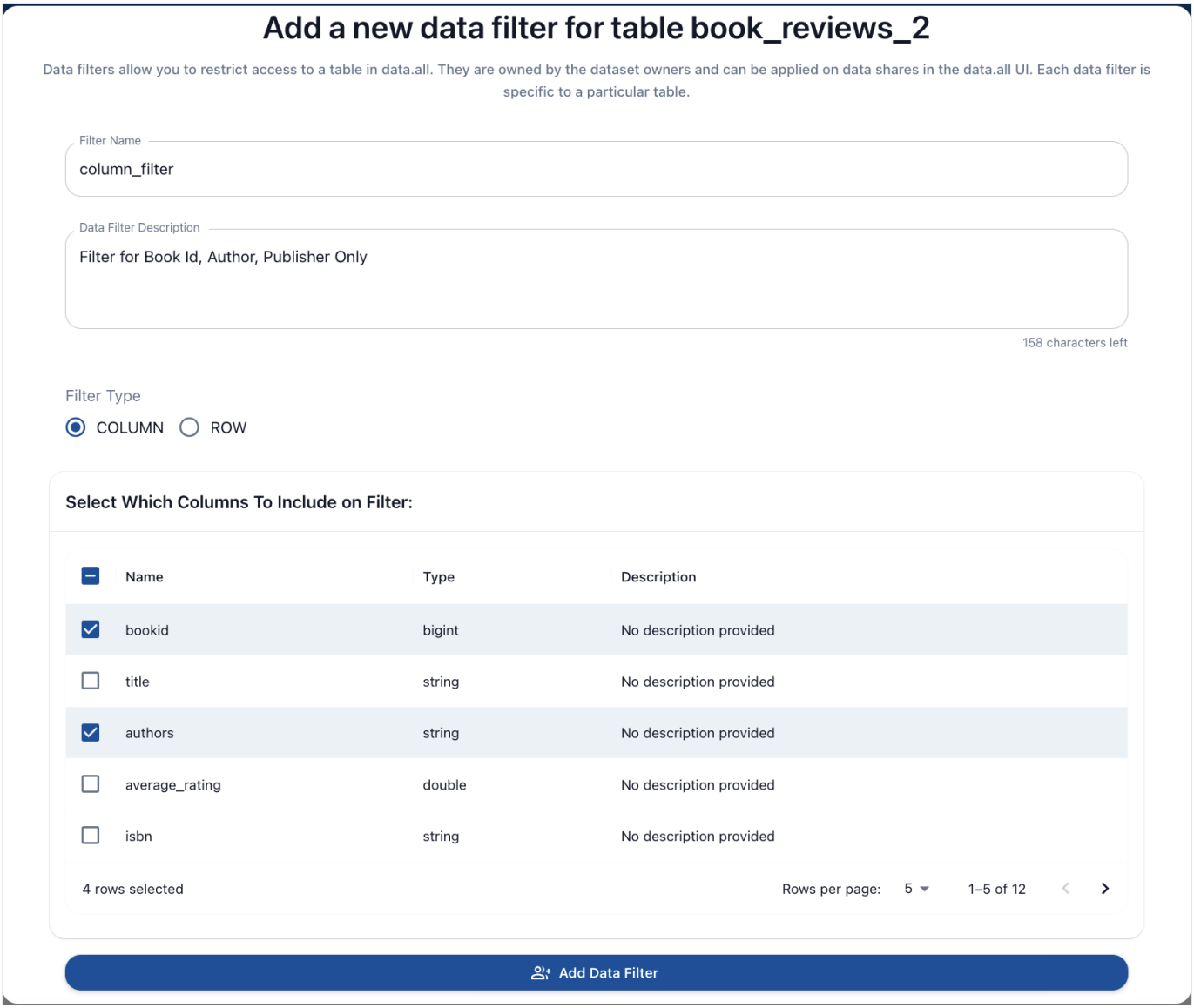
Additionally, dataset owners can create column-level or row-level filters on their dataset tables to more granularly restrict data access when sharing with other teams.

To do so dataset owners can navigate to the **Filters** Tab for a given table and select **Add New Filter**:



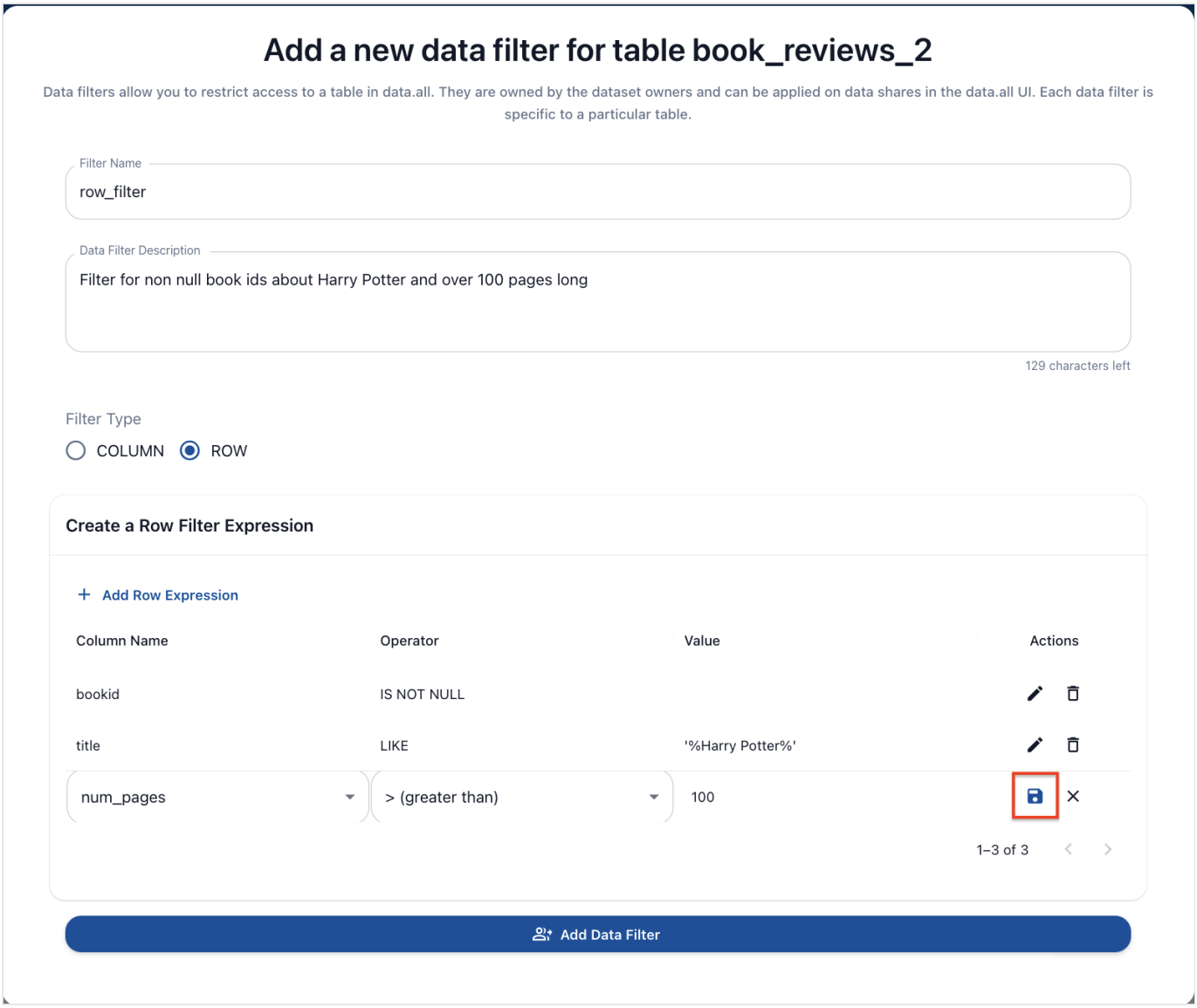
When creating filters, you have the choice to create a column-level filter or a row-level filter. Column-level filters prompt the user to select a subset of columns to include for the table. Row-level filters use row expressions to specify the rows to include in for the table.

An example of creating a column filter is below:



This filter restricts access on the table to only the 3 selected columns: `book_id`, `author`, and `publisher`.

An example of creating a row filter is below:



This filter restricts access to only rows where `book_id` is not null, `title` is LIKE `%Harry Potter%` AND `num_pages` is greater than `100`. It is important to note that: - The row filter acts as the intersection (logical 'AND') of the row expression(s) - if you need the union (logical 'OR') of multiple expressions you can create separate filters here and apply multiple to the table share item - When creating a new row expression be sure to save the row expression by clicking the save icon (highlighted in red in the above) before creating the filter

Once the filters are created, they will show in the Filters Table Tab:

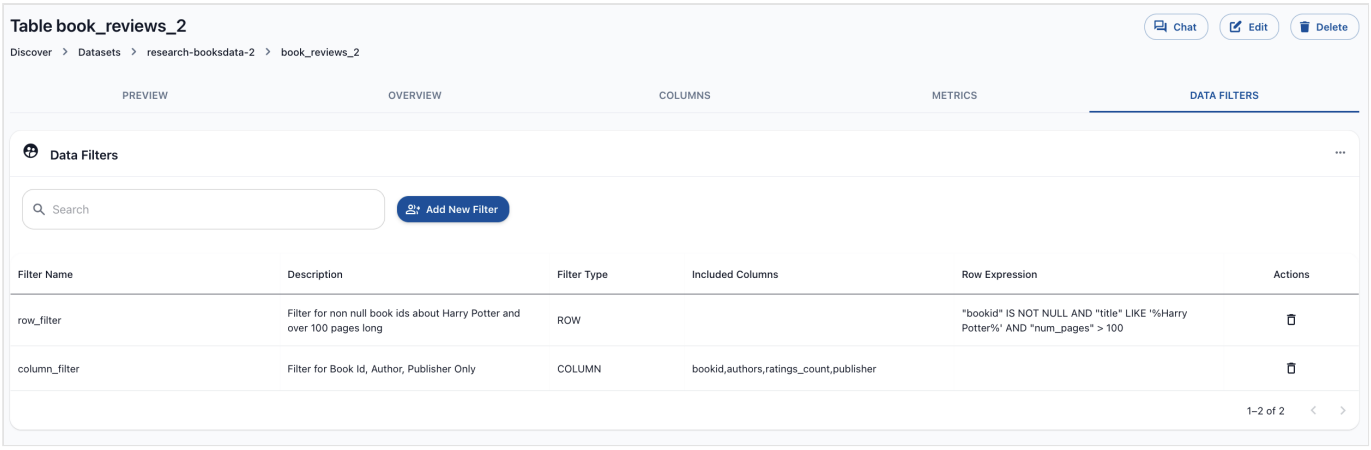


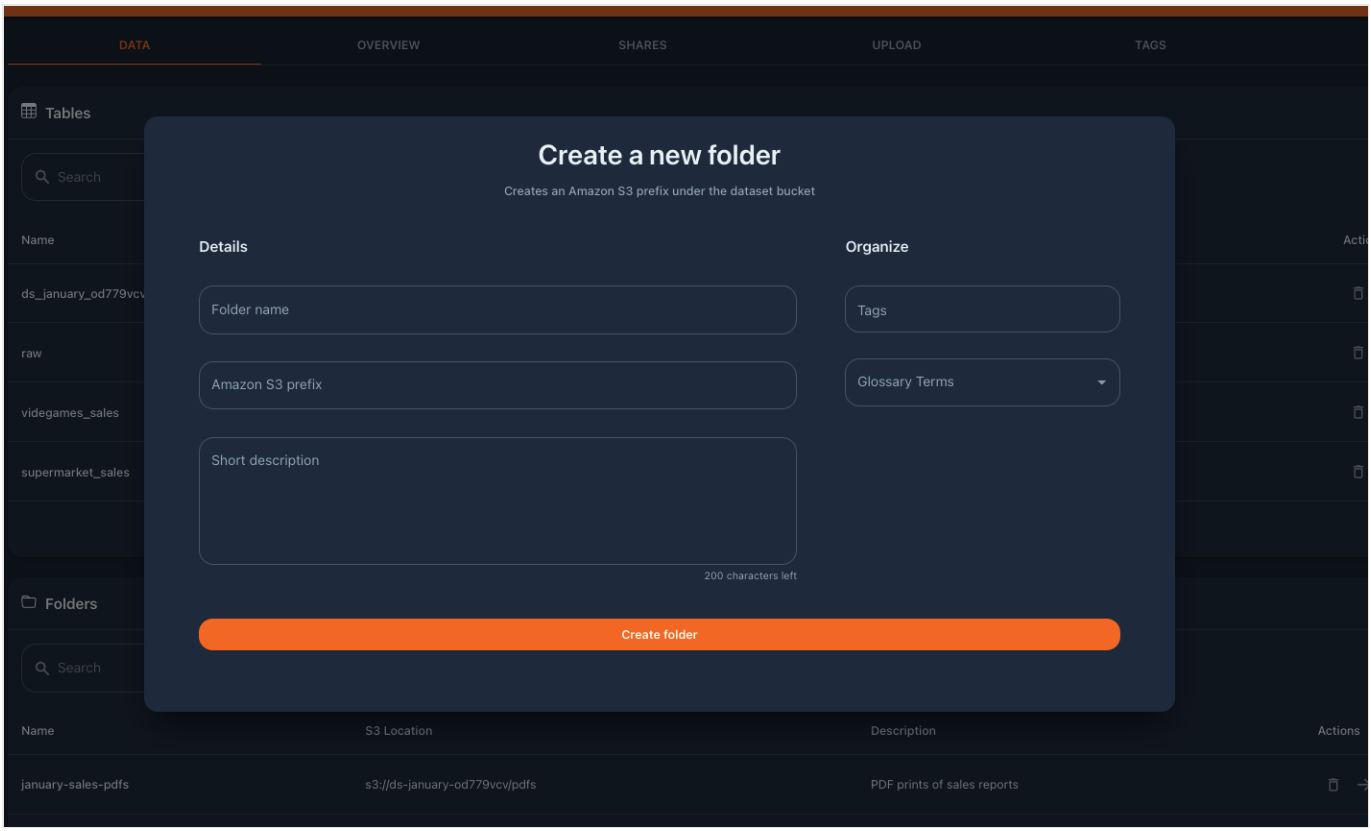
Table filters are not editable. To update an existing filter you must:

1. Revoke all associated share items using the filter (if applicable)
2. Delete the table filter
3. Create a new table filter with any updates as necessary

These filters can be used when reviewing and approving share objects with table share items to more granularly limit data access.

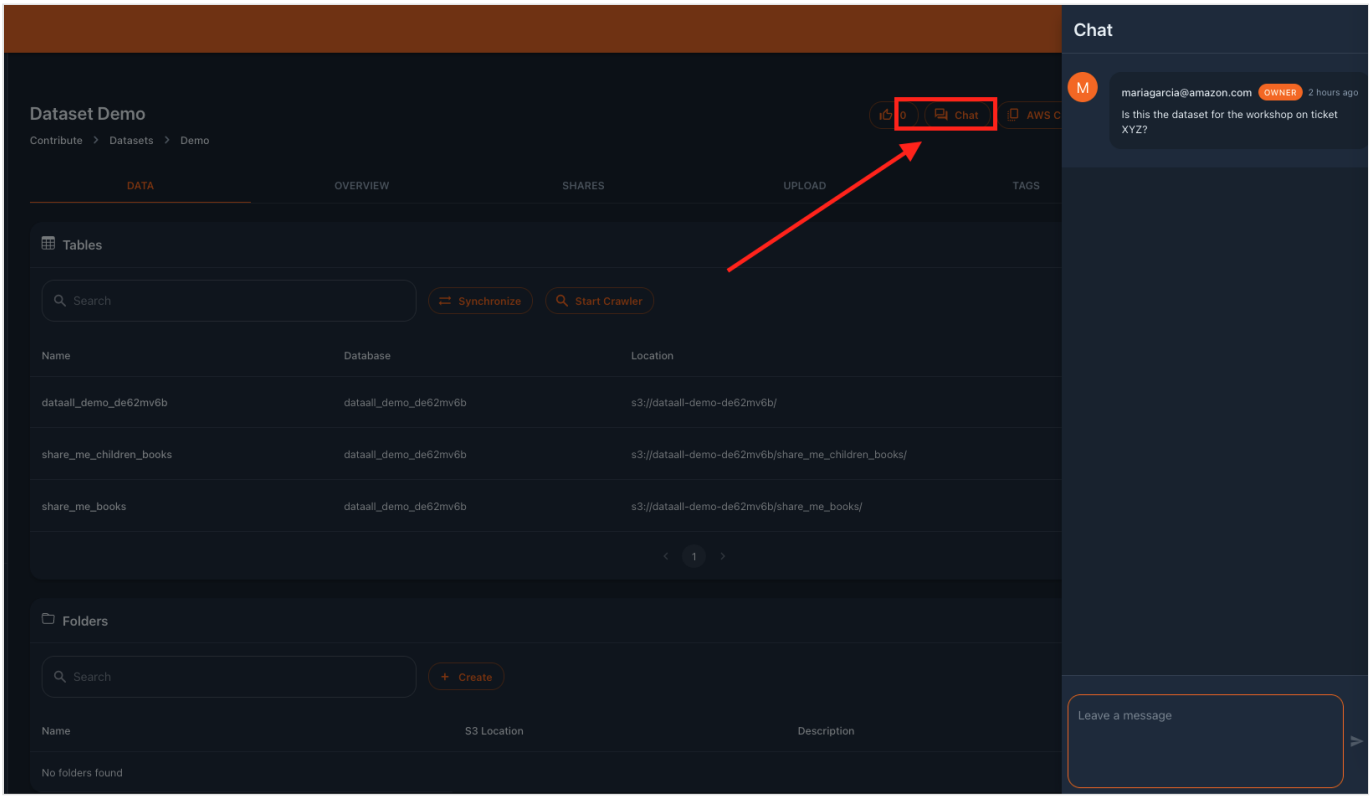
Folders

As previously defined, folders are prefixes inside our dataset S3 bucket. To create a folder, go to the **Data** tab and on the folders section, click on Create. The following form will appear. We will dive deeper in how to use folders in the folders section.



4.1.8 Leave a message in Chat

In the **Chats** button users can interact and leave their comments and questions on the Dataset Chat.



4.1.9 Create key-value tags

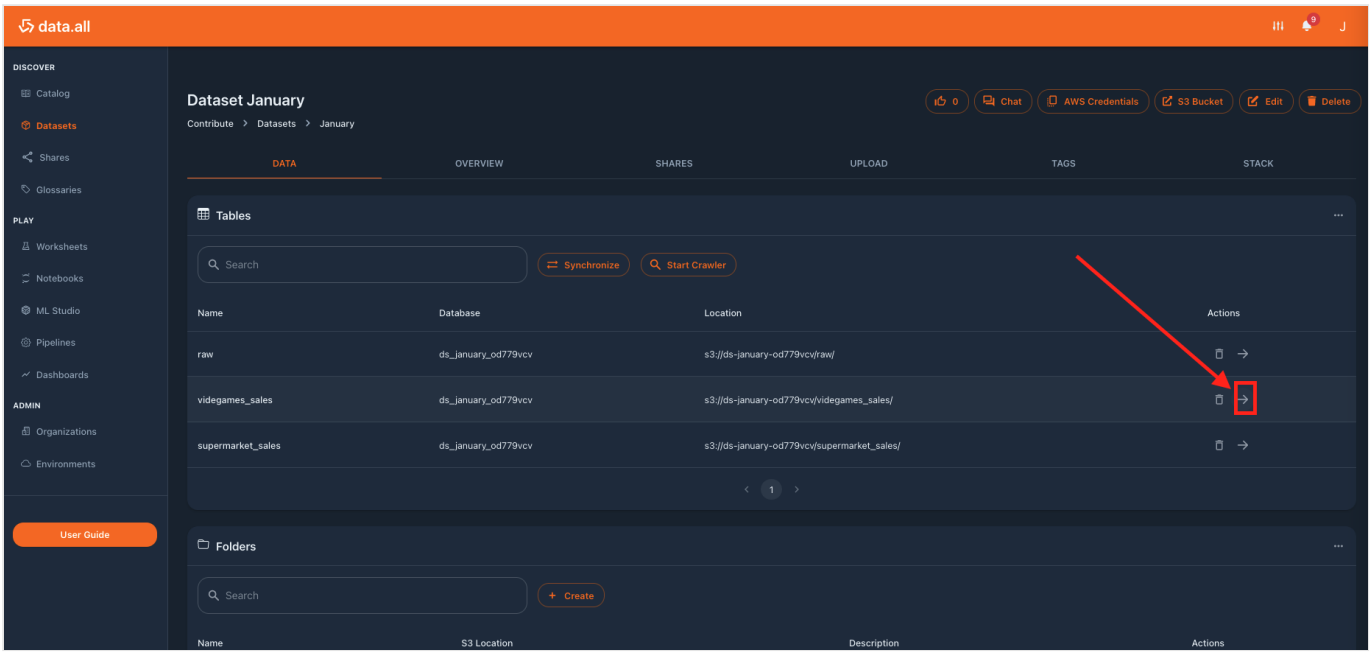
Same as in environments. In the **Tags** tab of the dataset window, we can create key-value tags. These tags are not data.all tags that are used to tag the dataset and find it in the catalog. In this case we are creating AWS tags as part of the dataset CloudFormation stack. There are multiple tagging strategies as explained in the documentation.

4.2 Glue Tables and S3 Folders

4.2.1 **Glue Tables**

In this section we will go through the different tabs in the Table window. We can reach this view:

1. by selecting a table from the data Catalog
2. or in the dataset view, in the **Tables** tab clicking on the arrow in the Actions column for the chosen table.



Check table metadata

Also in the table window, go to the **Overview** tab where you will find the following information:

- URI: unique table identifier
- Name: name of the registered table in the Glue Catalog
- Tags
- Glossary terms
- Description
- Organization, Environment, Region, Team: inherited from the dataset
- Created: creation time of the table
- Status: `INSYNC`

> **Description, Tags and Glossary terms are not inherited!**
>
> If a dataset is tagged with Tags and Glossary terms, the child tables do not inherit these tags and terms. In the Overview tab, by clicking on **Edit** is where you can add them. Same applies for the description. Adding tags and terms to your tables will make them more discoverable in the Catalog.

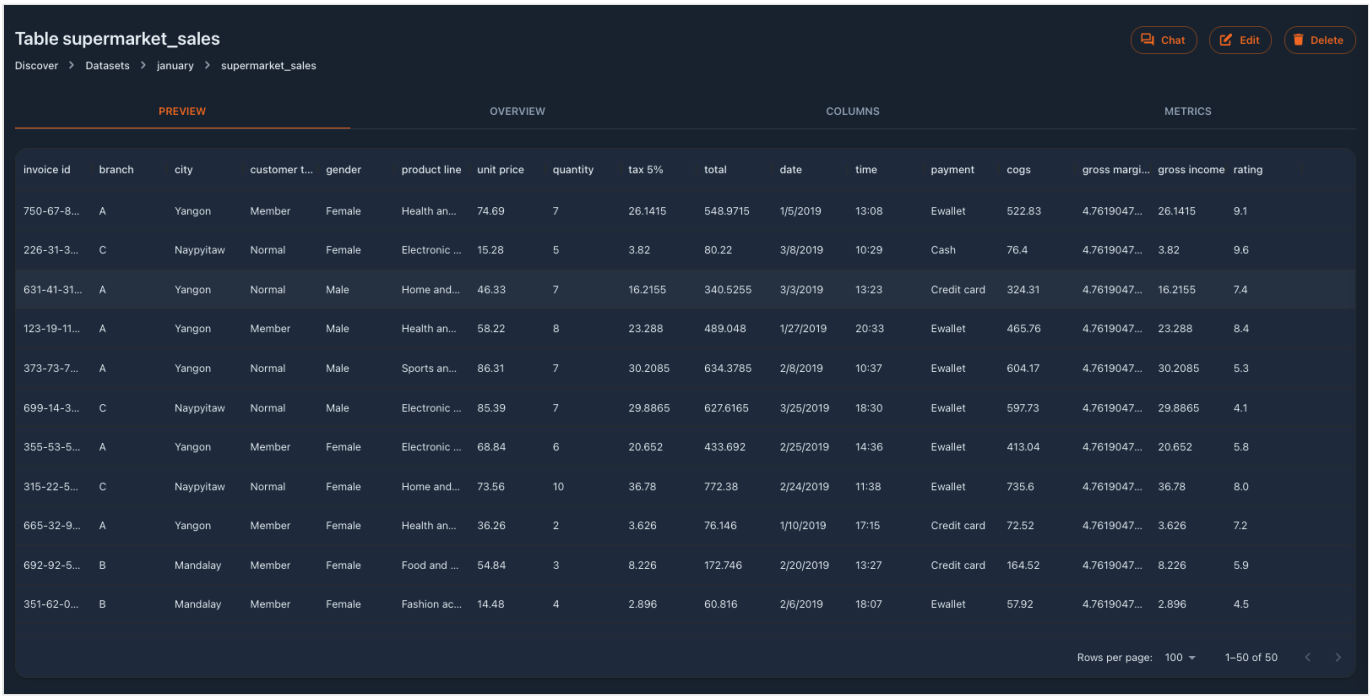
Add or edit table metadata

Edit your table metadata by clicking on the **Edit** button.

Preview data

Data preview gives you the ability to preview a subset of the data available on data.all. Preview feature is available for data you own or data that's shared with you.

Just select a table and in the **Preview** tab you will find the results of an SQL select subset of the table.



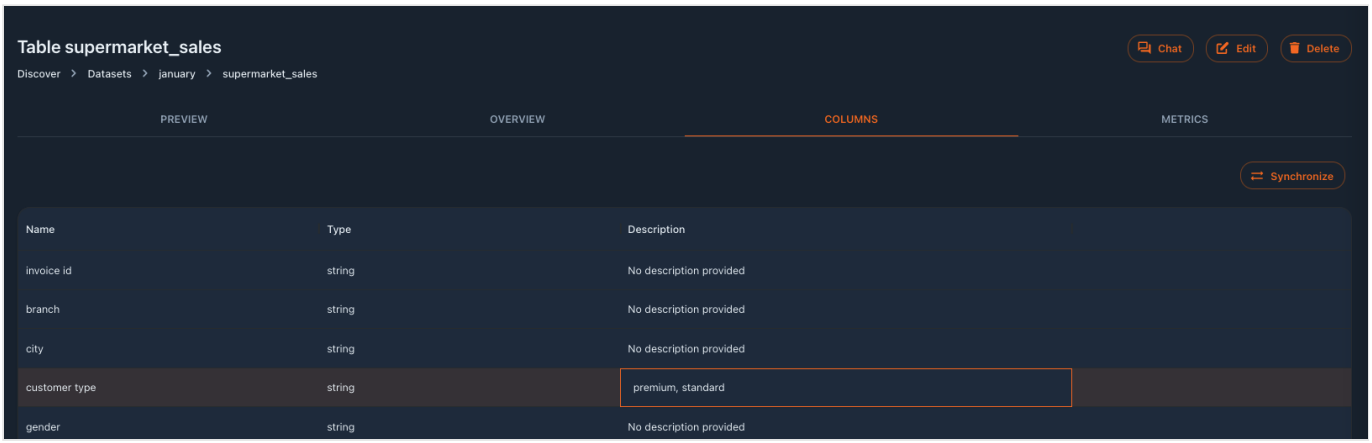
Leave a message in Chat

As with datasets, in the **Chats** button users can interact and leave their comments and questions on the Table Chat.

Add column description

Metadata makes more sense when columns description fields are not empty. With data.all you can add columns description and avoid the pain of figuring out fields purpose.

Select one table and in the **Columns** tab, directly type the description in the Description column as shown in the picture.



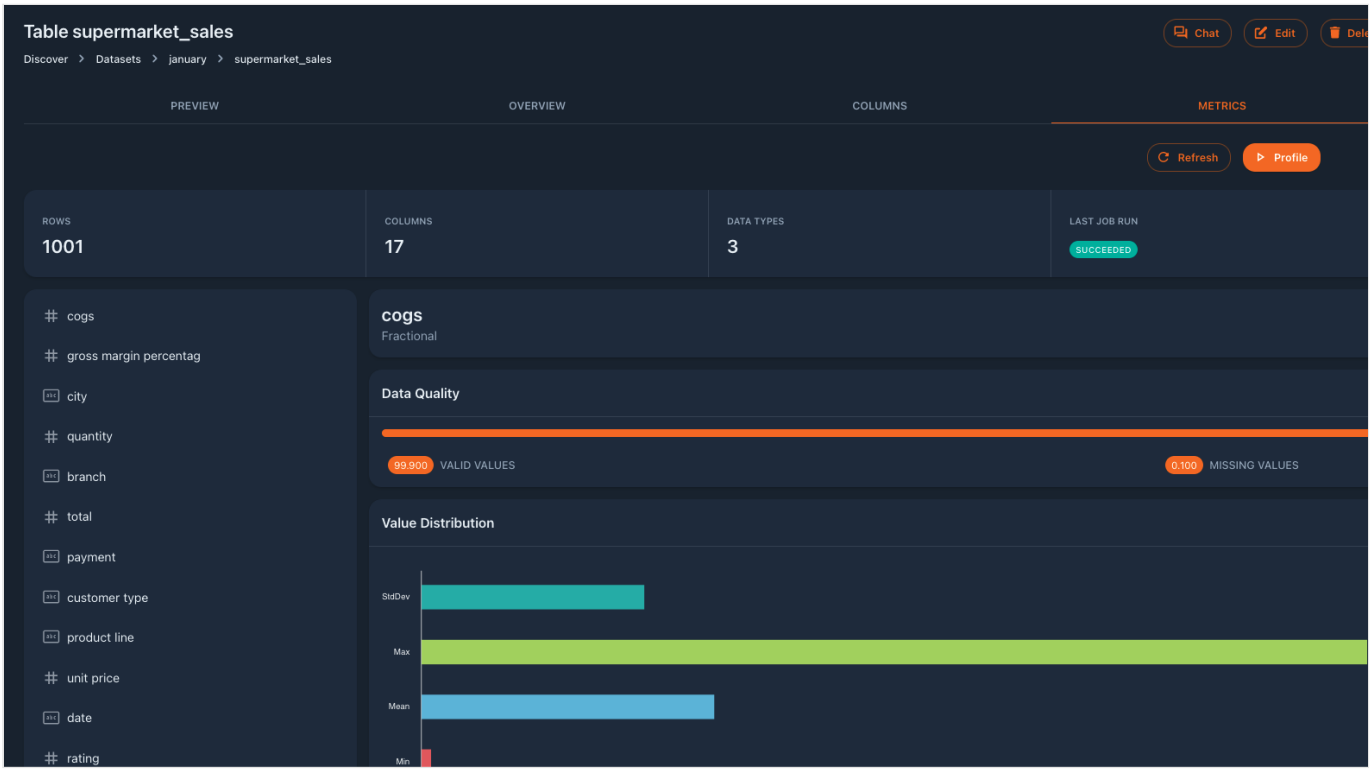
Profile data

Data profiling refers to the process of examining, analyzing, and reviewing the data available in the source by collecting statistical information about the data set's quality and hygiene. This process is called also data archaeology, data assessment, data discovery, or data quality analysis. Data profiling helps in determining the accuracy, completeness, structure, and quality of your data.

Data profiling in data.all involves:

- Collecting descriptive statistics like minimum, maximum, mean, median, and standard deviation.
- Collecting data types, along with the minimum and maximum length.
- Determining the percentages of distinct or missing data.
- Identifying frequency distributions and significant values.

By selecting the **Metrics** tab of your data table you can run a profiling job (click in the **Profile** button) , or view the latest generated data profiling metrics:

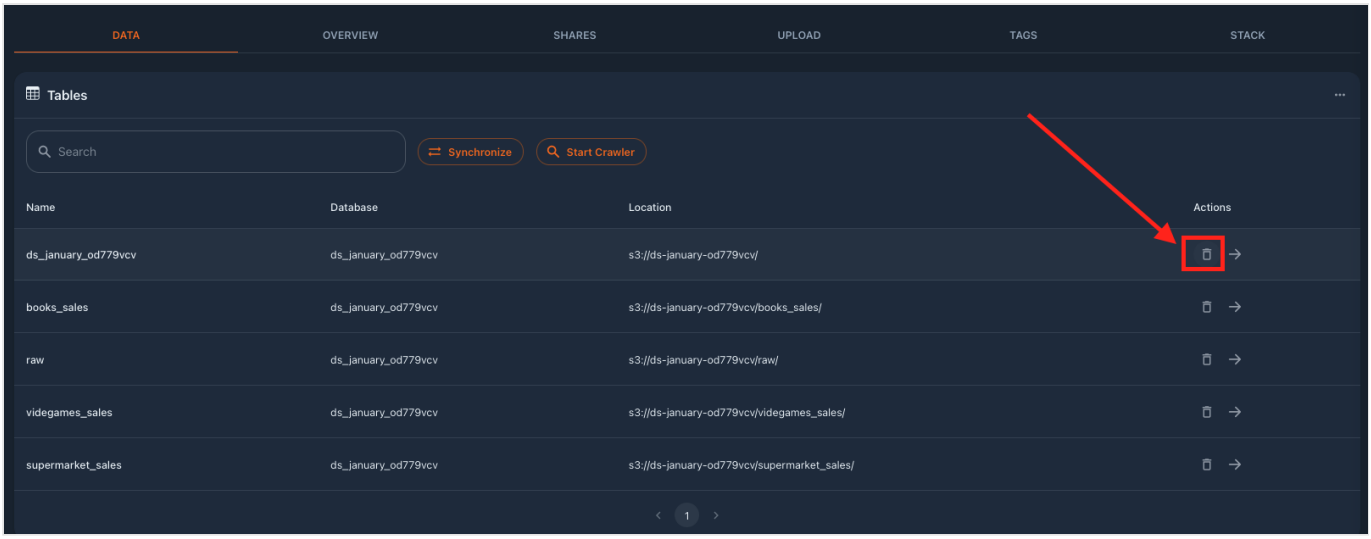


> **Profiling Job Prerequisite**
>
> Before running the profiling job you will need to ensure that the **default** Glue Database exists in the AWS Account where the data exists (by default this database exists for new accounts). This is required to enable the Glue profiling job to use the metadata stored in the Glue Catalog.

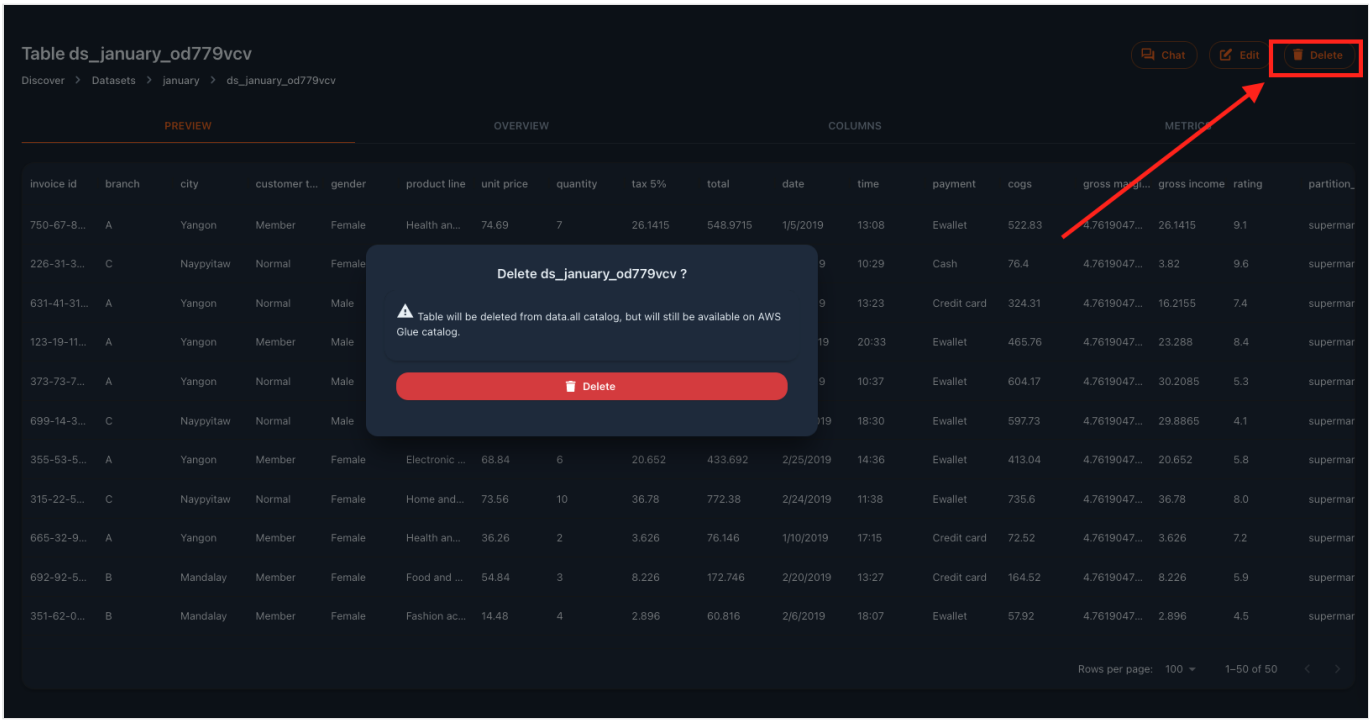
Delete a table

Deleting a table means deleting it from the data.all Catalog, but it will be still available on the AWS Glue Catalog. Moreover, when data owners delete a table, they are **not** deleting its data from the dataset S3 bucket. Teams with shared access to the dataset cannot delete tables or folders, even if they are shared.

It is possible to delete a table from the dataset **Tables** tab with the trash can icon next to each of the tables in the Actions column.



Another option is to go to the specific table (on the above picture click on the arrow icon next to the trash can icon). Click on the **Delete** button in the top right corner and confirm the deletion.

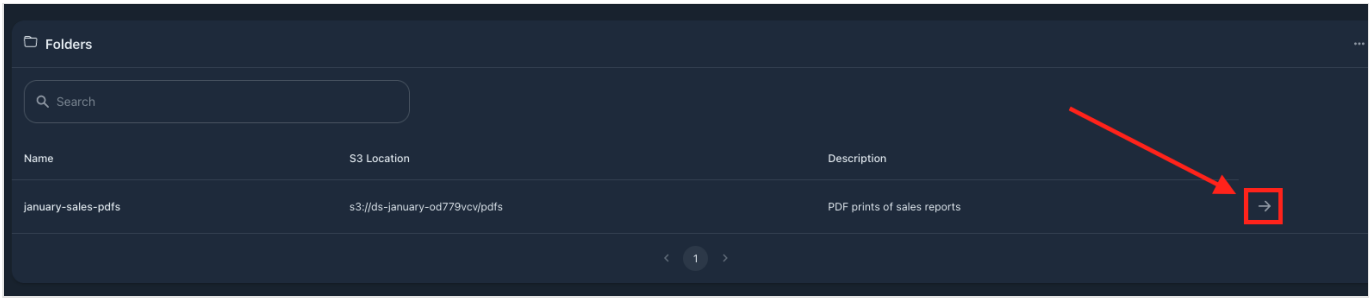


An error occurred (ResourceShared) when calling DELETE_DATASET_TABLE operation: Revoke all table shares before deletion

To protect data consumers, if the table is shared you cannot delete it. The share requests to the table need to be revoked before deleting the table. Check the Shares section to learn how to grant and revoke access.

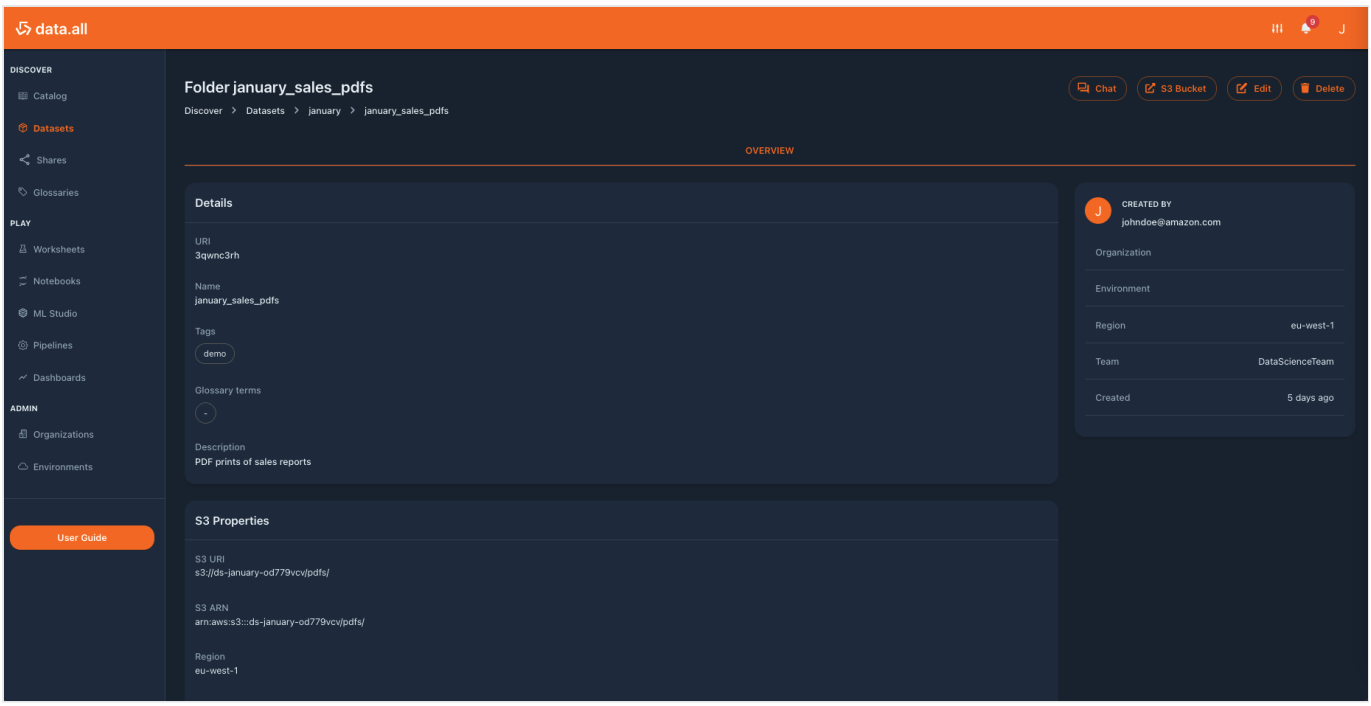
4.2.2 S3 Folders

To open the Folder window you can either find your chosen folder in the Catalog or navigate to the dataset and then in the **Folders** tab click on the arrow in the Actions column of your folder:



Check folder and S3 metadata

The **Overview** tab of the folder contains folder metadata: - URI: unique folder identifier - Name: name of the folder, it is made out of the dataset name concatenated with the S3 prefix - Tags - Glossary terms - Description - Organization, Environment, Region, Team: inherited from the dataset - Created: creation time of the table



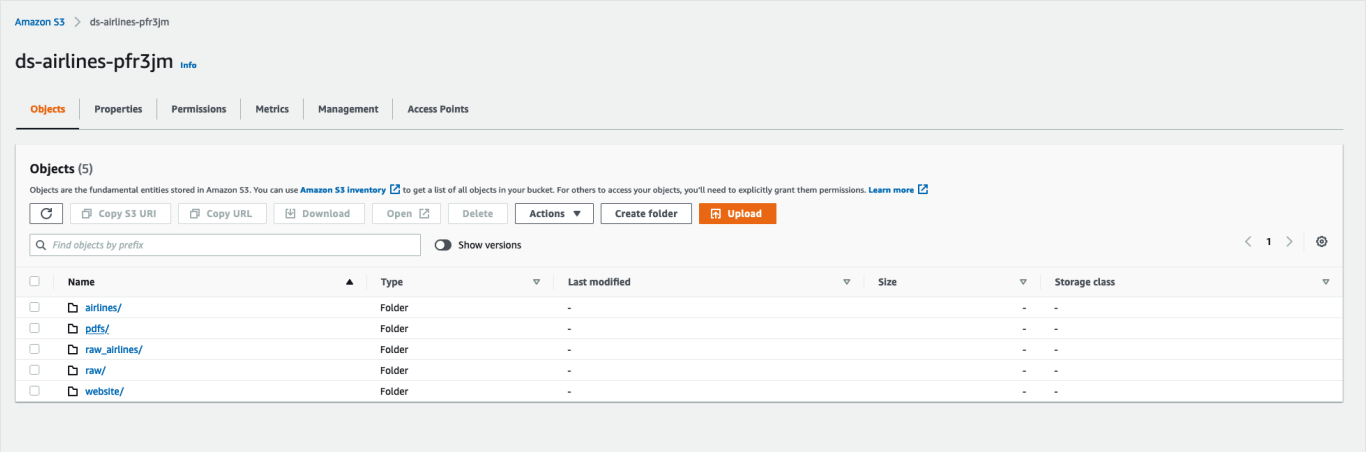
Add or edit table metadata

Edit your folder metadata by clicking on the **Edit** button.

> **Description, Tags and Glossary terms are not inherited**
>
> Careful, those 3 fields are not synced with their dataset metadata. Just click on the **Edit** button of the folder to complete any missing information. This is especially useful to improve Catalog search of your folders.

Check the content of your folder

To check what kind of files does our prefix content, we can access the AWS S3 console on the **S3 Bucket** button of the Folder **Overview** tab.



Leave a message in Chat

Exactly the same as with tables. Allow your teams to discuss directly on the Folder Chat.

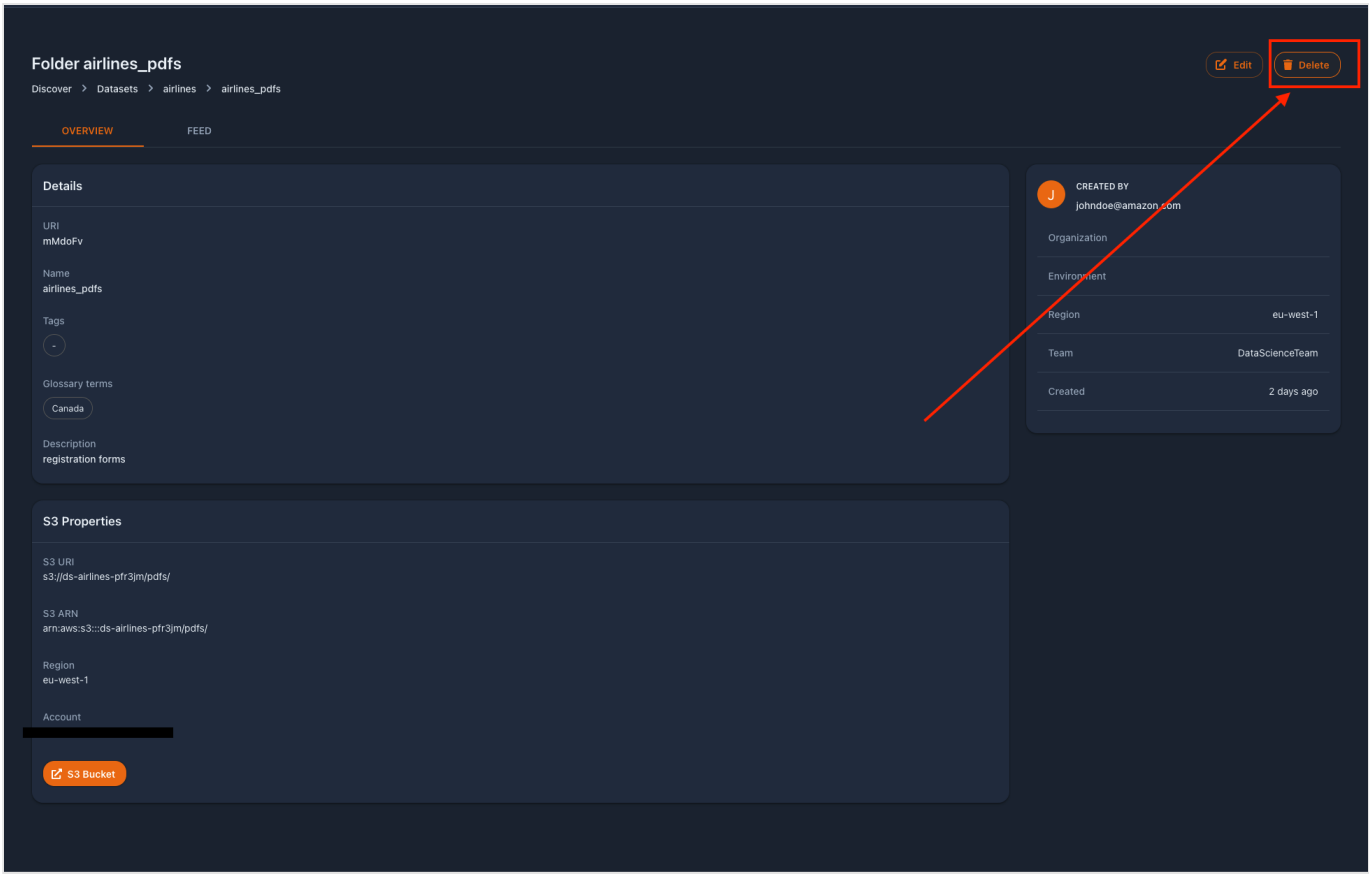
Delete a folder

Deleting folders is analogous to deleting tables. Deletion means deletion from the data.all Catalog and the content of the S3 prefix remains in the dataset S3 bucket. Only dataset owners can delete dataset folders.

The steps to delete a folder are exactly the same as with tables. You can either go to the dataset and in the **Folders** tab click on the can trash icon on the Actions column of the selected folder; or you can navigate to the Folder and click on the **Delete** button.

An error occurred (ResourceShared) when calling DELETE_DATASET_FOLDER operation: Revoke all folder shares before deletion

To protect data consumers, if the table is shared you cannot delete it. The share requests to the table need to be revoked before deleting the table. Check the Shares section to learn how to grant and revoke access.


Folder airlines_pdfs
Discover > Datasets > airlines > airlines_pdfs
Edit
Delete
OVERVIEW
FEED
Details
URI
mMdoFv
Name
airlines_pdfs
Tags
-
Glossary terms
Canada
Description
registration forms
CREATED BY
johndoe@amazon.com
Organization
Environment
Region
eu-west-1
Team
DataScienceTeam
Created
2 days ago
S3 Properties
S3 URI
s3://ds-airlines-pfr3jm/pdfs/
S3 ARN
arn:aws:s3:::ds-airlines-pfr3jm/pdfs/
Region
eu-west-1
Account
S3 Bucket

4.3 Redshift Datasets

Data producers can import their Redshift tables into data.all and make them discoverable and shareable in an easy and secure manner.

In data.all we will work with 2 main constructs:

- **Redshift Connections**, which store the necessary metadata to connect to a Redshift namespace
- **Redshift Datasets**, group of tables imported into data.all Catalog using a data.all Redshift Connection.

4.3.1 Redshift Connections

Data.all Redshift Connections are metadata used by data.all and by data.all users to connect to Redshift namespaces.

1) Both Redshift Serverless and Provisioned clusters are supported 2) Connections use AWS Secrets Manager secrets or Redshift users to connect to the namespace. Check the documentation to understand each mechanism. Additional connection mechanisms might be considered in the future.

Connection Types

Here is a table to summarize the 2 different types of connections, keep reading to understand each type in depth.

Connection type	Purpose in data.all	Redshift permissions required	Grantable permissions
`DATA_USER`	Import Redshift Datasets	READ Redshift tables	None
`ADMIN`	Process Redshift share requests	MANAGE Redshift datashares	`Use Connection in share request`

DATA USER CONNECTIONS

`DATA_USER` connections are used to IMPORT Redshift dataset into data.all. The Redshift user used in the connection should have READ permissions to the tables to be imported.

Recommendations

In the following example there are 2 teams, `ClusterAdminTeam` and `MarketingTeam`. Both have been onboarded to data.all and can log in to the UI. The `ClusterAdminTeamA` is a team that administrates a Redshift cluster `RedshiftClusterA` in the AWS Account of a data.all environment `EnvironmentA`. The `MarketingTeam` works in this cluster creating some tables `marketingTables`

It has been agreed that `marketingTables` should be imported to data.all. **Which type of connection should we use?** We need to create a `DATA_USER` connection with a user that can read `marketingTables`.

And, which team should own the connection? This depends on the data ownership requirements of your teams. The connection owners will be able to import the Redshift dataset, becoming the dataset owners. The Redshift dataset owners are in charge of managing the metadata of the dataset, editing/deleting and approving/revoking share requests. If in your organization the `ClusterAdminTeamA` is in charge of managing all operations on the datasets then they should be the owners of the connection. If on the contrary, your organization has more distributed control over the operations on the data.all dataset, then the `MarketingTeam` should own the connection.

ADMIN CONNECTIONS

`ADMIN` connections are used by data.all to process Redshift data share requests. The Redshift user used in the connection should have enough permissions to MANAGE DATASHARES in the cluster.

Recommendations

We will continue the example of DATA_USER connections. Let's imagine that the `MarketingTeam` has happily imported the `marketingTables` Dataset and it is now published in the data.all Catalog. In another AWS Account `AccountB`, linked to data.all as `EnvironmentB`, the `ResearchTeam` works in a Redshift cluster `RedshiftClusterB` managed by `ClusterAdminTeamB`. The `ResearchTeam` wants to request access to `marketingTables`.

Which type of connection should we use? We need to create an `ADMIN` connection with a user that can manage Redshift datashares in both the `RedshiftClusterA` and `RedshiftClusterB`.

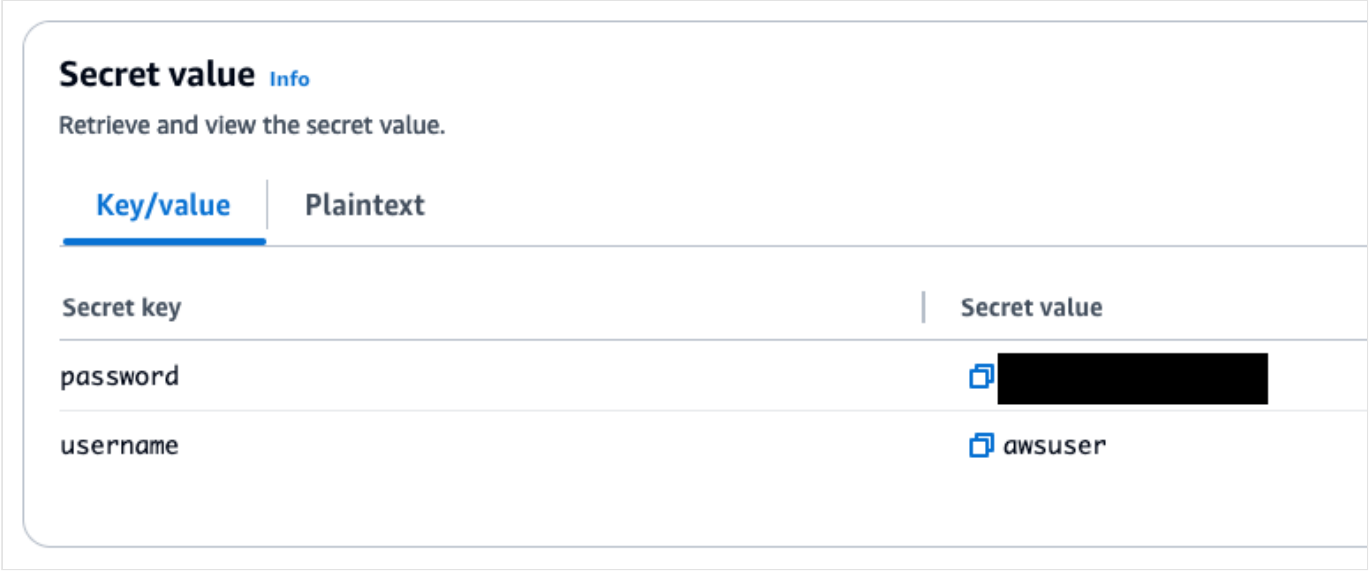
And, which team should own the connection? The Connection owners should be teams with administrative rights over the clusters. In this case the `ClusterAdminTeamA` and `ClusterAdminTeamB` should own the `AdminConnectionA` and `AdminConnectionB` respectively.

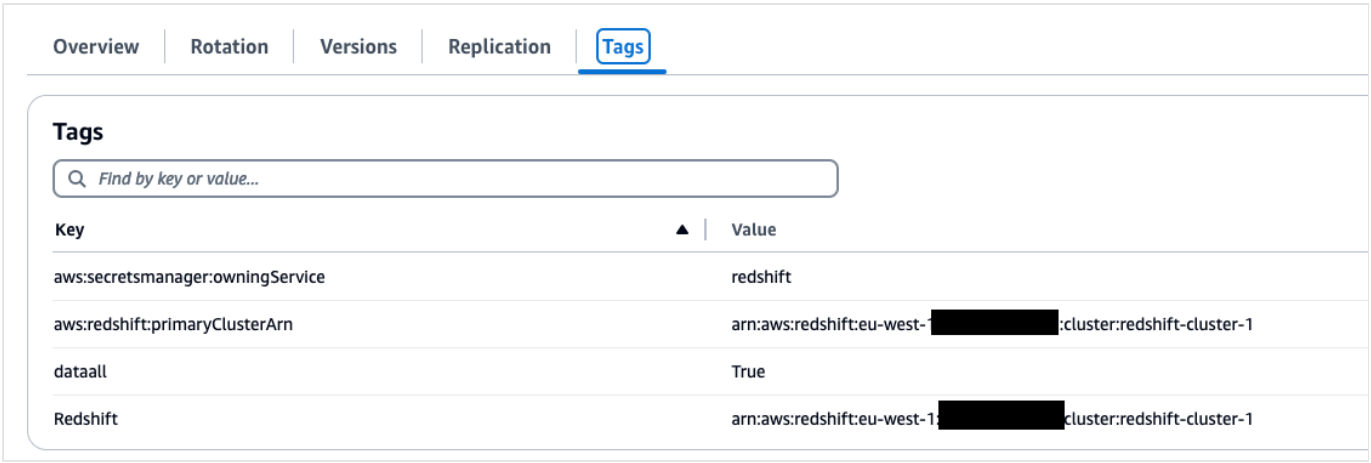
How can the `ResearchTeam` use the `AdminConnectionB`? The `ClusterAdminTeamB` needs to grant "Use Connection in share request" permissions for the connection `AdminConnectionB` to the `ResearchTeam`. After that the `ResearchTeam` will be able to open share requests, but they won't be able to edit/delete the `AdminConnectionB`. The steps to grant these permissions are explained in the Update Connection permissions subsection.

Create a Redshift Connection

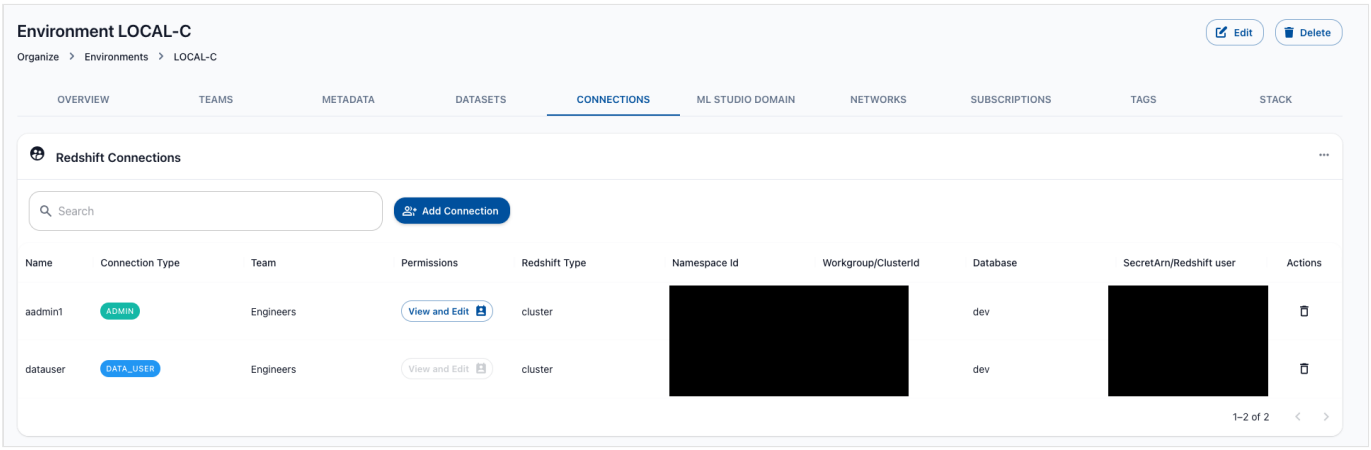
data.all requires Redshift clusters and users to be managed by a dedicated team and infrastructure created outside of data.all. For this reason, data.all will work "importing" existing infrastructure and users, requiring the following information on import:

- Redshift Serverless namespace/workgroup or Provisioned cluster: the user creating the connection must know the `namespace ID` and the `workgroup` for Redshift Serverless or the `cluster ID` for the case of Redshift Provisioned clusters.
- Redshift user: Redshift administrators manage Redshift users outside of data.all.
- Connection details:
- Redshift user (only valid for Provisioned clusters): data.all will generate a temporary password to connect to the database. In this case no password or secret needs to be provided to data.all.
- AWS Secrets Manager Secret (recommended): the username and password for the Redshift user can be stored in a Secret that **MUST** be tagged with 2 tags. Check the pictures below to see how it should look in the AWS Console.
  - tagKey: dataall, tagValue: True - Needed for data.all to be able to access the Secret
  - tagKey: Redshift, tagValue: Any - Needed by Redshift to use as connection

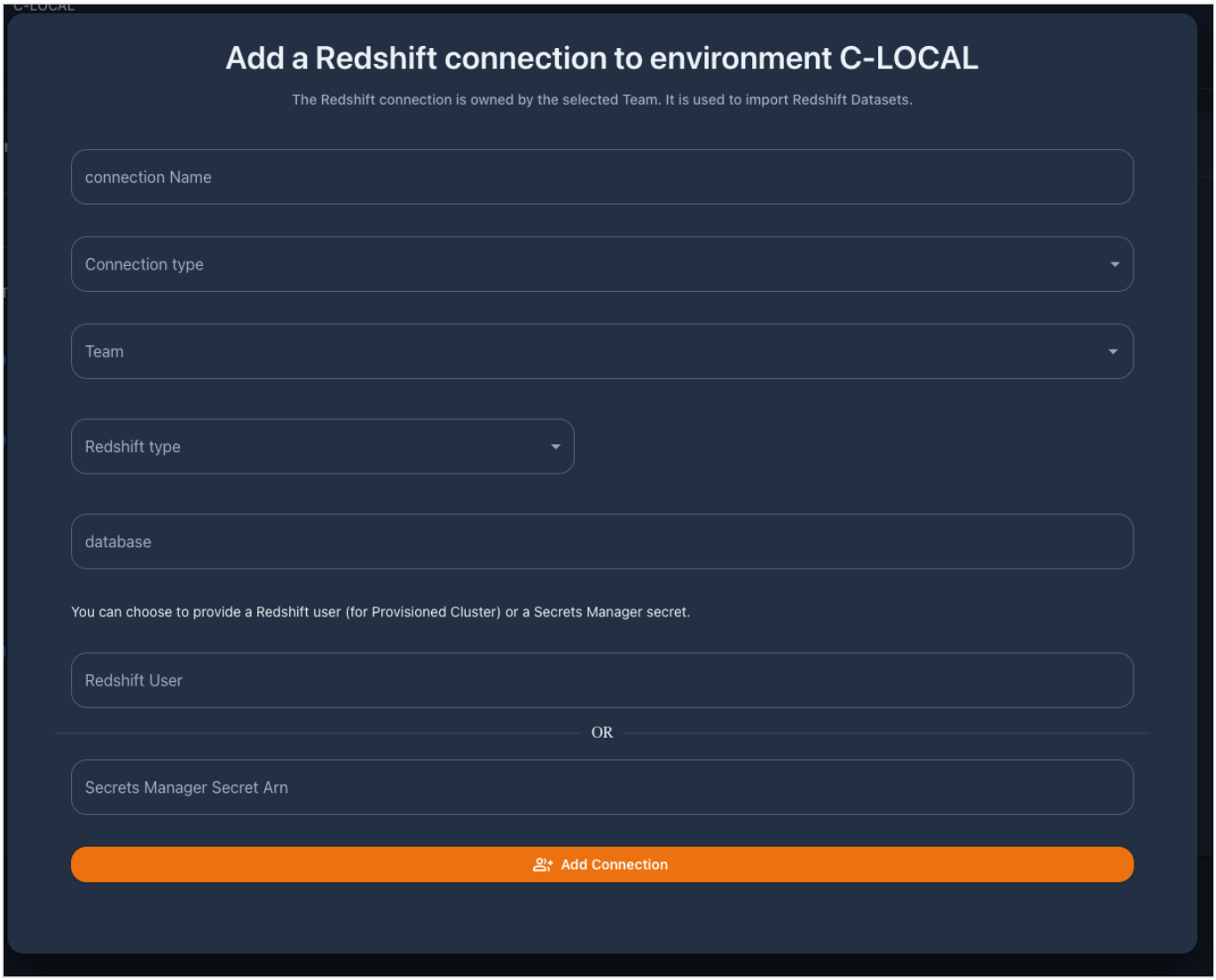




Redshift Connections are created inside the Environment view. Select an Environment and navigate to the **Connections** tab. Here you can click on the **Add Connection** button to create a new Redshift Connection.



Then, fill in the following form:


Add a Redshift connection to environment C-LOCAL
The Redshift connection is owned by the selected Team. It is used to import Redshift Datasets.
connection Name
Connection type
Team
Redshift type
database
You can choose to provide a Redshift user (for Provisioned Cluster) or a Secrets Manager secret.
Redshift User
OR
Secrets Manager Secret Arn
Add Connection

Field	Description	Required	Editable	Example
Connection name	Name of the Redshift connection	Yes	No	main-cluster-admin
Connection type	Level of access of the connection. It can either be `ADMIN` or `DATA_USER`. See definitions above.	Yes	No	`ADMIN`
Team	Team that owns the connection. This team is the only team that can use this connection to import datasets.	Yes	No	DataScienceTeam
Redshift type	Type of Redshift Namespace. It can either be `serverless` or `cluster`.	Yes	No	`serverless`
Cluster Id	If the Redshift type is `cluster`, we need to introduce the cluster Id.	Yes	No	redshift-cluster-1
Namespace Id	If the Redshift type is `serverless`, we need to introduce the namespace Id.	Yes	No	0000000-0000-0000-0000-000000000000
Workgroup	If the Redshift type is `serverless`, we need to introduce the workgroup.	Yes	No	workgroup1
Database	Database that we will connect to inside the cluster.	Yes	No	dev
Redshift User	Only available for `cluster` Redshift type. This is the user	Yes	No	user1

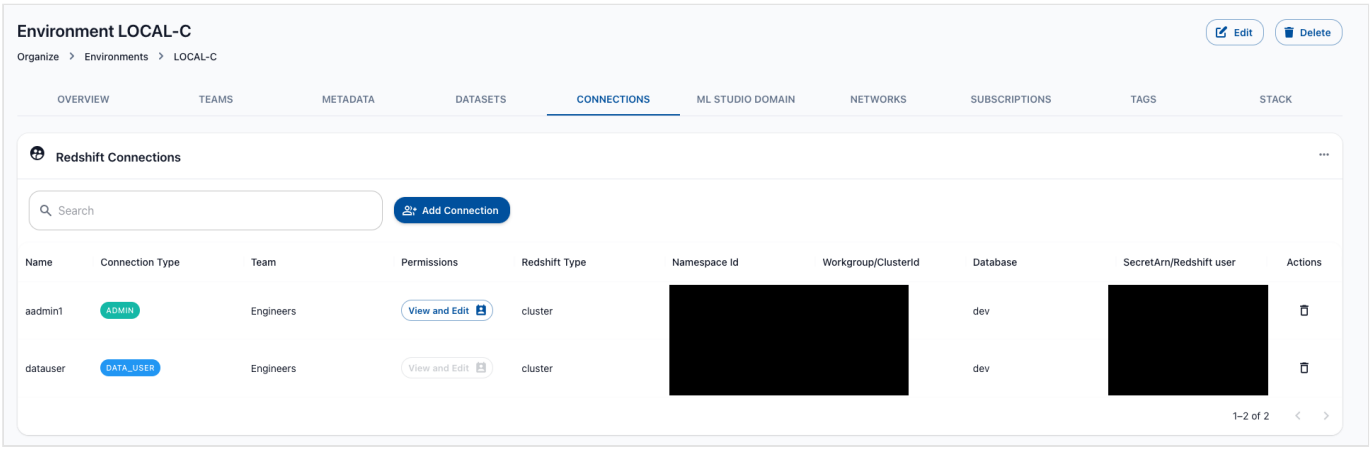
Field	Description	Required	Editable	Example
Secret Arn	Secrets Manager secret arn storing username and password for the connection. See pre-requisites section above.	Yes	Yes	arn:aws:secretsmanager:eu-west-1:000000000000:redshift!redshift-cluster-1-awsuser

Data.all will verify the connection upon creation. If the database does not exist or if the connection details are not accessible or do not correspond to cluster it will notify the user in the error banner.

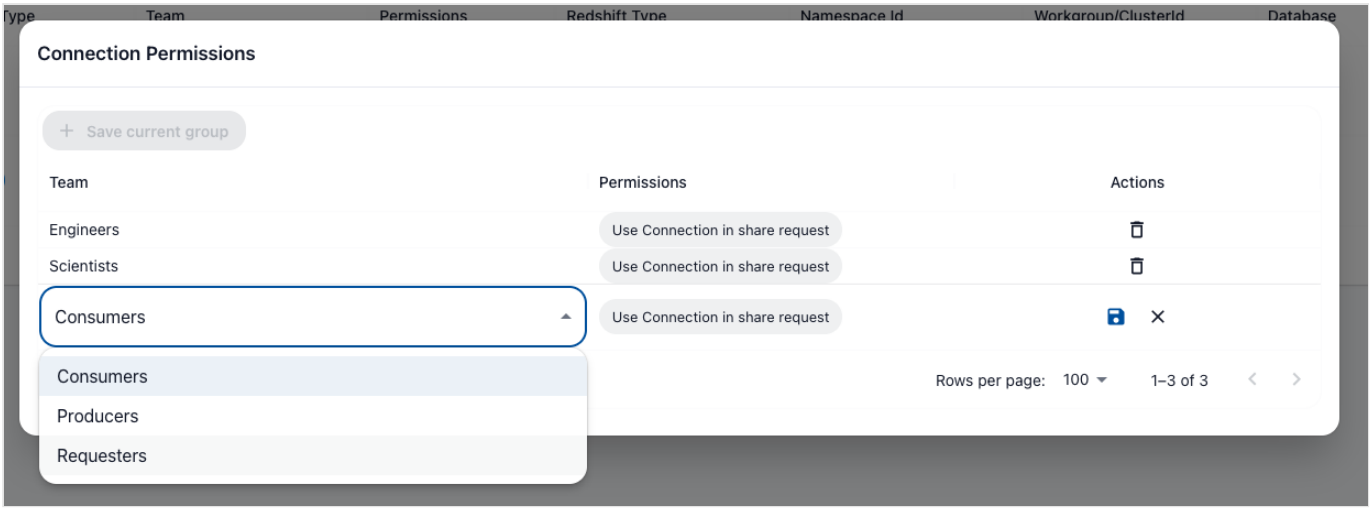
Update Connection permissions

The owners of an `ADMIN` connection can grant other teams permissions to use the Connection in a share request. At the moment this is the only type of permission that can be granted and it is only available for `ADMIN` connections. Check the section on Connection types to understand the usage of this permission.

To update the permissions, select your environment and navigate to the Connections tab. You will see that the `ADMIN` connections have a button in the Permissions tab called `View and Edit` (it is disabled for `DATA_USER` connections).



If you click on the button the following window will open. Here you can press the `Add group` and select a group that will get "Use Connection in share request" permissions to the connection. Do not forget to click on the save icon to save the permissions.

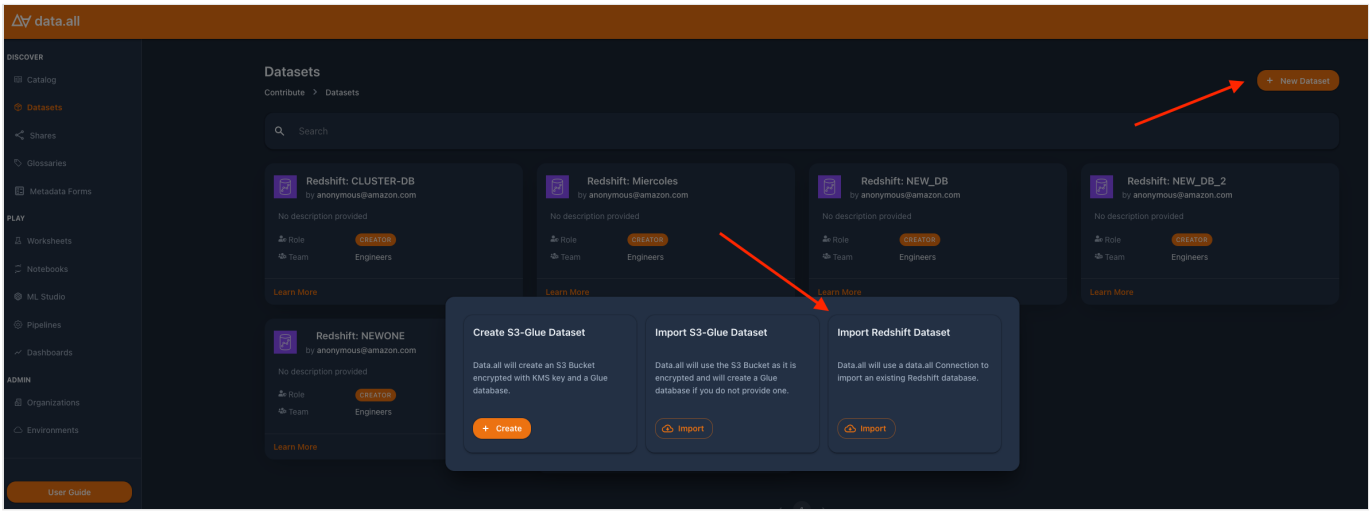


Delete a Connection

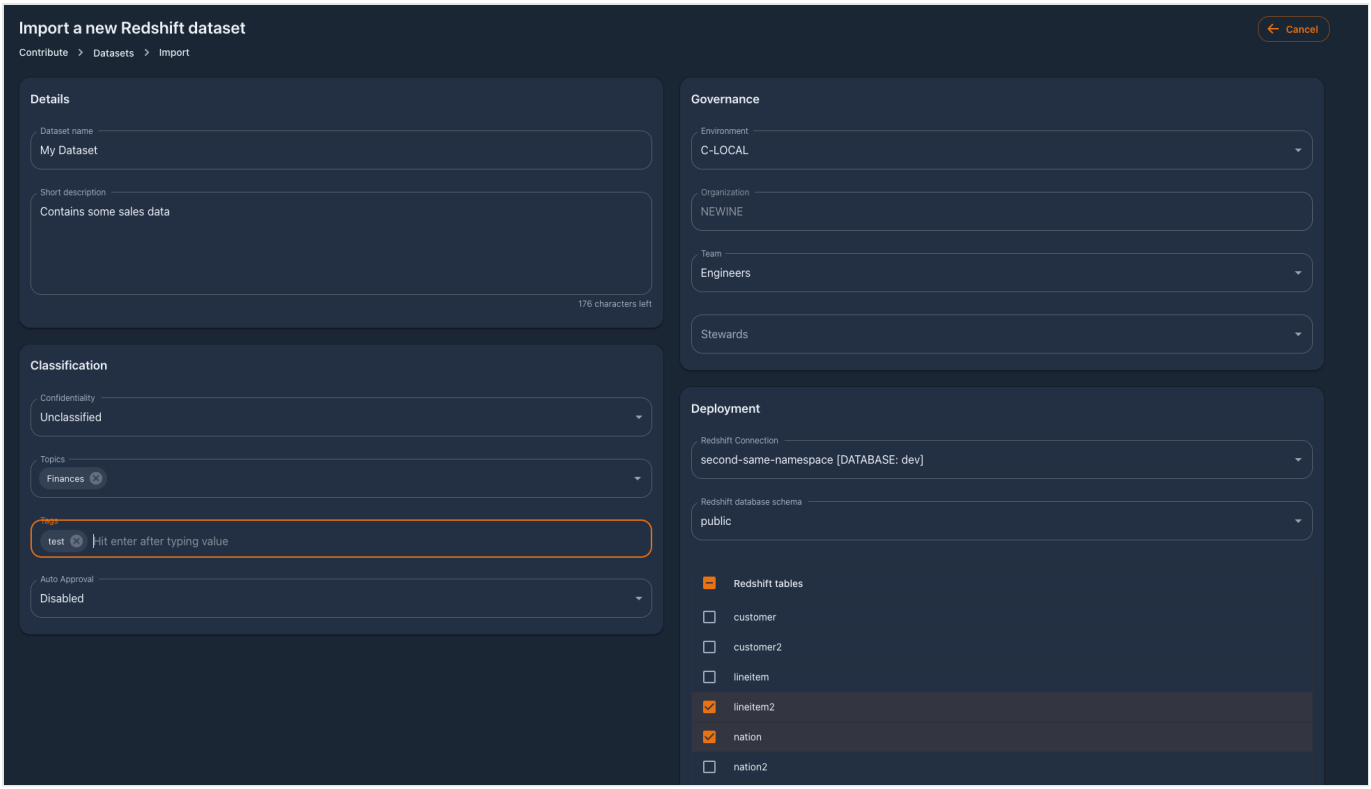
To delete a connection, click on the trash icon next to the item in the Actions column. If the Connection has been used to import datasets it cannot be removed until all associated datasets are deleted.

4.3.2 NEW Import a Redshift Dataset

To create a new dataset, navigate to the Datasets view and click on **New Dataset**. A window like the one in the picture will allow you to select the type of Dataset you want to create or import. In this case you need to select the Import Redshift Dataset option.



Next, fill in the creation form with the Dataset details. To import Redshift Datasets, only connections of the type `DATA_USER` can be used. Therefore, data.all will list the Redshift `DATA_USER` connections owned by the selected team in the environment and fetch the schemas and tables from Redshift. It is possible to select all tables or a subset of tables as appears in the picture.



Generic dataset fields

Field	Description	Required	Editable	Example
Dataset name	Name of the dataset	Yes	Yes	AnyDataset
Short description	Short description about the dataset	No	Yes	For AnyProject predictive model
Environment	Environment (mapped to an AWS account)	Yes	No	DataScience
Organization (auto-filled)	Organization of the environment	Yes	No	AnyCompany EMEA
Team	Team that owns the dataset	Yes	No	DataScienceTeam
Stewards	Team that can manage share requests on behalf of owners	No	Yes	FinanceBITeam, FinanceMgmtTeam
Confidentiality	Level of confidentiality: Unclassified, Oficial or Secret	Yes	Yes	Secret
Topics	Topics that can later be used in the Catalog	Yes, at least 1	Yes	Finance
Tags	Tags that can later be used in the Catalog	Yes, at least 1	Yes	deleteme, ds
Auto Approval	Whether shares for this dataset need approval from dataset owners/ stewards	Yes (default `Disabled`)	Yes	Disabled, Enabled

Redshift Dataset fields

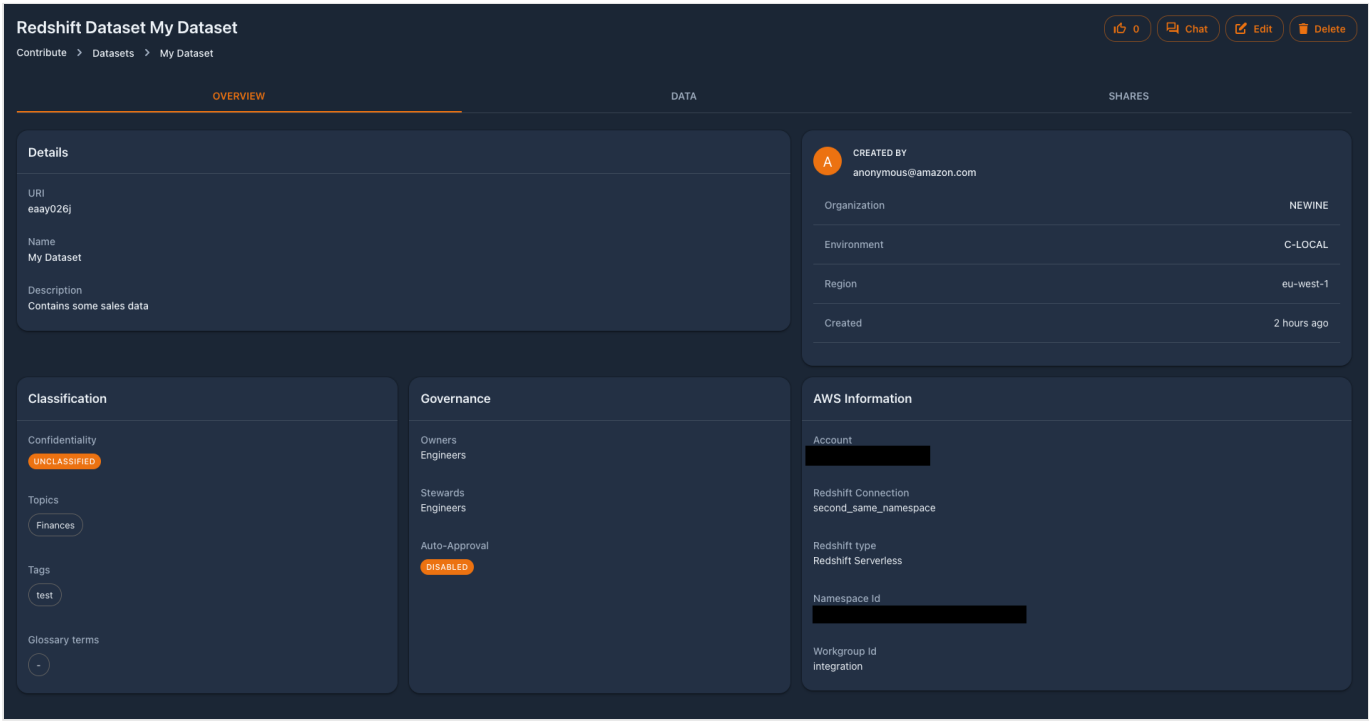
Field	Description	Required	Editable	Example
Redshift Connection	Name of the Redshift connection used to read the Redshift tables. Only `DATA_USER` connections can be used to import Datasets.	Yes	No	main-cluster-userA
Redshift database schema	Name of the Redshift schema where the tables are stored	Yes	No	public
Redshift tables	List of tables to be imported. They can be added at a later stage	No	Yes	customer, orders

Once a Redshift dataset has been imported, the dataset and its imported tables can be searched by any user in the Catalog.

4.3.3 Navigate Redshift dataset tabs

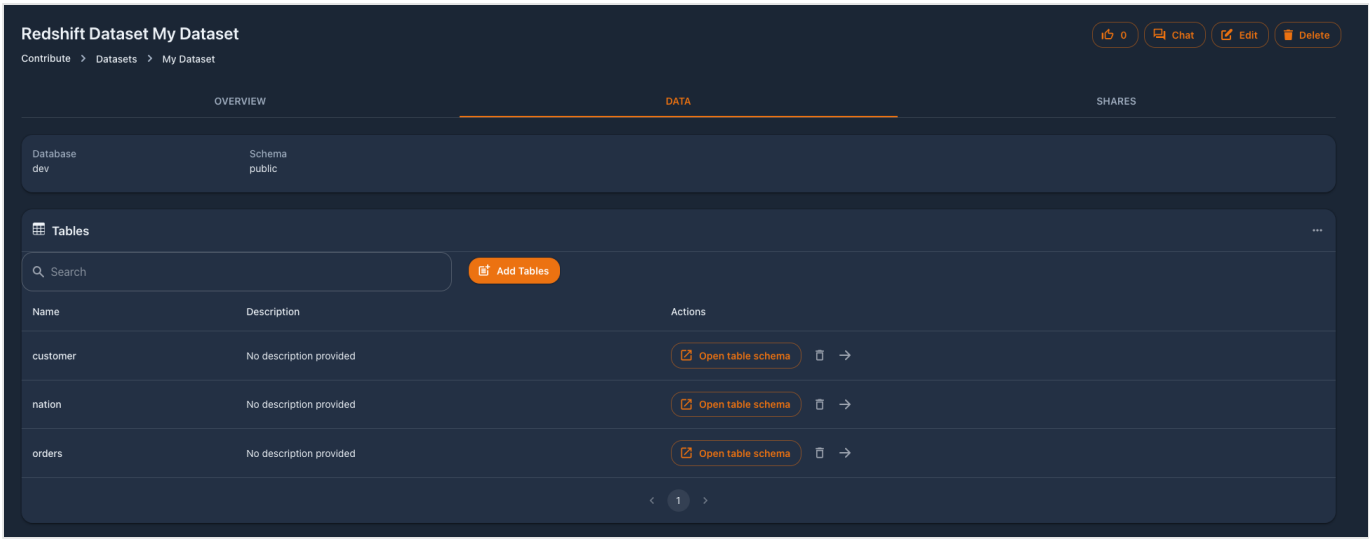
Overview

This tab includes meaningful metadata about the dataset and the Redshift connection used.



Data

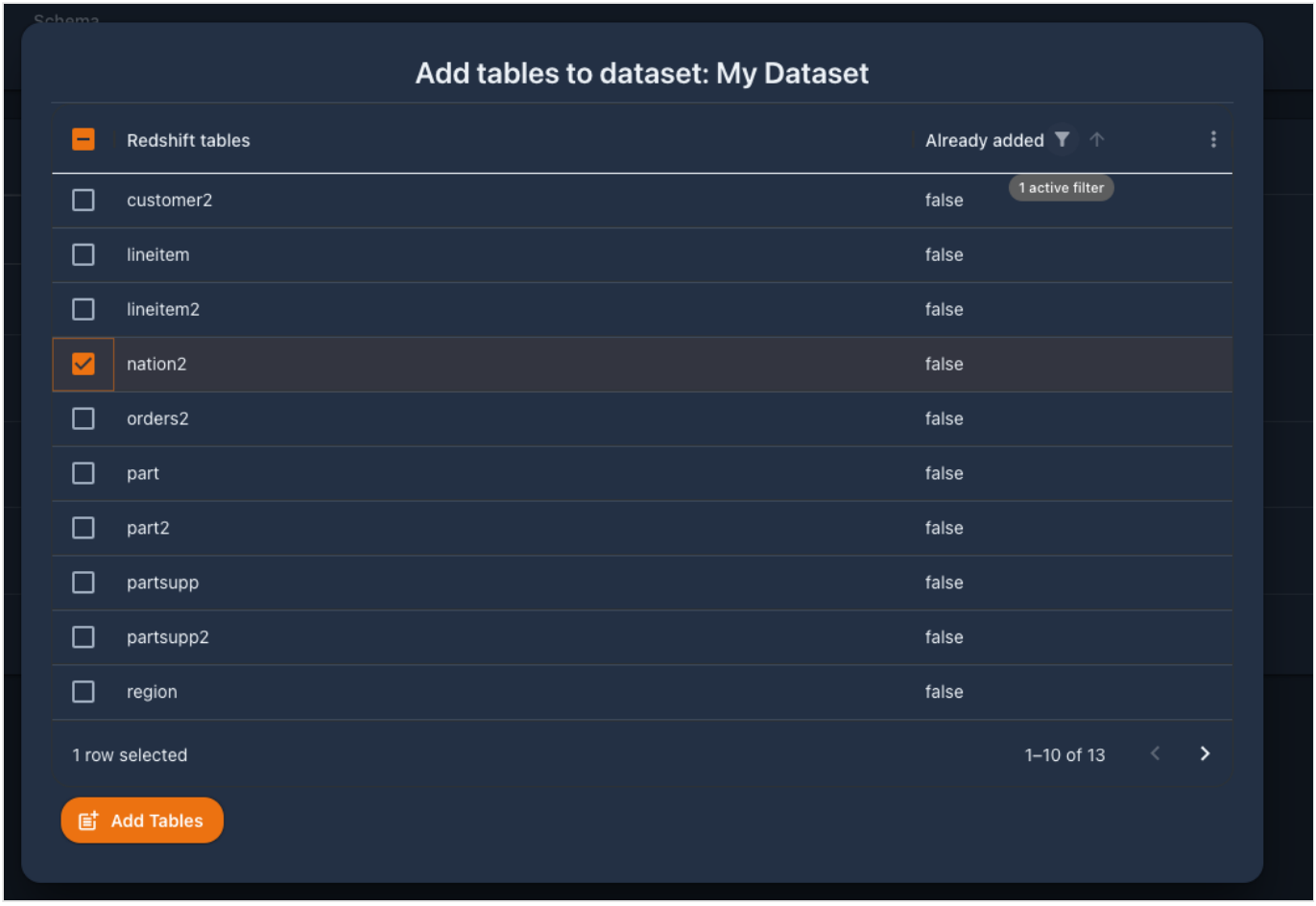
This tab shows the Redshift database, schema and tables imported. From here we can add, edit, delete and see the details of a table.



Shares Show a list of the share requests for this Dataset. It is possible to verify the health and reapply shares for the entire Dataset

Manage Redshift Tables

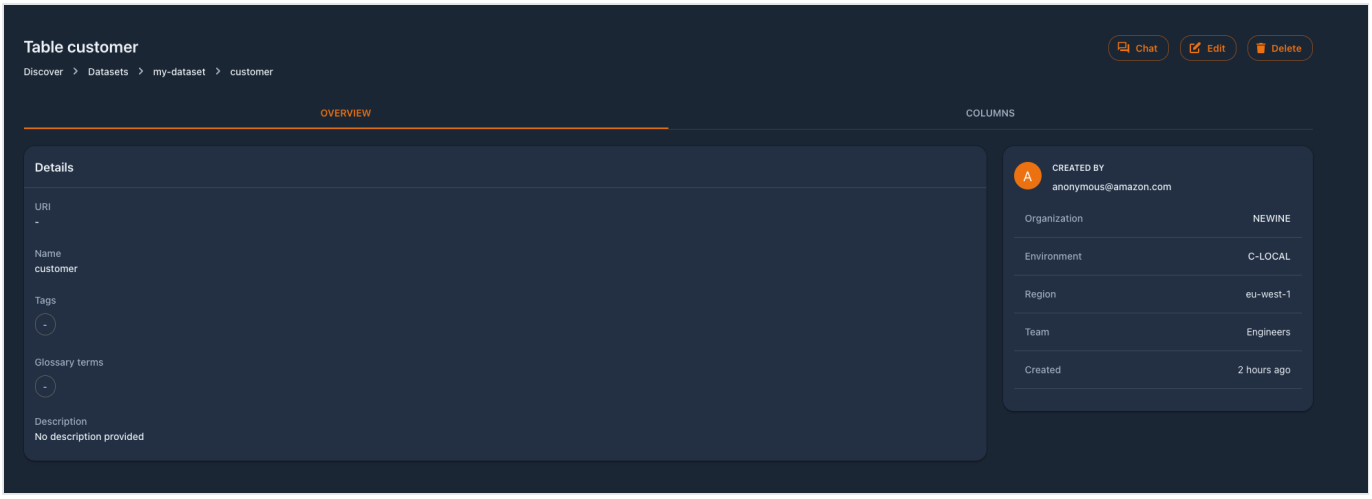
Add tables



View and edit tables

We can view the schema of a table directly from the Data tab, by clicking on the **Open table schema** button.

We can also see a full view of the table by selecting the arrow in the Actions column. A new window for the table will open. In this view we can edit the metadata of the table in data.all (Tags, glossary, description) and we can see the schema in full-width in the Columns tab.



Delete a table

We can delete Redshift tables by clicking on the trash icon next to the table we want to "un-import". Un-import is a better word to describe what will happen: the metadata of the table will be deleted from data.all Catalog, but the original Redshift table still exists in Redshift.

Dataset owners need to revoke access to the table before deleting. Data.all prevents deletion of a table if there are share requests currently sharing the table.

4.3.4 Edit and update a dataset

Data owners can edit the dataset by clicking on the **edit** button, editing the editable fields and saving the changes.

4.3.5 Delete a dataset

To delete a dataset, in the selected dataset window click on the **delete** button in the top-right corner. data.all Redshift Datasets don't deploy any CloudFormation stack, no additional resources need to be cleaned up. The original Redshift tables will still exist in Redshift.

In the same way as it happens with single tables, Dataset owners need to revoke access to all tables before deleting. Data.all prevents deletion of a dataset if there are share requests currently sharing any dataset table.

4.4 Centralized Catalog and glossaries

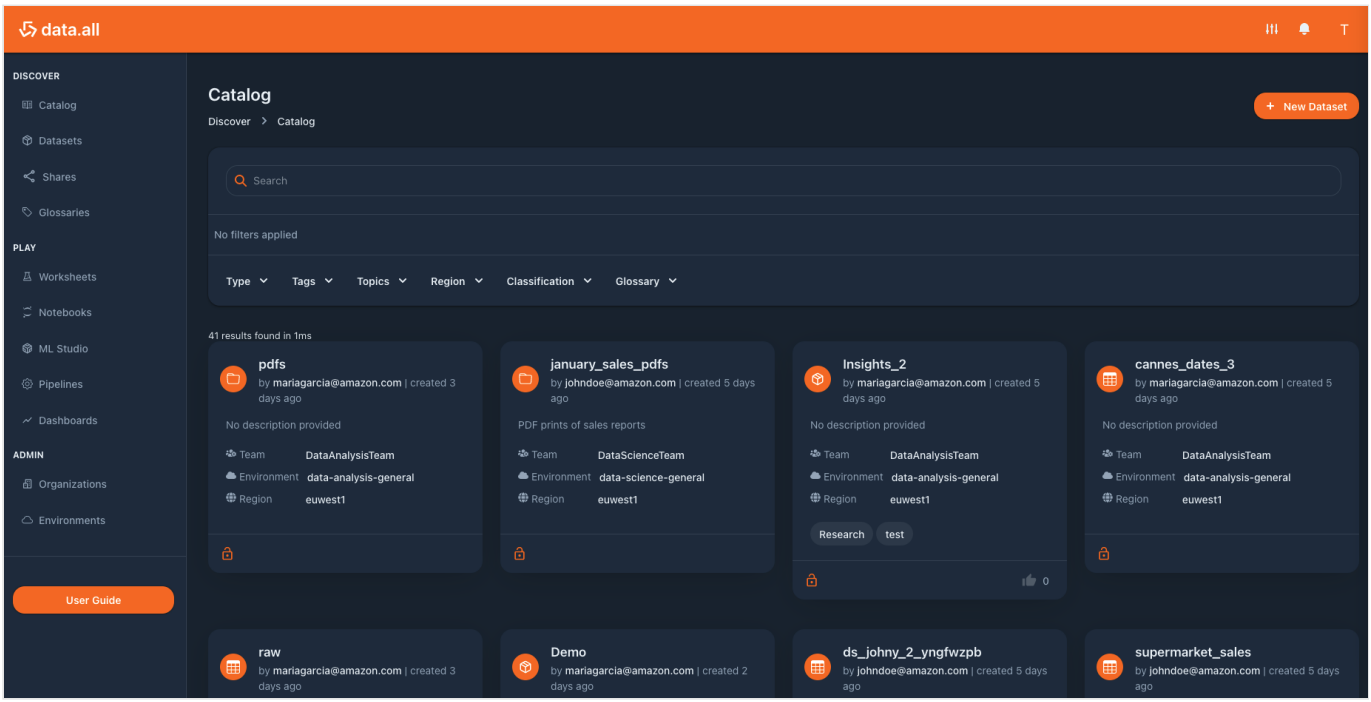
4.4.1 Catalog

In the Catalog we have a record with metadata for each dataset, table, folder and dashboard in data.all. Users come to this centralized Catalog to search and find data owned by other teams. Once users find a data asset they are interested in, they will create a Share request.

How do users find the data that they need?

Data needs to be discoverable, for this reason data.all Catalog offers a variety of filters that use business context to improve your search:

- **Type of data**: `dataset`, `table`, `folder` and/or `dashboard`
- **Tags**: tags of the data asset.
- **Topics**: filter by general topics created by the user.
- **Region**: AWS region where the data asset is located.
- **Classification**: `unclassified`, `official` and/or `secret`
- **Glossary**: filter datasets by the glossary terms created by users. This helps in two ways: It lets you narrow down results quickly using granular glossary terms like "sales", "profit", etc. Traditionally, a data glossary is just used to organize data. However, data.all uses it to power its search. This further encourages users to enrich and maintain the glossary regularly.



4.4.2 Glossaries

A Glossary is a list of terms, organized in a way to help users understand the context of their datasets. For example, terms like "cost", "revenue", etc, can be used to group and search all financial datasets.

The use of familiar terminology helps in quickly understanding the data and its background. It is a crucial element of data governance as it helps in bringing the business understanding closer to an organization's data initiatives.

On data.all, glossary terms can be attached to any dataset and can be leveraged to power quick and ease data discovery in the Catalog.

Spotlight

Glossaries are built hierarchically. They are made of categories and terms. This structure allows for glossaries from multiple domains to co-exist.

Term:

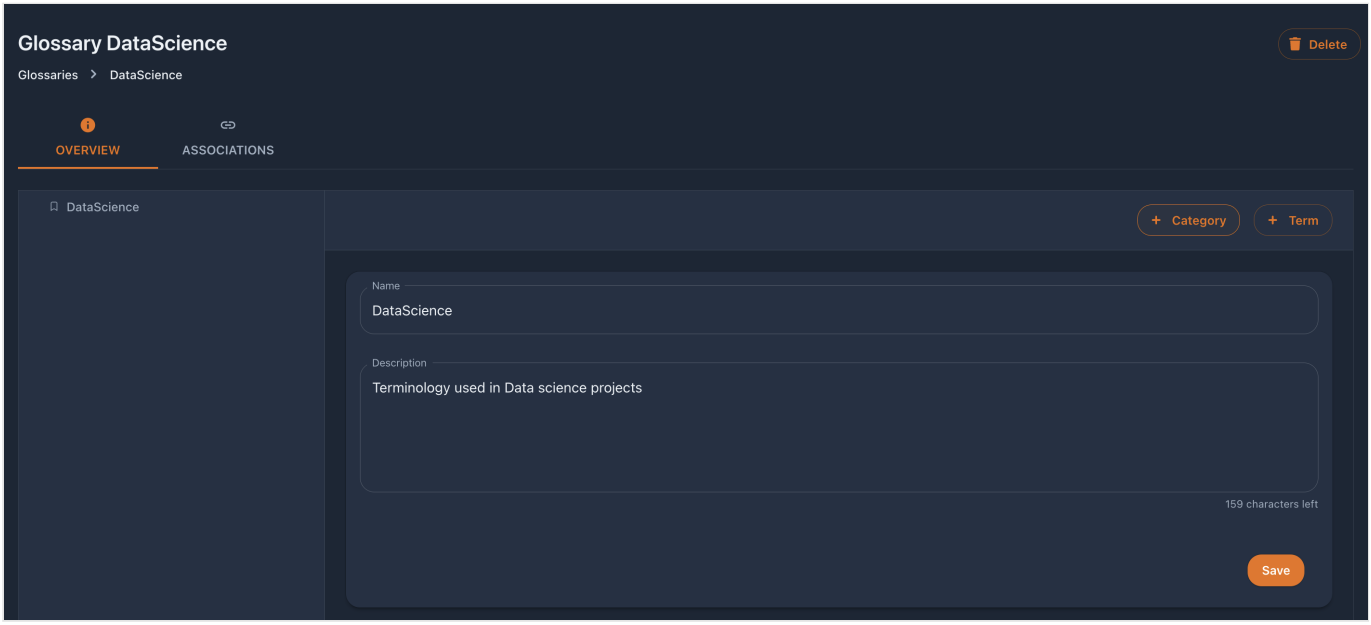
- A term is the lowest unit which is unique inside each glossary.
- It describes the content of the data assets in the most useful and precise way.
- It can exist independently, without belonging to any particular category or sub-category.

Category:

A category is used to group the terms of a similar context together. it is just a way of organizing terms.

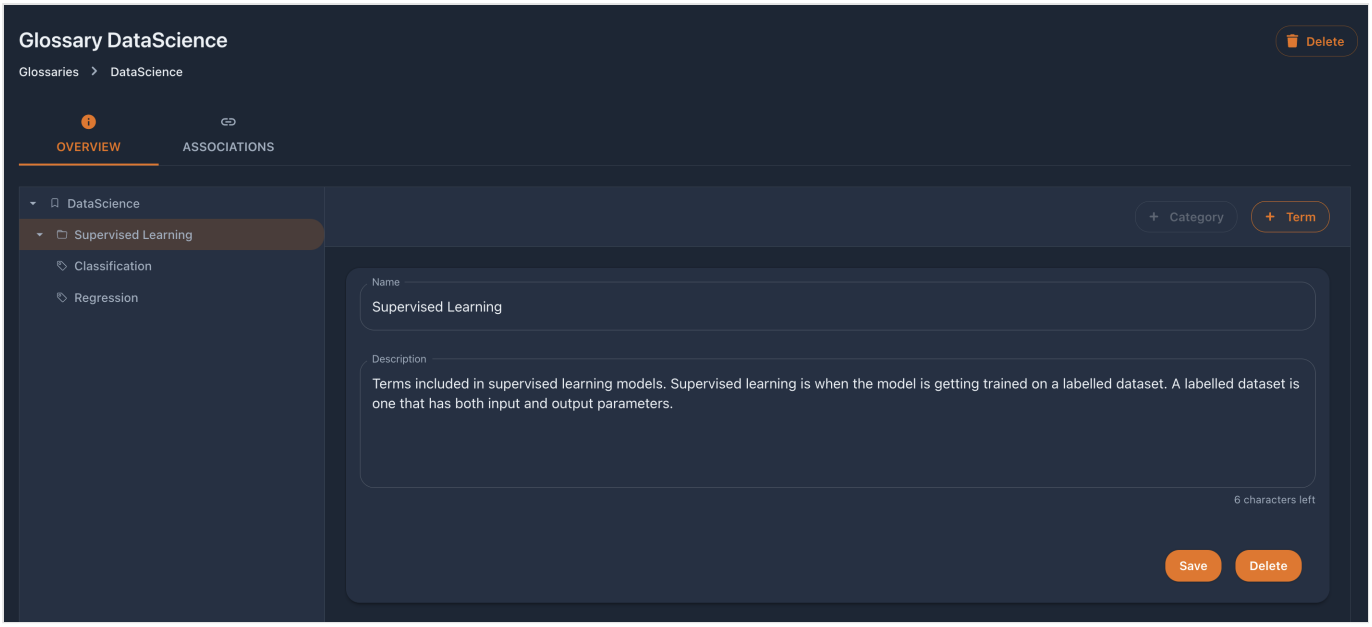
Create a new glossary

1. Go to **Glossaries** menu on the elft side pane.
2. Click on **Create**.
3. Fill the form and add a new glossary.



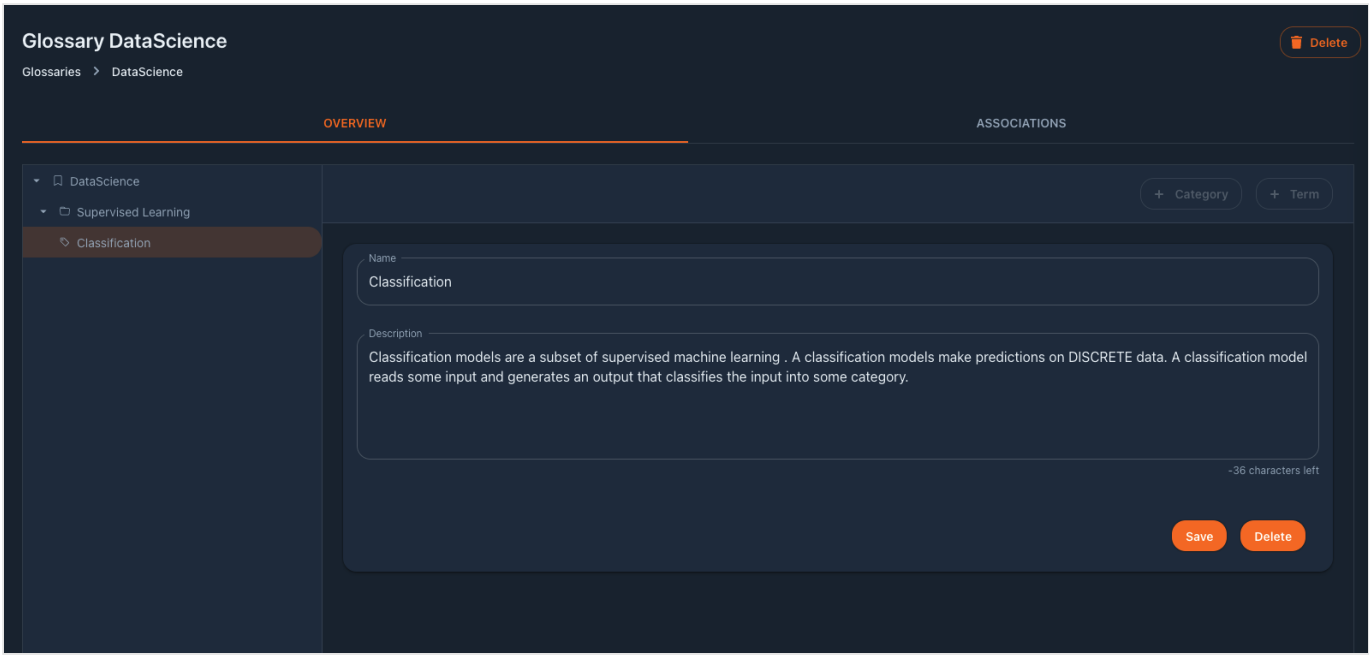
Add a category inside a Glossary

1. Click on the button "Add category" to add a new category.
2. Add a name and description to your category for better understanding.



Add terms to a category

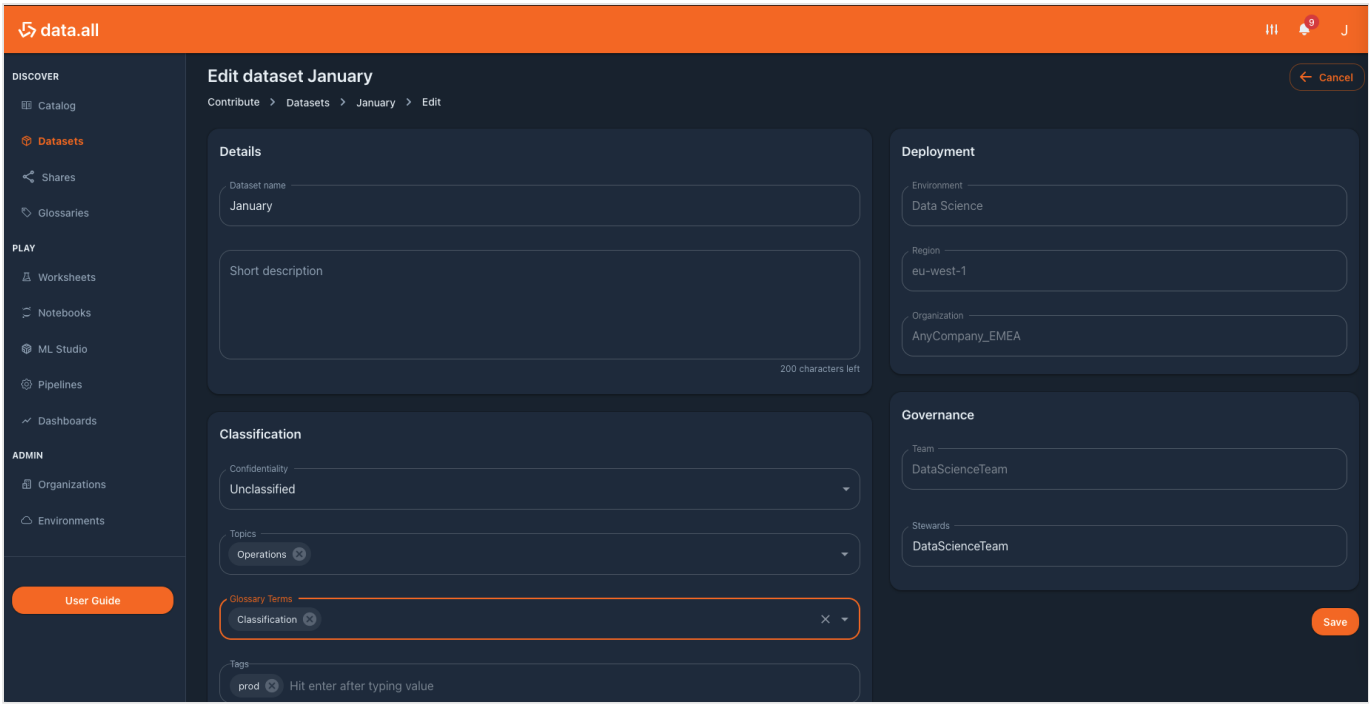
1. Click on the button "Add term" to add a new term to the category.
2. Give it an appropriate name and description.



> **Remember!**
>
> The term will be used to recognize and filter the datasets. Hence, keep it short and precise.

Link your data with appropriate glossary terms

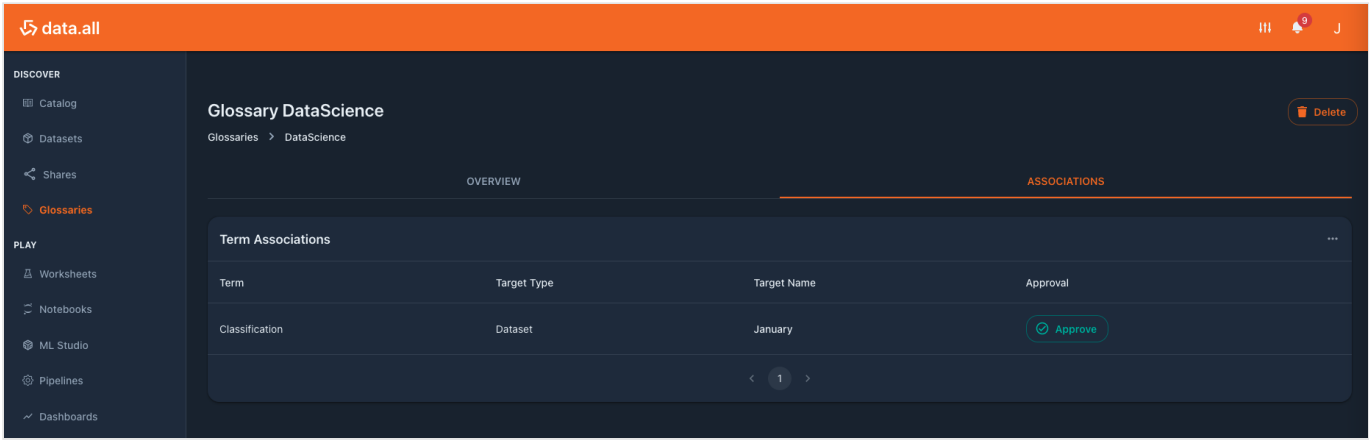
You can associate a glossary term to a dataset or a table. Go to a dataset click on "edit" and update the glossary terms field as shown below



Approve and Check all data related to a glossary

To see a list of all datasets and tables that have been linked with terms of a specific glossary, go to Glossaries and select the glossary. In the **Associations** tab it is possible to check the related data assets (target name), their types (e.g. dataset) and the specific term that they have used.

Important: Glossary owners need to approve the association. If it is not approved it won't be used as filter in the catalog.



4.5 Shares

Teams can browse data.all catalog and request access for data assets. data.all shares data between teams securely within and environment and across environments without any data movement.

Concepts

- Share request or Share Object: one for each dataset and requester team.
- Share Item refers to the individual Redshift table, Glue table, folder or S3 Bucket that is added to the Share request.

Shareable items

In data.all there are 2 types of datasets: S3 Datasets and Redshift Datasets. Here is an overview of the items that can be shared using data.all by type of dataset. A detailed explanation of the technical details for each type can be found in the AWS data sharing technical details section.

- From S3 Datasets we can share:
  - S3 Bucket of the Dataset - using IAM permissions and S3/KMS policies
  - one or multiple Glue Tables (Tables) - using Lake Formation to create access permissions to tables, meaning that no data is copied between AWS accounts.
  - one or multiple S3 Prefixes (Folders) - using S3 access points to manage granular S3 policies.
- From Redshift Datasets we can share:
  - one or multiple Redshift Tables - using Redshift datashares

Sharing workflow

Requesters create a share request and add items to it. Both requesters and approvers can work on this `DRAFT` of the request and add and delete items to the request Draft. Items that are added go to the `PENDINGAPPROVAL` status.

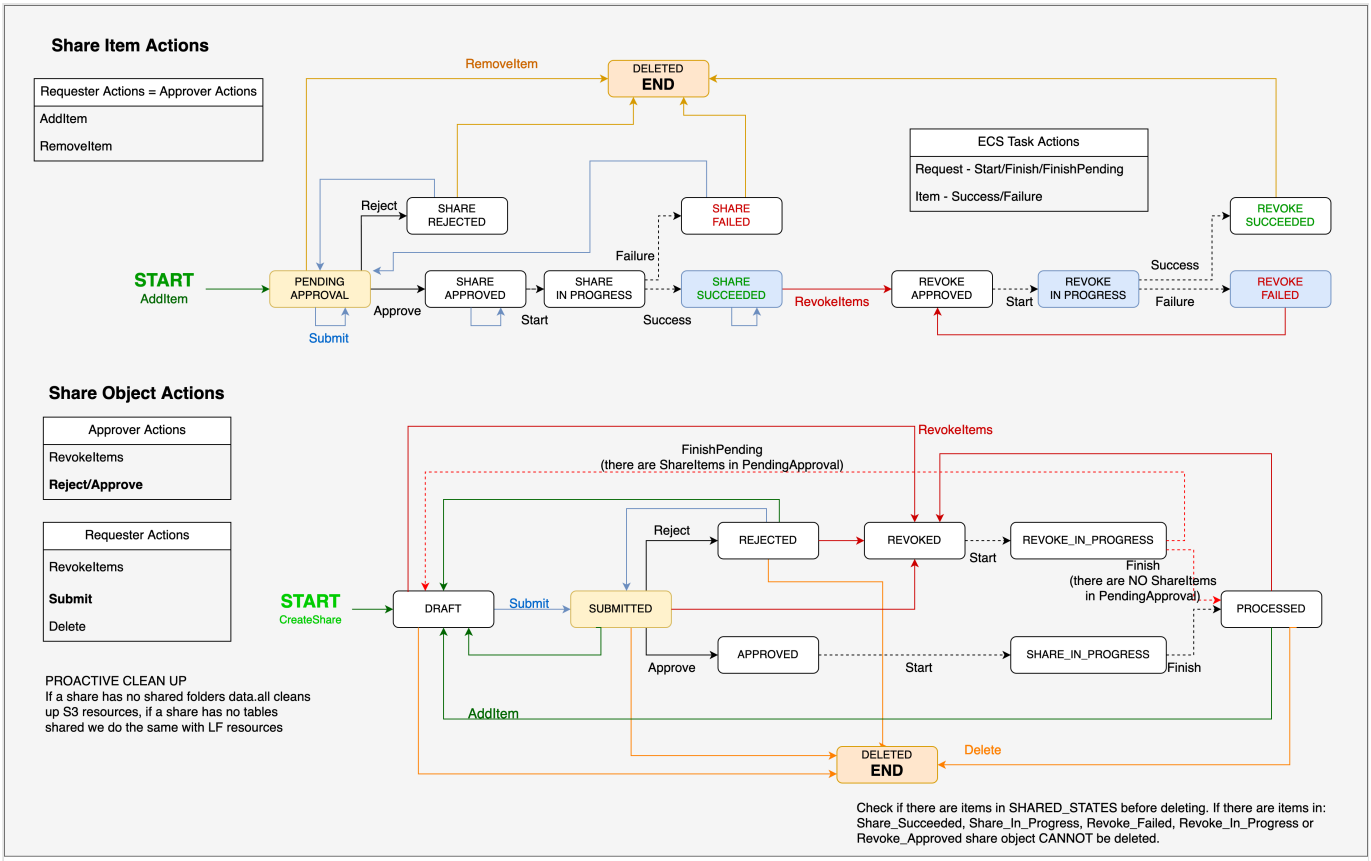
Once the draft is ready, requesters **submit** the request, which moves to the `SUBMITTED` status. Then, approvers **approve** or **reject** the request which will go to `APPROVED` or `REJECTED` status and its items to `SHARE_APPROVED` or `SHARE_REJECTED` correspondingly.

When the sharing task starts in the backend, both items and the share object move to `SHARE_IN_PROGRESS`. Once all items have been processed, the Share object is `PROCESSED` and each of the items is in either `SHARE_SUCCEEDED` or `SHARE_FAILED`. New items can be added to the share requests, the request will go back to `DRAFT` to be re-processed.

Both approvers and requesters can revoke access to shared items. They open the revoke items window and select which items should be revoked from the share request. The items move to `REVOKE_APPROVED` while the share is in `REVOKED` status.

While the revoking task is executing, the items and the request remain in `REVOKE_IN_PROGRESS` until the revoke is complete and items go to `REVOKE_FAILED` or `SUCCEEDED`. If there are share items in `PENDINGAPPROVAL` in the share request, it will go back to `DRAFT`. Otherwise, it will go to `PROCESSED`.

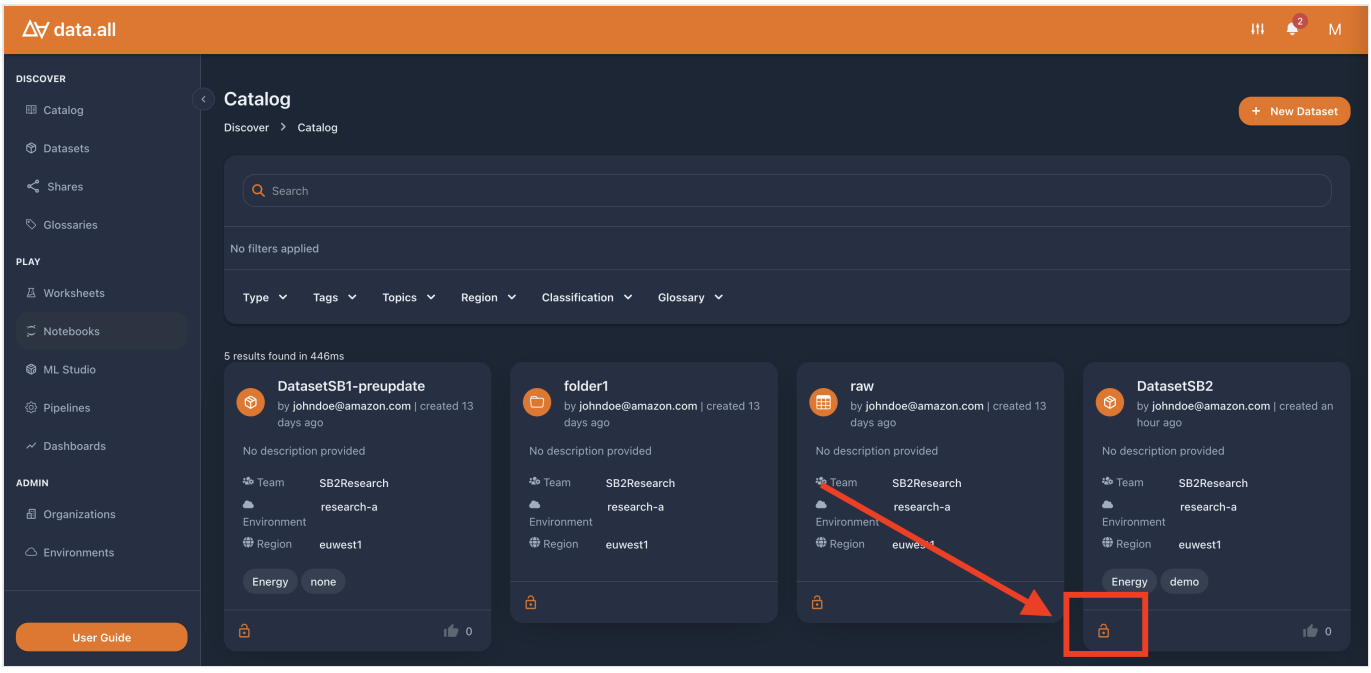
Requesters can delete the share request with the **delete** button. However, the request cannot contain any shared items. Users must revoke all shared items before deletion.



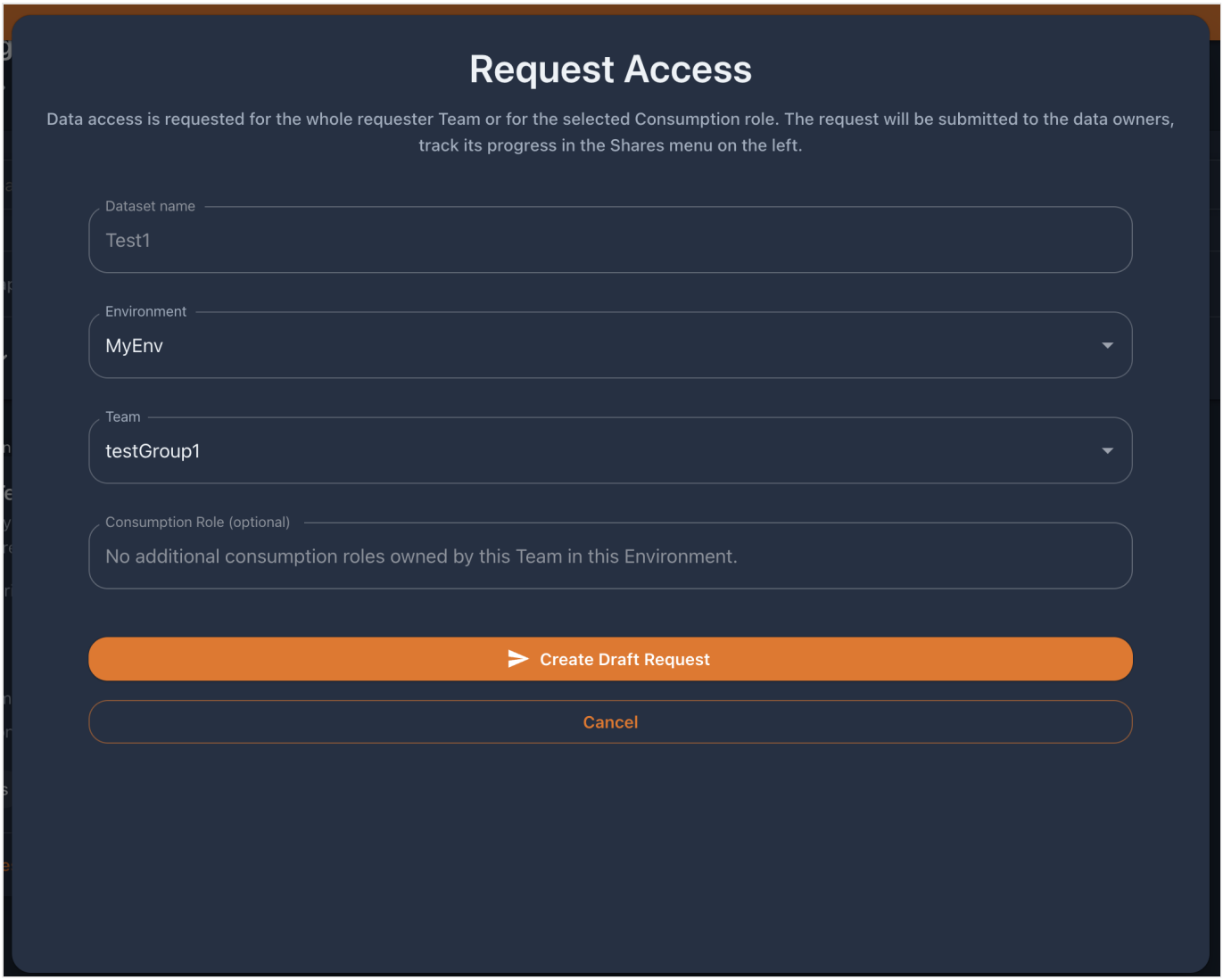
4.5.1 Create a share request (requester)

S3/Glue share request

On left pane choose **Catalog** then **Search** for the table you want to access. Click on the lock icon of the selected data asset.

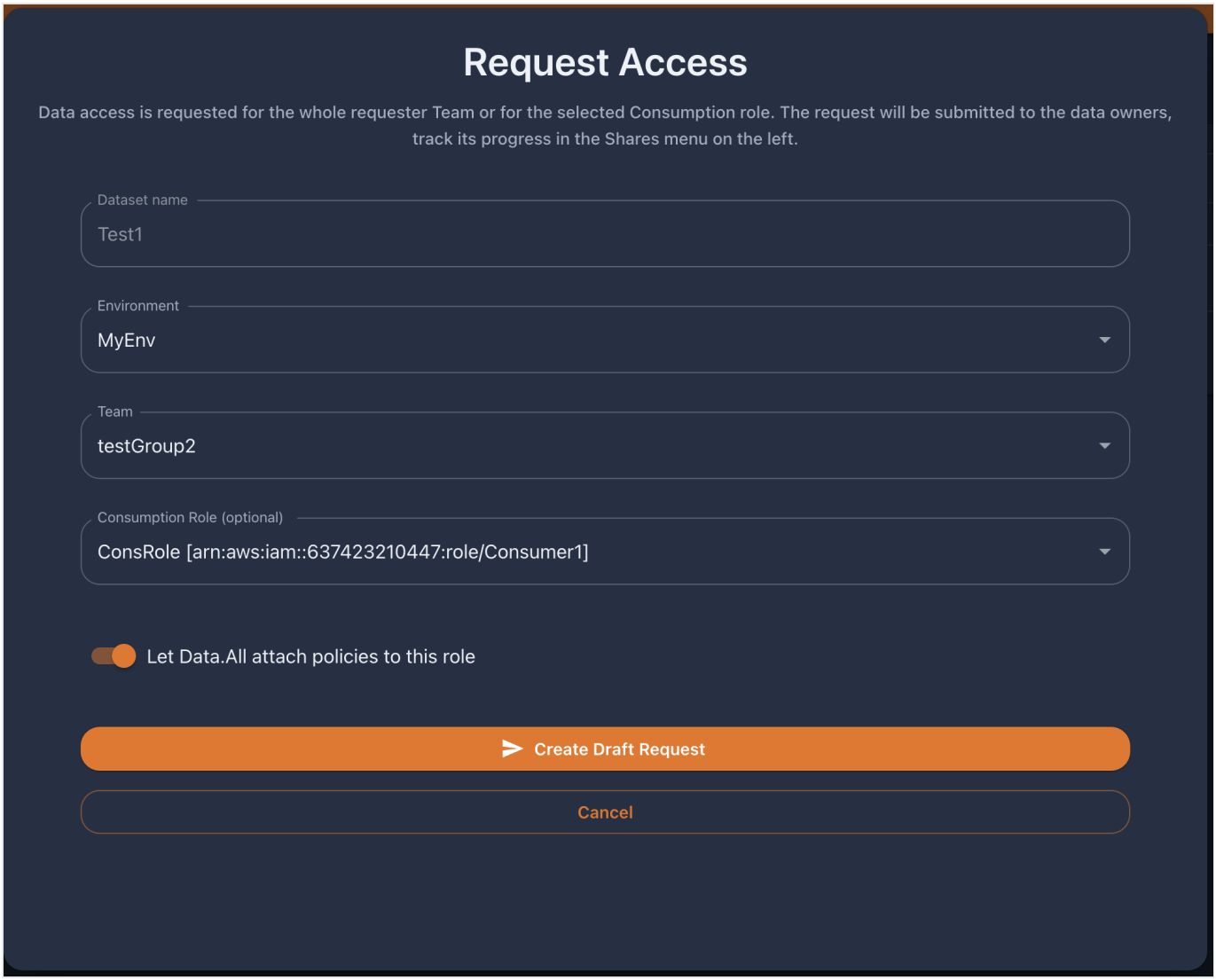


The following window will open. Choose your target environment and team.

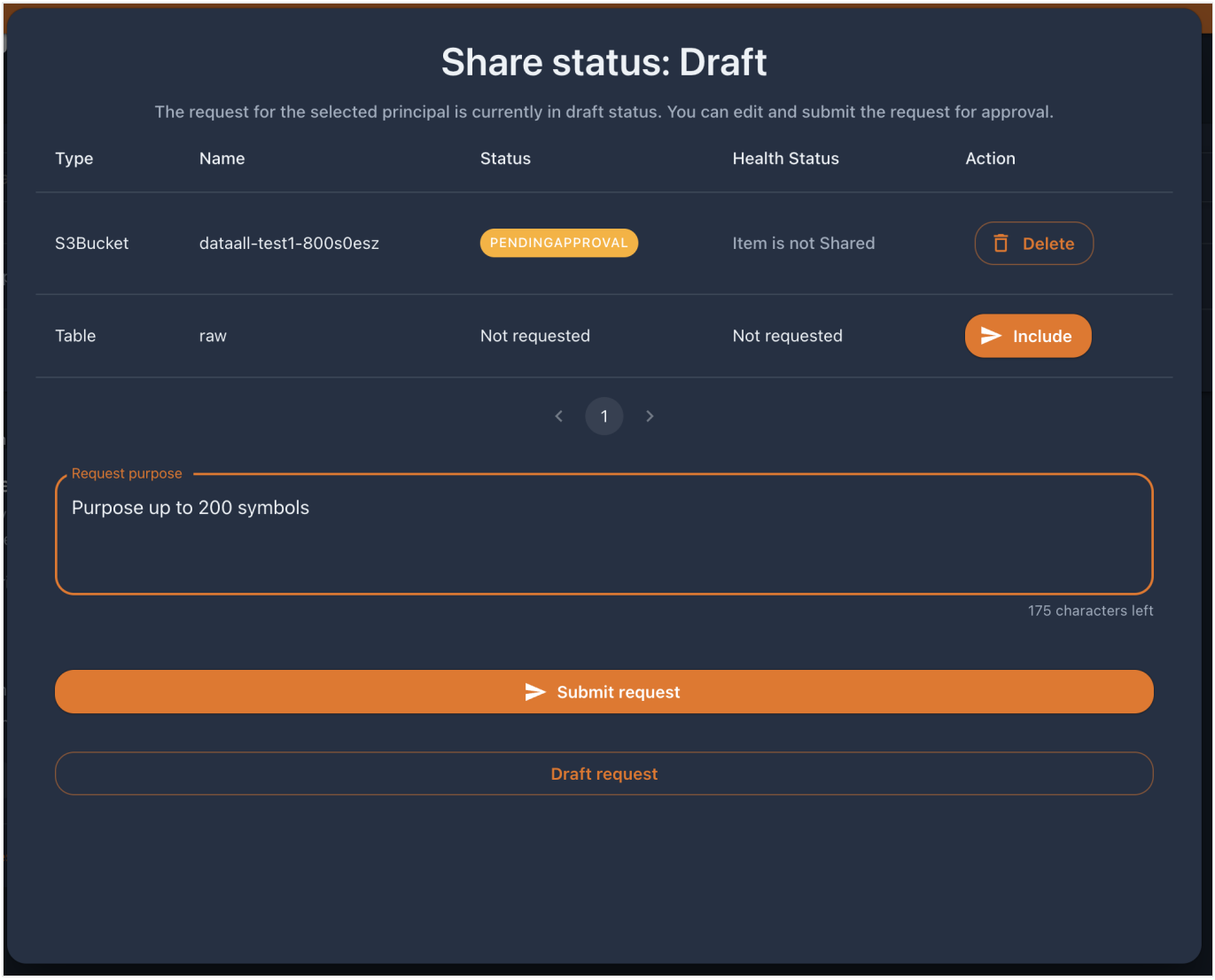


If instead of to a team, you want to request access for a Consumption role, add it to the request as in the picture below.

NOTE: If the consumption role selected is not data.all managed - you will have the option to allow data.all to attach the share policies to the consumption role for this particular share object (if not enabled here you will have to manually attach the share policies to be given access to data).


Request Access
Data access is requested for the whole requester Team or for the selected Consumption role. The request will be submitted to the data owners, track its progress in the Shares menu on the left.
Dataset name
Test1
Environment
MyEnv
Team
testGroup2
Consumption Role (optional)
ConsRole [arn:aws:iam::637423210447:role/Consumer1]
Let Data.All attach policies to this role
Create Draft Request
Cancel

Finally, click on **Create Draft Request**. This will create a share request or object for the corresponding dataset and if you have requested a table or folder it will add those items to the request. After that the modal window will switch to share edit form.



Here you can edit the list of items you want to request access to. Note that the request is in `DRAFT` status and that the items that we add are in `PENDINGAPPROVAL`. They are not shared until the request is submitted and processed. The share can not be submitted if the list of items is empty.

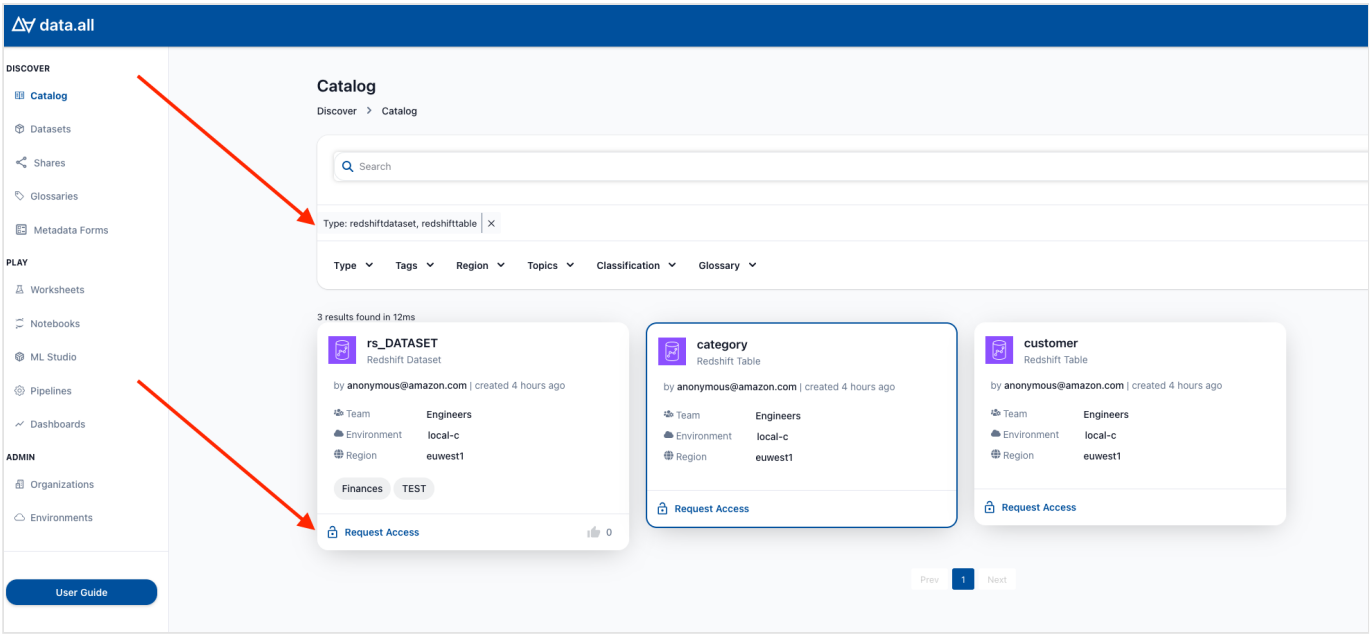
`Request purpose` is optional field, recommended length is up to 200 symbols.

When you are happy with the share request form, click **Submit Request** or click **Draft Request** if you want to return to this form later.

The share needs to be submitted for the request to be sent to the approvers.

Redshift share request

Navigate to the Catalog, on top of other filters, you can use the Redshift dataset and table filters to list only Redshift data items. Once you have found the item you want, click on Request access to open a share request.



Pre-requisites To be able to open a share request to a Redshift Dataset, a data.all Redshift Connection of type `ADMIN` in the namespace of the Redshift Dataset is required.

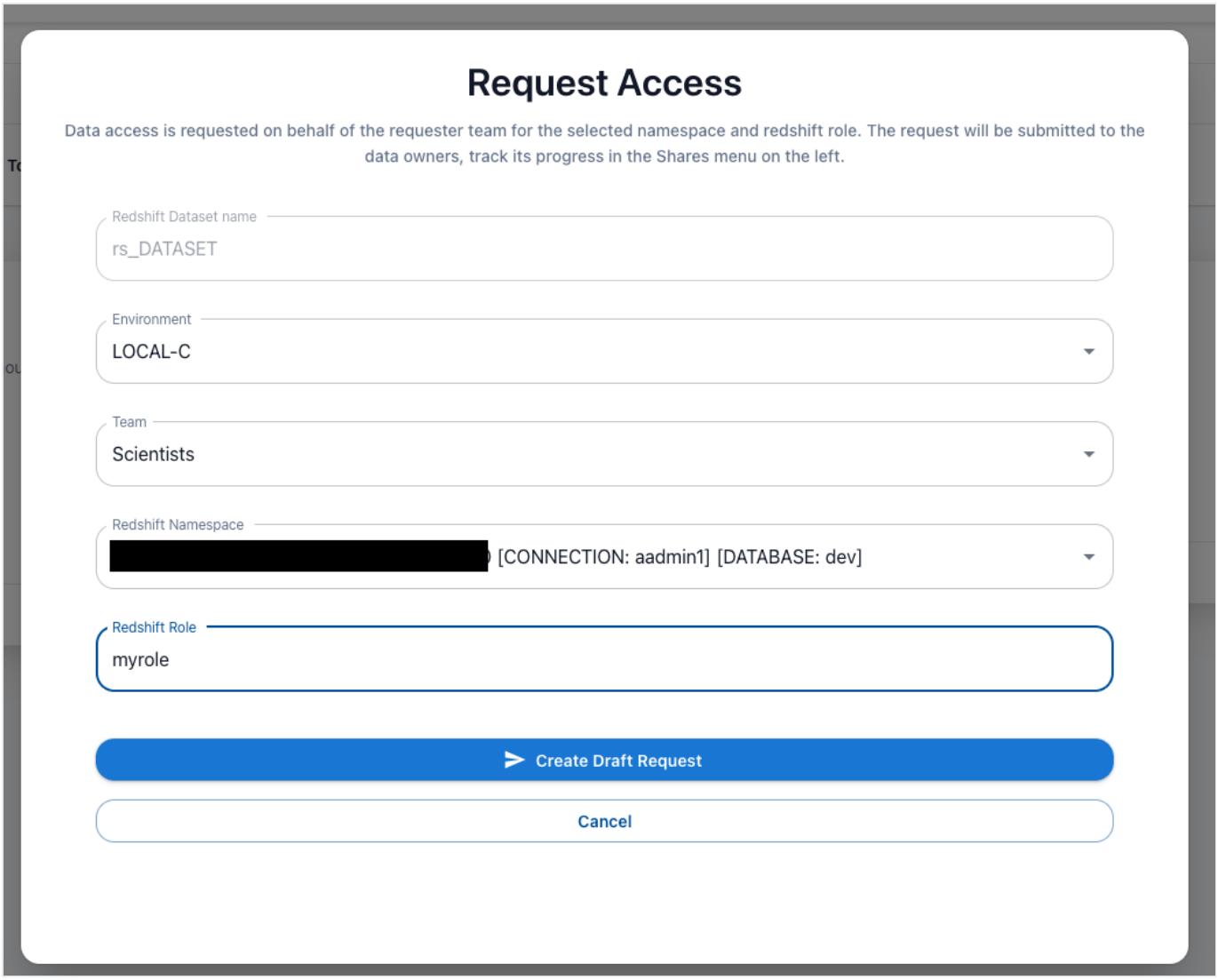
Similarly, the namespace that we want to use as target MUST have a data.all `ADMIN` connection that allows data.all to manage datashares in it. In addition, the group that we use as requester MUST have permissions to use that connection in a share request.

Taking the request in the picture as example. `rs_Dataset` is stored in `cluster-1` and the requester team `Scientists` wants to access the data from `cluster-2`.

- **Source connection**: the admin team of `cluster-1` has created a connection `connection1` of type `ADMIN` for this cluster.
- **Target connection**: the admin team of `cluster-2` has created a connection `connection2` of type `ADMIN` for this cluster. The `Administrators2` team has granted permissions to `Use Connection in share request` to the `Scientists` team.

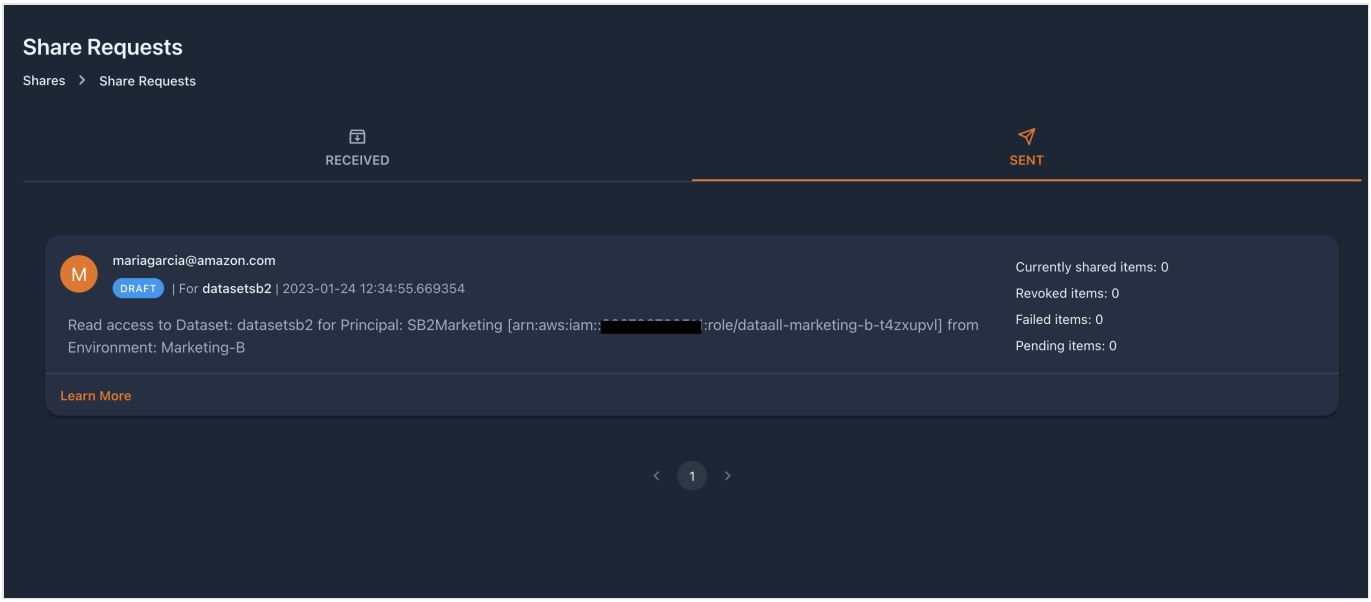
Check out the Redshift Datasets documentation for more information about `ADMIN` connections and how admins can update Connection permissions.

Once the pre-requisites are fulfilled, you will be able to open a share request specifying the target namespace and the Redshift role that will get access to the data.



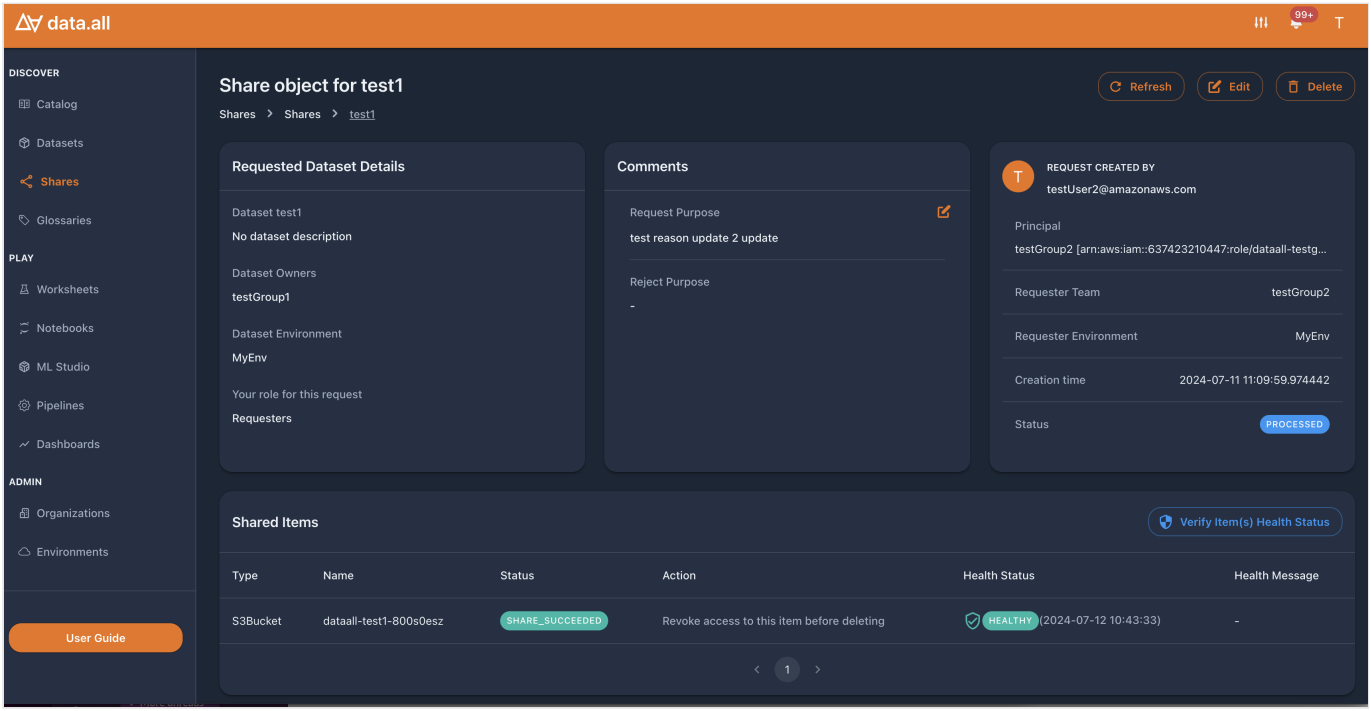
4.5.2 **Check your sent/received share requests**

Anyone can go to the Shares menu on the left side pane and look up the share requests that they have received and that they have sent. Click on **Learn More** in the request that you are interested in to start working on your request.



4.5.3 **Add/delete items**

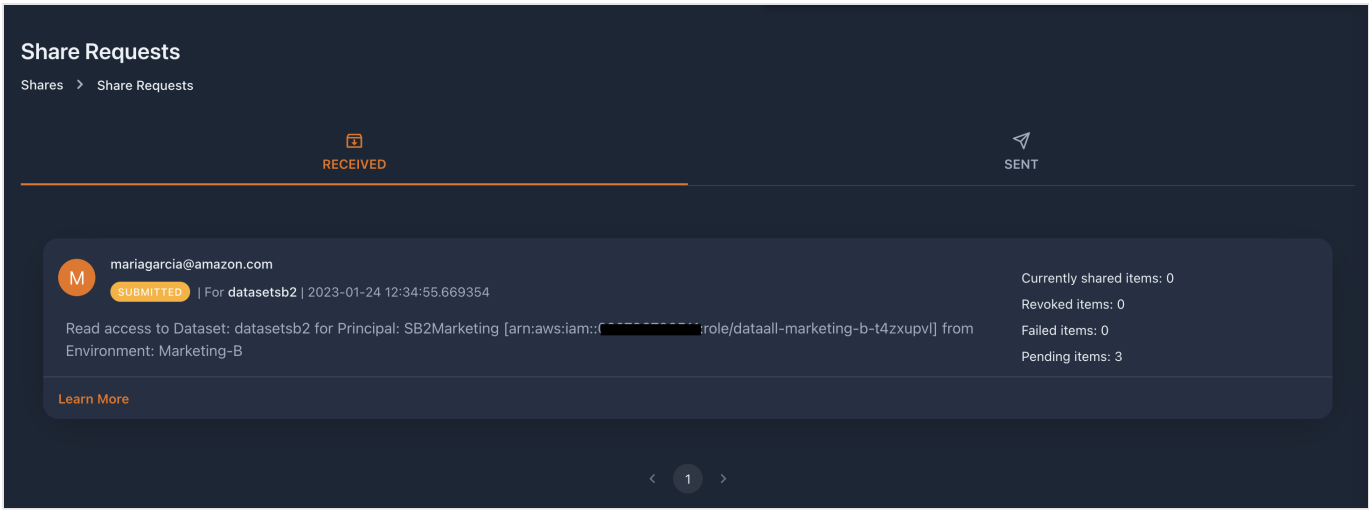
If the request is not being processed, it can be edited by clicking the **Edit** button on top of the page.



Edit button opens the modal window with the Share Edit Form, same as upon creating the share. Here you can edit list of shared items and request purpose. To remove an item from the request click on the **Delete** button with the trash icon next to it. We can only delete items that have not been shared. Items that are shared must be revoked, which is explained below.

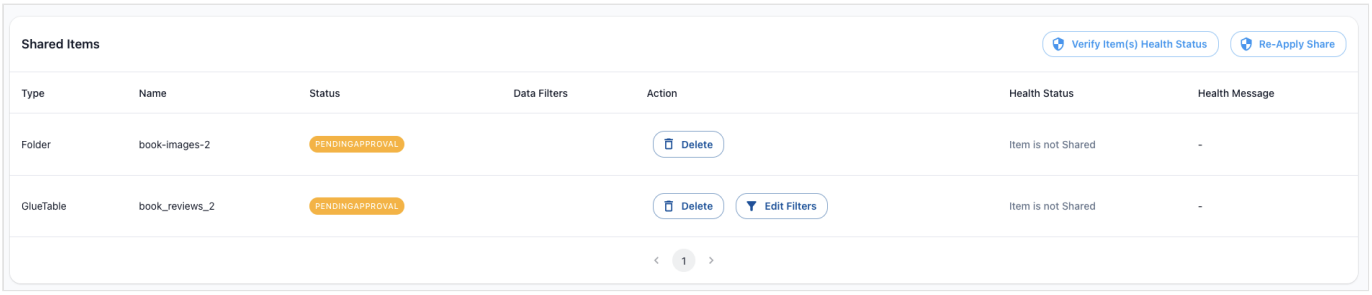
4.5.4 **Submit a share request (requester)**

Once the draft is ready, the requesters need to click on the **submit** button. The request should be now in the `SUBMITTED` state. Approvers can see the request in their received share requests, alongside the current shared items, revoked items, failed items and pending items.

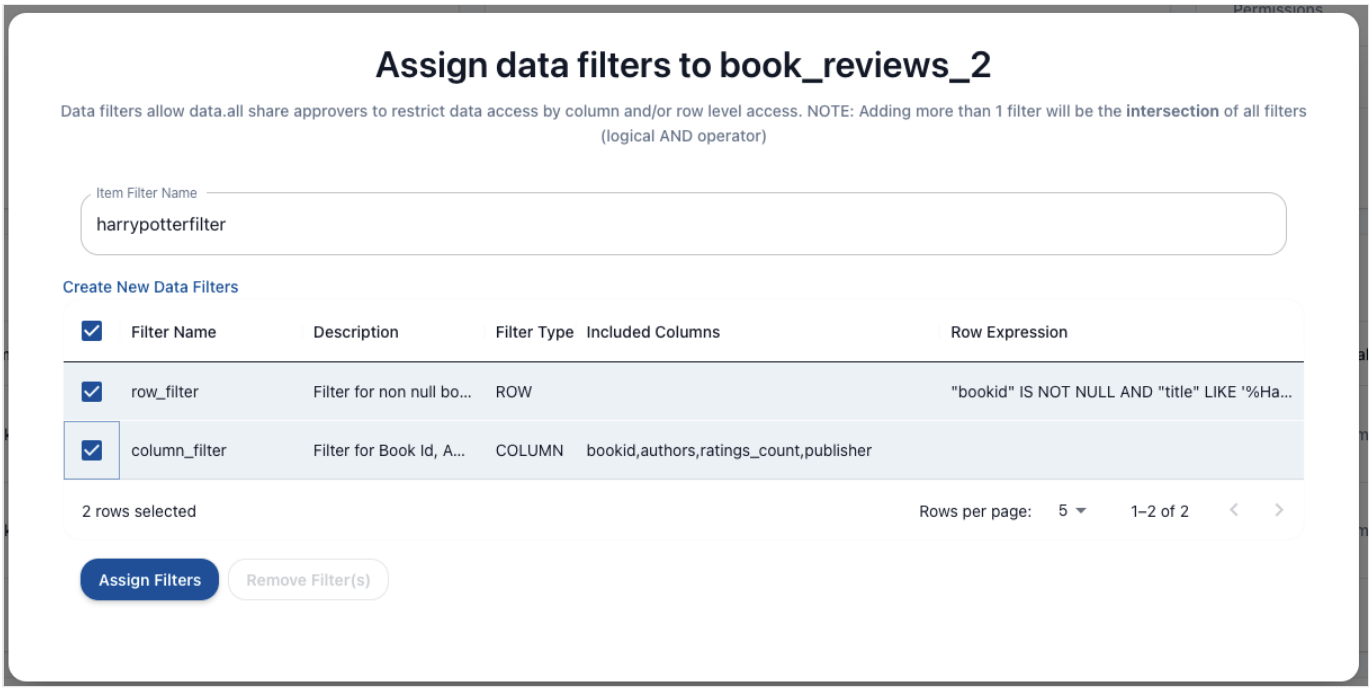


4.5.5 (Optional Pre-Approval Work) Adding Filters to Glue Table Share Items (approver)

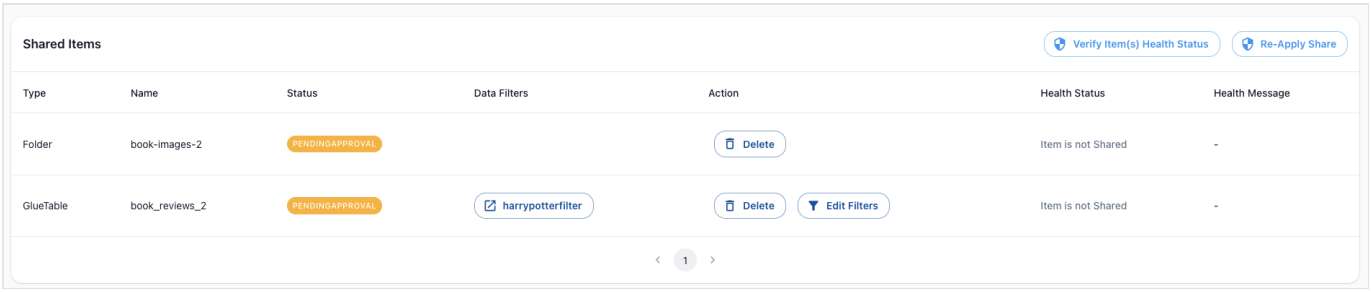
As an approver, you will also see the option to **Edit Filters** for Glue Table share items:

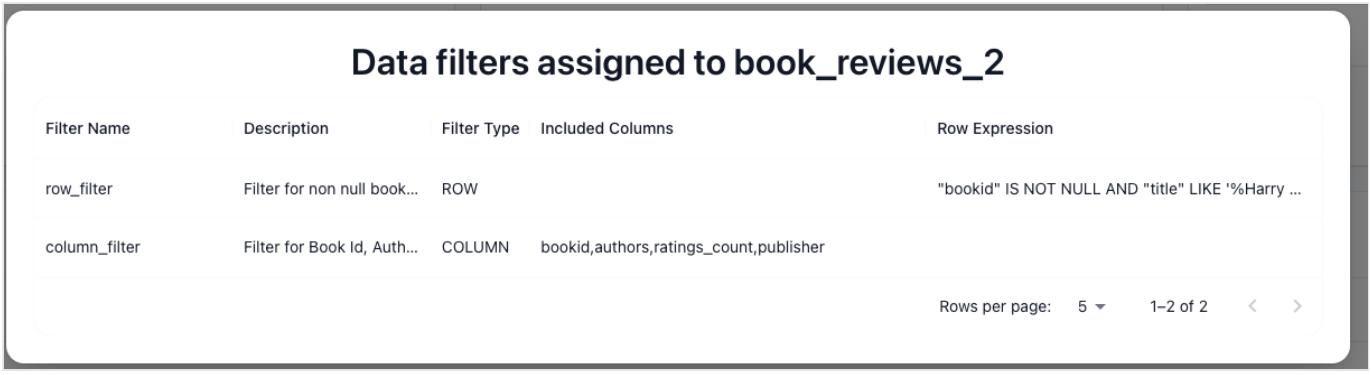


Here an approver can attach one or more filters that were created on the table previously to the table:



Once assigned, the filter will appear in the share object view and can be clicked on to view the underlying associated data filters assigned





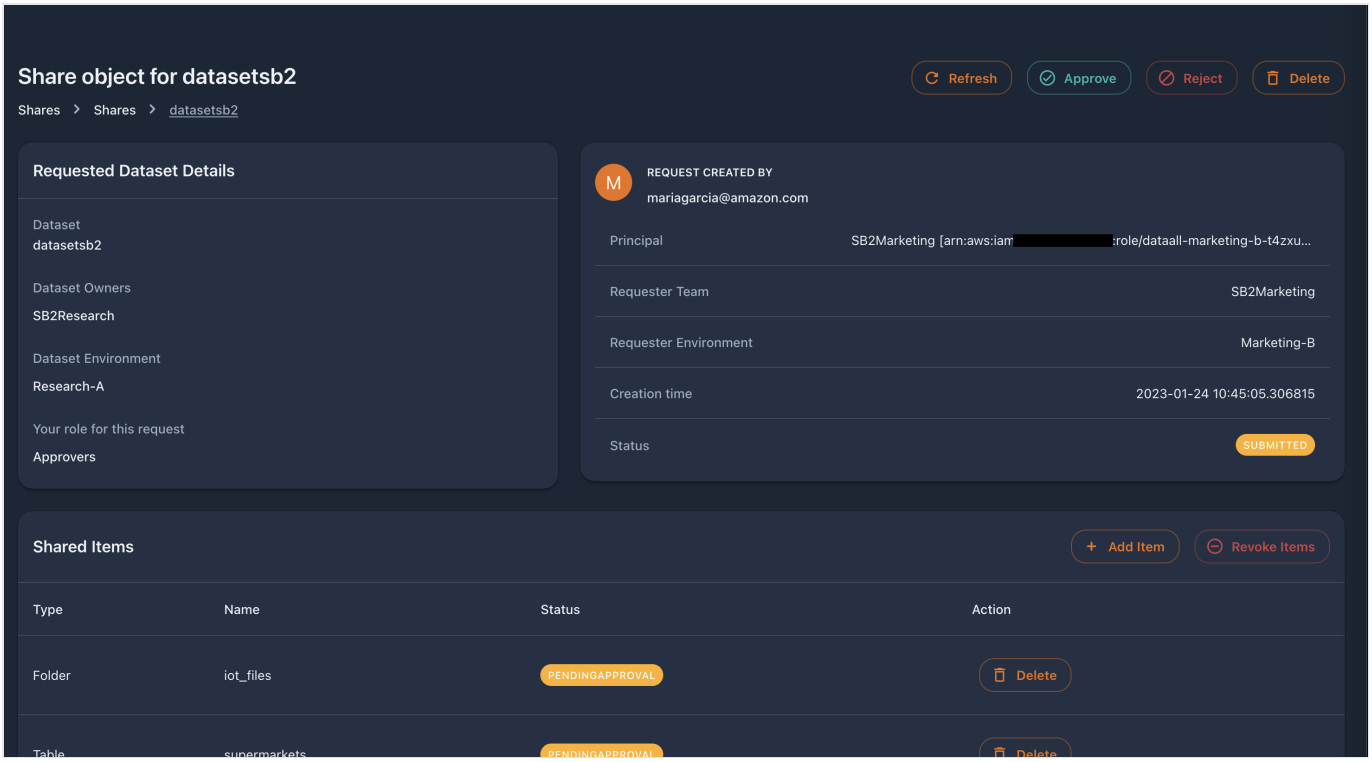
Before sharing as the table - approvers can also edit the assigned filter and remove underlying data filters or attach new ones as needed. Once the share is approved there is no longer the ability to edit filters and the table item must be revoked and re-shared to assign new filters.

NOTE: If more than 1 filter is assigned to a table share item, the resulting data access is evaluated as the union (logical 'OR') of the filters assigned.

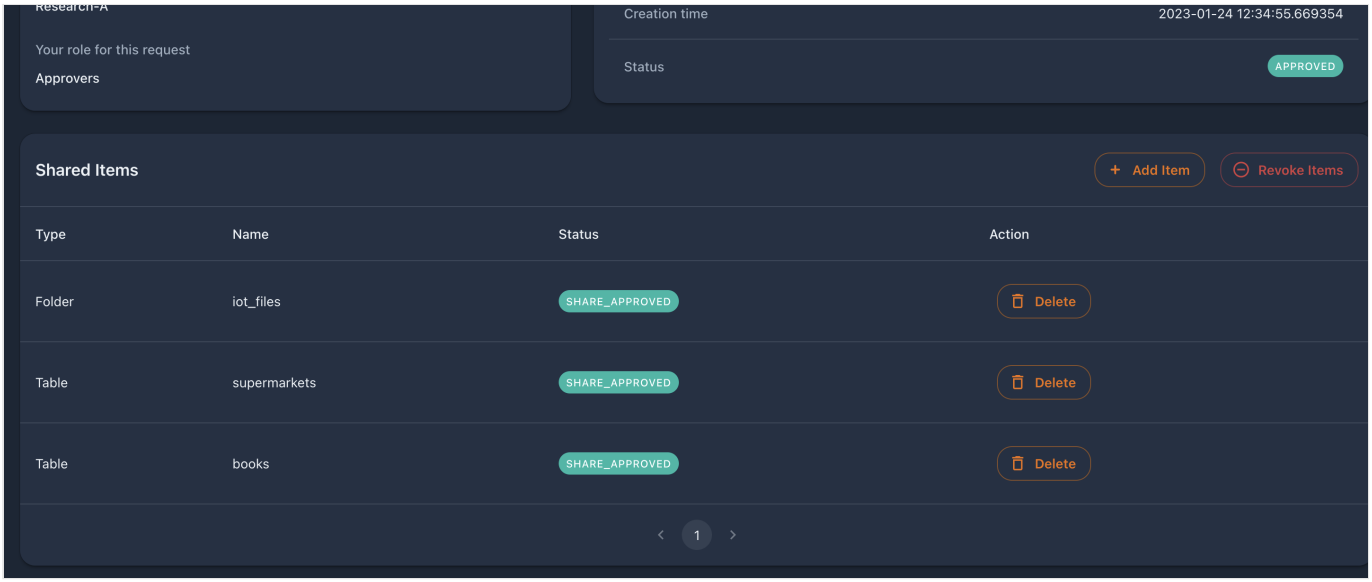
NOTE: If assigning filter(s) to a table share item, the **Item Filter Name** specified will be used in naming the table resource link for the consumer, meaning the consumer will be reading for table named - `tablename_filtername`

4.5.6 Approve/Reject a share request (approver)

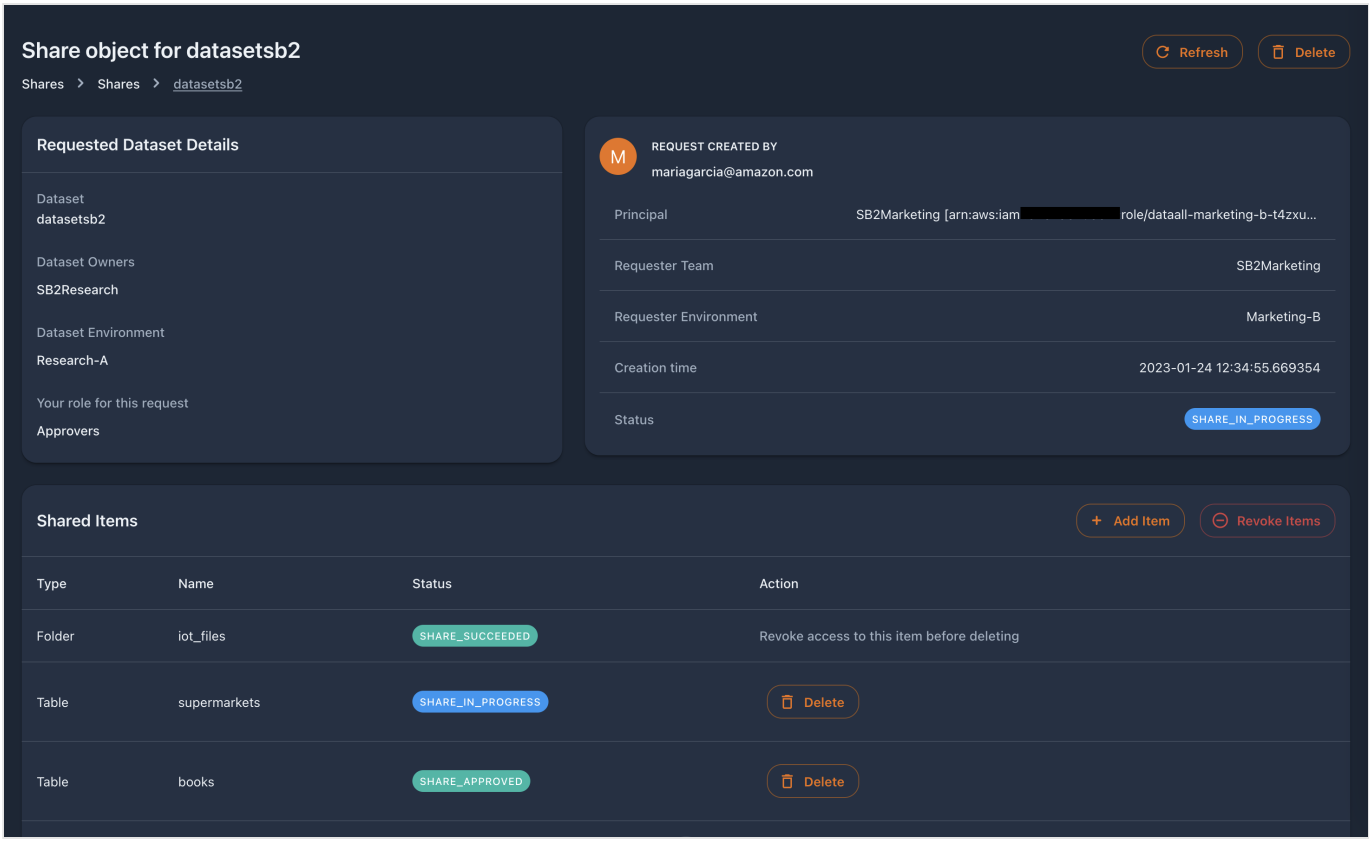
As an approver, click on **Learn more** in the `SUBMITTED` request and in the share view you can check the tables and folders added in the request. This is the view that approvers see, it now contains buttons to approve or reject the request.



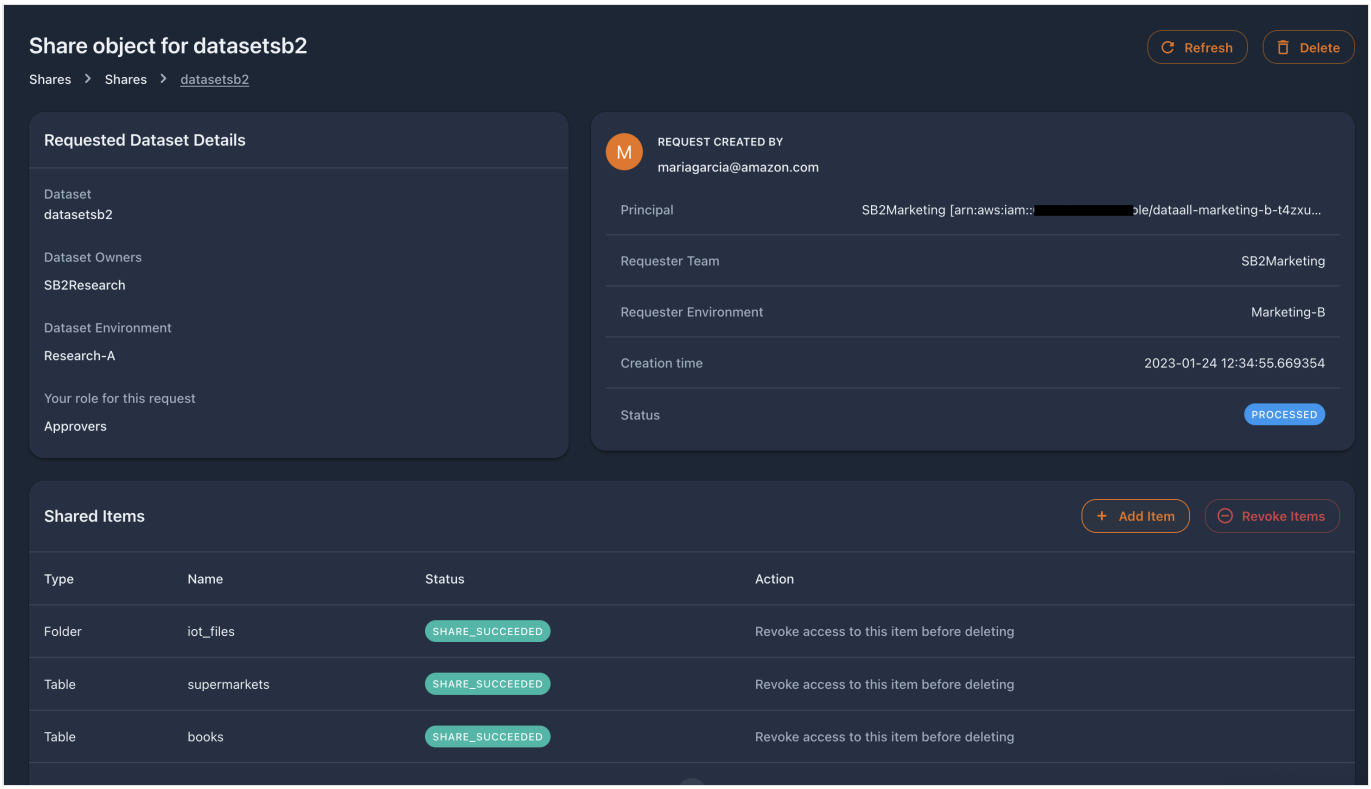
If the approvers **approve** the request, it moves to the `APPROVED` status. Share items IN `PENDINGAPPROVAL` will go to `SHARE_APPROVED`.



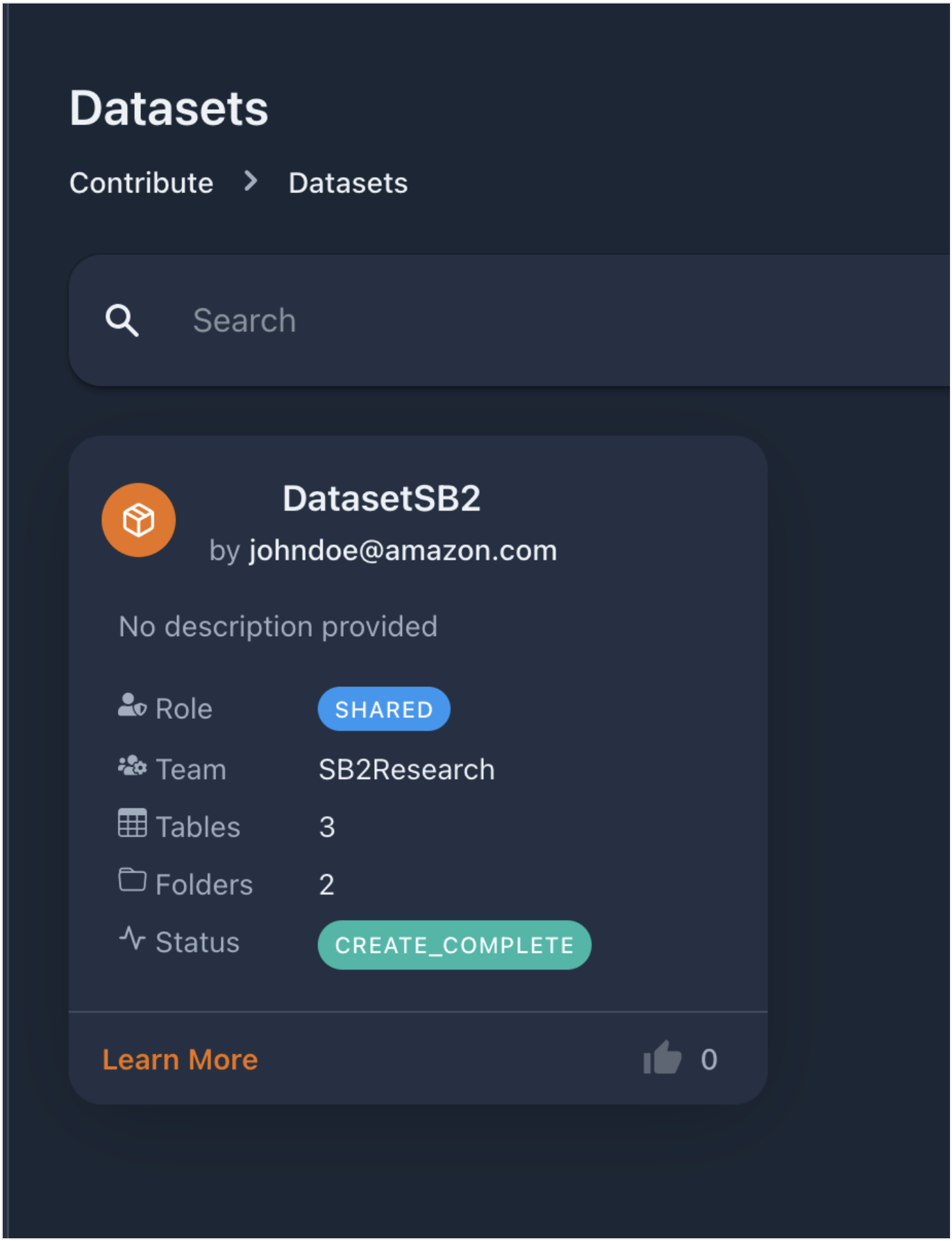
Data.all backend starts a sharing task, during which, items and the request are in `SHARE_IN_PROGRESS` state.



When the task is completed, the items go to `SHARE_SUCCEEDED` or `SHARE_FAILED` and the request is `PROCESSED`. To understand what happens under-the-hood when each share item is processed, check out the AWS data sharing technical details section.

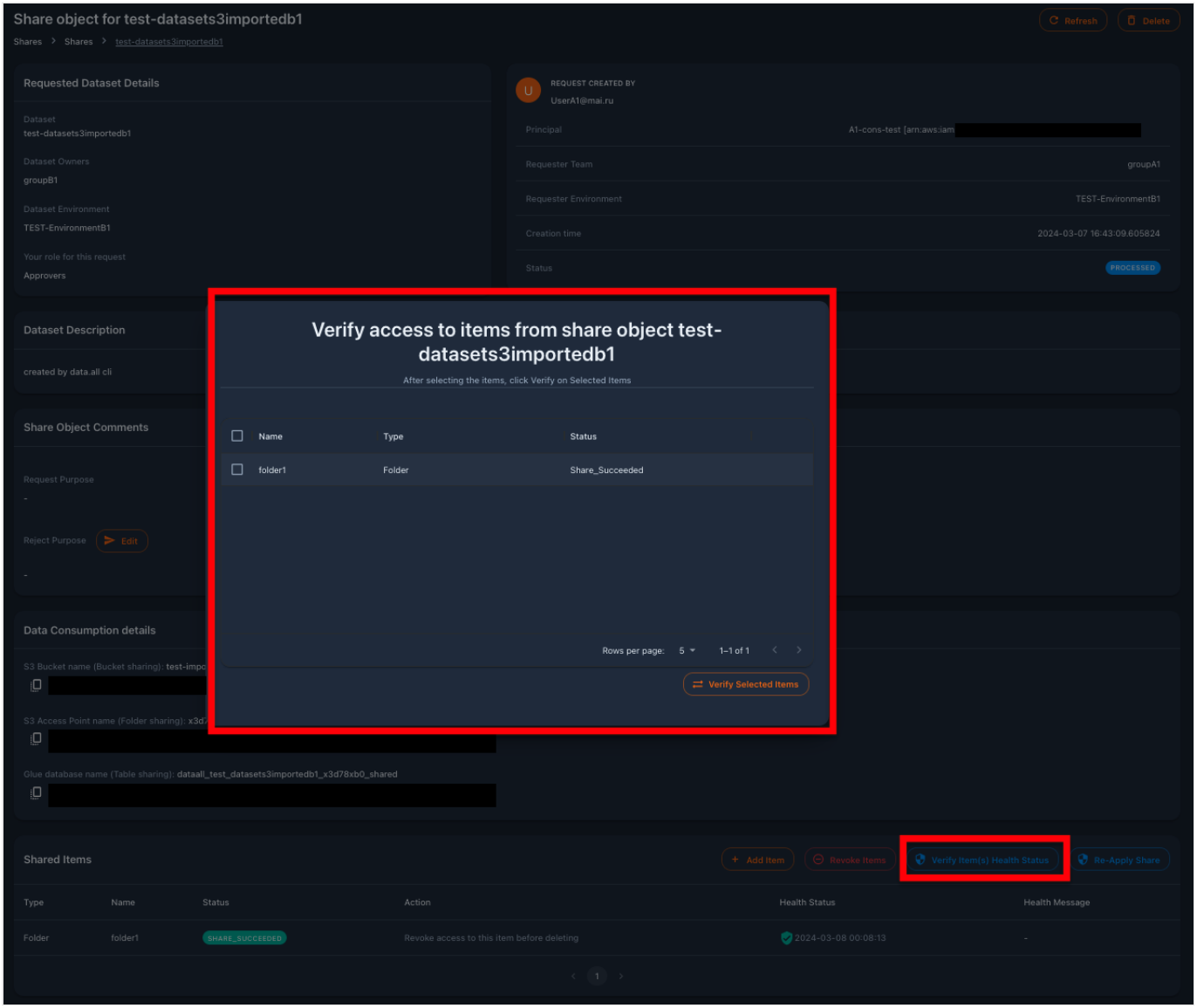


If a dataset is shared, requesters should see the dataset on their screens. Their role with regards to the dataset is `SHARED`.


Datasets
Contribute
Datasets
Search
DatasetSB2
by johndoe@amazon.com
No description provided
Role
SHARED
Team
SB2Research
Tables
3
Folders
2
Status
CREATE_COMPLETE
Learn More
0

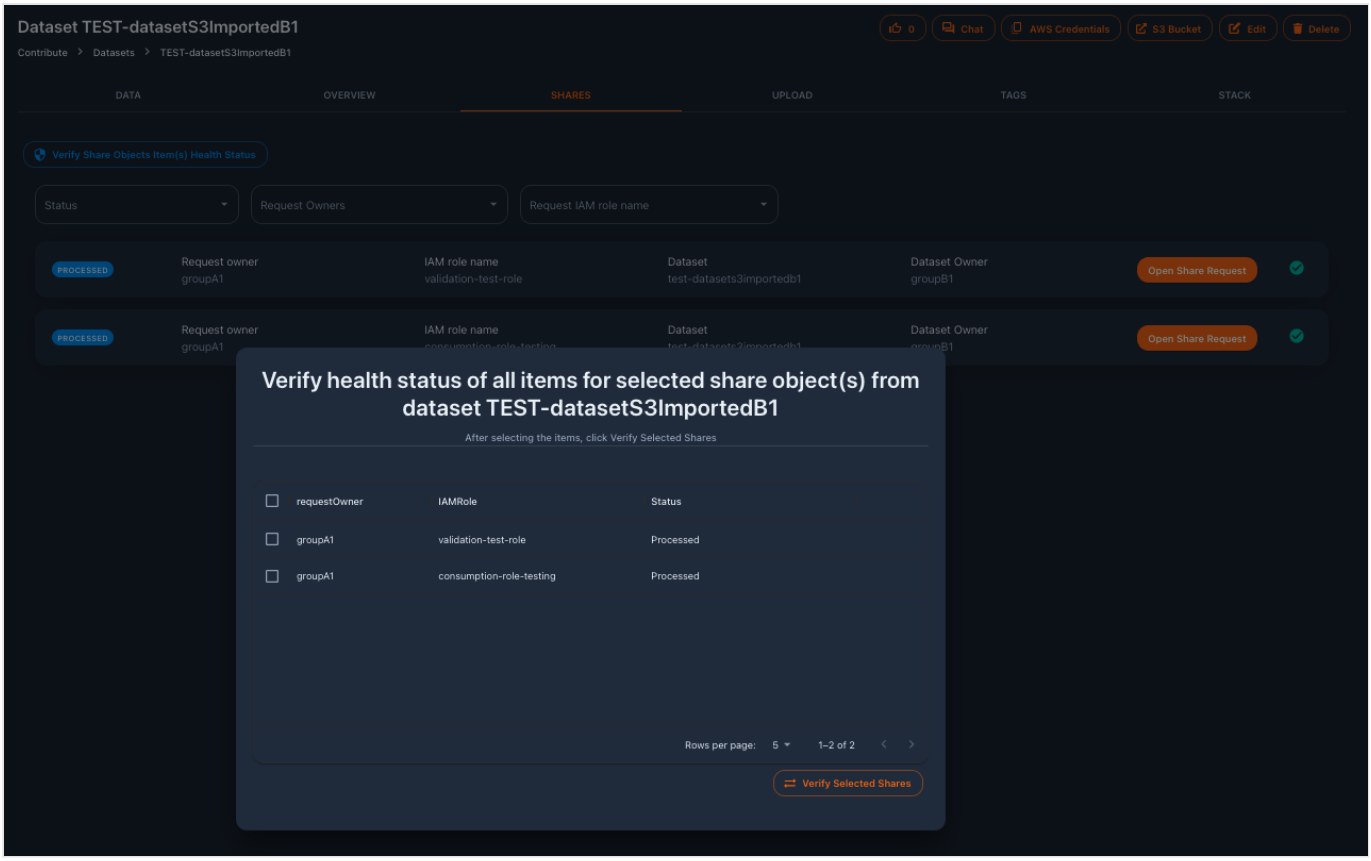
4.5.7 **Verify (and Re-apply) Items**

As of V2.3 of data.all - share requestors or approvers are able to verify the health status of the share items within their share request from the data.all UI. Any set of share items that are in a shared state (i.e. `SHARE_SUCCEEDED` or `REVOKE_FAILED` state) will be able to be selected to start a verify share process.



Upon completion of the verify share process, each share item's healthStatus will be updated with an updated healthStatus (i.e. `Healthy` or `Unhealthy`) as well as a timestamp representing the last verification time. If the share item is in an `Unhealthy` health status, there will also be included a health message detailing what part of the share is in an unhealthy state.

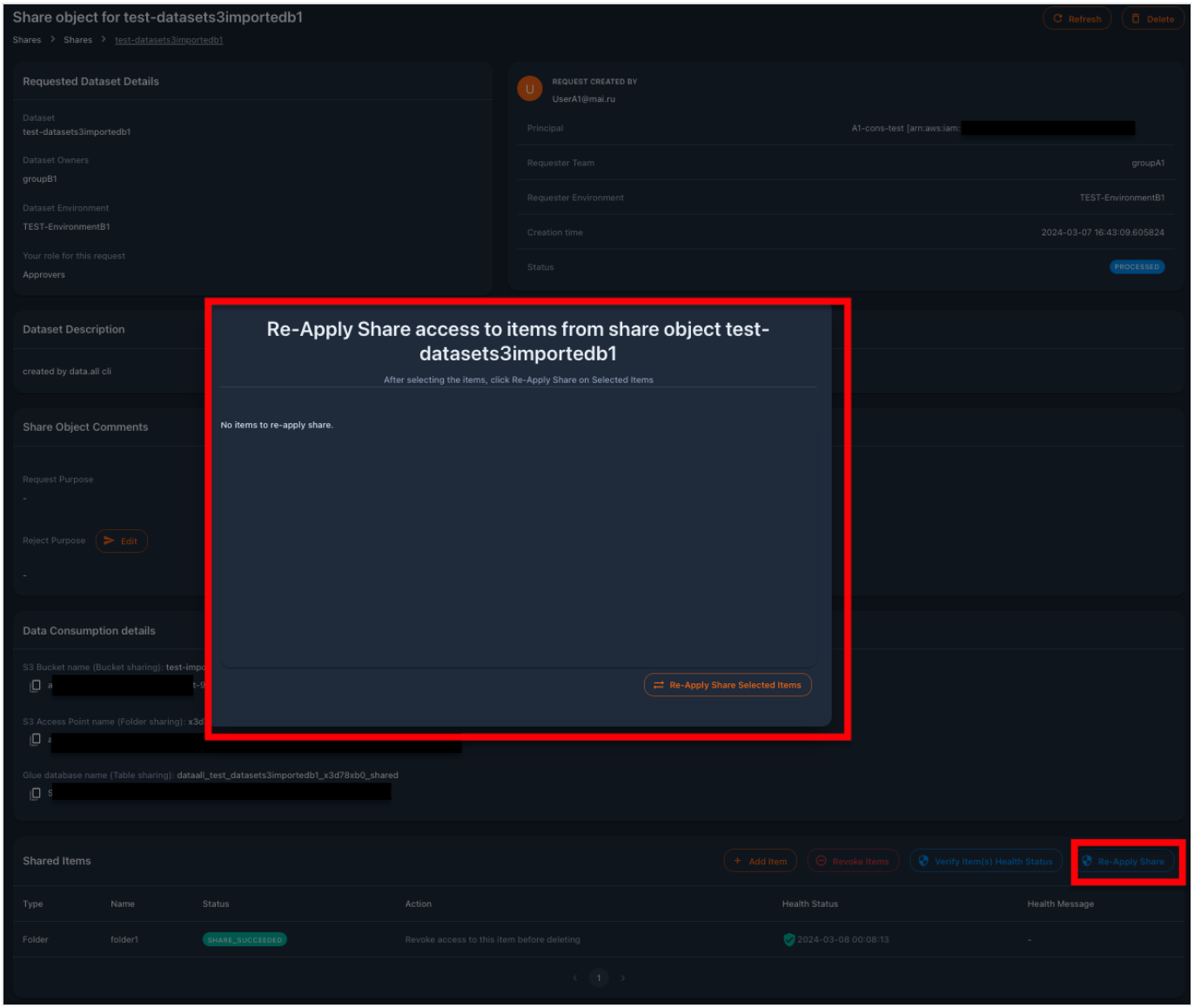
In addition to running a verify share process on particular items, dataset owners can run the verify share process on multiple share objects associated with a particular dataset. Navigating to the Dataset --> Shares Tab, dataset owners can start a verify process on multiple share objects. For each share object selected, the share items that are in a shared state for the associated share object will verified and updated with a new health status and so on.



Scheduled Share Verify Task

The share verifier process is run against all share object items that are in a shared state every 7 days by default as a scheduled task which runs in the background of data.all.

If any share items do end up in an `Unhealthy` status, the data.all approver will have the option to re-apply the share for the selected items that are in an unhealthy state.

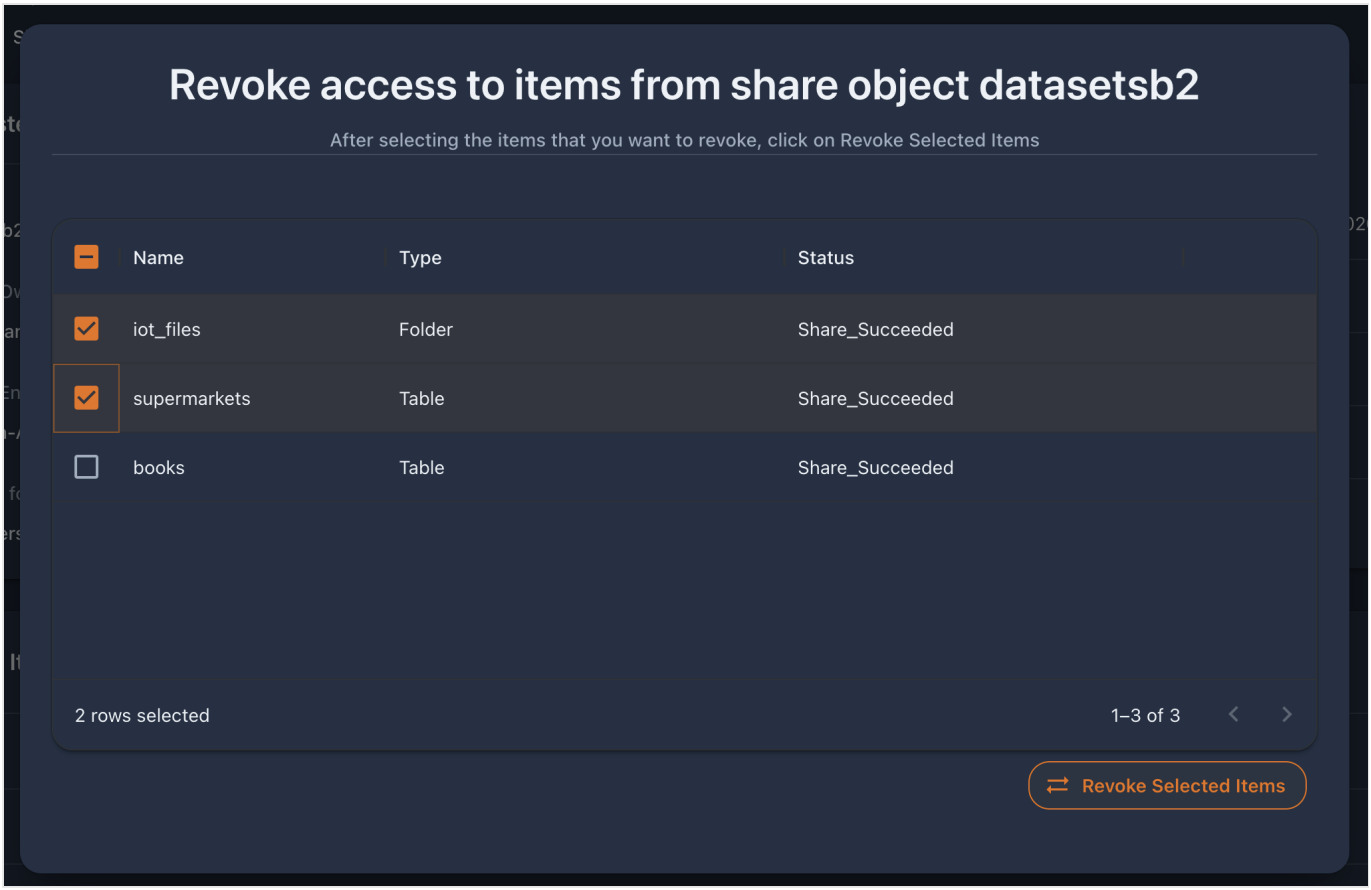


Upon successful re-apply process, the share items health status will revert back to a `Healthy` status with an updated timestamp.

4.5.8 **Revoke Items**

Both approvers and requesters can click on the button **Revoke items** to remove the share grant from chosen items.

It will open a window where multiple items can be selected for revoke. Once the button "revoke selected items" is pressed the consequent revoke task will be triggered.

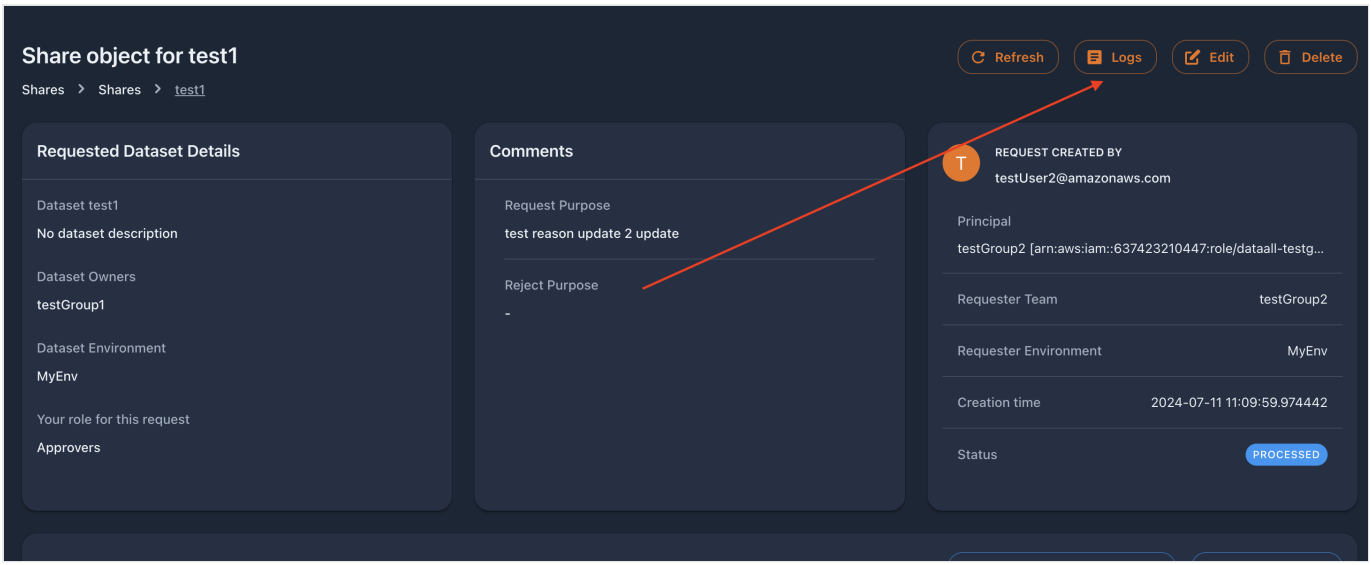


Proactive clean-up

In every revoke task, data.all checks if there are no more shared folders or tables in a share request. In such case, data.all automatically cleans up any unnecessary S3 access point or Lake Formation permission.

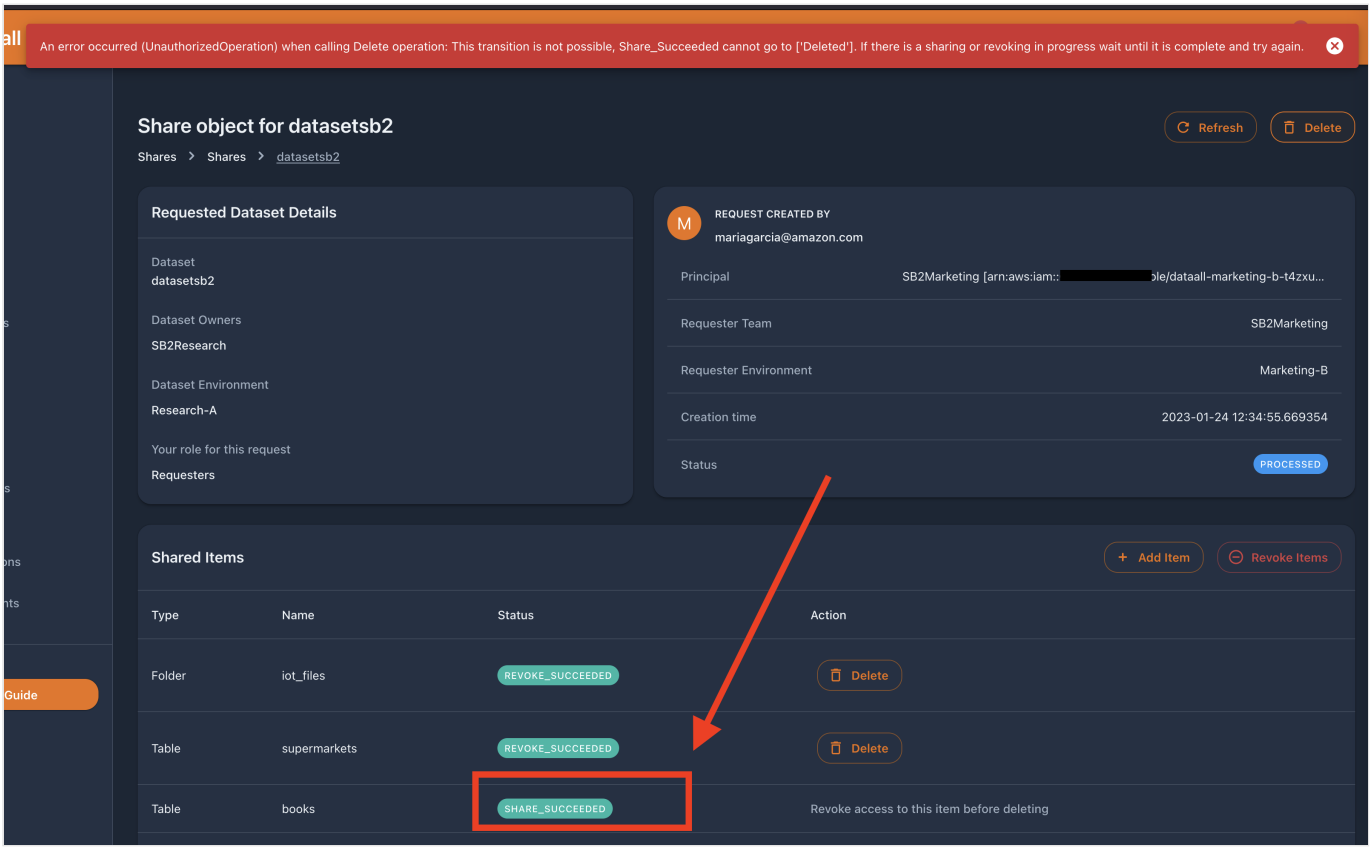
4.5.9 View Share Logs

For the share Approvers the logs of share processor are available via Data.all UI. To view logs of the latest share processor run, click **Logs** button in right upper conner of the Share View page.



4.5.10 **Delete share request**

To delete a share request, it needs to be empty from shared items. For example, the following request has some items in `SHARE_SUCCEEDED` state, therefore we receive an error. Once we have revoked access to all items we can delete the request.



4.5.11 **AWS data sharing technical details**

Here is a brief explanation of how each type of sharing mechanism is implemented in data.all. It is important to understand what really happens in AWS when dealing with downstream integrations that will consume shared data.

S3 Bucket sharing

In this type of share the permissions are granted to the IAM role specified in the request as principal. It can be either a data.all team IAM role or an external role defined as consumption role.

When processing a sharing task for an S3 Bucket, data.all will: 1. Update the S3 Bucket policy to add permissions to the principal IAM role 2. Create/Update the IAM policy "Share policy" that grants IAM permissions to the requested S3 bucket and KMS key. Attach this policy to the principal IAM role. 3. (If the Bucket is encrypted using a KMS key) Update the KMS Key policy to add permissions to the principal IAM role

Glue Table sharing

In this type of share the permissions are granted to the IAM role specified in the request as principal. It can be either a data.all team IAM role or an external role defined as consumption role.

When processing a sharing task for a Glue Table, data.all will: 1. Create a Glue database in the target account with name of the original database plus the suffix `_shared`. This database will be re-used if other share requests for the same source databaser are processed for other principals in the same environment. 2. (If the share is cross-account) Revoke IAMAllowedPrincipal permissions from the table to ensure Lake Formation is used in the management of the table access and update LakeFormation to use Version 3 if not already >=3 3. Grant Lake Formation permissions on the original database and table to the IAM principals

in the target. If the share is cross account this step will create a RAM invitation that data.all will identify and accept. 4. Create a resource link table from the original database table to the `_shared` database in the target account 5. Grant Lake Formation permissions to the resource link table for the IAM principals.

S3 Prefix sharing (Folders)

In this type of share the permissions are granted to the IAM role specified in the request as principal. It can be either a data.all team IAM role or an external role defined as consumption role.

When processing a sharing task for a Folder, data.all will: 1. Update the Dataset Bucket policy to allow access point sharing. This is a one-time operation 2. Create/Update an S3 Access Point and its policy granting permissions to the requested S3 prefix (folder) in the bucket for the principal IAM role. 3. Create/Update the IAM policy "Share policy" that grants IAM permissions to the S3 Access Point and KMS key. Attach this policy to the principal IAM role. 4. (If the Bucket is encrypted using a KMS key) Update the KMS Key policy to add permissions to the principal IAM role

Redshift Table sharing

In this type of share the permissions are granted to the Redshift role in the Redshift namespace specified in the request.

When processing a sharing task for a Redshift table, data.all will: 1. In the source namespace, create a Redshift datashare. Add requested schema and tables to the datashare. 2. Grant access to the datashare for the consumer namespace (same account) or for the consumer AWS account (cross account) 3. (If cross-account share) Authorize and associate datashare with the target namespace 4. In the target namespace, create local database for the datashare and grant permissions to the principal Redshift role. 5. In the target namespace, create external schema in local database and grant usage permissions to the principal Redshift role. 6. For the local database and for the external schema, grant select access to the requested table to the principal Redshift role.

4.5.12 Consume shared data

Knowing what we know form the previous section we can now define some ways of consuming the shared data for each type of shareable item.

S3 Bucket sharing

For S3 bucket sharing, IAM policies, S3 bucket policies, and KMS Key policies (if applicable) are updated to enable sharing of the S3 Bucket resource. Therefore, we can use S3 API calls to access the data referring the Bucket directly. We need to assume or use the credentials of the principal IAM role used in the share request (team IAM role or consumption IAM role).

Here is an example using the AWS CLI:

```
aws s3 ls s3://<BUCKET_NAME>
```

Glue Table sharing

Glue tables are shared using AWS Lake Formation, therefore any service that reads Glue tables and integrates with Lake Formation is able to consume the data.

We need to assume or use the credentials of the principal IAM role used in the share request (team IAM role or consumption IAM role).

S3 Prefix sharing (Folders)

For the case of folders, the underlying sharing mechanism used is S3 Access Points. You can read data inside a prefix executing API calls to the S3 access point.

We need to assume or use the credentials of the principal IAM role used in the share request (team IAM role or consumption IAM role).

For example, we could use the AWS CLI with the following access point:

```
aws s3 ls arn:aws:s3:<SOURCE_REGION>:<SOURCE_AWSACCOUNTID>:accesspoint/<DATASETURI>-<REQUESTER-TEAM>/<FOLDER_NAME>/
```

Redshift Table sharing

Redshift tables are shared through Redshift datashares and the principal of the share request is a Redshift role. Thus, we can consume data accessing the Redshift Query editor or other applications that consume from Redshift with a user that has access to the Redshift role.

4.5.13 Email Notification on share requests

In data.all, you can enable email notification to send emails to requesters and approvers of a share request. Email notifications are triggered during all share workflows - Share Submitted, Approved, Rejected, Revoked.

The content sent in email notification is similar to the UI based notification.

For example the email body will look like,

```
User <USERNAME> <SHARE_ACTION> share request for dataset <DATASET_NAME>

where <SHARE_ACTION> corresponds to "submitted", "approved", "revoked", "rejected"
```

Note - In order to enable email notification, you need to configure it in `config.json` and setup the AWS services needed for during the deployment phase. Please review steps for setting up email notification on data.all webpage in the `Deploy to AWS` section

4.6 Metadata Forms

Introduction

Metadata forms allow data.all users to add structured contextual information to various entities in the data.all platform. By creating and attaching metadata forms, users can standardize and enrich metadata in a customizable way.

Metadata forms serve several key purposes:

- Improve data discovery by enabling more consistent, complete, and meaningful metadata
- Capture domain-specific or organizational metadata standards
- Streamline metadata management workflows
- Search among all entities in data.all based on attached metadata

Metadata Forms visibility

Visibility setting defines who can view and attach a metadata form to an entity.

- Global visibility means the metadata form is visible and attachable to any entity by all users across the platform
- Organization/Environment-Wide visibility limits the form to a specific organization/environment - it can only be seen by members of this organization/environment and attached to entities in that organization/environment
- Team-Only visibility restricts the form to just members of a specific team, but does not restrict to which entities it can be attached to

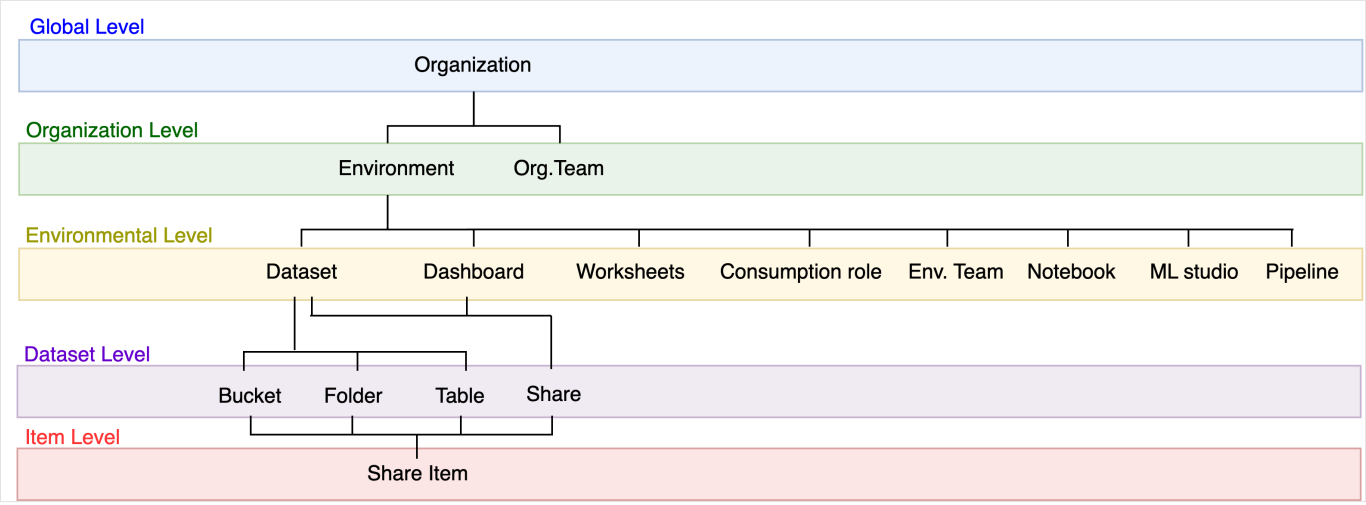
The same visibility restrictions apply to the attached metadata form. For example, if userA can see Metadata Form A, but can't see Metadata Form B, and both of these metadata forms are attached to a dataset; then when userA opens the dataset, they will see the attached metadata form A but not the attached metadata form B.

Metadata Form lifecycle and usage

1. User given the permission from data.all administrators can create metadata forms with the Global visibility and visibility for their teams. Owners and Admins of Organizations and Environments can create metadata forms with Environment-Wide and Organization-wide visibility for owned entities.
2. Once form is created, the owners can add enforcement rules (see the section below).
3. All changes in metadata form can be performed only by its owner.
4. In case owner deletes metadata form all its attached entities are deleted in cascade mode.
5. Metadata form can be attached to an entity by the user with sufficient permissions. Permissions are given by the owner or admin of the entity (in case of Environments and Organizations). In case of Datasets the owner and steward teams have these permissions and also the teams with whom the dataset was shared.
6. Attached metadata forms can be edited or deleted by any user with sufficient permissions.

Metadata Forms levels and enforcement

Metadata forms can be obligatory to fill in on different levels. User can select the metadata form and entity types, that should have this form attached. Enforcement affects selected entity types on all lower levels hierarchically.



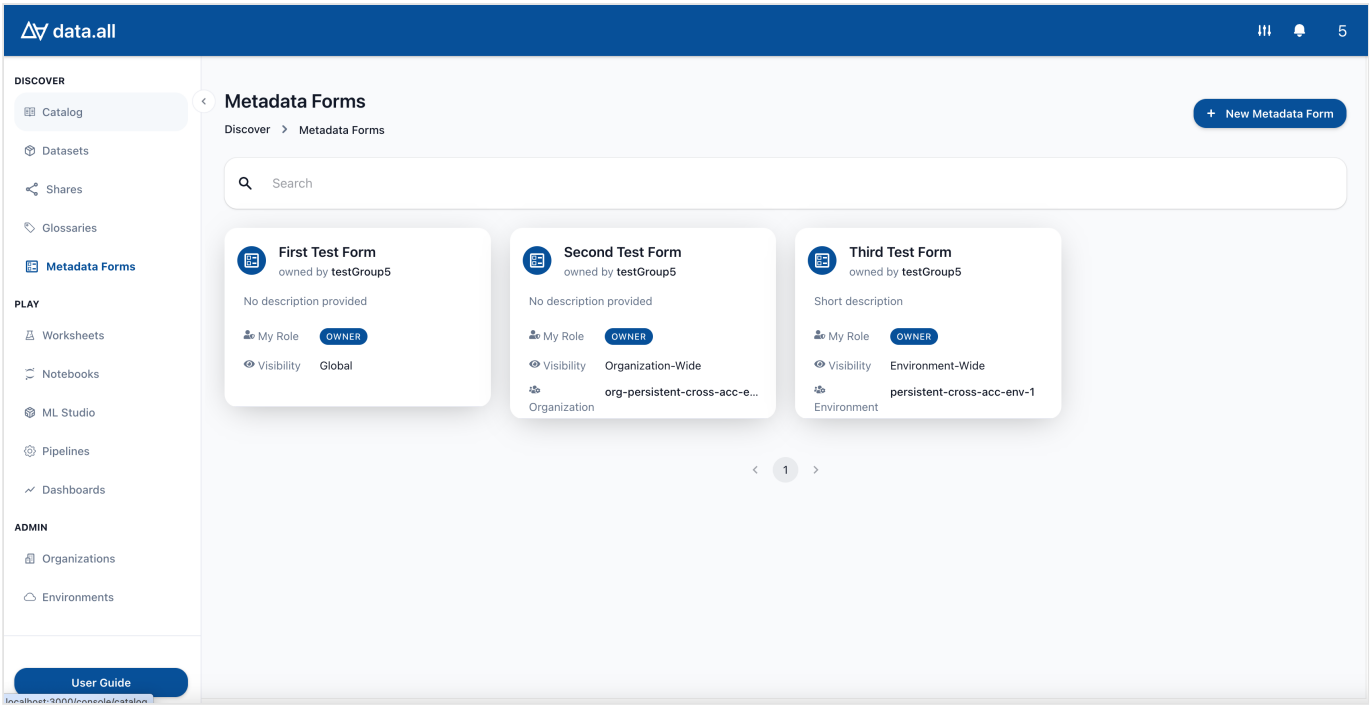
Who can enforce:

- Data.all admins can enforce any form on any level across the platform. They have full control over metadata form enforcement.
- Owners/admins of a data.all entity can enforce forms for these levels and levels below in the hierarchy. For example, an org admin can enforce a form for the org, all teams in that org, all environments in the org, all datasets in those environments, etc.
- Share approvers and requestors can enforce forms for a specific share they are involved with. However, they can only delete enforcement rules they created themselves - they cannot delete rules created by others

So in summary, enforcement capabilities cascade along with administrative privileges in the hierarchy. Global admins have full control, org/env admins can enforce for their sphere and below, dataset admins for the datasets and items in it, and share requesters and approvers for a specific share.

View Metadata Forms By clicking Metadata Forms in the Discovery section of the left side pane users can open a list of metadata forms visible for them. The criteria for visibility:

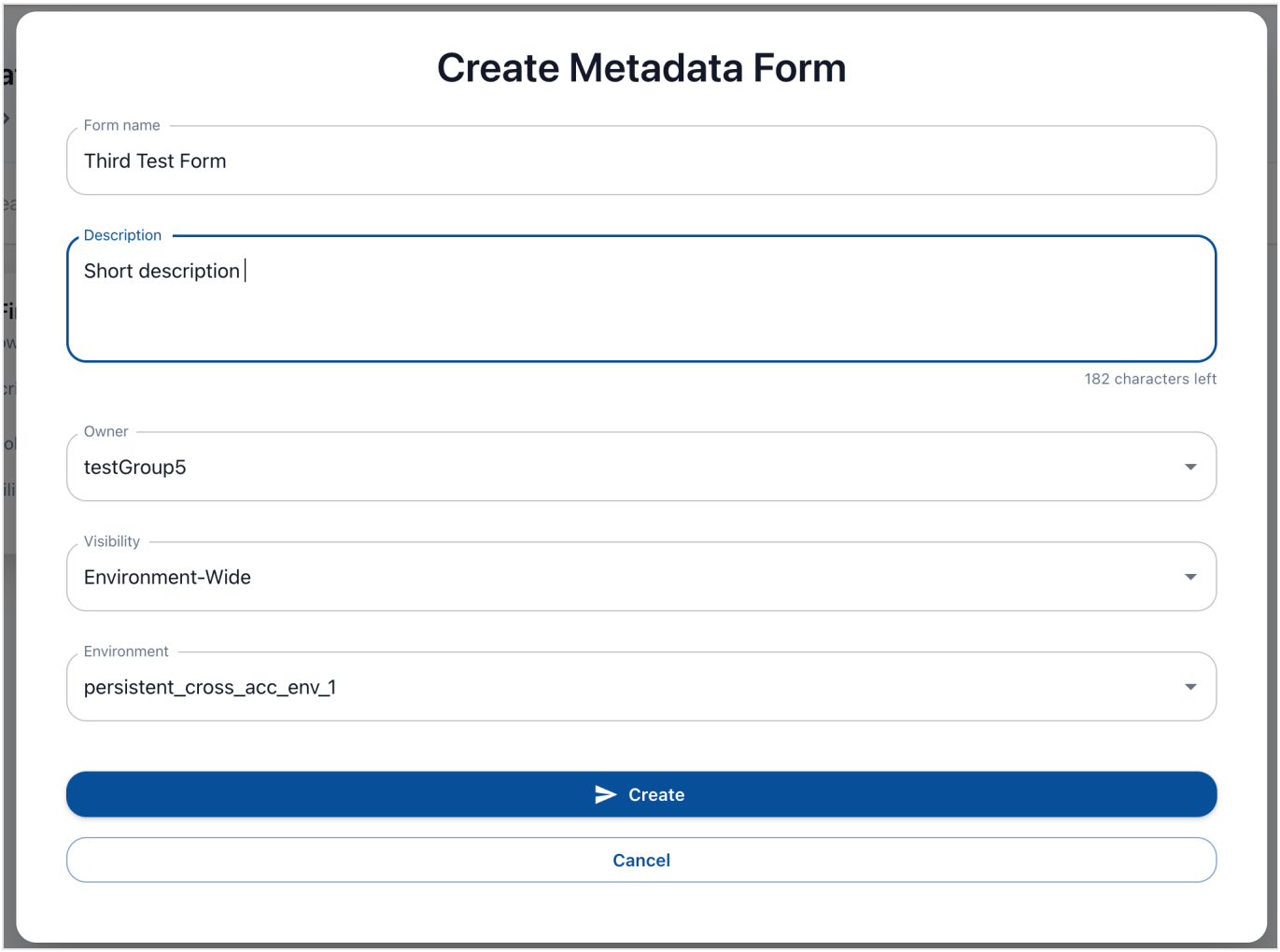
1. The group, that the user belong to, is an owner of the metadata form.
2. Metadata form has Global visibility.
3. Metadata form has Team-Only visibility and the user is a member of this team.
4. Metadata form has Environment-Wide of Organization-Wide visibility and the user has access to this environment/organization.
5. Administrators can view all metadata forms.



Create Metadata Form

To create a new metadata form:

1. Navigate to the Metadata Forms page under the Discovery section.
2. Click the "New Metadata Form" button in the top right corner.
3. In the pop-up dialog, enter a Name and Description for the form.
4. Select the Owner Team responsible for managing this form.
5. Choose a Visibility level to control access to the form. Options are Team Only, Environment-Wide, Organization-Wide, or Global.
6. If Visibility is not Global, select the Organization, Environment ot Team to scope the form.
7. Click "Create" to generate the form.

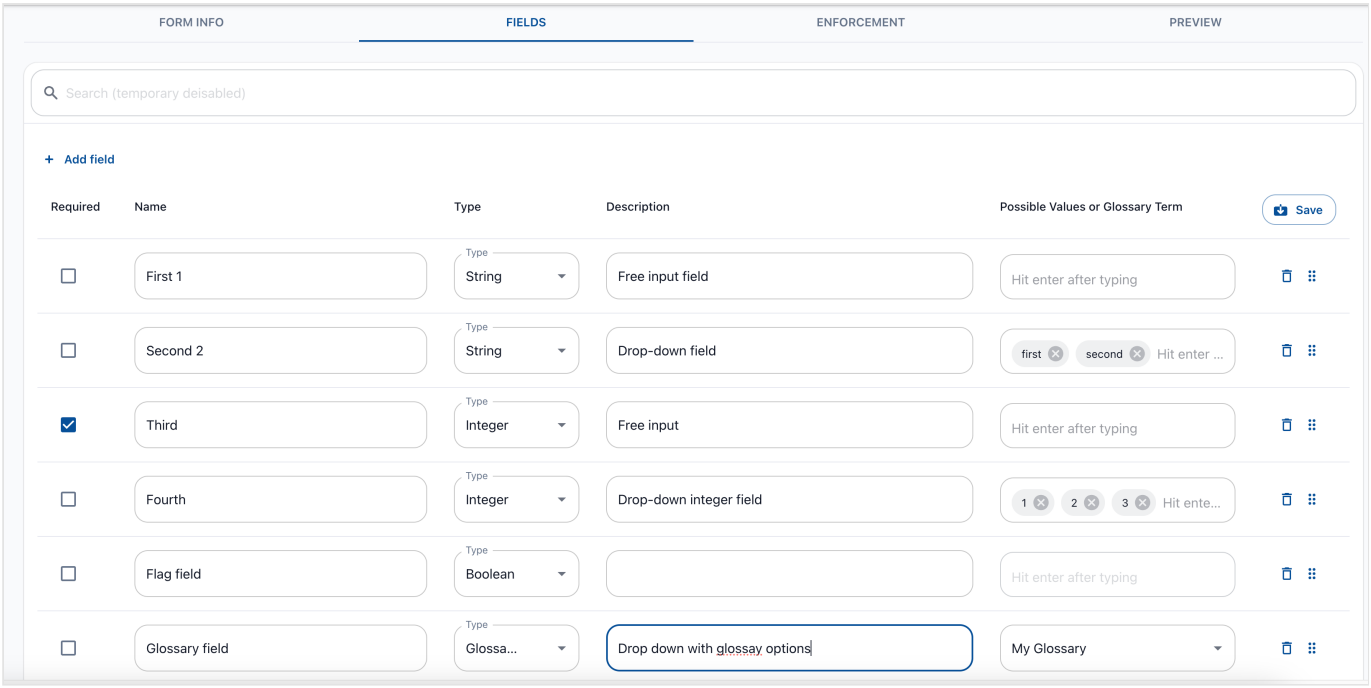


You will be redirected to the metadata form page. It contains details overview, instruments to edit form fields and preview tab.

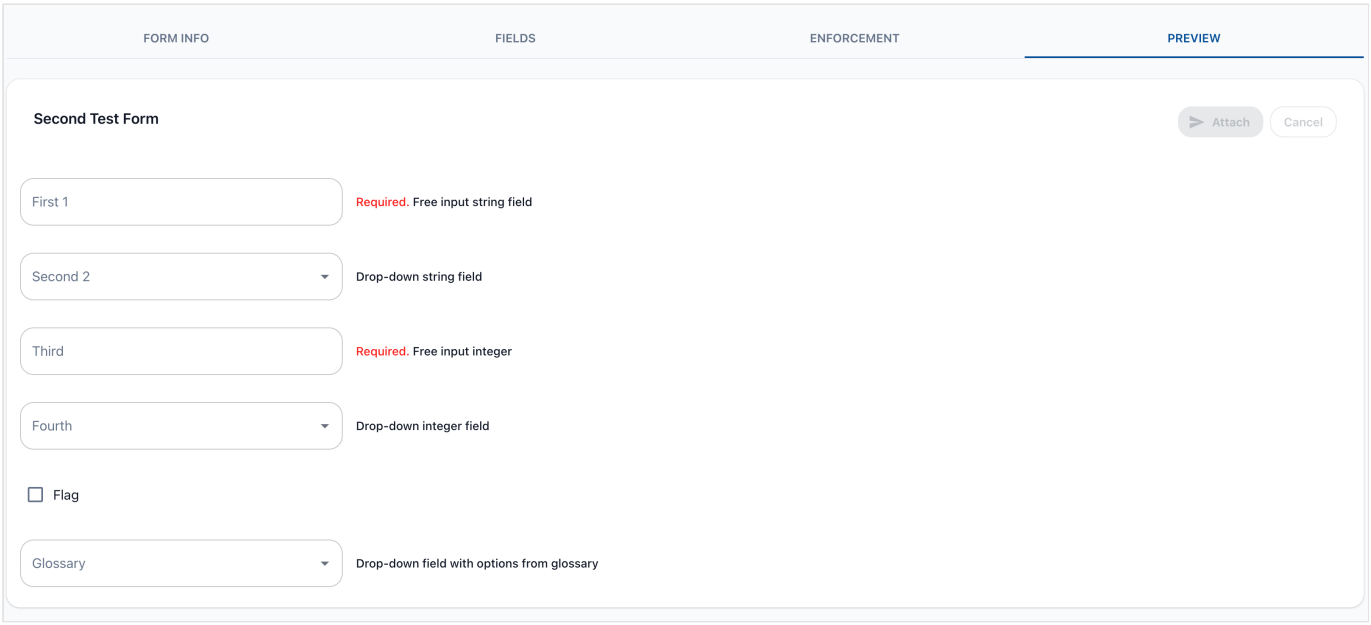
In the field editor, use the "Edit" button in the upper right corner of the table to update fields. When the editor mode is enabled, the user can

1. add fields using the button '+ Add field' in the left upper corner of the table;
2. delete fields using 'bin' icon in the end of the table row. When the user pushes the button, the row is disabled, which indicates it is marked for deletion, but user can restore the field by clicking the button (now with "refresh") again;
3. arrange fields using drag and drop, fields will be shown in rendered metadata form in the order they appear in this list (from top to bottom);
4. edit all parameters of the field.

When finished, click "Save" to apply changes.



In the "Preview" tab user can see, how the form will be rendered.



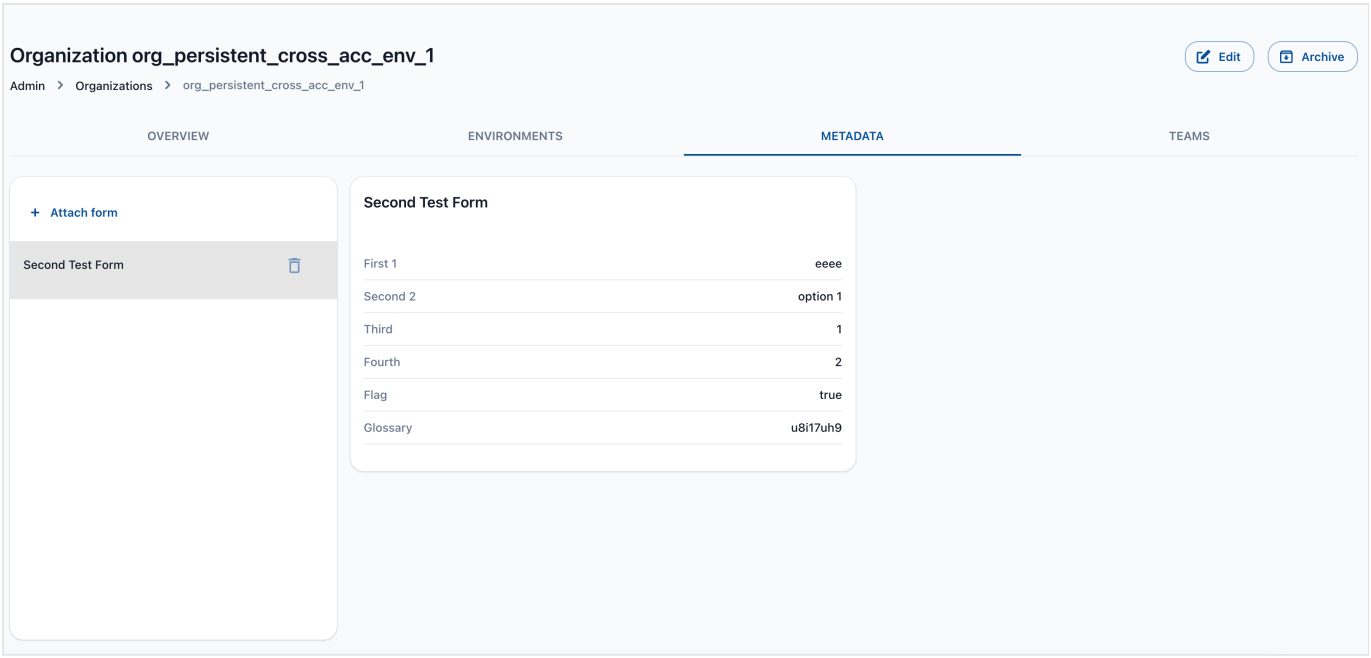
The new metadata form can now be attached to entities and filled out by users with access. Forms can be edited later to modify fields and settings.

User can delete the form with button "Delete" in the upper right corner of the form view page.

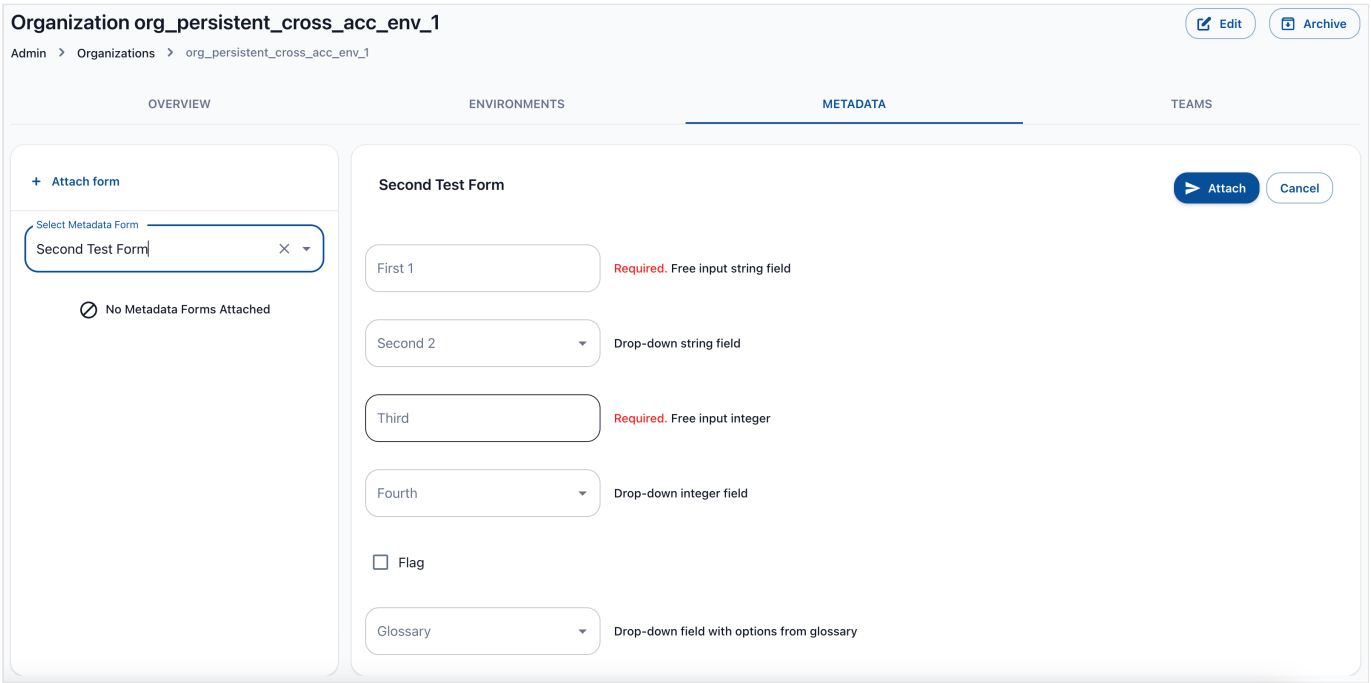
Attach Metadata Form

User with required access can attach metadata form to Organization, Environment or Dataset. To attach new metadata form or view already attached use the tab "Metadata" on entity view page.

In the column on the left all attached metadata forms are listed. When the user clicks on the form, its content appears on the right. The attached form can be deleted by click on "bin" icon next to form name in the list.



If user has permission to attach metadata forms to this entity, the button "+ Attach Form" appears over the attached metadata form list. After user clicks this button and selects the available form from drop-dow list, they can fill in the form displayed on the right. After all required fields are filled, press "Attach" button in the right upper corner of the editing area.



Permissions Summary

Create Metadata Form: User given the permission from data.all administrators can create metadata forms with the Global visibility and visibility for their teams. Owners and Admins of Organizations and Environments can create metadata forms with Environment-Wide and Organization-wide visibility for owned entities.

Edit and delete Metadata Form: These actions available only for owners of the metadata form.

Enforce Metadata Form Usage:

- Data.all admins can enforce any form on any level across the platform. They have full control over metadata form enforcement.
- Owners/admins of a data.all entity can enforce forms for these levels and levels below in the hierarchy. For example, an org admin can enforce a form for the org, all teams in that org, all environments in the org, all datasets in those environments, etc.
- Share approvers and requestors can enforce forms for a specific share they are involved with. However, they can only delete enforcement rules they created themselves - they cannot delete rules created by others

Attach Metadata Form:

- Users must have relevant permissions for target entity (ATTACH_METADATA_FORM, given by entity admin)
- Users must have access to view the target metadata form (see paragraph "View Metadata Forms")
- Target entity must be in target metadata form scope (see paragraph "Metadata Forms Levels")

Delete Attached Metadata Form: Users must have relevant permissions for target entity (ATTACH_METADATA_FORM, given by entity admin)

5. Play

5.1 Worksheets

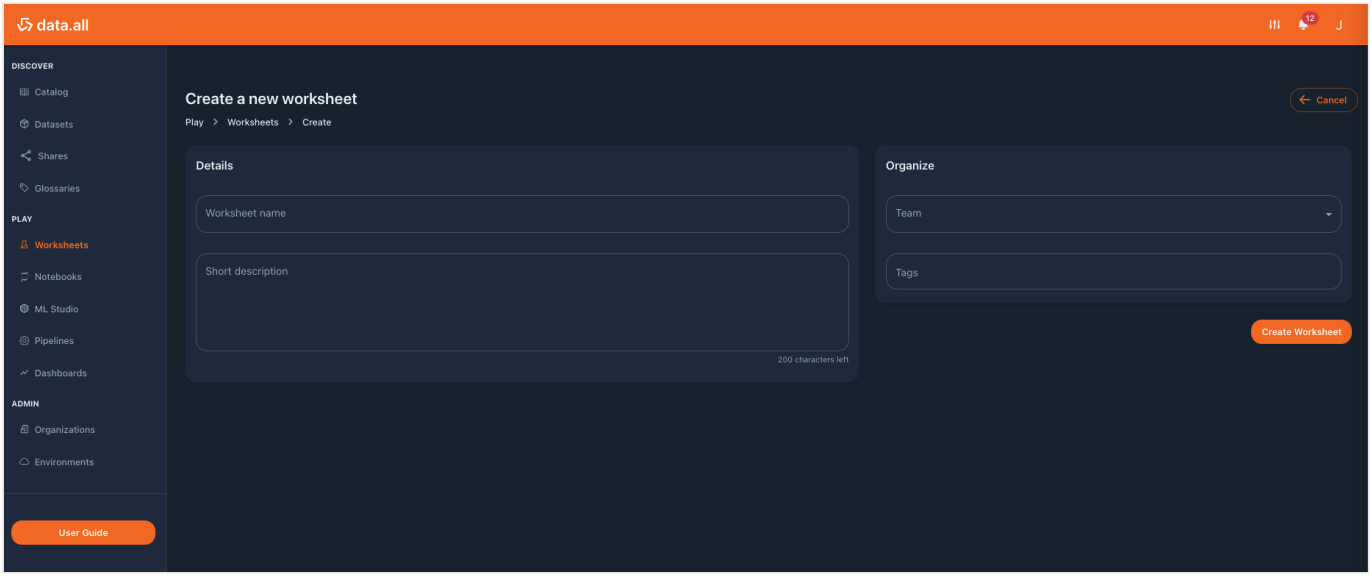
data.all offers a rich editor to write SQL queries and explore data. It is Athena on the backend that runs our queries on environments where our teams have been onboarded.

5.1.1 NEW Create a Worksheet

On the left pane under **Play** click on **Worksheets** to go to the Worksheet menu. Here you will find all Worksheets owned by your teams.

> **Shared queries = Seamless Collaboration**
>
> Check, learn from and collaborate with other members of your team to improve your analyses and get insights from your data, directly from data.all worksheets. No need to send queries by email, no need to create views :)

To create a new worksheet click on the **Create** button in the top right corner and fill the Worksheet form:



Field	Description	Required	Editable	Example
Worksheet name	Name of the worksheet	Yes	Yes	PalmDor
Short description	Short description about the worksheet	No	Yes	Query used to retrieve Palm D'or winners
Team	Team that owns the worksheet	Yes	No	DataScienceTeam
Tags	Tags	No	Yes	adhoc

> **No AWS resources**
>
> When we are creating a worksheet we are NOT deploying AWS resources. We don't provision clusters, we are not creating tables or views. We simply store the query in data.all database and we run it serverlessly on AWS Athena.

5.1.2 Edit worksheet metadata

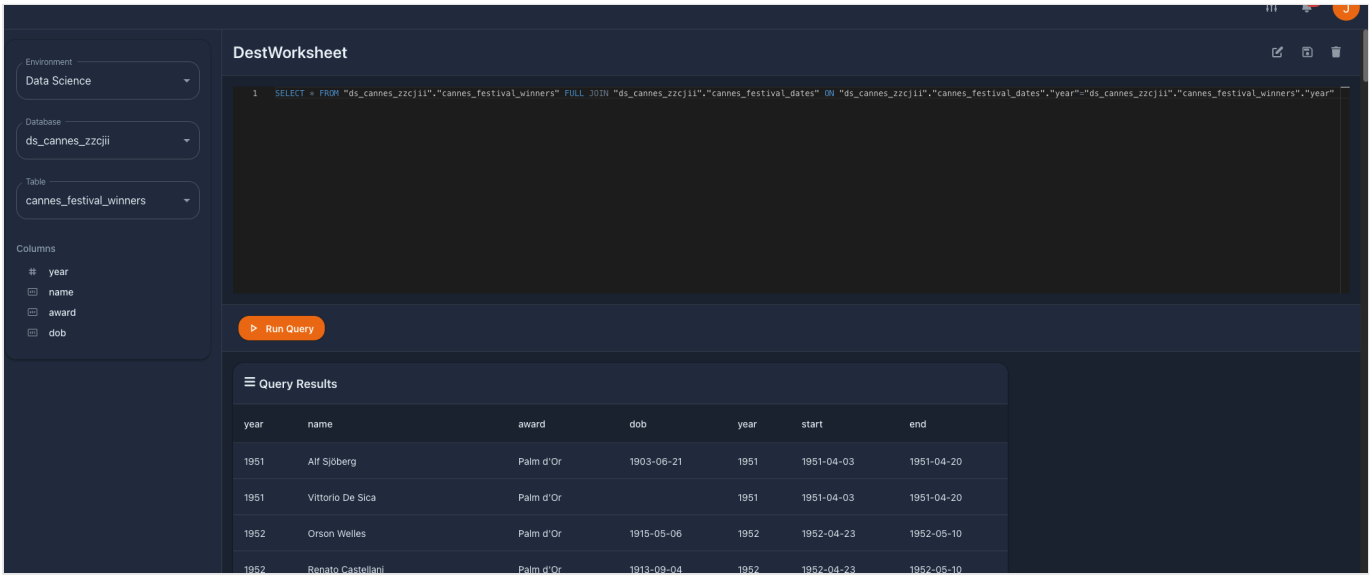
Select a worksheet and click on the pencil icon to edit the metadata of the worksheet. This includes worksheet name, description and tags. The ownership of the worksheet, its team, is not editable.

5.1.3 Delete a worksheet

Next to the edit button, there are 2 other buttons. To delete a worksheet click on the trash icon one. Worksheets are not AWS resources, they are a data.all construct whose information is stored in the data.all database. Thus, when we delete a worksheet we are not deleting AWS resources or CloudFormation stacks.

5.1.4 Write and save your queries

Select your worksheet and choose any of the environments, datasets and tables of your team to list column information. In the query editor write your SQL statements and click on **Run Query** to get your query results. Error messages coming from Athena will pop-up automatically.



If you want to save the current query for later or for other users, click on the **save** icon (between the edit and the delete buttons).

More than just SELECT

Worksheets can be used for data exploration, for quick ad-hoc queries and for more complicated queries that require joins. As far as you have access to the joined datasets you can combine information from multiple tables or datasets. Check the docs for more information on AWS Athena SQL syntax.

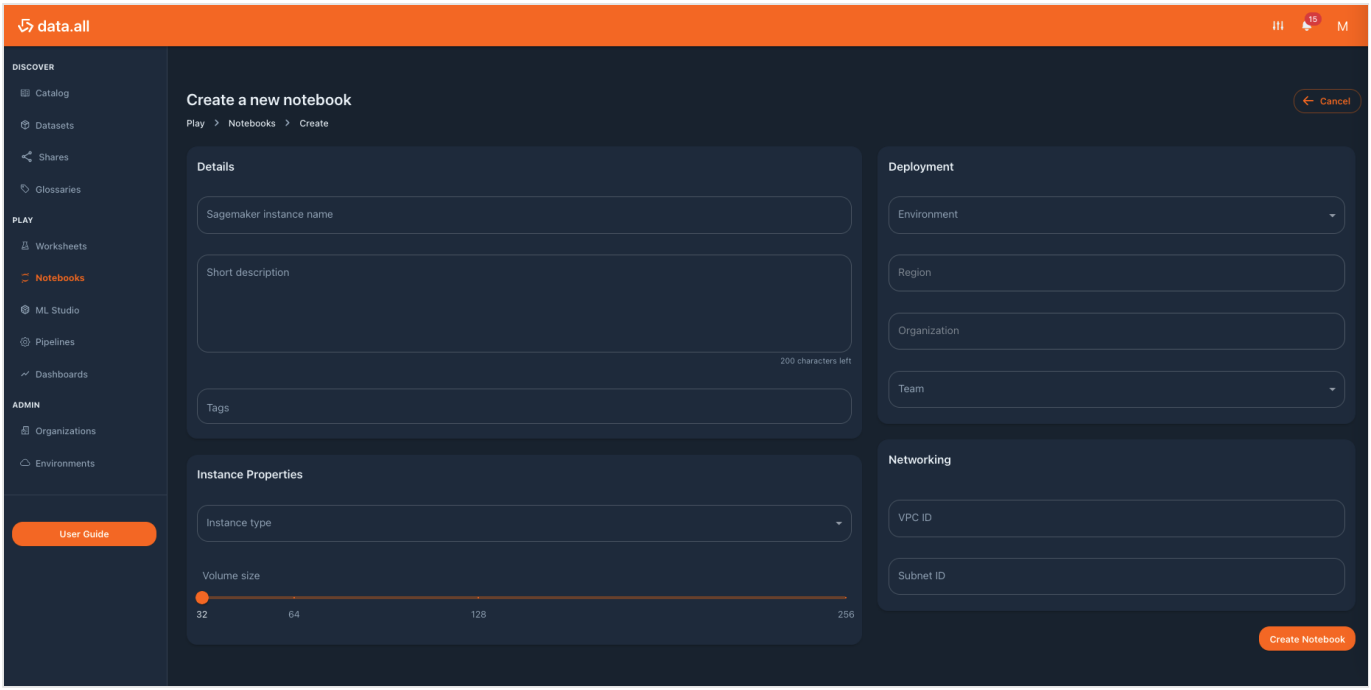
5.2 Notebooks

Data practitioners can experiment machine learning algorithms spinning up Jupyter notebook with access to all your datasets. data.all leverages Amazon SageMaker instance to access Jupyter notebooks.

5.2.1 NEW Create a Notebook

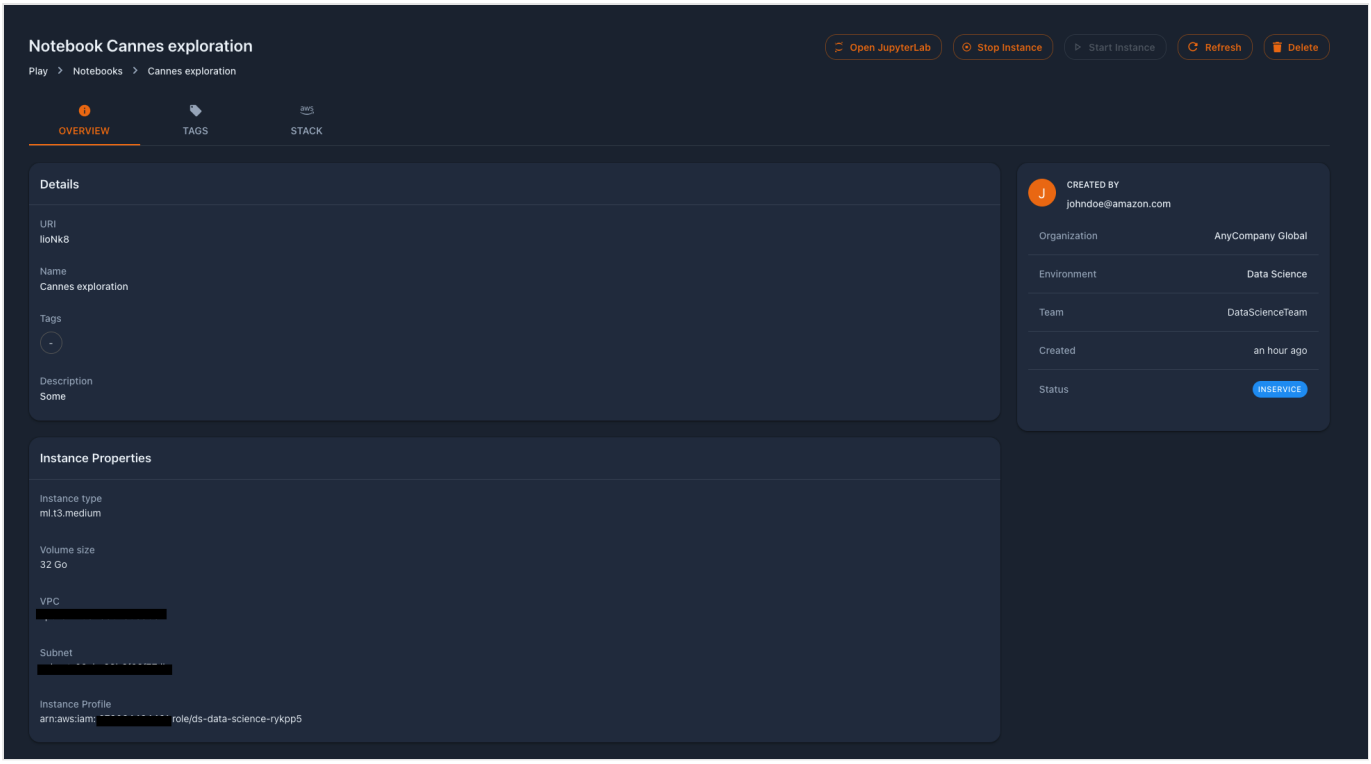
> **Pre-requisites**
>
> To use Notebooks you need to introduce your own VPC ID or create a Sagemaker Studio domain inside a VPC (read the docs). Provisioning the notebook instances inside a VPC enables the notebook to access VPC-only resources such as EFS file systems.

To create a Notebook, go to Notebooks on the left pane and click on the **Create** button. Then fill in the following form:



Field	Description	Required	Editable	Example
Sagemaker instance name	Name of the notebook	Yes	No	Cannes Project
Short description	Short description about the notebook	No	No	Notebook for Cannes exploration
Tags	Tags	No	No	deleteme
Environment	Environment (and mapped AWS account)	Yes	No	Data Science
Region (auto-filled)	AWS region	Yes	No	Europe (Ireland)
Organization (auto-filled)	Organization of the environment	Yes	No	AnyCompany EMEA
Team	Team that owns the notebook	Yes	No	DataScienceTeam
VPC Identifier	VPC provided to host the notebook	No	No	vpc-.....
Subnets	Subnets provided to host the notebook	No	No	subnet-....
Instance type	[ml.t3.medium, ml.t3.large, ml.m5.xlarge]	Yes	No	ml.t3.medium
Volume size	[32, 64, 128, 256]	Yes	No	32

If successfully created we can check its metadata in the **Overview** tab. Unlike other data.all resources, Notebooks are non-editable.



5.2.2 Check CloudFormation stack

In the **Stack** tab of the Notebook, is where we check the AWS resources provisioned by data.all as well as its status. As part of the Notebook CloudFormation stack deployed using CDK, data.all will deploy:

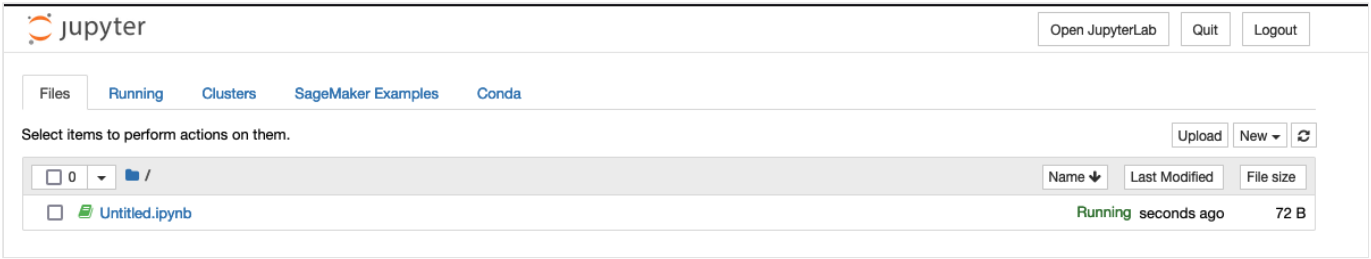
1. AWS EC2 Security Group
2. AWS SageMaker Notebook Instance
3. AWS KMS Key and Alias

5.2.3 Delete a Notebook

To delete a Notebook, simply select it and click on the **Delete** button in the top right corner. It is possible to keep the CloudFormation stack associated with the Notebook by selecting this option in the confirmation delete window that appears after clicking on delete.

5.2.4 Open JupyterLab

Click on the **Open JupyterLab** button of the Notebook window to start writing code on Jupyter Notebooks.



5.2.5 Stop/Start instance

As we briefly commented, data.all uses AWS SageMaker instances to access Jupyter notebooks. Be frugal and stop your instances when you are not developing. To do that, close the Jupyter window and click on **Stop Instance** in the Notebook buttons. It takes a couple of minutes, just refresh and check the Notebook Status in the overview tab. It should end up in `STOPPED`.

> **Save money, stop your instances**
>
> This feature allows users to easily manage their instances directly from data.all UI.

Same when you are coming back to work on your Notebook, click on **Start instance** to start the SageMaker instance. In this case the Status of the notebook should first be `PENDING` and once the instance is ready, `INSERVICE`.

5.2.6 Create Key-value tags

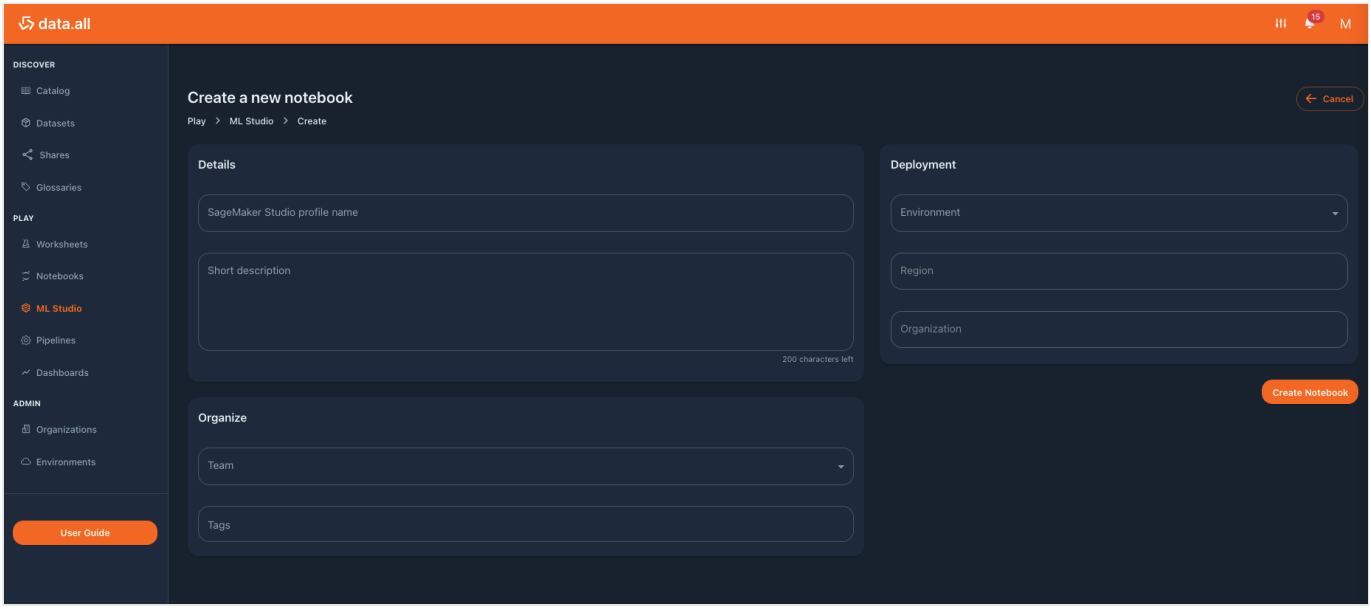
In the **Tags** tab of the notebook window, we can create key-value tags. These tags are not data.all tags that are used to tag datasets and find them in the catalog. In this case we are creating AWS tags as part of the notebook CloudFormation stack. There are multiple tagging strategies as explained in the documentation.

5.3 ML Studio

With ML Studio Profiles we can add users to our SageMaker domain and open Amazon SageMaker Studio. The SageMaker Studio domain is created as part of the environment stack.

5.3.1 NEW Create an ML Studio profile

To create a new ML Studio profile, go to ML Studio on the left side pane and click on Create. Then fill in the creation form with its corresponding information.



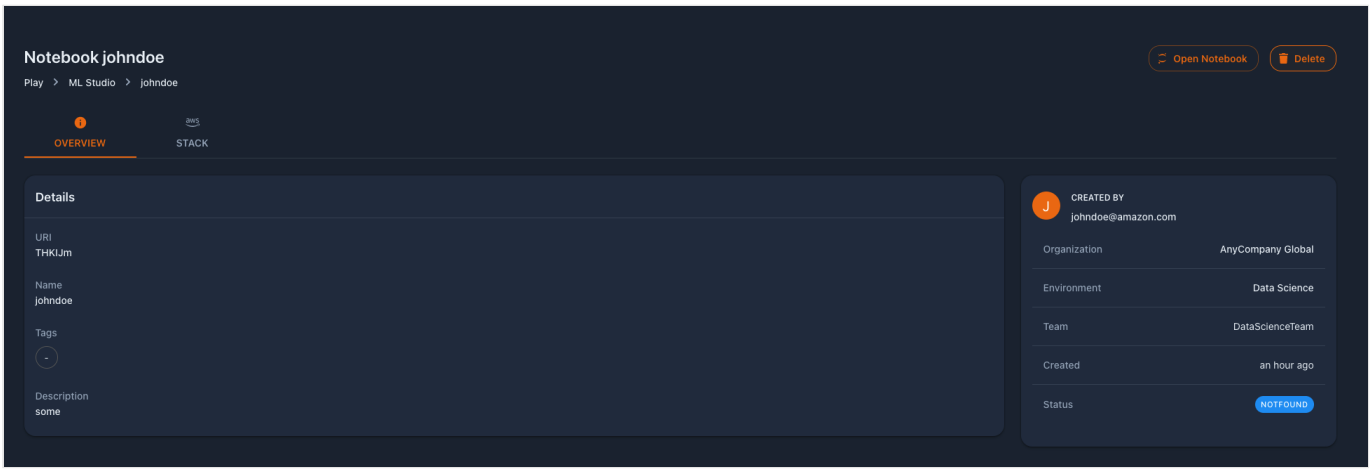
Field	Description	Required	Editable	Example
Sagemaker Studio profile name	Name of the user to add to SageMaker domain	Yes	No	johndoe
Short description	Short description about the user profile	No	No	Notebook for Cannes exploration
Tags	Tags	No	No	deleteme
Environment	Environment (and mapped AWS account)	Yes	No	Data Science
Region (auto-filled)	AWS region	Yes	No	Europe (Ireland)
Organization (auto-filled)	Organization of the environment	Yes	No	AnyCompany EMEA
Team	Team that owns the notebook	Yes	No	DataScienceTeam

5.3.2 Check CloudFormation stack

In the **Stack** tab of the ML Studio Profile, is where we check the AWS resources provisioned by data.all as well as its status. As part of the CloudFormation stack deployed using CDK, data.all will deploy some CDK metadata and a SageMaker User Profile.

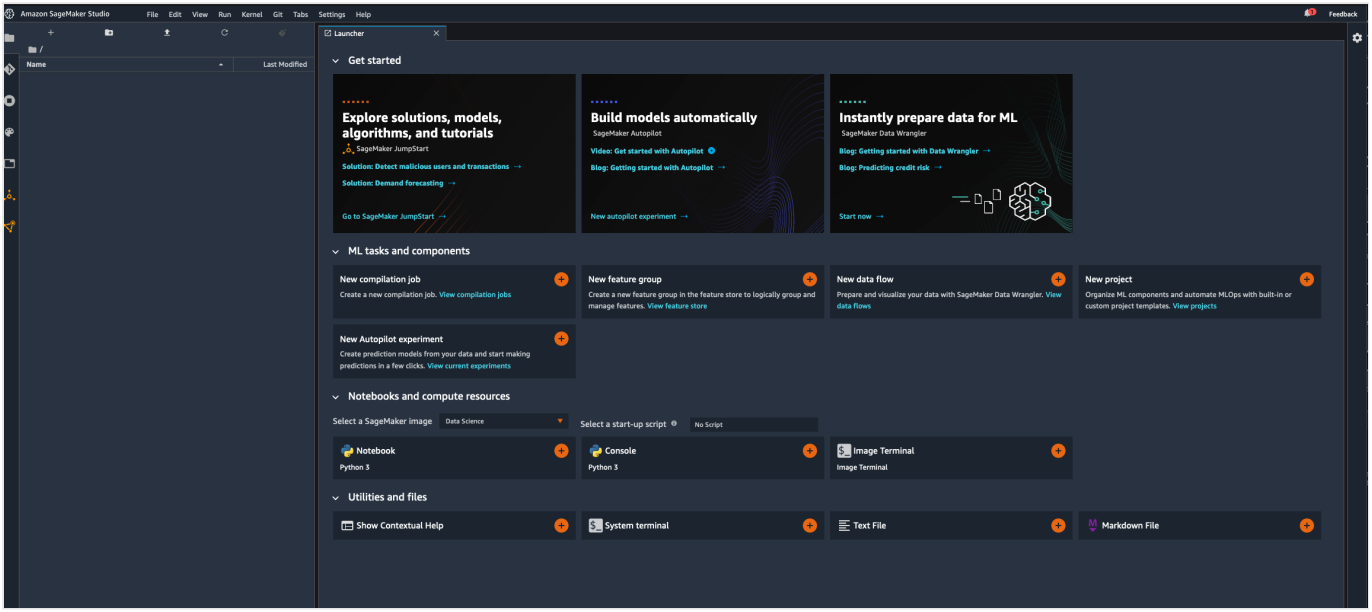
5.3.3 Delete an ML Studio user

To delete a SageMaker user, simply select it and click on the **Delete** button in the top right corner. It is possible to keep the CloudFormation stack associated with the User by selecting this option in the confirmation delete window that appears after clicking on delete.



5.3.4 Open Amazon SageMaker Studio

Click on the **Open ML Studio** button of the ML Studio notebook window to open Amazon SageMaker Studio.

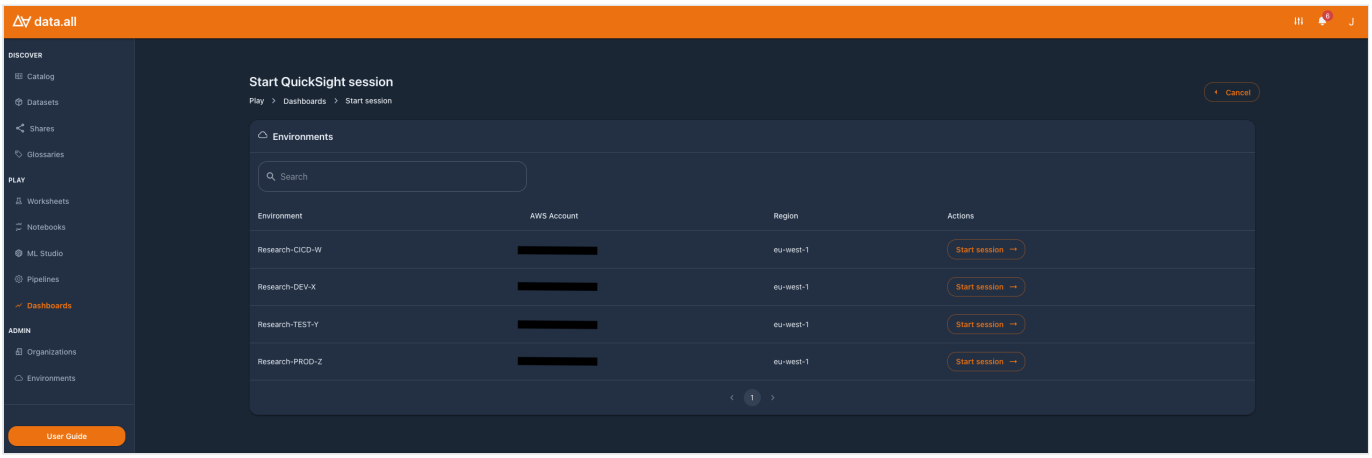


5.4 Dashboards

Data.all connects with Amazon Quicksight to allow users to quickly visualize and analyse their data.

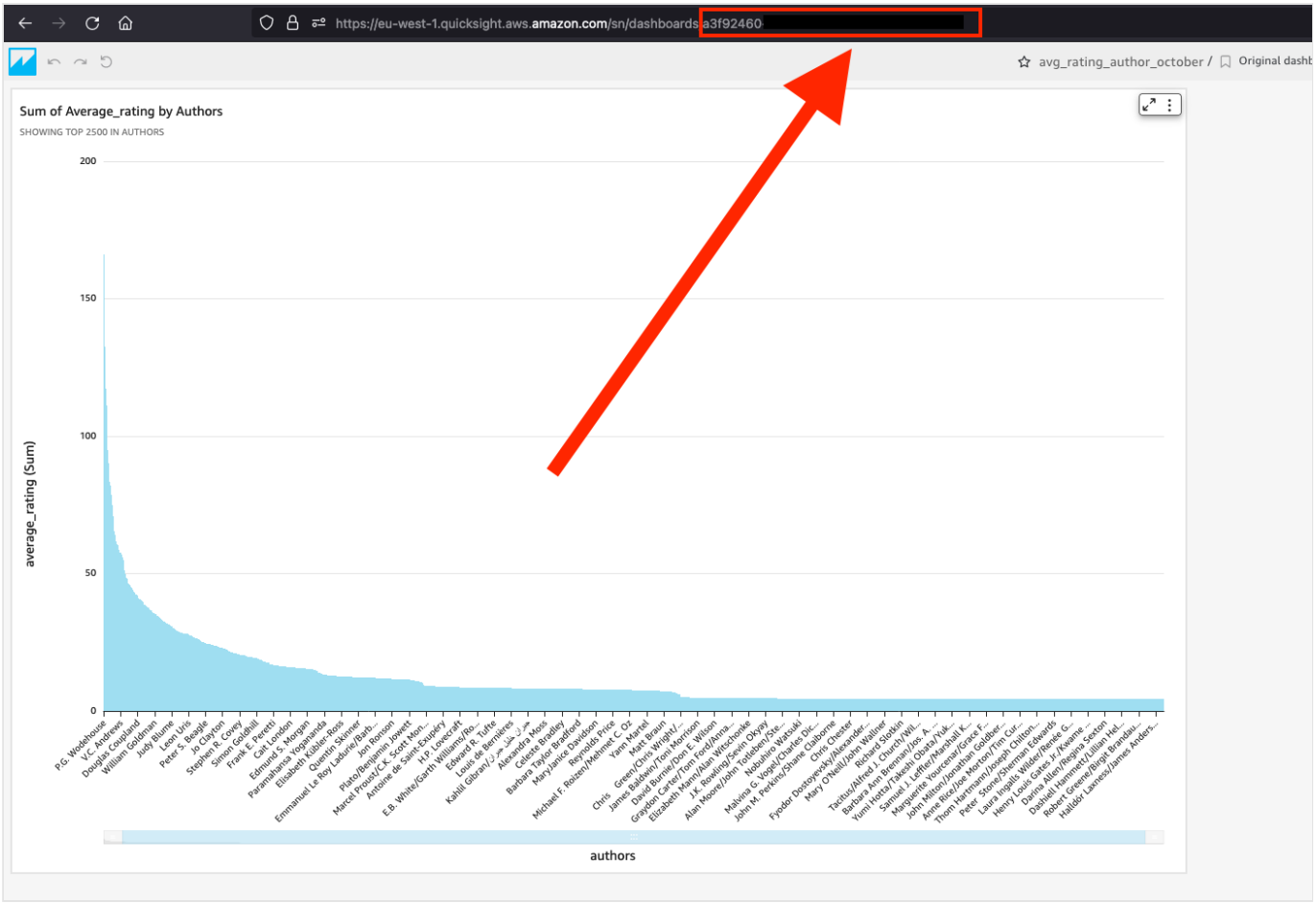
5.4.1 **Start Quicksight session**

Go to the Dashboards menu of data.all and click on the orange "Quicksight" button in the top right corner. It will redirect you to the following page, in which you can start a Quicksight session in one of your environment accounts. If Dashboards are not enabled in the environment, an error message will appear on the screen.

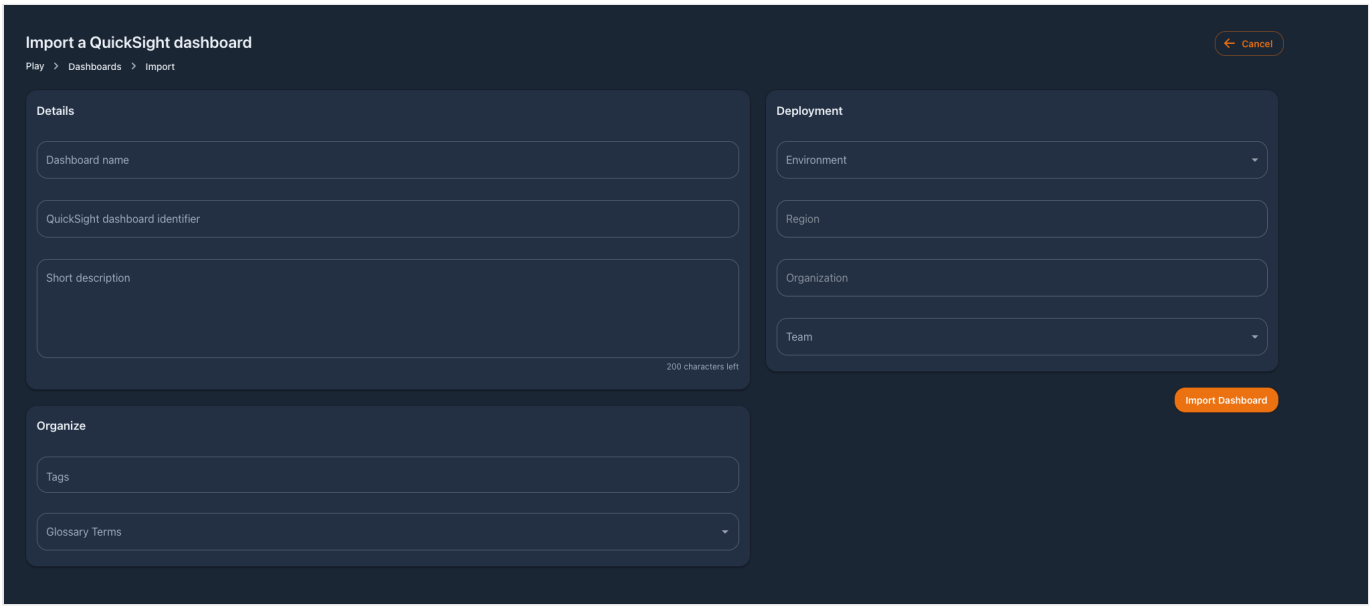


5.4.2 **Import a dashboard**

Our user has been working on a Dashboard in Quicksight and wants to register it and make it available in data.all. The first step is to copy the Dashboard ID to your clipboard. You can find this ID in the Quicksight URL.

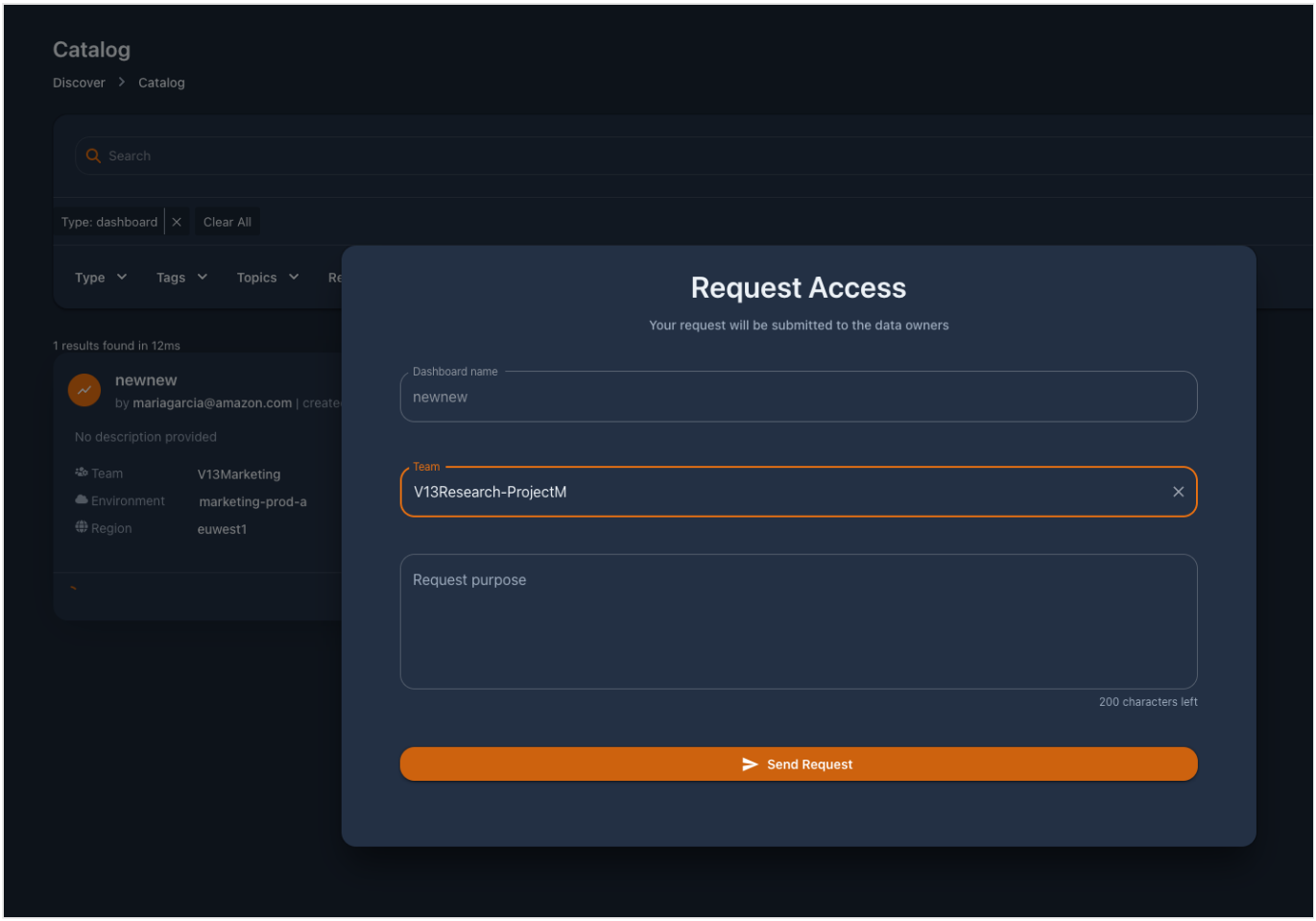


Now, go back to data.all and in the Dashboards menu, click on the Import button in the top-right corner. Fill in the following form and paste the dashboard ID correspondingly.



5.4.3 **Share a dashboard**

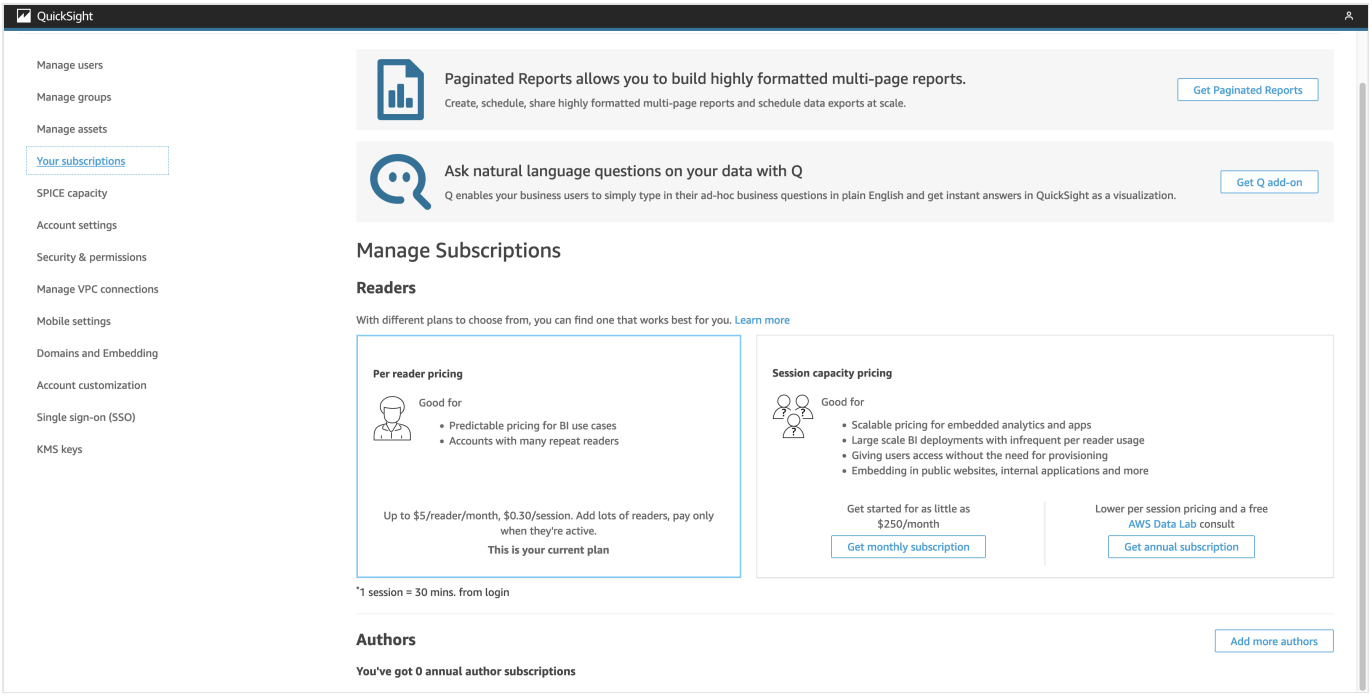
Once a dashboard is imported, it is catalogued in the central Catalog. Users can go to the Catalog, filter by dashboard and request access to a dashboard as shown in the picture below.



Once the request is open, the Dashboard owner team can accept or reject the request in the Dashboard Shares tab. If the request is approved, the requesters' team can visualize the Dashboard directly from data.all UI.

Session capacity pricing

To be able to embed the Dashboard using anonymous sessions, Session Capacity pricing needs to be enabled in the source Quicksight account. A Quicksight administrator in the AWS account needs to go to the **Manage Quicksight** menu and enable capacity pricing.


QuickSight
Manage users
Manage groups
Manage assets
Your subscriptions
SPICE capacity
Account settings
Security & permissions
Manage VPC connections
Mobile settings
Domains and Embedding
Account customization
Single sign-on (SSO)
KMS keys
Paginated Reports allows you to build highly formatted multi-page reports.
Create, schedule, share highly formatted multi-page reports and schedule data exports at scale.
Get Paginated Reports
Ask natural language questions on your data with Q
Q enables your business users to simply type in their ad-hoc business questions in plain English and get instant answers in QuickSight as a visualization.
Get Q add-on
Manage Subscriptions
Readers
With different plans to choose from, you can find one that works best for you. Learn more
Per reader pricing
Good for
Predictable pricing for BI use cases
Accounts with many repeat readers
Up to $5/reader/month, $0.30/session. Add lots of readers, pay only when they're active.
This is your current plan
Session capacity pricing
Good for
Scalable pricing for embedded analytics and apps
Large scale BI deployments with infrequent per reader usage
Giving users access without the need for provisioning
Embedding in public websites, internal applications and more
Get started for as little as $250/month
Get monthly subscription
Lower per session pricing and a free AWS Data Lab consult
Get annual subscription
*1 session = 30 mins. from login
Authors
Add more authors
You've got 0 annual author subscriptions

6. Security

Details on data.all security-first approach to building a modern data workspace

6.1 Data and metadata on data.all

6.1.1 Data virtualization

data.all is a fully virtualized solution that does not involve moving data from existing storage layers.

Any queries run on the data.all are pushed to existing processing layers (e.g. directly to your database, warehouse, or a processing layer such as Athena or Presto on top of S3).

6.1.2 Data and metadata storage

Data and metadata collected and created by data.all are stored in applications and databases within the customer's VPC (virtual private cloud). This includes information for data previews and queries, data quality, metadata, and user data.

6.1.3 Data previews and queries

data.all gives users the ability to see sample data previews for a datasets and results for any queries run on data.all.

In both cases, the request is pushed upstream to the original data source, and a 100-row sample of the result is provided to data.all users.

6.1.4 Data quality profile

Users can generate data quality metrics with the click of a button on data.all. Once generated, these metrics are stored in PostgreSQL on the customer's VPC.

6.1.5 Metadata

Dataset metadata, including metadata generated on data.all, is stored across Elasticsearch and Aurora PostgreSQL.

Elasticsearch is used to optimize search on the product, and Aurora PostgreSQL acts as a persistence backend.

6.1.6 User data

Data on users, roles, and IdP groups is stored in a PostgreSQL database.

Any user data transmitted over the internet is SSL-encrypted over HTTPS.

6.1.7 Authentication

The data.all authentication process is based SAML 2.0–based login. data.all can also integrate into organizations' existing SAML 2.0–based SSO authentication systems.

6.1.8 AWS Web Application Firewall (WAF)

Data.all supports the opportunity to add custom rules to AWS WAF. These rules are set in `cdk.json` at the root level of the repository. As a custom rules (property `custom_waf_rules`) customer can set two allow lists: * The Geo match allow-list (property `allowed_geo_list`) is an array of two-character country codes that you want to match against. If this property is specified, WAF will block web requests from all other countries, otherwise requests from all countries will be allowed. * The IP match allow-list

(property `allowed_ip_list`) is used to specify zero or more IP addresses or blocks of IP addresses. If this property is specified, WAF will block web requests from all other IP addresses, otherwise requests from all IP addresses will be allowed.

Example of custom WAF rules setting:

```
{
  "app": "python ./deploy/app.py",
  "context": {
    "@aws-cdk/aws-apigateway:usagePlanKeyOrderInsensitiveId": false,
    "@aws-cdk/aws-cloudfront:defaultSecurityPolicyTLSv1.2_2021": false,
    "@aws-cdk/aws-rds:lowercaseDbIdentifier": false,
    "@aws-cdk/core:stackRelativeExports": false,
    "tooling_region": "eu-west-1",
    "DeploymentEnvironments": [
        {
            "envname": "dev",
            "account": "000000000000",
            "region": "eu-west-1",
            "custom_waf_rules": {
                "allowed_geo_list": [
                  "US",
                  "CN"
                ],
                "allowed_ip_list": [
                  "192.0.2.44/32",
                  "192.0.2.0/24",
                  "192.0.0.0/16"
                ]
              }
        }
    ]
  }
}
```

7. Platform Monitoring

As an administrator of data.all I want to know the status of data.all. In this section we will focus on the following aspects of monitoring:

- Platform observability
- Platform usage

7.1 Observability

It refers to the infrastructure of data.all, the frontend and backend.

7.1.1 AWS CloudWatch

As part of the deployment, data.all deploys observability AWS resources with CDK and ultimately in CloudFormation. These include AWS CloudWatch Alarms on the infrastructure: on Aurora DB, on the OpenSearch cluster, on API errors... Operation teams can subscribe to a topic on Amazon SNS to receive near real time alarms notifications when issues are occurring on the infrastructure.

7.1.2 AWS CloudWatch RUM

Additionally, if we enabled CloudWatch RUM in the config.json file when we deployed data.all we will be able to collect and view client-side data about your web application performance from actual user sessions in near real time.

7.2 Platform usage

I want to know how my teams are using the platform. Inside this category we answer questions such as "how many environments or datasets are in data.all?".

7.2.1 RDS Queries

The first option is to query the RDS metadata database that contains all the information regarding environments, datasets and other data.all objects. You need access to the data.all infrastructure account, in which you will: 1) Navigate to RDS Console 2) Connect with secrets manager ARN 3) Get this ARN from AWS Systems Manager Parameter Store (search for "aurora") 4) Run SQL statements to extract insights about the usage of the platform

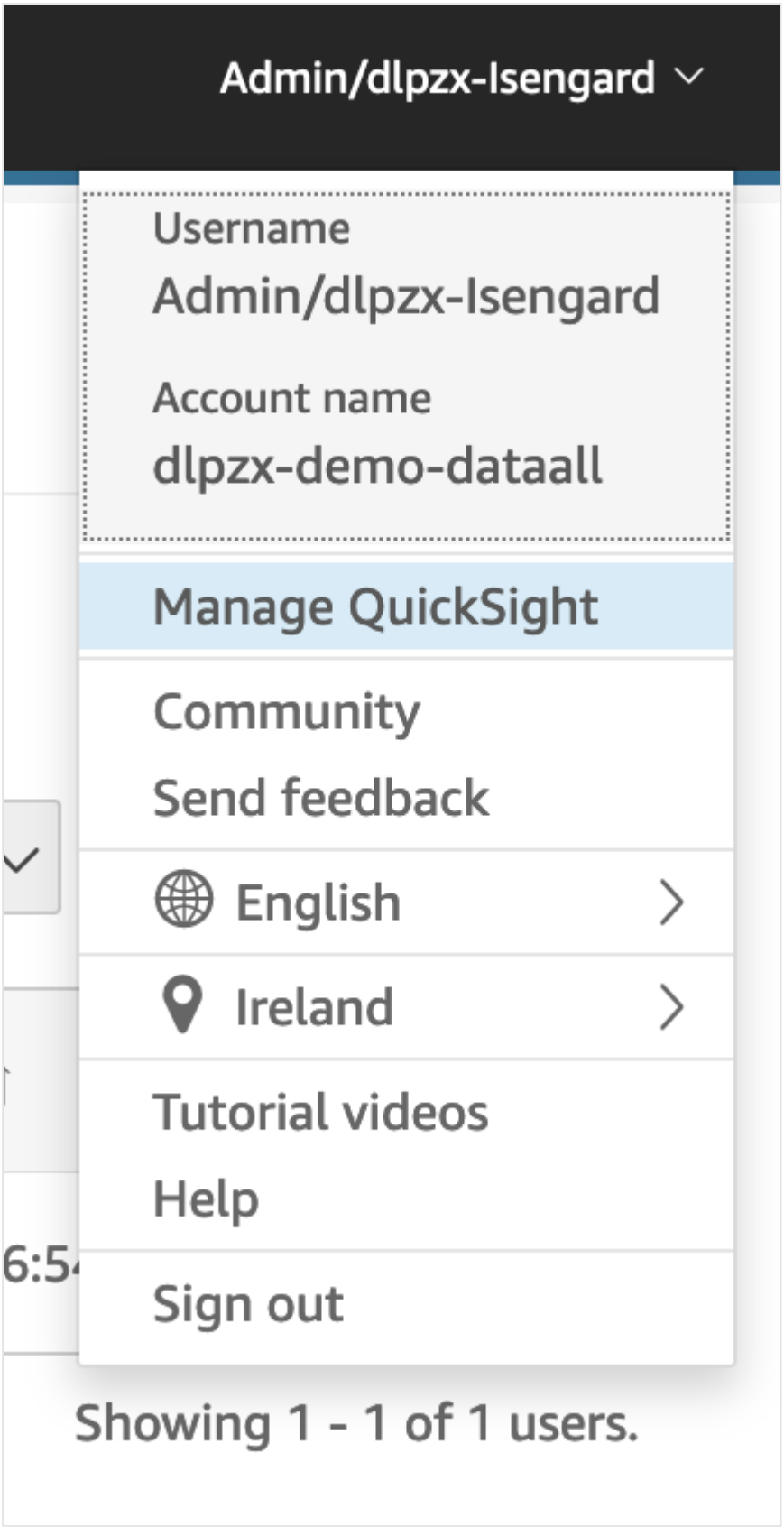
7.2.2 Quicksight enabled monitoring

When we deployed data.all, we can configure optional monitoring of Quicksight, this is the `enable_quicksight_monitoring` parameter. If enabled, we allow AWS Quicksight to establish a VPC connection with our RDS metadata database in that account. We modify the security group of our Aurora RDS database to communicate with Quicksight, then we can use AWS Quicksight to create rich dynamic analyses and dashboards based on the information on RDS. Once the deployment is complete you need to follow the next steps:

1) Pre-requisite: Quicksight Enterprise Edition We need to subscribe to Quicksight and allow data.all domain to embed dashboards, follow the instructions in the step 4 of the Linking environment section.

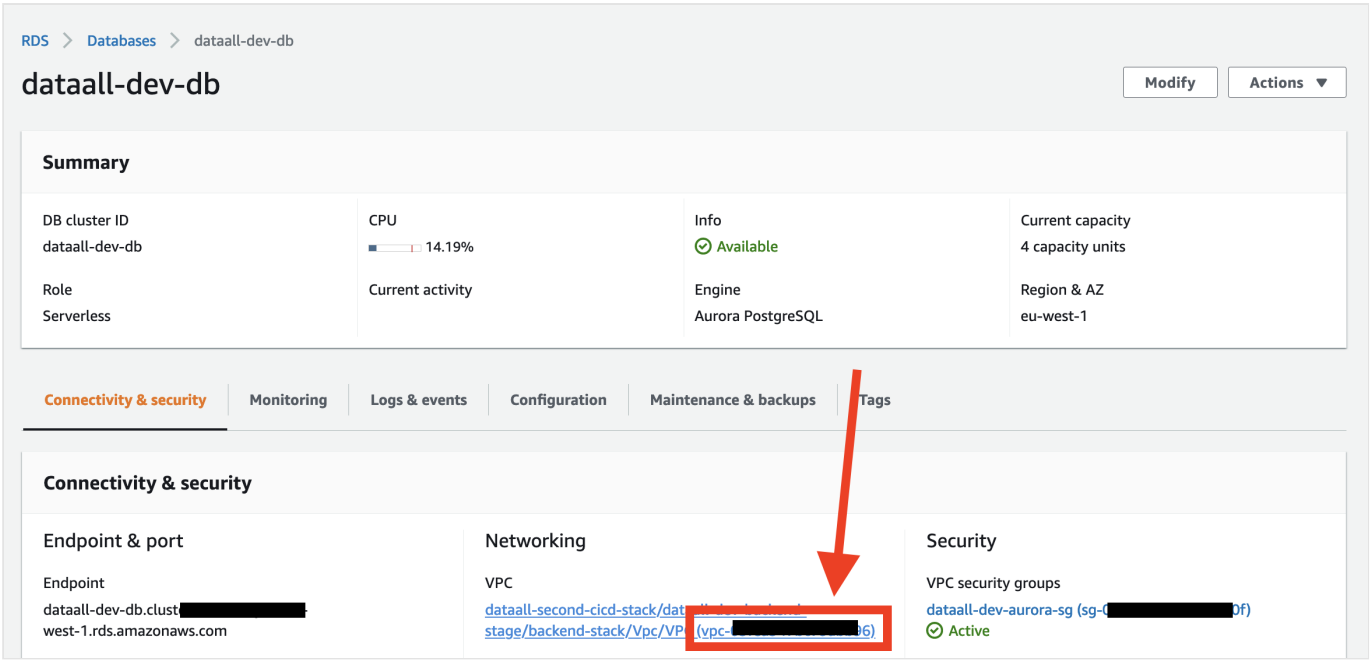
2) Create Quicksight VPC connection

Follow the steps in the documentation and make sure that you are in the same region as the infrastructure of data.all. For example, in this case Ireland region.



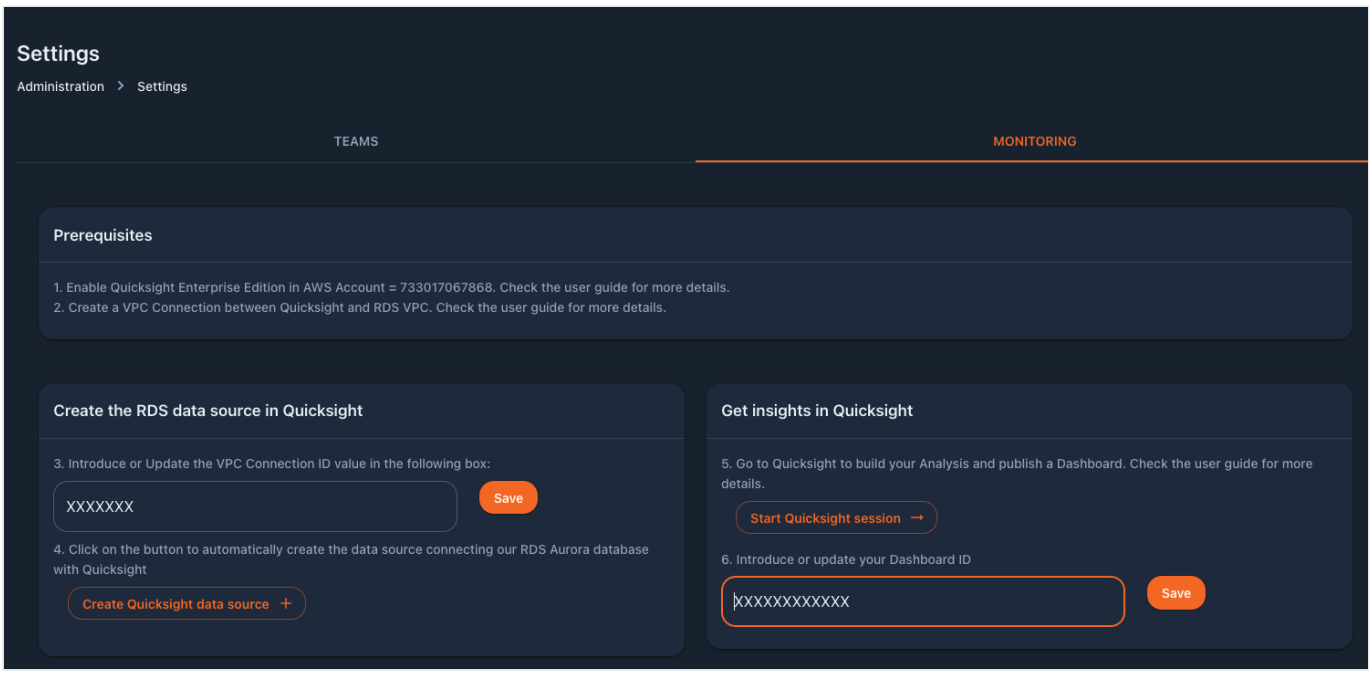
To complete the set-up you will need the following information:

- VPC_ID of the RDS Aurora database, which is the same as the data.all created one. If you have more than one VPC in the account, you can always check this value in AWS SSM Parameters or in the Aurora database as appears in the picture:

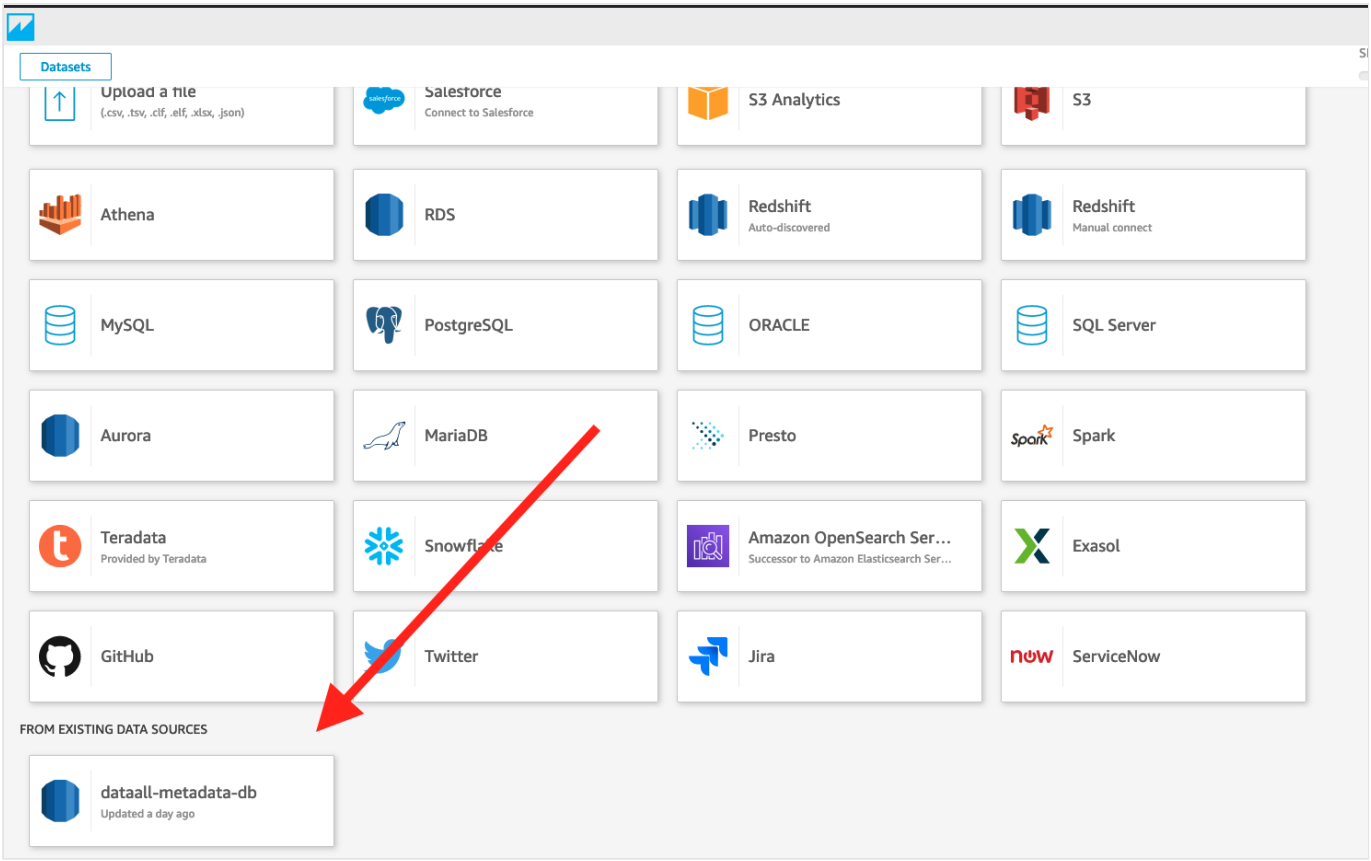


- Security group created for Quicksight: In the VPC console, under security groups, look for a group called `<resource-prefix>-<envname>-quicksight-monitoring-sg` For example using the default resource prefix, in an environment called prod, look for `dataall-prod-quicksight-monitoring-sg`.

3) Create Aurora data source We have automated this step for you! As a tenant user, a user that belongs to `DAADministrators` group, sign in to data.all. In the UI navigate to the **Admin Settings** window by clicking in the top-right corner. You will appear in a window with 2 tabs: Teams and Monitoring. In the Monitoring tab, introduce the VPC connection name that you created in step 2 and click on the Save button. Then, click on the Create Quicksight data source button. Right now, a connection between the RDS database and Quicksight has been established.



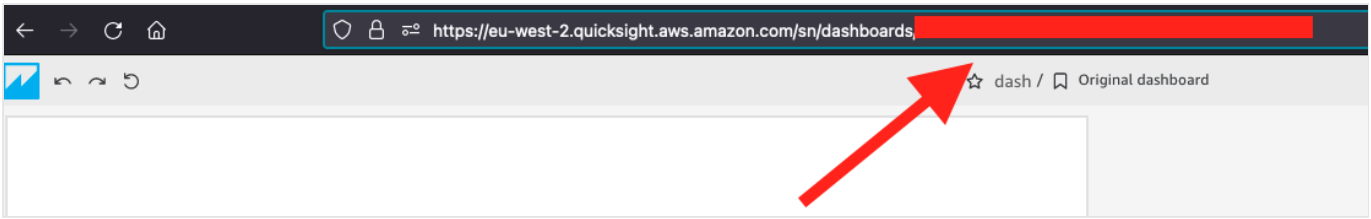
4) Customize your analyses and share your dashboards Go to Quicksight to start building your analysis by clicking on the Start Quicksight session button. First, you need to create a dataset. Use the **dataall-metadata-db** data source, this is our connection with RDS.



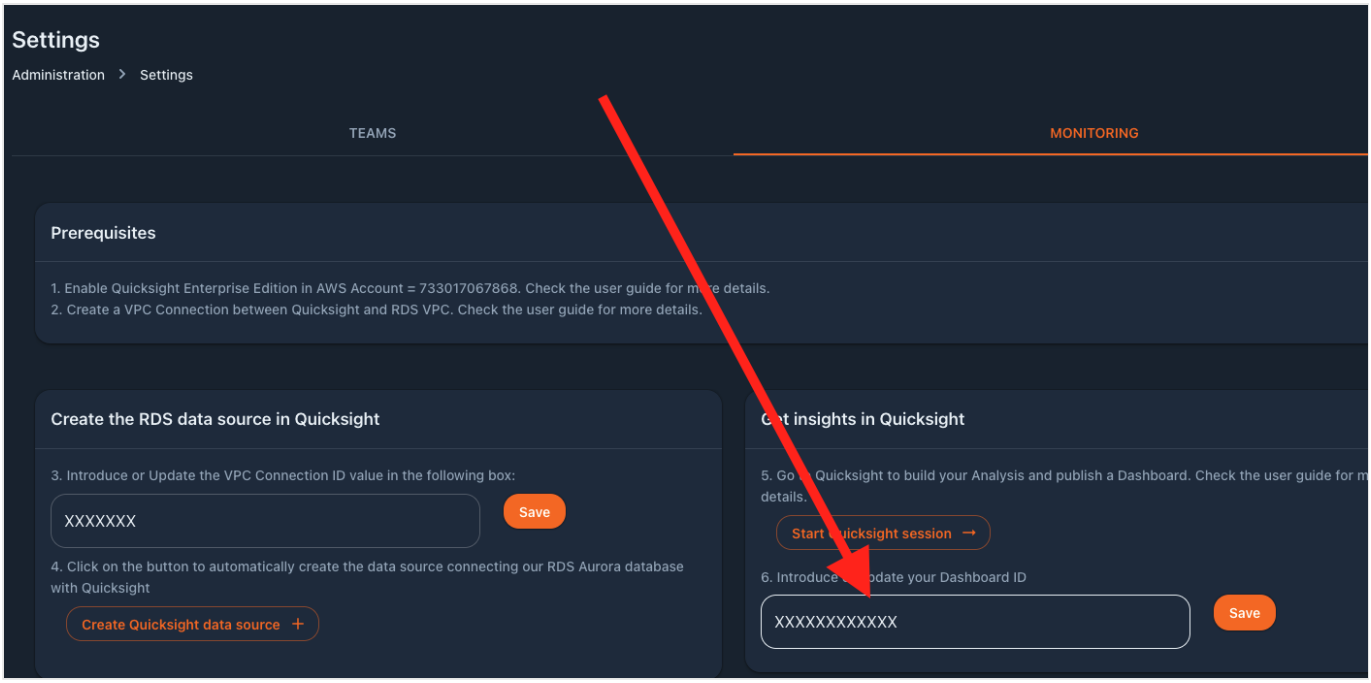
Use this dataset in an analysis (check the docs customization of analyses) and publish it as a dashboard (docs in publish dashboards)

> **Not only RDS**
>
> With Quicksight you can go one step further and communicate with other AWS services and data sources. Explore the documentation for cost analyses in AWS with Quicksight or AWS CloudWatch Logs collection and visualization with Quicksight.

5) Bring your dashboard back to data.all Once your dashboard is ready, copy its ID (you can find it in the URL as appears in the below picture)



Back in the data.all Monitoring tab, introduce this dashbaord ID. Now, other tenants can see your dashboard directly from data.all UI!


Settings
Administration > Settings
TEAMS
MONITORING
Prerequisites
1. Enable Quicksight Enterprise Edition in AWS Account = 733017067868. Check the user guide for more details.
2. Create a VPC Connection between Quicksight and RDS VPC. Check the user guide for more details.
Create the RDS data source in Quicksight
3. Introduce or Update the VPC Connection ID value in the following box:
XXXXXXX
Save
4. Click on the button to automatically create the data source connecting our RDS Aurora database with Quicksight
Create Quicksight data source +
t insights in Quicksight
5. Go to Quicksight to build your Analysis and publish a Dashboard. Check the user guide for more details.
Start Quicksight session
6. Introduce or update your Dashboard ID
XXXXXXXXXXX
Save

8. Labs

8.1 Hands-on Lab: Data Access Management with data.all teams

This document is a step-by-step guide illustrating some functionalities of the data.all "Teams" feature. This guide is far from exhaustive and mainly focuses on how users can share data across environment and teams. After completing it, you are free to continue exploring data.all and the functionalities it provides.

8.1.1 🎯 Scope of this guide

To follow this guide, you will need:

- An AWS account (#111111111111) where data.all is deployed. Your version of data.all must support the "Teams" feature
- An AWS account that will be used as a data.all environment for the data platform team (#222222222222)
- An AWS account that will be used as a data.all environment for the data science team (#333333333333)

The scenario you will implement in this guide is the following. The data platform team owns a dataset. Data scientists are interested by the content of this dataset for their analysis. There are however two different types of data scientists, that are members of two different teams: data science team A and data science team B.

A data scientist from team A will request access to the data platform dataset for its team. A data platform user will then accept the request, thus granting team A read-only access to the dataset. Team B does not have access to the dataset from the data platform team.

Then a user in team A creates a dataset in the data science environment. We will check that users in team B does not have access to this data.

You will go through the following steps to implement this scenario:

1. Create users and groups in Cognito
2. Create the Organisation and the Environment for the data platform team
3. Create the Dataset for the data platform team and upload some data
4. Create the Organisation and the Environment for the data science team
5. Invite team A and team B to the data science environment
6. Share data platform data with team A in the data science environment
7. Create a Dataset managed by team A in the data science account

Here is an illustration of the scenario:

1. Create users and groups in Cognito

First, you need to create users and groups from the Cognito console. This happens in the account where the infrastructure of data.all is deployed (#111111111111). You will later use these users to connect to data.all. Go to the AWS console and create five groups and four users as follow:

Cognito group

- **DAAdministrators**: group for data.all administrators
- **DataPlatformaAdmin**: group for data platform admin team
- **DataScienceAdmin**: group for data science admin team
- **TeamA**: first category of data scientists
- **TeamB**: second category of data scientists

Cognito users

- **data.alladmin**: create this user and add it to both in the DAAdministrators and DataPlatformAdmin groups. This user will be able to manage permissions of all teams in data.all (tanks to the DAAdministrators group membership) and will be able to create resources for the data platform team
- **ds-admin**: create this user and add it to the DataScienceAdmin group. This user will create resources for the data science team
- **ds-a**: create this user and add it to TeamA
- **ds-b**: create this user and add it to TeamB

> ⚠ **Warning**
>
> When creating users, you will need to **provide both the name of the user and its email.**

After creating users and assigning them to groups in Cogntio, you end-up with the following situation:

2. Create the Organisation and the Environment for the data platform team

We will start by creating the resources for the data platform team. Log into data.all with the user in the **DataPlatformAdmin** group. Create an Organisation for the Data Platform team. Make sure that the DataPlatformAdmin team manages this organization.

Now link a new environment to this organisation. You can do this by clicking on **Environment** and then **Link Environment**

When onboarding a new AWS account as an environment in data.all, you usually need to make some operations in the account first. The UI lists these operations for you: bootstrapping the AWS account and creating the data.allPivotRole notably. You will have to go through these operations if it is the first time you use this AWS account to create an environment in data.all. Then, create the environment by providing a name, the account ID (**#222222222222**) and the Team managing it (**DataPlatformAdmin**).

Wait until the stack is deployed successfully. You can check the status of the stack in the **stack** tab of the environment. Once the status is **create_complete**, create a new dataset in this environment. You can do it from the **Contribute** window

Deploy this dataset in the environment you have just created. Also make sure that the **DataPlatformAdmin** team owns this dataset (Governance section):

Wait until the dataset is created successfully. You can check the status in the **stack** tab of the dataset. Once the status is **create_complete**, you can start uploading some data from the **upload** tab:

From this window, you are able to upload files in your dataset. When uploading files, you can ask for a crawler running automatingly in your dataset, thus populating a glue database. To make sure the crawler will work, please upload a csv file of your choice. Insert any name you want in the **prefix** section. This will be the name of your Glue table.

Click on the **upload** button. This puts your file in the S3 bucket related to your data.all dataset. It also launches the Glue crawler populating the Glue database. Leave some time for the crawler to run and click on the **Tables** tab. If the crawler ran successfully, clicking on the **synchronize** button will display your table. At this point, feel free to explore your table from the data.all user interface.

We have completed all the tasks on the data platform side. This included the creation of the organisation, the environment, the dataset and the upload of a csv file. This is an illustration of where we are in the process:

Note: As you may have already noted down at the beginning of this guide, the data platform user is also part of the **DAAdministrators** group. Being part of this group enables this user to manage the permissions of all the other teams in data.all. To do so, click **Setting**. This provides the list of teams for which you can manage the permissions.

Click on the icon next to the team's name to manage its permissions

This opens a new window from where you can manage all permissions of the team.

3. Create the Organisation and the Environment for the data science team

You will now create data.all resources for the data science team. Log into data.all with the user in the **DataScienceAdmin** group. Create an Organisation for the data science team. Make sure that the **DataScienceAdmin** team manages this organization.

Now link a new environment to this organisation. Provide a name for this environment, the AWS account ID (**#333333333333**), and the team owning it (**DataScienceAdmin**).

You now have an organisation and an environment managed by the **DataScienceAdmin** team. The next step is to invite team A and team B to this data science environment. This will enable data scientists from team A and team B to access the environment.

4. Invite Team A and Team B to the data science environment

With the user in DataScienceAdmin team, select the data science environment and click on the **Teams** tab. You can invite other teams in your environment with the **invite** button.

This opens a new window asking you to indicate the name of the team you want to invite. You can also manage the permissions this team will have in your environment. Use this **invite** button to invite **TeamA** and **TeamB** in your data science environment.

Users from team A and team B now have access to your environment

5. Share data platform data with Team A in the data science environment

Log into data.all with user in **TeamA**. This user does not own any data, but wants to access data of the data platform team. Go on data.all **Data Catalog** tab. This shows all the datasets and tables you can request access to. There are different tools you can use in order to find the data you are looking for (you can find more information about these in the data.all documentation):

- Directly typing the name of the dataset or the table in the search bar
- Use tags or topics associated to the datasets
- Use data.all Glossary

In this case, data scientist in team A wants to access **mydpdata** uploaded by the data platform team. Use the search bar to find the data. When you see the table you want, click on **Request access**.

This button opens a new window where you can configure your request. When you share a dataset or a table in data.all, the share occurs at an environment and team level. You therefore need to indicate for which environment and for which team you make the request. In this case, the user in team A wants to access data in the data science environment. Fill the request accordingly.

When you click on **Send Request**, this does not directly send the request to the data latform team. It rather creates a Draft that you can still edit in the **Collaborate** tab, under **Sent**. Click on the **Submit** button to send the request

Now re-open a new data.all window connected as the user in the **DataPlatformAdmin** team. This team owns the dataset and is therefore responsible of accepting access requests. It is possible to delegate this right to other teams using **Data Stewards** but we did not set this up in this guide. Under **Collaborate** and **Received**, you can find all the access requests received by the data platform team. Locate the request you just made with the user in Team A. If you want to know more about this request (who is making it, for which table in the dataset,...), click on **Learn More**. If you agree to grant access, click on **Approve**.

This action triggers an ECS task that updates the permissions of the table in Lake Formation. Users in Team A are now able to access the data platform data. Let us verify it.

Re-open data.all connected as the user in **TeamA**. You can first visit the **Contribute** tab where you will see the dataset that has been shared with team A.

Quick reminder: The data platform team agreed to share the **mydpdata** table with TeamA in the data science environment called **DSENV**.

As a conclusion, the table **mydpdata** is accessible from the environment **DSENV**, through a role only team A can assume. Team A users can assume this role directly from the data.all user interface. Select the data science environment and go under the **Teams** tab. You will then see all the teams that have access to the environment. Find TeamA line and click on the AWS logo.

This opens a new window in the AWS console. The AWS account is the one you associated to the data science environment earlier in this guide (#333333333333). Also note that you are assuming a role specific to your team. Use the search bar to get to the Athena console. In the Athena Query editor, you will be able to see under **Database** the dataset shared by the data platform team. The name of this database is a concatenation of "dh" (for data.all), the name of the dataset (dpdataset) and random characters to ensure unicity. Under **Tables**, you can now see **mydpdata** which you can query using with Athena.

Explanation: When the data platform team uploaded the csv file under the **mydpdata** prefix, the crawler created a new Glue table called **mydpdata** in the AWS account associated to the data platform environment (#333333333333). When the data platform team accepted to share **mydpdata** with team A in the data science environment, it triggered an ECS task that updated the Lake Formation (AWS service managing data access) settings in both the data platform and data science environments. It updated the settings in a way that allows the IAM role of team A in the data science environment to read the **mydpdata** table stored in the data platform environment. In short, only the role of team A in the data science environment is able to read **mydpdata** table (in addition to data platform team of course). This is a **read-only** access, and the data is not moved from the data platform environment to the data science environment.

You can repeat the same thing to check that team B does not have access to the data. Log into data.all with a user in **TeamB** and select the data science environment. Under the **Teams** tab, click on the AWS logo to connect to the AWS console assuming the role of **TeamB**. Go to the Athena Query Editor and you will see that you won't be able to see data shared with team A.

At this stage of the guide, you should better understand how data sharing cross account works. The graph below illustrates where we are in the original scenario.

6. Create a Dataset managed by team A in the data science account

In the previous section of this guide, you went through an example of how you can share data across environment and teams. In this section, we will focus on the creation of datasets in a single account. Team A will create a dataset in the data science environment. We will make sure that other teams invited to the data science environment (t eamB) are not able to access the dataset of team A.

Open data.all with a user in **TeamA**. In the **Contribute** section, create a new dataset in the data science environment. Make sure that **TeamA** owns this dataset.

When the dataset is fully created, upload a csv file from the **upload** tab of the dataset. Upload this file under a prefix named **datateama** to create a new Glue table with the same name. After uploading the file, wait a few minutes to let the crawler do its job. In the **Tables** section, click on **Synchronize** to display your new table.

Now that the data is uploaded, team A is able to access the data as it is registered as the owner of the dataset. However, team B is not able to read the data even if it has access to the environment. If you log into data.all with the user in team B, you won't be able to see the **TeamADataset** in the **Contribute** section. In addition, you will find below two screenshot of the Athena console. In the first screenshot, we assume the role of **TeamA** in the data science environment (process already explained in the previous section). In the second screenshot, we assume the role of **TeamB** in the data science environment. When assuming the role of team A, we can see the team A dataset in the **database** section, and also the **datateama** table. We can then query the data with Athena. However, when assuming the role of team B in the data science environment, we are not able to see any dataset. This is because in this guide, we have not created or shared any dataset with team B. Team B is thus unable to query the data of team A.

This last section illustrated how you can use teams to manage data access in a single environment. You have reached the end of the guide that illustrated some capabilities that data.all brings. Now that you got the basis, fell free to explore all the other things you can do with your data.

Cleanup

When you are done with this guide, you delete your data.all resources (dataset, environment, organization). This also automatically deletes the Cloudformation stacks created in your AWS accounts.