aws

User Guide

The lakehouse architecture of Amazon SageMaker

The lakehouse architecture of Amazon SageMaker: User Guide

Table of Contents

What is the lakehouse architecture of Amazon SageMaker?

The lakehouse architecture of Amazon SageMaker unifies data across Amazon S3 data lakes and Amazon Redshift data warehouses so you can work with your data in one place. You can bring data from operational databases and business applications into your lakehouse in near real-time through zero-ETL integrations. Additionally, run federated queries on data stored across multiple external data sources to access and query your data in-place. The lakehouse architecture is compatible with the Apache Iceberg open standard, giving you the flexibility to use your preferred analytics engine. Secure your data in the lakehouse architecture by defining fine-grained permissions that are enforced across all analytics and machine learning (ML) tools and engines.

The lakehouse architecture works by creating a single catalog where you can discover and query all your data. When you run a query, AWS Lake Formation checks your permissions while the query engine processes data directly from its original storage location, whether that's Amazon S3 or Amazon Redshift.

The lakehouse architecture leverages Apache Iceberg table format for enhanced big data storage and analysis across multiple analytics engines. lakehouse architecture introduces Apache Iceberg REST API interface as part of the AWS Glue Data Catalog to support Iceberg compatible analytics query engines, both AWS and non-AWS. You can access both the Amazon S3 data lakes including Amazon S3 Tables and Amazon Redshift warehouse tables as Iceberg tables using the supported integrated engines, such as Amazon Athena and Spectrum.

What is a data lakehouse?

A data lakehouse is an architecture that unifies the scalability and cost-effectiveness of data lakes with the performance and reliability characteristics of data warehouses. This approach eliminates the traditional trade-offs between storing diverse data types and maintaining query performance for analytical workloads.

The lakehouse architecture provides the following key benefits:

- **Transactional consistency** – ACID compliance ensures reliable concurrent operations
- **Schema management** – Flexible schema evolution without breaking existing queries

- **Compute-storage separation** – Independent scaling of processing and storage resources
- **Open standards** – Compatibility with Apache Iceberg open standard
- **Single source of truth** – Eliminates data silos and redundant storage costs
- **Real-time and batch processing** – Supports both streaming and historical analytics
- **Direct file access** – Enables both SQL queries and programmatic data access
- **Unified governance** – Consistent security and compliance across all data types

Key components of the lakehouse architecture of Amazon SageMaker

The lakehouse architecture has the following key components, in addition to the components of AWS Glue Data Catalog and AWS Lake Formation.

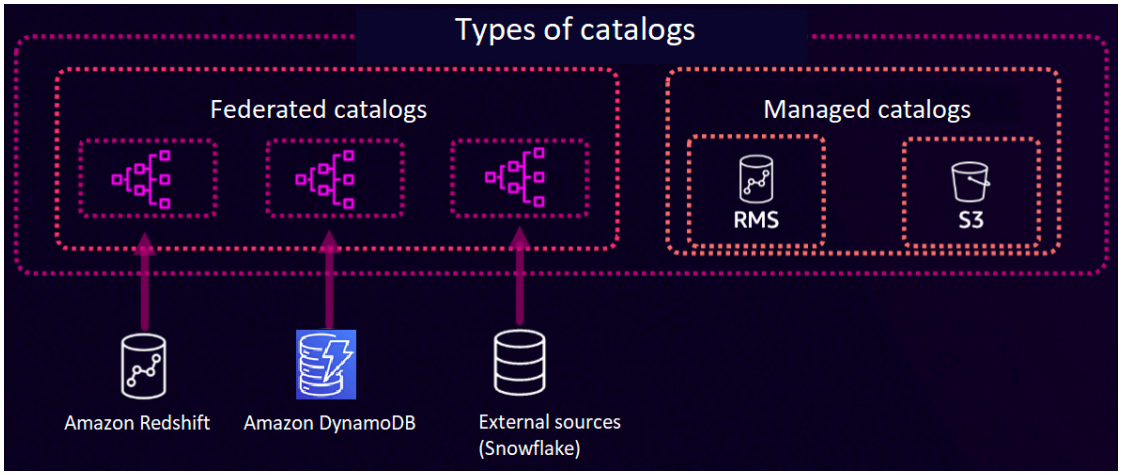
Storage

You can read and write data into Amazon S3 or Redshift Managed Storage (RMS) based on the storage type you choose to store data in the lakehouse.

Catalog

A catalog is a logical container that organizes objects from a data store, such as schemas, tables, views, or materialized views such as from Amazon Redshift. You can create nested catalogs to mirror the hierarchical structure of your data sources within the lakehouse architecture.

There are two types of catalogs in Lakehouse: federated catalogs and managed catalogs. A federated catalog mounts existing data sources you add to the lakehouse. A federated catalog can bring existing data in data sources such as Amazon Redshift, Amazon DynamoDB, and Snowflake. A managed catalog refers to a new catalog you create using Lakehouse. A managed catalog manages data using RMS or S3, as shown in the following diagram.



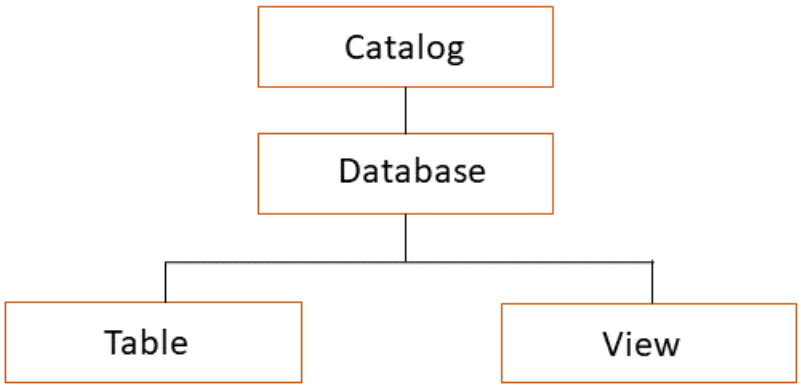
Database

Databases organize metadata tables in a catalog in the lakehouse architecture.

Table/View

Tables and views are database objects that define how to access and represent the underlying data. They specify details such as schema, partitions, storage location, storage format, and the SQL query required to access the data.

The following is a diagram of how catalogs, databases, tables/views work in Lakehouse.



How the lakehouse architecture of Amazon SageMaker works

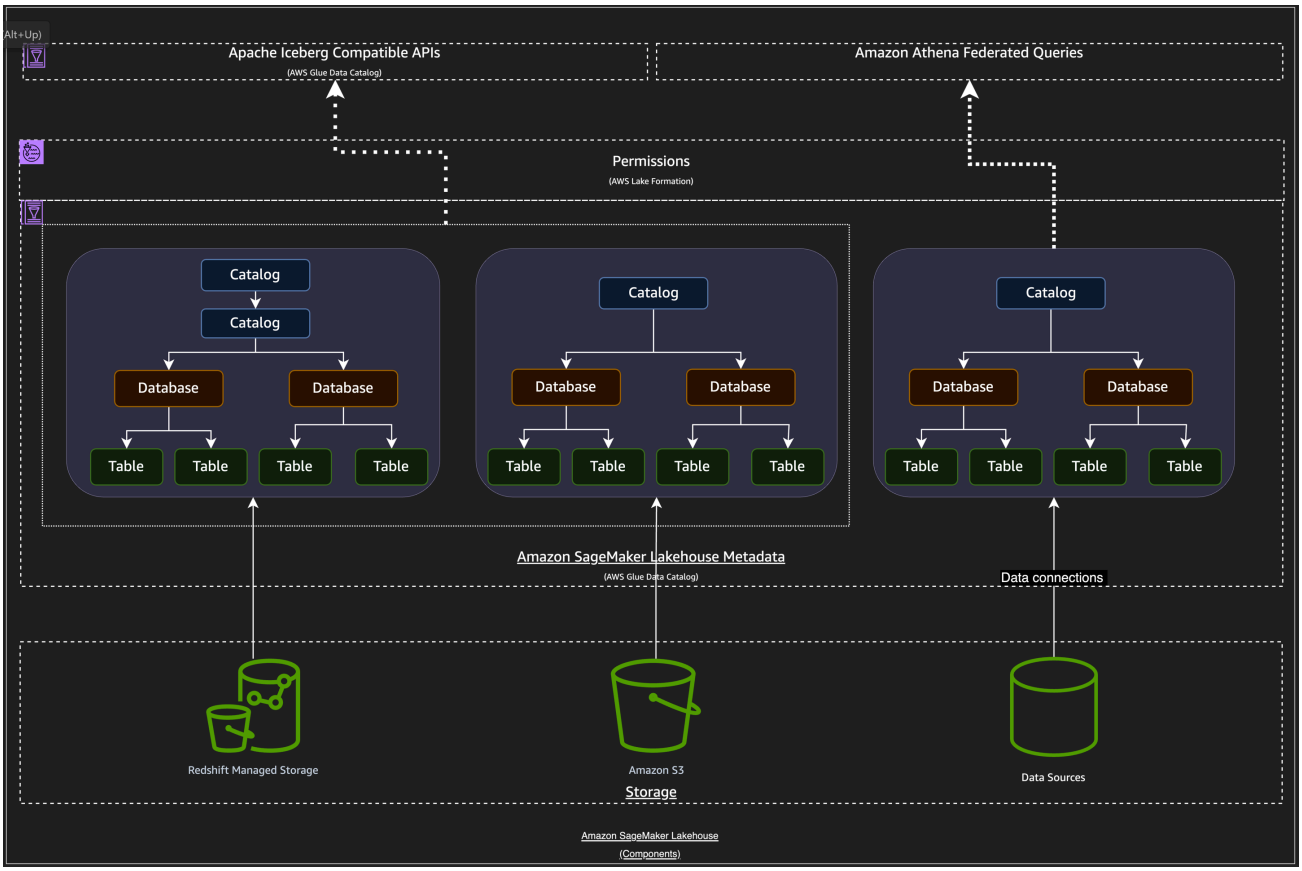
The lakehouse architecture organizes data from various sources into catalogs. Each catalog represents data from existing sources like Amazon Redshift data warehouses, Amazon S3 data lakes, databases, or business applications. You can also create new catalogs in the lakehouse to store data in S3 or Redshift Managed Storage (RMS). Additionally, these catalogs are mounted as

databases in Amazon Redshift, so you can connect and analyze your lakehouse data using SQL tools.

You can access the data as Apache Iceberg tables and query it using analytics engines that are integrated with the lakehouse architecture. These include Amazon Athena, Amazon Redshift Spectrum, Spark in Amazon EMR and AWS Glue 5.0 ETL, Amazon SageMaker Unified Studio, and other Iceberg compatible engines.

The lakehouse architecture is built on AWS Glue Data Catalog and AWS Lake Formation in your AWS account. With the lakehouse architecture, you can access and query your existing data in Amazon Redshift data warehouses and store new data in RMS from any Apache Iceberg compatible engine.

The following diagram shows how the lakehouse architecture works.



Data connections in the lakehouse architecture of Amazon SageMaker

The lakehouse architecture provides a unified approach to managing data connections across AWS services and enterprise applications. These connections provide a consistent experience for creating, testing, and exploring data sources, regardless of the underlying data platform.

Capabilities

With the lakehouse architecture connections, you can do the following:

- Create connections to a variety of data sources, including business applications and external data sources
- Manage data connections in a single place
- Test the connectivity of your data sources to ensure they are working as expected
- Browse the metadata and preview the data from your connected sources
- Reuse the same connection across different AWS services like AWS Glue, Amazon Athena and Amazon SageMaker AI
- Manage credentials using AWS Secrets Manager
- Authenticate using basic authentication methods such as OAuth2 and IAM

Supported data sources

The lakehouse architecture connections support several popular data sources, including the following:

Supported Data Sources

Data Source	Type
Google BigQuery	Database
Amazon DocumentDB	Database
Amazon DynamoDB	Database
Amazon Redshift	Database

Data Source	Type
MySQL	Database
PostgreSQL	Database
SQL Server	Database
Snowflake	Database
Oracle	Database
Amazon Aurora MySQL	Database
Amazon Aurora PostgreSQL	Database
Microsoft Azure SQL	Database

Note

The lakehouse architecture currently supports lowercase table, column, and database names. For optimal experience in the lakehouse architecture, ensure that all database identifiers are in lowercase.

Use the lakehouse architecture connections

After you've added data sources to the lakehouse architecture, you can use it in various AWS services:

- Amazon SageMaker Unified Studio – Browse metadata, preview sample data, and run SQL queries against the connected data.
- AWS Glue – Use the connection for ETL jobs and crawlers.
- Amazon Athena – Query data directly using Athena's federated query capabilities. For more information, see Register federated catalogs in Amazon Athena.
- Amazon SageMaker AI – Access data for building machine learning models.

Understanding resources for the lakehouse architecture of Amazon SageMaker

When you create a connection in Amazon SageMaker Unified Studio, several resources are created in your AWS account(s) behind the scenes. These resources can include:

- AWS Glue connection - A connection object is created in the AWS Glue crawler. This stores the core connection information and is used by various AWS services.
- Athena data catalog - For connections that will be used with Athena , an Athena data catalog is created. This allows Athena to query the external data source.
- AWS Glue data catalog entries - Databases, tables, and schemas from your external data source are registered in the Data Catalog. This enables AWS services to understand the structure of your external data.
- Lambda (for Athena Federated Query) - For some data sources, a Lambda function is created to facilitate federated queries. This function acts as a bridge between Athena and the external data source.

To view these resources, access the respective AWS service consoles (AWS Glue, Athena, IAM, etc.) in the AWS account associated with your Amazon SageMaker Unified Studio project.

In these consoles, look for resources with names that include your Amazon SageMaker Unified Studio project ID or connection name.

For more information about how to create a data connection and explore a connected data source, see Adding data sources in lakehouse architecture.

Support for the Apache Iceberg open standard in the lakehouse architecture of Amazon SageMaker

The lakehouse architecture provides support for Apache Iceberg, enabling organizations to unify data across Amazon S3 data lakes and Amazon Redshift data warehouses while building powerful analytics and AI/ML applications on a unified data layer.

With the lakehouse architecture, you gain the flexibility to access and query your data in-place using all Apache Iceberg compatible tools and engines, including open-source Apache Spark. This integration uses the AWS Glue Iceberg REST Catalog, which provides a standardized REST API

interface for managing Iceberg table metadata and enables seamless connectivity with third-party engines. For more information, see how to use AWS Glue Iceberg Rest Catalog for accessing Iceberg tables in Amazon S3.

Through fine-grained permissions enforced across all analytics and ML tools, the lakehouse architecture ensures secure data access while supporting advanced Iceberg features like ACID transactions, schema evolution, time travel queries, and efficient row-level operations—all essential capabilities for modern data-driven organizations seeking to process and analyze vast amounts of information efficiently.

The lakehouse architecture also supports multiple table optimization options with Glue Catalog to enhance the management and performance of Apache Iceberg tables that the AWS analytical engines and ETL jobs uses. These optimizers provide efficient storage utilization, improved query performance, and effective data management. For more information, see Optimizing Iceberg tables.

With the lakehouse architecture, you can calculate and update number of distinct values (NDVs) for each column in Iceberg tables with the AWS Glue Data Catalog. These statistics can facilitate better query optimization, data management, and performance efficiency for data engineers and scientists working with large-scale datasets. For more information, see Optimizing query performance for Iceberg tables.

Getting started with the lakehouse architecture of Amazon SageMaker

This guide helps you accomplish common tasks like finding relevant datasets, running SQL queries against your data warehouse and data lake simultaneously, collaborating with team members through data publishing, and maintaining data governance standards. Your administrator will provide the necessary access permissions and project roles to get started.

Topics

Prerequisites

- Your administrator must grant you access to the lakehouse architecture.

  If you don't have access to it, contact your administrator. For more information, see https://docs.aws.amazon.com/sagemaker-unified-studio/latest/userguide/getting-started-access-the-portal.html.

- You must have a Amazon SageMaker Unified Studio project and with the proper project membership role.

  If you don't have proper access to a project, contact your administrator. To view your project membership role, choose **Actions** on the top right corner of the project overview page, then choose **Manage members**. You will see your membership role in the **Role** column.

Create a project

You can create a project from a project profile, which defines a template for projects in your domain. To use lakehouse architecture, your project must be created using either Data analytics and AI-ML model development or SQL analytics project profile. For more information about creating a project, see Create a project from lakehouse architecture User Guide.

When using lakehouse architecture, you can create the following resources in the lakehouse:

1. **Databases in AWS Glue Data Catalog**

   lakehouse architecture is implemented on AWS Glue and AWS Lake Formation in your AWS account.

2. **A catalog to store data in Redshift Managed Storage (RMS) format**

   You will create a catalog in RMS format. To view the catalog, navigate to the AWS Lake Formation console at https://console.aws.amazon.com/lakeformation/, you should be able to see the catalog from the **Catalogs** list.

3. **Provisioning permissions**

   You will create an IAM role when you create a project. Each project has a dedicated IAM role. This IAM role has permission to the resources that are created from this project. The Amazon Resource Name (ARN) of this IAM role is visible from **Project details** section of the **Project overview** page.

Browse data

You can browse data in lakehouse architecture by completing the following steps.

To browse data

1. Choose a project to view the data.
2. On project page, from the left navigation, choose **Data**. This opens the **Data** explorer in the middle of the page.

   The **Data** explorer includes: **Lakehouse**, **Redshift**, and **S3**.

3. Expand **Lakehouse** to view catalogs, databases, tables.

Upload data

You can upload data in CSV or JSON format to a catalog. To upload data, follow the instructions in ???.

After uploading data is complete, you will see the table listed within the database under **AwsDataCatalog**.

Query data

You can query data using supported query editor.

To query data

1. On **Lakehouse**, choose **AwsDataCatalog** on top. Expand the catalog to view the list of databases. Choose a database.
2. From a selected database, choose a table. Then choose the three dot menu to the right of the table to view supported tools for data query.
3. Choose **Query with Athena**. This opens the **Data explorer** page where you can run SQL queries. You might find information in SQL reference for Athena helpful.
4. Choose **Query with Amazon Redshift**. This opens the **Data explorer** page where you can run SQL queries. You might find information in Querying a database using the query editor v2 helpful.

To subscribe an asset, see Request subscription to assets in Amazon SageMaker Unified Studio.

To publish data to the catalog from the lakehouse inventory, see ???.

Adding data sources in lakehouse architecture

lakehouse architecture supports several data sources. If you are new to data connections in lakehouse architecture, see ???.

In this topic:

- Uploading data
- Creating a catalog
- Adding existing databases and catalogs using AWS Lake Formation permissions
- Creating an AWS Glue database via Amazon SageMaker Unified Studio
- Deleting an AWS Glue database via Amazon SageMaker Unified Studio
- Amazon S3 tables integration

Creating connections in lakehouse architecture

Amazon SageMaker Unified Studio provides an interface for managing and utilizing data connections across various AWS services and external data sources. With Amazon SageMaker Unified Studio, you create, configure, and manage connections to databases, data warehouses, and applications all from a single platform. Amazon SageMaker Unified Studio allows you to explore your connected data sources, preview sample data, and seamlessly use these connections in SQL queries and Spark notebooks without having to switch between different interfaces or manage complex connection details manually.

Access the data explorer in a project

1. Open your web browser and navigate to Amazon SageMaker Unified Studio.
2. Enter your corporate credentials (usually integrated with Amazon IAM Identity Center).
3. After successful authentication, you'll be directed to the Amazon SageMaker Unified Studio home page. On the home page, you'll see a list of projects you have access to. Select the project you want to work with by clicking on its name.
4. From the dropdown menu, select the **Data** or **Data Management** option. This will open the Data section of the project overview page. In this data explorer, you can see a tree-like structure representing your data sources.

Create a new connection to add data sources

To add a new data source

1. In the data explorer, select the **+** button. Click this button to start adding a new data source.
2. In the modal, select **Add connection**. You'll be presented with a gallery of connector options. Select the connector you need. For supported data sources, see .

> **Note**
>
> lakehouse architecture currently supports lowercase table, column, and database names. For optimal experience in lakehouse architecture, ensure that all database identifiers are in lowercase.

3. You must configure your connector details. For example, if you choose to use a DynamoDB connection (preview), fill in the required fields, which can include:

   - Name: A unique identifier for this connection in Amazon SageMaker Unified Studio.
   - Description (optional): A description of the connection.

   > **Note**
   >
   > Each supported data source can have different parameters for the connection. Contact your administrator if you need them.

To see your DynamoDB tables displayed in lakehouse architecture after you add the connection, your administrator must grant you access through resource policies in the Amazon DynamoDB console.

To grant access to a DynamoDB table, your administrator can complete the following steps.

1. Sign in to the AWS Management Console and open the Amazon DynamoDB console at https://console.aws.amazon.com/dynamodb/.
2. On the left navigation of the DynamoDB console, choose **Tables**.
3. From the **Tables** page, choose the table to add access to.
4. On the details page of the selected table, choose **Permission**.
5. On the **Resource-based policy for table** section, update the policy with the project role ARN in `Condition`.

> **Note**
>
> You can find the project ARN on the Page details page in the lakehouse architecture.

The following is an example policy. It allows access of the IAM role named `datazone_user_role_projectid` to perform the allowed actions (`Query`, `Scan`, `DescribeTable`, `PartiQLSelect`) on the specified DynamoDB table. Administrators should choose to allow or deny the set of actions.

```
{
    "Sid": "Statement1",
    "Effect": "Allow",
    "Principal": "*",
    "Action": [
        "dynamodb:Query",
        "dynamodb:Scan",
        "dynamodb:DescribeTable",
        "dynamodb:PartiQLSelect"
     ],
    "Resource": "arn:aws:dynamodb:region:account:table/table_name",
    "Condition": {
        "ArnEquals": {
        "aws:PrincipalArn": "arn:aws:iam::region:role/datazone_user_role_projectid"
        }
    }
}
```

Explore a connected data source

After you have connected your data source, you can explore the data source in the data explorer.

1. After your connection is created, return to the data explorer.
2. You should now see your new connection listed in **Lakehouse**.
3. Expand the new connection to view available databases.
4. Expand a database to explore its schema.
5. You can select a table name to view more details about that table, such as Schema details and a list of tables. You can then examine the tables themselves by selecting a table.

6. You will be able to see tabs for **Columns** and **Sample data**. In the **Columns** view, you can view a list of columns in the table, as well as the data types for each column. In the **Sample data** view, you can see the rows of data from the table and use built-in sorting and filtering options to explore the data.

Authentication and tagging for creating connections

You administrator must create credentials and configure the secret tags for you before you create a connection.

Credentials

When creating a connection, if you choose a data source that requires the credentials for **Authentication**, contact your administrator because they must create and provide these credentials. There are two types of the credentials:

- User name and password
- AWS Secrets Manager

Secret tags

- To ensure the secret can only be used for a particular project, your administrator must tag with the `AmazonDataZoneProject` tag key and the value will be `projectId`.
- To use the secret across multiple projects, your administrator must tag the secret with `for-use-with-all-datazone-projects = true`.

Uploading data

You can upload data to the lakehouse architecture.

To upload data

1. On the **Data** section in the middle of the project page, choose **+** on the top. This opens **Add data source** on the right.
2. On **Add data source**, choose **Upload data**.
3. Choose **Click to upload** or drag and drop a CSV or JSON file. Complete the information in the form.

4. Choose **Upload data**.

Creating a catalog

You can create a catalog for your Redshift Managed Storage (RMS) objects.

To create a catalog

1. On the **Data** explorer in the middle of the project page, choose **+** on the top. This opens **Add data source** on the right.
2. On **Add data source**, choose **Create catalog**. Enter a name for your catalog.
3. Choose **Create**.

Adding existing databases and catalogs using AWS Lake Formation permissions

You can add existing databases and catalogs to the lakehouse architecture.

To add existing databases and catalogs using AWS Lake Formation permissions

1. Sign in to the lakehouse architecture by using the link your administrator gave you. If you don't have access to it, contact your administrator.
2. Choose a project to open the project page.
3. On the left navigation, choose **Project overview**. On **Project details**, copy the project role ARN.
4. Open the AWS Lake Formation console at https://console.aws.amazon.com/lakeformation/.
5. On the left navigation, from **Data catalog**, choose **Catalogs**.
6. On the **Catalogs** list view, choose a catalog you want to add to lakehouse architecture. From **Actions** on the right, choose **Grant**.
7. On the **Grant data lake permissions** page, choose **IAM users and roles** from **Principals**. Paste the IAM role you copied in the step 3.
8. On **Catalog permissions**, choose **Super user**. Choose **Grant**.

After you complete all the steps successfully, go back to the project page in the lakehouse architecture. You should see the Lake Formation catalog added to your lakehouse.

Creating an AWS Glue database via Amazon SageMaker Unified Studio

You can use this procedure to create a new AWS Glue database in your catalog in lakehouse.

> **Note**
>
> In order to complete this task, make sure that the project where you're creating a new database must be created with a project profile that contains the LakeHouseDatabase blueprint configured with the **On-Demand** mode. For more information, see Supported blueprints.

1. On the **Data** explorer in the middle of the project page, choose the ellipsis icon to the right of your catalog, and then choose **Create database**.
2. In the **Create database** pop up window, specify the name for the new database and then choose **Create database**.

Deleting an AWS Glue database via Amazon SageMaker Unified Studio

You can use this procedure to delete an AWS Glue database in your catalog in lakehouse.

> **Note**
>
> In order to complete this task, make sure that the project where you're creating a new database must be created with a project profile that contains the LakeHouseDatabase blueprint configured with the **On-Demand** mode. For more information, see Supported blueprints.

1. On the **Data** explorer in the middle of the project page, choose the ellipsis icon to the right of the database within your catalog that you want to delete, and then choose **Remove**.
2. In the **Drop database** pop up window, confirm the deletion and then choose **Drop database**.

Amazon S3 tables integration

The lakehouse architecture unifies all your data across Amazon S3 data lakes, Amazon Redshift data warehouses, and third-party data sources without having to copy data. Amazon S3

Tables delivers the first cloud object store with built-in Apache Iceberg support. The lakehouse architecture integrates with Amazon S3 Tables so you can access S3 Tables from AWS analytics services, such as Amazon Redshift, Athena, Amazon EMR, AWS Glue, or Apache Iceberg-compatible engines (Apache Spark or PyIceberg).

The lakehouse architecture integration with Amazon S3 Tables helps you secure analytic workflows by joining data from Amazon S3 Tables with sources, such as Amazon Redshift data warehouses, third-party, and federated data sources (Amazon DynamoDB or PostgreSQL). The lakehouse architecture also enables centralized management of fine-grained data access permissions for S3 Tables and other data, and consistently applies them across all engines. To get started, complete the steps in the following sections.

Prerequisites - complete all the steps in the Getting started with the lakehouse architecture of Amazon SageMaker.

Enable Amazon S3 integration

1. Navigate to the Amazon S3 console. In the left navigation pane, choose **Table buckets**.
2. Choose **Create table bucket**.
3. On the **Create table bucket** page, enter a **Table bucket name** and select **Enable integration**.
4. Choose **Create table bucket**.
5. You will see confirmation when Amazon S3 completes integration of your table buckets with the lakehouse architecture.

Onboard S3 Tables in the lakehouse architecture

To provide access to S3 tables, complete the following steps:

1. Navigate to the AWS Lake Formation console.
2. In the left navigation pane, choose **Catalogs** and choose **S3tablescatalog**.
3. From **S3tablescatalog**, under **Objects**, choose the name of your newly created **table bucket**.
4. From the **Actions** menu, select **Grant**.
5. In the **Grant permissions**, under IAM users and roles, select your Amazon SageMaker Unified Studio Project role. To grant full access, under **Catalog Permissions > Grant**, select **Super user**.

Create S3 Table and add data in the lakehouse architecture

1. Navigate to Amazon SageMaker Unified Studio, and select the project.
2. From the **Build** menu, select **Query Editor**, and ensure you have **Athena** selected in **Connections**.
3. Create a database using SQL.

```
CREATE DATABASE "s3tablescatalog/<Your Bucket Name>".<YourDBName>;
```

4. Create an S3 table using SQL.

```
CREATE TABLE "s3tablescatalog/<Your Bucket Name>".<YourDBName>.<YourTableName>
( c_salutation string,
  c_login string,
  c_first_name string,
  c_last_name string,
  c_email_address string)
  TBLPROPERTIES (
  'table_type'='ICEBERG'  );
```

5. Add data using SQL.

```
INSERT INTO "s3tablescatalog/<Your Bucket Name>".<YourDBName>.<YourTableName>
 VALUES('Dr.','1381546','Joyce','Deaton','Joyce.Deaton@qhtrwert.edu');
```

You can now use the following integrated analytics services:

- Amazon Athena - create databases, tables, query and add data in S3 Tables.
- Amazon Redshift - query data from S3 Tables.
- Amazon EMR - create table, namespace, query and add data in S3 Tables.
- AWS Glue - create table, namespace, query and add data in S3 Tables.
- AWS Lake Formation - grant fine-grained permissions for S3 table catalogs, databases, tables, columns, and cells.

> **Note**
>
> Access to S3 Tables with the lakehouse architecture is available in the AWS Regions where S3 Tables are available.

Publishing data in lakehouse architecture

After you have added data in the lakehouse architecture, you can publish the data to share it with other users in the lakehouse architecture. Data that is published is viewable as an asset in the project catalog and the Amazon SageMaker Catalog, and other users can create subscription requests in the Amazon SageMaker Catalog to include that data in their projects.

To publish data in the lakehouse architecture, complete the following steps:

1. Navigate to lakehouse architecture using the URL from your admin and log in using your SSO or AWS credentials.
2. Navigate to the project that contains the data that you want to publish in the lakehouse architecture. To do this, use the center menu at the top of the landing page and choose **Browse all projects**, then choose the name of the project that you want to navigate to.
3. In the center menu, choose **Data**. This takes you to the Data page.
4. Do either of the following:

   - If you want to publish a regular AWS Glue table, expand the catalog in the data navigation to view the list of databases in lakehouse architecture, then choose a database that contains the asset that you want to publish. Choose this table from the selected database and then proceed to the rest of the steps in this procedure to publish this table to the catalog.
   - If you want to publish an Amazon S3 table to the catalog, you must first complete the following steps to create a data source for the S3 Tables catalog and schedule its run job. Then you can proceed to the rest of the steps in this procedure to publish the S3 table to the catalog.
     - Navigate to **Data sources** and then choose **Create data source**.
     - On the **Step 1: Define source** page, specify the name for this data source, then under **Data source type** - choose **AWS Glue (Lakehouse)**, under **Data Selection** - choose **Enter the catalog name** and then speciy the name of your S3 tables catalog (s3tablescatalog/<catalog name>, then choose your database from that catalog (use the drop down menu), and then choose **Next**.

- On the **Step 2: Add details** page, leave all the default settings and choose **Next**.
- On the **Step 3: Set up schedule** page, choose a run preference and then choose **Next**.
- On the **Step 4: review** page, review your selections and then choose **Create**.

  Once the data source for the S3 tables catalog is created and run, you can proceed with the rest of the steps below to locate your S3 table and publish it to the catalog.

5. Expand the **Actions** menu, then choose **Publish to catalog**.
6. Confirm the action in the pop-up window by choosing **Publish to catalog**.

   The lakehouse architecture then fetches metadata for the asset. After a few minutes, the metadata is fetched and a success message appears.
7. (Optional) Choose **View details** to view the asset in the project catalog.

When it is successfully published you can view it in the **Assets** section of the project catalog and users in other projects can subscribe to it from the Amazon SageMaker Catalog.

You can use the project catalog to re-publish the data if you make changes, or to unpublish the data from Amazon SageMaker Catalog. For more information, see Data inventory and publishing.

Onboarding data into the lakehouse architecture of Amazon SageMaker

The lakehouse architecture of Amazon SageMaker is a unified data architecture that combines data lakes and data warehouses. It's compatible with Apache Iceberg and specifically optimized for building machine learning and generative AI applications on a single copy of data.

Data onboarding into the lakehouse architecture involves the provision of direct integration pathways for bringing existing Amazon Simple Storage Service (Amazon S3) data lakes, including Amazon S3 Tables, and Amazon Redshift data warehouses without complex migrations. The lakehouse architecture enables unified data access by connecting external sources such as Google BigQuery and Snowflake, allowing organizations to query and analyze both historical warehouse data and data lakes through a single interface.

The lakehouse architecture uses the AWS Glue Data Catalog (Data Catalog) for metadata management, providing a centralized repository that enables data discovery across your organization. For more information, see Data discovery and cataloging in AWS Glue.

After cataloging the data, you can use AWS Lake Formation to centrally manage data access permissions. Data lake administrators can grant fine-grained access permissions to other IAM principals (users or roles) within the same account or across accounts using tag-based access control (LF-Tags) and named resources methods. By using LF-Tags, data administrators can logically organize resources based on attributes such as domain and sensitivity level, ensuring consistent access controls across analytics and machine learning services including Amazon Athena, Amazon EMR, AWS Glue or Amazon Redshift Spectrum.

The lakehouse architecture integrates data from the following data sources:

1. **Amazon S3 table buckets** – Created directly within Amazon SageMaker Unified Studio with built-in Apache Iceberg support. For more information, see the section called “Working with Amazon S3 Tables”.
2. **Amazon Redshift Managed Storage** – Accessed through federation for unified querying. For more information, see the section called “Amazon Redshift Managed Storage”.
3. **Amazon S3 data lakes** – Direct integration of existing S3-based data assets. For more information, see the section called “Amazon S3 data lakes”.
4. **Federated catalogs** – External data sources accessible without data duplication. For more information, see the section called “Federated catalogs”.

Topics

Data source benefits

Combining data from data warehouses and data lakes in the lakehouse architecture of Amazon SageMaker offers specific benefits for machine learning workloads, from real-time feature engineering to historical model training datasets.

Amazon S3 Tables benefits

- **Built-in Apache Iceberg support** – First cloud object store optimized for analytics workloads.
- **Performance optimization** – Features designed to continuously improve query performance and reduce storage costs for tables.
- **AWS analytics integration** – Automatic discovery and access by AWS analytics services through AWS Glue Data Catalog and AWS Lake Formation.
- **Specialized storage** – Table buckets provide Amazon S3 storage optimized for analytics workloads.

Amazon Redshift Managed Storage benefits

- **Unified data access** – Query Amazon Redshift tables directly from your lakehouse environment using familiar SQL interfaces.
- **No data movement** – Access Amazon Redshift data in place without ETL processes or data duplication.
- **Consistent governance** – Apply unified access controls and data governance policies across data warehouse and data lake.
- **Performance optimization** – Leverage Amazon Redshift's columnar storage and query optimization for analytical workloads.

Amazon S3 data lakes benefits

- **Scalable storage** – Store structured, semi-structured, and unstructured data at any scale while maintaining high availability, security, and query performance.
- **Multi-zone architecture** – Raw data zone for original format retention, processed data zone for cleaned data, and curated data zone for business-ready datasets.
- **Cost optimization** – Use columnar formats like Parquet or ORC and implement Amazon S3 lifecycle policies for cost-effective storage.
- **Native AWS integration** – Seamless integration with Amazon Athena, AWS Glue, Amazon EMR, SageMaker AI, and Quick.

Federated catalogs benefits

- **No data duplication** – Enable in-place querying without data movement.
- **Streamlined connectivity** – Unified interface for connecting to diverse data sources.
- **Fine-grained permissions** – Catalog, database, table, and column-level access controls.
- **Cross-source analytics** – Support for ad hoc reporting and federated queries across multiple data sources.

Prerequisites for onboarding data to the lakehouse architecture of Amazon SageMaker

Before onboarding data into the lakehouse architecture, ensure you have completed the initial setup. For detailed setup instructions, see *Getting started*.

- AWS account with access to the following AWS services:
  - Amazon S3 including S3 Tables
  - IAM
  - Amazon SageMaker Unified Studio
  - AWS Lake Formation and AWS Glue Data Catalog
  - AWS Glue
- Create a user with administrative access.

- Have access to an IAM role that is a Lake Formation data lake administrator. For instructions, refer to Create a data lake administrator.
- Enable IAM Identity Center in the same AWS Region where you want to create your Amazon SageMaker Unified Studio domain. Set up your identity provider (IdP) and synchronize identities and groups with IAM Identity Center. For more information, refer to IAM Identity Center Identity source tutorials.

Working with Amazon S3 Tables in the lakehouse architecture of Amazon SageMaker

Amazon S3 Tables provide S3 storage that's specifically optimized for analytics workloads, improving query performance while reducing costs. The data in S3 Tables is stored in a new bucket type: a *table bucket*, which stores tables as subresources. S3 tables have built-in support for Apache Iceberg standard, which allows you to easily query tabular data in Amazon S3 table buckets using popular query engines like Apache Spark.

You can integrate Amazon S3 table buckets and tables with AWS Glue Data Catalog (Data Catalog), and register the catalog as a Lake Formation data location from the Lake Formation console or using service APIs. When your organization manages data in the Data Catalog, and register the data location with Lake Formation, you can use Lake Formation to control access to your datasets.

You can apply Lake Formation permissions using tag-based access control and the named resource method on the federated databases, and share them across multiple AWS accounts, AWS Organizations, and organizational units (OUs). You can also share the federated databases directly with IAM principals from another account.

For more information, see Using Amazon S3 Tables with AWS analytics services in the Amazon Simple Storage Service User Guide.

Enable Amazon S3 integration

Before creating Amazon S3 Tables in Amazon SageMaker Unified Studio, you must first enable the integration between Amazon S3 Tables and AWS analytics services.

Enable S3 integration

1. Navigate to the Amazon S3 console. In the left navigation pane, choose **Table buckets**.

2. Choose **Create table bucket**.
3. On the **Create table bucket** page, enter a **Table bucket name** and select **Enable integration**.
4. Choose **Create table bucket**.
5. Amazon S3 displays confirmation when integration of your table buckets with the lakehouse architecture completes.

When you enable the integration, Amazon S3 takes the following actions on your behalf:

- Creates a new service role that gives Lake Formation access to all your tables and table buckets in your current Region. This allows Lake Formation to manage access, permissions, and governance for all current and future table buckets in that Region.
- Creates the `S3tablescatalog` in the AWS Glue Data Catalog in your current Region.

Onboard S3 Tables in the lakehouse architecture

To provide access to S3 tables from Amazon SageMaker Unified Studio, you must grant permissions through Lake Formation.

Grant Lake Formation permissions

1. Navigate to the Lake Formation console.
2. In the left navigation pane, choose **Catalogs** and choose **S3tablescatalog**.
3. From **S3tablescatalog**, under **Objects**, choose your newly created **table bucket**.
4. From the **Actions** menu, select **Grant**.
5. In **Grant permissions**, under IAM users and roles, select your Amazon SageMaker Unified Studio Project role.
6. To grant full access, under **Catalog Permissions > Grant**, select **Super user**.

Integrate Amazon S3 tables with the lakehouse architecture using CLI

1. Register the S3 Tables catalog as a Lake Formation data location.

```
aws lakeformation register-resource \
  --resource-arn 'arn:aws:s3tables:us-east-1:123456789012:bucket/*' \
  --role-arn 'arn:aws:iam::123456789012:role/LakeFormationDataAccessRole' \
```

```
  --with-federation
  --with-privileged-access
```

2. Create a catalog.

```
aws glue create-catalog --cli-input-json file://input.json

'{
   "Name": "s3tablescatalog",
   "CatalogInput" : {
      "FederatedCatalog": {
         "Identifier": "arn:aws:s3tables:us-east-1:123456789012:bucket/*",
         "ConnectionName": "aws:s3tables"
      },
      "CreateDatabaseDefaultPermissions": [],
      "CreateTableDefaultPermissions": []
   }
}'
```

Creating S3 tables catalog in the lakehouse architecture

You can create Amazon S3 table buckets in the lakehouse architecture of Amazon SageMaker as a new data source within Amazon SageMaker Unified Studio. Amazon S3 Tables provide S3 storage optimized for analytics workloads. They include built-in Apache Iceberg support and features designed to continuously improve query performance and reduce storage costs for tables.

Data in S3 Tables is stored in table buckets, which are specialized buckets for storing tabular data. For additional details, see Working with Amazon S3 Tables and table buckets.

To get started using S3 Tables in Amazon SageMaker Unified Studio, create a new Lakehouse catalog with S3 table bucket source.

Create S3 Tables catalog

1. Open the Amazon SageMaker console at https://console.aws.amazon.com/sagemaker/ and use the Region selector in the top navigation bar to choose the appropriate AWS Region.
2. Select your Amazon SageMaker domain.
3. Select the project you want to create a table bucket in.
4. In the navigation menu select **Data**, then select **+** to add a new data source.

5. Select **Create Lakehouse catalog**.
6. In the add catalog menu, choose **Amazon S3 Tables** as the source.
7. Enter a name for the catalog, and a database name.
8. Choose **Create catalog**.

   This creates the following resources in your account:

   - A new S3 Table bucket and the corresponding AWS Glue child catalog under the parent catalog `s3tablescatalog`.
   - A new database within that AWS Glue Data Catalog child catalog. The database name matches the database name you provided. In S3 tables, this is the table namespace.
9. Begin creating tables in your database and querying them using query editor or Jupyter notebook.

Creating and querying Amazon S3 Tables

After setting up your Amazon S3 Tables integration and catalog, you can create databases, tables, and query data using SQL commands in Amazon SageMaker Unified Studio.

Create S3 Table and add data in the lakehouse architecture

1. Navigate to Amazon SageMaker Unified Studio, and select the project.
2. From the **Build** menu, select **Query Editor**, and ensure you have **Amazon Athena** selected in **Connections**.
3. Create a database using SQL.

```
CREATE DATABASE "s3tablescatalog/amzn-s3-demo-bucket".YourDBName;
```

4. Create an Amazon S3 table using SQL.

```
CREATE TABLE "s3tablescatalog/amzn-s3-demo-bucket".YourDBName.YourTableName;
( c_salutation string,
  c_login string,
  c_first_name string,
  c_last_name string,
  c_email_address string)
  TBLPROPERTIES (
  'table_type'='ICEBERG'  );
```

5. Add data using SQL.

```
INSERT INTO "s3tablescatalog/amzn-s3-demo-bucket".YourDBName.YourTableName
VALUES('Dr.','1381546','Joyce','Deaton','Joyce.Deaton@example.edu');
```

You can now use the following integrated analytics services with your Amazon S3 Tables:

- Amazon Athena - create databases, tables, query and add data in Amazon S3 Tables.
- Amazon Redshift - query data from Amazon S3 Tables.
- Amazon EMR - create table, namespace, query and add data in Amazon S3 Tables.
- AWS Glue - create table, namespace, query and add data in Amazon S3 Tables.
- Lake Formation - grant fine-grained permissions for Amazon S3 table catalogs, databases, tables, columns, and cells.

Amazon Redshift Managed Storage for the lakehouse architecture of Amazon SageMaker

You can manage Amazon Redshift tables in the lakehouse architecture of Amazon SageMaker by creating a Amazon Redshift managed catalog in the AWS Glue Data Catalog (Data Catalog). With lakehouse architecture integration, you can access Amazon Redshift tables stored in the Amazon Redshift managed storage (RMS) through Apache Iceberg REST APIs. The lakehouse architecture uses Data Catalog as the technical catalog. The Data Catalog functions as the centralized metadata repository, storing table schemas, partitioning information, and other metadata required for query planning and execution. AWS Lake Formation provides fine-grained access to Redshift tables stored in RMS. You can query and analyze Amazon Redshift data alongside your data lake assets.

Amazon Redshift managed storage overview

Amazon Redshift Managed Storage provides the following benefits for your lakehouse architecture:

- **Unified data access** - Query Amazon Redshift tables directly from your lakehouse environment using familiar SQL interfaces
- **No data movement** - Access Amazon Redshift data in place without ETL processes or data duplication

- **Consistent governance** - Apply unified access controls and data governance policies across data warehouse and data lake
- **Performance optimization** - Leverage Amazon Redshift's columnar storage and query optimization for analytical workloads

Prerequisites for managing Amazon Redshift managed catalog in the AWS Glue Data Catalog

This section covers the prerequisites needed to manage Amazon Redshift managed storage catalogs within the AWS Glue Data Catalog using Lake Formation permissions.

1. AWS account setup
   - AWS account with administrative permissions
   - Lake Formation service enabled in your Region
2. Lake Formation configuration
   - Create a data lake administrator – Create an IAM role that is authorized to create the AWS Glue Data Catalog objects (catalogs, databases, tables/views), and grant Lake Formation permissions to other users.

     For step-by-step instructions on creating a data lake administrator, see Create data lake administrator.

     If the IAM role used for creating federated catalogs is not a data lake administrator, you need to grant the role the `Create catalog` permission.

```
aws lakeformation grant-permissions \
--cli-input-json \
'{
    "Principal": {
     "DataLakePrincipalIdentifier":"arn:aws:iam::123456789012:role/Admin"
    },
    "Resource": {
        "Catalog": {
        }
    },
    "Permissions": [
        "CREATE_CATALOG",
        "DESCRIBE"
```

```
    ]
}'
```

- Create a read only administrator role to discover the Amazon Redshift federated catalogs in the Data Catalog from Amazon Redshift Query Editor v2.

  To query the Amazon Redshift tables in the federated catalog from Amazon Redshift Query Editor v2, ensure that the Read only administrator role policy contains the ARN for the Amazon Redshift service-linked role-`AWSServiceRoleForRedshift`.

```
 aws lakeformation put-data-lake-settings
        region us-east-1 \
        data-lake-settings \
 '{
   "DataLakeAdmins":
 [{"DataLakePrincipalIdentifier":"arn:aws:iam::123456789012:role/Admin"}],
   "ReadOnlyAdmins":
[{"DataLakePrincipalIdentifier":"arn:aws:iam::123456789012:role/aws-service-role/
redshift.amazonaws.com/AWSServiceRoleForRedshift"}],
   "CreateDatabaseDefaultPermissions":[],
   "CreateTableDefaultPermissions":[],
   "Parameters":{"CROSS_ACCOUNT_VERSION":"4","SET_CONTEXT":"TRUE"}

  }'
```

- Data Catalog configured to use Lake Formation permissions
- Default Data Catalog settings disabled (recommended)
- Cross-account version set to 4 or higher is required to grant cross account permissions on the federated catalog objects

3. Create a data transfer role that Amazon Redshift can assume on your behalf to transfer data to and from the Amazon S3 bucket.

   When you enable data lake access for Apache Iceberg compatible query engines such as Athena, Amazon EMR on Amazon EC2 to access the Amazon Redshift resources in the Data Catalog, you need to create an IAM role with the required permissions to perform data transfer to and from the Amazon S3 bucket.

   - glue:GetCatalog
   - glue:GetDatabase

- kms:GenerateDataKey
- kms:Decrypt

4. Add a trust policy (`sts:AssumeRole`) to the data transfer role for AWS Glue and Amazon Redshift services to assume the role to transfer data to and from the Amazon S3 bucket.
5. Add a key policy to the AWS KMS key if you're using a customer managed key to encrypt the data in the Amazon Redshift cluster/namespace. Replace the account number with a valid AWS account number, and specify data transfer role name. By default, the data in the Amazon Redshift cluster is encrypted using an KMS key. Lake Formation provides an option to create your custom KMS key for encryption. If you're using a customer managed key, you must add specific key policies to the key.

   For more information about managing the permissions of a customer managed key, see Customer managed keys.

Creating Amazon Redshift managed catalog in the AWS Glue Data Catalog

You can create a Amazon Redshift managed catalog in the AWS Glue Data Catalog with RMS storage. This catalog will contain the Amazon Redshift tables and databases that are accessible from open source engines to serve to business intelligence (BI) applications.

You can get started by creating an AWS Glue managed catalog using the `glue:CreateCatalog` API or the AWS Lake Formation console by setting the catalog type as `Managed` and `Catalog source` as **Redshift.** This step does the following:

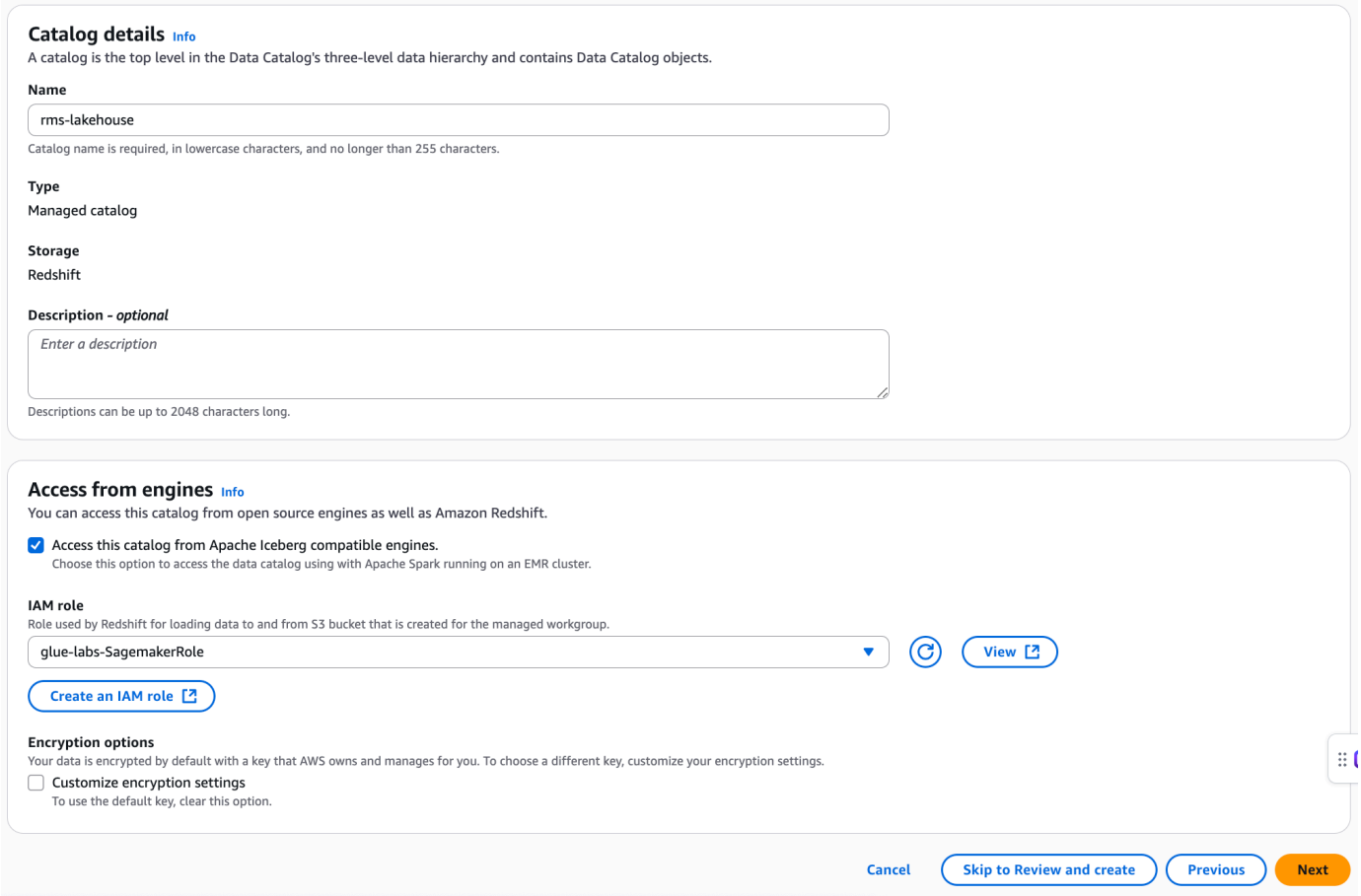
- Creates a catalog in the Data Catalog
- Registers the catalog as a Lake Formation data location
- creates an Amazon Redshift managed serverless-workgroup
- Links Amazon Redshift serverless workgroup and Data Catalog using a datashare object

Creating Amazon Redshift managed catalog using console

To create a managed catalog and set up permissions (console)

1. Open the Lake Formation console at https://console.aws.amazon.com/lakeformation/.
2. In the navigation pane, choose **Catalogs** under **Data Catalog**.

3. Select the option **Create catalog**.
4. Next, choose **Redshift managed storage** as the data source.
5. On the **Set catalog details** page, enter the following information:



- **Name** – A unique name for your managed catalog. The name can't be changed, and must be in lower case. The name can consist of a maximum of 255 characters maximum. account.
- **Description** – Enter a description for the catalog created from the data source.
- Under **Access from engines** make sure that **Access this catalog from Iceberg compatible engines** is selected.

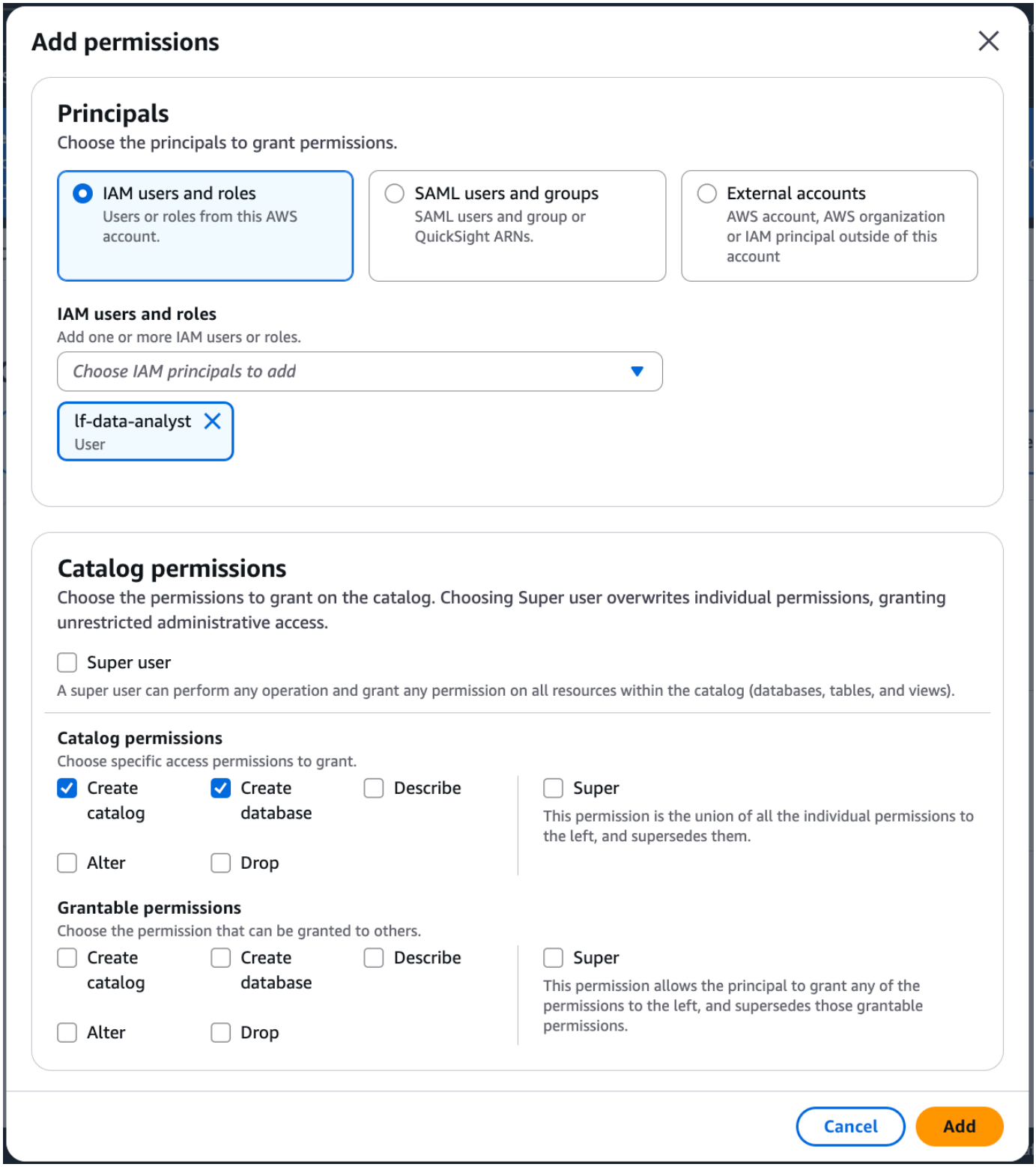
  You can use Apache Spark applications running on Amazon EMR on Amazon EC2 to access the Amazon Redshift databases in the AWS Glue Data Catalog.

  To enable Apache Spark to read and write to Amazon Redshift managed storage, AWS Glue creates a managed Amazon Redshift cluster with the compute and storage resources required to perform read and write operations without impacting Amazon Redshift data warehouse workloads.

- You also need to provide an IAM role with the permissions required to transfer data to and from the Amazon S3 bucket. For the permissions required for the data transfer role, see step 5 in the Prerequisites for managing Amazon Redshift managed catalog in the AWS Glue Data Catalog section.
- **Encryption options** – Choose **Customize encryption settings** option if you want to use a custom key to encrypt the catalog. To use a custom key, you must add additional custom managed key policy to your KMS key.

  By default, the data in the Amazon Redshift cluster is encrypted using an AWS managed key. Lake Formation provides an option to create your custom KMS key for encryption. If you're using a customer managed key, you must add specific key policies to the key.

6. Choose **Next** to grant permissions to other principals.
7. On the **Grant permissions** page, choose **Add permissions**.
8. On the **Add permissions** screen, choose the principals and the types of permissions to grant.



- In the **Principals** section, choose a principal type and then specify principals to grant permissions.
  - **IAM users and roles** – Choose one or more users or roles from the IAM users and roles list.

- **SAML users and groups** – For SAML and Amazon Quick users and groups, enter one or more Amazon Resource Names (ARNs) for users or groups federated through SAML, or ARNs for Amazon Quick users or groups. Press **Enter** after each ARN.
- **External accounts** – Select this option if you want to share the catalog with external accounts, organizations or IAM roles.

  For information about how to construct the ARNs, see AWS CLI grant and revoke AWS CLI commands.

- In the **Permissions** section, select permissions and grantable permissions.

  Under **Catalog permissions**, select one or more permissions to grant.

  Choose **Super user** to grant unrestricted administrative permissions on all resources within the catalog.

  Under **Grantable permissions**, select the permissions that the grant recipient can grant to other principals in their AWS account. This option is not supported when you are granting permissions to an IAM principal from an external account.

9. Choose **Next** to review the information and create the catalog. The **Catalogs** list shows the new managed catalog.

Creating Amazon Redshift managed catalog in the AWS Glue Data Catalog using CLI

To create a federated catalog (CLI)

- The following example shows how to create a federated catalog.

```
aws glue create-catalog --cli-input-json file://input.json

{
    "Name": "CatalogName",
    "CatalogInput": {
      "Description": "Redshift published Catalog",
      "CreateDatabaseDefaultPermissions" : [],
      "CreateTableDefaultPermissions": [],
      "CatalogProperties": {
        "DataLakeAccessProperties" : {
          "DataLakeAccess" : "true",
          "DataTransferRole" : "DTR arn",
```

```
          "KMSKey": "kms key arn",  // Optional
          "CatalogType": "aws:redshift"
        }
      }
    }
}
```

Glue get-catalog response

```
aws glue get-catalog \
  --catalog-id account-id:catalog-name \
  --region us-east-1

Response:
{
    "Catalog": {
        "Name": "CatalogName",
        "Description": "Glue Catalog for Redshift z-etl use case",
        "CreateDatabaseDefaultPermissions" : [],
        "CreateTableDefaultPermissions": [],
         "CatalogProperties": {
          "DataLakeAccessProperties" : {
            "DataLakeAccess": "true",
            "DataTransferRole": "DTR arn",
            "KMSKey": "kms key arn",
            "ManagedWorkgroupName": "MWG name",
            "ManagedWorkgroupStatus": "MWG status",
            "RedshiftDatabaseName": "RS db name",
            "NamespaceArn": "namespace key arn",
            "CatalogType": "aws:redshift"
         }
       }
    }
```

Accessing Amazon Redshift data

Once RMS integration is established, you can access Amazon Redshift data through multiple interfaces:

- **SQL queries** - Use Amazon Athena or other SQL engines to query Amazon Redshift tables alongside Amazon S3 data
- **Data discovery** - Browse Amazon Redshift schemas and tables through the lakehouse data catalog
- **Cross-source joins** - Perform federated queries that join Amazon Redshift data with data lake sources
- **ML workflows** - Access Amazon Redshift data directly in Amazon SageMaker Unified Studio for machine learning model training and inference

Best practices

Follow these best practices when working with Amazon Redshift Managed Storage:

- **Security** - Use IAM roles for authentication and implement least-privilege access principles
- **Performance** - Optimize queries by using appropriate filters and leveraging the distribution of Amazon Redshift and sort keys
- **Cost management** - Monitor query patterns and optimize Amazon Redshift cluster sizing based on usage
- **Data governance** - Apply consistent data classification and access policies across warehouse and lake

Amazon S3 data lakes for the lakehouse architecture of Amazon SageMaker

Amazon S3 serves as the foundational storage layer for your lakehouse architecture, providing unified access to data across Amazon S3 data lakes, Amazon S3 Tables, and Amazon Redshift data warehouses. This unified architecture enables you to build analytics and machine learning applications on a single copy of data, without data movement or duplication. You can also connect to federated data sources such as DynamoDB, Google BigQuery, and Snowflake.

With Amazon S3 Tables integration, you get built-in Apache Iceberg support, making it the first cloud object store optimized for analytics workloads. This enables you to store structured, semi-structured, and unstructured data at any scale while maintaining high availability, security, and query performance.

Amazon S3 data lake architecture

Amazon S3-based data lakes in lakehouse architecture provide unified access to data through:

- **Raw data zone** - Store ingested data in its original format for long-term retention and compliance
- **Processed data zone** - Store cleaned, transformed, and enriched data optimized for analytics
- **Curated data zone** - Store business-ready datasets organized for specific use cases and consumption
- **Metadata catalog** - AWS Glue Data Catalog and Amazon SageMaker Unified Studio catalog provide unified metadata management and data discovery

Data organization and partitioning

Effective data organization in Amazon S3 improves query performance and reduces costs:

- **Hierarchical structure** - Organize data using logical folder structures based on business domains, data sources, or time periods
- **Partitioning strategy** - Partition data by frequently queried dimensions such as date, region, or category
- **File formats** - Use columnar formats like Parquet or ORC for analytical workloads to optimize compression and query performance
- **Lifecycle policies** - Implement Amazon S3 lifecycle policies to automatically transition data to cost-effective storage classes

Integration with AWS analytics and ML services

Amazon S3 data lakes integrate seamlessly with AWS analytics and ML services, allowing you to query and analyze data in-place:

- **Athena** - Query data directly from Amazon S3 using standard SQL without data movement
- **AWS Glue** - Discover, catalog, and transform data with serverless ETL capabilities
- **Amazon EMR** - Process large datasets using Apache Spark, Hadoop, and other big data frameworks
- **SageMaker AI** - Access data for machine learning model training and inference

- **Quick** - Create interactive dashboards and visualizations

Security and governance

Implement comprehensive security and governance for your Amazon S3 data lake:

- **Access controls** - Use IAM policies, bucket policies, and Lake Formation for fine-grained access control
- **Encryption** - Enable server-side encryption for data at rest and use SSL/TLS for data in transit
- **Data classification** - Tag and classify data based on sensitivity and compliance requirements
- **Audit logging** - Enable CloudTrail and Amazon S3 access logging for comprehensive audit trails

Federated catalogs for the lakehouse architecture of Amazon SageMaker

The lakehouse architecture with integrated access controls for Athena federated queries enables you to connect to and query data across multiple data sources without moving or duplicating data. Federated catalogs establish secure links to external data sources, breaking down data silos while maintaining consistent governance and security controls. For a step-by-step guide, see Get started with lakehouse architecture integrated access controls for Athena federated queries.

Overview

Federated connections address common data infrastructure challenges:

- **Complex connectivity setup** - Streamline connection creation through a unified interface
- **Fragmented governance** - Centralize access control management through Lake Formation
- **Data duplication** - Enable in-place querying without data movement

Key capabilities include:

- Unified interface for connecting to diverse data sources
- Fine-grained permissions at catalog, database, table, and column levels
- In-place querying through federated catalogs
- Support for ad hoc reporting and proof of concept analysis

Supported data sources

The lakehouse architecture federated catalogs support the following data sources:

Data Source	Type
Google BigQuery	Database
Amazon DocumentDB	Database
DynamoDB	Database
Amazon Redshift	Database
MySQL	Database
PostgreSQL	Database
SQL Server	Database
Snowflake	Database
Oracle	Database
Aurora MySQL	Database
Aurora Postgres	Database
Microsoft Azure SQL	Database

Note

The lakehouse architecture currently supports lowercase table, column, and database names. For optimal experience, ensure that all database identifiers are in lowercase.

For more information about data connections and their capabilities, see Data connections in the lakehouse architecture of Amazon SageMaker.

Setting up federated connections

The process for implementing federated connections involves these high-level steps:

1. **Create federated connections**
   - Establish connections that serve as bridges between lakehouse architecture and external data sources
   - Configure secure connectivity while maintaining security boundaries
   - Eliminate the need for data movement or duplication
2. **Create federated catalogs**
   - Establish catalogs containing metadata about tables from connected data sources
   - Make external tables discoverable and queryable through the Lakehouse interface
   - Use catalogs as directories of available data assets
3. **Implement access controls**
   - Configure fine-grained permissions using Lake Formation
   - Apply column-level security for sensitive data
   - Ensure consistent security policies across all data sources
4. **Validate and query**
   - Test access permissions with different user personas
   - Run federated queries across multiple data sources
   - Verify security controls and data access policies

Prerequisites

Before setting up federated connections, ensure you have:

- AWS account with permissions to create IAM roles and policies
- Data lake administrator role in Lake Formation
- SageMaker AI Unified Studio domain with SQL Analytics profile enabled
- SageMaker AI Unified Studio projects configured for administration and data analysis
- Administrator access to target data sources

Best practices

Follow these best practices when implementing federated connections:

- **Security** - Implement least-privilege access and use column-level security for sensitive data
- **Performance** - Optimize queries by using appropriate filters and limiting data scanned
- **Monitoring** - Track query performance and resource usage across federated sources
- **Governance** - Maintain consistent data classification and access policies

For detailed implementation steps, see Get started with lakehouse architecture integrated access controls for Athena federated queries.

Share your data in the lakehouse architecture of Amazon SageMaker

Cross-account data sharing allows you to securely share data across accounts without duplicating data. This approach centralizes access control using fine-grained permissions and enables cross-account analytics and machine learning workflows.

Topics

Tutorial: configure cross-account access for Redshift federated catalog table

This tutorial provides step-by-step instructions on how to configure cross account sharing of Redshift federated catalog tables using Lake Formation permissions. For more information, see Sharing a data lake using Lake Formation fine-grained access control.

Prerequisites

Before proceeding, ensure you have the following prerequisites:

- Two accounts; account A as the sharing account, account B as the recipient account.
- Two AWS accounts with Lake Formation cross-account sharing version 4 and Lake Formation administrator configured. For more information, see Data lake administrator permissions and initial setup of Lake Formation.
- Permissions for managing Amazon Redshift namespaces in the Data Catalog granted to the Lake Formation administrator role on both accounts.
- An Amazon S3 bucket in the account A account to host the sample Iceberg table data.
- An IAM role in the account A account to register your Amazon S3 location for Iceberg table with Lake Formation. For more information, see Registering an Amazon S3 location and Requirements for roles used to register locations.
- An Redshift Serverless namespace in the account A account. Follow the instructions in Creating a data warehouse with Amazon Redshift Serverless to launch a serverless namespace with default settings.

- An IAM role, `Glue-execution-role`, in account B with appropriate policies.

Required IAM policies for `Glue-execution-role`

The `Glue-execution-role` in account B requires the following AWS managed policies:

- `AWSGlueServiceRole`
- `AmazonRedshiftDataFullAccess`

Additionally, create an inline policy with the following permissions:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "LFandRSserverlessAccess",
            "Effect": "Allow",
            "Action": [
                "lakeformation:GetDataAccess",
                "redshift-serverless:GetCredentials"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "glue.amazonaws.com"
                }
            }
        }
    ]
}
```

Add the following trust policy to `Glue-execution-role`:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "glue.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

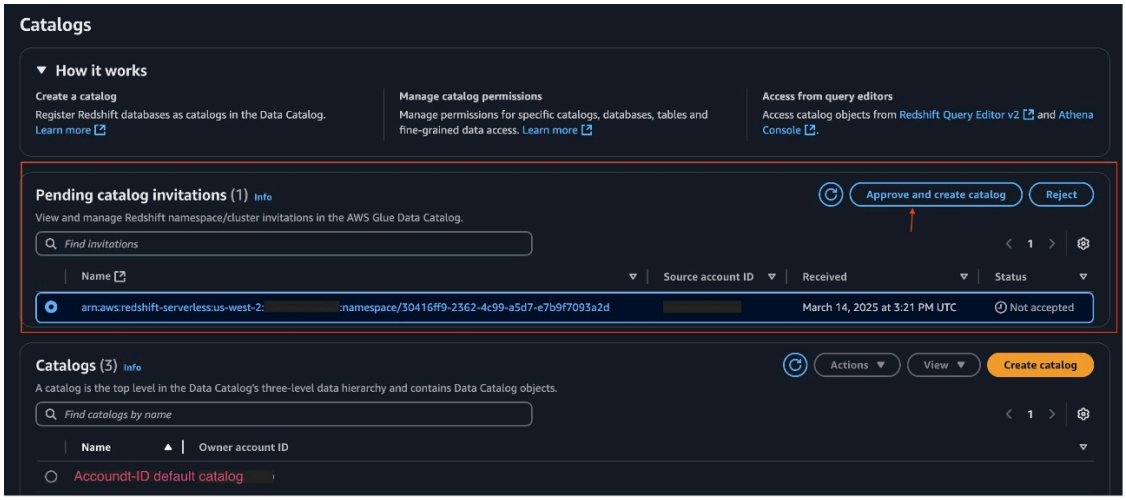
Configure account A

In account A, configure the catalog, grant permissions to account B, and register the object storage location with fine-grained permissions. You can use either your IAM administrator role (added as administrator) or a role with the permissions described in the prerequisites.

Configure your catalog

To configure your catalog in account A:

1. Log in to the AWS Management Console as an administrator.
2. Open the Amazon Redshift console, register your Amazon Redshift clusters and namespaces to the Data Catalog.
3. After the registration is initiated, you will see an invite from Amazon Redshift on the Lake Formation console.
4. Select the pending catalog invitation and choose **Approve and create catalog**.



5. On the **Set catalog details** page, configure your catalog:
   a. For **Name**, enter a name.
   b. Select **Access this catalog from Apache Iceberg compatible engines**.
   c. Choose the IAM role you created for the data transfer and choose **Next**.
6. On the **Grant permissions – optional** page, choose **Add permissions**.
   a. For **IAM users and roles**, choose **Admin**
   b. For **Catalog permissions**, grant **Super user** to catalog permissions and grantable permissions.
   c. Choose **Add**.
7. Review the details on the **Review and create** page and choose **Create catalog**.
8. Verify that the catalog is created.
9. Explore the catalog detail page to verify the database and table structure.
10. On the database **View** dropdown menu, view the table and verify that the table shows up.

> **Note**
>
> As the **Admin** role, you can also query the table in Amazon Athena and confirm that the data is available.

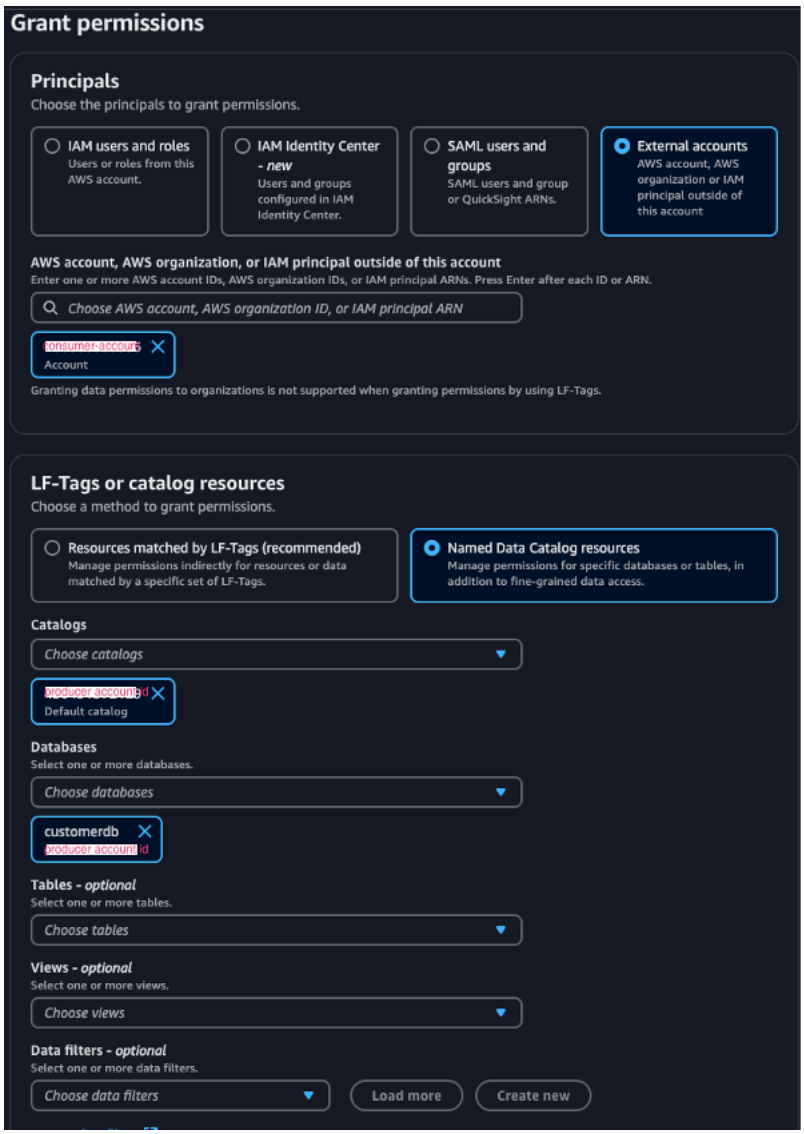
Grant permissions on the tables from account A to account B

Share the data warehouse federated catalog database and table, as well as the object storage-based Iceberg table and its database from the default catalog to account B.

> **Note**
>
> You cannot share the entire catalog to external accounts as a catalog-level permission; you can only share the database and table.

To grant permissions:

1. On the Lake Formation console, choose **Data permissions** in the navigation pane.
2. Choose **Grant**.
3. Under **Principals**, select **External accounts** and provide the account ID of account B.
4. Under **LF-Tags or catalog resources**, select **Named Data Catalog resources**.
5. For **Catalogs**, choose the account ID that represents the default catalog.
6. For **Databases**, choose the database you want to share (for example, *`customerdb`*).



7. Under **Database permissions**, select **Describe** under both **Database permissions** and **Grantable permissions**.
8. Choose **Grant**.
9. Repeat these steps to grant table-level **Select** and **Describe** permissions on the tables you want to share.
10. Repeat these steps again to grant database and table level permissions for the federated catalog database.
11. Choose **Data permissions** in the navigation pane and verify that account B has been granted database and table level permissions for both tables from the federated catalog and from the default catalog.

Register the Amazon S3 location

Register the object storage-based Iceberg table data location with fine-grained permissions so that it can be managed by permissions.

1. On the Lake Formation console, choose **Data lake locations** in the navigation pane.
2. Choose **Register location**.
3. For **Amazon S3 path**, enter the path for your storage bucket that contains the Iceberg table data.
4. For **IAM role**, provide the user-defined role *`LakeFormationS3Registration_custom`* that you created as a prerequisite.
5. For **Permission mode**, choose **Lake Formation**. Choose **Register location**.
6. Choose **Data lake locations** in the navigation pane to verify the S3 registration.

Configure account B

To configure the receipient account, account B, the Lake Formation administrator accepts the AWS Resource Access Manager (AWS RAM) shares, creates resource links that point to the shared catalog, database, and tables, and configures permissions for the AWS Glue execution role (*`Glue-execution-role`*).

Accept and verify the shared resources

Lake Formation uses AWS RAM shares to enable cross-account sharing with Data Catalog resource policies. To view and verify the shared resources from account A:

1. Log in to the AWS Management Console from account B and set the AWS Region to match the shared resource Region of account A.
2. Open the https://console.aws.amazon.com/lakeformation/. You will see a message indicating there is a pending invite.
3. Follow the instructions to review and accept the pending invites on the AWS RAM console.
4. When the invite status changes to **Accepted**, choose **Shared resources** under **Shared with me** in the navigation pane.
5. Verify that the shared resources display correctly with ID of account A under the **Owner ID** column.

> **Note**
>
> You won't see an AWS RAM share invite for the catalog level on the Lake Formation console, because catalog-level sharing isn't possible. You can review the shared federated catalog and Amazon Redshift managed catalog names on the AWS RAM console, or using the AWS Command Line Interface (AWS CLI) or SDK.

Create a catalog link container and resource links

A catalog link container is a data catalog object that references a local or cross-account federated database-level catalog from other AWS accounts. For more details, see Accessing a shared federated catalog.

Create a catalog link container that points to federated catalog in account A:

1. On the Lake Formation console, under **Data Catalog** in the navigation pane, choose **Catalogs**.
2. Choose **Create catalog**.
3. Provide the following details for the catalog:
   a. Enter a name for the catalog.
   b. For **Type**, choose **Catalog Link container**.
   c. For **Source**, choose **Amazon Redshift**.
   d. For **Target Redshift Catalog**, enter the ARN of the federated catalog in account A.

      ```
      arn:aws:glue:us-west-2:<<account A ID>>:catalog/redshiftserverless1-uswest2/ordersdb
      ```

   e. Under **Access from engines**, select **Access this catalog from Apache Iceberg compatible engines**.
   f. For **IAM role**, provide the Redshift-S3 data transfer role that you had created in the prerequisites.
   g. Choose **Next**.
4. On the **Grant permissions – optional** page, choose **Add permissions**.
   a. Grant the `Admin` user **Super user** permissions for **Catalog permissions** and **Grantable permissions**.
   b. Choose **Add** and then choose **Next**.

5. Review the details on the **Review and create** page and then choose **Create catalog**.

   Wait a few seconds for the catalog to show up.

6. In the navigation pane, choose **Catalogs** and verify that your catalog is created.

Create a database under the catalog link container

After creating the catalog link container, create a database under your catalog:

1. On the Lake Formation console, under **Data Catalog** in the navigation pane, choose **Databases**.
2. On the **Choose catalog** dropdown menu, choose your catalog link container.
3. Choose **Create database**.
4. Provide details for the database:
   a. Enter a name.
   b. For **Catalog**, choose select your catalog link container.
   c. Under **Default permissions for newly created tables**, clear the **Use only IAM access control for new tables in this database** box.
   d. Choose **Create database**.
5. Choose **Catalogs** in the navigation pane to verify that database is created under your catalog.

Create a table resource link for the shared federated catalog table

A resource link to a shared federated catalog table can reside only inside the database of a catalog link container. A resource link for such tables will not work if created inside the default catalog. For more details on resource links, see Creating a resource link to a shared Data Catalog table.

To create a table resource link:

1. On the Lake Formation console, under **Data Catalog** in the navigation pane, choose **Tables**.
2. On the **Create** dropdown menu, choose **Resource link**.
3. Provide details for the table resource link:
   a. For **Resource link name**, enter a name.
   b. For **Destination catalog**, choose catalog you created.
   c. For **Database**, choose your database.

d. Choose a region for **Shared table's region**.

e. For **Shared table**, choose the table name.

f. After you choose the **Shared table**, the **Shared table's database** and **Shared table's catalog ID** gets automatically populated.

g. Choose **Create**.

h. In the navigation pane, choose **Databases** to verify that table resource link is created under your database, inside the catalog you choose.

Create a database resource link for the shared federated catalog table

Create a database resource link in the default catalog to query the S3 based Iceberg table shared from account A. For details on database resource links, refer Creating a resource link to a shared Data Catalog database.

> **Note**
>
> A resource link is required to query from analytics engines, such as Athena, Amazon EMR, and AWS Glue. When you use AWS Glue with Lake Formation, the resource link name must be identical to the source account's resource. For additional details on using AWS Glue with Lake Formation, see Considerations and limitations.

To create a database resource link:

1. On the Lake Formation console, under **Data Catalog** in the navigation pane, choose **Databases**.
2. On the **Choose catalog** dropdown menu, choose the account ID to choose the default catalog.
3. Choose a darabase, and on the **Create** dropdown menu, choose **Resource link**.
4. Provide details for the resource link:
   a. For **Resource link name**, enter a name.

      The rest of the fields will automatically populate.

   b. Choose **Create**.
5. In the navigation pane, choose **Databases** and verify that your database is created under the default catalog. Resource link names will show in italicized font.

Verify access using Athena

Verify your access by running test queries in Athena:

1. Open the Athena console and ensure an Amazon S3 bucket is configured to store query results. For more information, see Specify a query result location using the Athena console.
2. In the navigation pane, verify both the default catalog and federated catalog tables by previewing them.
3. Run a join query using the three-point notation for referring to tables from different catalogs as show in the following example:

```
SELECT
    returns_tb.market as Market,
    sum(orders_tb.quantity) as Total_Quantity
FROM rl_link_container_ordersdb.public_db.rl_orderstbl as orders_tb
JOIN awsdatacatalog.customerdb.returnstbl_iceberg as returns_tb
ON orders_tb.order_id = returns_tb.order_id
GROUP BY returns_tb.market;
```

Grant permissions to the `Glue-execution-role`

Set up Lake Formation permissions on the catalog link container, databases, tables, and resource links for the AWS Glue job execution role *`Glue-execution-role`* that you created in the prerequisites:

1. On the Lake Formation console, choose **Data permissions** in the navigation pane.
2. Choose **Grant**.
3. Under **Principals**, select **IAM users and roles** and enter *`Glue-execution-role`*.
4. Under **LF-Tags or catalog resources**, select **Named Data Catalog resources**.
5. For **Catalogs**, choose your catalog and the account ID of account B, which indicates the default catalog.
6. Under **Catalog permissions**, select **Describe** for **Catalog permissions**.
7. Repeat these steps to grant additional permissions to *`Glue-execution-role`*.

Run a PySpark job in AWS Glue 5.0

Download the PySpark script . This AWS Glue PySpark script runs Spark SQL by joining the shared federated table in account A and S3 based table in account B to analyze the data and identify the total orders placed per market.

Create and run an AWS Glue job that joins the shared federated table in account A and Amazon S3 based table in account B:

1. On the AWS Glue console, in the navigation pane, choose **ETL jobs**.
2. Choose **Create job**, then choose **Script editor**.
3. For **Engine**, choose **Spark** and for **Options**, choose **Start fresh**.
4. Upload your PySpark script that contains the Spark SQL join query.
5. On the **Job details** tab:
   a. Provide the job name.
   b. Choose *`Glue-execution-role`* for the IAM role.
   c. For **Glue version**, select **Glue 5.0**.
   d. Under **Advanced properties**, for **Job parameters**, choose **Add new parameter** and add the following parameters:
      - `--datalake-formats = iceberg`
      - `--enable-lakeformation-fine-grained-access = true`
6. Save the job and choose **Run** to execute it.
7. Review the job run details from the **Output logs**.

Clean up the resources

To avoid incurring costs on your AWS accounts, delete the following resources that you created:

- Lake Formation permissions, catalog link container, database, and tables in account B
- AWS Glue job in account B
- Federated catalog, database, and table resources in account A
- Redshift Serverless namespace in account A
- Amazon S3 buckets that you created as part of data transfer in both accounts
- Athena query results bucket in account B

- IAM roles for the lakehouse architecture setup

Document history for the lakehouse architecture of Amazon SageMaker User Guide

The following table describes the documentation releases for the lakehouse architecture.

Change	Description	Date
Added new sections	Added the getting started and sections.	August 26, 2025
Initial release	Initial release of the lakehouse architecture of Amazon SageMaker User Guide.	July 31, 2025
Added new sections	Added Onboarding data and sharing data section.	March 10, 2025