Demystifying Data Analytics in Amazon SageMaker Unified Studio

Mental Model and Practical Quick Start with CloudFormation templates

Introduction

In this post, we'll demonstrate how Amazon SageMaker Lakehouse and Amazon SageMaker Unified Studio can streamline data analytics by seamlessly integrating diverse data sources, empowering teams to unlock meaningful insights at speed. We will also understand key building blocks and how they work together. With Zero-ETL capabilities, data teams can eliminate time-consuming and fragmented data preparation tasks while leveraging various built-in query editors to interact with their data using familiar SQL and visual interfaces. This allows customers to focus on extracting value from data such as developing new data products or identifying data patterns while Amazon SageMaker handles the complex infrastructure management and resource provisioning automatically behind the scenes.

Amazon SageMaker Unified Studio

Amazon SageMaker Unified Studio provides an integrated environment for domain-specific data projects. The environment is provisioned through AWS-managed blueprints powered by CloudFormation templates and organized into specialized projects built based on the project profiles ranging from SQL analytics, GenAI applicarion development to full-fledged project profile encompassing all capabilities (data exploration and processing, AI model development and training).

Users authenticated via IAM or SSO can work within these projects to unlock the value of data by:

- Connecting multiple data sources, including Amazon S3 data lakes, Amazon Redshift data warehouses and managed storage, and federated sources
- Unifying these sources through the Amazon SageMaker Lakehouse and registering in the technical data catalogs
- Analyzing data using query engines like Amazon Athena or Amazon Redshift Query Editor v2, or exploring and processing data programmatically using JupyterLab Notebooks
- Leveraging Generative SQL by interacting with a chatbot powered by Amazon Q and asking natural language questions in order to understand the actionable context of domain data and create SQL queries
- Governing unified and shared data access through assets exposed via a business catalog

Projects in Amazon SageMaker Unified Studio also serve as collaboration and permission boundaries, with consistent access policies using a single permission model with granular controls powered by AWS Lake Formation.

Amazon SageMaker Unified Studio makes it easy for customers to find and access data from across their organization and brings together purpose-built AWS analytics and AI/ML capabilities so that customers can act on their data using the best tool for the job across all types of common data use cases. With built-in Amazon Q Developer assistance, teams can accelerate their data projects through AI-powered guidance and intelligent SQL recommendations.

The following Figure 1. illustrates the key building blocks within Amazon SageMaker Unified Studio and their interconnections with familiar AWS data services. All blocks are labeled according to the official SageMaker Unified Studio terminology, providing a clear structure for understanding the platform's composition and its integration with the existing tooling. You can find definition of each component and concepts in the documentation: SageMaker Unified Studio terminology.

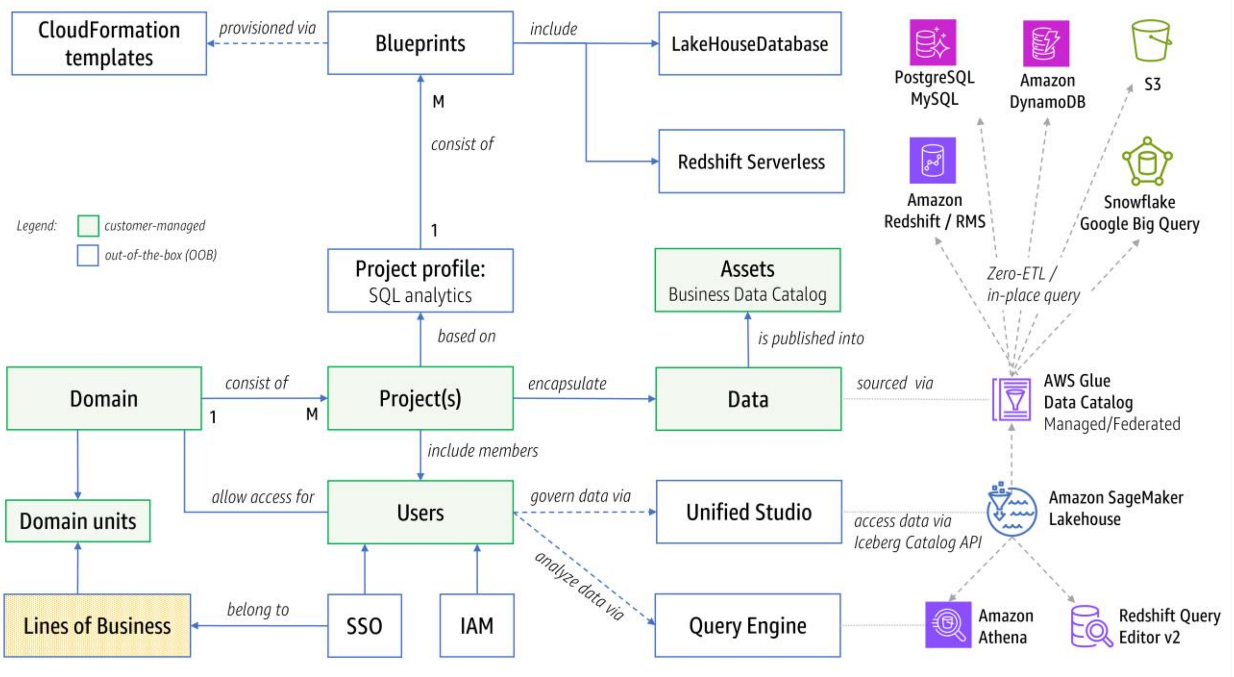


Figure 1. Amazon SageMaker Unified Studio - Building Blocks

Amazon SageMaker Lakehouse

Amazon SageMaker Lakehouse is a capability that unifies data across Amazon S3 data lakes, Amazon Redshift data warehouses and operational databases, enabling you to build powerful analytics and AI/ML applications on a single copy of data, while eliminating data silos, and accelerating time-to-insight through cohesive data access across your organization.

Amazon SageMaker Lakehouse offers three major benefits:

- **Unified and consistent data access** - Amazon SageMaker Lakehouse uses a technical data catalog powered by AWS Glue Data Catalog as the primary interface for interaction with selected data sources.
- **Centralized permission control** - Amazon SageMaker Lakehouse supports fine-grained access control to your data, regardless of the underlying storage formats or query engines used.
- **Open source compatibility** - Amazon SageMaker Lakehouse leverages open-source Apache Iceberg, enabling data interoperability across various Apache Iceberg compatible query engines and tools. Data in SageMaker Lakehouse can be accessed from Apache Iceberg compatible engines such as Amazon Athena, Apache Spark, or Amazon EMR.

Amazon SageMaker Lakehouse provides consistent approach to establish data connection and access new and existing data sources through data catalogs, and query data in-place using the Apache Iceberg open standard and Amazon Athena query editor. There are two types of catalogs in the Lakehouse:

- Managed data catalogs include:
    - Amazon S3 data lakes, including Amazon S3 Tables with built-in Apache Iceberg support, making it easy to query and join S3 data lake and S3 Tables with data in the lakehouse.
    - Lakehouse catalogs backed by Redshift managed storage (RMS catalog) and compatible with Apache Iceberg table format. Also, RMS catalogs are supported destinations for Zero-ETL integration from SaaS sources (like Salesforce and SAP) and Amazon DynamoDB
- Federated data catalogs include:
    - Federated data sources, allowing you to access and analyze live data directly through lakehouse architectures. You can connect to operational databases (PostgreSQL, MySQL, Microsoft SQL Server), AWS managed databases (supported engines on Amazon Aurora and RDS, Amazon DynamoDB, Amazon DocumentDB) and third-party data sources (Snowflake, Google BigQuery) - all without moving the data. When you set up a federated data connection in Unified Studio project, AWS automatically provisions the required infrastructure components, including AWS Glue connection, AWS Glue Data Catalog, and Lambda function, which acts as a bridge between lakehouse and the federated data source.

While Amazon SageMaker Lakehouse integrates data through data catalog interface, you can still take advantage of Redshift processing power to run analytics directly on the data residing within Redshift managed storage, particularly for complex queries on massive datasets. To work with data stored in Amazon Redshift in Amazon SageMaker Unified Studio, you can choose from these approaches:

- Connect your existing data warehouse as a compute engine to achieve maximum performance. This approach can provide access to the entire workgroup or node, depending on your permissions. You'll have access to both native databases and auto-mounted data catalogs. Auto-mounted catalogs include the Amazon S3 data catalog (named AWSDataCatalog) and Lakehouse catalog (RMS catalog powered by Redshift Managed Storage). These catalogs are automatically discovered as external databases in your Amazon Redshift data warehouses once you establish the necessary permissions. You can work with data (read/write) and analyze it using Redshift Query Editor v2 and Redshift SQL.

- Through AWS Glue Zero-ETL replication, you can bring data from operational databases (such as Amazon Aurora, Amazon RDS for MySQL, Amazon DynamoDB), and SaaS sources (like Salesforce, SAP, Zendesk), and load data into Amazon Redshift data warehouse or Redshift managed storage in near real-time without writing any ETL job.

- Connect existing data warehouse as a federated data source through Amazon SageMaker Lakehouse. You can select a specific native database. This database can be either local or a destination database where data is replicated through Zero-ETL. You can query data using Amazon Athena and Athena SQL. With this approach, consider performance implications of the underlying AWS Glue connection and Lambda configuration.

The next Figure 2. illustrates possible data access options supported by Amazon SageMaker Lakehouse and Amazon Redshift in context of Amazon SageMaker Unified Studio.

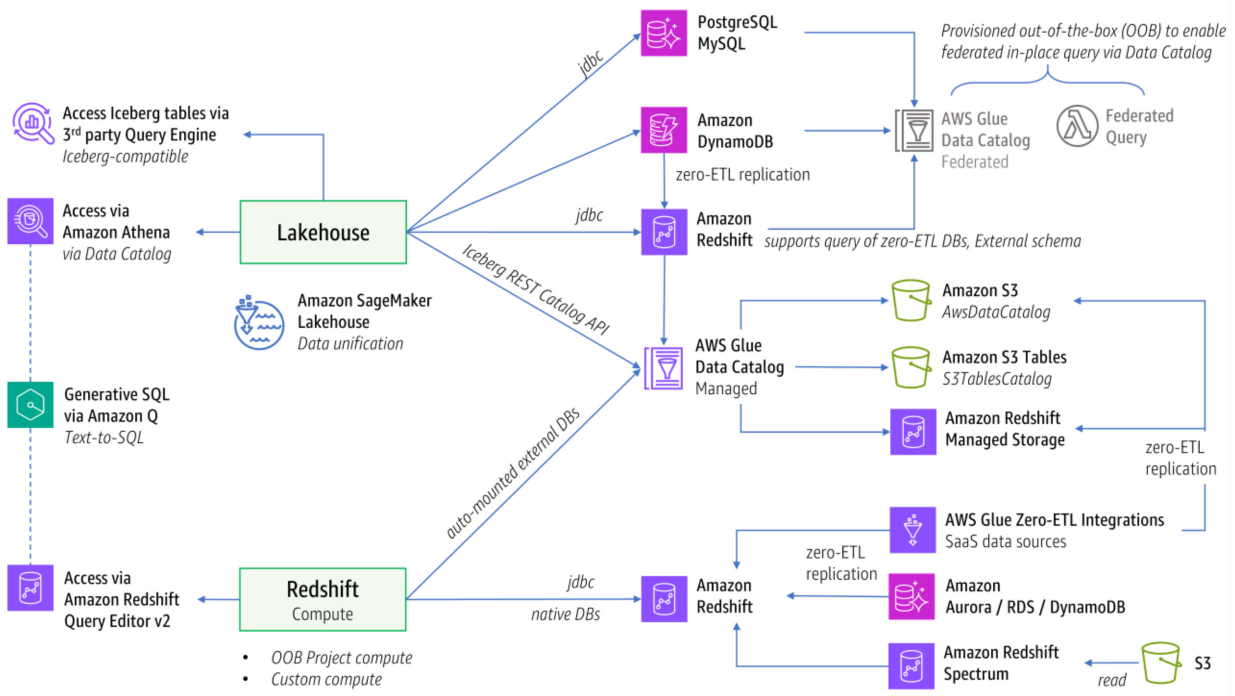


Figure 2. Amazon SageMaker Lakehouse

Data Access control

When reading or writing data, Amazon SageMaker Lakehouse follows multiple layers of security controls. The security begins with AWS Identity and Access Management (IAM), which provides coarse-grained permissions through project-specific roles. Next, AWS Lake Formation extends the foundational IAM functionality to deliver unified, fine-grained permission management across your data assets at metadata level. Lastly, resource-facing permissions complete this security architecture, using either resource-based policies or database user credentials to control access at the storage level.

AWS Lake Formation provides a central permission store for data sources registered in AWS Glue Data Catalog and available in Amazon SageMaker Lakehouse. Lake Formation uses Data Catalog as a unified layer for managing permissions consistently across disparate data sources: S3 data lakes, Redshift warehouses, RMS catalogs, federated data sources. It enables selected principals (such as Unified Studio project IAM role) to access catalog entities through flexible permissions ranging from full access to fine-grained controls at the column, row, and cell levels. This permission model is enforced consistently across AWS analytics and machine learning services, including Amazon Athena, Amazon QuickSight, Amazon Redshift Query Editor v2, Redshift Spectrum, Amazon EMR, and AWS Glue.

When project users perform SQL analytics in Amazon SageMaker Unified Studio, their access is authorized through a project-specific IAM role (with name following this format *datazone_usr_role_{ProjectID}_{EnvironmentID}*). This role incorporates the AWS-managed policy **SageMakerStudioProjectUserRolePolicy**, which manages broad permission sets for data lake, data warehouse, catalog resources and more. Connection to existing data sources requires the presence of specific tags on the secrets in AWS Secrets Manager — such as **AmazonDataZoneProject**—to activate the project permissions defined in the **SageMakerStudioProjectUserRolePolicy** policy.

The next Figure 3. illustrates a logical composition of this layered architecture in establishing secure access to the selected data sources:

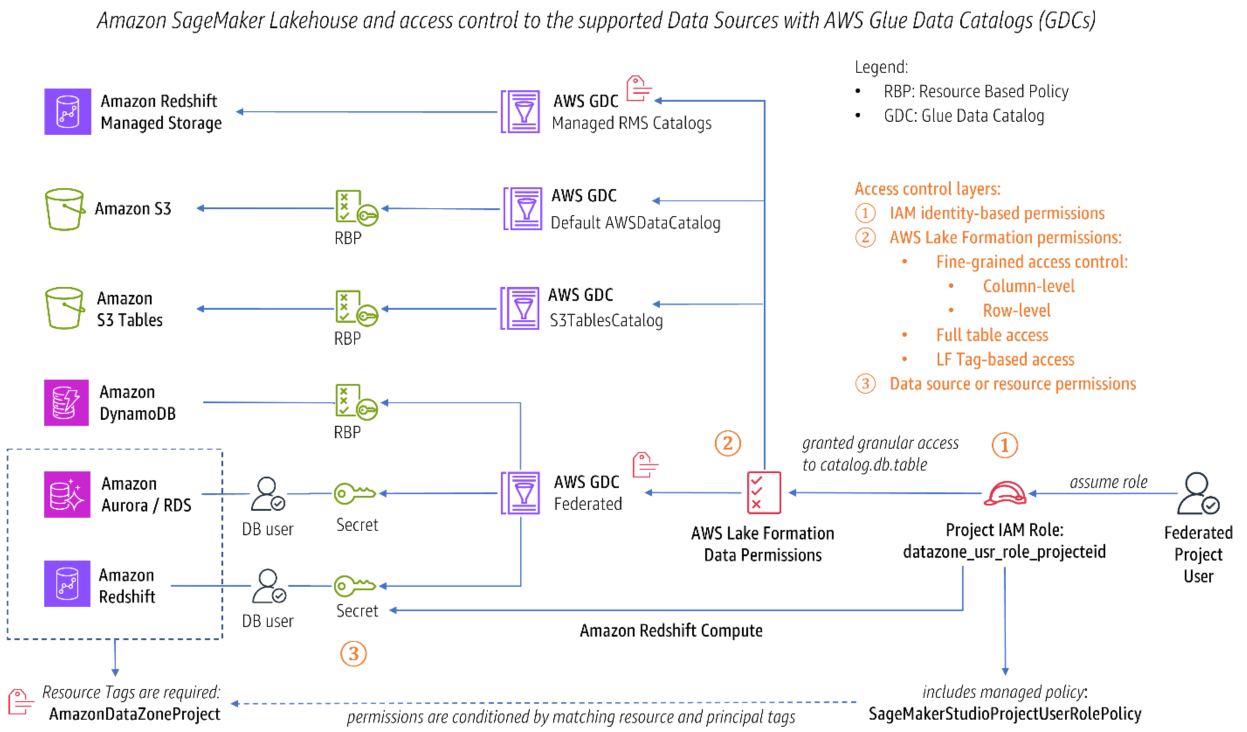

Amazon SageMaker Lakehouse and access control to the supported Data Sources with AWS Glue Data Catalogs (GDCs)
Legend:
RBP: Resource Based Policy
GDC: Glue Data Catalog
Access control layers:
1 IAM identity-based permissions
2 AWS Lake Formation permissions:
Fine-grained access control:
Column-level
Row-level
Full table access
LF Tag-based access
3 Data source or resource permissions
Amazon Redshift Managed Storage
AWS GDC Managed RMS Catalogs
Amazon S3
RBP
AWS GDC Default AWSDataCatalog
Amazon S3 Tables
RBP
AWS GDC S3TablesCatalog
Amazon DynamoDB
RBP
Amazon Aurora / RDS
DB user
Secret
AWS GDC Federated
Amazon Redshift
DB user
Secret
3
2
granted granular access to catalog.db.table
1
assume role
AWS Lake Formation Data Permissions
Project IAM Role: datazone_usr_role_projecteid
Federated Project User
Amazon Redshift Compute
Resource Tags are required: AmazonDataZoneProject
permissions are conditioned by matching resource and principal tags
includes managed policy: SageMakerStudioProjectUserRolePolicy

Figure 3. Amazon SageMaker Lakehouse - Data Access Layers

Now that you know what are the major components of Amazon SageMaker Unified Studio and SageMaker Lakehouse, let’s try data analytics in action.

Demo Solution Architecture

We have demo data residing in these three locations:

- Customer data in Amazon Aurora PostgreSQL
- Sales data with the inventory data in Amazon Redshift Serverless
- Financial data such as invoices residing in Amazon DynamoDB

To simplify provisioning resources for this blog, infrastructure components such as VPC, subnets, security groups will be reused and both Amazon SageMaker domain and data sources will be provisioned along each other. To establish connectivity with the Aurora PostgreSQL and Redshift Serverless, we will use database admin users and federate through out-of-the-box default IAM role (Project IAM Role) linked to the project in the Amazon SageMaker Unified Studio. Access to demo data sources is managed through AWS Lake Formation's default permissions model, which grants Super access to all federated data catalog tables. You can customize these permissions at the column, row, and cell levels using Lake Formation's grant/revoke mechanisms.

The next diagram illustrates these disparate data sources and explains the pathways by which data analysts can interact with these datasets after they are seamlessly integrated into Amazon SageMaker Studio.

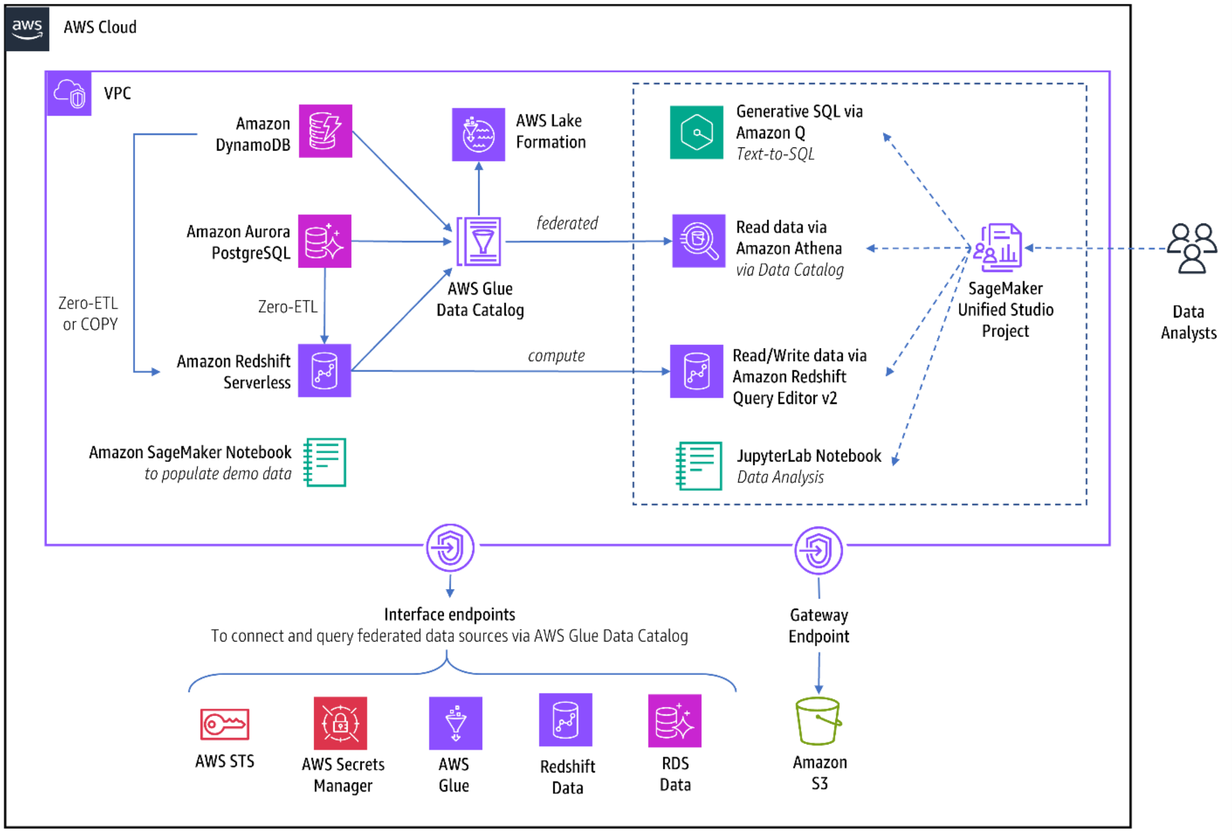


Figure 4. Demo Architecture

Demo Data Model

The following entity relationship diagram illustrates a typical data landscape where interconnected data of varying formats and types can be distributed across multiple lines of business, but easily united into cohesive data lakes through lakehouse.

- Customer data representing well-structured entities with complex relationships
- Sales and inventory data is optimized for analytics and data warehousing, to maintain large-scale reporting and complex queries
- Financial data like invoices often stored as semi-structured NoSQL format designed for high-performance, scalability

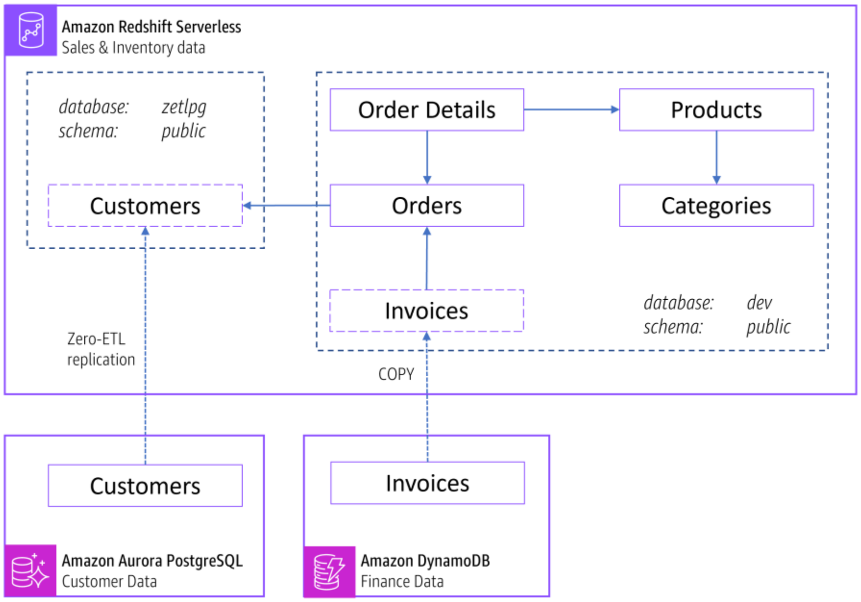


Figure 5. Demo Data Model

Zero-ETL integrations

In this post we will apply the following Zero-ETL integration options for near real-time analytics in Amazon SageMaker Lakehouse through the respective query editors:

- Federated Query of data in Amazon Aurora and Amazon DynamoDB using Amazon Athena. This approach queries data in-place at the source through a built-in connector without moving data.
- Zero-ETL replication from Amazon Aurora PostgreSQL based on a specified interval, automatically replicating data to the designated destination database and available for querying within Amazon Redshift. Additionally, you can also configure near real-time Zero-ETL replication from Amazon DynamoDB to the destination database.
- Zero-ETL replication from Amazon DynamoDB into Amazon Redshift Serverless by running the COPY command

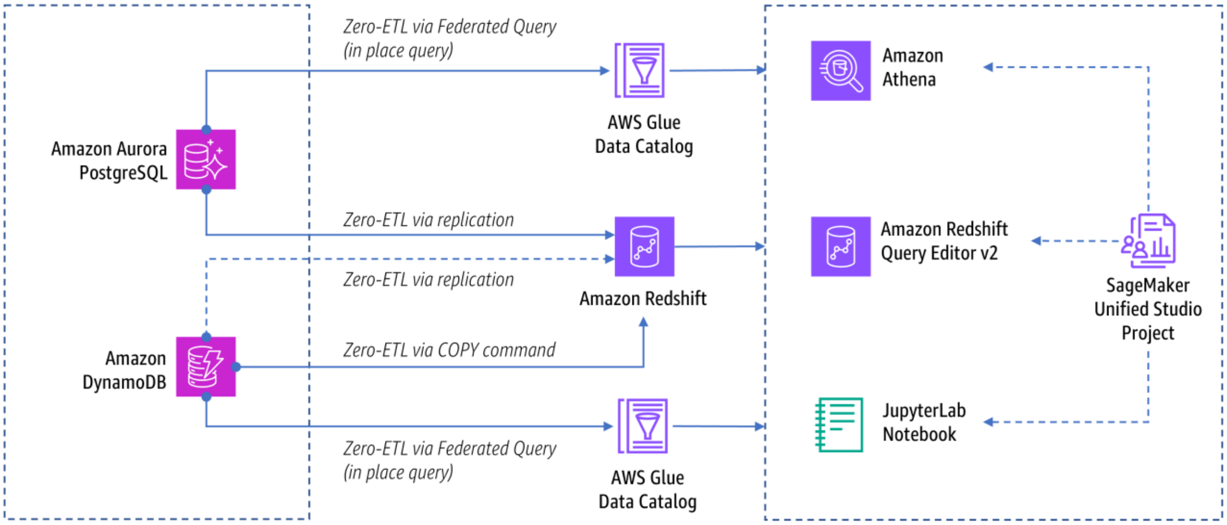


Figure 6. Zero-ETL Options

Deployment Steps

Prerequisites

Complete the following steps within AWS Management Console.

1. Create Amazon SageMaker Domain:
   - Open AWS Management console and navigate to Amazon SageMaker
   - Create a Unified Studio domain
   - Select Quick Setup and select a VPC with 3 subnets. It is recommended to *Create VPC* for Amazon SageMaker. The underlying CloudFormation template automatically provisions a properly-configured VPC with the necessary VPC endpoints.
   - Create an IAM Identity Center user with a select email and accept the invitation to activate the user account
   - Once the domain is created, copy the Amazon SageMaker Unified Studio URL: *https://{domain_id}.sagemaker.{region}.on.aws/*
2. Complete the following steps within Amazon SageMaker Unified Studio
   - Open Amazon SageMaker Unified Studio URL *https://{domain_id}.sagemaker.{region}.on.aws/* and login as a selected user
   - Create a project with the project profile: *All Capabilities*. This project profile enables Data analytics and AI/ML model and app development.
   - Once the project is ready, go to the project overview and copy Project ID and Project IAM Role ARN: *datazone_usr_role_{ProjectID}_{EnvironmentID}*

1. Collect the following inputs which will be required for deploying the CloudFormation template:
   - VPC ID where we provisioned SageMaker Domain
   - Subnet IDs
   - Security Group ID from Security Group with name datazone-{ProjectID}-dev
   - Project ID
   - Project IAM Role ARN

3. Register current user as Data Lake Administrator in the AWS Lake Formation. This will be required to manage and adjust fine-grained permissions.

Deployment

1. Deploy provided CloudFormation Templates: StackSMUSDataSources.yaml and specify parameters captured in the pre-requisite steps. If you have not created VPC as part of SageMaker Domain setup, run StackSMUSVPCEndpoints.yaml to create necessary VPC endpoints: STS, Secrets Manager, Glue, RDS Data, Redshift Data, Redshift Serverless Interface endpoints and S3 Gateway endpoint.
2. Review Output parameters

Post Deployment

1. In AWS Management Console, go to Amazon SageMaker AI → Notebooks, open the provisioned notebook and run scripts in InitDataSources.ipynb to populate demo data in Aurora PostgreSQL and DynamoDB
2. In AWS Management Console, go to Amazon SageMaker → Domains, open Domain URL, login into Amazon SageMaker Unified Studio and go to the previously created Project.
3. Open Data tab, click on '+' and then choose Add Data → Add Connection → Select connection type and specify configuration parameters where {x} are located in the CloudFormation Outputs:

Connection Type	Configuration parameters (from CloudFormation Outputs)

Amazon Aurora PostgreSQL	Name: demo-aurorapg
	Host: {AuroraPGHost}
	Port: {AuroraPGPort}
	Database: {AuroraPGDatabase}
	Authentification AWS Secrets Manager: {AuroraPGSecretArn}

Amazon Redshift Serverless	Name: demo-redshift
	Host: {RedshiftHost}
	Port: {RedshiftPort}
	Database: {RedshiftDatabase}
	Authentification AWS Secrets Manager: {RedshiftSecretArn}

Amazon DynamoDB	Name: demo-dynamodb

4. Open Compute tab and connect the provisioned compute resource on Redshift Serverless:
   - Add Compute → Connect to existing compute resources → Amazon Redshift Serverless and enter the following configuration parameters:

Compute Type	Configuration Parameters (from CloudFormation Outputs)
Amazon Redshift Serverless	Redshift compute: demo-wg Authentication – AWS Secrets Manager: {RedshiftSecretArn} Name: demo

   - On **demo-wg.redshift** compute details page, select *Actions* → *Open Query Editor* and ensure that the selected data source in the right top corner is *Redshift (demo-wg.redshift)* → *dev* → *public*
   - Run DDL + DML statements from redshift.sql to populate data in the redshift local dev database

Create Zero-ETL Integrations

Zero-ETL Integration between Redshift and Aurora PostgreSQL

Open Query Editor and select connection to the custom Redshift compute in the right top corner:

Connection: Redshift (demo-wg.redshift) | Database: dev | Schemas: public

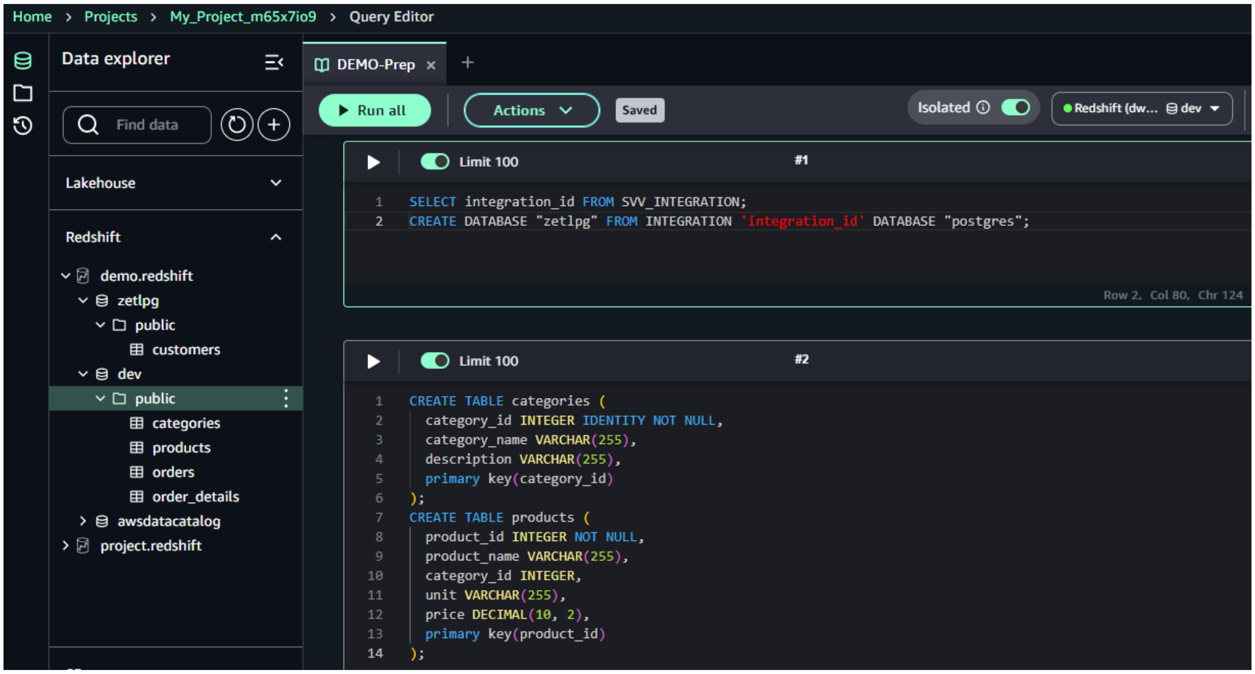
Run the following commands to create Zero-ETL database:

```
SELECT integration_id FROM SVV_INTEGRATION;
-- copy integration_id and paste into the nest statement:
CREATE DATABASE "zetlpg" FROM INTEGRATION 'integration_id' DATABASE "postgres";
```

Zero-ETL Integration between Redshift and DynamoDB

Copy Invoices data from Amazon DynamoDB into Redshift by running the following commands:

```
CREATE TABLE invoices (
            order_id integer not null,
            invoice_number varchar(200) not null,
            total integer not null,
            status varchar(10) not null,
            primary key(invoice_number)
);
COPY invoices from 'dynamodb://invoices'
IAM_ROLE default
readratio 50;
```



```
SELECT integration_id FROM SVV_INTEGRATION;
CREATE DATABASE "zetlpg" FROM INTEGRATION 'integration_id' DATABASE "postgres";
```

```
CREATE TABLE categories (
  category_id INTEGER IDENTITY NOT NULL,
  category_name VARCHAR(255),
  description VARCHAR(255),
  primary key(category_id)
);
CREATE TABLE products (
  product_id INTEGER NOT NULL,
  product_name VARCHAR(255),
  category_id INTEGER,
  unit VARCHAR(255),
  price DECIMAL(10, 2),
  primary key(product_id)
);
```

Figure 7. Initialize Zero-ETL Database and Local Database

Now let’s run queries to analyze data

SQL Analytics via Redshift Query Editor v2

Open SageMaker Unified Studio project and navigate to the Build → Query Editor. Open new Querybook tab and select Redshift connection pointing to our custom compute demo-wg.redshift:

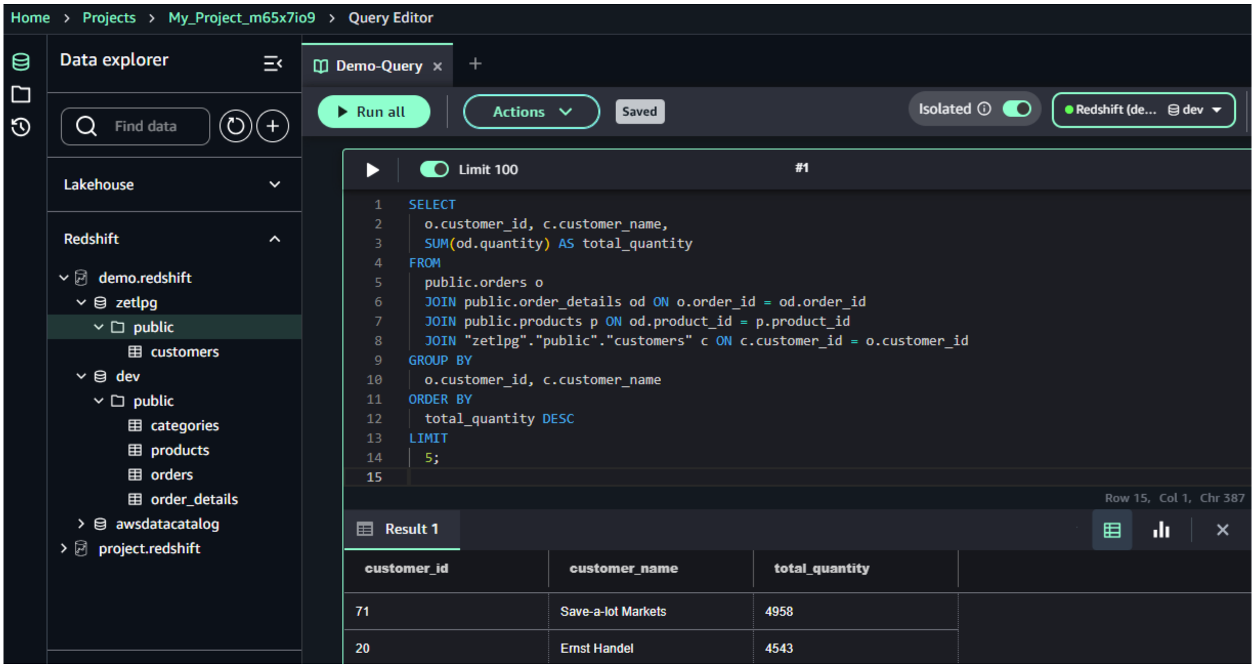
Connection: Redshift (demo-wg.redshift) | Database: dev | Schemas: public

Enter the following SQL to find an answer: *what are top 5 customers with maximum orders*.

Below SQL command joins tables from local database (public) with the customer table from Zero-ETL originated database (zetlpg) where data is being replicated from Amazon Aurora PostgreSQL database via Zero-ETL integration.

```
SELECT
  o.customer_id, c.customer_name,
  SUM(od.quantity) AS total_quantity
FROM
  public.orders o
  JOIN public.order_details od ON o.order_id = od.order_id
  JOIN public.products p ON od.product_id = p.product_id
  JOIN zetlpg.public.customers c ON c.customer_id = o.customer_id
GROUP BY
  o.customer_id, c.customer_name
ORDER BY
  total_quantity DESC
LIMIT
  5;
```

Review the results:



```
SELECT
  o.customer_id, c.customer_name,
  SUM(od.quantity) AS total_quantity
FROM
  public.orders o
  JOIN public.order_details od ON o.order_id = od.order_id
  JOIN public.products p ON od.product_id = p.product_id
  JOIN "zetlpg"."public"."customers" c ON c.customer_id = o.customer_id
GROUP BY
  o.customer_id, c.customer_name
ORDER BY
  total_quantity DESC
LIMIT
  5;
```

customer_id	customer_name	total_quantity
71	Save-a-lot Markets	4958
20	Ernst Handel	4543

Figure 8. Redshift SQL Query Execution

Generative SQL

Ensure that you selected Redshift connection pointing to our custom compute demo-wg.redshift:

Connection: Redshift (demo-wg.redshift) | Database: dev | Schemas: public

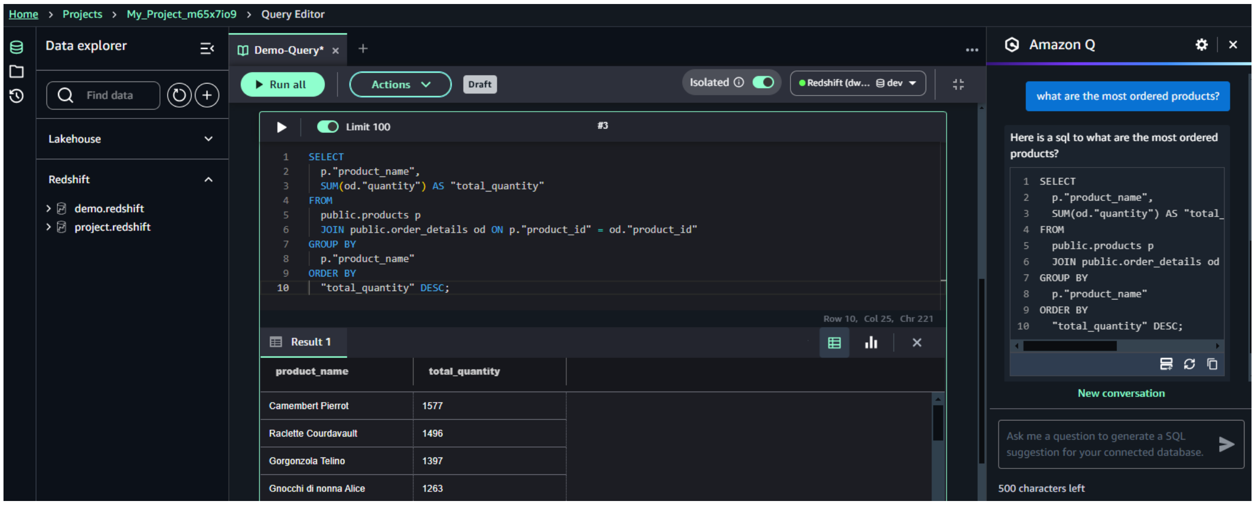
Next, open Amazon Q and type the following question:

Question: *What are the most ordered products?*

Amazon Q will generate SELECT statement similar to this one:

```
SELECT
  p."product_name",
  SUM(od."quantity") AS "total_quantity"
FROM
  public.products p
  JOIN public.order_details od ON p."product_id" = od."product_id"
GROUP BY
  p."product_name"
ORDER BY
  "total_quantity" DESC;
```

Click 'Add to querybook' and execute to confirm the results.



```
SELECT
  p."product_name",
  SUM(od."quantity") AS "total_quantity"
FROM
  public.products p
  JOIN public.order_details od ON p."product_id" = od."product_id"
GROUP BY
  p."product_name"
ORDER BY
  "total_quantity" DESC;
```

product_name	total_quantity
Camembert Pierrot	1577
Raclette Courdavault	1496
Gorgonzola Telino	1397
Gnocchi di nonna Alice	1263

what are the most ordered products?

Here is a sql to what are the most ordered products?

```
SELECT
  p."product_name",
  SUM(od."quantity") AS "total_
FROM
  public.products p
  JOIN public.order_details od
GROUP BY
  p."product_name"
ORDER BY
  "total_quantity" DESC;
```

Figure 9. SQL query generation with Amazon Q

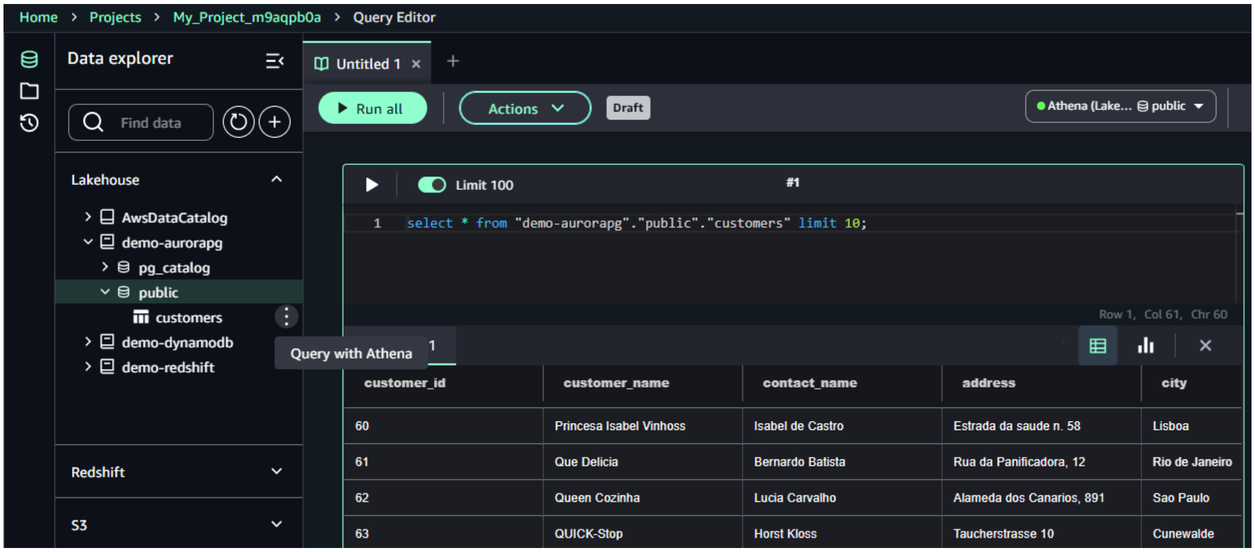
Question: *Identify all product categories associated with invoices that are currently in draft status. Include the outstanding amounts for these invoices.*

Amazon Q will generate SELECT statement similar to this one:

```
SELECT
  c."category_name",
  SUM(i."total") AS "outstanding_amount"
FROM
  "public"."categories" c
  JOIN "public"."products" p ON c."category_id" = p."category_id"
  JOIN "public"."order_details" od ON p."product_id" = od."product_id"
  JOIN "public"."invoices" i ON od."order_id" = i."order_id"
WHERE
  i."status" = 'DRAFT'
GROUP BY
  c."category_name";
```

SQL Analytics via Amazon Athena

Now let's query the federated data sources such as Amazon DynamoDB, Aurora PostgreSQL, Redshift Serverless. Once connections to the federated sources are established successfully, expand connection, select target table and click on '⋮' to query with Amazon Athena as shown on the screenshot below.



```
select * from "demo-aurorapg"."public"."customers" limit 10;
```

customer_id	customer_name	contact_name	address	city
60	Princesa Isabel Vinhoss	Isabel de Castro	Estrada da saude n. 58	Lisboa
61	Que Delicia	Bernardo Batista	Rua da Panificadora, 12	Rio de Janeiro
62	Queen Cozinha	Lucia Carvalho	Alameda dos Canarios, 891	Sao Paulo
63	QUICK-Stop	Horst Kloss	Taucherstrasse 10	Cunewalde

Figure 10. Athena SQL Query Execution

Here are simple queries to try:

```
select * from "demo-aurorapg"."public"."customers" limit 10;
select * from "demo-redshift"."public"."invoices" limit 10;
select * from "demo-dynamodb"."default"."invoices" limit 10;
```

Cleanup

Once experimentation with demo data sources is completed, delete deployed CloudFormation stack and Amazon SageMake Unified Studio domain to remove artifact and avoid incurring additional costs.

Conclusion

In this post, we explored the fundamental architecture of Amazon SageMaker Unified Studio and its essential component, Amazon SageMaker Lakehouse, highlighting its key benefits. We examined various Zero-ETL solutions that enable seamless integration with disparate data sources through a unified interface, along with specialized query editors for data analysis. We also demonstrated how Amazon Q's generative AI capabilities simplify SQL query creation, allowing analysts to focus on data exploration rather than SQL syntax. And lastly, all data operations are safeguarded by an access control framework that seamlessly integrates IAM roles, Lake Formation permissions, and resource-level access controls.

To get started with Amazon SageMaker Unified Studio and Amazon SageMaker Lakehouse, we recommend to go to AWS Management Console and create your domain by following the steps outlined in the Deployment Steps section. The provided CloudFormation template deploys demo data sources, allowing you to understand the reference configuration in Unified Studio and start exploring data analytics right away.

About the author

Natallia Bahlai is a Solutions Architect with a focus on the Financial Services industry. On most days Natallia is helping customers to craft efficient and well-designed architectures solving complex challenges. She has extensive experience working on data integrations with Salesforce. Her passion is to simplify and accelerate adoption of elegant solutions on AWS and improve businesses in meaningful ways.