

Implementation Guide

Guidance for Multi-Omics and Multi-Modal Data Integration and Analysis on AWS



Guidance for Multi-Omics and Multi-Modal Data Integration and Analysis on AWS: Implementation Guide

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

Table of Contents

Solution overview	1
Provision Amazon Omics resources to ingest, store and query genomics data	1
Provision serverless data ingestion pipelines for multi-modal data preparation and cataloging	1
Cancer Genome Atlas (TCGA) dataset ingestion	1
Ingestion of genomic datasets – 1000 Genomes Project and ClinVar	2
Visualize and explore clinical data through an interactive interface	2
Run interactive analytic queries against a multi-modal data lake	3
The Cancer Genome Atlas (TCGA) dataset	3
Genomic datasets – 1000 Genomes Project and ClinVar	3
Continuous integration and continuous delivery (CI/CD)	4
Architecture overview	6
The Cancer Genome Atlas (TCGA) dataset	7
Genomic datasets – 1000 Genomes Project and ClinVar	7
Components	10
CI/CD pipeline	10
Demonstration datasets	10
AWS Glue jobs	11
AWS Glue crawlers	11
AWS Glue workflow	12
AWS Glue data catalog	12
SageMaker AI notebook instance	12
Amazon S3 buckets	12
QuickSight dataset	12
Amazon Omics	13
Cost	14
Security	16
IAM roles	16
Design considerations	17
Regional deployment	17
AWS CloudFormation template	18
Automated deployment	19
Prerequisites	19
Launch the stack	19

Post-deployment tasks	22
Post-deployment overview	22
Step 1. Confirm crawler completion	22
Step 2. Confirm Omics resource creation	23
Step 3. Launch the QuickSight stack	23
Step 4. Explore the data using QuickSight	24
Step 5. Make Amazon Omics data available in Athena	26
Step 6. Query data in the data lake	27
Option 1: Use the provided Jupyter notebooks	27
Option 2: Run the drug response report query in the Amazon Athena console	28
Additional resources	30
Open-source licenses	31
Roles and permissions	32
Code deployment pipeline permissions	32
CloudFormation role	32
CodeBuild role	36
Source event role	38
AWS Glue and Amazon SageMaker AI notebook permissions	38
Job role	39
Glue job role	40
Runbook role	41
Uninstall this guidance	46
Using the AWS Management console	46
Using AWS CLI	46
Uninstall manually	46
Run the TCGA Glue workflow	49
Collection of operational metrics	50
Source code	51
Revisions	52
Contributors	53
Notices	54
AWS Glossary	55

Prepare genomic, clinical, mutation, expression, and imaging data for large-scale analysis and perform interactive queries against a data lake

Publication date: *July 2020* (*[last update: January 2023](#)*)

This guidance creates a scalable environment in AWS to prepare genomic, clinical, mutation, expression and imaging data for large-scale analysis and perform interactive queries against a data lake. This guidance demonstrates how to:

1. Provision [Amazon Omics](#) resources to ingest, store and query genomics data.
2. Provision serverless data ingestion pipelines for multi-modal data preparation and cataloging.
3. Visualize and explore clinical data through an interactive interface.
4. Run interactive analytic queries against a multi-modal data lake using [Amazon Athena](#) and an [Amazon SageMaker AI notebook instance](#).

1. Provision Amazon Omics resources to ingest, store, and query genomics data

This guidance uses [AWS CodeBuild](#), [AWS CodePipeline](#) and [AWS CloudFormation](#) to build, package, and deploy Amazon Omics resources: Reference store, Variant store and Annotation store. And Variant Call Files (VCFs). These resources are ingested and stored in a query ready format in the Variant and Annotation stores.

2. Provision serverless data ingestion pipelines for multi-modal data preparation and cataloging

The Cancer Genome Atlas (TCGA) dataset ingestion

For multi-modal data ingestion, the guidance retrieves public data from [The Cancer Genome Atlas \(TCGA\)](#) and [The Cancer Imaging Archive \(TCIA\)](#) using a set of [AWS Glue](#) jobs. The retrieved data sets are parsed, filtered, and stored in the data lake bucket in Parquet format. The guidance also

provides several AWS Glue crawlers, which catalog and infer the schema of the downloaded data sets.

Once all the detailed data sets have been retrieved, another AWS Glue job invokes [Amazon Athena](#) to summarize the data sets, store the results in Parquet format, and register the results as a Glue data catalog table in a single query operation. An AWS Glue workflow is provided to coordinate and sequence the multiple Glue jobs and crawlers created by the guidance.

The TCGA workflow [AWS Glue](#) workflow is provided to prepare the data. It invokes and sequences the AWS Glue jobs that extract, translate, and load (ETL) data from TCG and TCIA into Amazon S3; the AWS Glue crawlers, which register the data sets in the AWS Glue data catalog; and the final AWS Glue job, which creates a summary of the data sets through invoking an Amazon Athena query.

Ingestion of genomic datasets – 1000 Genomes Project and ClinVar

During setup, annotation data from [ClinVar](#) (in VCF format), an example VCF file, and a subset of the [1000 Genomes Project](#) dataset (in VCF format) are copied into the data lake bucket. An Amazon Omics reference store and variant store are configured. The example VCF along with the 1000 Genomes subset VCF are ingested into the variant store and made available for query. In addition, the ClinVar VCF is ingested into the Amazon Omics annotation store and made available for query similar to data in the variant store. A separate Athena database is configured to provide access to the variant and annotation store data using Athena.

3. Visualize and explore clinical data through an interactive interface

An [Quick](#) dataset is provisioned to provide users an interactive, drag-and-drop interface to explore clinical data. The dataset retrieves data from [Amazon Athena](#), joining the clinical table with the summary table to facilitate visualization of data availability and interactive cohort generation. The guidance includes detailed instructions for building your own visual queries on the data, and guidance on sharing the resulting analysis with other users.

4. Run interactive analytic queries against a multi-modal data lake

Note

PyAthena is a Python [DB API 2.0 \(PEP 249\)](#) compliant client for [Amazon Athena](#). PyDICOM is a Python library which can be used to read, modify, and write DICOM image files.

An [Amazon SageMaker AI](#) notebook instance is provisioned with several example Jupyter notebooks that demonstrates how to work with data in a multi-modal data lake.

The Cancer Genome Atlas (TCGA) dataset

The TCGA notebook uses Amazon Athena to construct a cohort of lung cancer patients from TCGA based on clinical and tumor mutation data to analyze signals in gene expression data. A subset of the identified patients having image data are retrieved and visualized within the notebook. This is done using a sequence of queries submitted by the PyAthena driver and image data retrieved and parsed using PyDICOM.

The 1000 Genomes Project dataset

The 1000 Genomes Project notebook uses Amazon Athena to identify genomic variants related to drug response for a given cohort of individuals. The below query is run against data in the data lake using the PyAthena driver to:

1. Filter by samples in a subpopulation.
2. Aggregate variant frequencies for the subpopulation-of-interest.
3. Join on the ClinVar dataset.
4. Filter by variants that have been implicated in drug-response.
5. Order by highest frequency variants. The query can also be run in the Amazon Athena console.

```
SELECT  count(*)/cast(numsamples AS DOUBLE) AS genotypefrequency
        ,cv.attributes['RS'] as rs_id
        ,cv.attributes['CLNDN'] as clinvar_disease_name
        ,cv.attributes['CLNSIG'] as clinical_significance
```

```
,sv.contigname
,sv.start
,sv."end"
,sv.referenceallele
,sv.alternatealleles
,sv.calls
FROM {variant_table_name} sv
CROSS JOIN
  (SELECT count(1) AS numsamples
   FROM
     (SELECT DISTINCT vs.sampleid
      FROM {variant_table_name} vs
      WHERE vs.sampleid LIKE 'NA12%'))
JOIN {annotation_table_name} cv
ON sv.contigname = cv.contigname
   AND sv.start = cv.start
   AND sv."end" = cv."end"
   AND sv.referenceallele = cv.referenceallele
   AND sv.alternatealleles = cv.alternatealleles
   AND cv.attributes['CLNSIG'] LIKE '%response%'
   AND sv.sampleid LIKE 'NA12%'
GROUP BY sv.contigname
         ,sv.start
         ,sv."end"
         ,sv.referenceallele
         ,sv.alternatealleles
         ,sv.calls
         ,cv.attributes['RS']
         ,cv.attributes['CLNDN']
         ,cv.attributes['CLNSIG']
         ,numsamples
ORDER BY genotypefrequency DESC LIMIT 50
```

Continuous integration and continuous delivery (CI/CD)

The guidance includes [continuous integration](#) and [continuous delivery](#) (CI/CD) using [AWS CodeCommit](#) source code repositories and [AWS CodePipeline](#) for building and deploying updates to the data preparation jobs, crawlers, data analysis notebooks, and the data lake infrastructure. This guidance fully leverages [infrastructure as code](#) principles and best practices that allow you to rapidly evolve the guidance. After deployment, you can modify the guidance to fit your particular

needs, for example, by adding new data preparation jobs and crawlers. Each change is tracked by the CI/CD pipeline, facilitating change control management, rollbacks, and auditing.

This implementation guide describes architectural considerations and configuration steps for deploying the Guidance for Multi-Omics and Multi-Modal Data Integration and Analysis on AWS in the Amazon Web Services (AWS) Cloud. It includes links to an [AWS CloudFormation](#) template that launches and configures the AWS services required to deploy this guidance using AWS best practices for security and availability.

The guide is intended for IT infrastructure architects, administrators, data scientists, software engineers, and DevOps professionals who have practical experience architecting in the AWS Cloud.

Architecture overview

The following diagram describes the overall data lake architecture; how data is ingested, curated, cataloged, and queried.

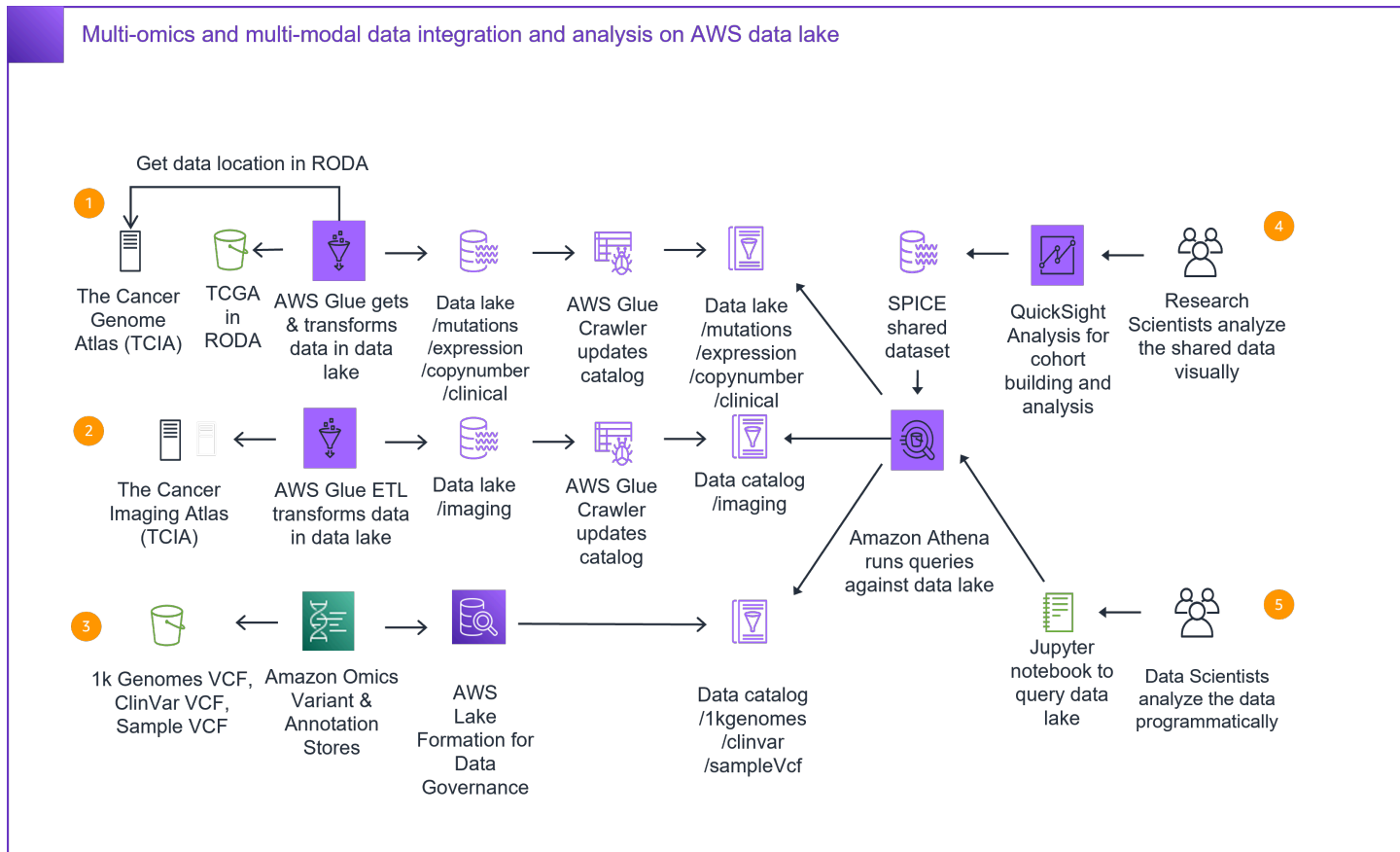


Figure 1: Guidance for Multi-Omics and Multi-Modal Data Integration and Analysis on AWS data lake architecture

This guidance demonstrates how to ingest common multi-omics data sets into a centralized data lake and work with that data using Amazon Athena and Jupyter notebooks. There are example ingestion pipelines for clinical, mutation, gene expression, and copy number data (TCGA), and imaging metadata (TCIA). An Amazon Omics Reference Store, Variant Store and Annotation Store are also created for genomic variant calls data (1000 Genomes), annotation data (ClinVar) and an example individual Variant Call File (VCF) data.

The Cancer Genome Atlas (TCGA) dataset

A set of data from two specific projects (Lung Adenocarcinoma, LUAD and Lung Squamous Cell Carcinoma, LUSC) is retrieved from The Cancer Genome Atlas (TCGA) during setup. The dataset, originally in raw formats direct from TCGA, is parsed and stored in Apache Parquet format and partitioned by project ID. The `tcga-clin`, `tcga-cnv`, `tcga-exp`, and `tcga-mut` crawlers are provided to crawl the datasets, infer the data schemas, and add tables to the guidance AWS Glue data catalog.

Imaging metadata from the same two projects (TCGA-LUAD and TCGA-LUSC) is retrieved from The Cancer Imaging Archive (TCIA) during deployment. The dataset is parsed and stored in Apache Parquet format and partitioned by project ID. The `tcga-img` crawler is provided to crawl the dataset, infer the data schemas, and add tables to the guidance AWS Glue data catalog.

A summary table stores counts of the number of TCGA and TCIA data records available for each patient in the dataset. This summary table is computed via an AWS Glue job, which invokes an Amazon Athena query, saving the results in Apache Parquet format. The `tcga-sum` crawler is provided to crawl the dataset, infer the data schemas, and add tables to the guidance AWS Glue data catalog.

Genomics datasets – 1000 Genomes Project and ClinVar

A portion of the 1000 Genomes Project dataset (Chromosome 22) is copied into the data lake bucket during setup. An automated step is initiated to ingest this Variant Call File (VCF) into the pre-created Amazon Omics Variant store.

A ClinVar VCF is copied into the data lake bucket during deployment. An automated step is initiated to ingest this VCF into the pre-created Amazon Omics Annotation store.

An example VCF is copied into the data lake bucket during setup. An automated step is initiated to ingest this VCF into the pre-created Amazon Omics Variant store.

The creation of Amazon Omics Variant and Annotation stores automatically shares these stores as [AWS Lake Formation](#) tables. With a few steps run by a Data lake Administrator, this data is made available in the AWS Glue data catalog for query using Athena.

Deploying this guidance with the default parameters builds the following environment in the AWS Cloud.

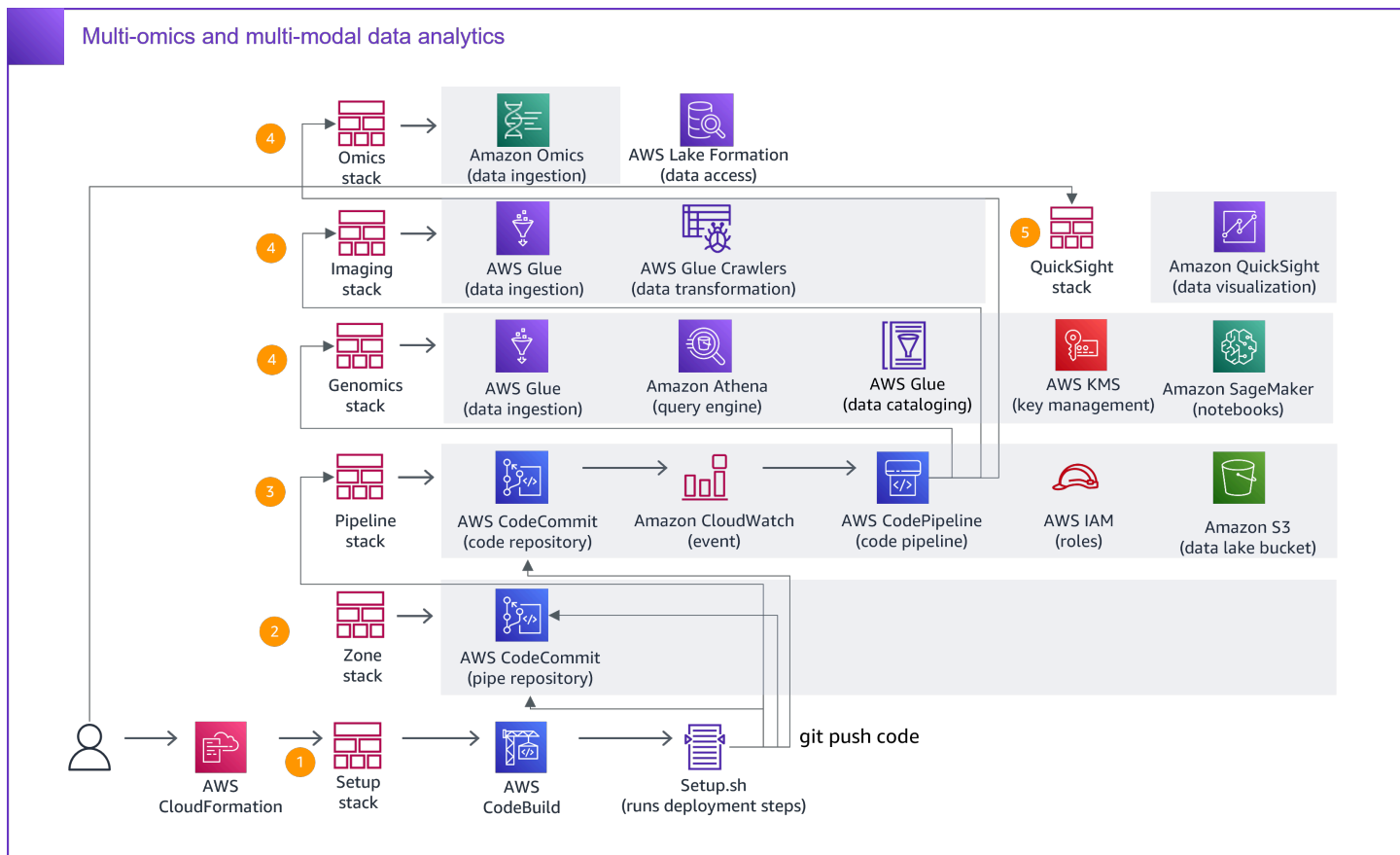


Figure 2: AWS Cloud environment built after deploying this guidance with default parameters

The AWS CloudFormation template creates six CloudFormation stacks in your AWS account including a setup stack to install the guidance. The other stacks include a landing zone (zone) stack containing the common resources and artifacts, a deployment pipeline (pipe) stack defining the guidance's CI/CD pipeline, and three codebase (genomics, imaging, and omics) stacks providing the ETL scripts, jobs, crawlers, Omics resources, a data catalog, and notebook resources. The installation also includes a seventh CloudFormation stack that can be launched via a quick start link to set up the QuickSight resources (quicksight).

1. The setup stack creates an [AWS CodeBuild](#) project containing the setup.sh script. This script creates the remaining CloudFormation stacks and provides the source code for both the AWS CodeCommit pipe repository and the code repository.
2. The landing zone (zone) stack creates the CodeCommit pipe repository. After the landing zone (zone) stack completes its setup, the setup.sh script pushes source code to the CodeCommit pipe repository.
3. The deployment pipeline (pipe) stack creates the CodeCommit code repository, an [Amazon CloudWatch](#) event, and the CodePipeline code pipeline. After the deployment pipeline (pipe)

stack completes its setup, the `setup.sh` script pushes source code to the CodeCommit code repository.

4. The CodePipeline (code) pipeline deploys the codebase (genomics, imaging, and omics) CloudFormation stacks. After the AWS CodePipeline pipelines complete their setup, the resources deployed in your account include [Amazon Simple Storage Service](#) (Amazon S3) buckets for storing object access logs, build artifacts, and data in your data lake; CodeCommit repositories for source code; an AWS CodeBuild project for building code artifacts (for example, third-party libraries used for data processing); an AWS CodePipeline pipeline for automating builds and deployment of resources; example AWS Glue jobs, crawlers, and a data catalog; and an Amazon SageMaker AI Jupyter notebook instance. An Amazon Omics Reference Store, Variant Store and Annotation store is provisioned and an example VCF, 1000 Genomes subset VCF and ClinVar annotation VCF are loaded into the solution for analysis using Amazon Athena.
5. The imaging stack creates a hyperlink to a CloudFormation quick start, which can be launched to deploy the QuickSight (quicksight) stack. The QuickSight stack creates IAM and QuickSight resources necessary to interactively explore the multi-omics dataset.

The example code includes the resources needed to prepare data for large-scale analysis and perform interactive queries against a multi-omics data lake.

Components

CI/CD pipeline

A complete continuous integration and continuous deployment (CI/CD) pipeline is created when you launch the guidance. This pipeline is built using AWS CodeCommit as the source code, AWS CodeBuild projects to build the guidance's artifacts (for example, `hail.jar`), and an AWS CodePipeline pipeline to run a build project and automate deployment (using AWS CloudFormation) after the updated source code is published.

The AWS resources that compose the CI/CD pipeline are defined in the `Pipe/template_cfn.yml` file. Changes to the guidance requiring new artifacts to be built require an update to the CI/CD pipeline definition.

Demonstration datasets

This guidance copies the following datasets into your data lake bucket.

- **The Cancer Genome Atlas, Lung Adenocarcinoma and Lung Squamous Cell Carcinoma (TCGA-LUAD and TCGA-LUSC)** – Two public cancer studies consisting of clinical, mutation, gene expression, and copy number data, partitioned by study ID and in Apache Parquet format. These studies are retrieved directly from The Cancer Genome Atlas (TCGA) APIs via a set of AWS Glue jobs. These datasets are used as the basis of cohort exploration and selection.
- **The Cancer Imaging Archive, Lung Adenocarcinoma and Lung Squamous Cell Carcinoma (TCIA)** – Imaging metadata for subjects in the TCGA studies listed above, partitioned by study ID and in Apache Parquet format. This metadata is retrieved directly from The Cancer Imaging Archive (TCIA) APIs via an AWS Glue job. This dataset can be used to filter patients by availability of imaging data, as well as retrieve individual images for high-level exploration.
- **1000 Genomes, Chromosome 22 (genome)** – A portion of the 1000 genomes public dataset of human genomic variant data in VCF format. This dataset is used as our cohort dataset for creating the drug response report.
- **ClinVar (annotation)** – The public dataset that aggregates information about genomic variation and its relationship to human health. We use the VCF format to be used with Amazon Omics Variant Store.
- **Individual Sample Variants (sample)** – An individual sample Variant Call File (VCF) dataset used to demonstrate queryability with Athena.

AWS Glue jobs

This guidance creates the following AWS Glue jobs:

- **Clinical, Cnv, Expression, and Mutation** – Retrieves data from TCGA, filters and transforms it to Apache Parquet format, and writes the resulting files to the data lake. For each of the four data types, there are two jobs, one for each of the two source TCGA projects (LUAD and LUSC). For a total of 8 jobs.
- **ImagingMetadata** – Retrieves imaging metadata from The Cancer Imaging Archive (TCIA), transforms them to Apache Parquet format, and writes the resulting files to the data lake. There is one job for each of the two source TCGA projects (LUAD and LUSC).
- **TcgaSummary** – Invokes Amazon Athena to generate a new database table containing summary metrics over all of the TCGA and TCIA data tables, saving the results in Apache Parquet format within the data lake, and registering the table with the Glue data catalog.

AWS Glue crawlers

This guidance creates the following AWS Glue crawlers:

- **tcga-clin** – Creates/Updates the `clinical_patient` and other clinical tables in the AWS Glue data catalog to reflect the data schema of the TCGA clinical data in the data lake.
- **tcga-cnv** - Creates/Updates the `tcga_cnv` table in the AWS Glue data catalog to reflect the data schema of the TCGA copy number data in the data lake.
- **tcga-exp** – Creates/Updates the `expression_tcga_luad` and `expression_tcga_lusc` tables in the AWS Glue data catalog to reflect the data schema of the TCGA expression data in the data lake.
- **tcga-mut** – Creates/Updates the `tcga_mutation` table in the AWS Glue data catalog to reflect the data schema of the TCGA mutation data in the data lake.
- **tcga-img** – Creates/Updates the `tcia_patients` and `tcia_image_series` tables in the AWS Glue data catalog to reflect the data schema of the TCIA image metadata in the data lake.
- **tcga-sum** – Creates/Updates the `tcga_summary` table in the AWS Glue data catalog to reflect the data schema of the TCGA summary data in the data lake.

AWS Glue workflow

This guidance creates an AWS Glue workflow which sequences and coordinates the AWS Glue jobs and crawlers as part of the TCGA and TCIA data sets. For each TCGA data type and for the TCIA data, two Glue jobs are invoked, followed by a trigger that signals the corresponding Glue crawler to run. Once all Glue crawlers are complete, the `TcgaSummary` job is invoked to create the summary table.

AWS Glue data catalog

This guidance creates an AWS Glue data catalog with a `genomicsanalysis` database that contains all the tables used by this guidance. AWS Glue is configured to encrypt the metadata stored in the data catalog, data files stored in Amazon S3 buckets, and all logs stored in Amazon CloudWatch.

SageMaker AI notebook instance

This guidance creates an Amazon SageMaker AI notebook instance that demonstrates how to use AWS Glue and Amazon Athena to identify variants related to drug response.

Amazon S3 buckets

This guidance creates the following Amazon S3 buckets. Each bucket has encryption and logging activated:

- **Data Lake Bucket** – Stores genomic variant and ClinVar variant annotation data.
- **Resources Bucket** – Stores notebooks and shell scripts.
- **Build Bucket** – Stores build artifacts deployed through the pipeline.
- **Logs Bucket** – Stores S3 access logs related to the three buckets listed above.

QuickSight dataset

This solution creates an Quick dataset derived from the Amazon Athena `clinical_patient` and `tcga_summary` tables, joined by the patient ID field, and filtered for key columns. The dataset is shared with the guidance owner and available for creating interactive analyses on clinical data.

Before using QuickSight to explore the data in this guidance, sign up for an Quick subscription. For details, refer to [Signing up for an Quick subscription](#) in the *Quick User Guide*.

Amazon Omics

This solution creates the following Amazon Omics resources:

- **Reference Store** – A data store for the storage of reference genomes in FASTA format.
- **Variation Store** – A data store that stores variant data at a population scale in VCF format.
- **Annotation Store** – A data store that stores annotation data in VCF, GFF3 and TSV/CSV formats downstream queryability.
- In addition, the example datasets (1000 Genomes, example VCF and ClinVar VCF) are ingested into these stores as part of the solution setup.

Cost

You are responsible for the cost of the AWS services used while running this reference deployment. As of the date of publication, the cost for running this guidance with default settings in the US East (N. Virginia) Region is approximately **\$61.00 per month**.

AWS service	Details	Cost per month [\$ USD]
AWS Glue	\$2.80 during setup to run the Glue jobs and crawlers	One-time cost (\$2.80)
AWS Glue	<p>AWS Glue job runs per job: 10 DPUs * 3/60 hour at \$0.44 per DPU-Hour or \$0.22 per job</p> <p>AWS Glue crawler runs per crawler: 1 DPUs * 1/6 hour at \$0.44 per DPU-Hour or \$0.0748 per crawler</p> <p>Athena query runs: Less than 0.0005 TB scanned * 0.0005 * \$5/TB = \$0.0025</p>	One-time cost (\$0.22 per job, \$0.0748 per crawler, \$0.0025 per TB)
AWS KMS	One key to encrypt the Glue data catalog	\$1.00
Amazon SageMaker AI	Notebook Instance	\$36.00 (\$0.05 per hour)
Amazon S3	Data lake data storage	\$0.06
Amazon Athena	\$0.00025 for each Athena query run for interpreting the data	Variable (\$0.00025 per query)

AWS service	Details	Cost per month [\$ USD]
Quick	Standard pricing per author	\$24.00*
Amazon Omics	One VCF is within Free Tier pricing	\$0.00
Total:		\$61.00

* Quick offers a free 30-day trial, after which the standard pricing per analysis author is \$24 per month.

We recommend creating a [budget](#) through [AWS Cost Explorer](#) to help manage costs. Prices are subject to change. For full details, refer to the pricing webpage for each AWS service used in this solution.

If you customize the guidance to analyze your genomics dataset, the cost factors include the storage size of the data being analyzed, the number of Extract Transform and Load (ETL) jobs and crawlers being used, compute resources required for each job, number of notebook instances provisioned and volume of data scanned when using Athena. For a more accurate estimate of cost, we recommend working with a sample dataset of your choosing as a benchmark.

Security

This guidance is preconfigured with all of the IAM policies and roles necessary to run the guidance with least privileges.

When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS. This shared model can reduce your operational burden as AWS operates, manages, and controls the components from the host operating system and virtualization layer down to the physical security of the facilities in which the services operate. For more information about security on AWS, visit the [AWS Cloud Security](#).

IAM roles

AWS Identity and Access Management (IAM) roles allow you to secure jobs and crawlers running in AWS Glue, and restrict access to the data catalog, the data lake bucket, and the notebook instance. All of the IAM roles in this guidance have been defined with least privileges. For details about roles and permissions used in this guidance, refer to [Roles and permissions](#).

Design considerations

This guidance fully leverages infrastructure as code principles and best practices that allow you to rapidly evolve the guidance. Storing your Extract Transform and Load (ETL) job, crawler, and data lake definitions as code makes them easier to share, inspect for compliance, and reproduce. Additionally, each change you make is tracked by the CI/CD pipeline, facilitating change control management, rollbacks, and auditing.

Regional deployment

This guidance uses the AWS CodePipeline service, which is currently available in specific AWS Regions only. Therefore, you must launch this guidance in an AWS Region where this service is available. For the most current service availability by AWS Region, refer to the [AWS Regional Services List](#). The guidance has been tested in all Regions.

AWS CloudFormation template

This guidance uses AWS CloudFormation to automate the deployment of the Guidance for Multi-Omics and Multi-Modal Data Integration and Analysis on AWS in the AWS Cloud. It includes the following AWS CloudFormation template, which you can download before deployment.

[View template](#)

guidance-for-multi-omics-and-multi-modal-data-integration-and-analysis-on-aws.template:

Use this template to launch this guidance and all associated components. The default configuration deploys an AWS CodePipeline deployment pipeline, an AWS CodeCommit repository for the pipeline code, an AWS CodeCommit repository for the guidance code, Amazon S3 buckets, Amazon Omics Reference, Variant and Annotation stores, AWS Glue jobs, crawlers, workflow, and a data catalog, AWS Identity and Access Management (IAM) roles and policies, an AWS Key Management Service (KMS) key, and an Amazon SageMaker AI notebook instance. You can also customize the template based on your specific needs.

Automated deployment

Before you launch the automated deployment, review the architecture, configuration, network security, and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy the guidance into your account.

Time to deploy: Approximately 60 minutes.

Prerequisites

Before deploying this guidance, verify that you have an administrator role in [AWS Identity and Access Management \(IAM\)](#) in your AWS account. Refer to [Roles and permissions](#) for details on permissions used by this guidance. For information about setting up an administrator user, refer to [Create an administrative user](#) in the *AWS Identity and Access Management User Guide*.

Also, verify that the Data Catalog settings allow for access control using IAM by navigating to the AWS Lakeformation Console. In the navigation pane on the left, go to **Data catalog – Settings** and make sure that the options **Use only IAM access for new databases** and **Use only IAM access for new tables in new databases** options are selected under **Default permissions for newly created databases and tables**.

Before you explore the data in this guidance using [Quick](#), you must first sign up for an Quick subscription. For more information about signing up for Quick, refer to [Signing up for an Quick subscription](#) in the *Quick User Guide*.

Launch the stack

Important

This guidance includes an option to send anonymous operational metrics to AWS. We use this data to better understand how customers use this guidance and related services and products. AWS owns the data gathered through this survey. Data collection is subject to the [AWS Privacy Policy](#).

To opt out of this feature, download the template, modify the AWS CloudFormation mapping section, and then use the AWS CloudFormation console to upload your template and deploy the guidance. For more information, refer to the [Collection of operational metrics](#) section of this guide.

This automated AWS CloudFormation template deploys the guidance in the AWS Cloud. You must have the appropriate IAM permissions before launching the stack.

Note

You are responsible for the cost of the AWS services used while running this guidance. Refer to the [Cost](#) section for more details. For full details, see the pricing webpage for each AWS service used in this guidance.

1. Sign in to the AWS Management Console and select the button to launch the guidance-for-multi-omics-and-multi-modal-data-integration-and-analysis-on-aws.template AWS CloudFormation template.

Launch solution

You can also [download the template](#) as a starting point for your own implementation.

2. The template launches in the US East (N. Virginia) Region by default. To launch this guidance in a different AWS Region, use the Region selector in the console navigation bar.

Note

This guidance uses the AWS CodePipeline and Amazon Omics services, which are currently available in specific AWS Regions only. Therefore, you must launch this guidance in an AWS Region where these services are available. For the most current service availability by AWS Region, refer to the [AWS Regional Services List](#).

3. On the **Create stack** page, verify that the correct template URL shows in the **Amazon S3 URL** text box and choose **Next**.
4. On the **Specify stack details** page, assign a name to your stack and provide a project name for the guidance installation. For information about naming character limitations, refer to [IAM and STS Limits](#) in the *AWS Identity and Access Management User Guide*.
5. Choose **Next**.
6. On the **Configure stack options** page, choose **Next**.

7. On the **Review** page, review and confirm the settings. Check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.
8. Choose **Create stack** to deploy the stack.
9. You can view the status of the stack in the AWS CloudFormation Console in the **Status** column. You should see a status of CREATE_COMPLETE in approximately 60 minutes.

Post-deployment tasks

After the stack has successfully deployed, complete these post-deployment tasks.

Post-deployment overview

Use the following steps to deploy this guidance on AWS. For detailed instructions, follow the links for each step.

[Step 1. Confirm crawler completion](#)

[Step 2. Confirm Omics resource creation](#)

[Step 3. Launch the QuickSight resource stack](#)

[Step 4. Explore the data using QuickSight](#)

[Step 5. Make Amazon Omics data available in Athena](#)

[Step 6. Query data in the data lake](#)

Step 1. Confirm crawler completion

Note

The Parquet version of the datasets are copied into the data lake bucket after the guidance is set up so that running these AWS Glue jobs is optional. We run the AWS Glue crawlers on these datasets automatically during setup of the guidance to create the tables in the data catalog. The AWS Glue jobs are provided for modification and reference.

The AWS Glue crawlers are launched after deployment completes. You must confirm that all crawlers are complete before proceeding with further steps to explore the data using QuickSight or Jupyter.

Use the following steps to verify the crawler status:

1. Sign in to the [AWS Glue console](#).

2. Select **Crawlers** on the left navigation pane.
3. Wait for all crawlers to show status as **READY**, and the **Tables added** column to each be **1 or larger**.

Step 2. Confirm Omics resource creation

The Amazon Omics Reference, Variant, and Annotation stores are created. 1000 Genomes, the example VCF, and ClinVar Annotation data are loaded into the stores once the solution is deployed.

Follow these steps to verify resource creation:

1. Sign into the [Amazon Omics console](#)
2. Select **Storage > Reference store** in the left navigation page.
3. Verify the presence of **hg38** under **Reference Genomes** with Import status set to **Active**.
4. Select **Analytics > Variant stores** in the left navigation page.
5. Verify that Variant store with name **omicsvariantstore** has status set to **Active**.
6. Choose **omicsvariantstore** and verify that there are two **Job IDs** with **Import** status set to **Completed**. These correspond to the import of the 1000 Genomes VCF and the example VCF.
7. Select **Analytics > Annotation stores** in the left navigation page.
8. Verify that Annotation store with the name **omicsannotationstore** has status set to **Active**.
9. Choose **omicsannotationstore** and verify that there is a **Job ID** with **Import** status set to **Completed**. This corresponds to the import of the ClinVar VCF.

Step 3. Launch the QuickSight stack

Note

To create Quick resources, the AWS CloudFormation stack requires the user to input their QuickSight User ID. This User ID can be retrieved from Quick through the following steps:

1. Sign in to Quick.
2. Select the dropdown menu in the upper right of the display.
3. Choose the menu option containing your username.
4. In the resulting **Account info** screen, copy the entire text under the **Username** field.

To assist the user in exploring the use of Quick to analyze multi-omics data, the guidance provides a CloudFormation template that automatically creates the necessary QuickSight data source and dataset, as well as adds a policy to the QuickSight service IAM role to grant it permissions to access the relevant data resources.

Use the following steps to create Quick resources for this guidance:

1. Sign in to the [AWS CloudFormation console](#).
2. Select **Stacks** in the left navigation pane, then select the **Imaging** stack.
3. Select the **Outputs** tab, then select the link shown in the **CreateQuicksightLink** output. This opens the CloudFormation **Create Stack** wizard with the stack name and project name pre-filled.
 - a. In the **QuicksightUserId** parameter field, paste in your username information from QuickSight.
 - b. If during the setup process of Quick, you selected an alternate IAM role for the QuickSight service, you must update the **QuicksightServiceRoleName** with the correct name of the IAM role being used by QuickSight.
 - c. Check the box to acknowledge that the template will create AWS Identity and Access Management (IAM) resources.
4. Choose **Create stack** to deploy the stack.
5. You can view the status of the stack in the AWS CloudFormation Console in the **Status** column. You should see a status of CREATE_COMPLETE in approximately 1 minute.

Step 4. Explore the data using QuickSight

Once the QuickSight template has completed, you can create a QuickSight Analysis to explore the data graphically with interactive filters. The steps below are recommended; however, feel free to follow your own path when exploring this data.

1. Open the QuickSight console from the AWS Management Console by searching for **QuickSight**.
2. In the **Datasets** tab, locate the new dataset with your selected project name and choose it to open.
3. Choose **Create analysis**.
4. Your analysis will begin with a pre-selected AutoGraph visual. Change this to a **Horizontal Bar Chart** by selecting the appropriate icon in the **Visual types** panel.

5. From the **Fields list**, drag the **age_at_dx_numeric** field to the **Y axis** field well at the top of the screen. Your visual will now show **Count of Records by Age_at_dx_numeric**.
6. The visual by default will zoom into the data, which you can navigate using the vertical scroll bar in the visual tile.
7. To view the entire data set, with the visual selected, choose the **Format visual** gear icon in the small context menu that appears to the upper right of the visual. Then, expand the **Y-axis** settings and uncheck **Show data zoom**.
8. Next, we will add a second visual to look at an additional field.
9. Choose the **+ Add** button in the upper-left and select **Add visual**.
10. From the **Visual types** panel, choose the **Heat map** visual type.
11. Drag the **egfr_mutation_status** field from the **Fields list** to the **Rows** field well, and drag the **eml4_alk_translocation_status** field to the **Columns** field well.
12. The visual now shows a heat map of the number of patients who have had their tumors tested for EML4-ALK translocation and EGFR mutation status. Because there are a substantial number of patients without test results, let's add some filters to the data set to clean this up.
13. Select the **Filter tab** and ensure that you have the newly created heat map visual selected.
14. Choose the **+ button** in the **Filters** section to add a new filter and select **eml4_alk_translocation_status** as the field to filter.
15. Choose the newly created filter and change the following settings:
 - a. Change the **Only this visual** dropdown to **All visuals of this dataset**.
 - b. In the **Include** list, unselect the **[Not Available]** and **[Unknown]** options.
16. Choose **Apply** to see the effect of the new filter, and then choose **Close**.
17. Repeat steps 11 through 13, but this time select the **egfr_mutation_status** field.
18. Next, create a pivot table that shows the number of data records available for each category of patients in our filtered cohort.
19. Choose the **+ Add** button in the upper-left and select **Add visual**.
20. From the **Visual types** panel, choose the **Pivot table** visual type.
21. Drag both the **egfr_mutation_status** and **eml4_alk_translocation_status** fields from the **Fields list** to the **Columns** fields well.
22. In the **Values** fields well, change it to "Row" mode by selecting **Row**, then drag all of the **num_..._records** fields into the **Values** field well.

- 23 For each entry in the **Values** field well, change the **Aggregate** function from **Sum** to **Count** by choosing the dropdown arrow, then selecting **Count** from the **Aggregate** menu.
- 24 Finally, create a table listing the patient IDs that are in the current analysis. This can be exported as a CSV file to create a cohort for further analysis.
- 25 Choose the **+ Add** button in the upper-left and select **Add visual**.
- 26 From the **Visual types** panel, choose the **Table** visual type.
- 27 Drag the **bcr_patient_barcode** field into the **Value** field well. You can add additional fields such as those we have used in the filters or other visuals.
- 28 To export the list of patients as a CSV file, in the small context menu that appears to the upper right of the visual, choose the **Menu options (...)** icon, then select **Export to CSV**.

Step 5. Make Amazon Omics data available in Athena and SageMaker Notebook

Once the Omics resources are created and verified, you can make the data in the Omics Variant Store and Omics Annotation Store available in Athena for querying through AWS Lake Formation.

1. Navigate to AWS Lake Formation, you should see the variant store and annotation store resource shares listed under **Data catalog > Tables** with the names for the Omics stores. In this case, **omicsvariantstore** and **omicsannotationstore**.
2. Select **omicsvariantstore**, choose **Actions** and choose **Create resource link**. Give the resource link a name, such as *variants*. This will be the table name that will show up in Athena. This table can be added to the existing solution database >genomicsanalysis. Once created, repeat the process for the annotation store **omicsannotationstore** and give it a name, such as *annotations*.
3. Once resource links are created, select the newly created table *variants* under **Tables**, choose **Actions** and select **Grant on target**. Select the user and role that you want to allow access.

Note

To allow query from the SageMaker Notebook created as part of the solution, select the IAM Role created for the SageMaker notebook instance. To identify the IAM Role name, navigate to AWS Identity and Access Management (IAM) console, select **Roles** under **Access management** in the left navigation page, search for **RunbookRole**. You

should get a result with the full name of the IAM Role used by the SageMaker notebook instance.

- Next, under **LF-Tags** or **catalog resources**, select **Named data catalog resources** and choose the resource link created for the database, such as `<account-id>-<variant-store-id>-variant`. Select **All tables**. Under **Table permissions**, check the **Select** and **Describe** box under **Table permissions and Grantable permissions**. Select **All data access** under **Data permissions** and choose **Grant**. Repeat the process for the `annotations` table.
- After the previous step completes successfully, navigate to Amazon Athena in the console, select **AwsDataCatalog** under **Data source** and **genomicsanalysis** under **Database**. Verify that you are able to see the newly created tables `variants` and `annotations` under **Tables and views**. Also verify that you have selected **Workgroup genomicsanalysis-aws-region**. You are now ready to run a query against all the data in these databases.

Step 6. Query data in the data lake

Note

An [Amazon SageMaker AI](#) notebook instance is provisioned with an example Jupyter notebook that demonstrates how to work with data in a genomics data lake. The notebook uses [Amazon Athena](#) to identify genomic variants related to drug response for a given cohort of individuals. The following query is run against data in the data lake using the PyAthena driver to 1) filter by samples in a subpopulation, 2) aggregate variant frequencies for the subpopulation-of-interest, 3) join on the ClinVar dataset, 4) filter by variants that have been implicated in drug-response, 5) order by highest frequency variants.

You can generate an example cohort creation and drug response by running each step in the provided demonstration Jupyter Notebooks.

Option 1: Use the provided Jupyter Notebooks

Use the following steps to run the notebook:

- Sign in to the [Amazon SageMaker AI console](#).
- Select **Notebook Instances** under the Notebook category on the left pane.

3. Select your notebook instance and choose **Open Jupyter**.
4. To explore cohort creation, clinical dataset enrichment, and imaging visualization, select **cohort-creation.ipynb**.
5. Run each step in the notebook.
6. To explore the creation of a drug response report, in the main Jupyter window, select **runbook.ipynb**.
7. Run each step in the notebook.

Option 2: Run the drug response report query in the Amazon Athena console

Use the following setups to run the query in the Amazon Athena console:

1. Sign in to the [Amazon Athena console](#).
2. Under **Data Source** select the **AwsDataCatalog** data source on the left pane.
3. Under **Database** select the **genomicsanalysis** database on the left pane.
4. In the query window paste the query below.
5. Select **Run Query**.

```
SELECT count(*)/cast(numsamples AS DOUBLE) AS genotypefrequency
  ,cv.attributes['RS'] as rs_id
  ,cv.attributes['CLNDN'] as clinvar_disease_name
  ,cv.attributes['CLNSIG'] as clinical_significance
  ,sv.contigname
  ,sv.start
  ,sv."end"
  ,sv.referenceallele
  ,sv.alternatealleles
  ,sv.calls
  FROM {variant_table_name} sv
  CROSS JOIN
    (SELECT count(1) AS numsamples
     FROM
       (SELECT DISTINCT vs.sampleid
        FROM {variant_table_name} vs
        WHERE vs.sampleid LIKE 'NA12%'))
  JOIN {annotation_table_name} cv
  ON sv.contigname = cv.contigname
```

```
    AND sv.start = cv.start
    AND sv."end" = cv."end"
    AND sv.referenceallele = cv.referenceallele
    AND sv.alternatealleles = cv.alternatealleles
    AND cv.attributes['CLNSIG'] LIKE '%response%'
    AND sv.sampleid LIKE 'NA12%'
GROUP BY sv.contigname
        ,sv.start
        ,sv."end"
        ,sv.referenceallele
        ,sv.alternatealleles
        ,sv.calls
        ,cv.attributes['RS']
        ,cv.attributes['CLNDN']
        ,cv.attributes['CLNSIG']
        ,numsamples
ORDER BY genotypefrequency DESC LIMIT 50
```

Additional resources

AWS services

- | | |
|---------------------------------------|--|
| • AWS CloudFormation | • AWS Glue |
| • AWS CodeBuild | • Amazon Athena |
| • AWS CodeCommit | • Amazon Simple Storage Service |
| • Amazon CloudWatch | • AWS Identity and Access Management (IAM) |
| • Amazon SageMaker AI | • Amazon Omics |
| • AWS CodePipeline | • AWS Lake Formation |

Open-source licenses

Guidance for Multi-Omics and Multi-Modal Data Integration and Analysis on AWS uses the following open-source bioinformatics tools as part of ETL jobs and Jupyter notebooks. The tools are publicly available under open-source licenses as provided in the following table. Ensure that these license specifications meet your organizational requirements.

Tool name	Description	License
PyAthena	PyAthena is a Python DB API 2.0 (PEP 249) compliant client for Amazon Athena.	
PyDICOM	PyDICOM is a Python library which can be used to read, modify, and write DICOM image files. For more information, refer to the PyDICOM documentation .	MIT

Roles and permissions

This guidance uses AWS CodeBuild, AWS CloudFormation, and AWS CodePipeline for Continuous Delivery (CD) and AWS Glue and Amazon Athena for scientific analysis using a genomics data lake. Review the following CodeBuild, AWS CloudFormation, CodePipeline, AWS Glue, and Amazon Athena permissions to ensure that you have the appropriate permissions activated.

Code deployment pipeline permissions

Use IAM to manage access to AWS CodeBuild jobs, AWS CloudFormation stacks, and the AWS CodePipeline code pipeline. CodeBuild jobs and the CodePipeline code pipeline have their own IAM roles and IAM policies.

The following code examples demonstrate the IAM roles and supporting IAM policies defined in the `GenomicsAnalysisPipe/pipe_cfn.yml` file; including `CodeBuildRole`, `CodePipelineRole`, `CloudFormationRole`, and `SourceEventRole`.

CloudFormation role

`CloudFormationRole` defines the permissions needed for AWS CloudFormation to provision IAM roles, S3 buckets, an Amazon SageMaker AI notebook instance, and AWS Glue resources. AWS CloudFormation uses the `CloudFormation` action type in the CodePipeline.

```
CloudFormationRole:
  Type: AWS::IAM::Role
  Properties:
    Path: /
    AssumeRolePolicyDocument:
      Version: 2012-10-17
      Statement:
        - Effect: Allow
          Action:
            - sts:AssumeRole
          Principal:
            Service:
              - cloudformation.amazonaws.com
    Policies:
      - PolicyName: CloudFormationRolePolicy
        PolicyDocument:
          Version: 2012-10-17
```

Statement:

- Effect: Allow
 - Action:
 - iam:CreateRole
 - iam>DeleteRole
 - iam:PutRolePolicy
 - iam:GetRolePolicy
 - iam>DeleteRolePolicy
 - iam:AttachRolePolicy
 - iam:DetachRolePolicy
 - iam:UpdateAssumeRolePolicy
 - iam:PassRole
 - iam:GetRole
 - Resource:
 - !Sub arn:aws:iam::\${AWS::AccountId}:role/\${ResourcePrefix>*
- Effect: Allow
 - Action:
 - glue:CreateJob
 - glue:UpdateJob
 - glue>DeleteJob
 - glue:GetJob
 - Resource: '*'
- Effect: Allow
 - Action:
 - glue:CreateSecurityConfiguration
 - glue:GetSecurityConfiguration
 - glue>DeleteSecurityConfiguration
 - Resource: '*'
- Effect: Allow
 - Action:
 - glue:CreateWorkflow
 - glue>DeleteWorkflow
 - glue:UpdateWorkflow
 - Resource: '*'
- Effect: Allow
 - Action:
 - glue:GetDataCatalogEncryptionSettings
 - glue:PutDataCatalogEncryptionSettings
 - glue>DeleteDataCatalogEncryptionSettings
 - Resource:
 - !Sub arn:aws:glue:\${AWS::Region}:\${AWS::AccountId}:catalog
- Effect: Allow
 - Action:
 - glue>CreateDatabase

```

    - glue:UpdateDatabase
    - glue>DeleteDatabase
    - glue:GetDatabase
    - glue:GetDatabases
    - glue:GetCrawler
    - glue:CreateCrawler
    - glue:UpdateCrawler
    - glue>DeleteCrawler
    - glue:StopCrawler
    - glue:StopTrigger
    - glue:GetTrigger
    - glue>CreateTrigger
    - glue>DeleteTrigger
    - glue:UpdateTrigger
  Resource:
    - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:catalog
    - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:database/
      ${ResourcePrefixLowercase}
    - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:table/
      ${ResourcePrefixLowercase}/*
    - !Sub arn:aws:glue:${AWS::Region}:
      ${AWS::AccountId}:userDefinedFunction/${ResourcePrefixLowercase}/*
    - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:crawler/
      ${ResourcePrefixLowercase}*
    - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:trigger/
      ${ResourcePrefixLowercase}*
  - Effect: Allow
  Action:
    - glue:CreateTable
    - glue:UpdateTable
    - glue>DeleteTable
    - glue:SearchTables
    - glue:GetTable
    - glue:GetTables
    - glue:GetPartitions
  Resource:
    - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:catalog
    - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:database/
      ${ResourcePrefixLowercase}
    - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:table/
      ${ResourcePrefixLowercase}/*
  - Effect: Allow
  Action:
    - lambda:CreateFunction

```

```

    - lambda:DeleteFunction
    - lambda:GetFunctionConfiguration
    - lambda:GetFunction
    - lambda:InvokeFunction
    - lambda:ListTags
    - lambda:TagResource
    - lambda:UntagResource
    - lambda:UpdateFunctionCode
    - lambda:UpdateFunctionConfiguration
  Resource:
    - !Sub arn:aws:lambda:${AWS::Region}:${AWS::AccountId}:function:
      ${ResourcePrefix}*
  - Effect: Allow
  Action:
    - lambda:PublishLayerVersion
    - lambda>DeleteLayerVersion
    - lambda:GetLayerVersion
  Resource:
    - !Sub arn:aws:lambda:${AWS::Region}:
      ${AWS::AccountId}:layer:OmicsApiModels
    - !Sub arn:aws:lambda:${AWS::Region}:
      ${AWS::AccountId}:layer:OmicsApiModels:*
  - Effect: Allow
  Action:
    - athena:GetWorkGroup
    - athena>CreateWorkGroup
    - athena>DeleteWorkGroup
  Resource:
    - !Sub arn:aws:athena:${AWS::Region}:${AWS::AccountId}:workgroup/
      ${ResourcePrefixLowercase}-${AWS::Region}
  - Effect: Allow
  Action:
    - kms>CreateKey
    - kms:GenerateDataKey
  Resource: '*'
- Effect: Allow
  Action:
    - s3>CreateBucket
    - s3>DeleteBucket
    - s3:GetObject
  Resource:
    - !Sub ${BuildBucket.Arn}
    - !Sub ${BuildBucket.Arn}/*
    - !Sub ${ResourcesBucket.Arn}

```

```

    - !Sub ${ResourcesBucket.Arn}/*
  - Effect: Allow
    Action:
      - s3:GetObject
    Resource:
      - !Sub ${ResourcesBucket.Arn}/artifacts/*
      - arn:aws:s3:::aws-genomics-static-us-east-1/*
  - Effect: Allow
    Action:
      - sagemaker:CreateNotebookInstanceLifecycleConfig
      - sagemaker:DescribeNotebookInstanceLifecycleConfig
      - sagemaker:UpdateNotebookInstanceLifecycleConfig
      - sagemaker>DeleteNotebookInstanceLifecycleConfig
      - sagemaker:CreateNotebookInstance
      - sagemaker:UpdateNotebookInstance
      - sagemaker:StartNotebookInstance
      - sagemaker:DescribeNotebookInstance
      - sagemaker>DeleteNotebookInstance
      - sagemaker:StopNotebookInstance
    Resource:
      - !Sub arn:aws:sagemaker:${AWS::Region}:${AWS::AccountId}:notebook-
instance-lifecycle-config/${ResourcePrefixLowercase}*
      - !Sub arn:aws:sagemaker:${AWS::Region}:${AWS::AccountId}:notebook-
instance/${ResourcePrefixLowercase}*

```

CodeBuild role

CodeBuildRole defines the permissions needed for CodeBuild to run a code build job that copies the resources needed to Amazon S3 buckets and build custom scripts that support Amazon Omics resource creation using AWS CloudFormation. The CodeBuild job is run using the CodeBuild action type in the CodePipeline.

```

CodeBuildRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: 2012-10-17
      Statement:
        - Action:
            - sts:AssumeRole
          Effect: Allow
        Principal:

```

```
Service:
  - codebuild.amazonaws.com
Path: /
Policies:
  - PolicyName: CodeBuildAccess
    PolicyDocument:
      Version: 2012-10-17
      Statement:
        - Effect: Allow
          Resource:
            - !Sub arn:aws:logs:${AWS::Region}:${AWS::AccountId}:log-group:/aws/codebuild/${ResourcePrefix}*
          Action:
            - logs:CreateLogGroup
            - logs:CreateLogStream
            - logs:PutLogEvents
        - Effect: Allow
          Action:
            - s3:GetObject
            - s3:GetObjectVersion
            - s3:PutObject
          Resource:
            - !Sub ${BuildBucket.Arn}/*
            - !Sub ${ResourcesBucket.Arn}/*
        - Effect: Allow
          Action:
            - s3:ListBucket
          Resource:
            - !Sub ${ResourcesBucket.Arn}
            - !Sub ${DataLakeBucket.Arn}
        - Effect: Allow
          Action:
            - s3:PutObject
            - s3:PutObjectAcl
          Resource:
            - !Sub ${ResourcesBucket.Arn}
            - !Sub ${ResourcesBucket.Arn}/*
            - !Sub ${DataLakeBucket.Arn}
            - !Sub ${DataLakeBucket.Arn}/*
```

Source event role

SourceEventRole defines the permissions needed for an Amazon CloudWatch event to initiate the deployment pipeline.

```
SourceEventRole:
  Type: AWS::IAM::Role
  DependsOn: CodePipeline
  Description: IAM role to allow Amazon CloudWatch Events to trigger AWS CodePipeline execution
  Properties:
    AssumeRolePolicyDocument:
      Statement:
        - Action: sts:AssumeRole
          Effect: Allow
          Principal:
            Service:
              - events.amazonaws.com
          Sid: 1
    Policies:
      - PolicyName: CloudWatchEventPolicy
        PolicyDocument:
          Statement:
            - Action:
                - codepipeline:StartPipelineExecution
              Effect: Allow
              Resource:
                - !Sub arn:aws:codepipeline:${AWS::Region}:${AWS::AccountId}:
                    ${CodePipeline}*

```

AWS Glue and Amazon SageMaker AI notebook permissions

Use IAM to manage access to the datasets and scripts in Amazon S3 using AWS Glue, and to define the permissions for your Amazon SageMaker AI Jupyter notebook instance. Adding new AWS Glue jobs and crawlers does not require any changes to the following roles or policies, as long as you add those resources with the `${Project}` prefix.

The following code examples demonstrate the IAM roles and supporting IAM policies defined in the `GenomicsAnalysisCode/code_cfn.yml` and `GenomicsAnalysisCode/TCIA_etl.yml` files, including `JobRole`, `GlueJobRole`, `CrawlerRole`, and `RunbookRole`.

Job role

JobRole defines the permissions needed for AWS Glue to run Extract, Transform, and Load (ETL).

This role must be updated to get or put objects in additional S3 buckets in your AWS account or S3 buckets in other accounts.

```
JobRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: 2012-10-17
      Statement:
        - Effect: Allow
          Principal:
            Service:
              - glue.amazonaws.com
          Action:
            - sts:AssumeRole
    Path: /
    ManagedPolicyArns:
      - arn:aws:iam::aws:policy/service-role/AWSGlueServiceRole
    Policies:
      - PolicyName: s3_access
        PolicyDocument:
          Version: 2012-10-17
          Statement:
            - Effect: Allow
              Action:
                - athena:StartQueryExecution
                - athena:GetQueryExecution
                - athena:GetQueryResults
              Resource:
                - !Sub arn:aws:athena:${AWS::Region}:${AWS::AccountId}*
            - Effect: Allow
              Action:
                - s3:GetObject
                - s3:ListBucket
              Resource:
                - !Sub arn:aws:s3:::${ResourcesBucket}
                - !Sub arn:aws:s3:::${ResourcesBucket}/*
            - Effect: Allow
              Action:
```

```

    - s3:PutObject
    - s3:GetObject
    - s3:ListBucket
    - s3:DeleteObject
  Resource:
    - !Sub arn:aws:s3:::${DataLakeBucket}
    - !Sub arn:aws:s3:::${DataLakeBucket}/*
- PolicyName: kms_access
PolicyDocument:
  Version: 2012-10-17
  Statement:
    - Effect: Allow
      Action:
        - kms:GenerateDataKey
        - kms:Decrypt
        - kms:Encrypt
      Resource:
        - !GetAtt DataCatalogEncryptionKey.Arn

```

Glue job role

GlueJobRole defines the permissions needed for AWS Glue to run Extract, Transform, and Load (ETL). In addition, it defines the permissions needed to run an AWS Glue crawler on a dataset in an Amazon S3 bucket, infer the schema, and add or update a table in the AWS Glue data catalog.

This role must be updated to get or put objects in additional S3 buckets in your AWS account or S3 buckets in other accounts.

```

GlueJobRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
        - Effect: "Allow"
          Principal:
            Service: "glue.amazonaws.com"
          Action: "sts:AssumeRole"
    Path: "/"
    ManagedPolicyArns:
      - arn:aws:iam::aws:policy/service-role/AWSGlueServiceRole
    Policies:
      - PolicyName: athena_access

```

```

PolicyDocument:
  Version: 2012-10-17
  Statement:
    - Effect: Allow
      Action:
        - athena:StartQueryExecution
        - athena:GetQueryExecution
        - athena:GetQueryResults
      Resource:
        - !Sub arn:aws:athena:${AWS::Region}:${AWS::AccountId}:workgroup/
primary
- PolicyName: kms_access
  PolicyDocument:
    Version: 2012-10-17
    Statement:
      - Effect: Allow
        Action:
          - kms:GenerateDataKey
          - kms:Decrypt
          - kms:Encrypt
        Resource:
          - !ImportValue
            Fn::Sub: '${ResourcePrefix}-DataCatalogEncryptionKeyArn'
- PolicyName: "CrawlerAccess"
  PolicyDocument:
    Version: "2012-10-17"
    Statement:
      - Effect: "Allow"
        Action:
          - s3:PutObject
          - s3:GetObject
          - s3:ListBucket
          - s3:DeleteObject
        Resource:
          - !Sub 'arn:aws:s3:::${DataLakeBucket}'
          - !Sub 'arn:aws:s3:::${DataLakeBucket}/*'

```

Runbook role

RunbookRole provides the permissions needed for the Amazon SageMaker AI Jupyter notebook instance to access the AWS Glue data catalog and use Amazon Athena to run queries against the data lake.

RunbookRole:

Type: AWS::IAM::Role

Properties:**AssumeRolePolicyDocument:**

Version: 2012-10-17

Statement:

- Effect: Allow
- Principal:
 - Service:
 - sagemaker.amazonaws.com
- Action:
 - sts:AssumeRole

Path: /

Policies:

- PolicyName: logs_access

PolicyDocument:

Version: 2012-10-17

Statement:

- Effect: Allow
- Action:
 - logs:CreateLogStream
 - logs:DescribeLogStreams
 - logs:CreateLogGroup
 - logs:PutLogEvents

Resource:

- !Sub arn:aws:logs:\${AWS::Region}:\${AWS::AccountId}:log-group:/aws/sagemaker/*
- !Sub arn:aws:logs:\${AWS::Region}:\${AWS::AccountId}:log-group:/aws/sagemaker/*:log-stream:aws-glue-*

- PolicyName: s3_access

PolicyDocument:

Version: 2012-10-17

Statement:

- Effect: Allow
- Action:
 - s3:ListBucket
 - s3:GetBucketLocation
- Resource:
 - !Sub arn:aws:s3:::\${DataLakeBucket}
 - !Sub arn:aws:s3:::\${ResourcesBucket}
- Effect: Allow
- Action:
 - s3:GetObject

```

    - s3:GetObjectAcl
    - s3:PutObject
    - s3>DeleteObject
  Resource:
    - !Sub arn:aws:s3:::${DataLakeBucket}/*
- Effect: Allow
  Action:
    - s3:GetObject
  Resource:
    - !Sub arn:aws:s3:::${ResourcesBucket}/*
- PolicyName: glue_access
  PolicyDocument:
    Version: 2012-10-17
    Statement:
      - Effect: Allow
        Action:
          - glue:StartCrawler
          - glue:StartJobRun
          - glue:StartTrigger
        Resource:
          - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:crawler/
            ${ResourcePrefixLowercase}*
          - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:job/
            ${ResourcePrefixLowercase}*
          - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:trigger/
            ${ResourcePrefixLowercase}*
      - Effect: Allow
        Action:
          - kms:GenerateDataKey
          - kms:Decrypt
          - kms:Encrypt
        Resource:
          - !GetAtt DataCatalogEncryptionKey.Arn
- PolicyName: glue_table_access
  PolicyDocument:
    Version: 2012-10-17
    Statement:
      - Effect: Allow
        Action:
          - glue:GetDatabases
        Resource: '*'
      - Effect: Allow
        Action:
          - glue:GetDatabase

```

```

    - glue:CreateDatabase
  Resource:
    - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:database/default
- Effect: Allow
  Action:
    - glue:GetTable
    - glue:GetTables
    - glue:CreateTable
    - glue:UpdateTable
    - glue:GetDatabase
    - glue:GetPartition
    - glue:GetPartitions
    - glue:GetDevEndpoint
    - glue:GetDevEndpoints
    - glue:UpdateDevEndpoint
  Resource:
    - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:catalog
    - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:database/
    ${ResourcePrefixLowercase}
    - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:table/
    ${ResourcePrefixLowercase}/*
    - !Sub arn:aws:glue:${AWS::Region}:${AWS::AccountId}:devEndpoint/*
- PolicyName: athena_access
  PolicyDocument:
    Version: 2012-10-17
    Statement:
      - Effect: Allow
        Action:
          - athena:StartQueryExecution
          - athena:GetQueryExecution
          - athena:GetQueryResults
        Resource:
          - !Sub arn:aws:athena:${AWS::Region}:${AWS::AccountId}:workgroup/
primary
- PolicyName: cfn_access
  PolicyDocument:
    Version: 2012-10-17
    Statement:
      - Effect: Allow
        Action:
          - cloudformation:DescribeStacks
        Resource:
          - !Sub arn:aws:cloudformation:${AWS::Region}:${AWS::AccountId}:stack/
    ${ResourcePrefix}*

```

```
- PolicyName: kms_access
  PolicyDocument:
    Version: 2012-10-17
    Statement:
      - Effect: Allow
        Action:
          - kms:GenerateDataKey
          - kms:Decrypt
          - kms:Encrypt
        Resource:
          - !GetAtt DataCatalogEncryptionKey.Arn
```

Uninstall this guidance

You can uninstall Guidance for Multi-Omics and Multi-Modal Data Integration and Analysis on AWS using the AWS Management Console, the AWS Command Line Interface (AWS CLI), or manually.

Note

Uninstalling this guidance deletes the Amazon Simple Storage Service (Amazon S3) buckets and the data in those buckets; AWS CodeCommit repositories and the code in them; the Quick dataset; AWS CodeCommit repositories and the code in them; the AWS Glue jobs, crawlers, triggers, and data; and the Amazon SageMaker AI notebook instance.

Using the AWS Management console

1. Sign in to the [AWS CloudFormation console](#).
2. Select your installation stack that was launched first and typically has a name ending in -Setup. All other guidance stacks will be deleted automatically.
3. Choose **Delete**.

Using AWS CLI

Determine whether AWS CLI is available in your environment. For installation instructions, see [What Is the AWS Command Line Interface](#) in the *AWS CLI User Guide*. After confirming the AWS CLI is available, run the following command.

```
$ aws cloudformation delete-stack --stack-name <installation-stack-name>
```

Uninstall manually

To manually uninstall this solution, you must delete the related AWS CloudFormation stacks using the following procedure, in the specified order.

1. Delete all *<project-name>*-*bucket Amazon S3 bucket contents.
2. Delete the *<project-name>*-Quicksight stack.

⚠ Important

Wait for deletion to complete successfully before proceeding.

3. Delete the *<project-name>*-Imaging stack.

⚠ Important

Wait for deletion to complete successfully before proceeding.

4. Delete the *<project-name>*-Omics stack.

⚠ Important

Wait for deletion to complete successfully before proceeding. If Omics stack deletion fails, manually delete the *variants*, *annotation* and *reference* stores using the Amazon Omics console or API, followed by a re-attempt to delete the Omics stack.

5. Delete the *<project-name>*-Genomics stack.

⚠ Important

Wait for deletion to complete successfully before proceeding.

6. Delete the *<project-name>*-Pipeline stack.

⚠ Important

Wait for deletion to complete successfully before proceeding.

7. Delete the *<project-name>*-LandingZone stack.

⚠ Important

Wait for deletion to complete successfully before proceeding.

8. Delete the installation stack.

9. Delete the *<project-name>* AWS CodeCommit repository.

10 Delete the *<project-name>*-Pipe AWS CodeCommit repository.

11 [Cancel your Quick subscription.](#)

Run the TCGA Glue workflow

This guidance includes an example AWS Glue workflow to process the TCGA data. You can run the workflow using either the AWS Command Line Interface (AWS CLI) or the AWS Glue console.

To start the workflow using the AWS CLI, run the following command:

```
aws glue start-workflow-run --name TCGAWorkflow
```

Use the following process to run the crawler in the AWS Glue console.

1. Sign in to the [AWS Glue console](#).
2. Choose **Workflows** from the left navigation menu. On the **Workflows** page, select the name of the example workflow –TCGAWorkflow.
3. Choose **Actions** and select **Run**.

Collection of operational metrics

This guidance includes an option to send anonymous operational metrics to AWS. We use this data to better understand how customers use it and related services and products. When activated, the following information is collected and sent to AWS:

- **Solution ID** - The guidance identifier.
- **Unique ID (UUID)** - Randomly generated, unique identifier for each deployment.
- **Timestamp** - Data-collection timestamp.

AWS owns the data gathered through this survey. Data collection is subject to the [AWS Privacy Policy](#). To opt out of this feature, complete the following steps before launching the AWS CloudFormation template.

1. Download the [AWS CloudFormation template](#) to your local hard drive.
2. Open the AWS CloudFormation template with a text editor.
3. Modify the AWS CloudFormation template mapping section from:

```
AnonymousData:  
  SendAnonymousData:  
    Data: Yes
```

to:

```
AnonymousData:  
  SendAnonymousData:  
    Data: No
```

4. Sign in to the [AWS CloudFormation console](#).
5. Select **Create stack**.
6. On the **Create stack** page, **Specify template** section, select **Upload a template file**.
7. Under **Upload a template file**, choose **Choose file** and select the edited template from your local drive.
8. Choose **Next** and follow the steps in [Launch the stack](#) in the Automated Deployment section of this guide.

Source code

You can visit our [GitHub repository](#) to download the templates and scripts for this guidance, and to share your customizations with others.

Revisions

Date	Change
July 2020	Initial release
July 2022	Release v2.0.0: Added guidance for multi-modal multi-omic analysis. For more information, refer to the CHANGELOG.md in the GitHub repository.
January 2023	Release v3.0.0: Added support for variant query using Amazon Omics and Amazon Athena. For more information, refer to the CHANGELOG.md in the GitHub repository.

Contributors

- Karl Gutwin (Bioteam)
- Nicholas George (Bioteam)
- Sujaya Srinivasan (AWS)
- Ryan Ulaszek (AWS)
- Olivia Choudhury (AWS)
- Nadeem Bulsara (AWS)

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Guidance for Multi-Omics and Multi-Modal Data Integration and Analysis on AWS is licensed under the terms of the Apache License Version 2.0 available at [The Apache Software Foundation](#).

AWS Glossary

For the latest AWS terminology, see the [AWS glossary](#) in the *AWS Glossary Reference*.