AWS re:Invent

ANT 307

Athena deep dive

Janak Agarwal
Product Manager
Athena, Amazon Web Services

Anthony Virtuoso
Principal Engineer
Athena, Amazon Web Services

AWS re:Invent

aws

Amazon Athena

Amazon Athena is an interactive query service that makes it easy to analyze data using standard SQL. Athena is serverless, so there is no infrastructure to manage, and you pay only for the queries that you run.

- Query data in your Amazon S3 based data lake
- Analyze infrastructure, operation, and application logs
- Interactive analytics using popular BI tools
- Self-service data exploration for data scientists
- Embed analytics capabilities into your applications

Related breakouts

ANT205 What’s new with Amazon Athena

ANT218 Data lakes and data integration with AWS Lake Formation

ANT222 Analytics with Amazon Athena (workshop)

How will we spend our time today

Common Athena usage patterns

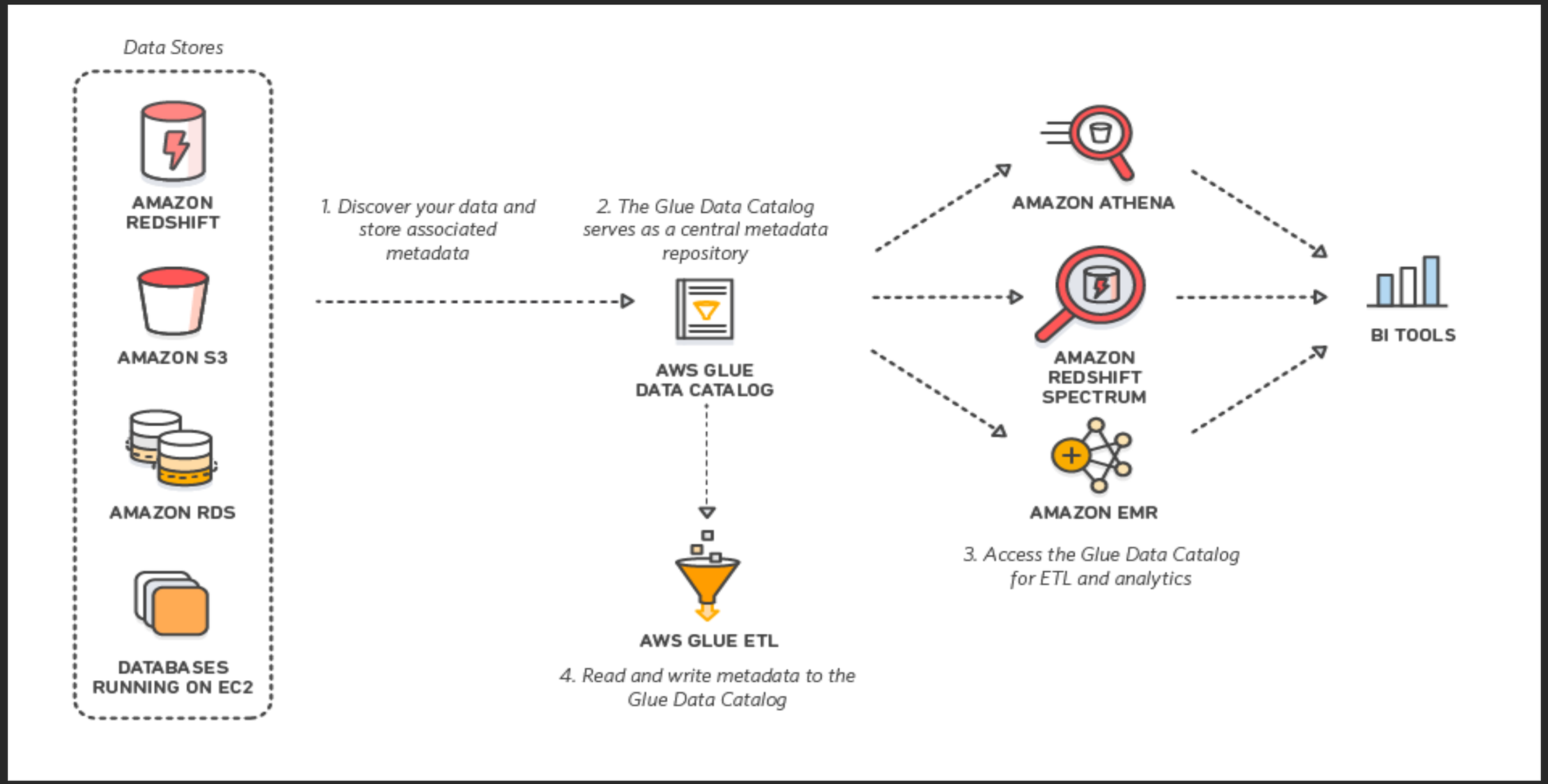
Dissect key use-cases and challenges

Introduce new features to help solve the challenges

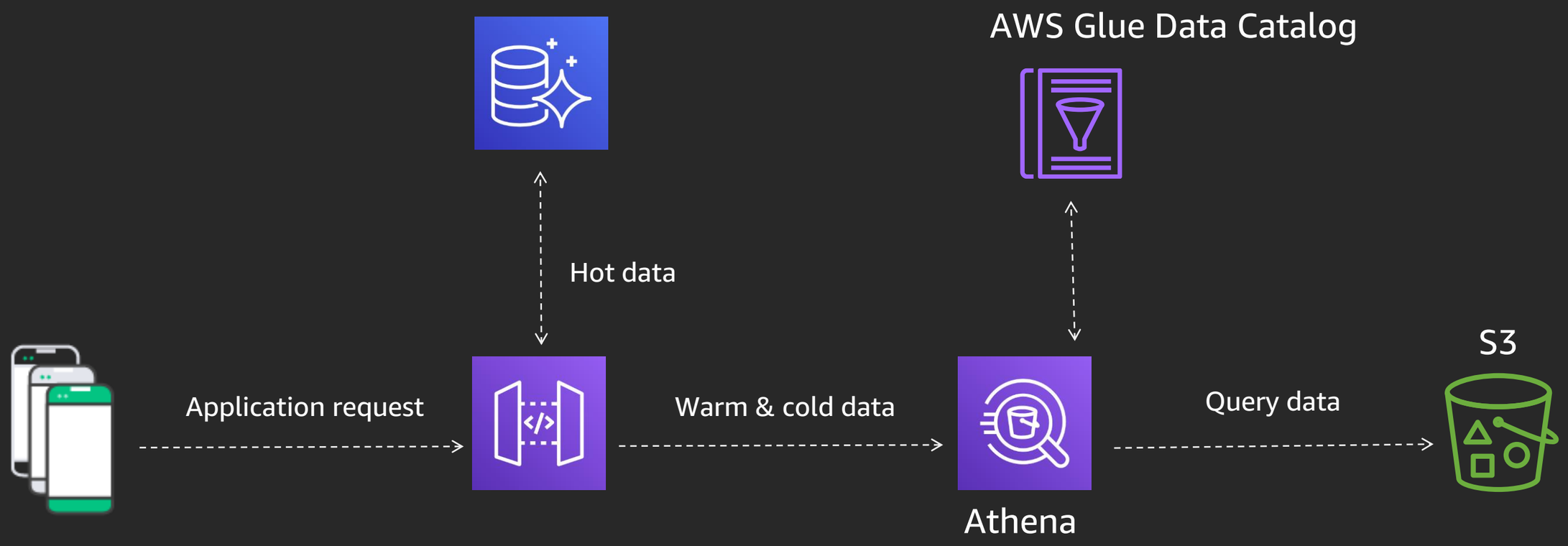
Demo

Common Athena usage patterns

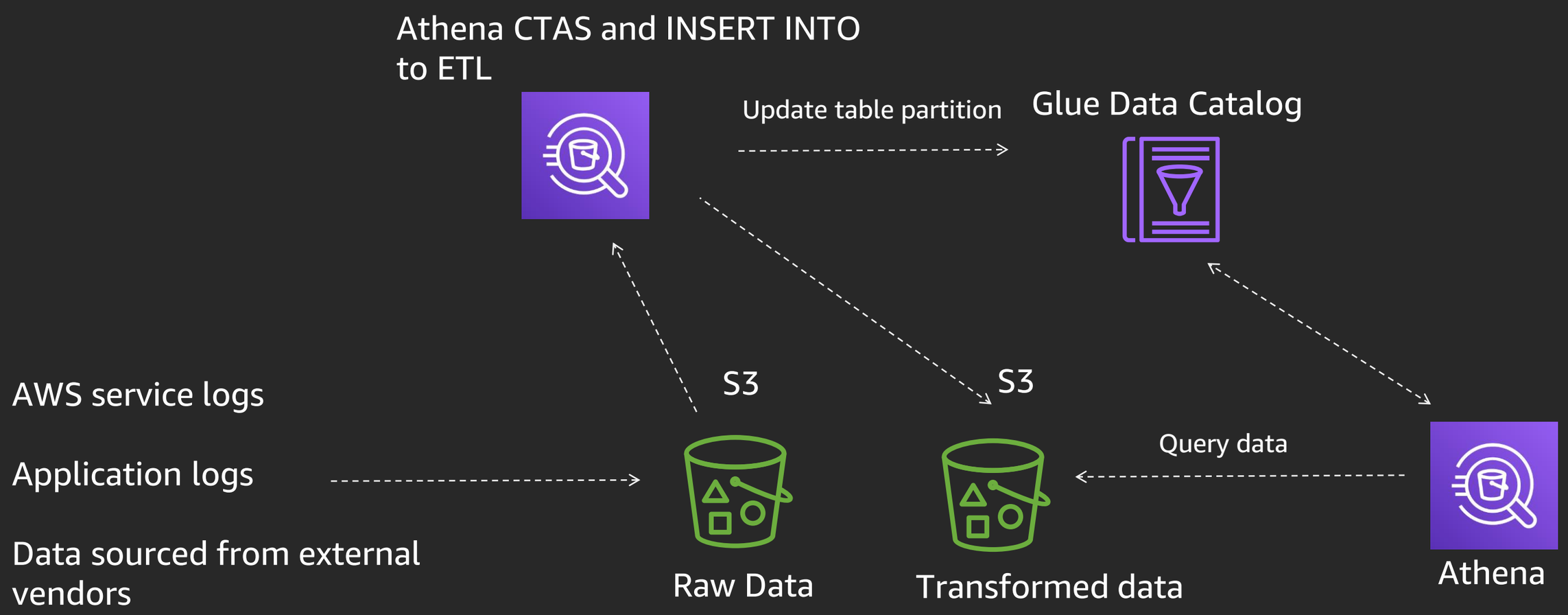
1: Ad-hoc use-case



2: SaaS use-case

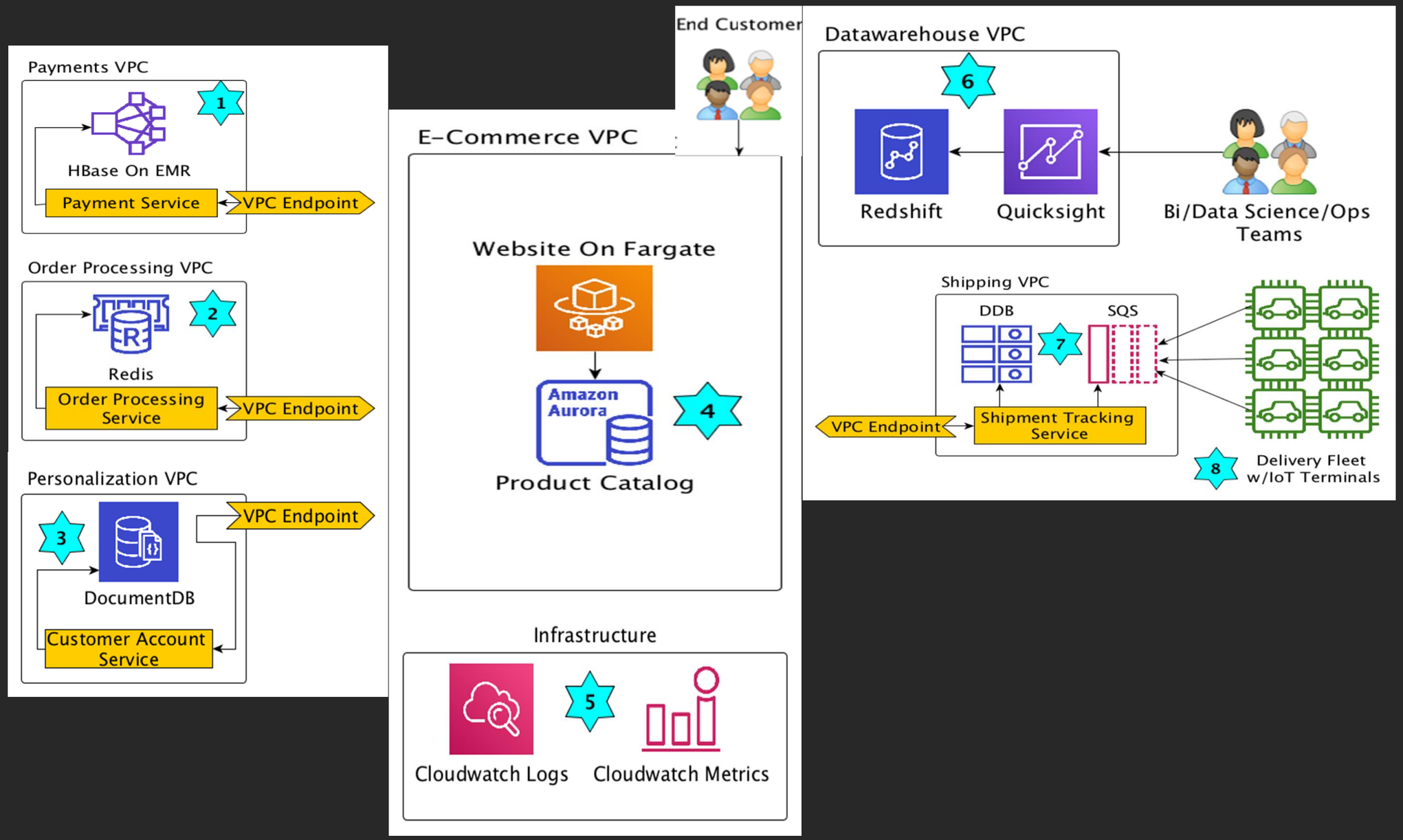


3: ETL and query use-case



Key personas in an organization

Example: E-commerce firm architecture



Key personas in a typical organization

Ana – the Analyst

Carlos – the Administrator

Richard – the Engineer

Maria – the Scientist

Use-cases of the key personas

Ana – the Analyst

Schedules reports

Ad-hoc investigations

Carlos – the Administrator

Manages data lake, security, & cost

Health and performance of data systems

Richard – the Engineer

Builds and manages SaaS applications using Athena APIs

Maria – the Scientist

Builds and Trains models

Helps analysts with Machine Learning

Challenges vary by personas

Ana – the Analyst

Schedules reports

Ad-hoc investigations

Data spread across systems

Pipelines require complex languages

Dependent on Data engineering

Carlos – the Administrator

Manages data lake, security & cost

Health and performance of data systems

Data in a variety of sources

Manage and audit access

Teams want to experiment with tech

Richard – the Engineer

Builds and manages SaaS applications using Athena APIs

Support data formats

Support new ML models and use-cases

Learn new object paradigms

Maria – the Scientist

Builds and Trains models

Helps analysts with Machine Learning

Complex ETL to retrieve data

Analysts rely on her to run inference

How do we solve all these challenges ?

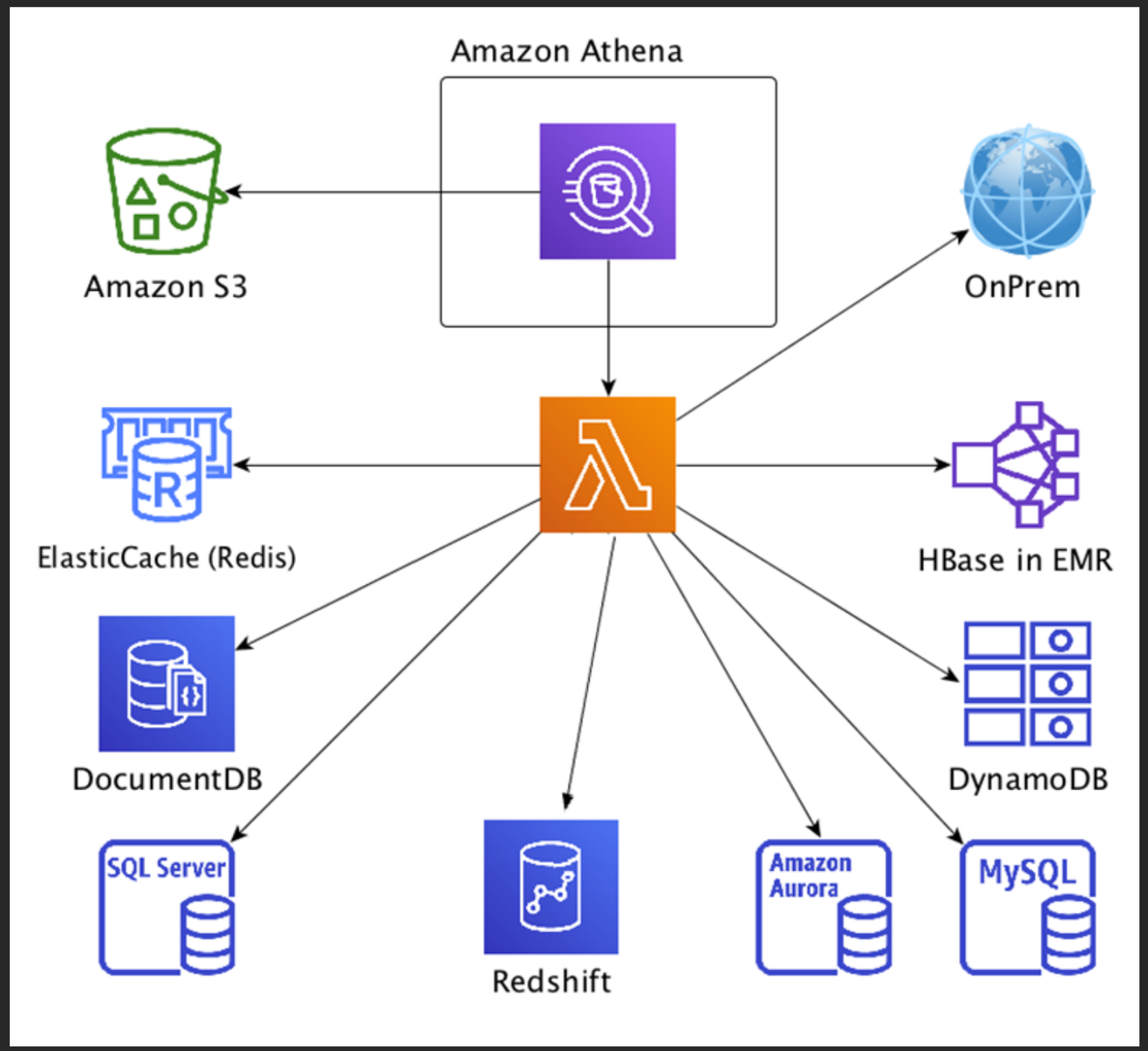
Introducing federated query in Athena (Preview)

AWS re:Invent

aws

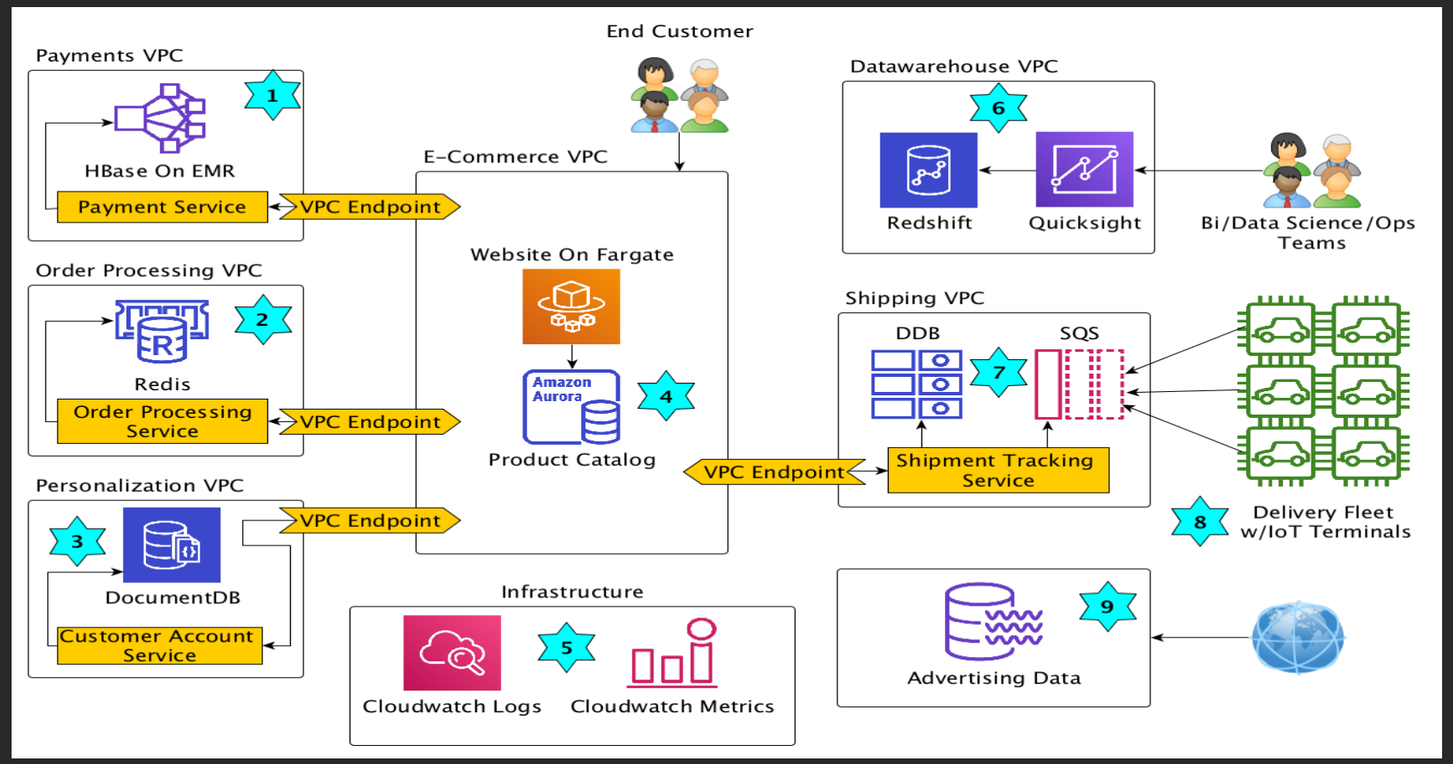
What is federated query?

- Run query across relational, non-relational, object, or custom data sources
- Run query across On-Premises or cloud data sources
- Can be used for ad-hoc investigations, or complex pipelines, or applications



Why do you need federated query

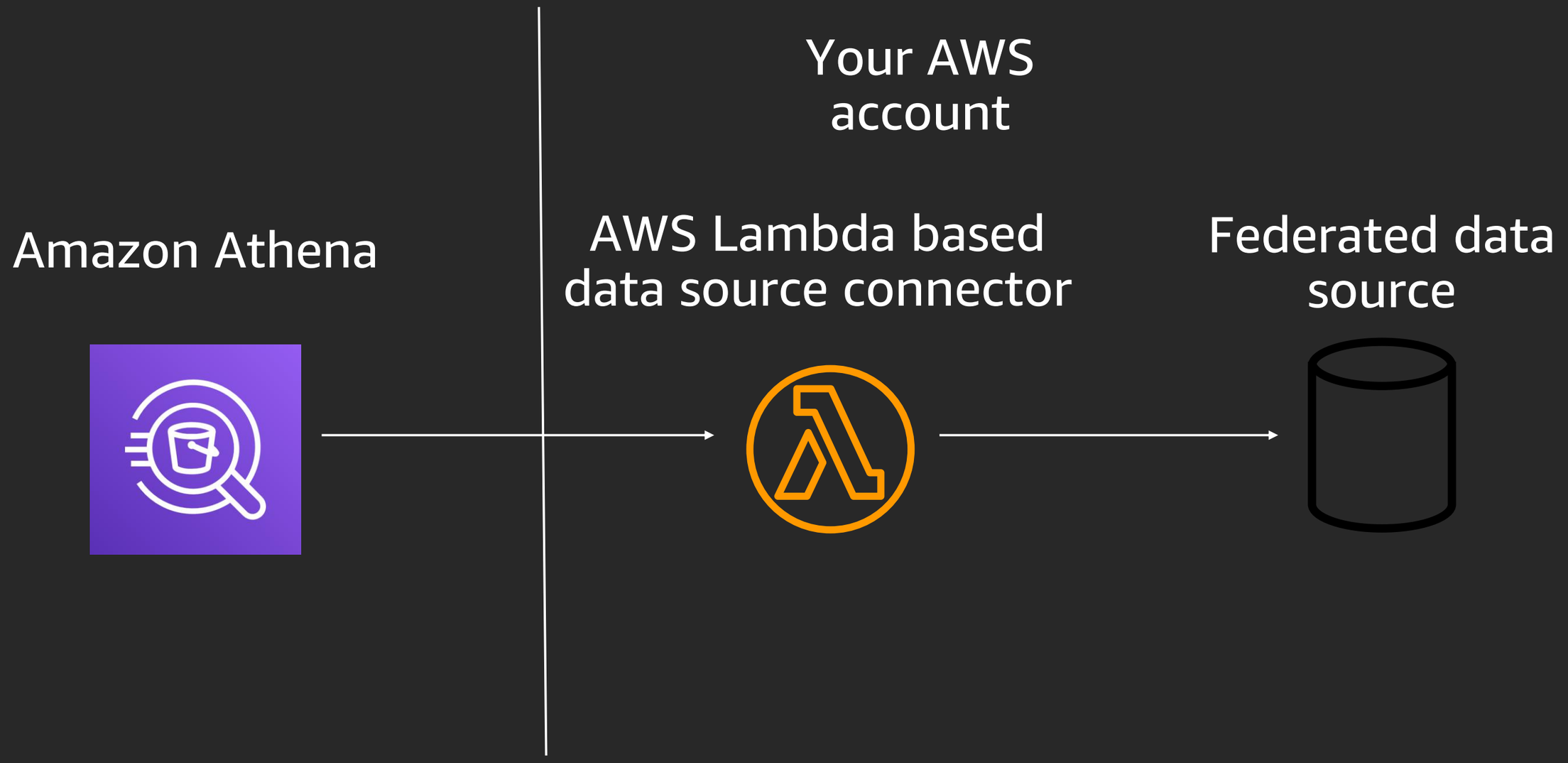
Evolving architecture



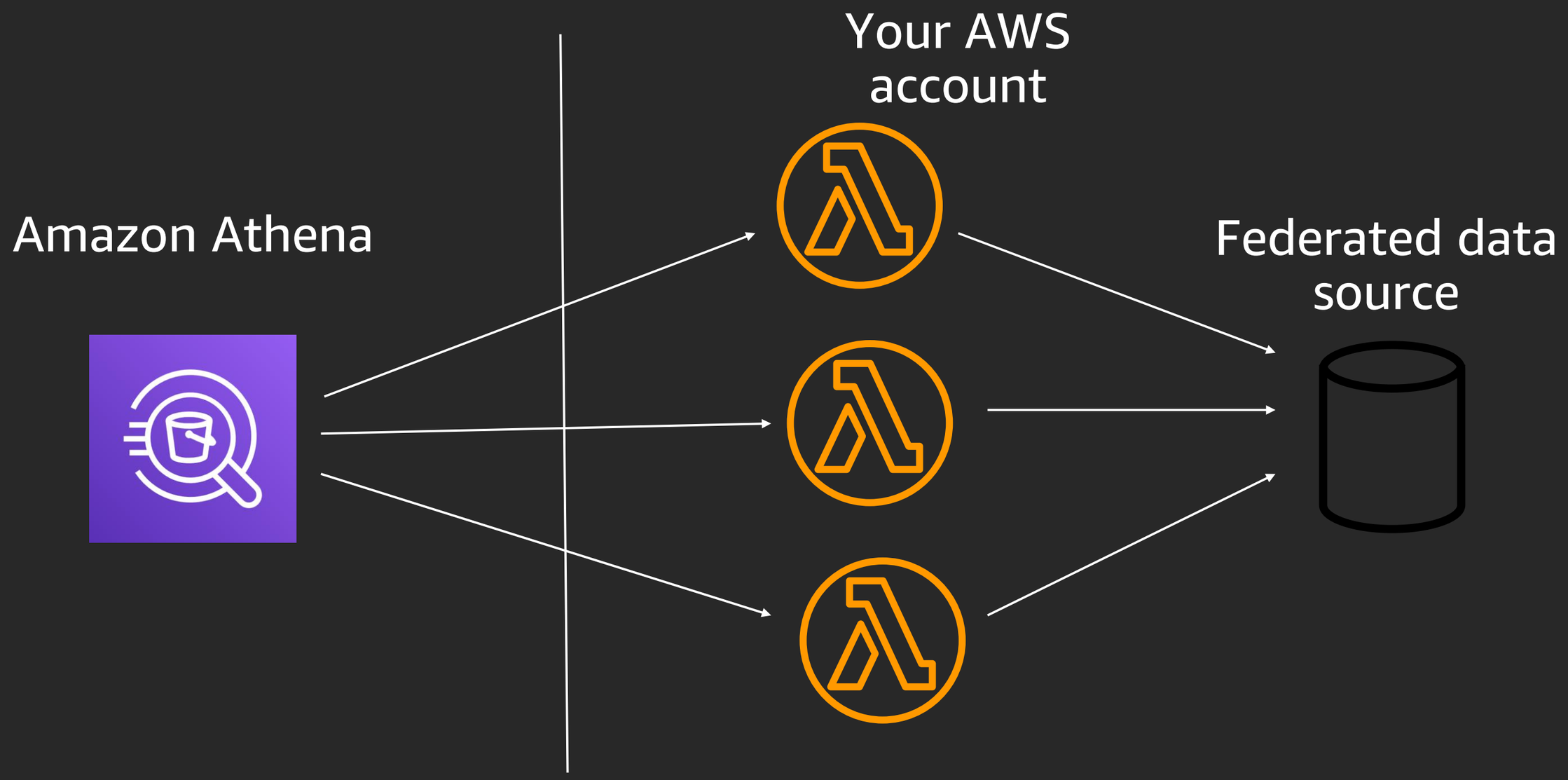
Engineering teams use fit for purpose databases

Aggregating data for analytics is a challenge

Anatomy of a federated query



Running a federated query



Federated query is simple to use

1

Deploy data source connector

2

Register data source connector. Specify a catalog name

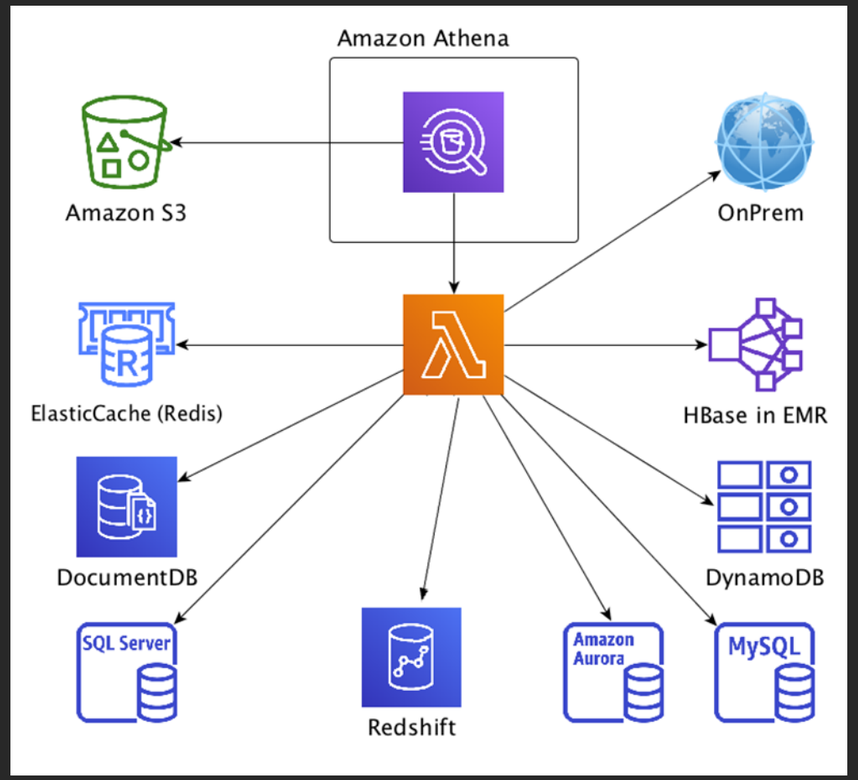
3

Write SQL Query

<CatalogName>.Database.Table

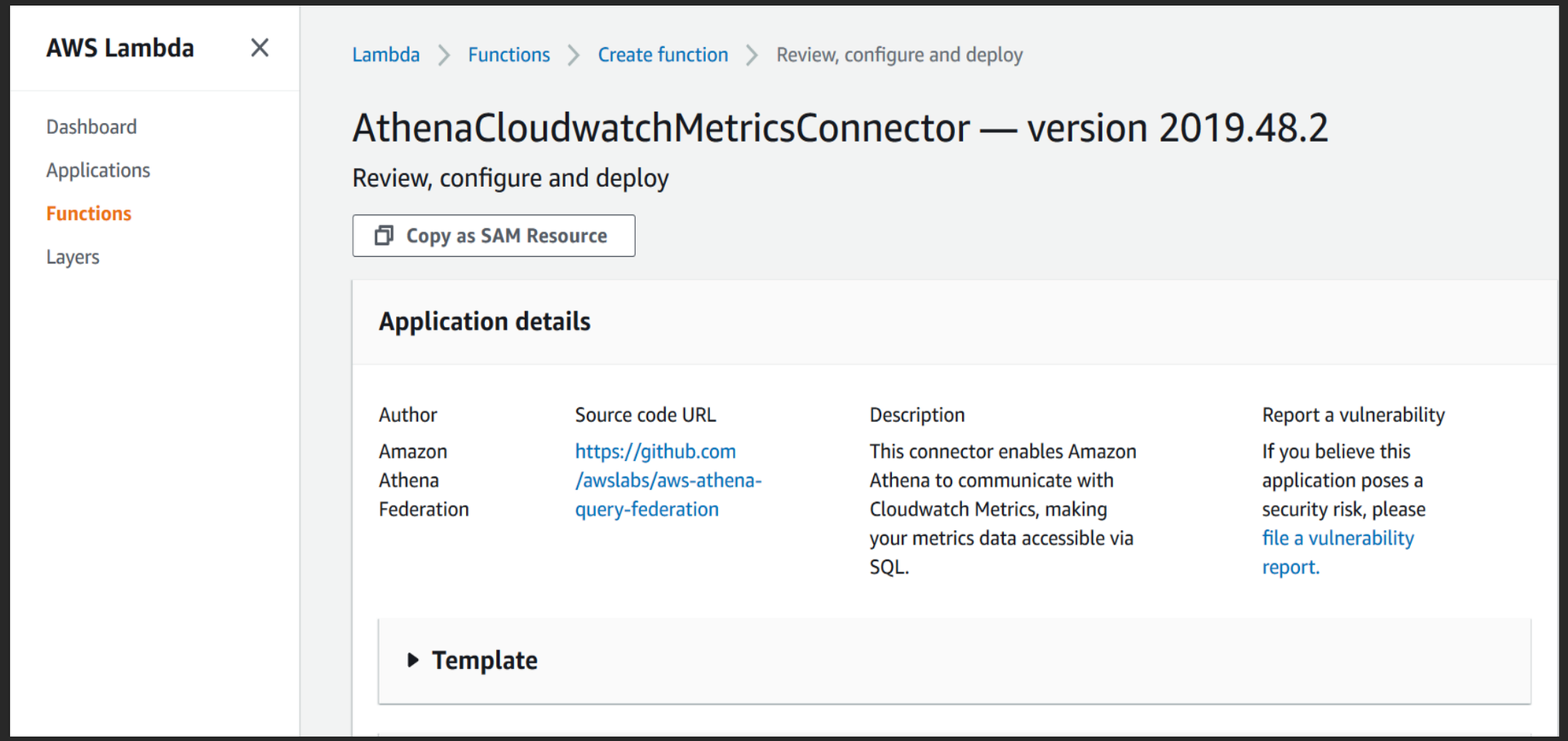
How to deploy a data source connector

- Athena uses AWS Lambda based data source connectors
- Two ways to deploy connector
  - One-Click deploy using AWS Serverless Application Repository
  - Deploy connector code to Lambda



One-click deploy using Serverless Application Repository

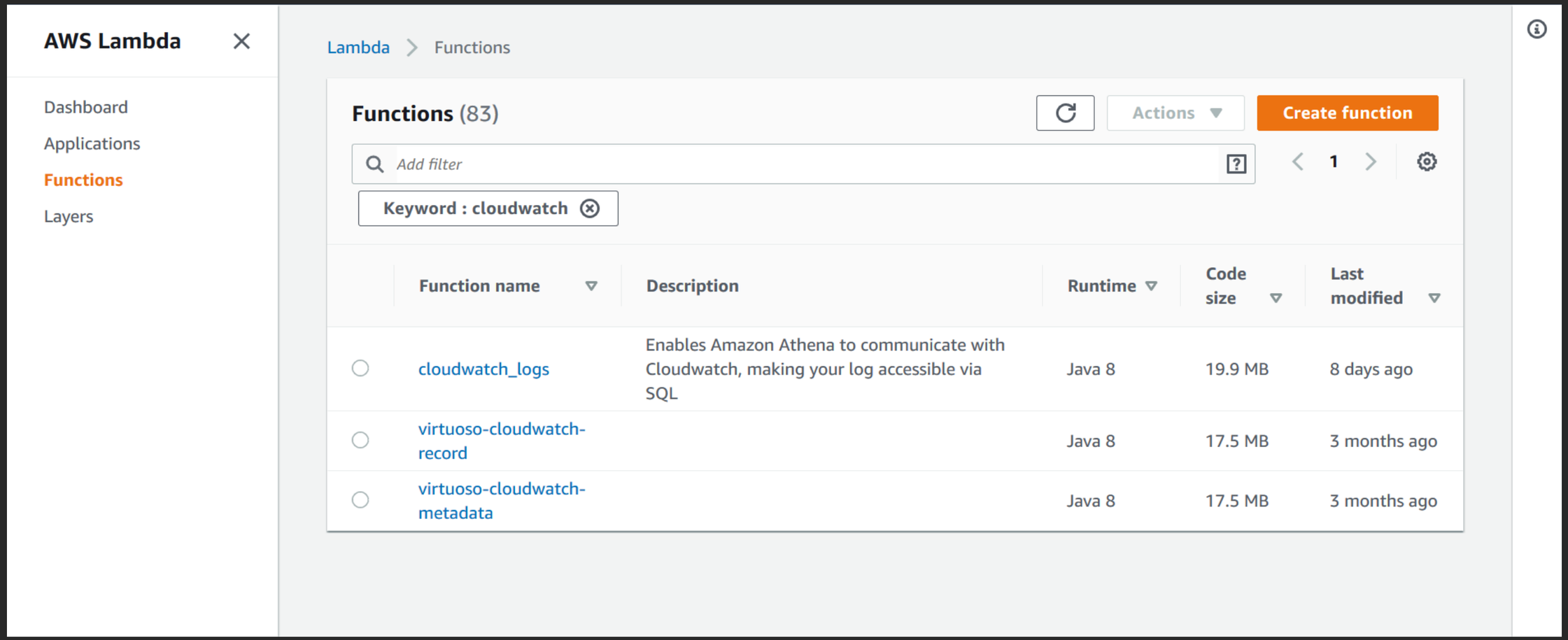
Upload connector to AWS Serverless Application Repository



Deploy | Register | Use

Deploy connector to AWS Lambda

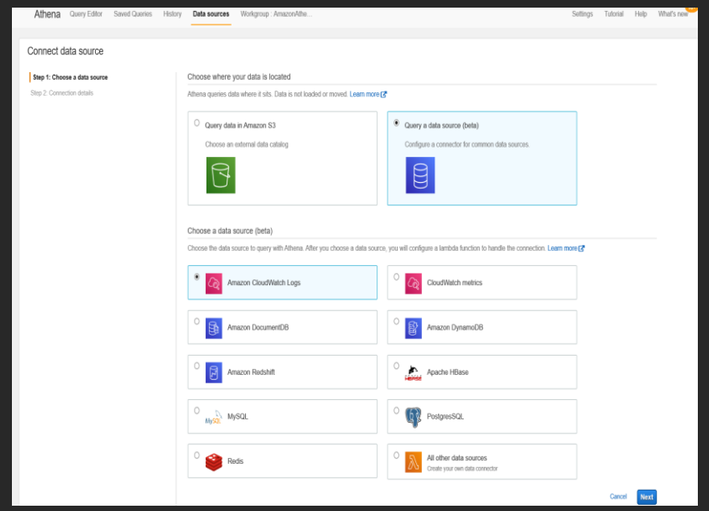
Upload connector to AWS Lambda using Lambda API, UI



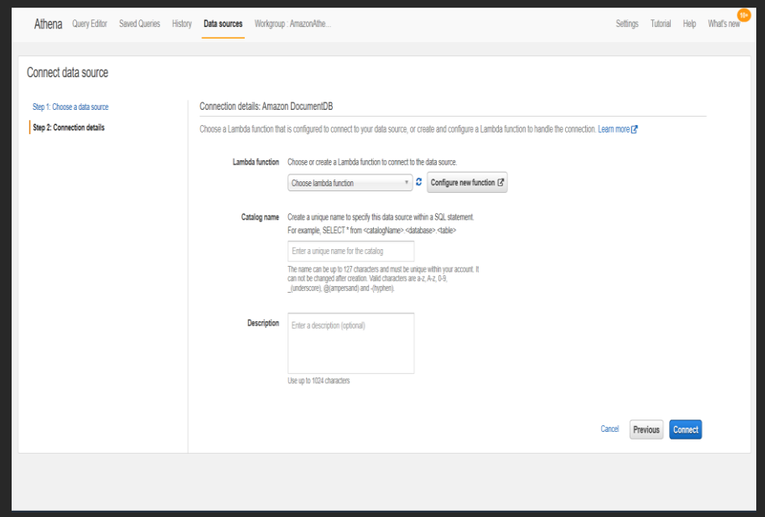
Deploy | Register | Use

Use Athena Console to register connector

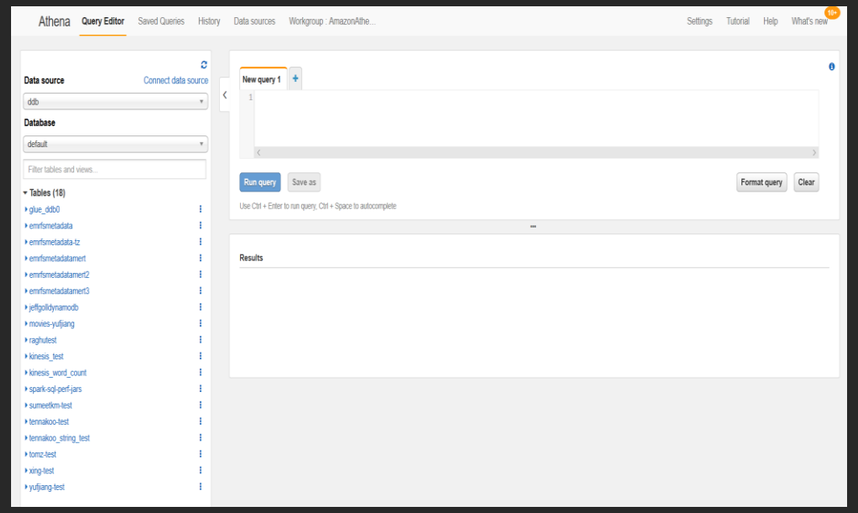
To use an existing data source connector



Discover



Select



Query

Registration-less federated query

- Useful for quick prototyping
- Add the prefix “lambda:<function_name>”. as the catalog name
- Example: “SELECT * from “lambda:cmdb”.e2.ec2_instances” would run a federated query to query our ec2 instance list

Data source connecters available today

- Hbase
  - Parallelizes by region server and supports predicate pushdown.
- DocumentDB
  - On-the-fly schema inference or configure explicit schema using the Glue Data Catalog.
  - Supports predicate pushdown.
- DynamoDB
  - On-the-fly schema inference or configure explicit schema using the Glue Data Catalog.
  - Supports parallel scan and predicate pushdown.
- JDBC
  - Works with Aurora, MySQL, Postgres, and Redshift and supports parallel scans and predicate pushdown.

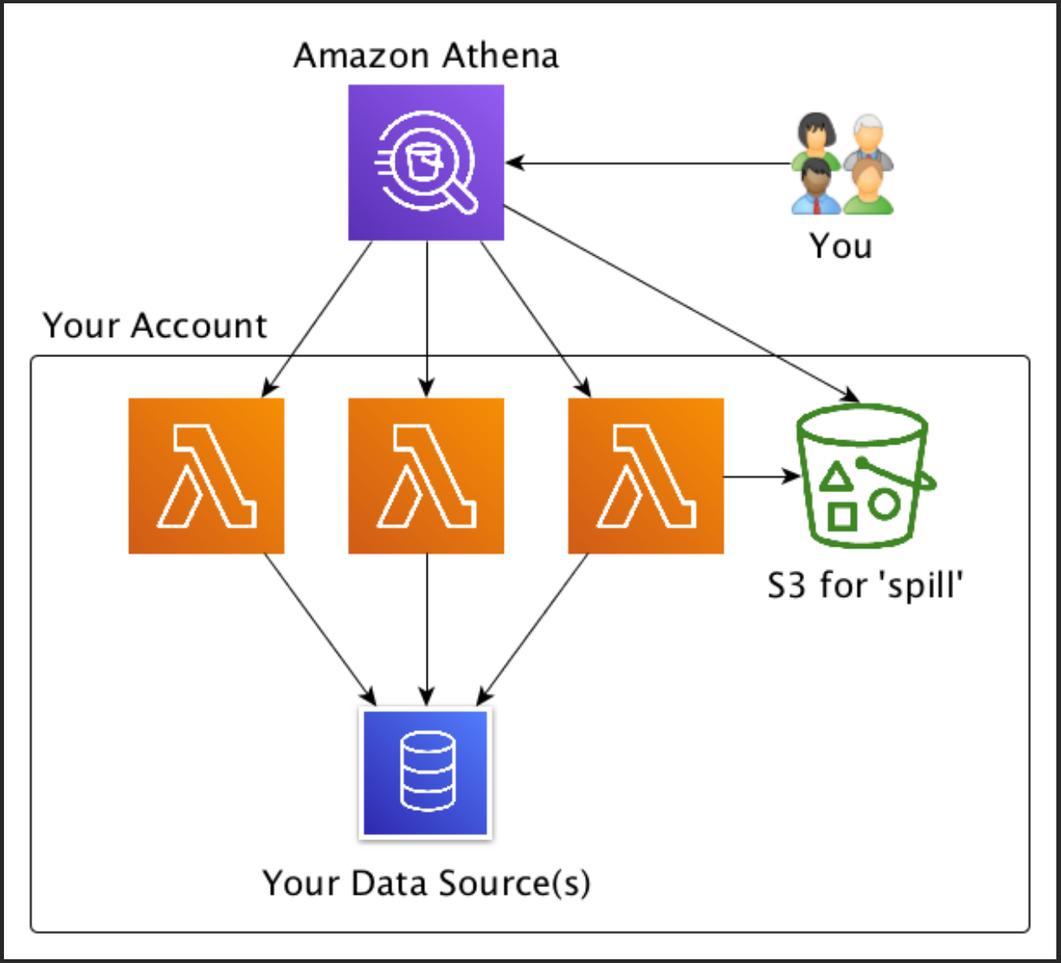
Data source connectors available today (cont'd)

- Redis
  - Use your Redis z-sets, hmaps, or key prefixes to define tables in the Glue Data Catalog and then query them from Athena
- CloudWatch Logs
  - Support parallel scan of log streams, predicate pushdown support, and rich regular expressions
- CloudWatch Metrics
  - Support parallel scan of metric namespaces and dimension as well a predicate pushdown
- TPDS Data Generator
  - Supports parallel scans and predicate pushdown as a reference implementation for building your own connector

Also, build your own data source connector

Use Athena Query Federation SDK and create connector to your own data source

- Features:
  - S3 spill
  - Partition pruning
  - Parallel scans
  - Portable columnar memory-format (Apache Arrow)
  - Authorization
  - Congestion control/avoidance



https://github.com/awslabs/aws-athena-query-federation

Self-service ETL jobs using federated query

1

One SQL query reading data from multiple sources

Output in S3

2

CTAS and INSERT INTO to create tables and convert to optimized format

3

Schedule using Lambda or build applications

https://aws.amazon.com/blogs/big-data/simplify-etl-data-pipelines-using-amazon-athenas-federated-queries-and-user-defined-functions/

Introducing Athena support for Hive Metastore (Preview)

AWS re:Invent

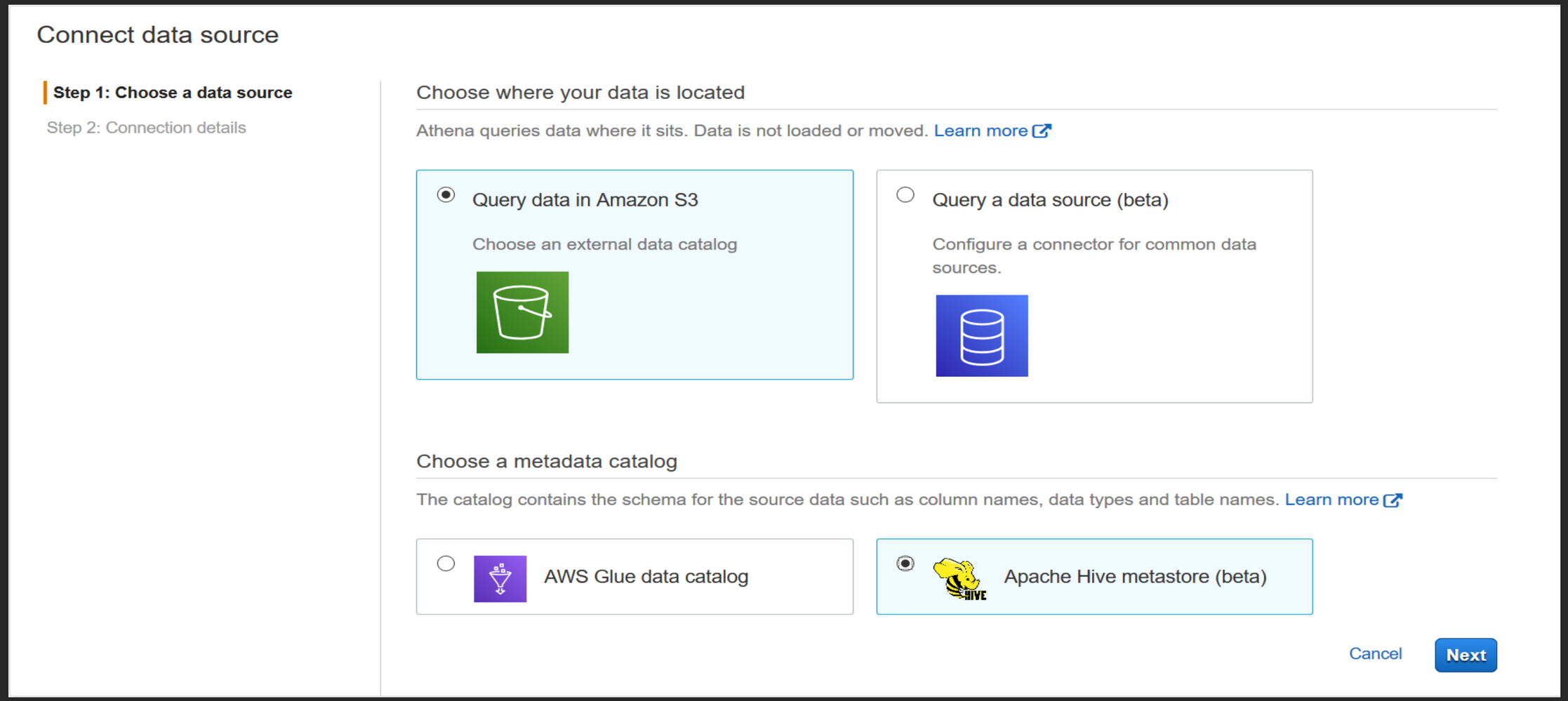
aws

Support for custom metadata store

Use data source connector to connect Athena to any metastore

- Customers using a custom metadata store and not the Glue catalog can now use Athena
- Reference implementation for the Hive Metastore provided
- Run query that scans data across Hive Metastore, Glue catalog, or any other federated data source

Connect Athena to your Hive Metastore



Deploy | Register | Use

How does federated query help our personas

Ana – the Analyst

Carlos – the Administrator

Richard – the Engineer

Maria – the Scientist

SELECT data from any data source for quick analysis

Serverless ETL in one SQL query

Does not need to learn different data access paradigms

Use data from any data source to train ML model

Able to create a data driven narrative in real time

Can append to existing table and handles data transforms using CTAS and INSERT INTO

Does not need to scale ingestion pipelines

Quick access to data helps create accurate ML models

Introducing User Defined Functions (UDFs) in Athena (Preview)

AWS re:Invent

aws

What are the challenges without UDFs

- Difficult to pre- or post-process data without UDFs
- Duplication of raw data for access controls to columns
- Learn and use multiple applications for invoking custom code and using SQL queries for analysis

Invoke your own functions in Athena queries

- UDFs powered by AWS Lambda
- Network calls supported
- Invoke UDF in SELECT and/or FILTER phase
- Athena optimizes performance, you focus only on processing logic

UDFs in Athena

Write once

Deploy once

Invoke as many times as needed in a query

https://aws.amazon.com/blogs/big-data/simplify-etl-data-pipelines-using-amazon-athenas-federated-queries-and-user-defined-functions/

Athena UDFs code sample

- Simple to write, deploy, and invoke
- Scalar functions
- Powered by AWS Lambda

Athena Query

```
USING FUNCTION totalprice(quantity int, unitprice DOUBLE)
                RETURN DOUBLE TYPE lambda_udf
        WITH (lambda_udf='ecommerselambdaudf'),
USING FUNCTION isInternational(fullAddress VARCHAR) RETURN BOOLEAN
        TYPE LAMBDA_UDF WITH (lambda_udf='ECommerseLambdaUdf')
SELECT productname,
       productid,
       totalprice(productquantity, unitprice)
FROM   productcatalog
WHERE  isInternational(product.vendor.addr)
```

UDF Lambda Code

```
public class ECommerceLambdaUdfHandler extends ScalarUdfHandler {

    public double totalPrice(int quantity, double unitPrice) {
        return quantity * unitPrice;
    }

    public boolean isInternational(String encryptedAddress) {
        String customerAddr = cipher.decrypt(encryptedAddress);
        return isInternational(customerAddr);
    }
}
```

https://github.com/awslabs/aws-athena-query-federation/tree/master/athena-udfs

How do UDF capabilities help our personas

Ana – the Analyst

Carlos – the Administrator

Richard – the Engineer

Maria – the Scientist

Invoke custom code for quick analysis

No need to duplicate data for access control

Build and deploy a library of UDFs once for any user of the organization to use

Transform data easily for use in training ML models

No need to shuffle between multiple applications to only run custom code in SQL query

Easily transform data in ETL workflow

Invoke custom code in applications

Apply pre and post processing logic to data or inference result

Introducing ML capabilities in Athena (Preview)

AWS re:Invent

aws

Why do you need ML capabilities in Athena

Number of employees:

SQL proficiency > ML proficiency

SQL proficiency > Python proficiency

SQL proficiency > JAVA proficiency

...

...

Running inference in SQL queries is an advantage

Invoke machine learning models for inference in SQL Queries

- Deploy ML model once on Amazon SageMaker, use n times
- Run inference on data anywhere
- No need to build applications to enable inference
- No additional setup required

Use Athena to train ML model

Federated Athena query to select data from any data source

Transform data using UDFs in Athena

Train and deploy model on Amazon SageMaker

Use Athena to run inference using ML model

Deploy ML model on SageMaker

Write UDF to pre or post process data

Anyone in organization can run inference on data from any data source

Sample ML use-cases

- Find IP addresses associated with suspicious activity in application logs
- Find products with revenue anomalies (+/-)
- Find suspected fraud in transaction records
- Predict whether a proposed new video game would be a hit

https://aws.amazon.com/blogs/big-data/prepare-data-for-model-training-and-invoke-machine-learning-models-with-amazon-athena/

Sample query to invoke inference

```
USING FUNCTION predict(platform int, genre int, critic_score int, user_score int, rating int) returns double TYPE SAGEMAKER_INVOKE_ENDPOINT
WITH (sagemaker_endpoint='xgboost-2019-11-22-00-52-22-742'),

USING FUNCTION normalize_genre(value VARCHAR) RETURNS int TYPE LAMBDA_INVOKE
WITH (lambda_name='VideoNormalization'),

SELECT predict (platform, genre, critic_score, user_score, rating), name
FROM
   (SELECT name,
        normalize_genre(genre) AS genre,
        critic_score,
        user_score,
FROM video_game_data.video_games);
```

How do Athena's ML capabilities help our personas

Ana – the Analyst

Carlos – the Administrator

Richard – the Engineer

Maria – the Scientist

Can easily run inference on data from any data source

No need to duplicate data in multiple formats to support ML use-cases

Can incorporate ML capabilities in SaaS applications without learning new technologies and access patterns

Is no longer an inference bottleneck in the organization

Can incorporate ML derivations in analysis to generate richer reports

Does not need to maintain extra application to enable analysts to run ML inference

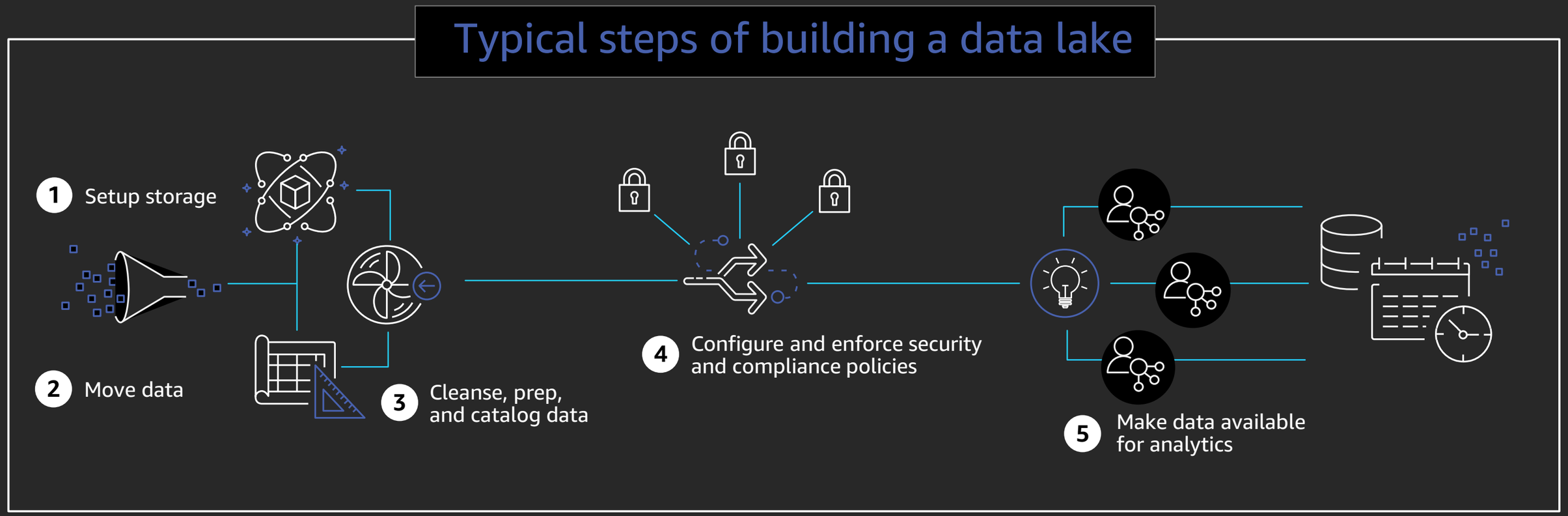
Can focus on creating accurate ML models

Introducing Athena integration with AWS Lake Formation

AWS re:Invent

aws

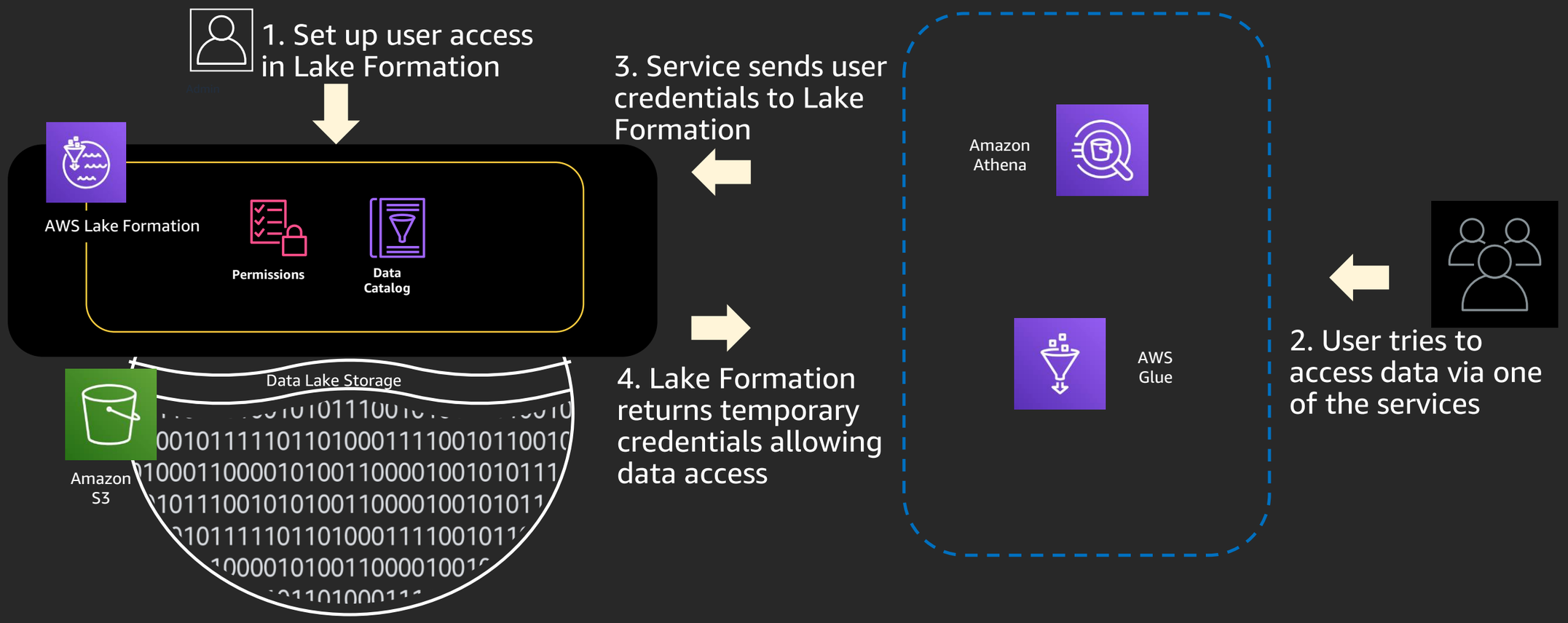
Typical steps in setting up a data lake



How does Athena's Lake Formation integration help

- Specify permission policies centrally in Lake Formation
- Fine-grained access controls
  - Column level permission controls supported
- Real-time audit and monitoring
  - Use Lake Formation APIs or Console to easily audit

Athena user request workflow

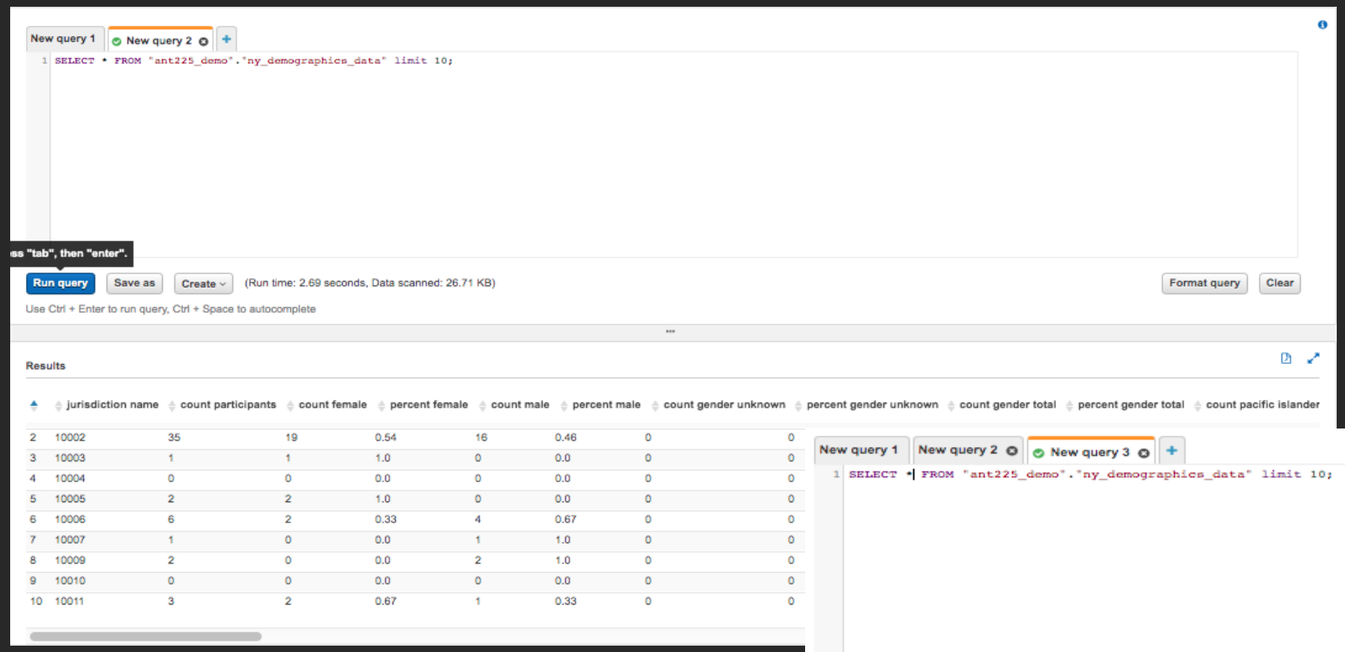


Granting users access to table data in Lake Formation

- analyst_1 (Ana) is granted access to all columns of the table
- analyst_2 (Maria) is granted access to only the *jurisdiction name* column

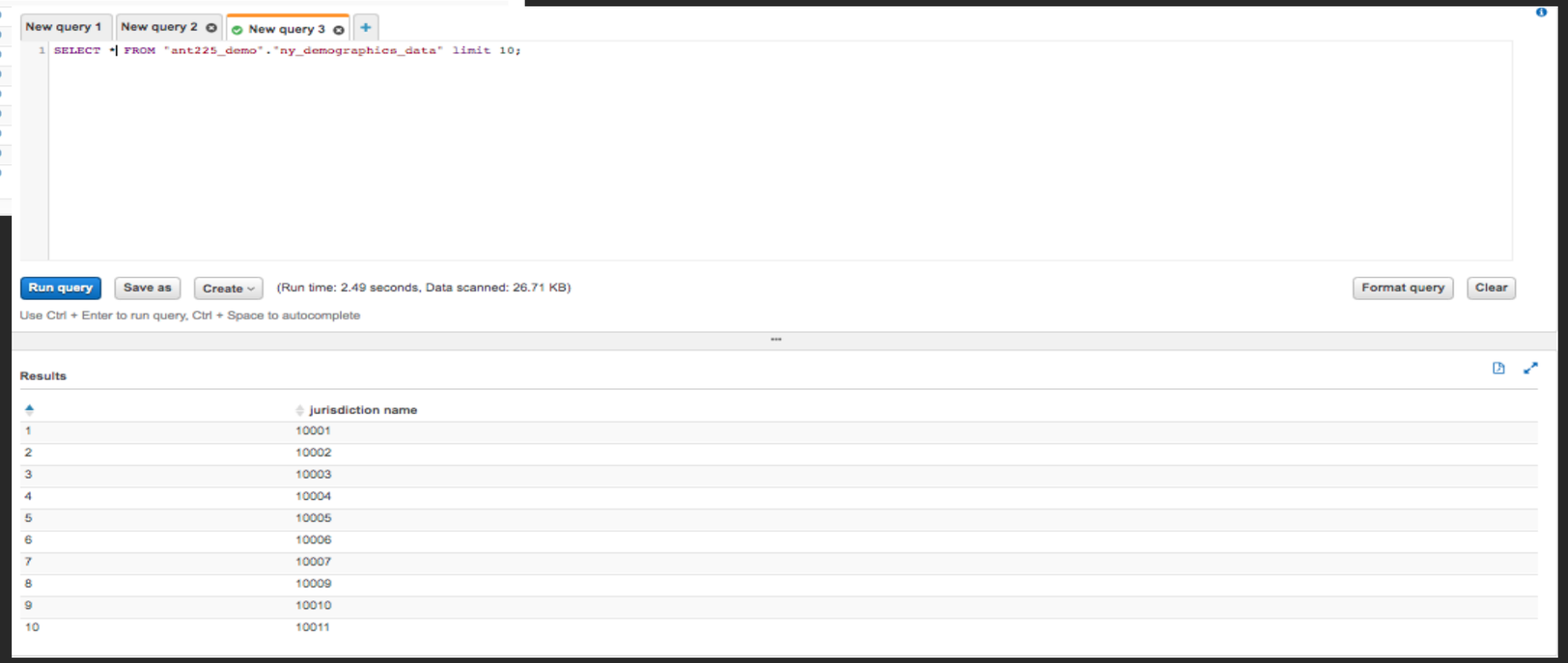
analyst_1	IAM user	Column	ant225_demo.ny_demographics_data.*	Select	-
analyst_2	IAM user	Column	Include: ant225_demo.ny_demographics_data.jurisdiction name	Select	-

Query the table using Amazon Athena



	jurisdiction name	count participants	count female	percent female	count male	percent male	count gender unknown	percent gender unknown	count gender total	percent gender total	count pacific islander
2	10002	35	19	0.54	16	0.46	0				0
3	10003	1	1	1.0	0	0.0	0				0
4	10004	0	0	0.0	0	0.0	0				0
5	10005	2	2	1.0	0	0.0	0				0
6	10006	6	2	0.33	4	0.67	0				0
7	10007	1	0	0.0	1	1.0	0				0
8	10009	2	0	0.0	2	1.0	0				0
9	10010	0	0	0.0	0	0.0	0				0
10	10011	3	2	0.67	1	0.33	0				0

analyst_1 (Ana) is able to retrieve all columns



	jurisdiction name
1	10001
2	10002
3	10003
4	10004
5	10005
6	10006
7	10007
8	10009
9	10010
10	10011

analyst_2 (Maria) only retrieves *jurisdiction name* column

More features !

Many more features added …

- Advanced Geospatial functions *(onboard to preview)*
- Athena Workgroups
- INSERT INTO
- Okta IDP support
- Updated JDBC/ODBC support

…

…

Steps to onboard to preview:

https://aws.amazon.com/athena/faqs/#Preview_features

How do organizations benefit when using Athena

- Choose fit for purpose database strategy
- Athena does not bind you to only proprietary formats
  - Use any metadata store
  - Use any IDP
  - Use any data store
  - Use any data format
- Increase your analytics and ETL velocity by querying data using Athena

Demo

Thank you!

Janak Agarwal
@JanakAgarwal
linkedin.com/in/janakagarwal/

Anthony Virtuoso
@AnthonyVirtuoso
linkedin.com/in/avirtuos/

AWS re:Invent

aws

Please complete the session survey in the mobile app.

AWS re:Invent

aws