AWS re:Invent

ANT342

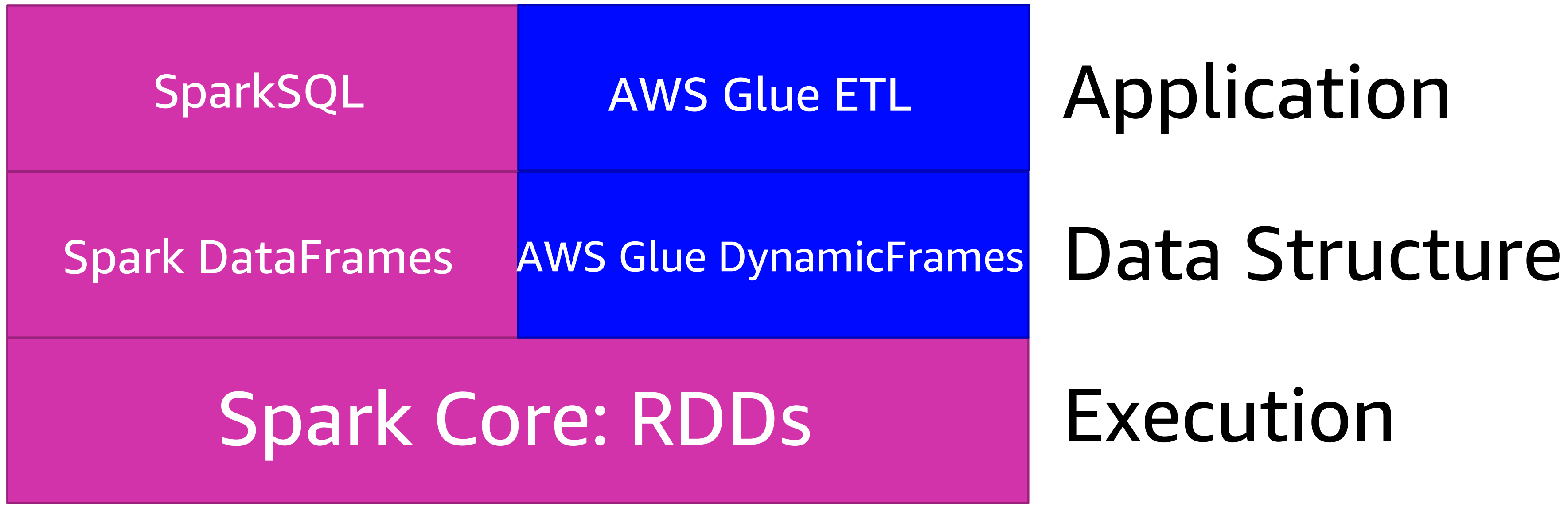
WORKING WITH RELATIONAL DATABASES IN AWS GLUE ETL

Benjamin Sowell
Principal Engineer
AWS Glue

AWS re:Invent

aws

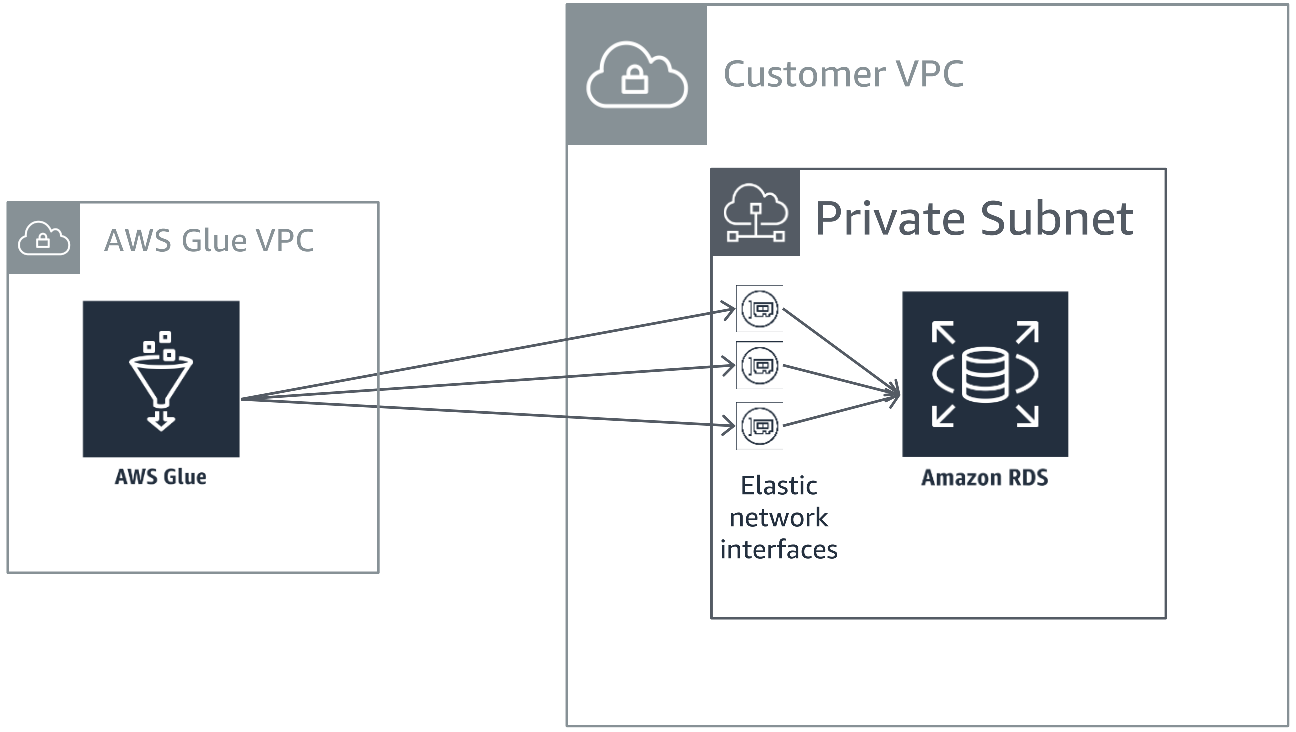
Apache Spark and AWS Glue ETL



- Apache Spark is a distributed data processing engine with rich support for complex analytics.
- AWS Glue builds on the Apache Spark runtime to offer ETL specific functionality.

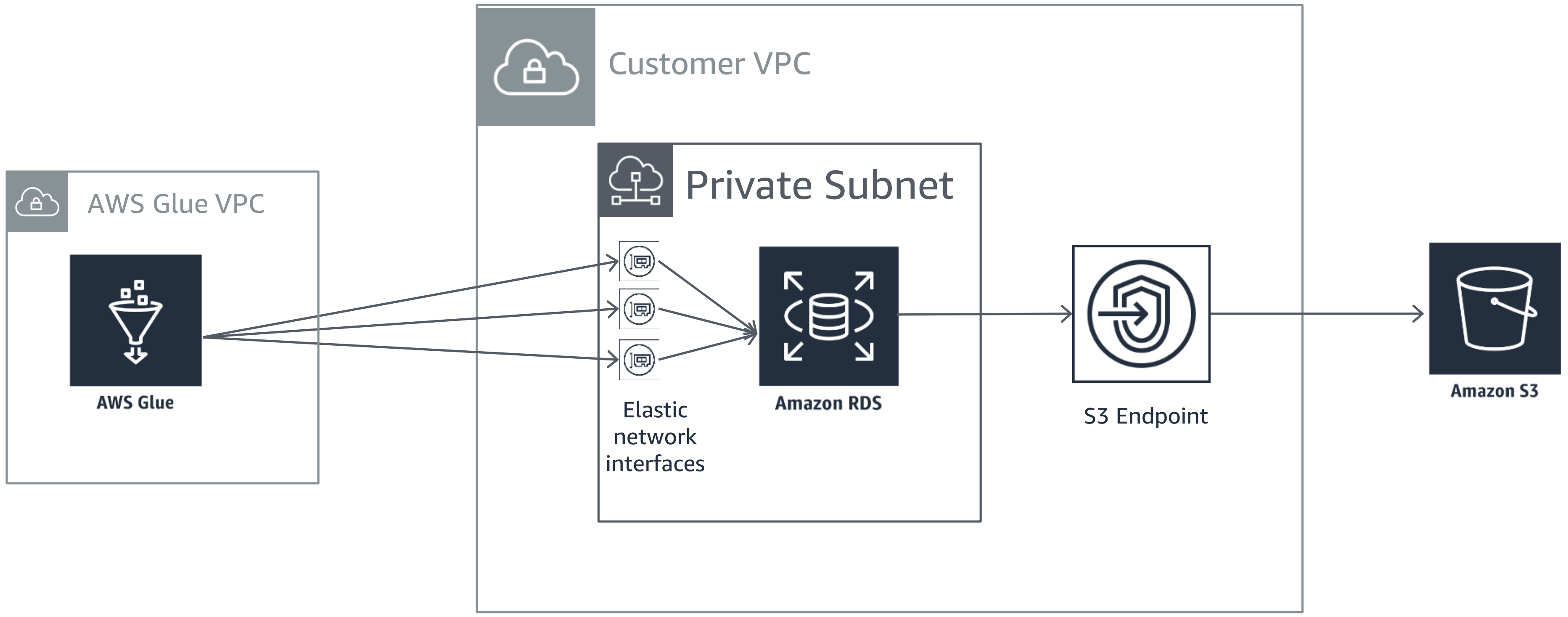
Database Connectivity in AWS Glue

- **AWS Glue connects to your Amazon Virtual Private Cloud (Amazon VPC) by creating an ENI:**



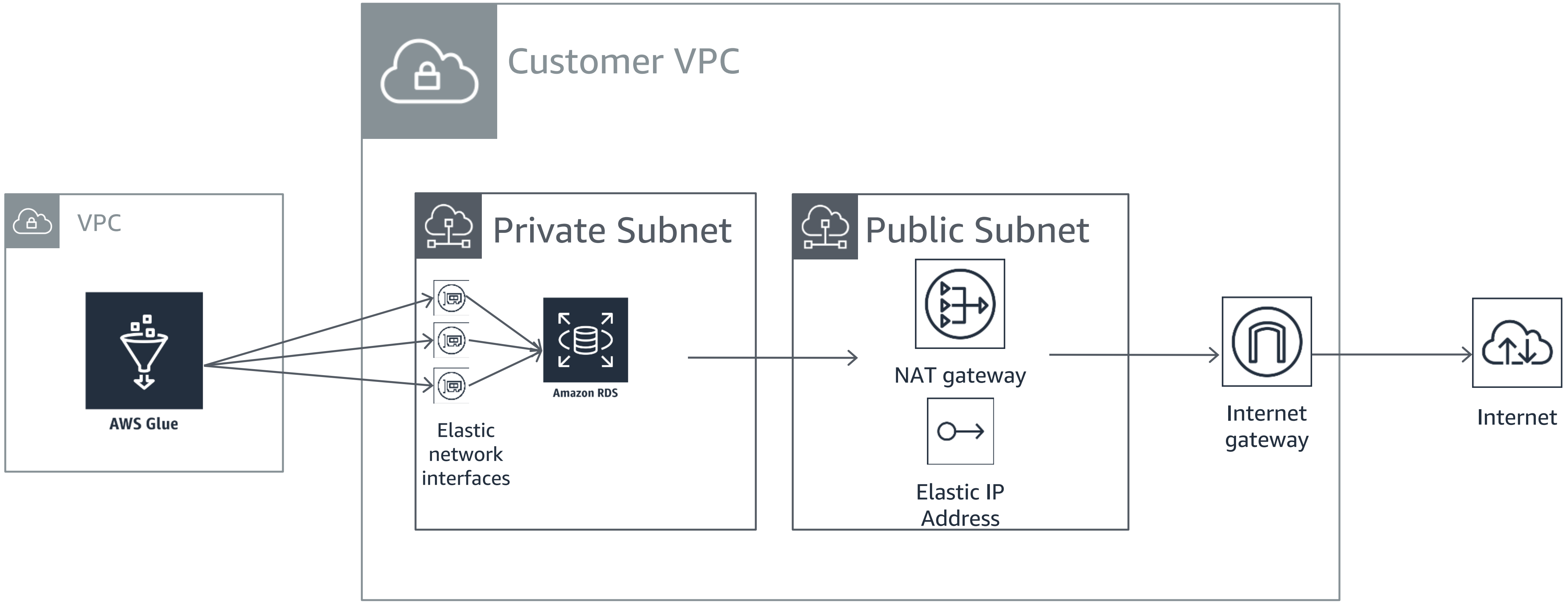
Database Connectivity in AWS Glue

- Two ways to provide Amazon S3 and/or internet connectivity:



Database Connectivity in AWS Glue

- **Two ways to provide Amazon S3 and/or internet connectivity:**

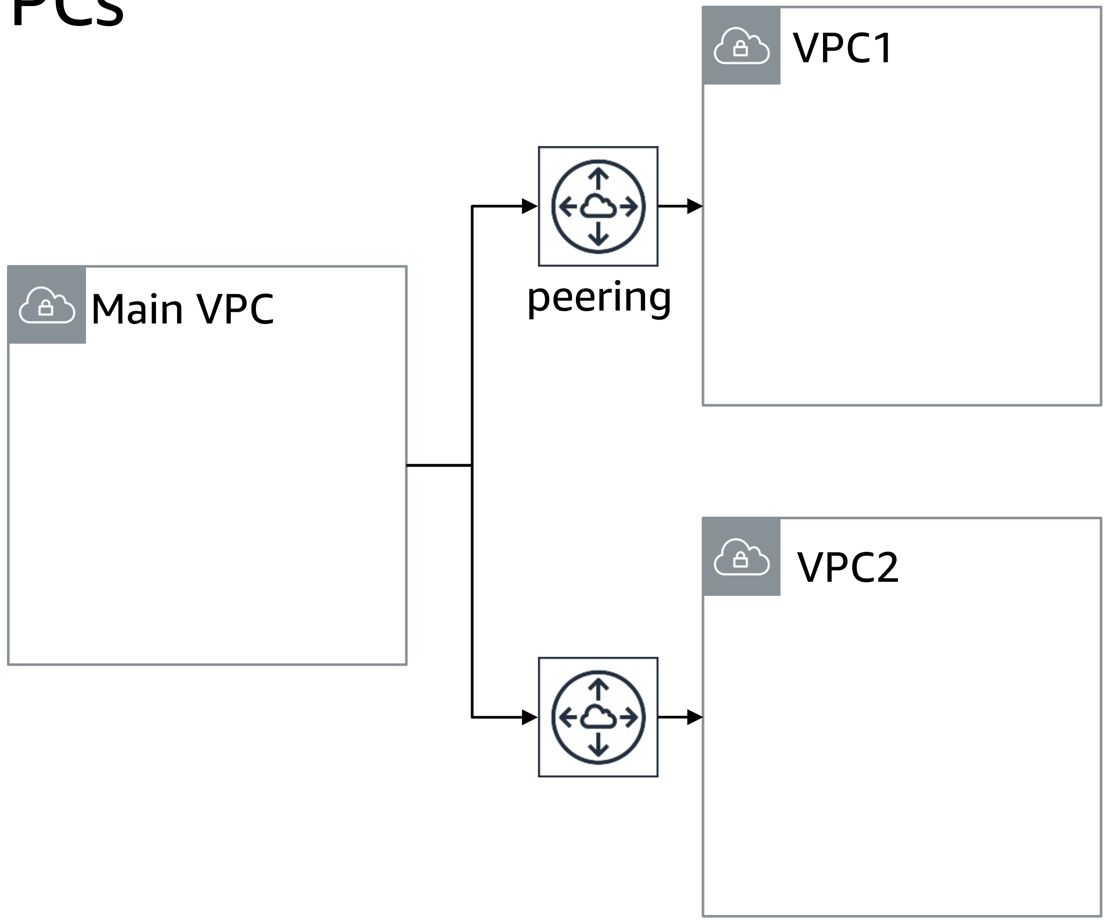


Database Connectivity in AWS Glue

- Your VPC must have both *DNS resolution* and *DNS hostnames* enabled.
- Your database must be inside a security group with a *self-referencing* inbound rule.

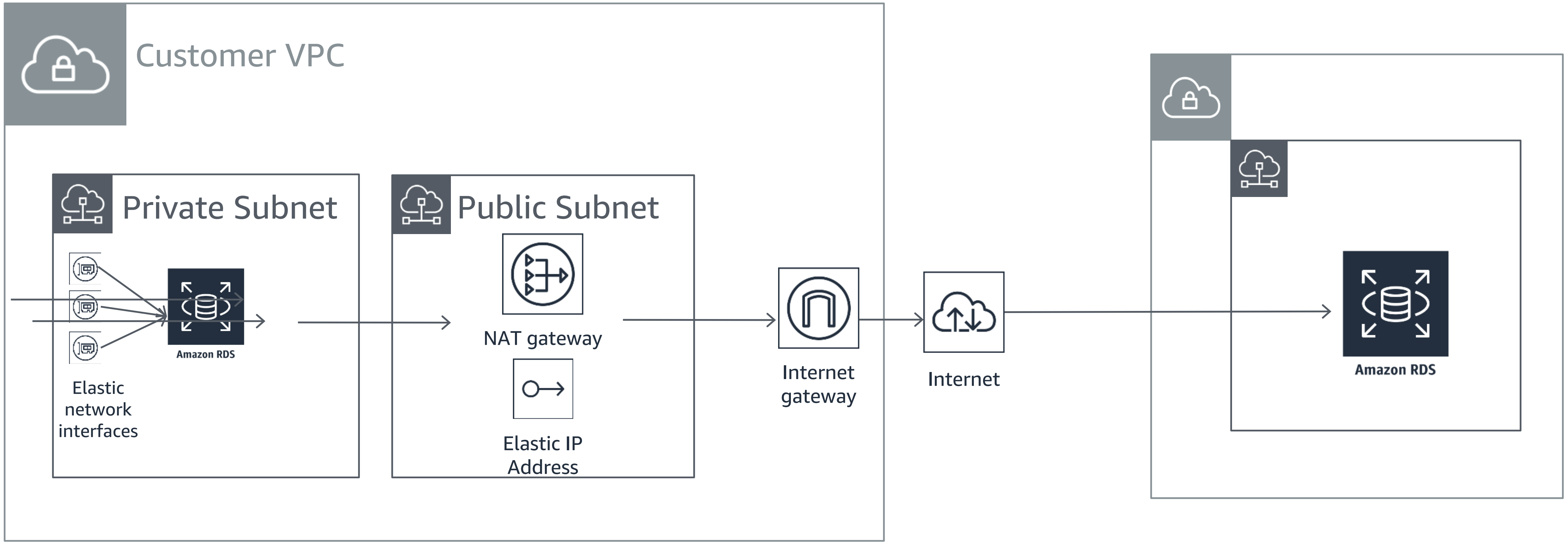
More complex connectivity scenarios

- Multiple VPCs

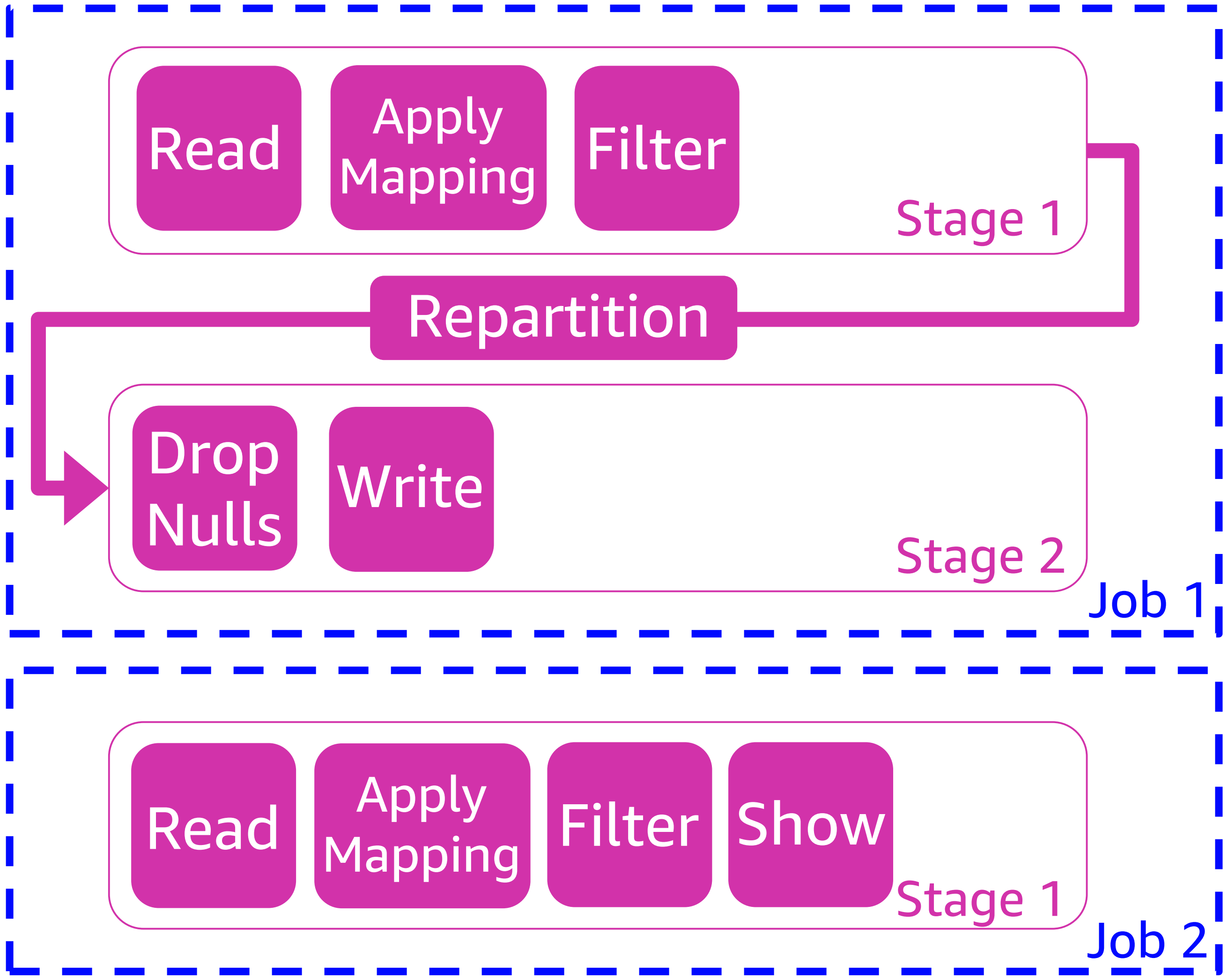


More complex connectivity scenarios

- Cross region access. Security group must allow ingress from NAT EIP.

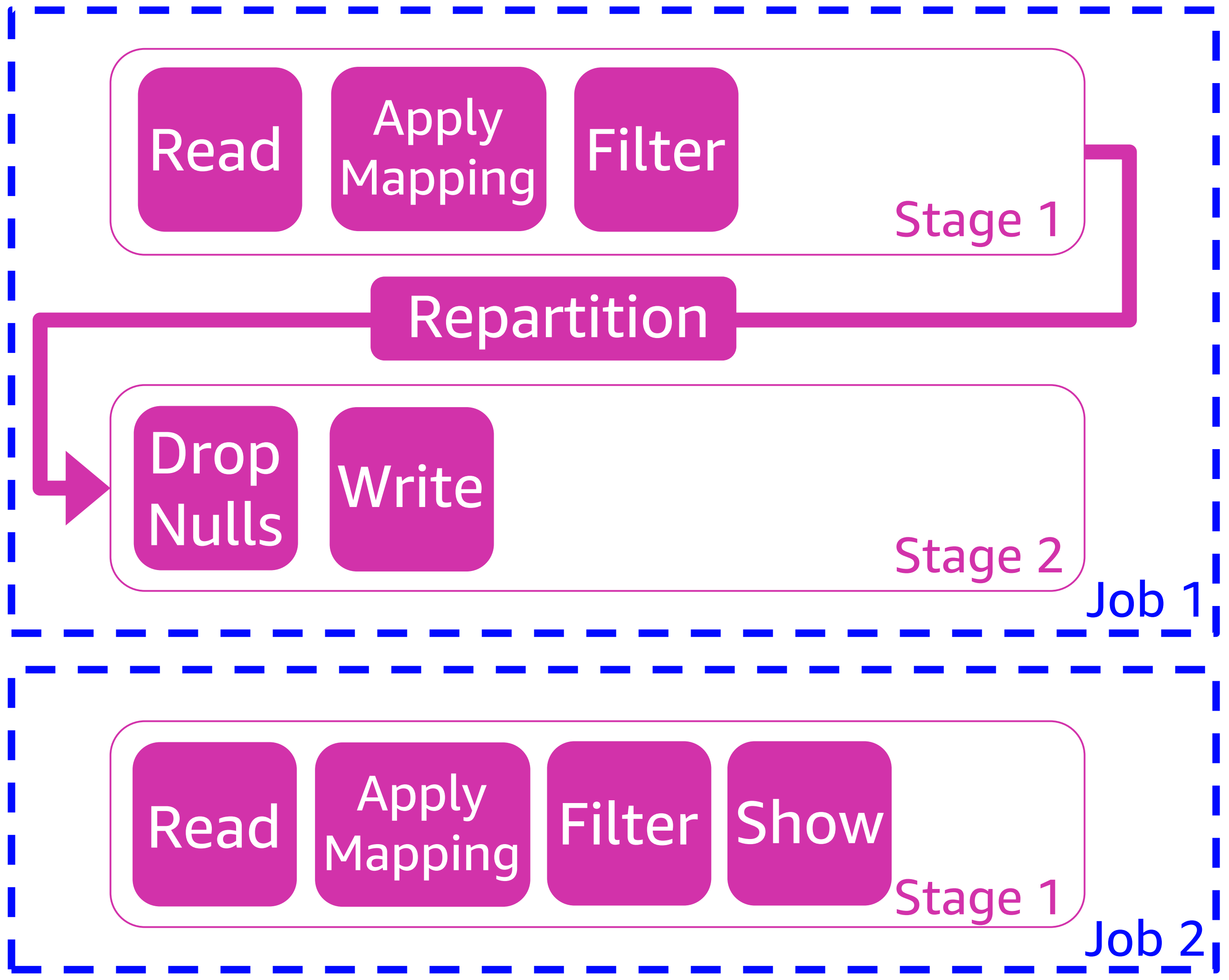


AWS Glue execution model: jobs and stages



```
df = glueContext.getSource(…)

applyMapping = df.applyMapping(…)

filter = applyMapping.filter(…)

repartition = filter.repartition(10)

dropNulls = repartition.dropNulls()

glueContext.getSink(…)\
    .WriteDynamicFrame(dropNulls)


filter.show()
```

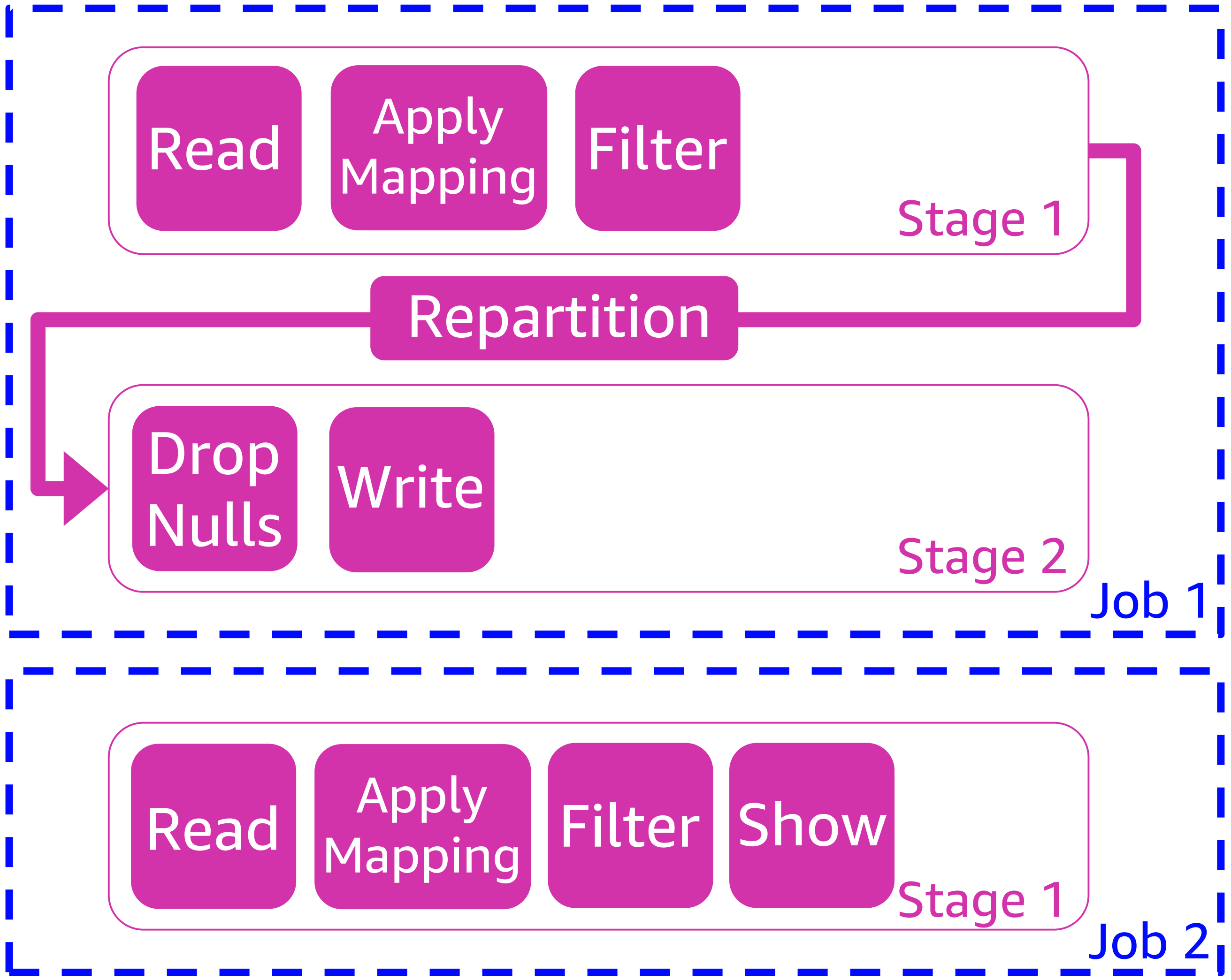
AWS Glue execution model: jobs and stages



```
df = glueContext.getSource(…)

applyMapping = df.applyMapping(…)

filter = applyMapping.filter(…)

repartition = filter.repartition(10)

dropNulls = repartition.dropNulls()

glueContext.getSink(…)\
   .writeDynamicFrame(dropNulls)

filter.show()
```

Actions

AWS Glue execution model: jobs and stages

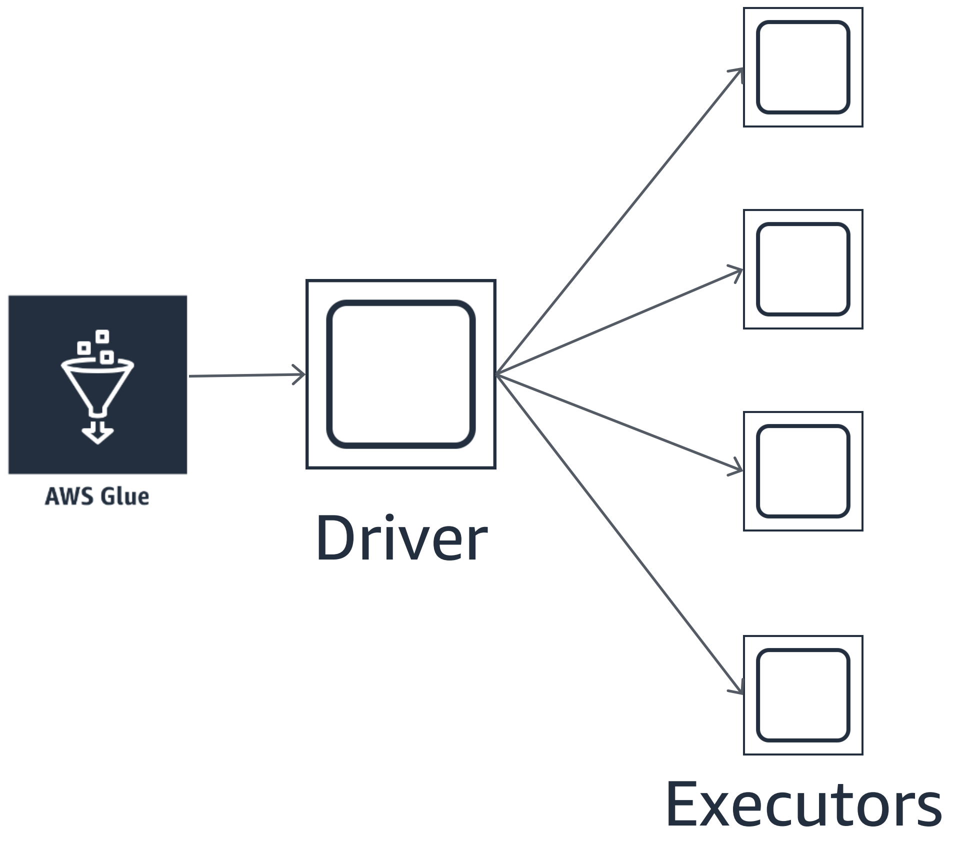


```
df = glueContext.getSource(…)

applyMapping = df.applyMapping(…)

filter = applyMapping.filter(…)

repartition = filter.repartition(10)

dropNulls = repartition.dropNulls()

glueContext.getSink(…)\
    .writeDynamicFrame(dropNulls)
```

```
filter.show()
```

Jobs

AWS Glue execution model



Apache Spark and AWS Glue are *data parallel*. Data is divided into *partitions* (shards) that are processed concurrently.

Jobs are divided into *stages*

1 stage x 1 partition = 1 *task*

Driver schedules tasks on *executors.*
2 executors per DPU

Overall throughput is limited by the number of partitions (shards)

AWS Glue performance: key questions

How is your application divided into jobs and stages?

How is your dataset partitioned?

AWS re:Invent

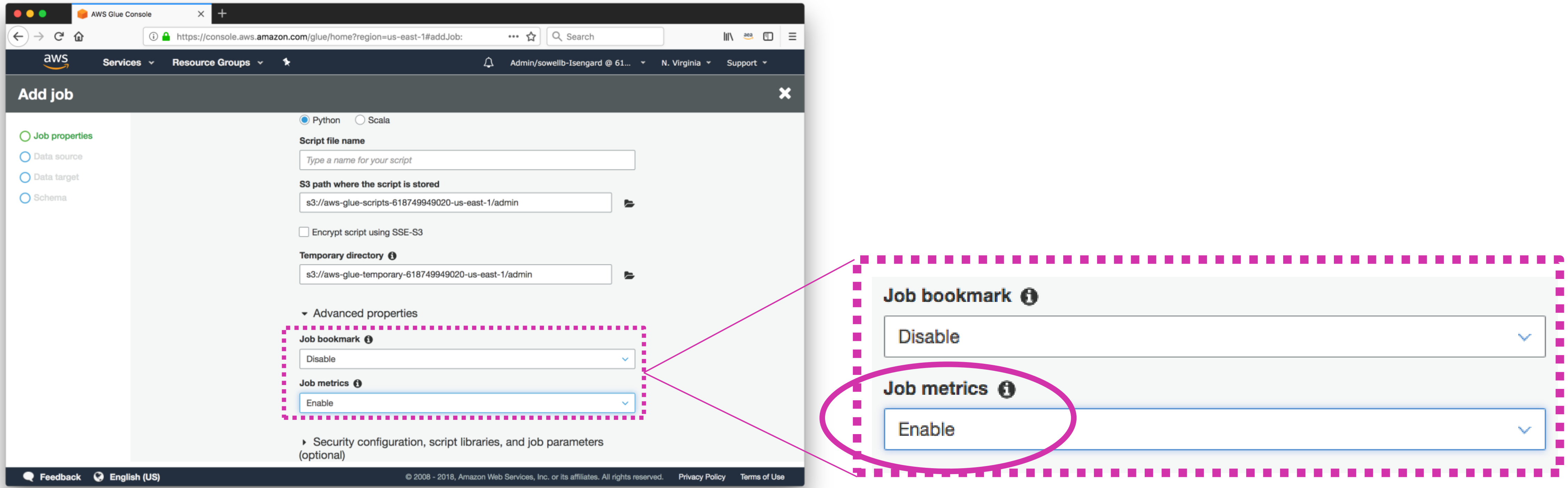
AWS Glue file-based partitions

- For file-based sources, AWS Glue creates a partition for each input file.
- When possible, AWS Glue will *split* large files into multiple partitions.
- When there are many small files, AWS Glue will *group* multiple files into each partition.
- Amazon Redshift sources behave like file sources.
  - AWS Glue uses the Amazon Redshift UNLOAD command to copy the data to Amazon S3 in parallel and then read the data from Amazon S3.
  - For Amazon Redshift sinks, AWS Glue uses the COPY command for parallel loads.

AWS Glue JDBC partitions

- For JDBC sources, by default each table is read as a single partition.
- AWS Glue automatically partitions datasets with fewer than 10 partitions after the data has been loaded.
  - This forces a shuffle operation and can sometimes be quite expensive.
- Example: Let's look at the publicly available NYC Taxi dataset that shows taxi rides over the period of one month.
- We will use AWS Glue *job metrics* to analyze the behavior.

Enabling job metrics



AWS Glue Console

https://console.aws.amazon.com/glue/home?region=us-east-1#addJob:

Add job

Python Scala

Script file name

Type a name for your script

S3 path where the script is stored

s3://aws-glue-scripts-618749949020-us-east-1/admin

Encrypt script using SSE-S3

Temporary directory

s3://aws-glue-temporary-618749949020-us-east-1/admin

Advanced properties

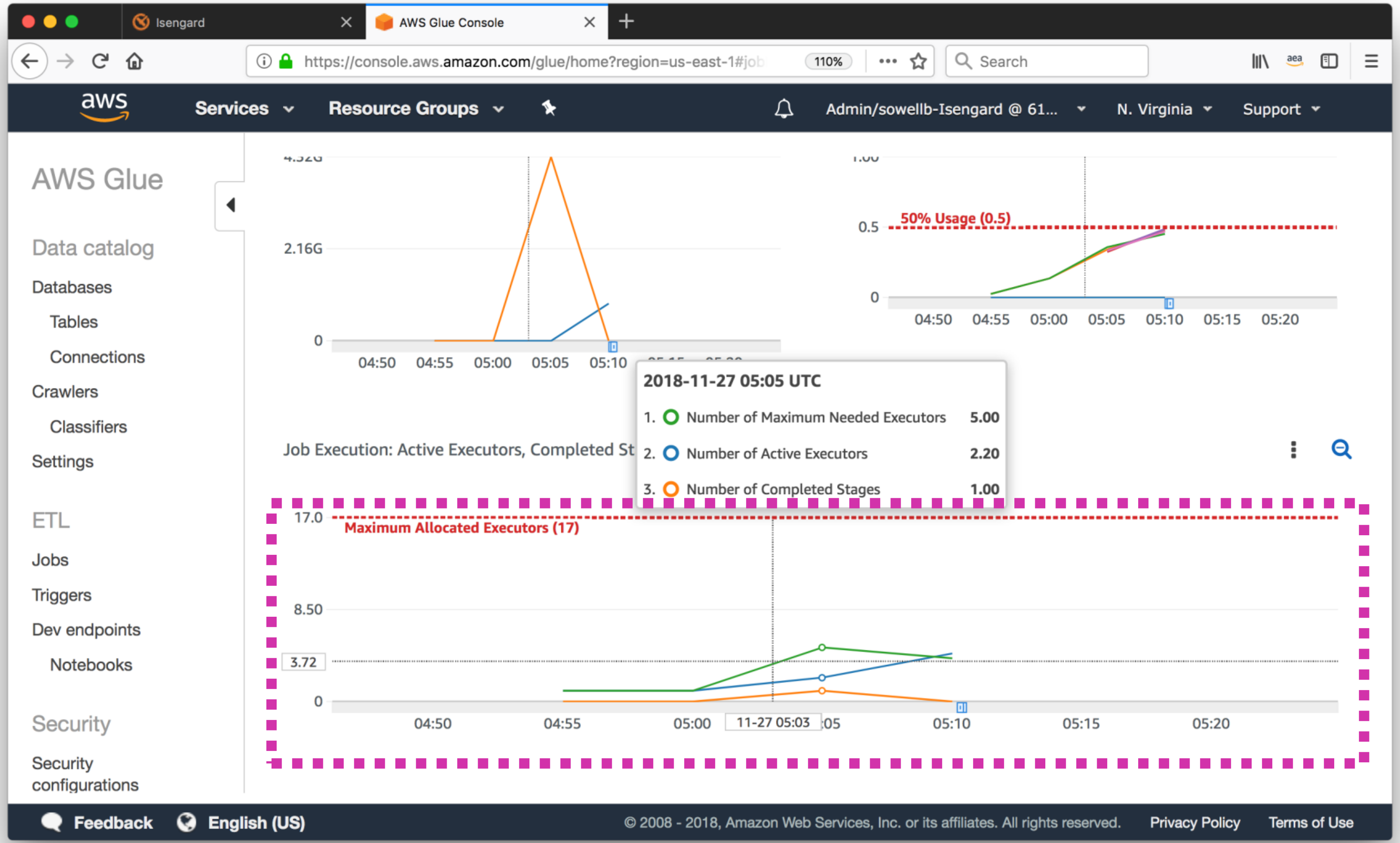
Job bookmark: Disable

Job metrics: Enable

Security configuration, script libraries, and job parameters (optional)

- Metrics can be enabled in the CLI/SDK by passing `--enable-metrics` as a job parameter key.

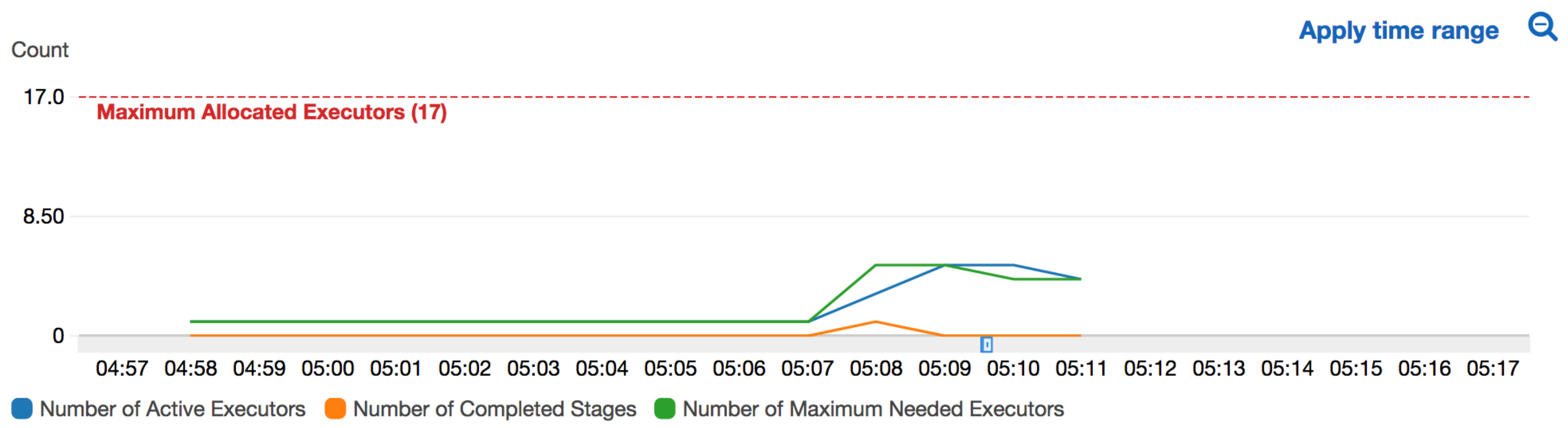
Reading JDBC partitions



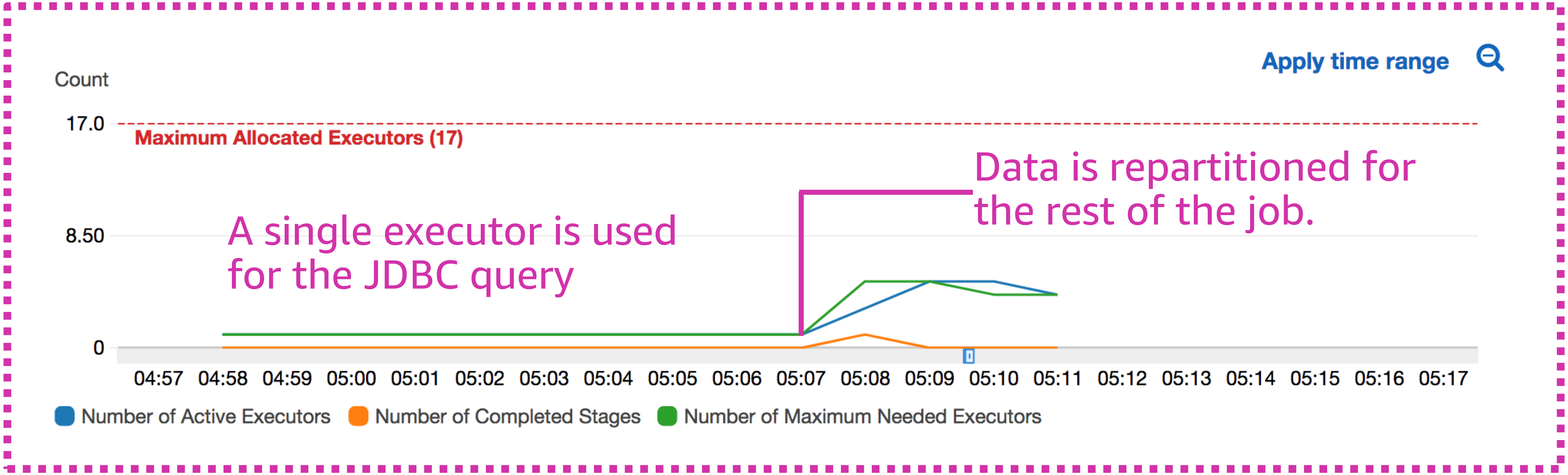
2018-11-27 05:05 UTC

1.	Number of Maximum Needed Executors	5.00
2.	Number of Active Executors	2.20
3.	Number of Completed Stages	1.00

Reading JDBC partitions



Reading JDBC partitions



Options for reading database tables in parallel

- *hashfield* – Single column to use for distribution.
- *hashexpression* – Integer expression to use for distribution.
- *hashpartitions* – Number of parallel queries to make. Default is 7.

- Turns into a collection of queries of the form

```
SELECT *
FROM <table>
WHERE <hashexpression> % <num_partitions> = <partition>
```

- These queries will be executed *concurrently*, but may not run in parallel depending on the database setup.

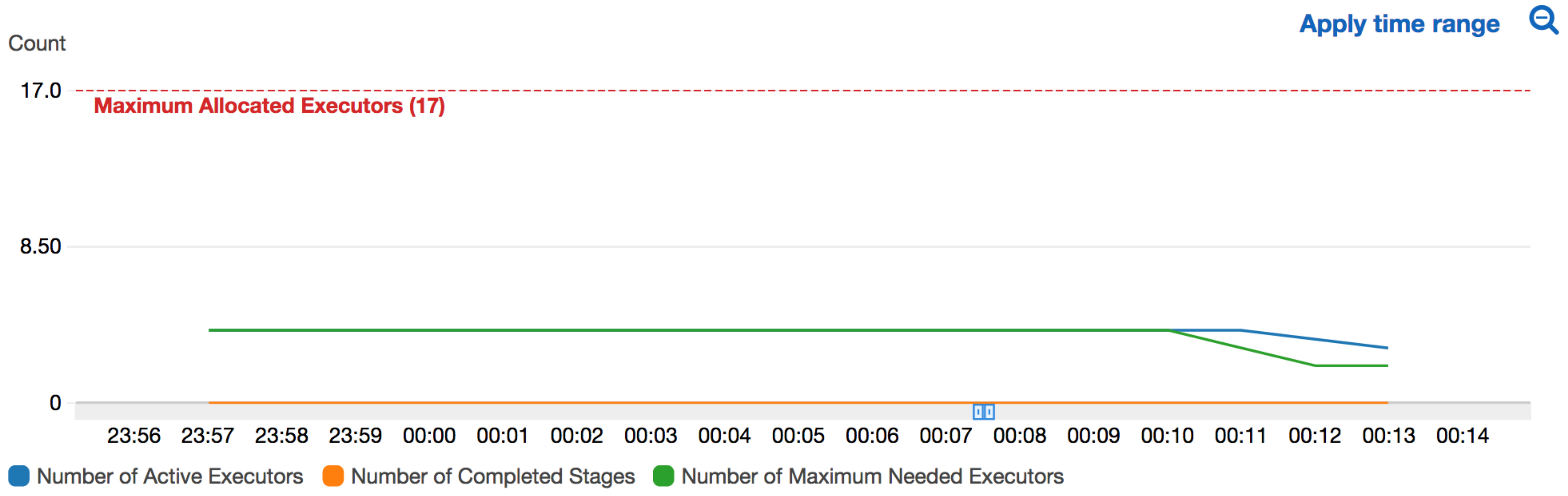
Options for reading database tables in parallel

- Guidelines for picking distribution keys.
  - For *hashfield*, choose a column that is evenly distributed across values. A primary key works well.
  - If no such field exists, use *hashexpression* to define one.
- Example: The taxi dataset does not have a primary key, so we set hashexpression to partition based on day of the month:

```
day(lpep_pickup_datetime)
```

```
datasource0 = glueContext.create_dynamic_frame.from_catalog(
    database = "nyctaxi",
    table_name = "green-mysql-large",
    additional_options={'hashexpression': 'day(lpep_pickup_datetime)',
                        'hashpartitions': 15})
```

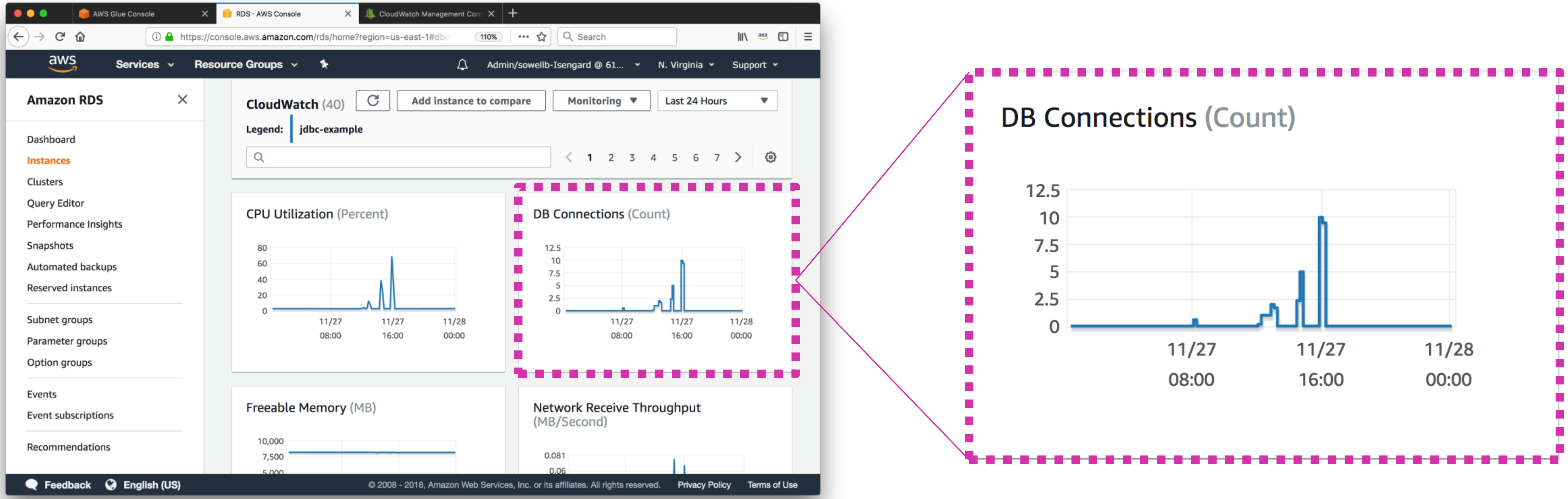
Options for reading database tables in parallel



- Four executors can process 16 partitions concurrently.

Options for reading database tables in parallel

- Make sure to understand impact to database engine.



Job Bookmarks for JDBC Queries

- Job bookmarks safely store which records have been processed so that only new data is read.
- Job bookmarks only work when the source table has an *ordered primary key*.
- Updates are not handled today.

Thank you!

Benjamin Sowell

AWS re:Invent

aws

Please complete the session survey in the mobile app.