Windjammer EMR Accelerator

Deployment and POC Guide

For AWS EMR Spark

Windjammer EMR Accelerator: Guide to Deployment and A/B Benchmarking POCs

1.0 Introduction

1.1Contents

This document describes the procedures to deploy EMR6.X + Windjammer's EMR Spark Accelerator clusters and perform A/B benchmark POCs.

Section 2 of this guide describes EMR Accelerator deployment, management, and A/B benchmarking

- Section 2.1 summarizes the Windjammer Cloud release bucket and additions for EMR cluster creation
- Section 2.2 provides step by step instructions for using the EMR console to create EMR spark clusters deployed with EMR Accelerator and for performing controlled A/B experiments by adding a custom bootstrap action and custom JAR steps
- Section 2.3 describes using Windjammer Cloud customizable tools for easy deployment and management of EMR Spark clusters with Windjammer EMR Accelerator and for performing controlled A/B benchmarking experiments
- Section 2.4 describes modifying your current cluster management scripts with required EMR Accelerator-specific extensions using Windjammer's tools as examples.

Section 3 of this guide describes best practices in configuration, experimental setup, and obtaining support.

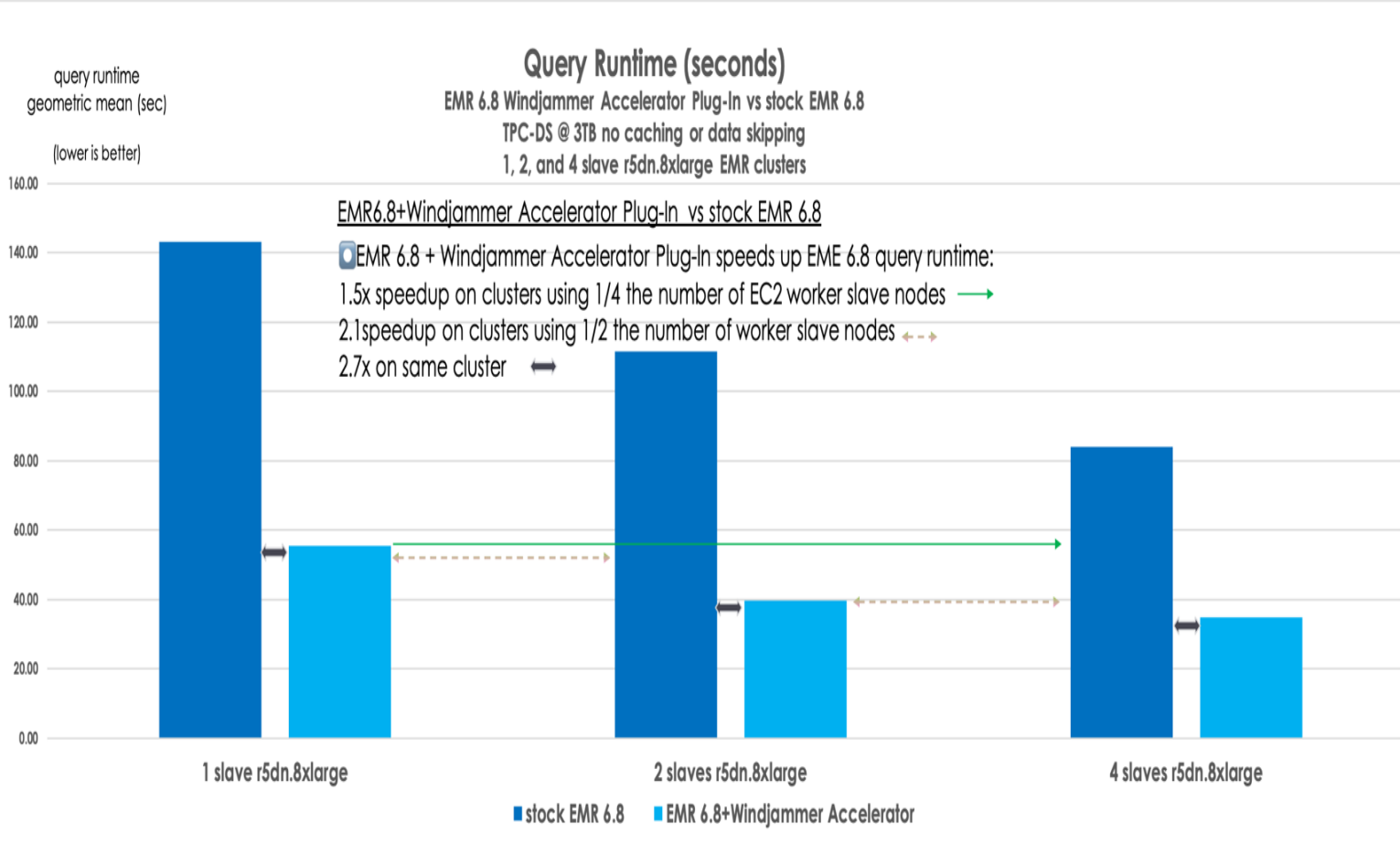
1.2 Overview of Windjammer EMR Accelerator Plug In

EMR Accelerator architecture, design, and benchmarking results are discussed in detail in the Windjammer technical whitepaper, click Technical White Paper to view. Key elements of Windjammer's EMR Accelerator include:

- Native query execution
    - Slashes CPU usage to ¼ of stock EMR6.x to accelerate queries and eliminate server sprawl, eliminates JVM overheads and query runtime variances
- Massively Parallel Processing (MPP) data flow query execution across the compute cluster
- Precise, parallel, asynchronous S3 prefetching for Data Lake access optimization with optional data caching
    - Provides 3x utilization vs stock EMR 6.X of each compute node's S3 bandwidth to drive MPP data flow pipelines
- S3-based efficient and automatic query checkpointing and fault recovery
    - Enables broad query fault tolerance (including cluster and spot instance interruptions and auto-scale events) using fully disaggregated compute nodes and cloud storage
- Transparent plug-in native execution engine for AWS EMR 6.x
    - Transparently accelerates and cost reduces Spark workload deployments with 100% compatibility while increasing fault tolerance and predictability

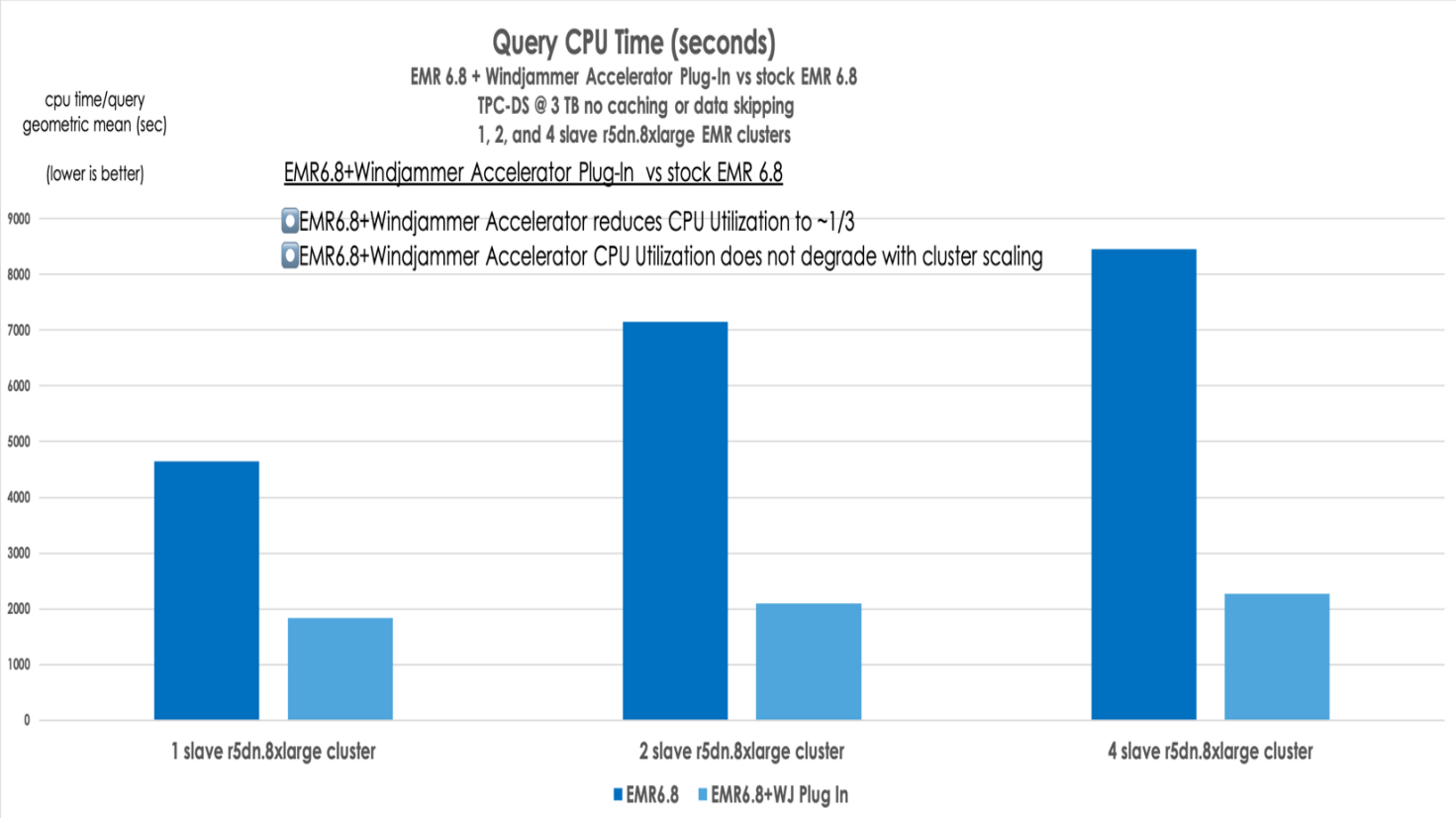
The following charts show the increased performance and the cost savings realized with Windjammer's EMR Accelerator Plug-In relative to stock EMR 6.8 running the TPC-DS benchmark at 3TB scale without caching or data skipping. The detailed TPC-DS measurement data used for these charts is posted at EMR 6.8+Windjammer Accelerator Plug-In Measurements and you can validate the data with A/B experiments in your own clusters as discussed in below this Windjammer EMR Accelerator Deployment/POC Guide. Note that balanced clusters using other EC2 instance types and cluster sizes yield similar TPC-DS acceleration and cost reduction results at 3 TB, 10TB, and higher scale. All versions of EMR 6.x achieve similar benefits with Windjammer Accelerator.

The Query Runtime chart below shows both the Query Speedup and the Reduction of the Number of Required Spark Worker EC2 Slave Nodes in clusters running EMR6.8 with the Windjammer Accelerator Plug-In relative to stock EMR 6.8 Spark clusters using r5dn.8xlarge (32 vCPU, 25Gbps networking) EC2 instances with 1,2, and 4 slave nodes. EMR 6.8 clusters with the Windjammer Accelerator Plug-In relative to stock EMR 6.8 clusters provide 1.5x query runtime speedup on clusters using one quarter the number of EC2 worker slave, 2.1x speedup on clusters using half the number of nodes, and 2.7x speed-up on the same clusters.

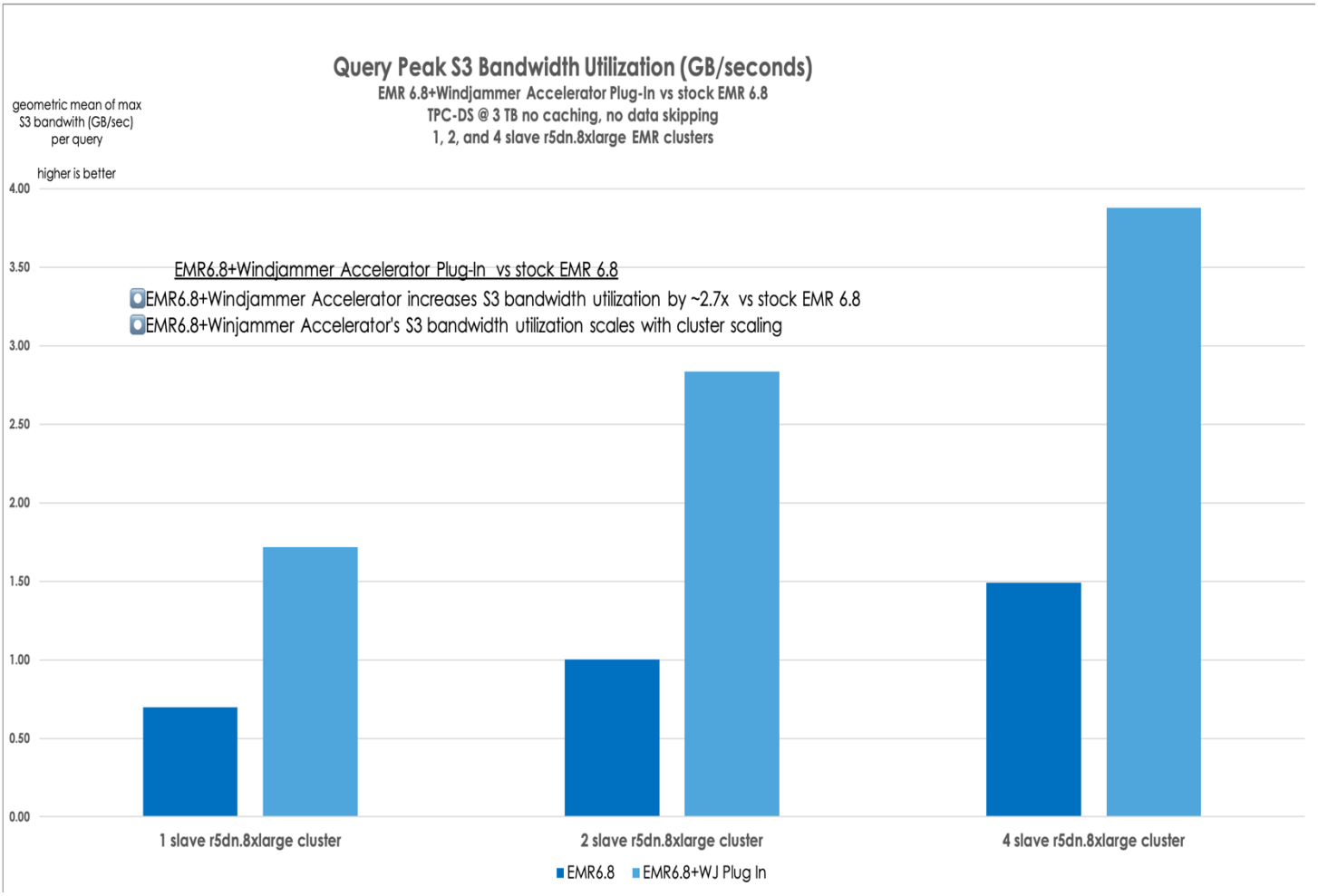


Generally, deployments of EMR6.X+Windjammer Accelerator Plug-In require much smaller cluster sizes than stock EMR6.X, resulting in both significant cost savings and performance acceleration.

The Query CPU Utilization chart below shows the reduced CPU usage of EMR6.8 +WJ Plug-In relative to stock EMR 6.8. Clusters running EMR6.X+WJ Plug-In reduce per query CPU usage to ~1/3 of stock EMR 6.X clusters and EMR6.X+WJ Plug-In query CPU utilization remains flat as a function of cluster size whereas EMR6.x CPU overhead per query grows with cluster size. Note that deployments of EMR6.X+WindjammerAccelerator Plug-In should utilize EC2 instance types with sufficient network bandwidth (approximately 1Gbps/vCPU) to achieve balanced slave instances with high core utilization.



The Query S3 Bandwidth Utilization chart below shows the Increased S3 Bandwidth Utilization of EMR6.8+WJ-Plug-In clusters relative to stock EMR 6.8 clusters. Note that EMR6.X+Windjammer Accelerator Plug-In clusters utilizes ~3x the S3 bandwidth per query relative to stock EMR6.X clusters. Deployments of EMR6.X+Windjammer Plug-In Accelerator should utilize EC2 instance types with sufficient network bandwidth (approximately 1Gbps/vCPU) to achieved balanced EC2 instances.



2.0 Deployment

2.1 Windjammer EMR Accelerator Cloud Bucket

When you subscribe to EMR Accelerator in the AWS Marketplace, your AWS account automatically gets access to Windjammer's AWS Cloud. This enables you to you deploy Windjammer's EMR accelerator into EMR 6.x Spark clusters in your AWS account.

Windjammer EMR Accelerator is deployed from an S3 bucket in Windjammer's AWS Cloud named s3://wjm-build-1-5. Account access to the release bucket is provided automatically during AWS marketplace subscription via a policy on the bucket. The contents of the release bucket include:

- bootstrap used by EMR for cluster standup
- main.tgz most WJ cluster software, installed by bootstrap
- wjm-prep.tgz WJ software installed after cluster initialization
- wjm-sperf.tgz WJ performance measurement and reporting tools
- emrtools.tgz WJ tools to create and drive cluster activity

Additions for EMR console cluster creation or an AWS EMR cluster creation command

- cluster bootstrap
- EMR bucket WJM_DATASETS for TPC-DS datasets
- EMR Step to install windjammer JAR
- EMR steps to run A/B benchmark POCs of TPC-DS queries or customer-specific queries

Notable directories on the master node of the deployed cluster:

- /opt/wjm root of WJ software
- /home/sne owner of WJ software
- /opt/wjm/bin executables
- /opt/wjm/queries TPCDS queries and customer queries source

Windjammer software on the cluster is owned and executed by the non–privileged user `sne`. This user is granted permission to reach other nodes in the cluster via SSH using its own public key (stored locally).

During cluster operation, EMR Accelerator dynamically collects into a protected bucket in Windjammer Cloud detailed cluster and query measurement data, usage records, logs, and query checkpoints to enable dynamic optimization of your cluster's performance and transparent query fault tolerance.

Windjammer Cloud also provides tools for controlled A/B benchmark POCs on your cluster using the standard TPC DS benchmark and your own queries and datasets. TPCDS and customer-specific queries can be evaluated by EMR Spark with EMR acceleration, by stock EMR Spark alone, or both modes sequentially. Detailed performance measurement reports are generated.

2.2 Choice of EC2 Instance Type and Cluster Sizing with EMR Accelerator

As discussed in section 1.2 above, EMR6.X+Windjammer Accelerator Plug In uses ¼ the CPU time per query, 3x the S3 bandwidth per query, and requires ½ or ¼ the number of slaves relative to stock EMR6.X while still providing acceleration. As a result, the selection of EC Instance types and cluster sizes is different than for stock EMR6.X.

Cluster EC2 Instance type and cluster sizing considerations with the Windjammer Accelerator Plug In:

- Use EC2 instances with approximately 1Gbps/vCPU networking performance to enable balanced, high core utilization slaves. Note that EC2 instance types that are balanced for stock EMR6.X (e.g., c4.8xlarge 36 cores, 10Gb network) are unbalanced for EMR Accelerator. Due to EMR Accelerator's ¼ CPU usage and 3X network bandwidth utilization, using EC2 instances with high vCPU counts and low network bandwidth results in low core utilization due to network starvation. It is therefore important to select EC2 instances for EMR Accelerator deployments with sufficient network bandwidth (approximately 1Gps/vCPU). This includes EC2 instances with an "n" in their name.
- Local SSDs are not needed
- Balanced EMR6.X+WJ Plug In clusters for a given workload typically require far fewer EC2 instances than stock EMR6.X while still accelerating performance
- EMR6.X+WJ Plug In has perfect vertical scaling within EC2 instance families that have sufficient networking performance. For example, an EMR6.X+WJ Plug cluster with 4 slaves of EC2s instance type r5dn.4xlarge has the same performance as an EMR6.X+WJ Plug cluster with 2 slaves of EC2 instance type r5dn.8xlarge. As a result, you can use any size EC2 instance within an instance family if it has approximately 1 Gbps/vCPU networking performance.
- You can select the number of EC2 instances in your cluster to trade off cluster cost vs query runtime and for concurrent queries
- The EC2 instance type selection and cluster sizing shown below in section 2.3.2 selecting cluster nodes" represents a balanced cluster configuration for running the EMR Accelerator framework with the TPC-DS workload at 3TB or 10 TB scale (2 worker core nodes of EC2 instance type c5dn.8xlarge).

2.3 EMR Accelerator Deployment and A/B Benchmarking POCs using the EMR Console

The EMR Accelerator A/B Benchmarking facility makes use of an S3 bucket to serve the TPCDS dataset files. There are currently three copies of this bucket located in the following regions:

- `us-east-1`
- `us-east-2`
- `us-west-2`

These regional TPCDS dataset buckets have a name of the form `s3://wjm-datasets-<region>`. For example, the bucket for the `us-west-2` region is named `s3://wjm-datasets-us-west-2`.

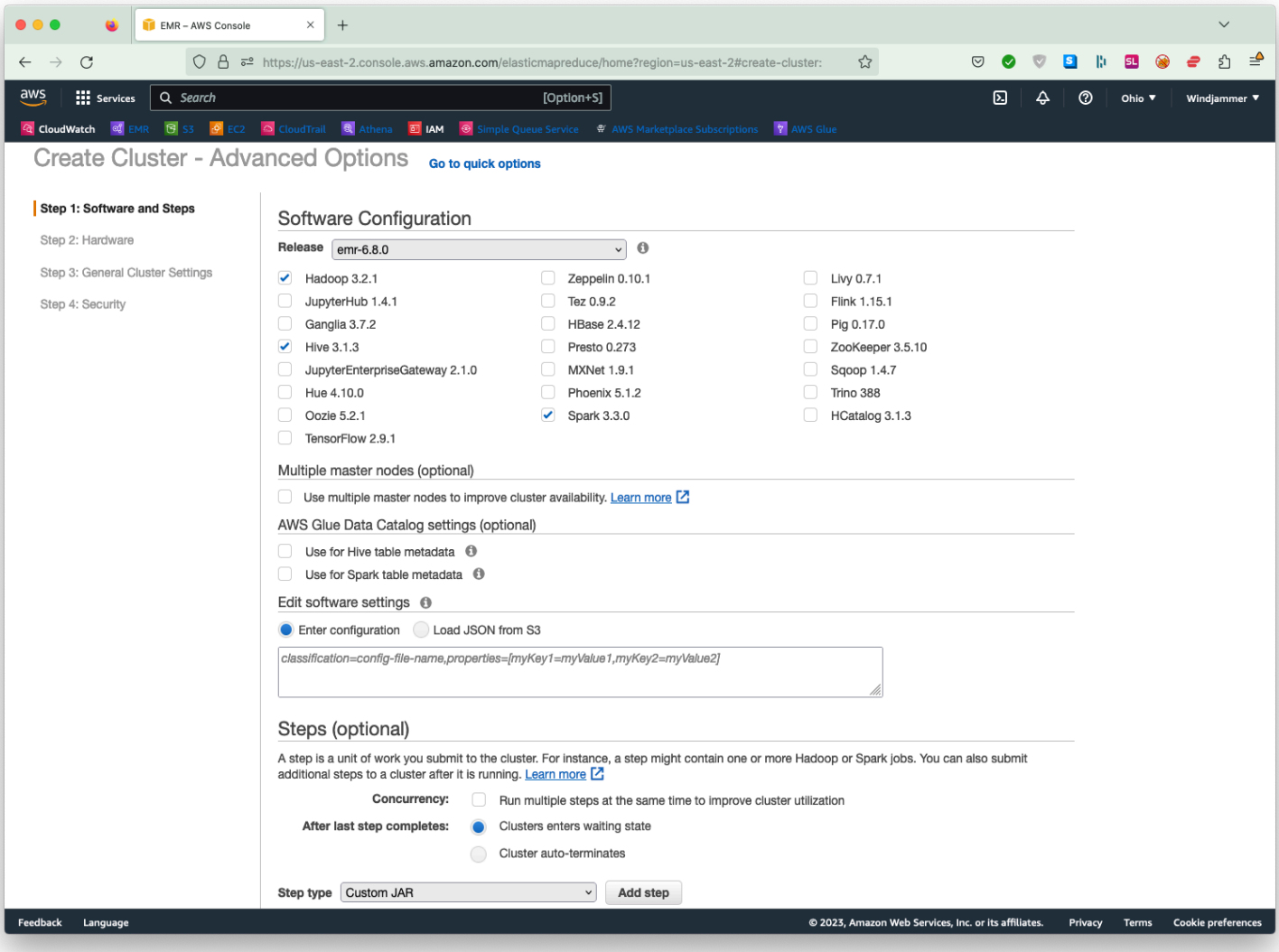
If your cluster is not created in one of these regions, you should copy the contents of `s3://wjm-datasets-<region>` to a local S3 bucket. This will yield the best performance and lowest cost when performing your tests.

The TPC-DS datasets buckets contain the 3TB dataset only. Windjammer supports the 10TB TPC-DS dataset in the S3 bucket `s3://wjm-datasets` located in region `us-east-2`. If you wish to test with the 10TB dataset, do the following to ensure optimal performance:

- Create an S3 bucket in your local region.
- Copy the 10TB dataset from the `s3://wjm-datasets` bucket:
  `aws s3 cp s3://wjm-datasets/tpcds10t s3://my-tpcds-bucket`

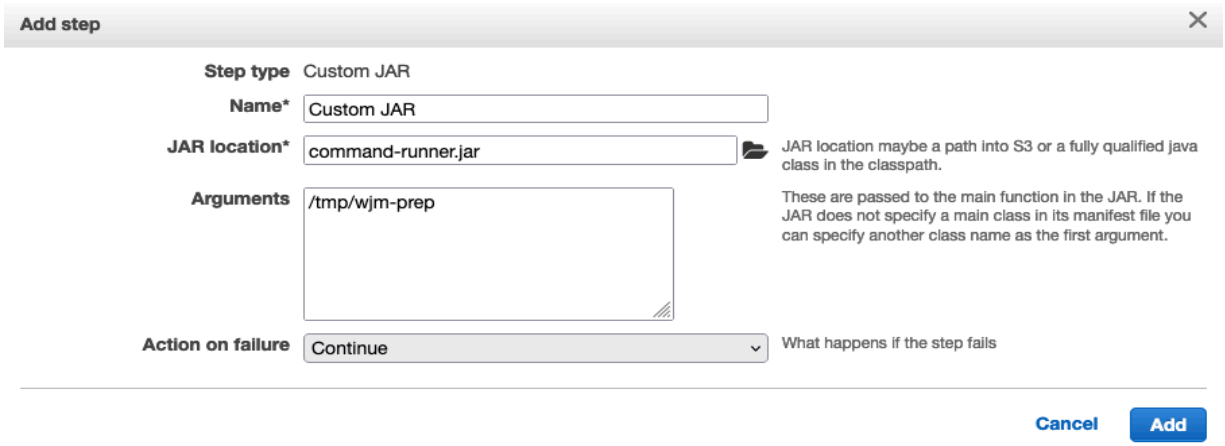
2.3.1 Selecting Cluster Services and Adding Bootstrap Step

- From the EMR dashboard, select **Create cluster**.
- In **Create Cluster – Quick Options** select **Go to advanced options**.
- In **Create Cluster – Advanced Options** - **Software Configuration**, select the **emr-6.2.0 Release**.
- Minimally select **Hadoop**, **Hive** and **Spark**.

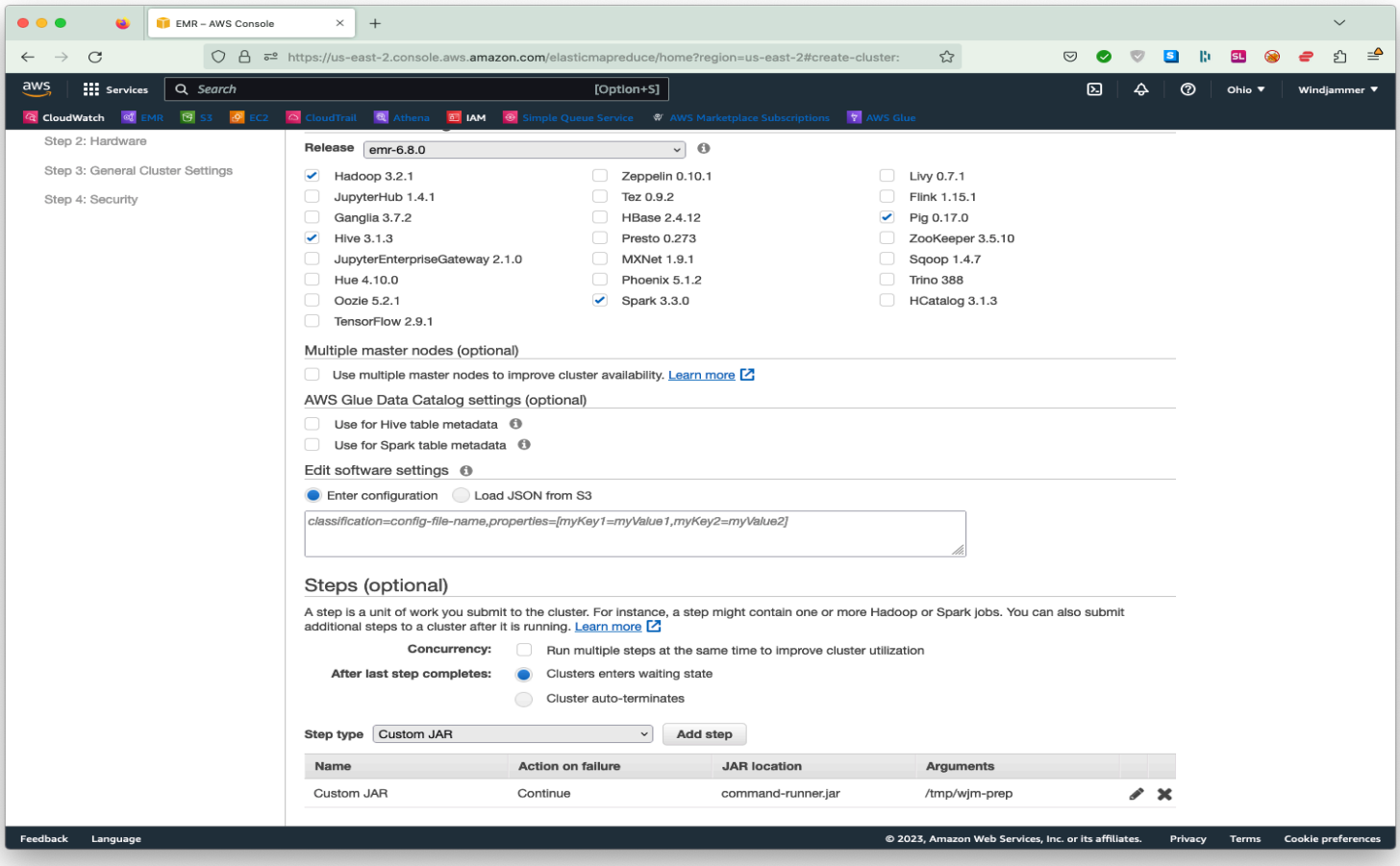


In **Steps (optional)**,

- select **Step type Customer JAR**.
- Click **Add step**.
- In **Jar location**, enter `command-runner.jar`.
- In **Arguments**, enter `/tmp/wjm-prep` and click **Add**.

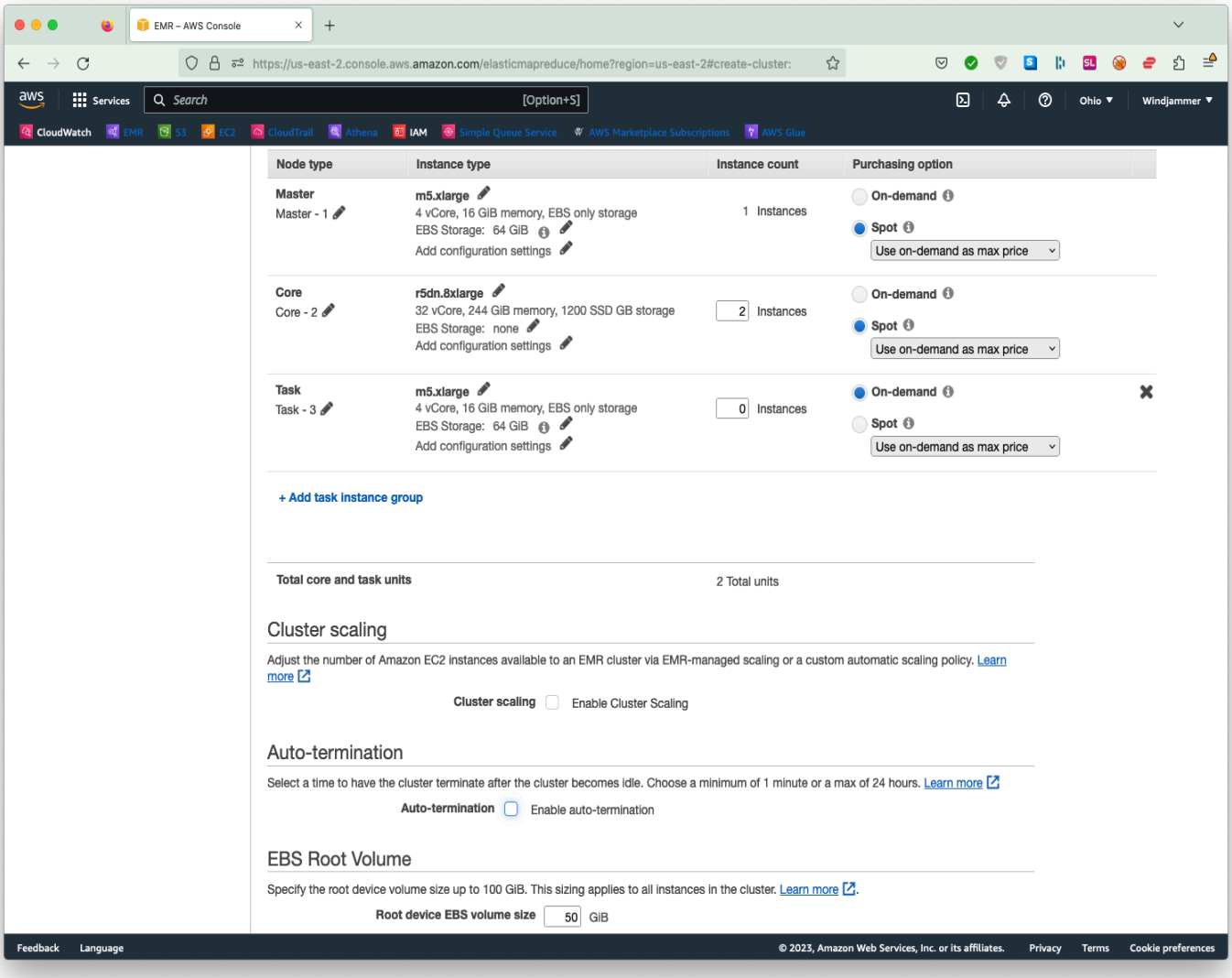


- Click **Next**.



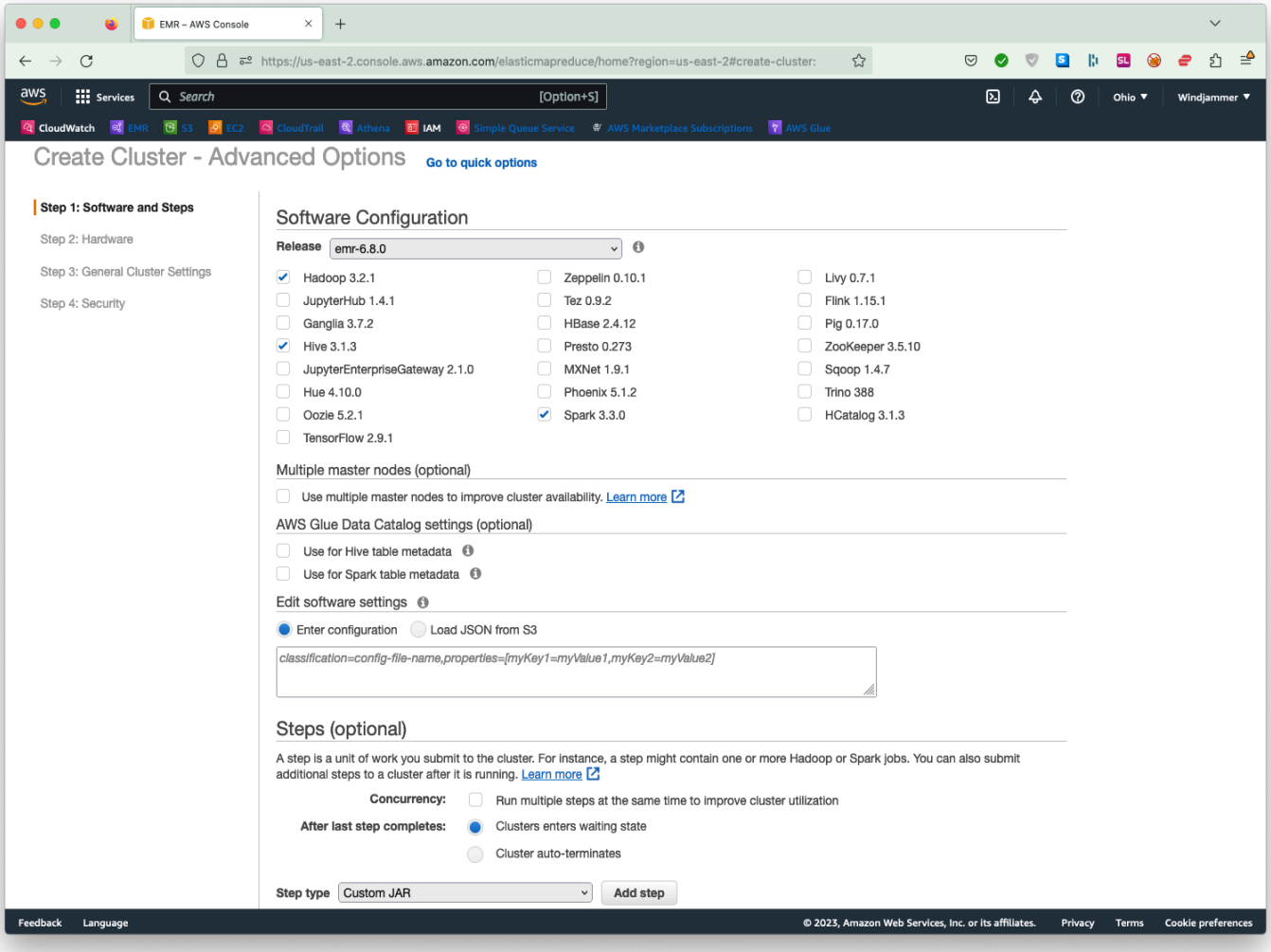
2.3.2 Selecting Cluster Nodes

- In **Create Cluster – Advanced Options** – **Cluster Nodes and Instances**, select **Master/Core** nodes.
- In **EBS Root Volume**, set **Root device EBS volume size** to at least 50 GB.
- Click **Next**.

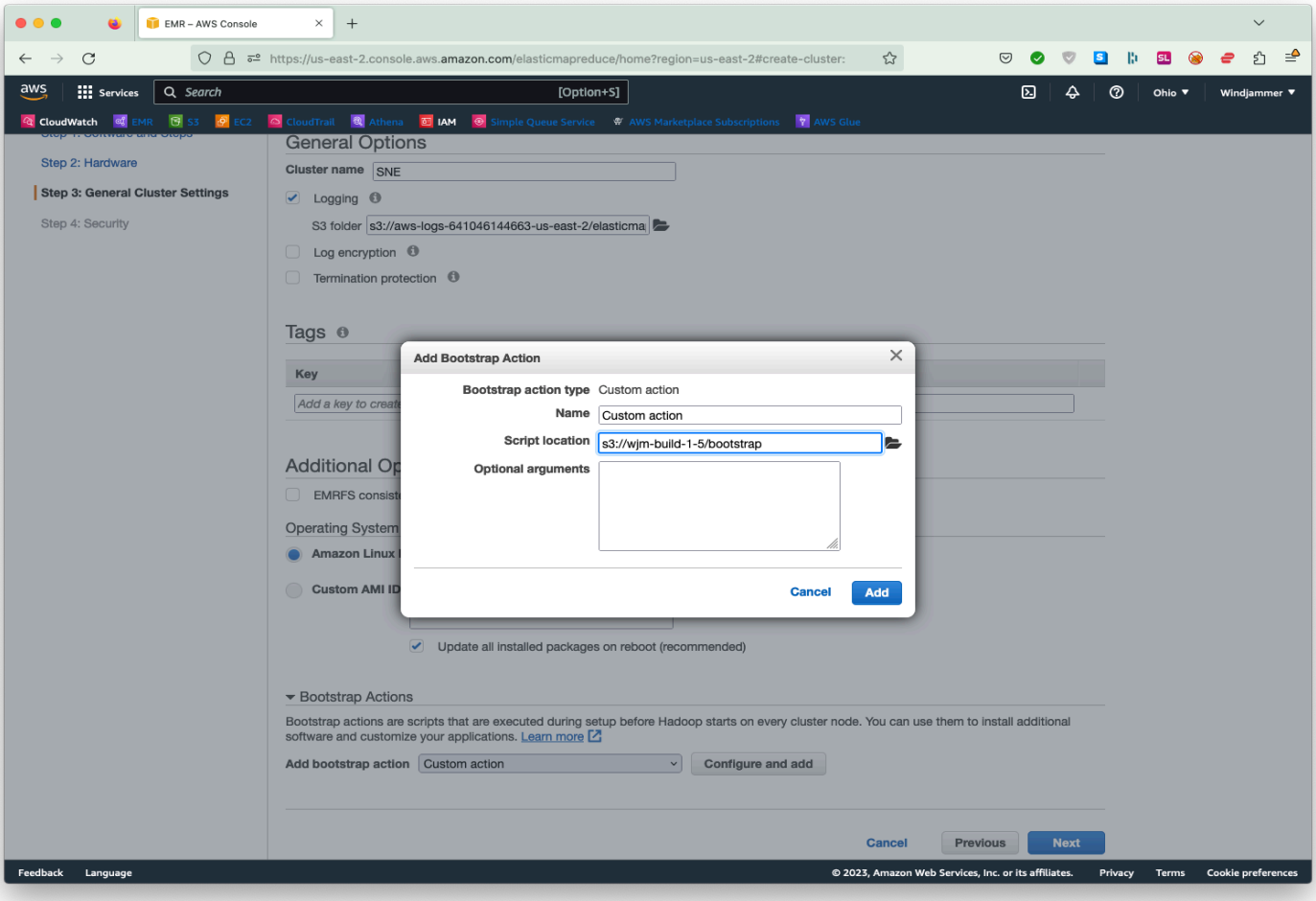


2.3.3 Configuring the Bootstrap Action

- In **Create Cluster – Advanced Options** – **General Cluster Settings**, select **Custom action** from **Add bootstrap action**.
- Click **Configure** and **Add**.

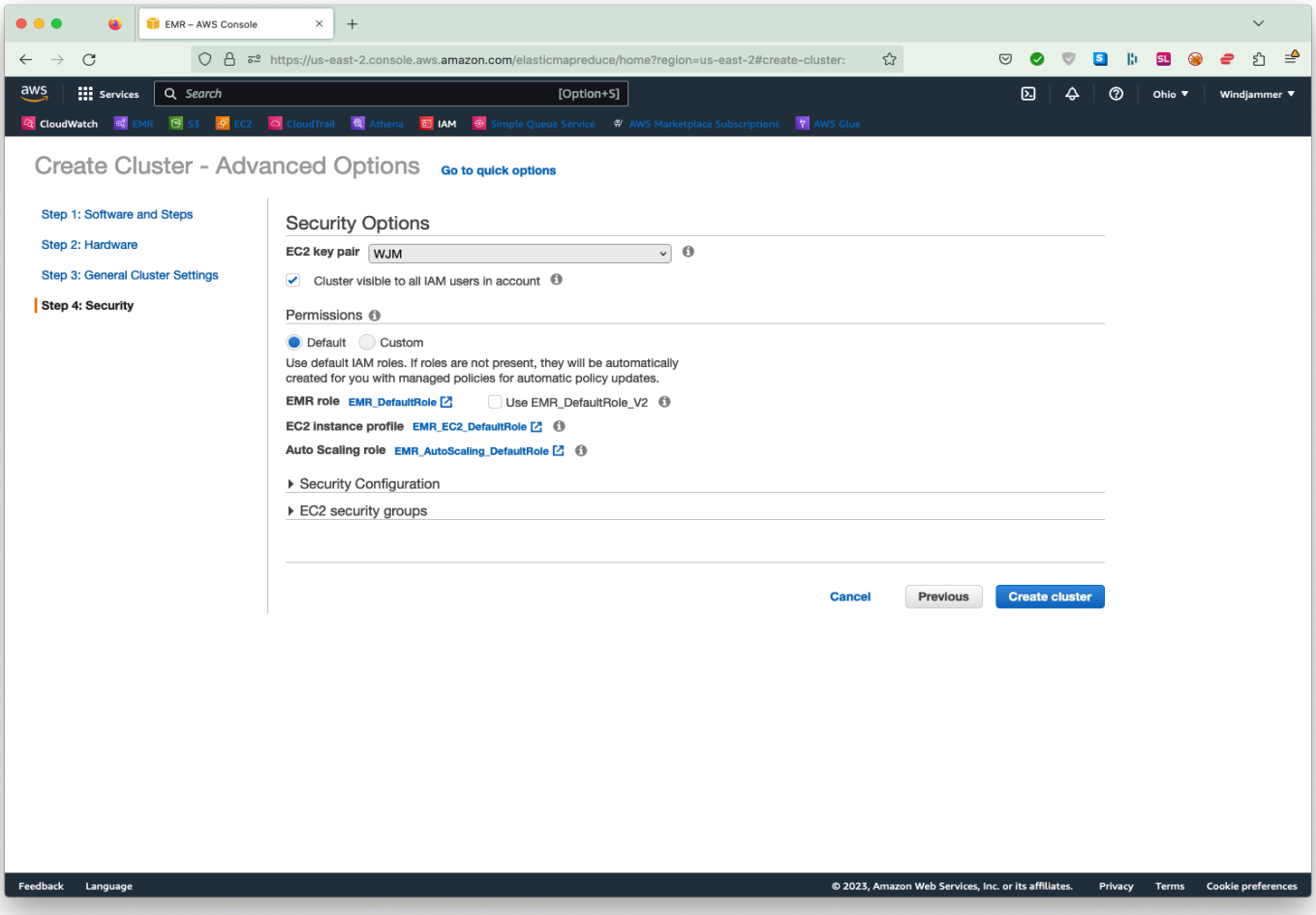


- In **Script location**, enter `s3://wjm-build-1-5/bootstrap`.
- If you have copied the `s3://wjm-datasets-<region>` bucket locally, enter its name in the **Optional arguments** as **`WJM_DATASETS`**`=<local bucket name>`. Otherwise, you may leave **Optional arguments** empty.
- Click **Add** and **Next**.

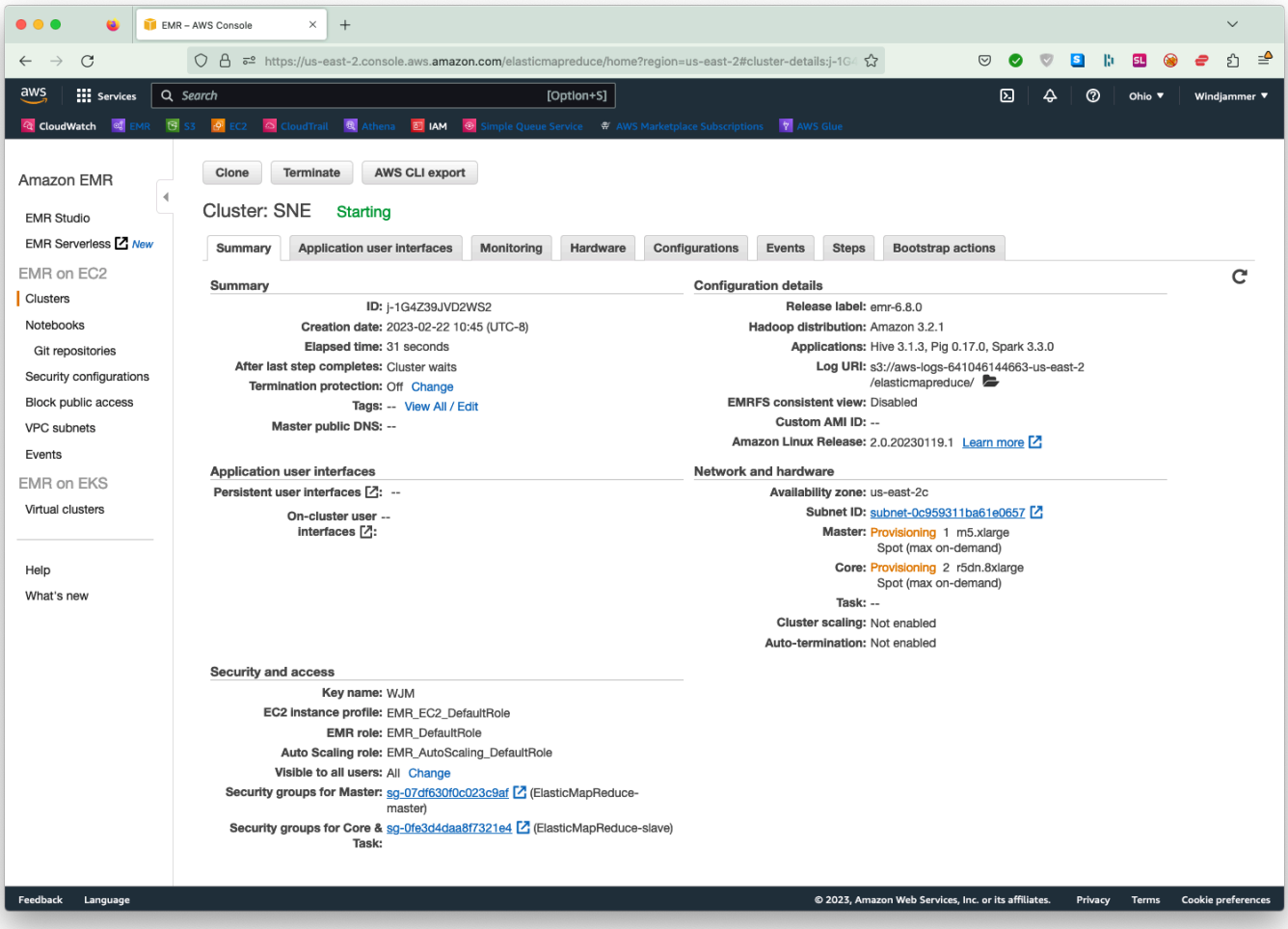


2.3.4 Selecting Security Key and Creating Cluster

- In **Create Cluster – Advanced Options – Security Options**, select a security key from **EC2 key pair**.
- Click **Create cluster**.

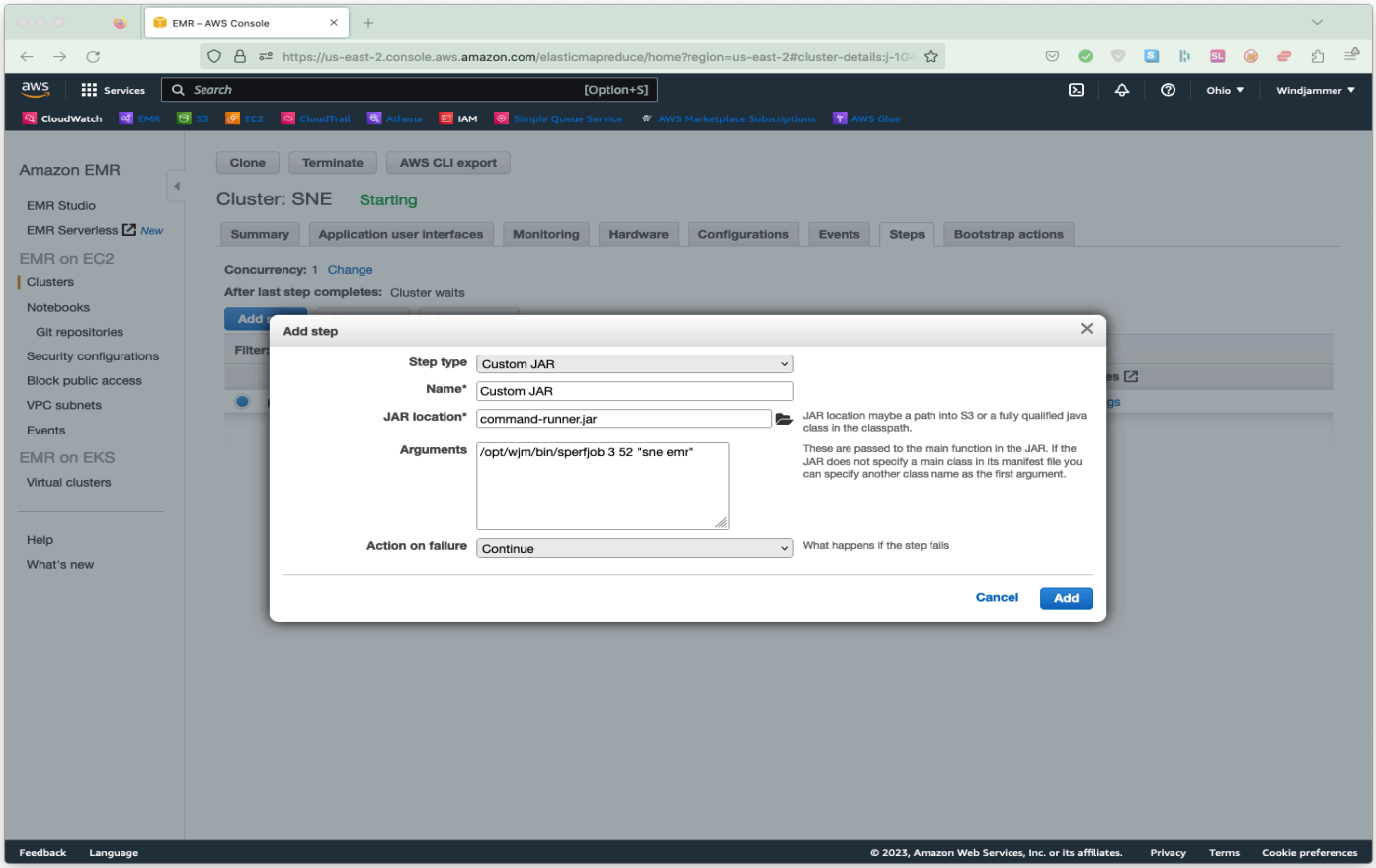


- The cluster will now boot.



2.3.5 Executing an A/B Benchmark from the Installed Set of TPCDS Queries

- In the cluster dashboard, select **Steps**.
- Select **Add step**.
- In **JAR location** enter `command-runner.jar`
- In **Arguments**, enter `/opt/wjm/bin/sperfjob 3 52 "sne emr"`
- Click **Add** and the query will now execute 3 times.
- You may change the number of execution iterations to 1 by adding a "0 1" to the Arguments line: `/opt/wjm/bin/sperfjob 3 52 "sne emr" 0 1`



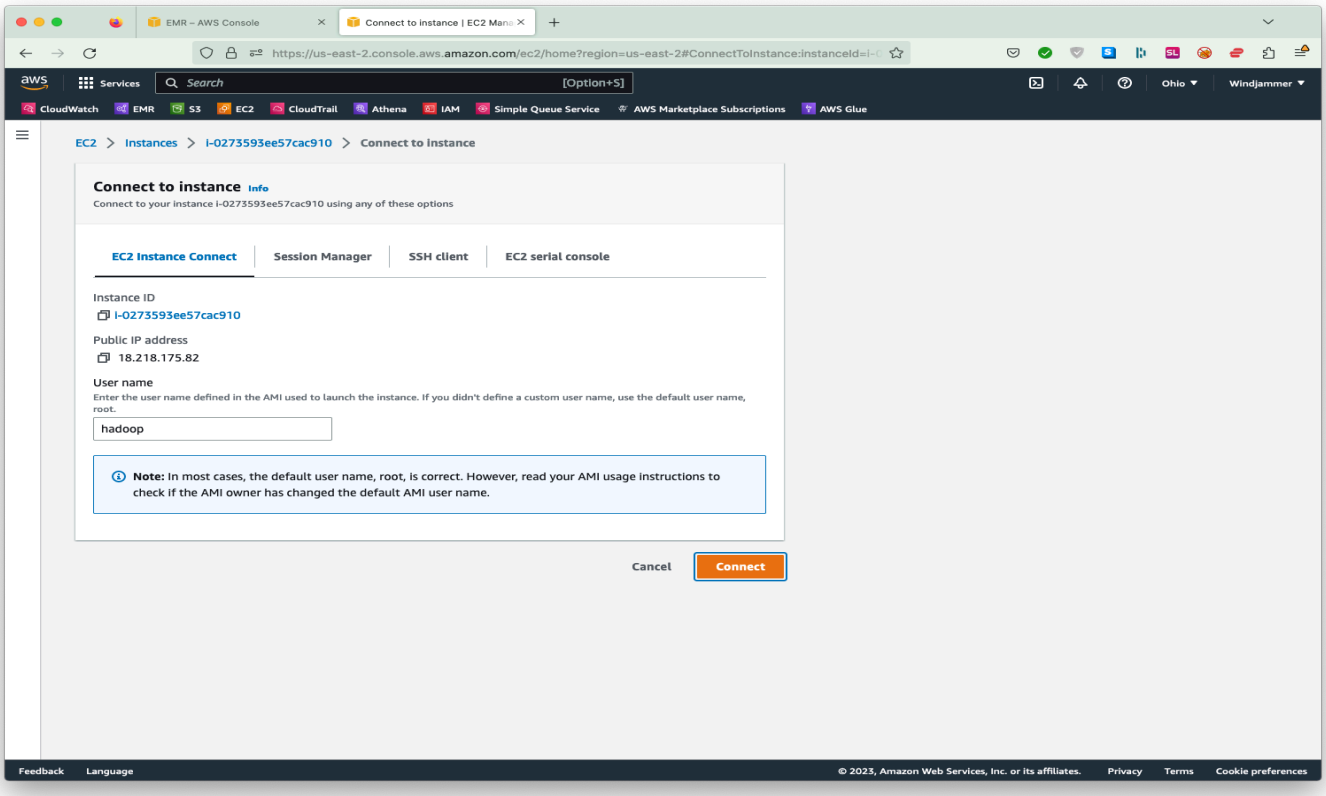
2.3.6 Executing an A/B Benchmark from a Customer Query

A/B benchmark queries 1-99 are reserved for the installed TPCDS SQL query files. A/B benchmark queries 1000-1099 queries are reserved for customer queries. For example, `q52.sql` refers to an installed query while `q1000.sql` refers to a customer query. In the example below, we use the `q52.sql` statements in place of a customer's SQL statements. In practice, the customer would add their own SQL file as `q1000.sql`.

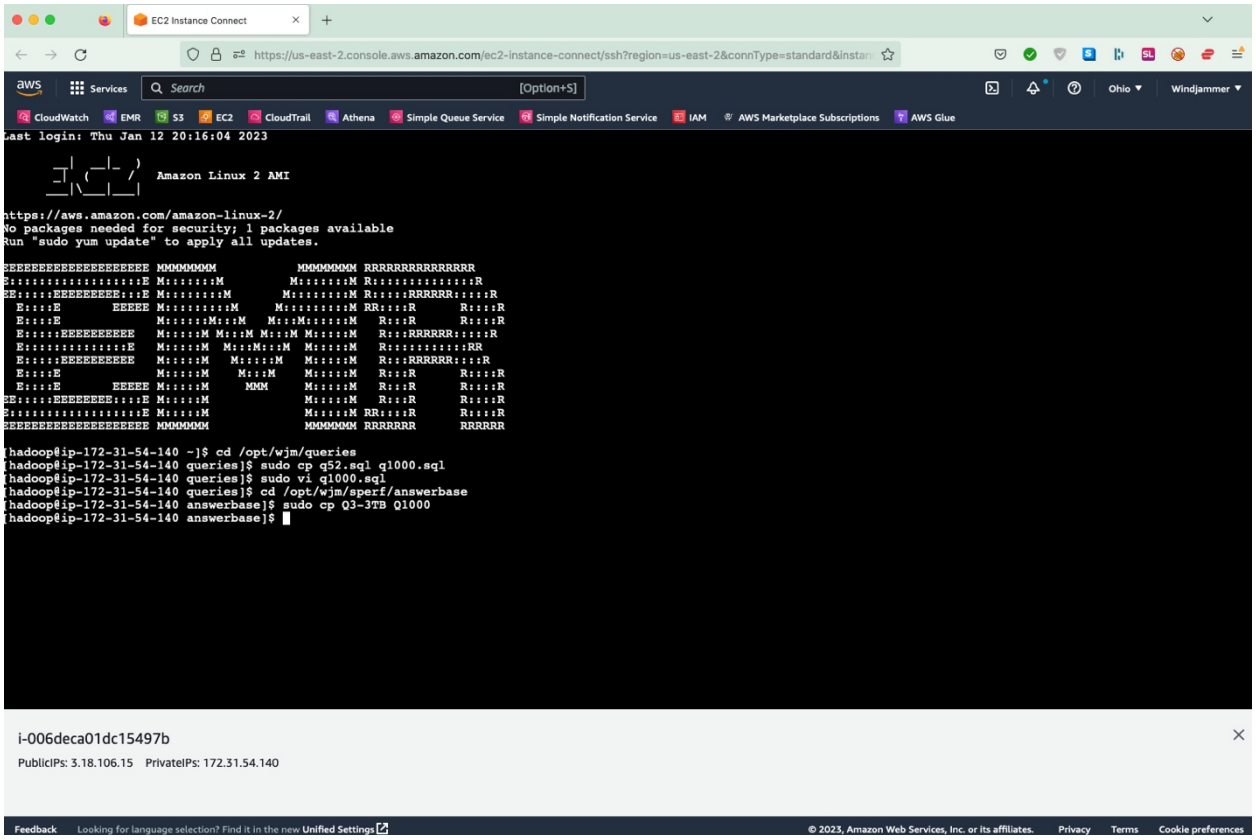
The Windjammer performance monitoring system validates that query results match expected results in A/B experiments. For customer queries, if the final query result is returned through stdout and the query result is stored in the answer database, the performance monitor compares the query results of each query run in A/B experiments with the stored answer and reports match/fail in the autorun report. To store customer query results in the answer database, follow the steps in the example below.

The steps below use the installed TPCDS SQL and result files as examples of customer files.

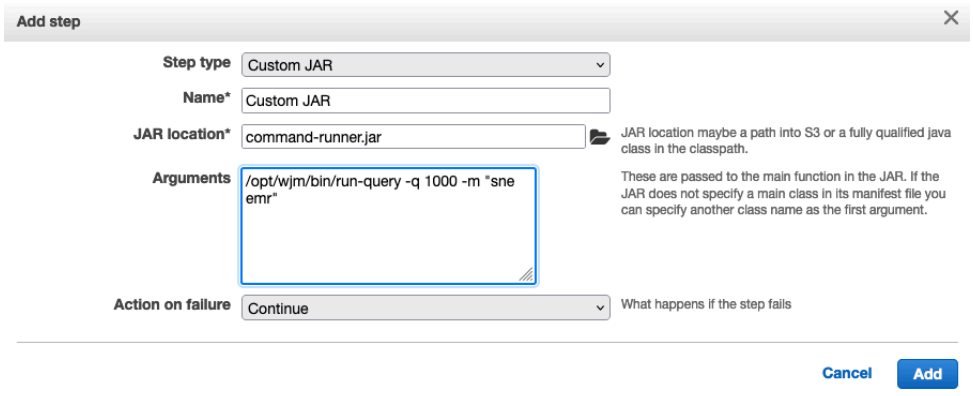
- From the **EC2** dashboard, select the cluster master node and click **Connect**.
- In the **Connect to instance** page, enter "`hadoop`" in the **User name** field in the **EC2 Instance Connect** tab.
- Click **Connect**.
- The master node terminal view will appear.
- Note that to use EC2 Instance Connect, your account must be enabled to use it.
- Alternatively, you may use any SSH client to access the master node.



- Execute the following to simulate installation of a customer query file:
    - In the master terminal window, execute `cd /opt/wjm/queries`.
    - Execute `sudo cp q52.sql q1000.sql`.
    - Execute `sudo vi q1000.sql` and insert use `tpcds3t;` before the first SQL statement.
- Execute the following to simulate installation of a customer results file:
    - Execute `cd /opt/wjm/sperf/answerbase`.
    - Execute sudo `cp Q52-3TB Q1000`.

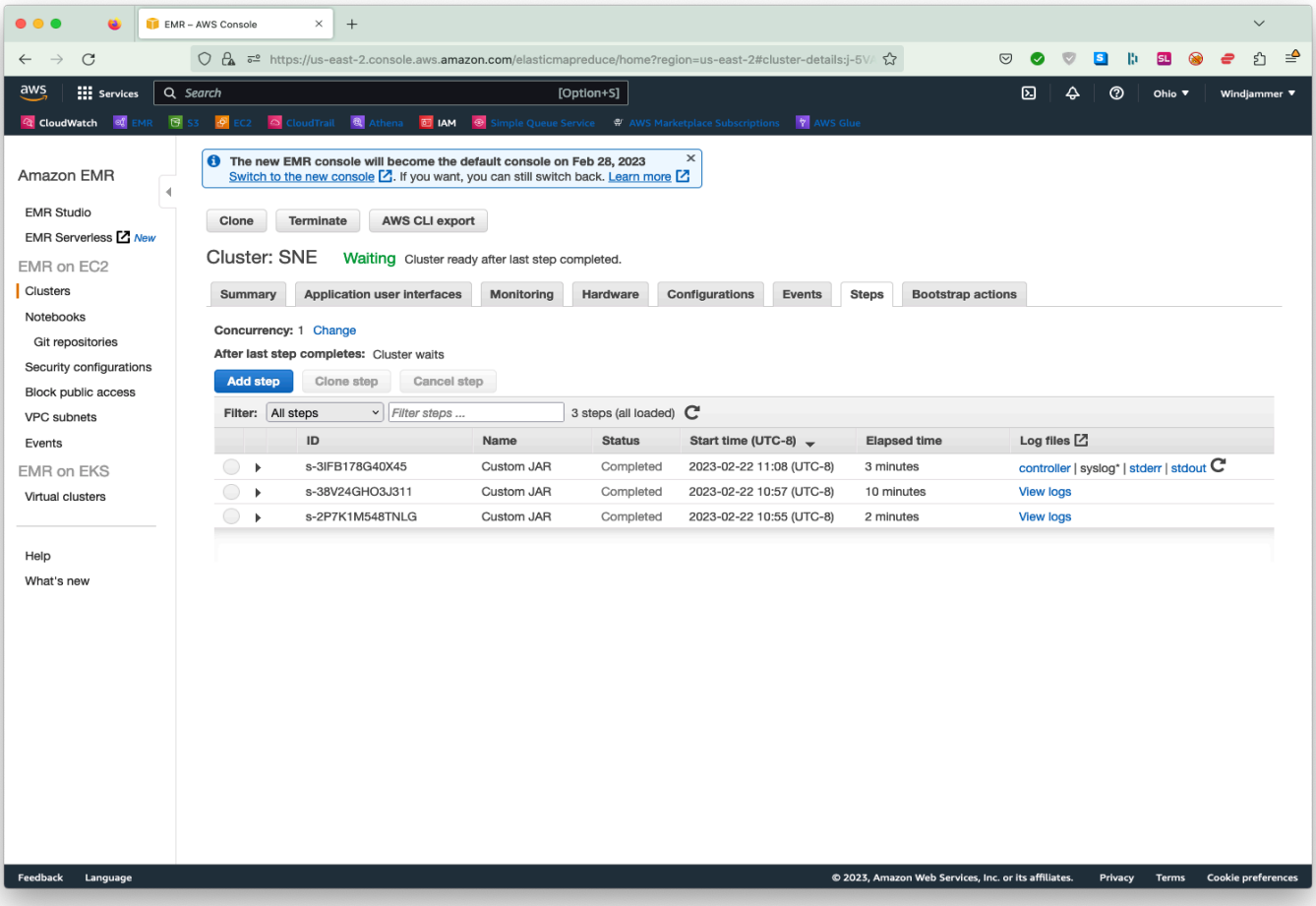


- In the cluster dashboard, select **Add step**.
- In the **Jar location** field, enter `command-runner.jar`
- In the **Arguments** field, enter `/opt/wjm/bin/run-query -q 1000 -m "sne emr"`
- Click Add.
- The customer query will now execute.



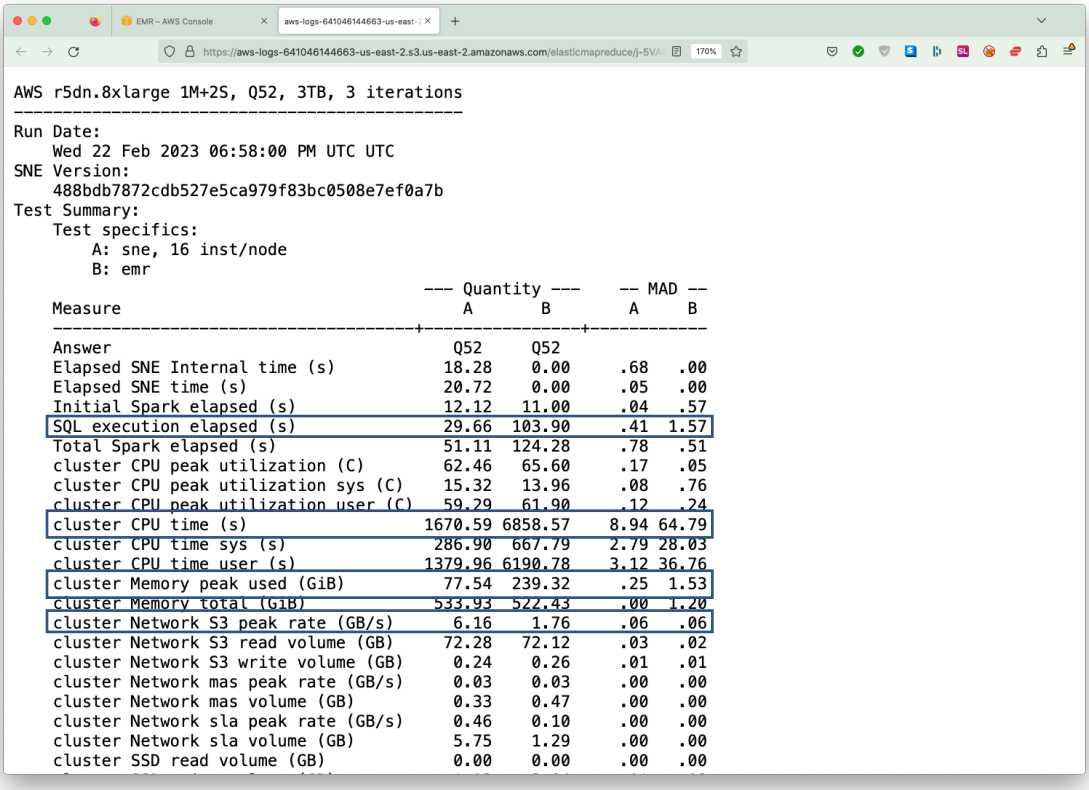
2.3.7 Viewing the Results of an A/B Benchmark Query Experiment

- From the cluster dashboard, select the **View logs** link.
- Select the **stdout** link once it is enabled.



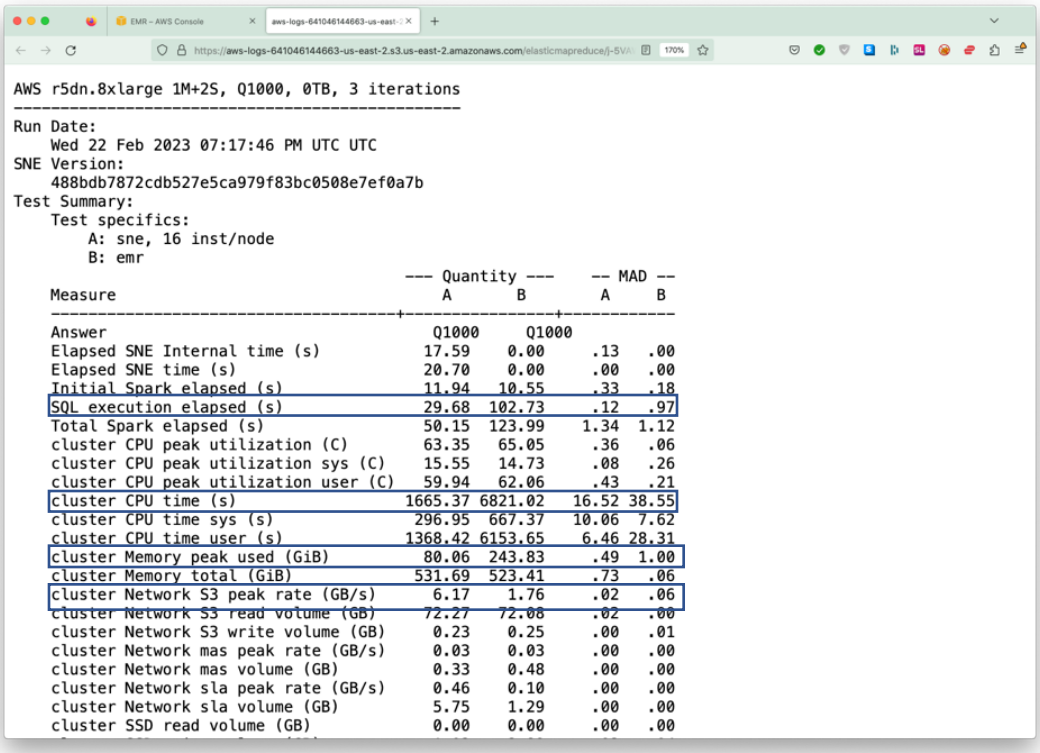
- The Windjammer A/B performance report will appear.

- In this example, TPCDS query Q52 performance running SNE and EMR Spark is shown.
- MAD is median absolute deviation, which is an indication of variability between iterations.



```
AWS r5dn.8xlarge 1M+2S, Q52, 3TB, 3 iterations
-----------------------------------------------
Run Date:
    Wed 22 Feb 2023 06:58:00 PM UTC UTC
SNE Version:
    488bdb7872cdb527e5ca979f83bc0508e7ef0a7b
Test Summary:
    Test specifics:
        A: sne, 16 inst/node
        B: emr
                                        --- Quantity ---     -- MAD --
    Measure                                 A        B         A     B
    ------------------------------------+----------------+------------
    Answer                                 Q52      Q52
    Elapsed SNE Internal time (s)        18.28     0.00      .68   .00
    Elapsed SNE time (s)                 20.72     0.00      .05   .00
    Initial Spark elapsed (s)            12.12    11.00      .04   .57
    SQL execution elapsed (s)            29.66   103.90      .41  1.57
    Total Spark elapsed (s)              51.11   124.28      .78   .51
    cluster CPU peak utilization (C)     62.46    65.60      .17   .05
    cluster CPU peak utilization sys (C) 15.32    13.96      .08   .76
    cluster CPU peak utilization user (C) 59.29   61.90      .12   .24
    cluster CPU time (s)               1670.59 6858.57     8.94 64.79
    cluster CPU time sys (s)            286.90  667.79     2.79 28.03
    cluster CPU time user (s)          1379.96 6190.78     3.12 36.76
    cluster Memory peak used (GiB)       77.54   239.32      .25  1.53
    cluster Memory total (GiB)          533.93   522.43      .00  1.20
    cluster Network S3 peak rate (GB/s)    6.16     1.76     .06   .06
    cluster Network S3 read volume (GB)   72.28    72.12     .03   .02
    cluster Network S3 write volume (GB)   0.24     0.26     .01   .01
    cluster Network mas peak rate (GB/s)   0.03     0.03     .00   .00
    cluster Network mas volume (GB)        0.33     0.47     .00   .00
    cluster Network sla peak rate (GB/s)   0.46     0.10     .00   .00
    cluster Network sla volume (GB)        5.75     1.29     .00   .00
    cluster SSD read volume (GB)           0.00     0.00     .00   .00
```

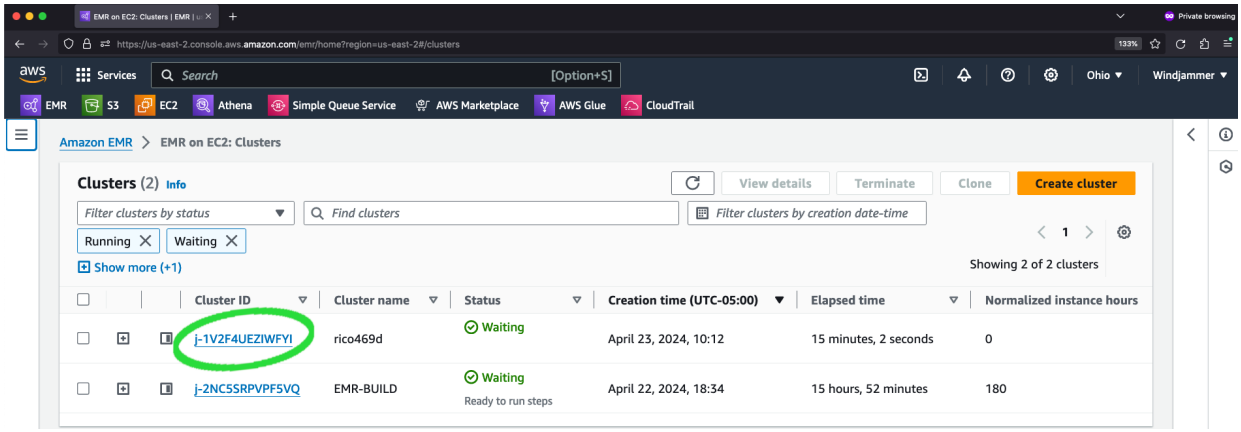
- In this example, the “customer” query Q1000 is shown.



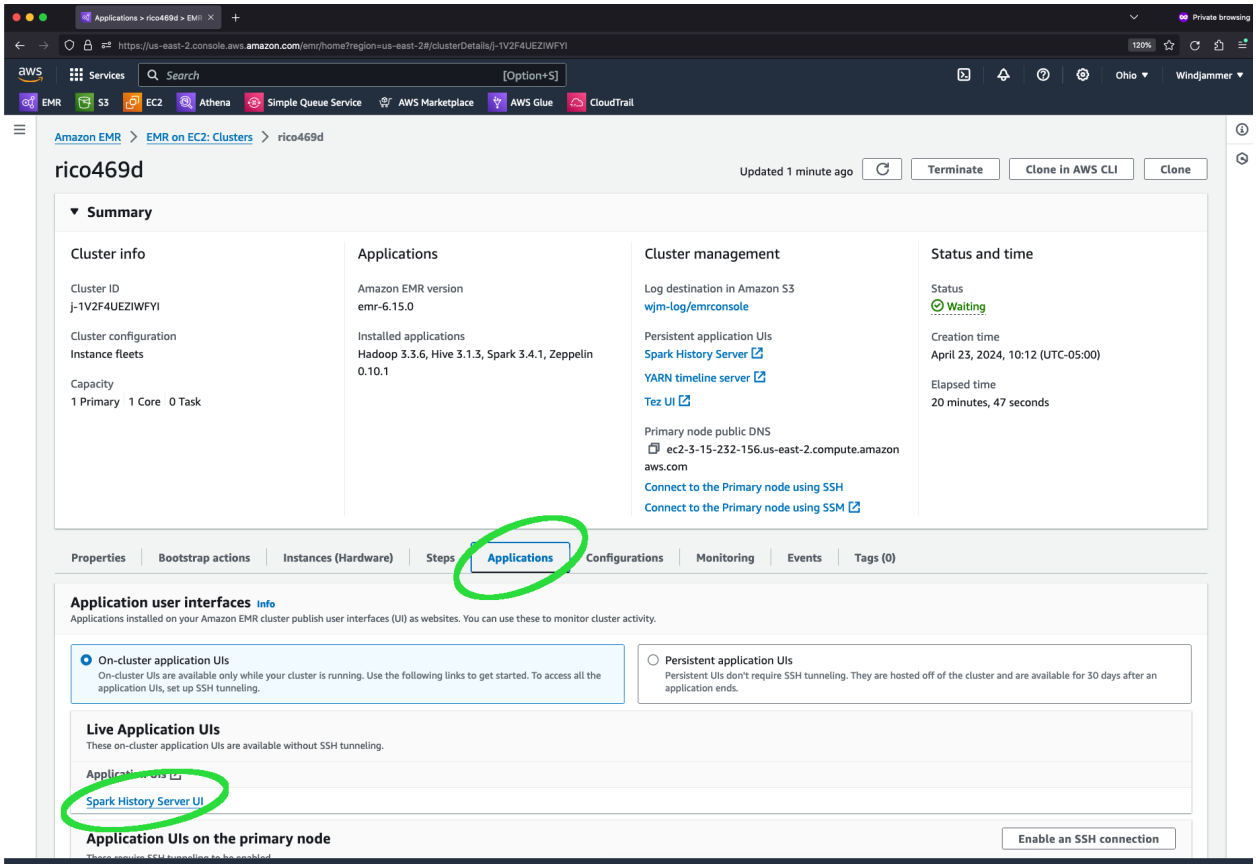
```
AWS r5dn.8xlarge 1M+2S, Q1000, 0TB, 3 iterations
------------------------------------------------
Run Date:
    Wed 22 Feb 2023 07:17:46 PM UTC UTC
SNE Version:
    488bdb7872cdb527e5ca979f83bc0508e7ef0a7b
Test Summary:
    Test specifics:
        A: sne, 16 inst/node
        B: emr
                                          --- Quantity ---     -- MAD --
    Measure                                  A        B        A     B
    -------------------------------------+--------------+-------------
    Answer                                  Q1000     Q1000
    Elapsed SNE Internal time (s)          17.59     0.00     .13   .00
    Elapsed SNE time (s)                   20.70     0.00     .00   .00
    Initial Spark elapsed (s)              11.94    10.55     .33   .18
    SQL execution elapsed (s)              29.68   102.73     .12   .97
    Total Spark elapsed (s)                50.15   123.99    1.34  1.12
    cluster CPU peak utilization (C)       63.35    65.05     .36   .06
    cluster CPU peak utilization sys (C)   15.55    14.73     .08   .26
    cluster CPU peak utilization user (C)  59.94    62.06     .43   .21
    cluster CPU time (s)                 1665.37  6821.02   16.52 38.55
    cluster CPU time sys (s)              296.95   667.37   10.06  7.62
    cluster CPU time user (s)            1368.42  6153.65    6.46 28.31
    cluster Memory peak used (GiB)         80.06   243.83     .49  1.00
    cluster Memory total (GiB)            531.69   523.41     .73   .06
    cluster Network S3 peak rate (GB/s)     6.17     1.76     .02   .06
    cluster Network S3 read volume (GB)    72.27    72.08     .02   .00
    cluster Network S3 write volume (GB)    0.23     0.25     .00   .01
    cluster Network mas peak rate (GB/s)    0.03     0.03     .00   .00
    cluster Network mas volume (GB)         0.33     0.48     .00   .00
    cluster Network sla peak rate (GB/s)    0.46     0.10     .00   .00
    cluster Network sla volume (GB)         5.75     1.29     .00   .00
    cluster SSD read volume (GB)            0.00     0.00     .00   .00
```

2.4 Windjammer Spark History Server Extensions

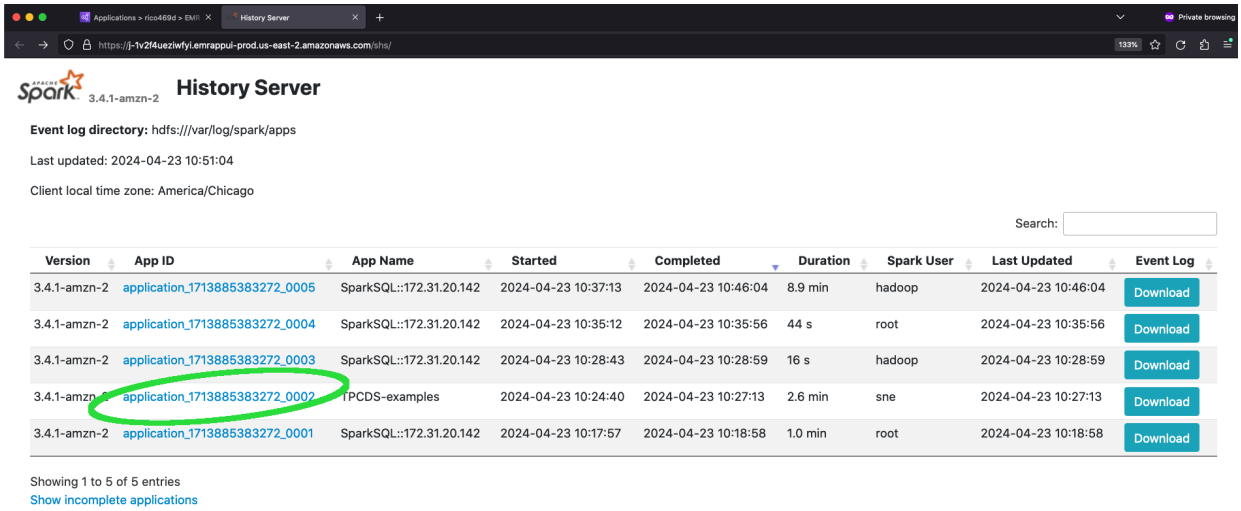
EMR Dashboard shows known clusters. Click on one to see further details:



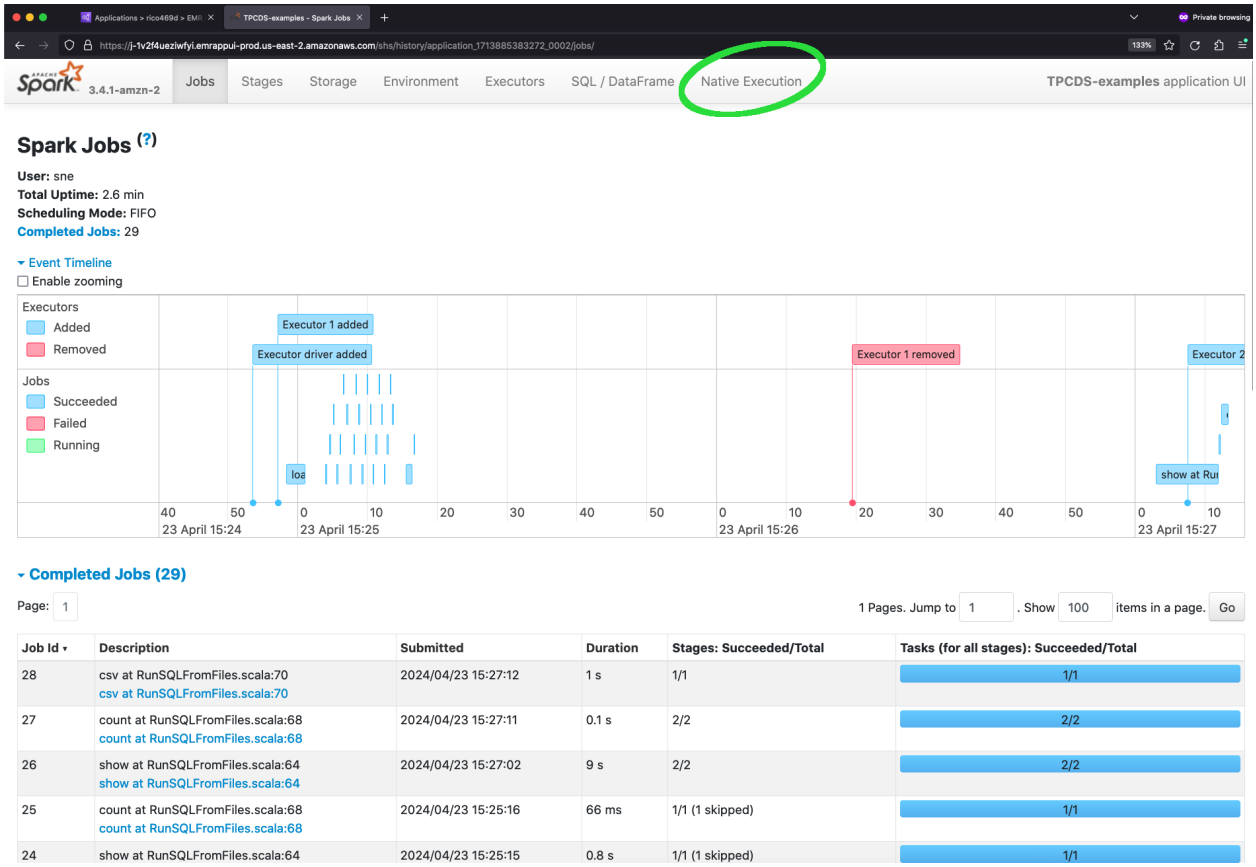
Run one or more Spark queries. Query status can be inspected while Spark is evaluating a query, but additional info is available after that query completes. The info is presented by the Spark History Server that can be accessed by selecting the "Applications" tab, then clicking the "Spark History Server UI" link:



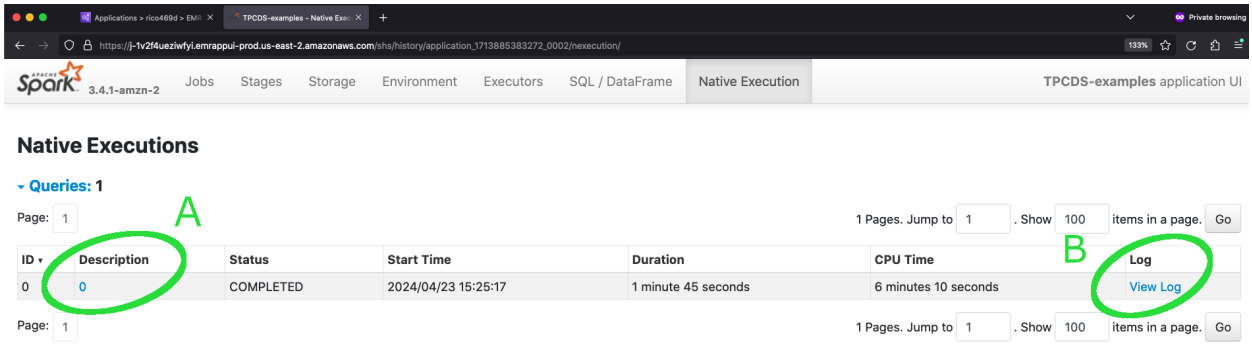
At the History Server top page, select the application (Spark JVM) of interest. By default, newest applications are listed first:



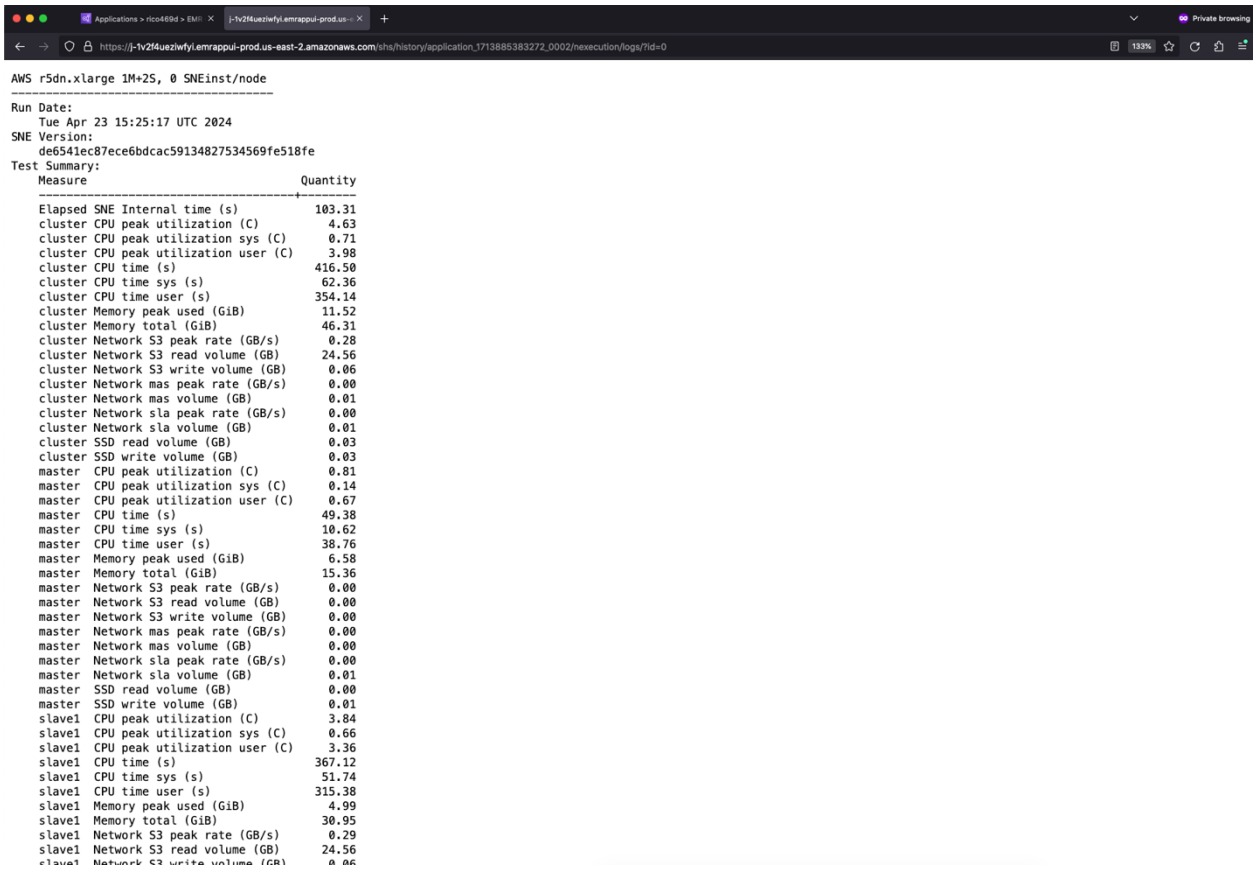
The top-level view of the selected application shows the Spark jobs created to service the one or more queries submitted to the JVM. Besides the standard tabs that detail the state and activities of the JVM, Windjammer also provides this "Native Execution" tab:



The Native Executions subpage will list each query accelerated by SNE. Two clickable links are provided per query, with the "View log" link (marked 'B') showing a report for that query:

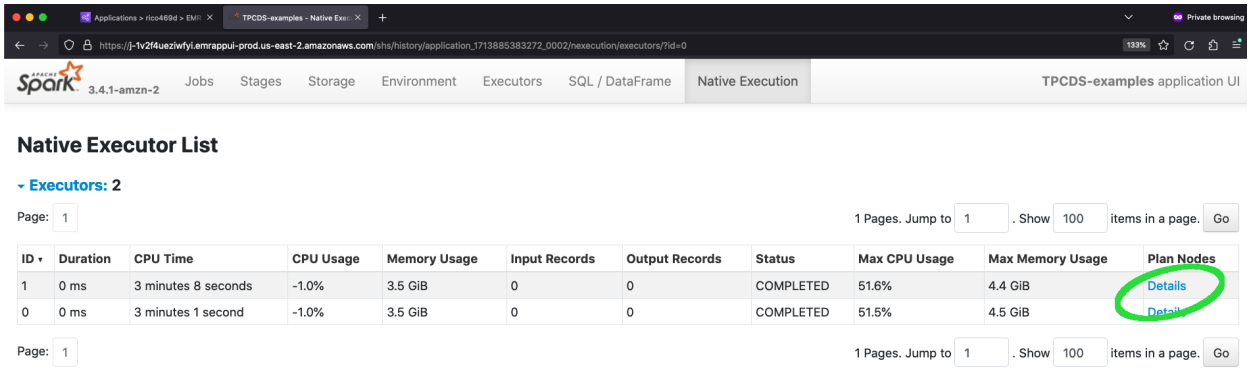


The report summarizes the resources used for a query, both whole cluster and per node:

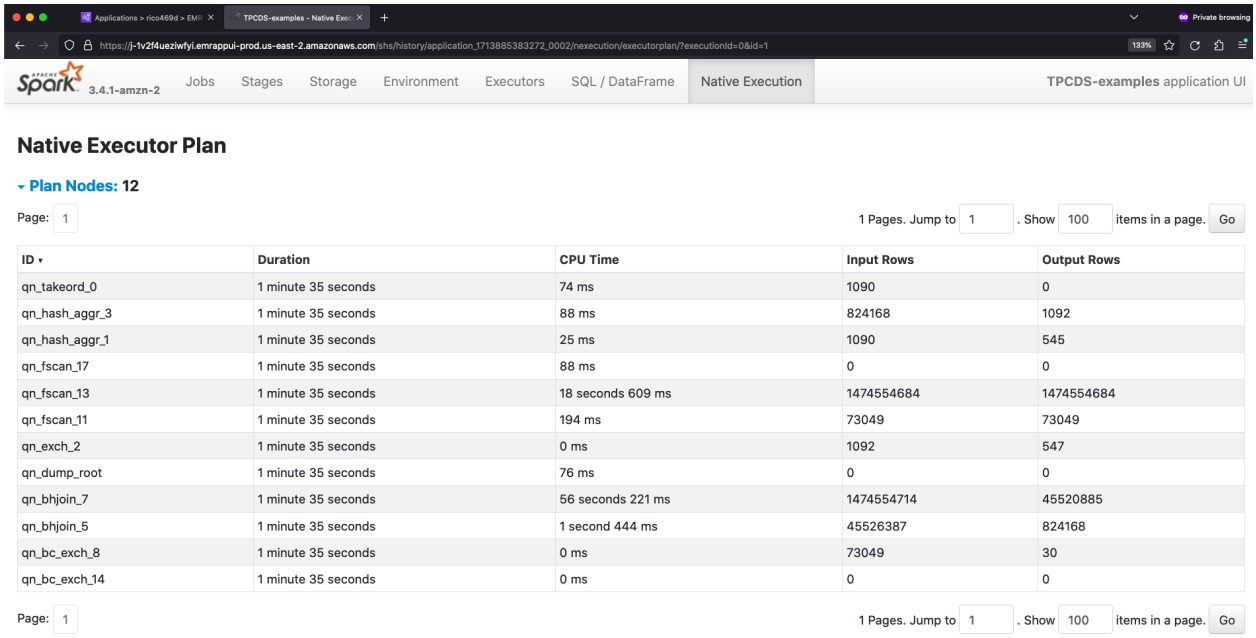


```
AWS r5dn.xlarge 1M+2S, 0 SNEinst/node
--------------------------------------
Run Date:
    Tue Apr 23 15:25:17 UTC 2024
SNE Version:
    de6541ec87ece6bdcac59134827534569fe518fe
Test Summary:
    Measure                                 Quantity
    ----------------------------------------+--------
    Elapsed SNE Internal time (s)             103.31
    cluster CPU peak utilization (C)            4.63
    cluster CPU peak utilization sys (C)        0.71
    cluster CPU peak utilization user (C)       3.98
    cluster CPU time (s)                      416.50
    cluster CPU time sys (s)                   62.36
    cluster CPU time user (s)                 354.14
    cluster Memory peak used (GiB)             11.52
    cluster Memory total (GiB)                 46.31
    cluster Network S3 peak rate (GB/s)         0.28
    cluster Network S3 read volume (GB)        24.56
    cluster Network S3 write volume (GB)        0.06
    cluster Network mas peak rate (GB/s)        0.00
    cluster Network mas volume (GB)             0.01
    cluster Network sla peak rate (GB/s)        0.00
    cluster Network sla volume (GB)             0.01
    cluster SSD read volume (GB)                0.03
    cluster SSD write volume (GB)               0.03
    master  CPU peak utilization (C)            0.81
    master  CPU peak utilization sys (C)        0.14
    master  CPU peak utilization user (C)       0.67
    master  CPU time (s)                       49.38
    master  CPU time sys (s)                   10.62
    master  CPU time user (s)                  38.76
    master  Memory peak used (GiB)              6.58
    master  Memory total (GiB)                 15.36
    master  Network S3 peak rate (GB/s)         0.00
    master  Network S3 read volume (GB)         0.00
    master  Network S3 write volume (GB)        0.00
    master  Network mas peak rate (GB/s)        0.00
    master  Network mas volume (GB)             0.00
    master  Network sla peak rate (GB/s)        0.00
    master  Network sla volume (GB)             0.01
    master  SSD read volume (GB)                0.00
    master  SSD write volume (GB)               0.01
    slave1  CPU peak utilization (C)            3.84
    slave1  CPU peak utilization sys (C)        0.66
    slave1  CPU peak utilization user (C)       3.36
    slave1  CPU time (s)                      367.12
    slave1  CPU time sys (s)                   51.74
    slave1  CPU time user (s)                 315.38
    slave1  Memory peak used (GiB)              4.99
    slave1  Memory total (GiB)                 30.95
    slave1  Network S3 peak rate (GB/s)         0.29
    slave1  Network S3 read volume (GB)        24.56
```

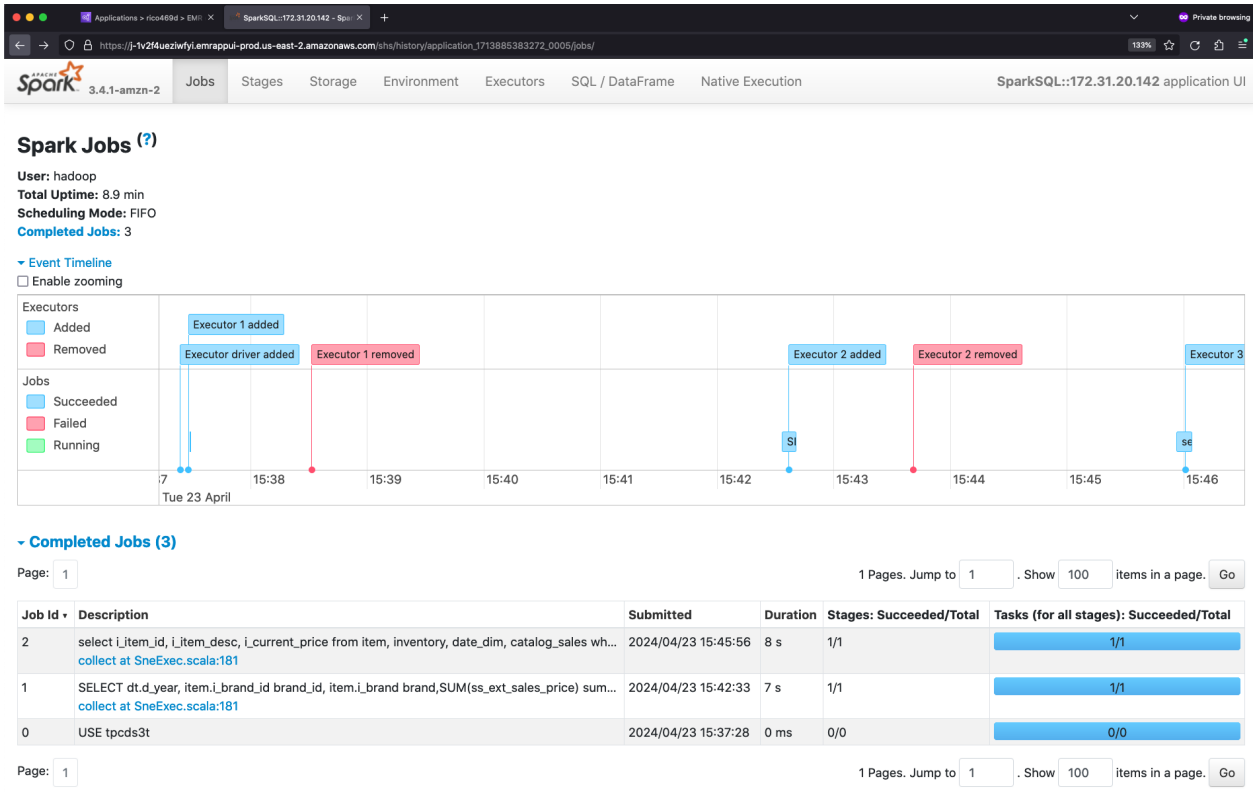
Clickable link marked 'A' shows each SNE executor created for the query and a link to show additional details:



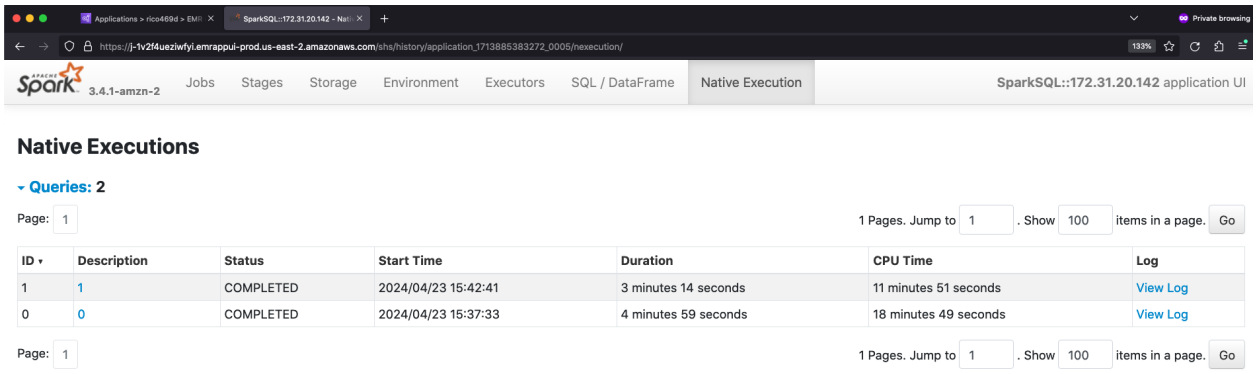
Additional details for an SNE executor include elapsed time, CPU usage and traffic flow as associated with nodes of the physical plan:



Example of a Spark application (spark-sql) with two submitted queries (SELECT statements):



Listing of those two queries from the "Native Execution" tab:



2.5 EMR Accelerator Deployment and Performing A/B Benchmarking POCs using Windjammer Cloud Tools

2.5.1 Command Script Summary

While `aws` commands can be used directly to manage clusters, Windjammer provides a simple set of tools to manage the cluster and execute queries

- `emrstandup` start a new cluster (edit this file)
- `emrlist` list all active clusters
- `emrteardown` terminate a cluster
- `emrstepstart` start a job for autorun (edit this file)
- `emrsteplist` list all steps associated with a cluster
- `emrstepcancel` cancel a PENDING step
- `emrstepwait` wait for a step to complete, show results

You can modify the generic tools to create other clusters and run other queries.

- Edit `emrstandup` configuration variables as required. (recommend US East region)
- Execute `emrstandup`.
- Execute `emrlist` to show the state of the EMR/SNE cluster. Wait for `RUNNING` state.
- Edit `emrstepstart` configuration variables to include actual cluster name and execute.
- Execute `emrstepwait` which gathers the performance report(s) output by `emrstepstart`. These are also available on the EMR Jobs dashboard.

These should be downloaded to the remote machine where EMR activities are directed. The tools are in the bootstrap bucket as `emrtools.tgz`.

To download the Windjammer scripts that manage EMR clusters from a remote machine, type the following commands to the shell:

```
bash$ aws s3 cp s3://wjm-build-1-5/emrtools.tgz .
download: s3://wjm-build-1-5/emrtools.tgz to ./emrtools.tgz
bash$ tar xzf emrtools.tgz
bash$ ls -l emr
total 192
-rwxrwxr-x 1 wjm wjm 167040 Sep 13 04:31 emrclusters
-rwxrwxr-x 1 wjm wjm     58 May 13  2021 emrlist
-rwxrwxr-x 1 wjm wjm   1921 Sep 13 04:41 emrstandup
-rwxrwxr-x 1 wjm wjm    296 Apr 30  2021 emrstepcancel
-rwxrwxr-x 1 wjm wjm    154 Apr 30  2021 emrsteplist
-rwxrwxr-x 1 wjm wjm   1001 Sep  1  2021 emrstepstart
-rwxrwxr-x 1 wjm wjm    493 Apr 30  2021 emrstepwait
-rwxrwxr-x 1 wjm wjm    284 Oct  8  2021 emrteardown
```

To execute these shell scripts, ensure the "emr" directory is in your search path.
The remote machine must be configured with the aws command.
Customers can use these scripts in their own AWS accounts to easily evaluate EMR 6.2 + SNE 1.5 using the testing environment deployed on cluster creation with EMR Accelerator. In addition to the cluster creation settings described in the previous section, the following AWS cluster creation options may be updated to match the customer AWS environment:

- `--auto-scaling-role`
- `--ec2-attributes`
  - `KeyName`
  - `InstanceProfile`
  - `SubnetId`
  - `EmrManagedSlaveSecurityGroup`
  - `EmrManagedMasterSecurityGroup`
- `--service-role`

2.5.2 Configure Cluster and Launch

The cluster to be launched is specified by parameters at the head of the `emrstandup` script. After editing those parameters, execute the script to start an EMR 6.2 + SNE cluster.

Parameters include:

- cluster name
- machine instance types for master/slave
- slave count
- local SSDs
- region
- EMR boot bucket
- EMR log path
- EMR TPCDS dataset bucket (only for A/B evaluation)

emrstandup Deploy EMR/SNE cluster.

parameters Edit script variables to configure the cluster:

CLUSTERNAME=mycluster	Cluster name
MASTERMACHINETYPE=m5.xlarge	Master machine type
MASTERBOOTDISKSIZE=50	Master boot disk size GB
WORKERMACHINETYPE=r5nd.4xlarge	Slave machine type
WORKERBOOTDISKSIZE=200	Slave boot disk size GB
WORKERCOUNT=2	Slave count 3TB 2
DATASETS=my-tpcds-bucket	copy if not in: us-east-1 us-east-2 us-west-2
RELEASE=emr-6.8.0	also emr-6.5 to emr-6.8
BOOTSTRAP=s3://wjm-build-1-5/bootstrap	

return The cluster identifier.

emrstepstart Start an EMR step.

parameters Edit script variables to configure the cluster.

CLUSTER=nnnnnnnn	The cluster identifier
NAME=emrsne	The name of the cluster
DATASETS="3"	TPCDS dataset size (TB). 3TB supported.
QUERIES="3 37 55 82"	Blank separated list of query numbers.

	MODES="sne emr"	Blank separated list of test modes: sne Windjammer SNE. emr EMR Spark. sne emr Both.
	INST="0"	SNE executors/slave: If set to 0 then SNE selects default based on slave type. Default is #vCPUs/2
	ITER=3	Test iteration count.

emrlist	List clusters.	
parameters	None.	
return	The list of clusters active in the region.	
emrteardown	Terminate a cluster.	
parameters	ClusterID	The cluster identifier
return	None.	
emrsteplist	Show EMR steps for a cluster.	
parameters	Edit script variables to configure the step list.	
	ClusterID	The cluster identifier
return	List of steps.	
emrstepwait	Gather step output at conclusion of the step (stdout).	
parameters	Edit script variables to configure the step wait.	
	ClusterID	The cluster identifier
	Key	The cluster SSH key
	StepID	The step identifier
emrstepcancel	Cancel an EMR step.	

```
parameters        Edit script variables to configure the step cancel.
                  ClusterID                       The cluster identifier
                  StepID                          The step identifier
```

2.6 EMR Accelerator Deployment and A/B Benchmarking POCs using the emr create-cluster command (see emrstandup and emrstepstart scripts as examples)

2.6.1 Cluster Deployment

```
aws emr create-cluster \
--name <cluster_name> \
--release-label emr-6.2.0 \
--region us-east-2 \
--bootstrap-actions'[
      {
        "Path": "s3://wjm-build-1-5/bootstrap",
        "Args": [
             "WJM_DATASETS=m my-tpcds-bucket",
        ],
        "Name": "Deploy SNE"
      },

…<other standard EMR options >…

   --steps '[{
        "Name": "010",
        "Args": [
                "/tmp/wjm-prep"
                ],
                "Type": "CUSTOM_JAR",
                "Jar": "command-runner.jar",
                "ActionOnFailure": "CONTINUE"
                }]' \
```

2.6.2 A/B Benchmarking Step

```
NAME=mystepname
DATASETS="3"
QUERIES="3 37"
MODES="sne emr"
INST="0"
ITER=1

main( )
{
        inst="0"
        aws emr add-steps --cluster-id `prepend j "$CLUSTER"` --steps
'[{
                "Name": "'$NAME'",
                "Args": [
                        "/opt/wjm/bin/sperfjob", "'"$DATASETS"'",
"'"$QUERIES"'", "'"$MODES"'", "'"$inst"'", "'"$ITER"'"
                ],
                "Type": "CUSTOM_JAR",
                "Jar": "command-runner.jar",
                "ActionOnFailure": "CONTINUE"
        }]'
}
```

Appendix A: Best Practices

Spark Worker Slave EC2 Node Type Selection

Windjammer recommends using EC2 machine instances with approximately 1 Gbps network bandwidth per vCPU. EC2 instance types with an "n" in their name are network optimized and have sufficient network bandwidth. For example, depending on your workload's scaling and DRAM requirements, your worker nodes could use EC2 worker slave instances such as:

```
r5dn.4xlarge = 16 vCPUs, up to 25 Gbps network bandwidth, 128 GB DRAM
r5dn.8xlarge = 32 vCPUs, up to 25 Gbps network bandwidth, 256 GB DRAM
c5n.4xlarge  = 16 vCPUs, up to 25 Gbps network bandwidth, 42 GB DRAM
c5n.9xlarge  = 36 vCPUs, up to 50 Gbps network bandwidth, 96 GB DRAM
```

SNE Executors per worker slave node (INST)

Windjammer will default the number of SNE executors per worker EC2 slave node to the number of vCPUs/slave/2. The `emrstepstart` command script and SNE will set the default based on EC2 machine type. INST=0 selects the default SNE executor number = # of vCPUs/EC2 slave/2. If your query requires more memory, set the configuration parameter INST to # vCPUs/EC2 slave/4 to override the default so less memory for SNE executors is used. For example, if INST=0 (default) is used for EC2 r5dn.4xlarge worker slave node types, 8 SNE executors per EC2 worker slave node will be created; change INST=4 if more memory for SNE executors is required for your query.

Test Iterations

Windjammer recommends that when executing SNE performance benchmarks, at least 3 test iterations should be specified to account for EMR and network/S3 performance variability. Test iterations are specified in the `emrstepstart` command script or sperfjob command line for emr console steps.

Selecting EMR Path for TPCDS in controlled A/B benchmarking experiments

If performing TPCDS benchmarking and operating clusters in EMR region `us-east-1, us-east-2,` or `us-west-2`, then you may skip the copy of the TPCDS dataset files to a local bucket. Otherwise, because of the performance degradation of accessing a dataset across regions, you should create a bucket in the same region as the clusters you are installing and copy the files to the local data bucket.

Execute the following `aws` commands to copy the contents of the Windjammer TPCDS bucket to a cluster local TPCDS dataset bucket. An example of creating a local TPC-DS bucket in the `us-west-1` region is shown below:

- Create a dataset bucket to maintain the TPC-DS datasets in your local region if your cluster is not located in US East where you can use Windjammer's TPC DS data sets

  ```
  aws mb --region us-west-1 s3://my-tpcds-bucket
  ```

- Copy TCP-DS datasets from `s3://wjm-datasets-us-west-2` to your local dataset bucket

```
aws s3 sync s3://wjm-datasets-us-west-2/tpcds3t s3://my-tpcds-bucket
```

Enter this bucket path into the `emrstandup` script or EMR console when creating a EMR 6.2 + SNE cluster:

```
DATASETS=s3://my-data-bucket
```

The TPC-DS datasets buckets contain the 3TB dataset only. Windjammer supports the 10TB TPC-DS dataset in the S3 bucket `s3://wjm-datasets` located in region `us-east-2`. If you wish to test with the 10TB dataset, do the following:

- Create an S3 bucket in your local region.
- Copy the 10TB dataset from the `s3://wjm-datasets` bucket:
  `aws s3 cp s3://wjm-datasets/tpcds10t s3://my-tpcds-bucket`
- This will ensure optimal performance.

A/B Benchmarking of Customer Queries and Datasets

You can execute arbitrary queries using the cluster benchmarking framework.

- Create a EMR 6.15 +SNE 1.5 cluster.
- Ensure that your SQL file prepends all table names with the dataset name. The `use` statement would be overridden by the TPCDS test framework.
- Copy your SQL query file to the `/opt/wjm/queries` directory on the cluster master. It must have a file name of the form `q<number>.sql`. Query numbers below 1000 are reserved for TPCDS. Query numbers 1000-1099 are reserved for customer queries.
- Install access to your datasets on the cluster master.
- Edit the `emrstepstart` or EMR console variables and execute.
- Gather the performance report output by using the `emrstepstart` command or the EMR console Steps dashboard.

Data Cache

The Windjammer data cache will satisfy database reads by SNE from local storage. If needed data is absent from the cache then the associated Parquet files are loaded from cloud storage. The loaded data then populates the cache for future use. Three parameters control cache behavior:

WJM_CACHE_ENABLED
When set to "1" the cache facility is enabled. Default is "1".

WJM_CACHEDIRS
A comma-separated list of directories where cache files will be stored on every slave node.

Default is /mnt/[1-8], i.e. /mnt/1,/mnt/2,/mnt/3,...
If no directories are available then caching is disabled.

WJM_CACHESIZE
A numeric value that specifies the space in GiB permitted to SNE for cached files in each cache directory. Alternatively, the parameter may specify a percentage, in which case the permitted GiB is calculated from the capacity of the underlying storage device. Default parameter setting is "50%".

These parameters can be set at cluster launch as shown by the "dpstandup" script, or by equivalent means in the dataproc Dashboard. The parameters can also be changed after the cluster is running by editing the "/tmp/wj-cluster-env" file on the master node.

Spill Management

SNE avoids excessive memory use by writing intermediate data to local storage. Two parameters control spilling:

WJM_SPILLDIR
A directory on each slave node where files will be written if needed. The default varies but is typically "/mnt/1/snetmp".

WJM_SPILLTHRESHOLD
Specifies the number of rowgroups that can be memory resident before being spilled. The value is automatically scaled by the available memory on the slave in 64GiB increments. The default is 50. A value of 0 disables spilling.

Windjammer Cloud Directory

CLOUDDIR is an S3 path where SNE stores ongoing data. The default location is formed from the following template

s3://wjm-log-AAAAAAAAAAAA-RRR

where AAAAAAAAAAAA is the 12-digit AWS AccountID of the customer and RRR is the AWS Region, for example

s3://wjm-log-123456789012-us-east-2

If it doesn't exist, the hosting bucket will be created under the customer's account, read access will be granted to Windjammer for purposes of customer support, and S3 Lifecycle configurations will be installed. If the bucket already exists then no such actions are taken.

Folders under the CLOUDDIR path include

```
checkpoints/
        Subfolders contain intermediate results of queries in
        progress across all EMR clusters within the region.
        S3 Lifecycle is 3 days.
```

```
logs/
        Queries executed with the "sperf" facility are
        instrumented for performance, and results are stored in
        this folder as both text files and tarballs in the form

                log...tgz
                rep...txt

        The report files (rep...txt) are formatted for customer
        review, while the tarballs (log...tgz) are used by
        Windjammer to gain insight into SNE and cluster
        performance.  S3 Lifecycle is 2 weeks.
Aux/
        Basic diagnostics including SNE messages, physical plan,
        and EMR cluster configuration are recorded.  Two files
        are created for each evaluated query, specifically a
        tarball of the raw data, and a report summarizing SNE
        performance and usage of cluster resources.  The file
        names in the "aux/" folder include the EMR cluster ID,
        time/date stamp, and unique ID.  Example listing:

                j-2GD15B7KC1SKV_202301040308_29350.txt
                j-2GD15B7KC1SKV_202301040308_29350.tgz
```

Customer can override the default CLOUDDIR location. One of three different methods should be used depending on the way the cluster is launched.

```
Method 1: WJM standup script
----------------------------
Assign the CLOUDDIR parameter in the "emrstandup" script. Examples:

        CLOUDDIR=s3://mycloudir
        CLOUDDIR=s3://mycloudir/2022/january

Method 2: Customer standup script
---------------------------------
Adjust the bootstrap action. Example when using the "aws" command:

        --bootstrap-actions "Path=s3://wjm-build-1-
5/bootstrap,Args=[WJM_CLOUDDIR=s3://mycloudir]"

Method 3: EMR Dashboard
-------------------------------
When launching a cluster from the EMR Dashboard, use Advanced Options. On Page 3,
Additional Options, add a Bootstrap Options line as follows:

        Bootstrap action type: Custom action
        Name: Bootstrap action
```

```
Script location: s3://wjm-build-1-5/bootstrap
Optional arguments: WJM_CLOUDDIR=s3://myclouddir
```

Windjammer Support Requests

Windjammer Support Requests

Please use you Windjammer Technologies Slack channel that is created when you subscribe for any questions or support requests. For support requests regarding customer-specific queries, please include the following information:
1. The SQL query file.
2. The Spark SQL plan.
3. A sample of query data.

To generate a Spark SQL plan, follow these steps:

```
1. spark-sql --conf spark.sql.maxMetadataStringLength=33000
2. explain <sql query statements>
```