AWS re:Invent

NOV. 29 – DEC. 3, 2021 | LAS VEGAS, NV

ANT202

What’s new with Amazon EMR

Vincent Gromakowski
Principal Solutions Architect, Analytics
Amazon Web Services

Amazon EMR

BIG DATA ANALYTICS USING OPEN-SOURCE FRAMEWORKS: APACHE SPARK, PRESTO, TRINO, HADOOP, HIVE, HBASE & FLINK

Differentiated performance for runtimes

Performance-optimized runtime for popular frameworks like Spark, Hive, Presto, and Flink with 100% open-source API compatibility

Latest open-source features

New open-source features available within 30 days of release in open source

Best price-performance for big data analytics

Reduce cost using Amazon EC2 Spot, Amazon EMR managed scaling, and per-second billing

Self-service data science New!

Data science IDE with EMR Studio and deep integration with Amazon SageMaker Studio provides ability to use open-source UX and frameworks to build, visualize, and debug applications

Run workloads on Amazon EC2, Amazon EKS, or on premises New!

EMR provides flexibility to run big data workloads on EC2, EKS, and on premises with AWS Outposts

S3 data lake integration New!

Fine-grained access controls with AWS Lake Formation and Apache Ranger, and integrations with Apache HUDI to enable Amazon S3 data lake use cases

Best price-performance for big data analytics

Differentiated Spark runtime performance

Over 3x faster than standard Apache Spark 3.0 in derived TPC-DS 3 TB benchmark

Takes advantage of AWS-native Graviton2 instances to provide the best performance

100% compliance with open-source APIs makes moving applications to EMR easy

Performance improvements are enabled by default

Dynamic-sized executors

Adaptive join selection

Dynamic pruning of data columns

Operator optimization

Early worker allocation

Intelligent filtering

Parallel/async initialization

Redundant scan elimination

Data pre-fetch

Broadcast join w/o statistics

Stats inference

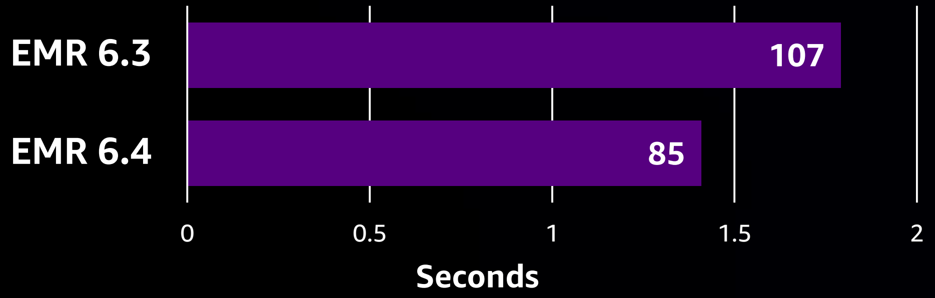
Optimized metadata fetch

Amazon EMR runtime for Apache Hive

NEW!

1.25X FASTER PERFORMANCE WITH APACHE HIVE 3.1.2 ON EMR 6.4

Geometric mean of 98 derived query runtimes
(lower is better)



Apache Hive 3.1.2 on EMR 6.4 vs. EMR 6.3*

**Based on TPC-DS 3 TB benchmarking running 16 node M5.8xlarge cluster*

EMR's performance-optimized Apache Hive runtime

Best performance

- **1.25x faster on geometric mean**
- **Up to 2x improvement on individual queries**
- Improves query planning time for AWS Glue Data Catalog
- Improves query execution time for ORC data from Amazon S3

100% compliant with open-source Apache Hive APIs

Additional capabilities to reduce costs

WITH EMR, YOU CAN DO WAY MORE WITH WAY LESS

Performance optimizations

- Runtime improvements
- Transactions in data lakes

Compute optimizations

- Graviton instances
- Spot instances
- Instance fleets

Cluster management

- Managed scaling
- Cluster auto-termination

Improvements in cluster startup times

STARTING OR SCALING AN EMR ON EC2 CLUSTER IS NOW 35% FASTER

Task nodes provisioned alongside core and primary nodes

Have ready-to-use proxy instances available to improve start time for cluster launched in private subnet

Optimized retry policies for EC2 throttling

AWS Graviton2 instances have the best price-performance within their instance families

REALIZE UP TO 30% BETTER PRICE-PERFORMANCE WITH GRAVITON2 INSTANCES

12%–16% improvement in performance
compared to M5 instance types

20% lower cost
vs. same-sized comparable M5 instances

Up to 30% better price-performance

Amazon EC2 Spot Instances

ACCELERATE COMPUTE FOR LESS

Low, predictable prices

Up to 90% discount over On-Demand prices

Faster results

Increase throughput up to 10x while staying in budget

Easy to use

Launch through AWS services (ex. ECS, EKS, AWS Batch, EMR) or integrated third parties

Benefits of EMR with Spot Instances

EMR IS THE BEST FIT FOR RUNNING BIG DATA WORKLOADS USING SPOT INSTANCES

Accelerate compute

Run parallel tasks on a multitude of instance types running Spark, Hive, Flink, or Presto

Further reduce costs

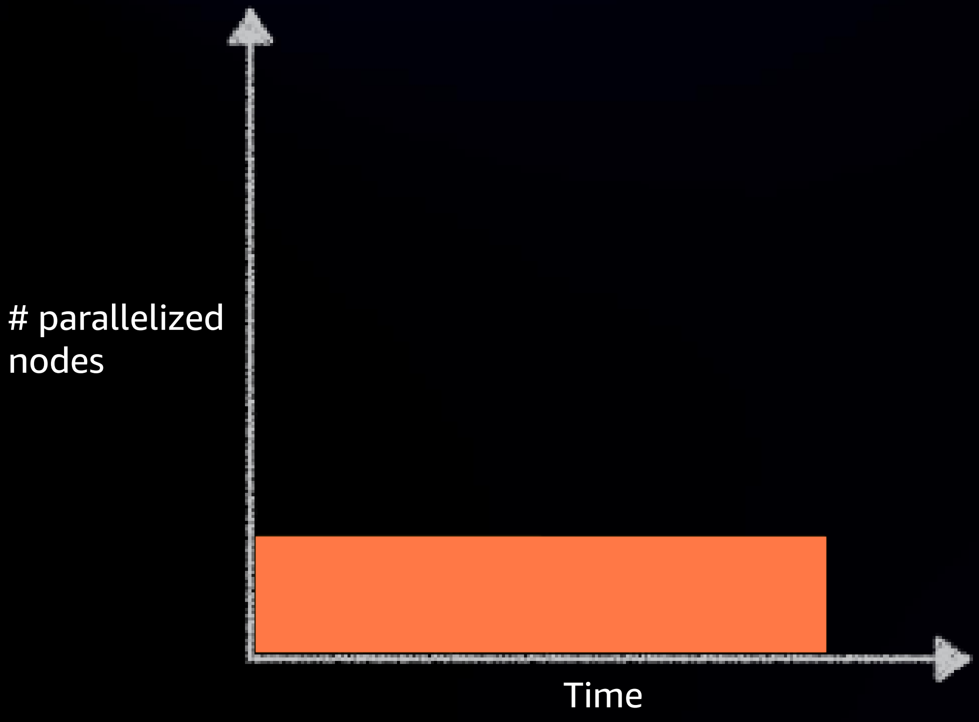
Access instances at up to a 90% discount vs. On-Demand

Build for scale

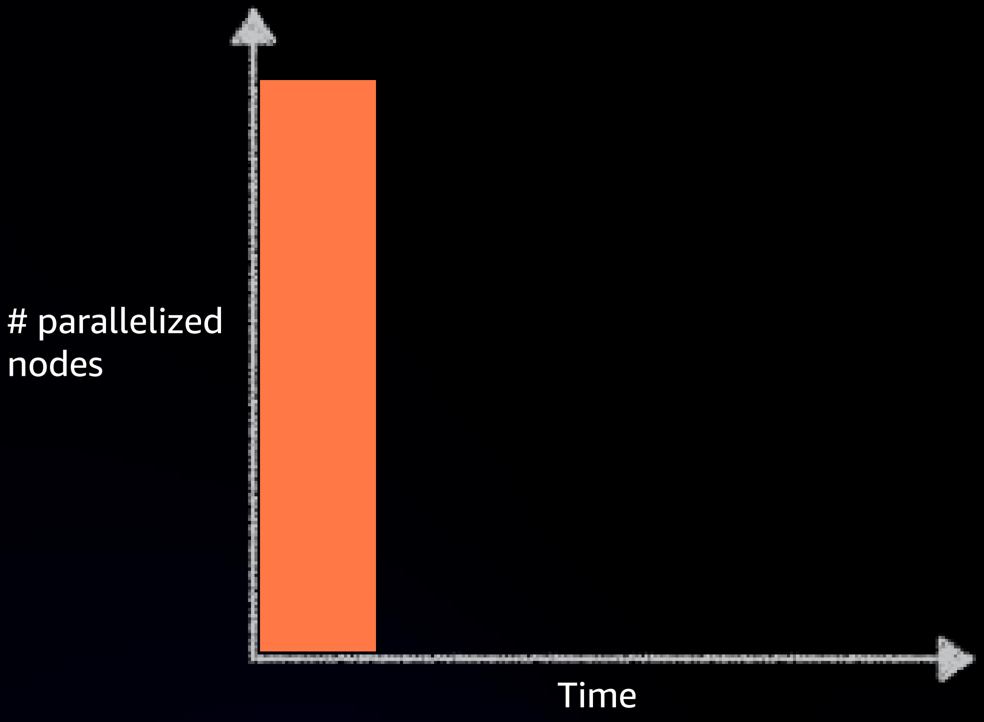
Quickly ramp up short-lived but massive data jobs by scaling compute and storage independently

Amazon EMR on Spot Instances

REDUCE TIME-TO-INSIGHT WITH HYPER-PARALLELIZED WORKLOADS

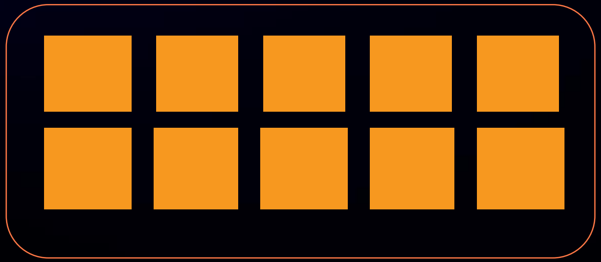


Job running time: **10 hours**



Job running time: **1 hour**

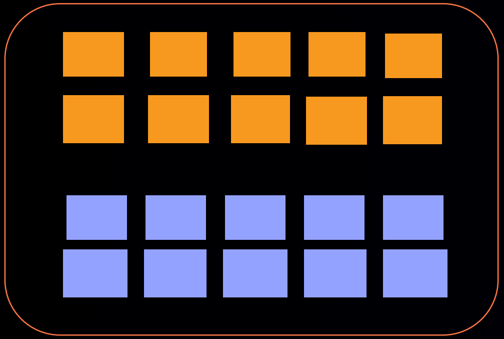
Scale up cluster with Spot Instances



10-node cluster running for 14 hours

Cost = 1.0 * 10 * 14 = $140

Add 10 more nodes on Spot



20-node cluster running for 7 hours

Cost = 1.0 * 10 * 7 = $70

= 0.5 * 10 * 7 = $35

Total $105

50% less runtime (hours: 14 → 7)

25% less cost (dollars: 140 → 105)

Use a mix of On-Demand and Spot Instances for different scenarios

Scenario	Leader node	Core nodes	Task nodes
Long-running clusters and data warehouses	On-Demand	On-Demand or instance-fleet mix	Spot or instance-fleet mix
Cost-driven workloads (without a production SLA)	Spot	Spot	Spot
Data-critical workloads	On-Demand	On-Demand	Spot or instance-fleet mix
Application testing	Spot	Spot	Spot

Amazon EMR on Spot Instances

EMR IS THE BEST PLACE TO RUN BIG DATA WORKLOADS USING SPOT CAPACITY

Save 75%–90% on compute with Spot Instances

Low, predictable prices

Minimal interruptions, <5%

No bidding

Salesforce has observed an increase in iterative job performance and saved 80% vs. On-Demand pricing

"With AWS, we can manage flexible capacity changes, contain overall costs on daily compute tasks, and manage overall infrastructure growth."

Roopak Gupta

Vice President, Software Engineering, Salesforce DMP

Managed scaling feature overview

AUTOMATICALLY REDUCE COST BY SHAPING CLUSTER SIZE

Constantly improving EMR managed algorithm that gives you a fully managed experience

High-resolution metrics enabled with managed scaling

Only min/max cost constraints configurations required

More data points and faster reaction time than auto scaling

Save 20%–60% of costs

Managed scaling enhancements

NEW!

NEW ENHANCEMENTS ENABLED BY DEFAULT TO FURTHER REDUCE COSTS AND SPOT INTERRUPTIONS

Capacity awareness in instance groups enabled by default for all supported EMR versions

Integrated with real-time EC2 Spot capacity metrics to scale the right task group based on instance pool depth

Shuffle awareness enabled by default from EMR 6.4

Ensure nodes with active shuffle data are not scaled down

Support for PrestoDB and Trino available from EMR 6.4

Sign up for preview access via aws-support@amazon.com

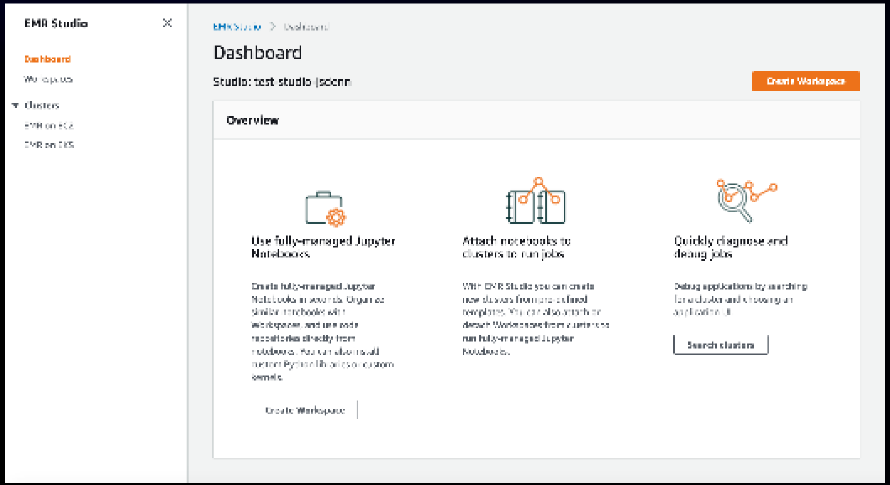
“Acxiom uses Spark on Amazon EMR on Spot Instances to run 3 trillion inferences in less than 15 hours. By using Amazon EMR, we could utilize Spot compute capacity across the entire AWS Region and speed up the run time of our inference pipeline that typically took 11–15 days every month to under 15 hours.”

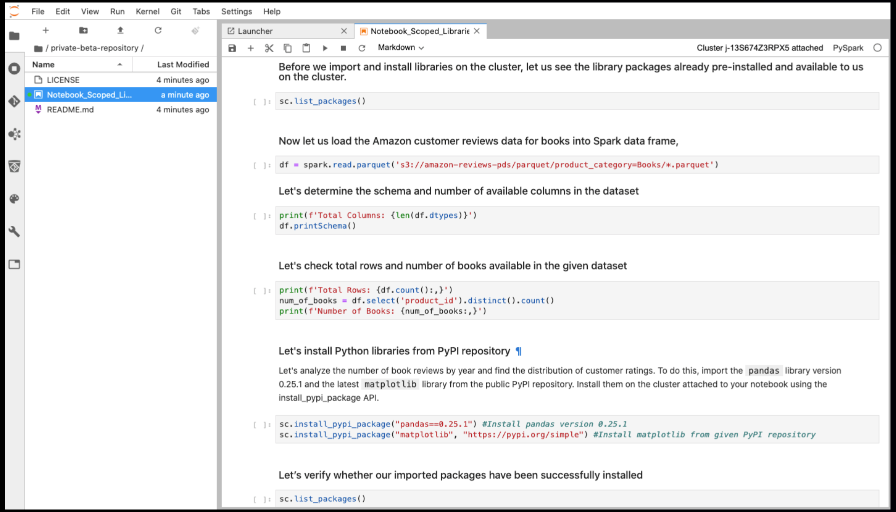
Varadarajan “Raj” Srinivasan
Sr. Director, ML Engineering and Data Science, Acxiom

Self-service data science with EMR Studio and Amazon SageMaker Studio

EMR Studio

FULLY MANAGED IDE FOR INTERACTIVE DATA ANALYTICS: DEVELOP, VISUALIZE, AND DEBUG APPLICATIONS


EMR Studio
Dashboard
Workspaces
Clusters
EMR on EC2
EMR on EKS
Dashboard
Studio: test studio [illegible]
Create Workspace
Overview
Use fully-managed Jupyter Notebooks
Attach notebooks to clusters to run jobs
Quickly diagnose and debug jobs
Search clusters
Create Workspace



Before we import and install libraries on the cluster, let us see the library packages already pre-installed and available to us on the cluster.

```
[ ]: sc.list_packages()
```

Now let us load the Amazon customer reviews data for books into Spark data frame,

```
[ ]: df = spark.read.parquet('s3://amazon-reviews-pds/parquet/product_category=Books/*.parquet')
```

Let's determine the schema and number of available columns in the dataset

```
[ ]: print(f'Total Columns: {len(df.dtypes)}')
     df.printSchema()
```

Let's check total rows and number of books available in the given dataset

```
[ ]: print(f'Total Rows: {df.count():,}')
     num_of_books = df.select('product_id').distinct().count()
     print(f'Number of Books: {num_of_books:,}')
```

Let's install Python libraries from PyPI repository ¶

Let's analyze the number of book reviews by year and find the distribution of customer ratings. To do this, import the `pandas` library version 0.25.1 and the latest `matplotlib` library from the public PyPI repository. Install them on the cluster attached to your notebook using the install_pypi_package API.

```
[ ]: sc.install_pypi_package("pandas==0.25.1") #Install pandas version 0.25.1
     sc.install_pypi_package("matplotlib", "https://pypi.org/simple") #Install matplotlib from given PyPI repository
```

Let's verify whether our imported packages have been successfully installed

```
[ ]: sc.list_packages()
```

Single sign-on integration with IdP

Fully managed Jupyter Notebooks

Integrated with Git Repositories

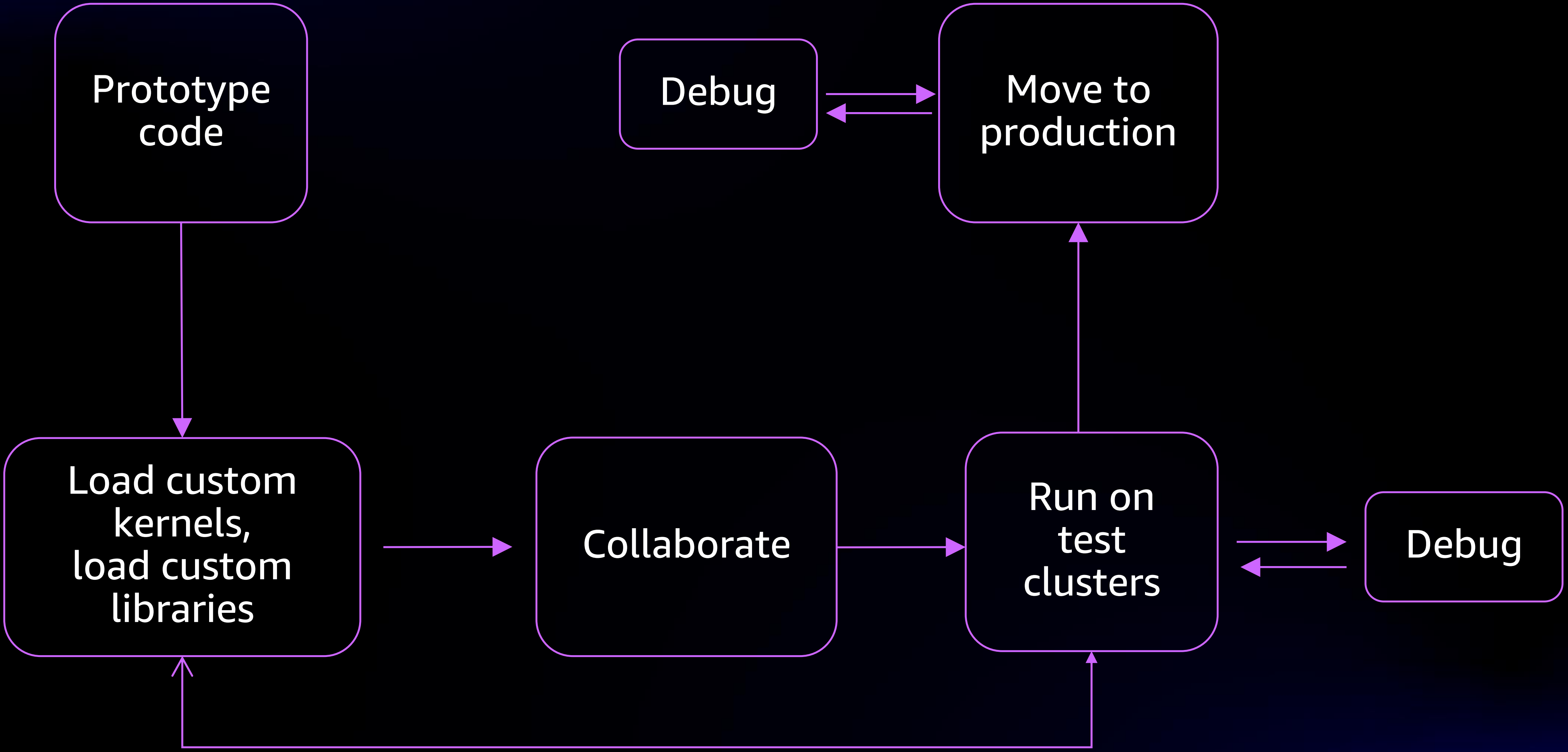
Simplified debugging with Spark UI and YARN UI

Browse, create, or delete EMR clusters

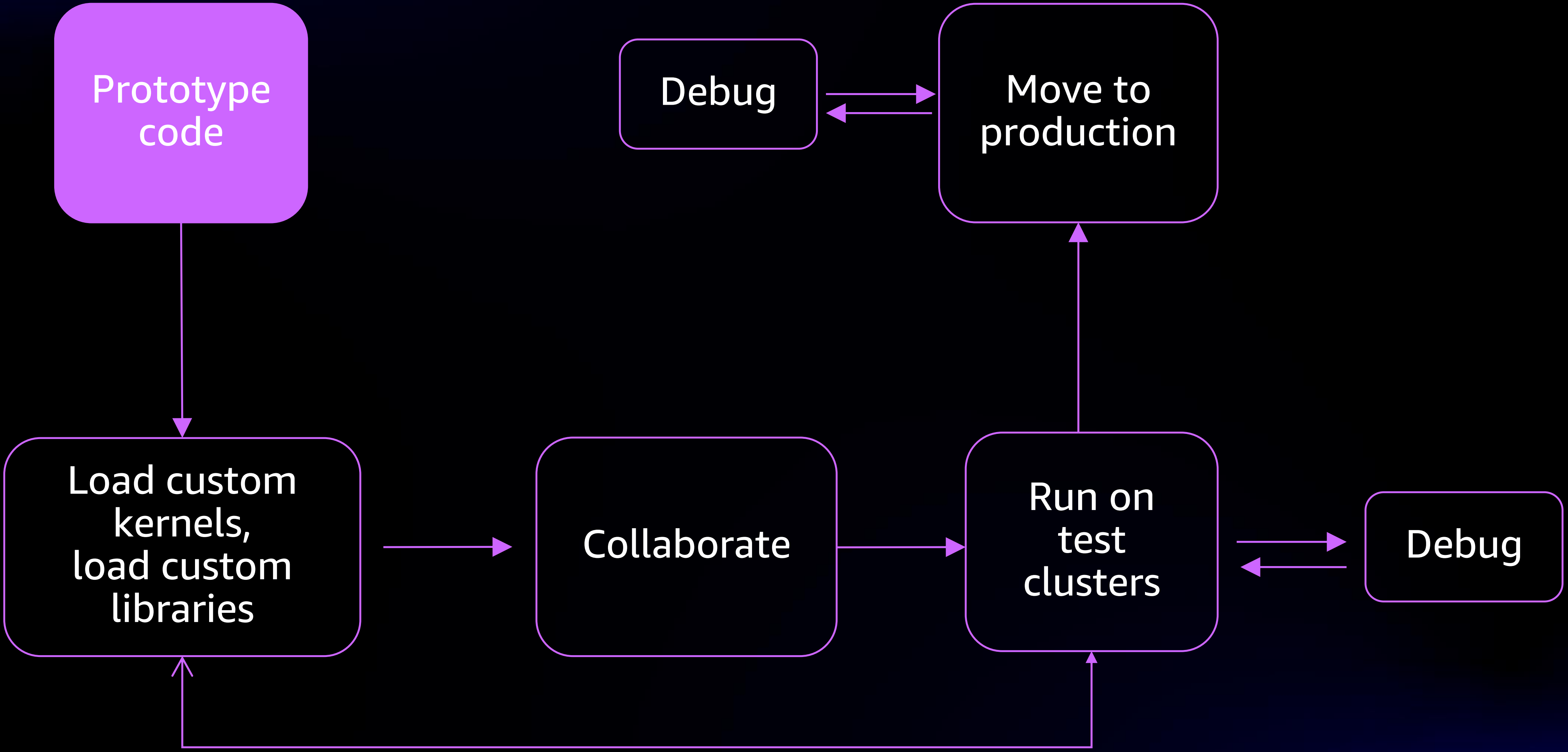
Run Notebooks in workflows using APIs

Run interactive data analysis using EMR or EKS clusters

Data science and engineering workflows



Data science and engineering workflows

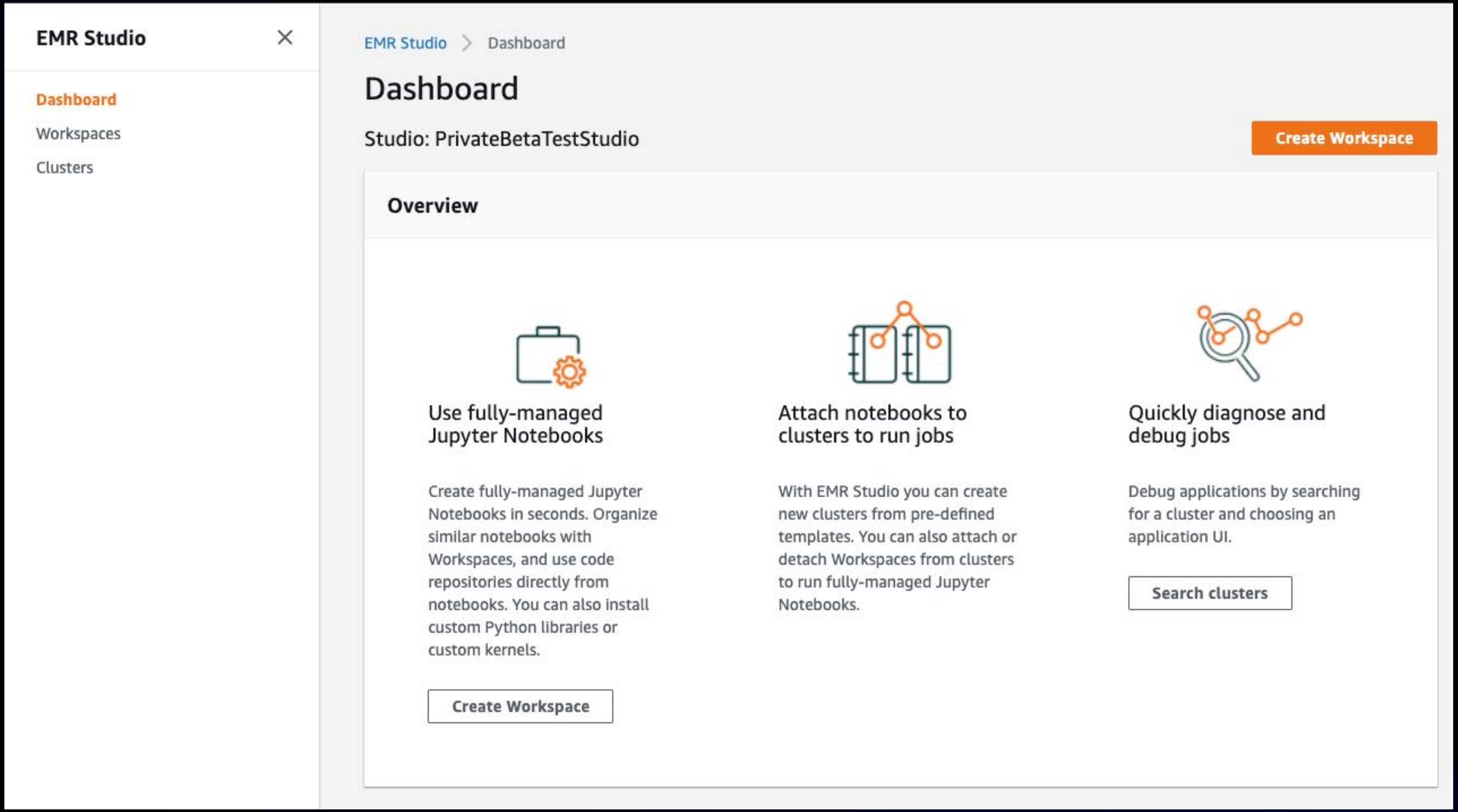


Log in to EMR Studio without logging into the AWS Management Console

Data scientist

Data engineer

Log in with corporate identity using AWS Single Sign-On



Dashboard

Studio: PrivateBetaTestStudio

Overview

Use fully-managed Jupyter Notebooks

Create fully-managed Jupyter Notebooks in seconds. Organize similar notebooks with Workspaces, and use code repositories directly from notebooks. You can also install custom Python libraries or custom kernels.

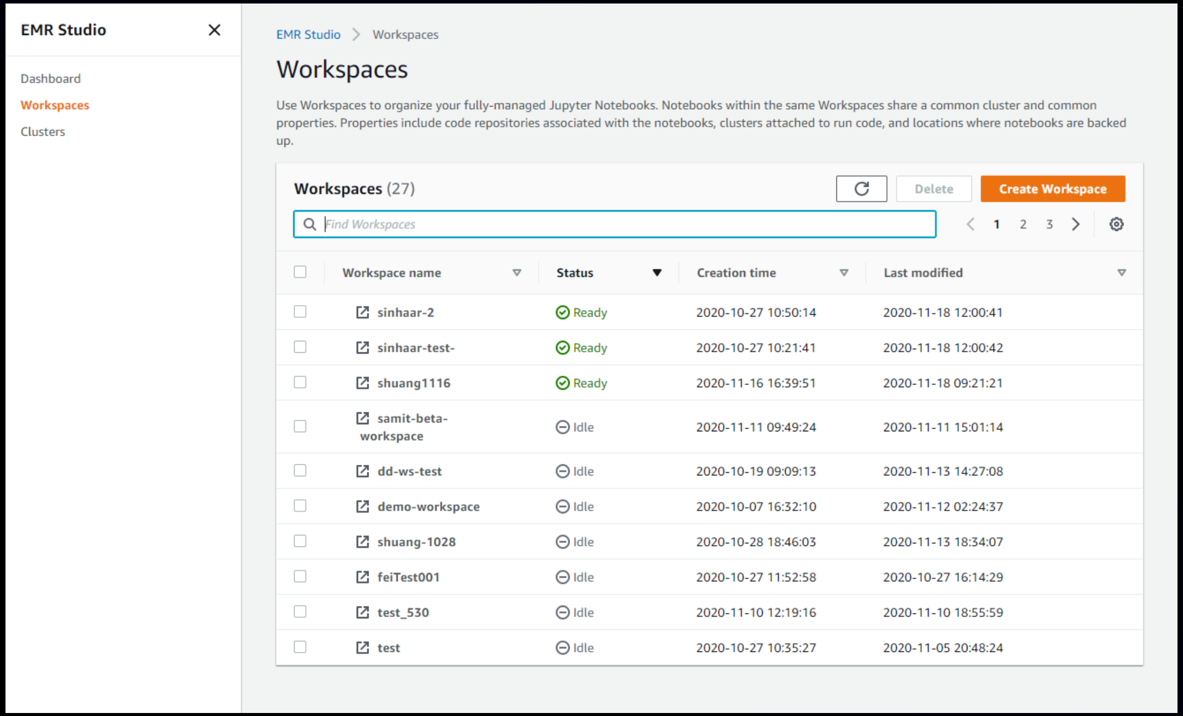
Attach notebooks to clusters to run jobs

With EMR Studio you can create new clusters from pre-defined templates. You can also attach or detach Workspaces from clusters to run fully-managed Jupyter Notebooks.

Quickly diagnose and debug jobs

Debug applications by searching for a cluster and choosing an application UI.

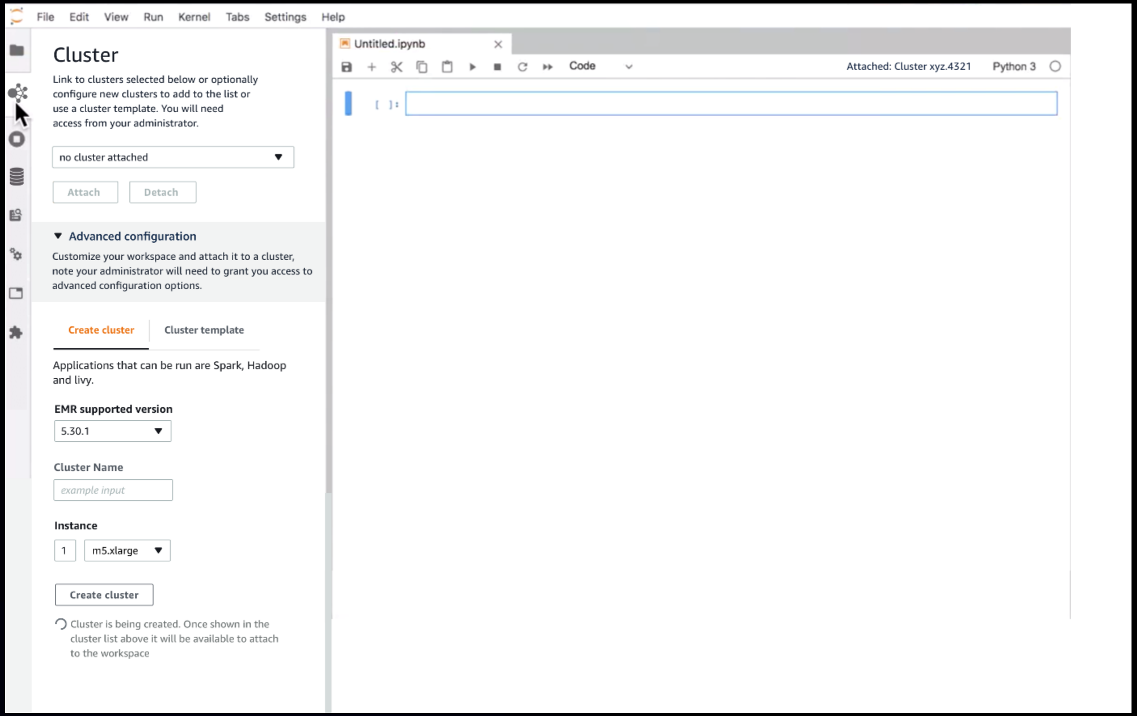
EMR Studio gives you a fully managed notebook



Workspaces

Use Workspaces to organize your fully-managed Jupyter Notebooks. Notebooks within the same Workspaces share a common cluster and common properties. Properties include code repositories associated with the notebooks, clusters attached to run code, and locations where notebooks are backed up.

Workspace name	Status	Creation time	Last modified
sinhaar-2	Ready	2020-10-27 10:50:14	2020-11-18 12:00:41
sinhaar-test-	Ready	2020-10-27 10:21:41	2020-11-18 12:00:42
shuang1116	Ready	2020-11-16 16:39:51	2020-11-18 09:21:21
samit-beta-workspace	Idle	2020-11-11 09:49:24	2020-11-11 15:01:14
dd-ws-test	Idle	2020-10-19 09:09:13	2020-11-13 14:27:08
demo-workspace	Idle	2020-10-07 16:32:10	2020-11-12 02:24:37
shuang-1028	Idle	2020-10-28 18:46:03	2020-11-13 18:34:07
feiTest001	Idle	2020-10-27 11:52:58	2020-10-27 16:14:29
test_530	Idle	2020-11-10 12:19:16	2020-11-10 18:55:59
test	Idle	2020-10-27 10:35:27	2020-11-05 20:48:24


File Edit View Run Kernel Tabs Settings Help
Untitled.ipynb
Code
Attached: Cluster xyz.4321
Python 3
[ ]:

Cluster

Link to clusters selected below or optionally configure new clusters to add to the list or use a cluster template. You will need access from your administrator.

no cluster attached

Advanced configuration

Customize your workspace and attach it to a cluster, note your administrator will need to grant you access to advanced configuration options.

Applications that can be run are Spark, Hadoop and livy.

EMR supported version: 5.30.1

Cluster Name: *example input*

Instance: 1 m5.xlarge

Cluster is being created. Once shown in the cluster list above it will be available to attach to the workspace

Workspaces help organize notebooks

Workspaces share similar properties

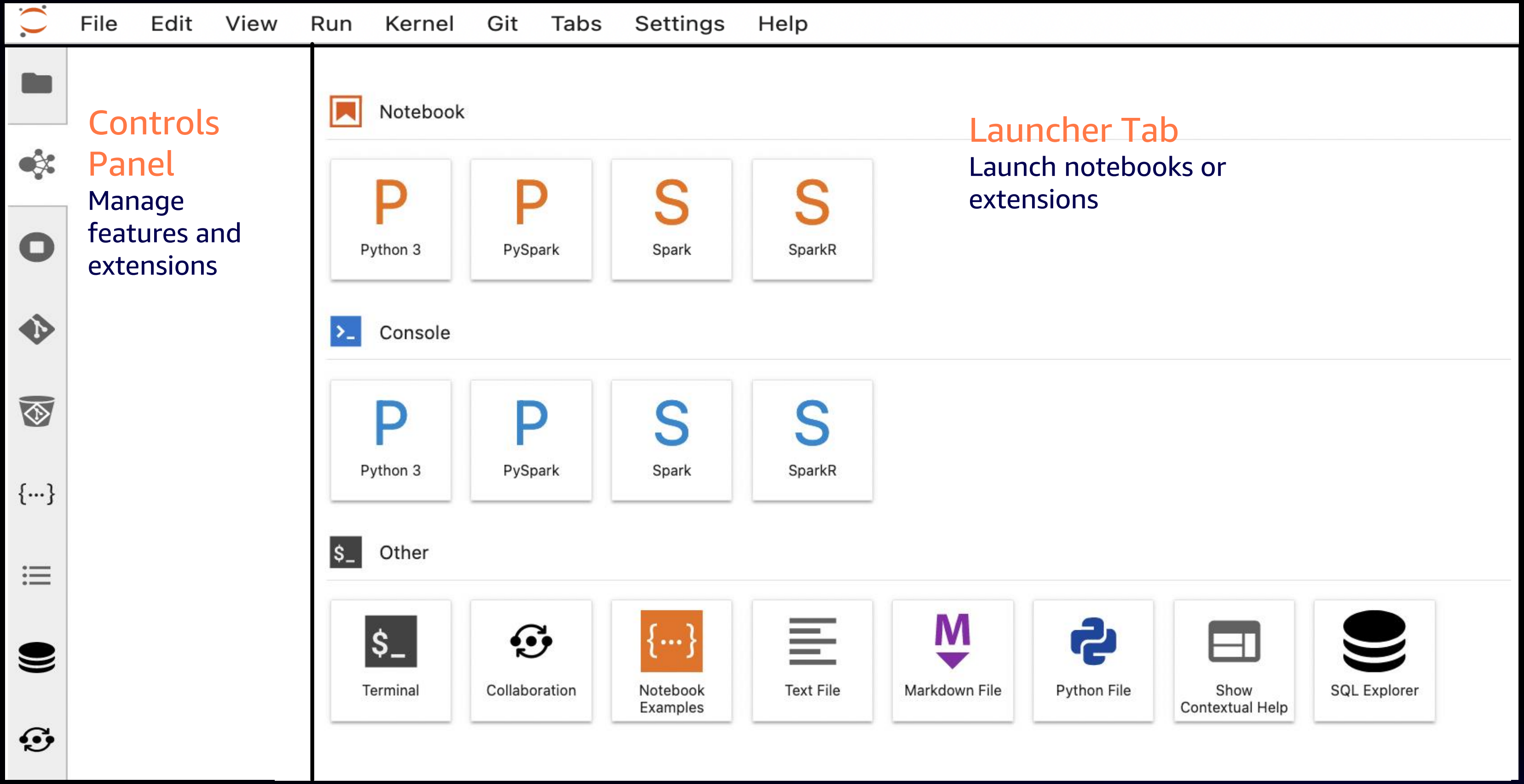
Fully managed Jupyter Notebooks

Write Python, R, PySpark, Scala

Workspace: Single IDE for interactive data analysis

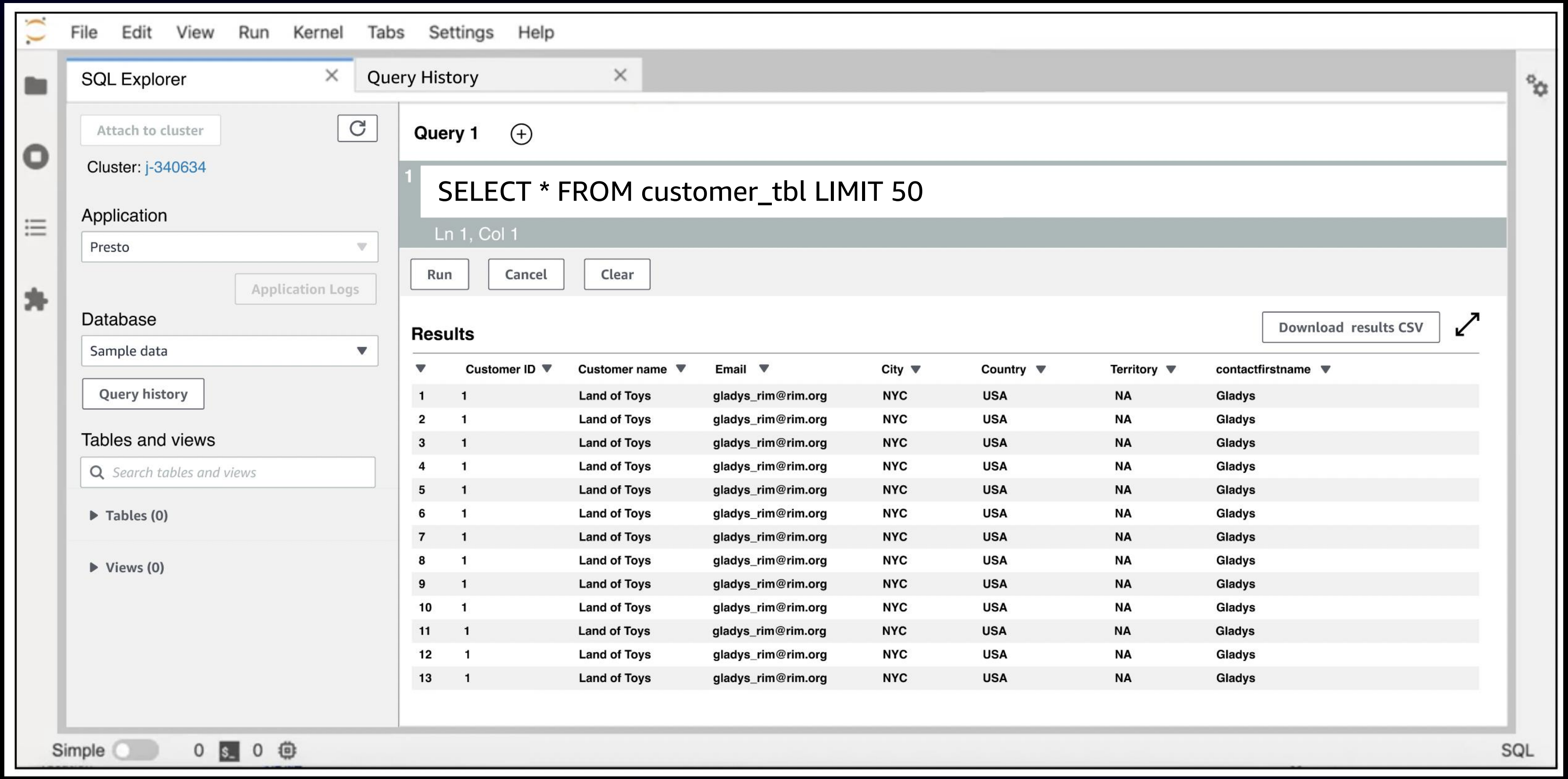
WITH CURATED LIST OF EXTENSIONS TO ENHANCE THE JUPYTER EXPERIENCE

- File browser
- **Attach/detach EMR clusters**
- Kernel management
- Git operations
- **Workspace Git link**
- **Sample notebooks**
- Table of contents
- COMING SOON **SQL Explorer**
- COMING SOON **Collaboration**



SQL Explorer integrated in Jupyter

COMING SOON



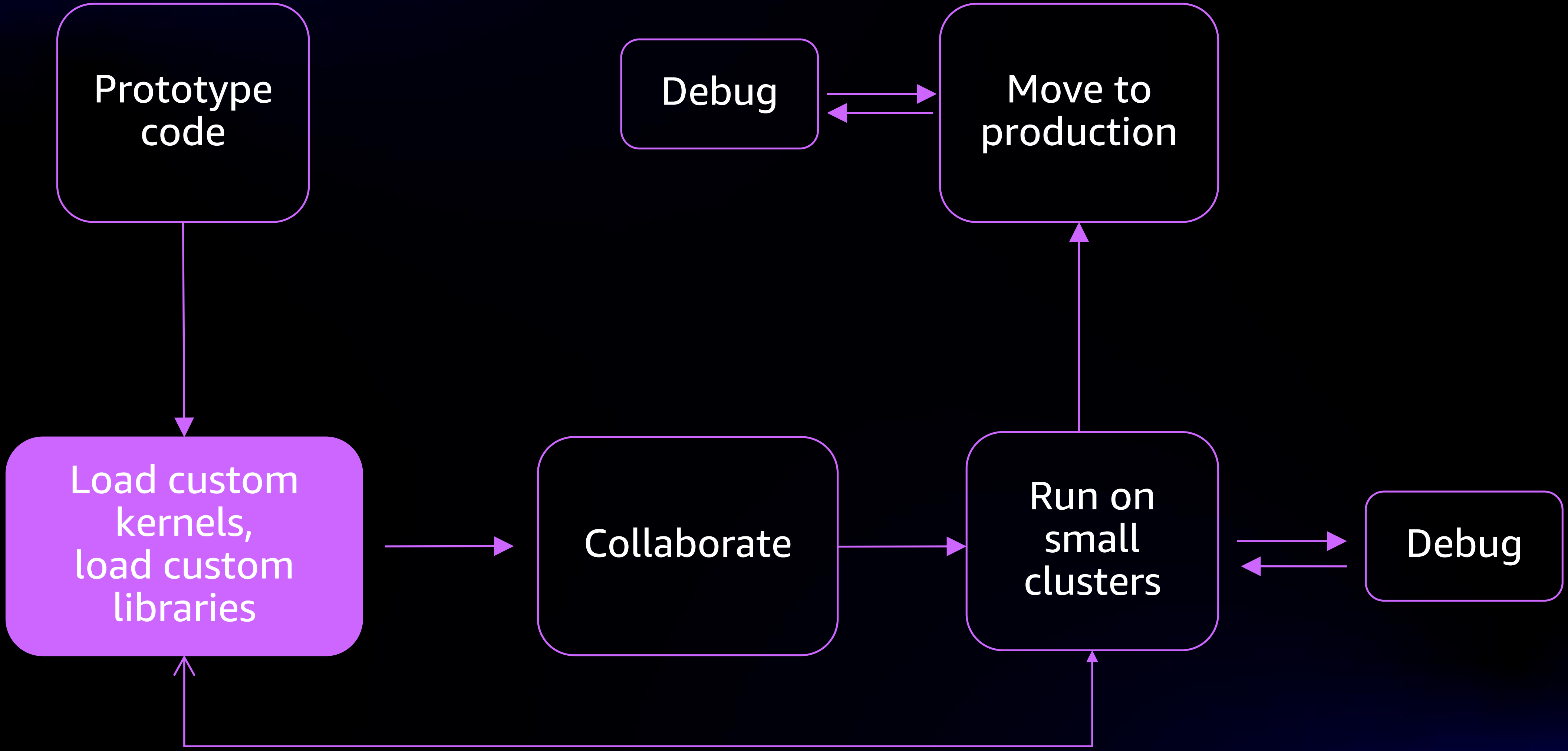
```
SELECT * FROM customer_tbl LIMIT 50
```

	Customer ID	Customer name	Email	City	Country	Territory	contactfirstname
1	1	Land of Toys	gladys_rim@rim.org	NYC	USA	NA	Gladys
2	1	Land of Toys	gladys_rim@rim.org	NYC	USA	NA	Gladys
3	1	Land of Toys	gladys_rim@rim.org	NYC	USA	NA	Gladys
4	1	Land of Toys	gladys_rim@rim.org	NYC	USA	NA	Gladys
5	1	Land of Toys	gladys_rim@rim.org	NYC	USA	NA	Gladys
6	1	Land of Toys	gladys_rim@rim.org	NYC	USA	NA	Gladys
7	1	Land of Toys	gladys_rim@rim.org	NYC	USA	NA	Gladys
8	1	Land of Toys	gladys_rim@rim.org	NYC	USA	NA	Gladys
9	1	Land of Toys	gladys_rim@rim.org	NYC	USA	NA	Gladys
10	1	Land of Toys	gladys_rim@rim.org	NYC	USA	NA	Gladys
11	1	Land of Toys	gladys_rim@rim.org	NYC	USA	NA	Gladys
12	1	Land of Toys	gladys_rim@rim.org	NYC	USA	NA	Gladys
13	1	Land of Toys	gladys_rim@rim.org	NYC	USA	NA	Gladys

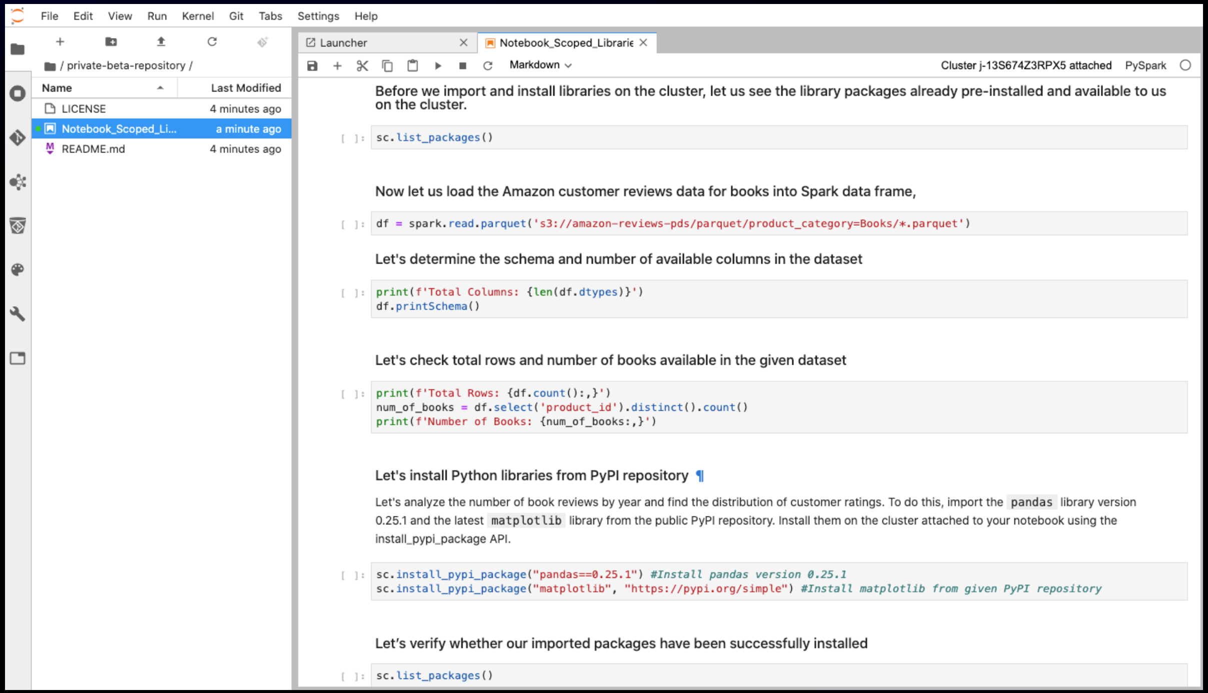
Data catalog browser

Multiple SQL editors

Data science and engineering workflows



Simple to load custom libraries and kernels



Cluster j-13S674Z3RPX5 attached PySpark

Before we import and install libraries on the cluster, let us see the library packages already pre-installed and available to us on the cluster.

```
sc.list_packages()
```

Now let us load the Amazon customer reviews data for books into Spark data frame,

```
df = spark.read.parquet('s3://amazon-reviews-pds/parquet/product_category=Books/*.parquet')
```

Let's determine the schema and number of available columns in the dataset

```
print(f'Total Columns: {len(df.dtypes)}')
df.printSchema()
```

Let's check total rows and number of books available in the given dataset

```
print(f'Total Rows: {df.count():,}')
num_of_books = df.select('product_id').distinct().count()
print(f'Number of Books: {num_of_books:,}')
```

Let's install Python libraries from PyPI repository ¶

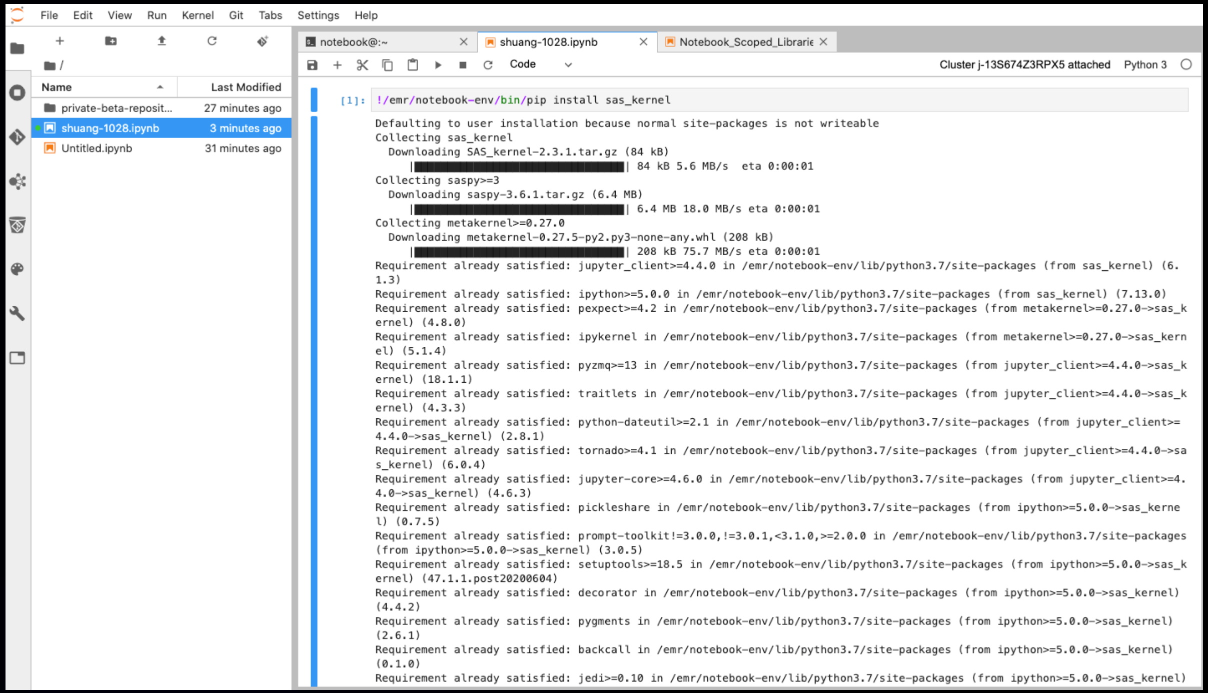
Let's analyze the number of book reviews by year and find the distribution of customer ratings. To do this, import the `pandas` library version 0.25.1 and the latest `matplotlib` library from the public PyPI repository. Install them on the cluster attached to your notebook using the install_pypi_package API.

```
sc.install_pypi_package("pandas==0.25.1") #Install pandas version 0.25.1
sc.install_pypi_package("matplotlib", "https://pypi.org/simple") #Install matplotlib from given PyPI repository
```

Let's verify whether our imported packages have been successfully installed

```
sc.list_packages()
```

File Edit View Run Kernel Git Tabs Settings Help
/ private-beta-repository /
Name Last Modified
LICENSE 4 minutes ago
Notebook_Scoped_Li... a minute ago
README.md 4 minutes ago
Launcher
Notebook_Scoped_Librarie
Markdown



Cluster j-13S674Z3RPX5 attached Python 3

```
[1]: !/emr/notebook-env/bin/pip install sas_kernel
```

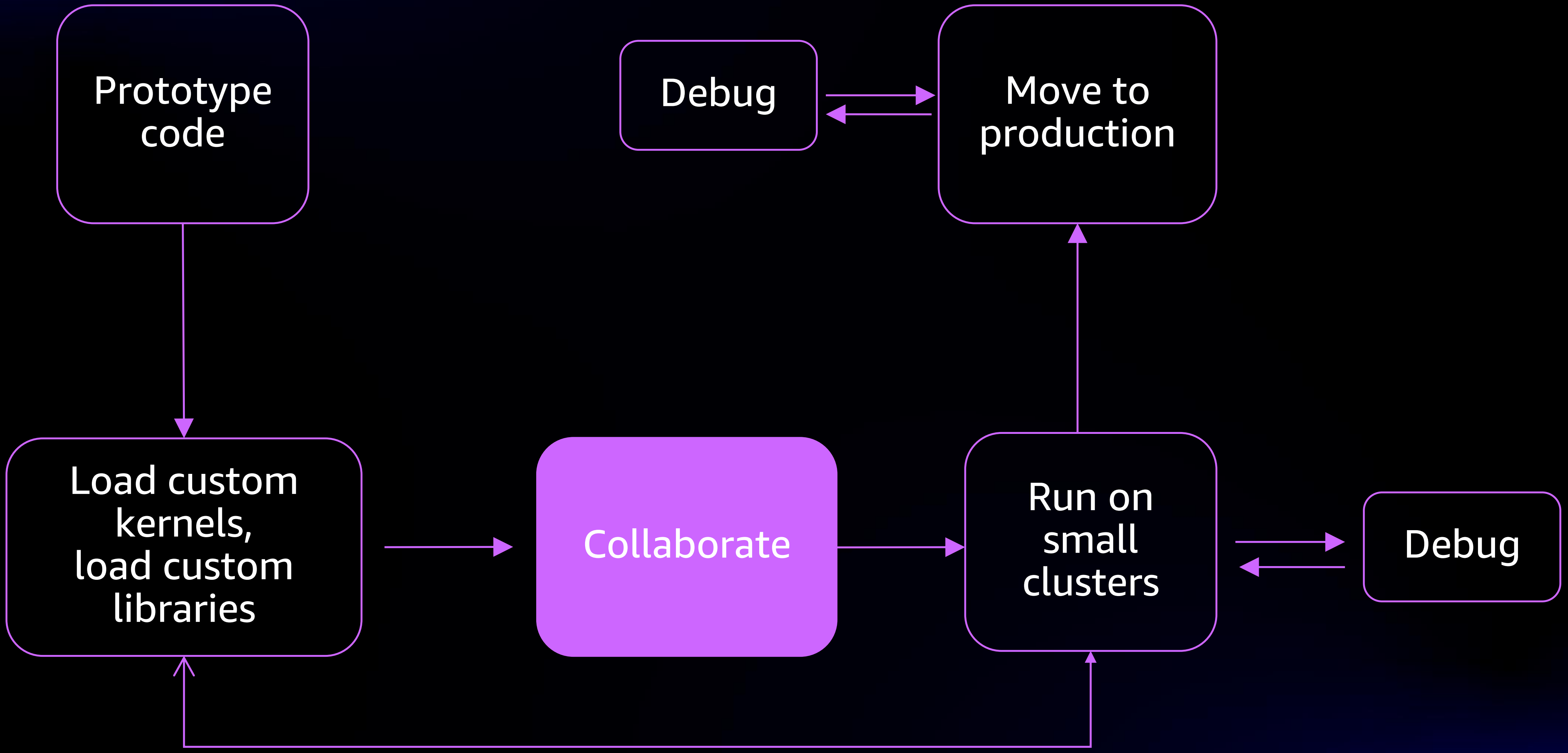
```
Defaulting to user installation because normal site-packages is not writeable
Collecting sas_kernel
  Downloading SAS_kernel-2.3.1.tar.gz (84 kB)
     |████████████████████████████████| 84 kB 5.6 MB/s  eta 0:00:01
Collecting saspy>=3
  Downloading saspy-3.6.1.tar.gz (6.4 MB)
     |████████████████████████████████| 6.4 MB 18.0 MB/s eta 0:00:01
Collecting metakernel>=0.27.0
  Downloading metakernel-0.27.5-py2.py3-none-any.whl (208 kB)
     |████████████████████████████████| 208 kB 75.7 MB/s eta 0:00:01
Requirement already satisfied: jupyter_client>=4.4.0 in /emr/notebook-env/lib/python3.7/site-packages (from sas_kernel) (6.
1.3)
Requirement already satisfied: ipython>=5.0.0 in /emr/notebook-env/lib/python3.7/site-packages (from sas_kernel) (7.13.0)
Requirement already satisfied: pexpect>=4.2 in /emr/notebook-env/lib/python3.7/site-packages (from metakernel>=0.27.0->sas_k
ernel) (4.8.0)
Requirement already satisfied: ipykernel in /emr/notebook-env/lib/python3.7/site-packages (from metakernel>=0.27.0->sas_kern
el) (5.1.4)
Requirement already satisfied: pyzmq>=13 in /emr/notebook-env/lib/python3.7/site-packages (from jupyter_client>=4.4.0->sas_k
ernel) (18.1.1)
Requirement already satisfied: traitlets in /emr/notebook-env/lib/python3.7/site-packages (from jupyter_client>=4.4.0->sas_k
ernel) (4.3.3)
Requirement already satisfied: python-dateutil>=2.1 in /emr/notebook-env/lib/python3.7/site-packages (from jupyter_client>=
4.4.0->sas_kernel) (2.8.1)
Requirement already satisfied: tornado>=4.1 in /emr/notebook-env/lib/python3.7/site-packages (from jupyter_client>=4.4.0->sa
s_kernel) (6.0.4)
Requirement already satisfied: jupyter-core>=4.6.0 in /emr/notebook-env/lib/python3.7/site-packages (from jupyter_client>=4.
4.0->sas_kernel) (4.6.3)
Requirement already satisfied: pickleshare in /emr/notebook-env/lib/python3.7/site-packages (from ipython>=5.0.0->sas_kerne
l) (0.7.5)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in /emr/notebook-env/lib/python3.7/site-packages
(from ipython>=5.0.0->sas_kernel) (3.0.5)
Requirement already satisfied: setuptools>=18.5 in /emr/notebook-env/lib/python3.7/site-packages (from ipython>=5.0.0->sas_k
ernel) (47.1.1.post20200604)
Requirement already satisfied: decorator in /emr/notebook-env/lib/python3.7/site-packages (from ipython>=5.0.0->sas_kernel)
(4.4.2)
Requirement already satisfied: pygments in /emr/notebook-env/lib/python3.7/site-packages (from ipython>=5.0.0->sas_kernel)
(2.6.1)
Requirement already satisfied: backcall in /emr/notebook-env/lib/python3.7/site-packages (from ipython>=5.0.0->sas_kernel)
(0.1.0)
Requirement already satisfied: jedi>=0.10 in /emr/notebook-env/lib/python3.7/site-packages (from ipython>=5.0.0->sas_kernel)
```

File Edit View Run Kernel Git Tabs Settings Help
/
Name Last Modified
private-beta-reposit... 27 minutes ago
shuang-1028.ipynb 3 minutes ago
Untitled.ipynb 31 minutes ago
notebook@:~
shuang-1028.ipynb
Notebook_Scoped_Librarie
Code

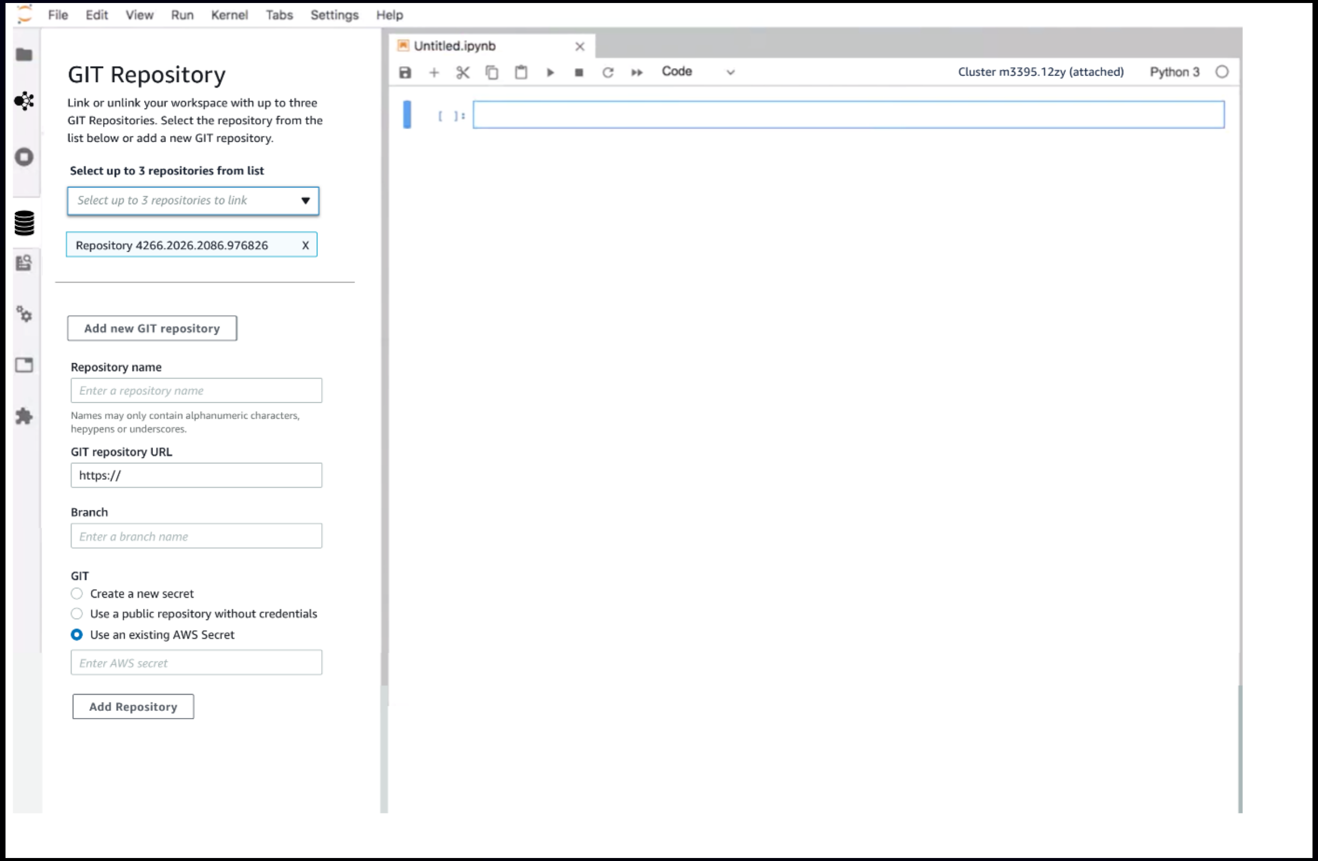
Install notebook-scoped libraries with PySpark kernel

Install additional Python libraries and kernels on the leader node of the cluster

Data science and engineering workflows



Simple to connect to code repositories



GIT Repository

Link or unlink your workspace with up to three GIT Repositories. Select the repository from the list below or add a new GIT repository.

Select up to 3 repositories from list

Select up to 3 repositories to link

Repository 4266.2026.2086.976826 X

Add new GIT repository

Repository name

Enter a repository name

Names may only contain alphanumeric characters, hepypens or underscores.

GIT repository URL

https://

Branch

Enter a branch name

GIT

Create a new secret

Use a public repository without credentials

[✓ Use an existing AWS Secret]

Enter AWS secret

Add Repository

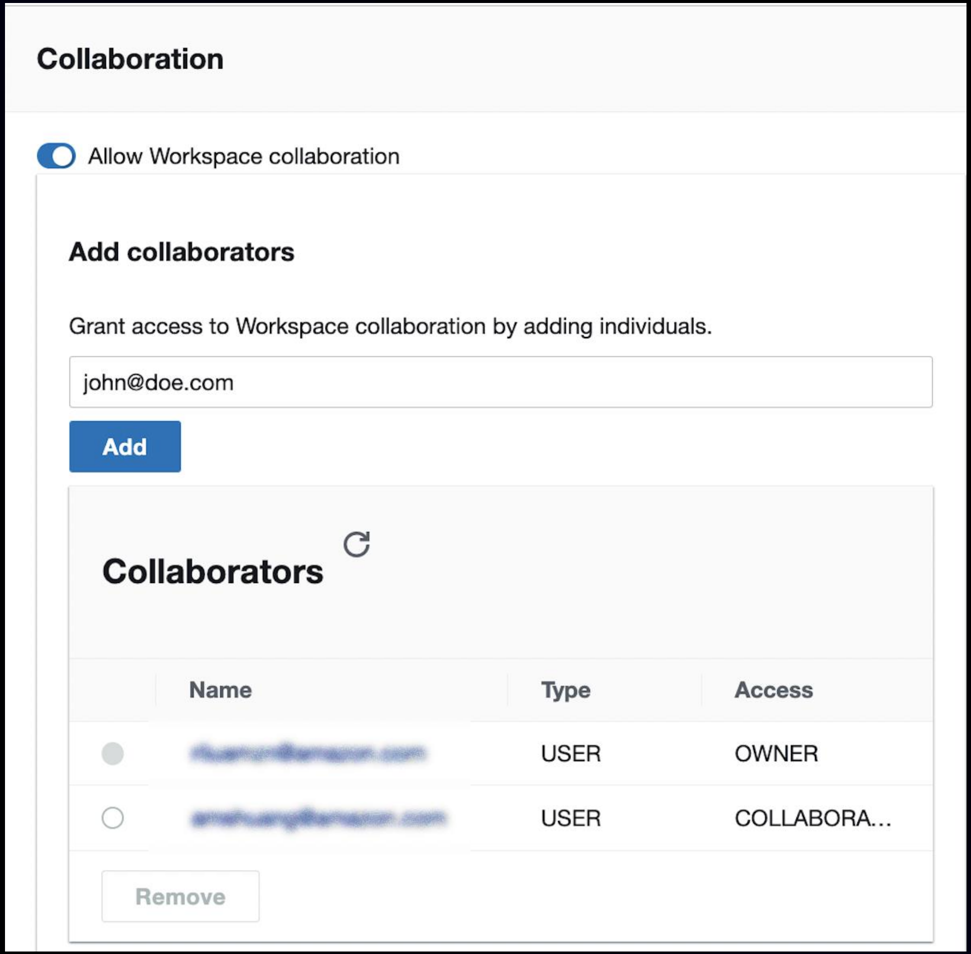
Cluster m3395.12zy (attached)

Connect to AWS CodeCommit, GitHub, and Bitbucket

Select existing or add new Git repositories

Collaborate in real time

COMING SOON



Collaboration

Allow Workspace collaboration

Add collaborators

Grant access to Workspace collaboration by adding individuals.

john@doe.com

Collaborators

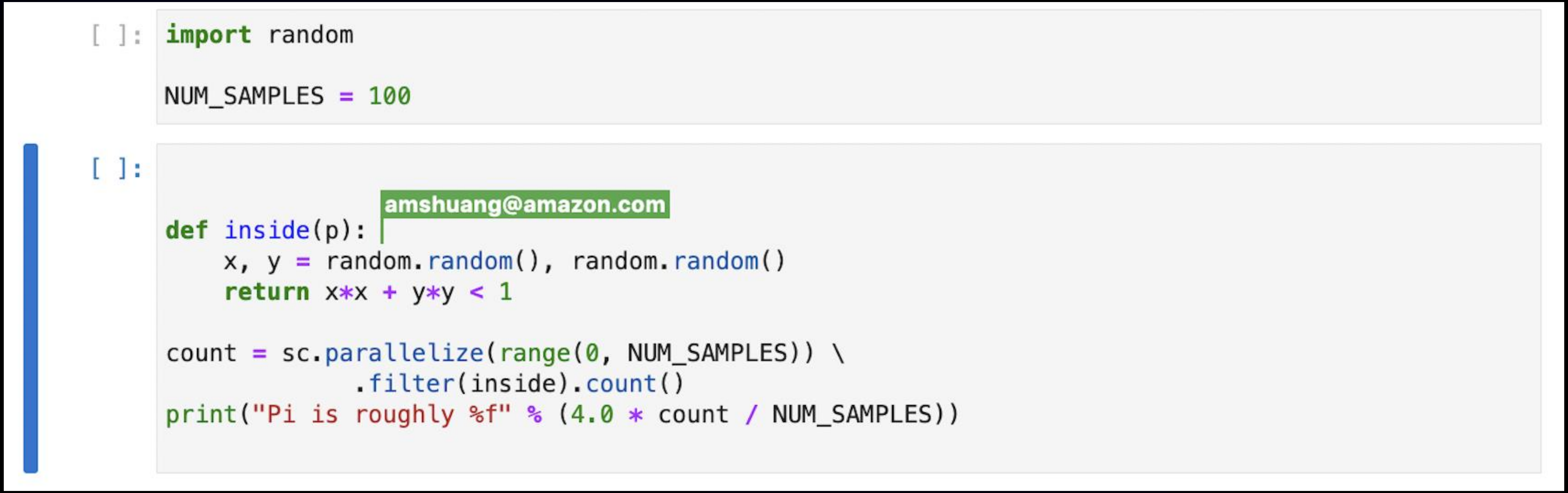
	Name	Type	Access
●	[redacted]	USER	OWNER
○	[redacted]	USER	COLLABORA...

Enable Workspace collaboration

Invite collaborators to your Workspace

Collaborate in real time

COMING SOON

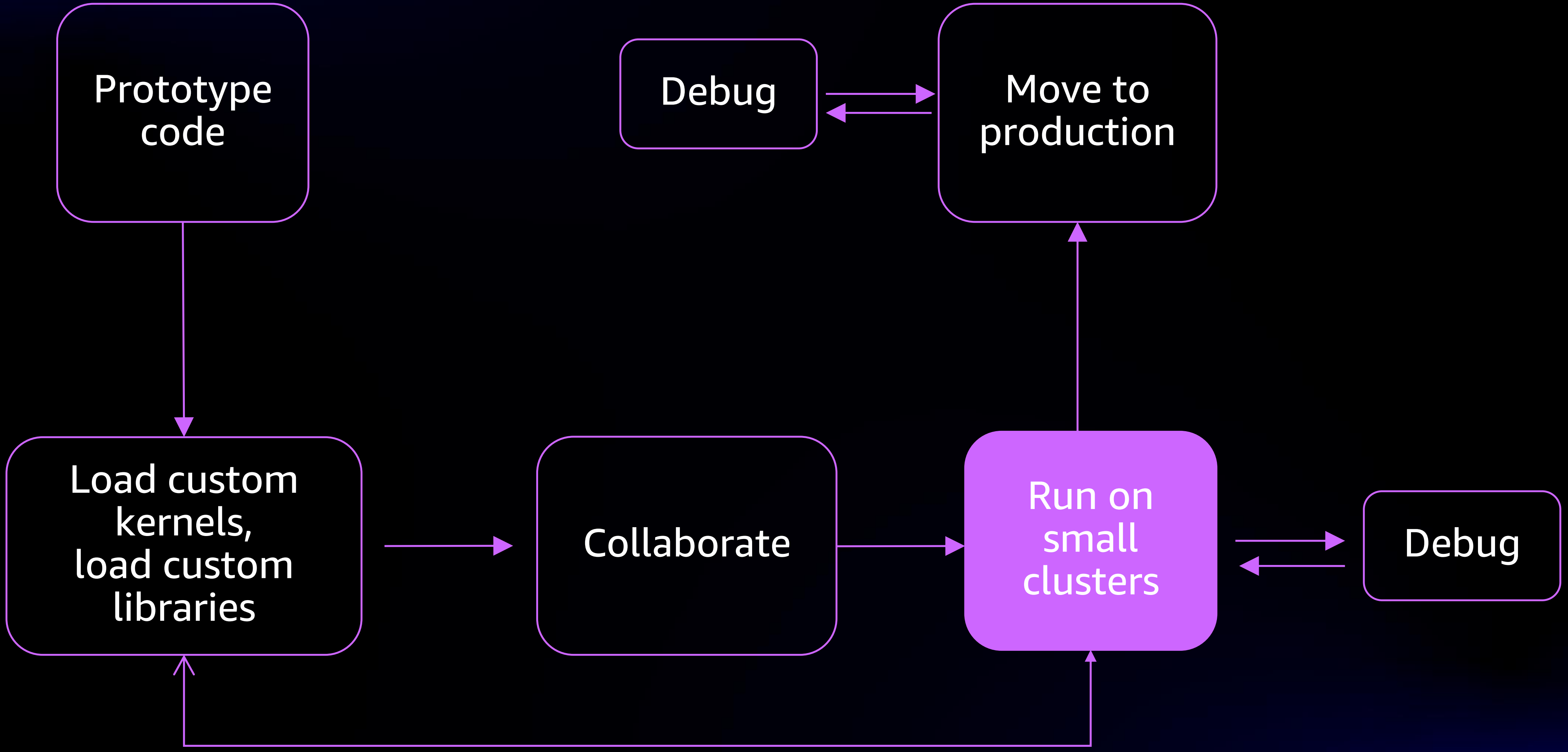


```
[ ]: import random

     NUM_SAMPLES = 100
```

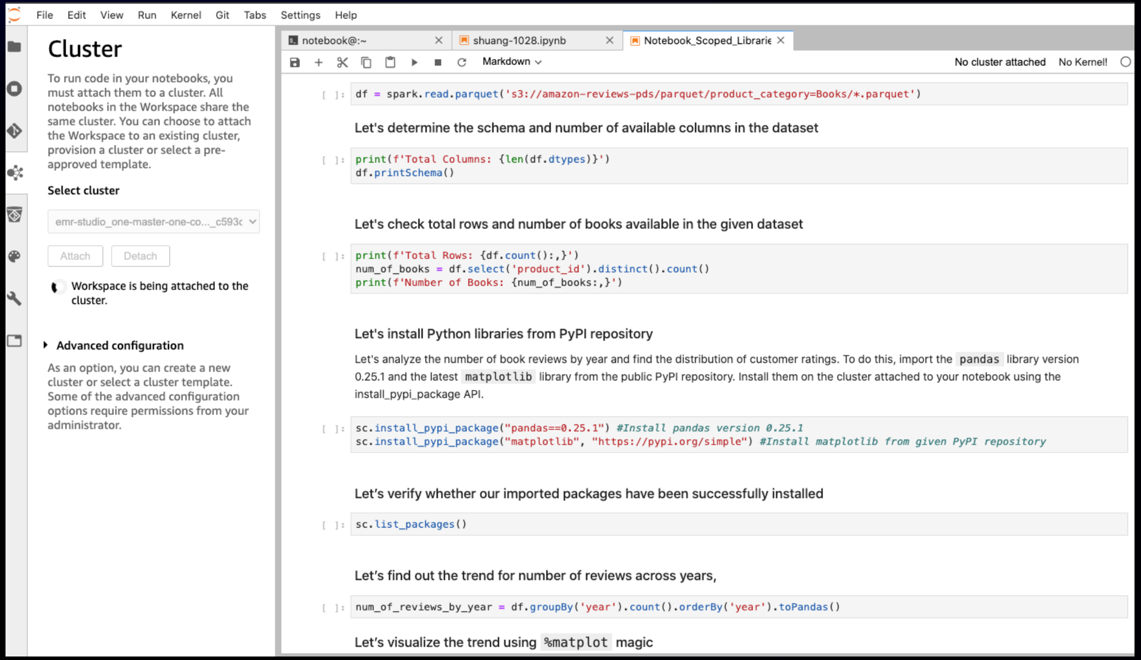
```
[ ]:
     def inside(p):
         x, y = random.random(), random.random()
         return x*x + y*y < 1

     count = sc.parallelize(range(0, NUM_SAMPLES)) \
                  .filter(inside).count()
     print("Pi is roughly %f" % (4.0 * count / NUM_SAMPLES))
```

Collaborate in real time

Data science and engineering workflows



Single-click attach to clusters to run jobs



Cluster

To run code in your notebooks, you must attach them to a cluster. All notebooks in the Workspace share the same cluster. You can choose to attach the Workspace to an existing cluster, provision a cluster or select a pre-approved template.

Select cluster

emr-studio_one-master-one-co..._c593c

Workspace is being attached to the cluster.

Advanced configuration

As an option, you can create a new cluster or select a cluster template. Some of the advanced configuration options require permissions from your administrator.

```
df = spark.read.parquet('s3://amazon-reviews-pds/parquet/product_category=Books/*.parquet')
```

Let's determine the schema and number of available columns in the dataset

```
print(f'Total Columns: {len(df.dtypes)}')
df.printSchema()
```

Let's check total rows and number of books available in the given dataset

```
print(f'Total Rows: {df.count():,}')
num_of_books = df.select('product_id').distinct().count()
print(f'Number of Books: {num_of_books:,}')
```

Let's install Python libraries from PyPI repository

Let's analyze the number of book reviews by year and find the distribution of customer ratings. To do this, import the `pandas` library version 0.25.1 and the latest `matplotlib` library from the public PyPI repository. Install them on the cluster attached to your notebook using the install_pypi_package API.

```
sc.install_pypi_package("pandas==0.25.1") #Install pandas version 0.25.1
sc.install_pypi_package("matplotlib", "https://pypi.org/simple") #Install matplotlib from given PyPI repository
```

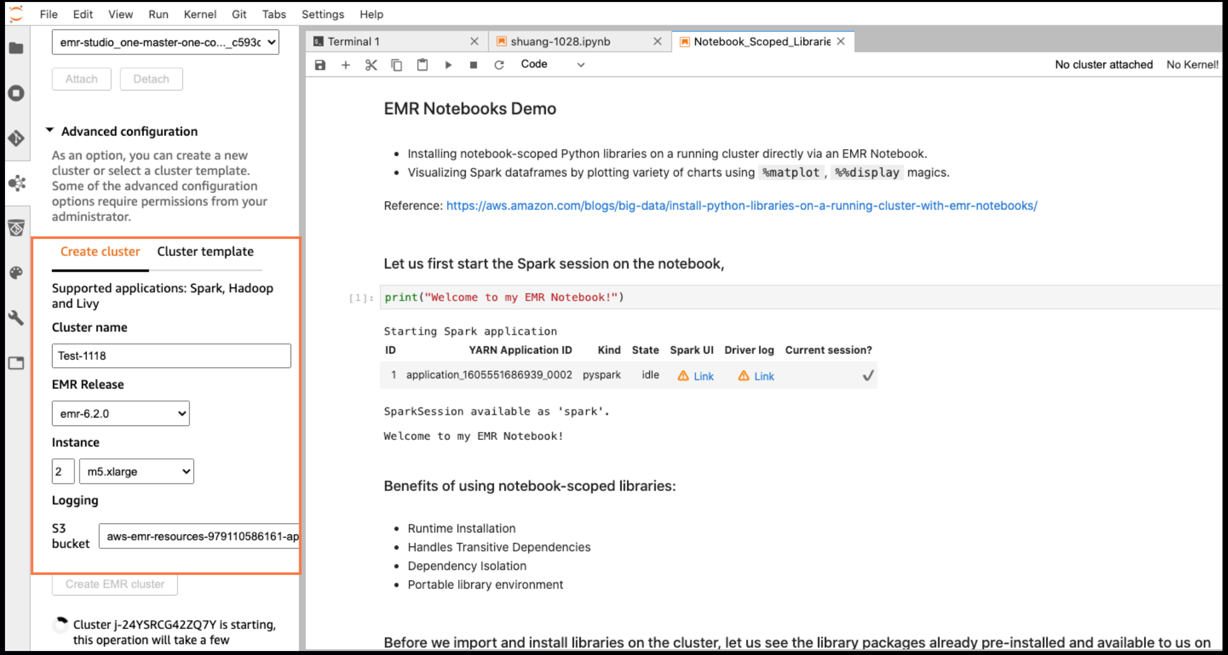
Let's verify whether our imported packages have been successfully installed

```
sc.list_packages()
```

Let's find out the trend for number of reviews across years,

```
num_of_reviews_by_year = df.groupBy('year').count().orderBy('year').toPandas()
```

Let's visualize the trend using `%matplot` magic



emr-studio_one-master-one-co..._c593c

Advanced configuration

As an option, you can create a new cluster or select a cluster template. Some of the advanced configuration options require permissions from your administrator.

Supported applications: Spark, Hadoop and Livy

Cluster name: Test-1118

EMR Release: emr-6.2.0

Instance: 2 m5.xlarge

Logging

S3 bucket: aws-emr-resources-979110586161-ap

Cluster j-24YSRCG42ZQ7Y is starting, this operation will take a few

EMR Notebooks Demo

- Installing notebook-scoped Python libraries on a running cluster directly via an EMR Notebook.
- Visualizing Spark dataframes by plotting variety of charts using `%matplot`, `%%display` magics.

Reference: https://aws.amazon.com/blogs/big-data/install-python-libraries-on-a-running-cluster-with-emr-notebooks/

Let us first start the Spark session on the notebook,

```
print("Welcome to my EMR Notebook!")
```

```
Starting Spark application
```

ID	YARN Application ID	Kind	State	Spark UI	Driver log	Current session?
1	application_1605551686939_0002	pyspark	idle	Link	Link	✓

```
SparkSession available as 'spark'.

Welcome to my EMR Notebook!
```

Benefits of using notebook-scoped libraries:

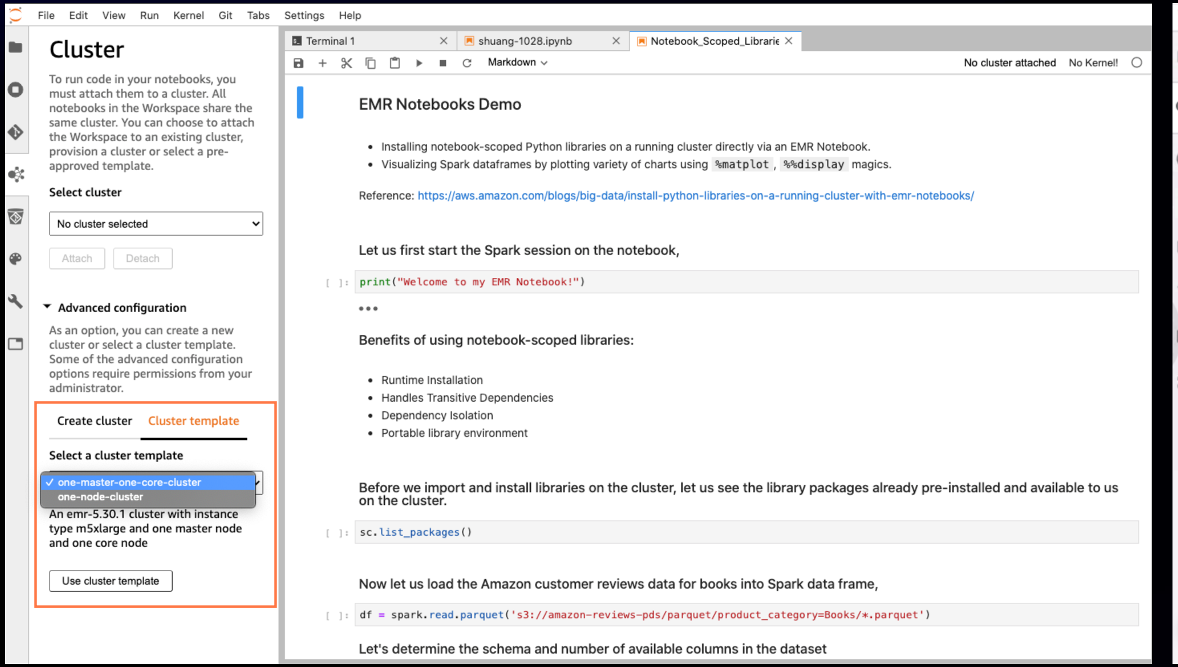
- Runtime Installation
- Handles Transitive Dependencies
- Dependency Isolation
- Portable library environment

Before we import and install libraries on the cluster, let us see the library packages already pre-installed and available to us on

Attach workspace to an existing EMR cluster

Provision EMR clusters using simple configurations (you can limit users to either cluster templates or creating their own EMR cluster)

Single-click attach to clusters to run jobs



Cluster

To run code in your notebooks, you must attach them to a cluster. All notebooks in the Workspace share the same cluster. You can choose to attach the Workspace to an existing cluster, provision a cluster or select a pre-approved template.

Select cluster

No cluster selected

Attach Detach

Advanced configuration

As an option, you can create a new cluster or select a cluster template. Some of the advanced configuration options require permissions from your administrator.

Create cluster | Cluster template

Select a cluster template

✓ one-master-one-core-cluster
one-node-cluster

An emr-5.30.1 cluster with instance type m5xlarge and one master node and one core node

Use cluster template

EMR Notebooks Demo

- Installing notebook-scoped Python libraries on a running cluster directly via an EMR Notebook.
- Visualizing Spark dataframes by plotting variety of charts using `%matplot`, `%%display` magics.

Reference: https://aws.amazon.com/blogs/big-data/install-python-libraries-on-a-running-cluster-with-emr-notebooks/

Let us first start the Spark session on the notebook,

```
[ ]: print("Welcome to my EMR Notebook!")
```

Benefits of using notebook-scoped libraries:

- Runtime Installation
- Handles Transitive Dependencies
- Dependency Isolation
- Portable library environment

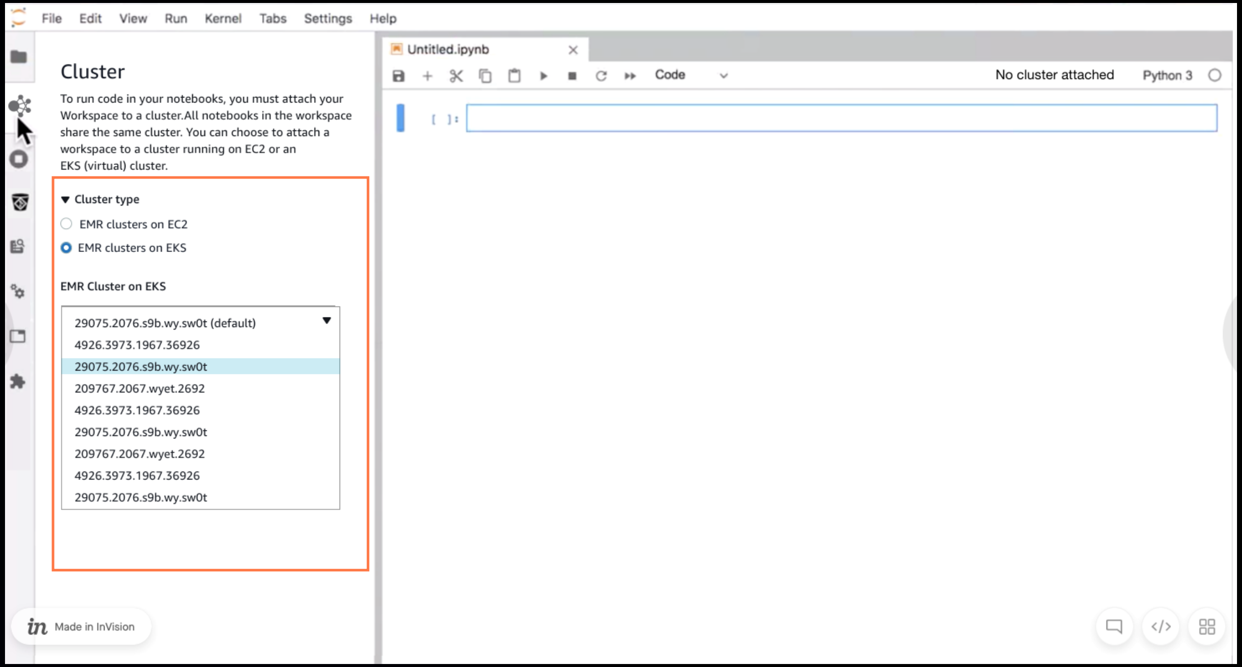
Before we import and install libraries on the cluster, let us see the library packages already pre-installed and available to us on the cluster.

```
[ ]: sc.list_packages()
```

Now let us load the Amazon customer reviews data for books into Spark data frame,

```
[ ]: df = spark.read.parquet('s3://amazon-reviews-pds/parquet/product_category=Books/*.parquet')
```

Let's determine the schema and number of available columns in the dataset



Cluster

To run code in your notebooks, you must attach your Workspace to a cluster.All notebooks in the workspace share the same cluster. You can choose to attach a workspace to a cluster running on EC2 or an EKS (virtual) cluster.

Cluster type

( ) EMR clusters on EC2
(●) EMR clusters on EKS

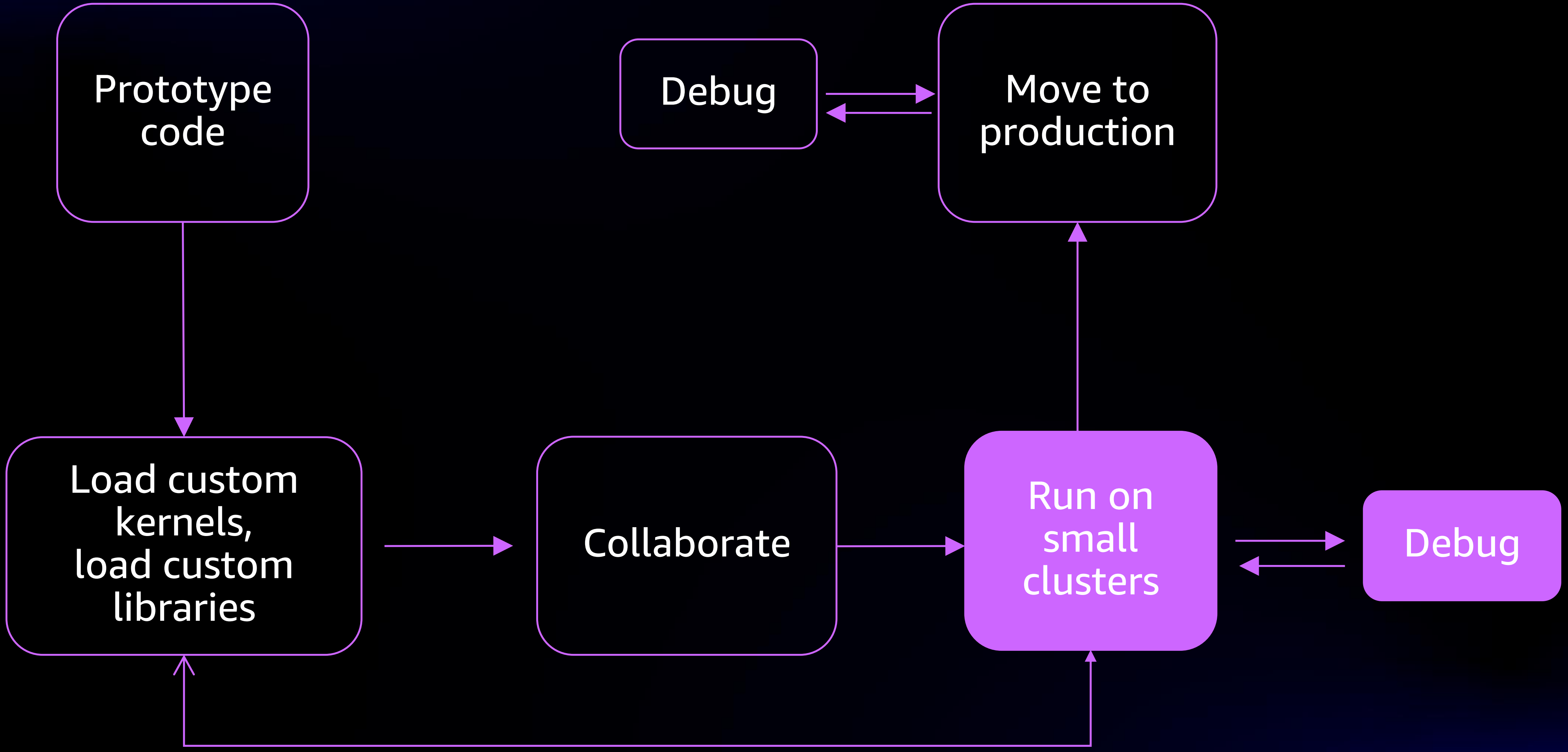
EMR Cluster on EKS

29075.2076.s9b.wy.sw0t (default)
4926.3973.1967.36926
29075.2076.s9b.wy.sw0t
209767.2067.wyet.2692
4926.3973.1967.36926
29075.2076.s9b.wy.sw0t
209767.2067.wyet.2692
4926.3973.1967.36926
29075.2076.s9b.wy.sw0t

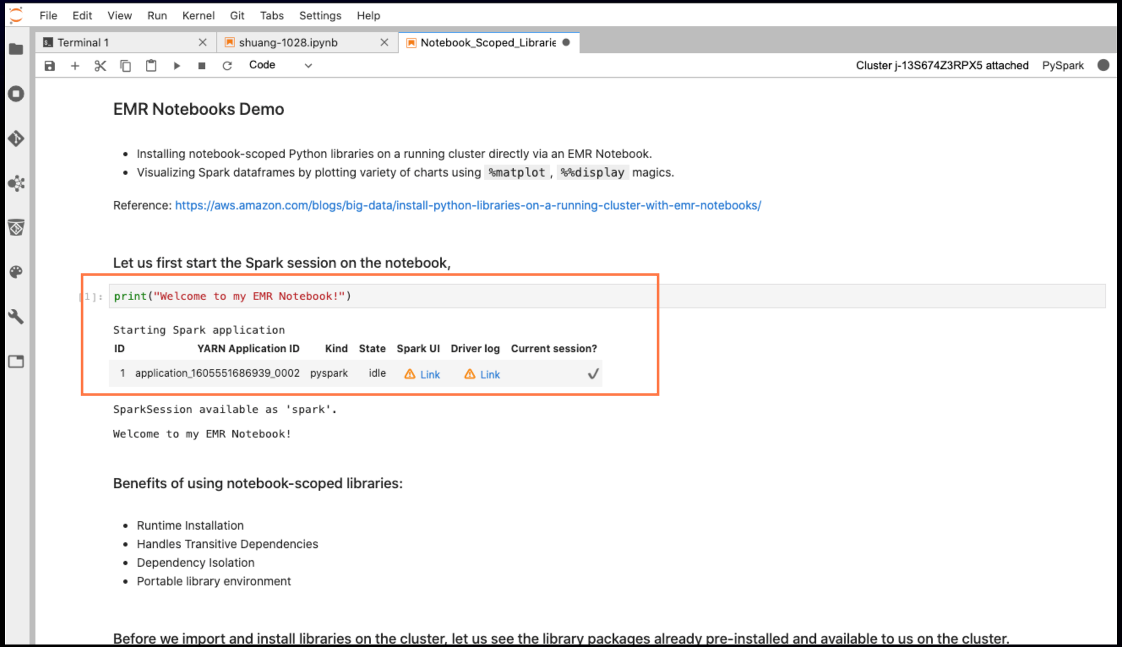
Provision EMR clusters using preconfigured cluster templates via AWS Service Catalog

Connecting to clusters from Amazon EKS

Data science and engineering workflows



Live debugging is simple


File Edit View Run Kernel Git Tabs Settings Help
Terminal 1
shuang-1028.ipynb
Notebook_Scoped_Librarie
Code
Cluster j-13S674Z3RPX5 attached
PySpark

EMR Notebooks Demo

- Installing notebook-scoped Python libraries on a running cluster directly via an EMR Notebook.
- Visualizing Spark dataframes by plotting variety of charts using `%matplot`, `%%display` magics.

Reference: https://aws.amazon.com/blogs/big-data/install-python-libraries-on-a-running-cluster-with-emr-notebooks/

Let us first start the Spark session on the notebook,

```
[1]: print("Welcome to my EMR Notebook!")
```

```
Starting Spark application
```

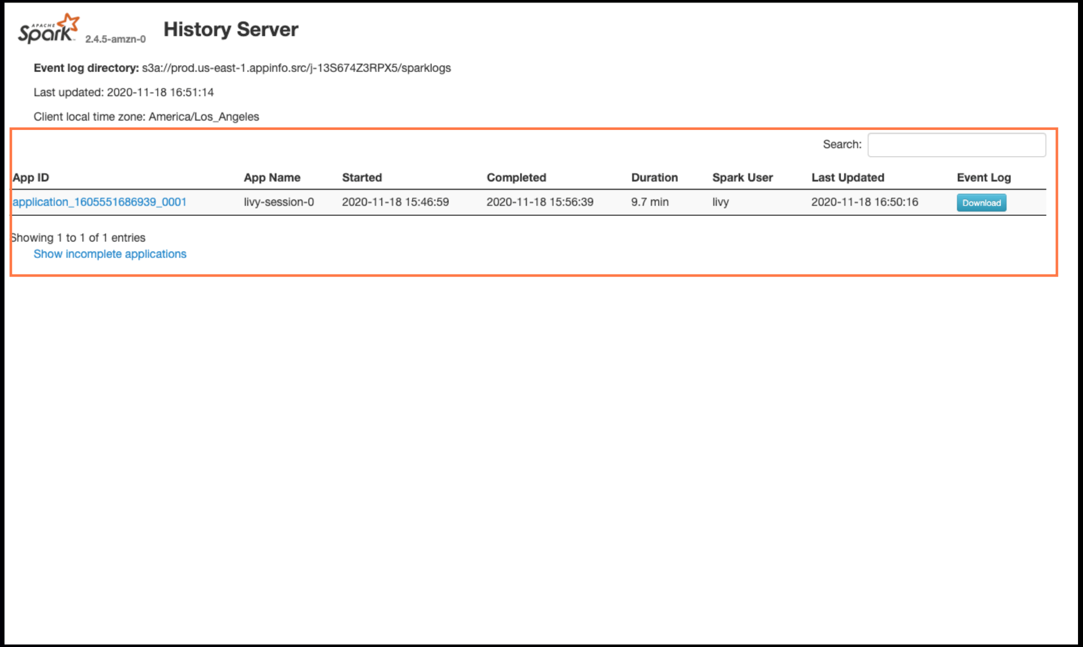
ID	YARN Application ID	Kind	State	Spark UI	Driver log	Current session?
1	application_1605551686939_0002	pyspark	idle	Link	Link	✓

```
SparkSession available as 'spark'.
Welcome to my EMR Notebook!
```

Benefits of using notebook-scoped libraries:

- Runtime Installation
- Handles Transitive Dependencies
- Dependency Isolation
- Portable library environment

Before we import and install libraries on the cluster, let us see the library packages already pre-installed and available to us on the cluster.



History Server

Event log directory: s3a://prod.us-east-1.appinfo.src/j-13S674Z3RPX5/sparklogs

Last updated: 2020-11-18 16:51:14

Client local time zone: America/Los_Angeles

Search:

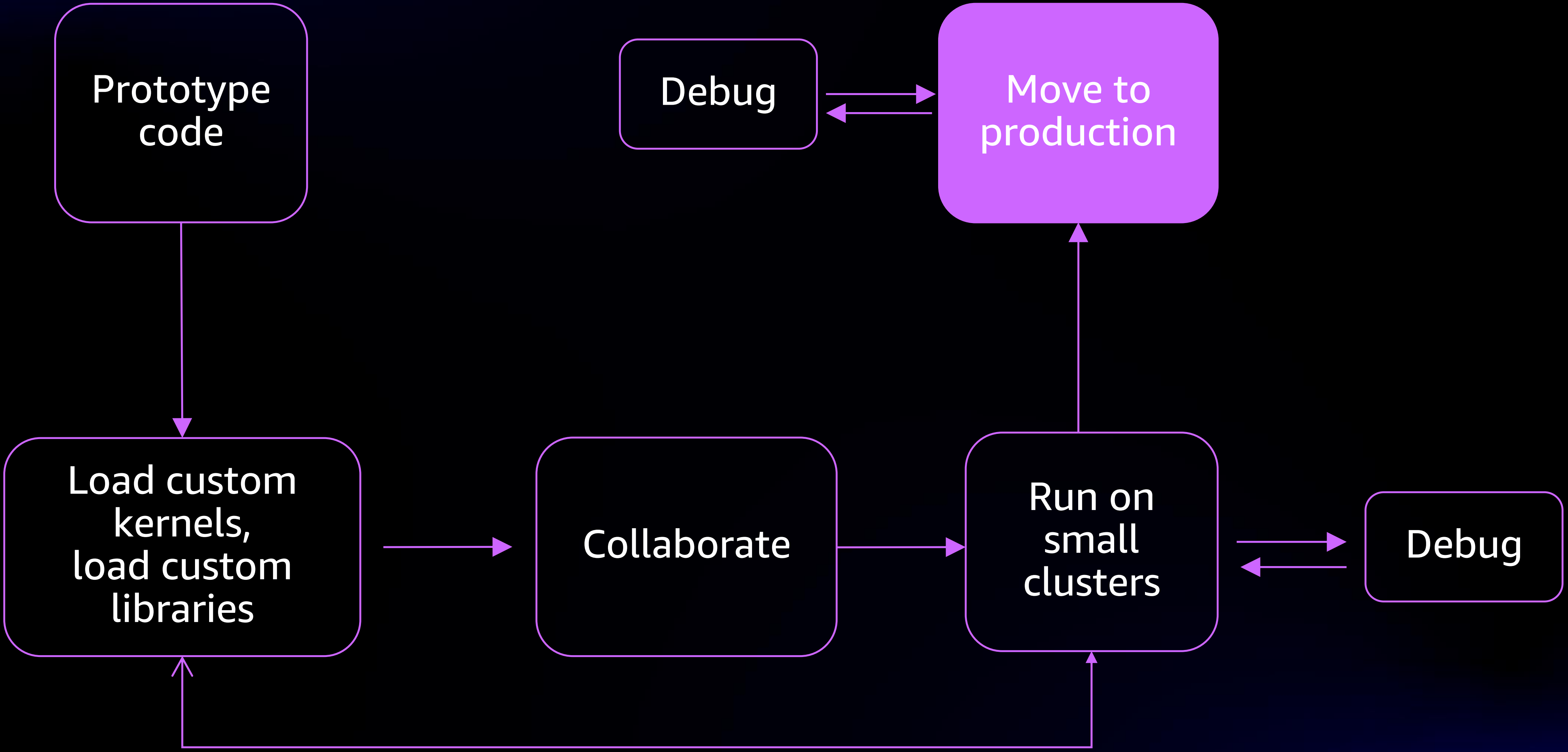
App ID	App Name	Started	Completed	Duration	Spark User	Last Updated	Event Log
application_1605551686939_0001	livy-session-0	2020-11-18 15:46:59	2020-11-18 15:56:39	9.7 min	livy	2020-11-18 16:50:16	Download

Showing 1 to 1 of 1 entries
Show incomplete applications

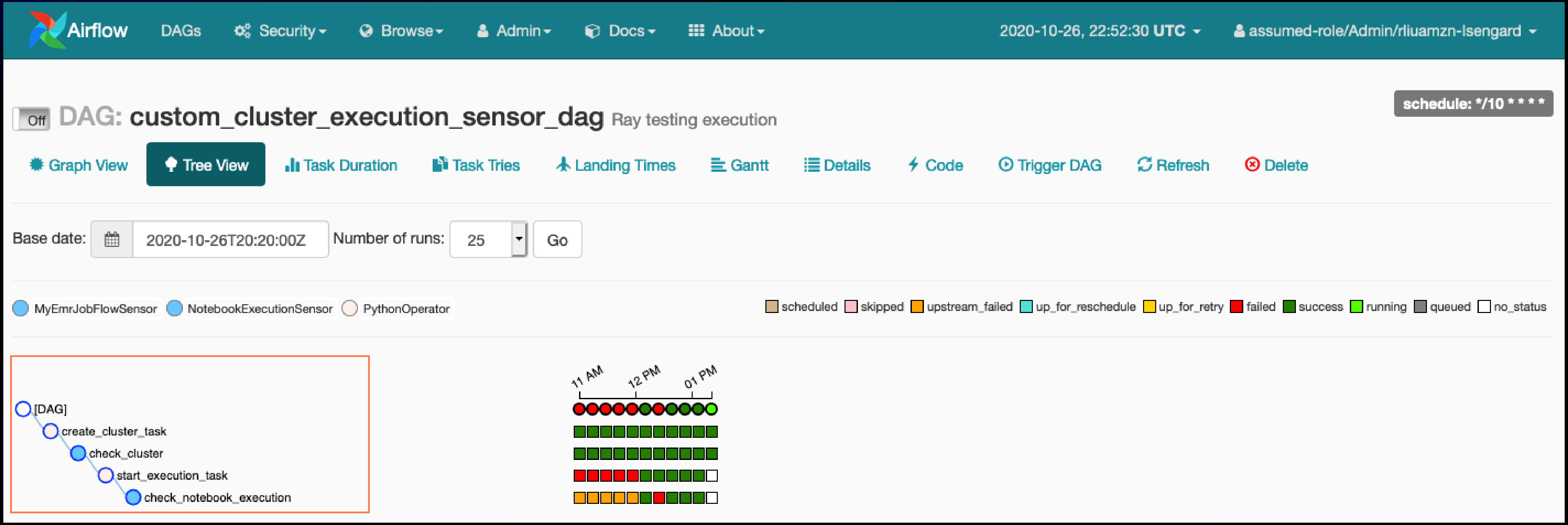
Debug by clicking the "Spark UI" link in notebook to navigate to the live on-cluster Spark UI

View debugging information in Spark history server for the application in a separate browser tab

Data science and engineering workflows



Simplify building pipelines from notebooks



2020-10-26, 22:52:30 UTC

assumed-role/Admin/rliuamzn-Isengard

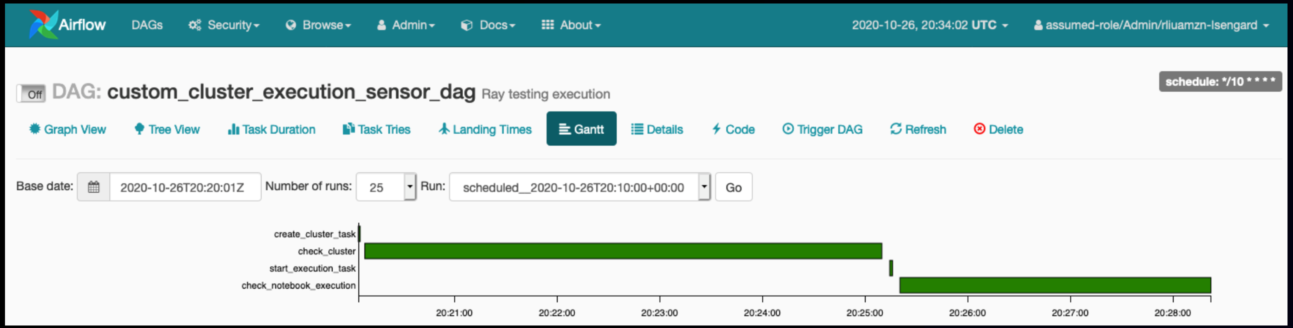
DAG: custom_cluster_execution_sensor_dag Ray testing execution

schedule: */10 * * * *

Base date: 2020-10-26T20:20:00Z Number of runs: 25

MyEmrJobFlowSensor NotebookExecutionSensor PythonOperator

[DAG]
create_cluster_task
check_cluster
start_execution_task
check_notebook_execution



2020-10-26, 20:34:02 UTC

assumed-role/Admin/rliuamzn-Isengard

DAG: custom_cluster_execution_sensor_dag Ray testing execution

schedule: */10 * * * *

Base date: 2020-10-26T20:20:01Z Number of runs: 25 Run: scheduled__2020-10-26T20:10:00+00:00

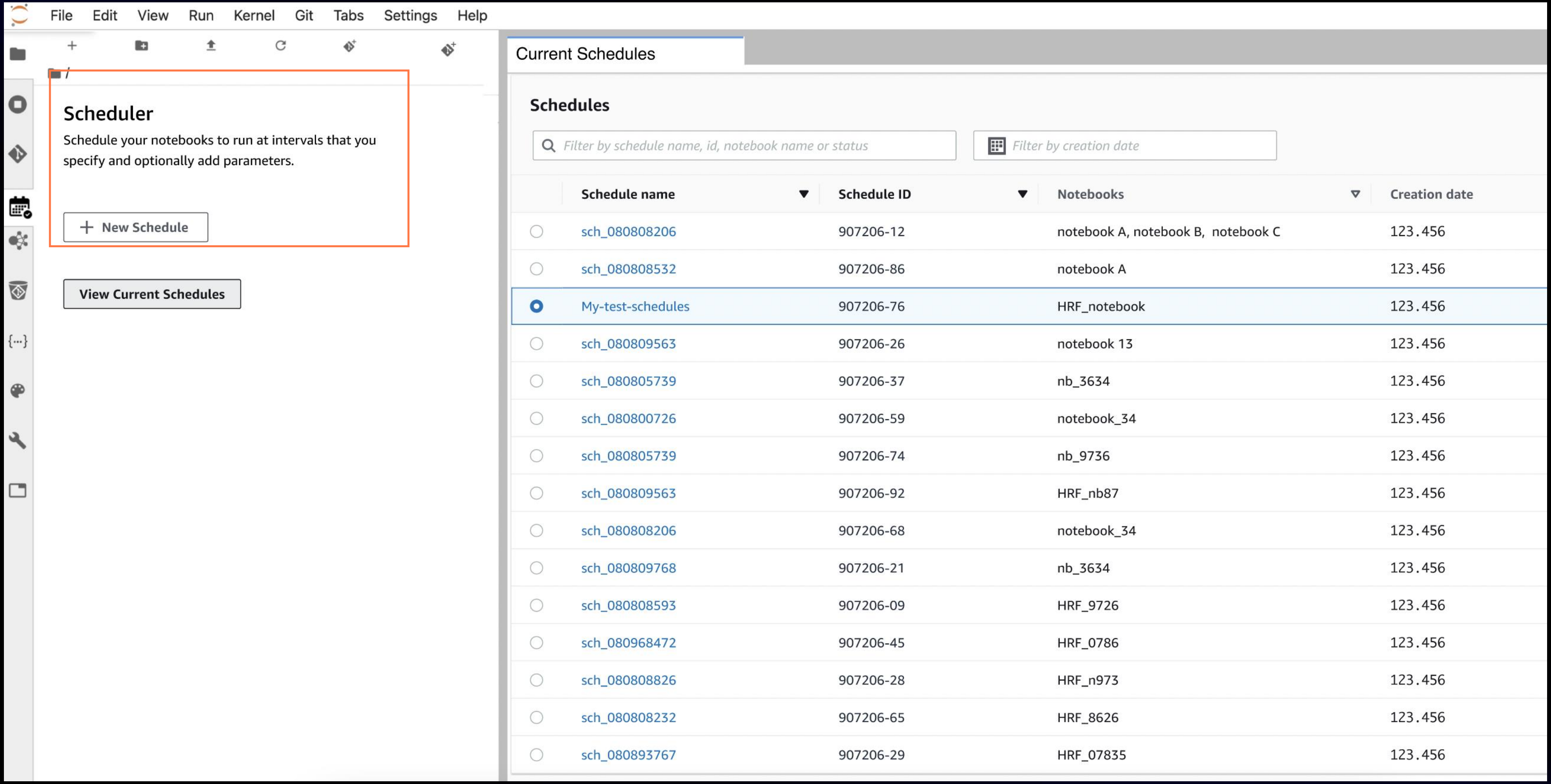
create_cluster_task
check_cluster
start_execution_task
check_notebook_execution

Run notebooks as pipelines via Amazon Managed Workflows for Apache Airflow (MWAA)

Parameterize and chain notebooks that can be run as pipelines

Schedule notebook pipelines

Coming soon



Scheduler

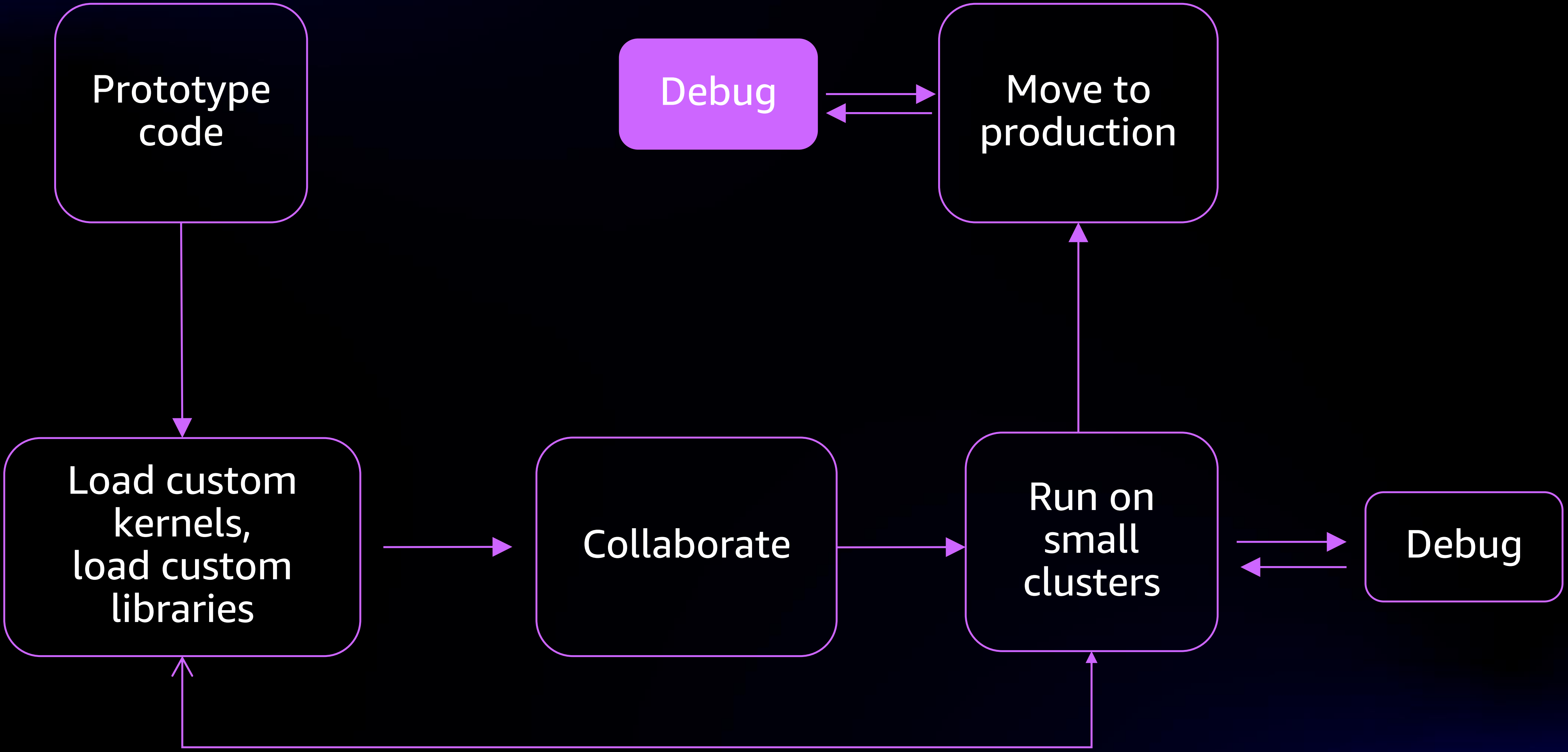
Schedule your notebooks to run at intervals that you specify and optionally add parameters.

Schedules

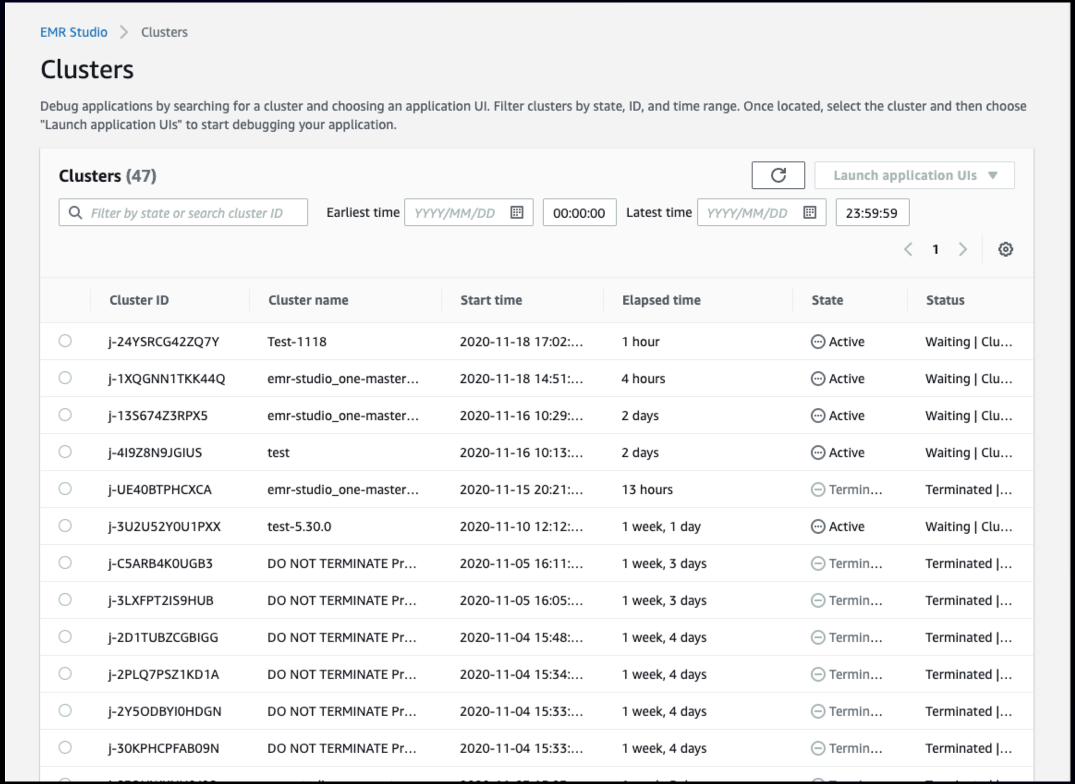
	Schedule name	Schedule ID	Notebooks	Creation date
○	sch_080808206	907206-12	notebook A, notebook B, notebook C	123.456
○	sch_080808532	907206-86	notebook A	123.456
●	My-test-schedules	907206-76	HRF_notebook	123.456
○	sch_080809563	907206-26	notebook 13	123.456
○	sch_080805739	907206-37	nb_3634	123.456
○	sch_080800726	907206-59	notebook_34	123.456
○	sch_080805739	907206-74	nb_9736	123.456
○	sch_080809563	907206-92	HRF_nb87	123.456
○	sch_080808206	907206-68	notebook_34	123.456
○	sch_080809768	907206-21	nb_3634	123.456
○	sch_080808593	907206-09	HRF_9726	123.456
○	sch_080968472	907206-45	HRF_0786	123.456
○	sch_080808826	907206-28	HRF_n973	123.456
○	sch_080808232	907206-65	HRF_8626	123.456
○	sch_080893767	907206-29	HRF_07835	123.456

Schedule notebooks from within Jupyter

Data science and engineering workflows



Debugging production pipelines is easy



EMR Studio > Clusters

Clusters

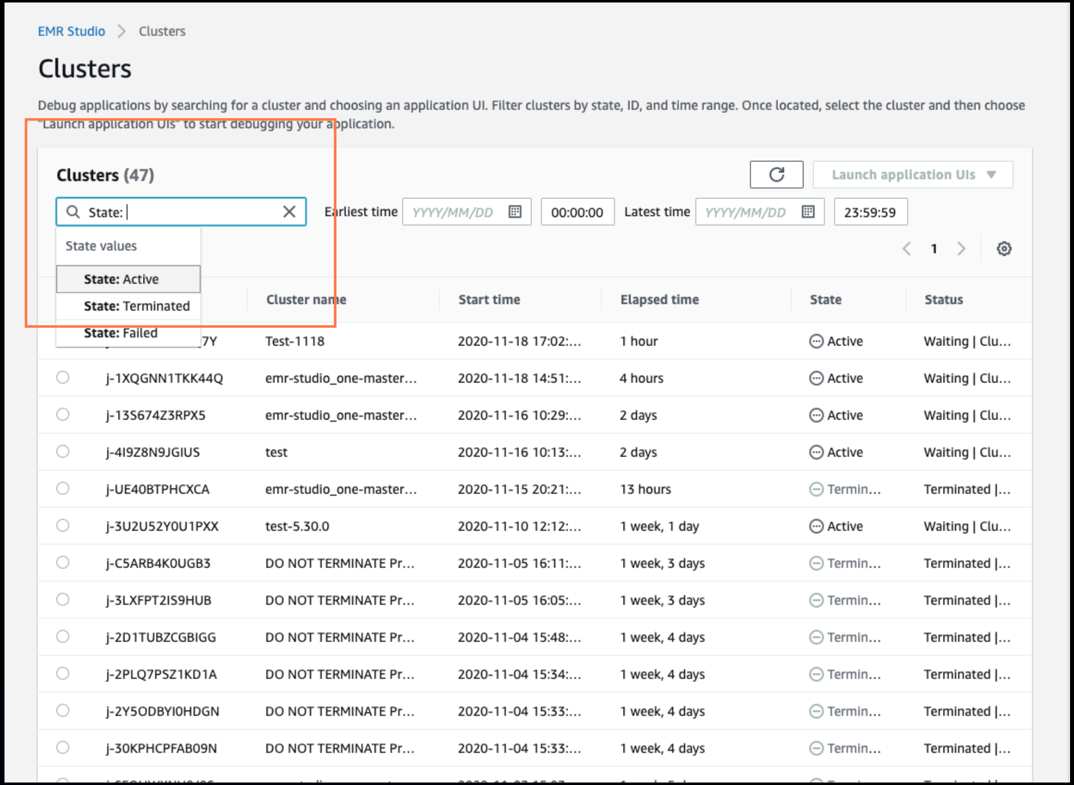
Debug applications by searching for a cluster and choosing an application UI. Filter clusters by state, ID, and time range. Once located, select the cluster and then choose "Launch application UIs" to start debugging your application.

Clusters (47)

Filter by state or search cluster ID | Earliest time YYYY/MM/DD 00:00:00 | Latest time YYYY/MM/DD 23:59:59 | Launch application UIs

Cluster ID	Cluster name	Start time	Elapsed time	State	Status
j-24YSRCG42ZQ7Y	Test-1118	2020-11-18 17:02:...	1 hour	Active	Waiting | Clu...
j-1XQGNN1TKK44Q	emr-studio_one-master...	2020-11-18 14:51:...	4 hours	Active	Waiting | Clu...
j-13S674Z3RPX5	emr-studio_one-master...	2020-11-16 10:29:...	2 days	Active	Waiting | Clu...
j-4I9Z8N9JGIUS	test	2020-11-16 10:13:...	2 days	Active	Waiting | Clu...
j-UE40BTPHCXCA	emr-studio_one-master...	2020-11-15 20:21:...	13 hours	Termin...	Terminated |...
j-3U2U52Y0U1PXX	test-5.30.0	2020-11-10 12:12:...	1 week, 1 day	Active	Waiting | Clu...
j-C5ARB4K0UGB3	DO NOT TERMINATE Pr...	2020-11-05 16:11:...	1 week, 3 days	Termin...	Terminated |...
j-3LXFPT2IS9HUB	DO NOT TERMINATE Pr...	2020-11-05 16:05:...	1 week, 3 days	Termin...	Terminated |...
j-2D1TUBZCGBIGG	DO NOT TERMINATE Pr...	2020-11-04 15:48:...	1 week, 4 days	Termin...	Terminated |...
j-2PLQ7PSZ1KD1A	DO NOT TERMINATE Pr...	2020-11-04 15:34:...	1 week, 4 days	Termin...	Terminated |...
j-2Y5ODBYI0HDGN	DO NOT TERMINATE Pr...	2020-11-04 15:33:...	1 week, 4 days	Termin...	Terminated |...
j-30KPHCPFAB09N	DO NOT TERMINATE Pr...	2020-11-04 15:33:...	1 week, 4 days	Termin...	Terminated |...

Browse all clusters in one place



EMR Studio > Clusters

Clusters

Debug applications by searching for a cluster and choosing an application UI. Filter clusters by state, ID, and time range. Once located, select the cluster and then choose "Launch application UIs" to start debugging your application.

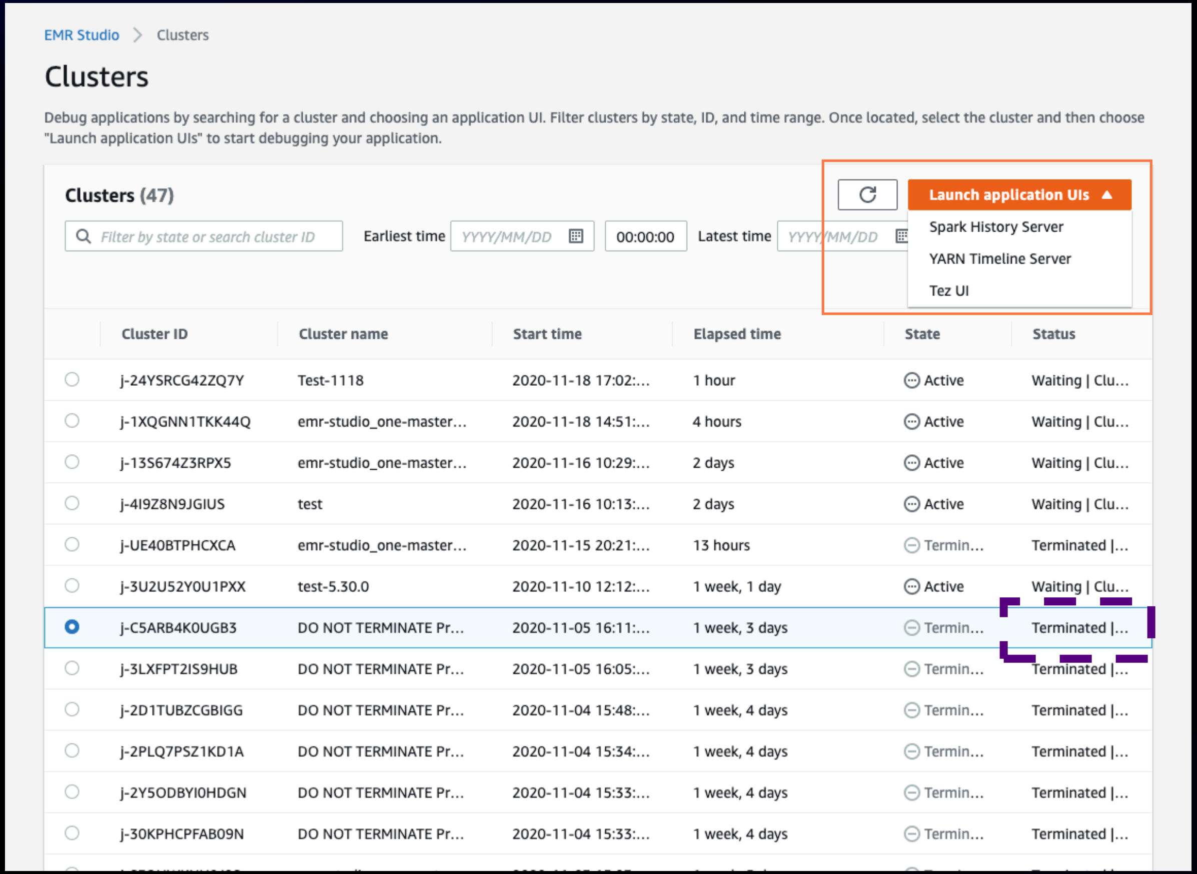
Clusters (47)

State: | Earliest time YYYY/MM/DD 00:00:00 | Latest time YYYY/MM/DD 23:59:59 | Launch application UIs

Cluster ID	Cluster name	Start time	Elapsed time	State	Status
7Y	Test-1118	2020-11-18 17:02:...	1 hour	Active	Waiting | Clu...
j-1XQGNN1TKK44Q	emr-studio_one-master...	2020-11-18 14:51:...	4 hours	Active	Waiting | Clu...
j-13S674Z3RPX5	emr-studio_one-master...	2020-11-16 10:29:...	2 days	Active	Waiting | Clu...
j-4I9Z8N9JGIUS	test	2020-11-16 10:13:...	2 days	Active	Waiting | Clu...
j-UE40BTPHCXCA	emr-studio_one-master...	2020-11-15 20:21:...	13 hours	Termin...	Terminated |...
j-3U2U52Y0U1PXX	test-5.30.0	2020-11-10 12:12:...	1 week, 1 day	Active	Waiting | Clu...
j-C5ARB4K0UGB3	DO NOT TERMINATE Pr...	2020-11-05 16:11:...	1 week, 3 days	Termin...	Terminated |...
j-3LXFPT2IS9HUB	DO NOT TERMINATE Pr...	2020-11-05 16:05:...	1 week, 3 days	Termin...	Terminated |...
j-2D1TUBZCGBIGG	DO NOT TERMINATE Pr...	2020-11-04 15:48:...	1 week, 4 days	Termin...	Terminated |...
j-2PLQ7PSZ1KD1A	DO NOT TERMINATE Pr...	2020-11-04 15:34:...	1 week, 4 days	Termin...	Terminated |...
j-2Y5ODBYI0HDGN	DO NOT TERMINATE Pr...	2020-11-04 15:33:...	1 week, 4 days	Termin...	Terminated |...
j-30KPHCPFAB09N	DO NOT TERMINATE Pr...	2020-11-04 15:33:...	1 week, 4 days	Termin...	Terminated |...

Narrow down clusters for investigation using filters such as cluster state

Debugging production pipelines is easy



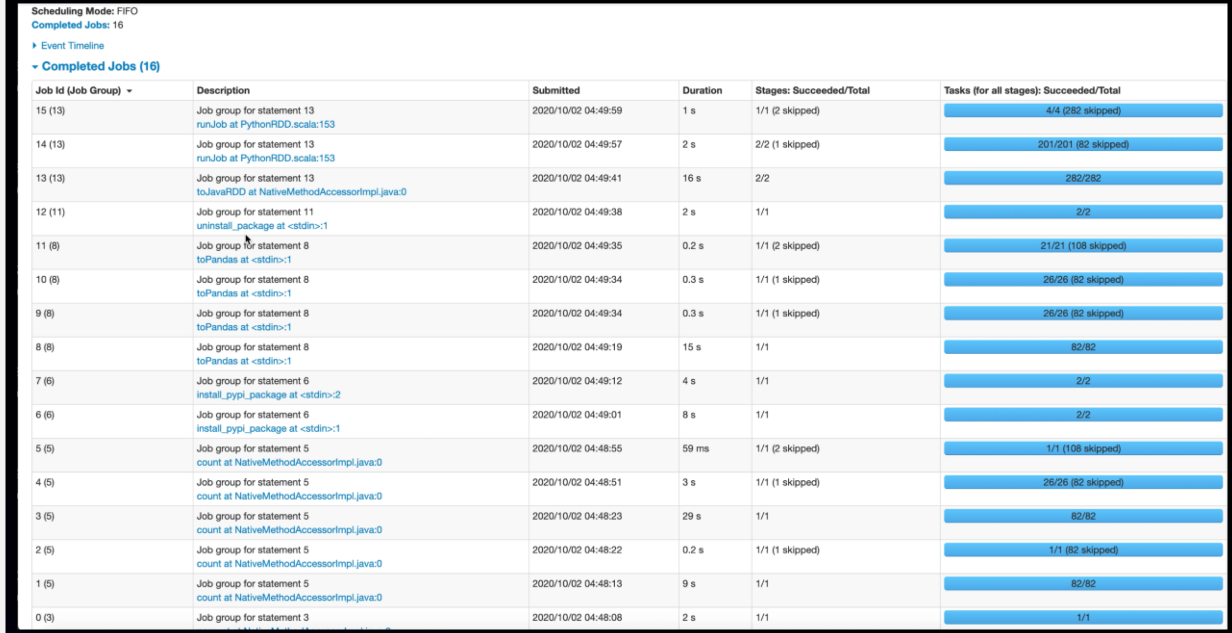
EMR Studio > Clusters

Clusters

Debug applications by searching for a cluster and choosing an application UI. Filter clusters by state, ID, and time range. Once located, select the cluster and then choose "Launch application UIs" to start debugging your application.

Clusters (47)

	Cluster ID	Cluster name	Start time	Elapsed time	State	Status
○	j-24YSRCG42ZQ7Y	Test-1118	2020-11-18 17:02:...	1 hour	Active	Waiting | Clu...
○	j-1XQGNN1TKK44Q	emr-studio_one-master...	2020-11-18 14:51:...	4 hours	Active	Waiting | Clu...
○	j-13S674Z3RPX5	emr-studio_one-master...	2020-11-16 10:29:...	2 days	Active	Waiting | Clu...
○	j-4I9Z8N9JGIUS	test	2020-11-16 10:13:...	2 days	Active	Waiting | Clu...
○	j-UE40BTPHCXCA	emr-studio_one-master...	2020-11-15 20:21:...	13 hours	Termin...	Terminated |...
○	j-3U2U52Y0U1PXX	test-5.30.0	2020-11-10 12:12:...	1 week, 1 day	Active	Waiting | Clu...
●	j-C5ARB4K0UGB3	DO NOT TERMINATE Pr...	2020-11-05 16:11:...	1 week, 3 days	Termin...	Terminated |...
○	j-3LXFPT2IS9HUB	DO NOT TERMINATE Pr...	2020-11-05 16:05:...	1 week, 3 days	Termin...	Terminated |...
○	j-2D1TUBZCGBIGG	DO NOT TERMINATE Pr...	2020-11-04 15:48:...	1 week, 4 days	Termin...	Terminated |...
○	j-2PLQ7PSZ1KD1A	DO NOT TERMINATE Pr...	2020-11-04 15:34:...	1 week, 4 days	Termin...	Terminated |...
○	j-2Y5ODBYI0HDGN	DO NOT TERMINATE Pr...	2020-11-04 15:33:...	1 week, 4 days	Termin...	Terminated |...
○	j-30KPHCPFAB09N	DO NOT TERMINATE Pr...	2020-11-04 15:33:...	1 week, 4 days	Termin...	Terminated |...



Scheduling Mode: FIFO
Completed Jobs: 16

Completed Jobs (16)

Job Id (Job Group)	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
15 (13)	Job group for statement 13 runJob at PythonRDD.scala:153	2020/10/02 04:49:59	1 s	1/1 (2 skipped)	4/4 (282 skipped)
14 (13)	Job group for statement 13 runJob at PythonRDD.scala:153	2020/10/02 04:49:57	2 s	2/2 (1 skipped)	201/201 (82 skipped)
13 (13)	Job group for statement 13 toJavaRDD at NativeMethodAccessorImpl.java:0	2020/10/02 04:49:41	16 s	2/2	282/282
12 (11)	Job group for statement 11 uninstall_package at <stdin>:1	2020/10/02 04:49:38	2 s	1/1	2/2
11 (8)	Job group for statement 8 toPandas at <stdin>:1	2020/10/02 04:49:35	0.2 s	1/1 (2 skipped)	21/21 (108 skipped)
10 (8)	Job group for statement 8 toPandas at <stdin>:1	2020/10/02 04:49:34	0.3 s	1/1 (1 skipped)	26/26 (82 skipped)
9 (8)	Job group for statement 8 toPandas at <stdin>:1	2020/10/02 04:49:34	0.3 s	1/1 (1 skipped)	26/26 (82 skipped)
8 (8)	Job group for statement 8 toPandas at <stdin>:1	2020/10/02 04:49:19	15 s	1/1	82/82
7 (6)	Job group for statement 6 install_pypi_package at <stdin>:2	2020/10/02 04:49:12	4 s	1/1	2/2
6 (6)	Job group for statement 6 install_pypi_package at <stdin>:1	2020/10/02 04:49:01	8 s	1/1	2/2
5 (5)	Job group for statement 5 count at NativeMethodAccessorImpl.java:0	2020/10/02 04:48:55	59 ms	1/1 (2 skipped)	1/1 (108 skipped)
4 (5)	Job group for statement 5 count at NativeMethodAccessorImpl.java:0	2020/10/02 04:48:51	3 s	1/1 (1 skipped)	26/26 (82 skipped)
3 (5)	Job group for statement 5 count at NativeMethodAccessorImpl.java:0	2020/10/02 04:48:23	29 s	1/1	82/82
2 (5)	Job group for statement 5 count at NativeMethodAccessorImpl.java:0	2020/10/02 04:48:22	0.2 s	1/1 (1 skipped)	1/1 (82 skipped)
1 (5)	Job group for statement 5 count at NativeMethodAccessorImpl.java:0	2020/10/02 04:48:13	9 s	1/1	82/82
0 (3)	Job group for statement 3	2020/10/02 04:48:08	2 s	1/1	1/1

Diagnose jobs on both active and terminated clusters using Spark UI, Tez UI, and Yarn Timeline Service

Overlay execution context on jobs, even for terminated clusters and jobs

Verana Health

“Choosing EMR Studio as our official workflow for Jupyter Notebooks on EMR has enabled us to reduce costs and time spent supporting data users. The built-in Git-based workflow has streamlined our previously cluttered landscape of notebooks. Connecting to an EMR cluster is as simple as selecting it in a dropdown box, avoiding the need to have personal clusters running 24/7.”

Phil Austin,
Director of DevOps
Verana Health

“EMR Studio allows us to prototype Spark applications and data science models that power large-scale data processing and transformations. The integrated development environment makes it easy for data scientists and engineers to perform ad hoc analysis and debug data processing workloads.”

Saba El-Hilo
Head of Data Platform, Mapbox

Deep integration between EMR and SageMaker Studio

GET ALL THE BENEFITS OF EMR FROM SAGEMAKER STUDIO

Process petabyte-scale data easily to train ML models using EMR Spark, Hive, and Presto from SageMaker Studio

Use EMR's integration with EC2 Spot and Graviton instances to run large-scale data processing at lower costs

Discover and connect to an EMR cluster from SageMaker Studio

Run Apache Spark, Hive, and Presto jobs on EMR from SageMaker Studio

Use familiar debugging tools such as Spark UI

Create, scale, and auto-terminate EMR clusters using AWS Service Catalog templates

Run Spark workloads on Amazon EKS easily

Amazon EMR on Amazon EKS

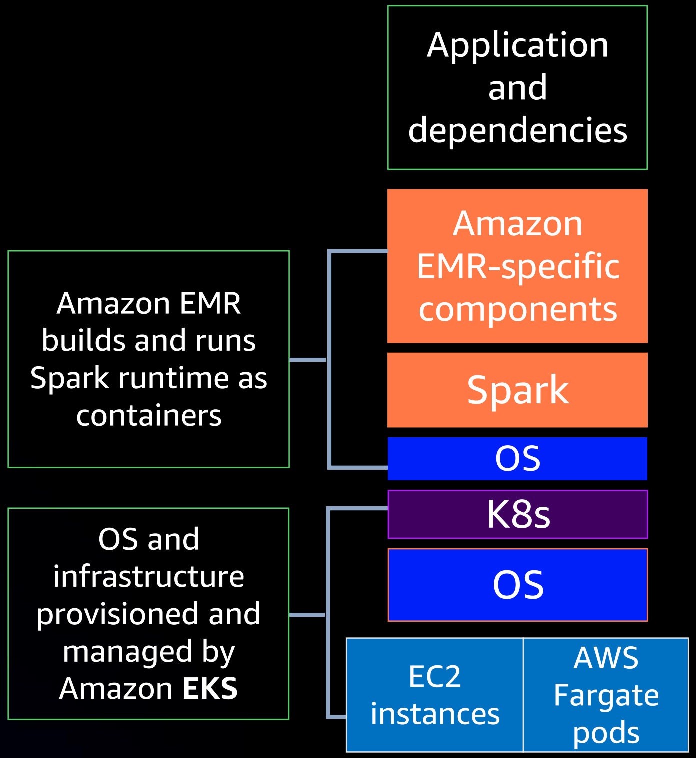
CONSOLIDATE ANALYTICS WORKLOADS WITH OTHER WORKLOADS ON EKS

Simplify infrastructure management

Consolidate multiple versions of Spark on same EKS cluster

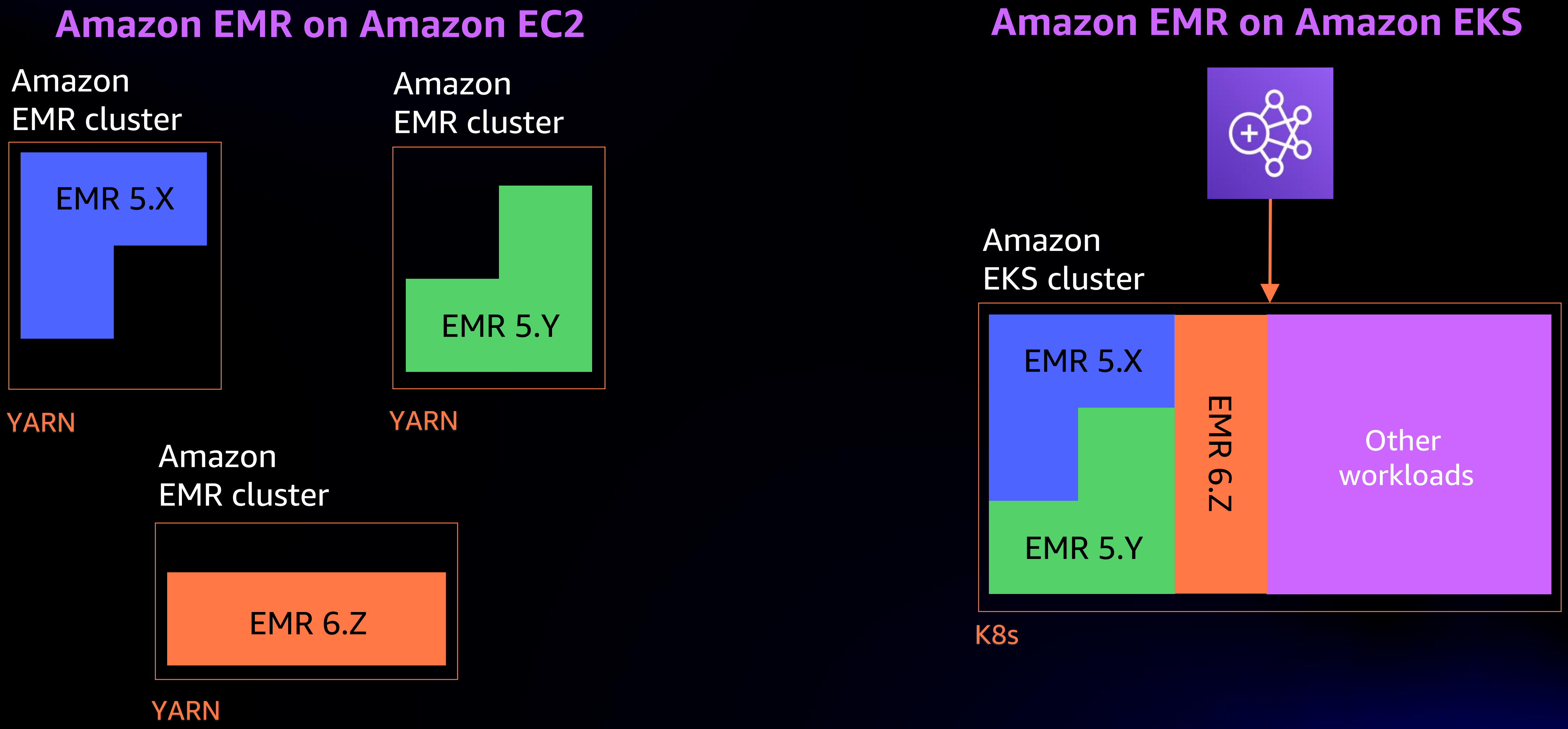
Simplify Spark application upgrades

Add Multi-AZ resiliency by EKS with worker nodes across multiple AZs


Application and dependencies
Amazon EMR builds and runs Spark runtime as containers
Amazon EMR-specific components
Spark
OS
K8s
OS and infrastructure provisioned and managed by Amazon EKS
OS
EC2 instances
AWS Fargate pods

Consolidate workloads with EMR on EKS

WORKLOAD CONSOLIDATION DRIVES HIGHER RESOURCE UTILIZATION AND LOWER COSTS



Running jobs on EMR on EKS is easy

NO NEED TO LEARN DIFFERENT TOOLS TO RUN SPARK JOBS USING EMR ON EKS

AWS CLI/SDK

EMR Studio, self-managed notebooks

Apache Airflow

AWS Step Functions

```
aws emr-containers start-job-run \
    --virtual-cluster-id cluster_id \
    --name sample-job-name \
    --execution-role-arn execution-role-arn \
    --release-label emr-6.3.0-latest \
    --job-driver '{
        "sparkSubmitJobDriver": {
            "entryPoint": "local:///usr/lib/spark/examples/src/main/python/pi.py",
            "sparkSubmitParameters": "--conf spark.executor.instances=2 --conf spark.executor.memory=2G --conf 
spark.executor.cores=2 --conf spark.driver.cores=1"
        }
    }'
```

Custom container images

MANAGING APPLICATION DEPENDENCIES IS SIMPLE ON EMR ON EKS

- Install and configure packages specific to different workloads in different container images
- Use custom image validation tool to detect errors when creating custom container image

Create a custom container image

Upload image to your registry

Use image in multiple EMR on EKS jobs with near instantaneous job start time

Pod templates

SPECIFYING HOW TO RUN A SPARK DRIVER OR EXECUTOR POD IS SIMPLE

Optimize price-performance by scheduling Spark executors to run on Amazon EC2 Spot or Graviton instances

Run a separate “sidecar” container next to the Spark driver or executor for logging or additional monitoring purposes

Run an “init” container that prepares the environment, e.g., downloads and installs dependencies

STITCH FIX

"Migration to EMR on EKS from open-source Spark on Kubernetes helped us to consolidate on two fronts – multiple Spark versions on same EKS cluster and Spark workloads alongside other workloads on same EKS cluster. This consolidation led to significant cost savings and reduced operational overhead."

Ujjwal Sarin
Data Platform Engineering, Stitch Fix

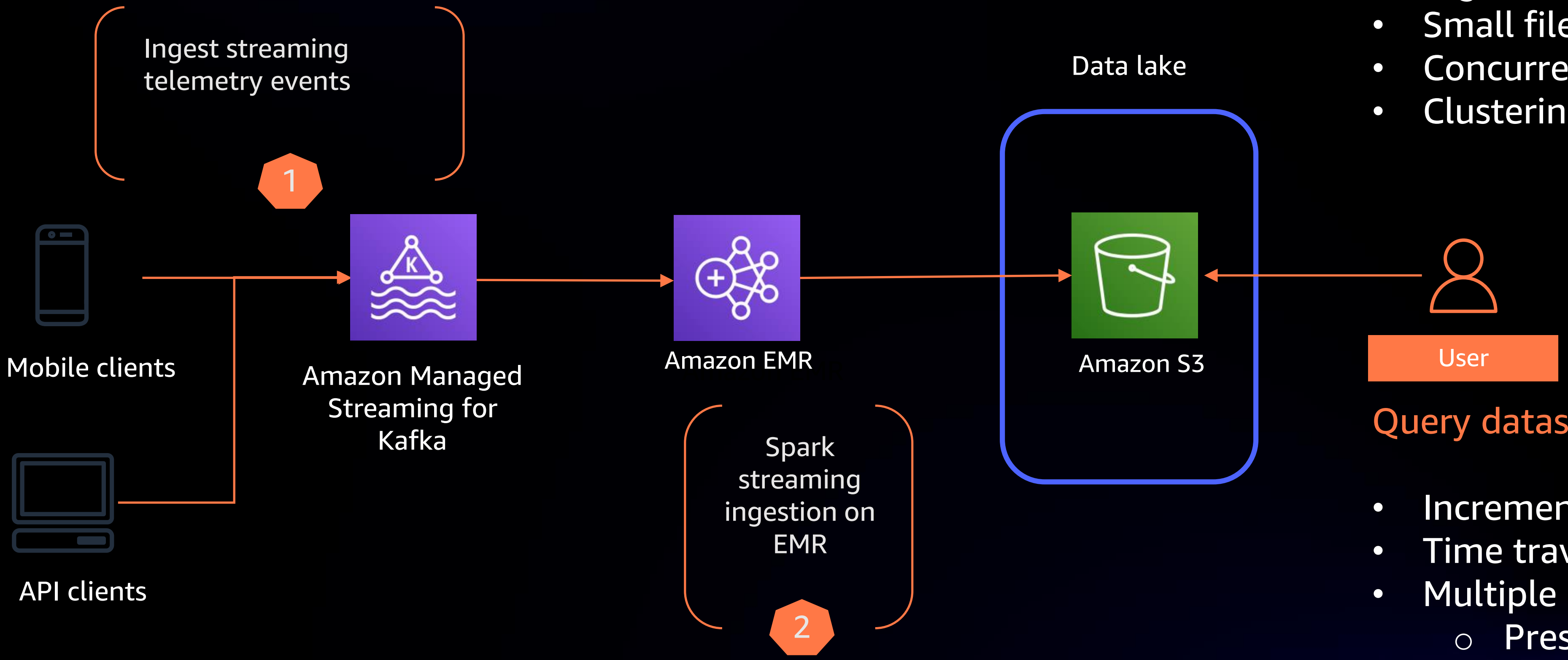
RIVIAN

"The Battery Data Science Team at Rivian redefines how batteries are developed, monitored, and improved with near-real-time processing of massive datasets from our R&D labs and vehicle test fleet. As Rivian begins production and the volume of our battery data grows significantly, EMR on EKS enables our team to seamlessly scale our analytics capabilities. We use a variety of AWS-managed services, including ECR for Docker, EKS with Fargate for Kubernetes, and MSK for Kafka, for the flexibility and development speed of the open-source frameworks we want without the headaches of managed infrastructure."

Rob Jenks
Principal Software Engineer – Cloud, Rivian

EMR simplifies building data lakes

Streaming data ingestion pipelines


Ingest streaming telemetry events
1
Mobile clients
API clients
Amazon Managed Streaming for Kafka
Amazon EMR
Spark streaming ingestion on EMR
2
Data lake
Amazon S3
User

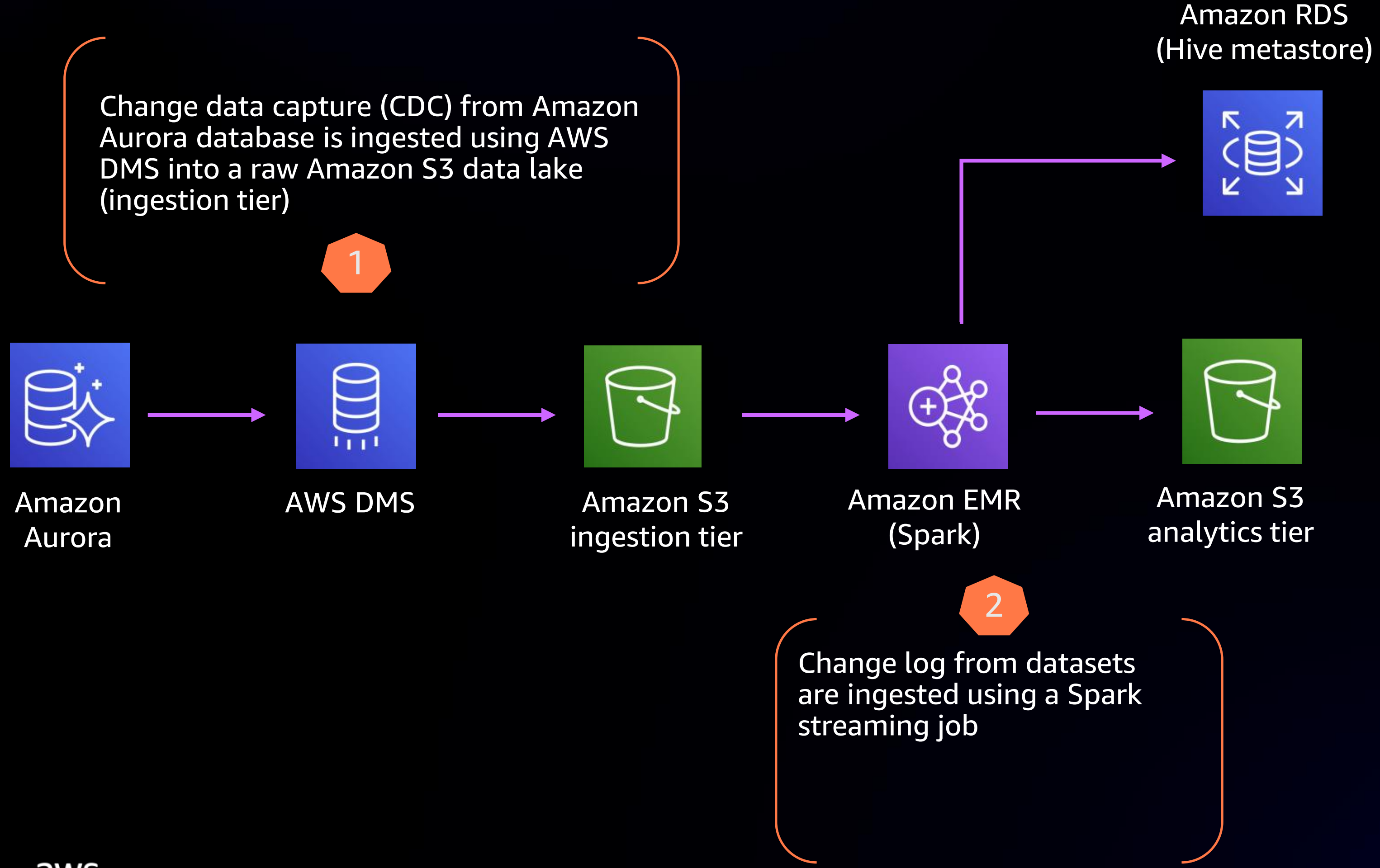
Writing dataset challenges

- Make atomic changes
- Reader – writer isolation
- High throughput ingestion
- Small file compactions
- Concurrent writers
- Clustering by secondary keys

Query dataset challenges

- Incremental query
- Time travel query
- Multiple engine support
  - Presto, Hive, Spark SQL
  - Amazon Athena, Amazon Redshift Spectrum

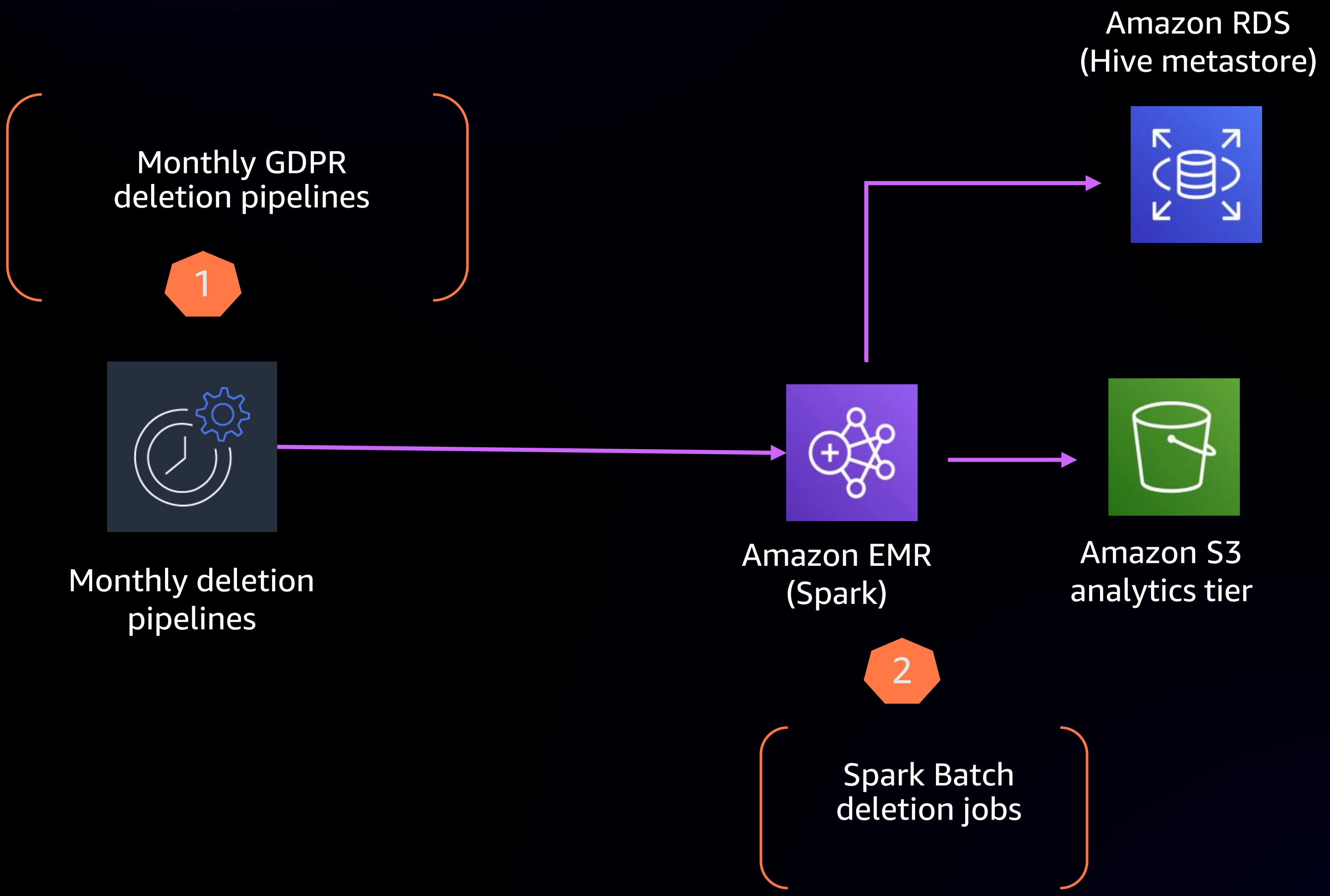
CDC ingestion pipelines challenges


Amazon RDS
(Hive metastore)
Change data capture (CDC) from Amazon Aurora database is ingested using AWS DMS into a raw Amazon S3 data lake (ingestion tier)
1
Amazon Aurora
AWS DMS
Amazon S3 ingestion tier
Amazon EMR (Spark)
Amazon S3 analytics tier
2
Change log from datasets are ingested using a Spark streaming job

Challenges

- Make atomic changes
- Reader – writer isolation
- High throughput ingestion
- Small file compactions
- Row-level upserts and deletes
- Clustering by secondary keys

GDPR (data erasure) pipelines challenges


Amazon RDS
(Hive metastore)
Monthly GDPR deletion pipelines
1
Monthly deletion pipelines
Amazon EMR (Spark)
Amazon S3 analytics tier
2
Spark Batch deletion jobs

Challenges

- Row-level upserts and deletes
- Concurrent writers

Apache Hudi enables transactional data lakes

TRANSACTIONS, RECORD-LEVEL UPDATES/DELETES, AND CHANGE STREAMS TO DATA LAKES!

Ingestion

- Transactions (ACID) – reader and writer isolation
- Transactions (ACID) – concurrent writer support
- Record-level upserts and deletes
- High throughput streaming ingestion
- Spark, Flink, and Java Writer Support
- Automatic compaction of small files
- Spark SQL DML support (Hudi 0.9.0) NEW!

Query

- Spark, PrestoDB/Trino, and Hive support
- Efficient queries across partitions and files
- Incremental query support
- Time travel query support

Apache Hudi enables transactional data lakes

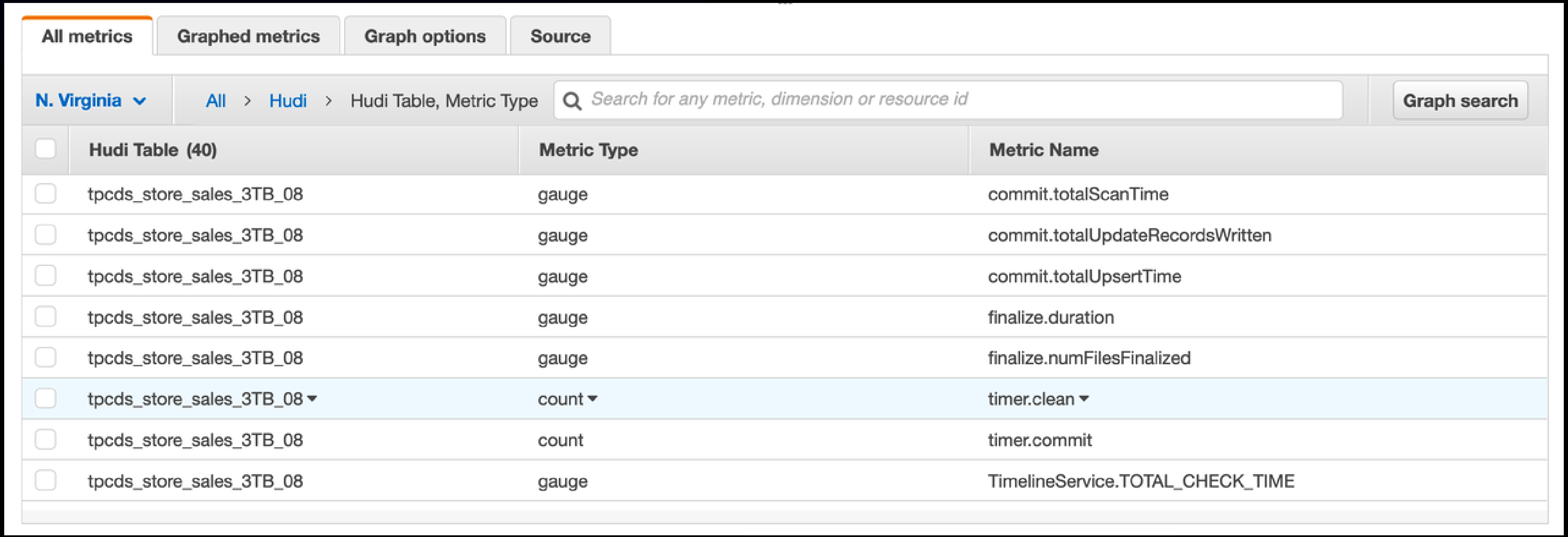
AUTOMATE TABLE MANAGEMENT ACTIVITIES

Administration

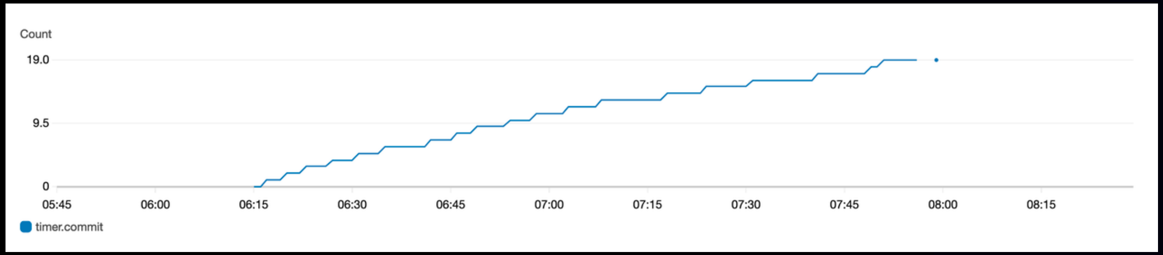
- Async background compaction of files
- Async background sorting and clustering of keys
- Automatically clean up files beyond retention period
- Metrics for past commits or rollbacks

Easily operationalize at scale using detailed metrics

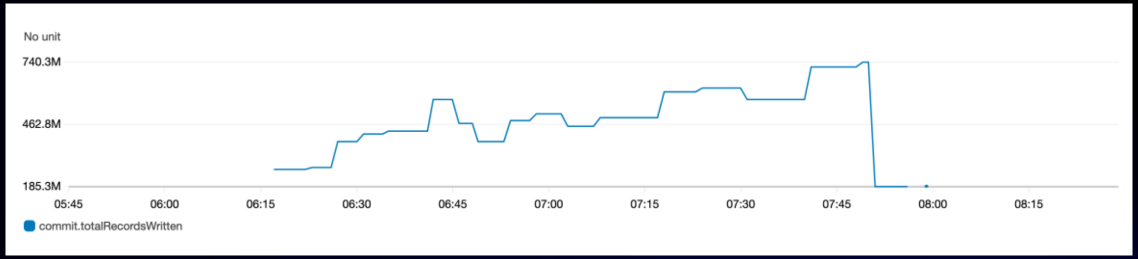
AMAZON CLOUDWATCH INTEGRATION


All metrics
Graphed metrics
Graph options
Source
N. Virginia
All > Hudi > Hudi Table, Metric Type
Search for any metric, dimension or resource id
Graph search

Hudi Table (40)	Metric Type	Metric Name
tpcds_store_sales_3TB_08	gauge	commit.totalScanTime
tpcds_store_sales_3TB_08	gauge	commit.totalUpdateRecordsWritten
tpcds_store_sales_3TB_08	gauge	commit.totalUpsertTime
tpcds_store_sales_3TB_08	gauge	finalize.duration
tpcds_store_sales_3TB_08	gauge	finalize.numFilesFinalized
tpcds_store_sales_3TB_08	count	timer.clean
tpcds_store_sales_3TB_08	count	timer.commit
tpcds_store_sales_3TB_08	gauge	TimelineService.TOTAL_CHECK_TIME



No. of commits



Total records written

Apache Hudi is widely supported on AWS

BROAD ECOSYSTEM SUPPORT FOR APACHE HUDI ON AWS

Spark, Hive, Presto, Flink
Support on Amazon EMR

AWS Glue Catalog
and ETL support

AWS Lake Formation
FGAC support

Amazon Athena
Native query support

Amazon Redshift Spectrum
Native query support

AWS Database Migration Service
CDC ingestion support

Amazon CloudWatch
Integration for metrics

Security controls in EMR

Comprehensive security features

Isolation

- VPC
- Private subnets
- Security groups

Authentication

- LDAP
- Kerberos
- AWS SSO (EMR Studio)
- AWS IAM (EMR Studio)

Authorization

- Cluster IAM role
- User execution role (preview) NEW!
- FGAC using Apache Ranger
- FGAC using AWS Lake Formation (preview) NEW!

Encryption

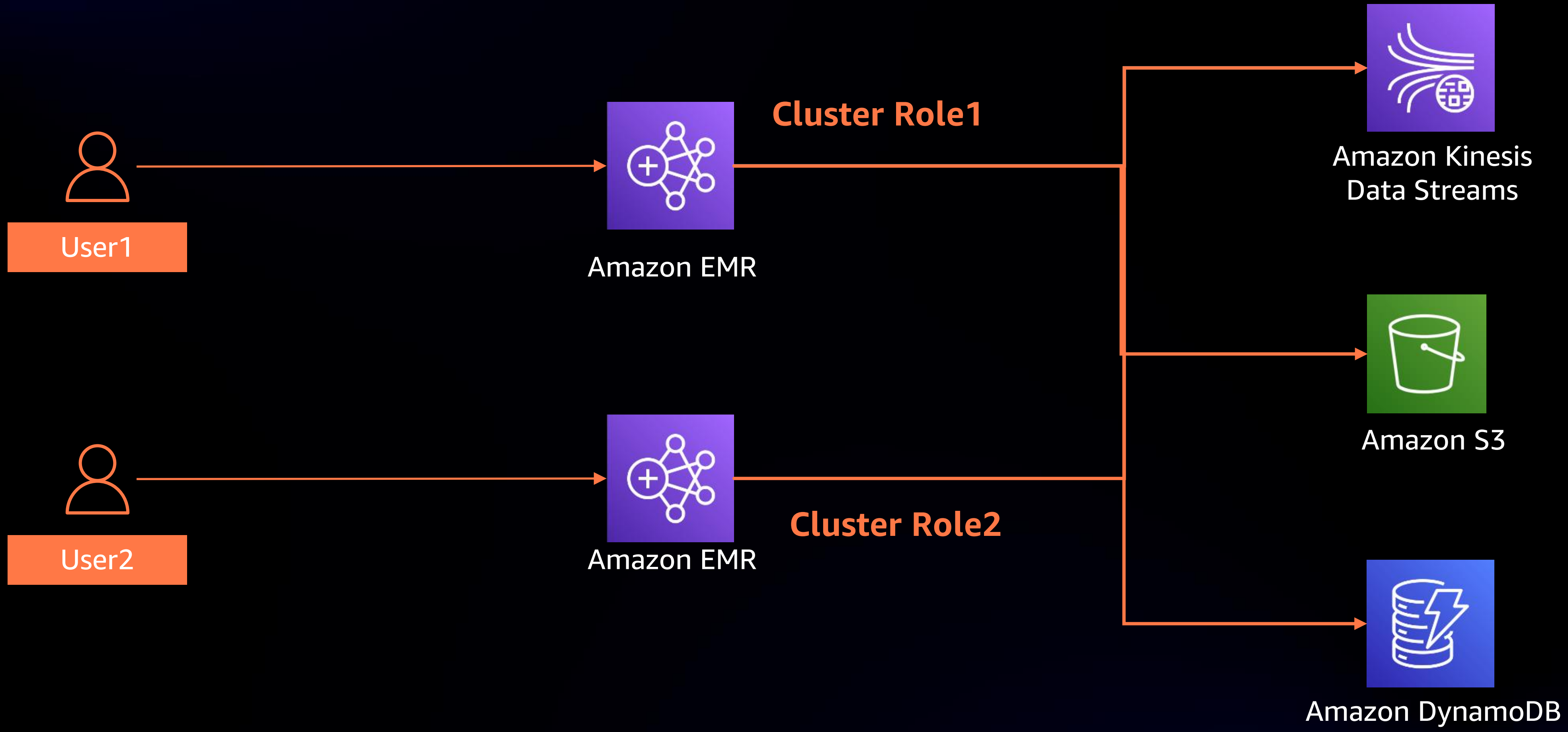
- Encryption at rest
- Encryption in transit
- Key management

Audit

- Audit using Ranger via Amazon CloudWatch Logs
- Audit using AWS Lake Formation via AWS CloudTrail NEW!

aws

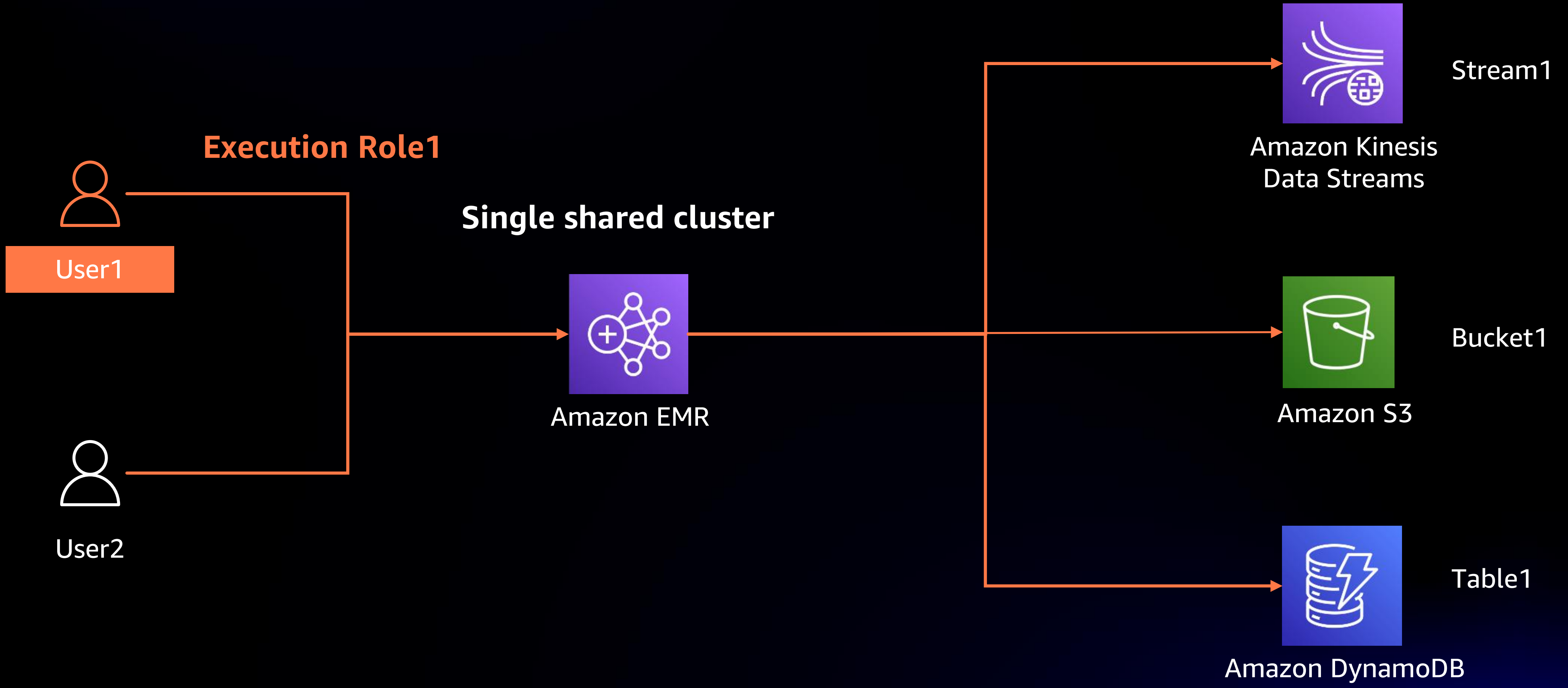
One cluster – One role



Enable multi-tenant shared clusters

NEW!

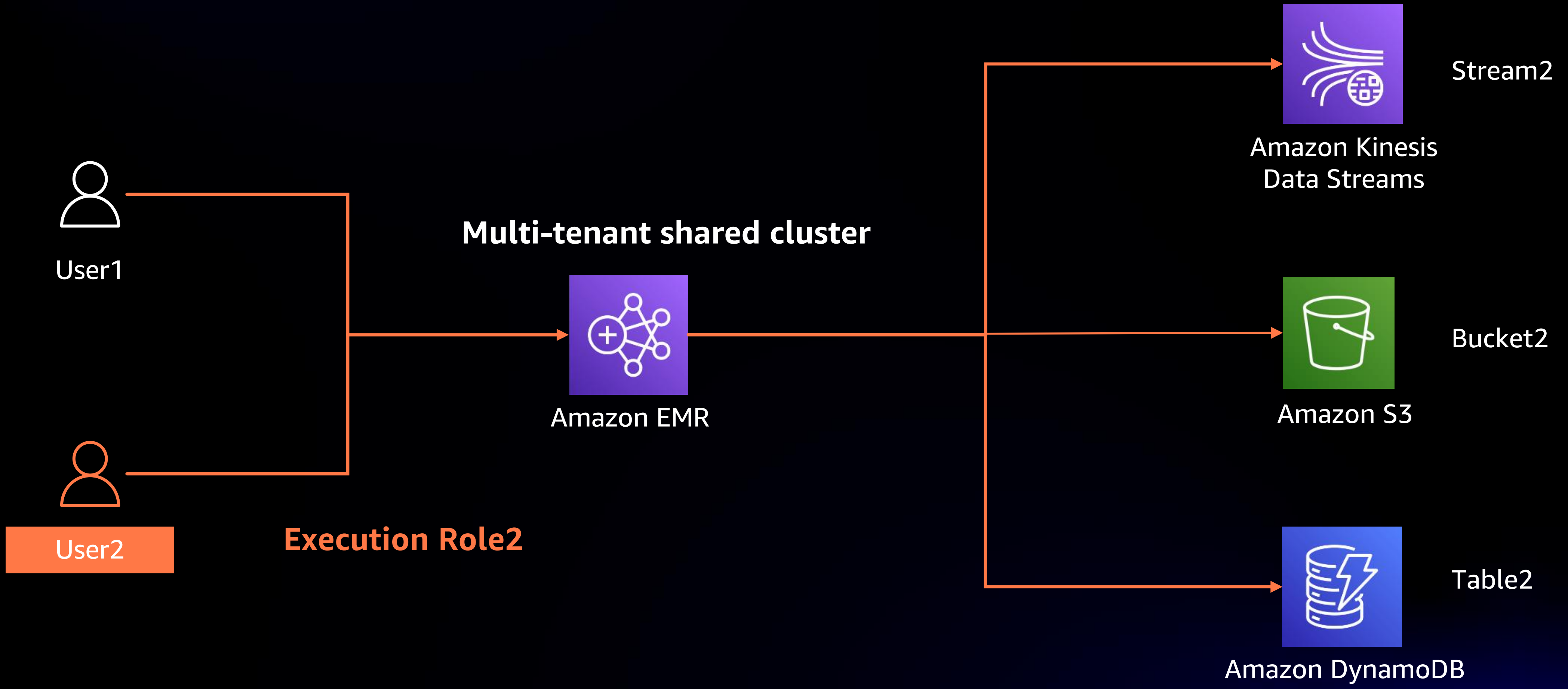
User execution role (preview): User1 has access to Stream1, Bucket1, and Table1



Enable multi-tenant shared clusters

NEW!

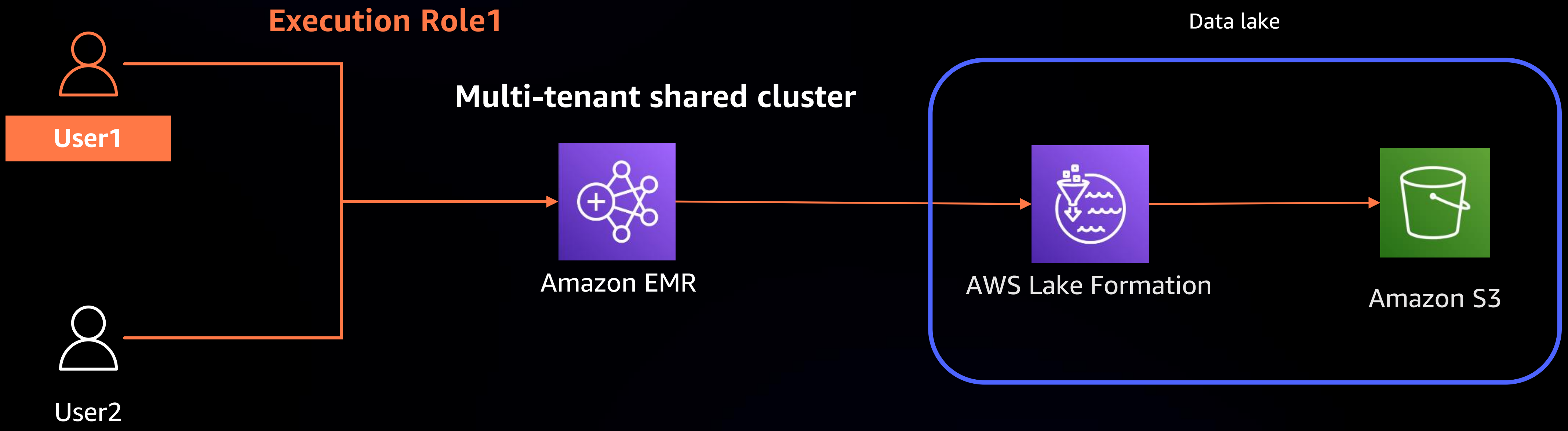
User Execution Role (preview): User2 has access to Stream2, Bucket2, and Table2



Enable fine-grained access control

NEW!

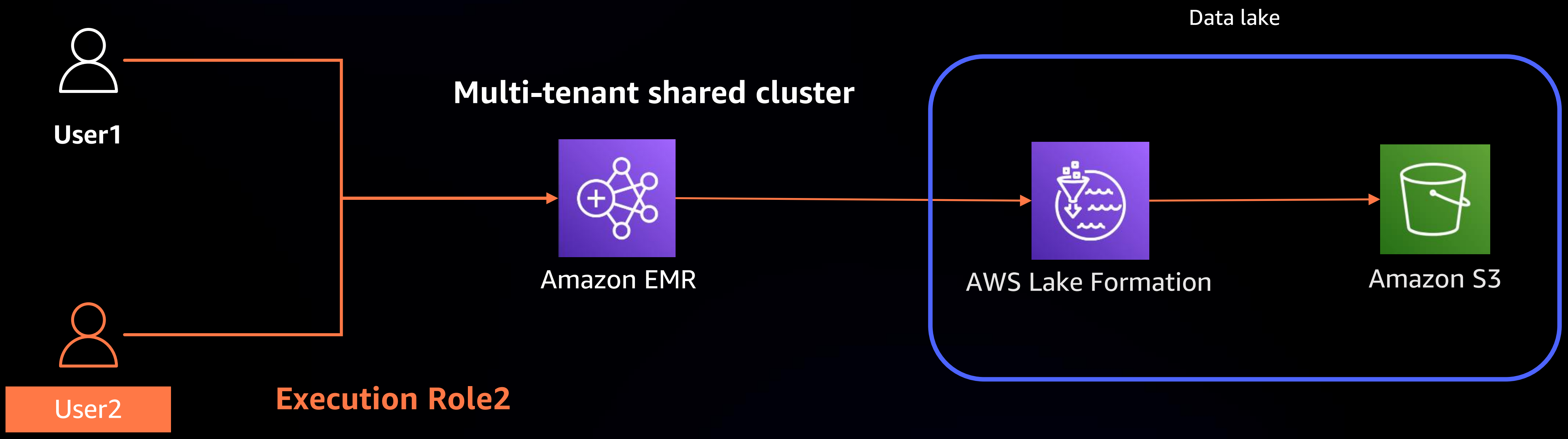
FGAC using AWS Lake Formation (preview): User1 has access to Table1, Columns 1–10



Enable fine-grained access control

NEW!

FGAC using AWS Lake Formation (preview): User2 has access to Table1, Columns 5–10



Thank you!

Vincent Gromakowski

gromav@amazon.com