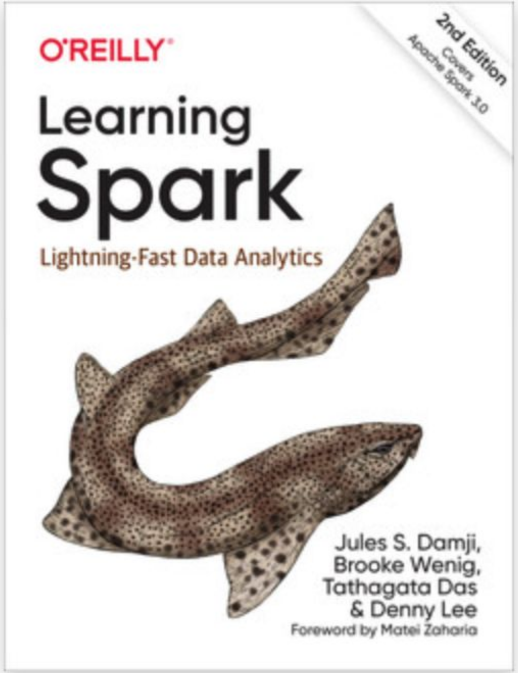
Introduction to Ray for Distributed Applications

Jules Damji, Anyscale
@2twitme

RAY SUMMIT 22

$whoami (Jules)

- Lead Developer Advocate @Anyscale
- Senior Developer Advocate @Databricks
- Led Developer Advocacy @Hortonwork
- Held SWE positions:
  - Sun Microsystems
  - Netscape
  - @Home
  - Loudcloud/Opsware
  - Verisign

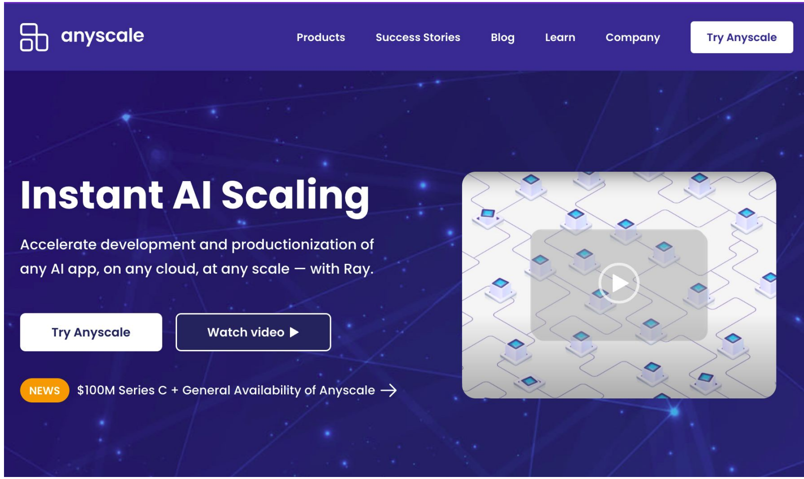

O'REILLY
2nd Edition
Covers Apache Spark 3.0
Learning Spark
Lightning-Fast Data Analytics
Jules S. Damji, Brooke Wenig, Tathagata Das & Denny Lee
Foreword by Matei Zaharia

Anyscale

Who we are: Original creators of Ray

What we do: Managed compute platform to scale AI & Python workloads

Why do it: Scaling is a necessity, scaling is hard; make distributed computing easy and simple for everyone



Instant AI Scaling

Accelerate development and productionization of any AI app, on any cloud, at any scale — with Ray.

NEWS $100M Series C + General Availability of Anyscale →

Agenda

→ Why & What's Ray & Ray Ecosystem

→ Ray Architecture & Components

→ Ray Core Design Patterns & APIs

→ Modules [1 - 3]

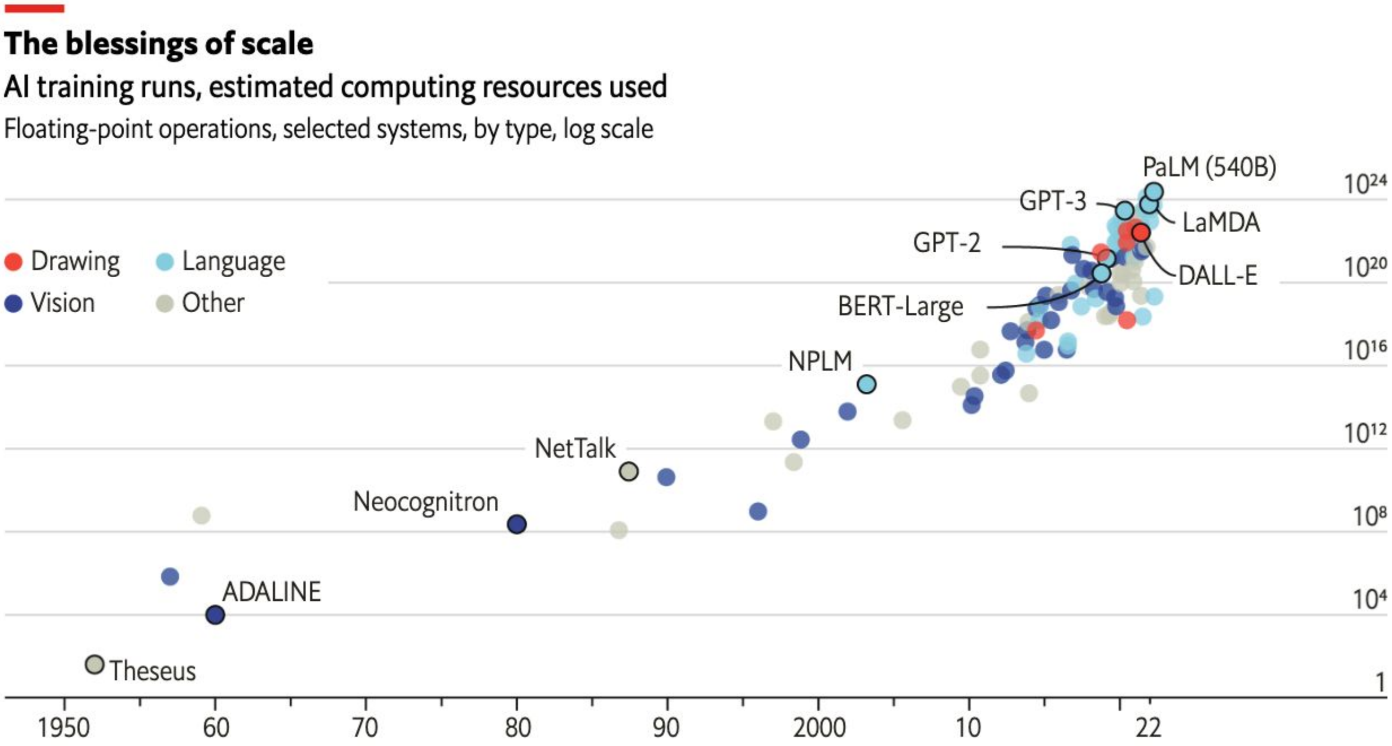
→ Closing Q & A

→ Happy Hour 🍻 + Meetup

Why Ray

- → Machine learning is pervasive
- → Distributed computing is a necessity
- → Python is the default language for DS/ML

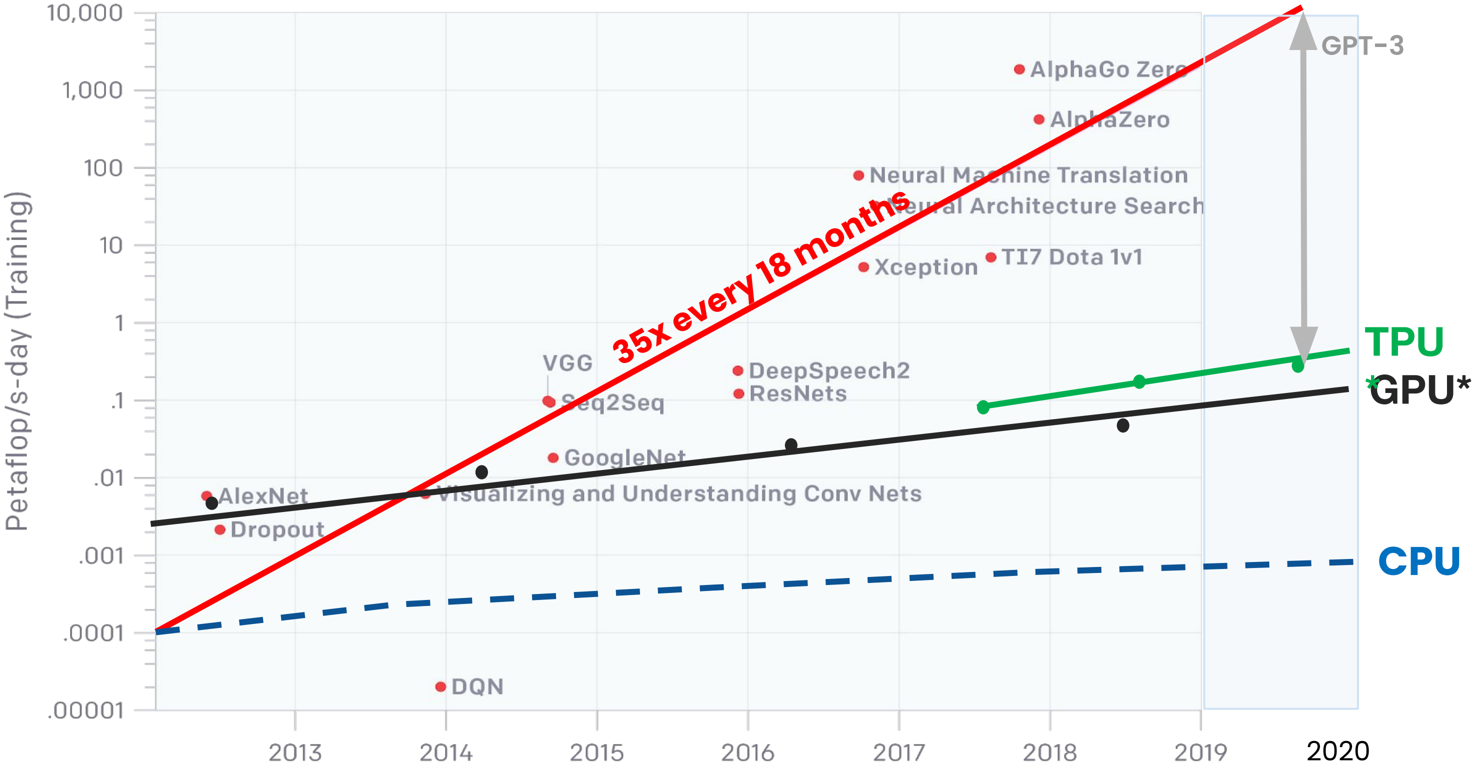
Blessings of scale …



Sources: "Compute trends across three eras of machine learning", by J. Sevilla et al., arXiv, 2022; Our World in Data

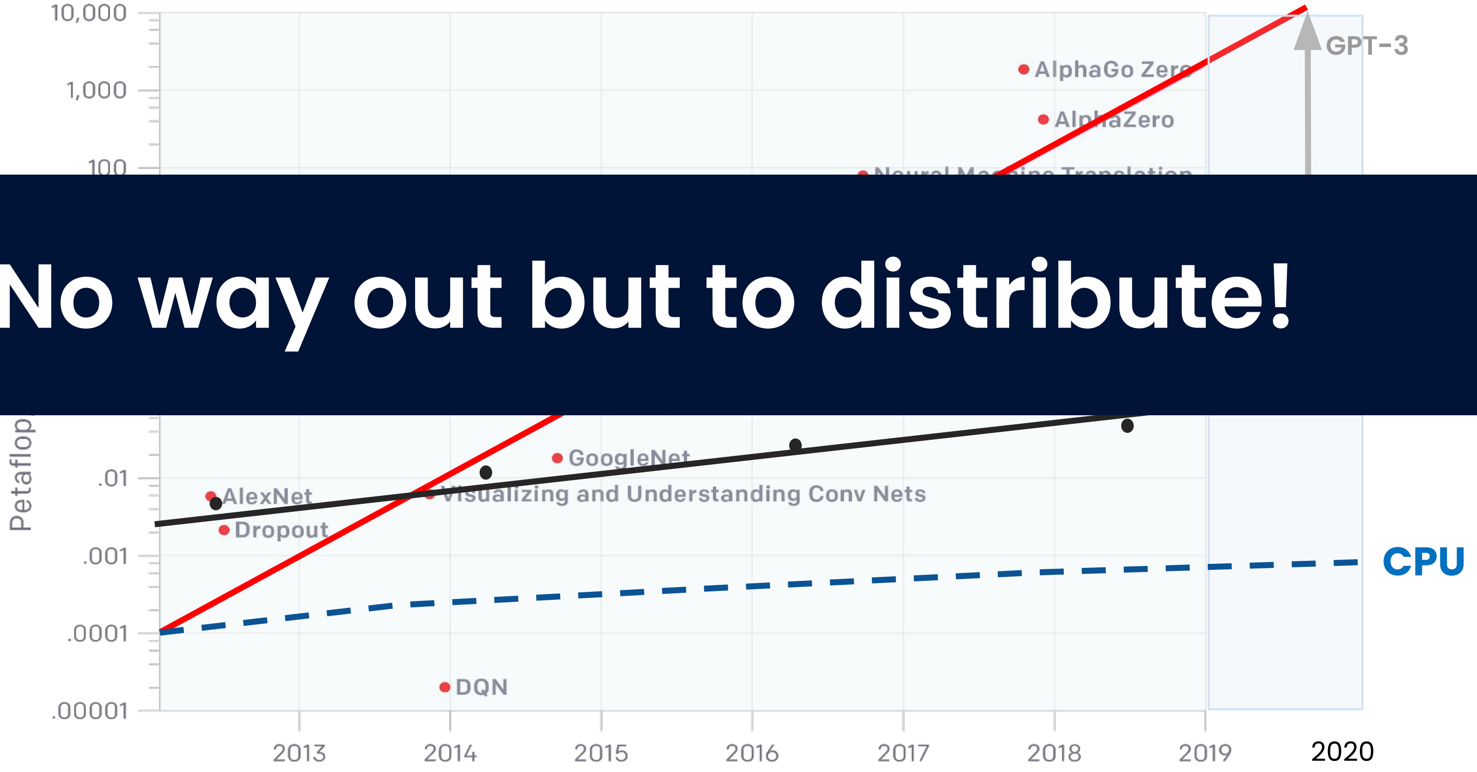
RAY SUMMIT 22

Compute – supply demand problem



https://openai.com/blog/ai-and-compute/

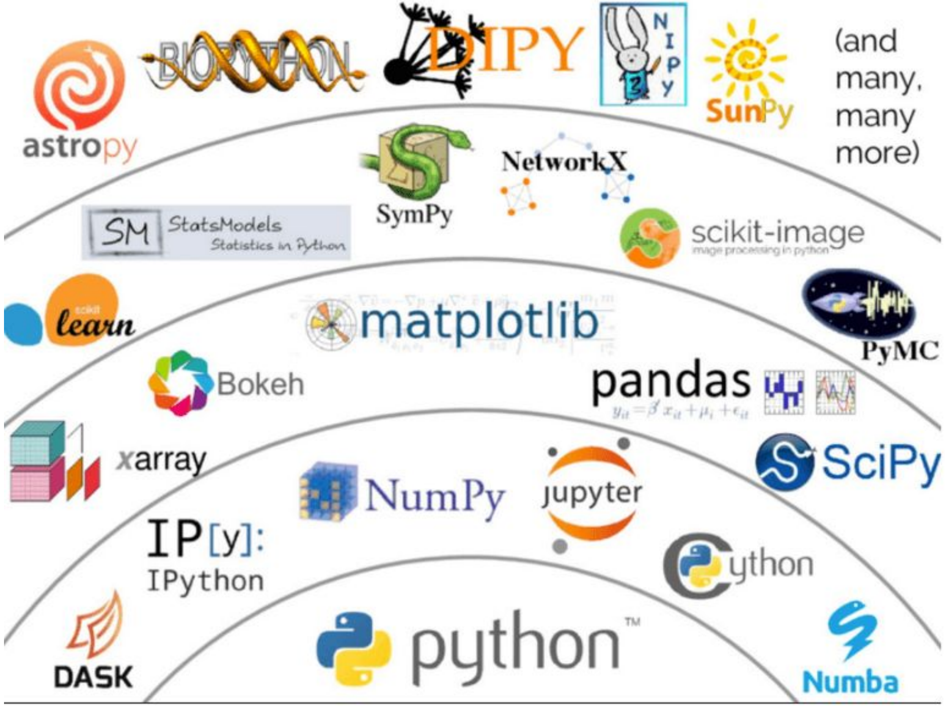
Specialized hardware is not enough

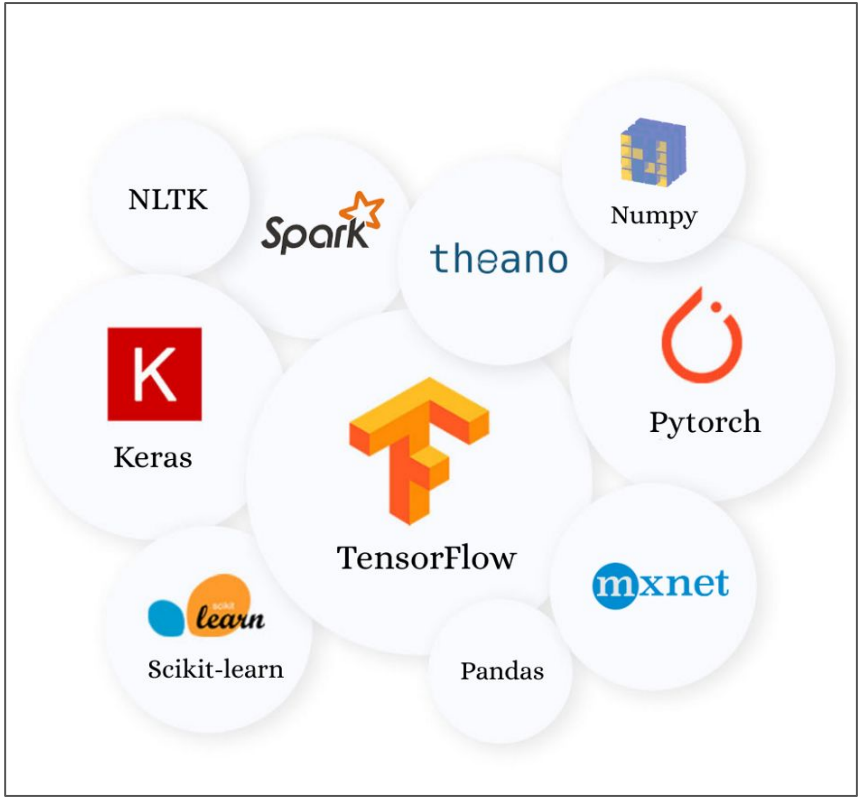


No way out but to distribute!

https://openai.com/blog/ai-and-compute/

Python data science ecosystem



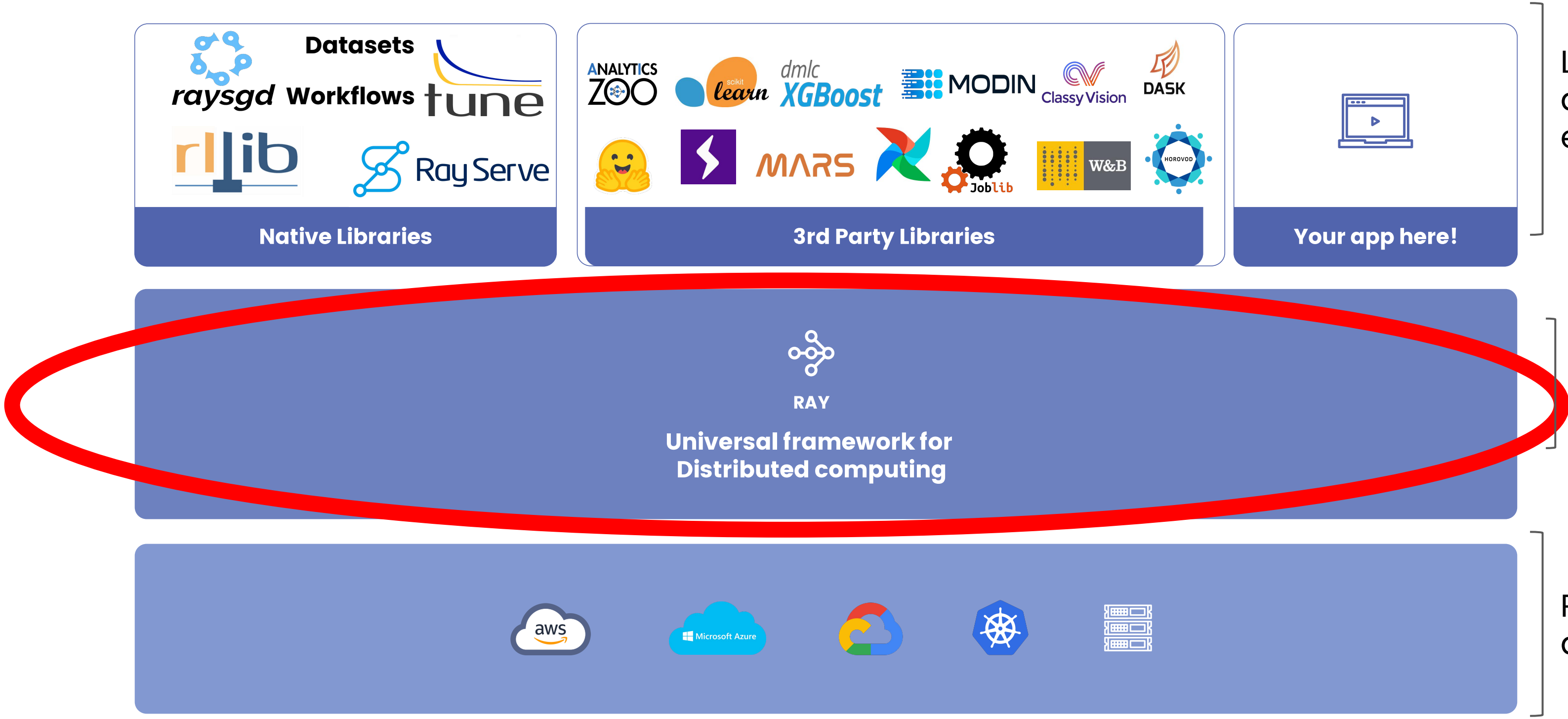


What is Ray

- → A simple/general-purpose library for distributed computing
- → An ecosystem of Python libraries (for scaling ML and more)
- → Runs on laptop, public cloud, K83, on-premise

A layered cake of functionality and capabilities for scaling ML workloads

A Layered Cake and Ecosystem



Native Libraries

3rd Party Libraries

Your app here!

RAY

Universal framework for Distributed computing

Library + app ecosystem

Universal framework for distributed computing

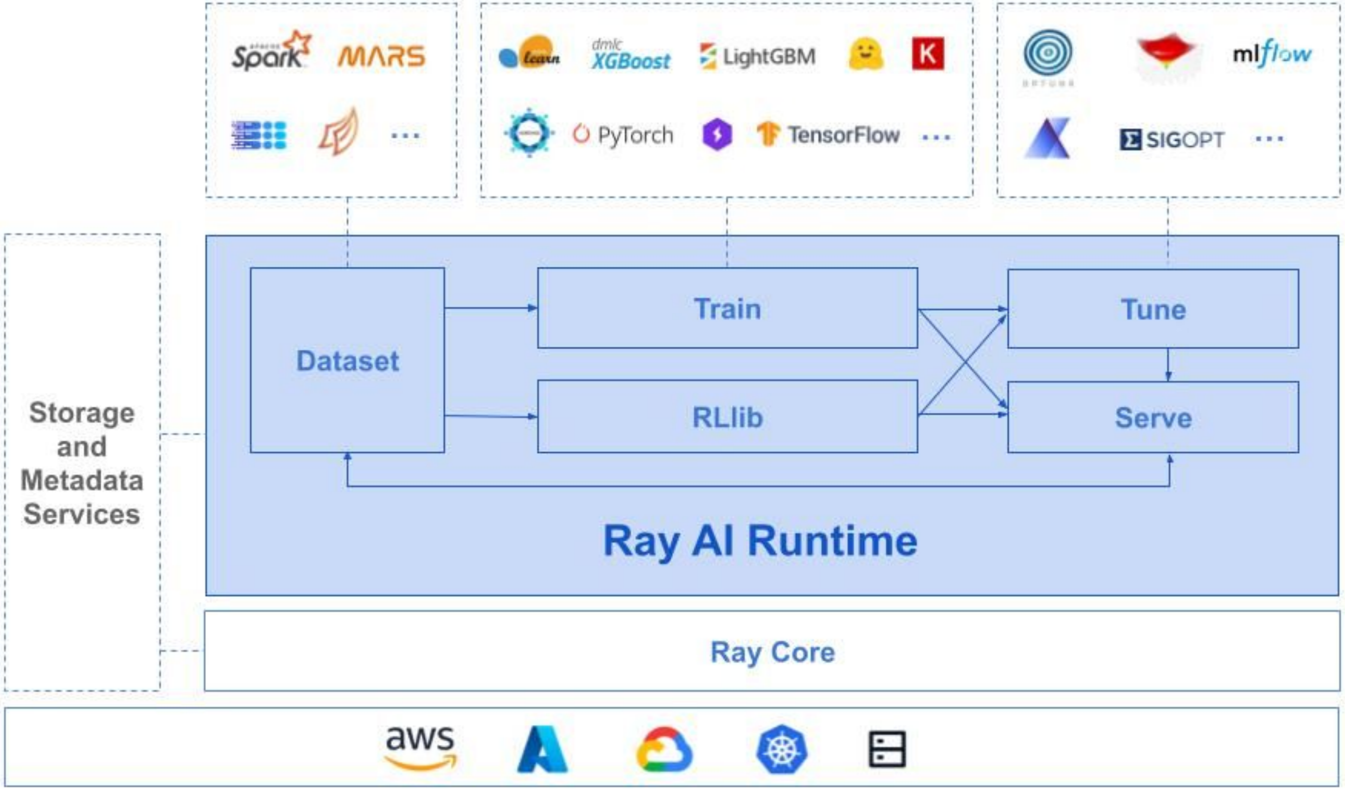
Run anywhere

Rich ecosystem of scaling ML workloads

	Data Processing	Training	Serving	Hyper. Tuning	Reinforcement Learning
Built-in **“batteries included”** libraries	Ray Core + Datasets	Ray Train	Ray Serve	tune	rllib
	MODIN MARS DASK	PyTorch scikit learn HOROVOD	SELDON	HYPEROPT OPTUNA	

Only use the libraries you need!

Ray AI Runtime (Ray AIR)



Ray 2.0 & Ray AIR sessions

→ Introduction Ray AI Runtime

→ State of Ray Serve in 2.0

→ Shuffling 100TB with Ray Datasets

→ Ray Observability: Present & future

→ Many others in Ray Deep Dives track …

Who’s using it

amazon

intel

dendra SYSTEMS

Uber

Microsoft

VISA

McKinsey & Company

ANT GROUP

rbi restaurant brands international

WILD LIFE

robinhood

NVIDIA

shopify

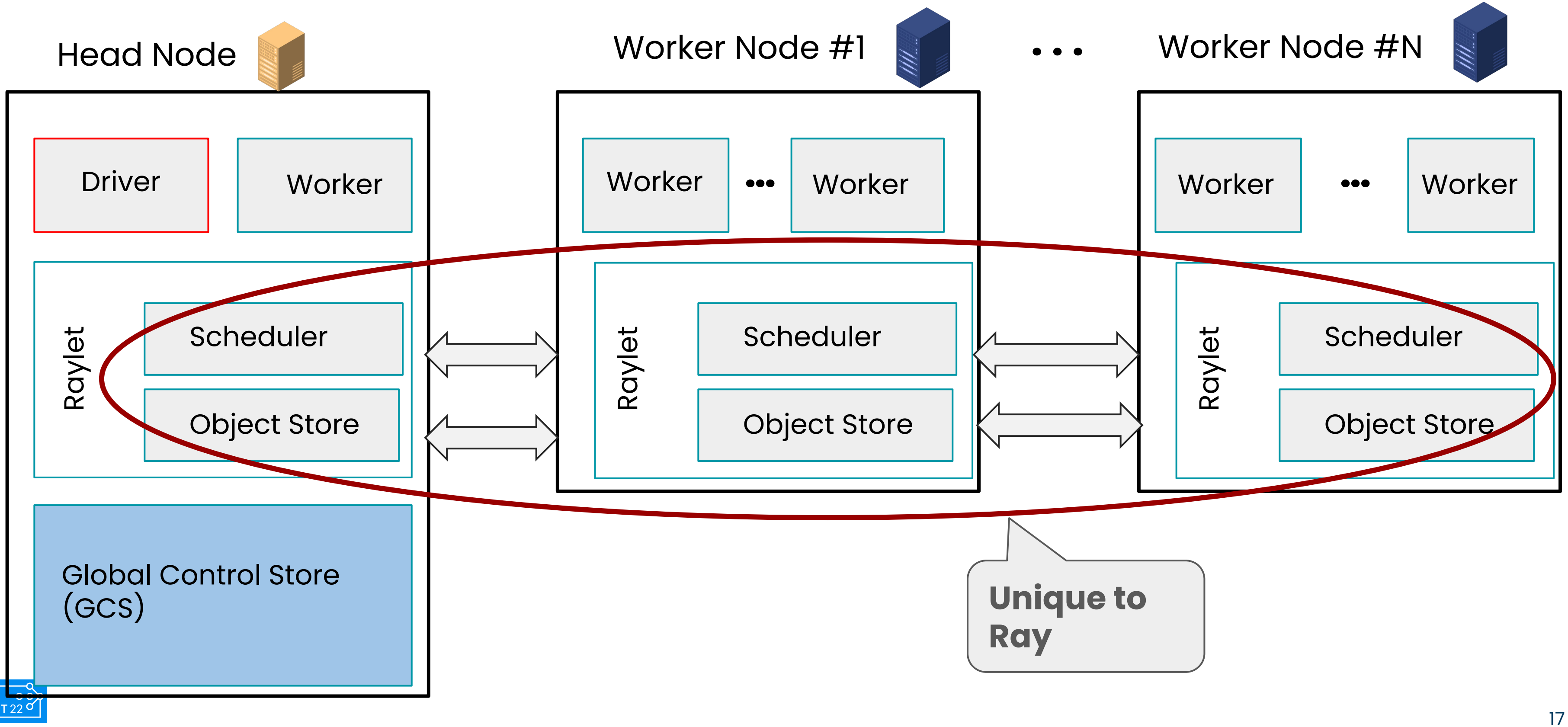
IBM

2σ TWO SIGMA

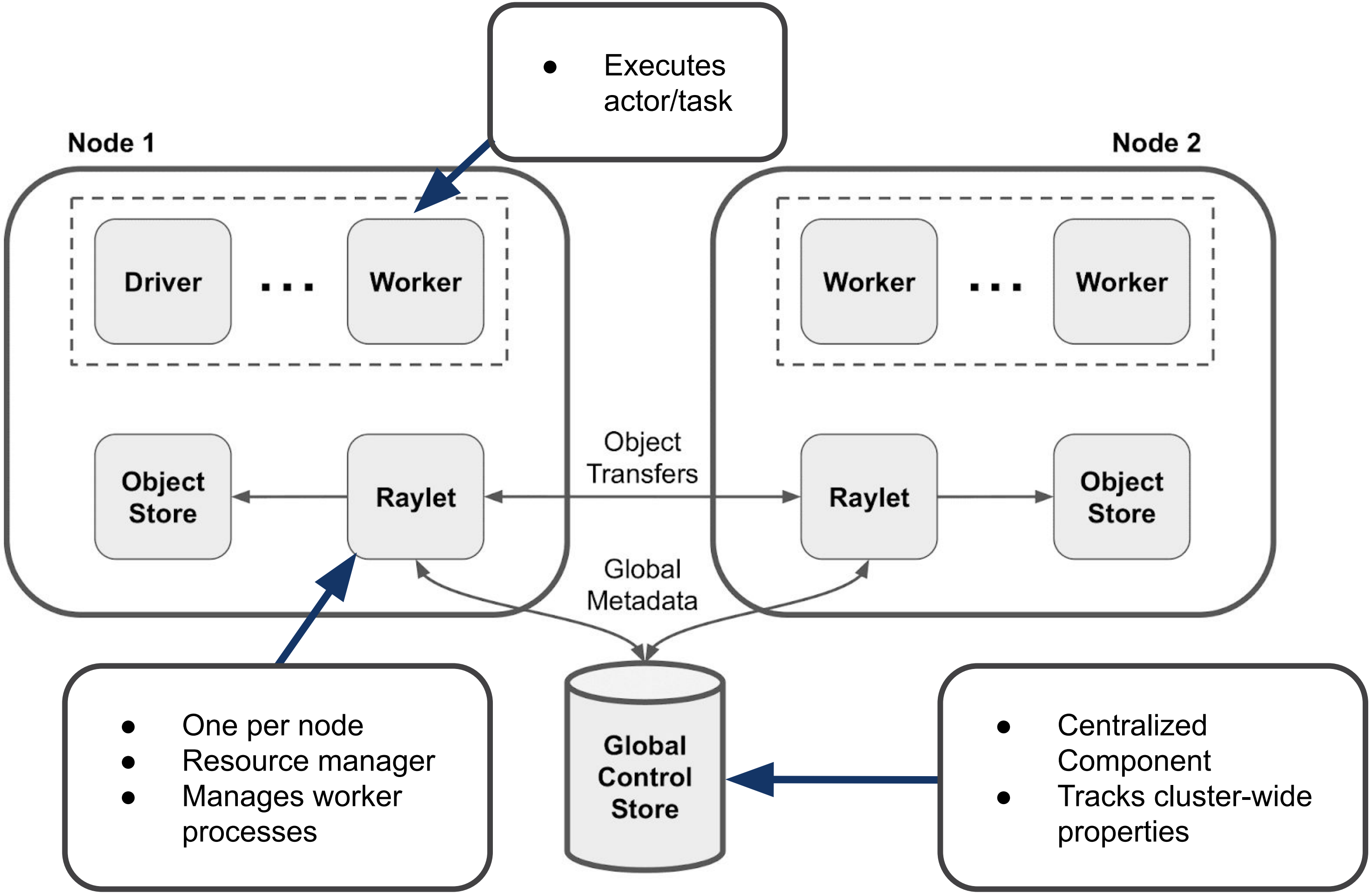
Alibaba.com

Ray Architecture & Components

An anatomy of a Ray cluster



Unique to Ray



- Executes actor/task

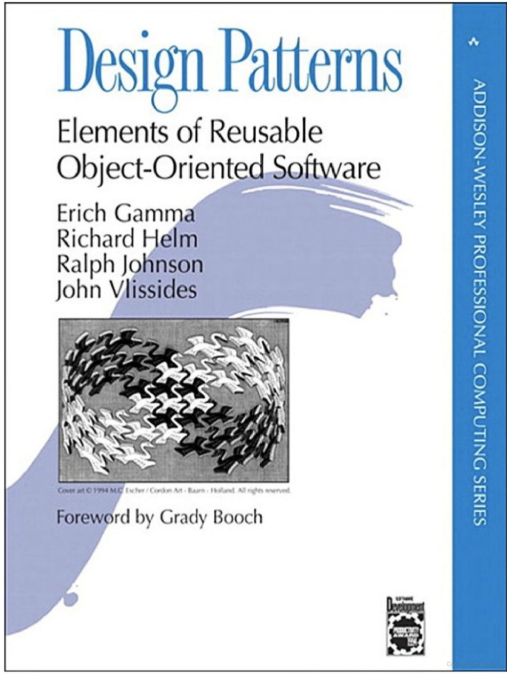
- One per node
- Resource manager
- Manages worker processes

- Centralized Component
- Tracks cluster-wide properties

Ray distributed design patterns & APIs

Ray Basic Design Patterns

- → Ray Parallel Tasks
  - + Functions as stateless units of execution
  - + Functions distributed across the cluster as tasks
- → Ray Objects as Futures
  - + Distributed (immutable objects) store in the cluster
  - + Fetched when materialized
  - + Enable massive asynchronous parallelism
- → Ray Actors
  - + Stateful service on a cluster
  - + Enable Message passing


Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

1. Patterns for Parallel Programming
2. Ray Design Patterns
3. Ray Distributed Library Integration Patterns

RAY SUMMIT 22

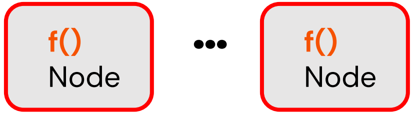
Python → Ray APIs

```
def f(x):
   # do something with x:
   y= …
   return y
```

Task

```
@ray.remote
def f(x):
   # do something with x:
   y= …
   return y
```

Distributed

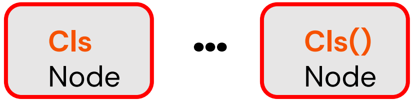

f()
Node
•••
f()
Node

```
class Cls():
  def __init__(self,
x):
  def f(self, a):
       …
  def g(self, a):
       …
```

Actor

```
@ray.remote
class Cls():
  def
__init__(self, x):
  def f(self, a):
       …
  def g(self, a):
       …
```

Distributed

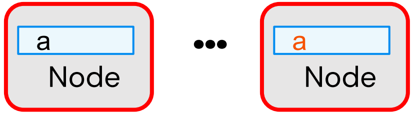

Cls
Node
•••
Cls()
Node

```
import numpy as np
a= np.arange(1, 10e6)
b = a * 2
```

Distributed immutable object

```
import numpy as np
a = np.arange(1, 10e6)
obj_a = ray.put(a)
b = ray.get(obj_a) * 2
```

Distributed


a
Node
•••
a
Node

Function → Task

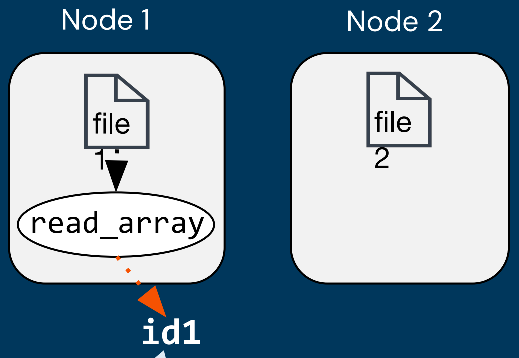
```
@ray.remote
def read_array(file):
    # read ndarray “a”
    # from “file”
    return a

@ray.remote
def add(a, b):
    return np.add(a, b)

id1 = read_array.remote(file1)
id2 = read_array.remote(file2)
id = add.remote(id1, id2)
sum = ray.get(id)
```

Class → Actor

```
@ray.remote(num_gpus=1)
class Counter(object):
    def __init__(self):
        self.value = 0
    def inc(self):
        self.value += 1
        return self.value

c = Counter.remote()
id4 = c.inc.remote()
id5 = c.inc.remote()
```

Task API

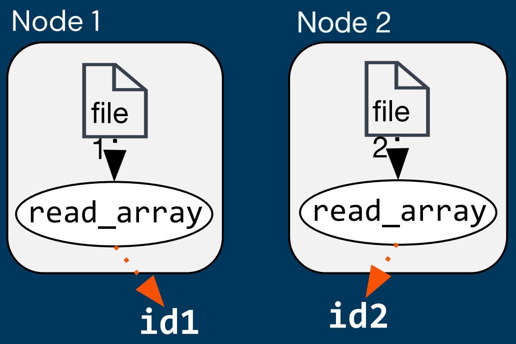
```
@ray.remote
def read_array(file):
    # read ndarray “a”
    # from “file”
    return a

@ray.remote
def add(a, b):
    return np.add(a, b)

id1 = read_array.remote(file1)
id2 = read_array.remote(file2)
id = add.remote(id1, id2)
sum = ray.get(id)
```



Return id1 (future) immediately, before read_array() finishes

Task API

```
@ray.remote
def read_array(file):
    # read ndarray “a”
    # from “file”
    return a

@ray.remote
def add(a, b):
    return np.add(a, b)

id1 = read_array.remote(file1)
id2 = read_array.remote(file2)
id = add.remote(id1, id2)
sum = ray.get(id)
```

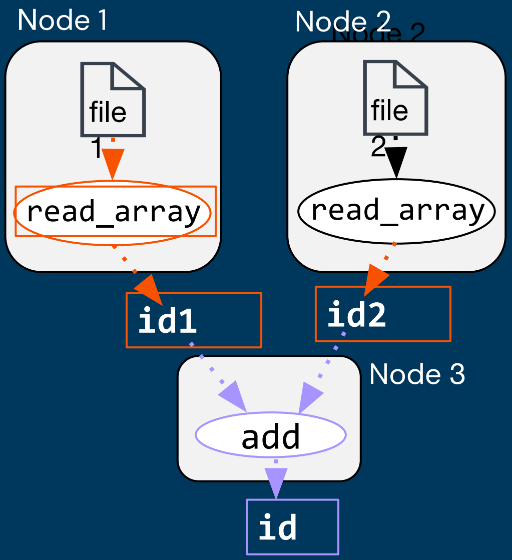


Dynamic task graph: build at runtime

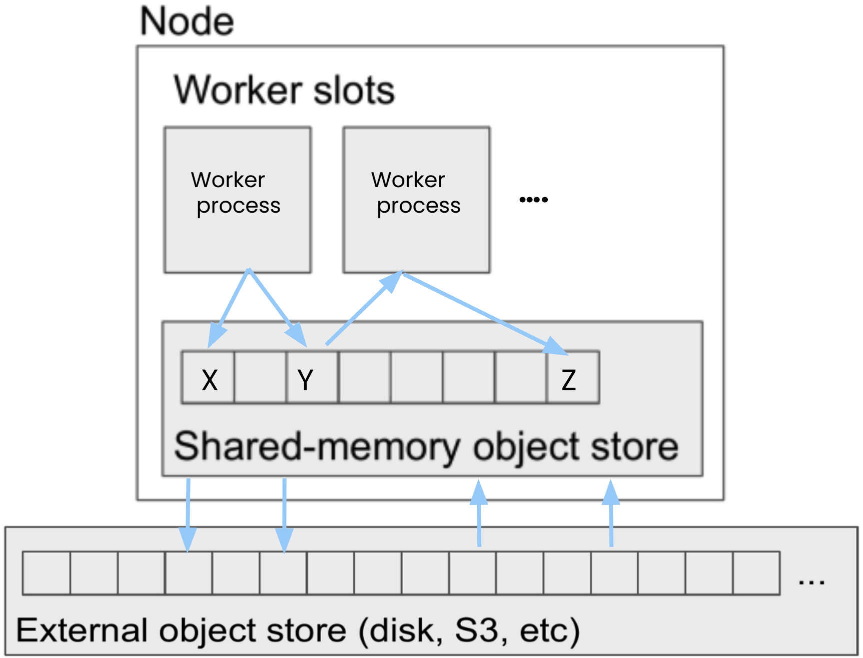
Task API

```
@ray.remote
def read_array(file):
    # read ndarray “a”
    # from “file”
    return a

@ray.remote
def add(a, b):
    return np.add(a, b)

id1 = read_array.remote(file1)
id2 = read_array.remote(file2)
id = add.remote(id1, id2)
sum = ray.get(id)
```

ray.get() block until result available

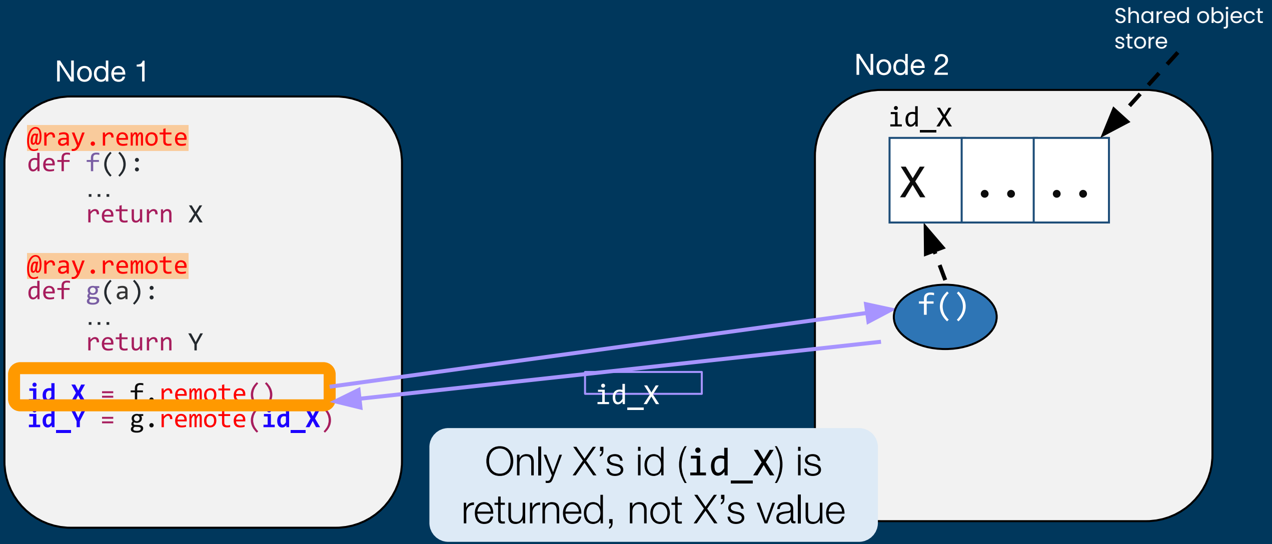


Distributed Immutable object store


Node
Worker slots
Worker process
Worker process
....
X
Y
Z
Shared-memory object store
...
External object store (disk, S3, etc)

Spill over to external storage

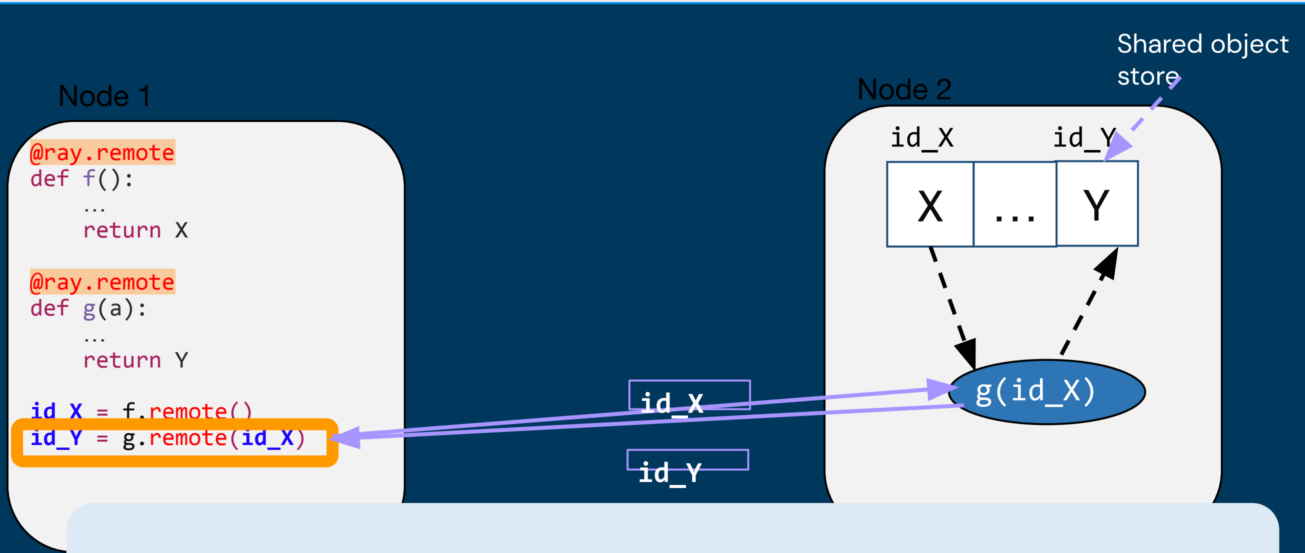
Distributed object store



```
@ray.remote
def f():
    ...
    return X

@ray.remote
def g(a):
    ...
    return Y

id_X = f.remote()
id_Y = g.remote(id_X)
```

Only X's id (`id_X`) is returned, not X's value

Distributed object store



```
@ray.remote
def f():
    ...
    return X

@ray.remote
def g(a):
    ...
    return Y

id_X = f.remote()
id_Y = g.remote(id_X)
```

g(id_X) is scheduled on same node, so X is never transferred

Examples of Distributed Applications with Ray

Distributed Applications with Ray

ML Libraries

- Ray AI Runtime
  - Ray Data, Tune, Train, Serve, RLlib
- Distributed scikit-learn/Joblib
- Distributed XGBoost on Ray
- Dask on Ray
- Modin on Ray

All using Ray design patterns

Monitoring Services

- WhyLabs
- Arize AI
- W & B

All using Ray design patterns

ML Platforms & Integrations

- Merlin (Shopify)
- Zero Copy (IBM)
- TorchX
- MLflow, Comet
- AirFlow
- HuggingFace
- Pycaret
- Ludwig AI

All using Ray design patterns

Ray Ecosystem: https://docs.ray.io/en/latest/ray-overview/ray-libraries.html

Key Takeaways

- → Distributed computing is a necessity & norm
- → Ray’s vision: make distributed computing simple
  - + Don’t have to be distributed programming expert
- → Build your own disruptive apps & libraries with Ray
- → Scale your ML workloads with Ray libraries (Ray AIR)

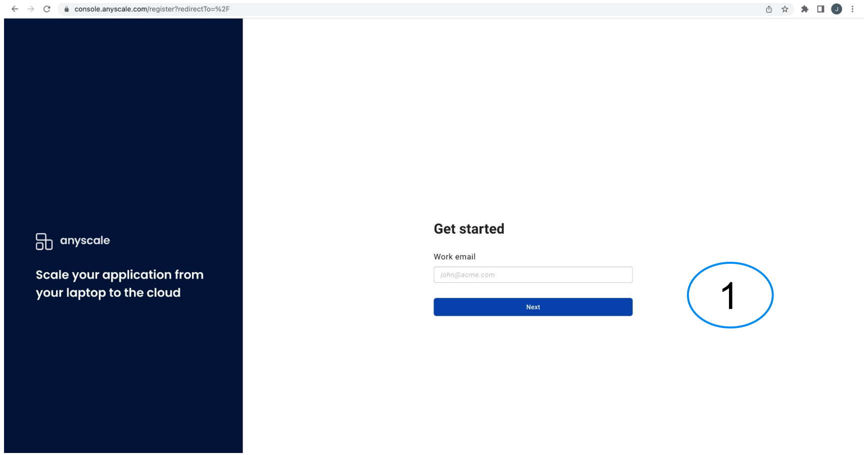
Let’s go with

Your Anyscale Cluster

→ Console: http://console.anyscale.com/

→ User name: <username@gmail.com>

→ Password : password



console.anyscale.com/register?redirectTo=%2F

anyscale

Scale your application from your laptop to the cloud

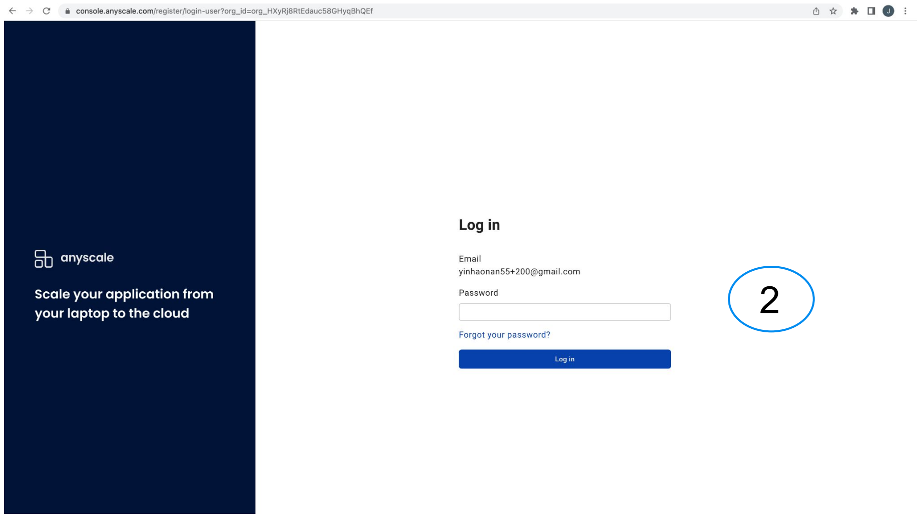
Get started

Work email

john@acme.com

Next

1



console.anyscale.com/register/login-user?org_id=org_HXyRj8RtEdauc58GHyqBhQEf

anyscale

Scale your application from your laptop to the cloud

Log in

Email
yinhaonan55+200@gmail.com

Password

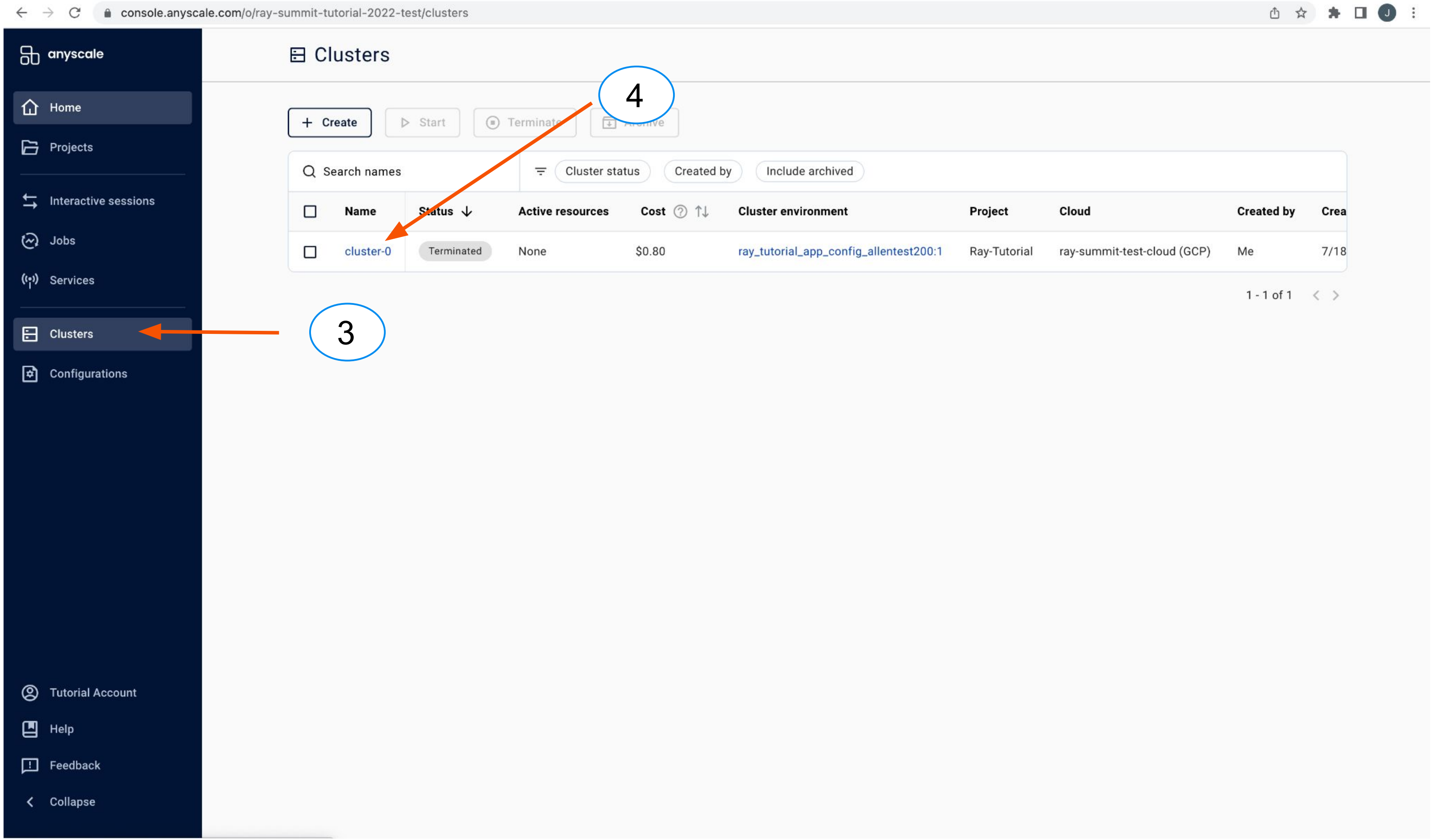
Forgot your password?

Log in

2

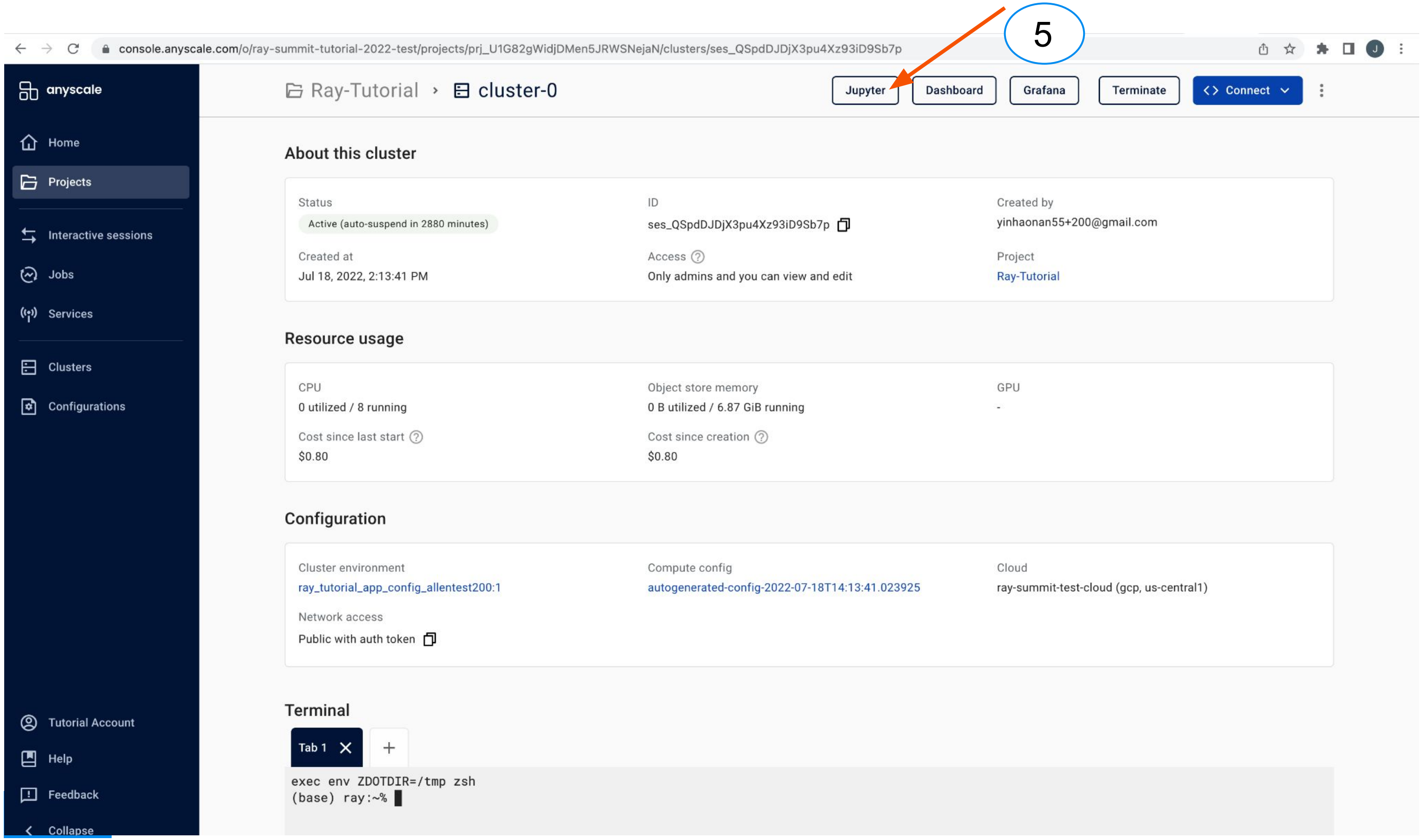
RAY SUMMIT 22

Your Anyscale Cluster



□	Name	Status	Active resources	Cost	Cluster environment	Project	Cloud	Created by	Crea
□	cluster-0	Terminated	None	$0.80	ray_tutorial_app_config_allentest200:1	Ray-Tutorial	ray-summit-test-cloud (GCP)	Me	7/18

Your Anyscale Cluster



About this cluster

Status: Active (auto-suspend in 2880 minutes)

ID: ses_QSpdDJDjX3pu4Xz93iD9Sb7p

Created by: yinhaonan55+200@gmail.com

Created at: Jul 18, 2022, 2:13:41 PM

Access: Only admins and you can view and edit

Project: Ray-Tutorial

Resource usage

CPU: 0 utilized / 8 running

Object store memory: 0 B utilized / 6.87 GiB running

GPU: -

Cost since last start: $0.80

Cost since creation: $0.80

Configuration

Cluster environment: ray_tutorial_app_config_allentest200:1

Compute config: autogenerated-config-2022-07-18T14:13:41.023925

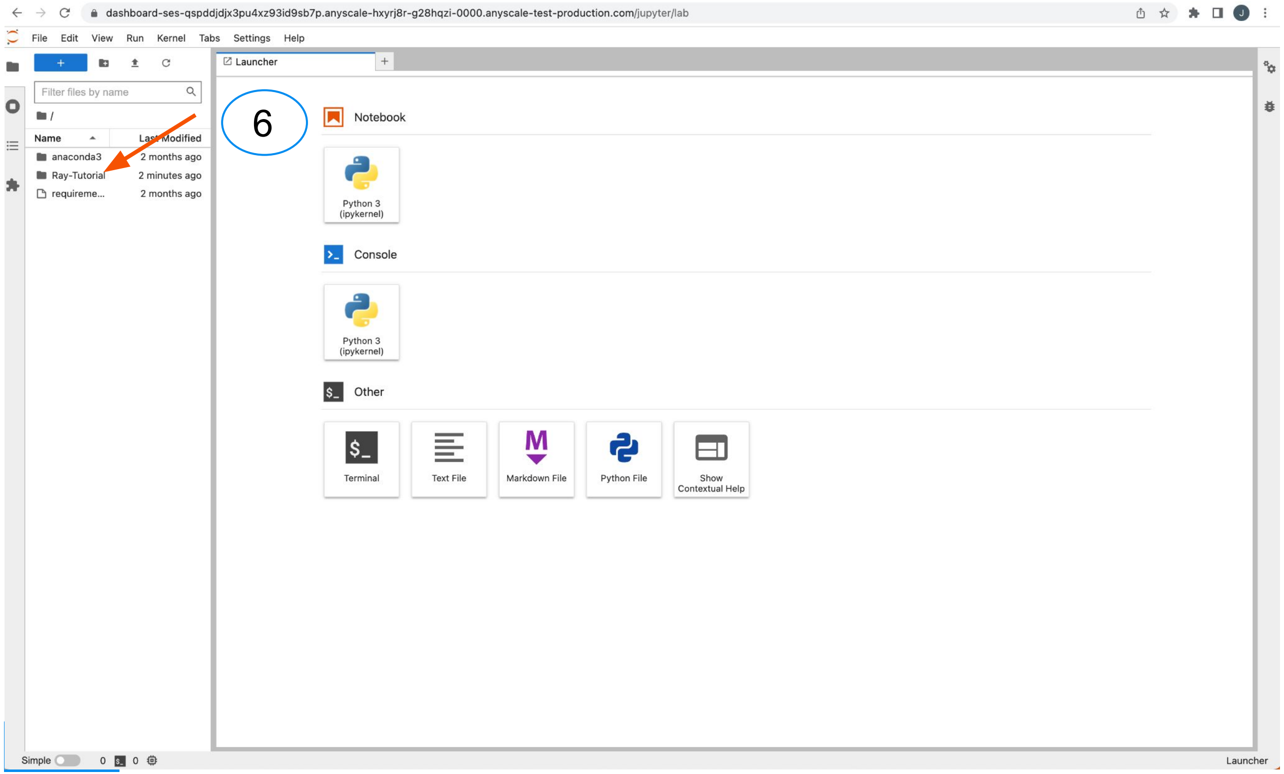
Cloud: ray-summit-test-cloud (gcp, us-central1)

Network access: Public with auth token

Terminal

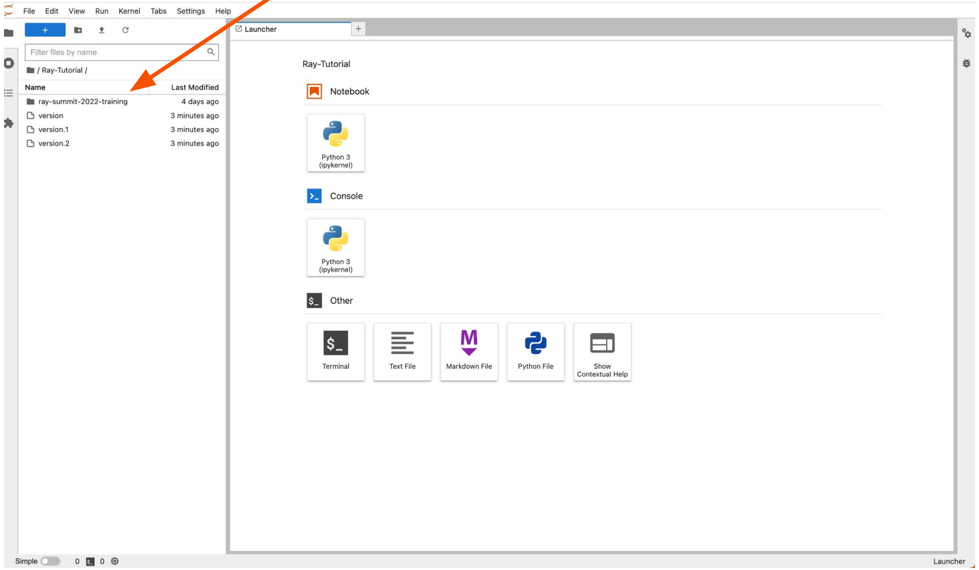
```
exec env ZDOTDIR=/tmp zsh
(base) ray:~%
```

Your Anyscale Cluster



Your Anyscale Cluster

7

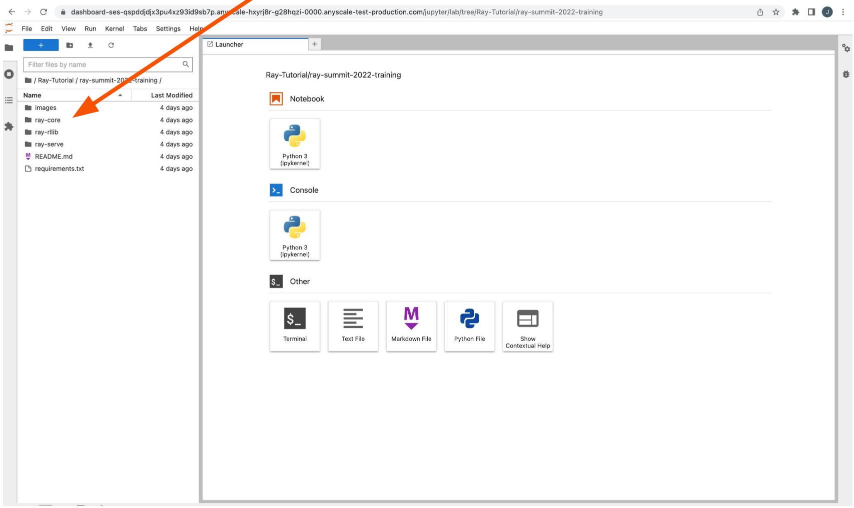


/ Ray-Tutorial /

Name	Last Modified
ray-summit-2022-training	4 days ago
version	3 minutes ago
version.1	3 minutes ago
version.2	3 minutes ago

Ray-Tutorial

8



dashboard-ses-qspddjdjx3pu4xz93id9sb7p.anyscale-hxyrj8r-g28hqzi-0000.anyscale-test-production.com/jupyter/lab/tree/Ray-Tutorial/ray-summit-2022-training

/ Ray-Tutorial / ray-summit-2022-training /

Name	Last Modified
images	4 days ago
ray-core	4 days ago
ray-rllib	4 days ago
ray-serve	4 days ago
README.md	4 days ago
requirements.txt	4 days ago

Ray-Tutorial/ray-summit-2022-training

Thank you.

Tell us what you think…

https://bit.ly/ray-core-summit2022

Tell us what you think…

https://bit.ly/ray-core-summit2022

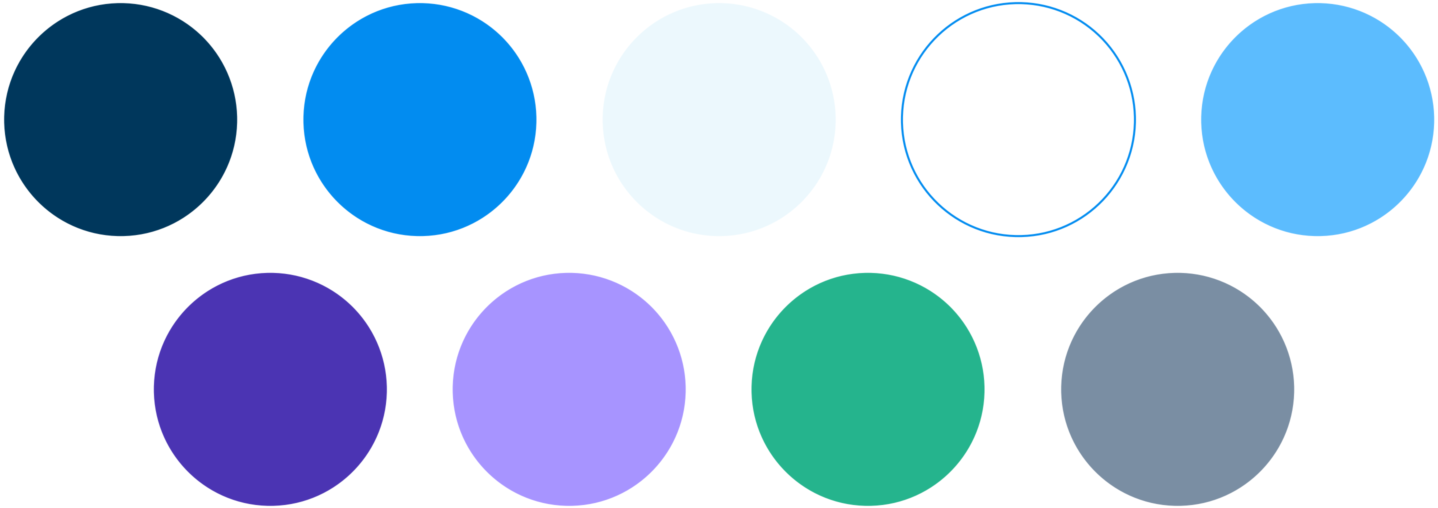
RAY SUMMIT 22

Here is the Title Slide

Firstname Lastname, Company

RAY SUMMIT 22

Colors



Here is a basic Dark Slide

Here is a Basic Light Slide

How about a slide with 2 options?

Here is an info card

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incid idunt ut labo re et dolore magna aliqu Ut enim ad minim veniam, quis nostrud exercitation

Here is an info card

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incid idunt ut labo re et dolore magna aliqu Ut enim ad minim veniam, quis nostrud exercitation

How about a slide with 3?

Here is an info card

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incid idunt ut labo re et dolore magna aliqu Ut enim ad minim veniam, quis nostrud exercitation

Here is an info card

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incid idunt ut labo re et dolore magna aliqu Ut enim ad minim veniam, quis nostrud exercitation

Here is an info card

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incid idunt ut labo re et dolore magna aliqu Ut enim ad minim veniam, quis nostrud exercitation

Here is a Section Header

Here is a Section Header

Here is a Section Header

Thank you.

Follow up information can go here.