UNIVERSITÀ
DEGLI STUDI
DI BERGAMO

Tecnologie Cloud & Mobile

Tecnologie Cloud E Mobile

Big Data and Applications in the Cloud

mauro.pelucchi@gmail.com
Mauro Pelucchi

21059 - Tecnologie cloud e mobile

Cosa vedremo?

- Python & PySpark
- MongoDB
- Aws Glue

Python

Python

Python in 10 minutes…

Python

- Interpreted language: work with an evaluator for language expressions
- Dynamically typed: variables do not have a predefined type
- Rich, built-in collection types:
  - Tuples
  - Dictionaries
  - Sets
- Concise
- Open source general-purpose language
- Object Oriented, Procedural, Functional
- Easy to interface with C/ObjC/Java/Fortran
- Great interactive environment

Features

- Indentation instead of braces
- Several sequence types
  - Strings ‘...’: made of characters, immutable
  - Lists [...]: made of anything, mutable
  - Tuples (...) : made of anything, immutable
- Powerful subscripting (slicing)
- Exceptions as in Java
- Simple object system
- Iterators (like Java)

References

https://github.com/Naviden/Python_Introduction

Naviden / Python_Introduction

Watch 1 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Actions | Projects 0 | Wiki | Security | Insights

Branch: master ▾ | Python_Introduction / Py_intro.ipynb | Find file | Copy path

Naviden Rrformatting sections 7c320ca on Feb 23

1 contributor

19833 lines (19833 sloc) | 731 KB | Raw | Blame | History

About this course

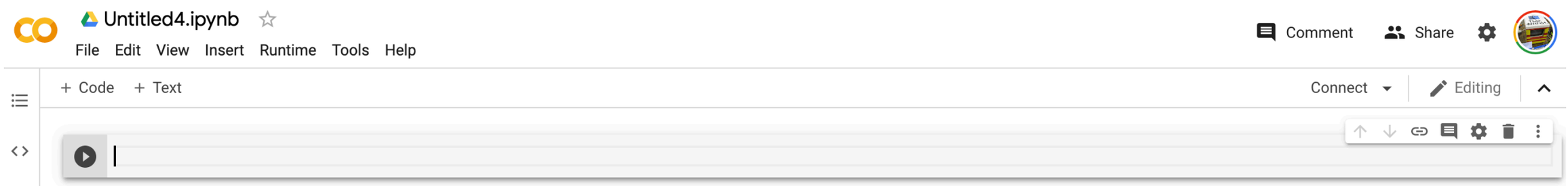
This introduction course, presented for the first time in BIBDA Master, tends to give you an overview of Python programming language and its general structure as in the future modules of Master like web scrapping, Machine learning and text mining Python will be used as the principal programming language. Please note that due to the vast range of arguments/tools related to Python language we won't/can't cover them all here as this course designed to help you not to become a Python programmers but to learn its basics and enable you to expand your data science skills using it as a modern programming language.

Course structure

Our goal here is not to provide a deep insight into Python and act as an official reference since you can easily can access hundreds of online sources to learn about each and every detail of Python. Given the wide range of students' backgrounds, there is no assumption about prior knowledge about programming or data analysis. We've tried to provide you with simple examples which give you a clear understanding of functionality of different tools. There are also parts called **YOUR TURN** which ask you to put into practice what you have learned during the lesson. These parts are accompanied by three different symbols which show the difficulty of the exercise:

References

- The Python Tutorial (http://docs.python.org/tutorial/)
- PEP 8- Style Guide for Python Code
  - http://www.python.org/dev/peps/pep-0008/
- Google Colab



Which Python?

- **Python 3!!!!!**

Python 2.7 will retire in...

0	0	0	0	0	0
Years	Months	Days	Hours	Minutes	Seconds

Enable Guido Mode Huh?

What's all this, then?

Python 2.7 will not be maintained past 2020. Originally, there was no official date. Recently, that date has been updated to January 1, 2020. This clock has been updated accordingly. My original idea was to throw a Python 2 Celebration of Life party at PyCon 2020, to celebrate everything Python 2 did for us. That idea still stands. (If this sounds interesting to you, email pythonclockorg@gmail.com).

Python 2, thank you for your years of faithful service.

Python 3, your time is now.

How do I get started?

If the code you care about is still on Python 2, that's totally understandable. Most of PyPI's popular packages now work on Python 2 and 3, and more are being added every day. Additionally, a number of critical Python projects have pledged to stop supporting Python 2 soon. To ease the transition, the official porting guide has advice for running Python 2 code in Python 3.

If you're here, you'd probably enjoy the full python release schedule (a glitch maintained by Dustin Ingram)

A Code Sample

Open Google Colab

https://colab.research.google.com/drive/1eoOuTMfnPC3oVJLykRwn388WMp4k6i67#scrollTo=y-EyvlQa2fTA

```
[1]  x = 5
     y = 6
     z = x + y
     print(z)
```

```
11
```

```
if z > 10:
  print("z > 10")
else:
  print("z <= 10")
```

```
z > 10
```

A Code sample

- Assignment uses = and comparison uses ==
- For numbers + - * / % are as expected
- Special use of + for string concatenation
- Special use of % for string formatting

- The basic printing command is print(…).
- The first assignment to a variable creates it
- Variable types don’t need to be declared
  - Python figures out the variable types on its own

Strings

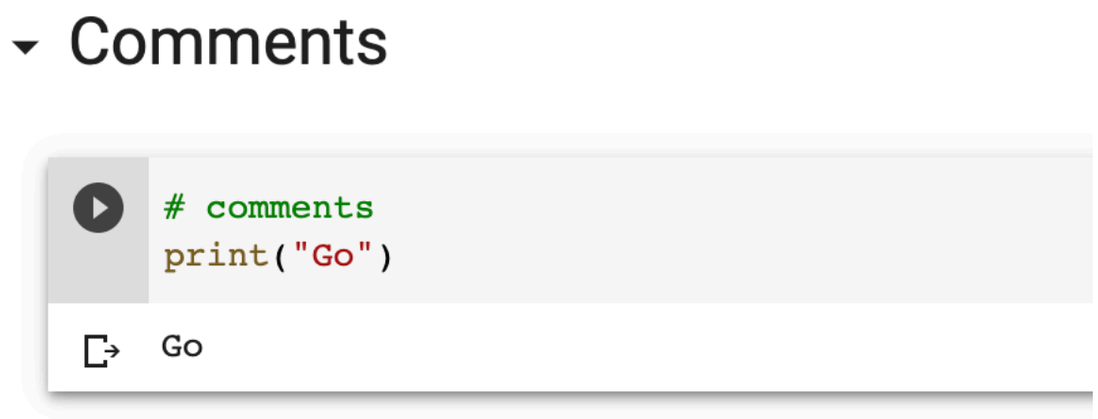
Strings

```
hello = 'Hello, '
name = "Unibg"
print(f"{hello} {name}!!!")
```

```
Hello,  Unibg!!!
```

Comments

- Start comments with # – the rest of line is ignored.



Libraries

- numpy
  - Offers Matlab-ish capabilities within Python
  - Fast array operations
  - 2D arrays, multi-D arrays, linear algebra etc.
- matplotlib
  - High quality plotting library
- pandas
  - Data processing
- sklearn
  - Machine Learning and …
- TensorFlow/keras
  - Deep Learning

Control Flows

```
if x == 3:
    print "X equals 3."
elif x == 2:
    print "X equals 2."
else:
    print "X equals something else."
print "This is outside the 'if'."
```

```
assert(number_of_players < 5)
```

```
x = 3
while x < 10:
    if x > 7:
        x += 2
        continue
    x = x + 1
    print "Still in the loop."
    if x == 8:
        break
print "Outside of the loop."
```

```
for x in range(10):
    if x > 7:
        x += 2
        continue
    x = x + 1
    print "Still in the loop."
    if x == 8:
        break
print "Outside of the loop."
```

Control Flows

Control Flows

```
[11] my_list = ['a', 'b', 'c']
     for item in my_list:
       print(item)
```

```
a
b
c
```

```
for c in range(0, 10):
  print(c)
```

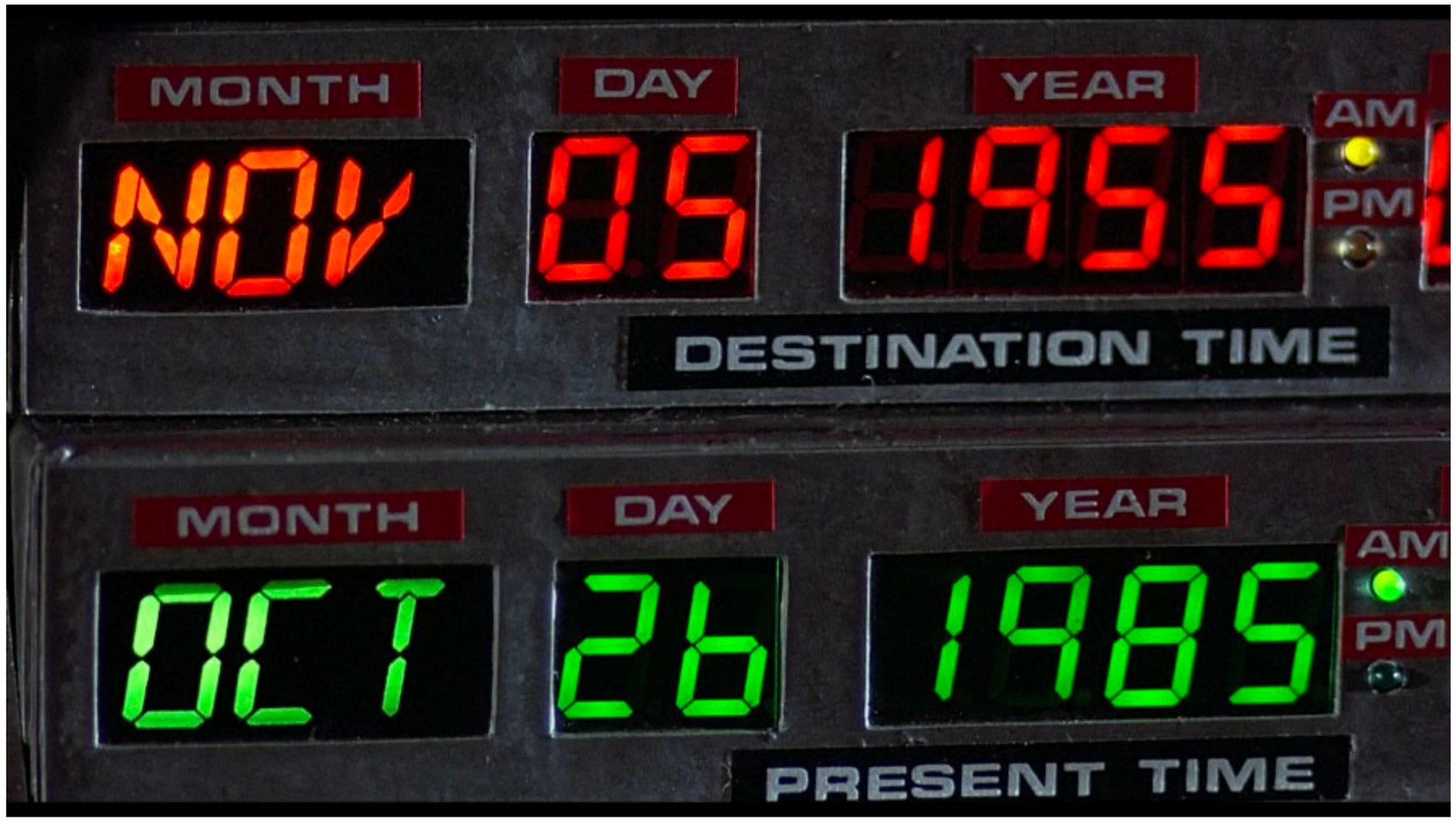
```
0
1
2
3
4
5
6
7
8
9
```

Assignment

- In Google Colab
  - Calcolate in Python in fattoriale di un numero
    - Esempio 5!=5*4*3*2*1=120

Assignment

Ci vendiamo tra 15 minuti…



Solutions

Factorial

```
[9]  n=int(input('Digit a number: '))
     factorial=1

     for i in range(1,n+1):
         factorial = factorial*i

     print(factorial)
```

```
Digit a number: 10
3628800
```

```
n=int(input('Digit a number: '))

def factorial(n):
    if n==0:
        return 1
    else:
        return n*factorial(n-1)

print ('Factorial: ', factorial(n))
```

```
Digit a number: 10
Factorial:  3628800
```

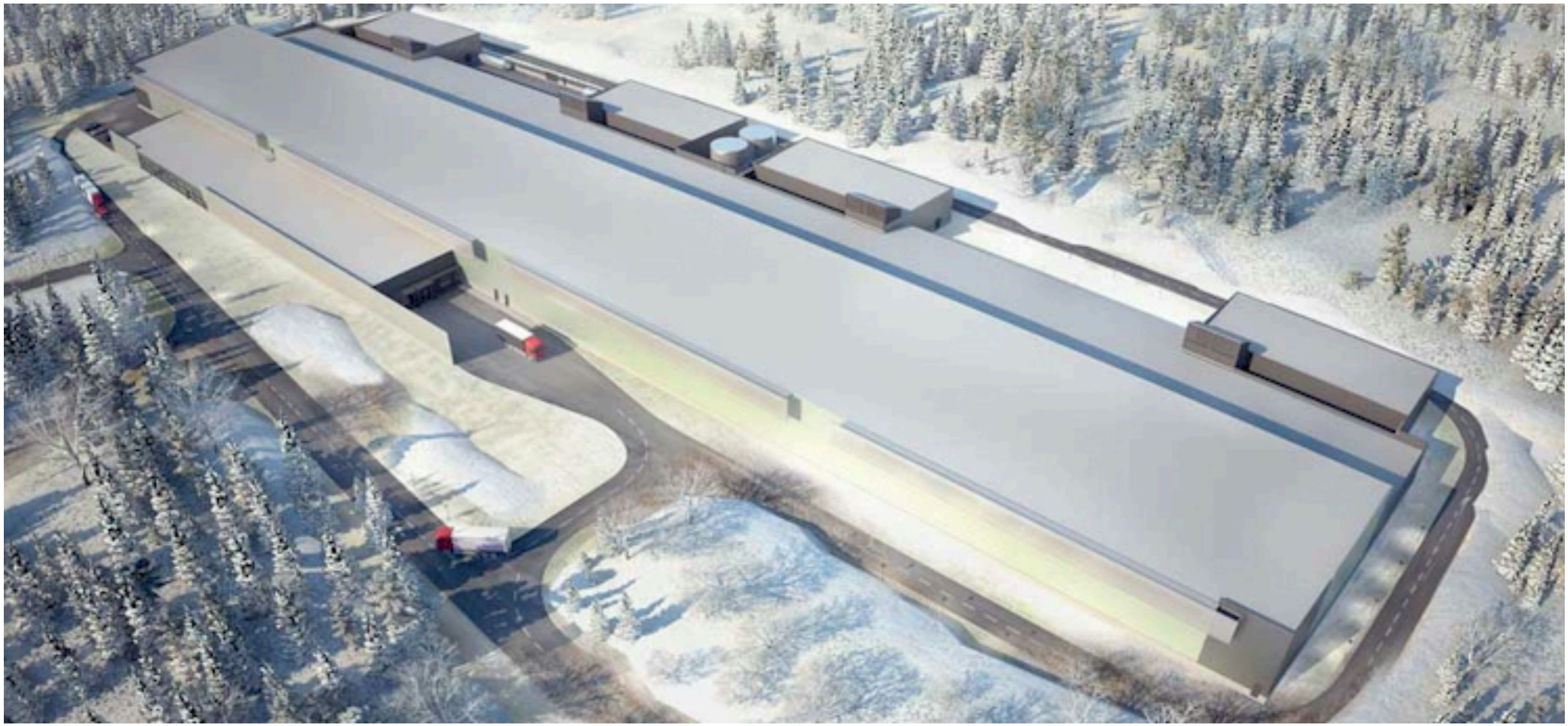
PySpark

Data is huge

- Banks, city councils, governments, shops, etc.
- Facebook processes > 750TB/day of data
- > 48k iPhones every day
- > 7PB of photo storage / month

This requires computers (a lot of them!)

Facebook datacenter



Scale Up vs Scale Down

- Scale-up
  - Increasing the power of your computer (i.e, disk, memory, processor)
- Scale-out
  - Use many standard computers and **distribute** data and computation over them

Data is fast

- Twitter fire hose
  - In 2011, 1.000 Tweets per second (TPS)
  - In 2014, 20.000 TPS
    - With peaks: 143K TPS
- Services on top
  - Not only internet companies!
  - Stock exchange, sensors in water network, smart-cities, fitness trackers, etc.

Perché Spark e Hadoop?

Challenges

Grossi volumi di dati

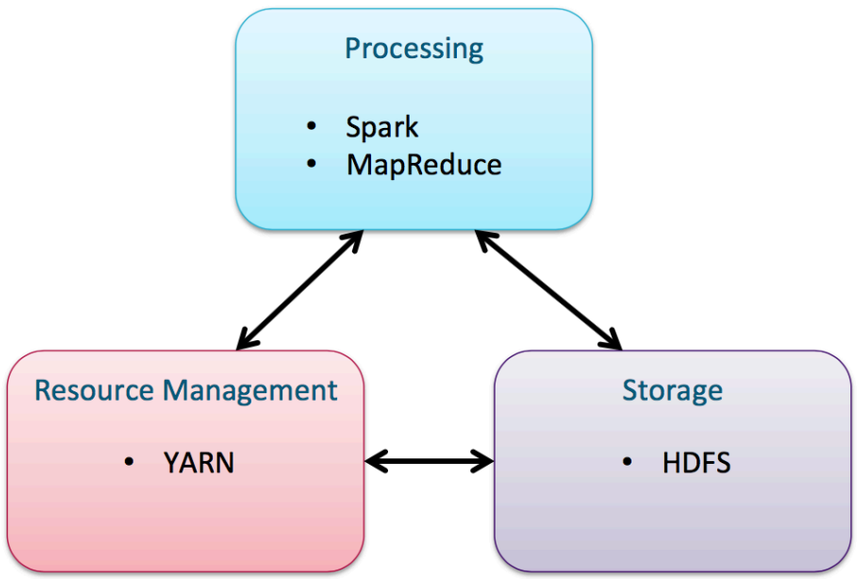
Dati non strutturati

Complessità dei sistemi distribuiti

Applicazioni complesse (ELT, Text mining, modelli predittivi, …)

Hadoop is a framework for distributed storage and processing

Perché Spark e Hadoop?



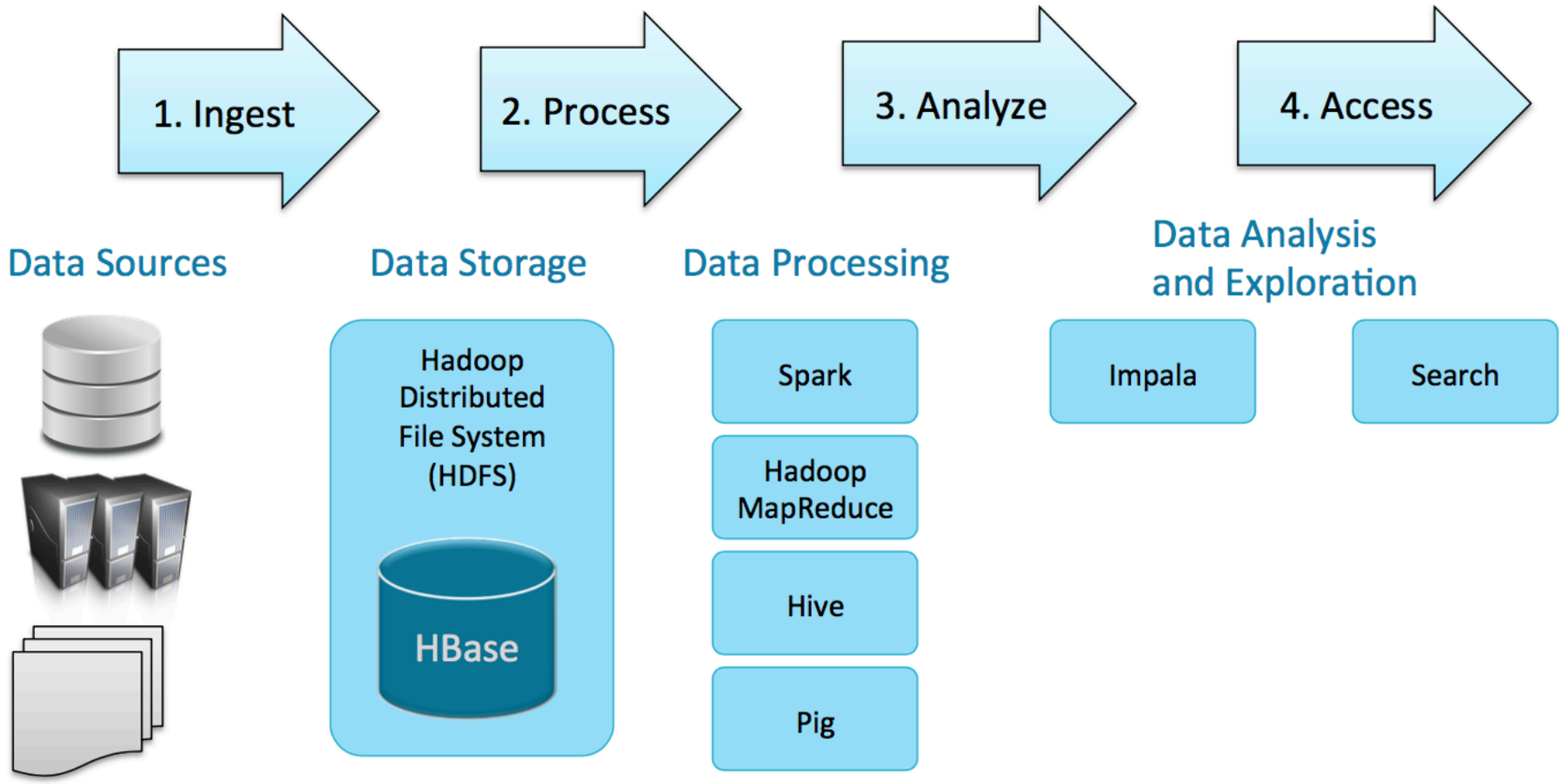
Hadoop: Bring the program to the data rather than the data to the program

- Data storage
- Scalable
- Distributed
- Fault tolerant

Spark is large-scale data processing engine

- General purpose
- Batch Processing
- Machine learning
- Business intelligence
- Streaming

cloudera

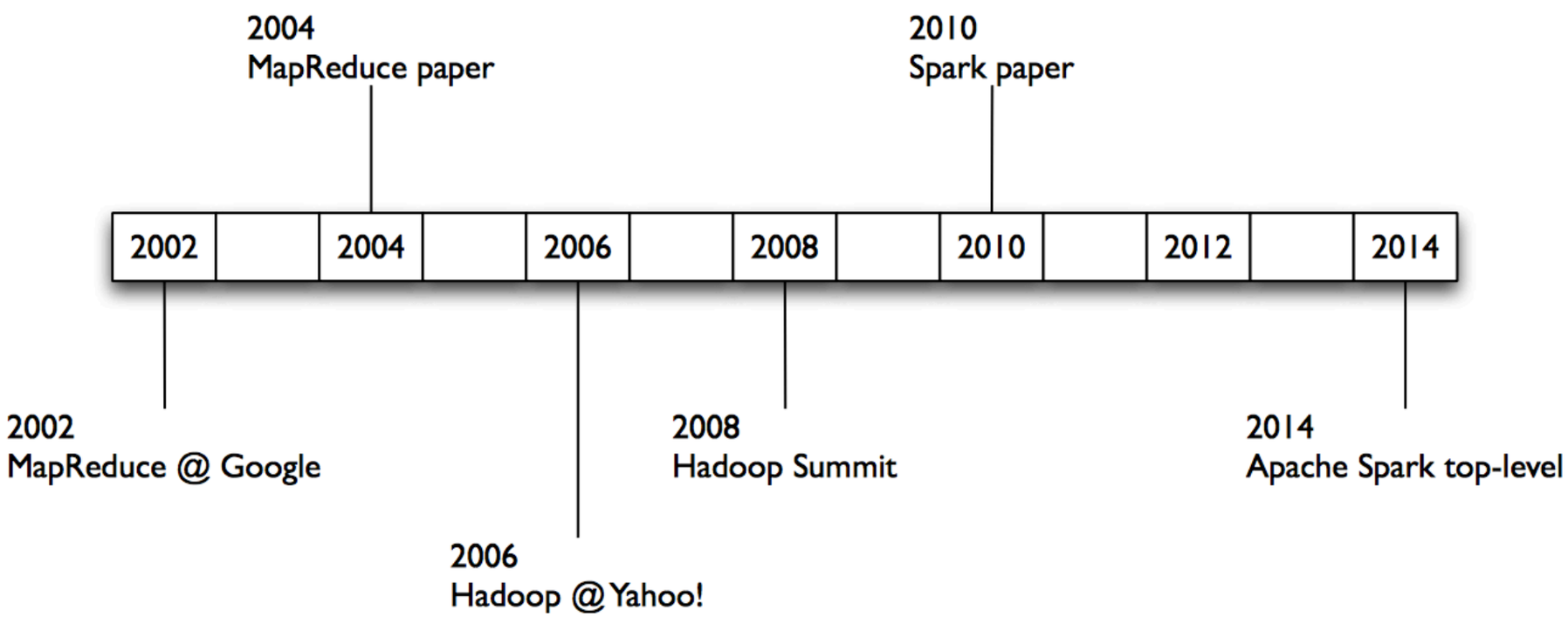


Spark

Apache Spark is an open-source framework for processing **huge volumes of data** (big data) with speed and simplicity.

Spark: Cluster Computing with Working Sets

Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, Ion Stoica
University of California, Berkeley



Matei Zaharia

https://www.usenix.org/legacy/event/hotcloud10/tech/full_papers/Zaharia.pdf

Spark

- Sviluppato a UC Berkeley (2010), estende il modello MapReduce per supportare in modo efficace più tipi di operazioni, query interattive e stream processing
  - Scritto in **Scala**
  - Compatibile con **Hadoop**
  - Api per **Java**, **Scala**, **Python**
- Semplice, funzionale —> modello di programmazione ad alto livello
- **Veloce**
- **In memory caching, query optimizations**
- **General purpose**, ideale per **ETL** e **Machine Learning**
- Ideale per dati non strutturati

Spark Vs Hadoop

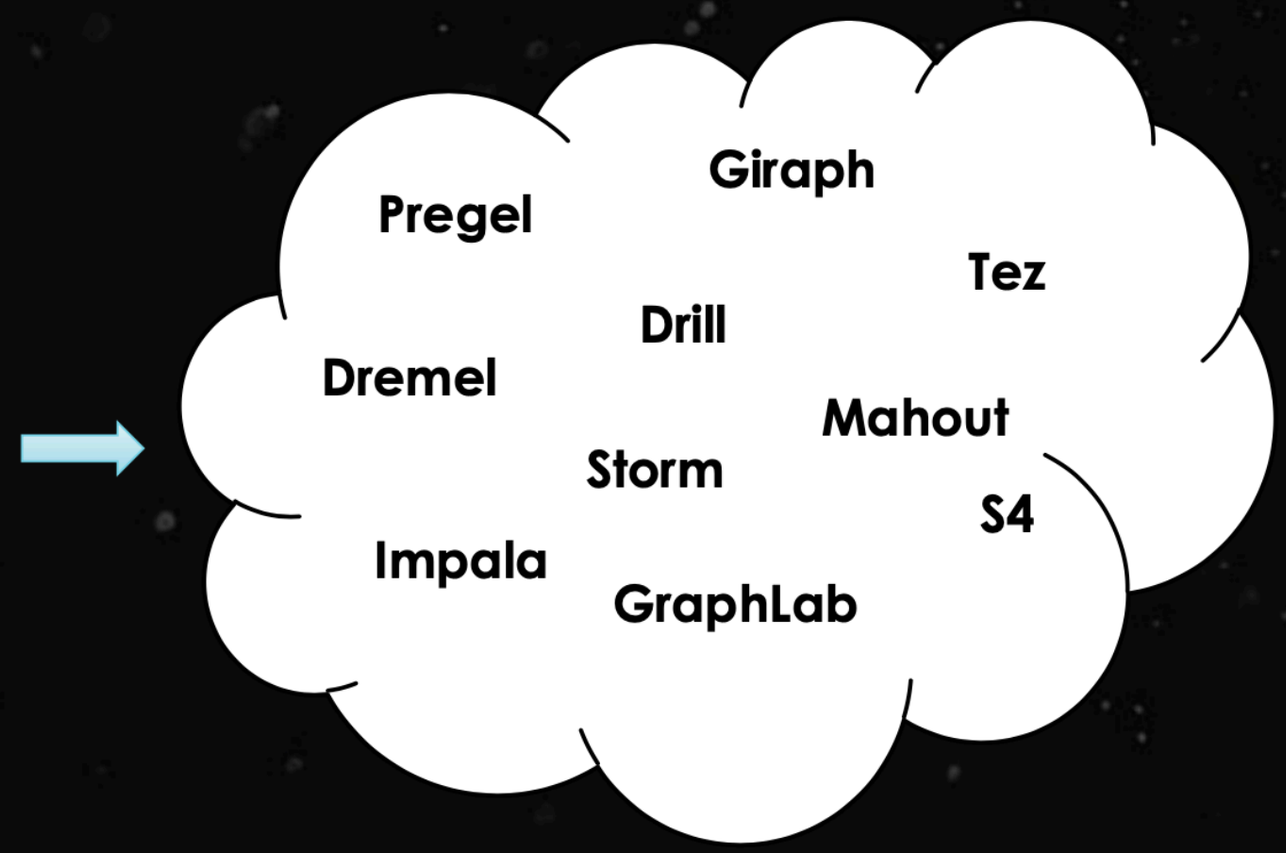
Spark supporta molte tipologie di tasks come interrogazioni **SQL**, applicazioni **streaming**, **machine learning** e **graph operations**.

Hadoop MR è ottimo per lavori pesanti, specifici e complessi su quantità massicce di dati. In ogni caso, molto spesso Spark è una soluzione ottima anche in questi casi perché ottimizza l'uso della memoria e offre un modello di programmazione ad alto livello (semplice).

(2004 – 2013)

(2007 – 2015?)

(2014 – ?)



General Batch Processing

Specialized Systems

(iterative, interactive, ML, streaming, graph, SQL, etc)

General Unified Engine

Why Spark?

Spark has become the default processing engine for a myriad of engineering & science problems

Spark Vs Hadoop (MR)

- Generalized patterns —> stesso framework per molti casi d’uso
- Lazy evaluation —> reduces wait states, better pipelining
- Functional programming —> più facile scrivere applicazioni complesse, applicazioni più facili da mantenere

Spark - Lo stack Spark

Spark SQL
structured data

Spark Streaming
real-time

MLlib
Machine Learning

GraphX
Graph processing

Spark Core

Standalone Scheduler

Hadoop / YARN

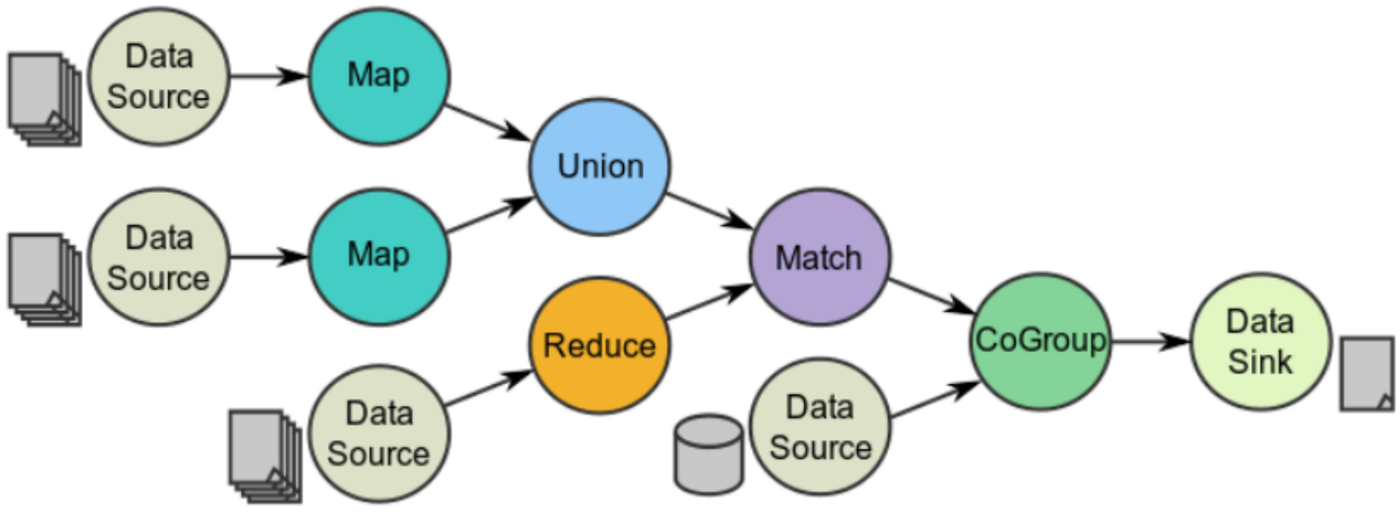
Mesos

Spark - Programming model

- Spark Programming Model si basa su operatori che possono *andare in parallelo (parallelizable operators)*

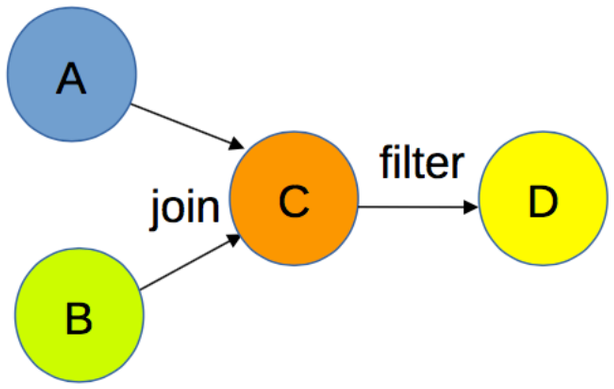
- Data flow —> un data flow è composto da sorgenti, operatori e nodi di dati, collegati da link di input e output

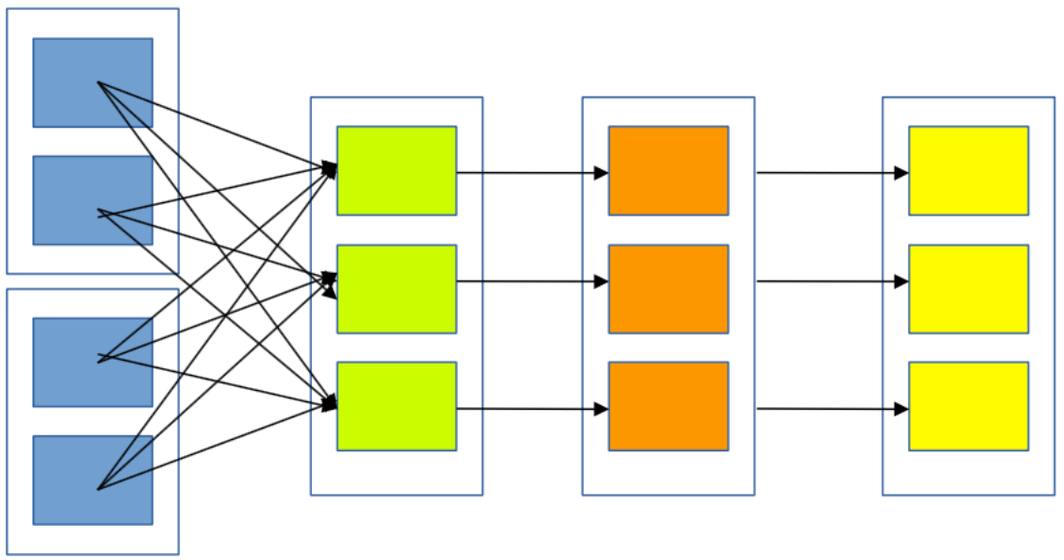
- Un JOB è descritto da un grafo aciclico diretto (directed acyclic graphs o DAG)



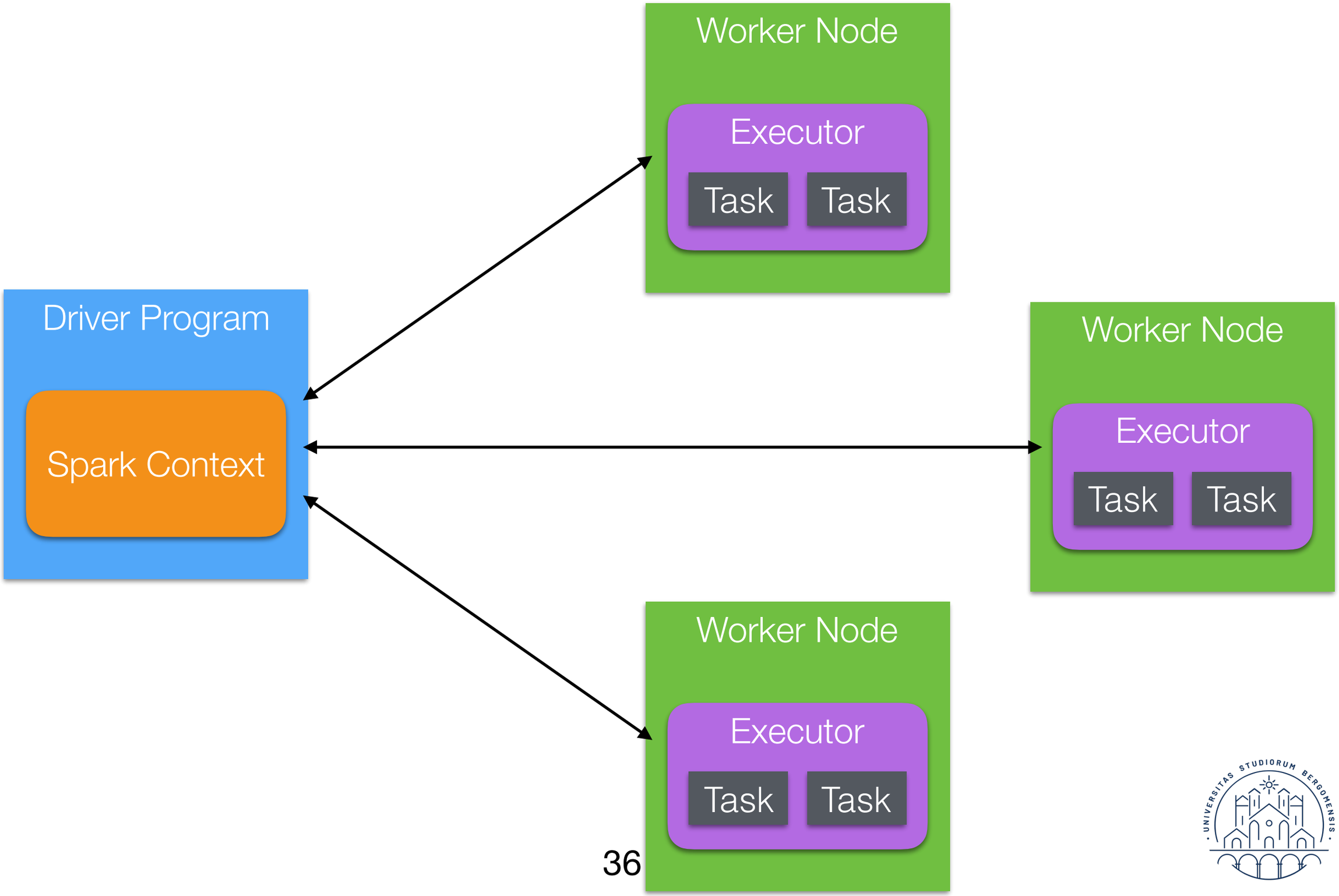
Spark - DAG

- Directed acyclic graphs o DAG), regole del DAG SCHEDULER
  - divide il grafo in una serie di stage di task (insiemi di task)
  - sottomette gli stage ai nodi del cluster

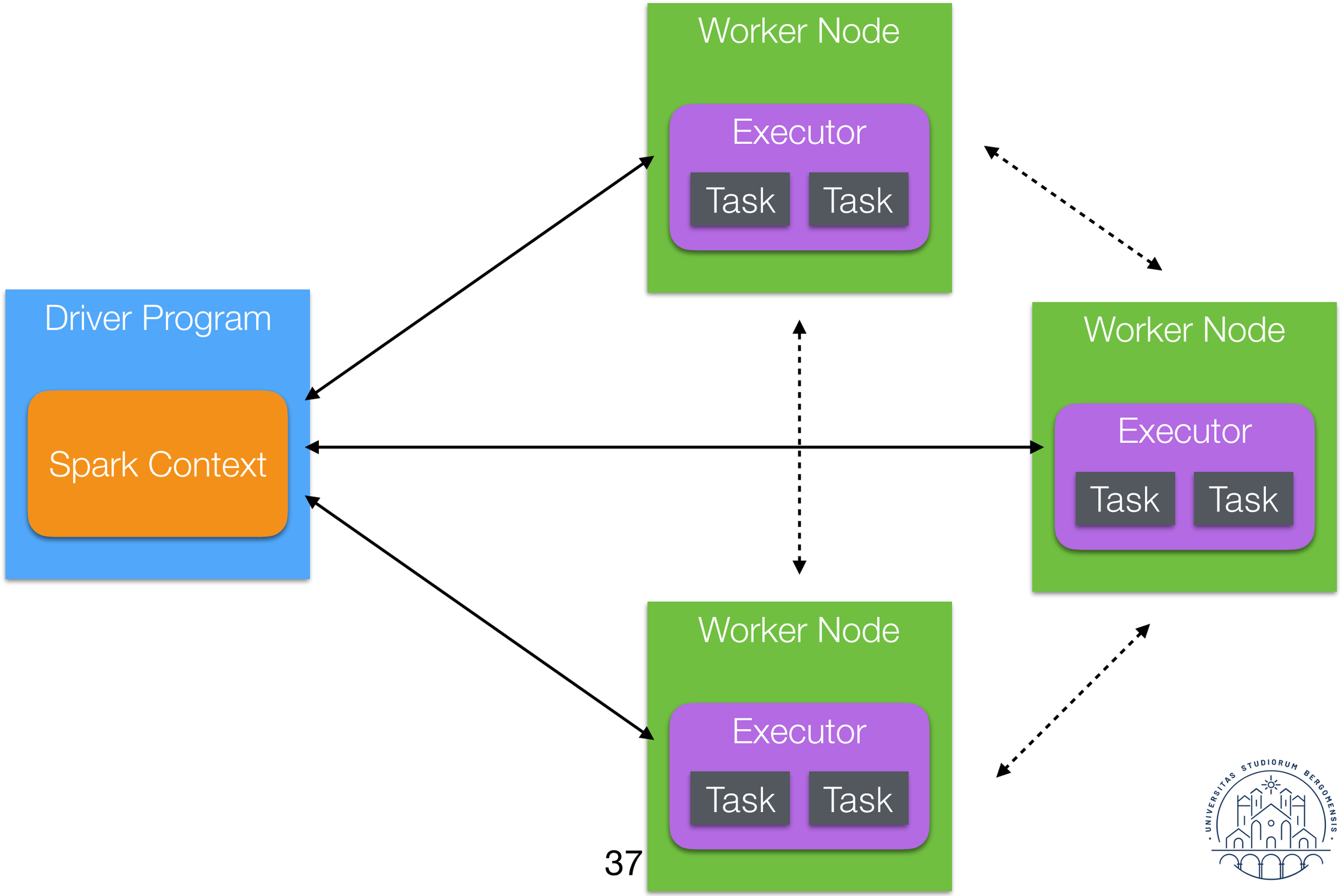




Spark - Come funziona?



Spark - Come funziona?



Spark - Come funziona?

1. L'utente esegue un job attraverso con la funzione **spark-submit**

2. Spark avvia il **driver program** e invoca il metodo **main** specificato dall'utente

3. Il driver program contatta il manager del cluster per chiedere le risorse per avviare gli **executors**

4. Il cluster manager lancia gli **executors** richiesti dal driver program

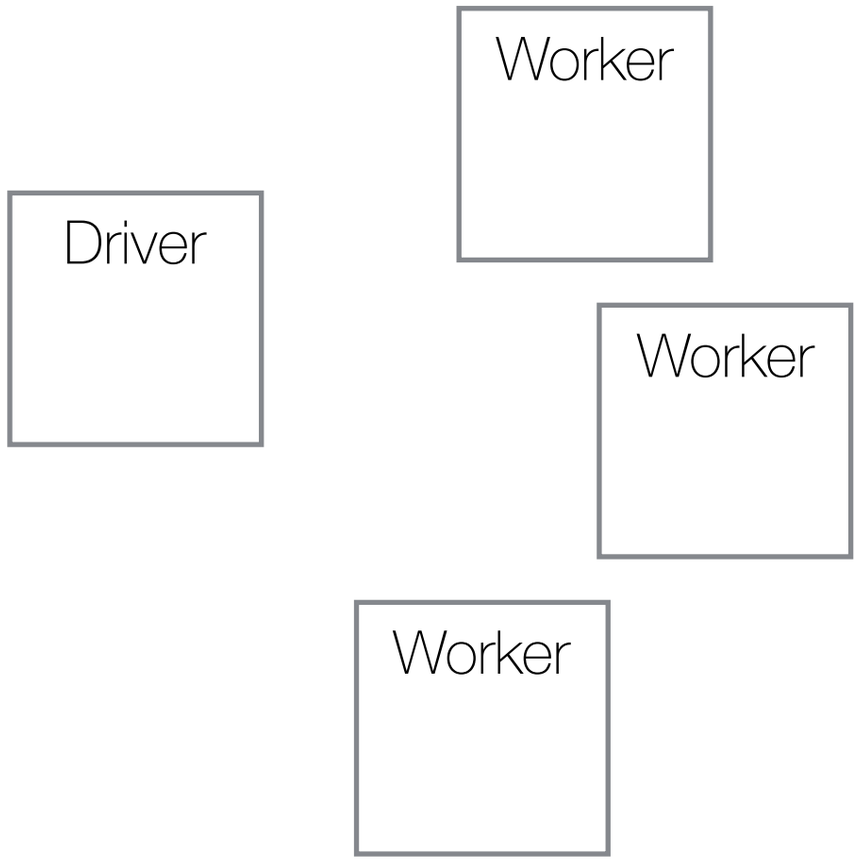
5. Il driver, in base alle azioni e alle trasformazioni richieste dal job dell'utente, manda il lavoro agli **executors** in forma di **tasks**

6. I tasks sono eseguiti dai processi executor ed il risultato viene salvato

7. Al termine dell'esecuzione vengono rilasciate le risorse

Spark - Esempio

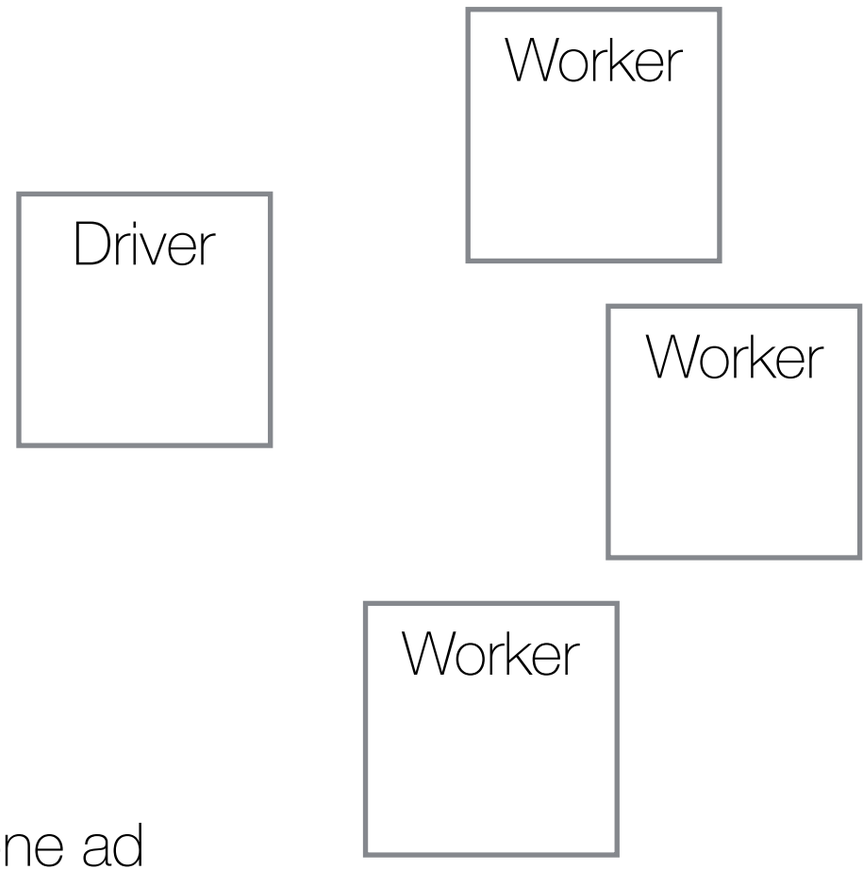
Analizziamo un file di log cercando gli errori relativi a MongoDB



Spark - Esempio

Carichiamo il file di log da hdfs e prepariamolo in **cache**

```
// leggiamo le righe dal file di log

val lines = sc.textFile(“hdfs://.../var/log/*“)

// applichiamo un filtro ed una trasformazione

val errors = lines.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))

// mettiamo in cache i messaggi

messages.cache()
```

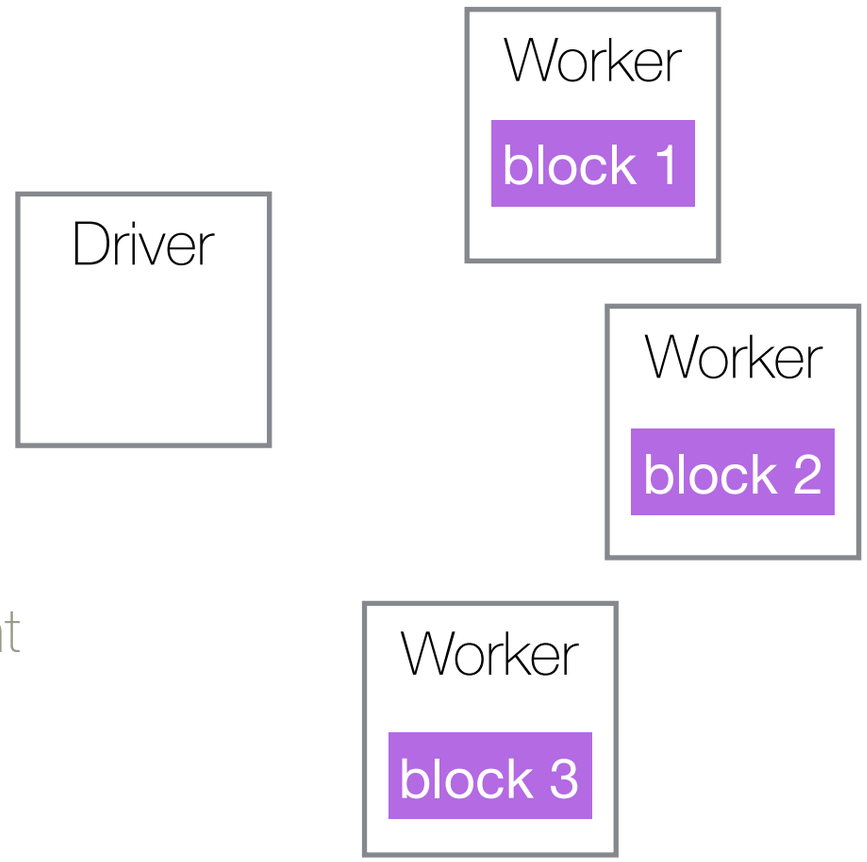


- Spark divide il dataset in 3 partizioni ed assegna ogni partizione ad un nodo
- Non avviene nulla: solo quando richiediamo un’azione sui dati Spark esegue le modifiche

Spark - Esempio

Contiamo le righe che contengono il termine mongodb

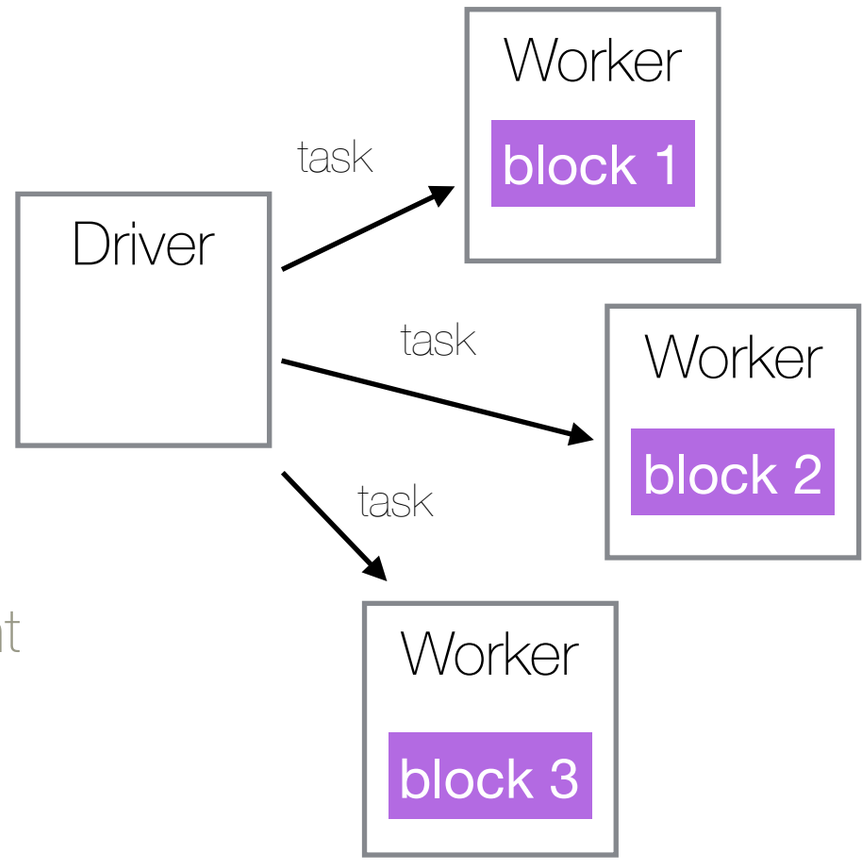
```
// leggiamo le righe dal file di log

val lines = sc.textFile(“hdfs://.../var/log/*“)

// applichiamo un filtro ed una trasformazione

val errors = lines.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))

// mettiamo in cache i messaggi

messages.cache()

// contiamo le righe di errore per MongoDB con l’azione count

messages.filter(_.contains("mongodb")).count()
```



Spark - Esempio

Contiamo le righe che contengono il termine mongodb

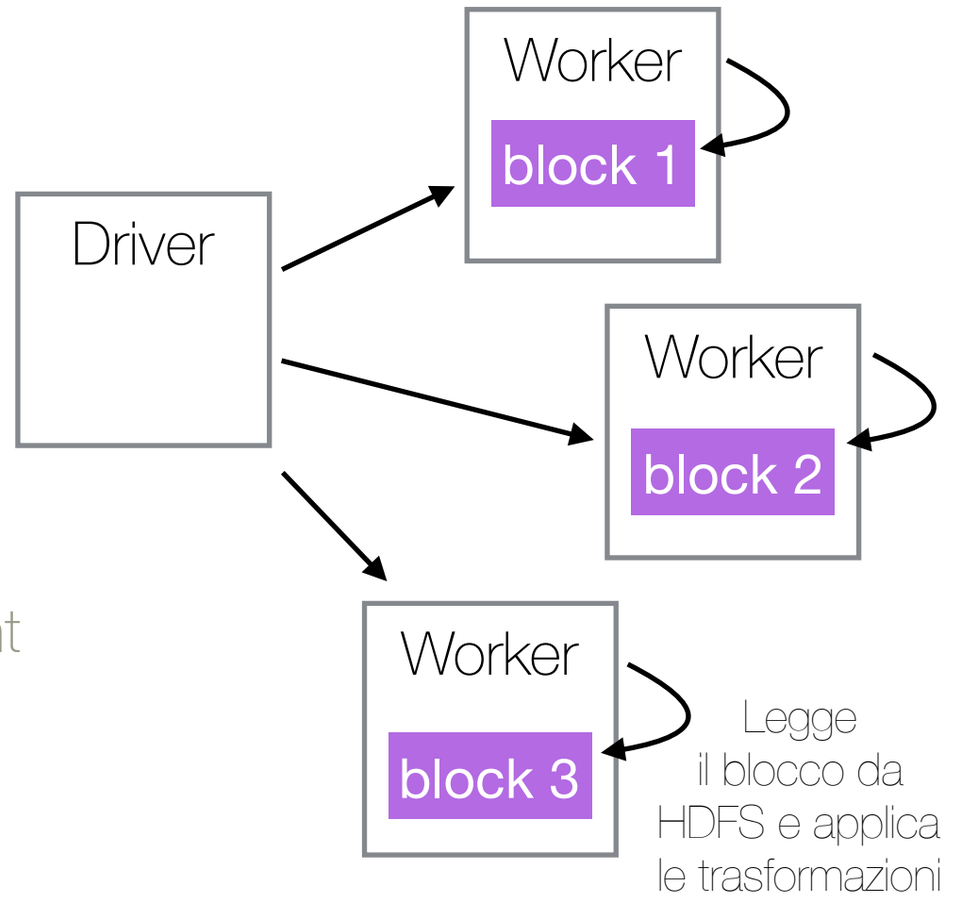
```
// leggiamo le righe dal file di log

val lines = sc.textFile(“hdfs://.../var/log/*“)

// applichiamo un filtro ed una trasformazione

val errors = lines.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))

// mettiamo in cache i messaggi

messages.cache()

// contiamo le righe di errore per MongoDB con l’azione count

messages.filter(_.contains("mongodb")).count()
```



Spark - Esempio

Contiamo le righe che contengono il termine mongodb

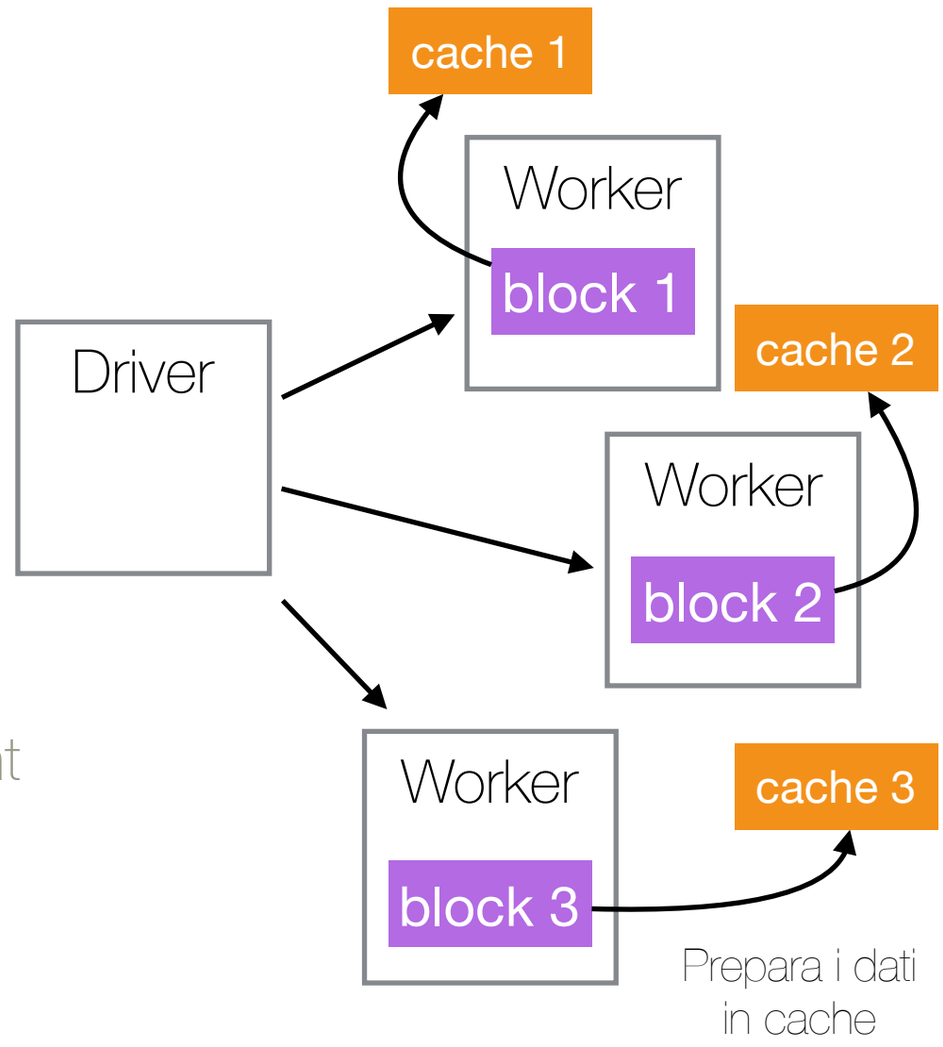
```
// leggiamo le righe dal file di log

val lines = sc.textFile(“hdfs://.../var/log/*“)

// applichiamo un filtro ed una trasformazione

val errors = lines.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))

// mettiamo in cache i messaggi

messages.cache()

// contiamo le righe di errore per MongoDB con l’azione count

messages.filter(_.contains("mongodb")).count()
```



Spark - Esempio

Contiamo le righe che contengono il termine mongodb

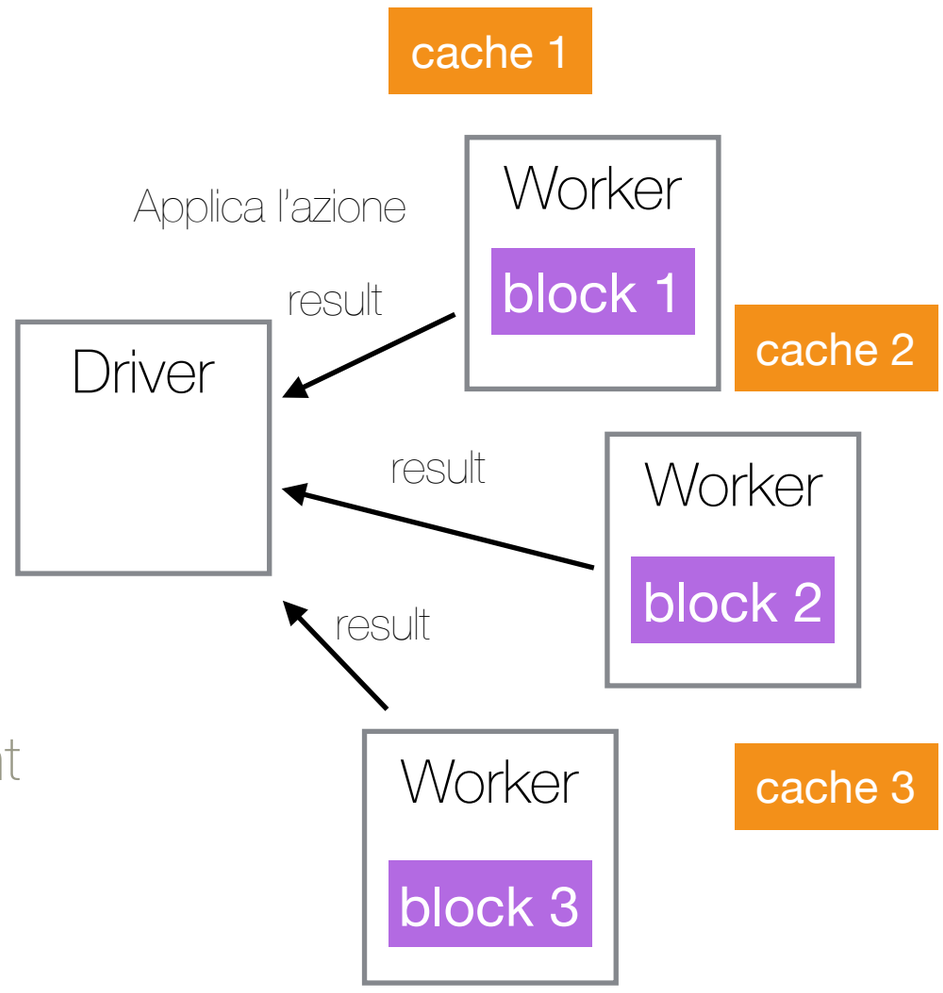
```
// leggiamo le righe dal file di log

val lines = sc.textFile(“hdfs://.../var/log/*“)

// applichiamo un filtro ed una trasformazione

val errors = lines.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))

// mettiamo in cache i messaggi

messages.cache()

// contiamo le righe di errore per MongoDB con l’azione count

messages.filter(_.contains("mongodb")).count()
```



Spark - Esempio

Contiamo le righe che contengono il termine mongodb

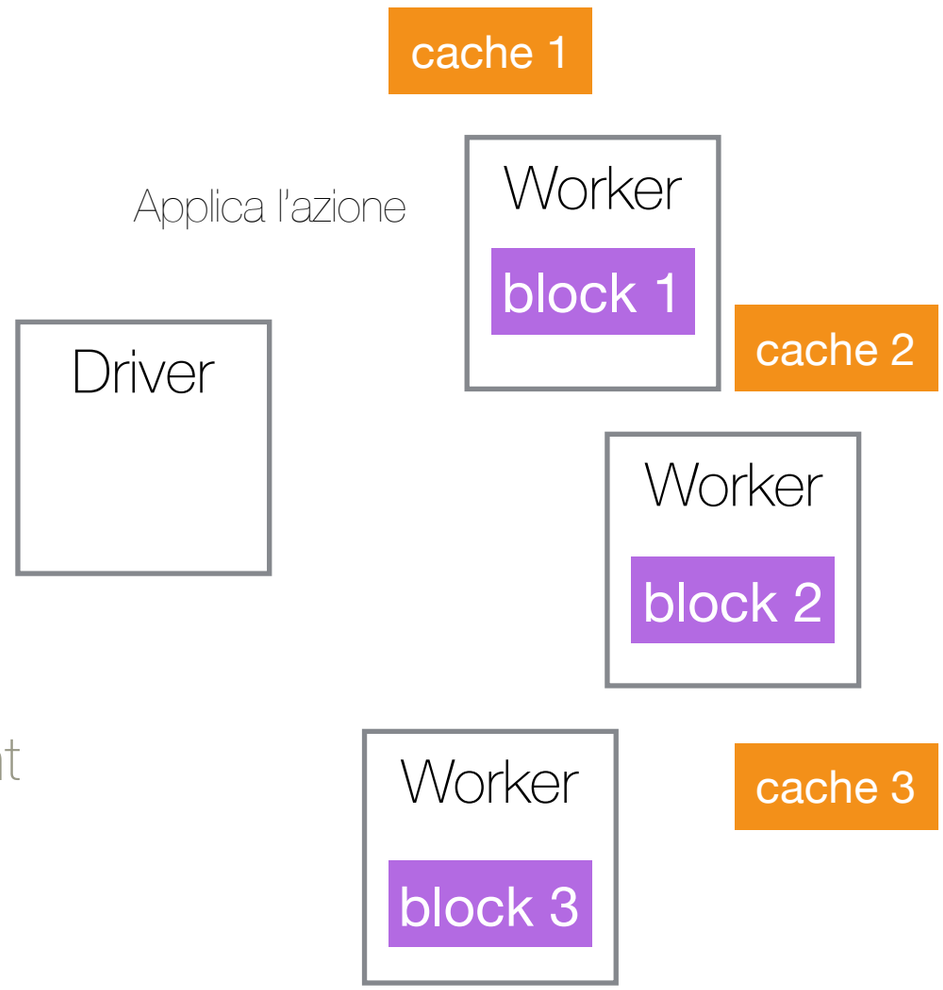
```
// leggiamo le righe dal file di log

val lines = sc.textFile(“hdfs://.../var/log/*“)

// applichiamo un filtro ed una trasformazione

val errors = lines.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))

// mettiamo in cache i messaggi

messages.cache()

// contiamo le righe di errore per MongoDB con l’azione count

messages.filter(_.contains("mongodb")).count()
```



Spark - Esempio

Contiamo le righe che contengono il termine mysql

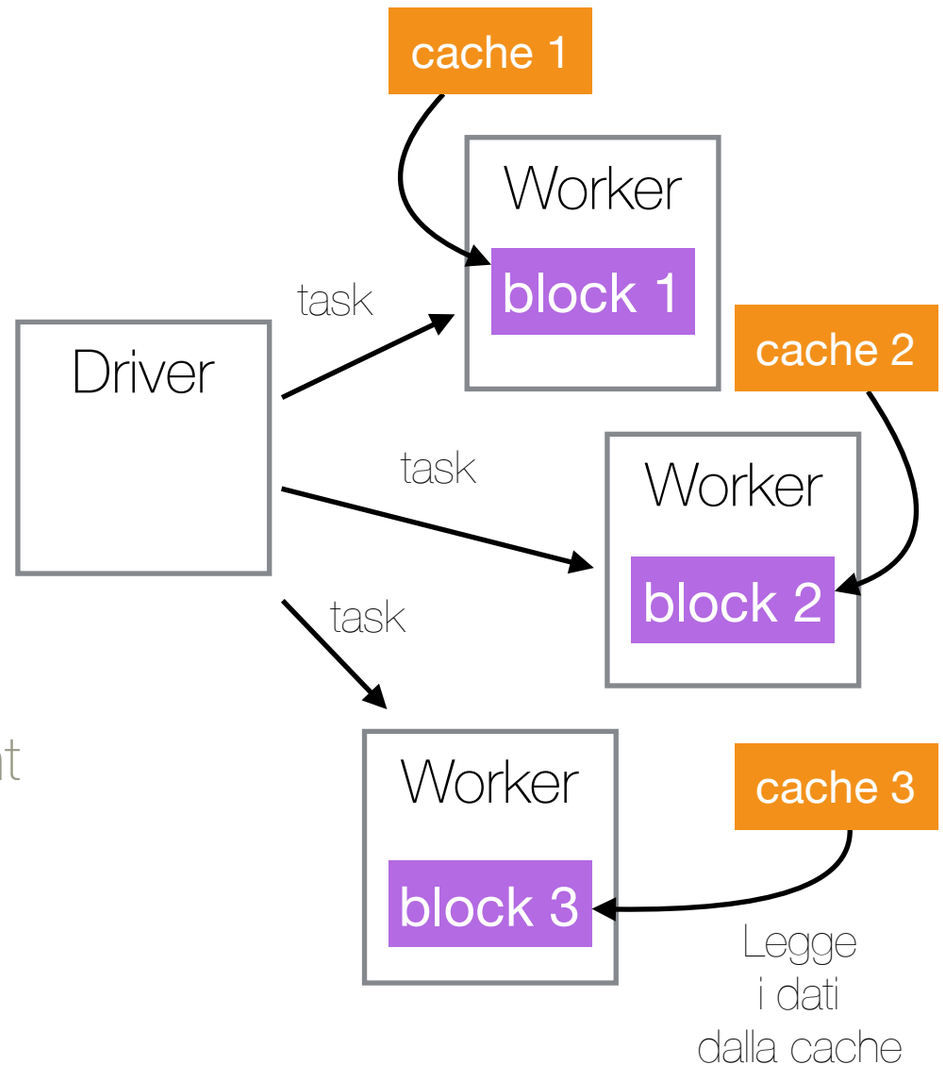
```
// leggiamo le righe dal file di log

val lines = sc.textFile(“hdfs://.../var/log/*“)

// applichiamo un filtro ed una trasformazione

val errors = lines.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))

// mettiamo in cache i messaggi

messages.cache()

// contiamo le righe di errore per MongoDB con l’azione count

messages.filter(_.contains(“mongodb")).count()

// contiamo le righe di errore per Mysql con l’azione count

messages.filter(_.contains("mysql")).count()
```



Spark - Esempio

Contiamo le righe che contengono il termine mysql

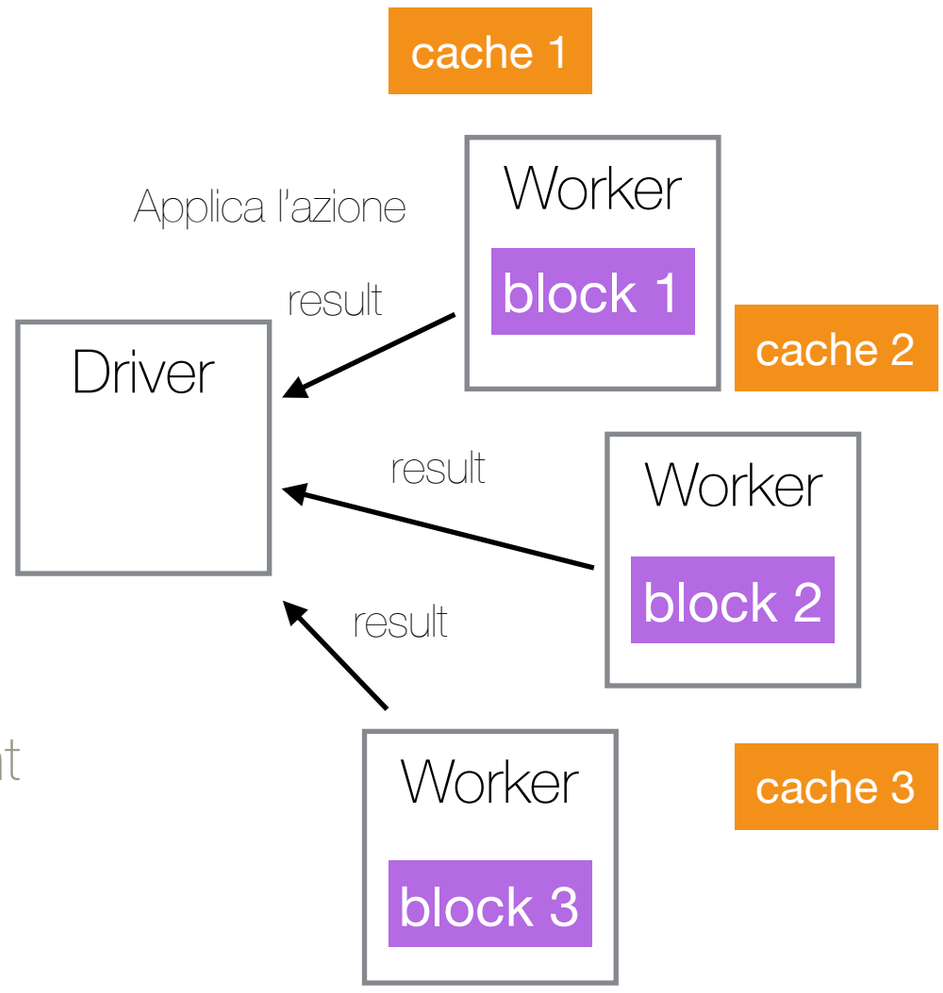
```
// leggiamo le righe dal file di log

val lines = sc.textFile(“hdfs://.../var/log/*“)

// applichiamo un filtro ed una trasformazione

val errors = lines.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))

// mettiamo in cache i messaggi

messages.cache()

// contiamo le righe di errore per MongoDB con l’azione count

messages.filter(_.contains(“mongodb")).count()

// contiamo le righe di errore per Mysql con l’azione count

messages.filter(_.contains("mysql")).count()
```



Spark - Esempio

Contiamo le righe che contengono il termine mysql

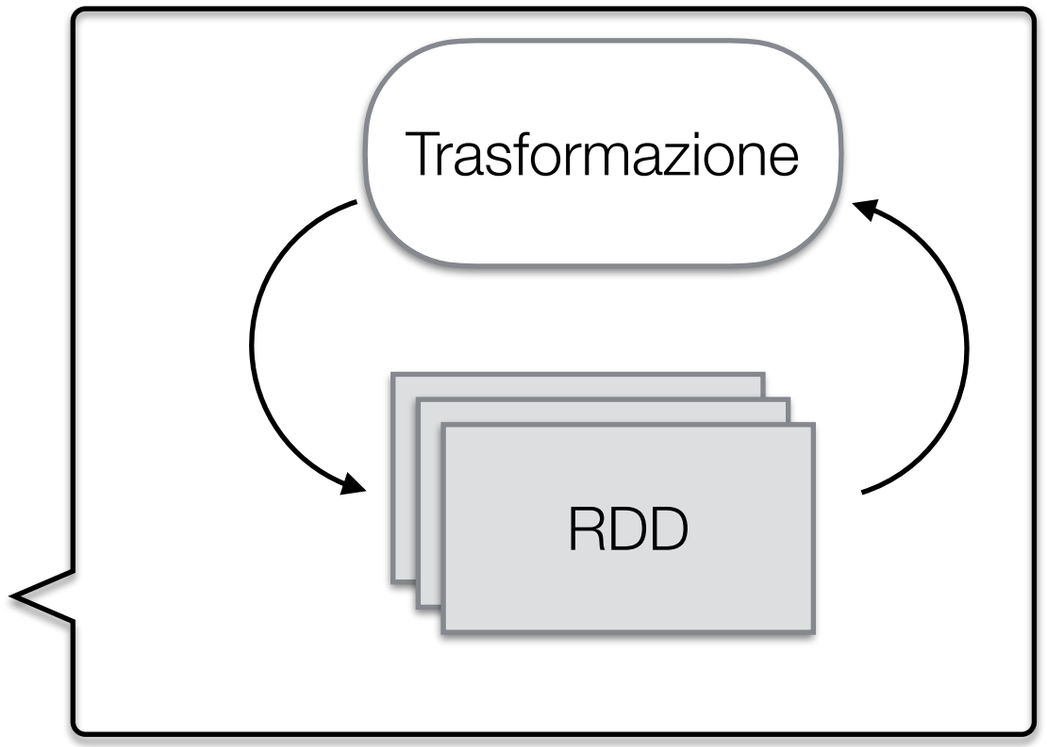
```
// leggiamo le righe dal file di log

val lines = sc.textFile(“hdfs://.../var/log/*“)

// applichiamo un filtro ed una trasformazione

val errors = lines.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))

// mettiamo in cache i messaggi

messages.cache()

// contiamo le righe di errore per MongoDB con l’azione count

messages.filter(_.contains(“mongodb")).count()

// contiamo le righe di errore per Mysql con l’azione count

messages.filter(_.contains("mysql")).count()
```



Spark - RDD

RDD - Resilient Distributed Datasets

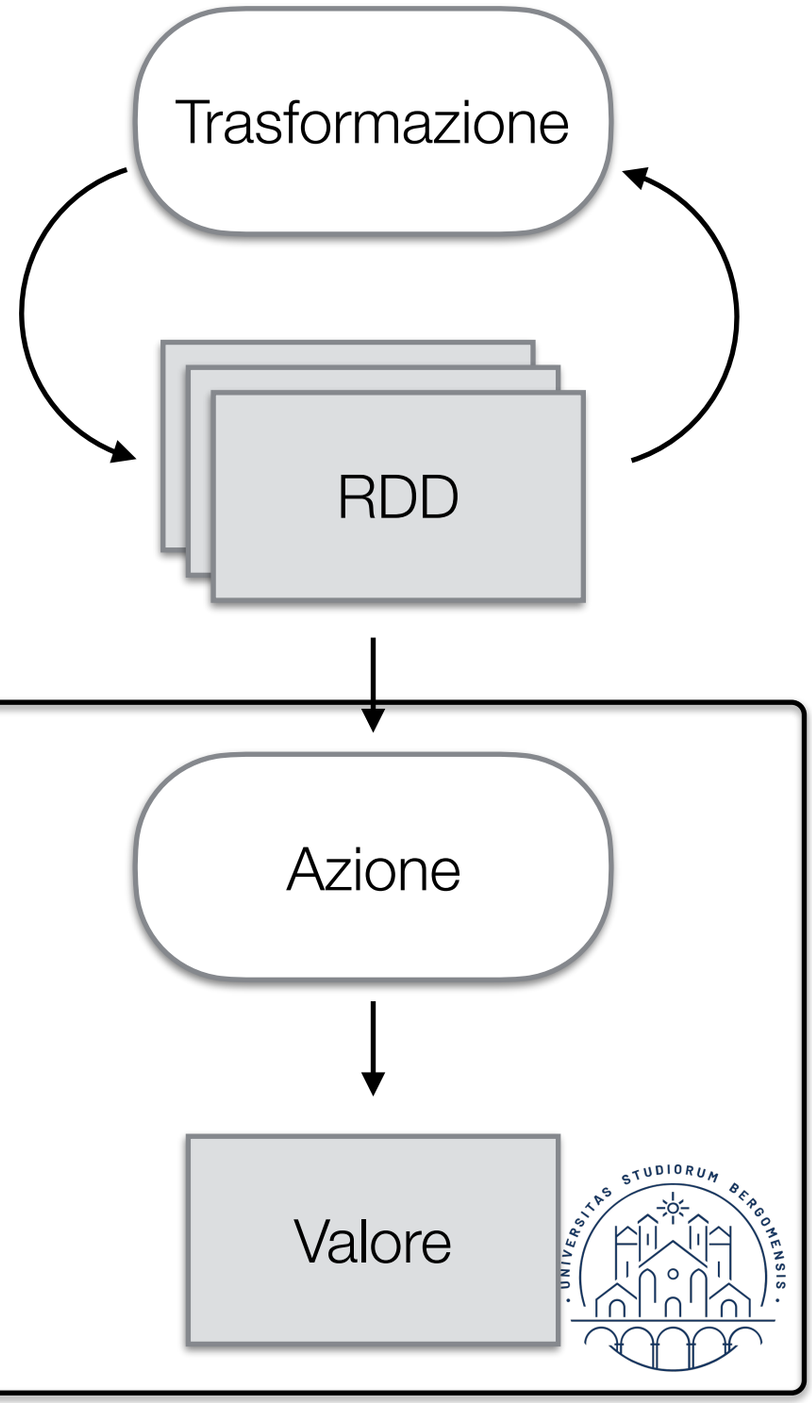
```
// leggiamo le righe dal file di log
val lines = sc.textFile(“hdfs://.../var/log/*“)

// applichiamo un filtro ed una trasformazione
val errors = lines.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))
// mettiamo in cache i messaggi
messages.cache()
// contiamo le righe di errore per MongoDB con l’azione count
messages.filter(_.contains(“mongodb")).count()
// contiamo le righe di errore per Mysql con l’azione count
messages.filter(_.contains("mysql")).count()
```



Spark - RDD

RDD - Resilient Distributed Datasets

```
// leggiamo le righe dal file di log
val lines = sc.textFile(“hdfs://.../var/log/*“)

// applichiamo un filtro ed una trasformazione
val errors = lines.filter(_.startsWith("ERROR"))
val messages = errors.map(_.split("\t")).map(r => r(1))
// mettiamo in cache i messaggi
messages.cache()
// contiamo le righe di errore per MongoDB con l’azione count
messages.filter(_.contains(“mongodb")).count()
// contiamo le righe di errore per Mysql con l’azione count
messages.filter(_.contains("mysql")).count()
```



Spark - SparkContext


```
master@master-2018: ~/Materiale_Master_BIBDA_2018/spark-2.3.0-bin-hadoop2.7
File  Edit  View  Search  Terminal  Help
master@master-2018:~$ cd Materiale_Master_BIBDA_2018/
master@master-2018:~/Materiale_Master_BIBDA_2018$ cd spark-2.3.0-bin-hadoop2.7/
master@master-2018:~/Materiale_Master_BIBDA_2018/spark-2.3.0-bin-hadoop2.7$ ./bin/spark
spark-class   sparkR        spark-shell   spark-sql     spark-submit
master@master-2018:~/Materiale_Master_BIBDA_2018/spark-2.3.0-bin-hadoop2.7$ ./bin/spark-shell
```

```
2018-05-01 15:51:12 WARN  Utils:66 - Your hostname, master-2018 resolves to a loopback address: 127.0.1.1; using 192.168.56.1
03 instead (on interface enp0s3)
2018-05-01 15:51:12 WARN  Utils:66 - Set SPARK_LOCAL_IP if you need to bind to another address
2018-05-01 15:51:15 WARN  NativeCodeLoader:62 - Unable to load native-hadoop library for your platform... using builtin-java
classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Spark context Web UI available at http://192.168.56.103:4040
Spark context available as 'sc' (master = local[*], app id = local-1525182702014).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.3.0
      /_/

Using Scala version 2.11.8 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_161)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

// dalla spark shell verifichiamo

// la presenza dello spark context

scala>sc

res0: org.apache.spark.SparkContext = org.apache.spark.SparkContext@23b087c3

- Ogni applicazione deve inizializzare uno Spark Context
- E' il punto di accesso per le API di Spark
- Contiene tutte le informazioni e la configurazione
  - Accesso al cluster
  - Parametri ed informazioni per la connessione con il data warehouse (Mysql, MongoDB, Elasticsearch, Solr, ….)
  - Parametri dell'applicazione
  - Parametri per Spark

Spark - Cluster Manager

sc.master —> determina quale cluster utilizzare

// verifichiamo come è impostato il master

scala>sc.master

res1: String = local[*]

Master URL	Meaning
`local`	Run Spark locally with one worker thread (i.e. no parallelism at all).
`local[K]`	Run Spark locally with K worker threads (ideally, set this to the number of cores on your machine).
`local[*]`	Run Spark locally with as many worker threads as logical cores on your machine.
`spark://HOST:PORT`	Connect to the given Spark standalone cluster master. The port must be whichever one your master is configured to use, which is 7077 by default.
`mesos://HOST:PORT`	Connect to the given Mesos cluster. The port must be whichever one your is configured to use, which is 5050 by default. Or, for a Mesos cluster using ZooKeeper, use `mesos://zk://...`. To submit with `--deploy-mode cluster`, the HOST:PORT should be configured to connect to the MesosClusterDispatcher.
`yarn`	Connect to a YARN cluster in `client` or `cluster` mode depending on the value of `--deploy-mode`. The cluster location will be found based on the `HADOOP_CONF_DIR` or `YARN_CONF_DIR` variable.

Spark - RDD

RDD - Resilient Distributed Datasets

Un RDD è una raccolta di elementi che possono essere utilizzati in **parallelo**

Resilient: possono essere ricreati
Distributed: vengono elaborati in tutto il cluster
Dataset: contengono dati provenienti da un file, da un data warehouse, …

Un RDD è una collezione distribuita immutabile di oggetti: gli elementi di un RDD sono partizionati su un insieme di macchine.
Se una partizione viene persa può sempre essere ricostruita.

Spark - Dataframe

- Limiti e problemi degli RDD
  - Non possono essere ottimizzati da Spark
  - Risultano lenti in Python e Java
  - E’ molto facile costruire una catena di trasformazioni non efficienti

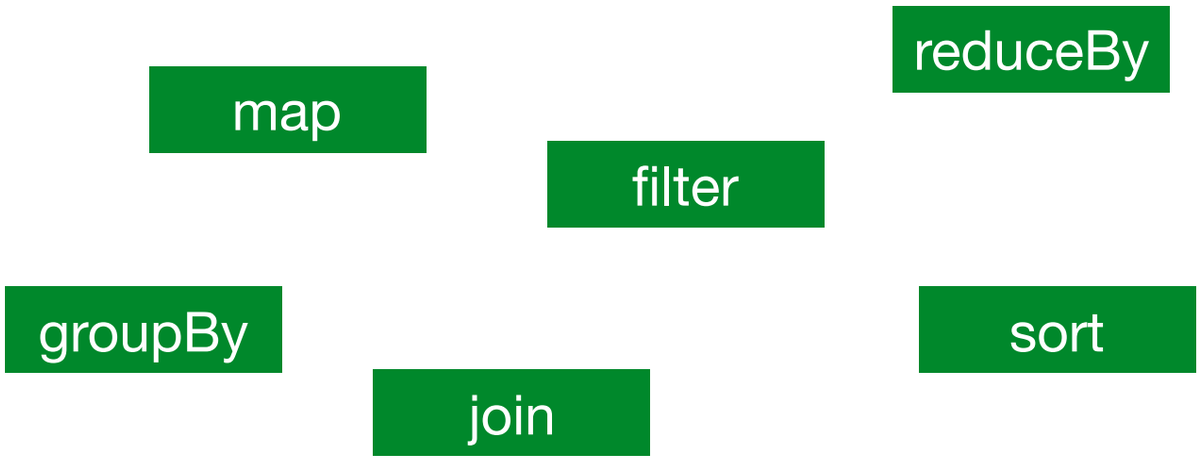
- **Dataframe** e **Dataset** superano questi problemi
- Le API **DataFrame** forniscono un livello di astrazione superiore: possiamo manipolare dati attraverso un linguaggio per query, anche SQL
  - Le query su DataFrame sono ottimizzate da Spark
  - Non sono type safety

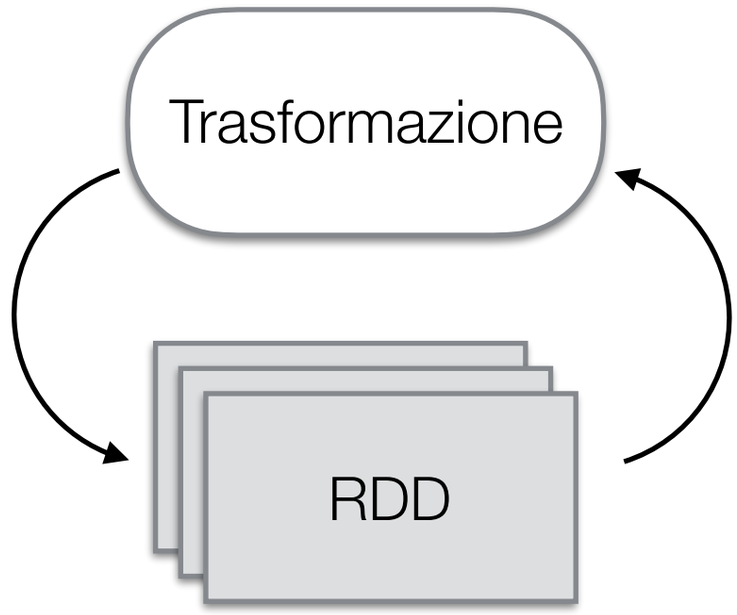
Spark - Dataset

- **Dataset**
  - Simili concettualmente a un RDD
  - Usano il framework Tungsten (encoding in-memory): migliori performance in termini di spazio utilizzato e velocità
  - Più facili dei Dataframe
  - Disponibili con Spark 2.*

Spark - Trasformazioni

Le trasformazioni creano un nuovo dataset da uno esistente

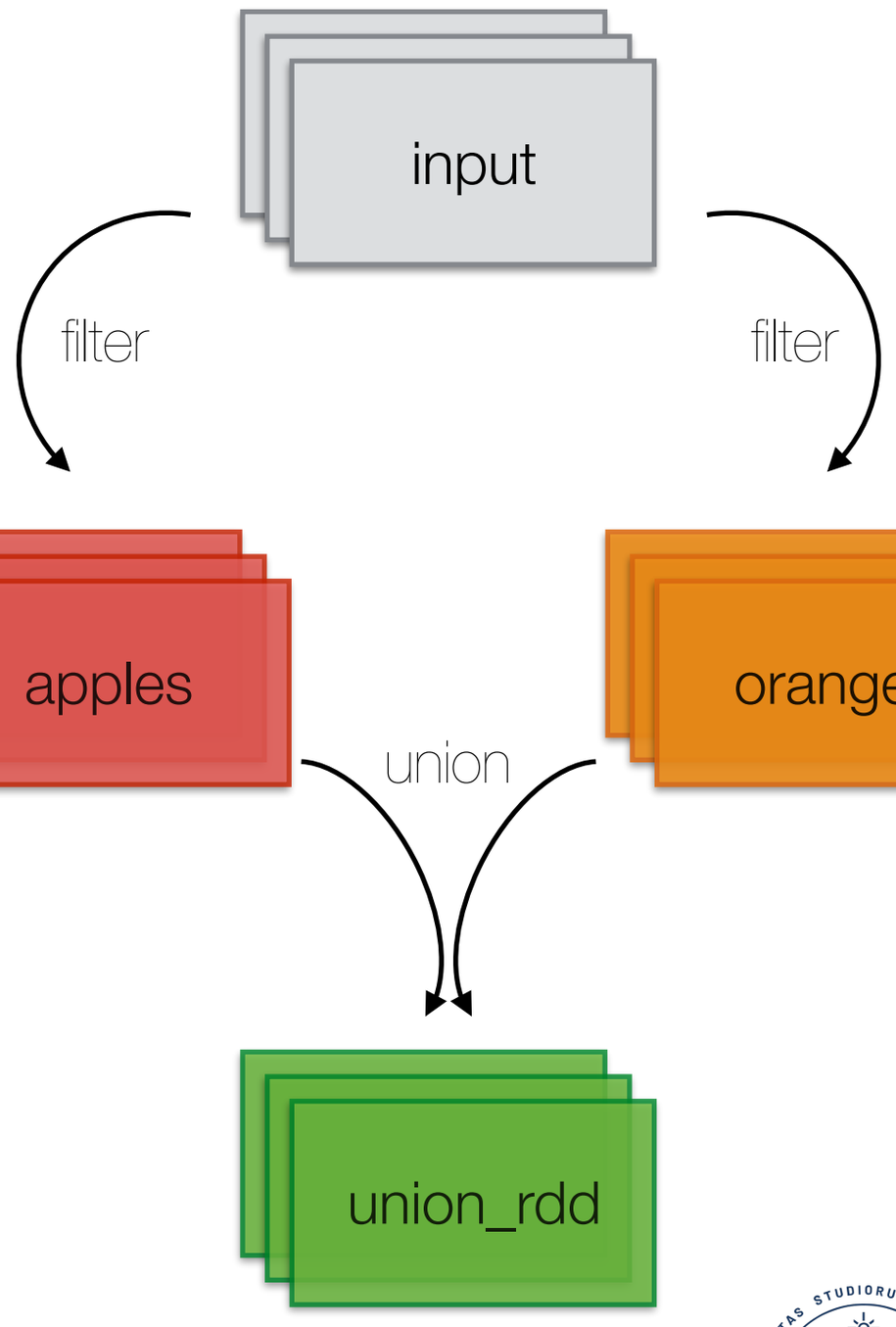




Tutte le trasformazioni sono lazy!!!
Non elaborano subito i loro risultati, in questo modo ottimizzano i calcoli e se necessario possono recuperare le partizioni di dati perse

Spark - Trasformazioni

```
// leggiamo le righe dal file di log
val input = sc.textFile(“myfile.txt")
// cerco le linee che contengono “apple”
val apples = input.filter(x => x.contains(“apple"))
// cerco le linee che contengono “orange”
val oranges = input.filter(x => x.contains("orange"))
// unisco i due dataset
val union_rdd = apples.union(oranges)
```



Spark - Azioni

Un azione indica che qualcosa deve essere effettuato

collect

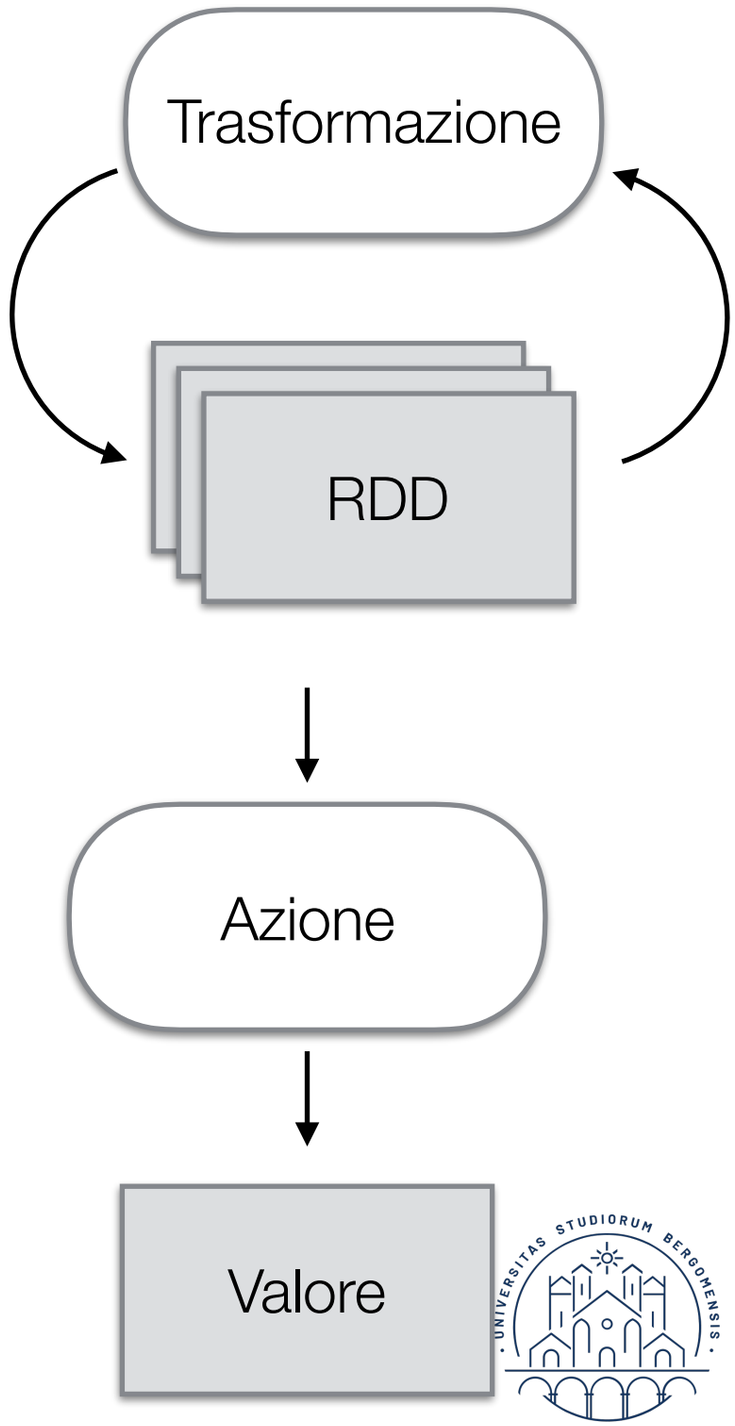
reduce

count

saveAsTextFile

take

first



Tutte le azioni vengono eseguite subito da Spark e producono un risultato.

Spark - Lazy Evaluation

Tutte le trasformazioni sono pigre… perché?

Spark - Lazy Evaluation

- Big Picture: se guardiamo dall'alto possiamo vedere più alternative
- Non eseguiamo operazioni non necessarie

Esempio:

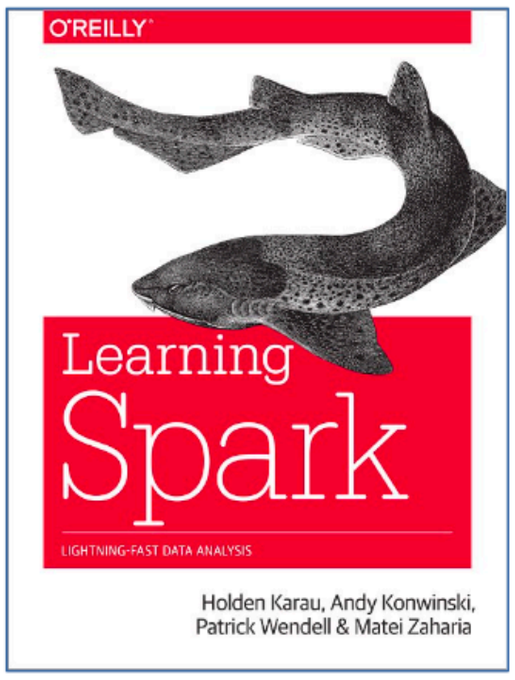
```
val input = sc.textFile("myfile.txt")
val apples = input.filter(x => x.contains("apple"))
val oranges = input.filter(x => x.contains("orange"))
val union_rdd = apples.union(oranges)
// prendo le prime 5 righe
val first5rows = union_rdd.take(5)


union_rdd.count()
```

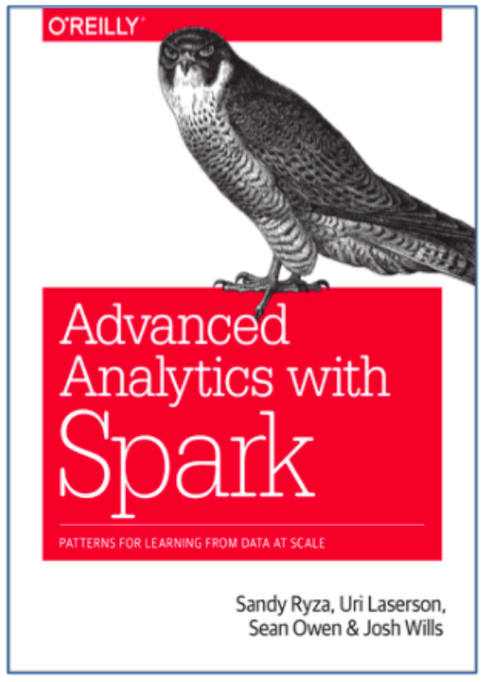
Dopo la union dei due dataset prendo solo le prime 5 righe:

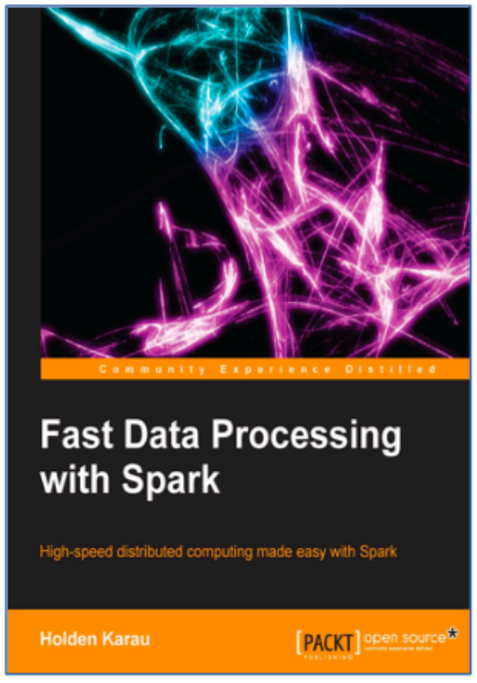
- se Spark fosse **eager** farebbe la union dei due dataset, trasformerebbe tutto in una collezione di elementi e prenderebbe le prime 5 righe
- - ma Spark è **pigro**! Non calcola l'intera unione, ma prepara solo i primi 5 elementi della collezione.

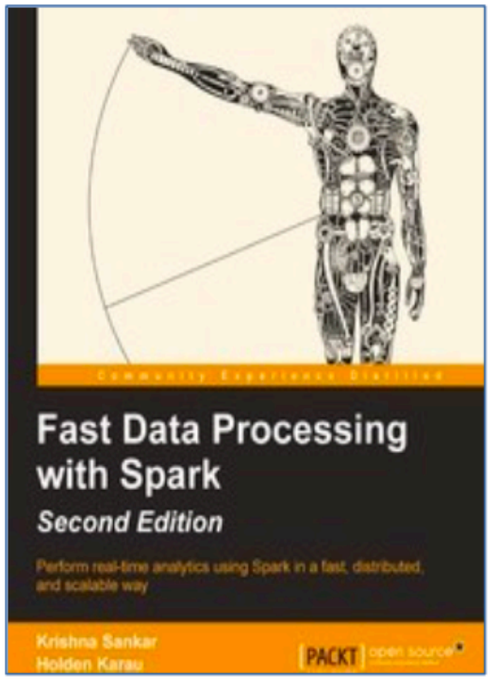
Resources

https://spark.apache.org









MongoDB

Quale è la differenza fra database e DBMS?

uhmm…

C’è una differenza?

Quali sono le proprietà di un DBMS relazionale?

Caratteristiche di NoSql

- Non usano SQL (Hbase, Cassandra, Redis,…)
- Sono generalmente progetti open source
- Gran parte di essi lavorano su cluster
- I RDBMS usano le transazioni ACID per gestire la consistenza, NoSQL utilizzano altri metodi
- Sono schema free o schema less
- NoSQL è un movimento piuttosto che una tecnologia
- I RDBMS non sono destinati a scomparire, ma la novità è che ora esistono altre possibilità

CAP Theorem

«Of three properties of shared data systems, data **Consistency**, system **Availability** and tolerance to network **Partitions**, only two can be achieved at any given moment in time»

University of California at Berkeley, Eric Brewer

- CA: relazionale tradizionale

- AP: le richieste vengono eseguite su ogni nodo anche se violano la consistenza (Cassandra, DynamoDB)

- CP: le richieste vengono eseguite solo sui nodi che garantiscono la consistenza (Hbase, MongoDB, Google BigTables)

In pratica…

If you cannot limit the number of faults and requests can be directed to any server and you insist on serving every request you receive then you cannot possibly be consistent

CAP Theorem

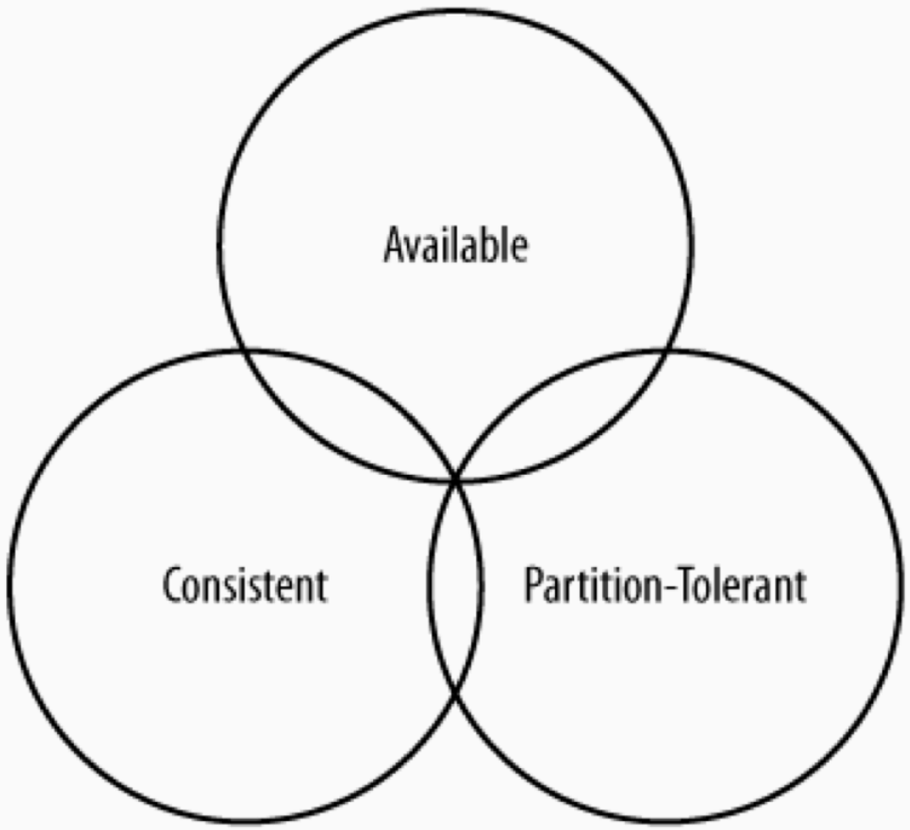


Figure 2-1. CAP Theorem indicates that you can realize only two of these properties at once

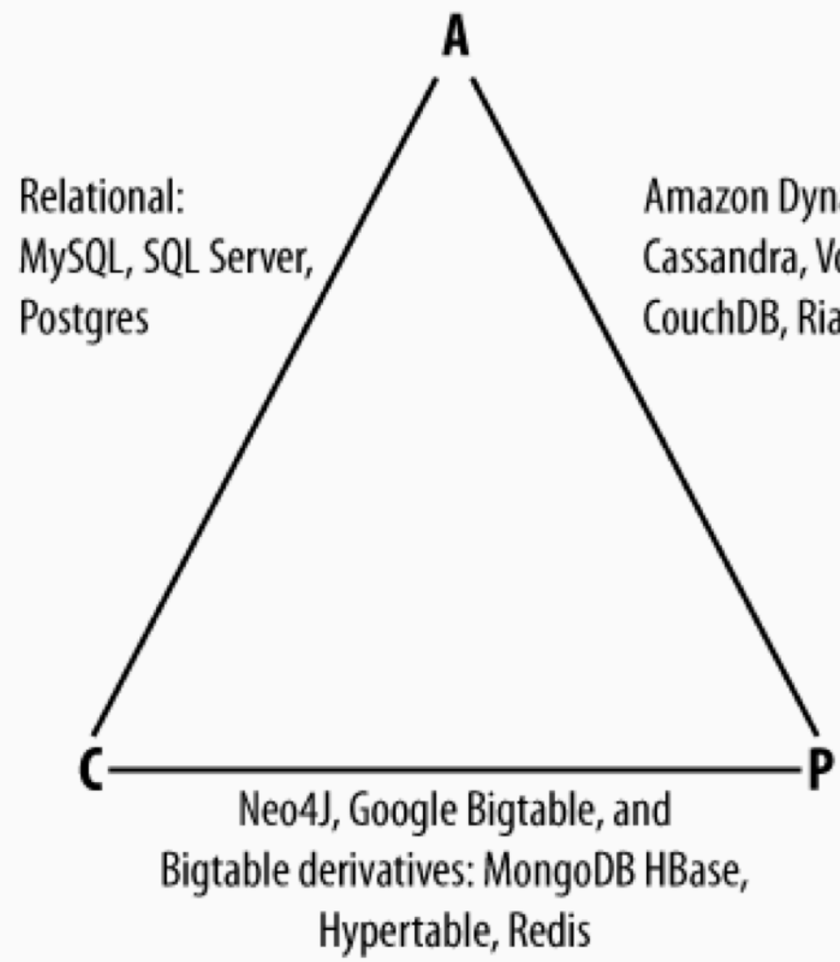
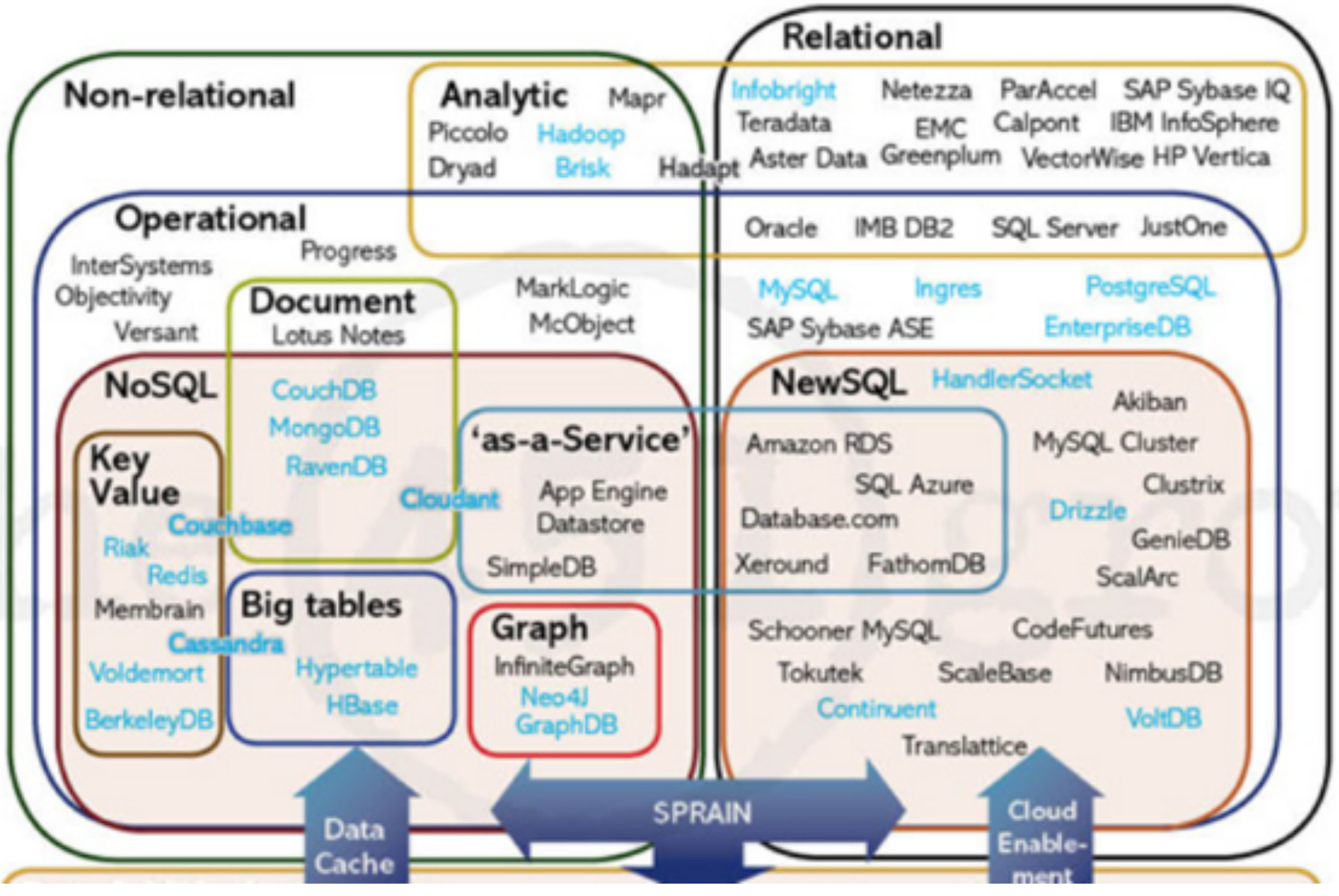


Figure 2-2. Where different databases appear on the CAP continuum

NoSQL

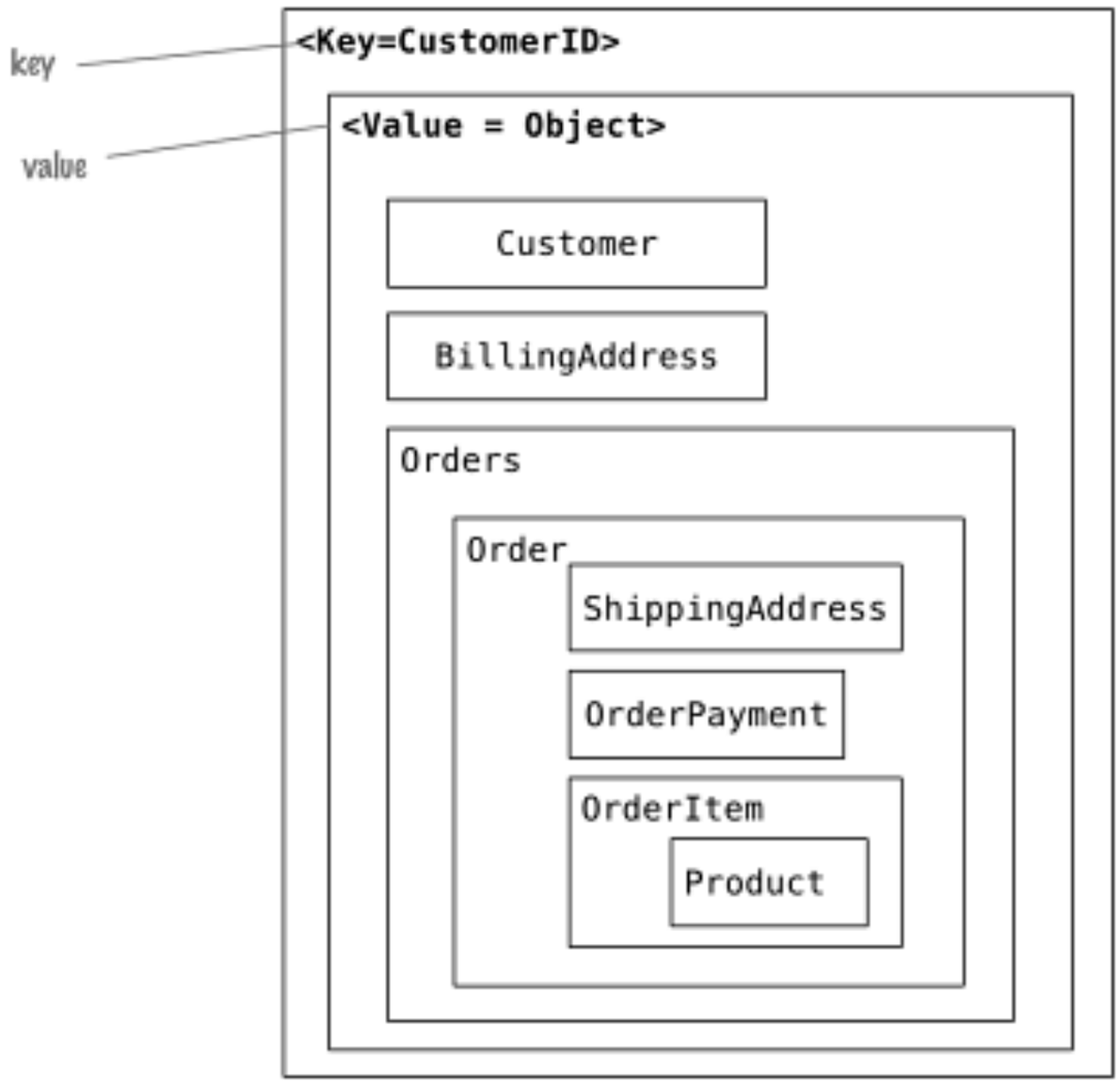
mongoDB



KeyValue Stores

- Key values stores
  - i dati vengono immagazzinati in un elemento che contiene una chiave assieme ai dati veri e propri
  - più semplice da implementare
  - più inefficiente se la maggior parte delle operazioni riguardano soltanto una parte di un elemento
  - Una chiave, un valore, nessun duplicato
  - Il valore è un oggetto binario chiamato «blob»: il database non sa cosa sia e non ha bisogno di saperlo

KeyValue Stores



Redis

WEBSITE	http://redis.io/
DATABASE MODEL	key-value store
INITIAL RELEASE	2009
LICENSE	
DESCRIPTION	Open source, in-memory data structure store, used as a database, cache and message broker.

flickr GitHub

stackoverflow

Voldemort

WEBSITE	http://www.project-voldemort.com/
DATABASE MODEL	key-value store
INITIAL RELEASE	2009
LICENSE	
DESCRIPTION	It is basically just a big, distributed, persistent, fault-tolerant hash table

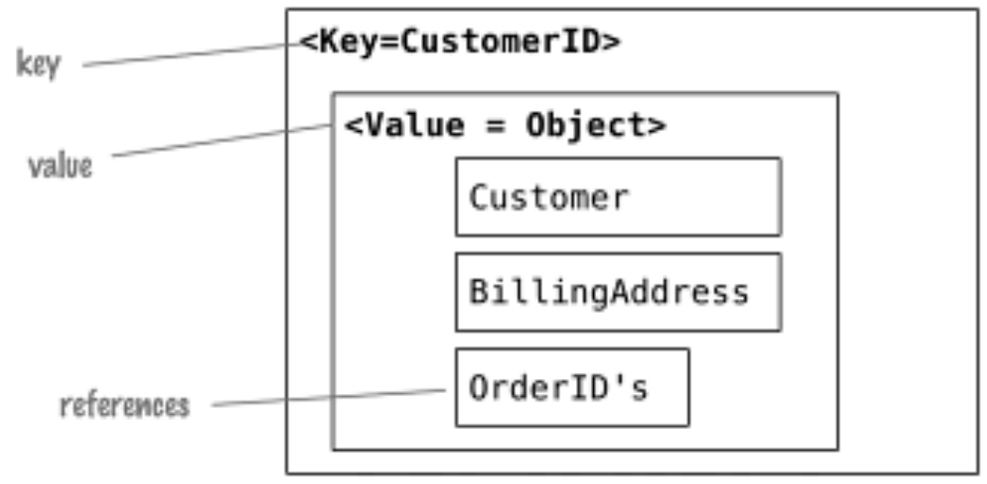
Document stores

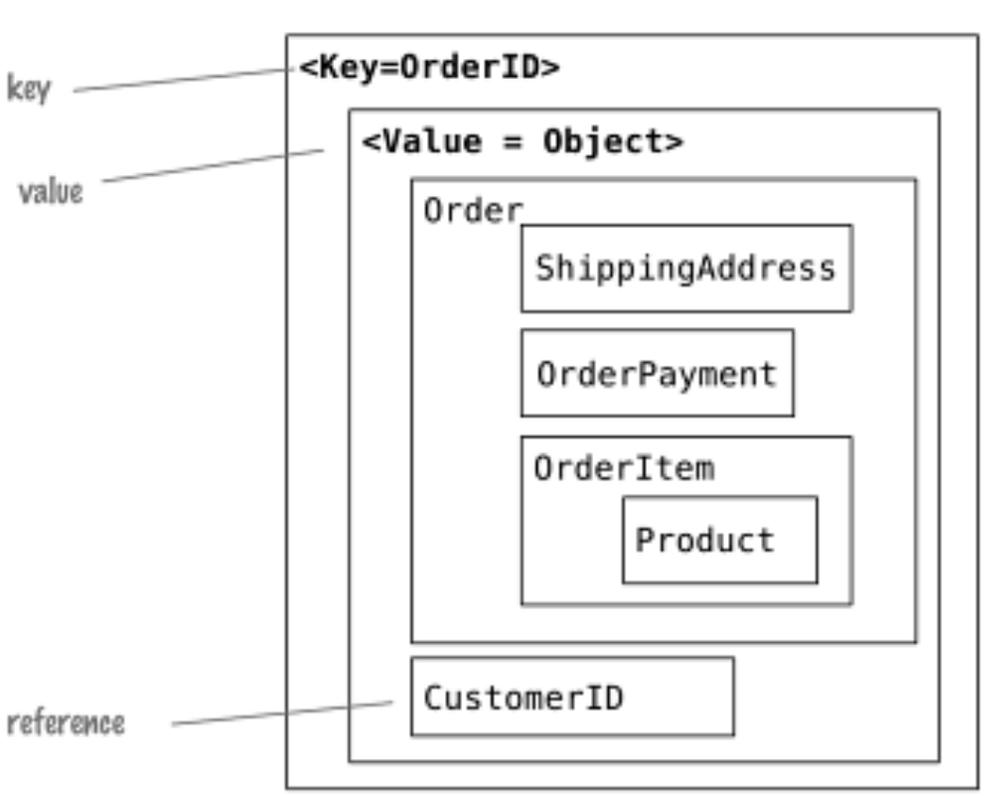
- Document stores
  - è l’evoluzione del metodo key/value
  - il valore è di solito strutturato e compreso dal DB
  - è possibile interrogare i dati (in modi diversi dall’uso della semplice chiave)

Document stores

Rispetto ai normali database relazionali invece che immagazzinare i dati in tabelle con dei campi fissi, questi vengono messi in un documento che può contenere illimitati campi di illimitata lunghezza, così se ad esempio di una persona conosciamo solo nome e cognome, ma magari di un’altra persona anche indirizzo, data di nascita e codice fiscale, si evita che per il primo nominativo ci siano campi inutilizzati che occupano inutilmente spazio

Document stores





MongoDB

WEBSITE	https://www.mongodb.com/
DATABASE MODEL	document store
INITIAL RELEASE	2009
LICENSE	AGPL Free Software Free as in Freedom
DESCRIPTION	MongoDB is a document database with the scalability and flexibility that you want with the querying and indexing that you need

DynamoDB

WEBSITE	https://aws.amazon.com/dynamodb/
DATABASE MODEL	document store
INITIAL RELEASE	2012
LICENSE	commercial
DESCRIPTION	È un database cloud interamente gestito che supporta sia i modelli di storage document store sia quelli di tipo chiave-valore.
WHO USES	

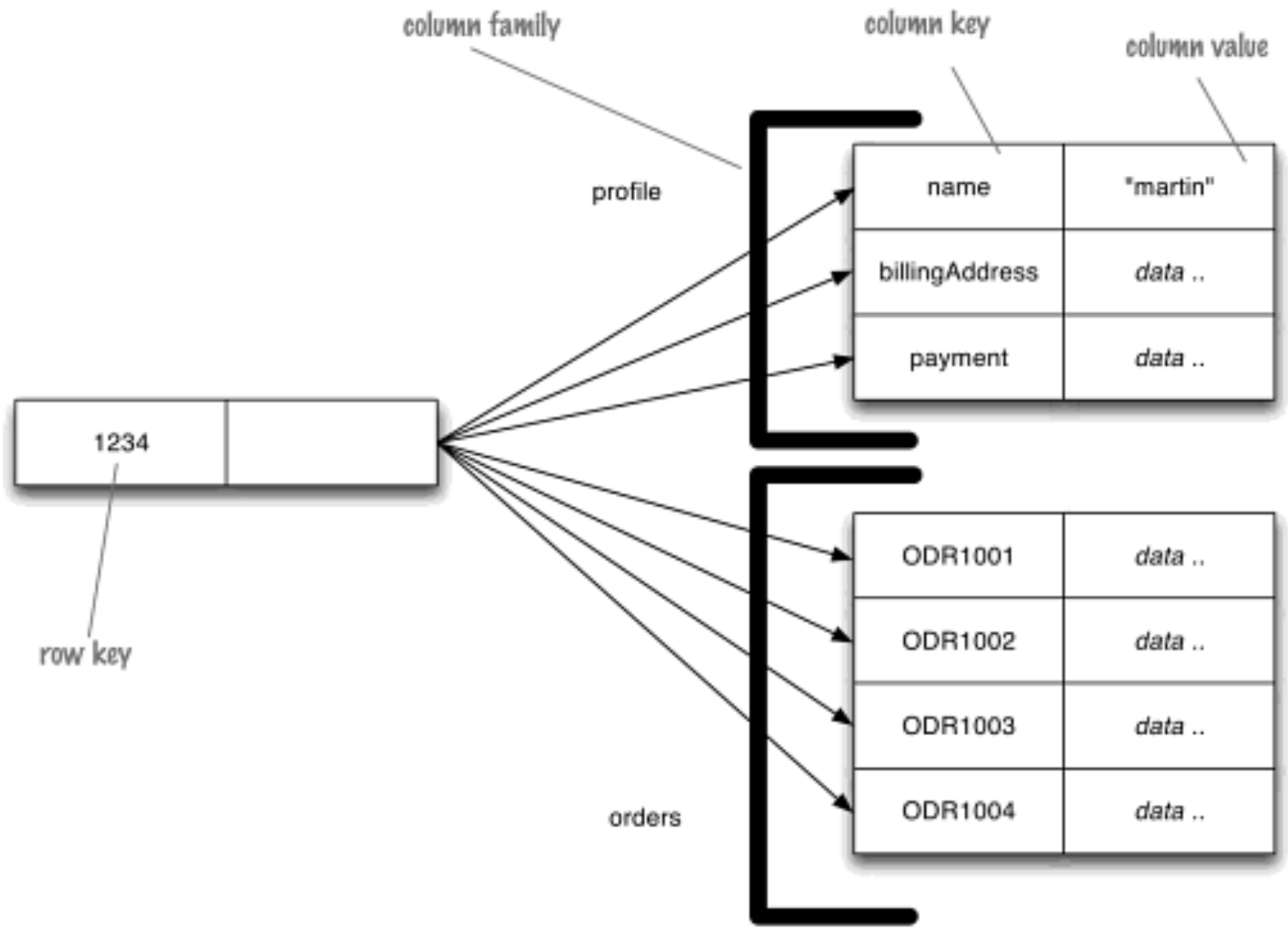
SHAZAM

NETFLIX

Column family stores

- Google’s BigTable
  - Il nome deriva dalla sua struttura composta da colonne sparse e nessuno schema
- Non dobbiamo pensare alla sua struttura come una tabella ma come una mappa a due livelli
- Molti database utilizzano le righe come unità di storage
- Tuttavia in molti casi è necessario accedere a poche colonne di molte righe

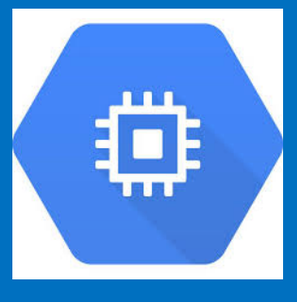
Column family stores



Cassandra

WEBSITE	http://cassandra.apache.org/
DATABASE MODEL	wide column store
INITIAL RELEASE	2008
LICENSE	
DESCRIPTION	Wide-column store based on ideas of BigTable and DynamoDB

Google Big Table



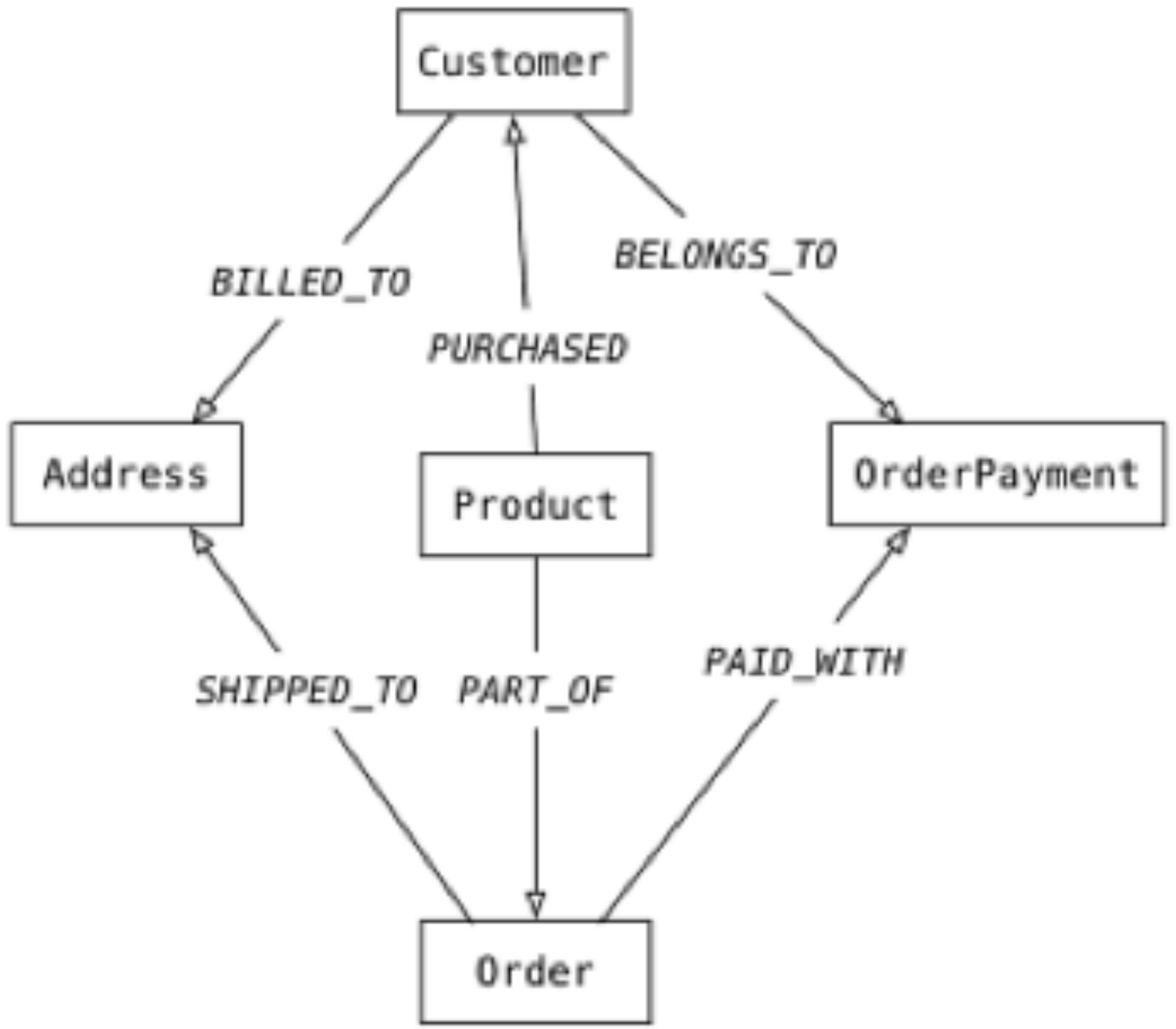
WEBSITE	https://cloud.google.com/bigtable/
DATABASE MODEL	wide column store
INITIAL RELEASE	2015
LICENSE	commercial
DESCRIPTION	Google's NoSQL Big Data database service. It's the same database that powers many core Google services, including Search, Analytics, Maps, and Gmail.

Graph databases

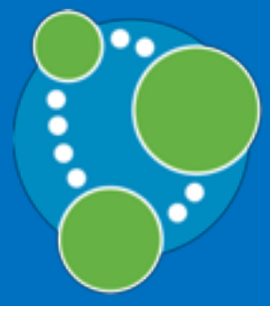
- Graph databases

  - i dati vengono immagazzinati sotto forma di strutture a grafi, rendendo più performante l’accesso a questi da applicativi orientati agli oggetti

Graph databases



Neo4J



WEBSITE	https://neo4j.com/
DATABASE MODEL	graph DBMS
INITIAL RELEASE	2007
LICENSE	GPL Free Software Free as in Freedom
DESCRIPTION	Highly scalable native graph database that leverages data relationships as first-class entities

Walmart ebay

Time series

- Un Time series DBMS è un sistema ottimizzato per la gestione dei dati delle serie temporali: ogni voce è associata a un timestamp

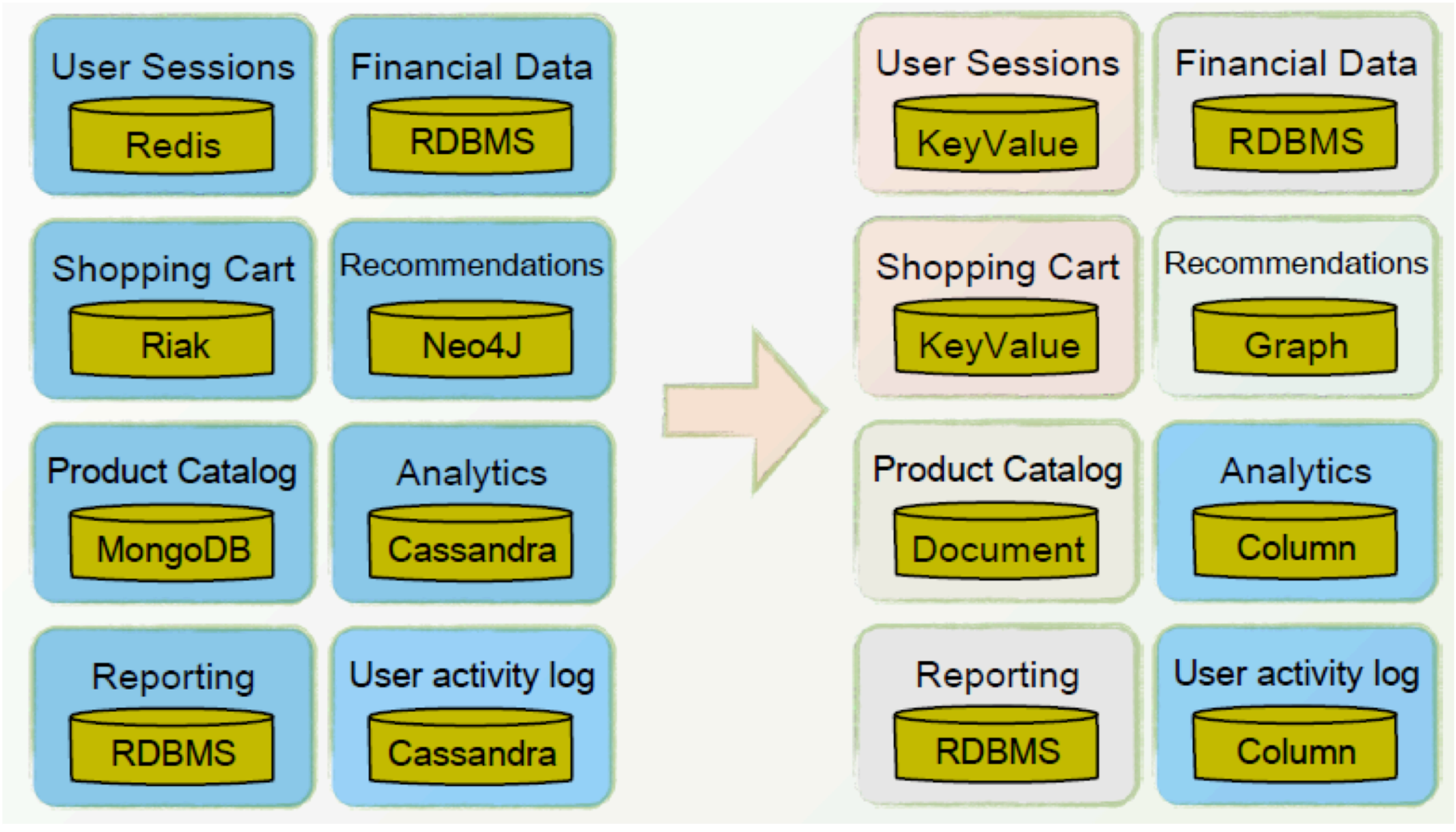
- I dati delle serie temporali possono essere prodotti da sensori, smart meter o RFID nel mondo IoT o possono rappresentare i titoli di borsa di un sistema di negoziazione di titoli ad alta frequenza

- Sono progettati per raccogliere, archiviare e interrogare in modo efficiente serie temporali con elevati volumi di transazioni

Quando usare un Document Store

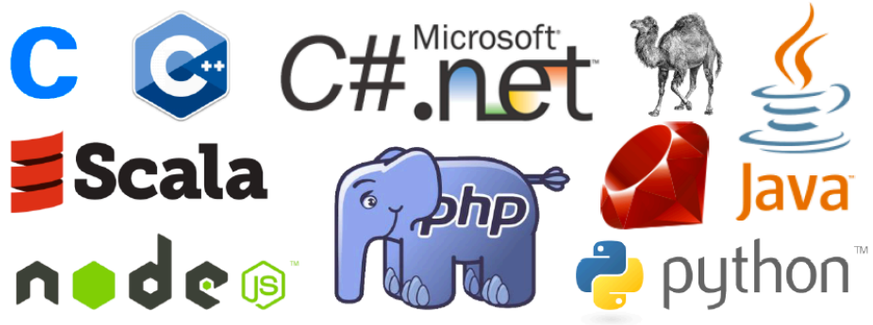
- Quando i dati rappresentano collezioni di oggetti simili
- Semi strutturati o sparsi piuttosto che tabulari
- Quando i campi degli oggetti hanno valori multipli

Polyglot persistence



Cosa è MongoDB?

- https://www.mongodb.com/
- Nasce nel 2017 (10gen)
  - La prima versione per produzione è pronta nel 2010
  - Ora siamo alla versione 4
- **Document Oriented NoSQL Database**
  - Hash-based, schema-less database
  - No Data Definition Language
  - Ogni documento è identificato da un identificativo (**_id**)
  - BSON format (Based on JSON –B stands for Binary)
- Le applicazioni gestiscono lo schema dei dati
- Scritto in C++
- Supporta API (driver) per molti diversi linguaggi
  - JavaScript, Python, Ruby, Perl, Java, JavaScala, C#, C++, Haskell, Erlang
  - Off-the-shelf drivers (Hadoop, BI tools,…)
  - REST interface
- MultiOS (Windows, Linux, macOS)
- Cloud (AWS, Azure, GCP,…) + MongoDB Atlas



Funzionalità

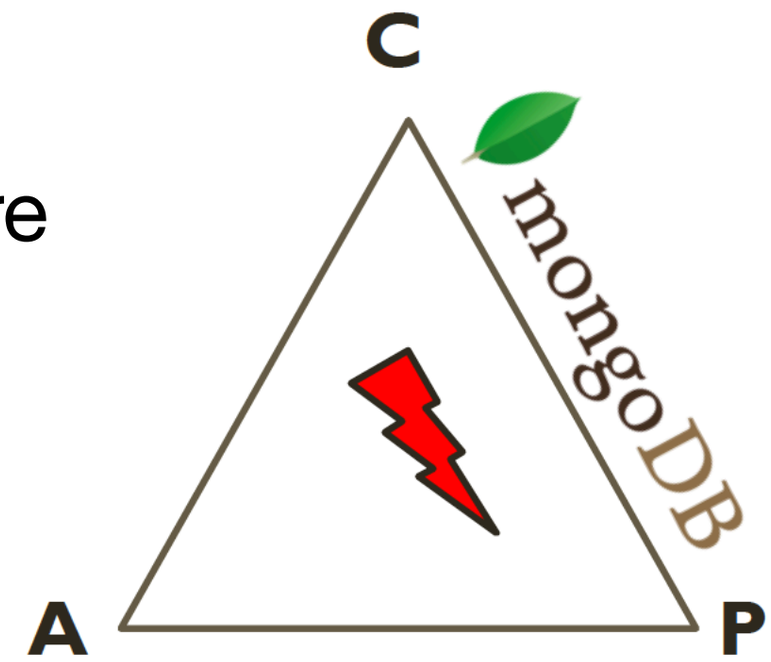
- Dynamic schema
  - No DDL
- Document-based database
- Secondary indexes
- Query language via an API
- Atomic writes and fully-consistent reads
  - If system configured that way
- Master-slave replication with automated failover (replica sets)
- Built-in horizontal scaling via automated range-based partitioning of data (sharding)
- **No joins nor transactions**

Perché usare MongoDB?

- Query molto semplici
  - Linguaggio di scripting molto potente
  - Scale out
- Integrazione dei dati veloci
- Non adatto per transazione complesse

MongoDB e CAP

- Consistenza
  - Tutte le repliche contengono la stessa versione dei dati
- Disponibilità
  - il sistema rimane attivo in caso di failure
- Partition tolerance
  - entry points multipli
  - il sistema rimane attivo in caso di split



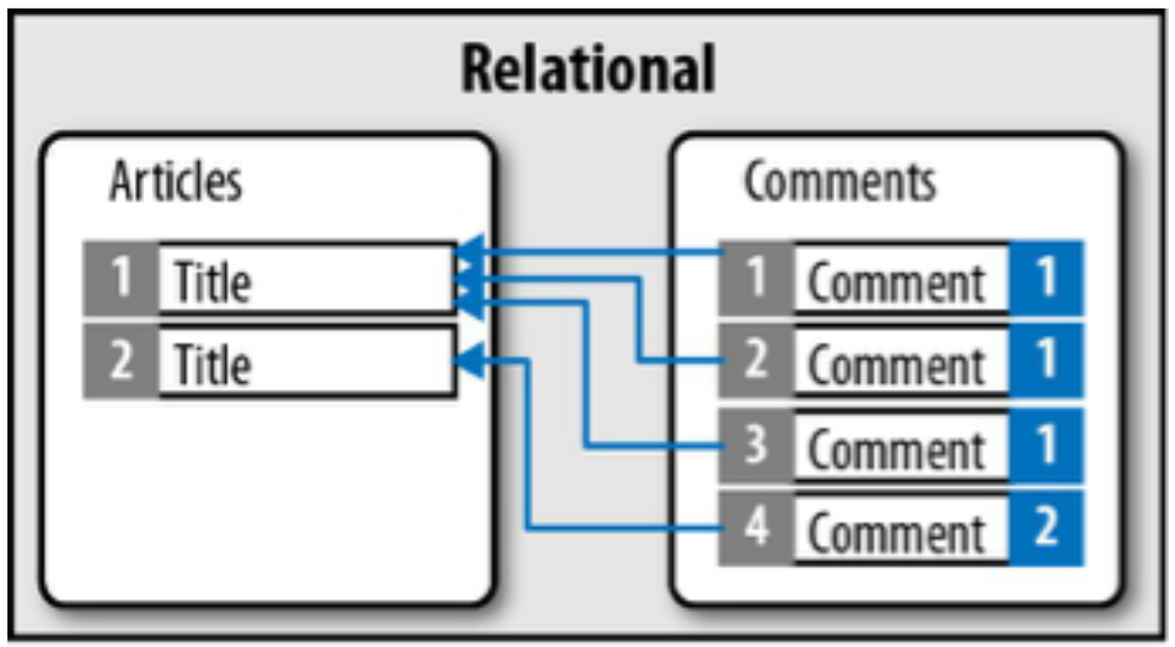
Hierarchical Objects

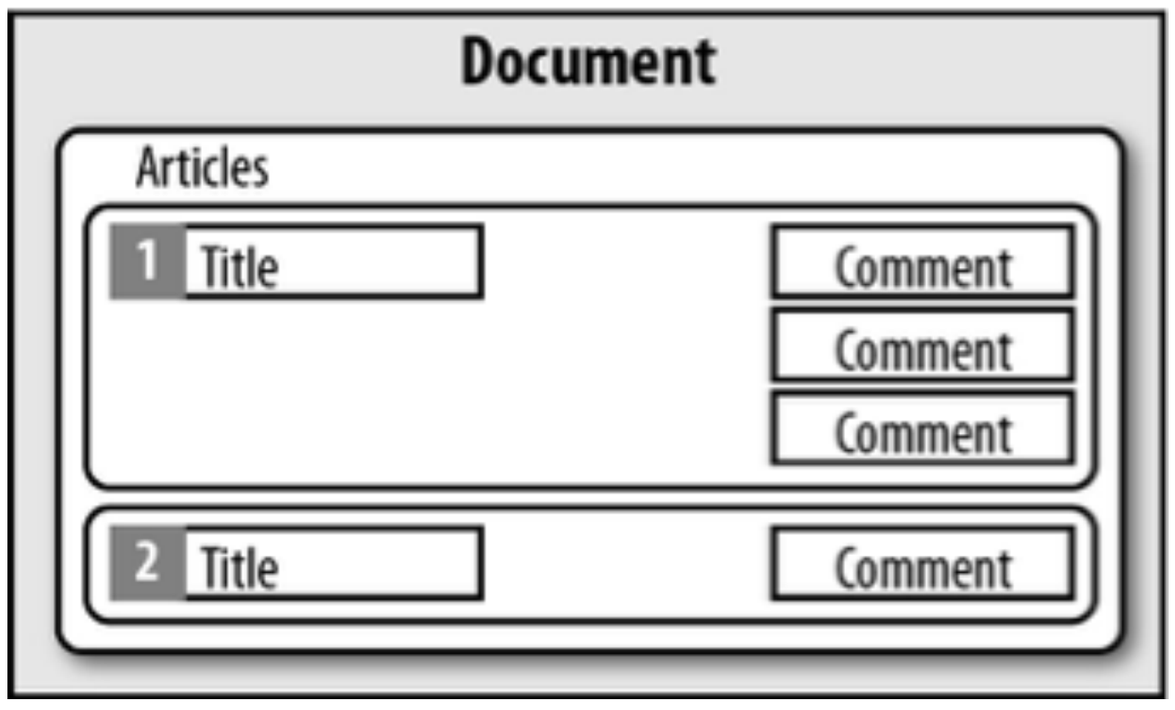
- A MongoDB instance may have zero or more ‘databases’
- A database may have zero or more ‘collections’
- A collection may have zero or more ‘documents’
- A document may have one or more ‘fields’
- MongoDB ‘Indexes’ function much like their RDBMS counterparts

RDB vs MongoDB

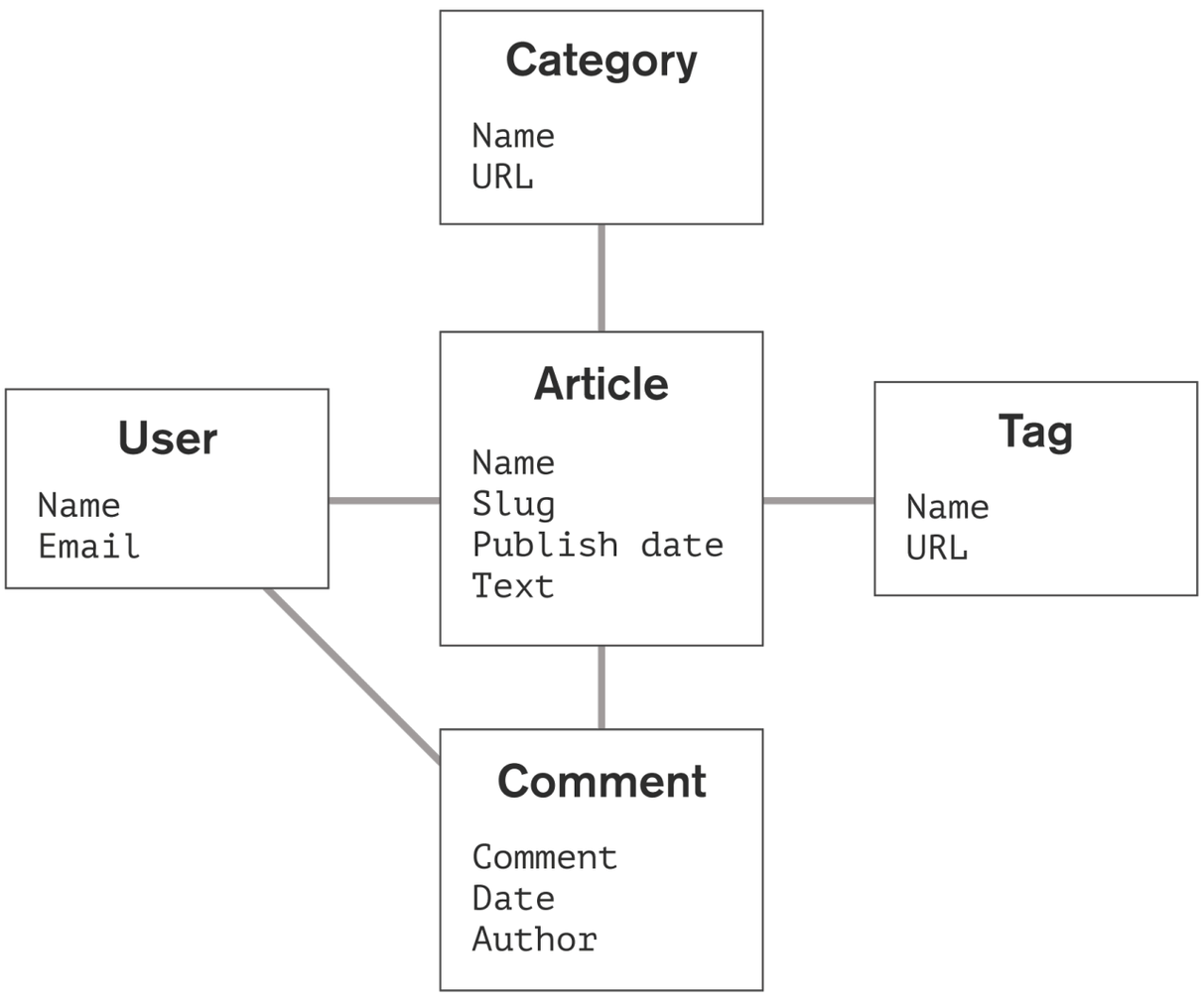
RDBMS	MongoDB
Database	Database
Table, view	Collection
Row	Document
Column	Field
Index	Index
Join	Embedded Document
Foreign key	Reference
Partition	Shard

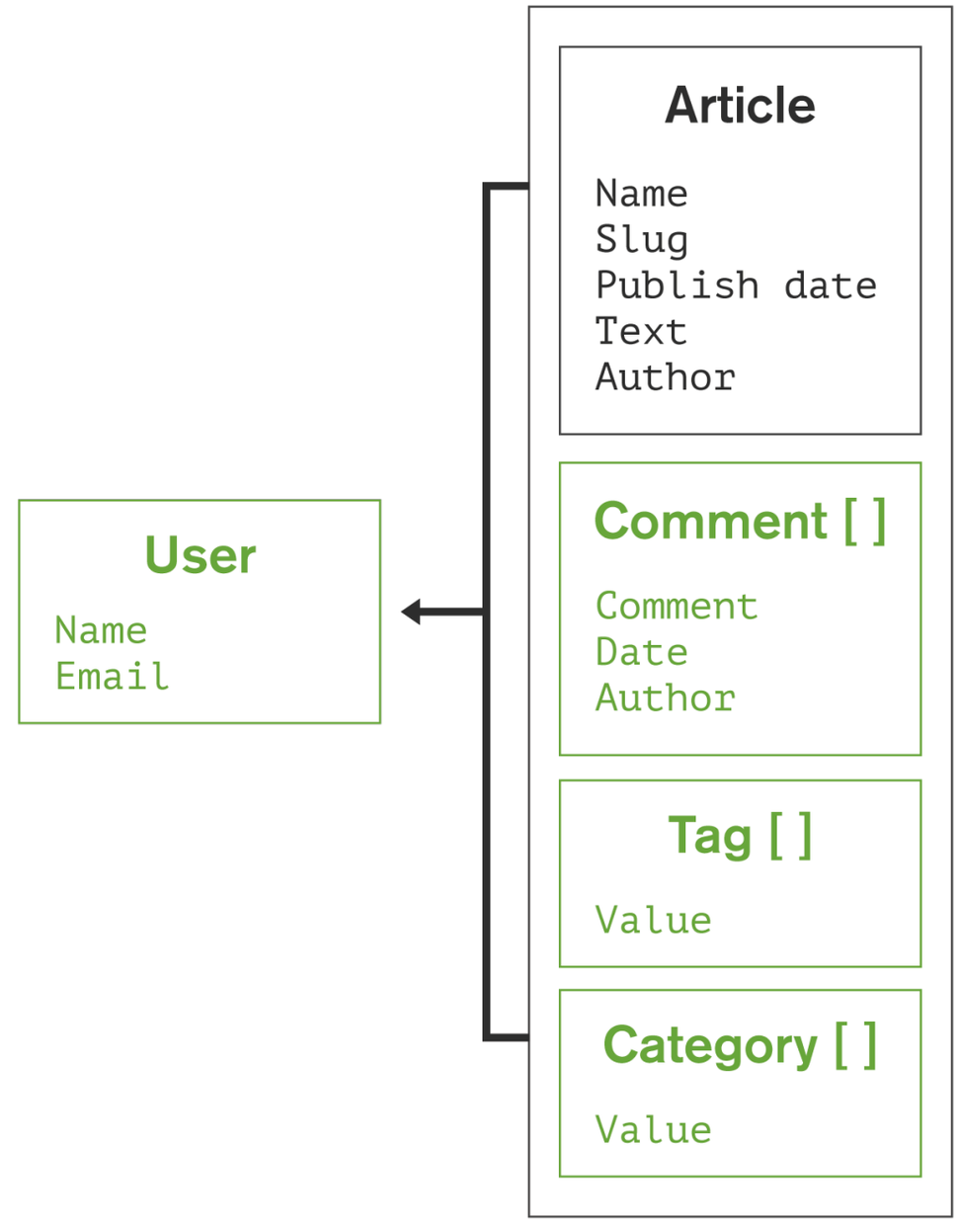
RDB vs MongoDB





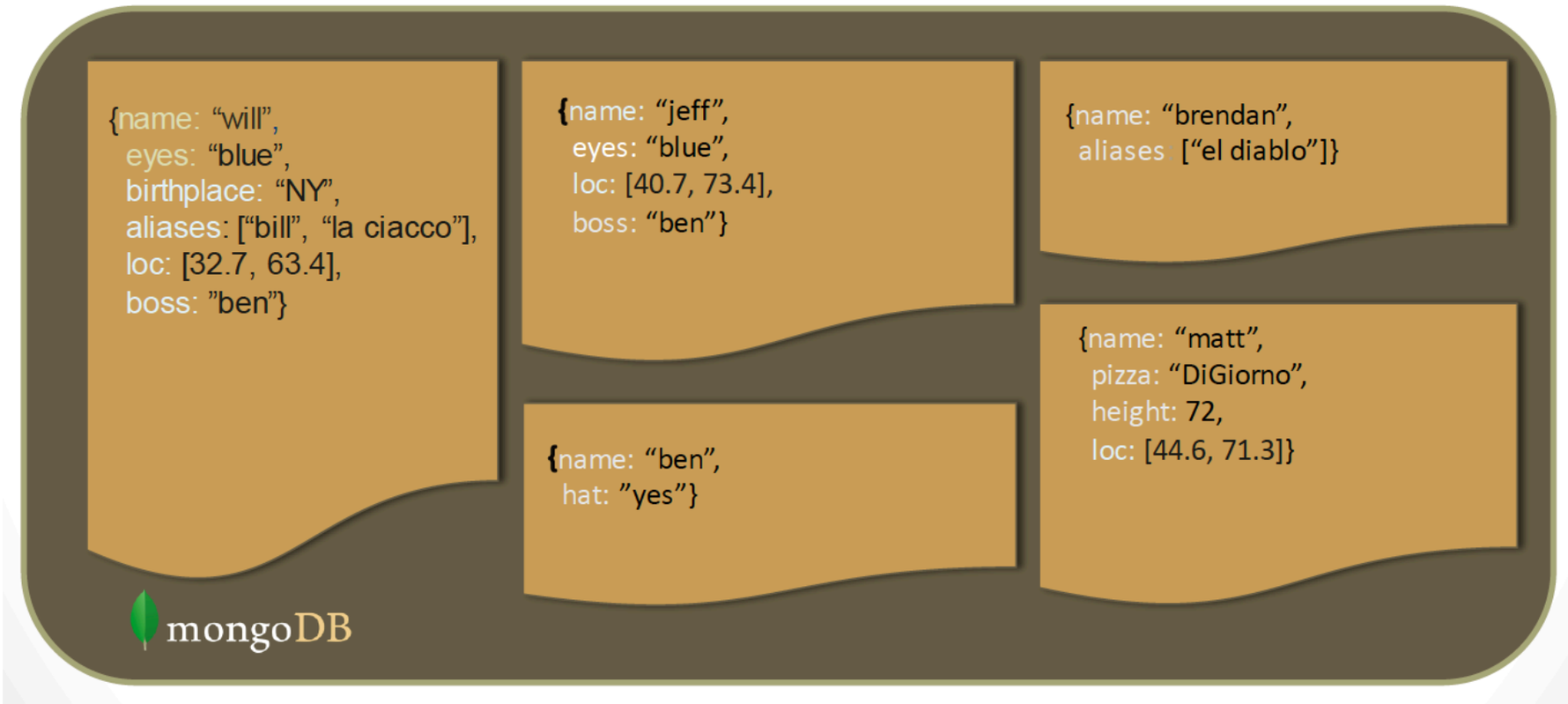
Data As Documents





What is a document?

- MongoDB, such as almost all NoSQL DBs, stores data in json format (BSON format)
- A single document has it's own json and any json has it's own structure. So multipe documents in a collection can have totally different structures (or schemas). This is schema-free paradigm.
- The maximum BSON document size is 16 MB. The maximum document size helps ensure that a single document cannot use excessive amount or RAM or, during transmission, excessive amount of bandwidth.



Mongo Document


```
{
    name: ‘Mauro Pelucchi',
    address: {
        street: ‘via Finazzi 47',
        city: 'Carvico'
}
}
```

Mongo Document


```
{
    _id: ObjectId(“4efa8d2b7d284dad101e4bc9”),
    name: ‘Mauro Pelucchi',
    address: {
        street: ‘via Finazzi 47',
        city: 'Carvico'
}
}
```

Mongo Document


```
{

    _id: 12345,

    name: ‘Mauro Pelucchi',

    address: {

        street: ‘via Finazzi 47',

        city: 'Carvico'

}

}
```

Come posso inserire gli indirizzi di spedizione?

Mongo Document


```
{
    _id: 12345,
    name: ‘Mauro Pelucchi',
    address: {
        street: ‘via Finazzi 47',
        city: 'Carvico'
    },
    shipAdresses: [
        {strett: ‘via Finazzi 47’, city: ‘Carvico’},
        {strett: ‘via Cavour 23’, city: ‘Carvico’, note: ‘suonare a Gianpietro’},
        {strett: ‘Edificio U7 - Università degli Studi di Milano Bicocca’, city: ‘Milano’}
    ]
}
```

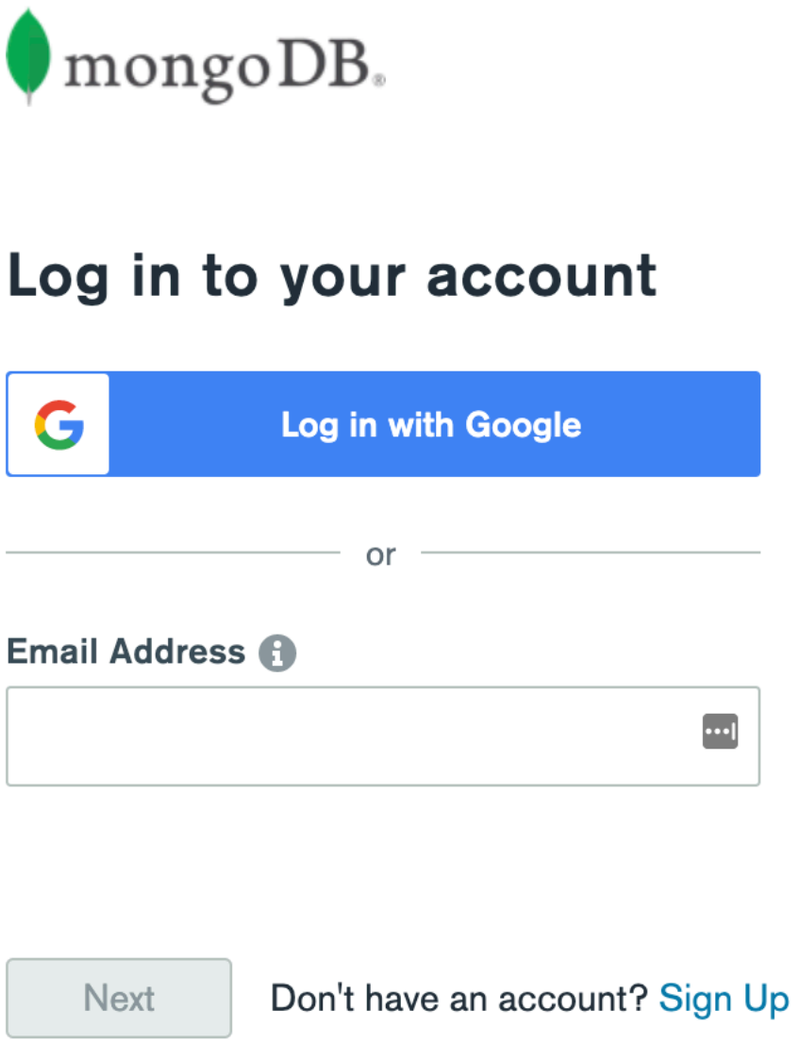
MongoDB Atlas

Database as a Service for MongoDB

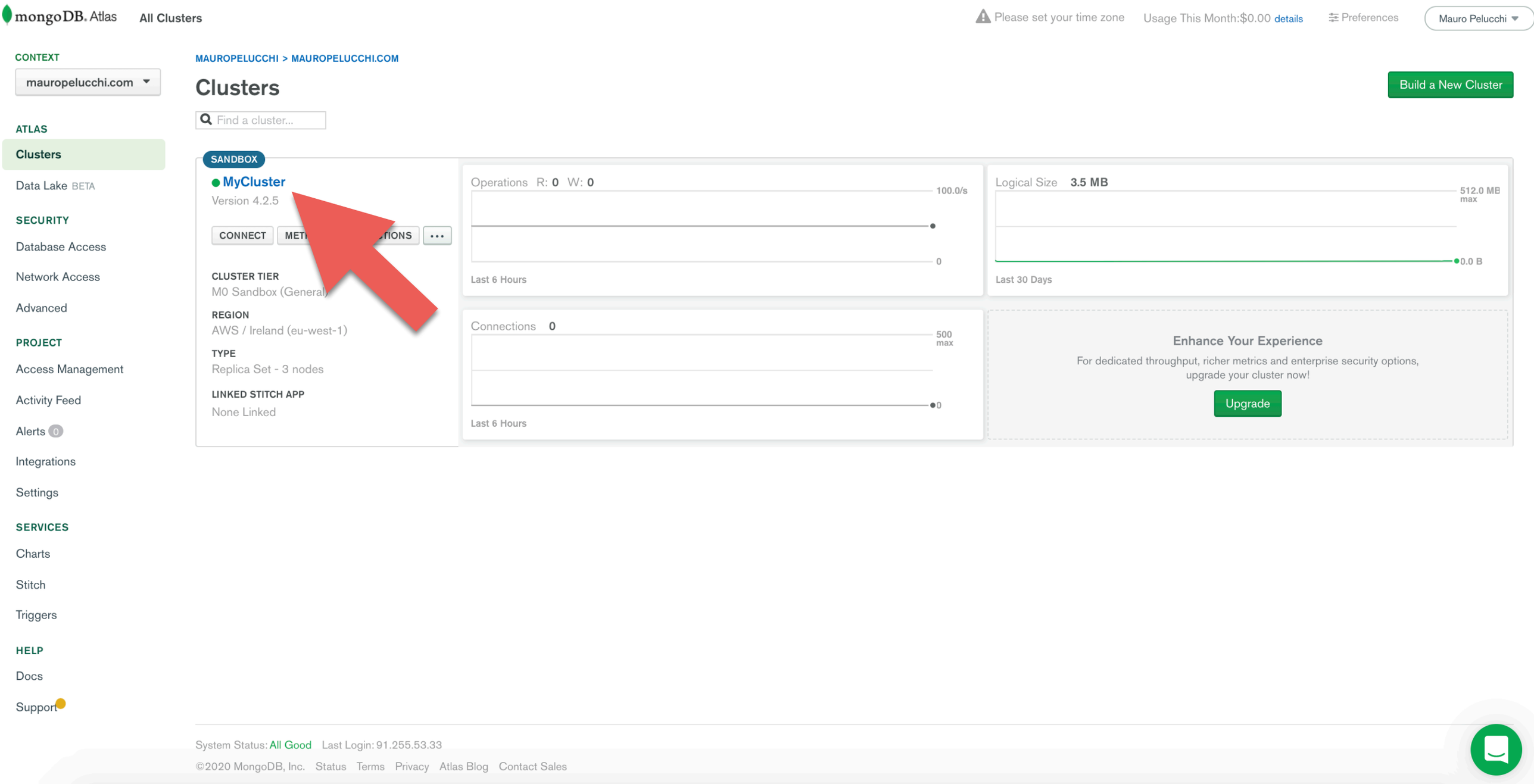
- Scalable back-end for your application on-demand
- Secure by default
- Highly available, even while scaling
- Patch maintenance performed for you
- Your own MongoDB cluster in the cloud

MongoDB Atlas

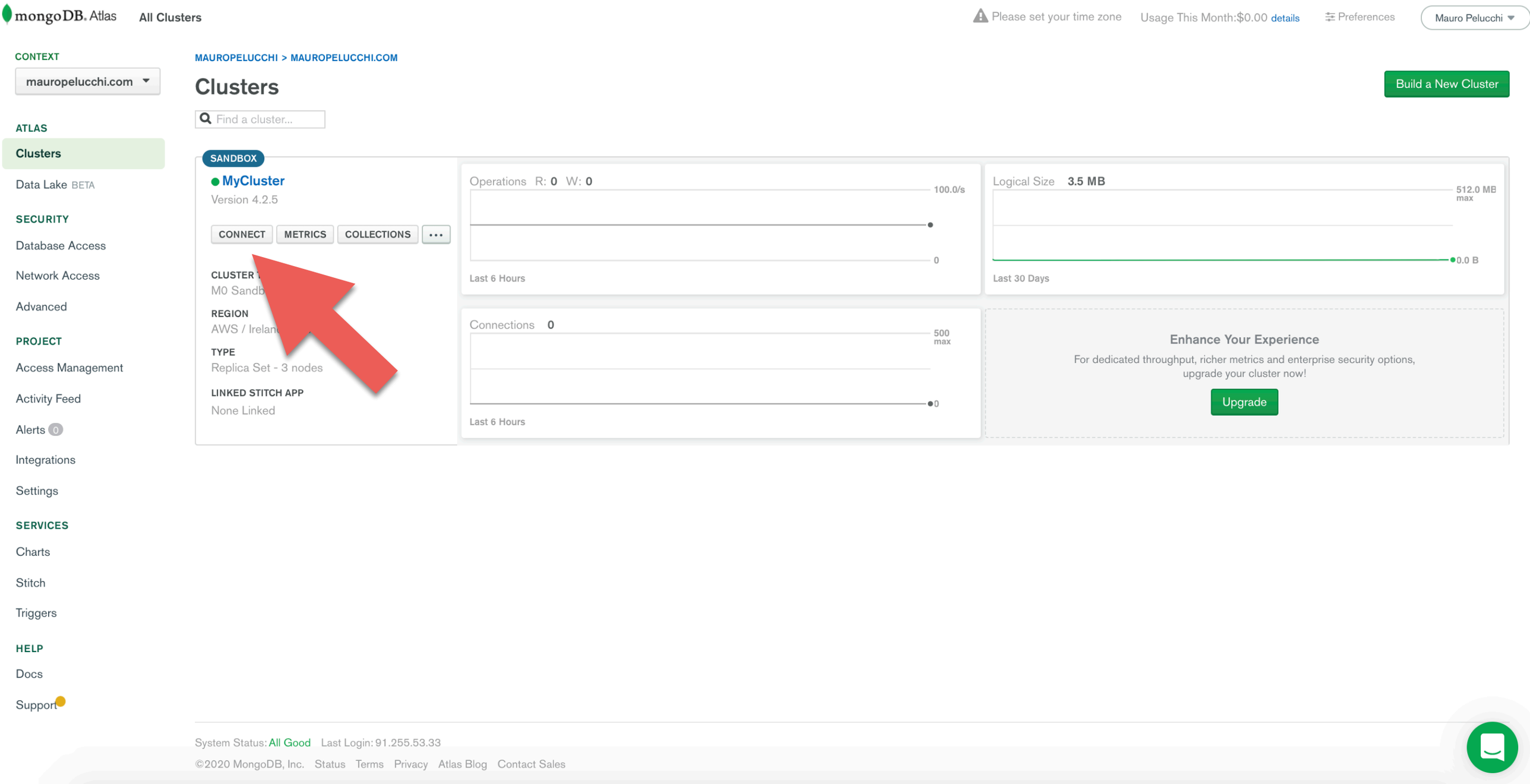
https://account.mongodb.com/account



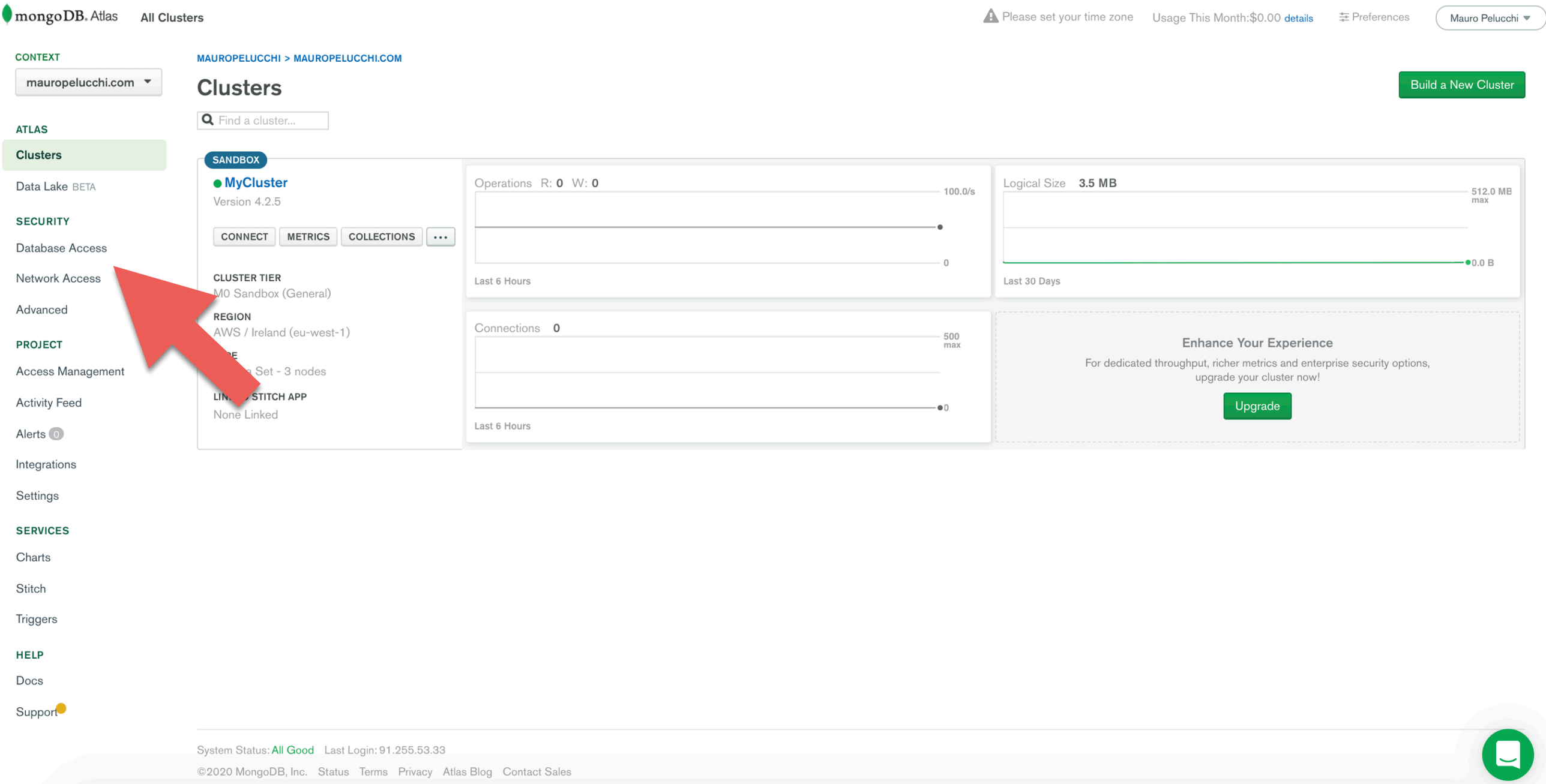
MongoDB Atlas



MongoDB Atlas



MongoDB Atlas

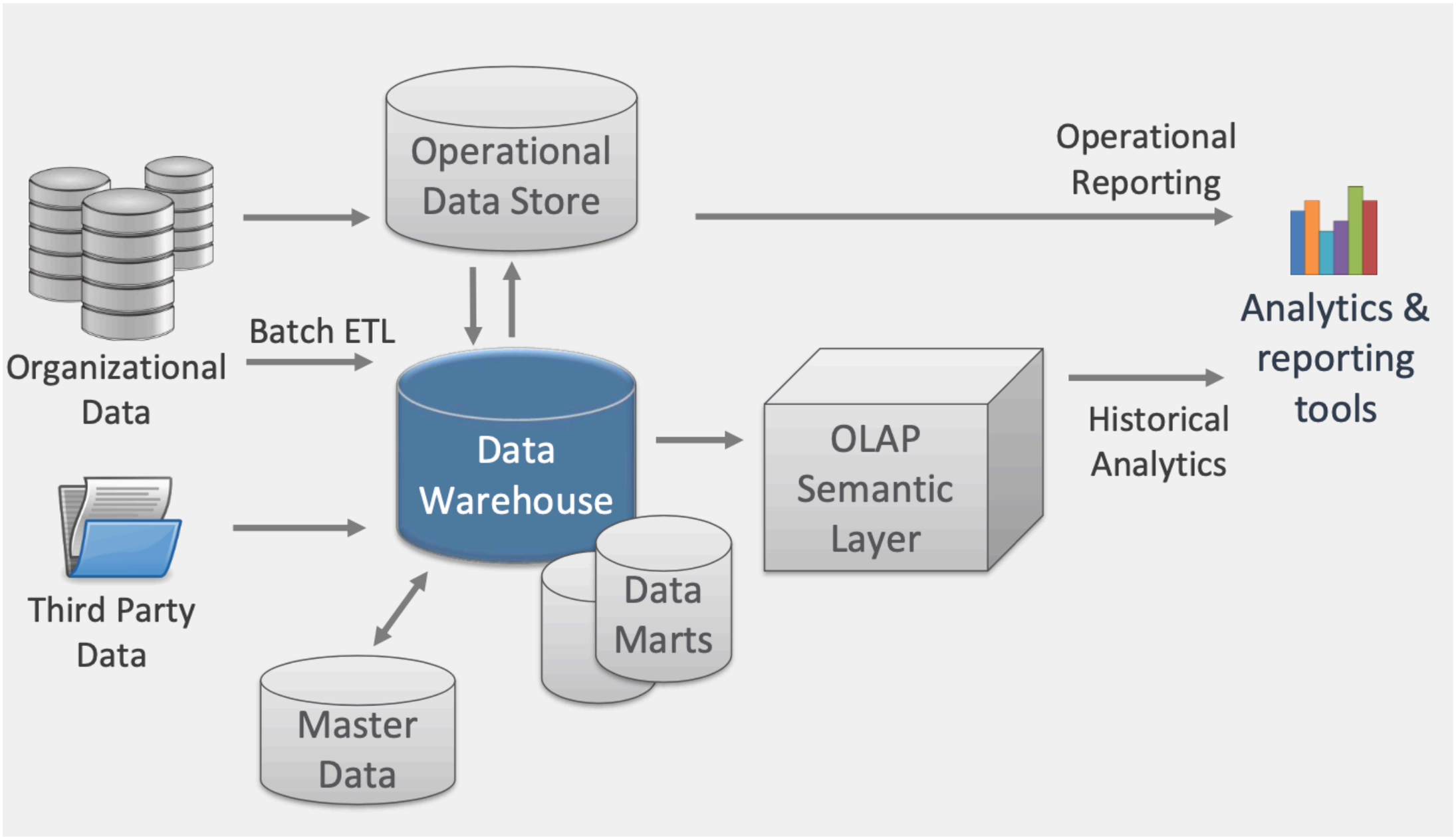


AWS Glue

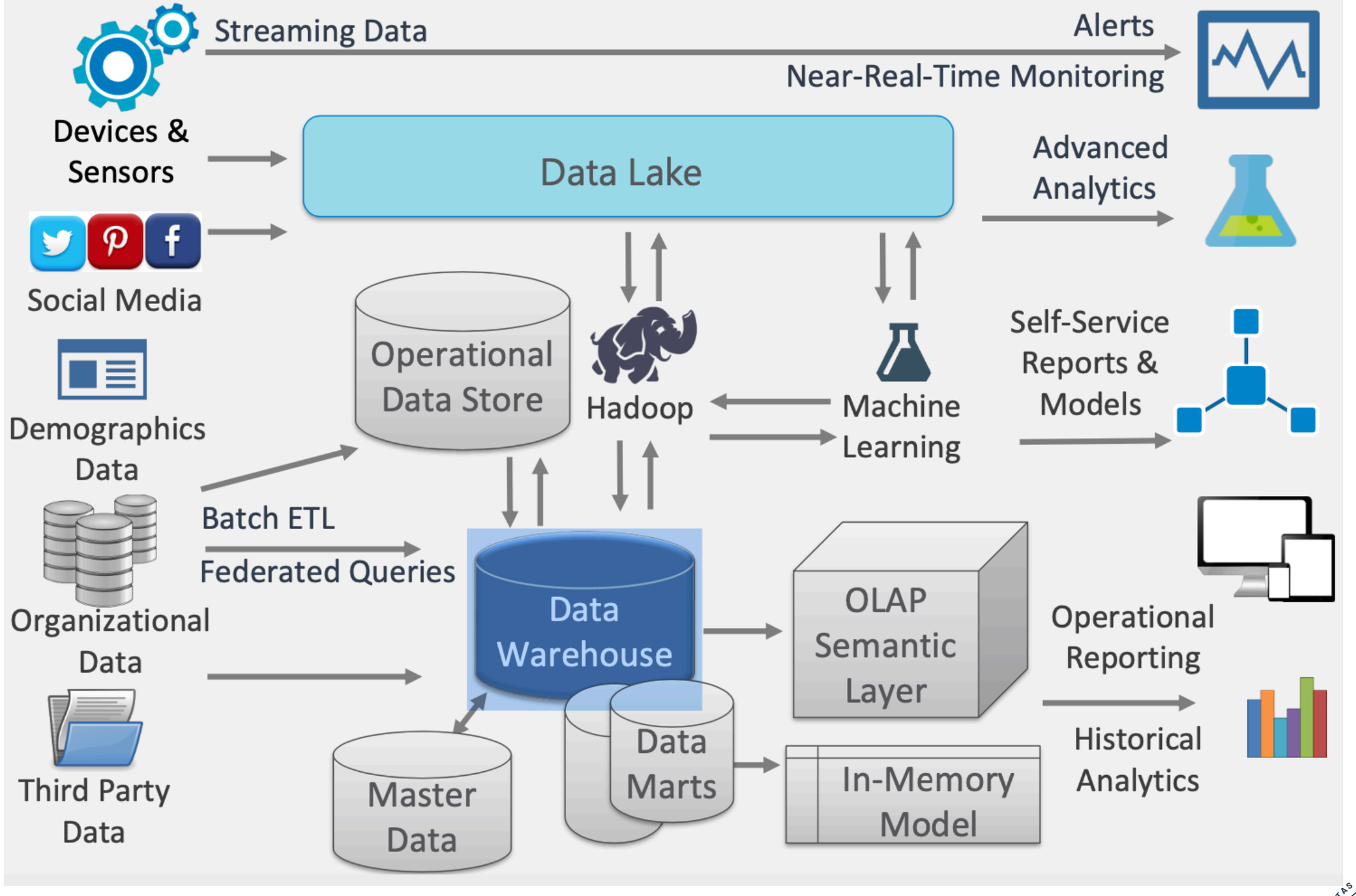
Cosa è un data warehouse?

Come deve essere fatto un data warehouse moderno?

Traditional Data Warehousing



Modernizing an Existing DW



Features for a Modern Data Warehouse

- Coexists with Data lake
- Variety of data sources
- Multi Platform architecture
- Flexible deployment
- Support all user types & levels
- Near real-time data
- Advanced analytics
- Cloud integration
- APIs
- Data catalog
- Scalable architecture
- Governance model & MDM

Data Lake

- A repository for analyzing large quantities of disparate sources of data in its native format

- One architectural platform to house all types of data:

- Machine-generated data (ex: IoT, logs)

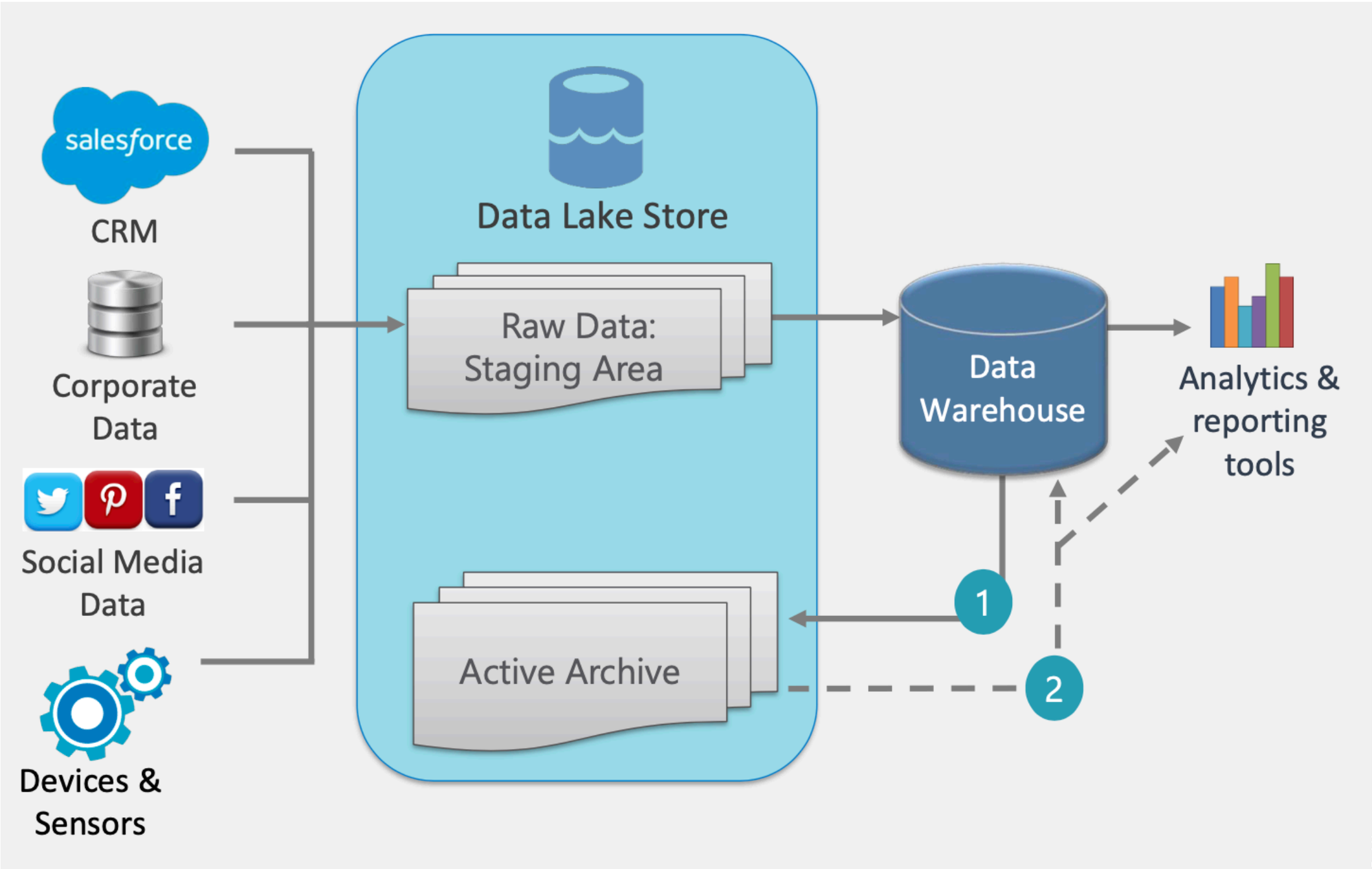
- Human-generated data (ex: tweets, e-mail, video, images, …)

- Traditional operational data (ex: sales, inventory)

Data Lake

- Reduce up-front effort by ingesting data in any format without requiring a schema initially
- Make acquiring new data easy, so it can be available for data science & analysis quickly
- Store large volume of multi-structured data in its native format
- Defer work to schematize after value & requirements are known
- Achieve agility faster than a traditional data warehouse can
- Speed up decision-making ability
- Storage for additional types of data which were historically difficult to obtain

Data Lake


salesforce
CRM
Corporate Data
Social Media Data
Devices & Sensors
Data Lake Store
Raw Data: Staging Area
Active Archive
Data Warehouse
Analytics & reporting tools
1
2

Data Lake Implementation

- A data lake is a conceptual idea. It can be implemented with one or more technologies.
  - HDFS (Hadoop Distributed File Storage) is a very common option for data lake storage.
  - NoSQL databases are also very common.
  - **MongoDB!!!**
  - Object stores (like Amazon S3 or Azure Blob Storage)

Data Lake Challenge

Complex, multi-layered architecture	Unknown storage & scalability	Working with un-curated data	Performance
Data retrieval	Data quality	Governance	Redundant Effort

Supponete di dover raccogliere ed
elaborare dati per un grande sito di ecommerce…

Dovrete gestire dati transazionali, testi, immagini, video, log,
…

Da database tradizionali e db NoSQL

Per costruire un DWH per supportare le analisi delle vendite
da una App

Quali sfide vi aspettano?

Challenges

- Ingest data from various data sources and join them together
- Enrich raw data
- Clean & Merge raw data
- Convert data to a new format to ensure efficient querying
- Grant access to roles based on the data classification
- SQL Access for Data Scientists
- Data Visualization with charts and graphs

Data Pipeline Challenges

Short timeline to produce a data layer.

Resources are not data scientists or ETL developers.

Data is in different spreadsheets or existing databases

Catalog vs. warehousing

What is AWS Glue?

Fully-managed,
server-less extract-transform-load (ETL) service

AWS Glue

- Serverless ETL
- Universal Data Catalog
- Open source Apache Spark environment
- DynamicFrame – Built in functions
- Seamless integration with AWS services
- Support for on-premises data stores

AWS Glue Components

Data Catalog

Job Authoring

Job Execution

Discover

Automatic crawling

Apache Hive Metastore compatible

Integrated with AWS analytic services

Develop

Auto-generates ETL code

Python and Apache Spark

Edit, Debug, and Explore

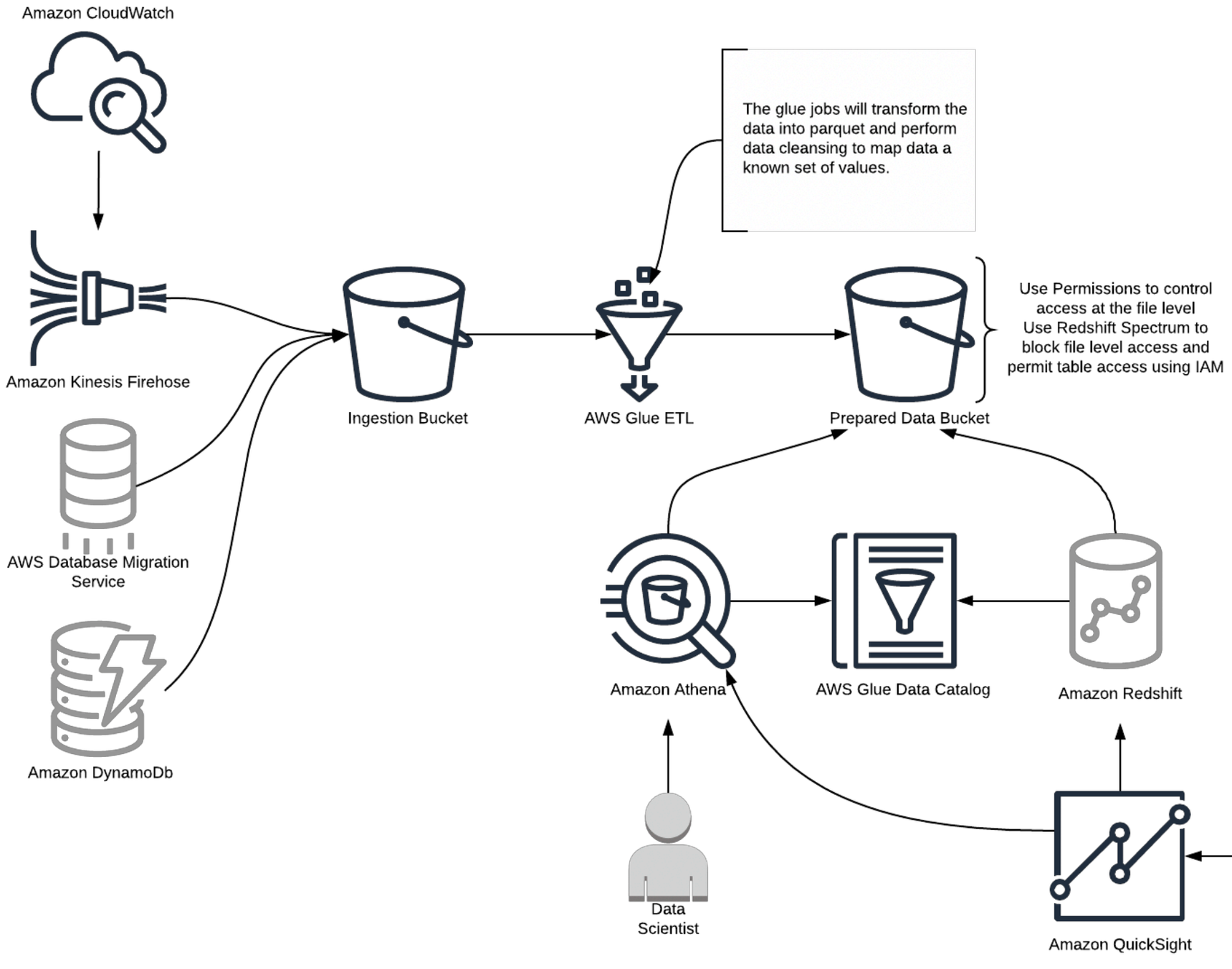
Deploy

Serverless execution

Flexible scheduling

Monitoring and alerting

High-level architecture



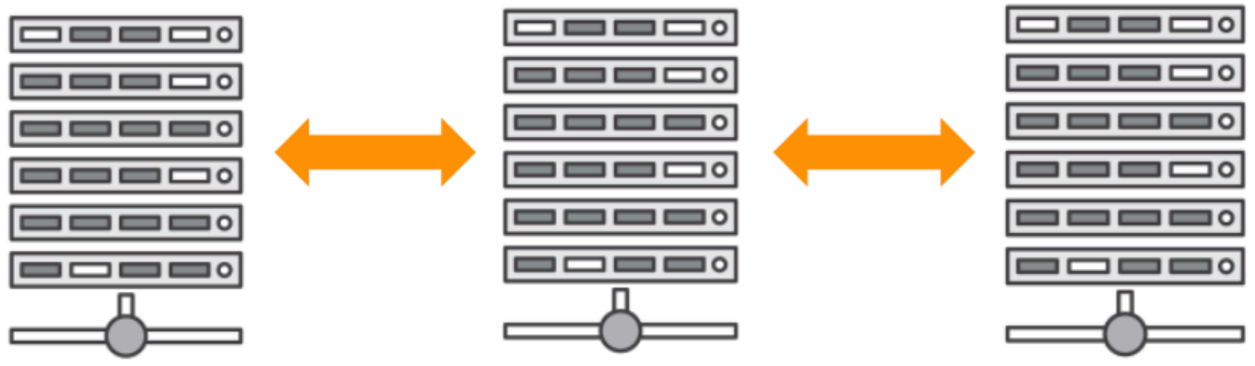
4 Steps

- Crawl and catalogue your data
- Specify mappings to generate scripts
- Interactively edit and explore with dev-endpoints
- Schedule a job for running in production

Server-less job execution

- No need to provision, configure, or manage servers
- Auto-configure VPC & role-based access security & isolation preserved
- Customers can specify job capacity (DPUs)
  - 1 DPU = 16GB x 4 vCPU
- Automatically scale resources
- Only pay for the resources you consume per-second billing (10-minute min)

Under the hood



SparkSQL	AWS Glue ETL
Dataframes	Dynamic Frames
Spark core: RDDs	

MyTEDx

Iniziamo!

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

AWS Management Console

TED

AWS Management Console

AWS services

Find Services
You can enter names, keywords or acronyms.

Example: Relational Database Service, database, RDS

▼ **Recently visited services**

IAM
S3
AWS Glue
Step Functions
CloudWatch

▶ **All services**

Build a solution

Get started with simple wizards and automated workflows.

Launch a virtual machine
With EC2
2-3 minutes

Build a web app
With Elastic Beanstalk
6 minutes

Build using virtual servers
With Lightsail
1-2 minutes

Access resources on the go

Access the Management Console using the AWS Console Mobile App. Learn more

Explore AWS

Amazon Redshift

Fast, simple, cost-effective data warehouse that can extend queries to your data lake. Learn more

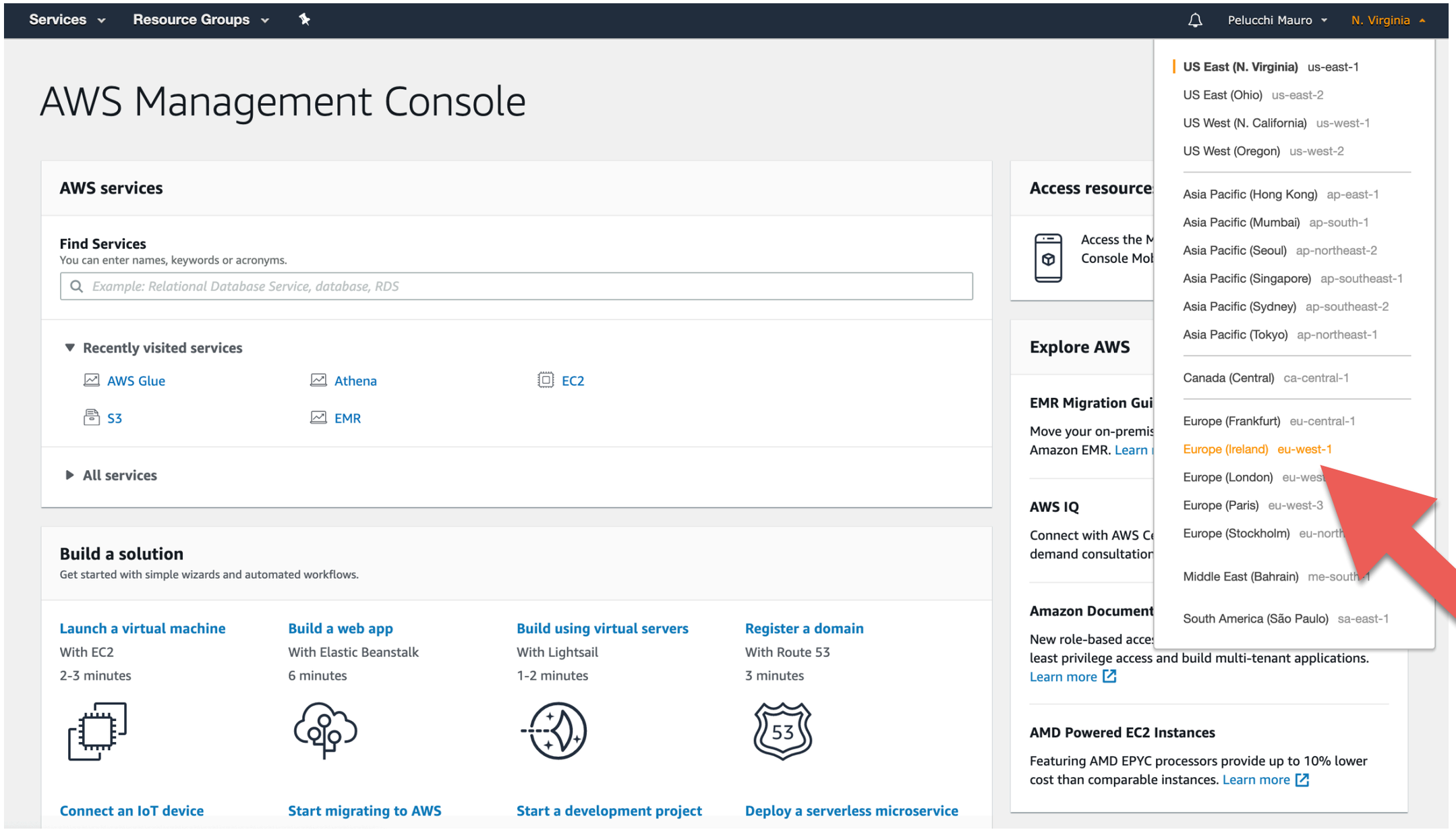
Run Serverless Containers with AWS Fargate

AWS Fargate runs and scales your containers without having to manage servers or clusters. Learn more

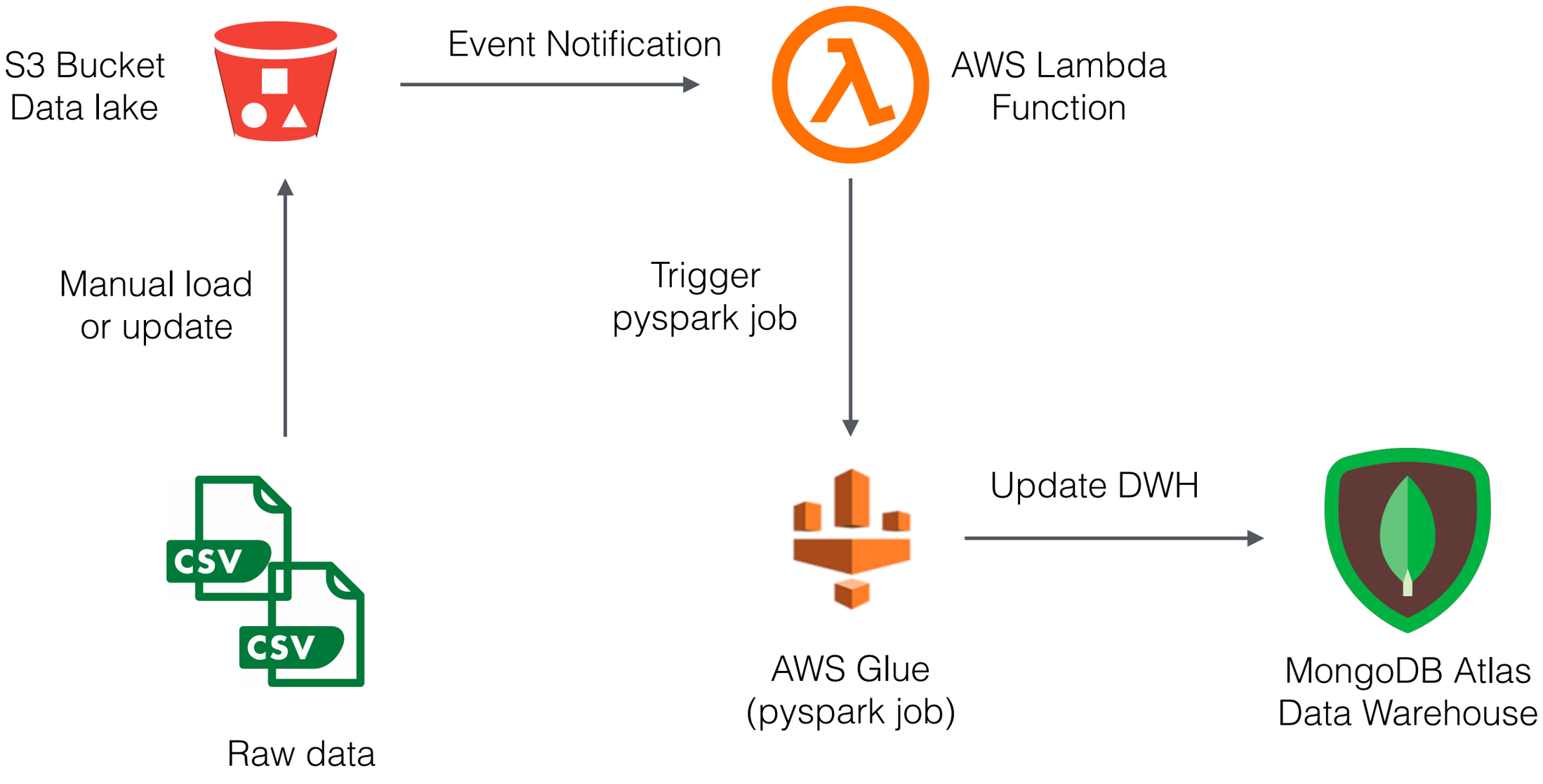
Scalable, Durable, Secure Backup & Restore with Amazon S3

Discover how customers are building backup & restore solutions on AWS that save money.

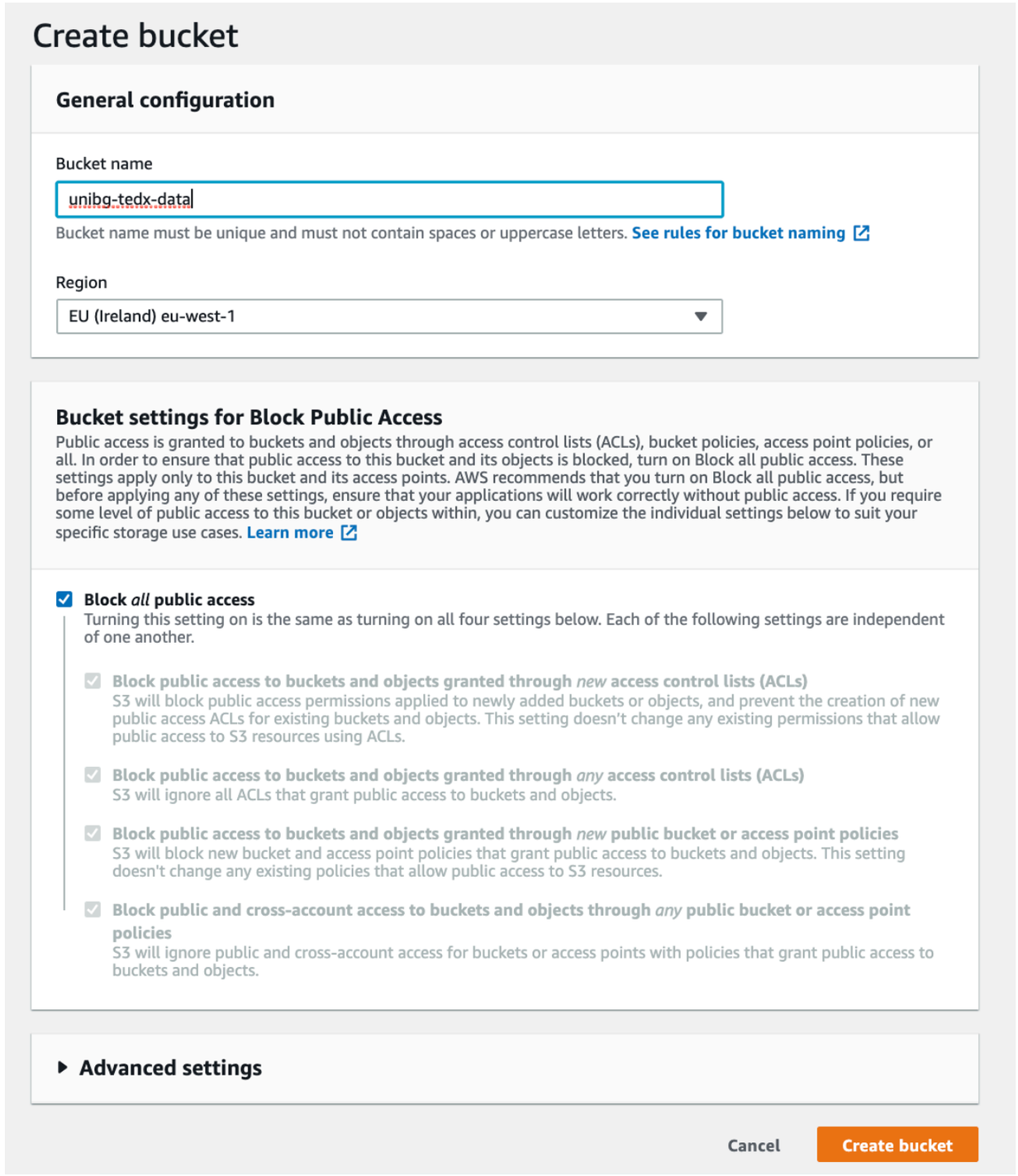
Check your zone



Main Architecture



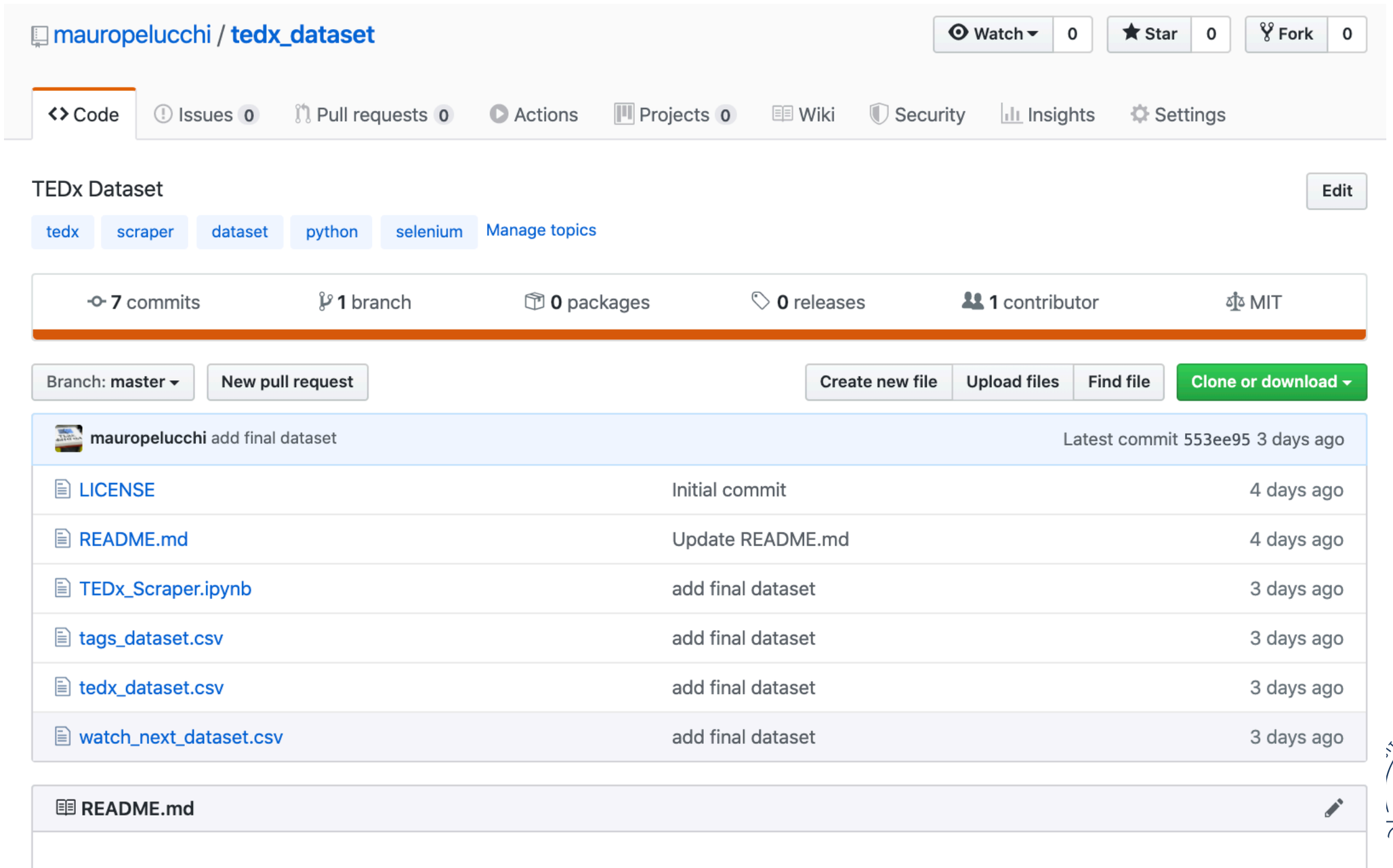
Data lake



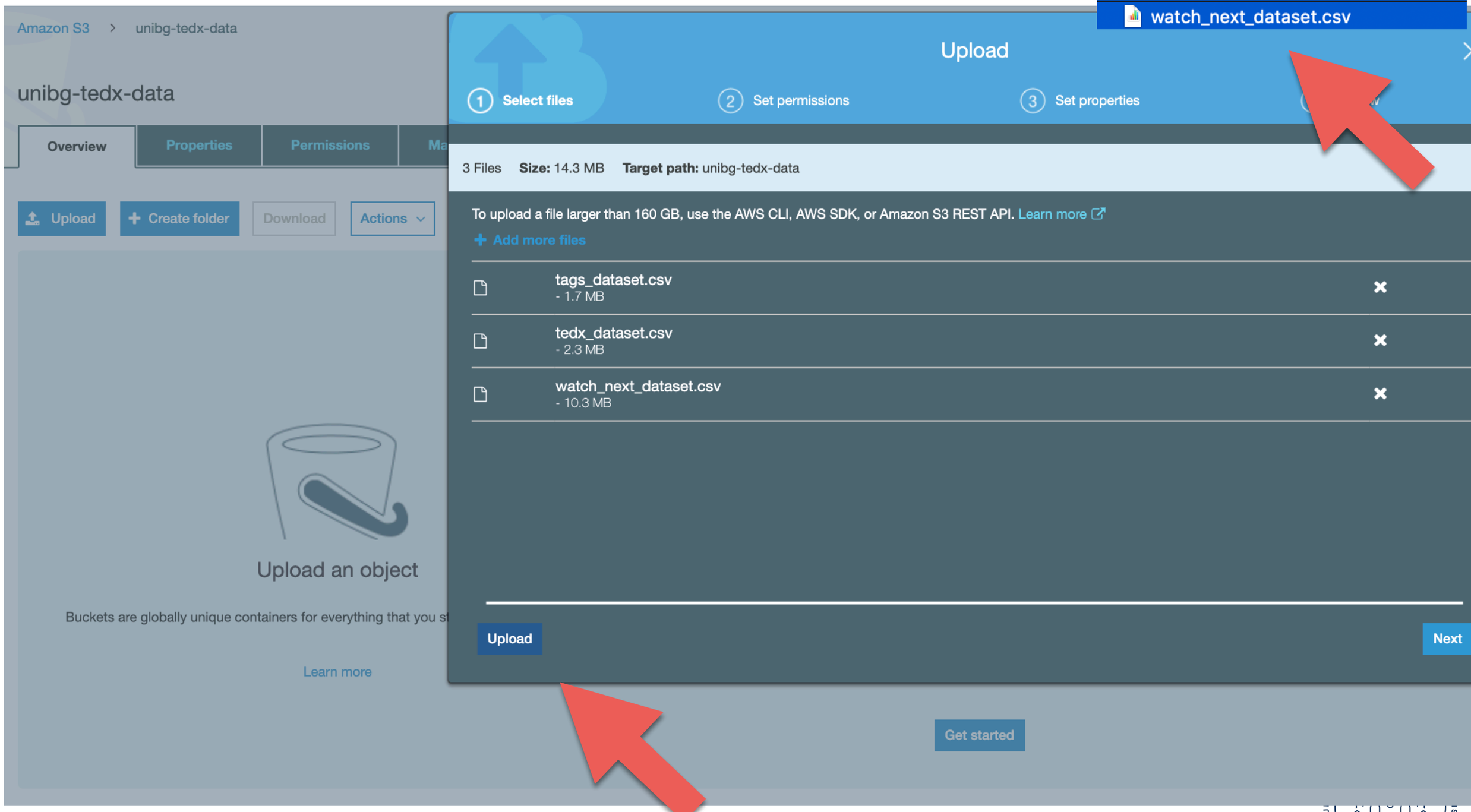
Create a bucket
for our data lake,
in the bucket we will put
our raw data

Download raw data from GitHub

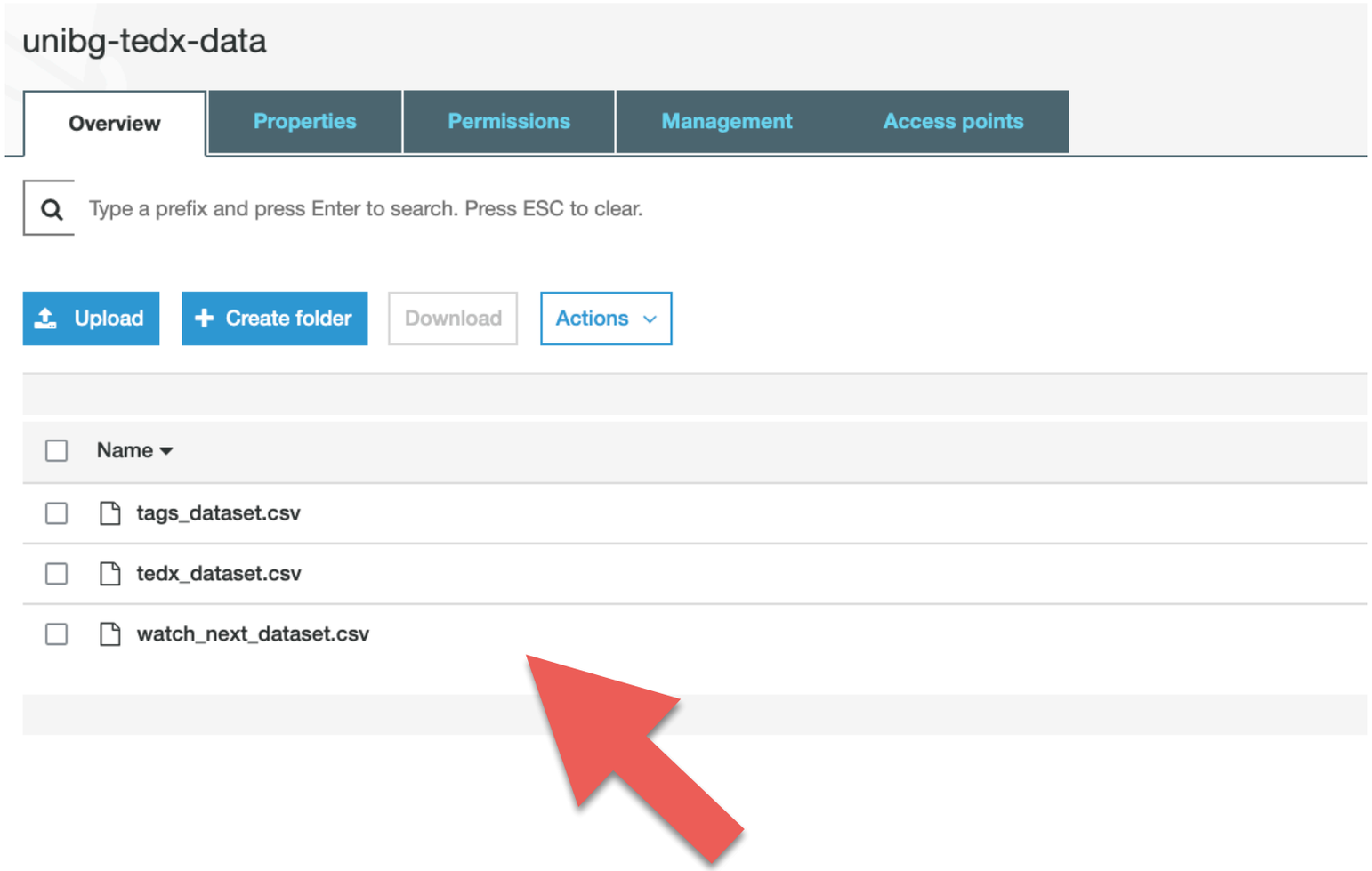
https://github.com/mauropelucchi/tedx_dataset



Data lake

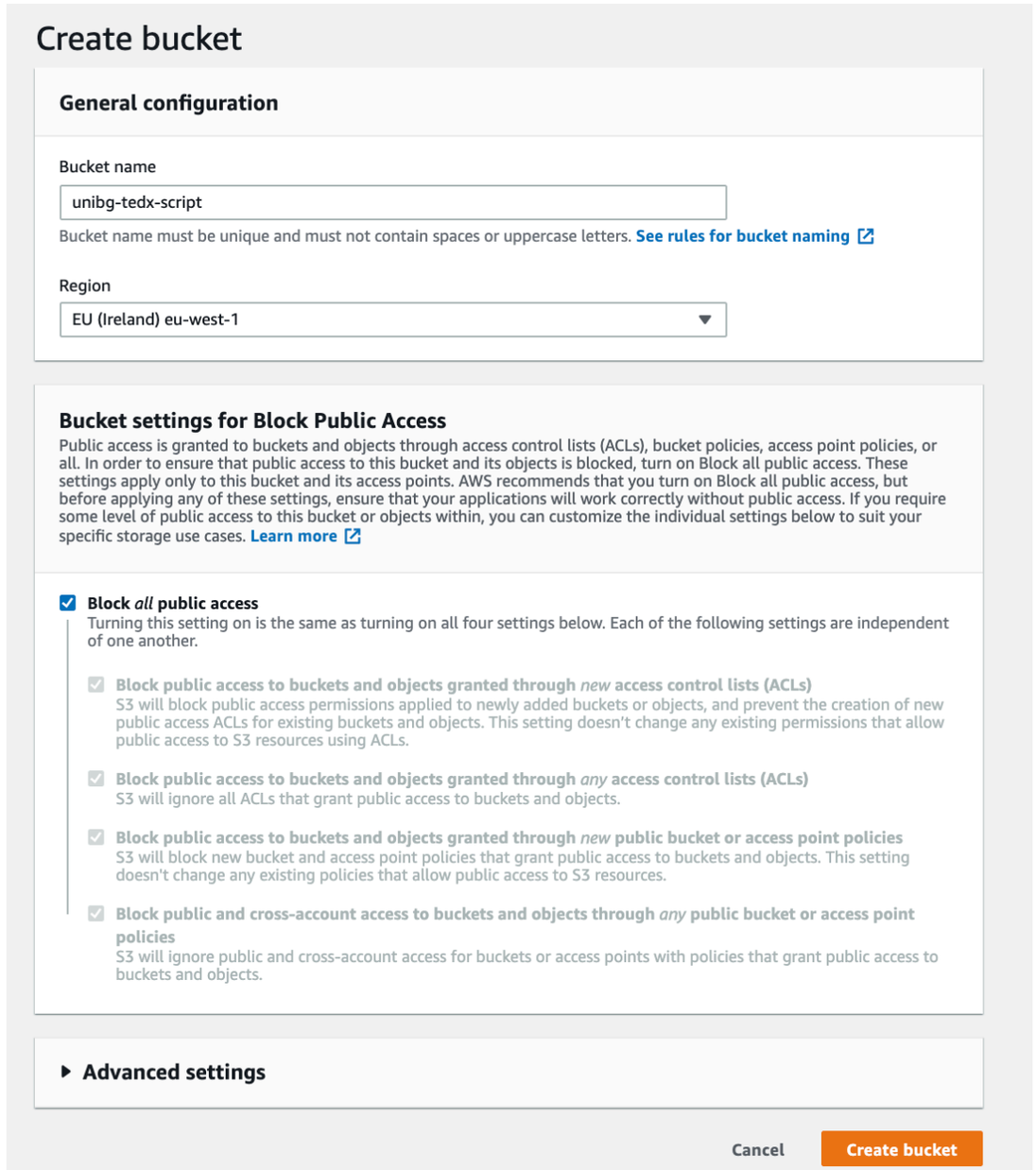

tags_dataset.csv
tedx_dataset.csv
watch_next_dataset.csv
Amazon S3 > unibg-tedx-data
unibg-tedx-data
Overview
Properties
Permissions
Upload
Create folder
Download
Actions
Upload an object
Buckets are globally unique containers for everything that you st
Learn more
Upload
1 Select files
2 Set permissions
3 Set properties
3 Files Size: 14.3 MB Target path: unibg-tedx-data
To upload a file larger than 160 GB, use the AWS CLI, AWS SDK, or Amazon S3 REST API. Learn more
Add more files
tags_dataset.csv
- 1.7 MB
tedx_dataset.csv
- 2.3 MB
watch_next_dataset.csv
- 10.3 MB
Upload
Next
Get started

Data lake on S3



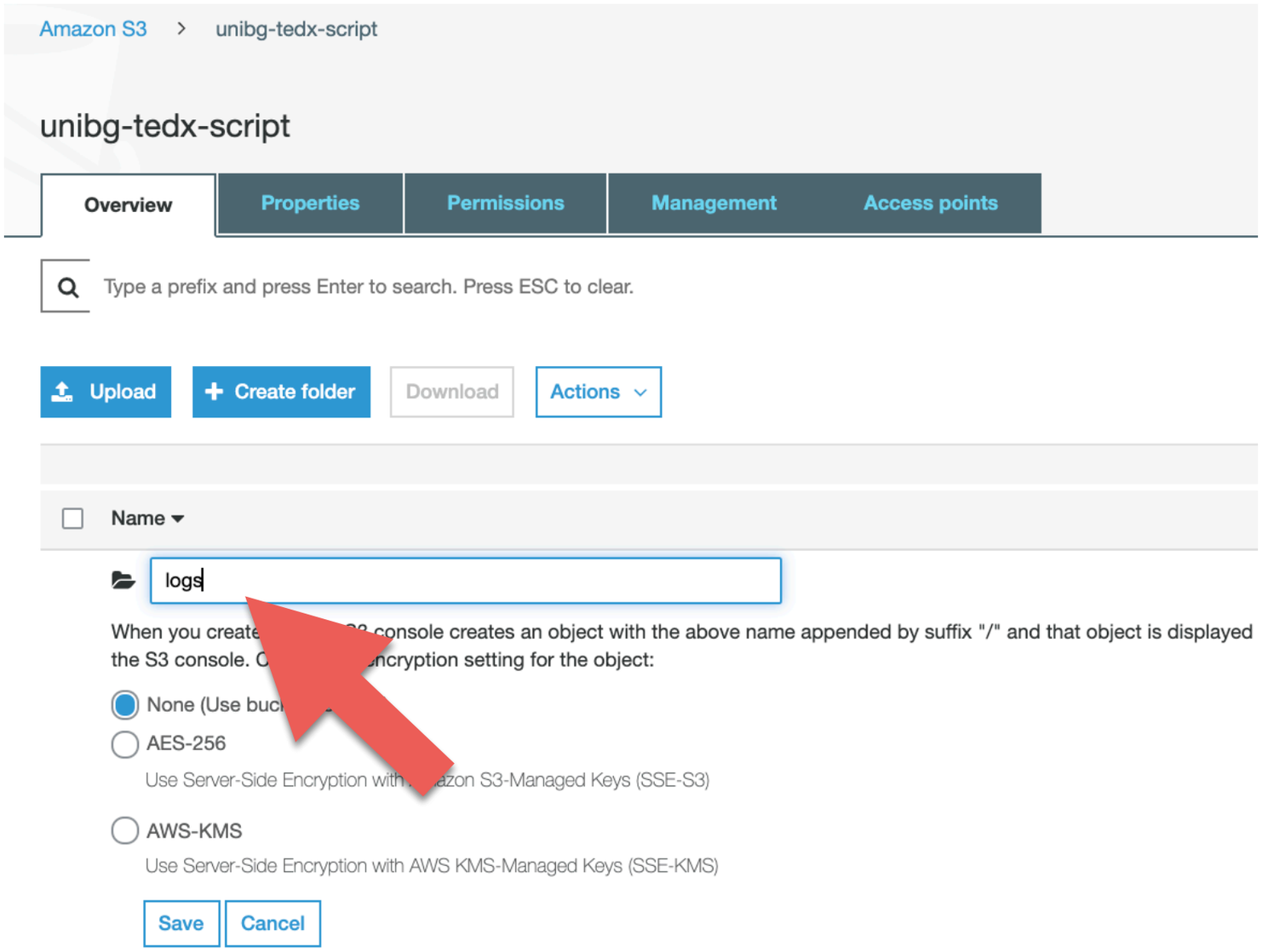
S3 Bucket for Scripts and logs

Let’s create a bucket to store scripts, logs, …
unibg-tedx-script

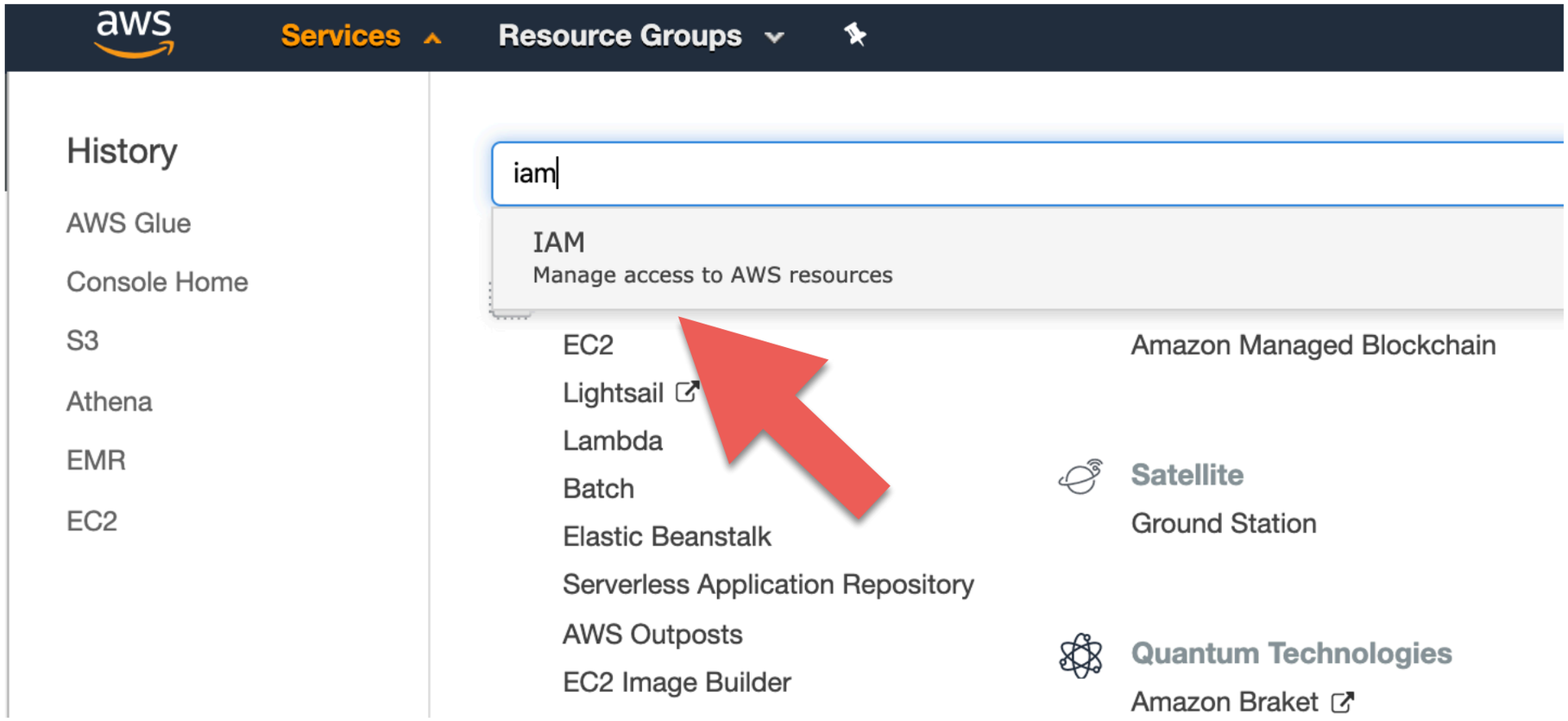


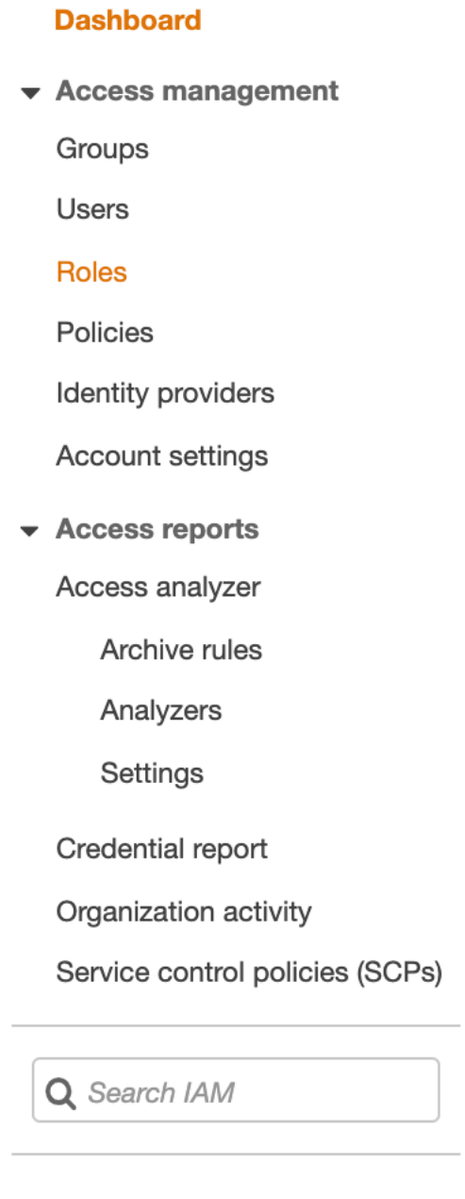
S3 Bucket for Scripts and logs

Create a “logs” folder…



Manage Access on AWS resources





Manage Access on AWS resources —> Roles

Identity and Access Management (IAM)

- Dashboard
- Access management
  - Groups
  - Users
  - Roles
  - Policies
  - Identity pr
  - Account set gs
- Access reports
  - Access analyzer
    - Archive rules
    - Analyzers
    - Settings
  - Credential report
  - Organization activity
  - Service control policies (SCPs)

Search IAM

AWS account ID:
386545469924

Roles

What are IAM roles?

IAM roles are a secure way to grant permissions to entities that you trust. Examples of entities include the following:

- IAM user in another account
- Application code running on an EC2 instance that needs to perform actions on AWS resources
- An AWS service that needs to act on resources in your account to provide its features
- Users from a corporate directory who use identity federation with SAML

IAM roles issue keys that are valid for short durations, making them a more secure way to grant access.

Additional resources:

- IAM Roles FAQ
- IAM Roles Documentation
- Tutorial: Setting Up Cross Account Access
- Common Scenarios for Roles

Create role | Delete role

Search

Role name	Trusted entities	Last activity
AWSServiceRoleForSupport	**AWS service:** support (Service-Linked role)	None
AWSServiceRoleForTrustedAdvisor	**AWS service:** trustedadvisor (Service-Linked ...	None

Manage Access on AWS resources —> Roles

- Now we create a role to execute AWS Glue Job

AWS Glue Role

TED

AWS service
EC2, Lambda and others

Another AWS account
Belonging to you or 3rd party

Web identity
Cognito or any OpenID provider

SAML 2.0 federation
Your corporate directory

Allows AWS services to perform actions on your behalf. Learn more

Choose a use case

Common use cases

EC2
Allows EC2 instances to call AWS services on your behalf.

Lambda
Allows Lambda functions to call AWS services on your behalf.

Or select a service to view its use cases

API Gateway	CodeDeploy	EMR	KMS	RoboMaker
AWS Backup	CodeGuru	ElastiCache	Kinesis	S3
AWS Chatbot	CodeStar Notifications	Elastic Beanstalk	Lambda	SMS
AWS Support	Comprehend	Elastic Container Service	Lex	SNS
Amplify	Config	Elastic Transcoder	License Manager	SWF
AppStream 2.0	Connect	ElasticLoadBalancing	Machine Learning	SageMaker
AppSync	DMS	Forecast	Macie	Security Hub
Application Auto Scaling	Data Lifecycle Manager	Global Accelerator	MediaConvert	Service Catalog
Application Discovery Service	Data Pipeline	Glue	Migration Hub	Step Functions
Batch	DataSync	Greengrass	OpsWorks	Storage Gateway
Chime	DeepLens	GuardDuty	Personalize	Textract
CloudFormation	Directory Service	Health Organizational Vie	QLDB	Transfer

STUDIORUM BERGOMENSIS

AWS Glue Role

TED

Create role

1 2 3 4

Attach permissions policies

Choose one or more policies to attach to your new role.

Create policy

Filter policies | glue | Showing 4 results

	Policy name	Used as
☑	AWSGlueConsoleFullAccess	None
☑	AWSGlueConsoleSageMakerNotebookFullAccess	None
☑	AWSGlueServiceNotebookRole	None
☑	AWSGlueServiceRole	None

Create role

1 2 3 4

Add tags (optional)

IAM tags are key-value pairs you can add to your role. Tags can include user information, such as an email address, or can be descriptive, such as a job title. You can use the tags to organize, track, or control access for this role. Learn more

Key	Value (optional)	Remove
Project	MyTEDx	✖
Add new key		

You can add 49 more tags.

AWS Glue Role

Create role

1 2 3 4

Review

Provide the required information below and review this role before you create it.

Role name* Unibg-TEDxGlueRole

Use alphanumeric and '+=,.@-_' characters. Maximum 64 characters.

Role description Allows Glue to call AWS services on your behalf.

Maximum 1000 characters. Use alphanumeric and '+=,.@-_' characters.

Trusted entities AWS service: glue.amazonaws.com

Policies
- AWSGlueConsoleFullAccess
- AWSGlueServiceRole
- AWSGlueServiceNotebookRole
- AWSGlueConsoleSageMakerNotebookFullAccess

Permissions boundary Permissions boundary is not set

The new role will receive the following tag

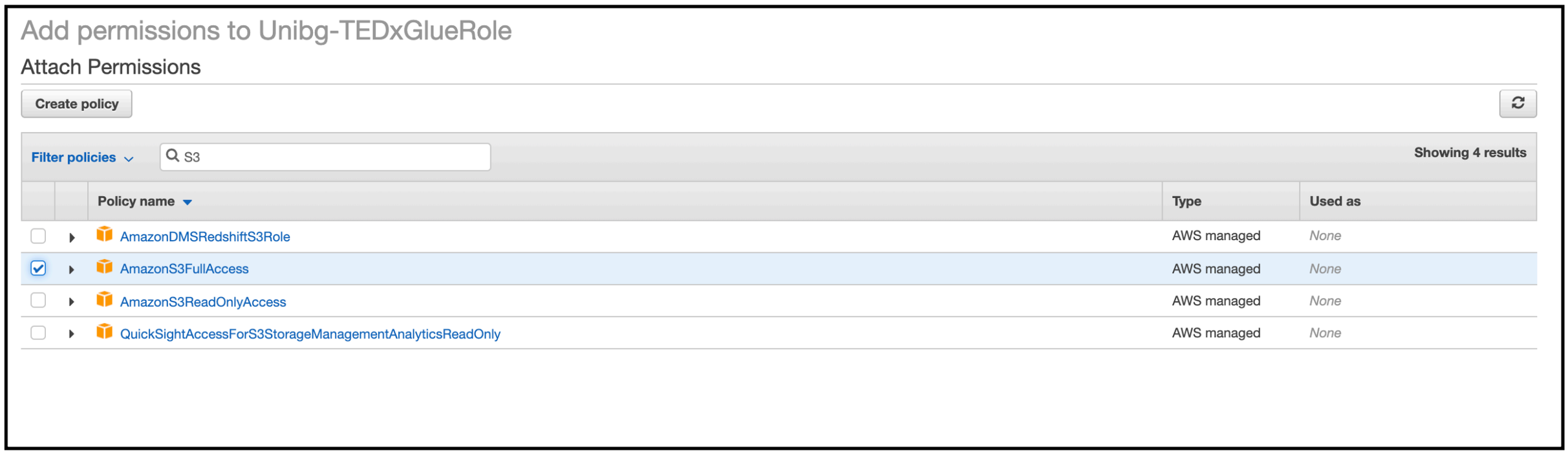
Key	Value
Project	MyTEDx

* Required

Cancel Previous Create role

AWS Glue Role

Edit the Unibg-TEDx-Role and add AmazonS3FullAccess



AWS Glue

TED

AWS services

Find Services

You can enter names, keywords or acronyms.

glue

AWS Glue
AWS Glue is a fully m aged ETL (extract, transform, and load) service

AWS Lake Formatio
AWS Lake Formation m cy to set up a secure data lake

S3 EMR

All services

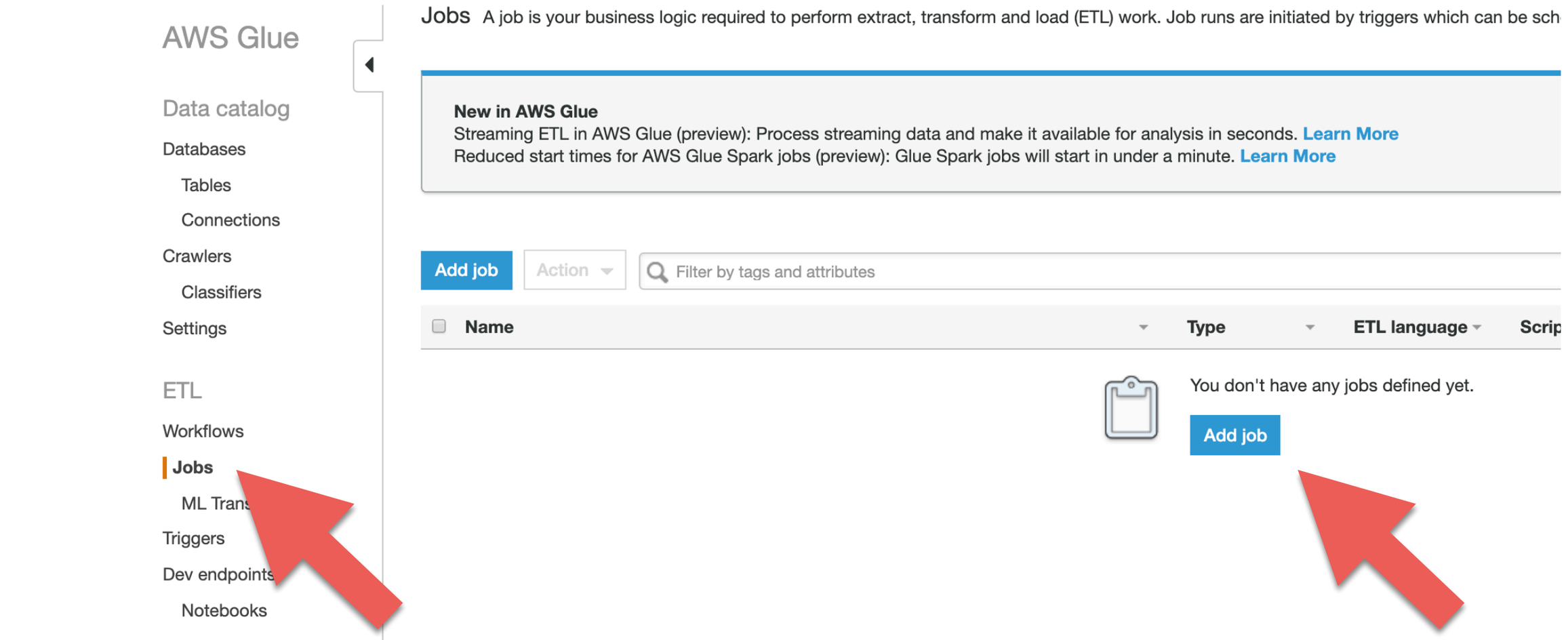
AWS Glue

AWS Glue is a fully managed ETL (extract, transform, and load) service that makes it simple and cost-effective to categorize your data, clean it, enrich it, and move it reliably between various data stores.

Get started

Getting started guide

AWS Glue
Add a new JOB



We have to create a new job
to read the raw data and populate our NoSql DWH

Configure the job properties

Configure the job properties

Name

TEDx-Load-Aggregate-Model

IAM role

Unibg-TEDxGlueRole

Ensure that this role has permissio… …azon S3 sources, targets, temporary directory, scripts, and any libraries used by the job. Create IAM role.

Type

Spark

Glue version

Spark 2.4, Python 3 (Glue Version 1.0)

This job runs

- A proposed script generated by AWS Glue
- An existing script that you provide
- A new script to be authored by you

Script file name

TEDx-Load-Aggregate-Model

S3 path where the script is stored

s3://unibg-tedx-script

Temporary directory

s3://unibg-tedx-script/logs

Save and edit script

Connections

Choose connections required by this job. These connections are used to set up access to your data and must match connections referenced in the

Showing: 0 - 0

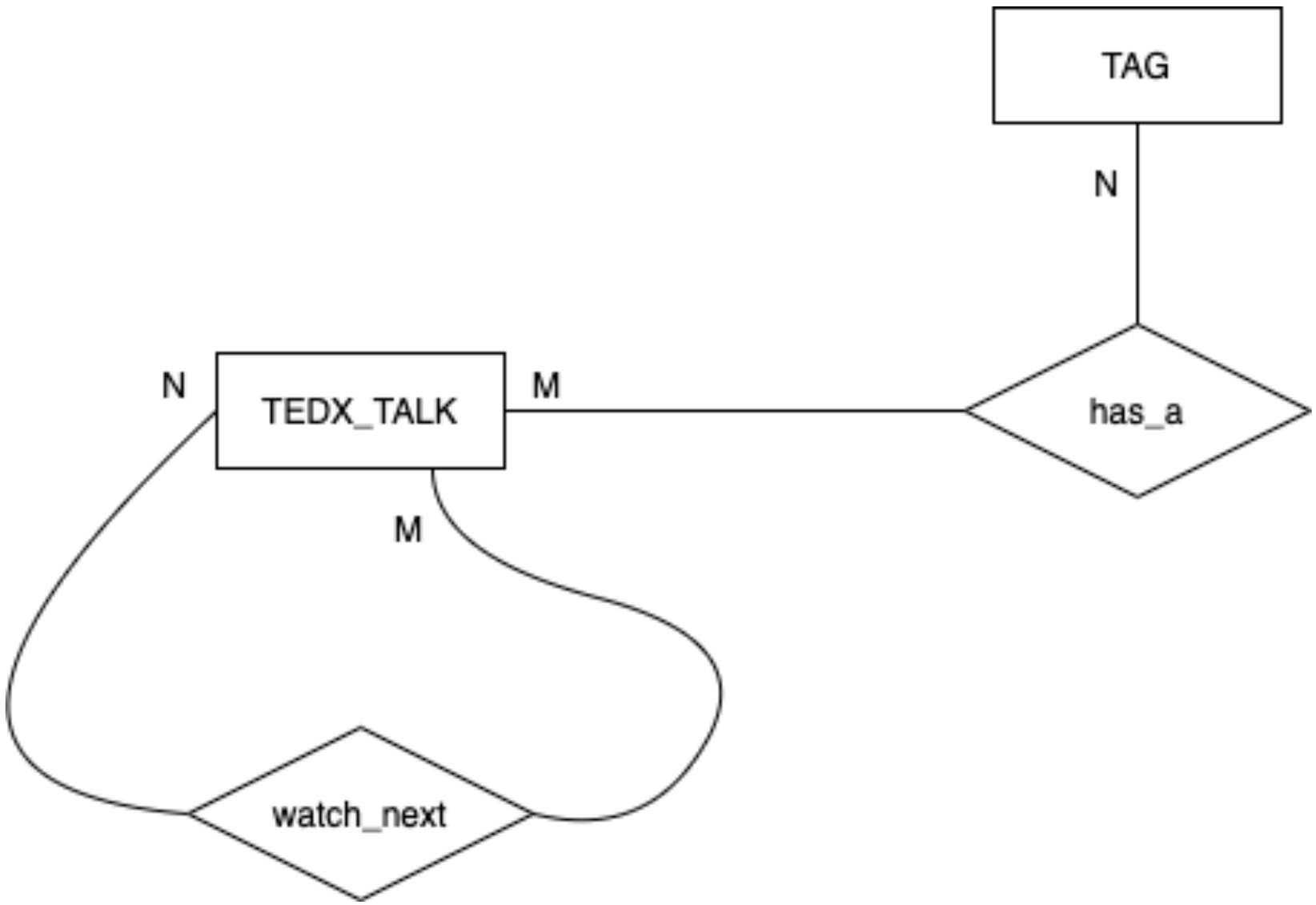
All connections	Required connections
No items available	No

Add connection

Back

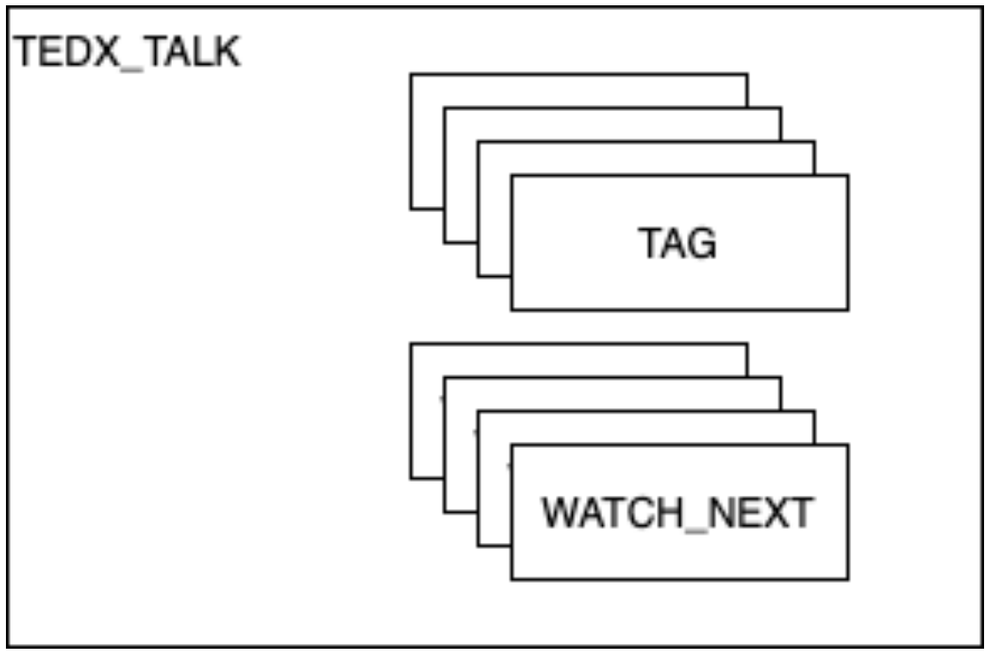
Save job and edit script

ER Model



Come può essere secondo voi un modello aggregato?

Aggregate model



Pyspark Job

- Import modules
- Read talks dataset —> tedx_dataset
- Init job
- Read tags dataset —> tags_dataset
- Group by tags by talk —> tedx_dataset_agg
- Join tedx_dataset with tedx_dataset_agg
- Store data on MongoDB

Import modules

```
###### TEDx-Load-Aggregate-Model
######

import sys
import json
import pyspark
from pyspark.sql.functions import col, collect_list, array_join

from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
```


Read talks dataset
tedx_dataset


```
##### FROM FILES
tedx_dataset_path = "s3://unibg-tedx-data/tedx_dataset.csv"

###### READ PARAMETERS
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

##### START JOB CONTEXT AND JOB
sc = SparkContext()

glueContext = GlueContext(sc)
spark = glueContext.spark_session



job = Job(glueContext)
job.init(args['JOB_NAME'], args)
```

Init job

Read talks dataset
tedx_dataset


```
#### READ INPUT FILES TO CREATE AN INPUT DATASET
tedx_dataset = spark.read \
    .option("header","true") \
    .option("quote", "\"") \
    .option("escape", "\"") \
    .csv(tedx_dataset_path)

tedx_dataset.printSchema()
```

Read the CSV file

```
#### FILTER ITEMS WITH NULL POSTING KEY
count_items = tedx_dataset.count()
count_items_null = tedx_dataset.filter("idx is not null").count()

print(f"Number of items from RAW DATA {count_items}")
print(f"Number of items from RAW DATA with NOT NULL KEY {count_items_null}")
```

Check & filter record with idx is null

Read tags dataset
tags_dataset


```
## READ TAGS DATASET
tags_dataset_path = "s3://unibg-tedx-data/tags_dataset.csv"
tags_dataset = spark.read.option("header","true").csv(tags_dataset_path)
```


Group by tags by talk
tedx_dataset_agg

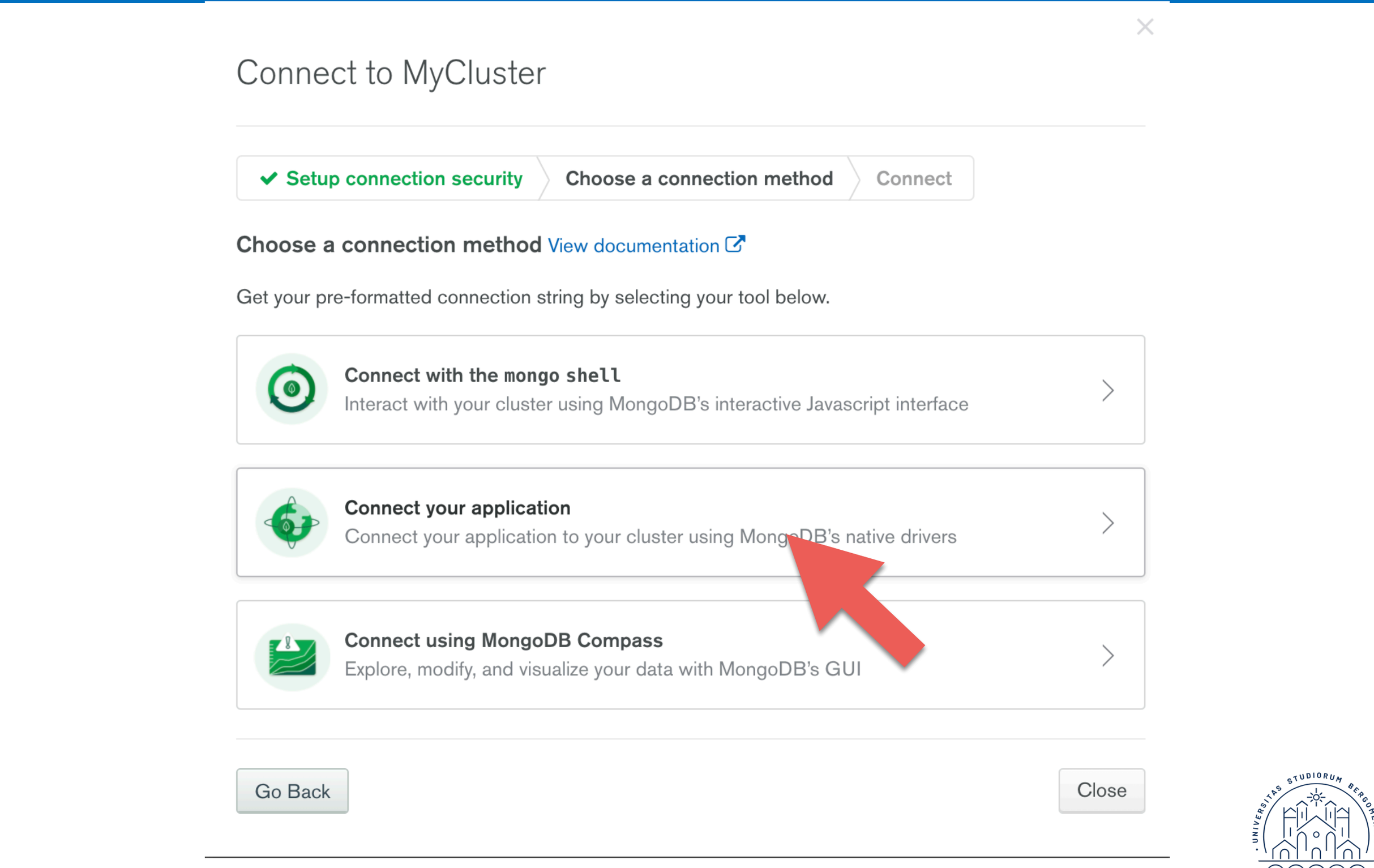

```
# CREATE THE AGGREGATE MODEL, ADD TAGS TO TEDX_DATASET
tags_dataset_agg = tags_dataset.groupBy(col("idx").alias("idx_ref")).agg(collect_list("tag").alias("tags"))
tags_dataset_agg.printSchema()
tedx_dataset_agg = tedx_dataset.join(tags_dataset_agg, tedx_dataset.idx == tags_dataset_agg.idx_ref, "left") \
    .drop("idx_ref") \
    .select(col("idx").alias("_id"), col("*")) \
    .drop("idx") \

tedx_dataset_agg.printSchema()
```

Join the 2 datasets

Store data on MongoDB


```
mongo_uri = "mongodb://mycluster-shard-00-00-250eh.mongodb.net:27017,mycluster-shard-00-01-250eh.mongodb.net:27017,mycluster-shard-00-02-250eh.mongodb.net:27017"

write_mongo_options = {
    "uri": mongo_uri,
    "database": "unibg_tedx",
    "collection": "tedz_data",
    "username": "admin",
    "password": "VFT0TMyEbfu2CJh0",
    "ssl": "true",
    "ssl.domain_match": "false"}
from awsglue.dynamicframe import DynamicFrame
tedx_dataset_dynamic_frame = DynamicFrame.fromDF(tedx_dataset_agg, glueContext, "nested")

glueContext.write_dynamic_frame.from_options(tedx_dataset_dynamic_frame, connection_type="mongodb", connection_options=write_mongo_options)
```


MongoDB Atlas
How Connect to MyCluster



MongoDB Atlas
How Connect to MyCluster

Connect to MyCluster

✔ Setup connection security › ✔ Choose a connection method › Connect

1 Select your driver and version

DRIVER: Scala — VERSION: 2.2 or later

2 Add your connection string into your application code

Connection String Only | Full Driver Example

```
mongodb+srv://admin:<password>@mycluster-250eh.mongodb.net/test?r
```

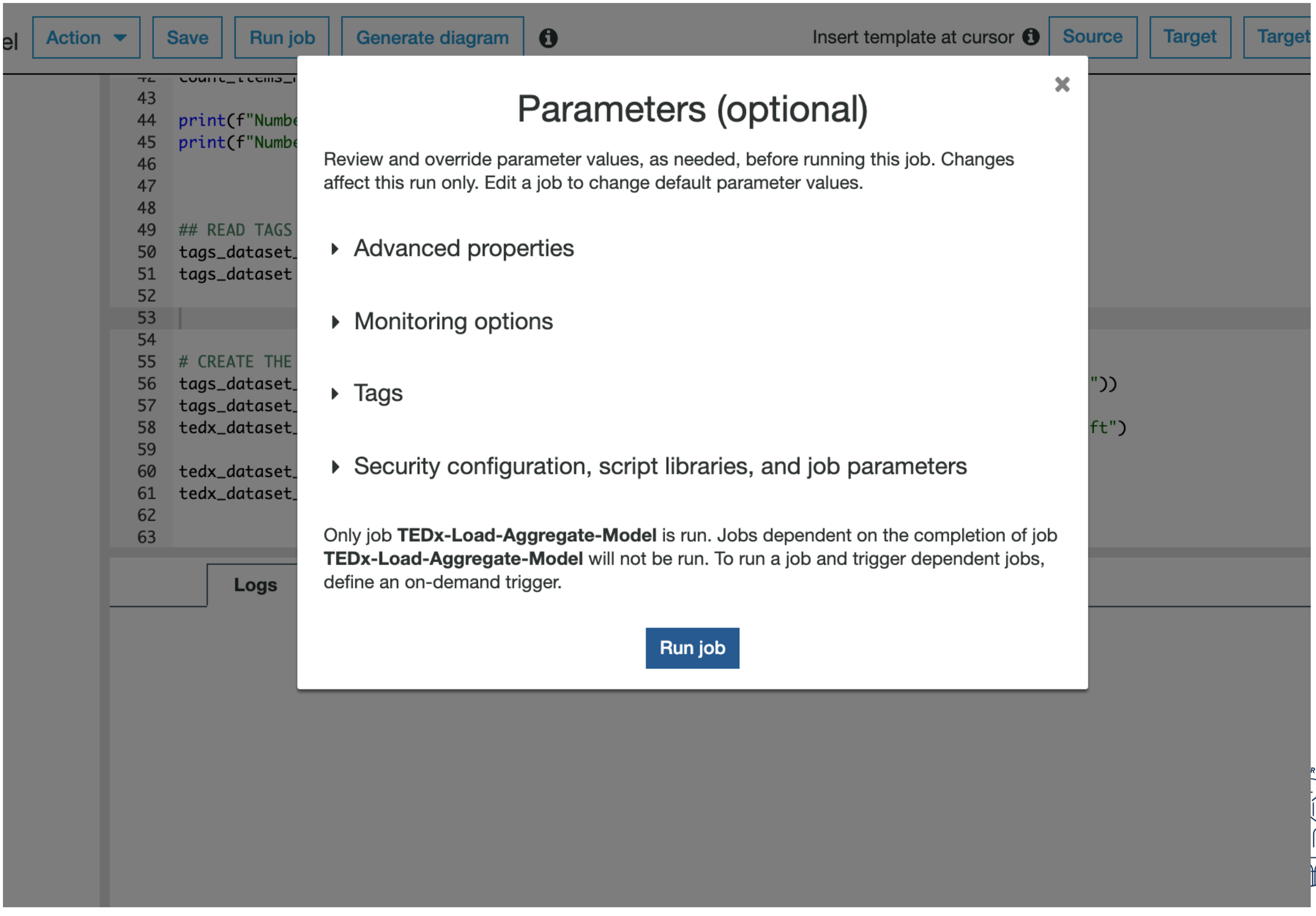
Copy

Replace **<password>** with the password for the user, **admin**, and ensure all special characters are URL encoded.

Having trouble connecting? View our troubleshooting documentation

Go Back | Close

Run your job


Action
Save
Run job
Generate diagram
Insert template at cursor
Source
Target
Target
Parameters (optional)
Review and override parameter values, as needed, before running this job. Changes affect this run only. Edit a job to change default parameter values.
Advanced properties
Monitoring options
Tags
Security configuration, script libraries, and job parameters
Only job TEDx-Load-Aggregate-Model is run. Jobs dependent on the completion of job TEDx-Load-Aggregate-Model will not be run. To run a job and trigger dependent jobs, define an on-demand trigger.
Run job
Logs

Job History

Add job | Action | Filter by tags and attributes

Name
TEDx-Load-Aggregate-Model

History | Details | Script | Metrics

View run metrics | Rewind job bookmark

Run ID	Retry attempt	Run status		Logs	Error logs
jr_deb6e9f845dd…	-	Succeeded		Logs	
jr_cbc1d4a1a07c…	-	Failed	Sy…	Logs	Error logs

Check result on your DWH

MyCluster

Overview | Real Time | Metrics | **Collections** | Profiler | Performance Advisor | Command Line Tools

DATABASES: 1 COLLECTIONS: 1

+ Create Database

NAMESPACES

- unibg_tedx
  - tedz_data

unibg_tedx.tedz_data

COLLECTION SIZE: 3.3MB TOTAL DOCUMENTS: 4494 INDEXES TOTAL SIZE: 208KB

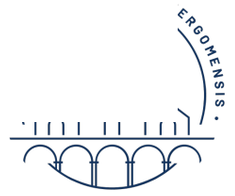
Find | Indexes | Aggregation | Search BETA

FILTER {"filter":"example"}

QUERY RESULTS 1-20 OF MANY


```
_id: "8d2005ec35280deb6a438dc87b225f89"
main_speaker: "Alexandra Auer"
title: "The intangible effects of walls"
details: "More barriers exist now than at the end of World War II, says designer..."
posted: "Posted Apr 2020"
url: "https://www.ted.com/talks/alexandra_auer_the_intangible_effects_of_wal..."
› tags: Array
```

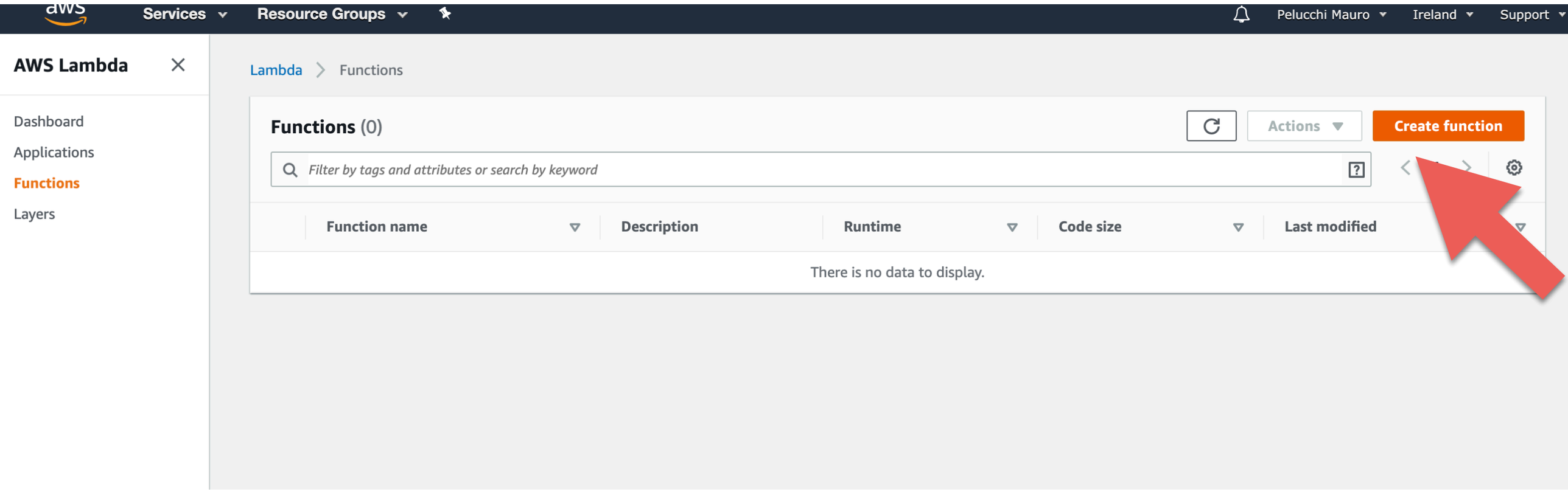
```
_id: "b3072cd11f40eb57fd259555264476c6"
main_speaker: "Elizabeth Gilbert"
title: "It's OK to feel overwhelmed. Here's what to do next"
details: "If you're feeling anxious or fearful during the coronavirus pandemic, ..."
posted: "Posted Apr 2020"
url: "https://www.ted.com/talks/elizabeth_gilbert_it_s_ok_to_feel_overwhelme..."
› tags: Array
```

AWS Lambda Functions

TED

Create a new Lambda function



Create a new Lambda function

Basic information

Function name
Enter a name that describes the purpose of your function.

Trigger_Refresh_DWH1

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime Info
Choose the language to use to write your function.

Python 3.8

Permissions Info
Lambda will create an execution role wit… …o upload logs to Amazon CloudWatch Logs. You can configure and modify permissions further when you add triggers.

▼ Choose or create an execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.

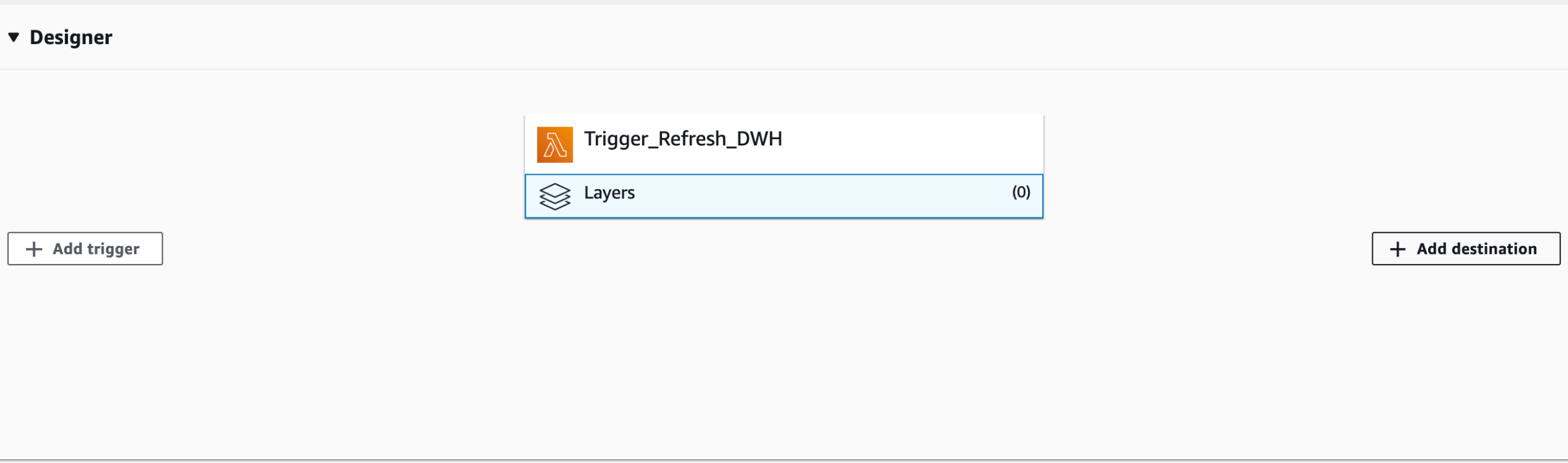
- (•) Create a new role with basic Lambda permissions
- ( ) Use an existing role
- ( ) Create a ne… …m AWS policy templates

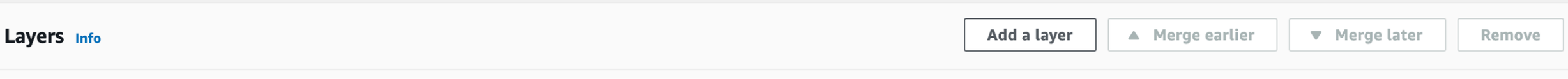
Role creati… …ni… …e a few minutes. Please do not delete the role or edit the trust or permissions policies in this role.

Lambda will create an execution role named Trigger_Refresh_DWH1-role-4ryv1as3, with permission to upload logs to Amazon CloudWatch Logs.

Cancel | Create function

Create a new Lambda function





Call AWS Glue Job From lambda function


```
# Set up logging
import json
import os
import logging
logger = logging.getLogger()
logger.setLevel(logging.INFO)

# Import Boto 3 for AWS Glue
import boto3
client = boto3.client('glue')

# Variables for the job:
glueJobName = "TEDx-Load-Aggregate-Model"

# Define Lambda function
def lambda_handler(event, context):

    response = client.start_job_run(JobName = glueJobName)
    logger.info('## STARTED GLUE JOB: ' + glueJobName)
    logger.info('## GLUE JOB RUN ID: ' + response['JobRunId'])
    return response
```


Call AWS Glue Job From lambda function

TED

Function code Info

Code entry type: Edit code inline

Runtime: Python 3.8

Handler Info: index.lambda_handler

File Edit Find View Go Tools Window Save Test

Environment

Trigger_Refresh_DV

index.py

index.py

```
1  # Set up logging
2  import json
3  import os
4  import logging
5  logger = logging.getLogger()
6  logger.setLevel(logging.INFO)
7
8  # Import Boto 3 for AWS Glue
9  import boto3
10 client = boto3.client('glue')
11
12 # Variables for the job:
13 glueJobName = "TEDx-Load-Aggregate-Model"
14
15 # Define Lambda function
16 def lambda_handler(event, context):
17
18     response = client.star                (JobName = glueJobName)
19     logger.info('## STARTED              ' + glueJobName)
20     logger.info('## GLUE JOB             ' + response['JobRunId'])
21     return response
```

Enable Events on Our S3 Data Lake

unibg-tedx-data

Overview | Properties | Permissions | Management | Access points

Versioning
Keep multiple versions of an object in the same bucket.
Learn more
Disabled

Server access logging
Set up access log records that provide details about access requests.
Learn more
Disabled

Static website hosting
Host a static website, which does not require server-side technologies.
Learn more
Disabled

Object-level logging
Record object-level API activity using the CloudTrail data events feature (additional cost).
Learn more
Disabled

Default encryption
Automatically encrypt objects when stored in Amazon S3
Learn more
Disabled

Advanced settings

Object lock
Prevent objects from being deleted.
Learn more
Disabled

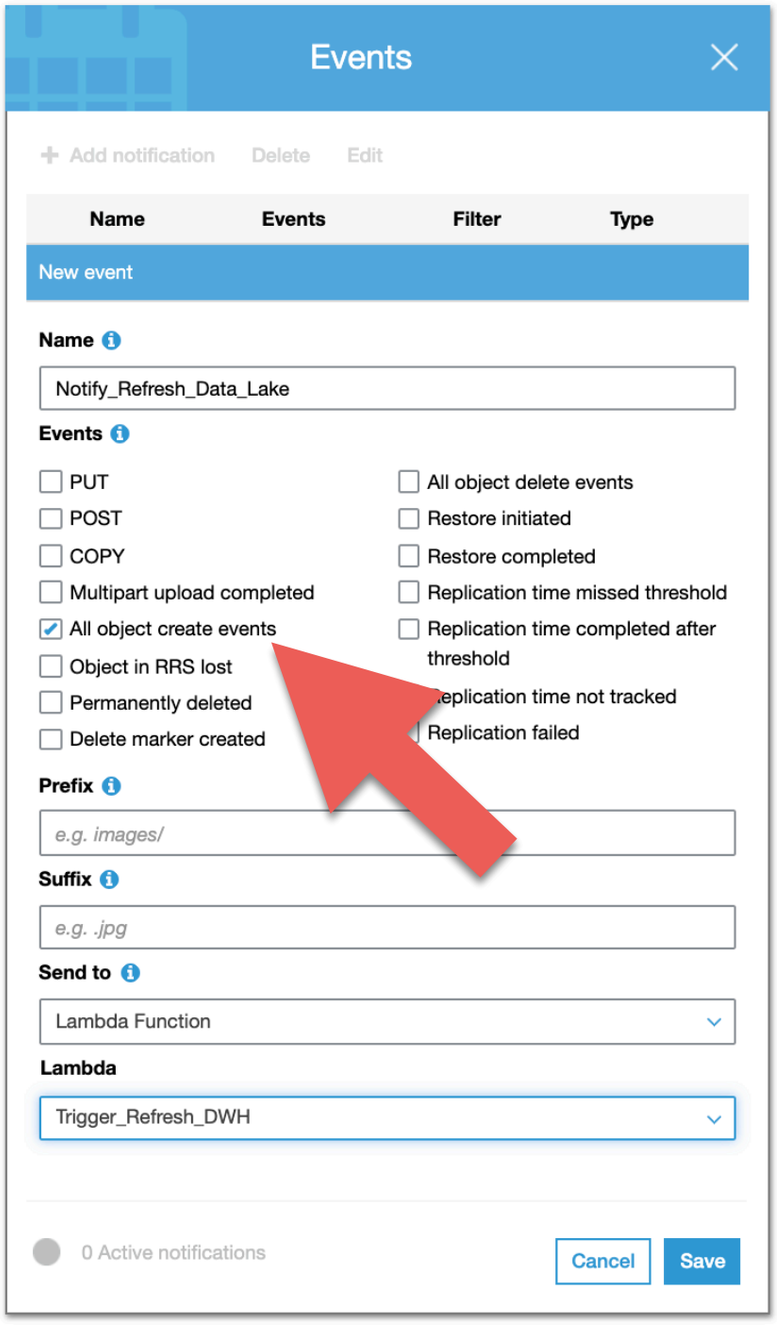
Tags
Use tags to track your cost against projects or other criteria.
Learn more
0 Tags

Transfer acceleration
Enable fast, easy and secure transfers of files to and from your bucket.
Learn more
Suspended

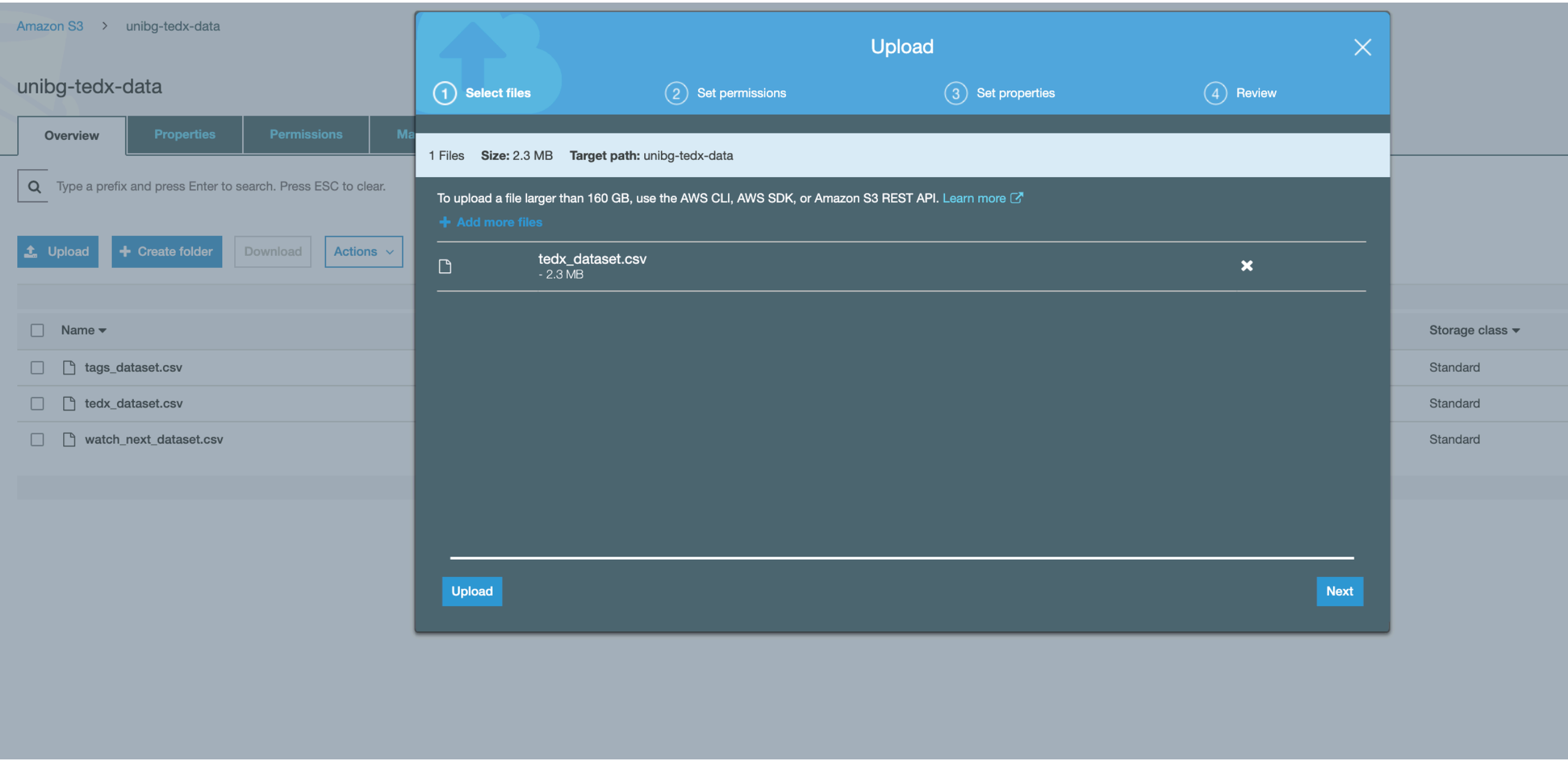
Events
Receive notifications when specific events occur in your bucket.
Learn more
0 Active notifications

Requester pays
The requester (instead of the bucket owner) will pay for requests and data transfer.
Learn more
Disabled

Enable Events on Our S3 Data Lake



Try to add a new file to the bucket



Perché la lambda function fallisce?

Manage Access on AWS resources —> Roles

Roles > Trigger_Refresh_DWH-role-y67bgjru

Summary

Delete role

Role ARN	arn:aws:iam::386545469924:role/service-role/Trigger_Refresh_DWH-role-y67bgjru
Role description	Edit
Instance Profile ARNs	
Path	/service-role/
Creation time	2020-04-10 13:53 UTC+0200
Last activity	Not accessed in the tracking period
Maximum CLI/API session duration	1 hour Edit

Permissions | **Trust relationships** | **Tags** | **Access Advisor** | **Revoke sessions**

▾ Permissions policies (1 policy applied)

Attach policies | **Add inline policy**

Policy name	Policy type	
AWSLambdaBasicExecutionRole-cdca2e44-e2b6-4af8-9a61-8191e5ca4eaa	Managed policy	✖

▸ Permissions boundary (not set)

Manage Access on AWS resources —> Roles

Add permissions to Trigger_Refresh_DWH-role-y67bgjru

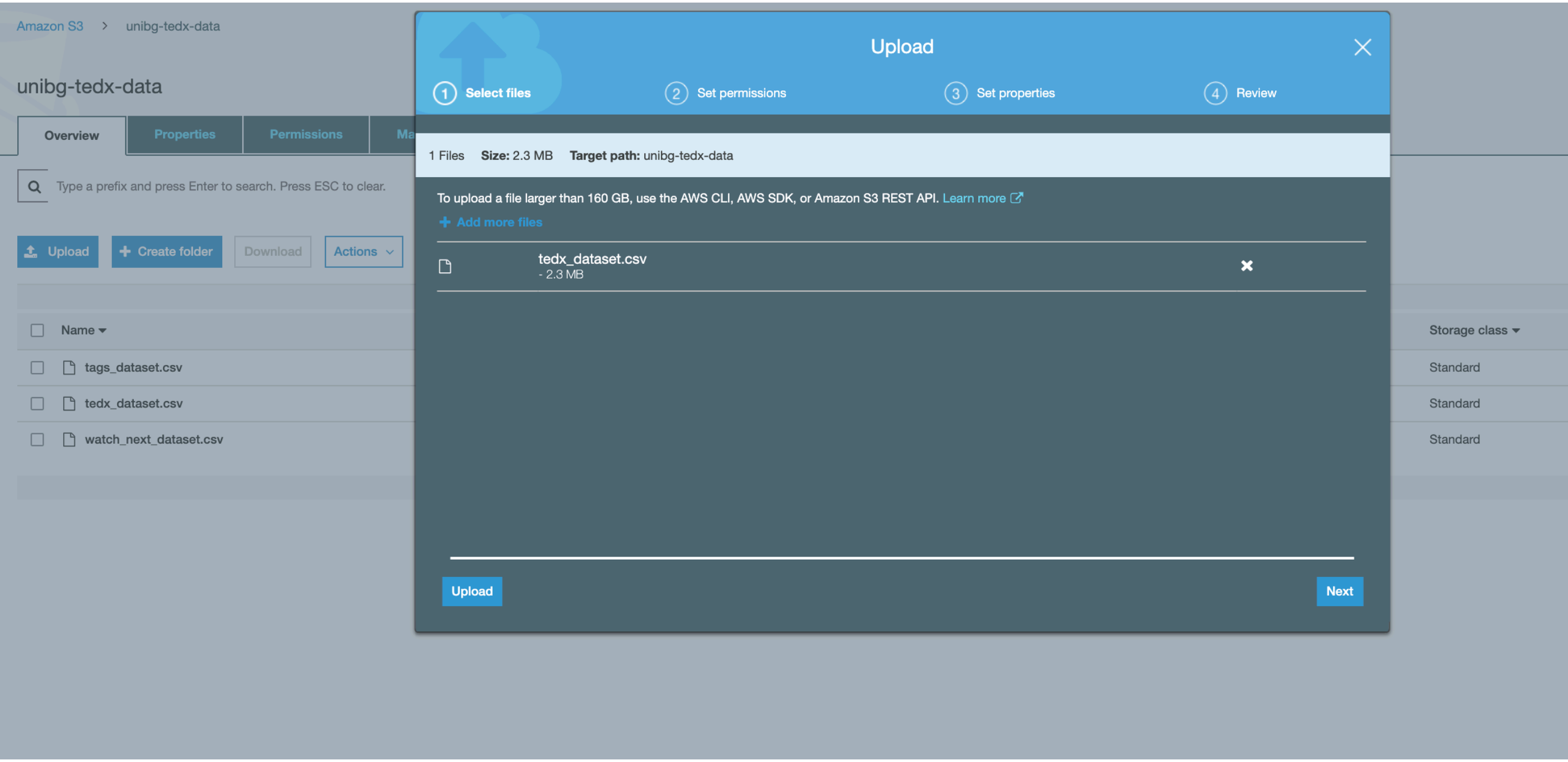
Attach Permissions

Create policy

Filter policies | glue

	Policy name
☑	AWSGlueConsoleFullAccess
☐	AWSGlueConsoleSageMakerNotebookFullAccess
☐	AWSGlueServiceNotebookRole
☐	AWSGlueServiceRole

Try to re-add a new file to the bucket



Ora tocca a voi…